

# THIẾT BỊ NGOẠI VI VÀ KỸ THUẬT GHÉP NỐI

Bộ môn Kỹ thuật Máy tính  
Viện CNTT&TT- ĐH BKHN



# Mục tiêu môn học

- Giúp cho sinh viên nắm được:
  - Cấu trúc và hoạt động của các hệ vi xử lý nói chung
  - Kỹ thuật nối ghép thiết bị ngoại vi với máy tính, hệ vi xử lý, hệ nhúng
  - Cấu trúc và hoạt động của các thiết bị ngoại vi cơ bản
  - Cơ bản về lập trình giao tiếp sử dụng các giao thức ghép nối thông dụng.



# Nội dung môn học

Chương 1. Mở đầu

Chương 2. Kỹ thuật ghép nối

Chương 3. Thiết bị ngoại vi

Chương 4. Lập trình ghép nối

Chuyên đề:

- Tìm hiểu các giao thức ghép nối UART, SPI, I2C, USB
- Lập trình vi điều khiển (8051,ARM)



# Chương 1-Mở đầu

- 1.1. Kiến trúc hệ vi xử lý/máy tính/hệ nhúng
- 1.2. Hoạt động của hệ thống máy tính





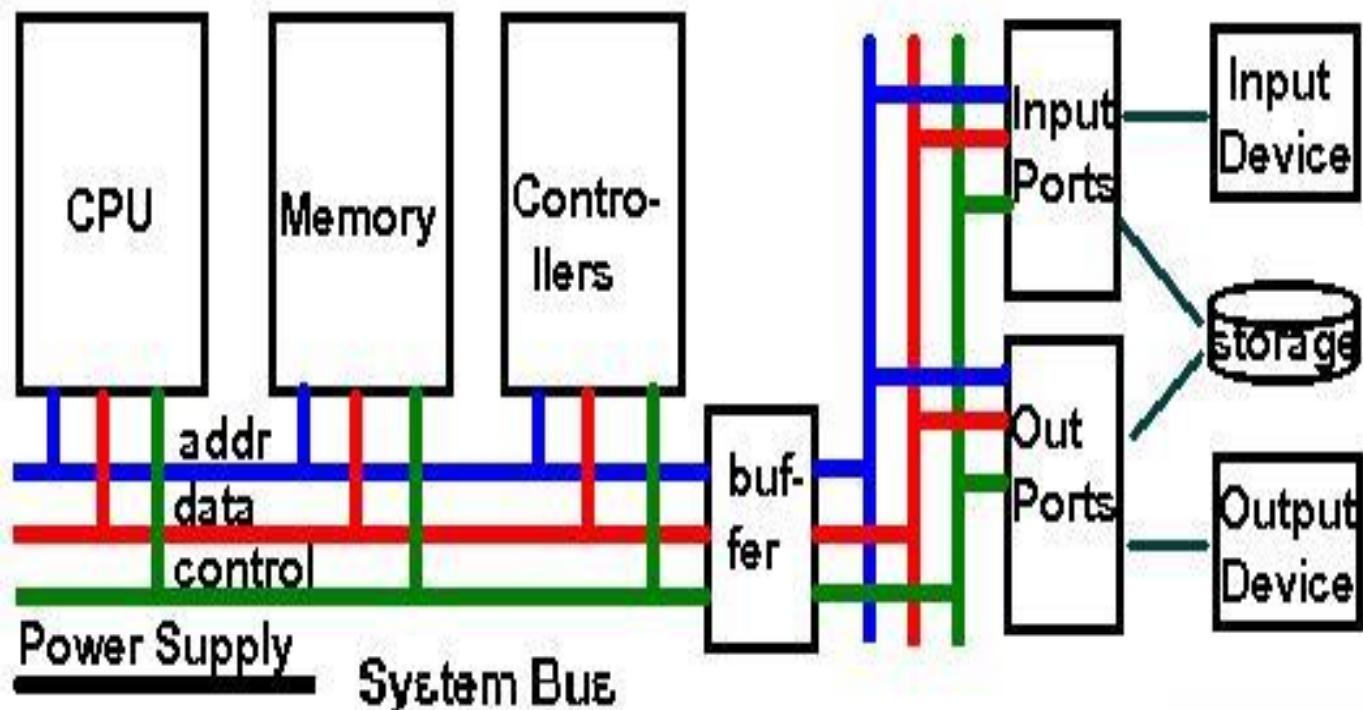
# 1. Kiến trúc Hệ vi xử lý/ máy tính/ hệ nhúng

• Gồm 4 phần:

✓ Hệ trung tâm  
(CS)

✓ Giao tiếp  
(Interface)

✓ Ngoại vi  
(Peripheral, Wide  
world)



Hình 1. Sơ đồ khái niệm hệ VXL/MT kinh điển



## 1.1 Kiến trúc Hệ vi xử lý/ máy tính/ hệ nhúng

- **Hệ vi xử lý:** mang nghĩa tổng quát, là một hệ thống bao gồm các thành phần cơ bản như trên, có khả năng tương tác và xử lý công việc.
- **Hệ nhúng:** là một hệ vi xử lý được thiết kế có chức năng chuyên dụng.  
Vd:Các hệ thống trong Lò vi sóng, Máy giặt,...
- **Máy tính cá nhân:** một trường hợp cụ thể của hệ vi xử lý \*

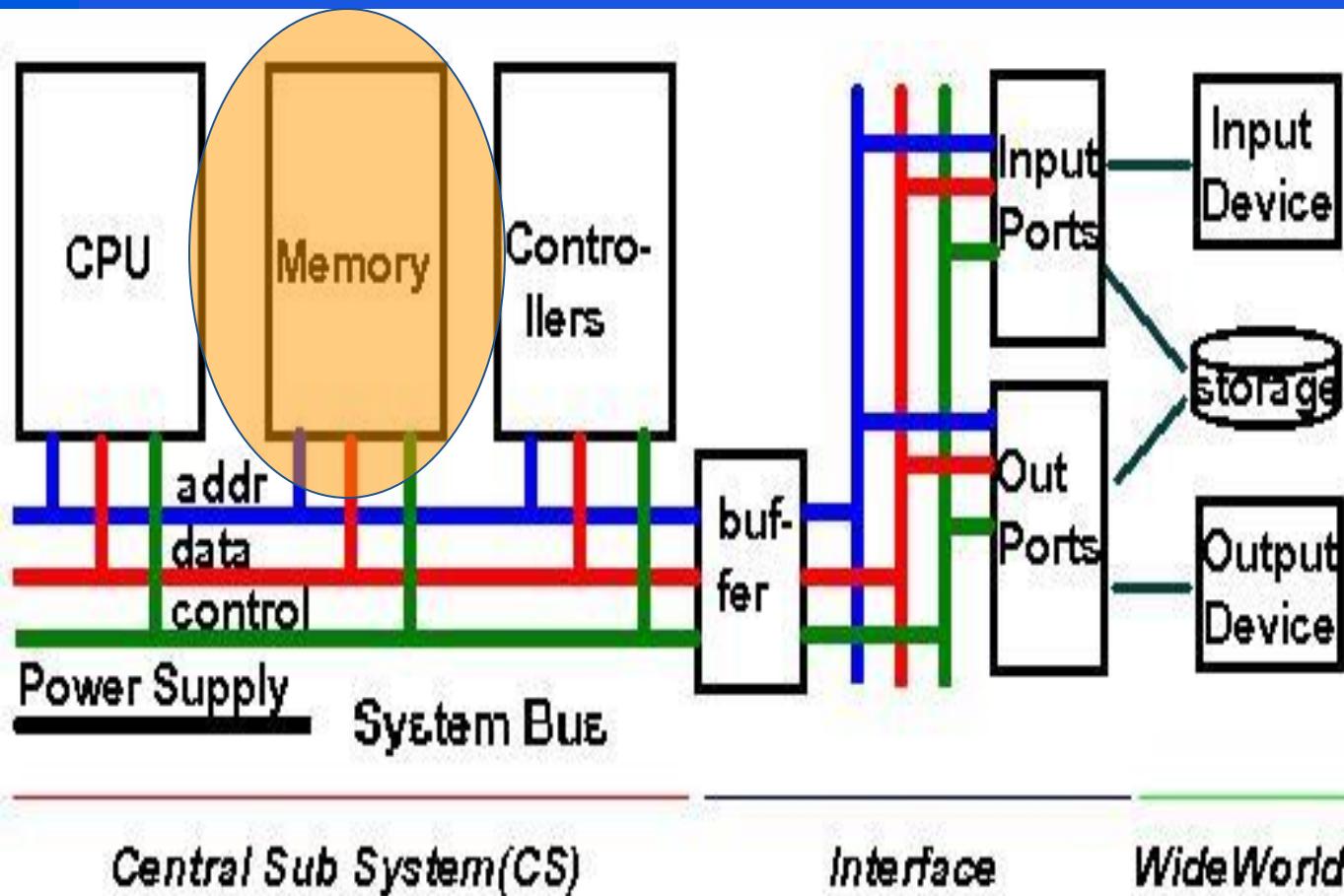


# Hệ trung tâm (CS)

- **Hệ trung tâm (Central Sub System) gồm:**
  1. Bộ nhớ chính (Main Memory)!
  2. Khối xử lý trung tâm!  
(CPU-center processing unit)
  3. Các điều khiển (Controllers)!



# Memory



Hình 1. Sơ đồ khái niệm hệ VXL/MT kinh điển



# Bộ nhớ (Memory)

- Gồm 2 loại chính: ROM và RAM
- ROM (Read Only Memory): Bộ nhớ chỉ đọc
  - Chỉ thực hiện được thao tác đọc từ bộ nhớ (!)\*
  - Thông tin vẫn được lưu lại khi không có nguồn cấp điện
  - Thường lưu trữ các chương trình và dữ liệu cố định của hệ thống
  - Hầu hết các hệ vi xử lý có dung lượng ROM nhỏ, lưu trữ các chương trình hoặc dữ liệu nhỏ, quan trọng và thường xuyên được sử dụng.

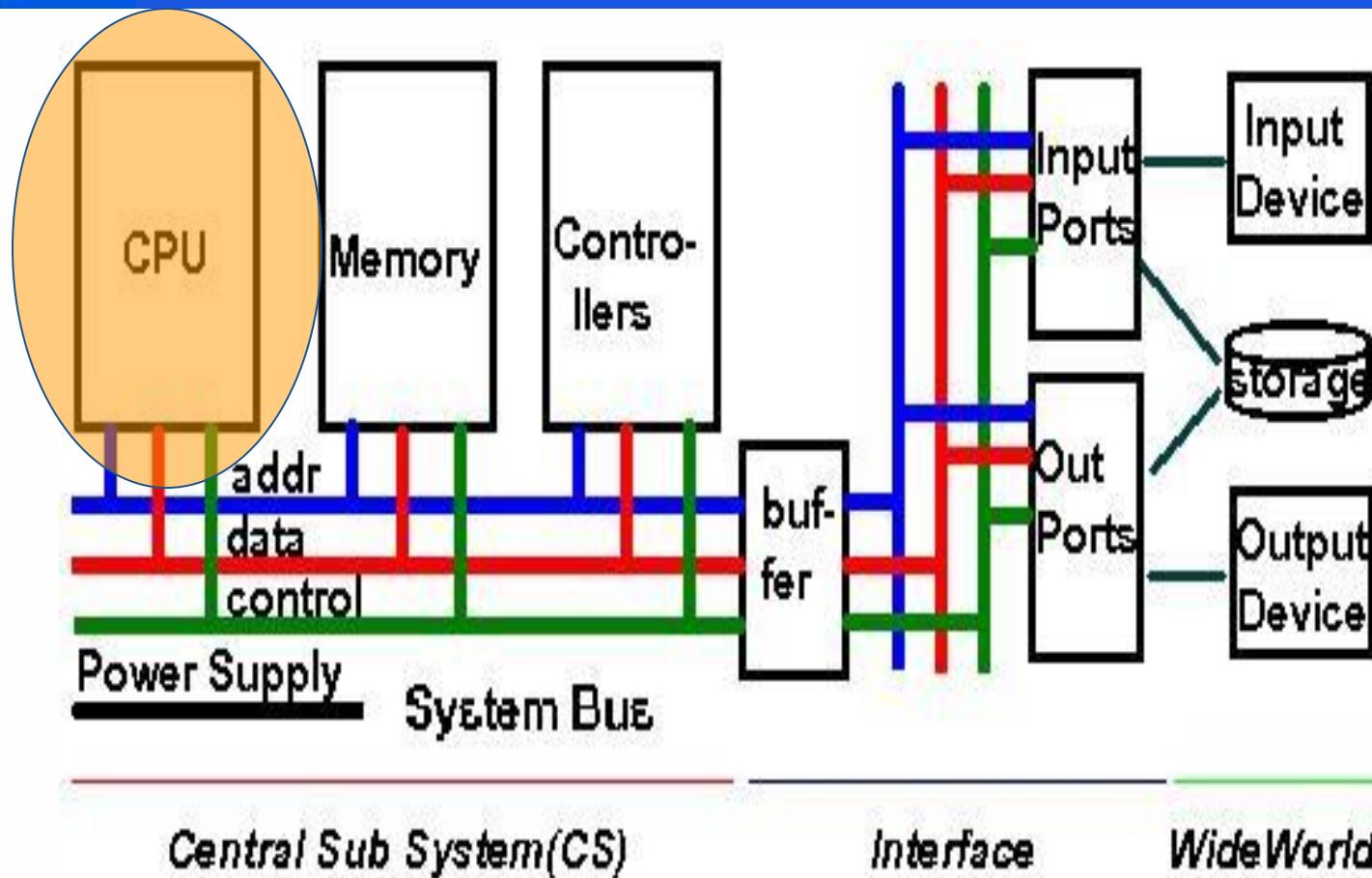


## Bộ nhớ (Memory)

- RAM (Random Access Memory)-Bộ nhớ truy cập ngẫu nhiên:
  - Có thể thực hiện đọc/ghi đối với bộ nhớ
  - khi không có nguồn cấp điện ?  
thông tin bị mất
  - Thường lưu trữ các thông tin tạm thời.
  - Trong các hệ vi xử lý, dung lượng của RAM thường lớn hơn ROM.



CPU



Hình 1. Sơ đồ khái niệm VXL/MT kinh điển



# Khối xử lý trung tâm CPU

## ▪ CPU:

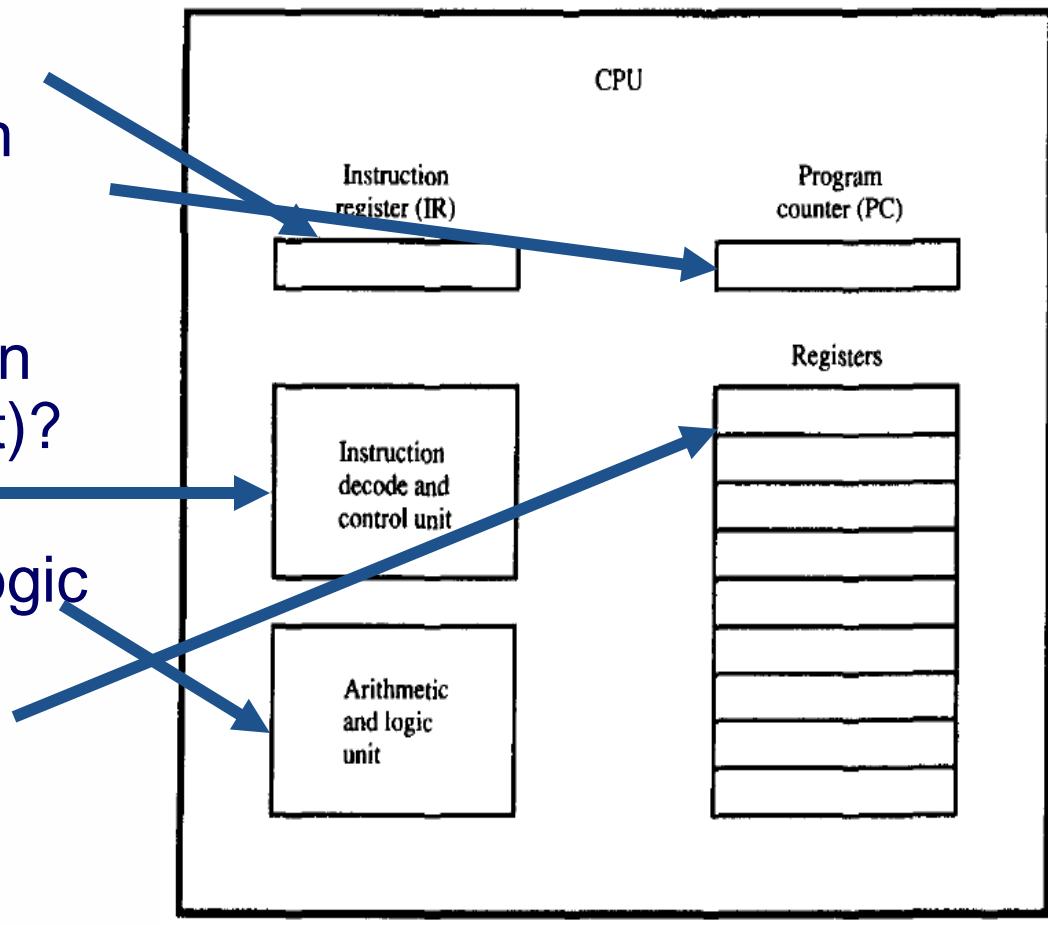
- Là khối xử lý trung tâm, đóng vai trò chủ đạo trong hệ vi xử lý.
- Thực hiện công việc được giao trong bộ nhớ chương trình ?:
  - ✓ *Đọc lệnh* : đọc mã lệnh được ghi dưới dạng các bit nhị phân từ bộ nhớ chương trình.
  - ✓ *Giải mã lệnh*: giải mã các lệnh thành dãy các xung điều khiển ứng với các thao tác trong lệnh.
  - ✓ *Điều khiển thực hiện lệnh*: phát các tín hiệu điều khiển (đã giải mã được) tới các khối khác để thực hiện yêu cầu của lệnh.



# Khối xử lý trung tâm CPU

## ▪ CPU:

- Thanh ghi lệnh (IR)?
- Bộ đếm chương trình (PC)?
- Khối giải mã và điều khiển lệnh (Instruction decode & control unit)?
- Khối logic và số học (ALU-Arithmetic & Logic Unit)?
- Tập các thanh ghi?
- Ngoài ra còn có các đường bus.





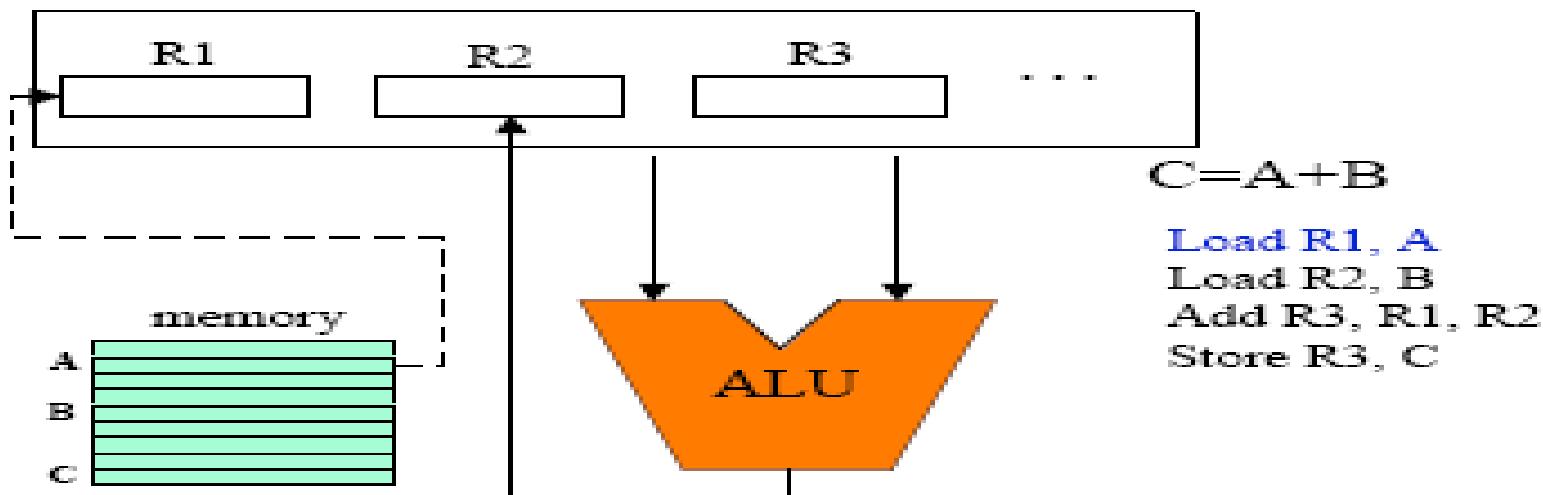
# Khối xử lý trung tâm(CPU)

- Thanh ghi lệnh (IR):
  - chứa mã nhị phân của lệnh đang được thực hiện.
- Bộ đếm chương trình (PC):
  - Chứa địa chỉ của lệnh tiếp theo sẽ được thực hiện
- Khối giải mã lệnh và điều khiển:
  - Xác định lệnh cần thực hiện và đưa ra các thao tác để thực hiện lệnh



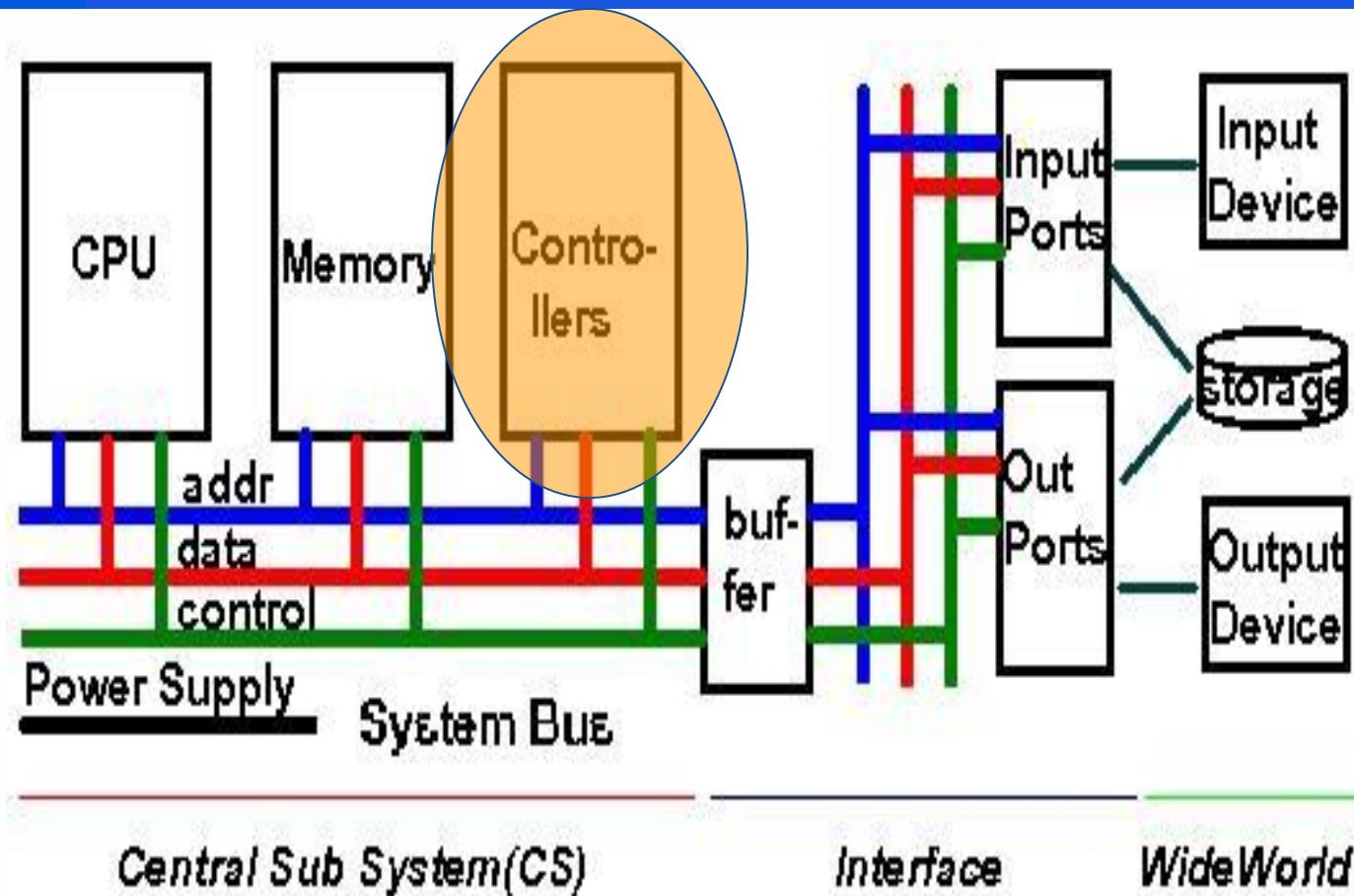
# Khối xử lý trung tâm CPU

- Tập các thanh ghi:
  - Lưu giữ tạm thời các dữ liệu khi thực hiện lệnh
  - Là nơi chứa dữ liệu đầu vào và ra của ALU
- Khối logic và số học (ALU):
  - Thực hiện các phép toán logic ( and, or, xor, not), số học (cộng, trừ, nhân, chia) \*





# Controller



Hình 1. Sơ đồ khái niệm VXL/MT kinh điển



# Các bộ điều khiển (Controllers)

- **Controllers:** Là các vi mạch có chức năng điều khiển nhằm nâng cao hiệu năng hoạt động của hệ thống, bao gồm: (vd?)
  - Bộ điều khiển truyền tin nối tiếp (COM) 8250
  - Bộ điều khiển ưu tiên ngắn: PIC – Priority Interrupt Controller, Intel 8259A
  - Bộ điều khiển truy nhập trực tiếp bộ nhớ DMA – Direct memory Access Controller, Intel 8237A.

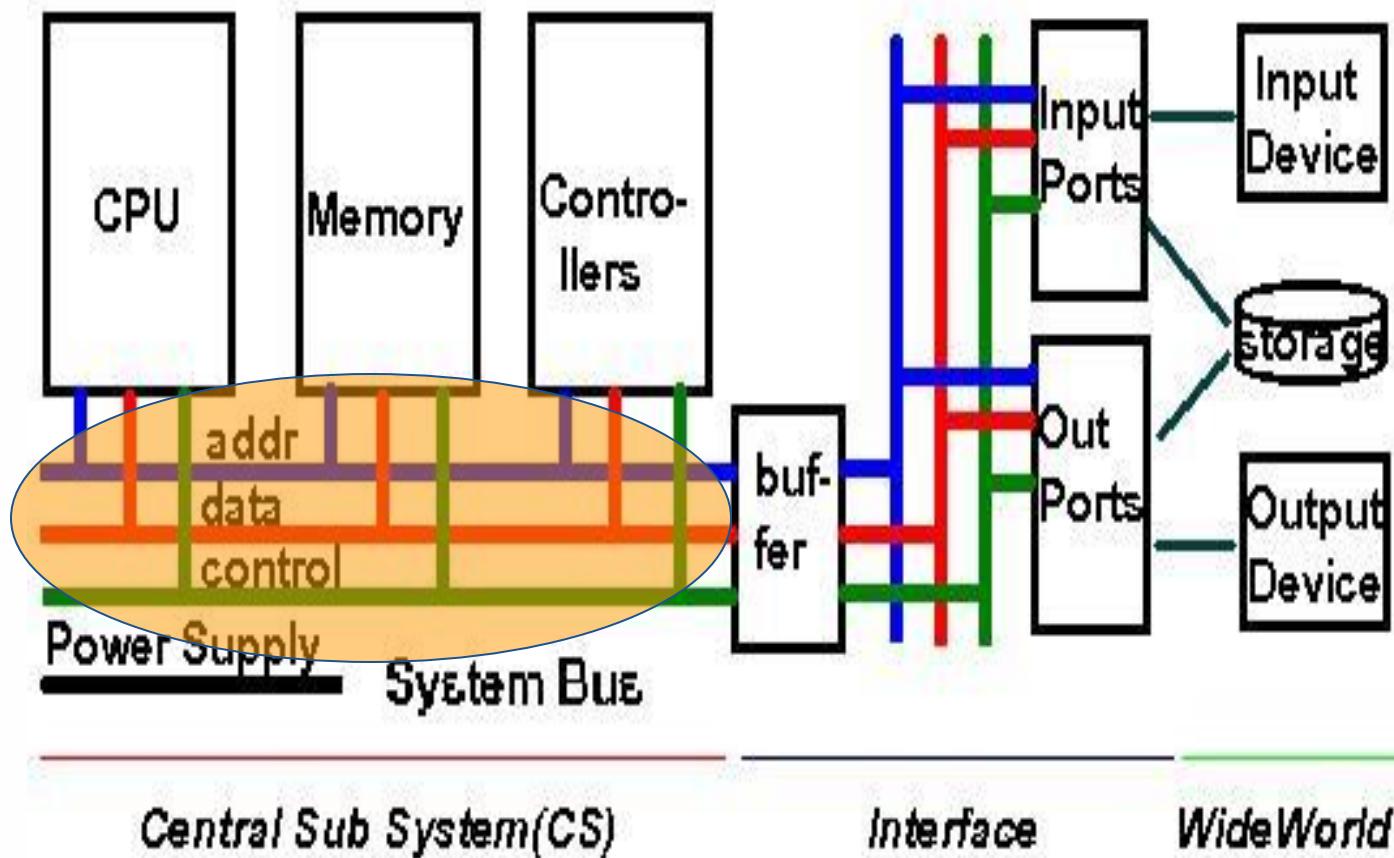


# Các bộ điều khiển (Controllers)

- Bộ định thời (Timer): mạch tạo khoảng thời gian PIT- Programmable Interval Timer, Intel 8254.
- Mạch quản lý bộ nhớ: MMU- Memory Management Unit, sau này thường được built on chip với CPU.
- Bus controller/Arbitor



Bus



Hình 1. Sơ đồ khái niệm VXL/MT kinh điển



- Khái niệm chung về bus:

- Một Bus: là tập hợp các đường kết nối để truyền thông tin giữa các thành phần của hệ vi xử lý. Thông tin trên các đường kết nối này nhằm phục vụ cho cùng một mục đích nào đó.
- Độ rộng bus: là số đường dây có thể truyền thông tin đồng thời của bus đó.

- Có 3 loại bus:

- Bus địa chỉ (address bus)
- Bus dữ liệu (data bus)
- Bus điều khiển (control bus)





## ▪ Bus địa chỉ:

- Thông tin trên bus là địa chỉ : của các ô nhớ, các cổng vào-ra
- Chiều?

Là bus 1 chiều: từ CPU, DMAC, PCI host Controller tới bộ nhớ, các cổng.

- Độ rộng bus địa chỉ: xác định dung lượng cực đại của bộ nhớ hệ thống. Nếu độ rộng là N (bit) ?:
  - ✓ Có thể truyền đồng thời N bit địa chỉ
  - ✓ Có khả năng đánh địa chỉ được  $2^N$  ( ngăn nhớ, cổng vào-ra)



# Bus dữ liệu

- Bus dữ liệu (data bus):

- Thông tin trên bus là dữ liệu:
  - ✓ Các lệnh
  - ✓ Dữ liệu
- Chiều?

Bus dữ liệu là 2 chiều, dữ liệu được truyền:

- ✓ CPU  $\Leftrightarrow$  Bộ nhớ
- ✓ CPU  $\Leftrightarrow$  Module vào-ra



## Bus dữ liệu (tiếp)

- Độ rộng bus dữ liệu: xác định số bit dữ liệu có thể được trao đổi đồng thời.
- Máy tính (bộ xử lý): 8 bit/16 bit/32 bit (?):
- ~ đường bus địa chỉ là 8 bit/16 bit/ 32 bit ~ tập các thanh ghi thông dụng là 8 bit/16 bit/32 bit.



# Bus điều khiển

- Bus điều khiển (control bus):
  - Thông tin trên bus là các tín hiệu điều khiển:
    - ✓ Tín hiệu điều khiển hoặc trả lời từ CPU tới các controller, mô đun nhớ, mô đun vào-ra.
    - ✓ Tín hiệu yêu cầu từ các controller, mô đun nhớ, mô đun vào-ra tới CPU.
  - Chiều?  
Bus 2 chiều





# Bus điều khiển

- Ví dụ:

- Các tín hiệu phát ra từ CPU để điều khiển đọc-ghi:
  - ✓ Memory Read (MEMR) : điều khiển đọc dữ liệu từ một ngăn nhớ có địa chỉ xác định **lên** bus dữ liệu.



# Bus điều khiển

- VD(tiếp):

- Các tín hiệu phát ra từ CPU để điều khiển đọc-ghi:

- ✓ Memory Write (MEMW) : điều khiển ghi dữ liệu có sẵn trên bus dữ liệu lên 1 ngăn nhớ có địa chỉ xác định
    - ✓ I/O Read (IOR) : điều khiển đọc dữ liệu từ một cổng vào-ra có địa chỉ xác định lên bus dữ liệu
    - ✓ I/O Write (IOW) : điều khiển ghi dữ liệu có sẵn trên bus dữ liệu lên một cổng vào-ra có địa chỉ xác định



# Bus điều khiển

- VD (tiếp):

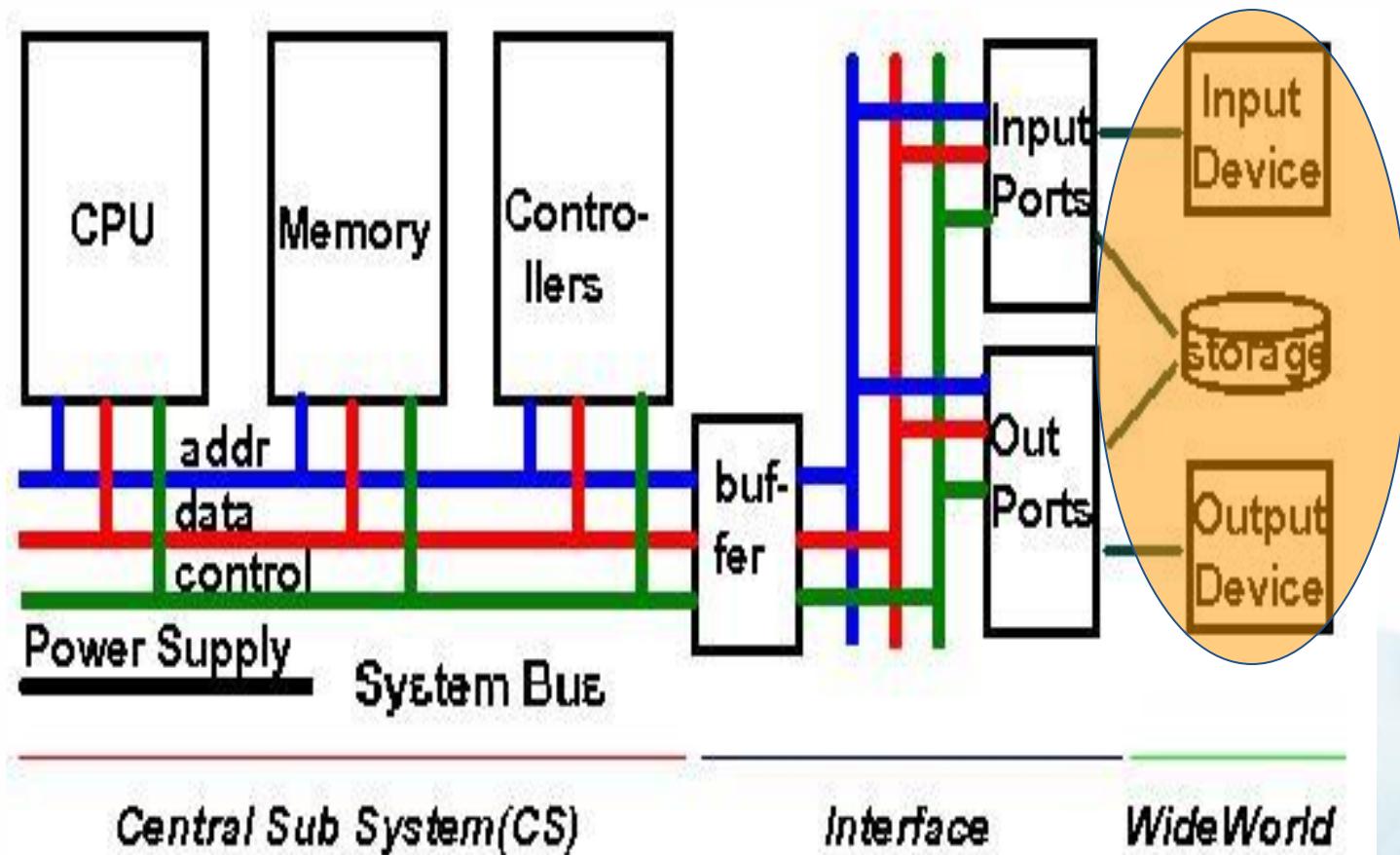
- Các tín hiệu yêu cầu gửi tới CPU:

- ✓ Interrupt Request (INTR) : tín hiệu từ bộ điều khiển vào-ra gửi yêu cầu ngắt tới CPU
    - ✓ Non Maskable Interrupt (NMI) : tín hiệu yêu cầu ngắt không che được gửi tới CPU \*.

Che được và không che được?



# Peripheral

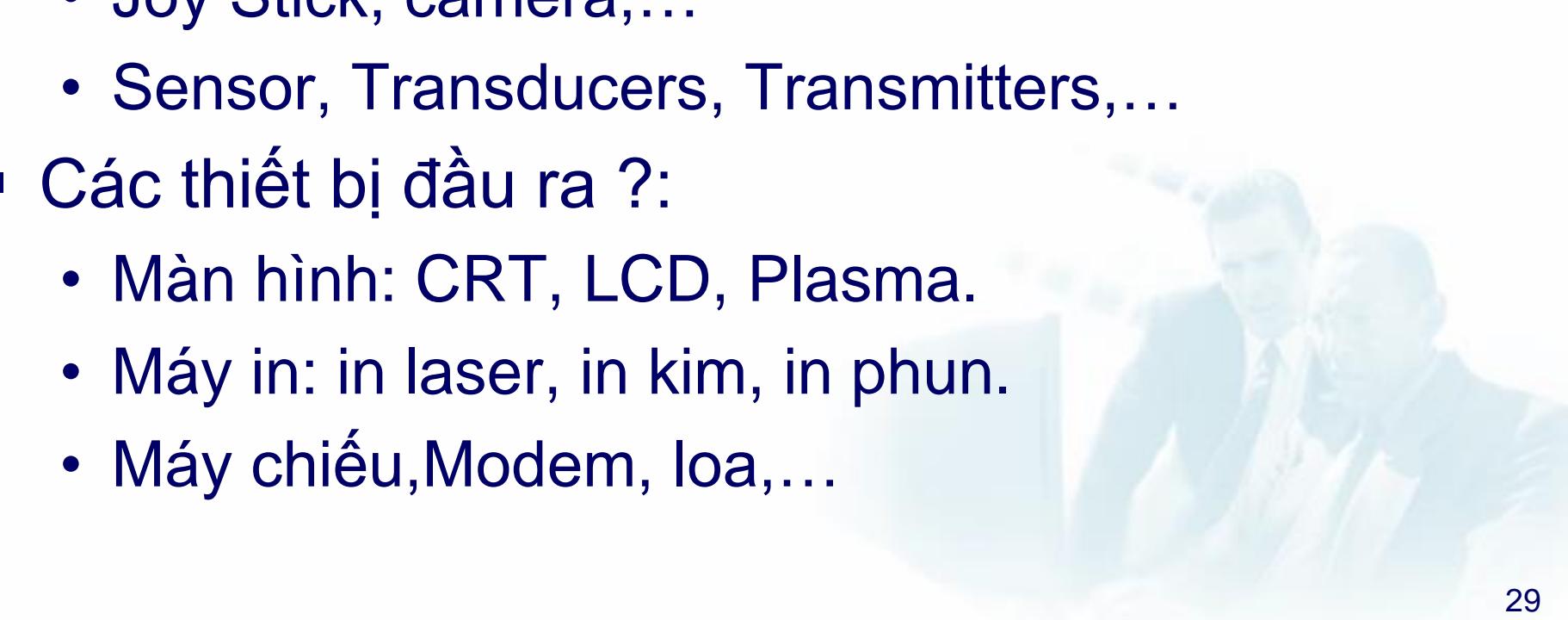


Hình 1. Sơ đồ khái niệm VXL/MT kinh điển



# Thiết bị ngoại vi

- Các thiết bị đầu vào :
  - Chuột, bàn phím, touch screen.
  - Modem, NIC
  - Joy Stick, camera,...
  - Sensor, Transducers, Transmitters,...
- Các thiết bị đầu ra :
  - Màn hình: CRT, LCD, Plasma.
  - Máy in: in laser, in kim, in phun.
  - Máy chiếu, Modem, loa,...





# Thiết bị ngoại vi

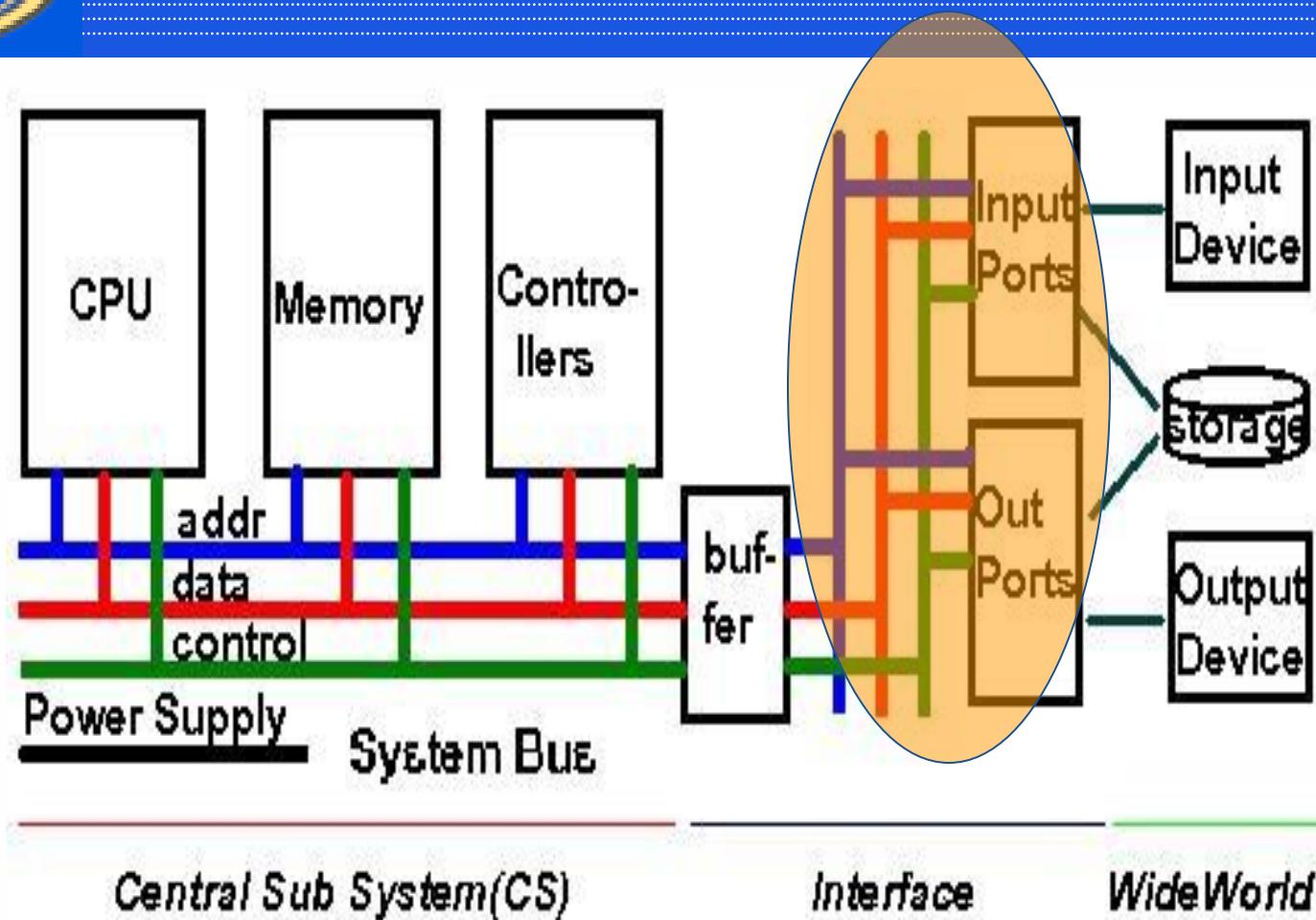
## ▪ Các thiết bị lưu trữ ?:

- Đĩa mềm
- Đĩa cứng
- Đĩa quang: CD, DVD
- Bộ nhớ flash \*





# Interface



Hình 1. Sơ đồ khái niệm VXL/MT kinh điển



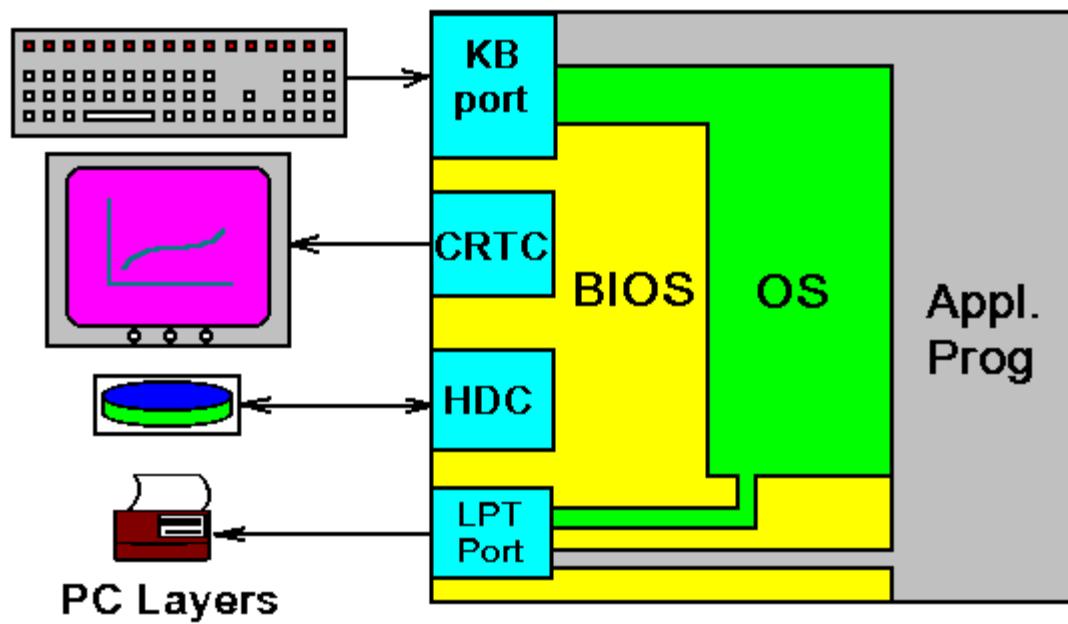
# Giao tiếp (interface)

- Giao tiếp (interface): Làm nhiệm vụ kết nối Hệ trung tâm (CS) với Thiết bị ngoại vi.
- Lý do cần có các interface ?:
  - Sự khác nhau giữa CS với các thiết bị ngoại vi: mức tín hiệu, tốc độ, chế độ làm việc đồng bộ hay không,...
- Để thực hiện giao tiếp, cần có:
  - Mạch điện tử thích ứng và Chương trình điều khiển (device driver).
  - Mạch điện tử: Cổng vào-ra, Controller, các bộ chuyển đổi AD-DA.



Ví dụ

Vd: Máy IBM-PC \*





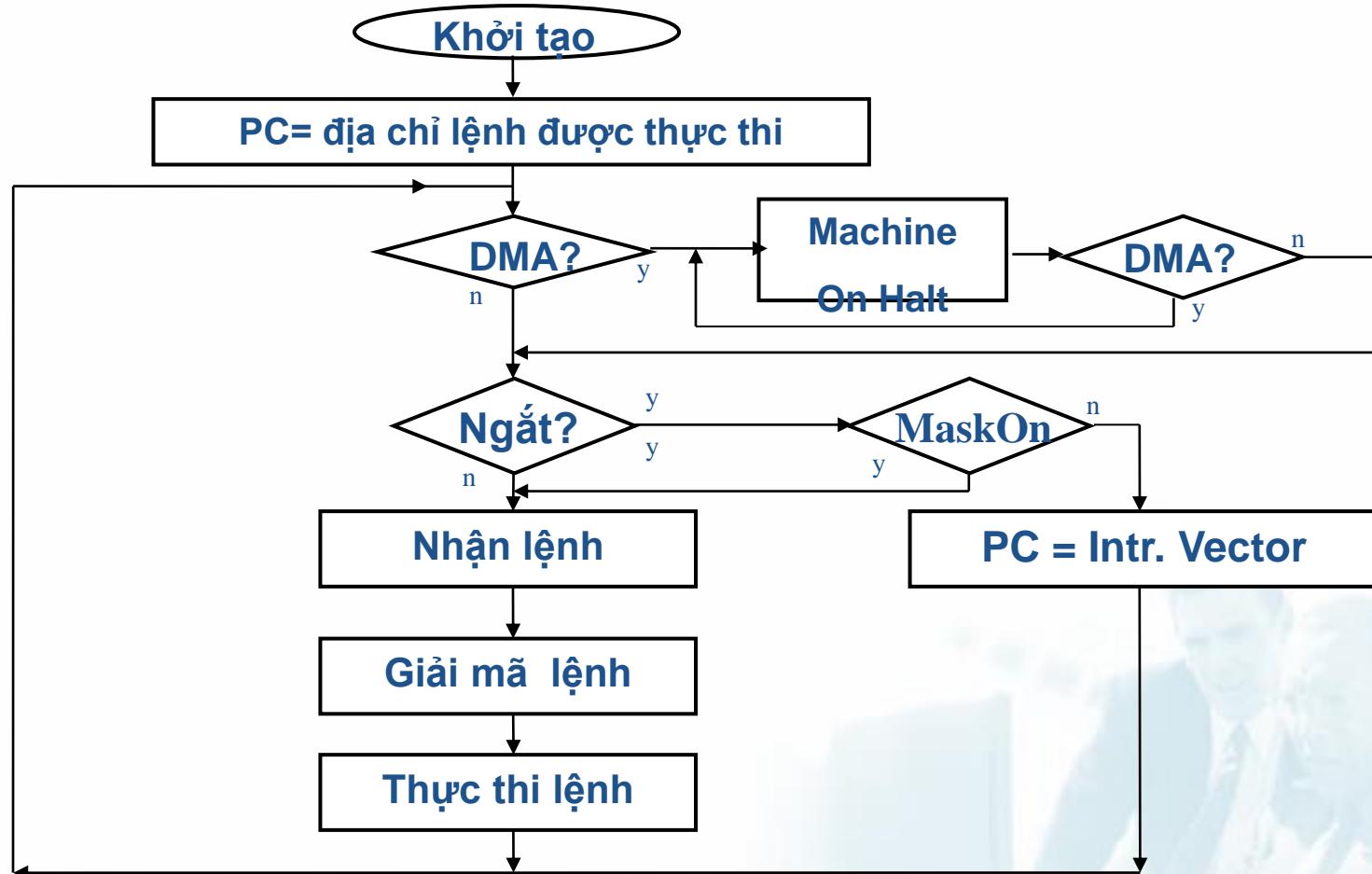
## 1.2. Hoạt động của hệ thống

- **Hoạt động của hệ thống:**

- Khởi tạo hệ thống
- Thực hiện truy cập bộ nhớ trực tiếp DMA (nếu có)
- Thực hiện chương trình con phục vụ ngắn (nếu có)
- Thực hiện chương trình:  
nhận lệnh, giải mã, thực thi lệnh.



# Hoạt động của hệ thống



Luật tàng quật của hệ VXL



# Khởi tạo hệ thống

- Khởi tạo hệ thống: xảy ra khi ?:
  - Hệ thống được bật lần đầu :Cold boot/cold start
  - Hệ thống bị khởi động lại do nguồn điện cung cấp tắt-bật đột ngột: hard reboot/cold reboot/frozen reboot/ hard reset
    - ✓ vd: khi bấm nút reset



# Khởi tạo hệ thống

- Khởi tạo hệ thống: xảy ra khi:
  - Hệ thống được khởi động lại với sự kiểm soát của phần mềm, nguồn điện cung cấp vẫn được duy trì: Warm reboot/ soft reboot / soft reset
    - ✓ vd: khi bấm Ctr + Alt + Dell hoặc chọn restart



# Khởi tạo hệ thống

- Khi được khởi tạo/ hard reset/ soft reset, hệ thống thực hiện chạy firmware code/ BIOS và thực hiện quá trình POST (Power-on Self Test):
  - Xác định lý do thực hiện POST (do cold reset hay warm reset) để thực hiện các chức năng tương ứng \*
  - Tìm kiếm, xác định dung lượng và kiểm tra bộ nhớ chính (main memory : RAM, ROM, Cache)



# Khởi tạo hệ thống

## ▪ POST (tiếp):

- Tìm, khởi tạo, liệt kê hệ thống bus và các thiết bị
- Tìm nạp hệ điều hành và chuyển quyền điều khiển hệ thống (trong các máy PC)



# Thực hiện chương trình

- Chương trình máy tính là một tập các lệnh có cấu trúc nhằm thực hiện một nhiệm vụ nào đó.
- Chương trình được thực hiện bằng cách lặp đi lặp lại chu trình lệnh ?:
  - Nhận lệnh
  - Giải mã và thực thi lệnh





# Nhận lệnh

- Bắt đầu mỗi chương trình, CPU sẽ nhận lệnh từ bộ nhớ chính và đưa vào bên trong CPU
- Bên trong CPU có 2 thanh ghi liên quan trực tiếp đến quá trình nhận lệnh:
  - Thanh ghi bộ đếm chương trình ( Program Counter –PC): chứa địa chỉ của lệnh sẽ được nhận vào.
  - Thanh ghi lệnh ( Instruction Register – IR): Nội dung của lệnh sẽ được chứa trong IR.



