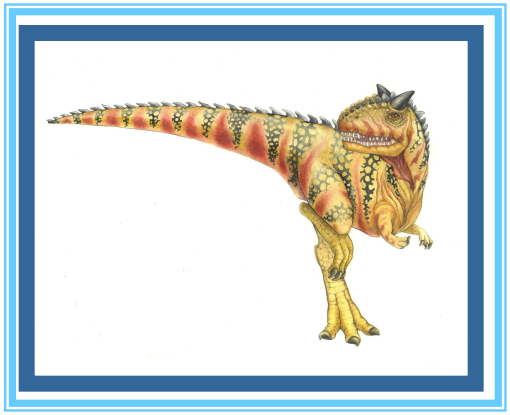
פרק 1 - מבוא



Revised and updated by David Sarne

מבוא לקורס

- מערכות הפעלה – מרכיב עיקרי וחיוני בכל מערכת מחשב
- הקורס ידון ב:
  - מה הן מערכות הפעלה?
  - מה הן יודעות לעשות?
  - כיצד הן בנויות (designs and structures)
  - מרכיבים עיקריים של מערכות הפעלה:
    - Processes, Threads, CPU-scheduling, Synchronization, Deadlocks, Memory Management, Virtual Memory, File system interface
- ספר הקורס:
  - Silberschatz, Galvin and Gagne, Operating System Concepts – 9th Edition

Main Themes

Topic	#
Introduction	1
Operating System Structures	2
Processes	3
Threads	4
CPU Scheduling	5
Process Synchronization	6
Deadlocks	7
Memory Management	8
Virtual Memory	9
File-System Interface	10
File-System Implementation	11
Windows / Unix	12

מטרת השיעור (היום)

- להכיר את המרכיבים העיקריים של מערכות הפעלה
- חזרה על מבנה מחשב

The Von Neumann Architecture

- הכרת מבנה המחשב (מצגת שקפים)

ממבנה מחשב למערכת מרובת תהליכים

- מספר תהליכים שרצים במערכת
- תקשורת בין התהליכים
- התקני קלט ופלט

- אנלוגית המטבח – מה נמצא במטבח?
  - מתכונים
  - מוצרי מזון
  - כלים ומכשירים לעיבוד מזון

נהלים

- הרצאה שבועית, 3 שעות
- מבנה הציון:
  - 75% בחינה (ובבחינה שאלה של 20 נקודות על נושאי התרגול)
  - 25% - תרגילים
  - בונוסים
- מעבר הקורס – קיום שני התנאים:
  - ציון בחינה של לפחות 60
  - ממוצע תרגילים של לפחות 60
- אתר הקורס - במודל

מהי מערכת הפעלה?

- הגדרה:
  - תכנית אשר משמשת כחוצץ (intermediary) בין המשתמש של מערכת מחשב והחומרה של מערכת המחשב
- מטרות מערכת ההפעלה:
  - הרצת תכניות המשתמש (user programs) בצורה קלה
  - הפיכת השימוש במערכת המחשב לנוח יותר
  - שימוש יעיל יותר (efficient) בחומרת המחשב

Example – MS-Paint over Windows

- Assume we are using MS-Paint over Windows - when do we need to access the OS?
  - Loading the application / terminating the application
  - Interrupts Management
  - Memory allocation / management (e.g., paging)
  - Access to IO devices – keyboard, mouse, printer, monitor
  - CPU allocation
  - Copy / Paste (inter-process communication)

Design & Goals - מערכת הפעלה

- לכל מערכת הפעלה יש מטרות (goals) ועיצוב (design) שונים:
  - Mainframe – ניצול מקסימלי של משאבי החומרה
  - PC – תמיכה מקסימלית בהרצת תכניות של המשתמש
  - Handheld – ממשק נוח להרצת אפליקציות; ביצועים טובים פר יחידת ניצול של הסוללה

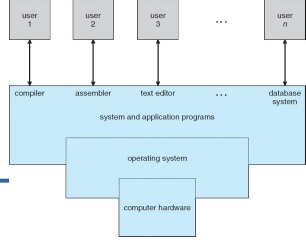
performance,
resource utilization

ease of use

Mainframe, PC, Handheld

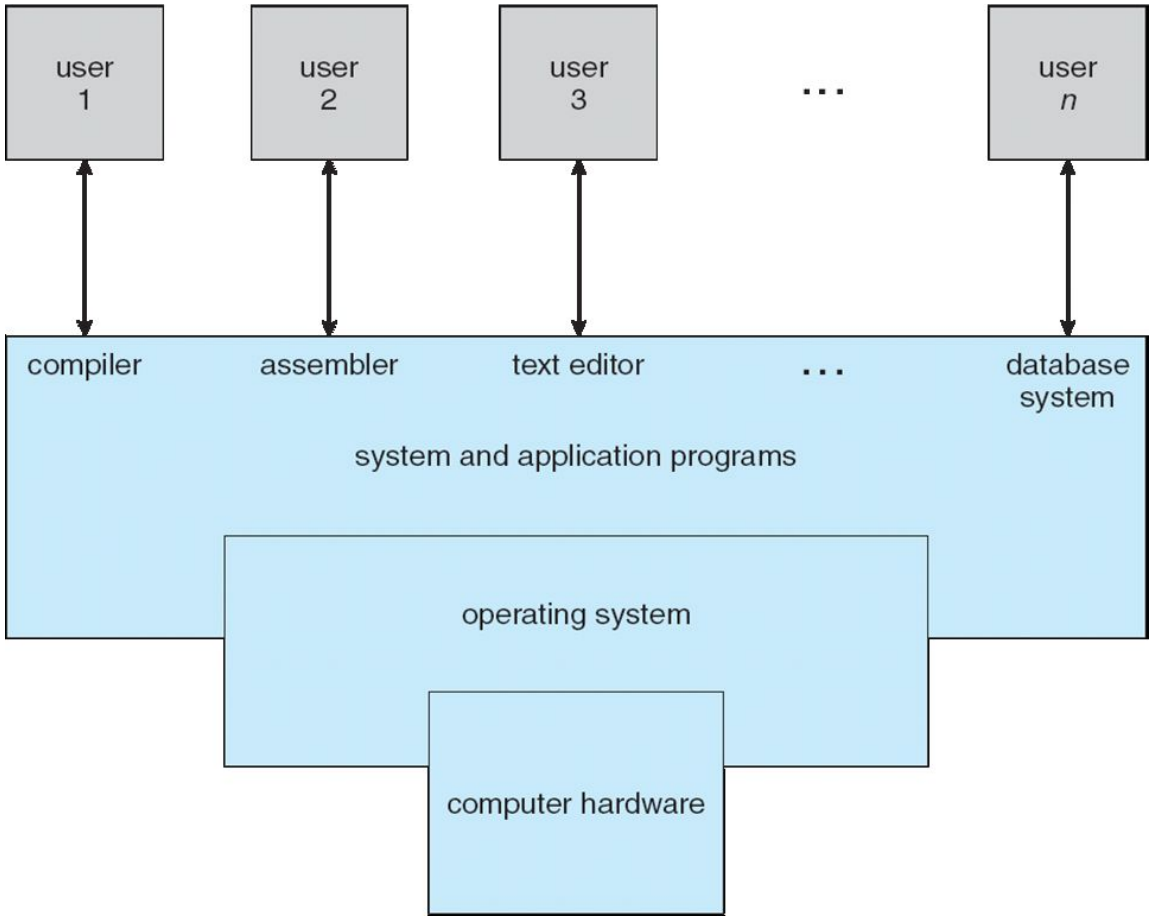
- **Supercomputer** - computer at the frontline of current processing capacity, particularly speed of calculation
- **Mainframe** – powerful computers used mainly by large organizations for critical applications (the term originally referred to the large cabinets that housed the central processing unit and main memory of early computers. Later the term was used to distinguish high-end commercial machines from less powerful units)
- **Personal Computer (PC)** - any general-purpose computer whose size, capabilities, and original sales price make it useful for individuals (and which is intended to be operated directly by an end-user with no intervening computer operator)
- **Handheld** - pocket-sized computing device, typically having a display screen with touch input and/or a miniature keyboard.
- Of course, one generation's "supercomputer" is the next generation's "mainframe"

