DIGITAL CIRCUITS

★ QUICK REVISION NOTES ★

NUMBER SYSTEM

DEFINITION

Number System ek method hota hai numbers ko represent karne ka, jismein digits ka use karke values ko express kiya jaata hai.

IMPORTANCE

- Digital Circuits binary (0 & 1) par kaam karte hain.
- Sabhi data (text, image, sound, video) binary mein convert kiya jaata hai.
- Computer memory, logic gates, processors sab binary system use karte hain.

1. TYPES OF NUMBER SYSTEMS

System	Base (Radix)	Digits	Example
1 Binary	2	0, 1	$(1011)_2$
2 Octal	8	0–7	$(753)_8$
3 Decimal	10	0–9	$(462)_{10}$
4 Hexadecimal	16	0–9, A–F	$(2A3F)_{16}$

★ **Note:** A=10, B=11, C=12, D=13, E=14, F=15

2. PLACE VALUE & POSITION

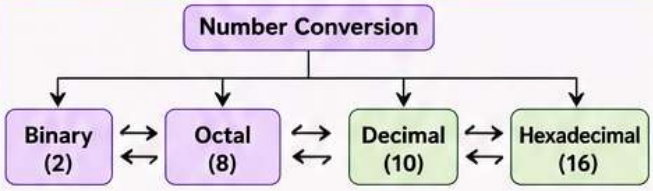
Kisi bhi number ki value uske digit aur uski position (par nirbhar karti hai.

Example: $(1\ 0\ 1\ 1)_2$

2^3	2^2	2^1	2^0
8	4	2	1

Value = $(1\times8) + (0\times4) + (1\times2) + (1\times1) = 11_{10}$

3. NUMBER CONVERSION FLOWCHART



- Binary ↔ Octal : 3 bits = 1 octal digit
- Binary ↔ Hex : 4 bits = 1 hex digit
- Any to Decimal : Apni base ke weights se multiply karke add karein.
- Decimal to Any : Successive division (base se divide karein).

4. MEMORY TRICK

- Binary hai 2 ka power
- Octal group of 3 ka
- Hex group of 4 ka
- Decimal sab ka base 10 ka

Yaad rakho:
3 – 8 – 4
(Octal – Binary – Hex Grouping Rule)

5. CONVERSION TABLE (QUICK REFERENCE)

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

6. BINARY GROUPING RULES

1 Binary to Octal →	Binary digits ko right side se 3-3 ke groups mein divide karo.	3 Bits = 1 Octal Digit
2 Binary to Hex →	Binary digits ko right side se 4-4 ke groups mein divide karo.	4 Bits = 1 Hex Digit
3 Octal/Hex to Binary	Har octal/hex digit ko 3 ya 4 bits binary mein replace karo.	(Reverse Process)

7. EXAMPLE CONVERSIONS

A. Binary to Decimal

$(101101)_2 = 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$
$= 32 + 0 + 8 + 4 + 0 + 1 = 45_{10}$

B. Binary to Octal (Group of 3)

$(1101011)_2 \rightarrow 001\ 101\ 011 \rightarrow (153)_8$

C. Binary to Hex (Group of 4)

$(1101011)_2 \rightarrow 0110\ 1011 \rightarrow (6B)_{16}$

D. Hex to Decimal

$(2A3)_{16} = 2\times16^2 + A\times16^1 + 3\times16^0 = 2\times256 + 10\times16 + 3 = 563_{10}$

8. ONE-LINERS

Computer ki language hai BINARY.	Decimal human friendly hai, Binary machine friendly hai.	Digital circuits sirf 0 aur 1 ko samajhte hain.	Sabhi data akhir mein Binary hai.

DIGITAL CIRCUITS

★ QUICK REVISION NOTES ★

SIGNED NUMBER REPRESENTATION

DEFINITION

Signed Number Representation ka use negative aur positive numbers ko binary form mein represent karne ke liye hota hai. MSB (Most Significant Bit) ka use sign ko indicate karne ke liye kiya jaata hai.

IMPORTANT POINTS

- Binary system naturally unsigned hota hai.
- Signed numbers ke liye different methods use hote hain.
- MSB sign ko show karta hai.
- Positive numbers +ve, Negative numbers −ve.
- Computers arithmetic operations signed numbers par bhi karte hain.

1. SIGN–MAGNITUDE REPRESENTATION

- MSB (leftmost bit) sign ko represent karta hai.
- MSB = 0 → Positive number
- MSB = 1 → Negative number
- Baaki (n−1) bits magnitude ko represent karte hain.

