Løsningsforslag øving 1
Digitalteknikk og Datamaskiner – TFE4105 – H12

Oppgave 1: Designrepresentasjon og abstraksjonsnivå

a) En strukturell representasjon beskriver produktet som et sett av komponenter og deres sammenkobling, og sier derfor ikke direkte noe om produktets funksjonalitet.

b) Når man skal beskrive et design med strukturell representasjon på registernivå så benyttes typisk komponenter som adderer, komparatorer, tellere, registre etc.

Oppgave 2: Generelle tallsystemer

a) 1) $100.10_{(5)} = 1*5^2 + 0*5^1 + 0*5^0 + 1*5^{-1} + 0*5^{-2} = \underline{25.20_{(10)}}$
   2) $364.63_{(7)} = 3*7^2 + 6*7^1 + 4*7^0 + 6*7^{-1} + 3*7^{-2} = \underline{193.92_{(10)}}$

Her må det understrekes at svaret i 2) ikke er eksakt. Det ser man lett hvis man forsøker å konvertere svaret tilbake til 7-tallssystemet. Svaret i 1) er derimot eksakt.

b) 1) $720.12_{(10)} \rightarrow$ 6-tallssystemet:

Heltallsdel	**Rest**
720:6 = 120	0 (LSD)
120:6 = 20	0
20:6 = 3	2
3:6 = 0	3 (MSD)

Fraksjon	**”Rest”**
0,12*6 = 0.72	0 (MSD)
0,72*6 = 4,32	4
0,32*6 = 1,92	1 (LSD)
0.92*6 = 5.22	5

Svar: $720.12_{(10)} = \underline{3200.042_{(6)}}$

Legg merke til at det siste tallet må avrundes til 2 siden sifferet regnet ut etter LSD er 5 og $(0.0005)_6 = (5*6^{-4})_{10} = (0,0039)_{10} \geq \frac{1}{2}(6^{-3})_{10} = (0,0023)_{10}$.

2) $600.75_{(10)} \rightarrow$ 5-tallssystemet:

Heltallsdel	**Rest**
600:5 = 120	0 (LSD)
120:5 = 24	0
24:5 = 4	4
4:5 = 0	4 (MSD)

Fraksjon	**”Rest”**
0.75*5 = 3,75	3 (MSD)
0.75*5 = 3,75	3
0.75*5 = 3,75	3 (LSD)
0.75*5 = 3,75	3

Svar: $600.75_{(10)} = \underline{4400.334_{(5)}}$

Legg merke til at det siste tallet må avrundes til 4 siden sifferet regnet ut etter LSD er 3 og $(0.0003)_5 = (3*5^{-4})_{10} = (0,0048)_{10} \geq \frac{1}{2}(5^{-3})_{10} = (0,0040)_{10}$.

c) Resultat $-573.34_{(8)}$ er framkommet etter avrunding
1) Resultatet dekker tallområdet [-573.334(8), -573.344(8)>. Dvs. fra og med -573,334 og opp til, men ikke inkludert, -573,344. -573,3437777 er altså for eksempel med.
2) Usikkerheten til resultatet er gitt ved $\pm \frac{1}{2}(8^{-2})_{10} = (0,0078)_{10}$.

Oppgave 3: Viktige tallsystemer

a) $32.11_{(8)}$ = $011\ 010.\ 001\ 001_{(2)}$

b) $11100110111.\ 01_{(2)}$ = $011\ \ 100\ \ 110\ \ 111.\ 010_{(2)}$ = $3467.2_{(8)}$

c) $\text{FADE. BABE}_{(16)}$ = $1111\ 1010\ 1101\ 1110.\ 1011\ 1010\ 1011\ 1110_{(2)}$

d) $1101\ 1101\ 1110_{(2)}$ = $\text{DDE}_{(16)}$

e) På BCD representasjon gir grupper av fire bit et enkelt siffer (0 - 9). Her får vi altså 0011 = 3 og 0110 = 6. Med andre ord 36.

