Digital Electronics & Logic Design

Dr. Shilpi Gupta

Associate Professor

Electronics Engineering Department

SVNIT, Surat

Syllabus

- **PN DIODE AND TRANSITOR** **(04 Hours)**

  PN Diode Theory, PN Characteristic and Breakdown Region, PN Diode Application as Rectifier, Zener Diode Theory, Zener Voltage Regulator, Diode as Clamper and Clipper, Photodiode Theory, LED Theory, 7 Segment LED Circuit Diagram and Multi Colour LED, LASER Diode Theory and Applications, Bipolar Junction Transistor Theory, Transistor Symbols And Terminals, Common Collector, Emitter and Base Configurations, Different Biasing Techniques, Concept of Transistor Amplifier, Introduction to FET Transistor And Its Feature.

- **WAVESHAPING CIRCUITS AND OPERATIONAL AMPLIFIER** **(06 Hours)**

  Linear Wave Shaping Circuits, RC High Pass and Low Pass Circuits, RC Integrator and Differentiator Circuits, Nonlinear Wave Shaping Circuits, Two Level Diode Clipper Circuits, Clamping Circuits, Operational Amplifier OP-AMP with Block Diagram, Schematic Symbol of OP-AMP, The 741 Package Style and Pinouts, Specifications of Op-Amp, Inverting and Non-Inverting Amplifier, Voltage Follower Circuit, Multistage OP-AMP Circuit, OP-AMP Averaging Amplifier, OP-AMP Subtractor.

- **BOOLEAN ALGEBRA AND SWITCHING FUNCTIONS** **(04 Hours)**

  Basic Logic Operation and Logic Gates, Truth Table, Basic Postulates and Fundamental Theorems of Boolean Algebra, Standard Representations of Logic Functions- SOP and POS Forms, Simplification of Switching Functions-K-Map and Quine-Mccluskey Tabular Methods, Synthesis of Combinational Logic Circuits.

- **COMBINATIONAL LOGIC CIRCUIT USING MSI INTEGRATED CIRCUITS** **(07 Hours)**

  Binary Parallel Adder; BCD Adder; Encoder, Priority Encoder, Decoder; Multiplexer and Demultiplexer Circuits; Implementation of Boolean Functions Using Decoder and Multiplexer; Arithmetic and Logic Unit; BCD to 7-Segment Decoder; Common Anode and Common Cathode 7-Segment Displays; Random Access Memory, Read Only Memory And Erasable Programmable ROMS; Programmable Logic Array (PLA) and Programmable Array Logic (PAL).

- **INTRODUCTION TO SEQUENTIAL LOGIC CIRCUITS** **(04 Hours)**

  Basic Concepts of Sequential Circuits; Cross Coupled SR Flip-Flop Using NAND or NOR Gates; JK Flip-Flop Rise Condition; Clocked Flip-Flop; D-Type and Toggle Flip-Flops; Truth Tables and Excitation Tables for Flip-Flops; Master Slave Configuration; Edge Triggered and Level Triggered Flip-Flops; Elimination of Switch Bounce using Flip-Flops; Flip-Flops with Preset and Clear.

- **SEQUENTIAL LOGIC CIRCUIT DESIGN** **(06 Hours)**

  Basic Concepts of Counters and Registers; Binary Counters; BCD Counters; Up Down Counter; Johnson Counter, Module-N Counter; Design of Counter Using State Diagrams and Table; Sequence Generators; Shift Left and Right Register; Registers With Parallel Load; Serial-In-Parallel-Out (SIPO) And Parallel-In-Serial-Out(PISO); Register using Different Type of Flip-Flop.

- **REGISTER TRANSFER LOGIC** **(04 Hours)**

  Arithmetic, Logic and Shift Micro-Operation; Conditional Control Statements; Fixed-Point and Floating-Point Data; Arithmetic Shifts; Instruction Code and Design Of Simple Computer.

- **PROCESSOR LOGIC DESIGN** **(03 Hours)**

  Processor Organization; Design of Arithmetic Logic Unit; Design of Accumulator.

- **CONTROL LOGIC DESIGN** **(04 Hours)**

  Control Organization; Hard-Wired Control; Micro Program Control; Control Of Processor Unit; PLA Control.

Books Recommended

1. Donald L. Schilling and E. Belove, Electronics Circuits- Discrete and Integrated, 3rd Edition, McGraw-Hill, 1989, Reprint 2008.
2. Millman Jacob, Halkias Christos and C. Parikh, Integrated Electronics, 2nd Edition, McGraw-Hill, 2009.
3. H. Taub, Mothibi Suryaprakash and Millman J., Pulse, Digital and Switching Waveforms, 2nd Edition, McGraw-Hill, 2007.
4. Mano Morris, Digital Logic and Computer Design, 5th Edition, Pearson Education, 2005.
5. Lee **Samual**, Digital Circuits and Logic Design, 1st Edition, PHI, 1998.

Other Reference material:

Robrert L. Boylestad and Louis Nashelsky, Electronics Devices and Circuit Theory, 11th edition, Pearson education, 2013 (e book available)

J. B. Gupta, Electronics Devices and Circuits, 5th edition, 2013, S. K. Kataria & Sons 5th Edition, S. K. Kataria & Sons

Related NPTEL Videos

Course Outcome

CO1	acquire knowledge about different types of diodes and circuits.
CO2	apply the knowledge of gates, Boolean algebra and operational amplifier in designing logical and integrated circuits.
CO3	analyse the logical, integrated, and operational amplifier based circuits.
CO4	evaluate the different circuits and compare their performance.
CO5	design ALU and control unit.

Portion to be covered

- **BOOLEAN ALGEBRA AND SWITCHING FUNCTIONS** **(04 Hours)**

  Basic Logic Operation and Logic Gates, Truth Table, Basic Postulates and Fundamental Theorems of Boolean Algebra, Standard Representations of Logic Functions- SOP and POS Forms, Simplification of Switching Functions-K-Map and Quine-Mccluskey Tabular Methods, Synthesis of Combinational Logic Circuits.

- **COMBINATIONAL LOGIC CIRCUIT USING MSI INTEGRATED CIRCUITS** **(07 Hours)**

  Binary Parallel Adder; BCD Adder; Encoder, Priority Encoder, Decoder; Multiplexer and Demultiplexer Circuits; Implementation of Boolean Functions Using Decoder and Multiplexer; Arithmetic and Logic Unit; BCD to 7-Segment Decoder; Common Anode and Common Cathode 7-Segment Displays; Random Access Memory, Read Only Memory And Erasable Programmable ROMS; Programmable Logic Array (PLA) and Programmable Array Logic (PAL).

- **REGISTER TRANSFER LOGIC** **(04 Hours)**

  Arithmetic, Logic and Shift Micro-Operation; Conditional Control Statements; Fixed-Point and Floating-Point Data; Arithmetic Shifts; Instruction Code and Design Of Simple Computer.

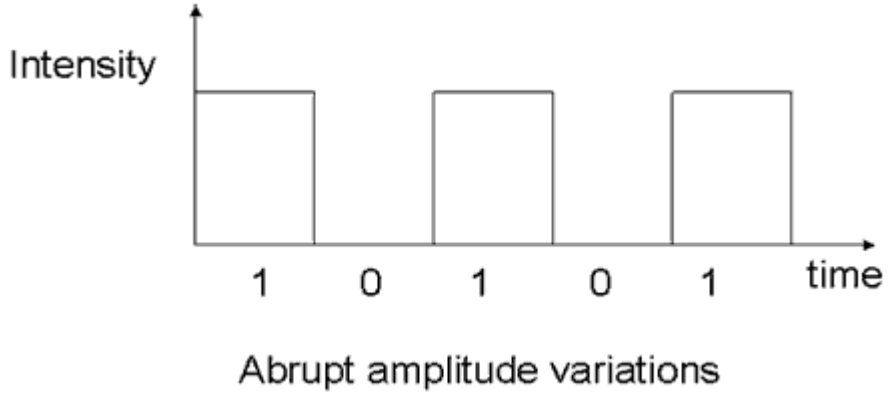
Digital and Analog Signals

- **Signals** carry information and are defined as any physical quantity that varies with time, space, or any other independent variable.
- For example, a sine wave whose amplitude varies with respect to time or the motion of a particle with respect to space can be considered as signals.
- A **System** can be defined as a physical device that performs an operation on a signal.
- For example, an amplifier is used to amplify the input signal amplitude. In this case, the amplifier performs some operation(s) on the signal, which has the effect of increasing the amplitude of the desired information-bearing signal.

