IOWA STATE UNIVERSITY

Department of Electrical and Computer Engineering

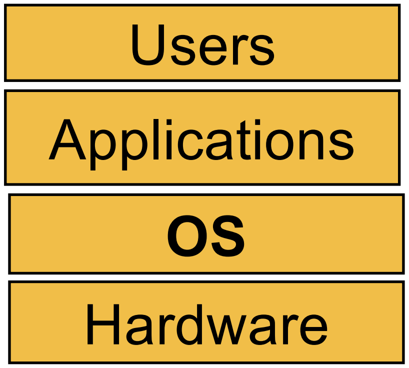
Lecture 17:
Memory Management

Agenda

- Recap
- Introduction to Memory Management
  - Early Systems
  - Virtual Memory

Recap

- OS
  - a resource manager
  - a control program
  - an extended/virtualized machine with abstraction
- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files



Users

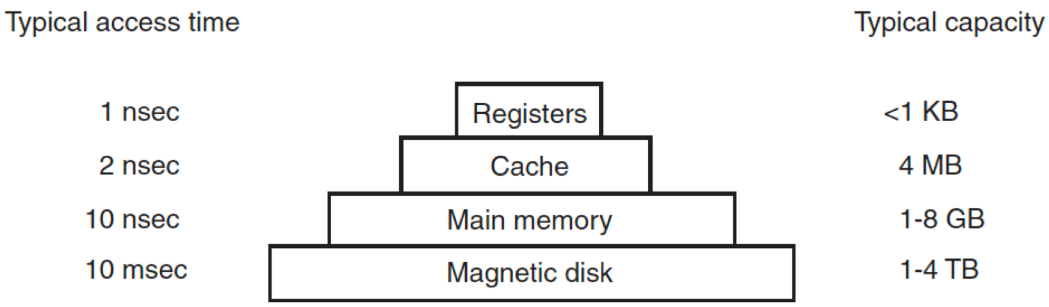
Applications

OS

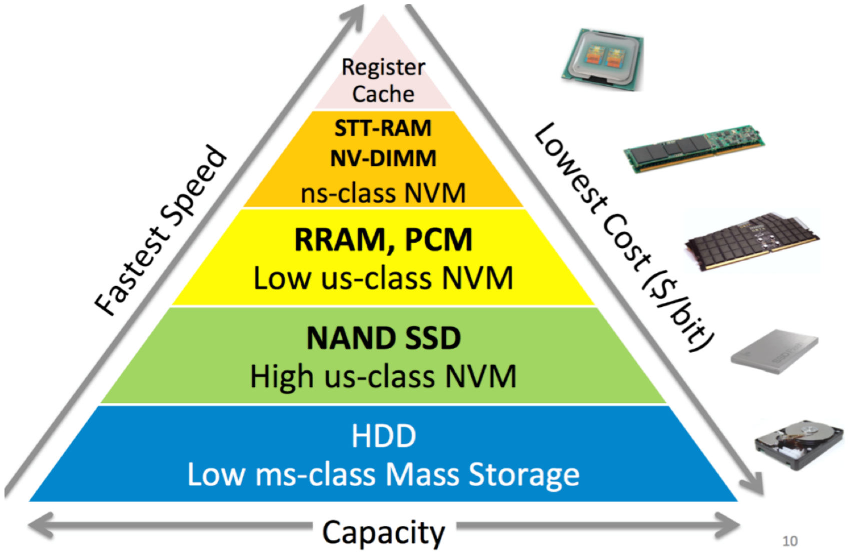
Hardware

Recap

- Memory Hierarchy
  - diverse technologies with tradeoffs
    - latency, capacity, persistency, cost, …
    - non-volatile memories are revolutionizing the market!
      - A “disruptive” technology



Typical access time		Typical capacity
1 nsec	Registers	<1 KB
2 nsec	Cache	4 MB
10 nsec	Main memory	1-8 GB
10 msec	Magnetic disk	1-4 TB



Register Cache

STT-RAM NV-DIMM ns-class NVM

RRAM, PCM Low us-class NVM

NAND SSD High us-class NVM

HDD Low ms-class Mass Storage

Fastest Speed

Lowest Cost ($/bit)

