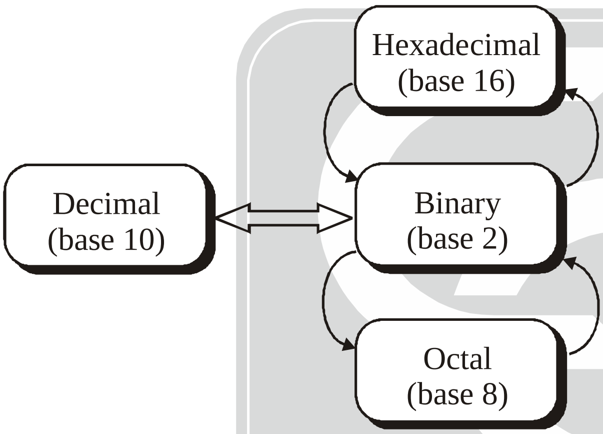
CHAPTER

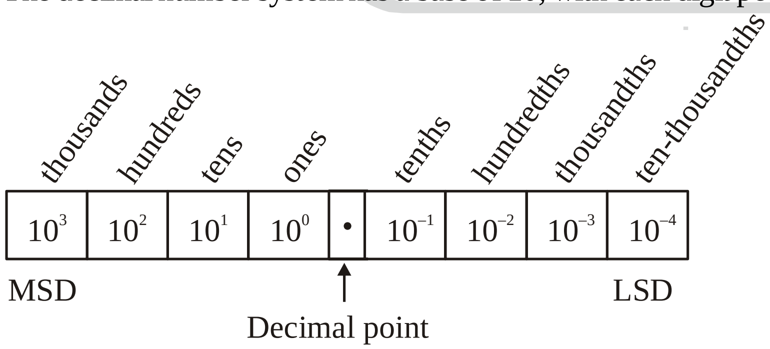
2

Data Representation



Decimal Numbers:

- Deci = ten
- The decimal number system has ten digits:
  0, 1, 2, 3, 4, 5, 6, 7 8, 9.
- The decimal number system has a base of 10, with each digit position weighted by a power of 10:



MSD = most significant digit

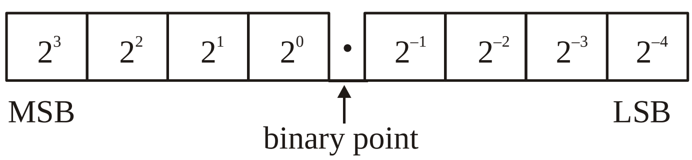
LSD = least significant digit

Binary Numbers:

- Bi = two
- The binary number system has two digits: 0 and 1
- The binary number system has a base of 2 with each digit position weighted by a power of 2.

MSB = most significant bit

LSB = least significant bit



- Subscript $_2$ indicates the number is in binary notation. (base 2)
- Example:

10011_2 is:

2^4	2^3	2^2	2^1	2^0
1	0	0	1	1

$$1\times 2^4+0\times 2^3+0\times 2^2+1\times 2^1+1\times 2^2=9$$

Integers and fractions in binary:

- Binary numbers can represent fractional values as well as integes:

– e.g.: 10011.011_2

2^4	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}
1	0	0	1	1	•	0	1	1

- 10011.011_2 is an 8-bit number.

- What if we want to represent $(19.376)_{10}$ with an 8-bit binary number?

Conversion: decimal to binary (method 1)

- Express the decimal number as the sum of powers of 2.
- 1s and 0s written in the corresponding bit positions.

