# DIGITAL ELECTRONICS AND LOGIC DESIGN

*Unit II — Combinational Logic Design & Code Converters*

**Topics Covered in this Unit**

Classification of Digital Codes: Weighted (BCD 8421, 2421, 5421) vs. Non-Weighted Codes (Excess-3, Gray Code)
Self-Complementary Codes (9's Complement Property) & Reflective Codes (Unit Distance Gray Code)
Error Detecting & Correcting Codes: Parity Generation/Checking & 7-Bit Hamming Code Mechanics
Code Converters: Binary-to-Gray, Gray-to-Binary & BCD-to-Excess-3 Design using K-Maps & XOR Gates
Arithmetic Combinational Circuits: Half Adder, Full Adder (using 2 Half Adders + OR gate), Half & Full Subtractors
4-Bit Parallel Binary Adder / Subtractor (Ripple Carry Adder vs. Carry Lookahead Adder - CLA)
BCD Decade Adder: Illegal BCD States, Decimal Sum Correction (+0110_2) & Circuit Architecture
Data Routing Circuits: Multiplexers (2:1, 4:1, 8:1, 16:1 MUX) & Demultiplexers (1:2, 1:4, 1:8, 1:16 DEMUX)
Universal Logic Implementation of SOP & POS Boolean Expressions using MUX and DEMUX
Magnitude Comparators: 1-Bit, 2-Bit & 4-Bit Magnitude Comparator (IC 7485 Cascading Architecture)
High-Yield University Exam Question Bank with Detailed Model Answers & Unit Summary

*Compiled in a structured, easy-to-understand, textbook style format*
*Specially curated for B.Tech / B.E. / BCA / MCA / B.Sc Electronics, Electrical, CS & AI Students*

# COURSE SYLLABUS & MAPPING

**[ UNIT II SYLLABUS — PRESCRIBED UNIVERSITY CURRICULUM ]**

*This section contains the official syllabus for Unit II. Verify with your university curriculum mapping.*

**Unit II Syllabus Breakdown:**

Unit II Combinational Logic Design: Classification of Codes : Weighted and Non-weighted Codes, Error Detecting and Correcting Codes ,Self-complementary codes, Reflective Codes ( Binary , BCD, Gray code and Excess-3 code), Code Conversion : Binary to Gray and Gray to Binary code conversion,BCD to Excess-3 Half-Adder, Full Adder, Half Subtractor, Full Subtractor, 4 Bit Parallel Adder, BCD adder,Multiplexers (MUX), Demultiplexers (DEMUX) , Implementation of SOP and POS using MUX, DMUX,Magnitude Comparator

• Course Code: _____ • Semester / Year: _____

# TABLE OF CONTENTS

# 2.1 Classification of Digital Codes & Error Correction

| Code Category | Defining Characteristic | Representative Code Examples | Applications & Properties |
|---|---|---|---|
| Weighted Codes | Each bit position is assigned a fixed positional weight (w_3, w_2, w_1, w_0). | BCD 8421, 2421, 5421, 84-2-1. | Direct numerical arithmetic, digital displays, ADC/DAC. |
| Non-Weighted Codes | Bits do not have assigned positional arithmetic weights. | Excess-3 (XS-3), Gray Code. | Shift register counters, optical shaft encoders, arithmetic carry logic. |
| Self-Complementary Codes | The 1's complement of the code word directly produces the 9's complement of the decimal digit. | Excess-3, 2421, 84-2-1. (Condition: Sum of weights = 9). | Simplifies decimal subtraction circuitry in BCD processors. |
| Reflective / Unit Distance | Adjacent code words differ by only ONE bit change (Hamming distance = 1). | Gray Code (Reflected Binary Code). | Shaft angle encoders, eliminating glitch spikes in state transitions. |

## Error Detecting and Correcting Codes: Hamming Code

**Definition: 7-Bit Hamming Code (Richard Hamming, 1950)**

*Hamming code adds k redundant Parity Bits (P_1, P_2, P_4, ...) at bit positions that are powers of 2 (1, 2, 4, 8) to m data bits, satisfying: 2^k ≥ m + k + 1.*

- *Parity Bit Assignment: P_1 checks bit positions with LSB=1 (1, 3, 5, 7); P_2 checks bits with 2nd bit=1 (2, 3, 6, 7); P_4 checks bits with 3rd bit=1 (4, 5, 6, 7).*
- *Error Syndrome: S = [S_4 S_2 S_1]_2 gives the exact bit position of the error (S=000 indicates zero errors).*

# 2.2 Code Converters: Binary ↔ Gray & BCD ↔ Excess-3

## 1. Binary to Gray Code Conversion

- **• Logic Equations:**

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

MSB is copied directly; each subsequent Gray bit is obtained by XORing adjacent binary bits.

