INF 1400 - Oppsummering

Dawid Kuleczko

November 14, 2016

Temaer:

Digital representasjon og binær logikk

Boolsk algebra

Karnaugh diagram

Kobinatorisk logikk

Sekvensiell logikk

Tilstandsmaskiner

Datamaskinarkitektur

CMOS

Digital representasjon og binær logikk:

Tallsystemer:

I dette kurset skal vi legge spesielt vekt på binære tall, men heksadesimale tall og octale tall, samt tall fra andre tallsystemer, kan også forekomme.

Til daglig bruker vi desimal, altså 10-tallsystemet. Et desimalt tall er representert ved symbolene fra 0 til 9.

eks. $(951)_{dec} = 9 \cdot 10^2 + 5 \cdot 10^1 + 1 \cdot 10^0$

Et binært tall er representert ved symbolene 0 og 1.

eks. $(101)_{bin} = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$

Et heksadesimalt tall er representert ved symbolene fra 0 til F der tallene etter 9 er: 10 = A 11 = B 12 = C 13 = D 14 = E 15 = F

eks. $(5BF)_{hex} = 5 \cdot 16^2 + 11 \cdot 16^1 + 15 \cdot 16^0$

Et oktalt tall er representert ved symbolene fra 0 til 8.

eks. $(853)_{oct} = 8 \cdot 8^2 + 5 \cdot 8^1 + 3 \cdot 3^0$

Konvertering:

Konvertering fra grunntall r til desimaltall:

Generelt: $(...a_2a_1a_0, a_{-1}a_{-2}...) = ... + a_2 \cdot r^2 + a_1 \cdot r^1 + a_0 \cdot r^0, a_{-1} \cdot r^{-1} a_{-2} \cdot r^{-2} + ..$

eks. $(1A5, 1C)_{16} = 1 \cdot 16^2 + A \cdot 16^1 + 5 \cdot 16^0 + 1 \cdot 16^{-1} + 12 \cdot 16^{-2} = (421, 109375)_{10}$

Konvertering av desimal til binær:

Eks. $(23)_{10}$

$$23/2 = 11 + 1/2 \qquad a_0 = 1$$

$$11/2 = 5 + 1/2 \qquad a_1 = 1$$

$$5/2 = 2 + 1/2 \qquad a_2 = 1$$

$$2/2 = 1 + 0/2 \qquad a_3 = 0$$

$$1/2 = 0 + 1/2 \qquad a_4 = 1$$

Vi leser det binære tallet nedfra $(10111)_2$

Konvertering fra desimal til grunntall "r":

Samme måte som over bare med "r" isteden for 2.

Eks. $(12)_{10}$ til 3-tallsystemet:

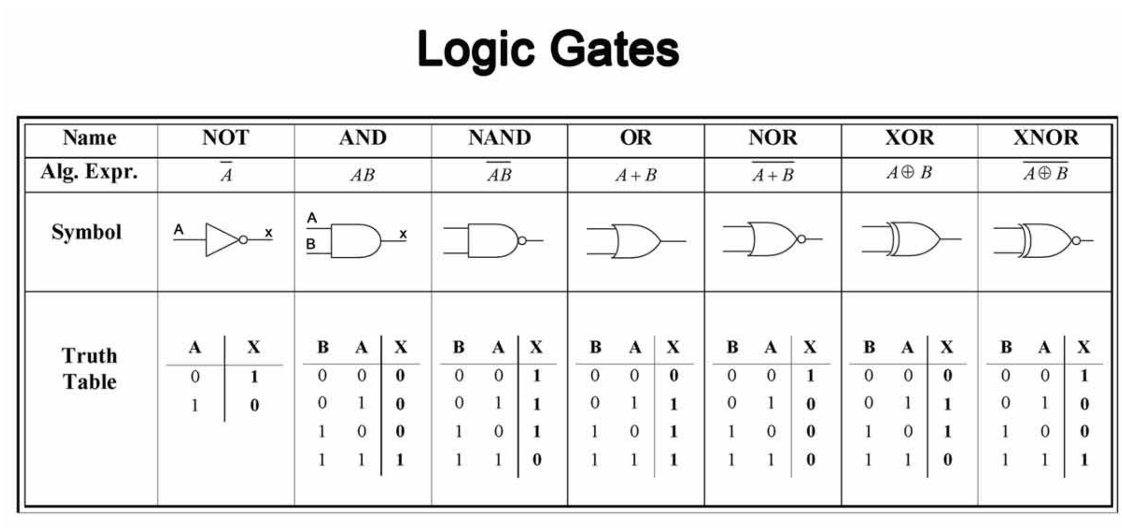
$$12/3 = 4 + 0/3 \qquad a_0 = 0$$

$$4/3 = 1 + 1/3 \qquad a_1 = 1$$

$$1/3 = 0 + 1/3 \qquad a_2 = 1$$

Tallet blir da $(110)_3$ i 3 tallsystemet.

