OS Organization

User and Kernel Mode
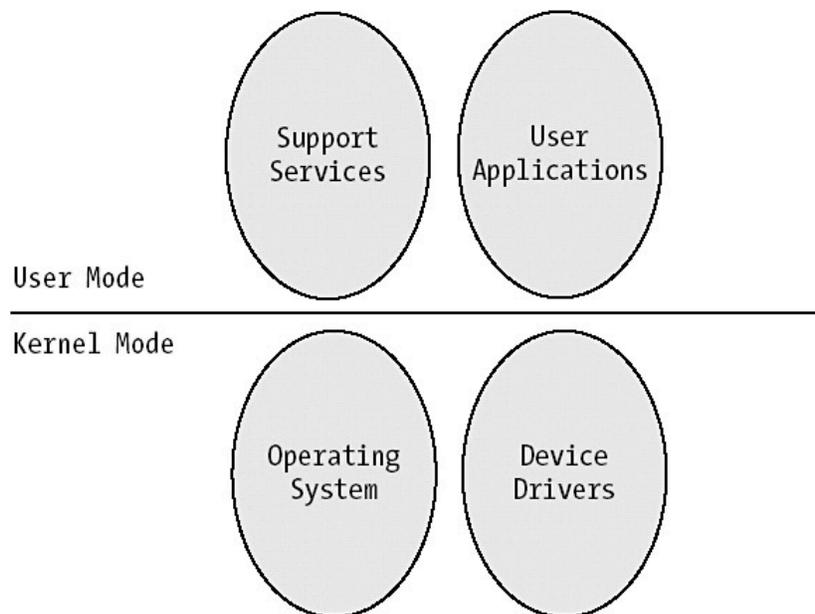
System Calls
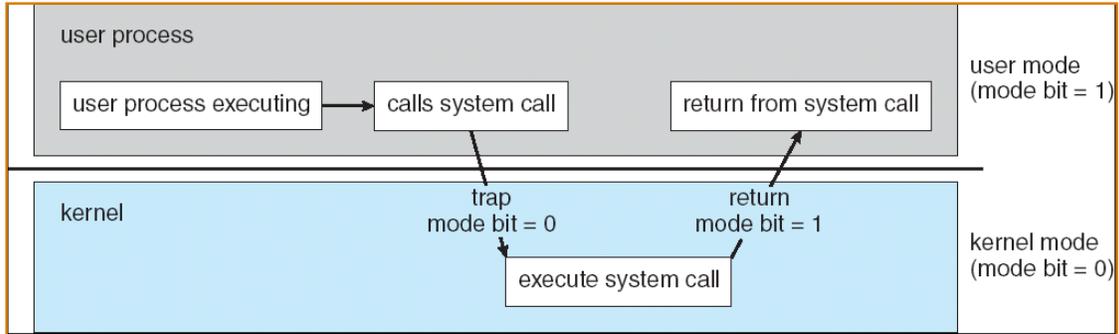
# Making a System Call
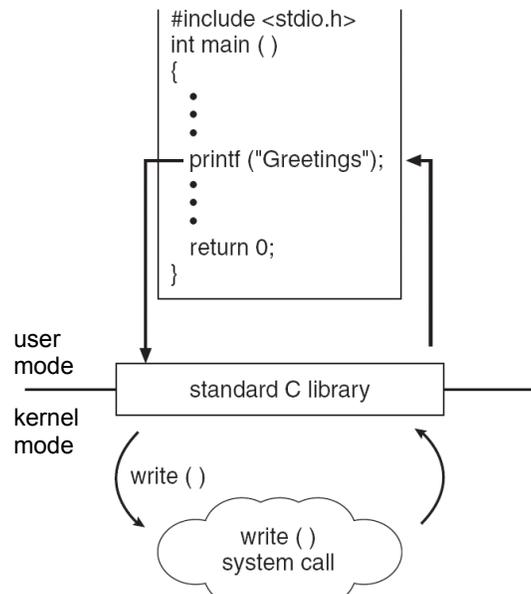
Example System Calls

	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitlializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

