11/13/16

Segmentation ~~like paging with diff size frames~~

can have diff sized segments (~~like frames in paging~~)

like paging logical address has 2 parts

|— logical —|

$\underbrace{010}_{\text{segt}} \quad \underbrace{00001}_{\text{offset}}$

diff is the segment table

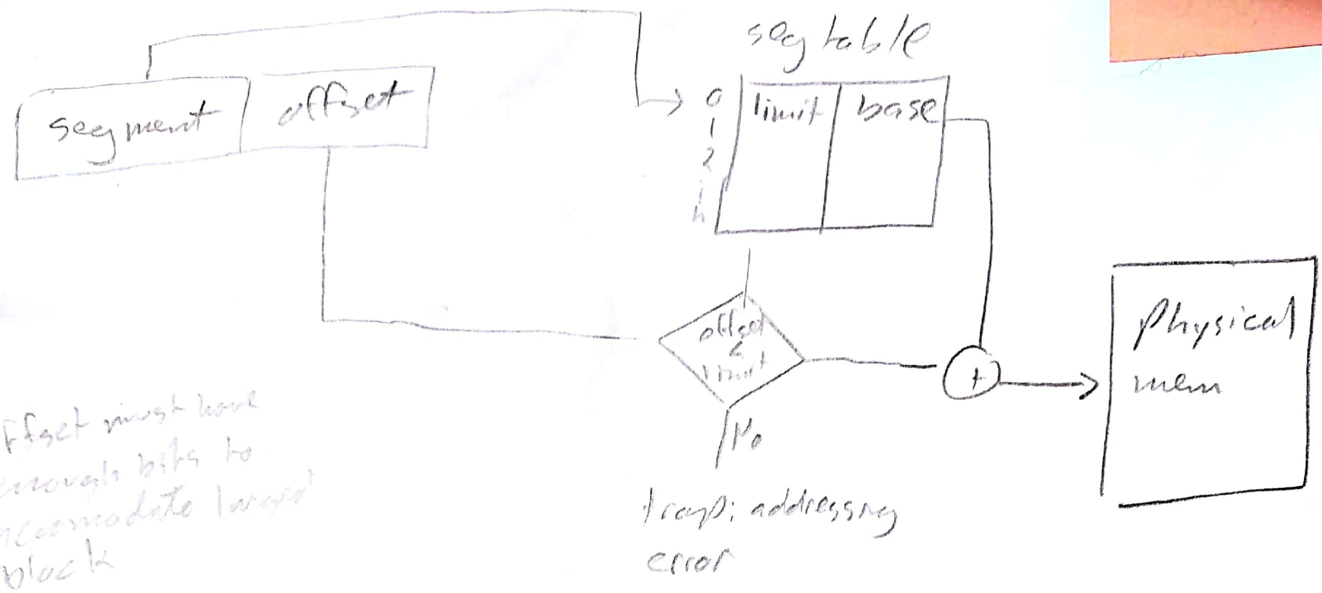
has base & length
↑ like paging
↑ max size

- these segments map to blocks of memory
- no internal frag. (external though) best f.t? first f.t?
- compilers can control how process is segmented

(data, program)

bit more complex than simple paging.

still all process in mem at once

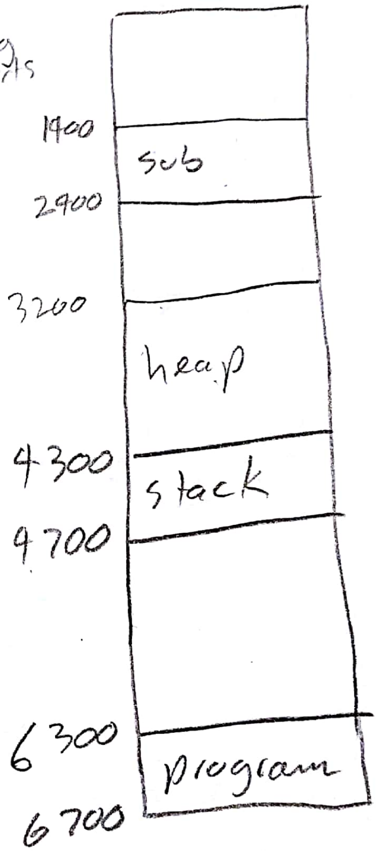


offset must have enough bits to accomodate largest block

Seg #	
3	Heap
2	stack
1	program
0	subroutine

blocks are diff sizes
user defined prog parts

	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200



also bits to determine read, write, dirty ww.cs.columbia.edu/~junfeng/10sp-w4118/lectures/...