Sannhetstabell:



Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR
Alg. Expr.	$\overline{A}$	AB	$\overline{AB}$	$A+B$	$\overline{A+B}$	$A \oplus B$	$\overline{A \oplus B}$
Symbol	A x	A B x					

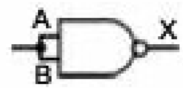
Truth Table

A	X (NOT)
0	**1**
1	**0**

B	A	X (AND)	X (NAND)	X (OR)	X (NOR)	X (XOR)	X (XNOR)
0	0	**0**	**1**	**0**	**1**	**0**	**1**
0	1	**0**	**1**	**1**	**0**	**1**	**0**
1	0	**0**	**1**	**1**	**0**	**1**	**0**
1	1	**1**	**0**	**1**	**0**	**0**	**1**

Figure 1: Logiske porter

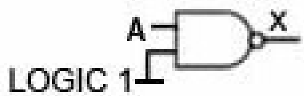
a.



NOT

A	B	X
0	0	1
1	1	0

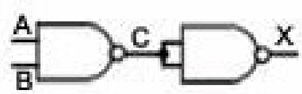
b.



NOT

A	B	X
0	1	1
0	1	1
1	1	0
1	1	0

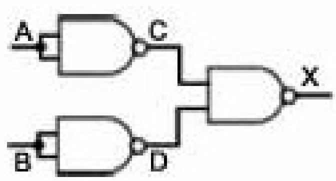
c.



AND

A	B	C	X
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

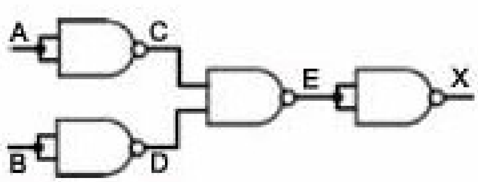
d.



OR

A	B	C	D	X
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	1

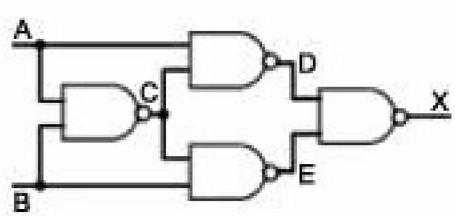
e.



NOR

A	B	C	D	E	X
0	0	1	1	0	1
0	1	1	0	1	0
1	0	0	1	1	0
1	1	0	0	1	0

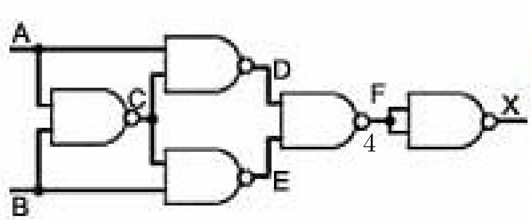
f.



XOR

A	B	C	D	E	X
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	0

g.



XNOR

A	B	C	D	E	F	X
0	0	1	1	1	0	1
0	1	1	1	0	1	0
1	0	1	0	1	1	0
1	1	0	1	1	0	1

Figure 2: Logiske porter som 2 inputs NAND-porter

Huntington's postulater

Huntington postulater

(P0)	Mengden {0,1} er lukket under "+" og "•"		lukket
(P1)	$x + x = x$	$x \bullet x = x$	
(P2)	$x + 0 = x$	$x \bullet 1 = x$	ident.el.
(P2b)	$x + 1 = 1$	$x \bullet 0 = 0$	
(P5)	$x + x' = 1$	$x \bullet x' = 0$	komplem.
(P6)	$0 \neq 1$		minst 2 el.
(P3)	$x + y = y + x$	$x \bullet y = y \bullet x$	kommutativ
(P4)	$x \bullet (y + z) = x \bullet y + x \bullet z$	$x + (y \bullet z) = (x+y) \bullet (x+z)$	distributiv
(P5)	$(x')' = x$		

