

Chapter 1

Introduction

Yunmin Go

School of CSEE



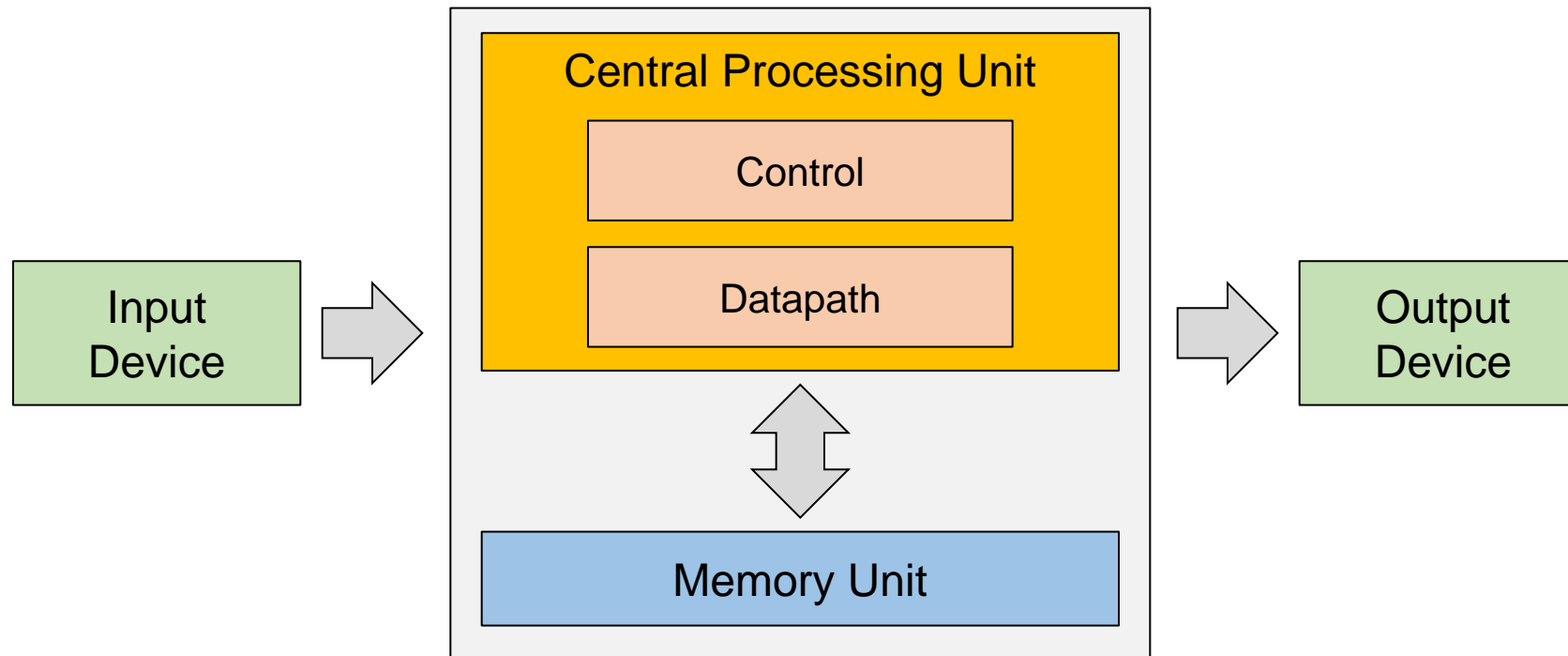
Agenda

- What Happens When a Program Runs
 - What is an Operating System
 - Interrupt-Driven Program
 - Operating System Design Goals

