Computer System Overview

Basic Elements

- Processor
- Main Memory
- I/O modules
- System Bus

- At a top level, a computer consists of processor, memory, and I/O components, with one or more modules of each type. These components are interconnected in some fashion to achieve the main function of the computer, which is to execute programs.

Processor

Controls the operations of the computer

Performs the data processing functions

Referred to as the Central Processing Unit

Main Memory

- Stores data and programs
- Typically, volatile
  - Contents of the memory is lost when the computer is shut down
- Referred to as real memory or primary memory
- Hard disk (Secondary storage) and main memory are two different things.

I/O Modules

Move data between the computer and its external environments

- Secondary memory devices
- Communication equipment
- Terminals

System Bus

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

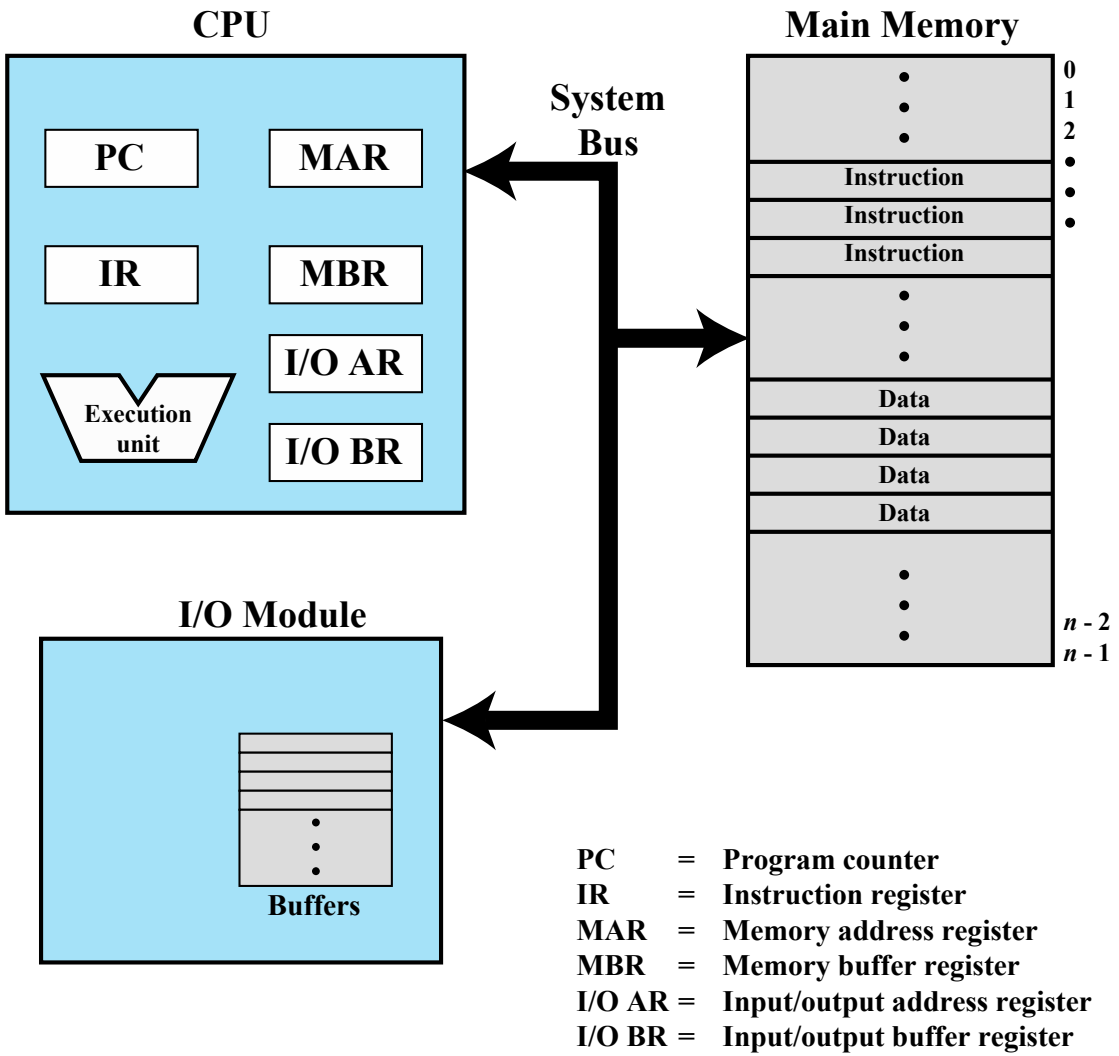


Figure 1.1 Computer Components: Top-Level View

Microprocessor

- Contains a processor on a single chip
- Microprocessors are now multiprocessors.