Oppgave 4: Komplement-tall-aritmetikk

a) Legg merke til (i tabellen nedenfor) at man i sign-magnitude kan representere $2^N - 1$ tall med N bit siden representasjonen av 0 ikke er entydig. I 2’s komplement greier man å representere $\mathbf{2^{N-1}}$ negative tall. Man greier derimot bare å representere $\mathbf{2^{N-1} - 1}$ positive tall siden fordi man må representere null også. Dermed kan man representere et større område med negative tall enn positive tall med 2’s komplement.

Desimaltall	Binærtall (Sign-magnitude)	Binærtall (2’s komplement)
7	0111	0111
6	0110	0110
5	0101	0101
4	0100	0100
3	0011	0011
2	0010	0010
1	0001	0001
0	0000 og 1000	0000
-1	1001	1111
-2	1010	1110
-3	1011	1101
-4	1100	1100
-5	1101	1011
-6	1110	1010
-7	1111	1001
-8	-	1000

b) $A = 0110_{(2)}$
$B = 0101_{(2)}$

A-B:

$0110_{(2)}$
+ $1011_{(2)}$ (2’s komplement av B)
= $\cancel{1}0001_{(2)}$

Ignorer mente ut fra MSB (trunkerer). Svaret er positivt.

B-A:

$0101_{(2)}$
+ $1010_{(2)}$ (2’s komplement av A)
= $1111_{(2)}$

Ser av MSB at svaret er negativt. 2’s komplement av $1111_{(2)}$ er $0001_{(2)}$. (Se tabell.) Det betyr at svaret er –1.

c) Generelt kan vi aldri få overflyt (overflow) hvis tallene vi adderer har motsatt fortegn. Hvis tallene derimot har likt fortegn vil overflow vise seg hvis MSB (mest signfkante bit) i svaret (etter trunkering) er ulikt MSB i hver av de to tallene som inngår i operasjonen.

For 2’s komplement gjelder dessuten at det er overflyt dersom mente inn og mente ut av fortegnsbittet (MSB) er forskjellig.

d)

C+D:

10000 (mente)
$1000_{(2)}$ (2’s komplement av C)
+ $1101_{(2)}$ (2’s komplement av D)
= $\cancel{1}0101_{(2)}$

Vi ser at MSB i svaret er ulikt MSB i tallene som inngår i operasjonen. Vi ser også at mente i kolonnen for fortegnsbittet er 0, mens resultatet av addisjonen i denne kolonnen gir mente ut = 1. (jfr. c)). Dette svaret ble galt fordi det ikke er representerbart med 4 bit. Hvis vi tar med det 5 bit’et (fortegnsforlenger før addisjonen) får vi at svaret er $10101_{(2)}$ ($-11_{(10)}$). Svaret ville dermed blitt riktig hvis vi hadde brukt 5 bit.

e) Alternativ e1 er korrekt. Vi har foretatt fortegnsforlenging ved å repetere fortegnsbittet to ganger.

Oppgave 5: Binær multiplikasjon og divisjon

a) A = $10101_{(2)}$ (2’s komplement.)
B = $10011_{(2)}$ (2’s komplement.)

Se også eksempel 2.4 i Gajski. Dette er helt vanlig multiplikasjon, med to unntak:

- De skiftede multiplikandene summeres *etterhvert*, istedenfor å summere dem tilslutt.
- Man tar toer-komplementet av den siste skiftede multiplikanden. Dette gjøres helt konsekvent. Årsaken til dette er at fortegnsbittet har negativ posisjonsvekt.

Disse to punktene gjør det lett å realisere multiplikasjon i maskinvare.

```
Multiplikand   * Multiplikator
       10101   * 10011
--------------------------------------------------------------------------------
      000000   Partielt produkt (med fortegns-forlengelse) før vi starter.
     +110101   Multiplikand (med fortegns-forlengelse).
     1110101   Første partielle produkt (med fortegns-forlengelse).
    +110101    Skiftet (1 posisjon) multiplikand med (fortegns-forlengelse).
    11011111   Andre partielle produkt (med fortegns-forlengelse).
   +000000     Skiftet (2 posisjoner) multiplikand (med fortegns-forlengelse).
   111011111   Tredje partielle produkt (med fortegns-forlengelse).
  +000000      Skiftet (3 posisjoner) multiplikand (med fortegns-forlengelse).
  1111011111   Fjerde partielle produkt (med fortegns-forlengelse).
 +001011       2’s kompl av skiftet (4 posisjoner) multiplikand (mff).
  0010001111   = 143(10) (Ignorer mente ut).
```

mff = ”med fortegns-forlengelse”.