Dualitet for postulatene:

Kan bytte "•" med + hvis man bytter "0" med "1"

Presedens:

Først utføres "0", så "´", så "•" og til slutt "+"

Figure 3: Huntington's postulater

Boolsk algebra:

DeMorgans teorem

$$(x \bullet y)' = x' + y'$$

$$(x + y)' = x' \bullet y'$$

På invertert form:

$$x \bullet y = (x' + y')'$$

$$x + y = (x' \bullet y')'$$

Figure 4: DeMorgans teorem

Komplement av funksjon

Inverterer begge sider og bruker DeMorgan

Eksempel:

$F = x'yz'+x'y'z$

$F' = (x'yz'+x'y'z)'$

$F' = (x'yz')'(x'y'z)'$

$F' = (x+y'+z)(x+y+z')$

Eksempel:

$F = x(y'z'+yz)$

$F' = (x(y'z'+yz))'$

$F' = x'+(y'z'+yz)'$

$F' = x'+(y'z')'(yz)'$

$F' = x'+(y+z)(y'+z')$

Figure 5: Komplement av funkson

Regneregler - oversikt

$x + 0 = x$	$x \bullet 1 = x$
$x + x' = 1$	$xx' = 0$
$x + y = y+x$	$xy = yx$
$x + (y+z) = (x+y) + z$	$x\,(yz) = (xy)z$
$x(y+z) = xy + xz$	$x + (yz) = (x+y)(x+z)$
$x + x = x$	$x \bullet x = x$
$x + 1 = 1$	$x \bullet 0 = 0$
$x + xy = x$	$x(x+y) = x$
$(x+y)' = x'y'$	$(xy)' = x' + y'$

Figure 6: Regne regler for boolsk algebra

Boolske funksjoner med sannhetstabell

En boolsk funksjon kan visualiseres i en sannhetstabell. En gitt funksjon har kun en sannhetstabell Men, en gitt sannhetstabell har uendelig mange funksjonsuttrykk.

Eksempel: F = x + y' z

XYZ	F
000	0
001	1
010	0
011	0
100	1
101	1
110	1
111	1

Forenkling av uttryk

En funksjon kan forenkles ved regneregler for å gjøre den lettere å håndere og implementere.

Forenklingseksempler

Eksempel:

F = x(x'+y)

F = xx'+xy

F = 0+xy

F = xy

Eksempel:

F = x+x'y

F = (x+x')(x+y)

F = 1(x+y)

F = x+y

Eksempel:

F = (x+y)(x+y')

F = x + (yy')

F = x + 0

F = x

Figure 7: Forenklingseksempler av noen funksjoner

Forenklingseksempler II

Eksempel:

$F = xy+x'z+yz$

$F = xy+x'z+yz(x+x')$

$F = xy+x'z+xyz+x'yz$

$F = xy(1+z)+x'z(1+y)$

$F = xy+x'z$

Eksempel:

$F = (x+y)(x'+z)(y+z)$

$F = (x+y)(x'+z)$ Dualitet

Figure 8: Forenklingseksempler av noen funksjoner

Maksterm/Minterm

Variable			Minterm		Maxterm	
x	y	z	Term	Designation	Term	Designation
0	0	0	x'y'z'	m_0	x+y+z	M_0
0	0	1	x'y'z	m_1	x+y+z'	M_1
0	1	0	x'yz'	m_2	x+y'+z	M_2
0	1	1	x'yz	m_3	x+y'+z'	M_3
1	0	0	xy'z'	m_4	x'+y+z	M_4
1	0	1	xy'z	m_5	x'+y+z'	M_5
1	1	0	xyz'	m_6	x'+y'+z	M_6
1	1	1	xyz	m_7	x'+y'+z'	M_7

Figure 9: Tabell for minterm og maksterm

Mintermer har notasjon m_x

$F(x,y,z) = \sum(m_3, m_6) = \sum(3,6) = x'yz' + xyz'$

Makstermer har notasjon M_x

$F(x,y,z) = \Pi(M_3, M_6) = \Pi(3,6) = (x+y'+z')(x'+y'+z)$

Minterm:

I en funksjon kan en binær variabel x opptre som x eller x'. En funksjon kan være gitt på "sum av produkt" form. Eksempel: F = xy + xy' + x

Hvert "produktledd" som inneholder alle variablene kalles en minterm. For to variable finnes det 4 forskjellige mintermer: xy + xy' + x'y + x'y' For 3 variable finnes det 2^3 forskjellige mintermer.

