Lösningsförslag övningsuppgifter 2023-03-23

OBS! Lite information om de Morgans Teorem finns längst bak i dokumentet!

1. Omvandla följande binära tal till deras respektive osignerade decimala samt hexadecimala motsvarigheter:

a) $0001\ 1010_2$
b) $0111\ 1111_2$
c) $1101\ 0011_2$
d) $1111\ 1110_2$

Lösning

För omvandling till decimal form, summera värdet av samtliga ettor, där $1111\ 1111_2 = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1$.
För omvandling till hexadecimal form, ersätt fyra bitar 0000 – 111 med motsvarande hexadecimala tal 0 – F.

a) $0001\ 1010_2 = 32 + 8 + 2 = 42_{10} = 1A_{16}$
b) $0111\ 1111_2 = 64 + 32 + 16 + 8 + 4 + 2 + 1 = 127_{10} = 7F_{16}$
c) $1101\ 0011_2 = 128 + 64 + 16 + 2 + 1 = 211_{10} = D3_{16}$
d) $1111\ 1110_2 = 128 + 64 + 32 + 16 + 8 + 4 + 2 = 254_{10} = FE_{16}$

2. Omvandla följande tal till deras respektive binära motsvarigheter:

a) 49_{10}
b) 102_2
c) 212_{10}
d) $AC2_{16}$
e) $FA452C_{16}$

Lösning

För omvandling till decimal form, beräkna vilka ettor i ett givet 8-bitars binärt tal som behövs för att realisera talet, exempelvis är $25_{10} = 16 + 8 + 1$.

Börja från det närmaste tal som är lägre eller lika med det sökta talet, exempelvis 16 för 25. Kvar återstår sedan 25 – 16 = 9.
Närmaste tal är sedan 8, som läggs till, vilket medför att 9 – 8 = 1 återstår.
Slutligen läggs 1 till, vilket medför att resten är 0.
Därmed gäller att $25_{10} = 16 + 8 + 1$, vilket på binär form motsvarar $0001\ 1001_2$.

För omvandling till hexadecimal form, ersätt varje hexadecimalt tecken 0 - F med motsvarande bitar 0000 – 1111.

a) $49_{10} = 32 + 16 + 1 = 0011\ 0001_2$.
b) $102_{10} = 64 + 32 + 4 + 2 = 0110\ 0110_2$.
c) $212_{10} = 128 + 64 + 16 + 4 = 1101\ 0100_2$.
d) $AC2_{16} = 1010\ 1100\ 0010_2$.
e) $FA452C_{16} = 1111\ 1010\ 0100\ 0101\ 0010\ 1100_2$.

3. I nedanstående uppgifter ska 4-bitars 2-komplement användas:
a) Omvandla -6_{10} till dess 4-bitars binära motsvarighet.
b) Omvandla det signerade binära talet 1001_2 till dess decimala motsvarighet.

Lösning

Förhållandet mellan ett signerat decimal tal i och motsvarande osignerade tal u är följande:

$$\begin{cases} MSB = 0 => i = u \\ MSB = 1 => i = u - 2^n, \end{cases}$$

där n är antalet bitar och 2^n utgör det så kallade 2-komplementet. För 4-bitars tal utgör 2-komplementet $2^4 = 16$.

a) Omvandla först -6_{10} till motsvarande osignerade 4-bitars tal:

$$i = u - 2^n => u = i + 2^n = -6 + 2^4 = 10_{10}$$

Omvandla sedan det osignerade talet 10_{10} till binär form:

$$10_{10} = 8 + 2 = 1010_2$$

Därmed gäller att $-6_{10} = 1010_2$.

b) Omvandla först det binära talet 1001_2 till decimal osignerad form:

$$1001_2 = 8 + 1 = 9_{10}$$

Eftersom MSB = 1 är talet negativt på signerad form och ska därmed subtraheras med 2-komplementet, i detta fall 2^4. Omvandla det osignerade decimala talet till motsvarande signerade tal med 2-komlementet:

$$MSB = 1 => \ i = u - 2^n = 9 - 2^4 = -7_{10}$$

Därmed gäller att $1001_2 = -7_{10}$.

