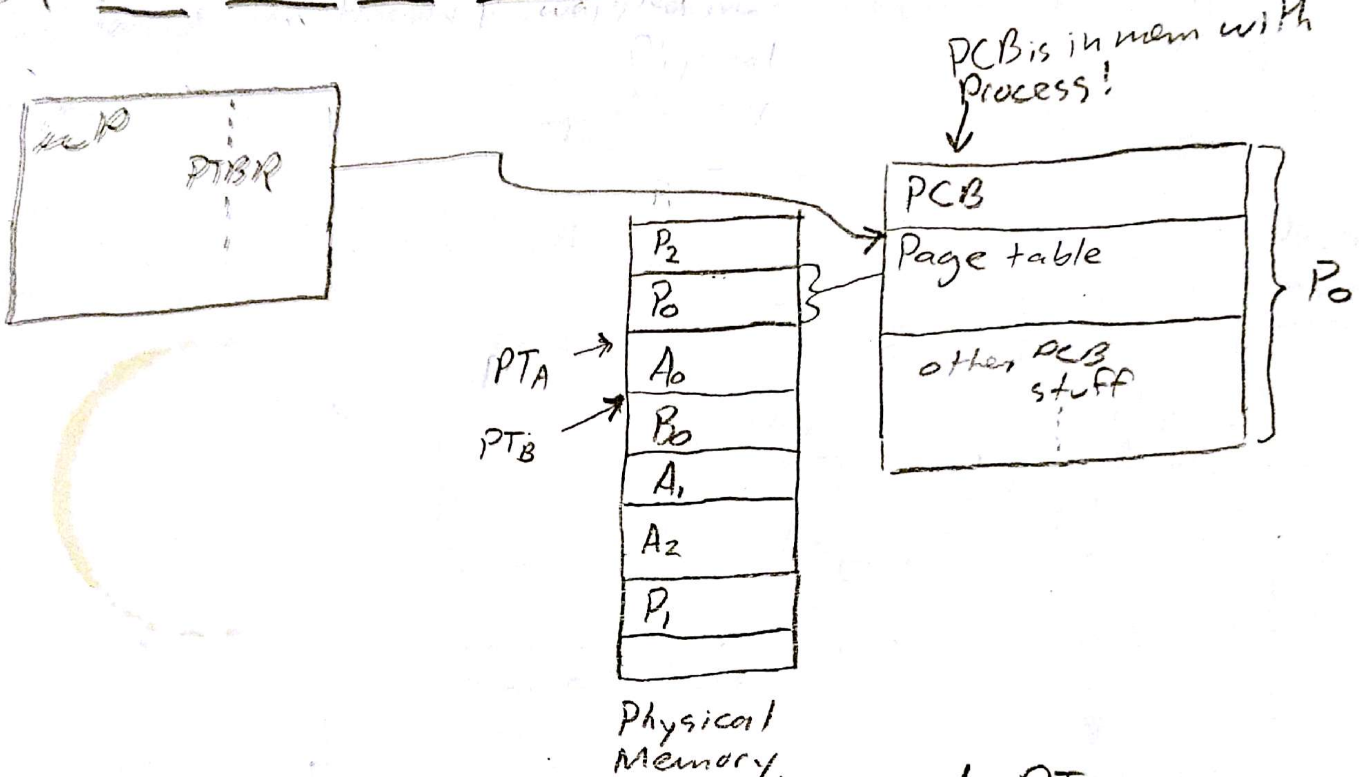
(How Processes stored & accessed in mem?)

}Its in the PCB (part of mem but how is it used?)

PTBR - page table base register. Points to page table in mem.



