UNIVERSITETET I OSLO
Fysisk Institutt

Implementasjon av Ultra Lav Spenning Kombinatoriske og Sekvensielle Kretser i 90 nm CMOS

Masteroppgave
60 studiepoeng

Mustafa Karais

01. Oktober 2009

Abstrakt

Redusere forsyningsspenningen til digitale kretser (og også analoge kretser) er noe som er veldig relevant i og med at dette resulterer ofte i en redusering av effektforbruket. Dette blir viktigere og viktigere ettersom IC kretser minker og vi får plass til flere transistorere i en og samme chip som fører til at effektforbruket øker.

I denne oppgaven vil vi se på implementasjon av logiske porter med ultra lav V_{DD}, disse portene vil vi så bruke til å lage en ALU, fire forskjellige D flip flop'er og to frekvensdelere. Vi vil da se at med de antakelsene vi gjør (V_{DD} = 180 mV, f = 250 kHz, T = -40 ^{o}C, 27 ^{o}C og 85 ^{o}C) vil disse fungere, noe som betyr at det er mulig å redusere V_{DD} og operere i subterskel område. Dette betyr også i prinsippet at vi er i stand til å lage enhver FSM og datamaskin med ultra lav spenning.

Vi vil også se at redundans er et hjelpemiddel som det går an å bruke for å oppnå bedre resultater, med tanke på å få mer stabile kretser med samme forsyningsspenning. Samme resultat får vi også ved å doble gatelengden (L = 0,2µm), men vi vil se at dobling av gatelengden er bedre enn å bruke redundans = 2. Ved å bruke redundans (av flere orden) og ved å doble gatelengden, vil vi også klare å få kretsene til å fungere med lavere V_{DD}'er, men dette betyr også at arealet av kretsen vil øke.

Forord

Masteroppgaven er utført som en del av graden *Master of Science* i mikroelektronikk, og den er utført ved Det Matematiske og Naturvitenskaplige fakultet og Institutt for Informatikk(IFI), ved Universitetet i Oslo(UiO). Masteroppgaven ble startet i september 2008 og avsluttet i oktober 2009.

Oppgaven har vært interresant å jobbe med, selvom det ble litt utfordrende til tider. Disse utfordringene klarte jeg å komme over takket være min veileder i denne oppgaven, Snorre Aunet. Jeg vil også benytte anledningen til å takke Hans Kristian Otnes Berge som hjalp meg ved hjelp av noen små diskusjoner her og der. Abdurrahim Gunaydin og Ahmed Barzanjee fortjener også en spesiell takk.

Og til slutt vil jeg takke familien min for moralsk støtte hele veien gjennom hele universitets studiene.

Lørenskog, Oktober 2009

Mustafa Karais

Innholdsfortegnelse

Figur liste

Tabell liste

Begreper og tekniske parametere:

- W = Watt eller bredden på en transistor
- M = Mega
- IC = Integrerte kretser
- V = Volt
- VLSI = Very Large Scale Integration
- V_{DD} = Forsyningsspenning
- V_t = Terskelspenning
- I_{OFF} = Strømmen i subterskel område
- L_{EFF} = Den effektivet gatelengden
- V_G = Gate spenningen
- kT/q = Termiske spenningen
- ^oC = Grader Celsius
- L = Transisrtorens kanallengde
- µm = Mikrometer
- T = Temperatur eller Periode
- GND = Elektrisk jord
- FA = Full adder
- MUX = Multiplekser
- ALU = Aritmetic Logic Unit
- CPU = Central Processing Unit
- GPU = Graphic Processing Unit
- f = Frekvens
- CLK = Klokke
- k = Kilo
- µs = Mikrosekunder
- CMOS = Complementary Metal – Oxide Semiconductor
- FSM = Finite State Machine
- Hz = Hertz

Kapittel 1

Innledning

1.1 Bakgrunn lav effekt elektronikk

"Power consumption awereness began worldwide around 1990-1992. Before that, only niche markets required low-power integrated circuits (ICs). Today, every circuit has to face the power consumption issue, for both portable devices aiming at longer battery life and high- end circuits avoiding cooling packages and reliability issues that are complex."[2]

Da de første datamaskinene ble oppfunnet tidlig i 1940 og transistoren i 1947, var ikke effektforbruk et problem [3]. Dette skjønner man fort hvis man ser på effekt forbruket til de tidligste datamaskinene. «The ENIAC» (1944) som er regnet som den første elektroniske datamaskinen hadde en effekt forbruk på 150 000 W [2]. Videre følger datamaskiner som «the Whirlwind» (1952) og «IBM 360 Model 91» (annonsert i 1964) som hendholdsvis brukte 750 000 W [2] og 1 000 000 W eller 1MW[3]. Noe av grunnen til at de tidlige datamaskinene hadde så høy effektforbruk, skyldtes i stor grad at man designet disse datamaskinene med radiorør for å øke hastigheten signifikant over datidens elektromekaniske maskiner.

Introduksjonen av transistoren i datamaskin design førte til at effektforbruket gikk drastisk ned, selvom det å minske effektforbruket ikke var transistorens hovedformål. «TX0» (1957) hadde en effektforbruk på 1000 W, og «The 12 bit PDP 8» (1965) brukte 780 W [2]. Dette viser oss at transistoren er en lav effekts apparat sammenlignet med radiorør, og at oppfinnelsen av den var det første store steget til lav effekt elektronikk. Men hvorfor skulle man starte å bruke transistorer istedenfor radiorør? Selvom transistorer var mindre i størrelse og brukte røflig 1000 ganger mindre effekt enn en radiorør, var det vel ingen som hadde behov det? Datamaskiner var jo ikke interessant for personen i gata og da spilte det vel ingen rolle at den fylte et helt rom, og kanskje flere, og brukte massevis av effekt.

I februar 1953 annonserte Sonotone det første høre apparatet som inneholdt 5 transistorer, men også noen miniatyserte radiorør [2]. I et høreapparat, eller annet portabel apparat som bruker strøm, er det viktig at batteriet varer lenge. I 1953 var det ikke så mange portable apparater som vi har nå (kalkulatorer, mobil telefoner, bærbare datamasikener osv...), derfor så gikk man ikke over til transistorer med en gang. Det var heller ikke noen effektiv måte å pakke transistorer på, før integrerte kretsen (IC) ble oppfunnet av Jack Kilby og Robert Noyce i 1958 [4].

Felt effekt transistoren kom også ut i 1958, oppfunnnet av Stanislas Teszner. Og allerede i 1962 klarte et firma med navnet RCA å pakke 16 MOS – transistorer på en brikke[2], men det var ikke før de første mikroprosserene kom at MOS – transistorer var første valget når man skulle lage en elektronisk krets. MOS – transistoren ble også brukt fordi man klarte å pakke flere transistorer på en brikke, og ikke på grunn av dens lave effekt forbruk i forhold til radiorør. Tabell 1.1.1 og tabell

1.1.2 viser henholdsvis de første Intel – mikroprossesorene, og de første Motorola mikroprossesorene.

Year	µP	Technology	Nb of MOS	Address
1971	4,004	P-MOS 8 µm	2,300	4K
1971	8,008	P-MOS 8 µm	3,500	16K
1974	8,080	N-MOS 6 µm	5,000	64K
1976	8,085	N-MOS 4 µm	6,000	64K
1978	8,086	N-MOS 3 µm	29,000	1M
1982	80,286	N-MOS 2.3 µm	130,000	16M
1985	80,386	CMOS 2 µm	275,000	4G

Tabell 1.1.1: De første Intel prosessorene[11].

Year	µP	Nb of MOS	Technology	Frequency
1974	6,800	5,000	N-MOS 6 µm	2 MHz
1979	68,000	68,000	N-MOS 4 µm	8 MHz
1984	68,020	200,000	CMOS 2 µm	16 MHz
1987	68,030	275,000	CMOS 1.3 µm	20 MHz
1989	68,040	2,000,000	CMOS 0.8 µm	25 MHz

Tabell 1.1.2: De første Motorola prossesorene[12].

Disse tabellene viser oss at det var P – MOS teknologi de første Intel prossesorene baserte seg på, men de ble tidlig skiftet ut til N – MOS teknologi for det var raskere. Dette skyldes at nMos – transistorer har bedre mobilitet enn pMos – transistorer. Vi ser også at N – MOS teknologi ikke dominerte så lenge(ca 1974- 1984), før CMOS teknologi tok over. Overgangen til CMOS teknologi skyldtes varme generering, elektronmigrasjon og pålitelighetsproblemer[2]. Derfor kan vi si at introduksjonen av CMOS – teknologi er det tredje store steget til lav effekt elektronikk, etter transistoren og IC' ene som allerede er nevnt, og CMOS er i dag den ledende teknologien. Men selv om man gikk fra N – MOS til CMOS så tidlig var det ikke for å få ned effektforbruket, og forskyningsspenningen lå fortsatt på 5V, og ingen tippet at det kom til å forandre seg med det første, som også kommer fram av Moore's lov[14].

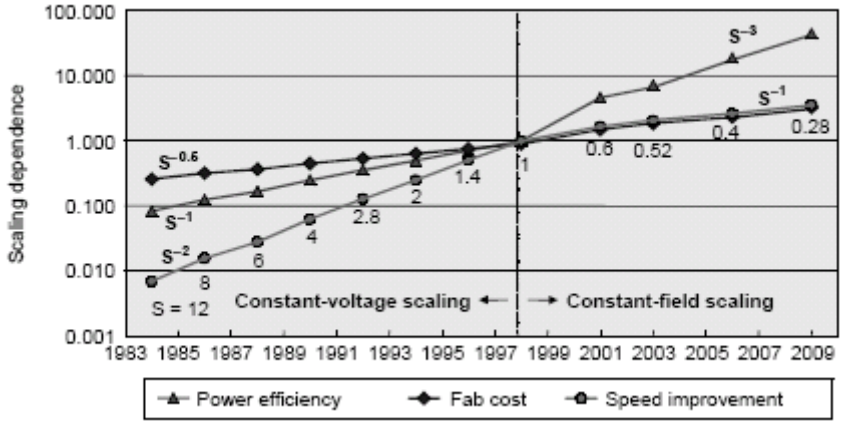
"It's a 5 volt world, and to change to 1,5 volt would mean that the whole world would have to change!" **Gordon Moore**

Fram til 1990 tallet var ikke effekt noe problem for meste parten av IC' ene, det var kun et fåtalls barbære produkter som var avhengig av lavt effektforbruk. "Bærbare" produkter på den tiden var elektroniske klokker, høreapparater og pacemakere. Utviklingen av disse hatt en stor

betydning i lav effekt design, men for det meste er det ikke dokumenter tilgjengelig så vi kan ikke se på utviklingen av dem. En ting vi har dokumenter på er utviklingen av elektroniske klokker.

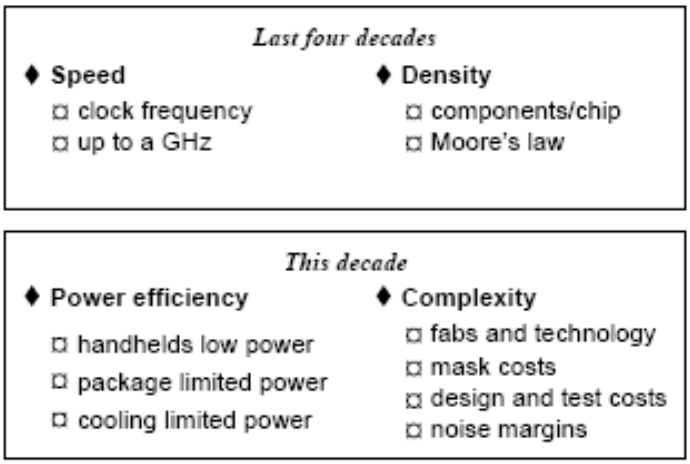
''The Horological Electronics Center'' eller ''Center Electronique Horloger''(CEH) utviklet den første elektroniske klokken, en sveitsisk kvarts klokke kalt Beta[15,16]. Den ble designet i bipolar teknologi, siden bipolar teknologi var mer pålitelig på den tiden, og kom ut i 1963. I 1964 tok CEH i bruk CMOS – teknologi, 20 år før de store halvleder firmaene(Intel og Motorola tabell 1.1.1 og tabell 1.1.2).

Figur 1.1.1 viser skalering av kretser og virkninger det har på forskjellige parametere. Som vi ser av figuren har trendene endret seg de siste årene. Fram til ca. 1997 var det konstant spenning skalering, som rett og slett betyr at forsyningsspenningen ble holdt fast ved 5V selv om transistorene ble mindre, etter 1997 gikk man over til konstant felt skalering, som betyr at forsyningsspenningen ble redusert i takt med transistorenes kanal lengde [44].



Figur 1.1.1: Skalering av forskjellige parametere [44]

Grafen viser endring av tre viktige parametere: hastighet økning, fabrikasjonskostnader og ''effekt effektivitet'' (hvor mye hastighet vi får for hver W) [44]. Som vi ser så var det mest hastighetsøkning før 1997, fabrikasjonskostnadene økte litt saktere mens effektiviteten var det som økte tregest. Etter 1997 er det effektiviteten som øker raskest, mens hastigheten og fabrikasjonskostnadene øker like fort. Dette viser oss at det er effektiviteten til kretsene som har blitt den viktigste parameteren denne dekaden. Figur 1.1.2 viser oss skaleringstrender før og i nærmeste fremtid.



Figur 1.1.2: Skaleringstrender før og nå [44]

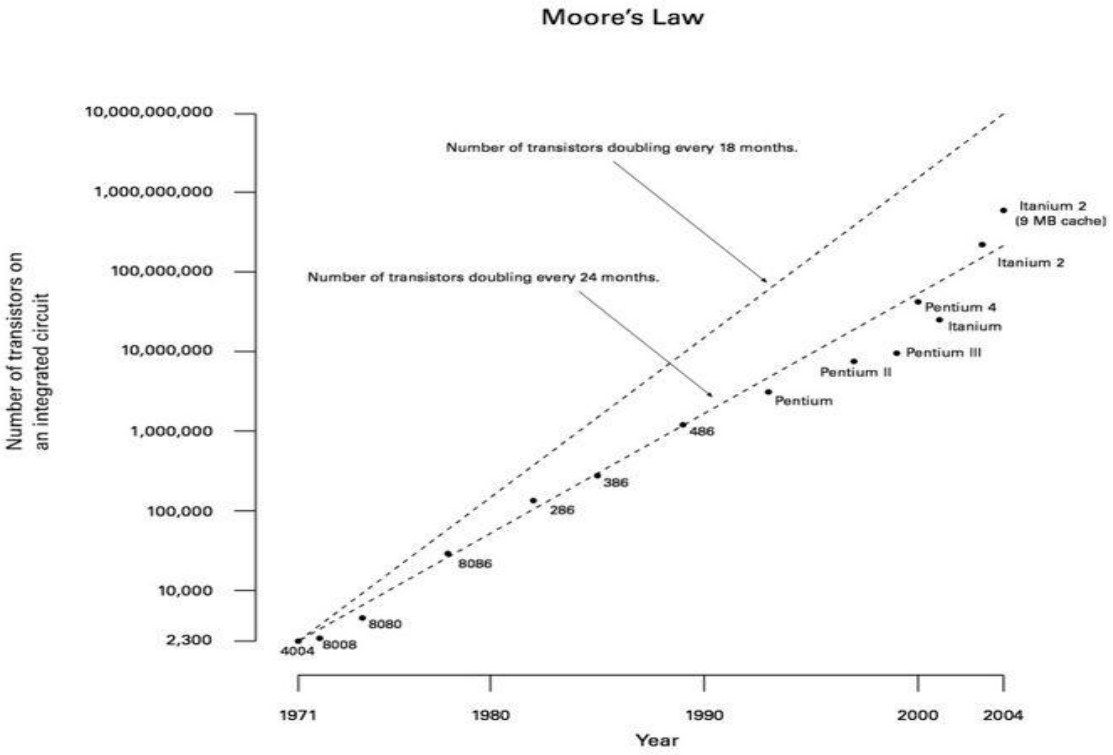
Som vi ser var det hastighet og tetthet som var de viktigste parameterene før, mens det nå er effektiviteten og kompleksiteten som er de viktigste parameterene. Dette viser oss også hvor viktig det er å kunne designe kretser som bruker lav effekt i fremtiden.

1.2 Moore's lov

Tabell 1.1.1 og 1.1.2 viser oss hvor mye antallet til transistorer i en mikroprossesor har økt i takt med tiden. Dette er blitt forklart av Gordon Moore i det som blir kalt Moore's lov.

"The complexity for minimum component cost has increased at a rate of roughly a factor two per year ... Certainly over the short term this rate can be expected to continue, if not increase. Over the longer term, the rate of increase is a bit more uncertain although there is no reason to believe it will not remain nearly constant for at least 10 years. That means by 1975 the number of components per integrated circuit for minimum cost will be 65,000. I believe such a large circuit can be built on a single wafer." [34]

Moore's lov sier oss at transistorer i en IC blir doblet hver 24. måned og **ikke** hver 18. måned som det ofte blir oppgitt. Opprinnelig gjaldt loven komponenter i en IC, altså både transistorer og motstander, men som vi ser i figur 1.2.1 gjelder den veldig bra for antall transistorer i en prossesor.



Figur 1.2.1: Reel vekst i transistorantall sammenlignet med Moore's lov. [35]

På grunn av denne dramatiske økningen i antallet transistorer på et areal, vil effektforbruket per areal øke dramatisk [1]. Intel har kommet fram til at hvis utviklingen fortsetter slik den har gjort til nå, vil effekt tettheten til en mikroprossesor overgå effekt tettheten til solens overflate [36]. Dette gjør at kontrollering av effektforbruket kommer til å bli det viktigste i fremtidig VLSI design [37].

Å redusere forsyningsspenningen er det mest direkte og dramatiske enkeltgrepet som kan

gjøres for å redusere effektforbruket i elektroniske kretser [38,39], og det å operere CMOS kretser i subterskel området er regnet som den mest effekt effektive løsningen for lav til medium prestasjons applikasjoner [40]. Dette kan vi se enkelt ved å ta en titt på formelen for dynamisk effektforbruk i CMOS teknologi [47].

$$P = NfCV_{DD}^2 \qquad (1.2.1)$$

Som vi ser av formelen er P avhengig av kvadratet til V_{DD}. Dette beviser hvor viktig det er å holde V_{DD} lav for å få en lav effektforbruk. For øvrig så er N antall transistorer, f er frekvens og C er gjennomsnittlig gatekapasitans.

1.3 Lav effekt design teknikker

Som nevnt tidligere så er det reduseringen av forsyningsspenningen som er det mest direkte og dramatiske som kan gjøres for å redusere effektforbruket. Andre teknikker for å redusere effektforbruket er for eksempel: "Pipelining" og pararellisering (for å redusere frekvensen og V_{DD}) [41,42], asynkron eller "self-timed" arkitektur (for å fjerne klokken som er kjent for å være en stor effektforbruker) [2], "gatede klokker" og aktivitetsredusering (igjen for å redusere effektforbruket til klokken) [2] og subterskels operasjon.

1.3.1 Subterskel operasjon

Subterskel operasjonsområde er definert som det området hvor forsyningsspenningen er mindre enn absolutt verdien på terskelspenningen til transistoren, $V_{DD} < |V_t|$ [1]. Ved å la V_{DD} være lavere enn terskelspenningen sørger vi for at transistorkanalen aldri går helt fra Source til Drain, som gjør at vi transistoren blir operert i svak eller moderat inversjon når den er *på*.

Tradisjonelt sett er subterskel område for en transistor regnet som et område hvor transistoren er av, og derfor har det ikke blitt vist interesse til det området. Strømmen i subterskel området kan defineres som [2]:

$$I_{OFF} \approx a\frac{1}{L_{EFF}}\exp\left(\frac{q(V_G - V_T)}{kT}\right)$$

Fordelen med å operere en transistor i subterskel område er at vi får mindre inngangs kapasitanser som kombinert med lav forsyningsspenning fører til at transistorer i subterskel område bruker mindre effekt enn transistorer i sterk inversjonsområde [1]. Men siden det er lekasje strømmen i subterskel området som blir brukt under utregninger, har disse kretsene begrensninger når det gjelder høyfrekvente operasjoner og de når sin grense ved noen MHz for moderne prosseser [43].

1.4 Kapittel oversikt

Innholdet i oppgaven er delt opp i forskjellige kapitler som vil gi oss et innblikk i design av kretser med ultra lav V_{DD}. Oppgaven starter med en innledning til lav effekt elektronikk, fulgt opp av implementering og simulering av logiske porter som er de "minste byggeblokkene" i digital elektronikk. Videre blir disse logiske portene brukt til å lage tre forskjellige kretser som til sammen blir til en ALU i kapittel 4. I kapittel 5 finner vi fire forskjellige flip floper som vi i kapittel 6 har brukt til å lage en klokkedeler.

Alt er realisert i Cadence 90 nm teknologi med standard terskelspenning transistorer.

- Kapittel 1: Innledning
- Kapittel 2: Simuleringsresultater for logiske porter
- Kapittel 3: Simuleringsresultater for noen enkle kretser (FA, Dekoder, MUX)
- Kapittel 4: Simuleringsresultater for en enkel ALU
- Kapittel 5: Simuleringsresultater for fire flip flop'er
- Kapittel 6: Simuleringsresultater for to klokkedelere

Kapittel 2

Simuleringsresultater for logiske porter

2.1 Logiske porter

En logisk port er en elektronisk krets som opererer med en eller flere inngangssignaler for å produsere et utgangssignal. Dette utgangssignalet er representert med 2 verdier i digital elektronikk, 0 eller 1 hvor de representerer *lave* og *høye* spenninger respektivt. For eksempel så kan et system definere 0 – nivået som 0 V, mens 1 – nivået som 5 V (eller V_{DD}). I denne oppgaven er 0 – nivået valgt som 0 V og 1 – nivået som V_{DD} med en akseptabel feilmargin på 25% på hvert av nivåene. Dette betyr at hvis et utgangsssignal har en verdi som er 25% eller mindre enn V_{DD} så er det en godkjent 0'er, og hvis signalet har 75% eller mer av verdien til V_{DD} er det regnet som en 1'er. Hvis signalet havner mellom 25% og 75%, er det regnet som ikke definert.

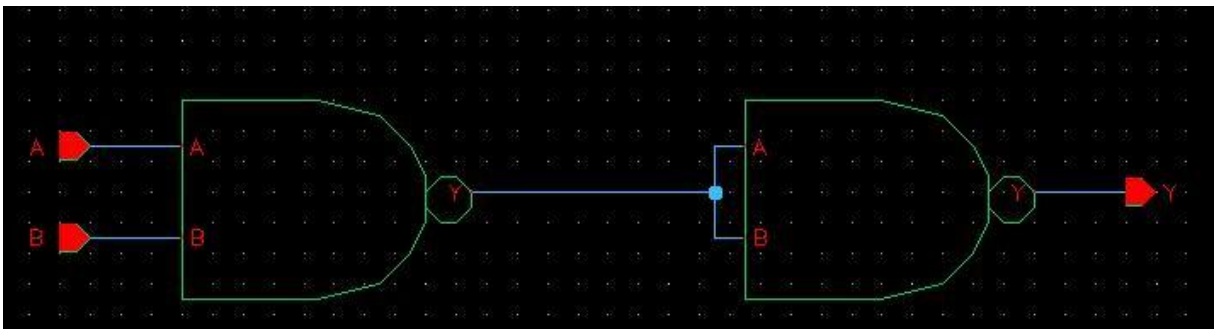
2.1.1 AND2

Sannhetstabellen for en AND2 port kan man se i tabell 2.1.1.1, og den viser virke måten til porten. Tabellen viser oss at utgangen av porten er *høy* hvis og bare hvis begge innganene har *høye* verdier, og at hvis en av inngangene er *lav* vil vi få en 0'er på utgangen. Dette betyr at en AND2 port effektivt vil gi oss minimumsverdien til inngangene.

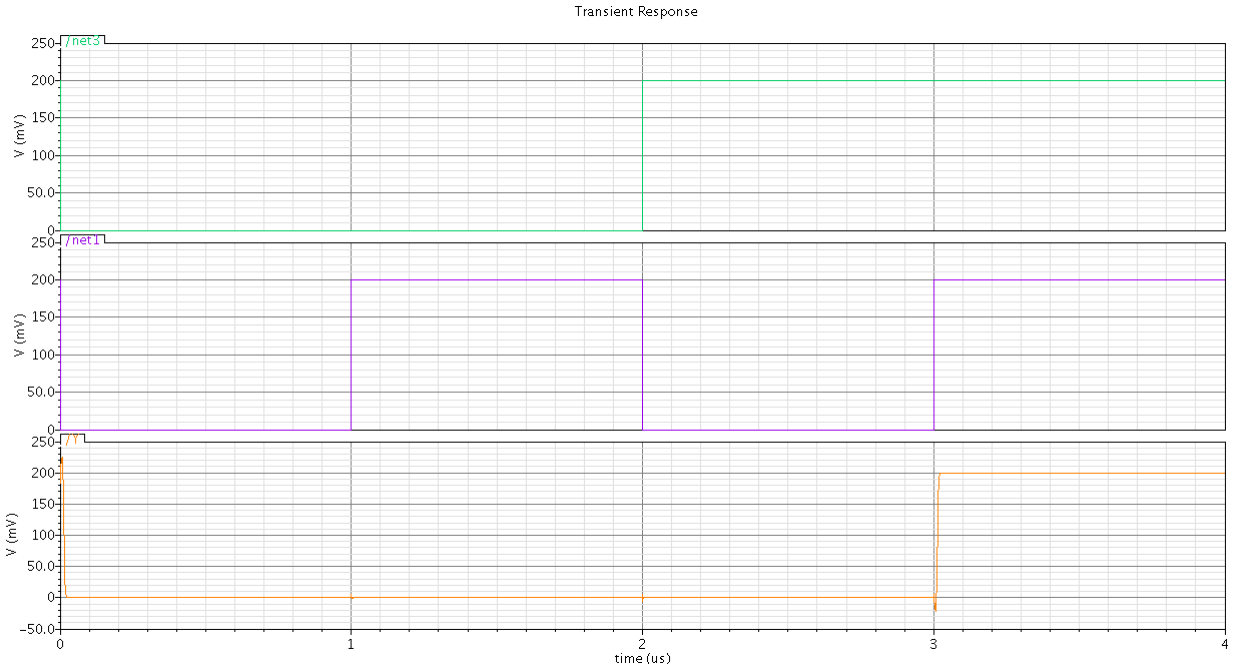
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Tabell 2.1.1.1: Sannhetstabellen for en AND2 port

En AND2 port realisert ved hjelp av NAND2 porter (kapittel 2.1.4) vil se ut som på figur 2.1.1.1. Vi kan se at denne konfigurasjonen stemmer ved å se på simuleringsresultatet på figur 2.1.1.2 og sammenligne med tabell 2.1.1.1.



Figur 2.1.1.1: AND2 port satt sammen av 2 NAND2 porter



Figur 2.1.1.2: Simuleringsresultater av en AND2 port realisert med NAND2 porter ved 27 ^{o}C og V_{dd} = 200 mV (turkis innganger, blå utgang)

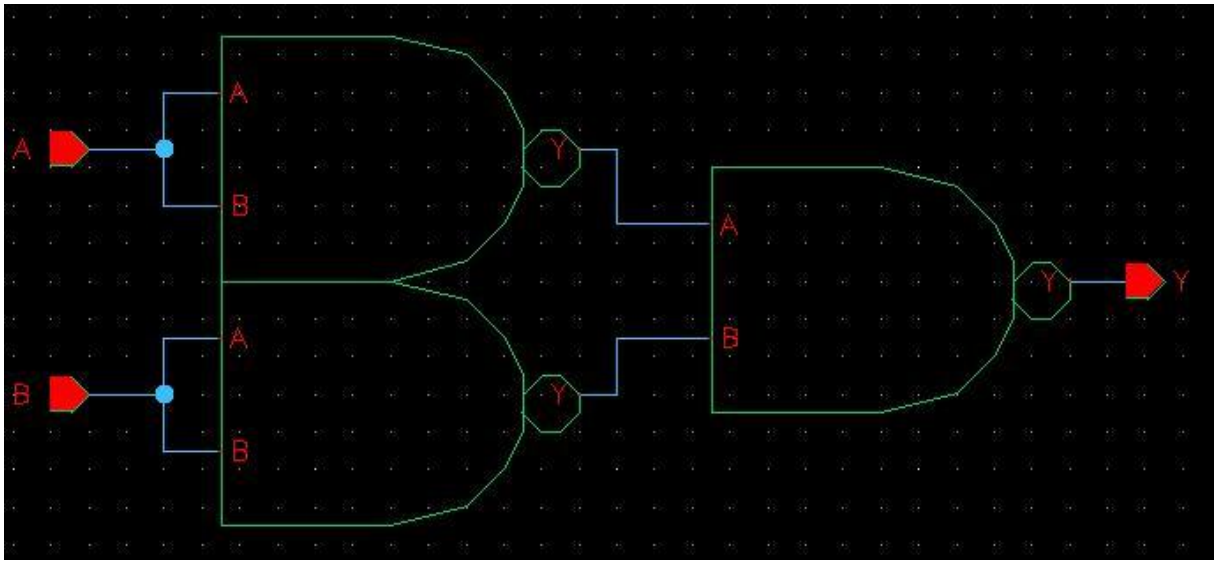
2.1.2 OR2

Sannhetstabellen for en OR2 port kan man se i tabell 2.1.2.1, og den viser virke måten til porten. Tabellen viser oss at utgangen av porten er *lav* hvis og bare hvis begge innganene har *lave* verdier, og at hvis en av inngangene er *høy* vil vi få en 1'er på utgangen. Dette betyr at en OR2 port effektivt vil gi oss maksimumsverdien til inngangene.

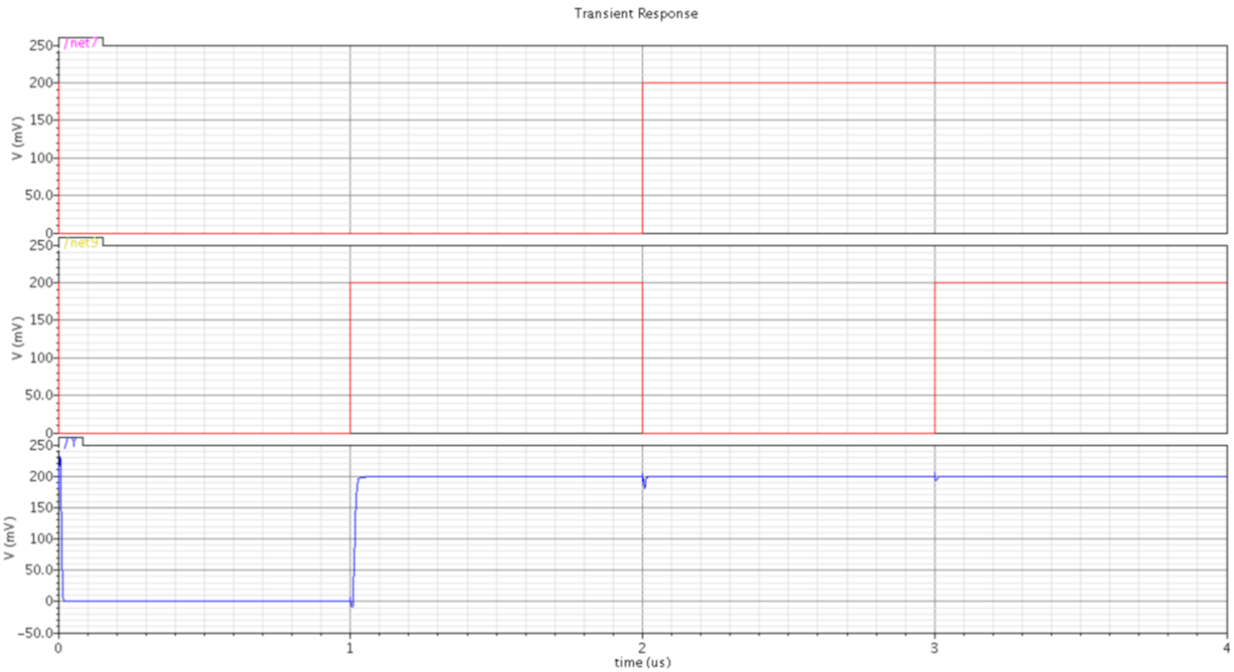
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Tabell 2.1.2.1: Sannhetstabellen for en OR2 port

En OR2 port realisert ved hjelp av NAND2 porter (kapittel 2.1.4) vil se ut som på figur 2.1.2.1. Vi kan se at denne konfigurasjonen stemmer ved å se på simuleringsresultatet på figur 2.1.2.2 og sammenligne med tabell 2.1.2.1.



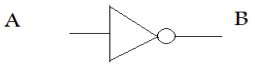
Figur 2.1.2.1: OR2 port satt sammen av 2 NAND2 porter



Figur 2.1.2.2: Simuleringsresultater av en OR2 port realisert med NAND2 porter ved 27 ^{o}C og V_{dd} = 200 mV (rød innganger, blå utgang)

2.1.3 INVERTER

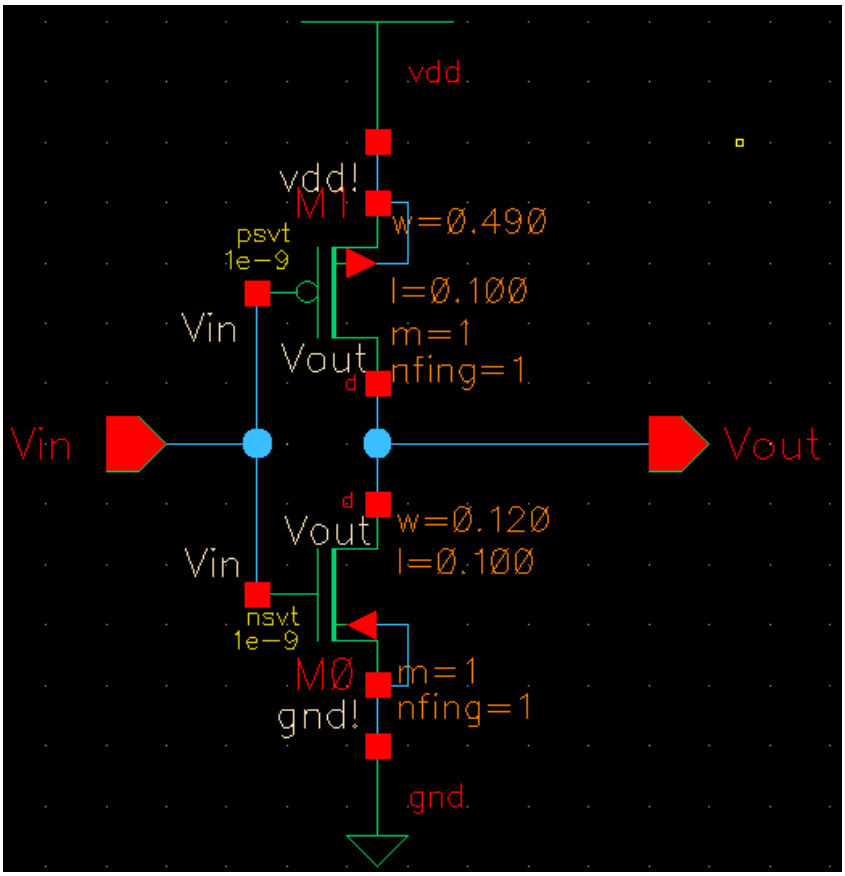
En inverter er en logisk port som inverterer verdien den har på inngangen, slik navnet tilsier. Dette er også den enkleste porten å realisere og simuleringer viser også at den er den mest robuste porten. Dens robusthet kommer av at den er bygget opp av få transistorer, i dette tilfelle av en pMos og en nMos i serie (figur 2.1.3.2), i motsetning til andre porter som er bygget opp av flere (NAND2 2nMos og 2 pMos). Sannhetstabellen til en inverter er vist i tabell 2.1.3.1. Symbolet til en inverter er vist i figur 2.1.3.1.



Figur 2.1.3.1: Symbolet til en inverter

A	A'
0	1
1	0

Tabell 2.1.3.1: Sannhetstabellen for en Inverter



Figur 2.1.3.2: Skjematikk for en inverter

For å finne det rette forholdet mellom pMos- og nMos transistoren, setter man inngangen til $V_{DD}/2$ og justerer på bredden til pMos transistoren. Bredden til nMos transistoren er på sitt minimum for prosessen. Når man får en 'matching' av transistorene, vil utgangen være lik inngangen, som er $V_{DD}/2$. Matching betyr at 2 transistorer har like parametere, som f.eks V_t. Matching er veldig viktig spesielt for analoge kretser (minnekretser blir også regnet som analoge)[5], men også for digitale kretser. Spredningen av V_t fører til at man får ulike forsinkelser i identiske logiske kretser, dette er ugunstig spesielt for høyhastighets kretser hvor man er avhengig av at signalene ikke opplever for store forsinkelser[5]. 'Matching' er i denne oppgaven definert som symmetri om svitsjepunktet. Tabellene 2.1.3.2 og 2.1.3.3 viser størrelser på pMos og nMos transistorer ved matching ved ulike forskyningsspenninger og transistorlengder.

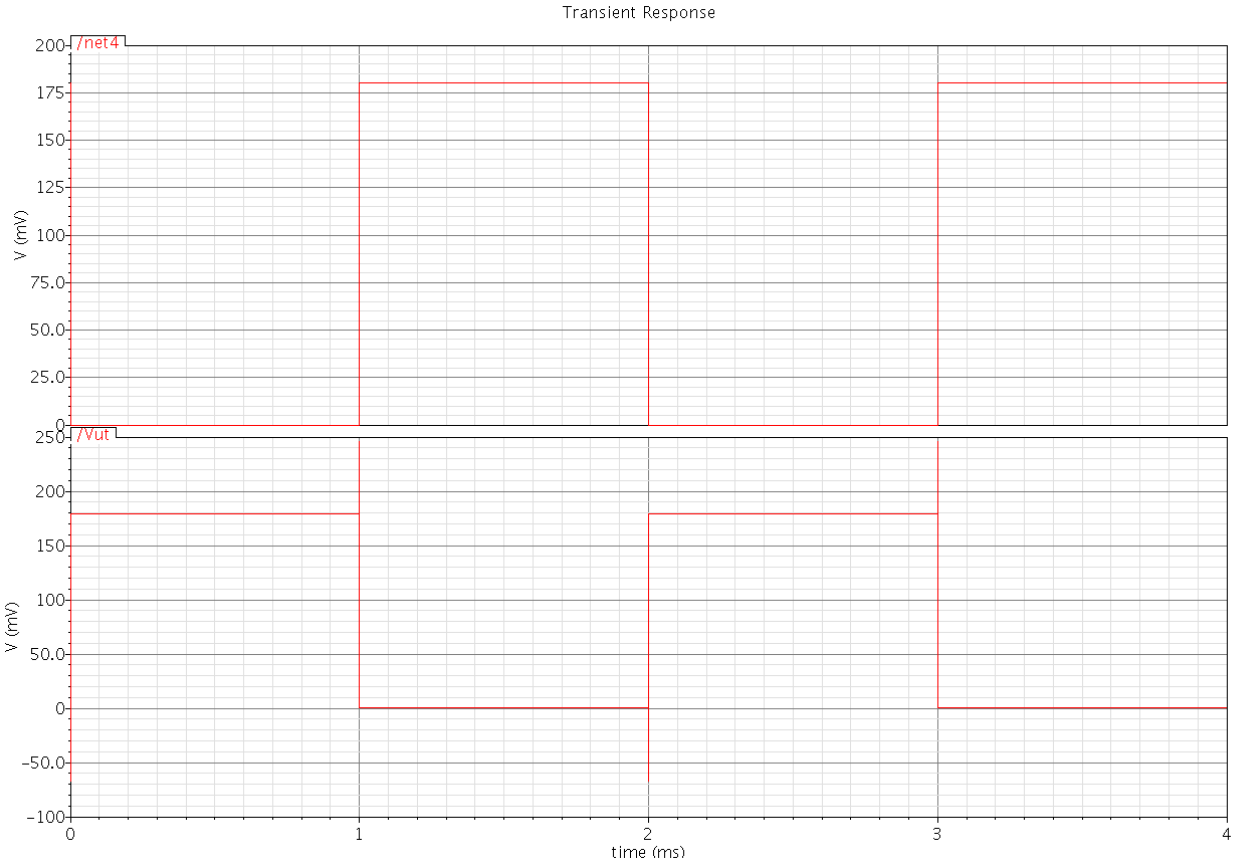
Vdd	nMos	pMos
180mV	0.12µm	0.49µm
1V	0.12µm	0.44µm

Tabell 2.1.3.2: W på transistorene ved matching, L= 0,1µm T= 27 °C

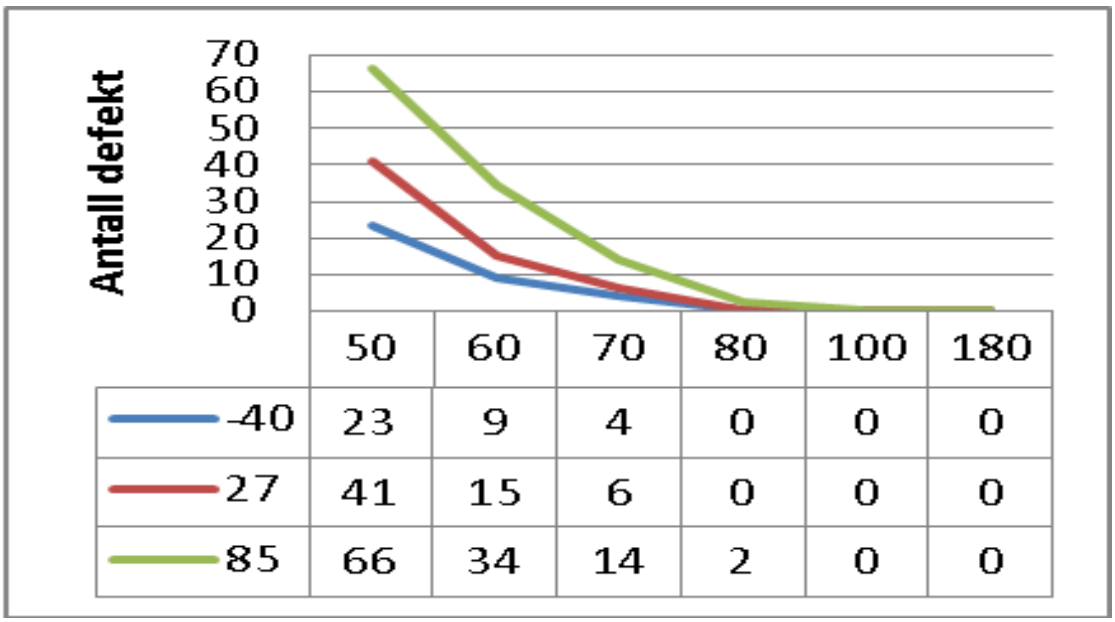
Vdd	nMos	pMos
180mV	0.12µm	0.77µm

Tabell 2.1.3.3: W på transistorene ved matching, L=0,2µm T= 27 °C

Figur 2.1.3.3 viser simulering av inverteren på figur 2.1.3.2. Vi ser at inngangen og utgangen er det motsatte av hverandre slik man forventer ut i fra sannhetstabellen.



