Konvertering mellom tallsystemer

Hans Petter Taugbøl Kragset
hpkragse@ifi.uio.no

November 2014

1 Introduksjon

Dette dokumentet er ment som en referanse for konvertering mellom det desimale, det binære, og det heksadesimale tallsystemet, og er tilsiktet studenter i INF1000 ved Institutt for Informatikk, Universitetet i Oslo.

1.0.1 Notasjon

Det er vanlig å skrive grunntallet til et tall med nedsenket skrift etter et tall, så det vi er vant til som 100 er mest presist hvis vi sier 100_{10}. Det er viktig å være presis med dette når vi jobber frem og tilbake mellom tallsystemer.

2 Konvertering til desimal

2.1 Generelt

Det som kommer er kanskje banalt og virker dumt, men det er viktig å ha god forståelse av prinsippene bak et tallsystem, for reglene er direkte overførbare mellom systemene. På tvers av tallsystemene er det mulig å anvende de samme teknikkene, og vi har en grunnformel som vi skal forholde oss til:

$$verdi = siffer * grunntall^{posisjon} \tag{1}$$

Denne formelen gir oss verdien av et gitt siffer i et gitt tall. *Grunntallet* i et tallsystem er alltid det samme som antall tilgjengelige siffer i det samme tallsystemet. *Posisjon* er her sifferets posisjon i tallet, hvor posisjonen lengst til høyre er 0, og øker mot venstre.

2.2 Desimale tall

Dette er tatt med for å vise hvordan formelen fungerer på tall vi er veldig kjent med fra før. Vi bruker formelen (1) på tallet 368_{10} og får følgende utregning av verdier:

Posisjon	Siffer	Utregning
0	8	$8 * 10^0 = 8 * 1 = 8_{10}$
1	6	$6 * 10^1 = 6 * 10 = 60_{10}$
2	3	$3 * 10^2 = 3 * 100 = 300_{10}$

Summen av alle verdiene er "selvfølgelig" 368, som var tallet vi startet med. Her har vi regnet fra desimal til desimal!

2.3 Binære tall

Det binære tallsystemet har grunntallet 2, og består kun av sifferene 0 og 1. Vi bruker igjen formelen (1). For tallet 101010_2 får vi følgende utregning av verdier:

Posisjon	Siffer	Utregning
0	0	$0 * 2^0 = 0 * 1 = 0_{10}$
1	1	$1 * 2^1 = 1 * 2 = 2_{10}$
2	0	$0 * 2^2 = 0 * 4 = 0_{10}$
3	1	$1 * 2^3 = 1 * 8 = 8_{10}$
4	0	$0 * 2^4 = 0 * 16 = 0_{10}$
5	1	$1 * 2^5 = 1 * 32 = 32_{10}$

Summen av verdiene er nå

$$32 + 8 + 2 = 42_{10} = 101010_2$$

Som vi ser tar binære tall mye større plass enn desimale tall.

2.4 Heksadesimale tall

Det heksadesimale tallsystem har grunntallet 16, og har verdiene 0 til og med 9, og a til og med f. Nå begynner det å se annerledes ut, vi blander bokstaver og tall. Her er en liten oversikt over hvordan vi oversetter bokstavene til tall, store eller små bokstaver spiller ingen rolle:

Heksadesimal	Desimal
A	10
B	11
C	12
D	13
E	14
F	15

Hvis vi bare holder tunga rett i munnen og bruker den samme formelen, ser vi at utregning til desimal ikke er noen sak, la oss prøve tallet $BEEF_{16}$:

Posisjon	Siffer	Utregning
0	F	$15 * 16^0 = 15 * 0 = 15_{10}$
1	E	$14 * 16^1 = 14 * 16 = 224_{10}$
2	E	$14 * 16^2 = 14 * 256 = 3584_{10}$
3	B	$11 * 16^3 = 11 * 4096 = 45056_{10}$

Summen av verdiene er altså

$$15 + 224 + 358 + 45056 = 48879_{10} = BEEF_{16}$$

Det heksadesimale tallet tar her mindre plass enn det desimale.

3 Konvertering til binær

For å regne om til binære tall bruker vi andre metoder enn når vi regner til desimale tall. Metoden vi bruker for å komme fra heksadesimal til binær skiller seg også fra måten vi bruker på desimale tall. Vi kommer nærmere inn på dette.

3.1 Desimale tall

Algoritmen for å konvertere fra desimale tall til binære tall er ganske enkel i teorien:

- Del tallet på to (heltalssdivisjon). Får vi rest?
  - Ja: skriv ned resten (alltid 1 for binære tall)
  - Nei: skriv 0
- Start på toppen igjen, helt til svaret på heltallsdivisjonen er 0

La oss prøve med tallet 79:

Verdi	Utregning	Rest
79	$79 \div 2 = 39$, $79\%2 = 1$	1
39	$39 \div 2 = 19$, $39\%2 = 1$	1
19	$19 \div 2 = 9$, $19\%2 = 1$	1
9	$9 \div 2 = 4$, $9\%2 = 1$	1
4	$4 \div 2 = 2$, $4\%2 = 0$	0
2	$2 \div 2 = 1$, $2\%2 = 0$	0
1	$1 \div 2 = 0$, $1\%2 = 1$	1
$\underline{0}$		

Det ferdige binære tallet leser vi så *nedenfra og opp* i kolonnen Rest. I dette tilfellet er

$$79_{10} = 1001111_2$$

Vi gjør en modulo-beregning med % som vi kjenner fra Java for å finne resten. I praksis ser vi egentlig bare på om tallet er et oddetall eller partall, oddetall får alltid rest.

