User mode & Kernel mode

→ They are two distinct worlds within an OS.

→ Kernel mode has direct hardware access while user mode interacts with user

→ When CPU is in user mode it only does user mode work and when it is in kernel mode it only does kernel mode work.

→ CPU switch between user mode and kernel mode by software interrupt.

→ As user mode and kernel mode are also kept separate.

eg:- when in CLI we write command mkdir

↓

it will decode mkdir and will understand that a new folder or directory needs to be created

↓

Switches to kernel mode and goes to file management department and will make a directory.

↓

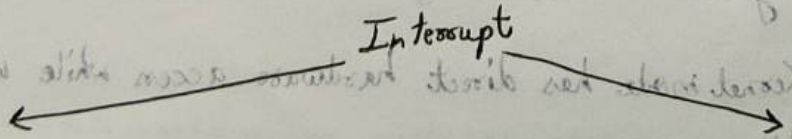
Then returns feedback that directory is created

Definitions

User mode: A restricted CPU mode where user applications run with limited access to system resources and cannot directly interact with hardware

Kernel mode: A privileged CPU mode where the OS runs with full access to hardware and system resources.

Interrupt: An interrupt is a signal sent to the CPU that temporarily halts the current process to attend to an urgent task after which normal execution resumes.



Hardware interrupt

→ Generated by hardware devices to signal the CPU.

Software interrupt

→ Generated by a program using a special instruction

→ Used to request services from the OS.

Extra

Interrupt processing

→ The basic method of interrupting the CPU is done by activating a control lines that connects the interrupt source to the CPU.

CPU => • recognizes the presence of interrupt
• execute a specific interrupt handling program
• determines the source of interrupt
• determines the address of the interrupt handling program

Interrupt Service Routine (ISR)

→ It is a software routine that is executed in response to an interrupt.

→ It is designed to handle the interrupt and restore the system to its normal state.

→ Functions:

• Handling the interrupt
• Restore to normal state after handling.
• Perform necessary actions to ensure system stability/ performance.

→ Types:

• Hardware ISR for Hardware interrupts
• Software ISR for Software interrupts

→ Benefits:

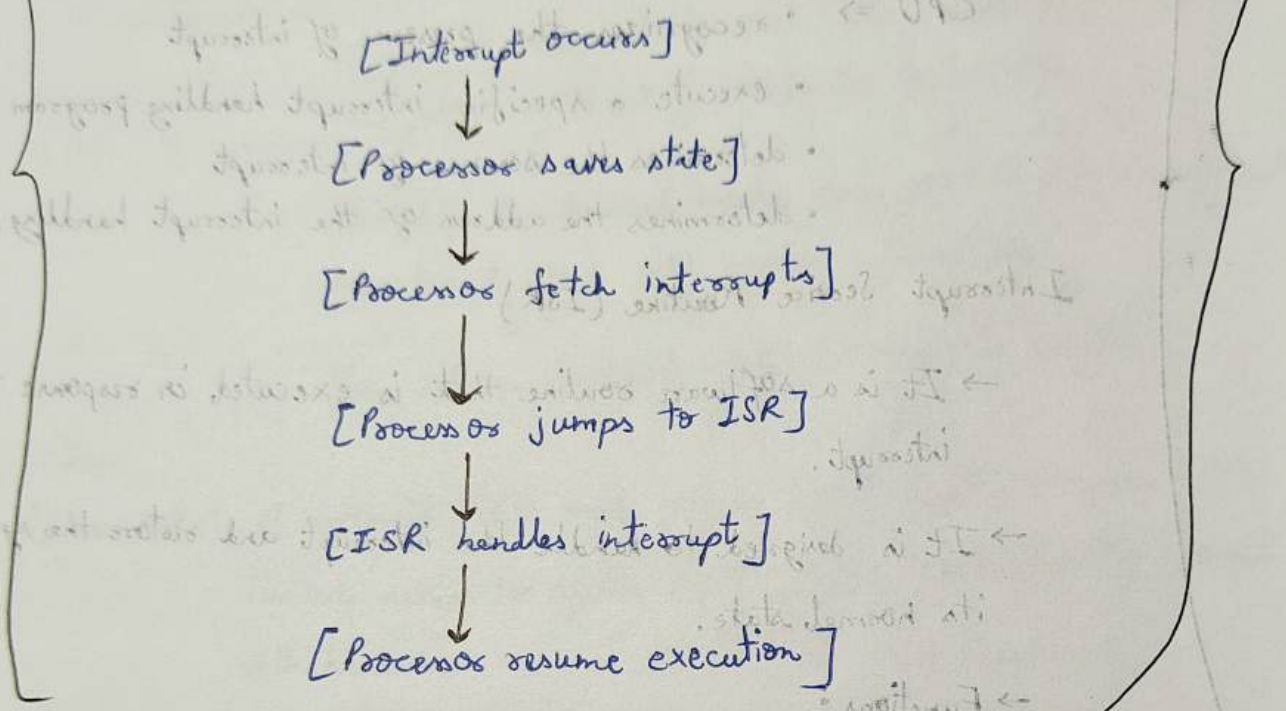
① System stability
② System performance.

