# Chapter 03: Computer Arithmetic

Lesson 11:

**Design of ALU**

# Objective

- Understand the units in ALU

# ALUs

# ALU

- Each processor has an ALU
- At ALU the arithmetic and logic operations performed.
- ALU performs two types of operations.
- 1. Fixed point operations
- 2. Floating point operations

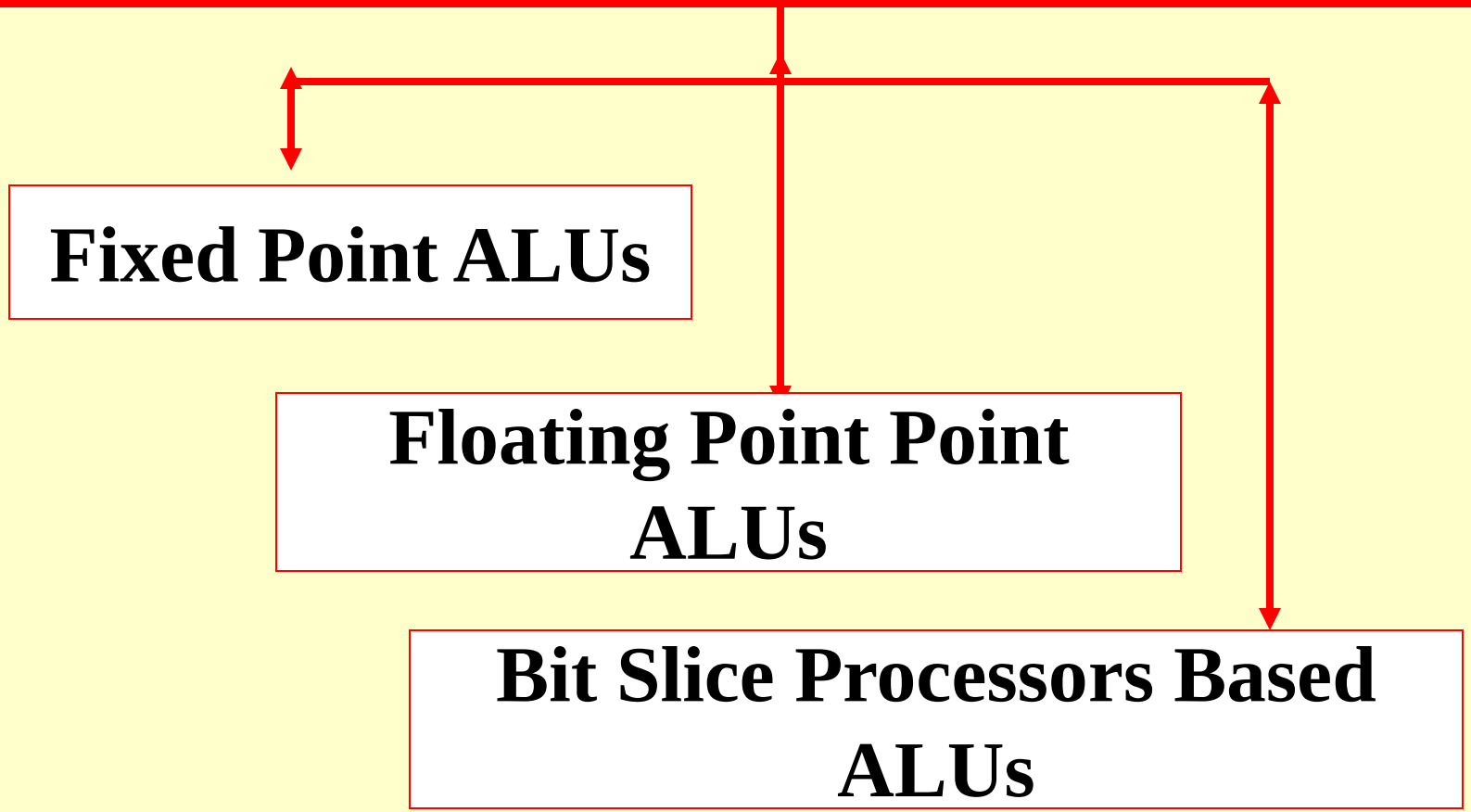
# ALU Design

**Combinational Logic Circuits-based ALUs**

**Sequential Logic Circuits Based ALUs**

**Bit Slice Processors**

# ALUs



# ALU Operations

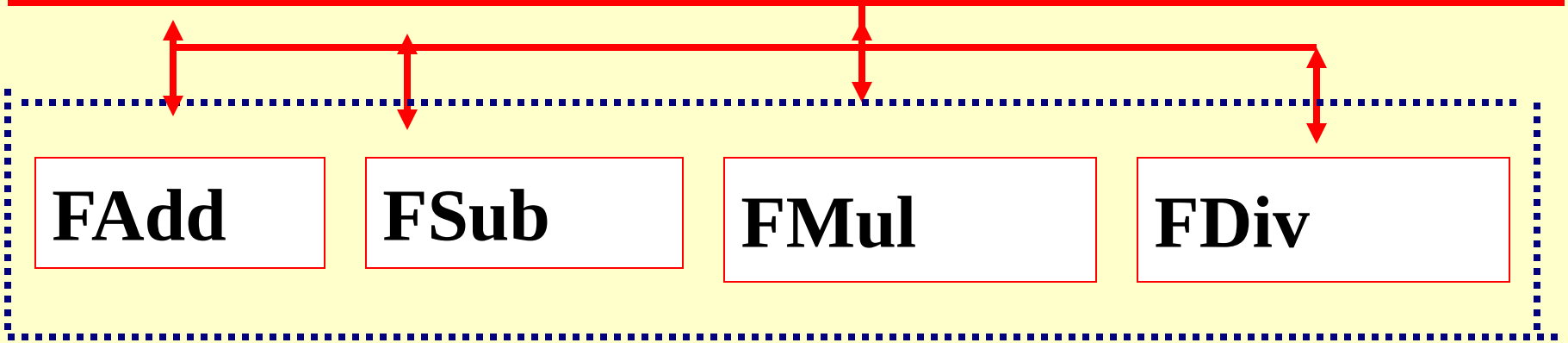
# ALU two types of operations

1. Fixed point operations
2. Floating point operations

# Fixed Point (Integer) ALU Operations

- Add
- Add with carry
- AND
- XOR
- Subtract
- Subtract with Borrow
- TEST
- NOT
- CMP
- OR
- L-Shift
- Divide
- Unsigned Divide
- R-Shift
- Unsigned Multiply
- Multiply
- Arth R-Shift

# Floating Point ALU Operations



# Complex ALU Operations

**MAC**

**VLIW operations**

---

# Common arithmetic operations

1. Addition, subtraction
2. Addition-with-carry (from a previous operation)

# Common arithmetic operations

3. Subtraction with borrow from a previous operation. Common circuit for addition and subtraction.

- ALU common flag for carry and borrow (many processors) or common flag for carry and not-borrow (8096)
- Carry flag an output carry from a chain of bit-adders

# Common arithmetic operations

4. Division
5. Increment and decrement
6. Logical shift left and logical shift right

# Logical Shift

- 0b00100111 after ‘logical left shift’

= 0b01001110

- 0b00100111 after ‘logical right shift’

= 0b00010011

# Common arithmetic operations

- Arithmetic shift left and logical shift right
- Arithmetic shift left and logical shift left same.
- Arithmetic shift right and logical shift right are different.

# Arithmetic Shift

- 0b00100111 after arth. left shift'

= 0b01001110

- 0b00100111 after arth. right shift'

= 0b00010011

- 0b10100111 after arth. right shift'

= 0b11010011

# Common logic operations

- NOT
- AND, OR, XOR
- COMPARE
- TEST

# Logic Operations

- 0b00100111 after NOT

= 0b11011000

- 0b00100111 and 0b11011001 after AND = 0b00000001
- 0b00100111 and 0b11011001 after OR = 0b11111111
- 0b00100111 and 0b11011001 after XOR = 0b11111110