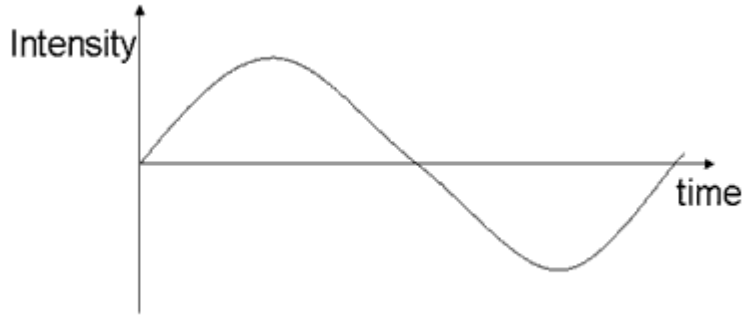
Signals can be categorized in various ways;

discrete and continuous time domains.

Discrete-time signals are defined only on a discrete set of times. Continuous-time signals are often referred to as continuous signals even when the signal functions are not continuous; an example is a square-wave signal.



Abrupt amplitude variations



Discrete-valued and Continuous-valued or Digital and Analog signals.

- Digital signals are discrete-valued and analog signals are continuous electrical signals that vary in time.
- Analog devices and systems process signals whose voltages or other quantities vary in a continuous manner. They can take on any value across a continuous range of voltage, current, or other metric.
- **Example,** the telephone transmitter converts the sounds into an electrical voltage signal. The intensity of the voice causes electric current variations. Therefore, the two are analogous hence the name analog. At the receiving end, the signal is reproduced in the same proportion. Hence the electric current is a model and is an electrical representation of one's voice.

- Digital signals are non-continuous, they change in individual steps. They consist of pulses or digits with discrete levels or values.
- The value of each pulse is constant, but there is an abrupt change from one digit to the next.
- Digital signals have two amplitude levels. The value of which are specified as one of two possibilities such as 1 or 0, HIGH or LOW , TRUE or FALSE and so on.
- A digital system is the one that handles only discrete values or signals. Any set that is restricted to a finite number of elements contains discrete information.
- The word digital describes any system based on discontinuous data or events. Digital is the method of storing, processing and transmitting information through the use of distinct electronic pulses that represent the binary digits 0 and 1. Examples of discrete sets are the 10 decimal digits, the 26 letters of the alphabet etc. A digital system would be to flick the light switch on and off. There's no 'in between' values.

Advantages of Digital Signals

Noise Margin (resistance to noise/robustness) :

Digital circuits are less affected by noise. If the noise is below a certain level (the noise margin), a digital circuit behaves as if there was no noise at all. The stream of bits can be reconstructed into a perfect replica of the original source.

However, if the noise exceeds this level, the digital circuit cannot give correct results.

Error Correction and Detection (Reliability) :

Digital signals can be regenerated to achieve lossless data transmission, within certain limits. Analog signal transmission and processing, by contrast, always introduces noise. Digital systems are highly reliable one of the reasons for that is use of error correction codes.

Easily Programmable :

Digital systems interface well with computers and are easy to control with software.

It is often possible to add new features to a digital system without changing hardware, and to do this remotely, just by uploading new software.

Design errors or bugs can be worked-around with a software upgrade, after the product is in customer hands.

A digital system is often preferred because of (re-)programmability and ease of upgrading without requiring hardware changes.

High speed

Digital processing of data ensures high speed of operation which is possible due to advances in Digital Signal Processing. Digital devices can produce 500 million results having operating speed of 2 ns.

Design is easy

The design of digital systems which require use of Boolean algebra and other digital techniques is easier compared to analog designing.

Flexibility

these are more flexible to design since their design involves a set of logical steps and have various discrete and integrated options available to the user.

Cheap Electronic Circuits

More digital circuitry can be fabricated per square millimeter of integrated-circuit material. Information storage can be much easier in digital systems than in analog ones.

In particular, the great noise-immunity of digital systems makes it possible to store data and retrieve it later without degradation.

In an analog system, aging and wear and tear will degrade the information in storage, but in a digital system, as long as the wear and tear is below a certain level, the information can be recovered perfectly.

Theoretically, there is no data-loss when copying digital data. This is a great advantage over analog systems, which faithfully reproduce every bit of noise that makes its way into the signal.

Disadvantages

The world in which we live is analog, and signals from this world such as light, temperature, sound, electrical conductivity, electric and magnetic fields, and phenomena such as the flow of time, are for most practical purposes continuous and thus analog quantities rather than discrete digital ones.

For a digital system to do useful things in the real world, translation from the continuous realm to the discrete digital realm must occur, resulting in quantization errors.

This problem can usually be mitigated by designing the system to store enough digital data to represent the signal to the desired degree of fidelity. The Nyquist-Shannon sampling theorem provides an important guideline as to how much digital data is needed to accurately portray a given analog signal.

- Digital systems can be fragile, in that if a single piece of digital data is lost or misinterpreted, the meaning of large blocks of related data can completely change.
- This problem can be diminished by designing the digital system for robustness. For example, a parity bit or other error-detecting or error-correcting code can be inserted into the signal path so that minor data corruptions can be detected and possibly corrected.

Digital circuits are made from analog components, and care has to be taken to

- all noise and timing margins,
- to parasitic inductances and capacitances,
- to proper filtering of power and ground connections,
- to electromagnetic coupling amongst data lines

Inattention to these can cause problems such as "glitches", pulses do not reach valid switching (threshold) voltages, or unexpected ("undecoded") combinations of logic states.

the fact that digital circuits are made from analog components is the fact that digital circuits are slower to perform calculations than analog circuits that occupy a similar amount of physical space and consume the same amount of power. However, the digital circuit will perform the calculation with much better repeatability, due to the high noise immunity of digital circuitry.

Basic Digital Devices

Logic Gates

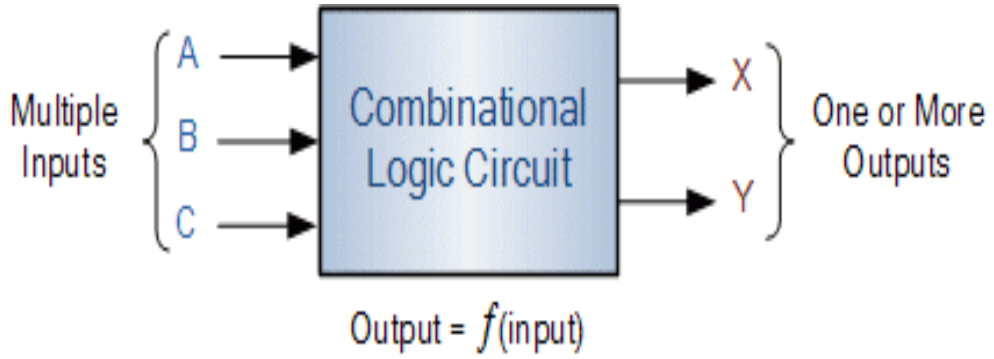
- Basic element that make up a digital system .
- An circuit which is able to operate on number of binary inputs in order to perform a particular logic function.
- NAND, NOR, AND, NOT, OR, Ex-OR, Ex-NOR
- Gate is a digital circuit that may have more than one I/P but only one output.
- BY connecting different gates we can build various circuits that can perform arithmetic and other functions.

