Content borrowed from www.cs.columbia.edu/~junfeng/10sp-w4118/lectures/...

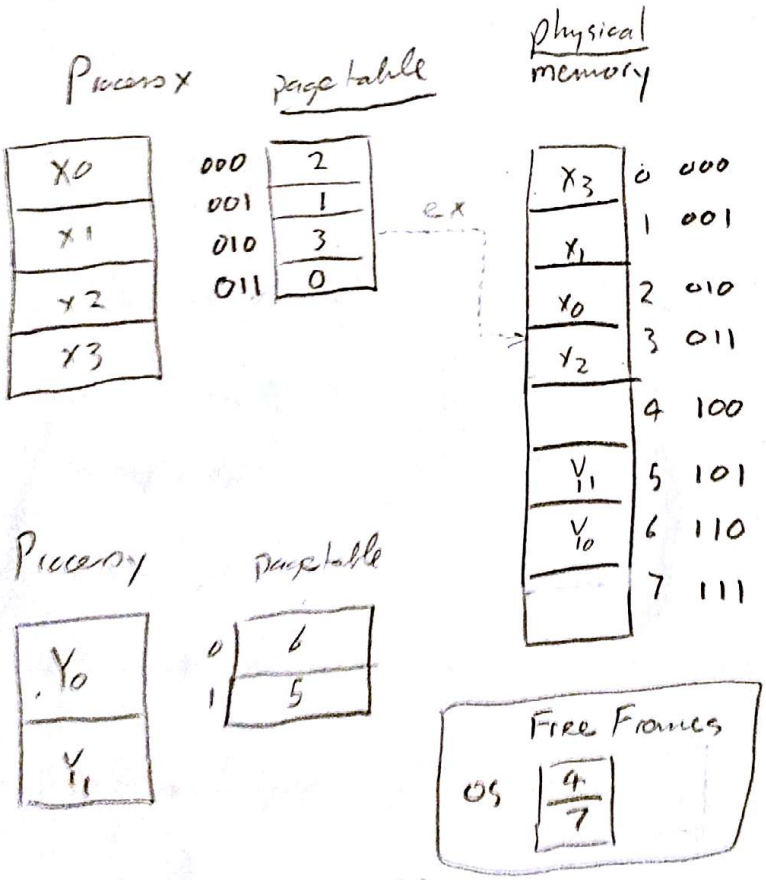
Paging

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

Process page = frame size

slide 37 – 41

each page table kept with process control block



say have 8 bit address space

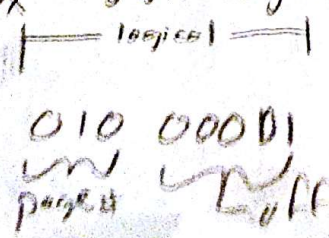
have 8 frames (3 bits)

