



02.07.20 11:22

CSCI 150

Introduction to Digital and Computer System Design

Midterm Review I



Jetic Gū
2020 Summer Semester (S2)

Overview

- Focus: Review
- Architecture: Combinational Logic Circuit
- Textbook v4: Ch1-4; v5: Ch1-3
- Core Ideas:
 1. Digital Information Representation (Lecture 1)
 2. Combinational Logic Circuits (Lecture 2)
 3. Combinational Functional Blocks, Arithmetic Blocks (Lecture 3)

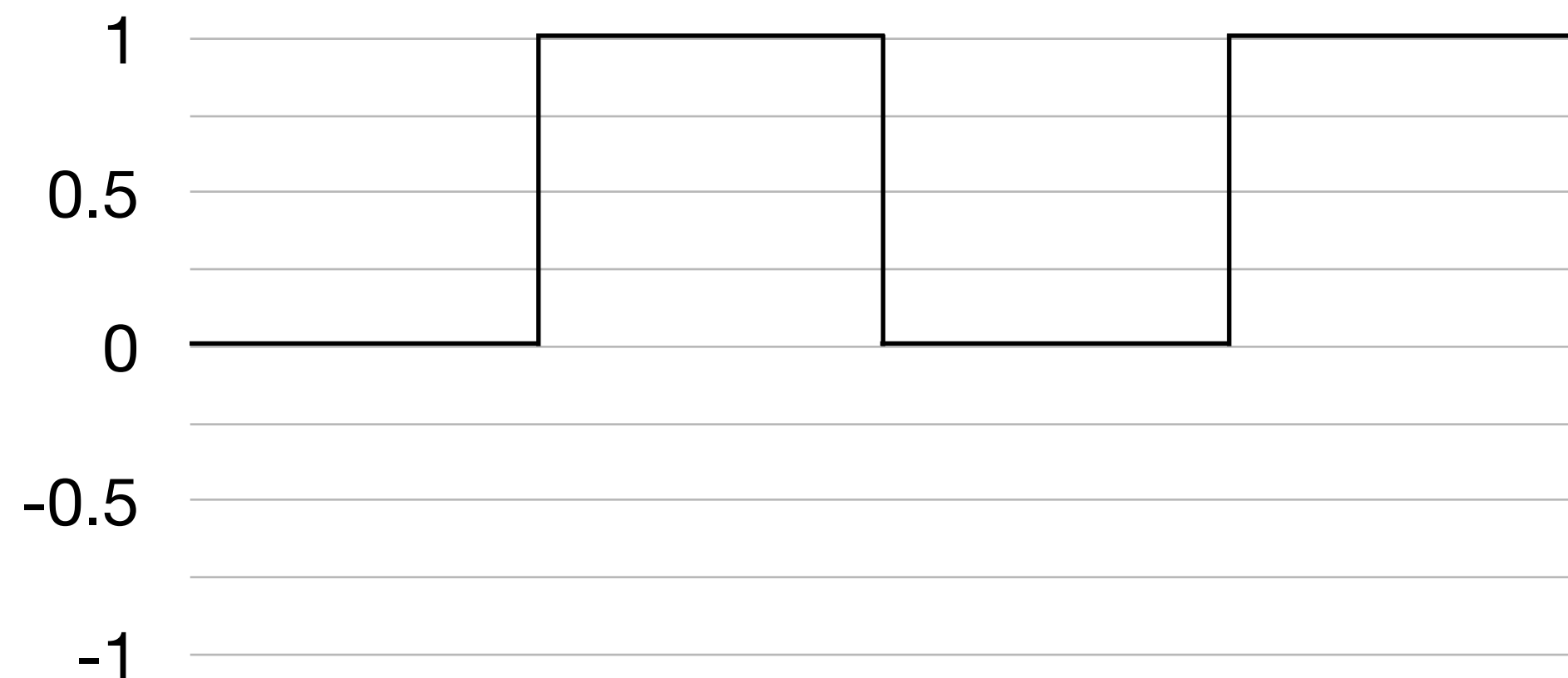
Lecture 1: Digital Information Representation

Analog vs Digital circuits; Modern computer architectures; Embedded systems;
Number Systems; Conversions;
Arithmetic Operations; Alphanumeric Codes

Analog vs Digital circuits

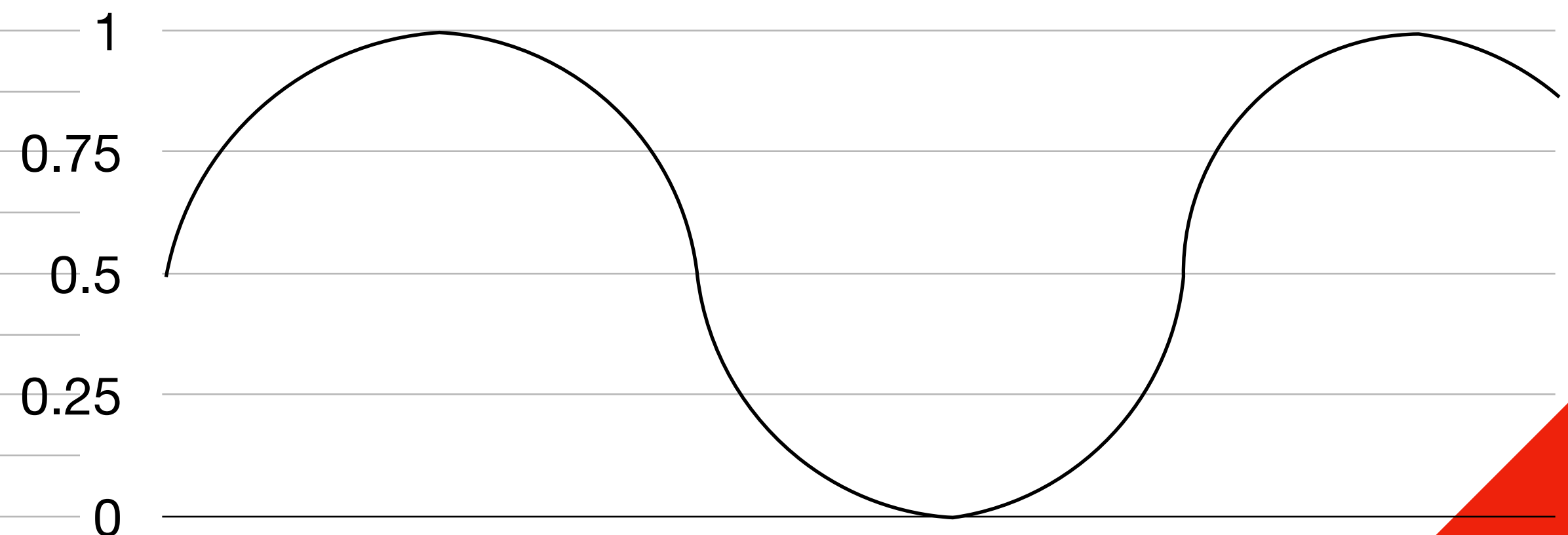
- Digital Circuits

- Process digital signals
- Current/Voltage represent discrete logical and numeric values



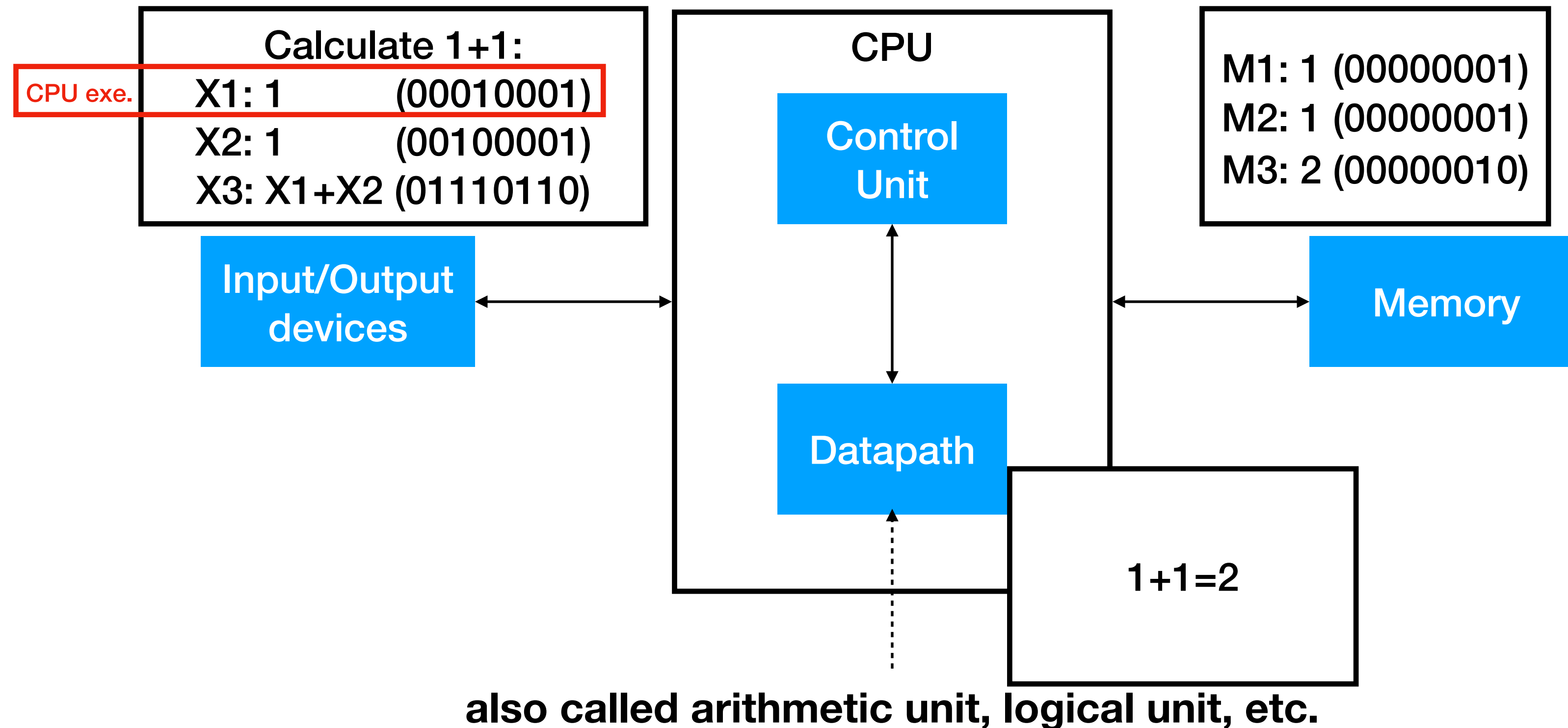
- Analog Circuits

- Process analog signals
- Current/Voltage vary continuously to represent information



Von Neumann Architecture

A very rough example



Computer

What's it like compared to a human?

