

# Chương 5: Phối ghép với bộ nhớ và thiết bị vào ra

## Kiến trúc máy tính

ThS. Đinh Xuân Trường  
[truongdx@ptit.edu.vn](mailto:truongdx@ptit.edu.vn)



Posts and Telecommunications  
Institute of Technology  
Faculty of Information Technology 1



CNTT1  
Học viện Công nghệ Bưu chính Viễn thông

January 15, 2023

# Mục tiêu Buổi 9

Các tín hiệu của CPU

Các mạch phụ trợ

Phối ghép CPU với bộ nhớ

Giới thiệu bộ nhớ

Giải mã địa chỉ cho bộ nhớ

Giải mã địa chỉ bộ nhớ sử dụng mạch logic cơ bản

Giải mã địa chỉ bộ nhớ sử dụng mạch tích hợp

Giải mã địa chỉ bộ nhớ sử dụng Chip nhớ EEPROM,

Phối ghép CPU với thiết bị vào ra

Một số mạch cổng đơn giản

## VXL 8088 có 40 chân tín hiệu:

► Nhóm tín hiệu địa chỉ:

- AD0-AD7: 8 chân dồn kênh cho phần thấp bus A và bus D;
- A8-A15: 8 chân tín hiệu phân cao bus A
- A16/S3-A19/S6: 4 chân dồn kênh cho phần cao bus A và bus C;

► Nhóm tín hiệu dữ liệu:

- AD0-AD7: 8 chân dồn kênh cho phần thấp bus A và bus D;
- Khi chân chốt ALE=0 là tín hiệu dữ liệu, ALE=1 à tín hiệu địa chỉ.

		8088 CPU	MIN MODE	MAX MODE
GND	1	40	V <sub>CC</sub>	
A14	2	39	A15	
A13	3	38	A16/S3	
A12	4	37	A17/S4	
A11	5	36	A18/S5	
A10	6	35	A19/S6	
A9	7	34	SS <sub>0</sub> (HIGH)	
A8	8	33	MN/MX	
AD7	9	32	RD	
AD6	10	31	HOLD ( $\overline{RQ}/GTO$ )	
AD5	11	30	HLDA ( $\overline{RQ}/GT_1$ )	
AD4	12	29	WR ( $\overline{LOCK}$ )	
AD3	13	28	I <sub>O</sub> /M ( $\overline{S}_2$ )	
AD2	14	27	DT/R ( $\overline{S}_1$ )	
AD1	15	26	DEN ( $\overline{S}_0$ )	
AD0	16	25	ALE (QS0)	
NMI	17	24	INTA (QS1)	
INTR	18	23	TEST	
CLK	19	22	READY	
GND	20	21	RESET	

# Các tín hiệu của CPU (cont.)

## Nhóm tín hiệu điều khiển hệ thống:

- ▶  $IO/\overline{M}$ : tín hiệu CPU chọn làm việc với thiết bị vào ra hay bộ nhớ.
  - $IO/\overline{M} = 1$ : CPU chọn làm việc với thiết bị vào ra
  - $IO/\overline{M} = 0$ : CPU chọn làm việc với bộ nhớ. Địa chỉ tương ứng của bộ phận được lựa chọn xuất hiện trên bus địa chỉ.
- ▶  $DT/\overline{R}$ : Tín hiệu xác định chiều vận chuyển dữ liệu trên bus dữ liệu.
  - $DT/\overline{R} = 1$ : dữ liệu đi ra từ CPU;
  - $DT/\overline{R} = 0$ : dữ liệu đi đến CPU.
- ▶  $\overline{RD}$ : Xung cho phép đọc (đảo). Khi  $\overline{RD} = 0$  bus dữ liệu sẵn sàng nhận dữ liệu từ bộ nhớ hoặc thiết bị ngoại vi.
- ▶  $\overline{WR}$ : Tín hiệu cho phép ghi. Khi  $\overline{WR} = 0$ , dữ liệu đã ổn định trên bus dữ liệu và được ghi vào bộ nhớ hoặc thiết bị vào ra khi  $\overline{WR} = 1$ .
- ▶  $\overline{DEN}$ : Tín hiệu báo cho mạch ngoài biết dữ liệu đã ổn định trên bus dữ liệu.

# Các tín hiệu của CPU (cont.)

- ▶ SS0: Tín hiệu trạng thái được sử dụng kết hợp với  $IO/\bar{M}$  và  $DT/\bar{R}$  để giải mã các chu kỳ hoạt động của bus.
- ▶ READY: Tín hiệu báo cho CPU biết tình trạng sẵn sàng của thiết bị ngoại vi hay bộ nhớ.
  - Khi  $READY = 1$ , CPU có thể thực hiện đọc ghi ngay mà không cần chèn thêm các chu kỳ đợi;
  - Khi thiết bị ngoại vi hay bộ nhớ chưa sẵn sàng, chúng gửi  $READY=0$  báo cho CPU kéo dài lệnh đọc ghi bằng cách thêm các chu kỳ đợi.