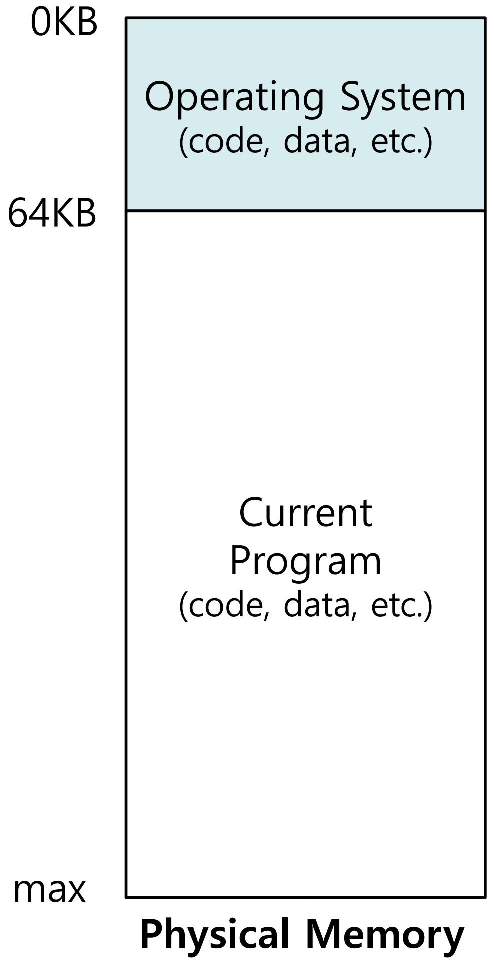
Capacity

Agenda

- ~~Recap~~
- Introduction to Memory Management
  - Early Systems
  - Virtual Memory

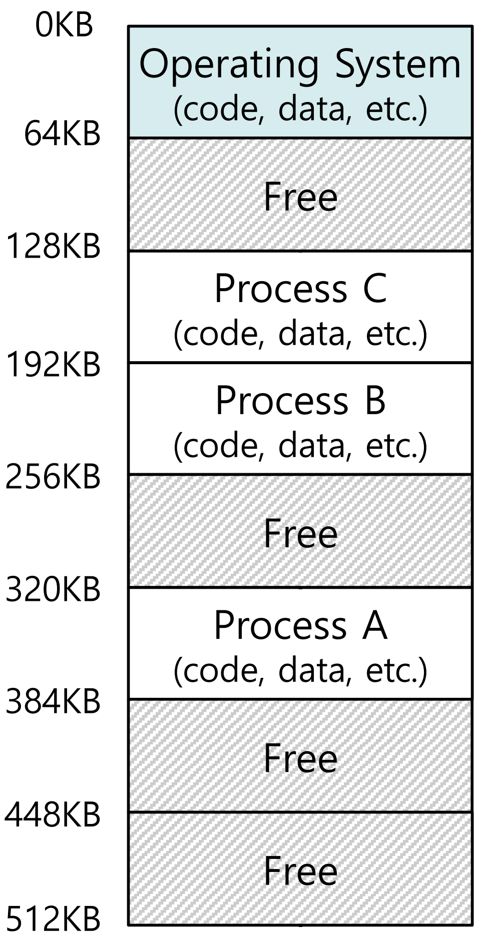
Memory Management in Early Systems

- Load only one program in memory
  - poor utilization/efficiency



Memory Management in Early Systems

- Multiprogramming
  - Execute each for a while
  - Switch processes between them in memory
  - Increase utilization and efficiency



0KB

Operating System (code, data, etc.)

64KB

Free

128KB

Process C (code, data, etc.)

192KB

Process B (code, data, etc.)

256KB

Free

320KB

Process A (code, data, etc.)