Figur 2.1.3.3: Simulering av en inverter ved 27 ^{o}C og V_{dd} = 180 mV



	50	60	70	80	100	180
-40	23	9	4	0	0	0
27	41	15	6	0	0	0
85	66	34	14	2	0	0

Figur 2.1.3.4: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner av kretsen i figur 2.1.3.2, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}.

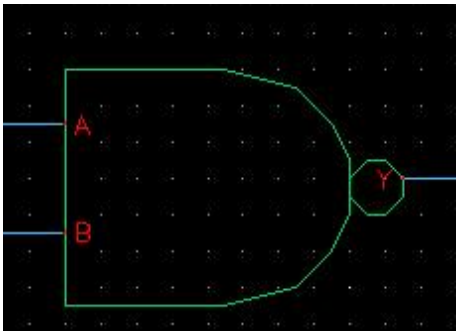
Vi ser ut i fra grafen og tabellen at inverteren funker 100% helt nede ved en V_{DD} = 100 mV, selvom kretsen er matchet for en V_{DD} = 180 mV.

2.1.4 NAND2

NAND2 porten er den universielle logiske porten, siden alle Boolske funskjoner kan bli implementert ved hjelp av den[6]. For å vise at dette er sant, så må man vise at det går an å lage AND, OR og INVERTER ved hjelp av en NAND port. Dette er bevist i henholdsvis kapittel 2.1.1 og 2.1.2. I figur 2.1.1.1 ser vi at den siste NAND2 porten er koblet som en inverter, og dermed har vi vist at NAND porten kan brukes til alt. Grunnen til denne konversjonen, og at NAND porter blir fremtrukket framfor AND og OR porter, skyldes at en NAND2 port har maksimalt to transistorer i serie og at de som oftest har et mer kompakt utlegg. Grunnen til at vi ikke ønsker mer enn to – tre transistorer i serie er for å ha en rask krets [45]. En NAND port er en invertert AND port (Not AND) slik tabell 2.1.4.1 også viser. En NAND2 port består av to pMos transistorer i pararell opp mot V_{DD} og to nMos transistorer i serie ned til GND(figur 2.1.4.2), og i figur 2.1.4.1 kan vi se symbolet til en NAND2 port.

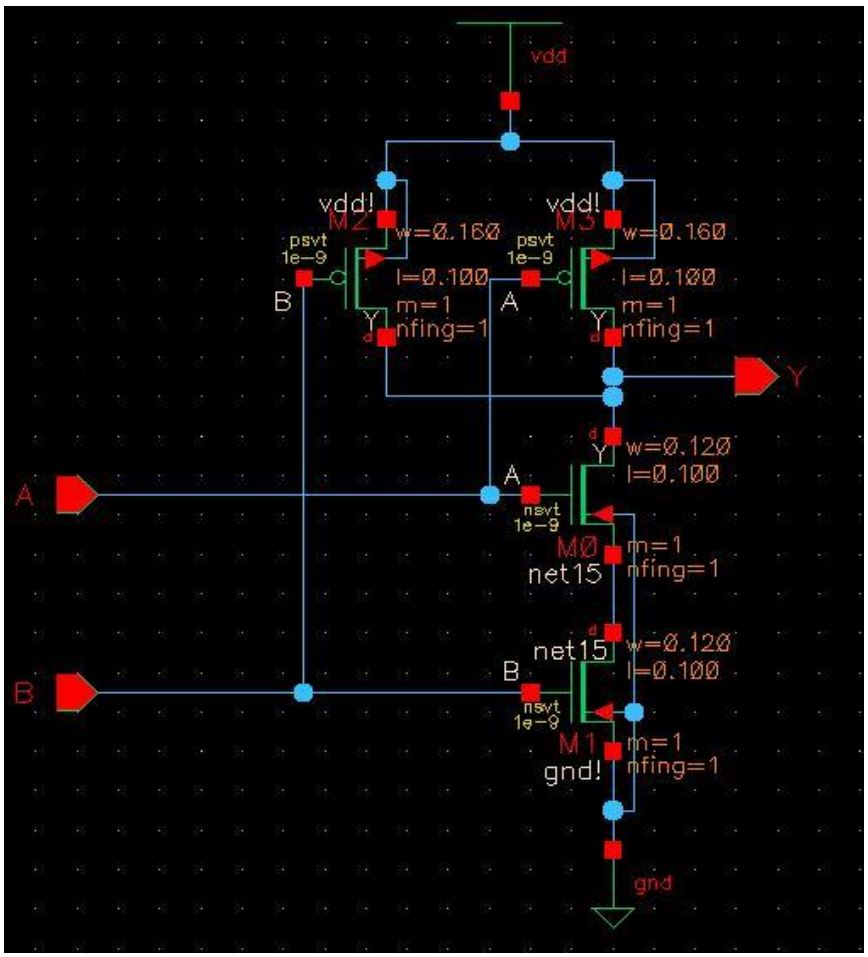
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

Tabell 2.1.4.1: Sannhetstabellen for en NAND2 port



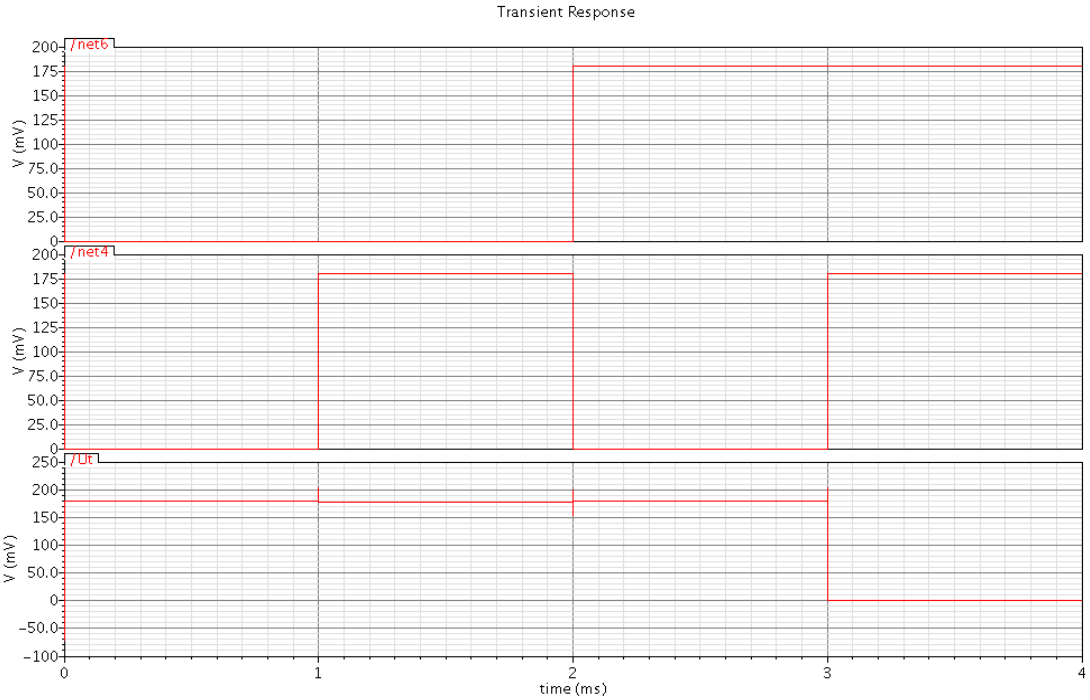
Figur 2.1.4.1: Symbolet til en NAND2 port

NAND2 porten som er vist i figuren på neste side er dimensjonert slik at det er matching mellom nMos- og pMos transistorene ved 27 ^{o}C og ved V_{DD} = 180 mV. Simuleringen til kretsen i figuren er gitt i figur 2.1.4.3. Vi ser at porten fungerer slik vi vil at den skal fungere og at det er samsvar med sannhetstabellen.



Figur 2.1.4.2: Skjematikk for en NAND2 port

Tabellene 2.1.4.2 og 2.1.4.3 viser størrelses forholdet mellom nMos transistorene og pMos transistorene ved matching med forskjellige verdier for V_{DD} og L.



Figur 2.1.4.3: Simulering av en NAND2 port ved 27 ^{o}C og V_{DD} = 180 mV (de 2 øverste grafene er inngangene og den nederste er utgangen)

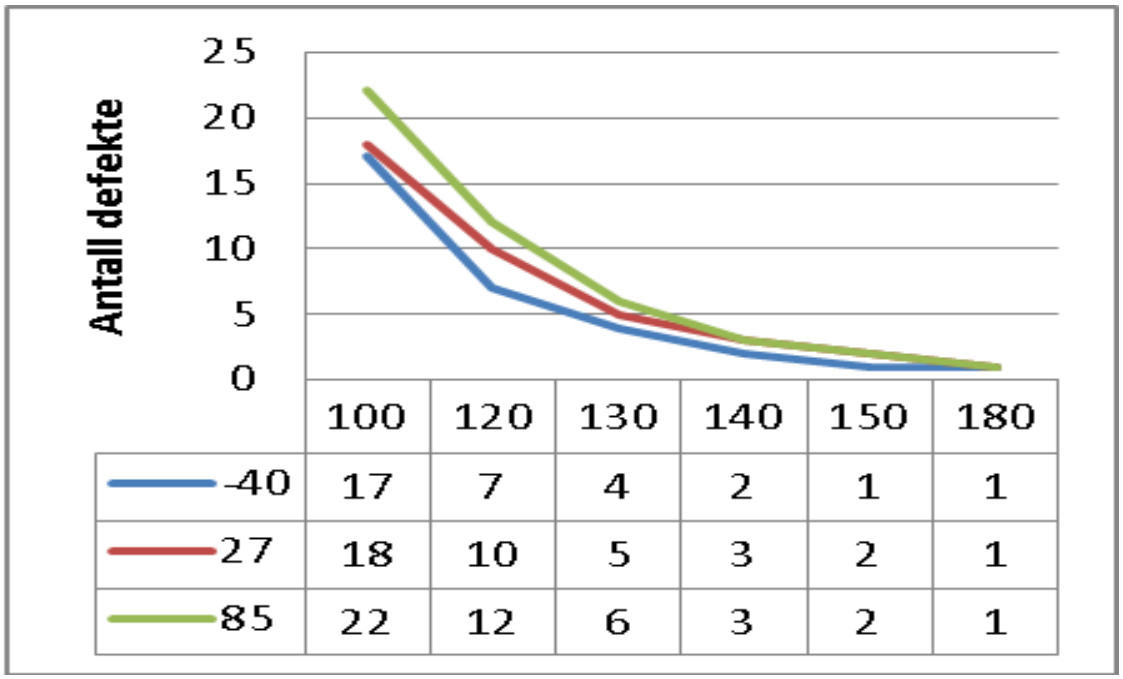
Vdd	nMos	pMos
180mV	0,12µm	0,16µm
1V	0,15µm	0,12µm

Tabell 2.1.4.2: W på transistorene ved matching, L= 0,1µm T= 27 ^oC

Vdd	nMos	pMos
180mV	0,12µm	0,205µm

Tabell 2.1.4.3: W på transistorene ved matching, L= 0,2µm T= 27 ^oC

Vi ser i tabell 2.1.4.2 at når V_{DD} = 1V må nMos transistoren bli større, mens pMos transistoren må ned i minimumsbredde for å oppnå matching.



Figur 2.1.4.4: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner av kretsen i figur 2.1.4.2, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}.

Figuren viser oss at selvom vi åpner en matching ved en V_{DD} = 180mV, så virker ikke 100 % av kretsene.

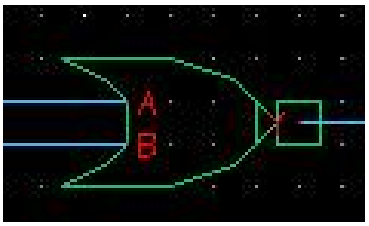
2.1.5 NOR2

En NOR port er en invertert OR port (not OR). Sannhetstabellen for en NOR2 port ser vi i tabell 2.1.5.1. Videre så stemmer det vi har sagt om NAND porter for NOR porter også. NOR porter kan brukes til å implementere hvilken som helst annen logisk port. For å vise at dette stemmer må vi vise at det går an å lage en INVERTER, en OR og en AND port ved hjelp av NOR porten. Vi ser ut i fra sannhetstabellen at hvis vi kobler sammen inngangene får vi en invertering av dem og følgelig kan det kalles en INVERTER. For å få en OR port kobler vi bare denne inverteren på utgangen til en NOR port, og for å få en AND port kobler vi inverteren på begge inngangene av en NOR port.

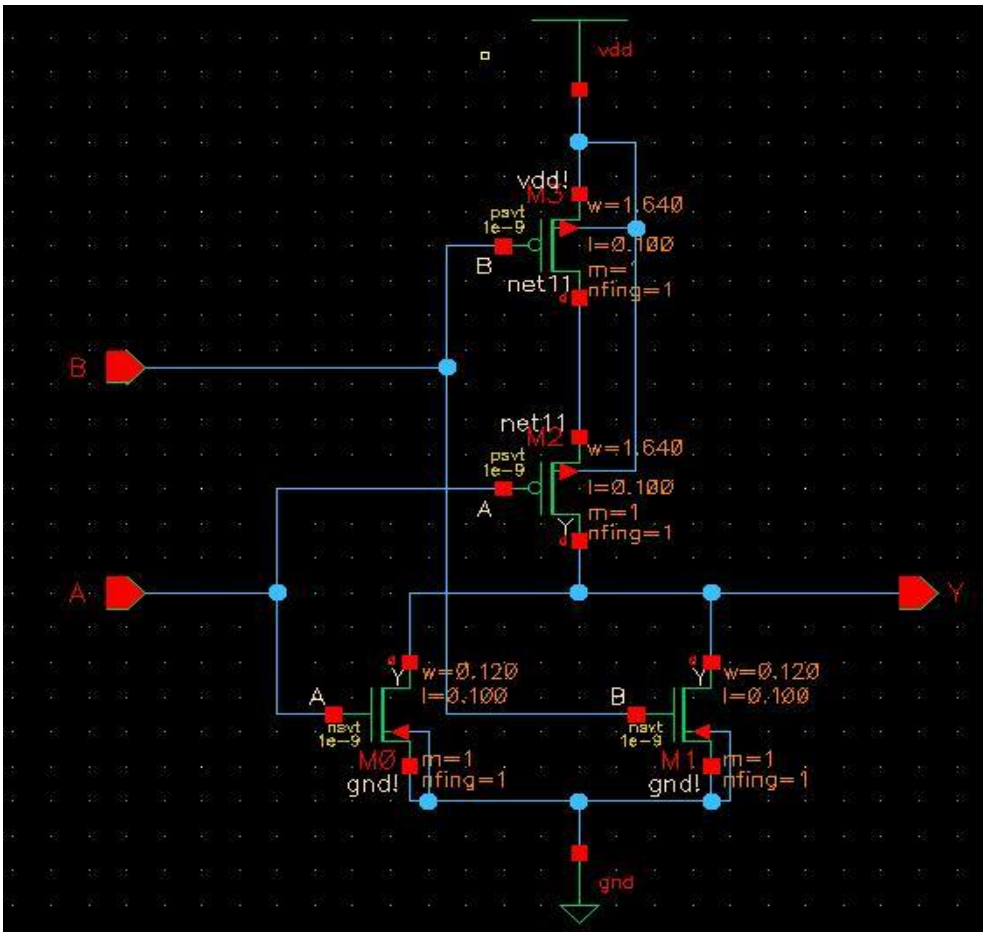
A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

Tabell 2.1.5.1: Sannhetstabellen for en NOR2 port

I figur 2.1.5.1 ser vi symbolet til en NOR2 port og i neste figur ser vi skjematikk av en NOR2 port(figur 2.1.5.2)

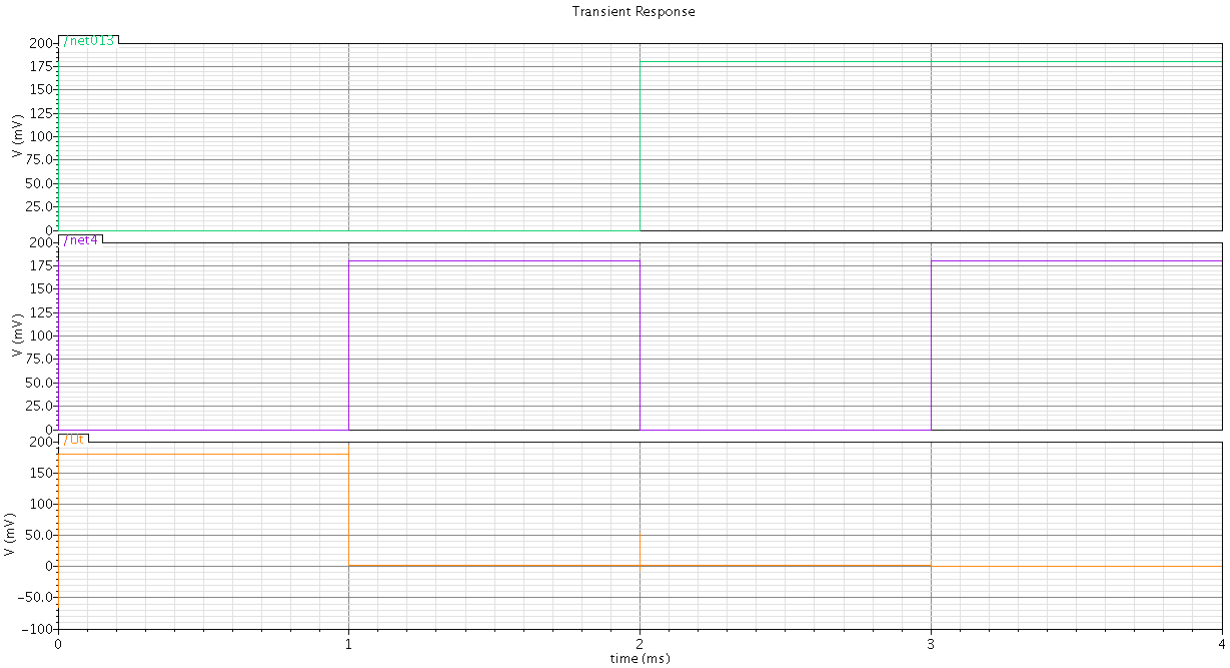


Figur 2.1.5.1: Symbolet til en NOR2 port



Figur 2.1.5.2: Skjematikk for en NOR2 port

Kretsen i figuren ovenfor er dimensjonert slik at det er en matching mellom pMos - transistorene og nMos – transistorene ved T = 27 ^{o}C og ved V_{DD} = 180 mV. Simuleringen av denne kretsen er gitt i neste figur.



Figur 2.1.5.3: Simulering av en NOR2 port ved 27 ^{o}C og V_{DD} = 180 mV (de 2 øverste grafene er inngangene og den nederste er utgangen)

Vi ser at simuleringen samsvarer med sannhetstabellen til NOR2 porten gitt i tabell 2.1.5.1. Dette viser at krets konstruksjonen i figur 2.1.5.2 stemmer.

Tabellene 2.1.5.2 og 2.1.5.3 viser størrelses forholdet mellom nMos transistorene og pMos transistorene ved matching med forskjellige verdier for V_{DD} og L.

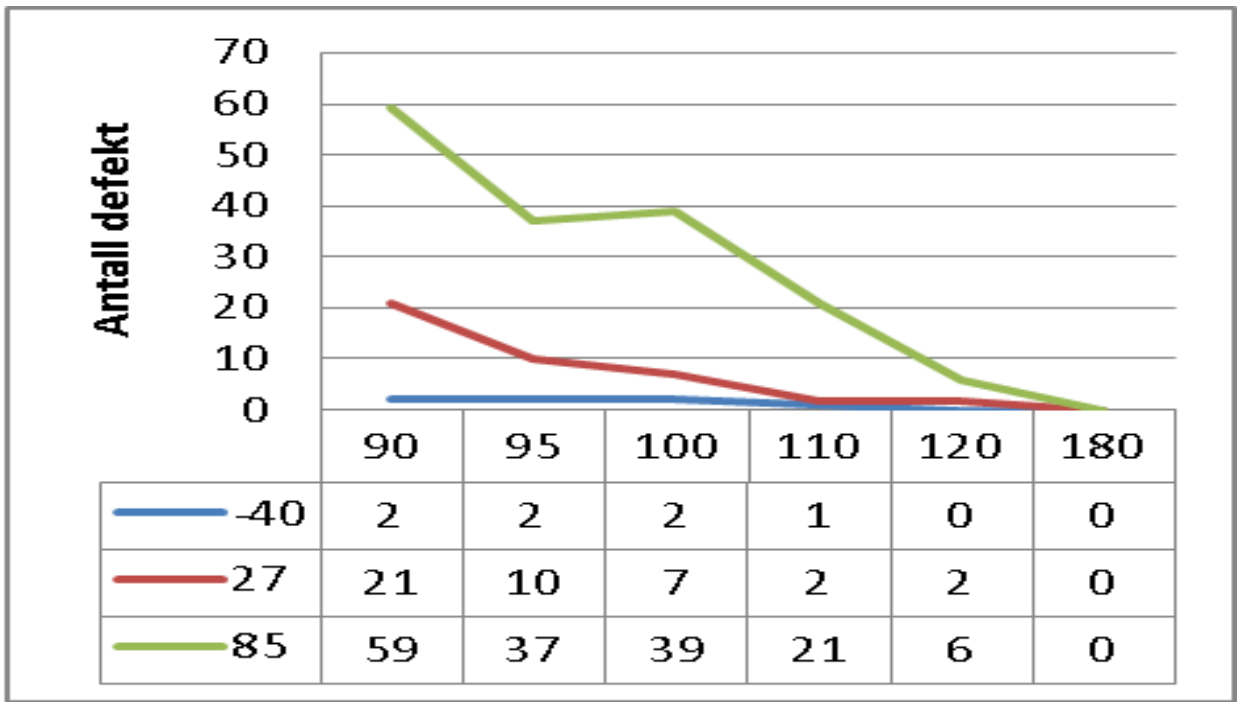
Vdd	nMos	pMos
180mV	0,12µm	1,64µm
1V	0,12µm	1,775µm

Tabell 2.1.5.2: W på transistorene ved matching, L= 0,1µm T= 27 ^{o}C

Vdd	nMos	pMos
180mV	0,12µm	2,595µm

Tabell 2.1.5.3: W på transistorene ved matching, L= 0,2µm T= 27 ^{o}C

Tabellene over viser at pMos – transistorene må være ganske mye større enn nMos – transistorene for å åpnå matching. 21,625 ganger større enn nMos må pMos være når lengden på transistorene er 0,2µm.



	90	95	100	110	120	180
-40	2	2	2	1	0	0
27	21	10	7	2	2	0
85	59	37	39	21	6	0

Figur 2.1.5.4: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner av kretsen i figur 2.1.5.2, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}.

Vi ser at kretsen fungerer perfekt for en V_{DD} = 180 mV. Den virker også ved -40 ^{o}C med en V_{DD} = 120 mV, men siden den ikke fungerer 100 % ved de andre temperaturene så er kan vi ikke si at den fungerer med 100 % ”yield” ved 120 mV.

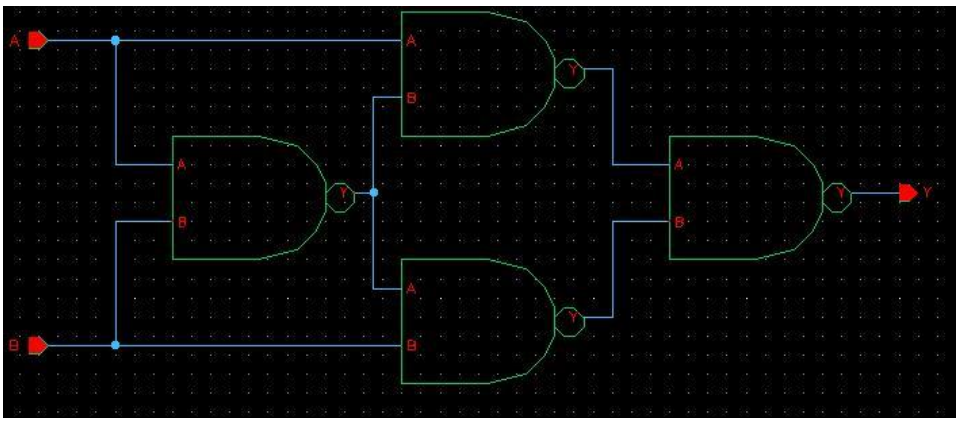
2.1.6 XOR

En XOR port er en port som gir *høy* på utgangen hvis og bare hvis inngangene har forskjellige verdier, slik det også kommer fram av sannhetstabellen i tabell 2.1.6.1. Dette kommer også fram av navnet på porten egentlig, eksklusiv (X) OR, som betyr at det er kun en inngang som kan være høy for at utgangen skal være høy.

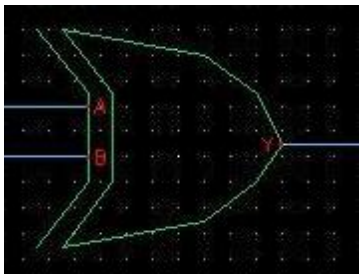
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Tabell 2.1.6.1: Sannhetstabellen for en XOR2 port

XOR porter kan lages ved hjelp av fire NAND2 porter [7]. Figur 2.1.6.1 ser vi skjematikken til en XOR port som er realisert ved hjelp av fire NAND porter, og i figuren etter det ser vi symbolet til XOR porten.

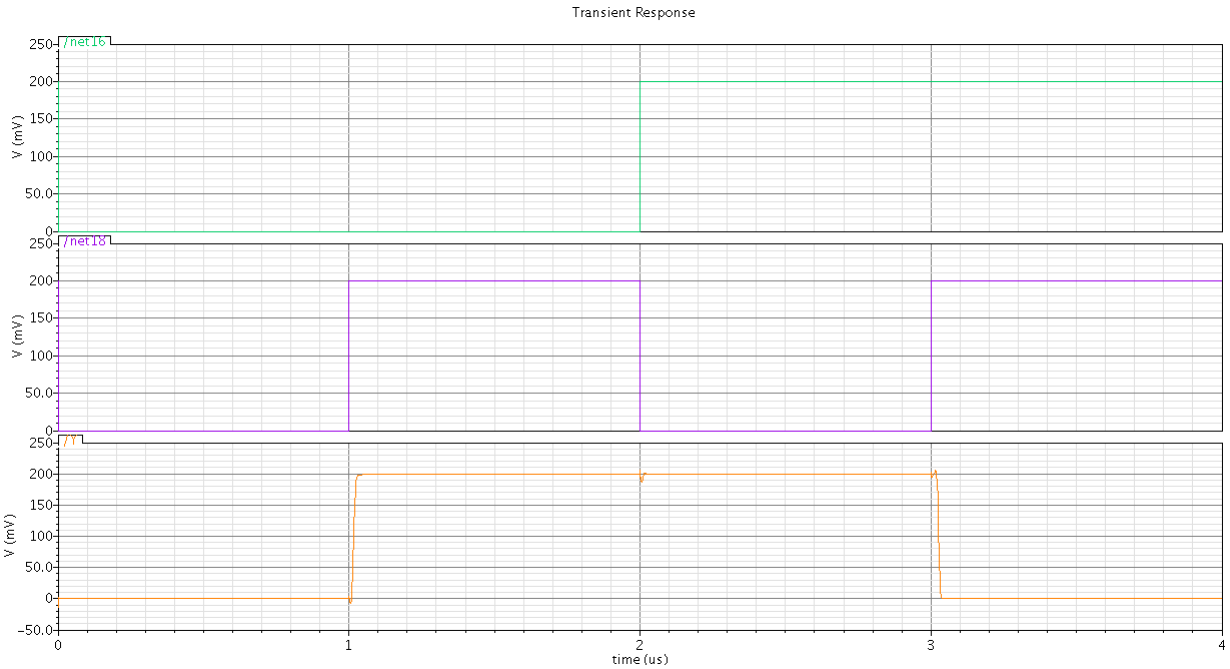


Figur 2.1.6.1: Skjematikk for en XOR port



Figur 2.1.6.2: Symbolet til en XOR port

XOR portens bruksområde er feil detektering[8] og addering pga portens egenskap om at utgangen er *høy* hvis og bare hvis en av inngangene er *høy*. Neste figur viser simuleringen av kretsen i figur 2.1.6.1.



Figur 2.1.6.3: Simulering av en XOR port ved 27 ^{o}C og V_{DD} = 180 mV (de 2 øverste grafene er inngangene og den nederste er utgangen)

Grunnen til at det ikke står et 2 tall etter XOR navnet slik det gjør i de andre portene, er at det ikke gir noe mening "å XOR'e" flere enn 2 innganger. Hvis man skal "XOR'e" flere signaler er det vanlig å XOR'e 2 signaler først, også ta resultatet av det og XOR'e med neste signal. Ved å gjøre dette så kan vi lage en paritets generator som vi kan bruke til å sjekke om vi har et partall eller oddetall av *høye* eller *lave* innganger.

2.2 Logiske porter med redundans

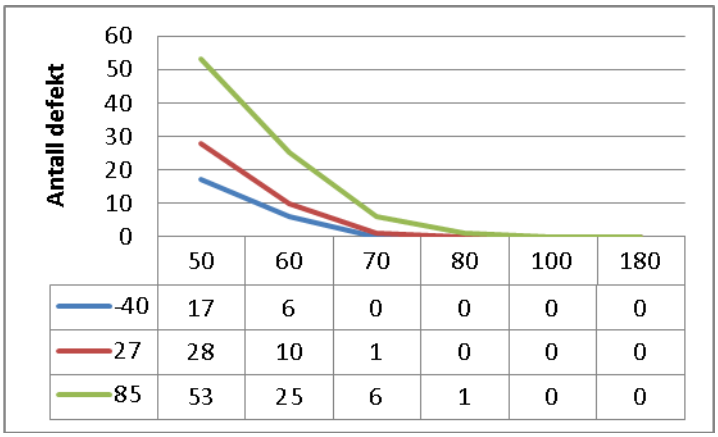
Redundans kalles innformasjon som gjentar allerede etablert kunnskap uten å tilføre noe nytt. Redundant innformasjon kan derfor også kalles overskuddsinformasjon. Slik innformasjon kan ofte være med å tydeligjøre en mening, men kan også sees på som noe som tar unødvendig plass som bør fjernes. Redundans i logiske kretser kan oppnås ved å koble flere like kretser i pararell, med samme innganger og utgangene til hver av kretsene koblet sammen. På denne måten så kan vi få en mer stabil utgang, som er avhengig av redundans graden. Høyere ordens redundans (flere identiske kretser i pararell) gir mer stabilitet.

En annen fordel med redundans er at hvis to systemer jobber i pararellt, kan den ene ta over hvis den andre skulle bli ødelagt. Hvis vi da har flere identiske kretser som jobber i pararell, kan vi ved en eventuell feil (at en krets skulle vise en annen verdi en de resterende kretsene), så kan det svaret som meste parten av kretsene gir bli regnet som riktig. Dette gjør at vi får en mer feiltolerant system, noe som er veldig gunstig å ha.

Ulempen med å ha redundans i kretser er at areal forbruket til en krets øker som følge av økende redundans. Hvis vi skal ha flere kretser som jobber i pararell, f. Eks. To datamaskiner så sier det seg selv at effekt forbruket også blir større og følgelig varmegenerering og kravet til nedkjøling vil øke. Ved å sammenligne fordelene med ulempene kan vi ved en bestemt sitiuasjon overveie om det er gunstig å ha redundans.

2.2.1 Redundans i en INVERTER

Hvis vi setter to invertere i pararell med samme innganger og utgangene koblet sammen, så de jobber sammen om samme oppgave for vi ved MonteCarlo simuleringer følgende graf.



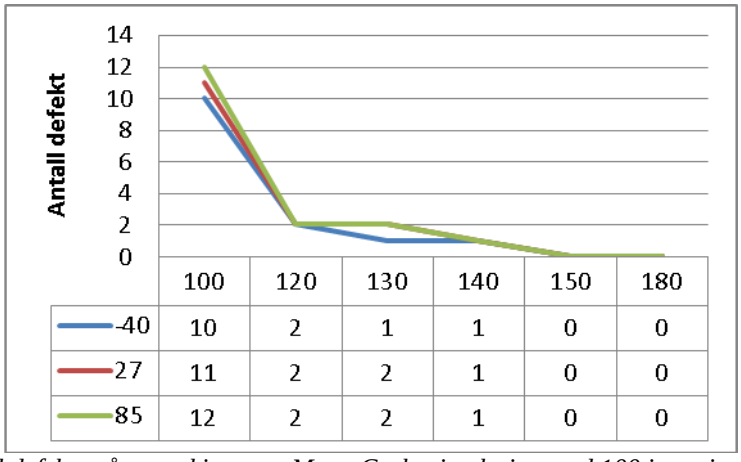
	50	60	70	80	100	180
-40	17	6	0	0	0	0
27	28	10	1	0	0	0
85	53	25	6	1	0	0

Figur 2.2.1.1: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}, ved redundans lik 2.

Redundans lik 3 gir laveste fungerende V_{DD} = 75 mV, ved redundans lik 4 fungerer den ved V_{DD} = 75 mV og med en redundans på 5 for vi inverteren til å fungere ved alle temperaturer (-40^{o}C, 27^{o}C, 85^{o}C) med en V_{DD} på 70mV. (Se figur C.1.1)

2.2.2 Redundans i en NAND2 port

Hvis vi setter to NAND2 porter i pararell med samme innganger og utgangene koblet sammen, så de jobber sammen om samme oppgave for vi ved MonteCarlo simuleringer følgende graf.



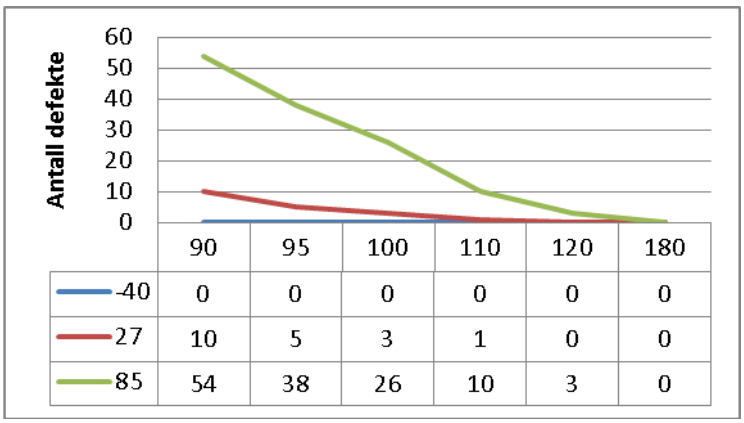
	100	120	130	140	150	180
-40	10	2	1	1	0	0
27	11	2	2	1	0	0
85	12	2	2	1	0	0

Figur 2.2.2.1: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}, ved redundans lik 2.

Redundans lik 3 gir laveste fungerende V_{DD} = 120 mV, ved redundans lik 4 fungerer den også ved V_{DD} = 120 mV og med en redundans på 5 for vi NAND2 porten også til å fungere ved alle temperaturer (-40°C, 27°C, 85°C) med en V_{DD} på 120 mV. Dette viser oss at det ikke er noe å hente ved å øke redundansen fra 3 – 5 for NAND2 porten. (Se figur C.1.1)

2.2.3 Redundans i en NOR2 port

Hvis vi setter to NOR2 porter i pararell med samme innganger og utgangene koblet sammen, så de jobber sammen om samme oppgave for vi ved MonteCarlo simuleringer følgende graf.



	90	95	100	110	120	180
-40	0	0	0	0	0	0
27	10	5	3	1	0	0
85	54	38	26	10	3	0

Figur 2.2.3.1: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}, ved redundans lik 2.

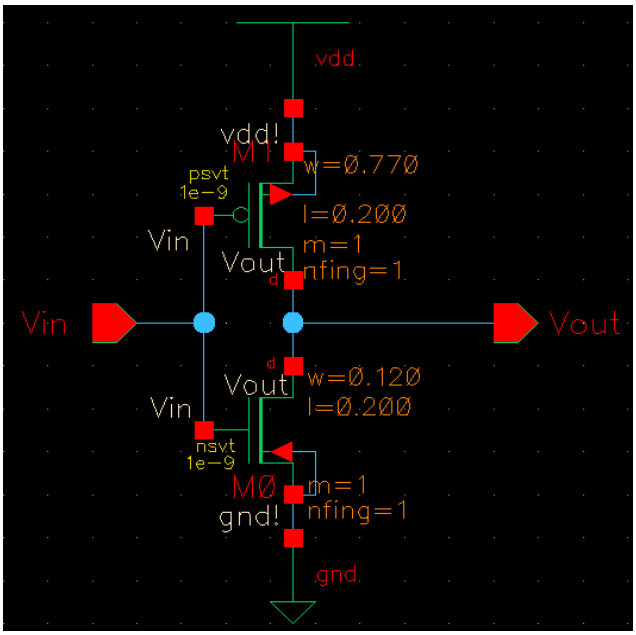
Ved redundans = 3 og redundans = 4 vil kretsen fungere helt nede i 120 mV og ved redundans = 5 vil NOR2 porten fungere helt nede ved V_{DD} = 110 mV. Dette betyr at man må vurdere om det lønner seg å bruke redundans = 5 framfor redundans = 3, om fordelene er mer enn ulempene. (Se figur C.1.1)

2.3 Logiske porter med dobbel gatelengde

En annen måte å øke stabiliteten til en krets på er å øke gatelengden. Ved å bruke større transistorer enn minimumslengde transistorer får vi en mer stabil krets, som er mer robust for endringer i spesifikasjoner. Ved å øke gatelengden og gate oksid tykkelsen, får vi en mindre lekasjestrøm [10]. Mindre lekasjestrømmer gir raskere kretser, som også er mer pålitelige. Dette betyr at for en gatelengde = 0,2 µm istedenfor 0,1 µm kan vi forvente at kretsene fungerer ved lavere V_{DD}'er, selvom de er designet for å fungere ved V_{DD} = 180 mV.

