Lecture 18
EE 421 / CS 425
Digital System Design

Fall 2025

Shahid Masud

Topics

- Floating Point Adder and Multiplier Operation (quick recap)
- Digital Circuit Design for Floating Point Module
- Introduction to Programmable Logic Hardware Device
- Nomenclature and types
- Logic Array Circuits
- PLA and PAL
- Programmable Logic Devices – PLD

Floating Point Arithmetic – Digital Design

$N=(-1)^S \times (1+F)\times 2^E$ E.g. $91.820734 \times 10^{-34}$

- A signed-magnitude system for the fractional part and a biased notation for the exponent
- Three subfields
  - Sign S
  - Fraction F (or Significand or Mantissa)
  - Exponent E
- Sign bit is 0 for positive numbers, 1 for negative numbers
- Fractions always start from **1**.xxxx, hence the integer **1** is not written (register has xxxx)
- Exponent is biased by +127 (add 127 to whatever is in register bits)
- Normalize: Express numbers is the standard format by shifting of bits and adding / subtracting from Exponent register

IEEE 754 Floating Point Representation

Single Precision IEEE 754

Sign	Exponent	Fraction
1 bit	**8 bits**	**23 bits**

1 or 2 bits

Guard Bits

← 32 bits →

Double Precision IEEE 754

Sign	Exponent	Fraction
1 bit	**11 bits**	**52 bits**

1 or 2 bits

Guard Bits

← 64 bits →

Examples of Floating Point Representation

$-(13.45)_{10}$
= (1101.01 1100 1100 1100 ……..)^2 ; this is un-normalized
= (1.10101 1100 1100 1100 1100 1) x 2^3; normalized
Fraction part is 10101 1100 1100 1100 1
Biased Exponent is 3+127 = 130
Sign = 1

5.0345
= 101 . 0000 1000 1101 0100 1111 110; this is un-normalized
= 1. 01 0000 1000 1101 0100 1111 110 x 2^2; normalized
Biased Exponent = 2+127 = 129 = $(1000\ 0001)_2$
Fraction = 01 0000 1000 1101 0100 1111 110
Sign = 0

Floating Point Arithmetic – Digital Design

$N=(-1)^S \times (1+F) \times 2^E$ E.g. $91.820734 \times 10^{-34}$

- A signed-magnitude system for the fractional part and a biased notation for the exponent
- Three subfields
  - Sign S
  - Fraction F (or Significand or Mantissa)
  - Exponent E
- Sign bit is 0 for positive numbers, 1 for negative numbers
- Fractions always start from **1**.xxxx, hence the integer **1** is not written (register has xxxx)
- Exponent is biased by +127 (add 127 to whatever is in register bits)
- Normalize: Express numbers is the standard format by shifting of bits and adding / subtracting from Exponent register

Floating Point Multiplication

Denormalize

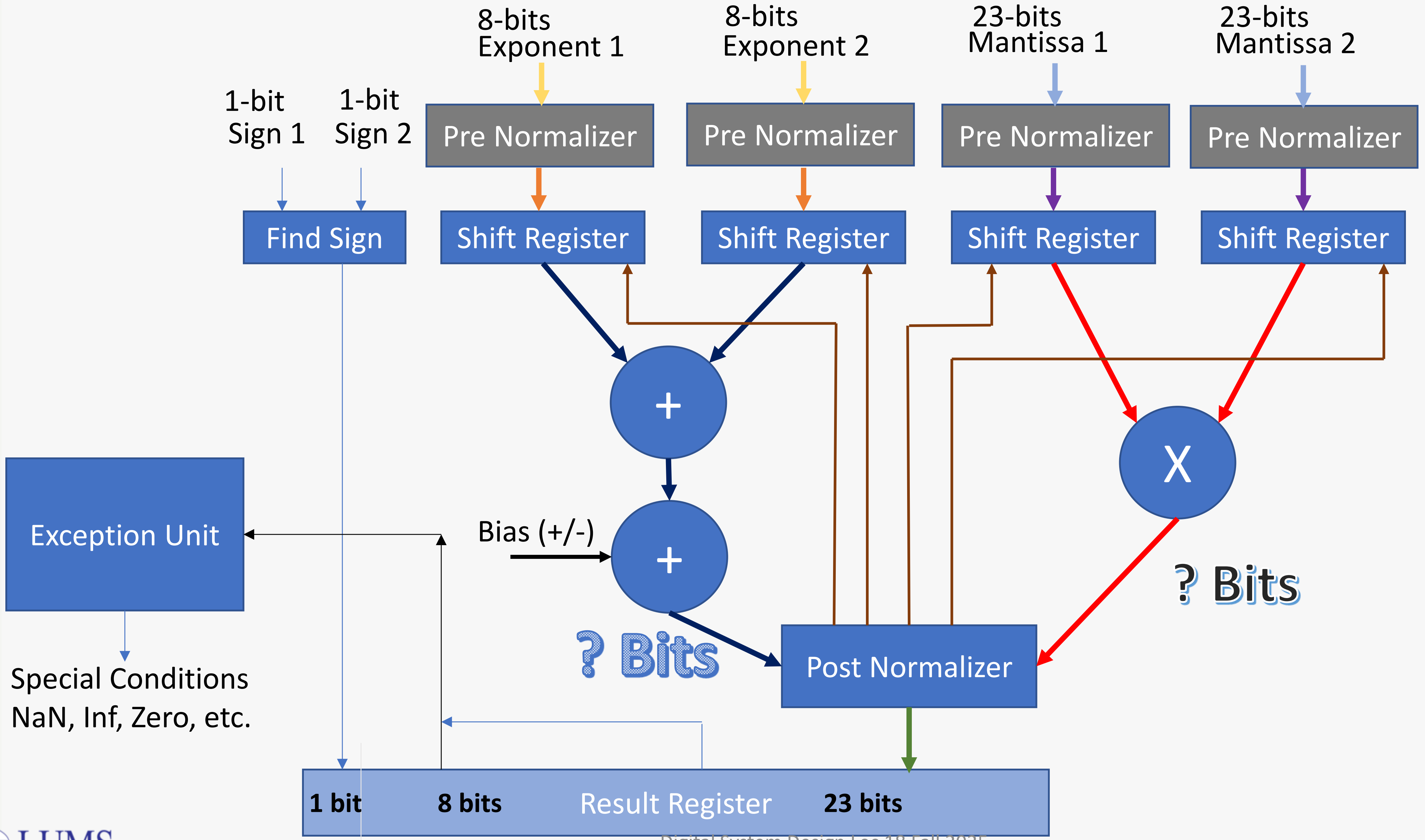
Consider two floating point numbers:
$(F_1 \times 2^{E1})$ and $(F_2 \times 2^{E2})$

The product of these two numbers is:
$= (F_1 \times 2^{E1}) \times (F_2 \times 2^{E2})$
$= (F_1 \times F_2) \times 2^{(E1+E2)}$
$= F \times 2^{E}$

Sign of result depends on Sign of the two numbers

Normalize

Data Path of Floating Point Multiplier



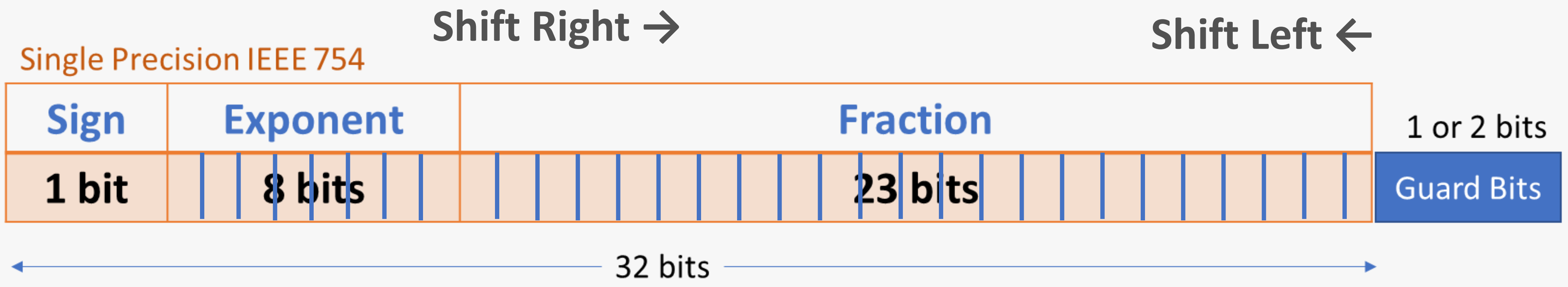
Floating Point Addition

Consider two floating point numbers:
$(F_1 \times 2^{E1})$ and $(F_2 \times 2^{E2})$

Initial Denormalize
The addition of these two numbers is:
$= (F_1 + F2) \times (2^{Enew})$
Enew is obtained by Shl or Shr of F1 and F2 so that E1 becomes same as E2
$= F \times 2^{Enew}$
Final Normalize
Take into account Guard Bits

Sign of result depends on Sign of the two numbers

Complexity of Normalize and DeNormalize



What is the Complexity, in terms of time and area?

Can this be improved?

Programmable Logic
Reconfigurable Logic

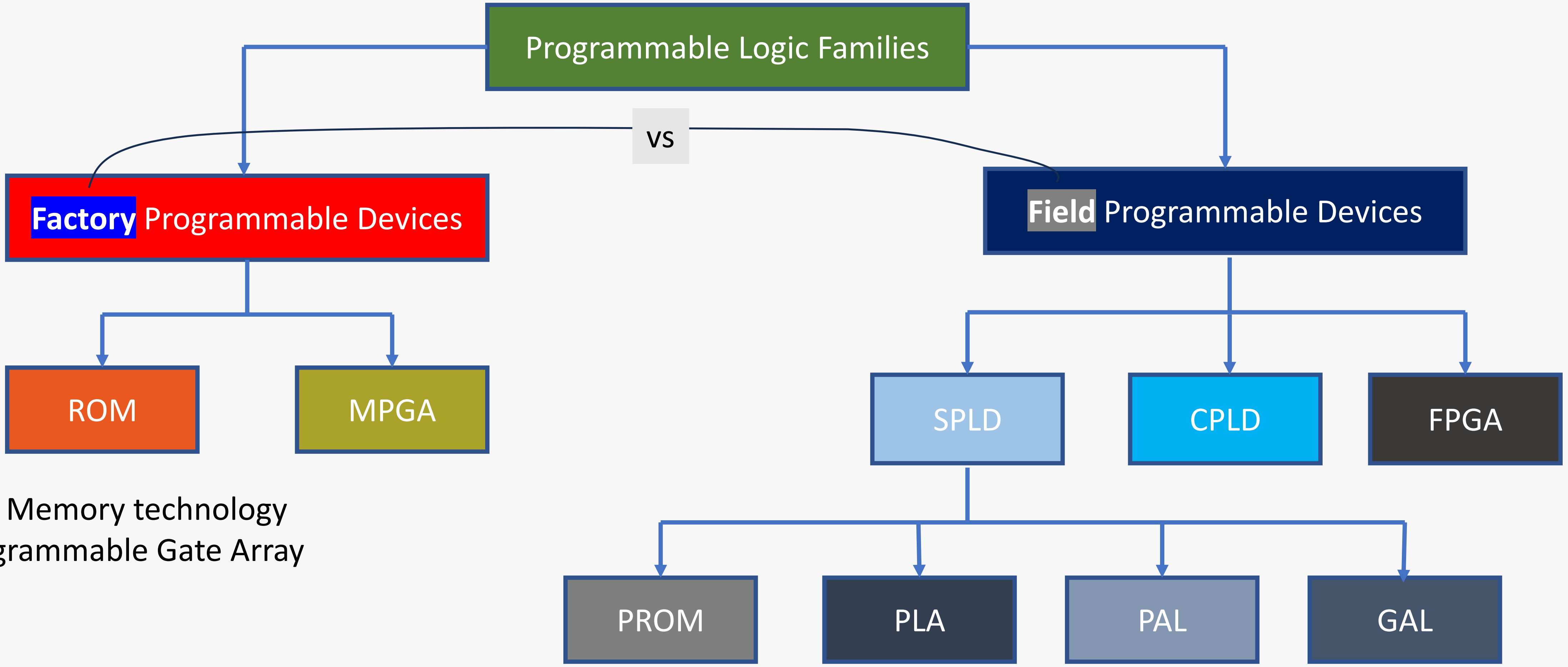
Definition of Programmable Logic

- A family of semiconductor integrated circuits that can be reconfigured through:
  - special hardware-description-languages, and
  - associate Electronic Design Automation tools
  - Special configuration equipment

to produce a variety of hardware functionality from the same integrated circuit chip.

- The functionality in hardware can be erased and reconfigured in a manner analogous to design of a software system using microprocessor.
- All other types of semiconductors have fixed and pre-defined functionality.
- Performance vs Flexibility tradeoffs exist, as with any other digital system.