- OS will switch PTBR to point to correct PT on Process swap

- Get to stripe your program all over memory

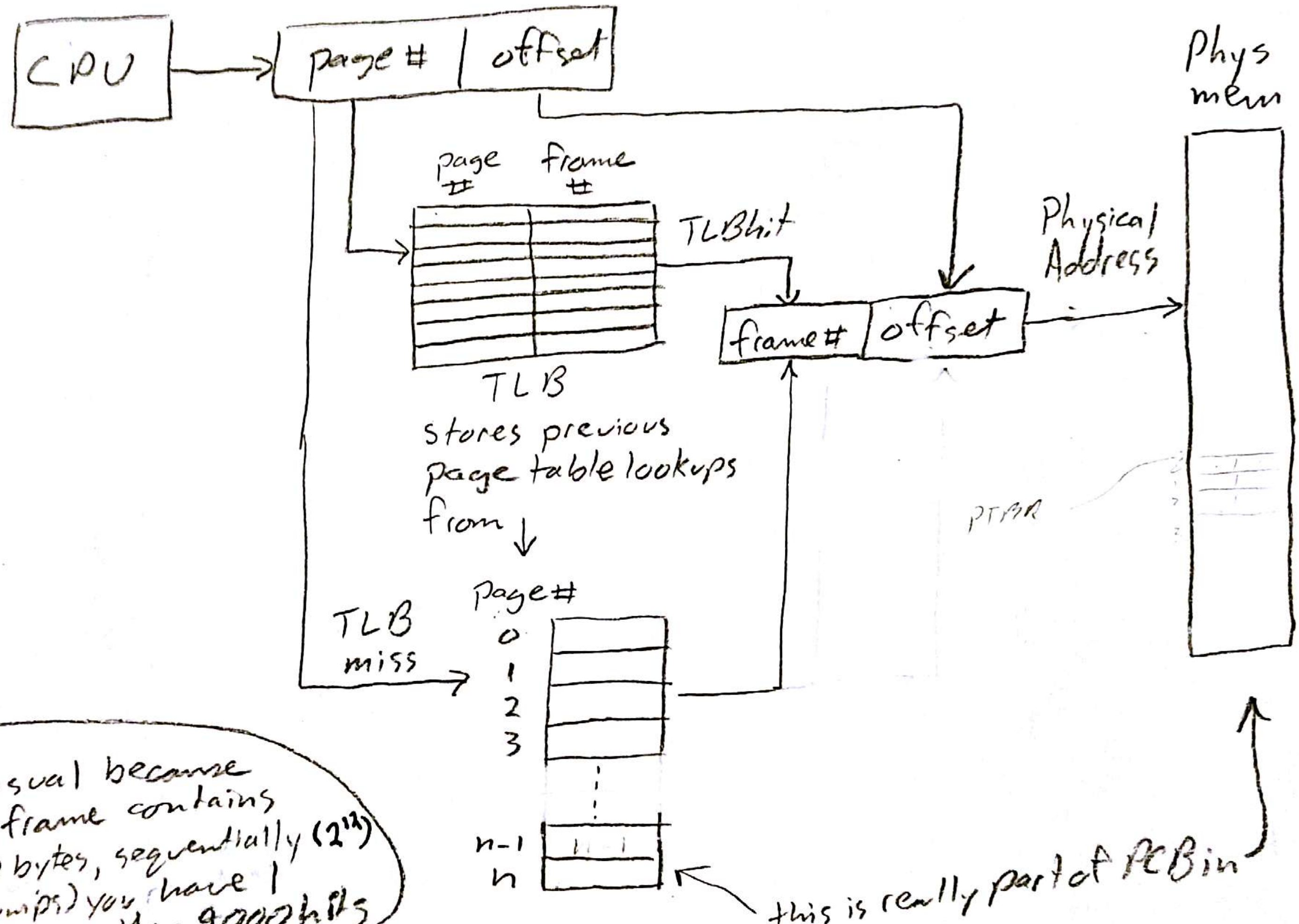
tradeoff is 2 memory accesses (PT lookup & then Physical address build then physical address lookup

Lets see if we can fix that Translation Lookaside Buffer (TLB)

Fast Cache (often on chip) <u>much</u> faster than memory

stores recent translations of logical address → physical address

Small



Its usual because each frame contains ~4,000 bytes, sequentially (2^{12}) (no jumps) you have 1 TLB miss & then 4000 hits

TLB hit (usual) get fast frame #

TLB miss (unusual) get it from mem (page table of process n mem

Thats nice! what happens when context switches?

1. Flush entire TLB & start anew (x86) makes sense since all TLB entries are for switched out process

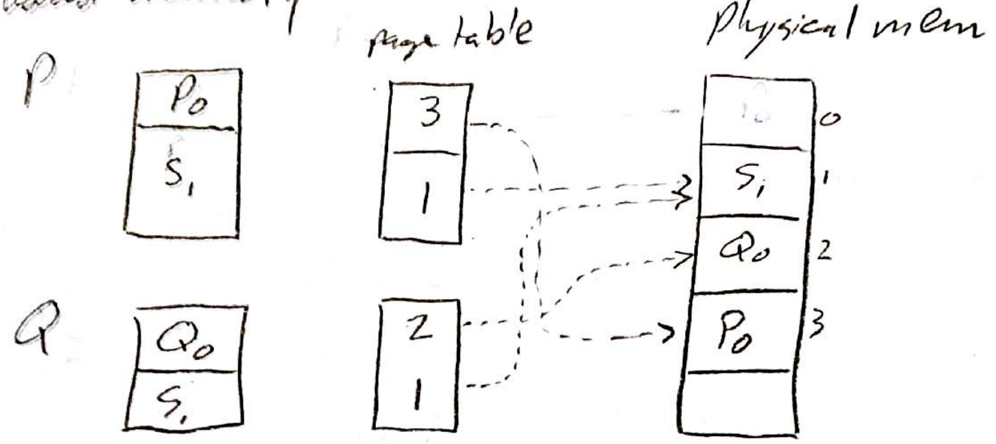
2. Attach process ID to each TLB entry
   - requires additional space (Process ID)
   - only useful if have TLB that is mostly empty

Page Sharing

Threads can share host process memory to communicate

What about processes?