## Nhóm tín hiệu điều khiển bus:

- ▶ HOLD: Tín hiệu yêu cầu treo CPU để mạch ngoài thực hiện trao đổi dữ liệu với bộ nhớ theo phương pháp truy nhập trực tiếp bộ nhớ.
  - Khi  $HOLD=1$ , CPU sẽ tự treo bằng cách tách ra khỏi bus A, D và một phần bus C để mạch DMAC điều khiển quá trình trao đổi dữ liệu trực tiếp giữ bộ nhớ và thiết bị vào ra.

# Các tín hiệu của CPU (cont.)

- HLDA: Tín hiệu báo cho mạch ngoài biết yêu cầu treo CPU đã được chấp nhận. CPU treo bằng cách tách ra khỏi bus A, D và một số tín hiệu của bus C.
- INTA: Tín hiệu báo cho mạch ngoài biết yêu cầu ngắt INTR được chấp nhận. CPU đưa ra  $\text{INTA}=0$  để báo cho mạch ngoài biết nó đang chờ mạch ngoài đưa số hiệu ngắt lên bus dữ liệu.
- ALE: Xung chốt địa chỉ xác định tín hiệu trên các chân dồn kênh AD là tín hiệu địa chỉ hay dữ liệu. Khi  $\text{ALE}=1$  thì tín hiệu trên các chân dồn kênh AD là tín hiệu địa chỉ.

## Nhóm tín hiệu điều khiển CPU:

- ▶ NMI: Tín hiệu yêu cầu ngắt không che được – không bị hạn chế bởi cờ ngắt IF. Khi nhận được yêu cầu ngắt NMI, CPU hoàn tất lệnh đang thực hiện và chuyển sang chu kỳ phục vụ ngắt.
- ▶ INTR: Tín hiệu yêu cầu ngắt che được – bị hạn chế bởi cờ ngắt IF.
  - Yêu cầu ngắt INTR sẽ bị từ chối khi cờ ngắt  $\text{IF}=0$ .

# Các tín hiệu của CPU (cont.)

- Khi nhận được yêu cầu ngắt INTR và cờ ngắt IF=1, CPU hoàn tất lệnh đang thực hiện và chuyển sang chu kỳ phục vụ ngắt và gửi ra tín hiệu chấp nhận ngắt INTA=0.
- ▶ RESET: tín hiệu khởi động lại 8086/8088. Khi RESET = 1 kéo dài ít nhất trong thời gian 4 chu kỳ đồng hồ thì 8086/8088 bị buộc phải khởi động lại: nó xoá các thanh ghi DS, ES, SS, IP và FR về 0 và bắt đầu thực hiện chương trình tại địa chỉ CS:IP=FFFF:0000H.
- ▶ MN/ $\overline{MX}$ : chân tín hiệu xác định chế độ làm việc của CPU ở chế độ MIN hay MAX.
  - Trong chế độ MIN (MN/ $\overline{MX}$  nối vào nguồn 5V), CPU tự sinh các tín hiệu điều khiển bus;
  - Trong chế độ MAX (MN/ $\overline{MX}$  nối đất), CPU chuyển các tín hiệu trạng thái cho mạch ngoài tạo các tín hiệu điều khiển bus.
- ▶  $\overline{TEST}$ : Tín hiệu  $\overline{TEST}$  được kiểm tra bởi lệnh WAIT. Khi CPU thực hiện lệnh WAIT trong khi  $\overline{TEST} = 1$ , nó sẽ đợi đến khi  $\overline{TEST} = 0$  mới thực hiện lệnh tiếp theo.

# Các tín hiệu của CPU (cont.)

## Nhóm tín hiệu đồng hồ và nguồn:

- ▶ CLK: Xung nhịp đồng hồ cung cấp nhịp làm việc cho CPU.
- ▶  $V_{cc}$ : chân cung cấp nguồn nuôi 5V.
- ▶ GND: Chân nối đất.