GPU

- Provide efficient computation on arrays of data using Single-Instruction Multiple Data (SIMD) techniques pioneered in supercomputers
- Not just used for rendering graphics
  - Training and running neural networks
  - Scientific computation
  - Crypto mining
  - Big data analysis
  - And many more….

Digital Signal Processor (DSP)

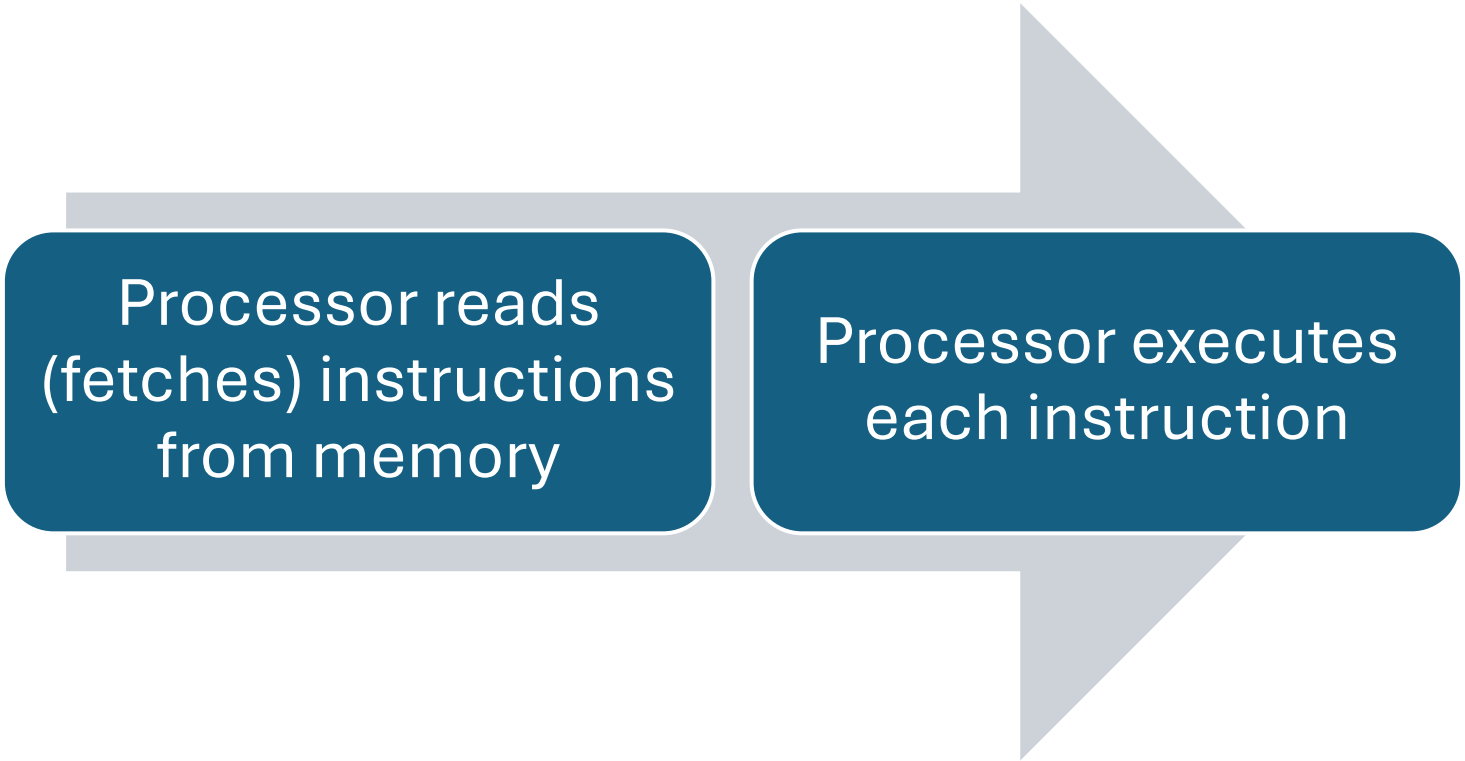
- Deal with streaming signals such as audio or video.
- Used to be embedded in I/O devices like modems
  - Are now becoming first-class computational devices, especially in handhelds
- Encoding/decoding speech and video (codecs)
- Provide support for encryption and security
  - Ex: Broadcom BCM5862x Series DSP