Computer System Structure



- ניתן לחלק את מערכת המחשב לארבעה מרכיבים:
- חומרה (hardware) – מספקת את משאבי המחשוב הבסיסיים:
  - CPU, memory, I/O devices, file storage space
- מערכת ההפעלה:
  - שולטת על ומתאמת את השימוש במשאבי החומרה בין התהליכים והמשתמשים השונים
- אפליקציות (Application programs) – מגדירים כיצד מבוצע השימוש במשאבי המערכת על מנת לפתור בעיות חישוביות של המשתמשים:
  - Word processors, compilers, web browsers, database systems, video games
- משתמשים:
  - People, other computers

Four Components of a Computer System



מהו טבעה של מערכת ההפעלה?

- מערכת ההפעלה היא **resource allocator:**
  - מנהלת את כל המשאבים (resources) – בדומה לממשלה
  - מחליטה במצבים של קונפליקט (conflicting requests) על מנת שהשימוש במשאבים יהיה יעיל והוגן (efficient and fair)
- מערכת ההפעלה היא **control program:**
  - שולטת על ההרצה (execution) של התוכניות השונות על מנת למנוע שגיאות (errors) ושימוש מוטעה/חורג (improper) במערכת המחשב

חשוב במיוחד כאשר ישנם מספר משתמשים המחוברים לאותו mainfrme או microcomp uter

הגדרת/תכולת מערכת ההפעלה

- אין הגדרה אחת אוניברסלית
- יש שסוברים:
  - Everything a vendor ships when you order an "operating system"
- התוכנית/תהליך שרץ/ה כל הזמן במערכת המחשב נקראת kernel. כל שאר התהליכים שרצים מקוטלגים כ- system programs (תכניות שהגיעו יחד עם מערכת ההפעלה) או application program

The matter of what constitutes an operating system is important!
In 1998 this was the essence of a suit filed by the United State Department of Justice against Microsoft
(even though today most mobile OS include much functionality)

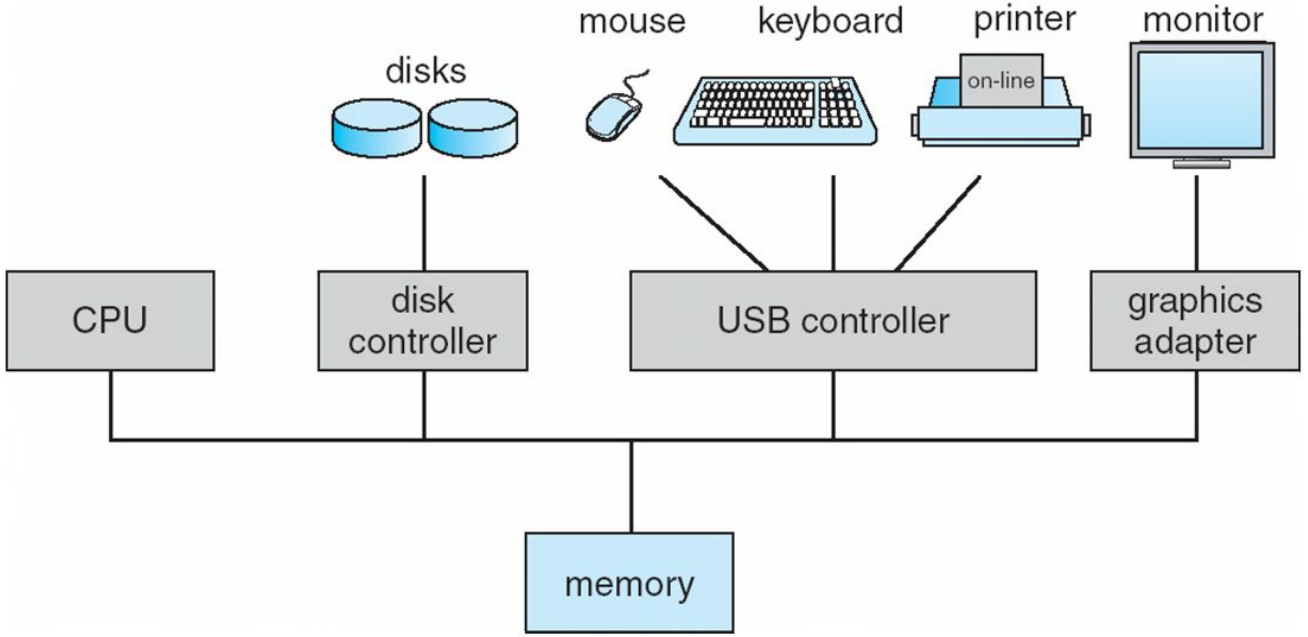
Computer Startup

- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system (CPU registers, device controllers, memory contents, etc.)
  - Loads operating system kernel and starts execution

OS initializes, starts its first process and waits for an event…

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles (through memory controller)



Device Controller

- כל device controller אחראי על סוג מסוים של devices
- המידע מ/אל ה- device מנוהל באמצעות local buffer:
  - המעבד מעביר מידע מ/אל הזיכרון הראשי (main memory) אל/מ ה- local buffers
  - גם ה- I/O מ/אל ה- device עצמו מועבר מ/אל ה- local buffer של ה- controller
- ה- device controller מודיע למעבד כאשר הוא מסיים את הטיפול ב- I/O באמצעות interrupt