## Nhóm các tín hiệu trạng thái:

- ▶ S3, S4: phối hợp cho biết trạng thái truy nhập các thanh ghi đoạn:
  - 00: CPU truy nhập đoạn dữ liệu phụ ES
  - 01: CPU truy nhập đoạn ngắn xếp SS
  - 10: CPU truy nhập đoạn mã hoặc không đoạn nào
  - 11: CPU truy nhập đoạn dữ liệu
- ▶ S5: S5 phản ánh giá trị cờ IF
- ▶ S6: S6 luôn bằng 0

# Các tín hiệu của CPU (cont.)

## Chu kỳ bus

IO/M	DT/R	SS0	Ý nghĩa
0	0	0	Đọc mã lệnh
0	0	1	Đọc bộ nhớ
0	1	0	Ghi bộ nhớ
0	1	1	Buýt rỗi
1	0	0	Chấp nhận yêu cầu ngắt
1	0	1	Đọc thiết bị ngoại vi
1	1	0	Ghi thiết bị ngoại vi
1	1	1	Dừng

# Các tín hiệu của CPU (cont.)

## Chế độ Min/Max

VXL có thể làm việc ở 2 chế độ: Min và Max

### ► Chế độ Min:

- Chân MN/MX nối nguồn 5V
- CPU tự sinh các tín hiệu điều khiển bộ nhớ và các thiết bị ngoại vi truyền thông
- Các tín hiệu: IO/M, WR, INTA, ALE, HOLD, HLDA, DT/R, DEN

### ► Chế độ Max:

- Chân MN/MX nối đất
- CPU gửi các tín hiệu trạng thái đến mạch phụ trợ và các mạch này sinh các tín hiệu điều khiển bộ nhớ và các thiết bị ngoại vi
- Các tín hiệu: RQ/GT0, RQ/GT1, LOCK, S2, S1, S0, QS0, QS1

# Các tín hiệu của CPU (cont.)

## Các tín hiệu riêng của chế độ Max

- ▶  $\overline{RQ}/\overline{GT0}$  và  $\overline{RQ}/\overline{GT1}$  : Các tín hiệu yêu cầu dùng buýt của các bộ xử lý khác hoặc thông báo chấp nhận treo của CPU để cho các bộ vi xử lý khác dùng bus.  $\overline{RQ}/\overline{GT0}$  có mức ưu tiên hơn  $\overline{RQ}/\overline{GT1}$ .
- ▶  $\overline{LOCK}$  : Tín hiệu CPU đưa ra để cấm các bộ xử lý khác trong hệ thống sử dụng bus khi nó đang thực hiện một lệnh có tiếp đầu LOCK.
- ▶ QS0, QS1: Tín hiệu thông báo các trạng thái khác nhau của đệm lệnh (hàng đợi lệnh).
- ▶  $\overline{S2}, \overline{S1}$  và  $\overline{S0}$ : Các chân trạng thái dùng trong chế độ MAX để ghép với mạch điều khiển bus 8288. Các tín hiệu này được 8288 dùng để tạo ra các tín hiệu điều khiển trong các chu kỳ hoạt động của buýt.

# Các tín hiệu của CPU (cont.)

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Chu kỳ điều khiển của buýt	Tín hiệu
0	0	0	Chấp nhận yêu cầu ngắt	INTA
0	0	1	Đọc thiết bị ngoại vi	IORC
0	1	0	Ghi thiết bị ngoại vi	IOWC, $\overline{AIOWC}$
0	1	1	Dừng (halt)	Không
1	0	0	Đọc mã lệnh	MRDC
1	0	1	Đọc bộ nhớ	MRDC
1	1	0	Ghi bộ nhớ	MWTC, $\overline{AMWC}$
1	1	1	Buýt rỗi (nghỉ)	Không

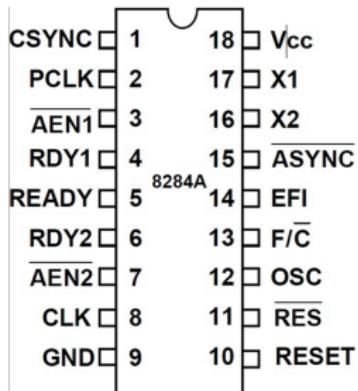
# Các mạch phụ trợ

Các mạch phụ trợ là các mạch cung cấp tín hiệu đầu vào hoặc hỗ trợ CPU điều khiển trong chế độ max.

Các mạch phụ trợ điển hình bao gồm:

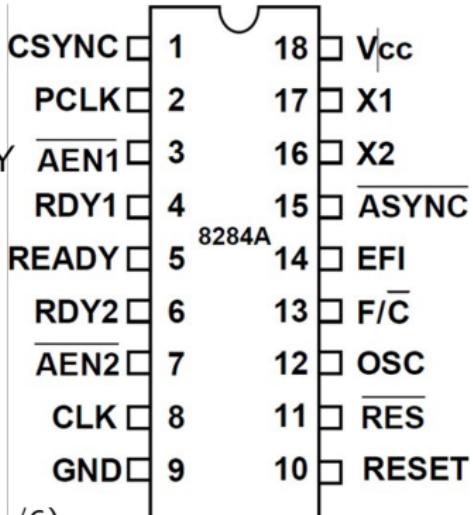
- ▶ Mạch tạo xung nhịp 8284
- ▶ Mạch điều khiển bus 8288

