Computer Architecture

Ngo Lam Trung & Pham Ngoc Hung, Hoang Van Hiep
Department of Computer Engineering
School of Information and Communication Technology (SoICT)
Hanoi University of Science and Technology
E-mail: [trungnl, hungpn, hiephv]@soict.hust.edu.vn

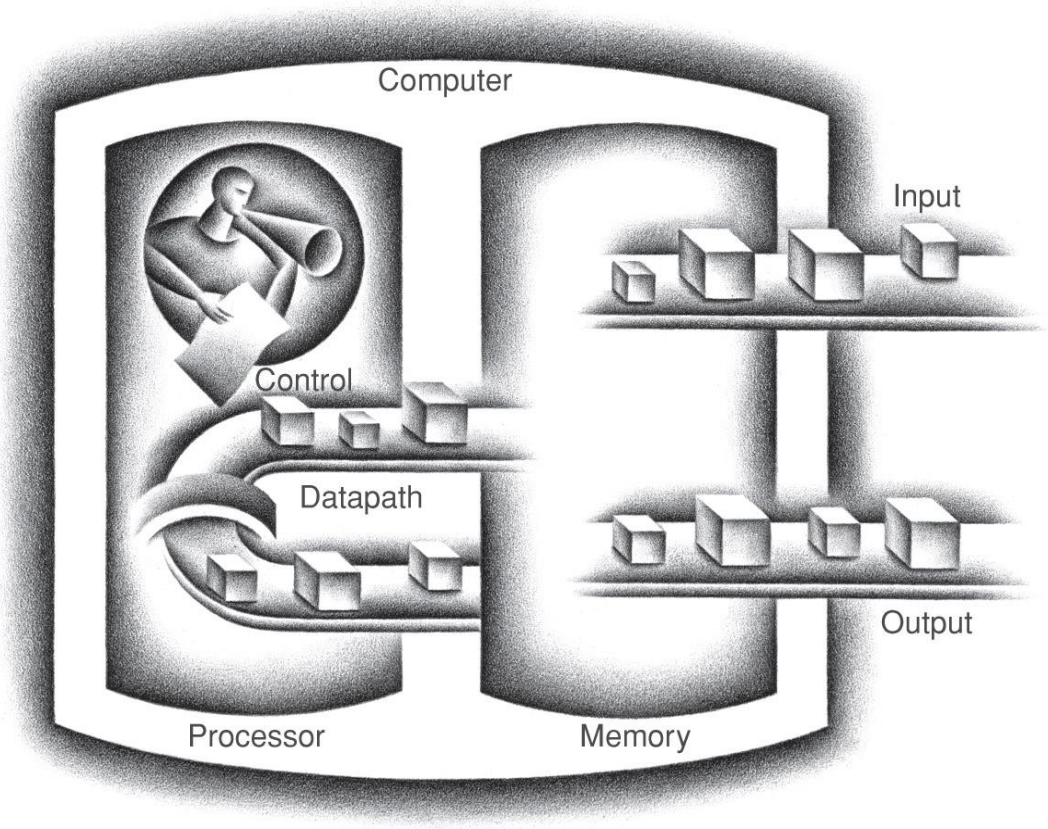
Chapter 2: Top-level view of Computer Functions and Interconnection

1. Computer Components
2. Computer Functions
3. Inter-connection Structures

[with materials from *Computer Organization and Architecture, 10th Edition*, William Stallings, ©2016, Pearson]

Computer Organization

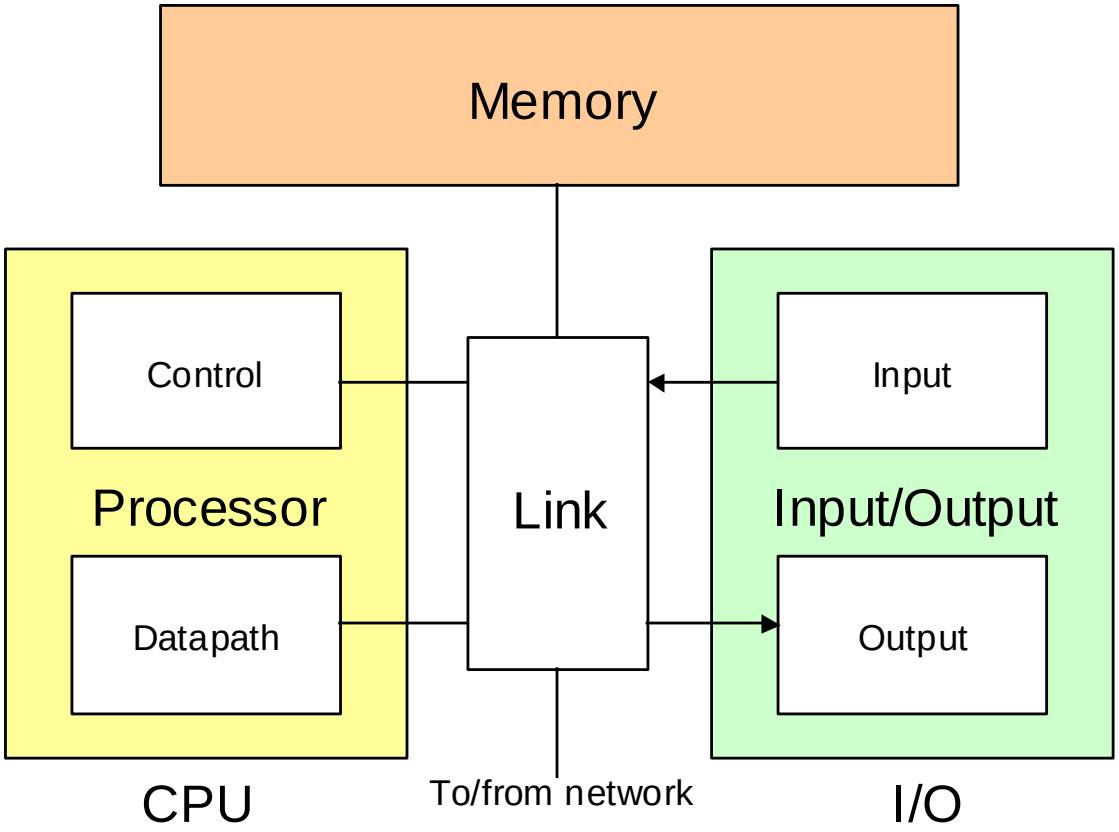
- From Chap.1: classic components of a computer



