Bài 1: OPERATION SYSTEM

1. LAYER ARCHITECTURE.

Embedded system

+ Software:

+ Hardware:

OS manage hardware

\* chia hệ điều hành thành 2 loại:

- Phân loại theo độ chính xác, REALTIME

+ embedded OS = RTOS : FreeRTOS, uOS

- specific

- real time

- luôn luôn chính xác

+ GPOS (general purpose ) : Windows, Linux

- có thể bị crash

- Not real time

- đa nhiệm

- hệ điều hành là một chương trình hay một hệ chương trình

- HDH hoạt động giữa ứng dụng người dùng và phần cứng của máy tính

- HDH quản lí phần cứng - xài chung cho mọi apps

- HDH giúp người dùng, nhà phát triển ứng dụng dễ dàng hơn

A diagram of a service

Description automatically generated

|  |  |
| --- | --- |
| Have OS | Non OS |
| * Dễ quản lí chức năng(task) * Phân chia công việc dễ * Bảo trì và nâng cấp * Có thể cài đặt ứu tiên cho ttask dễ dàng * Quản lý tốc độ đáp dứng được đễ dàng hơn * Phù hợp dự án cử lí nhiều công việc nhưng vẫn phải đáp ứng mặt thời gian | * Đễ thiệt lập và lập trình * - phù hợp dự án đơn giả, xử lí ít công việc |

* **Hệ thống song song**
* The system has many cores(CPUs)
* The system can run multiple proccesses (thăng tốc độ xử lí)  
  A screenshot of a computer

  Description automatically generated
* **Hệ thống phân tán**
* là một hệ gồm nhiều hệ thống con đặt ở xa nhau. Mỗi hệ thống con thực thi những chức năng riêng biệt.
* các hệ thống con liên lạc với nhau qua mạng lưới IOT: RS485, RF, Wifi, Bluetooth,Zigbee,…
* tính chất:
  + HT nhỏ không ảnh hưởng lẫn nhau.
  + Tăng khă năng, tốc độ xử lí.
  + Thu thập được nhiều loại tài nguyên.

A diagram of a computer network

Description automatically generated

* **Hệ thống xử lý thời gian thực**
* Là thệ thống đắp ứng được đúng khoảng thời gian đặt ra ban đầu
* Việc đáp ứng nhanh hơn/ chậm hơn sẽ dây lỗi hệ thống
* VD: Lò phản wunsg hạt nhân, HT túi khí xe hơi,…
* Chia làm 2 loại:

Hard Rt: túi khí xe hơi,

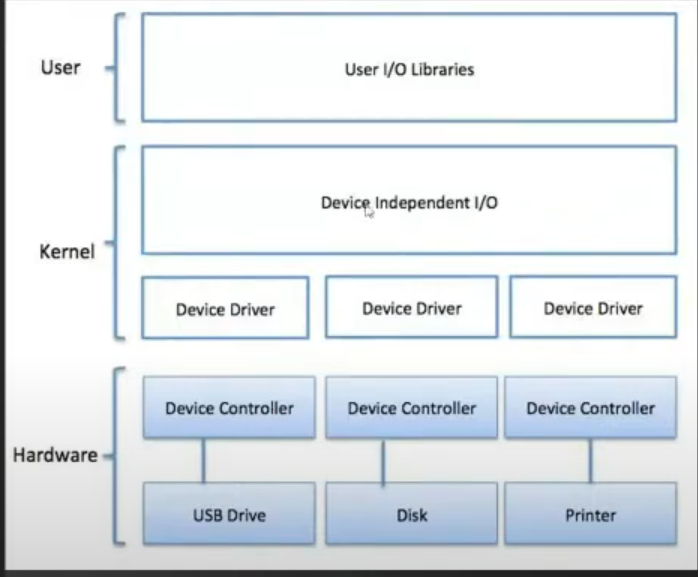
Soft RT: Live Stream videos.

