SEGMENTATION IMPLEMENTATION

MMU contains Segment Table (per process)

- Each segment has own base and bounds, and protection bits
- Example: 14 bit logical address, 4 segments.
- How many bits for segment?
- How many bits for offset?

Segment	Base	Bounds	R W
0	0x2000	0x6ff	1 0
1	0x0000	0x4ff	1 1
2	0x3000	0xfff	1 1
3	0x0000	0x000	0 0

remember:
1 hex digit->4 bits

SEGMENTATION IMPLEMENTATION

MMU contains Segment Table (per process)

- Each segment has own base and bounds, protection bits
- Example: 14 bit logical address, 4 segments.
- How many bits for segment? ANSWER: 2 (2^2 is 4)
- How many bits for offset? ANSWER: 12 (14 – 2 is 12)

Segment	Base	Bounds	R W
0	0x2000	0x6ff	1 0
1	0x0000	0x4ff	1 1
2	0x3000	0xfff	1 1
3	0x0000	0x000	0 0

remember:
1 hex digit->4 bits

Segmentation Implementation

How address translation is done with segmentation:

1. MMU extracts segment number from msbs of logical/virtual address
2. MMU accesses base register for that segment to get base address
3. MMU compares offset from logical/virtual address to bounds register for segment; if less, address is valid, but if offset >= bounds, invalid address (segmentation fault) and CPU will call OS to terminate process

Quiz: Address Translations with Segmentation

MMU contains Segment Table (per process)

- Each segment has own base and bounds, protection bits
- Example: 14 bit logical address, 4 segments; how many bits for segment? How many bits for offset?

Segment	Base	Bounds	R W
0	0x2000	0x6ff	1 0
1	0x0000	0x4ff	1 1
2	0x3000	0xfff	1 1
3	0x0000	0x000	0 0

remember:
1 hex digit->4 bits

Translate logical addresses (in hex) to physical addresses

0x0240:

0x1108:

0x265c:

0x3002:

Quiz: Address Translations with Segmentation

MMU contains Segment Table (per process)

- Each segment has own base and bounds, protection bits
- Example: 14 bit logical address, 4 segments; how many bits for segment? How many bits for offset?

Segment	Base	Bounds	R W
0	0x2000	0x6ff	1 0
1	0x0000	0x4ff	1 1
2	0x3000	0xfff	1 1
3	0x0000	0x000	0 0

remember:
1 hex digit->4 bits

Translate logical addresses (in hex) to physical addresses

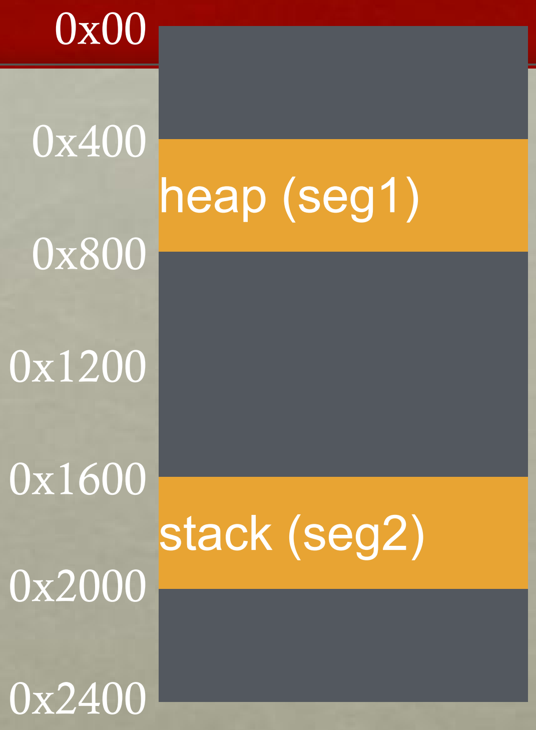
0x0240: Physical address is: 0x2000 + 0x240 = 0x2420

0x1108: Physical address is: 0x0000 + 0x108 = 0x0108

0x265c: Physical address is: 0x3000 + 0x65c = 0x365c

0x3002: Physical address is: 0x0000 + 0x002 = 0x0002; SEGFAULT (R and W both 0)

