Aktivera Kursens mål: LV2 Fo4

- Konstruera en dator mha grindar och programmera denna

Aktivera Förra veckans mål:

- Beskriva grindar och de verktyg som behövs under konstruktionsarbetet av datorn
- Hur kodas tal och tecken i datorn

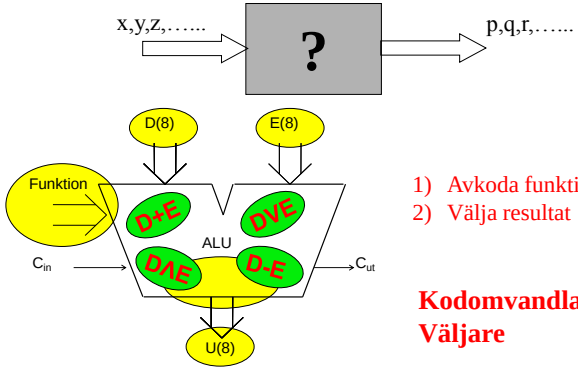
Veckans mål:

- Konstruktruera de olika kombinatoriska nät som ingår i en dator. Exempel på sådana nät är väljare, kodomvandlare och ALU (beräkningsenheten i processorn).
- Studera hur addition/subtraktion utförs och signalera vid fel (flaggor)
- Konstruera register som används för att lagra data i datorn.
- Koppla samman register och ALU med bussar till en enkel dataväg (som är i processorn)

Dagens mål, Du ska kunna…..

- Konstruera Kodomvandlare
- Först innebörden av och använda Don't care – termer
- Konstruera Väljare
- Förstå Fördelare
- Konstruera och använda Heladderare
- Koda och använda tal med och utan tecken

4 Kombinatoriska nät

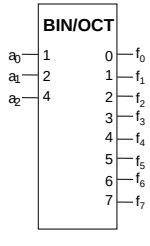


1) Avkoda funktion
2) Välja resultat

Kodomvandlare
Väljare

Kodomvandlare S4.11-12

Avkodare (eng **decoder**)



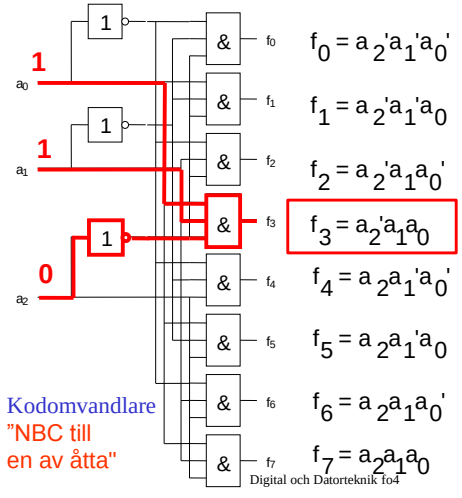
Figur 4.10 Prosamsymbol för "**NBC till en av åtta**" "Binary/Octal" kodomvandlare.

Binärsiffra in -
EN aktiv utsignal

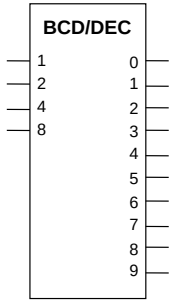
Ex: IN= 110
UT= f_6

Tabell 4.6 Funktionstabell för "NBC till en av åtta" kodomvandlare (avkodare).

3 IN			8 UT								
a_2	a_1	a_0	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	
0	0	0	1	0	0	0	0	0	0	0	$f_0 = a_2'a_1'a_0'$
0	0	1	0	1	0	0	0	0	0	0	$f_1 = a_2'a_1'a_0$
0	1	0	0	0	1	0	0	0	0	0	$f_2 = a_2'a_1a_0'$
0	1	1	0	0	0	1	0	0	0	0	$f_3 = a_2'a_1a_0$
1	0	0	0	0	0	0	1	0	0	0	$f_4 = a_2a_1'a_0'$
1	0	1	0	0	0	0	0	1	0	0	$f_5 = a_2a_1'a_0$
1	1	0	0	0	0	0	0	0	1	0	$f_6 = a_2a_1a_0'$
1	1	1	0	0	0	0	0	0	0	1	$f_7 = a_2a_1a_0$



Kodomvandlare



Figur 4.12 Prosamsymbol för
"**NBCD till en av tio**"
”BCD/Decimal" kodomvandlare.

NBCD-tal in -
EN aktiv utsignal

Ex: IN= 1001
UT= f_9

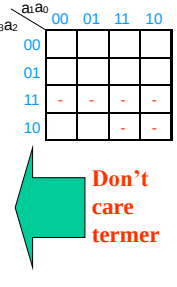
Ofullständig funktionstabell för "NBCD till en av tio“ S4.15

4 IN				10 UT										
a_3	a_2	a_1	a_0	dec	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	2	0	0	1	0	0	0	0	0	0	0
0	0	1	1	3	0	0	0	1	0	0	0	0	0	0
0	1	0	0	4	0	0	0	0	1	0	0	0	0	0
0	1	0	1	5	0	0	0	0	0	1	0	0	0	0
0	1	1	0	6	0	0	0	0	0	0	1	0	0	0
0	1	1	1	7	0	0	0	0	0	0	0	1	0	0
1	0	0	0	8	0	0	0	0	0	0	0	0	1	0
1	0	0	1	9	0	0	0	0	0	0	0	0	0	1