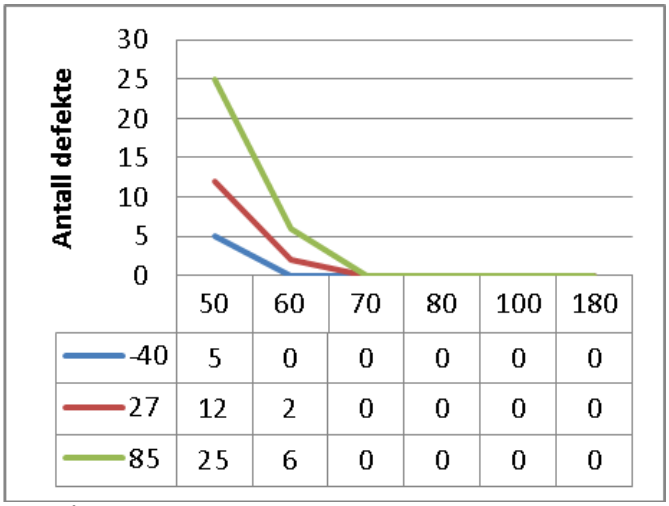
2.3.1 INVERTER med dobbel gatelengde

En inverter med en L = 0,2 µm ser ut som på figur 2.3.1.1. For å få en matching må pMos – transistorene ha en W = 0,77 µm, når nMos – transistorene har en minimumsgatebredde.



Figur 2.3.1.1: Skjematikk for en inverter

Ved å simulere på kretsen ovenfor får vi følgende resultater.



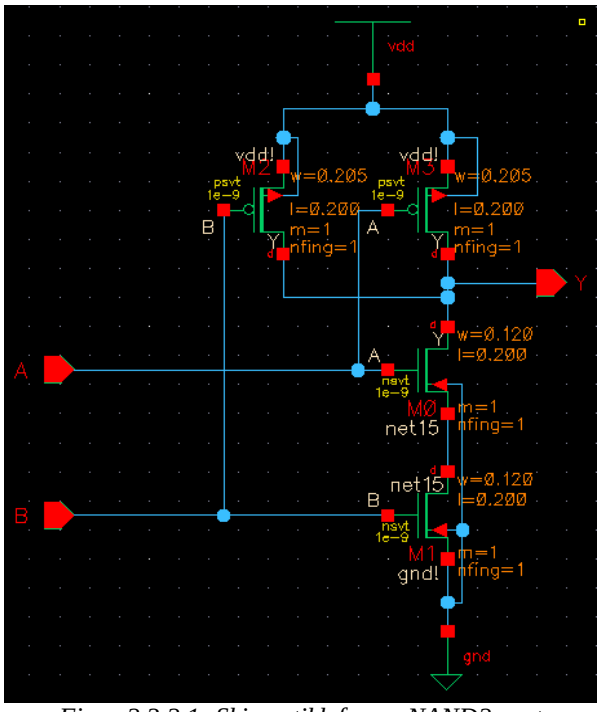
	50	60	70	80	100	180
-40	5	0	0	0	0	0
27	12	2	0	0	0	0
85	25	6	0	0	0	0

Figur 2.3.1.2:Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}, med L = 0,2 µm.

Figur 2.3.1.2 viser oss at inverteren på figur 2.3.1.1 fungerer helt nede med en V_{DD} = 70 mV, selvom kretsen er matchet for å fungere med en V_{DD} = 180 mV.

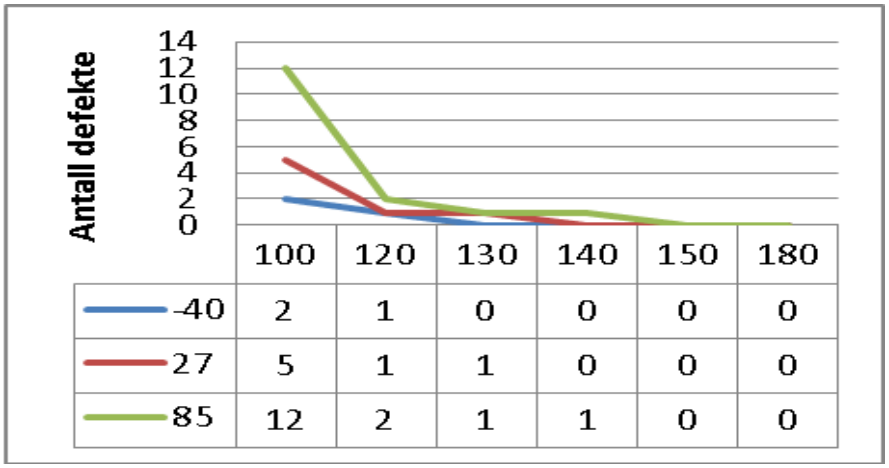
2.3.2 NAND2 port med dobbel gatelengde

En NAND2 port med en L = 0,2 µm ser ut som på figur 2.3.2.1. For å få matching må pMos transistorene ha en W = 0,205 µm, når nMos transistorene har en minimumsgatebredde.



Figur 2.3.2.1: Skjematikk for en NAND2 port

Simulering av kretsen overfor gir følgende resultater.



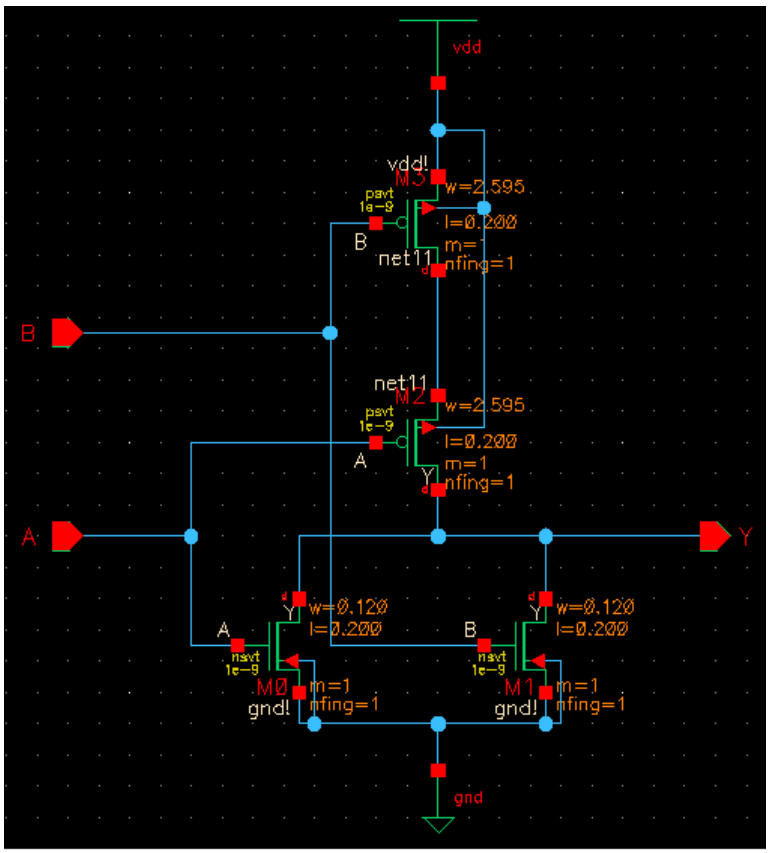
	100	120	130	140	150	180
-40	2	1	0	0	0	0
27	5	1	1	0	0	0
85	12	2	1	1	0	0

Figur 2.3.2.2:Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}, med L = 0,2 µm.

Grafen og tabellen i figur 2.3.2.2 viser at NAND2 porten i figur 2.3.2.1 fungerer med en V_{DD} = 150 mV.

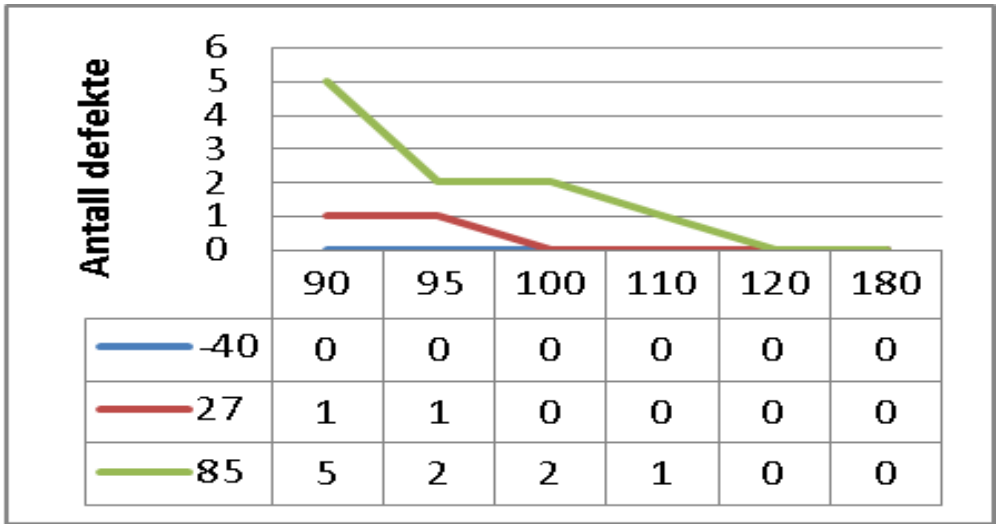
2.3.3 NOR2 port med dobbel gatelengde

En NOR2 port med en L = 0,2 µm ser ut som på figur 2.3.3.1. For å få en matching må pMos transistorene ha en W = 2,595 µm, når nMos transistorene har en minimumsgatebredde.



Figur 2.3.3.1: Skjematikk av en NOR2 port

Ved å simulere på kretsen ovenfor får vi følgende resultater.



	90	95	100	110	120	180
-40	0	0	0	0	0	0
27	1	1	0	0	0	0
85	5	2	2	1	0	0

Figur 2.3.3.2:Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 3 forskjellige temperaturer og 6 forskjellige verdier av V_{DD}, med L = 0,2 µm.

Figur 2.3.3.2 viser oss at NOR2 porten i figur 2.3.3.1 fungerer med en V_{DD}= 120 mV.

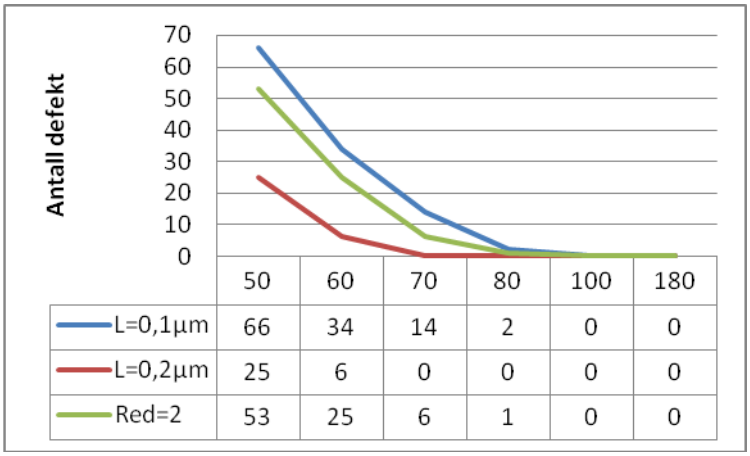
2.4 Diskusjon av forskjellige kretsteknikker

Vi har sett på tre forskjellige måter å lage kretser på for en INVERTER, en NAND2 port og en NOR2 port. Først var dem designet med en minimumslengde L = 0,1 µm, så brukte vi redundans på de kretsene, og til slutt så designet vi kretsene ved å doble lengden til L = 0,2 µm. Hvem av disse teknikkene vil være best hvis vi sammenligner dem? Det er klart at minimumslengde er det "dårligste" alternativet, men er det det å doble gatelengden, eller det å ha redundans som er det beste alternativet? Dette kommer vi til å se på for hver av de tre portene(INVERTER, NAND2 og NOR2), hver for seg.

Vi har sett at ved simuleringer er det 85 ^{o}C som er den mest problematiske temperaturen å få til å fungere. Hvis vi tenker litt på teknologien rundt oss vil vi se at dette ikke er så veldig overraskende. En datamaskin som blir varm blir tregere og derfor så kjøles den ned. Det er ikke bare at datamaskinen blir tregere som gjør at den blir kjølet ned, men det er en av grunnene.

En annen ting som er verdt å ta i betraktning før vi begynner å sammenligne er at vi bruker redundans = 2 under sammenligningene. En redundans av høyere orden vil bruke mye areal på "chippen", mer enn transistorer med dobbel gatelengde. Så får å få en mest mulig rettferdig sammenligning, sammenligner vi ved T = 85 ^{o}C ("worst case") og med redundans = 2.

2.4.1 Diskusjon av INVERTER



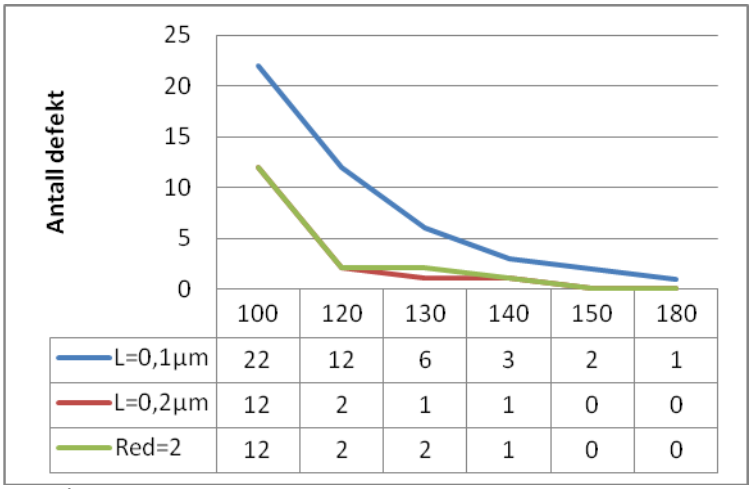
	50	60	70	80	100	180
L=0,1µm	66	34	14	2	0	0
L=0,2µm	25	6	0	0	0	0
Red=2	53	25	6	1	0	0

Figur 2.4.1.1: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 85 ^{o}C og 6 forskjellige verdier av V_{DD}.

Figur 2.4.1.1 viser oss for en inverter, som vi vil skal være mest mulig stabil ved lavere V_{DD} verdier enn 180 mV som den er designet for, vil det beste alternativet være å doble gatelengden. Med en gatelengde L = 0,2 µm vil inverteren fungere helt til V_{DD} = 70mV. Å ha redundans på denne porten vil ikke ha noen effekt, for inverteren med L = 0,1µm vil også fungere med V_{DD} = 100 mV slik som inverteren med en redundans = 2.

At inverter'en fungerer med så lav V_{DD} tyder på at det er en veldig stabil og pålitelig krets. Dette kan skyldes det faktum at den er veldig enkel å implementere. Som vi vet består jo den kun av en nMos- og en pMos transistor i serie mellom GND og V_{DD} (figur 2.1.3.2).

2.4.2 Diskusjon av NAND2



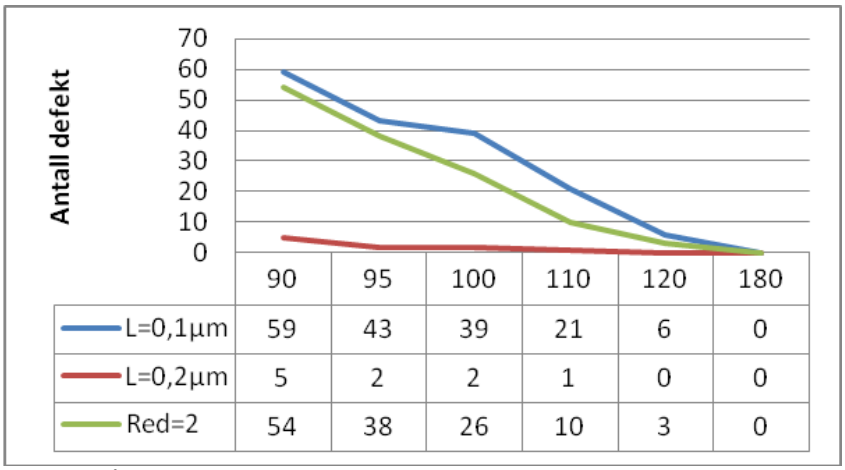
	100	120	130	140	150	180
L=0,1µm	22	12	6	3	2	1
L=0,2µm	12	2	1	1	0	0
Red=2	12	2	2	1	0	0

Figur 2.4.2.1: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 85 ^{o}C og 6 forskjellige verdier av V_{DD}.

Figuren ovenfor viser oss at for en NAND2 port vil det å doble gatelengden være like bra som det å ha en redundans = 2. Begge deler vil fungere med en V_{DD} = 150 mV noe som er bedre en NAND2 porten med L = 0,1 µm som har en defekt andel på 1% ved V_{DD} = 180 mV.

At NAND2 porten ikke virker 100% med V_{DD} = 180 mV viser oss at den er mindre pålitelig enn det inverteren var. Dette er ikke overaskende i og med at NAND2 porten er bygget opp ved hjelp av flere transistorer enn hva det inverteren var (figur 2.1.4.2), og som en følge av det har NAND2 porten flere parasittiske kapasitanser som er med på å gjøre den mindre stabil.

2.4.3 Diskusjon av NOR2



	90	95	100	110	120	180
L=0,1µm	59	43	39	21	6	0
L=0,2µm	5	2	2	1	0	0
Red=2	54	38	26	10	3	0

Figur 2.4.3.1: Antall defekte når man kjører en MonteCarlo simulering med 100 iterasjoner, ved 85 ^{o}C og 6 forskjellige verdier av V_{DD}.

Figuren over viser oss at nok en gang så er det dobling av gatelengden som er det beste alternativet, hvis vi vil at kretsen skal fungere med en lavere V_{DD} enn det porten er designet for. Ved dobling av gatelengden får vi NOR2 porten til å fungere med en V_{DD} = 120 mV. Redundans = 2 vil her ikke få porten til å fungere ved en lavere V_{DD}, men det vil gjøre at defekt andelen blir mindre etter som V_{DD} blir mindre.

Figuren viser oss at NOR2 porten har 100 % ”yield” for V_{DD} = 180 mV, selv når den er bygget opp ved hjelp minimumslengde transistorer. Dette betyr at NOR2 porten er mer pålitelig enn NAND2 porten. Dette skyldes at NOR2 porten har bredere transistorer i opptrekket (figur 2.1.5.2) enn NAND2 porten. Dette fører til at selvom NOR2 porten har transistorer i serie i opptrekket i motsetning til NAND2 porten som har transistorer i pararell i opptrekket, vil klare å dra opp til ’1’ mye bedre.

Kapittel 3

Simuleringsresultater for FA, Dekoder, MUX

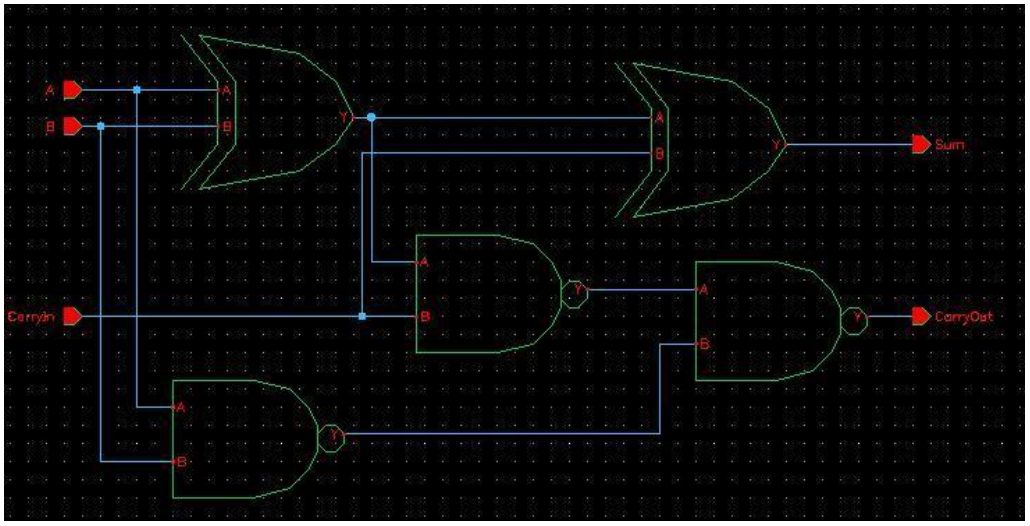
3.1 2 – bit Full- Adder

En 2 – bit Full – Adder er som navnet tilsier en addisjons krets som adderer 2 bit. Selv om den adderer 2 bit har den tre inngangs bit, hvor det tredje bit'et er et "carry" bit, som stammer fra en tidligere addering hvis det er foretatt. 2 – bit Full – Adderen er den minste byggeblokka i en adderingskjede, og man kan i realiteten sette sammen uendelig mange Full – Addere i serie for å få "uendelig bit addisjon". Sannhetstabellen for en 2 – bit Full – Adder finner vi tabell 3.1.1.

A	B	C_{In}	C_{Out}	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

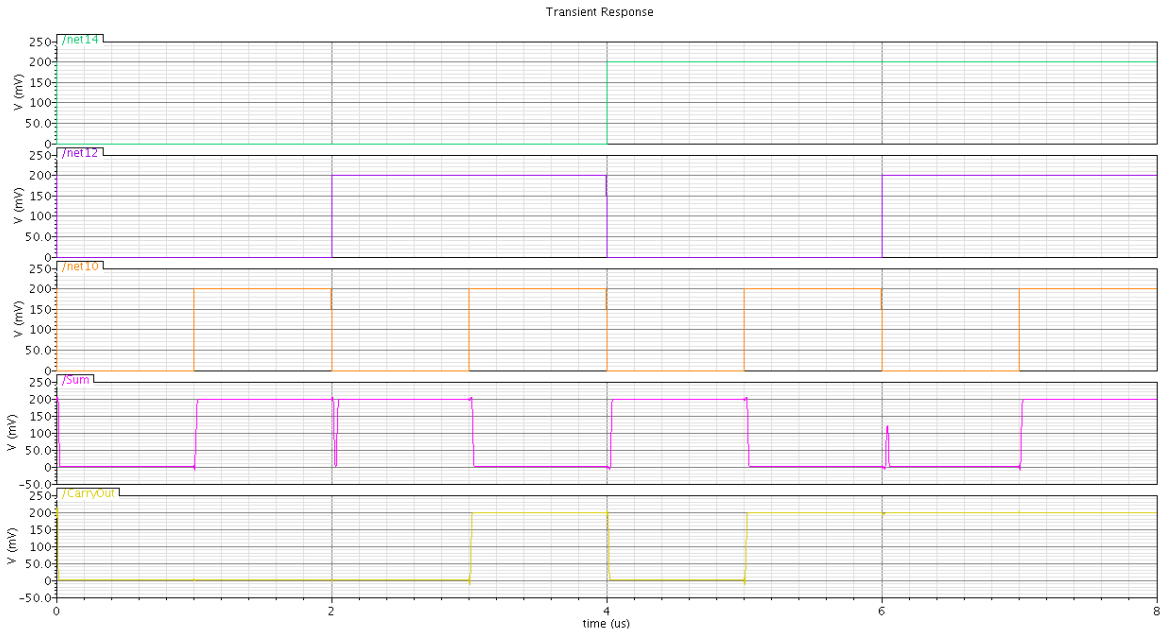
Tabell 3.1.1: Sannhetstabellen for en 2 – bit Full – Adder(A,B,C_{In}=innganger og C_{Out}, S= utganger)

Det finnes mange måter å realisere en 2 – bit Full – Adder på på kretsnivå, men en implementasjon kan se ut som på figur 3.1.1. Som vi ser av figuren består den av to XOR – porter og tre NAND2 – porter som gjør at den er forholdsvis enkel å implementere.

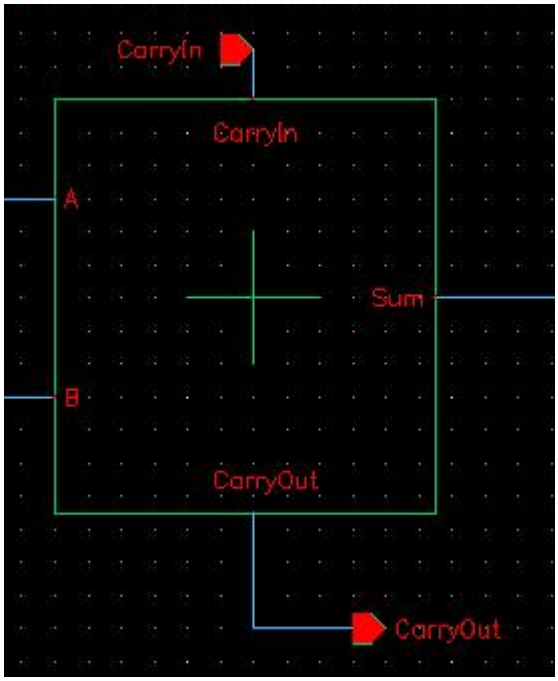


Figur 3.1.1: Skjematikk for en 2 – bit Full – Adder med Carry.[17]

Hvis vi simulerer på denne kretsen, vil vi få et resultat som på figur 3.1.2. Som vi ser av figuren stemmer simuleringene med sannhetstabellen til kretsen. De tre øverste grafene er innganger (A, B og C_{In} henholdsvis), og de to nederste er utganger hvor den nederste er C_{Out} og den over er S. Symbolet til en 2 – bit Full – Adder ser vi i figur 3.1.3.



Figur 3.1.2: Simulering av en 2 – bit Full – adder(V_{DD} = 200mV, T=27°C)

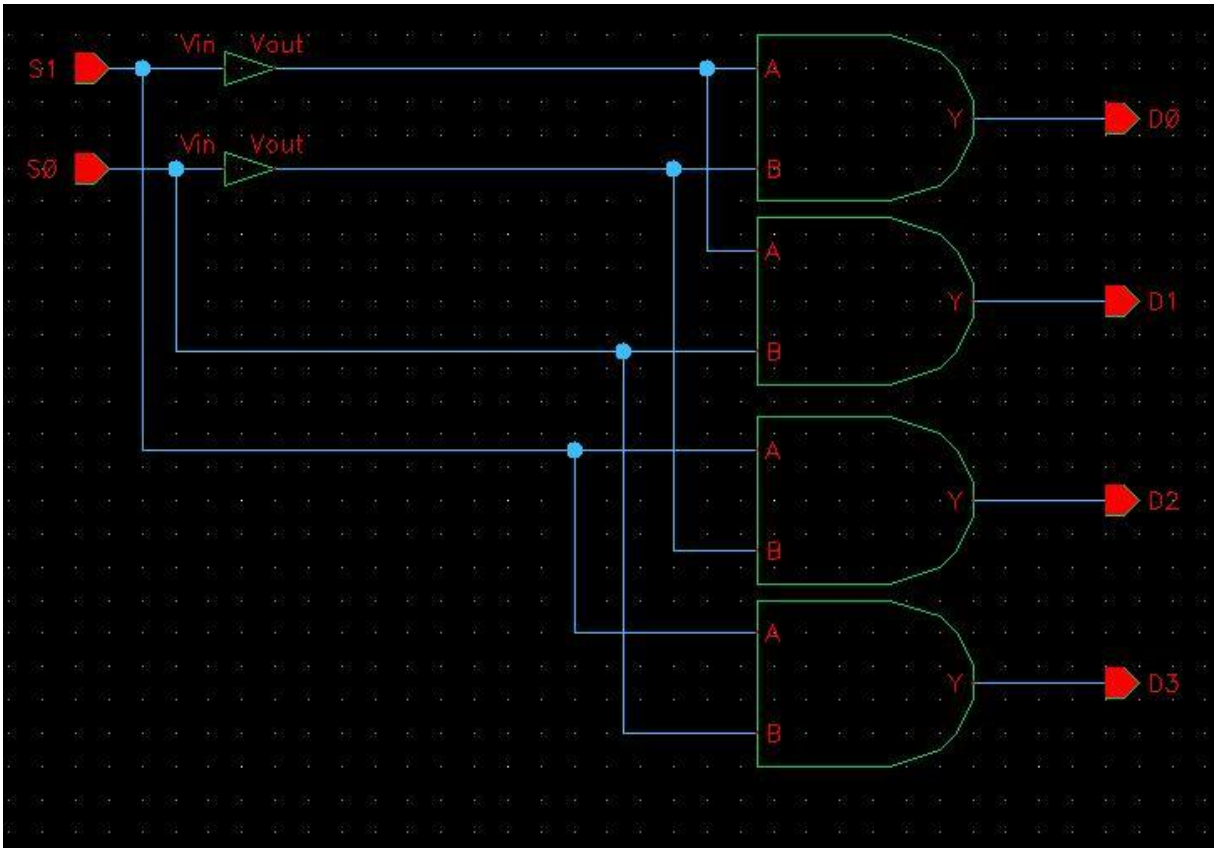


Figur 3.1.3: Symbolet til en 2 – bit Full – Adder

3.2 2:4 Dekoder

I digitale systemer er informasjon representert ved hjelp av binære koder(0 og 1), som vi vet. En binær kode på n – bit er kapabel til å representere 2^n forskjellige elementer av kodet informasjon. F. eks vil en 2 – bit binær kode representere $2^2 = 4$ forskjellig informasjon(00, 01, 10, 11). En dekoder er en kombinatorisk krets som konverterer binær informasjon fra n – innganger til et maksimum av 2^n utganger[18]. Grunnen til at det er maksimum er at inngangene kan ha en

kombinasjon som vi ikke bruker. En n – 2^n kan se ut som på neste figur. Dekoderen i figuren er som vi ser en 2 – 4 dekoder.



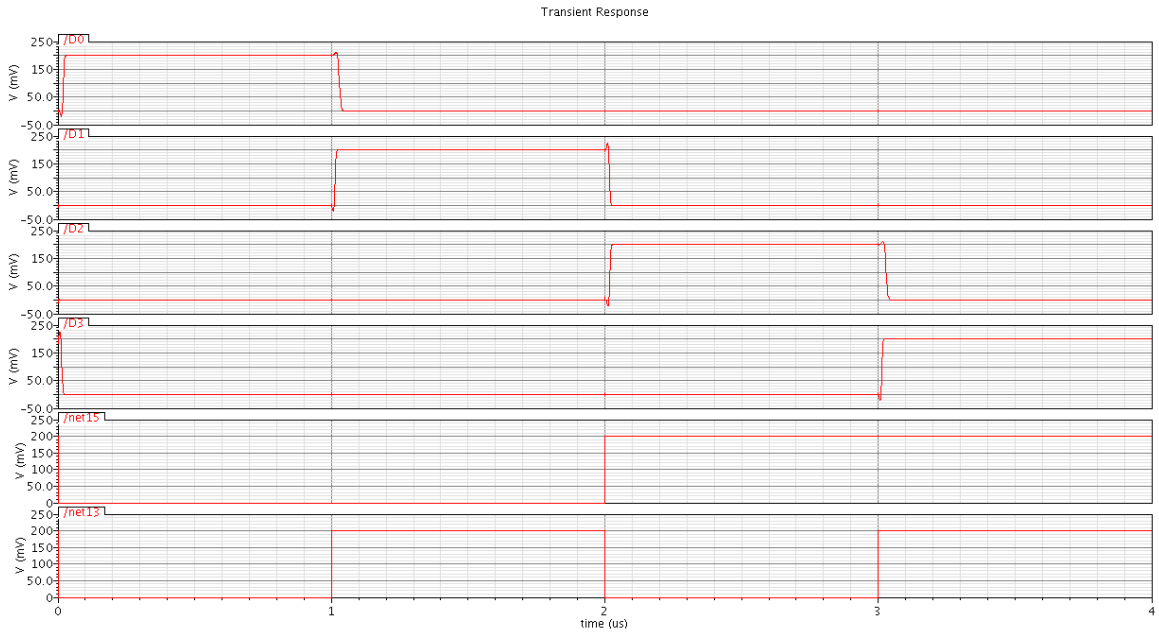
Figur 3.2.1: Skjematikk for en 2 – 4 dekoder [19].

Denne dekoderens oppgave er å ha kun en utgang *høy* om gangen for ulike innganger. På denne måten kan vi velge hvem kombinasjoner av innganger vi vil jobbe med. Virkemåten til denne dekoderen kan bli klarere å se ved å ta en titt på sannhetstabellen i tabell 3.2.1. Som vi ser av tabellen er kun en utgang *høy* mens de resterende tre utgangene er *lave* for en inngangskombinasjon.

S1	S0	D0	D1	D2	D3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

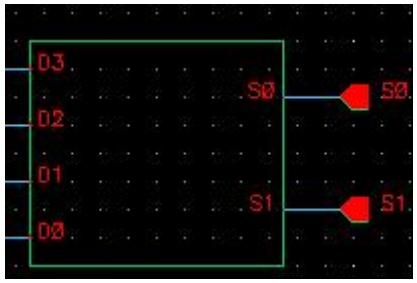
Tabell 3.2.1: Sannhetstabellen for en 2 – 4 dekoder.

Simulering av kretsen i figur 3.2.1 gir resultatene som vi ser i neste figur, og stemmer godt overens med sannhetstabellen. Grafene i figuren er henholdsvis D0, D1, D2, D3, S1 og S0.



Figur 3.2.2: Simulering av en 2 – 4 dekoder(V_{DD}=200mV og T=27^{o}C)

Dekoderen i figur 3.2.1 kan representeres ved hjelp av følgende symbol.



Figur 3.2.3: Symbolet til en 2 – 4 dekoder

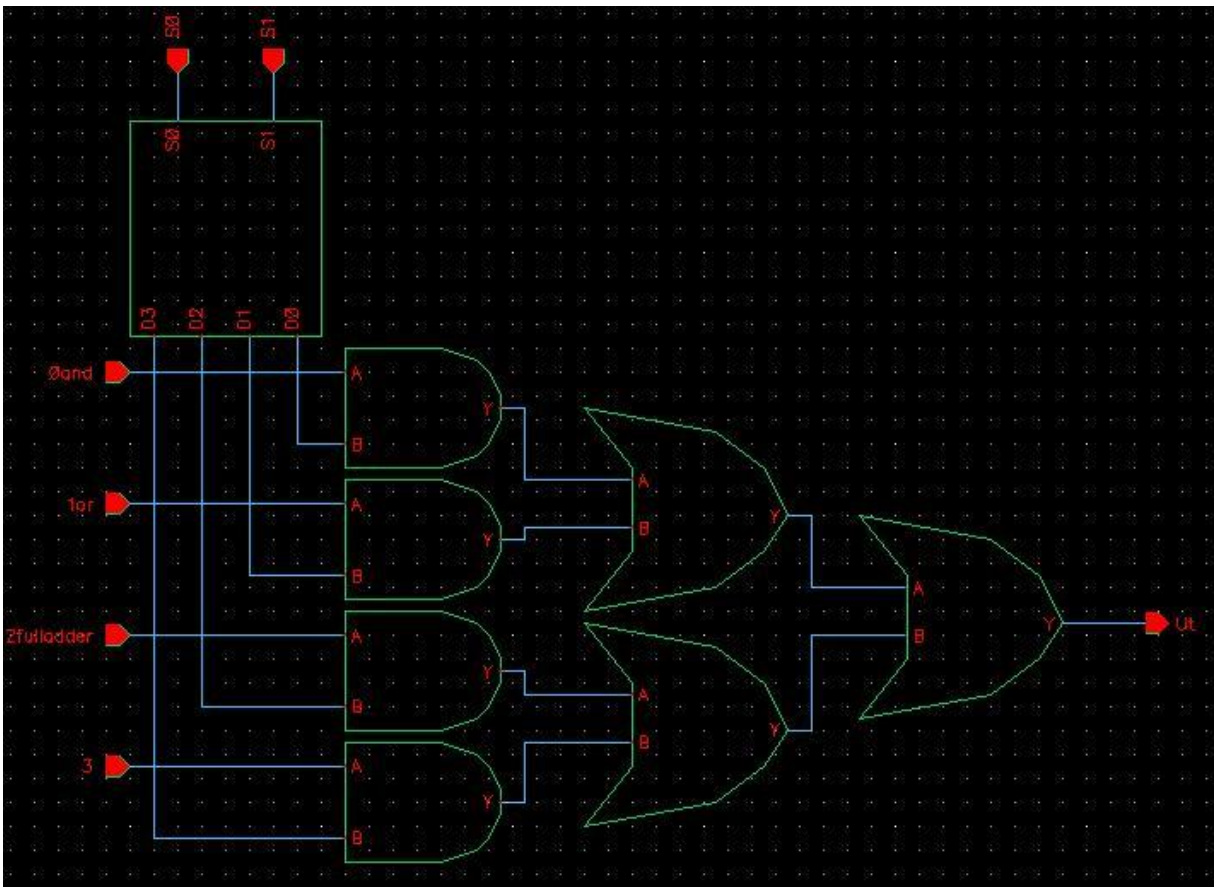
3.3 MUX

En multiplekser eller MUX, er en krets som velger verdien til en av mange innganger og kjører det ut på utgangen. Valget av innganger blir styrt av noen styre innganger. Vanligvis er det 2^n forskjellige innganger og n styre signaler. Sannhetstabellen til en 4 inngangs MUX med 2 styresignaler ser slik ut.

S1	S0	Out
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

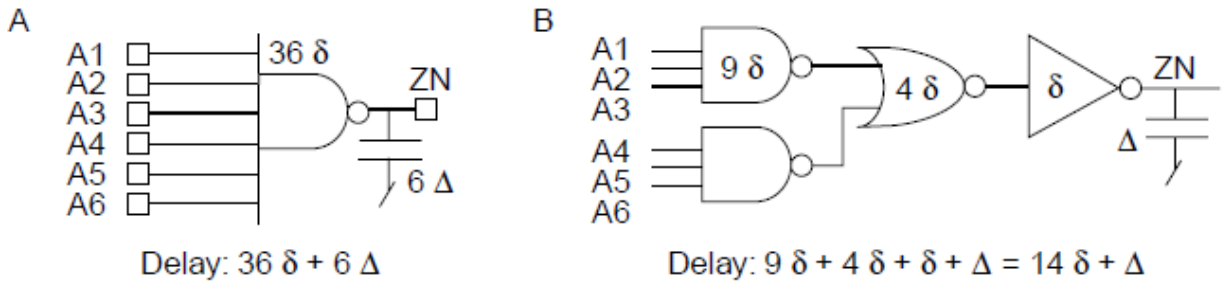
Tabell 3.3.1: Sannhetstabellen til en 4 til 1 MUX

$I_{0\text{-}3}$ representerer de 4 ulike inngangene som multiplekseren har. En skjematisk tegning av en 4 – 1 multiplekser kan se ut som på figur 3.3.1. Vi ser at det blir brukt en dekoder for å dekode styresignalene. Dette gjøres fordi en styresignal på 2 bit kan representere fire forskjellige verdier, og vi har kun lyst på en *høy* styresignal om gangen, som skal AND'es med den inngangen som er valgt og levere den på utgangen. Vi husker at for at en AND port skal gi et *høyt* signal på utgangen må begge inngangene være *høye*. Dette gjør at alle andre innganger som ikke blir valgt vil gi en 0'er på utgangen sin og ikke ha en innvirkning på den endelige utgangen til MUX'en.



Figur 3.3.1: Skjematikk for en 4 – 1 MUX.

Vi ser også at utgangene til de fire AND portene er koblet sammen ved hjelp av tre OR2 porter, istedenfor en OR4 port. Dette skyldes at en OR4 port som er dekomponert til porter med mindre enn tre innganger får en mindre forsinkelse[20]. Figur 3.3.2 viser dekomposisjonen av en NAND6 port.