System on a Chip (SoC)

- The classic microprocessor is giving way to the SoC.
- Other components of the system, such as DSPs, GPUs, I/O devices (such as codecs) and main memory, in addition to the CPUs and caches, are on the same chip.
- Used in handheld devices
  - Qualcomm Snapdragon 8 Gen 3:

Instruction Execution

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



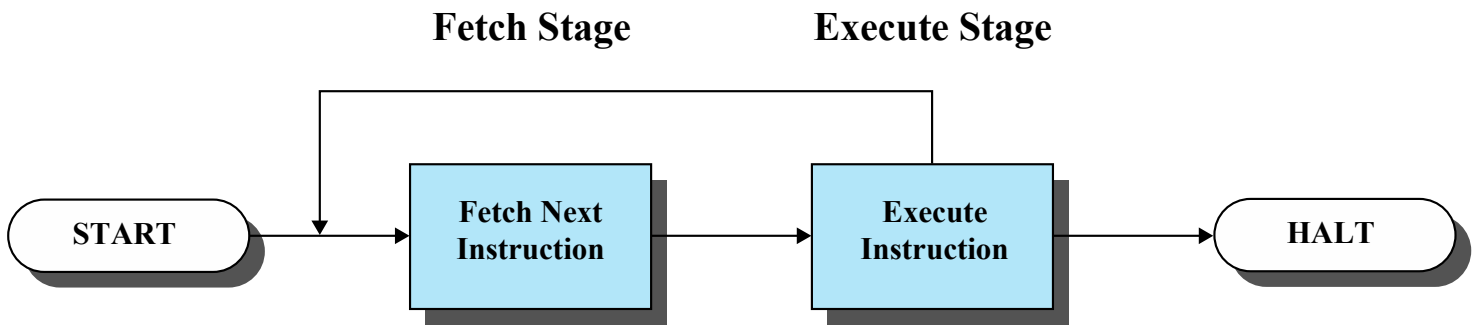


Figure 1.2 Basic Instruction Cycle

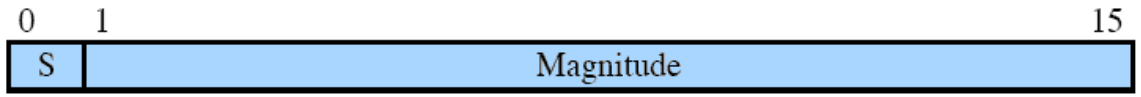
Instruction Fetch and Execute

- The processor fetches an instruction from memory
- The program counter (PC) holds the address of the next instruction to be fetched
  - PC is incremented after each fetch.

Instruction Register

- Fetched instruction is loaded into Instruction Register (IR)
- Processor then interprets the instruction and performs the required action.
  - Processor↔Memory
  - Processor ↔ I/O
  - Data Processing (arithmetic opertations)
  - Control (e.g., branching 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

