中国科学院大学
University of Chinese Academy of Sciences

B0911006Y-01 2024-2025学年春季学期

计算机组成原理

Principles of Computer Organization

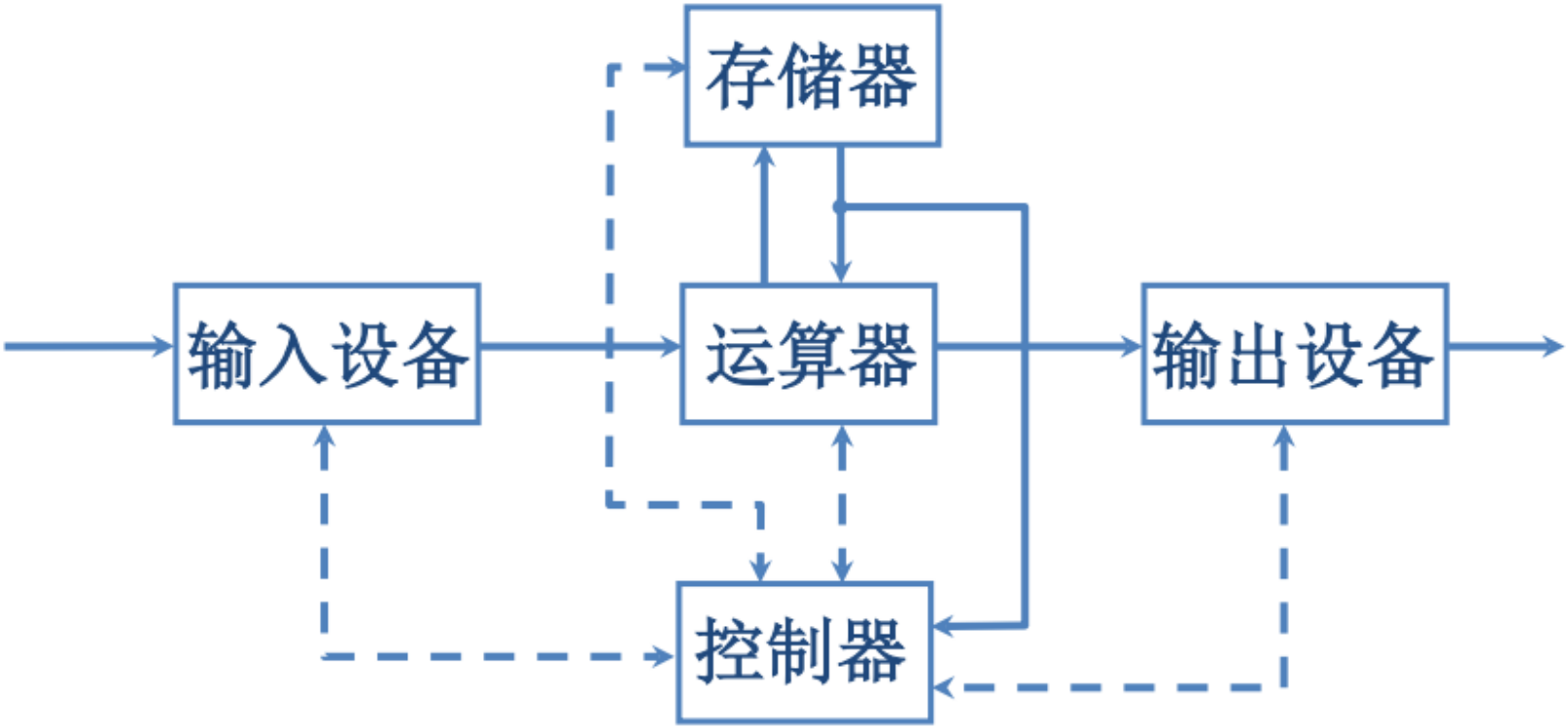
单周期处理器 I

基本组件，如何设计、组装数据通路

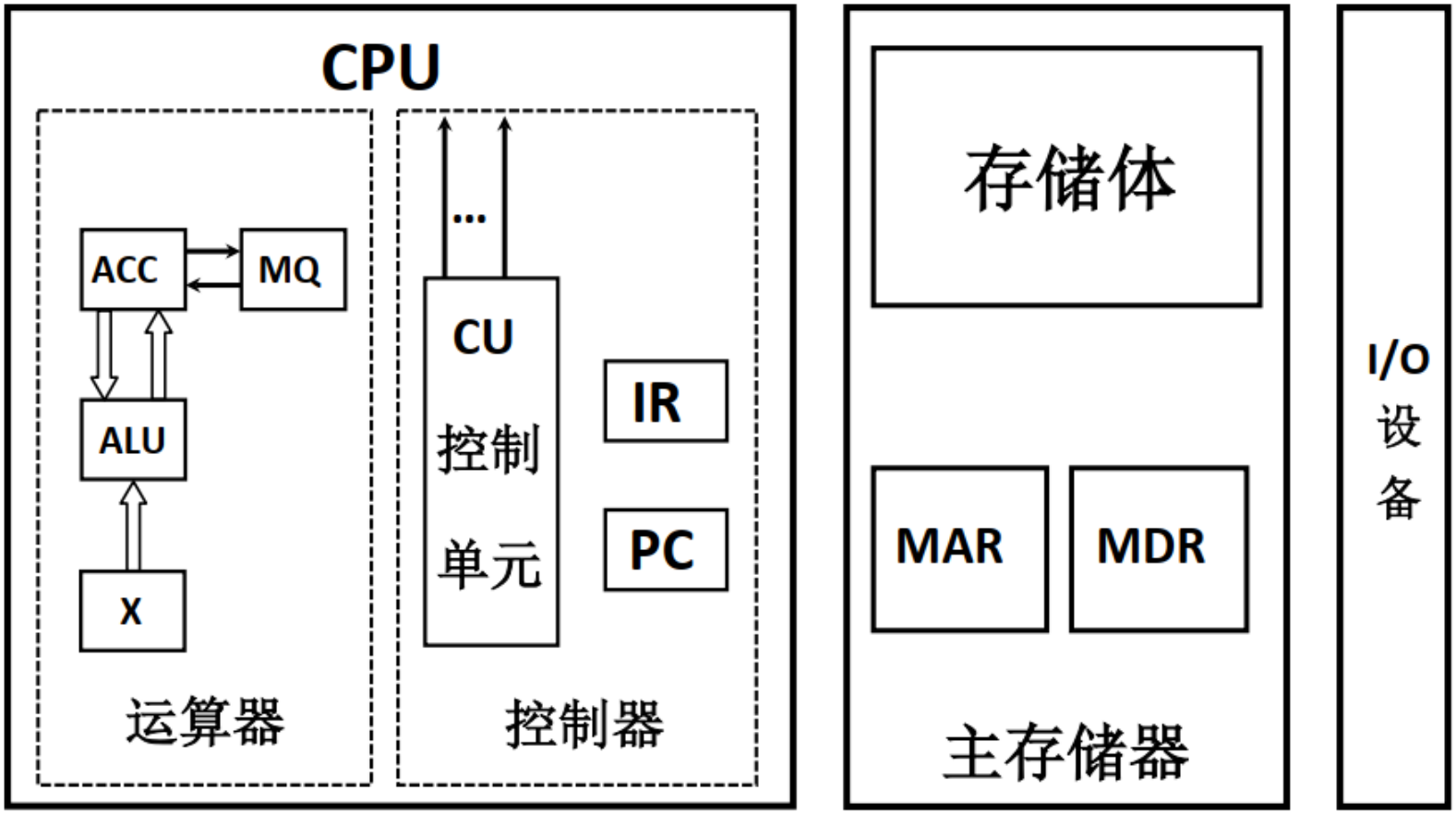
主讲教师：石 侃
shikan@ict.ac.cn

2025年4月9日

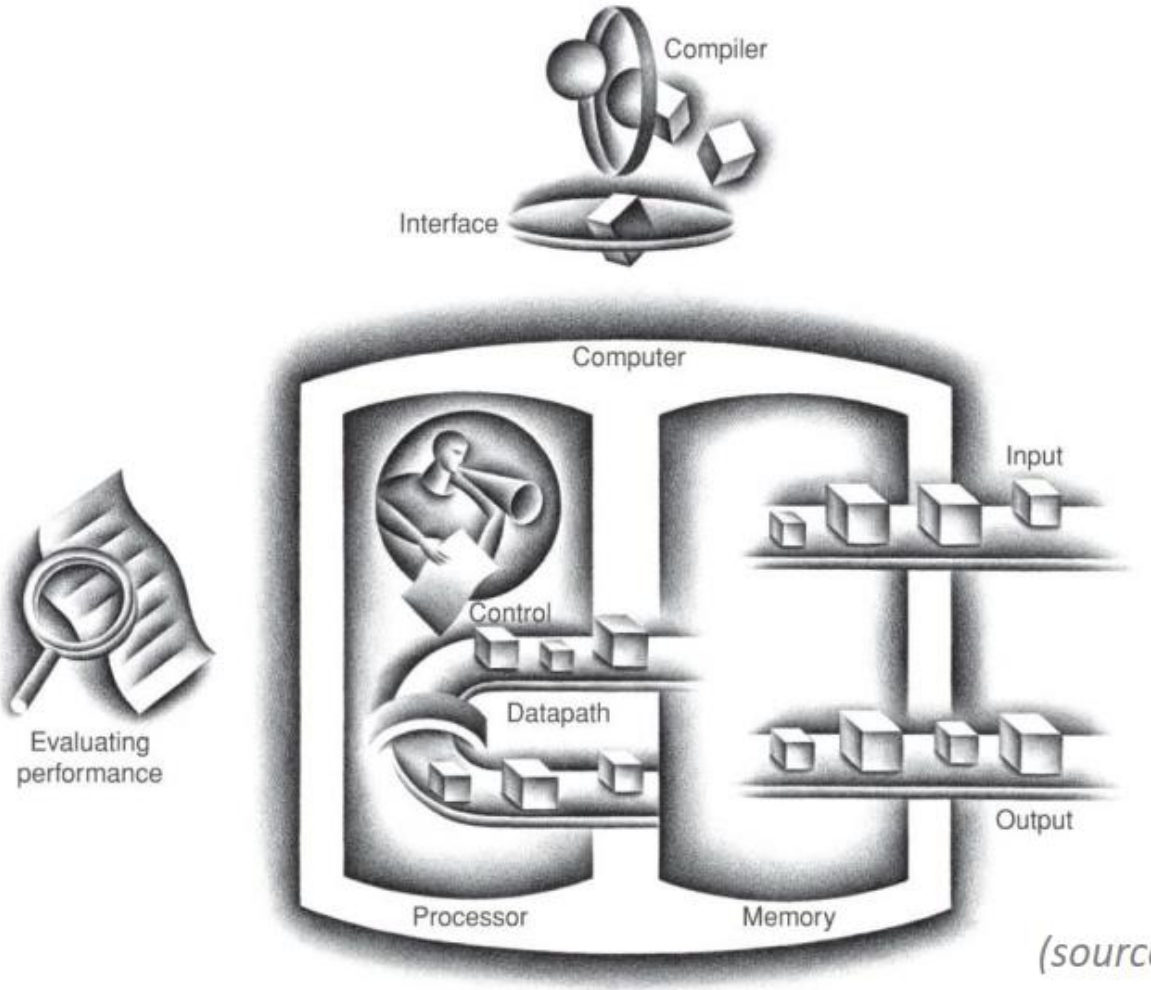
回顾：冯诺依曼计算机五大部件



回顾：冯诺依曼计算机五大部件



回顾：冯诺依曼计算机五大部件



(source: P&H-COD)

The Processor

- **Processor (CPU)**: Implements the instructions of the Instruction Set Architecture (ISA)

The Processor

- **Processor (CPU)**: Implements the instructions of the Instruction Set Architecture (ISA)
  - *Datapath*: part of the processor that contains the hardware necessary to perform operations required by the processor ("the brawn")
