Unit 4. Design Examples

Page No.
Date

Arithmetic Circuits

Definition – Circuits designed to perform basic arithmetic operations such as addition, subtraction multiplication and division.

Types of Arithmetic circuits

- Half Adder
- Full Adder
- Subtractor

The working principle of these circuits is that they use logic gates (XOR, AND, OR) to generate sum and carry.

Applications –
Used in ALUs, processors and calculators.

1) Half Adder –

- It is a basic combinational logic circuit used for the addition of two single-bit binary numbers
- It has 2 inputs (A and B) and two outputs (Sum and carry)
- The name Half Adder comes from the fact that it cannot handle carry input from a previous stage.
- When adding two binary digits,
Sum (S) → Result of XOR operation between A and B
Carry (C) → Result of AND operation between A and B.

This is because

$0 + 0 = 0$ (sum = 0, carry = 0)

$0 + 1 = 1$ (sum = 1, carry = 0)

$1 + 0 = 1$ (sum = 1, carry = 0)

$1 + 1 = 10$ (sum = 0, carry = 1)

Truth table

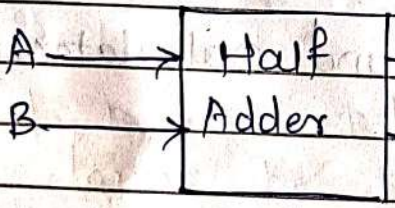
input A	Input B	Sum(S)	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Logic expression

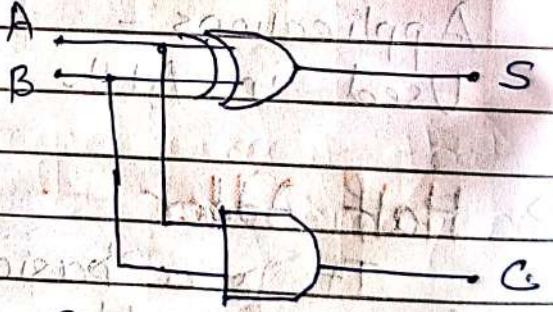
Sum (S) $\Rightarrow S = A \oplus B$ (XOR gate)

Carry (C) $\Rightarrow C = A \cdot B$ (AND gate)

1.



Block diagram



Circuit diagram

Limitations of Half Adder

1) It can't add more than one bit at a time.
2) It doesn't consider carry input from previous addition.
3) Only suitable for basic single-bit binary addition.

Applications

1) digital calculator for basic addition
2) building block of Full adder
3) digital measurement devices.

Full Adder

A Full Adder is a combinational logic circuit that performs the addition of three binary bits.

- It has two significant inputs (A and B)
- One carry-in (Cin) from a previous addition
- It produces two outputs Sum (S) and Carry out (Cout)

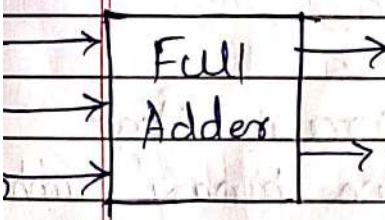
It adds three inputs (A, B, Cin)

Sum (S) = Result of XORing all three inputs,

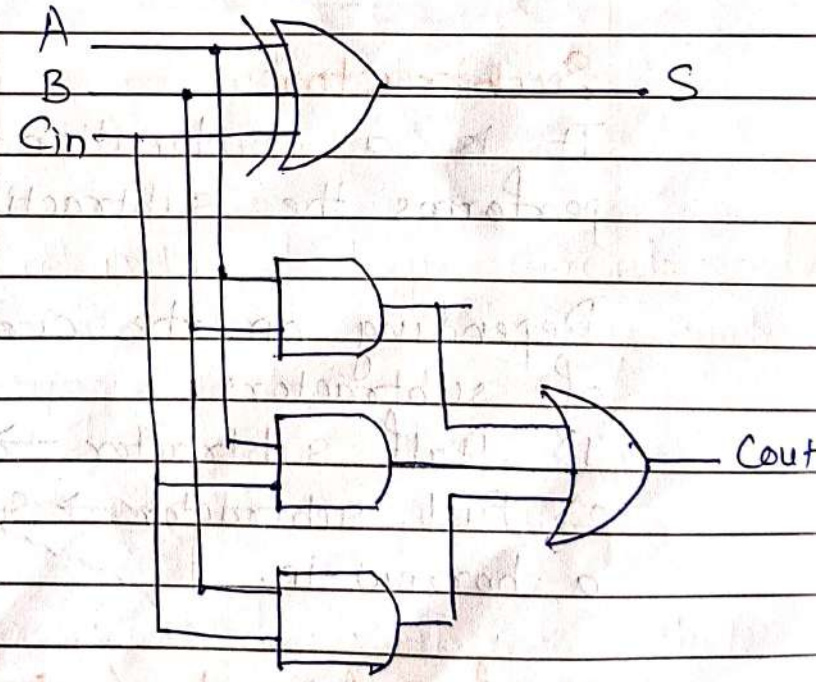
$$S = A \oplus B \oplus Cin$$

Carry-out ⇒ (Cout) ⇒ generated when at least two inputs are 1.