Device controller

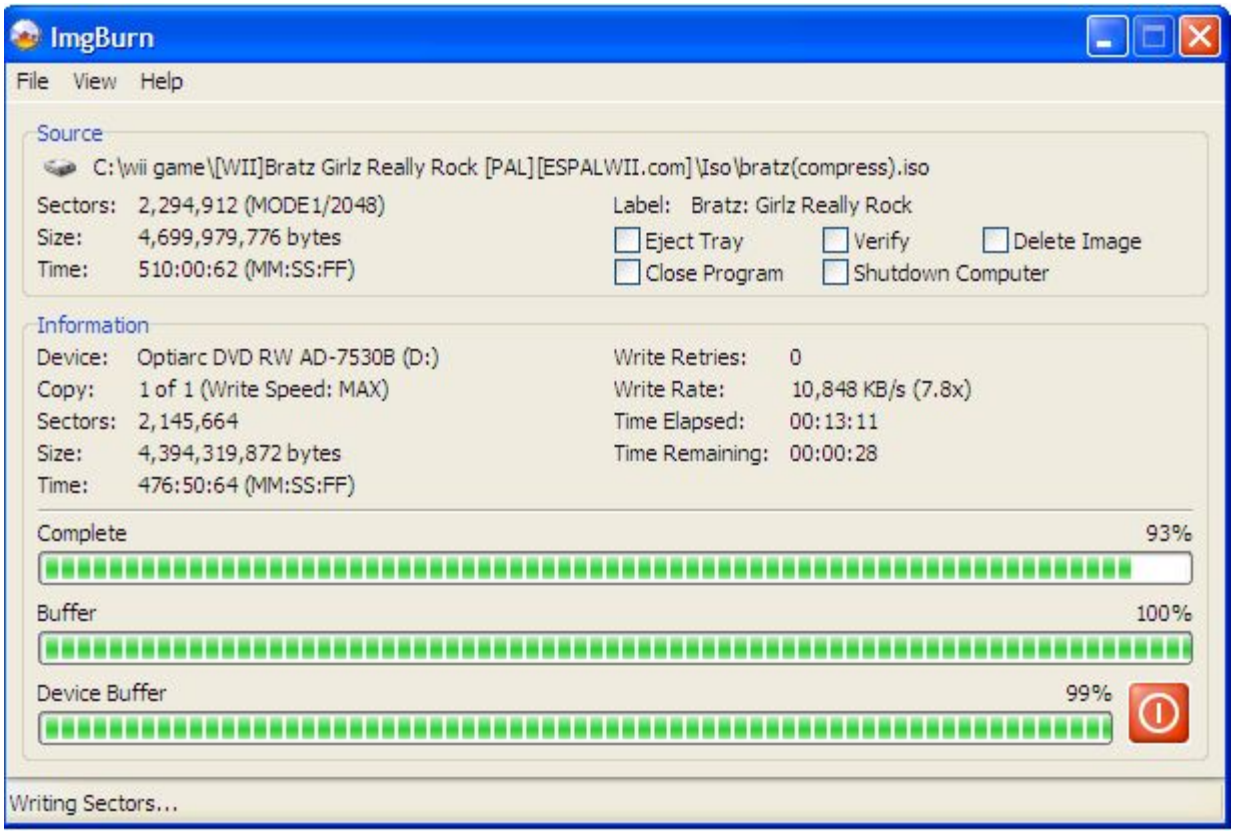
From Wikipedia, the free encyclopedia

A **device controller** is a part of a computer system that makes sense of the signals going to, and coming from the CPU processor. There are many device controllers in a computer system. Any device connected to the computer is connected by a plug and socket, and the socket is connected to a device controller. Device controllers use binary and digital codes. Each device controller has a local buffer and a command register.It communicates with the cpu by INTERRUPTS. Device Controller play an important role in order to operate that device.Its just like a bridge between device and operating system.

This short article can be made longer. You can help Wikipedia by adding to it.

SCSI device controller

Many Buffers in a Computer System



ארכיטקטורת מערכת מחשב

- ברוב המערכות (מ- mobile ועד mainframe ו- super computer):
  - מעבד אחד או יותר מסוג general purpose
  - מעבדים נוספים מסוג special purpose – למשל digital signal processor (DSP) או graphic processing unit (GPU)
- מערכות Multiprocessors
  - שני מעבדים ויותר עם תקשורת ביניהם, החולקים את אותו ה- bus ולעיתים גם את ה- clock והזיכרון (memory)
  - הרבה פעמים נקראים parallel systems, tightly-coupled systems
  - יתרונות:

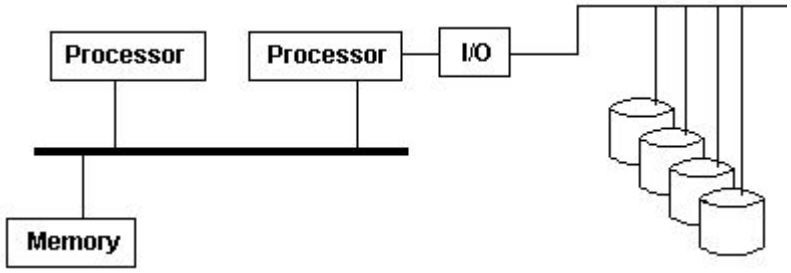
Increased throughput, Economy of scale, Increased reliability – graceful degradation or fault tolerance

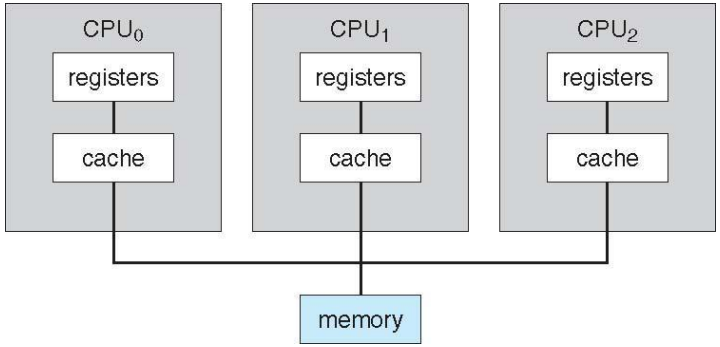
Multiprocessors systems

- Two types of Multiprocessing:

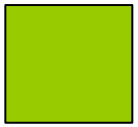
Key role – the scheduler

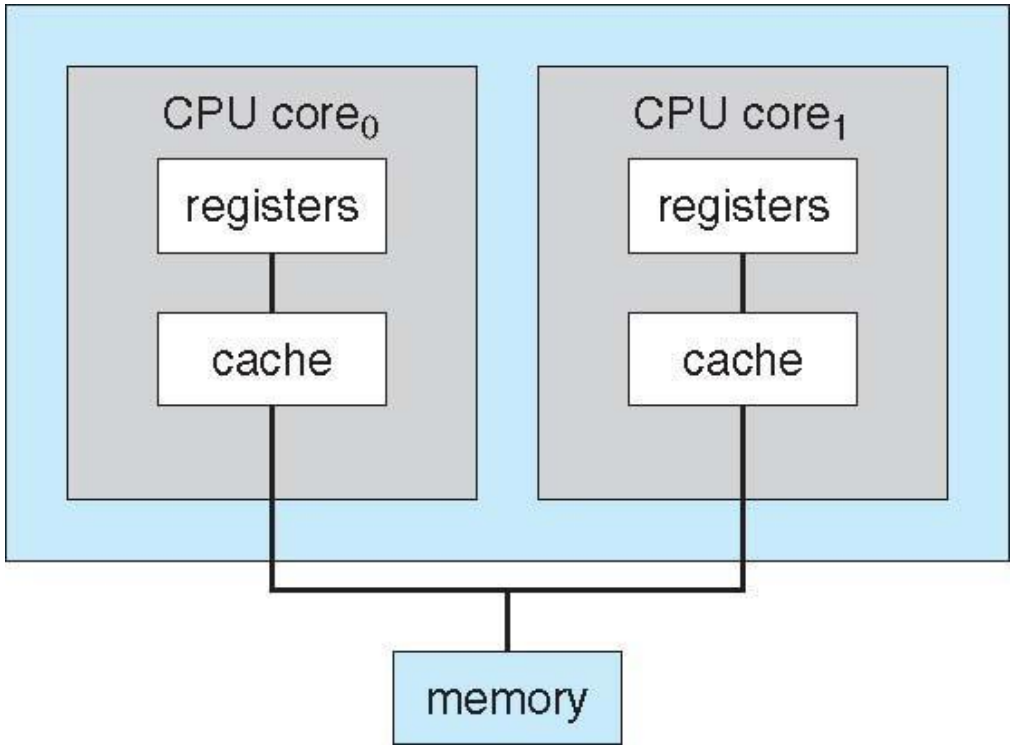
1. Asymmetric Multiprocessing - assigns certain tasks only to certain processors.
    1. In particular, only one processor may be responsible for handling all of the interrupts in the system or perhaps even performing all of the I/O in the system
2. Symmetric Multiprocessing - treats all of the processing elements in the system identically