Ví dụ: Mạch tạo xung nhịp 8284



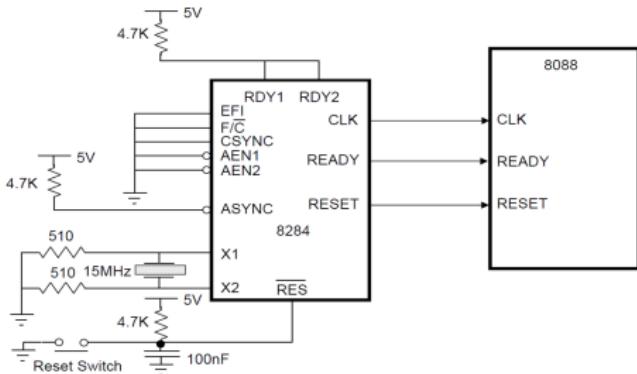
## Mạch phụ trợ - tạo xung nhịp 8284:

- ▶ Cung cấp các tín hiệu CLOCK, READY và RESET ghép nối với CPU.
- ▶ OSC: Xung nhịp đã được khuếch đại có tần số bằng  $f_x$  của bộ dao động.
- ▶ EFI: Lối vào xung nhịp ngoài
- ▶ CLK: Xung nhịp ( $f_{CLK} = f_x/3$ )
- ▶ PCLK: Xung nhịp ngoại vi ( $F_{PCLK} = f_x/6$ )
- ▶ X1, X2: Nối với hai chân của thạch anh với tần số  $f_x$ , thạch anh này là một bộ phận của một mạch dao động bên trong 8284 có nhiệm vụ tạo xung chuẩn dùng làm tín hiệu đồng hồ cho toàn hệ thống.



# Các mạch phụ trợ (cont.)

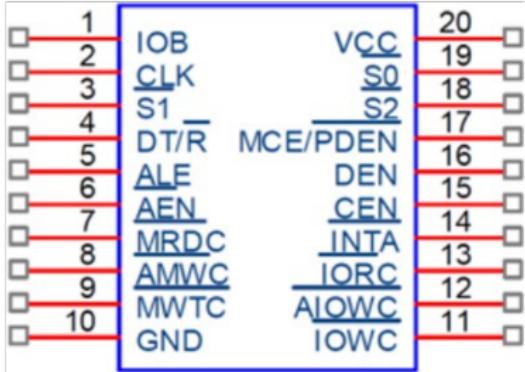
- ▶ AEN1, AEN2: Tín hiệu chọn đầu vào tương ứng RDY1, RDY2 báo tình trạng sẵn sàng của bộ nhớ hoặc thiết bị ngoại vi.
- ▶ RDY1, RDY2: cùng AEN1, AEN2 để tạo các chu kỳ đợi ở CPU.
- ▶ F/C: để chọn nguồn tín hiệu chuẩn cho 8284. Khi chân này ở mức cao thì xung đồng hồ bên ngoài sẽ dùng làm xung nhịp cho 8284, ngược lại xung đồng hồ của mạch dao động bên trong dùng thạch anh được chọn để làm xung nhịp.
- ▶ Mạch tạo xung nhịp 8284 ghép nối với CPU:



# Các mạch phụ trợ (cont.)

## Mạch phụ trợ - điều khiển bus 8288:

- ▶ Mạch điều khiển bus 8288 nhận các tín hiệu trạng thái ( $\overline{S2}$ ,  $\overline{S1}$  và  $\overline{S0}$ ) từ CPU và sinh các tín hiệu điều khiển bus thay cho CPU.
- ▶ 8288 chỉ sử dụng trong chế độ MAX.
- ▶ Các chân tín hiệu:
  - $\overline{S2}$ ,  $\overline{S1}$  và  $\overline{S0}$  : các chân tín hiệu vào trạng thái từ CPU.
  - CLK: Xung đồng hồ lấy từ mạch tạo xung đồng hồ 8284 để tạo nhịp làm việc và đồng bộ với CPU.
  - CEN: Là tín hiệu đầu vào để cho phép đưa ra tín hiệu DEN và các tín hiệu điều khiển khác của 8288.