OBS: For å forstå oppgave 5a fullt ut, bør man først sette seg inn i:

- Fortegnsforlengelse.
- Hvorfor man tar toer-komplementet av den siste skiftede multiplikanden.

b) $C = 10100000_{(2)}$
$D = 111_{(2)}$

Dette er helt vanlig divisjon. Se også side 44 i Gajski. Merk at tallene her er positive.

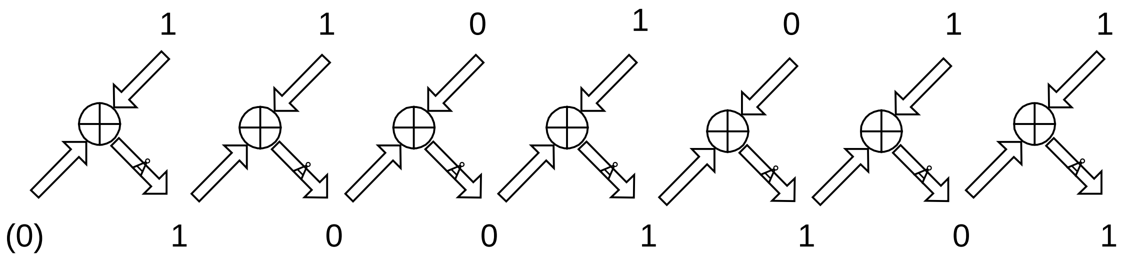
```
 10100000    : 111 =  10110
- 111                 Skiftet divisor går opp          (MSB = 1)
   110000             Ny dividend
-  000                Skiftet divisor går ikke opp     (0)
   110000             Ny dividend
-   111               Skiftet divisor går opp          (1)
    10100             Ny dividend
-    111              Skiftet divisor går opp          (1)
      110             Ny dividend
-     000             Skiftet divisor går ikke opp     (LSB = 0)
      110             Rest = 6(10)
```

Det vil si at $10100000_{(2)} : 111_{(2)} = 10110_{(2)} + 110_{(2)}/111_{(2)} = 22_{(10)} + 6/7_{(10)}$

Oppgave 6: Gray-kode

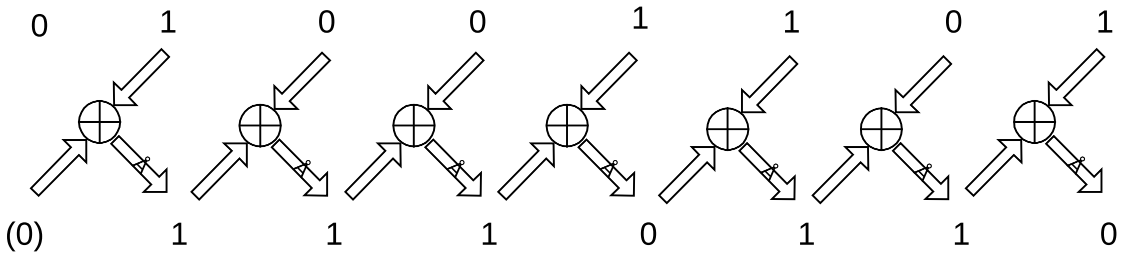
a)

1)



Svar:1001101

2)



Svar:1110110

b)

1) $0 \Rightarrow\oplus\Leftarrow 0 \Rightarrow\oplus\Leftarrow 1 \Rightarrow\oplus\Leftarrow 1 \Rightarrow\oplus\Leftarrow 1 \Rightarrow\oplus\Leftarrow 1 \Rightarrow\oplus\Leftarrow 1 \Rightarrow\oplus\Leftarrow 1$