    - 指令执行过程中，数据所经过的路径，包括路径中的部件
    - 是指令执行的部件
    - 组合元件和存储元件通过总线或分散方式连接而成的进行数据存储、处理和传送的路径。

The Processor

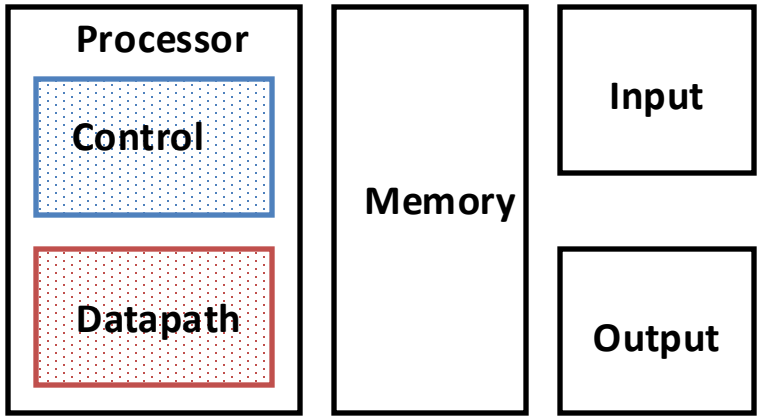
- **Processor (CPU)**: Implements the instructions of the Instruction Set Architecture (ISA)
  - *Datapath*: part of the processor that contains the hardware necessary to perform operations required by the processor ("the brawn")
    - 指令执行过程中，数据所经过的路径，包括路径中的部件
    - 是指令执行的部件
    - 组合元件和存储元件通过总线或分散方式连接而成的进行数据存储、处理和传送的路径。
  - *Control*: part of the processor (also in hardware) which tells the datapath what needs to be done ("the brain")
    - 对指令进行译码，生成指令对应的控制信号，控制数据通路的动作
    - 是指令的控制部件，对执行部件发出控制信号

Processor Design Process

- Five steps to design a processor:

Datapath

- 1. Analyze instruction set → datapath requirements
- 2. Select set of datapath components & establish clock methodology
- 3. Assemble datapath meeting the requirements

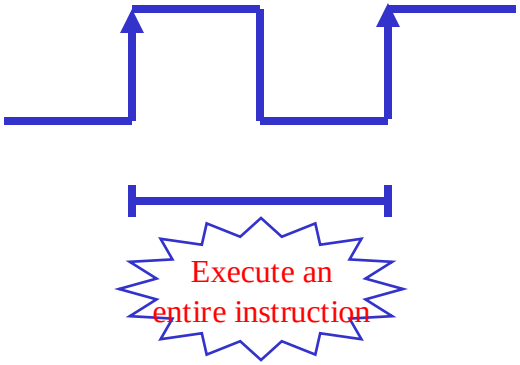


Control

- 4. Analyze implementation of each instruction to determine setting of control points that effects the register transfer
- 5. Assemble the control logic
  - Formulate Logic Equations
  - Design Circuits

The Big Picture: The Performance Perspective

- Processor design (datapath and control) will determine:
  - Clock cycle time
  - Clock cycles per instruction
- Starting today:
  - Single cycle processor:
    - Advantage: One clock cycle per instruction
    - Disadvantage: long cycle time



