Preliminaries: Combinational Logic

Chulyun Kim

* This material is based on the lecture slides provided by Morgan Kaufmann

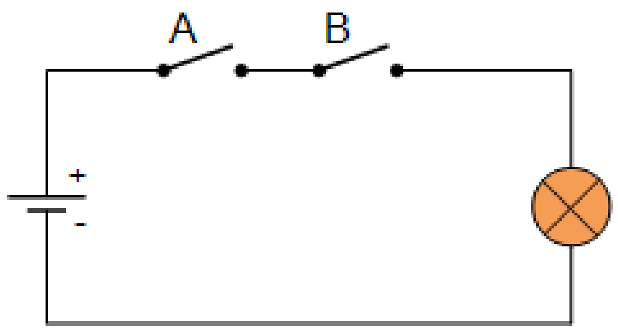
SOOKMYUNG WOMEN'S UNIVERSITY

Outline

- Truth Table
- Gates
- Decoder
- Multiplexor
- BUS
- ALU

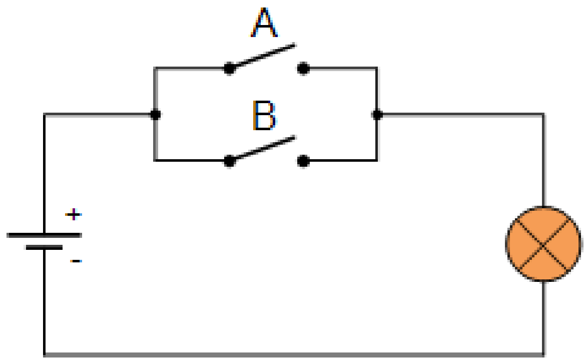
Truth Table

INPUTS		OUTPUTS					
A	B	AND	NAND	OR	NOR	EXOR	EXNOR
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1



Lamp - ON = "1"
Lamp - OFF = "0"

Switch A - Open = "0", Closed = "1"
Switch B - Open = "0", Closed = "1"

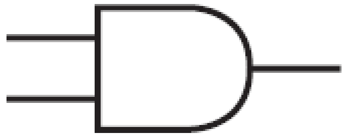


Lamp - ON = "1"
Lamp - OFF = "0"

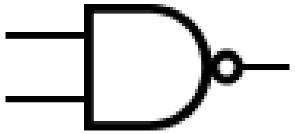
Switch A - Open = "0", Closed = "1"
Switch B - Open = "0", Closed = "1"

Gates

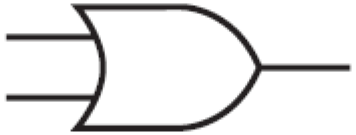
- AND: $A \bullet B$



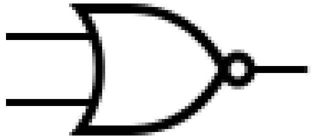
NAND: $\overline{A \bullet B}$



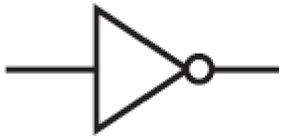
- OR: $A+B$



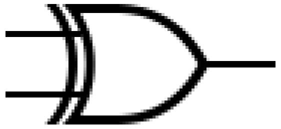
NOR: $\overline{A+B}$



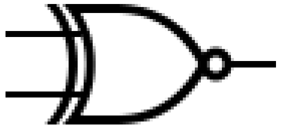
- NOT: $\overline{A}$



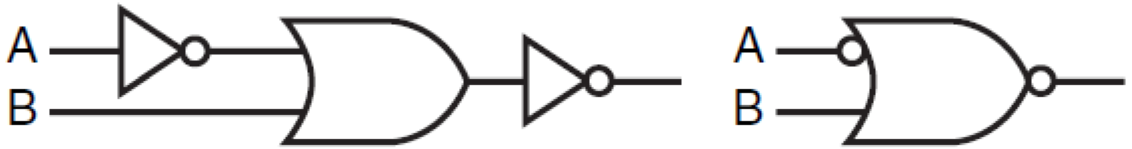
- XOR: $A \oplus B$



XNOR: $\overline{A \oplus B}$



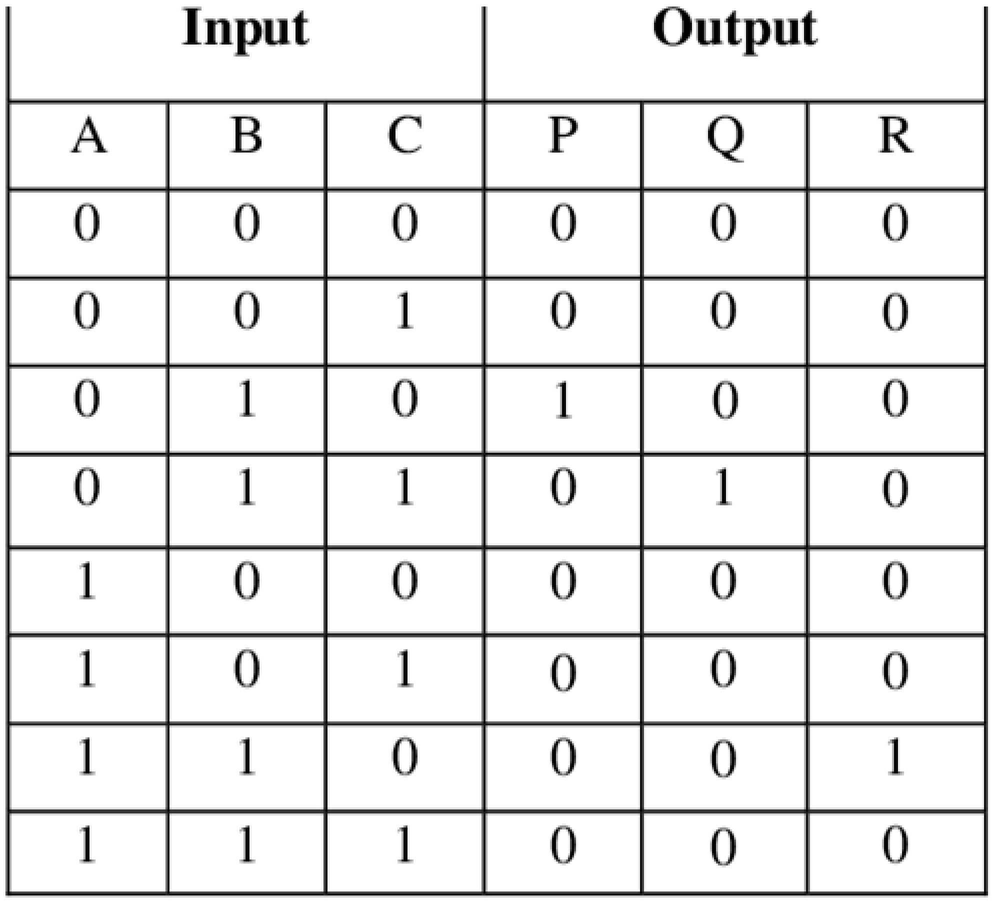
- Example: $\overline{\overline{A}+B}$



How to Implement Output?

Input			Output		
A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

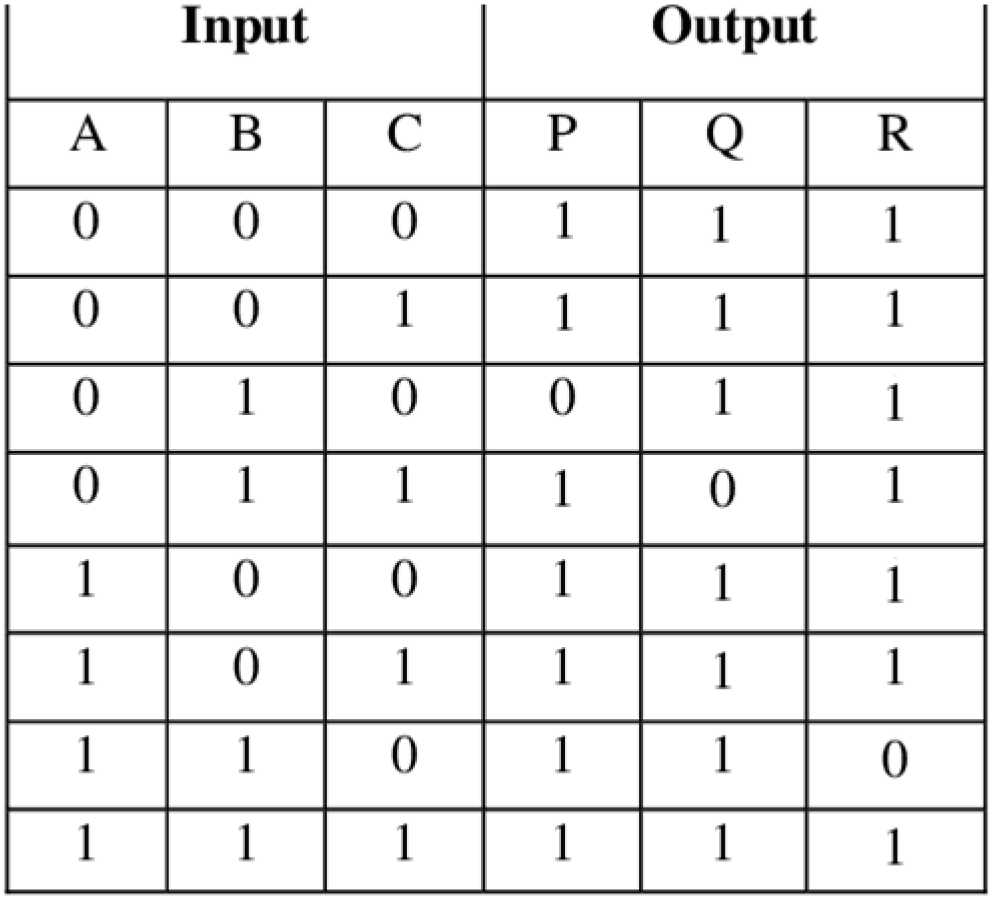
How Make One-Hot



Input			Output		
A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	1
1	1	1	0	0	0

