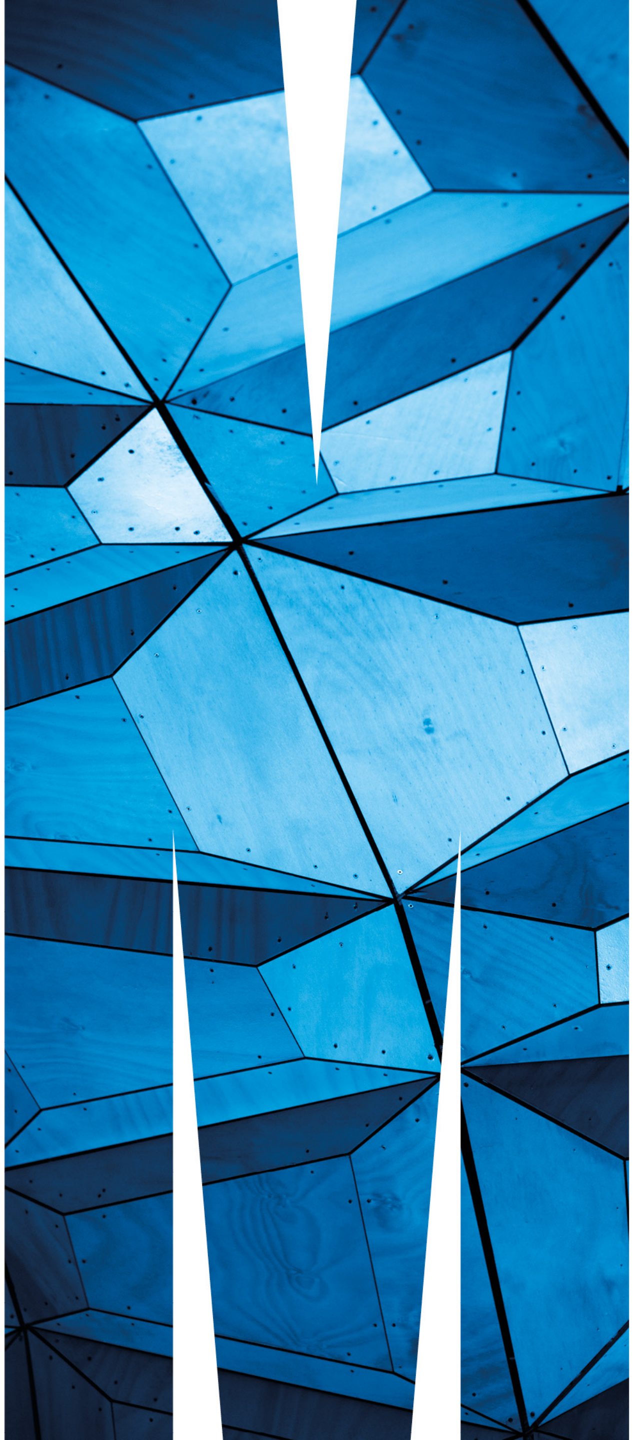
MONASH
INFORMATION
TECHNOLOGY

FIT2100 Semester 2 2017 Lecture 1: Computer System Overview

Jojo Wong



Lecture 1: Learning Outcomes

- Upon the completion of this lecture, you should be able to:
  - Describe the building blocks of a computer system
  - Understand various terminologies with respect to resource allocation for the execution of programs
  - Explain the steps taken by a processor to execute a program instruction
  - Understand the concept of interrupts and how and why a processor uses interrupts

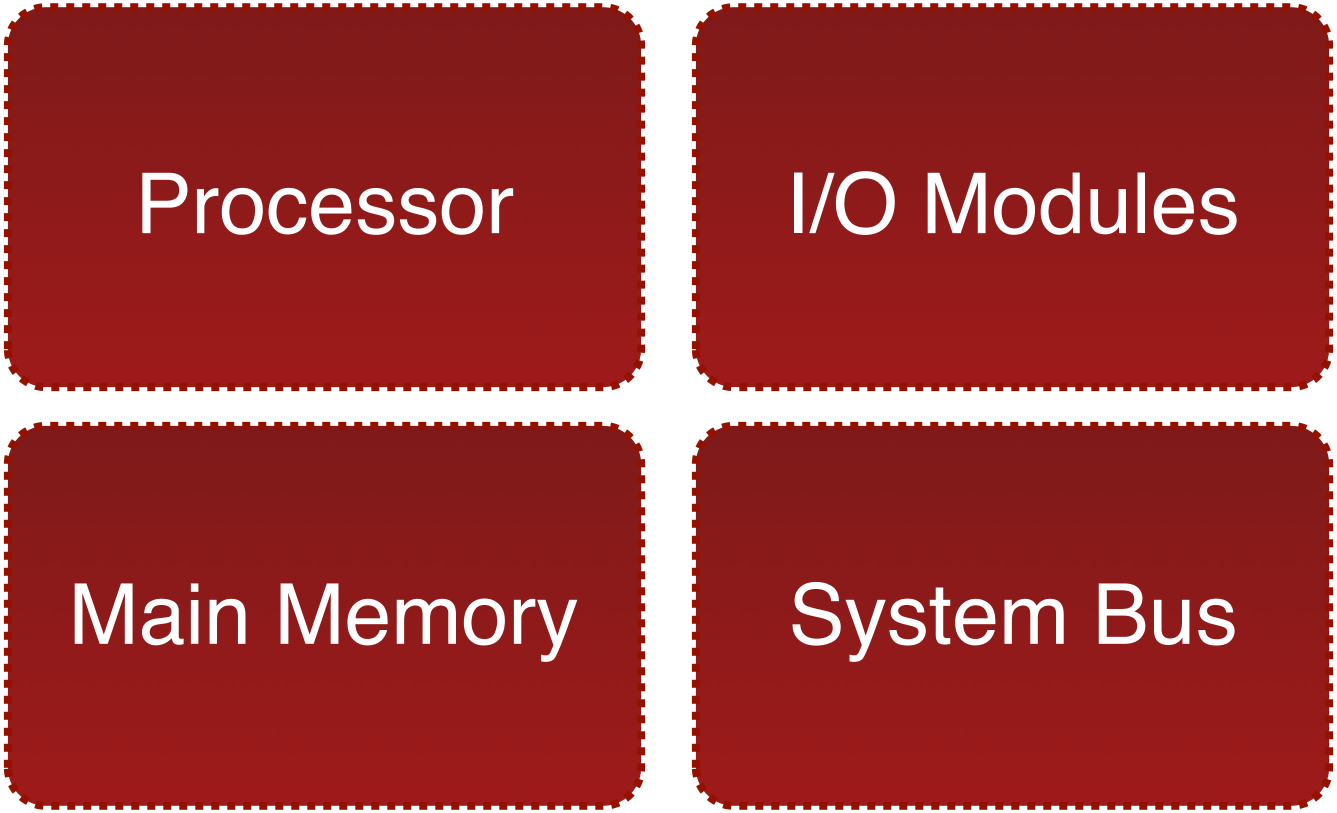
What is an operating system?

Operating System

- Exploits the hardware resources of one or more processors
- Provides a set of services to system users
- Manages secondary memory and I/O (input/output) devices on behalf of its users

What are the basic elements of a computer system?

Computer System: Basic Elements



Processor

I/O Modules

Main Memory

System Bus

Processor

Controls the operation of the computer

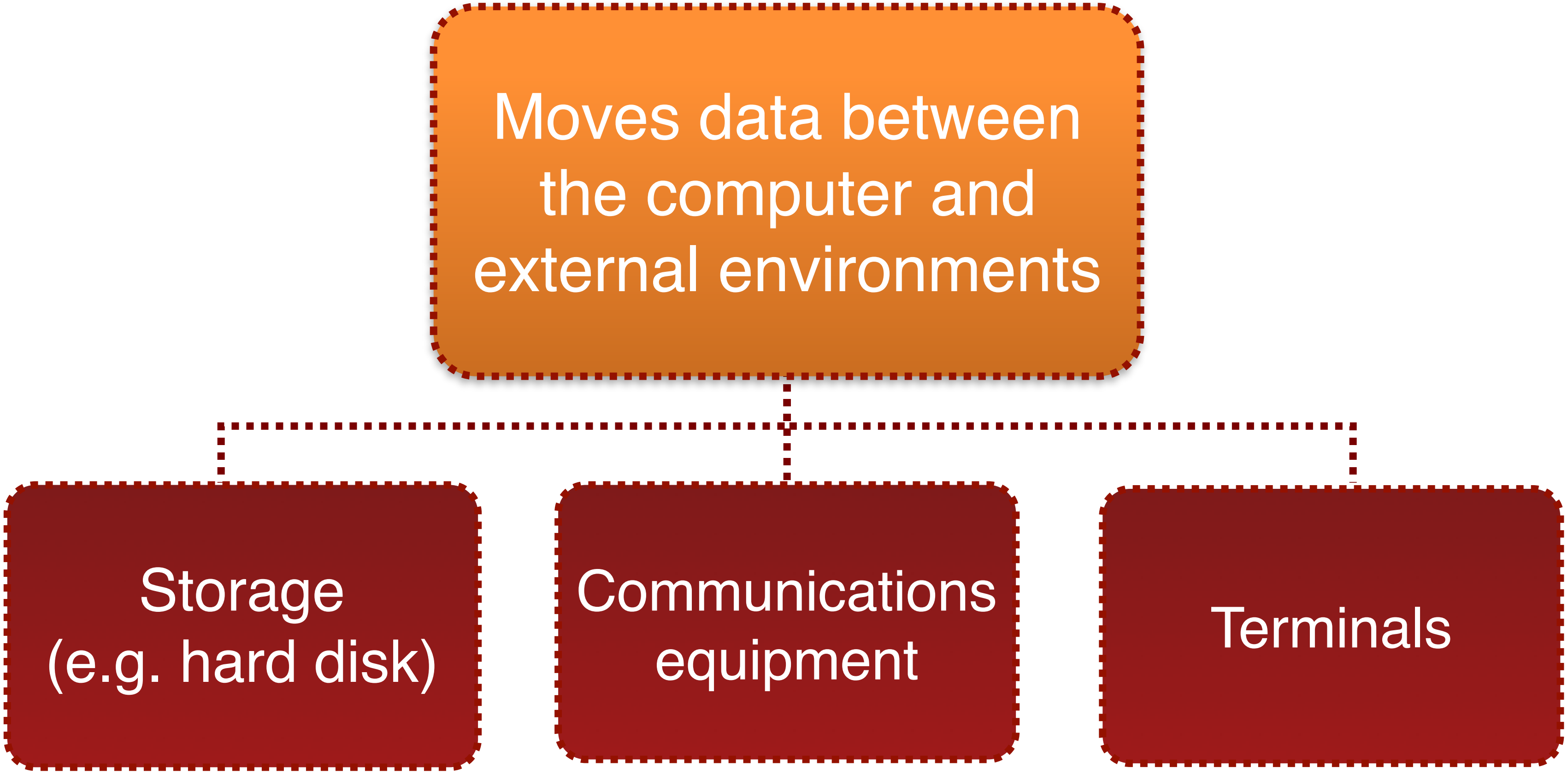
Performs the data processing functions

Central Processing Unit (CPU)

Main Memory

- Stores data and programs
- Volatile
  - Contents of the memory is lost when the computer is shut down
- A.k.a. real memory or primary memory

I/O Modules



Moves data between the computer and external environments

Storage (e.g. hard disk)

