

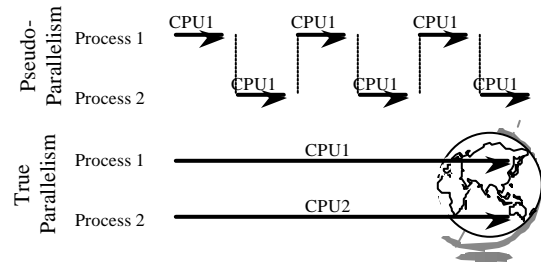


Operating Systems

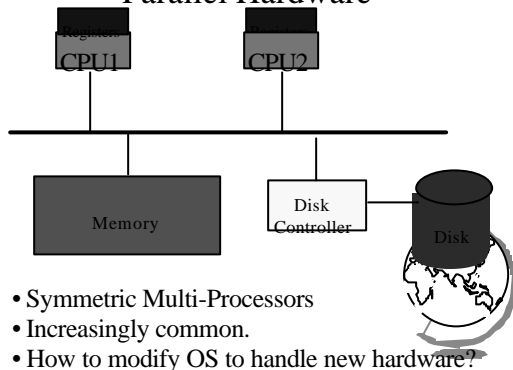
Parallel Systems
(Soon to be basic OS knowledge)

Parallelism

◆ Multiple processes concurrently



Parallel Hardware



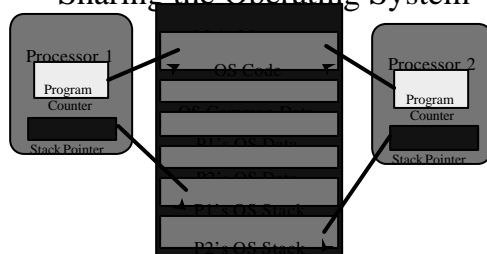
- Symmetric Multi-Processors
- Increasingly common.
- How to modify OS to handle new hardware?

Two Operating Systems

- ◆ Divide memory in two
- ◆ Run an independent OS in each
- ◆ Each has it's own processes
- ◆ Drawbacks
 - Twice as much memory used for OS
 - IPC tough
 - Who controls memory and disk? (convenient)
 - Inefficient scheduling (efficient)



Sharing the Operating System



Shared? (see sample code)
 stack process table
 current process device queues

Race Conditions!

Example Multiprocessor OSES

- ◆ Almost all new OSES ! ◆ Unix
- ◆ Designed from start
 - Windows NT
 - Mach
 - AT&T System V
 - Sun Solaris
 - HP Unix
 - Digital Unix
 - IBM AIX
 - SGI Irix





Threads

Software Multi-Processors

Threads (Lightweight Processes)

◆ Basic unit of CPU utilization

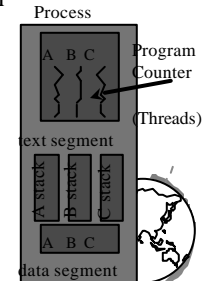
- ("What?!" you say)

◆ Own

- program counter
- register set
- stack space

◆ Shares

- code section
- data section
- OS resources



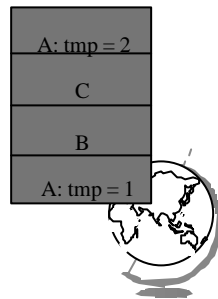
"Multithreaded Program"

Stack

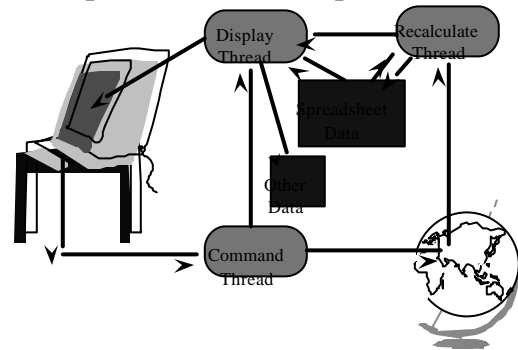
```

A(int tmp) {
    B();

    printf(tmp);
}
B() {
    C();
}
C() {
    A(2);
}
    
```



Example: A Threaded Spreadsheet



What Kinds of Programs to Thread?

◆ Independent tasks

- ex: debugger needs gui, program, perf monitor...
- especially when blocking for I/O!

◆ Single program, concurrent operation

- Servers
 - ◆ ex: file server, web server
- OS kernels
 - ◆ concurrent requests by multiple users -- no protection needed in kernel



Thread Benefits

◆ "What about just using processes with shared memory?"

- fine
- debugging tougher (more thread tools)
- processes slower
 - ◆ 30 times slower to create on Solaris
 - ◆ slower to destroy
 - ◆ slower to context switch among
- processes eat up memory
 - ◆ few thousand processes not ok
 - ◆ few thousand threads ok



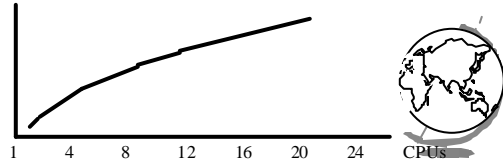
Threads Standards

- ◆ POSIX (Pthreads)
 - Common API
 - Almost **all** Unix's have thread library
- ◆ Win32 and OS/2
 - very different from POSIX, tough to port
 - commercial POSIX libraries for Win32
 - OS/2 has POSIX option
- ◆ Solaris
 - started before POSIX standard
 - likely to be like POSIX



Do they Work?

- ◆ Operating systems
 - Mach, Windows NT, Windows 95, Solaris, IRIX, AIX, OS/2, OSF/1
 - Millions of (unforgiving) users
- ◆ NFS, SPECfp, SPECint



Levels of Threads

