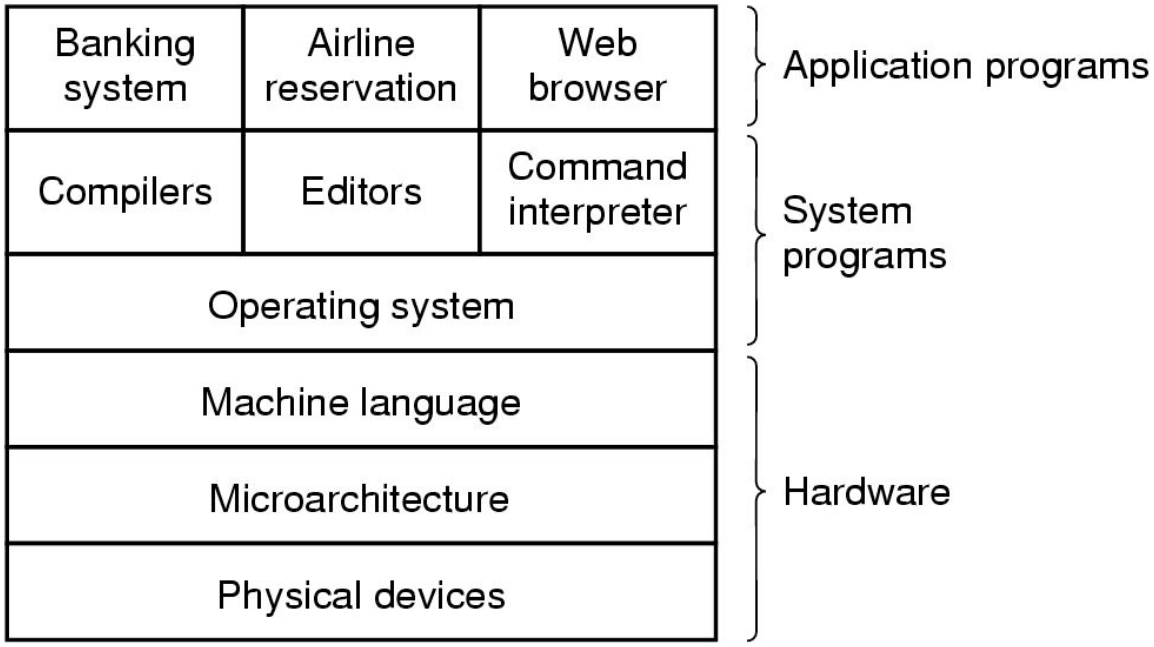
16.480/552

Embedded Operating Systems

Introduction

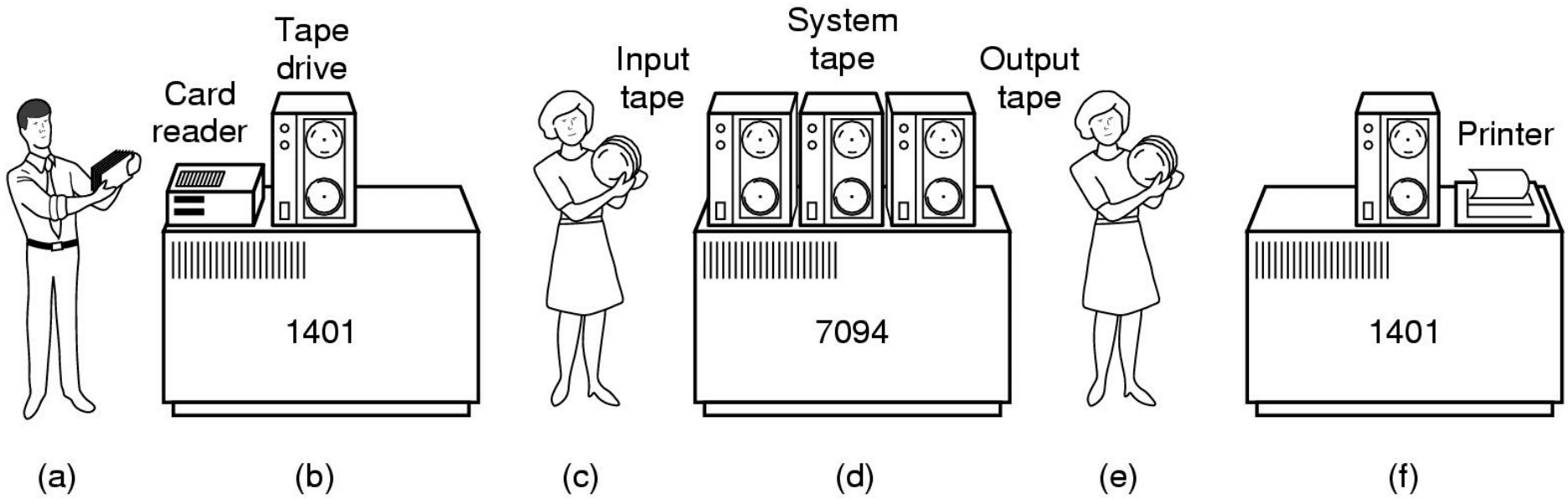


- A computer system consists of
  - hardware
  - system programs
  - application programs

What is an Operating System

- It is an extended machine
  - Hides the messy details which must be performed
  - Presents user with a virtual machine, easier to use
- It is a resource manager
  - Each program gets time with the resource
  - Each program gets space on the resource

History of Operating Systems (1)



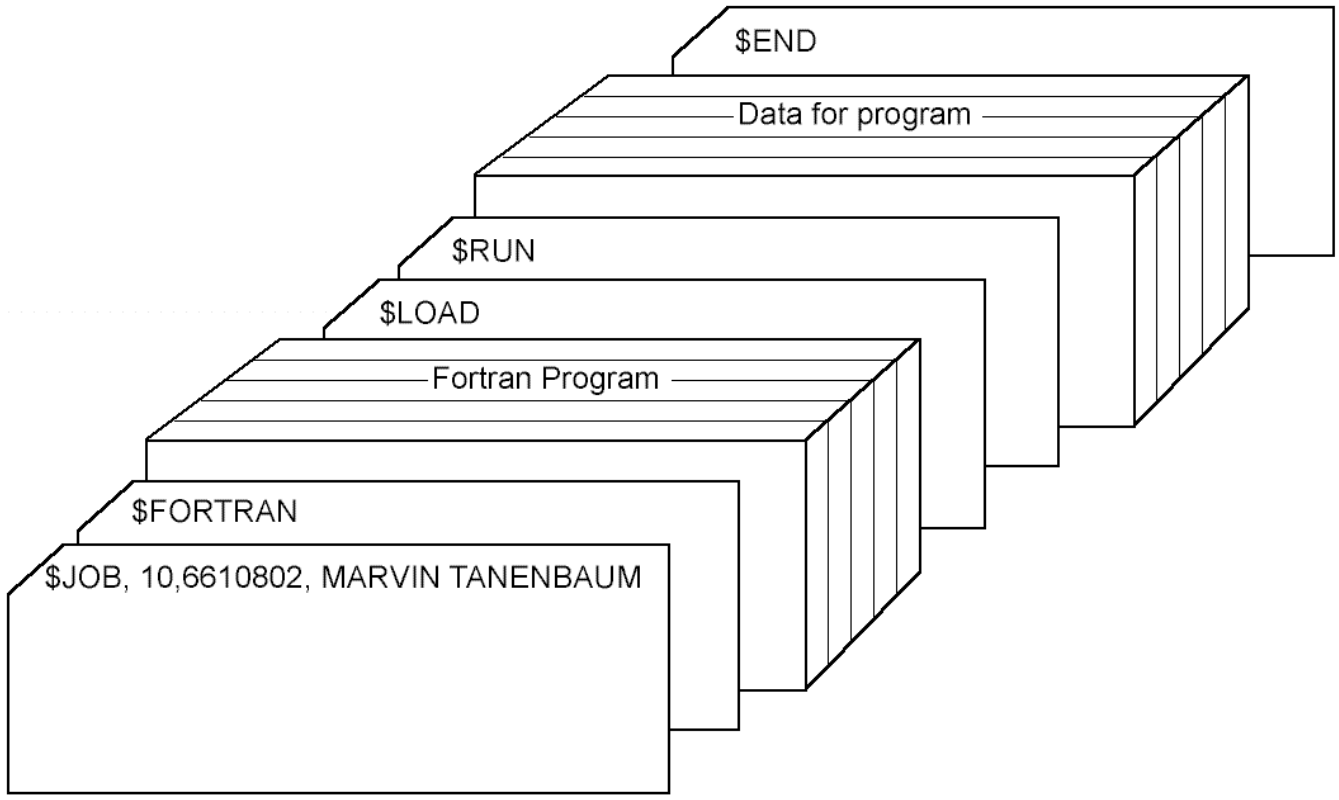
Early batch system

- bring cards to 1401
- read cards to tape
- put tape on 7094 which does computing
- put tape on 1401 which prints output

History of Operating Systems (2)

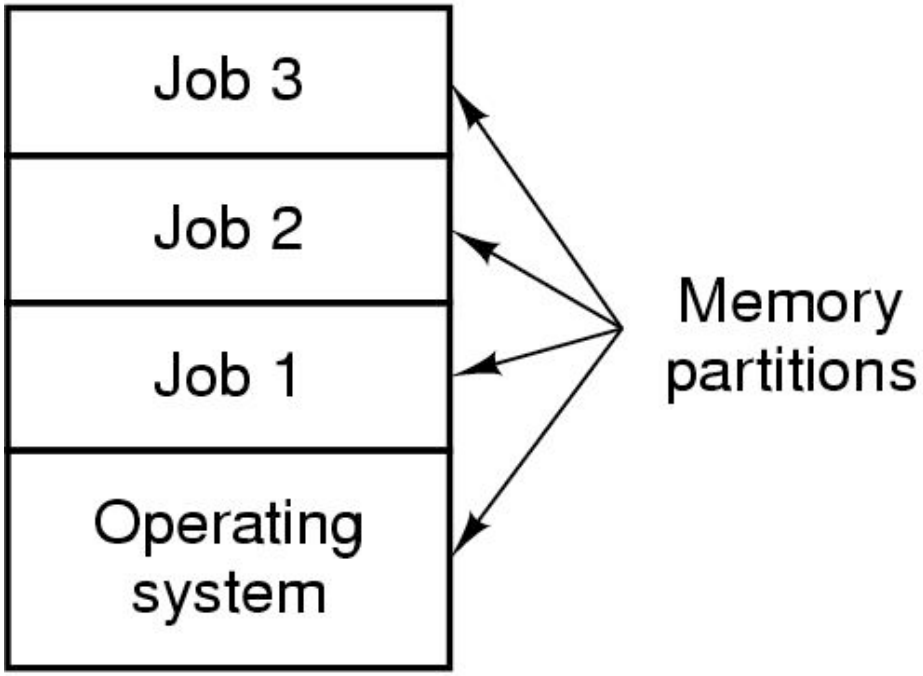
- First generation 1945 - 1955
  - vacuum tubes, plug boards
- Second generation 1955 - 1965
  - transistors, batch systems
- Third generation 1965 – 1980
  - ICs and multiprogramming
- Fourth generation 1980 – present
  - personal computers

History of Operating Systems (3)



- Structure of a typical FMS job – 2nd generation

History of Operating Systems (4)