# Nhận lệnh(tiếp)

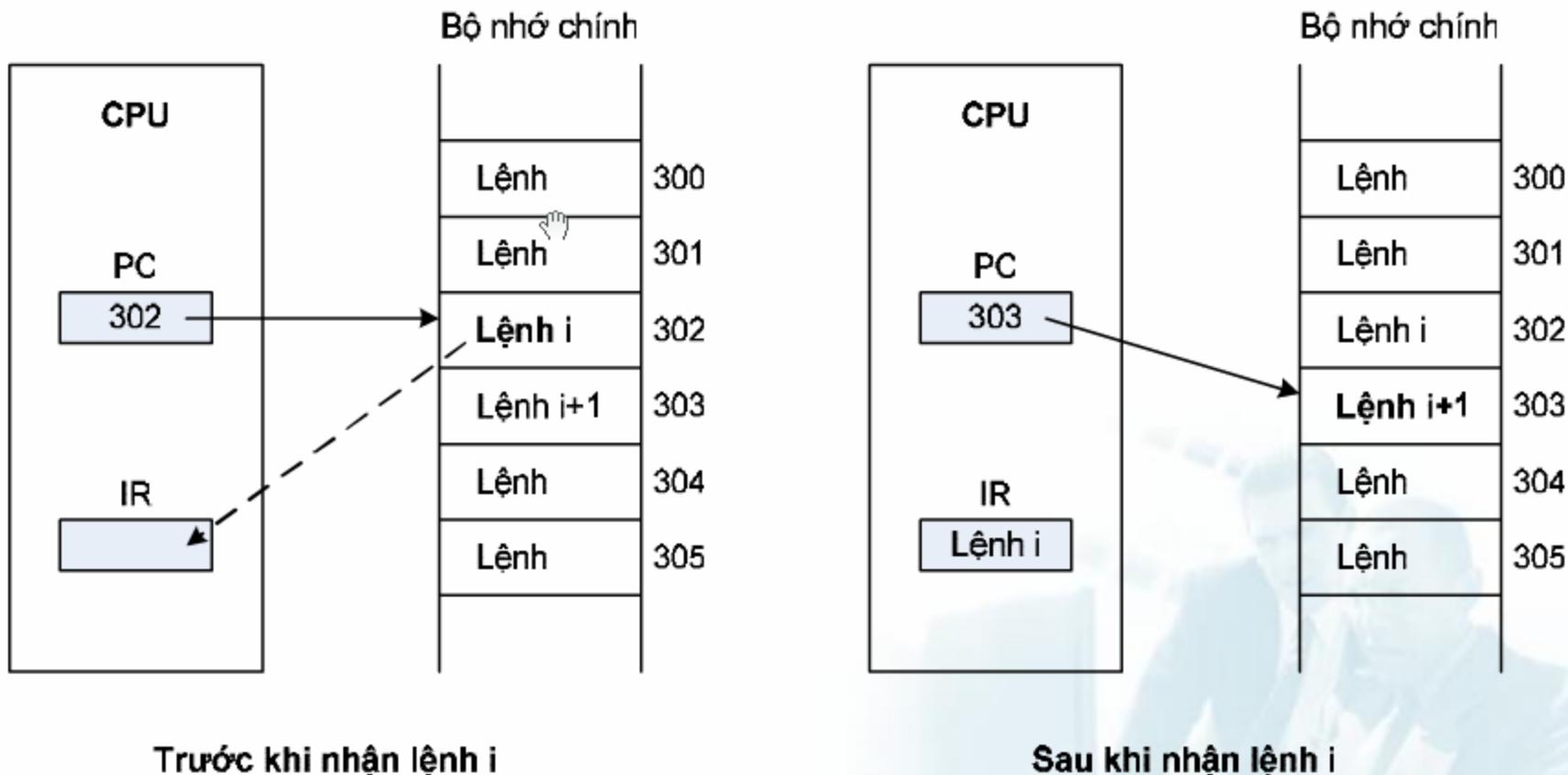
- **Hoạt động nhận lệnh :**

- CPU phát địa chỉ của lệnh cần nhận từ PC tới bộ nhớ chính
- CPU phát tín hiệu điều khiển đọc bộ nhớ chính (Memory Read –MEMR) tại địa chỉ tương ứng
- Tiếp (?)
- Lệnh từ bộ nhớ chính được chuyển vào IR
- Nội dung PC tự động tăng để chỉ tới lệnh kế tiếp



# Nhận lệnh (tiếp)

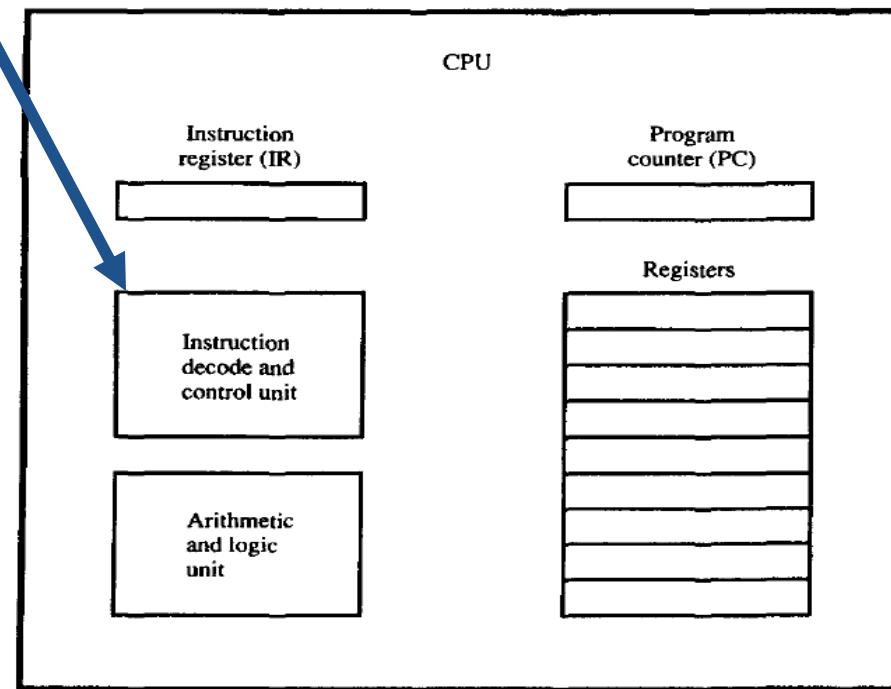
## ■ Minh họa quá trình nhận lệnh:





# Giải mã và thực thi lệnh

- Lệnh nằm ở IR sẽ được chuyển sang **Khối giải mã lệnh và điều khiển**. Tại đây lệnh sẽ được giải mã và đơn vị điều khiển sẽ phát ra các tín hiệu để thực thi các thao tác mà lệnh yêu cầu





# Giải mã và thực thi lệnh(tiếp)

- Các kiểu thao tác của lệnh:

- Trao đổi dữ liệu giữa CPU và bộ nhớ chính
- Trao đổi dữ liệu giữa CPU và các môđun vào-ra
- Xử lý dữ liệu: thực hiện các phép toán số học hoặc logic
- Điều khiển rẽ nhánh
- Kết hợp các thao tác trên





# Hoạt động ngắt

- Khái niệm chung về ngắt (interrupt):
  - ngắt là cơ chế cho phép CPU tạm dừng chương trình đang thực hiện để chuyển sang thực hiện một chương trình khác, gọi là chương trình con phục vụ ngắt.
- Các loại ngắt:
  - Ngắt do lỗi khi thực hiện chương trình: tràn số, chia 0,...
  - Ngắt do lỗi phần cứng
  - Ngắt do các môđun vào-ra yêu cầu trao đổi dữ liệu

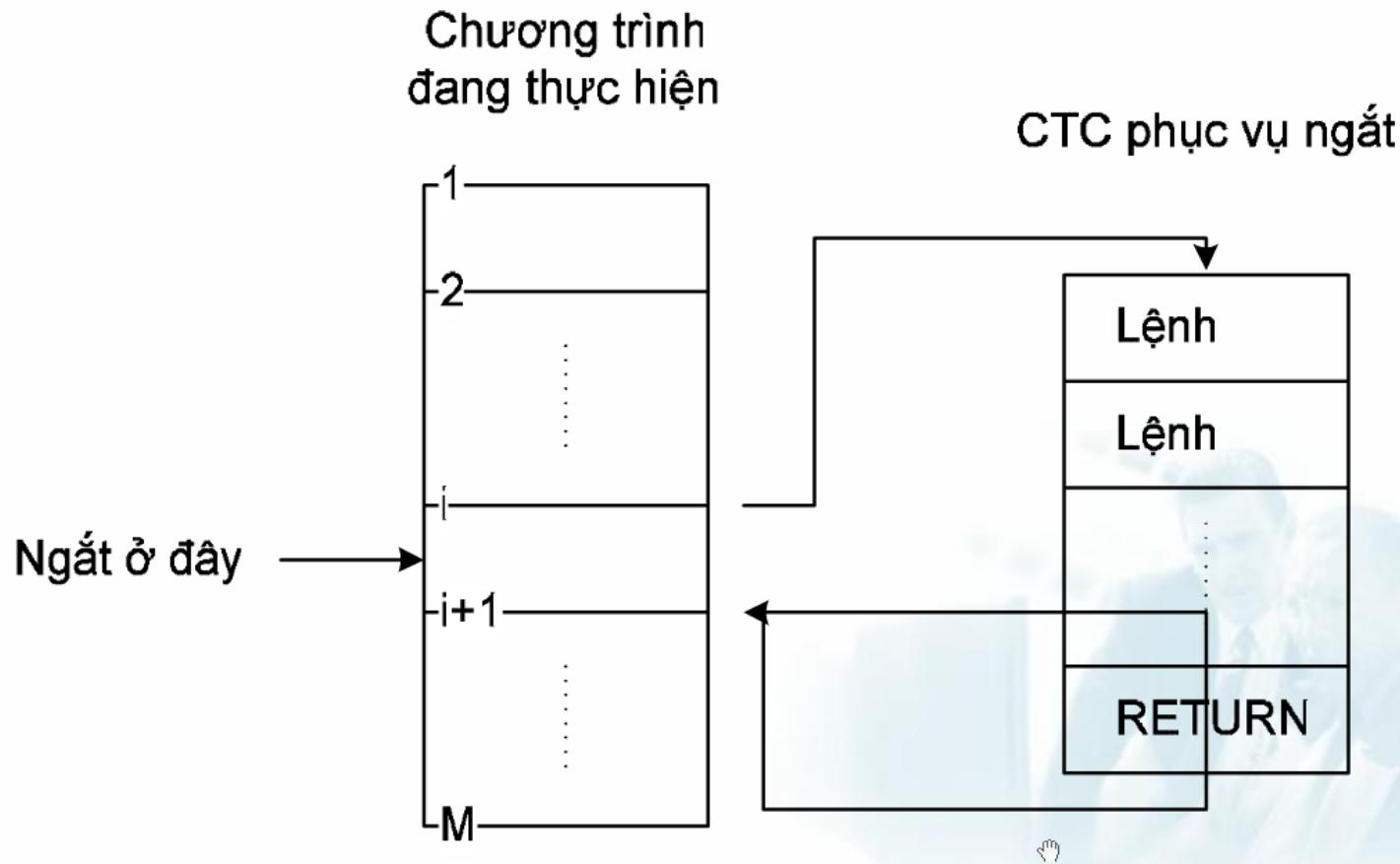


# Chu trình xử lý ngắt

- tương ứng Sau khi hoàn thành một lệnh, CPU kiểm tra xem có yêu cầu ngắt gửi tới hay không
  - Nếu không có tín hiệu yêu cầu ngắt thì CPU nhận lệnh kế tiếp
  - Nếu có yêu cầu ngắt và ngắt đó được chấp nhận thì :
    - ✓CPU cất **ngữ cảnh** hiện tại của chương trình đang thực hiện (các thông tin liên quan tới chương trình bị ngắt)
    - ✓CPU chuyển sang thực hiện chương trình con phục vụ ngắt
    - ✓Kết thúc chương trình con đó, CPU khôi phục lại **ngữ cảnh** và tiếp tục thực hiện chương trình chính

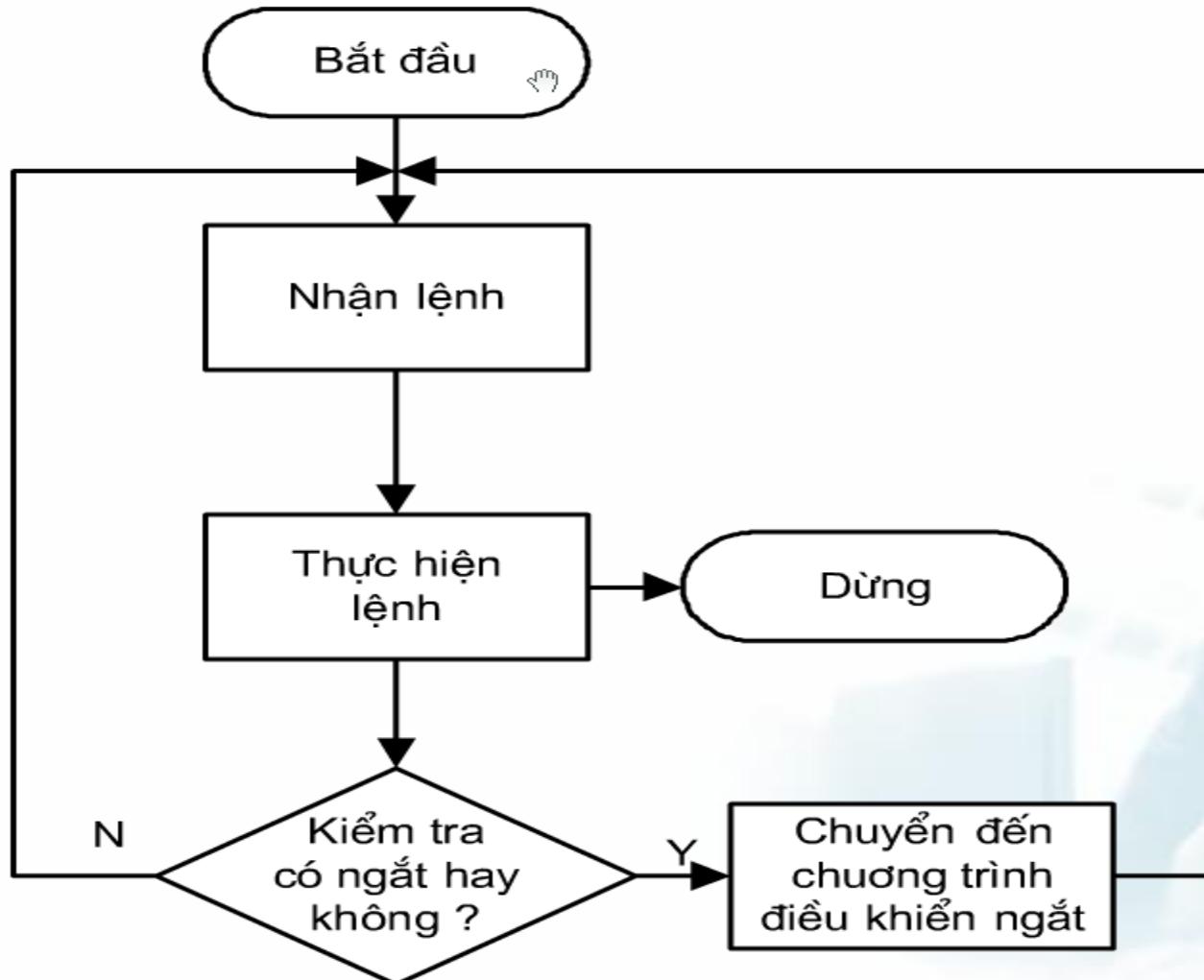


# Hoạt động ngắt (tiếp)





# Hoạt động ngắt (tiếp)

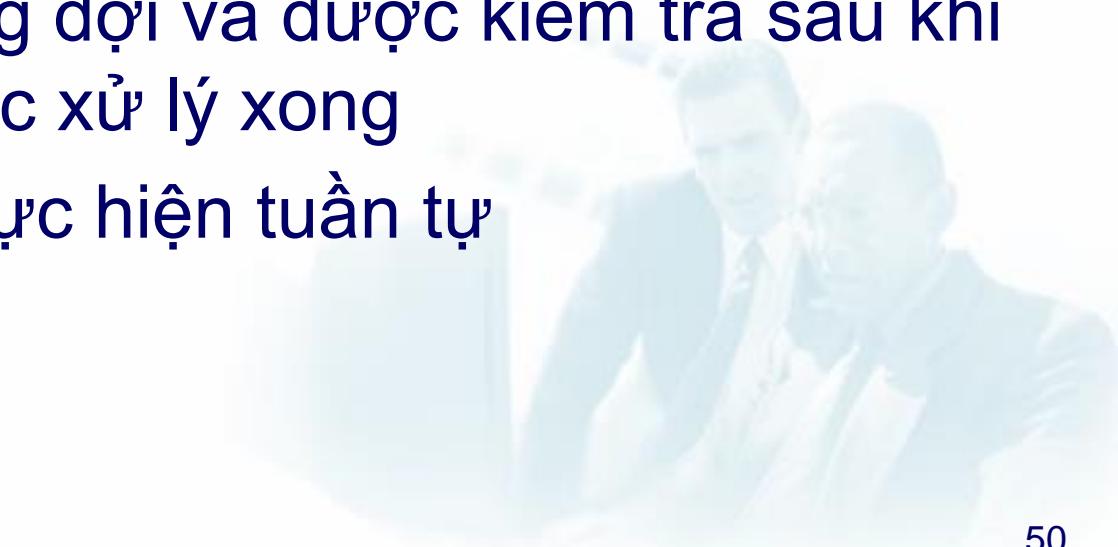




# Xử lý với nhiều tín hiệu yêu cầu ngắt

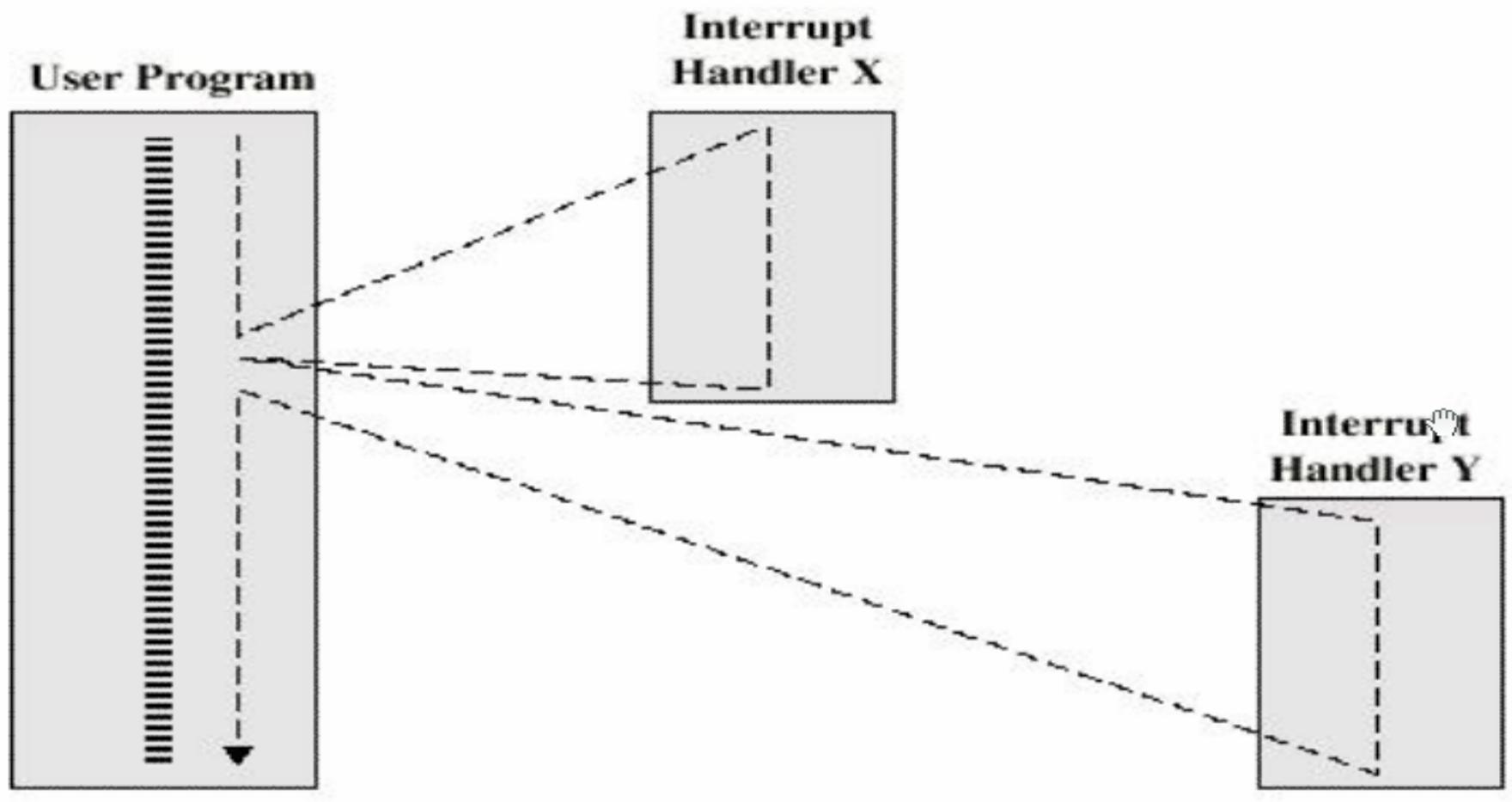
- Xử lý ngắt tuần tự:

- Khi một ngắt đang được thực hiện, các ngắt khác sẽ bị cấm
- Bộ xử lý sẽ bỏ qua các ngắt tiếp theo trong khi đang xử lý một ngắt
- Các ngắt vẫn đang đợi và được kiểm tra sau khi ngắt đầu tiên được xử lý xong
- Các ngắt được thực hiện tuần tự





# Xử lý ngắt tuần tự





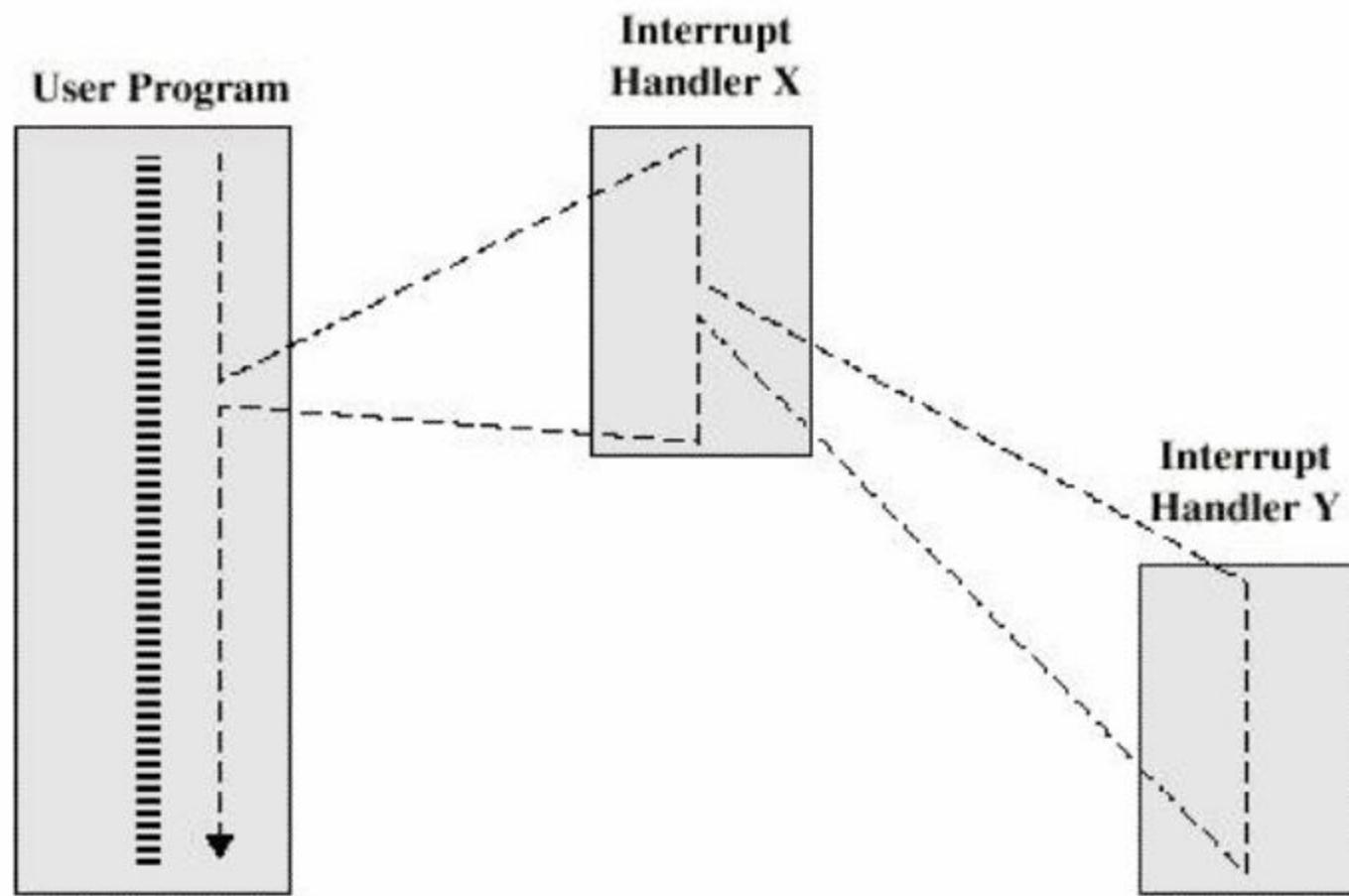
## Xử lý ngắt ưu tiên

- Các ngắt được định nghĩa mức ưu tiên khác nhau
- Ngắt có mức ưu tiên thấp hơn có thể bị ngắt bởi ngắt có ưu tiên cao hơn





# Xử lý ngắt ưu tiên





## Truy nhập trực tiếp bộ nhớ (DMA)

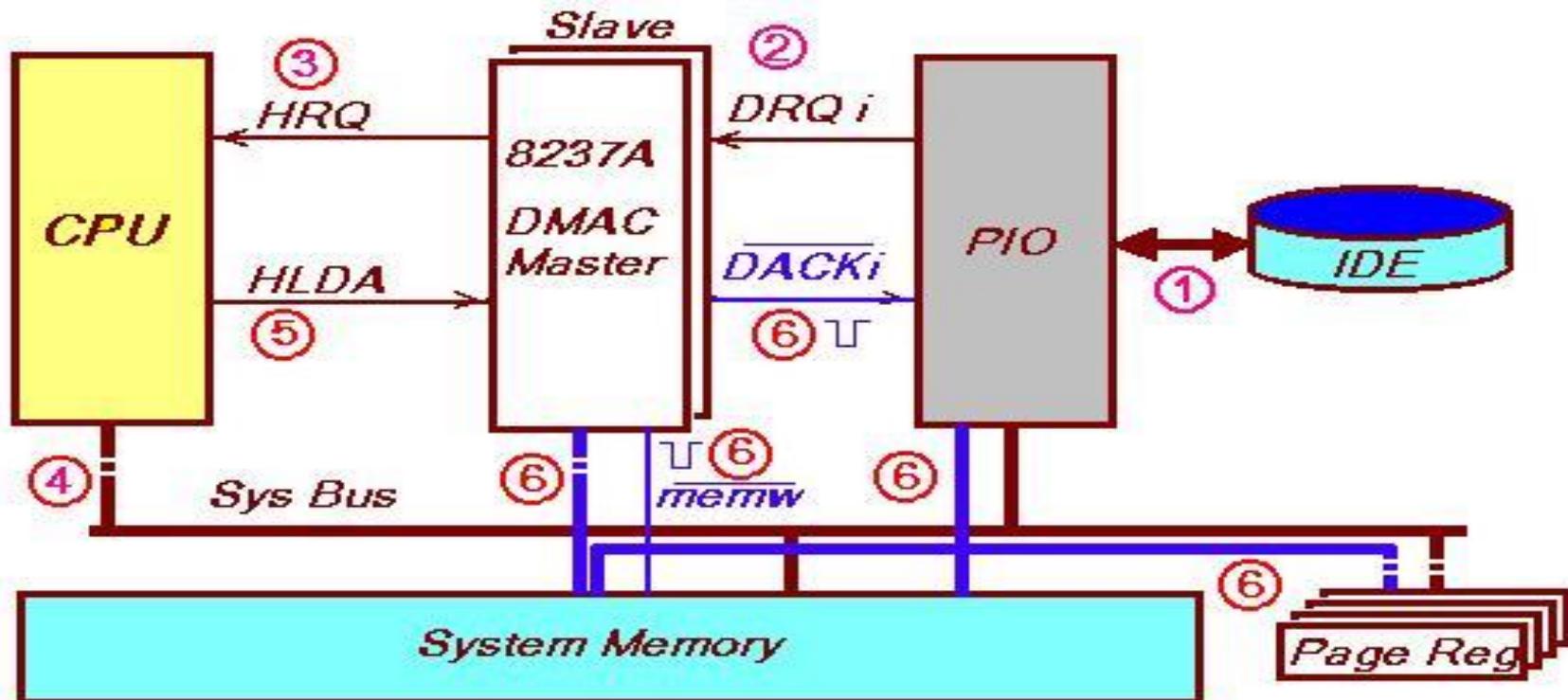
- Việc thực hiện trao đổi dữ liệu được thực hiện bằng phần cứng DMA Controller
- Vì xử lý không phải đọc, giải mã và thực hiện các lệnh để trao đổi dữ liệu, nó hoàn toàn chuyển quyền điều khiển bus cho chip DMAC và số liệu được truyền trực tiếp giữa ngoại vi và bộ nhớ dưới sự điều khiển của DMAC.





# Truy nhập trực tiếp bộ nhớ DMA

## ▪ Hoạt động DMA:

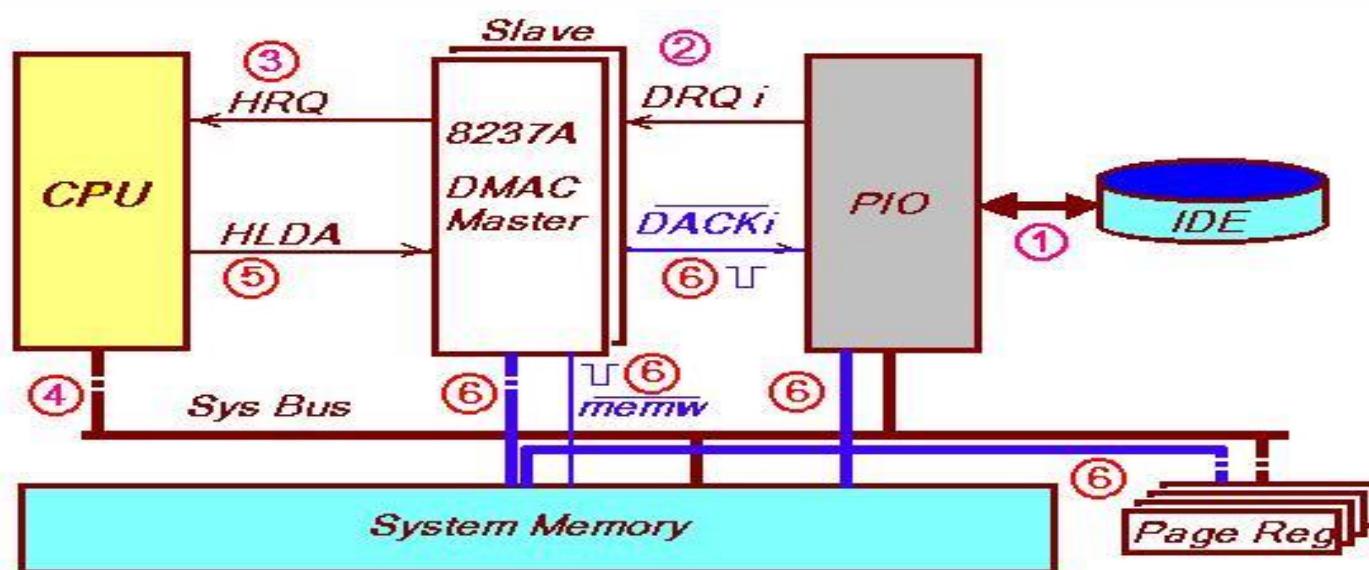


Hình 3.8. Sơ đồ mô tả hoạt động DMA của máy PC



# Truy nhập trực tiếp bộ nhớ DMA

- 1) Ngoại vi chuyển dữ liệu tới cổng vào ra
- 2) Cổng vào ra phát tín hiệu yêu cầu truy cập bộ nhớ trực tiếp DRQ (Direct Request) tới DMAC



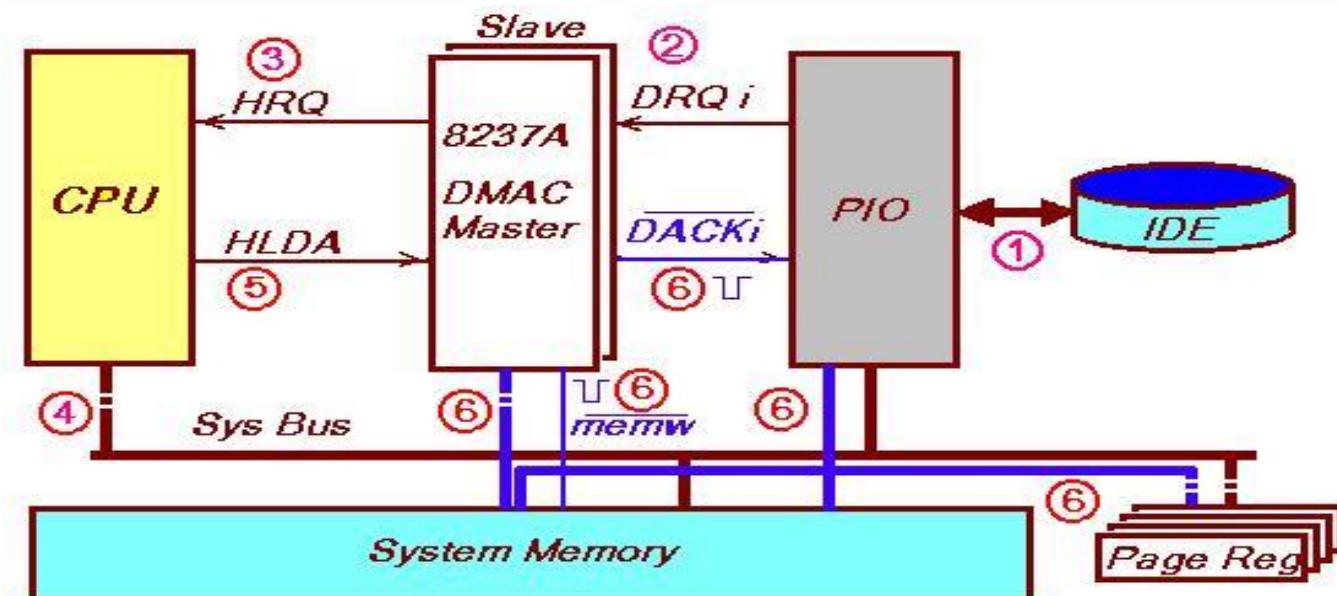
Hình 3.8. Sơ đồ mô tả hoạt động DMA của máy PC



# Truy nhập trực tiếp bộ nhớ DMA

3)DMAC gửi tín hiệu yêu cầu được nắm giữ bus:  
Hold Request

4)CPU thả nỗi bus



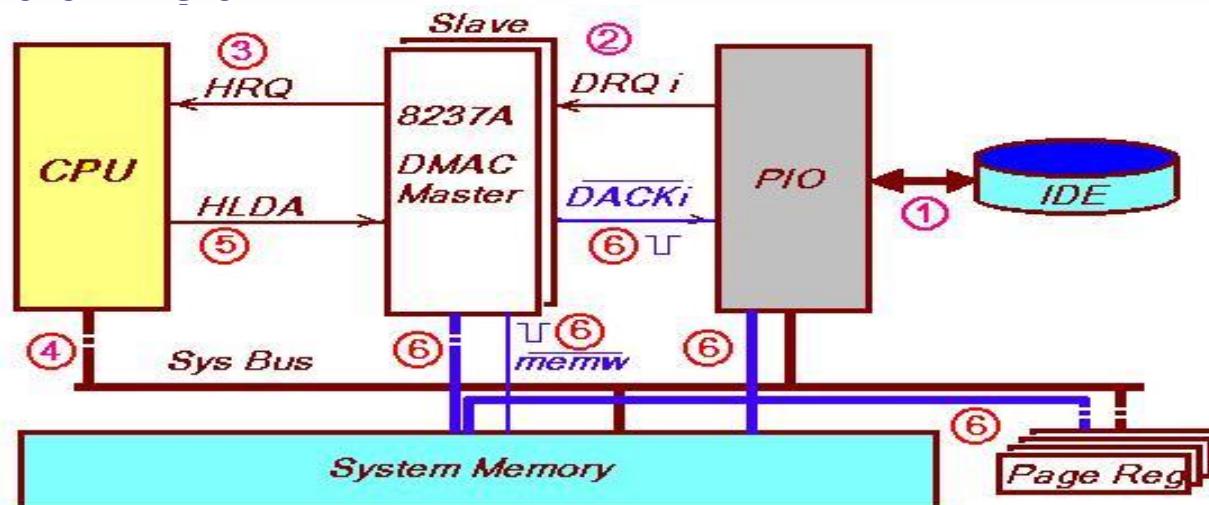
Hình 3.8. Sơ đồ mô tả hoạt động DMA của máy PC



# Truy nhập trực tiếp bộ nhớ DMA

5)CPU gửi tín hiệu báo cho DMAC biết : HLDA ( Hold Acknowledgement) đây, tôi ko dùng đến bus, anh dùng đi.

6)DMAC thực hiện gửi các tín hiệu điều khiển để trao đổi dữ liệu



Hình 3.8. Sơ đồ mô tả hoạt động DMA của máy PC



# Chương 2-Kỹ thuật ghép nối

2.1. Các phương pháp truyền tin

2.2. Giao thức ghép nối



# Các phương pháp truyền tin

- Truyền song song (Parallel) & truyền nối tiếp (Serial)
- Truyền đồng bộ (Synchronous) & không đồng bộ (Asynchronous)





## 2.1. Các phương pháp truyền tin

### Truyền song song

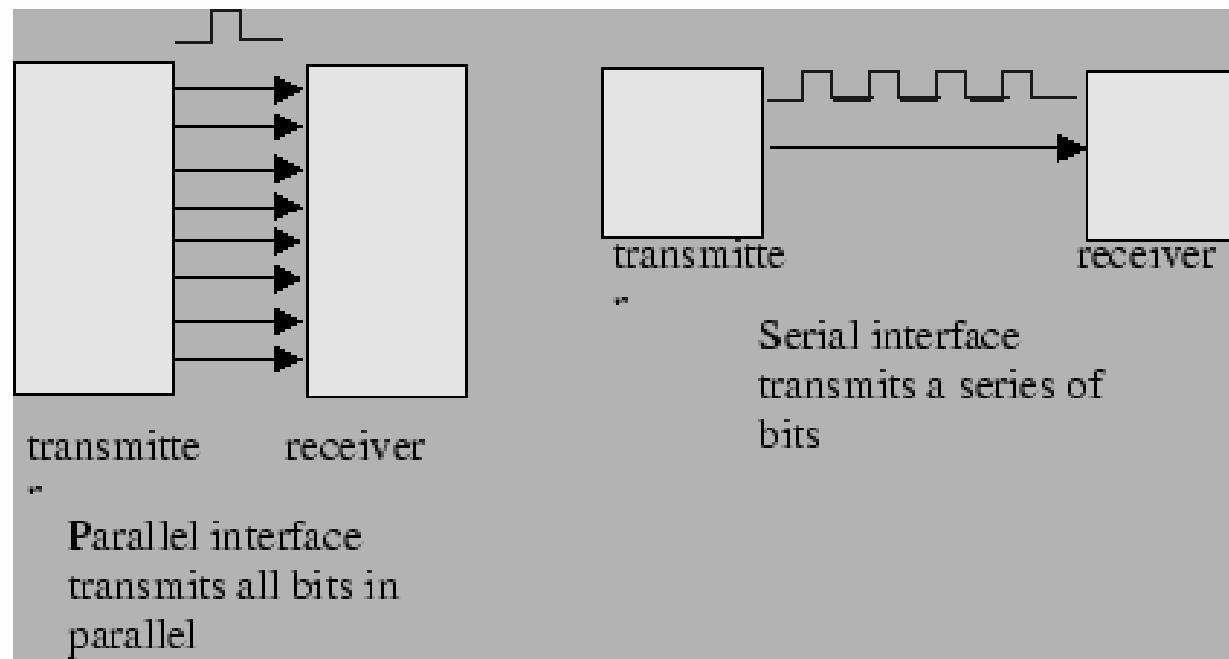
- Cho phép truyền số liệu đồng thời các bit của một từ dữ liệu trong một nhịp truyền.
- **Ưu điểm:**  
Truyền đồng thời được nhiều bit dữ liệu
- **Nhược điểm:**  
Khoảng cách truyền ngắn
- VD: cỗng song song (LPT), bus vào ra (PCI, ISA)



## 2.1. Các phương pháp truyền tin

### Truyền nối tiếp

- Trao đổi thông tin theo phương thức truyền nối tiếp là truyền liên tiếp từng bit một trên một đường truyền.
- VD: PS2, COM, USB...





# Truyền tin nối tiếp

## ▪ Lý do sử dụng truyền thông tin nối tiếp:

- Bus dữ liệu của hệ VXL được thiết kế để trao đổi dữ liệu song song với các mạch vào-ra. Tuy nhiên trong nhiều trường hợp, người ta phải thực hiện trao đổi thông tin nối tiếp có tốc độ chậm hơn
- Khoảng cách giữa hai đơn vị cần trao đổi dữ liệu là tương đối lớn -> giảm giá thành





## Truyền tin nối tiếp

Hệ thống trao đổi thông tin nối tiếp gồm có các dạng:

- Đơn công (Simplex Connection): số liệu chỉ được truyền theo 1 hướng.





# Truyền tin nối tiếp

- Bán song công (Half-Duplex): số liệu có thể truyền đi theo 2 hướng, nhưng mỗi thời điểm chỉ được truyền theo 1 hướng.



- Song công (Full-Duplex): số liệu được truyền đồng thời theo 2 hướng





# Truyền tin nối tiếp

## ▪ Phương thức hoạt động:

- Ở đầu phát, dữ liệu dưới dạng song song đầu tiên được chuyển thành dữ liệu dạng nối tiếp. Tín hiệu nối tiếp sau đó được truyền đi liên tiếp từng bit trên đường dây.
- Ở đầu thu, tín hiệu nối tiếp sẽ được biến đổi ngược lại để chuyển sang dạng song song thích hợp cho việc xử lý tiếp theo



## Truyền dữ liệu nối tiếp đồng bộ

- Các thiết bị sử dụng chung 1 nguồn xung clock phát bởi 1 thiết bị hoặc từ nguồn ngoài. Mỗi bit truyền đi tại thời điểm xung clock chuyển mức (sườn lên hoặc sườn xuống của xung).
- Bộ nhận sử dụng sự chuyển mức của xung để xác định thời điểm đọc các bit. Nó có thể đọc các bit tại sườn lên hay sườn xuống của xung hoặc theo mức logic cao thấp.



## Truyền dữ liệu nối tiếp không đồng bộ

- Một gói dữ liệu truyền đi bao gồm bit start để đồng bộ hóa nguồn clock, 1 hoặc nhiều hơn 1 bit stop để báo kết thúc truyền 1 byte.
- Ngoài ra khung truyền còn có bit parity có thể là even, odd, mark hay space.



## 2.2. Giao thức ghép nối

- 2.2.1. Giao thức ghép nối (Interfacing Protocol)
- 2.2.2. Chuẩn RS232
- 2.2.3. Chuẩn LPT
- 2.2.4. Chuẩn USB
- 2.2.5. Chuẩn IEEE1394



## 2.2.1. Giao thức ghép nối

- Giao thức ghép nối là các quy định về
  - Tín hiệu (signals)
  - Khuôn dạng dữ liệu (Data format)
  - Tốc độ (Rate)
  - Cơ chế phát hiện và sửa lỗi (Error Detection & Error correction)
  - Lệnh truyền & phản hồi (Command set & response)
  - Kịch bản (scenario)



# Giao thức ghép nối

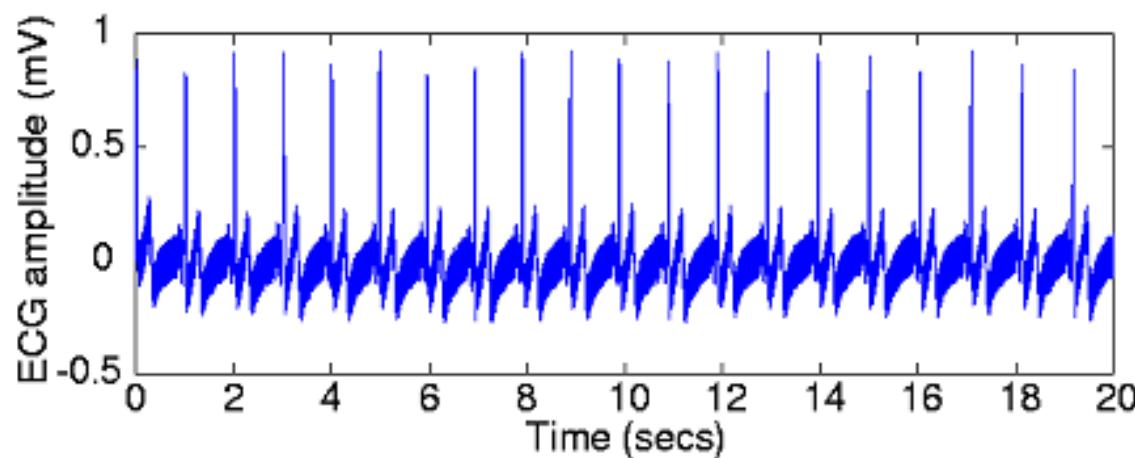
- Tín hiệu (signals)

- Khái niệm về tín hiệu: tương tự, rời rạc, lượng tử hóa, số
- Mô hình ghép nối Analog



## Khái niệm và phân loại

- Tín hiệu là biểu hiện vật lý của thông tin
- Về mặt toán, tín hiệu là hàm của một hoặc nhiều biến độc lập. Các biến độc lập có thể là: thời gian, áp suất, độ cao, nhiệt độ...
- Biến độc lập thường gấp là thời gian.
- Một ví dụ về tín hiệu có biến độc lập là thời gian: tín hiệu điện tim.

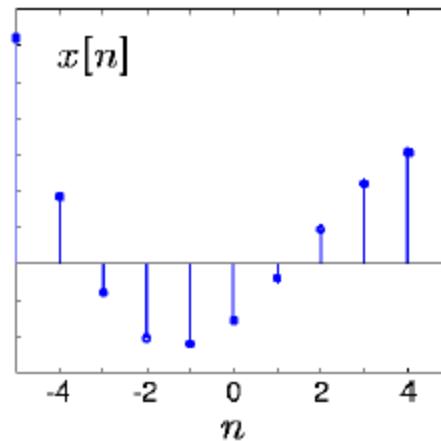
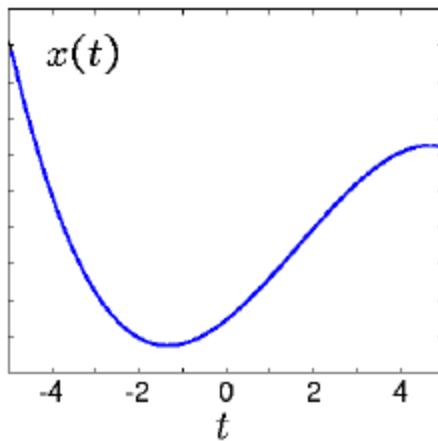




# Khái niệm và phân loại

## ■ Phân loại:

Xét trường hợp tín hiệu là hàm của biến thời gian



**Tín hiệu tương tự: biên độ (hàm), thời gian (biến) đều liên tục. Ví dụ:  $x(t)$**

**Tín hiệu rời rạc: biên độ liên tục, thời gian rời rạc.  
Ví dụ:  $x(n)$**

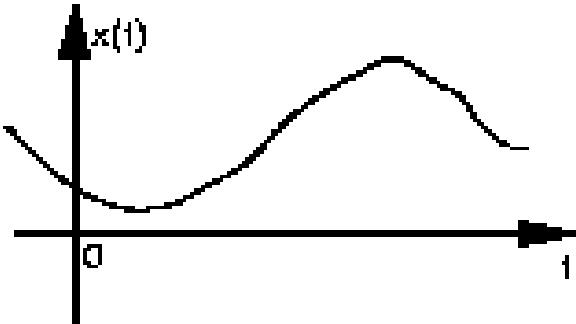
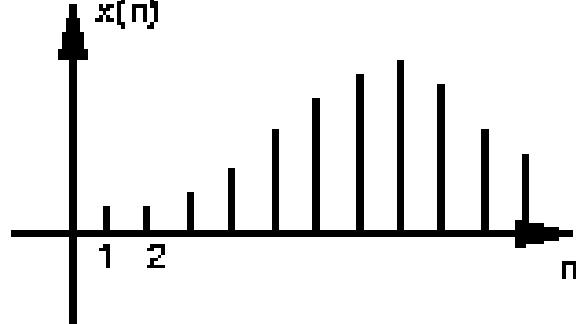
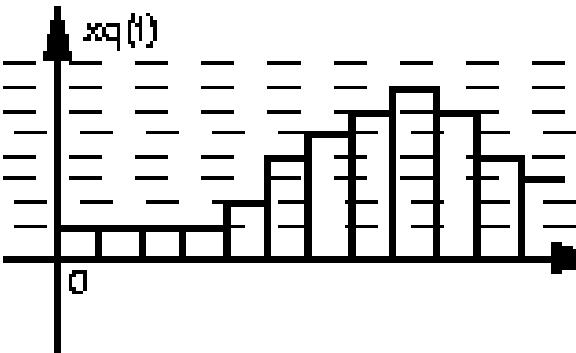
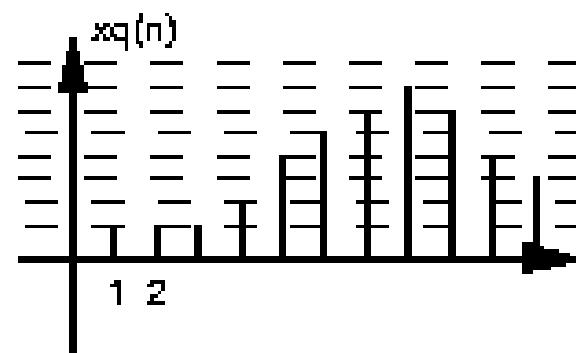


# Khái niệm và phân loại

- **Tín hiệu lượng tử hóa (Quantified signal): thời gian liên tục và biên độ rời rạc.** Đây là tín hiệu tương tự có biên độ đã được rời rạc hóa.
- **Tín hiệu số (Digital signal): thời gian rời rạc và biên độ cũng rời rạc.** Đây là tín hiệu rời rạc có biên độ được lượng tử hóa.



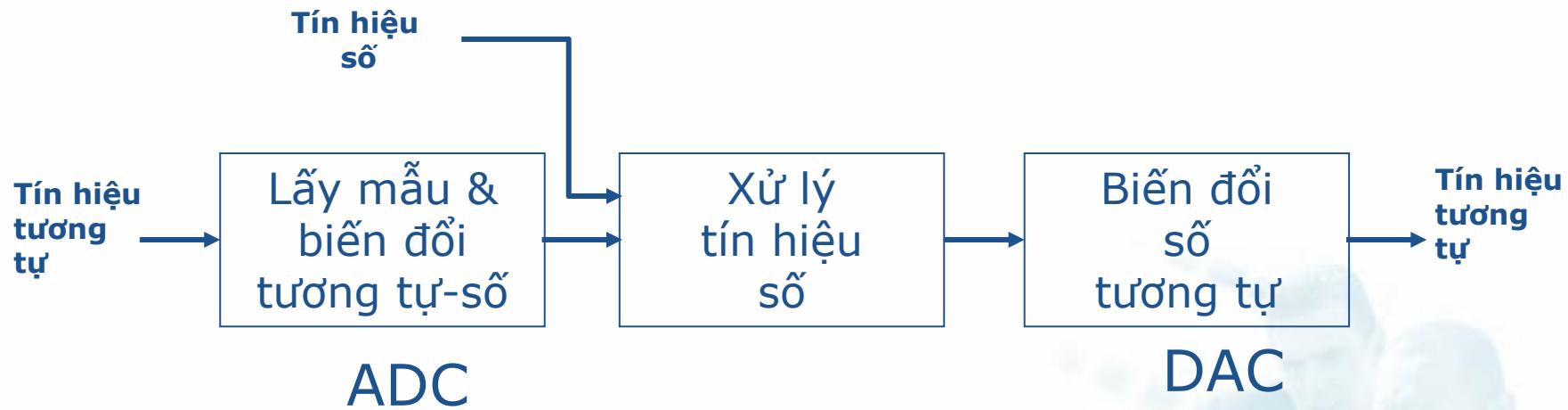
# Khái niệm và phân loại

	TEMPS CONTINU	TEMPS DISCRET
AMPLITUDE CONTINUE		
AMPLITUDE DISCRETE		



# Xử lý số tín hiệu

- Sơ đồ khái niệm xử lý số tín hiệu



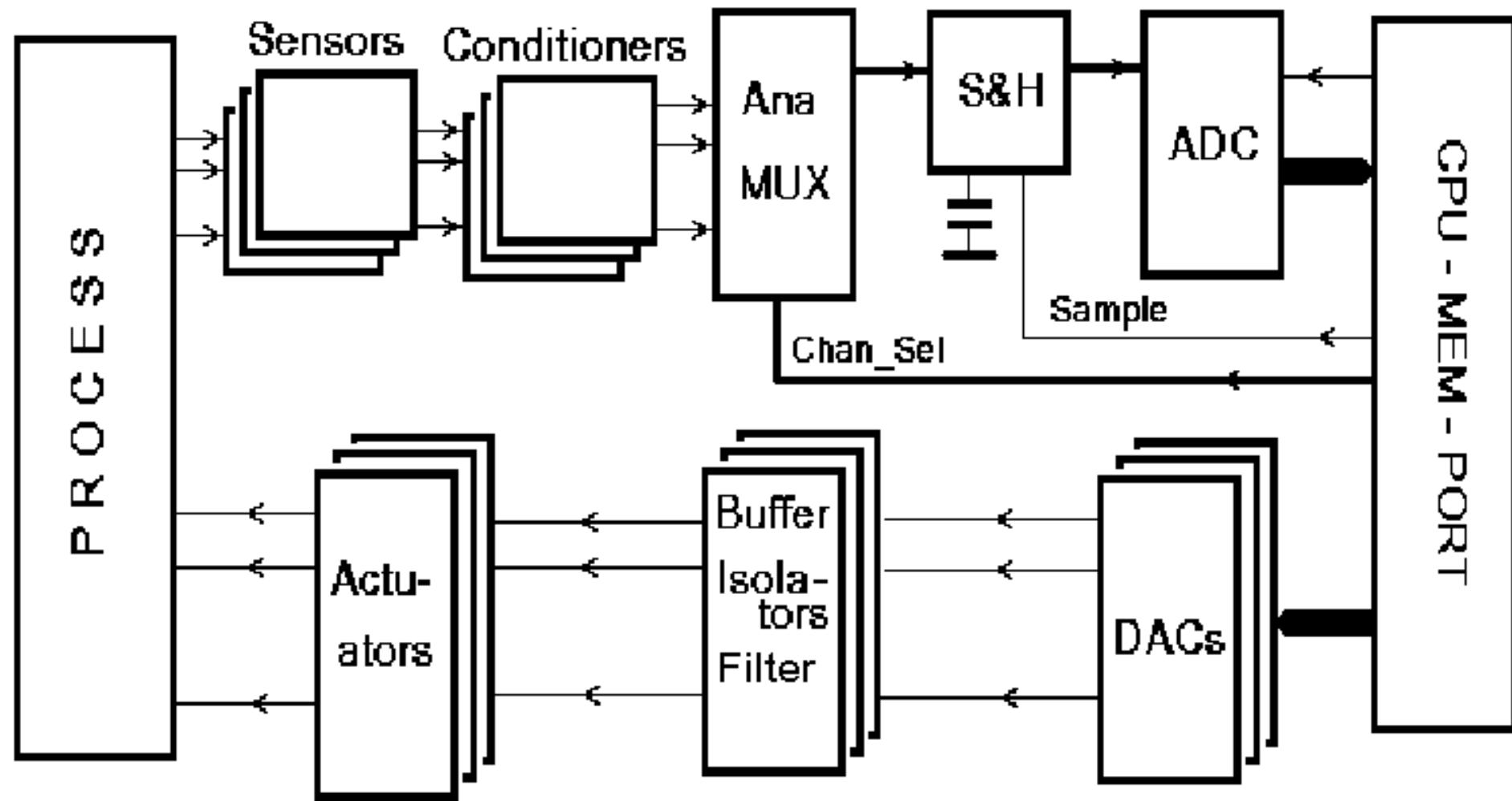


# Tại sao lại sử dụng tín hiệu số ?

- Để có thể xử lý tự động (bằng máy tính)
- Giảm được nhiễu
- Cho phép sao lưu nhiều lần mà chất lượng không thay đổi
- Các bộ xử lý tín hiệu số (DSP)
- Khi được chế tạo hàng loạt có chất lượng xử lý đồng nhất và chất lượng xử lý không thay đổi theo thời gian



# Mô hình ghép nối hệ đo lường-điều khiển số

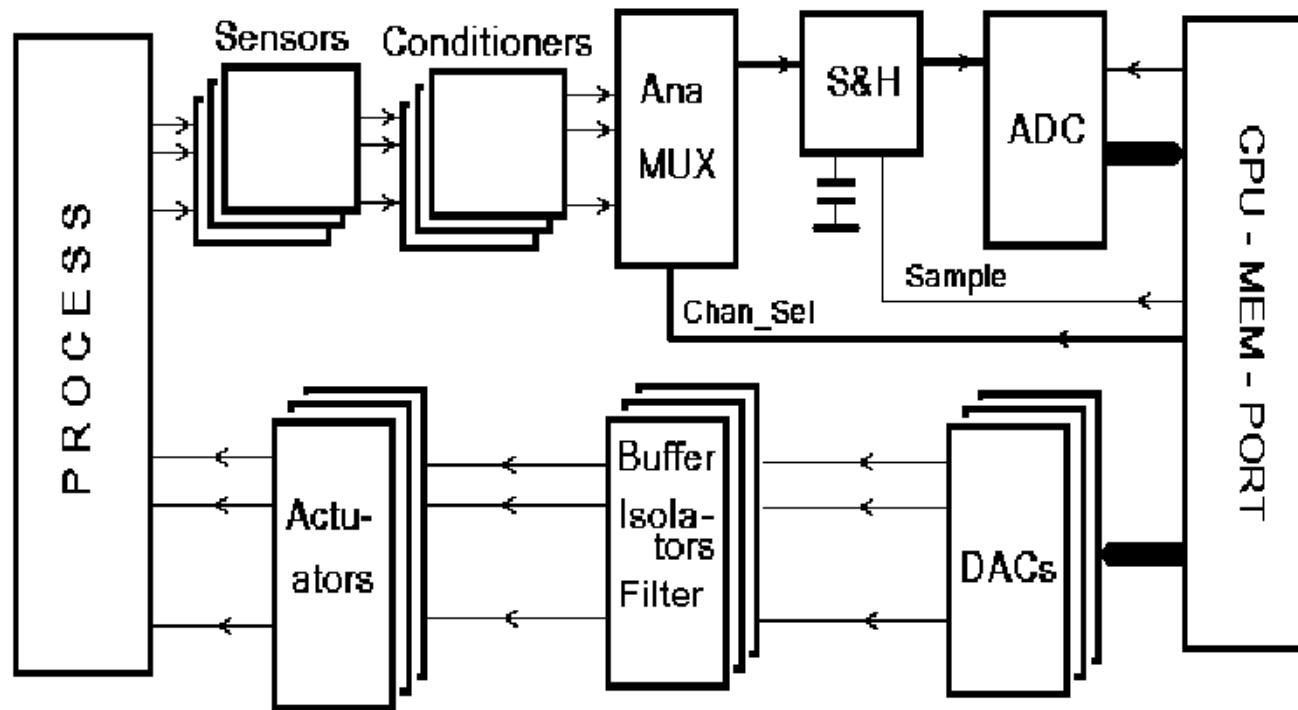




# Mô hình ghép nối hệ đo lường-điều khiển số

## ■ Process:

- Là các quá trình công nghệ như: dây chuyền giấy; dây chuyền luyện-nung-cát-thép, sản xuất-trộn phân bón NPK, các nhà máy phát điện...