Hvis man generer en funksjon ut i fra sannhetstabellen får man en sum av mintermer

Eksempel: F = x'y'z + xy'z' + xyz'

XYZ	F
000	0
001	1
010	0
011	0
100	1
101	0
110	1
111	0

En sannhetstabell kan sees på som en liste av mintermer.

Maksterm:

En funksjon kan være gitt på "produkt av sum" form.

Eksempel: F = (x+y)(x+y')y Hvert "summeledd" som inneholder alle variablene kalles maksterm.

For to variable finnes det 4 forskjellige makstermer: (x'y)(x+y')(x'+y)(x'+y')

For n variable finnes det 2^n forskjellige makstermer.

Designprosedyre

Det er ikke alltid at det enkleste funksjonsuttrykket resulterer i den enkleste port-implementasjonen.

Ved forenkling på portnivå må man vite hvilke porter man har til rådighet, og så justere funksjonsuttrykket mot dette. (håndverk)

Generell design prosedyre

1.Bestem hvilke signal som er innganger og utganger

2.Sett opp sannhetstabell for alle inngangskombinasjoner

3.Generer funksjonsuttrykket som sum av mintermer

4.Tilpass / forenkle funksjonsuttrykket mot aktuelle porter

Karnaugh diagram:

Grafisk metode for forenkling av Boolske uttryk

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