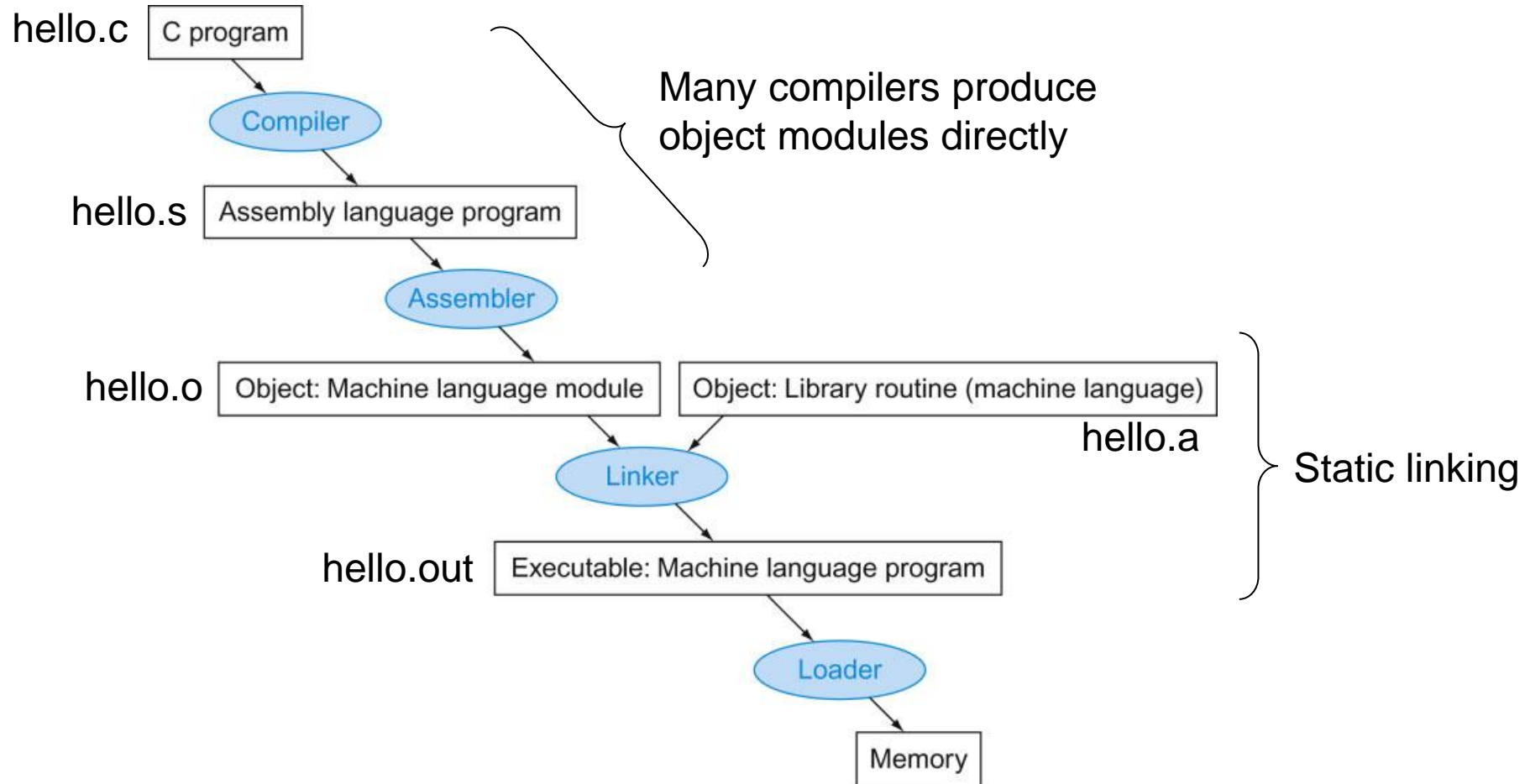


Von Neuman Architecture

- Both program and data are stored in memory
 - stored-program computer



Program Translation



Loading a Program

- Load from image file on disk into memory
 1. Read header to determine segment sizes
 2. Create virtual address space
 3. Copy text and initialized data into memory
 4. Set up arguments on stack
 5. Initialize registers (including \$sp, \$fp, \$gp)
 6. Jump to startup routine
 - Copies arguments to \$a0, ... and calls main
 - When main returns, do exit syscall

Executable file header		
	Text size	300 _{hex}
	Data size	50 _{hex}
Text segment	Address	Instruction
	0040 0000 _{hex}	lw \$a0, 8000 _{hex} (\$gp)
	0040 0004 _{hex}	jal 40 0100 _{hex}

	0040 0100 _{hex}	sw \$a1, 8020 _{hex} (\$gp)
	0040 0104 _{hex}	jal 40 0000 _{hex}
Data segment
	Address	
	1000 0000 _{hex}	(X)

	1000 0020 _{hex}	(Y)

Virtual Memory

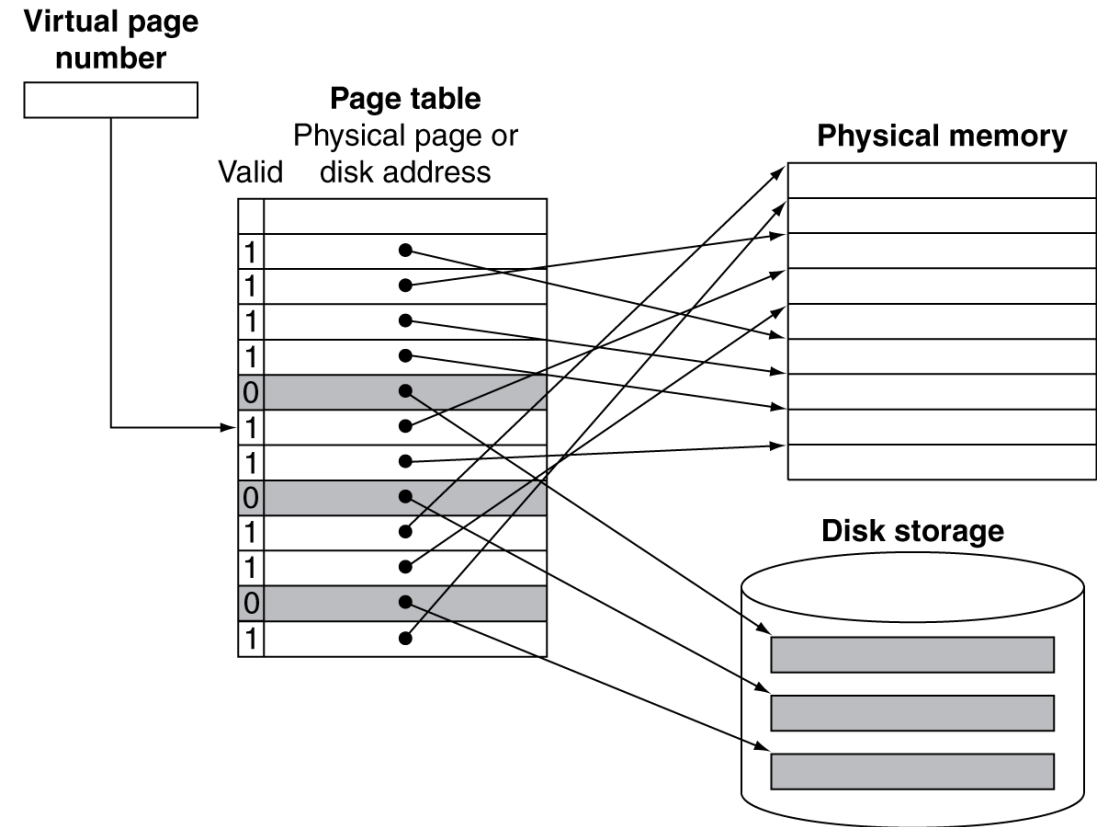
- Main memory is not enough to accommodate all processes on multiprogramming/multitasking system

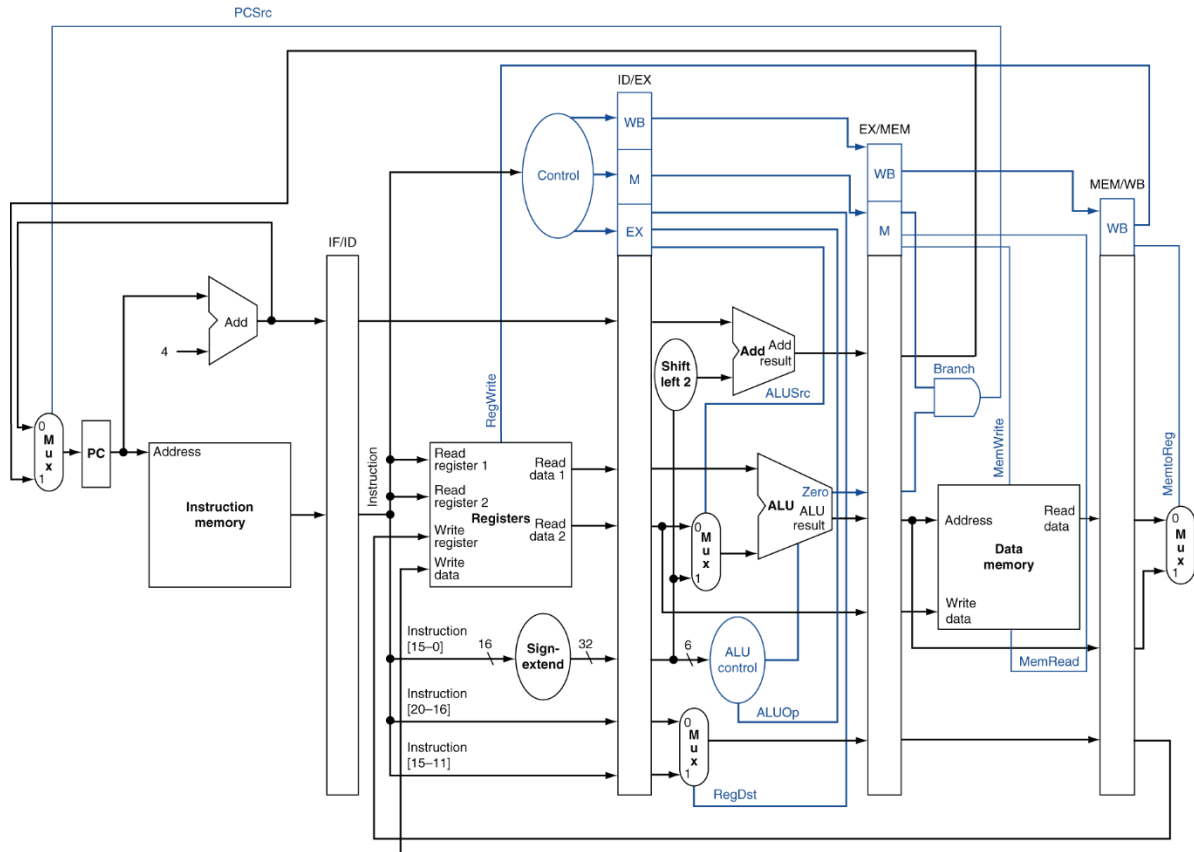
Executable file header		
	Text size	300 _{hex}
	Data size	50 _{hex}
Text segment	Address	Instruction
	0040 0000 _{hex}	lw \$a0, 8000 _{hex} (\$gp)
	0040 0004 _{hex}	jal 40 0100 _{hex}

