Part1

Wednesday, April 3, 2019 9:38 AM

Components of a simple PC

- Back in the day, computers would run null processes. Now they can "sleep".
- Video controller, hard drive controller, usb controller, network controller on buses inside computer.
- "Computer" can be used for anything with these types of controllers and processors. Phones are like mini computers.

Multicore CPUs

- Pipeline CPU
  - Fetch - Decode - Execute
- Superscalar CPU
  - Fetch - Decode v
    - Buffer - Execute
  - Fetch - Decode ^
  - Allows processors to act as ig they are actually multiple processors and the os won't know the difference
    - EX: school soe system has 64 physical cores that act as 256 cores
- Dual-core CPU
  - Unified Caches vs Separate Caches
    - Choice depends on which one you need

Storage pyramid

- Lower has larger capacity but slower access latency
  - Registers
  - On-chip caches
  - Cache SRAM
  - Main memory DRAM
  - Flash memory
  - Magnetic Disk
  - Magnetic tape / optical disk
- Goal: really large memory with very low latency
- Solution: move data between levels to create illusion of large memory with low latency
  - Some movement is done in hardware
  - Most is done by the software, like OS

Disk drive structure

- Data stored on surfaces
  - Up to 2 surfaces per platter
  - Multiple platters per disk
- Data in concentric tracks
  - Tracks broken in sectors
  - Cylinder = corresponding tracks on all surfaces
- Data read and written by heads
  - Actuator moves heads
  - Heads move in unison

Flash memory structure

- Flash is divided into erase blocks
  - Blocks must be erased before being written
- Flash is read and written in pages
- Flash Translation Layer - FTL - manages the device

    - Allows the file system to work with Flash like how it works with the disk
    - Maps positions that the file system think contains the object to where it actually is contained in flash
- Flash will eventually run out unlike magnetic stuff
- If you want to write, you need to erase the block; unlike how a disc can be written to whenever you wish to

Memory

- In virtual memory, user data and user programs are separate from operating system
- In physical memory, everything is connected
- Single base/limit pair: set for each process
- Two base/limit registers: one for program, one for data
- Operating system is kind of like a function
    - Something causes a jump to the OS
    - Maybe it's the CPU or someone typing a keyboard or maybe a hacker from someplace else

Anatomy of a device request

- Left: Sequence as seen by hardware
    - Request sent to controller, then to disk
    - Disk responds, signals disk controller which tells interrupt controller
    - Interrupt controller notifies CPU
- Right: Interrupt handling (software pov)
    - Interrupt
    - Process interrupt
    - Return

Processes

- Process = program in execution
    - Address space (memory) the program can use
    - State (registers, etc)
- OS keeps track of all processes in a process table
- Processes can create other processes
    - Process tree tracks these relationships
    - A is the root
    - A created three child processes: B, C, D
    - C created two child processes: E, F
    - D created one child process: G

Inside a Unix process

- Stack - 0x7fffffff
- …
- …
- Data
- Test - 0
- Processes have three segments
    - Test: program code
    - Data: program data
        - Malloc, new
        - Variables
        - Malloc extends the data segment
    - Stack automatic variables
    - Procedure call information
- Address space growth
    - Text doesn't grow
    - Data grows up
    - Stack grows down

Hierarchy helps us simplify complex things

Each process thinks it’s the only process going on at a time

- Inter-process communications