(d) Partial list of opcodes

Figure 1.3 Characteristics of a Hypothetical Machine

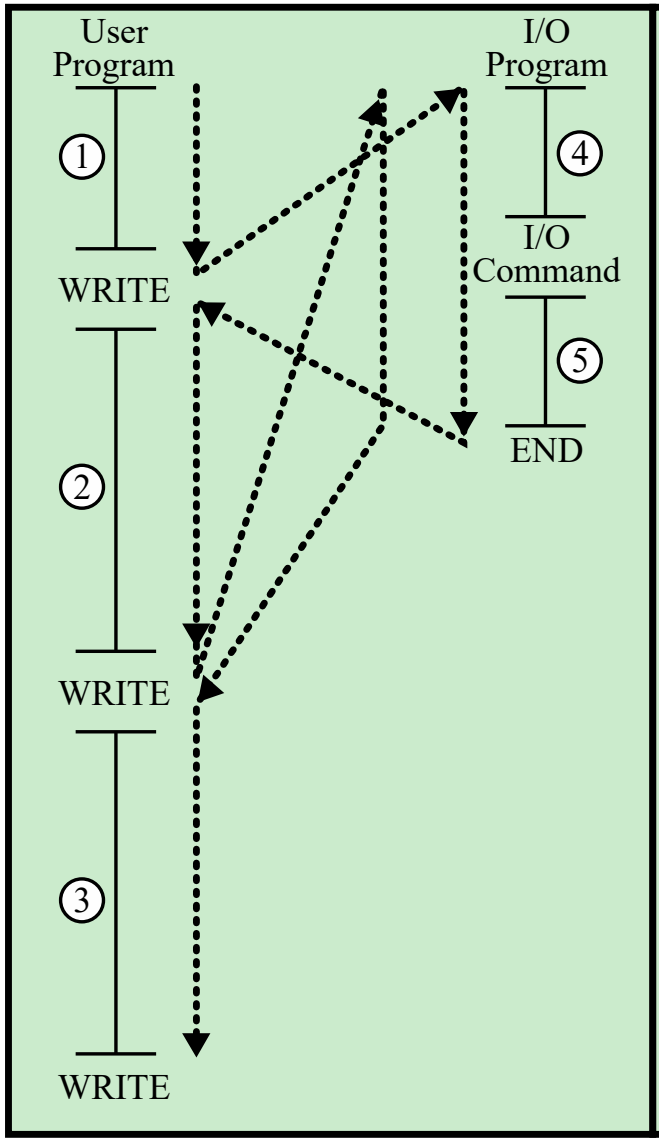
Interrupts

- Mechanism by which other modules may interrupt the normal sequencing of the processor
- Provided to improve processor utilization
    - Most I/O devices are slower than the processor
    - Processor must pause to wait for device
    - Wasteful use of the processor

Types of interrupts

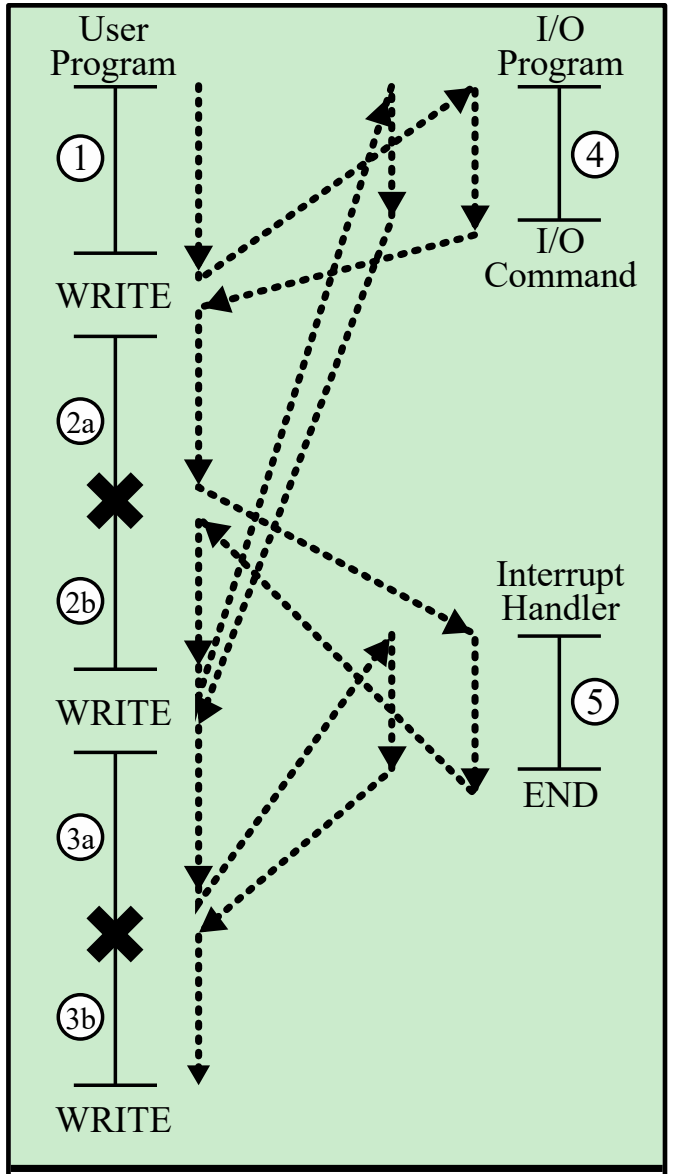
Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows 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 failure parity error.

