


COMP 310/ECSE 427
Lecture #2

Overview of General Purpose Operating Systems




Announcements

- Extra books



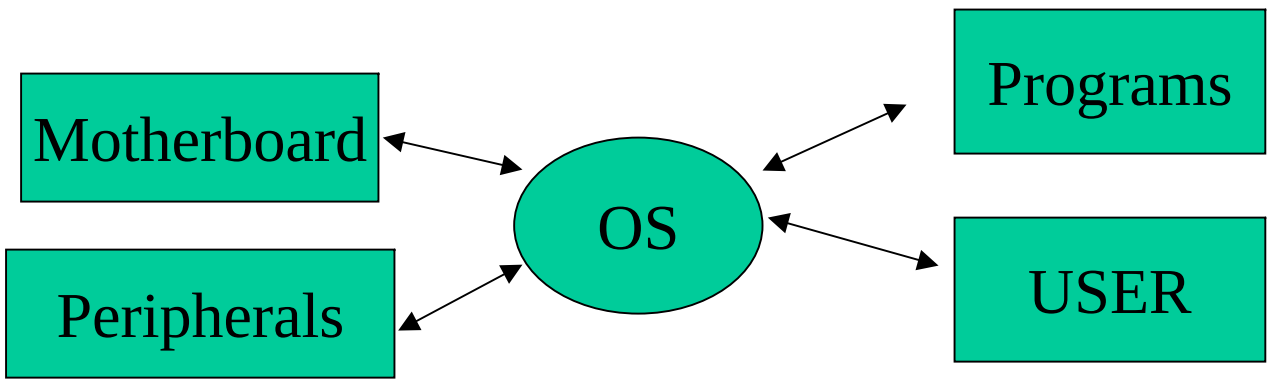

Understanding Operating Systems

- Operating systems are engineered
- This means they are evolutionary
- Current technology builds on older technology
- A gradual introduction to modern OS technology can be viewed through the lense of history...




What Is a General Purpose Operating System?

A program that manages and integrates the hardware resources in a machine, allowing other software, machines and humans to access these resources in a controlled and friendly manner.






Part 1

The Evolution of OS Technology




1930 - Now

Operating Systems

ENIAC

General Purpose OS

Mainframes

Desktop

Hybrid

Real Time

Handheld

- Single user batch
- Simulated Real-time
- Time Sharing

- Real-time single
- Time Sharing

- Multi-processor
- Distributed
- Clustered
- Code migration

AS 400
Sys 360
Unix

DOS
Windows
Mac
Linux

Cell phones
Gas pumps
Washing machines

Windows
Palm

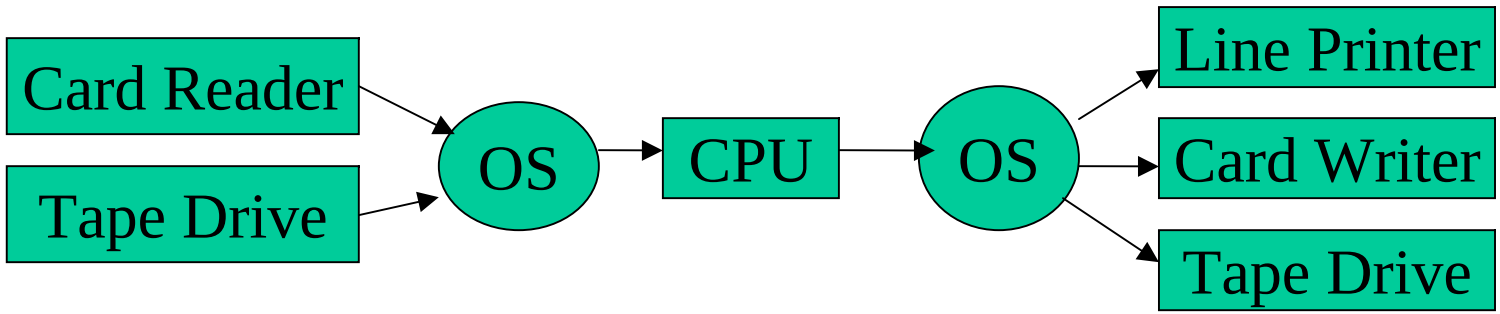
Simulated Real-Time ~ Web Forms (modern version) --- CICS with Cobol (old version)?




1950's

Main Frame Computers (>$100K, large, room size, complex)?