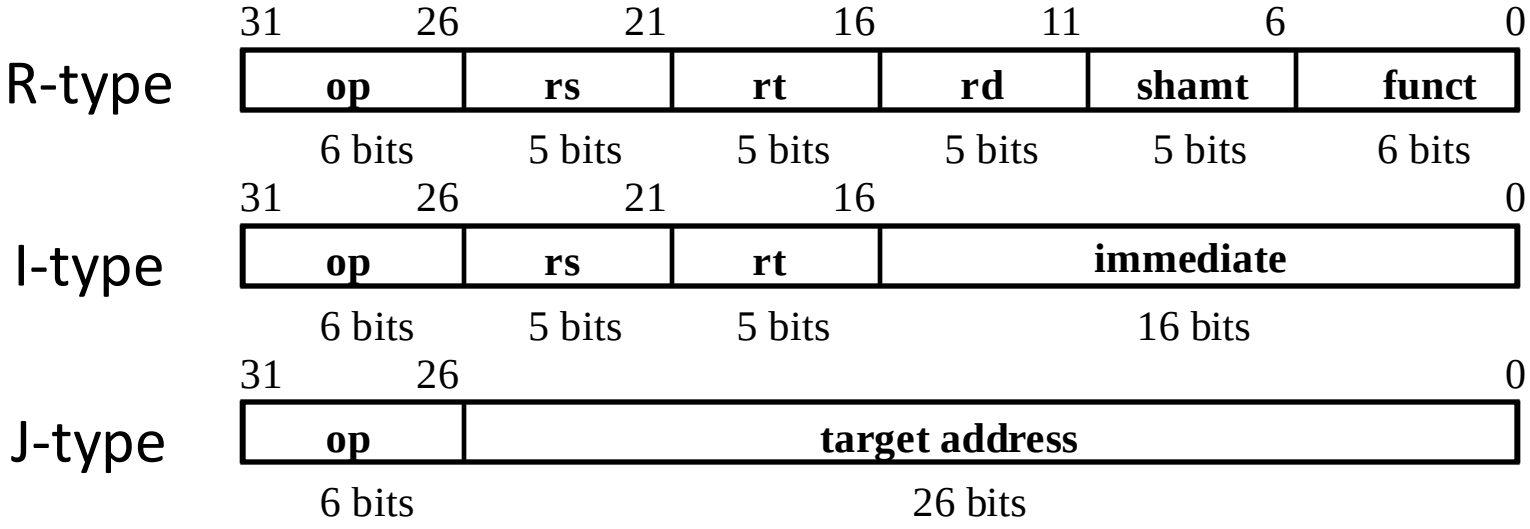
- CPUTime(ET) = IC× CPI × Cycle Time
  - 指令数目由编译器和ISA决定
  - 时钟周期和CPU由CPU的设计和实现决定

Processor Datapath and Control

- We‘re ready to look at an implementation of a simplified MIPS CPU contains only:
  - memory-reference instructions: lw, sw
  - arithmetic-logical instructions: add, sub, and, or, slt
  - control flow instructions: beq
- Generic Implementation:
  - use the program counter (PC) to supply instruction address
  - get the instruction from memory
  - read registers
  - use the instruction to decide exactly what to do
- Which instructions will use the ALU after register reading?
  - memory-reference? arithmetic? control flow?
  - **ALL of THESE**

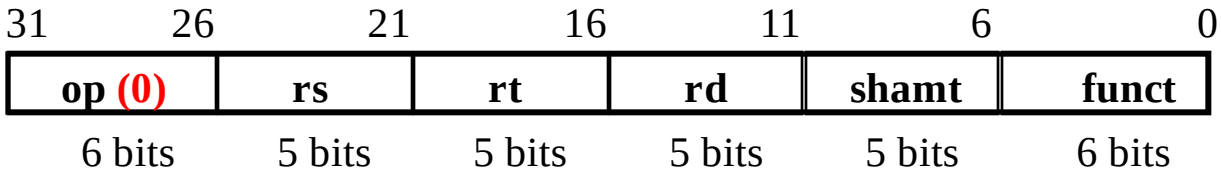
MIPS Instruction Formats

- 无内部互锁流水级的微处理器
  (Microprocessor without Interlocked Piped Stages)
- All instructions 32-bits long
- 3 Formats:

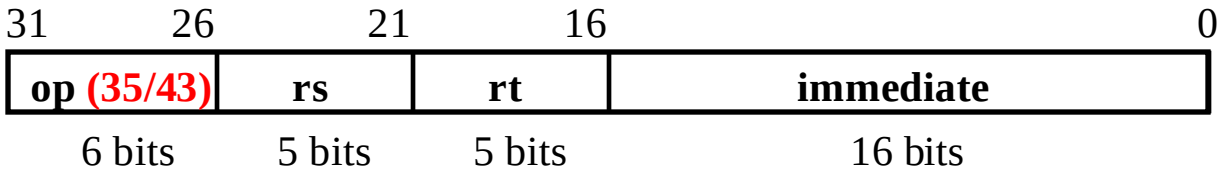


The MIPS Subset

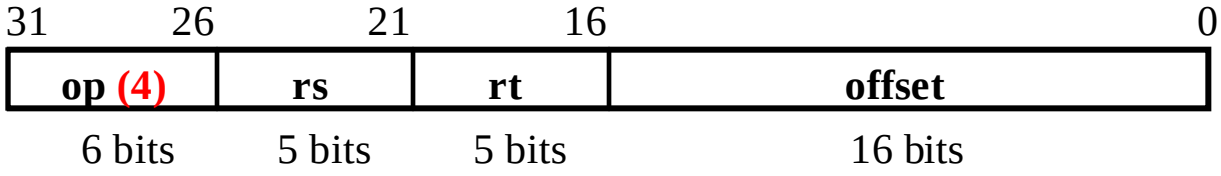
- R-Type
  - add rd, rs, rt
  - sub, and, or, slt



- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16



- BRANCH:
  - beq rs, rt, imm16



Basic Steps of Execution

- Instruction Fetch
  - Where is the instruction?