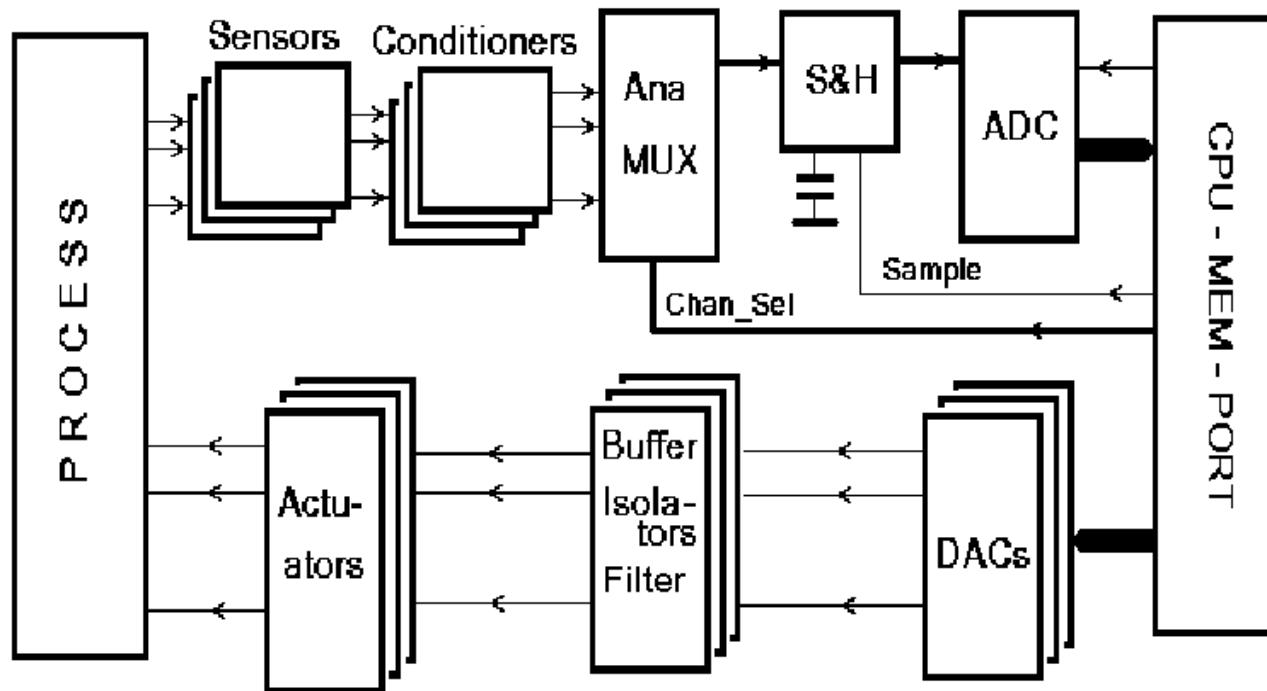




# Mô hình ghép nối hệ đo lường-điều khiển số

- Sensors:

- Là vật liệu/thiết bị dùng để chuyển đổi các đại lượng vật lý không điện (T, RH, p, L, v, a, F, pH, F,...) thành tín hiệu điện (u, i, R, f)
- Vật liệu: do đặc tính tự nhiên của vật chất – ví dụ cắp nhiệt độ, điện tim, ...

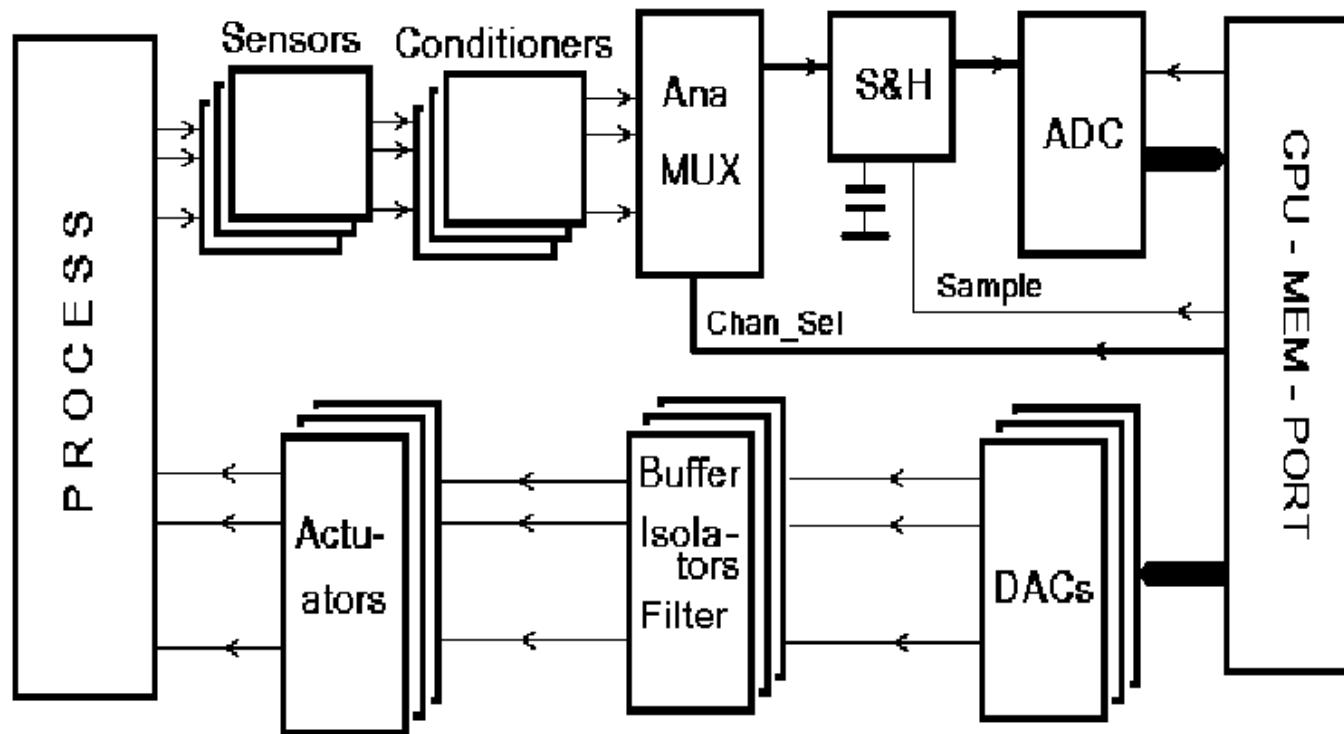




# Mô hình ghép nối hệ đo lường-điều khiển số

## ■ Conditioners:

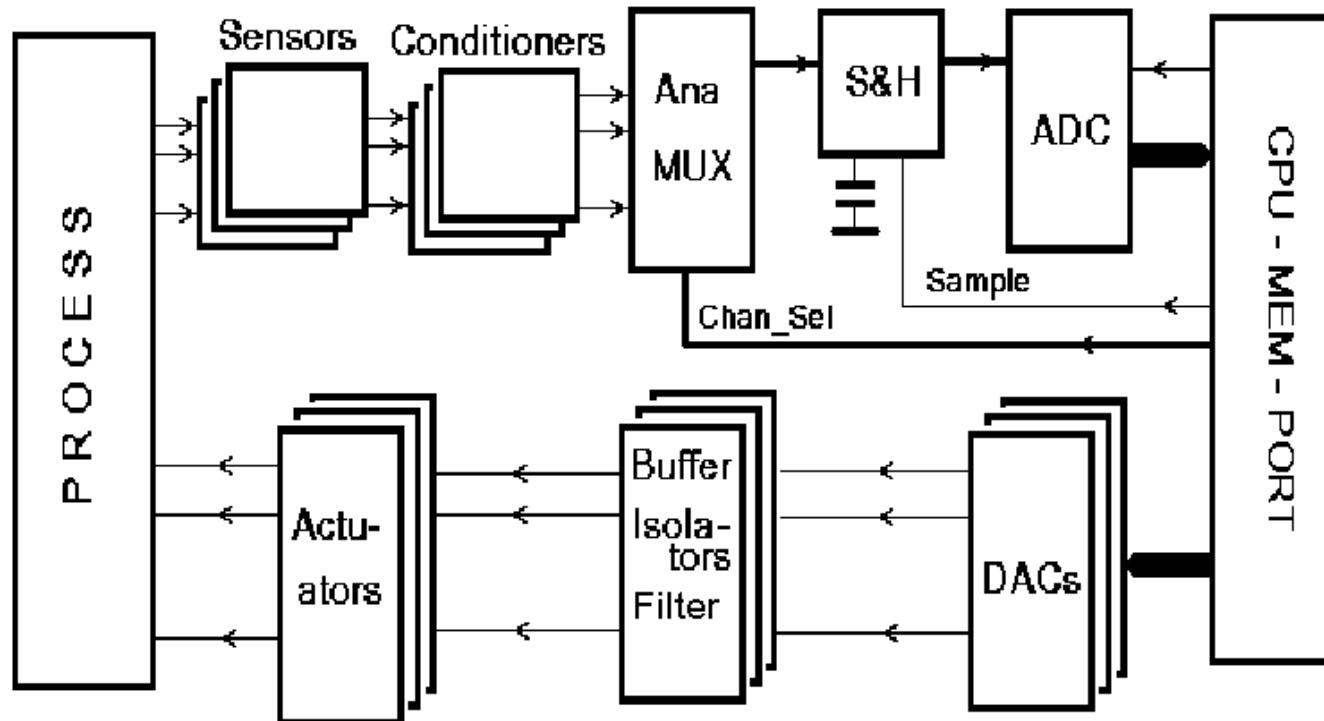
- Vì tín hiệu từ sensors thường rất nhỏ, có nhiễu và phi tuyến => có mạch điện tử analog để xử lý tín hiệu: khuếch đại, lọc nhiễu, bù phi tuyến... cho phù hợp.





# Mô hình ghép nối hệ đo lường-điều khiển số

- MUX: analog multiplexer – bộ dồn kênh
  - Inputs: n bit chọn kênh, có  $2^n$  kênh số đo analog, đánh số từ 0.. $2^n$ -1;
  - Output: 1 kênh chung thông với 1 trong số  $2^n$  inputs và duy nhất;
  - Như vậy chỉ cần 1 hệ VXL/MT và ADC vẫn thu thập được nhiều điểm đo công nghệ

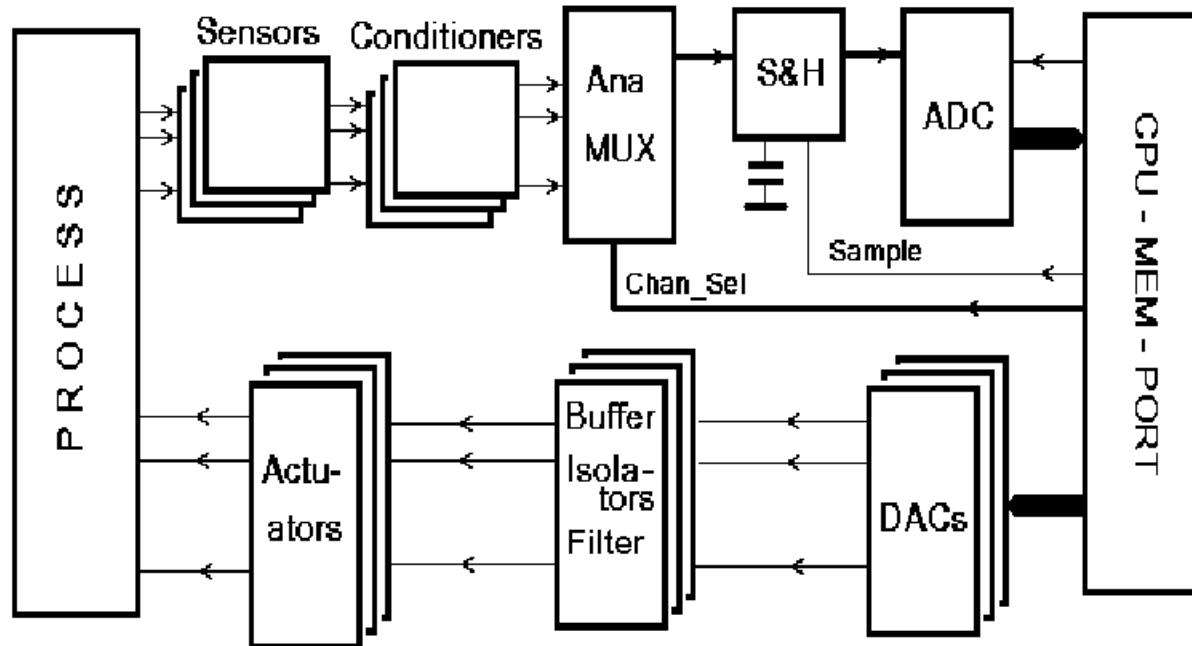




# Mô hình ghép nối hệ đo lường-điều khiển số

- Trích mẫu và giữ - Sample & Hold:

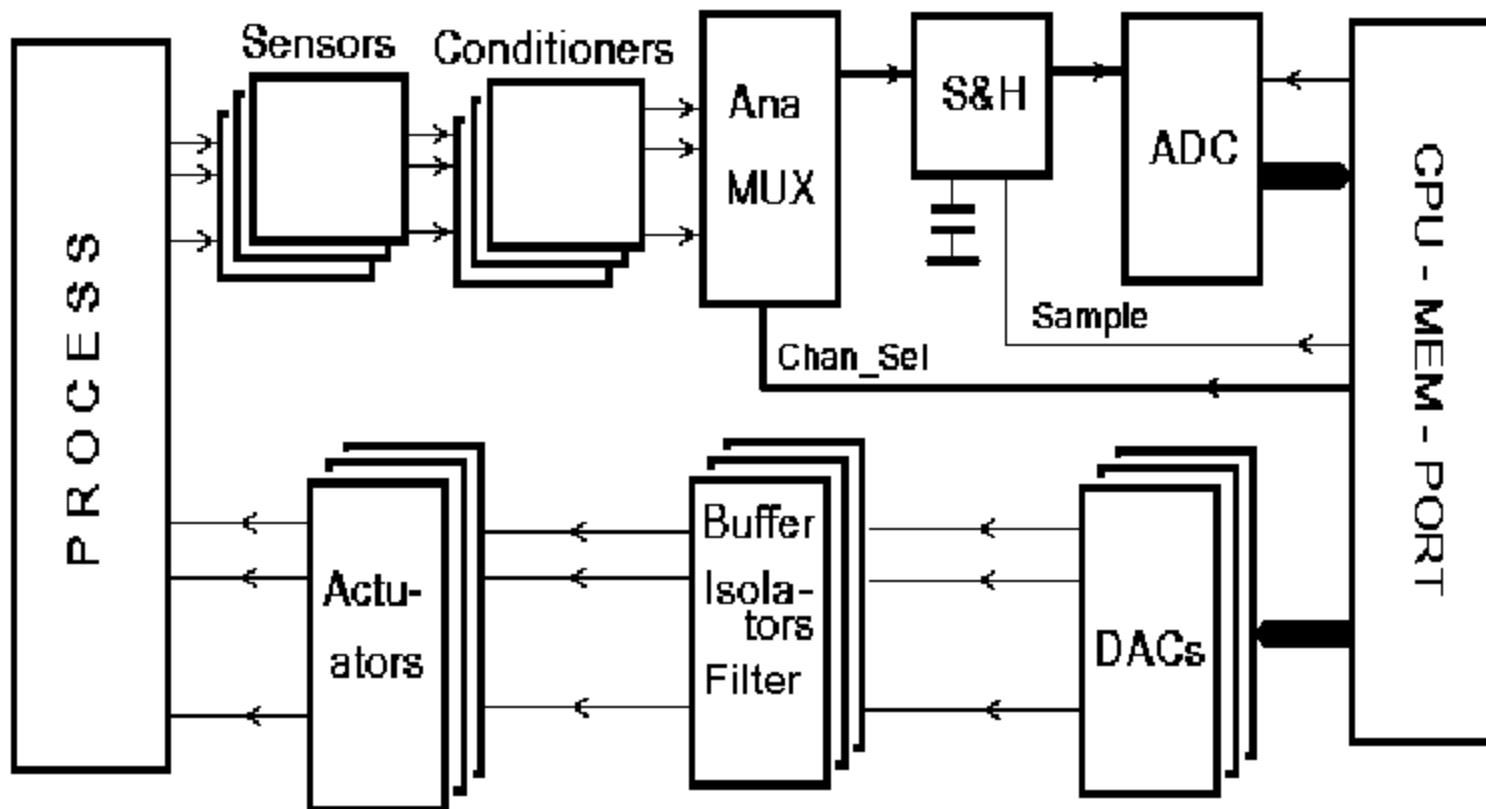
- Dùng để trích mẫu của t/h khi có xung sample (100s ns.. vài us) và giữ nguyên giá trị của t/h trong khoảng thời gian lâu hơn để ADC chuyển đổi được ổn định;
- Chỉ dùng trong các trường hợp tín hiệu biến thiên nhanh tương đối so với thời gian c/d của ADC;
- Nâng cao độ chính xác và tần số của tín hiệu.





# Mô hình ghép nối hệ đo lường-điều khiển số

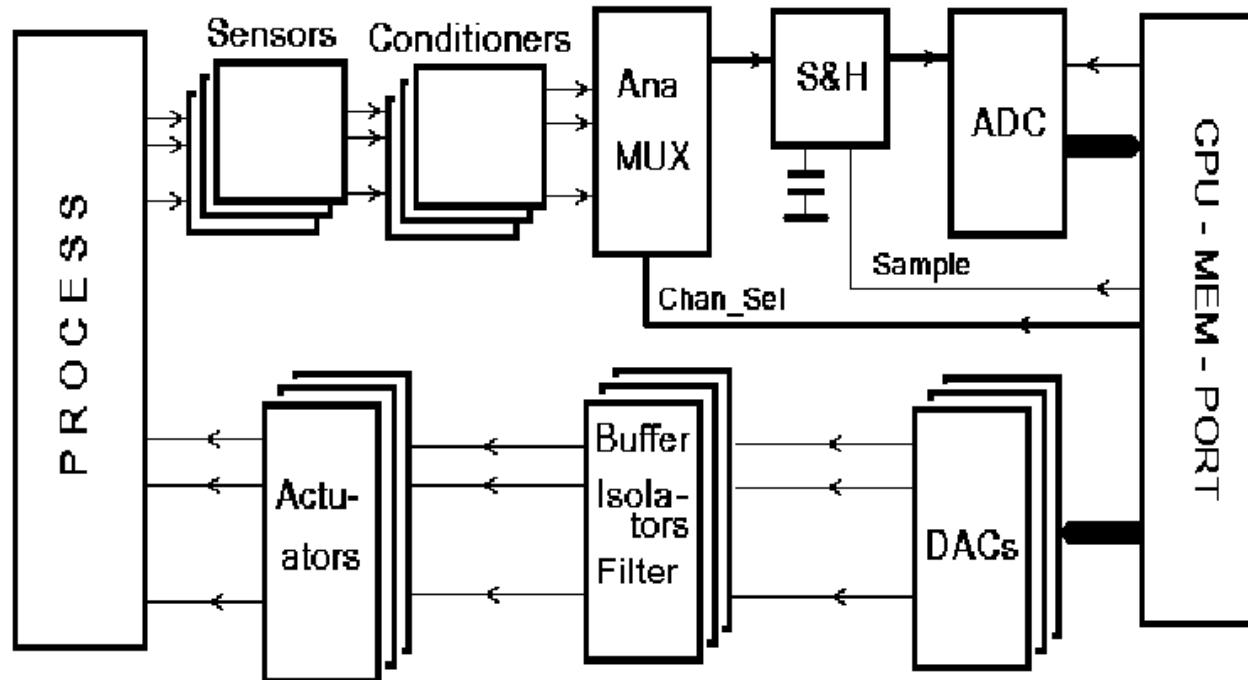
- ADC: analog to digital convertor:
  - Rời rạc hóa t/h về thời gian và số hóa t/h – lượng tử hóa
  - Có nhiều phương pháp/tốc độ/địa chỉ ứng dụng của chuyển đổi





# Mô hình ghép nối hệ đo lường-điều khiển số

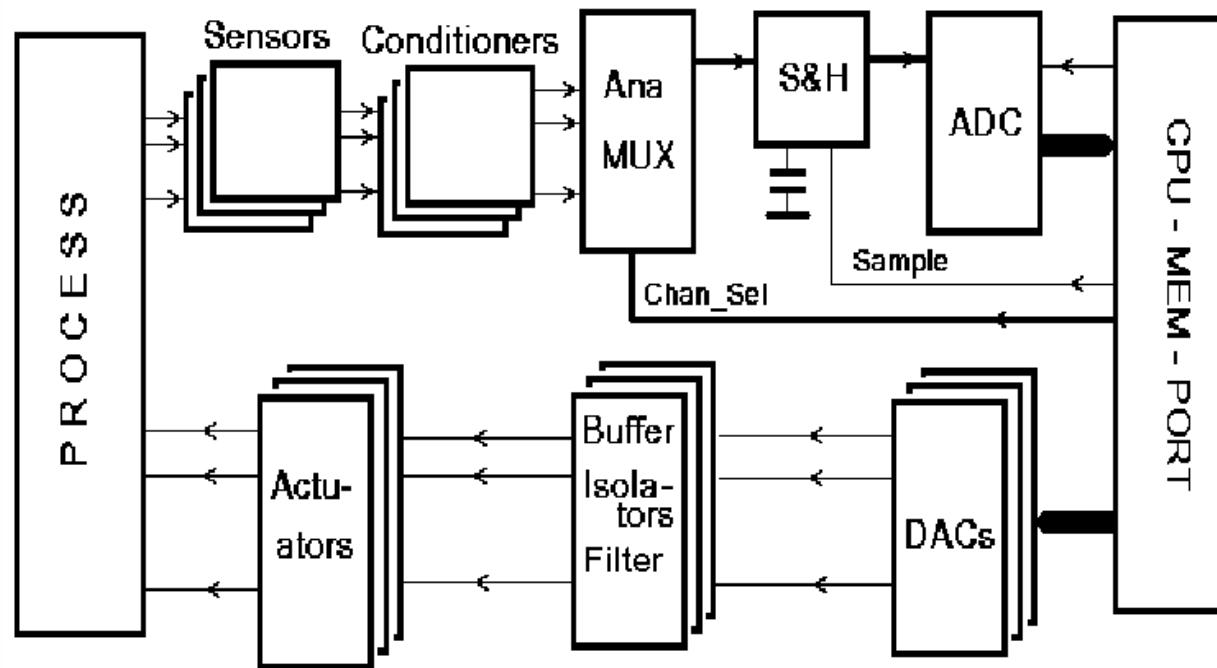
- Central system: hệ nhúng/MT:
  - CPU, mem, bus, IO port, CSDL, net;
  - thu thập và xử lý số đo.
- DAC: digital to analog convertor
  - Biến đổi tín hiệu số => liên tục về tg nhưng vẫn rời rạc về gt;





# Mô hình ghép nối hệ đo lường-điều khiển số

- Mạch điện tử analog:
  - Có nhiều kiểu chức năng tùy thuộc ứng dụng:
    - ✓ Lọc – tái tạo, tổng hợp âm thanh;
    - ✓ Khuếch đại để đến các cơ cấu chấp hành;
    - ✓ Cách ly quang học để ghép nối với các thiết bị công suất lớn (motor, breaker, ...)

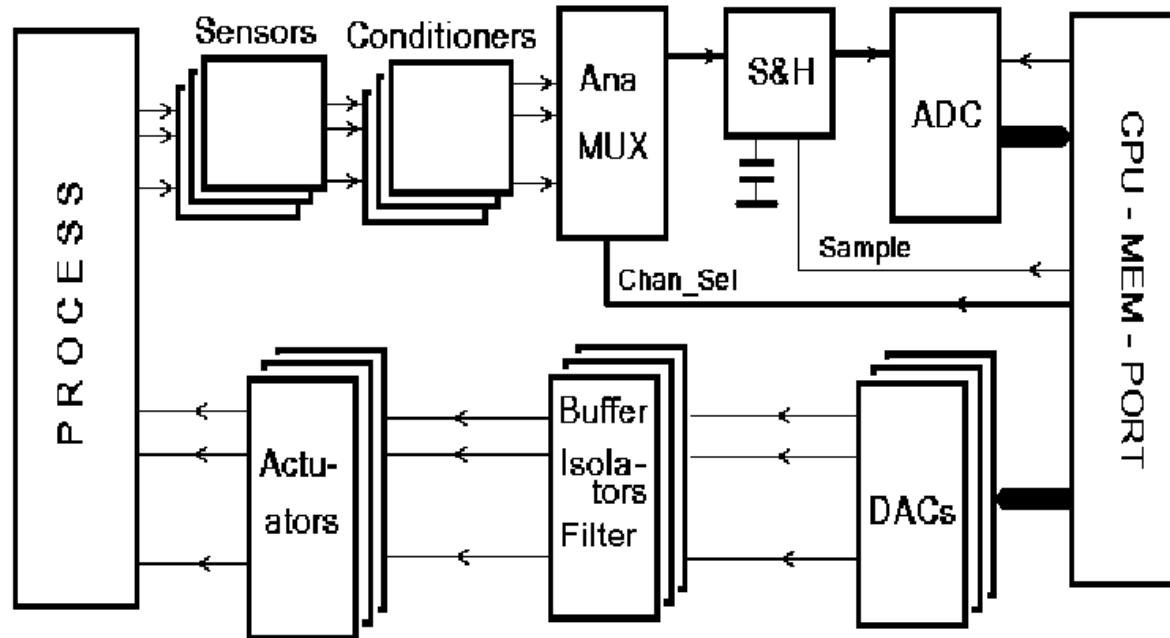




# Mô hình ghép nối hệ đo lường-điều khiển số

- Actuators: các cơ cấu chấp hành

- Là 1 lớp các thiết bị để tác động trở lại dây chuyền công nghệ;
- Cơ học: motor (3 phase Sync/Async, single phase, dc, step) như robot, printer's motor, FDC/HDC motors...
- Điều khiển dòng năng lượng điện: SCR (thyristor), Triac, Power MOSFET, IGBT...
- Điều khiển dòng chất lỏng/khí/gas: valves (percentage, ON/OFF valves)





# Một số hệ thống sử dụng ADC - DAC



Hệ thu thập số liệu – Điều khiển từ xa



## 2.2.1. Giao thức ghép nối

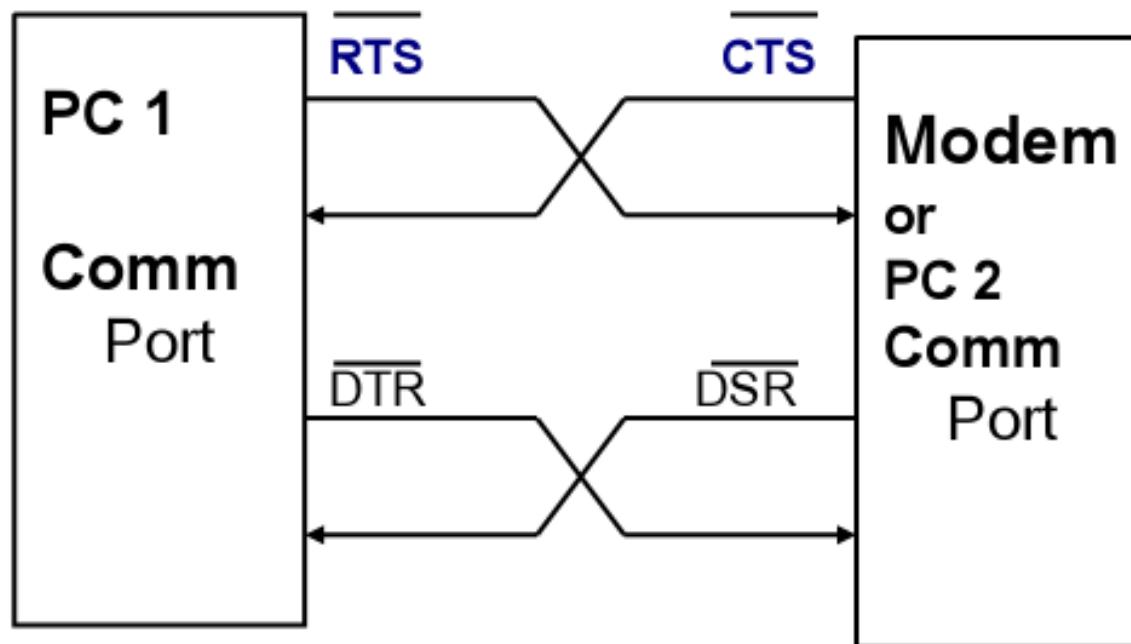
- Tín hiệu (Signals)

- Khi thiết kế ghép nối máy tính, cần đặc biệt chú ý tới tín hiệu theo các yêu cầu
  - ✓ Hơn một thiết bị -> cần bus/mạng
  - ✓ Dữ liệu xa hay gần, cần tốc độ truyền nhanh hay chậm -> nối tiếp hay song song
  - ✓ Các tín hiệu điều khiển (control signals), trạng thái (status signals) và bắt tay (handshaking signals)



## 2.2.1. Giao thức ghép nối

- Tín hiệu (signals)



Tín hiệu bắt tay khi ghép nối với máy tính qua cổng COM



## 2.2.1. Giao thức ghép nối

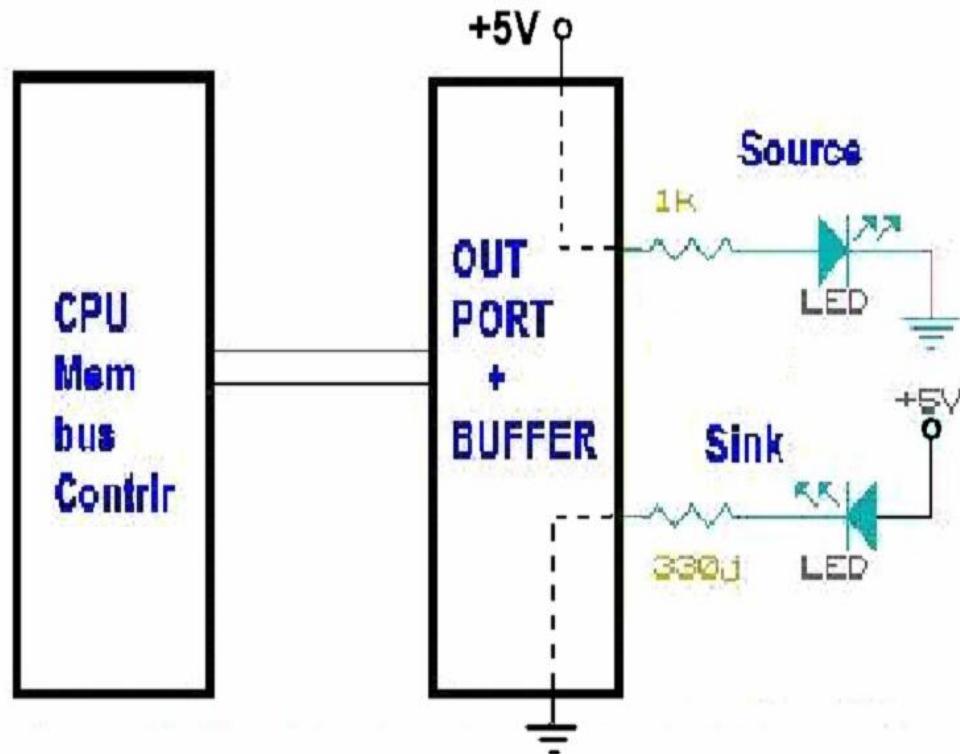
- Tín hiệu (signals)

- Phương pháp biến đổi tín hiệu: điều biên, điều tần, điều pha
- Mức điện áp
- Tín hiệu đơn cực (single end) hay vi sai (differential)
- Khả năng hot swap, hot pluggable
- Có cần cách ly không (isolated): cách ly quang học (Opto-Coupler)
- Khe cắm, cổng cắm, số đường tín hiệu...



## 2.2.1. Giao thức ghép nối

- **Dòng cung cấp (Source current)**
  - ✓ Cường độ dòng điện tối đa mà chân vào ra có thể cung cấp khi nó được đưa lên mức điện áp cao.
  - ✓ Nếu tải sử dụng quá dòng này thì sẽ gây sụt áp.
- **Dòng hấp thụ (Sink current)**
  - ✓ Là dòng tối đa mà chân vào ra có thể hấp thụ khi nó bị kéo xuống mức điện áp thấp.
  - ✓ Nếu dòng thực tế lớn hơn dòng hấp thụ -> hỏng thiết bị



Sink & Source connection

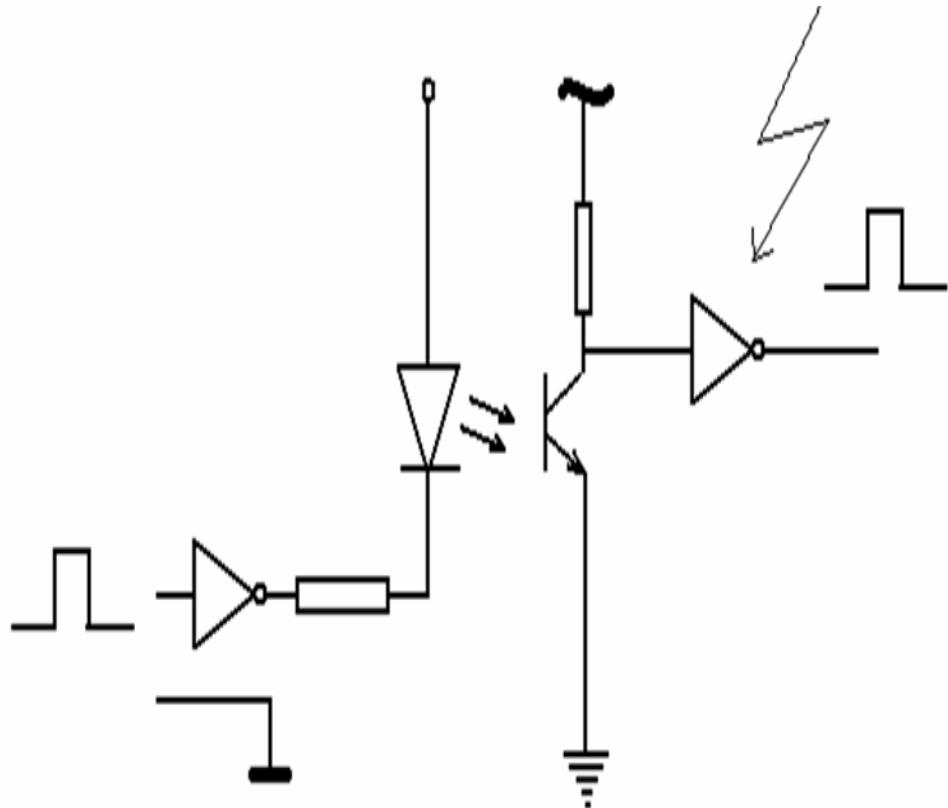


Sử dụng bộ đệm



## 2.2.1. Giao thức ghép nối

- Cách ly
  - Cách ly giữa mạch trung tâm và mạch ngoại vi
  - Khi có sự biến đổi đột ngột về điện áp tại mạch ngoại vi (VD: sét đánh) -> không ảnh hưởng tới mạch trung tâm



Cách ly quang học  
(Opto-Coupler)



## 2.2.1. Giao thức ghép nối

- Khe cắm: ISA, PCI ...
- Cổng: COM (DB9, DB25),  
LPT ...
- Cáp:
  - Đồng trục (coaxial)
  - Xoắn (twisted)
  - Thường (normal)
  - Quang (optical)





## 2.2.1. Giao thức ghép nối

- Khuôn dạng dữ liệu (Data Format)

- Nhu cầu đóng gói dữ liệu
  - ✓ Khi cần truyền nhiều byte dữ liệu (USB, TCP-IP)
  - ✓ Khi cần truyền không đồng bộ (RS232, RS485...)
- Một số khuôn dạng dữ liệu

SYNC	PID	DATA	CRC	EOP
8 bits (low/full)/32 bits (high)	8 bits	up to 8 bytes (low)/1023 bytes (full)/1024 bytes (high)	16 bits	n/a

Gói tin theo chuẩn USB



## 2.2.1. Giao thức ghép nối

- Tốc độ trao đổi thông tin (Rate)
  - Phụ thuộc:
    - ✓ Khoảng cách
    - ✓ Môi trường truyền
    - ✓ Cách điều chế tín hiệu...
  - Tốc độ của một số chuẩn giao thức ghép nối
    - ✓ LPT:
      - SPP mode: 50->100kbps
      - ECP mode: 2->4Mbps
    - ✓ LAN/Ethernet 10/100/1000 Mbps
    - ✓ RS232: 1200/2400/4800/9600...bps



## 2.2.1. Giao thức ghép nối

- Kiểm tra, sửa lỗi, nâng cao độ tin cậy
  - Khi trao đổi thông tin thường có lỗi, đặc biệt khi truyền xa hoặc chuyển đổi tín hiệu.
  - Các phương pháp hỗ trợ kiểm tra và sửa lỗi (cả phần cứng và phần mềm)
    - ✓ Kiểm tra chẵn lẻ
    - ✓ Mã CRC
    - ✓ Redundancy: truyền dư thừa, trao đổi dữ liệu nhiều hơn một lần để so sánh



## 2.2.1. Giao thức ghép nối

- Bộ lệnh và thông tin trả về (Command set & Response)
  - Cần thiết khi ghép nối với các thiết bị có nhiều tham số và chế độ hoạt động (mouse, Printer, Modem...)
  - Bộ lệnh (Command set): tập hợp các yêu cầu từ hệ trung tâm (CS) gửi cho thiết bị ngoại vi
    - ✓ VD: Tập lệnh AT, Open-AT...
  - Thông tin trả về: phản hồi và trạng thái gửi từ thiết bị ngoại vi



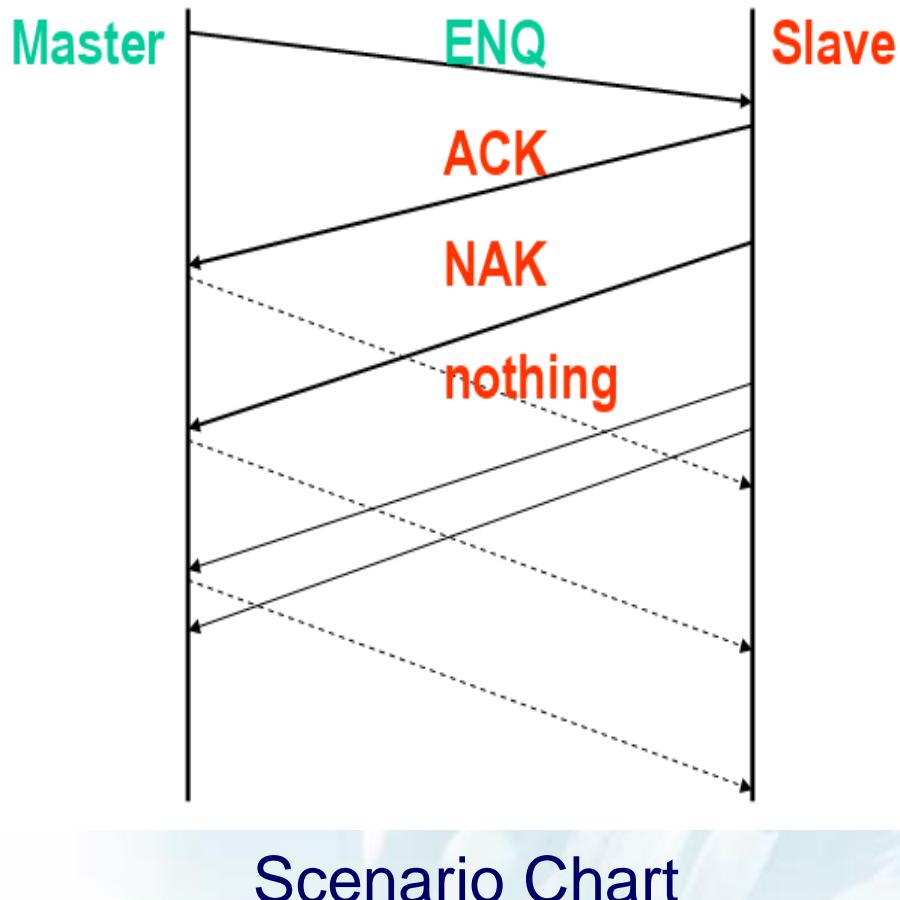
## 2.2.1. Giao thức ghép nối

- 1 command/response thường có cấu trúc
  - Mã bắt đầu ký tự riêng @, #, \$...
  - Mã lệnh
  - Tham số lệnh
  - Mã check sum
  - Mã kết thúc, ký tự riêng
  - Có thêm các mã đối thoại, sử dụng các ký tự điều khiển của ASCII như ENQ, ACK, NACK, Bell, OK, ERR, BUSY...



## 2.2.1. Giao thức ghép nối

- Kịch bản đối thoại (Scenario)
  - Liệt kê các trường hợp có thể rồi áp các phép xử lý tương ứng để đảm bảo việc ghép nối: không mất tin, không thừa tin, không quẩn, treo...
  - Thường xây dựng theo
    - ✓ Step list
    - ✓ Chart





## 2.2.2. Chuẩn RS232

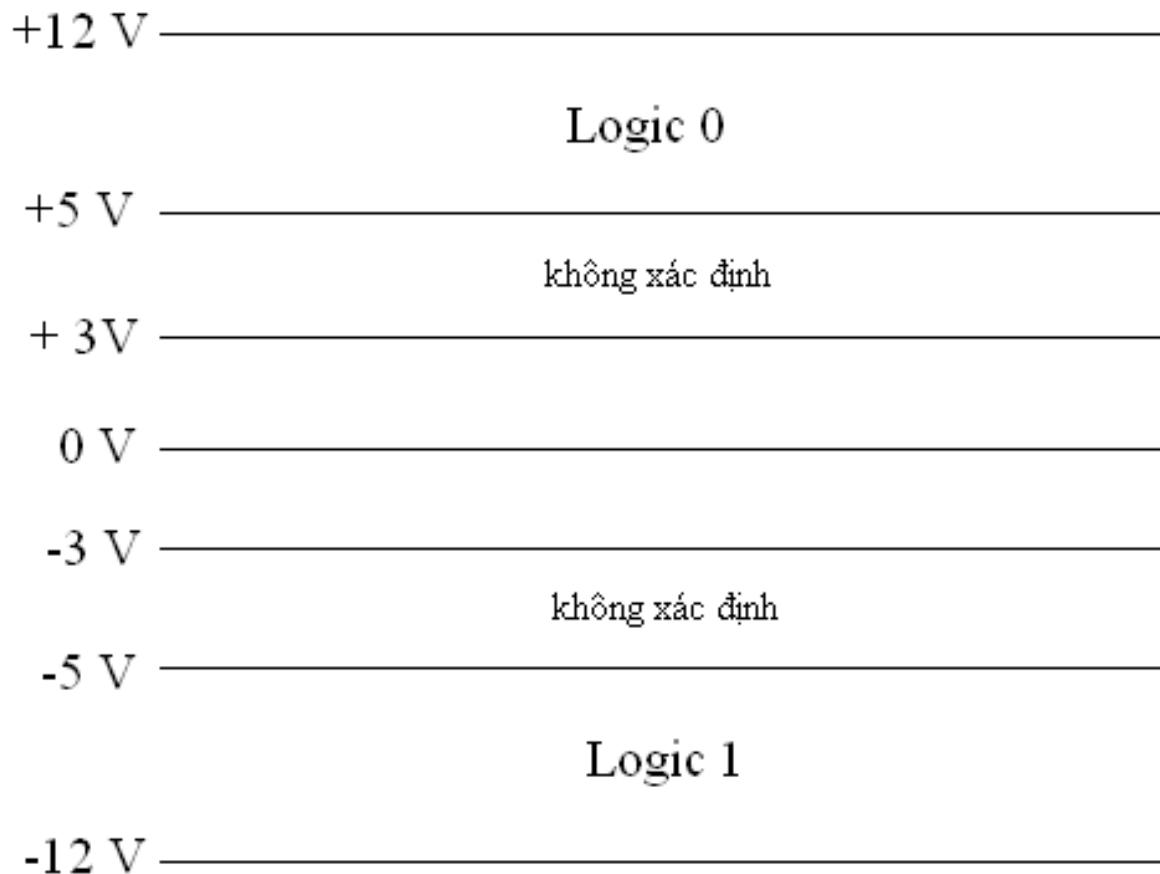
- Mức điện áp đường truyền
- Chuẩn đầu nối trên máy tính PC
- Khuôn dạng khung truyền
- Tốc độ truyền
- Kích bản truyền





# Chuẩn RS232

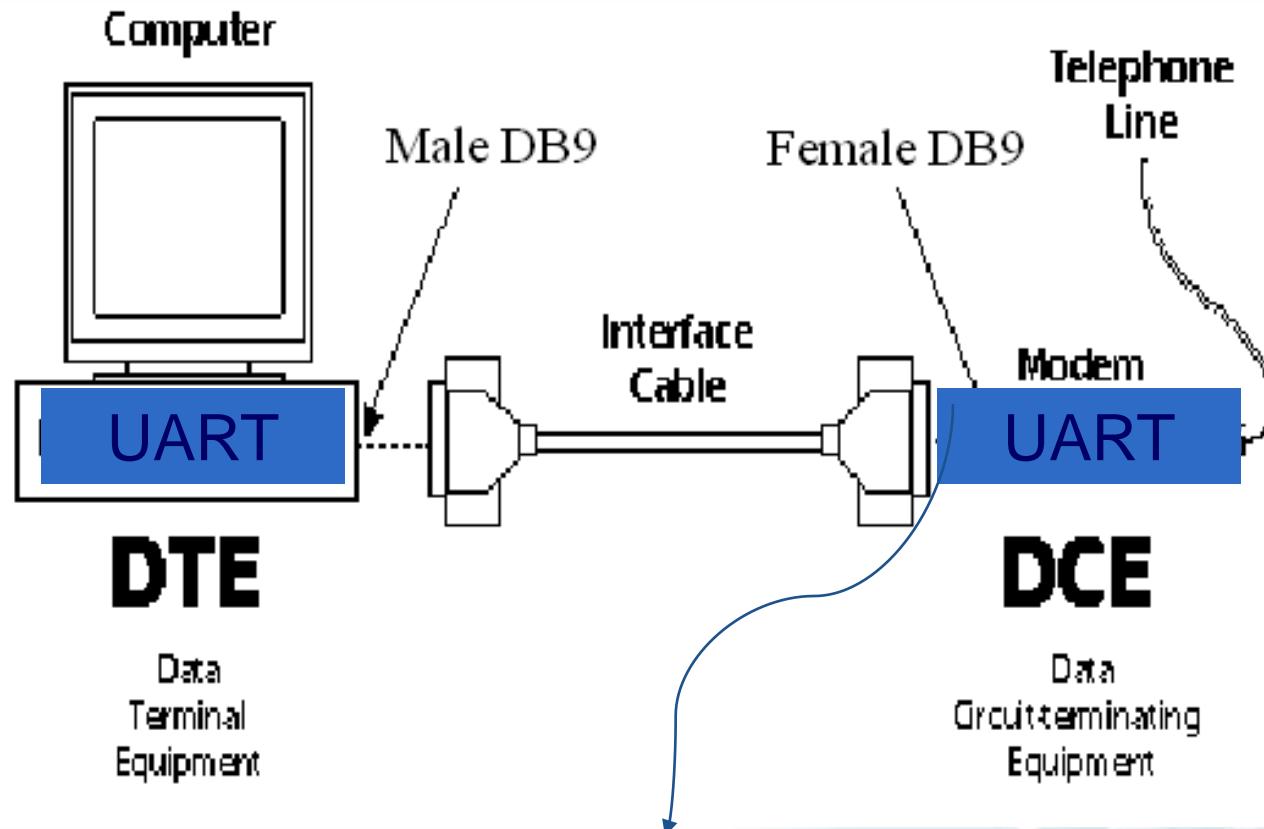
- Mức điện áp đường truyền (Chuẩn RS-232C)





# Chuẩn RS232

- Chuẩn đầu nối trên PC



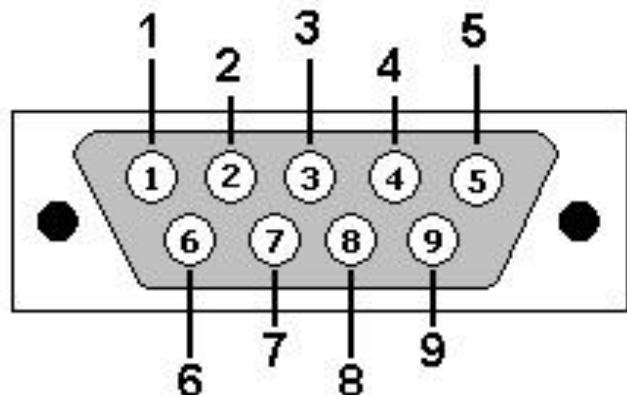
UART (Universal Asynchronous receiver/transmitter)



# Chuẩn RS232

## ▪ Chuẩn đầu nối trên PC

- Chân 1 (DCD-Data Carrier Detect): phát hiện tín hiệu mang dữ liệu
- Chân 2 (RxD-Receive Data): nhận dữ liệu
- Chân 3 (TxD-Transmit Data): truyền dữ liệu
- Chân 4 (DTR-Data Terminal Ready): đầu cuối dữ liệu sẵn sàng
- Chân 5 (Signal Ground): đất của tín hiệu
- Chân 6 (DSR-Data Set Ready): dữ liệu sẵn sàng
- Chân 7 (RTS-Request To Send): yêu cầu gửi
- Chân 8 (CTS-Clear To Send): Xóa để gửi
- Chân 9 (RI-Ring Indicate): báo chuông





# Chuẩn RS-232:

Chân số		Tên	Kí hiệu	Hướng DTE - DCE	Chức năng
D-25	D-9				
1	-	Frame Ground	FG	-	Thường được nối với vỏ bọc kim loại của cáp dẫn hoặc đất
2	3	Transmit Data	TxD	→	Số liệu được phát từ DTE tới DCE qua đường TxD
3	2	Receive Data	RxD	←	Số liệu được thu từ DCE vào DTE
4	7	Request To Send	RTS	→	DTE đặt đường này ở mức tích cực khi nó sẵn sàng phát số liệu
5	8	Clear To Send	CTS	←	DCE đặt đường này ở mức tích cực để thông tin cho DTE rằng nó sẵn sàng nhận số liệu



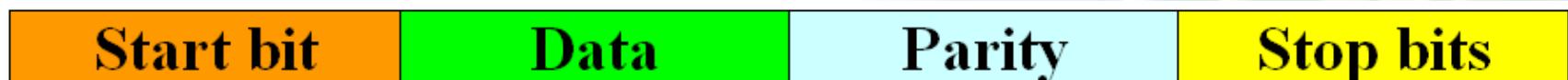
# Chuẩn RS-232

Chân số		Tên	Kí hiệu	Hướng DTE - DCE	Chức năng
D-25	D-9				
6	6	Data Set Ready	DSR	←	Chức năng tương tự như CTS nhưng được kích hoạt bởi DTE khi nó sẵn sàng nhận số liệu
20	4	Data Terminal Ready	DTR	→	Chức năng tương tự như RTS nhưng được kích hoạt bởi DCE khi nó muốn phát số liệu
8	1	Data Carrier Detect	DCD	←	DCE đặt đường này ở mức tích cực để báo cho DTE biết là đã thiết lập được liên kết với DCE từ xa (nhận được sóng mang từ bên DCE đối tác)
22	9	Ring Indicator	RI	←	DCE (loại lắp ngoài) báo với DTE có một cuộc gọi từ xa vừa gọi đến
7	5	Signal Ground	SG	-	GND



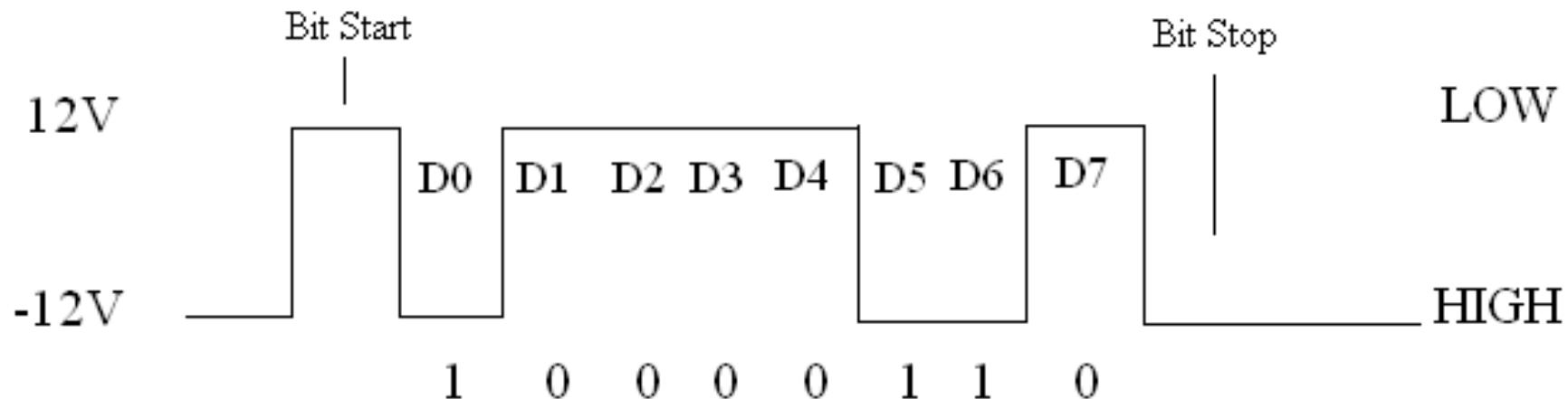
## ▪ Khuôn dạng khung truyền

- PC truyền nhận dữ liệu qua cổng nối tiếp RS-232 thực hiện theo kiểu không đồng bộ (Asynchronous)
- Khung truyền gồm 4 thành phần
  - ✓ 1 Start bit (Mức logic 0): bắt đầu một gói tin, đồng bộ xung nhịp clock giữa DTE và DCE
  - ✓ Data (5,6,7,8 bit): dữ liệu cần truyền
  - ✓ 1 parity bit (chẵn (even), lẻ (odd), mark, space): bit cho phép kiểm tra lỗi
  - ✓ Stop bit (1 hoặc 2 bit): kết thúc một gói tin





- Khuôn dạng khung truyền

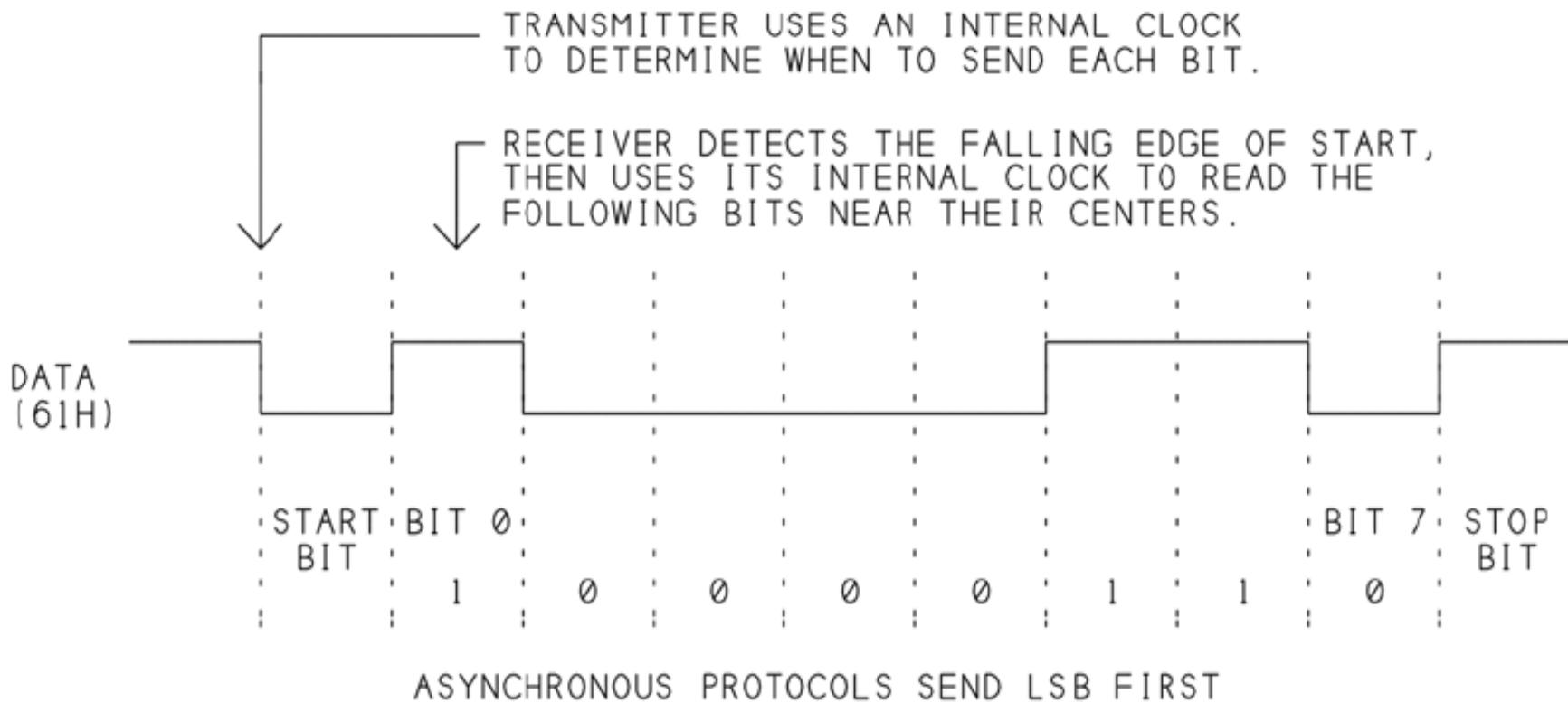


Khung truyền cho dữ liệu 61h theo khung: (8, N, 1)

- 8 bit dữ liệu
- 1 bit stop
- Không có bit parity



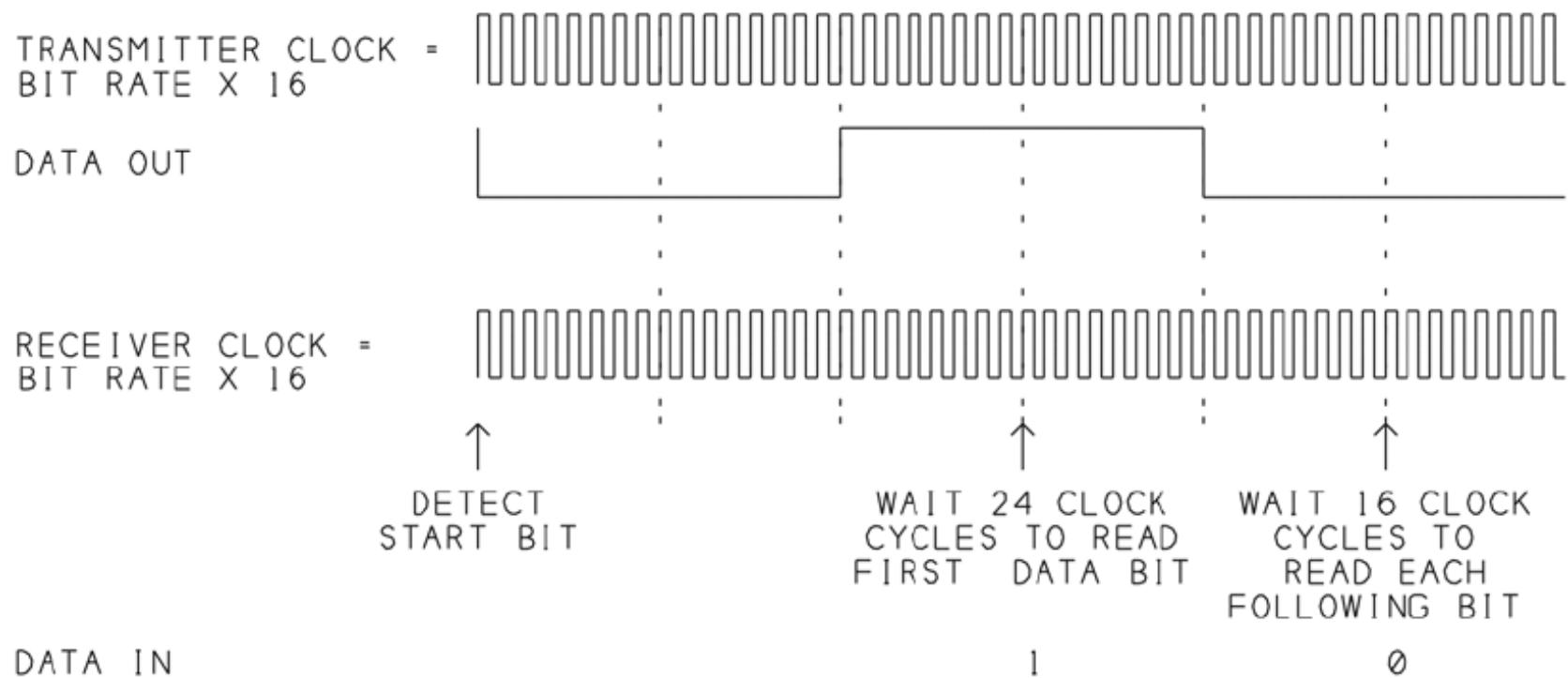
## ■ Khuôn dạng khung truyền



Xác định thời điểm truyền và nhận dữ liệu



- Khuôn dạng khung truyền



Quá trình truyền và nhận dữ liệu dưới tác động của các xung nhịp đồng hồ tại bên truyền và bên nhận