- Input/Output devices
 - Interaction (Mouth, hands and feet, eyes, etc.)
- CPU + Memory
 - Processing information, thinking (Brain, short-term memory)
- Storage?
 - Part of I/O devices (Books, long-term memory)

Embedded Systems

- Similar to computers: processes information
- Difference
 - Function is usually simpler, and very very specific
 - Not programmable

Decimal System

7 2 4 . 0 5
2 1 0 -1 -2

- Numbers as strings of digits, each ranging from 0-9
- The decimal system is of base(radix) 10

Decimal System

$$\begin{array}{cccccc} 7 & 2 & 4 & . & 0 & 5 \\ 2 & 1 & 0 & -1 & -2 & \end{array}$$
$$= 7 \times 10^2 + 2 \times 10^1 + 4 \times 10^0 + 0 \times 10^{-1} + 5 \times 10^{-2}$$

- Numbers as strings of digits, each ranging from 0-9
- The decimal system is of base(radix) 10

Numbers of base N

- Default base: 10
- When there are numbers represented in different bases, attach base
 - Decimal: $754.05 \rightarrow (754.05)_{10}$
 - e.g. Base 5: $(432.1)_5 = ?$

$$= 4 \times 5^2 + 3 \times 5^1 + 2 \times 5^0 + 1 \times 5^{-1} = (117.2)_{10}$$

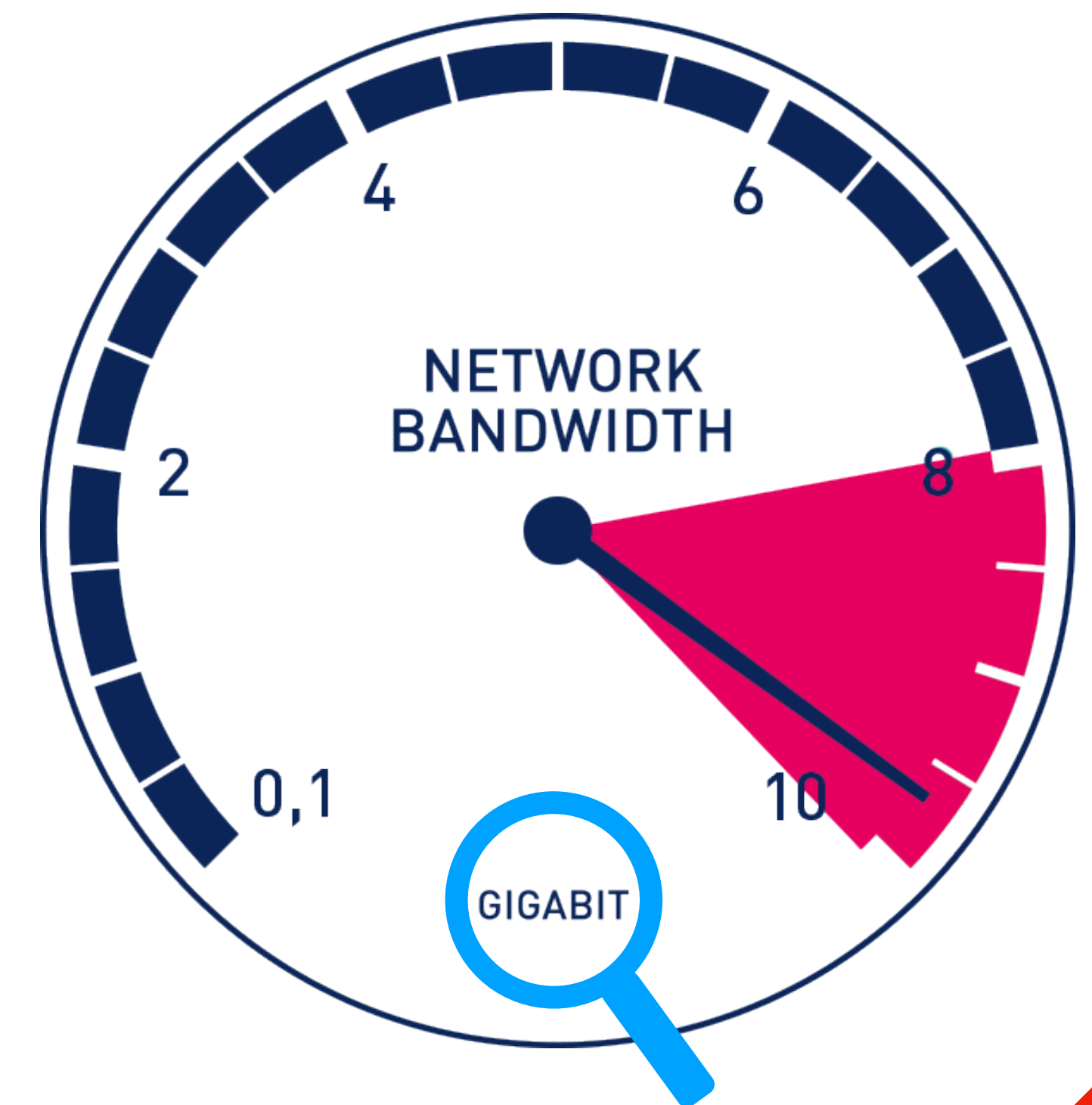
Binary Systems in Computers

- Every 8bit is called a Byte
- $1,024 = 2^{10}$ is called K (Kilo)
- $1,024 \times 1,024 = 2^{20}$ is called M (Mega)
- $1,024 \times 1,024 \times 1,024 = 2^{40}$ is called G (Giga)
- Tera, Peta, Exa, Zetta, Yotta

Binary Systems in Computers



- What is the difference between MBps and Mbps?
- MegaBytes per second vs MegaBits per second
- 8x difference!



Concept

Octal and Hexadecimal Systems

- Octal: base 8
 - digits: 0-7
- Hexadecimal: base 16
 - digits: 0-9, A-F (10-15)

Conversions

10	9	8	7	6	5	4	3	2	1
1024	512	256	128	64	32	16	8	4	2

- Binary-to-
Octal: 3bits per octal digit
Hexadecimal: 4bits per hexa digit
Decimal: use the chart
- Decimal-to-
Binary: use the chart
Oct/Hex: do binary first

Arithmetics

- The same as decimal (mostly)
-

$$\begin{array}{r} 0010 \\ +0011 \\ \hline 0101 \end{array} \quad \begin{array}{r} 0101 \\ -0011 \\ \hline 0010 \end{array}$$

Example (binary)

Arithmetics

OCTAL Multiplication

Octal

$$\begin{array}{r} \text{Octal} \\ 762 \\ \times 54 \\ \hline 4672 \\ 3710 \\ \hline 43772 \end{array}$$

$$5 \times 2 = 12$$

$$5 \times 6 + 1 = 37$$

$$5 \times 7 + 3 = 46$$

...

Decimal

$$10 = (12)_8$$

$$31 = (37)_8$$

$$38 = (46)_8$$

...

Signed & Unsigned Integers

- Unsigned 8bit:
 - $(11111111)_2 = 255$
- Signed 8bit (only in digital circuits):
 - $127 \rightarrow '01111111'$
 - $-127 \rightarrow '11111111'$

First digit:

- 0 for positive
- 1 for negative

10001111

(binary, 8bit, signed)

Signed & Unsigned Integers

- Unsigned 8bit integer: 0 - 255
 - Signed 8bit integer: -128 - 127
- Unsigned 32bit integer: 0 - 4,294,967,295
 - Signed 32bit integer: -2,147,483,648 - 2,147,483,647
- Unless otherwise specified, treat as unsigned

Binary Coded Decimal

- Decimal numbers, each digit represented in 4bit binary, but separately
- $185 = (0001\ 1000\ 0101)_{\text{BCD}} = (10111001)_2$
- Used in places where using decimals directly is more convenient, such as digital watches etc.

ASCII

- American Standard Code for Information Interchange
- Assign each character with a 8bit binary code (e.g. '0'-'9', 'A'-'Z', 'a'-'z')
- The first bit is always 0

Parity Code

- For error detection in data communication
 - e.g. resulting from packet loss or other forms of interference
- One parity bit for n-bits
 - An extra even parity bit: whether the number of 1s is not even
 - An extra odd parity: whether the number of 1s is not odd
 - Can be placed in any fixed position
 - Does it always work?

Parity Code

Original 7bits	with Even parity	with Odd parity
1000001	<u>0</u> 1000001	<u>1</u> 1000001
1010100	<u>1</u> 1010100	<u>0</u> 1010100

Circuits

- Circuits
 - Digital and Analog
- Integrated systems
 - Von Neumann computers
 - Embedded systems

Number Systems

- Number systems of base N
- Binary systems
- Octal and Hexadecimal systems
- Arithmetics

Number Systems in DC

- Bit, Byte, Representation ranges
- Signed and Unsigned Binary Integers
- BCD, ASCII, UTF8
- Parity bit

Digital to Analog Conversion

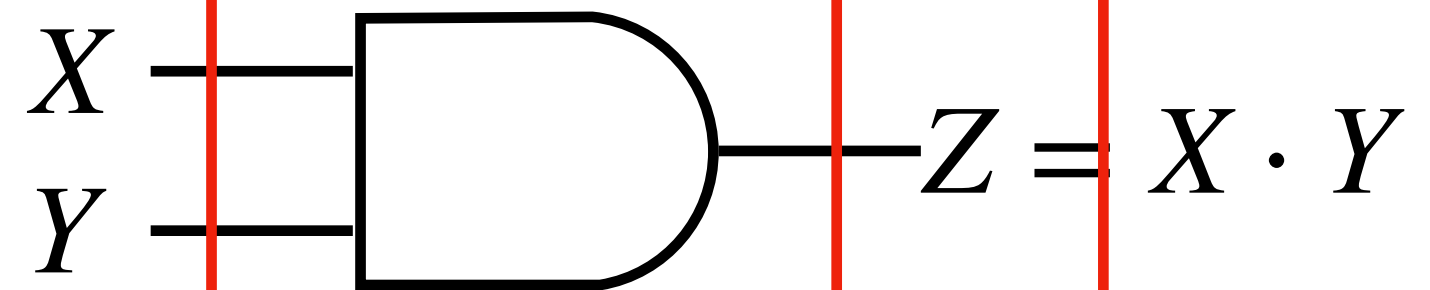
- Frequency: number of cycles per second
- Sample rate: number of samples per unit time
- Bitrate: number of bits per second

Lecture 2: Combinational Logic Circuits

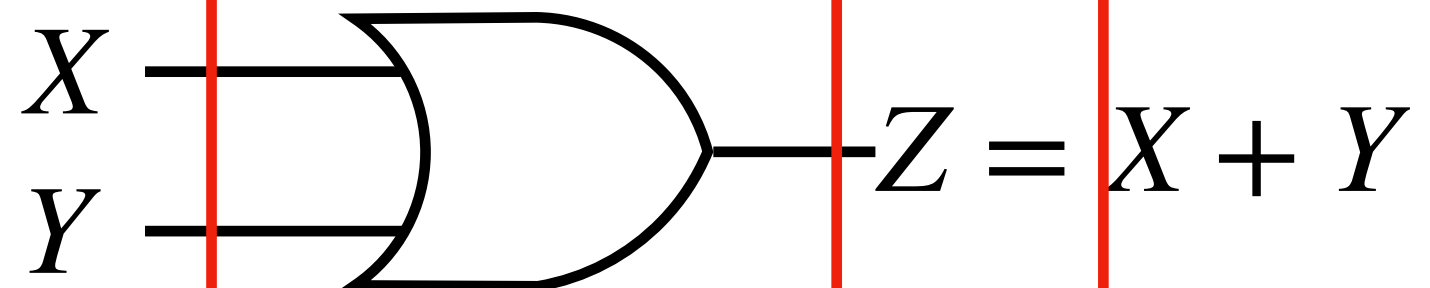
Logic Gates; Boolean Algebra; Minterm/Maxterm; K-
Map; Some Other Gate Types