### 2. Gray to Binary Code Conversion

- • **Logic Equations:**

$B\_3 = G\_3,\ \ B\_2 = B\_3 \oplus G\_2,\ \ B\_1 = B\_2 \oplus G\_1,\ \ B\_0 = B\_1 \oplus G\_0$

MSB is copied directly; each subsequent binary bit is obtained by XORing previous calculated binary bit with current Gray bit.

### 3. BCD to Excess-3 Converter

- • **Mechanism:** Adds binary 0011_2 (+3) to the 4-bit BCD digit (A, B, C, D). Derived using 4-variable K-maps:

$W = A + B\,C + B\,D,\ \ X = B'\,C + B'\,D + B\,C'\,D',\ \ Y = C \oplus D,\ \ Z = D'$

## 2.3 Arithmetic Circuits: Adders, Subtractors & BCD Adder

| Arithmetic Circuit | Boolean Sum / Difference Equation | Boolean Carry / Borrow Equation | Gate Implementation |
|---|---|---|---|
| Half Adder | Sum = A ⊕ B | Carry = A • B | 1 XOR gate + 1 AND gate |
| Full Adder | Sum = A ⊕ B ⊕ C_in | C_out = A B + B C_in + A C_in = A B + C_in (A ⊕ B) | 2 Half Adders + 1 OR gate |
| Half Subtractor | Diff = A ⊕ B | Borrow = A' • B | 1 XOR gate + 1 NOT gate + 1 AND gate |
| Full Subtractor | Diff = A ⊕ B ⊕ B_in | B_out = A' B + B B_in + A' B_in = A' B + B_in (A ⊕ B)' | 2 Half Subtractors + 1 OR gate |

### 4-Bit Parallel Adder / Subtractor & Carry Lookahead Adder (CLA)

- • **4-Bit Adder/Subtractor:** Cascades 4 Full Adders using an M-mode control line (M=0 for Addition; M=1 for Subtraction via B ⊕ M and C_0=M).
- • **Carry Lookahead Adder (CLA):** Eliminates ripple carry propagation delay by computing carries in parallel using Generate (G_i = A_i B_i) and Propagate (P_i = A_i ⊕ B_i) terms: C_1 = G_0 + P_0 C_0, C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0, etc. Reduces delay from O(n) to O(1) constant time!

### BCD Decade Adder

**BCD Adder Architecture & Correction Logic**

*A BCD Adder adds two 4-bit BCD digits using a 4-bit binary adder. If the binary sum exceeds 9 (Sum > 1001_2) or produces an output carry C_out = 1, it represents an Invalid BCD State. The circuit detects this condition via:*

*Correction_Detect = C_out + S_3 S_2 + S_3 S_1*

*When detected, binary 0110_2 (+6) is added via a second 4-bit adder to skip the 6 illegal states and generate the final correct BCD digit and BCD carry.*

## 2.4 Multiplexers (MUX) & Demultiplexers (DEMUX)

**Definition: Multiplexer (Data Selector) & Demultiplexer (Distributor)**

• *Multiplexer (2^n-to-1 MUX): Selects one of 2^n input data lines and routes it to a single output line based on the binary state of n select lines (S_n-1, ..., S_0).*

• *Demultiplexer (1-to-2^n DEMUX / Decoder with Enable): Routes a single input data line to one of 2^n output lines based on n select lines.*

### Universal Logic Implementation using Multiplexers

Any Boolean function of n variables can be implemented directly using a 2^(n-1)-to-1 MUX (connecting n-1 variables to Select lines, and deriving MUX data inputs from the remaining variable: 0, 1, Variable, or Variable').