Communications equipment

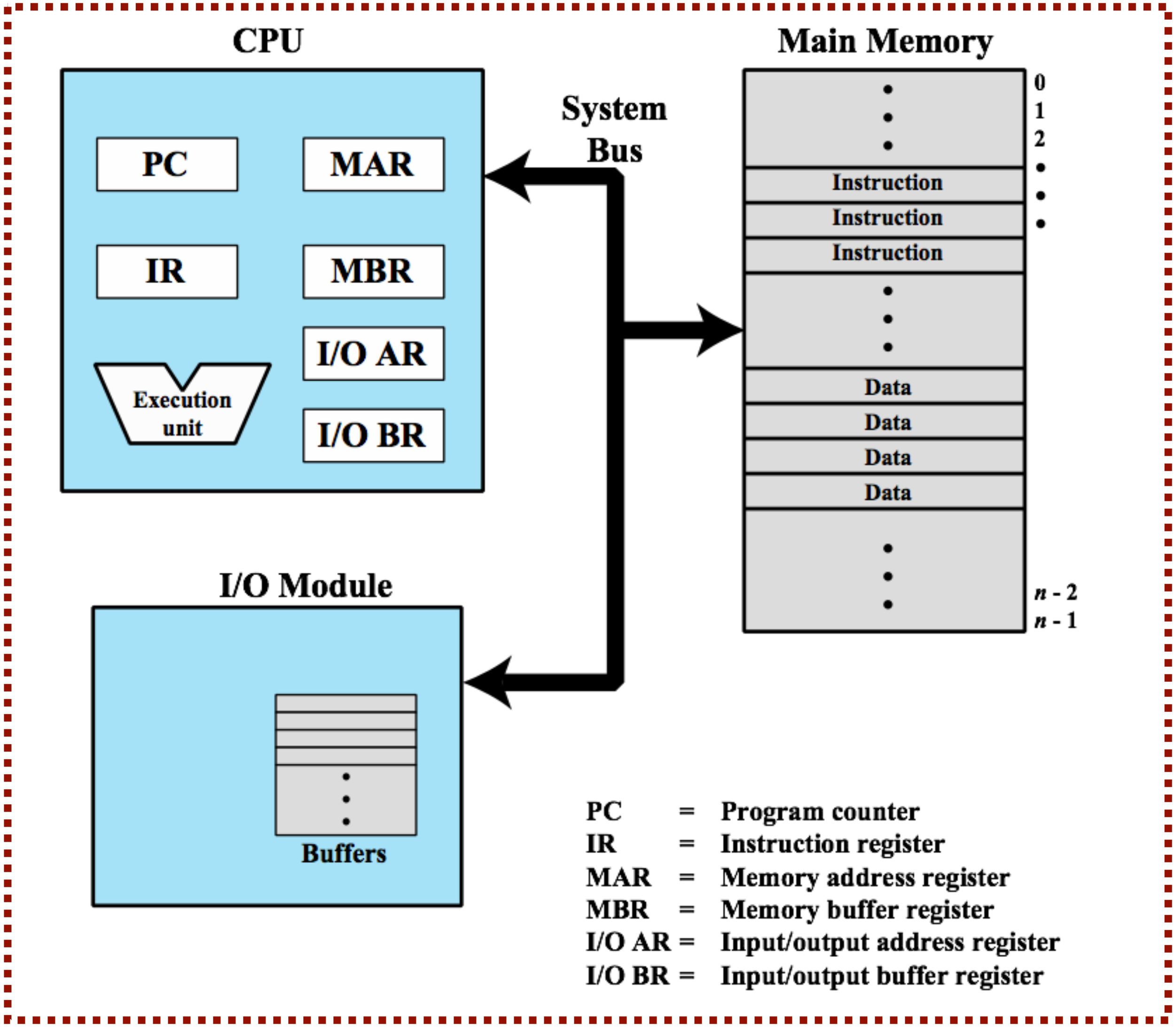
Terminals

System Bus

- Provides for communication among processors, main memory, and I/O modules

- Comprises of:
  - Address bus
  - Data bus
  - Control bus

Computer Components: Top-level View

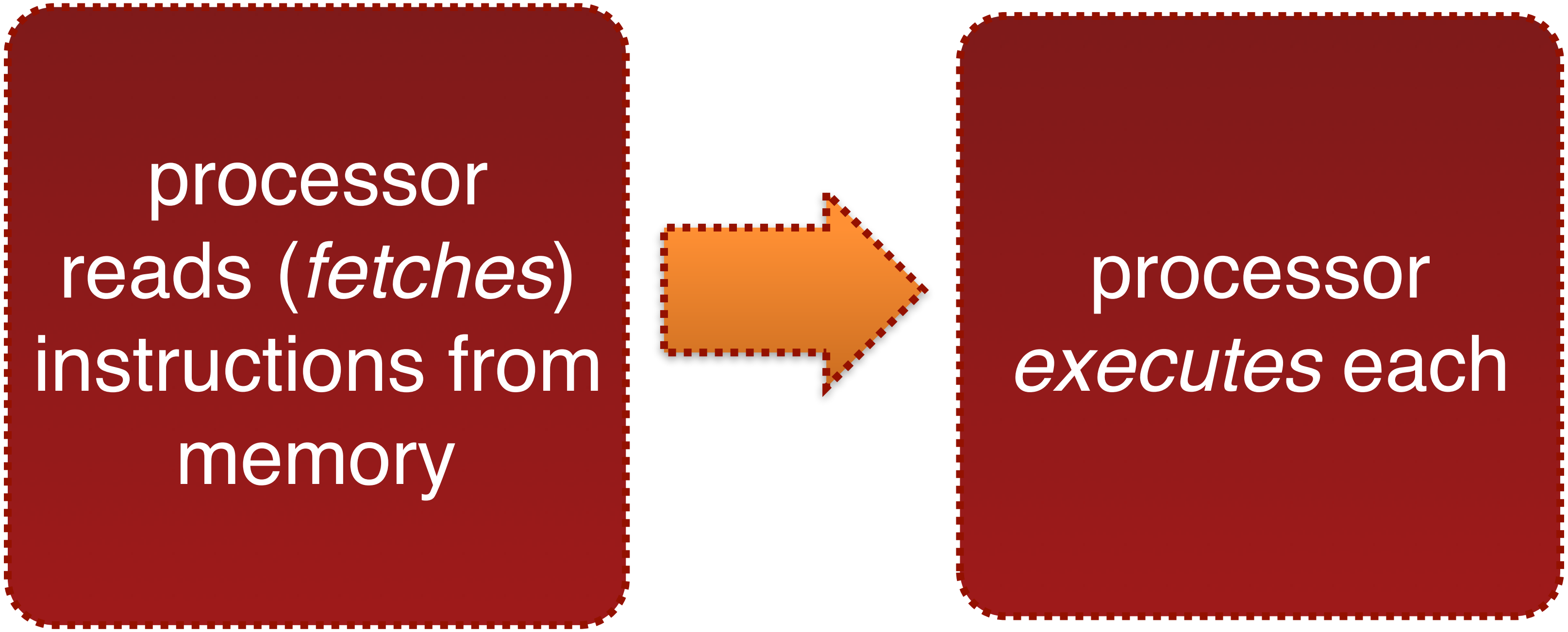


PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

How does a program get executed by a processor?

Instruction Execution

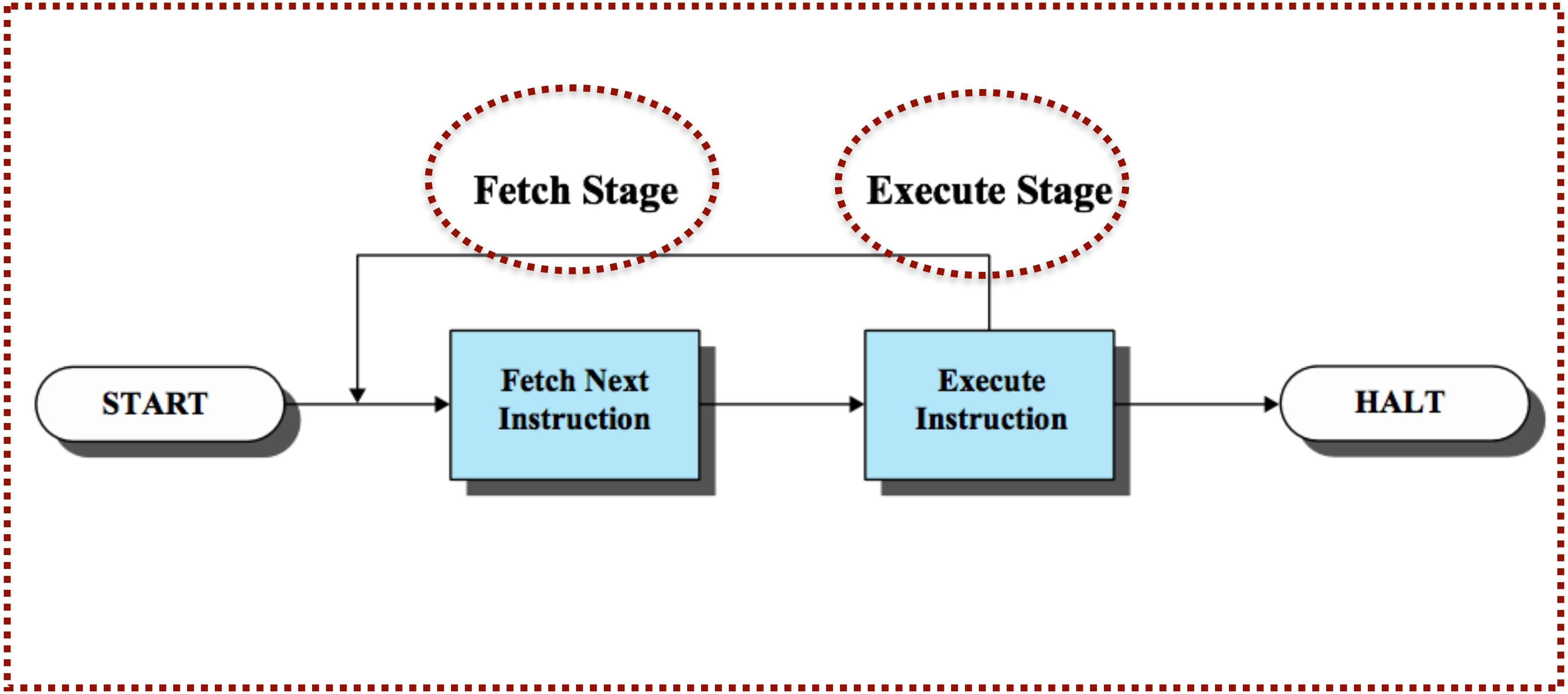
- A program consists of a set of instructions stored in memory



processor reads (*fetches*) instructions from memory

processor *executes* each

Basic Instruction Cycle



Fetch Instructions

- The processor fetches the instruction from memory
- Program Counter (PC) holds address of the instruction to be fetched next
- PC is incremented after each fetch

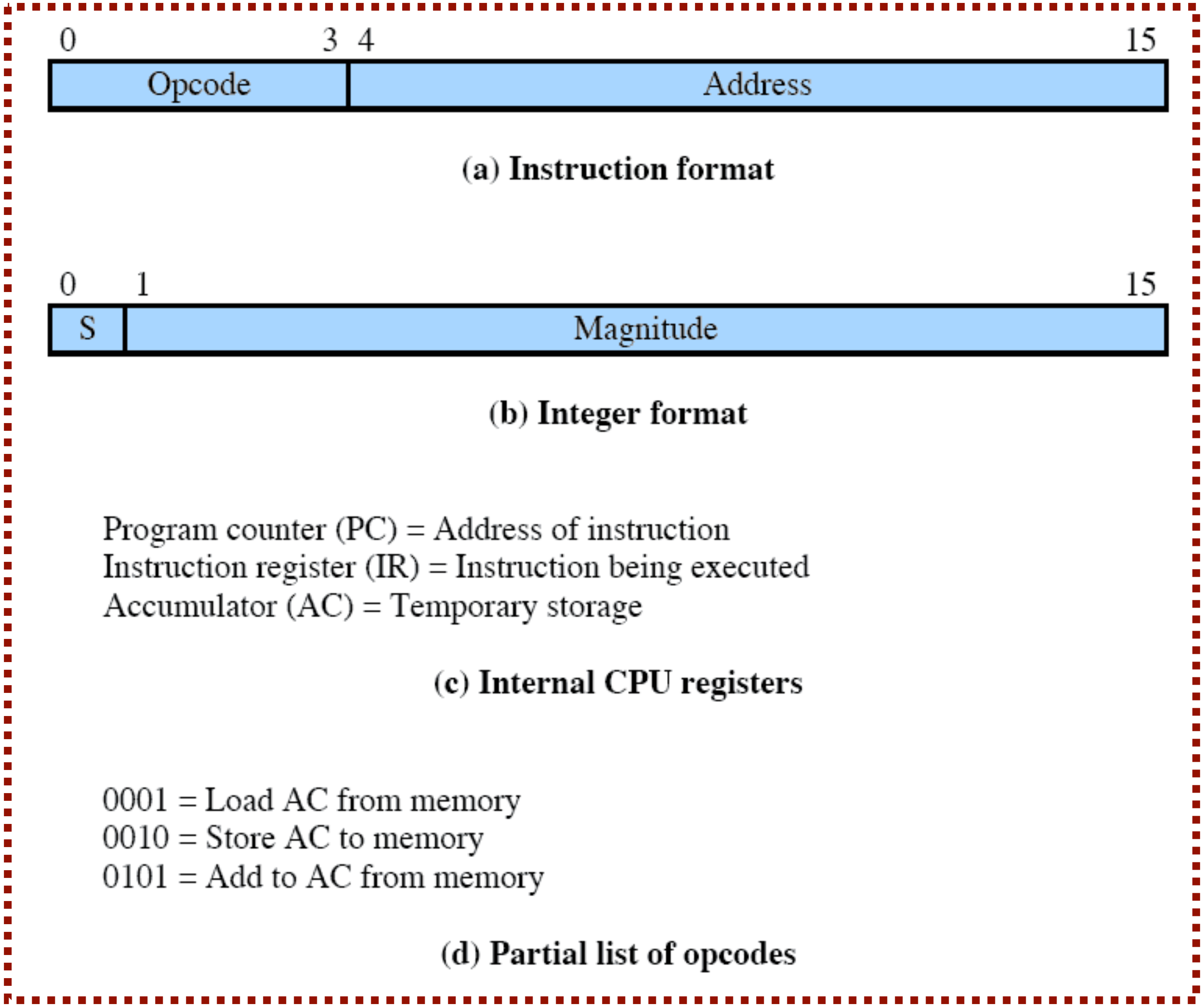
Execute Instructions

- Fetched instructions are loaded into Instruction Register (IR)

- The processor interprets the instruction and performs the required action.