# Các mạch phụ trợ (cont.)

- IOB: tín hiệu để điều khiển mạch 8288 làm việc ở các chế độ bus khác nhau. Khi  $\text{IOB} = 1$  8288 làm việc ở chế độ bus vào/ ra, khi  $\text{IOB} = 0$  mạch 8288 làm việc ở chế độ bus hệ thống.
- $\overline{\text{MRDC}}$ : tín hiệu điều khiển đọc bộ nhớ. Nó kích hoạt bộ nhớ đưa dữ liệu ra bus.
- $\overline{\text{MWTC}}$ ,  $\overline{\text{AMWC}}$ : các tín hiệu điều khiển ghi bộ nhớ hoặc ghi bộ nhớ kéo dài.  $\overline{\text{AMWC}}$  (advanced memory write command) tương tự như  $\overline{\text{MWTC}}$ , nhưng tăng thêm thời gian ghi dành cho các bộ nhớ có tốc độ chậm.
- $\overline{\text{IORC}}$ : tín hiệu điều khiển đọc thiết bị ngoại vi - kích hoạt các thiết bị được chọn để các thiết bị này đưa dữ liệu ra bus.
- $\overline{\text{IORC}}$ ,  $\overline{\text{AIOC}}$ : các tín hiệu điều khiển ghi thiết bị ngoại vi hoặc ghi thiết bị ngoại vi kéo dài.  $\overline{\text{AIOC}}$  (advanced IO write command) tương tự như  $\overline{\text{IORC}}$ , nhưng tăng thêm thời gian ghi dành cho các thiết bị ngoại vi có tốc độ chậm.

# Các mạch phụ trợ (cont.)

- INTA: Tín hiệu báo cho mạch ngoài biết yêu cầu ngắt INTR được chấp nhận. CPU đưa ra INTA=0 để báo cho mạch ngoài biết nó đang chờ mạch ngoài đưa số hiệu ngắt lên bus dữ liệu.
- ALE: Xung chốt địa chỉ xác định tín hiệu trên các chân dồn kênh AD là tín hiệu địa chỉ hay dữ liệu. Khi ALE=1 thì tín hiệu trên các chân dồn kênh AD là tín hiệu địa chỉ.
- DT/ $\overline{R}$ : Tín hiệu xác định chiều vận chuyển dữ liệu trên bus dữ liệu.
  - ▶ DT/ $\overline{R}$ =1 à dữ liệu đi ra từ CPU;
  - ▶ DT/ $\overline{R}$  =0 à dữ liệu đi đến CPU.
- DEN: đây là tín hiệu để điều khiển buýt dữ liệu trở thành buýt cục bộ hay buýt hệ thống.
- MCE/PDEN: đây là tín hiệu dùng để định chế độ làm việc cho mạch điều khiển ngắt PIC 8259 để nó làm việc ở chế độ chủ.

# Các mạch phụ trợ (cont.)

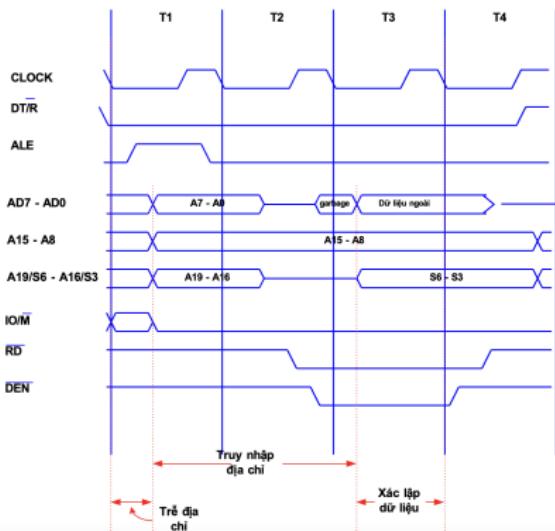
## Định thời và chu trình đọc ghi bus

- ▶ Truy nhập bộ nhớ, vào/ra tính theo chu trình bus. Chu trình bus tiêu biểu gồm 4 xung nhịp đồng hồ (T):
  - Sinh tín hiệu địa chỉ trên bus địa chỉ ( $T_1$ )
  - Sinh tín hiệu đọc/ghi trong xung ( $T_2 - T_3$ )
  - Đọc/Lưu dữ liệu trên bus dữ liệu ( $T_3$ )
- ▶ Để truyền dữ liệu không lỗi, các tín hiệu trên bus cần được tạo và duy trì trong chu trình bus:
  - Biến dạng do trở kháng (tự cảm, điện dung)
  - Trễ tín hiệu khi lan truyền trên bus
  - Hình dạng xung (sườn lên, xuông, độ rộng)
- ▶ 4 xung nhịp đồng hồ:
  - $T_1$ : khởi đầu chu trình. Sinh các tín hiệu điều khiển chốt, kiểu thao tác, hướng dữ liệu và địa chỉ