$\Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow$

0 1 0 0 0 0 0

Svar: 0100000

2)

$$\begin{array}{ccccccccccccccc}
0 & \Rightarrow\oplus\Leftarrow & 1 & \Rightarrow\oplus\Leftarrow & 0 & \Rightarrow\oplus\Leftarrow & 0 & \Rightarrow\oplus\Leftarrow & 0 & \Rightarrow\oplus\Leftarrow & 0 & \Rightarrow\oplus\Leftarrow & 0 & \Rightarrow\oplus\Leftarrow & 0 \\
 & \Downarrow & & \Downarrow & & \Downarrow & & \Downarrow & & \Downarrow & & \Downarrow & & \Downarrow & \\
 & 1 & & 1 & & 0 & & 0 & & 0 & & 0 & & 0 &
\end{array}$$

<u>Svar: 1100000</u>

Legg her merke til at tallene i 1) og 2) følger etter hverandre i en binærsekvens, og at for å komme fra det ene til det andre må man invertere hele 7 bit. I Graykoden ser vi at de to tallene skiller seg fra hverandre i kun *en* bitposisjon.

Oppgave 7: Boolske funksjoner, algebraisk forenkling av uttrykk og logiske kretser.

a) Alternativ a2 gjengir ikke en av DeMorgans teoremer. Det som vises er en forenkling ved hjelp av aksiomene om distributivitet og identitet.

b) Både (1) og (2) er korrekte. I Boolsk algebra er + distributiv over • og motsatt.

c) Utfører De Morgan på hele F og deretter på hvert sub-ledd og får c1.

F'(A,B,C,D) = (A B' C + A D + A B C' D')' = (A B' C)' · (A D)' · (A B C' D')'
= (A' + B + C ') · (A' + D') · (A' + B' + C + D)

Trekker deretter ut A' (lovlig siden + er distributiv over ·) og bytter rekkefølgen på de to første summeleddene for å få c2.

F'(A,B,C,D) = A' + ((B + C') · D' · (B' + C + D)) = A' + D' · (B + C') · (B' + C + D)

I c3 er det bare foretatt invertering av hvert enkelt produktledd. Dette er opplagt feil.

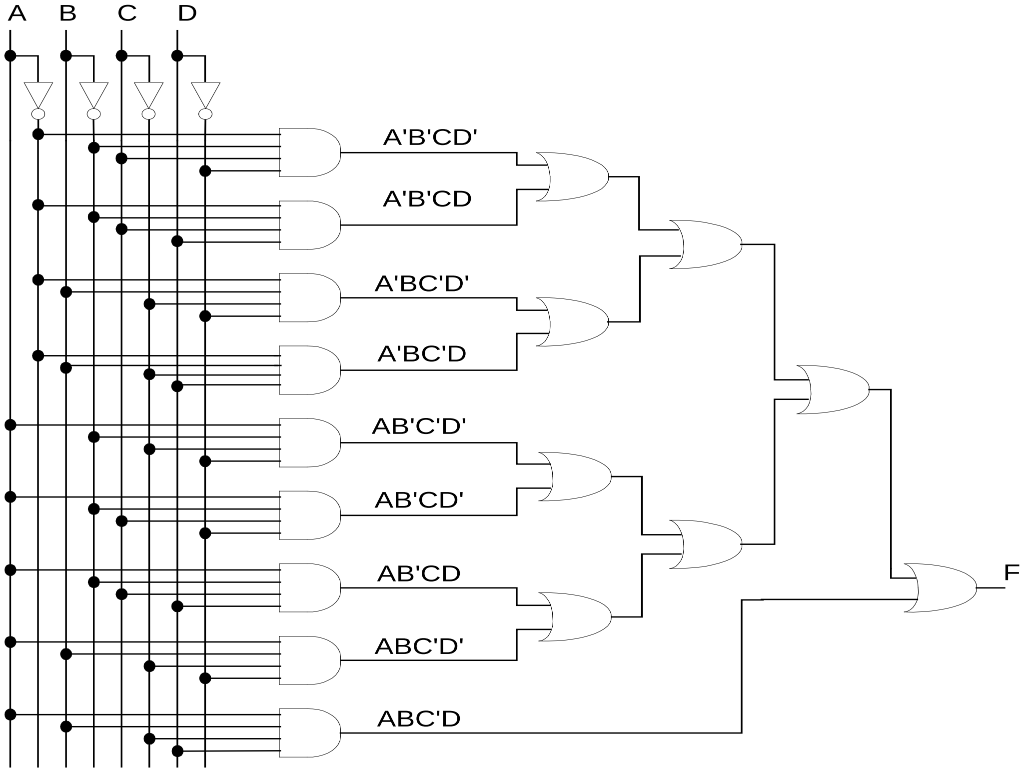
d) En minterm er et *produkt*, der alle tilgjengelige literaler inngår. (I dette tilfellet A, B, C og D) Når vi skal uttrykke funksjonen som sum av mintermer, finner vi først fra sannhetstabellen hvilke kombinasjoner av literaler som gir funksjonsverdi 1. Hver av disse kombinasjone assosieres så med en minterm. For eksempel minterm 2: (A'B'CD'). Man summerer så disse mintermene:

F = A'B'CD' + A'B'CD + A'BC'D' + A'BC'D + AB'C'D' + AB'CD' + AB'CD + ABC'D' + ABC'D
= <u>Σ(2,3,4,5,8,10,11,12,13)</u>

En maxterm er en *sum*, hvor alle tilgjengelige literaler inngår. Produkt av maxtermer finnes ved å velge kombinasjonene av literaler som gir funksjonsverdi 0. Hver av disse kombinasjonene av literaler assosierer man så med en maxterm. Man bruker her omvendt logikk. Derfor blir for eksempel maxterm 0: (A+B+C+D). Deretter tar man produktet av disse maxtermene:

F = (A+B+C+D)(A+B+C+D')(A+B'+C'+D)(A+B'+C'+D')(A'+B+C+D')(A'+B'+C'+D)(A'+B'+C'+D')
= <u>Π(0,1,6,7,9,14,15)</u>

e)



Figur 1.1: AND-OR implementasjon

f)

F = A’B’(CD’+CD) + A’B(C’D’+C’D) + AB’(C’D’+CD+CD’) + AB(C’D+C’D’)

= A’B’C + A’BC’ + AB’(C’D’ + C) + ABC’

= A’(B’C + BC’) + AB’C’D’ + A(B’C + BC’)

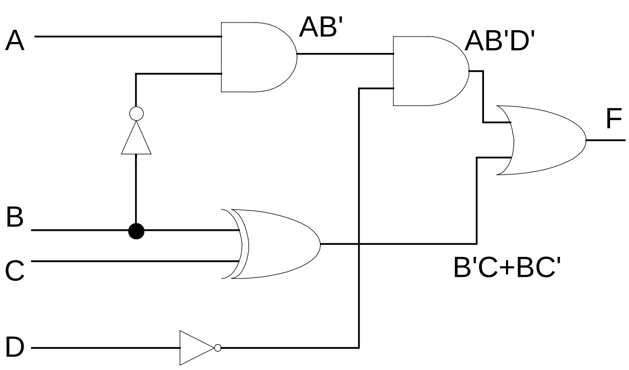
= BC’ + B’C + AB’C’D’ = BC’ + B’(C + AC’D’) **(*)**

= BC’ + B’(C + AD’)(C + C’) = <u>BC’ + B’C + AB’D’</u>

(*) Her benyttes det at uttrykk i boolsk algebra er distributive med hensyn på + operatoren. X + YZ = (X + Y)(X + Z). I vårt eksempel er X=C, Y=AD’ og Z=C’.

g) Før vi tegner denne kretsen merker vi oss at svaret fra c) kan skrives på følgende måte:

BC’ + B’C + AB’D’ = <u>(B ⊕ C) + AB’D’</u>



Figur 1.2: Implementasjon av forenklet funksjon.