The OS sandwich

Runs the software and manages the hardware

- RISC vs CISC
- ADDRESS BUS
- DATA BUS
- BIG/LITTLE ENDIAN
- PIPELINING
- LOAD/STORE
- ADDRESSIBILITY
- ALIGNMENT
- ISA
- ETC

SOFTWARE

HARDWARE

Operating System
HW organization details

- **Bus**
  - Transfers one “word” at a time
    - Fundamental system parameter
    - Amount can fetch from memory at one time
    - Tends to be the size of the data bus

- **I/O Devices**
  - System’s connection to the external world
  - Transfers information back and forth between the I/O bus and an I/O device

- **Main Memory**
  - Temporary storage
  - Holds both the program and the data it manipulates
    - Von Neumann architecture
  - Is organized as a linear array of bytes each with its own unique address starting at zero
HW organization details (cont)

- Processor (CPU)
  - Interprets/executes instructions stored in main memory
  - Updates the PC to point to the next instruction

- PC
  - Points at (contains the address of) some machine-language instruction in main memory

- Register file
  - Small storage device that consists of a collection of word-sized registers, each with their own name

- ALU
  - Computes new data and address values

- ISA – instruction set architecture defines
  - The processor state
  - The format of the instructions
  - The effect each instruction will have on the state
  - Instructions:
Memory (chp. 6)

- Example... go from the bedroom to the kitchen
  - In a studio apartment
  - In a town house
  - In a medium priced home
  - In the White House

- Example... find an address in computer memory
  - When you have 10 bytes of memory
  - When you have 10,000 bytes of memory

- Cache
  - Smaller and faster
  - Temporary staging areas
  - Goal → make the transfer/copy operations happen asap
    - It’s easier and cheaper to make processors run faster than it is to make main memory run faster!

- Hierarchy
  - The storage at one level serves as a cache for storage at the next lower level
  - The farther you are away from the action, the longer it takes to make something happen
Abstraction

Provided by the OS

Process (chp. 8)
- The running of a program done by the processor
- Threads = multiple execution units
- Includes memory and I/O device (i.e. files abstraction)

Virtual Memory (chp. 9)
- Provides each process with the illusion that it has exclusive use of the main memory
- Program code and data
  - Includes files
  - Begins at same fixed address for all processes
  - Address space (chp. 7)

Files (chp. 10)
- Sequence of bytes
Address Space... a quick look

- An array of 8-bit bytes
- A pointer is just an index into this array

<table>
<thead>
<tr>
<th>ADDRESS SPACE</th>
<th>Description/info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel virtual memory</td>
<td>Memory invisible to user code</td>
</tr>
<tr>
<td>User stack (created at run time)</td>
<td>Implements function calls</td>
</tr>
<tr>
<td>Memory mapped region for shared libraries</td>
<td>Ex. printf function</td>
</tr>
<tr>
<td>Run-time heap (created at run time by malloc calloc)</td>
<td>Dynamic in size</td>
</tr>
<tr>
<td>Read/write data</td>
<td>Program (executable file)</td>
</tr>
<tr>
<td>Read-only code and data</td>
<td>Fixed size</td>
</tr>
</tbody>
</table>

32/64 bit starting address

Address 0

Notice symbolically drawn with memory “starting” at the bottom
What is a system?

“A collection of intertwined hardware and systems software that must cooperate in order to achieve the ultimate goal of running application programs”