$$P = \bar{A} \cdot B \cdot \bar{C}$$

How to Make One-Cold



Input			Output		
A	B	C	P	Q	R
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	0
1	1	1	1	1	1

$$P = A + \overline{B} + C$$

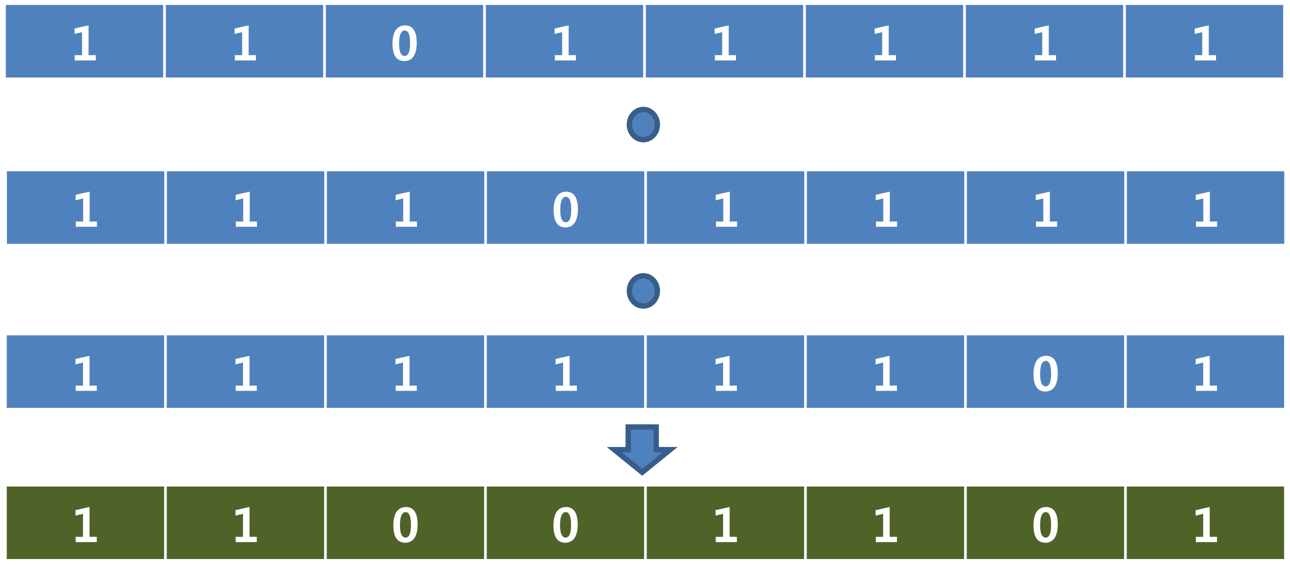
Disjunctive normal form (DNF)

- Combine one-hot clauses
  - Conjunction of literals → one-hot clause
  - Disjunction of clauses → combine one-hots

$$\begin{array}{c}
\begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&1&0&0&0&0&0\\ \hline\end{array}\\
+\\
\begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&0&1&0&0&0&0\\ \hline\end{array}\\
+\\
\begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&0&0&0&0&1&0\\ \hline\end{array}\\
\Downarrow\\
\begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&1&1&0&0&1&0\\ \hline\end{array}
\end{array}$$

Conjunctive normal form (CNF)

- Combine one-cold clauses
  - Disjunction of literals → one-cold clause
  - Conjunction of clauses → combine one-colds



Normal Form

- Conjunctive normal form (CNF)
  - it is a conjunction of one or more clauses, where a clause is a disjunction of literals

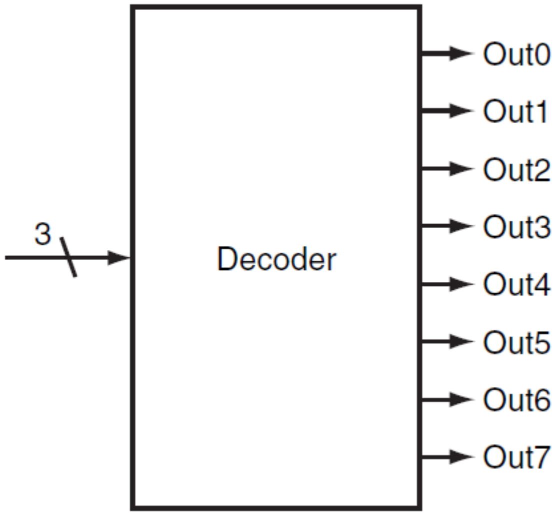
$$(A \lor B) \land (\neg B \lor C \lor \neg D) \land (D \lor \neg E)$$

- Disjunctive normal form (DNF)
  - it is a disjunction of one or more clauses, where a clause is a conjunction of literals

$$(A \land \neg B \land \neg C) \lor (\neg D \land E \land F)$$

Decoder

- An n-bit input and 2^n outputs
- If the value of the input is *i*, then Out*i* will be true and all other outputs will be false

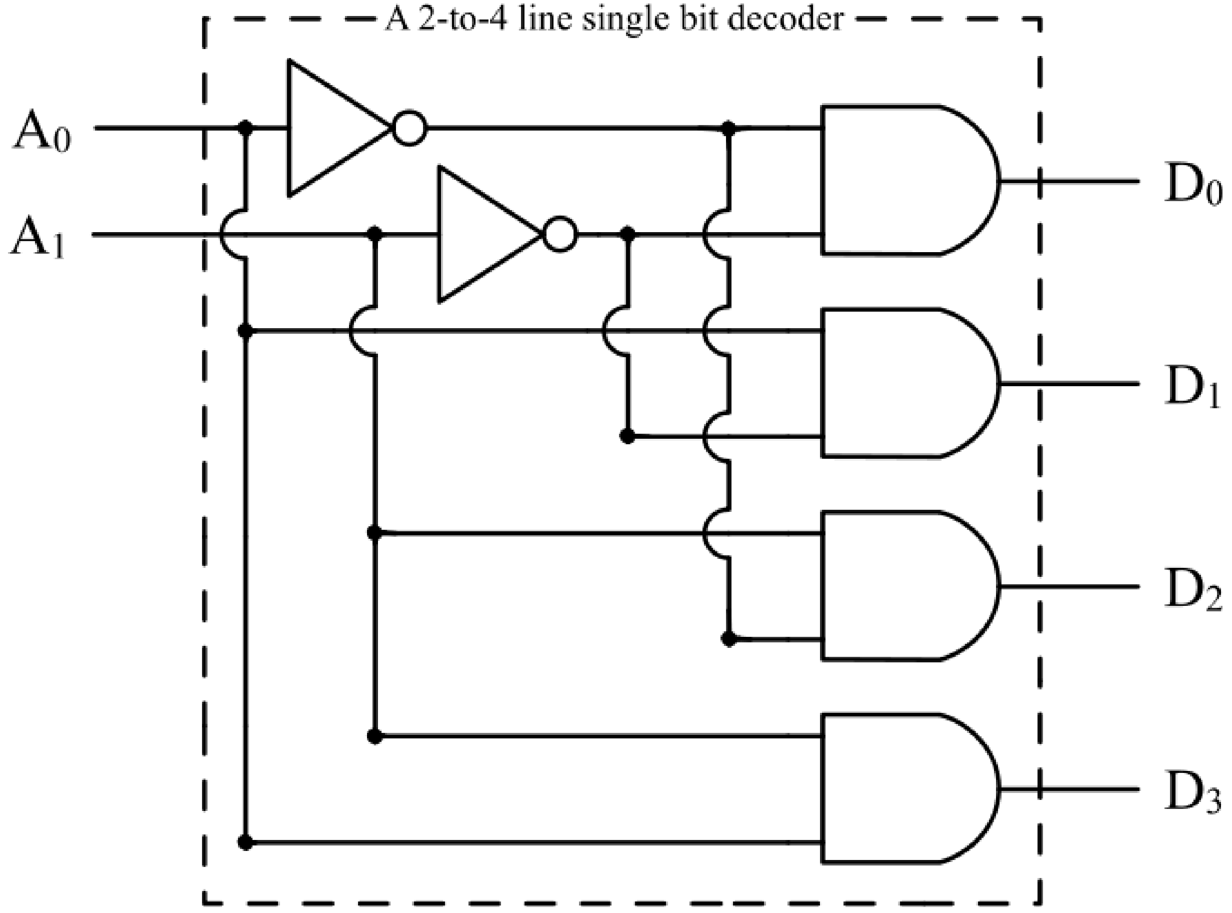


a. A 3-bit decoder

Inputs			Outputs							
I2	I1	I0	Out7	Out6	Out5	Out4	Out3	Out2	Out1	Out0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