- Computer
  - Input data
  - Execute stored programs
  - Output result

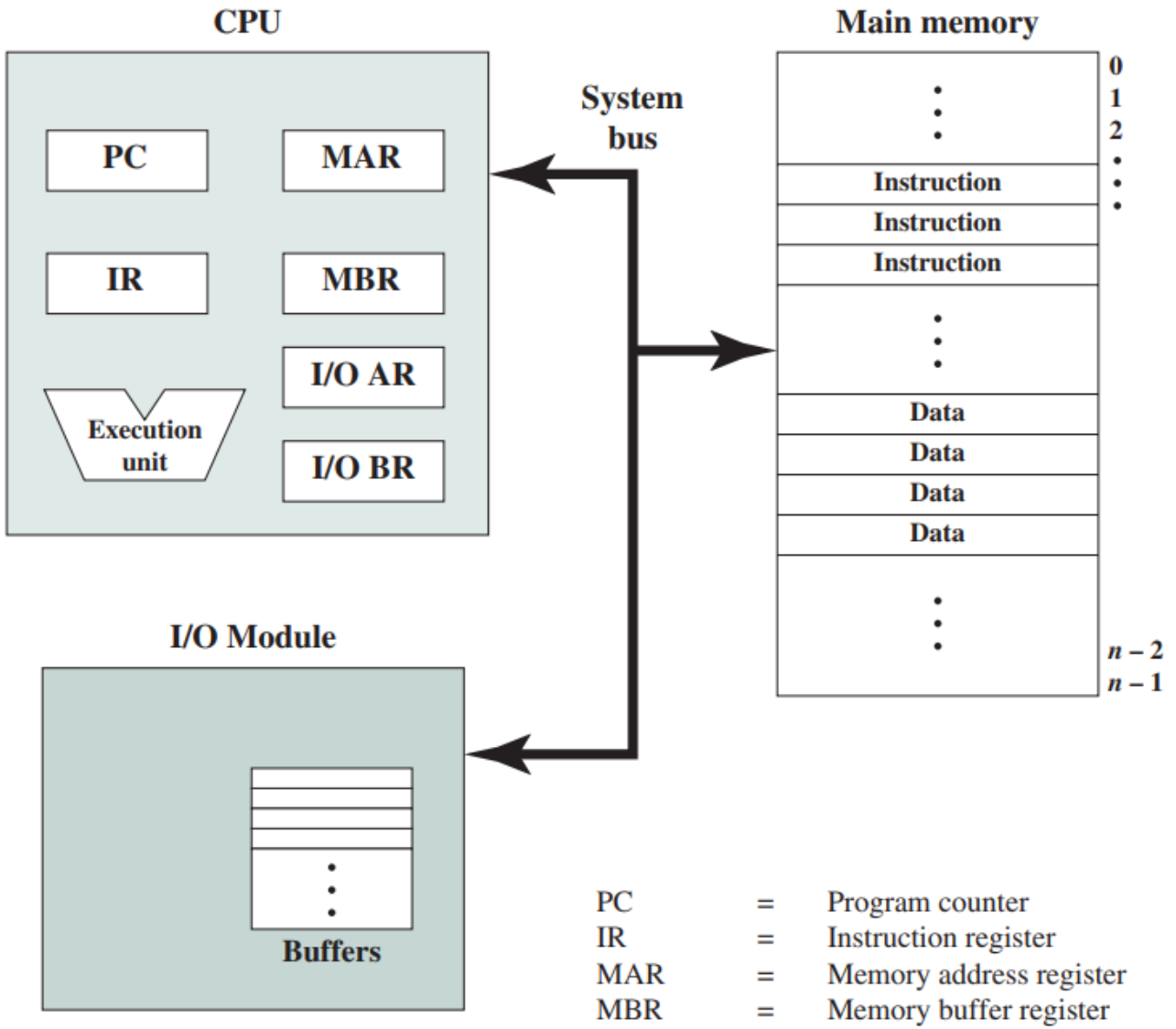
1. Computer Components

- More detailed computer organization



Computer organization with system link

Computer components: top-level view



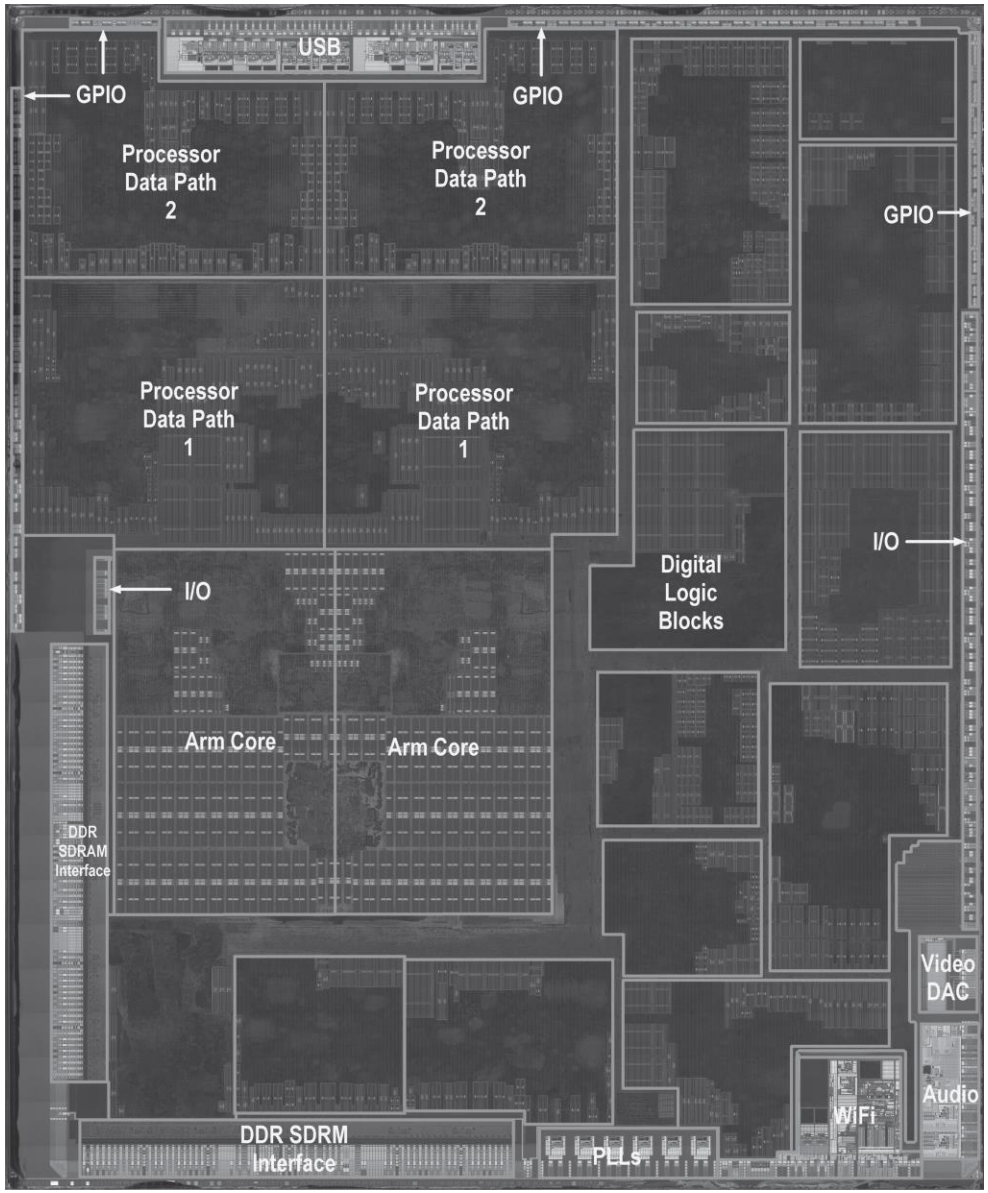
PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

CPU (Central Processing Unit)

- Control Unit
  - Fetch instruction from memory.
  - Interpret instruction.
  - Control other components to execute instruction.
- Datapath: performs arithmetic operations to process data (i.e., Arithmetic and Logic Unit).
- Register file (chapter 3): small and fast data storage for instruction execution.
- Some other dedicated components