384KB

Free

448KB

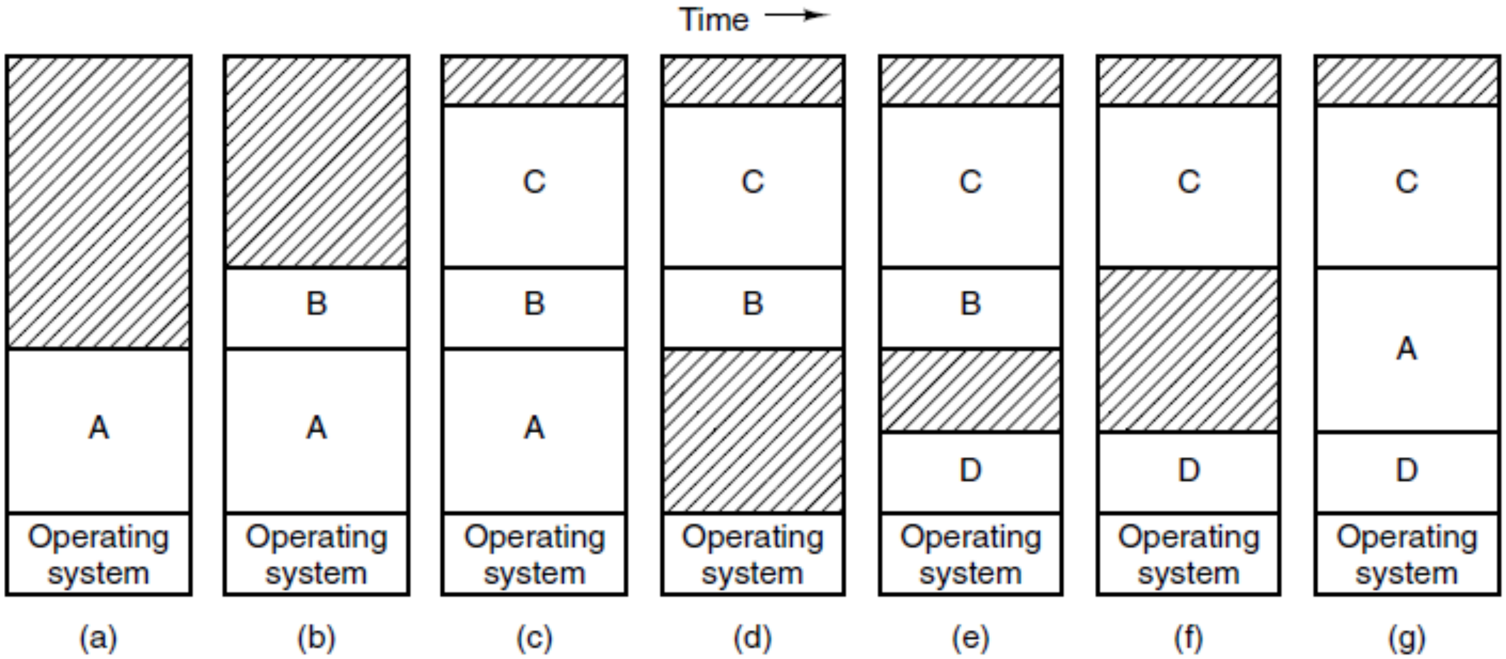
Free

512KB

Physical Memory

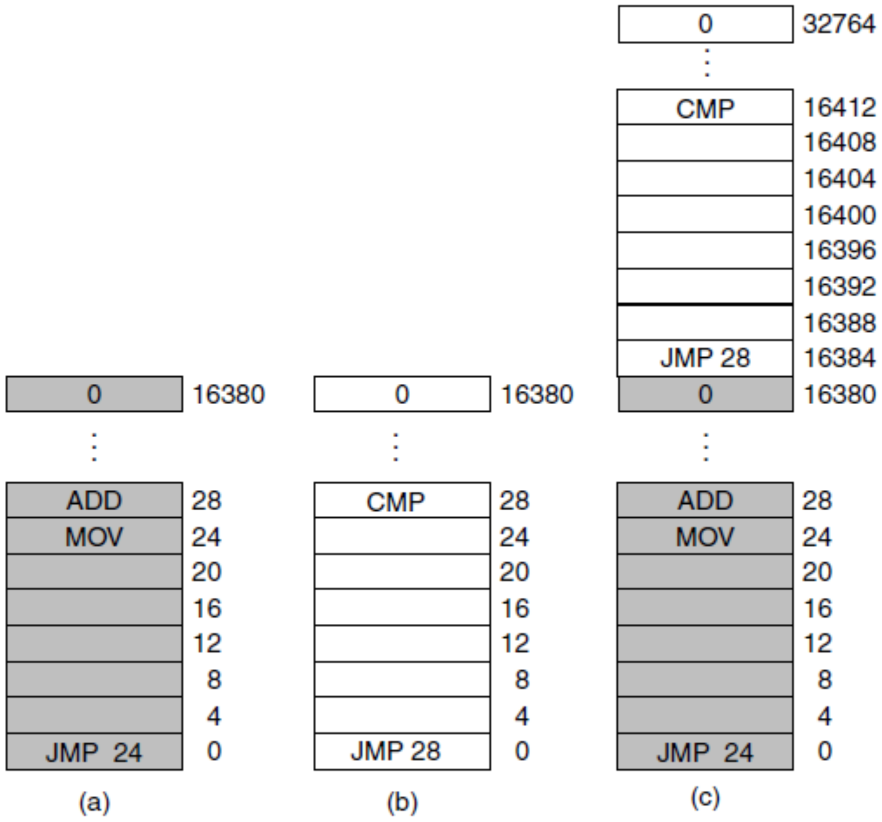
Memory Management in Early Systems

- Multiprogramming
  - Memory allocation changes as processes come and leave