Example (4-bit Sign–Magnitude)

Decimal	Sign	Magnitude (3 bits)	4-bit Sign–Magnitude (Representation)
+5	0	101	0 101
+2	0	010	0 010
+0	0	000	0 000
−0	1	000	1 000
−2	1	010	1 010
−5	1	101	1 101

★ NOTE: Sign–Magnitude mein +0 aur −0 dono exist karte hain.

Range (n-bit): $+(2^{n-1}-1)$ to $-(2^{n-1}-1)$

Example: 4-bit → +7 to −7

2. 1's COMPLEMENT REPRESENTATION

- Negative number = Positive number ke bits ko invert kar do.
- MSB = 0 → Positive number
- MSB = 1 → Negative number

Example (4-bit)

Decimal	Positive (Binary)	Negative (1's Complement)
+5	0101	1010
+2	0010	1101
+0	0000	1111 (−0)
−2	1101	0010
−5	1010	0101

★ NOTE: 1's Complement mein +0 (0000) aur −0 (1111) dono exist karte hain.

Range (n-bit): $+(2^{n-1}-1)$ to $-(2^{n-1}-1)$

Example: 4-bit → +7 to −7

3. 2's COMPLEMENT REPRESENTATION

- Negative number = (1's Complement) + 1
- MSB = 0 → Positive number
- MSB = 1 → Negative number

Example (4-bit)

Decimal	Positive (Binary)	1's Complement	2's Complement
+5	0101	1010	1011
+2	0010	1101	1110
+0	0000	1111	0000 (+0)
−2	1101	0010	0011
−5	1010	0010	0110

★ NOTE: Sirf ek zero hota hai (+0 = −0 = 0000).

Range (n-bit): -2^{n-1} to $+(2^{n-1}-1)$

Example: 4-bit → −7 to +7

4. COMPARISON OF REPRESENTATIONS

Feature	Sign–Magnitude	1's Complement	2's Complement
Sign Bit (MSB)	Yes	Yes	Yes
Positive Number Representation	Same as binary	Same as binary	Same as binary
Negative Number Representation	MSB = 1, Magnitude	Invert all bits	Invert all bits + 1
Zero Representation	+0 and −0 (dono)	+0 and −0 (dono)	Sirf ek Zero (0000)
Arithmetic Operations	Complex	Easy (subtraction)	Easiest (Hardware friendly)
Range (n-bit)	$\pm(2^{n-1}-1)$	$\pm(2^{n-1}-1)$	-2^{n-1} to $+(2^{n-1}-1)$
Most Used	Rare	Rare	Most Common (Used in CPUs)

5. HOW TO GET 2's COMPLEMENT (n-bit)

Binary Number (Positive) → Step 1: Take 1's Complement (Invert all bits) → Step 2: Add 1 to the result → Result = 2's Complement (Negative Number)

Example: Find 2's Complement of $(0101)_2$ (4-bit)

Step	Binary	Explanation
Original Number	0101	Positive (+5)
Step 1: 1's Complement	1010	Invert all bits
Step 2: Add 1	1011	1010 + 1 = 1011

SHORT TRICK:
Right side se 1st '1' tak bits same rehte hain, 1st '1' aur uske baad ke bits invert karo.

Ex: 0101 → 1011 (2's Complement)

6. ARITHMETIC USING 2's COMPLEMENT

- Addition aur Subtraction dono same adder se hote hain.
- Subtraction (A − B) = A + (2's Complement of B)
- Agar final carry aaye → Ignore karo.
- Agar MSB = 1 → Negative result (2's Complement lo).
- Agar MSB = 0 → Positive result.

Example: 4-bit → A = 0101 (+5), B = 0011 (+3), Find A − B

1. 2's Complement of B (0011):
   1's Complement = 1100
   + 1 = 1101
2. A + (2's Complement of B):

```
  0101
+ 1101
-------
1 0010  → Carry = 1 (Ignore)
```

Result = 0010 = +2

Result Rules (2's Complement)

- MSB = 0 → Positive Number
- MSB = 1 → Negative Number

(To get actual magnitude, take 2's Complement again)

7. PROPERTIES OF 2's COMPLEMENT

- Sirf ek zero hota hai.
- Addition aur subtraction hardware friendly.
- Overflow detect karna easy hota hai.
- Most CPUs aur digital systems isi method ka use karte hain.

8. QUICK FACTS

MSB = 0	MSB = 1	2's Complement	Computers	Sign Bit + Magnitude
Positive Number	Negative Number	best for arithmetic operations.	use 2's Complement representation.	ya Inverted form use hota hai.

