Digitala tal och Boolesk algebra

Innehåll

- Talsystem och koder
- Aritmetik för binära tal
- Grundläggande logiska operationer
- Logiska grindar
- Definitioner i Boolesk algebra
- Räknelagar

Binära talsystemet

- Binärt
  - Positionssystem
  - Två symboler används, B = { 0, 1 }
  - Binära tal gör det lätt att bygga elektronik baserade på elektroniska omkopplare
  - En algebra utvecklad av Boole gör det lätt att hantera logiska uttryck baserade på binära tal

Positionsbaserade talsystem

- Ett generellt positionsbaserat talsystem med basen *b*

Låt

$S=\{s_0, s_1, \ldots, s_{b-2}, s_{b-1}\}$ = mängd symboler

b=antal symboler ≡ basen

N = positivt heltal, $N \geq 0$

q = antal positioner som krävs för att representera N i basen b

Representera positiva heltal

- För positiva heltal:

$$N = \sum_{i=0}^{q-1} s_i b^i$$

- *ex.,* Decimalt tal

$$(234)_{10} = s_2 b^2 + s_1 b^1 + s_0 b^0$$

låt $b = 10, \quad S = \{0,1,\ldots,8,9\}$

$$(234)_{10} = 2 \bullet 10^2 + 3 \bullet 10^1 + 4 \bullet 10^0$$

$$(234)_{10} = 2 \bullet 100 + 3 \bullet 10 + 4 \bullet 1$$

$$(234)_{10} = 200 + 30 + 4$$

Representera positiva heltal

- *exempel*, Binärt tal (basen 2)

$(1011)_2 = s_3 b^3 + s_2 b^2 + s_1 b^1 + s_0 b^0$

låt $b = 2$, $S = \{0,1\}$

$(1011)_2 = 1 \bullet 2^3 + 0 \bullet 2^2 + 1 \bullet 2^1 + 1 \bullet 2^0$

$(1011)_2 = 1 \bullet (1000)_2 + 0 \bullet (100)_2 + 1 \bullet (10)_2 + 1 \bullet (1)_2$

$(1011)_2 = (1)_{10} \bullet (8)_{10} + (0)_{10} \bullet (4)_{10} + (1)_{10} \bullet (2)_{10} + (1)_{10} \bullet (1)_{10}$

$(1011)_2 = (8+2+1)_{10} = (11)_{10}$

Representera delar av heltal (*eng. Fraction*)

Låt

$S = \{s_{-1}, \ldots, s_{r-2}, s_{r-1}\}$ = mängd symboler

b=antal symboler i S ≡ basen

$1 > F \geq 0$

p = Antal positioner som krävs för att representera F i basen b, eller antal positioner som tillåts

då blir

$$F = \sum_{i=-p}^{-1} s_i b^i$$

Exempel på decimal tal

- Låt p = 3 (*tre positioner till höger om decimalpunkten*)

$$b = 10$$

$$F = s_{-1}b^{-1} + s_{-2}b^{-2} + s_{-3}b^{-3}$$

$$(0.625)_{10} = 6 \bullet 10^{-1} + 2 \bullet 10^{-2} + 5 \bullet 10^{-3}$$

$$(0.625)_{10} = 6 \bullet 0.1 + 2 \bullet 0.01 + 5 \bullet 0.001$$

$$(0.625)_{10} = 0.600 + 0.0200 + 0.005$$

Exempel på Binära ”decimaltal”?