## 2.5 Magnitude Comparators (IC 7485)

- • **2-Bit Comparator Logic:** For inputs A = A_1 A_0 and B = B_1 B_0:
  - • **A = B:** (A_1 ⊙ B_1) • (A_0 ⊙ B_0)
  - • **A > B:** A_1 B_1' + (A_1 ⊙ B_1) A_0 B_0'
  - • **A < B:** A_1' B_1 + (A_1 ⊙ B_1) A_0' B_0

## 2.6 High-Yield University Exam Question Bank & Model Answers

## Part A: Short Answer Questions (2–3 Marks)

- **Q1. What are Self-Complementary codes? Give two examples.**

Answer: A self-complementary code has the property that the 1's complement of the binary code word for decimal digit d produces the exact code word for (9 - d) (the 9's complement of d). Examples: Excess-3 code and 2421 code.

- **Q2. Why is Gray code called a Unit Distance Code?**

Answer: Gray code is called a unit distance (reflective) code because any two successive code words differ by only a single binary bit (Hamming distance = 1), preventing false transient switching states (glitches) in optical shaft encoders.

- **Q3. State the logic equations for a Full Adder using two Half Adders.**

Answer: Sum = (A $\oplus$ B) $\oplus$ C_in; Carry_Out = A B + C_in (A $\oplus$ B).

- **Q4. Why is binary 0110 (6) added in BCD addition?**

Answer: In 4-bit binary, there are 16 total states (0-15), but BCD only uses 10 valid states (0-9). If a sum exceeds 9, adding 0110_2 (+6) skips the 6 invalid states (1010 to 1111), correctly rolling the sum over into the next BCD decade digit.

## Part B: Long Answer Questions (8–10 Marks)

- **Q5. Design a BCD Adder circuit in detail. Explain the invalid BCD condition detection logic and draw the complete circuit diagram using two 4-bit binary adders (IC 7483).**

Model Answer Outline:

- 1. Principle of BCD arithmetic: Valid range 0000 to 1001.
- 2. Analysis of invalid conditions: When binary sum is between 10 and 19 (Sum > 9 or Carry Out = 1).
- 3. Derivation of Correction Condition K = C_out + S_3 S_2 + S_3 S_1 using truth table and K-map.
- 4. Two-stage architecture: Adder 1 computes raw binary sum; Correction logic detects invalid state; Adder 2 adds 0110_2 when K=1.
- 5. Complete architectural schematic showing IC 7483 adders, OR gates, and AND gates.

- **Q6. Implement the Boolean function F(A, B, C, D) = $\sum$ m(1, 3, 4, 11, 12, 13, 14, 15) using: (1) An 8-to-1 Multiplexer, and (2) A 4-to-1 Multiplexer with external gates.**

Model Answer Outline:

- 1. Implementation using 8:1 MUX: Connecting A, B, C to Select Lines S_2, S_1, S_0.
- 2. Derivation table mapping minterms to MUX Data Inputs (I_0 to I_7) as functions of variable D (D, D', 0, 1).

- 3. Implementation using 4:1 MUX: Connecting A, B to S_1, S_0 and deriving inputs from C, D using K-maps.
- 4. Complete logic circuit diagrams for both implementations.

## 2.7 Unit II Summary & Quick Revision Sheet

| Combinational Topic | Key Theoretical & Design Takeaways for Exams |
|---|---|
| Code Systems | Weighted (8421, 2421), Non-weighted (Excess-3, Gray). Excess-3 is self-complementary. Gray is unit distance. |
| Hamming Codes | 7-Bit single error correction (2^k ≥ m+k+1). Parity bits at positions 1, 2, 4. Syndrome gives error bit position. |
| Code Converters | Binary → Gray (G_i = B_i ⊕ B_(i+1)), Gray → Binary (B_i = B_(i+1) ⊕ G_i). BCD → XS-3 (Add 0011). |
| Adders & Subtractors | Full Adder = 2 HAs + OR gate. CLA Adder computes C_i in parallel via G_i (Generate) and P_i (Propagate). |
| BCD Adder & MUX | BCD Adder: Detect K = C_out + S_3 S_2 + S_3 S_1 and add 0110_2 (+6). MUX is a universal logic implementer. |