Basic Digital Devices

Combinational Circuits

When logic gates are connected together to produce a specified output for certain specified combinations of input variables, with no storage involved

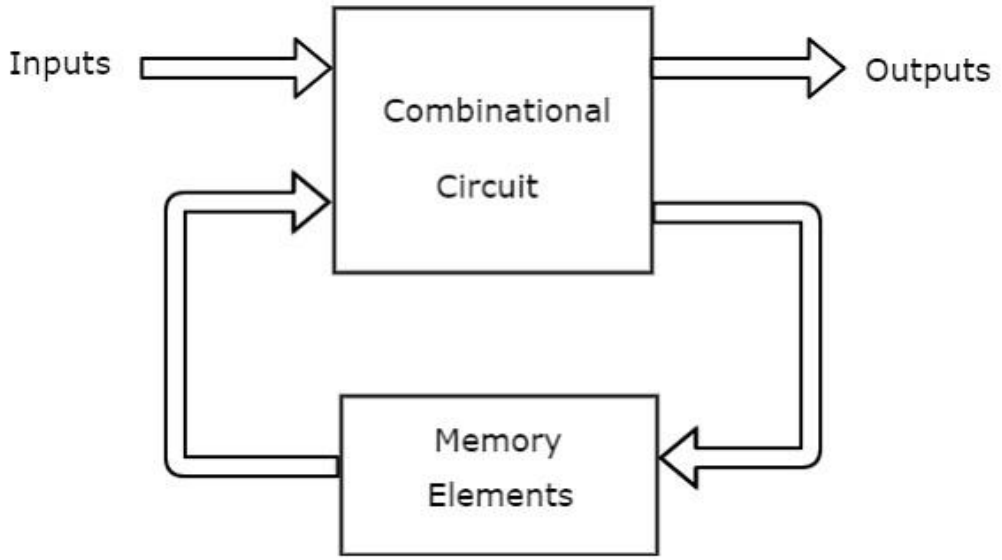
In such circuits output variables are all time dependent upon combination of input variables.



Sequential Circuits

- Applications in which digital outputs are required to be generated in accordance with the sequence in which the input signals are received.
- This requirement can be satisfied using sequential logic system.
- These applications require outputs to be generated that are not only dependent on the present input conditions but they also depend upon the past history of these inputs.
- Past history is provided by feedback from the output back to the input.

- Basic unit for this memory element is Flip Flop (FF).
- It is a device with two stable states i.e its output is either 0 (Logic 0) or +5V dc (logic 1).
- Flip Flop maintains its output state until directed by an input signal to change its state.
- Means can store 1-bit information.



Digital ICs

An Integrated circuit (IC) is a silicon semiconductor crystal, called a chip, containing the electronic components for constructing digital gates.
Various gates connected inside to form required circuit.
Logic gates are the basic building blocks of digital logics and circuits.

An integrated circuit, or IC, is small chip that can function as an amplifier, oscillator, timer, microprocessor, or even computer memory.

Need of Integrated Circuit (IC)

- The traditional method of combinational circuit design involves simplification and realization of logic function using gates.
- This method of designing combinational circuit is effective for small circuit.
- When complexity is more, more no of gates are required with more no of wires between them.
- Designing of such circuits may be time consuming and less reliable.
- To avoid such problems, a collection of one or more gates are fabricated on a single silicon chip. Such circuit is called Integrated Circuit.

Classification of ICs

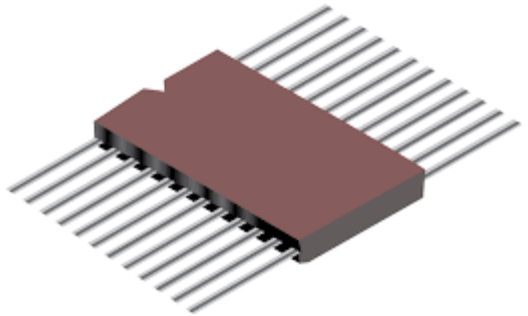
classification of **different types of ICs** basis on their chip size.

Small scale integration (SSI) :

3 – 30 gate circuits per chip. These include basic gate functions and flip flops.

These ICs are fabricated in one of two main configurations

a) Dual in line package b) Flat Pack



Medium scale integration (MSI):
It contains the equivalent of 30 – 300 gates per chip.
These ICs include the more complex logic functions such as encoders, decoders, counters registers, arithmetic circuits and small memories and others.

Large scale integration (LSI):
300 – 3,000 gates per chip.
It includes memories, microprocessors, programmable logic devices and customized devices.

Very large scale integration (VLSI) :

Beyond LSI.

Complexity is usually stated in terms of transistor count than gate count.

- Any IC with over 1,00,000 transistor is VLSI IC.
- It includes advanced microprocessors, large memories, larger programmable logic devices, and customized devices.
- It reduces system cost, size, and the no of external wire connections improving reliability of the system.

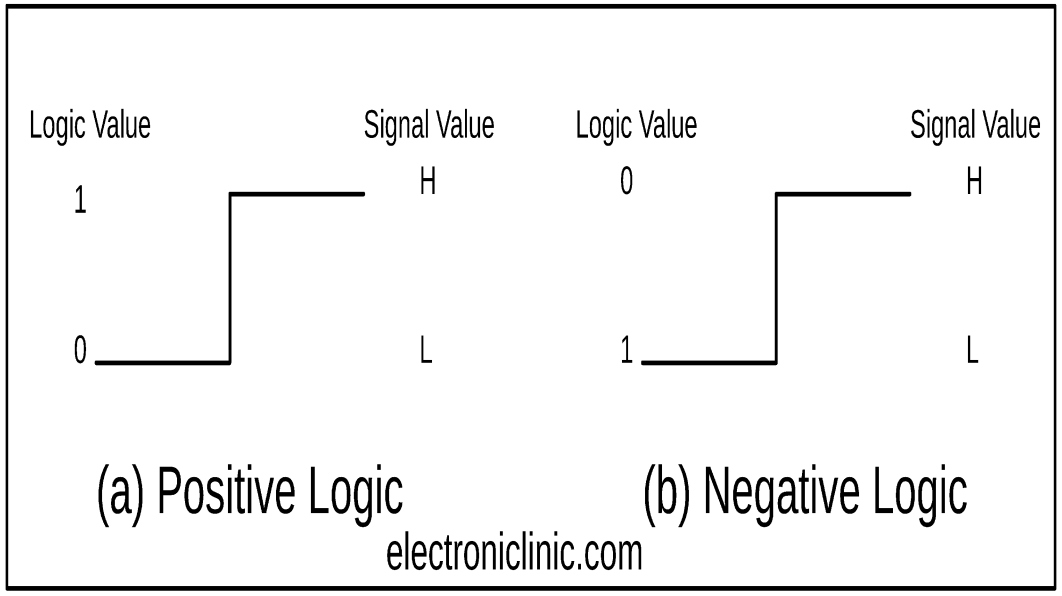
Positive and Negative Logic

GROW

+ve logic	-ve logic
high voltage correspond to logic '1'	high voltage correspond to logic '0'
'+5V' = LOGIC "1"	'+5V' = LOGIC "0"
'0V' = LOGIC "0"	'0V' = LOGIC "1"