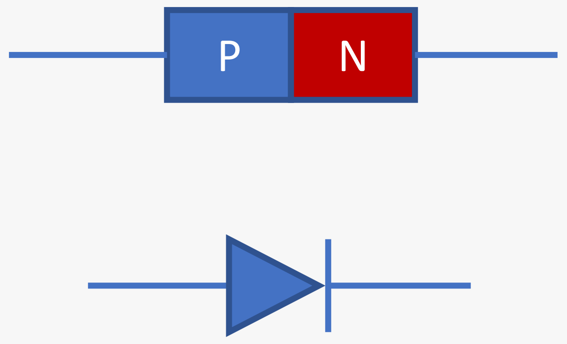
Programmable Logic Broad Classification

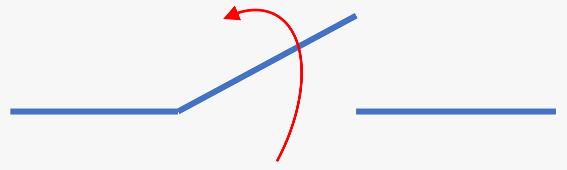

Programmable Logic Families
vs
Factory Programmable Devices
Field Programmable Devices
ROM
MPGA
SPLD
CPLD
FPGA
PROM
PLA
PAL
GAL

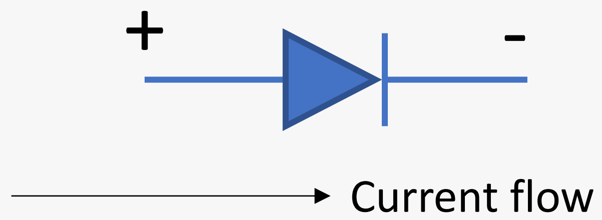
Read only Memory technology
Mask Programmable Gate Array

Simple and Complex Programmable Logic Devices
Field Programmable Gate Array
Generic Array Logic (complex PAL by lattice semi), Programmable Logic Array

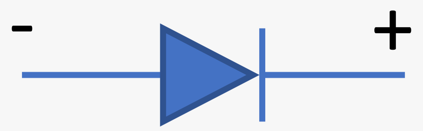
Diode – Principle of operation







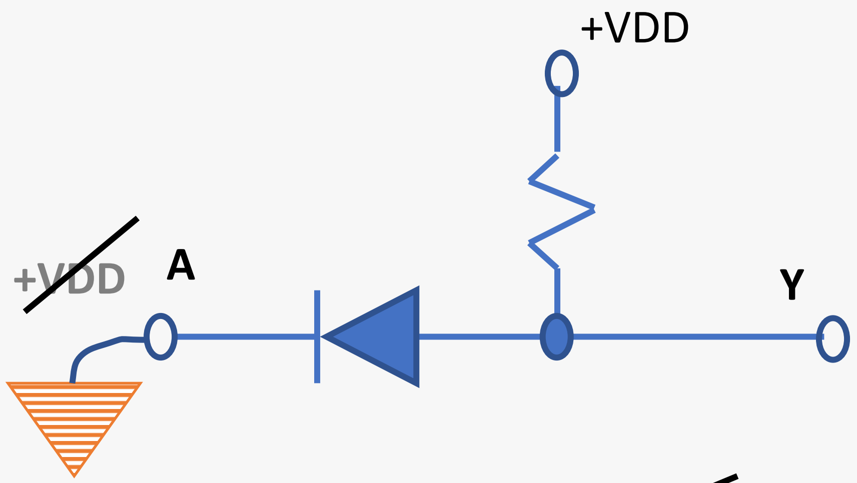
Forward Biased



X Current flow

Reverse Biased

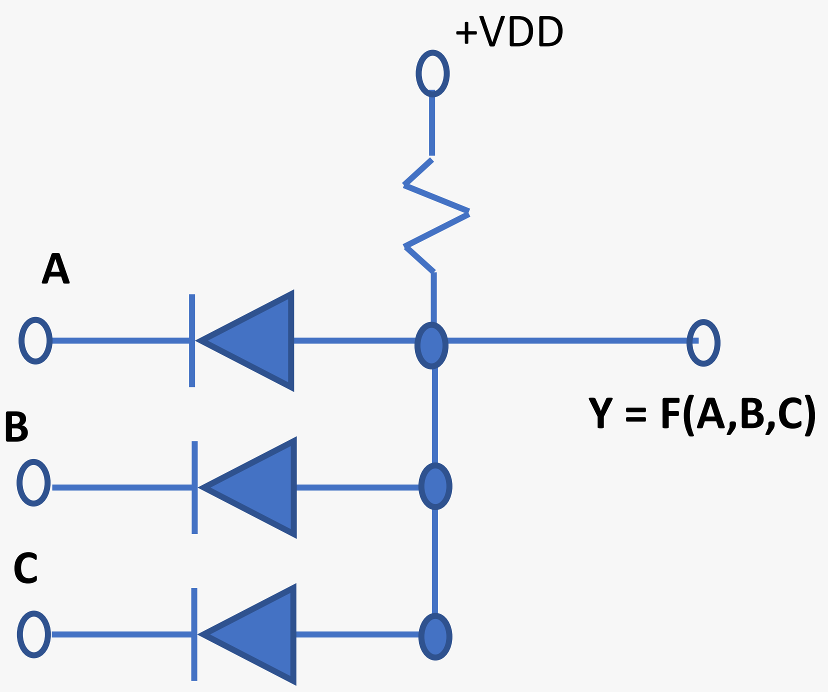
Diode in logic gates



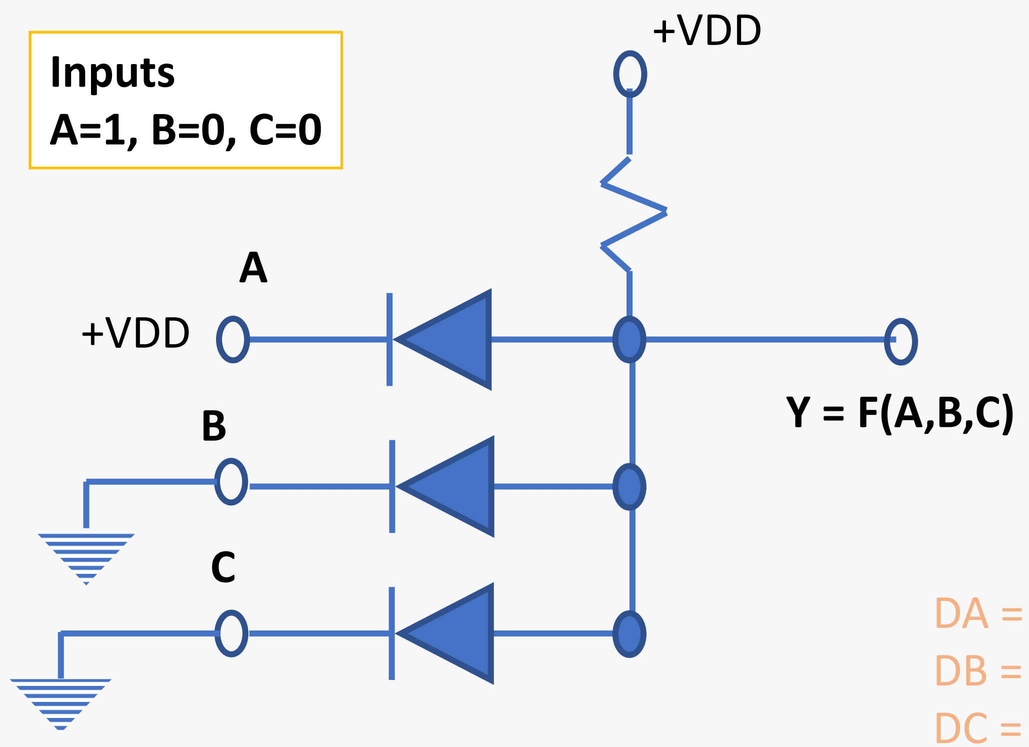
~~When A=1, Diode is Reverse Biased~~
~~Hence Output Y = 1 through +VDD~~

When A=0, Diode is Forward Biased
Hence Output Y = 0 through A = 0 Volt

Diode based AND Logic Array



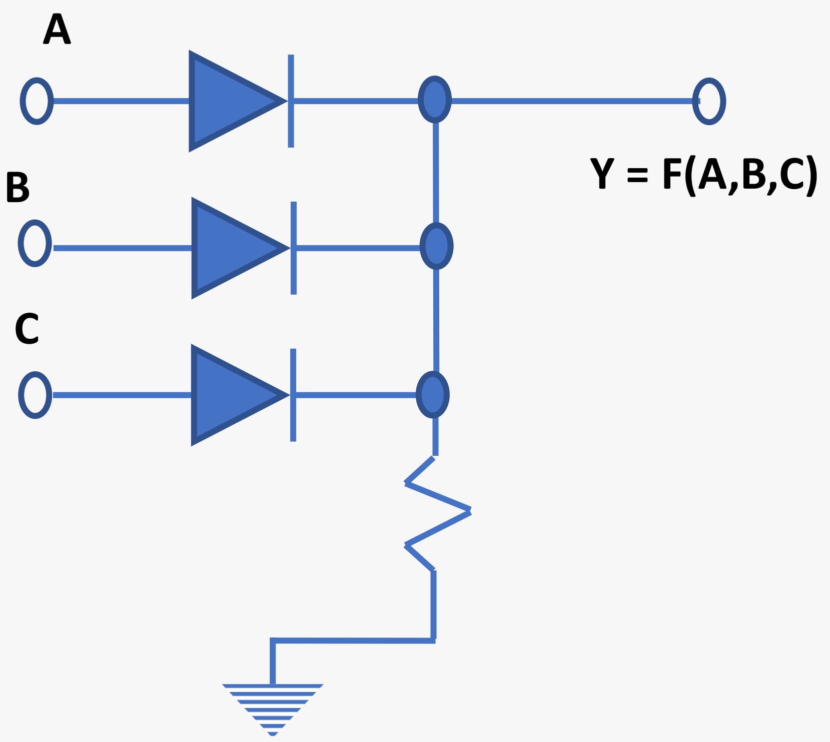
Inputs
A=1, B=0, C=0



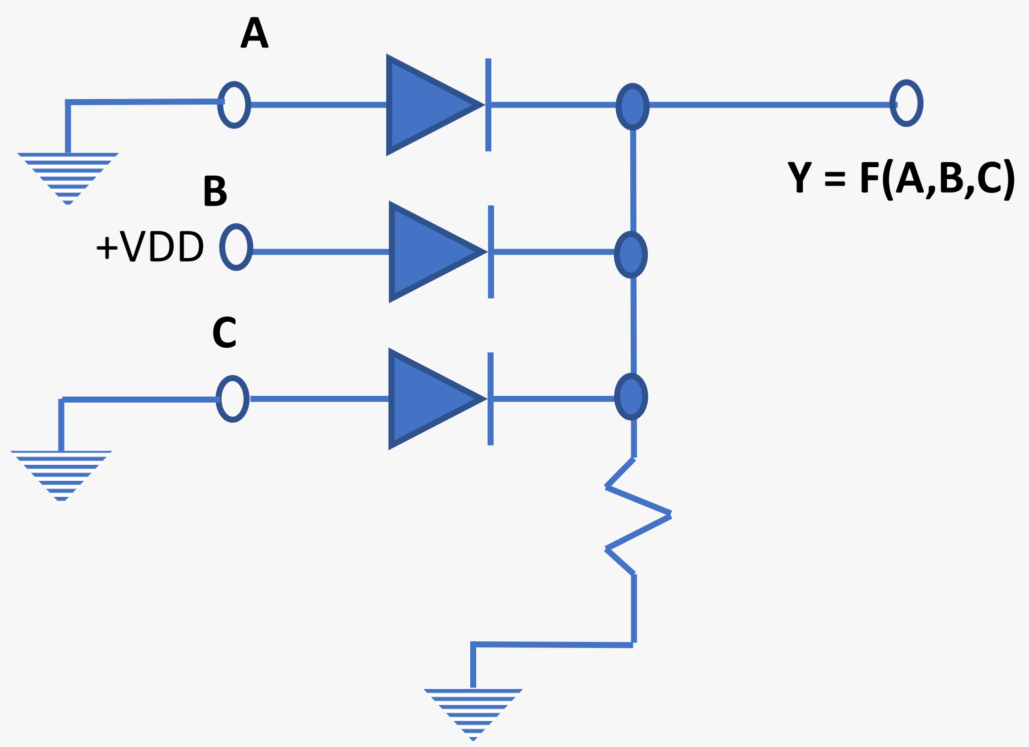
DA = Diode A
DB = Diode B
DC = Diode C

DA=Open Circuit, DB = Forward Biased Short, DC = Forward Biased Short
Y= Logic ‘0’ as it is Shorted to Zero due to DB or DC

Diode based OR array

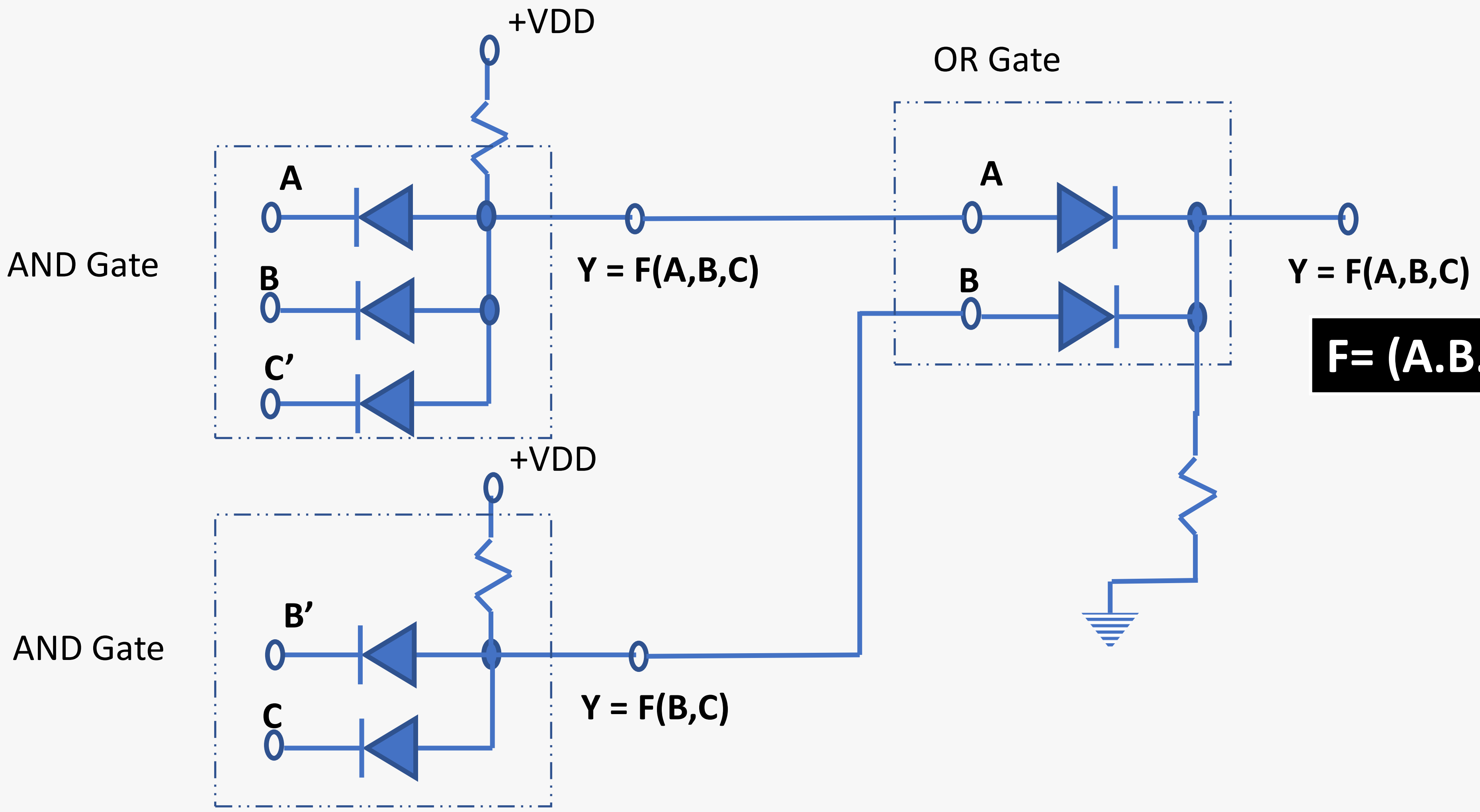


Inputs A = 0, B = 1, C = 0



DA and DC = Open Circuit due to Zero input, DB = Short Circuit due to +VDD input
Y = +VDD Logic ‘1’ due to DB being short circuit, other paths are open circuit