VISUAL INTERPRETATION



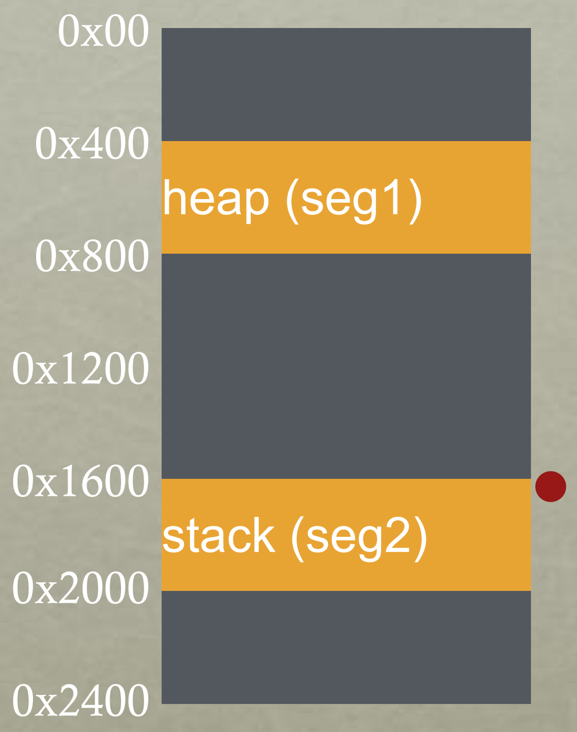
Virtual (hex)	Physical
load 0x2010, R1	

Segment numbers:

0: code+data

1: heap

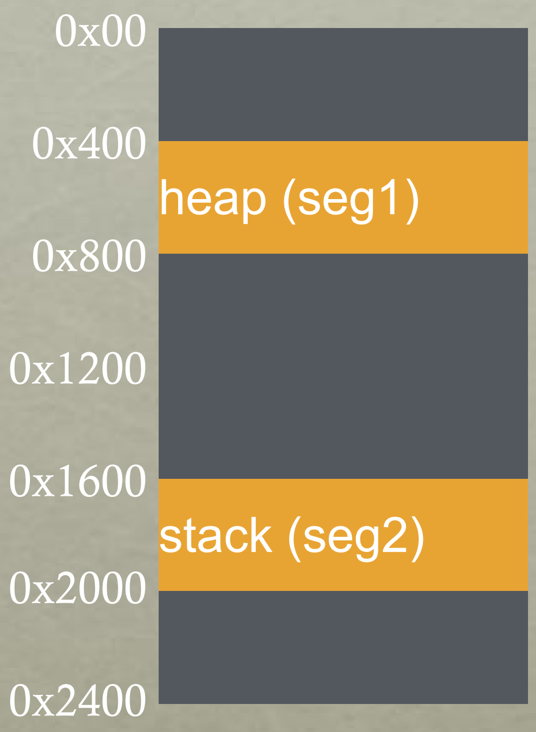
2: stack



Virtual (hex)	Physical
load 0x2010, R1	0x1600 + 0x010 = 0x1610

Segment numbers:

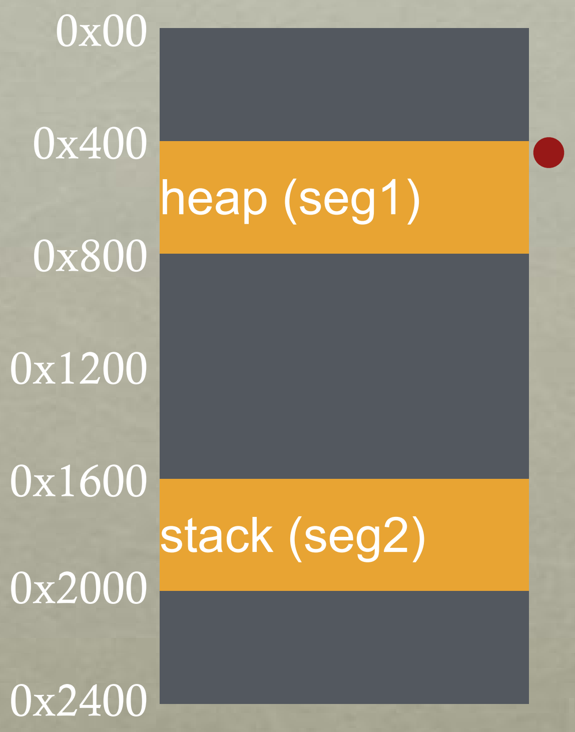
0: code+data
1: heap
2: stack



Virtual (hex)	Physical
load 0x2010, R1	0x1600 + 0x010 = 0x1610
load 0x1010, R1	

Segment numbers:

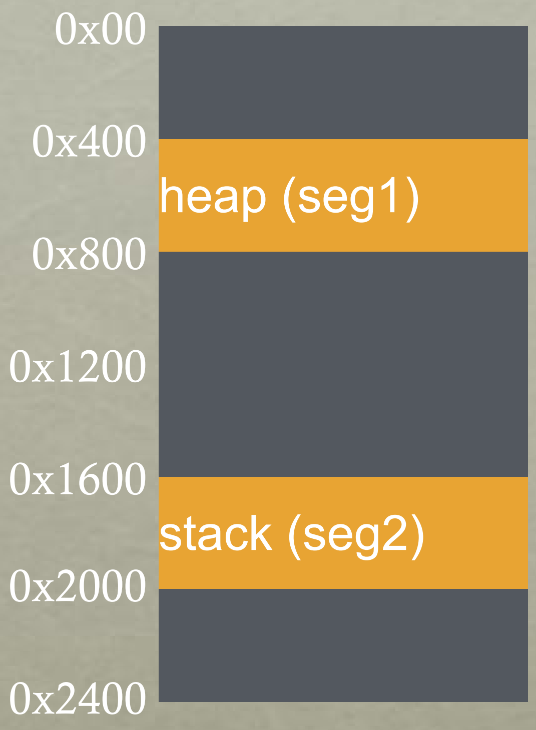
0: code+data
1: heap
2: stack



Virtual (hex)	Physical
load 0x2010, R1	0x1600 + 0x010 = 0x1610
load 0x1010, R1	0x400 + 0x010 = 0x410

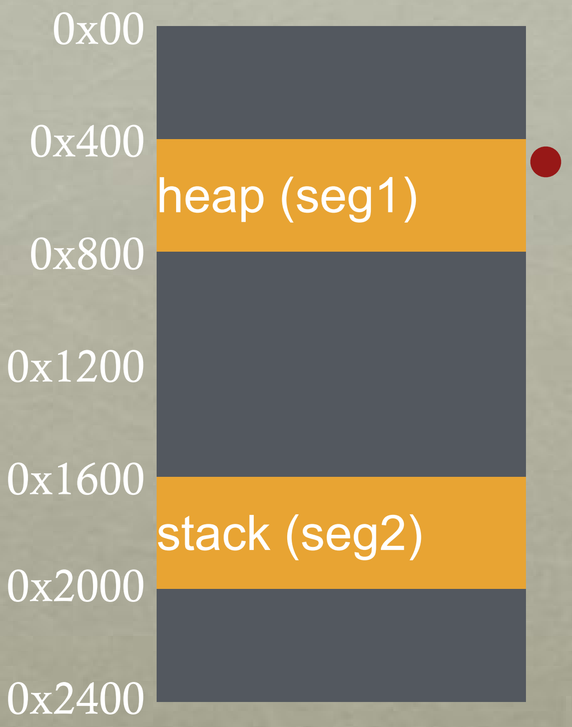
Segment numbers:

0: code+data
1: heap
2: stack



Virtual	Physical
load 0x2010, R1	0x1600 + 0x010 = 0x1610
load 0x1010, R1	0x400 + 0x010 = 0x410
load 0x1100, R1	

Segment numbers:
0: code+data
1: heap
2: stack



Virtual	Physical
load 0x2010, R1	0x1600 + 0x010 = 0x1610
load 0x1010, R1	0x400 + 0x010 = 0x410
load 0x1100, R1	0x400 + 0x100 = 0x500

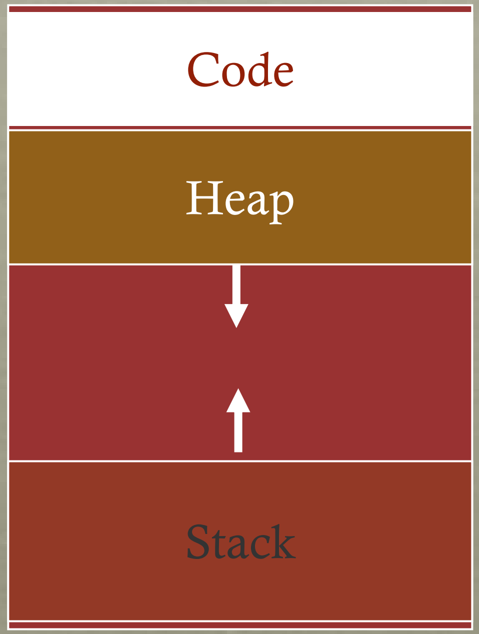
Segment numbers:
0: code+data
1: heap
2: stack

ADVANTAGES OF SEGMENTATION

- Enables sparse allocation of address space
  - Stack and heap can grow independently
  - Heap: If no memory space on free list, dynamic memory allocator requests more from OS
  - Stack: OS recognizes reference outside legal segment, extends stack implicitly (by increasing bounds register value, and moving stack segment if necessary to different part of memory
- Different protection for different segments
  - Read-only status for code
- Enables sharing of selected segments (two processes can have same

base and bounds values for a shared segment)

- Supports dynamic relocation of each segment



Disadvantages of Segmentation

Each segment must be allocated contiguously (from beginning to end); segments cannot be subdivided into smaller pieces

- May not have sufficient free/available physical memory for large segments

Fix in next slide set with paging…

CONCLUSION

HW+OS work together to virtualize memory

- Give illusion of private address space to each process, even though in fact, multiple processes share memory

Add MMU registers for base+bounds so translation of addresses is fast

- OS not involved with every address translation, only involved on context switch or errors

Dynamic relocation with segments is good building block

- Next class: Solve fragmentation (we will see definition of this) with paging