9/16/2009

Recap:
- memory maps
- programs
- simple OS
- multitasking

![Diagram of OS, stack, heap, and program code]

Context switching

Simple processor with no protection
OS will deny P1 access to memory locations outside P1's bounds.

- User/Supervisor Mode:
  - U: allowed to do privileged instruction
  - S: restricted

- OS switching: using Interrupts

Context:
- SP
- Base + Bounds

Before:
- Switch from P1 to P2
  - P1, saved SP = SP
  - SP = P2, saved SP
  - return

New:
- P1, saved SP = SP
  - base = P2, base
  - bounds = P2, bounds
  - SP = P2, saved SP
- except for the first process, all processes get
  fork() & Unix
  spawn (forked process) => Win32

process1

process2

if ready
  switch
  interrupt

process1

80 -> 90, 90 -> 50
Win32:

```c
spawn ("program", args)
```

P1

```c
os
P2
"program"
```

Unix:

File I/O

```
key = read_terminal (#)
```

L: not good; not reusable, not adaptable

Solution:

```c
handle = open(terminal1) handle
read (handle)
write (...)```

```c
Read
Read_term
```