DIGITAL CIRCUITS

★ QUICK REVISION NOTES ★

ARITHMETIC OPERATIONS ON ALL NUMBER SYSTEM

Including Signed Bit Representation & Complement Methods

DEFINITION

Digital systems mein numbers par addition, subtraction, multiplication aur division perform kiye jaate hain. Ye operations sabhi number systems (Binary, Octal, Decimal, Hexadecimal) mein kiye ja sakte hain.

IMPORTANT POINTS

- Rules same hote hain, difference sirf base (radix) ka hota hai.
- Har number system ka base (r) uske digits decide karta hai.
- Carry/borrow values base ke barabar ya usse manage karna hota hai.
- Decimal = base 10, Binary = base 2, Octal = base 8, Hexadecimal = base 16

SYMBOLS

+	Addition
−	Subtraction
×	Multiplication
÷	Division
C	Carry (r)
B	Borrow (r)

1. GENERAL RULES (BASE = r)

- Digits allowed: 0, 1, 2, ..., (r − 1)
- Addition: Agar sum ≥ r ho, to r subtract karo aur carry 1 le lo.
- Subtraction: Agar top < bottom, to r add karo aur borrow 1 le lo.
- Multiplication: Normal multiplication; products ko base r mein likho.
- Division: Normal division; quotient aur remainder base r mein hote hain.

Operation	Condition	Action	Example (Base r)
Addition	Sum ≥ r	r subtract karo, carry 1	$(7+6)_8 = 15_8 = 17_8$ (1 carry, 5 write)
Subtraction	Top < Bottom	r add karo, borrow 1	$(3-5)_8 = (3+8)-5 = 11-5 = 6$ (borrow)
Multiplication	Product ≥ r	Product ko base r mein convert karo	$(7\times7)_8 = 49_{10} = 61_8$
Division	Remainder < r	Quotient & remainder in base r	$(25 \div 4)_8 = 5\ R\ 1$

2. ADDITION IN ALL NUMBER SYSTEMS

BINARY (Base 2)

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Example: $(1011)_2 + (0110)_2 = 1\ 0\ 0\ 0\ 1\ (= 17_{10})$

OCTAL (Base 8)

A	B	Sum	Carry
7	0	7	0
7	1	0	1

Example: $(725)_8 + (436)_8 = 1\ 3\ 6\ 3_8\ (= 883_{10})$

DECIMAL (Base 10)

A	B	Sum	Carry
9	0	9	0
9	1	0	1

Example: $(5687)_{10} + (3795)_{10} = 9\ 4\ 8\ 2_{10}$

HEXADECIMAL (Base 16)

A	B	Sum	Carry
F	0	F	0
F	1	0	1

Example: $(3AF)_{16} + (7B6)_{16} = B\ 6\ 6\ 5_{16}$

3. SUBTRACTION IN ALL NUMBER SYSTEMS

BINARY (Base 2)

A	B	Diff	Borrow
0	0	0	0
1	0	1	0
0	0	1	1
1	1	0	0

Example: $(10110)_2 - (01101)_2 = 0\ 1\ 0\ 0\ 1_2\ (= 9_{10})$

OCTAL (Base 8)

A	B	Diff	Borrow
0	0	0	0
0	1	7	1

Example: $(572)_8 - (346)_8 = 2\ 2\ 4_8\ (= 148_{10})$

DECIMAL (Base 10)

A	B	Diff	Borrow
0	0	0	0
0	1	9	1

Example: $(8421)_{10} - (3578)_{10} = 4\ 8\ 4\ 3_{10}$

HEXADECIMAL (Base 16)

A	B	Diff	Borrow
0	0	0	0
0	1	F	1

Example: $(7A3)_{16} - (3F6)_{16} = 3\ A\ D_{16}$

4. MULTIPLICATION IN ALL NUMBER SYSTEMS

BINARY (Base 2)

Rules:
- 0 × 0 = 0
- 0 × 1 = 0
- 1 × 0 = 0
- 1 × 1 = 1

Example: $(1011)_2 \times (110)_2$

```
  1011
× 110
------
  0000   (×0)
 1011    (×1 shift left 1)
1011     (×1 shift left 2)
-------
1000010₂  (= 66₁₀)
```

OCTAL (Base 8)

Rules:
- Multiply normally
- Product ko base 8 mein convert karo

Example: $(73)_8 \times (6)_8 = ?$

$73_8 = 7\times8 + 3 = 59_{10}$

$59 \times 6 = 354_{10} = 5\ 6\ 2_8$

DECIMAL (Base 10)

Rules:
- Normal multiplication

Example: $(456)_{10} \times (78)_{10} = 35568_{10}$

HEXADECIMAL (Base 16)

