

---

# **CSE 161 – Design and Architecture of Computer Systems**

## **Chapter Five**

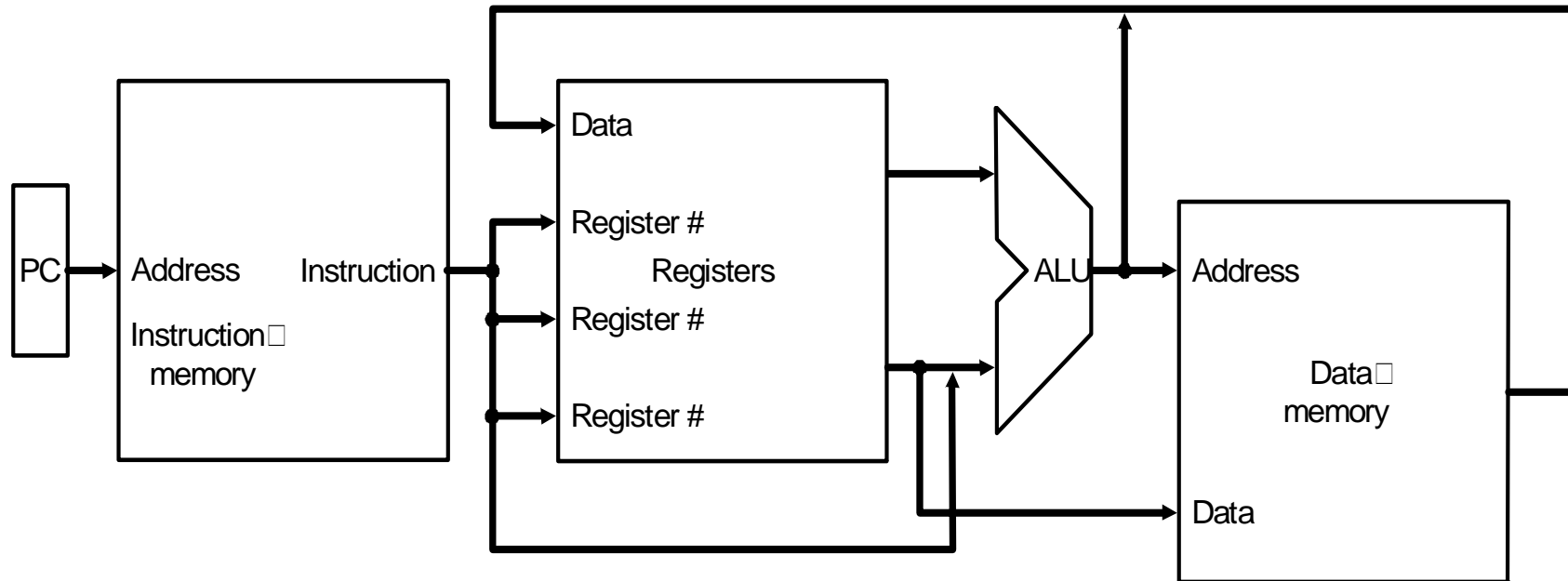
# The Processor: Datapath & Control

---

- ❑ We're ready to look at an implementation of the MIPS
- ❑ Simplified to contain only:
  - memory-reference instructions: lw, sw
  - arithmetic-logical instructions: add, sub, and, or, slt
  - control flow instructions: beq, j
- ❑ Generic Implementation:
  1. use the program counter (PC) to supply instruction address
  2. get the instruction from memory
  3. read registers
  4. use the instruction to decide exactly what to do
- ❑ All instructions use the ALU after reading the registers  
Why? memory-reference? arithmetic? control flow?