# Compare Operations

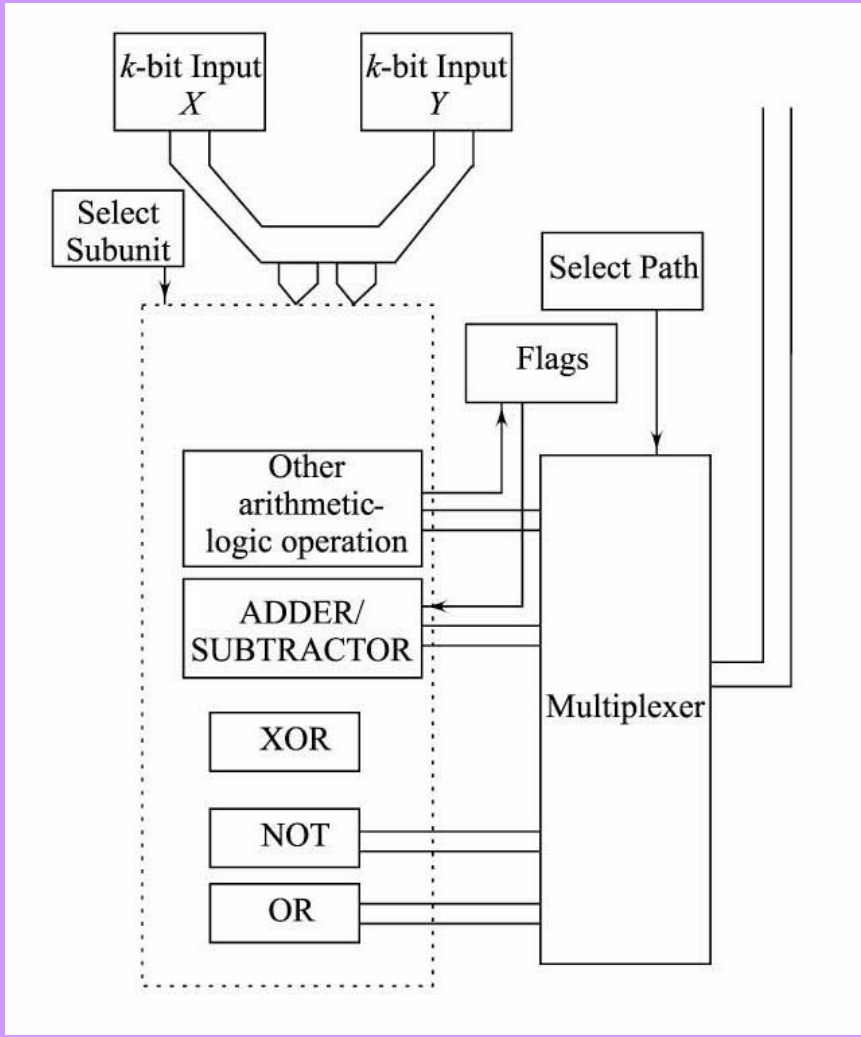
- 0b00100111 and 0b11011001 after operation Sign Flag = 1, Zero Flag = 0
- 0b11011001 and 0b 00100111 after operation Sign Flag = 0, Zero Flag = 0
- 0b 00100111 and 0b 00100111 after operation Sign Flag = 0, Zero Flag = 1