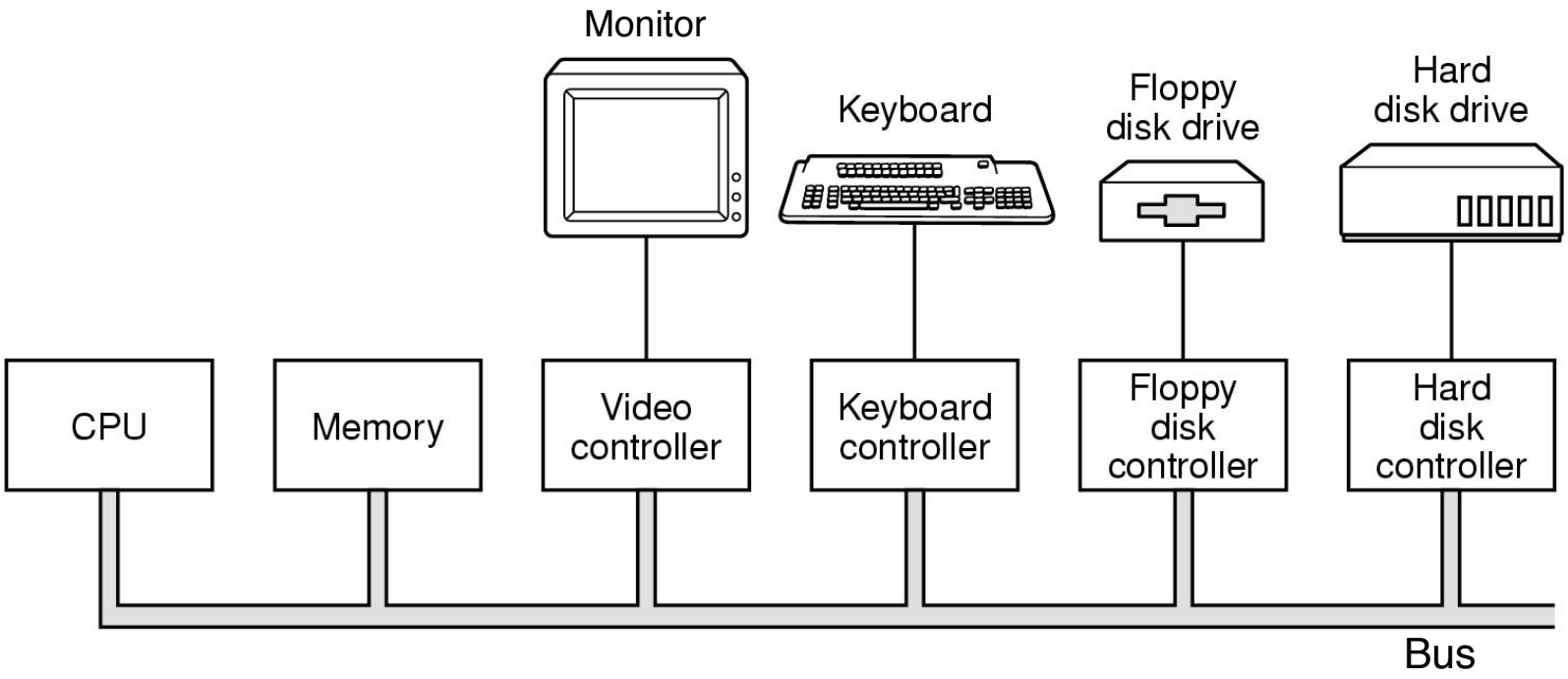


- Multiprogramming system
  - three jobs in memory – 3rd generation

The Operating System Zoo

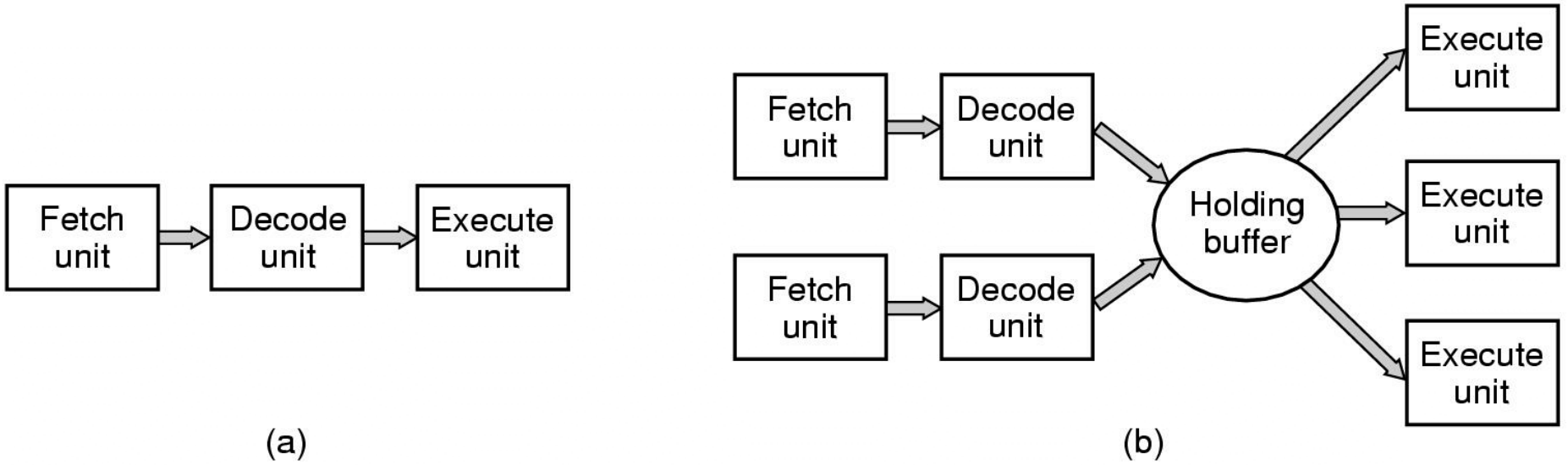
- Mainframe operating systems
- Server operating systems
- Multiprocessor operating systems
- Personal computer operating systems
- Real-time operating systems
- Embedded operating systems
- Smart card operating systems

Computer Hardware Review (1)



- Components of a simple personal computer

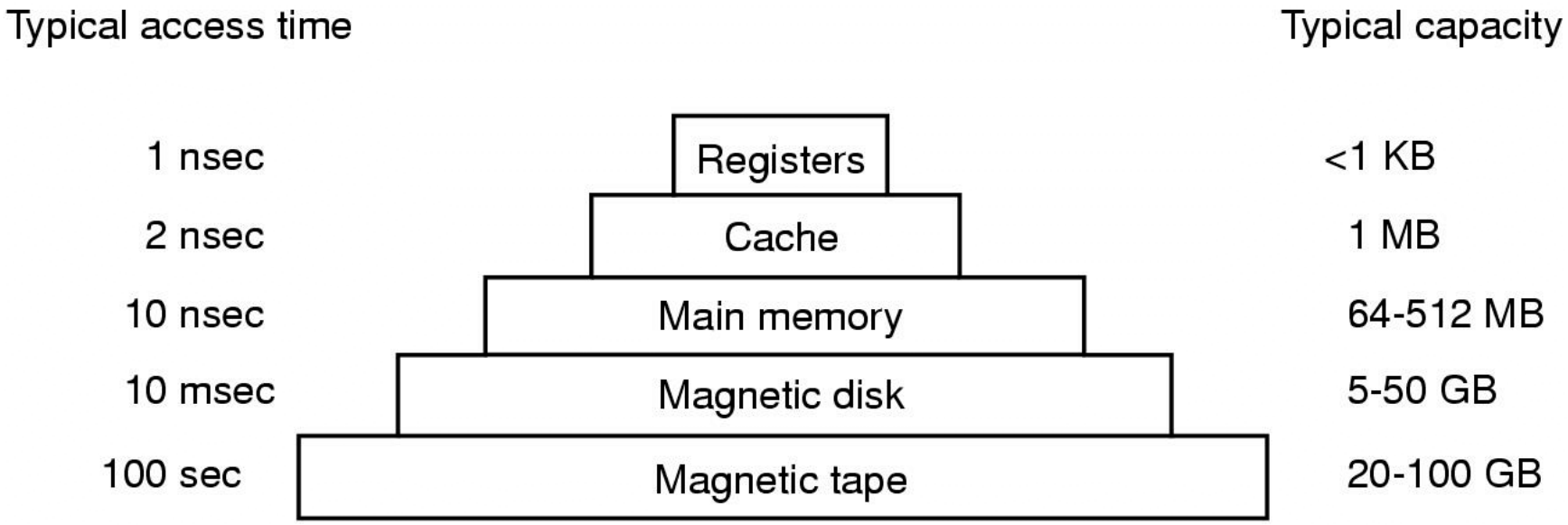
Computer Hardware Review (2)



(a) A three-stage pipeline

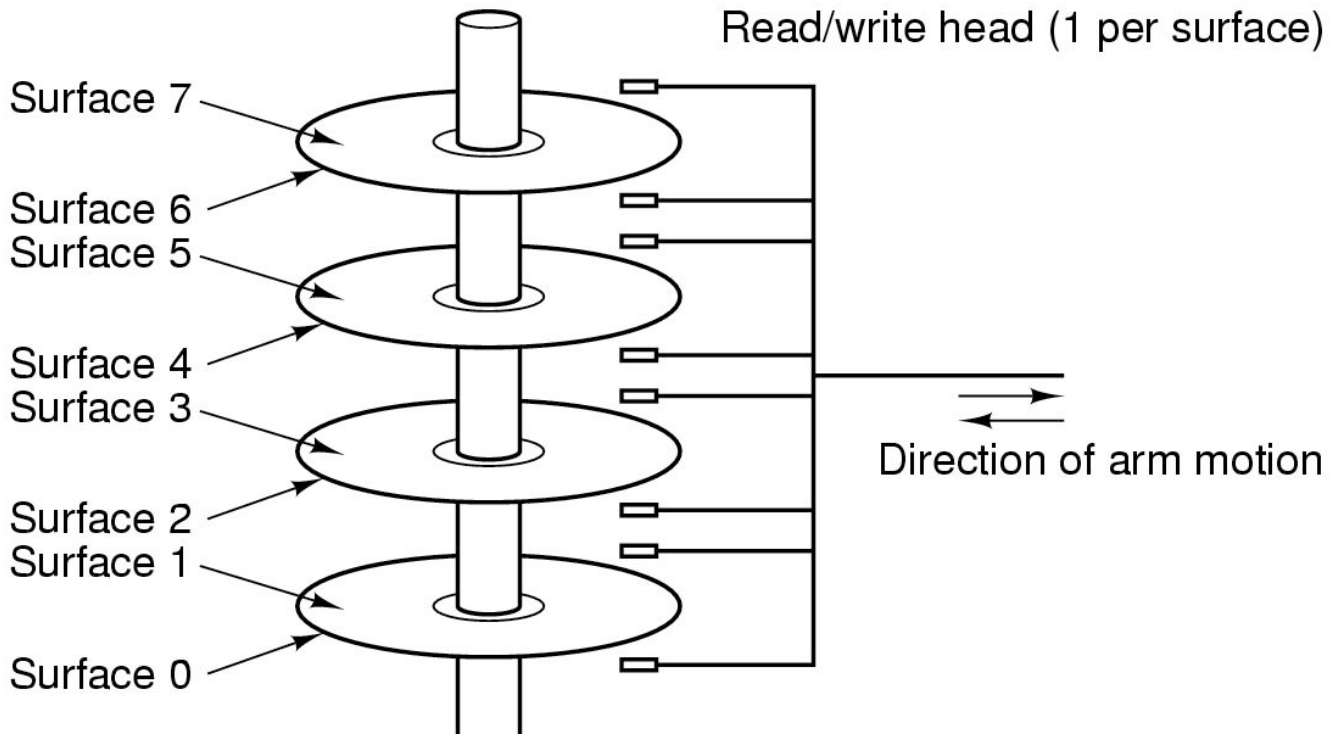
(b) A superscalar CPU

Computer Hardware Review (3)