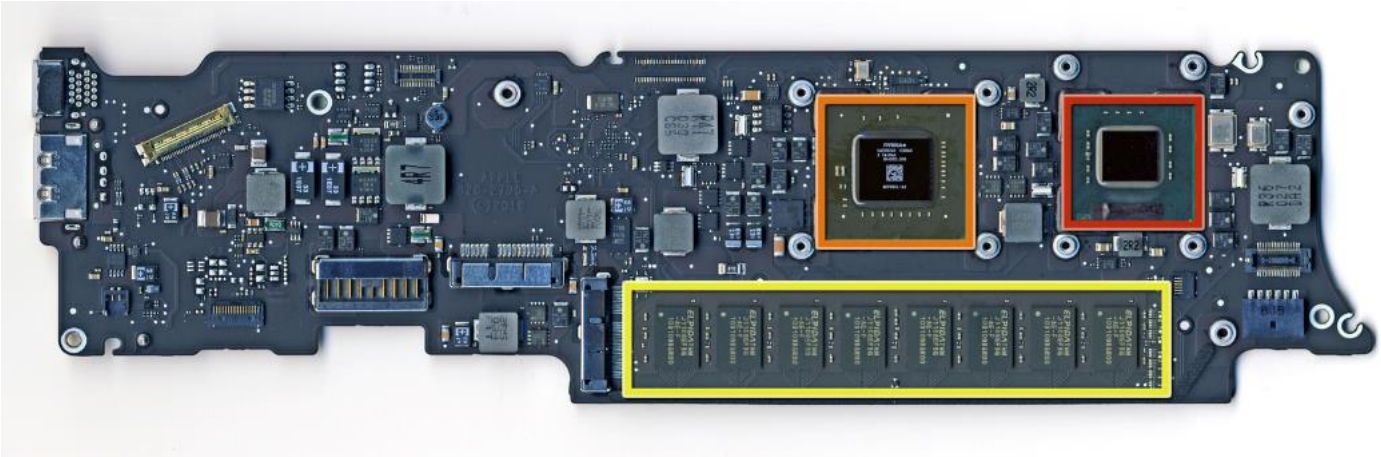
CPU

- Example: Apple A5



Memory

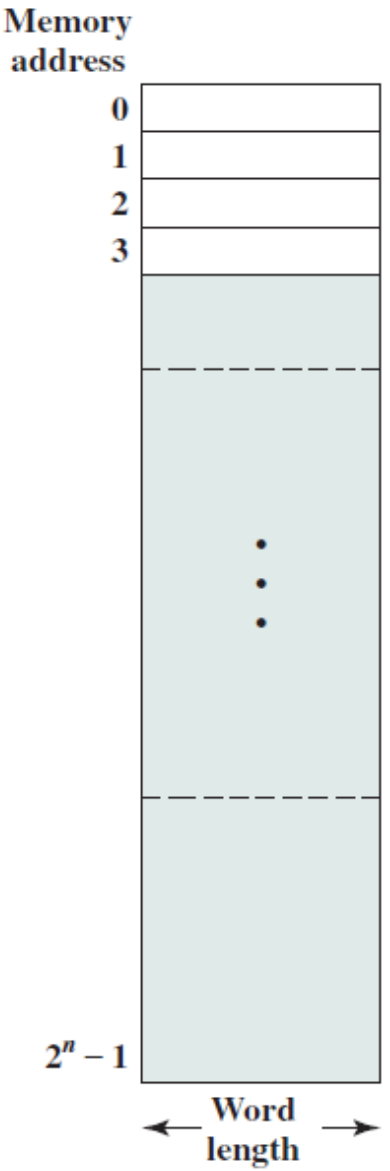
- Store instructions of the running programs.
- Store data that are currently in use.



Further reading: memory technologies

Memory

- Logical organization
  - Array of memory cells
  - Each cell holds one byte of data
  - Each cell is assigned an unique adress
  - Data value can be changed, address is fixed
- Data are stored on memory cells
  - 8-bit integer requires 1 cell
  - 32-bit integer requires 4 cells
  - Array requires consecutive cells according tc its size.
  - …

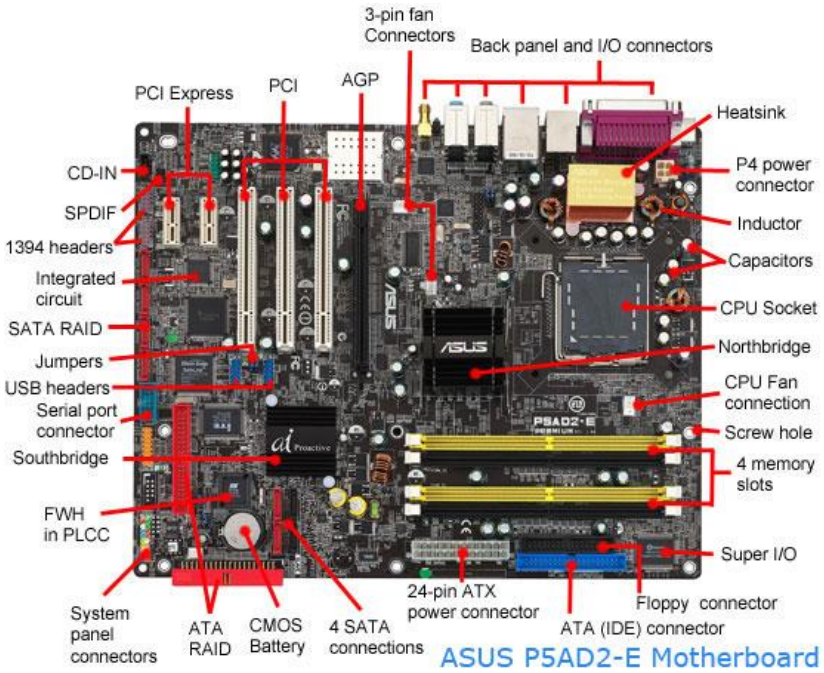


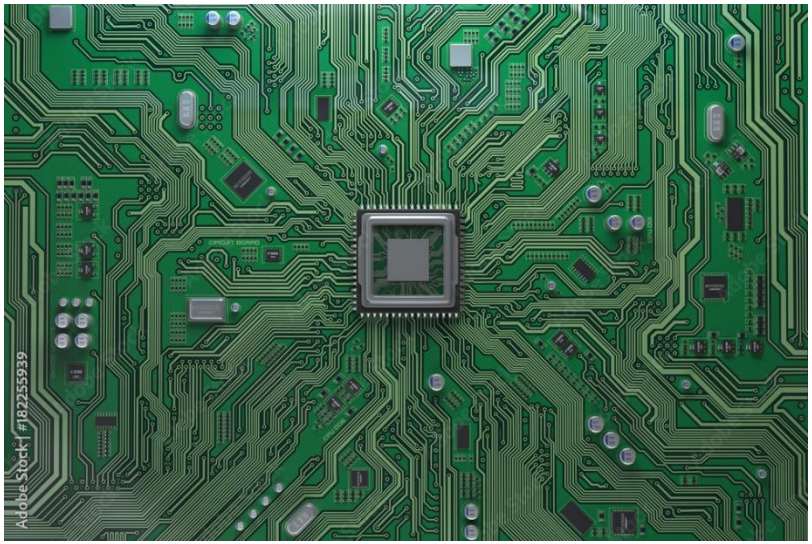
Input/output

- Interfacing computer with physical world/environment.
- Types of I/O device
  - Input: mouse, keyboard, webcam…
  - Output: display, printer, speaker…
  - Storage: HDD, SSD, optical, USB drives…
  - Communication: WiFi, Ethernet, Bluetooth modules…

Link: System interconnection

- The fabric to connect all components
- Huge number of connection, requires very good design so that all components function properly





- A modern computer motherboard (mainboard) typically has 4 to 12 layers.