    - the shaded regions are unused memory



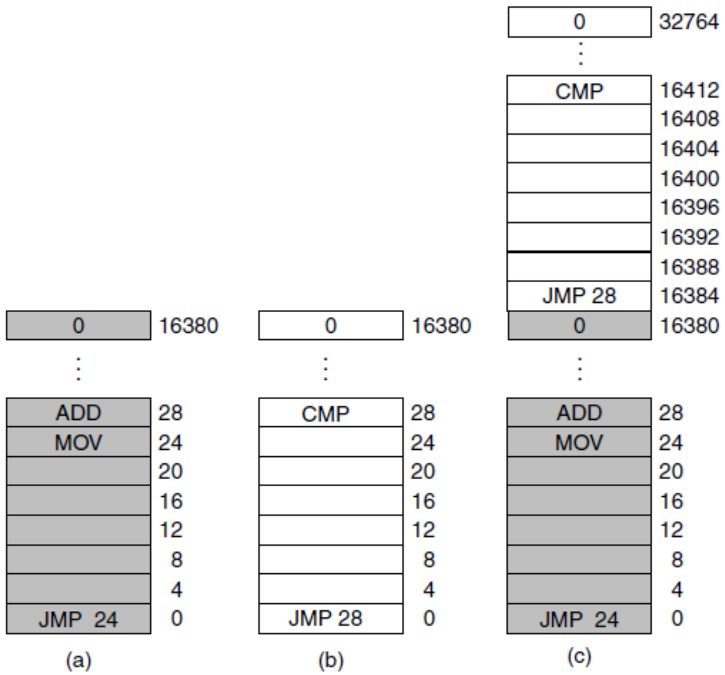
Memory Management in Early Systems

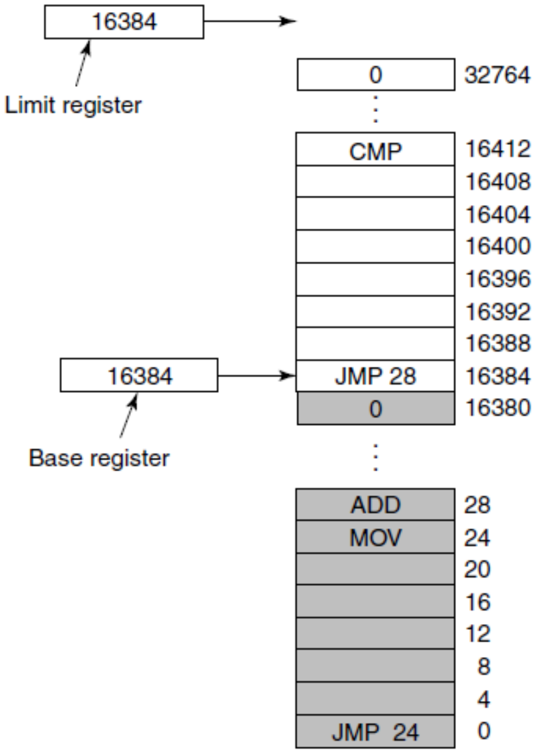
- Multiprogramming
  - Relocation problem