Single User Batch Systems

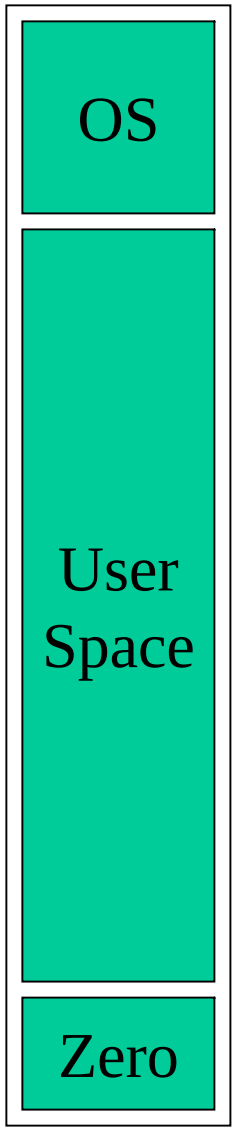


No direct access to the computer

(discuss operation)?
?
(Flowchart it?)?




Memory Organization



Human Procedure:

- Write code on punch cards (1950s) or in a script file (1960s)?
- Submit code to system operator
- System operator groups common programs by resource ("Batched")?
- Batched scripts are inserted into the card reader in order
- OS sucks in all the cards
- Line printer outputs results (syntax errors or correct output)?

Single User Batch Operating System Procedure:

- Read in cards until special marked card that says end of program
- Store as binary in user space
- Pass control to CPU by assigning IP to first instruction
- Wait for control to be returned (if not because:crash or infinite loop)?
- Results have been output while program executed
- Verify if more cards in reader, if so go to step 1 else step 7
- Computer stops, goes to system operator interface
  - Turn off machine
  - Busy wait loop
  - Start procedure

The cards were a natural queue

(read batch, process, repeat next batch)?




Single User Batch OS ARCHITECTURE:

- Input Medium: Cards or Tape spools

- Loader: converts cards / tape into binary and stores in RAM
- Controller: Assign IP to first instruction (Pass Control to CPU)

- Output Module: Binary RAM to Card/Tape
  - "Line Printers" were available in later systems (dot-matrix)?

1950's

Whirlwind

1960's

IBM 360

An Early Batch File

This is the entire program = code + OS commands

TAPE SPOOL = ACCOUNTING
FILE DESIGNATION = 5, ACCOUNTS.TXT
FILE DESIGNATION = 6, USERS.TXT
OUTPUT = LPT1
BEGIN PROGRAM

```
main(void){
    int x;
    for(x=0;x<10;x++) fprintf(6,“%d”,x);
}
```

END OF PROGRAM
END OF FILE

Or
PROGRAM = filename

This should be FORTRAN or COBOL




1960's

Keyboard and Screen

Tapes

Multi-user Multi-Process Batch System

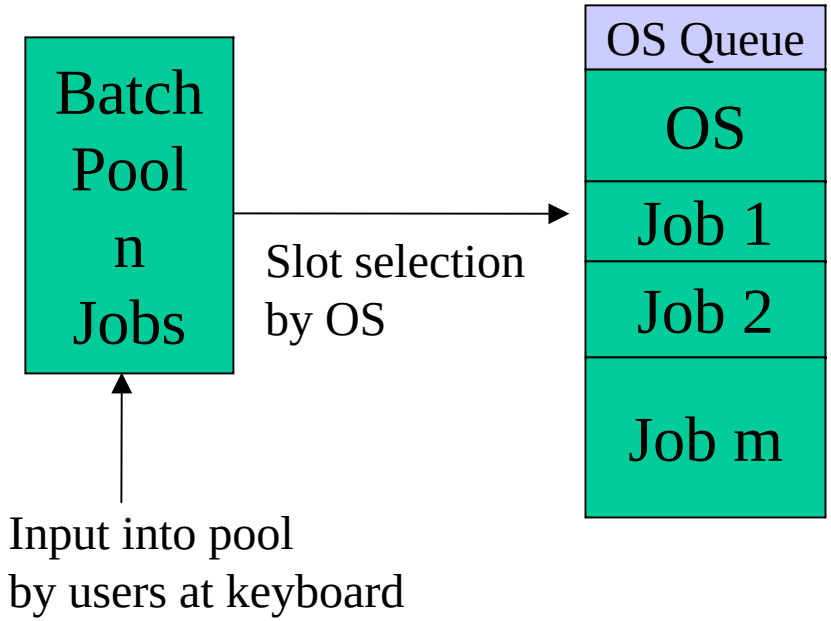
- Similar to Single-user batch systems
- Direct access to computer via typewriter
- User created script files that behaved as auto batch files
- Operating system could perform interrupt-processing
- Output devices: cards, tape, line printer, and screen
- Input devices: typewriter, tape, and cards