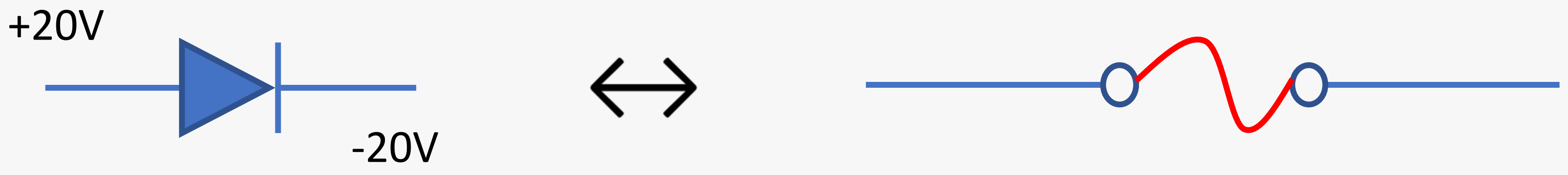
Diode based AND-OR Array



F= (A.B.C’) + (B’.C)

Diode as fuse

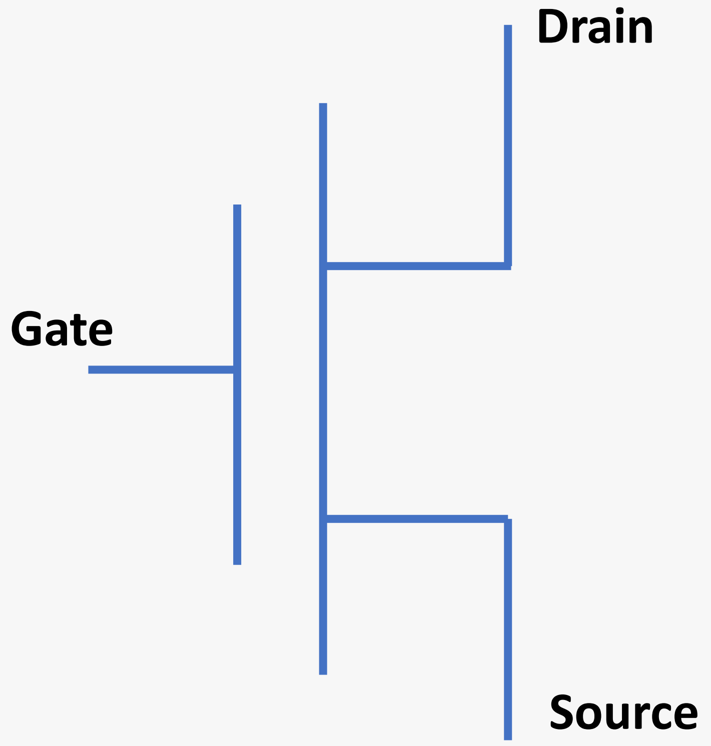
Momentarily apply a higher voltage where open circuit path is required

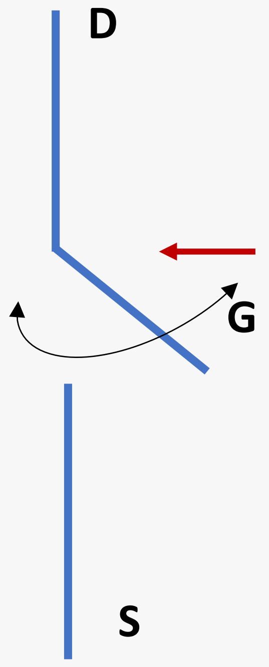


Desired paths from AND gates and OR gates will be retained, others will be blown out

Desired Paths are selected through a MASK during final stages of FABRICATION PROCESS

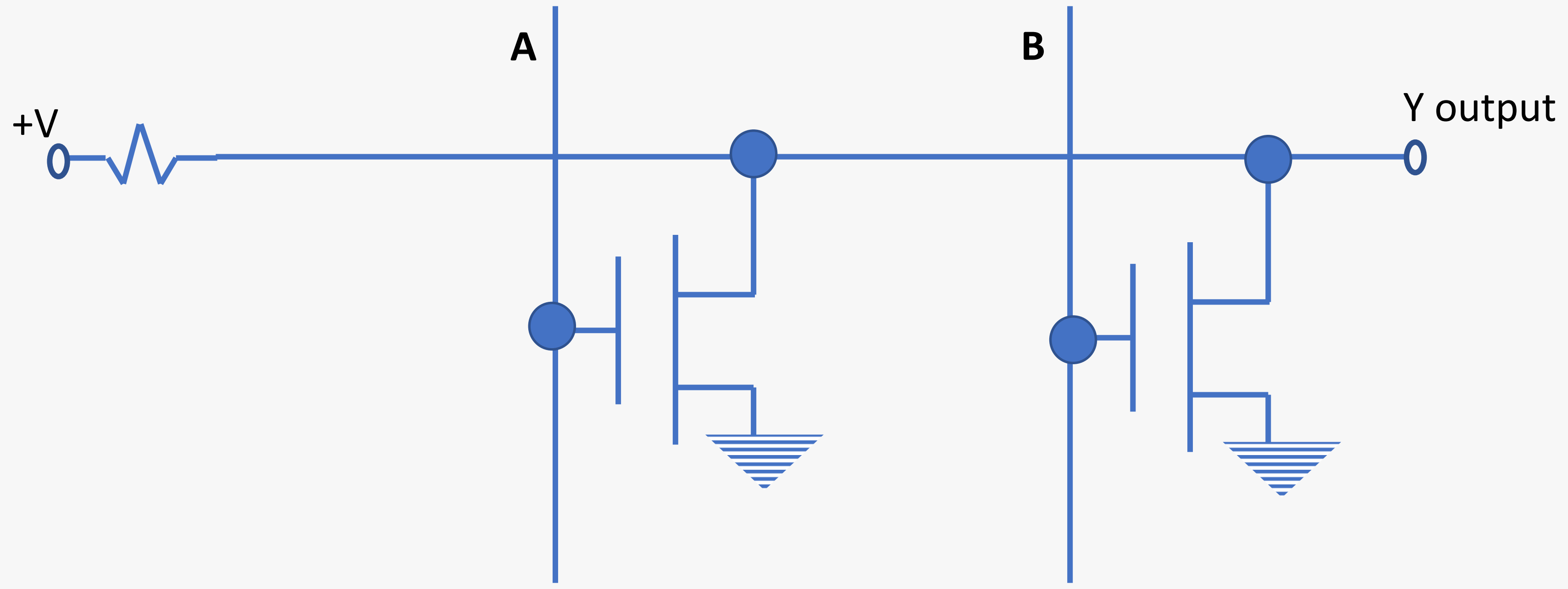
MOS Transistor as Switch

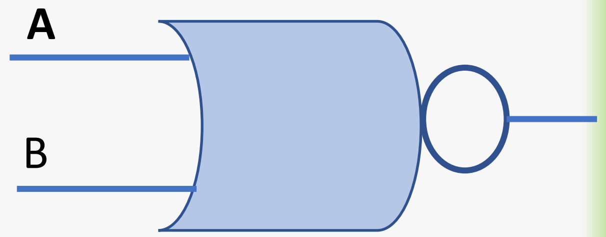




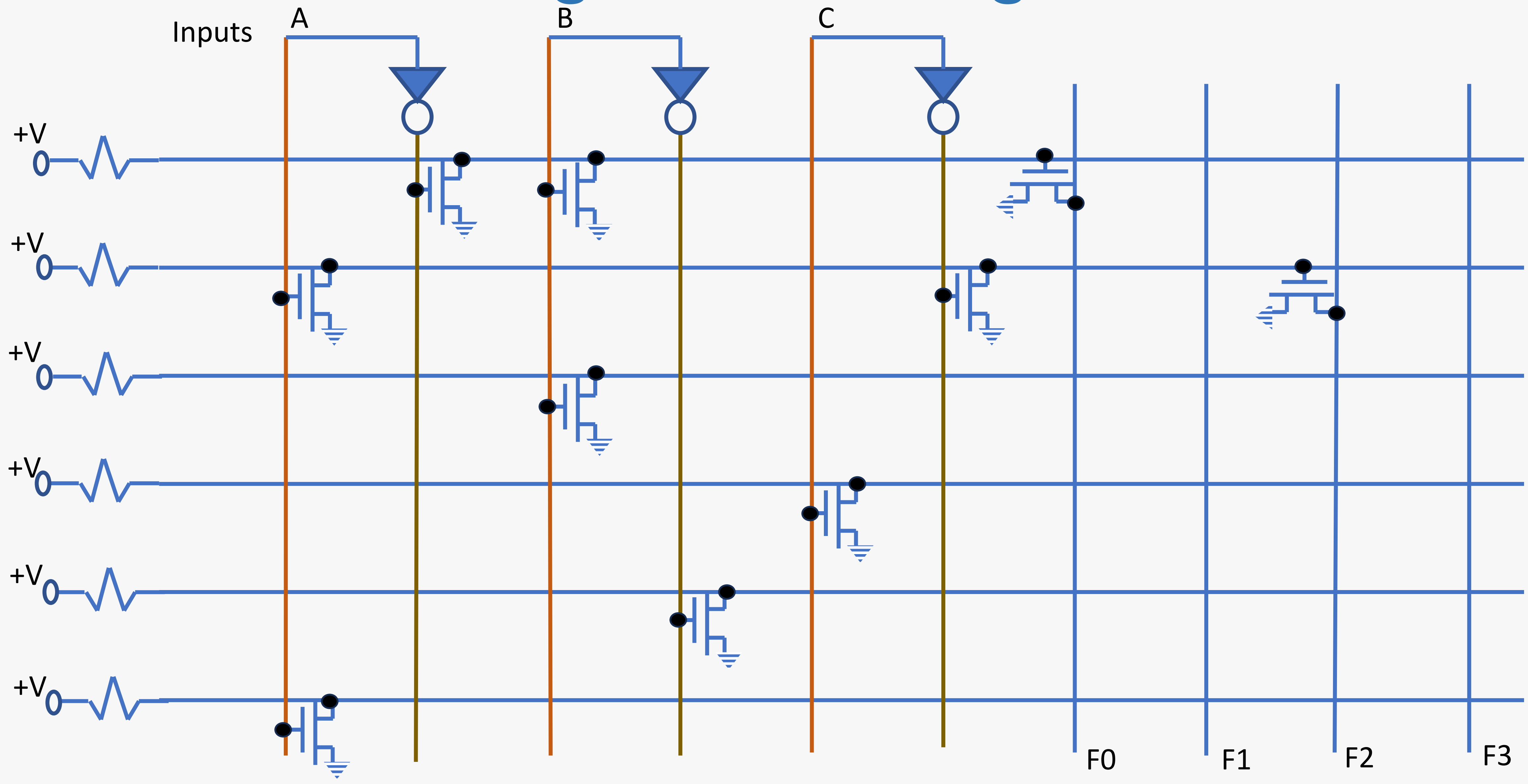
Programming is achieved through controlling GATE Voltage
High GATE Voltage will ensure flow of current, like a short circuit
Low GATE Voltage will ensure no flow of current, like an open circuit