$$Cout = (A \cdot B) + (B \cdot Cin) + (A \cdot Cin)$$



Block diagram



Circuit diagram

Truth Table

A	B	Cin	Sum (S)	Carry-out (Cout)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Applications

- Arithmetic Logic Units (ALU's) in processors
- Binary calculators
- Digital counters
- Multipliers and dividers
- Used in memory addressing and data processing u

Subtractor

It is a combinational logic circuit that performs the subtraction of two binary numbe

Depending on the circuit, there are two types of subtractors.

1) Half subtractor → Subtracts two bits (no borro
2) Full subtractor → Subtracts 2 bits considering a borrow-in.

Half Subtractor circuit

- A circuit that subtracts two binary digits A-B.
- It produces two outputs Difference (D), Borrow (Bo)

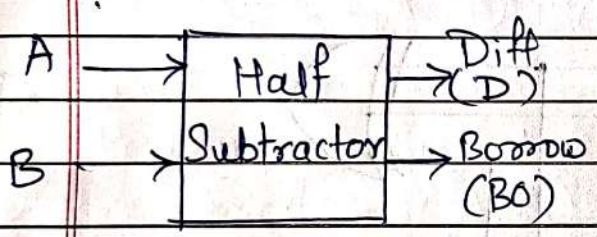
Truth table

A	B	Difference (D)	Borrow (BO)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

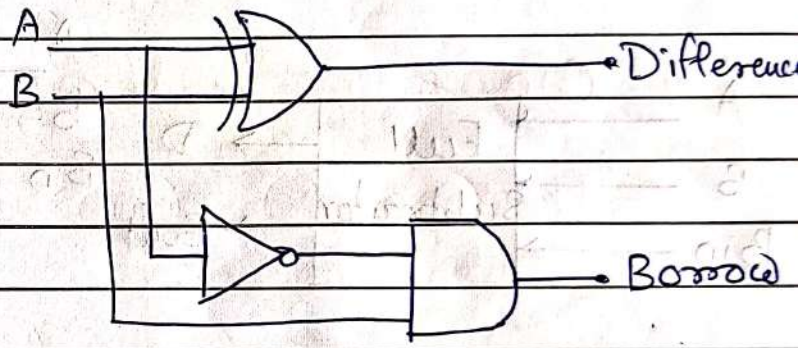
Logical expressions

Difference $D = A \oplus B$

$BO = \bar{A} \cdot B$



Block diagram



Circuit diagram

Full Subtractor Circuit

A circuit that subtracts two bits with a borrow-in (Bin)

It has 3 input A, B, Bin

It gives two inputs Difference (D), Borrow out (Bout)