11/13/16 ①

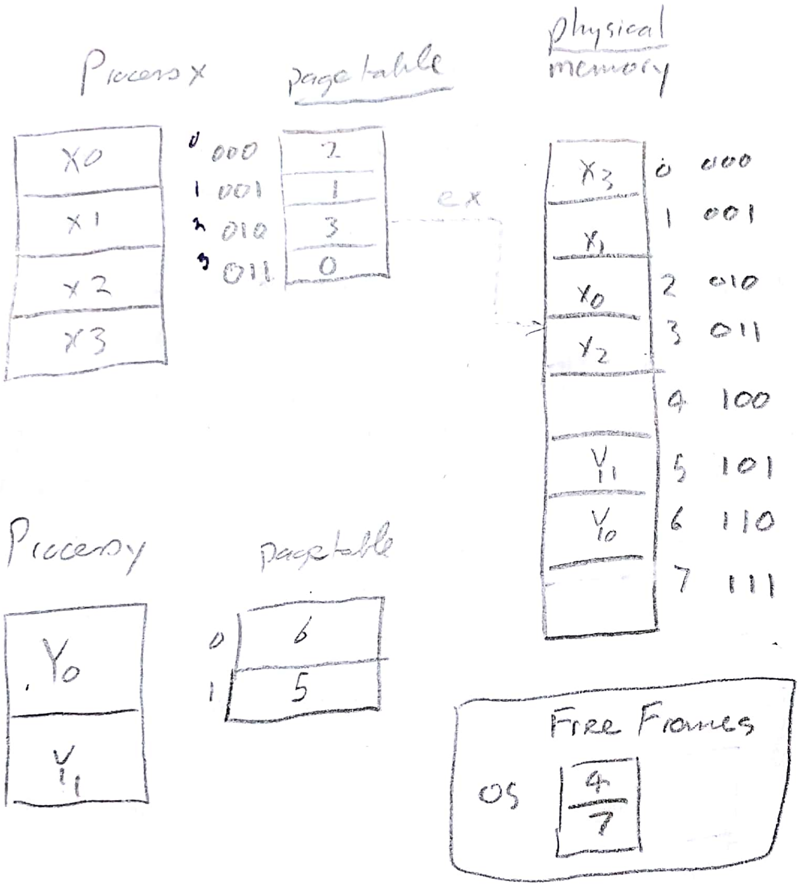
paging

16 bits 6 for frame # then have 2^6 frames each with 2^{10} words
64 1024

Process page = frame size

slide 37 – 41

each page table kept with process control block



say have 8 bit address space

have 8 frames (3 bits)

frames have $2^5 = 32$ words

but still need all of process in mem at once!

also internal frag on last block of process.

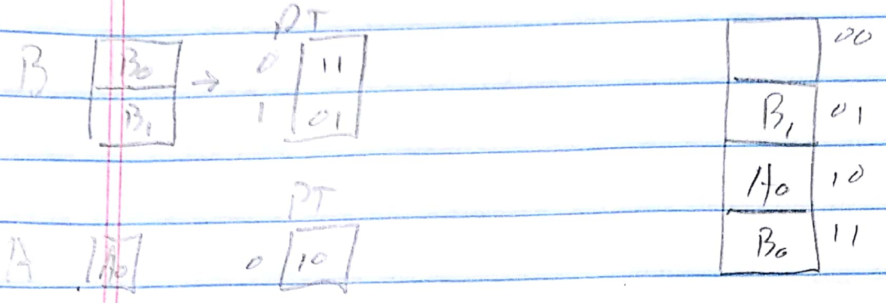
ex 8 bit logical address for process X

logical

$\underbrace{010}_{\text{page\#}}\ \underbrace{00001}_{\text{offset}}$

ex. 5 bits 2 add 3 data

$2^2 = 4$ blocks 2^3 8 rows block



free blocks
0, ~~1~~, ~~2~~, ~~3~~

what if A had 3 blocks? Next chapter, virtual

have a system virtual address is 00|1111

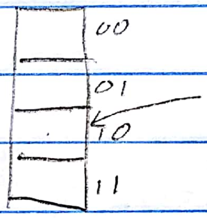
physical address is 10|1111

what can you tell me?