**OS FUNCTIONS**

1. Memory management

* Primary memory: contains the running proccesses and ready proccesses

1. Device(I/O) management



* Device Independent I/O software: uniform interfacing for device drivers. This makes new driver easier to plug in the system.
* Buffer: 1 bọ nhớ giao tiếp với ngoại vi
  + Synchronize for different speed
  + Easy for store date while R/W with I/O
* Device controllers: works like an interface between a device and a device drivẻ
  + I/O units (phím, chuột, …) typicallt consist of a mechanicalcomponent and an

1. Proccess management
2. Security
3. Error dêtction
4. Coordination between S/w and User
5. Job accouting

* Command interpreter
* Help to communicate between User and OS
* These commands will invoke the system call to execute
* A screenshot of a computer

  Description automatically generated

**Bài 2 : OBIECTIVES**

* **Distinguish 3 definitions: multiprogramming, multitasking, multiprocessing**
* Multiprogramming: xử lí công việc từ trên xuongs dưới

A screenshot of a phone

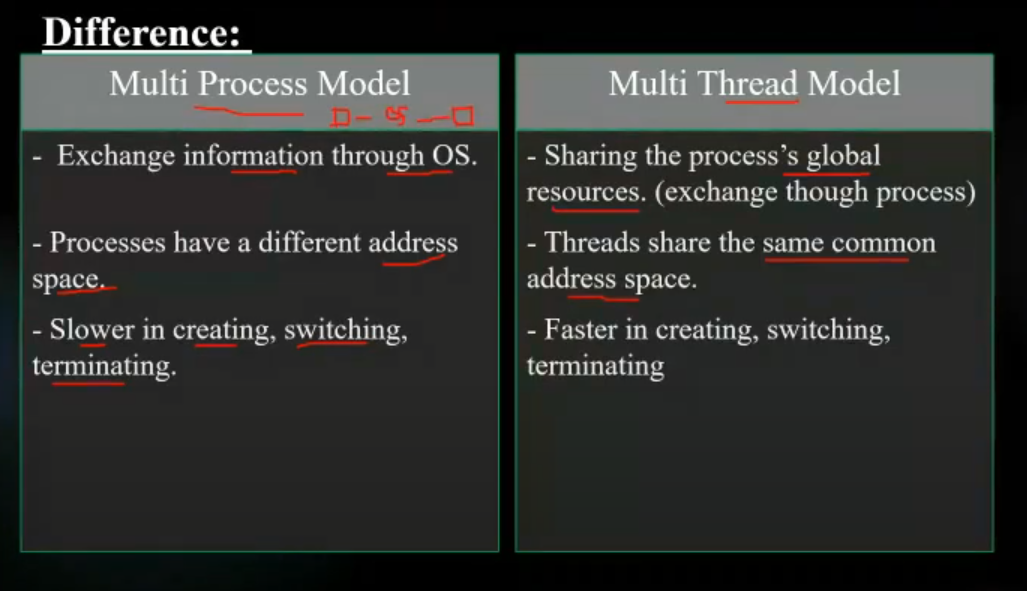
Description automatically generated

* **Multitasking: xử lí công việc xen kẽ theo miền thời gian (mỗi người chạy 1 tí).**
* **Multiprocessing: system has two or more cores, so that it can excute many programs in parallel**

A screenshot of a phone

Description automatically generated

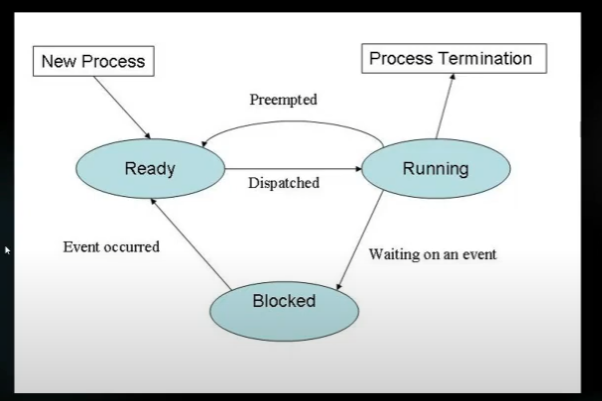
* **There are two motivations for moderrn Ó generally supports multitasking and multiprocessing**
* **Processs and Multi process model.**
* 
* **Thread:**
* **Is a sub-process or tiny processs which runs inside a process**
* **Threads in the same process share the common address space, which means threads can share the process’s global resources.**
* **Thread A can access Thread B’s stack**
* **Multithread: is the term used to run multiple threafs inside a single prrocess in order to excute multiple tasks at the same time**



A black screen with white text

Description automatically generated

* **Process life cycle.**



* **Quản lí tiến trình.**
* **Khi 1 tiến trình ra đời thì sẽ có 1 tài nguyên dành riêng cho nó**
* **Tại 1 thời điểm 1Cpu chỉ có 1 tiến trình được chạy.**
* **Vòng đời của tiến trình.**
* Trạng thái của 1 tiến trình tại 1 thời điểm được xác định bởi trạng thái ở thời gian đó.
* Tiến trình sẽ chuyển đổi trạng thái nếu:
  + đợi tài nguyên (Running -> spon),
  + đợi ngoại vi, hết thời gian thực thi,
  + có tiến trình độ ưu tiên cao hơn chen vào.

