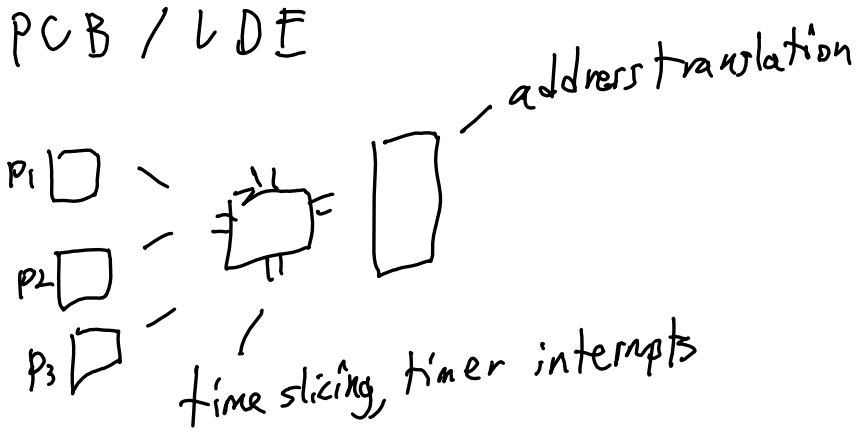
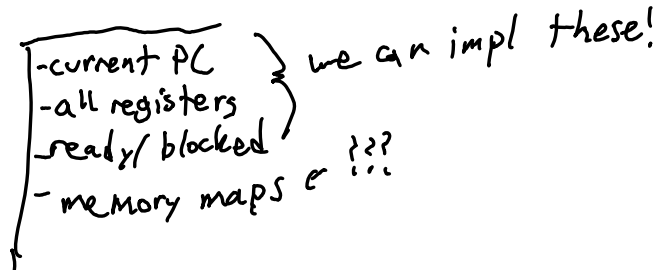


recall: PCB / LDE

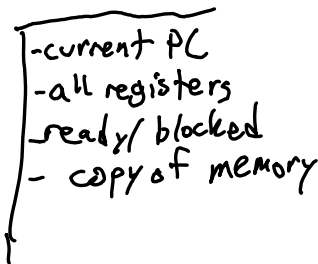


Process control block (PCB)



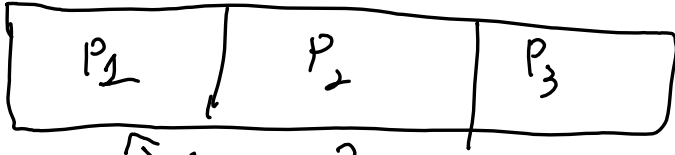
- goal! -keep LDE philosophy
- easy to program
  - isolation

Option 1: PDP/11



← expensive context switch!!!

# Option 2: physically slice mem

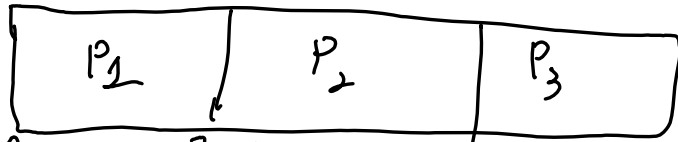


issue: C

A circuit diagram showing a switch with two terminals. The top terminal is labeled '0' and the bottom terminal is labeled '6400'. The text 'isolation??' is written to the right of the switch.

Solution? Ask the hardware!

limit registers: base and bound



A circuit diagram showing a switch with two terminals. The top terminal is labeled '5' and the bottom terminal is labeled '6405'. Below the switch, the text 'error: P1 bound violation' is written. An 'X' is drawn over the value '6405'.

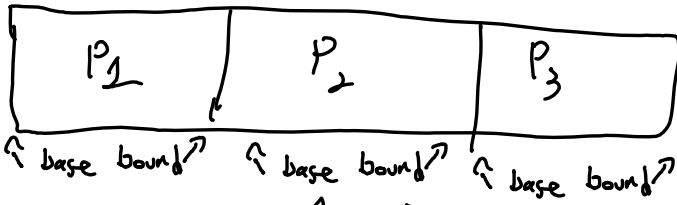
PCB

- PC
- reg
- status
- base
- bound

we can implement!

} kernel sets these

Base/Bound: P2?



note: addresses off by 6400!



problem? yes: again, C.

"transparency" of conceptual machine

solution: address translation

load 5 → load 6405

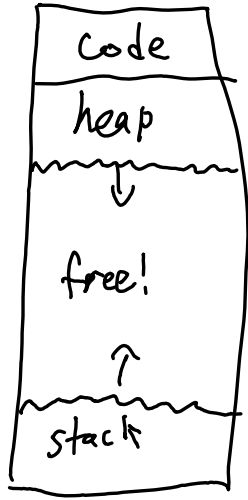
in hw:  $loc \rightarrow loc + base$   
↑  
extra adder circuit

load 6405 → error (always)

how? ↙  
compare circuit

Aside: bound is region size

# C memory assumptions: review



how do we exec:

load 1000

load 1004

add

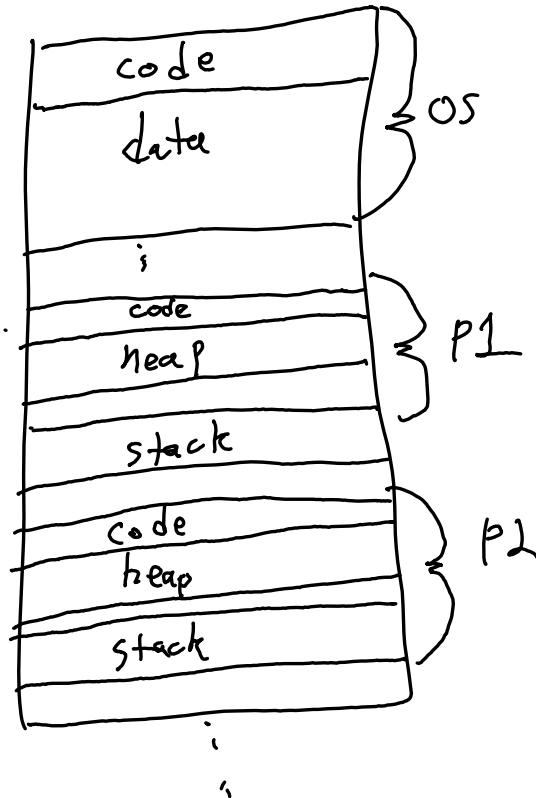
store 1008

physical mem:

Aside: kernel memory (kmem).

How to access?

interact: what does a process switch look like?



interact: do we like this?

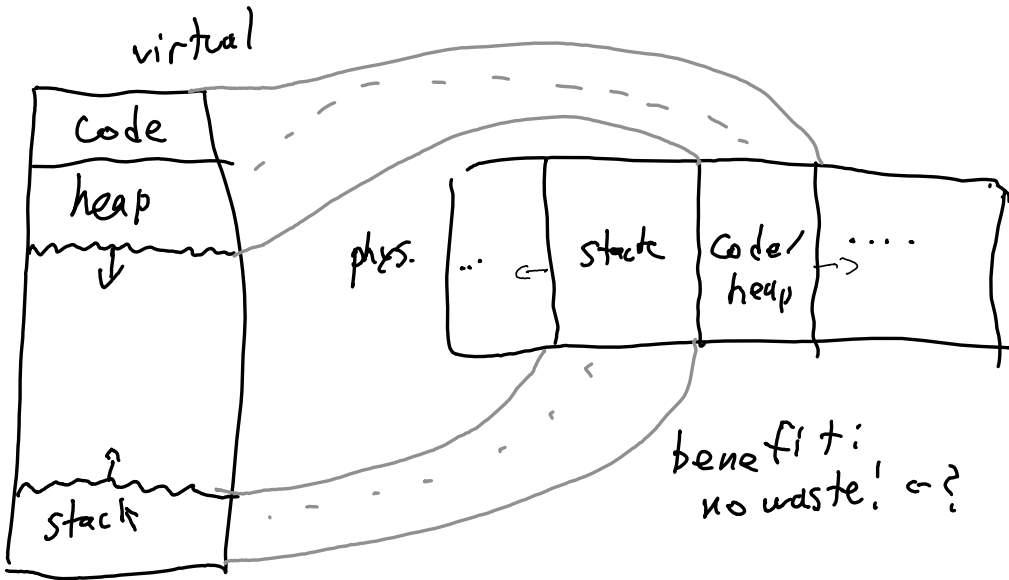
option 2: segmentation



avoid this

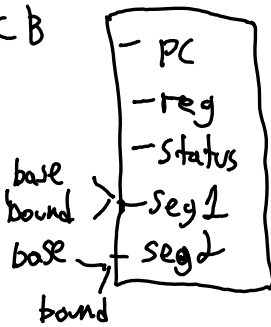
main idea: more than one region/process!

physical



(needs more registers!)

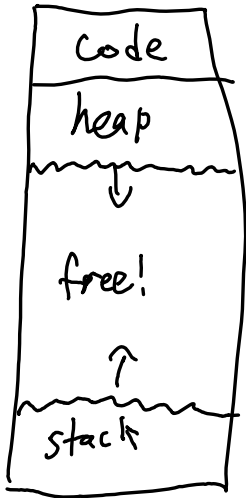
PCB



} kernel sets these

aside: brk / sbrk, C mem review!

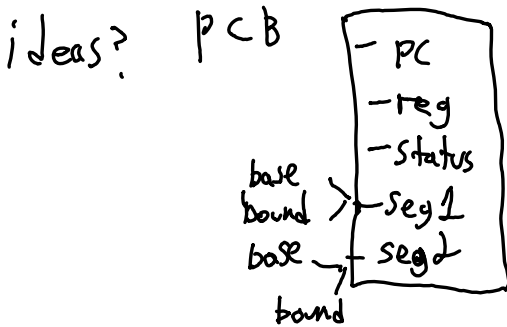
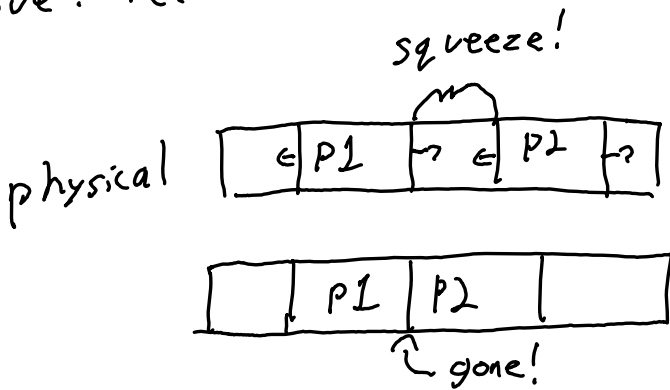
issues: relocation?  
: translation?



how do we exec:  
 load 1000 } what segment?  
 load 1004 }  
 add  
 store 1008

segment knowledge: ideas? (interact!)  
 (high-order bits, implicit, lookup)

issue: relocation:



relocation  
 final note: hw reg  
 for stack (backwards)