2. Computer functions

- Executing program
- Interrupt
- Input/Output

2.1 Executing program

- ➔ the most basic function of computers.
- Program: a set of instructions.
- Instruction: a set of binary bits in a predefined format (usually consist of two main parts: Opcode and Operands)
- Computers execute instructions sequentially.
- Instruction cycle: the processing required for a single instruction execution
  - Instruction fetch (fetch cycle): control unit fetches an instruction from memory
  - Instruction execution (execute cycle):
    - control unit decodes instruction,
    - then “tells” datapath and other components to perform the required action.
    - More details in Chapter 5.

Instruction fetch

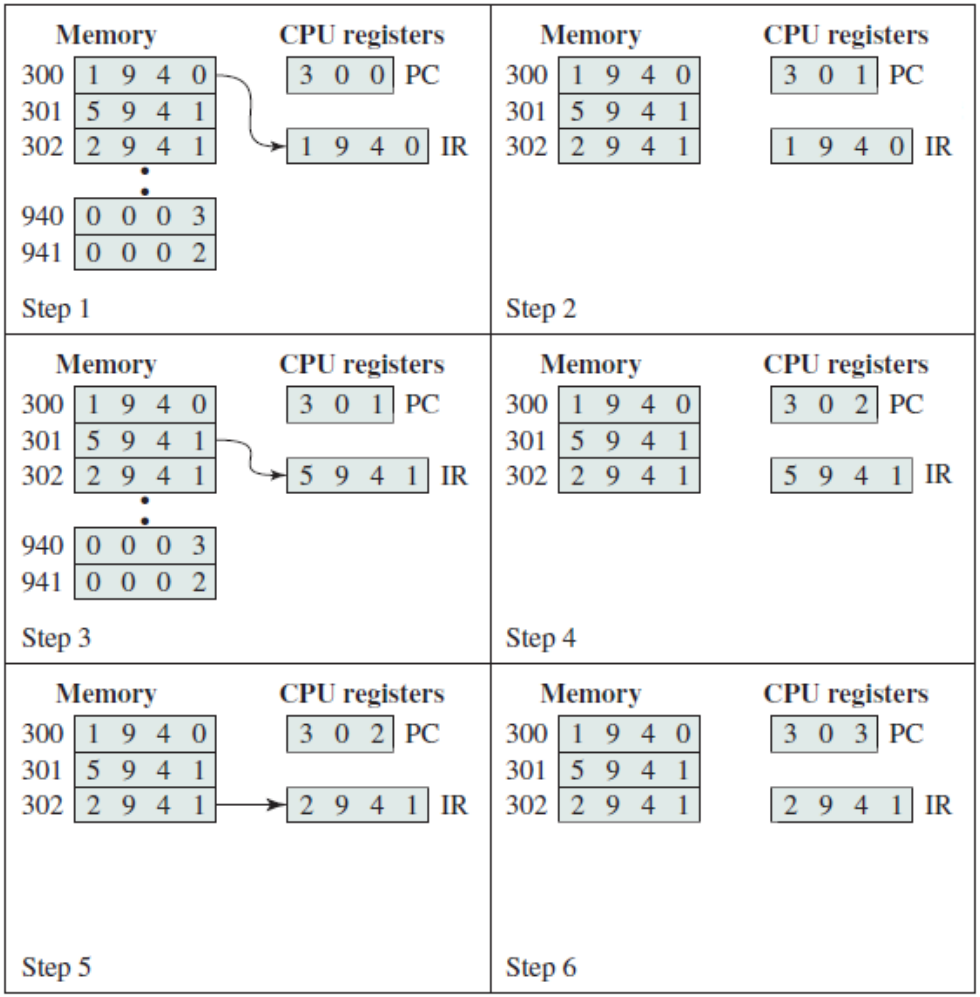
- Questions:
  - How does the CPU know which instruction to fetch next?
  - Where is the location of the fetched instruction inside CPU?

Instruction fetch

- Importance
  - To get the correct instruction.
  - To execute all instructions in a program sequentially.
- At the beginning of each instruction cycle the processor fetches an instruction from memory.
- The program counter (PC) holds the address of the instruction to be fetched.
- The processor increases PC after each instruction fetch so that PC points to the next instruction in sequence.
- The fetched instruction is loaded into the instruction register (IR).