3.2 Heksadesimale tall

Det er en veldig tett sammenheng mellom binære og heksadesimale tall, og det er hovedgrunnen til at vi lører oss det heksadesimale tallsystemet. Den enkleste måten å gå fra et heksadesimalt tall til et binært tall er ved å rett og slett se på hvert enkelt siffer i det heksadesimale tallet og finne det tilsvarende binære tallet, for så å slå de binære tallene vi har funnet sammen i akkurat den samme rekkefølgen som de sto i det heksadesimale tallet. Denne metoden er utrolig rask, og det er derfor en fordel å lære seg det binære tallet for tallene $0 - f_{16}$ som vi trenger i denne metoden. Vi tar et eksempel, vi gjør $D3A3_{16}$ om til binær:

Heksadesimal siffer	Binært tall	Desimalt tall
D	1001	13
3	0011	3
A	0110	10
3	0011	3

Videre konkatenerer vi de fire binære siffergruppene, og her er det veldig viktig at vi får med alle tallene, også nullene. Vær oppmerksom på at + her ikke betyr matematisk plussing, men konkatenering. Pass på at rekkefølgen til siffergruppene er det samme som rekkefølgen de heksadesimale sifrene sto i før vi begynte konverteringen.

$$1001 + 0011 + 0110 + 0011 = 10010011\ 01100011$$

Legg her spesielt merke til at vi går ovenfra og ned i motsetning til da vi regnet fra desimale tall, og at vi gjerne deler det binære tallet opp i grupper på 8 siffer for å forbedre leseligheten.

4 Konvertering til heksadesimal

4.1 Desimale tall

Utregningen fra et desimalt tall til et heksadesimalt tall følger de samme linjene som konvertering fra et desimalt tall til et binært tall, bortsett fra at vi deler på 16 i steden for 2. Algoritmen blir da som følger:

- Del tallet på 16 (heltalssdivisjon). Får vi rest?
  - Ja: skriv ned resten
  - Nei: skriv 0
- Start på toppen igjen, helt til svaret på heltallsdivisjonen er 0

Hvis vi prøver dette for tallet 435_{10} får vi følgende utregning:

Verdi	Utregning	Rest	Heksadesimal rest
435	$435 \div 16 = 27,\ 79\%16 = 3$	3	3
27	$27 \div 16 = 1,\ 27\%16 = 11$	11	b
1	$1 \div 16 = 0,\ 1\%16 = 1$	1	1
$\underline{0}$			

Det ferdige heksadesimale tallet leser vi så *nedenfra og opp* i kolonnen Heksadesimal rest. I dette tilfellet er

$$435_{10} = 1b3_{16}$$

Det å finne resten ved heltalsdivisjon med 16 er litt komplisert å gjøre i hodet, men med kalkulator er det en smal sak. La oss ta for oss vårt eksempel:

- Kalkulatoren gir at $435 \div 16 = 27,1875$
- Vi kan se bort i fra det etter komma, og regne videre, nå *ganger* vi 16 med 27, altså $16*27 = 432$
- Deretter tar vi differansen av det oprinnelige tallet, 435, og produktet i det siste regnestykket, 432, altså $435 - 432 = 3$

Oppskriften blir det samme for alle rundene vi må gjennom. Vi kan trøste oss med at siden vi deler med et relativt stort tall, 16, blir regnestykkene raskt enklere, og vi får heller ikke 10-12 rader som vi kan få med samme teknikken anvendt på binære tall.

4.2 Binære tall

I korthet går dette ut på å gjøre metoden for konvertering fra heksadesimale tall til binære tall, bare baklengs. I steden for å gjøre hver enkelt siffer i det heksadesimale tallet om til et firesifret binært tall, gjør vi fire og fire siffer i det binære tallet om til et heksadesimalt siffer. Vi begynner bakerst i det binære tallet, altså til høyre, og velger de fire bakerste sifrene. Deretter pluker vi fire nye siffer, helt til det ikke er flere igjen. Dersom det er færre enn 4 siffer igjen må vi bare fylle ut de manglende sifrene med 0. Ikke misforstå, vi ønsker ikke å snu rundt på sifrene, men å beholde de i samme rekkefølge. Vi skal se på et eksempel, der vi ønsker å gjøre om 101011_2 til heksadesimal.

- Vi begynner med å dele tallet opp i grupper på fire, vi starter bakerst.
  - Først får vi 1011, deretter 0010
- Vi fyller ut de manglende plassene med to ekstra 0-er
- Rekkefølgen av sifrene er akkurat som før vi begynte.

Vi har nå en tabell som ligner den vi hadde da vi konverterte andre veien:

Binært tall	Heksadesimalt tall	Desimalt tall
1011	b	13
0010	2	3

Nå kan vi bare konkatenere de heksadesimale sifrene vi har fått, og vi må igjen passe på at rekkefølgen er det samme som rekkefølgen til de assosierte siffergruppene i det binære tallet.

$$2 + b = 2b$$

Hvis du synes det er mer logisk å snu tabellen opp ned for å lese ovenfra og nedover, vær så god. Jeg gjør det på denne måten fordi vi alltid skal begynne telling av fire og fire siffer bakfra.