Memory Wrap Up

CS 241

Sept. 20, 2013

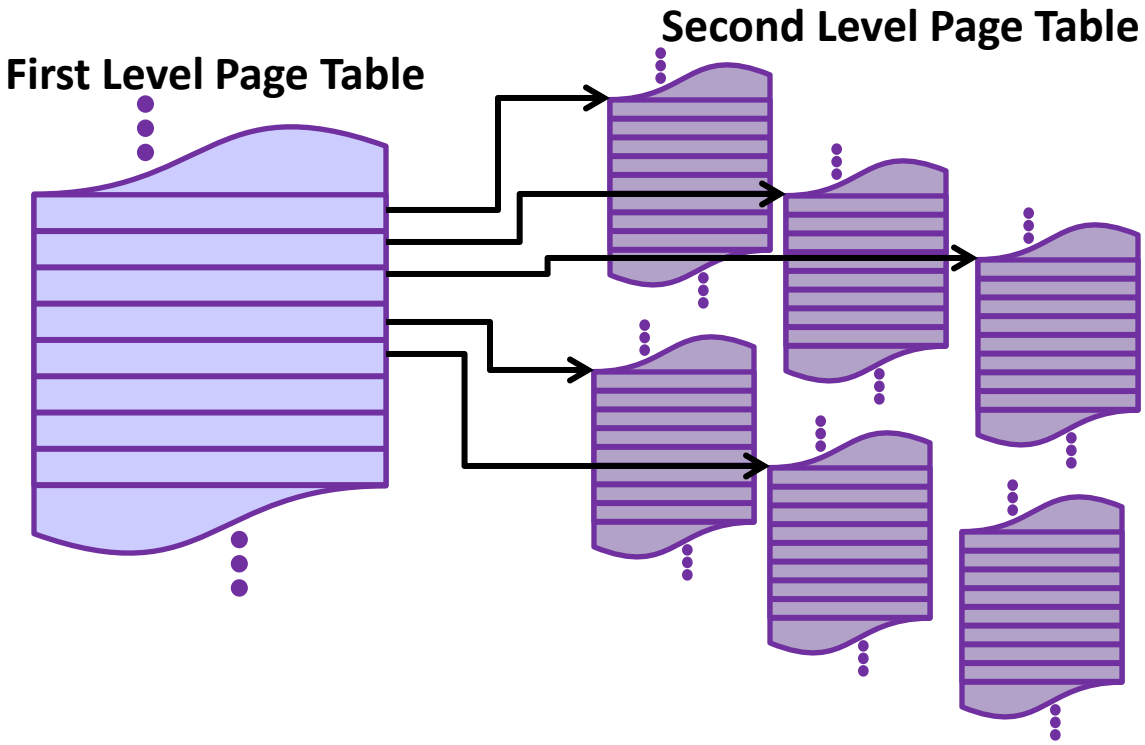
x86 Page Table

- In x86:
  - Pages are 4 KB in size
  - Virtual Addresses are 32-bits
  - Each PTE is 4 B in size

- How large is the Page Table for each process?

Multi-Level Page Table

- **Solution**: Create multiple levels of tables to look up a physical memory address.



Multi-Level Page Tables

- Each virtual address can now be divided into (n+1) different pieces for an (n) level page table.
  - **Example:** Two Level Page Table:
    - First Level Page Number
    - Second Level Page Number
    - Page Offset

- Given
  - 32-bit Virtual Addresses
  - 4 KB Pages
  - 12-bit First Level Page Table Number

- What are the components of the address:

`0x48503423`

- Given
  - 32-bit Virtual Addresses
  - 64 KB Pages
  - 8-bit First Level Page Table Number

- What are the components of the address:

`0x48503423`

- Given
  - 32-bit Virtual Addresses
  - 4 KB Pages
  - 4 B page table entries

- If **every-level** page table fits into a single page:
  - How many levels are in the page table?

  - How many bits is the index of each level?

- **Given:**
  - Each PTE is 16 B
  - The pointer to top-level of the page table is **0x1000**.
  - *: "PTE Contents" shows the contents of the memory if it was read as a PTE, and only shows the address field of the PTE.

- **Q: On system with a single-level page table and 256 B pages:**
  - What is the physical address of the virtual address **0x241**?

Memory Address	PTE Content*
0x1000	0x2000
0x1010	0x2100
0x1020	0x2200
0x1030	0x2300
0x1040	0x2400
0x1050	0x2500
0x2000	0x10000
0x2010	0x20000
0x2020	0x30000
0x2100	0x40000
0x2110	0x50000
0x2120	0x60000
0x2200	0x70000
0x2210	0x80000
0x2220	0x90000
0x2300	0xa0000
0x2310	0xb0000

- **Given:**
  - Each PTE is 16 B
  - The pointer to top-level of the page table is **0x1000**.
  - *: “PTE Contents” shows the contents of the memory if it was read as a PTE, and only shows the address field of the PTE.

- **Q: On system with a two-level page table where the index of each level is 4-bits:**
  - What is the physical address of the virtual address **0x1234**?