Tabell 4.7 Fullständig funktionstabell för "NBCD till en av tio“

Tabell 4.7

a_3	a_2	a_1	a_0	dec	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	2	0	0	1	0	0	0	0	0	0	0
0	0	1	1	3	0	0	0	1	0	0	0	0	0	0
0	1	0	0	4	0	0	0	0	1	0	0	0	0	0
0	1	0	1	5	0	0	0	0	0	1	0	0	0	0
0	1	1	0	6	0	0	0	0	0	0	1	0	0	0
0	1	1	1	7	0	0	0	0	0	0	0	1	0	0
1	0	0	0	8	0	0	0	0	0	0	0	0	1	0
1	0	0	1	9	0	0	0	0	0	0	0	0	0	1
1	0	1	0	-	-	-	-	-	-	-	-	-	-	-
1	0	1	1	-	-	-	-	-	-	-	-	-	-	-
1	1	0	0	-	-	-	-	-	-	-	-	-	-	-
1	1	0	1	-	-	-	-	-	-	-	-	-	-	-
1	1	1	0	-	-	-	-	-	-	-	-	-	-	-
1	1	1	1	-	-	-	-	-	-	-	-	-	-	-



Karnaughdiagram

$f_0 = a_3'a_2'a_1'a_0'$

$f_1 = a_3'a_2'a_1'a_0$

$f_2 = a_2'a_1a_0'$

$f_3 = a_2'a_1a_0$

$f_4 = a_2a_1'a_0'$

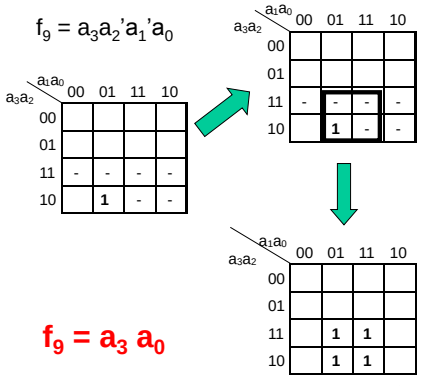
$f_5 = a_2a_1'a_0$

$f_6 = a_2a_1a_0'$

$f_7 = a_2a_1a_0$

$f_8 = a_3a_0'$

$f_9 = a_3a_2'a_1'a_0$



$f_9 = a_3\ a_0$

Don't care - termer

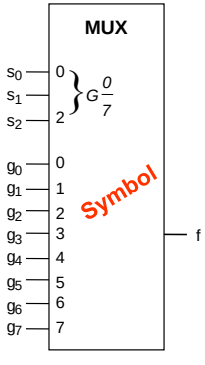
- En delmängd av "Alla kombinationer av insignaler" används. Övriga insignaler ger upphov till Don't care-termer i Karnaughdiagrammet.
- En Don't care – term kan väljas som etta eller nolla.
- Resultat: Konstruktionen får färre grindar.
- OBS! Konstruktionen kan inte användas för ogiltiga insignaler.

- ***Studera upg 4.12 och 4.20 (+ facit) i blåa boken***

Väljare

S4.16

(eng **multiplexer)**



"1 av 8 väljare"

Funktionstabell

s_2	s_1	s_0	f
0	0	0	g_0
0	0	1	g_1
0	1	0	g_2
0	1	1	g_3
1	0	0	g_4
1	0	1	g_5
1	1	0	g_6
1	1	1	g_7

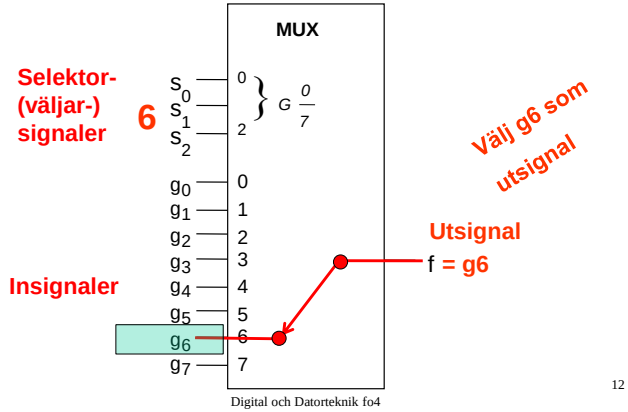
Funktion

$f = m_0g_0 + m_1g_1 + m_2g_2 + m_3g_3 + m_4g_4 + m_5g_5 + m_6g_6 + m_7g_7$

m_0 är motsvarande minterm ($m_0 = s'_2 s'_1 s'_0$)