First 3 Gates

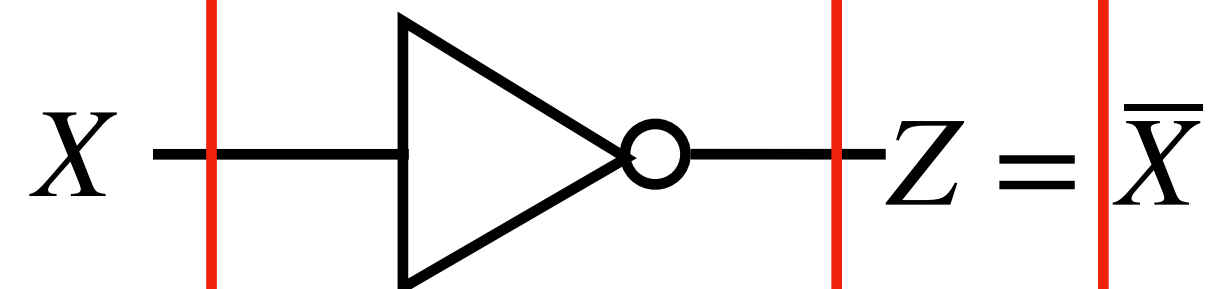
AND Gate



OR Gate



NOT Gate



Input

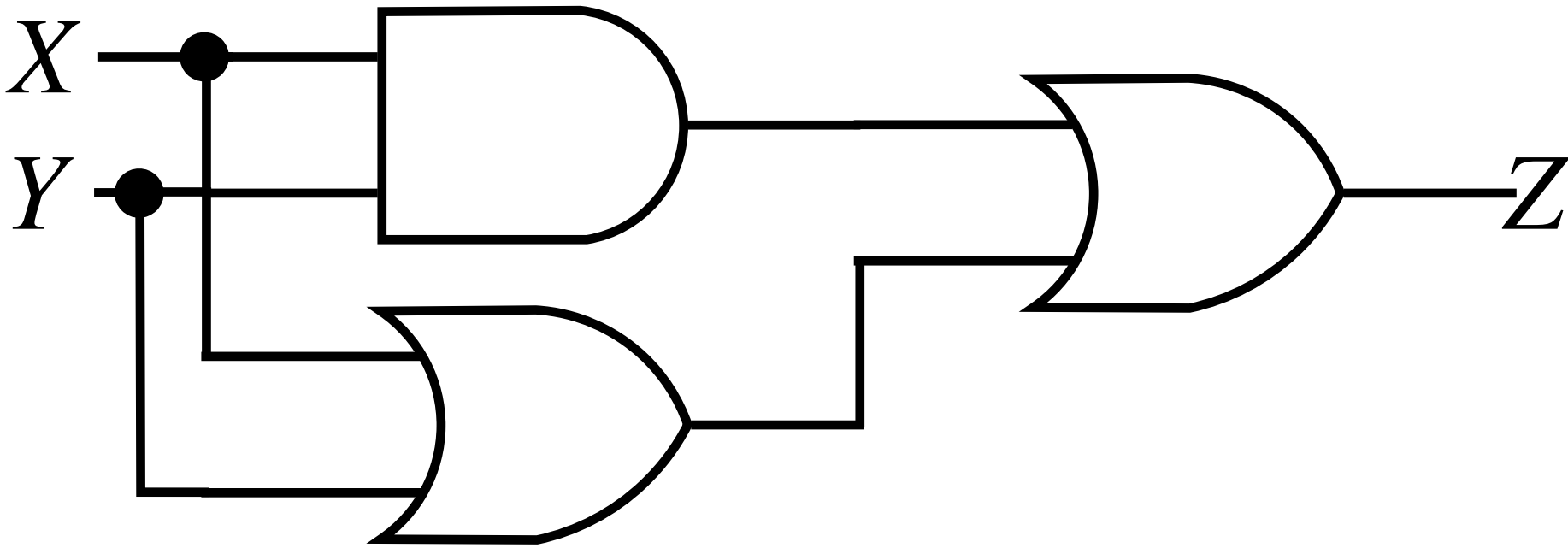
Output

Concept

Truth Table

Truth Table

X	Y	$Z = (X \cdot Y) + (X + Y)$
0	0	0
0	1	1
1	0	1
1	1	1



Basic Identities

- Boolean Algebra solving
 - **Identify** rules **applicable** to the expression
 - **Apply** rules that can help you **simplify** the expression
 - **Simplification**: reducing the number of variables and operators in an expression without changing its truth table values
 - **Atomic element**: an element that can't have the number of its variables and operators reduced any further

Basic Identities

1. $X + 0 = X$

2. $X \cdot 1 = X$

3. $X + 1 = 1$

4. $X \cdot 0 = 0$

5. $X + X = X$

6. $X \cdot X = X$

7. $X + \bar{X} = 1$

8. $X \cdot \bar{X} = 0$

9. $\bar{\bar{X}} = X$

Basic Identities

- Commutative

$$10. X + Y = Y + X$$

$$11. XY = YX$$

- Associative

$$12. X + (Y + Z) = (X + Y) + Z$$

$$13. X(YZ) = (XY)Z$$

- Distributive

$$14. X(Y + Z) = XY + XZ$$

$$15. X + (YZ) = (X + Y)(X + Z)$$

- DeMorgan's

$$16. \overline{X + Y} = \bar{X} \cdot \bar{Y}$$

$$17. \overline{\bar{X} \cdot \bar{Y}} = \bar{X} + \bar{Y}$$

Basic Identities

A. $X + XY = X$

B. $XY + X\bar{Y} = X$

C. $X + \bar{X}Y = X + Y$

D. $X(X + Y) = X$

E. $(X + Y)(X + \bar{Y}) = X$

F. $X(\bar{X} + Y) = XY$

Complementation

- \overline{F} : complement (invert) representation for a function F , obtained from an interchange of 1s to 0s and 0s to 1s for the values of F in the truth table
- Apply DeMorgan's Rule

$$16. \overline{X_1 + X_2 + \dots + X_n} = \overline{X_1} \cdot \overline{X_2} \cdot \dots \cdot \overline{X_n}$$

$$17. \overline{X_1 \cdot X_2 \cdot \dots \cdot X_n} = \overline{X_1} + \overline{X_2} + \dots + \overline{X_n}$$

Algebraic Manipulation

Difficulty: Simple

Simplify the following expressions

- $\bar{X} \cdot \bar{Y} + XYZ + \bar{X}Y$
- $X + Y(Z + \overline{X + Z})$

Algebraic Manipulation

Difficulty: Mid

Simplify the following expressions

- $\overline{W}X(\overline{Z} + \overline{Y}Z) + X(W + \overline{W}YZ)$
- $(AB + \overline{A}\overline{B})(\overline{C}\overline{D} + CD) + AC$

Algebraic Manipulation

Difficulty: Mid

Simplify the following expressions

- $\bar{A} \cdot \bar{C} + \bar{A}BC + \bar{B}C$

- $\overline{A + B + C} \cdot \overline{ABC}$

Algebraic Manipulation

Difficulty: Mid

Simplify the following expressions

- $AB\overline{C} + AC$
- $\overline{A} \cdot \overline{B}D + \overline{A} \cdot \overline{C}D + BD$

Algebraic Manipulation

Difficulty: HARDCORE

Prove the identity of each of the following Boolean equations

- $AB\bar{C} + B\bar{C} \cdot \bar{D} + BC + \bar{C}D = B + \bar{C}D$
- $WY + \bar{W}Y\bar{Z} + WXZ + \bar{W}X\bar{Y} = WY + \bar{W}X\bar{Z} + \bar{X}Y\bar{Z} + X\bar{Y}Z$
- $A\bar{D} + \bar{A}B + \bar{C}D + \bar{B}C = (\bar{A} + \bar{B} + \bar{C} + \bar{D})(A + B + C + D)$

Standard Forms

- Equivalent expressions can be written in a variety of ways
Standard forms: typical such ways that incorporates some **unique characteristics** -> **simplify the implementation** of these designs
- **Product terms** (AND terms): e.g. $\bar{X}YZ$
Literals with inverts connected through only AND operators
- **Sum terms** (OR terms): e.g. $X + \bar{Y} + Z$
Literals with inverts connected through only OR operators

Minterms and Maxterms

- Minterm**

Product term; Contains **all variables**; Has only **one Positive row** in the truth table

	X	Y	$m_0 = \bar{X}\bar{Y}$	$m_1 = \bar{X}Y$	$m_2 = X\bar{Y}$	$m_3 = XY$
(00) ₂ =0	0	0	1	0	0	0
(01) ₂ =1	0	1	0	1	0	0
(10) ₂ =2	1	0	0	0	1	0
(11) ₂ =3	1	1	0	0	0	1

Minterms and Maxterms

- Maxterm**

Sum term; Contains **all variables**; Has only **one Negative row** in the truth table