Example 1:

$$50_{10}=32+18$$
$$=32+16+2$$
$$=1\times 2^5+1\times 2^4+1\times 2^1$$
$$50_{10}=110010_2$$

Example 2:

$$338.5_{10}=256+82.5$$
$$=256+64+18.5$$
$$=256+64+16+2.5$$
$$=256+64+16+2+0.5$$
$$=1\times 2^8+1\times 2^6+1\times 2^4+1\times 2^1+1\times 2^{-1}$$
$$338.5_{10}=(101010010.1)_2$$

Conversion: decimal to binary (method 2)

- Repeated division

		Quotient	Remainder
50/2	=	25	0
25 / 2	=	12	1
12 / 2	=	6	0
6 / 2	=	3	0
3 / 2	=	1	1
1 / 2	=	0	1

$$50_{10}=110010_2$$

Conversion: binary to decimal

Determine the power of 2 for the position of each 1, and then sum.

Examples: $(10101)_2=2^4+2^2+2^0=16+4+1=(21)_{10}$

Binary Addtion:

• Firstly, recall decimal addition:

	1	1		
A	1	2	5	4
+B		7	8	2
Sum	2	0	3	6

• Binary addition follows the same pattern, but

$0+0 = 0$ carry 0

$0+1=1+0 = 1$ carry 0

$1+1 = 0$ carry 1

$1+1+1 = 1$ carry 1

	1	1		
A	0	1	1	1
+B	0	1	1	0
Sum	1	1	0	1

Note that to calculate each bit S_0 of the sum, we need to consider the values of 3 input bits:

- The corresponding bits of A and B, (a_n and b_n).
- The carry-out from the previous addition.

Each bit or column of the binary addition generates two outputs.

- Sum S_0 and carry-out.

Hexadecimal numbers:

• Base 16 representation

• Easy to convert to and from binary numbers

• More compact to write, easier for us to read than binary.

It is used in Microprocessor deals with instructions and data that use hexadecimal number system for programming purposes.

DECIMAL	BINARY	HEXADECIMAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Hexadecimal number conversions:

- **Binary to hexadecinmal:**
  1. Separate the binary number into 4-bit groups.
  2. Replace each group with the hexadecimal equivalent.
- **Hexadecimal to decimal:**
  1. Re-write each hexadecimal digit as the 4-bit binary equivalent.
  2. Convert the binary number to decimal.
- **Decimal to hexadecimal**
  • Repeated division by 16

BINARY CODED DECIMAL (BCD)

- Use 4-bit binary represented one decimal digit
- Easy to convert between decimal $\leftrightarrow$ binary.
- Wastes bits (4 bits can represent 16 values, but only 10 values are used) and 6 value are invalid
- Used extensively in financial calculations.
- It is also known as "8-4-2-1 code"

DECIMAL DIGIT	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

- Convert 0110100100010111 (BCD) to its decimal equivalent:

  0110 1001 0001 0111

  6 9 1 7
- Convert the BCD number 011111000001 to decimal

  0111 1100 0001

  7 ? 1

The forbidden code group indicates an error has occured.

Putting it all together

Decimal	Binary	Octal	Hexadecimal	BCD
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101

Note: BCD code is used in calculators, counters, digital voltmeters, digital clocks etc.

3-bit Binary to Gray code convertor

Binary				Grey code		
B_2	B_1	B_0		G_2	G_1	G_0
0	0	0	0	0	0	0
0	0	1	1	0	0	1
0	1	0	2	0	1	1
0	1	1	3	0	1	0
1	0	0	4	1	1	0
1	0	1	5	1	1	1
1	1	0	6	1	0	1
1	1	1	7	1	0	0

K-map for G_0

$b_2 \mid b_1 b_0$	00	01	11	10
0		1		1
1		1		1

$$G_0 = b_1 \oplus b_2$$

k-map for G_2

$b_2 \backslash b_1b_0$	00	01	11	10
0	0	0	0	0
1	1	1	1	1

$G_2 = b_2$

$G_0 = \Sigma m(1, 2, 5, 6)$

$G_1 = \Sigma m(2, 3, 4, 5)$

$G_2 = \Sigma m(4, 5, 6, 7)$

$G_2 = b_2$

K-map for G_1

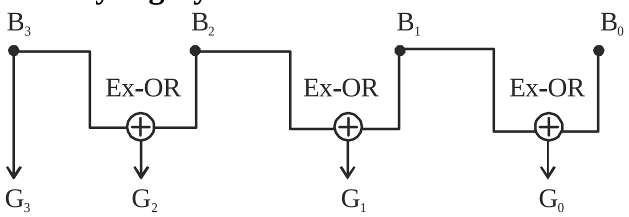
$b_2 \mid b_1 b_0$	00	01	11	10
0			1	1
1		1	1	