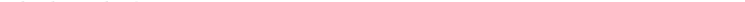
	0040 0100 _{hex}	sw \$a1, 8020 _{hex} (\$gp)
Data segment	0040 0104 _{hex}	jal 40 0000 _{hex}

	Address	
	1000 0000 _{hex}	(X)

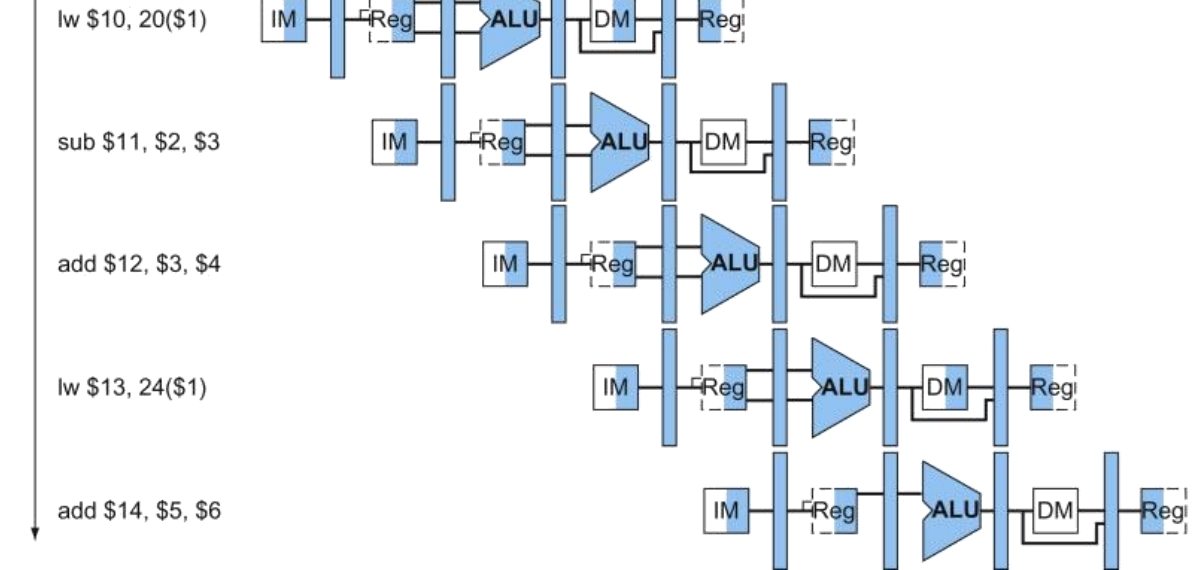
	1000 0020 _{hex}	(Y)





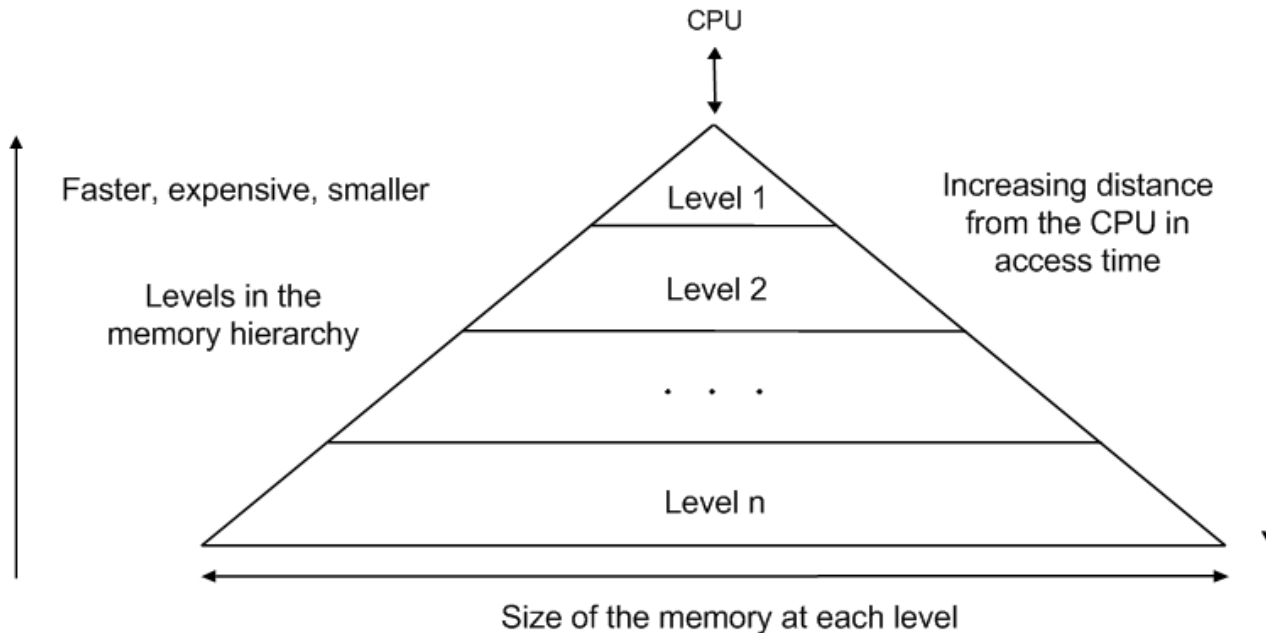
Time (in clock cycles) 

Program
execution
order
(in instructions)