Memory Address	PTE Content*
0x1000	0x2000
0x1010	0x2100
0x1020	0x2200
0x1030	0x2300
0x1040	0x2400
0x1050	0x2500
0x2000	0x10000
0x2010	0x20000
0x2020	0x30000
0x2100	0x40000
0x2110	0x50000
0x2120	0x60000
0x2200	0x70000
0x2210	0x80000
0x2220	0x90000
0x2300	0xa0000
0x2310	0xb0000

Fragmentation

- Throughout all of memory, we have **completely unused space** that we call fragmentation.
  - **Internal Fragmentation:** Allocated memory that is never used by the algorithm.
    - Ex: Buddy system using 32 B for a 18 B request
    - ***Not the same thing as overhead!***
  - **External Fragmentation**: Unallocated memory, but too small to be useful.
    - Ex: A best-fit for a 72 B request may be 75 B.
      - The remaining 3 B is a very small hole, effectively useless.

Thrashing

- **Thrashing** occurs when a computer's virtual memory subsystem is in a constant state of paging, rapidly exchanging data in memory for data on disk.

Regions of Virtual Memory /process:

Strategies to manage the heap:

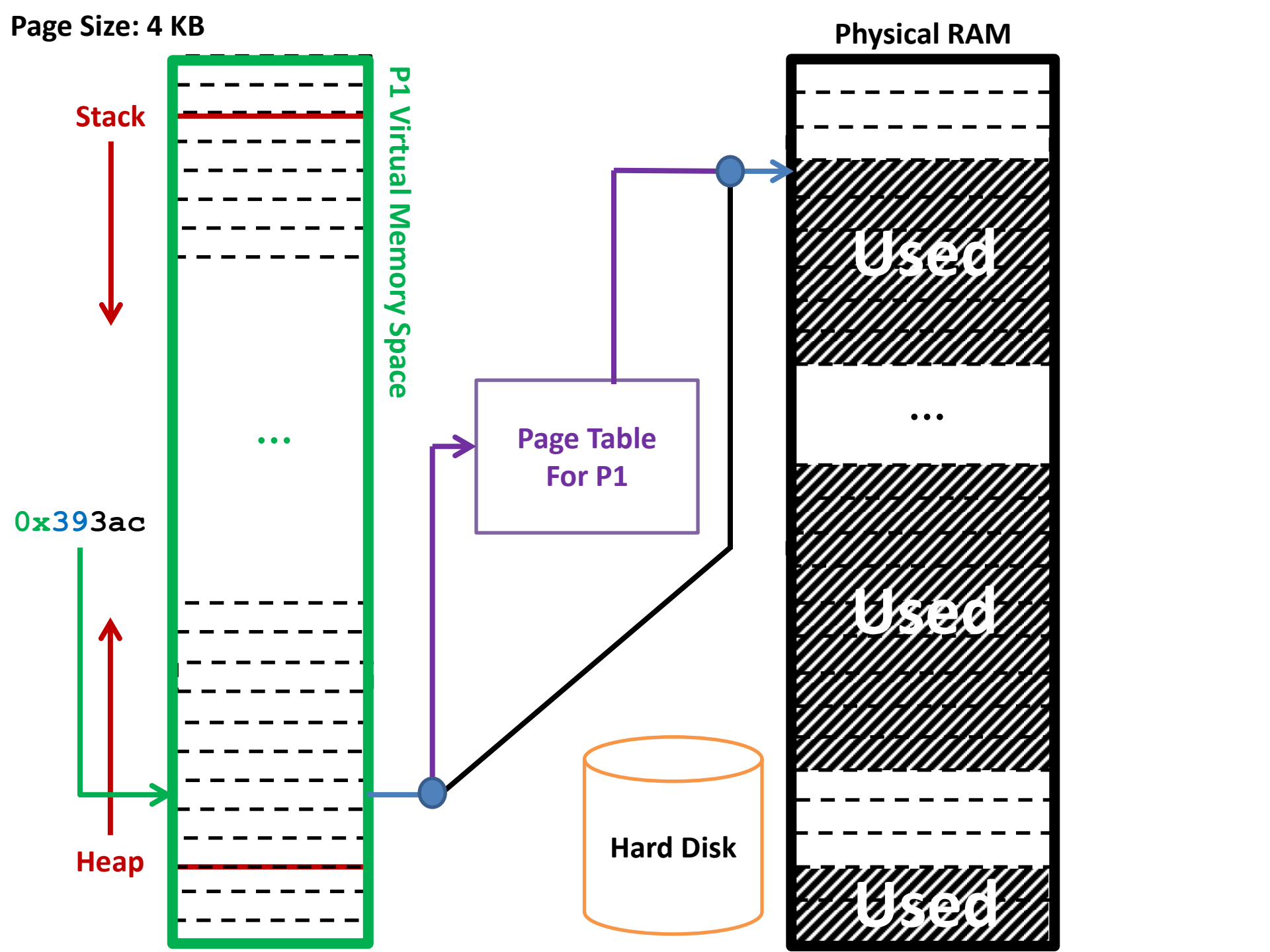
Two ways to map virtual memory to physical:

Virtual address components:

Contents of a PTE:

Page eviction strategies:

Vocab:


Page Size: 4 KB
Physical RAM
Stack
P1 Virtual Memory Space
...
Page Table
For P1
0x393ac
Heap
Hard Disk
Used
...
Used
Used