- Four categories of actions:
  - Processor-memory
  - Processor-I/O
  - Data processing
  - Control

Example: A Hypothetical Processor



(a) Instruction format

(b) Integer format

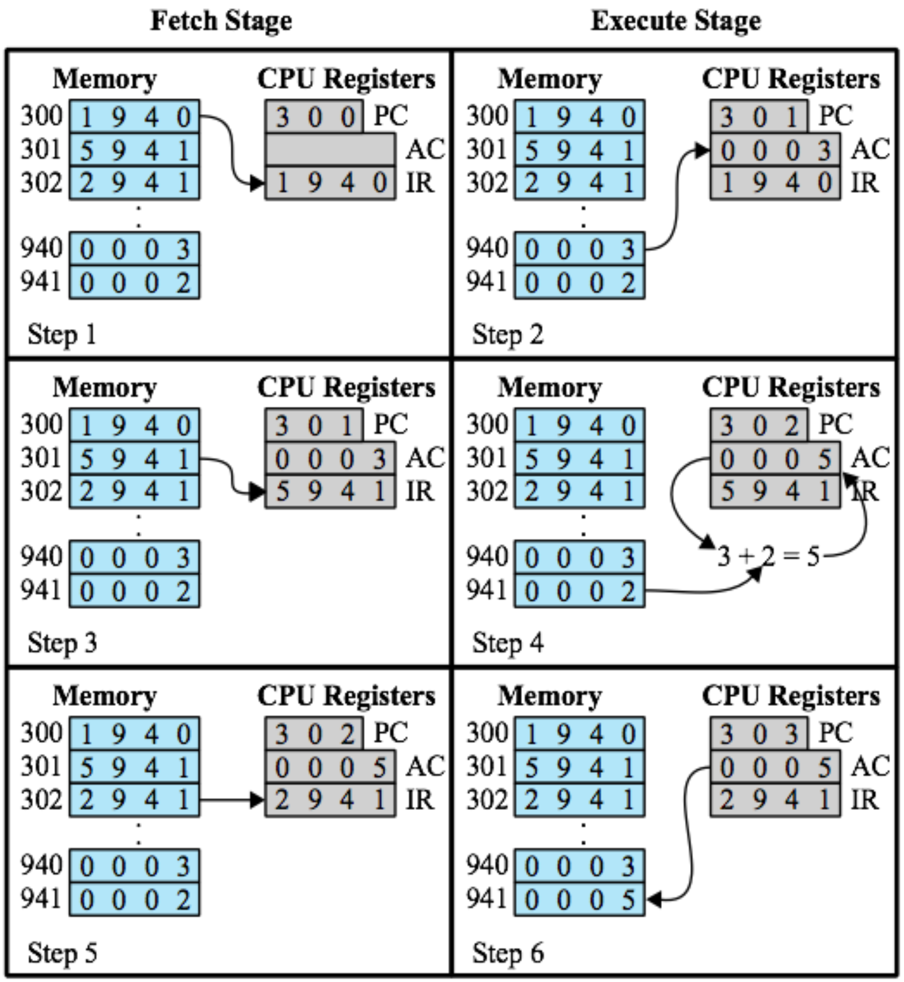
Program counter (PC) = Address of instruction
Instruction register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

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

(d) Partial list of opcodes

Example: Program Execution



What are interrupts?
Why a processor uses interrupts?

Interrupts

- Interrupt the normal sequencing of the processor
- Provided to improve processor utilisation
- Most I/O devices are slower than the processor
  - Processor must pause to wait for the device to complete the operation
  - Wasteful use of the processor

Classes of Interrupts

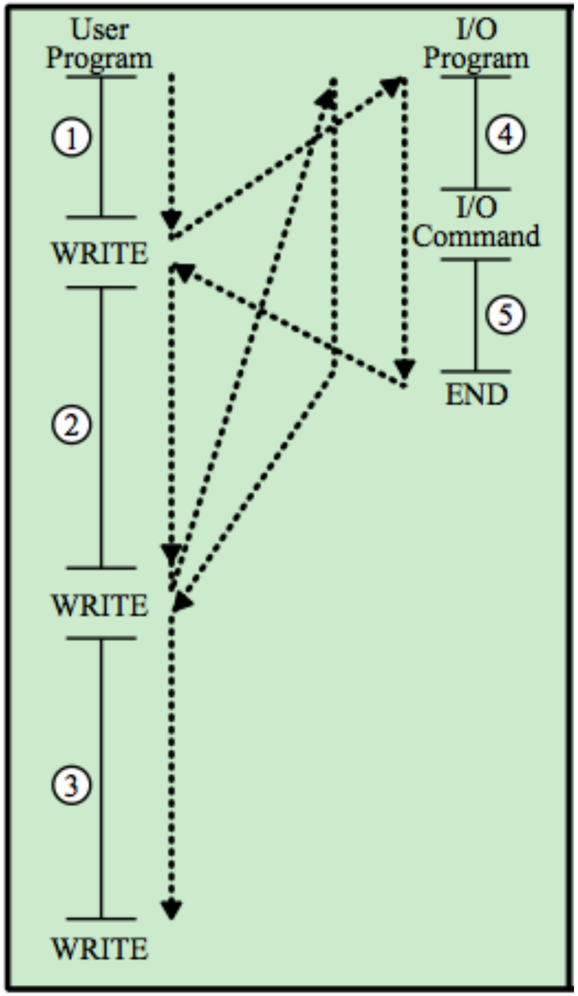
Program Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.

Timer Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.

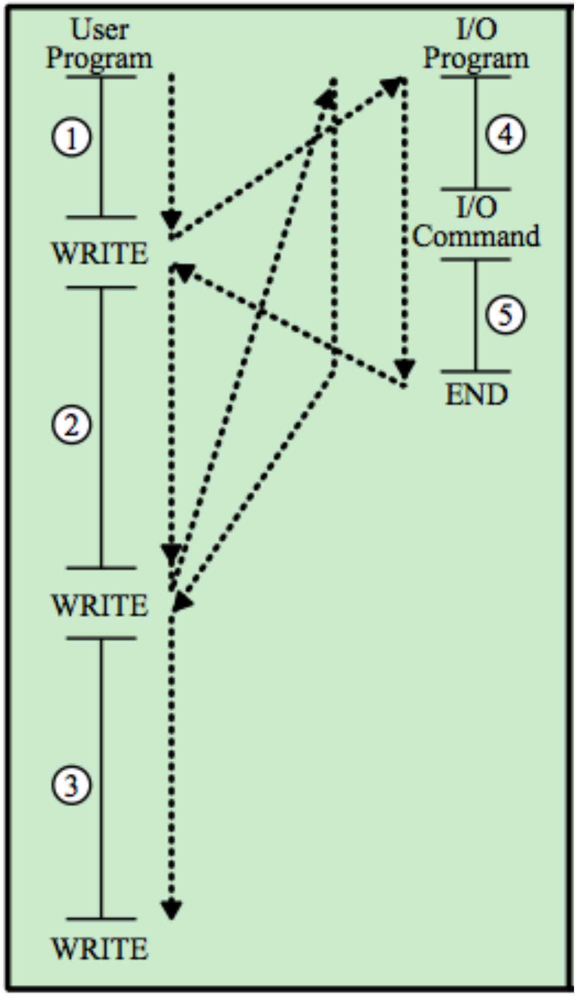
I/O Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.

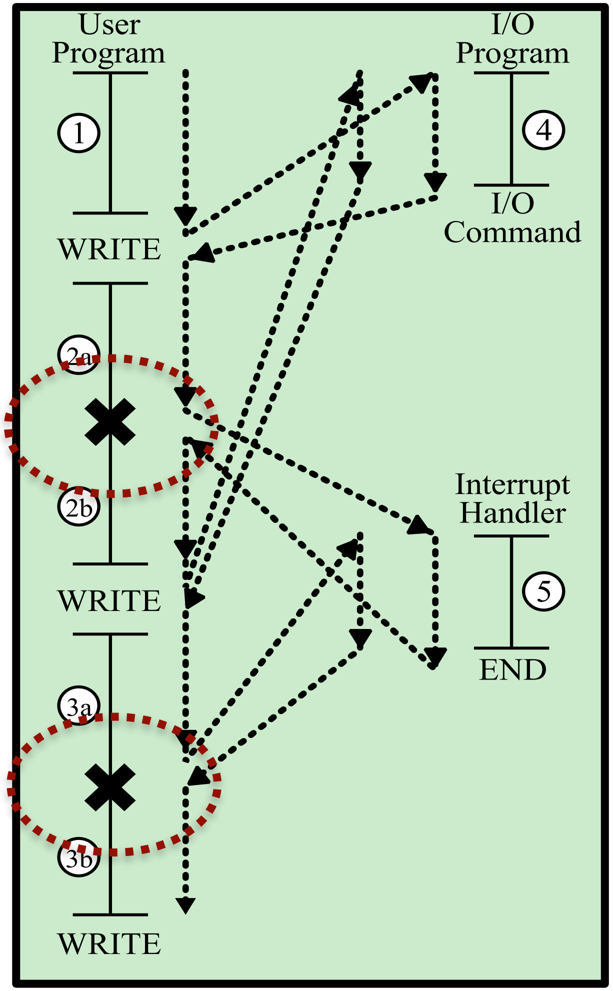
Hardware failure Generated by a failure, such as power failure or memory parity error.