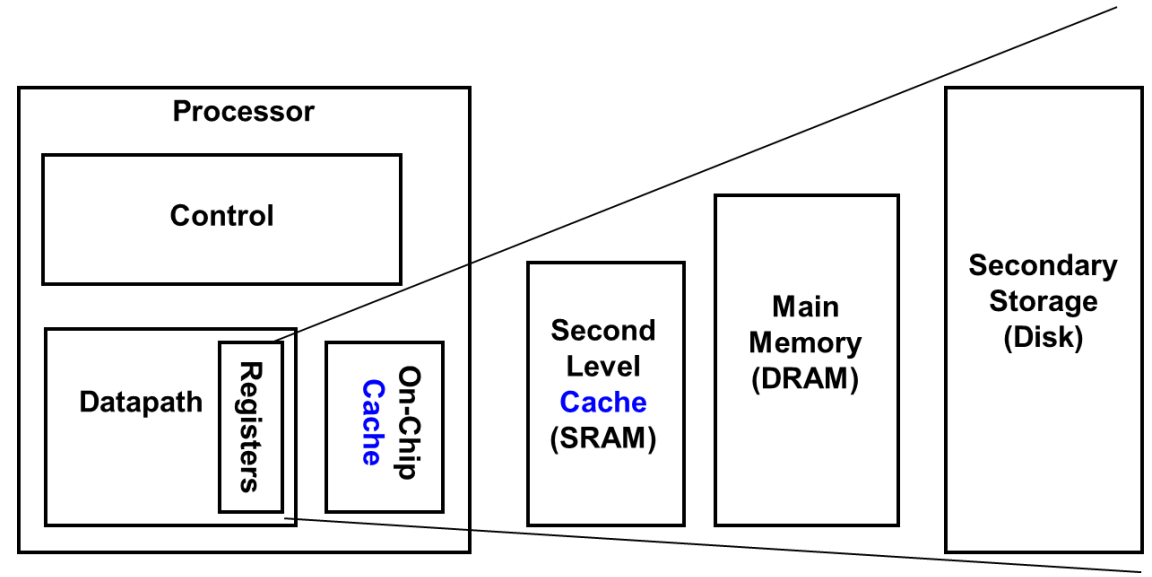
Memory Hierarchy

- Memory hierarchy, illusion of fast and large memory
 - By taking advantage of the principle of locality, it presents the user with as much memory as is available in the cheapest technology and provides access at the speed offered by the fastest technology
 - Temporal locality: Items accessed recently are likely to be accessed again soon
 - Spatial locality: Items near those accessed recently are likely to be accessed soon



Taking Advantage of Locality

- Memory hierarchy
- Store everything on disk
- Copy recently accessed (and nearby) items from disk to smaller DRAM memory
 - Main memory
- Copy more recently accessed (and nearby) items from DRAM to smaller SRAM memory
 - Cache memory attached to CPU



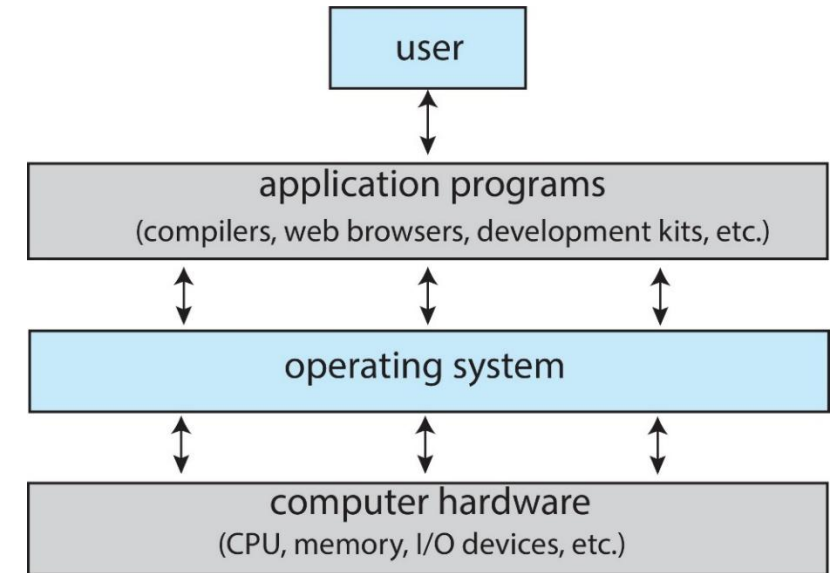
Agenda

- What Happens When a Program Runs
- What is an Operating System
- Interrupt-Driven Program
- Operating System Design Goals



What is an Operating System?

- An **operating system** (OS) is system software that manages computer hardware and software resources, and provides common services for computer programs.*
- Roughly, an OS is ...
 - Intermediary between the user and H/W
 - Kernel + additional programs
 - **Kernel**: core of OS that runs at all times
 - System / application programs are not included



* https://en.wikipedia.org/wiki/Operating_system

What Operating System Do?

- OS provides **environment**
 - Performs no useful function by itself.
 - However, the user can do something easily in the environment provided by OS.
- OS manages **system resources**
 - Let the users and the programs share the system resources in time and space.
 - Let the user use the computer hardware in an efficient manner.

**OS is in charge of making sure the system operates
correctly and efficiently!**

What Operating System Do?

- The OS takes a physical resource and transforms it into a virtual form of itself.
 - Physical resource: Processor, Memory, Disk, ...
 - The virtual form is more general, powerful and easy-to-use.
- The OS manage resources such as CPU, memory and disk.
- The OS allows...
 - Many programs to run → Sharing the CPU
 - Many programs to concurrently access their own instructions and data → Sharing memory
 - Many programs to access devices → Sharing disks

Why We Study Operating Systems?