Map shared memory



Page tables are great but... they are too big (I have not shown full PT

- 32 bit address space (4 billion) = 2^{32}
- 4 Kb per page = 2^{12}
- have $2^{32}/2^{12} = 2^{20}$ page table entries (= 1,000,000)
- each row 2.5 bytes / 2.5 bytes = 5 bytes
  20 bits 20 bits virtual physical
- page table size 5 Mbytes - kinda large for something that is very sparse.

stopped here 11/16/16 first class

fix1 - Hierarchial paging (multilevel)

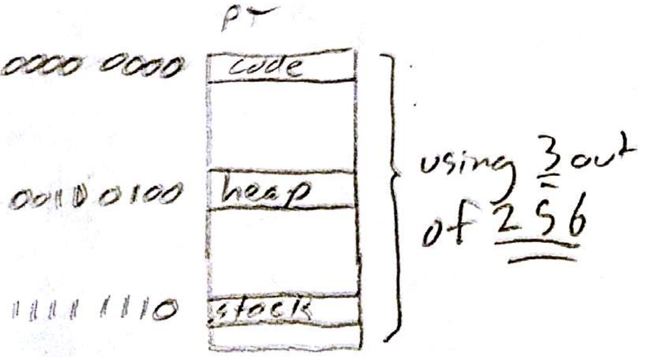
break PT into multilevel PT's

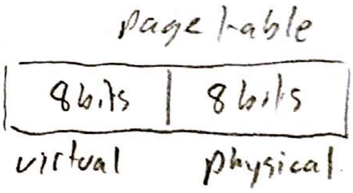
ex. 14 bits mem => $2^{14} = 64K$

64 bytes/frame => $2^6 = 64$

page table entries => $2^{14}/2^6 = 2^8 = 256$ pages

size PT = 2 bytes x 256 pages = 500 bytes





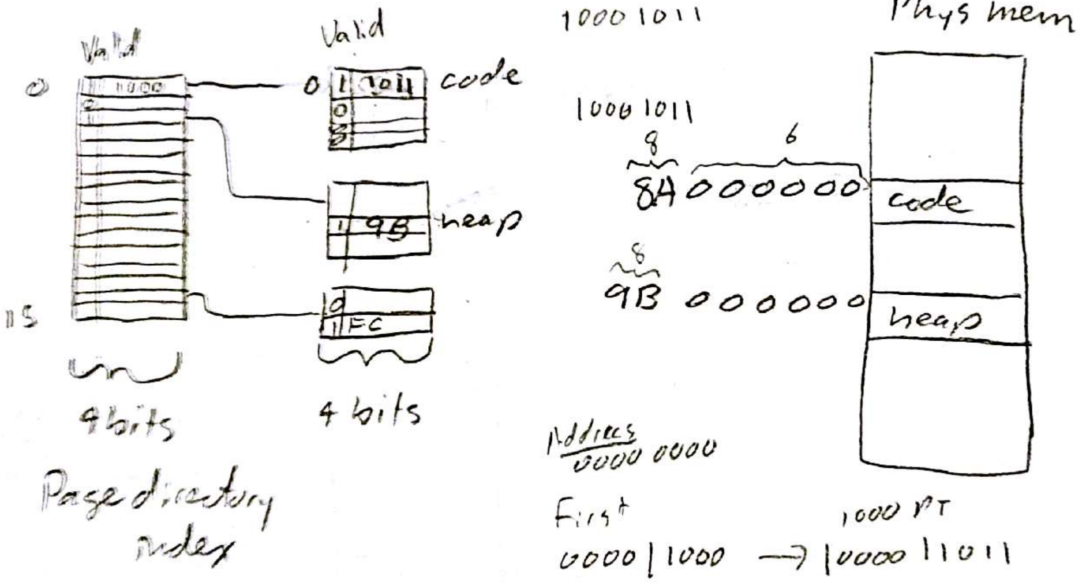
Break page table into page sized units (64 bytes/page)

256 entries * 2 bytes/entry = 512 bytes = 2^9

if break into 64 byte sized chunks

$2^{10}/2^6 = 16$

can divide initial 256 entry PTE into 16 entries



gone from needing 16 pages to 4

---

for 32 bit addressing

go from needing 1,000,000 to 4

---

disadvantage - 3 lookups for each memory access