Flow of Control: Without Interrupts

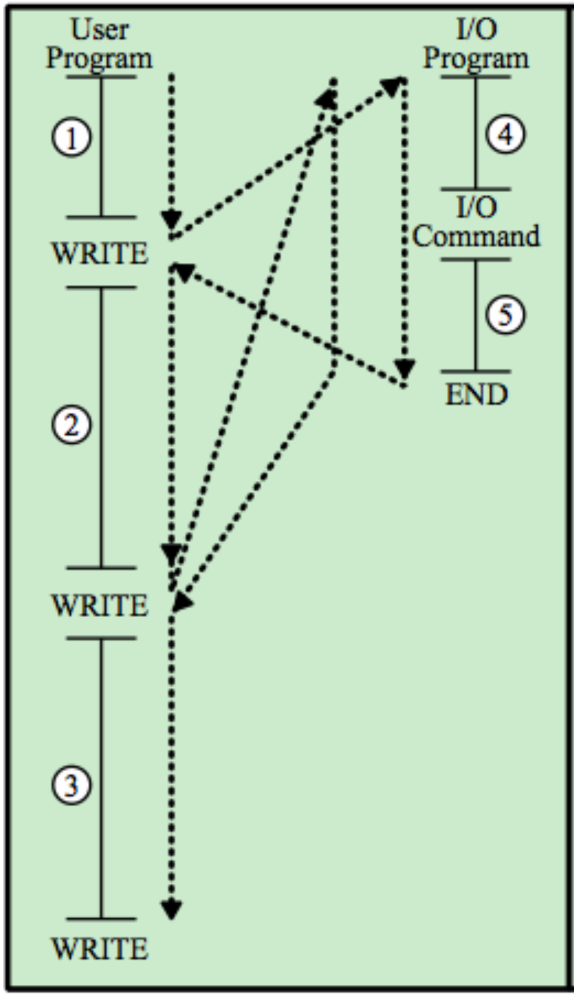


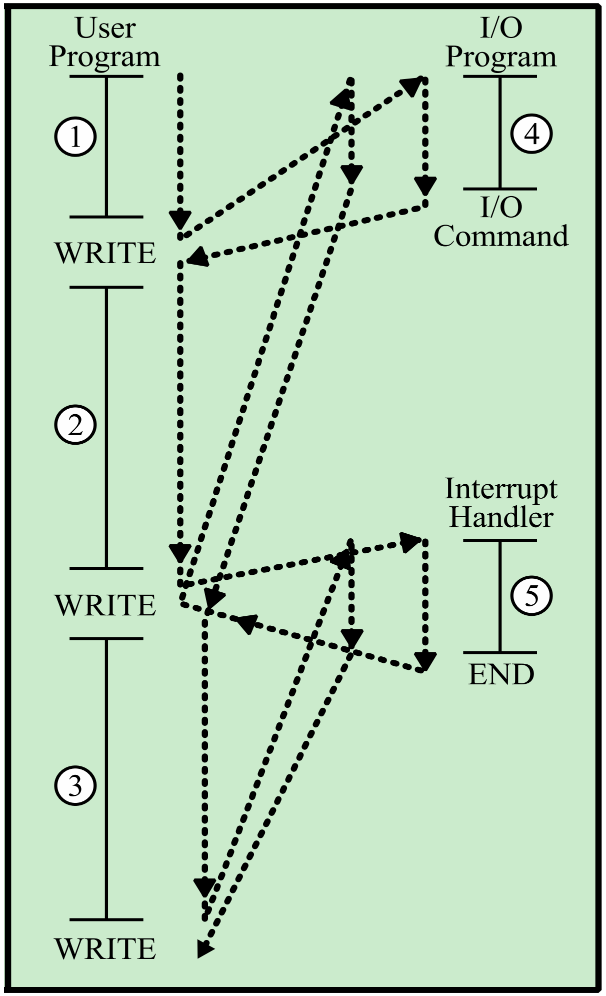
Flow of Control: Short I/O Wait



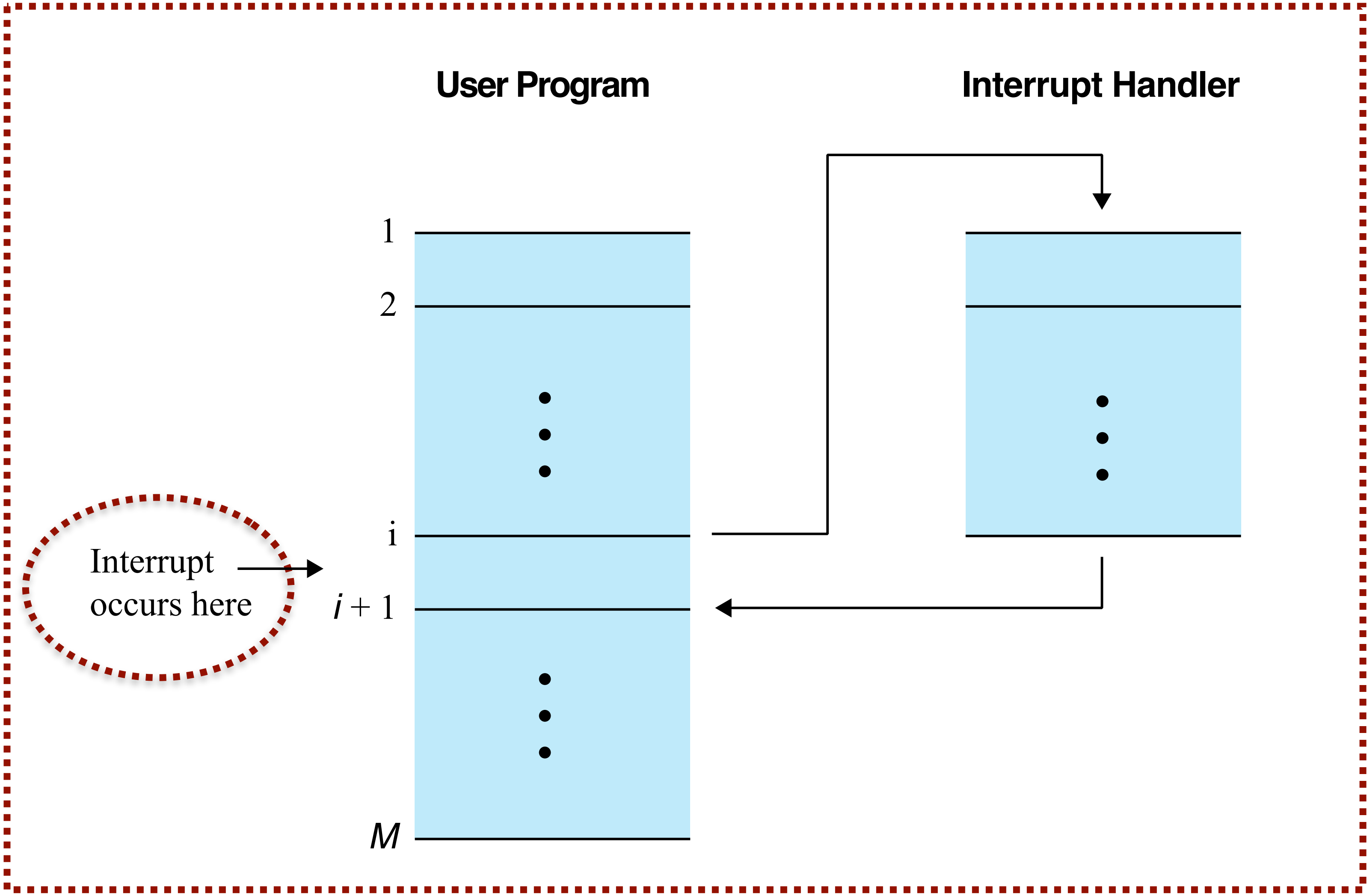


Flow of Control: Long I/O Wait

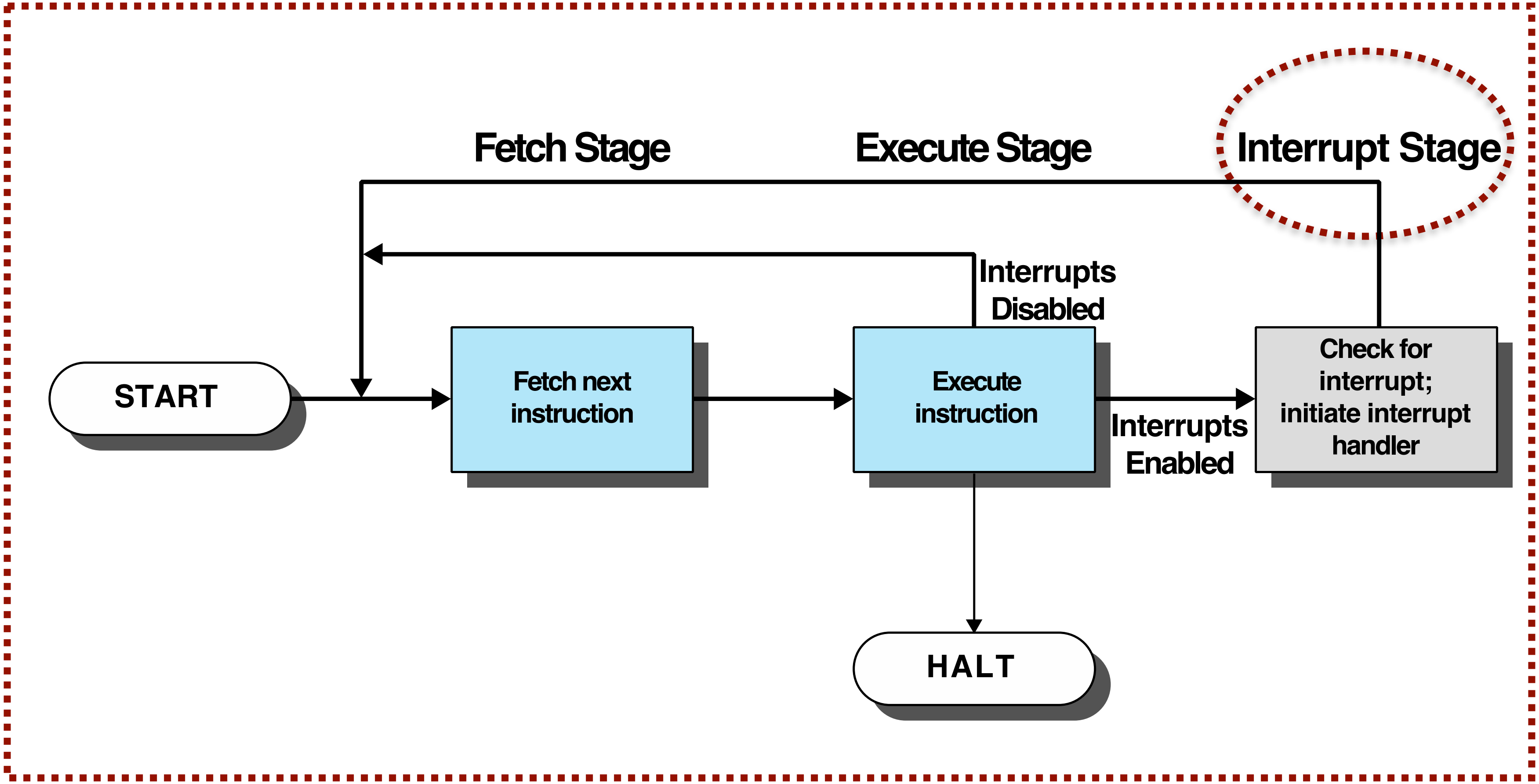




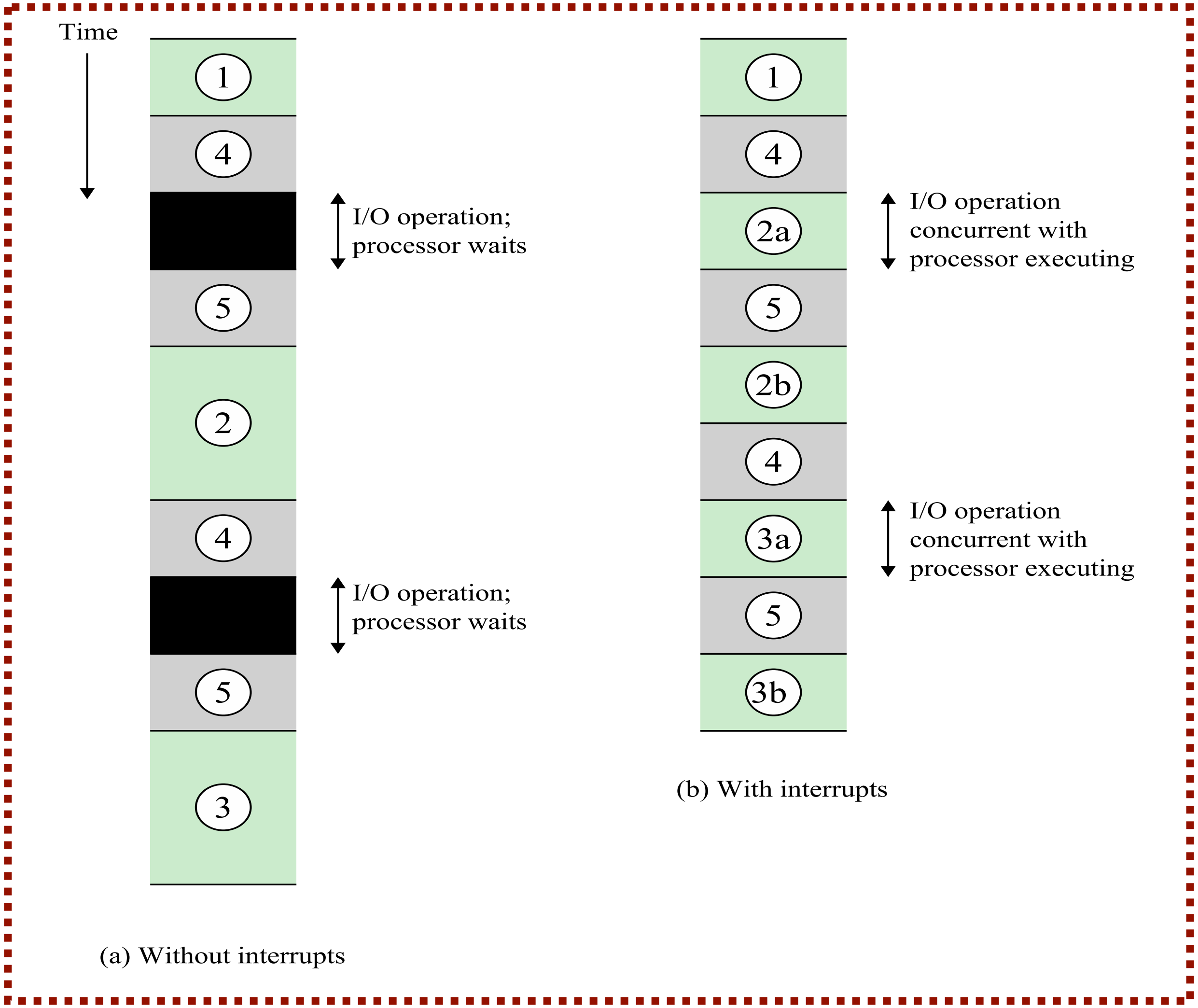
Interrupts: Transfer of Control



Instruction Cycle with Interrupts



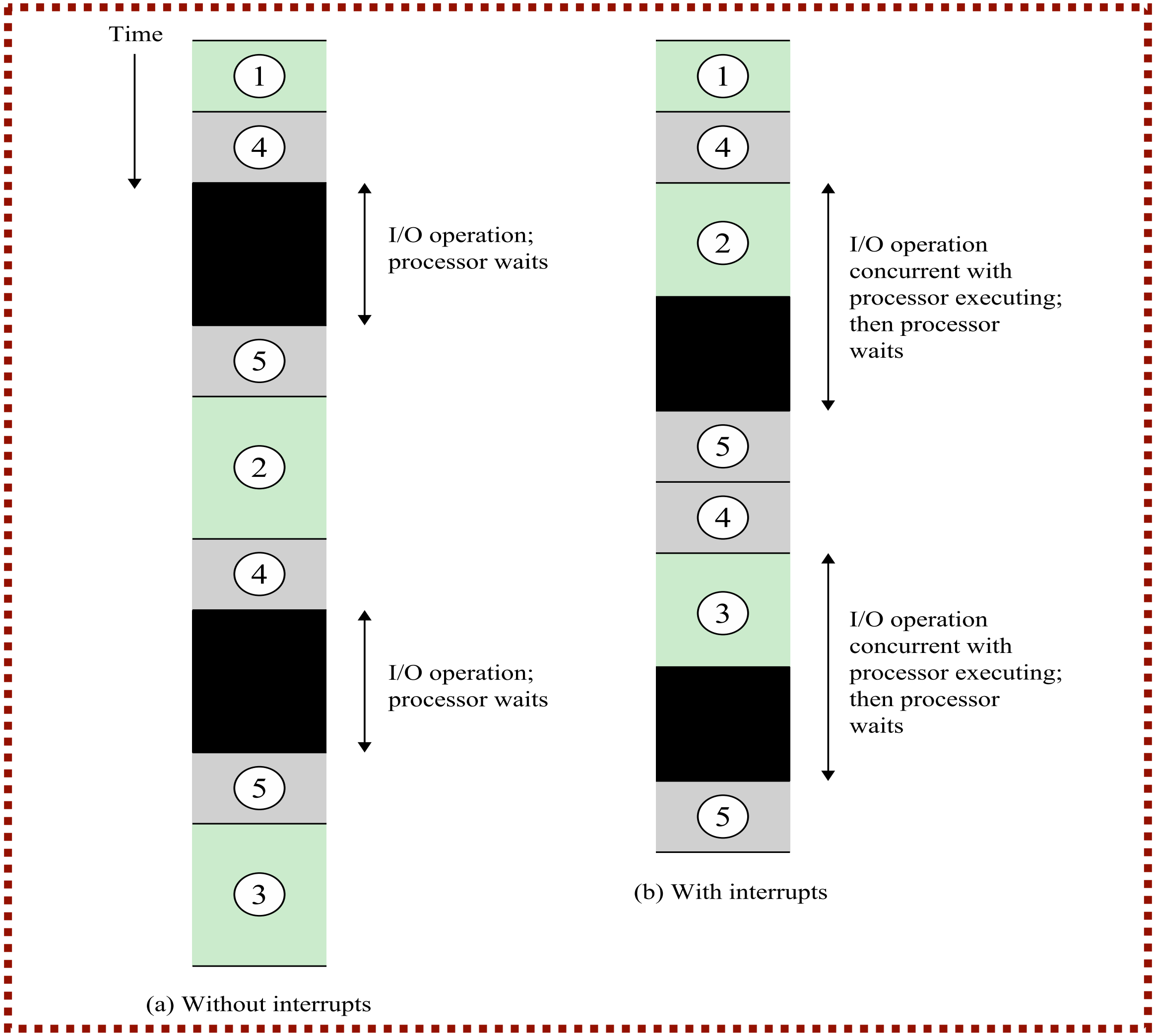
Program Timing: Short I/O Wait



(b) With interrupts

(a) Without interrupts

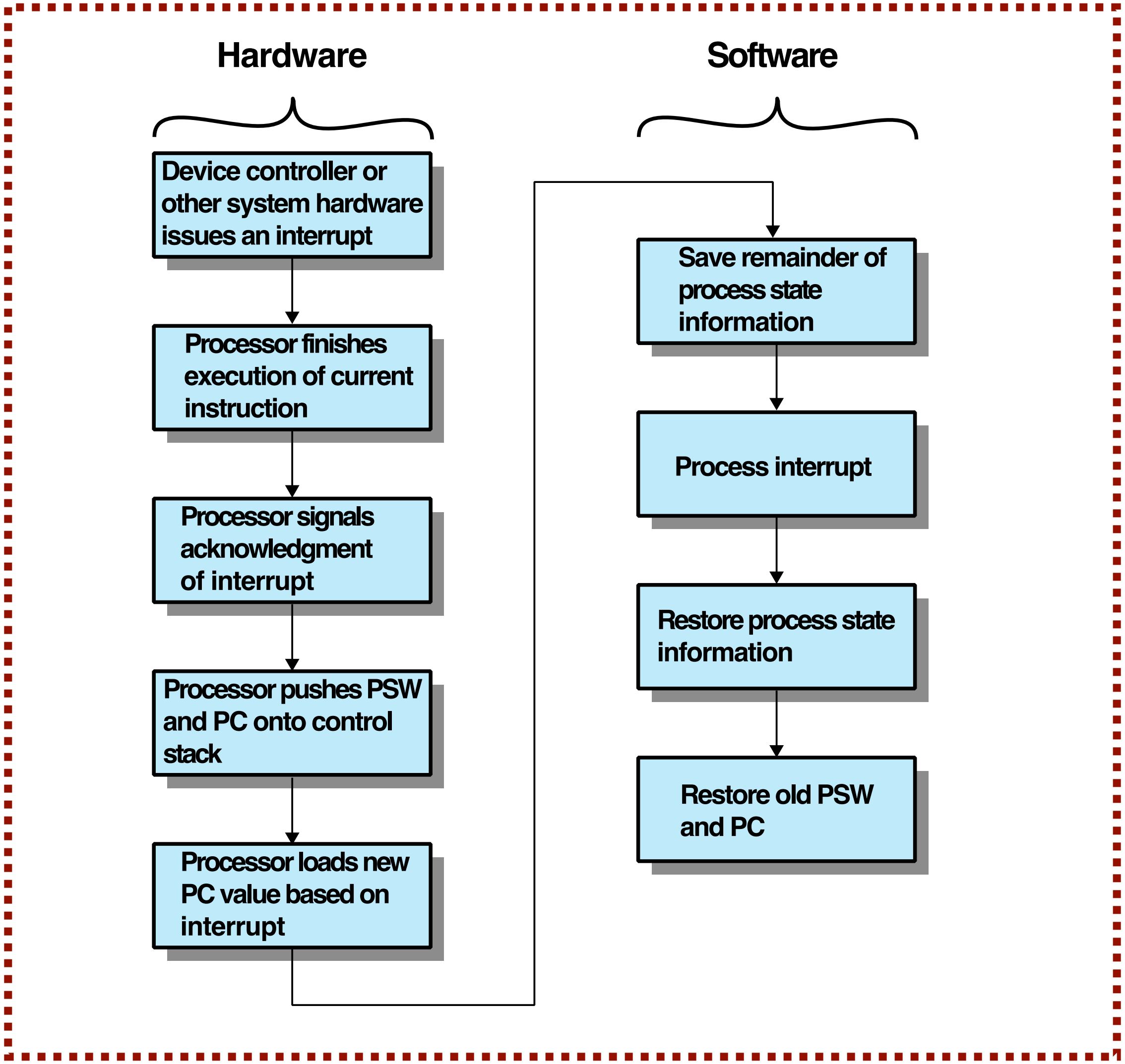
Program Timing: Long I/O Wait


Time
1
4
I/O operation; processor waits
5
2
4
I/O operation; processor waits
5
3
1
4