Logic Gates Design with Transistor Switches





Transistors as Programmable Logic Switches

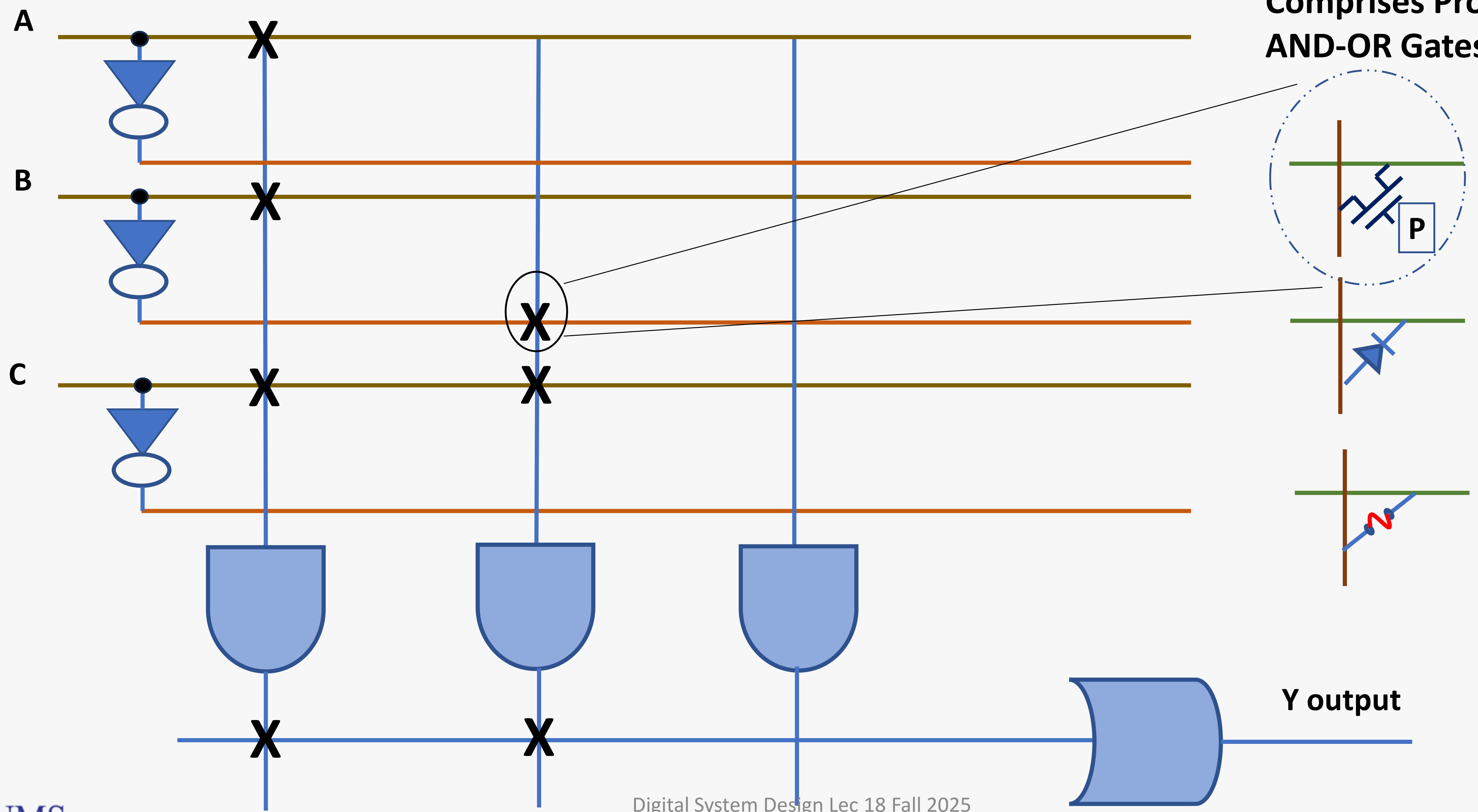


outputs

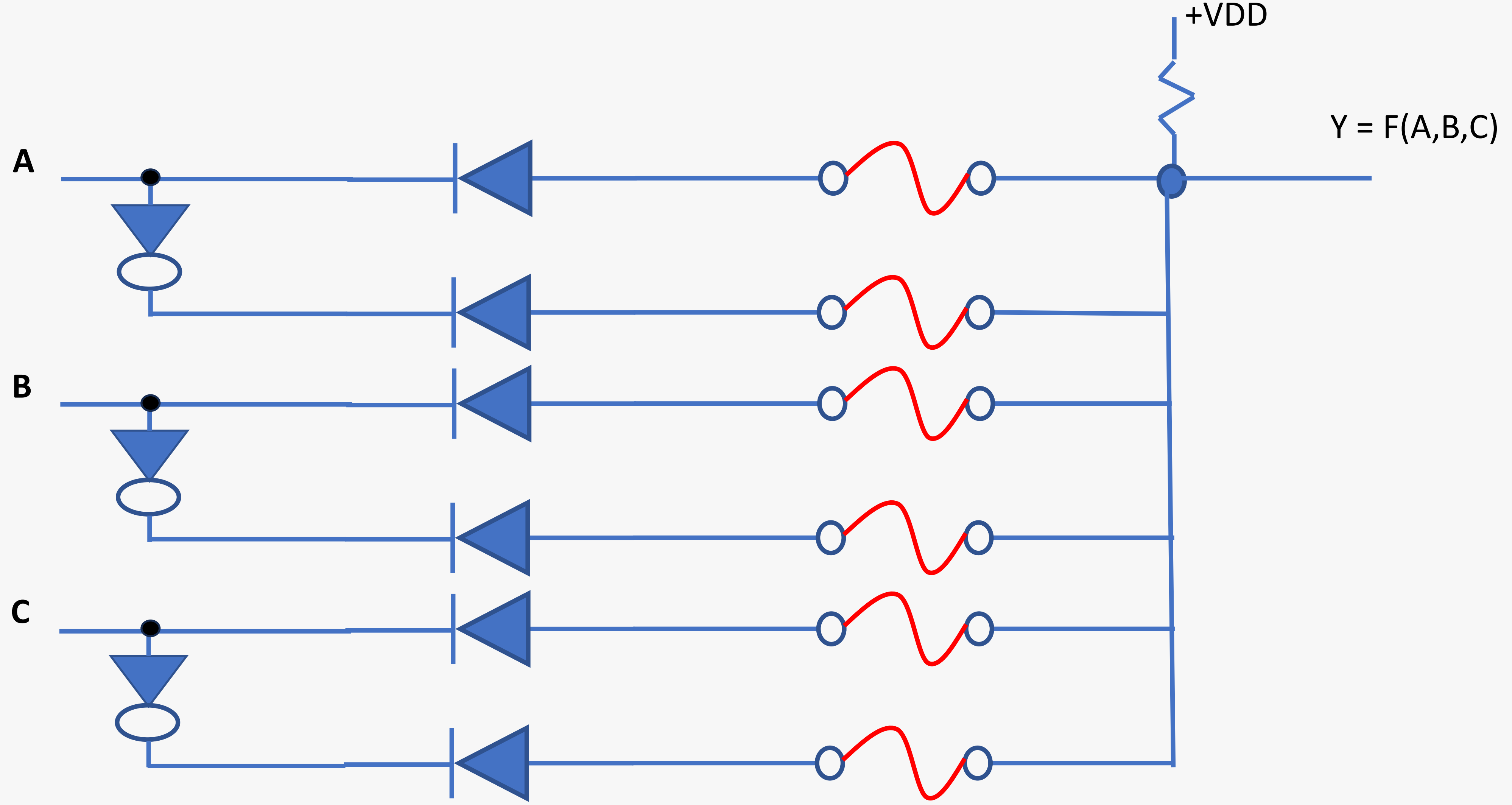
Different logic functions are realized through transistor switches

AND-OR Logic Array

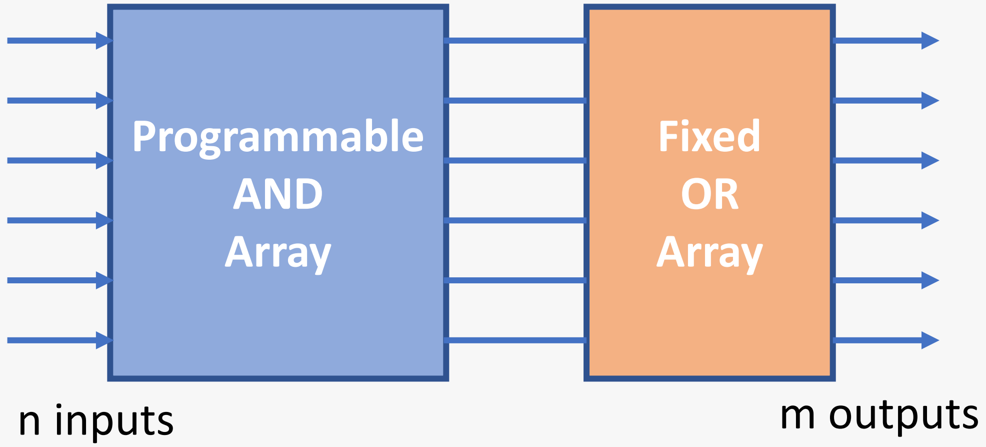
PLA = Programmable Logic Array
Comprises Programmable AND-OR Gates