# More Implementation Details

## □ Abstract / Simplified View:



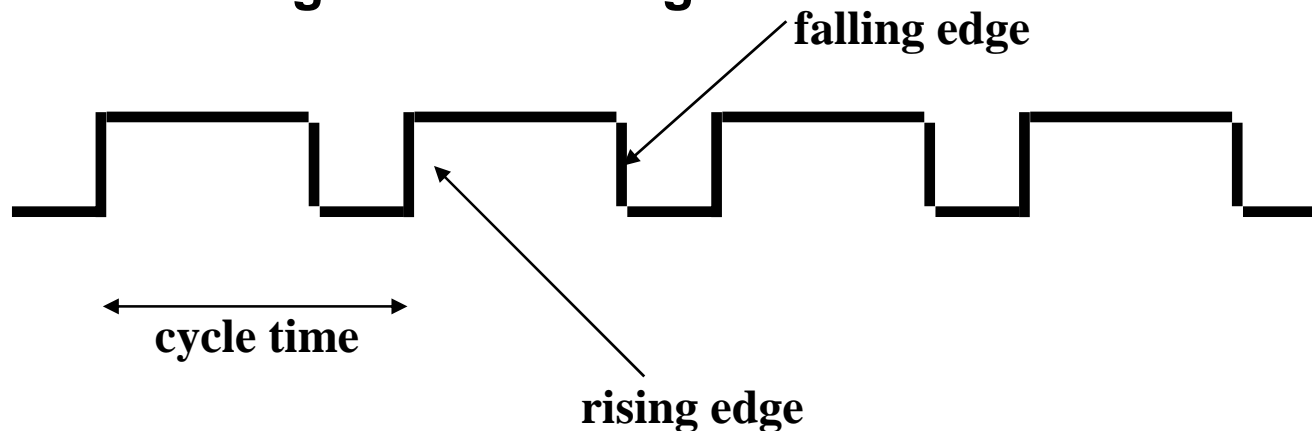
## □ Two types of functional units:

- **elements that operate on data values (combinational)**
  - ALU, multiplier
- **elements that contain state (sequential)**
  - Instruction/data memories, registers

# State Elements

---

- ❑ **Unclocked vs. Clocked**
- ❑ **Clocks used in synchronous logic**
  - **When should an element that contains state be updated?**
  - **Edge-triggered clocking – state elements all update their internal storage on clock edges**



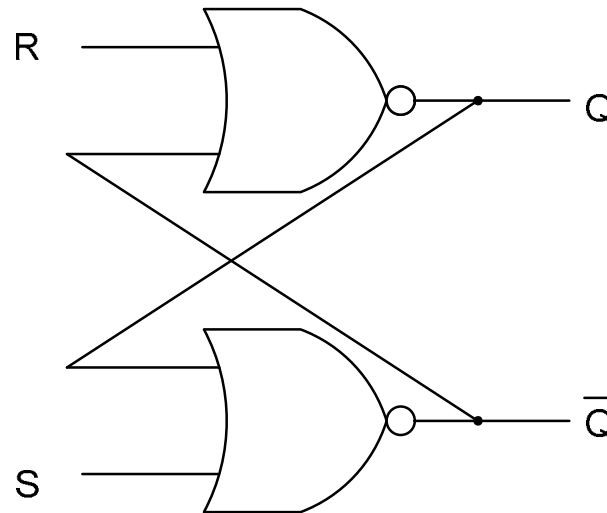
- **Any combinational logic must have its inputs coming from a set of state ele. and its outputs written into a set of state ele.**

# An unclocked state element

---

## □ The set-reset latch

- output depends on present inputs and also on past inputs



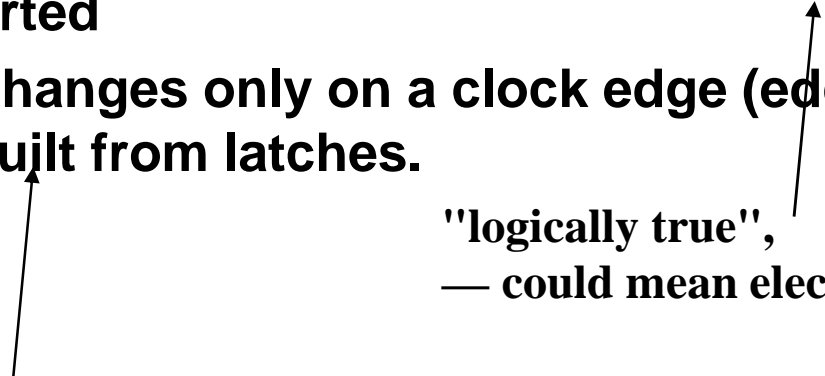
*S-R latch: S and R can not be simultaneously asserted*

# Latches and Flip-flops: simplest mem. elements

---

- ❑ Output is equal to the stored value inside the element  
(don't need to ask for permission to look at the value)
- ❑ Change of state (value) is based on the clock
- ❑ Latches: state is changed whenever the inputs change, and the clock is asserted
- ❑ Flip-flop: state changes only on a clock edge (edge-triggered methodology). Built from latches.