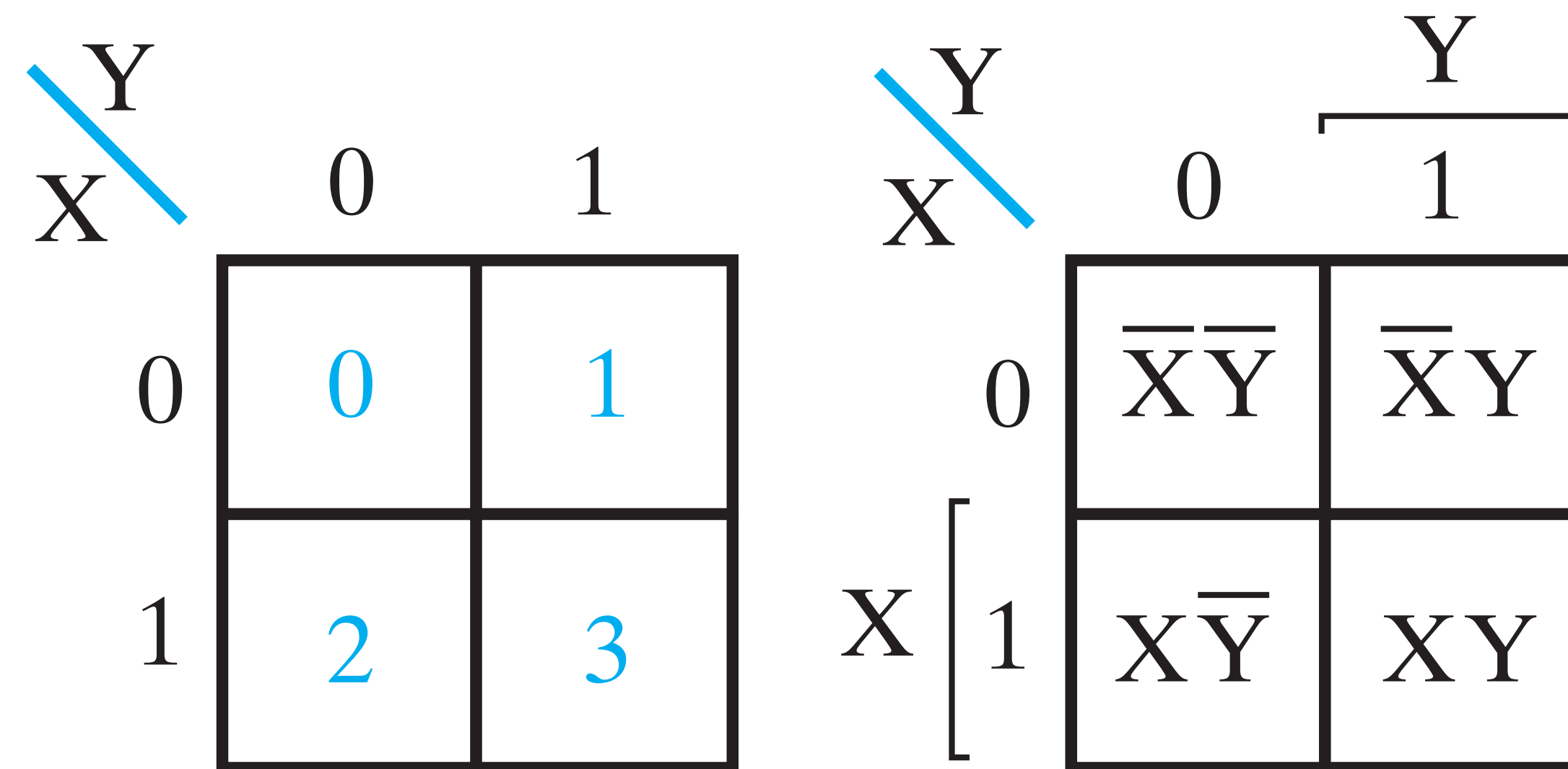
$$M_i = \overline{m_i}$$

X	Y	$M_0 = X + Y$	$M_1 = X + \bar{Y}$	$M_2 = \bar{X} + Y$	$M_3 = \bar{X} + \bar{Y}$
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Minterms and Maxterms

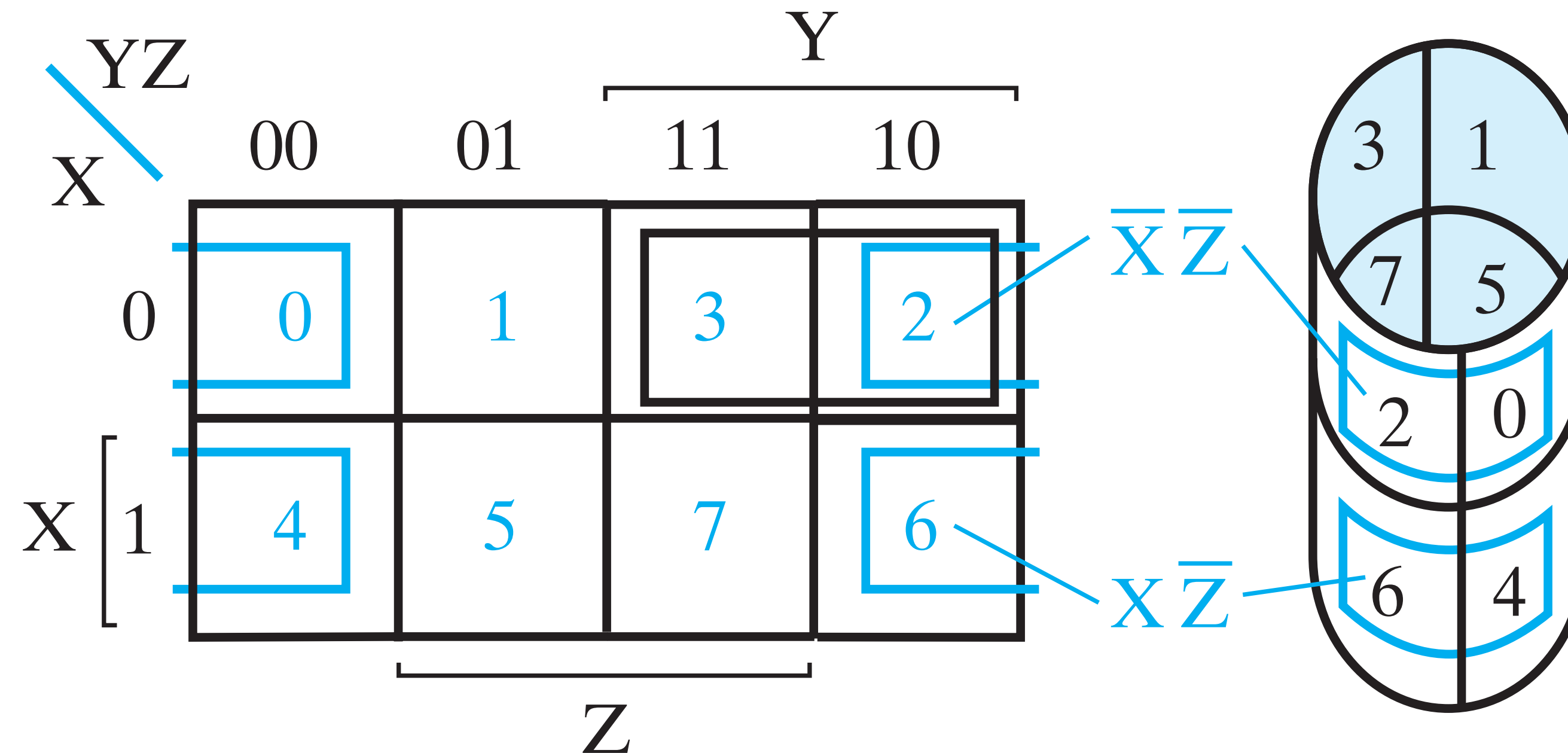
- e.g. $M_3 = X + \bar{Y} + \bar{Z} = \overline{\bar{X}Y\bar{Z}} = \overline{m_3}$
- Sum of Minterms
 - e.g. $F = \bar{X}\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + XYZ = m_0 + m_2 + m_5 + m_7$
 $= \Sigma m(0,2,5,7)$
- Product of Maxterm
 - e.g. $F = (X + Y + Z)(X + \bar{Y} + Z)(\bar{X} + Y + \bar{Z})(\bar{X} + \bar{Y} + \bar{Z})$
 $= M_0M_2M_5M_7$
 $= \Pi M(0,2,5,7)$

Two Variable Maps



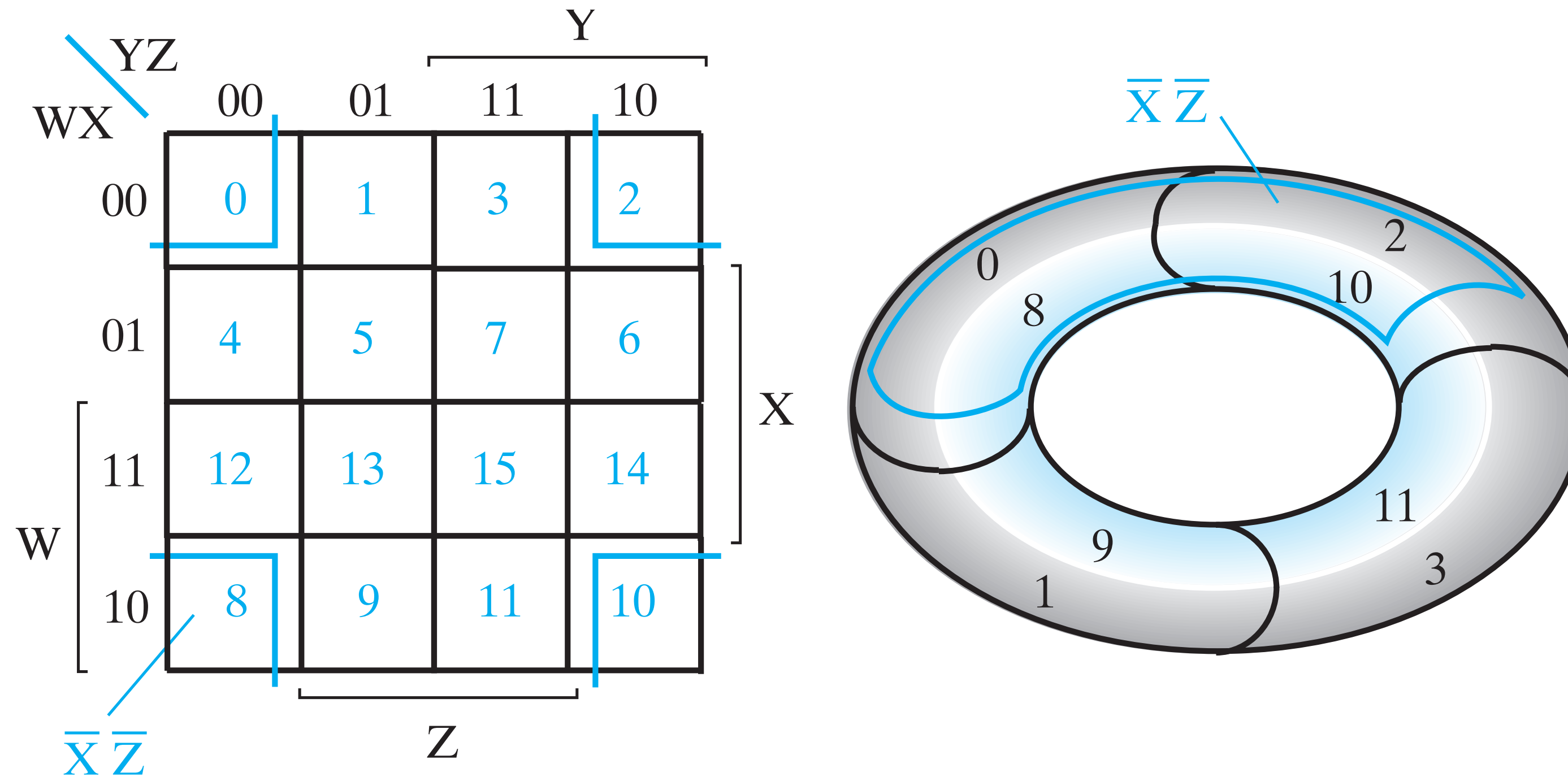
- Number of squares in each map is equal to the number of minterms for the same number of variables, light blue digit above is the index (of minterm)
- Two squares are adjacent if they only differ in one variable
- Binary value inside at each position indicates the truth table value for that term