Memory Management in Early Systems

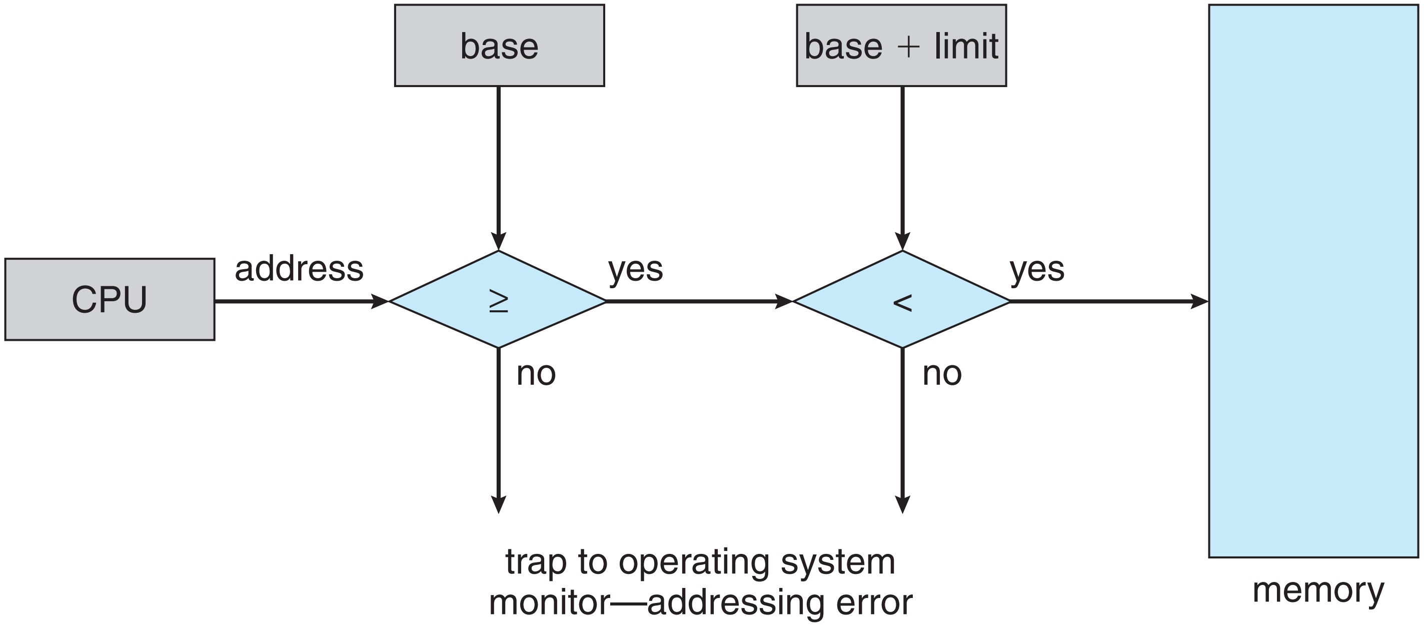
- Multiprogramming
  - Relocation using a pair of *base* and *limit* registers





Memory Management in Early Systems

- Multiprogramming
  - Basic protection via base/limit registers
    - CPU checks every memory address generated in user program to be sure it is between base and limit for that user



Agenda

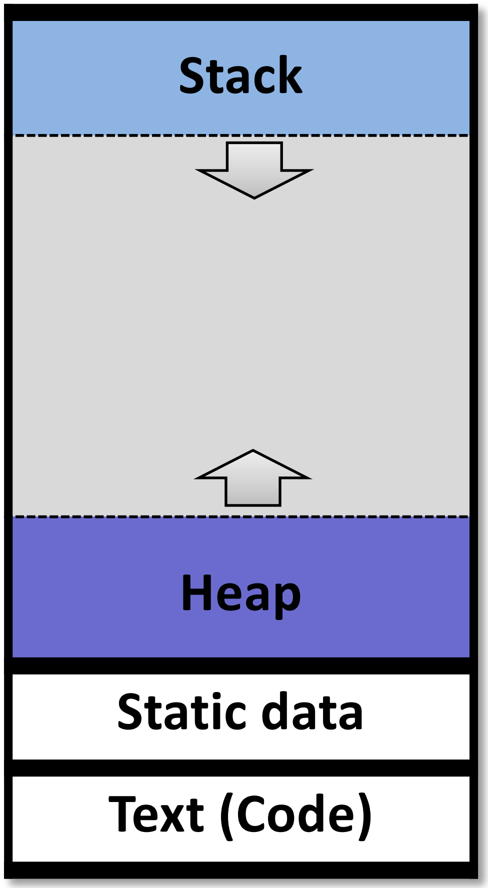
- ~~Recap~~
- Introduction to Memory Management
  - ~~Early Systems~~
  - Virtual Memory

Virtual Memory

- Modern OS provides a virtual memory for each process
  - provides the illusion that each process uses the whole memory itself
  - can handle programs that are too large to fit in physical memory
- Benefits
  - memory efficiency
  - isolation & protection among processes