Instruction memory
address: PC

- Decode
  - What's the incoming instruction?
  - Where are the operands in an instruction?

Register file

- Execution: ALU
  - What is the function that ALU should perform?

ALU

- Memory access
  - Where is my data?

Data memory
address: effective address

- Write back results to registers
  - Where to write?

Register file

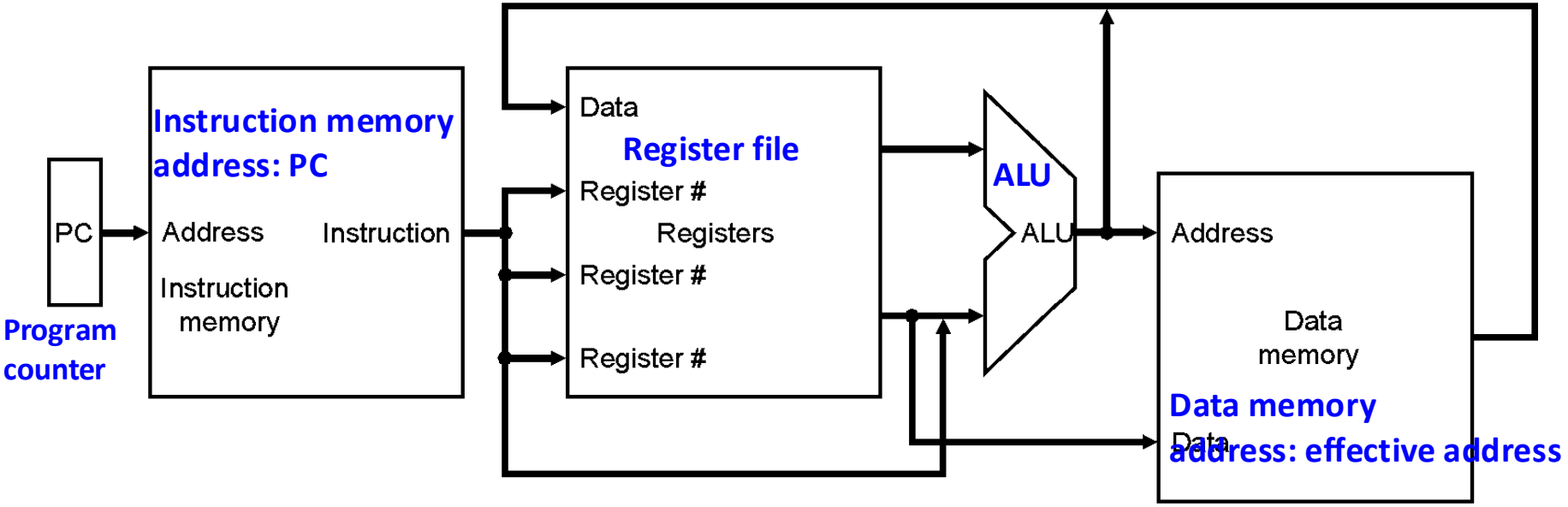
- Determine the next PC

Program counter

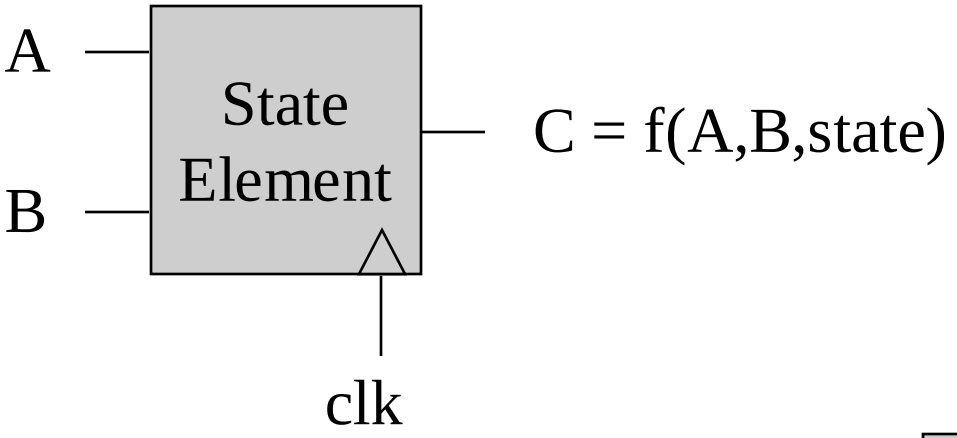
Generic Mechanism of Processor

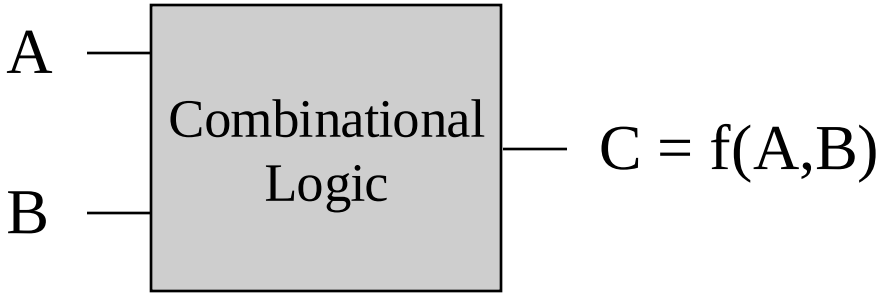
- Use Program Counter (PC) to supply an instruction address
- Get the instruction from memory
- Use the instruction to decide (control) exactly which register(s) to read or write
- Use the instruction to decide (control) exactly what operation(s) to execute

