Department of Electronics & Communication Engg.

Course : Operating Systems -15EC553. **Sem.: 5th (2018-19 ODD)**

Course Coordinator:

Prof. Nyamatulla M Patel

Operating Systems-15EC553

Module-01
Overview of Operating System

Introduction

- Twin aims of an operating system are:
  - User convenience
  - System performance
    - Performance measures depend on the computing environment.
    - Number of requests serviced per unit time, etc

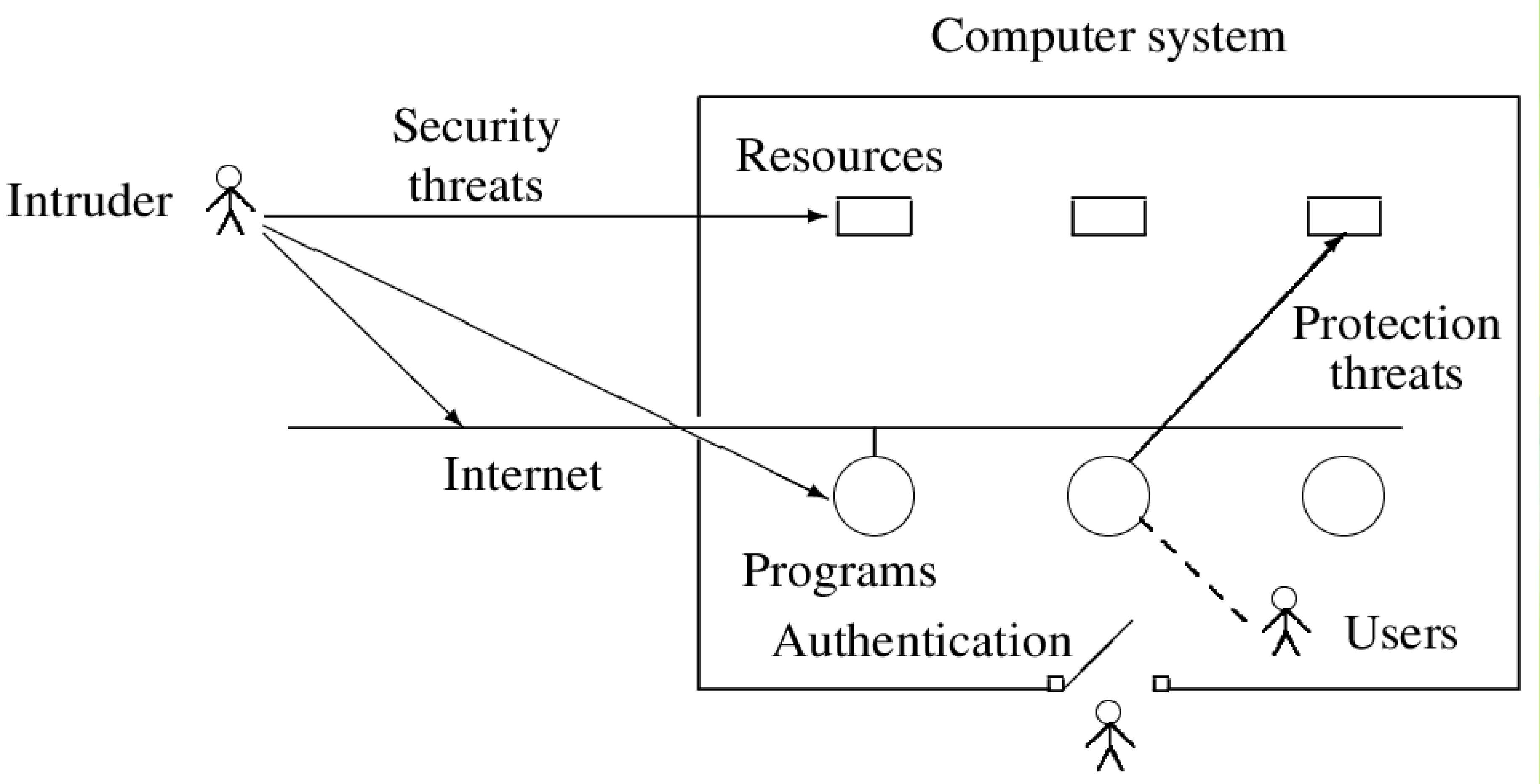
Fundamental tasks of an OS

- Management of Programs
  * Organize their execution by sharing the CPU
  * Ensure good user service and efficient use.