Instruction fetch

- Automatic increment of PC in fetch cycle

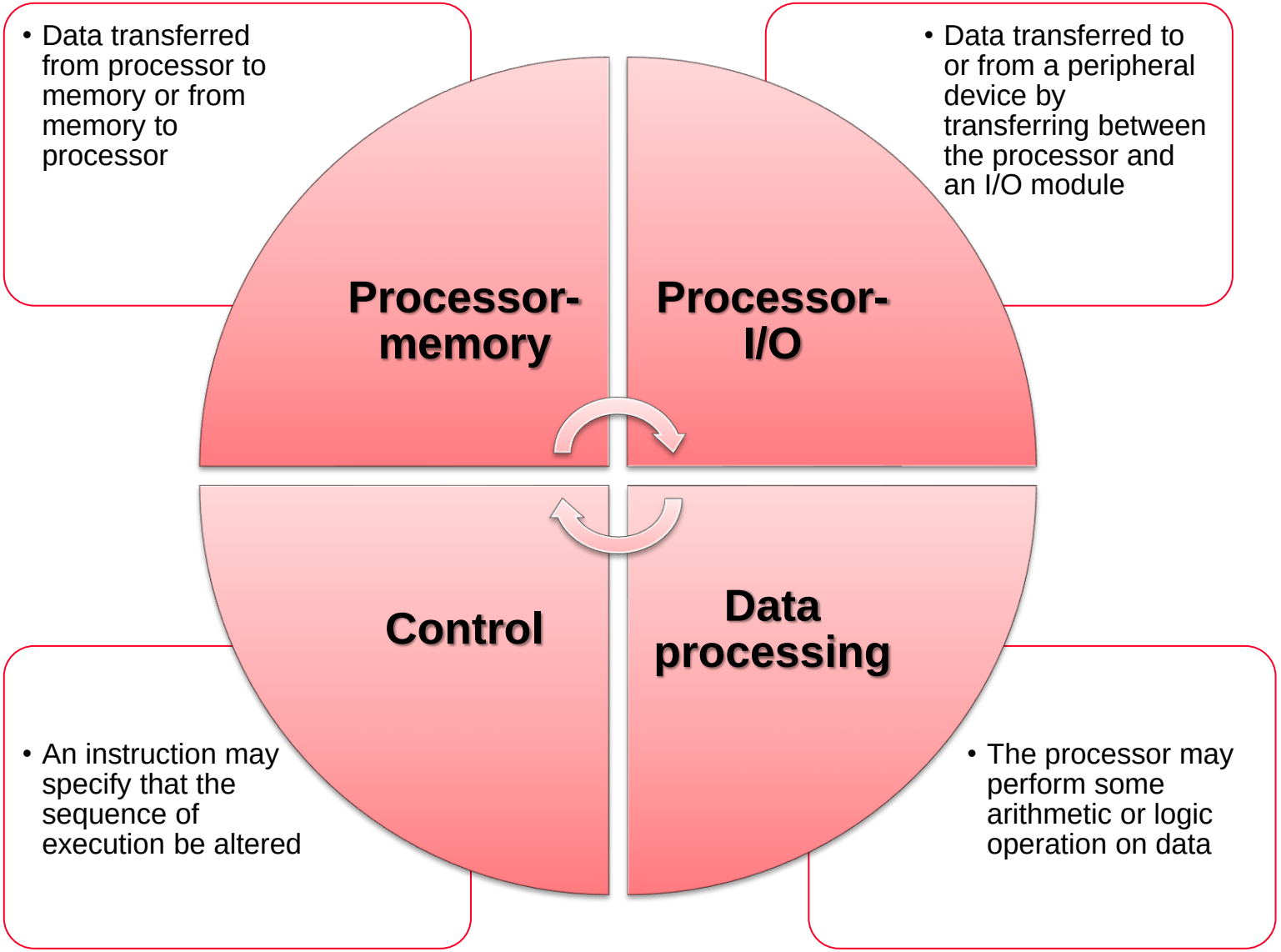


Elaboration: How to support branching?

Instruction execution

- Instruction (fetched and stored in IR) is decoded to get
  - The operation that the processor needs to do
  - The location to get input data (source operands)
  - The location to store output data (destination operand)
- Operand address calculation: calculate the address of operands
- Operand fetch: fetch source operands
- Data operation: perform the action on source operands and get result
- Operand store: store result into destination operand

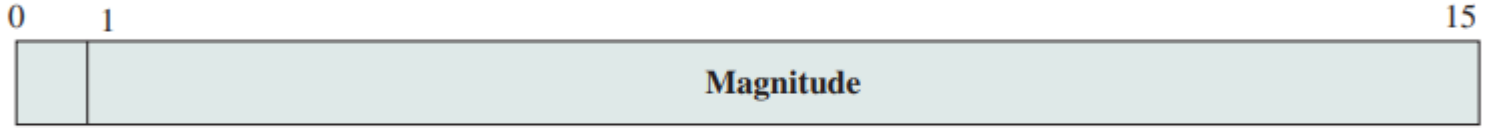
Types of operation



Example

- Consider following hypothetical computer
  - The computer contains a single register named AC (16 bits)
  - Instruction format: 16 bits, 4 bits for the Opcode (representing 16 different opcodes/instructions)

(a) Instruction format



(b) Integer format

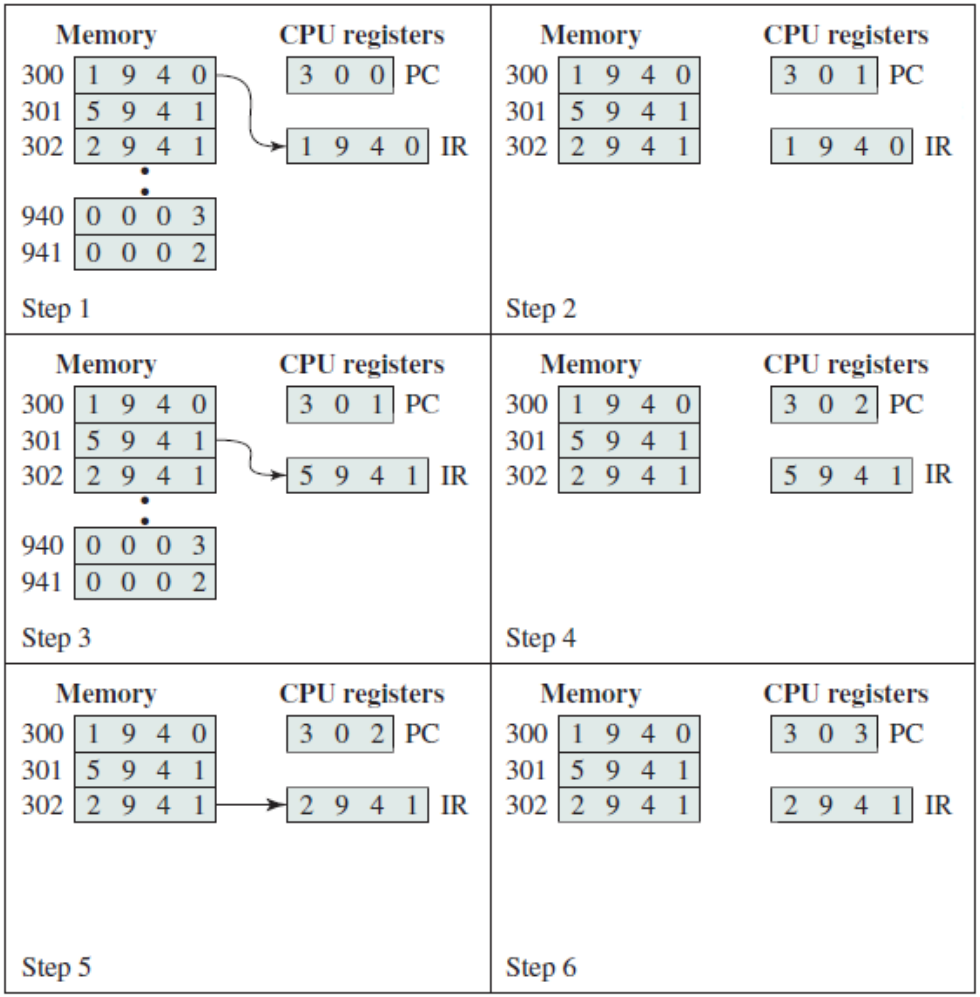
Program counter (PC) = Address of instruction
Instruction register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

0001 = Load AC from memory
0010 = Store AC to memory
0101 = Add to AC from memory