# TEST Operations

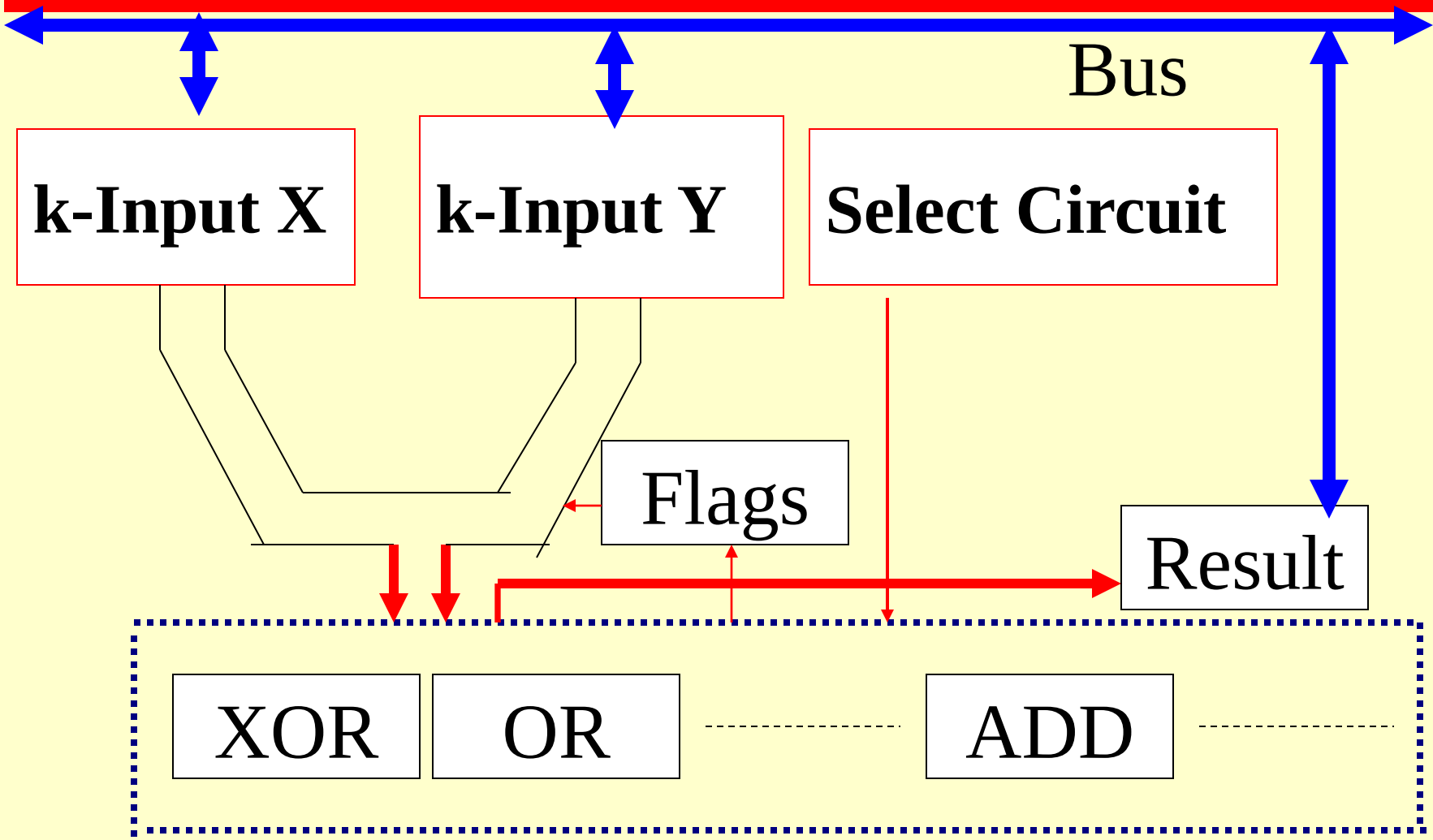
- 0b00100111 and 0b11011000 after operation Test Flag = 0, Zero Flag = 1
- 0b11011001 and 0b 11011001 after operation Test Flag = 1, Zero Flag = 0
- 0b 00100111 and 0b 00000111 after operation Test Flag = 0, Zero Flag = 0