Three Variable Maps



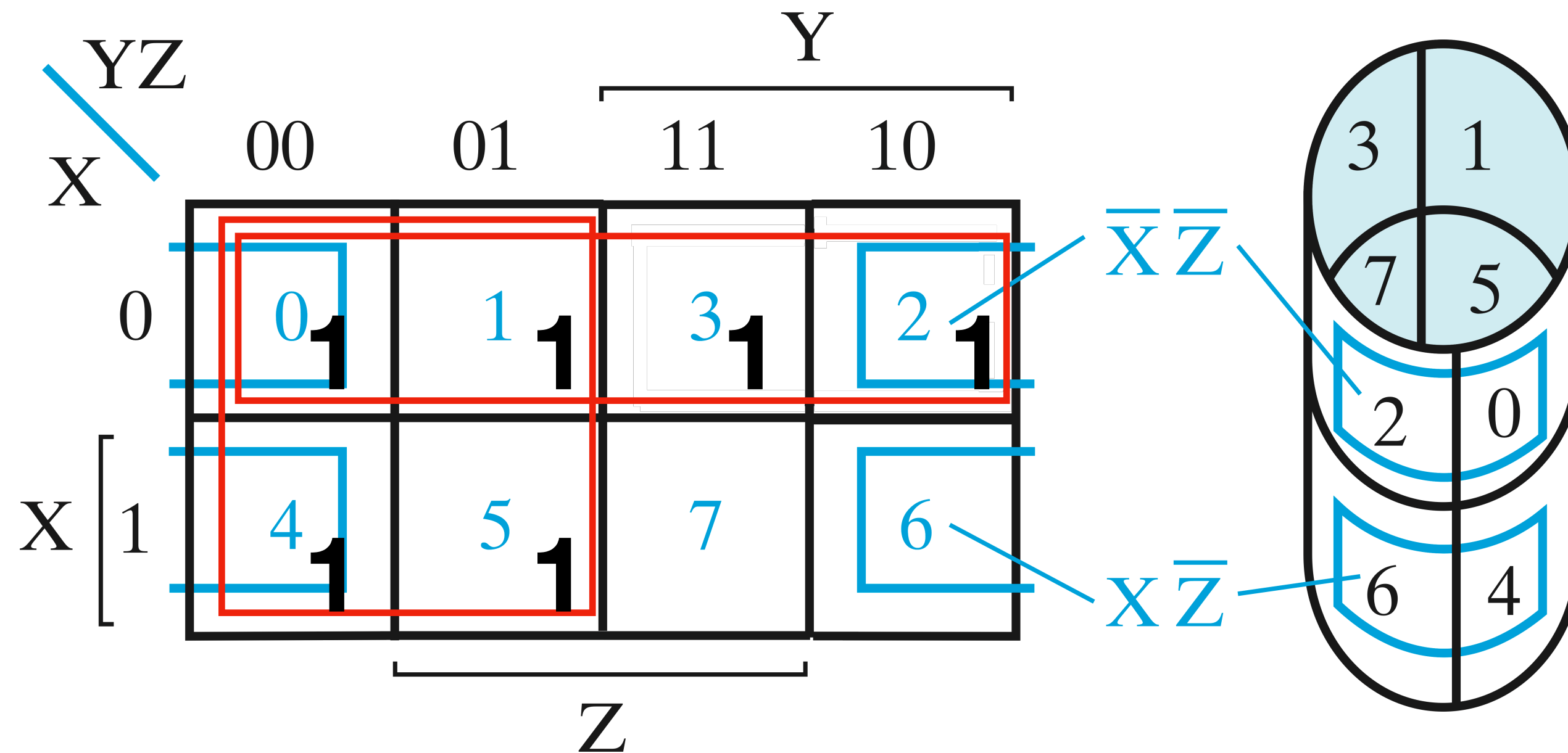
- Number of squares in each map is equal to the number of minterms for the same number of variables, light blue digit above is the index (of minterm)
- Two squares are adjacent if they only differ in one variable
- Binary value inside at each position indicates the truth table value for that term

Four Variable Maps



- Number of squares in each map is equal to the number of minterms for the same number of variables, light blue digit above is the index (of minterm)
- Two squares are adjacent if they only differ in one variable
- Binary value inside at each position indicates the truth table value for that term

K Map Optimisation



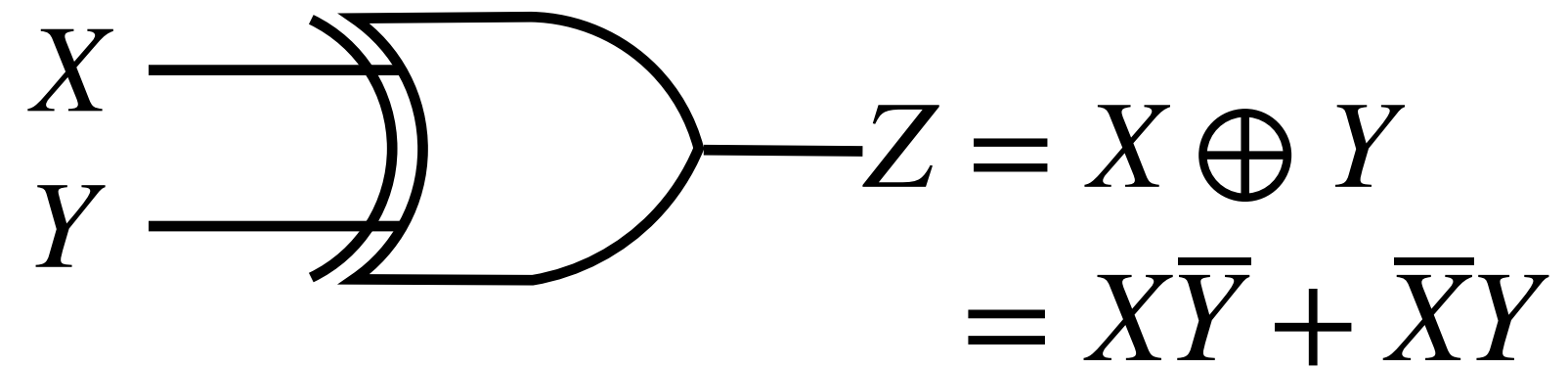
$$F(X, Y, Z) = \sum m(0, 1, 2, 3, 4, 5)$$

$$= \bar{X} + \bar{Y}$$

- Step 1: Enter the values
- Step 2: Identify the set of **largest** rectangles in which **all values are 1**, covering **all 1s**
- Step 3: **Read off** the selected rectangles. If rectangle has odd length edges (excluding 1), split

XOR Gate

XOR Gate
Exclusive-OR



- $X \oplus 0 = X$
- $X \oplus 1 = \bar{X}$
- $X \oplus X = 0$
- $X \oplus \bar{X} = 1$
- $X \oplus \bar{Y} = \overline{X \oplus Y}$
- $\bar{X} \oplus Y = \overline{X \oplus Y}$

XOR Truth Table

X	Y	$Z = X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

XOR Gate

- $X \oplus 0 = X$

- $X \oplus X = 0$

- $X \oplus \bar{Y} = \overline{X \oplus Y}$

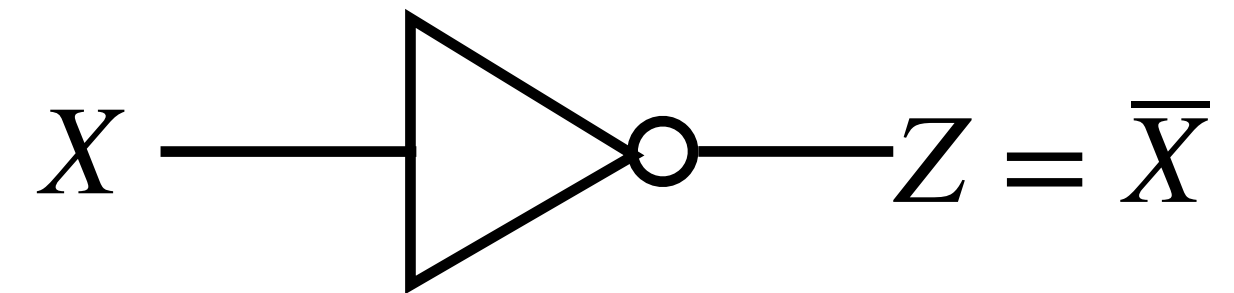
- $X \oplus 1 = \bar{X}$

- $X \oplus \bar{X} = 1$

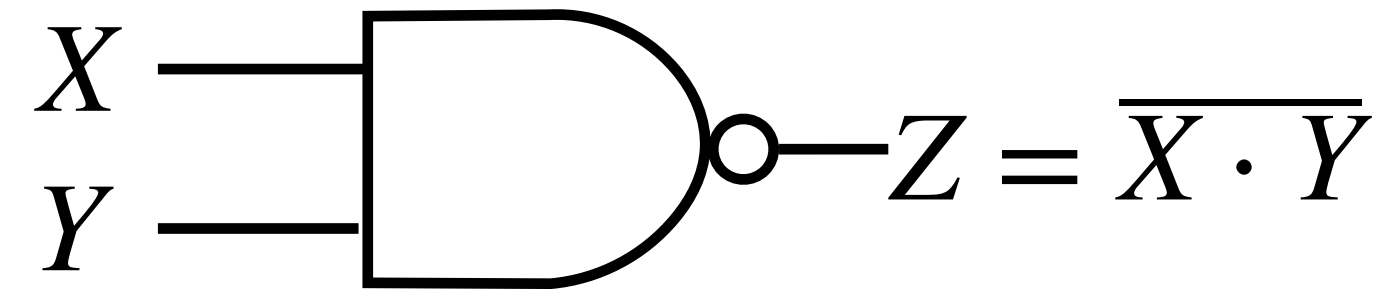
- $\bar{X} \oplus Y = \overline{X \oplus Y}$