- What if we do not have interrupts
- Let's look at a flow control without interrupts.



(a) No interrupts

- With interrupts



(b) Interrupts; short I/O wait

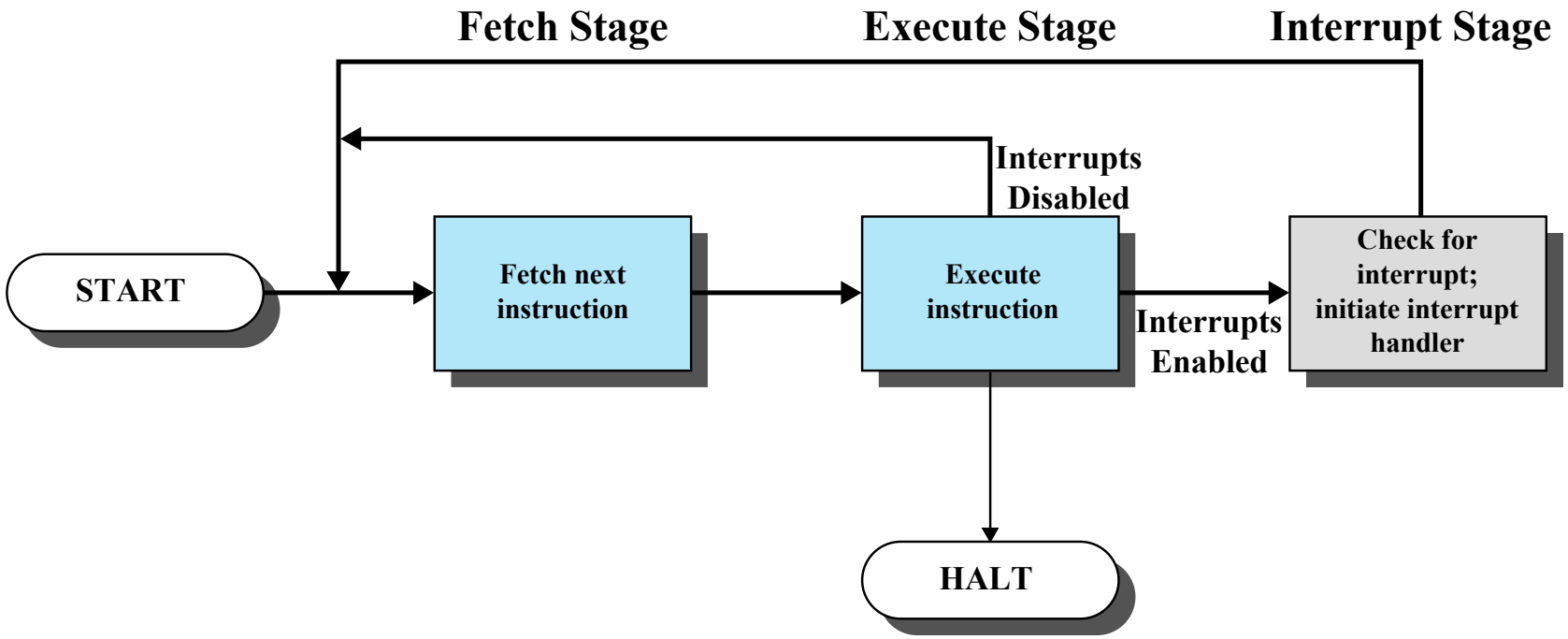


Figure 1.7 Instruction Cycle with Interrupts

Operating System

- OS is the computing system that manages all the hardware and software.
  - Controls the execution of application programs.
  - Acts as an interface between applications and hardware.
- Provides a set of services to system users.
- Objectives:
  - Convenience
  - Efficiency
  - Ability to evolve

Operating System

- In the first part of this course, we will mainly focus on how OS works, the related concepts as well as algorithms.
- Some concepts related to distributed computing and network operating systems will be covered later in the course.
- Some concepts on computer architecture and hardware will be covered as well.

What is OS composed of?