EXAMPLE =

Logic 0 = -5V



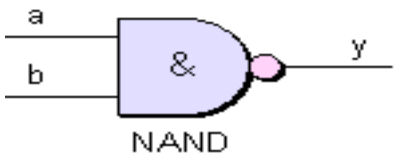
Physical Mapping

a	b	y
0V	0V	+ve
0V	+ve	+ve
+ve	0V	+ve
+ve	+ve	0V

0V = False
+ve = True
(positive logic)

a	b	y
F	F	T
F	T	T
T	F	T
T	T	F

a	b	y
0	0	1
0	1	1
1	0	1
1	1	0



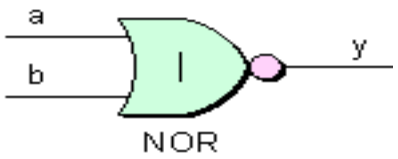
0V = True
+ve = False
(negative logic)

a	b	y
T	T	F
T	F	F
F	T	F
F	F	T

reorder

a	b	y
F	F	T
F	T	F
T	F	F
T	T	F

a	b	y
0	0	1
0	1	0
1	0	0
1	1	0



Number System

Problems:

1. Convert (475.25)8 to its decimal equivalent.
2. Convert (9 B2.1A)H to its decimal equivalent
3. Convert (3102.12)4 to its decimal equivalent.
4. Convert (614.15)7 to its decimal equivalent.
5. Convert (3509)10 to its hexadecimal equivalent
6. Convert (22.64)10 to hexadecimal number
7. Determine the value of base x if

a) (211)x= (152)8 b) (193)x= (623)8

Complements

Why Complements???

Representation of positive and negative no. using 1's complement (+5, -5)

Maximum Positive no. for n= 4 bit number

Maximum negative no.

Problem in 1's complement

Representation of positive and negative no. using 2's complement (+6, -6)

Maximum positive integers

Maximum negative integers

One zero

Express +25 and -25 in 8 bit sign magnitude, 1's complement and 2's complement form

Signed and Unsigned Binary Numbers

Unsigned Binary Numbers

- All the bits are used as magnitude only , No sign bit
- With 8 bit unsigned arithmetic, all the magnitudes must be restricted between 0 and (255)10 or (00)H to (FF)H
- Answer also must fall within this range.
- For the magnitude greater than 255 we have to use 16 bit arithmetic.

Overflow:

If result is greater than 255 it is called overflow

Sign magnitude numbers

MSB of binary number is used to represent the sign and the remaining bits are used for representing the magnitude.

0	1	0	1	1	0	0	1

= +89

Range of signed numbers

Largest negative no. (-127)

Largest positive no. (+127)

With the sign magnitude numbers, 8 bit arithmetic can be used as long as the input data range falls in decimal -127 to +127.

With 16 bit numbers, the range of sign magnitude numbers extends from decimal (-32,767) to (+32,767).

**These nos. have limited use because they require complicated circuits. These are often used in A to D converters

Signed Complement numbers

Represent -7 in signed number system (using 8 binary bit)

1. Signed- Magnitude Representation

1	0	0	0	0	1	1	1

2. Signed- 1's complement Representation

1	1	1	1	1	0	0	0

3. Signed- 2's complement Representation

1	1	1	1	1	0	0	1

In computer to represent negative numbers complement system is used. 2's complement is most popular method.

Signed Binary Numbers

Table 1.3
Signed Binary Numbers

Decimal	Signed-2's Complement	Signed-1's Complement	Signed Magnitude
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
−0	—	1111	1000
−1	1111	1110	1001
−2	1110	1101	1010
−3	1101	1100	1011
−4	1100	1011	1100
−5	1011	1010	1101
−6	1010	1001	1110
−7	1001	1000	1111
−8	1000	—	—

Few Points:

- Positive no.s in all 3 representations are same and having 0 in the leftmost position.
- All negative nos. have a 1 in the leftmost bit position
- The signed 2's complement system has only one representation for 0, which is always positive

Addition of Signed numbers

Case 1: Addition of both positive numbers

Case 2: Smaller no is negative

Case 3: Bigger number negative

Case 4: Both no. negative

- If carry: discard it. Look for MSB
- If MSB is 1 no. is negative so take 2's complement of the no. and it will be final answer.
- If MSB is 0: no. is positive so take the answer as it is.

Example

- +7 +12
- -7 + 12
- +7 -12
- -7-12

Binary subtraction using 1's and 2's complements

Subtraction using 1's complements (A-B)

Steps:

1) Convert number B into its 1's complement
2) Add A and 1's complement of B
3) If final carry is 1 then add it to result of addition. It is final answer in true form
4) If carry is 0 then result obtained is negative and in it's 1's complement form and needs to be converted.

Subtraction using 2's complements (A-B)

Steps:

1) Convert number B into its 2's complement
2) Add A and 2's complement of B
3) If final carry is 1 then final answer is positive. Carry is to be discarded
4) If carry is 0 then result obtained is negative and in it's 2's complement form and needs to be converted.

Examples

Subtraction using 1’s complement

1011 – 0100

0100 – 1001

Subtraction using 2’s complement

1001 – 0101

0011 - 0110

Binary Codes

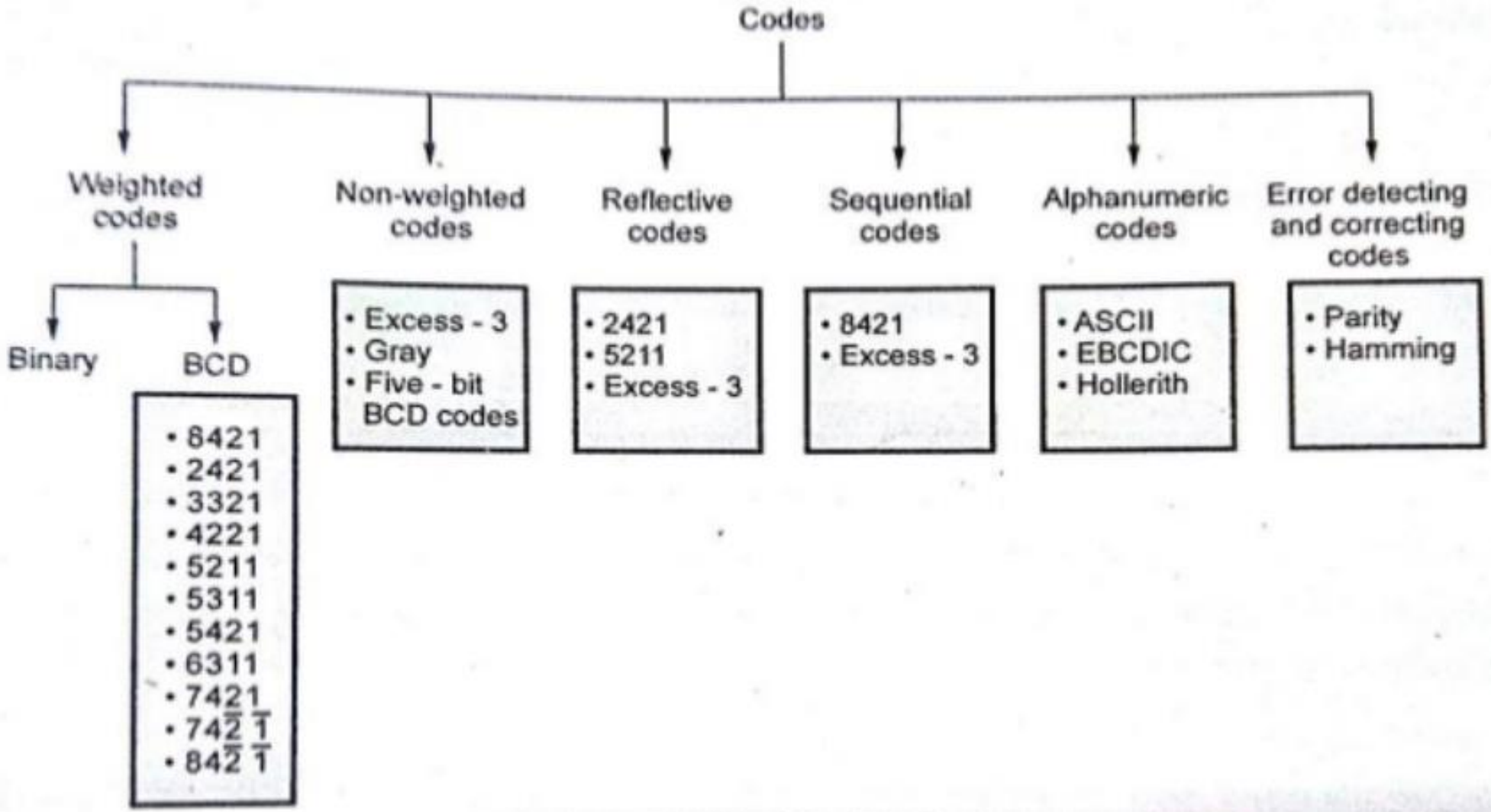
- When numbers, alphabets or words are represented by a specific group of symbols i.e. they are encoded
- The group of symbols used to encode them are called codes
- Digital data is represented, stored and transmitted as groups of binary digits (bits).