# Combinational Logic Circuits-based ALUs

# An ALU using combinational circuits

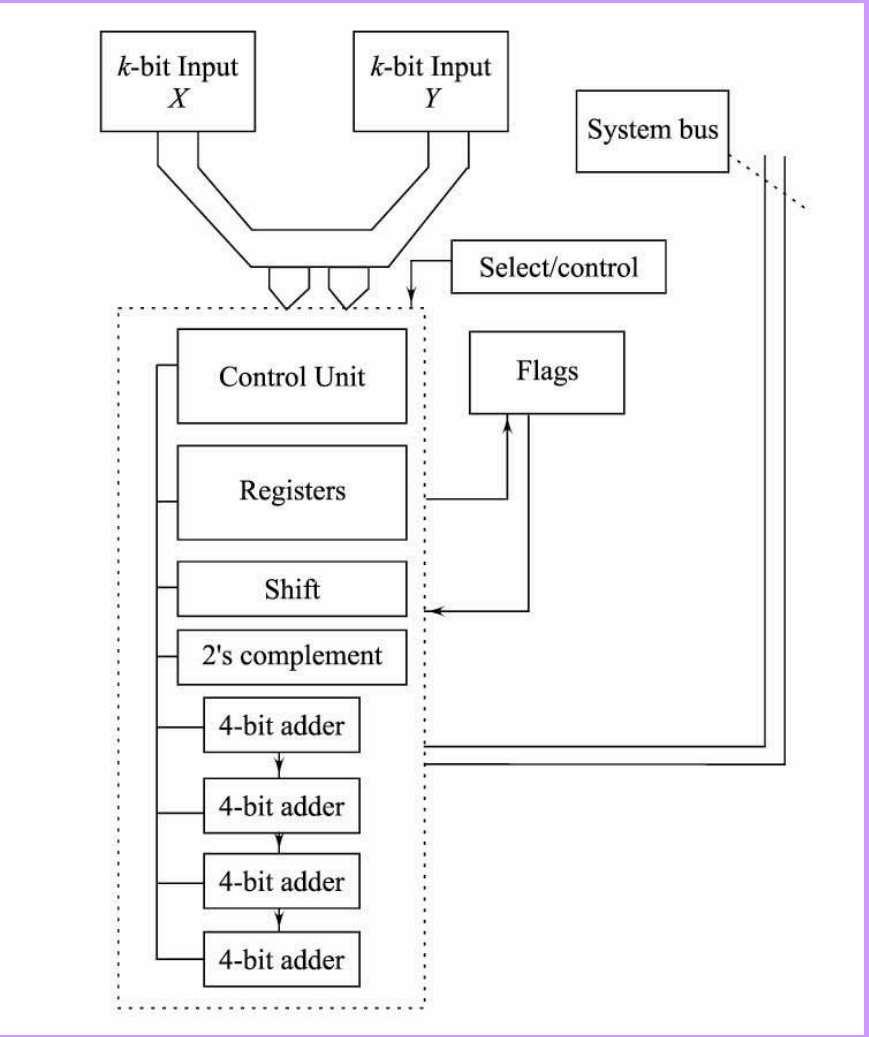


# Combinational Circuits Based ALU

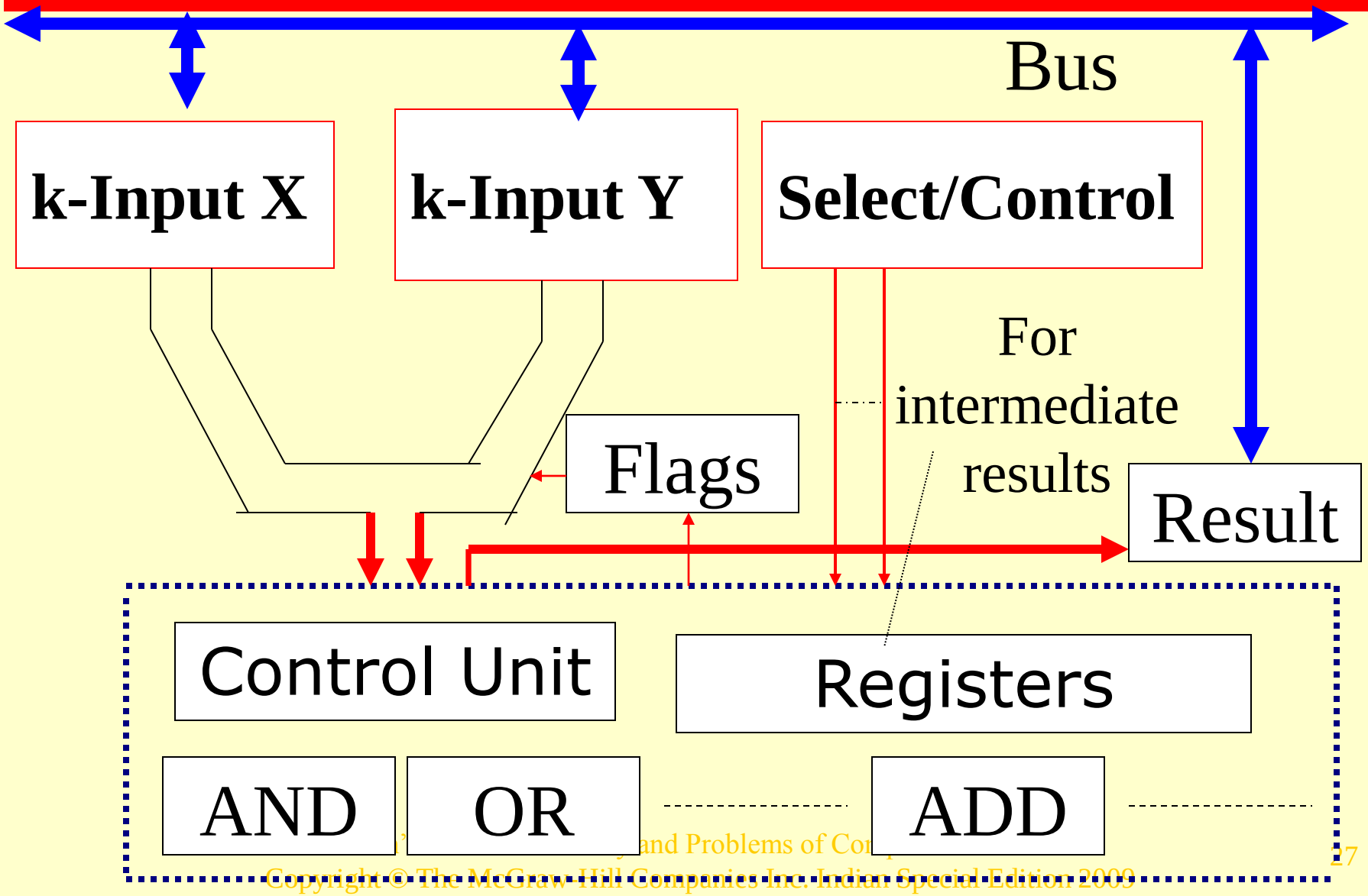


# Sequentias Circuits-based ALUs

# An ALU using sequential circuit

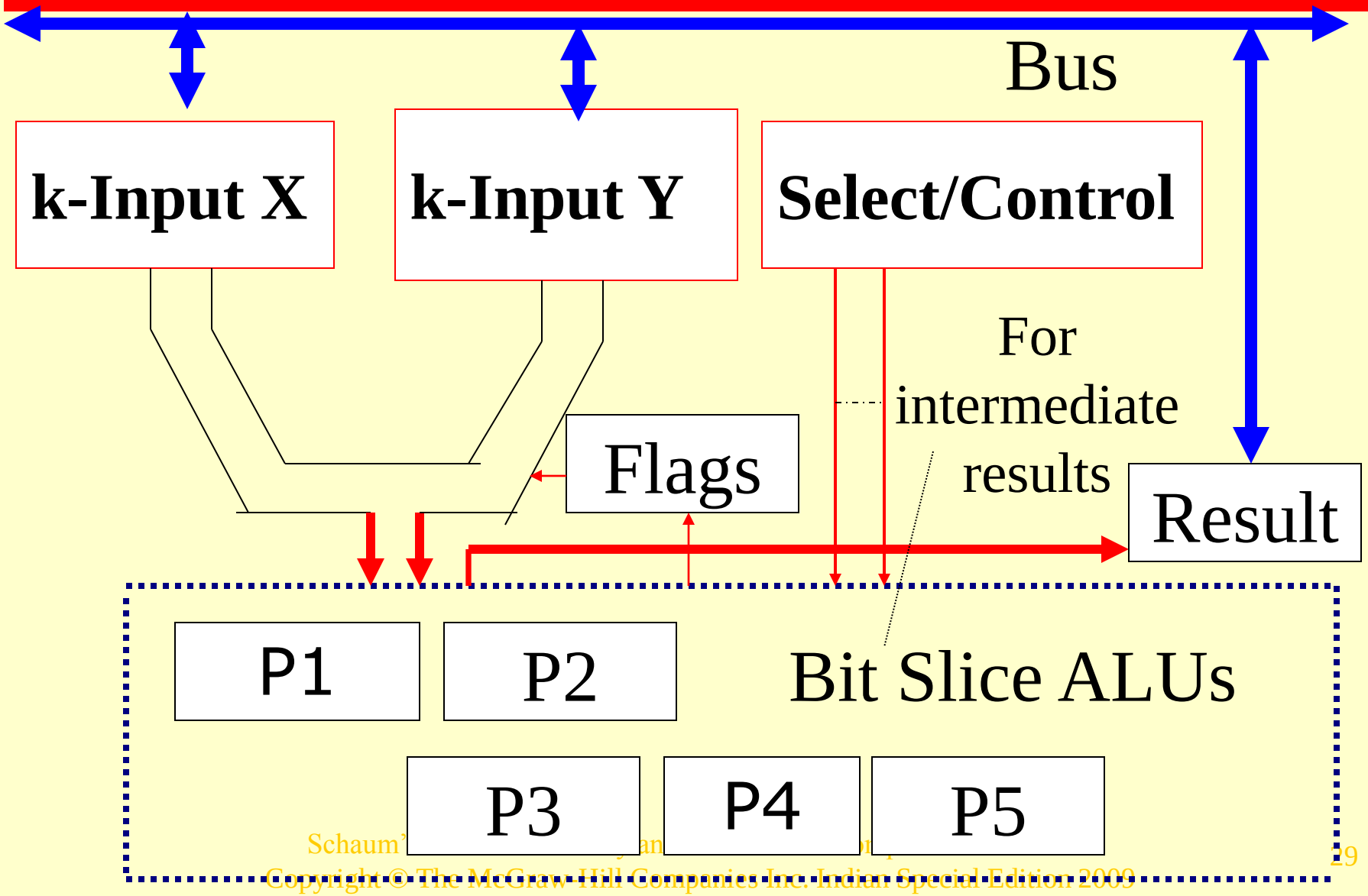