2
I/O operation concurrent with processor executing; then processor waits
5
4
3
I/O operation concurrent with processor executing; then processor waits
5

(a) Without interrupts

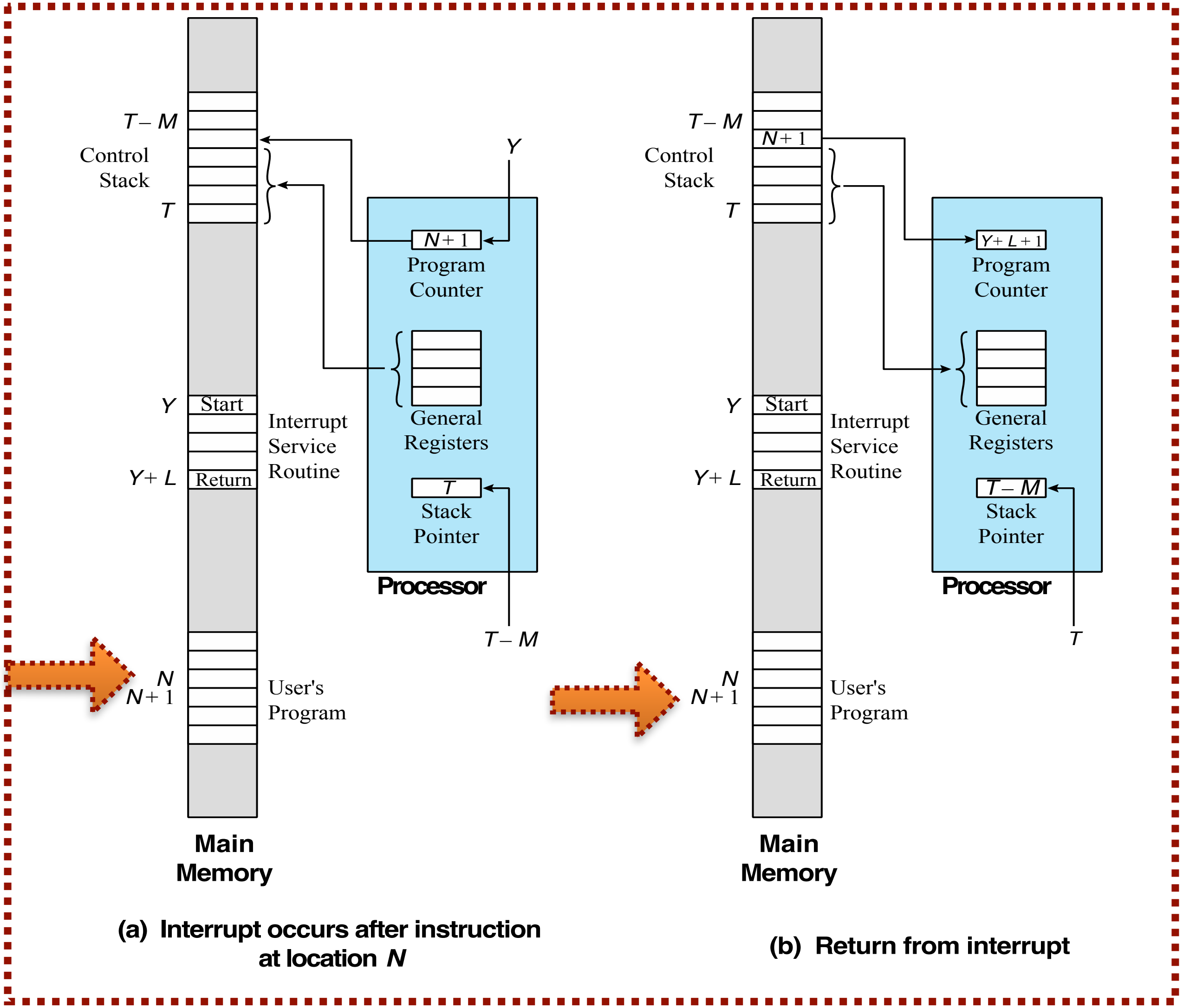
(b) With interrupts

Simple Interrupt Processing



Hardware
Software
Device controller or other system hardware issues an interrupt
Processor finishes execution of current instruction
Processor signals acknowledgment of interrupt
Processor pushes PSW and PC onto control stack
Processor loads new PC value based on interrupt
Save remainder of process state information
Process interrupt
Restore process state information
Restore old PSW and PC

