COSC450 Overview

Major Topic #1: Overview of Operating System

Slides #1

Slide #	**Short Summary**
1	Preview
2	Elements used in Computer, OS: Protected software provide interface between hardware and software
3	Macroscopic Diagram Breakdown
4	Hardware in Computer System (Physical Devices, Micro-architecture, Machine Language)
5	Macroscopic Diagram Breakdown (Circle- Software, Shell, Kernel, Hardware)
6	Von Newmann Architecture Diagram
7	Overview of Von Newmann
8	Von Newmann Bottleneck: Memory bottleneck since CPU is faster
9	CPU: executes instructions, utilized instruction cycle (Fetch & Execute Cycle)
10	Fetch Cycle: Reads instruction address, loads it, and moves program counter
11	Diagram (Instruction Cycle: Fetch Cycle)
12	Execute Cycle: Contents of IR decoded & executed, potentially involving interaction with memory and ALU.
13	An OS is: an extended machine from the user's perspective, and a resource manager from the developer's perspective.
14	N/A
15	The first gen of 'OS' consisted of vacuum tubes and plugboards (1945~1980)
16	N/A
17	First Gen used plugboards + tapes, took up whole room, 50 multiplication/sec
18	Second Gen of 'OS': Transistors and Batch System (1955~1965)
19	Picture of Second Gen Tape (from IBM)
20	^N/A
21	^N/A
22	N/A
23	N/A
24	N/A
25	N/A
26	Second Gen used Batch System to optimize use of expensive computer
27	Picture demonstrating workflow
28	N/A
29	Third Gen- IC and Multiprogramming (1965~1980)

30	Third gen launched by IBN, moved to electronic computer systems
31	Third gen used integrated circuit and CPU optimization techniques
32	Third gen used Multiprogramming, where processed are loaded into RAM and ran concurrently. Also uses CPU scheduler.
33	Diagram (Multiprogramming System)
34	Third Gen used Spooling, a buffering mechanism where data is temporarily stored as a file to be processed later
35	^ Spooling improved efficiency and multitasking
36	Diagram (Third gen)
37	Diagram (Third gen time sharing system)
38	Fourth Gen: Personal Computer built with LSI (Large Scale IC), VLSI, ULSI (1980~Present)
39	Fifth Gen: Mobile Computers (i.e. Smartphones)
40	Tons of operating system names

Slides #2

Slide #	**Short Summary**
1	Review (Vonn Newman, Generations of Computers)
2	Preview- Computer System Architecture, Operating System Implementation
3	Preview- Multiprocessor System Types, Multiprogramming, Operating System Operation
4	Modern computers consist of CPU + Memory + I/O Devices, with each device controller maintaining local buffer storage and in charge of a I/O device (also having a device driver, which becomes part of OS)
5	Diagram: I/O Devices -> USB controller, video controller, disk controller, etc.
6	CPU: Brain of computer, retrieves instructions (cycle) from memory to execute. Has cache and registers due to Von Newmann Bottleneck
7	CPU composed of: ALU (Arithmetic Logic Unit), Control Unit, Cache, & Registers (General, Program Counter (PC), Stack Pointer, Program Status Word (PSW))
8	CPU: When process stops running, OS saves content of each register in Process Table to finish the job. CPU performance can be improved by using pipelined design for fetch, decode, and execute process
9	CPU: may have multiple cores/calculation units
10	When I/O devices are ready to receive/send data, interrupts OS by sending signal. For I/O operations, instructions (read/write) are sent to device controller's register, then sent to device local buffer, then checks any error, then driver gives control to other parts of OS
11	Interrupts: key part of how OS & hardware interact. Sends interrupt to CPU via system bus, in which immediately sends execution to fix location for service

	routine. CPU resumes once this is finished. For quickness, OS maintains a table of pointers (interrupt vectors) in low memory for this.
12	Hardware has CPU wire called interrupt-request line, of which the CPU reads the interrupt number from and jumps to handler routine corresponding to number in interrupt vector.
13	CPUs have non-maskable interrupt line & maskable interrupt line
14	Diagram: Interrupt vector layout
15	Interrupts are used throughout OS systems to handle asynchronous events, with device controllers and hardware faulters raising interrupts. This allows the most urgent work to be done first (interrupt priorities), and is used heavily for time-sensitive processing
16	CPU can only load instructions from RAM. Secondary Memory -> Main Memory (RAM) -> Cache Memory -> Registers in CPU
17	Large portion of OS is dedicated to managing I/O, though interrupt-driven I/O can produce high overhead when used for bulk data movement. For this, DMA (Direct Access Memory) controllers can be used, independent from CPU
18	OS controls all I/O devices by: Issue commands to devices, Catching interrupts, handling errors. OS also provides interact between devices & the rest of the system
19	Most I/O devices consist of Mechanical Component, Electrical Components, and Device Driver (Software)
20	The bus in PC is the common pathway between CPU & peripheral devices. Parallel buses use slots on the motherboard, while serial buses have external ports
21	Diagram: Parallel & Serial Transmission
22	Parallel Buses: Offers fast data communication, however supports short distance communication due to crosstalk between the parallel line. Costs more and more pins to connect. Can come in form of Peripheral Component Interconnect (PCI) or Accelerated Graphics Port (AGP)
23	Serial Busses: Offers lower data connection, but supports long distance communication and costs less. Can come in form of Universal Serial Bus (USB) or FireWire (IEEE 1394)
24	Earlier Parallel Buses: Industry Standard Architecture (ISA), Extended ISA (EISA), Micro Channel (MCA), & VESA Local Bus (VL)
25	Diagram: Parallel Buses (Earlier Versions)
26	Single Processor Systems: contains one CPU with a single core. The core executes instructions and has registers to store data locally. These systems may have special-purpose processors as well that do not run processes. OS sends information to these special-purpose processors and monitors their status.
28	Multiprocessor systems allow increased throughput (calc. power), with N number of processors having slightly less than an N speed-up ratio. Includes multicore systems, which can be more efficient than multiple chips with single