**Có 5 trạng thái của tiến trình**

* tạo mới
* sẵn sàng: tiến trình được cấp tài nguyên và đợi CPU chạy
* chạy: được Cpu cấp tài nguyên và thực thi
* Chặn/Khóa: tiến trình bận hay đợi I/O, thì chuyển sang trạng thái tậm dừng.
* Dừng: khi tiến trình đã chạy xong

Khởi tạo Process -> đưa vào ready list (khi được cấp phát tài nguyên) -> chuyển sang trại thái Running khi được đưa code vào CPU -> nếu bận đời I/O hay yêu cầu thêm tài nguyên mới mà chưa được cấp phát thì chuyển sang trạng thái Blocked -> khi tiến trình được cấp phát tài nguyên thì sẽ chuyển tại trạng thái ready -> tiến trình chạy xong thì dừng hoặc là hệ thống yêu cầu dừng thì sẽ chuyển sang trạng thái Termination -> nếu bị chiếm quyền bởi tiến trình khác thì sẽ quay lại ready thì cái chương trình chiếm quyền thì sẽ sang trạng thái running.

-Ready: là trạng thái đã có đủ tài nguyên

- Blocked là trạng thái chương trình đa chờ đợi tài nguyên được HDH cấp phát.

A diagram of a process

Description automatically generated

A screenshot of a computer

Description automatically generated

* **Process control bolock (PCB)**

A diagram of a computer hardware

Description automatically generated

1. **Process identifier: help to distinguish this with the others**
2. **Process state:**
   1. **Status: determine the curent state of the process**
   2. **Specify the ready list / or waiting list in which the process belongs.**

* **Nếu A đang chiếm dụng tài nguyên 1 nhưng B cũng muốn tài nguyên 1 thì A phải thực hiện xong hoặc nhả 1 ra thì B mới được cấp cho 1, rtong khi A đang chiếm 1 thì B vào trạng thái blocked (waiting list)**

1. **Process contexxt: describe the resources that process manages during operating, inncluding:**
   1. **CPU state: scheduling strtegy, and information of IP (PC), CPU registers**
   2. **Processeor: determine which CPU core number that process is using (multiprocessing system)**
   3. **Main store; contain, page table, segment table, and list of memory block allocated for the process**
   4. **Resource: the list of resources that the processiss using**
   5. **Created resource: a list of resources created by the process**
2. **Communication information: reflect the information about the relationship of the process with the others**
   1. **Parent: the process that created this process**
   2. **Progeny: the processes created by this process**
   3. **Priority: each process has a priority level, the system uses this for deciding which the process wil be granted CPU**
3. **Statistical information: the summary information, including: CPU usage time, wating time, time limits,…**

**This information is be useful for accessing the process health predicting the future scenarios**

**Summary:**

**Idd tiến trình**

**Trạng thái tiến trình**

**Tài nguyên tiến trình quản lý**

**Mối quan hệ của tiến trình**

**Thông tin liên quan đến tiến trình mà hệ thống quản lý**



**Bài 3; Vai trò, tài nguyên, scheduler, dispatcher, contextswwitch của HDH với tiến trình**

* **Vai trò**

**The operating system provides the following primary operations on a process:**

* **Create process**
  + **Define a new identifier**
  + **Put the process on the systems management list (ready/waiting list)**
  + **Define Priority for Process**
  + **Creat PCB for Process**
  + **Allocate initial resource to the process**
* **Terminate the process (destroy )**
  + **Diễn ra bởi 2 lí do : chạy xong chương trình hoặc người dùng hay tiến trình khác yêu cầu dừng.**
  + **Khi tiến trình dừng**
    - **Thu hồi tài nguyên cấp phát cho tiến trình**
    - **Xóa hết tên trong tên của các danh sách mà HDH quản lí**
    - **Xóa PCB**
  + **Hầu hết OS k cho tiến trình con tồn tại nếu tiến trình cha bị xóa**
* **Suspend the process(block/sleep)**
* **Reactive the process(resume)**
* **Change the process Priority**