Interrupt: Memory and Register Changes

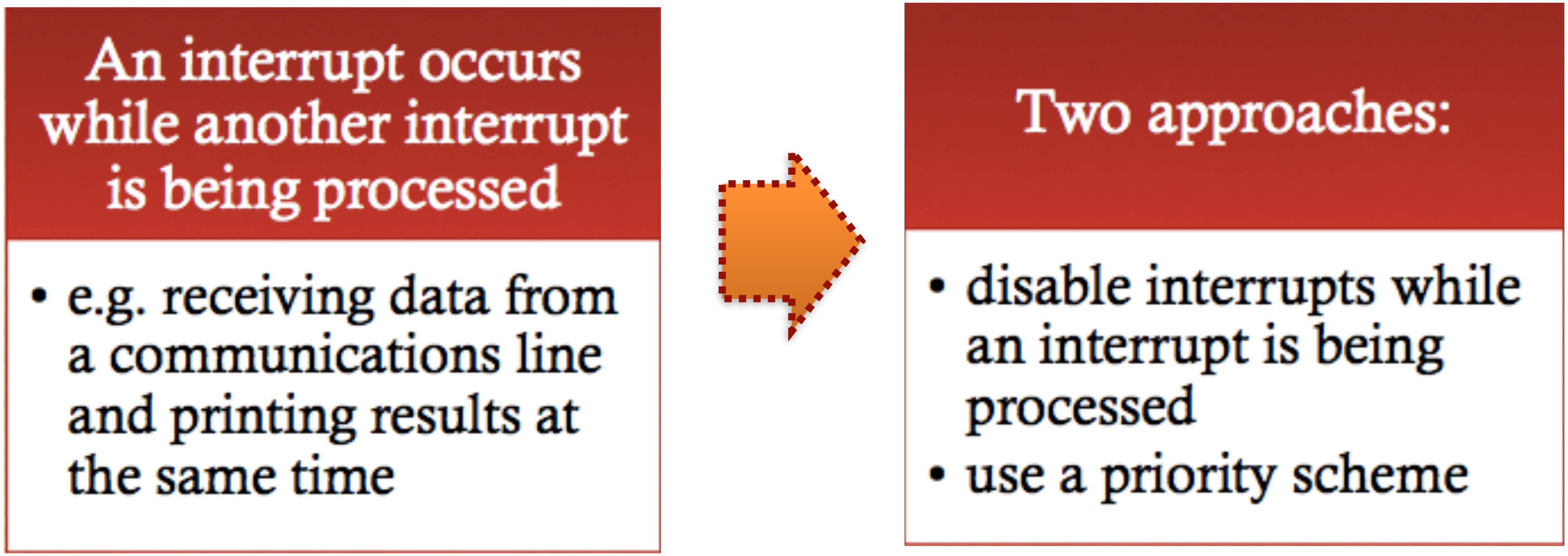


(a) Interrupt occurs after instruction at location *N*

(b) Return from interrupt

MONASH University

Multiple Interrupts



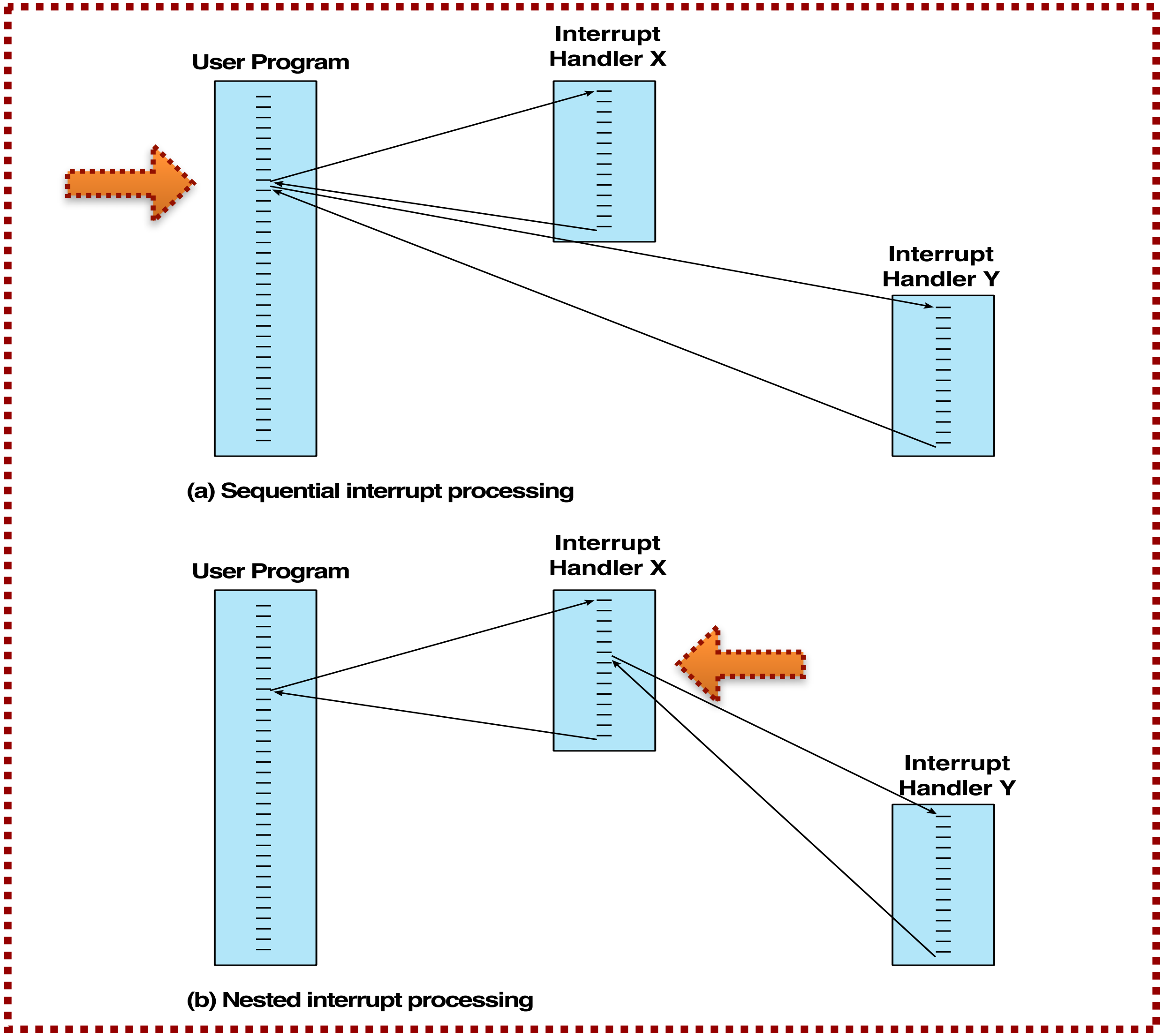
An interrupt occurs while another interrupt is being processed

- e.g. receiving data from a communications line and printing results at the same time

Two approaches:

- disable interrupts while an interrupt is being processed
- use a priority scheme

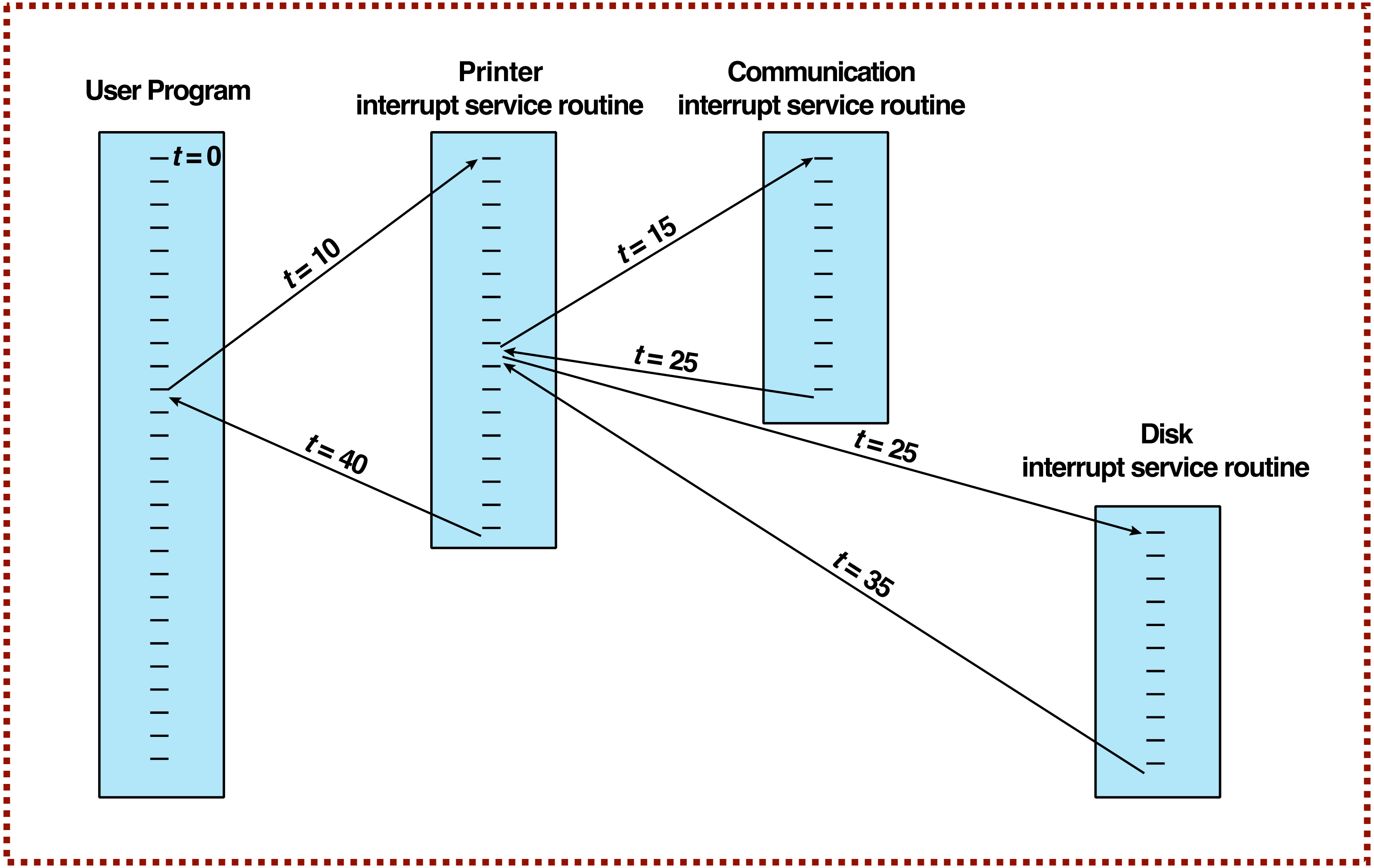
Multiple Interrupts: Transfer of Control



(a) Sequential interrupt processing

(b) Nested interrupt processing

Multiple Interrupts: Time Sequence



MONASH University

What are the design constraints of a computer’s memory?

Computer Memory

- Major constraints in a computer’s memory:
  - Amount (capacity)
  - Speed (access time)
  - Expense (cost)

- Memory must be able to keep up with the processor

- Cost of memory must be reasonable in relationship to other components

Memory Relationships

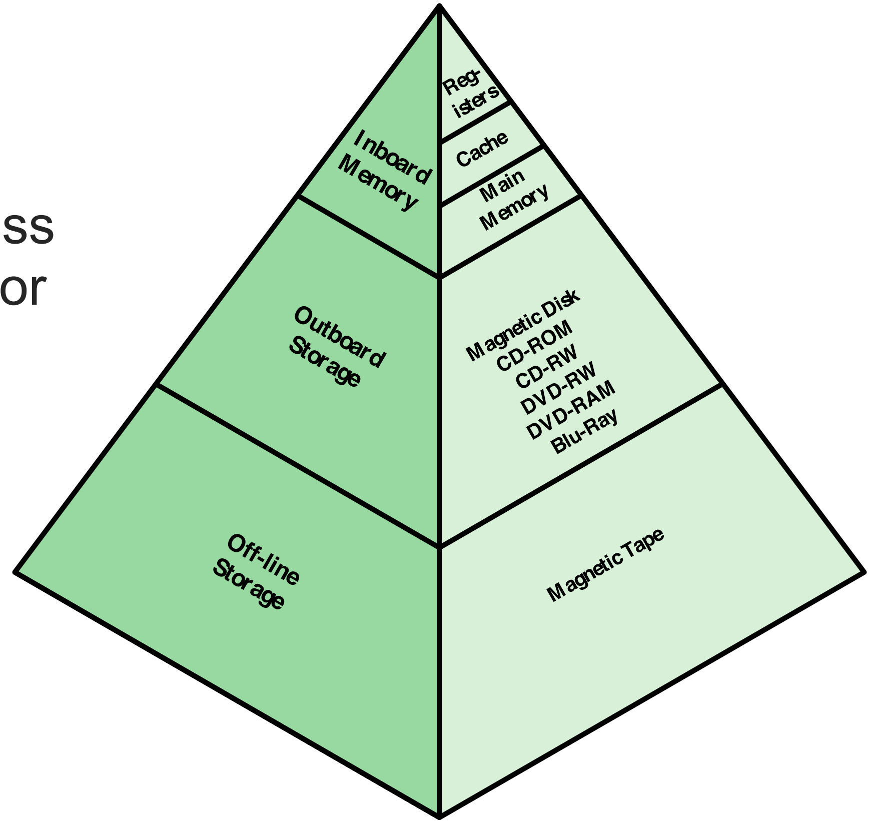
Faster access time = greater cost per bit