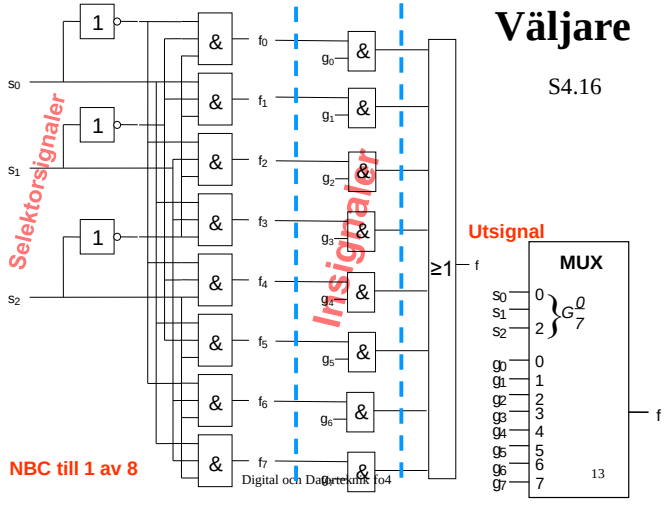
Väljare

S4.16



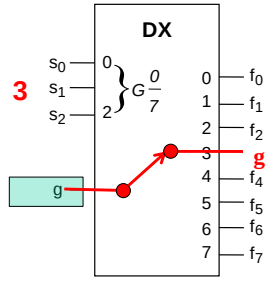
Väljare

S4.16



Fördelare (eng **demultiplexer**)

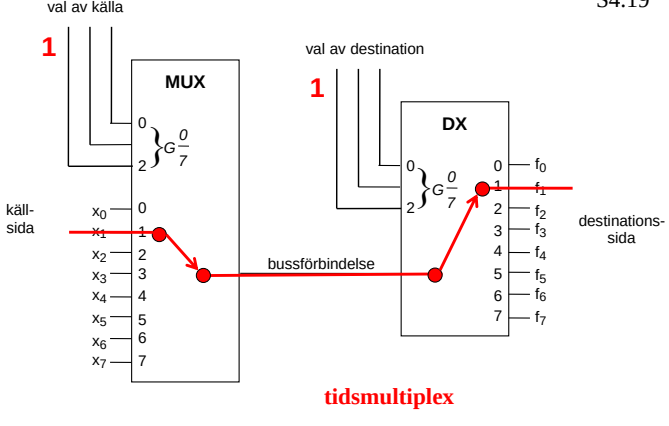
S4.19



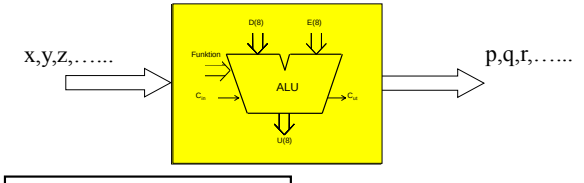
s_2	s_1	s_0	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	0	0	g	0	0	0	0	0	0	0
0	0	1	0	g	0	0	0	0	0	0
0	1	0	0	0	g	0	0	0	0	0
0	1	1	0	0	0	g	0	0	0	0
1	0	0	0	0	0	0	g	0	0	0
1	0	1	0	0	0	0	0	g	0	0
1	1	0	0	0	0	0	0	0	g	0
1	1	1	0	0	0	0	0	0	0	g

Bussförbindelse

S4.19



Kap 4 Kombinatoriska nät



Metodik
1. Formulera uppgiften
2. Funktionsbeskrivning
3. Lösning
4. Realisering

1. Beskrivning

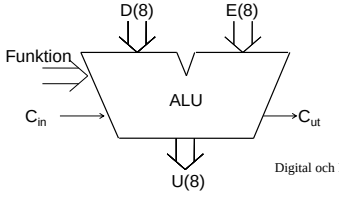
Konstruera en additionsenhet för två binära siffror.

Adderare

2. **Funktionsbeskrivning** *(ex 4.1 forts)*

Addera två binära siffror……???
Testa!!!!

$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0 \end{array} \quad \begin{array}{r} 0 \\ +\ 1 \\ \hline 1 \end{array} \quad \begin{array}{r} 1 \\ +\ 0 \\ \hline 1 \end{array} \quad \begin{array}{r} 1 \\ +\ 1 \\ \hline 0 \end{array} \qquad \begin{array}{r} a \\ +\ b \\ \hline s \end{array}$$



a	b	s
0	0	
0	1	
1	0	
1	1	

Adderare

2. **Funktionsbeskrivning** *(ex 4.1 forts)*

Addera två binära siffror……???
Testa!!!!

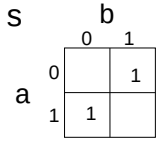
$$\begin{array}{r} \mathbf{0}\ 0 \\ +\ 0 \\ \hline 0 \end{array} \quad \begin{array}{r} \mathbf{0}\ 0 \\ +\ 1 \\ \hline 1 \end{array} \quad \begin{array}{r} \mathbf{0}\ 1 \\ +\ 0 \\ \hline 1 \end{array} \quad \begin{array}{r} \mathbf{1}\ 1 \\ +\ 1 \\ \hline 0 \end{array} \qquad \begin{array}{r} c\ a \\ +\ b \\ \hline s \end{array}$$