Terminology

- Single-user: Only one person used the computer in a fixed period of time (e.g. my laptop)?
- Multi-user: More than one person can use a computer in a fixed period of time (e.g. in the last 15 minutes 10 people used the PC ~ one at a time, single-user fashion, using batch processing)?
  - A Process: An algorithm executing in RAM (dynamic not static as a program on disk)?
- Multi-Process: more than one algorithm in RAM at the same time but not executing at the same time
  - Time Sharing: each process executes for a quanta then stops and allows the next process to execute
- Parallel Computing: the algorithms each have their own CPU and run independently at the same time.

Describe…



M < N
M fixed slots in RAM
N jobs in pool
Loaded jobs wait turn in queue

JOB SCHEDULING

- Step 0: Load m jobs from pool
- Step 1: Execute Job_i all other jobs on hold.
- Step 2: Job executes **until I/O** operation
- Step 3: Job "suspended" waiting for I/O (find tape, wined tape)?
- Step 4: OS Switches to next job
- Repeat until all jobs complete

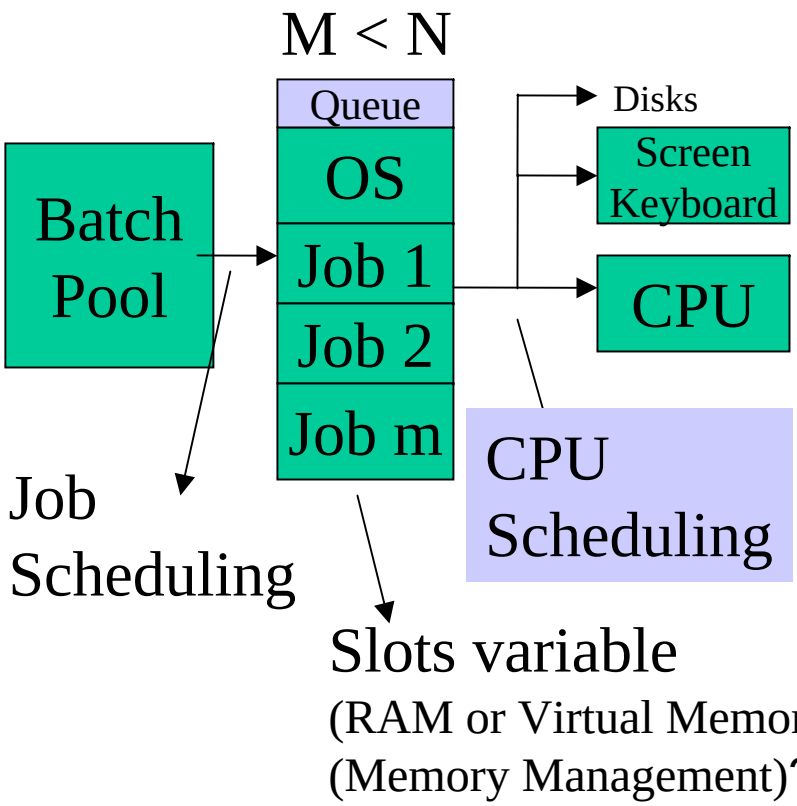
MULTI-USER & BATCH

•Good: CPU idle time minimized

•Problem: No user interaction

1970's Time Sharing Systems - Multi-tasking

(Multi-Process)?



- Step 0: Job in RAM called a "Process"
- Step 1: Time Slice (**quanta**) = milliseconds
- Step 2: Execute JOB_i until
  a- Completion
  b- I/O operation then suspend
  c- Time slice used up
- Step 3: Switch to next process (called: task switching)?
- Repeat – if slot available load another job from the pool

•Good: All around
•Problem: Not multi-CPU!