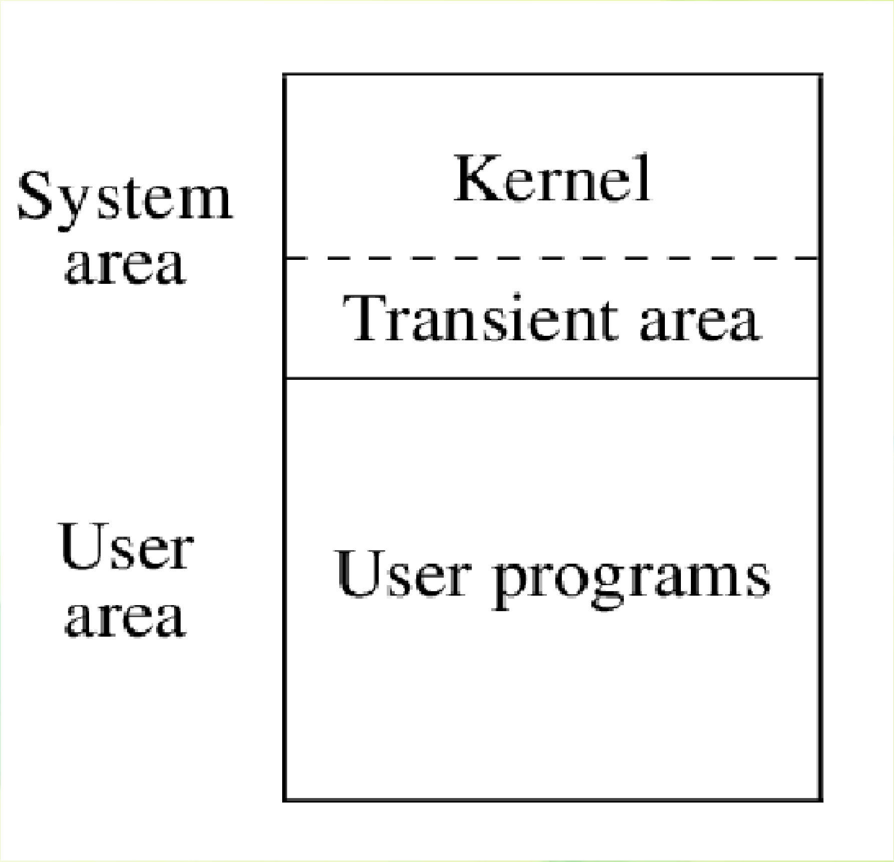
- Management of Resources
  * Efficient allocation/de-allocation without constraining user programs.

- Security and Protection
  * Ensure absence of interference with programs and resources by entities within and outside the OS

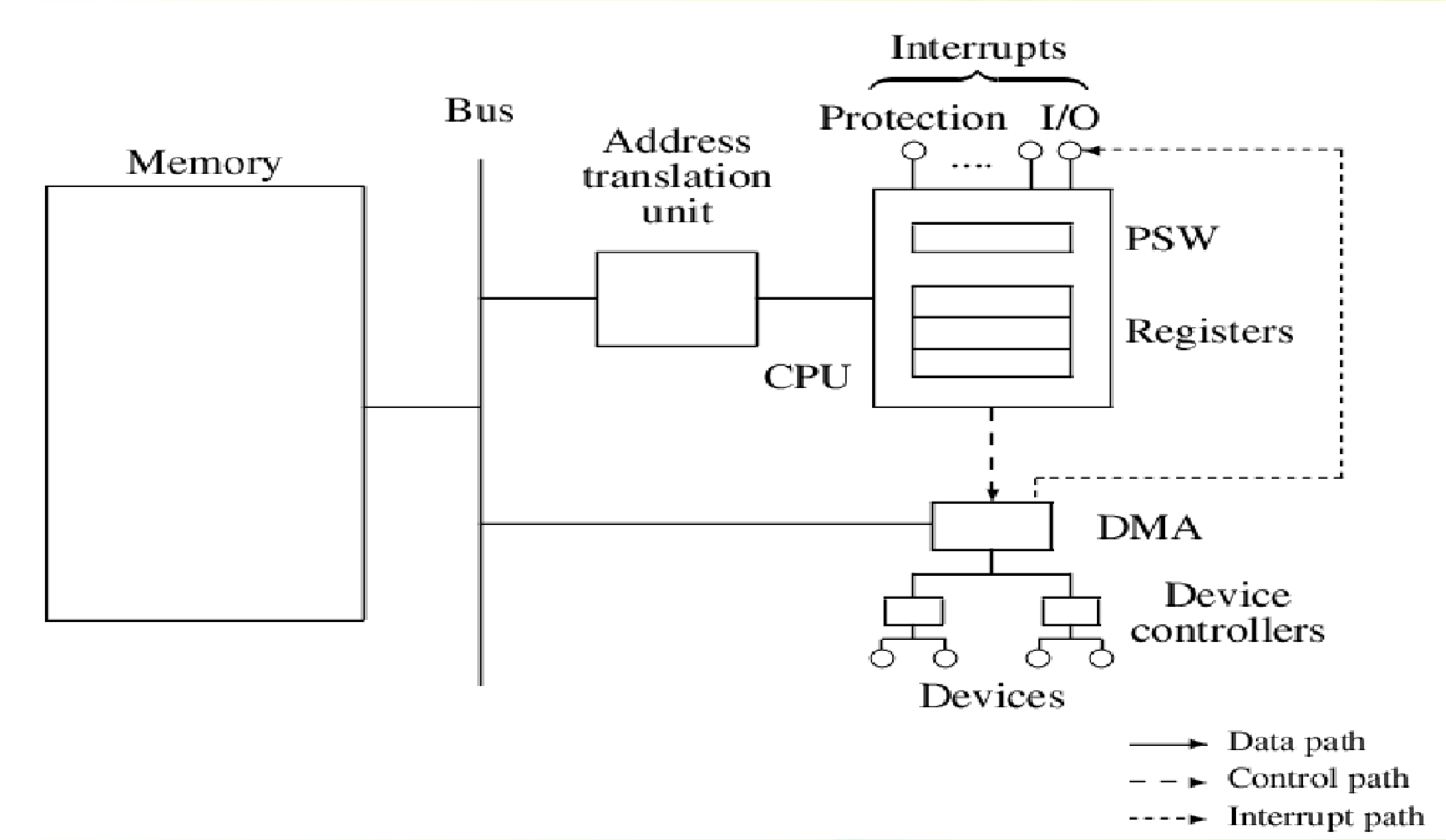
Overview of Security and Protection Threats