Binary Codes

Classification of Binary Codes

- Weighted codes
- Non-weighted codes
- Reflective codes
- Sequential codes
- Alphanumeric codes
- Error Detecting and Correcting codes

CLASSIFICATION OF BINARY CODES



Weighted codes

These codes obey the positional weight principle

Each position of number represents a specific weight.

In this code each decimal digit is represented by a 4-bit binary number.

BCD is a way to express each of the decimal digits with a binary code. In the BCD, with four bits we can represent sixteen numbers (0000 to 1111).

But in BCD code only first ten of these are used (0000 to 1001). The remaining six code combinations i.e. 1010 to 1111 are invalid in BCD.

Packed And Unpacked BCD Numbers

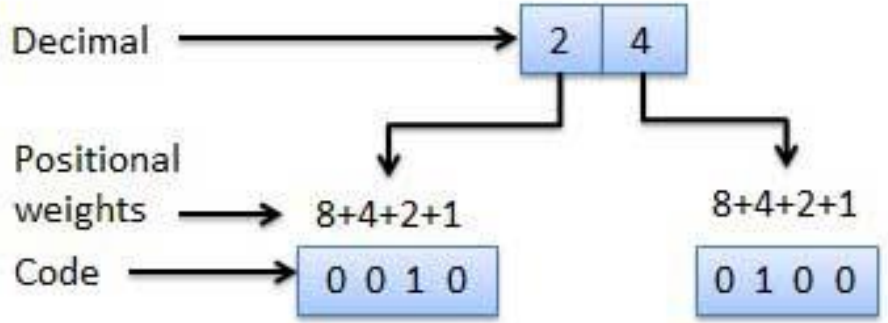
In the case of unpacked BCD numbers,

- each four-bit BCD group corresponding to a decimal digit is stored in a separate register inside the machine. In such a case, if the registers are eight bits or wider, the register space is wasted.

In the case of packed BCD numbers,

- two BCD digits are stored in a single eight-bit register. The process of combining two BCD digits so that they are stored in one eight-bit register involves shifting the number in the upper register to the left 4 times and then adding the numbers in the upper and lower registers. The process is illustrated by showing the storage of decimal digits ‘5’ and ‘7’:

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



Decimal digit 5 is initially stored in the eight-bit register as: 0000 0101.
Decimal digit 7 is initially stored in the eight-bit register as: 0000 0111.

- After shifting to the left 4 times, the digit 5 register reads: 0101 0000.
- The addition of the contents of the digit 5 and digit 7 registers now reads: 0101 0111.

Advantages of BCD Codes

- It is very similar to decimal system.
- We need to remember binary equivalent of decimal numbers 0 to 9 only.

Disadvantages of BCD Codes

- The addition and subtraction of BCD have different rules.
- The BCD arithmetic is little more complicated.
- BCD needs more number of bits than binary to represent the decimal number. So BCD is less efficient than binary.

Decimal	8421 BCD	2421 BCD
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	1011
6	0110	1100
7	0111	1101
8	1000	1110
9	1001	1111

Weighted codes

	BCD-8421	BCD-5421	BCD-4221	BCD-5211
0	0000	0000	0000	0000
1	0001	0001	0001	0001 | 0010
2	0010	0010	0010 | 0100	0100 | 0011
3	0011	0011	0011 | 0101	0101 | 0110
4	0100	0100	1000 | 0110	0111
5	0101	1000	1001 | 0111	1000
6	0110	1001	1010 | 1100	1001 | 1010
7	0111	1010	1011 | 1101	1100 | 1011
8	1000	1011	1110	1110 | 1101
9	1001	1100	1111	1111

BCD addition

Add two numbers as same as binary addition

Case 1: If the result is less than or equals to 9 and carry is zero then it is valid BCD.

Case 2: If result is greater than 9 and carry is zero then add 6 in four bit combination.

Case 3: If result is less than or equals to 9 but carry is 1 then add 6 in four bit combination.

Carry from →	0		0	
previous digit	7		0111	
+	6	+	0110	
	13		1101	
			110	← Add 6 because 1101 is
Carry to next digit →		1	0011	not a valid BCD digit
Carry from →	1		1	
previous digit	9		1001	
	9	+	1001	
	19	1	0011	
			110	← Add 6 because of carry
Carry to next digit →		1	1001	to next digit

BCD Arithmetic

Example:		1	carry
	59	0101 1001	
+	39	0011 1001	

		1001 0010	
		0110	

		1001 1000	(98)

BCD	1	1		
	0001	1000	0100	184
	+0101	0111	0110	+576
Binary sum	0111	10000	1010	
Add 6		0110	0110	
BCD sum	0111	0110	0000	760

BCD Subtraction

	Regular Subtraction	9's Complement Subtraction	
(a)	8	8	
	− 2	+ 7	9's complement of 2
	6	(1) 5	
		+ 1	Add carry to result
		6	
(b)	28	28	
	− 13	+ 86	9' complement of 13
	15	(1) 14	
		+ 1	Add carry to the result
		15	
(c)	18	18	
	− 24	+ 75	9' complement of 24
	−6	93	9's complement of result (No carry indicates that the answer is negative and in complement form)
		− 06	

E.G. $8 - 3 = 8 + [9\text{'s COMP. OF } 3]$
$= 8 + 6$

```
  1000
  0110
  1110    ← INVALID (>9)
  0110    ← CORRECTION
(1) 0100
  └──→ 1  ← END AROUND CARRY
  0101 = 5
```

(b) $3 - 8 = -5$

```
  0011
  0001
  0100
```

NO CARRY >>> NEGATIVE
9's COMP. OF 0100 = **0101** = **-5**

(c) $87 - 39$ >>> $87 + [9\text{'s COMP OF } 39]$

```
  8 7     1000    0111
  6 0     0110    0000
        ──────────────
          1110    0111
INVALID   0110
        ──────
        (1) 0100  0111
          └──────────→ 1
        ──────────────
          0100    1000
    =       4       8
```

Signed Magnitude BCD Number

Similar to representation of signed numbers in binary.

Sign is represented in 4 bits

Plus sign is denoted by four zeros

Minus sign with BCD equivalent of 9.