b. The truth table for a 3-bit decoder

2-bit Decoder



Truth Table

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

Minterm Equations

$$D_0 = \overline{A_1} \cdot \overline{A_0}$$

$$D_1 = \overline{A_1} \cdot A_0$$

$$D_2 = A_1 \cdot \overline{A_0}$$

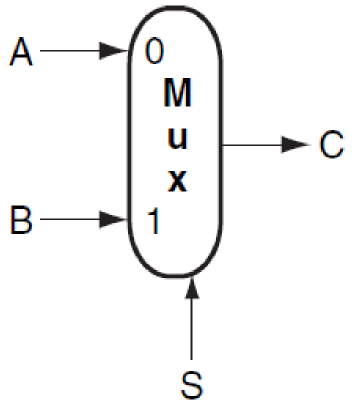
$$D_3 = A_1 \cdot A_0$$

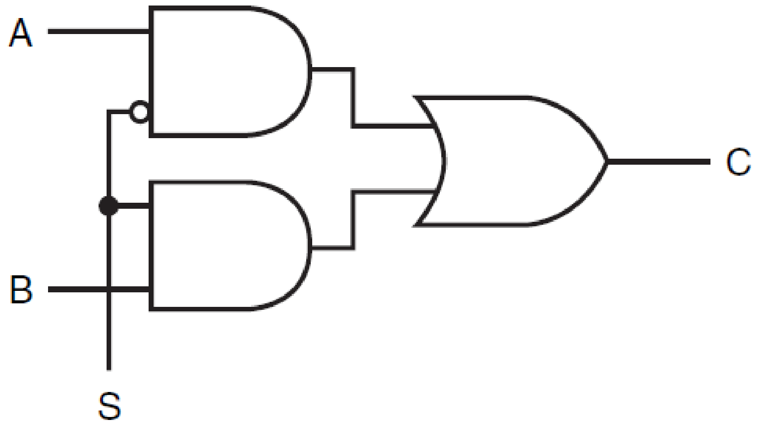
Example

- 3-bit Decoder

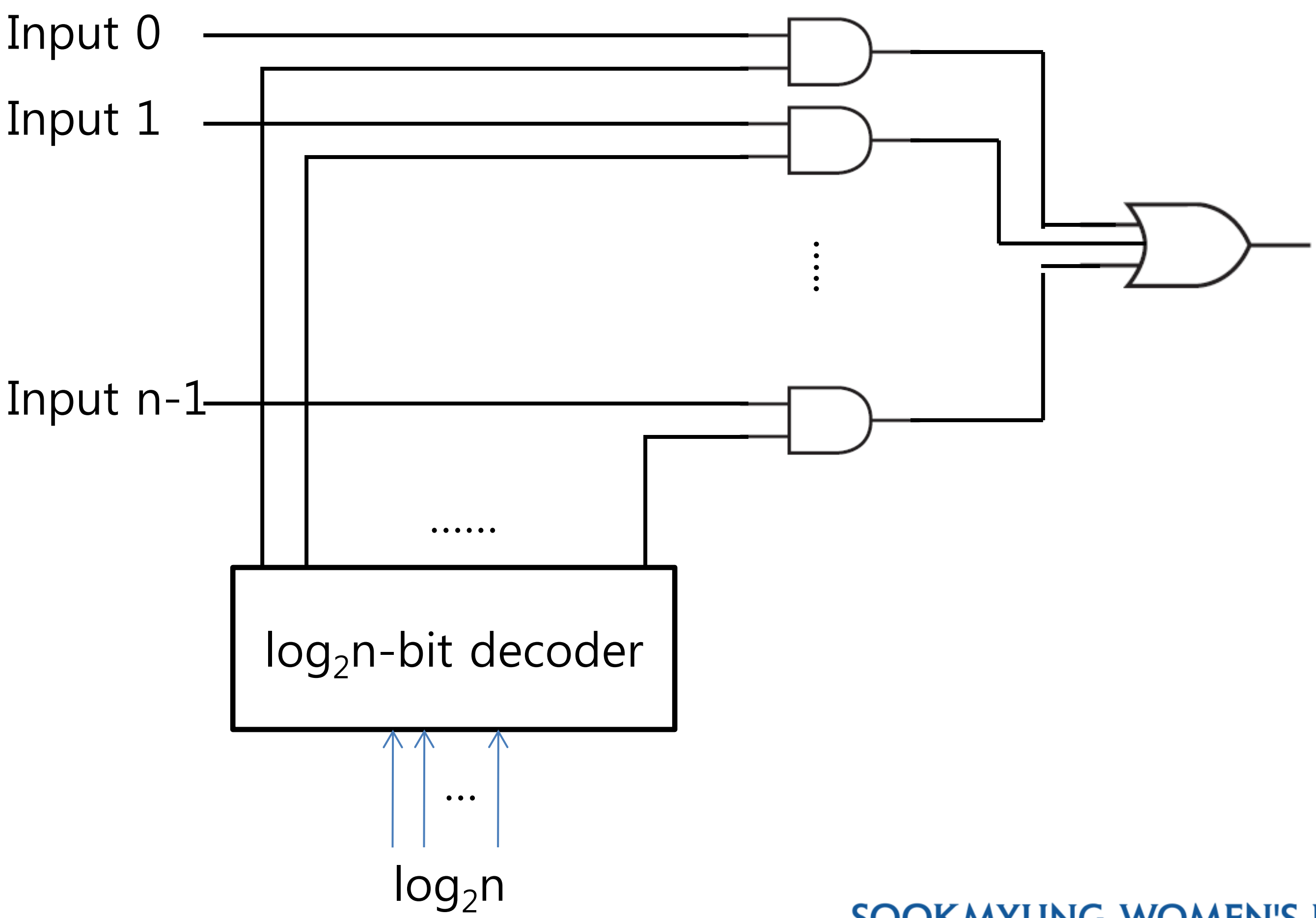
Multiplexor

- Selector
  - Its output is one of the inputs that is selected by a control
  - If there are n data inputs, there will need to be $\lceil \log_2 n \rceil$ selector inputs
- 2-input Multiplexor (or 1-bit Multiplexor)
  - $C = (A \cdot \overline{S}) + (B \cdot S)$





n-input Multiplexor

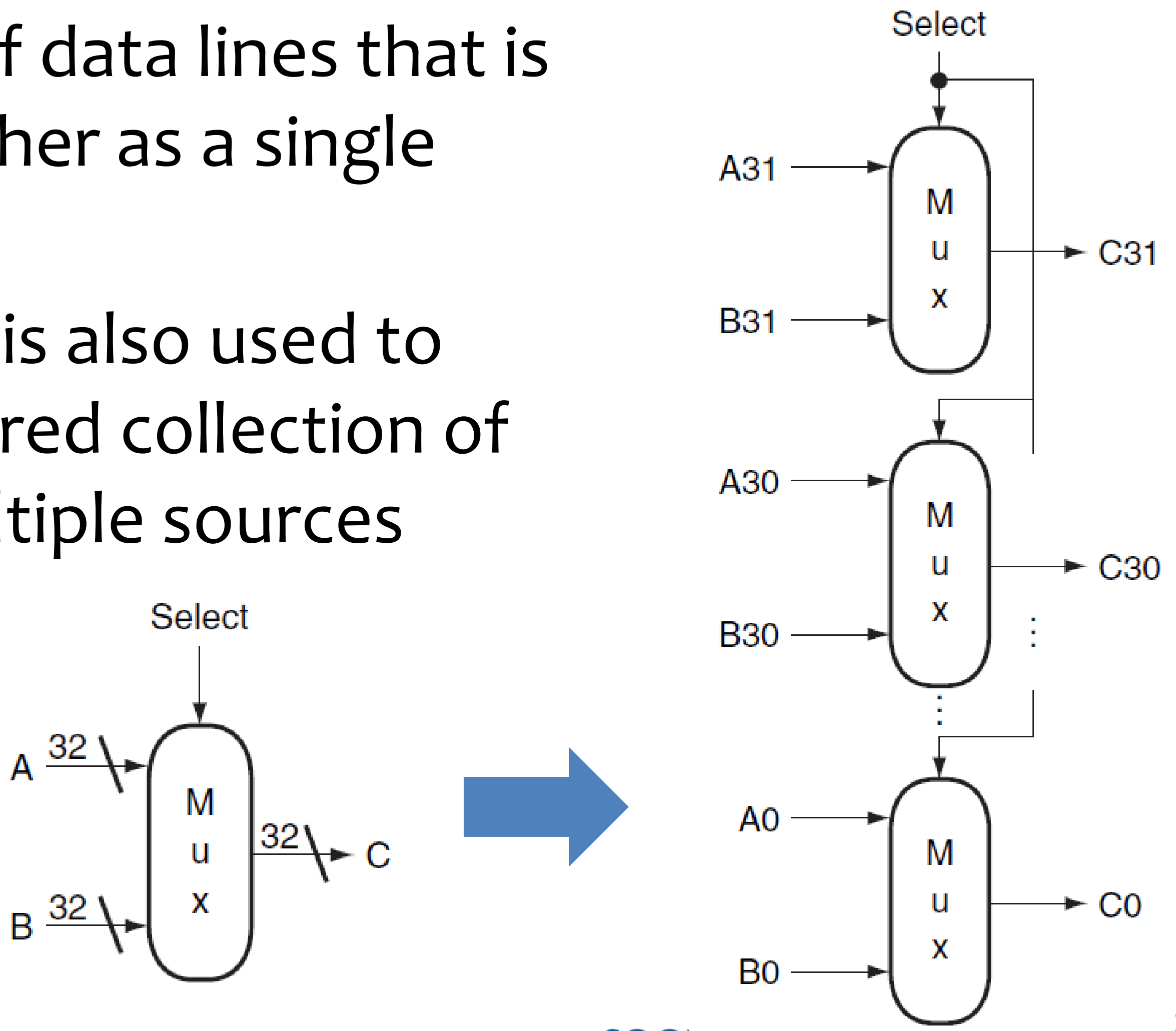


Example

- 4-input (or 2-bit) Multiplexor

BUS

- A collection of data lines that is treated together as a single logical signal
- The term bus is also used to indicate a shared collection of lines with multiple sources