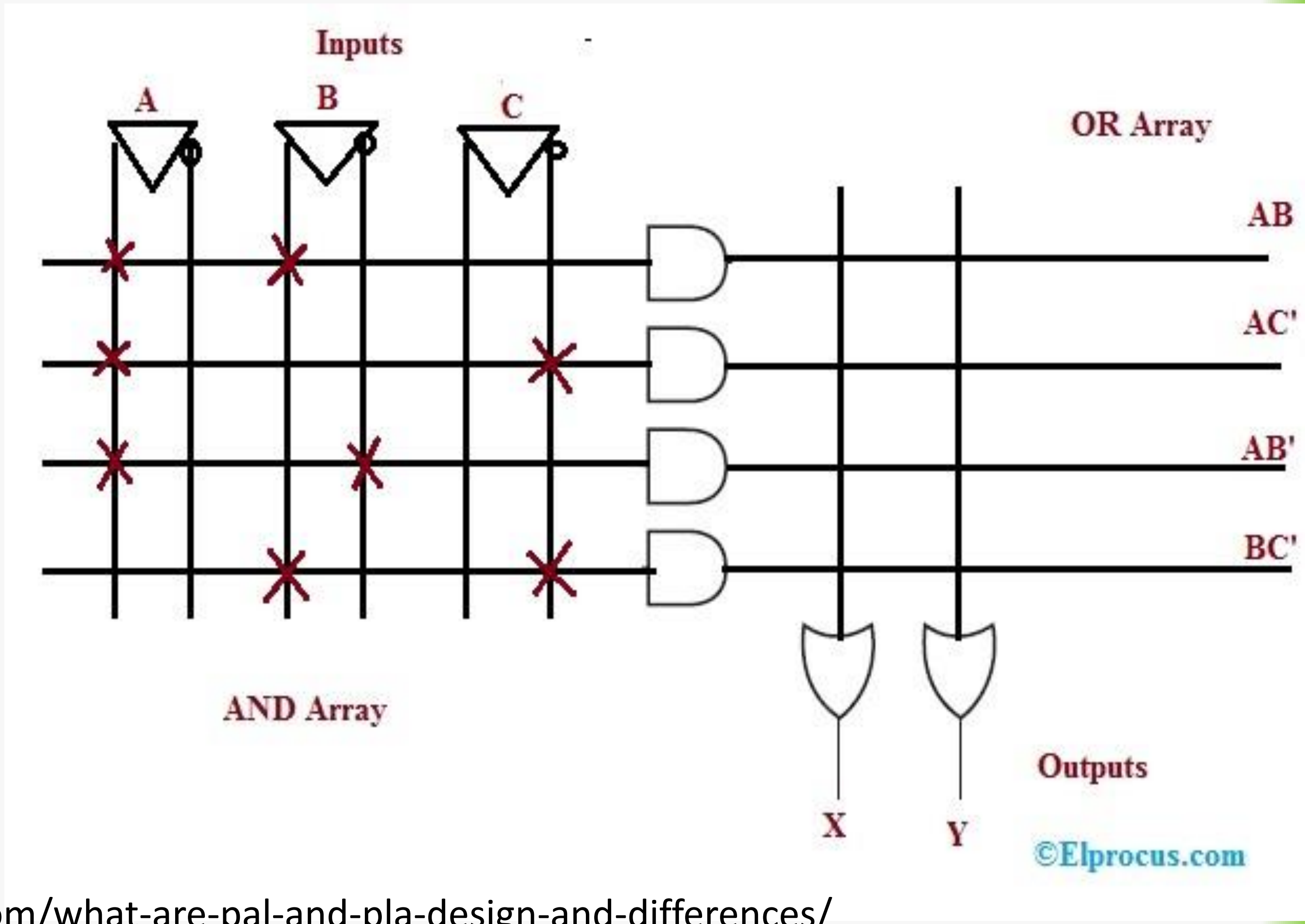


Fuse Programmable Logic Array



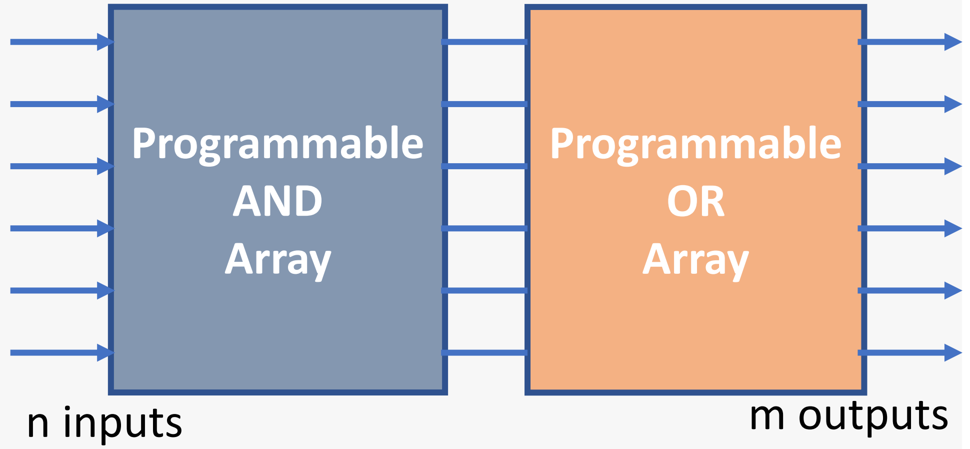
PAL – Programmable Array Logic

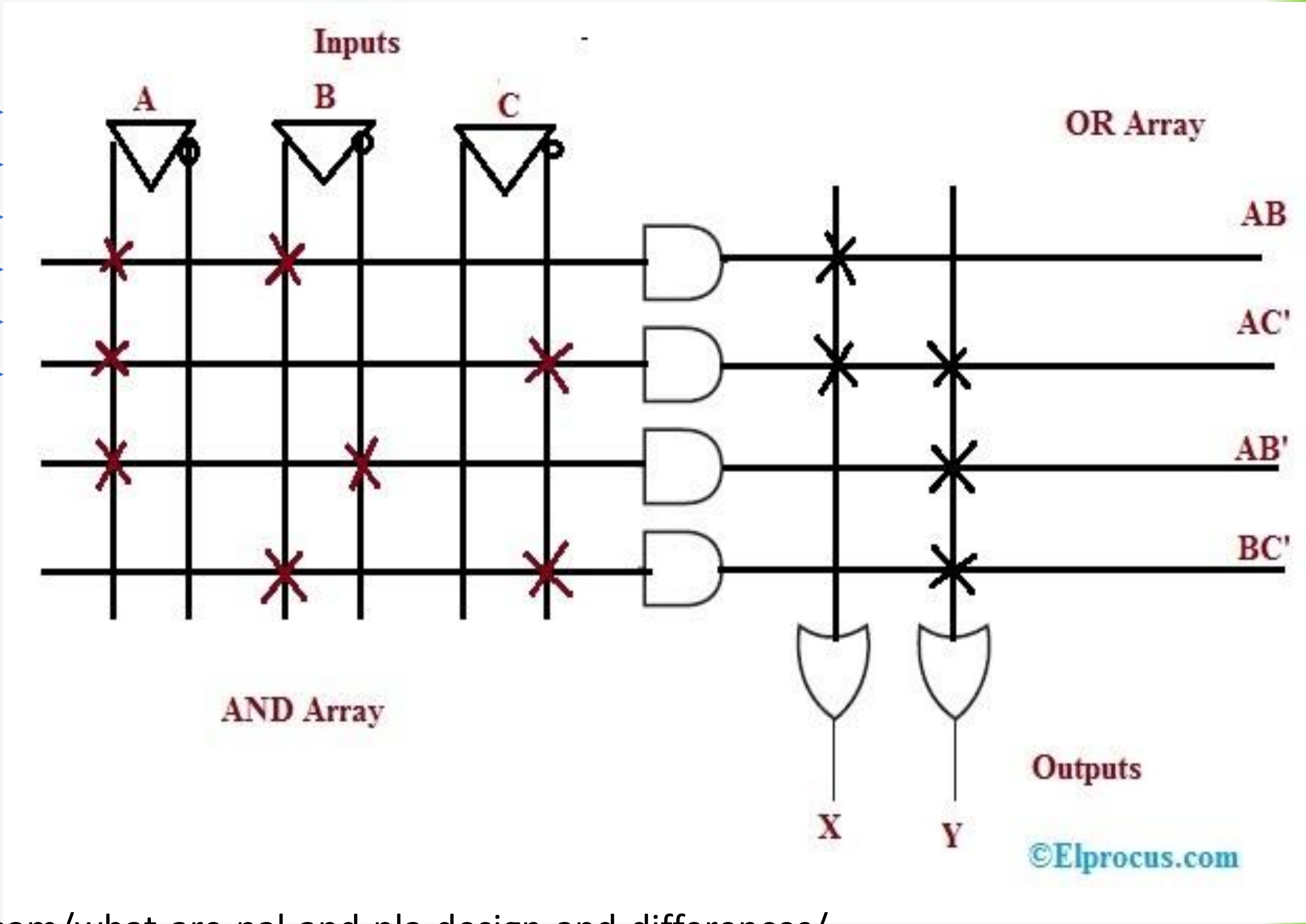




https://www.elprocus.com/what-are-pal-and-pla-design-and-differences/

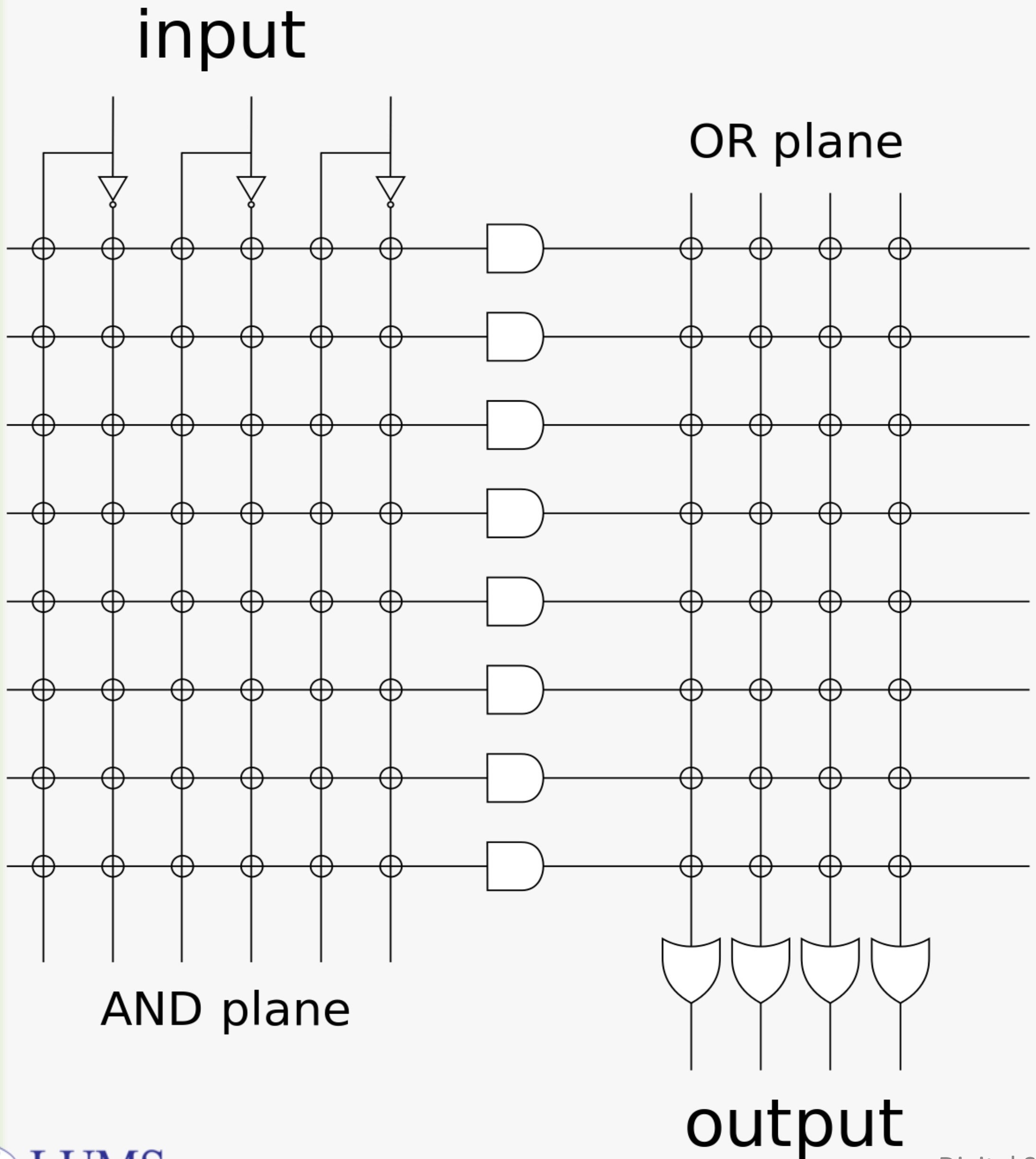
PLA – Programmable Logic Array

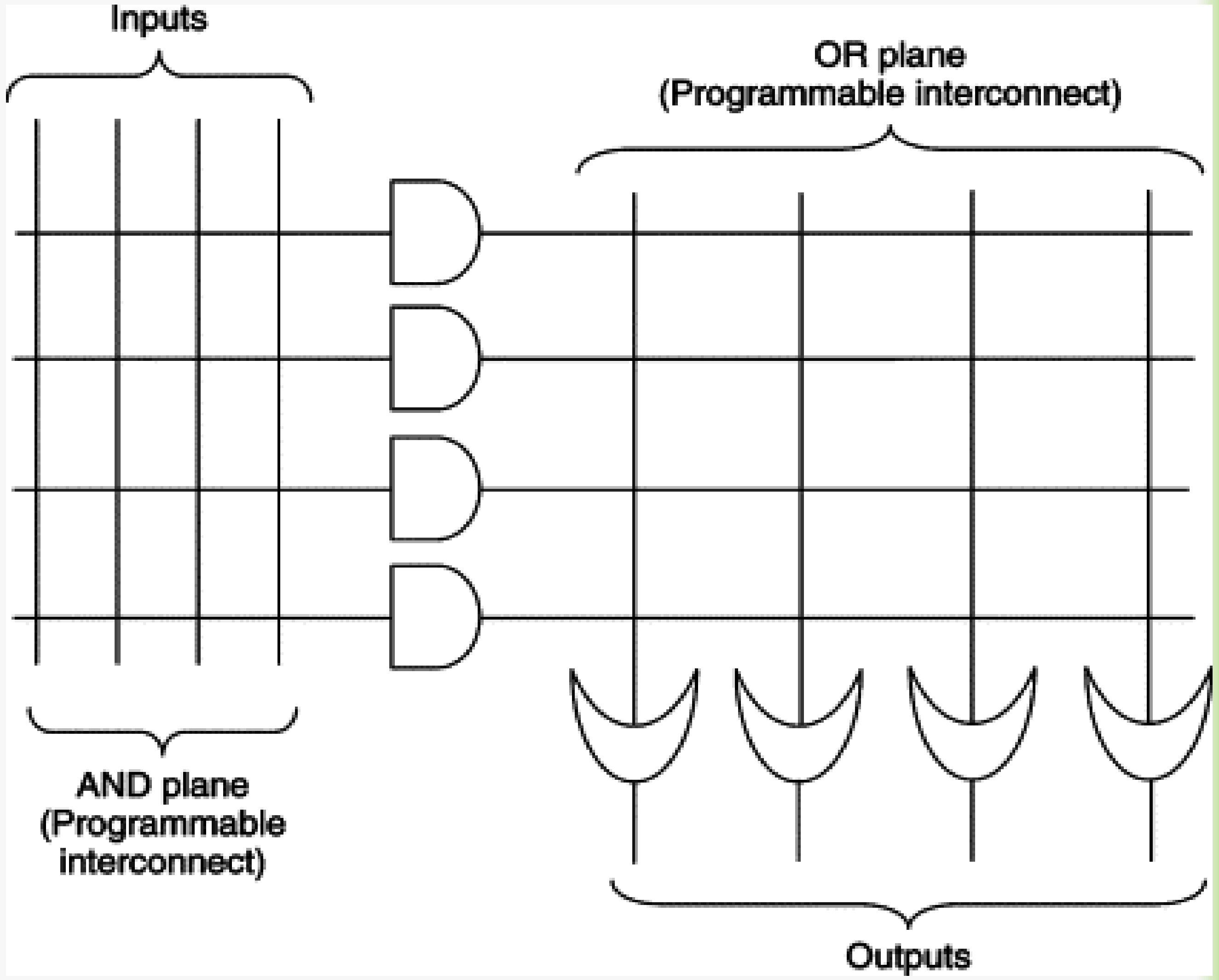




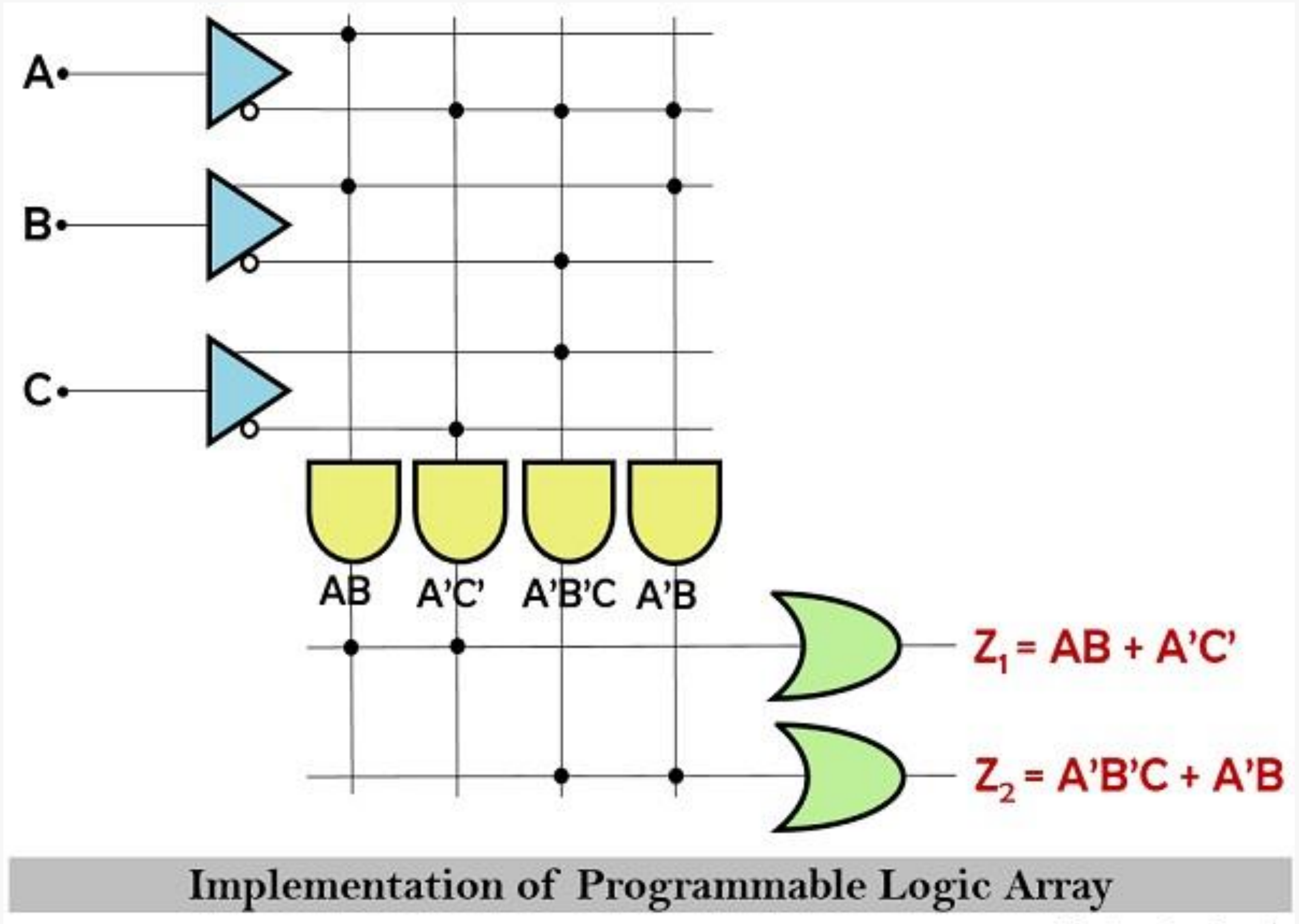
https://www.elprocus.com/what-are-pal-and-pla-design-and-differences/