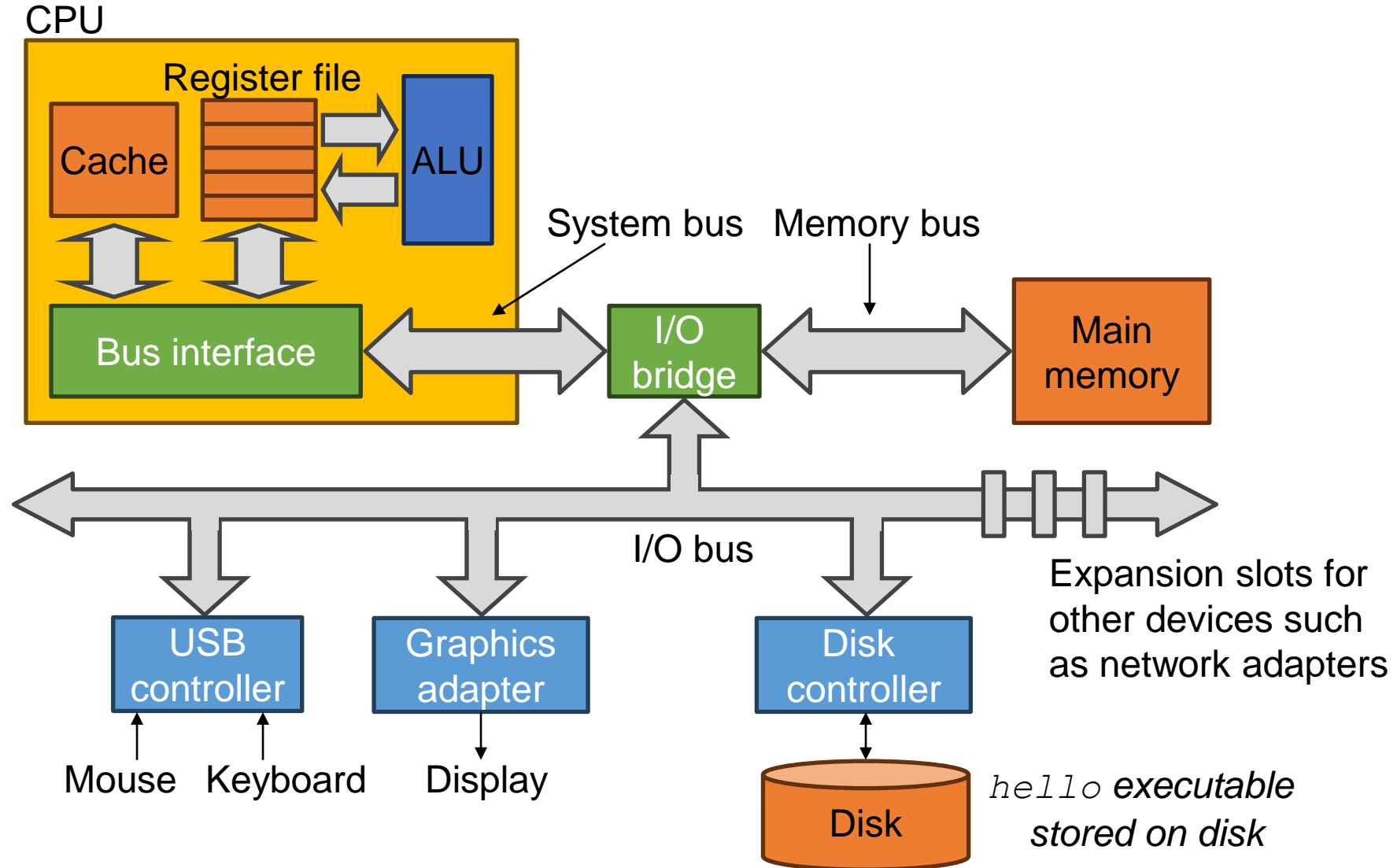
- Why study operating systems and how they work?
- Simply because, as **almost all code runs on top of an operating system**, knowledge of how operating systems work is **crucial to proper, efficient, effective, and secure programming**.
- Understanding the fundamentals of operating systems, how they derive computer hardware, and what they provide to applications is not only **essential to those who program them** but also **highly useful to those who write programs on them and use them**.

Agenda

- What Happens When a Program Runs
- What is an Operating System
- **Interrupt-Driven Program**
- Operating System Design Goals



Computer System Organization



Interrupt

- **Interrupt**: an asynchronous signal from hardware or software indicating the need for attention
 - Supported by H/W
 - Each type of interrupt is associated with a number (IRQ number)
 - Handled by interrupt handler
- Separate segments of code determine what action should be taken for each type of interrupt.
 - Table of interrupt handler: **interrupt vector**

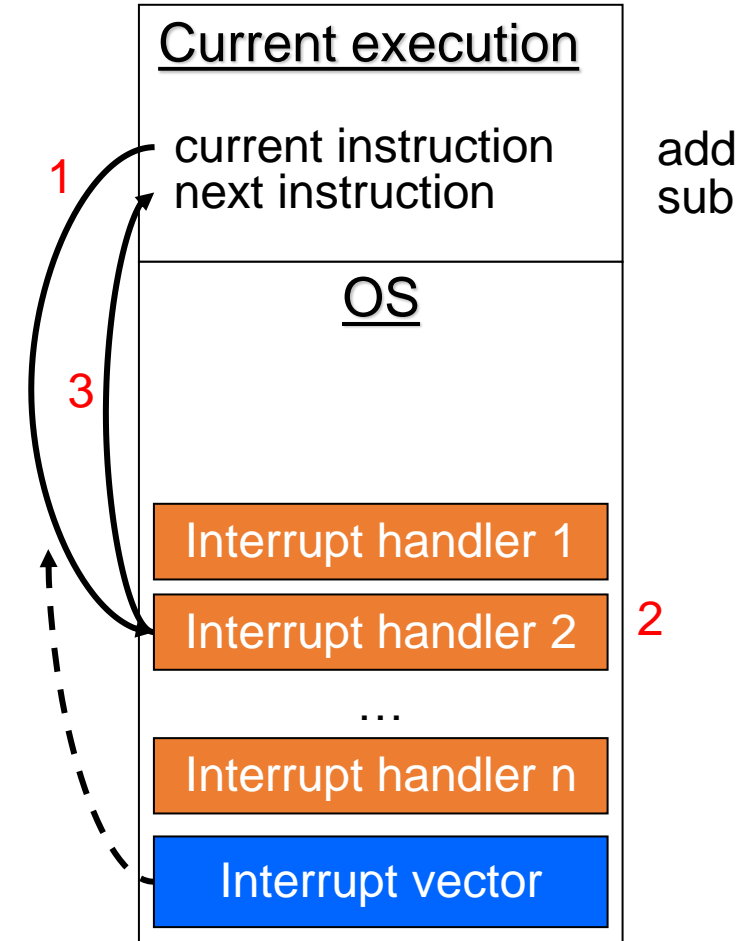
Interrupt

- A mechanism to process event
 - H/W interrupt
 - Sending signal to CPU
 - S/W interrupt
 - System call (=monitor call): request from program
 - I/O access, memory allocation, ...
 - Exception
 - Divide by zero, invalid memory access, I/O exception, ...