Memory utilization during operation of an OS



Model of a Computer System



Fields of Program Status Word (PSW)

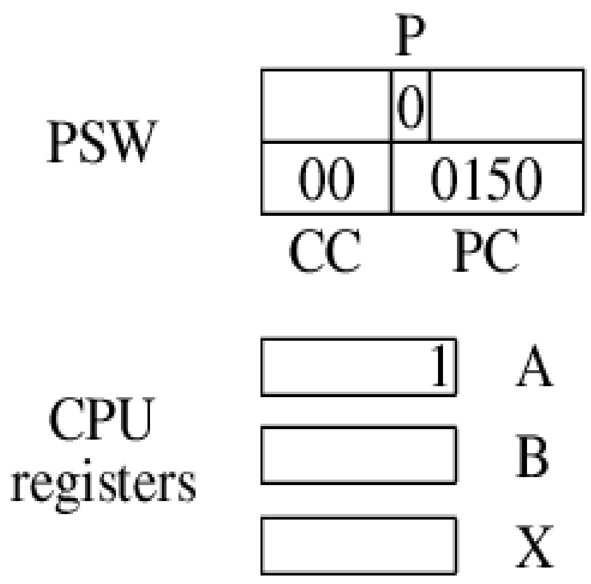
IM	P	IC	MPI	CC	PC

IM : Interrupt Mask
P : Privileged mode
IC : Interrupt Code
MPI : Memory Protection Information
CC : Condition Code
PC : Program Counter

(a) Program (b) State of CPU after executing COMPARE instruction

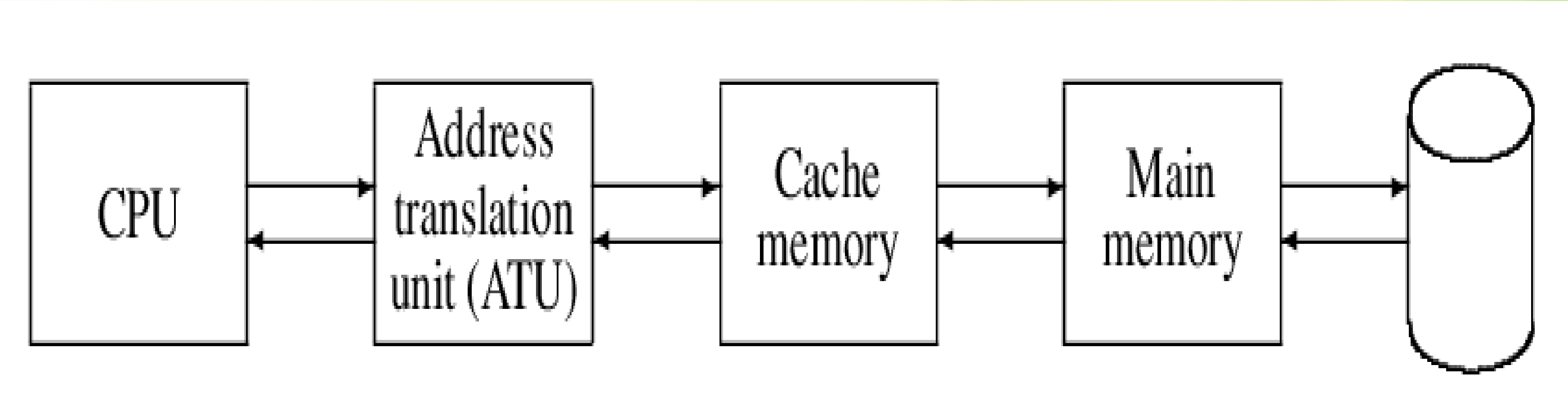
Address		Instruction	
0142		MOVE	A, ALPHA
0146		COMPARE	A, 1
0150		BEQ	NEXT
		...	
0192	NEXT		
		...	
0210	ALPHA	DS	1

(a)



(b)