	cores due to between-chip communication. One-chip structure uses significantly less power than multiple chips.
29	Diagram: Multiprocessor Systems
30	Multiprocessor Systems: Symmetric Multiprocessing (SMP)- All CPUs share global memory
31	Diagram (Symmetric Multiprocessing Architecture)
32	Multiprocessor Systems: Non-Uniform Memory Access (NUMA)- Each CPU has its own local memory
33	Diagram (Non-Uniform Memory Access)
34	Multiprocessor Systems: Clustered Systems
35	Types of Clustered Systems: Asymmetric Clustering (One machine is hot-standby mode) and Symmetric Clustering (Both machines running application & monitoring each other)
36	Multiprogramming: Tasks are stored in RAM and assigned to CPU using OS CPU scheduler
37	Multiprogramming: Several tasks are uploaded to RAM, and OS maintains process table containing essential information to facilitate multiprogramming
38	Multitasking: a logical extension of multiprogramming, allowing for users to perform more than one task at a time and switch back and forth among them.
39	Dual-Mode and Multimode Operation: Must distinguish between the execution of OS (kernel) and user code (user). A mode bit is added to hardware to indicate current mode (kernel 0, user 1). System must change mode from user to kernel when necessary.
40	Dual mode provides a means of protecting OS from errant users or hackers. This is accomplished by designing machine instructions to only be executed in kernel mode.
41	A system call allows a user program to request services from the operating system by triggering a trap to the interrupt vector, switching to kernel mode, where the OS verifies parameters, executes the request, and then returns control to the user program.
42	System Call example- read function
43	Diagram (Dual-Mode and Multimode Operation: System Call)
44	Step-by-step process, showcasing OS changing to kernel mode w/ System Call
45	Timer: OS maintains control over CPU, assigning it to process with timer. The timer may be fixed or variable and can interrupt after specified period. When the period is expired before finishing a job, process must wait for CPU time in ready queue.

Slides #3

Slide #	**Short Summary**
1	Review- Multiprocessor System Types, Dual Mode & Multimode Operation, Multitasking vs. Multiprogramming
2	Preview- OS as resource manager, Operating System Structures
3	OS as Resource Manager- Process, Memory, File, I/O System, Deadlock, and Cache Management
4	Process Management: OS is responsible for creating/deleting, scheduling, and suspending/resuming processes. OS must also provide means for process synchronization and IPC
5	For Program to be executed, CPU must load, read, & write data from memory. When the program terminates, memory space is declared available for next program
6	Memory Management: OS must monitor/manage utilization of memory, dynamically assign/release memory, and determine the processes/data to transfer in/out of RAM
7	OS must provide uniform, logical storage view for user to save a file. OS must implement mass storage, organizing into directories and controlling which user can access a file and how the user accesses it
8	File Management: OS is responsible for creating/deleting files/directories, supporting primitives for manipulating files/directories (ex. read), mapping files onto mass storage (ex. HDD/SSD), Backing up files on stable storage media
9	Mass-Storage Management: OS is responsible for mounting/unmounting, free-space management, storage allocation, disk scheduling (HDD), partitioning, and protection
10	Cache is a hardware or software component that stores data which might be used again soon. Limited size, so cache management is an important OS operation. Can greatly improve performance
11	Movement of information between levels of storage may be controlled by either hardware or OS (cache->CPU/registers is hardware, disk->memory is OS)
12	Deadlocks between processes occur due to limited shared resources, with some processes needing more
13	Deadlock Management: Resource Allocation Graph
14	Deadlock Management: Example- Two processes require same resource, so they stay in blocked state forever
15	Deadlock Management: Four strategies for dealing with deadlock: Ignore, Detection and Recover, Dynamic avoidance by careful allocation, and Prevention by negating one of the four conditions necessary to cause deadlock (Mutual Exclusion, Circular wait, Hold and wait, No preemptive)
16	I/O: OS uses I/O subsystems for managing I/O devices. These subsystems consist of a memory management system (buffering, spooling, caching),

	General device drivers interface, drivers for specific hardware, and processor for specific hardware
17	OS controls all I/O devices by issuing commands, catching interrupts, and handling errors. OS provides interface between devices & rest of system
18	OS Structures: Monolithic, Layered System, Microkernels, Virtual Machine, Client-Server Module, Exokernels
19	Monolithic System: Written as a collection of procedures, with each procedure having a well-defined interface and with each one being free to call any other one
20	Possible structure for a monolithic system: A main program, service functions, and utility functions
21	Diagram- Structure of Monolithic OS
22	Layered System: OS is divided into several layers: process management (layer 0), memory management (1), IPC (2), I/O management (3), Deadlock management (4), user program (5), system operator process (6)
23	With layered system approach, designers can choose kernel-user boundary
24	Microkernels: Achieve high reliability by splitting the OS into small well-defined modules. Only one module runs in kernel mode (rest in user)
25	Diagram: Microkernels
26	Virtual Machine: Virtual Machine Monitor (Hypervisor) runs on bare hardware and does multiprogramming by providing several virtual machines. Each virtual machine are exact copies of bare hardware, with different virtual machines being able to run a different OS
27	Virtual Machine: Virtualization example would be company running several web servers on same machine
28	Diagram: Virtual Machine
29	Diagram: Virtual Machine
30	N/A