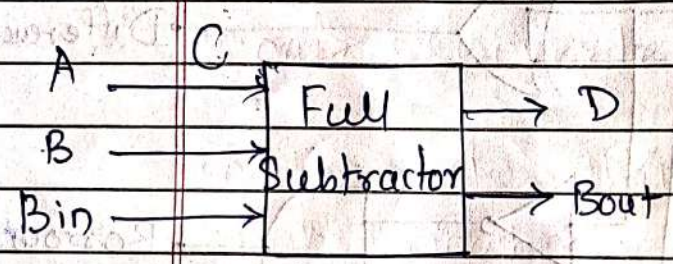
Logic expressions

$D = A \oplus B \oplus Bin$

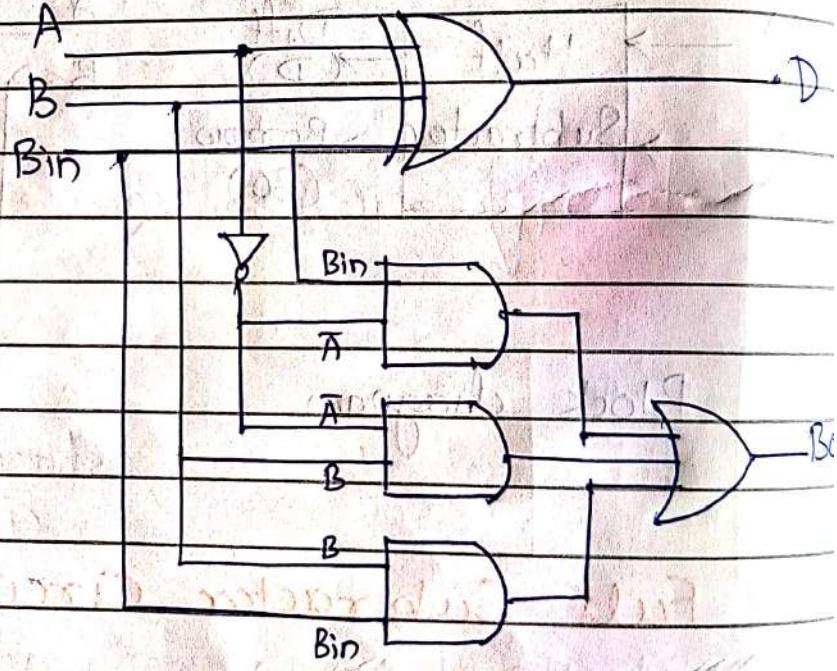
$Bout = (\bar{A} \cdot B) + (\bar{A} \cdot Bin) + (B \cdot Bin)$

Truth table of Full subtractor

A	B	Bin	Difference (D)	Borrow out (Bout)
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



Block diagram



Circuit diagram

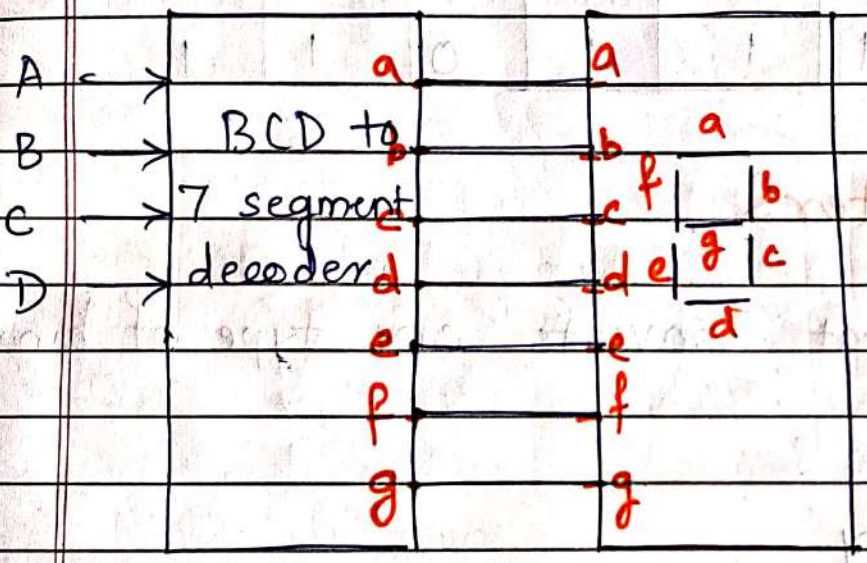
Application of Subtractors

- Digital Arithmetic operations
- Calculators
- Counters and timers
- Used in Comparators
- Microprocessor for subtraction instructions.

BCD to 7 segment Decoder

A circuit that converts Binary coded decimal (BCD) input (4 bits) into signals to drive a 7-segment display, to show a specific decimal digit.

It's inputs : 4 bits (0000 to 1001 for 0 - 9)
It has output 7 signals (a-g) for the LED display.



7 segment
LED display

These decoders are used in digital electronics to interface binary data with visual display,
eg. TTL 74LS47 (Common anode displays)
74LS48 (Common Cathode displays)

Applications
- Digital Clock and timers
- Digital voltmeters and other measurement instrument
- Calculators & electronic meters.

Truth table for BCD to 7 seg display

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Code Converters

The circuits that converts one type of binary code to another.

1) BCD to Binary
2) Binary to Gray code
3) Excess 3 to BCD

1) BCD to Binary Code converter

It takes a number written in BCD format and convert it into pure binary representation.