For Exercise





Example of PLA

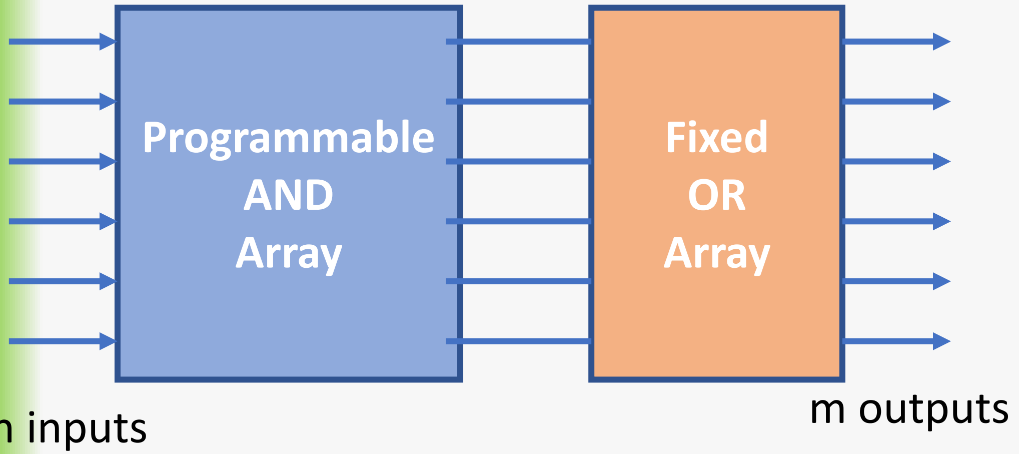


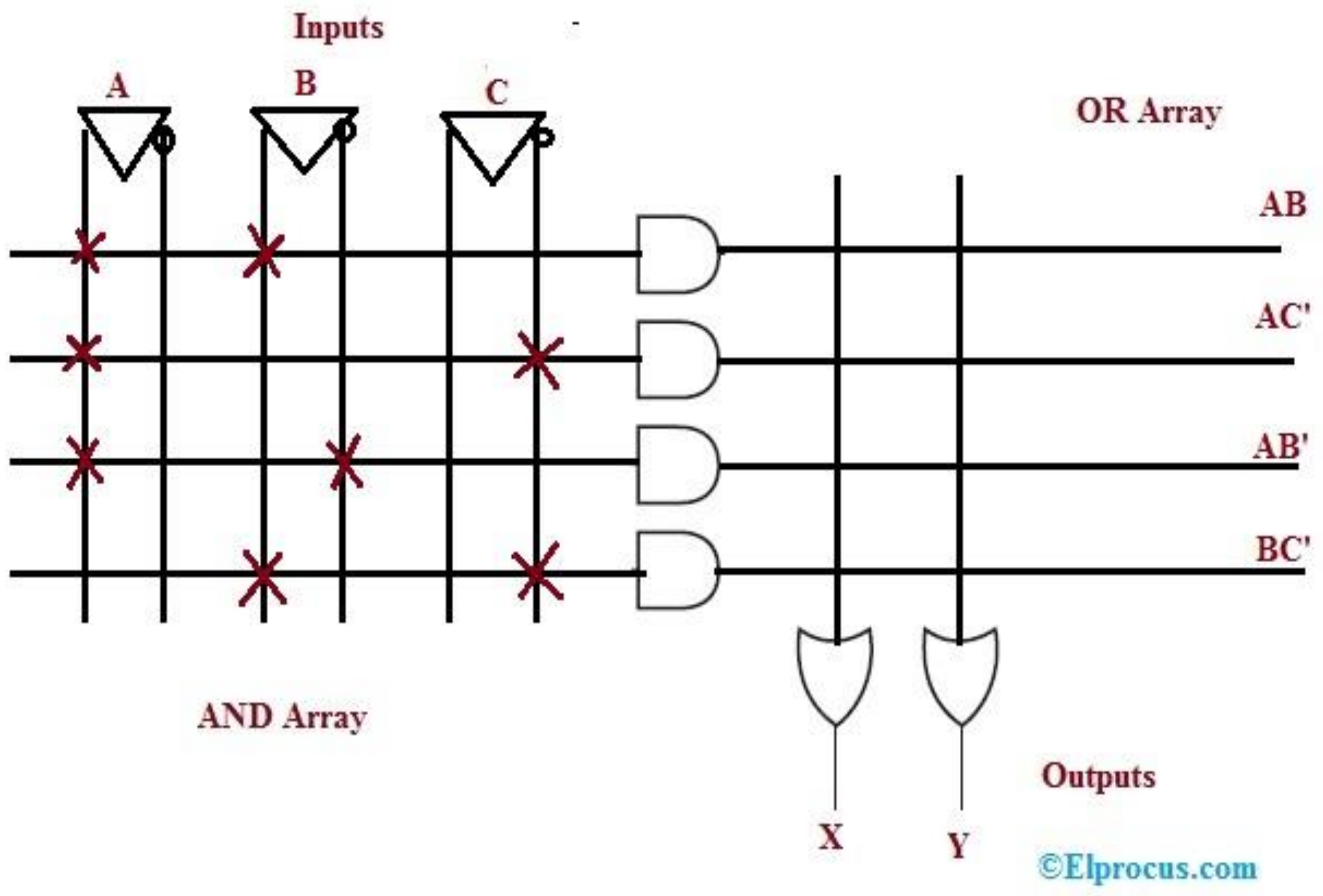
Implementation of Programmable Logic Array