Extra

Interrupt vector table

→ It is a data structure used in computer systems to manage interrupts.

→ It is a table that contains memory address of ISR

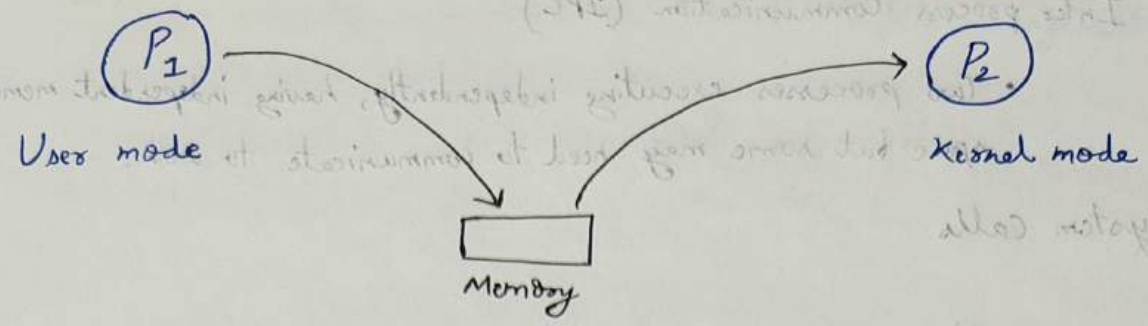


Interview Question

(1) User mode & Kernel mode communication??

→ There is a method provided by OS itself (in process management) which is called or done through interprocess communication (IPC).

↳ A mechanism that allows processes to communicate. It helps processes synchronize their activities, share information and avoid conflicts while accessing shared resources



Shared memory

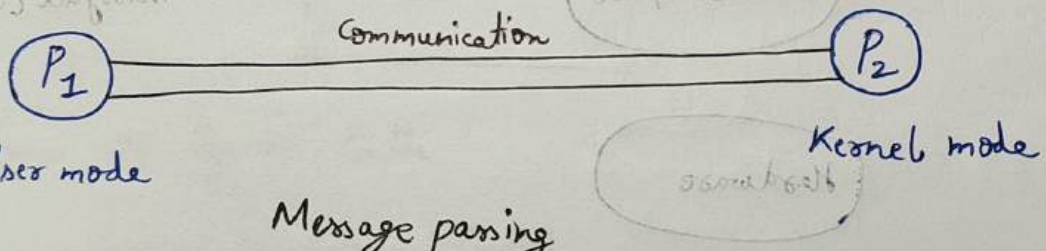
→ P_1 process is executing in user mode and P_2 process is executing in kernel mode.

→ So they can communicate by shared memory.

→ If P_1 wants to communicate to P_2 it writes on a shared memory (shared by P_1 & P_2).

→ P_2 then reads from that memory the message given by P_1.

→ Thus they communicate via shared memory.



Message passing

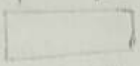
→ In this OS provides methods, functions or calls to communicate between P_1 & P_2

→ A logical pipe is established between processes or a channel for them to communicate.

→ System calls are used and there are some protocol or calls or policies through which they communicate.