Truth table

BCD (D_3 D_2 D_1 D_0)	Decimal	Binary (B_3 B_2 B_1 B_0)
0 0 0 0	0	0 0 0 0
0 0 0 1	1	0 0 0 1
0 0 1 0	2	0 0 1 0
0 0 1 1	3	0 0 1 1
0 1 0 0	4	0 1 0 0
0 1 0 1	5	0 1 0 1
0 1 1 0	6	0 1 1 0
0 1 1 1	7	0 1 1 1
1 0 0 0	8	1 0 0 0
1 0 0 1	9	1 0 0 1

eg 1) Convert BCD 0101 1001 into Binary

BCD → 0101 1001

Convert it in decimal digit

∴ 0101 → 5

1001 → 9

∴ 0101 1001 → 59 (in decimal)

Convert decimal 59 to Binary

2	59	
2	29	1
2	14	1
2	7	0
2	3	1
	1	1

1 1 1 0 1 1

Binary to Gray code

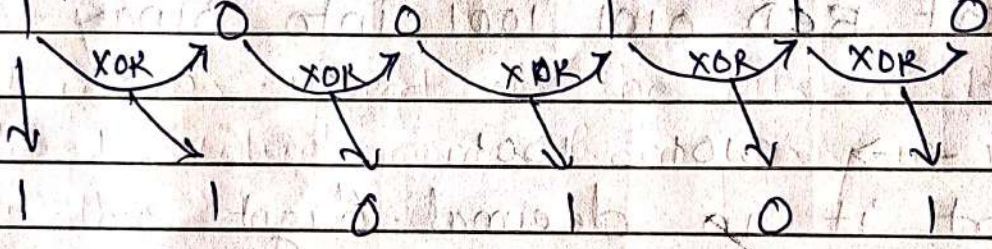
Gray code is a binary number system where two successive values differ by only one bit.

Rules

The most significant bit (MSB) of Gray code is the same as the MSB of the binary number.

Each subsequent gray code bit is obtained by XORing the previous binary bit with current binary bit.

Binary code



Convert Binary number 1011 to Gray code

Binary → 1011

1 0 1 1
B_3 B_2 B_1 B_0

Copy MSB

$G_3 = B_3 = 1$

XOR pairs

$G_2 = B_3 \oplus B_2 = 1 \oplus 0 = 1$

$G_2 = B_2 \oplus B_1 = 0 \oplus 1 = 1$

$G_1 = B_1 \oplus B_0 = 1 \oplus 1 = 0$

(1011)

Binary to Gray code

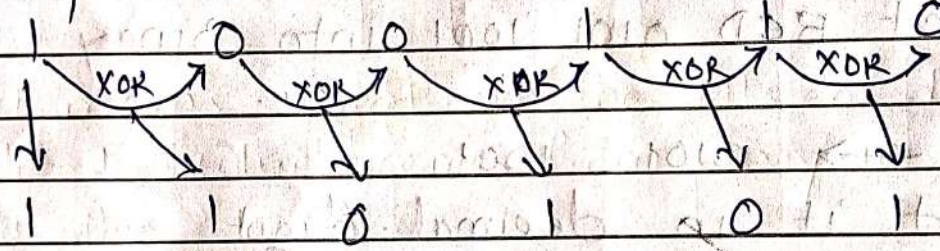
Gray code is a binary num. System where two successive values differ by only one bit.

Rules

1> The most significant bit (MSB) of Gray code is the same as the MSB of the binary num.

2> Each subsequent gray code bit is obtained by XORing the previous binary bit with current binary bit.

Binary code



Ex. 1> Convert Binary number 1011 to Gray code

Binary → 1011

1 0 1 1
B_3 B_2 B_1 B_0

Copy MSB

$G_3 = B_3 = 1$

XOR pairs

$G_2 = B_3 \oplus B_2 = 1 \oplus 0 = 1$

$G_2 = B_2 \oplus B_1 = 0 \oplus 1 = 1$

$G_1 = B_1 \oplus B_0 = 1 \oplus 1 = 0$

$(1011)_2 \rightarrow 1110$

Excess 3 to BCD

Excess 3 is a decimal coding system where each decimal digit is represented in 4 bits, but shifted by adding 3 (0011) to the BCD equivalent.

Decimal digit → BCD + 3 → Excess 3
to get back to BCD, we just subtract 3 from the Excess 3 code.

1 eg. Convert Excess 3 → 0100 to BCD

- write it in decimal binary value 0100 → 4
- Subtract 3 (decimal)
  ∴ 4 − 3 = 1

```
  0 1 0 0
- 0 0 1 1
---------
  0 0 0 1
```

- Convert the result into BCD
  Decimal 1 = BCD 0001

2 Convert Excess 3, 1011 to BCD

Binary 1011 = 11 (decimal)
Subtract 3 ⇒ 11 − 3 = 8

BCD for 8 = 1000

∴ **1011 (Excess 3) = 1000 (BCD)**

Adder and their use as a subtractor

- An Adder is a digital circuit that performs the arithmatic operation of addition
  - Half Adder - (HA) adds two ~~signed~~ single - bit binary numbers (A & B)
  - Full Adder Add three single bits (A, B and Carry in)
  - Digital circuits often use two's complement to represent negative numbers.
    subtraction can be performed by,

$$A - B = A + (\overline{B} + 1)$$

That means - invert all the bits of B (1's complement)
Add 1 (which comes from Carry-in input of full adder)
Add this to A.