frames have $2^5 = 32$ words

~~but still need all of process in mem at once!~~

~~also internal frag on last block of process.~~

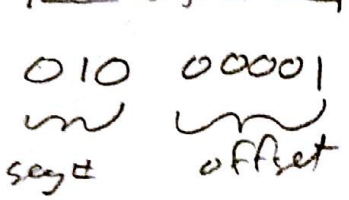
ex 8 bit logical address for process X



Segmentation like ~~paging with diff size frames~~

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

like paging logical address has 2 parts



diff is the segment table

has base + length

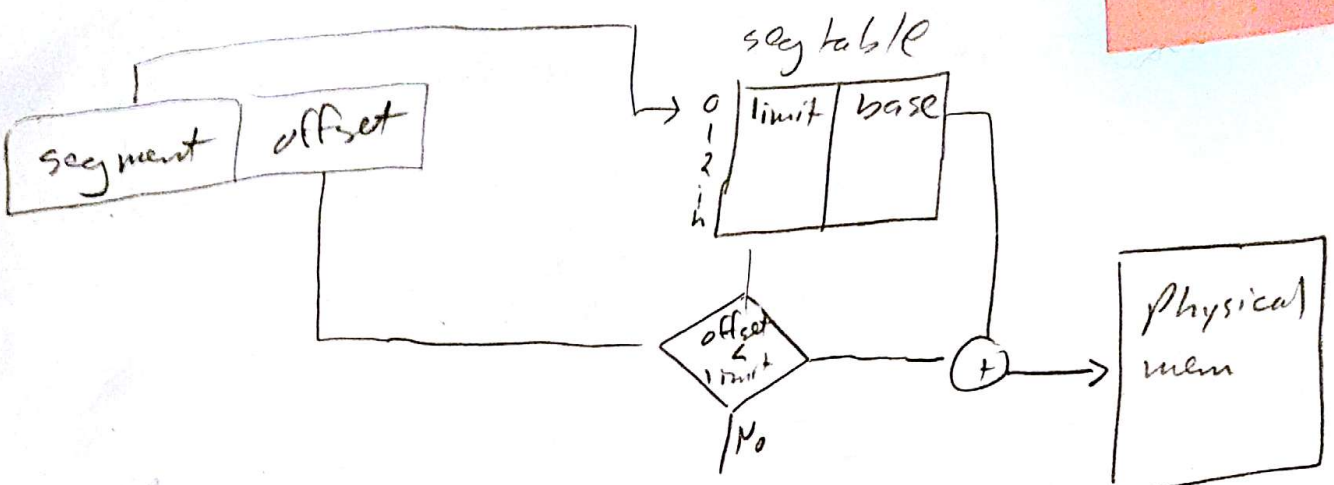
↑ like paging

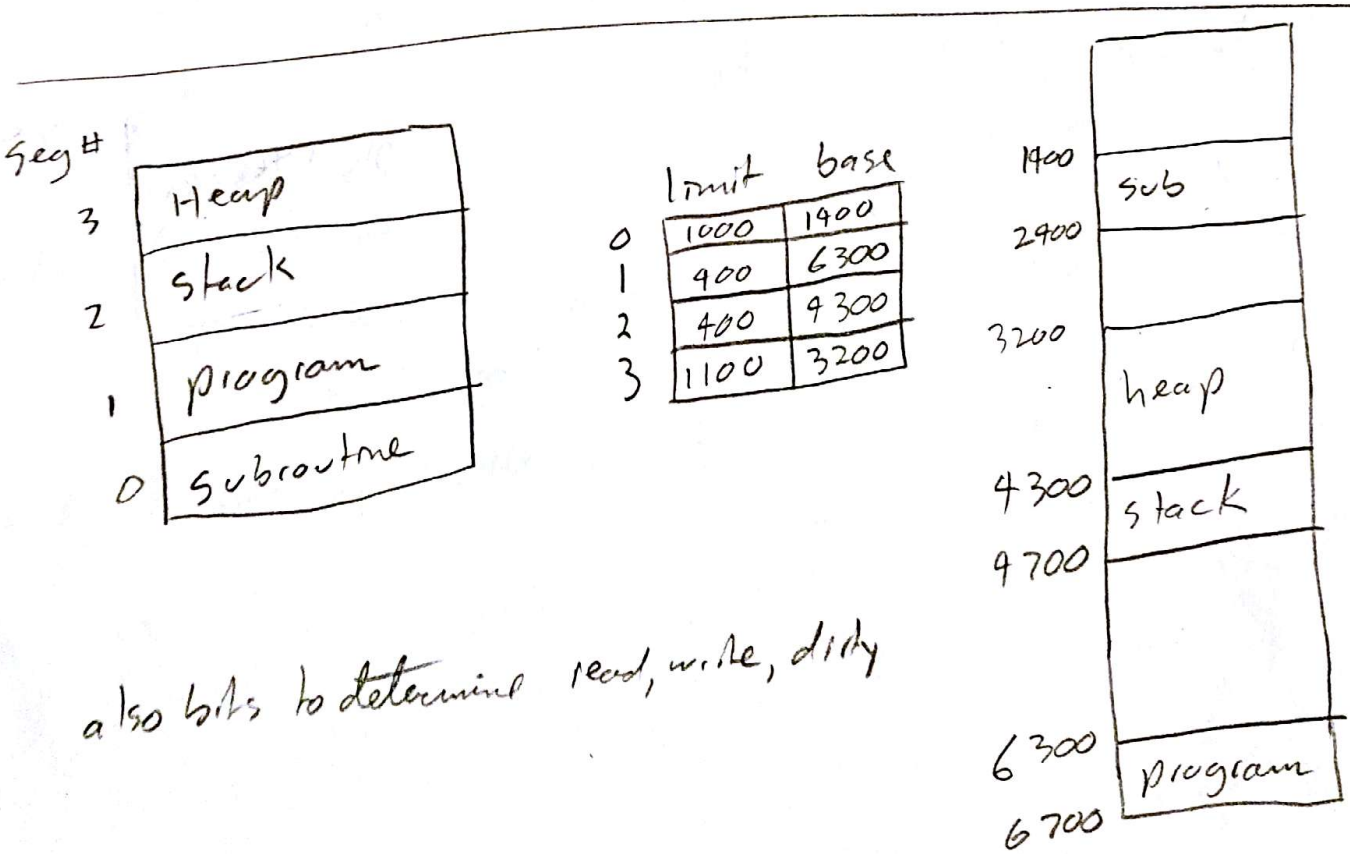
↑ max size

- these segments map to blocks of memory
- no internal frag. (external though) best fit? first fit?
- compilers can control how process is segments (data, program)

bit more complex than simple paging.

~~still all process in mem at once~~





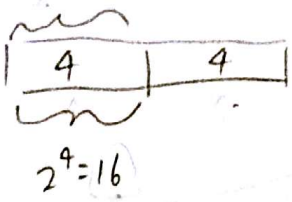
also bits to determine read, write, dirty

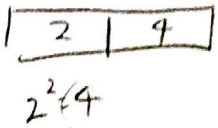
Paging

6 bit logical address 8 bit physical address

each frame is 16 bytes

① how many page table entries
② what if have 8 bit logical address? 16





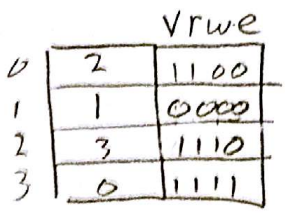
③ how big physical pages? $2^4 = 16$ entries/frame
④ how many logical (6 bits) $2^2 = 4$ (8 bits) $2^4 = 16$

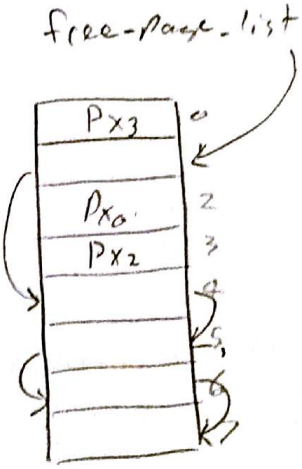
---

add some bits to page table

valid bit: map to a valid physical page
read/write/execute bits

all checked by MMU on each mem access





can allocate from free-page-list. one page at a time from head of this list.