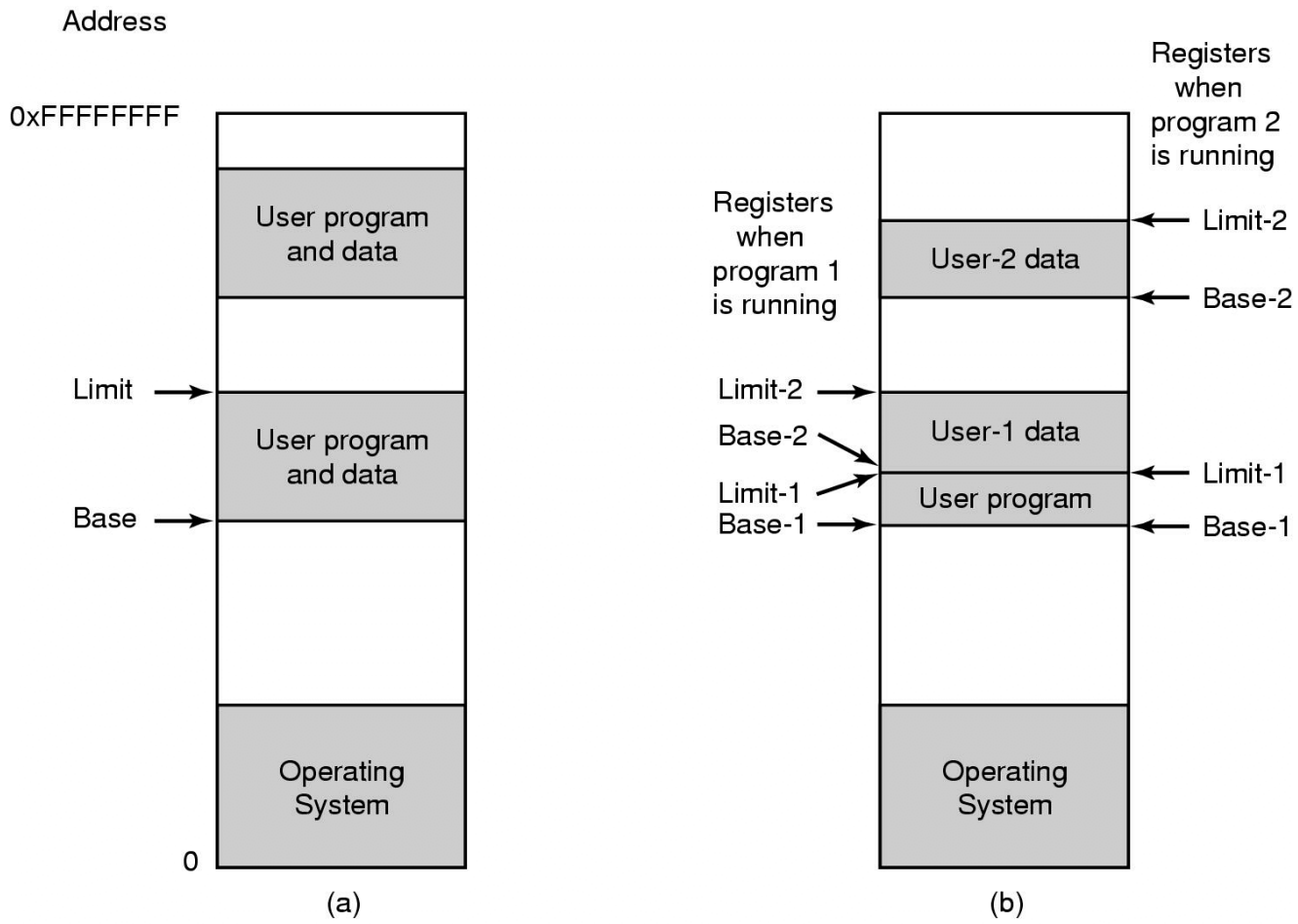
- Typical memory hierarchy
  - numbers shown are rough approximations

Computer Hardware Review (4)



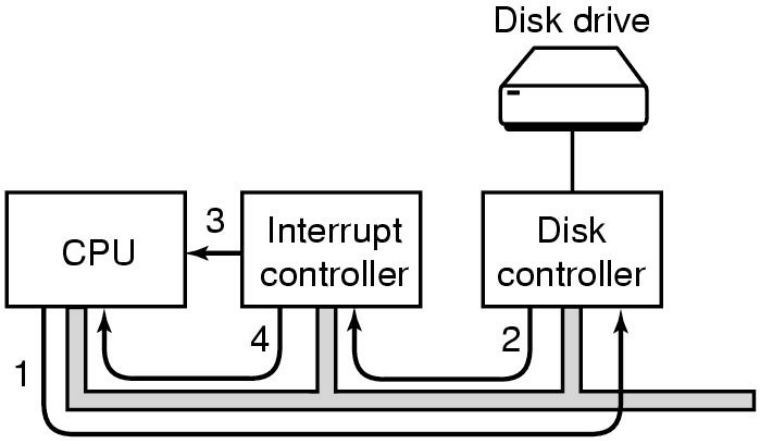
Structure of a disk drive

Computer Hardware Review (5)

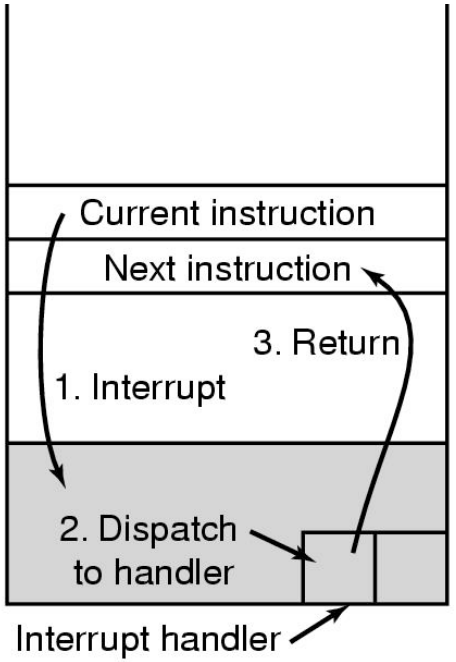


One base-limit pair and two base-limit pairs

Computer Hardware Review (6)



(a)

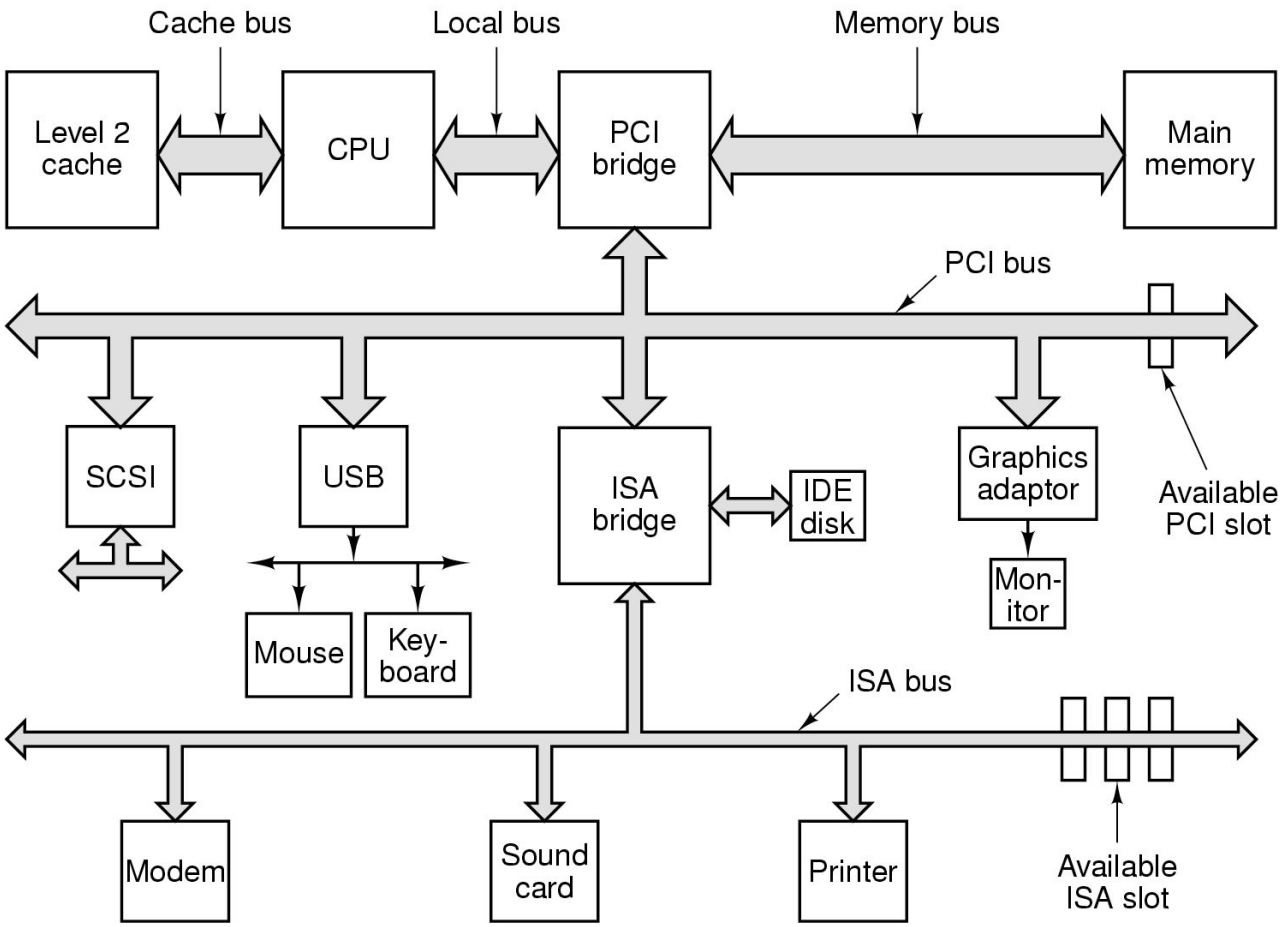


(b)

(a) Steps in starting an I/O device and getting interrupt

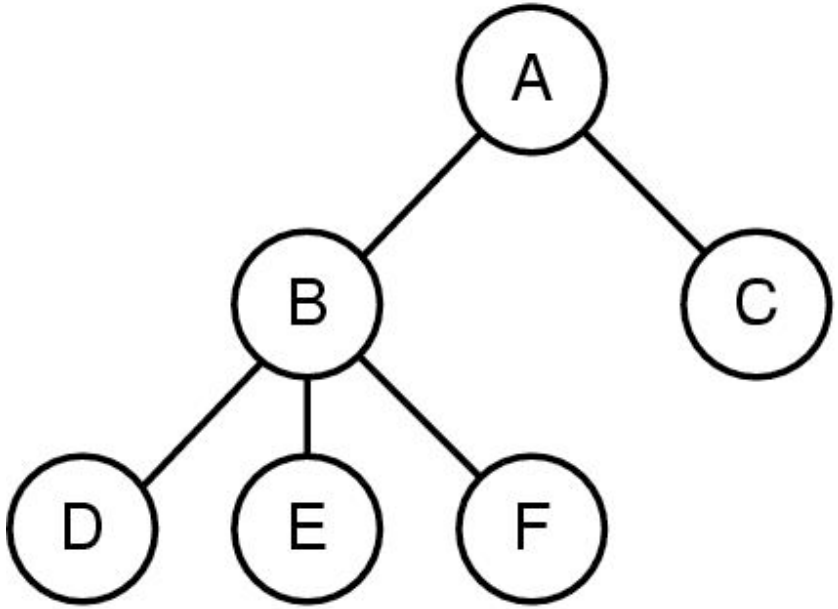
(b) How the CPU is interrupted

Computer Hardware Review (7)