Example

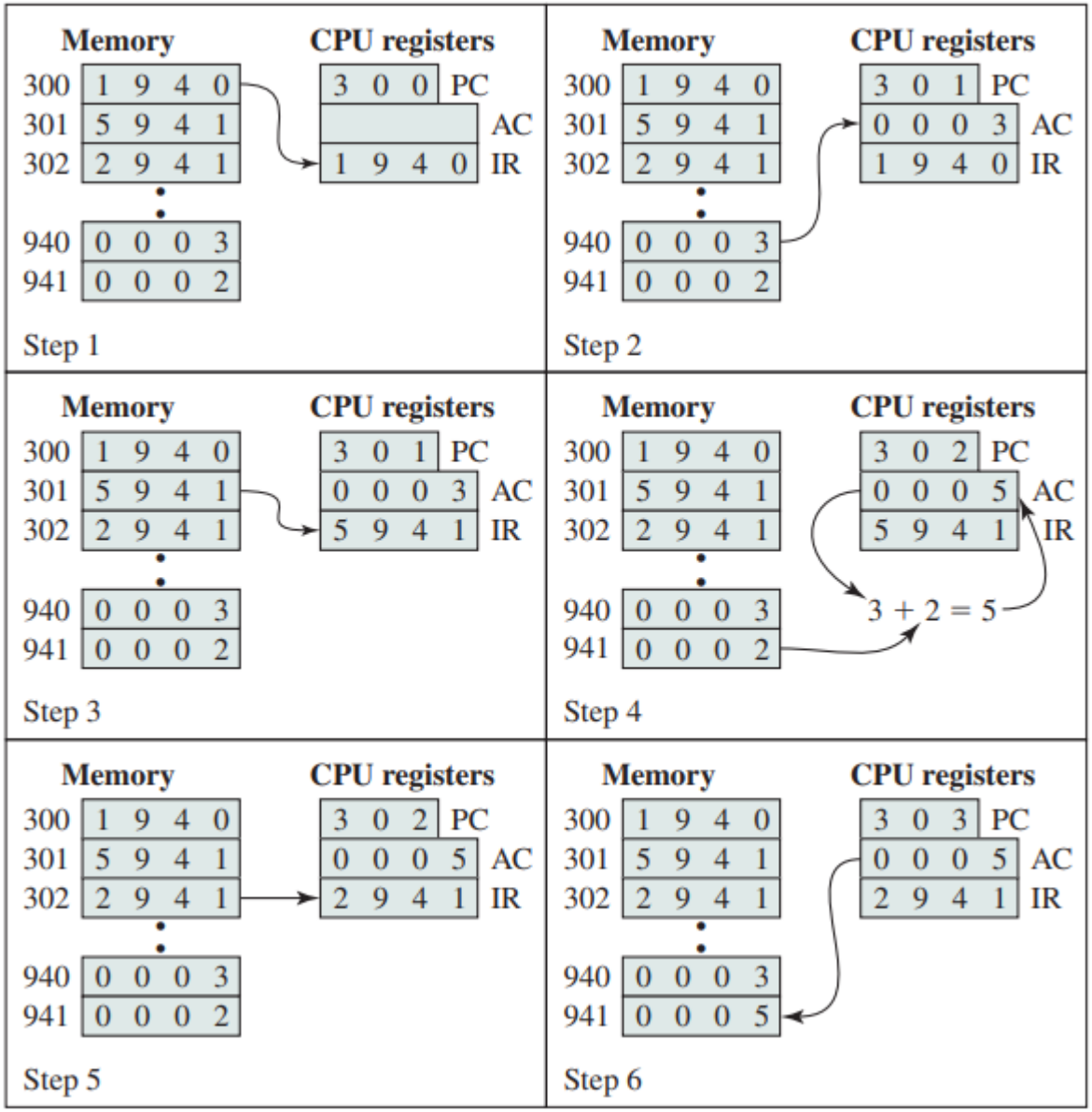
- What is the value of AC register in step 2, 3, 4, 5, 6?



Elaboration: How to support branching?

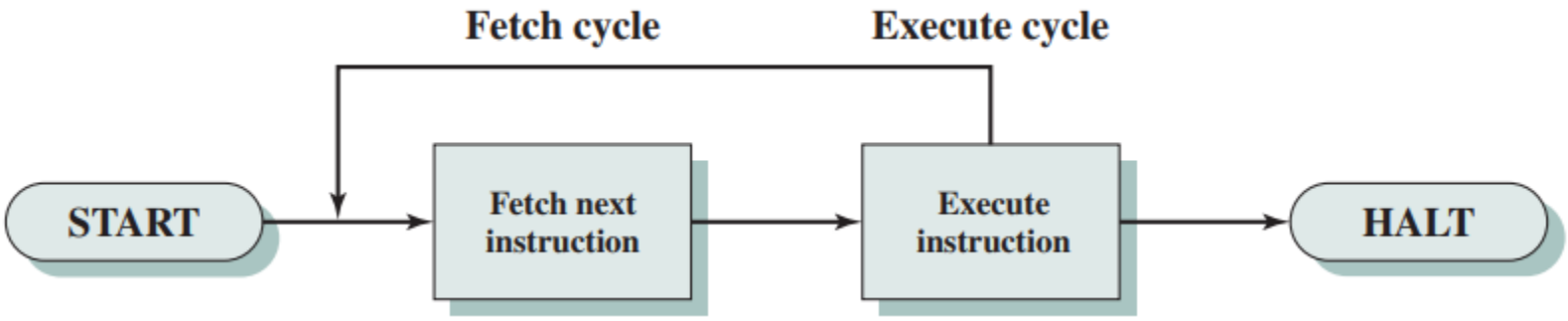
Example

- Solution



2.2 Interrupt

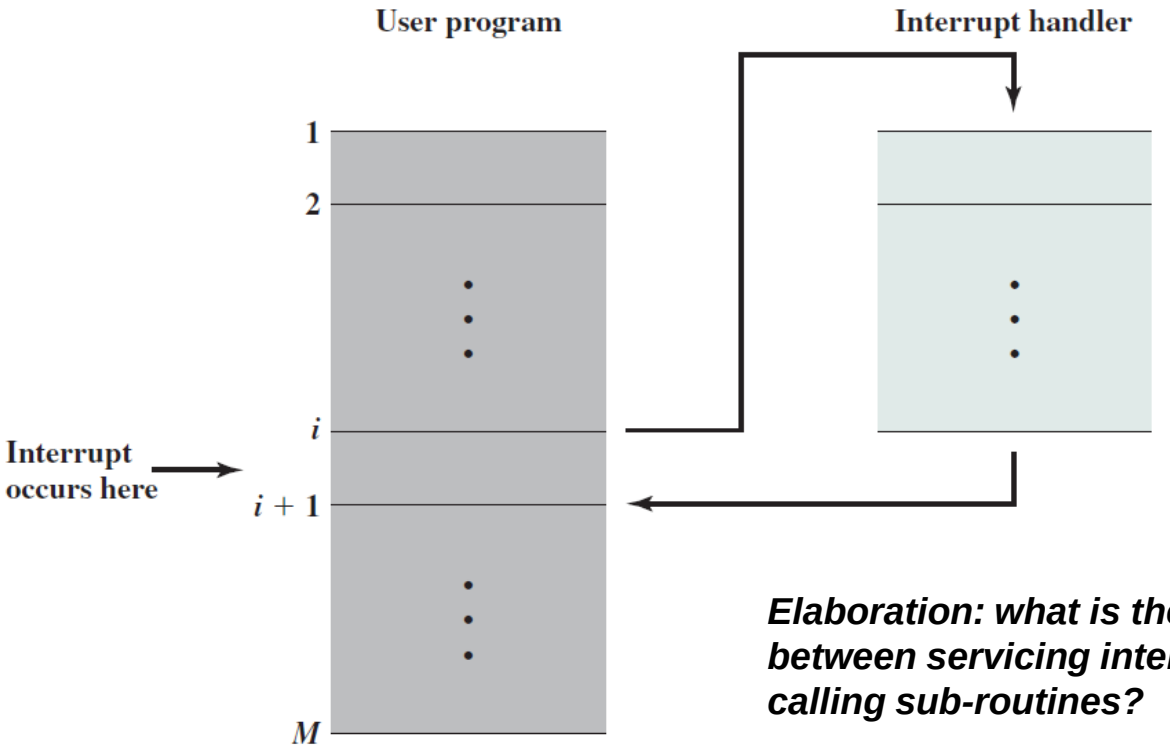
- What if CPU operates without interrupt?