$$G_1 = \bar{b}_2 b_1 + b_2 \bar{b}_1 = b_2 \oplus b_1$$

Example:

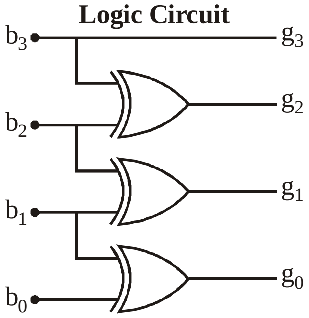
B_2 $\oplus$	B_1 $\oplus$	B_0	Binary	1	1	0	1
$\downarrow$				$\downarrow$			
G_2	G_1	G_0	Grey	1	0	1	1

4-bit binary to gray convertor



$$\left\{\begin{matrix} G_3 & = & B_3 & G_1 & = & B_2 \oplus B_1 \\ G_2 & = & B_3 \oplus B_2 & G_0 & = & B_1 \oplus B_0 \end{matrix}\right\}$$

Logic Circuit



Gray to Binary code convertor

G_2	G_1	G_0			B_2	B_1	B_0
0	0	0	=	0	0	0	0
0	0	1	=	1	0	0	1
0	1	1	=	3	0	1	0
0	1	0	=	2	0	1	1
1	1	0	=	6	1	0	0
1	1	1	=	7	1	0	1
1	0	1	=	5	1	1	0
1	0	0	=	4	1	1	1

$B_0 = \Sigma m(1, 2, 7, 4)$

$B_1 = \Sigma m(2, 3, 4, 5)$

$B_2 = \Sigma m(4, 5, 6, 7)$

K-map for B_2

$G_2 \mid G_1G_0$	00	01	11	10
0				
1	1	1	1	1

$B_2 = G_2$

K-map for B_1

$G_2 \mid G_1G_0$	00	01	11	10
0			1	1
1	1	1		

$B_1 = G_2\bar{G}_1 + \bar{G}_2G_1 = \boxed{G_2 \oplus G_1}$

K-map for B_0

$G_2 \mid G_1G_0$	00	01	11	10
0		1		1
1	1		1	

$B_0 = G_2 \oplus G_1 \oplus G_0$

Convertion:

Gray	G_2	G_1	G_0
Binary	B_2	B_1	B_0

B 1 0 1 1

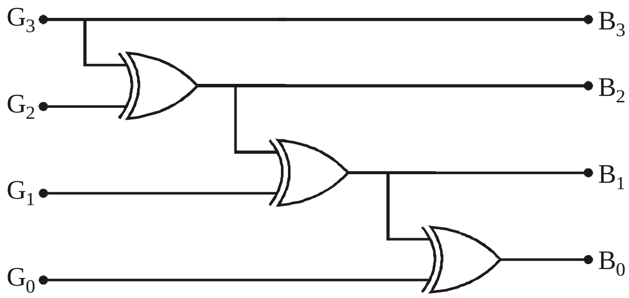
$\downarrow$

G 1 1 1 0

$\downarrow \nearrow \quad \nearrow \quad \nearrow$

B 1 0 1 1

$$B_3 = G_3$$
$$B_2 = G_3 \oplus G_2$$
$$B_1 = G_3 \oplus G_2 \oplus G_1$$
$$B_0 = G_3 \oplus G_2 \oplus G_1 \oplus G_0$$