a	b	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

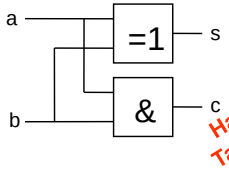
Adderare

3. **Lösning** *(ex 4.1 forts)*
c = ab
s = a'b+ab' → s = a ⊕ b

a	b	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



4.Realisering

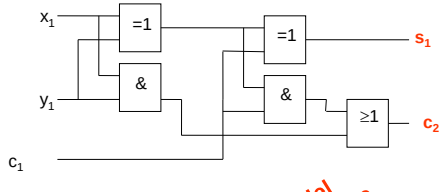


Halv Adderare
Tar ej Hänsyn till CARRY IN

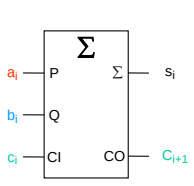
Adderare

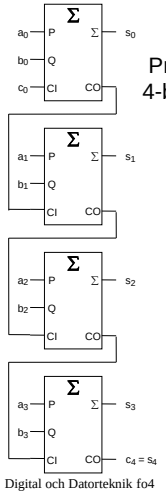
c_1	x_1	y_1	s_1	c_2
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\begin{array}{r} c_2\ c_1 \\ x_1\ x_0 \\ +\quad y_1\ y_0 \\ \hline =\quad s_1\ s_0 \end{array}$$

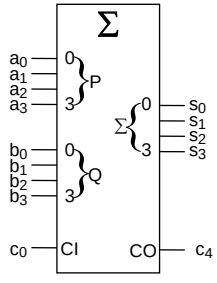


Hel Adderare





Prosamsymbolen för en 4-bitars binär heladderare.



0 1 1 1

$0\ 1\ 1\ 1_2$

$+\ 0\ 1\ 0\ 1_2$

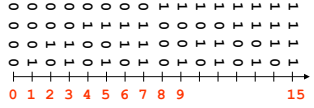
$= 1\ 1\ 0\ 0_2$

Tal MED och UTAN tecken

Arb s 36-37

$b_3\ b_2\ b_1\ b_0$

4 bitar - Representera 16 olika tal



	b_3	b_2	b_1	b_0
15	1	1	1	1
14	1	1	1	0
13	1	1	0	1
12	1	1	0	0
11	1	0	1	1
10	1	0	1	0
9	1	0	0	1
8	1	0	0	0
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

Binärtal = Tal utan tecken [0,15]

Arb s 36-37

Tal utan inbyggd tecken

		Tal med inbyggd tecken
15	1 1 1 1	
14	1 1 1 0	
13	1 1 0 1	
12	1 1 0 0	
11	1 0 1 1	
10	1 0 1 0	
9	1 0 0 1	
8	1 0 0 0	
7	0 1 1 1	7
6	0 1 1 0	6
5	0 1 0 1	5
4	0 1 0 0	4
3	0 0 1 1	3
2	0 0 1 0	2
1	0 0 0 1	1
0	0 0 0 0	0
	1 1 1 1	-1
	1 1 1 0	-2
	1 1 0 1	-3
	1 1 0 0	-4
	1 0 1 1	-5
	1 0 1 0	-6
	1 0 0 1	-7
	1 0 0 0	-8

Def 2-Komplement:

Arb s 37-39

Pos: Y = Y **Neg: (-Y) = 2^n - |Y| = Y_{2K}**

Att tvåkomplementera: Ex 4 bit: $2^n=2^4=16$

$\mathbf{2^n-Y} = 2^n\text{-}1\text{-}Y +1 = 16\text{-}1\text{-}Y+1 = 15\text{-}Y+1$ (= $Y_{1K}+1$)

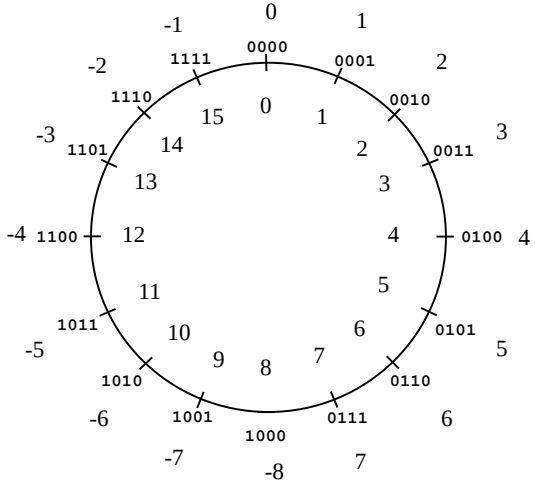
Ex Y=6: 0110 Hitta (-Y)

15_{10}	1111	
- Y	- 0110	
= $\mathbf{Y_{1K}}$	=1001	INVERSEN! **Def 1komp! Y_{1K}**
addera 1	+0001	
= $\mathbf{Y_{2K}}$	=1010	$\mathbf{Y_{1K} +1= Y_{2K}}$ **(-Y = 1010)**

(-6 = 1010)

Att subtrahera:

$X\text{-}Y = X + Y_{2K} = X + Y_{1K} +1$



Veckans mål: LV2 Fo 5

- Konstruktruera de olika kombinatoriska nät som ingår i en dator. Exempel på sådana nät är väljare, kodomvandlare och ALU (beräkningsenheten i processorn).
- Studera hur addition/subtraktion utförs och signalera vid fel
- Konstruera register som används för att lagra data i datorn.
- Koppla samman register och ALU med bussar till en enkel dataväg (som är i processorn)

Dagens mål, Du ska kunna….