**One process (called a parent ) can create a new process (called a child) by sending the system call for the OS**

**When the parent create a new child, the parent can act in one of two ways:**

* **The parent and the child continue to process concurrently**
* **The parent wait until a certain child process or all child processes dinish processing**

**Each resource is represented through a data structure, which esentially contains the following information:**

* **Rid : Resource identifier**
* **Resource state: what part of the resource already allocated to the process, what part is still usable?**
* **Queue a resource: a list of processes wating to be allocated the corresponding resource.**
* **Allocator (Pointer): point to the sourrce code that takes care of allocating the resource**

**The goal of allocating resource:**

* **Đảm bảo chỉ 1 process 1 resouce 1 time**
* **Cấp phát taafi nguyên cho tiến trình trong 1 khoảng thời gina delay hợp lệ**
* **Tối ưu hiệu năng**

**The scheduler (Bộ điều phối): xác định trình tự process được thực thi theo1 giải thuật nào đó.**

**The dispatcher(Bộ phân phối): will be responsible forr context switchng and handing the Cpu to the process selected by the scheduler for next processing**

A screenshot of a computer screen

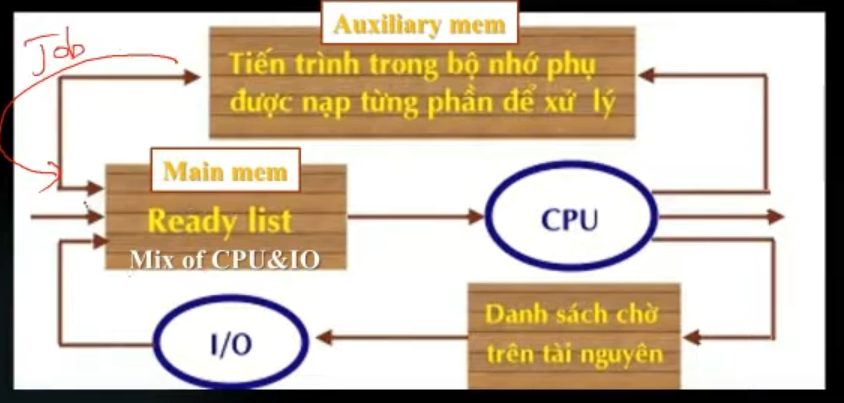
Description automatically generated

A black board with red writing

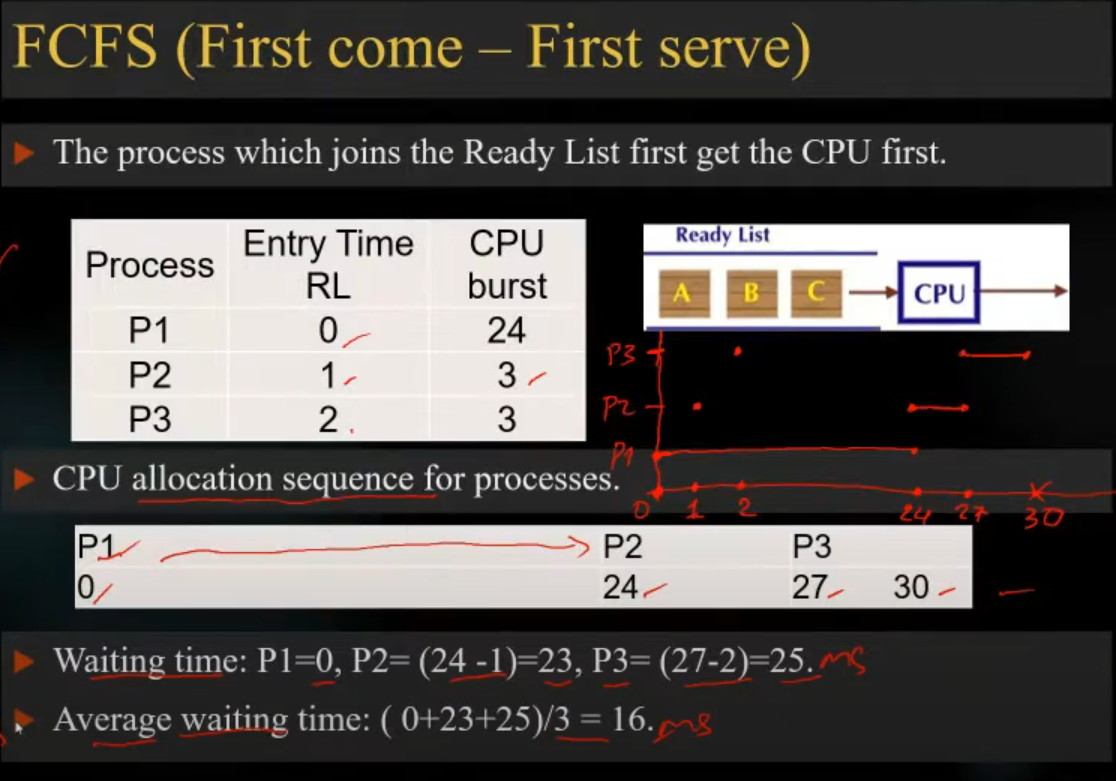