Address Space

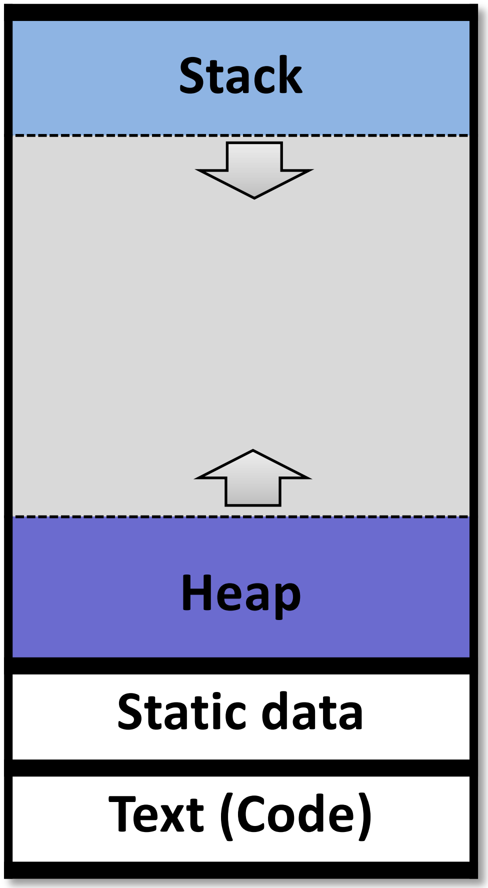
- An abstraction of physical memory for a process
  - the set of all virtual addresses visible to a program
  - hold all info. about a running process
  - consist of different regions
    - stack
    - heap
    - static data
    - code (text)



Address Space

Virtual Address

- Every byte in the address space has a unique virtual address
  - every address in a running program is a virtual address



Address Space

Virtual Address

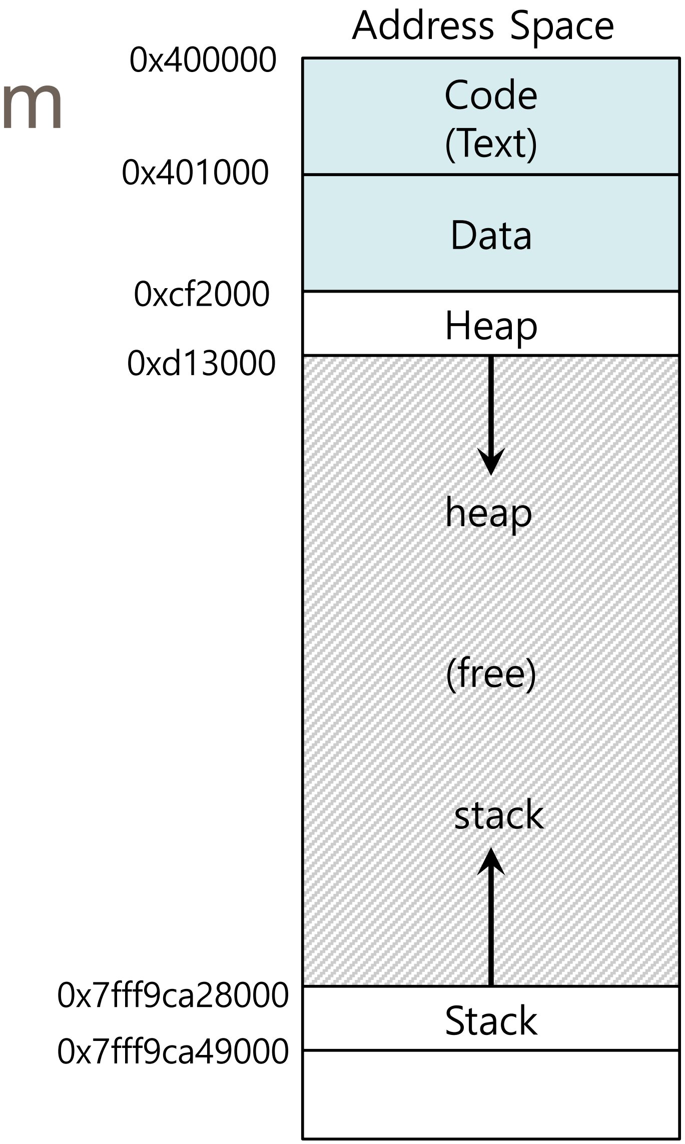
- E.g., virtual address in a program

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]){

    printf("location of code  : %p\n", (void *) main);
    printf("location of heap  : %p\n", (void *) malloc(1));
    int x = 3;
    printf("location of stack : %p\n", (void *) &x);

    return x;
}
```

Virtual Address

- E.g., virtual address in a program
  - output on a 64-bit Linux machine