An operating system is an interrupt driven program

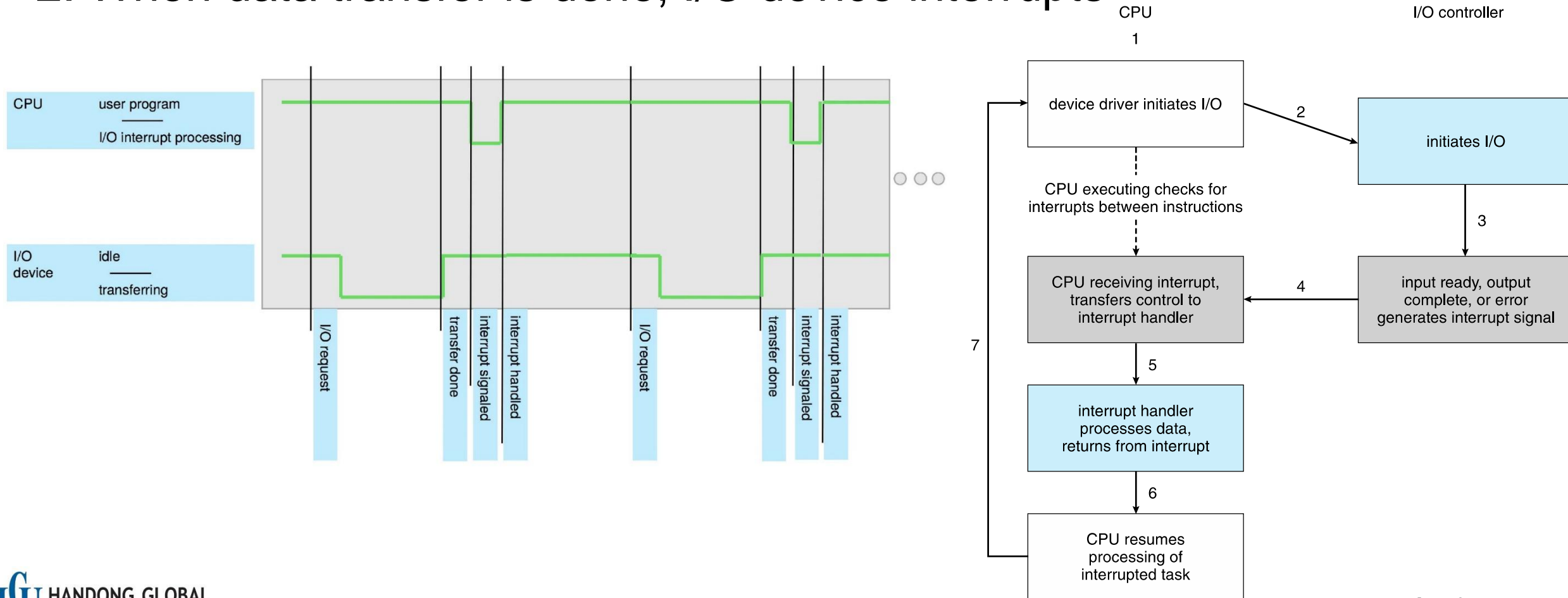
Interrupt Handling

1. The operating system saves the state of the CPU by storing the registers and the program counter and transfers execution to interrupt handler
 - Interrupt vector: table of interrupt handlers for each types interrupt
2. Interrupt is handled by corresponding handler.
3. Restore the state of the CPU and return to the interrupted program.



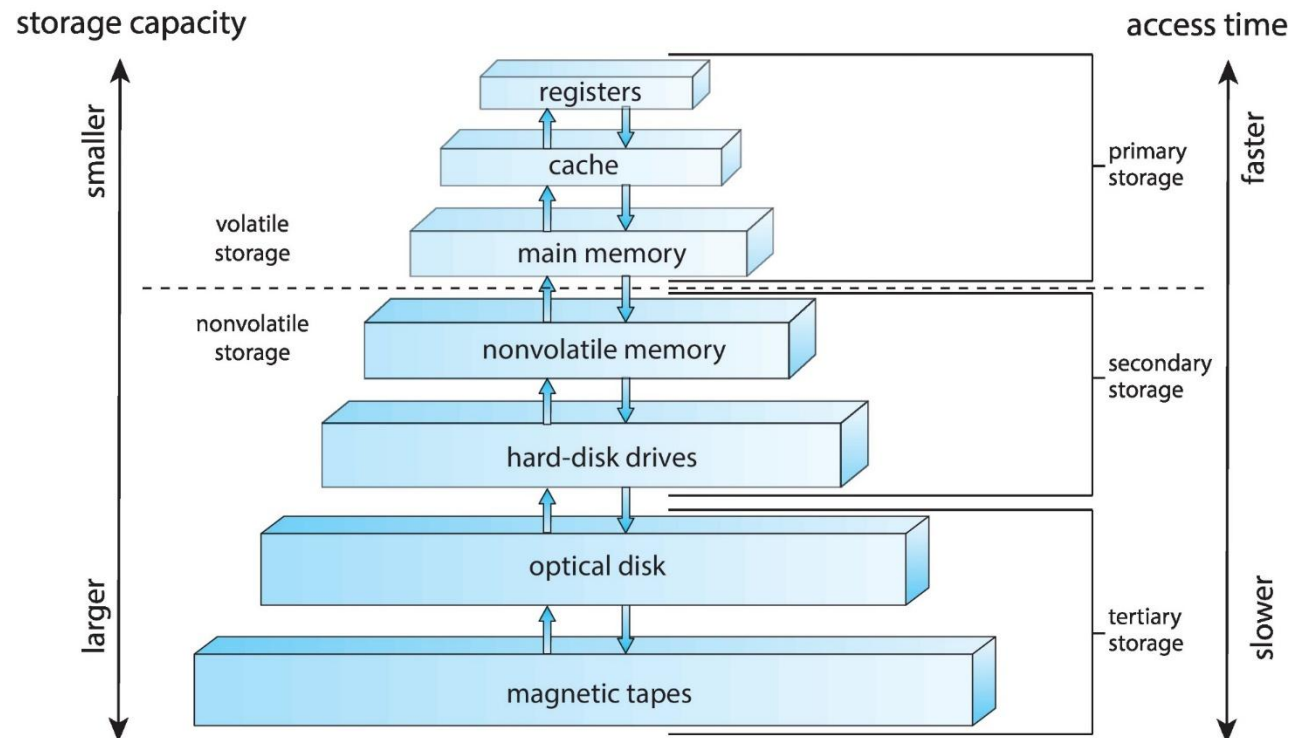
Interrupt Driven I/O

1. CPU sends request and continue current process or do another job.
2. When data transfer is done, I/O device interrupts