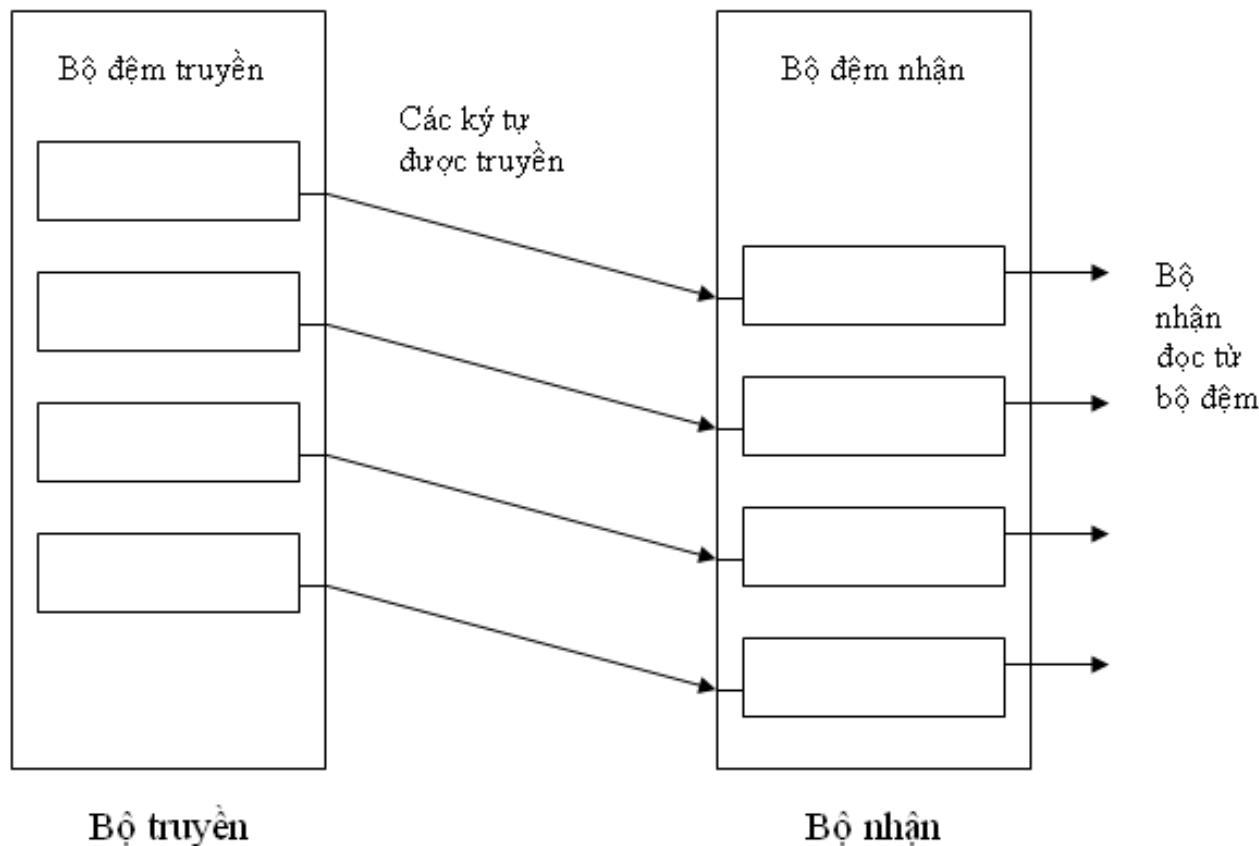


## ■ Tốc độ truyền

- Tính bằng đơn vị bit/giây: bps (bit per second)
- Thuật ngữ khác: baud
  - ✓ Đơn vị đo dùng cho modem
  - ✓ Số lần thay đổi tín hiệu trong một giây
  - ✓ Đối với modem, mỗi lần thay đổi tín hiệu, có thể truyền được nhiều bit : **tốc độ baud <= tốc độ bit**
- Tốc độ tối đa = Tần số xung nhịp clock / hằng số
- VD: trong máy PC, tần số 1.8432MHz, hằng số =16 -> tốc độ tối đa: 115,200 bps
- Bên trong UART hỗ trợ các thanh ghi cho phép xác định các tốc độ làm việc khác, vd: 1200, 2400, 4800, 9600, 19200, 38400... bps



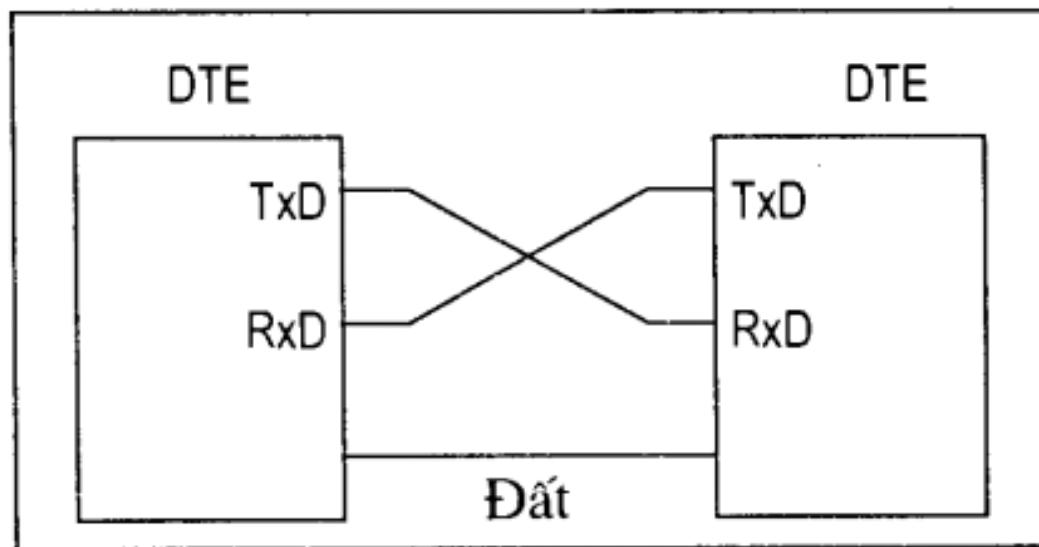
- Kịch bản truyền
  - Quá trình truyền và nhận các ký tự





## ▪ Kịch bản truyền

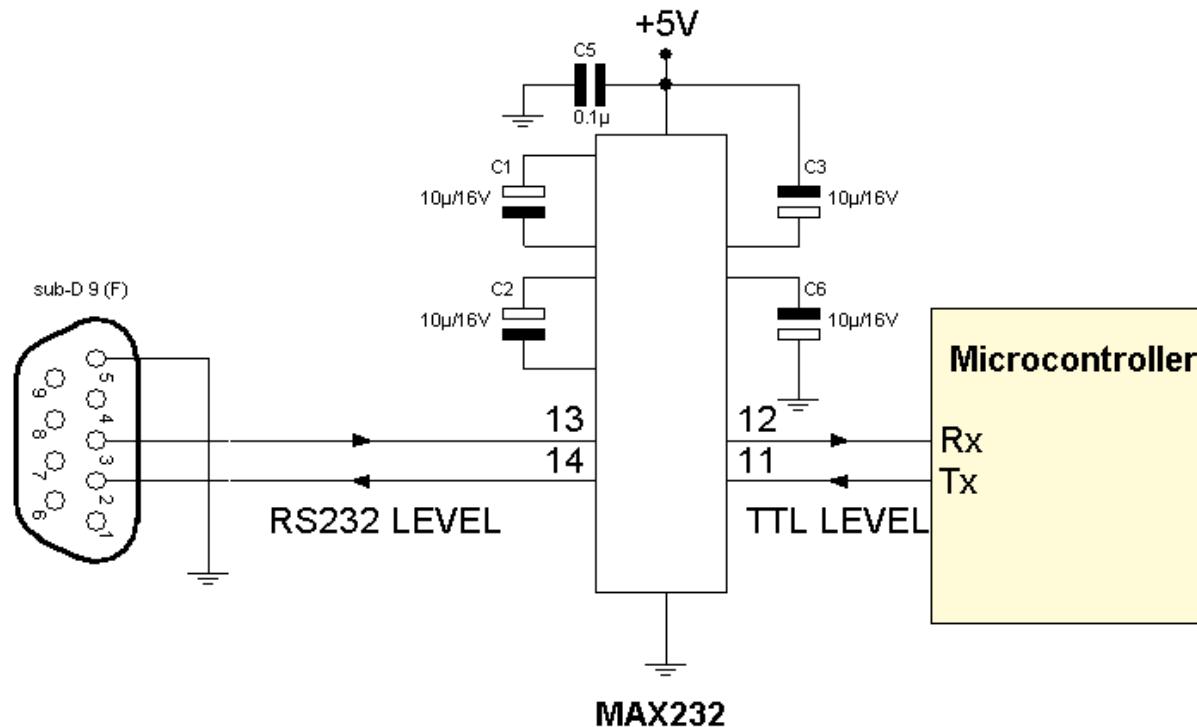
- Không có bắt tay (none-handshaking): máy thu có khả năng đọc các ký tự thu trước khi máy phát truyền ký tự tiếp theo



Kết nối không cần bắt tay giữa hai thiết bị  
(cùng mức điện áp)



## ▪ Kịch bản truyền



Ghép nối không bắt tay giữa hai thiết bị  
(Khác nhau về mức điện áp)

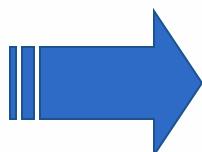


- Kịch bản truyền

- Có bắt tay (handshaking):

- ✓ Máy thu nhận các ký tự và lưu vào một vùng nhớ gọi là bộ đệm thu (receive buffer)
    - ✓ Nếu ký tự tại bộ đệm thu không được đọc kịp trước khi ký tự khác được truyền tới -> có thể xảy ra hiện tượng các ký tự hiện tại bị ghi đè bởi các ký tự mới

Cần tín hiệu điều khiển, buộc máy phát ngừng phát cho đến khi máy thu đọc xong các ký tự đang nằm trong bộ đệm thu





- Kịch bản truyền

- Có bắt tay (handshaking)

- ✓ Bắt tay bằng phần cứng

- Sử dụng các tín hiệu bắt tay RTS, CTS, DTR, DSR

- ✓ Bắt tay bằng phần mềm

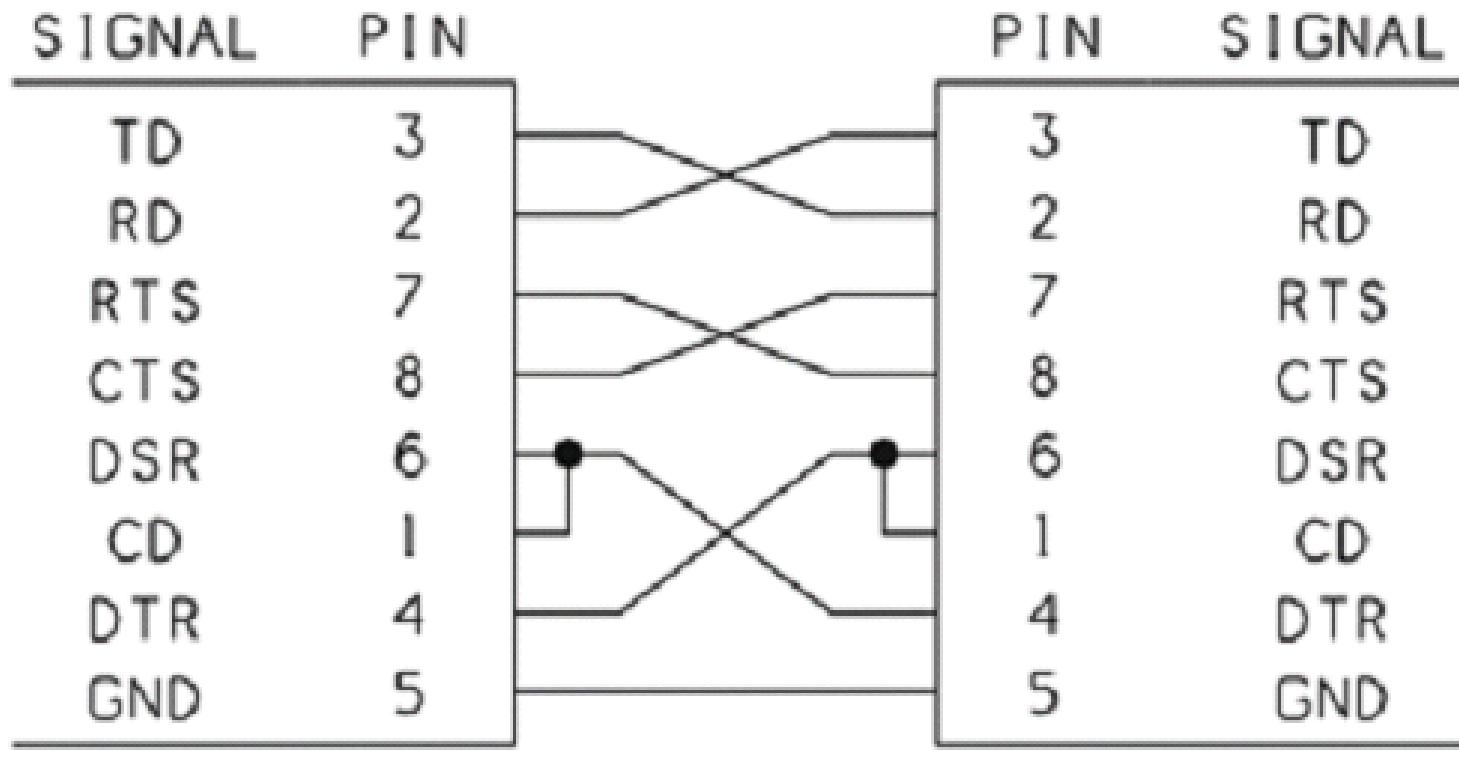
- Sử dụng các gói tin đặc biệt truyền các ký tự Xon, Xoff.

- ✓ Bắt tay kết hợp cả phần cứng và phần mềm



## ▪ Kịch bản truyền

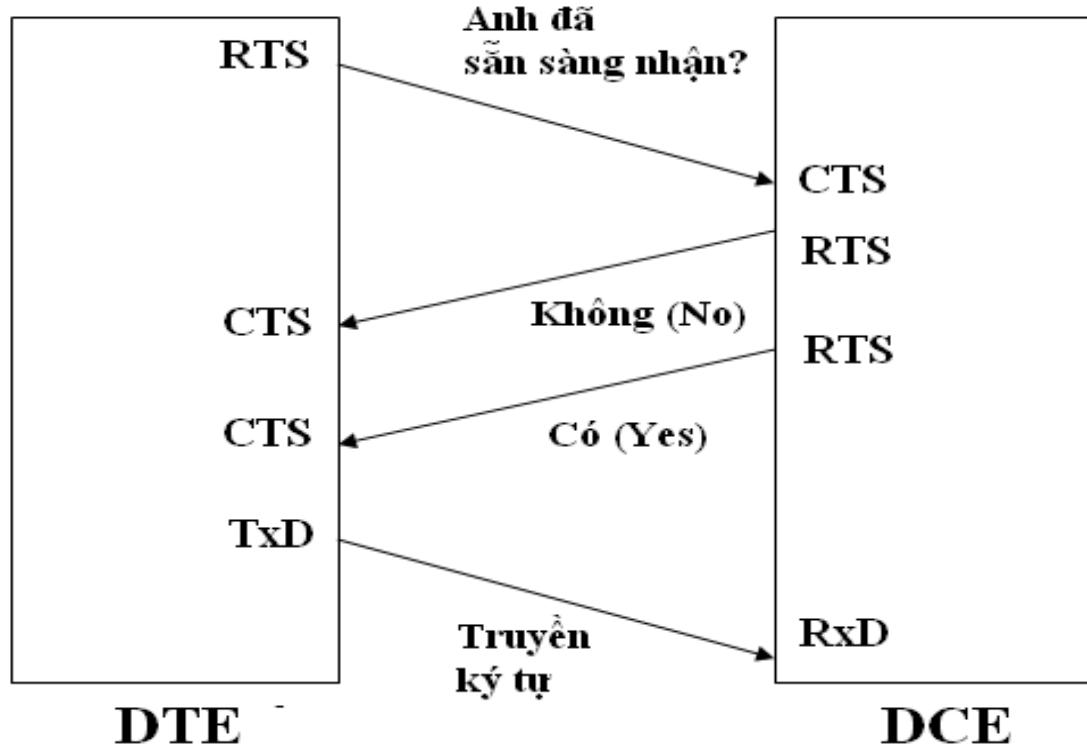
- Bắt tay bằng phần cứng



Sơ đồ đấu nối tín hiệu bắt tay bằng phần cứng



- **Kịch bản truyền**
  - Bắt tay phần cứng

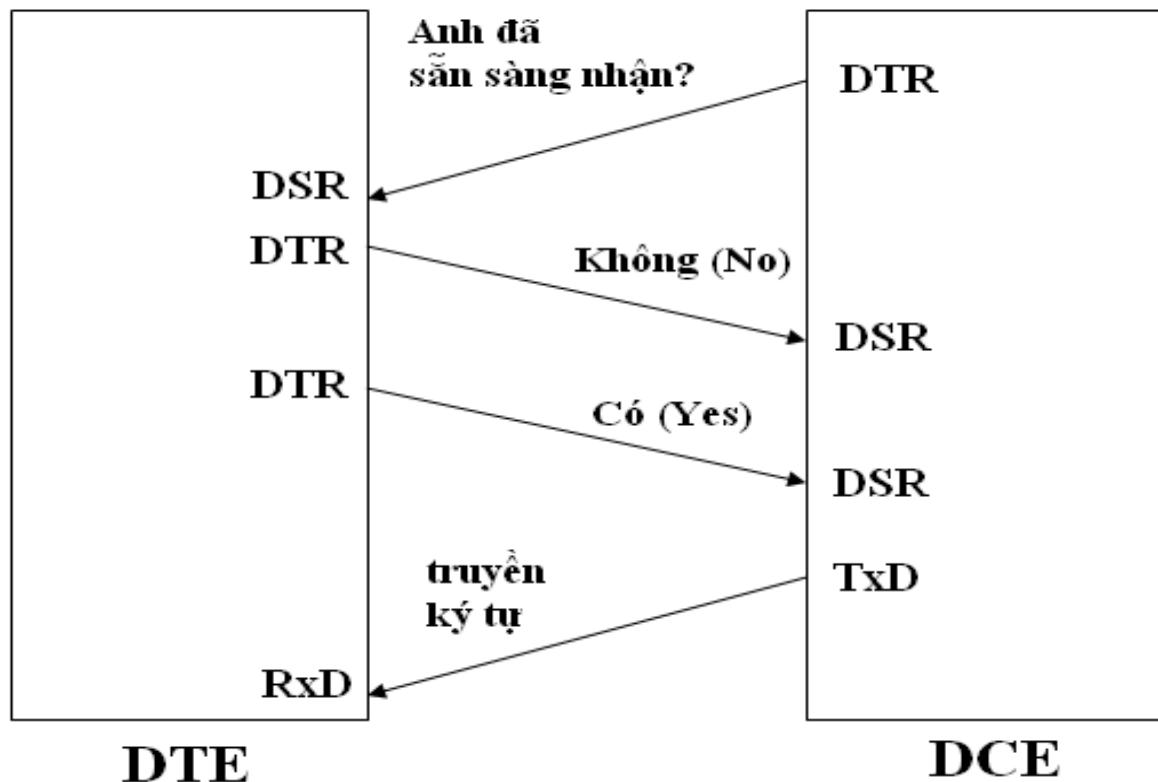


Các đường tín hiệu bắt tay được sử dụng khi DTE truyền dữ liệu



## ▪ Kịch bản truyền

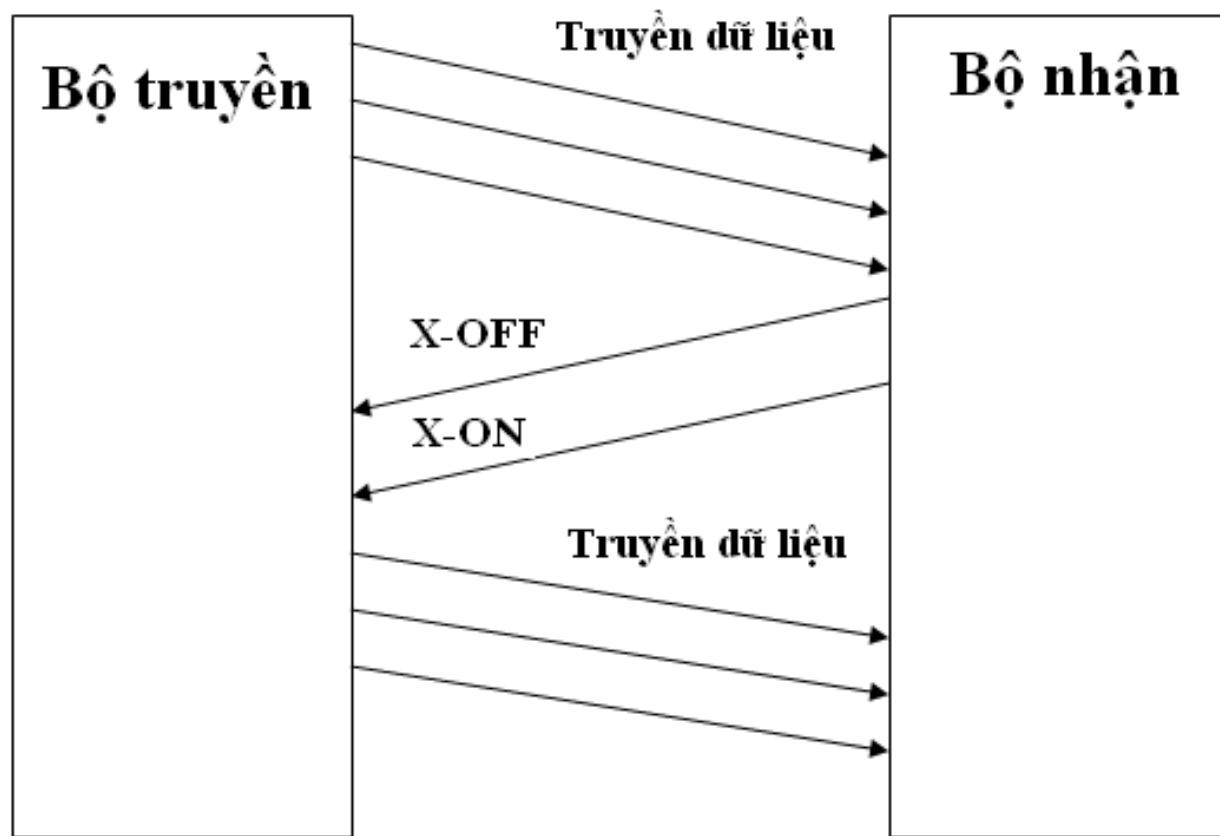
- Bắt tay bằng phần cứng



Các đường tín hiệu bắt tay được sử dụng khi DTE nhận dữ liệu



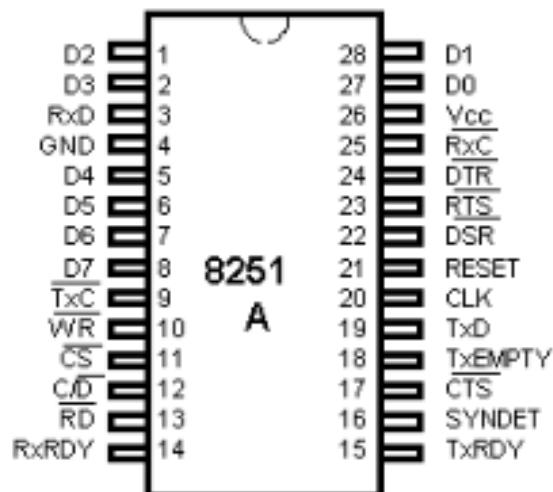
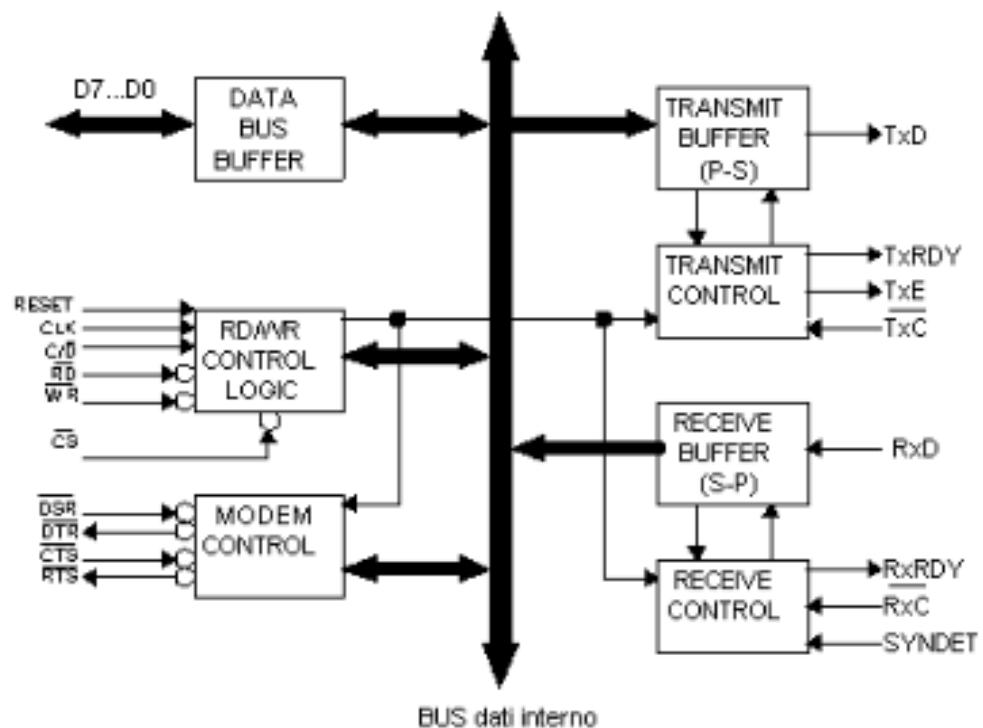
- Bắt tay bằng phần mềm
  - Sử dụng hai ký tự ASCII: X-ON (Ctrl-S) và X-OFF (Ctrl-Q)





# Mạch USART 8251A

- Có thể dùng cho cả hai chế độ truyền đồng bộ và dị bộ





# Tổ chức chân 8251A

- CLK: chân nối tới xung đồng hồ của hệ thống
- TxRDY: tín hiệu báo bộ đệm giữ rỗng (sẵn sàng nhận dữ liệu mới từ CPU)
- RxRDY: tín hiệu báo bộ đệm thu đầy (có ký tự nằm chờ CPU đọc vào)
- TxEMPTY: báo cả bộ đệm giữ và thu đều rỗng
- C/D: CPU chọn thanh ghi lệnh hay thanh ghi dữ liệu
- RxC và TxC: xung đồng hồ cung cấp cho thanh ghi dịch của phần thu và phát



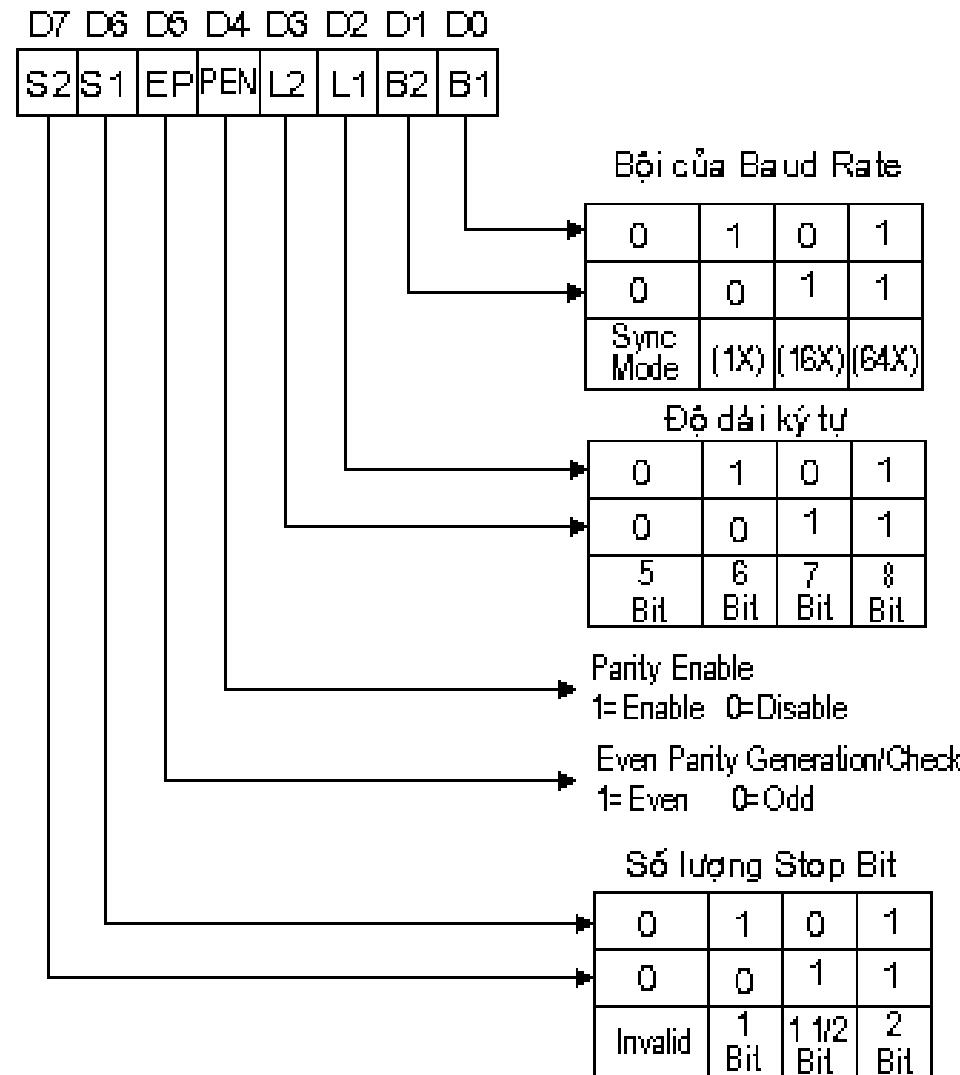
# Các thanh ghi bên trong 8251A

- Thanh ghi đệm dữ liệu thu
- Thanh ghi đệm dữ liệu phát
- Thanh ghi trạng thái
- Thanh ghi điều khiển (thanh ghi lệnh và thanh ghi chế độ)
- Sử dụng tín hiệu tại chân C/D, chân WR và RD để chọn thanh ghi làm việc

A0	RD	WR	Chọn ra
0	0	1	Thanh ghi đệm thu
0	1	0	Thanh ghi đệm phát
1	0	1	Thanh ghi trạng thái
1	1	0	Thanh ghi điều khiển

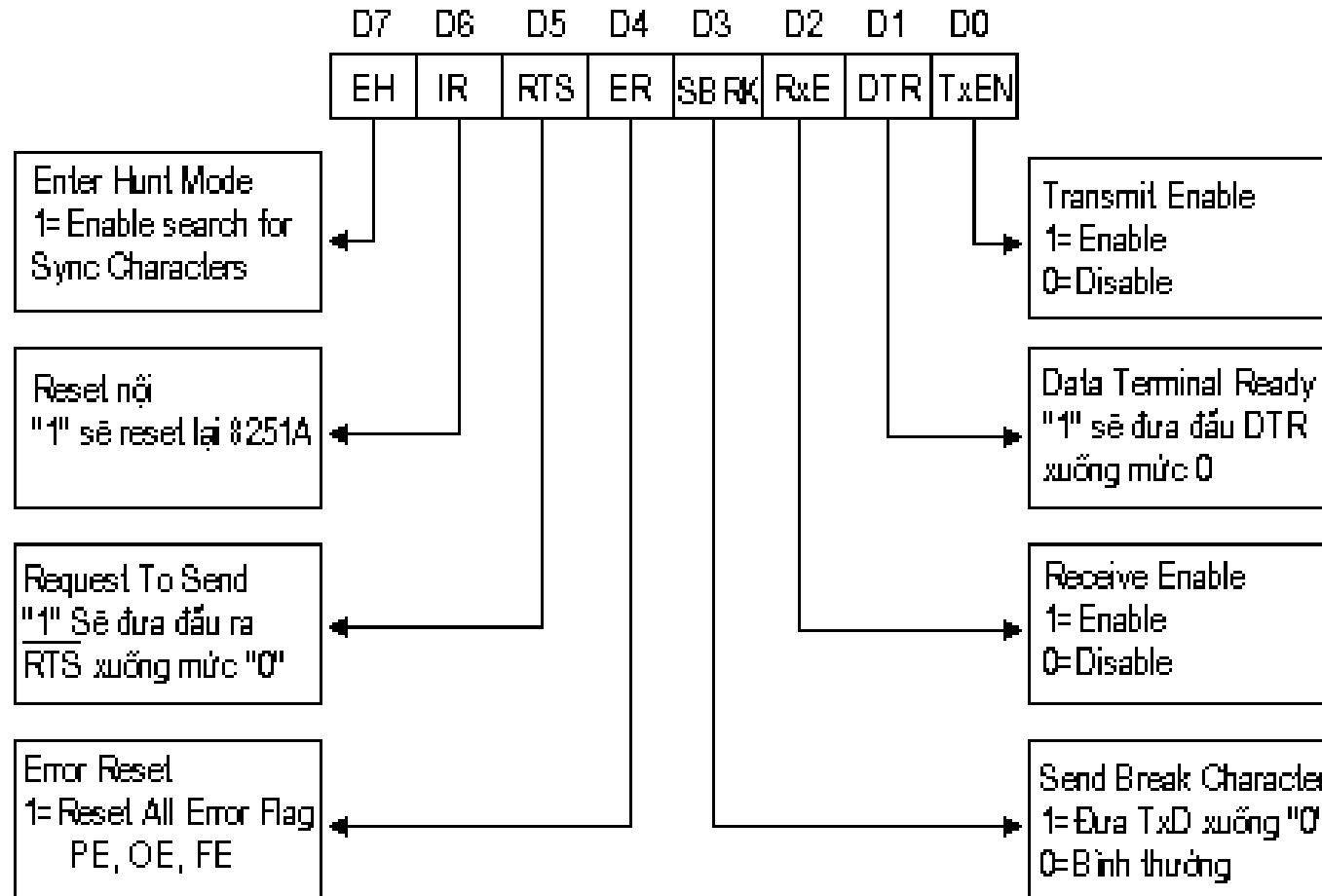


# Thanh ghi chế độ của 8251A



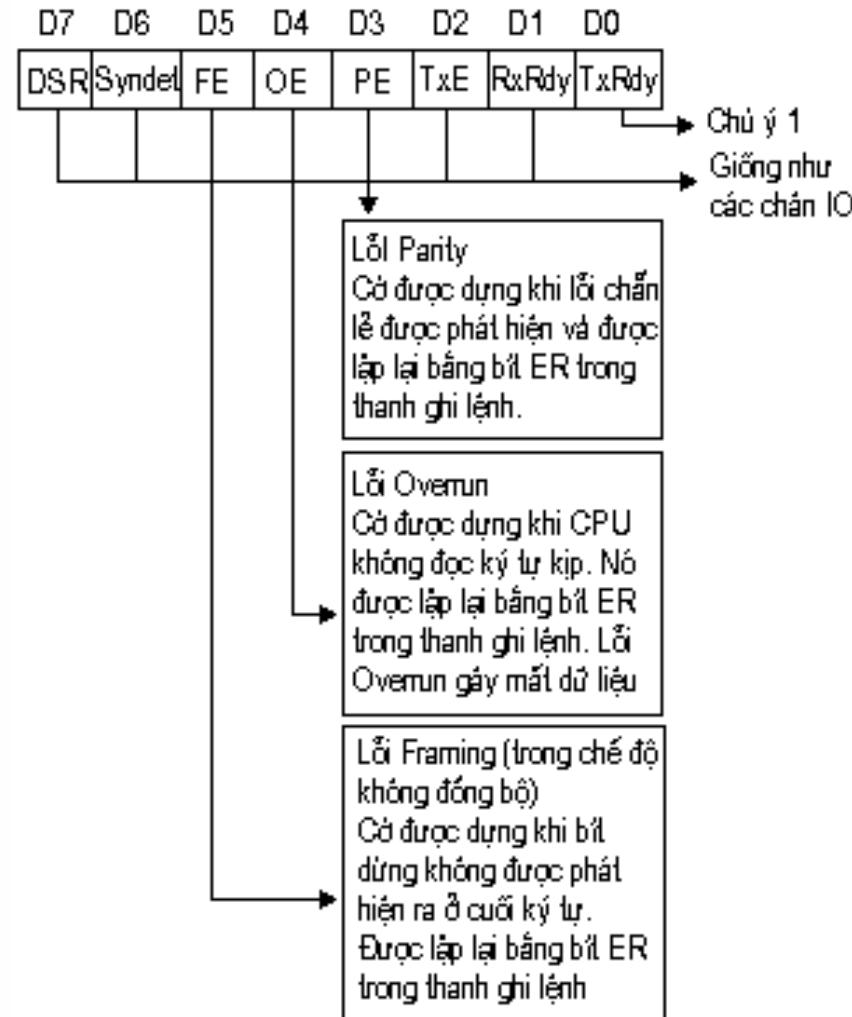


# Thanh ghi lệnh của 8251A



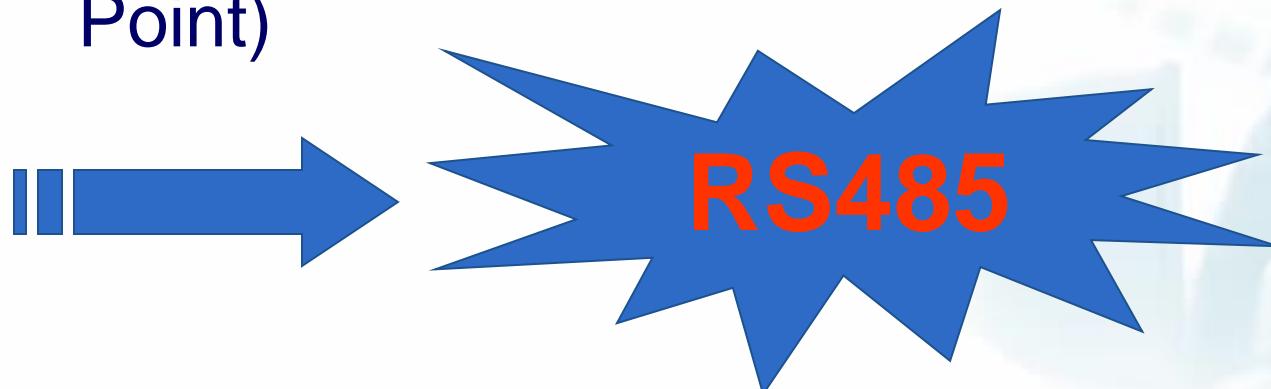


# Thanh ghi trạng thái của 8251A





- Ưu điểm
  - Đơn giản
  - Vẫn còn khá nhiều thiết bị được ghép nối qua chuẩn này
- Nhược điểm
  - Khoảng cách truyền tương đối ngắn ~ 15m
  - Chỉ cho phép kết nối điểm-tới-điểm (Point-to-Point)





- Một số đặc điểm của chuẩn RS485
  - Tín hiệu vi sai (differential signal) -> chống nhiễu, cho phép truyền xa hơn ~ 10km
  - Cho phép ghép nối nhiều thiết bị (32 thiết bị) -> mô hình mạng
  - Lập trình hoàn toàn tương tự chuẩn RS232 -> dễ dàng thực thi



# Chuẩn RS232

- Một số ứng dụng



AD-4329

(Đầu cân điện tử)



CPT-711

(Máy đọc  
mã vạch di  
động)



Routing switch



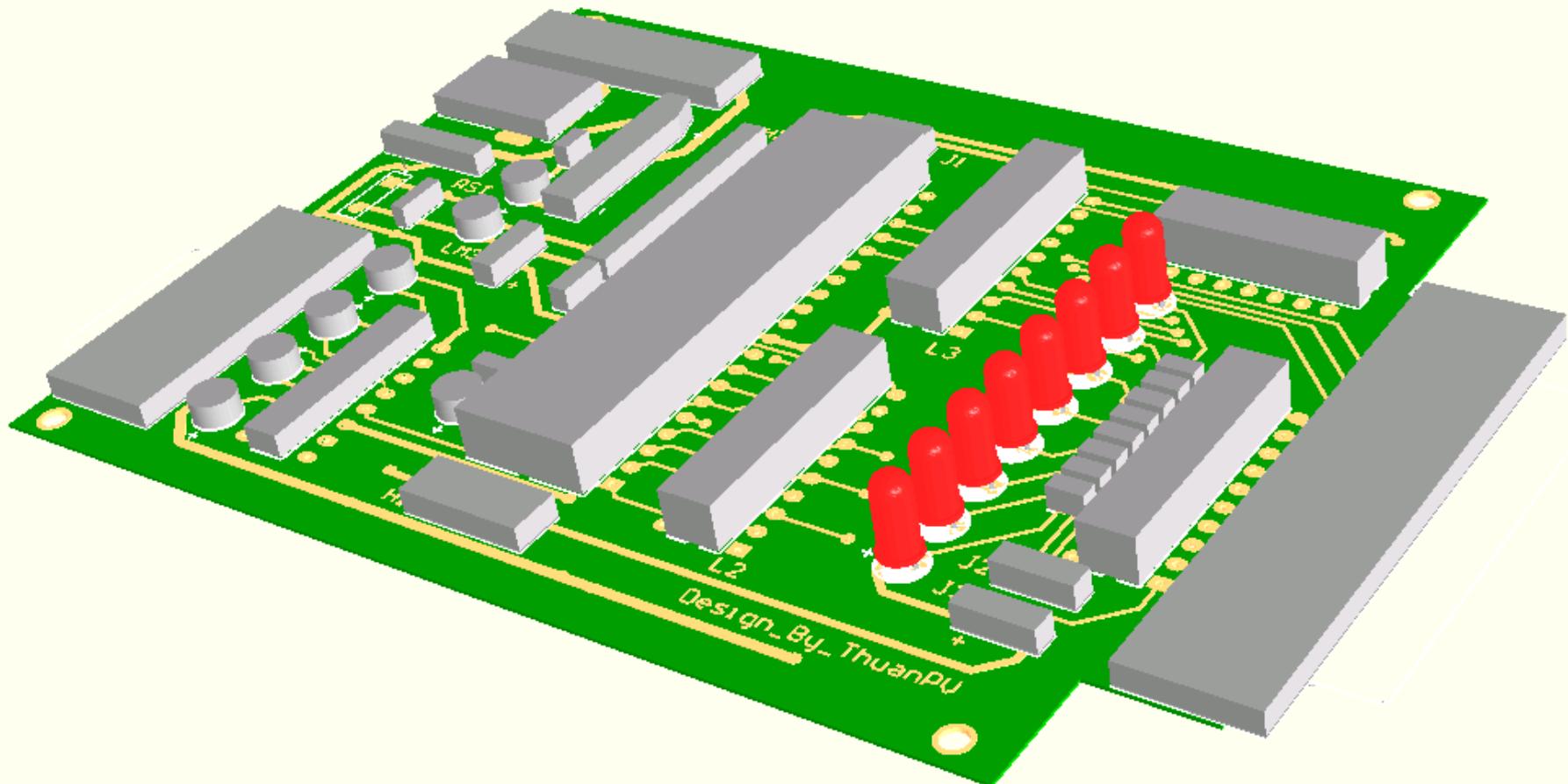
# Chuẩn RS232



Đầu đọc thẻ RFID  
(RFID card reader)



# Ví dụ chuẩn RS232



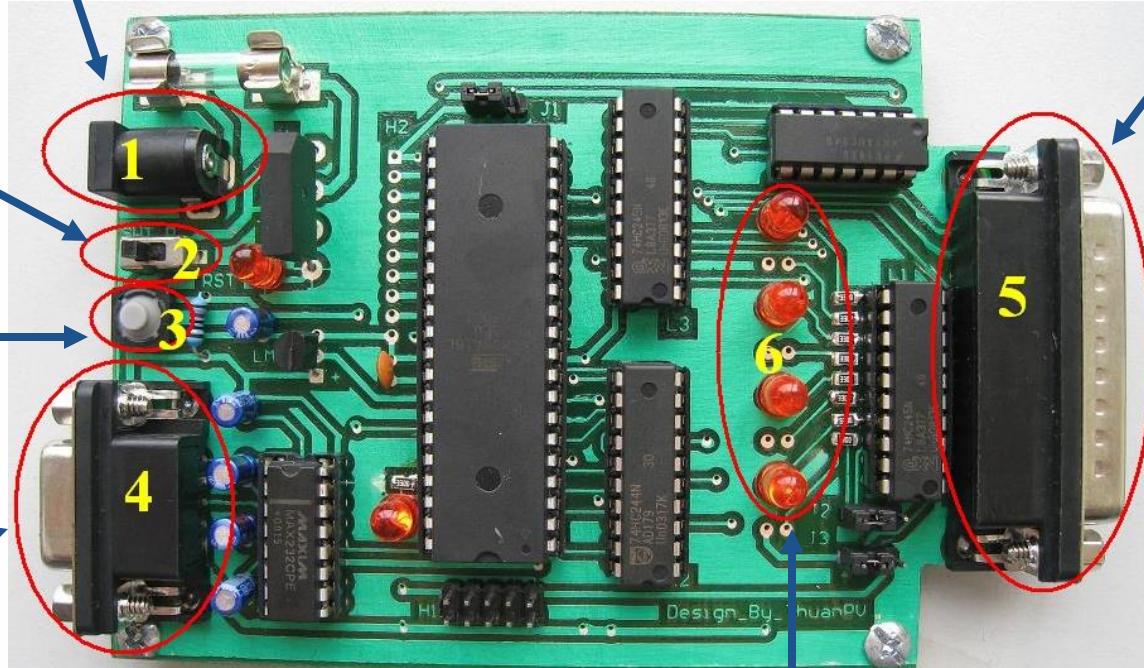
Mạch đo nhiệt độ



# Ví dụ chuẩn RS232

Công tắc nguồn  
Công tắc reset  
Cổng COM ghép nối máy tính

Jắc cắm nguồn  
Cổng LPT ghép nối máy tính



Các đèn Led được điều khiển bởi máy tính qua cổng LPT



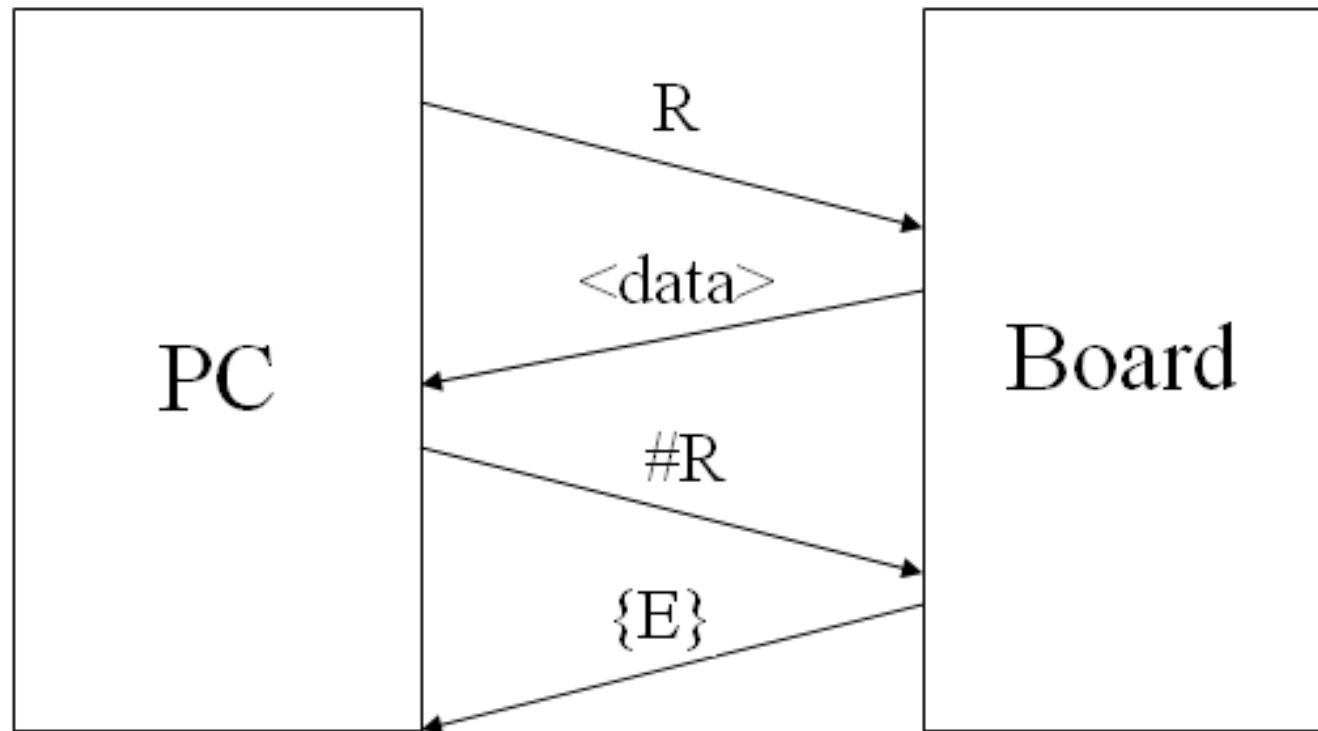
## Ví dụ chuẩn RS232

- Mạch đo nhiệt độ kết nối với máy tính qua cổng COM theo chuẩn RS232 với các thông số thiết lập như sau
  - Tốc độ truyền: 9600 bps
  - Khuôn dạng truyền: **9600, n, 8, 1**
    - ✓ 8 bit Data
    - ✓ 1 bit Stop
    - ✓ Không kiểm tra chẵn lẻ
  - Không sử dụng bắt tay





# Kịch bản ghép nối





# Kịch bản ghép nối

- Khi chương trình trên máy tính muốn đọc thông tin nhiệt độ, chương trình phải gửi một ký tự ‘R’ (Request) xuống board thí nghiệm.
- Chương trình trên Board sẽ kiểm tra, nếu đúng là ký tự ‘R’ được gửi, nó sẽ trả lại cho chương trình trên máy tính một gói tin có định dạng “<data>”
  - Data là số nguyên 8 bit
  - Để chuyển đổi số liệu này thành nhiệt độ đo được trên board, chúng ta sử dụng công thức sau

$$T = \text{data}/4$$

- Khi chương trình trên Board kiểm tra, nếu ký tự gửi xuống từ máy tính không phải là ‘R’, nó sẽ trả lại cho chương trình trên máy tính một gói tin có định dạng “{E}” để báo chương trình trên máy tính đã gửi sai lệnh.



## 2.2.3. Chuẩn LPT

- Giao diện song song: cho phép trao đổi nhiều bit dữ liệu tại cùng một thời điểm.
- Một số giao diện song song trên máy tính:
  - **LPT: cổng máy in**
  - SCSI (Small computer system interface): cho phép tối đa 7 thiết bị có thể kết nối với PC thông qua một cáp nối duy nhất, mỗi thiết bị có một địa chỉ duy nhất. (Ổ cứng, CD-ROMs...)
  - IEEE-488: cho phép tối đa 15 thiết bị có thể giao tiếp ở tốc độ 1MB/s



# Phân loại cổng LPT



## ECP (Extended Capabilities Port)

- Bus dữ liệu hai chiều, hỗ trợ DMA
- Có thể cấu hình để chạy với chuẩn SPP, PS/2, EPP

## EPP (Enhanced Parallel Port)

- Bus dữ liệu hai chiều, có sử dụng tín hiệu bắt tay
- Có thể cấu hình để chạy với chuẩn SPP, PS/2

## PS/2-type (Simple Bidirectional)

- Bus dữ liệu 2 chiều

## SPP (Standard Parallel Port)

- Thiết kế dựa trên giao tiếp của máy in Centronics
- Bus dữ liệu 1 chiều (PC-> ngoại vi), bus trạng thái 1 chiều, bus điều khiển 2 chiều



# SPP (Standard Parallel Port)

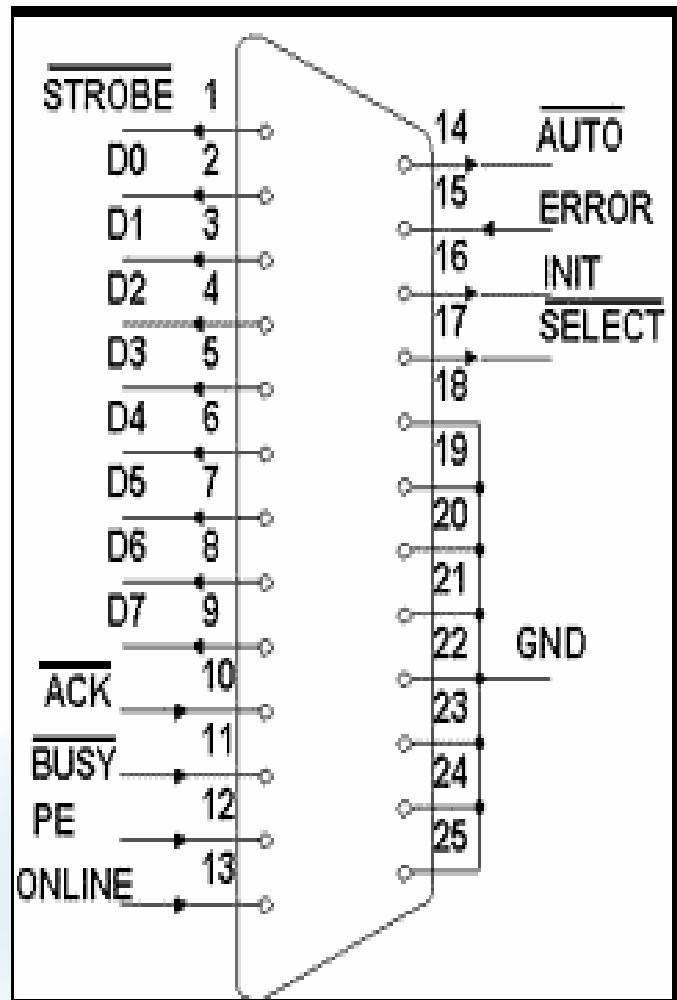
- **Mức điện áp làm việc:** mức TTL
- **Chuẩn đầu nối trên PC:** cổng DB-25, female





## Chức năng của các chân tín hiệu:

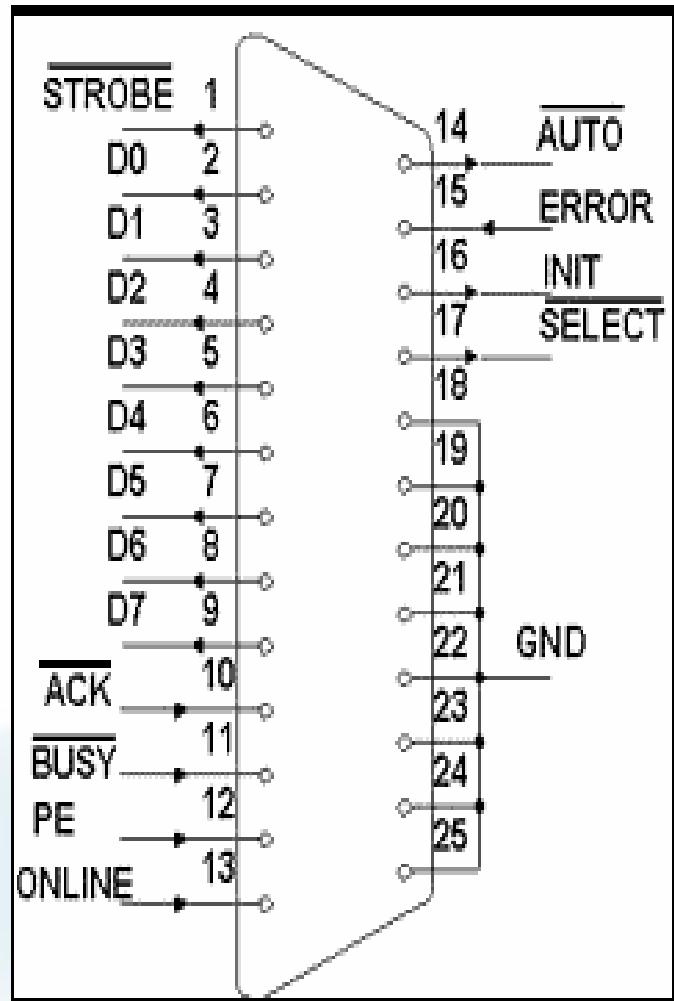
- **Strobe** (chân số 1): =0~máy tính báo cho máy in là sẵn sàng truyền một byte
- **D0-D7**: Các đường dẫn dữ liệu
- **ACK** (Acknowledgement): =0~Máy in thông báo cho máy tính biết là đã nhận được ký tự vừa gửi và có thể tiếp tục nhận
- **BUSY**: =1~máy in thông báo cho máy tính biết các bộ đệm trong máy in đầy
- **Paper Empty**: 1~ Hết giấy
- **ONLINE (SELECT)**: 1 ~ Máy in đang ở trạng thái được kích hoạt





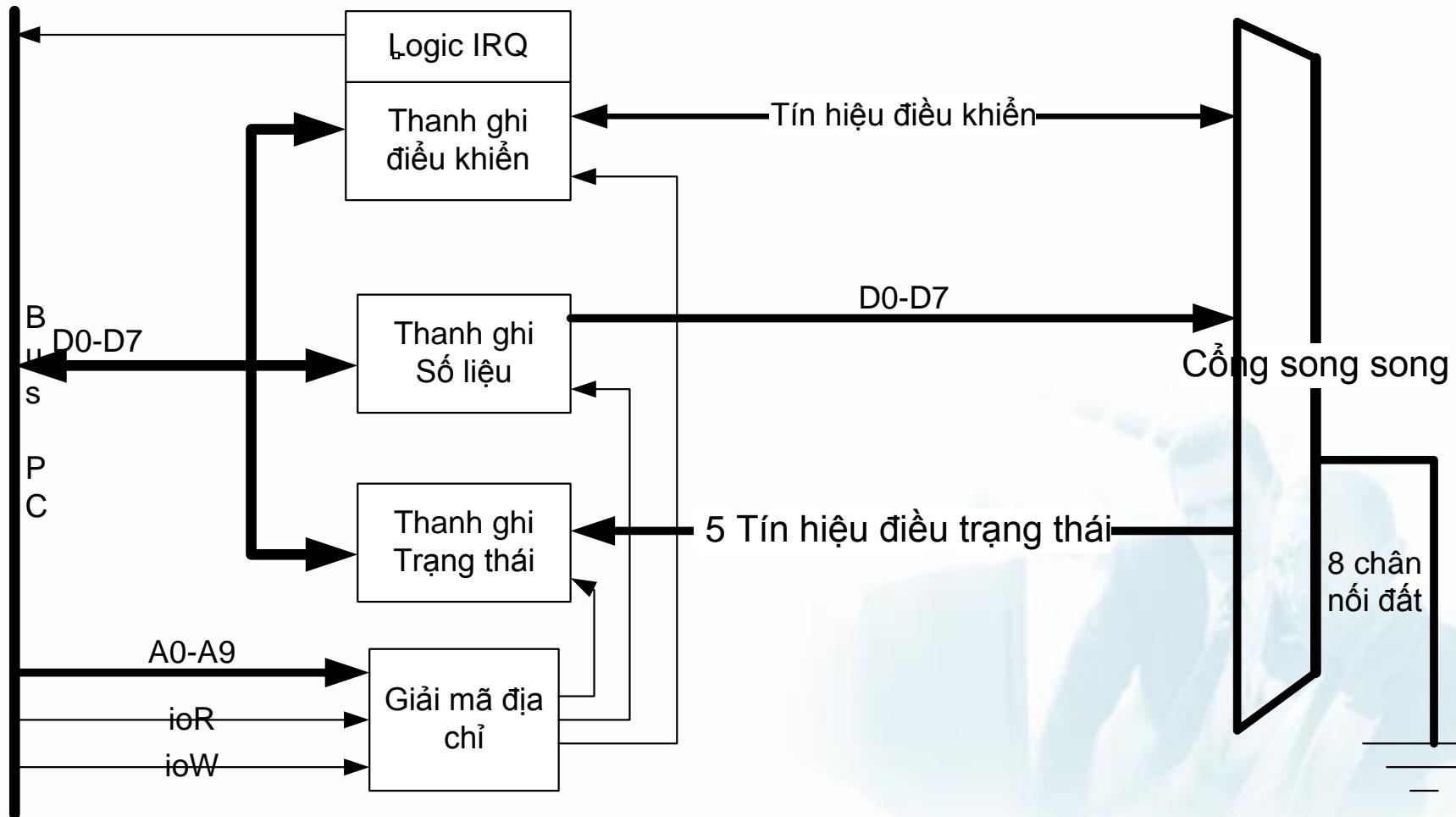
# Cổng LPT

- **Auto Linefeed:** 1~PC nhắc máy in tự động nạp dòng mới mỗi khi kết thúc một dòng
- **Error:** 0~máy in báo lỗi (kẹt giấy, máy in ở trạng thái off-line...)
- **INIT (RESET):** 0~máy in được đặt trở lại trạng thái được xác định ban đầu
- **Select Input:** 0~ máy in được lựa chọn bởi máy tính.
- Các chân từ 19-25 được nối đất.