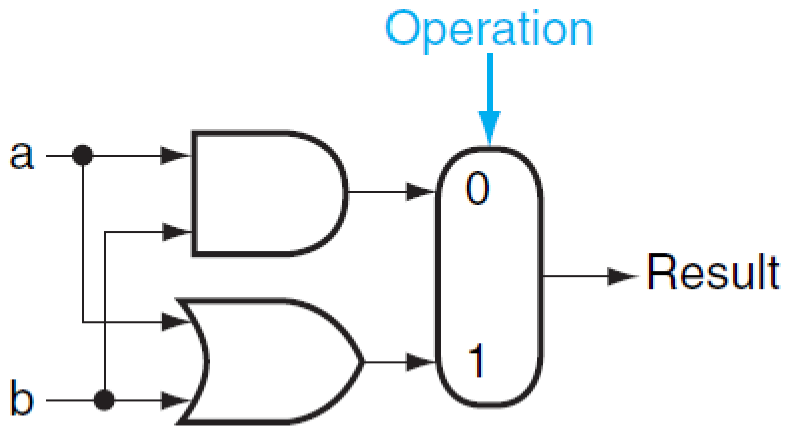


ALU: Arithmetic Logic Unit

- Device that performs
  - Arithmetic operations like addition and subtraction
  - Logical operations like AND and OR

Logical Unit for AND/OR

- 1-bit Logical Unit for AND/OR



- Example
  – 2-bit Logical Unit for AND/OR

1-bit Adder

Inputs			Outputs		Comments
a	b	CarryIn	CarryOut	Sum	
0	0	0	0	0	$0 + 0 + 0 = 00_{two}$
0	0	1	0	1	$0 + 0 + 1 = 01_{two}$
0	1	0	0	1	$0 + 1 + 0 = 01_{two}$
0	1	1	1	0	$0 + 1 + 1 = 10_{two}$
1	0	0	0	1	$1 + 0 + 0 = 01_{two}$
1	0	1	1	0	$1 + 0 + 1 = 10_{two}$
1	1	0	1	0	$1 + 1 + 0 = 10_{two}$
1	1	1	1	1	$1 + 1 + 1 = 11_{two}$

1-bit Adder

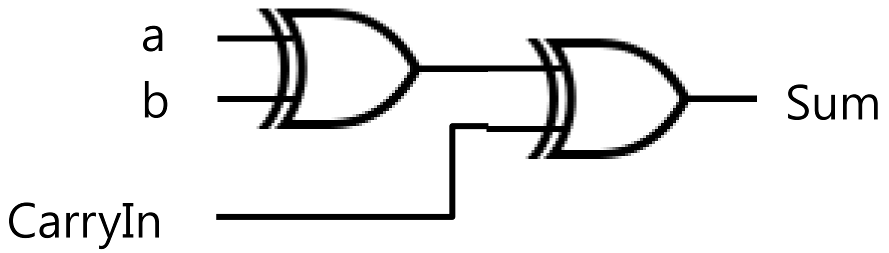
- $\mathrm{Sum} = (a \cdot \overline{b} \cdot \overline{\mathrm{CarryIn}}) + (\overline{a} \cdot b \cdot \overline{\mathrm{CarryIn}})$

  $+ (\overline{a} \cdot \overline{b} \cdot \mathrm{CarryIn}) + (a \cdot b \cdot \mathrm{CarryIn})$

  $= (a \cdot \overline{b} + \overline{a} \cdot b) \cdot \overline{\mathrm{CarryIn}} + (\overline{a} \cdot \overline{b} + a \cdot b) \cdot \mathrm{CarryIn}$

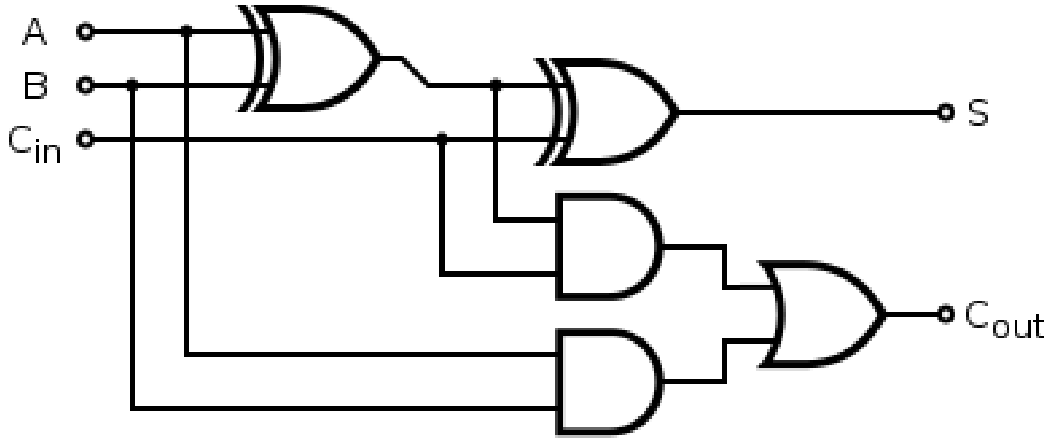
  $= a \oplus b \cdot \overline{\mathrm{CarryIn}} + \overline{a \oplus b} \cdot \mathrm{CarryIn}$

  $= (a \oplus b) \oplus \mathrm{CarryIn}$

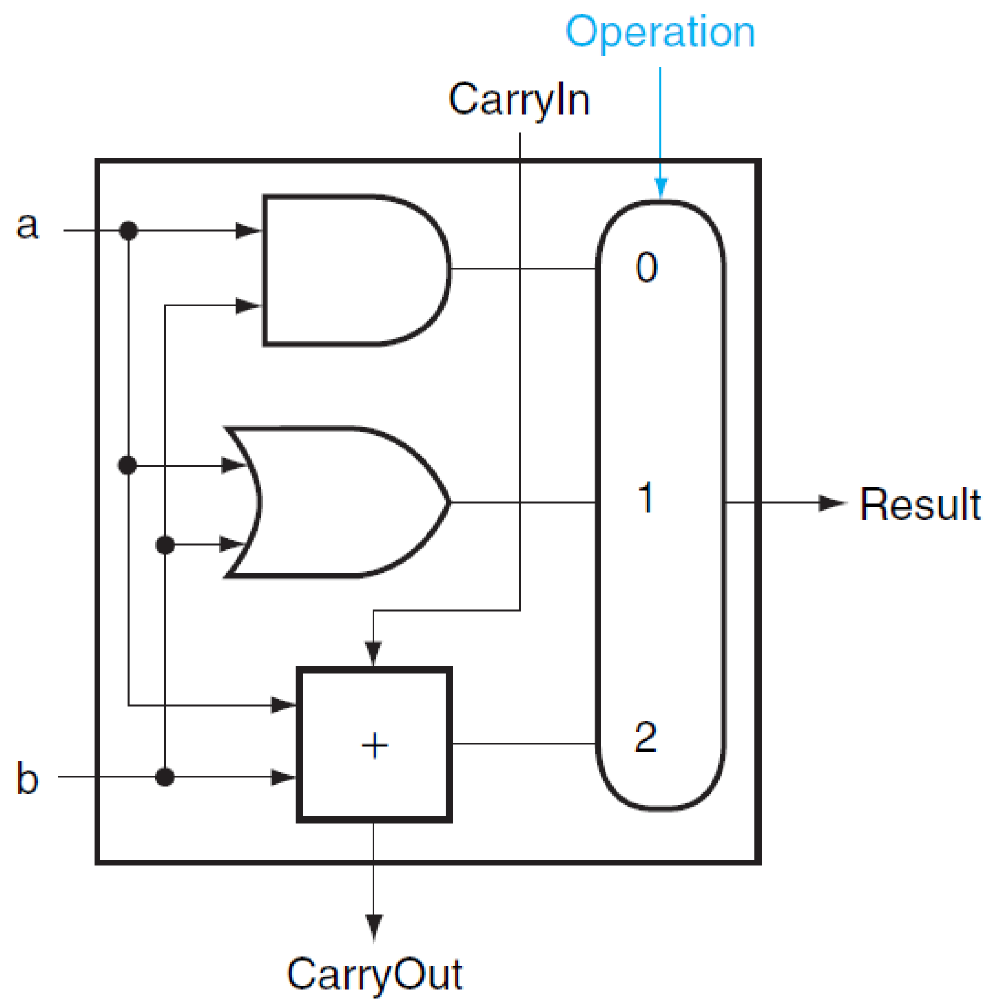


1-bit Adder

- $\text{CarryOut} = (\overline{a}\cdot b\cdot \text{CarryIn}) + (a\cdot \overline{b}\cdot \text{CarryIn})$
  $+ (a\cdot b\cdot \overline{\text{CarryIn}}) + (a\cdot b\cdot \text{CarryIn})$
  $=(\overline{a}\cdot b + a\cdot \overline{b}) \cdot \text{CarryIn} \; + a\cdot b\cdot(\overline{\text{CarryIn}}+\text{CarryIn})$
  $=(a\oplus b) \cdot \text{CarryIn} + a\cdot b$