Rules:
- Multiply normally
- Product ko base 16 mein convert karo

Example: $(2A)_{16} \times (1F)_{16} = ?$

$2A_{16} = 2\times16 + 10 = 42_{10}$

$1F_{16} = 31_{10}$

$42 \times 31 = 1302_{10} = 13\ 02_{16}$

5. DIVISION IN ALL NUMBER SYSTEMS

	– Rules –	– Example –
BINARY (Base 2)	• Normal division karo • Remainder hamesha 0 ya 1	$(11011)_2 \div (101)_2 = (101)_2\ R\ (10)_2$
OCTAL (Base 8)	• Normal division karo • Remainder < 8	$(7256)_8 \div (23)_8 = (153)_8\ R\ (7)_8$
DECIMAL (Base 10)	• Normal division karo • Remainder < 10	$(9856)_{10} \div (32)_{10} = 308\ R\ 0$
HEXADECIMAL (Base 16)	• Normal division karo • Remainder < 16	$(2AF6)_{16} \div (1A)_{16} = (16B)_{16}\ R\ (C)_{16}$

NOTE: Division mein quotient aur remainder hamesha same base mein express kiye jaate hain.

6. QUICK COMPARISON (CARRY / BORROW VALUES)

Number System	Base (r)	Digits	Carry Value (= r)	Borrow Value (= r)
Binary	2	0,1	2 (10_2)	2 (10_2)
Octal	8	0–7	8 (10_8)	8 (10_8)
Decimal	10	0–9	10 (10_{10})	10 (10_{10})
Hexadecimal	16	0–9, A–F	16 (10_{16})	16 (10_{16})

7. SIGNED NUMBER REPRESENTATION & COMPLEMENT METHODS

SIGNED NUMBER REPRESENTATION

- Signed numbers = Positive + Negative + Zero.
- MSB (Most Significant Bit) sign indicate karta hai.

Sign Bit (MSB)	Meaning
0	Positive number (including +0)
1	Negative number

Example (4-bit system)

0110 = +6

1110 = −2 (depends on method)

COMPLEMENT METHODS (Base r)

r's Complement of N = $(r^n - 1) - N$ (where n = number of bits/digits)

For Binary (r = 2)

1's Complement (r − 1's Complement)

- Invert all bits (0 ↔ 1)
- r = 2 → 1's Complement

Example (4-bit)

N = 0101 (+5)

1's Comp = 1010 (−5)

2's Complement (r's Complement)

- 1's Complement + 1
- Easiest for arithmetic

Example (4-bit)

N = 0101 (+5)

1's Comp = 1010

2's Comp = 1011 (−5)

PROPERTIES (Binary)

1. 2's Complement of 0 = 0
2. 2's Complement representation range (n-bit): -2^{n-1} to $+(2^{n-1}-1)$
3. Addition/Subtraction hardware same for signed numbers (2's Complement).
4. Overflow tab hota hai jab result range ke bahar ho.

EXAMPLE (4-BIT 2's COMPLEMENT)

Add: (+5) + (−3)

+5 = 0101

−3 (2's Comp) = 1101

Result = 1 0010 (Ignore carry) = 0010 (+2)

HOW COMPLEMENT METHOD ARITHMETIC EASY BANATA HAI?

Subtraction using 2's Complement:
A − B = A + (2's Complement of B)

Example (4-bit): 7 − 3

7 = 0111

3 = 0011 → 2's Comp = 1101

```
 0111
+1101
------
1 0100  → Ignore carry → 0100 = 4
```

Addition using 2's Complement:
Signed numbers ka addition normal binary addition ki tarah hota hai.

Example: (+6) + (−2)

+6 = 0110

−2 (2's Comp) = 1110

```
 0110
+1110
------
1 0100  → Ignore carry → 0100 = +4
```

NOTE

- Binary systems (2's Complement) digital circuits aur computers mein sabse zyada use hote hain.
- Octal/Hex mein bhi complement method same rule se apply hota hai (r base ke hisaab se).

8. ONE-LINERS (QUICK REVISION)

- Sum ≥ r ho to r subtract karo aur carry 1 lo.
- Top < Bottom ho to r add karo aur borrow 1 lo.
- Product ko base r mein convert karo.
- Remainder hamesha < r hota hai.
- 2's Complement signed arithmetic ka best method hai.
- Same hardware se signed aur unsigned dono operate hote hain.
- Rules same, base change → system change.

DIGITAL CIRCUITS

★ QUICK REVISION NOTES ★

BINARY CODES

PART 1 OF 2

DEFINITION

Binary code ek n-bit binary combination hai jo decimal digits (0–9) ya kuch alag types of data ko represent karta hai.

Binary codes digital system mein information representation, storage aur transmission ke liye use hote hain.