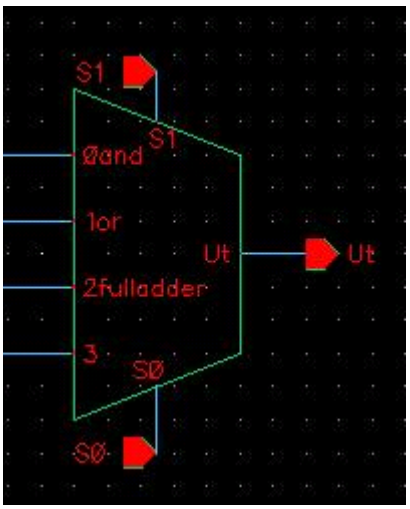


Figur 3.3.2: Dekomposisjon av en NAND6 port[21].

En forsinkelses modell [21] sier at en n – inngans NAND port har en n transistorer i serie som fører til en økning av den interne resistansen av n*δ, og den interne parasittiske kapasitansen øker også med en faktor n. Dette betyr at den interne forsinkelsen til en n – inngangs NAND er $n^2*\delta$. Last forsinkelsen er n*Δ, fordi utgangskapasitansen må lades og utlades av n transistorer i serie. Dette gir oss en total forsinkelse for en n inngangs NAND port,

$$forsinkelse = (n^2 * \delta) + (n * \Delta) \qquad (3.3.1)$$

Ved å bruke denne formelen kan vi regne oss fram til de verdiene som er vist i figur 3.3.2.



Figur 3.3.3: Symbolet til en 4-1 mulitplekser.

Figur 3.3.3 viser symbolet til en 4- 1 multiplekser.

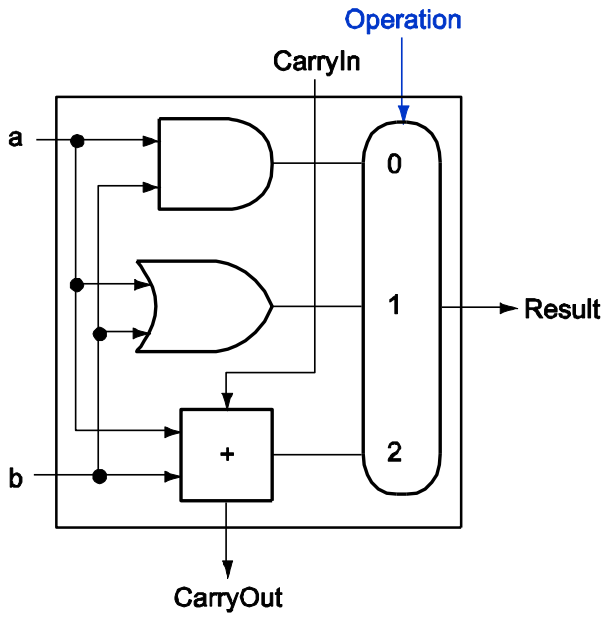
Kapittel 4

Simuleringsresultater for en enkel Aritmetisk Logisk Enhet(ALU)

4.1 Hva er en ALU?

Aritmetisk logisk enhet eller en ALU, er som navnet tilsier en krets som utfører aritmetiske og logiske operasjoner. En aritmetisk operasjon er en matematisk operasjon som addisjon, subtraksjon, multiplikasjon osv., mens en logisk operasjon er en digital operasjon som AND, OR, INVERT og så videre. ALU'en er den mest fundementale kretsen i en CPU og selv den enkleste mikroprossesoren inneholder en. Moderne CPU'er og GPU'er inneholder ofte flere veldig komplekse ALU'er som gjerne jobber i pararell.

En ALU får sine data fra registre, utfører operasjonen den er bedt om å utføre, og skriver så resultatene i andre registre. En kontroller forteller ALU'en hvilken operasjon den skal utføre, og andre utenfor liggende kretser sørger for å flytte data mellom minnet og registre. På figuren nedenfor ser vi en enkel ALU som kan utføre AND, OR og addisjon av to bit, men ALU'en trenger nødvendigvis ikke å utføre så enkle operasjoner. Denne kan også designes til å utføre mer kompliserte operasjoner som f. eks utregning av kvadratrøtter, men jo mer kompliserte operasjonene blir desto dyrere, strømkrevende og plasskrevende blir den. Derfor er det viktig å designe en ALU som holder til dine formål.

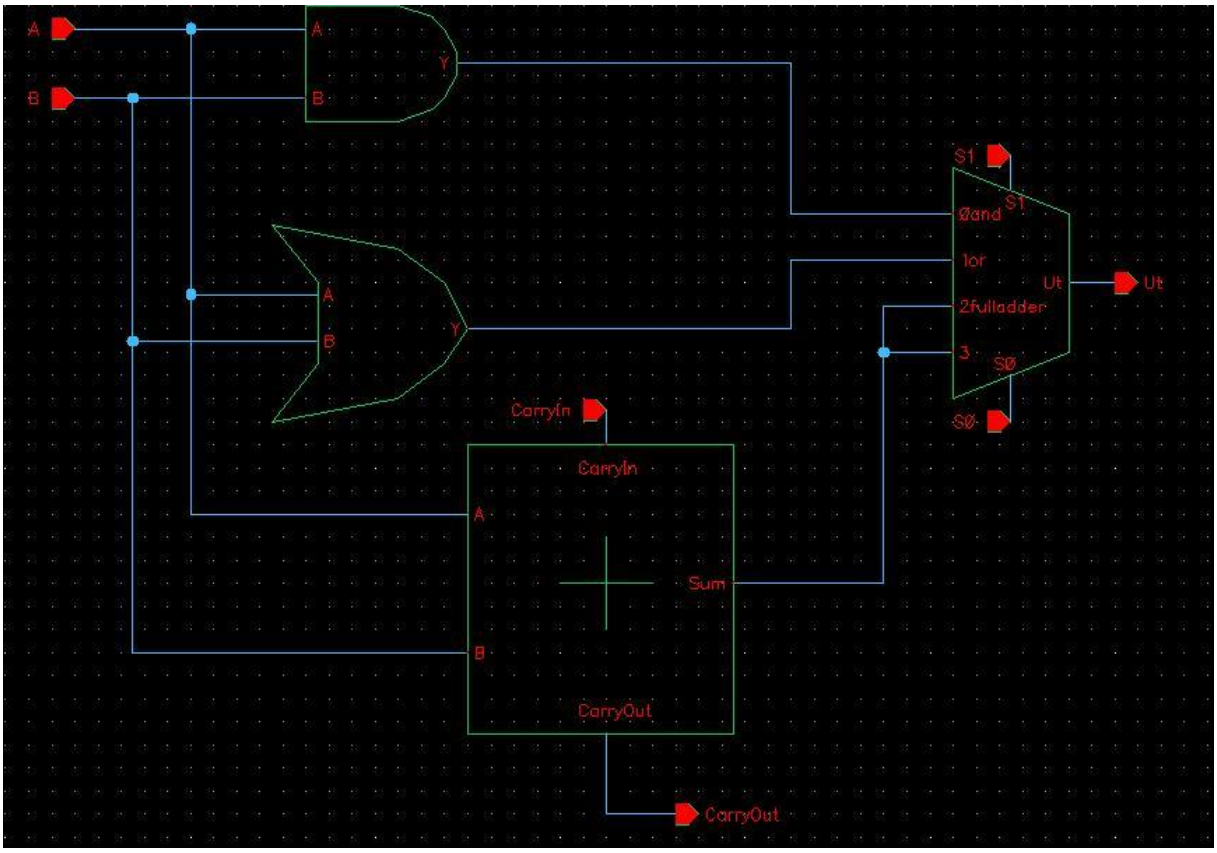


Figur 4.1.1: En enkel ALU som utfører AND, OR og addisjon.[17]

Grunnen til at ALU'en ble utviklet, var at en matematiker ved navn John von Neumann hevdet at ALU var en nødvendighet i en datamaskin, fordi det var hevet over all tvil at en ville måtte utføre regneoperasjoner som addering, subtrahering, multiplisering og dividering. Han mente at det var naturlig at en datamaskin skulle ha spesialiserte «organer» for slike operasjoner [22].

4.1.1 Implementasjon av en enkel ALU

Vi kan designe en ALU som den på figur 4.1.1 ved å bruke de logiske kretsene og portene vi har vist i tidligere kapitler med transistor lengder på L = 0,1 µm, og da vil ALU'en se slik ut.



Figur 4.1.1.1: Skjematikk for en enkel ALU som utfører AND, OR og addisjon

Vi ser at ALU'en har to innganger (A og B) og at den har et to bit styresignal S (S0 og S1). Avhengig av hvilken operasjon vi vil utføre sender vi inn styresignalene som er vist i tabell 4.1.1.1.

S1	S0	Ut
0	0	AND
0	1	OR
1	0	FULLADDER
1	1	FULLADDER

Tabell 4.1.1.1: Sanhetstabellen til ALU'ens virkemåte

Vi ser at vi ved hjelp av en 2 – bit styresignal har fire valgmuligheter for operasjoner, men i og med at vi kun har tre operasjoner som ALU'en utfører har vi bestemt at for S = 10 og S = 11 vil ALU'en addere.

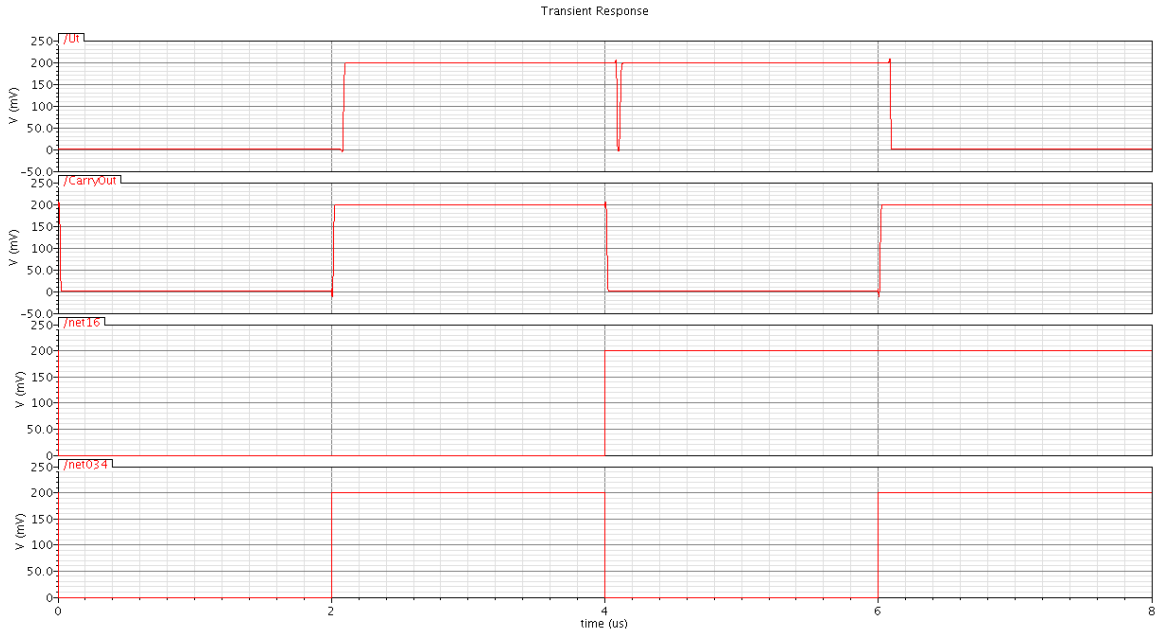
ALU'en i figur 4.1.1.1 er designet slik at alle operasjoner utføres hele tiden, men en MUX bestemmer hva den skal sende ut på utgangen. Dette vil ikke være effektivt med tanke på effekt

forbruk, for da hadde det lønnet seg og kun la de operasjonene vi ønsker på utgangen arbeide og la de andre ikke jobbe, men slik er det ikke i denne enkle ALU'en.

For å simulere denne ALU' en må vi bestemme for et sett med innganger for å teste om kretsen virker. Hvis vi setter A = 0 og B = 1 vil AND2 porten alltid gi 0 på utgangen, mens OR2 porten vil gi ut en 1'er. Addisjons kretsen vil gi ut 0 eller 1 avhengig av carry signalet inn, som vi varierer mellom 0 og 1. Satt i en sannhetstabell vil dette se slik ut.

A	B	CarryIn	S1	S0	Ut	CarryUt
0	1	0	0	0	0	0
0	1	1	0	1	1	1
0	1	0	1	0	1	0
0	1	1	1	1	0	1

Tabell 4.1.1.2: Sannhetstabellen for ALU'en.



Figur 4.1.1.2:Simuleringsresultater av ALU'en (V_{DD}=200 mV og T=27^{o}C)

I figur 4.1.1.2 ser vi simuleringen av ALU'en. Den øverste grafen er utgangen, graf nummer to er carry signalet ut og de to nederste grafene er styresignalet til ALU'en. Vi ser at simuleringsresultatene er det samme som vi kunne forventet av sannhetstabellen. Vi kan utføre MonteCarlo simuleringer med forskjellige temperaturer og forskjellige V_{DD} verdier for å få et bedre innblikk i hvordan ALU'en fungerer.

V_{DD}	-40 ^{o}C	27 ^{o}C	85 ^{o}C
150 mV	100	1	2
200 mV	73	0	0
250 mV	0	0	0

Tabell 4.1.1.3:Antall ALU'er som feiler ved 100 MonteCarlo simuleringer(L=0,1µm)

Vi ser at ALU'en fungerer 100% hvis vi setter opp V_{DD} til 250 mV, men fungerer ikke 100% hvis vi derimot setter ned V_{DD} til 150 mV. Ved V_{DD} = 200 mV som den er designet for, sliter kretsen med å fungere ved -40 ^{o}C hvor den har en feilandel på hele 73%.

Vi får mindre feil på kretsen ved å øke transistor lengdene. Nå vil ALU'en fungere 100% med en V_{DD} = 200 mV som den er designet for, og den vil også virke 100% ved 27 ^{o}C og 85 ^{o}C ved

V_{DD} = 150 mV. Ved -40 ^{o}C vil vi ha 100% feil som vi hadde også på den ALU'en som var designet med 0,1 µm lange transistorer.

V_{DD}	-40 ^{o}C	27 ^{o}C	85 ^{o}C
150mV	100	0	0
200mV	0	0	0
250mV	0	0	0

Tabell 4.1.1.4:Antall ALU'er som feiler ved 100 MonteCarlo simuleringer(L=0,2µm)

4.2 Diskusjon av ALU med byggeblokker

Vi ser at ALU'en fungerer bedre når byggeblokkene er laget ved hjelp av transistorer med L = 0,2 µm. Dette fant vi ut i kapittel 2 var sant så det er ikke overaskende at ALU'en har mindre defekt andel når transistorlengdene er doblet.

Når vi øker forsyningsspenningen til ALU'en vil den fungere bedre, enn når vi senker den. Dette kommer av at jo høyere V_{DD} vi har, desto mer får vi "slått på" transistorene i ALU'en. Vi får også slått transistorene mer "av". Dette fører til at operasjonsområdene til transistorene blir større og de er i stand til å trekke mer strøm og utføre operasjonene sine bedre.

ALU'en vi har sett på i dette kapittelet blir styrt av to styresignaler som gir oss fire binære inngangskombinasjoner, og hvor disse inngangskombinasjonene varer i 1 µs. Dette fungerer fint, men vi ser at vi får en liten forsinkelse på utgangen noe som tyder på at kretsen begynner å "slite" allerede ved f = 1 MHz. Dette kommer av at ALU'en har en V_{DD} = 200 mV, og det å ha en lav V_{DD} for å ha mindre effekt fører til at hastighet blir berørt negativt. En krets som er designet med en høyere V_{DD} vil være raskere.

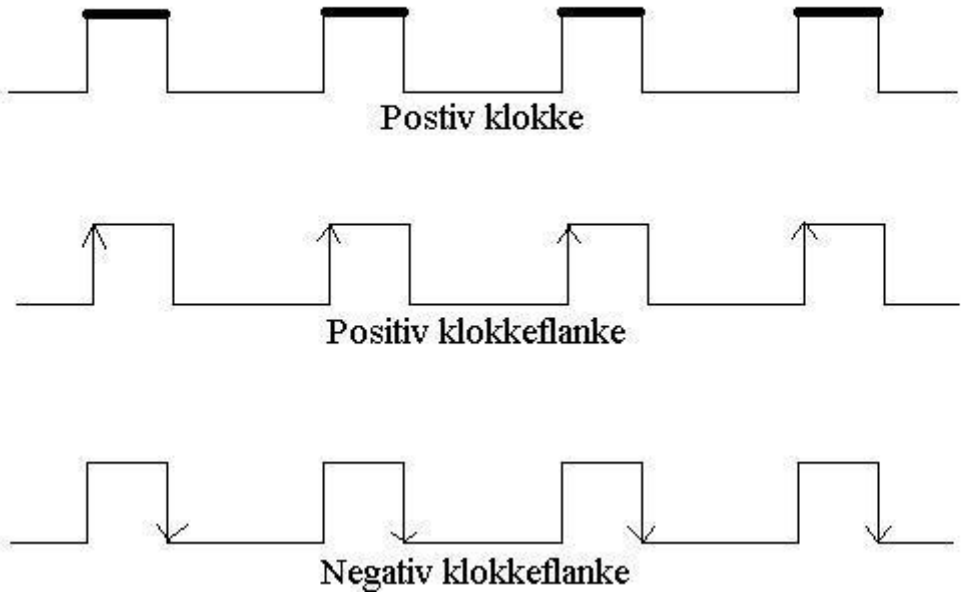
ALU'en sliter mer når temperaturen synker. Grunnen til dette er at strømmen til en transistor i subterskel operasjonsområde (formel 1.3.1) er avhengig av temperatur som $e^{x/T}$. Dette fører til at ved større systemer hvor vi har mange transistorer, så vil kretsen fungere dårligere ved lavere temperaturer, mens den vil virke bedre ved høyere temperaturer. Den vil dra mer strøm på høyere temperaturer som igjen vil føre til at vi får høyere spenninger.

Kapittel 5

Simuleringsresultater for 4 D Flip Flop' er

5.1 Hva er en Flip Flop?

En flip flop er et minne element. Den kan holde på sin verdi (så lenge den får strøm) helt til verdien blir forandret ved å forandre på inngangen(e). En flip flop forandrer bare på sin verdi ved klokkeflankene, enten positiv eller negativ, i motsetning til latchene alle flip floper blir laget av som skifter verdi ved positiv klokke [9]. Dette ser vi i figur 5.1.1. At en latch kan skifte verdi i hele den *høye* klokke perioden er veldig uheldig hvis den skal kobles sammen med andre kretser, som styres av en felles klokke. Det finnes flere typer flip flop'er, (T – flip flop, JK – flip flop og D – flip flop) og forskjellene mellom de ulike typer flip flopene er antall innganger og påvirkning av inngangen på utgangssignalet eller med andre ord sannhetstabellen.



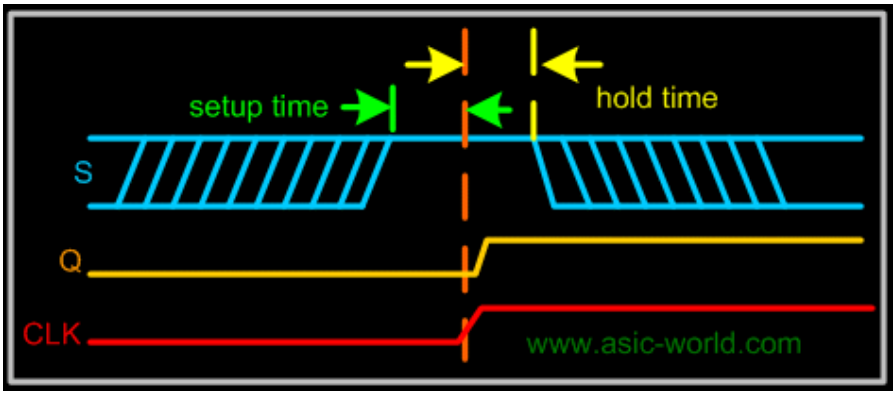
Figur 5.1.1: Klokke respons i latcher og flip floper[9].

I denne oppgaven skal vi ta for oss implementering av forskjellige typer D – flip flop'er. Det spesielle med D flip flop'er er at de kun skifter verdi i den ønskede klokkeflanken slik at inngangen blir ført rett til utgangen uten å ta hensyn til hva den forrige verdien på utgangen var. Dette kan vi se i tabellen under. D flip flop'en er også den mest økonomiske og effektive av alle flip flopene, fordi den er den flip flop'en som kan realiseres med minst antall logiske porter [26]. Andre flip flop'er kan realiseres ved å bruke en D flip flop og eksterne porter.

D	Q(t+1)
0	0
1	1

Tabell 5.1.1: Sannhetstabellen for en D flip flop

En veldig viktig parameter for en flip flop er dens setup og holdetid. Setup tid er den tiden inngangen må være stabil **før** den ønskede klokkeflanken kommer for at kretsen skal klare å lese inngangen og sende den ut på utgangen. Holdtid er den tiden som inngangen må være stabil **etter** at den ønskede klokkeflanken har kommet slik at kretsen skal klare å tolke det som en inngangsverdi og legge den ut på utgangen. Dette kommer tydelig fram i figur 5.1.2.

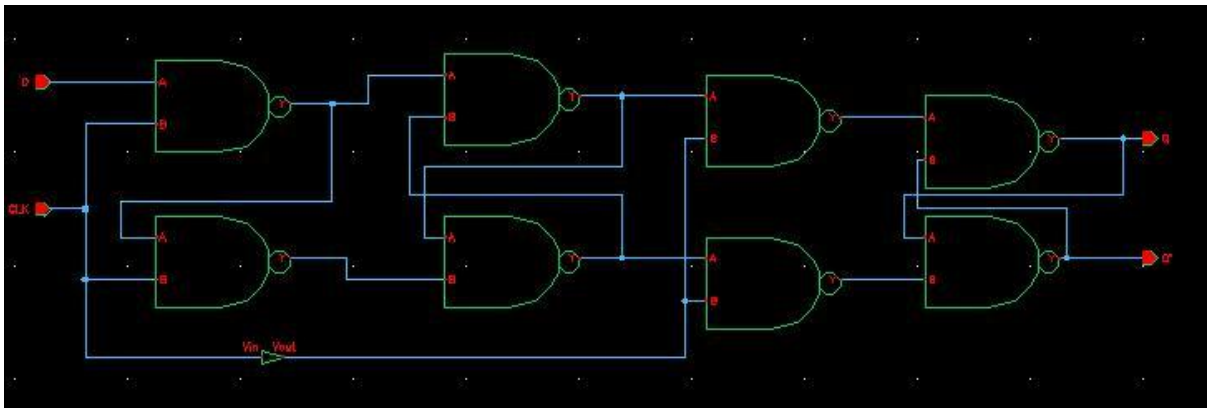


Figur 5.1.2: Setup og hold tid [23]

Hvis inngangen skifter verdi verdi mellom setup og hold vinduet rekker ikke kretsen å registrere verdien på inngangen, og flip flopen kan havne i en *meta stabil fase*. Det vil si at utgangen på flip flopen oscillerer mellom 0 og 1 og vet ikke hvilken verdi som er riktig. Hvis flip flop'en har havnet i en metastabil fase, vil det ta tid før den begynner å fungere ordentlig igjen. Dette er noe vi vil unngå, og derfor er setup- og holdetider veldig viktig for flip flop'er. Spesielt interesserte kan lese mer om flip flop'er og latcher i [46].

5.1.1 Master Slave D flip flop med NAND2 porter

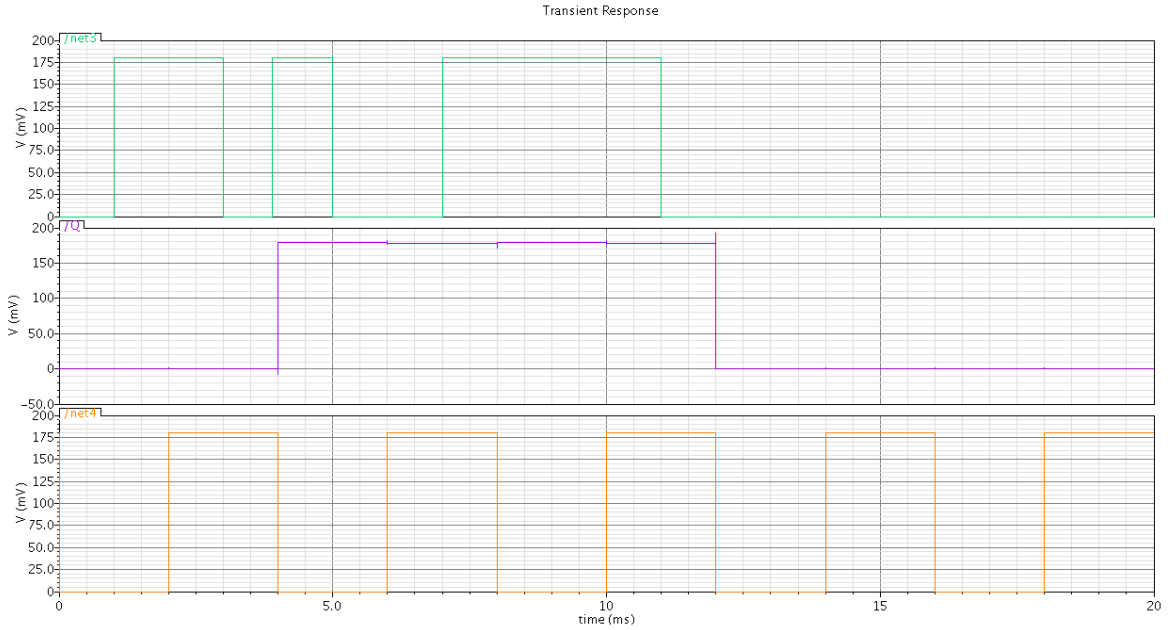
En D flip flop som skifter sin verdi på negativ klokkeflanke og som består av åtte NAND2 porter og en inverter er vist i figur 5.1.1.1. Det er en Master - Slave D flip flop.



Figur 5.1.1.1: Negativ klokkeflanke D flip flop[24]

Den har kun en D inngang (pluss klokke), og har både vanlig og invertert utgang. For å simulere på en flip flop krets må man definere en inngang som endrer seg slik at vi kan se om

kretsen kun skifter verdi ved den negative klokkeflanken. Dette er gjort i simulering på figur 5.1.1.2. Øverste grafen er D inngangen, midterste er utgangen og den nederste grafen er klokke signalet. Vi ser at den fungerer som forventet (med inngagnen som er definert), nemlig at utgangen kun skifter verdi ved negativ klokkeflanke og holder på sin verdi ellers uansett hvordan inngangen oppfører seg.



Figur 5.1.1.2: Simulering av en D flip flop som skifter verdi på negativ klokkeflanke (V_{DD} = 180 mV, T = 27 ^{o}C)

Temperatur (^{o}C)	Antall feil
-40	0
27	0
85	0

Tabell 5.1.1.1:Antall feil i D flip flop'en ved 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

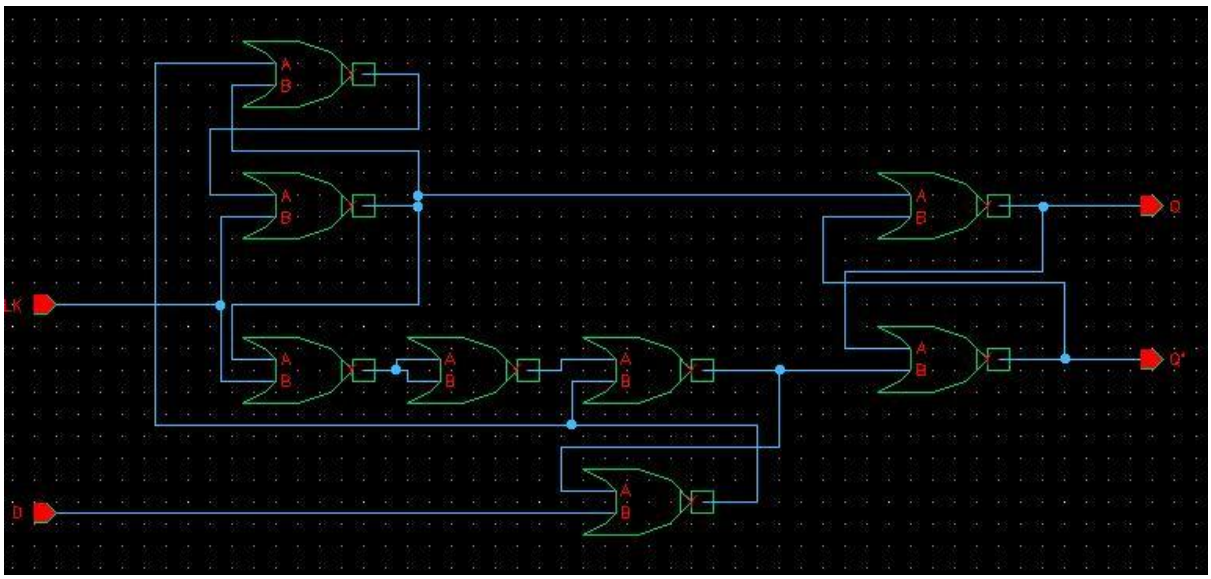
Tabellen ovenfor viser oss at D flip flop'en har 100% "yield" fra -40 ^{o}C til og med 85 ^{o}C når V_{DD} = 180 mV. Setup- og holdetidene for flip flop'en som gir 100% "yield" er gitt i neste tabell. Den viser oss at D flip flop'en har en "perfekt" holdtid. Setuptiden er høy ved lave temperaturer, men den synker etter hvert som temperaturen stider.

Temperatur (^{o}C)	Setuptid (ns)	Holdetid (ns)
-40	1160	0
27	90	0
85	22	0

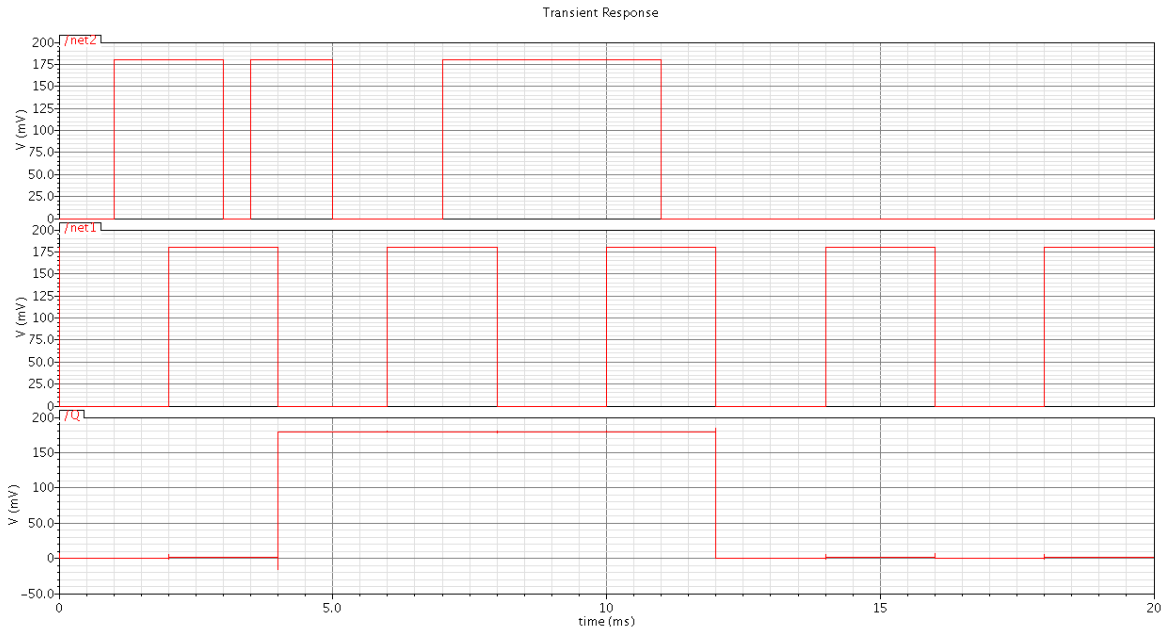
Tabell 5.1.1.2: Setup- og holdetider for D flip flop'en

5.1.2 Negativ klokkeflanke D Flip Flop med bare NOR2 porter

En D flip flop som skifter verdi ved negativ klokkeflanke og som kun består av NOR2 porter er gitt i figur 5.1.2.1, som har tatt utgangspunkt i [25]. Som vi ser så er NOR3 porten erstattet med tre NOR2 porter av grunner diskutert tidligere (kapittel 3.3). Figur 5.1.2.2 viser simuleringsresultatet for kretsen, og viser oss at D flip flop'en fungerer som den skal.



Figur 5.1.2.1: Skjematikk for en negativ klokkeflanke D flip flop



Figur 5.1.2.2: Simulering av en D flip flop som skifter verdi på negativ klokkeflanke (V_{DD} = 180 mV, T = 27 ^oC)

Temperatur (^oC)	Antall feil
-40	0
27	0
85	0

Tabell 5.1.2.1:Antall feil i D flip flop'en ved 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

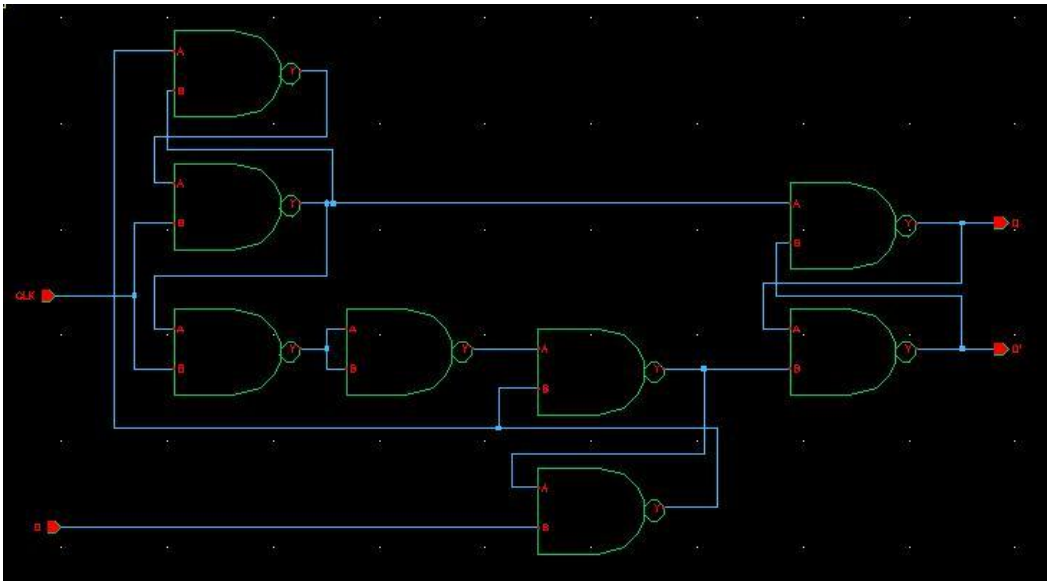
Tabellen ovenfor viser oss at flip flop'en har 100% "yield" fra -40 ^{o}C til og med 85 ^{o}C når V_{DD} = 180 mV. Setup- og holdetidene for flip flop'en som gir 100% "yield" er gitt i neste tabell. Den viser oss at holdtiden og setup tiden til D flip flop'en er høy ved lave temperaturer, men synker etter hvert som temperaturen stiger.

Temperatur (^oC)	Setuptid (ns)	Holdetid (ns)
-40	970	550
27	90	40
85	23	9

Tabell 5.1.2.2: Setup- og holdetider for D flip flop'en

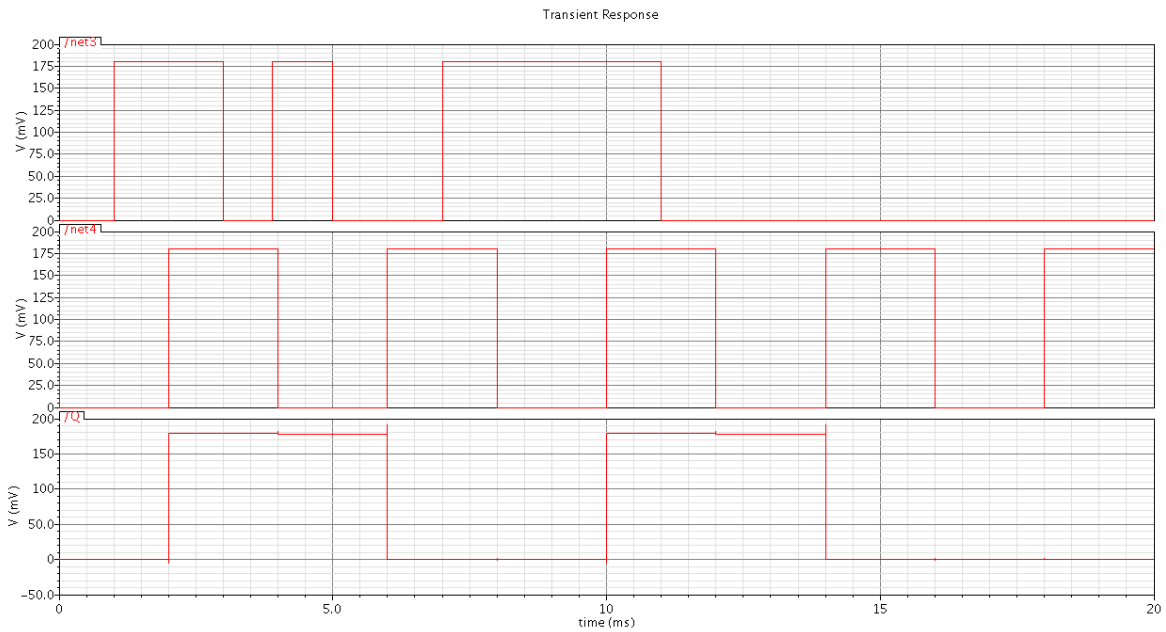
5.1.3 Positiv klokkeflanke D Flip Flop med bare NAND2 porter

En D flip flop som skifter verdi ved positiv klokkeflanke og som bare består av NAND2 porter er vist i figur 5.1.3.1. Denne kretsen tar utgangspunkt i [27], men NAND3 porten er erstattet med tre NAND2 porter. Disse tre NAND2 portene som erstatter NAND3 porten vil ha en forsinkelse på 12 δ + 2 Δ, i følge formel 3.3.1. Samme formel gir oss en forsinkelse på 9 δ + 3 Δ for NAND3 porten. Dette betyr at dekomposisjonen av NAND3 porten til NAND2 porter fører til en høyere forsinkelse.



Figur 5.1.3.1: Skjematikk for en positiv klokkeflanke D flip flop

Simulering av kretsen i figur 5.1.3.1 er gitt i figur 5.1.3.2. Den øverste grafen er inngangen til flip flopen, den midterste er klokka og den nederste er utgangen. Vi ser at kretsen fungerer som forventet og skifter kun verdi ved positive klokkeflanker.



Figur 5.1.3.2: Simulering av D flip flop'en i figur 5.1.3.2 (V_{DD} = 180 mV, T = 27 °C)

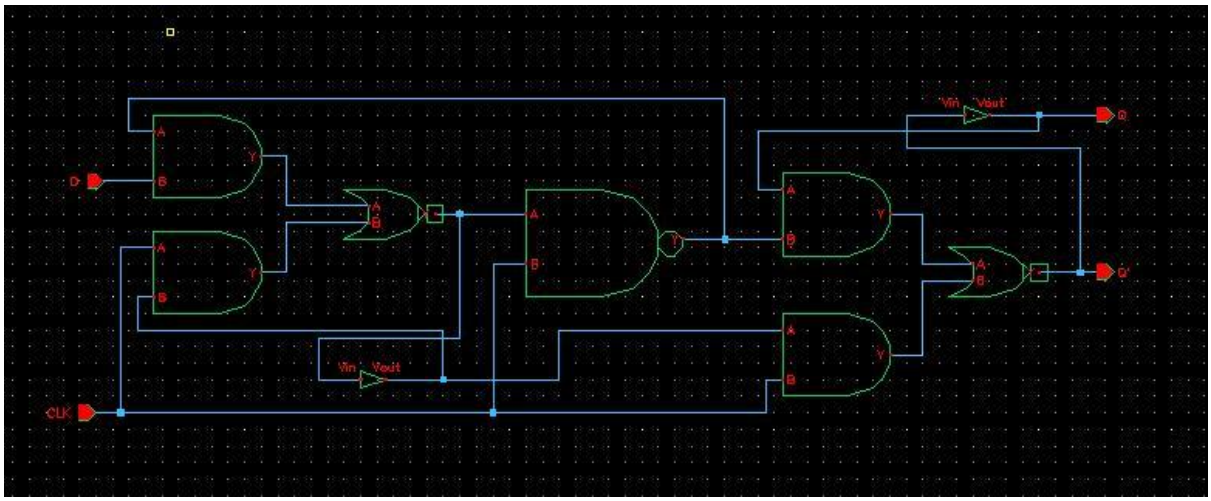
Temperatur (^{o}C)	Antall feil
-40	18
27	15
85	13

Tabell 5.1.3.1: Antall feil i D flip flop'en ved 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

Tabellen ovenfor viser oss at flip flopen har en veldig dårlig "yield" for de gitte temperaturene når V_{DD} = 180 mV. Feilene blir mindre og mindre etter hvert som temperaturen øker, men uten betydelig forbedring.