4. I nedanstående uppgifter ska 8-bitars 2-komplement användas:
a) Omvandla -104_{10} till dess 8-bitars binära motsvarighet.
b) Omvandla det signerade binära talet $1001\ 0100_2$ till dess decimala motsvarighet.

Lösning

Samma metodik som ovan, med skillnaden att 2-komplementet nu är $2^8 = 255$.

a) Omvandla först -104_{10} till motsvarande osignerade 8-bitars tal:

$$i = u - 2^n => u = i + 2^n = -104 + 2^8 = 152_{10}$$

Omvandla sedan det osignerade talet 152_{10} till binär form:

$$152_{10} = 128 + 16 + 8 = 1001\ 1000_2$$

Därmed gäller att $--104_{10} = 1001\ 1000_2$.

b) Omvandla först det binära talet $1001\ 0100_2$ till decimal osignerad form:

$$1001\ 0100_2 = 128 + 16 + 4 = 148_{10}$$

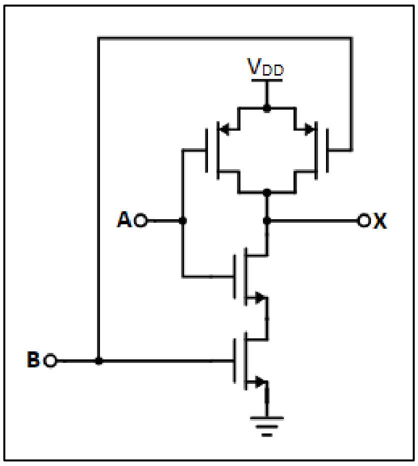
Eftersom MSB = 1 är talet negativt på signerad form och ska därmed subtraheras med 2-komplementet, i detta fall 2^8:

$$MSB = 1 => \ i = u - 2^n = 148 - 2^8 = -108_{10}$$

Därmed gäller att $1001\ 0100_2 = -108_{10}$.

5. Rita upp en NAND-grind med CMOS-transistorer och visa spänningsfallen i kretsen för insignaler AB = 00 – 11.

En logisk grind består av ett övre och ett nedre nät. Det nedre nätet kallas pulldown-nät och består utav NMOS-transistorer. För att realisera den logiska funktionen Y = A * B ska pulldown-nätet realisera funktionen A * B, vilket åstadkommes genom att seriekoppla två NMOS-transistorer med A och B som insignaler. Då måste båda A och B vara höga för att vägen mellan jord och utsignal X ska vara fri, annars är vägen spärrad.