IMPORTANT POINTS

- Binary codes ka use: Data representation, Error detection,
- Data transmission, Input devices, Memory addressing.
- n bits se maximum 2^n combinations possible hote hain.
- Standard codes: Weighted codes, Non-weighted codes.
- Weighted codes mein har bit ka fixed weight hota hai.
- Non-weighted codes mein weight fixed nahi hota.

SYMBOLS

0 → Binary 0
1 → Binary 1
n → Number of bits
MSB → Most Significant Bit
LSB → Least Significant Bit

1. WEIGHTED BINARY CODES

1.1 BCD (8421) CODE

Most widely used weighted code.

Weights: 8 4 2 1

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

1.2 2421 CODE

Weights: 2 4 2 1

Decimal	2	4	2	1	2421 Code
0	0	0	0	0	0000
1	0	0	0	1	0001
2	0	0	1	0	0010
3	0	0	1	1	0011
4	0	1	0	0	0100
5	1	0	0	1	1001
6	1	0	1	0	1010
7	1	0	1	1	1011
8	1	1	0	0	1100
9	1	1	0	1	1101

1.3 5211 CODE

Weights: 5 2 1

Decimal	5	2	1	5211 Code
0	0	0	0	0000
1	0	0	1	0001
2	0	1	0	0010
3	0	1	1	0011
4	1	0	0	1000
5	1	0	1	1000
6	1	1	0	1001
7	1	1	1	1100
8	1	1	1	1100
9	1	1	1	1101

1.4 8421 + 3 (EXCESS-3) CODE

It is a non-weighted code but easy conversion from BCD.

Decimal	Excess-3 Code
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

NOTE: Weighted codes mein har bit ki fixed value hoti hai, isliye inka arithmetic operation aasaan hota hai.

2. NON-WEIGHTED BINARY CODES

2.1 GRAY CODE (REFLECTED BINARY CODE)

Sirf 1 bit change hoti hai adjacent numbers between.

Decimal	Gray Code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

2.2 EXCESS-3 CODE (SELF COMPLEMENTING)

Excess-3 code self complementing hota hai.

Decimal	Excess-3 Code
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

2.3 XS-3 (EXCESS-3) GRAY CODE

Gray code + 3 (add 0011 to Gray).

Decimal	XS-3 Gray Code
0	0011
1	0100
2	0101
3	0110
4	1001
5	1010
6	1011
7	1000
8	1111
9	0000

NOTE: Non-weighted codes error detection aur noise reduction mein zyada reliable hote hain (especially Gray code).

3. ALPHANUMERIC CODES

3.1 ASCII CODE (7-bit Code)

Standard code for letters, numbers & symbols.

Character	ASCII (7-bit)
A	1000001
B	1000010
C	1000011
...	...
Z	1011010
0	0110000
1	0110001
2	0110010
...	...
9	0111001

3.2 EBCDIC CODE (8-bit Code)

Extended Binary Coded Decimal Interchange Code.

Character	EBCDIC (8-bit)
A	11000001
B	11000010
C	11000011
...	...
Z	11111010
0	11110000
1	11110001
2	11110010
...	...
9	11111001

3.3 UNICODE (16-bit Code)

Universal character representation.

Character	Unicode (16-bit)
A	0000 0000 0100 0001
B	0000 0000 0100 0010
C	0000 0000 0100 0011
...	...
Z	0000 0000 0101 1010
0	0000 0000 0011 0000
1	0000 0000 0011 0001
2	0000 0000 0011 0010
...	...
9	0000 0000 0011 1001

4. OTHER SPECIAL CODES

4.1 BINARY (STRAIGHT) CODE

Normal binary counting code.

Decimal	Binary Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
...	...
15	1111

4.2 SEQUENTIAL CODE

Simple sequential binary code.

Decimal	Sequential Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
...	...
15	1111

4.3 ONE-HOT CODE

Sirf ek bit "1" hoti hai, baaki sab "0".

Decimal	One-Hot Code (8-bit Example)
0	0000 0001
1	0000 0010
2	0000 0100
3	0000 1000
4	0010 0000
6	0100 0000
7	1000 0000

5. COMPARISON OF CODES

Code	Type	Weight	Self Complementing	Adjacent Change	Error Detection	Usage
BCD (8421)	Weighted	8,4,2,1	No	Multiple bits	Low	Decimal representation, Calculations
2421	Weighted	2,4,2,1	No	Multiple bits	Low	Decimal representation
5211	Weighted	5,2,1,1	No	Multiple bits	Low	Decimal representation
Excess-3	Non-weighted	–	Yes	Multiple bits	Medium	Decimal representation
Gray Code	Non-weighted	–	Yes	1 bit	High	Counters, ADC, Encoders
XS-3 Gray	Non-weighted	–	Yes	1 bit	High	Counters, ADC
ASCII	Alphanumeric	–	No	Multiple bits	Medium	Communication
EBCDIC	Alphanumeric	–	No	Multiple bits	Medium	Mainframe Systems
Unicode	Alphanumeric	–	No	Multiple bits	High	Universal character set
One-Hot	Special	–	No	Multiple bits	Very High	Control systems, State machines