Storage Hierarchy

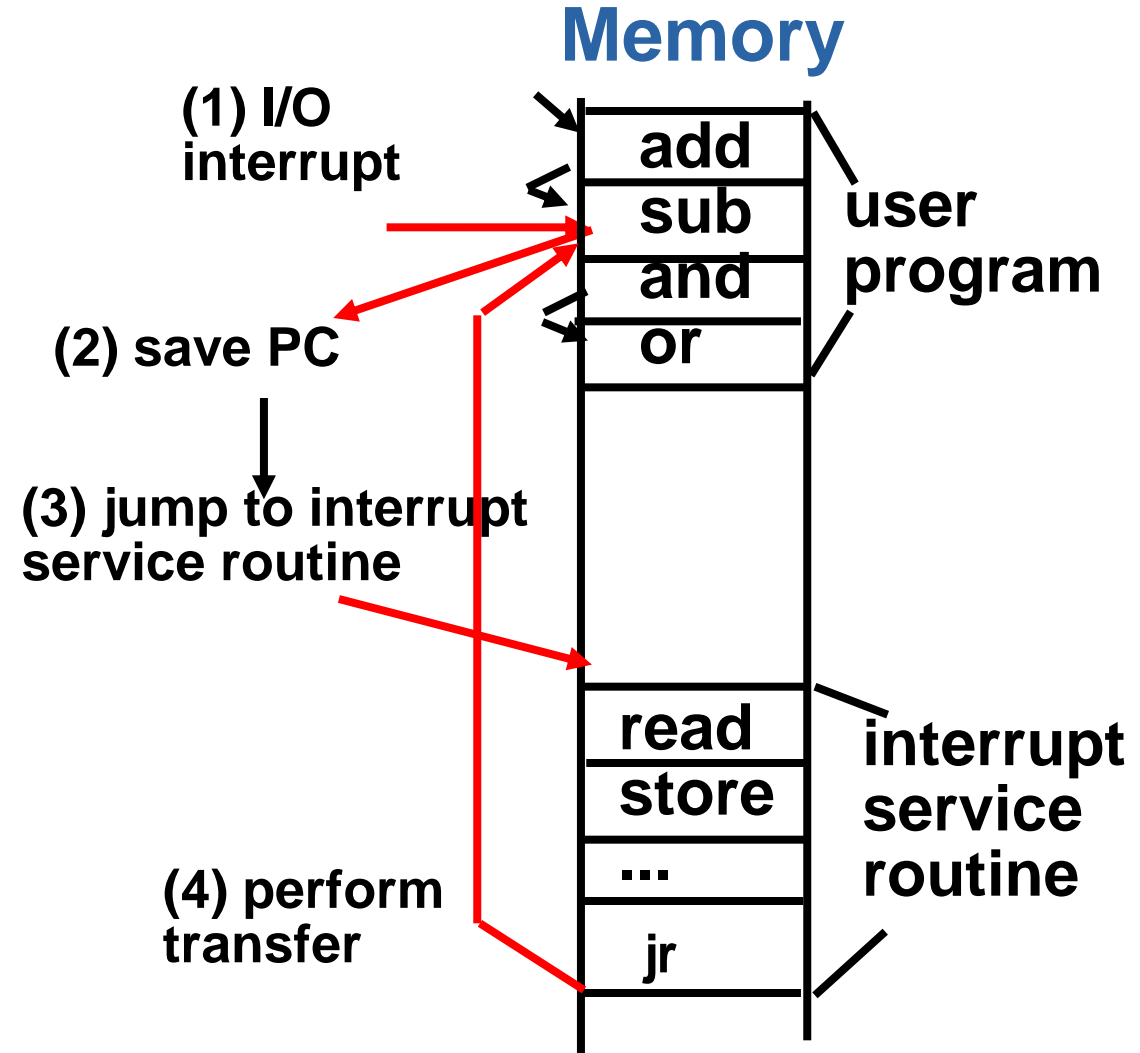
- Caching: copying information into faster storage system
 - Main memory can be viewed as a cache for secondary storage
- Device driver for each device controller to manage I/O
 - Provides uniform interface between controller and kernel



Interrupt Driven Data Transfer

- Advantage:
 - User program progress is only halted during actual transfer
- Disadvantage, special hardware is needed to:
 - Cause an interrupt (I/O device)
 - Detect an interrupt (processor)
 - Save the proper states to resume after the interrupt (processor)
 - When transferring large block of data, processors should be involved in transferring the data.

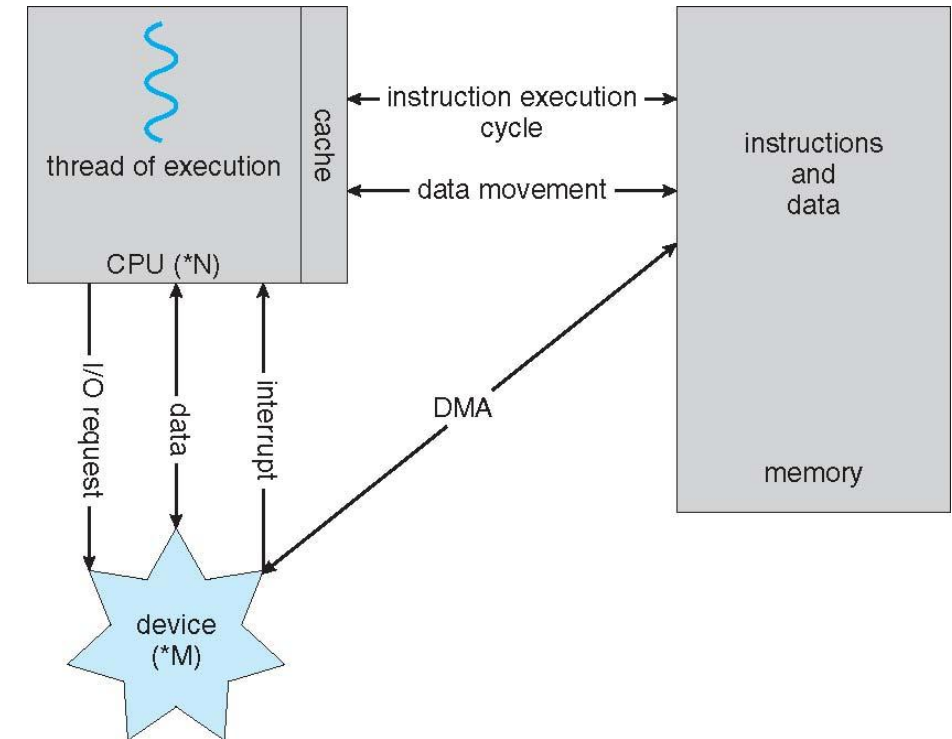
→ DMA is the solution



Direct Memory Access

■ DMA (Direct Memory Access)

- Device controller directly transfers large bulk of data into main memory
- CPU doesn't have to be involved in data transfer
- Appropriate for block transfer of high bandwidth I/O devices like hard disk



■ DMA Operation (interaction with OS)

1. The operating system sends a transfer request to the DMA controller
2. The DMA controller performs data transfer without involving the CPU
3. Once the transfer is complete, the DMA controller sends an interrupt to the