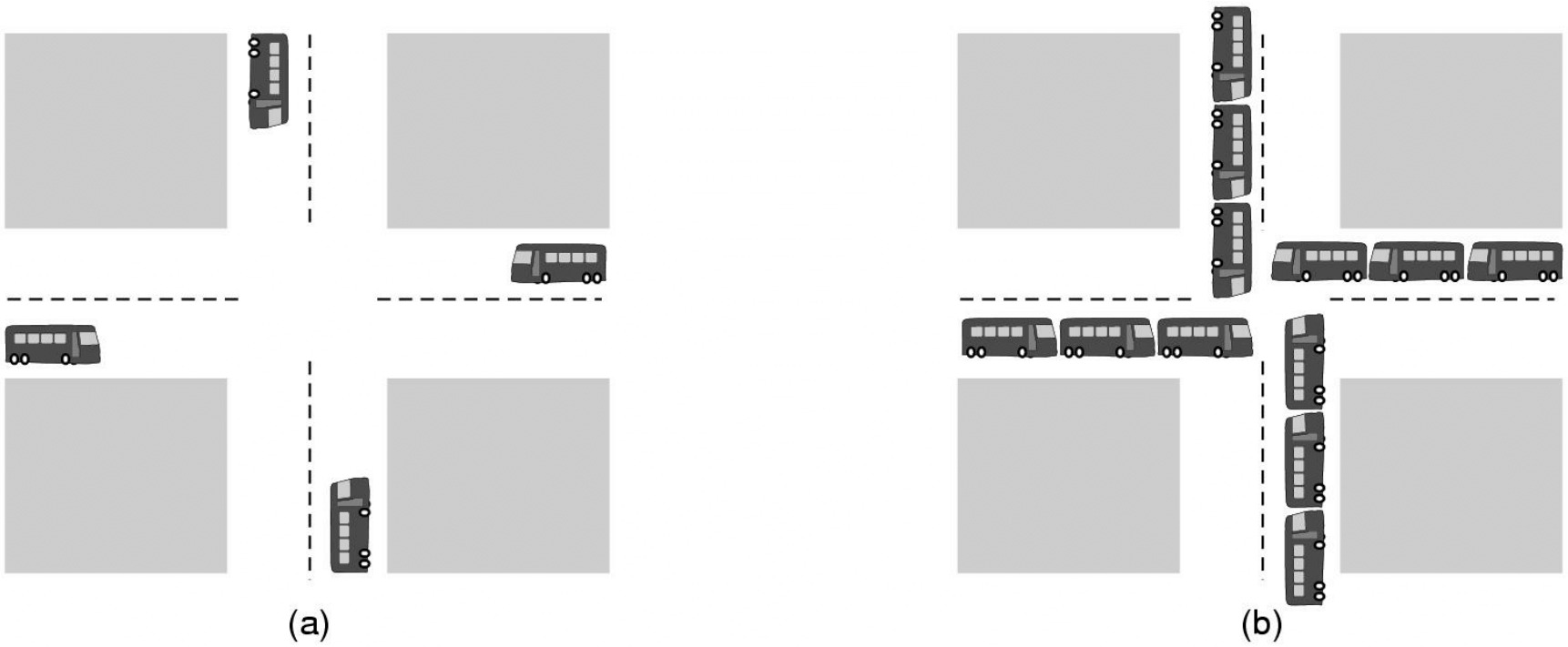
Structure of a large Pentium system

Operating System Concepts (1)



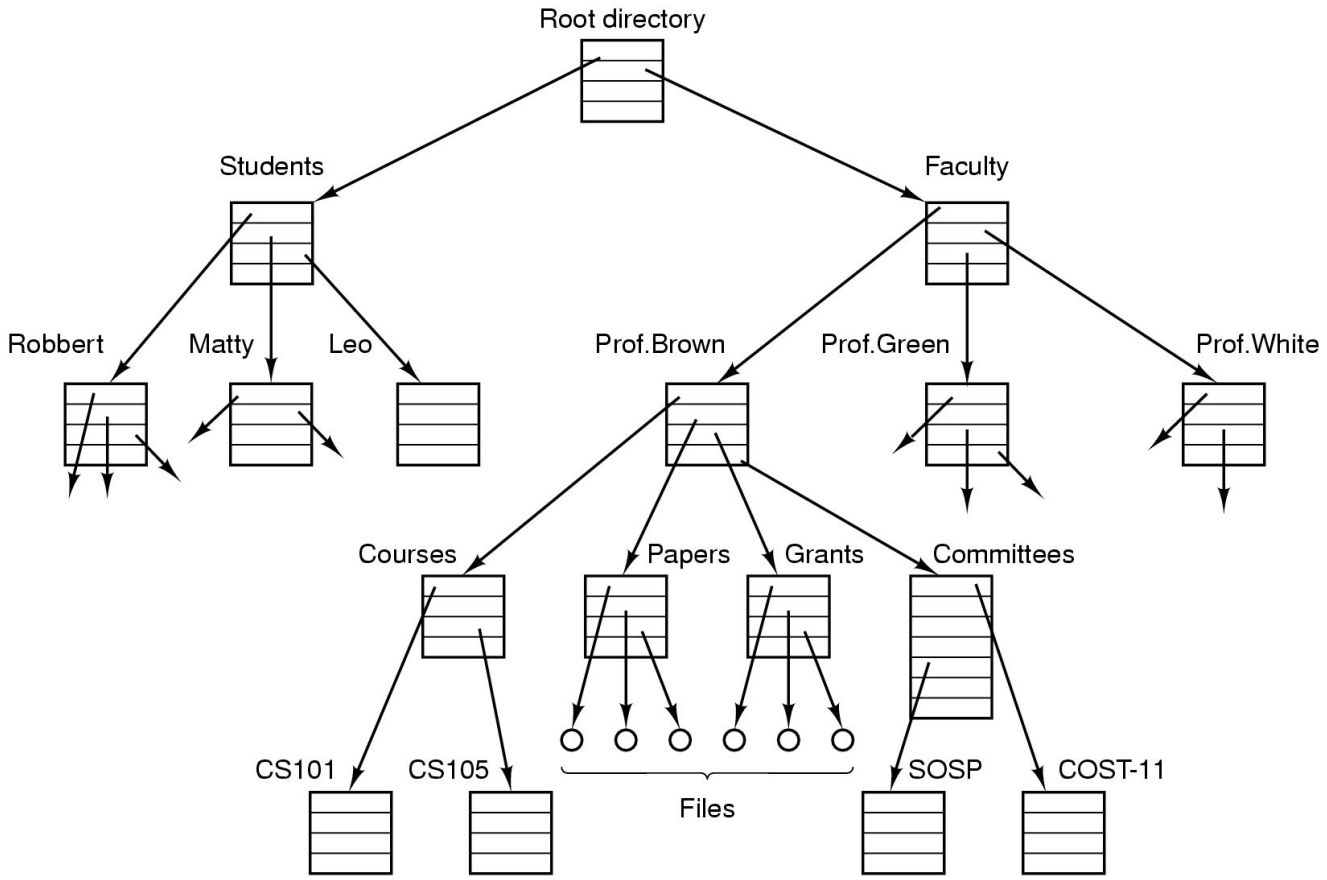
- A process tree
  - A created two child processes, B and C
  - B created three child processes, D, E, and F

Operating System Concepts (2)



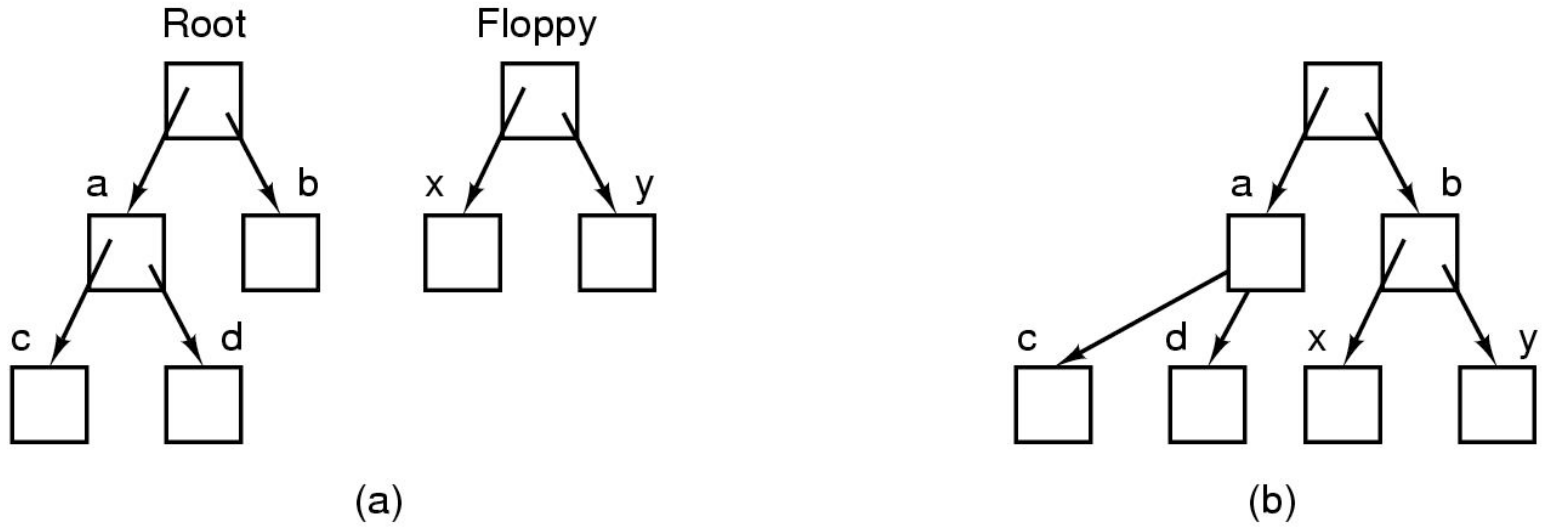
(a) A potential deadlock. (b) an actual deadlock.

Operating System Concepts (3)



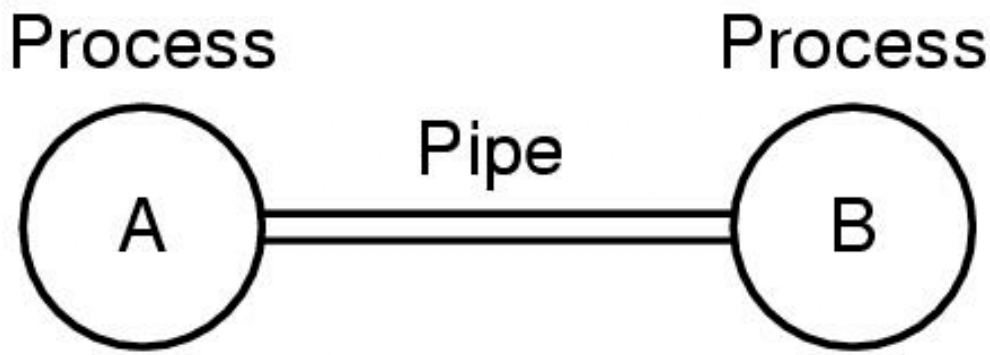
File system for a university department

Operating System Concepts (4)