Virtual Memory

- Memory understood as the following:
  - Physical Disk Space (limited HDD – compiled stored program)?
  - Physical Process Space (limited RAM – data + process executing)?
  - Logical Process Space (infinite variable contiguous space for process + data)
- Logical Memory understood as:
  - The complete memory space given to a single executing process (Behaves as if it is all alone in RAM)?
  - Physically this space can be distributed across Disk, RAM, Network or **not even** contiguous (even though addressed as if)?




Physical Memory

Addr: n-1

Job X (part 1)?

Addr: 0

RAM

Job X (part 2)?

Sector 0 to Z-1

Tape / Disk

OS has a split personality. It manages the job physically one way but executes it logically in another way.

Virtual Memory

Addr: m-1

Job X (part 1)?

Job X (part 2)?

Space: m=Z+n

Addr: 0

Logical Address



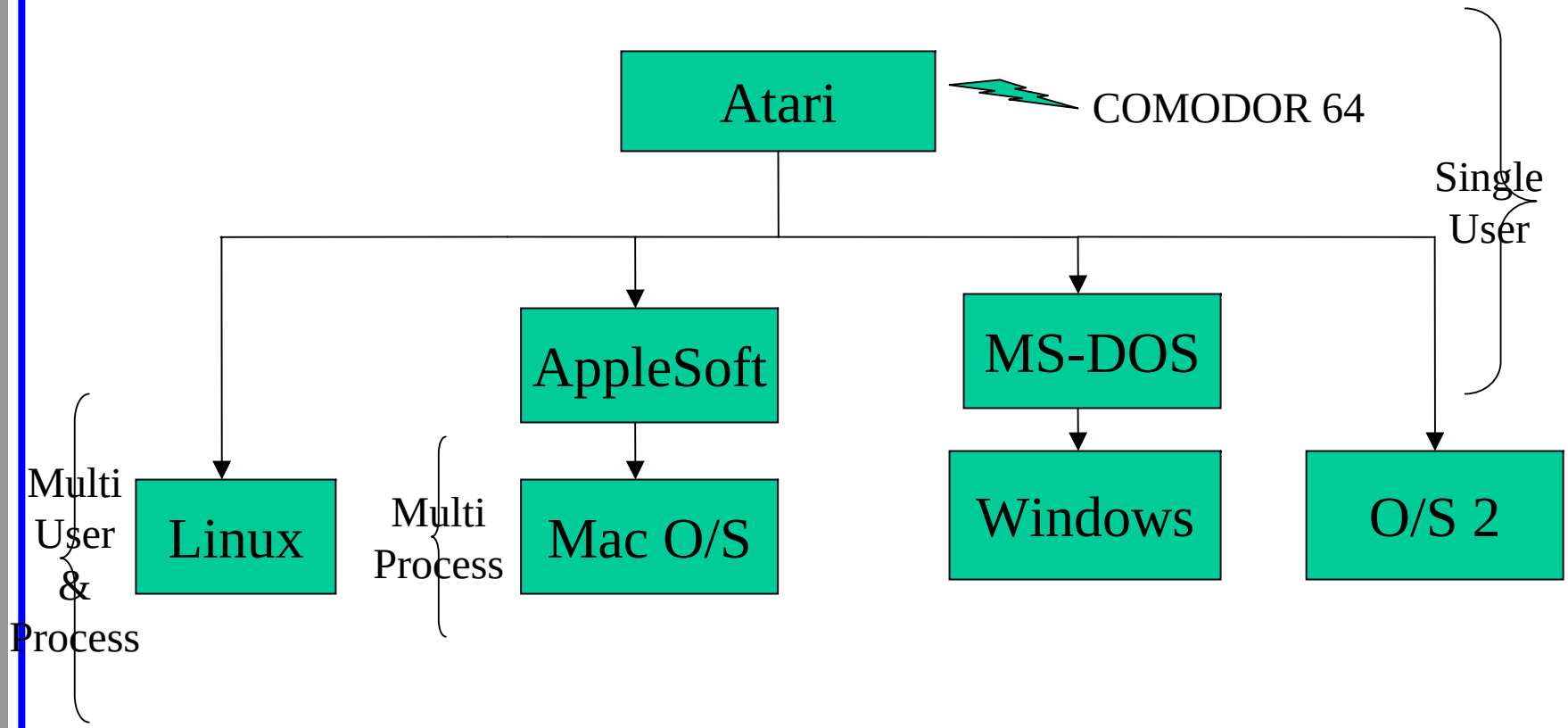

Part 2

Early Operating Systems (Personal Computers)?