N-Gates

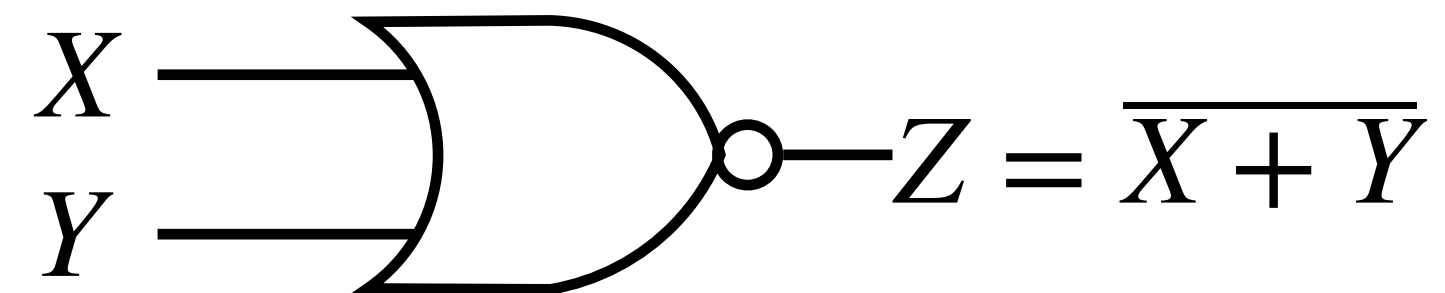
NOT Gate



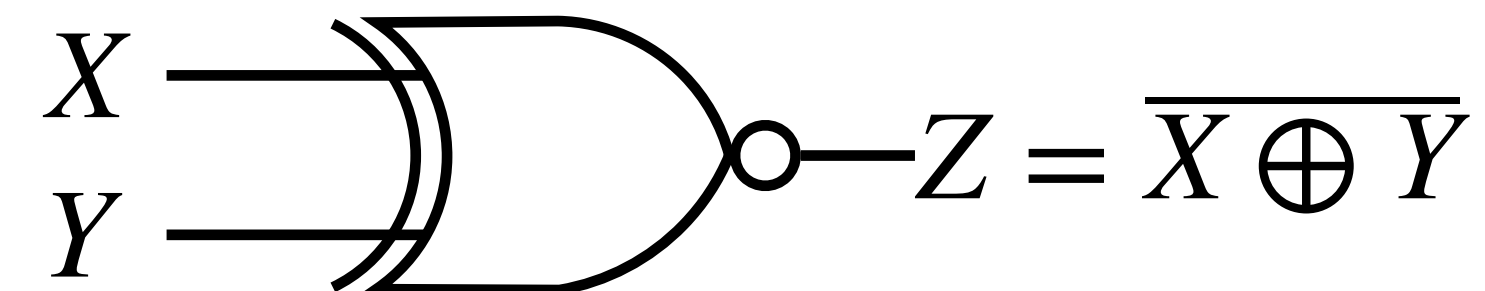
NAND Gate



NOR Gate



XNOR Gate



Boolean Algebra

I. AND, OR, NOT Operators and Gates

- Simple digital circuit implementation
- Algebraic manipulation using Binary Identities

II. Standard Forms

- Minterm & Maxterm
- Sum of Products & Product of Sums

III. Optimisation Using K-Map (For 2,3,4 Variables)

IV. XOR, NAND, NOR, XNOR

Lecture 3: Combinational Logic Design

5 Steps Systematic Design Procedures; Functional
Blocks; Decoder, Enabler, Multiplexer; Arithmetic Blocks

Systematic Design Procedures

1. **Specification:** Write a specification for the circuit
2. **Formulation:** Derive relationship between inputs and outputs of the system
e.g. using truth table or Boolean expressions
3. **Optimisation:** Apply optimisation, minimise the number of logic gates and literals required
4. **Technology Mapping:** Transform design to new diagram using available implementation technology
5. **Verification:** Verify the correctness of the final design in meeting the specifications

Hierarchical Design

- "divide-and-conquer"
- Circuit is broken up into individual functional pieces (blocks)
 - Each block has explicitly defined **Interface** (I/O) and **Behaviour**
 - A single block can be **reused** multiple times to simplify design process
 - If a single block is too complex, it can be **further divided into smaller blocks**, to allow for easier designs

Value-Fixing, Transferring, and Inverting

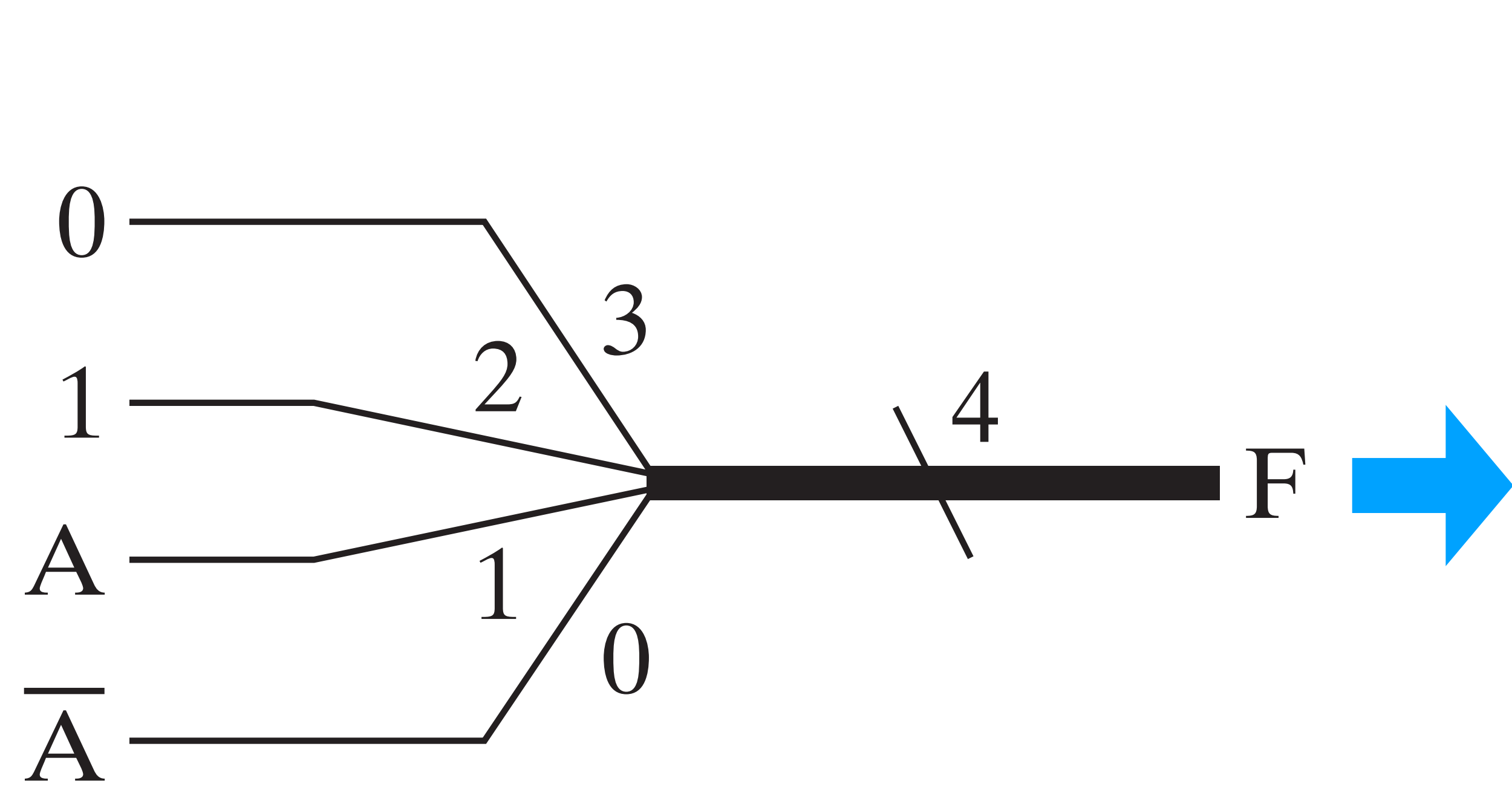
- ① **Value-Fixing:** giving a constant value to a wire
 - $F = 0; F = 1;$
- ② **Transferring:** giving a variable (wire) value from another variable (wire)
 - $F = X;$
- ③ **Inverting:** inverting the value of a variable
 - $F = \bar{X}$

Vector Denotation

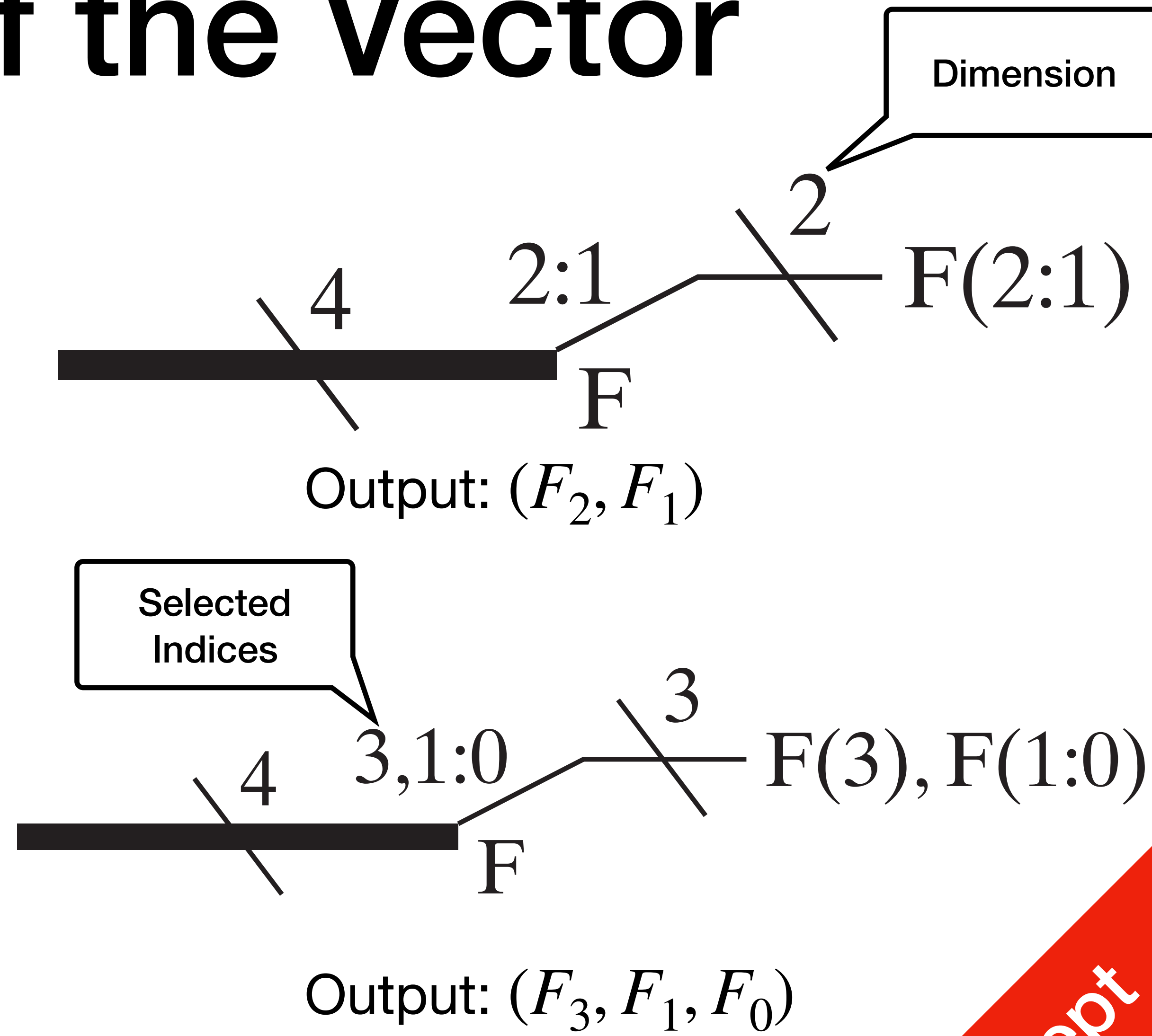
④ Multiple-bit Function

- Functions we've seen so far has only one-bit output: 0/1
- Certain functions may have n -bit output
- $F(n - 1 : 0) = (F_{n-1}, F_{n-2}, \dots, F_0)$, each F_i is a one-bit function
- Curtain Motor Control Circuit: $F = (F_{\text{Motor}_1}, F_{\text{Motor}_2}, F_{\text{Light}})$