- Förstå och använda Tvåkomplementsrepresentation
- Addera /Subtrahera med 2-komp-tal
- Ange Flaggbitar
- Förstå uppbyggnaden av en ALU (Bit-slice)

Def 2-Komplement:

Pos: Y = Y **Neg: (-Y) = 2^n - |Y| = Y_{2K}**

Att tvåkomplementera: Ex 4 bit: $2^n=2^4=16$

2^n-Y = 2^n-1-Y +1 = 16-1-Y+1 = 15-Y+1 (= Y_{1K}+1)

Ex Y=6: 0110 Hitta (-Y)

15_{10}	1111
- Y	- 0110
= $\mathbf{Y_{1K}}$	=1001

INVERSEN! **Def 1komp!**

Y_{1K}

addera 1	+0001
= $\mathbf{Y_{2K}}$	=1010

Y_{1K} +1= Y_{2K} **(-Y = 1010)**

(-6 = 1010)

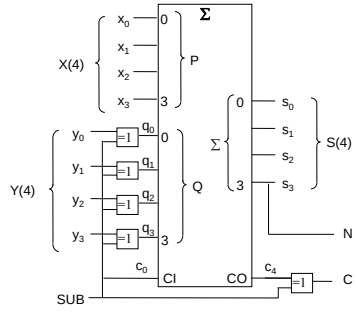
Arb s 40

Def Flaggor

Statusflaggor ut från ALU:n som indikerar om resultatet blev rätt eller fel

C	**Carry**	**Tal utan tecken**	**[0,15]** **(ADD : minnessiffra; SUB: lånesiffra)**
V	**Overflow**	**Tal med tecken**	**[-8,7]**
N	**Negative**	**Tal med tecken**	**[-8,7]**
Z	**Zero**	**Tal med och utan tecken**	

C=1: Resultatet av operationen blev ***fel*** för en operation ***utan tecken***
V=1: Resultatet av operationen blev ***fel*** för en operation ***med tecken***
N=1: Resultatet av operationen blev ***negativt*** för en operation ***med tecken***
Z=1: Resultatet av operationen blev ***noll***



Koppling för addition och subtraktion med 4-bitars tal. Flaggorna N och C bildas också

2-komplementrepresentation av heltal

En lösning på problemet tal med tecken får man med 2- komplementrepresentation.

Denna innebär att man behåller positiva tal som de är.

Negativa tal byts ut mot 2-komplementet av motsvarande positiva tal.

Exempel.

För 8-bitars tal gäller:

+5 används som det är dvs $(00000101)_2$

-5 byts mot $2^8 - 5 =$

$256 - 5 = (100000000)_2 - (00000101)_2 = (11111011)_2$

-38 byts mot $2^8 - 38 =$

$256 - 38 = (100000000)_2 - (00100110)_2 = (11011010)_2$

Läs Ext 6

Systematisk genomgång av addition och subtraktion med 2-k aritmetik

Additionen X + Y skrivs som: $S = X + Y = (\pm|X|) + (\pm|Y|)$

Subtraktionen X – Y skrivs som: $S = X - Y = (\pm|X|) - (\pm|Y|)$

Positiva heltal eller 0 anges som: $(+|X|)$ och då gäller att $0 \leq |X| \leq 127$

Negativa heltal anges som: $(-|X|)$ och då gäller att $1 \leq |X| \leq 128$

Addition med 2-komplementaritmetik

Följande fyra fall förekommer: (1c är lika med 1b om X och Y byter plats.)

1a. $S = X + Y = (+|X|) + (+|Y|)$ 1b. $S = X + Y = (+|X|) + (-|Y|)$

1c. $S = X + Y = (-|X|) + (+|Y|)$ 1d $S = X + Y = (-|X|) + (-|Y|)$

Sammanfattning av addition och subtraktion med 2-komplementaritmetik

Vanlig binär addition och subtraktion kan användas för tal med 2-komplementrepresentation.

Overflow kan inträffa och innebär att gränsen för det tillgängliga talområdet har överskridits.

Overflow kan inträffa i fallen 1a, 1d, 2b och 2c.

I fallen 1a och 1d inträffar overflow vid addition av tal med lika tecken om summan får motsatt tecken.

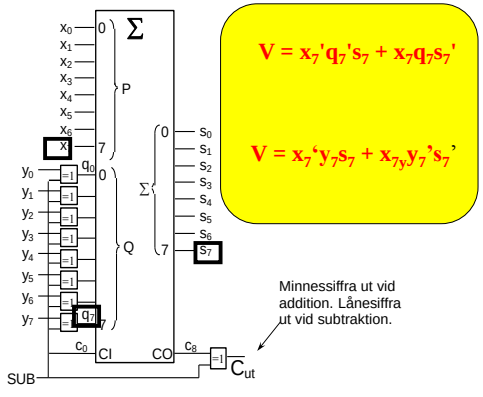
(Pos + Pos = Neg eller Neg + Neg = Pos)

I fallet 2b inträffar overflow vid subtraktion av ett negativt tal från ett positivt tal om skillnaden tolkas som negativ.