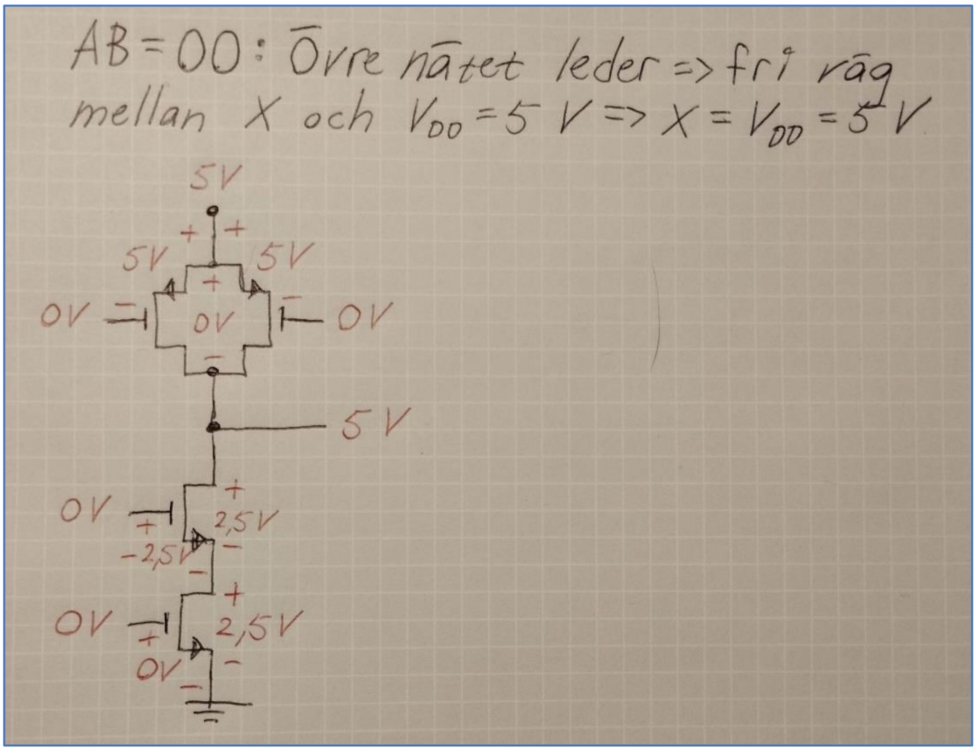


Figur 1: NAND-grind konstruerad med CMOS-transistorer.

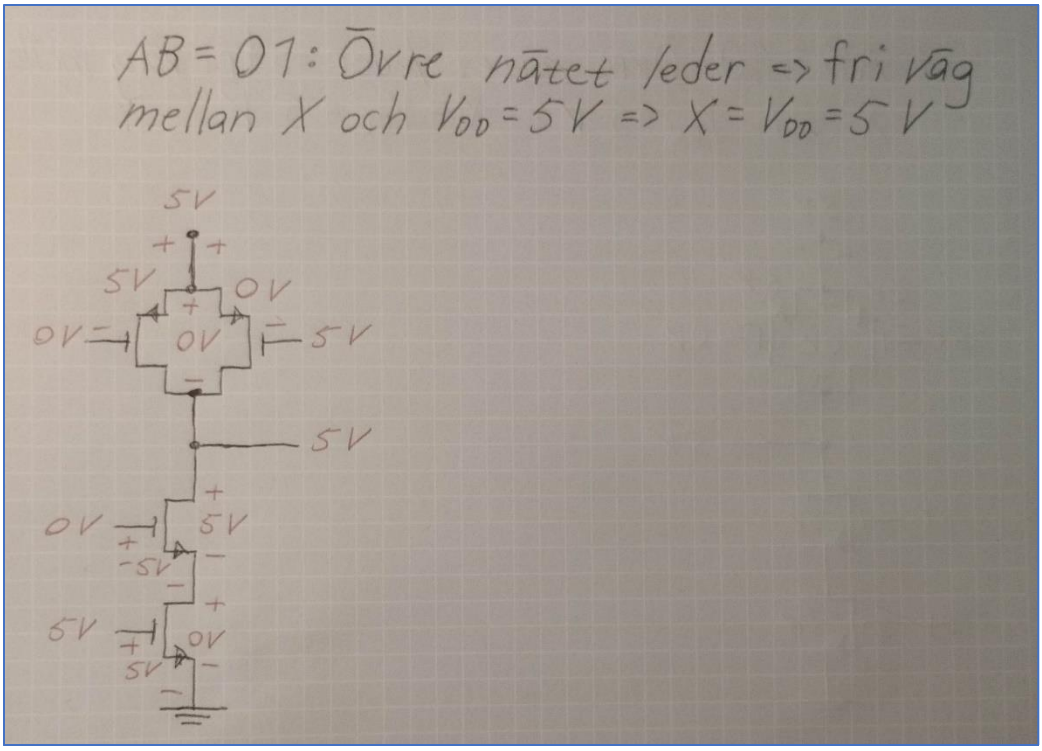
Det övre nätet kallas pullup-nät och består utav PMOS-transistorer. Eftersom detta ska utgöra rake motsatsen till pulldown-nätet (när det ena leder ska det andra spärra) sätts dessa till den inversa funktionen. De Morgans teorem ger att (A* B)’ = A’ + B’. Därmed placerar vi två PMOS-transistorer seriellt, där det räcker med att antingen A eller B är lika med 0 för att vägen mellan matningsspänningen V_{DD} och utsignal X ska bara fri, annars är båda vägar spärrade.

Genom att sätta ihop pulldown- och pullup-nätet erhålls NAND-grinden i figuren till höger.
Genom att koppla utgången till en NOT-grind hade en AND-grind realiserats.
Inverterande grindar är enklare att konstruera, då transistorswitchar i sin natur inverterar signalen, där hög insignal medför låg utsignal.