Where We're Going: The High-level View



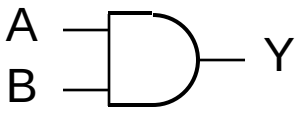
Review: Two Logical Components



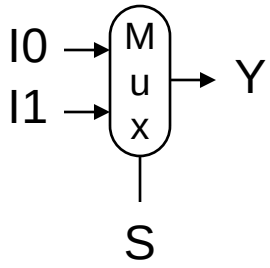


Combinational Elements

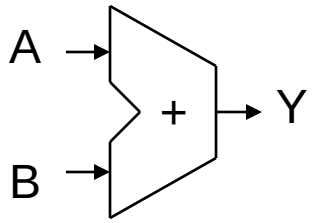
- AND-gate
  - Y = A & B



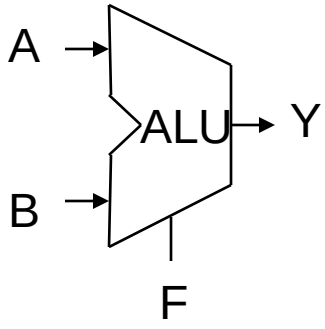
- Multiplexer
  - Y = S ? I1 : I0



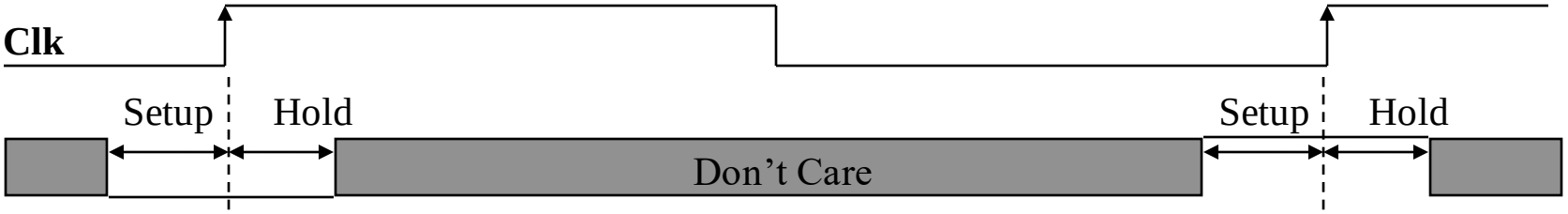
- Adder
  - Y = A + B



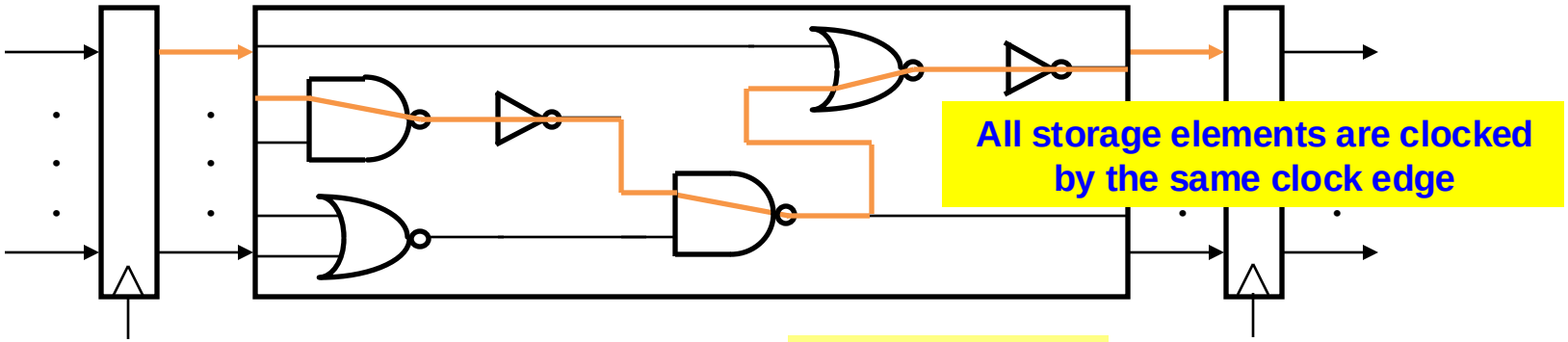
- Arithmetic/Logic Unit
  - Y = F(A, B)



Clocking Methodology (定时方法)