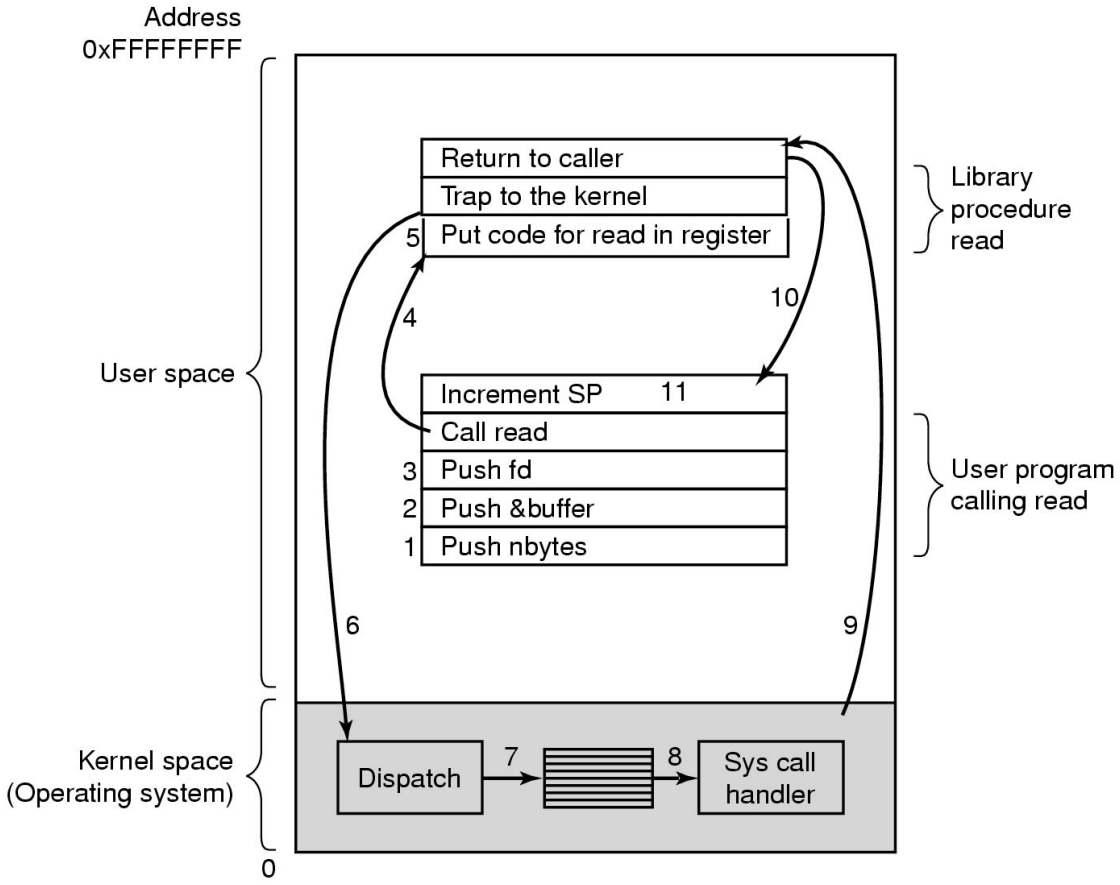
- Before mounting,
  - files on floppy are inaccessible
- After mounting floppy on b,
  - files on floppy are part of file hierarchy

Operating System Concepts (5)



Two processes connected by a pipe

Steps in Making a System Call



There are 11 steps in making the system call

read (fd, buffer, nbytes)

Some System Calls For Process Management

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

Some System Calls For File Management

File management

Call	Description
fd = open(file, how, ...)	Open a file for reading, writing or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

Some System Calls For Directory Management

Directory and file system management

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

Some System Calls For Miscellaneous Tasks

Miscellaneous

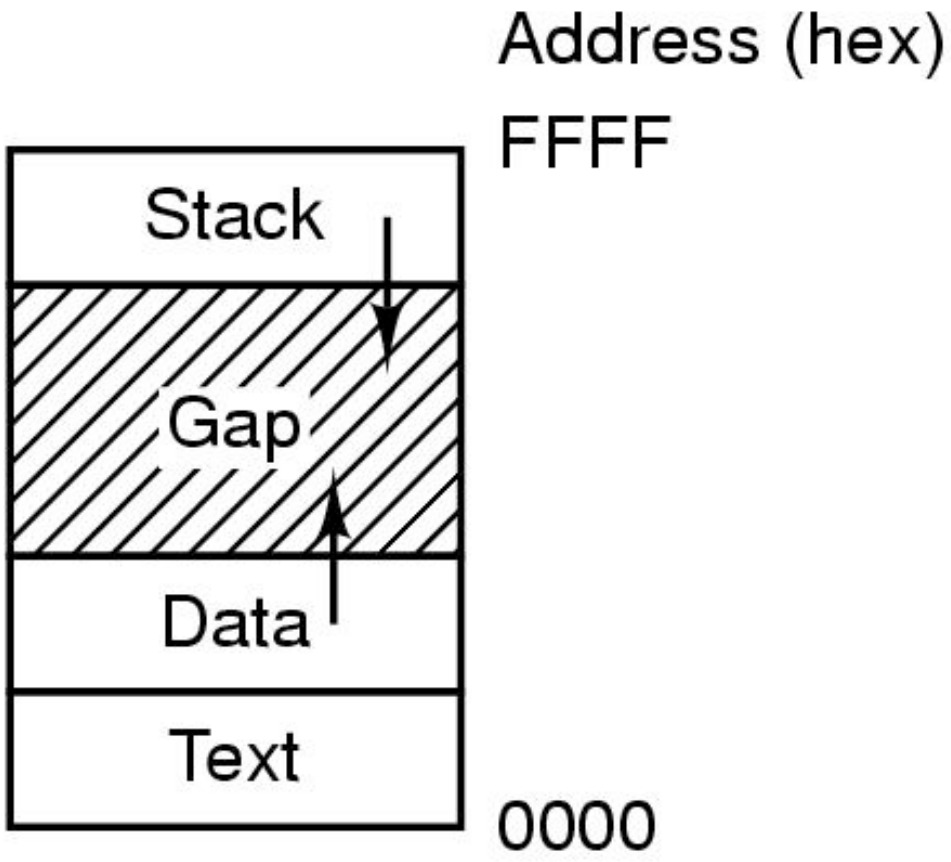
Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

System Calls (1)

- A stripped down shell:

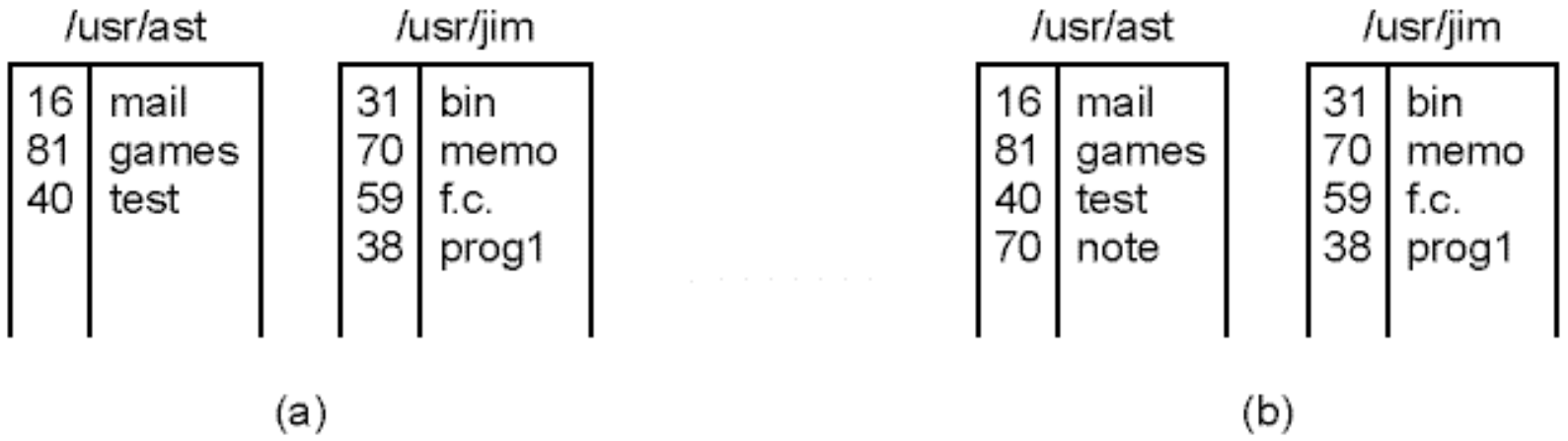
```
while (TRUE) {                                  /* repeat forever */
    type_prompt( );                             /* display prompt */
    read_command (command, parameters)          /* input from terminal */

if (fork() != 0) {                              /* fork off child process */
    /* Parent code */
    waitpid( -1, &status, 0);                   /* wait for child to exit */
} else {
    /* Child code */
    execve (command, parameters, 0);            /* execute command */
 }
}
```

System Calls (2)



- Processes have three segments: text, data, stack

System Calls (3)



(a) Two directories before linking *usr/jim/memo* to ast's directory

(b) The same directories after linking

System Calls (4)

(a) File system before the mount

(b) File system after the mount

System Calls (5)

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Some Win32 API calls

Embedded OS

- What is an embedded Operating System
- Who are the players
- Linux as an embedded OS
- Tools and development
- Applications and products
- The embedded OS market
- Embedded OS Resources

OS Flavors

- Desktop
  - Windows (9X, XP Home, XP/2000 Pro)
  - Mac
- Server
  - Windows (XP/2000 Server &Advanced Server)
  - Unix Varieties
- Embedded
  - Many

What is an Embedded OS?

- An "embedded system" is any computer system or computing device that performs a dedicated function or is designed for use with a specific embedded software application.

- Embedded systems may use a ROM-based operating system or they may use a disk-based system, like a PC. But an embedded system is not usable as a commercially viable substitute for general purpose computers or devices.

What makes a good Embedded OS?

- Modular
- Scalable
- Configurable
- Small footprint
- CPU support
- Device drivers
- etc, etc, etc...

What is Real Time?

"A real time system is one in which the correctness of the computations not only depends upon the logical correctness of the computation but also upon the time at which the result is produced. If the timing constraints of the system are not met, system failure is said to have occurred."

- Donald Gillies

What is Real Time?

"Real time in operating systems:

The ability of the operating system to provide a required level of service in a bounded response time."

- POSIX Standard 1003.1

Hard vs. Soft Real Time

- Hard
  - *guaranteed worst-case* response times
  - absolutely, positively, first time every time
- Soft
  - Kinda, sorta, usually

What makes a good RTOS?

- Multi-threaded and pre-emptible
- Thread priority has to exist because no deadline driven OS exists
- Must support predictable thread synchronization mechanisms
- A system of priority inheritance must exist

Who are the Embedded OS players?

- Wind River Systems
  - VxWorks
  - pSOS
- QNX Software Systems
  - QNX
- Green Hills Software
  - Integrity

Who are the Embedded OS players?

- Mentor Graphics
  - VRTX
- Palm Computing
  - PalmOS
- Symbian
  - SymbianOS

Microsoft

- Embedded NT/XP
  - "Real-time" control
- Windows CE (CE.NET)
  - Internet devices
- Pocket PC 2002
  - Handheld PC's and PDA's

Commercial Embedded Linux

- AMIRIX Embedded Linux
  - derived from Debian
- Coollogic Coollinux
  - combines Linux and Java for Internet apps
- Coventive Xlinux
  - kernel can be as small as 143KB
- Esfia RedBlue Linux
  - 400K, designed for wireless apps

Commercial Embedded Linux

- KYZO Pizza Box Linux
  - SAMBA based file, print, CD server
- Lineo Embedix
  - supports real time and high availability apps
- LynuxWorks BlueCat
  - general purpose embedded solution
- MontaVista Linux
  - general purpose embedded solution

Commercial Embedded Linux

- Neoware NeoLinux
  - Red Hat derived for information appliances
- PalmPalm Tynux
  - Internet appliuance and multimedia
- Red Hat Embedded Linux
  - general purpose embedded solution
- RedSonic Red-Ice Linux
  - runs from DiskonChip or CompactFlash

Commercial Embedded Linux

- RidgeRun DSP Linux
  - for multimedia, wireless, RT on DSP
- TimeSys Linux GPL
  - low latency enhanced kernel
- Tuxia TASTE
  - distro targets Internet appliances
- Vital Systems vLinux
  - for ARM based embedded apps

Open Source Embedded Linux

- Embedded Debian Project
  - convert Debian to an embedded OS
- ETLinux
  - for PC104 SBC's
- uCLinux
  - for microprocessors that don't have MM
- uLinux (muLinux)
  - distro fits on a single floppy

Commercial Linux RTOS

- FSMLabs - Open RT Linux
- Lineo - Embedix Realtime
- LynuxWorks - BlueCat RT
- MontaVista Software - Real Time Extensions
- REDSonic - REDICE Linux
- TimeSys - Linux/Real-Time