Electronics Coach

https://electronicscoach.com/programmable-logic-array.html

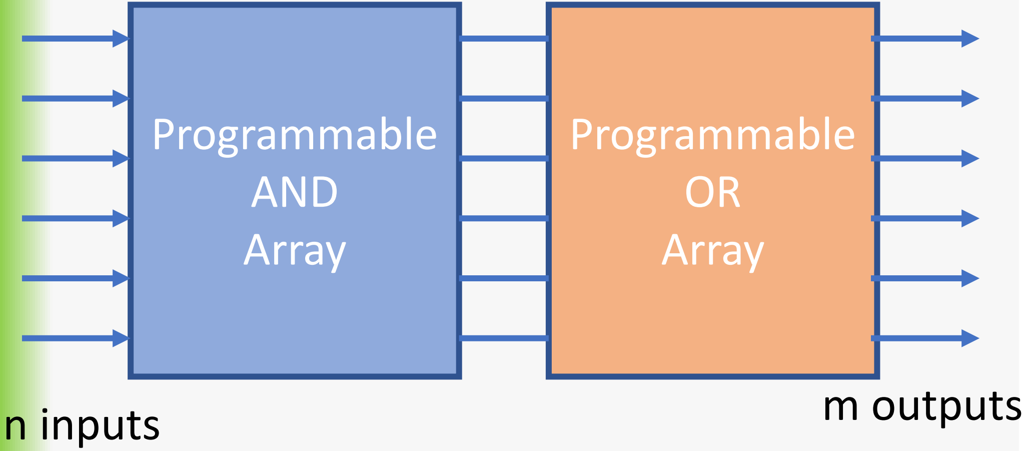
PAL – Programmable Array Logic

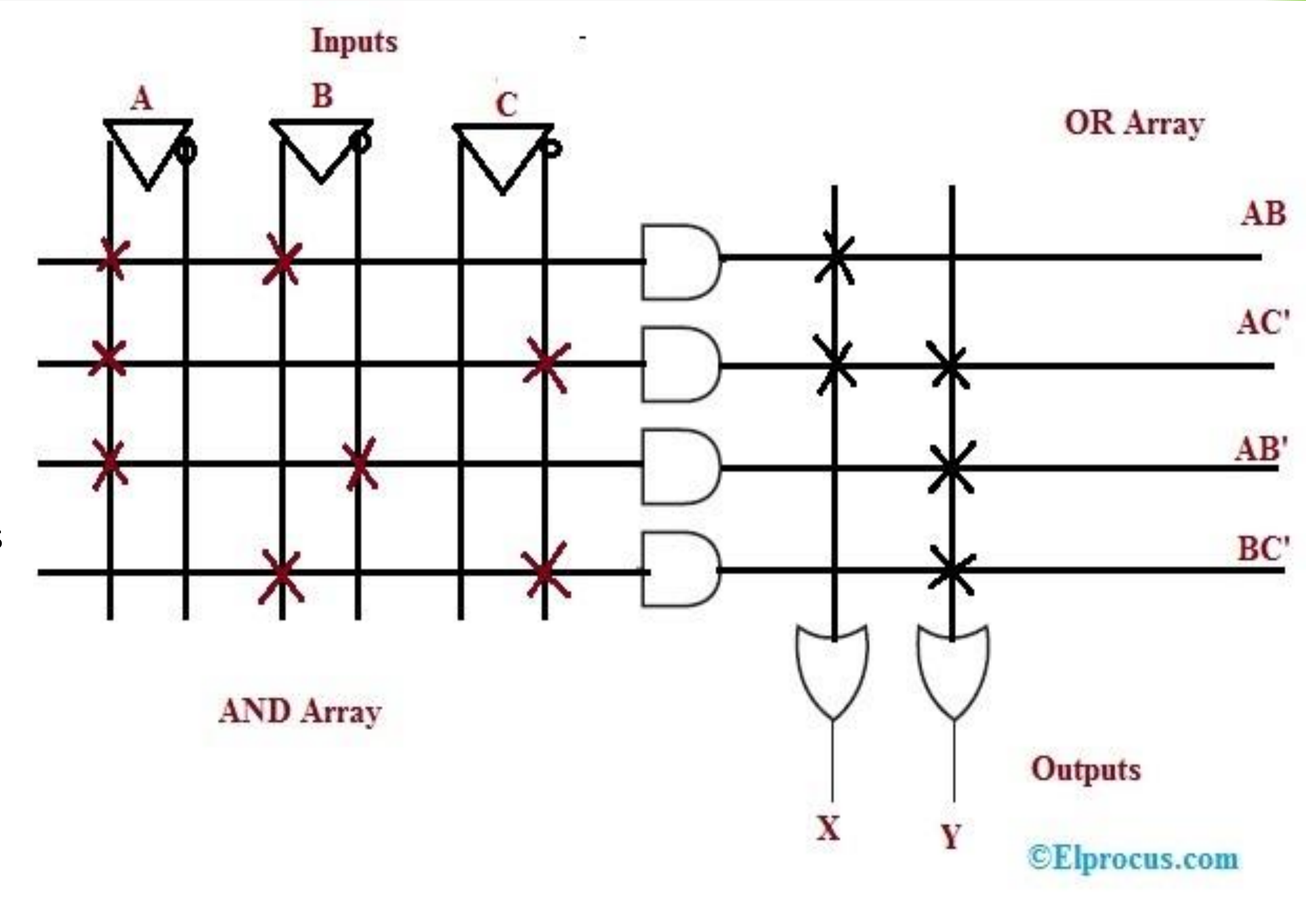




https://www.elprocus.com/what-are-pal-and-pla-design-and-differences/

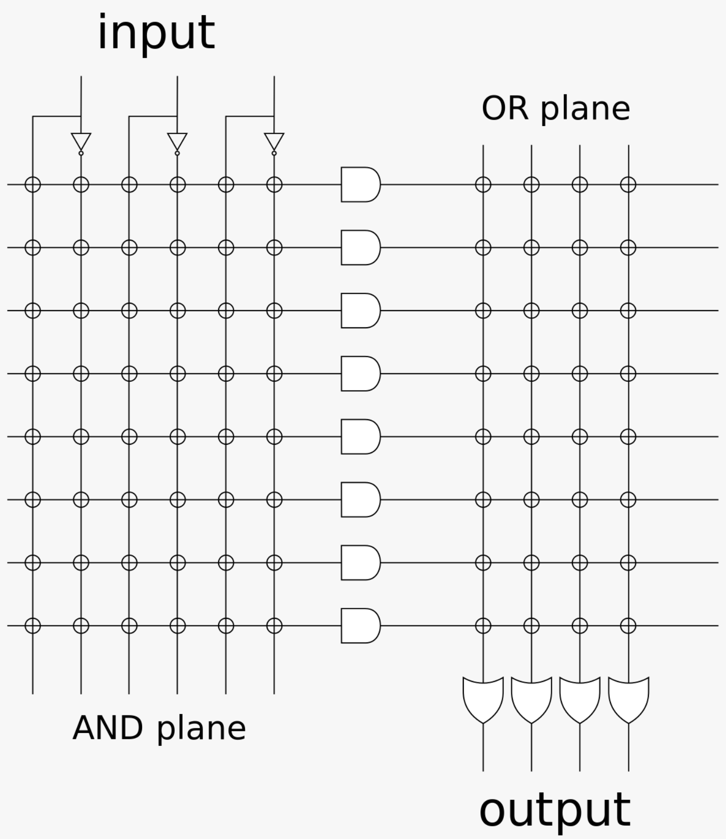
PLA – Programmable Logic Array

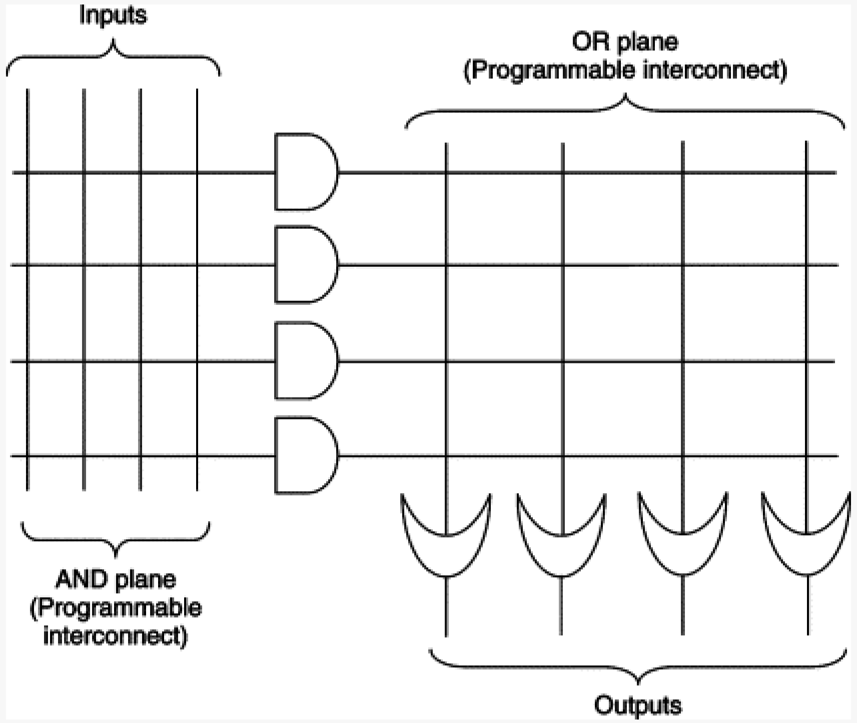


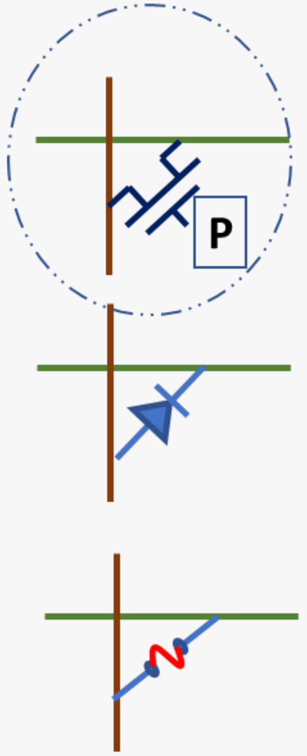


https://www.elprocus.com/what-are-pal-and-pla-design-and-differences/

For Exercise

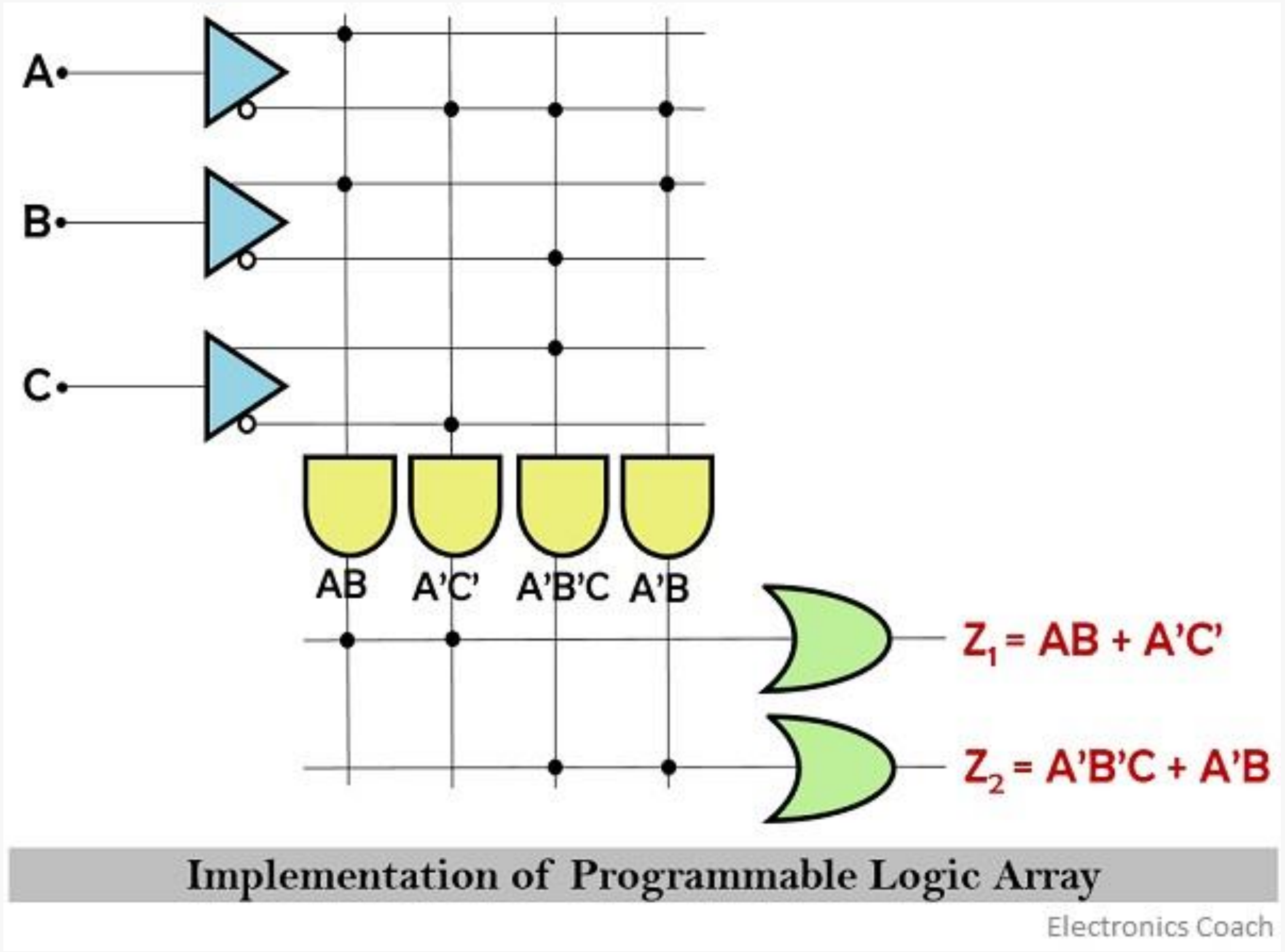






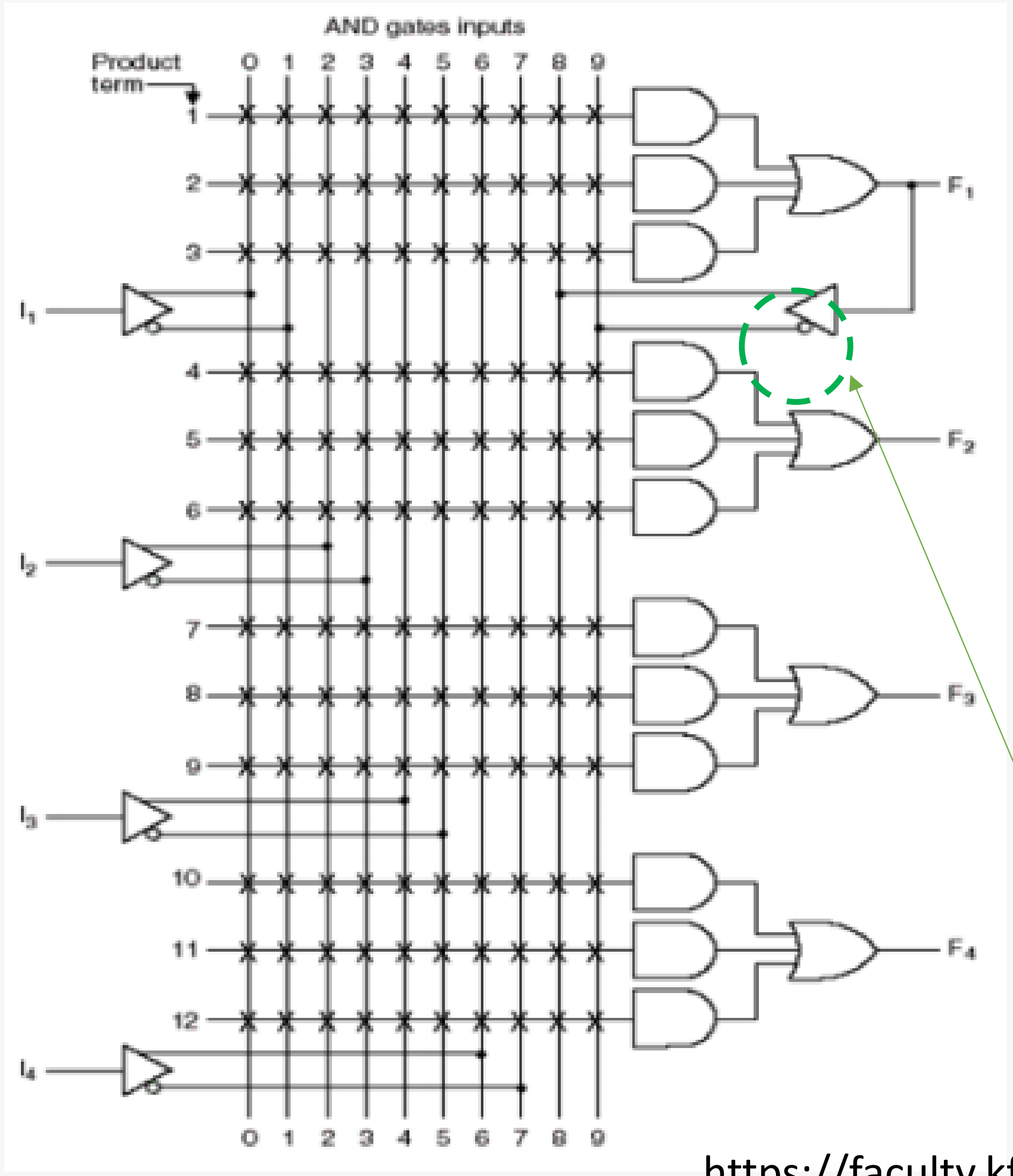
How to Configure?

Example of PLA



https://electronicscoach.com/programmable-logic-array.html

PAL – Programmable Array Logic Example



- PAL Device is a PLD with fixed OR array and a programmable AND array
- As only AND gates are programmable, PAL is easier to program

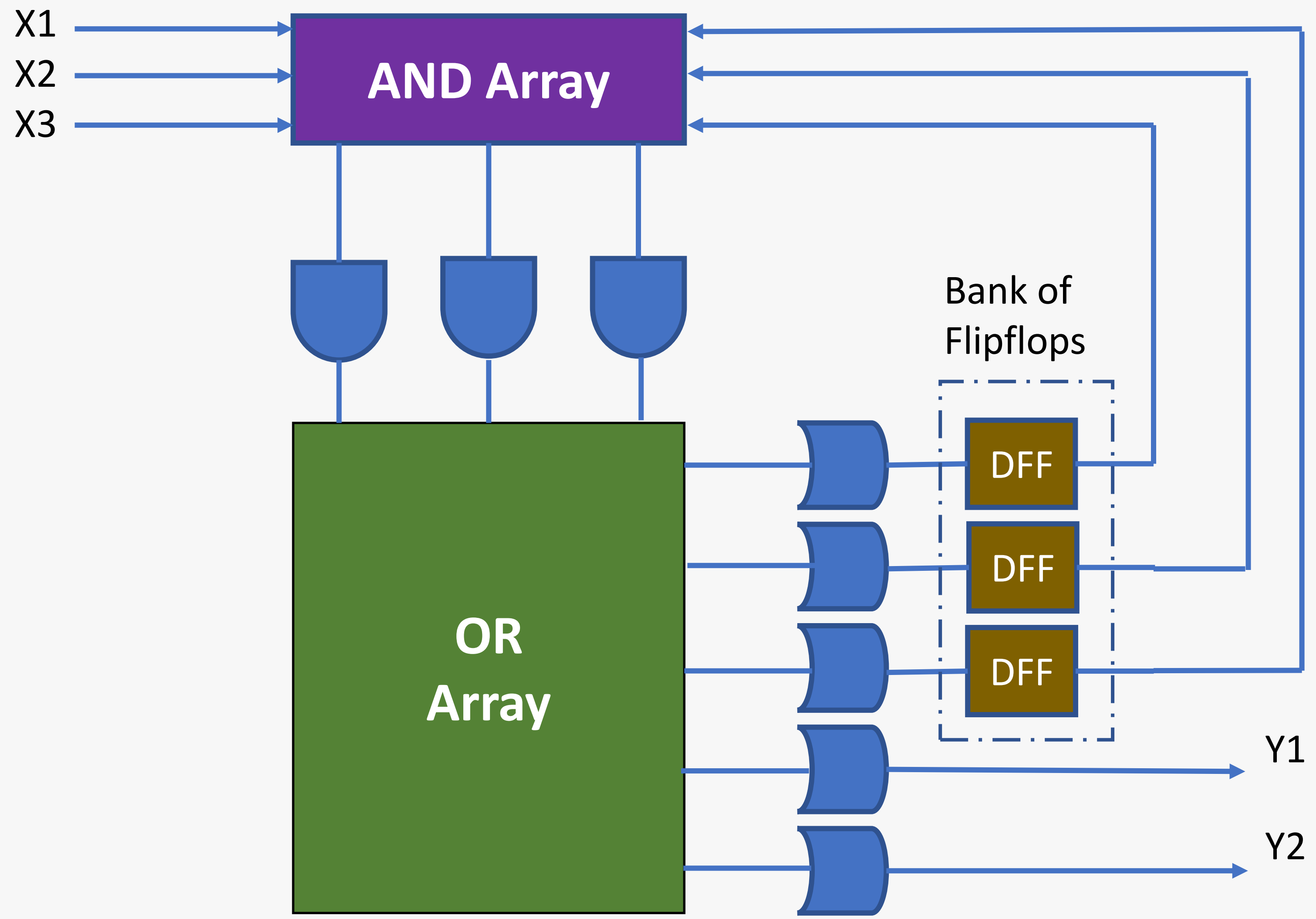
but it is less flexible compared to PLA devices

Example device:

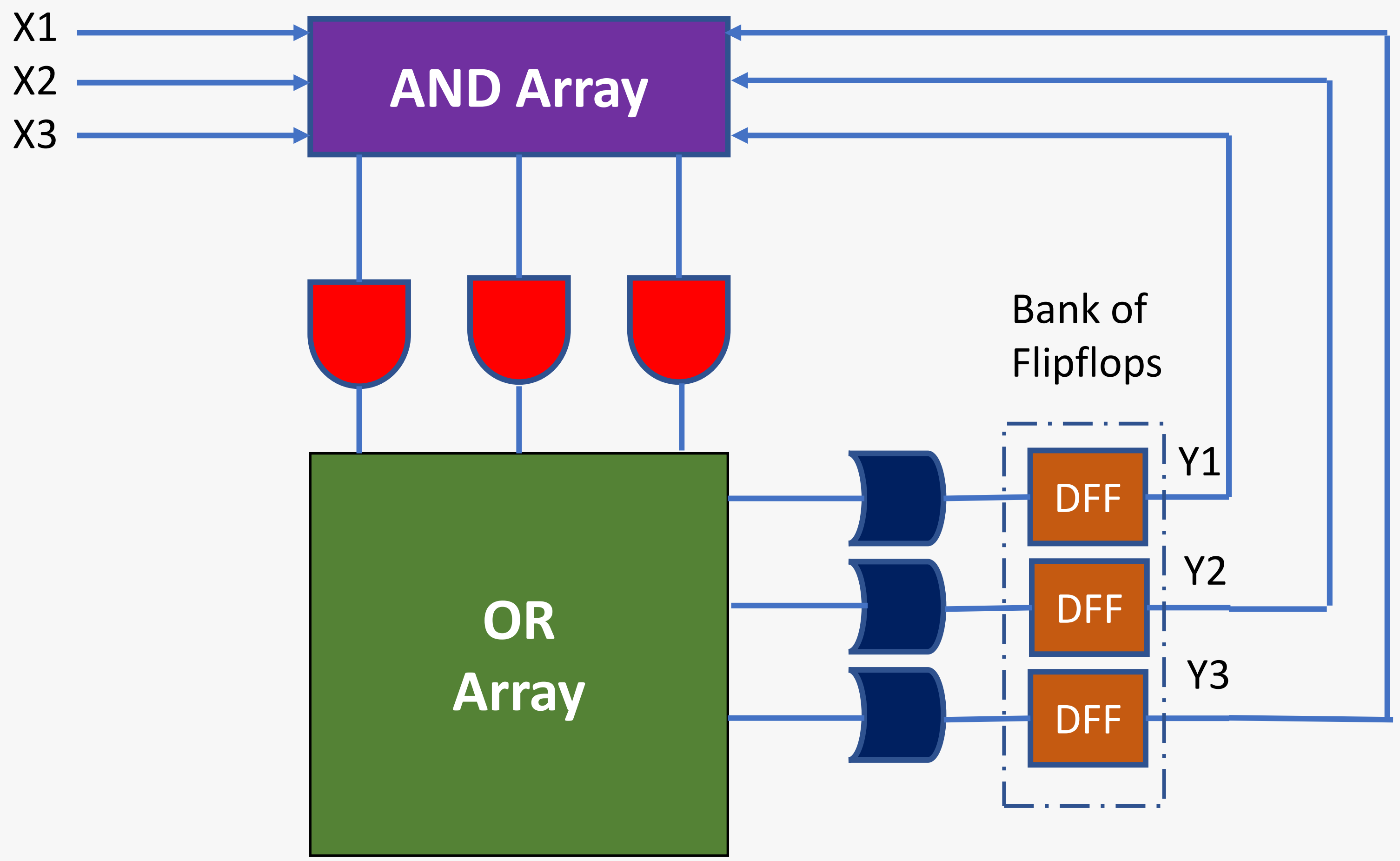
- This device has 4 inputs and 4 outputs.
- Each input has a buffer and inverter, output is from fixed OR gate
- The device has 4 sections: Each section comprises 3 wide AND-OR array, meaning three programmable AND arrays in each section
- Each AND gates has 10 programmable input connections
- One of the outputs (see F1) can be fed back to the inputs of AND gate through programmable connections