## Sơ đồ khối:





- Các đường dẫn của cổng LPT được nối với ba thanh ghi 8 bit khác nhau:
  - Thanh ghi dữ liệu (Data Register)
    - ✓ SPP: 8 đầu ra
  - Thanh ghi trạng thái (Status Register)
    - ✓ SPP: 5 đầu vào
  - Thanh ghi điều khiển (Control Register)
    - ✓ SPP: 4 chân tín hiệu hai chiều





# Thanh ghi dữ liệu

Địa chỉ cơ sở

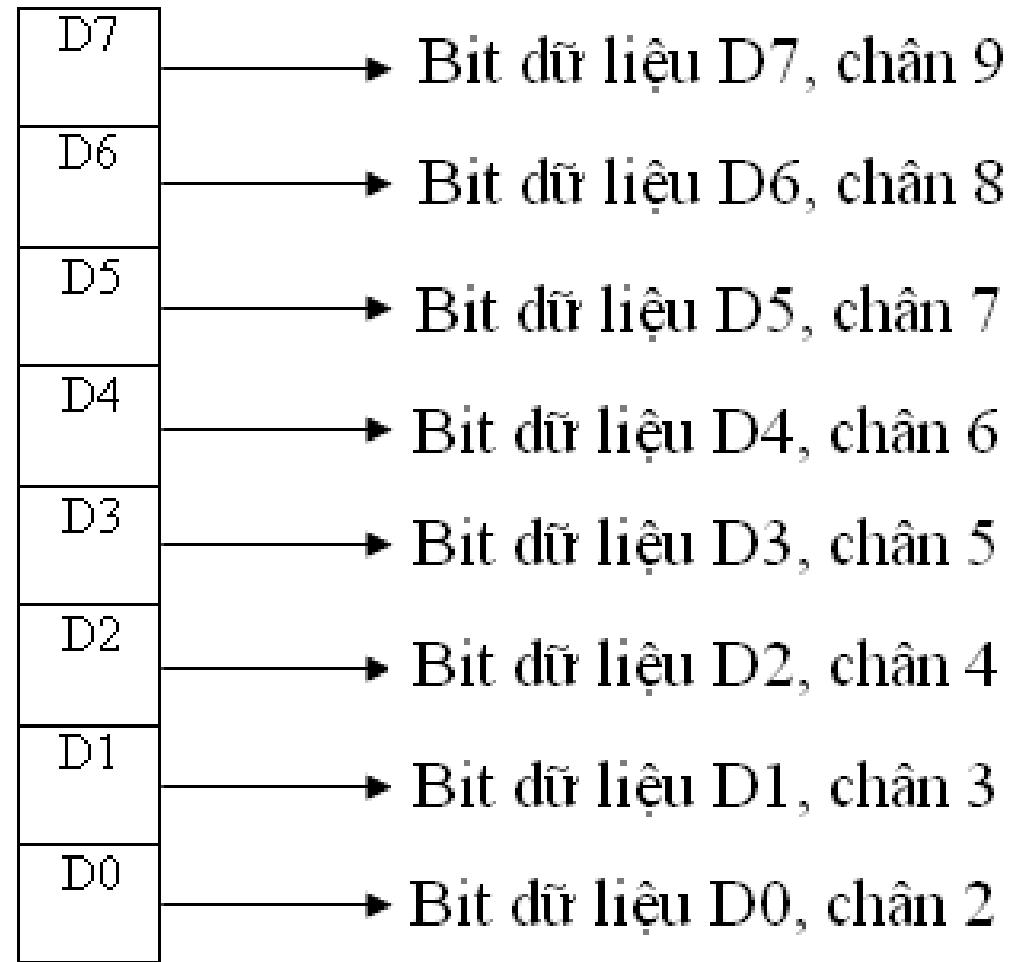
0378h

0278h

03BCh

02BCh

Thanh ghi dữ liệu





# Thanh ghi trạng thái

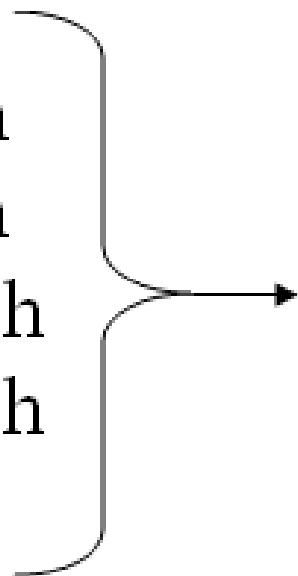
Địa chỉ cơ sở + 1

0379h

0279h

03BDh

02BDh



Thanh ghi trạng thái

7	Busy, chân 11
6	Acknowledge, chân 10
5	Paper Empty, chân 12
4	Select, chân 13
3	Error, chân 15

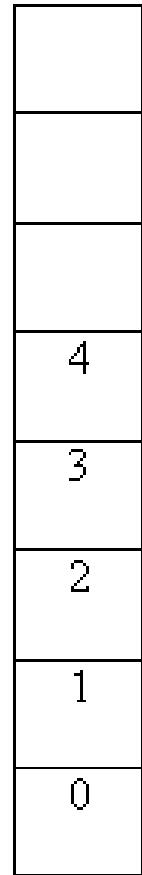


# Thanh ghi điều khiển

Địa chỉ cơ sở + 2

037Ah  
027Ah  
03BEh  
02BEh

Thanh ghi điều khiển



Interrupt Enable

Select Input, chân 17

Reset, chân 16

Auto Feed, chân 14

Strobe, chân 1



# Tìm kiếm các cổng LPT

- Một máy PC có thể hỗ trợ nhiều cổng LPT
  - Thông thường: 3 cổng LPT
  - Đặc biệt: 4 cổng (hiếm gặp)

Cổng song song (LPT)	Địa chỉ thanh ghi dữ liệu	Địa chỉ thanh ghi trạng thái	Địa chỉ thanh ghi điều khiển
LPT 1	3BCh	3BDh	3BEh
LPT 2	378h	379h	37Ah
LPT 3	278h	279h	27Ah
LPT 4	2BCh	2BDh	2BEh

Tên và địa chỉ các cổng LPT trong trường hợp máy có đầy đủ cả 3 (4) cổng



## Tìm kiếm các cổng LPT

- Khi khởi động, một chương trình BIOS tự động kiểm tra sự tồn tại các cổng LPT tại 3 địa chỉ theo thứ tự: 3BCh, 378h, 278h (**2BCh**)
  - Ghi dữ liệu ra cổng có địa chỉ tương ứng
  - Đọc ngược dữ liệu từ cổng, nếu cặp dữ liệu khớp nhau -> có tồn tại cổng tại địa chỉ đó -> Tiến hành gán tên cho cổng theo thứ tự LPT1, LPT2, LPT3, (**LPT4**)

Địa chỉ cơ sở của LPT1 có thể không phải là 3BCh, Địa chỉ cơ sở của LPT2 có thể không phải là 378h...





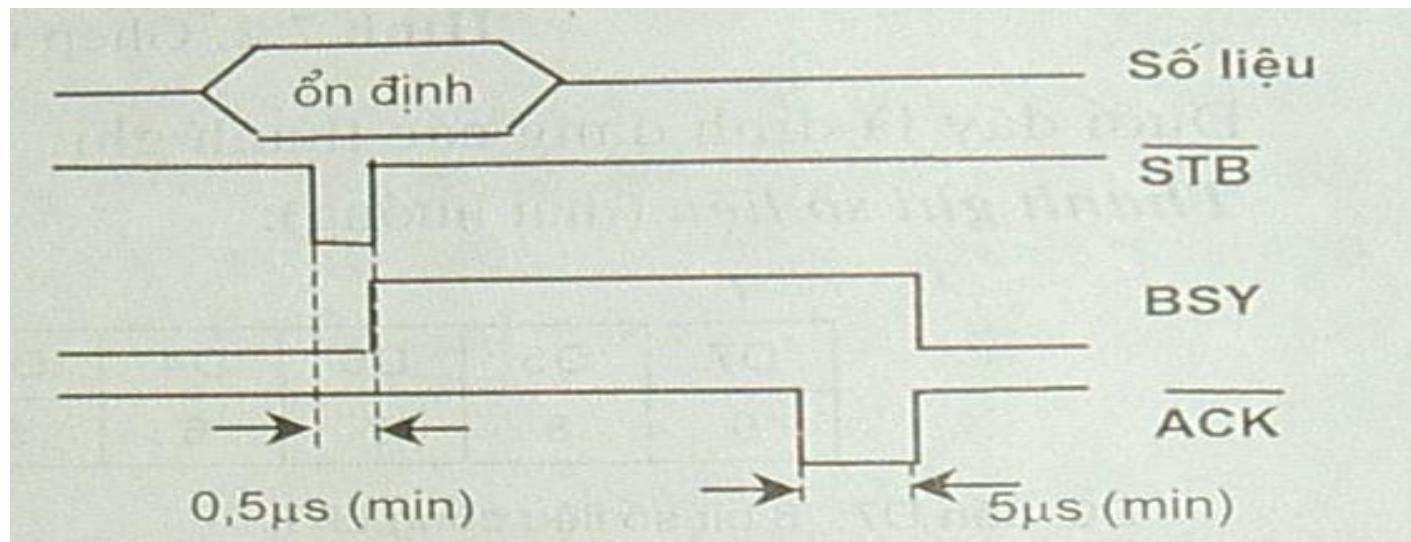
# Xác định địa chỉ cổng LPT

- 40:08 Địa chỉ cơ bản của LPT1
- 40:0A Địa chỉ cơ bản của LPT2
- 40:0C Địa chỉ cơ bản của LPT3
- 40:0E Địa chỉ cơ bản của LPT4**
- 40:11 Cấu hình hệ thống: bit thứ 7 và thứ 8  
chứa số được mã hóa nhị phân của  
cổng song song hiện có



# Kịch bản ghép nối máy in (Chuẩn SPP)

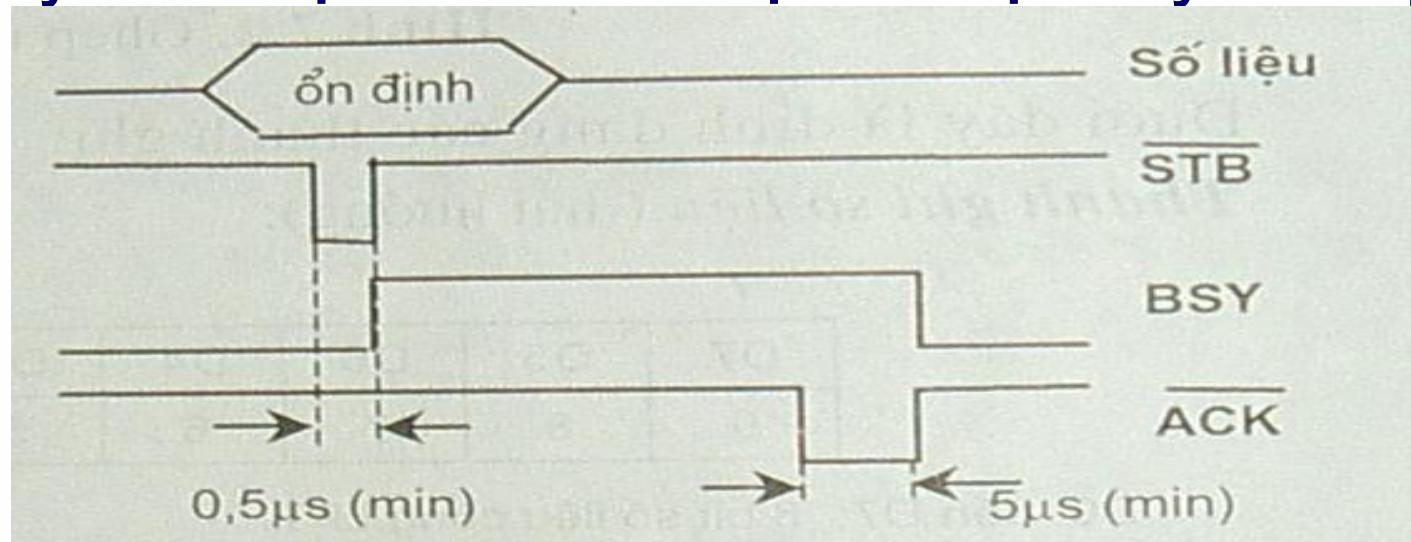
- Chế độ hoạt động cơ bản (Standard Parallel Port –SPP):
  - Truyền tin có báo nhận: đồng bộ qua STB và báo nhận qua ACK





# Kịch bản ghép nối máy in (Chuẩn SPP)

- Máy tính đặt số liệu lên bus
- Kích hoạt /STB : báo cho máy in dữ liệu đã ổn định.
- Máy in nhận xong sẽ trả lại /ACK ở mức thấp.
- Máy tính đợi BSY hết bận thì lại truyền tiếp

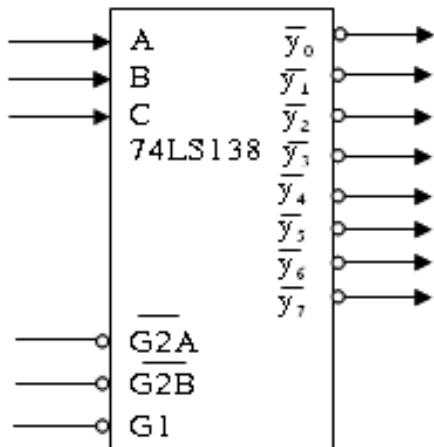




# Ghép nối qua cổng LPT (Mạch phụ trợ)

## Bộ giải mã 74LS138

- 3 đầu vào (A, B, C), 8 đầu ra đảo ( $y_0 \rightarrow y_7$ )
- 3 tín hiệu điều khiển (G2A, G2B, G1)



G1	G2A	G2B	Vào			Ra							
			C	A	B	$\bar{y}_0$	$\bar{y}_1$	$\bar{y}_2$	$\bar{y}_3$	$\bar{y}_4$	$\bar{y}_5$	$\bar{y}_6$	$\bar{y}_7$
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	0	0	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
Các tổ hợp khác			x	x	x	1	1	1	1	1	1	1	1



# Ghép nối qua cổng LPT (Mạch phụ trợ)

- Mạch chốt 74LS373: dùng để chốt
  - Chân OC (Output Control): tích cực mức thấp, điều khiển cho phép/ cấm dữ liệu ra
  - Chân G: chân chốt (Khi OC=0)
    - ✓  $G=0 \rightarrow$  đầu ra là dữ liệu trước đó
    - ✓  $G=1 \rightarrow$  đầu ra = đầu vào

3	D0	Q0	2
4	D1	Q1	5
7	D2	Q2	6
8	D3	Q3	9
13	D4	Q4	12
14	D5	Q5	15
17	D6	Q6	16
18	D7	Q7	19
19	OC		
11	G		

74LS373

Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	$Q_0$
H	X	X	Z

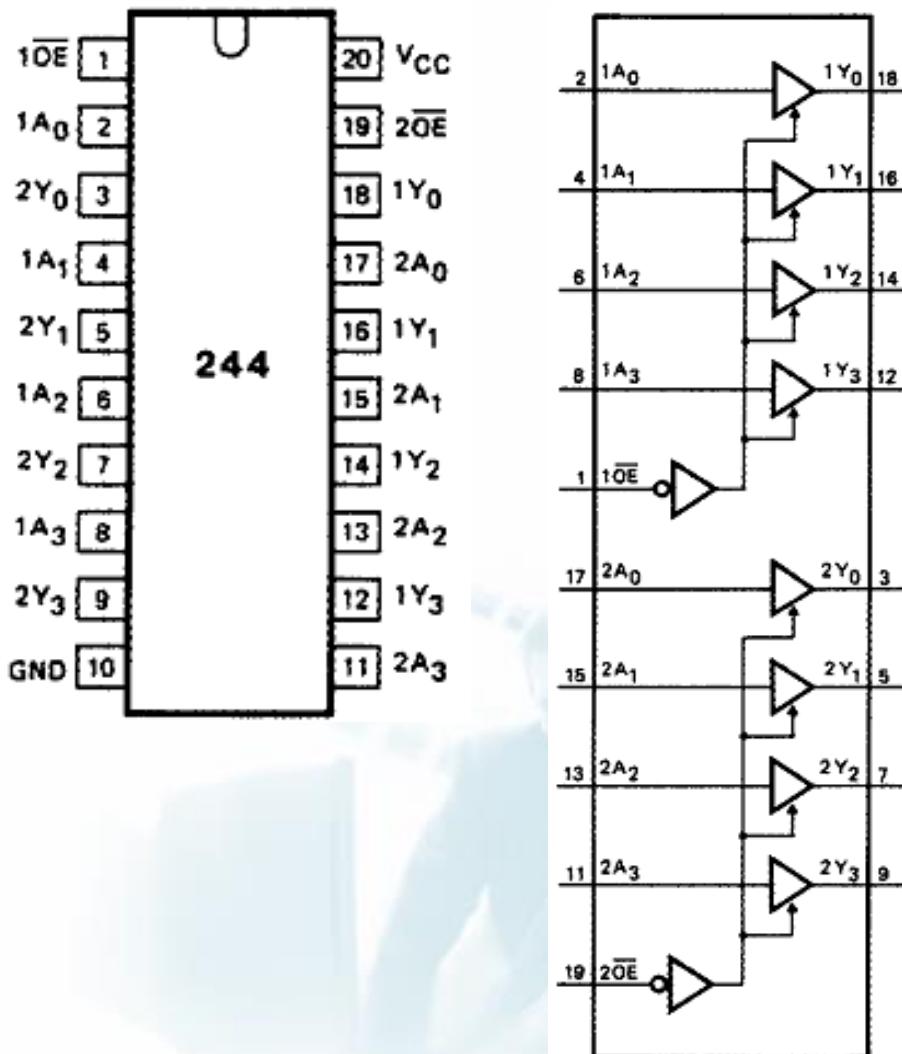
Bảng thật của 74LS373



# Ghép nối qua cỗng LPT (Mạch phụ trợ)

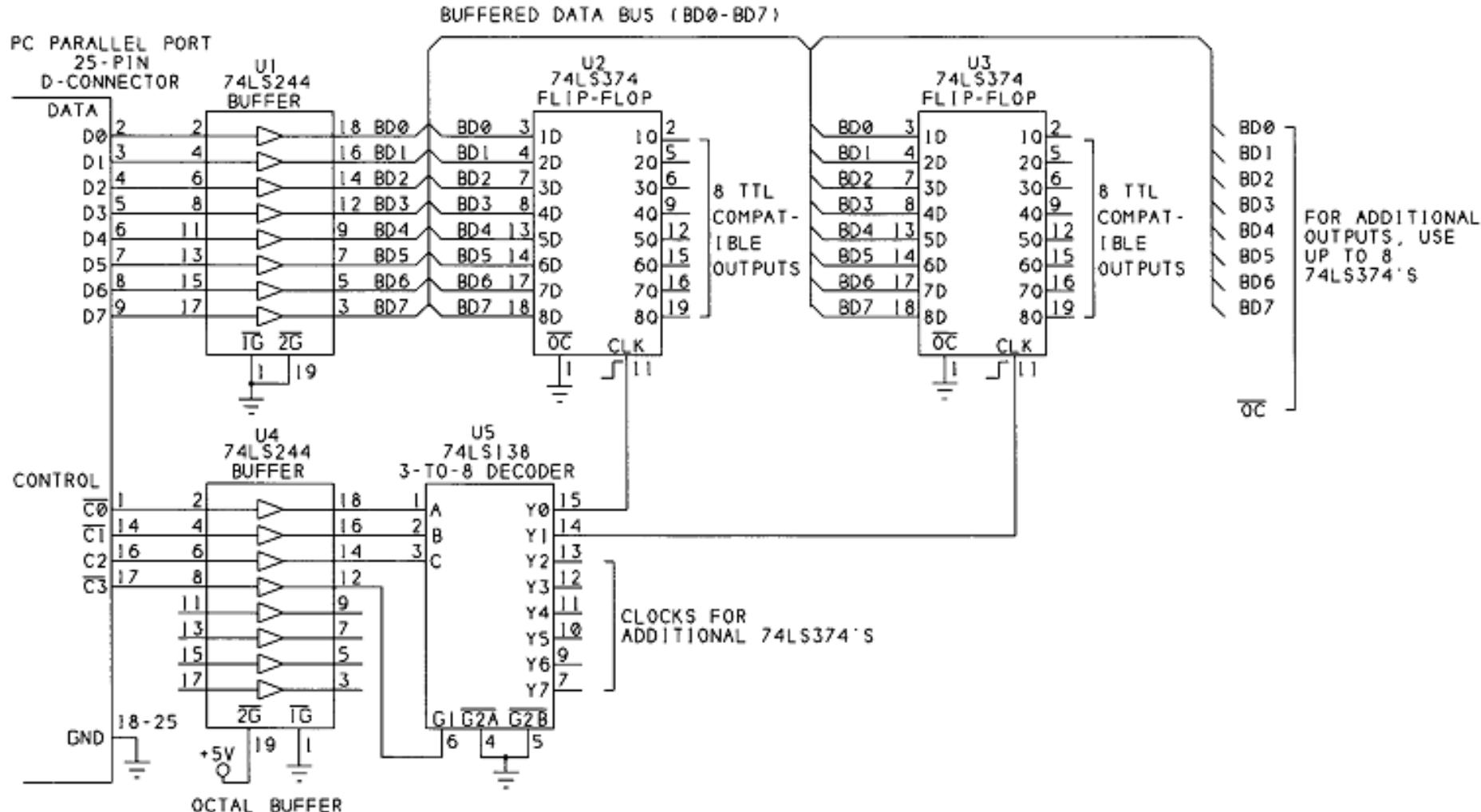
## Mạch khuếch đại & đếm một chiều 74HC244

- Có thể đếm từng nibble (4bit) hoặc đếm một byte (8bit)
- Điều khiển bởi hai chân 1OE và 2OE (Output Enable)





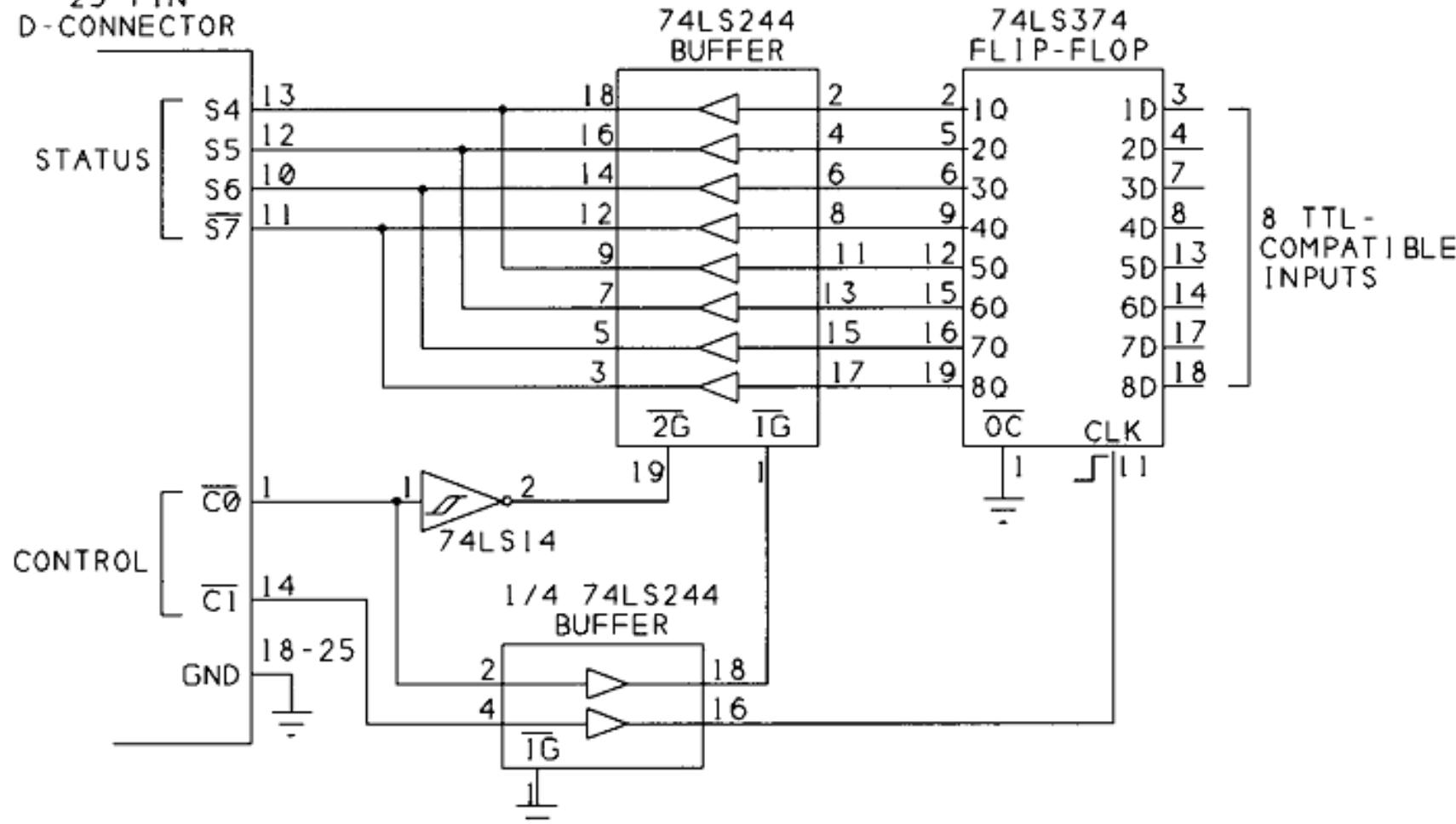
# Ghép nối qua cỗng LPT (Xuất dữ liệu)





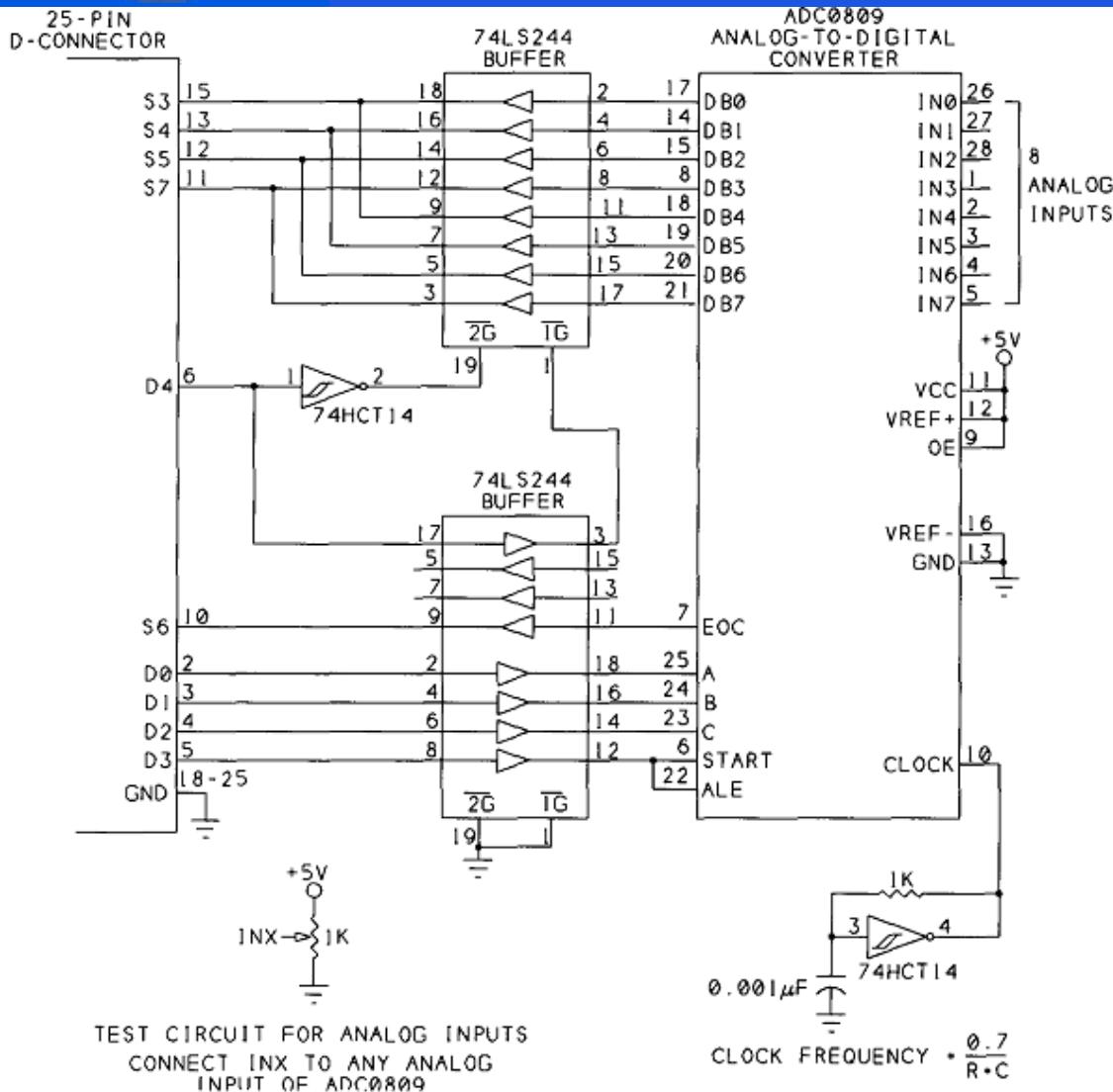
# Ghép nối qua cổng LPT (Nhập dữ liệu)

PC PARALLEL PORT  
25-PIN  
D-CONNECTOR





# Ghép nối qua cổng LPT (Nhập dữ liệu)



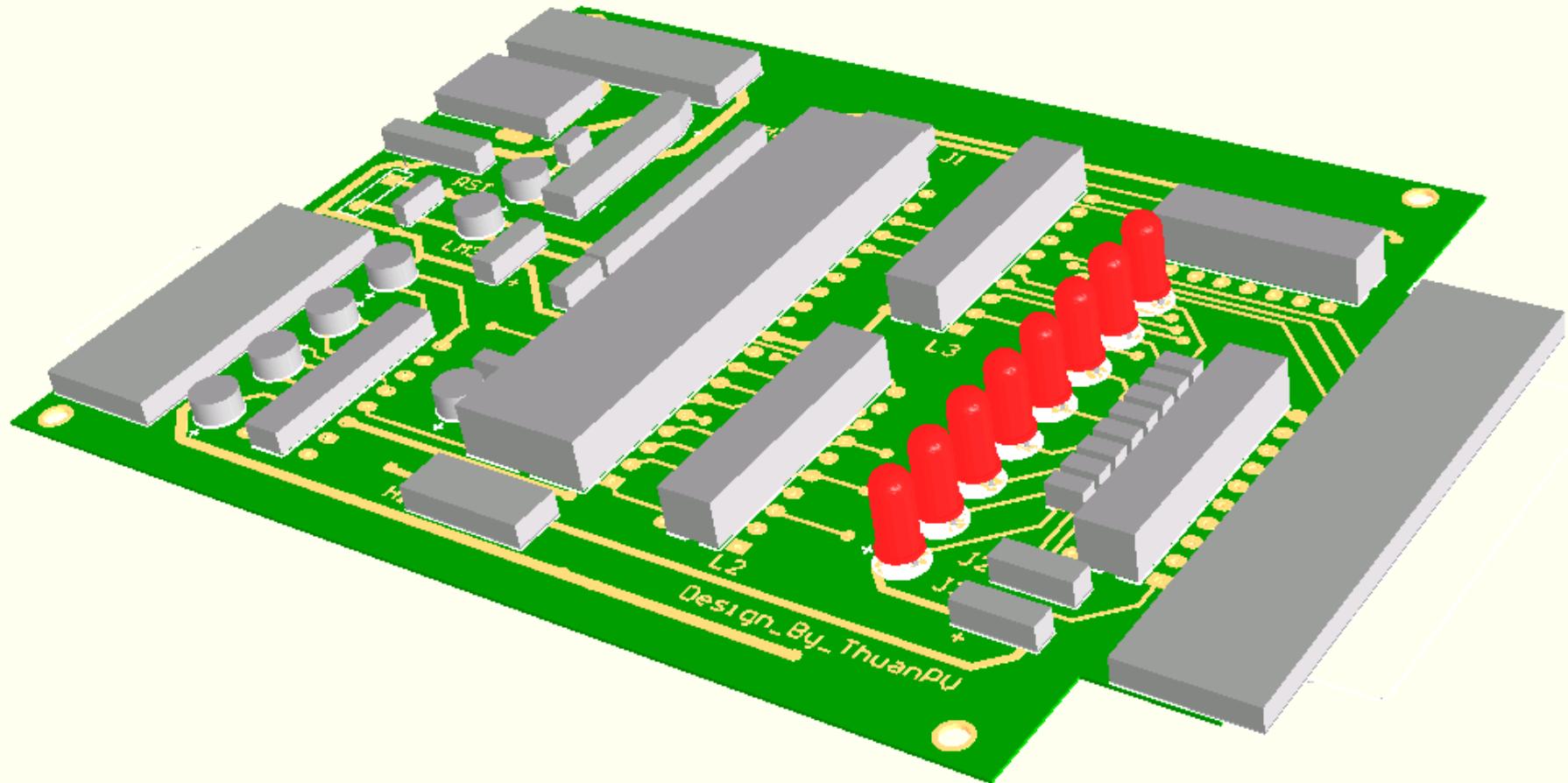
▪ Ghép nối với bộ chuyển đổi tương tự-số ADC0809

▪ Đọc dữ liệu theo từng nibble (qua 4 chân trạng thái)-> cần hai chu kỳ đọc để nhận 1 byte dữ liệu

▪ Sử dụng các chân dữ liệu để điều khiển



# Ví dụ chuẩn LPT



Mạch đo nhiệt độ



# Ví dụ chuẩn LPT



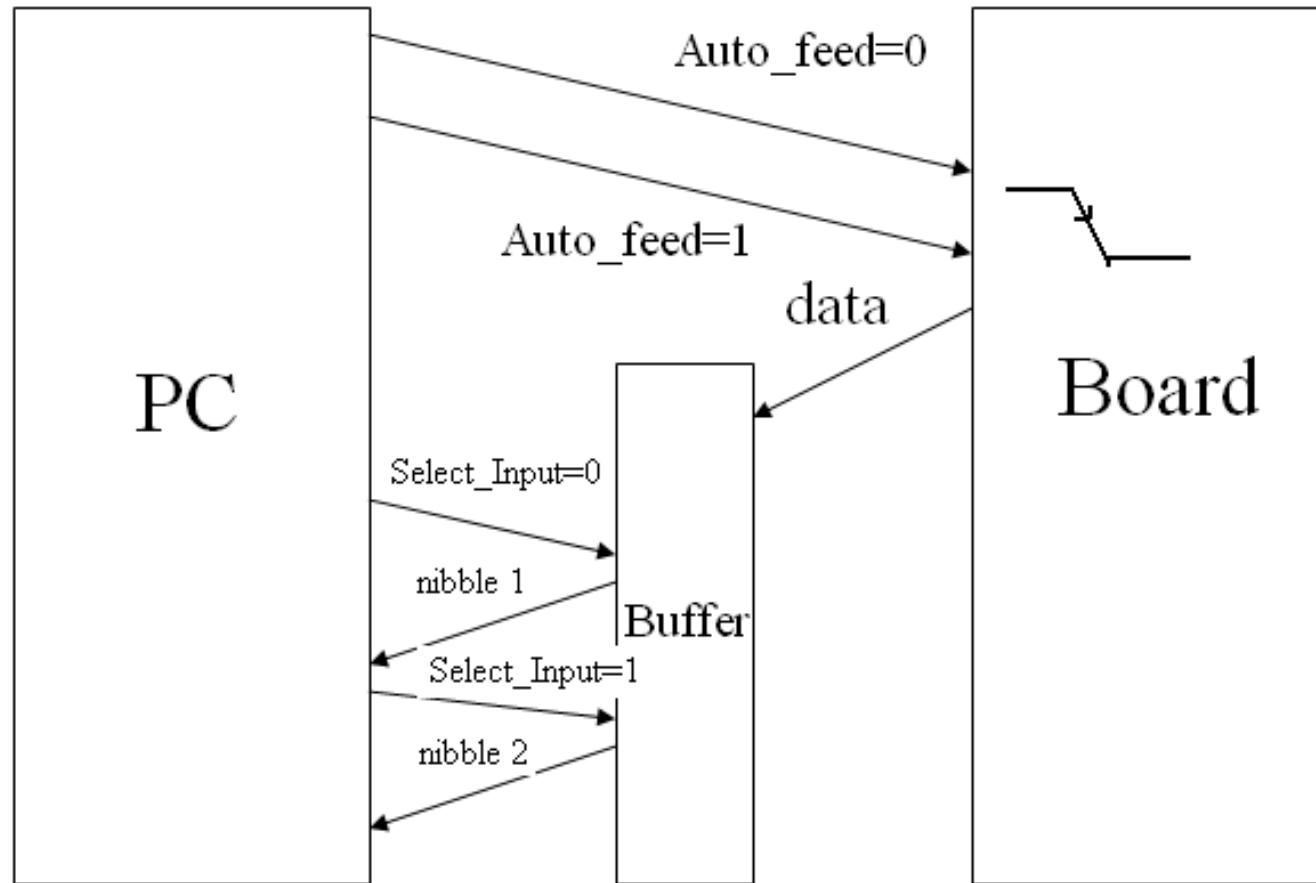


## Ví dụ chuẩn LPT

- Máy tính ghép nối với mạch đo nhiệt độ qua cổng LPT sử dụng chuẩn SPP
  - 8 chân dữ liệu để điều khiển bật, tắt các đèn led trên mạch đo nhiệt độ
  - 2 chân điều khiển dùng để gửi các tín hiệu điều khiển từ máy tính tới mạch đo nhiệt độ
    - ✓ Tín hiệu yêu cầu gửi số liệu nhiệt độ
    - ✓ Tín hiệu điều khiển để đọc dữ liệu nhiệt độ
  - 4 chân trạng thái sử dụng để đọc dữ liệu nhiệt độ gửi từ dưới mạch đo nhiệt độ



# Kịch bản ghép nối





# Kịch bản ghép nối

- Chương trình trên máy tính gửi một xung cao xuống thấp qua chân Auto\_feed (Thanh ghi điều khiển của cổng LPT) để yêu cầu board gửi dữ liệu nhiệt độ đo được
- Board gửi dữ liệu nhiệt độ (data) lên máy tính, dữ liệu được đặt tại bộ đệm
- Chương trình trên máy tính thiết lập chân select\_input lên mức 1 (bit select input=0, do select\_input là chân đầu ra đảo). Việc thiết lập này cho phép dồn kênh 4 bit cao của dữ liệu nhiệt độ lưu vào 4 bit cao của thanh ghi trạng thái. Chương trình trên máy tính đọc thanh ghi trạng thái.
- Chương trình trên máy tính tiếp tục xóa chân select\_input (bit select\_input=1 do select\_input là chân đầu ra đảo). Việc xóa chân select\_input cho phép dồn kênh 4 bit thấp của dữ liệu nhiệt độ vào 4 bit cao của thanh ghi trạng thái. Chương trình trên máy tính đọc thanh ghi trạng thái.



## 2.2.4. Chuẩn USB

- Chuẩn USB ra đời nhằm thỏa mãn các yêu cầu của một chuẩn ghép nối mới:
  - Dễ sử dụng
  - Tốc độ cao
  - Tin cậy, ít lỗi, có cơ chế tự động kiểm tra lỗi
  - Linh hoạt -> nhiều loại thiết bị ngoại vi có thể sử dụng
  - Giá thành phù hợp
  - Tiết kiệm điện
  - Hỗ trợ bởi hệ điều hành





# Sự phát triển của chuẩn USB

- Năm 1995: USB 1.0
  - Tốc độ Low-Speed: 1.5 Mbps
  - Tốc độ tối đa (Full-Speed): 12 Mbps
- Năm 1998: USB 1.1 (Sửa lỗi của USB 1.0)
  - Tốc độ tối đa (Full-Speed): 12 Mbps
- Năm 2001: USB 2.0
  - Tốc độ tối đa (High-Speed): 480 Mbps
- Năm 2006: Đánh dấu sự ra đời của chuẩn USB On-The-Go (USB OTG)
- Năm 2008: USB 3.0
  - Tốc độ tối đa (Super-Speed): 4.8 Gbps



# Đặc tính của chuẩn USB

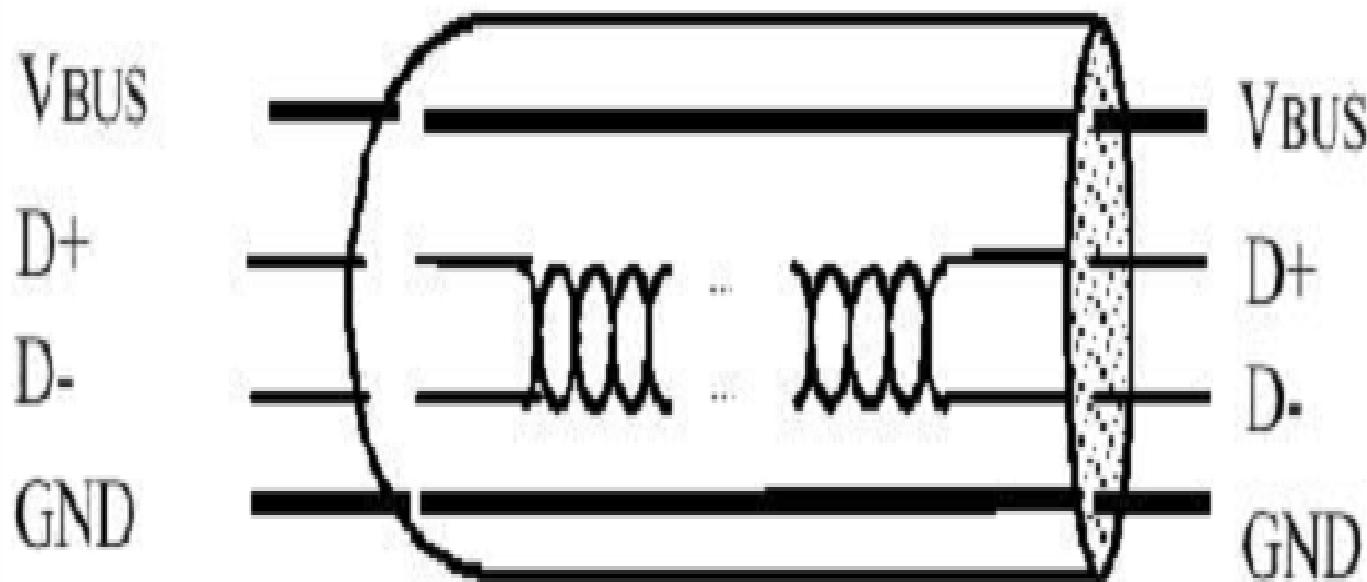
- Tín hiệu
- Chuẩn đầu nối
- Khuôn dạng khung truyền
- Tốc độ truyền
- Kích bản truyền





- Tín hiệu

- Truyền kiểu nối tiếp
- Tín hiệu trên hai đường D+ và D- là tín hiệu vi sai





## Mức điện áp

- Cổng USB trên máy tính cung cấp điện áp ra +5V, dòng tối đa là 500mA
  - Có thể lên đến 5.25V
  - Khi sụt áp có thể sụt còn 4.2V
    - > Những thiết bị sử dụng ít điện năng có thể sử dụng nguồn cấp ngay từ cổng USB
    - > Những thiết bị sử dụng nhiều điện năng bắt buộc phải sử dụng nguồn ngoài
- Tín hiệu trên hai đường D+ và D- là tín hiệu vi sai



# Chuẩn đầu nối USB



## Non-Standard Connector

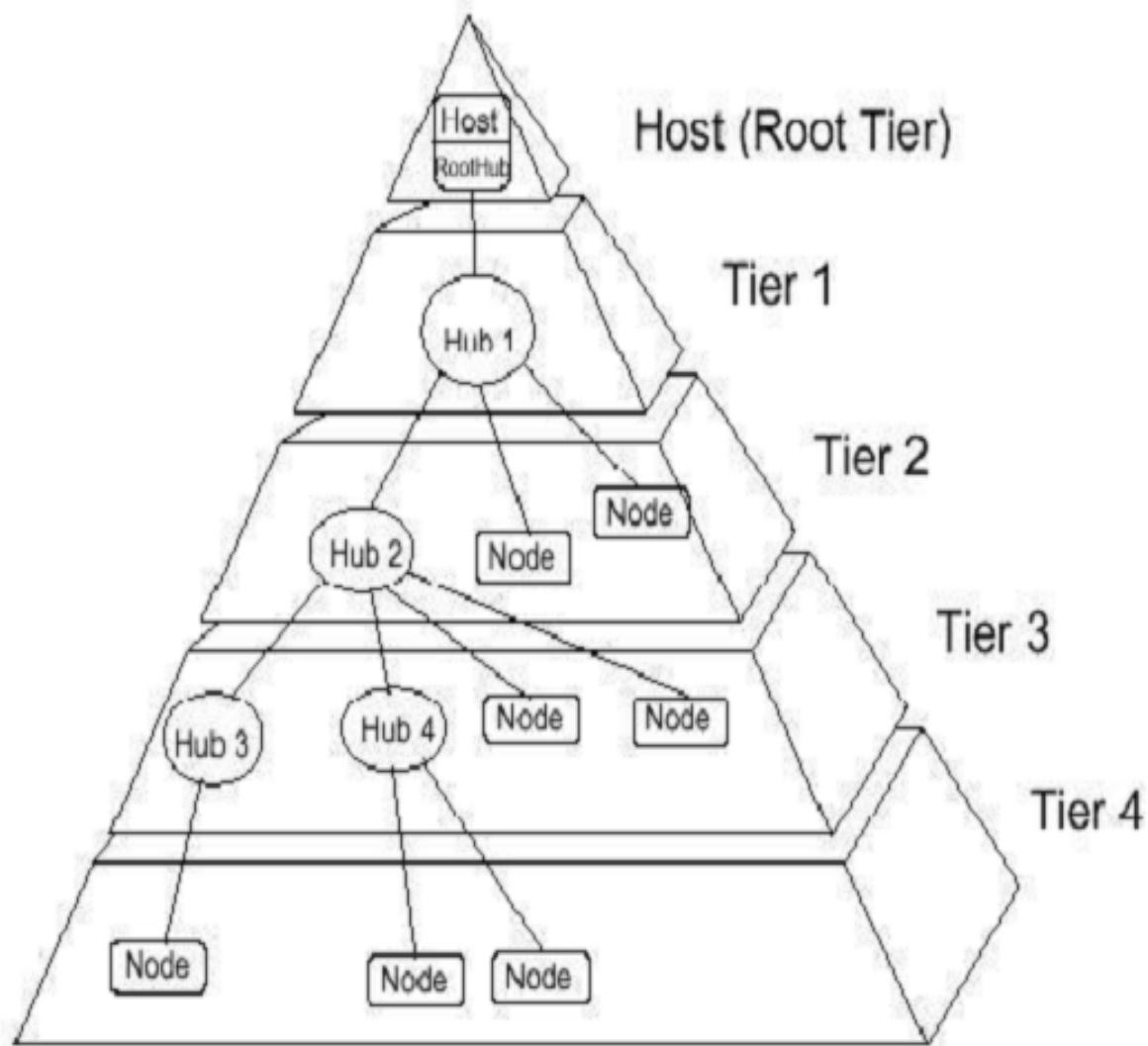
- Mini A/B (Plug and Receptacle)
- Micro A/B(Plug and Receptacle)

## Standard Connector

- USB A/B (Plug and Receptacle)



# Sơ đồ kết nối bus USB





# Các thành phần của bus USB

- Host: máy tính PC/ máy tính nhúng chứa USB host controller & root hub có chức năng điều khiển bus
- Devices: các thiết bị ngoại vi hoặc hub kết nối vào đường bus USB. Mỗi thiết bị cần có mạch điện tử và phần mềm điều khiển giao tiếp với Host
- Connectors: đầu đầu nối
- Cable: kết nối giữa thiết bị và hub



# Sơ đồ kết nối bus USB

- Sơ đồ kết nối bus USB theo kiểu hình sao
  - Trung tâm của mỗi “sao” là một hub
  - Các thiết bị được cắm vào hub
  - Mỗi hub thường cho 2, 4 hoặc 7 thiết bị có thể cắm vào đồng thời
- Sơ đồ kết nối chỉ là kết nối vật lý
  - Các giao tiếp trong bus chỉ cần quan tâm đến kết nối logic
  - Tại một thời điểm, chỉ một thiết bị có thể giao tiếp với host controller



# Vai trò của các thành phần

- Vai trò của USB host:

- Trao đổi dữ liệu với các thiết bị ngoại vi
- Điều khiển USB bus:
  - ✓ Quản lý được các thiết bị kết nối vào đường bus và khả năng của mỗi thiết bị đó: sử dụng cơ chế điểm danh (Enumeration)
  - ✓ Phân xử, quản lý luồng dữ liệu trên bus, đảm bảo các thiết bị đều có cơ hội trao đổi dữ liệu
- Kiểm tra lỗi: thêm các mã kiểm tra lỗi vào gói tin cho phép phát hiện lỗi và yêu cầu truyền lại gói tin
- Cung cấp nguồn điện cho tất cả các thiết bị



# Vai trò của các thành phần

- Vai trò của thiết bị ngoại vi

- Trao đổi dữ liệu với host
- Phát hiện các gói tin hay yêu cầu (request) được gửi tới thiết bị để xử lý phù hợp
- Kiểm tra lỗi: tương tự như Host, các thiết bị ngoại vi cũng phải chèn thêm các bit kiểm tra lỗi vào gói tin gửi đi
- Quản lý nguồn điện: các thiết bị có thể sử dụng nguồn điện ngoài hay nguồn từ bus. Nếu sử dụng nguồn từ bus, phải chuyển sang chế độ tiết kiệm điện năng.



# Enpoint & pipes

- Mỗi quá trình truyền nhận dữ liệu bao gồm một hay nhiều giao dịch (transactions), mỗi giao dịch gồm một hay nhiều packets
- > Để hiểu được các giao dịch, các packet và nội dung của chúng -> cần tìm hiểu hai khái niệm Enpoint và Pipes





- Endpoint của thiết bị:

- Chỉ có thiết bị mới có Endpoint, Host không có Endpoint
- Endpoint là bộ đệm (gửi, nhận)
- Các Endpoint được đánh địa chỉ và xác định hướng
  - ✓ In Endpoint: bộ đệm gửi
  - ✓ Out Endpoint: bộ đệm nhận
- Tất cả các thiết bị đều phải có Endpoint 0, đây là endpoint mặc định để gửi các thông tin điều khiển



- Pipes: kết nối Endpoint của thiết bị tới Host
  - Phải thiết lập pipe trước khi muốn trao đổi dữ liệu
  - Host thiết lập pipe trong quá trình điểm danh (Enumeration)
  - Các Pipe sẽ được hủy khi thiết bị ngắt kết nối khỏi bus
  - Tất cả các thiết bị đều có một đường ống điều khiển (control pipe) mặc định sử dụng Endpoint 0



## Kịch bản hoạt động của chuẩn USB

- Quá trình điểm danh: Host phát hiện các thiết bị kết nối vào bus hoặc ngắt kết nối khỏi bus và các thông tin về thiết bị đó
- Quá trình trao đổi dữ liệu: trao đổi dữ liệu giữa Host và các thiết bị kết nối vào đường bus USB





# Quá trình điểm danh

- Các phần mềm firmware trên thiết bị phản hồi lại các yêu cầu (request) của Host
- Các phần mềm firmware phải xác định mỗi yêu cầu, trả về thông tin yêu cầu và làm các công việc phù hợp
- Trên PCs, quá trình điểm danh được thực hiện bởi hệ điều hành (VD: Windows) -> không cần sự can thiệp của người dùng nếu driver cho thiết bị đó đã tồn tại trên máy



# Trình tự quá trình điểm danh

- Người sử dụng cắm một thiết bị vào cổng USB (hoặc máy tính khởi động với một thiết bị USB đã được cắm sẵn): Hub cung cấp nguồn cho thiết bị
- Hub phát hiện thiết bị
- Host tìm hiểu thông tin về thiết bị mới: Host gửi đến các Hub yêu cầu Get\_Port\_Status, hub sẽ gửi về thông tin trạng thái các cổng trên hub
- Hub kiểm tra xem thiết bị sử dụng low hay full speed: thông tin này được gửi lên Host trong thông tin phản hồi lại yêu cầu Get\_Port\_Status tiếp theo



# Trình tự quá trình điểm danh

- Hub khởi động lại thiết bị: khi Host tìm hiểu được một số thông tin về thiết bị mới, Host controller gửi một yêu cầu Set\_Port\_Feature xuống Hub yêu cầu Hub khởi động lại thiết bị (chỉ khởi động lại thiết bị mới cắm vào)
- Hub kiểm tra xem thiết bị có hỗ trợ high speed không
- Hub thiết lập đường trao đổi tín hiệu giữa Host và thiết bị: thiết bị giao tiếp với Host qua địa chỉ mặc định là 00 và Enpoint mặc định là 0



# Trình tự quá trình điểm danh

- Host đọc byte đầu tiên của bản tóm lược thiết bị để xác định kích thước gói dữ liệu
- Host gán cho thiết bị một địa chỉ bus riêng cho thiết bị
- Qua địa chỉ mới, Host đọc tất cả các thông tin cấu hình từ thiết bị: kích thước tối đa của packet cho Endpoint 0, các cấu hình mà thiết bị có thể hỗ trợ ...
- Host gán và load driver cho thiết bị
- Host gán cho thiết bị một trong các cấu hình cụ thể



# Device Classes

- Các thiết bị ngoại vi cùng chức năng (chuột, máy in, ổ nhớ flash...) có đặc tính truyền nhận dữ liệu chung -> Hệ điều hành có thể cung cấp driver chung cho các nhóm, các nhà sản xuất thiết bị không cần viết driver riêng.
- Các nhóm thiết bị đã được định nghĩa
  - Audio
  - Communication devices
  - **Human interface (HID)**
  - IrDA Bridge
  - **Mass Storage**
  - Cameras and scanners
  - Video



# Quá trình trao đổi dữ liệu

- Các thiết bị USB có thể trao đổi dữ liệu với Host theo 4 kiểu hoàn toàn khác nhau, cụ thể:
  - Truyền điều khiển (control transfer)
  - Truyền ngắt (interrupt transfer)
  - Truyền theo khối (bulk transfer)
  - Truyền đẳng thời (isochronous transfer)



# Các kiểu truyền

- **Truyền điều khiển:** để điều khiển phần cứng, các yêu cầu điều khiển được truyền. Chúng làm việc với mức ưu tiên cao và với khả năng kiểm soát lỗi tự động. Tốc độ truyền lớn vì có đến 64 byte trong một yêu cầu (request) có thể được truyền.
- **Truyền ngắn:** các thiết bị, cung cấp một lượng dữ liệu nhỏ, tuần hoàn chẵng hạn như chuột, bàn phím đều sử dụng kiểu truyền này. Hệ thống sẽ hỏi theo chu kỳ, chẵng hạn 10ms một lần xem có các dữ liệu mới gửi đến.



