Chapter 05: Basic Processing Units … Control Unit Design Organization

Lesson 06:

Performing an Arithmetic or Logical Operation

Objective

- Learn how an arithmetic or logic operation performed by sequences of microoperations

Arithmetic or logic operation

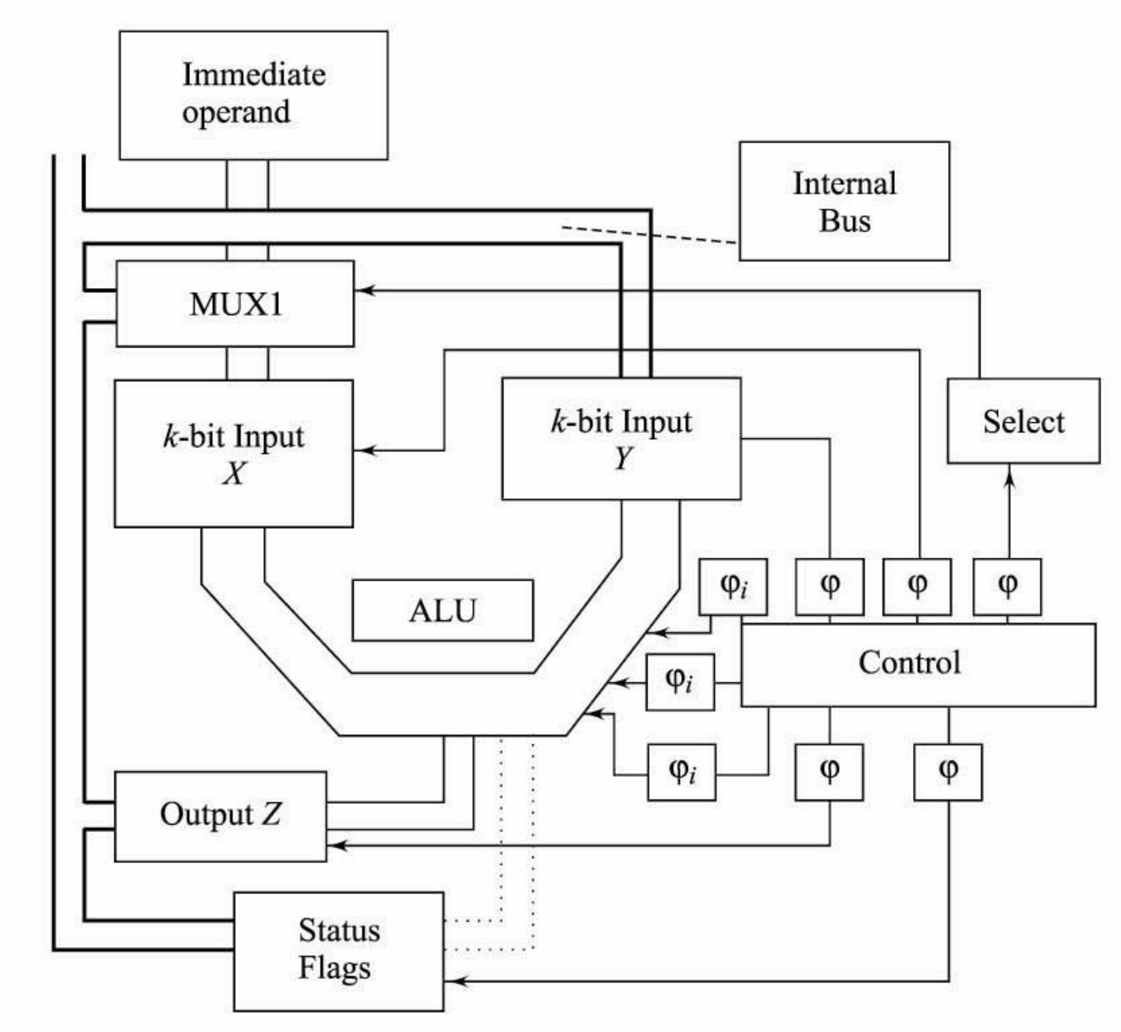
Execution of an ALU Instruction by Data Path Implementation

- Execution of an ALU instruction can be considered as the implementation of a specific data path flow, as per the specific ALU instruction
- The processing unit composition is as controlled data-path unit and control unit (controlling and sequencing unit)
- Control unit generates control signals to implement each step using signals ϕs

Sequence of actions to define the controlled transfers of data between processing subunits