A Dual-Core Design



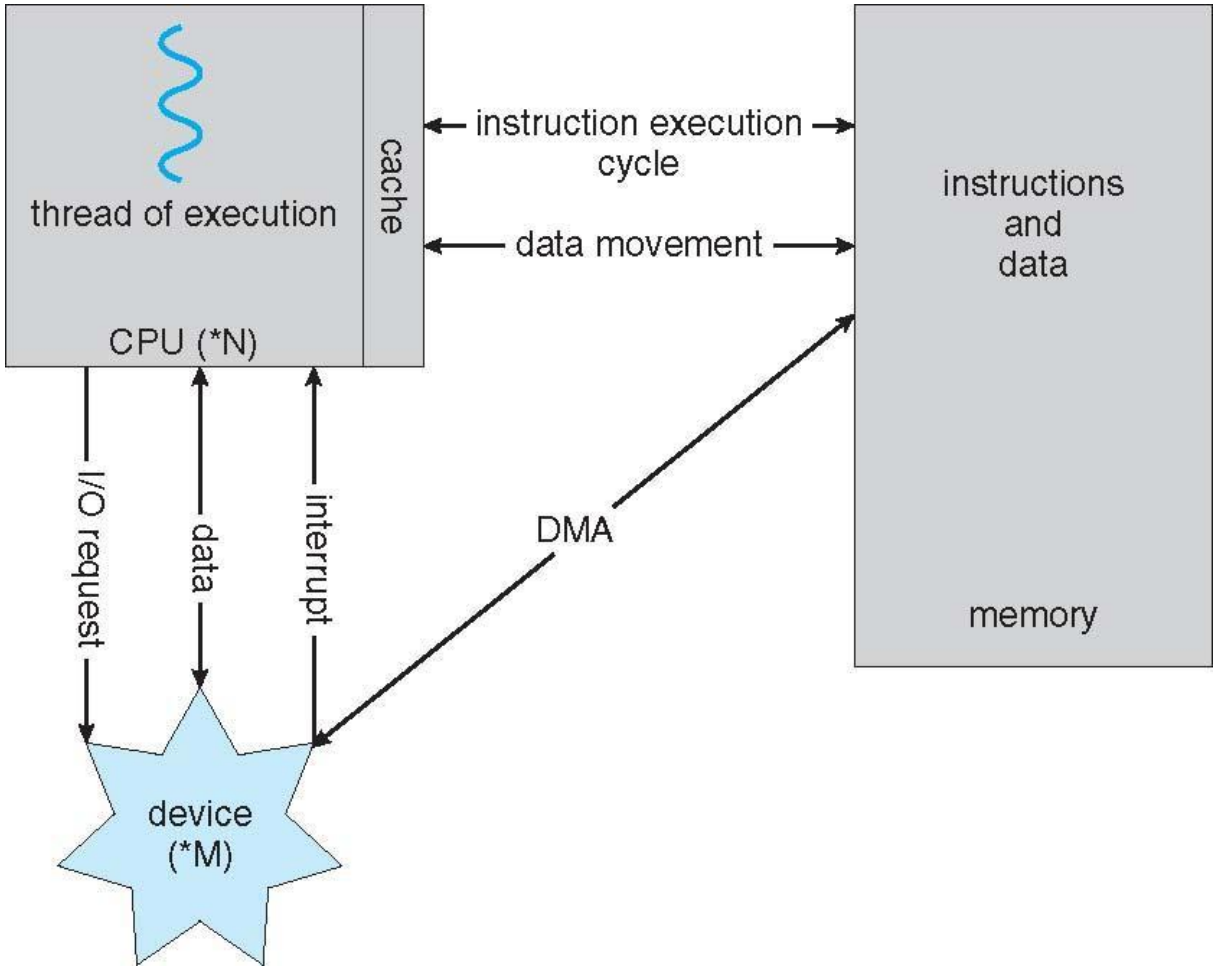


multiCore – use less electricity;
faster communication between cores

Clustered Systems

- Like multiprocessor systems, but **multiple systems working together**
  - Usually sharing storage via a storage-area network (SAN)
  - Provides a high-availability service which survives failures
    - Asymmetric clustering has one machine in hot-standby mode
    - Symmetric clustering has multiple nodes running applications, monitoring each other
  - Some clusters are for high-performance computing (HPC)
    - Applications must be written to use parallelization