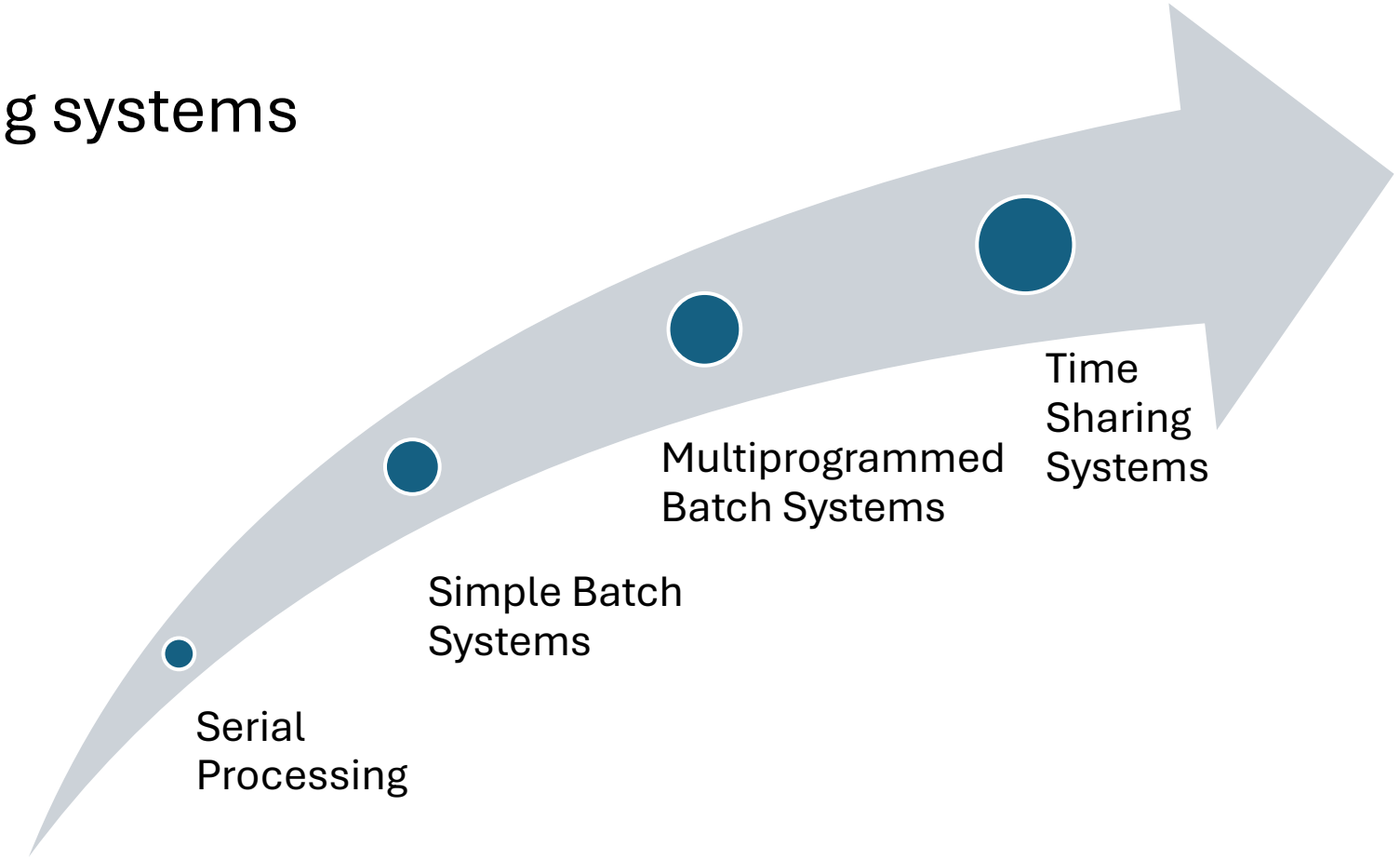
- Memory Manager
  - Controls and manages the main memory.
- Processor Manager
  - Decides how to allocate processing power to processes.
- Device Manager
  - Manages every device, channel, and control unit.
- File Manager
  - Manges file system.
- Network Manager
  - Manages network connections.

Operating system services

- Program development
- Program execution
- Access I/O devices
- Controlled access to files
- System access
- Error detection and response
- Accounting

Evolution of operating system

- Types of operating systems



Serial processing

- No operating system
- Programmers interacted directly with the hardware.
- Users have access to the computers in “series”
- Problems
  - Scheduling was horrendous.
  - Took considerable amount of time to set up a program to run.