- Låt p = 3

$$b = 2$$

$$(0.101)_2 = 1 \bullet 2^{-1} + 0 \bullet 2^{-2} + 1 \bullet 2^{-3}$$

$$(0.101)_2 = 1 \bullet \frac{1}{2} + 0 \bullet \frac{1}{4} + 1 \bullet \frac{1}{8}$$

$$(0.101)_2 = (0.100)_2 + (0.000)_2 + (0.001)_2$$

Bas-konvertering

- Ett tal med basen *b* skrivs om som:

$$N_0 = s_p \bullet b^p + s_{p-1} \bullet b^{p-1} + \ldots + s_1 \bullet b^1 + s_0 \bullet b^0$$
$$= (s_p \bullet b^{p-1} + s_{p-1} \bullet b^{p-2} + \ldots + s_1) \bullet b + s_0$$
$$= N_1 \bullet b + s_0$$

- En division av N_0 med *b* ger:

$$N_1 + \frac{s_0}{b}$$

Resten = s_0 = minst signifikanta siffran

$$N_2 + \frac{s_1}{b}$$

Resten = s_1 = näst minst signifikanta siffran

Procedur för bas-konvertering

Exempel: Omvandla 57_{10} till binärt tal

	kvot	*rest*
57 / 2 =	28	1
28 / 2 =	14	0
14 / 2 =	7	0
7 / 2 =	3	1
3 / 2 =	1	1
1 / 2 =	0	1

Minst signifikanta biten *(eng. Least Significant Bit)*

Mest signifikanta biten *(eng. Most Significant Bit)*

Svar: $57_{10} = 111001_2$

Att tänka på vid omvandling

- Syftet med binär representation är att erhålla tal i ett format som passar digital logik
- Ju större noggrannhet ett binärt tal har desto fler bitar krävs → mer digitala kretsar
- Alla tal i en bas kan INTE representeras exakt i en annan bas (avrundningsfel)

Binära, Oktala och Hexadecimala tal

- Det är lätt att konvertera binära tal till andra, mer lättarbetade format genom att gruppera bitar tillsammans och sedan konvertera till lämplig bas
  - Oktala tal S={ 0, 1, ... , 7 }, basen = 8
    - 3-bits grupper
  - Hexadecimal S={ 0, ... , 9, A, B, C, D, E, F }, basen = 16
    - 4-bits grupper

Exempel: Binär till Oktal omvandling

$N_2 = 1010110100.1$

gruppera i 3 - bitars grupper från "decimal"-punkten

$N_2 = 1\,\underline{010}\,\underline{110}\,\underline{100}.1$

fyll ut med två nollor för att få ett oktalt tal (en full grupp)

$N_2 = \underline{001}\,010\,110\,100.\underline{100}$

Konvertera varje 3 - bitars grupp

$N_8 = 1 \quad 2 \quad 6 \quad 4 \;.\; 4$

$N_8 = (1264.4)_8$

Exempel: Binär till hexadecimal

N_2=1010110100.1

gruppera i 4 - bitars grupper från "decimal"-punkten

N_2=10$\underline{1011}\,\underline{0100}$.1

fyll ut med nollor för att få en komplett grupp

N_2=$\underline{0010}$1011 0100.$\underline{1000}$

Konvertera varje 4 - bitars grupp

N_{16}= 2 B 4 . 8

N_{16}= $(2B4.8)_{16}$= $(2B4.8)_H$

Viktade koder

- Godtycklig vikt kan tilldelas varje position
  - Binary Coded Decimal (BCD)
    - 8, 4, 2, 1
    - exempel, 1001 = 8 + _ + _ + 1 = 9
    - Alla kodord används inte
      - Enbart 0_{10}-9_{10} används
      - Ej 10_{10}-15_{10}

Icke-viktade koder

- Cykliska
  - På varandra följande kodord skiljer sig åt med endast en bit och det gäller också då de ”slår runt”
  - Gray Code är den vanligaste

Gray kod

- Fördelar
  - Enkelt att konstruera för vilket antal bitar som helst
  - Cyklisk
  - Unik

3-bitars Gray kod

Ordning	*Kodord*
0	0 0 0
1	0 0 1
2	0 1 1
3	0 1 0
4	1 1 0
5	1 1 1
6	1 0 1
7	1 0 0

Alfanumeriska koder

- Alfanumeriska koder representerar både
  - Decimala siffersymboler
    - 0 - 9
  - Alfabetets tecken
    - A – Z, a – z
  - Övriga skrivbara tecken
    - T.ex: %, &, ?, *, @
  - Styrsymboler
    - Blanksteg, ny rad, etc.
  - Standardkoder: **ASCII**, EBCDIC, etc.