System Call Implementation
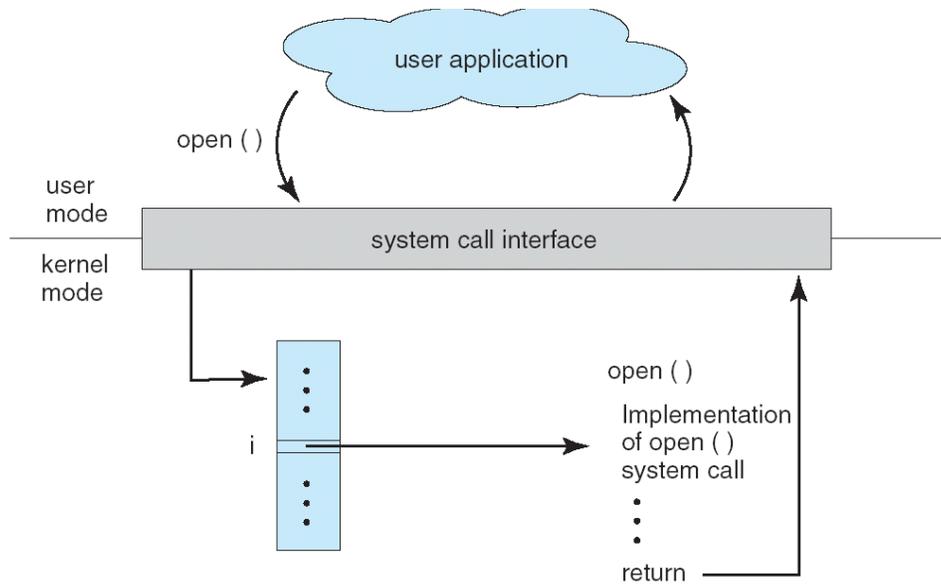
Traps

Memory Addresses

Trap Vector

|	
0: 0x00080000	Illegal address
1: 0x00100000	Memory violation
2: 0x00100480	Division by zero
3: 0x00123010	System call

⋮

Interrupts & I/O Control
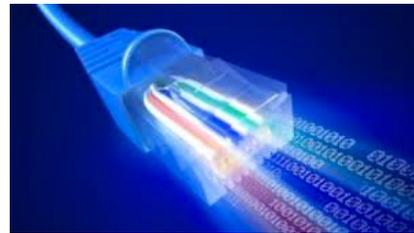
Interrupt Vector

0: 0x2ff080000	keyboard
1: 0x2ff100000	mouse
2: 0x2ff100480	hardware timer
3: 0x2ff123010	disk 1

Hardware Timer

OS Architecture: Mac OS X
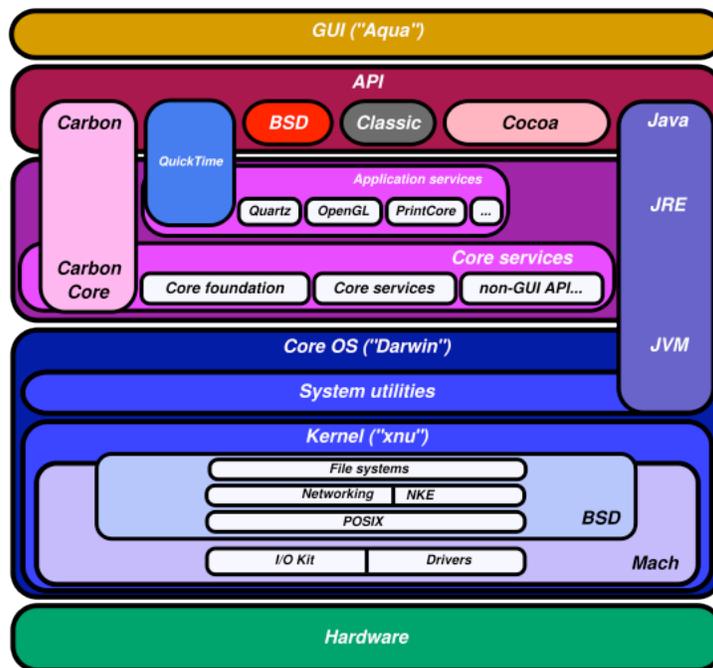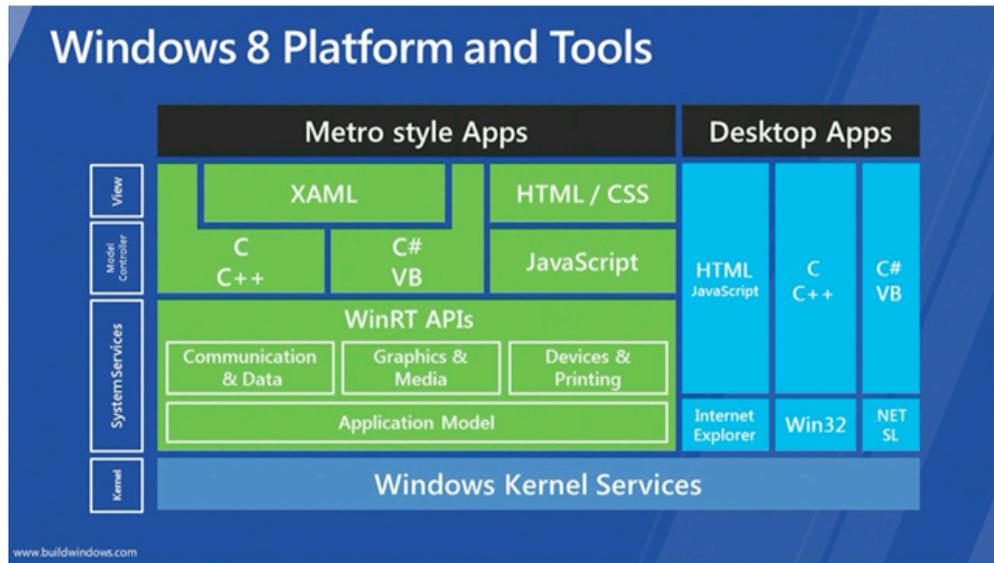
Monolithic Kernel Design