Inter process communication (IPC)

- Two processes executing independently, having independent memory space but some may need to communicate to work.



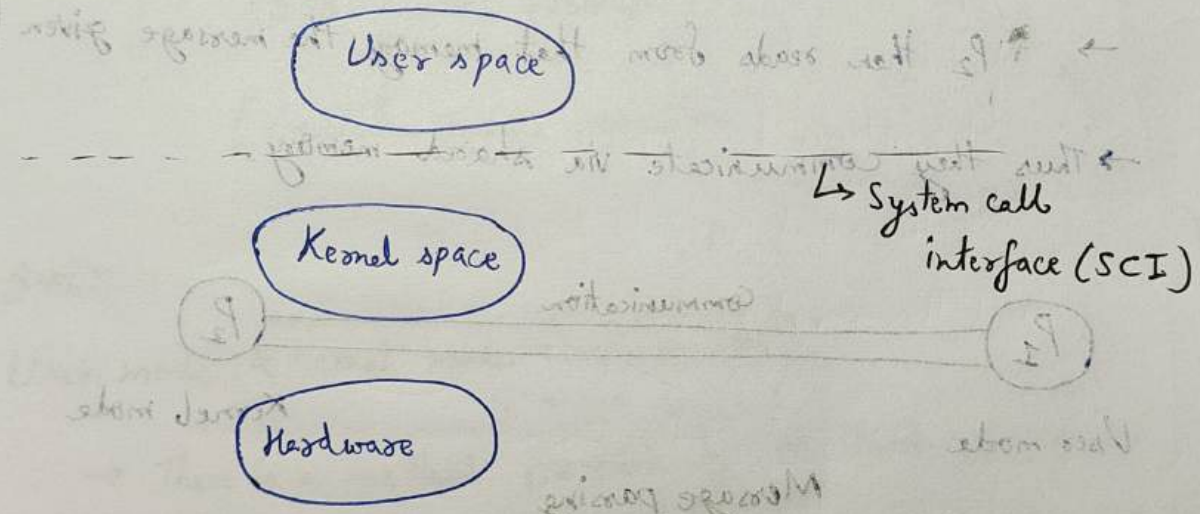
System calls

→ Using system calls apps interact with kernel.

→ A system call is a mechanism using which a user program can request a service from the kernel for which it does not have the permission to perform.

→ System calls are the only way through which a process can go into kernel mode from user mode.

→ System calls are implemented in C.



eg:- for command mkdir movies.