```
location of code  : 0x40057d
location of heap  : 0xcf2010
location of stack : 0x7fff9ca45fcc
```



Address Translation

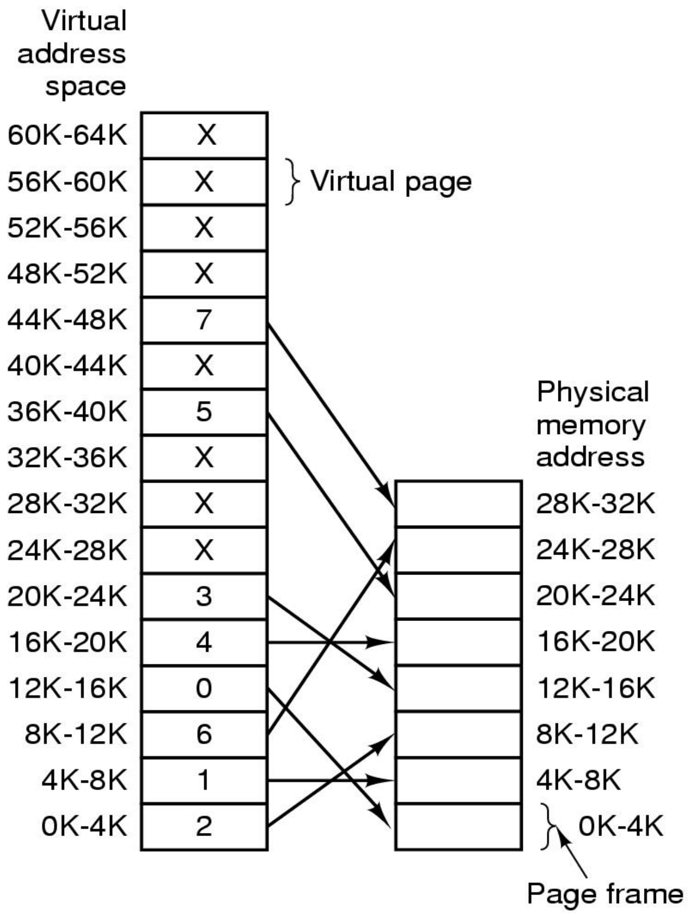
- Translate virtual address to physical address
- Static address translation
  - Map virtual address to physical address when loading a process into memory
  - Used when processes are fixed and known in advance
    - e.g., many embedded systems
- Dynamic address translation
  - Map virtual address to physical address at runtime

Paging

- Key technique of virtual memory management
  - Divide the address pace of a program into fixed-size small parts
    - called ***virtual pages*** or ***pages***
    - typically 4KB each
  - Divide the physical memory into fixed-size small parts
    - called *page frames*
    - typically 4KB each
  - Only load necessary (virtual) pages into (physical) page frames

Paging

- E.g.,
  - 64KB Virtual Address Space
    - divided into 16 pages
    - 4KB each page
  - 32KB Physical Memory
    - 8 page frames
    - 4KB each



Virtual address space	
60K-64K	X
56K-60K	X
52K-56K	X
48K-52K	X
44K-48K	7
40K-44K	X
36K-40K	5
32K-36K	X
28K-32K	X
24K-28K	X
20K-24K	3
16K-20K	4
12K-16K	0
8K-12K	6
4K-8K	1
0K-4K	2

Virtual page

Physical memory address

28K-32K
24K-28K
20K-24K
16K-20K
12K-16K
8K-12K
4K-8K
0K-4K

Page frame

Agenda

- ~~Recap~~
- ~~Introduction to Memory Management~~
  - ~~Early Systems~~
  - ~~Virtual Memory~~

Questions?

*acknowledgement: slides include content from "Modern Operating Systems" by A. Tanenbaum, "Operating Systems Concepts" by A. Silberschatz etc., "Operating Systems: Three Easy Pieces" by R. Arpaci-Dusseau etc., and anonymous pictures from internet.