- Fact: most external devices are much slower than CPU
- Suppose that CPU is transferring data with a printer using above instruction cycle scheme
  - After each write operator, CPU must wait until the printer catches up

Interrupt

- The mechanism to allow other components (memory, I/O) interrupt the normal processing of processor.
- Servicing interrupt: processor temporarily switch from the current program to execute a different (rather short) program, before continuing the original program.



Elaboration: what is the difference between servicing interrupt and calling sub-routines?

Sources of interrupt

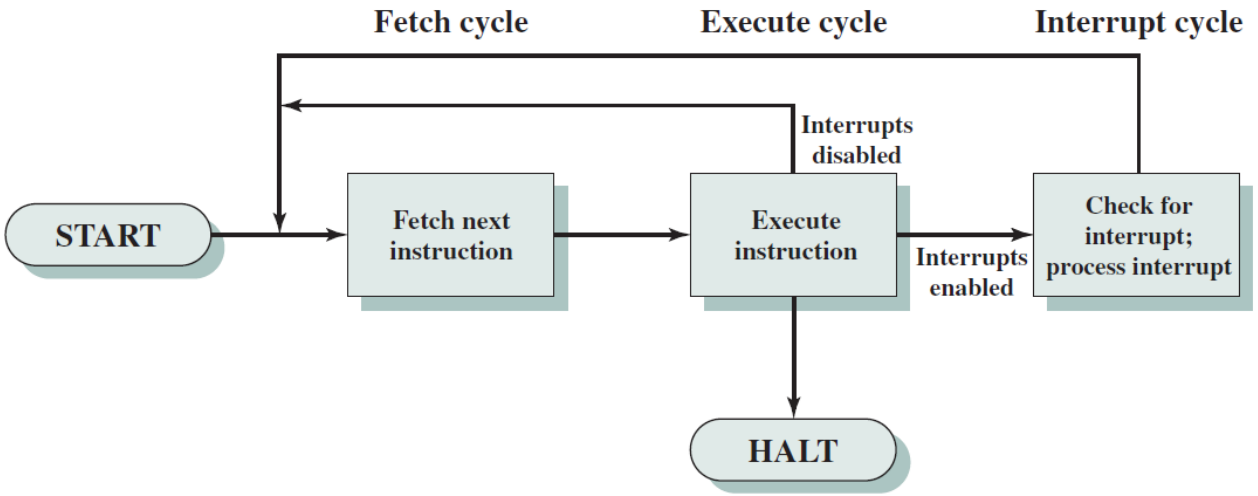
- Typical sources of interrupt:
  - Software/program: occurs during instruction execution upon some special condition such as division by 0, arithmetic overflow… Can also be called exception.
  - Timer: generated by system timer inside processor, to provide timing service, such as for operating system task scheduler service.
  - I/O: generated by I/O modules, to request service from processor or acknowledge the completion of an operation.
  - Hardware failure: generated when error happens with hardware.
- Example: detecting keyboard events (key up/key down)
  - Method 1: CPU checks keyboard status frequently
  - Method 2: keyboard issue interrupt to notify CPU upon key up/down
  - Which is better regarding CPU usage?

Interrupt handler/Interrupt service routine (ISR)

- Special programs to be executed to service interrupts.
- Usually a part of operating system or system software.
- Typical operation:
  - Determine the nature of interrupt: source and reason of interrupt.
  - Perform corresponding operation.
  - Return control to the interrupted program.
- Structure:
  - Interrupt handler ends with a special instruction, to restore context and value of PC, so that CPU can continue the interrupted program properly.

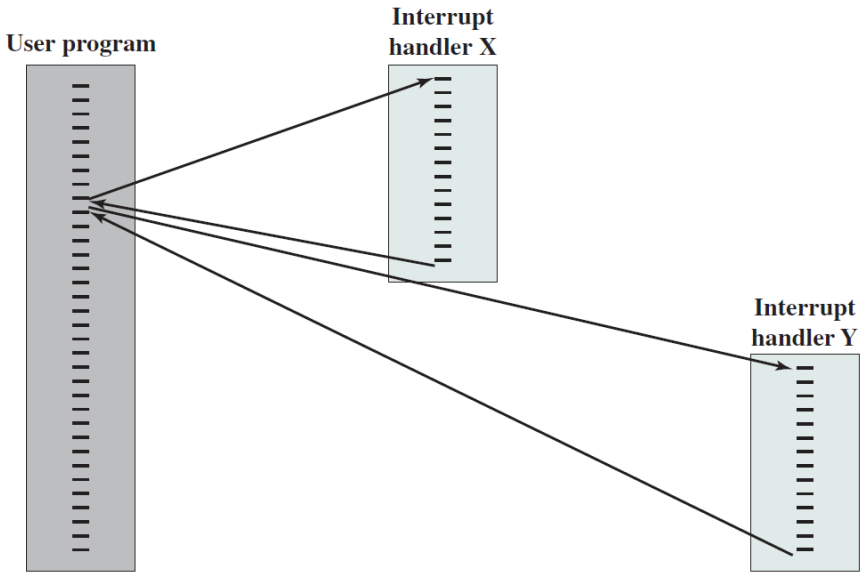
Servicing interrupts: instruction and interrupt cycle

- Interrupt is checked at the end of each instruction cycle
- Interrupt cycle if interrupt occurred:
  - CPU saves context of current program (current value of PC).
  - Address of interrupt handler is loaded to PC.
  - CPU continues with new instruction cycles, with new PC. Interrupt handler will be executed instead of original program.
  - At the end of interrupt handler, context will be restored including PC value. CPU return to the interrupted program.

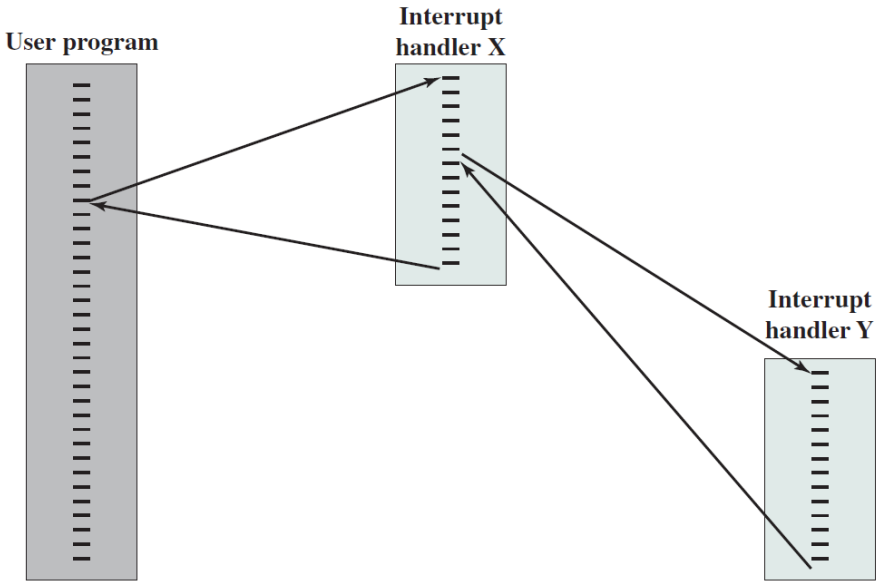


Multiple interrupt procesing

- Number of interrupt sources is (always) high.
- Interrupts can occur at the same time or overlap.
- Sequential vs nested interrupt processing
  - Usually priority-based.
  - More details in chapter 7.