Batch processing

- People wanted to maximize processor utilization.
- Introduced Monitor.
- Monitor
  - User no longer has direct access to processor.
  - Job is submitted to computer operator who batches them together and places them on an input device.
  - Program branches back to the monitor when finished.

Batch processing

- Monitor controls the sequence of events.
- Resident Monitor is a software that is always in the memory.
- Monitor reads in job and gives control.
- Job returns control to monitor.

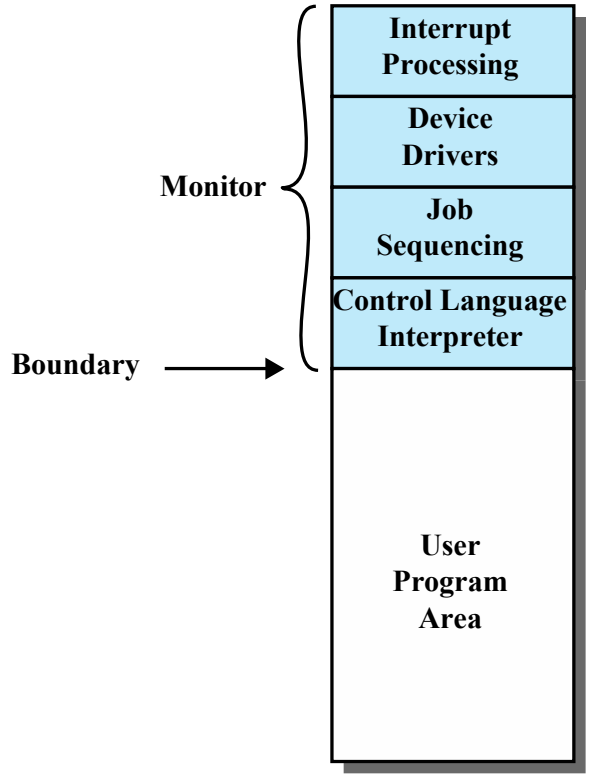


Figure 2.3 Memory Layout for a Resident Monitor

Batch processing

- From the point of view of the processor:
  - Processor executes instructions from the memory containing monitor.
  - At some point it has instruction to load and execute next job.
  - Processor executes the job until it ends or finds an error condition.
  - Either of these two event causes processor to invoke instructions from the monitor.

Important to note the following

- Monitor is just a software.
- For monitor to work properly special hardware features must be available.

Memory protection

- While the user program is executing, it must not alter the memory area containing the monitor

Timer

- Prevents a job from monopolizing the system

Privileged instructions

- Can only be executed by the monitor

Interrupts

- Gives OS more flexibility in controlling user programs

Memory protection led to modes of operation

User Mode

- User program executes in user mode
- Certain areas of memory are protected from user access
- Certain instructions may not be executed

Kernel Mode

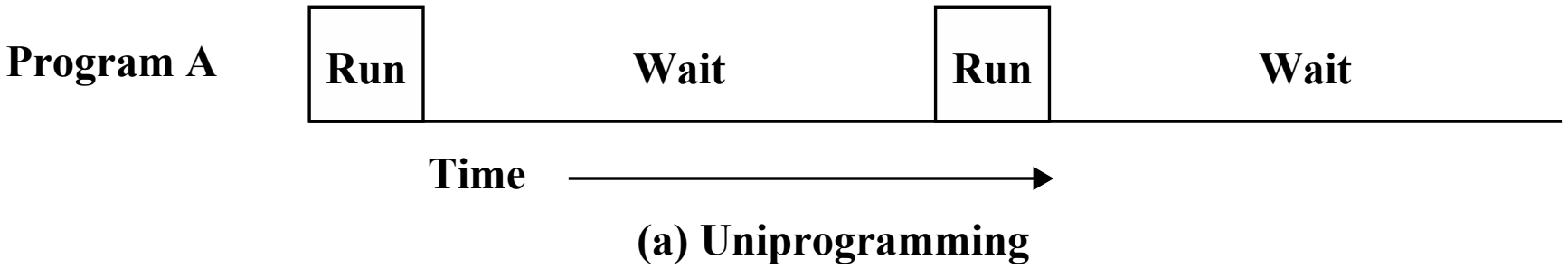
- Monitor executes in kernel mode
- Privileged instructions may be executed
- Protected areas of memory may be accessed

Read one record from a file	$15\mu s$
Execute 100 instructions	$1\mu s$
Write one record to file	$15\mu s$
Total	$31\mu s$
Percent of CPU utilization	$\frac{1}{31} = 0.032 = 3.2\%$

Multiprogrammed batch systems

- Even though batch systems somewhat improved processor utilization, processor is often idle because of I/O operations.
- This led to multiprogrammed batch systems.