Description automatically generated

* **Scheduling Goal**
  + **Công bằng: không ai phải chờ vô hạn**
  + **Hiệu quả**
  + **Thời gian đáp ứng : giảm thời gian đáp ứng tăng tương tác ng dùng**
  + **Thời gian lưu lại trong hệ thống**
  + **Thông lượng tối đa: tức là xét trong 1 khoảng thời gian có bao nhiêu proces**
* **Process characteritics** 
  + **Tính xuất nhập và tính cử lí**
  + **Tiến trình tương tác hay xử lí theo lô**
  + **Độ ưu tiên**
  + **Lượng thời gian đã sử dụng**
  + **Lượng thời gian còn lại để hoàn thành.**
* **Độc quyền và khong độc quyền:**
  + **Không độc quyền**
    - **Nếu có 1 process đang chạy, có 1 process độ ưu tiên cao hơn thì phải cho processs có độ ưu tiên cao hơn chạy**
    - **Cách này nhằm cho các hệ thống thời gian thực**
  + **Độc quyền**
    - **Ai đang chạy thì sẽ giữ CPU mãi, chỉ khi chuyển sang blocked thì mới để cho process khác chạy, phù hợp cho hệ thống Batch system**
* **Sheduling organization**
  + **Ready list : 1 core là 1 ready list**
  + **Waiting list: 1 resource là 1 waiting list**
  + A diagram of a computer system

    Description automatically generated
* **2 cấp độ scheduling**
  + **Điều phối tác vụ và điều phối tiến trình**
    - **Điều phối tác vụ :để tăng hiệu năng điều phối -> tăng khả năng lựa chọn**



* **Chiến lượt điều phối.**
  + **FCFS(First come – first serve ) (FIFO)**



* + **Chiến lượt điều phối : ROUND ROBIN: Tất cả process đều đc chạy khoảng thời gian Q(s)**