1970's & 1980's

Personal Computers (PC)?



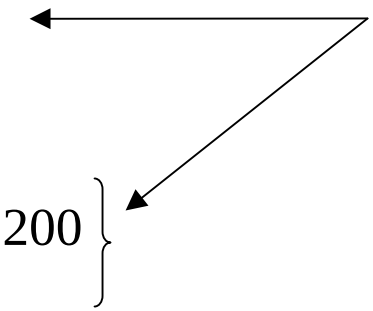
These OS systems are clones of mainframe OS systems but **without** true batch processing while **focusing on** real-time interfaces.

TSR – Terminate Stay Resident

(Hacking multi-processing in MSDOS) – Part 1 / 3

Standard Programs

- LOAD ADDRESS 100 TO 200
- main(void) {
- printf("hello"); }
- TERMINATE ADDRESS 100 TO 200
- RETURN TO OS INTERFACE



Compilers add this

Loader: tell OS we need x bytes in RAM for program.

Terminate: tell OS to free x bytes in RAM and make available for others.

Return: Display the OS user interface (command prompt or window)?




TSR – Terminate Stay Resident

(Hacking multi-processing in MSDOS) – Part 2 / 3

TSR Programs

- LOAD ADDRESS 100 TO 200
- main(void) {
- printf(“hello”);
- goto x;}
- LET x = ADDRESS OF INTERRUPT 10
- INTERRUPT 10 POINT_TO 100
- TERMINATE ADDRESS (empty)?
- RETURN TO OS INTERFACE

You add this

When another program is running and you issue interrupt 10 the OS switches to the TSR, the TSR then returns you back to OS and your program.

Interrupt: interrupt 10 is changed to activate the program at address 100




TSR – Terminate Stay Resident

(Hacking multi-processing in MSDOS) – Part 3 / 3

The Basic OS Loop:

- Load program
- Run program
- IF I/O THEN

  Since all I/O goes through OS (unless in assembler) stop program and switch to OS functions using an Interrup Table.
- IF END OF PROGRAM THEN

  Free memory and return to OS user interface
- IF DO I/O, THEN once done..

  Return to program to continue execution (go to step 2)?

Key idea




Part 3

Exotic Operating Systems (Modern Computers)?




Exotic Operating Systems (1980's)?

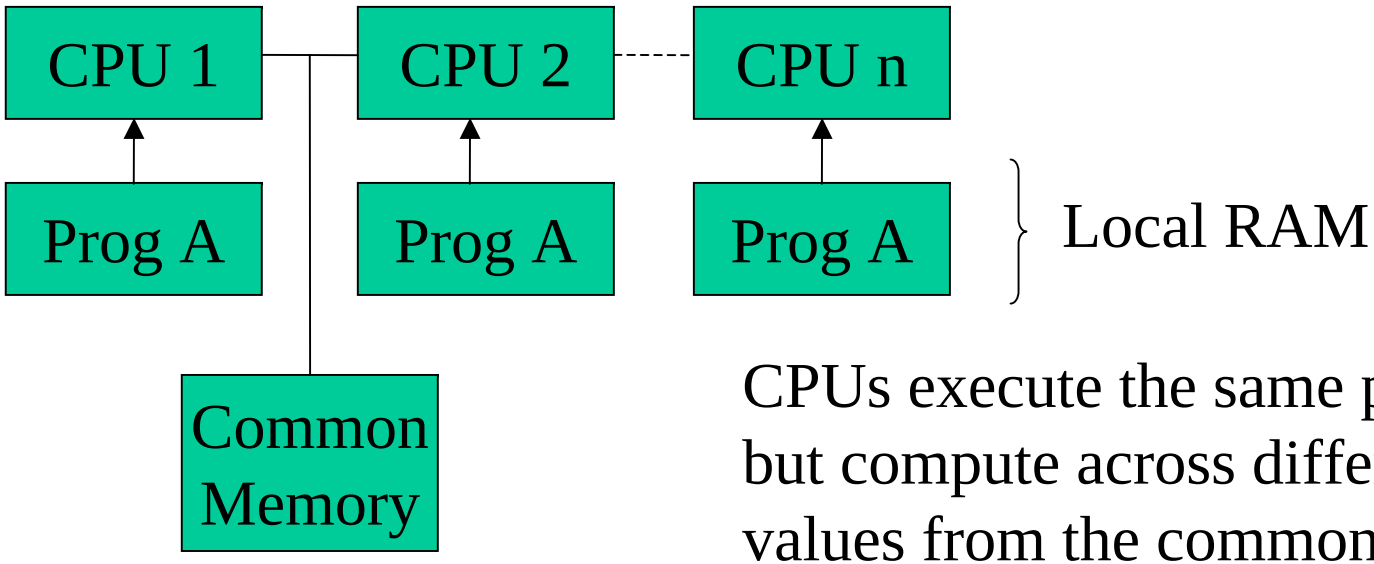
(Think how these would function…)?