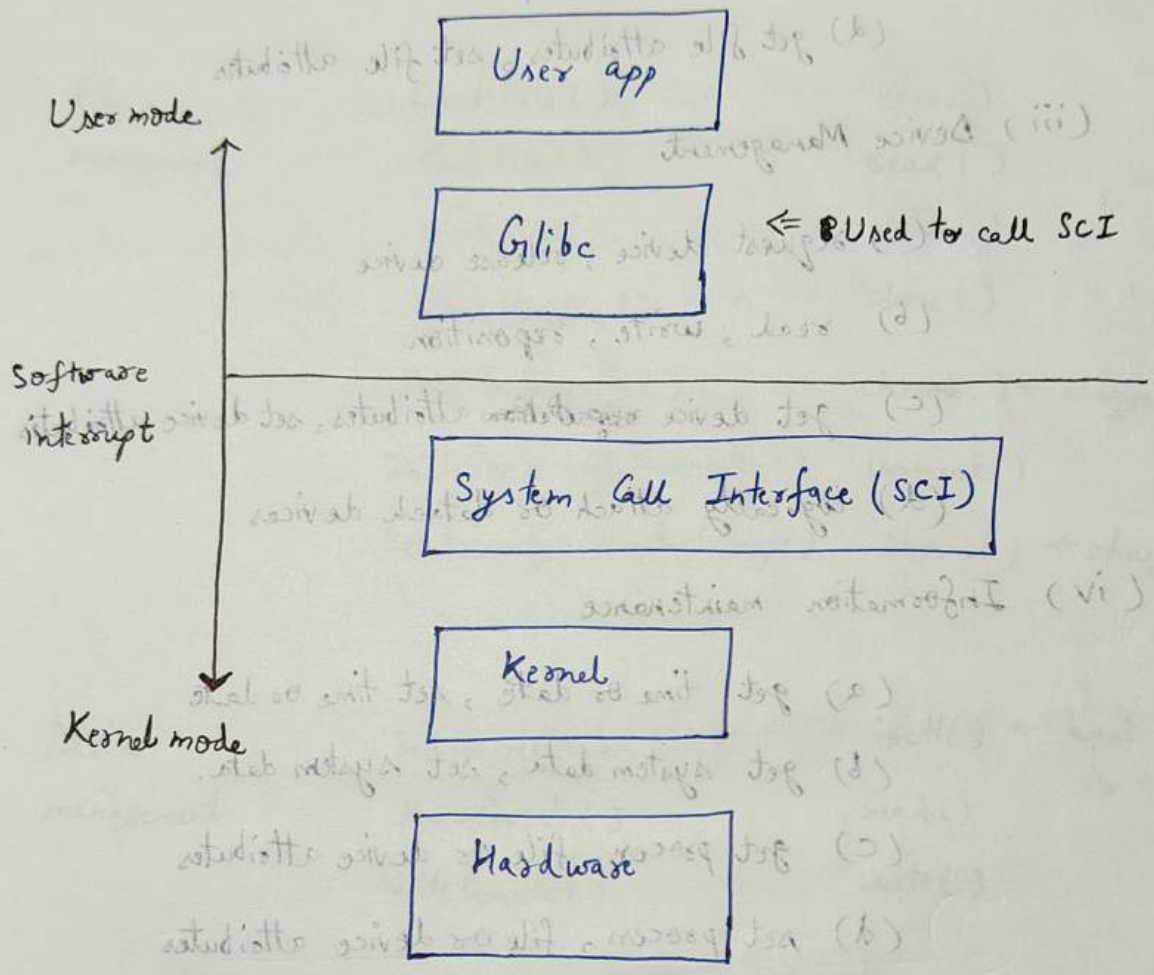
This command goes to SCI to take it to that space in kernel where implementation of mkdir is written.

SCI will tell to look for this implementation in kernel space and then kernel space will run this implementation then a directory is created.

→ Mkdir indirectly calls kernel and asked the file management module to create a new directory.

→ Mkdir is just a wrapper of actual system calls.

→ Mkdir interacts with kernel using system calls.



Types of System calls

(i) Process Control

(a) end, abort

(b) load, execute

(c) create process, terminate process

(d) get process attributes, set process attributes

(e) Wait for time

(f) wait event, signal event

(g) allocate and free memory.

(ii) File Management

(a) create file, delete file

(b) open, close

(c) read, write, reposition

(d) get file attributes, set file attributes

(iii) Device Management

(a) request device, release device

(b) read, write, reposition

(c) get device attributes, set device attributes

(d) logically attach or detach devices

(iv) Information maintenance

(a) get time or date, set time or date

(b) get system data, set system data.