+52 0000 0101 0010

-52 1001 0101 0010

When smaller number is to be subtracted from larger one

Regular subtraction

```
  678
- 234
-----
  444
```

Subtraction using 9's complement

```
  678
+765  ← (9's complement of 234)
-----
(1)443
  + 1
-----
  444
```

When larger number is to be subtracted from smaller one

Regular subtraction

```
  228
- 485
-----
- 257
```

Subtraction using 9's complement

```
  228
+514  ← (9's complement of 485)
-----
  742  (No carry indicates –ve value)
   ↓
- 257  (9's complement of result)
```

Signed Complement BCD Numbers

- 9’s Complement
- 10’s Complement

- +240 0000 0010 0100 0000

- -240 9240 (sign Magnitude)
  9 (999-240) = 9759 (signed 9’s Complement)
  9 760 (signed 10’s Complement)

Example

Add (+370) and (-240)

+370 = 0 370

-240 = 9 760 (10’s complement)

0370 + 9760 = 1 0130 (Discard Carry, Answer in true form)

Non-weighted code:-

In non-weighted code, there is no positional weight i.e. each position within the binary number is not assigned a prefixed value. No specific weights are assigned to bit position in non –weighted code.

The non-weighted codes are:-

a) The Gray code

b) The Excess-3 code

The Gray code:-

It is non weighted code in which each number differs from previous number by a single bit.

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

THE BCD TO EXCESS 3 CODE CONVERTER

- BCD Excess-3 circuit will convert numbers from their binary representation to their excess-3 representation. Hence our truth table is as below:

B3	B2	B1	B0		E3	E2	E1	E0
0	0	0	0		0	0	1	1
0	0	0	1		0	1	0	0
0	0	1	0		0	1	0	1
0	0	1	1		0	1	1	0
0	1	0	0		0	1	1	1
0	1	0	1		1	0	0	0
0	1	1	0		1	0	0	1
0	1	1	1		1	0	1	0
1	0	0	0		1	0	1	1
1	0	0	1		1	1	0	0

Reflective Codes

A code is said to be reflective, when the code for 9 is the complement for 0. Code for 8 is complement for 1, 7 for 2, 6 for 3, and 5 for 4.

2421 , 5211 and Excess-3 are Reflective codes

Sequential Codes

A code is said to be sequential when each succeeding code is one binary number greater than the preceding code.

This greatly simplifies the mathematical manipulations. 8421 and XS-3 are sequential codes

Alphanumeric Codes

Designed to represent the numbers as well as alphabetic characters.

- ASCII (American Standard code for Information Interchange)
- EBCDIC (Extended Binary Coded Decimal Interchange code)
- Hollerith Codes

TABLE American Standard Code for Information Interchange (ASCII)

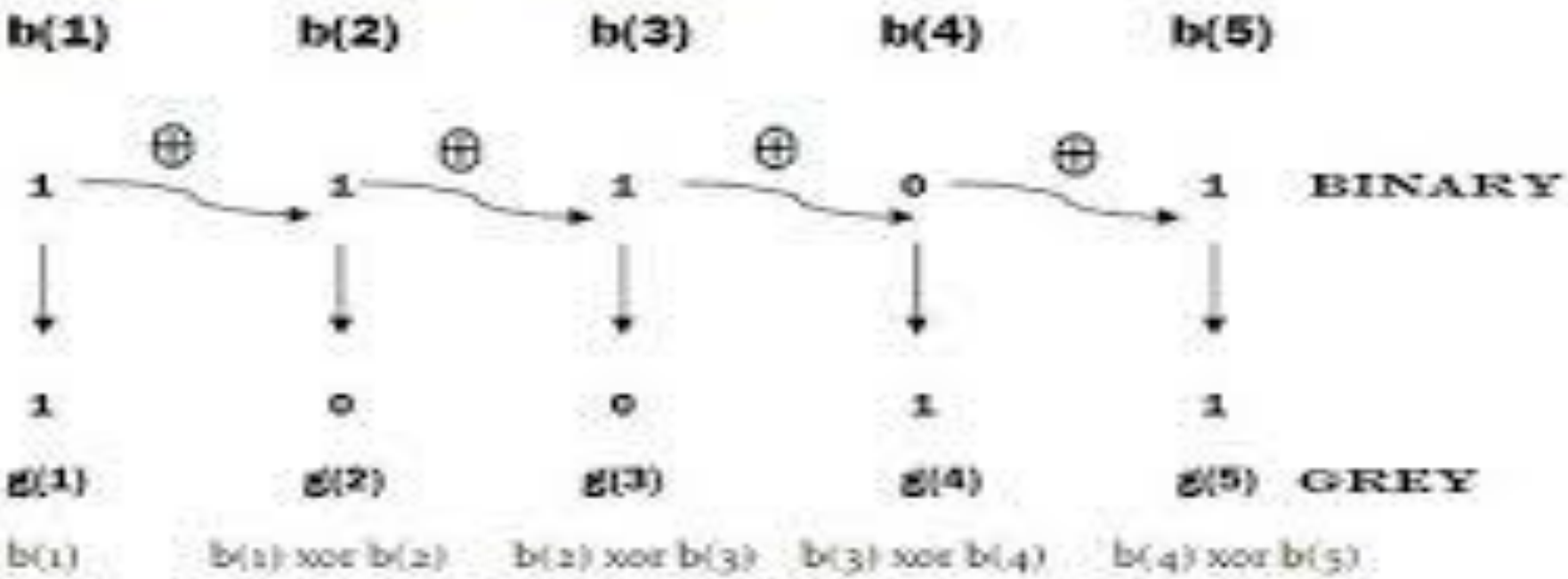
Character	Binary code	Character	Binary code
A	100 0001	0	011 0000
B	100 0010	1	011 0001
C	100 0011	2	011 0010
D	100 0100	3	011 0011
E	100 0101	4	011 0100
F	100 0110	5	011 0101
G	100 0111	6	011 0110
H	100 1000	7	011 0111
I	100 1001	8	011 1000
J	100 1010	9	011 1001
K	100 1011		
L	100 1100		
M	100 1101	space	010 0000
N	100 1110	.	010 1110
O	100 1111	(	010 1000
P	101 0000	+	010 1011
Q	101 0001	$	010 0100
R	101 0010	*	010 1010
S	101 0011	)	010 1001
T	101 0100	−	010 1101
U	101 0101	/	010 1111
V	101 0110	,	010 1100
W	101 0111	=	011 1101
X	101 1000		
Y	101 1001		
Z	101 1010		

Binary to Gray & Gray to Binary

Decimal Number	4 bit Binary Number ABCD	4 bit Gray Code $G_1G_2G_3G_4$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