# Các kiểu truyền

- **Truyền theo khối:** khi có lượng dữ liệu lớn cần truyền và cần kiểm soát lỗi truyền nhưng lại không có yêu cầu thúc ép về thời gian truyền thì dữ liệu thường được truyền theo khối. VD: máy in, máy quét
- **Truyền đẳng thời:** khi có khối lượng dữ liệu lớn với tốc độ dữ liệu đã được quy định, ví dụ như card âm thanh. Theo cách truyền này một giá trị tốc độ xác định được duy trì. Việc hiệu chỉnh lỗi không được thực hiện vì những lỗi truyền lẻ tẻ cũng không gây ảnh hưởng đáng kể.



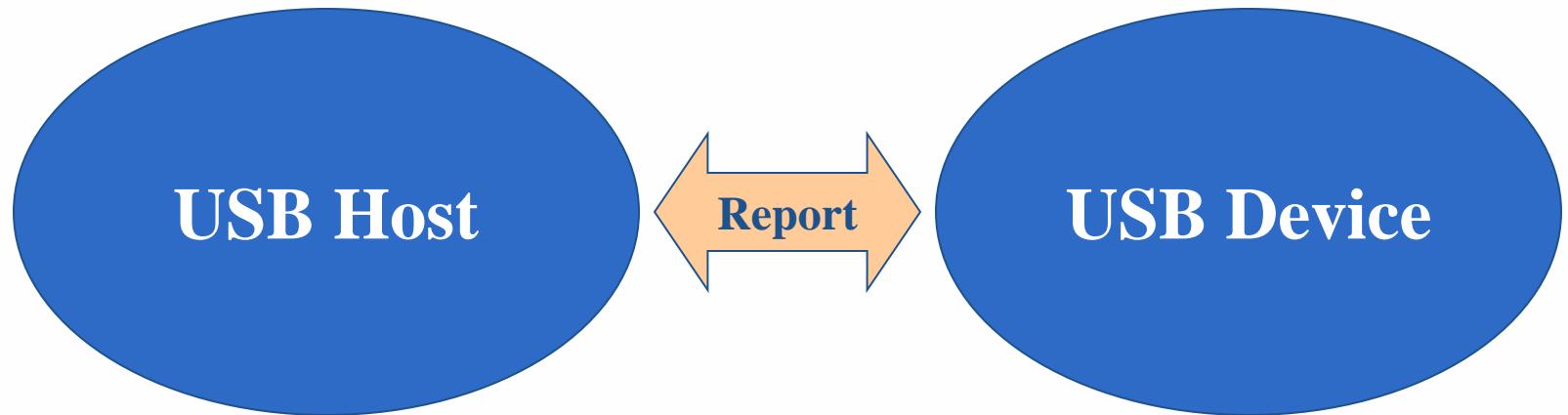
# Giới thiệu USB HID Class

- Mô hình truyền thông giữa máy tính và thiết bị theo chuẩn USB
- Giới thiệu HID class
- Lập trình firmware cho HID class





# Mô hình truyền thông



- Quá trình trao đổi dữ liệu giữa USB Host và USB Device thông qua các Report
- Lập trình: gửi, nhận, xử lý các report



# Giới thiệu HID class

- HID-Human Interface Device
- Một số thiết bị phổ biến theo HID class
  - Bàn phím
  - Chuột
  - Joystick (tay chơi game)
  - Điều khiển từ xa
  - Máy đọc mã vạch
  - ...





# Giới thiệu HID class

- Mỗi giao tiếp HID (HID interface) được phép có tối đa:
  - 1 interrupt IN endpoint (Bắt buộc)
  - 1 interrupt OUT endpoint (Optional)
- Tốc độ truyền (interrupt)
  - Low speed: 800 bytes/sec
  - Full speed: 64 kbytes/sec
  - High speed: 24 MBs/sec





# Giới thiệu HID class

- Các yêu cầu về phần cứng để cho một thiết bị theo chuẩn HID:
  - Endpoints
    - ✓ Chỉ sử dụng kiểu truyền Control và Interrupt
    - ✓ Bắt buộc có 1 interrupt IN endpoint
  - Reports
    - ✓ Phải có ít nhất một Input Report được định nghĩa trong HID Report Descriptor
  - Control transfers
    - ✓ HID hỗ trợ 6 kiểu gói tin yêu cầu: Set\_Report, Get\_Report, Set\_Idle, Get\_Idle, Set\_Protocol, Get\_Protocol



# HID Transfer Type

Transfer Type	Source of Data	Typical Data	Required Pipe?	Windows Support
Control	Device (IN transfer)	Data that doesn't have critical timing requirements.	yes	Windows 98 and later
	Host (OUT transfer)	Data that doesn't have critical timing requirements, or any data if there is no OUT interrupt pipe.		
Interrupt	Device (IN transfer)	Periodic or low-latency data.	yes	Windows 98 SE and later
	Host (OUT transfer)	Periodic or low-latency data.		



# Giới thiệu HID class

- Các yêu cầu về firmware để cho một thiết bị theo chuẩn HID:
  - Phải có Interface Descriptor theo HID
  - HID Descriptor
  - IN Endpoint Descriptor
  - Report Descriptor (Có thể là một trong ba loại: Input, Output, Feature)



# Device Descriptor

## // Device Descriptor

```
0x12, // Descriptor size in bytes  
0x01, // Descriptor type (Device)  
0x0200, // USB Specification release number (BCD) (2.00)  
0x00, // Class Code  
0x00, // Subclass code  
0x00, // Protocol code  
0x08, // Endpoint 0 maximum packet size  
0x0925, // Vendor ID (Lakeview Research)  
0x1234, // Product ID  
0x0100, // Device release number (BCD)  
0x01, // Manufacturer string index  
0x02, // Product string index  
0x00, // Device serial number string index  
0x01 // Number of configurations
```



# Configuration Descriptor

## // Configuration Descriptor

0x09, // Descriptor size in bytes

0x02, // Descriptor type (Configuration)

0x0029, // Total length of this and subordinate descriptors

0x01, // Number of interfaces in this configuration

0x01, // Index of this configuration

0x00, // Configuration string index

0xA0, // Attributes (bus powered, remote wakeup supported)

0x50, // Maximum power consumption (100 mA)



# Interface Descriptor

## // Interface Descriptor

```
0x09, // Descriptor size in bytes  
0x04, // Descriptor type (Interface)  
0x00, // Interface Number  
0x00, // Alternate Setting Number  
0x02, // Number of endpoints in this interface  
0x03, // Interface class (HID)  
0x00, // Interface subclass  
0x00, // Interface protocol  
0x00, // Interface string index
```



## // HID Descriptor

0x09, // Descriptor size in bytes

0x21, // Descriptor type (HID)

0x0110, // HID Spec. release number (BCD) (1.1)

0x00, // Country code

0x01, // Number of subordinate class descriptors

0x22, // Descriptor type (report)

002F, // Report descriptor size in bytes



# IN interrupt Endpoint Descriptor

## // IN Interrupt Endpoint Descriptor

0x07, // Descriptor size in bytes

0x05, // Descriptor type (Endpoint)

0x81, // Endpoint number and direction (1 IN)

0x03, // Transfer type (interrupt)

0x40, // Maximum packet size

0x0A, // Polling interval (milliseconds)



# OUT interrupt Endpoint Descriptor

## // OUT Interrupt Endpoint Descriptor

0x07, // Descriptor size in bytes

0x05, // Descriptor type (Endpoint)

0x01, // Endpoint number and direction (1 OUT)

0x03, // Transfer type (interrupt)

0x40, // Maximum packet size

0x0A // Polling interval (milliseconds)



# Lập trình firmware cho HID class

- **Bước 1:** Sử dụng một project mẫu
- **Bước 2:** Thêm/sửa mã nguồn phù hợp
  - Usbdesc.c: chỉnh sửa Endpoint, cấu trúc các report, thông tin sản phẩm....
  - Usbuser.c: chỉnh sửa các hàm xử lý sự kiện (gửi, nhận dữ liệu qua Endpoint)
  - Hiduser.c: chỉnh sửa các hàm đặc thù cho hid class
  - Demo.c: chỉnh sửa chương trình chính  
(Tham khảo Customize\_Keil\_USB\_Example.pdf)



A large, blue, five-pointed starburst shape is centered on the slide. Inside the starburst, the word "Demo" is written in a bold, red, sans-serif font.

Demo



- Lập trình cho thiết bị đóng vai trò như một bộ thu thập số liệu (nhiệt độ, độ ẩm, cường độ ánh sáng)
- Dữ liệu nhiệt độ được gửi về máy tính qua cổng USB định kỳ (VD: 100 ms)





# Demo HID class

TestUSB HID - ThuanPV

0	2	45	69
0	180	192	19
0	51	10	94
0	89	1	192
0	220	255	175
0	214	58	5
0	61	76	120
0	107	171	34
0	150	49	249
0	88	148	87
0	39	238	114
0	221	53	228
0	48	195	35
0	58	206	9
0	175	159	6
0	155	82	6
0	37	20	160
0	32	188	95
0	88	203	139
0	183	7	78

Nhiệt độ: 104   Độ ẩm: 100   Ánh sáng: 231



# Lợi ích của chuẩn USB

- Lợi ích cho người sử dụng

- Dễ sử dụng
  - ✓ Một giao tiếp dùng chung cho nhiều thiết bị ngoại vi khác nhau
  - ✓ Tự động cấu hình
  - ✓ Dễ dàng đấu nối
  - ✓ Hỗ trợ khả năng cắm nóng (Hot pluggable)
  - ✓ Thường không cần sử dụng nguồn ngoài
- Tốc độ cao và tin cậy: hỗ trợ nhiều tốc độ khác nhau (High speed: 480 Mbps, Full speed: 12 Mbps, Low speed: 1.5 Mbps)
- Giá thành phù hợp
- Tiết kiệm điện



# Lợi ích của chuẩn USB

- Lợi ích cho kỹ sư phát triển (thiết kế phần cứng, lập trình nhúng, lập trình ứng dụng)
  - Linh hoạt
    - ✓ Chuẩn USB hỗ trợ 4 kiểu truyền và 3 tốc độ khác nhau -> có thể phù hợp cho nhiều loại thiết bị ngoại vi.
    - ✓ Có thể hỗ trợ truyền các gói dữ liệu có ràng buộc/ không ràng buộc về thời gian -> tăng tính thời gian thực
    - ✓ Hỗ trợ giao thức để giao tiếp với các thiết bị chuẩn như máy in, bàn phím, ổ đĩa, đầu đọc thẻ ...



# Lợi ích của chuẩn USB

- Được hỗ trợ bởi hệ điều hành
  - Các hệ điều hành phổ biến đều hỗ trợ chuẩn USB: Windows, Linux, Macintosh
    - ✓ Phát hiện khi thiết bị được cắm vào hay rút ra khỏi hệ thống
    - ✓ Giao tiếp với thiết bị được cắm vào để tìm ra cách trao đổi dữ liệu
    - ✓ Hỗ trợ các giao diện hàm chuẩn (API) cho phép lập trình giao tiếp với thiết bị
- Được hỗ trợ bởi nhiều nhà sản xuất
  - Các chip chuyên dụng hỗ trợ giao tiếp theo chuẩn USB khá phổ biến và giá thành phù hợp



# Giới hạn của chuẩn USB

- Giới hạn về khoảng cách: giới hạn chiều dài cable là 5m
  - Có thể tăng khoảng cách bằng các mạch chuyển đổi (USB <-> RS485, ...)
- Giao tiếp peer-to-peer: mọi giao tiếp được thực hiện giữa Host computer và thiết bị ngoại vi
  - Đã phát triển thêm chuẩn USB-On-The-Go
- Không hỗ trợ việc truyền dữ liệu Broadcast (IEEE-1394 và Ethernet có hỗ trợ)



## 2.2.5. Chuẩn IEEE1394 (Firewire)

- Đối thủ cạnh tranh của USB
- Bus nối tiếp tốc độ cao thường được dùng để truyền tín hiệu video
- Chuẩn hỗ trợ tốc độ truyền tối đa lên đến 3.2Gbps
- Cho phép đấu nối tối đa 63 thiết bị
- Cho phép giao tiếp peer-to-peer





## Chương 3:Các thiết bị ngoại vi cơ bản

- Bàn phím
- Chuột
- Màn hình
- Máy in
- Đĩa cứng
- Đĩa quang
- Bộ nhớ trong
- Thẻ nhớ



# Bàn phím

## ■ Chức năng

- Thiết bị nhập thông tin vào cho máy tính xử lý
- Thông tin từ bàn phím là các ký tự, số và lệnh điều khiển

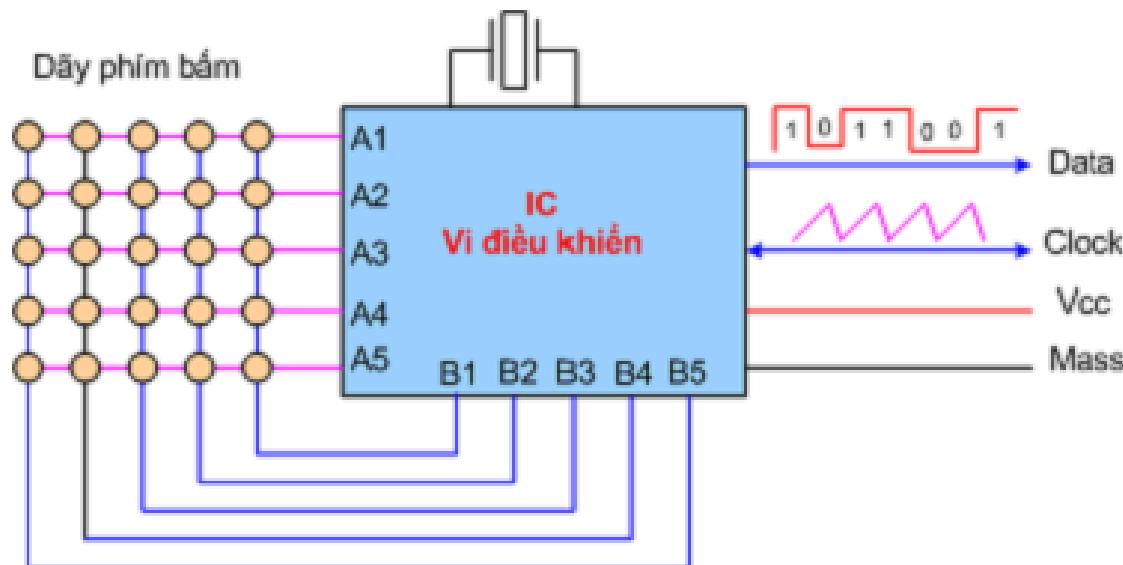




# Bàn phím

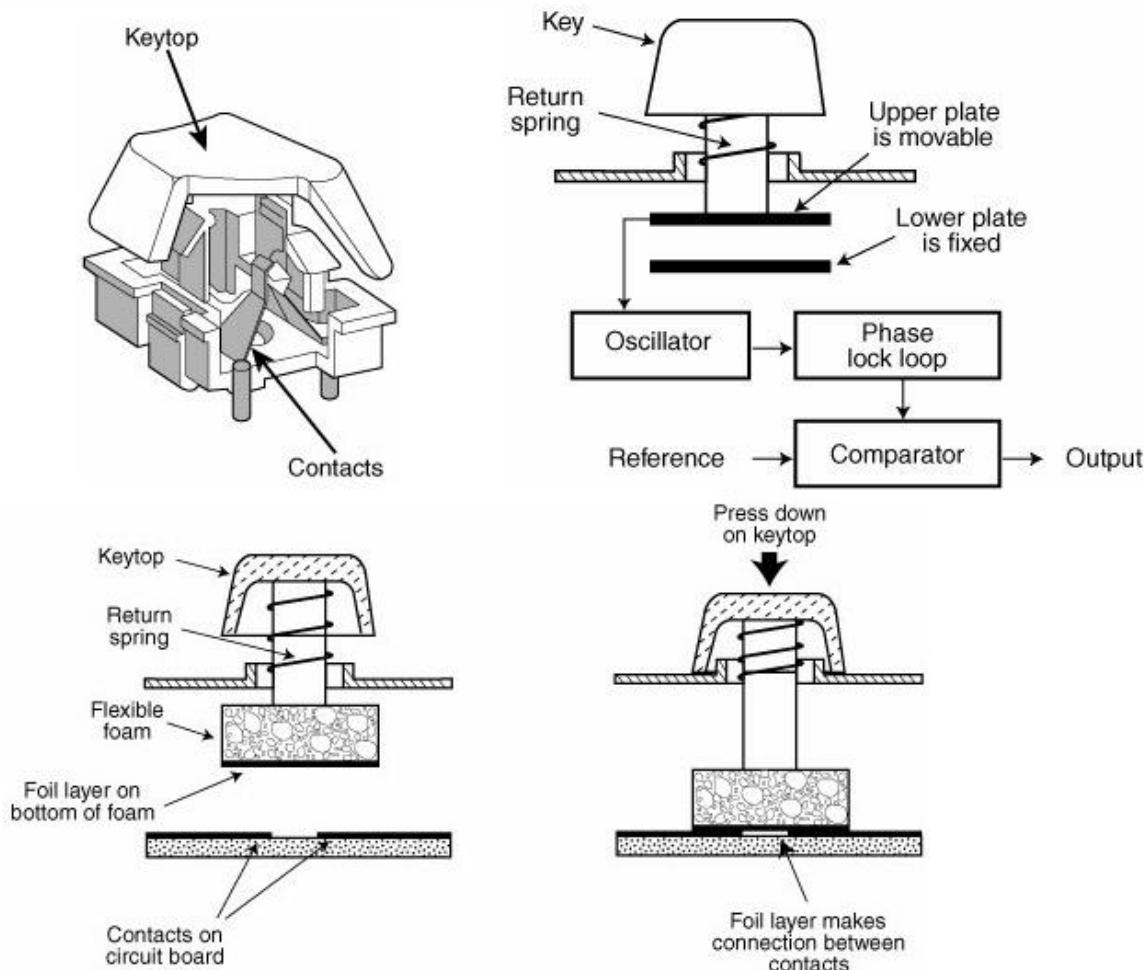
## ▪ Cấu tạo và hoạt động

- Mỗi phím bấm trên bàn phím tương ứng với một công tắc đấu chập một chân hàng a với một chân cột b
- Mỗi phím có một địa chỉ hàng và cột duy nhất



Sơ đồ mạch điện của bàn phím

## ■ Các loại công tắc





## ▪ Hoạt động

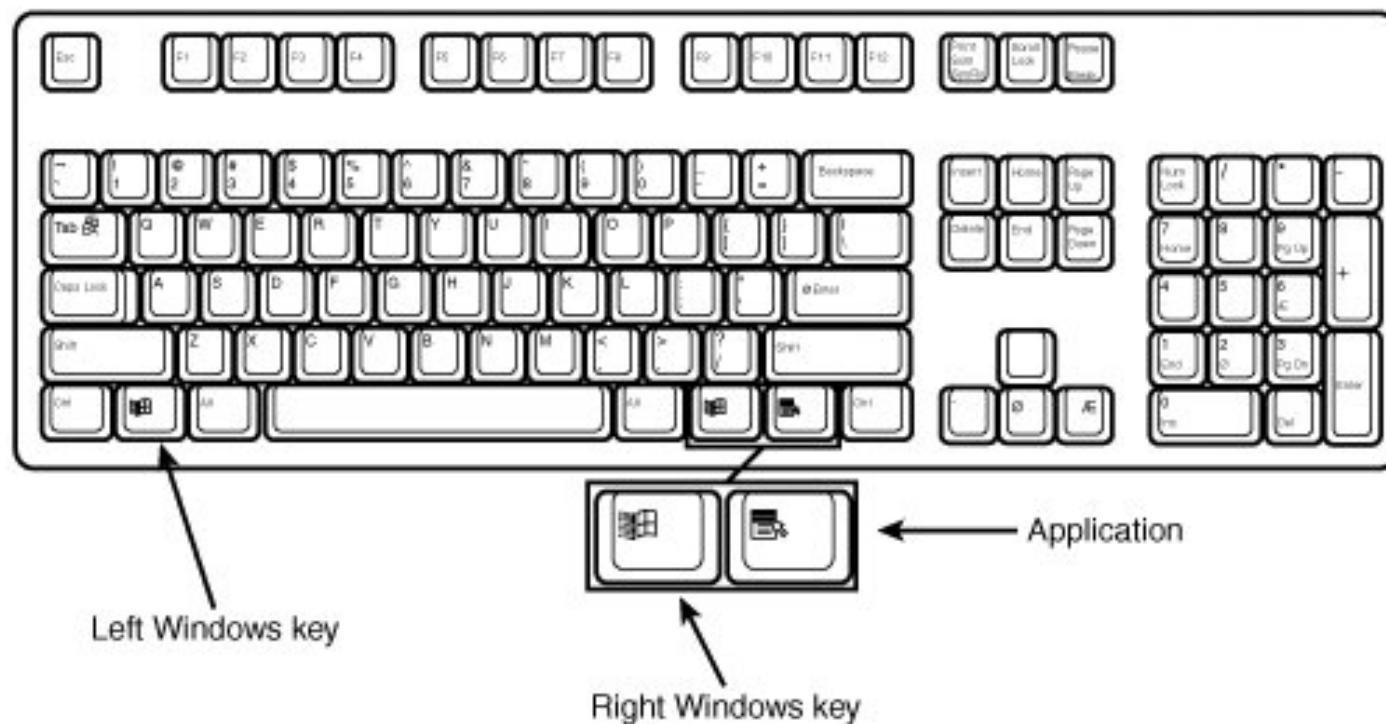
- Khi phím nhấn, bộ vi điều khiển sẽ xác định vị trí tọa độ của phím đó và gửi mã lên bộ xử lý trung tâm
- Với bàn phím cho máy cá nhân
  - ✓ Gửi mã make code
  - ✓ Khi người dùng nhả phím, gửi mã break code
  - ✓ Mã make code và mã break code theo tập mã cụ thể (thường là set 2)

<http://www.computer-engineering.org/ps2keyboard/scancodes2.html>



- Cấu trúc bàn phím điển hình

Figure 16.1. The 104-key Windows keyboard layout.





# Bàn phím

## ▪ Một số loại bàn phím

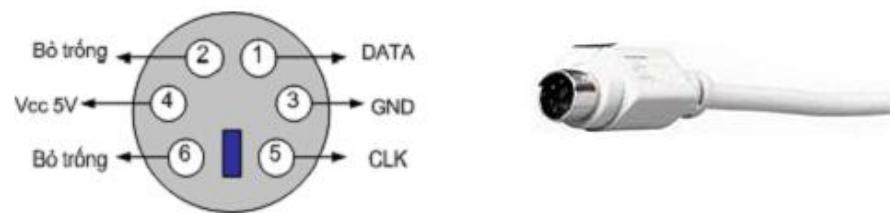
- Bàn phím cho Desktop
- Bàn phím cho Laptop + External keypad
- Bàn phím Multimedia





# Bàn phím

- Giao tiếp với máy tính
  - Chuẩn giao tiếp: nối tiếp
  - Jắc cắm

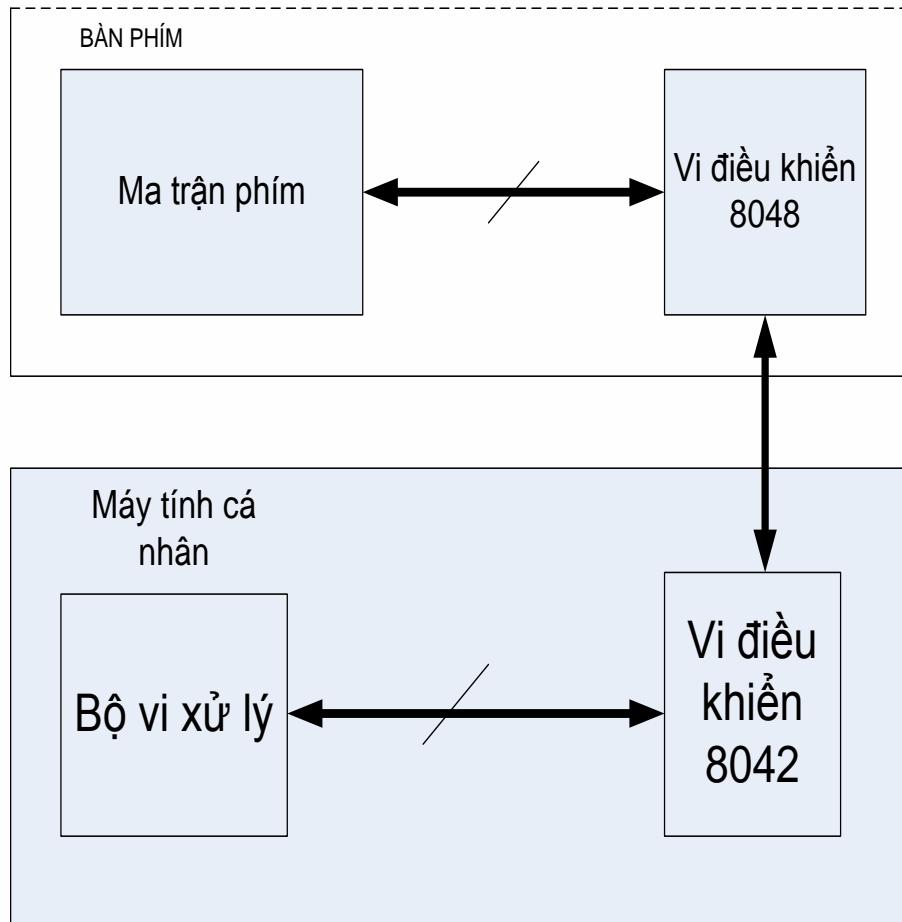


Trên bàn phím: Vi điều khiển 8048  
Trên máy tính: Vi điều khiển 8042



# Bàn phím

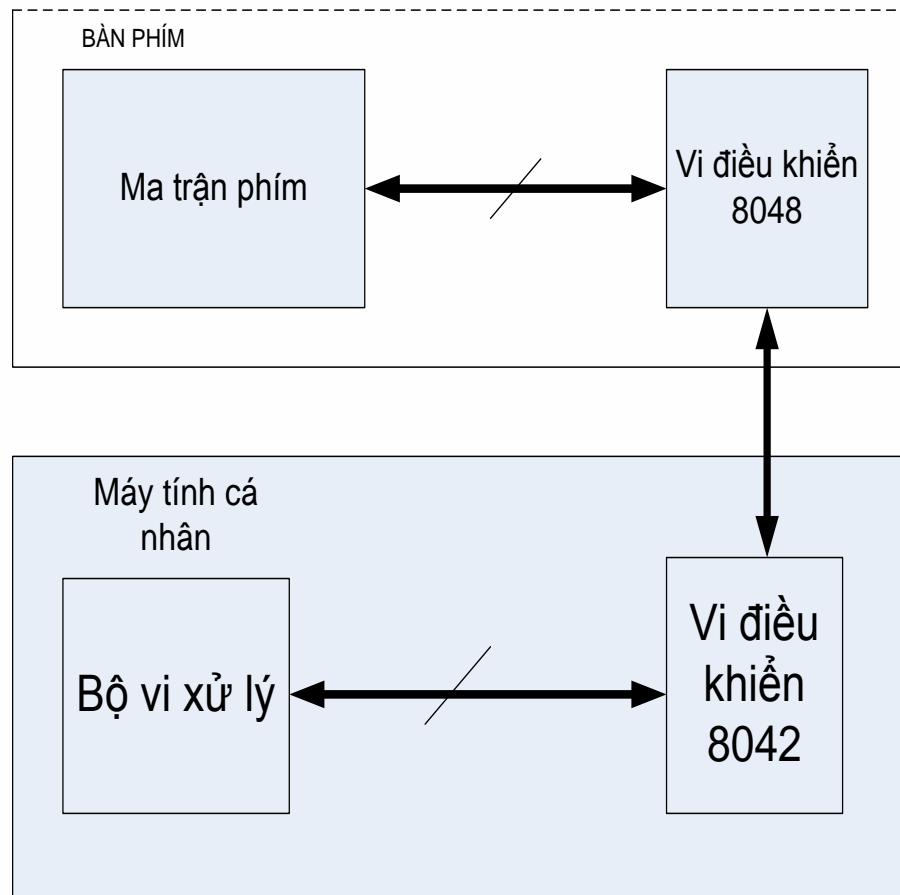
- 8048 có nhiệm vụ tính được vị trí nhấn, truyền tín hiệu từ bàn phím tới Máy tính.
- Khi 8048 phát hiện ra một phím được ấn, nó truyền cho 8042 mã yêu cầu để 8042 phát yêu cầu ngắt về bộ vi xử lý. Sau đó 8048 truyền tiếp mã quét (scan code) của phím
- 8042 nhận dữ liệu tuần tự từ bàn phím, dữ liệu được đưa lên cổng của 8255.





# Bàn phím

- 8042 gửi yêu cầu ngắt
- Bộ điều khiển ngắt 8259A sẽ xử lý yêu cầu ngắt và truyền số hiệu ngắt cho bộ vi xử lý.
- Bộ vi xử lý gọi chương trình xử lý ngắt. Chương trình này sẽ chuyển mã bàn phím thành mã ascii và đặt mã Ascii này vào bộ đệm trong RAM.





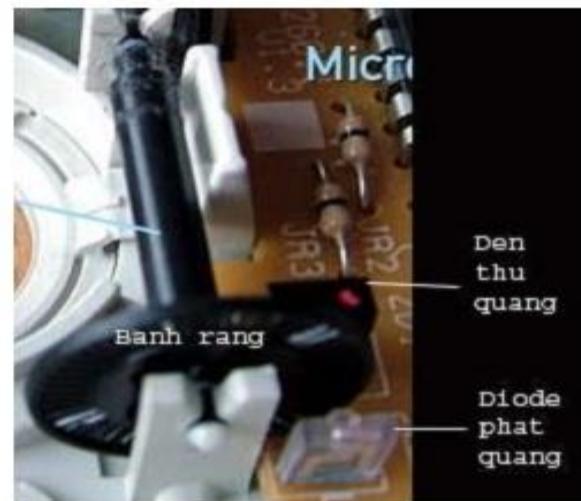
- Chuột:
  - Chuột là thiết bị vào cung cấp dữ liệu cho máy tính, dữ liệu là vận tốc tương đối của chuột khi nó được di chuyển trên 1 mặt phẳng.
  - Từ dữ liệu này máy tính tính ra vị trí của con trỏ (cursor) trên màn hình.
- Phân loại:
  - Chuột cơ
  - Chuột quang



# Chuột

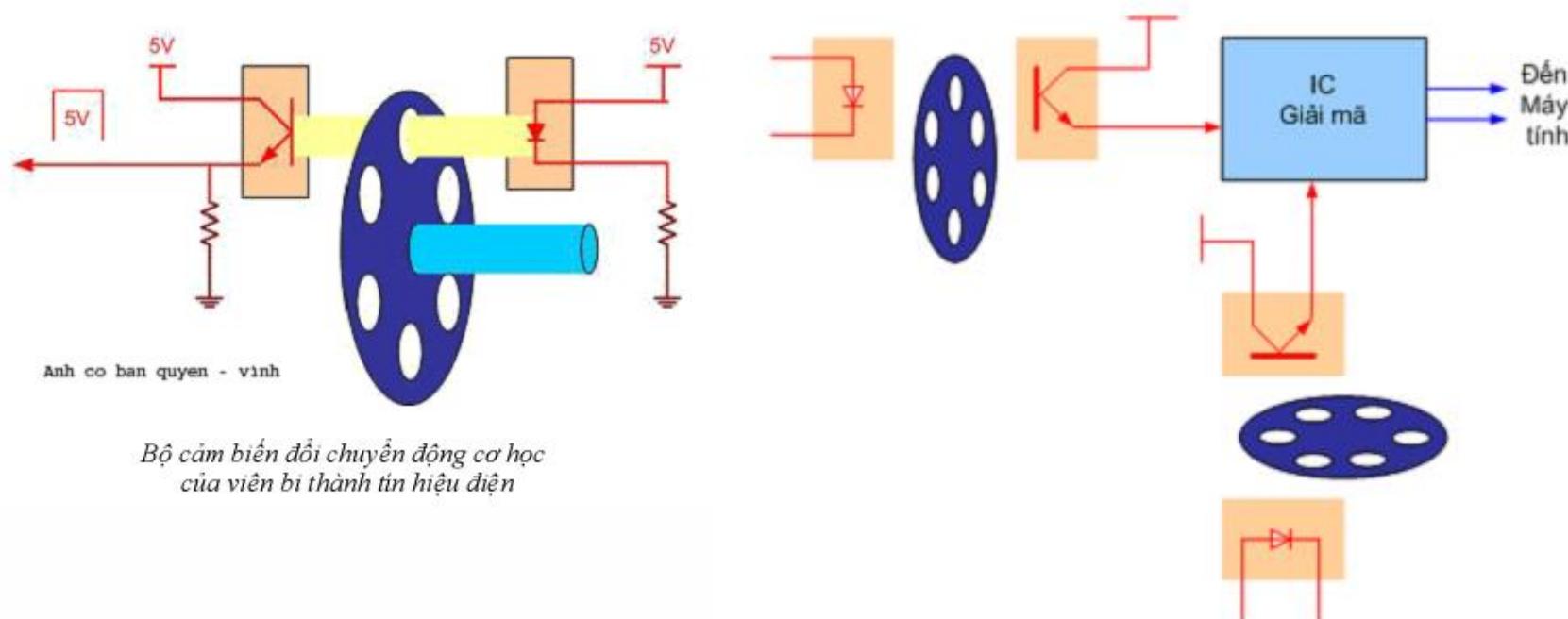
## ▪ Chuột cơ:

- Bên trong chuột có một viên bi cao su tỳ vào hai trục bằng nhựa được đặt vuông góc với nhau, khi ta di chuột thì viên bi quay làm cho 2 trục xoay theo.
- Hai trục nhựa được gắn vào bánh răng nhựa có đục lỗ, mỗi bánh răng được đặt lồng vào trong một cảm biến bao gồm một diode phát quang và một đèn thu quang



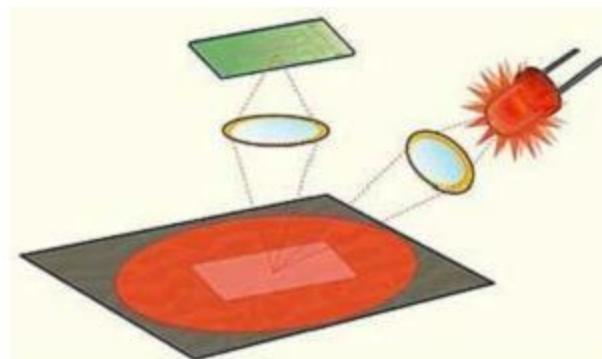
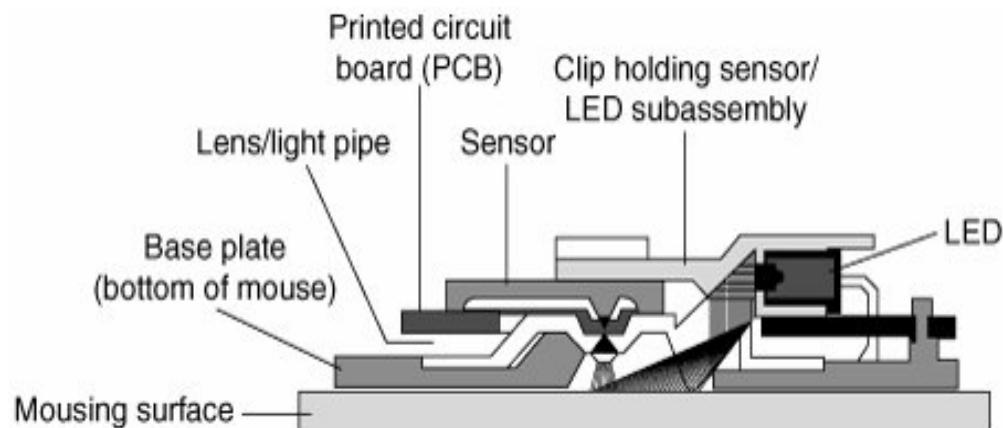


- Cấu tạo bên trong chuột cơ





## ▪ Chuột quang



Bộ phận quang học trong chuột quang

- Diode phát quang phát ra ánh sáng đỏ chiếu lên bề mặt của tấm di chuột, ảnh của bề mặt tấm di chuột được thấu kính hội tụ lên bề mặt của bộ phận cảm quang, bộ phận cảm quang sẽ phân tích sự di chuyển của bức ảnh và tạo thành tín hiệu điện gửi về máy tính.
- Bình thường đèn diode phát quang sẽ nhấp nháy liên tục, nhưng khi không có sự di chuyển, sau khoảng 3 giây, diode sẽ tự chuyển sang chế độ tối, tăng tuổi thọ của diode.



# Màn hình cảm ứng (touch screen)

- Màn hình cảm ứng:

- Dùng để điều khiển vị trí con trỏ và có công dụng như chuột.
- Dùng nhiều công nghệ khác nhau để xác định vị trí tác động trên màn hình:
  - ✓ Tia hồng ngoại
  - ✓ Màng cảm áp
  - ✓ Màng nhạy điện dung
  - ✓ Sóng siêu âm





# Màn hình cảm ứng

- Touch screen dùng tia hồng ngoại:
  - Dùng các nguồn tia hồng ngoại bố trí theo trục x và y của màn hình.
  - Đối diện các nguồn tia hồng ngoại là các điốt hay tế bào cảm quang.
  - Khi ngón tay di chuyển trên màn hình, nó sẽ ngắt nguồn tia hồng ngoại theo các trục x và y. Do đó tín hiệu tới các điốt/ tế bào cảm quang bị thay đổi. Căn cứ vào sự thay đổi đó, máy tính sẽ xác định được vị trí của vật tác động lên màn hình



## Màn hình cảm ứng

- Màn hình cảm ứng dùng Màng cảm áp:
  - Màng cảm áp được dán lên bề mặt màn hình
  - Lực tác động lên màn hình sẽ làm thay đổi điện trở cảm biến áp suất cáy trong màng này). Căn cứ vào sự thay đổi này mà máy tính xác định được vị trí của lực tác động.





# Màn hình cảm ứng

- Màn hình cảm ứng dùng Màng nhạy điện dung:
  - Màng trong suốt có công tắc điện dung.
  - Ngón tay (bút) đặt lên màn hình làm cho điện dung giữa hai điện cực tại vị trí ngón tay thay đổi. Căn cứ vào sự thay đổi này máy tính xác định được vị trí trên màn hình.



# Màn hình cảm ứng

- Màn hình cảm ứng dùng Sóng siêu âm:
  - Sóng siêu âm được phát song song bề mặt màn hình.
  - Ngón tay hay một vật bất kỳ chạm vào bề mặt màn hình khiến sóng siêu âm bị phản xạ trở lại. Bằng các đo thời gian phản xạ, bộ vi điều khiển tính được vị trí của vật tác động lên màn hình.





## Màn hình

- Màn hình tia âm cực CRT (cathode ray tube)
- Màn hình tinh thể lỏng LCD (liquid crystal display)
- Màn hình plasma PD (plasma display)



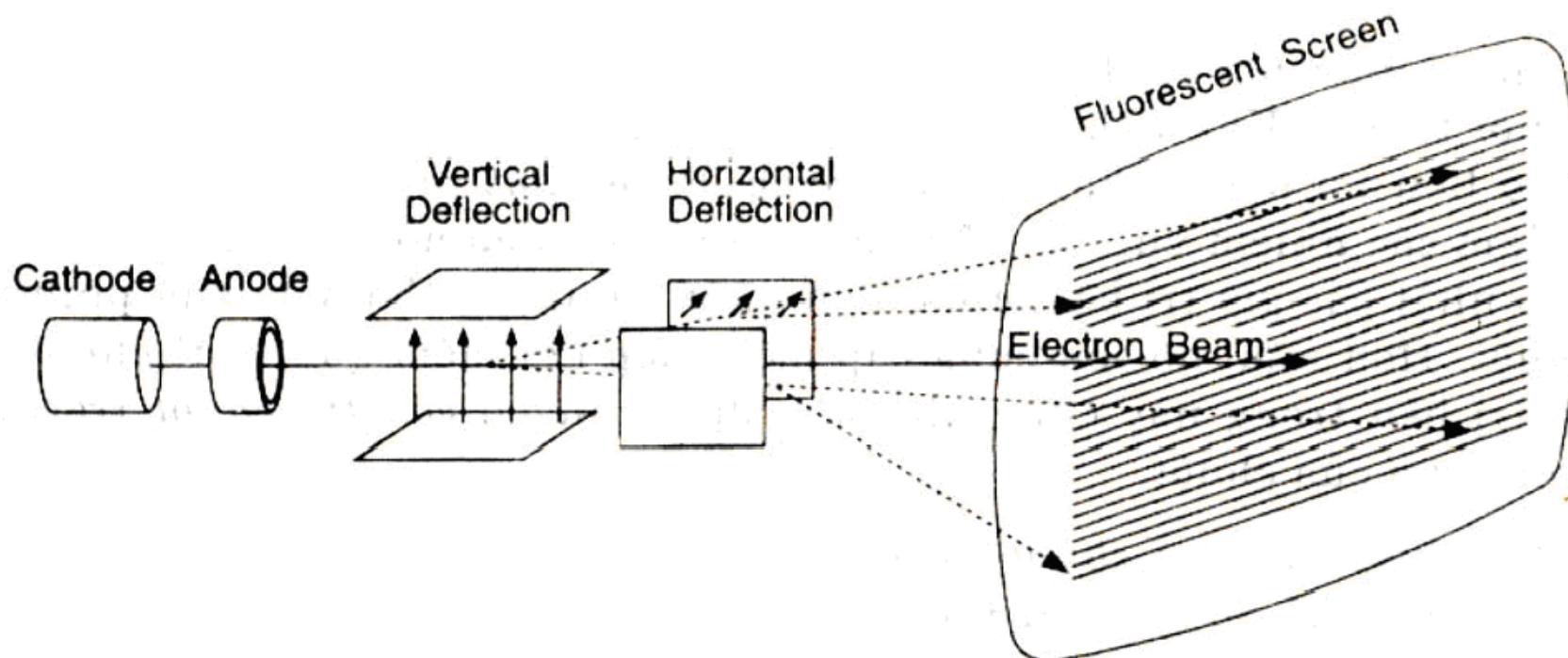


# Màn hình CRT

- Màn hình kiểu ống phóng tia âm cực CRT (cathode ray tube):
  - Các điện tử phát xạ từ katốt trong ống được hội tụ thành một chùm tia, sau đó được tăng tốc và được làm lệch hướng chuyển động bởi các bộ phận lái tia. Tia này sẽ đập vào màn hình có phủ chất huỳnh quang để tạo thành một điểm sáng gọi là *điểm ảnh*
  - Độ chói (sáng tối) được quyết định bởi cường độ chùm tia đập vào màn huỳnh quang
  - Một điểm màu tự nhiên được hiện nhờ sự trộn lẫn của 3 màu đỏ, xanh lá cây và xanh dương (RGB) theo một tỉ lệ nào đó. Ba màu này được hiện nhờ 3 tia điện tử cùng bắn vào 3 điểm trên màn hình kề cận nhau.



# Màn hình CRT



Cấu tạo ống hình CRT.

Cathode: *katốt*;

Anode: *anôt*;

Vertical Deflection: *lệch dọc*;

Horizontal Deflection: *lệch ngang*;

Fluorescent Screen: *màn phát quang*:

Electron Beam: *tia điện tử*.



- Tia điện tử được quét rất nhanh theo chiều ngang từ trái sang phải sẽ tạo nên một vệt sáng ngang được gọi là *dòng quét*.
- Đến cuối một dòng, nó được quét ngược trở lại về bên trái để quét tiếp dòng thứ hai bên dưới v.v.. Quá trình quét các dòng được dịch dần từ trên xuống dưới suốt chiều dọc của màn hình được gọi là **quét đọc** để tạo nên một **khung ảnh** gọi là **frame**



# Màn hình CRT

- Số khung (frame) tạo ra trong một giây được gọi là tốc độ khung, tốc độ quét dọc hay tốc độ làm tươi (refresh rate).
- Sau khi tia điện tử đi qua điểm phát sáng, cường độ sáng giảm dần, tốc độ giảm đi được gọi là **độ lưu ảnh**. Độ lưu ảnh cao có nghĩa là phải mất nhiều thời gian để cường độ phát sáng giảm đi hết.



# Màn hình CRT

- Nếu cường độ phát sáng giảm đi quá chậm, những hình ảnh chuyển động trên màn hình sẽ bị nhòe. Nếu cường độ phát sáng giảm đi quá nhanh, tốc độ làm tươi phải cao để tránh hiện tượng hình bị giật.
- Do hiện tượng lưu ảnh vĩnh mạc, để hình ảnh không “giật” thì tối thiểu phải phát 24 hình/s.
- Hiện nay các màn huỷnh quang sử dụng trong màn hình CRT có độ lưu ảnh thấp nên tốc độ làm tươi thường trên 70Hz.



## ▪ **Quét xen kẽ (interlaced):**

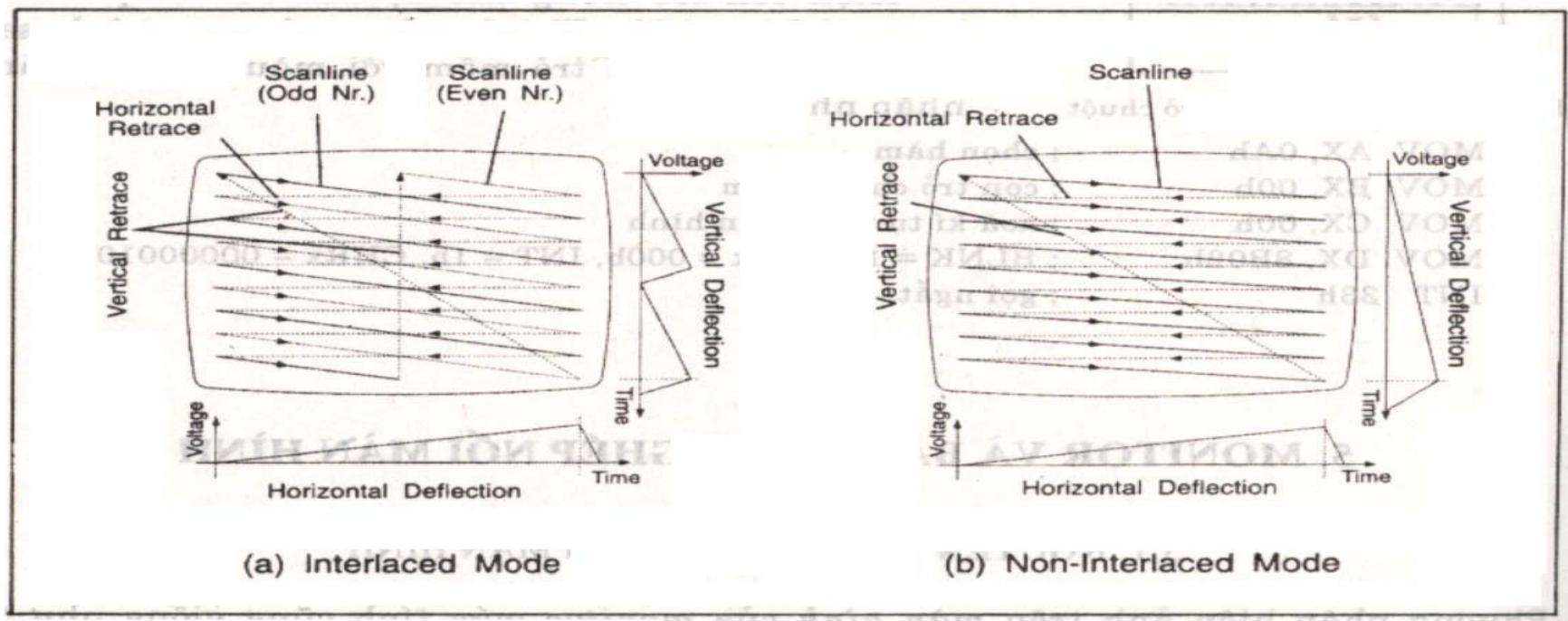
- Các dòng lẻ được quét trước cho đến hết màn hình theo chiều dọc, gọi là *frame lẻ*; sau đó các dòng chẵn tạo nên *frame chẵn* được quét sau. Phương pháp này có ưu điểm là thu hẹp được dải tần số làm việc của thiết bị nhưng có nhược điểm là hình ảnh bị nhấp nháy.

## ▪ **Quét không xen kẽ (non-interlaced):**

- Các dòng quét được thực hiện tuần tự. Ưu điểm là hình ảnh có thể được điều chỉnh chính xác và ổn định nhưng thiết kế mạch điện sẽ khó hơn vì phải giải quyết vấn đề tăng dải tần làm việc.

# Màn hình CRT

- Có 2 kiểu quét tia điện tử như hình sau:



Vertical Retrace: đường quét ngược dọc;  
Scanline (Odd Nr.): đường quét lẻ;  
Voltage: thế hiệu;  
Horizontal Deflect: lệch ngang;

Hai kiểu quét tia điện tử.

Horizontal Retrace: đường quét ngược ngang;  
Scanline (Even nr.): đường quét chẵn;  
Time: thời gian.  
Vertical Deflection: lệch dọc



# Màn hình LCD

- Tinh thể lỏng (liquid crystal) là chất lỏng hữu cơ mà phân tử của nó có khả năng phân cực ánh sáng dẫn đến thay đổi cường độ sáng. Trường tĩnh điện được dùng để điều khiển hướng phân tử tinh thể lỏng.
- Thay vì các điểm ảnh riêng biệt là các phần tử phát sáng được định địa chỉ một cách tuần tự. Do vậy, trên các monitor này hình ảnh cũng được phát ra từng dòng một. Quá trình quét ngược cũng không còn nữa vì ở đây đơn giản chỉ là việc thay đổi địa chỉ về phần tử đầu dòng tiếp theo

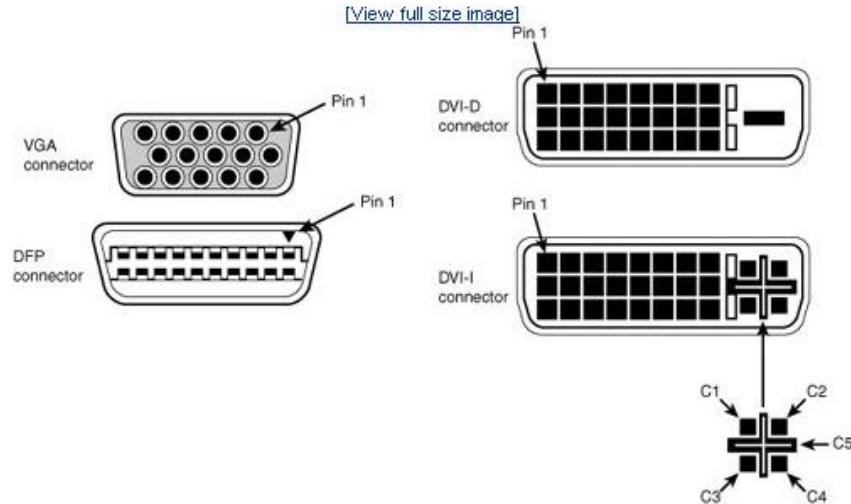


# Màn hình Plasma

- Nguyên tắc màn hình Plasma giống như nguyên tắc đèn neon.
- Màn hình plasma gồm nhiều ô khí trơ được hàn kín tương ứng với một điểm ảnh.
- Mỗi ô khí trơ có hai điện cực. Khi hiệu điện thế vượt qua giới hạn nhất định, khí trơ sẽ ion hóa và phát sáng.
- Nguyên tắc điều khiển của màn hình này đơn giản hơn LCD nhưng tiêu thụ nhiều năng lượng hơn.

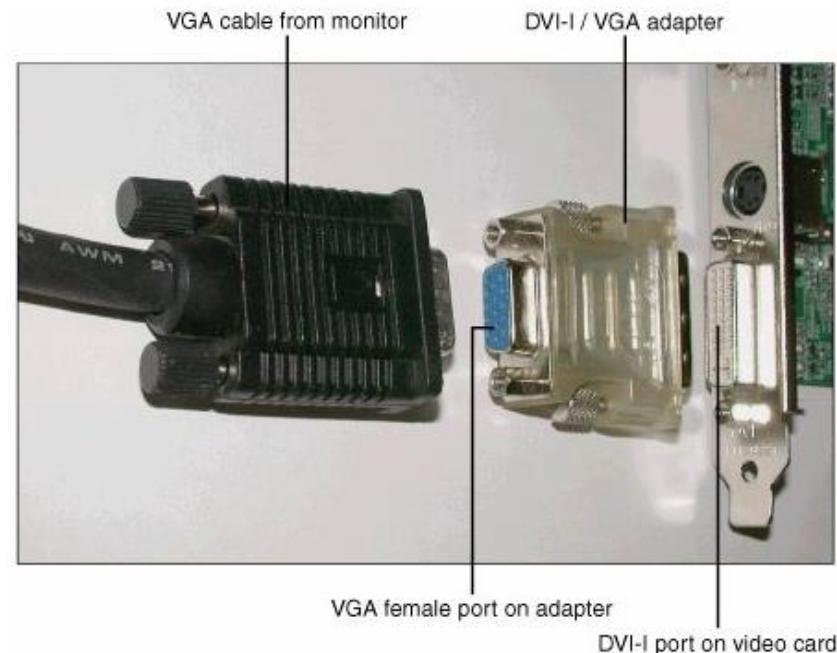


# Giao tiếp màn hình & máy tính



Jắc cắm VGA và DVI

Bộ chuyển đổi VGA - DVI





- Lập trình cho màn hình :

- INT 21h : Gọi CTC phục vụ ngắt 21h (DOS)
- AH : Hàm chức năng (AH=2 : in 1 kí tự, AH=9 : viết ra 1 xâu, ...)
- Hàm AH=2 của INT 21h là hàm hiển thị 1 kí tự có mã ASCII ở trong thanh ghi DL ra màn hình. VD: Sau khi thực hiện 3 lệnh sau, trên màn hình sẽ xuất hiện chữ cái A (mã ASCII là 65):

MOV AH, 2	; Dịch vụ số hiệu 2 của ngắt 21h - Hiện kí tự
MOV DL, 65	; Mã ASCII của kí tự cần hiện - 65 (41h) là A
INT 21h	; Gọi CTC phục vụ ngắt số hiệu 21h



# Màn hình

- VD1. Hãy cho biết kết quả hiển thị ra màn hình sau khi thực hiện đoạn lệnh sau:

MOV CL, 2

MOV AH, 2

MOV AL, 45h

LAP:

INC AL

MOV CH, 3

MOV DL, AL

QUAY:

INT 21h

DEC CH

JNZ QUAY

DEC CL

JNE LAP

RA:

....-> Kết quả in ra màn hình: FFFGGG



# Máy in

- Máy in có 3 loại
  - Máy in kim
  - Máy in phun
  - Máy in Laser





## ▪ Nguyên tắc chung

- Tia laser chỉ có vai trò là 1 tia sáng mảnh, cường độ lớn, có thể chiếu lên bề mặt thành 1 điểm sáng nhỏ, kích thước vài micrômet và có thể điều khiển tia laser viết, vẽ lên bề mặt như một ngòi bút ánh sáng.
- Bộ phận rất quan trọng ở máy in laser là một hình trụ bằng kim loại nhẹ, bên ngoài có phủ 1 lớp vật liệu đặc biệt gọi là vật liệu quang dẫn hay đơn giản hơn gọi là cái trống.



## Máy in laser (tiếp)

- Trống luôn được đặt vào một nơi tối, tức là bên trong vỏ kín của máy in. Giả sử bằng một cách nào đó ta tích điện dương cho mặt trên của trống tức là làm cho phía trên của lớp quang dẫn có điện tích dương. Lớp quang dẫn đang ở trong tối nên là vật liệu cách điện, mặt trên có điện tích dương thì ở mặt dưới có điện tích âm.
- Nếu chiếu tia laser lên mặt trống, chỗ được chiếu sáng sẽ trở thành dẫn điện (đó là tính chất của vật liệu quang dẫn) qua đó điện tích dương thoát đi, chỗ được chiếu sáng trở thành có điện tích âm như là ở phía dưới.