A screenshot of a computer

Description automatically generated

* + **Chiến lượt điều phối: PRIORITY SCHEDULING**

A screenshot of a computer

Description automatically generated

* + **độc quyền.**

A screenshot of a computer

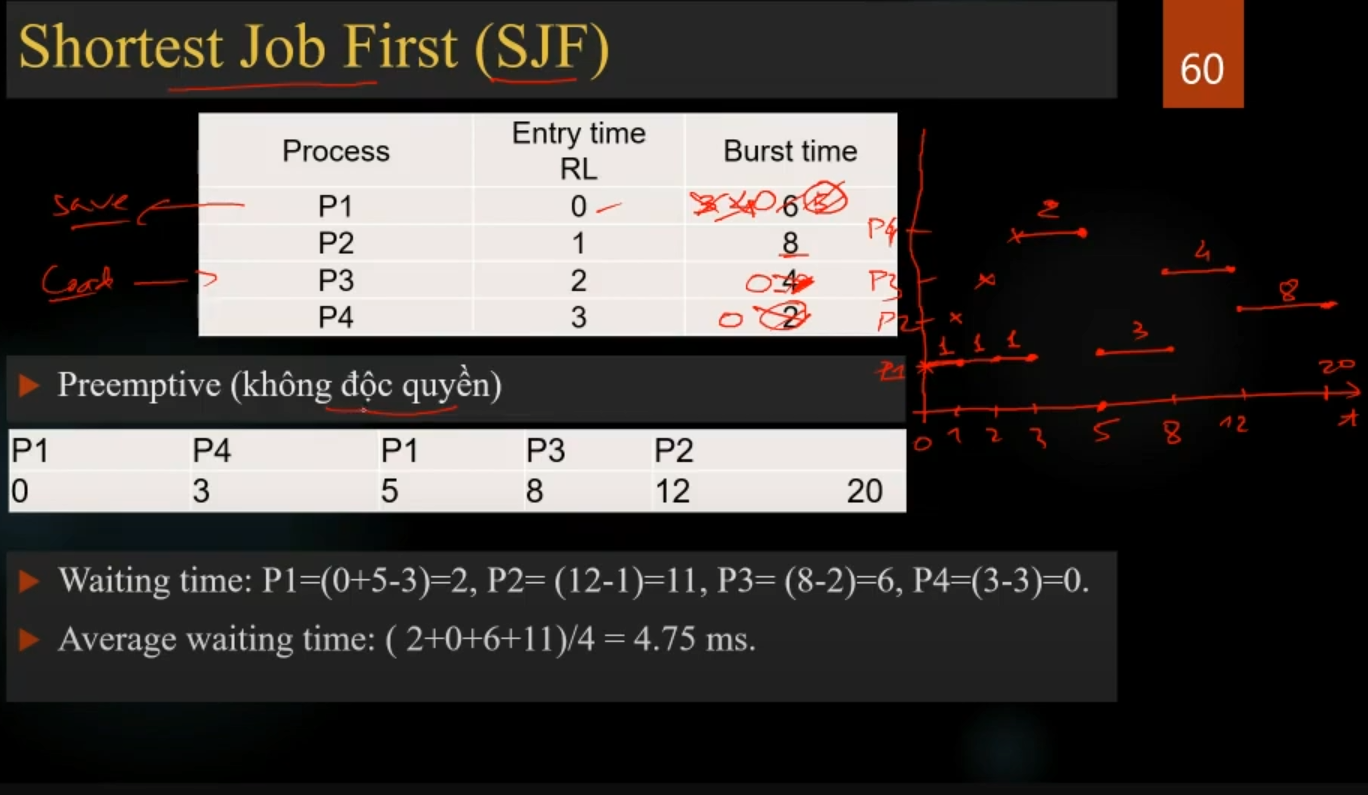
Description automatically generated

* + **Không độc quyền**

A screenshot of a computer

Description automatically generated

* + **Shortest Job First.**



* **Process communication.**
* Mục đích để giao tiếp giữa các tiến trình.

1. Tại sao phải giao tiếp giữa các tiến trình.

* Chia sẻ thong tin cho nhau.
* Phối hợp với nhau để thực hiện 1 công việc lớn.

1. Các vấn đề có thể xảy ra trong lúc giao tiếp.

* Các tiếp trình giao tiếp gián tiếp với nhau thông qua hệ điều hành
* Những vấn đề cần quan tâm:
  + Tường minh: người gửi biết đại chỉ người nhận
  + Tiềm ẩn: Người gửi k biết địa chỉ người nhận, nhưng vẫn gữi qua 1 cơ chế có trước.
  + Đồng bộ/ blocked: người gửi và người nhận có xung CLK, hoặc có bit ACK hoặc SYN để đồng bộ hóa.
  + Không đồng bộ hóa: K có, bên gửi nhưng k quan tâm bên nhận.
  + Hệ thống tập trung: tiến trình gửi và nhận chung 1 hệ thống
  + Hệ thống phân tán: người gửi và nhận nằm khác hệ thống.
* **Các cơ chế giao tiếp**

1. Signaling.

* Tín hiệu dùng để báo động cho tiến trình khi có 1 sự kiện nào đó xảy ra
* Là giao tiếp không đồng bộ(k biết khi nào tín hiệu đến)
* Chỉ gửi Tín hiệu báo động (Notification), kh gửi được data.
* Chỉ hoạt động trên hệ thống tập trung
* Là 1 Ngắt(interrupt)
* Ví dụ tiếng kêu khi cắm chuột hoặc bàn phím, hoặc thông báo đầy dung lượng khi download