https://faculty.kfupm.edu.sa/COE/aimane/coe202/PLDs.pdf

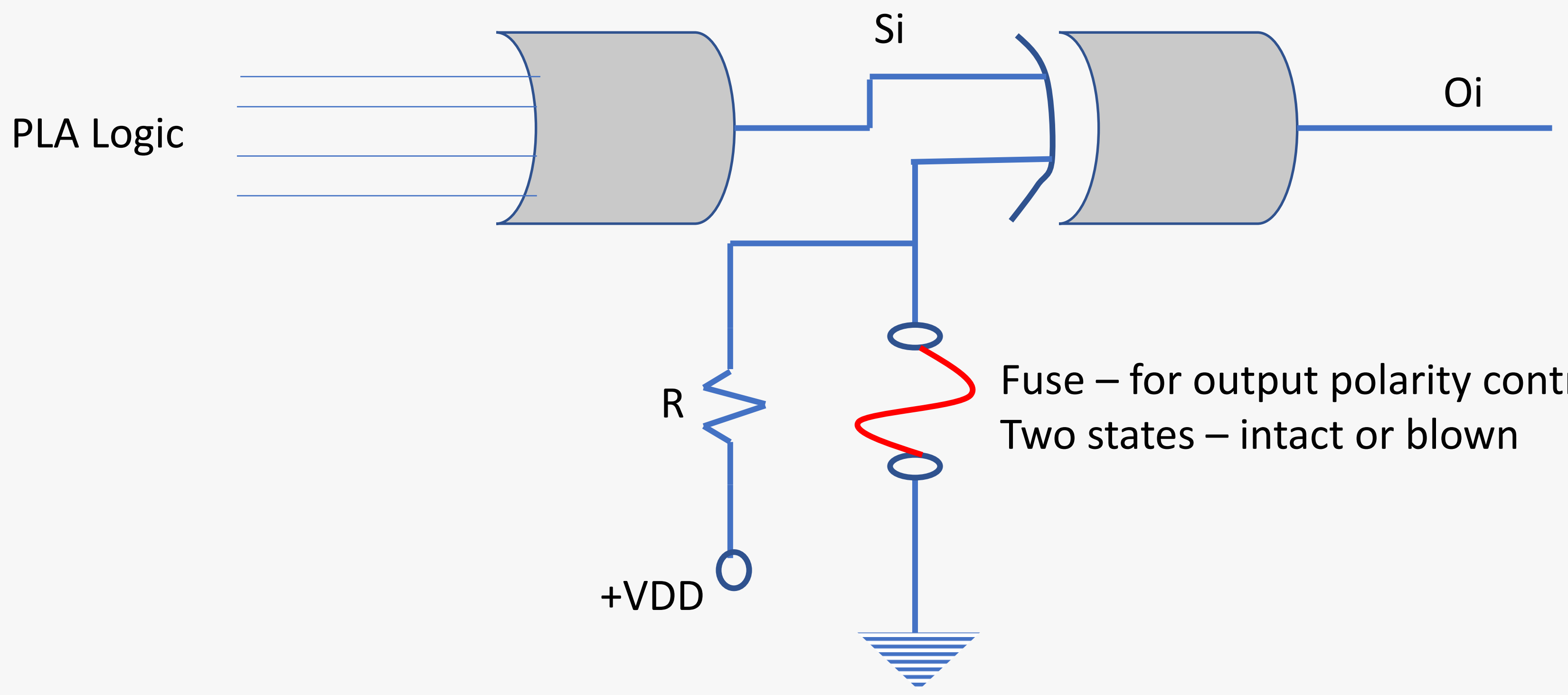
Sequential Circuits with PLD – Mealy Machine



Sequential Circuits with PLD – Moore Machine



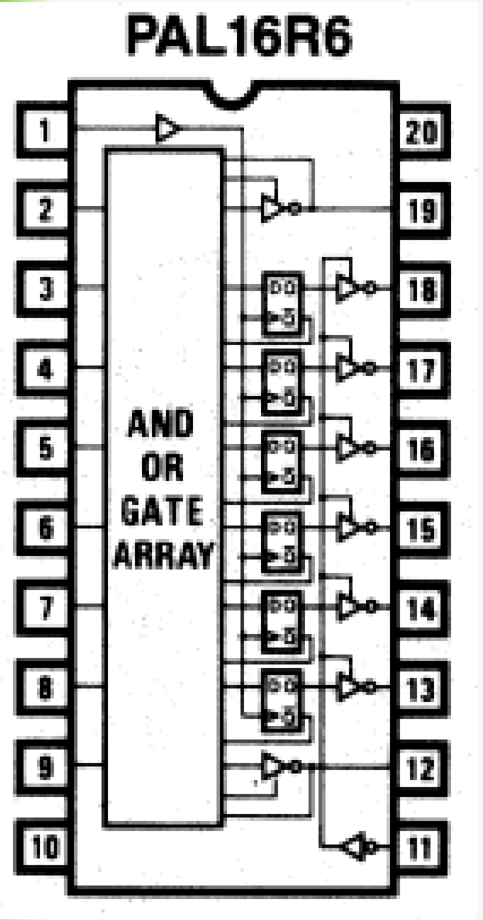
Output Polarity Control

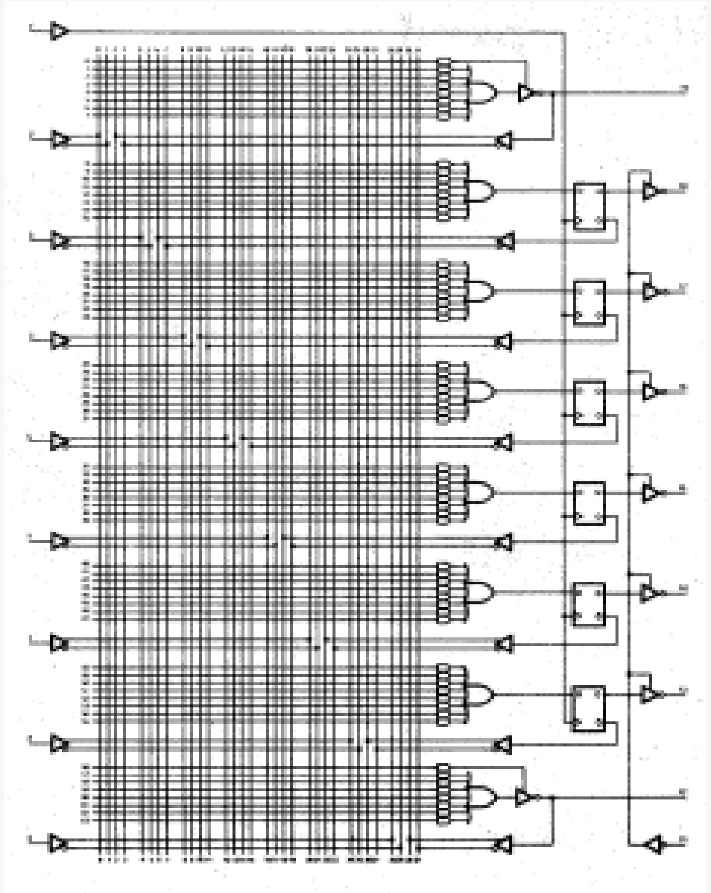


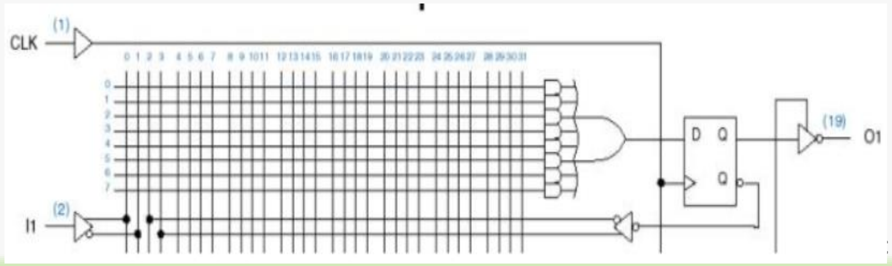
Fuse Intact → Oi = Si
Fuse Blown → Oi = Si’

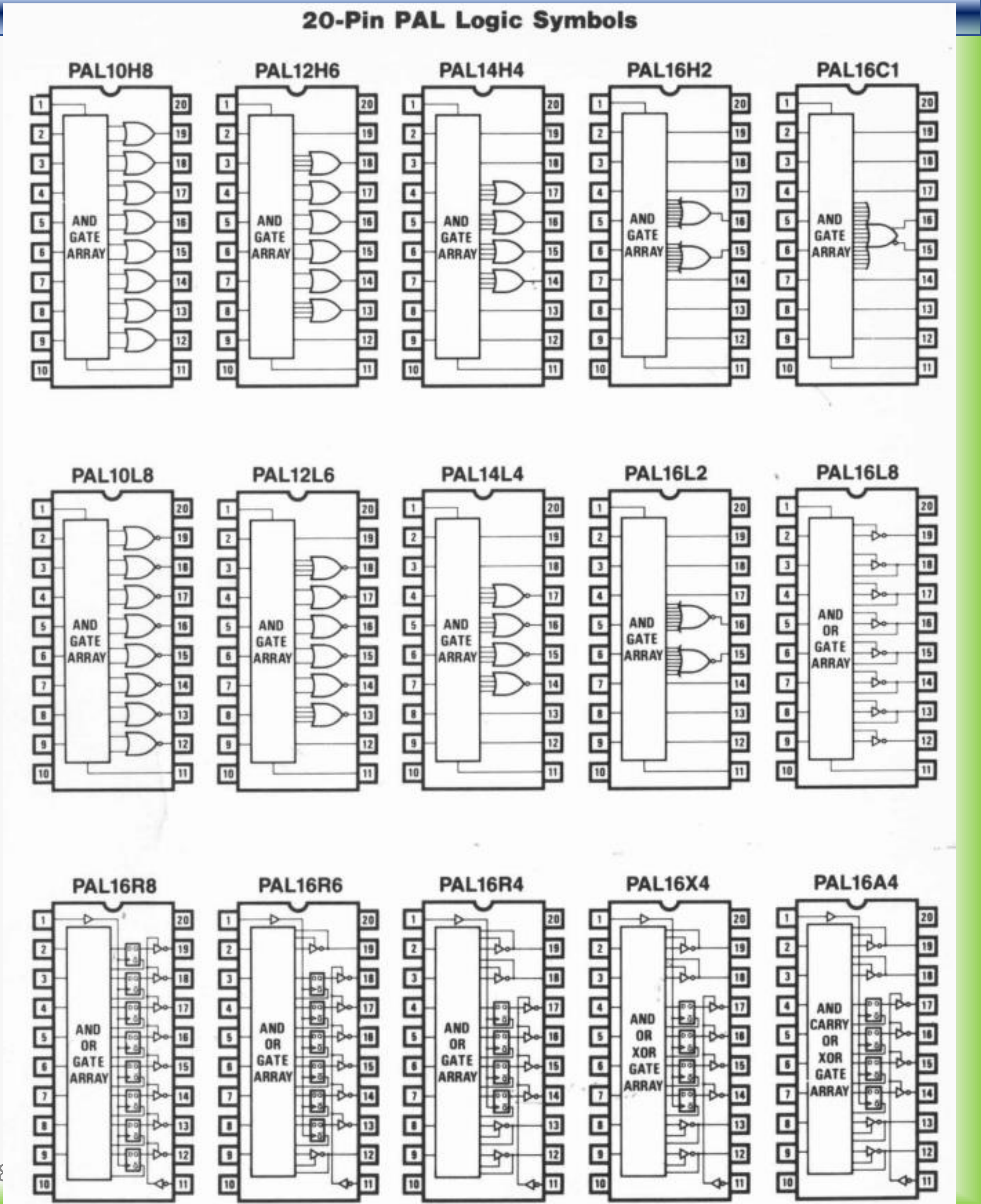
Fuse – for output polarity control
Two states – intact or blown

PLA Family Examples




PAL16R6 Logic Diagram


CLK
(1)
I1
(2)
D Q
Q
(19)
O1


20-Pin PAL Logic Symbols
PAL10H8
PAL12H6
PAL14H4
PAL16H2
PAL16C1
PAL10L8
PAL12L6
PAL14L4
PAL16L2
PAL16L8
PAL16R8
PAL16R6
PAL16R4
PAL16X4
PAL16A4
AND GATE ARRAY
AND OR GATE ARRAY
AND OR XOR GATE ARRAY
AND CARRY OR XOR GATE ARRAY