if we have a 4-bit parallel adder,
inputs $A_3 A_2 A_1 A_0$ and $B_3 B_2 B_1 B_0$,
Mode control input M,
if M = 0 perform addition → A + B
M = 1 perform subtraction → $A + \overline{B} + 1$
This is achieved by • passing B through XOR gate with control M.
if M = 0, B passes unchanged
M = 1 B is inverted

- Connecting M also to the Carry-in (Cin) of the adder

Thus Adder can work as both Adder and subtractor

Truth Table (for 1 bit Full adder / subtractor)

A	B	M(mode)	operation	Result(S)	Cout
0	0	0	A+B	0	0
0	1	0	A+B	1	0
1	0	0	A+B	1	0
1	1	0	A+B	0	1
0	0	1	A-B	0	0
0	1	1	A-B	1	1
1	0	1	A-B	1	0
1	1	1	A-B	0	0

An Adder can be used as a subtractor by applying the two's complement method using XOR gates to invert B when subtraction is required and setting the carry-in to 1.

Look Ahead Carry (CLA)

A Carry Look Ahead (CLA) is a fast adder circuit used in digital systems

It overcomes the main limitation of the Ripple Carry Adder (RCA) which is slow carry propagation

The CLA achieves high speed by calculating carry bits in parallel using special logic

CLA introduces two signals

- Generate (G_i) – A carry is generated at bit i.
  $G_i = A_i \cdot B_i$
- Propagate (P_i)
  A carry is propogated through bit i

Carry equation for stage i

$C_{i+1} = G_i + (P_i \cdot C_i)$

Carry equation for 4-bit CLA

For inputs $A_3 A_2 A_1 A_0$, $B_3 B_2 B_1 B_0$ and Carry in C_0

$C_1 = G_0 + P_0 \cdot C_0$

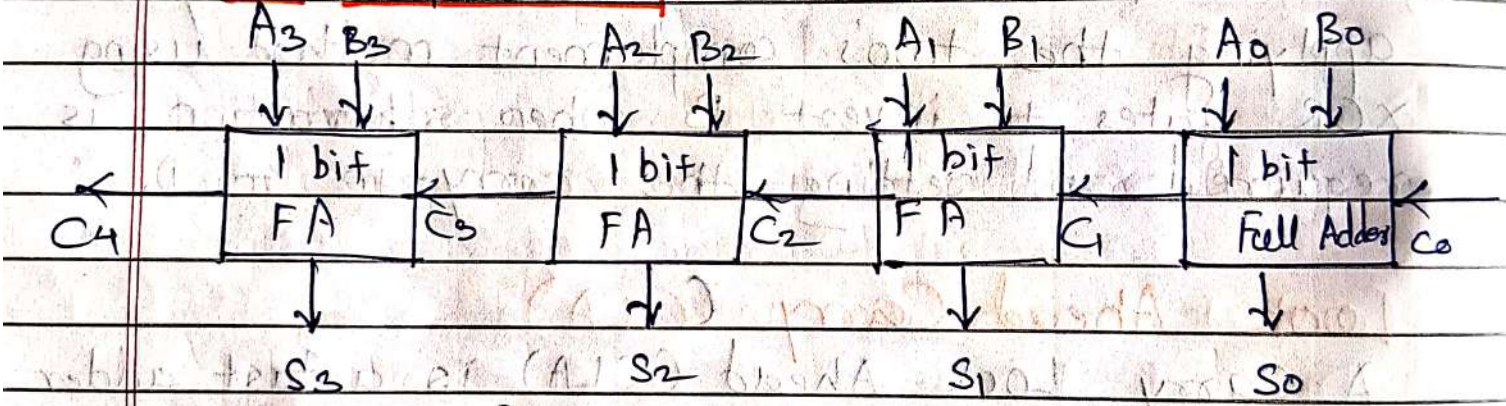
$C_2 = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$

$C_3 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0$

all carries (C_1 to C_4) are calculated in parallel

Sum equation

$S_i = P_i \oplus C_i$



• Block diagram •

Advantages

- Very fast compared to ripple carry adder
- Suitable for high speed processors.
- Parallel carry computation reduces delay

Disadvantage

- Circuit complexity increases
- Requires more gates.

