Konvertering mellom tallsystemer
En ikke-matematisk tilnærming

Nicolai August Hagen
nicolhag@ifi.uio.no

6. august 2015

1 Oversikt

Det desimale tallsystemet, titallssystemet, består av verdiene 0-9. Det hexadesimale tallsystemet (16-tallssystemet) består av 16 verdier, henholdvis 0-9 og A-F. Det binære tallsystemet (totallssystemet) bruker kun verdiene 0-1. Vi kan dermed bruke følgende tabell:

Binær	Hexadesimal	Desimal
0	0	0
1	1	1
	2	2
	3	3
	4	4
	5	5
	6	6
	8	7
	9	8
	A	9
	B	*10*
	C	*11*
	D	*12*
	E	*13*
	F	*14*

I tabellen ser vi blant annet at tallet 1 er representert som 1 i alle tre tallsystemer. Vi kan også se at desimaltallet 15 er representert som verdien F i det hexadesimale tallsystemet.

2 Binærtall

I datamaskinen, i minnet eller på harddisk, blir all data lagret i bolker på flere bit (verdien **0** eller **1**). Disse verdiene kan tolkes som brytere for av eller på. I eksemplet under skal vi se nærmere på hvordan datatypen *char* blir lagret i datamaskinen. Vi kan tenke på det som at *char* alltid er representert i bolker av 8 bit. En slik bolk på 8 bit kalles også for en *byte*.

Alle *char*'s har sin egen tallverdi, illustrert ved ASCII-tabellen under:

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

For eksempel ser vi at *char*'en **A** (Chr) har verdien 65 i titallsystemet (Dec). Som vi ser har **A** også den hexadesimale (Hx) verdien 41, som vi kommer tilbake til på neste side. Som nevnt lagres en *char* (som regel) som en rekke av 8 bit. I vårt tilfelle kan **A** representeres som den binære tallrekken:

$$01000001$$

For at dette skal gi mening for oss trenger vi å tolke tallrekken, fra høyre mot venstre. Dette gjøres ved å sette opp hvilken tallverdi *hver bit* representerer:

Tallverdirepresentert	128	64	32	16	8	4	2	1
Binær tallrekke	0	1	0	0	0	0	0	1

Ut i fra oversikten ser vi nå at den første biten (helt til høyre) representerer tallet 1. Den andre biten er ikke "slått på" og vi trenger dermed ikke å ha den med i regnestykket vårt. Det samme vil skje med resten hvor biten er satt til 0. Neste bit som er "slått på" er den som representerer tallet 64. Til slutt får vi dermed regnestykket:

$$64 + 1 = 65$$

3 Hexadesimale tall

Når vi ønsker å finne den hexadesimale verdien er det nødvendig å se på 4 og 4 bit av gangen. Derfor deler vi nå opp den binære tallrekken vår, *01000001*, i to like store bolker - 0100 og 0001. Hvis man ikke har 4 bit fyller man inn resten med 0 (helt til venstre).

BOLK 1

Tallverdirepresentert	8	4	2	1
Binær tallrekke	0	1	0	0

BOLK 2

Tallverdirepresentert	8	4	2	1
Binær tallrekke	0	0	0	1

Legg merke til at vi nå ser på hver enkelt bolk som en ny binær tallrekke. Vi får dermed utregningen:

$$\textit{BOLK 1} \rightarrow \ 0 + 4 + 0 + 0 = 4$$

$$\textit{BOLK 2} \rightarrow \ 0 + 0 + 0 + 1 = 1$$

Hver enkelt av disse 4-bits-tallrekkene kan til sammen representere totalt 16 ulike verdier, som tilsvarer alle mulige verdier man kan få i det hexadesimale tallsystemet. Hvis vi slår opp i tabellen fra starten av side 1, ser vi nå at vi får:

$$\textit{BOLK 1} \rightarrow \ \text{Hexadesimalverdi } 4$$

$$\textit{BOLK 2} \rightarrow \ \text{Hexadesimalverdi } 1$$

Slår vi dette sammen, ser vi at **A** er representert i hexadesimal form som *41*.