6. KEY POINTS (ONE-LINERS)

- BCD sabse common weighted code hai.
- Gray code mein adjacent numbers mein sirf 1 bit change hoti hai.
- Excess-3 code self complementing hota hai.
- Weighted codes arithmetic operations ke liye best hote hain.
- Non-weighted codes error detection ke liye useful hote hain.
- ASCII 7-bit code hai, EBCDIC 8-bit code hai.
- Unicode 16-bit code hai jo sabhi languages ko support karta hai.
- One-Hot code combinational circuits aur state machines mein use hota hai.

DIGITAL CIRCUITS

★ QUICK REVISION NOTES ★

BINARY CODES

PART 2 OF 2

7. ERROR DETECTION CODES

7.1 PARITY BIT (EVEN PARITY)

Parity bit add kiya jata hai taki total '1's count even ho jaye.

Data (3-bit)	No. of 1's	Parity Bit (P)	Codeword (3-bit + P)
000	0	0	0000
001	1	1	0011
010	1	1	0101
011	2	0	0110
100	1	1	1001
101	2	0	1010
110	2	0	1100
111	3	1	1111

NOTE: Odd parity mein total '1's count odd hota hai.

7.2 HINGE CODE (4-BIT)

Single bit error detect + correct kar sakta hai.

Decimal (4-bit)	Hinge Code				Check Bits (P1,P2)
	P1	P2	D1	D0	
0	0	0	0	0	00
1	0	0	0	1	01
2	0	0	1	0	01
3	0	0	1	1	00
4	0	1	0	0	10
5	0	1	0	1	11
6	0	1	1	0	11
7	0	1	1	1	10
8	1	0	0	0	11
9	1	0	0	1	10
10	1	0	1	0	10
11	1	0	1	1	11
12	1	1	0	0	01
13	1	1	0	1	00
14	1	1	1	0	00
15	1	1	1	1	01

7.3 HAMMING CODE (7-BIT) (SECDED)

Single bit error correct + Double bit error detect.

Data (4-bit)	Hamming Code (7-bit)							Coverage (Check Bits)
	P1	P2	D3	P4	D2	D1	D0	
0000	0	0	0	0	0	0	0	
0001	0	0	0	0	0	0	1	P_1: 1,3,5,7
0010	0	0	0	0	1	1	0	P_2: 2,3,6,7
0011	0	0	0	0	1	0	1	P_4: 4,5,6,7
0100	0	0	1	0	0	0	0	
0101	0	0	1	0	0	0	1	
0110	0	0	1	0	0	1	0	
0111	0	0	1	0	0	0	1	
1000	0	1	0	0	0	0	0	
1001	0	1	0	0	0	0	1	
1010	0	1	0	0	1	1	0	
1011	0	1	0	0	1	0	1	
1100	0	1	1	0	0	0	0	
1101	0	1	1	0	0	0	1	
1110	0	1	1	0	0	1	0	
1111	0	1	1	0	0	0	1	

8. ERROR CORRECTING CODES (MORE CAPACITY)

8.1 BCH CODE (n,k,t)

- Powerful error correcting code.
- Capable of correcting 't' number of errors.
- Example: (15,5,3) BCH code → 15 bits codeword, 5 data bits, correct up to 3 errors.

NOTE: Used in CDs, DVDs, QR codes, Satellite communication.

8.2 REED–SOLOMON CODE (RS CODE)

- Symbol based error correcting code.
- Very effective for burst error correction.
- Example: (255,239) RS code → 255 symbols, 239 data symbols.

NOTE: Used in CDs, DVDs, QR codes, Deep space communication.

8.3 CYCLIC REDUNDANCY CHECK (CRC)

- Error detection code using polynomial division.
- Sender: Data + Generator polynomial → Remainder appended.
- Receiver: If remainder = 0 → No error.

NOTE: Used in Networks, USB, Ethernet, Storage devices.

9. BCD VARIATIONS

9.1 EXCESS-3 BCD (SELF COMPLEMENTING)

Decimal digit mein 3 add karke 4-bit BCD.

Decimal	0	1	2	3	4	5	6	7	8	9
Excess-3 Code	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100

9.2 GRAY CODED BCD (8421 GRAY + BCD)

Binary se Gray code me convert karke BCD.