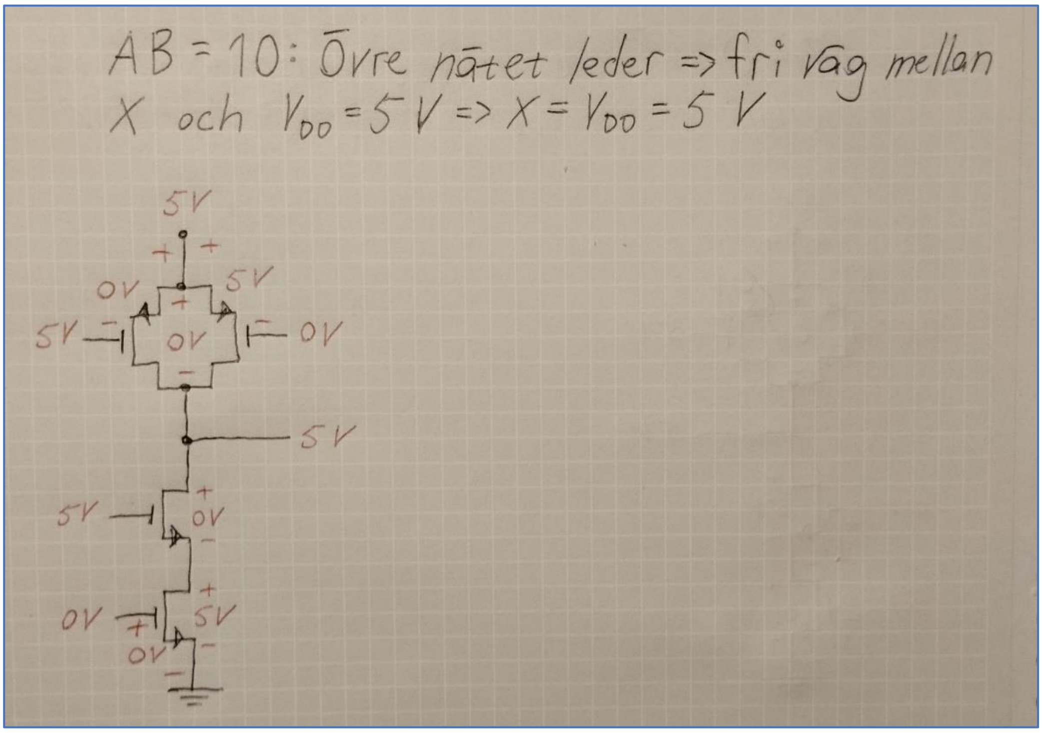
Se spänningar för respektive fall AB = 00 – 11 i figur 2 - 5 nedan. Samtliga transistorer antas ha en tröskelspänning U_T på 2,5 V (-2,5 V för PMOS-transistorerna, vilket medför att det krävs U_{GS} = 2,5 V mellan gate och source för NMOS-transistorerna och U_{SG} = 2,5 V mellan source och gate på PMOS-transistorerna för att de ska leda.



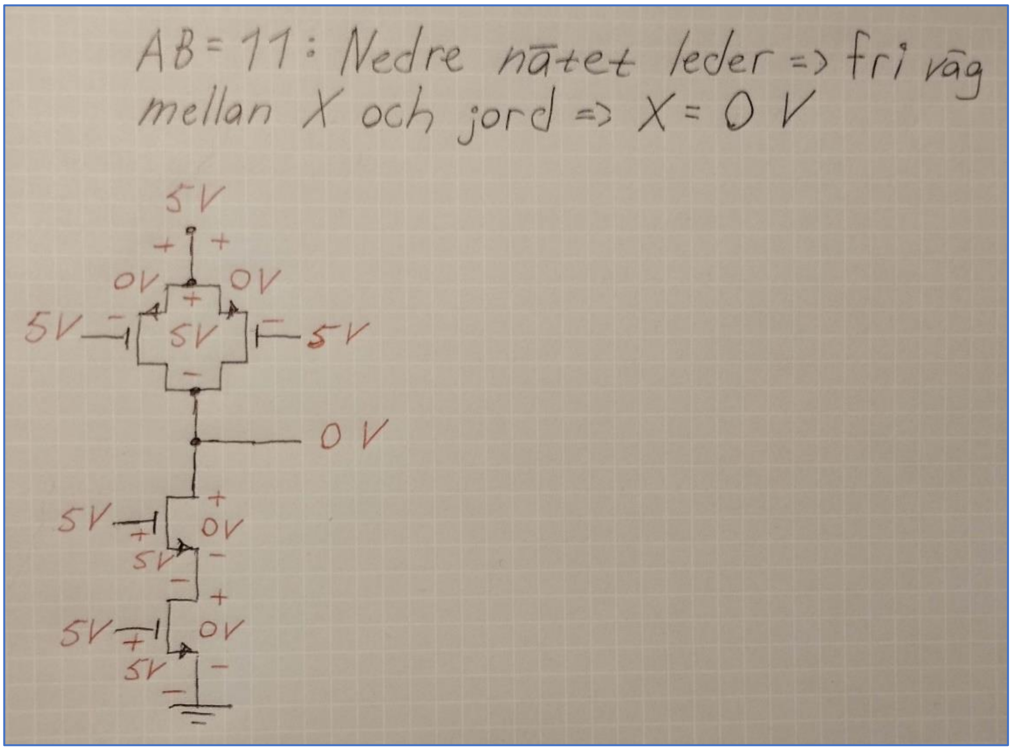
Figur 2: AB = 00, där det övre nätet leder och det nedre spärrar.