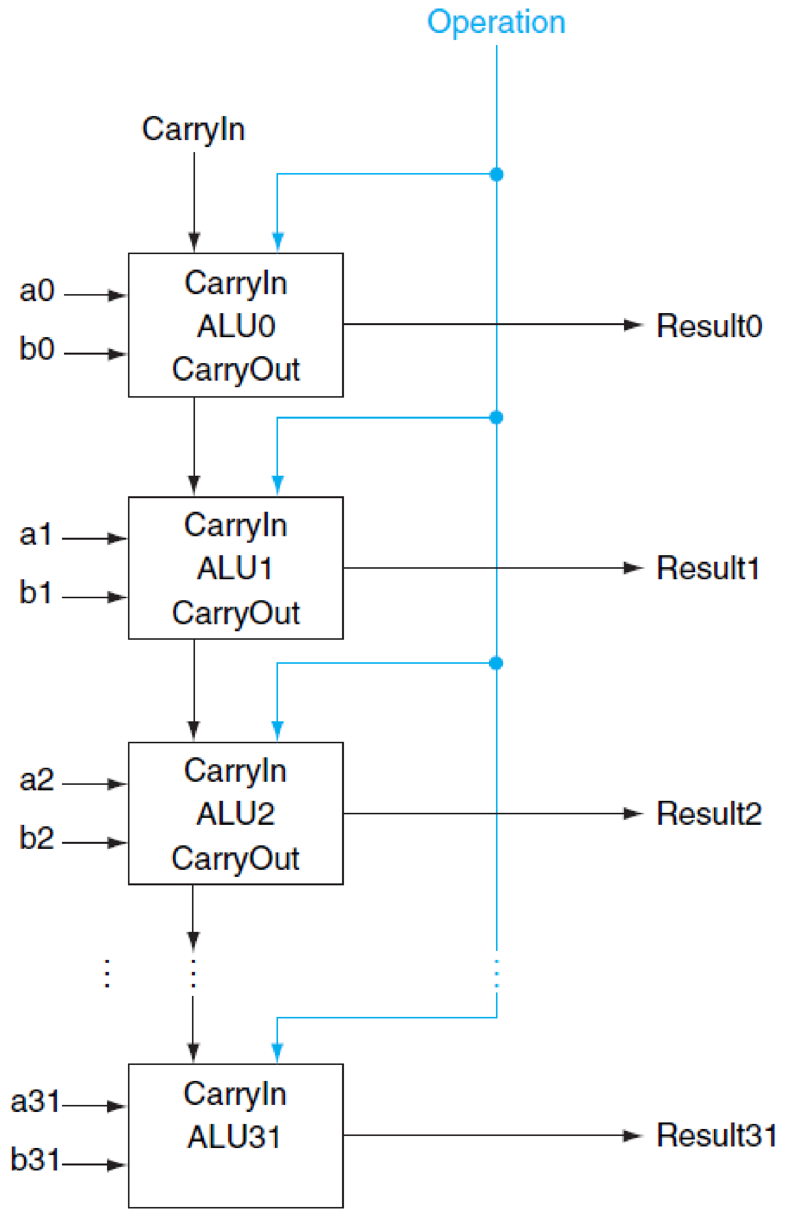


1-but ALU

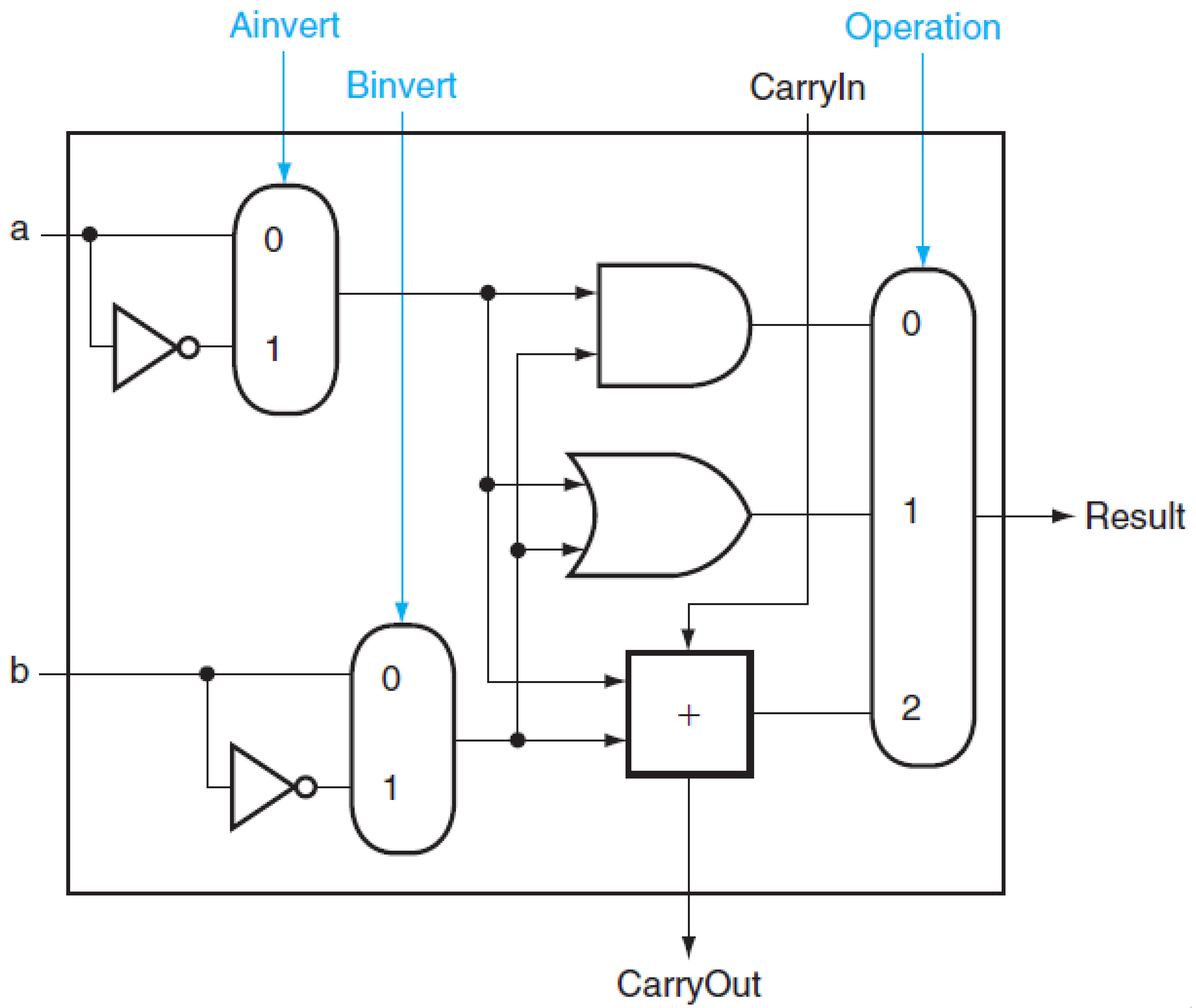


32-bit ALU

- Array of 1-but ALUs



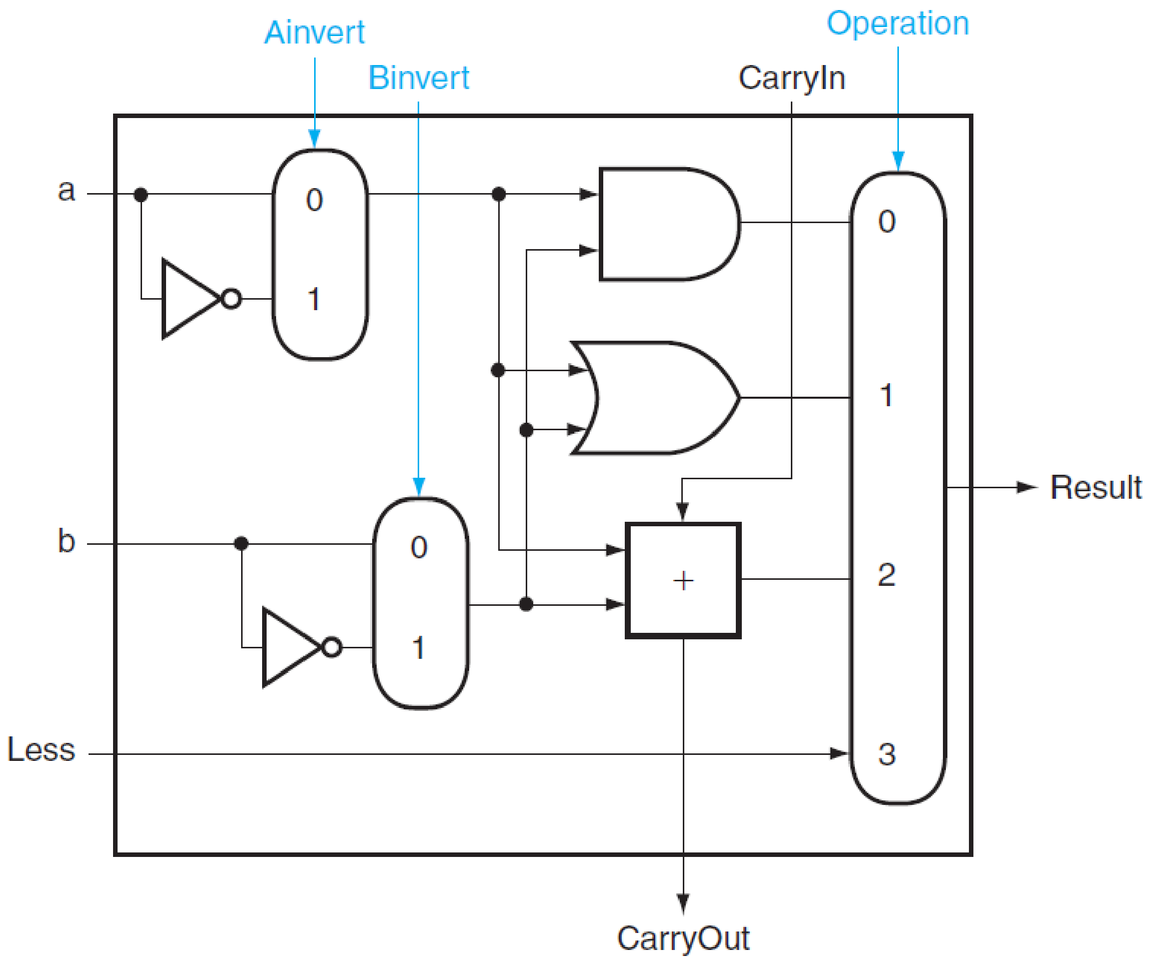
1-but ALU AND/OR/ADD/NOT



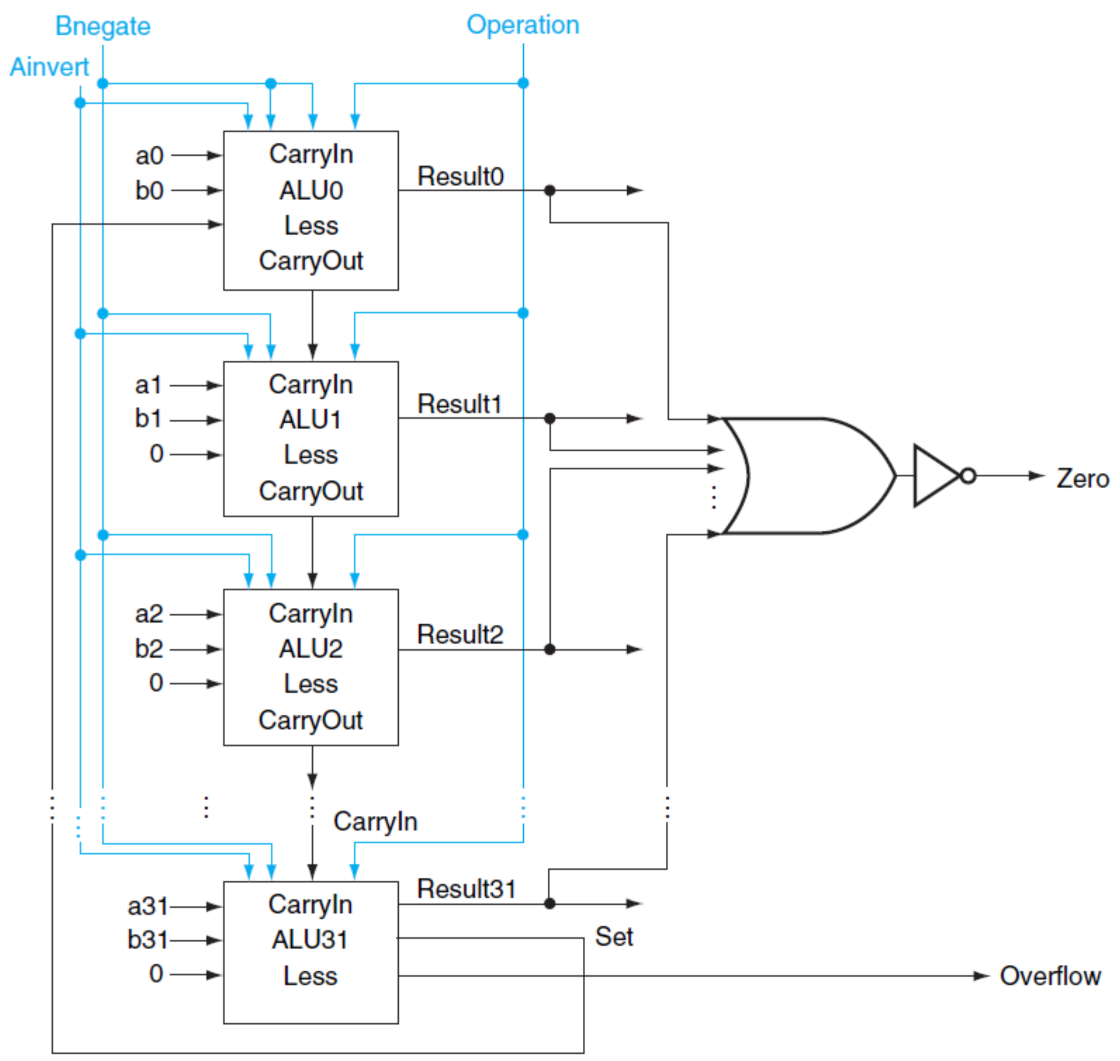
Example

- 2-bit ALU

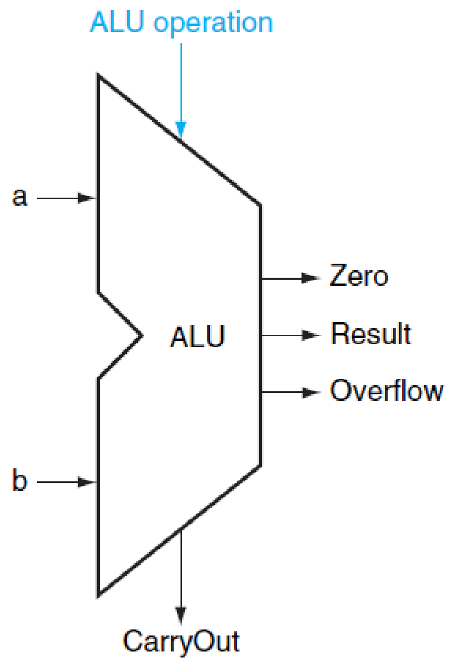
1-but ALU AND/OR/ADD/NOT/SLT



Final 32-bit ALU

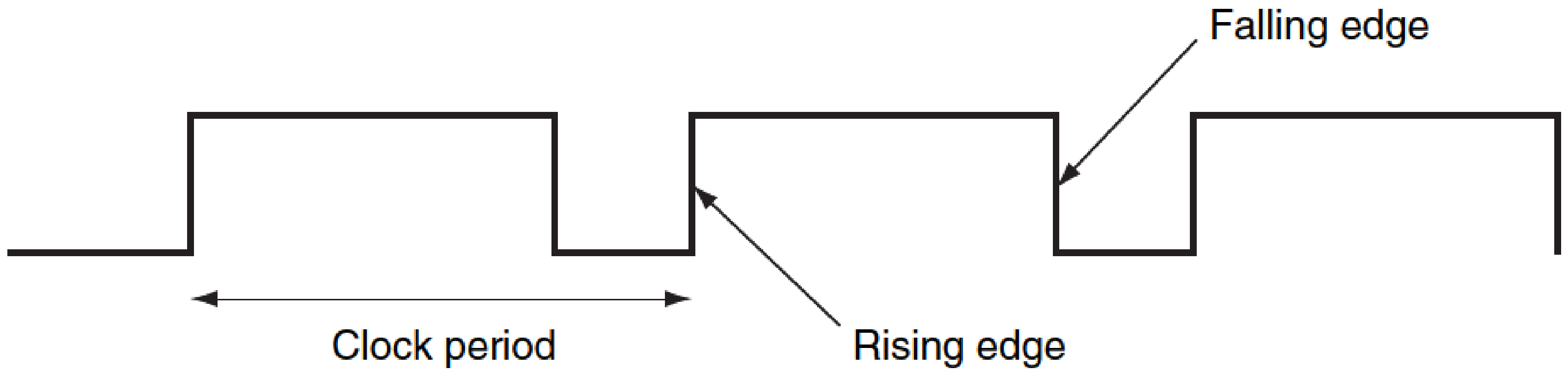


ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR



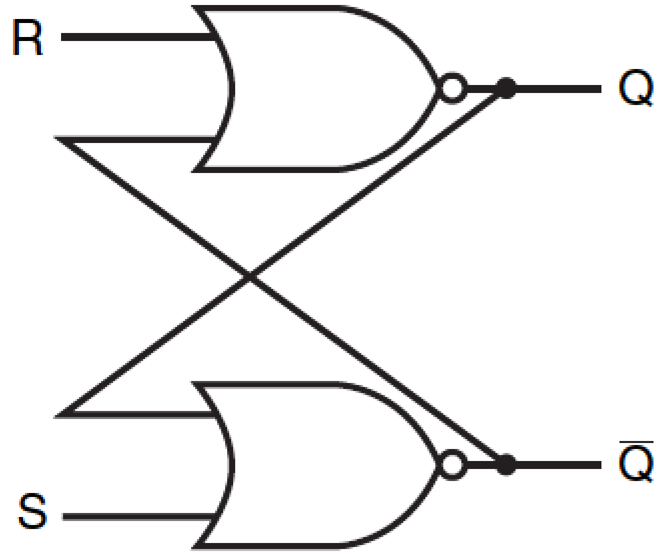
Clocks

- Clocks are needed to decide when a state should be updated
  - Free-running signal with a fixed cycle time
- Edge-triggered clocking
  - All state changes occur on a clock edge



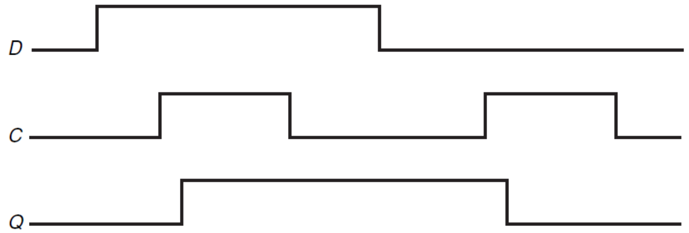
S-R Latch

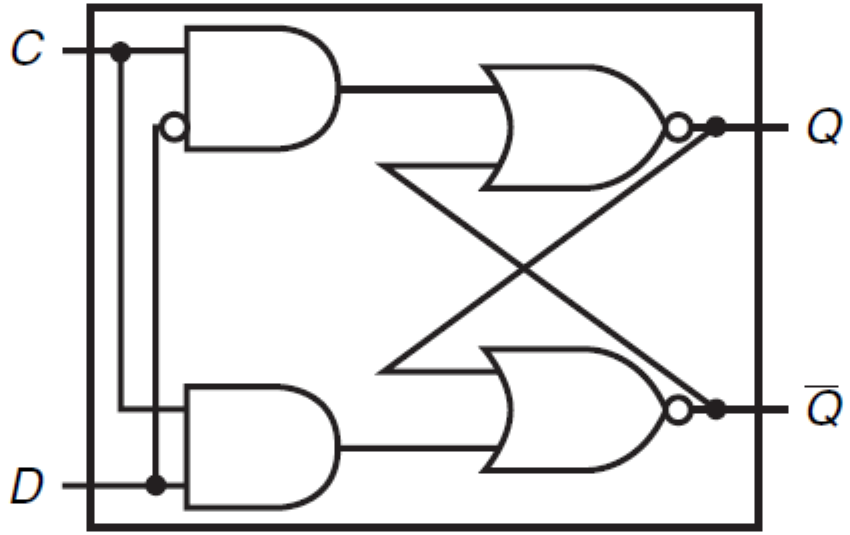
- Set-Reset Latch
  - Unclocked
  - If S=1 (or asserted) and R=0 (or deasserted)