Open Source Linux RTOS

- ART Linux - real time extension based on RTLinux
- KURT - event schedules with 10us resolution
- Linux-SRT - for soft real time apps like multimedia
- Qlinux - provides Quality of Service guarantees
- RTAI - “hard” Real Time Application Interface

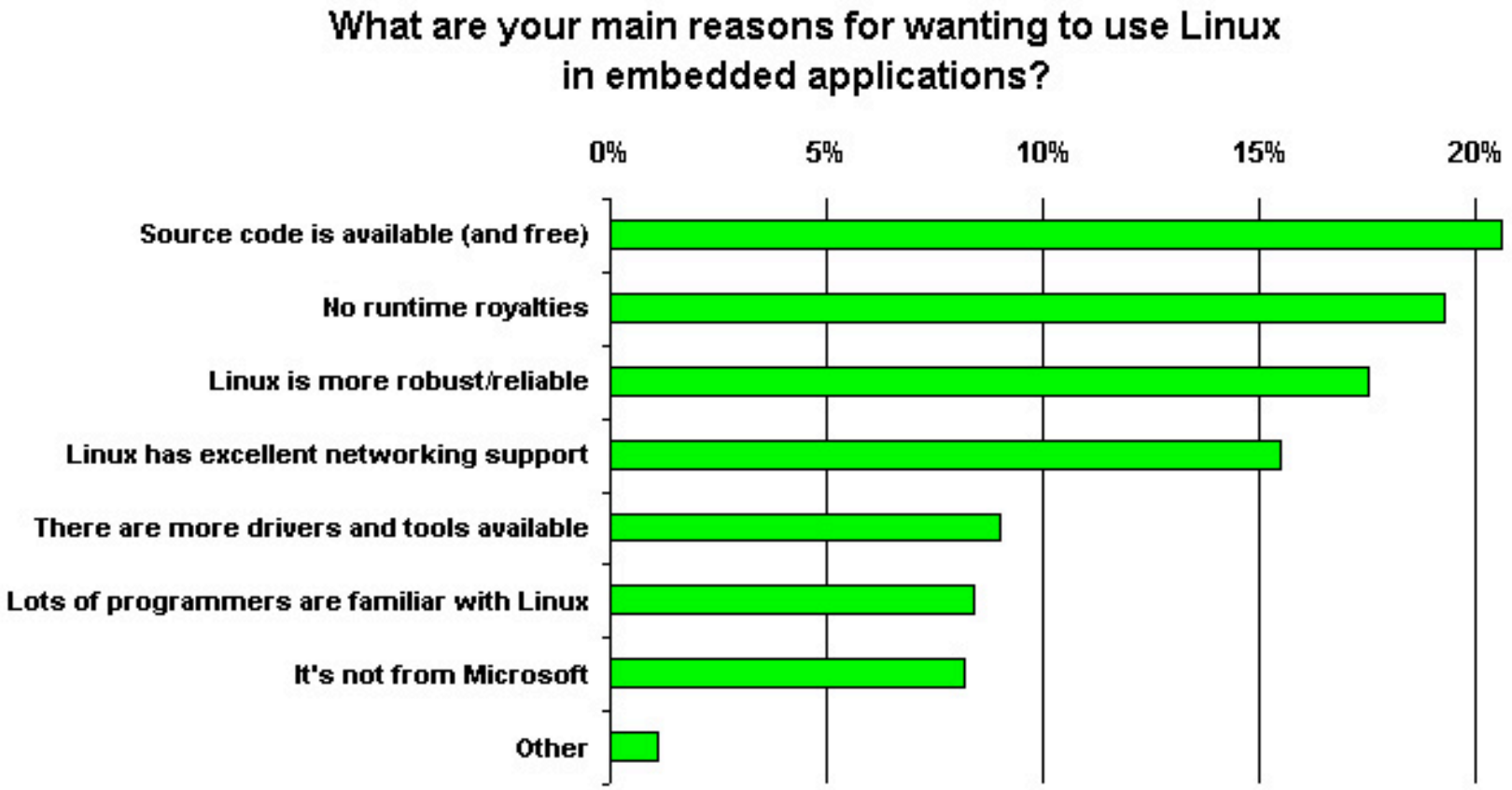
RTLinux

- A “hard real-time” mini operating system
- runs Linux as it’s lowest priority execution thread
- Linux thread completely preemptible
- Real time threads and interrupt handlers never delayed by non-realtime operations
- Supports user level programming
- MiniRTL implementation fits on a floppy

What's so special about Linux?

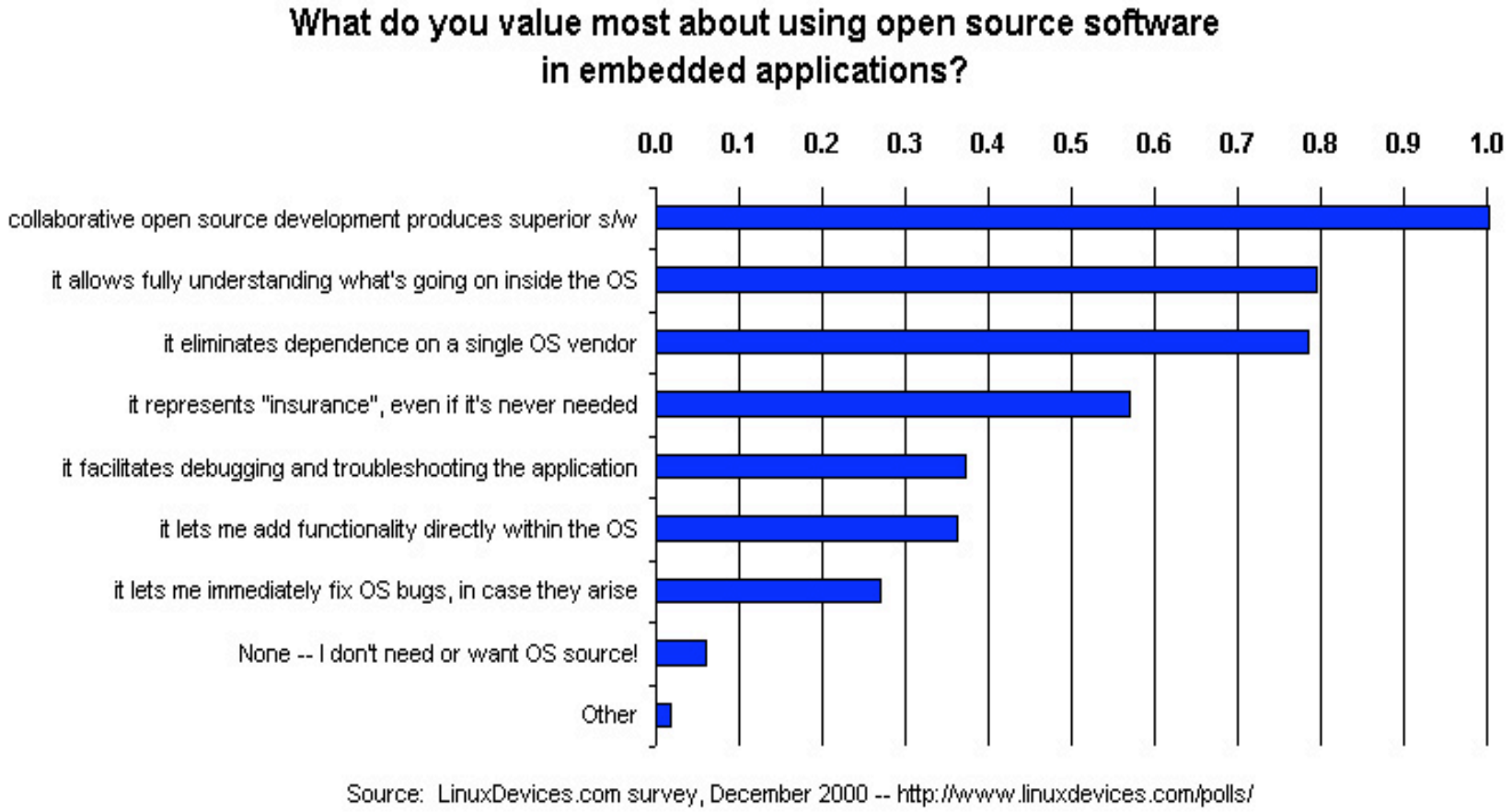
- Multiple choices vs. sole source
- Source code freely available
- Robust and reliable
- Modular, configurable, scalable
- Superb support for networking and Internet
- No runtime licenses
- Large pool of skilled developers

What's so special about Linux?



Source: LinuxDevices.com survey, December 2000 -- http://www.linuxdevices.com/polls/

What's special about Open Source?



Source: LinuxDevices.com survey, December 2000 -- http://www.linuxdevices.com/polls/

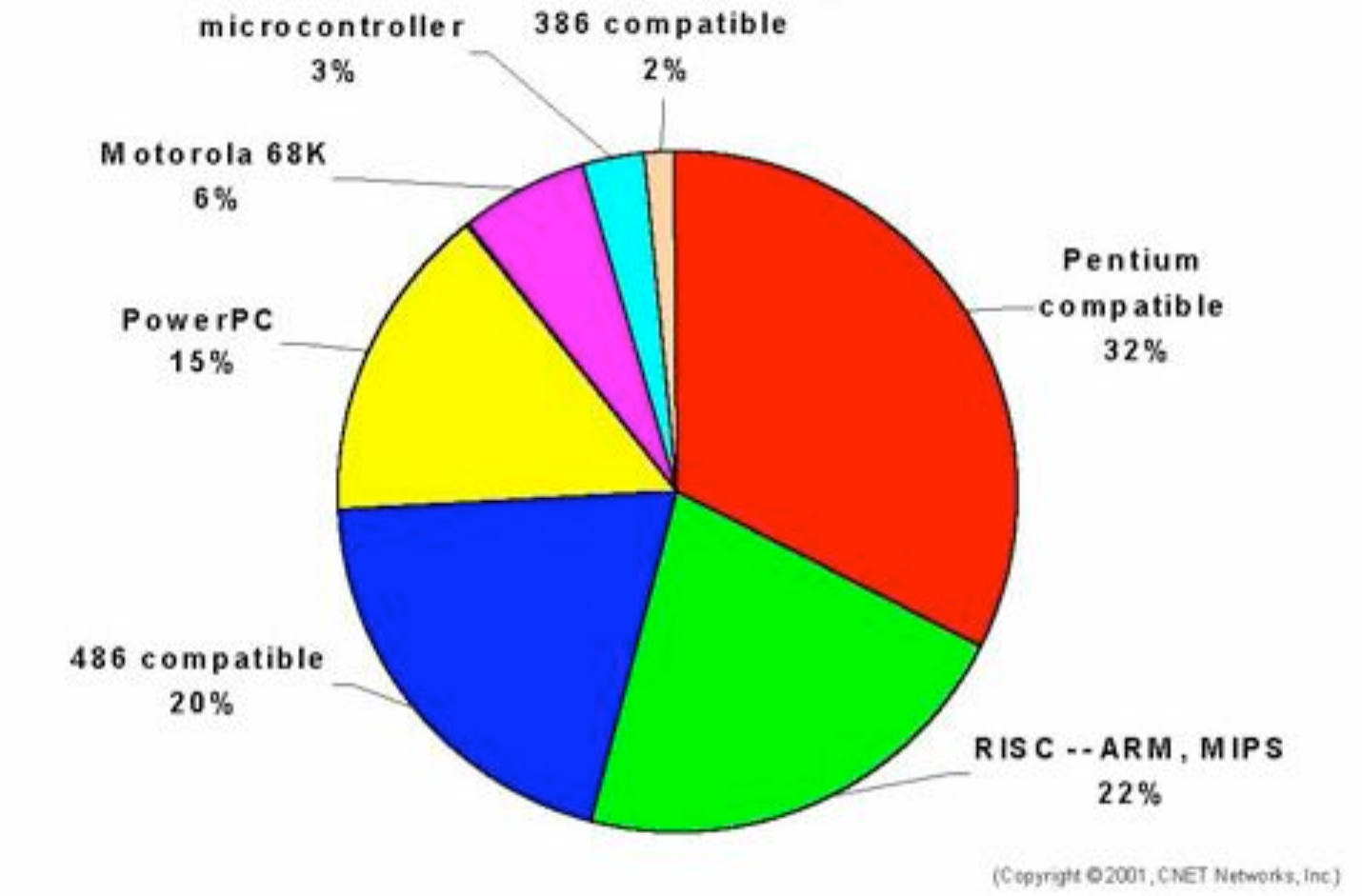
Flies in the Ointment

- Lack of hardware device drivers
- Competing and/or lacking standards
- No formalized qualification testing
- No single source for marketing
- GPL license issues
- Startup vendors with shaky futures

What CPU's will it run on?