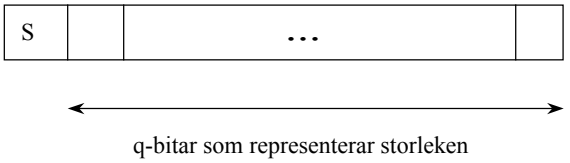
ASCII

- ASCII kodning
  - *American Standard Code for Information Interchange*
  - ASCII-tabell
    - Ger hexadecimal kod
    - Rad: minst signifikanta positionen
    - Kolumn: mest signifikanta positionen
    - Exempel: ASCII('C') = 43_{16}

	000	001	002	003	004	005	006	007
0	NUL	DLE	SP	0	@	P	`	p
1	STX	DC1	!	1	A	Q	a	q
2	SOT	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DCK	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Negativa tal

- teckenbit
  - Biten längst till vänster representerar talets tecken
    - 1 negativt
    - 0 positivt
  - Bitarna till höger om teckenbiten är storleken på talet



Exempel: tal med teckenbit

- 2 koder för noll
- Skiftning
  - Ger ej mult/div med 2,
  - Tar ej hänsyn till tecknet

Decimal	s	$(\text{storlek})_2$
+3	0	11
+2	0	10
+1	0	01
+0	0	00
-0	1	00
-1	1	01
-2	1	10
-3	1	11

Två-komplement

- Två-komplement representation
  - För ett n-bitars tal
    - är värdet för MSB -2^{p-1} (istället för $+2^{p-1}$)
    - övriga bitars värde är samma som för positiva tal
  - Procedur för att utföra två-komplement
    - invertera samtliga bitar i talet
    - addera 1 till talet
  - Exempel:

Bilda två-komplement till 8-bitars talet 00010001_2 ($=17_{10}$)

00010001
11101110
+ 1
11101111 = -17_{10}

Två-komplement – 3-bitars tal

- En kod för noll

Decimal	2-komp
+3	011
+2	010
+1	001
+0	000
-0	000
-1	111
-2	110
-3	101
-4	100

Addition och subtraktion

- Exempel:

3 + 2 = ?

0011
0010
0101 = 5_{10}

7 - 1 = ?

0111
1111
1 0110 = 6_{10}

-1 - 3 = ?

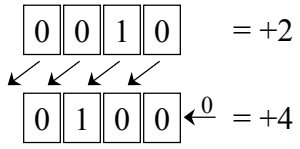
1111
1101
1 1100 = -4_{10}

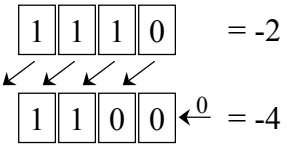
2 - 3 = ?

0010
1101
1111 = -1_{10}

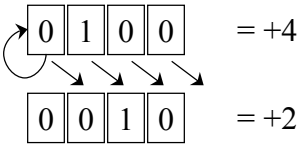
Multiplikation/Division med 2

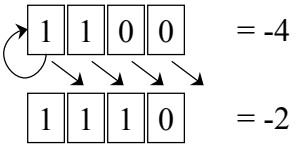
- Skifta det binära talet ett steg vänster





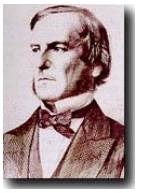
- Skifta det binära talet ett steg höger





Boolesk Algebra

- Historik



George Boole (1815-1864), en engelsk matematiker visade att logik kan uttryckas som algebraiska ekvationer. Han gav upphov till vad vi kallar *Boolesk algebra.*
(*1854: An Investigation of the Laws of Thought, on Which Are Founded the Mathematical Theories of Logic and Probabilities*)

Används idag inom matematik, informationsteori, switching algebra, grafteori, datorvetenskap och artificiell intelligens.

Edward Huntington: (1874-1952), en amerikansk matematiker som gav Boolesk algebra sina axiom.