Taking part of the Vector



④ Multiple-bit Function



Concept

Enabler

⑤ Enabler

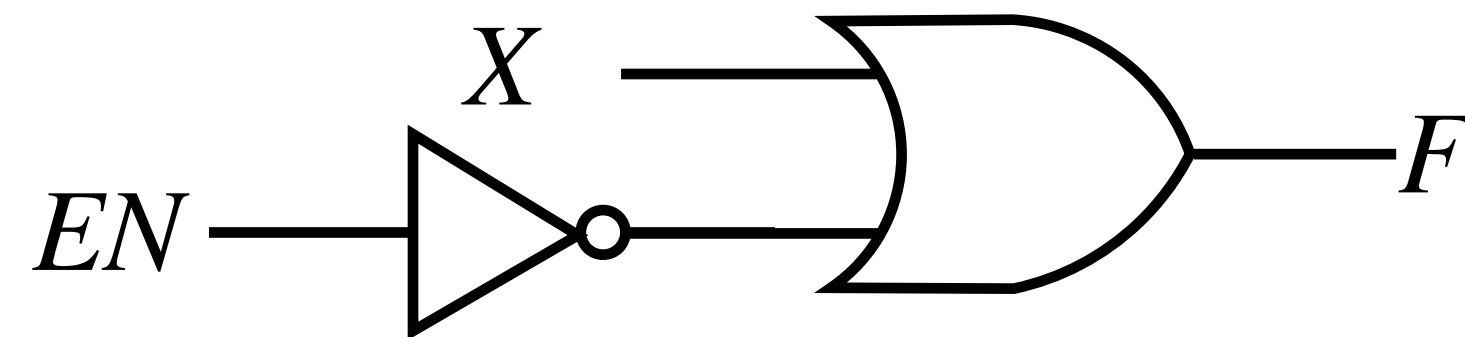
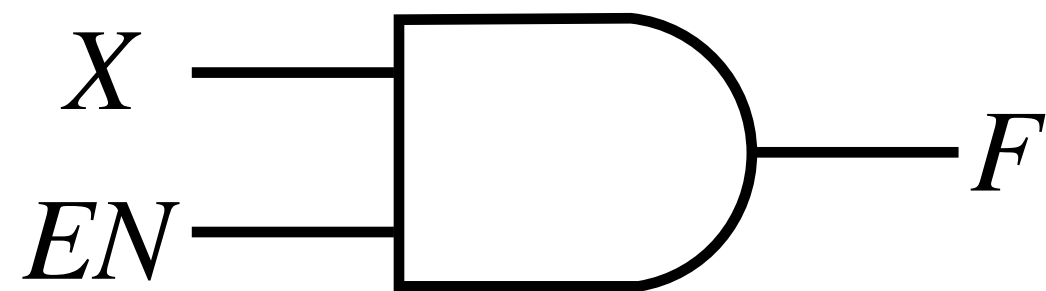
- Transferring function, but with an additional EN signal acting as switch

EN	X	F
0	X	0
1	0	0
1	1	1

Enabler

⑤ Enabler

- Transferring function, but with an additional EN signal acting as switch

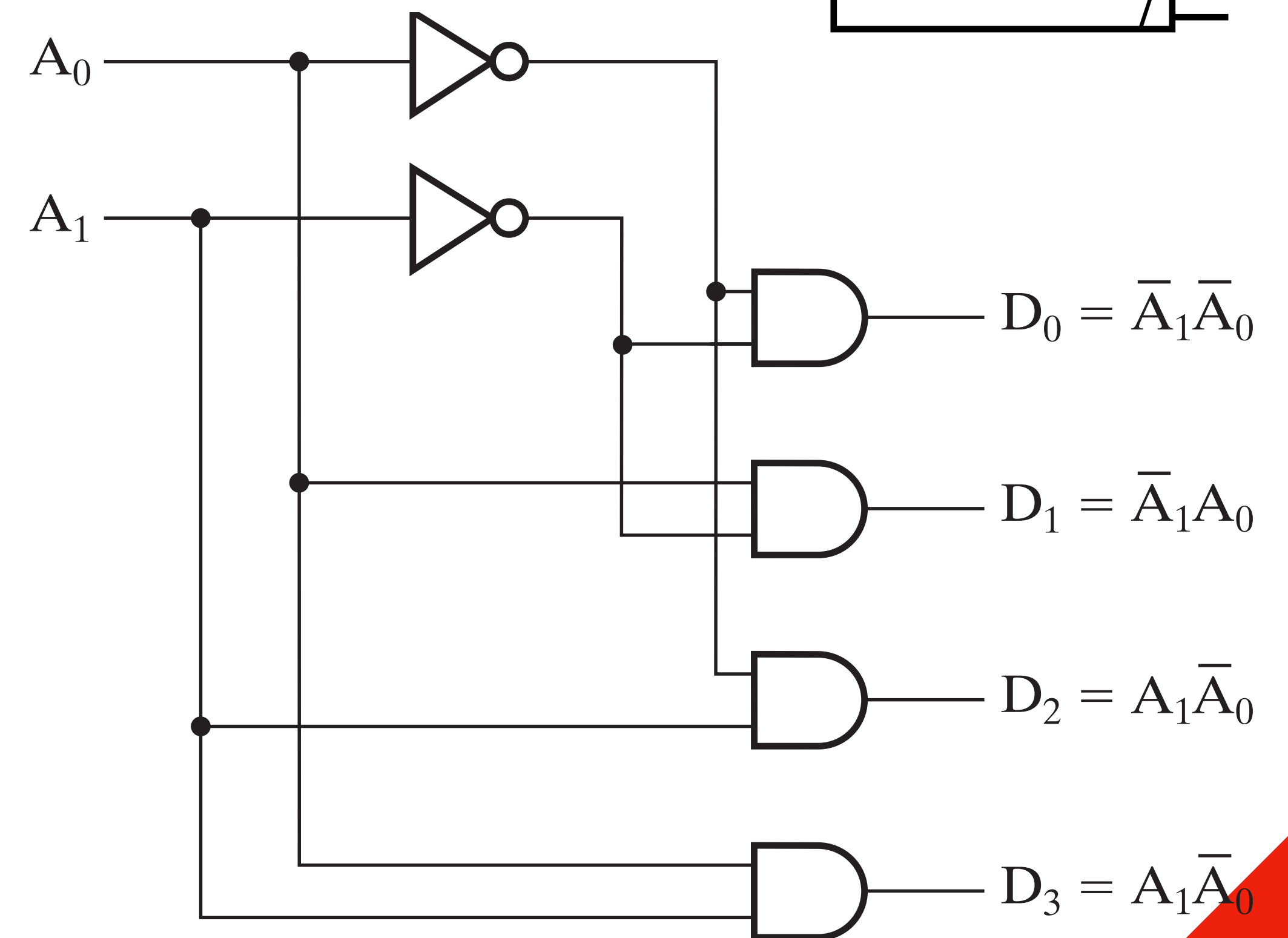
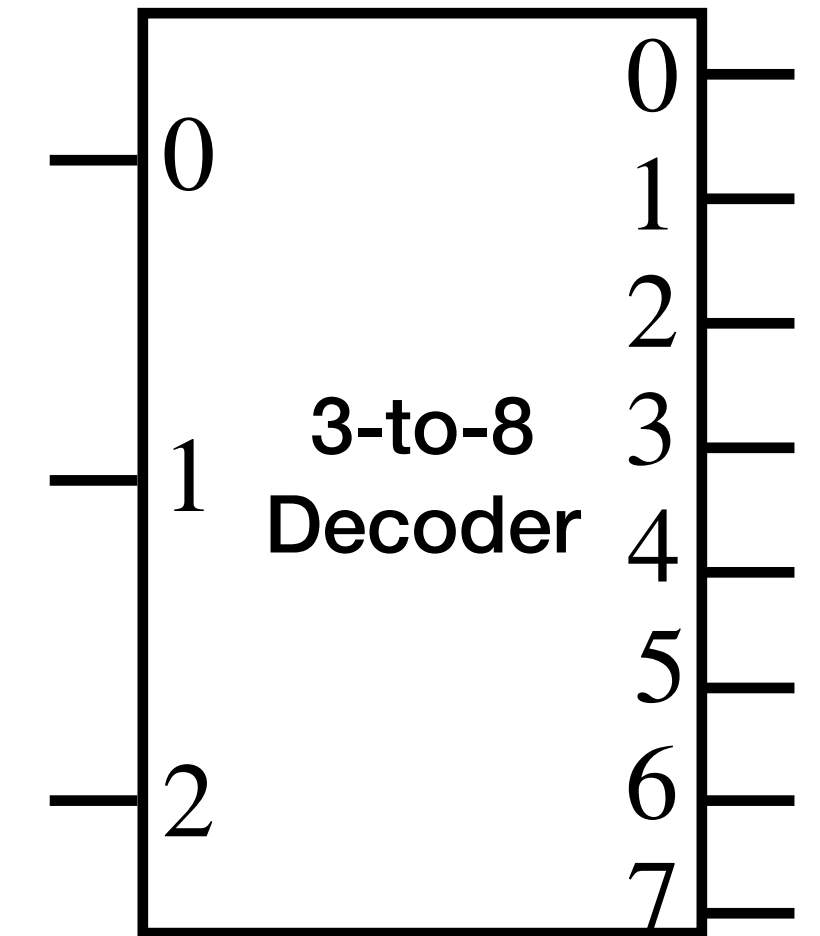


Decoder

- n -bit input, 2^n bits output
- $D_i = m_i$

- Design: use hierarchical designs!