    - Q=1 and $\overline{Q}$=0
  - If S=0 and R=1
    - Q=0 and $\overline{Q}$=1
  - If S=0 and R=0
    - The last values will continue



D Latch

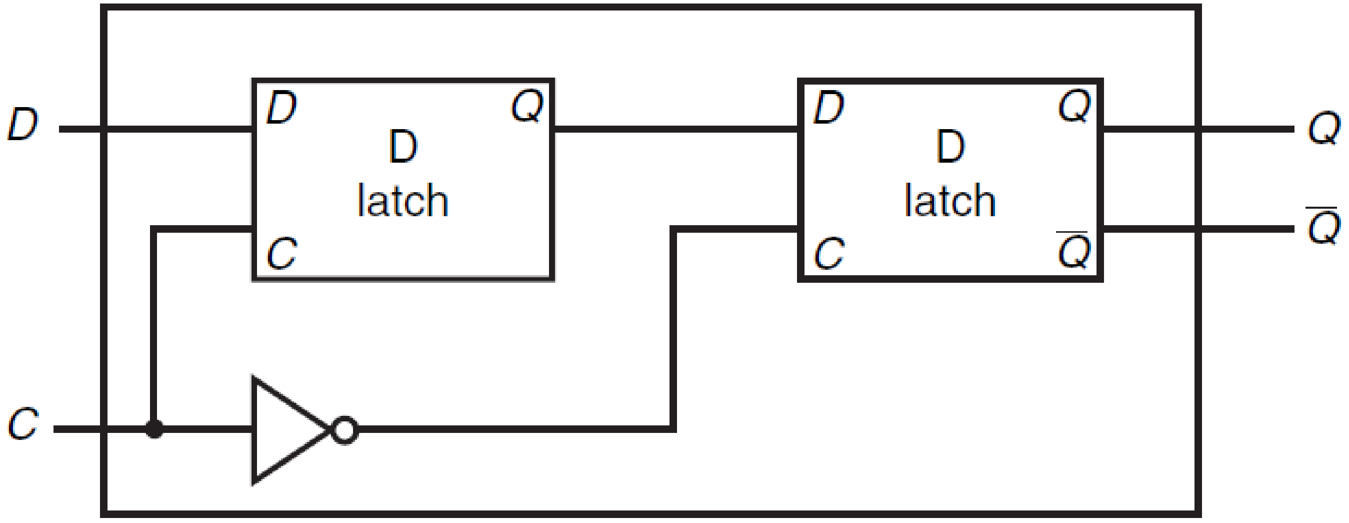
- A clock input
- The change of state is triggered by the clock
- If clock is deasserted (or 0)
  - The last values will continue
- If clock is asserted (or 1)
  - Q will be D
  - $\overline{Q}$ will be $\overline{D}$