## Máy in laser (tiếp)

- Khi điều khiển để tia laser vẽ nên chữ gì hình gì lên mặt trống thì phải do hiện tượng quang dẫn như đã nói trên, ở trên mặt trống sẽ có chữ, có hình như ta đã vẽ, tuy nhiên đây là chữ, hình điện tích âm, không nhìn thấy được người ta gọi là ảnh ẩn điện.
- Nếu lấy một cài ru lô có các hạt mực mang điện tích dương lăn lên trống, những chỗ có ẩn ảnh điện sẽ hút các hạt mực vì điện trái dấu hút nhau. Còn những chỗ trên trống không được chiếu sáng vẫn còn nguyên điện tích dương, nên đẩy các hạt mực ra, vì điện tích cùng dấu đẩy nhau. Cuối cùng nếu cho 1 tờ giấy lăn qua trống mực bị hút dính ở trống sẽ chuyển qua dính lên giấy, đặc biệt là khi giấy được tích một ít điện âm.



## Máy in Laser (tiếp)

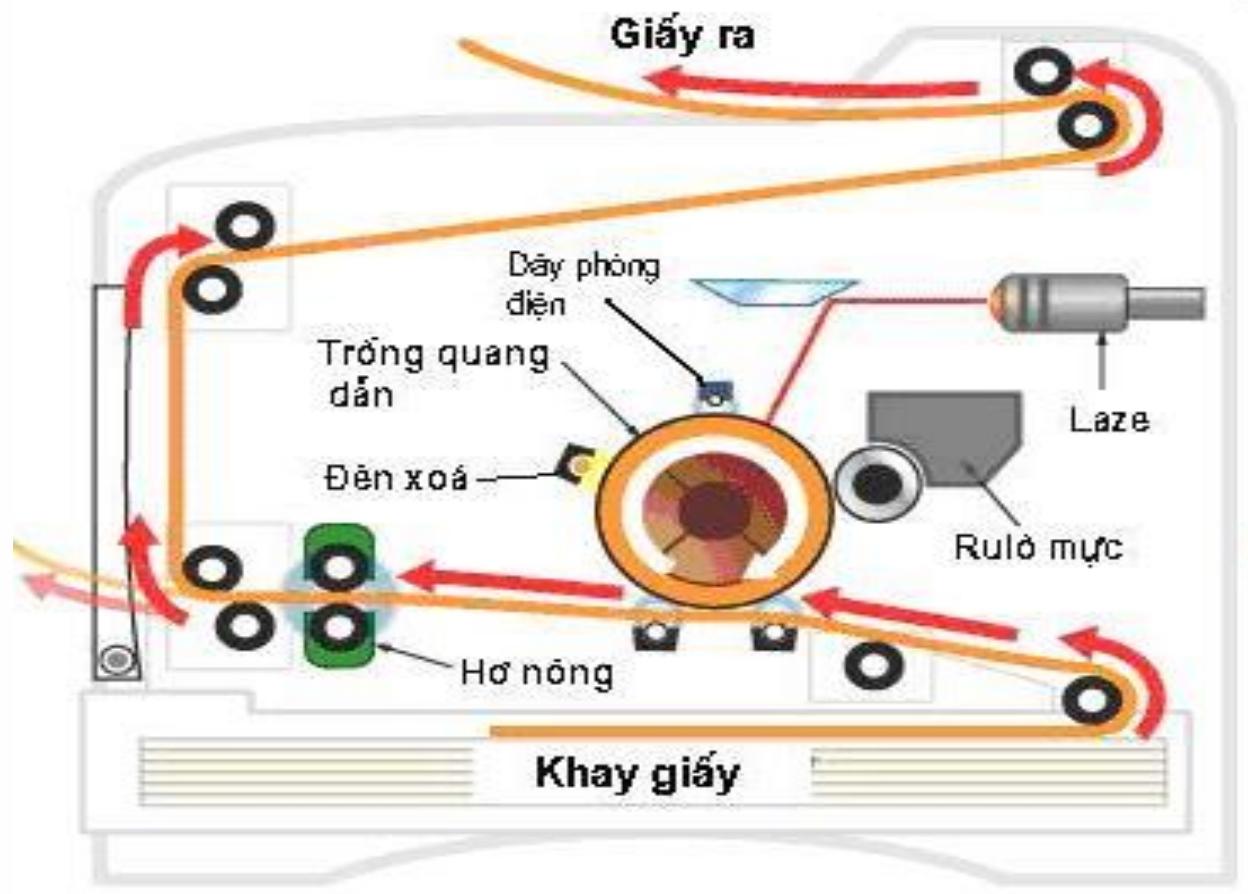
- Khi các hạt mực đã sơ bộ bám vào giấy sau khi lăn qua trống, người ta cho giấy đi qua chõ sưởi nóng (cõ ) và ép. các hạt chất dẻo hơi chảy ra mực sẽ dính chặt vào giấy.





# Máy in laser (tiếp)

## ■ Cấu tạo và hoạt động





# Máy in laser (tiếp)

## ▪ Hoạt động

- Dây phóng điện hào quang làm cho mặt trống ở dưới đó tích điện dương.
- Khi quay mặt trống tích điện dương quay đến chỗ có tia laser chiếu vào, nhờ máy tích điều khiển, tia laser viết, vẽ từng hàng trên mặt trống, tạo ra ảnh ẩn mang điện tích âm.
- Mặt trống quay đến chỗ có ru lô mang hạt mực điện tích dương. Vì ảnh ẩn trên trống mang điện tích âm nên hút các hạt mực mang điện tích dương, ảnh ẩn trở thành ảnh có các hạt mực trên trống.



## Máy in laser (tiếp)

- Giấy ở khay sau khi được tích điện âm chạy qua áp vào mặt trống. Các hạt mực ở trống bị hút lên giấy.
- Giấy được đưa qua chỗ sưởi nóng, ép các hạt mực nóng chảy, dính chặt với giấy. Mực đã bám chắc sau đó giấy được đưa ra ngoài.
- Mặt trống được đèn chiếu sáng, xoá hết điện tích còn lưu lại trên mặt trống, có cái gạt để giả sử còn ít hạt mực sót lại trên trống mực bị gạt ra. Mặt trống xem như được lau sạch, chuẩn bị để chạy qua dây phóng điện hào quang, tích điện dương cho mặt trống, tiếp tục quá trình.



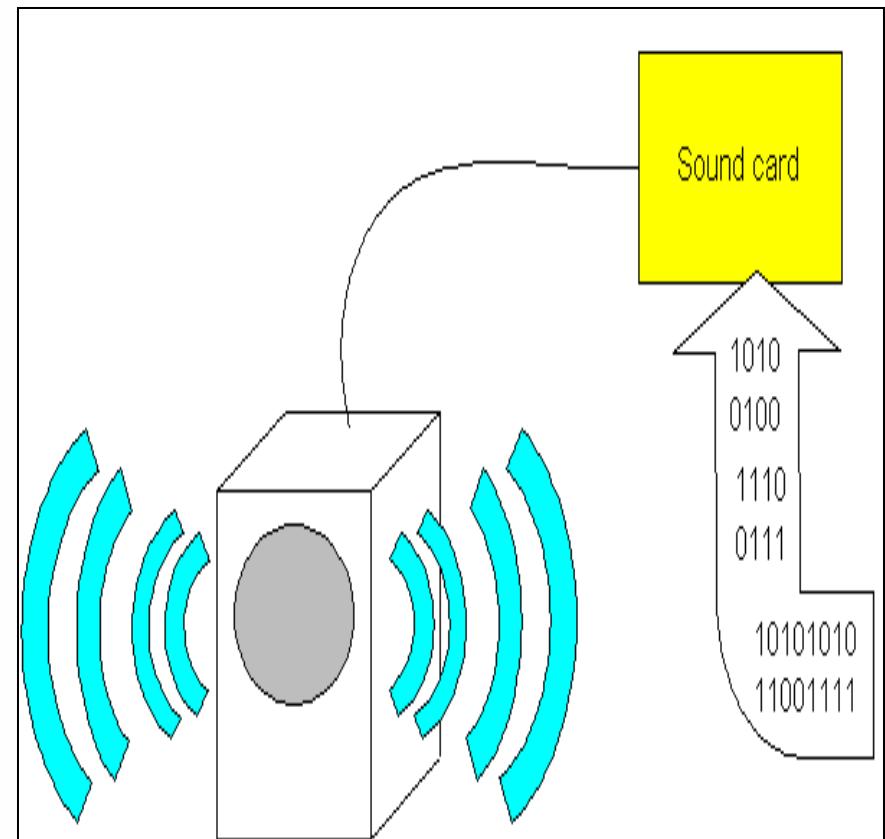
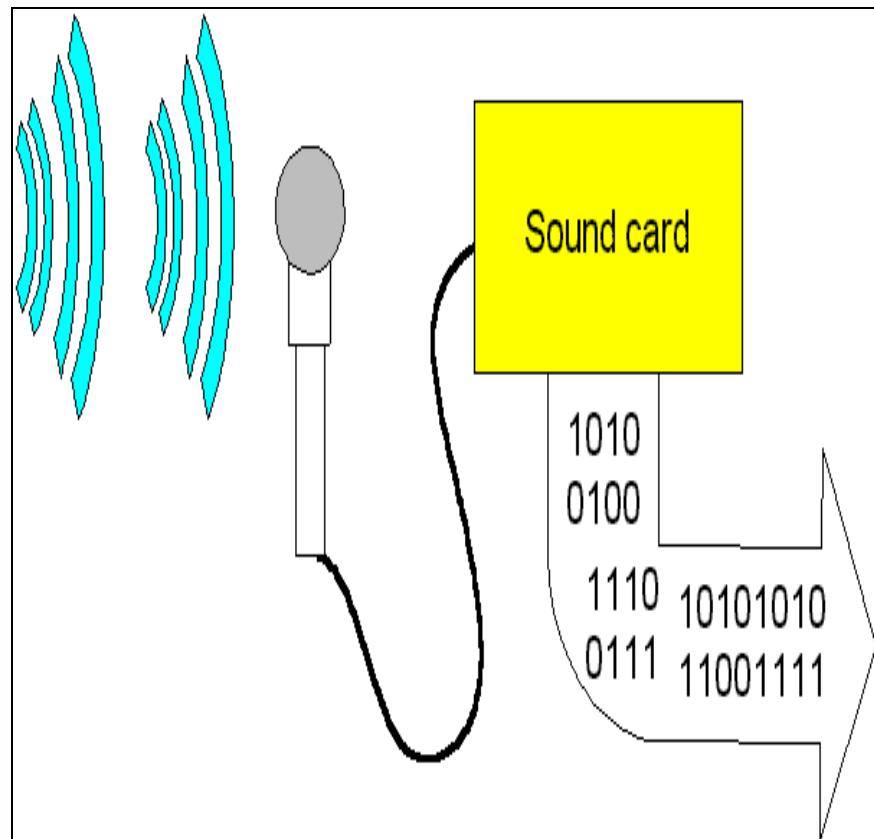
# Card âm thanh (Card Sound)

## ▪ Card âm thanh:

- Là thành phần nhận, xử lý và phát âm thanh. Một số mainboard cũng tích hợp sẵn card này.
- Những nhiệm vụ chính của card âm thanh là:
  - ✓ Nhận tín hiệu âm thanh
  - ✓ Lưu trữ và xử lý tín hiệu âm thanh
  - ✓ Phát âm thanh
  - ✓ Trao đổi tín hiệu âm thanh với các thiết bị khác



# Card âm thanh (Card Sound)





# Card âm thanh (Card Sound)

- Card âm thanh gồm những thành phần chính :
  - Mạch chuyển đổi tín hiệu tương tự sang tín hiệu số (ADC) và từ tín hiệu số sang tín hiệu tương tự (DAC)
  - Bộ phận xử lý tín hiệu số : xử lý và tổng hợp âm thanh
- Card âm thanh được kết nối với máy tính qua bus ISA, PCI hoặc có thể hay tích hợp ngay trong MainBoard.



# Card âm thanh (Card Sound)

- Lấy mẫu âm thanh là quá trình chuyển đổi tín hiệu điện từ micro thành tín hiệu số nhờ bộ ADC.
- Tín hiệu âm thanh được micro chuyển thành tín hiệu điện . Một mẫu âm thanh (sample sound) là một giá trị nhị phân đại diện cho biên độ âm thanh tại một thời điểm nhất định.
- Để có thể khôi phục lại chính xác tín hiệu âm thanh thì tần số lấy mẫu phải lớn hơn 2 lần tần số cực đại của tín hiệu âm thanh

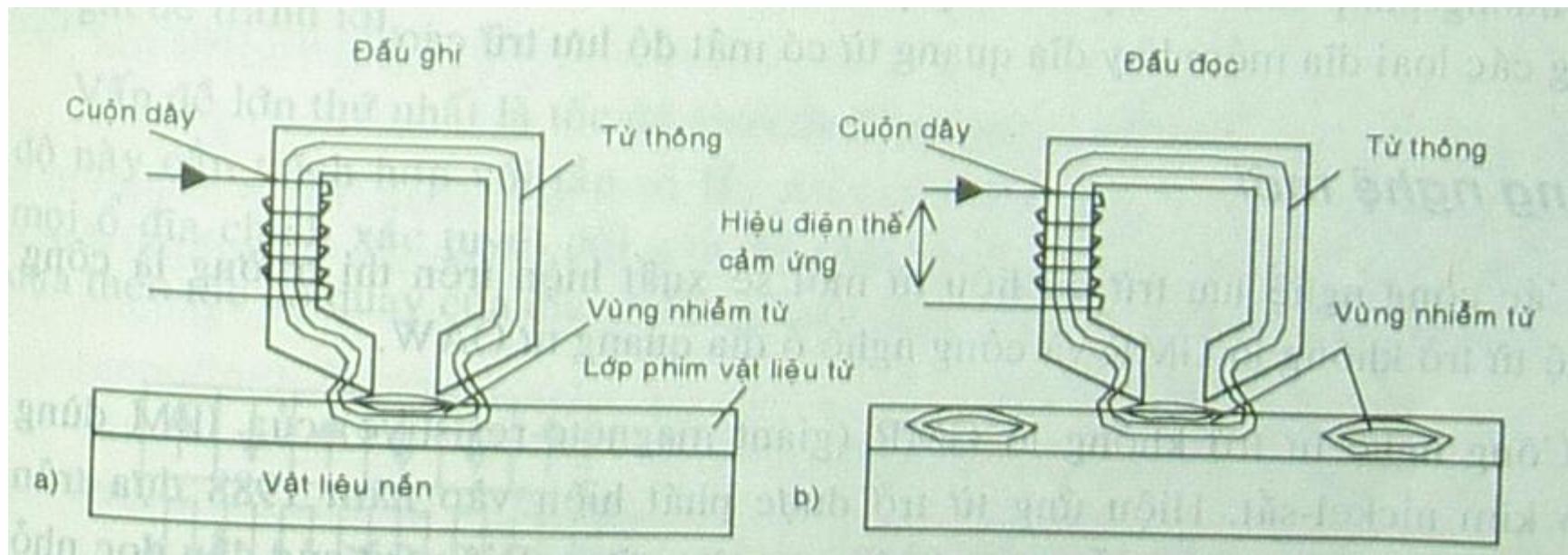


# Card âm thanh (Card Sound)

- Dải tần số của âm thanh mà tai con người có thể cảm nhận được là: 20 Hz – 20000Hz
- Tần số lấy mẫu phổ biến trong các card sound là: 41 000Hz.
- Tần số cao:tối bộ nhớ, chất lượng âm thanh tốt.



- Nguyên tắc lưu trữ thông tin trên vật liệu từ:
  - Thông tin là một chuỗi các phần tử nhiễm từ.
  - Trạng thái của một bit (0,1) được xác định dựa vào hướng từ trường của từng phần tử.
  - Ghi dữ liệu:





## ▪ Ghi dữ liệu lên vật liệu từ:

- Khi dòng điện chạy qua dây dẫn của đầu ghi, từ trường xuất hiện trong khung cảm từ khiến từ trường của các phần tử nhiễm từ có hướng nhất định.
- Chiều của từ trường phụ thuộc và chiều dòng điện.
- Bằng cách thay đổi chiều dòng điện ta có thể thay đổi hướng từ trường của từng phần tử



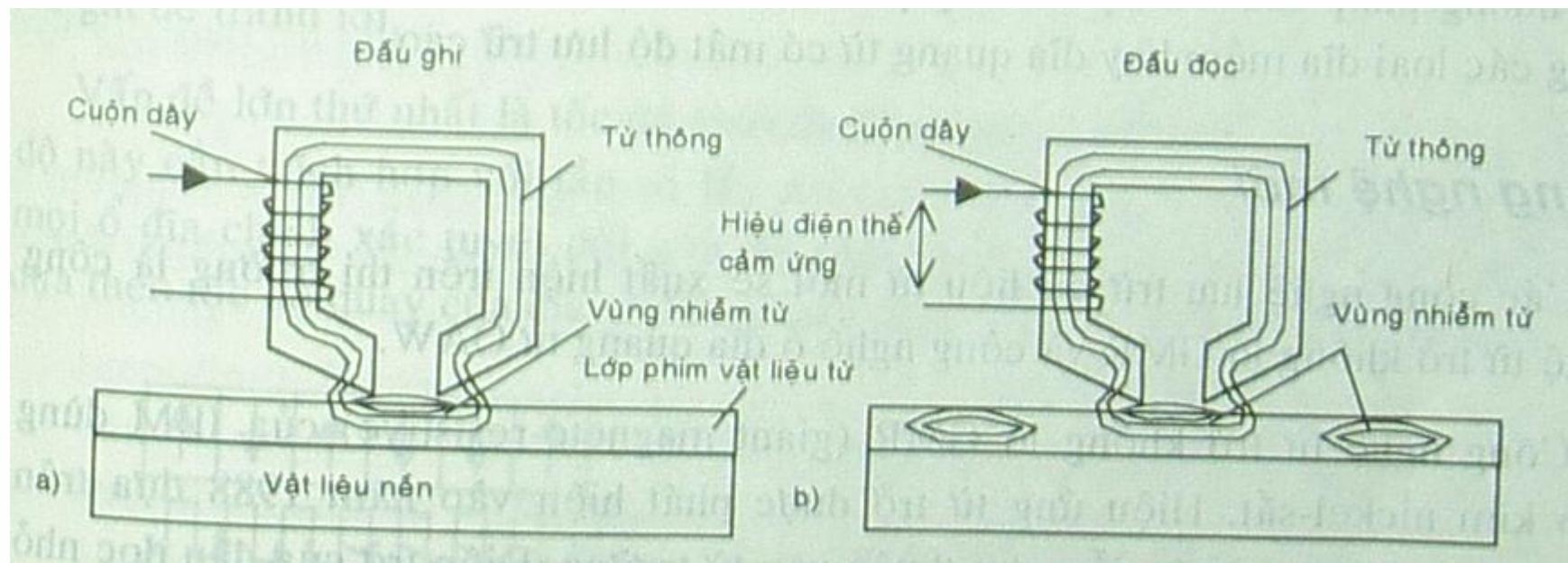
- Đọc dữ liệu trên vật liệu từ:

- Khi đầu đọc chạy qua một vùng nhiễm từ, từ trường xuất hiện trong khung gây ra dòng điện cảm ứng trong cuộn dây của đầu đọc.
- Chiều của dòng điện phụ thuộc vào chiều của từ trường của các phần tử nhiễm từ.
- Hiệu điện thế cảm ứng phụ thuộc vào chiều dòng điện ở cuộn dây => phân biệt được các phần tử nhiễm từ. (~phân biệt được các bit 0,1).



# Đĩa cứng

## b) Đọc dữ liệu





## ▪ Đĩa cứng

- Là thiết bị điện tử dùng để lưu trữ thông tin.
- Ổ đĩa cứng được sử dụng để lưu trữ hệ điều hành, các chương trình ứng dụng và dữ liệu.
- Trên ổ đĩa cứng có gắn các động cơ và các mạch điều khiển đĩa tạo thành một khối
- Bao gồm nhiều tấm đĩa kim loại được xếp đồng trực chồng lên nhau nằm trong một vỏ bọc kim loại.

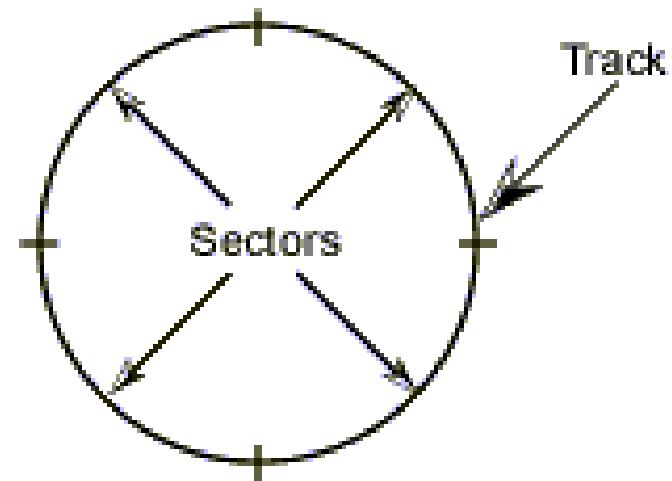
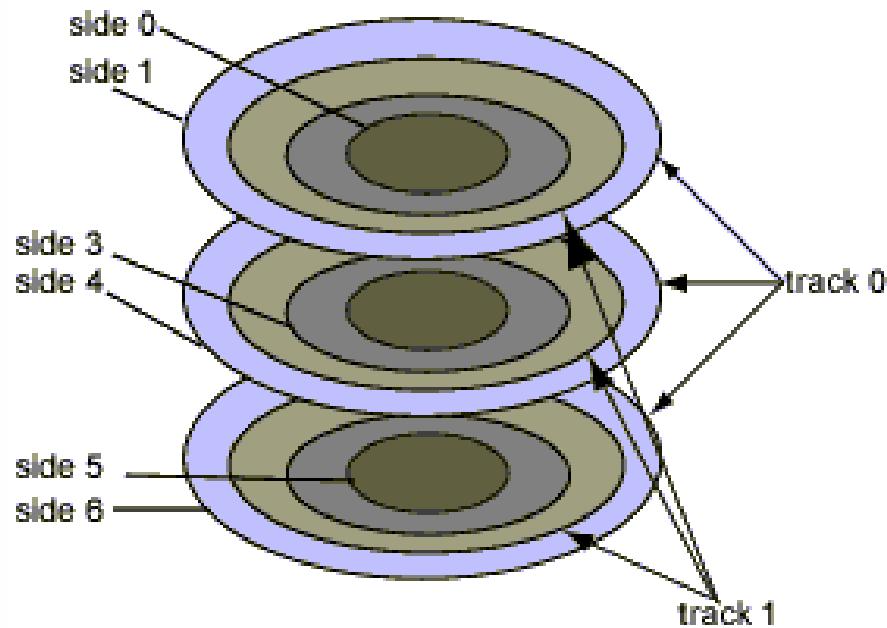


## Đĩa cứng

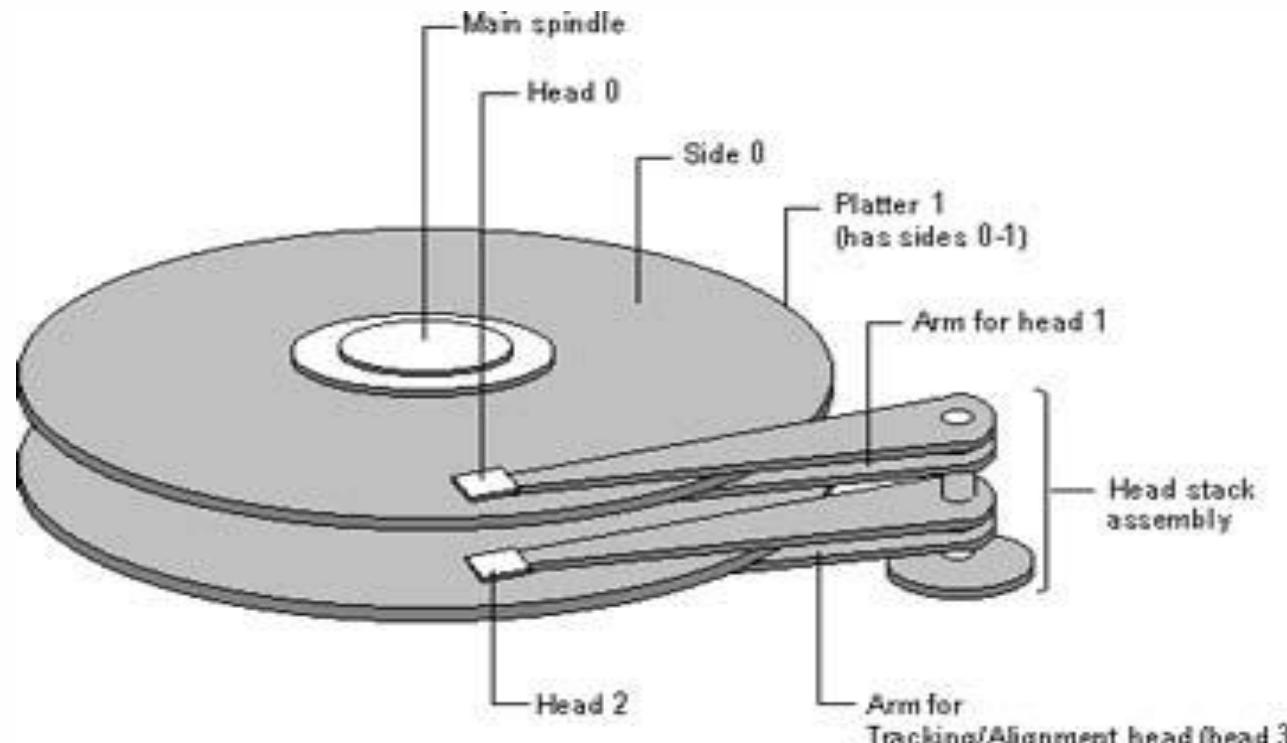
- Đĩa cứng bao gồm nhiều mặt (Side)
- Trên một mặt có nhiều vòng tròn đồng tâm gọi là rãnh từ (Track)
- Trên một rãnh từ (Track) ta chia nhỏ ra nhiều đoạn gọi là cung từ (Sector)
- Các rãnh có cùng bán kính nằm trên các mặt khác nhau tạo nên một *mặt trụ* (cylinder).
  - VD: Tập hợp tất cả các Track 0 tạo thành Cylinder 0,
  - Tập hợp tất cả các Track 1 tạo thành Cylinder 1



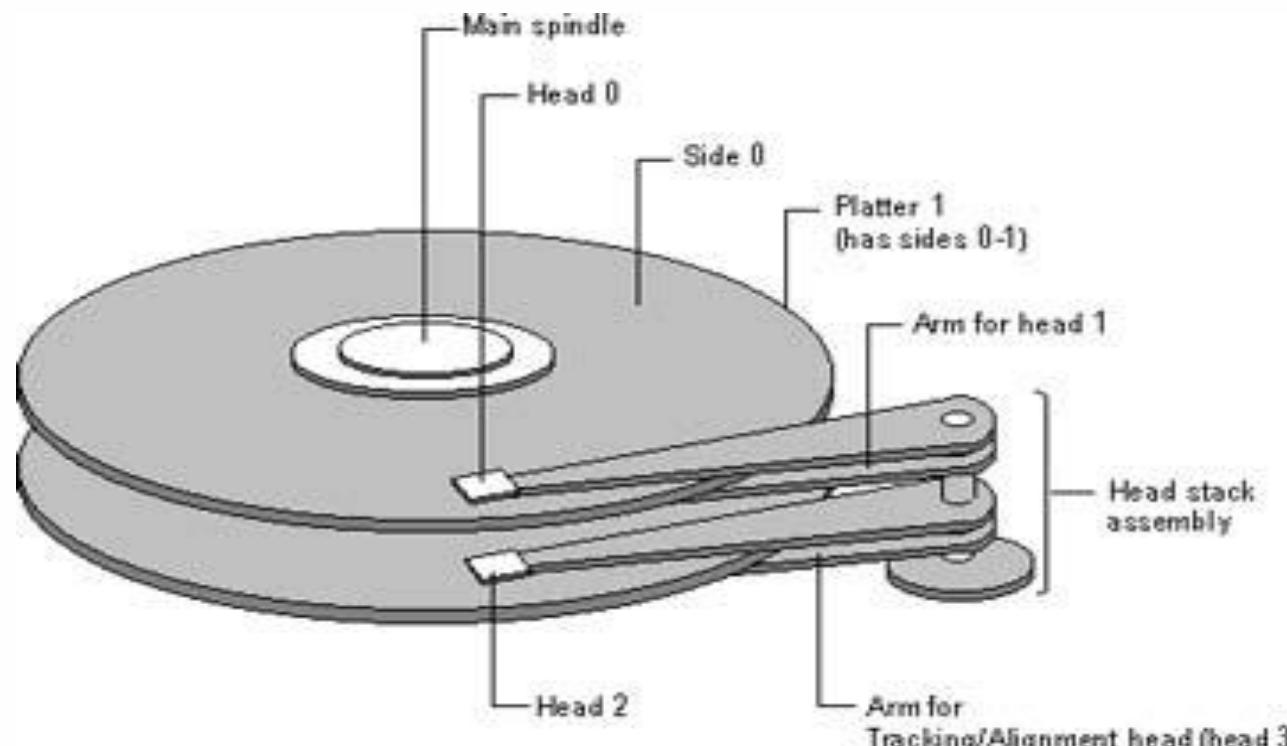
# Đĩa cứng



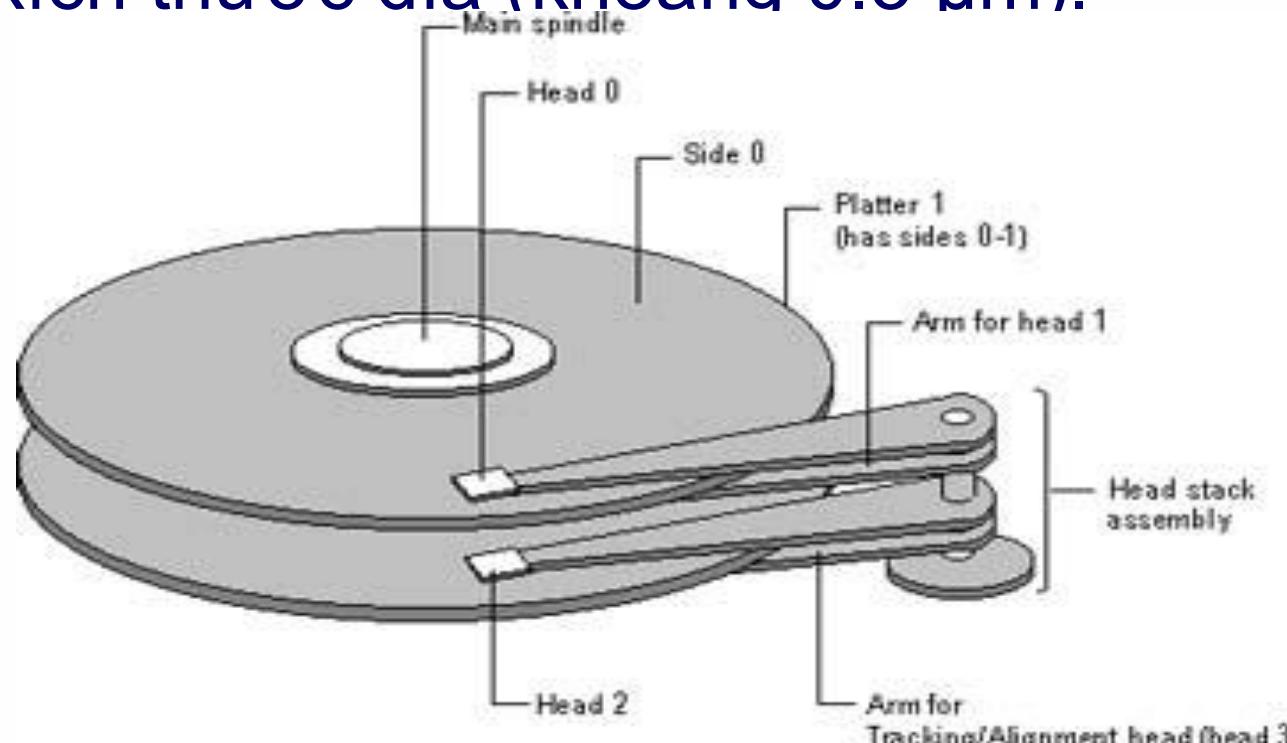
- Dung lượng của đĩa cứng khá lớn.
- Mỗi một mặt đĩa có một đầu từ đọc/ghi. Mỗi đĩa có nhiều đầu từ. Đầu từ có thể di chuyển đến mọi sector trên mặt đĩa tương ứng của nó nhờ hai động cơ điều khiển.



- Một động cơ điều khiển quay đĩa để đầu từ có thể ghi/đọc các cung khác nhau trên mỗi vòng tròn rãnh từ.
- Động cơ thứ hai điều khiển di chuyển đầu từ dọc theo bán kính của đĩa để cho phép đưa đến các rãnh khác nhau của bề mặt đĩa.



- Khi đĩa quay, đầu từ không chạm vào mặt đĩa mà cách một lớp đệm không khí.
- Khoảng cách giữa mặt đĩa và đầu từ tùy theo tốc độ quay và mật độ ghi dữ liệu của đĩa và rất nhỏ so với kích thước đĩa (khoảng 0.3 μm).





- **Các chuẩn nối ghép ổ đĩa thông dụng**
  - + **IDE** (*Integrated Drive Electronics*): Mỗi điểm nối ghép cho phép nối tối đa 2 ổ đĩa.
  - + **SCSI** (*Small Computer System Interface*): Cho phép nối ghép tối đa 7 hoặc 15 thiết bị vào-ra.
  - + **SATA**



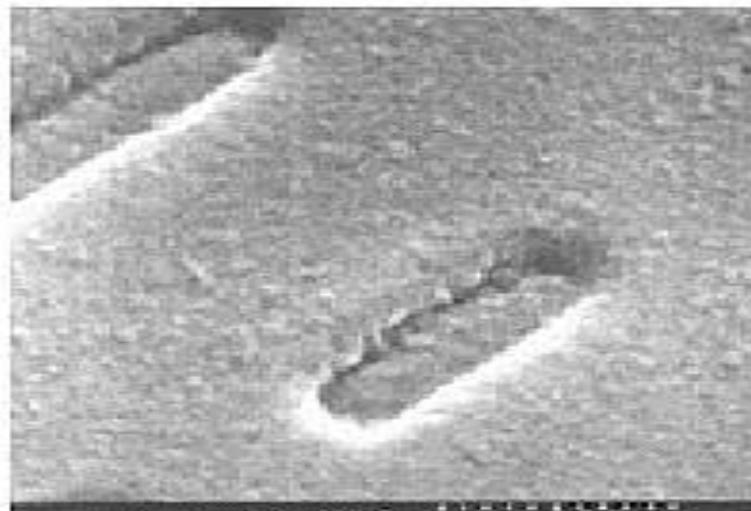
# Các công nghệ đĩa quang

- **1. Nguyên tắc lưu trữ quang**
- Thông tin được lưu trữ trên đĩa quang dưới dạng thay đổi tính chất quang của bề mặt đĩa.
- Tính chất này được phát hiện qua chất lượng phản xạ một tia sáng của bề mặt đĩa. Tia sáng này thường là một tia LASER với bước sóng cố định (790nm đến 850nm). Bề mặt đĩa được thay đổi khi ghi để có thể phản xạ tia laser tốt hoặc kém.

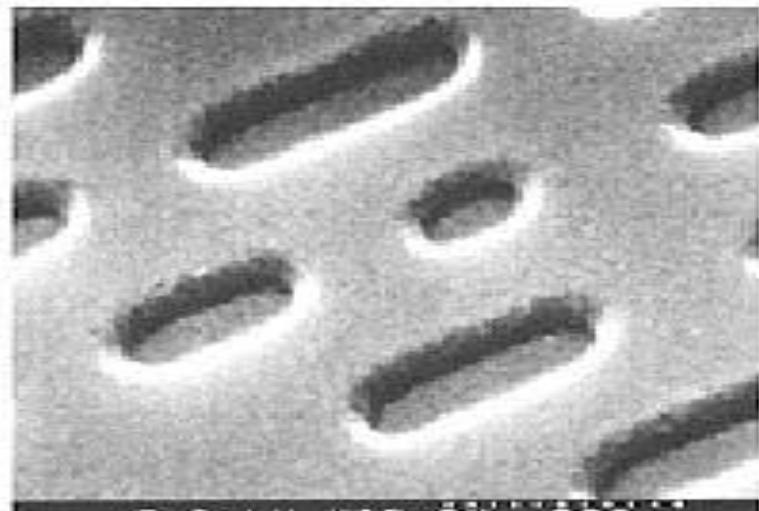


# Các công nghệ đĩa quang

- **Đĩa CD-ROM (Compact Disk Read Only Memory)**
- Dữ liệu tồn tại dưới dạng các mặt phẳng (land) và các lỗ (pit).



CD

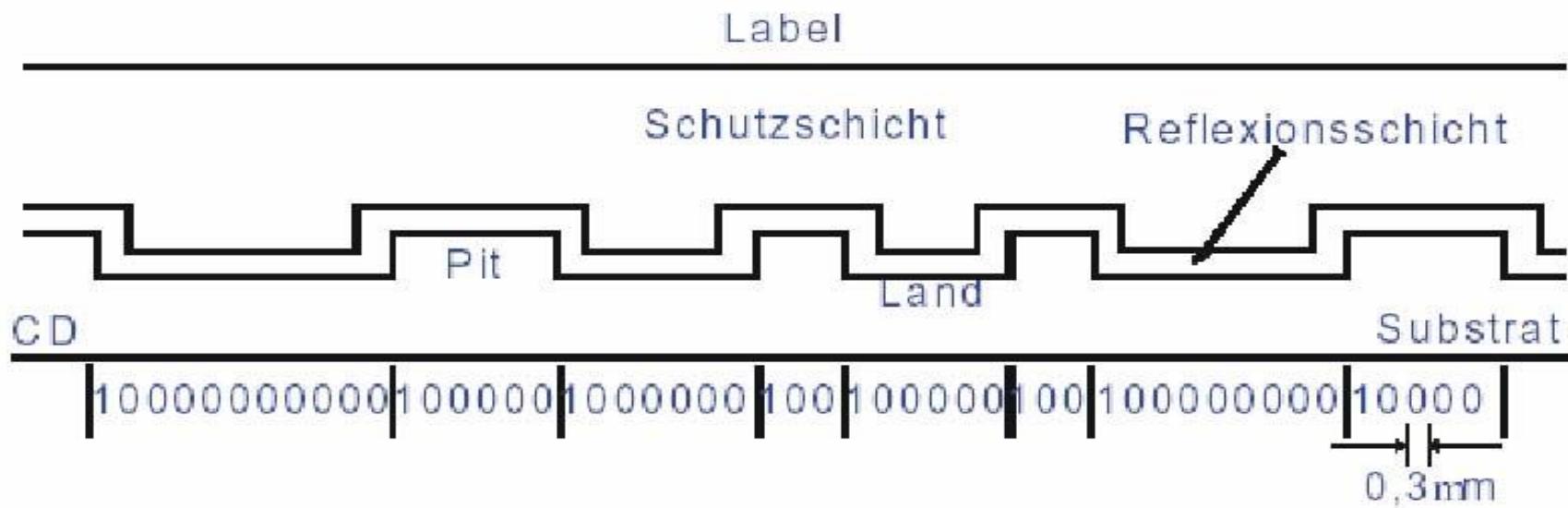


DVD



# Các công nghệ đĩa quang

- Bit 1 tương ứng với sự thay đổi từ mặt phẳng thành lỗ hay ngược lại; còn những lỗ hay mặt phẳng kéo dài (không có sự thay đổi) tương ứng với bit 0.





# Các công nghệ đĩa quang

## ▪ Đĩa CD-R (Compact Disk Recordable)

- Khi sản xuất ra, các đĩa này đều là đĩa trắng (chưa có thông tin). Sau đó có thể khi dữ liệu lên đĩa này nhưng chỉ ghi được một lần nhờ ổ ghi CD-R riêng.
- CD-R có cấu trúc và hoạt động tương tự như CD-ROM. Cấu tạo gồm nhiều lớp, trong đó lớp chứa dữ liệu là một lớp màu polymer hữu cơ. Khi bị tia laser đốt cháy, lớp màu này chuyển sang màu đen và đóng vai trò như các lỗ (pit) của CD-ROM.
- Các đĩa CD-R sau khi ghi có thể được đọc từ ổ CD-ROM hoặc từ ổ CD-R. Các đĩa CD-R còn được gọi là WORM (write one read multiple).



# Các công nghệ đĩa quang

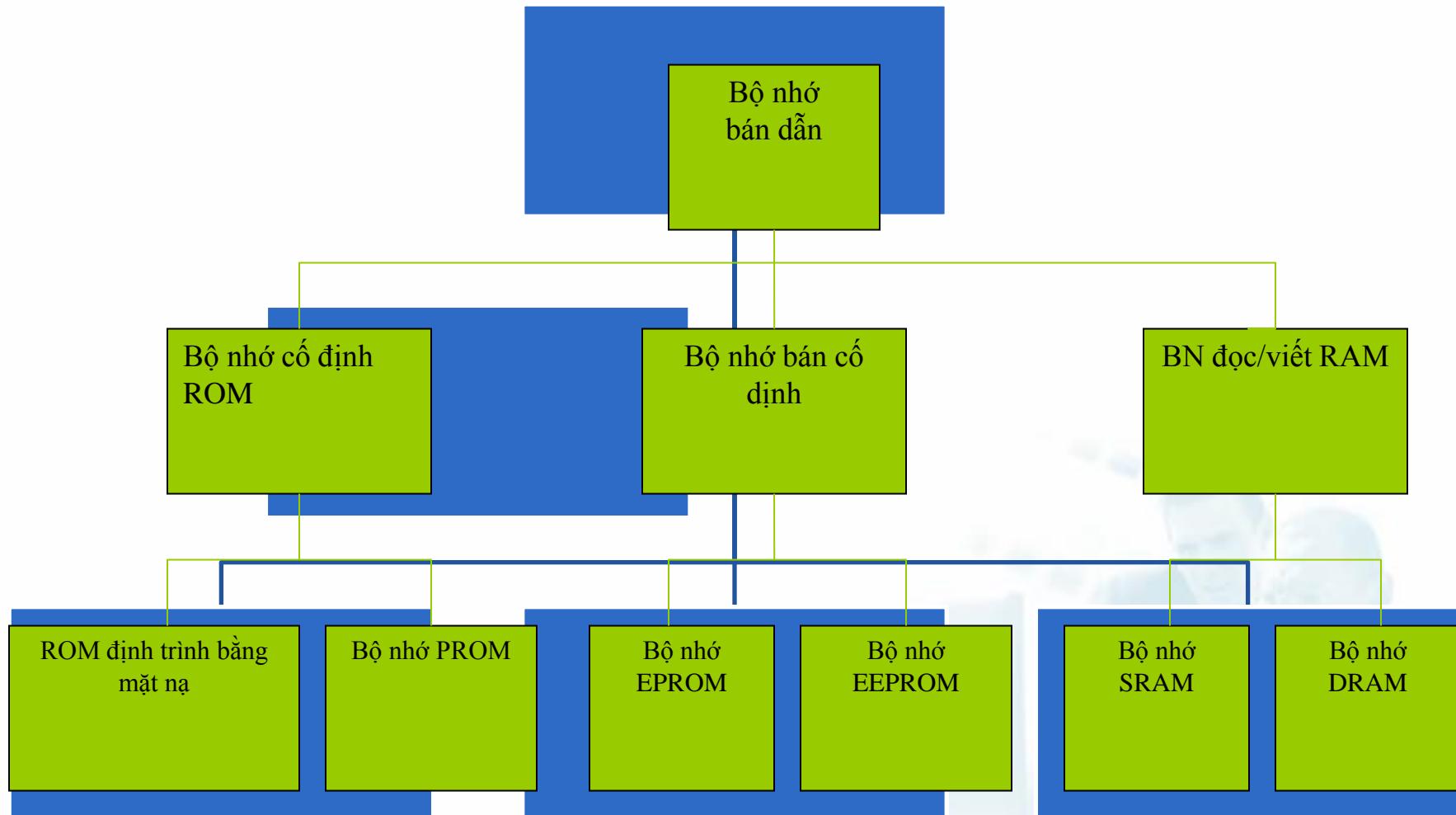
- **Đĩa CD-RW (Compact Disk Rewriteable)**
  - CD-RW có cấu trúc và hoạt động tương tự như CD-R. Trong đó lớp chứa dữ liệu là một lớp kim loại: trạng thái tinh thể (phản xạ ánh sáng - mặt phẳng) và trạng thái vô định hình (không phản xạ ánh sáng - vùng lỗ trong CD-ROM hay màu bị đốt đen trong CD-R).
  - Quá trình thay đổi trạng thái này có thể thay đổi bất kì tùy theo công suất laser nên đĩa CD-RW có thể được ghi rồi xóa đi ghi lại nhiều lần.
  - Để thực hiện nguyên tắc này, ổ đĩa CD-RW sử dụng 3 mức công suất laser khác nhau:
    - + Công suất cao hay còn gọi là công suất ghi, dùng để tạo lớp vô định hình (lớp không phản xạ).
    - + Công suất vừa hay còn gọi là công suất xóa, dùng để tạo lớp tinh thể (lớp phản xạ).
    - + Công suất thấp hay còn gọi là công suất đọc, dùng để đọc dữ liệu như CD thường.



# Các công nghệ đĩa quang

- **Đĩa DVD (Digital Video Disk - đĩa video số hay Digital Versatile Disk)**
  - Đây là loại đĩa quang có dung lượng lớn và có tốc độ nhanh hơn so với các đĩa quang trên.
  - Đĩa DVD có thể lưu trữ thông tin trên hai mặt, mỗi mặt có thể có đến 2 lớp dữ liệu.
  - Các đĩa DVD hiện nay thường có dung lượng là 4,7GB/mặt hoặc 9,4GB/mặt.
  - Tốc độ truy nhập cơ bản của ổ đĩa DVD là 1,321Mbyte/s.
  - DVD cũng có các loại DVD-ROM, DVD-R và DVD-RW

# Bộ nhớ trong





# RAM (Random Access Memory)

- Bộ nhớ truy cập ngẫu nhiên:có thể đọc/ghi nhiều lần,có hai loại:
  - Bộ nhớ RAM tĩnh SRAM (Static RAM) thường được xây dựng trên các mạch điện tử flip-flop
  - RAM động DRAM (Dynamic RAM) được xây dựng trên cơ sở nhớ các điện thích ở tụ điện



# ROM(Read Only Memory)

- ROM lập trình theo kiểu mặt nạ được gọi đơn giản là ROM.
- Nó được chế tạo trên một phiến silic theo một số bước xử lý như quang khắc và khuếch tán để tạo ra những tiếp giáp bán dẫn có tính dẫn điện theo một chiều (như diode, transistor trường).
- Người thiết kế định rõ chương trình muốn ghi vào ROM và thông tin này được sử dụng để điều khiển quá trình làm mặt nạ.



# PROM(Programmable ROM)

- PROM gồm có các diode vắt chéo. Mỗi diode lại được nối tiếp với một cầu chì.
  - Bình thường khi chưa lập trình, các cầu chì còn nguyên vẹn, nội dung của PROM sẽ toàn là 0. Khi định vị đến một bit bằng cách đặt một xung điện ở lối ra tương ứng, cầu chì sẽ bị đứt và bit này sẽ là 1.
- Có thể lập trình toàn bộ các bit trong ROM (1 lần duy nhất)



# EPROM(Easable Programmable ROM)

- Là loại ROM có thể lập trình và xoá được, số liệu được viết vào bằng điện nhưng có thể lưu giữ theo kiểu không bay hơi.
- Một ô nhớ EPROM được làm từ 1 transistor được gọi là FAMOST (Floating gate avanlanche injection MOS transistor).





# EEPROM(Electrically Erasable PROM)

- Bộ nhớ ROM có thể lập trình được nhiều lần.
- EEPROM có thể xoá số liệu bằng phương pháp điện khắc phục nhược điểm của cửa sổ thạch anh đắt và không tiện lợi.



## Chương 4-Lập trình ghép nối

1. Lập trình ghép nối với cổng COM
2. Lập trình ghép nối với cổng LPT
3. Lập trình ghép nối với cổng USB





# 1. Lập trình ghép nối cổng COM

- Sử dụng ngôn ngữ Visual Basic 6.0 và điều khiển MsComm
- Sử dụng ngôn ngữ C#.NET và điều khiển SerialPort



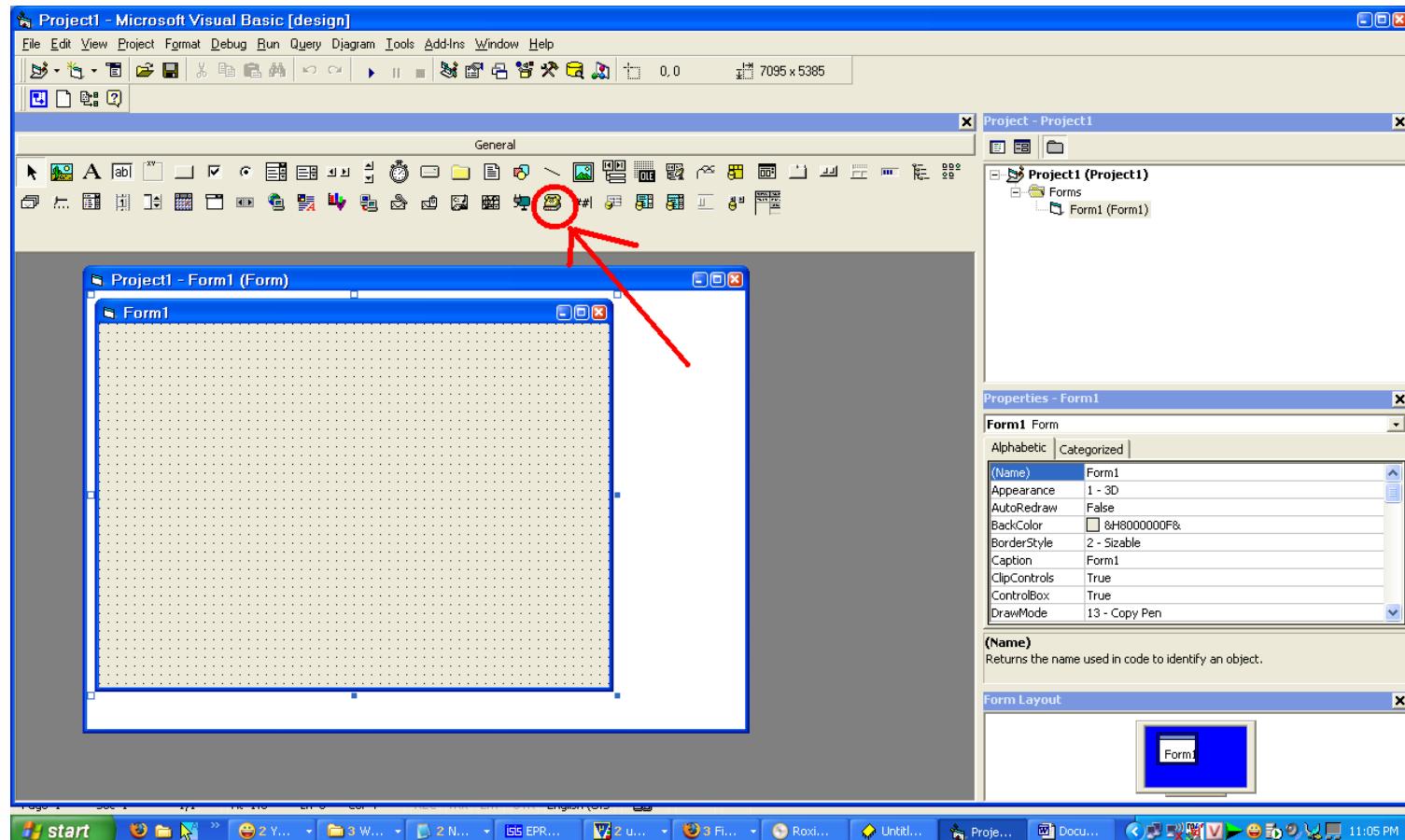
# Các bước lập trình

- Kiểm tra sự tồn tại và trạng thái các cổng COM trên máy
- Lập trình thiết lập các thông số kết nối
- Lập trình đóng, mở cổng COM
- Lập trình thực hiện gửi, nhận dữ liệu qua cổng COM
  - Gửi dữ liệu qua cổng COM
  - Bắt dữ liệu nhận được từ cổng COM khi có sự kiện comEvReceive



# Lập trình ghép nối cổng COM

- Sử dụng ngôn ngữ Visual Basic 6.0 và điều khiển MsComm





# Kiểm tra sự tồn tại của cổng COM

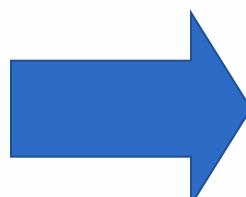
- Liên tục mở các cổng (thường trong dải từ 1 đến 4)
- Nếu mở cổng có lỗi thì bỏ qua
- Nếu mở cổng thành công thì thêm vào danh sách các cổng có thể sử dụng





# Thiết lập các thông số kết nối

- Các thuộc tính chính cần thiết lập
  - ComPort: Số hiệu cổng COM
  - Settings: thiết lập tốc độ, định dạng gói tin
  - RThreshold: thiết lập số ký tự nhận được trước khi điều khiển MsComm sinh sự kiện comEvReceive
    - ✓ RThreshold=0 -> Không bắt được sự kiện nhận ký tự
  - SThreshold: thiết lập số ký tự gửi đi trước khi điều khiển MsComm sinh sự kiện comEvSend
    - ✓ SThreshold=0 -> Không bắt được sự kiện gửi ký tự



**Có thể thiết lập theo hai cách**

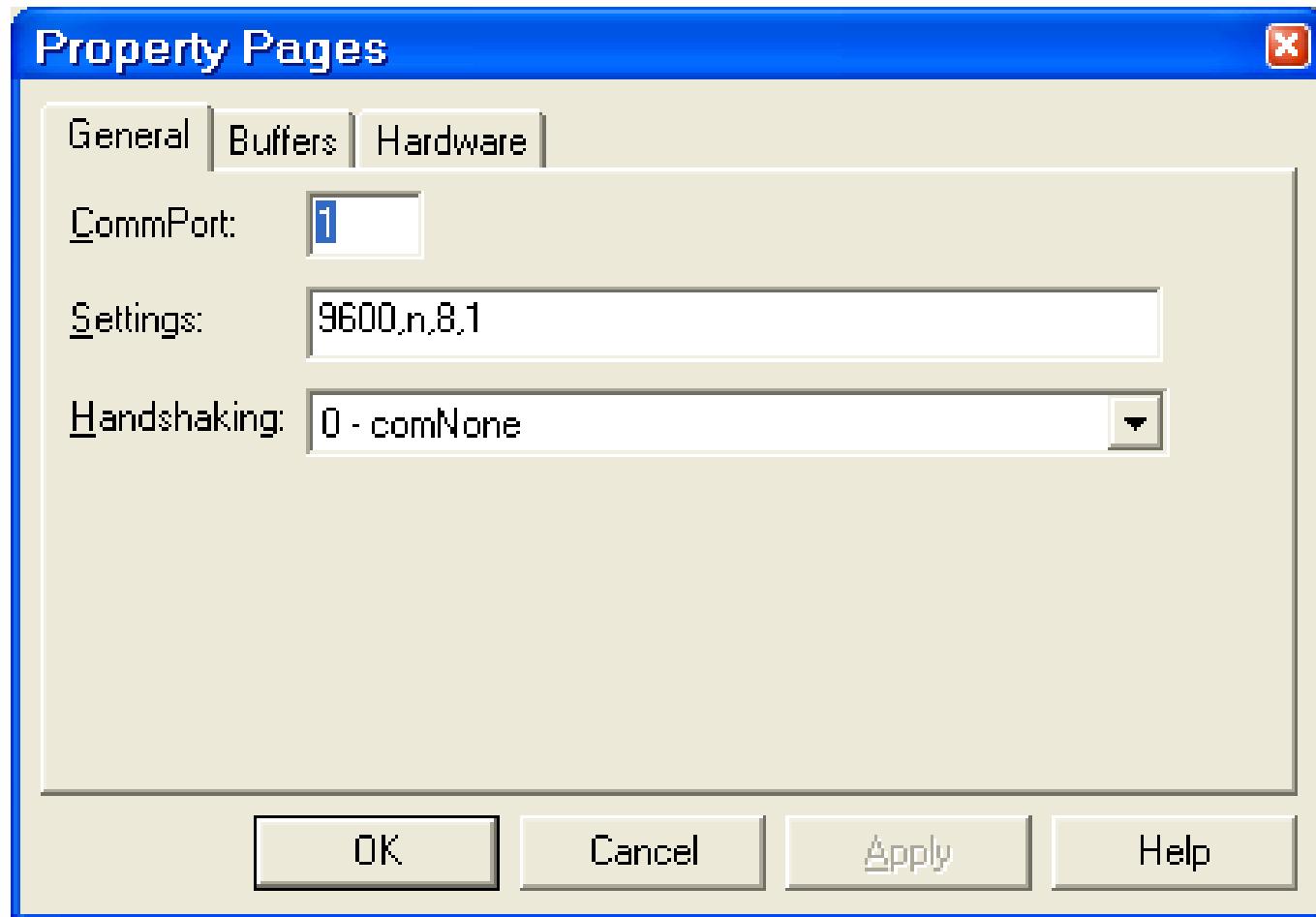
**-Sử dụng giao diện đồ họa**

**-Lập trình**



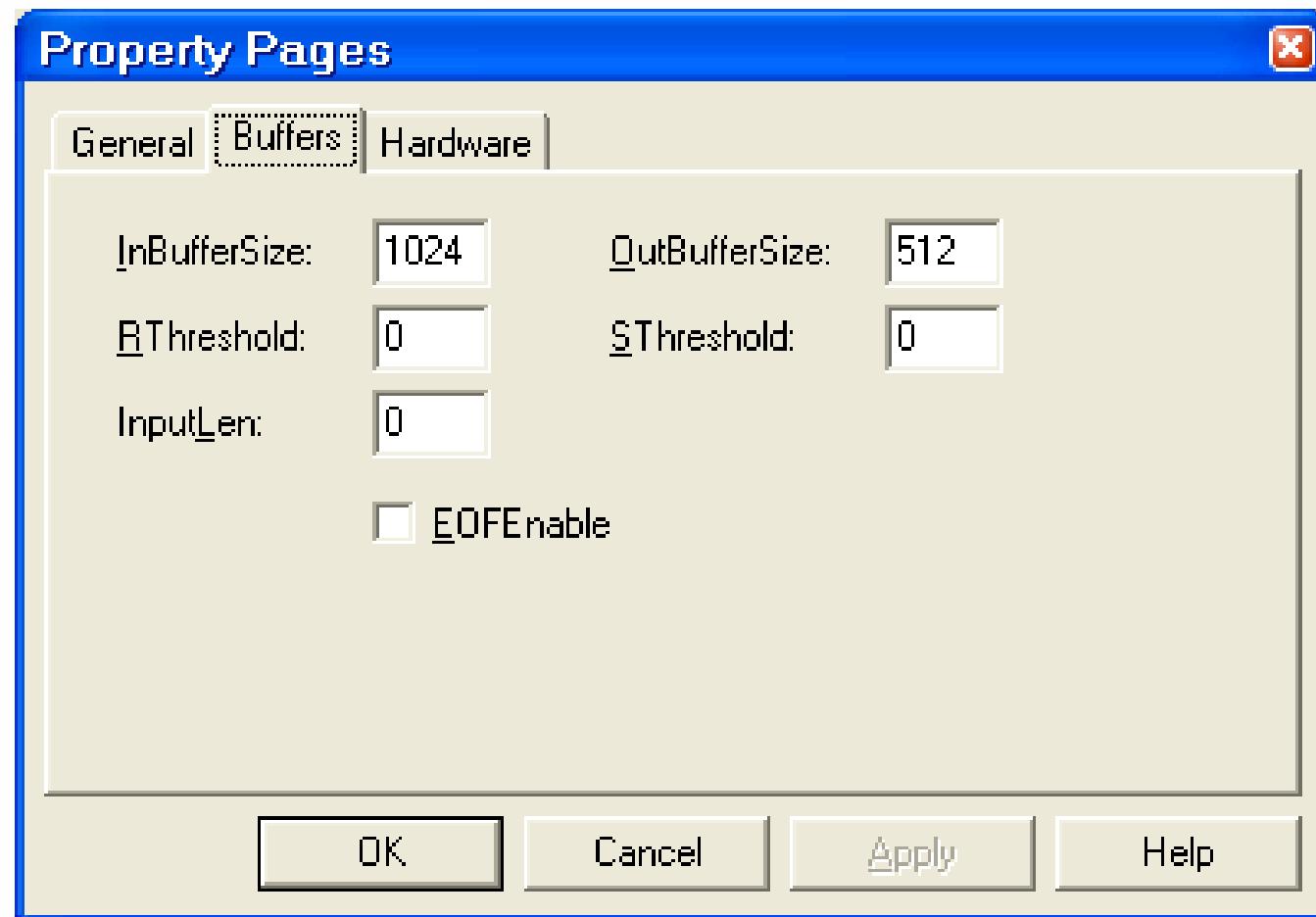
# Thiết lập các thông số kết nối

- Sử dụng giao diện đồ họa





# Thiết lập các thông số kết nối





# Lập trình đóng mở cổng COM

- Để đóng mở cổng COM cần tác động lên thuộc tính PortOpen của điều khiển MsComm
  - PortOpen=True -> mở cổng COM
  - PortOpen=False -> Đóng cổng COM
- **Hai thuộc tính này cũng được sử dụng để kiểm tra trạng thái hiện tại của cổng COM**



# Lập trình đóng mở cổng COM

- With MSComm1
- If .PortOpen = True Then
- .PortOpen = False
- Else
- .PortOpen = True
- If Err.Number <> 0 Then
- MsgBox “Error”
- Err.Clear
- End If
- End If
- End With



# Lập trình gửi nhận dữ liệu qua cổng COM

- Gửi dữ liệu qua cổng COM
  - Sử dụng thuộc tính **Output** của điều khiển MsComm
    - ✓ VD: `MsComm.Output="Test Send Data"` -> chuỗi "Test Send Data" sẽ được gửi qua cổng COM
  - **Khi gửi dữ liệu, có thể bắt và xử lý sự kiện gửi dữ liệu comEvSend nếu thiết lập SThreshold >0**





# Lập trình gửi nhận dữ liệu qua cổng COM

- Nhận dữ liệu nhận về từ cổng COM
  - Bắt sự kiện comEvReceive
    - ✓ Chỉ có thể bắt và xử lý sự kiện này khi thuộc tính RThreshold được thiết lập >0
  - Đọc dữ liệu thông qua thuộc tính **Input**
    - ✓ Dữ liệu được đọc từ bộ đệm nhận
    - ✓ Số lượng ký tự đọc về phụ thuộc thuộc tính **InputLen**
      - InputLen=0 -> Đọc toàn bộ bộ đệm nhận
      - InputLen#0 -> Đọc số lượng ký tự bằng giá trị thuộc tính **InputLen**



## Lập trình gửi nhận dữ liệu qua cổng COM

- 'Xu ly cac su kien voi dieu khien MSComm
- Private Sub MSComm1\_OnComm()
- With MSComm1
- Select Case MSComm1.CommEvent
  - Case comEvReceive 'Khi nhan duoc du lieu
  - Case comEvSend 'Khi gui du lieu
- End Select
- End With
- End Sub





# Lập trình cổng COM sử dụng VB 6.0

- Ví dụ minh họa

- Chương trình thu thập số liệu nhiệt độ
- Chương trình chat qua cổng COM





# Chương trình thu thập số liệu nhiệt độ

- Cắm mạch đo nhiệt độ vào máy tính qua cổng COM
- Chọn cổng COM phù hợp
- Mở cổng COM
- Sau mỗi khoảng thời gian được thiết lập (Sử dụng Timer), chương trình gửi lệnh xuống giao tiếp với mạch đo nhiệt độ, yêu cầu mạch gửi trả nhiệt độ tại thời điểm hiện tại.
- Hiển thị số liệu nhiệt độ



# Chương trình thu thập số liệu nhiệt độ

Form1

DS Cong COM: 3

Lenh can gui:

Mo cong

Dong cong

Gui lenh

Thoat

24.25

(Mã nguồn: Donhietdo\_COM\_VB6)



# Chương trình chat qua cổng COM

- Hai chương trình chạy ở hai máy, sử dụng dây cáp nối hai cổng COM của hai máy (hoặc sử dụng cổng COM ảo, nối theo kiểu NULL modem)
- Tại mỗi máy, chọn cổng COM phù hợp và mở cổng
- Tiến hành chat, gửi nhận tin nhắn qua lại giữa hai máy





# Chương trình chat qua cổng COM

The image shows two instances of a Windows application window titled "Form1".