(Pos - Neg = Neg)

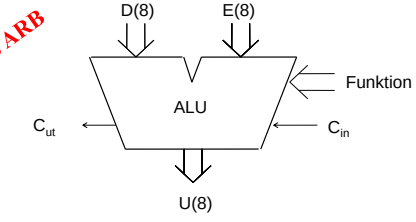
I fallet 2c inträffar overflow vid subtraktion av ett positivt tal från ett negativt tal om skillnaden tolkas som positiv.

(Neg - Pos = Pos)



ALU

Se ARB



ADD – SUB – AND - OR

Veckans mål:

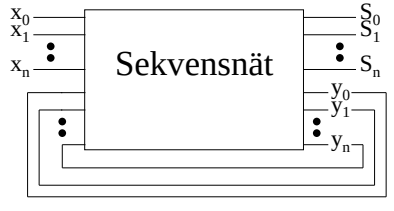
Fo 6

- Konstruktruera de olika kombinatoriska nät som ingår i en dator. Exempel på sådana nät är väljare, kodomvandlare och ALU (beräkningsenheten i processorn).
- Studera hur addition/subtraktion utförs och signalera vid fel (flaggor)
- Konstruera register som används för att lagra data i datorn.
- Koppla samman register och ALU med bussar till en enkel dataväg (som är i processorn)

Dagens mål, Du ska kunna…..

- Förstå vad sekvensnät är (och senare konstruera sekvensnät)
- Konstruera olika Latch:ar och använda dessa
- Förstå och använda vippor
- Förstå uppbyggnaden av register och hur dessa kopplas samman med hjälp av bussar.
- Förstå och använda 3-state-logik
- Använda RTN-beskrivning
- Koppla ihop en enkel dataväg (Systemexempel)

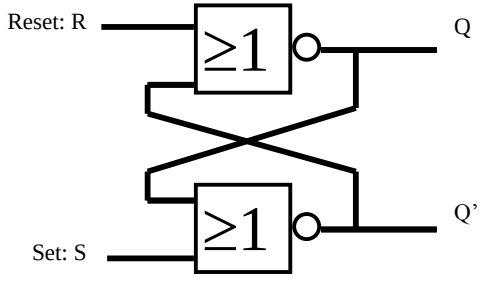
Läs smart! Lär dig mer!



- Återkoppling (feedback)
- Minnesfunktion

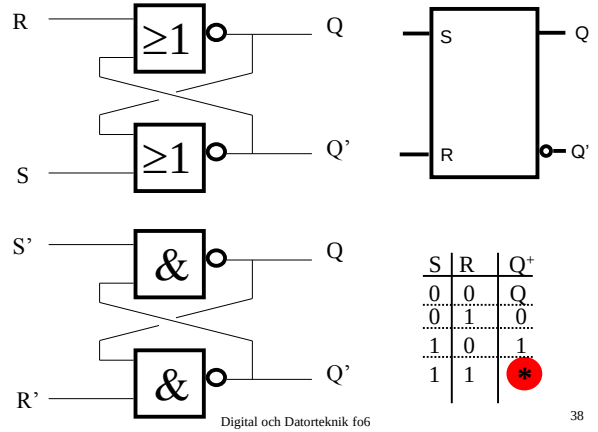
Låskretsar – (Latch'ar)

S5.3



SR - Latch

S5.4/6

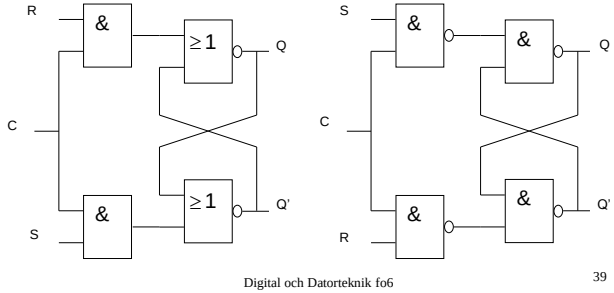


S	R	Q^+
0	0	Q
0	1	0
1	0	1
1	1	*

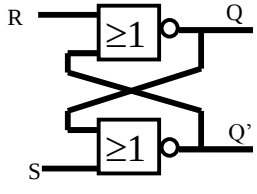
S5.7

Grindad SR-latch

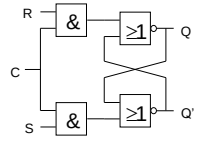
Ofta förses SR-latchar med en tredje ingång, till vilken en **styrpuls C** ansluts
Härvid erhålls en s k **grindad SR-latch**.



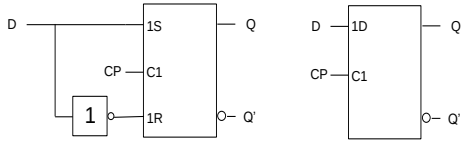
Latch'ar



S	R	Q^+
0	0	Q
0	1	0
1	0	1
1	1	*

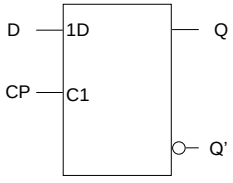


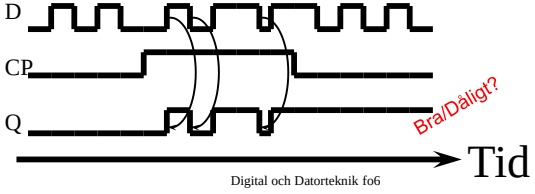
Klockad SR-latch.