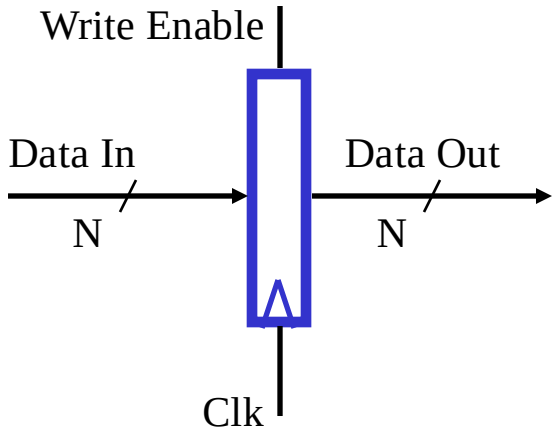
- Setup Time: how long the input must be stable before the CLK trigger for proper input read
- Hold Time: how long the input must be stable after the CLK trigger for proper input read
- CLK-to-Q Delay（锁存延迟）: how long it takes the output to change, measured from the CLK trigger

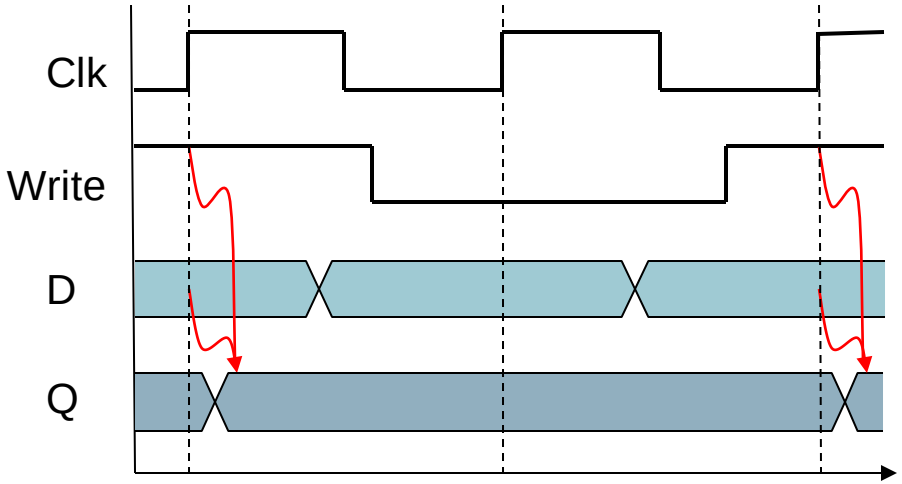


- The critical path is the longest delay between any two registers in a circuit
- Critical path determines length of clock period
  - The clock period must be longer than this critical path, or the signal will not propagate properly to that next register
  - This includes CLK-to-Q delay and setup delay

Storage Element: The Register

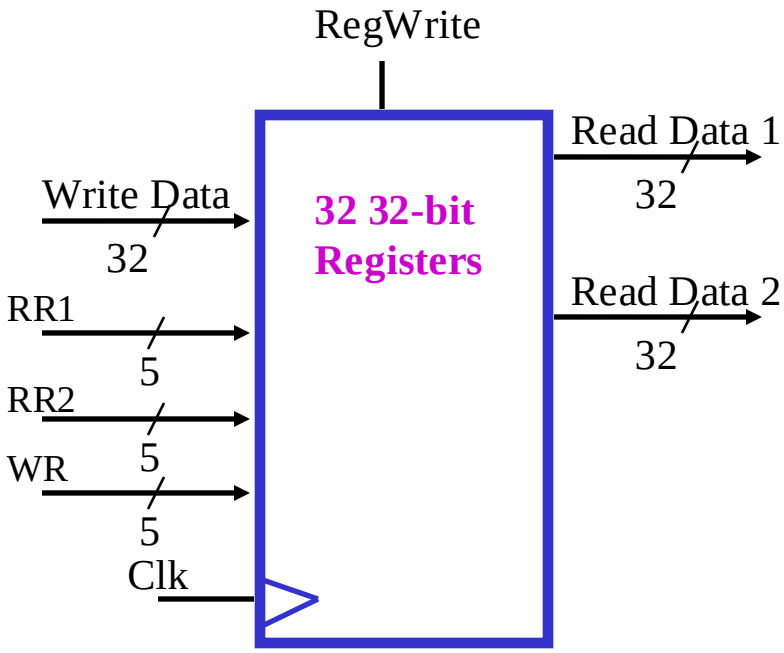
- Register
  - Similar to the D Flip Flop except
    - N-bit input and output
    - Write Enable input
- Write Enable:
  - 0: Data Out will not change
  - 1: Data Out will become Data In (on the clock edge)
    - Only updates on clock edge when write control input is 1
    - Used when stored value is required later