Logic Circuit

Types of Code

(a) BCD code range 0 to 9

In BCD code each digit is represented in 4 binary bit

$0 \rightarrow 0000$

$1 \rightarrow 0001$

$2 \rightarrow 0010$

$3 \rightarrow 0011$

$4 \rightarrow 0100$

$5 \rightarrow 0101$

$6 \rightarrow 0110$

$7 \rightarrow 0111$

$8 \rightarrow 1000$

$9 \rightarrow 1001$

$10 \rightarrow 0001-0000$

$15 \xrightarrow{\text{BCD}} 0001-0101$

$$\begin{bmatrix} 0 \\ \vdots \\ 9 \end{bmatrix} \text{Binary} = \text{BCD}$$

Note: It is weighted code [8421]

In BCD 10, 11, 12, 12, 14, 15 are don't care.

(b) X-3 code or excess 3 code

X-3 code = BCD code + 3

Range of X-3 code = 3 to 12

6 + 3 = 9

$6 \xrightarrow{\text{X-3}} 1001$

$4 \xrightarrow{\text{X-3}} 0111$

$9 \xrightarrow{\text{X-3}} 1100$

$12 \xrightarrow{\text{X-3}} 12 \rightarrow 0100, 0101$

$$\begin{array}{r} +33 \\ \hline 45 \end{array}$$

Don't care for X-3 code are,

$\boxed{0, 1, 2, 13, 14, 15} = X$

A	B	C	D
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0

Self complement code

E.g. 5211 (+9)

a	b	c	d
0	0	0	0
0	0	0	1
0	0	1	1
⋮	⋮	⋮	⋮
1	1	0	0
1	1	1	0
1	1	1	1

E.g. 84 – 2 – 1 [9]

Again self complement code
E.g. 4311 [9]
Self complementary code

(c) α numeric code/6 bit code
BCDIC $\rightarrow$ (interchanged code) 6-bit
ASCII $\rightarrow$ 7-bit code
EBCDID $\rightarrow$ 7-bit code
Hollerith $\rightarrow$ 12-bit code
(Used in punch card reader).

Gray Codes:

DECIMAL	BINARY	GRAY CODE
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001

Note: Gray code is also known as "minimum change code"
• BCD is "weighted code system"while gray code, excess-3 code are "unweighted code system"
• Only one bit in the code changes in the sequence.
• Useful for industrial control
• Binary code results in glitches. Gray code avoids glitches.

ASCII CODE:

- Codes representing letter is of the alphabet, puctuation marks and other special characters are called alphanumeric codes.
- The most widely used alphanumeric code is the American Standard Code for information interchange or ASCII.
- ASCII is pronounced "askee".
- It is a 7-bit code.

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LIVE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

The basics of signed numbers

- So far, the numbers are assumed to be positive.
  - There is no sign (+ or –) in the representations, so these numbers are called unsigned.
- How can we represent signed numbers?
- **Solution 1: Sign-magnitude**
  - Use one bit to represent the sign, and the remaining bits to represent the magnitude.

sign = 0 → +ve
sign = 1 → –ve

7	6 … 0
sign	magnitude

+27 = 00011011
–27 = 10011011

- **Problems:**
  - Need to handle sign and magnitude separately.
  - Two values for zero (e.g., 00000000, 10000000)
  - Not convenient for arithmetic.

One's complement representation:

In binary number if we replace each 0 by 1 and each 1 by 0, the obtained binary number is called one's complement of the given binary number.

For example:

$(+7)_{10} = 0111$

Sign bit '0' for positive number

$(-7)_{10} = 1000$

Sign bit '1' for negative number

This is obtained by taking one's complement of the given positive number.

The maximum positive and negative number that can be represented using 1's complement are $2^{n-1} - 1$ and $-(2^{n-1} - 1)$ respectively.

Example: The one's complement of a binary number 101010 is

Soln. **Method-1:** One's complement can be obtained by substracting the given number with 111111.