How a Modern Computer Works



Interrupts

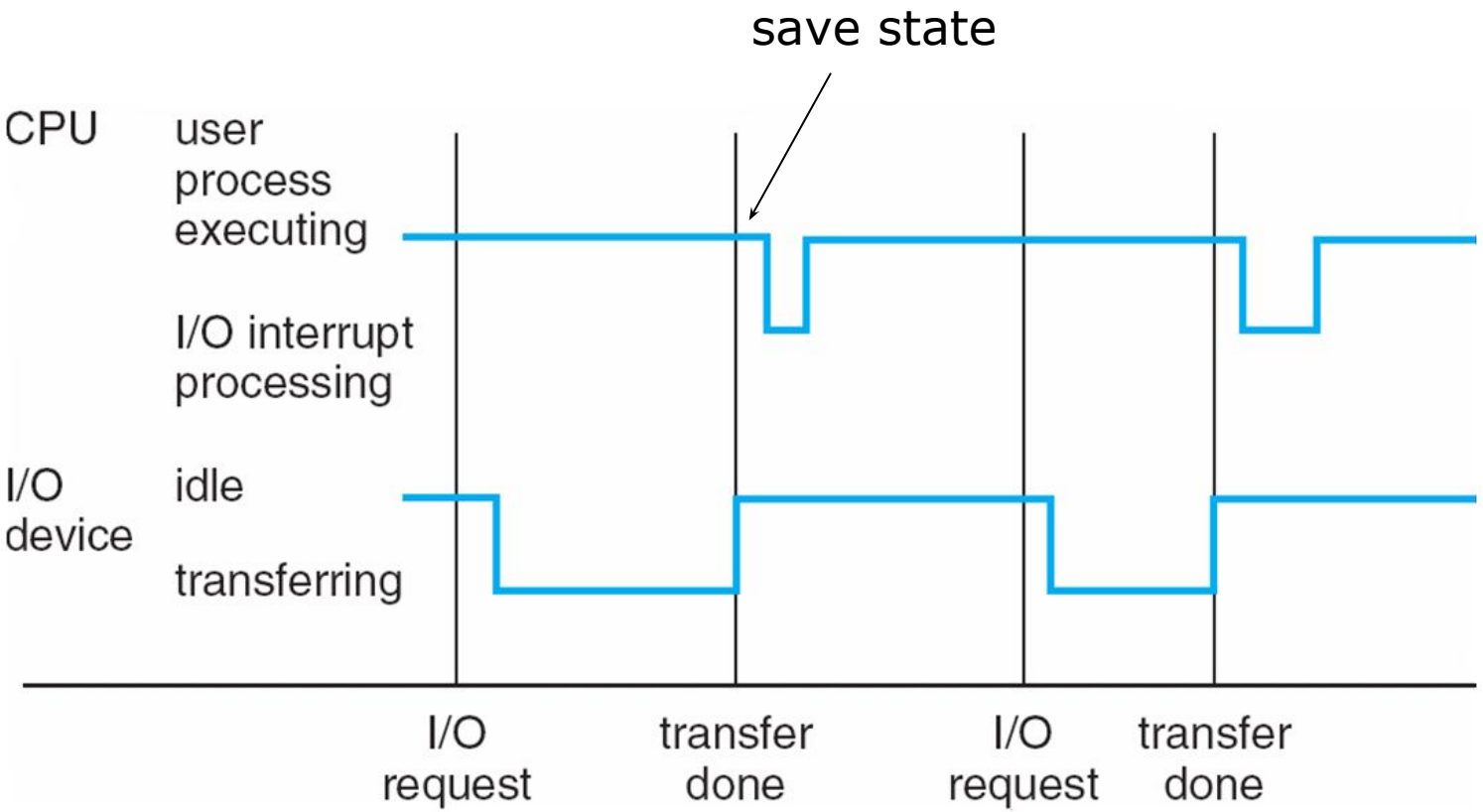
- ה- interrupt מעביר את השליטה (במעבד) ל- interrupt service routine
  - נוצר על-ידי device
  - מחייב שמירה של הכתובת של ה- interrupted instruction ושל תוכן הרגיסטרים
  - לרוב בעת הטיפול ב- interrupt המערכת תמנע (disable) שליחת interrupts נוספים על-מנת לא לאבד interrupts
- ה- ***trap*** **הוא** software-generated interrupt אשר נגרם על-ידי שגיאה (error) או בעקבות בקשה של תוכנית של המשתמש
- An operating system is **interrupt driven**

Interrupt Handling

- Determines which type of interrupt has occurred:
  - **polling**
  - **vectored** interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt

Ideally, we would have used a generic code for analyzing the interrupt information and deciding what code to run, however speed is critical here…

Interrupt Timeline



I/O Structure

- After I/O starts, control returns to user program only upon I/O completion

Synchronous

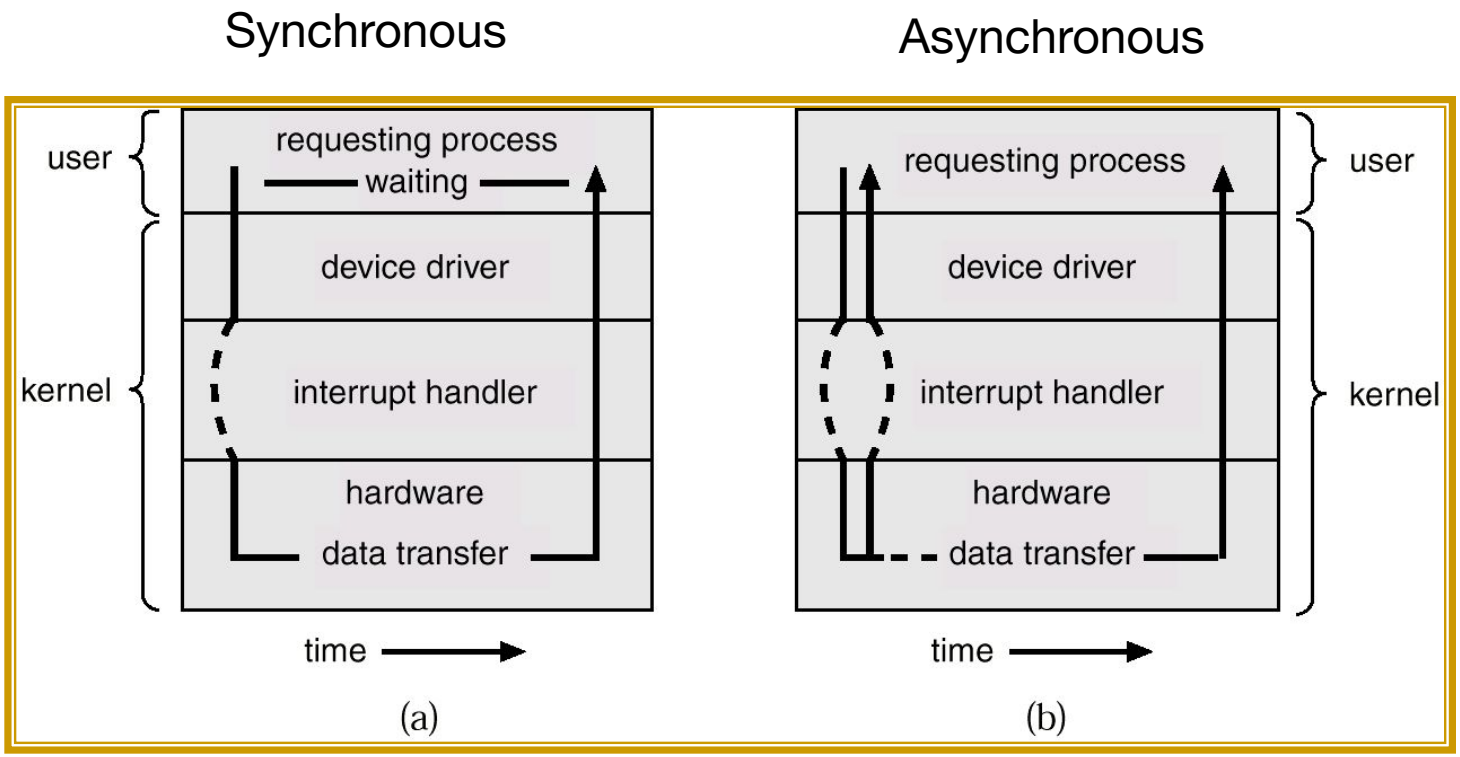
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access)
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing

- After I/O starts, control returns to user program without waiting for I/O completion