1. **Pipe.**

* Ống 1 chiều (unidirectional) dùng để giao tiếp giữa HAI tiếng trình với nhau.
* Mỗi ống chỉ có 1 chiều dữ liệu di chuyển.
* Có 2 trường hợp treo hệ thống (DEADLOCK).
  + 2 pipe cùng đầy, theo cơ chế thì bên gửi sẽ tiến vào trạng thái waiting dẫn đến việc cả 2 Proccess cùng tiến vào trạng thái Waiting dẫn đến trình trạng DEADLOCK.
  + 2 pipe cùng empty, nhưng 2 Process cùng muốn đọc data 🡪 2 Process waiting 🡪 DEADLOCK.(có thê set thời gian timeout)

1. **Chia sẻ vùng nhớ.**

* Cơ chế này sử dụng 1 vùng nhớ chung mà nhiều Process có thể truy cập.
* Được xem là cơ chế nhanh nhất.
* Khi có nhiều tiến trình cùng đọc và ghi cùng lúc thì sẽ có vấn đề phát sinh.
  + Overwriten (Dữ liệu bị ghi đè).
  + Dữ liệu đọc về không phải dữ liệu mới nhất.

1. **Message (Trạo đổi thông điệp)**

* **Truyền tường minh**
  + **Send (P2,Message)**
  + **Receive(P1,Mesage)**
* Truyền gián tiếp: truyền tin vào trong MAILBOX và B tới MAILBOX nhận
* Có thể kiểm soát được kiểu đồng bộ và không đồng bộ
* Có thể chạy trên cùng PC hoặc khác PC.

1. SOCKET.

* Là một cơ chế phổ biến.
* Cơ chế giao tiếp với nhau thông qua hệ thống mạng.
* Cơ chế Client – Server.
* IP address – Port :
* Chỉ có những chương trình truyền nhận dữ liệu trên hệ thống mạng mới có địa chỉ Port.
* Gói tin: Source ID + Port + Destiantion ID + Data.
* Cơ chế SOCKET có 2 loại:
  + UDP: nhanh vì k cần xác nhận bên phía nhận có nhận được thông tin hay không.
  + TCP: giao tiếp truyền nhận dữ liệu có điều khiển, trước tiên phải tạo ra 1 mối kết nối (connect) 🡪 Truyền nhận data (có thể kiểm tra data) 🡪 disconnect.
* Có thể xây dựng kiểu đồng bộ hoặc không đồng bộ.
* **SUMMARY**
* 
* **Vấn đề về đồng bộ hóa**
  + **Truy xuất độc quyền.(Có 2 loại tài nguyên)**
    - Tài nguyên có thể chia sẻ: cho phép nhiều tiến trình có thể truy cập và sử dụng.
    - Tài nguyên không thể chia sẻ: tại mỗi thời điểm chỉ cho1 tiến trình truy xuất.
    - 🡺với tài nguyên không thể chia sẽ cần phải có cơ chế quản lý, đồng bộ hóa.
  + Yêu cầu phối hợp: cơ chế về đồng bộ hóa theo tiến trình.
  + Critical section (miền găng) tại 1 thời điểm chỉ có 1 miền găng được chạy
* Cách giải quyết miền găng
  + Không quá 2 tiến trình có thể tham gia vào miền găng cùng lúc.
  + Dù cho hệ thống có nhiều CPU hay tốc độ nhanh thì vẫn kiểm soát đc .
  + Nếu như A không dùng miền găng thì A k được phép ngăng các tiến trình khác.
  + Không có tiến trinh nào đợi vô hạn.
* Summay:
  + Cơ chế giao tieeps tiến trình để chia sẽ thông tin, chia công việc lớn ra làm
  + Xem xét về các vấn đề cần quan tâm.
  + Signal, pipe, shared memory, Message, Socket.
  + Vấn đề về đồng bộ hóa.
    - Truy xuất độc quyền – citical section
    - Yêu cầu phối hợp.
  + Các vấn đề không có cơ chế đồng bộ phù hợp.
  + Miền găng: là 1 đoạn chương trình hoặc 1 loại tài nguyên, tại 1 thời điểm chỉ có 1 tiến trình truy cập.
    - Có 4 tiêu chí: chỉ 1 miền găng, kiểm soát tốt miền găng không phụ thuộc vào tốc độ là core CPU, k tiến trình nào được chặn miền găng nếu k đang truy cập, không có tiến trình nào chờ vô hạn

**BÀI 5 : ĐỒNG BỘ HÓA.**

* **Busy waiting.**
  + **Cờ hiệu (lock): đơn giản nhưng phải thực thi công việc vô ích**
    - **Dùng 1 biến lock để làm tín hiệu cho chương trình được vào miền găng.**