**Left Window (Top):**

- Label: DS Cong COM: Value 5
- Text Input: Gui: Chao ban
- Text Output: Nhan: Xin chao
- Buttons: Mo cong, Dong cong, Gui lenh

**Right Window (Bottom):**

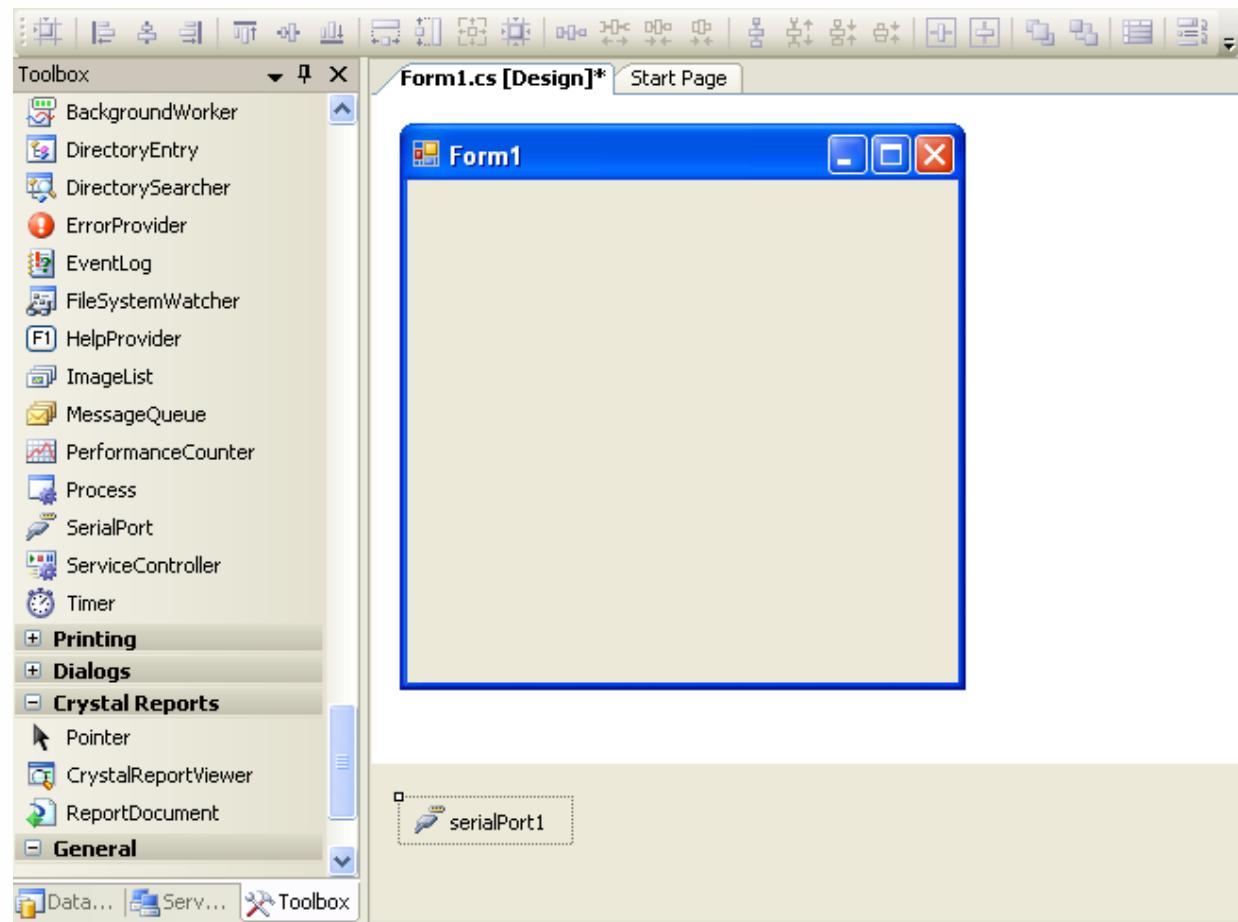
- Label: DS Cong COM: Value 6
- Text Input: Gui: Xin chao
- Text Output: Nhan: Chao ban
- Buttons: Mo cong, Dong cong, Gui lenh, Thoat

(Mã nguồn: chat\_COM\_VB6)



# Lập trình ghép nối cổng COM

- Sử dụng ngôn ngữ C#.NET và điều khiển SerialPort





# Kiểm tra sự tồn tại của cổng COM

- Sử dụng hàm **GetPortNames** lấy về danh sách tất cả các cổng COM trên máy tính

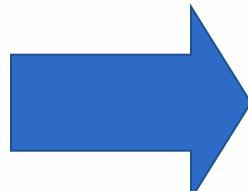
`String[] ComList=System.IO.Ports.SerialPort.GetPortNames();`

- Để kiểm tra trạng thái các cổng, tiến hành mở lần lượt các cổng
  - Nếu mở cổng có lỗi thì chứng tỏ cổng đó đã được mở
  - Nếu mở cổng thành công thì cổng đó vẫn chưa được mở nên có thể sử dụng được



# Thiết lập các thông số kết nối

- Các thuộc tính chính cần thiết lập
  - Baudrate: tốc độ
  - Databits: số bit data
  - Parity: bit kiểm tra chẵn lẻ
  - PortName: số hiệu cổng COM
  - ReceiveBytesThreshold: ngưỡng nhận



**Có thể thiết lập theo hai cách**

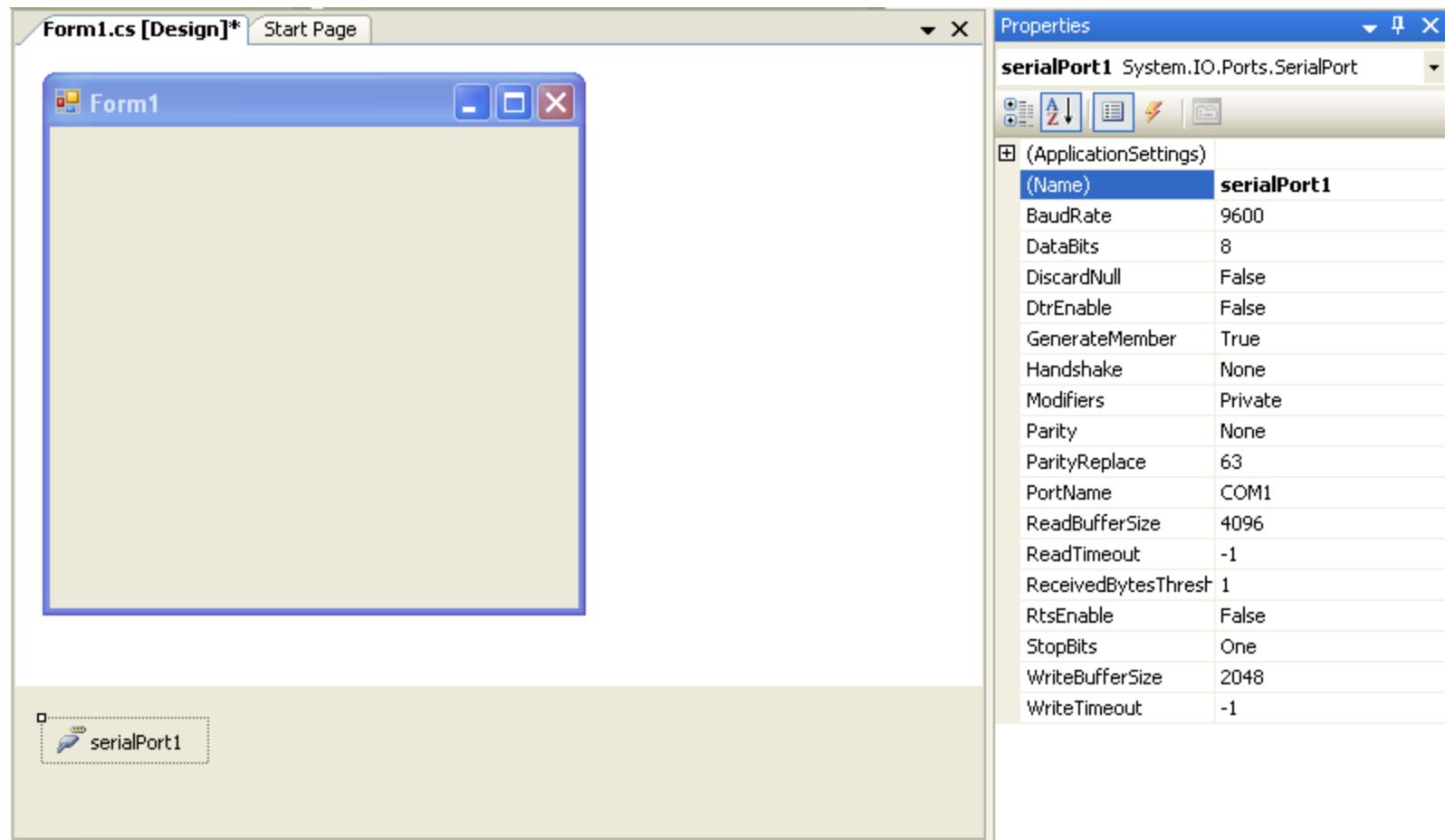
**-Sử dụng giao diện đồ họa**

**-Lập trình**



# Thiết lập các thông số kết nối

- Sử dụng giao diện đồ họa: cửa sổ hiển thị thuộc tính của đối tượng SerialPort





# Lập trình đóng mở cổng COM

- Để đóng mở cổng COM cần sử dụng hai phương thức
  - Phương thức Open: mở cổng COM
  - Phương thức Close: đóng cổng COM
- **Sử dụng thuộc tính IsOpen để kiểm tra trạng thái của cổng COM**
  - IsOpen=True -> cổng đang được sử dụng
  - IsOpen=False -> có đang rỗi



# Lập trình đóng mở cổng COM

```
■ if (!serialPort1.isOpen)
■ {
■     serialPort1.Open();
■ }
■ else
■ {
■     MessageBox.Show("Cổng COM bận");
■ }
```





# Lập trình gửi nhận dữ liệu qua cổng COM

- Gửi dữ liệu qua cổng COM
  - Sử dụng phương thức Write của đối tượng SerialPort
  - ✓ VD: **SerialPort1.Write("Test send method")**



# Lập trình gửi nhận dữ liệu qua cổng COM

- Nhận dữ liệu qua cổng COM
  - Tạo hàm bắt sự kiện khi có dữ liệu được nhận về qua cổng COM

```
serialPort1.DataReceived += new  
SerialDataReceivedEventHandler(OnReceive);
```

- OnReceive: hàm thực thi việc nhận dữ liệu và xử lý
  - Đọc dữ liệu sử dụng các phương thức
    - ✓ Read
    - ✓ ReadExisting
    - ✓ ReadLine
    - ✓ ...





# Lập trình cổng COM sử dụng C#.NET

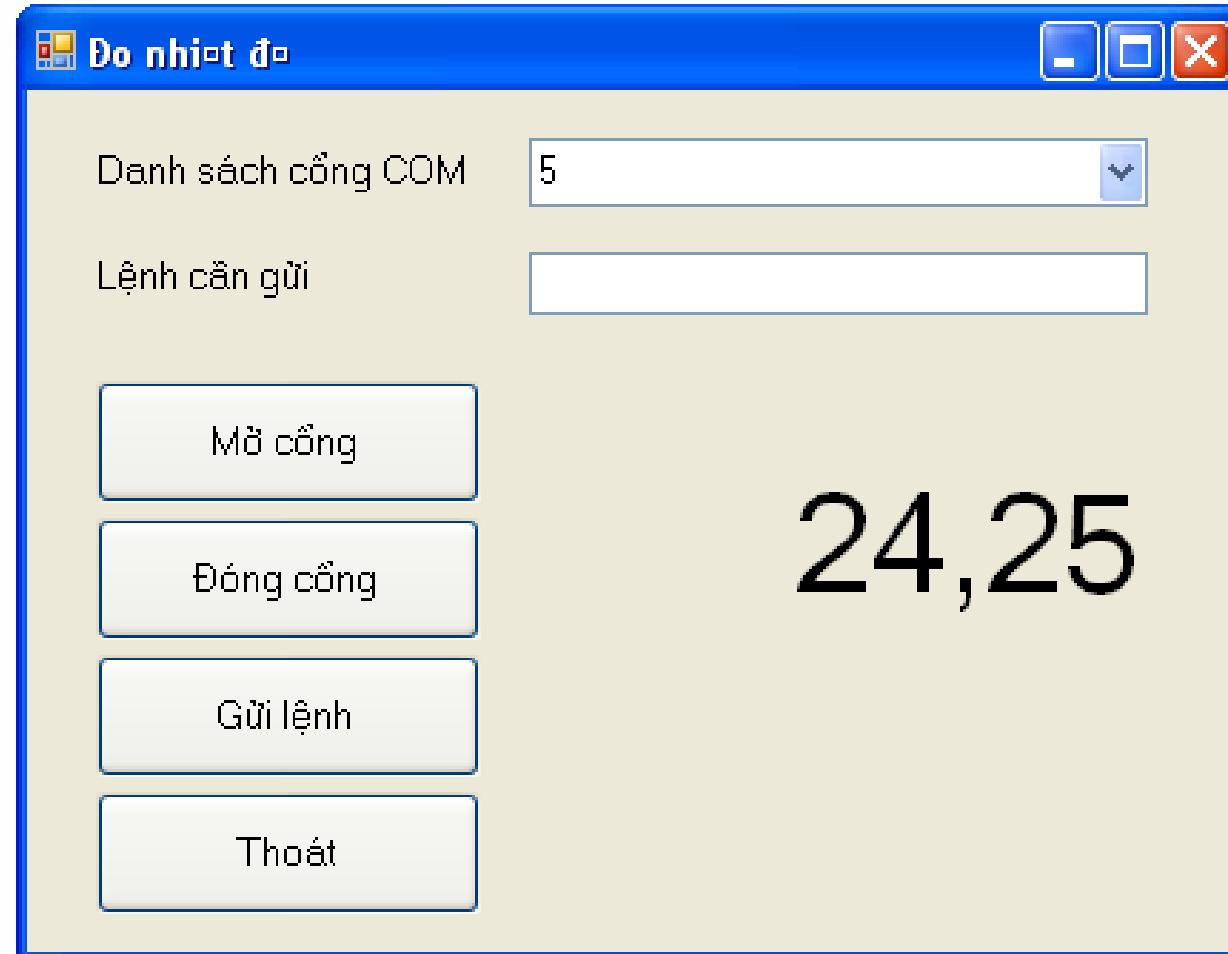
- Ví dụ minh họa

- Chương trình thu thập số liệu nhiệt độ
- Chương trình chat qua cổng COM





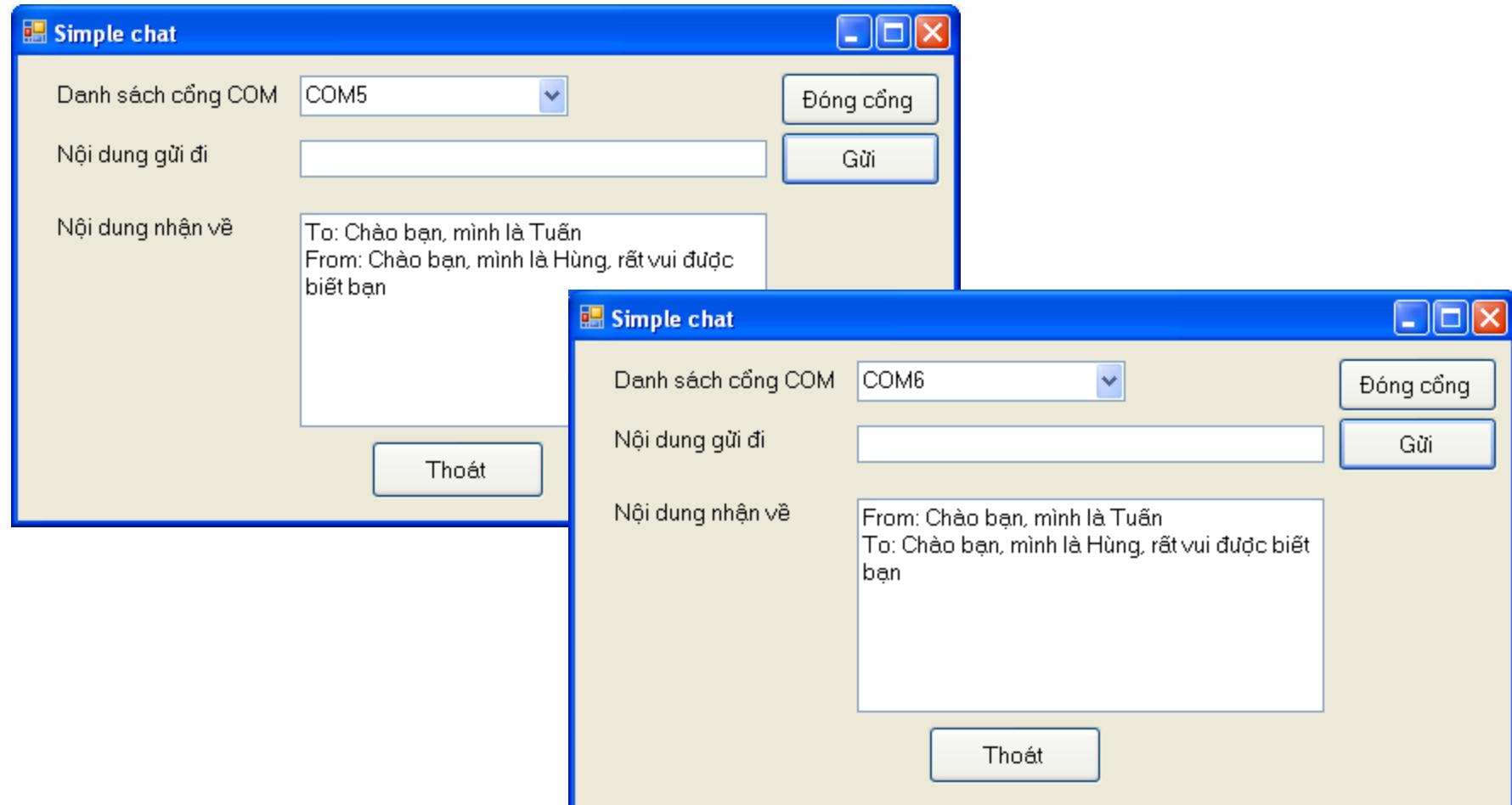
# Chương trình thu thập số liệu nhiệt độ



(Mã nguồn: [Donhietdo\\_COM\\_SHARP](#))



# Chương trình chat qua cổng COM

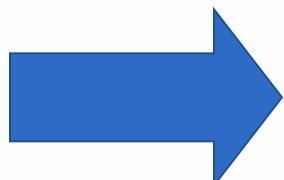


(Mã nguồn: chat COM C SHARP)



## 2. Lập trình ghép nối với cổng LPT

- Từ Windows 2000 trở đi không cho phép lập trình truy xuất trực tiếp các cổng vào ra



Sử dụng thư viện

VD: **Inpout32.dll**



# Lập trình ghép nối với cổng LPT

- Lập trình sử dụng Visual Basic 6.0
- Lập trình sử dụng C#.NET





# Lập trình sử dụng Visual Basic 6.0

- Cách sử dụng thư viện liên kết động (DLL) trong VB 6
  - Khai báo sử dụng thư viện và các hàm trong thư viện

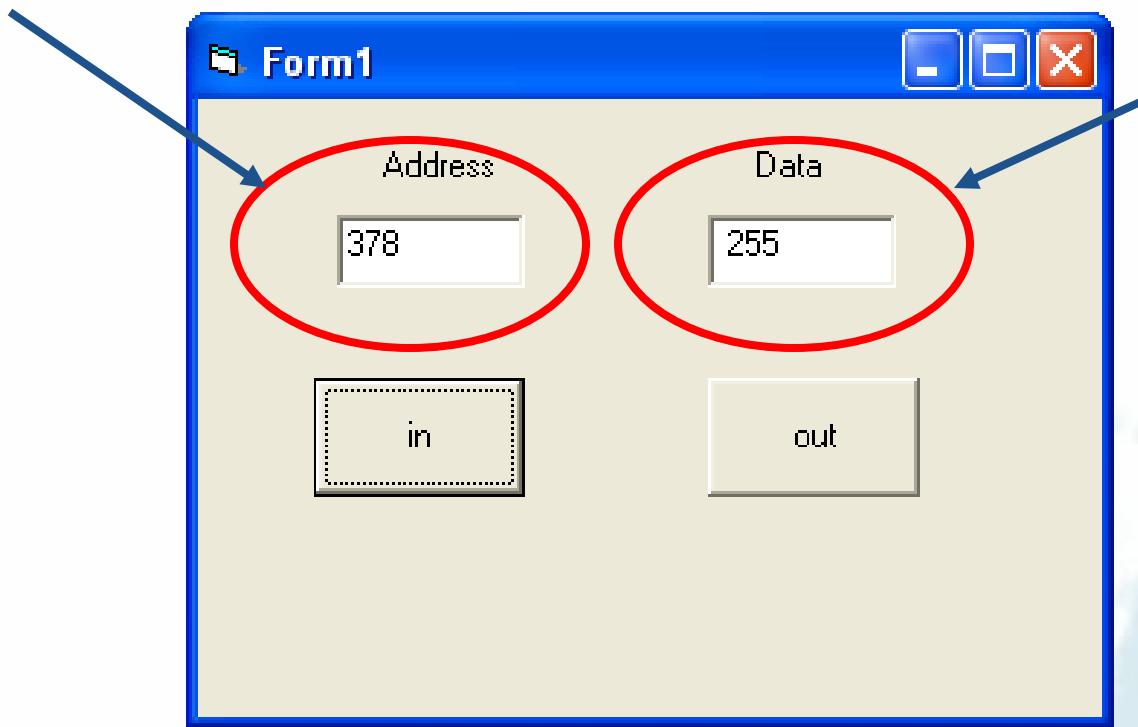
The screenshot shows the Microsoft Visual Basic 6.0 IDE with the title bar "inpouttest - inout (Code)". The code editor displays the following declarations:

```
Public Declare Function Inp32 Lib "inpout32.dll" _  
    (ByVal PortAddress As Integer) As Integer  
Public Declare Sub Out32 Lib "inpout32.dll" _  
    (ByVal PortAddress As Integer, ByVal Value As Integer)
```



# Lập trình sử dụng Visual Basic 6.0

Địa chỉ  
cổng

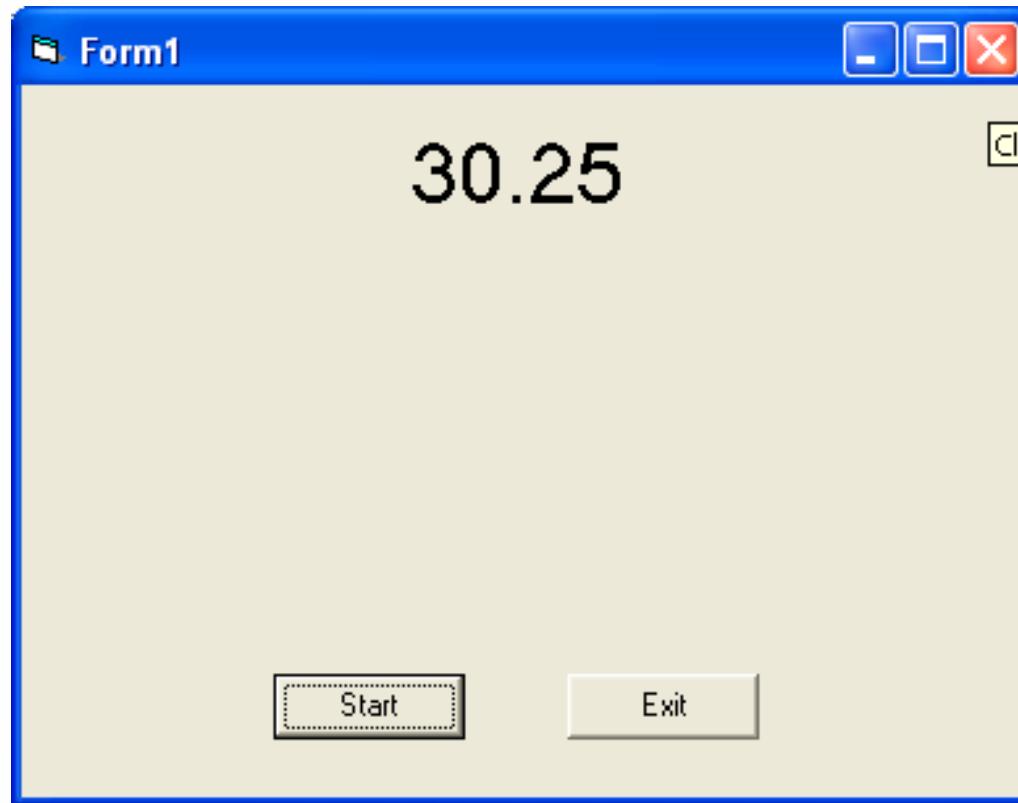


Dữ liệu đọc  
vào từ cổng  
hoặc muốn  
ghi ra cổng



# Chương trình minh họa

- Thu thập số liệu nhiệt độ



(Mã nguồn: [Donhietdo\\_LPT\\_VB6](#))



# Lập trình sử dụng C#.NET

- Cách sử dụng thư viện liên kết động (DLL) trong C#.NET

- Khai báo sử dụng Namespace: InteropServices  
using System.Runtime.InteropServices;

- Khai báo các thư viện và các hàm cần sử dụng  
[DllImport("inpout32.dll")]

```
public static extern int Inp32(int PortAddress);
```

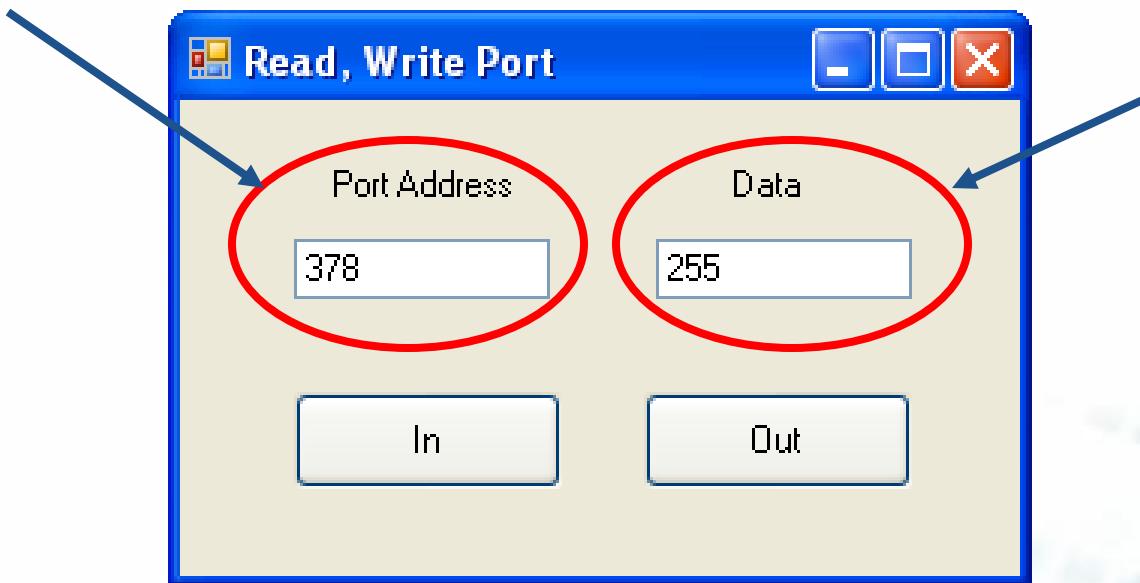
```
[DllImport("inpout32.dll")]
```

```
public static extern int Out32(int PortAddress, int value);
```



# Lập trình sử dụng C#.NET

Địa chỉ  
cổng



Dữ liệu đọc  
vào từ cổng  
hoặc muốn  
ghi ra cổng



### 3. Lập trình ghép nối cổng USB

- Sử dụng Visual Basic 6.0
- Sử dụng C#.NET

## Windows API functions





# Lập trình ghép nối cổng USB

- Cách tra cứu và sử dụng hàm API
  - **Tra cứu hàm API:** sử dụng bộ phần mềm API-Guide
  - **Sử dụng hàm API:** khai báo thư viện, các hàm, các cấu trúc cho các tham số của hàm (nếu cần thiết)



# Phần mềm API-Guide

API-Guide 3.7 - 925 functions found!

File Options Language Help

Back Forward Stop Refresh Home

AB alphabetical | Groups | Info | Parameters | Notes | Example(s) | .NET |

GetSystemInfo

GetNumberFormat  
GetNumberOfEventLogRecords  
GetObject  
GetObjectType  
GetOldestEventLogRecord  
GetOpenFileName  
GetOpenFileNamePreview  
GetParent  
GetPixel  
GetPolyFillMode  
GetPrinter  
GetPriorityClass  
GetPrivateProfileInt  
GetPrivateProfileSection  
GetPrivateProfileSectionNames  
GetPrivateProfileString  
GetProcAddress  
GetProcessHeap  
GetProcessMemoryInfo  
GetProcessTimes  
GetProcessWindowStation  
GetProfileInt  
GetProfilesDirectory  
GetProp  
GetPwrCapabilities  
GetQueueStatus  
GetRgrBox  
GetROP2  
GetSaveFileName  
GetSaveFileNamePreview  
GetSecurityDescriptorDacl  
GetShortPathName  
getsockopt  
GetStartupInfo  
GetStdHandle

System Information

```
Private Declare Sub GetSystemInfo Lib "kernel32" (lpSystemInfo As SYSTEM_INFO)
Private Type SYSTEM_INFO
    dwOemID As Long
    dwPageSize As Long
    lpMinimumApplicationAddress As Long
    lpMaximumApplicationAddress As Long
    dwActiveProcessorMask As Long
    dwNumberOfProcessors As Long
    dwProcessorType As Long
    dwAllocationGranularity As Long
    dwReserved As Long
End Type
Private Sub Form_Load()
    Dim SInfo As SYSTEM_INFO
    'KPD-Team 1998
    'URL: http://www.allapi.net/
    'KPDTeam@Allapi.net
    'Set the graphical mode to persistent
    Me.AutoRedraw = True
    'Get the system information
    GetSystemInfo SInfo
    'Print it to the form
    Me.Print "Number of processor:" + str$(SInfo.dwNumberOfProcessors)
    Me.Print "Processor:" + str$(SInfo.dwProcessorType)
    Me.Print "Low memory address:" + str$(SInfo.lpMinimumApplicationAddress)
    Me.Print "High memory address:" + str$(SInfo.lpMaximumApplicationAddress)
End Sub
```



# Ghép nối với USB Joystick





# Cấu trúc JOYINFO

- Windows định nghĩa cấu trúc JOYINFO để lưu các thông tin về tình trạng các nút bấm trên Joystick

```
typedef struct {  
    UINT wXpos;  
    UINT wYpos;  
    UINT wZpos;  
    UINT wButtons;  
} JOYINFO;
```

Nút trái, phải

Nút lên, xuống

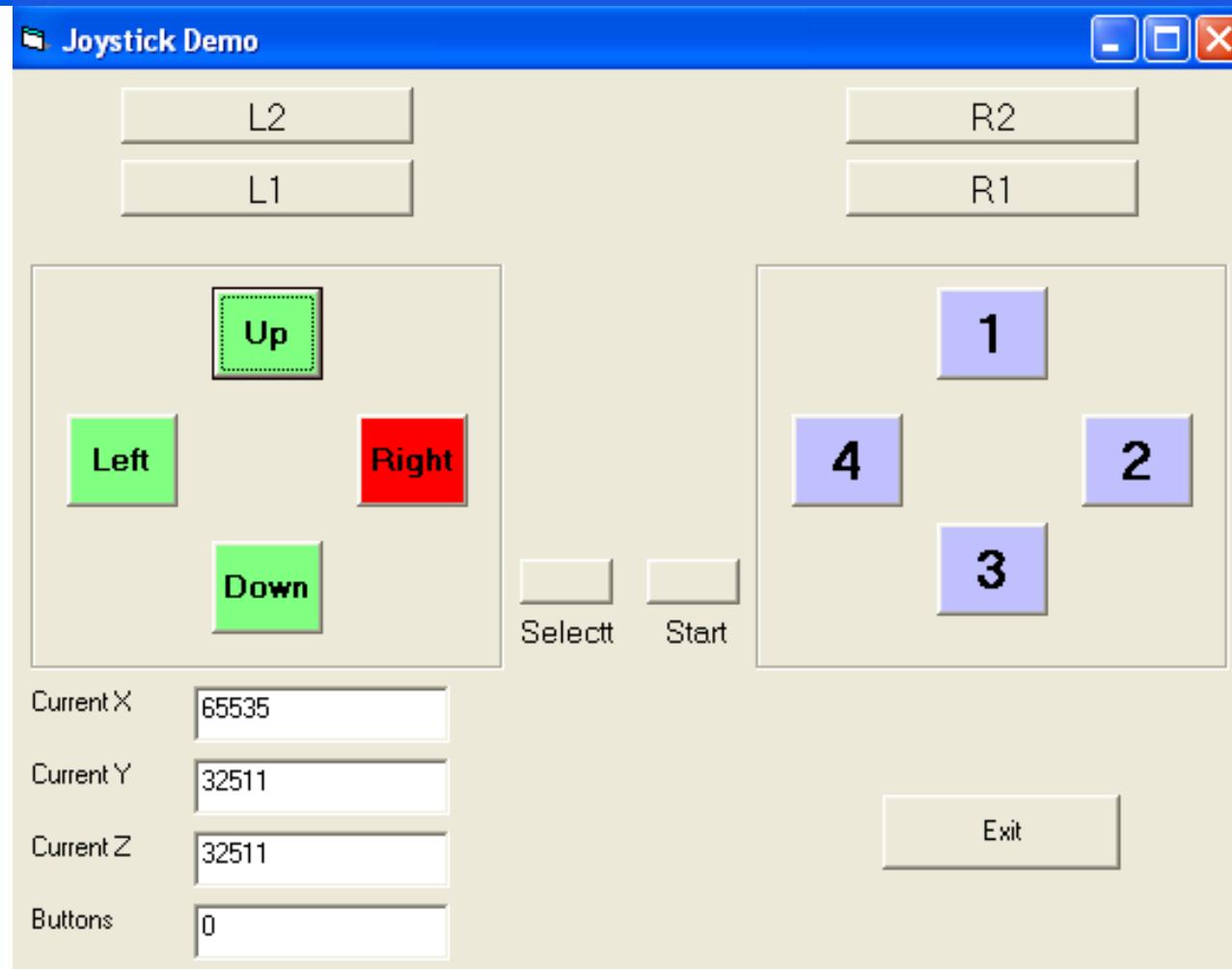
Các nút chức  
năng: 1, 2, 3, 4, L1,  
L2, R1, R2, Select,  
Start



- wXpos
  - wXpos=0 -> nút sang trái được bấm
  - wXpos=65535 -> nút sang phải được bấm
- wYpos
  - wYpos=0 -> nút lên được bấm
  - wYpos=65535 -> nút xuống được bấm
- wButtons: mỗi bit biểu diễn trạng thái của một nút chức năng
  - VD: Button 1 -> bit 0, Button 2 -> bit 1...



# Chương trình kiểm tra Joystick



(Mã nguồn: Test\_JoyStick\_VB)



# Ví dụ sử dụng USB Joystick

- Trò chơi xếp hình (Tetris)
  - Xếp hình 2D
    - ✓ Sử dụng Visual Basic 6.0
    - ✓ Sử dụng C#.NET
  - Xếp hình 3D
    - ✓ Sử dụng C#.NET





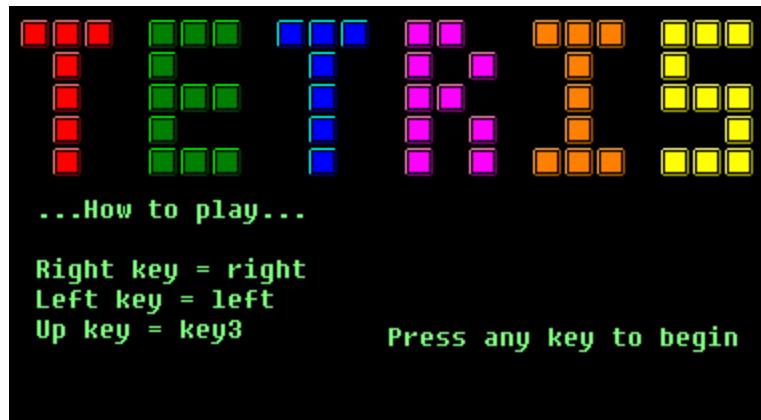
# Các bước lập trình

- Tìm đoạn mã xử lý sự kiện cần thay thế (VD: khi người dùng sử dụng bàn phím)
- Khai báo sử dụng hàm API: joyGetPos
- Gọi hàm API joyGetPos và thay thế cho đoạn mã xử lý sự kiện cũ, giữ nguyên các đoạn mã liên quan đến trò chơi.





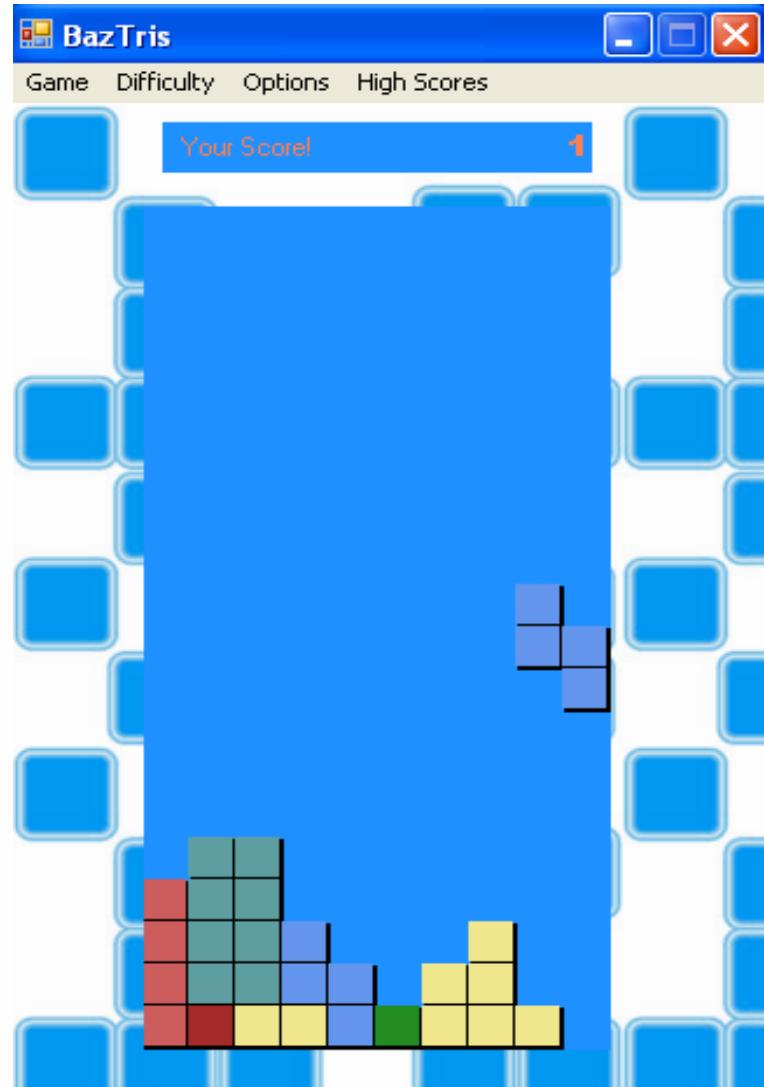
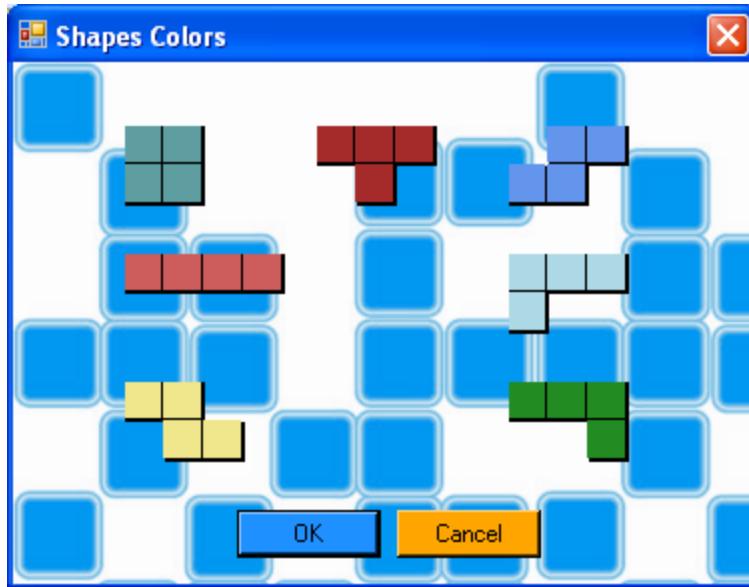
# Xếp hình 2D-Visual Basic



(Mã nguồn: Tetris\_2D\_VB)

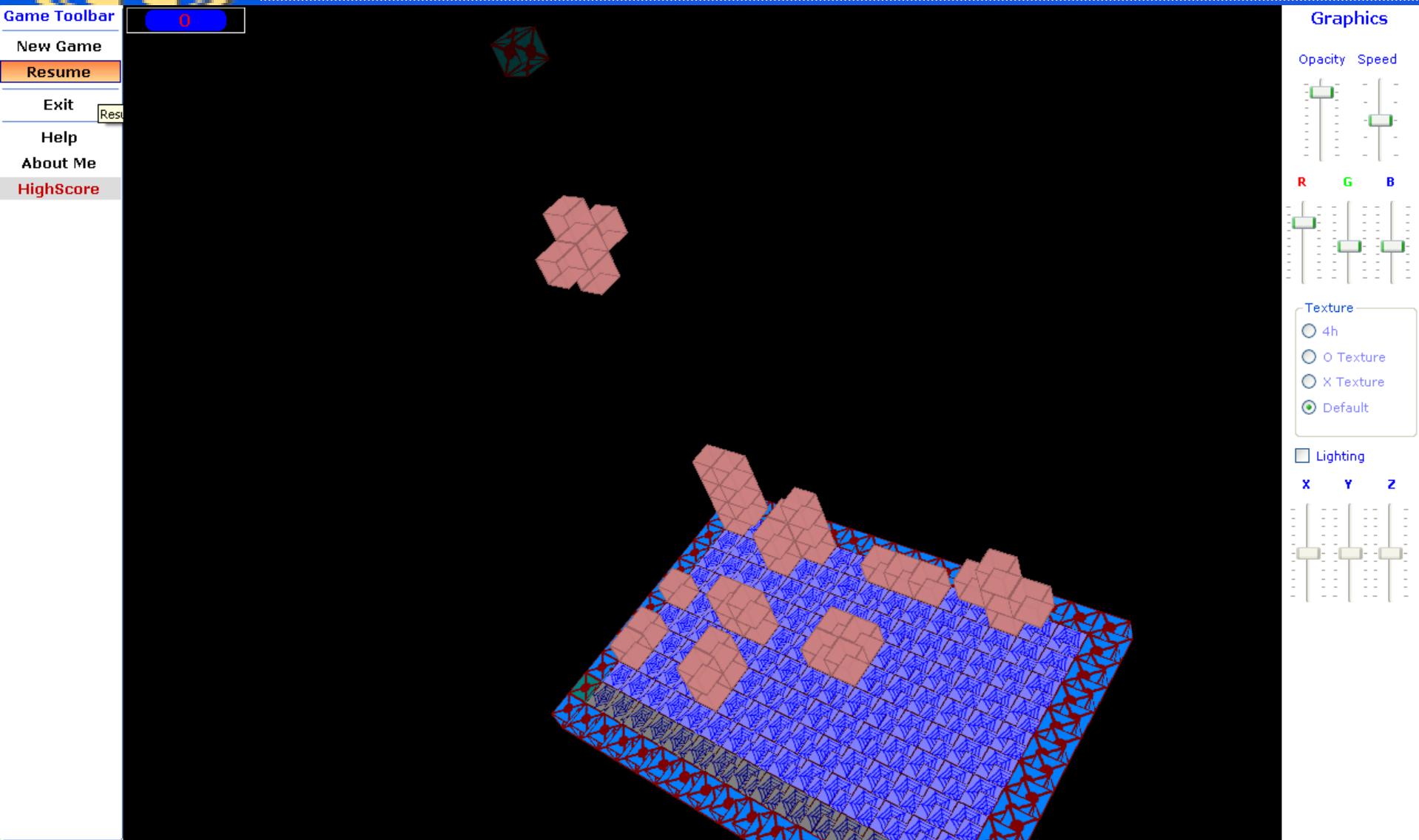


# Xếp hình 2D-C#.NET





# Xếp hình 3D – C#.NET



Pause Game !

TBNV&GN



# Ghép nối với Webcam

- Có hai cách thông dụng
  - Sử dụng hàm API
  - Sử dụng thư viện DirectShow





# Sử dụng hàm API

- Hàm `capCreateCaptureWindow` của thư viện `avicap32.dll`
- Hàm `SendMessage` của thư viện `user32`



# Lập trình kết nối HID class

## Bước 1: Lấy về danh sách các thiết bị theo chuẩn HID sử dụng hàm

```
[DllImport("hid.dll", SetLastError = true)]
static extern void HidD_GethidGuid(
    ref System.Guid lpHidGuid);

[DllImport("setupapi.dll", SetLastError = true)]
static extern IntPtr SetupDiGetClassDevs(
    ref System.Guid ClassGuid,
    String Enumerator,
    Int32 hwndParent,
    Int32 Flags);
```



# Lập trình HID class

## Bước 2: Lấy về thông tin thiết bị

```
[DllImport("setupapi.dll", SetLastError = true)]
static extern bool SetupDiEnumDeviceInterfaces(
    IntPtr DeviceInfoSet,
    Int32 DeviceInfoData,
    ref System.Guid lpHidGuid,
    Int32 MemberIndex,
    ref SP_DEVICE_INTERFACE_DATA lpDeviceInterfaceData);

//Using Unicode version
[DllImport("setupapi.dll", SetLastError = true)]
static extern bool SetupDiGetDeviceInterfaceDetailW(
    IntPtr DeviceInfoSet,
    ref SP_DEVICE_INTERFACE_DATA lpDeviceInterfaceData,
    IntPtr DeviceInterfaceDetailData,
    Int32 detailSize,
    ref Int32 requiredSize,
    IntPtr DeviceInfoData);
```



# Lập trình HID class

## Bước 3: Mở file

```
[DllImport("kernel32.dll", CharSet = CharSet.Auto)]
static public extern Int32 CreateFile(string lpFileName,
    UInt32 dwDesiredAccess, UInt32 dwShareMode,
    ref SECURITY_ATTRIBUTES lpSecurityAttributes,
    UInt32 dwCreationDisposition, Int32 dwFlagsAndAttributes,
    Int32 hTemplateFile);
```



# Lập trình HID class

## Bước 4: Gửi nhận dữ liệu

```
[DllImport("kernel32.dll")]
static public extern bool WriteFile(Int32 hFile,
    ref byte lpBuffer, Int32 nNumberOfBytesToWrite,
    ref int lpNumberOfBytesWritten, Int32 lpOverlapped);

[DllImport("kernel32.dll")]
static public extern Int32 ReadFile(Int32 hFile,
    ref byte lpBuffer, Int32 nNumberOfBytesToRead,
    ref Int32 lpNumberOfBytesRead,
    ref OVERLAPPED lpOverlapped);
```



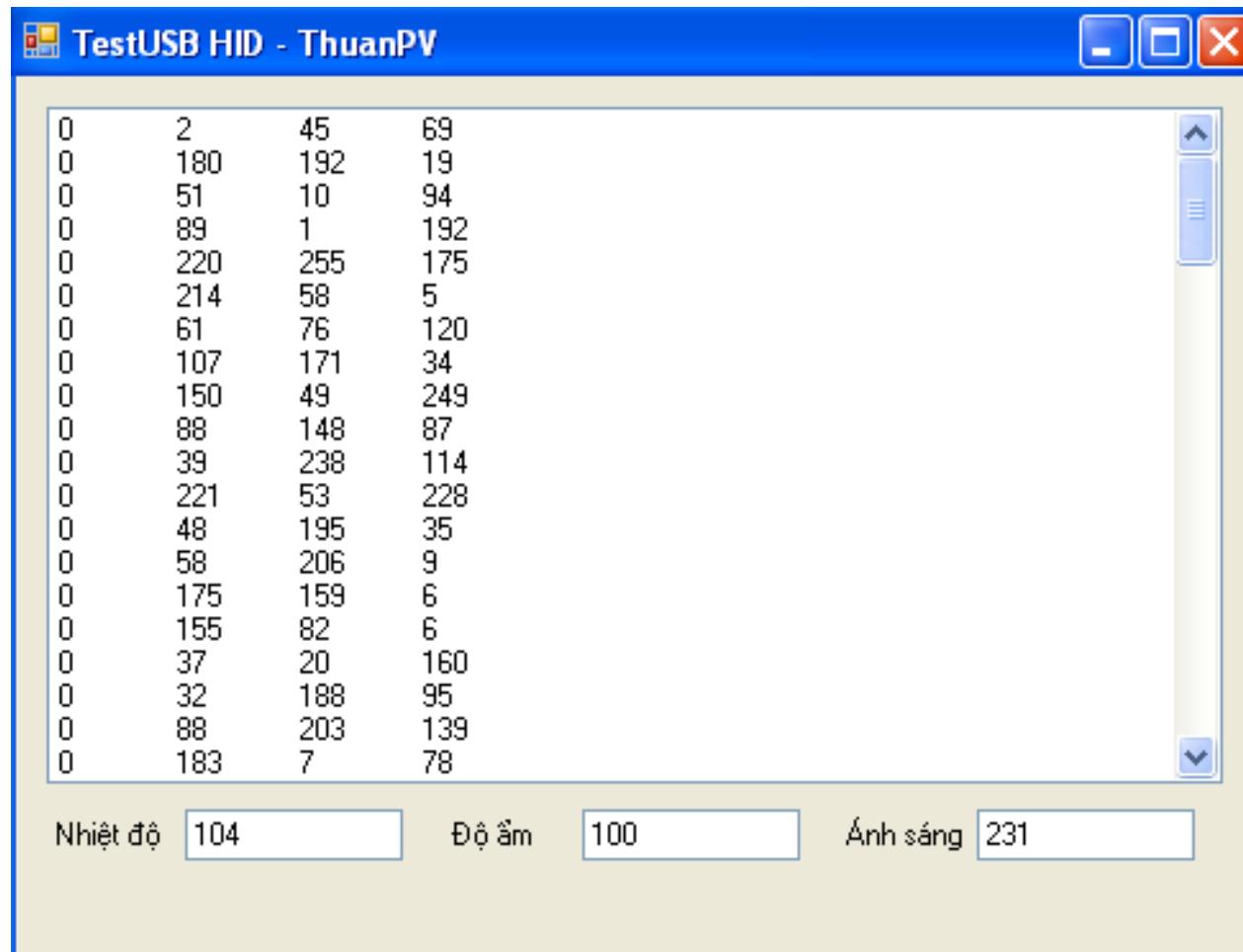


A large, blue, five-pointed starburst shape is centered on the slide. Inside the starburst, the word "Demo" is written in a bold, red, sans-serif font.

Demo



# Demo HID class





# Thư viện WinUSB

- Thư viện WinUSB hỗ trợ xây dựng các ứng dụng theo chuẩn bất kỳ, không theo các class sẵn có
- WinUSB gồm hai thành phần
  - WinUsb.sys: kernel-mode driver
  - WinUsb.dll: user-mode dll



# Lập trình sử dụng thư viện WinUSB

**Bước 1:** Tạo Device file sử dụng GUID của thiết bị

**Bước 2:** Cấu hình cho thiết bị

- Lấy về thông tin interface
- Lấy về thông tin các Endpoint
- Tạo ra các pipe

**Bước 3:** Giao tiếp với các Endpoint sử dụng các hàm:

- WinUsb\_ControlTransfer
- WinUsb\_WritePipe
- WinUsb\_ReadPipe