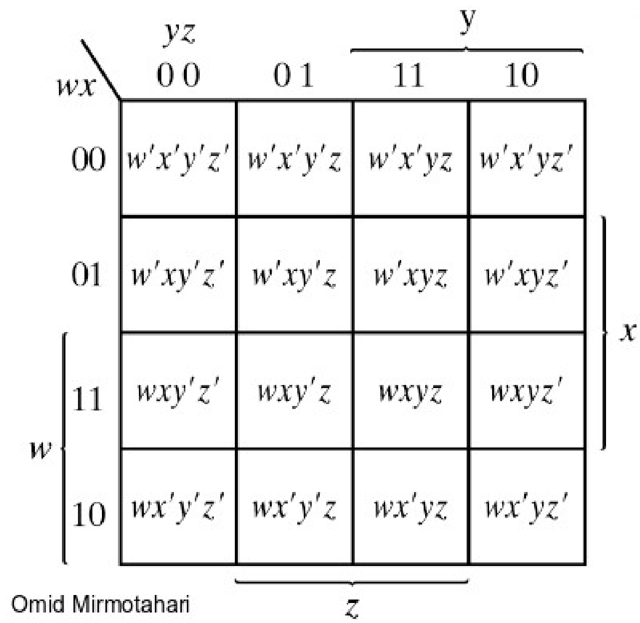


Figure 10: Karnaugh diagram med mintermene

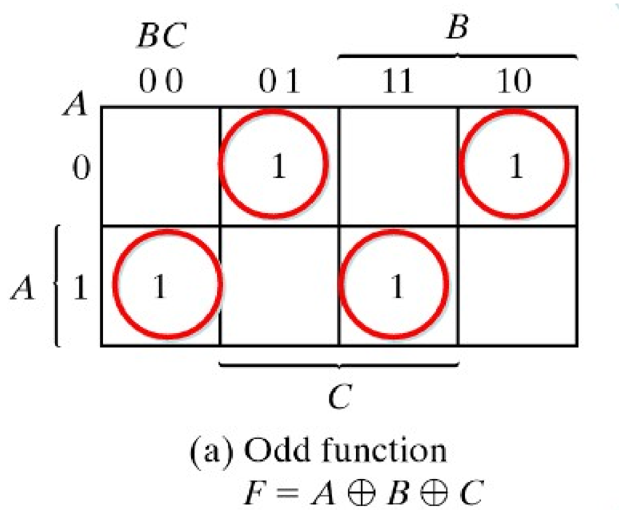


Figure 11: XOR

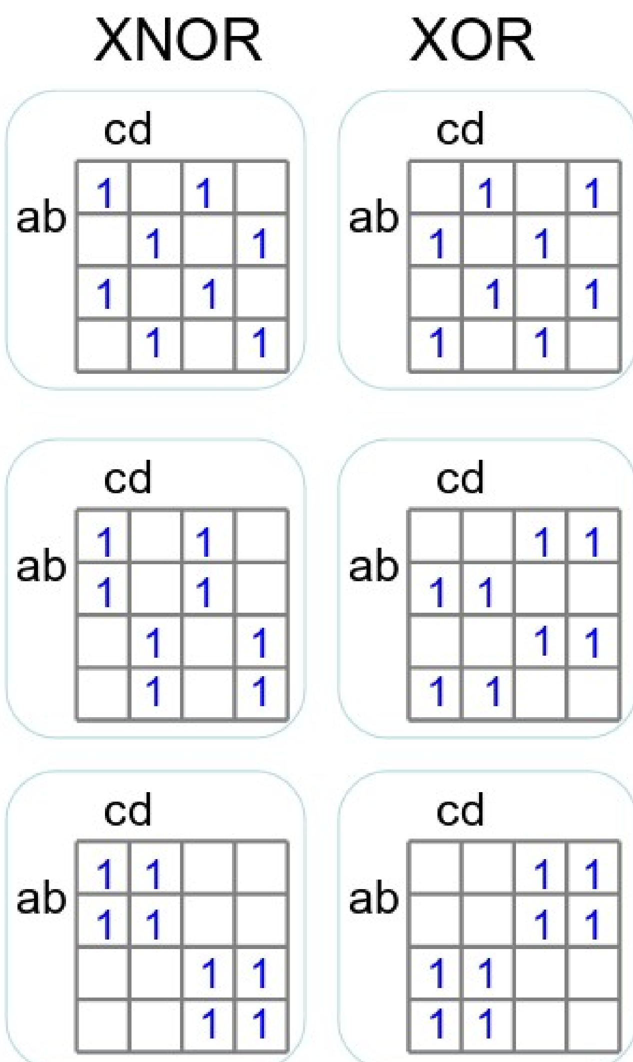


Figure 12: XNOR og XOR

Kobinatorisk logikk:

Binær addisjon:

Prosedyren for binær addisjon er identisk med prosedyren for desimal addisjon:

$$\text{Eks:} \begin{bmatrix} 0101 & \\ 1011 & + \\ ------- & \\ 10000 & \end{bmatrix} 5 + 11 = 16$$

Binær substraksjon:

7	0 1 1 1
6	0 1 1 0
5	0 1 0 1
4	0 1 0 0
3	0 0 1 1
2	0 0 1 0
1	0 0 0 1
0	0 0 0 0
-1	1 1 1 1
-2	1 1 1 0
-3	1 1 0 1
-4	1 1 0 0
-5	1 0 1 1
-6	1 0 1 0
-7	1 0 0 1
-8	1 0 0 0

Figure 13: Representasjon av binære negative tall

For å substrahere negative binære tall, bruker man toerkomplement metoden.

Det tallet som skal substraherers med må inverteres og plusses på 1, deretter skal det plusses med den andre tallet. Tallet til overs går ut.

$$\text{Eks:} \begin{bmatrix} 1101 & \\ 1011 & - \\ -------- & \end{bmatrix} \begin{bmatrix} 1101 & \\ 0101 & + \\ -------- & \\ (1)0010 & \end{bmatrix} \text{13-11} = 2$$

13+5=18 = 1 0 0 1 0

Binære addere:

Halvadder:

Halvaddere tar ikke mente inn.