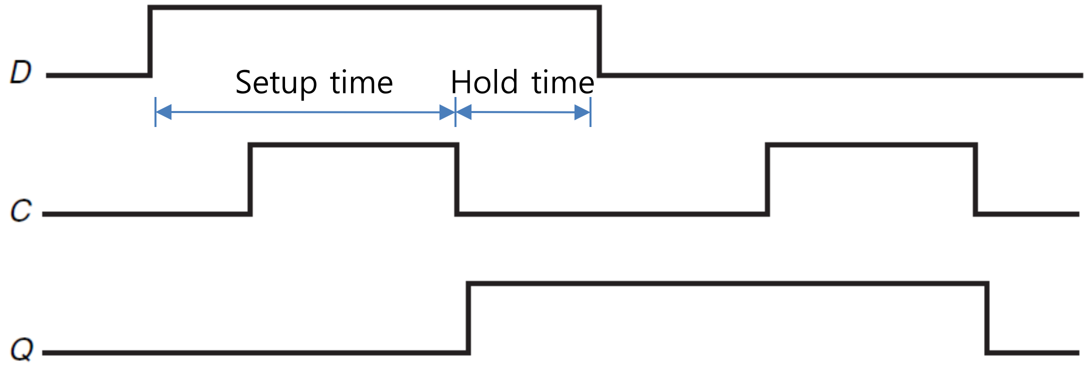




D flip-flop with a falling edge trigger

- Outputs change *only on the clock edge*

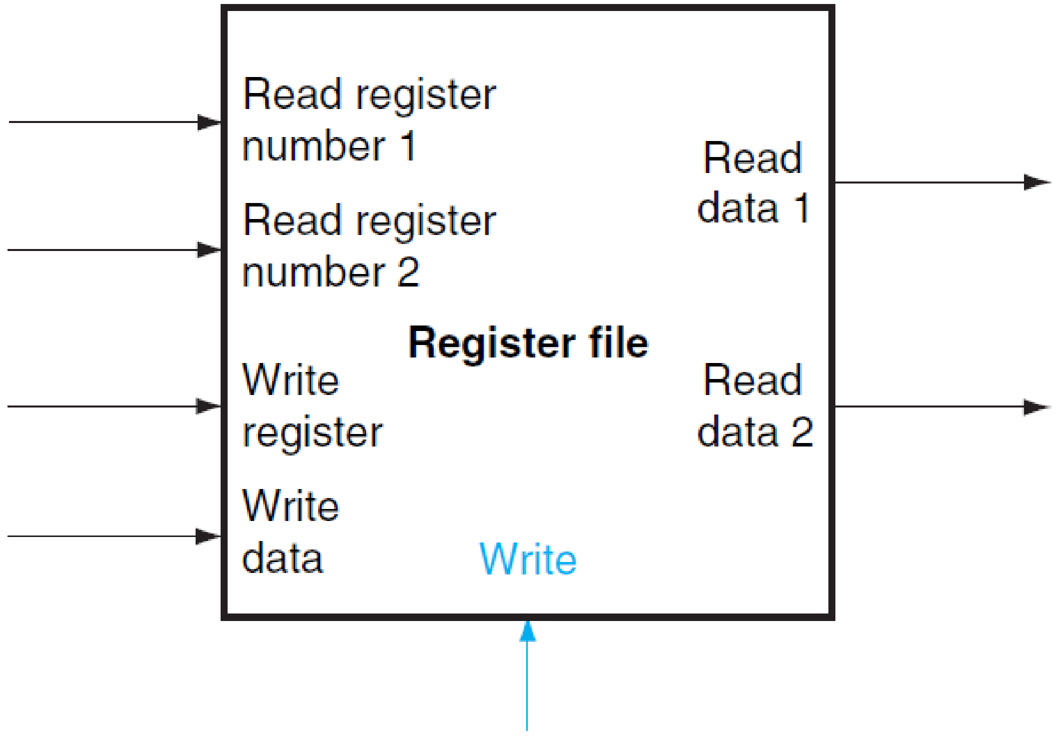




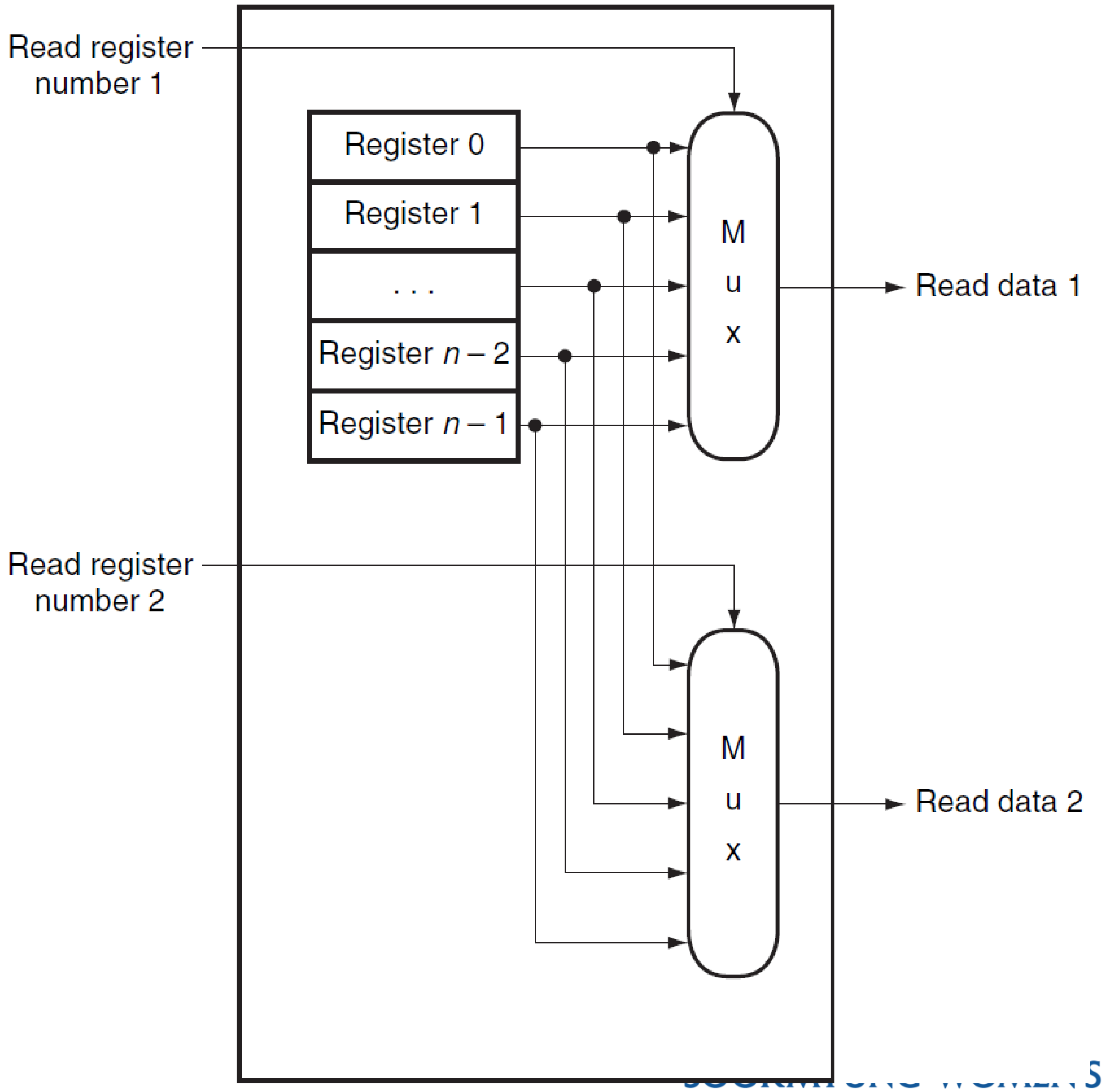
WOMEN'S UNIVERSITY

Register Files

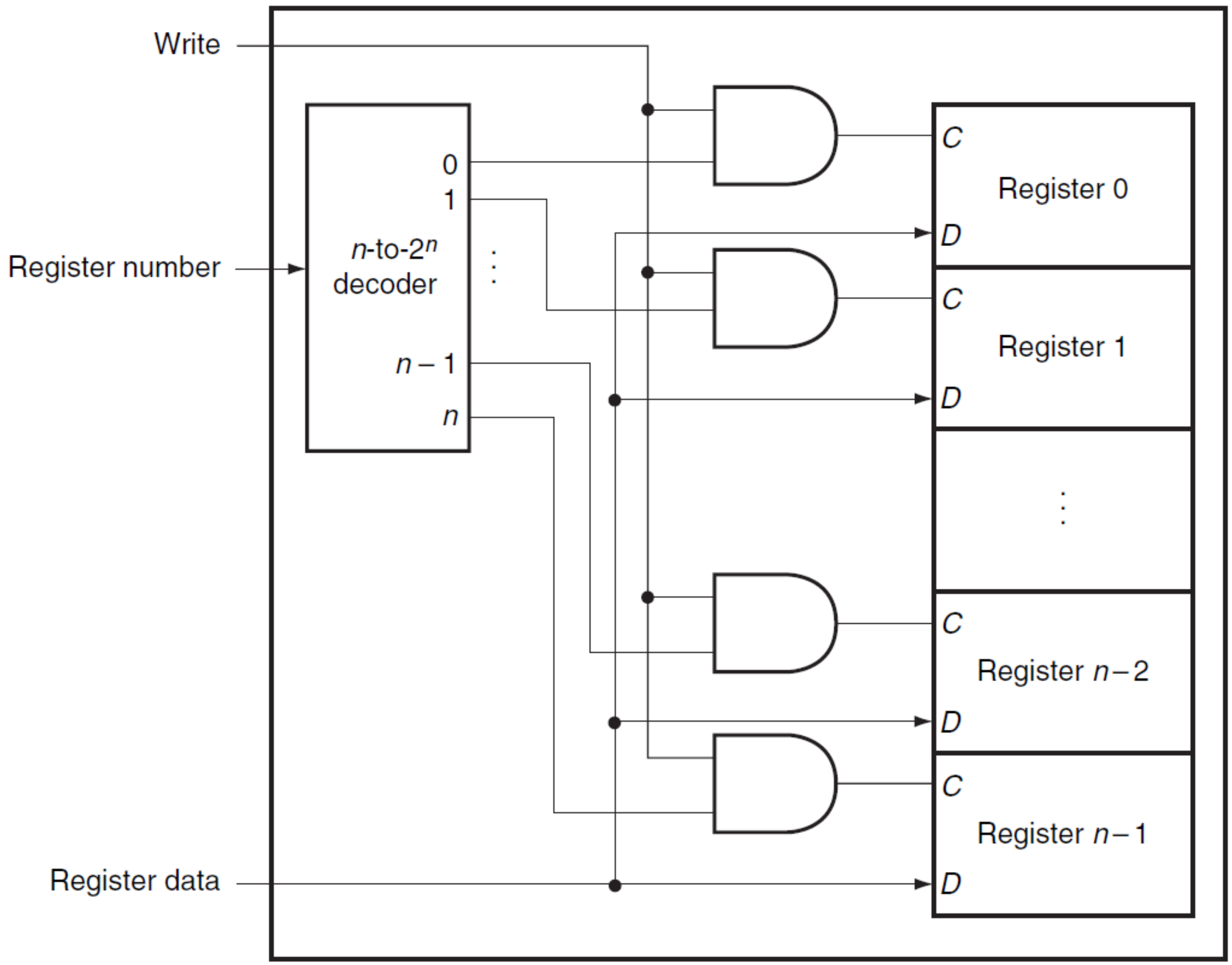
- A set of registers that can be read and written



Read


Read register number 1
Register 0
Register 1
. . .
Register n – 2
Register n – 1
M u x
Read data 1
Read register number 2
M u x
Read data 2

Write


Write
Register number
Register data
n-to-2^n decoder
0
1
n – 1
n
C
D
Register 0
Register 1
Register n – 2
Register n – 1

SITY