- Intel X86
- MIPS
- ARM
- StrongARM
- PowerPC
- Hitachi SuperH

Projected Target CPU's



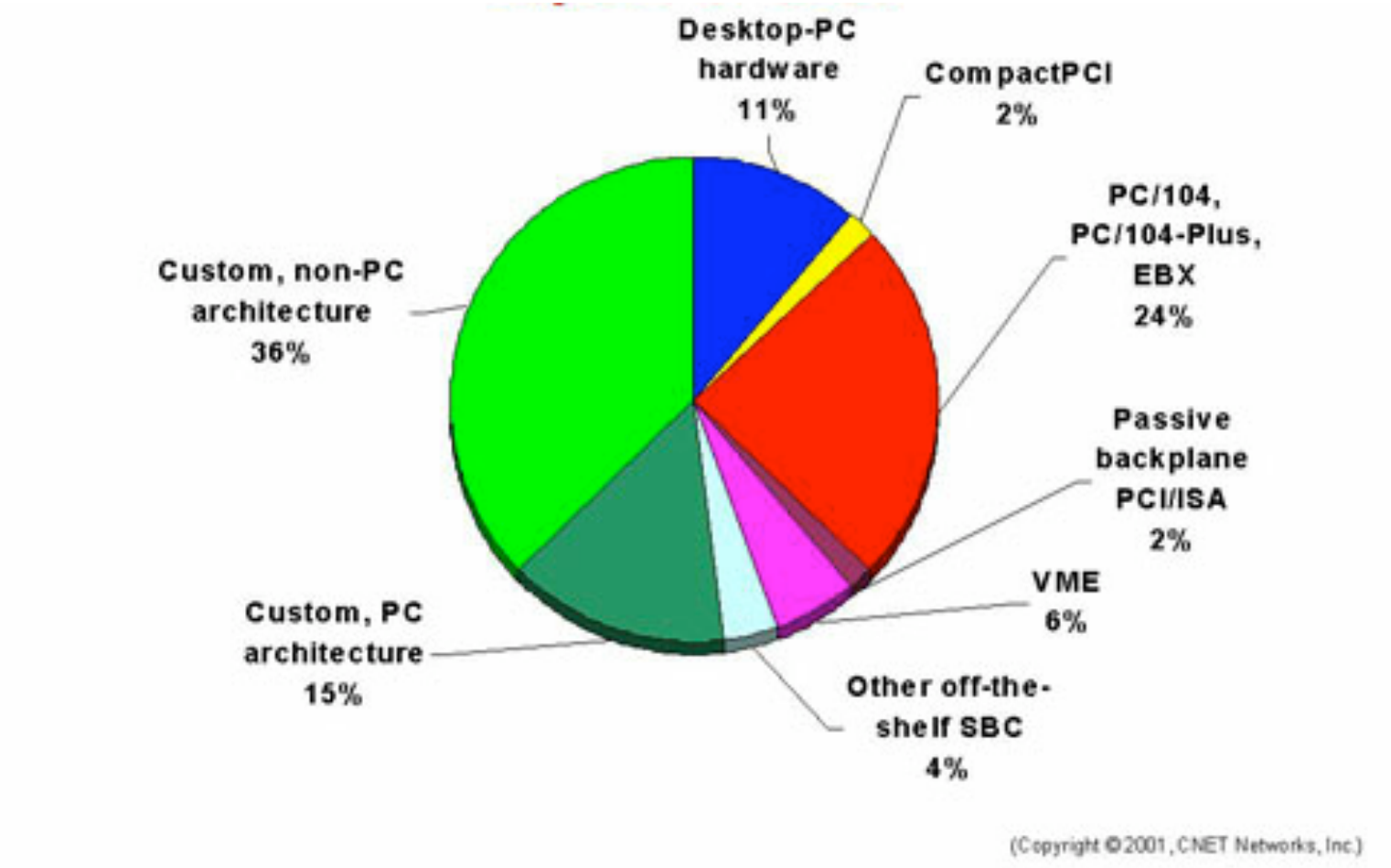
Single Board Computers

- **Little Board** (5.75 x 8.0 in.) -- complete systems on a single compact board, expandable with plug-on function modules
- **ISA "slot boards"** (full-length, 13.8 x 4.8 in.; half-length, 7.1 x 4.8 in.) -- IBM PC plug-in cards which could function as standalone SBCs backplanes)
- **PC/104 modules** (3.6 x 3.8 in.) -- compact, rugged, self-stacking modules featuring a reliable pin-and-socket board-to-board expansion bus

Single Board Computers with PCI

- **PC/104-Plus** -- PCI added to PC/104
- **EBX** -- PC/104-Plus added to Little Board

Target Devices



Any Development Tools Available?

- QT/Embedded
- Other GUI/Windowing toolkits
- Arcom Control Systems
- GNUPro Tools
- Vendor specific
- Standard Linux toolset

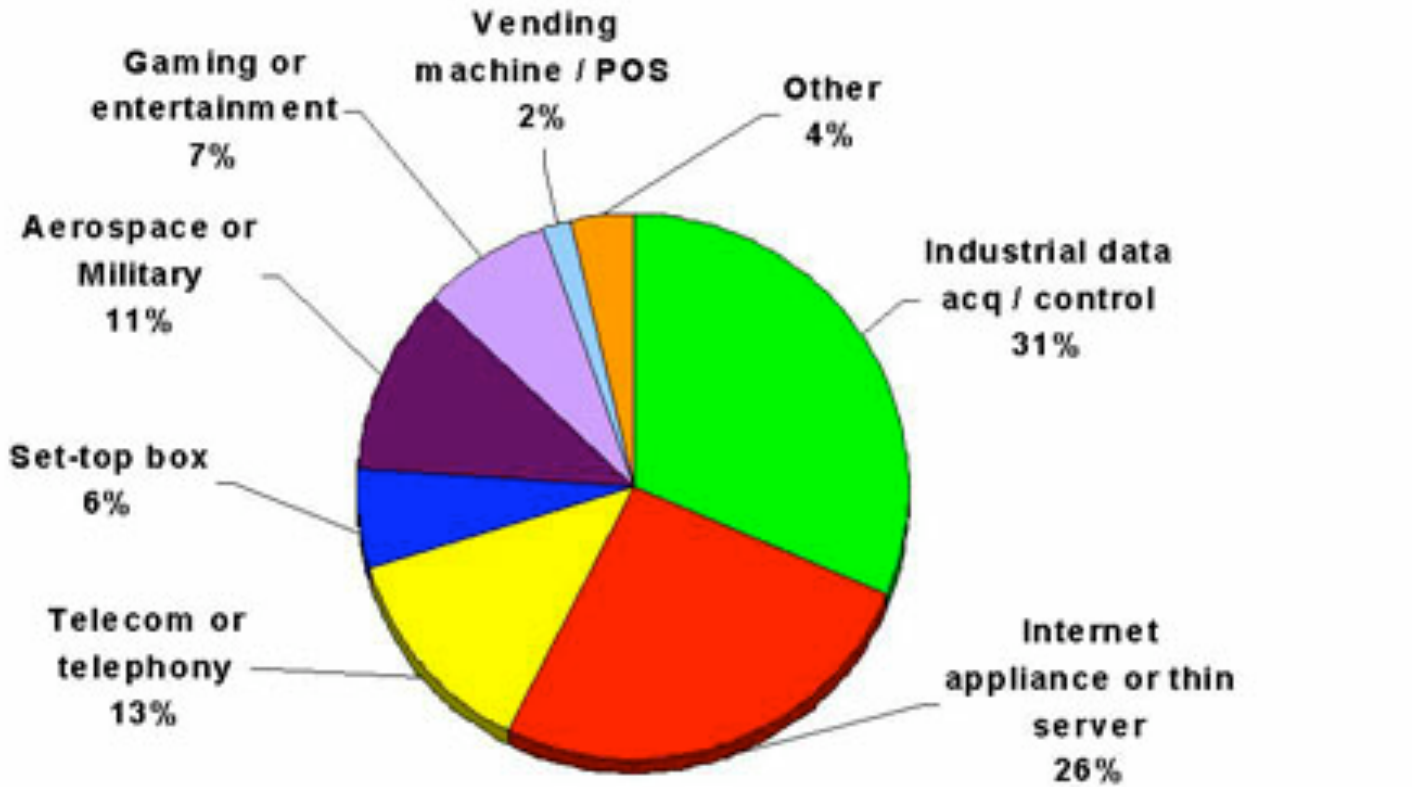
C or C++ for Development?

- In general C is a better choice
  - Advanced OOP features can cause code bloat
  - C++ compilers can generate many routines for a single function
  - Virtual methods and polymorphism slow program launch times significantly
- Size really *does* matter

What's It Being Used For?

- Control and Monitoring Applications
- Industrial Controllers
- TV Set Top Boxes (TiVO)
- Handheld PDA's
- Automobile Computers
- Telecomm and Networking Hardware
- Myriad and sundry other uses...

Target Applications



Cyclades TS-100

- Netlinos OS
- Dual 50MHz PowerPC
- 16MB SDRAM 4MB Flash
- Ethernet/Serial/RS485
- Size of a deck of cards

Sixnet VersaTRAK IPm

- Embedded Linux
- PowerPC CPU
- 16MB DRAM 16MB Flash
- Serial & Ethernet Ports
- Linux and IEC1131 Programming/Modbus

Sharp Zaurus

- Lineo Embedix
- 206 MHz StrongARM
- 64 MB DRAM 16MB Flash
- 3.5” display (320x240) 64K colors
- Opera browser & Qtopia
- QT/Embedded GUI

Cell and Web Phones

Telepong Mobile Phone

GITWiT Mobile Phone

Aplio/PRO IP Phone

TiVO Set Top Box

- Home grown port of Embedded Linux
- 54MHz PowerPC
- Multi GB hard disk

Axis 2120 Network Camera

- uCLinux
- Built-in Ethernet port
- 100 MHz ETRAX CPU
- 16 MB RAM

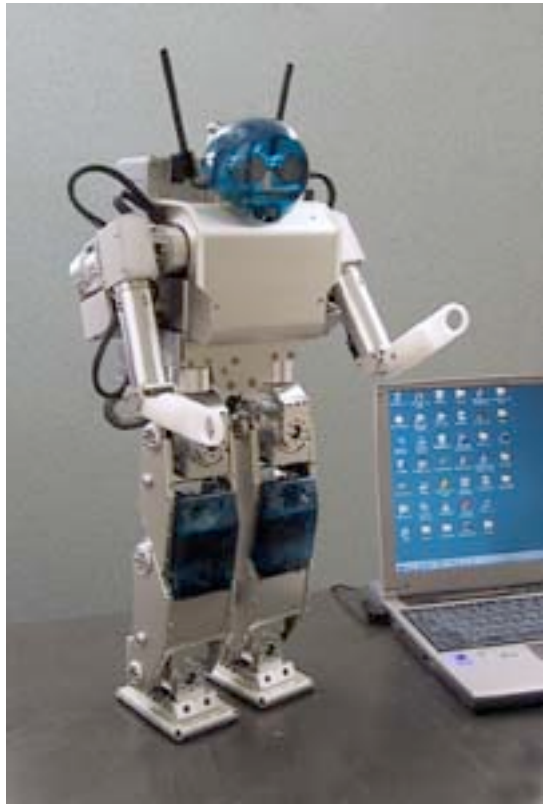
Humanoid Robots

- Isamu

- Univ. of Tokyo/Kawanda Ind.
- Dual Pentium CPU
- RT-Linux
- Height: 53 inches
- Weight: 121 lbs.