Halvadder implementasjon

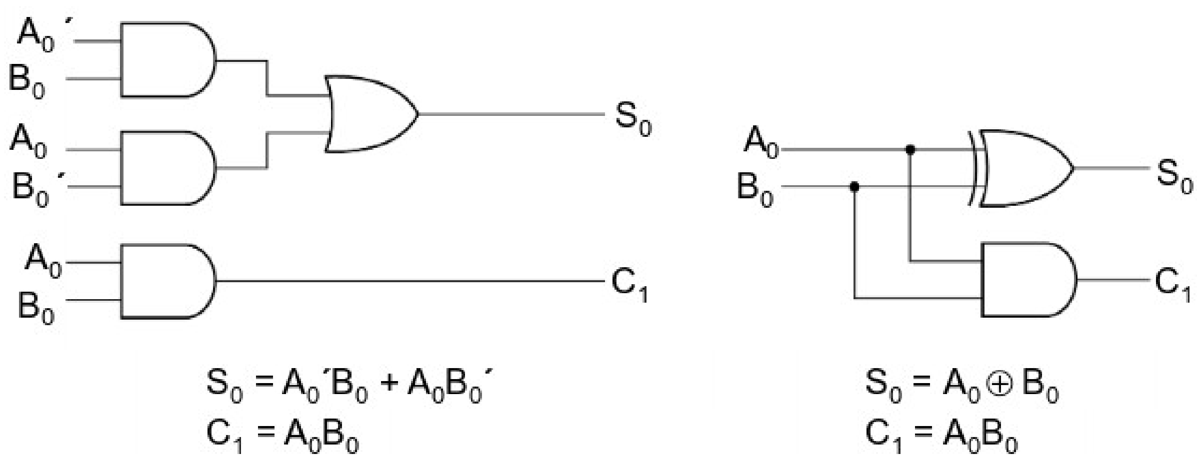


Figure 14: Halvadder implementasjon

Fulladder:

Fulladder tar mente inn.