Memory hierarchy containing cache, main memory and disk



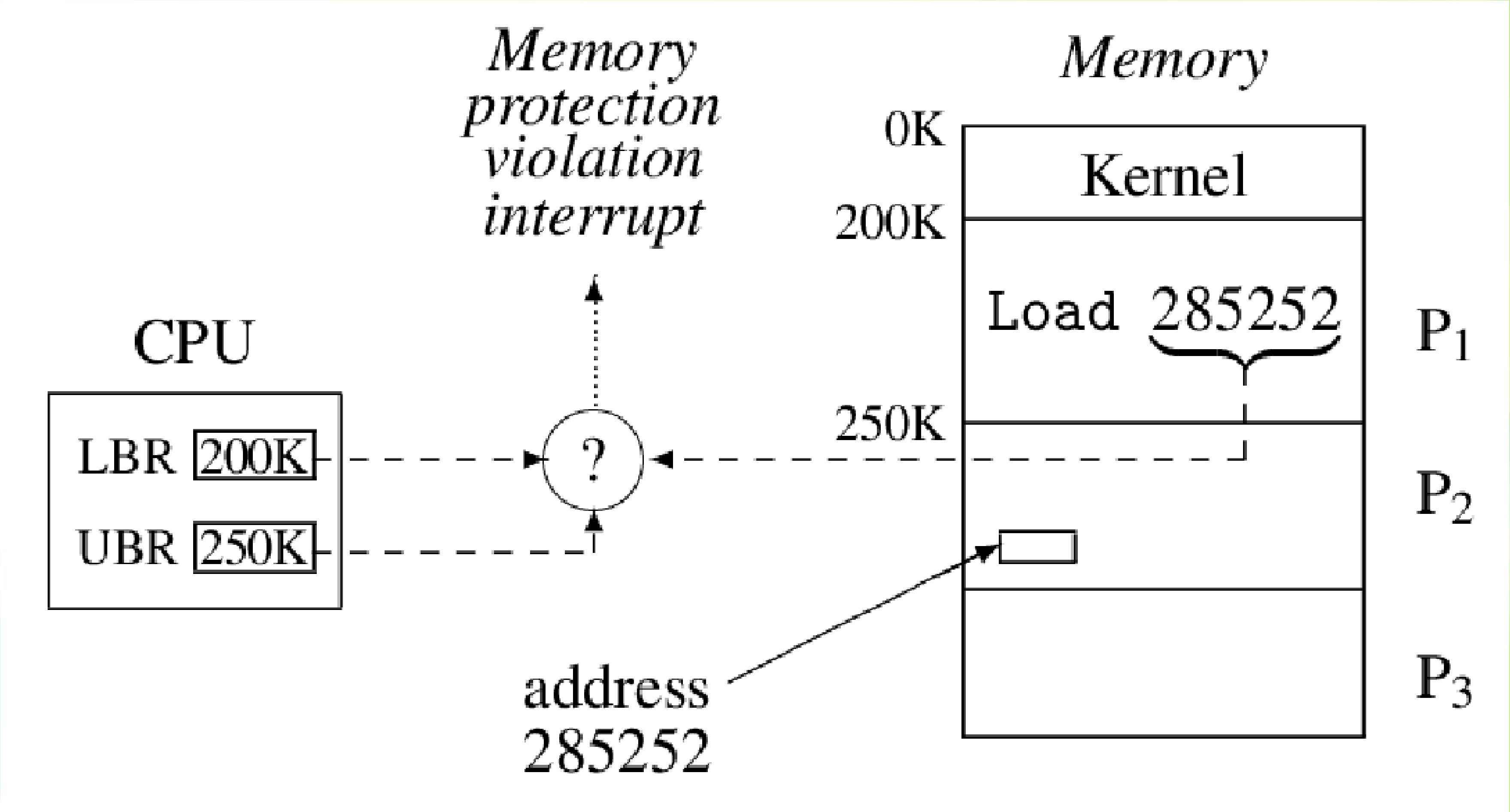
Memory hierarchy

- Cache memory
  * Organization – Cache block or cache line
    -- Inclusive or exclusive
    -- Direct, fully associative,
    -- Set associative

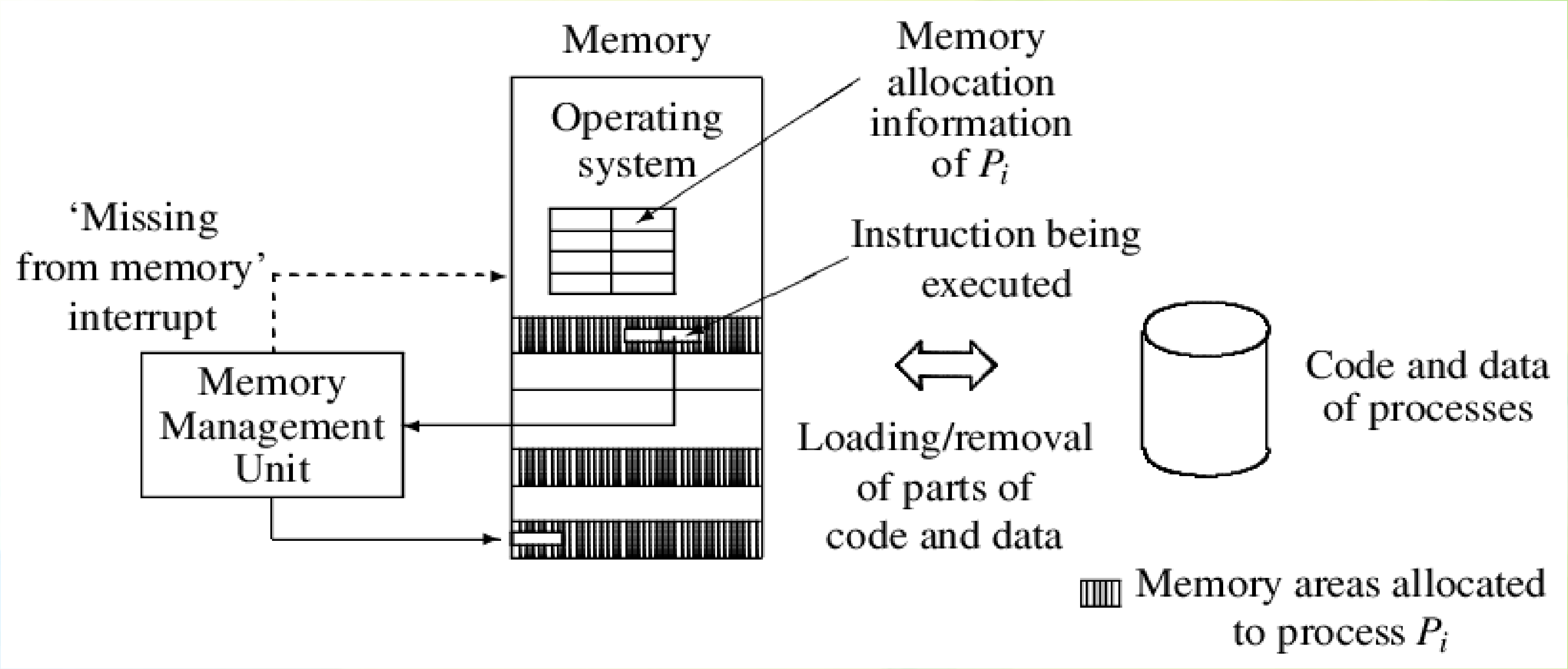
  * Different levels of caches – Why? How many levels?
  * Cache hit ratio – What factors influence it?