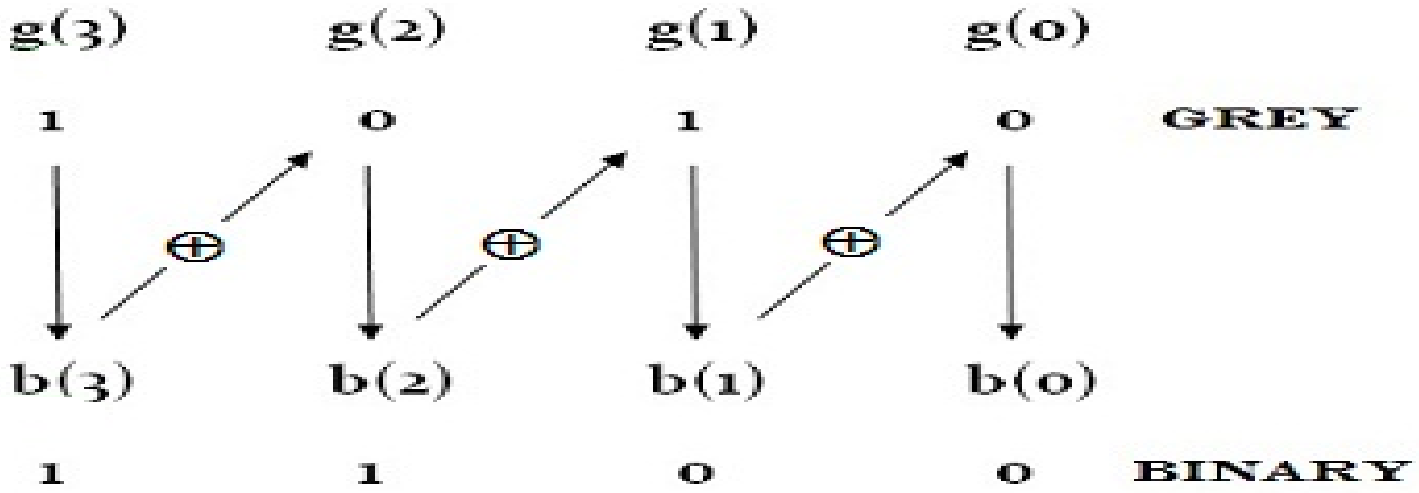
Binary to Grey Code Conversion

Convert the binary 11101_2 to its equivalent Grey code



Grey Code to Binary Conversion

Convert the Grey code 1010 to its equivalent Binary



i.e $b(3) = g(3)$

$b(2) = b(3) \oplus g(2)$

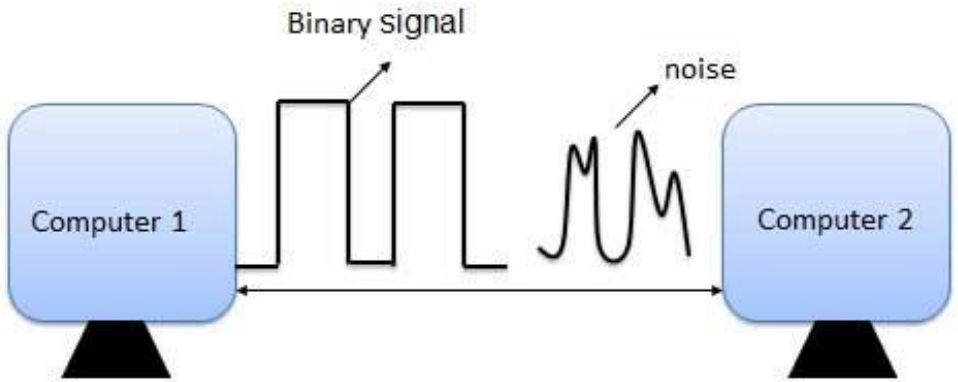
$b(1) = b(2) \oplus g(1)$

$b(0) = b(1) \oplus g(0)$

Excess-3 Arithmetic

Error Detecting and Correcting Codes

Data can be corrupted in transmission or storage by a variety of undesirable phenomenon, such as radio interference, electrical noise, power surges, bad spots on disks or tapes, or scratches or dirt on CD or DVD media. It is useful to have a way to to detect (and sometimes correct) such data corruption.



Errors come in several forms. The most common situation is that a bit in a stream of data gets flipped (a 0 becomes a 1 or a 1 becomes a 0). It is also possible for a bit to get deleted, or for an extra bit to be inserted. In some situations, burst errors occur, where several successive bits are affected.

Error-Detecting codes

Whenever a message is transmitted, it may get scrambled by noise or data may get corrupted. To avoid this, we use error-detecting codes which are additional data added to a given digital message to help us detect if an error occurred during transmission of the message. A simple example of error-detecting code is **parity check**.

Error-Correcting codes

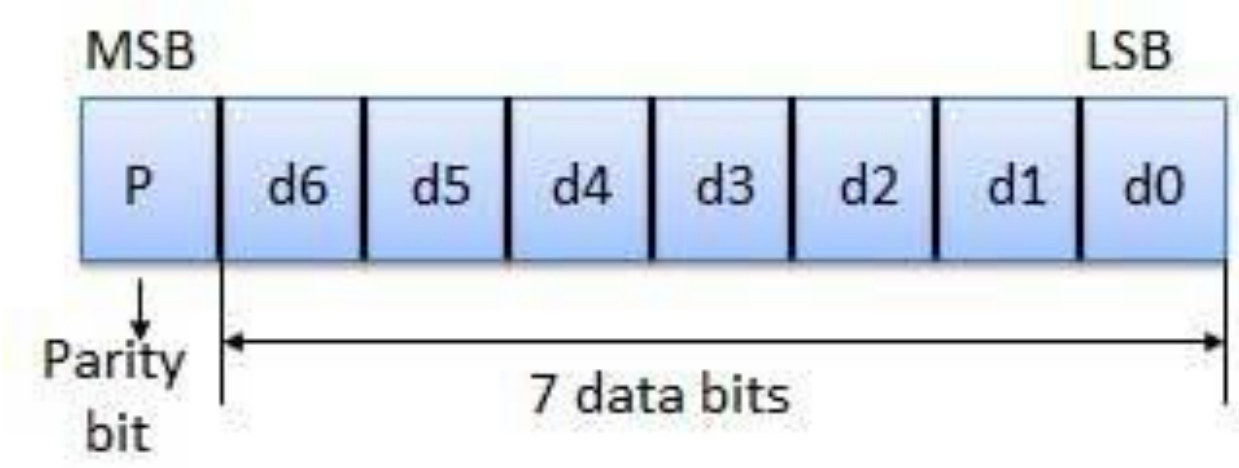
- Along with error-detecting code, we can also pass some data to figure out the original message from the corrupt message that we received.
- This type of code is called an error-correcting code. Error-correcting codes also deploy the same strategy as error-detecting codes but additionally, such codes also detect the exact location of the corrupt bit.
- In error-correcting codes, parity check has a simple way to detect errors along with a sophisticated mechanism to determine the corrupt bit location. Once the corrupt bit is located, its value is reverted (from 0 to 1 or 1 to 0) to get the original message.

How to Detect and Correct Errors?

- To detect and correct the errors, additional bits are added to the data bits at the time of transmission.
- The additional bits are called **parity bits**. They allow detection or correction of the errors.
- The data bits along with the parity bits form a **code word**.

Parity Checking of Error Detection

It is the simplest technique for detecting and correcting errors. The MSB of an 8-bits word is used as the parity bit and the remaining 7 bits are used as data or message bits. The parity of 8-bits transmitted word can be either even parity or odd parity.



Parity bit

We can detect single errors with a **parity bit**.

The parity bit is computed as the exclusive-OR or exclusive-NOR of all of the other bits in the word.

Thus, the resulting word with a parity bit will always have an even (for even parity) or odd (for odd parity) number of 1 bits in it.

If a single bit is flipped in transmission or storage, the received data will have the wrong parity, so we will know something bad has happened.

Even parity -- Even parity means the number of 1's in the given word including the parity bit should be even (2,4,6,....).

Odd parity -- Odd parity means the number of 1's in the given word including the parity bit should be odd (1,3,5,....).

Use of Parity Bit

The parity bit can be set to 0 and 1 depending on the type of the parity required.

•For even parity, this bit is set to 1 or 0 such that the no. of "1 bits" in the entire word is even. Shown in fig. (a).

•For odd parity, this bit is set to 1 or 0 such that the no. of "1 bits" in the entire word is odd. Shown in fig. (b).

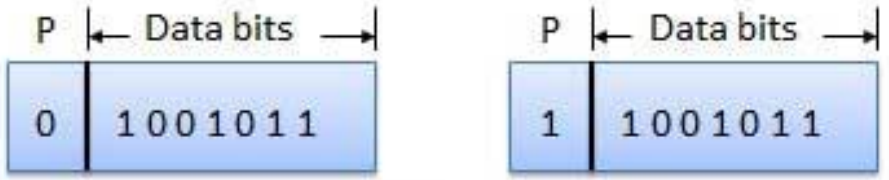


Fig. (a)

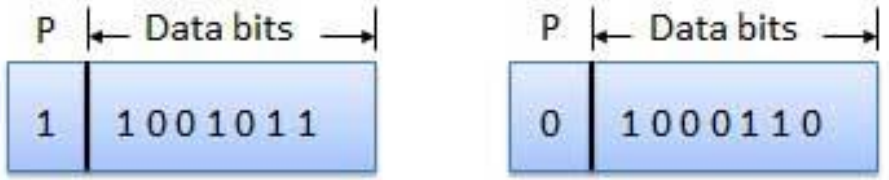


Fig. (b)

How Does Error Detection Take Place?