Fulladder implementasjon

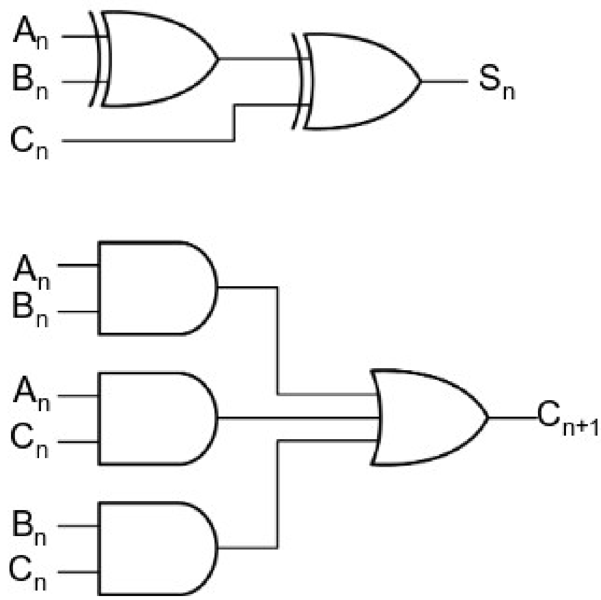


Figure 15: Fulladder implementasjon

Et adder system

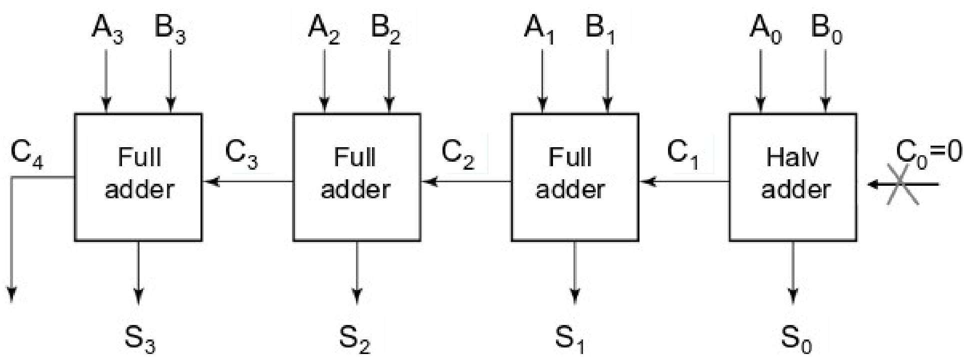


Figure 16: Et system av halv og fulladdere

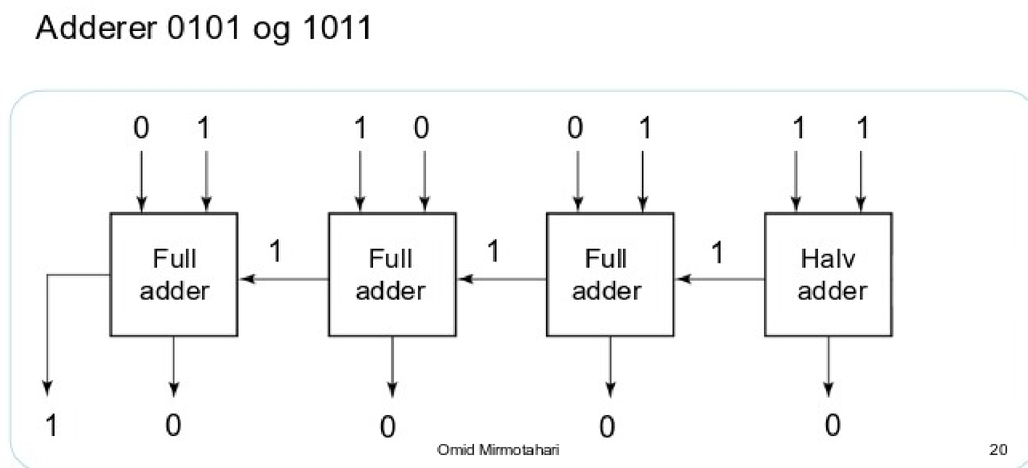


Figure 17: Et eksempel på addisjon av to binære 4-bits tall

"Carry Lookahead"

Ønsker å unngå menteforplantning – gir økt hastighet

G_i – generate: brukes i menteforplantningen P_i – propagate: påvirker menteforplantningen

Komparator:

Komparator – sammenligner to tall A og B

$3utganger : A = B, A > BogA < B$

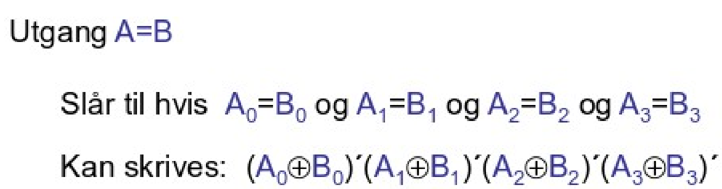


Figure 18: A=B

Utgang $A>B$ slår til hvis:

($A_3>B_3$) eller

($A_2>B_2$ og $A_3=B_3$) eller

($A_1>B_1$ og $A_2=B_2$ og $A_3=B_3$) eller

($A_0>B_0$ og $A_1=B_1$ og $A_2=B_2$ og $A_3=B_3$)

Kan skrives:

$$(A_3B_3') + (A_2B_2')(A_3 \oplus B_3)' + (A_1B_1')(A_2 \oplus B_2)'(A_3 \oplus B_3)' +$$

$$(A_0B_0')(A_1 \oplus B_1)'(A_2 \oplus B_2)'(A_3 \oplus B_3)'$$

Figure 19: $A > B$

Utgang $A<B$ slår til hvis:

($A_3<B_3$) eller

($A_2<B_2$ og $A_3=B_3$) eller

($A_1<B_1$ og $A_2=B_2$ og $A_3=B_3$) eller

($A_0<B_0$ og $A_1=B_1$ og $A_2=B_2$ og $A_3=B_3$)

Kan skrives:

$$(A_3'B_3) + (A_2'B_2)(A_3 \oplus B_3)' + (A_1'B_1)(A_2 \oplus B_2)'(A_3 \oplus B_3)' +$$

$$(A_0'B_0)(A_1 \oplus B_1)'(A_2 \oplus B_2)'(A_3 \oplus B_3)'$$

Figure 20: $A < B$

Dekoder:

Dekoder – tar inn et binært ord og gir ut alle mintermer Kan generere generelle logiske funksjoner direkte fra mintermene på utgangen Eksempel: 3bit inn/8bit ut

Eksempel: 3bit inn

Innganger			Utganger							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Figure 21: Sannhetstabell til en 3 bits decoder

Enkoder:

Enkoder er motsatt av dekoder

Eksempel: 8x3 enkoder

Innganger								Utganger
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x y z
1	0	0	0	0	0	0	0	0 0 0
0	1	0	0	0	0	0	0	0 0 1
0	0	1	0	0	0	0	0	0 1 0
0	0	0	1	0	0	0	0	0 1 1
0	0	0	0	1	0	0	0	1 0 0
0	0	0	0	0	1	0	0	1 0 1
0	0	0	0	0	0	1	0	1 1 0
0	0	0	0	0	0	0	1	1 1 1