Multi-Processor Systems

(Parallel CPU Systems / Tightly Coupled Systems)

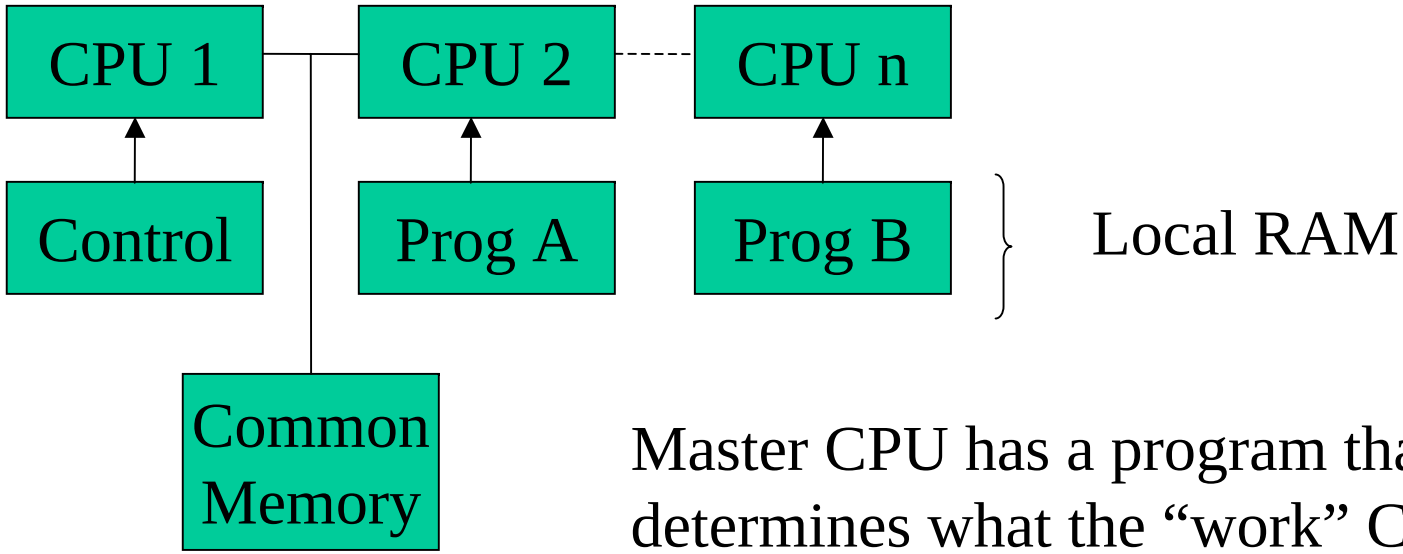
Def: 1 Machine with n CPUs.

Type 1: Symmetric Multiprocessing (SMP)?



CPUs execute the same program but compute across different data values from the common memory.

Type 2: Asymmetric Multiprocessing (AMP)?



Master CPU has a program that determines what the "work" CPUs will do. Work CPUs execute another program.

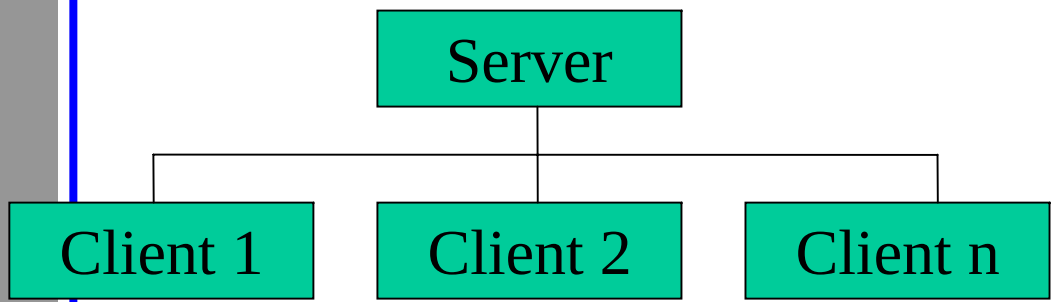



Distributed Systems

Networks: Connecting many individual computers together.