ALU (Arithmatic Logic Unit)

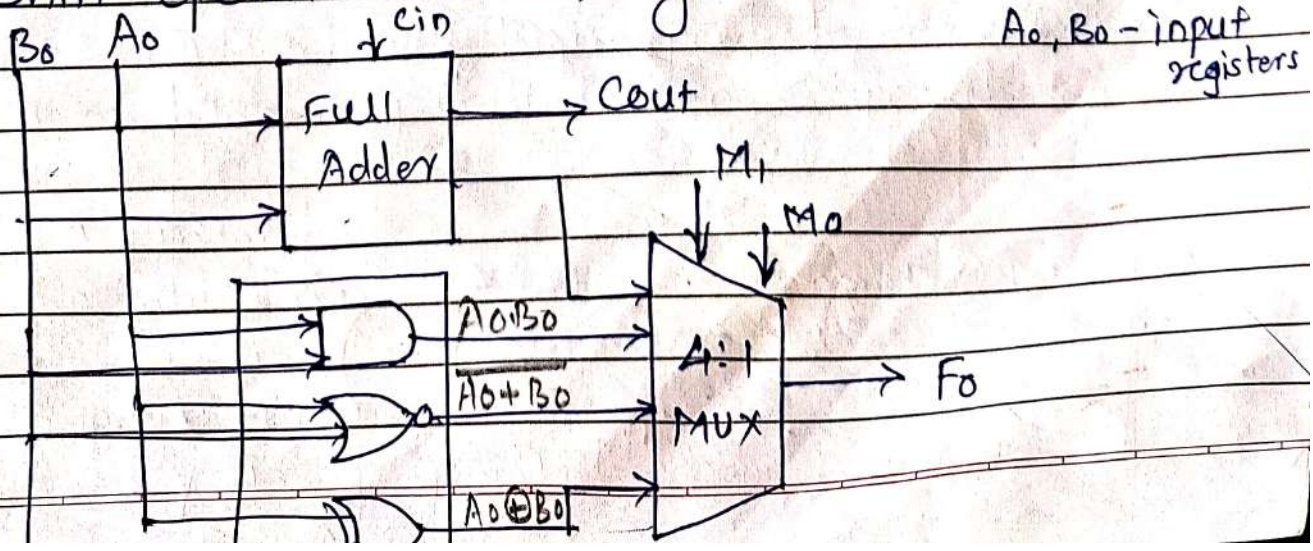
The ALU is a combinational digital circuit used to perform arithmatic and logical operations in a computer.

It is brain of CPU working under control of the control unit.

Function —

a) Arithmatic operations
- addition
- subtraction (using 2's complement)
- Multiplication (basic add / shift)
- Division (basic subtract / shift)
- Increment, Decrement
- Compare (A − B to check equal / greater / less)

b) Logical operations.
- AND
- OR
- NOT
- XOR
- NAND, NOR
- Shift operations (left, right, arithmetic shift, rotate)



ALU receives two input operands (A_0, B_0) and Opcode (operation code)

- Depending on control lines.
  Arithmetic circuit (for +, −, ×, ÷ etc)
  OR logic circuit (for AND, OR, XOR, etc)
  selected result is sent to the output register.

Advantages

- Performs both arithmetic and logical operation in one unit.
- Reduces hardware cost
- Flexible and programmable.

Disadvantages

- Complexity increases with number of operations.
- Requires more control logic.

Applications.

- In CPU
- Microcontroller
- Digital signal processing
- Embedded system

Digital Comparator

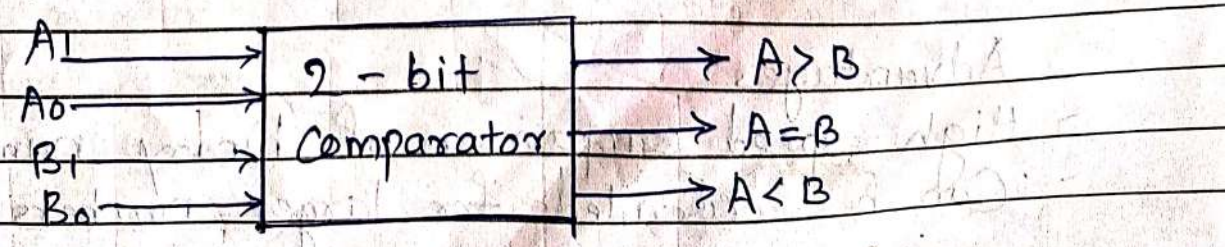
It is a combinational circuit that compares two binary numbers and determines their relative magnitude.
it outputs whether
$A = B$
$A > B$
$A < B$

Types of comparators

1) (One) 1-bit comparator
- compares two single bit inputs A and B
Outputs
$A = B \longrightarrow$ XNOR gate
$A > B \longrightarrow A \cdot \overline{B}$
$A < B \longrightarrow \overline{A} \cdot B$

2) Multi-bit Comparator
- Compares binary numbers of more than 1 bit (eg. 2 bit, 4 bit, 8 bit).
- Built by cascading 1-bit comparators.