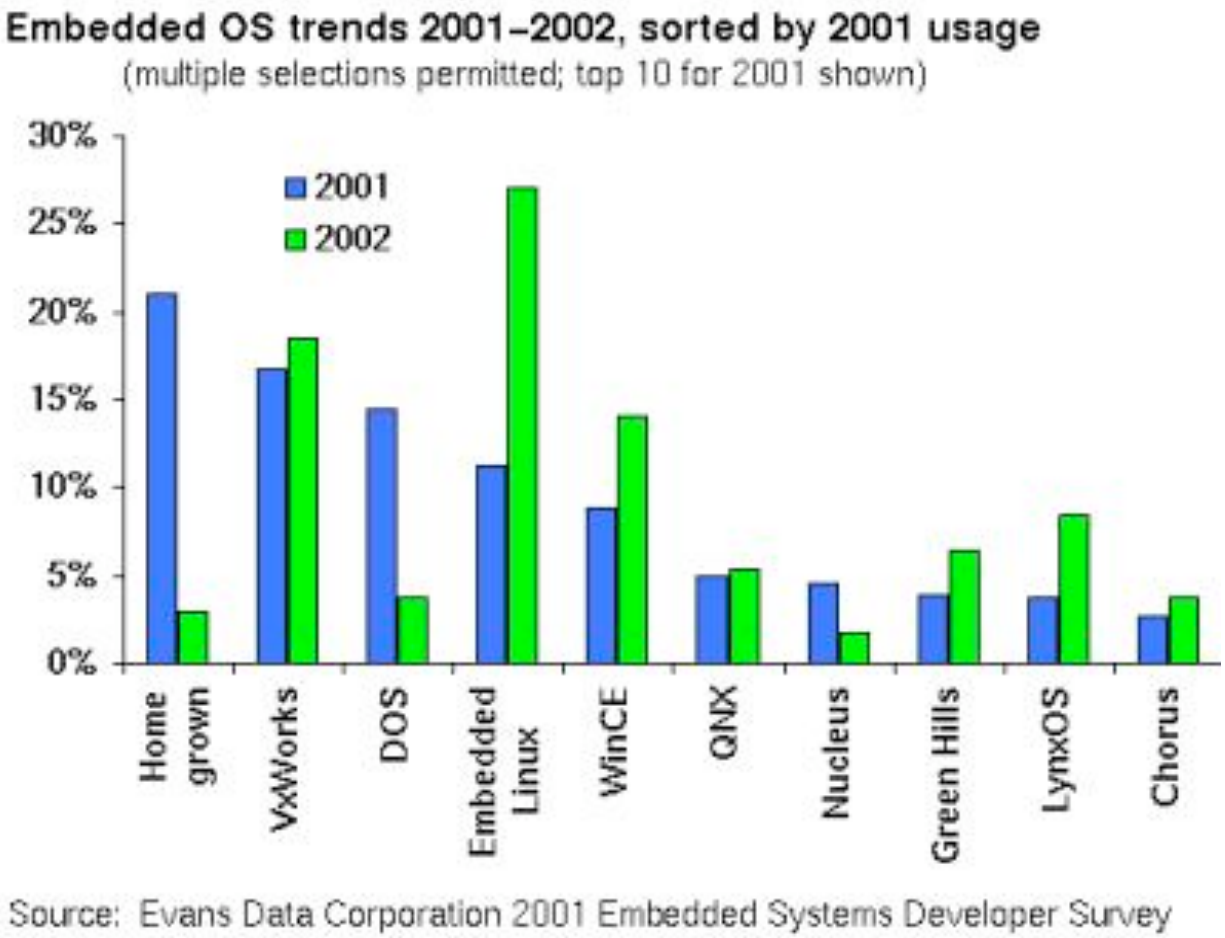
Humanoid Robots

- HOAP

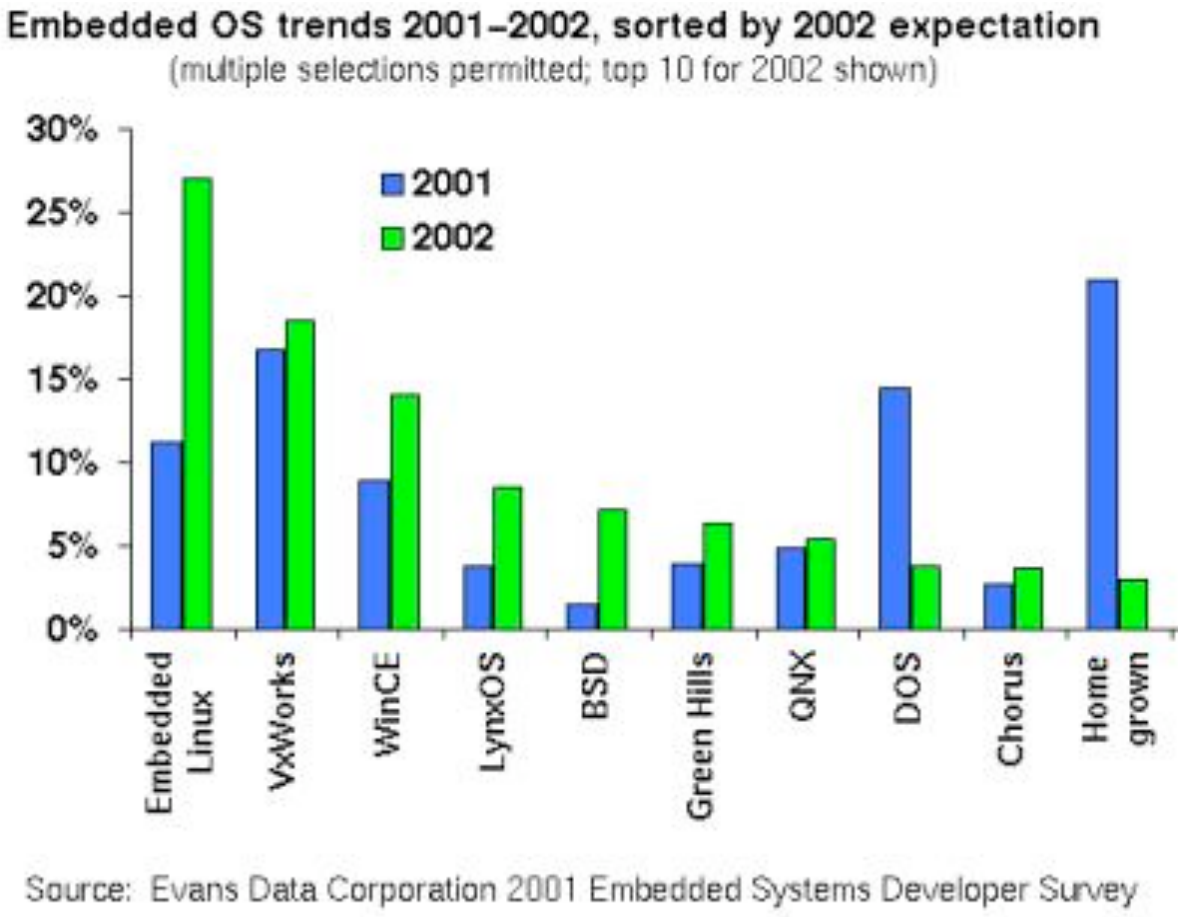


- Fujitsu
- RT-Linux
- Height: 48 cm
- Weight: 6 kg
- 100 units/yr

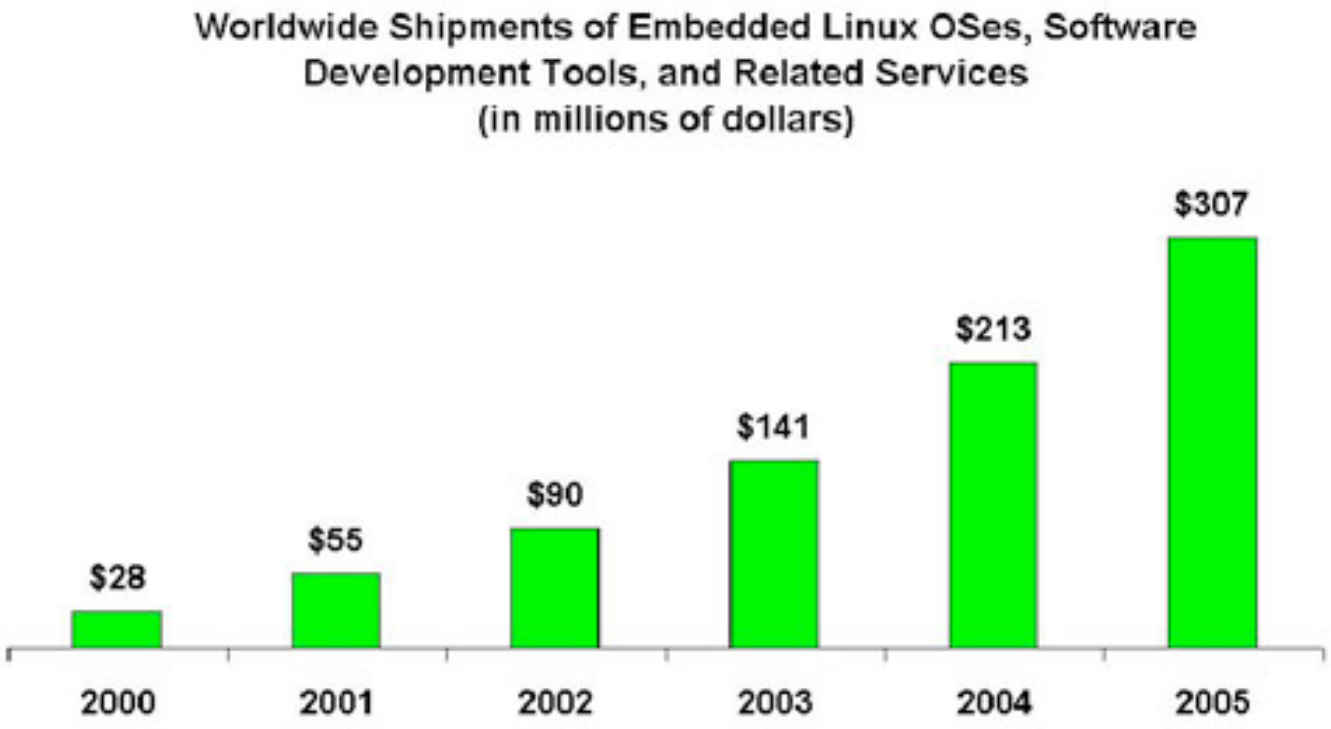
The Embedded OS Market 2001



The Embedded OS Market 2002



Growth of Embedded Linux



Source: Venture Development Corporation (VDC) 2000 Embedded Linux Market Study

For Further Info

Web Sites

- LinuxDevices.com
  http://www.linuxdevices.com/
- Embedded Linux Consortium
  http://www.embedded-linux.org/index.php3
- All Linux Devices
  http://alllinuxdevices.com/
- Embedded Linux StarGate
  http://linux-embedded.com/

For Further Info

Web Sites/Magazines

- Dr. Dobbs Embedded Systems
  http://www.ddjembedded.com/about/
- Embedded Linux Journal
  http://embedded.linuxjournal.com/
- Embedded Systems Programming
  http://www.embedded.com/mag.htm

References

Embedded Operating Systems and Linux
Bill Latura, April 23, 2002