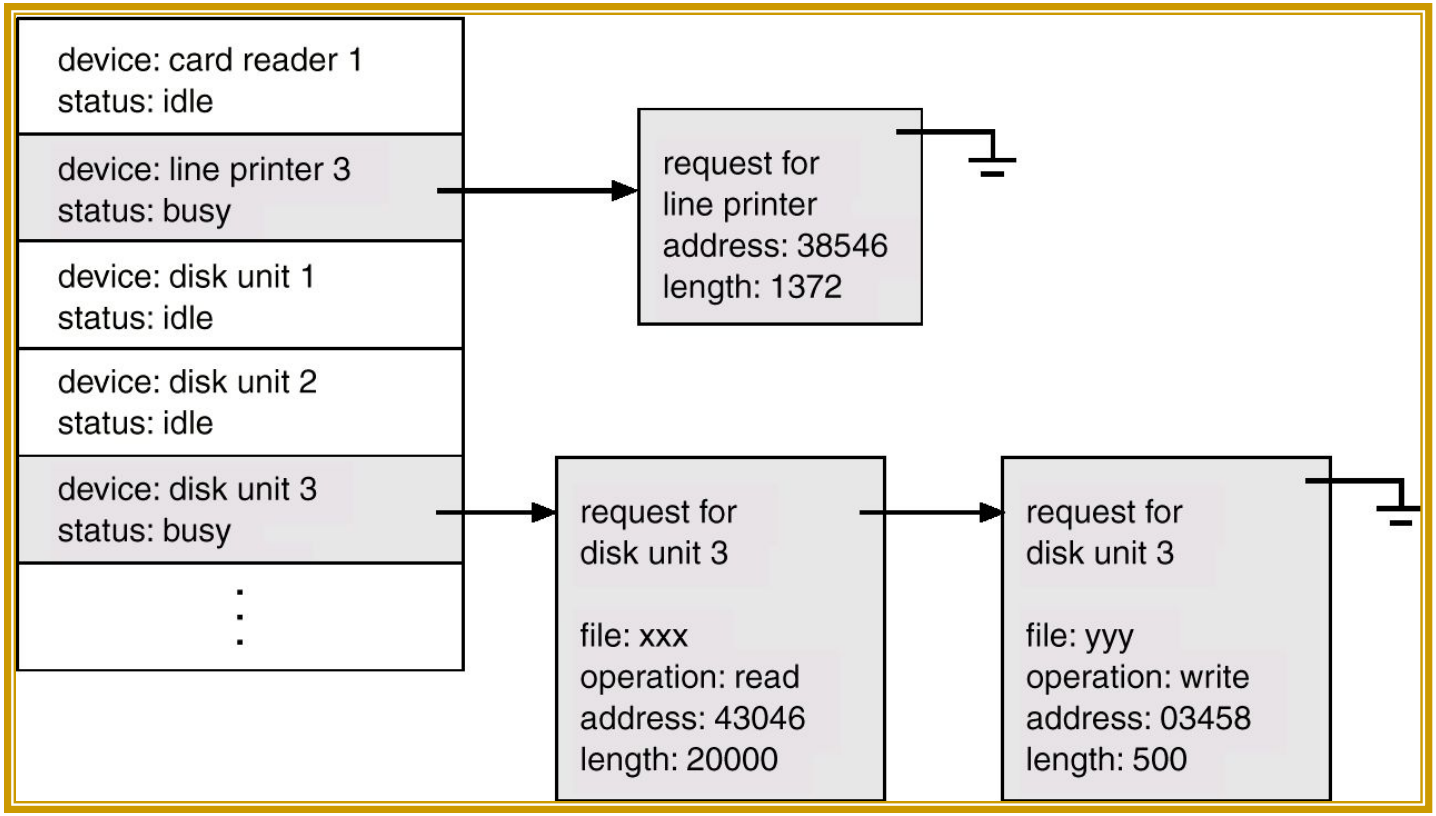
Asynchronous

  - **System call** – request to the operating system to allow user to wait for I/O completion
  - **Device-status table** contains entry for each I/O device indicating its type, address, and state
  - Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt

Two I/O Methods



Device-Status Table



Memory Management

- חשיבות הזיכרון:
  - ה- instructions חייבות להיות בזיכרון על-מנת שנוכל לבצע (להריץ) אותן
  - ה- data חייב להיות בזיכרון על-מנת שנוכל לעבד אותו
  - לאחר עיבודו חוזר שוב לזיכרון
- "ניהול הזיכרון" עוסק בהחלטה על מה יהיה/ישהה בזיכרון בכל רגע ורגע:
  - מרכיב קריטי לניצולת (utilization) של המעבד ובעל השפעה מכרעת על זמן התגובה למשתמש
- תפקידי מערכת ההפעלה בכל הקשור לניהול זיכרון:
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

Mass-Storage Management

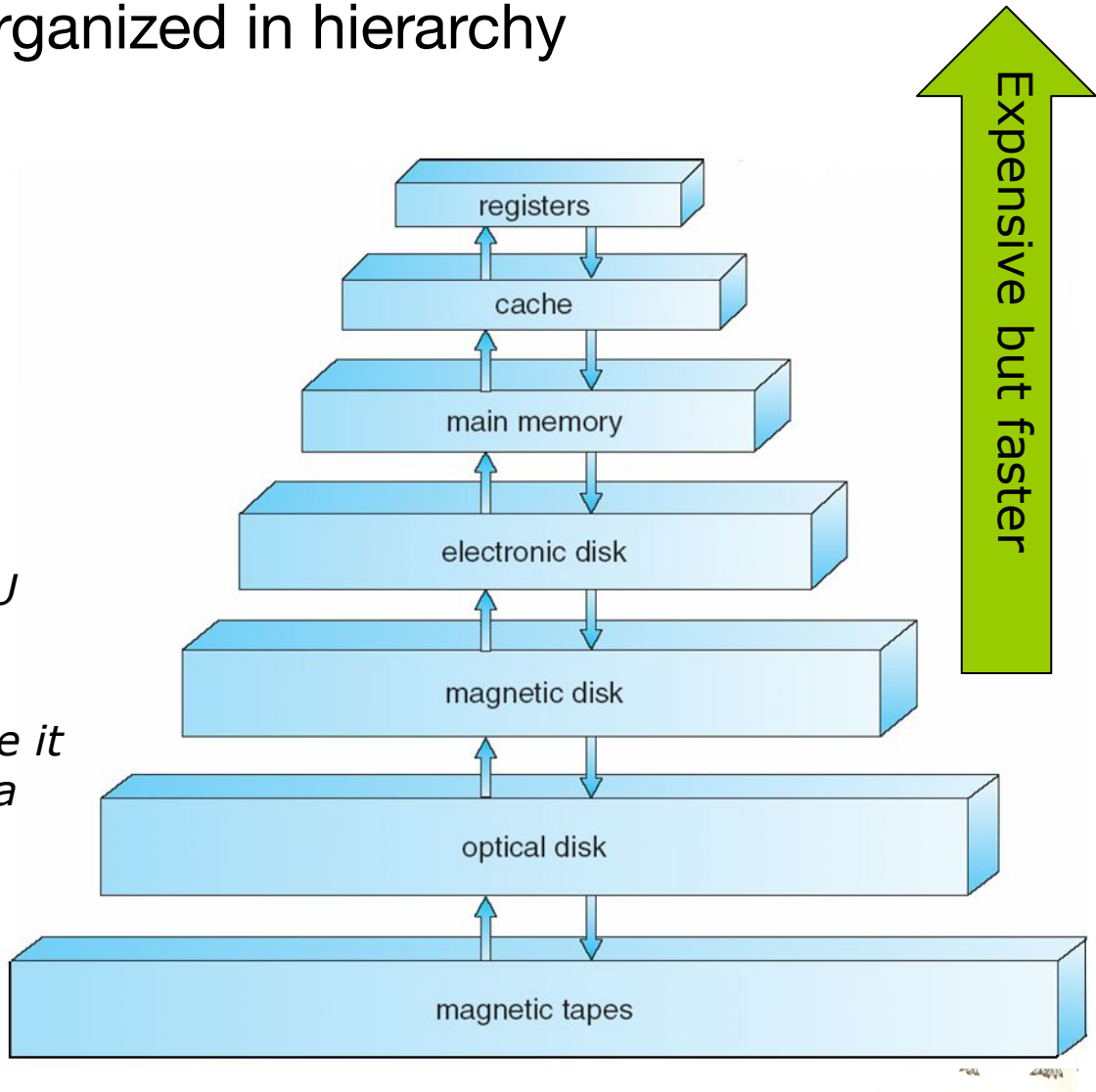
- Main memory – only large storage media that the CPU can access directly
- Why using disks?
  - Store data that does not fit in main memory
  - Store data that must be kept for a “long” period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Free-space management
  - Storage allocation
  - Disk scheduling