"logically true",  
— could mean electrically low

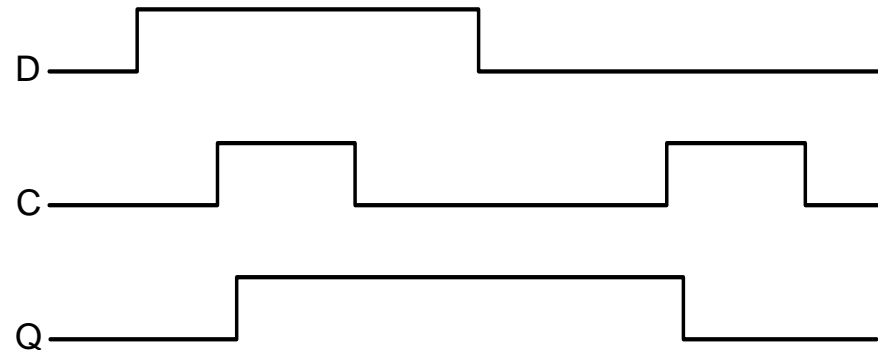
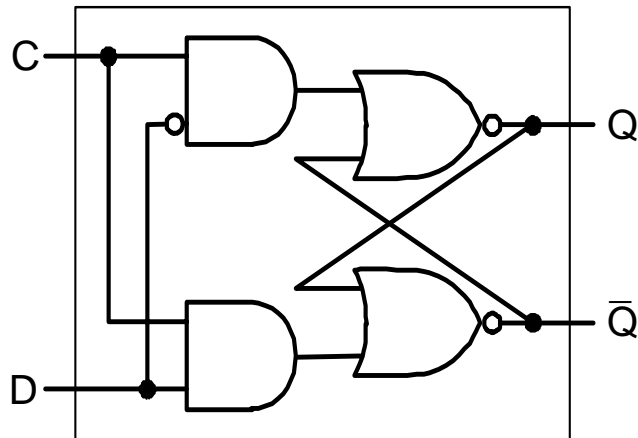


A clocking methodology defines when signals can be read and written  
— wouldn't want to read a signal at the same time it was being written