(c) get process, file or device attributes

(d) set process, file or device attributes

(v) Communication management

(a) create, delete communication connection

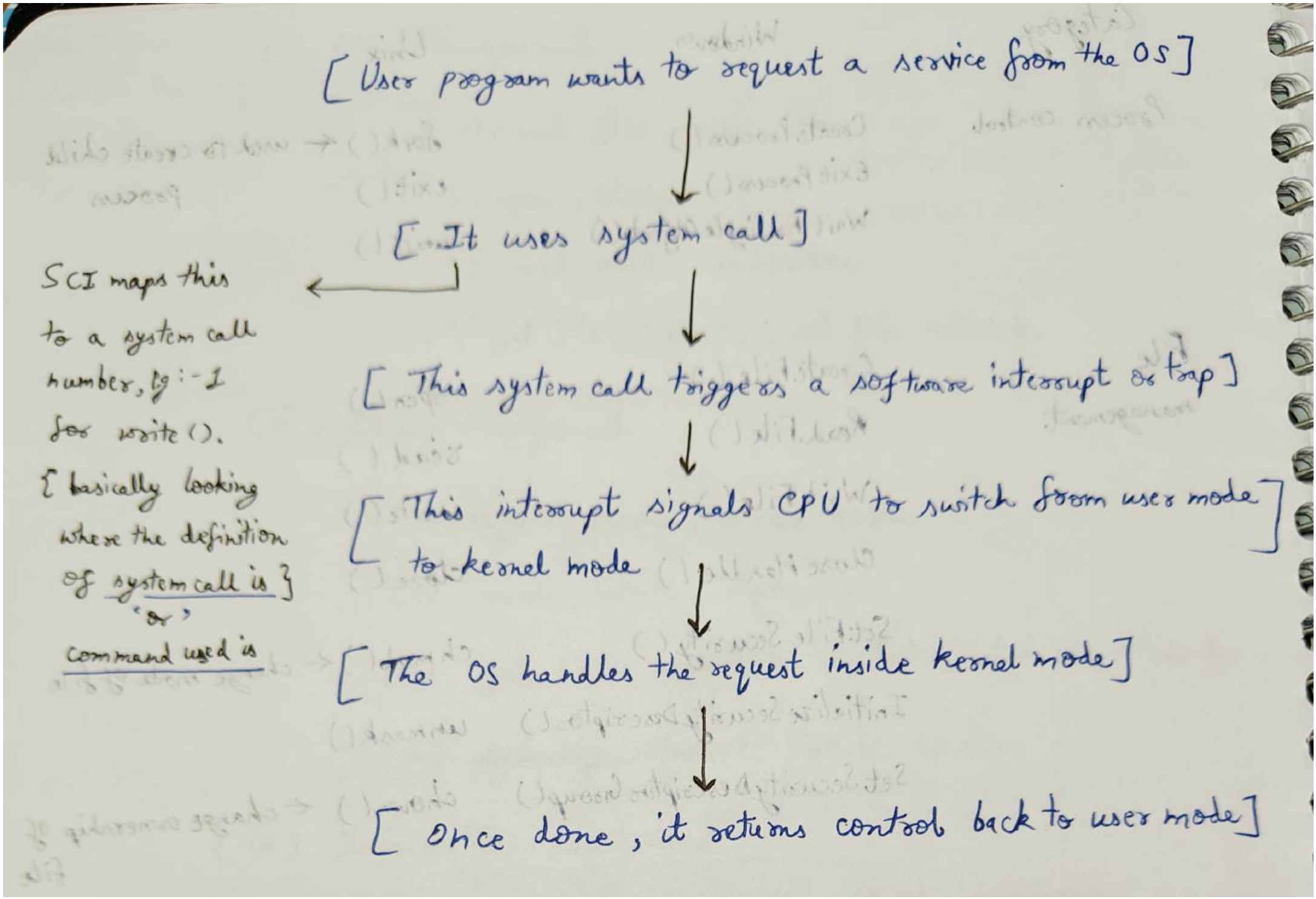
(b) send, receive messages.

(c) transfer status information

(d) attach or detach remote devices

Examples

Category	Windows	Unix
Process control	CreateProcess()	fork() ← used to create child process
	ExitProcess()	exit()
	WaitForSingleObject()	wait()
File management	CreateFile()	open()
	ReadFile()	read()
	WriteFile()	write()
	CloseHandle()	close()
	SetFileSecurity()	chmod() ← change mode of file
	InitializeSecurityDescriptor()	umask()
	SetSecurityDescriptorGroup()	chown() ← change ownership of file
Device management	SetConsoleMode()	ioctl() ← input output control. Used to talk to device drivers
	ReadConsole()	read()
	WriteConsole()	write()
Information management	GetCurrentProcessID()	getpid()
	SetTimer()	alarm()
	Sleep()	sleep()
Communication	CreatePipe()	pipe() ← used to implement message passing
	CreateFileMapping()	shmget() ← used to implement shared memory
	MapViewOfFile()	mmap()


[User program wants to request a service from the OS]
[It uses system call]
SCI maps this to a system call number, Eg:- 1 for write().
[basically looking where the definition of system call is]
'or'
command used is
[This system call triggers a software interrupt or trap]
[This interrupt signals CPU to switch from user mode to kernel mode]
[The OS handles the request inside kernel mode]
[Once done, it returns control back to user mode]