- Processing subunits─ registers, *X*, *Y*, ALU circuits, *Z*, and status register along a required data path
- MUX unit to select one data path among several

ALU design as data path with a control unit for arithmetic or logic operation



Schaum's Outline of Theory and Problems of Computer Architecture

Microoperations after receiving the instruction at IR

- Decoded by decoding logic
- Then the logic results at register instruction decoder (ID) initiate control actions
- Each control signal selects an action through a gate input ϕ at each step

Microoperations for executing the instruction using ALU

1. Step i: Transfers a k-bit input source operand through the bus or immediate operand to X Input operand through a MUX → X

- MUX— a multiplexer to select one among several channels at inputs as per the select subunit signal

Microoperations for executing the instruction using ALU

2. *Step i* + 1: X transfers k-bit input X to ALU—
   $X \rightarrow \text{ALU}$

Microoperations for executing the instruction using ALU

3. *Step i* + 2: Another input operand transfers k-bits to *Y* (through bus)— Input operand →*Y*

Microoperations for executing the instruction using ALU

4. *Step i* + 3: Transfer *Y* to ALU— *Y* → ALU

Microoperations for executing the instruction using ALU

5. *Step i* + 4: ALU processing unit select through one of the gates ϕ_i an operation as per the arithmetic or logic instruction, which was received at the IR

Microoperations for executing the instruction using ALU

6. *Step i* + 5: Transfers a *k*-bit output *Z* from ALU—

$Z \leftarrow$ ALU

Microoperations for executing the instruction using ALU

7. *Step i* + 6: Transfers status flags generated, for example, carry or overflow to status register —

Status Register ← ALU

Microoperations for executing the instruction using ALU

8. *Step* *i* + 7: Transfers from *Z* the result to destination operand through bus—

$(\text{Bus}) \leftarrow Z$

ALU instruction

- Eight steps in Z ← operation (X and Y operands), Flags ← status of operation and Bus ← Z

ADD operation

Add instruction

1. Input operand through a MUX → X
2. X → ALU-input
3. Input operand → Y
4. Y → ALU-input

Add instruction

5. Selects adder gate ϕ_{add} when ADD instruction received at the IR and decoded at the instruction decoder logic

- ϕ_{add} : ALU-output (0-k–1) ← X + Y

6. Z ← ALU-output (0-k–1)

7. Status Register ← ALU (status)

8. (Bus) ← Z

SUB instruction

1. Input operand through a MUX → X
2. X → ALU-input
3. Input operand → Y
4. Y → ALU-input

SUB operation

SUB instruction

5. Selects subtractor gate ϕ_{sub} when SUB instruction received at the IR and decoded at the instruction decoder logic

- ϕ_{sub}: ALU-output (0-k–1) ← ALU ← X + NOT(Y) + 1 for finding X – Y

6. Z ← ALU-output (0-k–1)

7. Status Register ← ALU (status)

8. (Bus) ← Z

Control Signal for selecting an ALU operation

ALU control input during an interval T step 5 for an Arithmetic Operation

- One active C_{alu} among eight ϕs for eight arithmetic operations

1. ϕ_{add} : $ALU \leftarrow X + Y$
2. ϕ_{sub} : $ALU \leftarrow X + NOT(Y) + 1$
3. ϕ_{adc} : $ALU \leftarrow X + Y + CY$
4. ϕ_{sbb} : $ALU \leftarrow X + NOT(Y) + 1 + Cy$

ALU control input during an interval T step 5 for an Arithmetic Operation

5. ϕ_{mul}: ALU ← X × Y
6. ϕ_{div} : ALU ← X ÷ Y
7. ϕ_{inc} : temp_Y ← 1; ALU ← X + temp_Y
8. ϕ_{dec} : temp_Y ← 1; ALU ← X + NOT(temp_Y) + 1

ALU control input during an interval T step 5 for a Logic Operation

- One active C_{alu} among six ϕs for six logic operations

1. ϕ_{XOR} : ALU ← X .XOR. Y
2. ϕ_{OR} : ALU ← X .OR. Y
3. ϕ_{AND} : ALU ← X .AND.Y
4. ϕ_{NOT} : ALU ← NOT (X)

ALU control input during an interval T step 5 for a Logic Operation

5. ϕ_{clear}: ALU ← all 0s
6. ϕ_{set} : ALU ← all 1s

Summary

We learnt

- An arithmetic or logic operation performed by sequences of microoperations
- Eight steps for an ALU operation among 8 arithmetic and six logic operations

End of Lesson 06 on

Performing an Arithmetic or Logical Operation