Claude Shannon (1916-2001), en amerikansk matematiker som beskrev informationens minsta beståndsdel som 0 eller 1. Han myntade begreppet ’bit’. Han lade grunden till informationsteorin som har haft avgörande betydelse för utvecklingen av kommunikationssystem.

Boolesk Algebra – Definitioner

Konstanter	
0	(Falsk)
1	(Sann)

Operationer	
+	(ELLER)
·	(OCH)
’	(ICKE)

Axiom

$0 + 0 = 0$
$1 \cdot 1 = 1$

$1 + 1 = 1$
$0 \cdot 0 = 0$

$0 + 1 = 1 + 0 = 1$
$1 \cdot 0 = 0 \cdot 1 = 0$

$0' = 1$
$1' = 0$

Räknelagar för en variabel

$x + x = x$
$x \cdot x = x$

$x + x' = 1$
$x \cdot x' = 0$

$x + 1 = 1$
$x \cdot 0 = 0$

$x + 0 = x$
$x \cdot 1 = x$

$(x')' = x$

Dessa räknelagar kan enkelt visas utifrån axiomen.

Visa att $x + x = x$

Låt $x = 1$: $1 + 1 = 1$

Låt $x = 0$: $0 + 0 = 0$

Räknelagar för flera variabler

- Associativa lagar
  $x + (y + z) = (x + y) + z$
  $x \cdot (y \cdot z) = (x \cdot y) \cdot z$
- Kommutativa lagar
  $x + y = y + x$
  $x \cdot y = y \cdot x$
- Distributiva lagar
  $x \cdot (y + z) = x \cdot y + x \cdot z$
  $x + y \cdot z = (x + y) \cdot (x + z)$

Räknelagar för flera variabler

- absorptionslagar
  - $x + x \cdot y = x$
  - $x \cdot (x + y) = x$

$x + x \cdot y = x \cdot (1 + y) = x \cdot 1 = x$

$x \cdot (x + y) = x \cdot x + x \cdot y = x + x \cdot y$
$= x \cdot (1 + y) = x \cdot 1 = x$

- Concensuslagen
  - $x \cdot y + x' \cdot z = x \cdot y + x' \cdot z + y \cdot z$

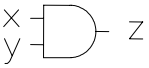
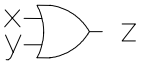
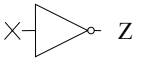
Augustus de Morgan

- De Morgans lag
  - $(x + y)' = x' \cdot y'$
  - $(x \cdot y)' = x' + y'$

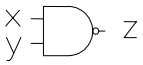
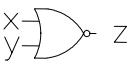
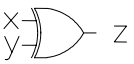
$\overline{x + y} = \overline{x} \cdot \overline{y}$

$\overline{x \cdot y} = \overline{x} + \overline{y}$

Grundläggande logiska grindar

Namn/operator	Symbol	Funktion	Logisk operation
OCH, *eng. AND* ·		X Y Z 0 0 0 0 1 0 1 0 0 1 1 1	Z = X • Y
ELLER, *eng. OR* +		X Y Z 0 0 0 0 1 1 1 0 1 1 1 1	Z = X + Y
ICKE, *eng. NOT* '		X Z 0 1 1 0	Z = X'

Grundläggande logiska grindar

Namn/operator	Symbol	Funktion	Logisk operation
NAND		X Y Z 0 0 1 0 1 1 1 0 1 1 1 0	$Z = \overline{X \cdot Y}$ $Z = (X \cdot Y)'$
NOR		X Y Z 0 0 1 0 1 0 1 0 0 1 1 0	$Z = \overline{X + Y}$ $Z = (X + Y)'$
XOR $\oplus$		X Y Z 0 0 0 0 1 1 1 0 1 1 1 0	$Z = X \oplus Y$

SLUT på Föreläsning 1

- Innehåll
  - Talsystem och koder
  - Aritmetik för binära tal
  - Grundläggande logiska operationer
  - Logiska grindar
  - Definitioner i Boolesk algebra
  - Räknelagar