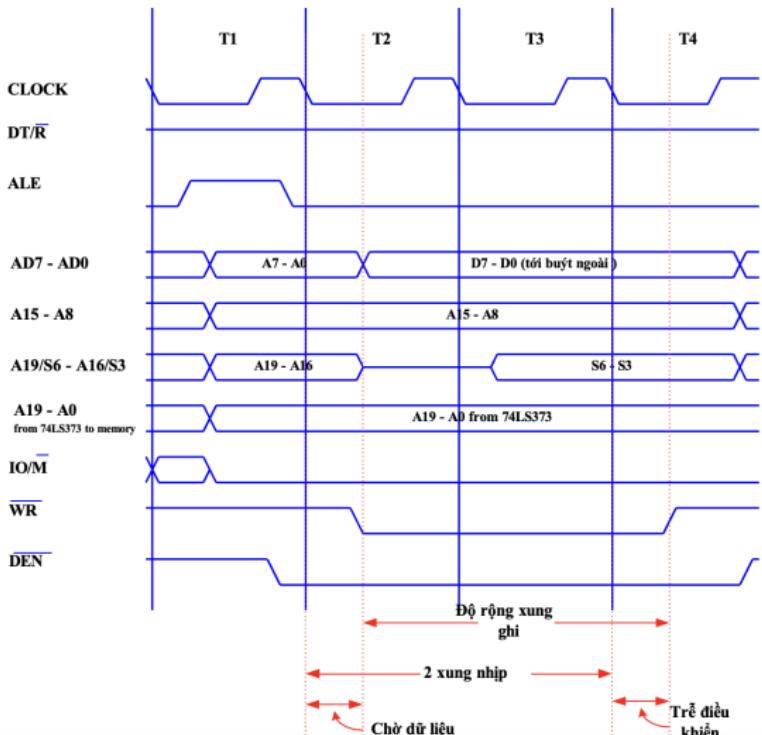
# Các mạch phụ trợ (cont.)

- $T_2$ : sinh tín hiệu điều khiển đọc/ghi. DEN báo dữ liệu ra sẵn sàng. READY báo dữ liệu vào sẵn sàng.
- $T_3$ : Đọc/Ghi dữ liệu
- $T_4$ : Kết thúc các tín hiệu điều khiển

## Chu trình đọc bus:



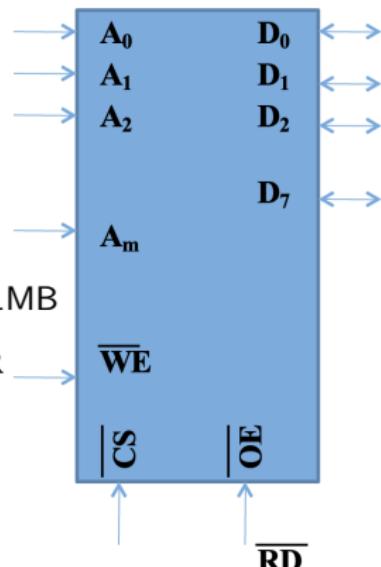
## Chu trình ghi bus:



### Cấu trúc vi mạch nhớ khái quát:

#### ► Nhóm tín hiệu địa chỉ:

- Số đường địa chỉ tương ứng với dung lượng của mạch nhớ
- $m=20$  dung lượng mạch nhớ là  $2^{20} = 1\text{MB}$
- $A_0 - A_m$ : 20 chân đường địa chỉ



#### ► Nhóm tín hiệu dữ liệu:

- Số đường dây dữ liệu quyết định độ dài từ nhớ của mạch nhớ
- $D_0 - D_7$ : 8 chân dữ liệu

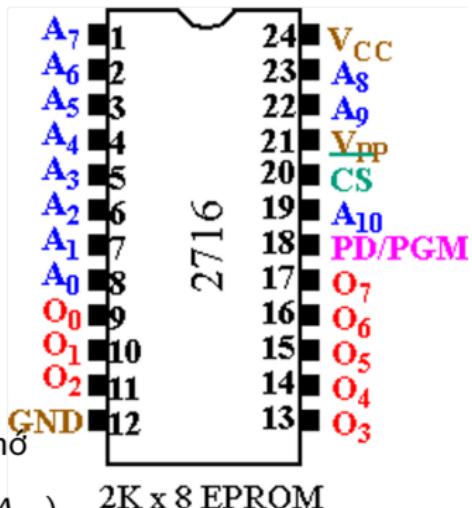
- Thông thường dung lượng và độ dài từ nhớ sẽ được nói cùng lúc: **1Kx8** (dung lượng 1KB, độ dài từ nhớ 8 bit)
- Các tín hiệu WE: Cho phép ghi, OE: Cho phép ra, CS: Kích hoạt

