EEG224 - INTRODUCTION TO SWITCHING AND LOGIC SYSTEMS (2 UNITS)

COURSE CONTENT

1. Number systems and codes:

- Review of Binary, Octal and Hexadecimal number systems

- Conversion methods: Complements, Signed and Unsigned Binary numbers

- Binary codes: Weighted and non-weighted codes

- Error detecting and error correcting codes

- Hamming codes

- Code converters: Binary to gray, gray to binary, gray to binary, bcd to excess 3 code.

2. Boolean Algebra:

- Switching functions

- Boolean prostitudes and laws: De Morgan’s theorem

- Boolean functions

- Minimisation of boolean expressions: Sum of products, product of sums

- Canonical forms – Karnaugh map minimisation (K-map)

- Don’t care conditions.

- Logic gates: AND, OR, NOT, NAND, NOR, X-OR X-NOR,

- Implementation of logic functions using basic gates NAND and NOR implementations

3. Combinational Logic Design:

- Definition

- Design procedure: Adders, Subtractors, Serial Adder/Subtractor, Parallel adder/Subtractor

- Magnitude Comparator – Multiplexer / Demultiplexer, Encoder/Decoder

- Parity Checker: Implementation of combinational logic using MUX, Decoder

4. Sequential Circuits:

- Flip Flops: SR Flipflops, JK flipflops, T, D, Master-Slave flip flops

- Characteristic table and equation: Application table, Edge triggering, Level triggering, Realization of one flipflop using other flip flops, Asynchronous/Ripple counters, Synchrounous counters – Modular couonters

- Number systems conversion between bases, arithmetic with bases other than ten, 1 ans 2s complement, BCD,

weighted and unweighted codes; Gray codes.

- Truth function and truth tables.

- Boolean algebra and De-Morgan theorem, truth function set or venn diagram and truth tables.

- Minimization of boolean function using boolean algebra and karnaught map(K.-Map).

- Switching relays, logic circuits.

- Realization of simple combinational circuits, binary single bit address, simple code conversion, bit comparators.

- Introduction to multi-vibrator circuits; Astable, Mono-stable and Bi-stable. Pre-requisite: EEG213

Binary number representation

1. Magnitude

1. Unsigned: Only positive binary numbers

2. Signed: Both

2. Complement

1. 1’s Complement: Both positive and negative binary numbers

2. 2’s Complement: Both positive and negative binary numbers

In all representations, positive numbers are represented in the same way

Unsigned:

+6 → 110

-6 → Can’t be represented

Signed Representation:

+6 → 110 → (adding a signed bit) 0110

-6 → 110 → (adding the signed bit) 1110.

110 is the magnitude

0 → +ve

1 → -ve

**Binary Coded Decimal**, or **BCD**, is another process for converting decimal numbers into their binary equivalents.

- It is a form of binary encoding where each digit in a decimal number is represented in the form of bits.

- This encoding can be done in either 4-bit or 8-bit (usually 4-bit is preferred).

- In a 4-bit method, each decimal digit is represented by a 4-bit binary number.

- It is a fast and efficient system that converts the decimal numbers into binary numbers as compared to the existing binary system.

- These are generally used in digital displays where is the manipulation of data is quite a task.

- Thus BCD plays an important role here because the manipulation is done treating each digit as a separate single sub-circuit.

Decimal number and decimal digits…

Positional Weight: The positional weights are 8-4-2-1, hence BCD code is also called 8-4-2-1

The BCD equivalent of a [decimal number](https://www.geeksforgeeks.org/decimal-number-system/) is written by replacing each decimal digit in the integer and fractional parts with its four bit [binary](https://www.geeksforgeeks.org/binary-number-system/) equivalent.the BCD code is more precisely known as 8421 BCD code  , with 8,4,2 and 1 representing the weights of different bits in the four-bit groups, Starting from MSB and proceeding towards LSB. This feature makes it a weighted code , which means that each bit in the four bit group representing a given decimal digit has an assigned weight.  
Many decimal values, have an infinite place-value representation in binary but have a finite place-value in binary-coded decimal. For example, 0.2 in binary is .001100… and in BCD is 0.0010. It avoids fractional errors and is also used in huge financial calculations.

BCD numbers representing decimal numbers beyond 9 are called **packed BCD**.

000101010110 is a packed BCD.

BCD is less efficient than binary number conversion because it uses more number of bits than binary.

The arithmetic operations of BCD (like addition and subtraction) are more complicated than binary numbers)

Though BCD is somewhat inefficient, the main advantage of the BCD code is the relative ease of converting to and from decimal.

(R-1)’s compliment

1. Find the 1’s compliments of (1010)\_2

2. Obtain 2’s compliments of (10111010)\_2

r-1’s compliment = r^n – N -1

=> (r^n – 1) – N

=> If r = 10,

r^n – 1 = 999999...n times

=> If n=4,

r^4 – 1 = 999

=> If r=8, n=4,

=> r^n – 1 = 777

The capital N, is the number given in the question.

Example,

Find the 1’s compliments of (1010)\_2.

N = 1010

As seen above, there are 4 digits

r=2, n=5

(2^5 – 1) – 1010

1111-1010

= 0101

Now if you look at this, the answer we got, is exactly the negation of each element

1010 -> 0101

So the next time, you can simpley find the compliment.

2’s compliments

1. Find 1’s compliment and then add 1 to it

Example:

Obtain 2’s compliments of (10111010)\_2

The 1’s compliment is 01000101

Then we add 1

01000101 + 1 = 01000110

SHORTCUT FOR THE TWO’S METHOD

1. Write down the number

2. Starting from LSB (least significant bit) from the right, copy all the zeros till the first (1) that you find

3. Copy the first number

4. Negate the remaining numbers