Parity checking at the receiver can detect the presence of an error, if the parity of the receiver signal is different from the expected parity.
That means, if it is known that the parity of the transmitted signal is always going to be "even" and if the received signal has an odd parity, then the receiver can conclude that the received signal is not correct.

- If an error is detected, then the receiver will ignore the received byte and request for retransmission of the same byte to the transmitter.

3 bit data			Message with even parity		Message with odd parity	
A	B	C	Message	Parity	Message	Parity
0	0	0	000	0	000	1
0	0	1	001	1	001	0
0	1	0	010	1	010	0
0	1	1	011	0	011	1
1	0	0	100	1	100	0
1	0	1	101	0	101	1
1	1	0	110	0	110	1
1	1	1	111	1	111	0

Error-correcting codes

- Error-correcting codes (ECC) are a sequence of numbers generated by specific algorithms for detecting and removing errors in data that has been transmitted over noisy channels.
- Error correcting codes ascertain the exact number of bits that has been corrupted and the location of the corrupted bits, within the limitations in algorithm.

ECCs can be broadly categorized into two types −

- **Block codes** − The message is divided into fixed-sized blocks of bits, to which redundant bits are added for error detection or correction.
- **Convolutional codes** − The message comprises of data streams of arbitrary length and parity symbols are generated by the sliding application of a Boolean function to the data stream.

Hamming Code

- Hamming code is a block code that is capable of detecting up to two simultaneous bit errors and correcting single-bit errors. It was developed by R.W. Hamming for error correction.
- In this coding method, the source encodes the message by inserting redundant bits within the message.
- These redundant bits are extra bits that are generated and inserted at specific positions in the message itself to enable error detection and correction.
- When the destination receives this message, it performs recalculations to detect errors and find the bit position that has error.

Encoding a message by Hamming Code

The procedure used by the sender to encode the message encompasses the following steps −

Step 1 − Calculation of the number of redundant (Parity) bits.

Step 2 − Positioning the redundant bits.

Step 3 − Calculating the values of each redundant bit.

Once the redundant bits are embedded within the message, this is sent to the user.

Calculation of the number of redundant (Parity) bits

Number of parity bits depends on the no of information bits.

If the no of information bits is designated as D, then the number of parity bits P is determined by the following relations:

$$2^P \geq m + P + 1$$

Example: if we have 4 information bits, D=4, then P is found by trial and error using eqn. Let P=2 then 2^2 = 4

D+P+1= 4+2+1 = 7

Equation not satisfied so try for P=3, 2^3=8

D+P+1= 4+3+1 = 8

Condition satisfied. Hence we will choose 3 parity bits.

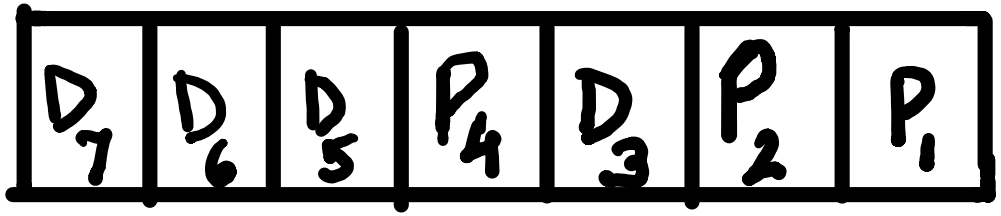
Step 2 – Positioning the redundant bits

- For 4 data bits 3 parity bits required hence codeword length is 7.
- The right most bit is designated as 1, next bit 2 and so on…

bit 7,bit 6, bit 5, bit 4, bit3, bit2, bit1

The parity bits are located in the positions that are numbered corresponding to ascending powers of two (1,2,4,8,16….)

So for 7 bit code, locations for parity bits and information bits are as under:



Assigning values to Parity Bit

Bit designation Bit location Binary location no		D15 15 1111	D14 14 1110	D13 13 1101	D12 12 1100	D11 11 1011	D10 10 1010	D9 9 1001	P8 8 1000	D7 7 0111	D6 6 0110	D5 5 0101	P4 4 0100	D3 3 0011	P2 2 0010	P1 1 0001
Information bits																
Parity Bits																

Assignment of P1: It has a 1 for its right most digits. This parity bit checks all bit locations. Including itself, that have 1s in the same location in the binary numbers. So p1 checks bit locations **1,3,5,7,9,11,13,15**

Assignment of P2: This parity bit checks all bit locations. Including itself, that have 1s in the 2 position in the binary numbers. So p2 checks bit locations **2,3,6,7,10,11,14,15**

Assignment of P4: This parity bit checks all bit locations. Including itself, that have 1s in the 3 position in the binary numbers. So p4 checks bit locations **4,5,6,7,12,13,14,15**

Assignment of P8: This parity bit checks all bit locations. Including itself, that have 1s in the 4 position in the binary numbers. So p8 checks bit locations **8,9,10,11,12,13,14,15**

Example

Encode the binary word 1011 into 7 bit even parity Hamming code.

Let p=3, 2^3 = 8

P+D+1= 3+4+1=8

So 3 parity bits are sufficient.

So total code bits=7

Bit designation **Bit location** **Binary location no**	**D7** **7** **0111**	**D6** **6** **0110**	**D5** **5** **0101**	**P4** **4** **0100**	**D3** **3** **0011**	**P2** **2** **0010**	**P1** **1** **0001**
Information bits	1	0	1		1		
Parity Bits				**0**		**0**	**1**

P1: 1,3,5,7……. P1 1 1 1, For even parity P1 should be 1
P2: 2,3,6,7……. P2 1 0 1, For even parity P2 should be 0
P3: **4,5,6,7……. P4 1 0 1, For even parity P4 should be 0**

To Solve

Determine the single error correcting code for the information code 10111 for odd parity

Detecting and Correcting an error

Assume that the even parity Hamming code in 0110011 is transmitted And that **0100011** is received. The receiver does not know what was transmitted.

Determine bit location where error has occurred using received code

Bit designation	**D7**	**D6**	**D5**	**P4**	**D3**	**P2**	**P1**
Bit Location	7	6	5	4	3	2	1
Binary Location No.	111	110	101	100	011	010	001
Received code	0	1	0	0	0	1	1

P1: 1,3,5, and 7 1 0 0 0 for even parity it should have even no of 1s

Hence parity checks for even parity is wrong …..i.e. 1(LSB)

P2: 2,3,6 and 7 1 0 1 0

Hence parity checks for even parity is correct …..i.e. 0

P4: 4,5,6 and 7 0 0 1 0

Hence parity checks for even parity is wrong …..i.e. 1

Resultant word is 1 0 1 means bit in the no 5 location in error

0100011 so corrected code will be **0110011**

To Solve

- The Hamming code is 101101101 is received. Correct if any errors.
- There are 4 parity bits and odd parity is used.

Pros & Cons

Advantages of Hamming code

- Hamming code method is effective on networks where the data streams are given for the single-bit errors.
- Hamming code not only provides the detection of a bit error but also helps you to indent bit containing error so that it can be corrected.
- The ease of use of hamming codes makes it best them suitable for use in computer memory and single-error correction.

Disadvantages of Hamming code

- Single-bit error detection and correction code. However, if multiple bits are founded error, then the outcome may result in another bit which should be correct to be changed. This can cause the data to be further errored.
- Hamming code algorithm can solve only single bits issues.