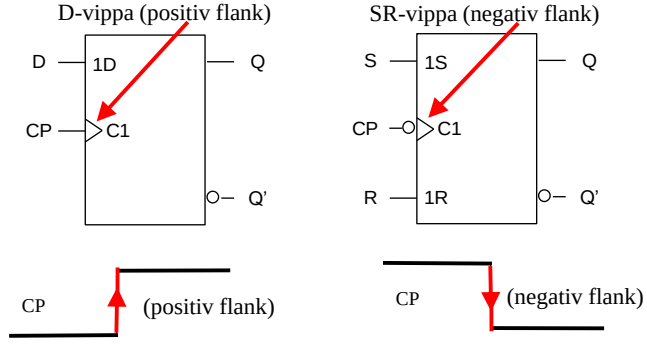
Klockad D-latch.

D	Q^+
0	0
1	1



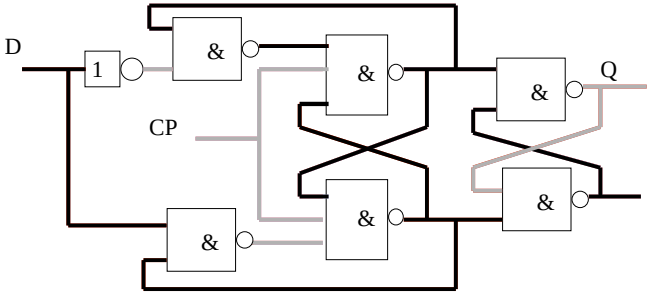


Flanktriggade vippor



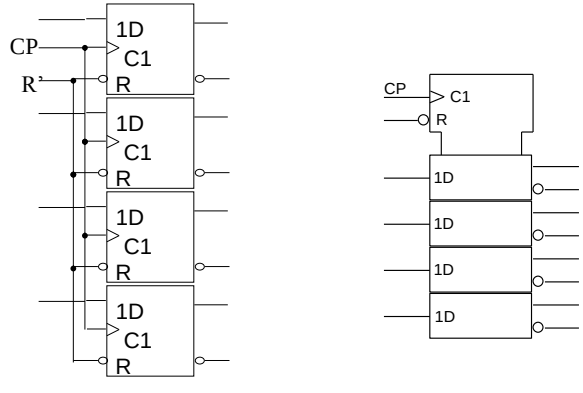
Flanktriggad Vippa (D Vippa)

S5.14

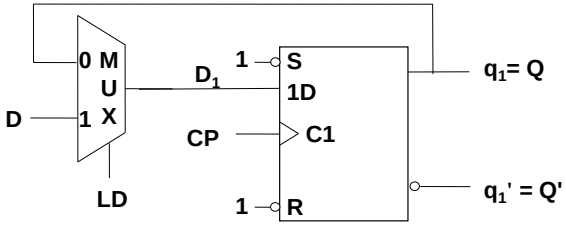


4-bitars register med möjlighet för nollställning.

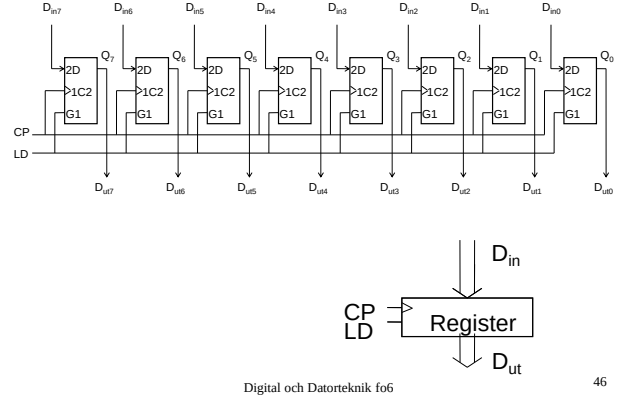
S5.21

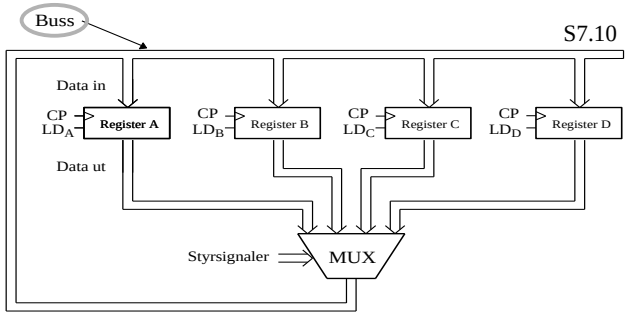


D-vippa med ”Load Enable”