# Phối ghép CPU với bộ nhớ (cont.)

## Giới thiệu bộ nhớ

### Mạch nhớ - EFROM Intel 2176(2Kx8):

- ▶  $A_0 - A_{10}$ : Tín hiệu địa chỉ
- ▶  $O_0 - O_7$ : Tín hiệu dữ liệu
- ▶ CS: chọn chíp (0-đọc,1-ghi)
- ▶ PD/PGM: Duy trì/Lập trình  $V_{pp} = 25V$
- ▶ Chíp nhớ 2Kx8 chiếm không gian  $2^{11}$  ô nhớ
- ▶ Vậy cần 11 bit địa chỉ nội bộ chip ( $A_0 - A_{10}$ )

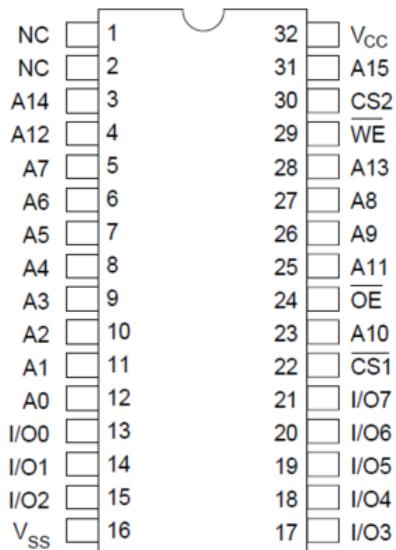


### Cấu trúc mạch nhớ - SRAM

Hitachi HM62864 - 64Kx8

Tốc độ 50-85ns

Pin Name	Function
A0 to A15	Address
I/O0 to I/O7	Input/output
CS1	Chip select 1
CS2	Chip select 2
WE	Write enable
OE	Output enable
NC	No connection
V <sub>cc</sub>	Power supply
V <sub>ss</sub>	Ground

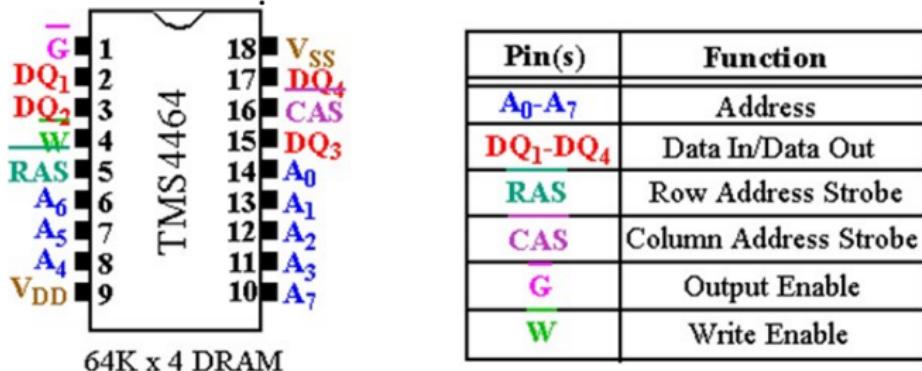


- ▶ Chip nhớ 64Kx8 chiếm không gian  $2^{16}$  ô nhớ
- ▶ Vậy cần 16 bit địa chỉ nội bộ chip ( $A_0 - A_{15}$ )

# Phối ghép CPU với bộ nhớ (cont.)

Giới thiệu bộ nhớ

## Cấu trúc mạch nhớ - DRAM TMS 4464: 64Kx4



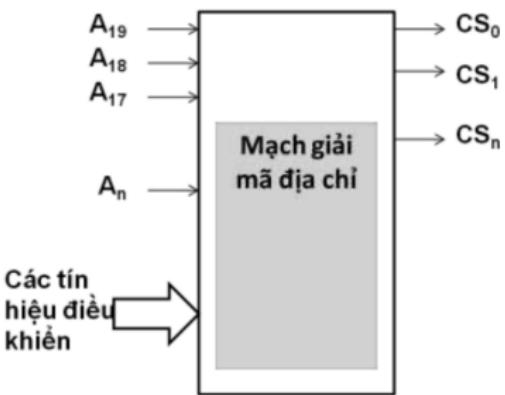
- ▶  $64K = \{RA_0 - RA_7\} + \{CA_0 - CA_7\}$
- ▶ Chip nhớ 64Kx4 chiếm không gian  $2^{16}$  ô nhớ
- ▶ Vậy cần 16 bit địa chỉ nội bộ chip  $\{RA_0 - RA_7\} + \{CA_0 - CA_7\}$  và kích thước của một từ nhớ là 4 bit