Greater capacity = smaller cost per bit

Greater capacity = slower access speed

The Memory Hierarchy

- Moving down the hierarchy:
  - Decreasing cost per bit
  - Increasing capacity
  - Increasing access time
  - Decreasing frequency of access to the memory by the processor



Principle of Locality

- Memory references by the processor tend to cluster

- Data is organised so that the percentage of accesses to each successively lower level is substantially less than that of the level above

- Can be applied across more than two levels of memory

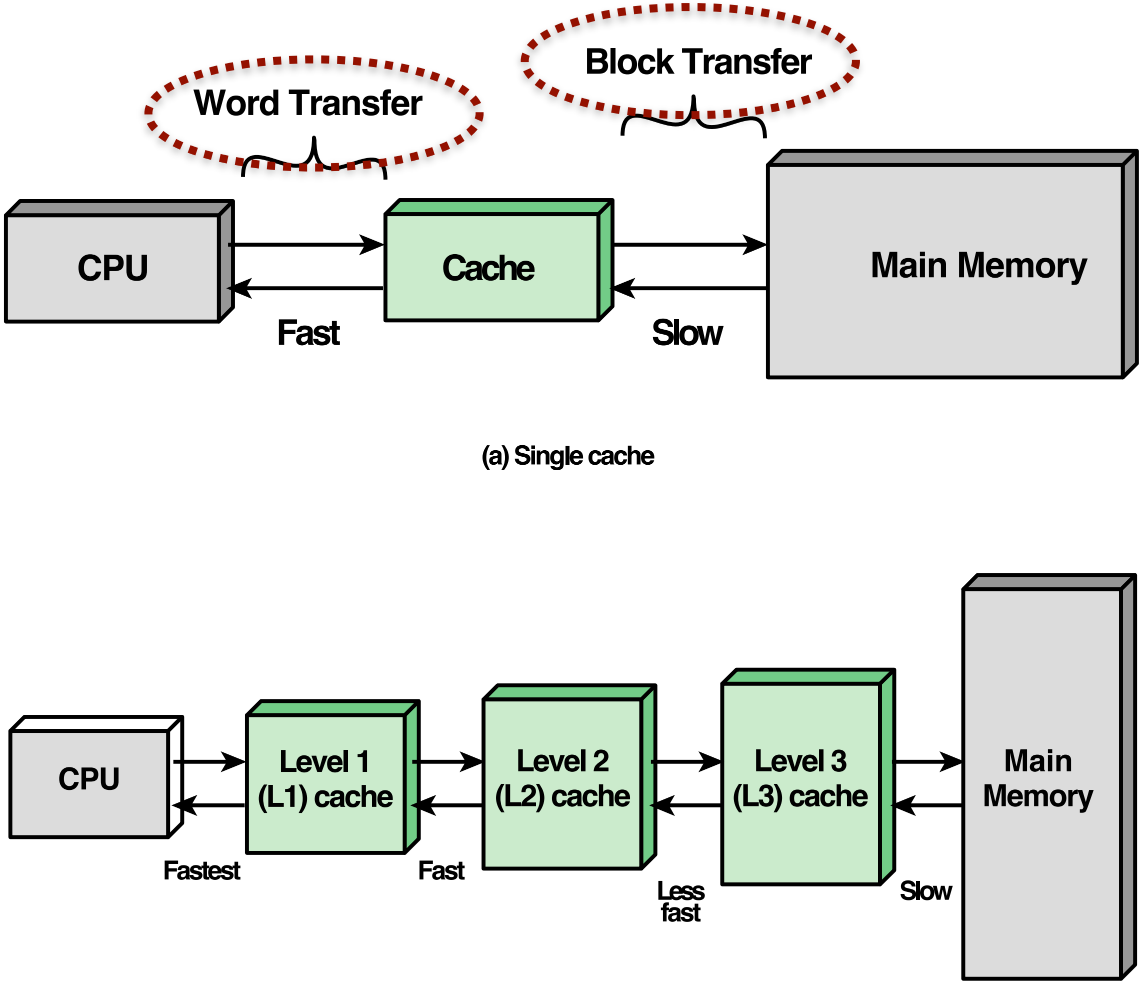
Cache Memory

- Invisible to the OS
- Interacts with other memory management hardware
- Motivation:
  - Processor must access memory at least once per instruction cycle
  - Processor execution is limited by memory cycle time
- Exploit the principle of locality with a small, fast memory

Cache Principles

- Cache contains a copy of portion of main memory
- Processor first checks the cache
- If not found, a block of main memory is read into the cache
- Because of locality of reference, it is likely that many of the future memory references will be to other bytes within the block

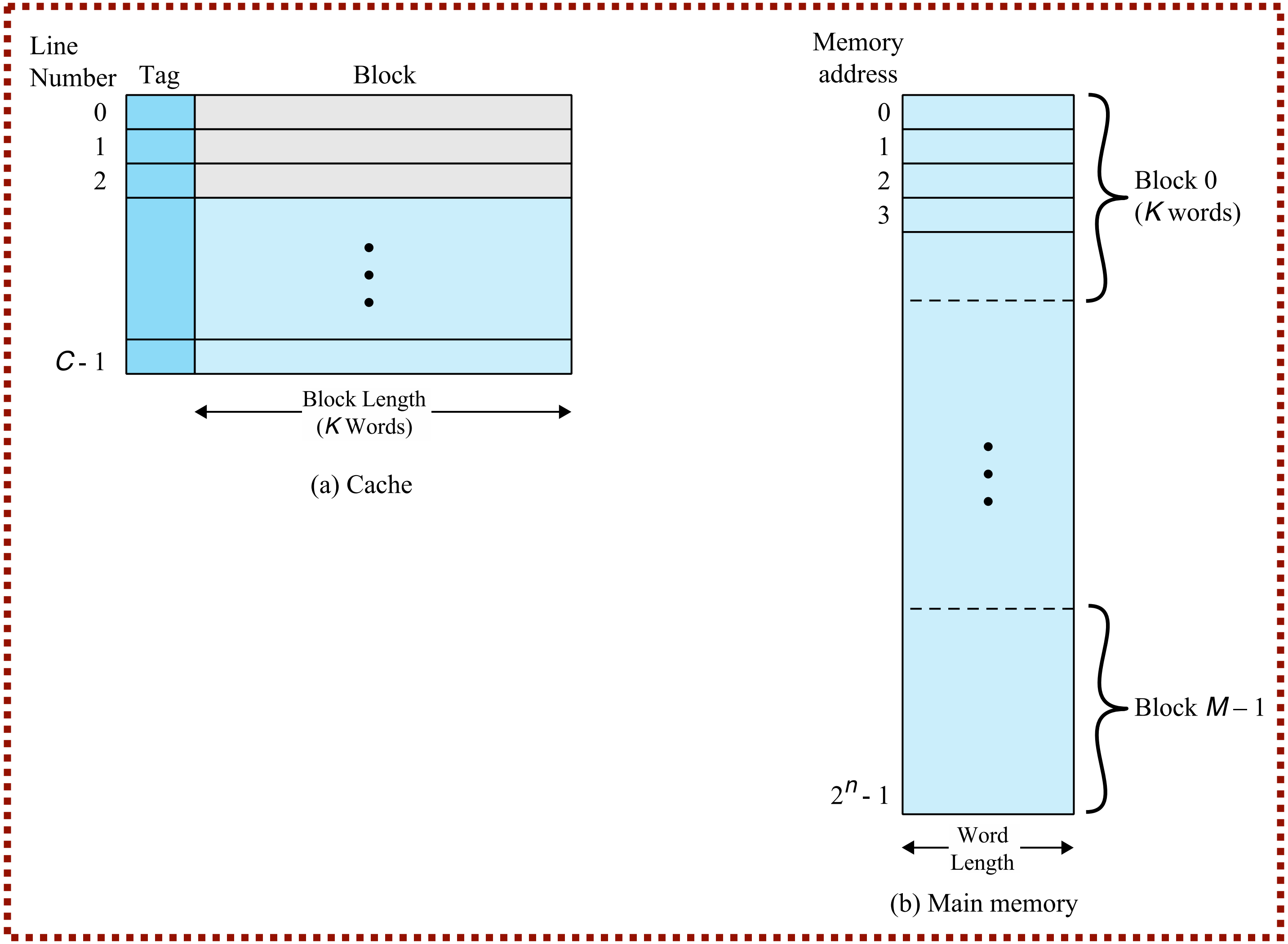
Cache vs Main Memory



(a) Single cache

(b) Three-level cache organization

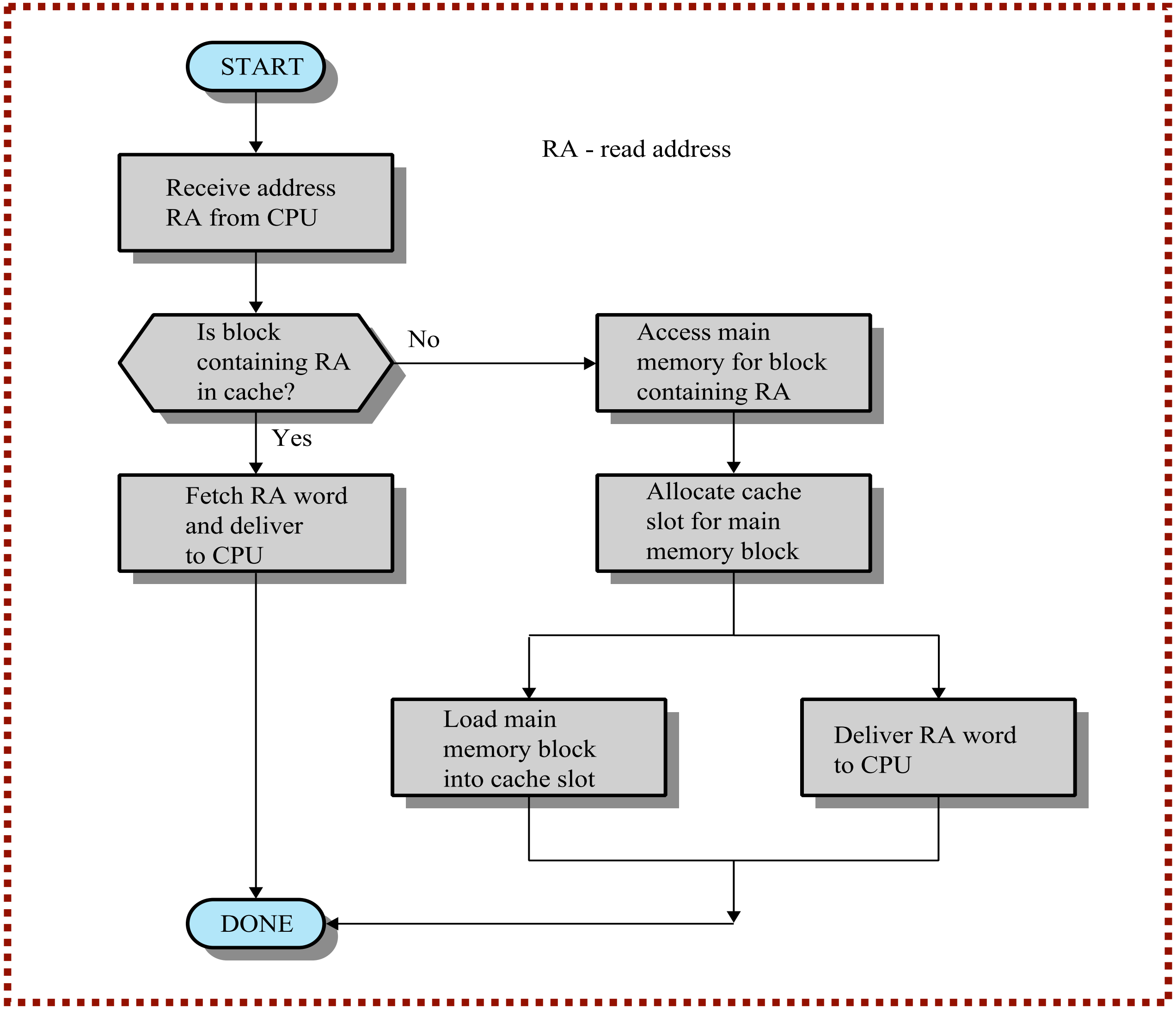
Structure: Cache vs Main Memory



(a) Cache

(b) Main memory

Cache: Read Operation


START
RA - read address
Receive address RA from CPU
Is block containing RA in cache?
No
Yes
Access main memory for block containing RA
Fetch RA word and deliver to CPU
Allocate cache slot for main memory block
Load main memory block into cache slot
Deliver RA word to CPU
DONE