2-bit Comparator
It has 4 binary inputs ($A_1 A_0$, $B_1 B_0$)
Outputs $A > B$, $A = B$, $A < B$



Truth Table

inputs				outputs		
A_1	A_0	B_1	B_0	$A > B$	$A = B$	$A < B$
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

$A = B$

$$(A_1 \cdot B_1 + \overline{A_1} \cdot \overline{B_1}) \cdot (A_0 \cdot B_0 + \overline{A_0} \cdot \overline{B_0})$$

$A > B$

$$A_1 \cdot \overline{B_1} + (A_1 \odot B_1)(A_0 \cdot \overline{B_0})$$

$A < B$

$$\overline{A_1} \cdot B_1 + (A_1 \odot B_1)(\overline{A_0} \cdot B_0)$$

Advantages

- High speed comparison of binary numbers
- Can be cascaded for larger numbers

Disadvantage
Hardware complexity increases with number of bits.

Applications
- Arithmetic operations
- sorting circuits
- memory address comparison
- password matching systems.

Parity Generator / checker

Parity is a simple error detection technique used in digital communication.

A parity bit is an extra bit added to a binary message to make the total number of 1's either
Even (Even parity) → total 1's = even
Odd (odd parity) → total 1's = odd

It has inputs (Bits - $D_0, D_1, D_2, \ldots$)
It gives output one extra bit parity bit, P.

a) Even parity generator:
Parity bit is choosen so that the total no. of 1's is even
$P = D_0 \oplus D_1 \oplus D_2 \oplus \ldots$

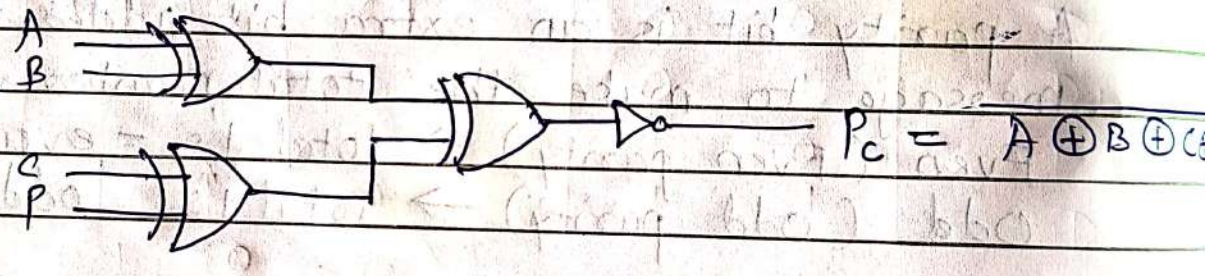
eg. Data = 1011 (three 1's → odd)
Even parity bit 1 (now total 4 ones → even)
Transmitted → 10111

Odd Parity Generator.
Parity bit is chosen so that the total numb
of 1's is odd.

$$P = \overline{D_0 \oplus D_1 \oplus D_2 \oplus \dots}$$

Example → Data = 1011 (three → 1's → odd)
odd parity bit = 0 (total number remains od
transmitted = 1011 0 =3

P



Parity checker
- Checks whether the received message has errors.

Operation — XOR all bits including parity bit
Result = 0 → No error (parity condition satisfied)
Result = 1 → Error detected (parity condition violated)

eg. Even parity checking
Transmitted 1011 1 (even parity)
Receiver XORs all bits 1 ⊕ 0 ⊕ 1 ⊕ 1 = 0
No error

If one bit flips during transmission 11111
XOR result = 1
error detected.

Truth table

Data bits (D_2 D_1 D_0)	Parity bit (P)	Transmitted bits
0 0 0	0	0 0 0 0
0 0 1	1	0 0 1 1
0 1 0	1	0 1 0 1
0 1 1	0	0 1 1 0
1 0 0	1	1 0 0 1
1 0 1	0	1 0 1 0
1 1 0	0	1 1 0 0
1 1 1	1	1 1 1 1

Application
- Data transmission in networks.
- Memory systems for error detection
- Communication devices
- Digital storage system