Figure 22: Sannhetstabell for en 8x3 enkoder

Prioritets-enkoder:

Hvis flere "1"ere inn - ser kun på inngang med høyst indeks (prioritet)

Multiplekser (MUX):

Velger hvilke innganger som slippes ut

Hver inngang kan bestå av ett eller flere bit

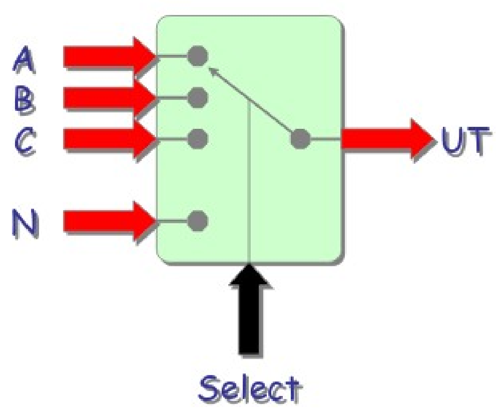


Figure 23: MUX

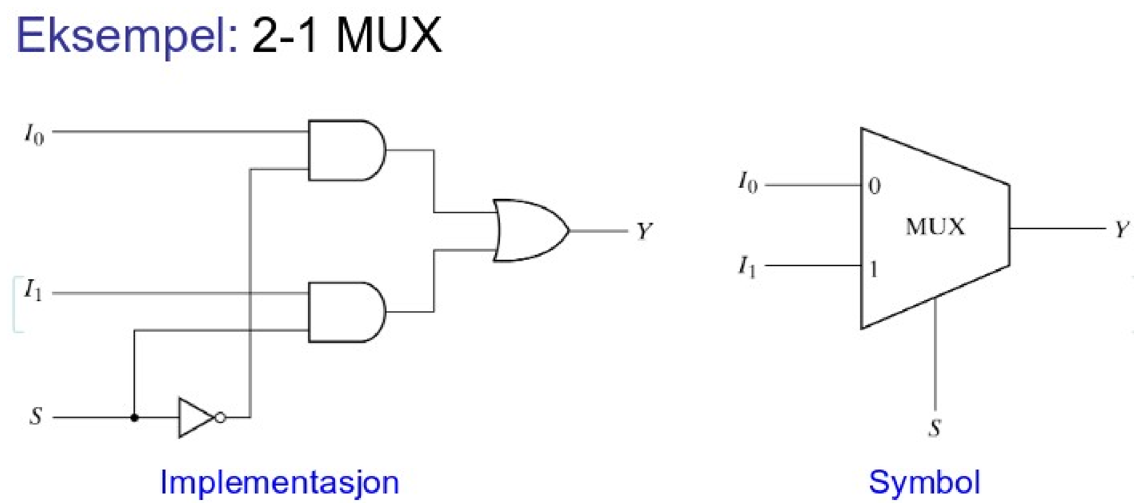


Figure 24: En 2-1 MUX

Demulitplekser:

Motsatt av MUX, velger hvilke utganger som slippes ut

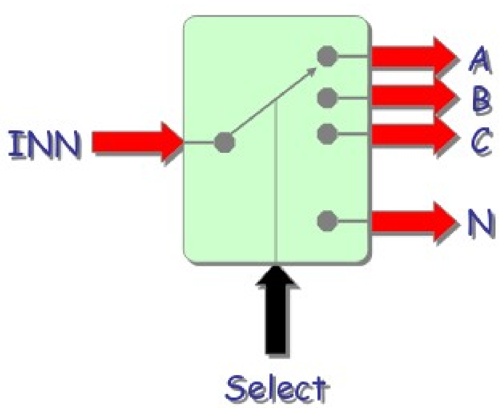


Figure 25: Demultiplekser

Aritmetisk logisk enhet (ALU):

En elektronisk krets som utfører aritmetiske og logiske operasjoner

Sekvensiell logikk:

Kombinatorisk logikk: Utgangsverdiene er entydig gitt av nåværende kombinasjon av inngangsverdier.

Sekvensiell logikk: Inneholder hukommelse (låsekretser) Utgangsverdiene er gitt av nåværende kombinasjon av inngangsverdier, samt sekvensen av tidligere inngangs-/utgangsverdier.

Tilstandsmaskiner:

Datamaskinarkitektur:

CMOS