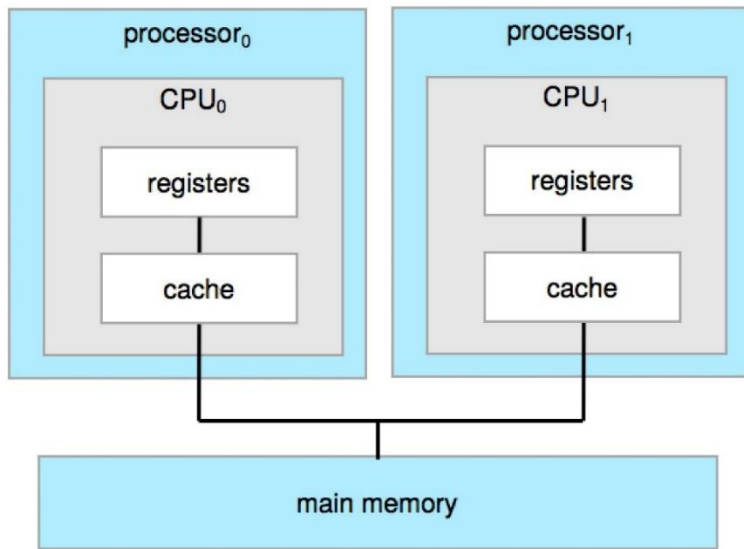
Agenda

- What Happens When a Program Runs
- What is an Operating System
- Interrupt-Driven Program
- Operating System Design

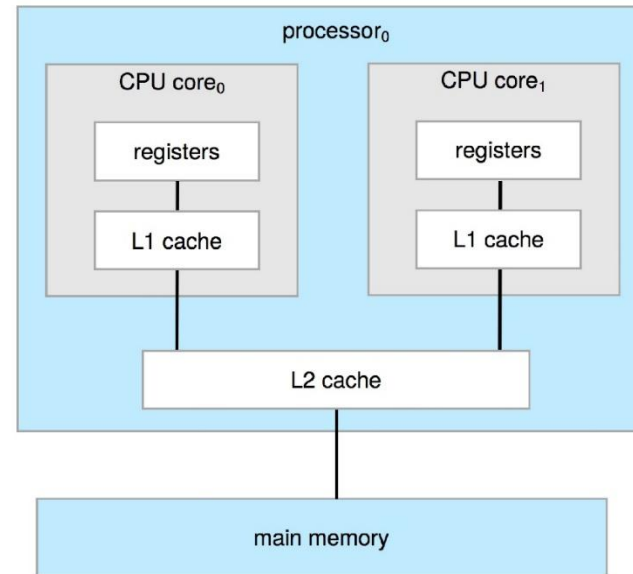


Computer-System Architecture

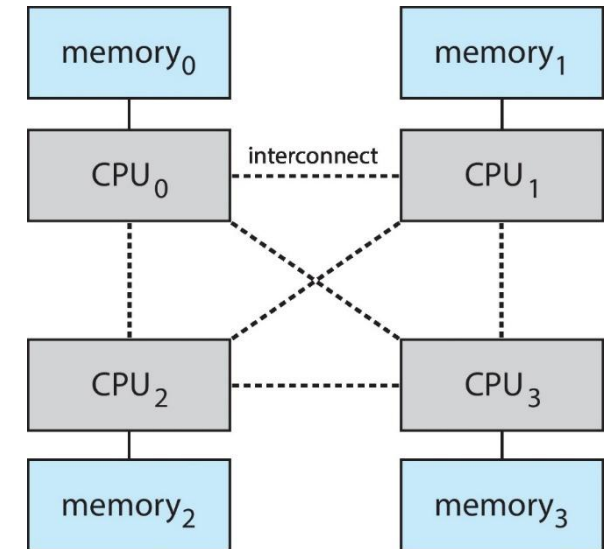
- Some operating systems consider various computer architecture.
 - On modern computers, **multiprocessor** systems now dominate the landscape of computing



<Symmetric multiprocessing architecture>



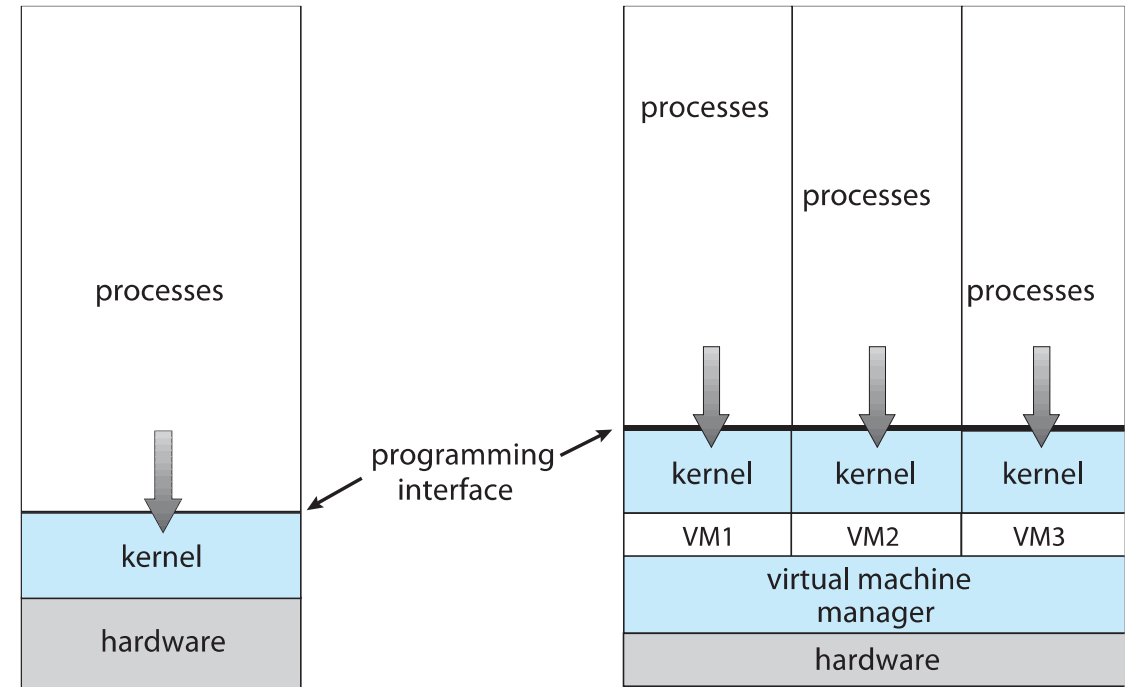
<A dual-core architecture>



<Non-uniform memory access>

Computing Environments

- There are different types of computing environments.
 - Mobile Computing
 - Client-Server Computing
 - Virtualization
 - Cloud Computing
 - Real-Time Embedded Systems



<Traditional OS structure>

<Virtualization>

Design Goals

- Let's explore the design goals of an operating system!
 - Please refer to OSTEP

Core Components of Operating Systems

- Process management
 - Process: program in execution
- Memory management
 - It determines what is in memory and when
- File System management
 - File and Directory
- Mass-Storage management
 - Free space management, storage allocation, disk scheduling
- Cache management
- I/O system management