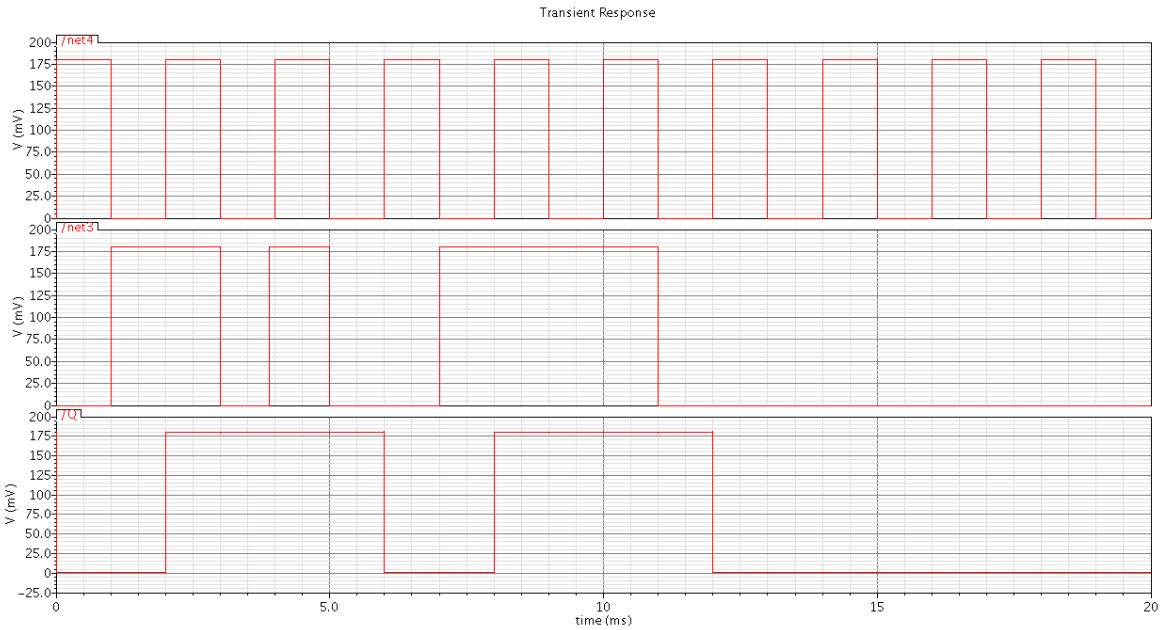
5.1.4 Race Free D Flip Flop med positiv klokkeflanke

En "Race Free" D flip flop er vist i figur 5.1.4.1. Dette er en flip flop som har blitt brukt i lav effekt biblioteker [21], og som har en effekt forbruk 2 – 4 ganger mindre enn flip flop'er i sterk inversjons biblioteker [28]. Vanlige Master – Slave flip flop'er bruker både klokke og en invertert klokke som kan føre til "hazarder" [29]. "Hazarder" er uønskede svinginger på utgangen av en krets, pga at forskjellige signal veier kan ha forskjellige forsinkelser som forårsaker midlertidige feil i utgangssignalet [30]. Med uønskede svingninger mener vi at kretsen kan havne i en uønskede tilstander. Derfor har race free'e flip floper blitt designet, for å få en hastighets uavhengig krets. En "race" oppstår når to eller flere logiske porter skal skifte verdi grunnet en endring i inngangen. Når signalveier har forskjellige forsinkelser, kan race'et føre til at interne variable skifter verdi ukontrollert. F. eks. hvis en internvariabel skal skifte verdi fra 00 til 11, kan forskjellige forsinkelser gjøre at variabelen skifter først til 10 så 00, eller 01 så 00 avhengig av hvilket av de 2 bit'ene som har størst forsinkelse [31].



Figur 5.1.4.1: Race free D flip flop med positiv klokkeflanke[28]

Simulering av den Race free'e D flip flopen er vist i figur 5.1.4.2. Vi ser av figuren at flip flop'en fungerer som den skal, og at den skifter verdi ved positiv klokkeflanke. Den øverste grafen er klokken, den midterste er inngangen og den nederste er utgangen.



Figur 5.1.4.2: Simulering av en Race free D flip flop med positiv klokkeflanke (V_{DD} = 180 mV T = 27 ^oC)

Temperatur (^oC)	Antall feil
-40	0
27	0
85	0

Tabell 5.1.4.1: Antall feil i Race Free D flip flop'en ved 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

Tabellen viser oss at når V_{DD} = 180 mV, så vil flip flop'en fungere for alle temperaturer fra -40 ^{o}C og til og med 85 ^{o}C.

5.2 Diskusjon av de ulike Flip Flop'ene

Vi har sett på 4 forskjellige flip flop'er, Master Slave D flip flop med NAND2 porter heretter kalt A, Negativ klokkeflanke D flip flop med bare NOR2 porter heretter kalt B, Positiv klokkeflanke D flip flop med bare NAND2 porter heretter kalt C og Race free D flip flop med positiv klokkeflanke heretter kalt D.

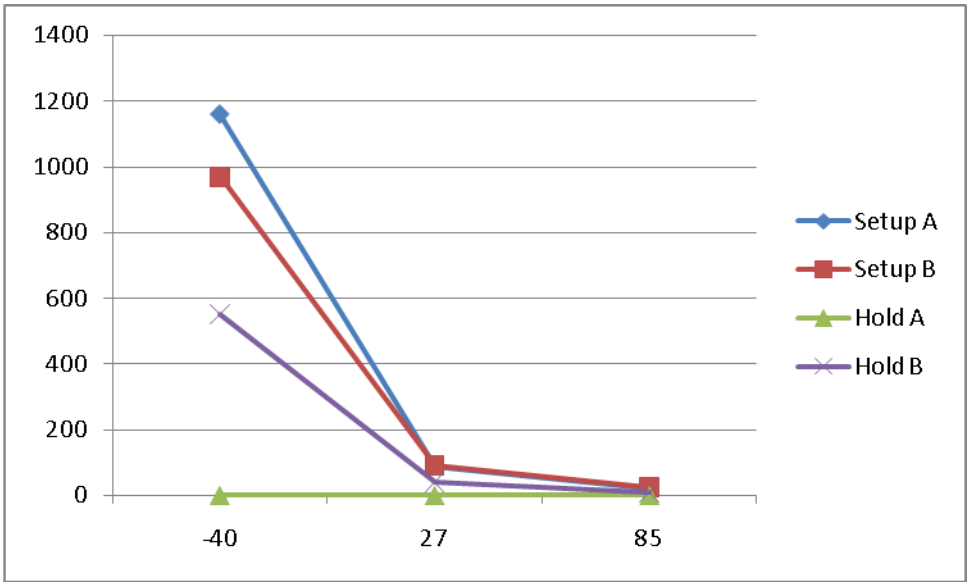
Flip flop C var den eneste av flip flop'ene som ikke hadde 100% "yield" i temperaturer fra -40 ^{o}C og opptil 85 ^{o}C med V_{DD} = 180 mV. Som vi ser av figuren til flip flop C, så består den kun av NAND2 porter. NAND3 porten i den opprinnelige kretsen er erstattet med tre NAND2 porter. Denne dekomposisjonen fører til at vi får en høyere forsinkelse, noe som kan føre til at kretsen ikke fungerer som den skal fordi NAND2 portene etter dekomposisjonen vil regne ut utgangssignalet med gale verdier hvis forsinkelsen blir for stor. Selv om det ikke er mange δ det er snakk om, så kan statistiske variasjoner og prosessvariasjoner gjøre at de blir større og problemet dukke opp.

Flip flop A som er en Master Slave struktur sliter tydeligvis ikke med race og hazarder. Dette skyldes at forsinkelsen i inverteren i kretsen er mindre enn NAND2 portene så slave'n får klokka si før inngangene. Dette kan bli et problem ved høyere klokkefrekvenser, men for vår klokkefrekvens (f = 250 kHz) så var det ikke noe problem.

Flip flop D virker ved en høyere klokkefrekvens enn de andre tre flip flopene. Hvis vi setter ned klokkefrekvensen til samme frekvens som for de andre flip flopene vil ikke den fungere fra start, men fungere etter hvert. Dette kan skyldes statistiske variasjoner og prosessvariasjoner som

gjør at alle nodene kan få forskjellige verdier. Dette betyr at initialbetingelsene blir feil, og vi vet ikke hvordan flip flop'en vil starte opp. En mulig løsning på dette problemet er å ha en "reset" utgang som kan nullstille flip flop'en ved oppstart.

For flip flopene A og B som vi fant setup- og holdtider for fant vi ut at både setuptiden og holdtiden blir bedre etter hvert som temperaturen øker. Dette ser man i grafen under. Vi ser at holdtidene alltid er lavere enn setuptidene. Holdtiden til flip flop A er 0 når den høyeste tilgjengelige oppløsningen er nano sekunder.



Figur 5.2.1: Sammenligning av setup- og holdetidene for A og B. (y akse i ns og x akse i ^{o}C)

For flip flop C og D har vi ikke noen verdier for setup- og holdetider. Dette er på grunn av at flip flop C ikke fungerer med 100 % "yield" med ønsket V_{DD} og innenfor det ønskede temperatur område med ønsket inngangsfrekvens. Flip flop D har heller ingen verdier og dette skyldes at vi må ha en høyere inngangsfrekvens for å få kretsen til å fungere fra start. Vi vil helst at den skal fungere på ønsket frekvens (250 kHz) for at den skal kunne integreres med frekvensdelerene i neste kapittel.

Kapittel 6

Simuleringsresultater for to klokkedelere

6.1 Hva er en frekvensdeler?

En frekvensdeler er en elektronisk krets som tar en inngangsfrekvens f_{inn} og deler på et tall n for å få en utgangsfrekvens f_{ut}.

$$f_{ut} = \frac{f_{inn}}{n} \qquad 6.1.1$$

Det er veldig enkelt å dele en frekvens med et heltall, og divisjon med et partall gir alltid 50 % ”duty cycle” på utgangen [32]. 50 % ”duty cycle” betyr, for en digitalpuls, at utgangspulsen er en perfekt firkantpuls (en puls som er *lav* like lenge som den er *høy*).

6.2 Simuleringsresultater for to klokkedelere

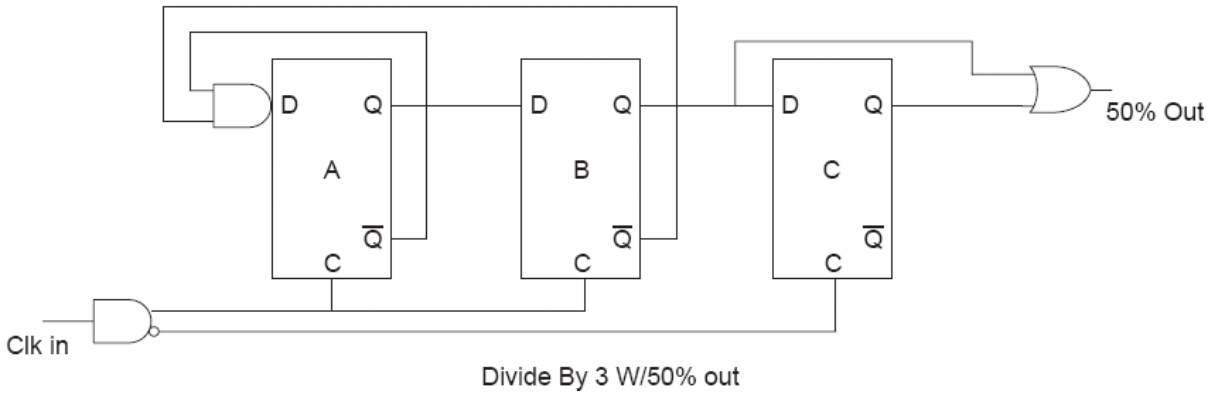
For å implementere en klokkedeler gjør vi først noen antagelser. Vi antar at klokken vi skal bruke på inngangen skal ha en f_{inn} = 250 kHz, at vi skal dele dette klokkesignalet på 3 og allikevel ha en 50 % ”duty cycle”. Dette skal vi implementere ved hjelp av to forskjellige frekvensdelere for så å sammenligne hvem av frekvensdelerene som er den beste for vårt formål. Fra og med nå vil vi kalle klokkedeler kretsene delt på 3 kretser.

6.2.1 ON Semiconductor versjonen

ON semiconductor versjonen av en delt på 3 krets med 50 % ”duty cycle” kan vi se på figur 6.2.1.1. Denne kretsen virker med en maksimal frekvens gitt av denne formelen [33]:

$$\frac{periode_{inn}}{2} = (forsinkelse\ flip\ flop\ B) + (Setuptid\ flip\ flop\ C) + (holdtid\ flip\ flop\ C) \qquad (6.2.1.1)$$

$$periode = 2 * ((forsinkelseflip\ flop\ B) + (Setuptid\ flip\ flop\ C) + (holdtid\ flip\ flop\ C)) \qquad (6.2.1.2)$$



Figur 6.2.1.1: ON Semiconductor versjonen av en delt på3 krets med 50 % "duty cycle" [33]

6.2.1.1 Master Slave D flip flop med NAND2 porter

Ved å bruke en Master Slave D flip flop med NAND2 porter kan vi regne oss fram til en maksimal forsinkelse ved å endre litt på formel 6.2.1.2.

$$periode = \frac{1}{frekvens} \qquad (6.2.1.1.1)$$

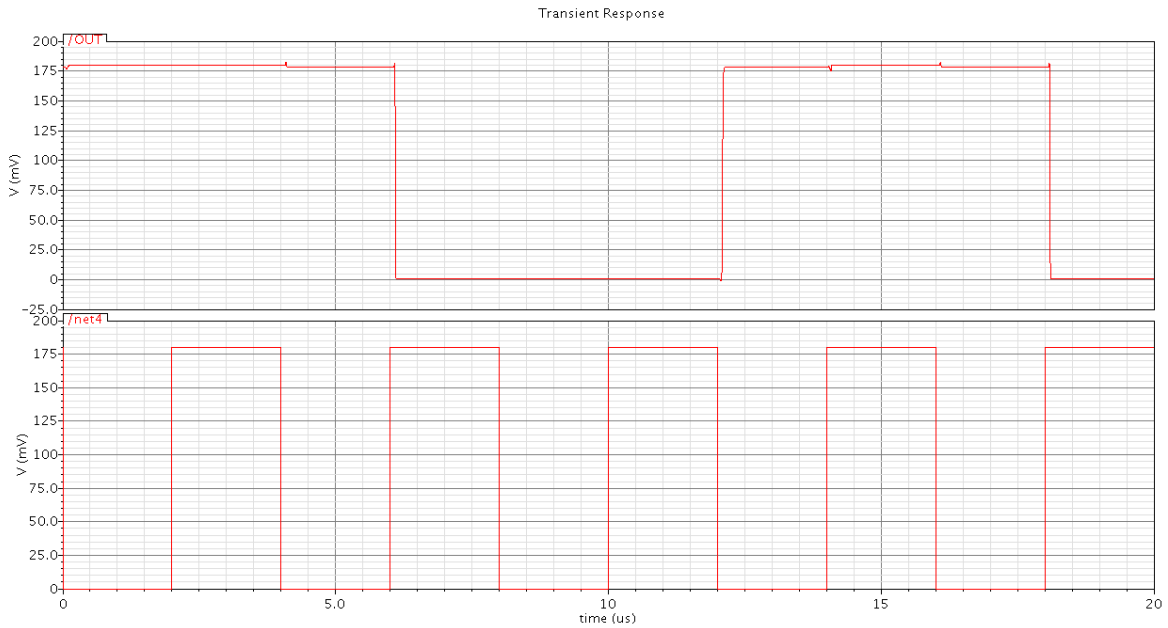
$$forsinkelse\ flip\ flop\ B = \left(\frac{1}{frekvens}\right) - (Setuptid\ flip\ flop\ C) - (holdtid\ flip\ flop\ C) \qquad (6.2.1.1.2)$$

En frekvens på 250 kHz gir oss en periode på 4 µs så ved å sette inn setup og holdtider for de forskjellige temperaturene kan vi regne oss fram til maksimal forsinkelse flip flop B i figur 6.2.1.1 kan ha for at kretsen skal fungere. Dette er gitt i følgende tabell.

Temperatur i ^{o}C	**Forsinkelse for flip flop B i µs**
-40	2,84
27	3,91
85	3,978

Tabell 6.2.1.1.1: Maksimal forsinkelse for flip flop B for at kretsen skal fungere

Selv om vi ikke har regnet ut forsinkelsen i en Master Slave D flip flop med NAND2 porter, kan vi regne med at de fungerer i og med at forsinkelsene som skal til for at kretsen ikke skal fungere er såpass mye. En forsinkelse på 3 µs betyr at vi har 1,5 periodersforsnikelse. Vi kan også simulere for å se at kretsen fungerer med vårt valg av flip flop'er, og dette kan vi se på figur 6.2.1.1.1. Vi ser at utgangssignalet har en T = 12 µs, og dette er nøyaktig tre ganger mer enn inngangssignalets periode på 4 µs.



Figur 6.2.1.1.1: Simulering av en delt på 3 krets (V_{DD} = 180mV T = 27 ^{o}C)

Temperatur i ^{o}C	Antall defekte fra start	Antall fungerende etter hvert	Total ”yield”
-40	30	0	0 %
27	15	15	100 %
85	15	15	100 %

Tabell 6.2.1.1.2: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

Tabellen ovenfor viser oss antall defekte frekvensdelere vi får med 30 MonteCarlo simuleringer. Som vi ser så fungerer ingen ved -40 ^{o}C, og ved 27 ^{o}C og 85 ^{o}C er det 15 stykker som fungerer. De resterende 15 fungerer ikke ved start, men fungerer etter hvert.

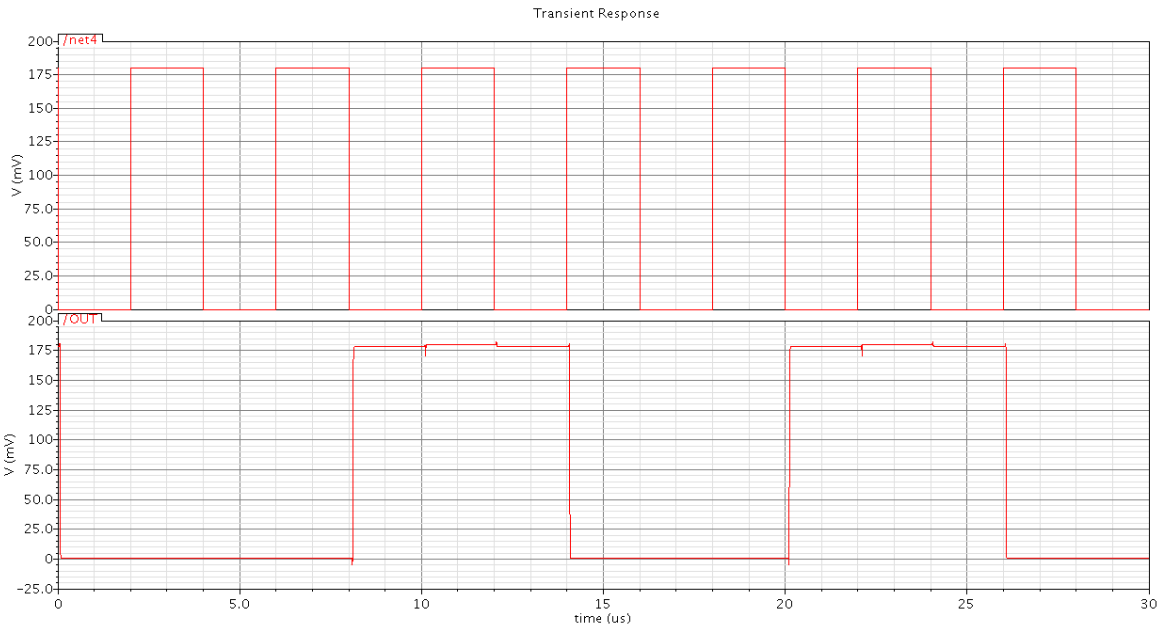
6.2.1.2 Negativ klokkeflanke D Flip Flop med bare NOR2 porter

Ved å bruke formel 6.2.1.1.2 kan vi regne oss fram til maksimal forsinkelse på flip flopen for at kretsen skal fungere. Forsinkelsen er vist i tabell 6.2.1.2.1.

Temperatur i ^{o}C	Forsinkelse for flip flop B i µs
-40	2,48
27	3,87
85	3,968

Tabell 6.2.1.2.1: Maksimal forsinkelse for flip flop B for at kretsen skal fungere

Som vi ser av tabellen er det ganske store forsinkelser som skal til for at kretsen ikke skal fungere, og vi kan regne med at den fungerer. Dette får vi også bekreftet i figur 6.2.1.2.1 som viser simuleringen av delt på 3 kretsen. Vi ser at den ikke fungerer fra starten, men at den fungerer etter hvert. (Med at den ikke fungerer fra start, menes at kretsen ikke deler klokken på 3 ved start).



Figur 6.2.1.2.1: Simulering av en delt på 3 krets (V_{DD} = 180mV T = 27 ^{o}C)

Temperatur i ^{o}C	Antall defekte fra start	Antall fungerende etter hvert	Total ”yield”
-40	30	0	0 %
27	10	10	100 %
85	14	14	100 %

Tabell 6.2.1.2.2: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

Tabellen ovenfor viser oss antallet defekte frekvensdelere vi får med 30 MonteCarlo simuleringer. Som vi ser så fungerer ingen ved -40 ^{o}C, ved 27 ^{o}C er det 20 stykker som fungerer fra start og ved 85 ^{o}C er det 16 stykker som fungerer fra start. De resterende 10 ved 27 ^{o}C og 14 ved 85 ^{o}C fungerer etter hvert.

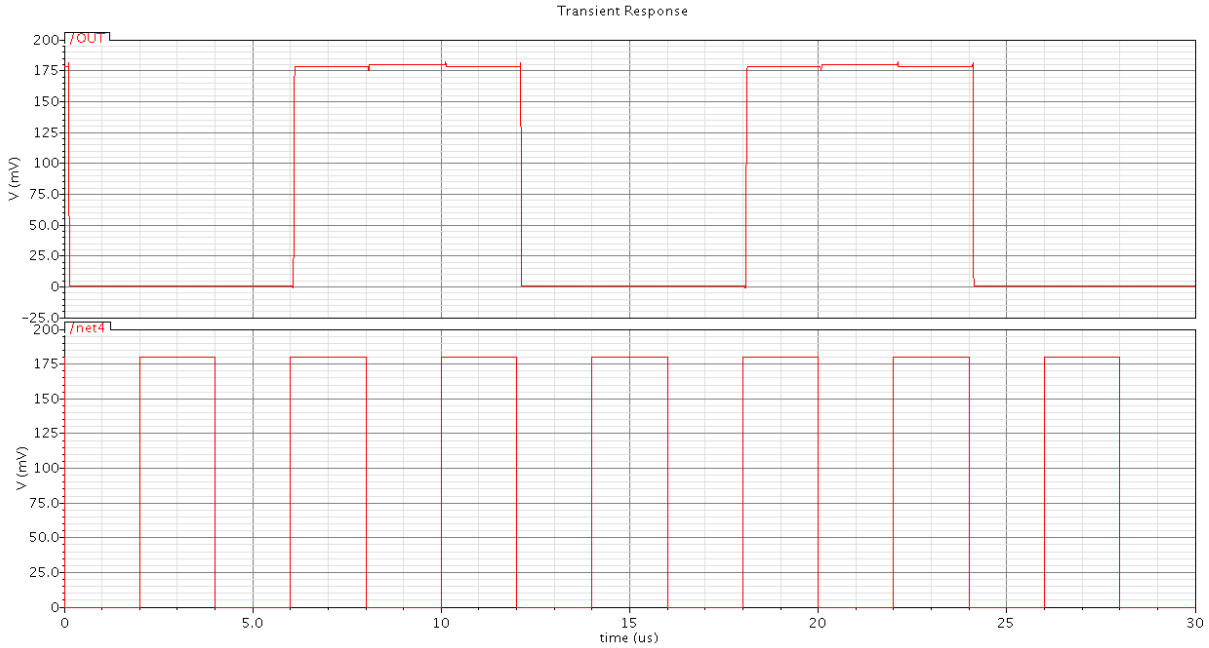
6.2.1.3 Positiv klokkeflanke D Flip Flop med bare NAND2 porter

Vi husker fra kapittel 5 at den positive klokkeflanke D flip flop’en med bare NAND2 porter ikke har 100 % ”yield”. Men vi kan allikevel prøve å lage en delt på 3 krets ved hjelp av disse flip flop’ene. Figur 6.2.1.3.1 viser oss simuleringen av en slik krets. Vi ser av figuren at kretsen fungerer, men hvis vi tar en MonteCarlo simulering får vi følgende tabell.

Temperatur i ^{o}C	Antall defekte fra start	Antall fungerende etter hvert	Total ”yield”
-40	30	0	0 %
27	25	18	67 %
85	21	14	67 %

Tabell 6.2.1.3.1: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

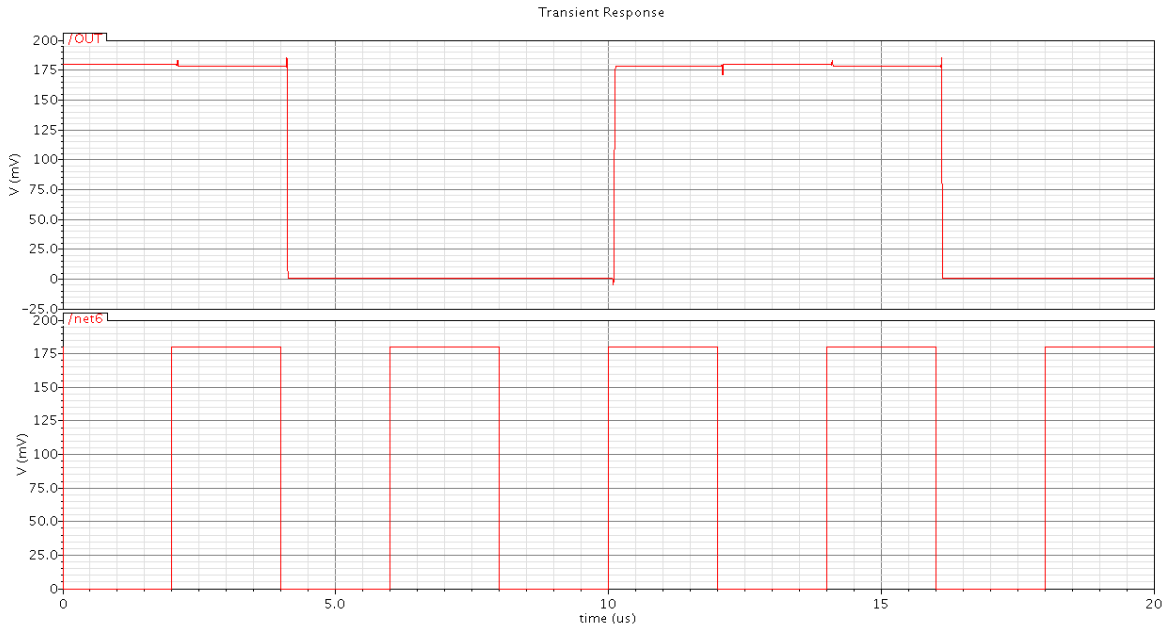
Tabellen viser oss at ved -40 ^{o}C så er det ingen av kretsene som fungerer, ved 27 ^{o}C er det 5 som fungerer fra start og ved 85 ^{o}C er det 9 som fungerer fra start. Av de resterende 25 ved 27 ^{o}C fungerer 18 etter hvert, og av de resterende 21 ved 85 ^{o}C fungerer 14 etter hvert. Dette betyr at vi har 7 defekte ved 27 ^{o}C og 7 ved 85 ^{o}C.



Figur 6.2.1.3.1: Simulering av en delt på 3 krets (V_{DD} = 180mV T = 27 ^{o}C)

6.2.1.4 Race Free D Flip Flop med positiv klokkeflanke

Simuleringen av en delt på 3 krets laget med Race free D flip flop'er med positiv klokkeflanke kan vi se i figur 6.2.1.4.1. Figuren viser oss at kretsen ikke fungerer fra start, men at den fungerer etter den "ukorekte" starten.



Figur 6.2.1.4.1: Simulering av en delt på 3 krets (V_{DD} = 180mV T = 27 ^{o}C)

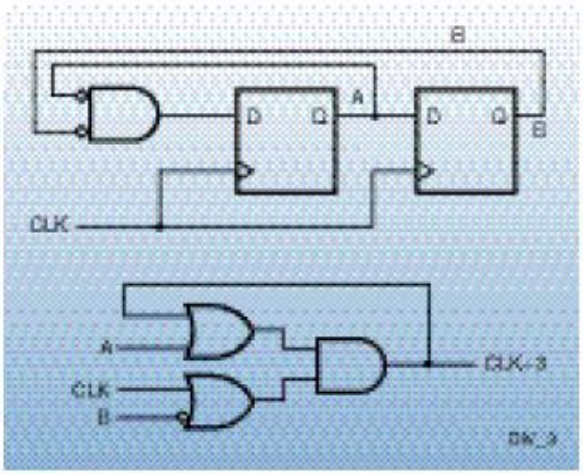
MonteCarlo simuleringer av kretsen gir oss tabell 6.2.1.4.1. Vi ser i tabellen at ved alle temperaturer i mellom -40 ^{o}C og 85 ^{o}C så er det ingen av kretsene som fungerer fra start, men ved 27 ^{o}C og 85 ^{o}C fungerer alle etter hvert.

Temperatur i ^{o}C	Antall defekte fra start	Antall fungerende etter hvert	Total ”yield”
-40	30	0	0 %
27	30	30	100 %
85	30	30	100 %

Tabell 6.2.1.4.1: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

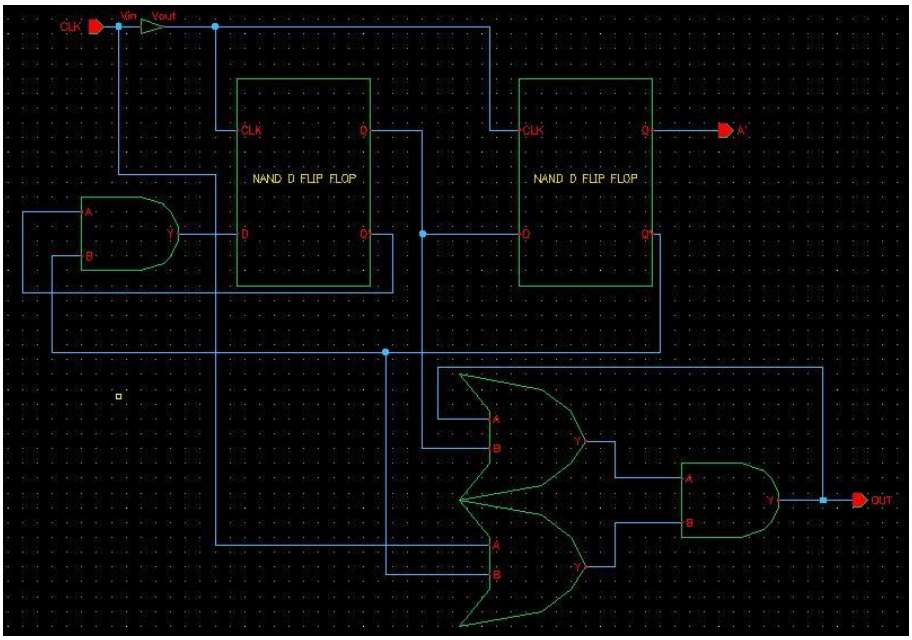
6.2.2 Xilinx versjonen

Xilinx versjonen av en delt på 3 krets kan vi se i figur 6.2.2.1. Sannhetstabellen til denne kretsen ser vi også på figuren. Som vi kan lese av sannhetstabellen er ikke denne kretsen designet for å fungere fra start, men etter hvert. De første 2 utgangene blir feil i følge vår antagelse om at den skal dele på 3 fra start. For hadde den skullet fungere fra start, så skulle de tre første inngangene vært *lave* (eller *høye*). Vi ser også i fra figuren at kretsen krever flip flop’er som skifter verdi på positiv klokkeflanke, så for å bruke den på flip flop’er som skifter verdi ved negativ flanke må den endres på litt. Denne endringen er ikke vanskeligere enn å invertere klokkesignalet for flip flop’en, mens den logiske porten som bruker klokke må ha den ikke inverterte klokken. Dette er vist i figur 6.2.2.2.



CLK	A	B	OUT
H	0	0	0
L	0	0	0
H	1	0	1
L	1	0	1
H	0	1	1
L	0	1	0

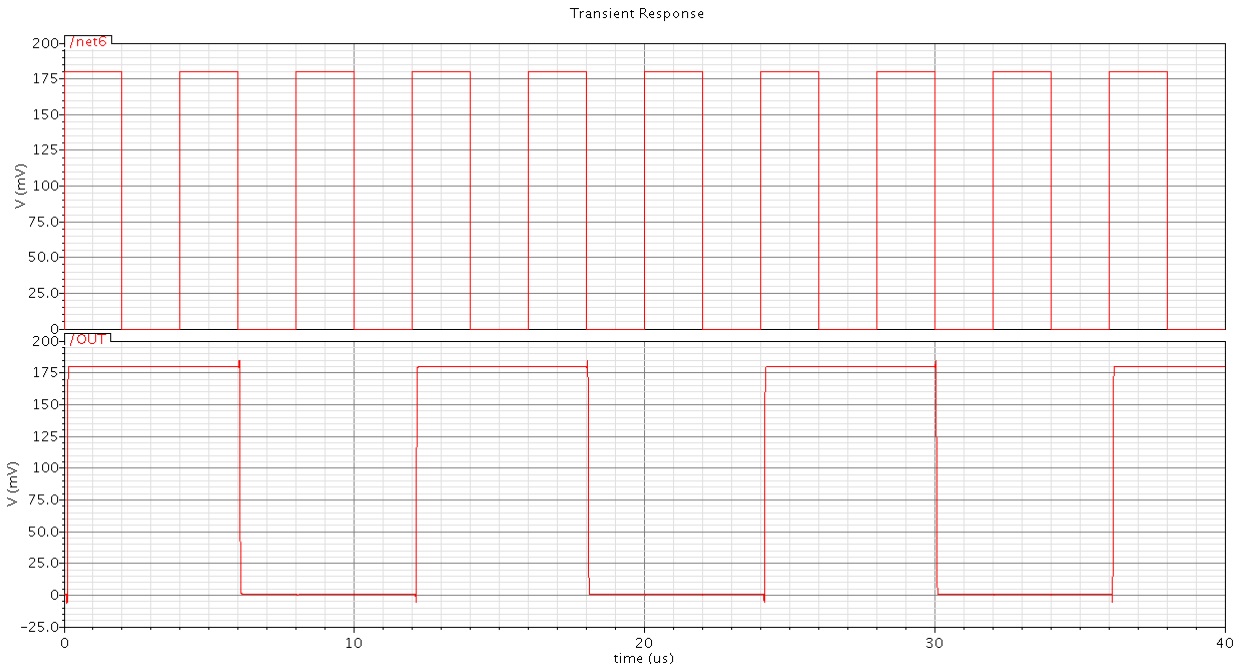
Figur 6.2.2.1: Xilinx versjonen av en delt på 3 krets med 50 % ”duty cycle” med sannhetstabell [32]



Figur 6.2.2.2: Xilinx delt på 3 krets for negative klokkeflanker.

6.2.2.1 Master Slave D flip flop med NAND2 porter

Simuleringen av Xilinx delt på 3 kretsen med modifikasjoner for å bruke flip flop'er med negative klokkeflanker ser vi i figur 6.2.2.1.1. Vi ser at kretsen ikke fungerer som sannhetstabellen sier at den skal fungere, men fungerer helt fra start (med dette så menes det at kretsen deler på 3 fra start).



Figur 6.2.2.1.1: Simulering av Xilinx delt på 3 krets for negative klokkeflanker (V_{DD} = 180 mV T = 27 ^{o}C)

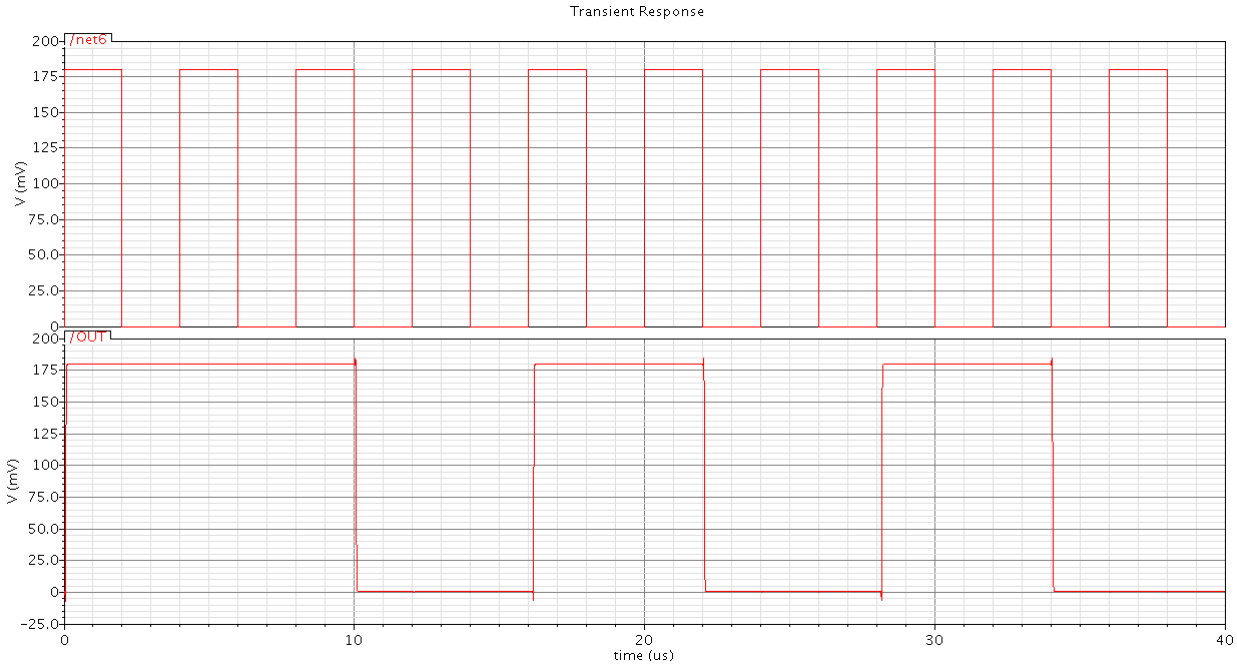
Temperatur i ^{o}C	Antall defekte i følge sannhetstabell	Antall fungerende etter hvert
-40	30	0
27	24	24
85	25	25

Tabell 6.2.2.1.1: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

Tabellen ovenfor viser oss antall defekte frekvensdelere vi får med 30 MonteCarlo simuleringer. Som vi ser så fungerer ingen ved -40 ^{o}C, ved 27 ^{o}C er det 6 stykker som fungerer fra start og ved 85 ^{o}C er det 5 stykker som fungerer fra start. De resterende 24 ved 27 ^{o}C og 25 ved 85 ^{o}C fungerer etter hvert.

6.2.2.2 Negativ klokkeflanke D Flip Flop med bare NOR2 porter

Simuleringen av Xilinx delt på 3 kretsen med modifikasjoner for å bruke flip flop'er med negative klokkeflanker ser vi i figur 6.2.2.2.1. Vi ser at kretsen ikke fungerer som sannhetstabellen sier at den skal fungere, men begynner å dele på 3 etter en liten stund (10 µs).