Mass Storage Management (2)

- Some storage need not be fast
  - Includes optical storage, magnetic tape
  - Not critical to the computer performance but still must be managed
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

Storage Hierarchy

- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility



It takes some time (several CPU cycles) to read/write to main memory – in the meantime the processor needs to stall because it doesn't have the necessary data

Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and dirs
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Caching

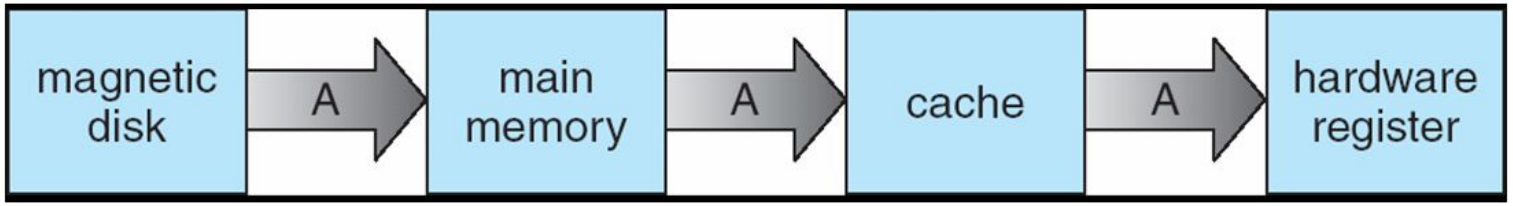
- **Caching** – copying information into faster storage system; main memory can be viewed as a last *cache* for secondary storage
- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Performance of Various Levels of Storage

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy



Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache

- Distributed environment situation even more complex
  - Several copies of a datum can exist

Direct Memory Access Structure

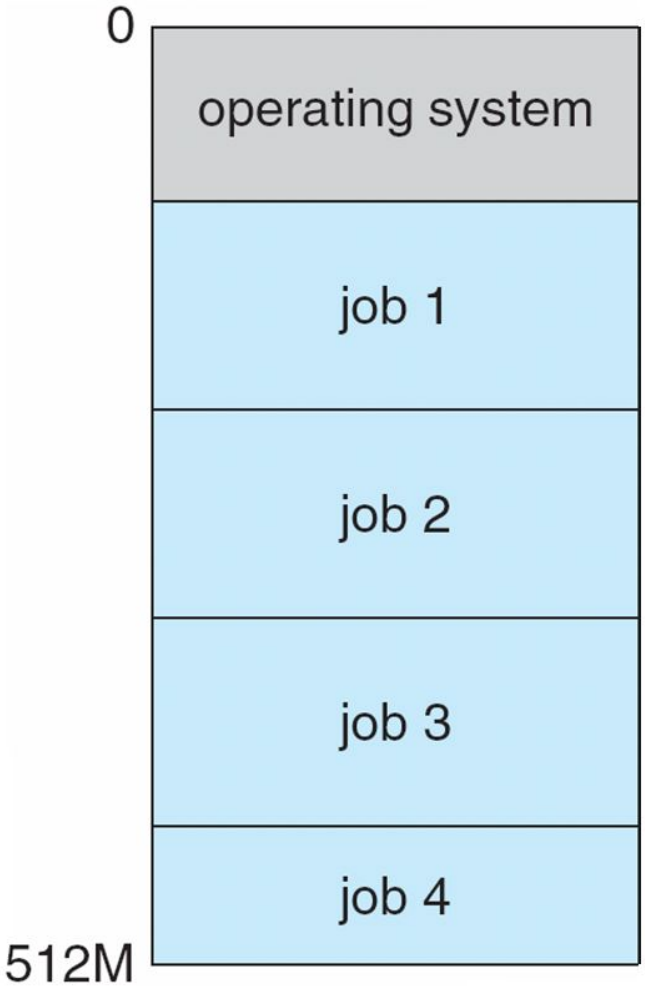
- Used for high-speed I/O devices able to transmit information at close to memory speeds
  - Good example: tape, disk
  - Bad example: keyboard
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention
- Only one interrupt is generated per block, rather than the one interrupt per byte

Operating System Structure

- **Multiprogramming** needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via **job scheduling**
  - When it has to wait (for I/O for example), OS switches to another job
    - Unlike sitting idle in a non-multiprogrammed system
  - The idea is common in other life situations (e.g., lawyers)

as long as at least one job needs to execute, the CPU is never idle…

Memory Layout for Multiprogrammed System



Operating System Structure (Cont.)

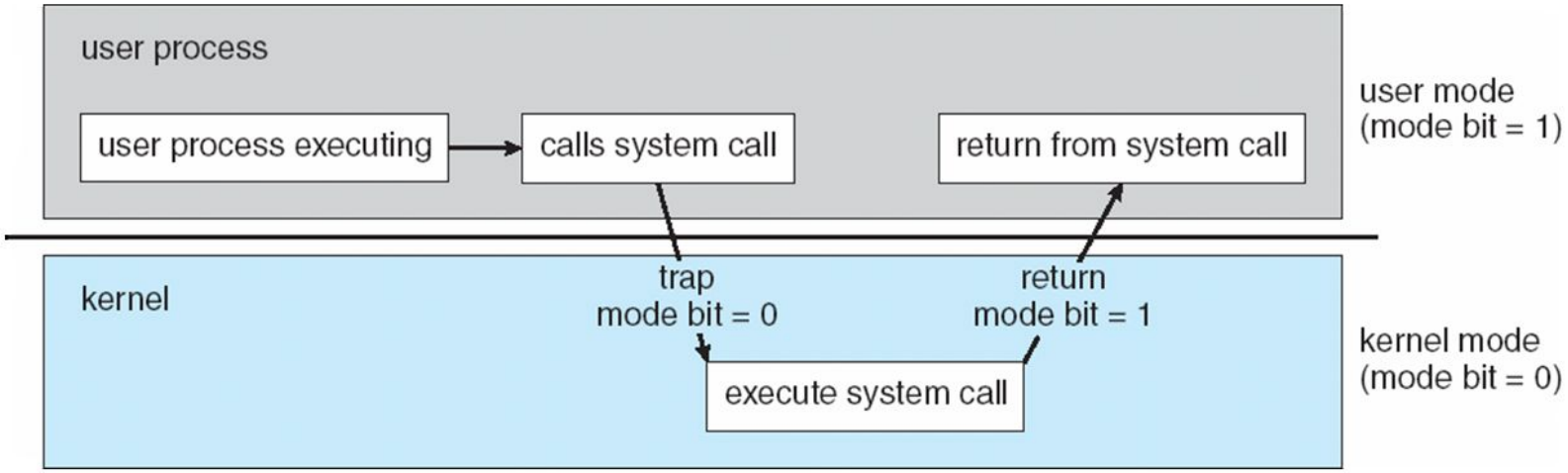
- **Timesharing (multitasking)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory □**process**
  - If several jobs ready to run at the same time □ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Operating-System Operations