Figur 3: AB = 01, där det övre nätet leder och det nedre spärrar.

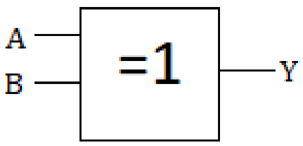


Figur 4: AB = 10, där det övre nätet leder och det nedre spärrar.



Figur 5: AB = 11, där det nedre nätet leder och det övre spärrar.

6. Vilken grind har följande symbol och sanningstabell?



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Svar

Det är en XOR-grind. XOR-grindens funktion beskrivs såsom visas nedan:

$$X = A \hat{} B = A'B + AB'$$

Därmed gäller att den ena av insignaler A och B ska vara 0 och den andre 1 för att utsignal Y ska bli 1, vilket indikeras via grindsymbolen, där =1 betyder att summan av A och B ska bli exakt 1. Notera skillnaden mot OR-grinden, där >= 1 betyder att summan av A och B ska vara större eller lika med 1 för att utsignal Y ska bli 1. Vid användning av XOR-grindar med fler än två insignaler blir utsignalen Y = 1 vid udda antal höga insignaler, annars 0.

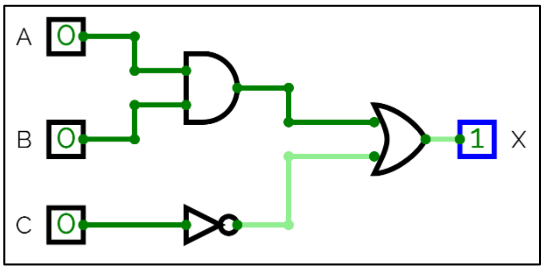
7. Realisera minimerade grindnät för följande logiska funktioner:

a) X = AB + C’
b) X = (A + B)’ * C’D
c) X = AB * AB’ + C
d) X = AB + AB’ + AC

Lösning

a) X = AB + C’ =>

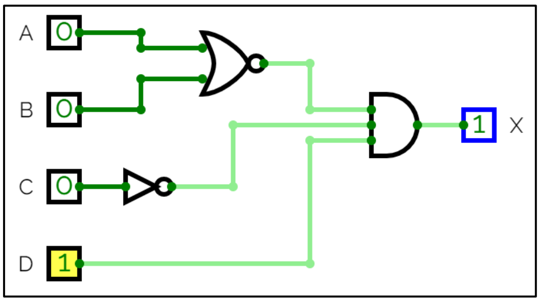
- AB utgör en AND-grind med A och B som insignaler
- AB + C’ utgör en OR-grind med AB samt C’ som insignaler
- Figur 6 till höger visar motsvarande grindnät.



Figur 6: Grindnät för X = AB+C’.

b) X = (A+B)’ * C’D

- (A + B)’ utgör en NOR-grind med A och B som insignaler.
- (A + B)’ * C’D utgör en AND-grind med (A+B)’, C’ och D som insignaler.
- Figur 7 till höger visar motsvarande grindnät.