Figur 6.2.2.2.1: Simulering av Xilinx delt på 3 krets for negative klokkeflanker (V_{DD} = 180 mV T = 27 ^{o}C)

Temperatur i ^{o}C	Antall defekte i følge sannhetstabell	Antall fungerende etter hvert
-40	30	0
27	12	12
85	6	6

Tabell 6.2.2.2.1: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

Tabellen ovenfor viser oss antall defekte frekvensdelere vi får med 30 MonteCarlo simuleringer. Som vi ser så fungerer ingen ved -40 ^{o}C, ved 27 ^{o}C er det 18 stykker som fungerer fra start og ved 85 ^{o}C er det 24 stykker som fungerer fra start. De resterende 12 ved 27 ^{o}C og 6 ved 85 ^{o}C fungerer etter hvert.

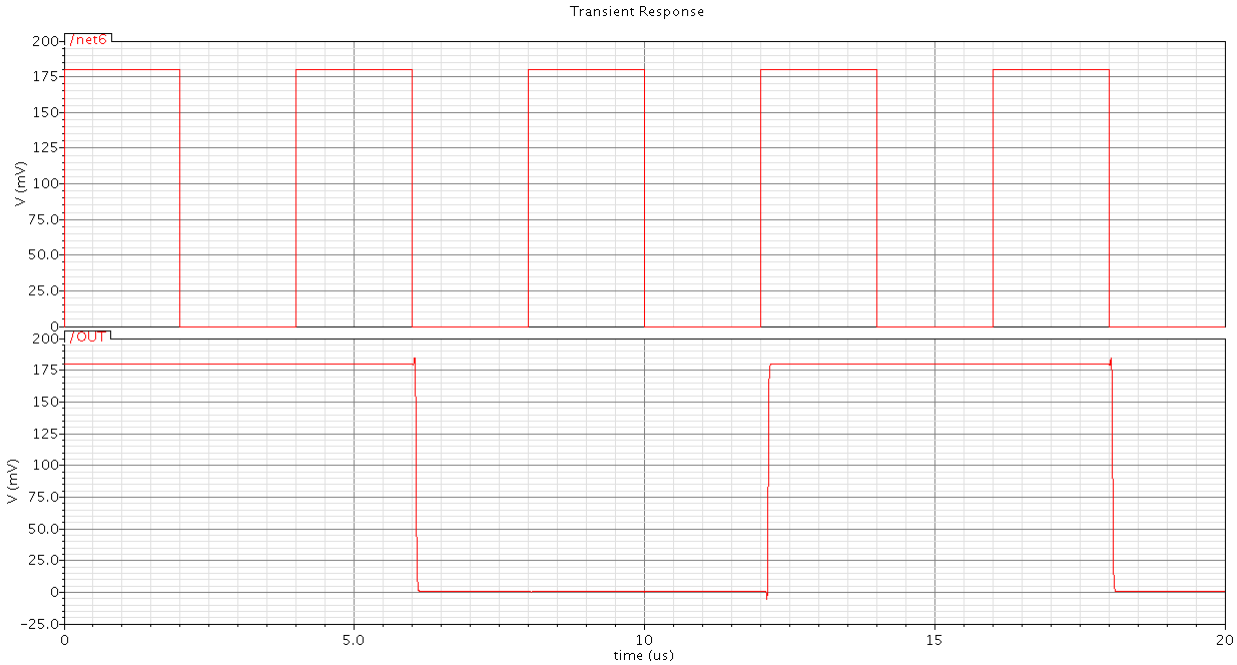
6.2.2.3 Positiv klokkeflanke D Flip Flop med bare NAND2 porter

Simuleringen av Xilinx delt på 3 kretsen ser vi i figur 6.2.2.3.1. Vi ser at kretsen ikke fungerer som sannhetstabellen sier at den skal fungere, men deler på 3 helt fra starten av.

Temperatur i ^{o}C	Antall defekte i følge sannhetstabell	Antall fungerende etter hvert
-40	30	0
27	27	21
85	22	16

Tabell 6.2.2.3.1: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

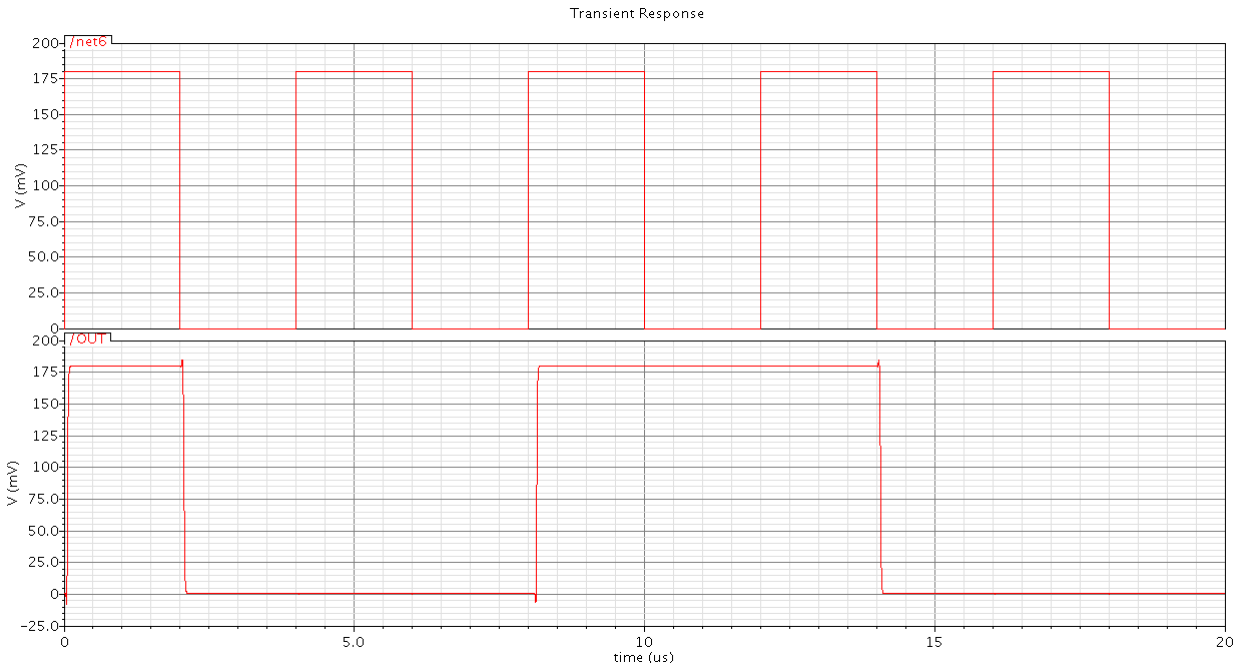
Tabellen viser oss at ved -40 ^{o}C så er det ingen av kretsene som fungerer, ved 27 ^{o}C er det 3 som fungerer fra start og ved 85 ^{o}C er det 8 som fungerer fra start. Av de resterende 27 ved 27 ^{o}C fungerer 21 etter hvert, og av de resterende 22 ved 85 ^{o}C fungerer 16 etter hvert. Dette betyr at vi har 6 defekte ved 27 ^{o}C og 6 ved 85 ^{o}C (80 % ”yield”).



Figur 6.2.2.3.1: Simulering av Xilinx delt på 3 krets for negative klokkeflanker (V_{DD} = 180 mV T = 27 ^{o}C)

6.2.2.4 Race Free D Flip Flop med positiv klokkeflanke

Simuleringen av Xilinx delt på 3 kretsen ser vi i figur 6.2.2.4.1. Vi ser at kretsen ikke fungerer som sannhetstabellen sier at den skal fungere, men fungerer etter hvert.



Figur 6.2.2.4.1: Simulering av Xilinx delt på 3 krets for negative klokkeflanker (V_{DD} = 180 mV T = 27 ^{o}C)

Temperatur i ^{o}C	Antall defekte i følge sannhetstabell	Antall fungerende etter hvert
-40	30	0
27	22	22
85	23	23

Tabell 6.2.2.4.1: Antall feil i delt på 3 kretsen med 30 MonteCarlo simuleringer (V_{DD} = 180 mV)

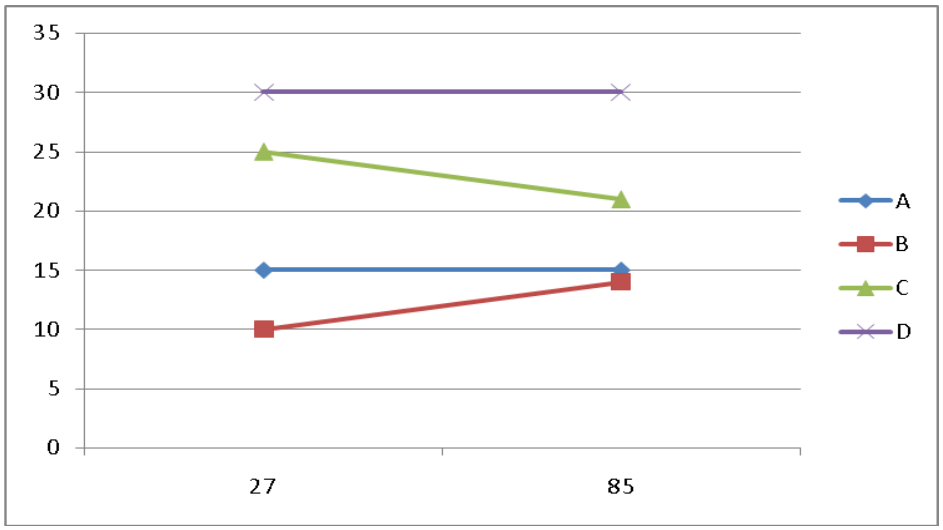
Tabellen ovenfor viser oss antall defekte frekvensdelere vi får med 30 MonteCarlo simuleringer. Som vi ser så fungerer ingen ved -40 ^{o}C, ved 27 ^{o}C er det 8 stykker som fungerer fra start og ved 85 ^{o}C er det 7 stykker som fungerer fra start. De resterende 22 ved 27 ^{o}C og 23 ved 85 ^{o}C fungerer etter hvert.

6.3 Diskusjon av 2 ulike frekvensdelere med 4 forskjellige flip flop'er

I dette kapittelet så har vi sett på implementering av to frekvensdelere, en fra Xilinx og en fra ON Semiconductor, og implementert med de fire forskjellige flip flop'ene vi så på i kapittel 5. De fire forskjellige flip flop'ene er: Master Slave D flip flop med NAND2 porter heretter kalt A, Negativ klokkeflanke D flip flop med bare NOR2 porter heretter kalt B, Positiv klokkeflanke D flip flop med bare NAND2 porter heretter kalt C og Race free D flip flop med positiv klokkeflanke heretter kalt D.

ON Semiconductor versjonen som vi så på først fungerte med 100 % "yield" med alle flip flop'ene ved 27 ^{o}C og 85 ^{o}C unntatt for flip flop C. Ved -40 ^{o}C var det ingen av kretsene som fungerte, alle hadde 0 % "yield". Det var ikke overaskende at klokkedeleren som er realisert med flip flop C ikke hadde 100 % "yield", siden C flip flop'en ikke hadde 100 % "yield" selv en gang.

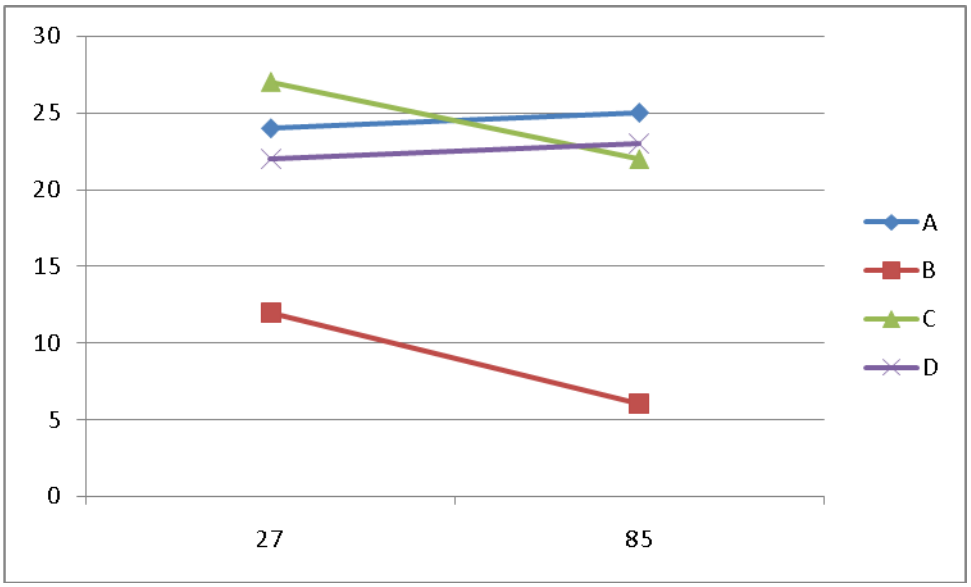
Selvom flip flop A, B og D hadde 100 % "yield" ved 27 ^{o}C og 85 ^{o}C ved V_{DD} = 180 mV, gjelder ikke den "yield"en fra oppstart. De trengte tid for å komme i gang, og som vi ser av figur 6.3.1 er det flip flop B som har best "yield" fra oppstart. Flip flop D har ingen fungerende kretser fra oppstart, men alle vil fungere etter hvert. At kretsene trenger tid for å komme i gang skyldes statistiske- og prosess variasjoner som fører til at flip flop'ene har feil oppstarts tilstander, men etter hvert som tiden går tilpasser de seg inngangene og begynner å fungere. For flip flop C vil 23,3 % av kretsene ikke fungere selv etter oppstart fasen.



Figur 6.3.1: Antall feil ved oppstart forON Semiconductor frekvensdeleren med de forskjellige flip flop'ene

Xilinx versjonen som vi så på sist fungerte med 100 % ”yield” med alle flip flop’ene ved 27 ^{o}C og 85 ^{o}C unntatt for flip flop C. Ved -40 ^{o}C var det ingen av kretsene som fungerte, alle hadde 0 % ”yield”. Det var ikke overaskende at klokkedeleren som er realisert med flip flop C ikke hadde 100 % ”yield”, siden C flip flop’en ikke hadde 100 % ”yield” selv en gang.

Selvom flip flop A, B og D hadde 100 % “yield” ved 27 ^{o}C og 85 ^{o}C ved V_{DD} = 180 mV, gjelder ikke den ”yield”en fra oppstart. De trengte tid for å komme i gang, og som vi ser av figur 6.3.2 er det flip flop B som har best ”yield” fra oppstart. Flip flop D har ingen fungerende kretser fra oppstart, men alle vil fungere etter hvert. At kretsene trenger tid for å komme i gang skyldes statistiske- og prosessvariasjoner som fører til at flip flop’ene har feil oppstarts tilstander, men etter hvert som tiden går tilpasser de seg inngangene og begynner å fungere. For flip flop C vil 20 % av kretsene ikke fungere selv etter oppstart fasen.



Figur 6.3.2: Antall feil ved oppstart for Xilinx frekvensdeleren med de forskjellige flip flop’ene

Både på On Semiconductor sin versjon av klokkedeleren og på Xilinx sin versjon, så er det flip flop B som fungerer best ved oppstart. Det virker også som om Xilinx versjonen er bedre med tanke på at det er en større andel kretser som fungerer ved oppstart med flip flop B. Med flip flop D er det også flere kretser som fungerer ved oppstart. Med flip flop C opplever vi at vi får en større andel av kretser som ikke fungerer ved oppstart, men vi får ned defekt andelen fra 23,3 % til 20 %. Flip flop A fungerer best ved oppstart med ON Semiconductor kretsen.

Kapittel 7

Diskusjon og Konklusjon

7.1 Oppsummering av kap. 1 - 6

- Å redusere forsyningsspenningen er det mest direkte og dramatiske enkeltgrepet som kan gjøres for å redusere effekt forbruket i elektroniske kretser.
- CMOS logiske byggeblokker er konstruert for symmetri i svitsjepunktet for en V_{DD} på 180 mV ved romtemperatur, med tanke på å benyttesi kombinatoriske kretser og minneelementer, for generelle digitale kretser implementert i 90 nm CMOS teknologi. Statistiske (”MonteCarlo”-) simuleringer er foretatt og indikerer at inverteren fungerer i temperatur området fra -40 ^{o}C til og med 85 ^{o}C, ned til en forsyningsspenning på 100 mV. For en NAND2 porten som er konstruert fungerte den for 99% (99 % ”yield” av 100 ”run”) i samme temperatur område. NOR2 porten som ble konstruert fungerte i samtlige 100 ”run” i nevnte temperatur område.
- NAND2 porten ble benyttet til å realisere mer komplekse porter som f.eks XOR, AND2 og OR2 med akseptable resultater for vår bruksområde (ALU).
- Redundans for defekt-toleranse er tatt i bruk for NAND2, NOR2 og inverterm der drevne utganger er kortsluttet. Ved to invertere i pararell (R = 2) kunne forsyningsspenningen reduseres til 80 mV for T = -40 ^{o}C og T = 27 ^{o}C samtidig som 100% ”yield” ble beholdt, mens den for 85 ^{o}C hadde 99 % ”yield”. R = 3,4 tillot en reduksjon av V_{DD} til 75 mV, mens R = 5 førte til en reduksjon av forsyningspenningen til 70 mV med 100 % ”yield” over hele temperatur området. Tilsvarende for NAND2 port ble V_{DD} = 140 mV (99% ”yield” i temperatur område) for R = 2, men for R = 3,4,5 ble V_{DD} = 120 mV med 100 % ”yield”. NOR2 porten hadde 100 % ”yield” for T = -40 ^{o}C og T = 85 ^{o}C med en V_{DD} = 120 mV, men den for T = 85 ^{o}C hadde 97 % ”yield” for R = 2. For R = 3,4 fungerte den ved 120 mV og for R = 5 ved 110 mV med en ”yield” på 100 % i hele temperatur området.
- Økning av transistor dimensjoner, inkludert dobling av gatelengde til L = 200 nm, uten redundans, ga nedre V_{DD} på 70 mV for INVERTER, på tvers av temperatur og prosessvariasjon med 100 % ”yield”. For NAND2- og NOR2 – porten ble tilsvarende minimum V_{DD} redusert til henholdsvis 150 mV og 120 mV.
- 1 – bit ALU er implementer ved hjelp av Full – Adder, dekoder, MUX, AND2, OR2 og XOR som er realisert med NAND2 porter og INVERTER’ere, for L = 0,1 μm. For tidligere omtalte temperaturområde fungerer 100 % av ALU’ene for V_{DD} = 250 mV. Med en forsyningsspenning på 200 mV får vi 100 % ”yield” ved T = 27 ^{o}C og 85 ^{o}C, mens den for T = -40 ^{o}C har en ”yield” på 27 %. For samme ALU med L = 0,2 μm fungerte den for alle ”runs” ved V_{DD} = 200 mV i temperaturområde. For V_{DD} = 150 mV fungerte den for T = 27 ^{o}C og 85 ^{o}C mens den for -40 ^{o}C hadde 0 % ”yield”.
- Fire typer D-Flip-Flop’s har blitt konstruert og MonteCarlo simuleringer foretatt for V_{DD} = 180 mV og en operasjonsfrekvens på 250 kHz. Disse er Master Slave D flip flop med NAND2 porter (type A), Negativ klokkeflanke D flip flop med bare NOR2 porter

(type B), Positiv klokkeflanke D flip flop med bare NAND2 porter (type C) og Race Free D flip flop med positiv klokkeflanke (type D). Alle bortsett fra ”type C” fungerte i temperaturområde -40 ^{o}C- 85 ^{o}C. I henhold til simuleringer har type A og type B så å si lik setuptider for T = 27 ^{o}C (begge har 90 ns) og T = 85 ^{o}C (type A har 22 ns og type B 23 ns), mens for T = -40 ^{o}C har type A en setuptid på 1,16µs og type B har 0,97µs. Flip flop A har 0 ns holdtid i temperatur området, mens flip flop B har holdtid på 550 ns, 40 ns og 9 ns for henholdsvis T = -40 ^{o}C, 27 ^{o}C og 85 ^{o}C.

- To ulike topologier for frekvensdelere har blitt konstruert med de fire forskjellige flip flop implmentasjonene, en ”ON Semiconductor type” og en ”Xilinx type”. Førstnevnte variant fungerte i henhold til simuleringer dårligst for flip flop C (77% ”yield” for både 27 ^{o}C og 85 ^{o}C). Ved -40 ^{o}C fungerte ingen av implementasjonene, mens flip flop A, B og D fungerte for 27 og 85 ^{o}C, for V_{DD} = 180 mV. Det verdt å nevne at ingen av flip flop’ene hadde 100 % ”yield” fra oppstart, men at alle fikk det etter en forsinkelse. Flip flop B viste mest robusthet under MonteCarlo simuleringene, med høyest ”yield” ved oppstart (67 % for 27 ^{o}C og 53% for 85 ^{o}C). For Xilinx implementasjonen var det på tilsvarende vis flip flop C som fungerte dårligst for 27 ^{o}C og 85 ^{o}C (80 % ”yield får begge temperaturene), for V_{DD} = 180 mV. Flip flop B hadde igjen høyest ”yield” ved oppstart (60 % ved 27 ^{o}C og 80 % ved 85 ^{o}C). Heller ikke her fungerte kretsene for -40 ^{o}C.

7.2 Diskusjon og konklusjon

Først og fremst så er det viktig å nevne at all implementasjon er gjort i Cadence 90 nm CMOS teknologi med standard terskel transistorer.

I denne oppgaven har vi implemetert logiske porter, ALU og minneelementer (D flip flop’er), som i prinsippet er nok til å implementere enhver FSM eller datamaskin, med ultra lav forsyningsspenning. Vi har sett at dette er fullt mulig å få til med 100 % ”yield” for de forutsetningene som er tatt (V_{DD} = 180 mV, f = 250 kHz, T fra -40 ^{o}C til 85 ^{o}C).

INVERTER’en er den logiske porten som fungerer ved lavest V_{DD} enn det den bli designet for (fungerte med en V_{DD} = 100 mV selv om den var dimensjonert slik at vi fikk symmetri om svitsjepunktet med V_{DD} = 180mV). Dette skyldes at det er den enkleste porten å realisere, med kun en transistor i ”opptrekket” og en i ”nedtrekket” som også resulterer i mindre areal og mindre effektforbruk. At den inneholder få transistorer fører jo som vi vet til mindre parasittiske kapasitanser og mindre motstander i kretsen.

Som vi så fungerte NOR2 porten bedre enn NAND2 porten. Grunnen til dette er at ved ”matching” om symmetripunktet vil arealet til NOR2 porten være større enn arealet til NAND2 porten. En større WL produkt vil føre til en økt robusthet, men ”prisen” man må betale for det er igjen økt areal og økt effektforbruk. Dette betyr at selv om NOR2 porten generelt sett fungerte bedre enn NAND2 porten så er det ikke ”feil” å bruke NAND2 porten til å realisere mer komplekse porter som AND2, OR2 og XOR, og i hvert fall ikke når vi implementerer disse ”mer komplekse portene” til å operere med en høyere V_{DD} (200 mV).

Redundans i logiske porter vil gi oss mer pålitelighet enn sine motstykker uten redundans, men igjen vil areal forbruket komme i fokus. En redundans på 2 vil doble arealet, en redundans på 3 vil tre doble arealet også videre, så her er det viktig å veie opp fordelene med ulempene før man tar i bruk det. Hvis man har en krets som skal fungere med en V_{DD} = 200 mV er det kanskje ikke noe å tjene på å bruke redundans, mens for V_{DD} = 140 mV er det noe å vurdere. Det er hele tiden viktig å se på problemstillingen sin, og ikke bruke mer arealet enn nødvendig for at applikasjonen skal

fungere. Det går også ann å bruke porter som har dobbel gatelengde. Vi fant ut at en dobling av gatelengden er bedre enn redundans 2 for samtlige av de tre portene vi testet. For en inverter vil L = 0,2 µm føre til at inverteren fungerer med en V_{DD} = 70 mV med 100 % "yield", mens en redundans på 2 vil gi en forsyningsspenning på 100 mV som er det samme som ved L = 0,1µm. For at vi skal få inverteren til å fungere med en V_{DD} = 70 mV med redundans, må vi ha redundans 5. Dette betyr at for en inverter så er det bedre å doble gatelengden enn å ta i bruk redundans.

ALU'en vi designet viser oss igjen at L = 0,2 µm er bedre enn L = 0,1 µm, og dette skulle bare mangle i og med at vi ikke kan forvente at ALU'en skal fungere bedre enn enkelt komponentene den er lagd av. ALU'en viser også oss at lav spenning CMOS ikke egner seg for høyhastighets design. Selv om ALU'en har en frekvens på 1 MHZ, som ikke er så veldig høyt, ser vi at den begynner å merkbare forsinkelser på utgangen. Dette er ikke noe problem for de "høye temperaturene", men for -40 ^{o}C vil dette føre til at vi får veldig dårlig "yield" for ALU'en designet med L = 0,1µm (27% for V_{DD} = 200 mV). Dette problemet kan enten løses ved å bruke større transistorer for å øke WL produktet så de statistiske variasjonene blir mindre, eller ved å øke forsyningsspenningen. Ved å øke V_{DD} får vi 100 % "yield" for ALU'en ved 250 mV for hele temperatur område som er fra -40 ^{o}C til og med 85 ^{o}C, og ved å doble gatelengden for vi samme effekt ved 200 mV over samme temperatur området. I begge tilfeller så må vi øke effektforbruket, mens arealet ikke blir berørt av å øke V_{DD}. Dette viser oss igjen at det er viktig å identifisere problemet sitt korrekt, og å ikke "ofre" for mye for å operere i ønsket bruksområde.

Flip flop'ene som vi implementerte viser oss at det er fullt mulig å lage minneelementer ved hjelp av logiske porter som har en lav forsyningsspenning. Det var kun en flip flop som ikke hadde 100 % "yield" og det var flip flop C (Positiv klokkeflanke D flip flop med bare NAND2 porter). Som vi har sett vil den kretsen ha en høyere forsinkelse på grunn av dekomposisjonen som er utført for å erstatte en NAND3 port med NAND2 porter. Det er også verdt å ta med seg at NAND2 portene som flip flop'en er bygget opp av ikke hadde 100 % "yield" ved 180 mV, som er ønsket forsyningsspenning for flip flop'en. Ved å da koble sammen flere NAND2 porter vil prosessvariasjonene og de statistiske variasjonene gjøre at vi får en høyere andel av 'defekte' flip flop'er. Master Slave D flip flop med NAND2 porter (flip flop A) er en annen flip flop som også er designet ved hjelp av stort sett NAND2 porter, men denne ser vi fungerer med ønsket V_{DD} for ønsket temperatur område. Dette viser oss at flip flop strukturen i figur 5.1.1.1 er bedre enn 5.1.3.1, for de logiske portene vi har konstruert. Å ha flip flop'er som fungerer med lav forsyningsspenning er positivt med tanke på fremtiden for ultra lav spennings design.

En viktig parameter for flip floper er dens setup og holdetider. Jo lavere de er desto kjappere er dem. Testene vi utførte på flip flop A og Negativ klokkeflanke D flip flop med bare NOR2 porter (flip flop B) viser oss at holdtidene for flip flop'ene er bedre enn setuptidene. Dette skyldes at kretsen trenger tid til å lese inngangssignalet og regne ut utgangsverdien, men når den først har regnet ut utgangsverdien er ikke inngangen så viktig lenger. Flip flop A hadde som vi så en holdtid på 0 ns, og dette betyr at klokkesignalet kan gå skifte verdi samtidig som inngangssignalet og vi vil allikevel få riktig verdi på utgangen.

De fire flip flopene vi lagde ble benyttet for å realisere to ulike topologier av frekvensdelere for å vise en spesiell anvendelse av de. Begge topologiene fungerte med alle flip flop'ene unntatt flip flop C for 27 ^{o}C og 85 ^{o}C. At frekvensdelerene ikke fungerte med flip flop C var ikke en overaskelse siden flip flop C ikke hadde 100 % "yield". For de andre flip flop'ene som fungerte fikk vi ikke 100 % "yield" fra start, men etter hvert, og ingen av kretsene fungerte ved -40 ^{o}C. Dette skyldes at statistiske variasjoner og prosess parameter variasjoner som gjør at vi havner i ukjente tilstander ved oppstart. Dette kan kanskje løses ved å innføre asynkron reset, så vi kan sette flip flop'ene i ønsket start tilstand, og når det gjelder å øke "yield"en ved lavere temperaturer kan dette kanskje løses ved å øke V_{DD} og/eller bruke høyhastighetstransistorer med lavere V_t. Igjen så må

ønsket operasjonsområde tas i betraktning. Hvis vi ikke trenger en krets som trenger å fungere ved oppstart og ved -40 ^{o}C er det ikke nødvendig å forbedre kretsen. Av de to topologiene som er testet, ON Semiconductor og Xilinx, så er det den sist nevnte som fungerer best ved oppstart for to av de tre flip flop'ene som har 100 % "yield" etter hvert.

Konklusjonen er at det er fullt mulig å designe kretser med ultra lav spenning som fungerer, men de er sterkt applikasjons avhengige. Vi har vist at det går an å lage logiske porter og minneelementer som fungerer med antakelsene vi har tatt. Dette betyr at vi i prinsippet kan lage enhver FSM og datamaskin. Riktignok blir det ikke høyhastighets applikasjoner, men det bli applikasjoner som har mindre effektforbruk som også vises av ligning 1.2.1. Vi kan også konkludere med at dobling av gatelengden fra 0,1 µm til 0,2 µm vil føre til at disse kretsene vil fungere bedre, og at økning av gatelengden er også bedre enn redundans med den antakelsen at arealet til dobling av kretsen er så å si lik arealet til redundans 2. Ved å øke redundansen kan vi få kretsene til å fungere bedre, men da må vi igjen ofre areal og effektforbruk. ALU'en var også et bevis på at L = 0,2 µm fungerte bedre enn L = 0,1 µm, men som vi har sett på skyldes det at arealet til transistorene øker og vi får mindre variasjoner. Vi har også funnet ut at økning av V_{DD} vil føre til mer stabile kretser.

Videre arbeid kan være som følger:

- Se på hvorfor bredden til nMos transistoren må være større enn pMos transistoren for en NAND2 port, når den er designet for å ha symmetri om svitsjepunktet når V_{DD} = 1 V.
- Se om dobling av gatelengde og redundans i flip flop'er vil føre til flip flop'er med mindre setup og holdetid, og om operasjonsfrekvensen kan økes.
- Se om en økning av V_{DD} i flip flop'er vil føre til flip flop'er med mindre setup og holdetid, og om operasjonsfrekvensen kan økes.
- Se om frekvensdelerene vil ha høyere "yield" ved oppstart hvis vi bruker flip flop'er med asynkron reset.
- Se om frekvensdelerene vil fungere ved lavere temperaturer hvis vi bruker flip flop'er som er bygget ved hjelp av høyhastighets transistorer med lav V_t og/ eller høyere V_{DD}
- Kanskje det mest interessante, utforske hvorfor logiske porter fungerer bedre ved *lave* temperaturer når større kretser fungerer bedre for *høyere* temperaturer. I følge formel 1.3.1.1 burde transistorene fungere bedre ved *høye* temperaturer.

Referanser

[1] K. Granhaug, "Reducing Power Dissipation while increasing Yield and Defect-Tolerance in Subthreshold CMOS", pp. 7, 2006.

[2] C. Piguet, "Low Power CMOS-circuits Technology, Logic Design and CAD-tools", ISBN 0849395372,pp 1.1- 1.14, 2006.

[3] M. J. Flynn, "Computer engineering 30 years after the IBM Model 91", IEEE Computer, Vol. 31, No. 4, April 1998, pp. 27-31.

[4] R. Reid, "The CHIP: How Two Americans Invented the Microchip and Launched a Revolution", 2nd edition, Random House Trade Paperbacks, New York, 2001.

[5] M. Vertregt, Embedded analog technology, "IEDM Short Course on a System-on-a-Chip Technology", Dec. 5, 1999.

[6] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.82.

[7] http://www.8052.com/users/Suresh/XOR%20using%20NAND.GIF

[8] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.97.

[9] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.169.

[10] C. Piguet, "Low Power CMOS-circuits Technology, Logic Design and CAD-tools", ISBN 0849395372,pp 2.2, 2006.

[11] G. J. Myers et al., "Microprocessors technology trends", Proc. IEEE, Vol. 74, No.12, December 1986, pp. 1605-1622.

[12] *IEEE MICRO Issue,* December 1996

[13] A. Kaveh, "The History of power dissipation", Electronics Cooling, Januar 2000, Vol. 6, No.1

[14] C. Freeman, "The Economics of Industrial Innovation", Penguin Books, 1974.

[15] C. Piguet, "The First Qyartz Electronic Watch", Invited talk at PATMOS, Sevilla, Spania, September 11- 13, 2002, pp. 1-15.

[16] M. S. Malone, "The Microprocessor. A Biography", Springer – Verlag, New York, Desember, 1995.

[17] S. Aunet, Forelesningsnotater Uke 1 TDT4255 Maskinvarekonstruksjon, NTNU, H2006

[18] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.134.

[19] http://www.geocities.com/SiliconValley/2072/2to4.gif

[20] C. Piguet, "Low Power CMOS-circuits Technology, Logic Design and CAD-tools", ISBN 0849395372,pp 7.9, 2006.

[21] C. Piguet et al, "Low- power low- voltage digital CMOS cell design", Proc. PATMOS ’94, Barcelona, Spania, Oktober 17-19, 1994, pp.132-139.

[22] http://no.wikipedia.org/wiki/Aritmetisk_logisk_enhet

[23] http://www.asic-world.com/digital/seq4.html

[24] http://www.play-hookey.com/digital/d_nand_flip-flop.html

[25] http://www.play-hookey.com/digital/d_nor_flip-flop.html

[26] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.175 – 176.

[27] http://wpcontent.answers.com/wikipedia/en/thumb/7/74/Edge_triggered_D_flip-flop.png/180px-Edge_triggered_D_flip-flop.png

[28] C. Piguet, J. Zahnd "Signal-Transition Graph- based Design of Speed-Independent CMOS Circuits", ESSCIRC’98, September 21-24 1998, Den Haag, Nederland, pp 432 – 435.

[29] C. Piguet, "Low Power CMOS-circuits Technology, Logic Design and CAD-tools", ISBN 0849395372,pp 7.6, 2006.

[30] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.349.

[31] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.379.

[32] P. Alfke, "Unusual Clock Dividers", Xilinx Applications

[33] C. Petty, P. Shockman, "Odd Number Divide By Counters With 50 % Outputs and Synchronous Clocks", Product Applications AND8001/D, ON Semiconductor.

[34] G. Moore, "Cramming more components onto interated circuits", Electronics, Volume 38, Number 8, April 19, 1965.

[35] http://no.wikipedia.org/wiki/Fil:Moore_Law_diagram_(2004).jpg

[36] G. Koch, "Discovering Multi-Core: Extending the Benefits of Moore’s Law" Technology@Intel Magazine, July 2005.

[37] P. P. Gelsinger, "Microprocessors for the New Millenium: Challenges, Opportunities, and New Frontiers", Digest of technical papers, 2001 IEEE Solid-State Circuits Conference, pp. 22-25, 2001.

[38] J. Rabaey, M. Pedram and P. Landman, "Low power Design Methodologies", Kluwer Academic Publishers, 1995.

[39] A. Wang, A. P. Chandrakasan and S. V. Kosonocky, "Optimal Supply and Threshold Scaling for Subthreshold CMOS Curciuts" IEEE Computer Society Annual Symposium on VLSI, ISVLSI, 2002.

[40] H. Soeleman, K. Roy and B. C. Paul, "Robust Subthershold Logic for Ultra-Low Power Operation", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 9, no.1, pp. 90-99, February 2001.

[41] A. P Chandrakasan, S Sheng, and R. W. Brodersen, "Low-power CMOS digital design", IEEE J. of Solid-State Cicuits, Vol. 27, No. 4, April 1992, pp. 473-484.

[42] W. Nebel and J. Mermet, eds., "Low-power Design in Submicron Electronics", NATO ASI Series, E 337, 1997, Kluwer Academic Publishers, Dordrecht, Chapter 4.2 and 9.1.

[43] K. Granhaug and S. Aunet, "Six Subthreshold Full Adder Cells characterized in 90 nm CMOS technology", in 9th IEEE workshop on Design & Diagnostics of Electronic Circuit & Sytstems, Architecture Processors (ASAP'05), 2005.

[44] C. Piguet, "Low Power CMOS-circuits Technology, Logic Design and CAD-tools", pp.1.1, ISBN 0849395372,pp 17.1- 17.2, 2006.

[45] C. Piguet, "Low Power CMOS-circuits Technology, Logic Design and CAD-tools", ISBN 0849395372,pp 7.2, 2006.

[46] M. Mano, "Digital Design 3rd edition", ISBN 0130355259, 2002, pp.169 – 180.

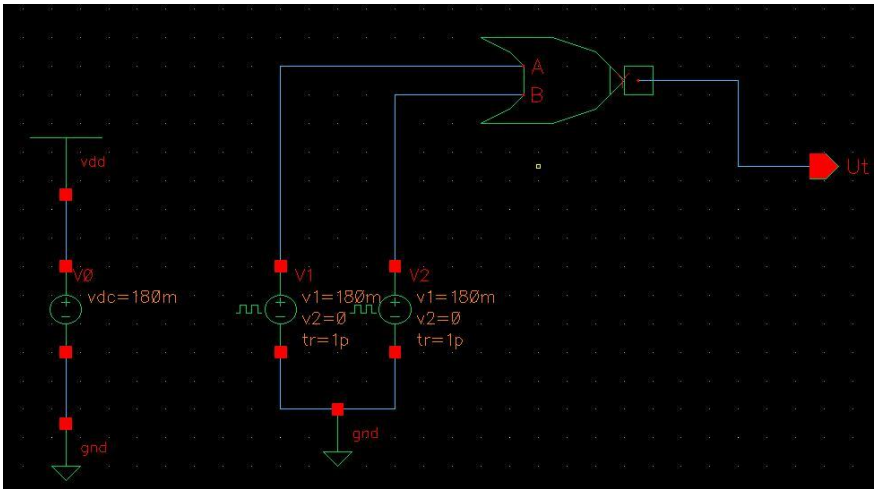
[47] N.H.E. Weste and K. Eshraghian, "Principles of CMOS VLSI Design", Addison Wesley, 2nd edition, 1993

APPENDIX A

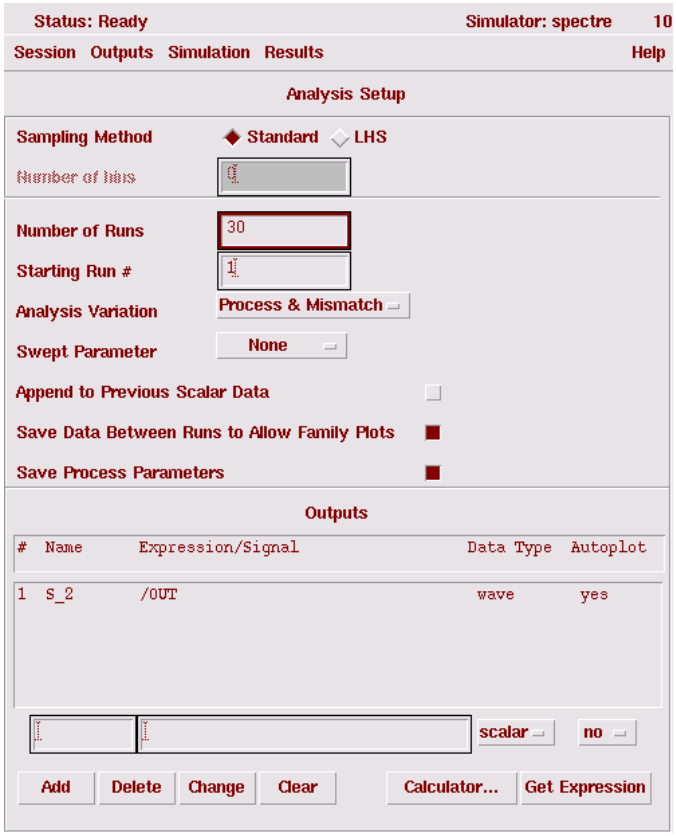
Testbenker for de forskjellige kretsene

A.1 Testbenk for logiske porter

Figur A.1 viser testbenken for en logisk port. Helt til venstre i bildet har vi spenningsgeneratoren for V_{DD}. Denne kan vi justere til de V_{DD}'ene vi ønsker å simulere på. Vi har en pulsgenerator på hver av inngangene til den logiske porten som vi sender inn firkantpulser med. Disse firkantpulsene har vi prøvd å gjøre så perfekte som mulig, og derfor har pulsene en stige- og falltid på 1 ps. AND2, OR2 og XOR portene er testet slik at en inngang varer i 1 µs, men NAND2, NOR2 og inverteren har hatt en inngang som varer i 1 ms. Som man også kan se av simuleringene i oppgaven er det utført transient analyser på de logiske portene som varte i 4 ms for NAND2, NOR2 innverteren og 4µs for AND2, OR2 og XOR portene. Ved MonteCarlo simuleringer har det vært oppsett som på figur A.1.2. Eneste som har vært annerledes er antall ganger den har blitt kjørt. Inngangene har alltid '0' nivå = 0 V og '1' nivå = V_{DD}.