- Main memory
  * Memory protection
  * Virtual memory?

Memory protection using Bound registers



A schematic of virtual memory operation



Input / Output organization

- Involvement of the CPU in I/O operations
  -- Should be the minimum possible due to imbalance between CPU and I/O speeds.

  -- CPU should be free to execute instructions while I/O operations are in progress.

- Different I/O modes
  -- Programmed I/O
  -- Interrupt I/O
  -- Direct memory access (DMA)

Input Output Modes

I/O mode	Description
Programmed I/O	Operands of an I/O instruction indicate details of an I/O operation. CPU decodes the instruction and sends a signal to an I/O device. Data transfer between the I/O device and the memory takes place through the CPU. CPU cannot perform any other operation while I/O is in progress.
Interrupt I/O	CPU executes an I/O instruction. It starts an I/O operation and frees the CPU to execute other instructions. An interrupt is raised when a data byte is to be transferred between the I/O device and the memory. The CPU executes an interrupt processing routine that transfers the byte.
Direct Memory Access (DMA)	An I/O instruction indicates the I/O operation to be performed and also the number of bytes to be transferred. I/O operation starts when the instruction is executed. Data transfer between the device and memory takes place over the system bus. The CPU is not involved in data transfer. An interrupt is raised when the transfer of all bytes is complete.