# Sequential Logic Circuits Based ALU



# Bit Slice Circuits-based ALUs

# Bit Slice Processors **Circuit Based ALU**



# Bit Slice

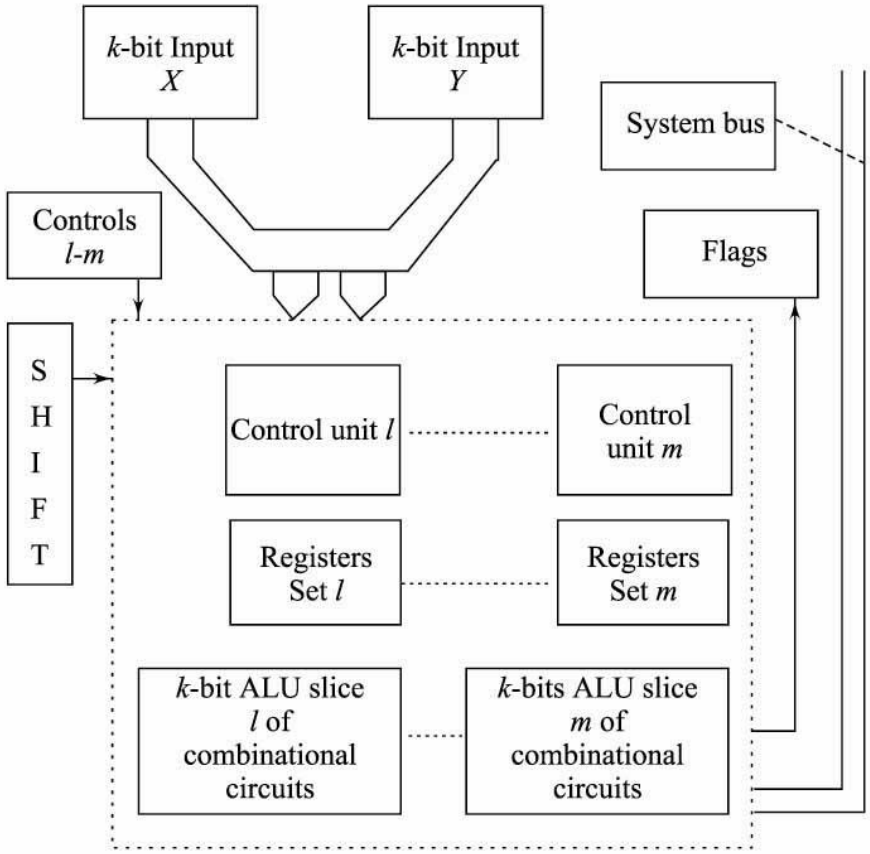



**Fig. 3.33** ***An ALU Using Bit-Slice Processors***

# Summary

# ALUs

- Fixed Point ALUs
- Floating Point Point ALUs
- Bit Slice Processors Based ALUs

# ALU Operations

- Add
- Add with carry
- AND
- XOR
- Subtract
- Subtract with Borrow
- TEST
- NOT
- CMP
- OR
- L-Shift
- Divide
- Unsigned Divide
- R-Shift
- Unsigned Multiply
- Multiply
- Arth R-Shift

# ALU Design

**Combinational Logic Circuits-based ALUs**

**Sequential Logic Circuits Based ALUs**

**Bit Slice Processors**

End of Lesson 11 on

**Design of ALU**