MONASH University

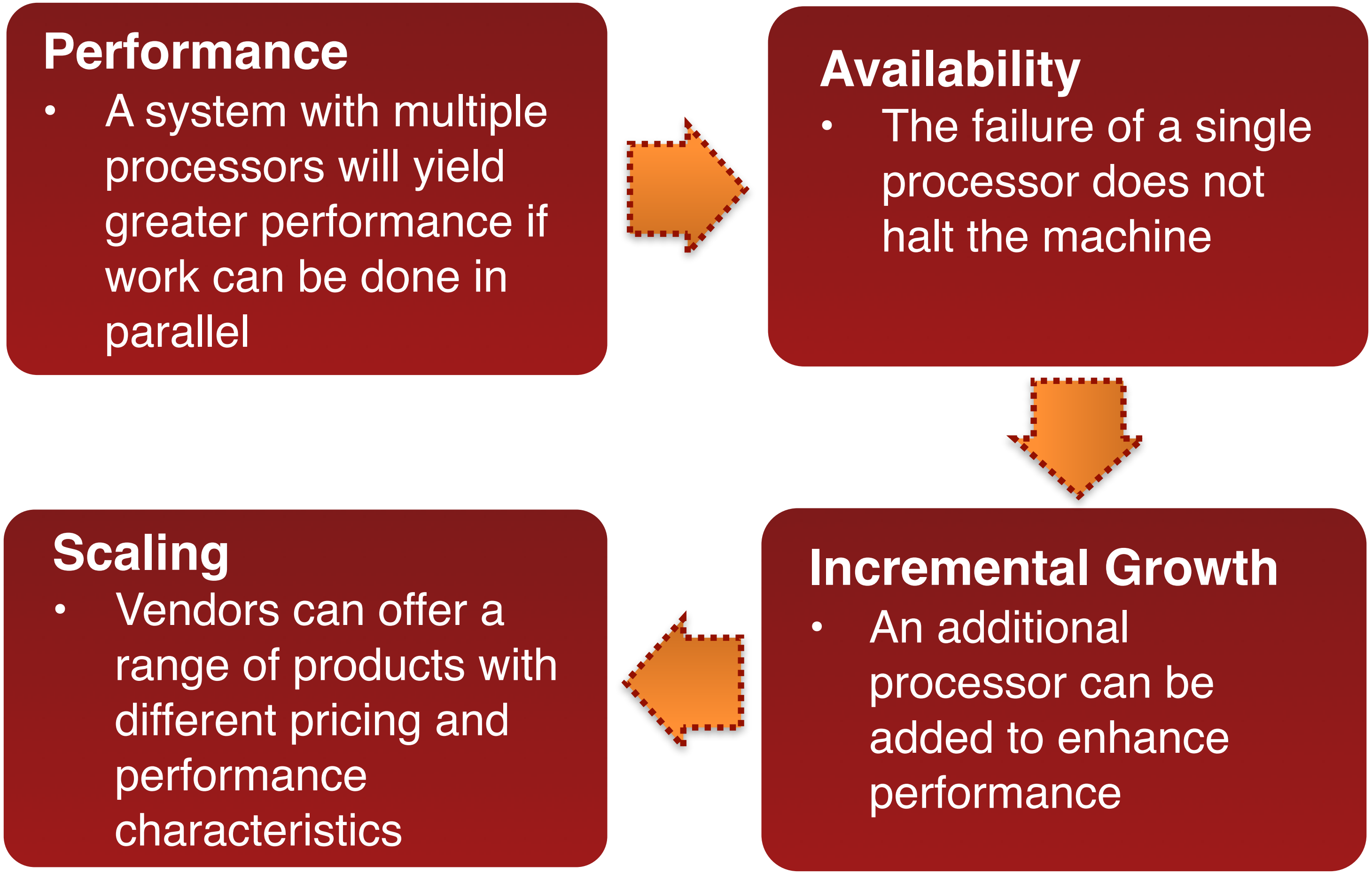
What are multiprocessor systems and multicore computers?

Symmetric Multiprocessors (SMP)

- A stand-alone computer system with the following characteristics:
  - Two or more similar processors of comparable capability
  - Processors share the same main memory and are interconnected by a bus or other internal connection scheme
  - Processors share access to I/O devices
  - All processors can perform the same functions (symmetric)
  - The system is controlled by an integrated operating system that provides interaction between processors and their programs at the job, task, file, and data element levels

The operating system takes care of scheduling of tasks on individual processors and of synchronisation among processors.

SMP: Advantages



Performance

- A system with multiple processors will yield greater performance if work can be done in parallel

Availability

- The failure of a single processor does not halt the machine

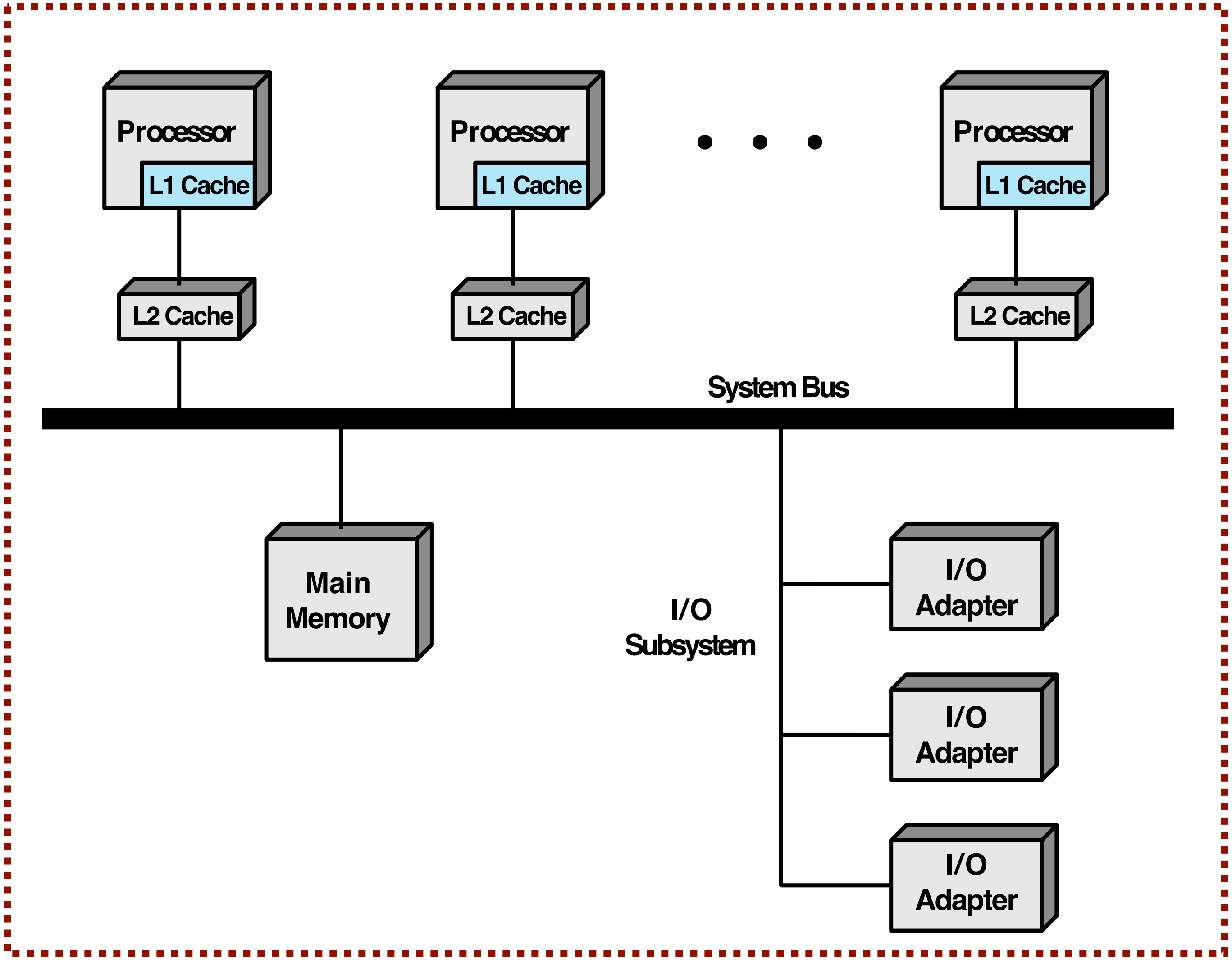
Incremental Growth

- An additional processor can be added to enhance performance

Scaling

- Vendors can offer a range of products with different pricing and performance characteristics

Symmetric Multiprocessor Organisation



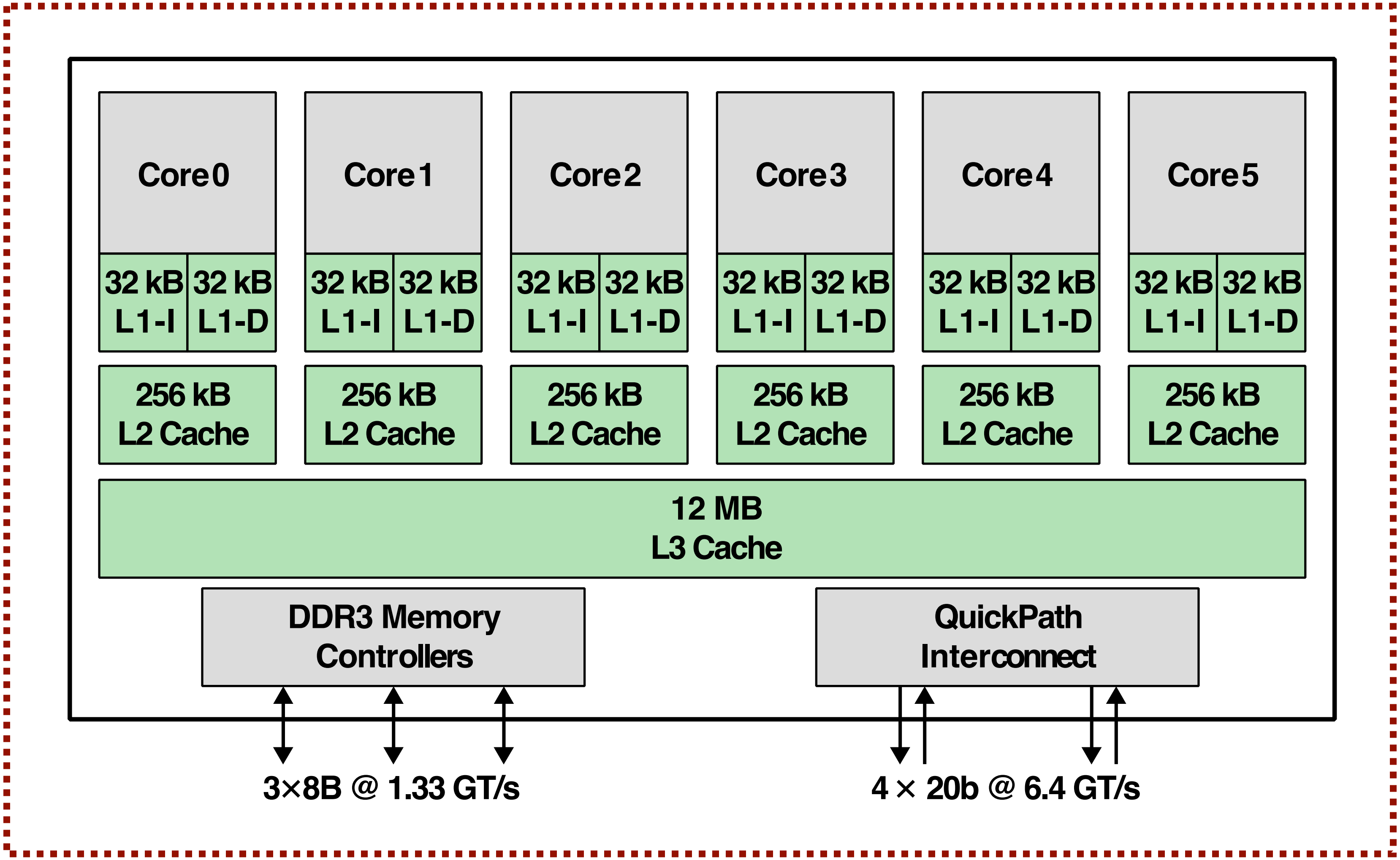
Multicore Computers

- A.k.a. a chip multiprocessor

- Combines two or more processors (cores) on a single piece of silicon (die):
  - Each core consists of all the components of an independent processor, including L1 cache

- Multicore chips also include L2 cache and in some cases L3 cache

Example: Intel Core i7-9990x (32-bit Architecture)



Summary

- So far, we have discussed:
  - Basic elements of a computer system
  - Instruction execution cycle
  - Interrupts and interrupt processing
  - Memory hierarchy and cache memory
  - Multiprocessor systems and multicore computers

- Next week:
  - Overview of operating systems

Reminder: Tutorials begin this week.