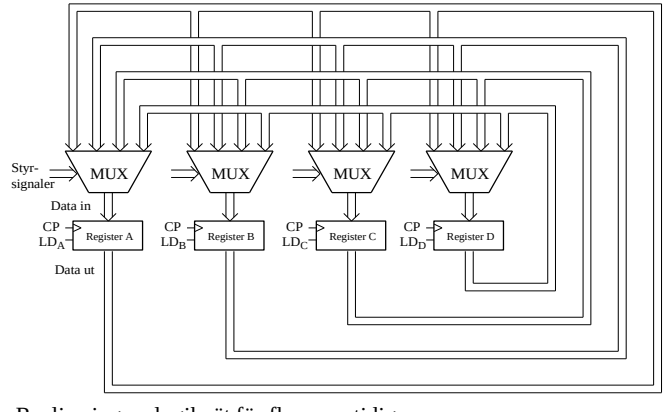
Register och bussar





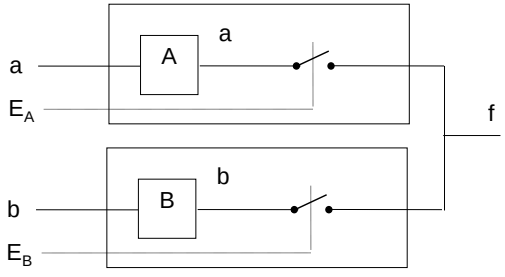
Enklare logiknät för dataöverföring mellan enheter.

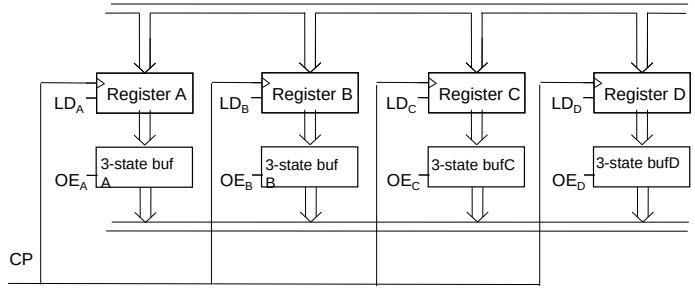
Vi skall ha möjlighet att göra **dataöverföring**.

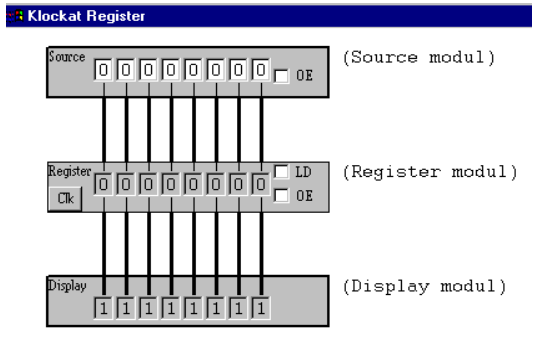


Realisering av logiknät för flera samtidiga dataöverföringar mellan register.

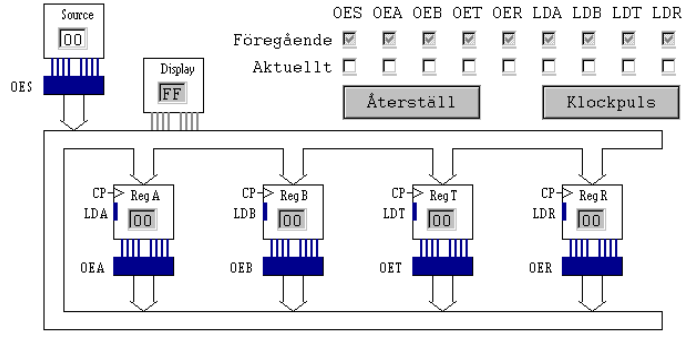
”Three-State”



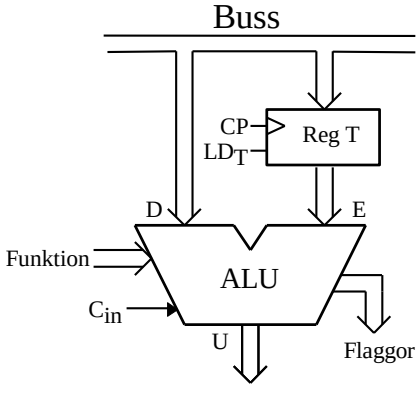




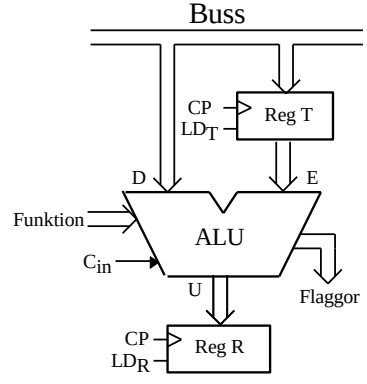
RTN: Register Transfer Notation



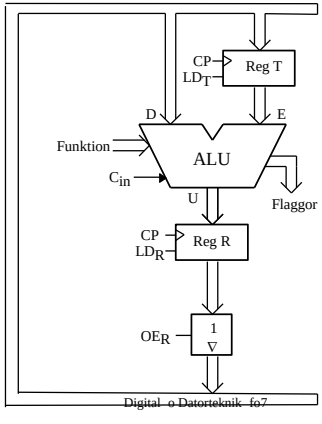
Användning av temporärregister (T) för lagring av indata till ALU.



Användning av resultatregister (R) för lagring av utdata från ALU.



Anslutning av resultatregister (R) till buss.



Logiknät för databehandling med aritmetik-logikenhet (ALU).