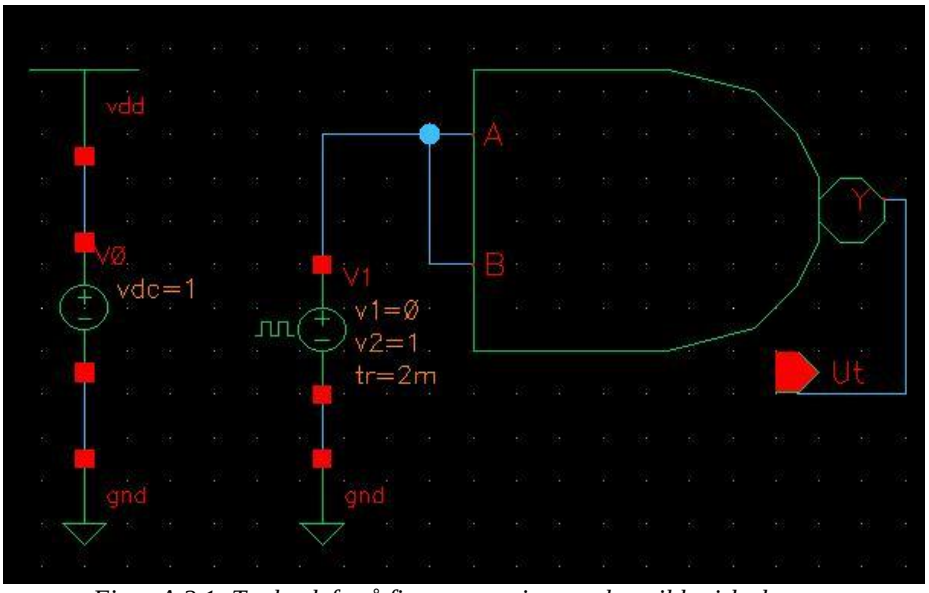
Figur A.1.1: Testbenk for logiske porter



Figur A.1.2: Oppsett for MonteCarlo simuleringer

A.2 Testbenk for operasjonspunkter til logiske porter

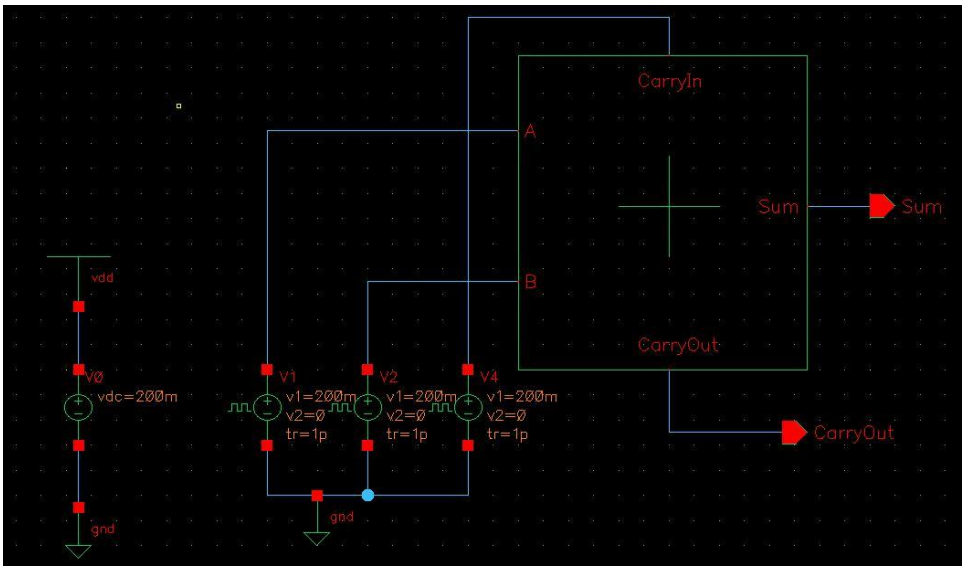
Figur A.2.1 viser tesbenken for å finne ut operasjonspunktet til logiske porter. Vi kobler inngangene sammen og har en inngangspenning på $V_{DD}/2$, som går fra ’0’ til ’1’ på 2 ms slik at vi får en ”rampe-graf”. Utgangen på kretsen skal være lik inngangen ved 1 ms som er det punktet hvor inngangen er $V_{DD}/2$. Hvis utgangen ikke er $V_{DD}/2$ ved 1 ms, må vi justere på W på pMos transistoren får å få operasjonspunktet til å justere seg til riktig punkt. Inngangen har et ’0’ nivå = 0 V og et ’1’ nivå = V_{DD}.



Figur A.2.1: Testbenk for å finne operasjonspunktet til logiske kretser

A.3 Testbenk for en 2 – bit Full – Adder

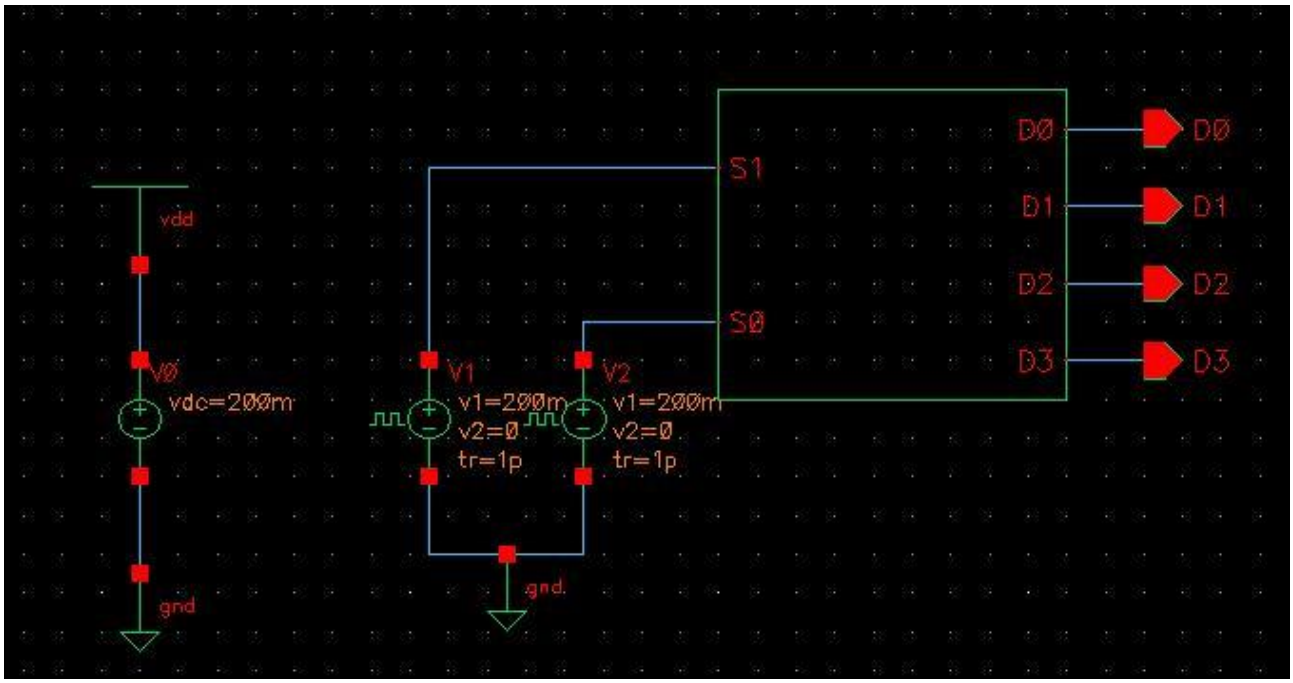
På figur A.3.1 ser vi testbenken for en 2 – bit Full – Adder. Her er V_{DD} = 200 mV og inngang A er en firkantpuls med f = 125 kHz, inngang B er en firkantpuls med f = 250 kHz og CarryIn er også en firkantpuls med f = 500 kHz. Inngangene har et ’0’ nivå = 0 V og et ’1’ nivå = V_{DD}.



Figur A.3.1: Testbenken for 2 – bit Full – Adderen

A.4 Testbenk for en 2 – 4 Dekoder

På figur A.4.1 ser vi testbenken for en 2 – 4 Dekoder. Her er V_{DD} = 200 mV og inngang S0 er en firkantpuls med f = 250 kHz og inngang S1 er også en firkantpuls med f = 125 kHz. Inngangene har et ’0’ nivå = 0 V og et ’1’ nivå = V_{DD}.



Figur A.4.1: Testbenken for 2 – 4 Dekoderen

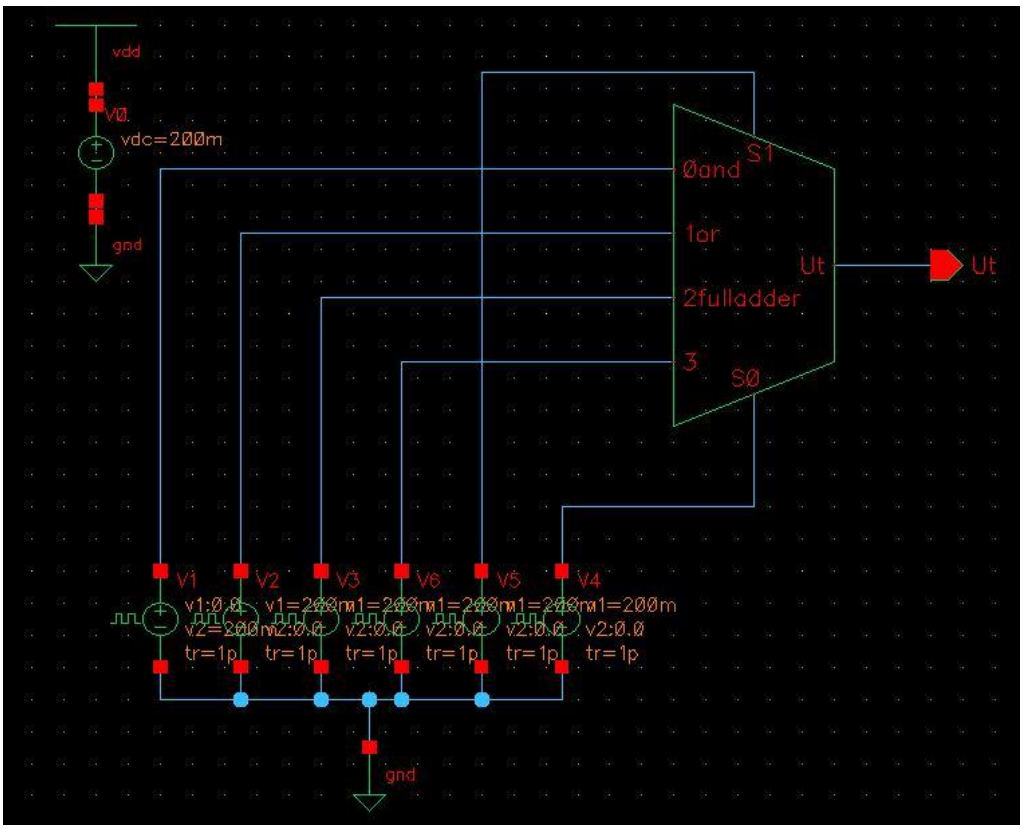
A.5 Testbenk for en 4 – 1 MUX

På figur A.5.1 ser vi testbenken for en 4 – 1 MUX. Her er V_{DD} = 200 mV og inngang S0 er en firkantpuls med f = 250 kHz og inngang S1 er også en firkantpuls med f = 125 kHz. S0 og S1 vil avgjøre hvilken av inngangene som blir koblet til utgangen og dette kan testet på en spesiell måte:

A = 0 fra 0 – 1 µs, ’1’ ellers,
B = 0 fra 1 – 2 µs, ’1’ ellers,
C = 0 fra 2 – 3 µs, ’1’ ellers og
D = 0 fra 3 – 4 µs, ’1’ ellers.

Hvis MUX’en er designet korrekt nå vil Ut = 0 for hele simuleringen hvis vi simulerer i 4 µs, fordi hver av inngangene vil dra utgangen *lav* i 1 µs. Vi kan også ”snu” på inngangene og ta en test til for å se om vi får Ut = 1. Hvis det går er MUX’en riktig.

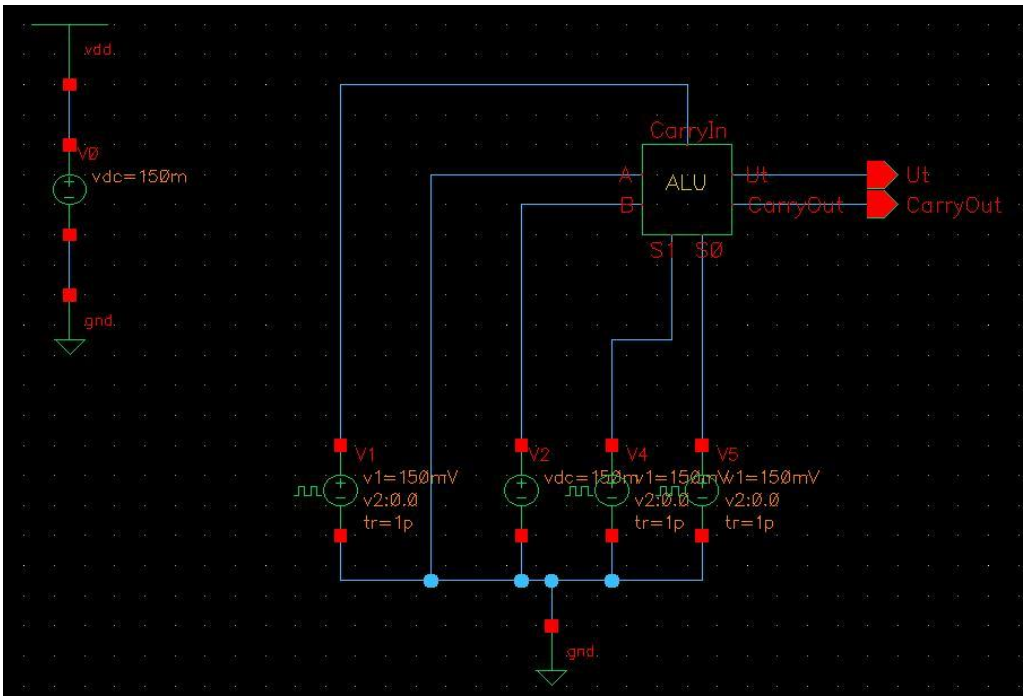
Inngangene har et ’0’ nivå = 0 V og et ’1’ nivå = V_{DD}.



Figur A.5.1: Testbenken til 4 – 1 MUX'en

A.6 Testbenk for ALU

På figur A.6.1 ser vi testbenken for en 4 – 1 ALU. Her er V_{DD} = 150 mV (kan endres) og inngang S0 er en firkantpuls med f = 250 kHz, inngang S1 er også en firkantpuls med f = 125 kHz og inngang CarryIn er også en firkantpuls med f = 250 kHz. Inngang A og inngang B er satt til faste verdier (A = 0, B = 1). Inngangene har et '0' nivå = 0 V og et '1' nivå = V_{DD}.



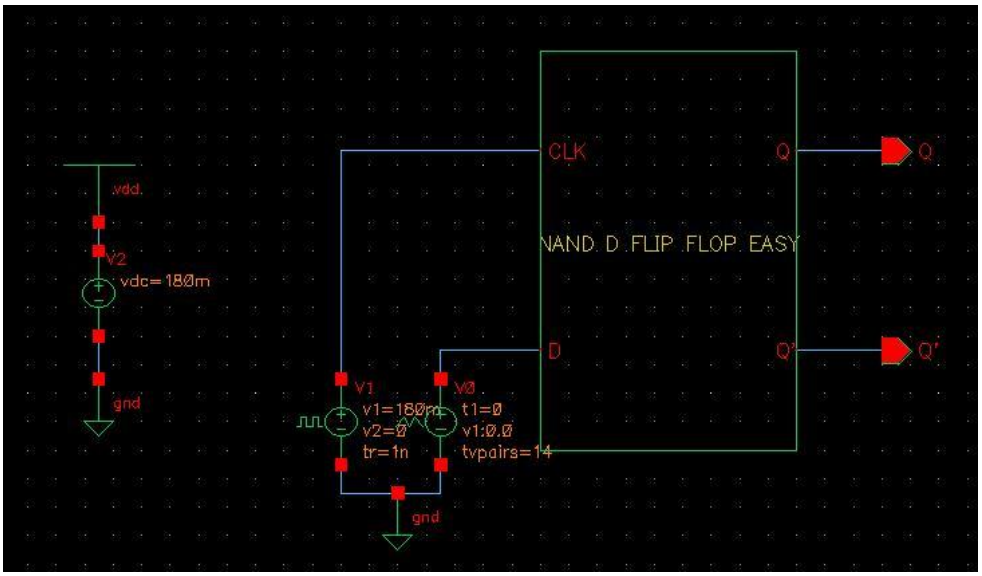
Figur A.6.1:Testbenken til ALU'en

A.7 Testbenk for de fire D – Flip flop'ene

På figur A.7.1 ser vi testbenken for de fire D – Flip Flop'ene. CLK er klokke ingangen med f = 250 kHz (f = 500kHz for Race Free D Flip Flop). D inngangen er en "array" med følgende verdier:

D = 0 fra 0 – 1 ms, 3 – 4 ms og 5 – 7 ms,
D = 1 fra 1 – 3 ms, 4 – 5 ms og 7 – 11 ms.

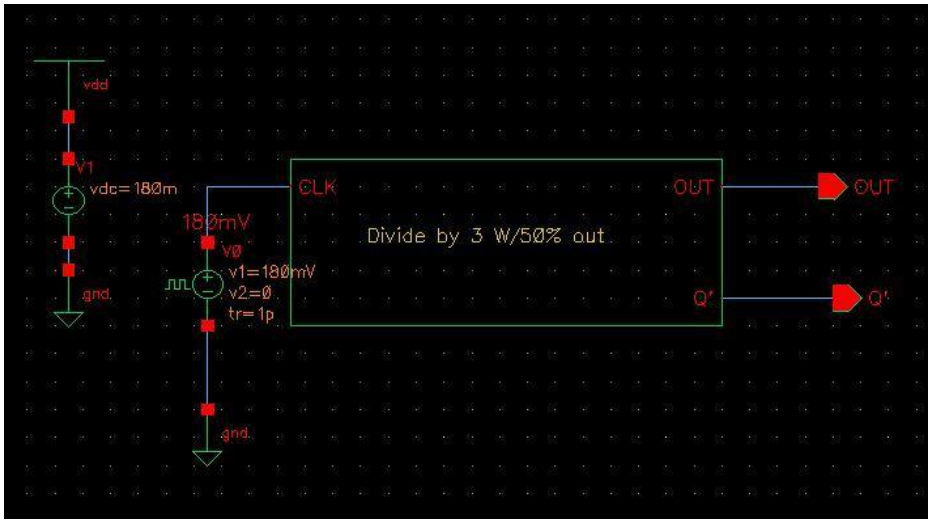
Inngangene har et '0' nivå = 0 V og et '1' nivå = V_{DD}.



Figur A.7.1: Testbenken for flip flop'ene

A.8 Testbenk for frekvensdelerene

På figur A.8.1 ser vi testbenken for frekvensdelerene. Den har kun en klokkeinngang som har en f = 250 kHz fast.



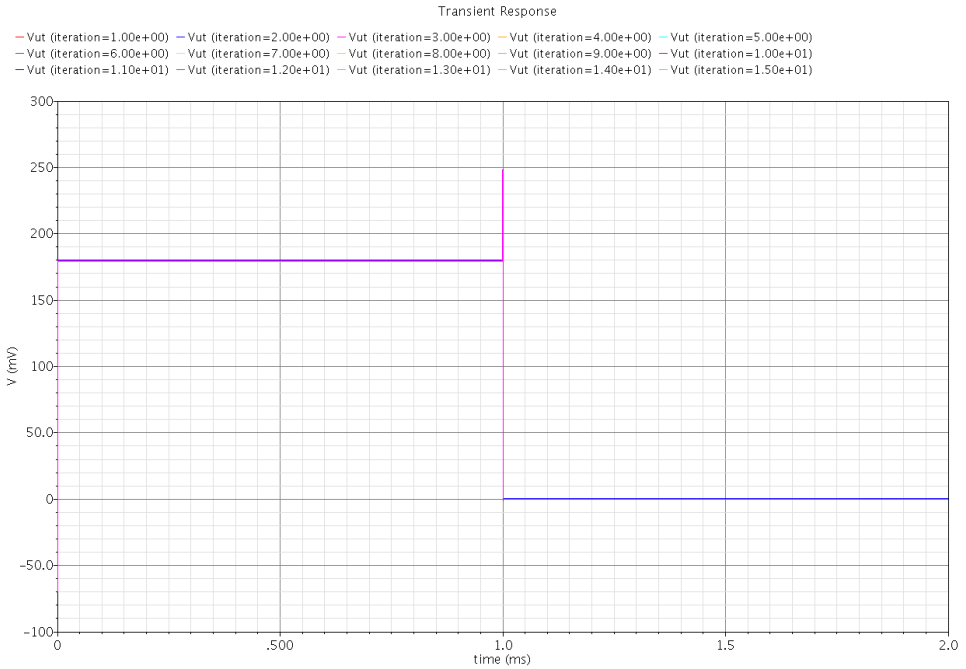
Figur A.8.1: Tesbenken for frekvensdelerene

APPENDIX B

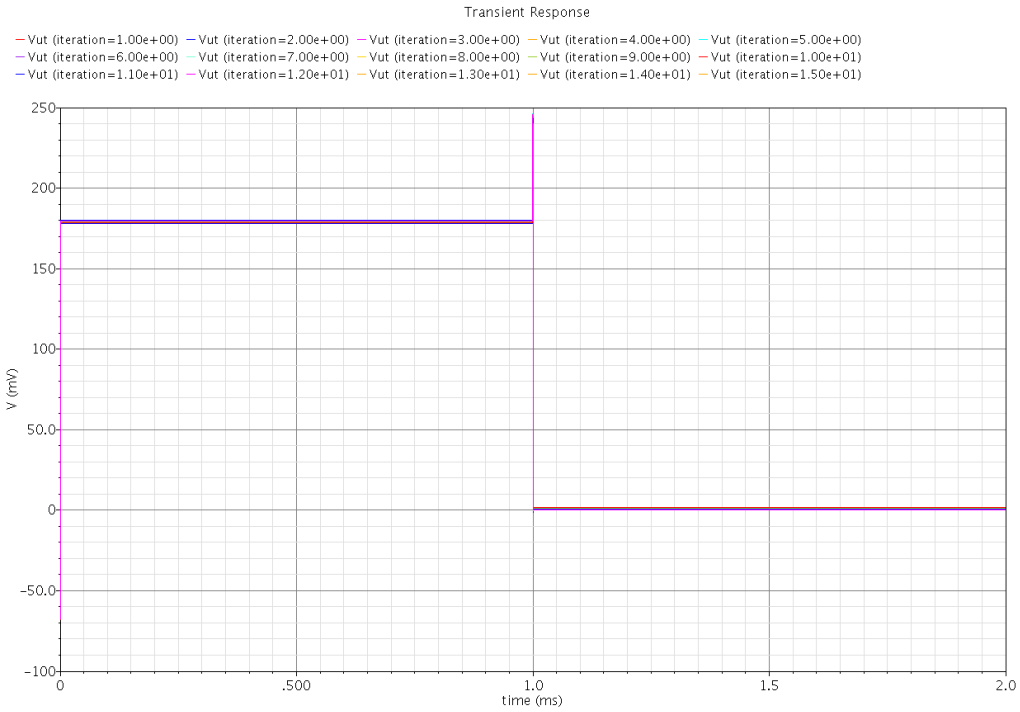
Et utvalg av MonteCarlo simuleringene for de forskjellige kretsene

B.1 MonteCarlo Simuleringer av INVERTER'en V_{DD} = 180 mV

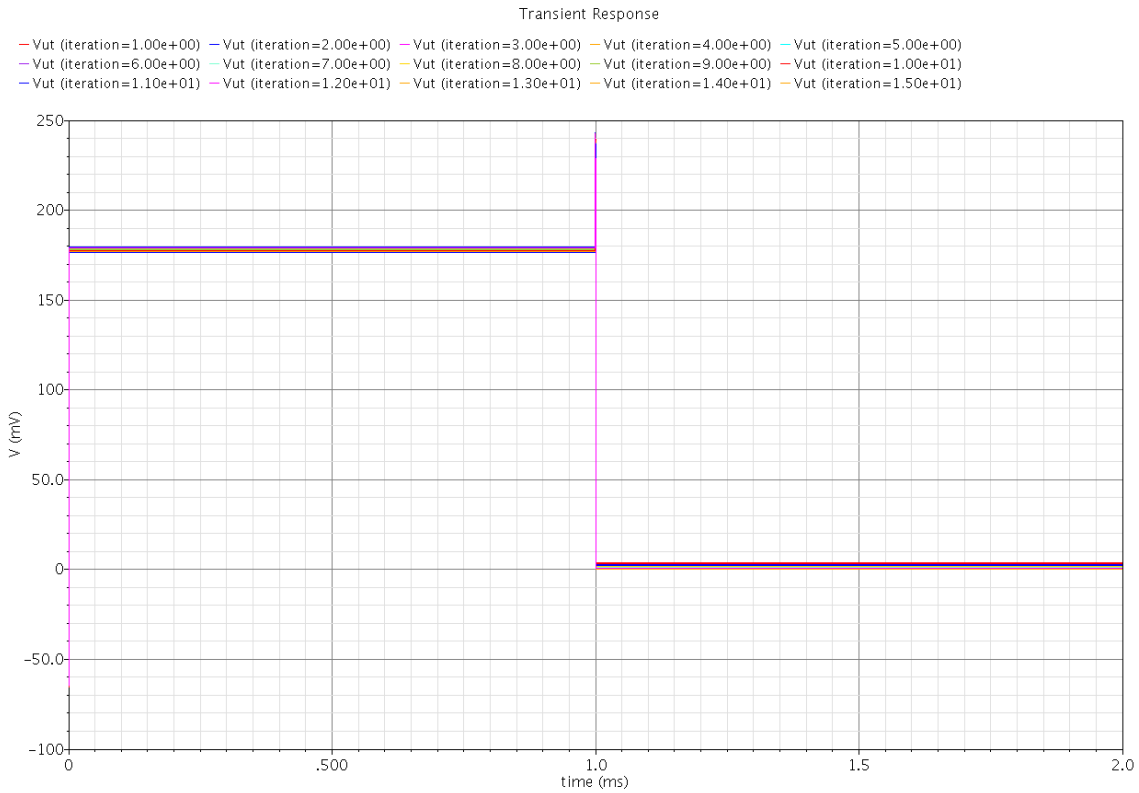
De tre neste figurene viser MonteCarlo simuleringene for INVERTER'en i figur 2.1.3.2 for V_{DD} = 180 mV.



Figur B.1.1: MonteCarlo simulering av INVERTER'en ved -40°C



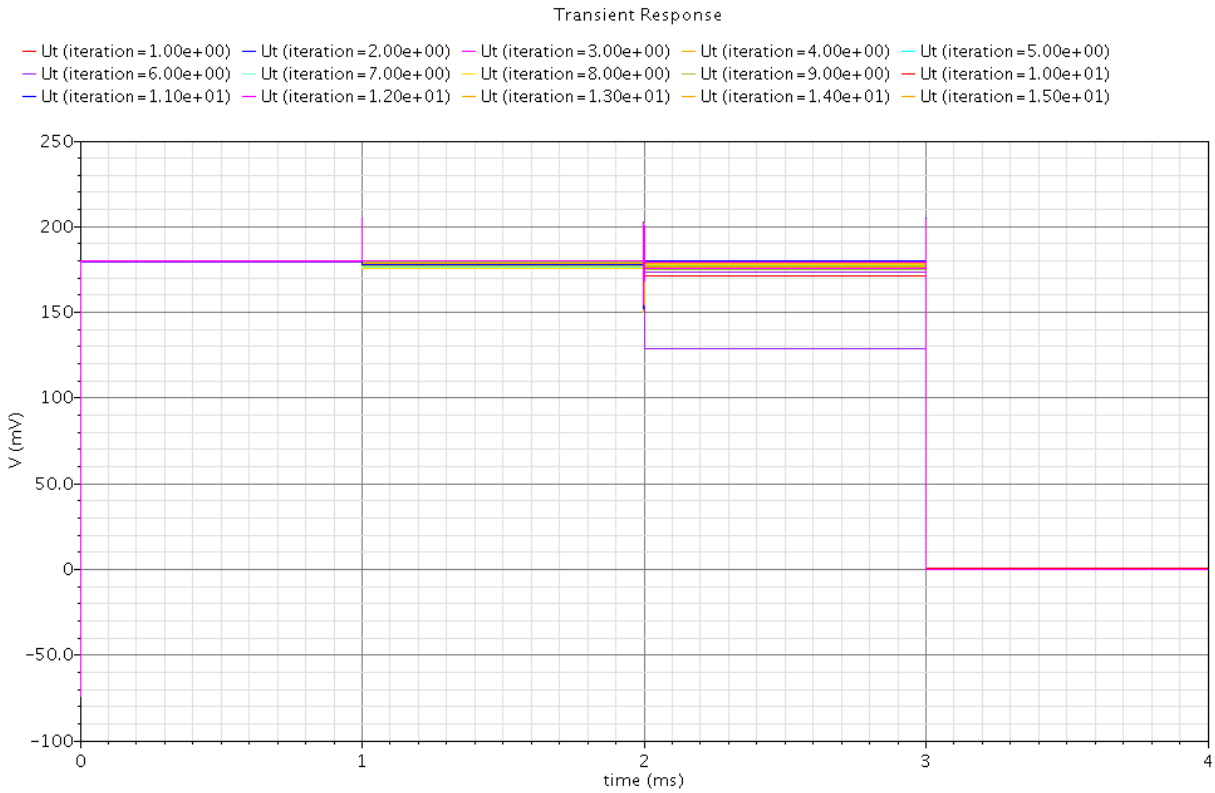
Figur B.1.2: MonteCarlo simulering av INVERTER'en ved 27°C



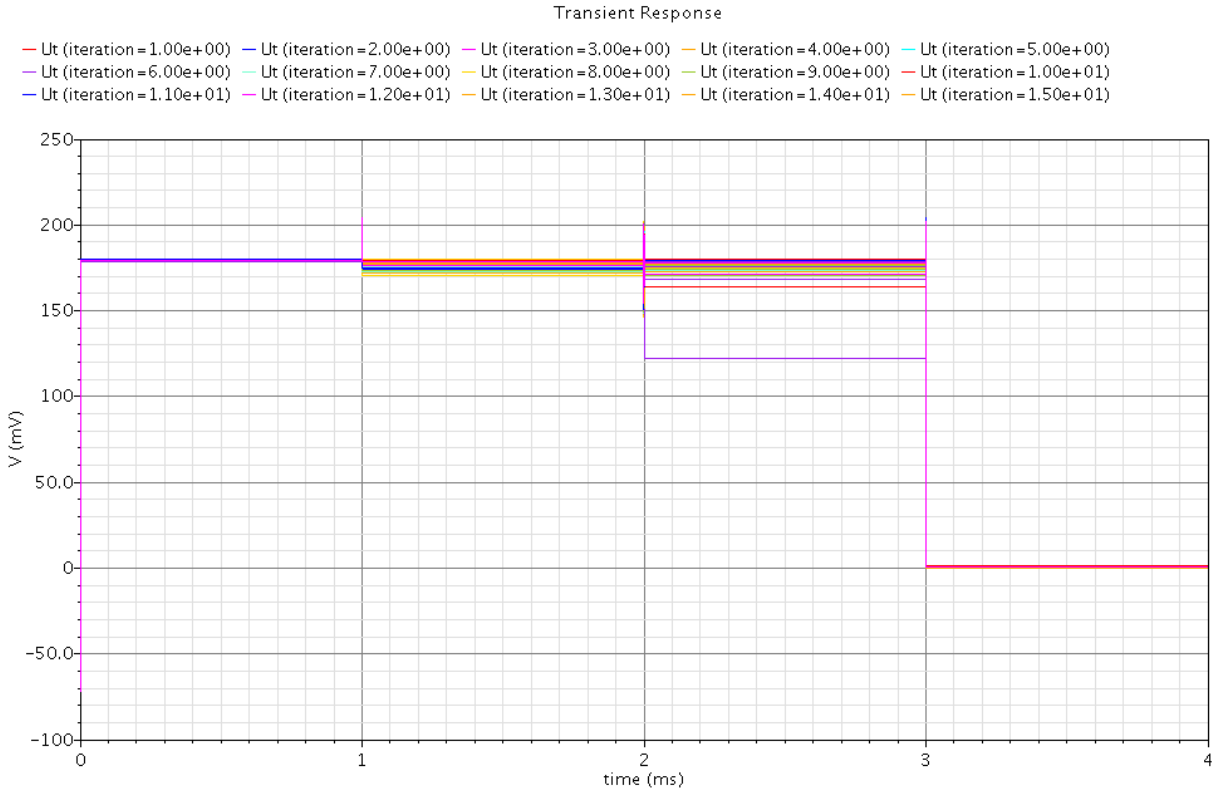
Figur B.1.3: MonteCarlo simulering av INVERTER'en ved 85 °C

B.2 MonteCarlo Simuleringer av NAND2 porten V_{DD} = 180 mV

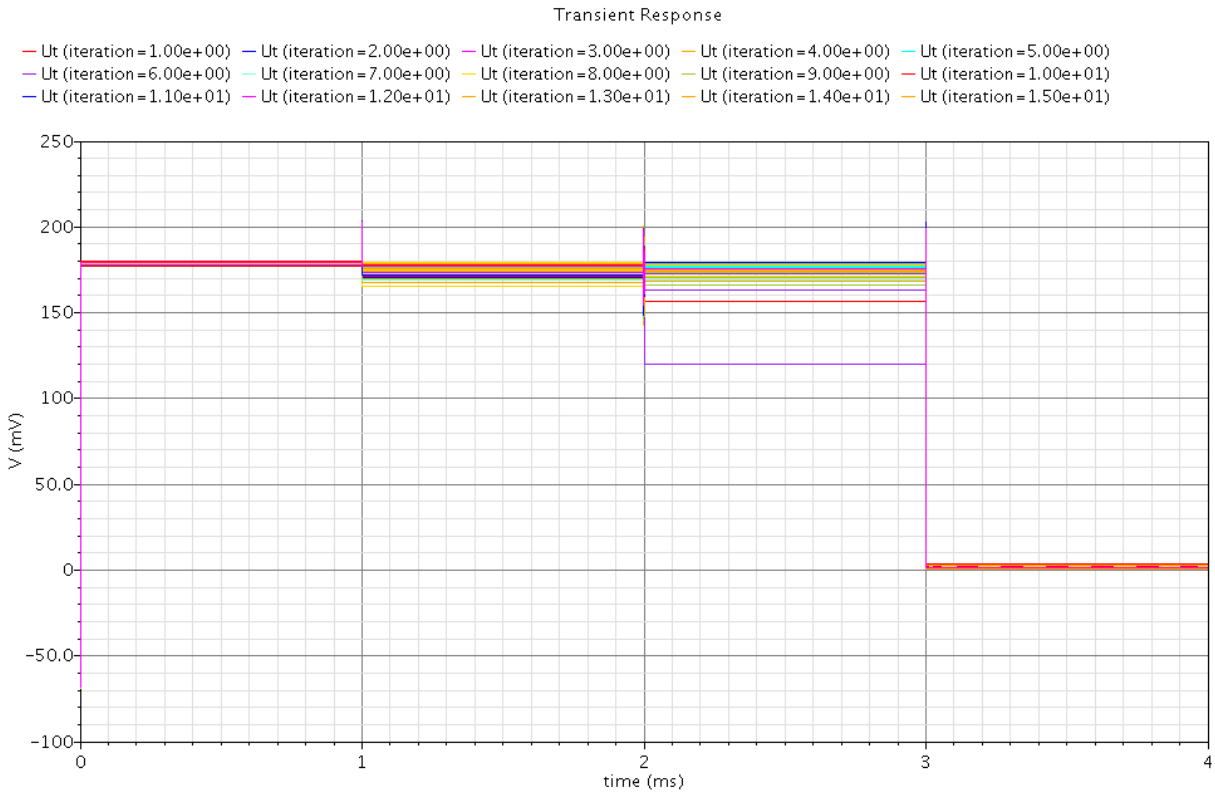
De tre neste figurene viser MonteCarlo simuleringene for NAND2 porten i figur 2.1.4.2 for V_{DD} = 180 mV.