Layered Kernel Design

User programs
Device drivers
Virtual memory
I/O channel
CPU scheduler
Hardware

Microkernel Design

User Processes

System Processes

File System

High-Level Scheduling

Thread System

External Paging

Network Support

User Mode

Microkernel

Communication

Low-Level VM   Protection

Processor Control

Kernel Mode

Hardware
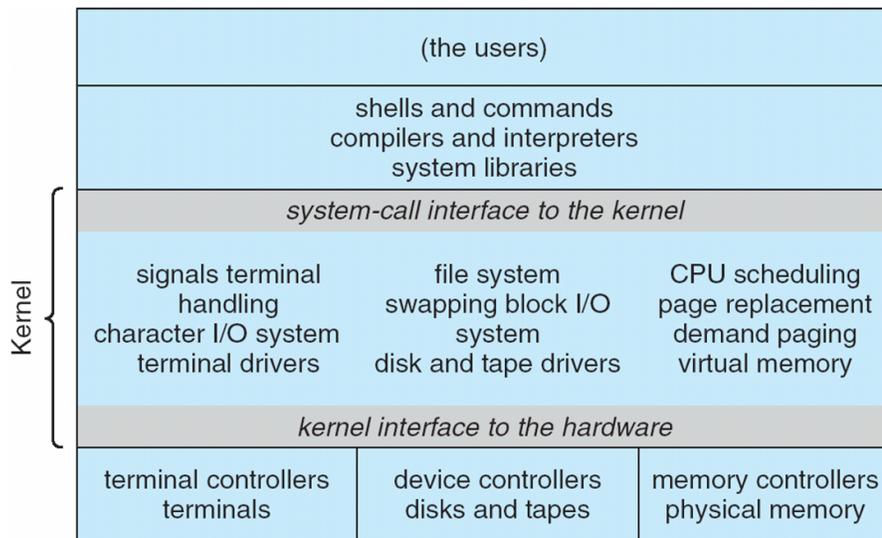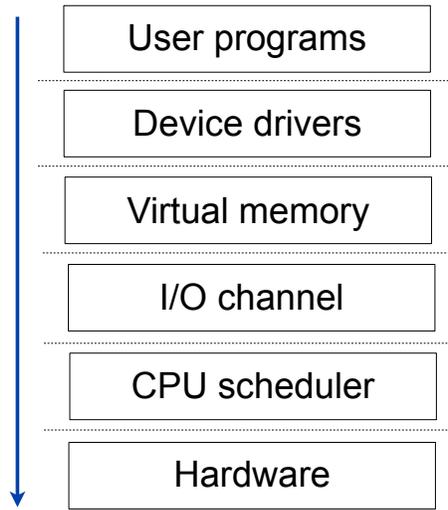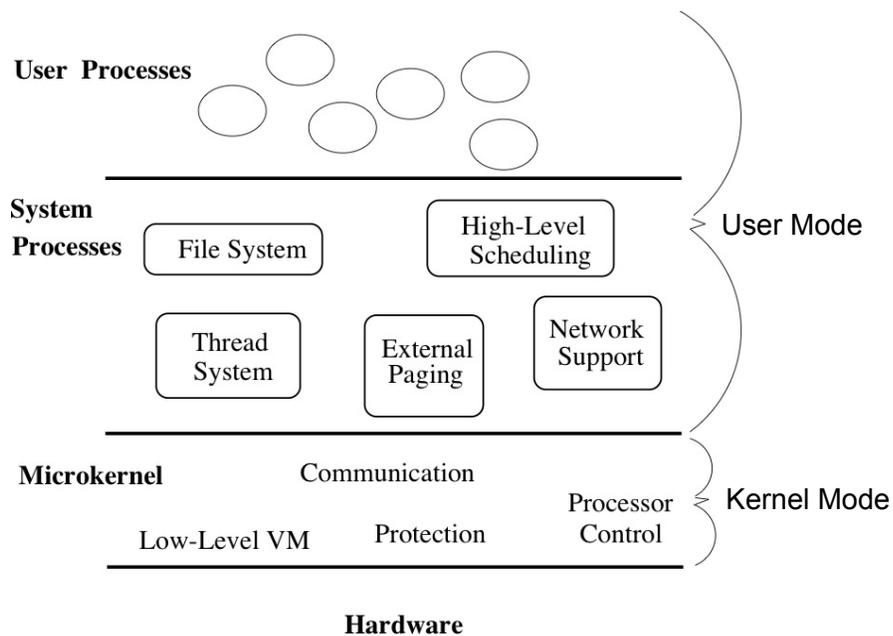
Hybrid Kernel Design in Mac OS X

Kernel Modules