# D-latch

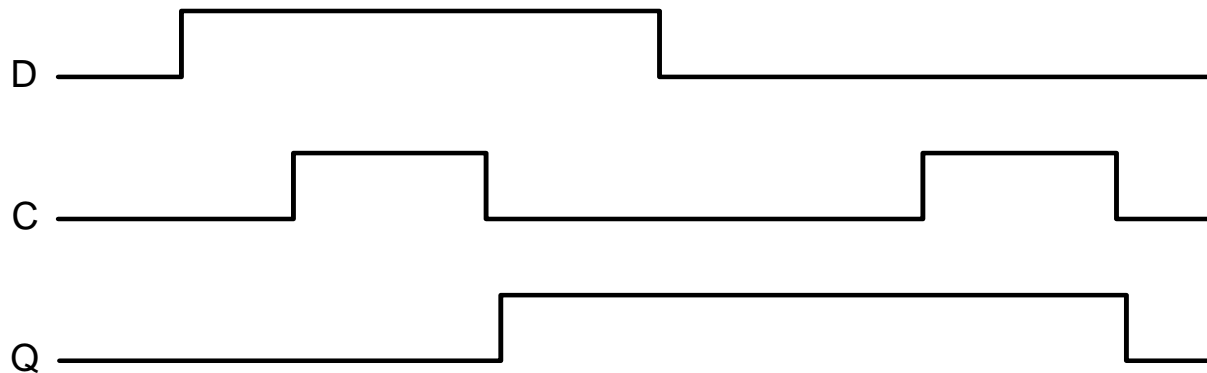
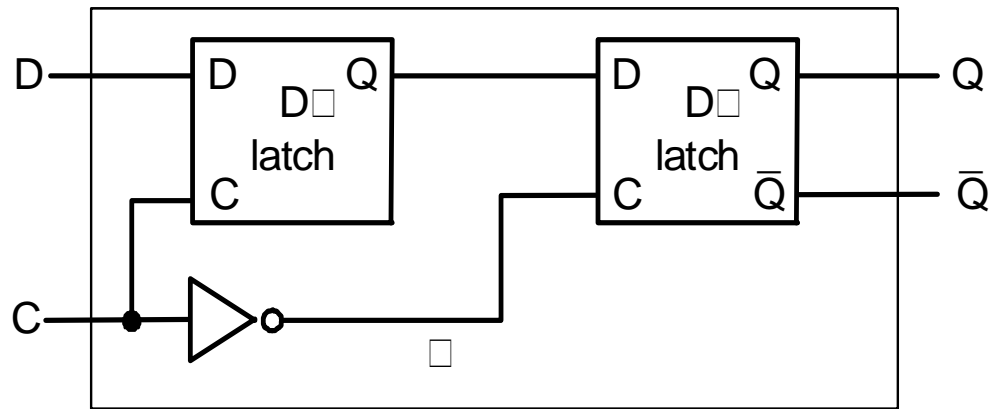
---

- ❑ Two inputs:
  - the data value to be stored (D)
  - the clock signal (C) indicating when to read & store D
- ❑ Two outputs:
  - the value of the internal state (Q) and it's complement



# D flip-flop

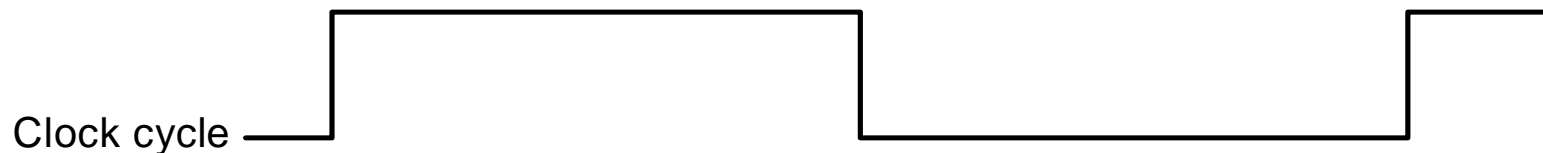
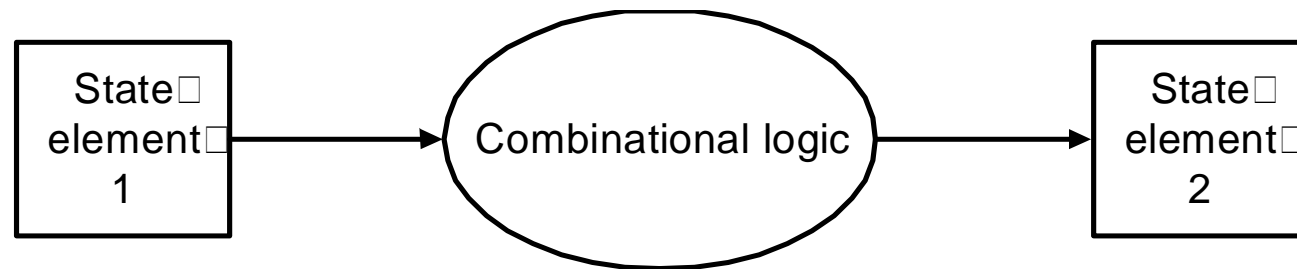
- Output changes only on the clock edge



# Our Implementation

---

- ❑ An edge triggered methodology
- ❑ Typical execution:
  - read contents of some state elements,
  - send values through some combinational logic
  - write results to one or more state elements



# Exercise

---

□ Draw the timing diagram of Q for a D-latch