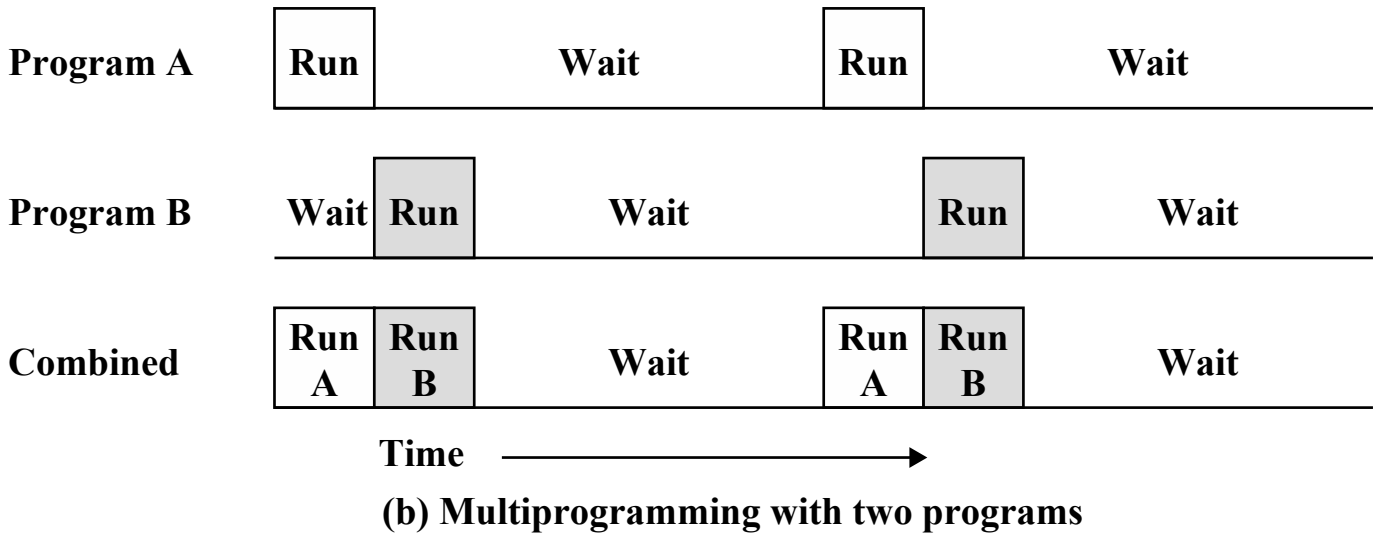
Uniprogramming



(a) Uniprogramming

The processor spends a certain amount of time executing, until it reaches an I/O instruction; it must then wait until that I/O instruction concludes before proceeding

Multiprogramming



(b) Multiprogramming with two programs

Time sharing systems

- In multiprogramming batch systems users did not interact with the computer directly.
- People needed computers to handle interactive jobs.
- For this to happen a new type of OS was introduced—Time sharing systems.
- Multiple users can simultaneously access the computer through terminals.
- OS interleaves execution of each user program in short burst or quantum of computation.

Time sharing systems vs Batch Multiprogramming systems

	Batch Multiprogramming	Time sharing
Principle objective	Maximize processor use	Maximize response time
Source directive to OS	Job control language commands provided with the job	Commands entered at the terminals.

OS generations

- Early computing (1940-1950)
  - No OS
- First generation (1950-1960)
  - Mainframes and batch systems
  - Multiprogramming
- Second generation (1960-1970)
  - Multiprogramming and timesharing
- Third generation (1970-1980)
  - Personal computing
  - Development of GUI
- Fourth generation (1980-2000)
  - Microkernels: A modular approach to OS design.
- Modern OS (2000-present)
  - Mobile OS
  - Could and virtualization: Cloud based OS (Ex: Azure)
  - Distributed OS: work seamless across multiple devices
  - Real-time and embedded systems: OS tailored for specific devices and applications