Types:
1. LAN - Local Area Network (one building)
2. WAN - Wide Area Network (one planet)
3. MAN - Metropolitan Area Network (one city)
4. SAN - Small Area Network (wireless eg. Blue Tooth)?

Type 1: Client-Server Systems



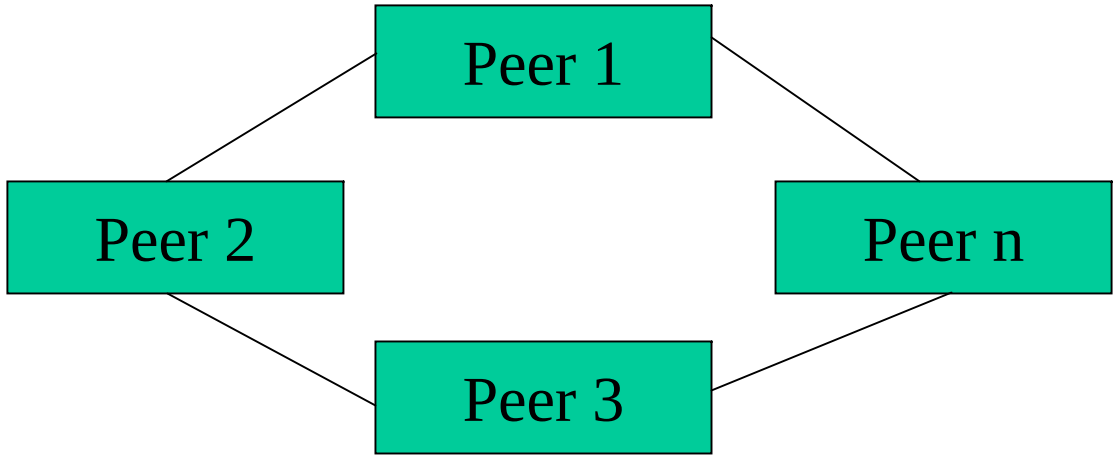
Server Form A:
Central System
(Unix) Server executes soft

Server Form B:
File Server System
(Windows) Stores data only




Type 2: Peer-to-Peer Systems

(Flowchart OS?)?



- No real server present
- Each Peer has its own RAM, Hard Disk and CPU
- Each Peer executes its own software
- Each Peer can also access resources marked “SHARED” from other Peers.
- Each Peer is given a name (like a drive letter) as the access method.

The WEB




Exotic Systems 1990's

Clustered Systems

- Like multi-processor systems but use multiple computers
- Computers interconnected via some type of network
- This is called CLUSTERING.

Type 1: Asymmetric Clustering (Master/Worker arrangement)?
Type 2: Symmetric Clustering (All run same program on diff. Data)?

(The computers are aware of each other and can synchronize their data access).

SETI
screen saver




Real-Time Systems

- Runs only one program until completion
- I/O processed in actual human time
- Direct interface and immediate processing of hardware signals

Eg. Robotic arms, Airplane flight control system, flight simulators, dish washer control panel, gas station pumps, cell phones.

(flowchart this?)?




Handheld Systems

- Same as single user PC systems
- Challenge is limited resources and novel interfaces

Type 1: Single User (can or cannot be connected to home PC)?
Type 2: Network Enabled (Internet, LAN, Infrared connections)?
Type 3: Multi-Process (not yet)?




Exotic Systems 2000's

Code Migration and Distributed OS

- Program divided into objects
- Each object can run as a program on its own
- All objects run on your machine until…
  - More resources needed - use network to send an object to another computer for processing
- All our computers become ONE processing environment SHARED with everyone.




Part 4

Try At Home…




Try At Home

- Try out SETI-AT-HOME
- Think about how e-mail functions over a distributed system (flowchart it)
- Given the OS computing environments described in this class, which do your home PC support?
- Question:
  - What is the main difficulty that a programmer must overcome in writing an operating system for a real-time environment?
  - In a two computer cluster system processing records from a database, how can they coordinate (work together)?