- Handle flow problems:
  - Software error or request creates **exception** or **trap**
    - Division by zero, request for operating system service
  - Other process problems include infinite loop, processes modifying each other or the operating system
  - Example: in MS-Dos, originally written for Intel 8088:
    - a user program can wipe out the operating system by writing over it with data

Transition from User to Kernel Mode

- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user



Example

- Which of the following instructions should be privileged?
  - b. Read the clock.
  - c. Clear memory.
  - d. Issue a trap instruction.
  - e. Turn off interrupts.
  - f. Modify entries in device-status table.
  - g. Switch from user to kernel mode.
  - h. Access I/O device.

Example

- Which of the following instructions should be privileged?

  b. Read the clock.

  c. Clear memory.

  d. Issue a trap instruction.

  e. Turn off interrupts.

  f. Modify entries in device-status table.

  g. Switch from user to kernel mode.

  h. Access I/O device.

Process Management

- Process and Program:
  - A **process** is a program in execution
  - **Program** is a *passive entity*, process is an *active entity*.
- התהליך (process) צריך לקבל משאבים על מנת להשלים את המשימה לטובתה נוצר
  - CPU, memory, I/O, files (received upon creation and along execution)
  - Initialization data (e.g., a process for presenting the status of a file)
- כאשר התהליך מגיע לסיומו (terminate) על מערכת ההפעלה לדאוג ל"איסוף" כל המשאבים (מסוג reusable)

Process Management

- Single-threaded process has one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads

Process Management Activities

מערכת ההפעלה אחראית לפעילויות הבאות בכל הקשור לניהול התהליכים:

- יצירה (create) ומחיקה/הריגה (delete) של תהליכי משתמש/מערכת (user and system processes)
- השהיה (suspend) והחזרה לפעילות (resume) של תהליכים
- אספקת מכאניזמים עבור:
  - process synchronization
  - process communication
  - deadlock handling

Protection and Security

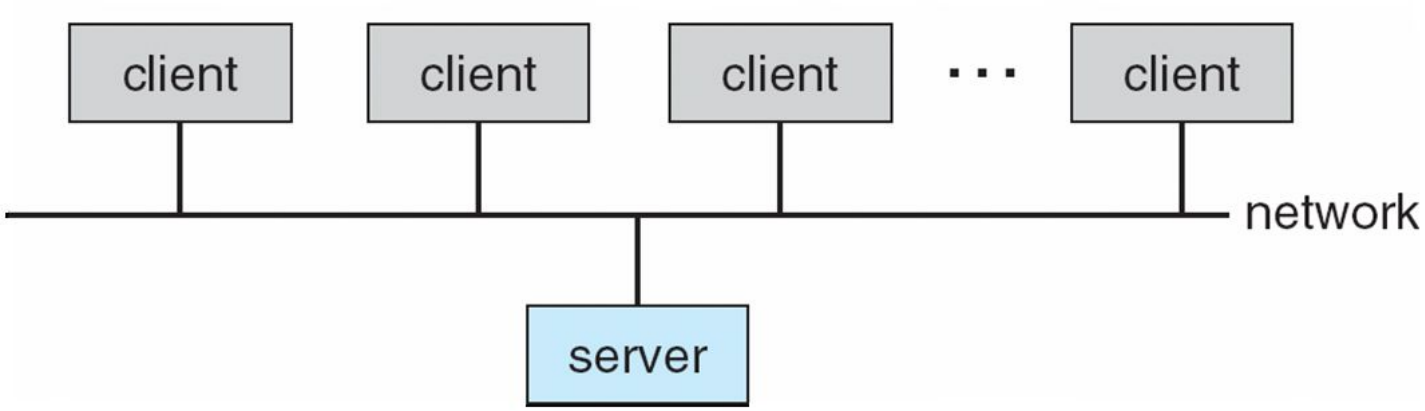
- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file

Computing Environments

- Traditional computer
  - Blurring over time
  - Office environment
    - PCs connected to a network, terminals attached to mainframe or minicomputers providing batch and timesharing
    - Now portals allowing networked and remote systems access to same resources
  - Home networks
    - Used to be single system, then modems
    - Now firewalled, networked

Computing Environments (Cont)

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server** provides an interface to client to request services (i.e. database)
    - **File-server** provides interface for clients to store and retrieve files



Peer-to-Peer Computing

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both
  - Node must join P2P network
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via **discovery protocol**
  - Examples include *Napster* and *Gnutella*

Open-Source Operating Systems

- Operating systems made available in source-code format rather than just binary closed-source
- Counter to the copy protection and Digital Rights Management (DRM) movement
- Started by Free Software Foundation (FSF), which has "copyleft" GNU Public License (GPL)
- Examples include GNU/Linux, BSD UNIX (including core of Mac OS X), and Sun Solaris

Answers from Last Year's Exam

שאלה 1 (4 נקודות)

יש הרואים במערכת ההפעלה resource allocator. אחרים רואים בה control program. תן דוגמה אחת בה משמשת מערכת ההפעלה כ- resource allocator ודוגמה בה היא משמשת כ- control program.

תשובה – resource allocation: הקצאת CPU, הקצאת זיכרון, הקצאת דיסק וכו'.

Control program – שליטה בהרצה של התכנית למניעת שגיאות ושימוש לא נכון במחשב (למשל גישה לאיזור אסור בזיכרון, system calls, שליטה על מרכיבי I/O).

שאלה 1 (10 נקודות)

למה משמש DMA? עבור אלו התקנים לא כדאי להשתמש ב-DMA? מהי הבעיה העיקרית הקשורה בשימוש ב-DMA כאשר המיפוי לכתובות בזיכרון הפיזי מבוצע בזמן הריצה?

שאלה 2 (10 נקודות)

ידוע כי העיצוב והמטרות התכנוניות של מערכת הפעלה (the operating system design and goals) נקבעים על-פי מערכת המחשב עליה היא מתוכננת לעבוד. בחר 3 מערכות מחשב השונות מבחינת מטרות מערכת ההפעלה המותקנת עליהן. כיצד שונות המטרות התכנוניות של מערכת ההפעלה למערכות השונות שבחרת ומדוע.

Video

- Direct memory access

End of Chapter 1