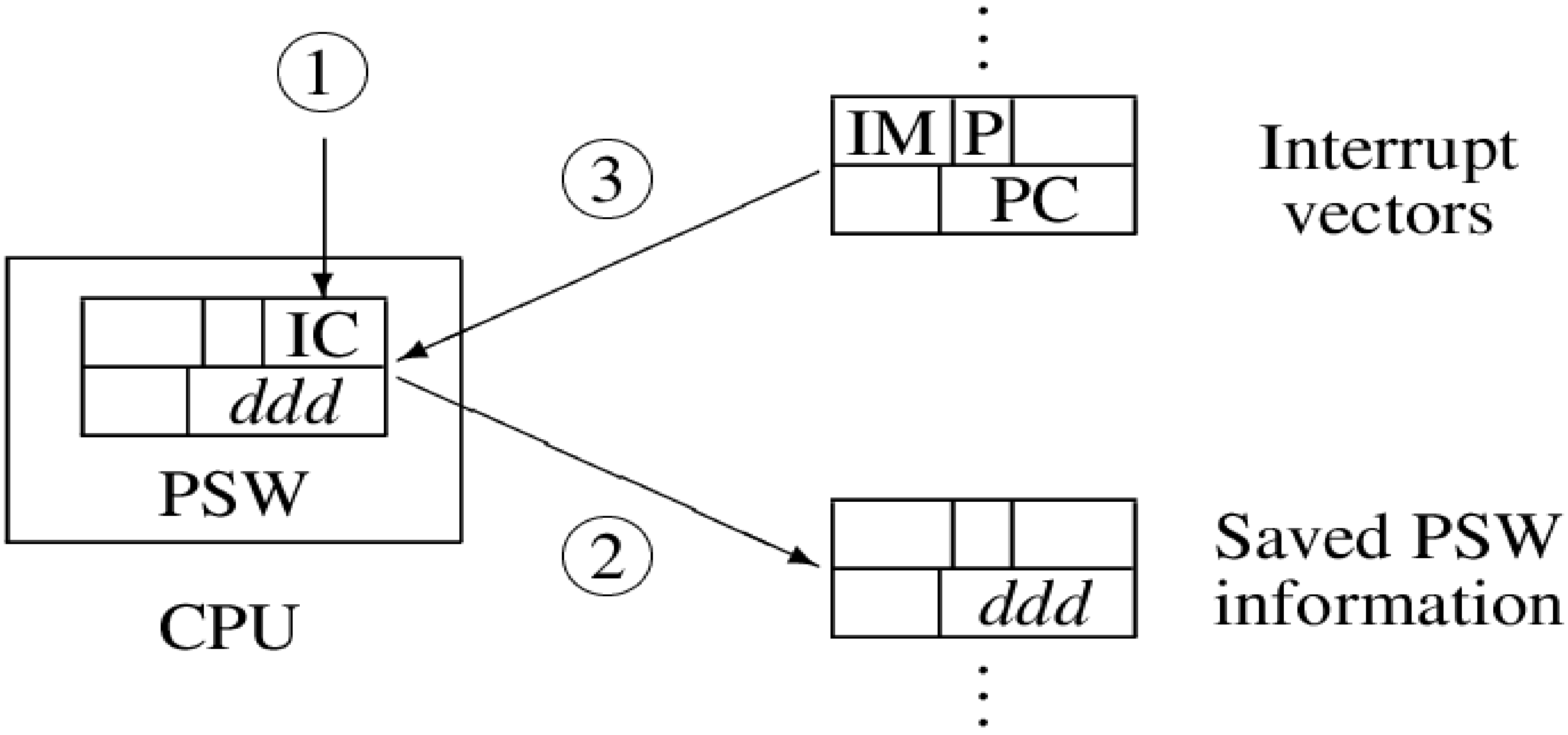
Interrupts

- An interrupt signals the occurrence of an event to the CPU.
- The CPU is diverted to execution of an OS routine.
- Different classes of interrupts convey occurrence of different kinds of events

Classes of Interrupts

Class	Description
Program interrupt	Caused by conditions within the CPU that require supervisor attention, for example, arithmetic exceptions like overflow and loss of precision, addressing exceptions and memory protection violations. An interrupt called a *software interrupt* is caused by execution of a special instruction called the software interrupt instruction.
I/O interrupt	Caused by conditions like I/O completion and malfunctioning of I/O devices.
Timer interrupt	Raised by the interval timer of the computer system.

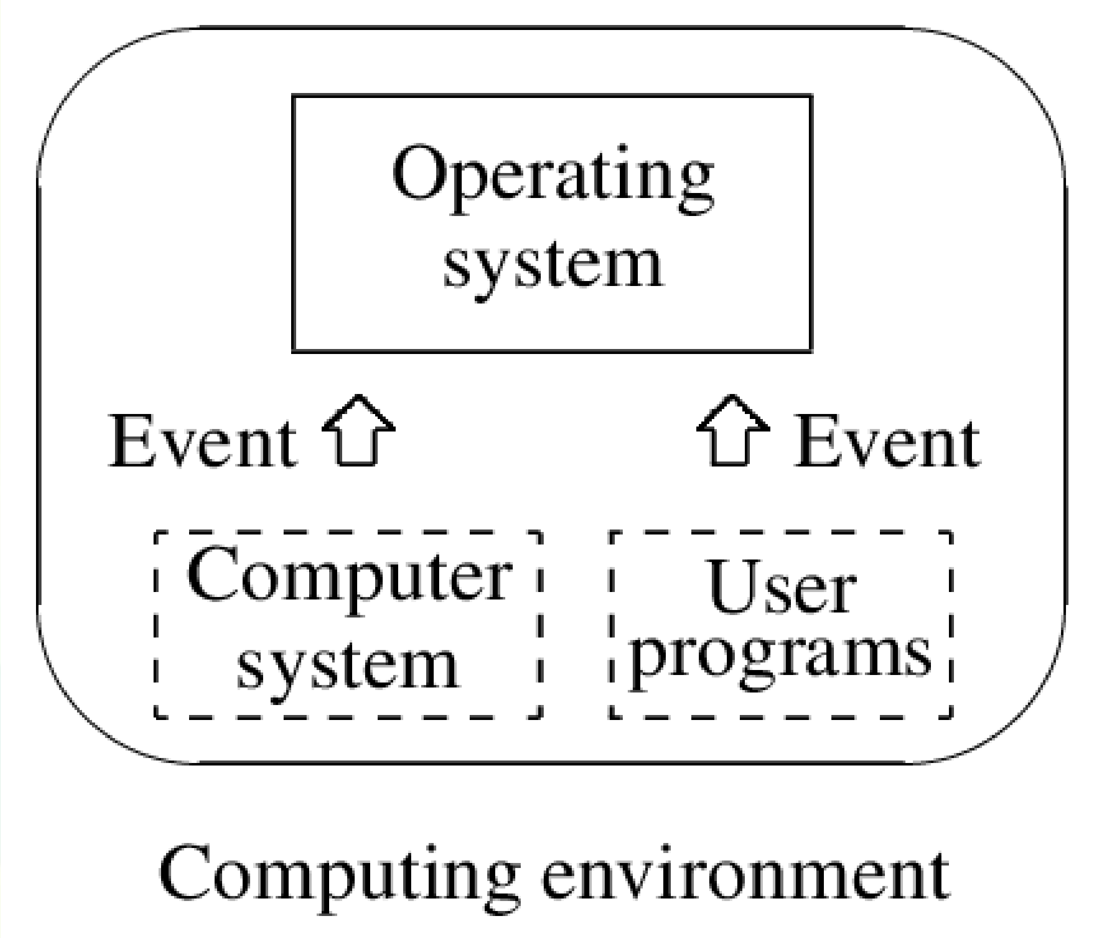
The interrupt action



Steps in Interrupt action

Step	Description
1. Set Interrupt code	The interrupt hardware for the chosen interrupt forms a code describing the cause of the interrupt. This code is stored in the *interrupt code* field of the PSW. For example, for an 'I/O completion' interrupt, the code could be the address of the I/O device causing the interrupt.
2. Save the PSW	The PSW is copied into the *saved PSW information* area in the unit corresponding to the interrupt class.
3. Load interrupt vector	The interrupt vector corresponding to the interrupt class is accessed. Information from the interrupt vector is loaded into the corresponding fields of the PSW. This action transfers control to an appropriate interrupt processing routine.

