virtual vs. physical memory

- types of physical memory
- paging

Wed. March 9, 2016
MIPS Memory

next three lectures
virtual address

514 - 398 - 3740

cell tower

cell phone

physical address
virtual memory (program addresses)

\[ 3^2 \times 2 \text{ bytes} \]

physical memory

- RAM
- disk
- flash
- etc...
virtual memory (program addresses)

How do multiple programs share the same (finite) address space?

How to reconcile different sizes of program vs. physical memory?

physical memory

- RAM
- disk
- flash
- etc...

"process" (running program)
Sizes of Memory

$2^{10} \approx 1 \text{ KB (kilobyte)}$

$2^{20} \approx 1 \text{ MB (megabyte)}$

$2^{30} \approx 1 \text{ GB (gigabyte)}$

$2^{40} \approx 1 \text{ TB (terabyte)}$

$2^{50} \approx 1 \text{ PB (petabyte)}$

$2^{60} \approx 1 \text{ EB (exabyte)}$
Floppy disk (1.4 MB) (obsolete)
CD \~ 1 GB

DVD \~ 10 GB

optical (laser)
Hard Disk Drive (HDD)

~ 1 TB
HDD access time

- 100 rotations per second

⇒ average access \( \frac{1}{2 \times 100} \) sec = 5 ms

⇒ \( 2 \times 10^9 \) clock cycles/sec \( \times \) \( 5 \text{ ms} \) access = \( \frac{7 \times 10^7 \text{ clock cycles}}{\text{access}} \)

SLOW!
Flash (SSD - solid state drive)
semi-conductor - very fast access $10^{-6}$ sec
(although it varies between technologies)

USB

SD card (digital camera, cell phone)

phone, tablet
( flash can replace HDD )
**RAM**
- fast and expensive
- access in one clock cycle (10⁻⁹ sec)

**SRAM**
- Slower but less expensive
- access in ~10 clock cycles

**DRAM**
- Semiconductor (silicon)
- 1 cm
- CPU SRAM
Volatile vs. non-volatile

- RAM (SRAM, DRAM)
- Flash (SSD)
- Disk (HDD, CD, DVD)

Does storage vanish when power is off?

YES NO
We would like to access Memory in one clock cycle. However, there is a tradeoff between the speed and size of physical memory (can't be large and fast).
Memory Hierarchy

- **SRAM**: fast and expensive (thus, small)
- **DRAM**: slow and cheap (thus, large)

- **HDD, SSD**: slow and cheap (thus, large)

- **Cache, Main Memory**: volatile
- **External, Secondary Storage**: non-volatile
virtual memory

- size
- number of clock cycles / access

physical memory

- SRAM
  - "cache"
- DRAM
  - "RAM", "main memory"
  - "HDD", "SSD" (tablet)

- MIPS Memory (2^32 bytes)
- instructions
- data
- other

<table>
<thead>
<tr>
<th>size</th>
<th>~MB</th>
<th>~GB</th>
<th>~TB</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of clock cycles / access</td>
<td>1</td>
<td>~10</td>
<td>10^6</td>
</tr>
</tbody>
</table>
lecture 16

virtual vs. physical memory

- types of physical memory
- paging
  - how to translate (map) virtual to physical?
  - page tables
  - page fault and page swap
There is nothing significant about the square tile geometry in this sketch.
Q: How to translate a virtual address to a physical address?

Note that both the user part of Memory and (part of the) kernel part of Memory is paged.
Example: suppose 1 page = $2^{12}$ bytes

How many pages?

Virtual Memory

(4 GB = $2^{32}$ bytes)

--> $2^{32} / 2^{12} = 2^{20}$

Physical Memory

RAM

(e.g. 1 GB = $2^{30}$ bytes)

--> $2^{30} / 2^{12} = 2^{18}$ pages

HDD

(e.g. 1 TB = $2^{40}$ bytes)

--> $2^{40} / 2^{12} = 2^{28}$ pages
How to translate (map) a virtual address to a physical address?

A virtual address is 32 bits. These are the program addresses we have been talking about for the last few weeks.

Again, take example that a page is $2^{12}$ bytes.
virtual address

physical address (RAM)  e.g.  1 GB = $2^{30}$ bytes
virtual address

31 ... 12 11 ...

virtual page number  page offset

table lookup

copy

physical page number  page offset

39 ... 12 11 ... 0

physical address  (HDD)  e.g.  1 TB = 2^40 bytes
"Page table"

Data structure in kernel that translates (maps) a virtual page number (address) to a physical page number (address).

"Valid bit" says whether page is in RAM (1) or on HDD (0).
Page tables are in a reserved data region in the kernel part of MIPS Memory. Note that they have both a virtual and a physical address. The page table region is not partitioned into pages. Rather, this region has a fixed mapping from virtual to physical memory.
Page Fault and Page Swap

- When a MIPS program tries to access an address whose physical page is on disk (HDD), we say that a "page fault" occurs. The page first must be brought into main memory (RAM) before the program can access that address.

- If there is no page available in main memory, then some page first must be moved out of main memory, and then the desired page can be moved in main memory. This is called a page swap.

- The page table must be updated (regardless of whether a page is swapped out).

Page swaps are done by a kernel program (OS) called the page fault handler (return to this in lecture 21 -- interrupts).
Next week's lectures

- more on page tables (we need a cache for them too !)
- how do caches work ?

replace these virtual memory boxes by caches