*Figur 7: Grindnät för X = (A + B)’ * C’D.*

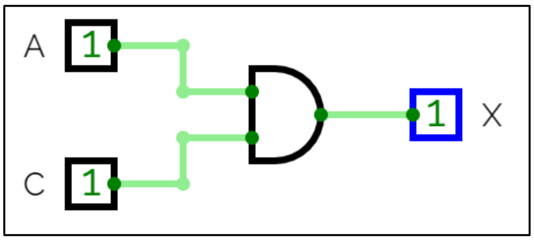
c) X = AB * AB’ + C

- Den logiska funktionen kan förenklas till
  X= A(B * B’) + C = A * 0 + C = C
- Utport X kan därmed anslutas direkt till inport C.
- Figur 8 till höger visar det minimerade nätet.

Figur 8: Grindnät för X = C.

d) X = AB + AB’ + AC

- Den logiska funktionen kan förenklas till
  X= A(B + B’) + AC = A * 1 + AC = A(1 + C) = AC
- AC utgör en AND-grind med A och C som insignaler.
- Figur 9 till höger visar det minimerade grindnätet.



Figur 9: Grindnät för X = AC.

8. Förenkla följande uttryck:

a) X = B + B’
b) X = B * B’
c) X = A + A’ + B
d) X = A * A’ + A
e) X = AB + ABC
f) X = AB’ + B

Lösning

Minnesregler:

- A + A’ = 1 + 0 = 1, då om A = 1 så är A’ = 0. Därmed blir A + A’ alltid lika med 1 + 0 = 1 oavsett värdet på A.
- A * A’ = 1 * 0 = 0, då om A = 1 så är A’ = 0. Därmed blir A * A’ alltid lika med 1 * 0 = 0 oavsett värdet på A.
- A + 1 = 1, oavsett värdet på A, då 1 + 1 = 1 samt 1 + 0 = 1
- A * 0 = 0, oavsett värdet på A, då 1 * 0 = 0 samt 0 * 0 = 0.
- A + 0 = A
- A * 1 = A
- AB’ + B = A + B. Beräkna AB’ + B för kombinationer 00 – 11 av A och B, så ser du att AB’ + B = 1 ifall A =1 eller B = 1:

$$\begin{cases} AB = 00 => AB' + B = 0 * 1 + 0 = 0 \\ AB = 01 => AB' + B = 0 * 0 + 1 = 1 \\ AB = 10 => AB' + B = 1 * 1 + 0 = 1 \\ AB = 11 => AB' + B = 1 * 0 + 1 = 1 \end{cases}$$

a) X = B + B’ = 1 + 0 = 1
b) X = B * B’ = 1 * 0 = 0
c) X = A + A’ + B = (A + A’) + B = 1 + B = 1
d) X = A * A’ + A = (A * A’) + A = 0 + A = A
e) X = AB + ABC = AB(1 + C) = AB * 1 = AB
f) X = AB’ + B = A + B

Lite information om De Morgans teorem

Enligt De Morgans teorem gäller att

$$(A + B)' = A'B'$$

Rita sanningstabellen för en OR-grind och beräkna när utsignal X = 0, dvs. X’. Vi vet redan att X = 1 ifall A = 1 eller B = 1, vilket är ekvivalent med att

$$X = A + B$$

Inversen till X är därmed lika med

$$X' = (A + B)'$$

Vi vet också att X = 0 om A = 0 samtidigt som B = 0, vilket är ekvivalent med att

$$X' = A'B'$$

Därmed gäller att

$$X' = (A + B)' = A'B'$$

Enligt De Morgans teorem gäller också att

$$(A * B)' = A' + B'$$

Rita sanningstabellen för en AND-grind och beräkna när utsignal X = 0, dvs. X’.
Vi vet redan att X = 1 ifall A = 1 och B = 1 är samtidigt, vilket är ekvivalent med att

$$X = A * B$$

Inversen till X är därmed lika med

$$X' = (A * B)'$$

Vi vet också att X = 0 om A = 0 eller B = 0, vilket är ekvivalent med att

$$X' = A' + B'$$

Därmed gäller att

$$X' = (A * B)' = A' + B'$$