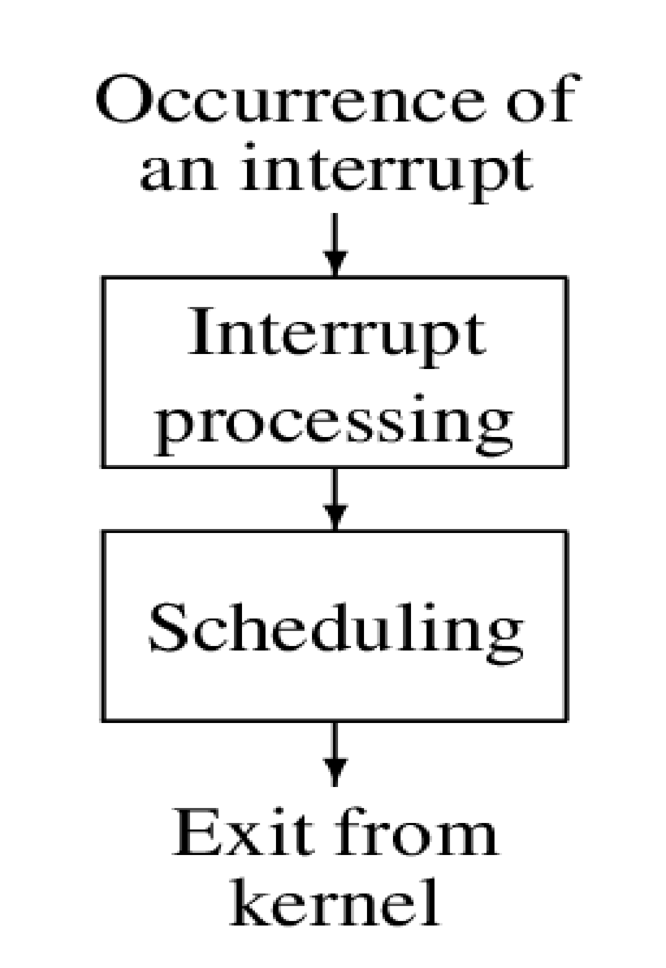
An operating system in its computing environment



System Call

- A system call is a request made by a program through a special instruction called a `software interrupt' instruction.
- The software interrupt is a program interrupt.
- When a software interrupt occurs, the interrupt hardware transfers control to a routine of the OS.
- The operand of the software interrupt instruction indicates what kind of request is being made by a program