Figur B.2.1: MonteCarlo simulering av NAND2 porten ved -40 °C



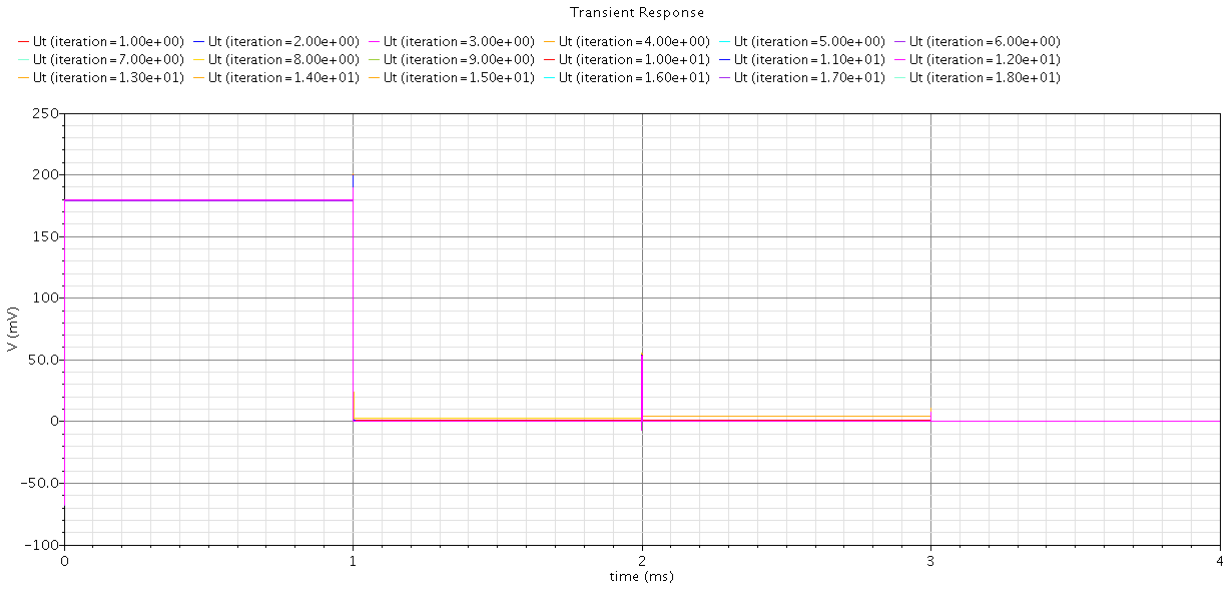
Figur B.2.2: MonteCarlo simulering av NAND2 porten ved 27 °C



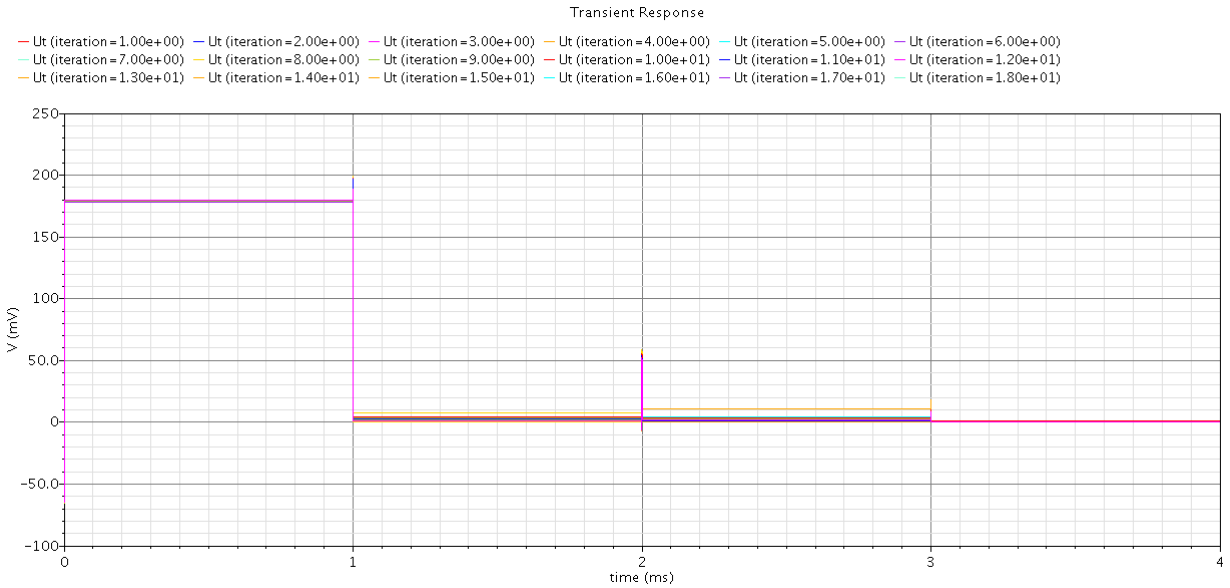
Figur B.2.3: MonteCarlo simulering av NAND2 porten ved 85 °C

B.3 MonteCarlo Simuleringer av NOR2 porten V_{DD} = 180 mV

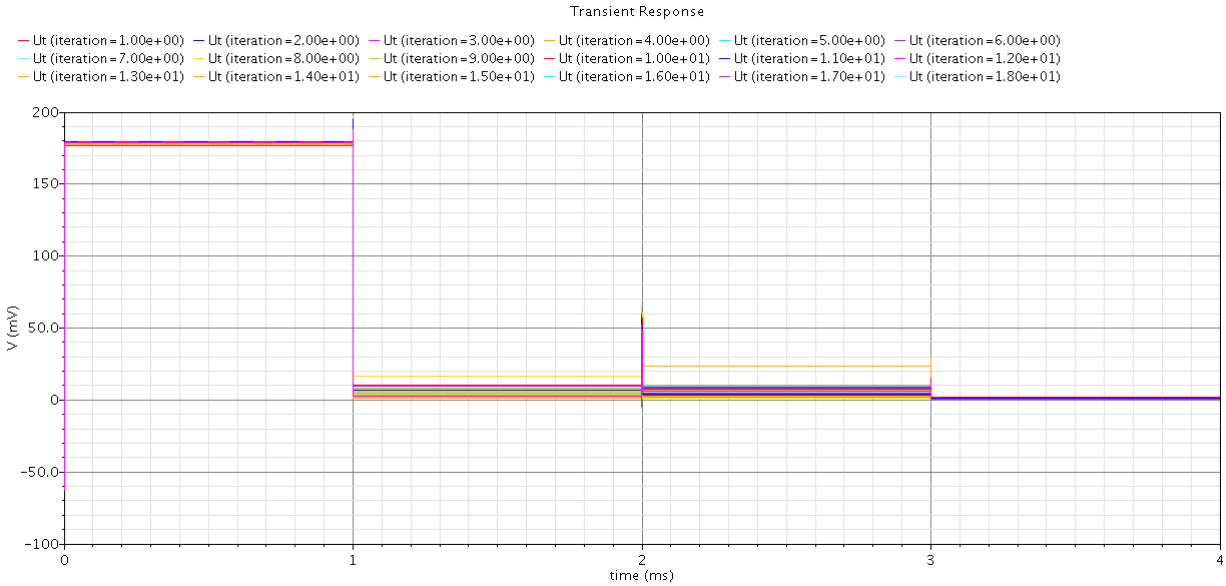
De tre neste figurene viser MonteCarlo simuleringene for NAND2 porten i figur 2.1.5.2 for V_{DD} = 180 mV.



Figur B.3.1: MonteCarlo simulering av NOR2 porten ved -40 ^{o}C



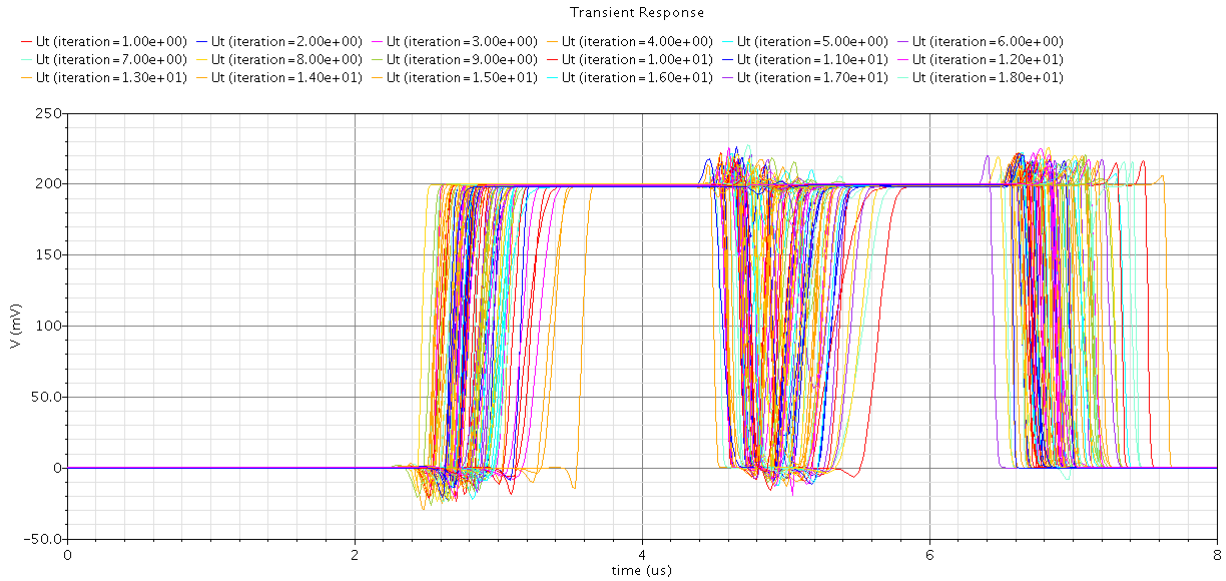
Figur B.3.2: MonteCarlo simulering av NOR2 porten ved 27 ^{o}C



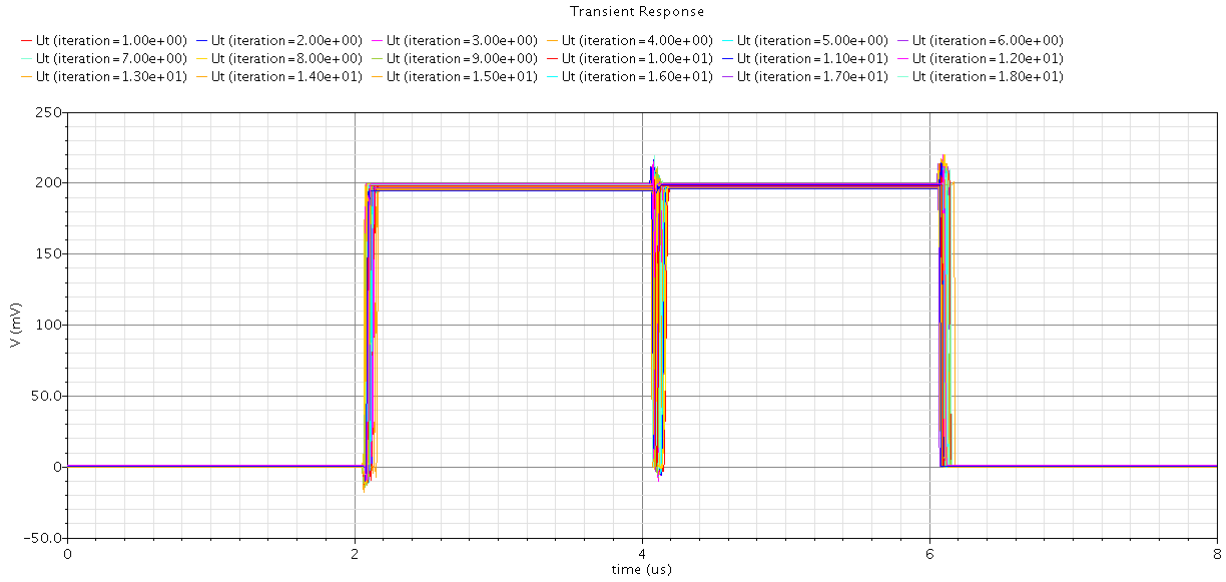
Figur B.3.3: MonteCarlo simulering av NOR2 porten ved 85 ^{o}C

B.4 MonteCarlo Simuleringer av ALU'en V_{DD} = 200 mV

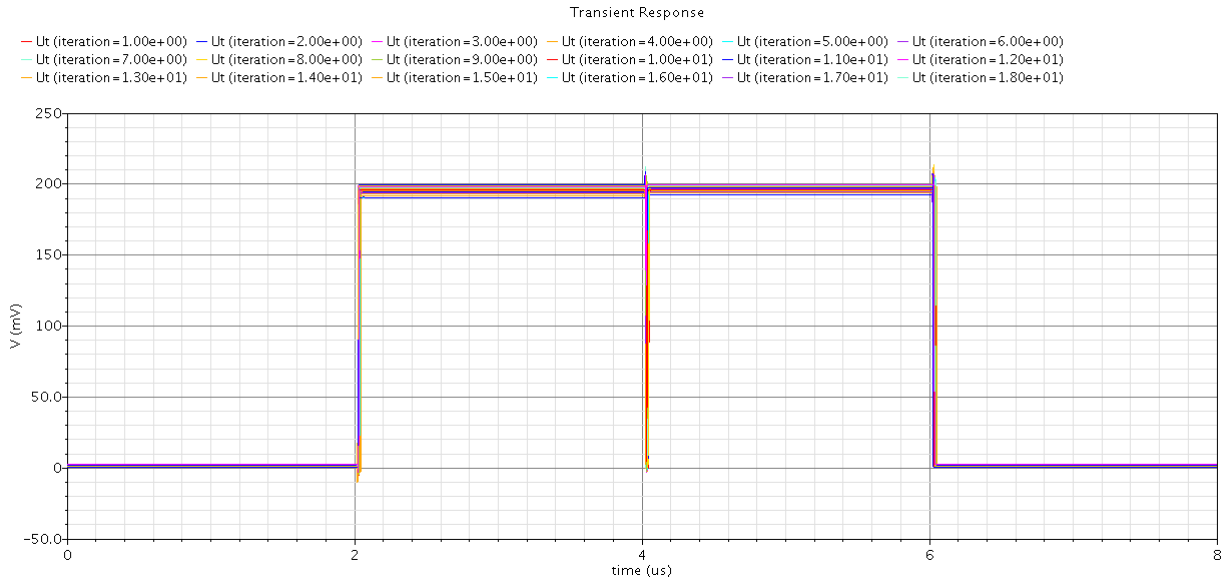
De tre neste figurene viser MonteCarlo simuleringene for ALU'en i figur 4.1.1.1 som er laget av transistorer med L = 0,1 µm for V_{DD} = 200 mV.



Figur B.4.1: MonteCarlo simulering av ALU'en ved -40 ^{o}C



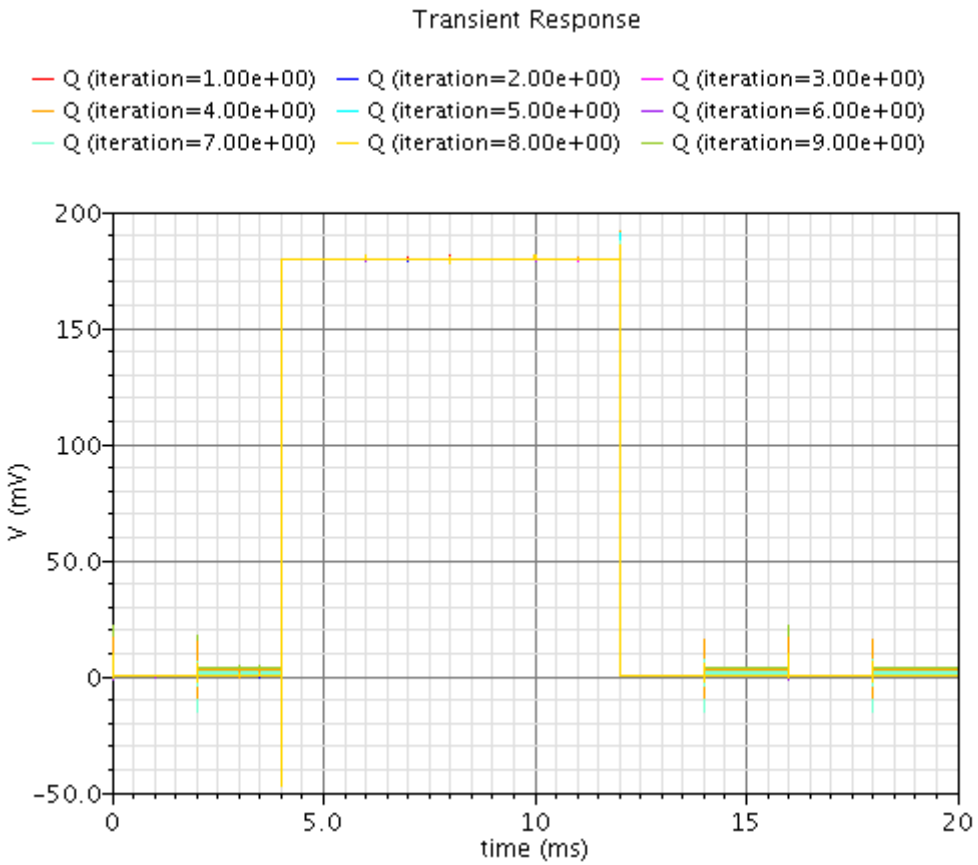
Figur B.4.2: MonteCarlo simulering av ALU'en ved 27 ^{o}C



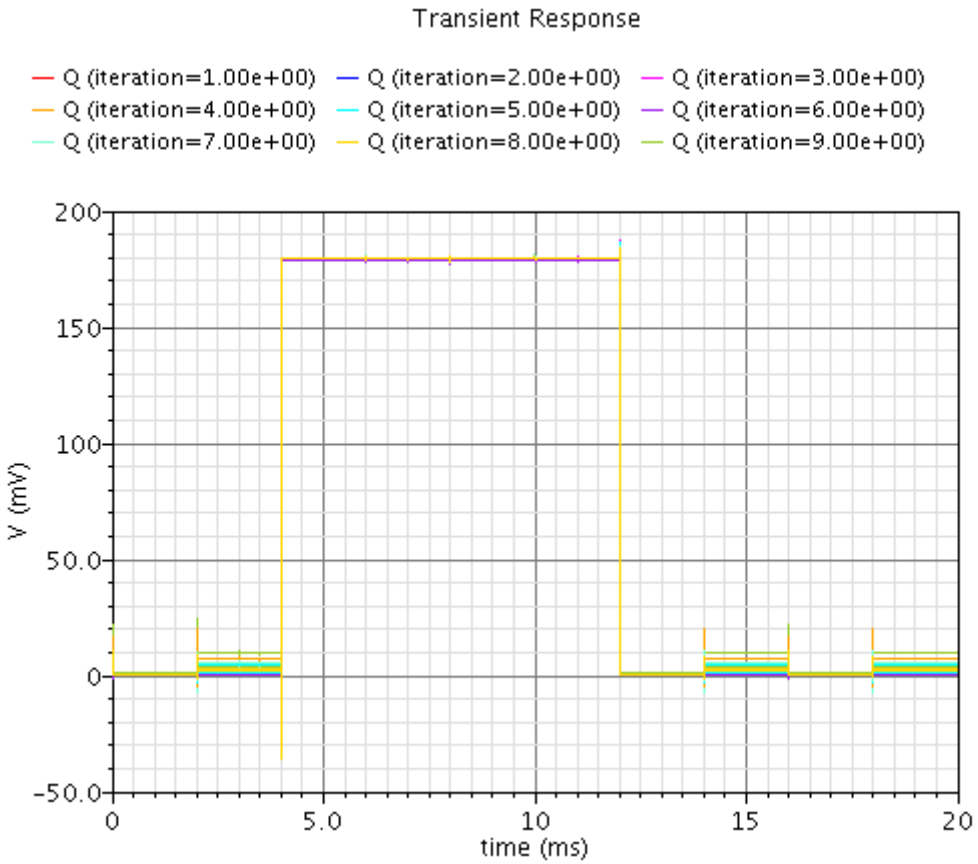
Figur B.4.3: MonteCarlo simulering av ALU'en ved 85 ^{o}C

B.5 MonteCarlo Simuleringer av negativ klokkeflanke D flip flop med V_{DD} = 180 mV

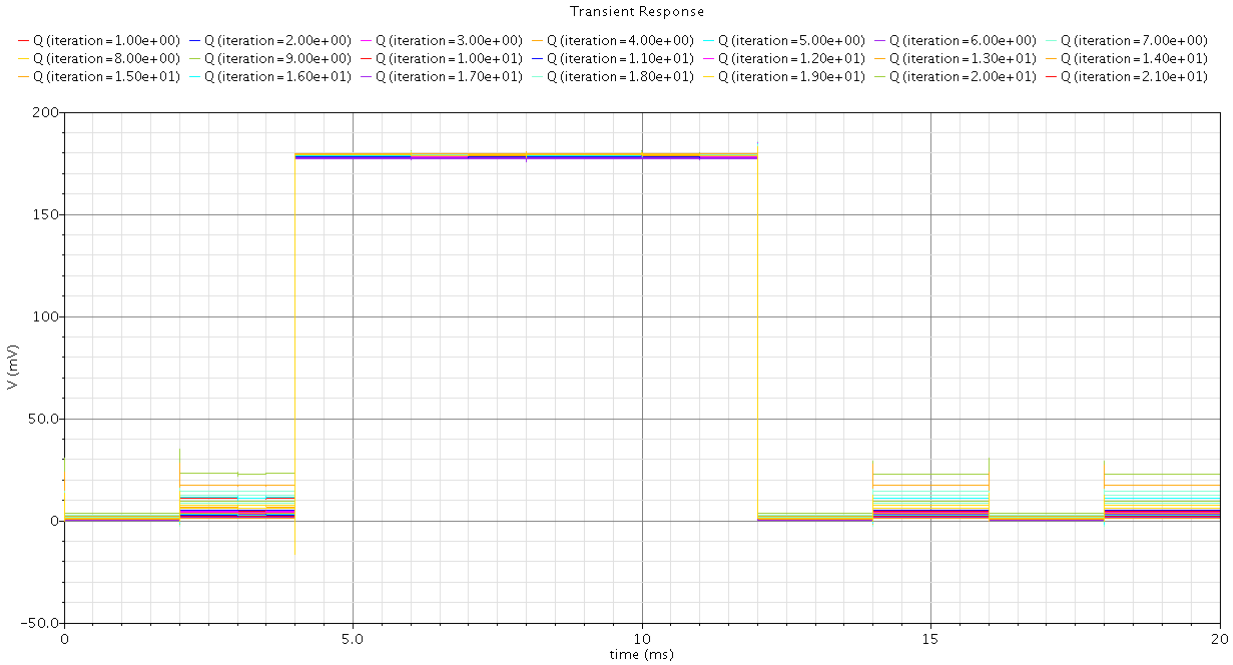
De tre neste figurene viser MonteCarlo simuleringene for flip flop'en i figur 5.1.2.1 for V_{DD} = 180 mV.



Figur B.5.1: MonteCarlo simulering av flip flop'en ved -40 °C



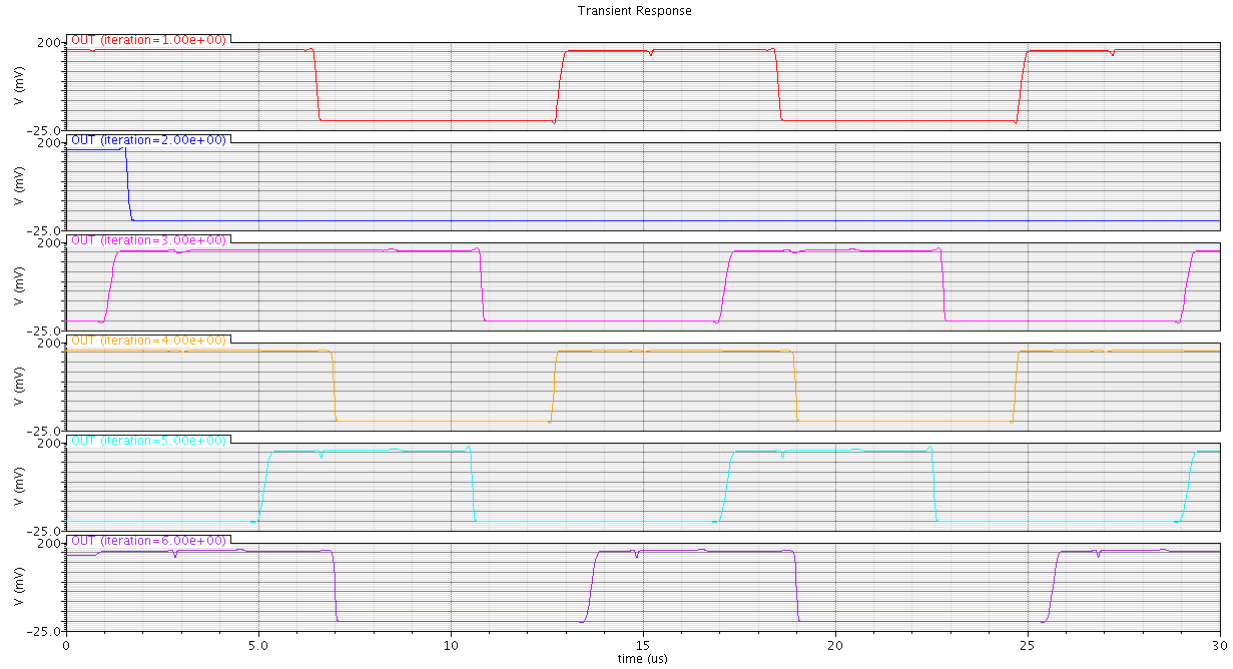
Figur B.5.2: MonteCarlo simulering av flip flop'en ved 27 °C



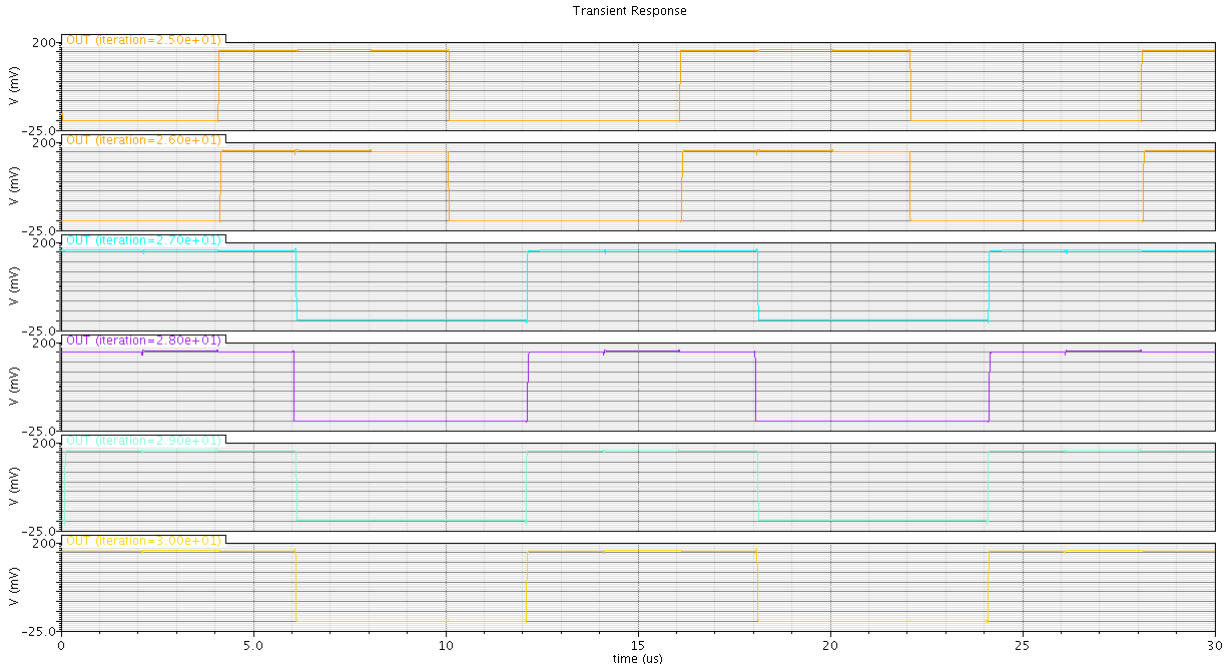
Figur B.5.3: MonteCarlo simulering av flip flop'en ved 85 ^{o}C

B.6 MonteCarlo Simuleringer av ON Semiconductor frekvensdeleren V_{DD} = 180 mV

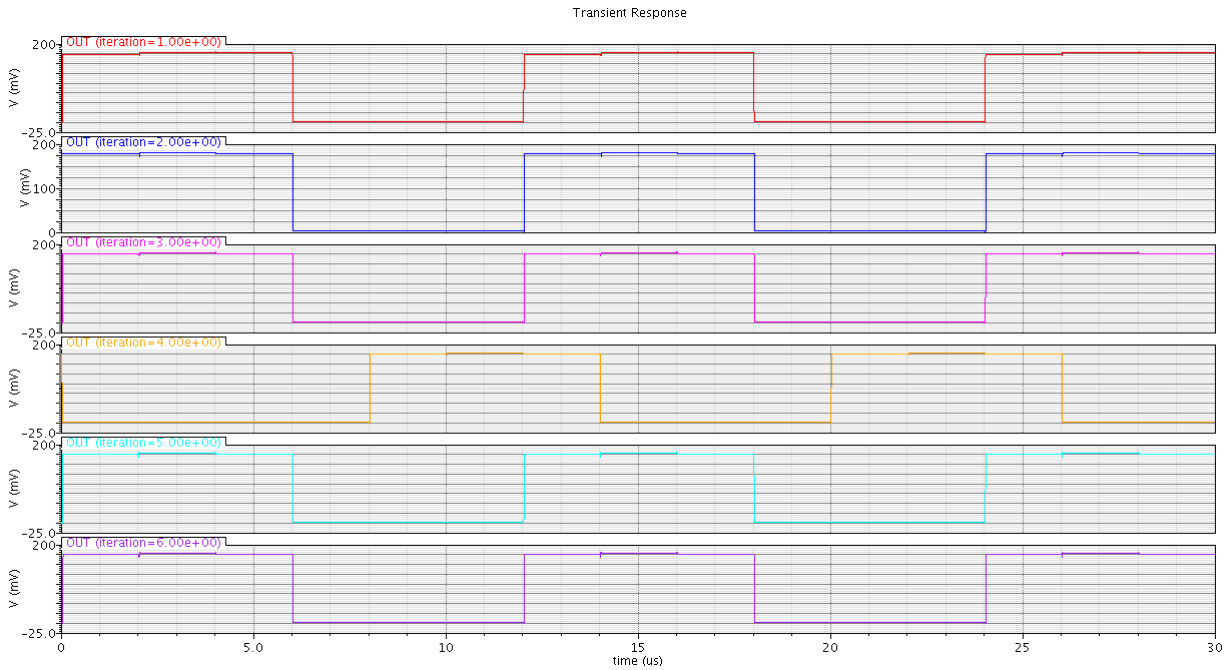
De tre neste figurene viser MonteCarlo simuleringene for frekvendeleren i figur 6.2.1.1 for V_{DD} = 180 mV. Kun MonteCarlo simuleringene for frekvensdeleren realisert med Negativ klokkeflanke D Flip Flop er tatt med for illustrasjonsskyld, og et utvalg av de 30 MonteCarlo simuleringene.



Figur B.6.1: MonteCarlo simulering av frekvensdeleren ved -40 ^{o}C



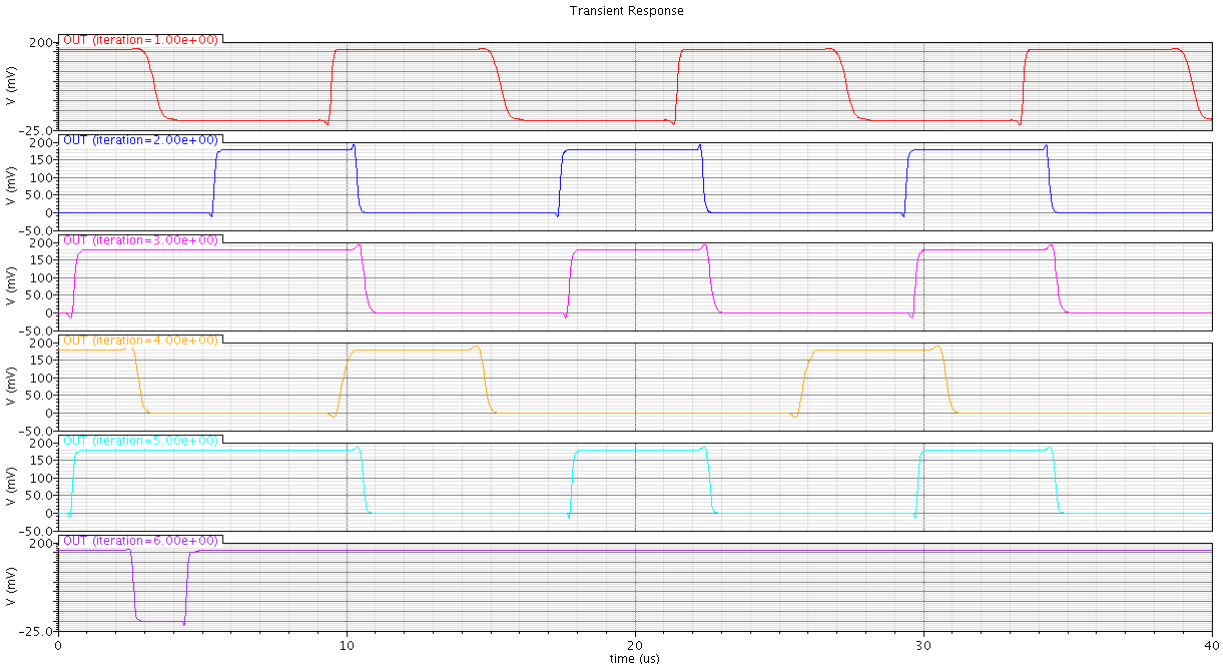
Figur B.6.2: MonteCarlo simulering av frekvensdeleren ved 27 ^{o}C



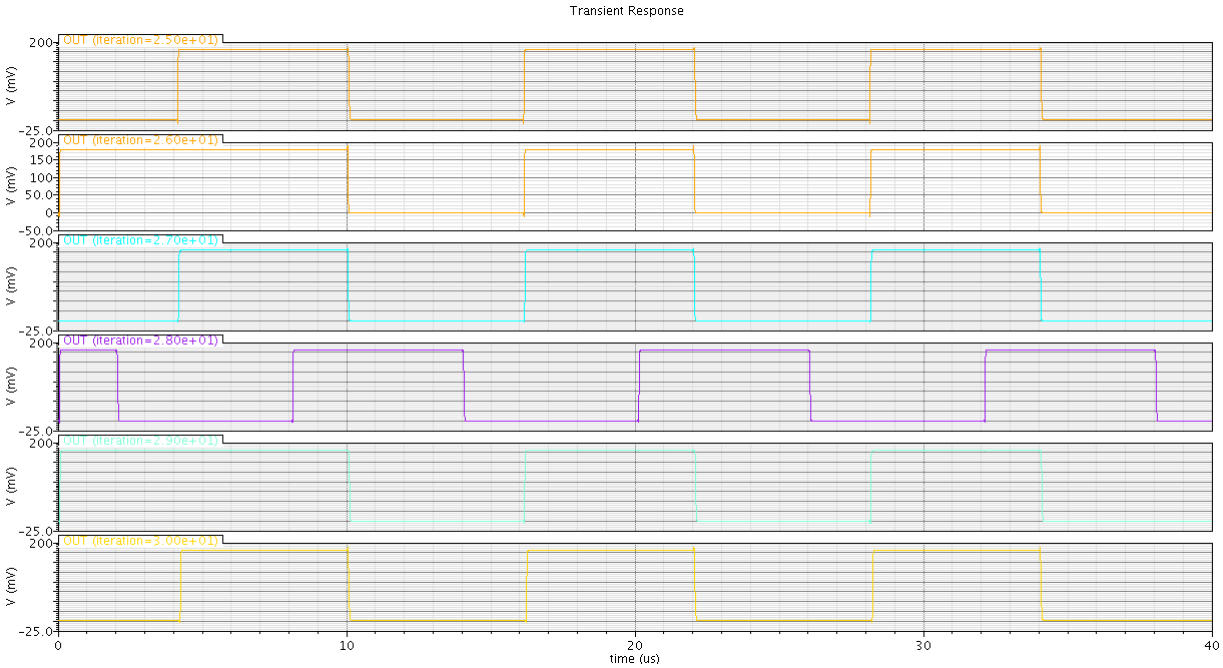
Figur B.6.3: MonteCarlo simulering av frekvensdeleren ved 85 ^{o}C

B.7 MonteCarlo Simuleringer av Xilinx frekvensdeleren V_{DD} = 180 mV

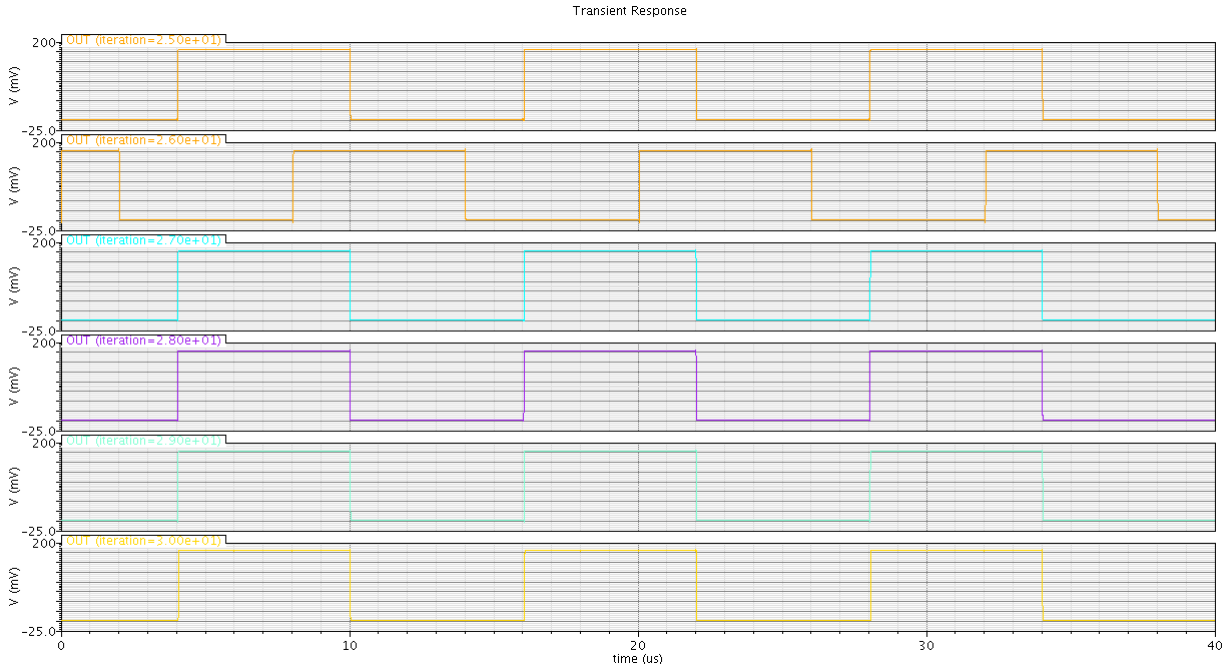
De tre neste figurene viser MonteCarlo simuleringene for frekvendeleren i figur 6.2.2.1 for V_{DD} = 180 mV. Kun MonteCarlo simuleringene for frekvensdeleren realisert med Negativ klokkeflanke D Flip Flop er tatt med for illustrasjonsskyld, og et utvalg av de 30 MonteCarlo simuleringene.



Figur B.7.1: MonteCarlo simulering av frekvensdeleren ved -40 ^{o}C



Figur B.7.2: MonteCarlo simulering av frekvensdeleren ved 27 ^{o}C



Figur B.7.3: MonteCarlo simulering av frekvensdeleren ved 85 ^{o}C

B.8 Kommentarer til valg av MonteCarlo simuleringer

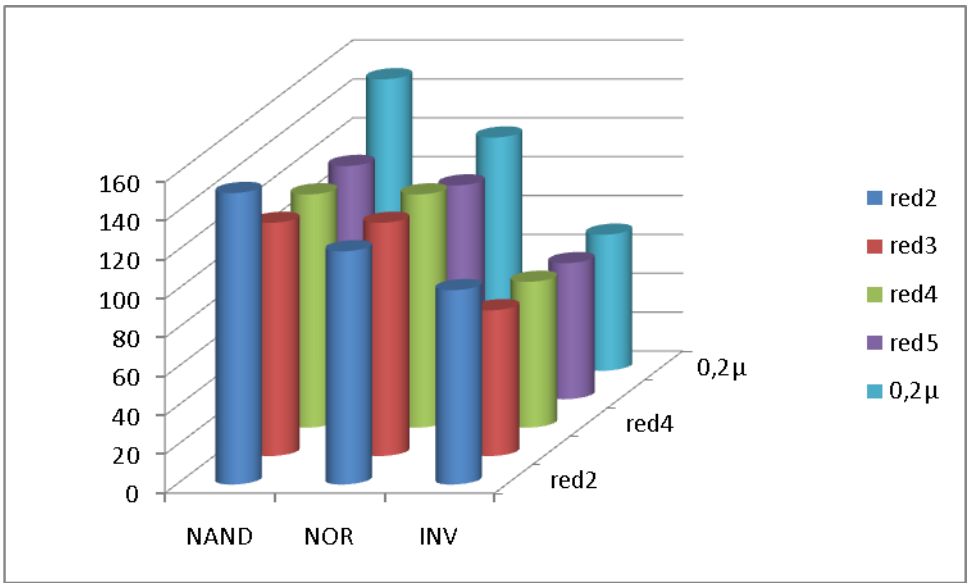
Kun et utvalg av alle MonteCarlo simuleringene er tatt med for enkelhetsskyld, og for å illustrere prinsippene ved simuleringene. Grunnen til at vi har valgt å ta med simuleringsresultatene for INVERTER'en, NAND2 porten og NOR2 er at det er ved hjelp av disse tre portene de litt større kretsene er laget med. Når det gjelder ALU'en så er ALU'en som er laget med L = 0,1 µm transistorer som er tatt med, siden det er den versjonen som er "hovedversjonen" uten forbedringer. For frekvensdelerene er det de som er realisert med negativ klokkeflanke D Flip Flop som er tatt med på grunn av at det er frekvensdelerene som er laget ved hjelp av dem som fungerer best, noe som førte til at vi tok med MonteCarlo simuleringene til negativ klokkeflanke D flip flop'en også.

APPENDIX C

Diverse figurer

C.1 Redundans i INVERTER, NAND2- og NOR2 porter

Figur C.1.1 viser laveste V_{DD} (med 100% "yield" med T = -40 ^{o}C, 27 ^{o}C, 85 ^{o}C) som de forskjellige portene fungerer med for forskjellige redundanser (fra 2- 5), og sammenligner det med dobbelgatelengde.



Figur C.1.1: Sammenligning av redundans og dobbelgatelengde for INVERTER, NAND2 og NOR2