- really need block size but at least 2 frame# bits

OK block size = 2

Know this much

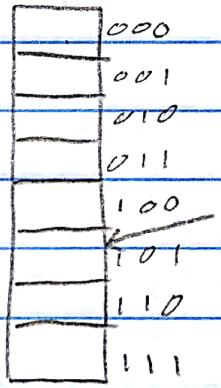


physical address is here

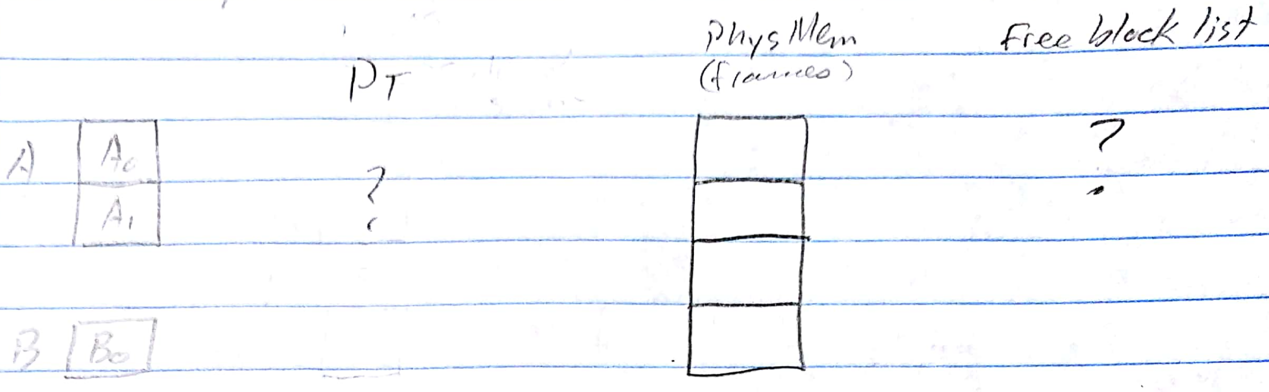
program has at least 1 block (because virtual is 00)

what if 101111? then prog has at least 3 blocks so 2 other blocks occupied

block size = 3

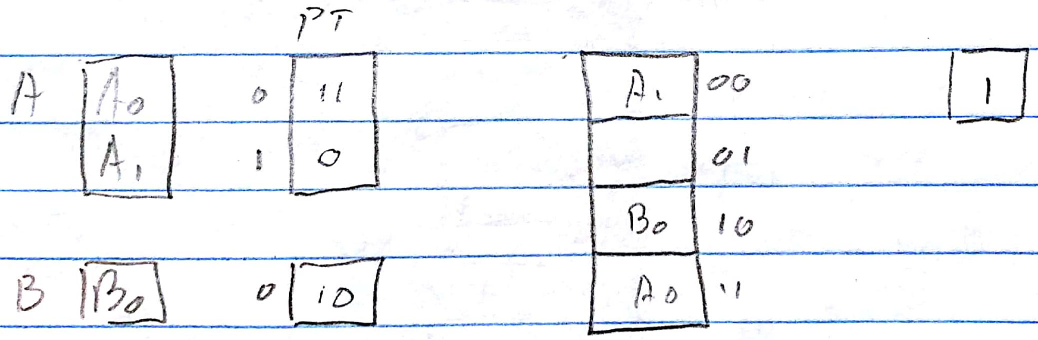


what if you have this



	virtual address	Physical Address
A	011111	001111
A	001111	111111
B	001111	101111

show where process pages are in physical mem.
show PT for each process
show free block list
determine block size (# page bits) & # address bits


PT
A
A0
A1
0
11
1
0
B
B0
0
10
A1
00
01
B0
10
A0
11
1