Therefore,
$$\begin{array}{r} 111111 \\ -\ 101010 \\ \hline 010101 \end{array} \Rightarrow \text{1's complement}$$

Method-2: One's complement of the given number can be obtained by replacing each '0' with '1' and each 1 with 0.

Therefore, $101010 \rightarrow 010101$

Two's complement:

- Solution 2 is to represent negative numbers by taking the magnitude inverting all bits, and adding one.
- This is called two's complement. The range of 2's complement number is -2^{n-1} to $(2^{n-1}-1)$

Positive number	+27	= 0001 1011
Invert all bits		1110 0100
Add 1	–27	= 1110 0101

This is called one's complement

- Taking the two's complement again give the original number.

	–27	= 1110 0101	
Invert all bits		0001 1010	
Add 1		0001 1011	= +27

Using 2's complement for signed numbers

- Using 2's complement we can represent positive and negative numbers.
- We call this signed two's complement form
- In order to do this, we change the meaning of the left-most bit (MSB)

• The MSB has a negative weighting in 2's complement form.

Example for 8-bit numbers:

unsigned number: 2^7 2^6 … 2^0 — b_7 … b_0

signed 2's complement: -2^7 2^6 … 2^0 — b_7 | … b_0

b_7 is also known as the sign bit

$$\boxed{x = -\,b_{x-1}2^{n-1} + b_{n-1}2^{n-2} + \ldots + b_1 2^1 + b_0 2^0}$$

The following numbers are in **signed two's complement form:**

0101, 1011, 0111, 1100

What are the decimal values?

2's complement form				Decimal
-8	4	2	1	
0	1	0	1	5
1	0	1	1	-5
0	1	1	1	7
1	1	0	0	-4

Note that the range of 4-bit numbers is different for unsigned and 2's complement.

4-bit unsigned 0 .... +15

4-bit 2's complement -8 ...... +7

Advantages of using 2's complement:

- The major advantage of using two's complement form is that subtraction can be performed by addition using the 2's complement of the substraction.

Example: Using normal substraction

$$\begin{array}{rrl} + & 27 & 0001\,1011_2 \quad \leftarrow \text{minuend} \\ - & 17 & 0001\,0001_2 \quad \leftarrow \text{subtrend} \\ \hline & 10 & 0000\,1010_2 \end{array}$$

27-17 using two's complement addition:

$$\begin{array}{rrl} + & +27 & 0001\,1011_2 \\ + & -17 & 1110\,1111_2 \\ \hline & +10 & 0000\,1010_2 \end{array}$$

r's complement representation:

In r's complement representation 'r' represents the radix. It can be divided as:

- (r – 1)'s complement
- r's complement

Similarly, for any radix we can calculate its compliment in two ways:

Binary number system
- 1's complement
- 2's complement

Decimal number system
- 9's complement
- 10's complement

Octal number system
- 7's complement
- 8's complement

Hexadecimal number system
- F's complement
- 16's complement

To determine (r – 1)'s complement first write maximum number in the given base, then substract the number.

Example: Determine 7's complement of the octal number 5672 ?

Soln. For a octal number maximum possible number of a 4-digit is 7777.

Therefore, 7's complement of 5672 is

$$= \begin{array}{r} 7\,7\,7\,7 \\ -5\,6\,7\,2 \\ \hline 2\,1\,0\,5 \\ \hline \end{array} \leftarrow \text{Ans.}$$

Note: To determine the r's complement, first write (r – 1)'s complement of the given by number, then add 1 in the least significant position.

Example: Determine 8's complement of an octal number 2570 ?

Soln. Firstly (r – 1)'s complement i.e., $\begin{array}{r} 7\,7\,7\,7 \\ -2\,5\,7\,0 \\ \hline 5\,2\,0\,7 \\ \hline \end{array} \leftarrow$ 7's complement

8's complement = 7's complement + 1 = 5207 + 1 = 5208 = 5210.