Decimal	0	1	2	3	4	5	6	7	8	9
Gray Code (4-bit)	0000	0001	0011	0010	0110	0111	0101	0100	1100	1101

9.3 5421 BCD

Another weighted BCD code.

Decimal	0	1	2	3	4	5	6	7	8	9
5421 Code	0000	0001	0010	0011	0100	1011	1100	1101	1110	1111

9.4 7421 BCD

Alternate weighted BCD code.

Decimal	0	1	2	3	4	5	6	7	8	9
7421 Code	0000	0001	0010	0011	0100	1010	1100	1101	1110	1111

10. ASCII CODE (7-BIT)

American Standard Code for Information Interchange.
128 characters represent karne ke liye.

Char.	Decimal	Binary (7-bit)	Char.	Decimal	Binary (7-bit)
A	65	1000001	N	78	1001110
B	66	1000010	O	79	1001111
C	67	1000011	P	80	1010000
D	68	1000100	Q	81	1010001
E	69	1000101	R	82	1010010
F	70	1000110	S	83	1010011
G	71	1000111	T	84	1010100
H	72	1001000	U	85	1010101
I	73	1001001	V	86	1010110
J	74	1001010	W	87	1010111
K	75	1001011	X	88	1011000
L	76	1001100	Y	89	1011001
M	77	1001101	Z	90	1011010

NOTE: Space = 32 (0100000), 0 = 48 (0110000), 9 = 57 (0111001), Enter = 13 (0001101)

11. UNICODE (16-BIT)

Universal character encoding standard.
Supports all languages worldwide.

Character	Unicode (Hex)	Binary (16-bit)
A	U+0041	0000 0000 0100 0001
a	U+0061	0000 0000 0110 0001
0	U+0030	0000 0000 0011 0000
9	U+0039	0000 0000 0011 1001
Space	U+0020	0000 0000 0010 0000
₹ (Rupee)	U+20B9	0010 0000 1011 1001

12. SUMMARY TABLE OF ALL CODES

Code	Type	Bits	Self Complementing	Error Detection	Error Correction	Usage
BCD (8421)	Weighted	4	No	No	No	Decimal representation
2421	Weighted	4	No	No	No	Decimal representation
5211	Weighted	4	No	No	No	Decimal representation
Excess-3	Non-weighted	4	Yes	No	No	Decimal representation
Gray Code	Non-weighted	n	Yes	Yes (1 bit)	No	Counters, ADC, Encoders
XS-3 Gray	Non-weighted	n	No	Yes (1 bit)	No	Counters, ADC
ASCII	Alphanumeric	7	No	No	No	Communication
EBCDIC	Alphanumeric	8	No	No	No	Mainframe Systems
Unicode	Alphanumeric	16	No	No	No	Universal character set
Hamming	Error Correcting	n	—	Yes	Yes (1 bit)	Memory, Communication
BCH / RS	Error Correcting	n	—	Yes	Yes (Multiple)	Storage, Deep space
CRC	Error Detecting	n	—	Yes	No	Networks, Data transfer

13. ONE-LINERS (QUICK REVISION)

- Gray code → Only 1 bit change between adjacent numbers.
- Excess-3 code → Self complementing hota hai.
- Hamming code → Single bit error correct, double bit detect.
- BCD → Decimal representation ka sabse common code.
- ASCII → 7-bit alphanumeric code.
- EBCDIC → 8-bit code, mainframe systems use karte hain.
- Unicode → 16-bit code, sabhi languages support karta hai.
- Parity bit → Error detection ke liye simple method.
- CRC → Polynomial division based error detection.
- Reed–Solomon → Burst errors ke liye best.

SYMBOLS RECAP

n	→	Binary 0
1	→	Binary 1
n	→	Number of bits
MSB	→	Most Significant Bit
LSB	→	Least Significant Bit

DIGITAL CIRCUITS

★ QUICK REVISION NOTES ★

K-MAP (KARNAUGH MAP)

DEFINITION

K-Map (Karnaugh Map) ek graphical technique hai jiska use Boolean expression ko simplify karne ke liye kiya jata hai.
Ye truth table ki values ko arrange karta hai taaki adjacent cells ke similar terms ko group karke minimum expression mil sake.

IMPORTANT POINTS

- Cells Gray Code order me arrange hote hain.
- Adjacent cells (horizontal/vertical) me sirf 1 variable change hota hai.
- Corners bhi adjacent hote hain.
- Grouping size hamesha 1, 2, 4, 8, 16... (power of 2) hoti hai.
- Groups rectangular hone chahiye.
- Groups jitne bade honge, expression utna simple hoga.