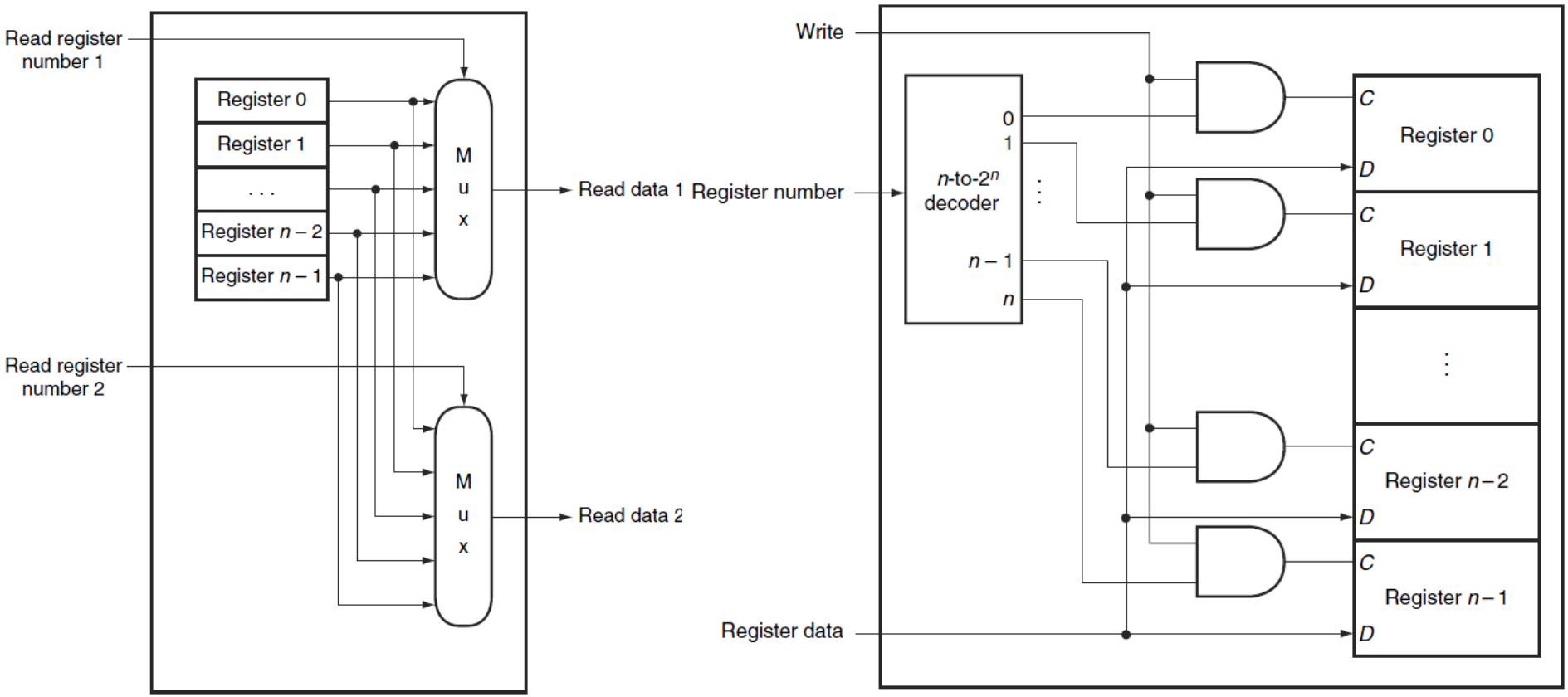
Storage Element: Register File

- Register File consists of (32) registers:
  - Two 32-bit output buses
  - One 32-bit input bus
- Register is selected by:
  - RR1 selects the register to put on bus “Read Data 1”
  - RR2 selects the register to put on bus “Read Data 2”
  - WR selects the register to be written
  - via WriteData when RegWrite is 1
- Clock input (CLK)



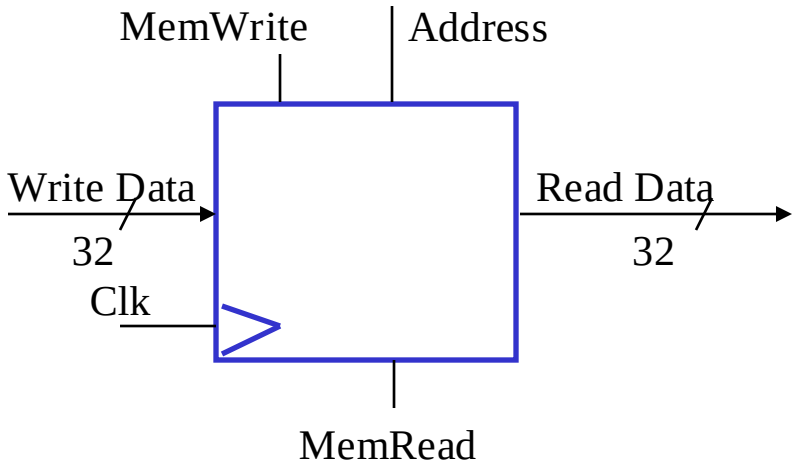
Inside the Register File

- The implementation of two read ports register file
  - n registers
  - done with a pair of n-to-1 multiplexors, each 32 bits wide.



Storage Element: Memory

- Memory
  - Two input buses: WriteData, Address
  - One output bus: ReadData
- Memory word is selected by:
  - Address selects the word to put on ReadData bus
  - If MemWrite = 1: address selects the memory word to be written via the WriteData bus
- Clock input (CLK)
  - The CLK input is a factor ONLY during write operation
  - During read operation, behaves as a combinational logic block:
    - Address valid => ReadData valid after "access time."



RTL: Register Transfer Language

- Describes the movement and manipulation of data between storage elements:
  - R[i]表示寄存器堆中寄存器i的内容
  - M[addr]表示存储单元addr的内容
  - 传送方向用<- 表示，传送源在右目的在左
  - 程序计数器PC直接用PC表示其内容

R[3] <- R[5] + R[7]

PC <- PC + 4 + R[5]

R[rd] <- R[rs] + R[rt]

R[rt] <- Mem[R[rs] + immed]