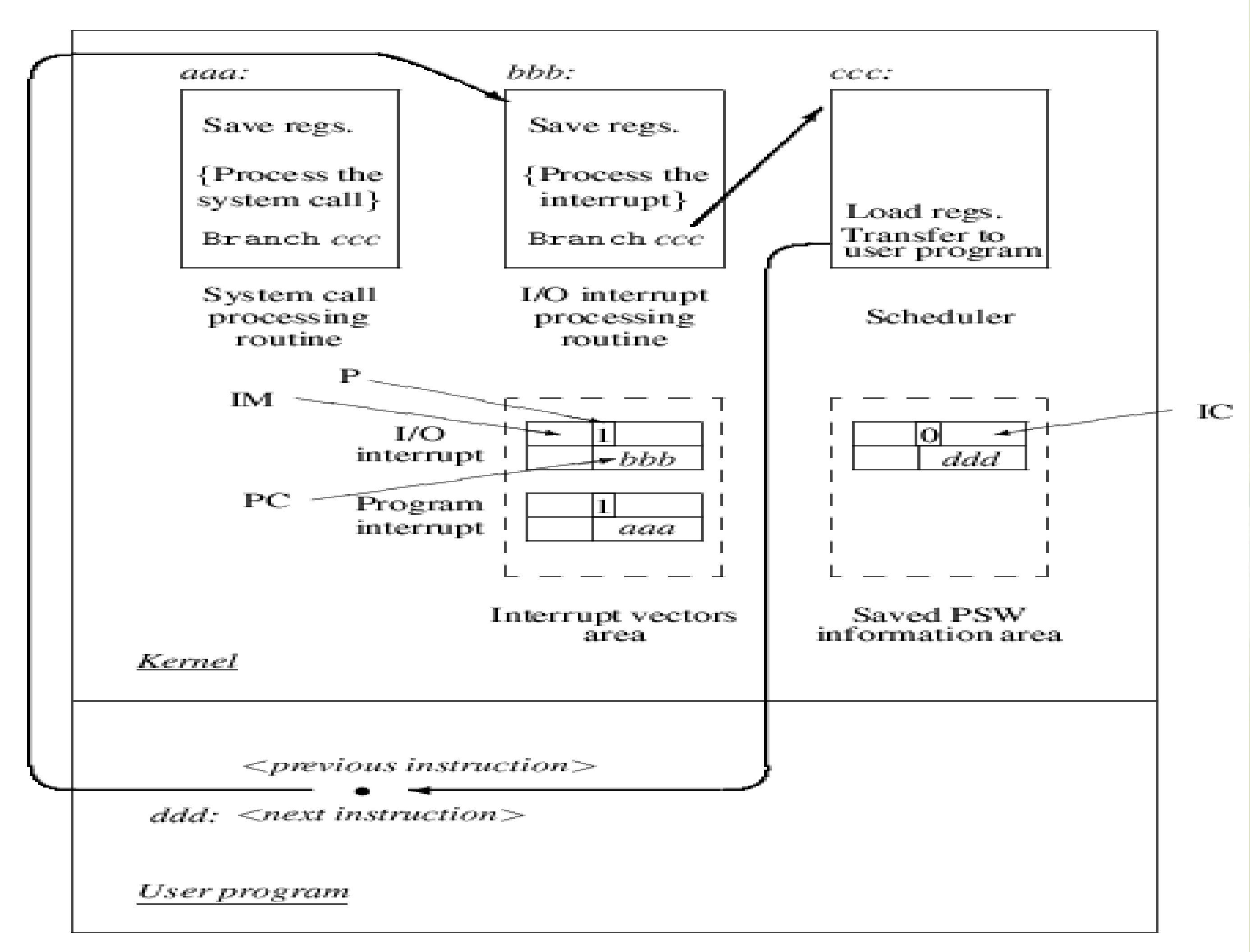
Interrupt driven operation of a kernel



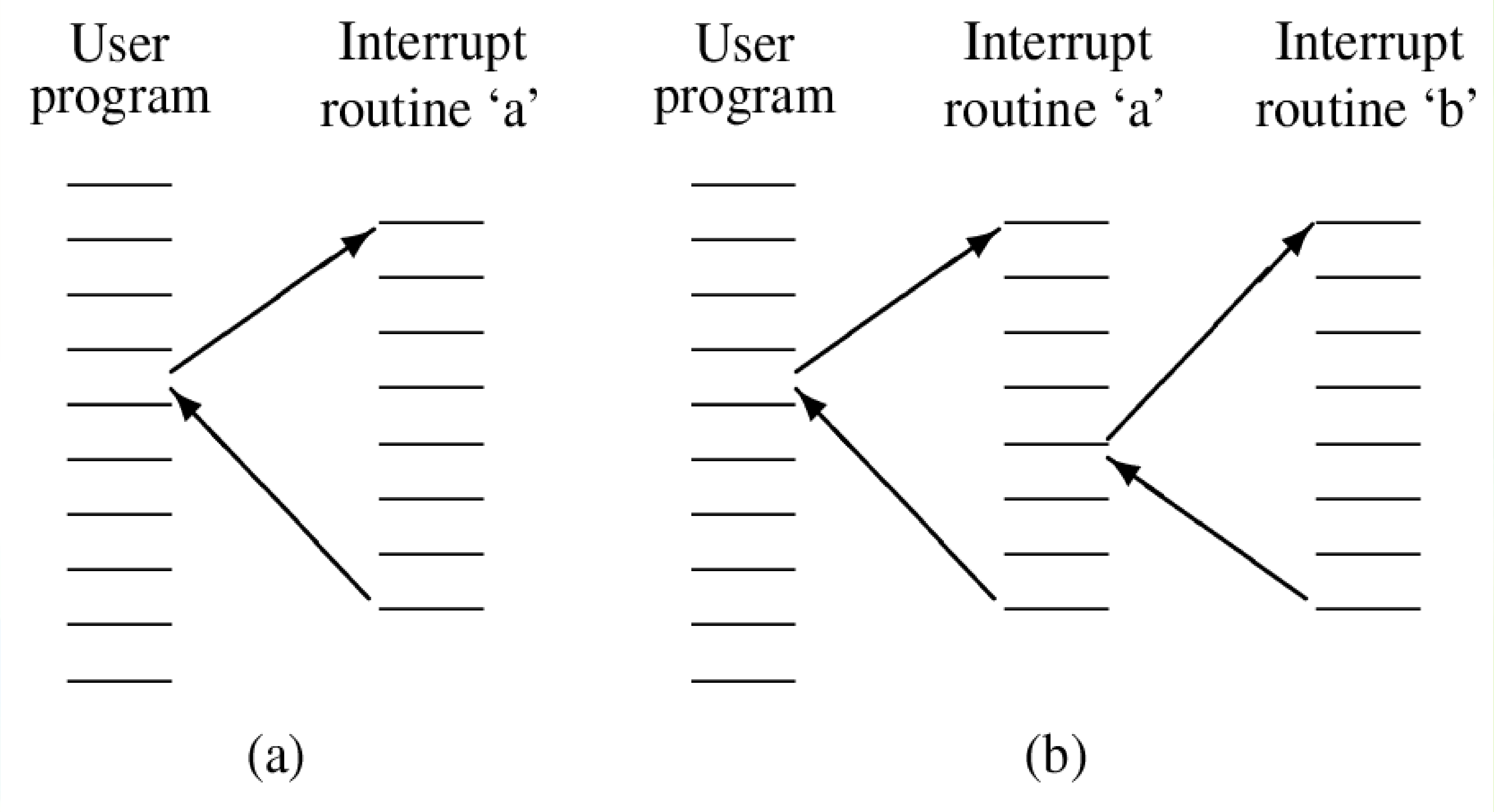
System Calls

Type	Examples
Resource	Allocate/deallocate resource, check resource availability
Program	Set/await timer interrupt, execute/terminate program
File	Open/close a file, read/write a file
Information	Get time and date, get OS information, get resource information
Communication	Send/receive message, setup/terminate connection

Interrupt processing and scheduling



Simple and nested interrupt processing



Queries ...?