SYMBOLS & TERMS

0 → Logic 0
1 → Logic 1
X → Don't Care
Adjacent → Side ya corner se touch karne wale cells
Group → 1, 2, 4, 8, 16... cells ka combination
Prime Implicant → Maximum size ka group
Essential Prime Implicant → Jo cover kare koi atleast ek 1 jo dusre group se cover nahi hota.

1. K-MAP ARRANGEMENT (GRAY CODE ORDER)

2 Variables (2×2)

A \	0	1
A,	m0	m1
1	m2	m3

3 Variables (2×4)

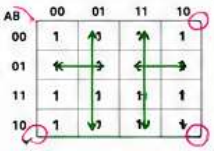
BC	00	01	11	10
A,	m0	m1	m3	m2
1	m4	m5	m7	m6

4 Variables (4×4)

AB	00	01	11	10
00	m0	m1	m3	m2
01	m4	m5	m7	m6
11	m12	m13	m15	m14
10	m8	m9	m11	m10

ADJACENT CELLS

- Horizontal / Vertical neighbours
- Corner (wrap-around) neighbours



2. HOW K-MAP WORKS?

1. Truth table se 1's (and X's) ko K-Map me fill karo.
2. Adjacent 1's ko largest possible groups me group karo.
3. Har group ke liye common variables identify karo.
4. OR (+) operations ke saath terms ko combine karo.

GROUPING RULES

- Group size = 1, 2, 4, 8, 16... (powers of 2 only)
- Groups rectangular hone chahiye.
- Groups overlap kar sakte hain.
- Largest possible groups banao.
- Don't care (X) ko 1 ki tarah use kar sakte hain agar expression simplify hota hai.

4. HOW TO GET TERM FROM A GROUP?

Variable jiska value group ke sabhi cells me same ho, wo term me rahega. Jo change hota hai, wo eliminate.

Example:

BC	00	01	11	10
A	1	1	1	1
1	0	0	0	0

Top row me A = 0 constant hai.
Isliye term = A'

3. GROUPING EXAMPLES

Single Cell (1)

0	1
0	0

Term = A'B

Pair (2)

1	1
0	0

Term = A'
(B eliminate hogaya)

Group of 4

1	1
1	1

Term = 1
(Sab variables eliminate)

Group of 8 (in 3-variable K-Map)

1	1	1	1
1	1	1	1

Term = B
(A & C eliminate)

Group of 4 (With Don't Care)

1	X
1	X

Term = A
(X used as 1)

5. SOLVED EXAMPLES

Example 1: 2-Variable K-Map

F(A,B) = Σm(1,2,3)

B	0	1
A 0	0	1
1	1	1

Groups:
- m_1, m_3 → B
- m_2, m_3 → A

F = A + B

Example 2: 3-Variable K-Map

F(A,B,C) = Σm(0,1,2,5,6,7)

A	00	01	11	10
0	1	1	0	1
1	0	1	1	1

Groups:
- (m_0, m_1) → A'B'
- (m_5, m_7, m_6, m_4) → B
- (m_6, m_7) → AC

F = A'B' + B + AC

Example 3: 4-Variable K-Map (with Don't Care)

F(A,B,C,D) = Σm(0,1,3,4,5,7, X = d(2,6))

AB	00	01	11	10
00	1	1	1	X
01	1	1	1	0
11	0	0	X	0
10	0	0	1	0

Groups:
- $(m_0, m_1, m_3, m_2(X))$ → A'
- (m_4, m_5) → AB'C'
- (m_1, m_5, m_7, m_3) → B D

F = A' + AB'C' + BD

TIPS & TRICKS

- Always start from largest groups.
- Use Don't cares to make bigger groups.
- Overlapping allowed, but redundant groups avoid karo.
- Essential Prime Implicants ko zarur cover karo.
- Leftover 1's ko minimum extra groups se cover karo.

GROUP SIZE vs ELIMINATED VARIABLES

Group Size	No. of Cells	No. of Variables Eliminated
1	1	0
2	2	1
4	4	2
8	8	3
16	16	4

VARIABLE ELIMINATION EXAMPLE

BC	00	01	11	10
0	1	1	1	1
1	1	1	1	1

All 8 cells = 3 variables eliminate
Term = 1 (Constant 1)

ADVANTAGES

- Simple & visual method.
- Fast simplification.
- Reduces hardware cost.
- Minimizes gates & connections.
- Widely used in digital circuit design.

ONE-LINER

K-Map ka goal hai → Maximum grouping, Minimum expression, Minimum hardware.

REMEMBER

Bada group = Zyada variables eliminate = Zyada simplification.

USE OF K-MAP

Combinational circuit design, Logic minimization, Digital system optimization.