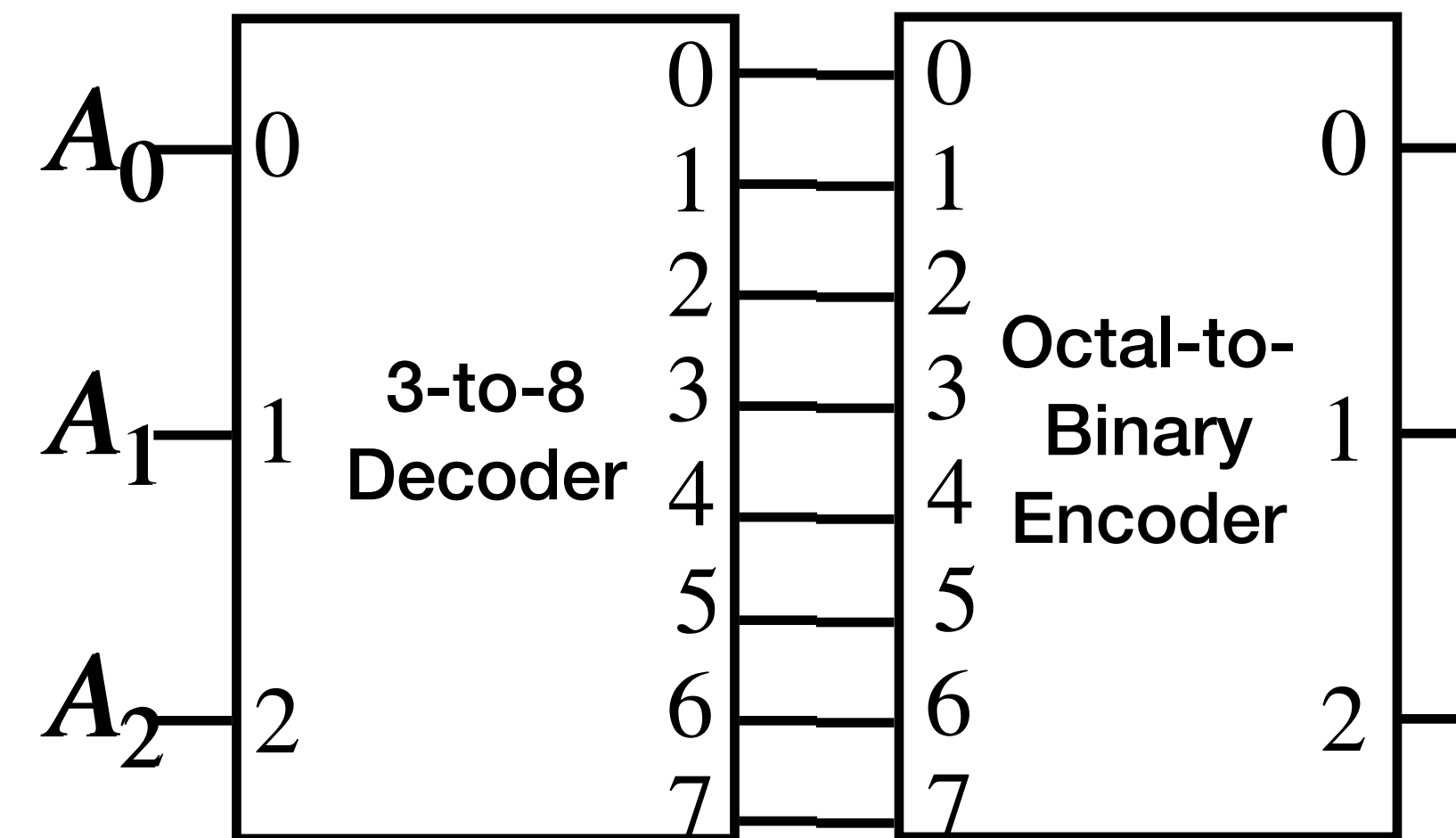
A_1	A_0	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



Concept

Encoder

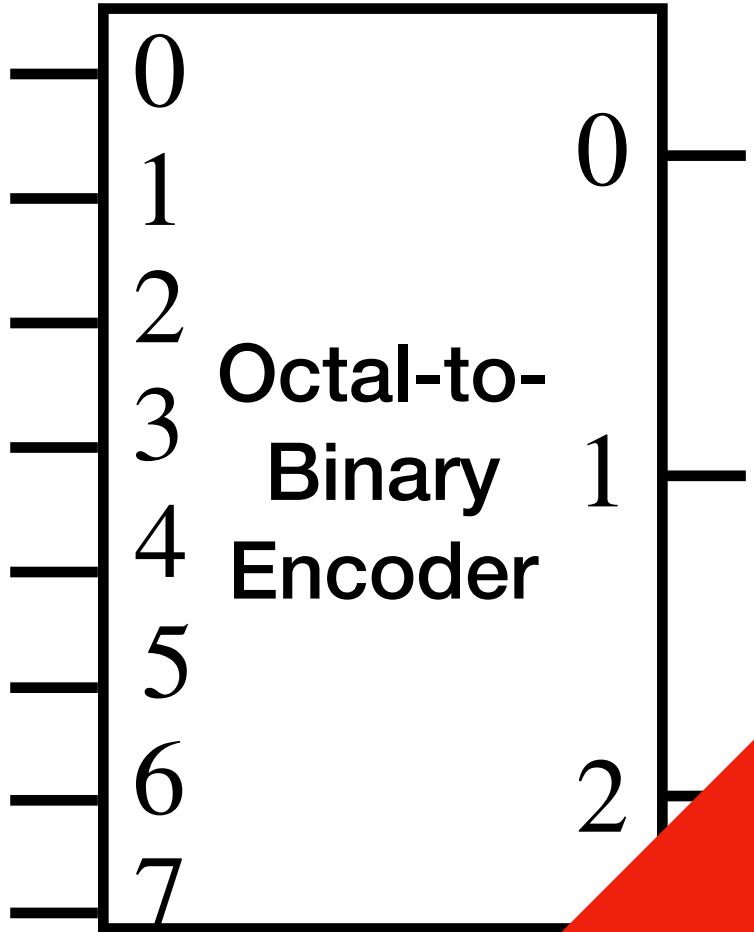
- Inverse operation of a decoder
- 2^n inputs, only one is giving positive input¹
- n outputs



Encoder

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀	A ₂	A ₁	A ₀
							1	0	0	0
						1		0	0	1
					1			0	1	0
				1				0	1	1
			1					1	0	0
		1						1	0	1
	1							1	1	0
1								1	1	1

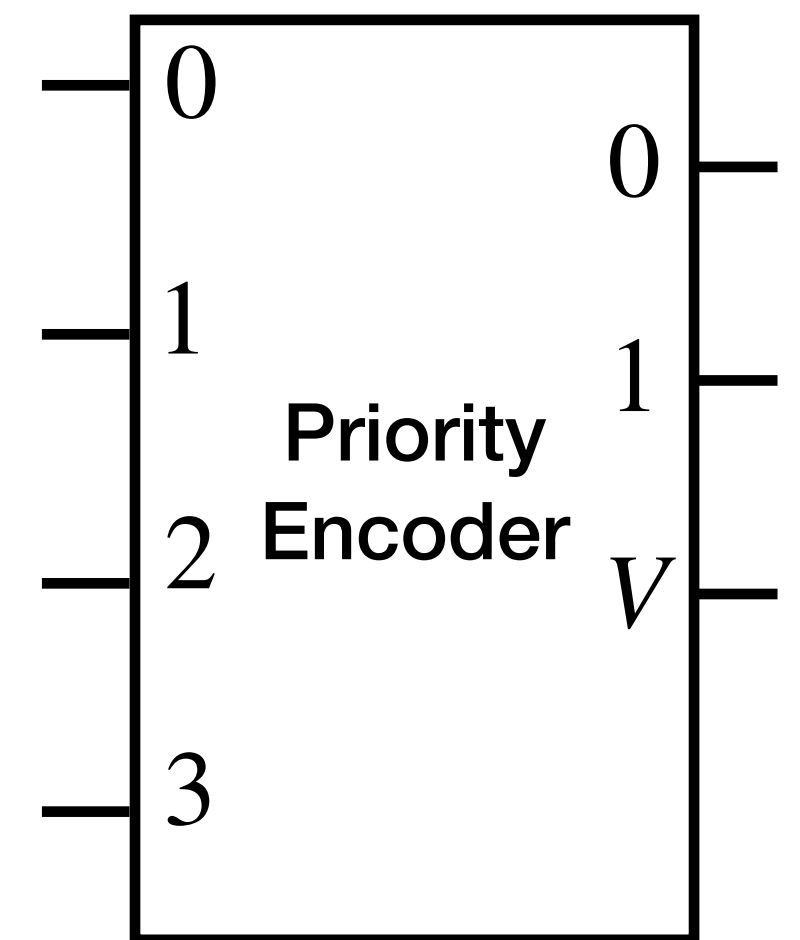
$$A_0 = D_1 + D_3 + D_5 + D_7$$
$$A_1 = D_2 + D_3 + D_6 + D_7$$
$$A_2 = D_4 + D_5 + D_6 + D_7$$



Concept

Priority Encoder

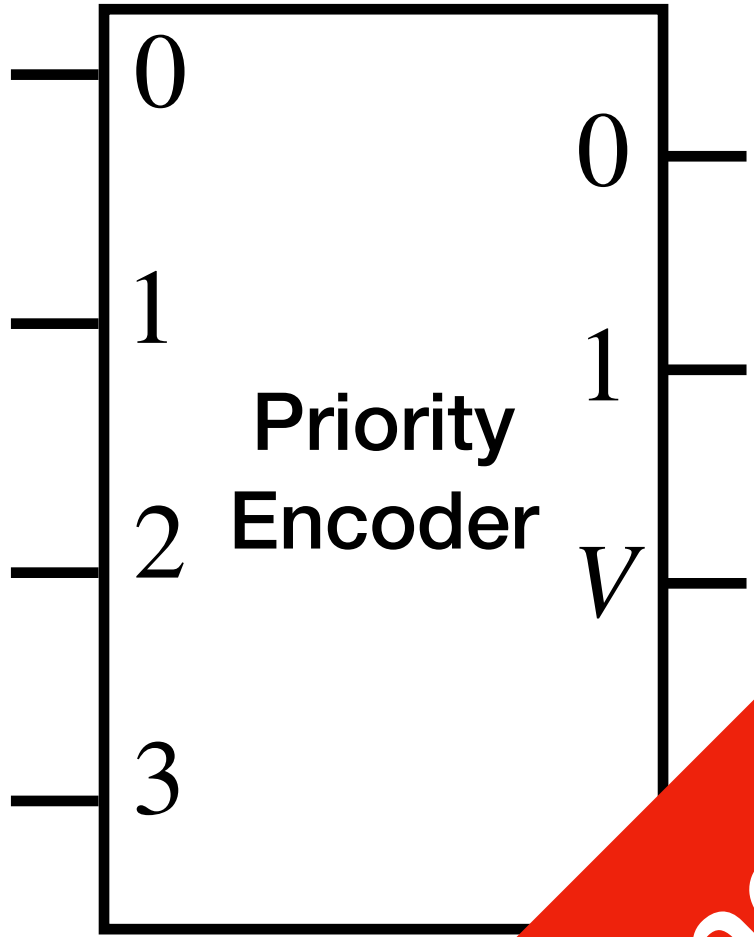
- Additional Validity Output V
 - Indicating whether the input is valid (contains 1)
- Priority
 - Ignores $D_{<i}$ if $D_i = 1$



Priority Encoder

D ₃	D ₂	D ₁	D ₀	A ₁	A ₀	V
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

$$V = D_3 + D_2 + D_1 + D_0$$
$$A_1 = D_3 + \overline{D_3}D_2 = D_2 + D_3$$
$$A_0 = \overline{D_3}\overline{D_2}D_1 + D_3$$
$$= \overline{D_2}D_1 + D_3$$



Concept

Multiplexer

- Multiple n -variable input vectors
- Single n -variable output vector
- Switches: which input vectors to output