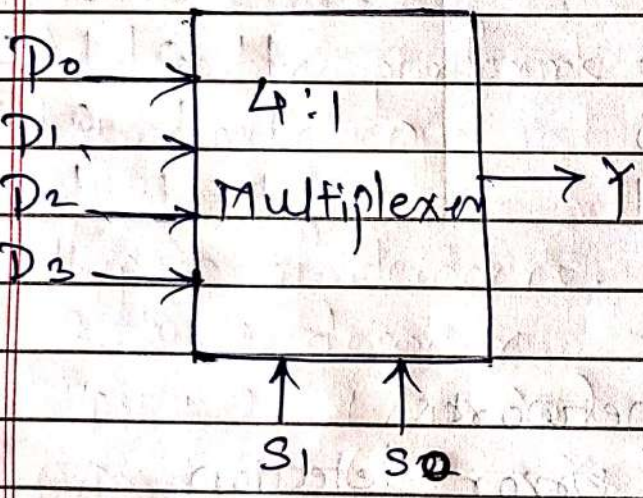
Advantages:
- simple and low cost error detection method
- easy to implement with XOR gates

Disadvantages:
- Can only detect odd number of errors
- Does not correct errors, only detects
- Fails if two or more bits change.

Design of Multiplexer (MUX)

Multiplexer

- A mux is a combinational circuit that selects one input from many and routes it to single output line.
- Also called data selector
- It has → 2^n data input
  n selection lines
  1 output



Truth table

Input		Output
S_2	S_1	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Equation

$$Y = \overline{S_1} S_0 D_0 + \overline{S_1} S_0 D_1 + S_1 \overline{S_0} D_2 + S_1 S_0 D_3$$

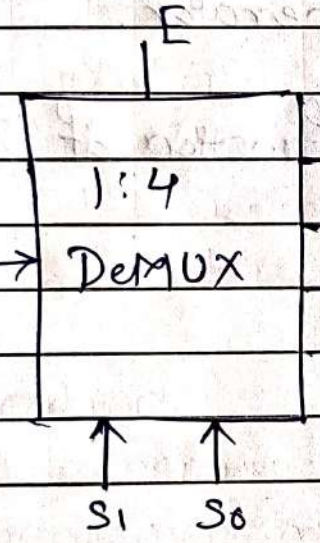
Applications

- Data routing
- Function generator
- Used in CPU
- Communication systems

Demultiplexer (DEMUX)

- A demux is the reverse of a MUX
- It takes 1 input and routes it to one of many outputs, depending on select lines.
- Also called a data distributer

It has → 1 input
n selection lines
2^n outputs.



Truth table

E	S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	Din
1	0	1	0	0	Din	0
1	1	0	0	Din	0	0
1	1	1	Din	0	0	0

Equation →
$$Y_0 = \overline{S_1}\,\overline{S_0}\,D$$
$$Y_1 = \overline{S_1}\,S_0\,D$$
$$Y_2 = S_1\,\overline{S_0}\,D$$
$$Y_3 = S_1\,S_0\,D$$

Application
- Data distribution
- Serial to parallel data conversion
- Used in communication (channel selection)
- Memory addressing (address decoder)

Decoder

A decoder is a combinational circuit that converts binary input into a unique output line.

It has — n input lines
2^n output lines.

only one output is active (High = 1) for each combination

- also called minterm generator.

- A decoder performs the function of binary-to-one-hot conversion

Types of decoders.

1) 2 to 4 decoder
i/ps : 2 (A_1, A_0)
o/ps : 4 (Y_0 — Y_3)
Enable (E) - used to activate / deactivate the decoder.

2) 3 to 8 decoder
i/ps : 3 (A_2 A_1 A_0)
o/ps : 8 (Y_0 - Y_7)
Can be built using two 2 to 4 decoders with enable

3) n to 2^n decoder.

Logic expression (2 to 4 decoder)

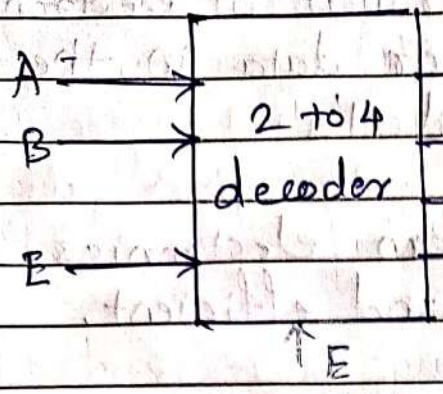
$Y_0 = \overline{A_1} \cdot \overline{A_0} \cdot E$

$Y_1 = \overline{A_1} \cdot A_0 \cdot E$

$Y_2 = A_1 \cdot \overline{A_0} \cdot E$

$Y_3 = A_1 \cdot A_0 \cdot E$

Block diagram



Truth table

A	B	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Applications

- Memory address Decoding
- Instruction decoding in CPU's
- Data demultiplexing
- 7 segment display decoder