(a) Sequential interrupt processing



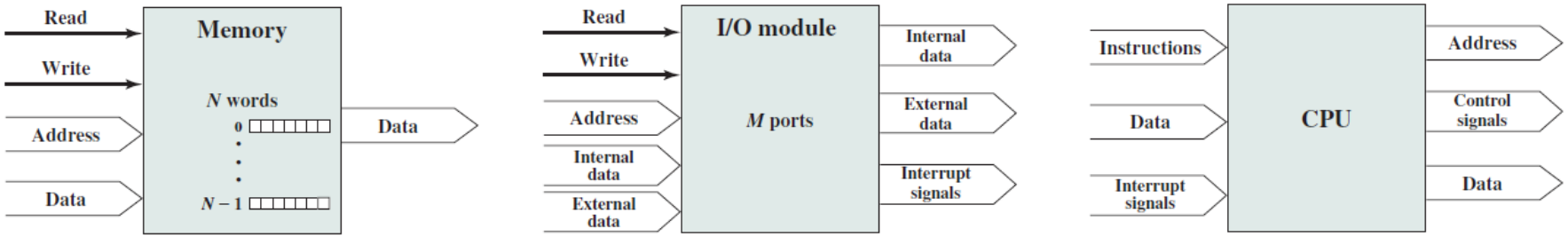
(b) Nested interrupt processing

2.3 Input/output

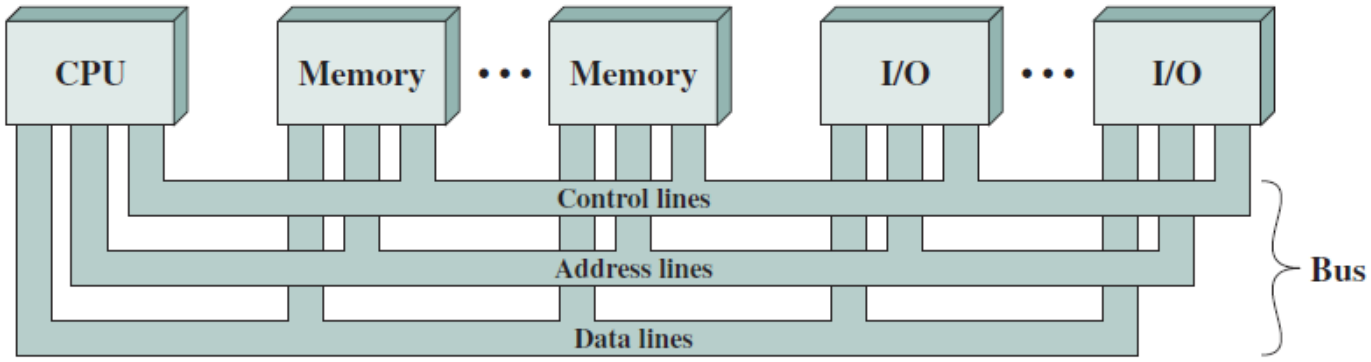
- The operation when data is transferred between I/O modules and CPU/memory.
- CPU-controlled data transfer: data is transferred between CPU and I/O, under the control of CPU.
- Direct memory access: data is transferred between memory and I/O, under the control of special controllers called DMAC.

3. System interconnection

- Interconnection model of each component
  - ➔ Need to connect these components

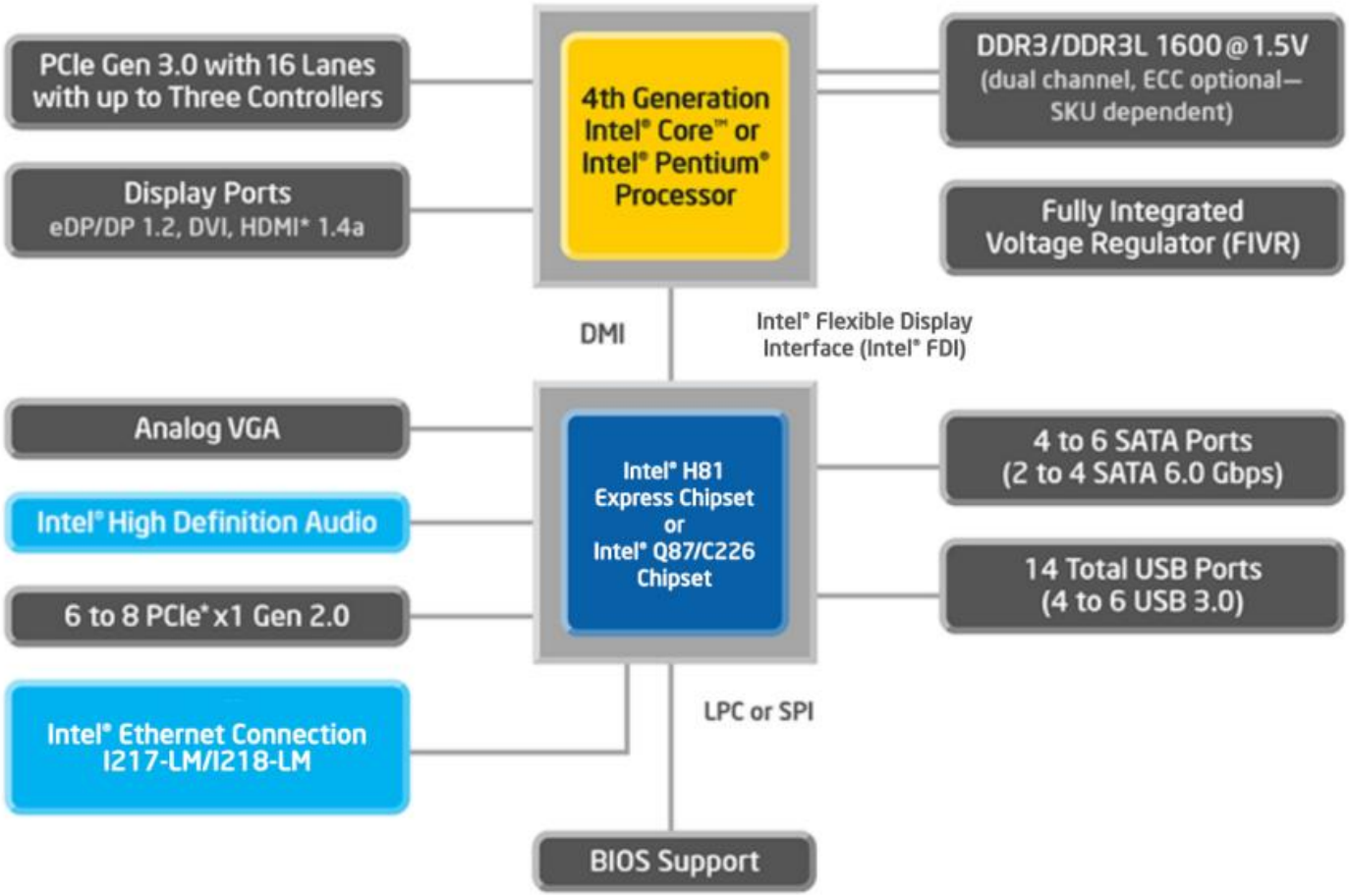


- Theoretical bus interconnection scheme



System interconnection

- Interconnection system for high performance computers: **hierarchical bus**



End of chapter 2