## Tại sao cần phối ghép CPU với bộ nhớ?

- ▶ Mỗi mạch nhớ phối ghép với CPU cần được CPU tham chiếu chính xác khi thực hiện thao tác đọc ghi.
- ▶ Mỗi mạch nhớ được gán cho một vùng địa chỉ riêng biệt trong không gian địa chỉ của bộ nhớ.
- ▶ Việc gán địa chỉ cho mạch nhớ thông qua một xung chọn vi mạch lấy từ mạch giải mã địa chỉ.
  - Đầu vào của mạch giải mã:
    - ▶ Tín hiệu địa chỉ 20 bit địa chỉ vật lý
    - ▶ Các tín hiệu điều khiển  $IO/\overline{M}$  và RD (đọc) hoặc WR (ghi)
  - Các loại mạch nhớ sử dụng:
    - ▶ ROM/EPROM
    - ▶ SRAM
    - ▶ DRAM

# Phối ghép CPU với bộ nhớ (cont.)

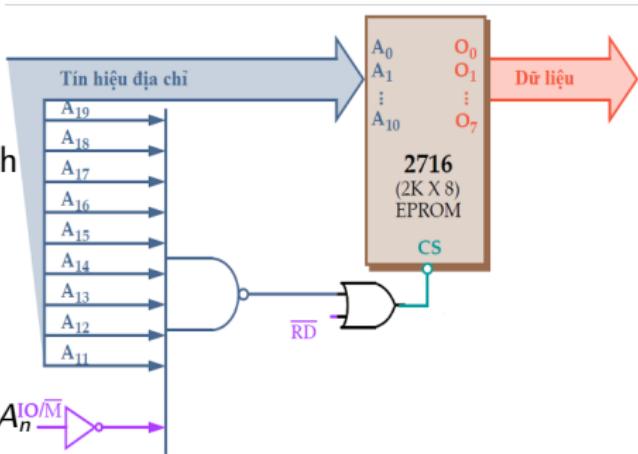
- Mạch phối ghép:



- Mạch logic cơ bản : NAND
- Mạch tích hợp: 74LS139, 74LS138
- Mạch tích hợp EPROM
- Mạch giải mã địa chỉ để chọn riêng các mạch nhớ đã định trước

## Mạch giải mã địa chỉ bộ nhớ:

- ▶ Ánh xạ các tín hiệu địa chỉ thành tín hiệu chọn chíp nhớ:  $A_{19}A_{18}\dots A_n \rightarrow CS_0, CS_1, \dots, CS_n$



- Có  $n+1$  chíp nhớ  $CS_i$
- Mạch giải mã lựa chọn  $n+1$  chíp nhớ từ các tín hiệu  $A_{19}\dots A_n$
- ▶ Giải mã đầy đủ:
  - Sử dụng  $A_{19}A_{18}\dots A_n$
  - Tín hiệu đầu ra chọn duy nhất 1 mạch nhớ.
- ▶ Giải mã rút gọn:
  - Sử dụng  $A_{19}A_{18}\dots A_n$ ;  $m > n$  đầu ra có thể chọn nhiều hơn 1 mạch nhớ.

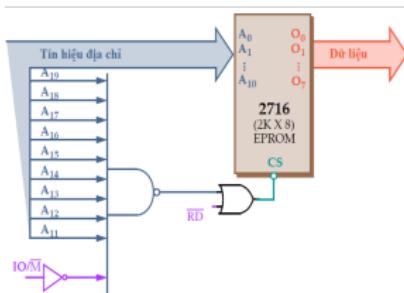
### Các bước thực hiện giải mã địa chỉ tổng quát

Xây dựng mạch giải mã địa chỉ cho 1 bộ nhớ ROM có dung lượng  $C_{IC}$ ; Biết rằng kích thước 1 vi mạch nhớ là  $IC$  và địa chỉ cơ sở là  $DCCS$

- ▶ **Bước 1:** Xác định số bit cho địa chỉ nội bộ chip và mạch giải mã  
Số bit cho nội bộ chíp là  $m = \log_2 IC \Rightarrow$  số bit cần cho mạch giải mã là  $20 - m$
- ▶ **Bước 2:** Phân giải địa chỉ cơ sở của các chíp  
Số lượng chíp nhớ là  $n = \frac{C_{IC}}{IC} \Rightarrow$  Dung lượng của 1 chíp nhớ là  $IC$   
Một chíp nhớ có Địa chỉ cuối = Địa chỉ đầu + Dung lượng chíp - 1  
*Địa chỉ của IC 1:* Từ  $DCCS$  đến  $(DCCS + IC - 1H)$   
*Địa chỉ của IC 2:* Từ  $(DCCS + IC)$  đến  $(DCCS + 2xIC - 1H)$   
...
- ▶ **Bước 3:** Lập bảng bit
- ▶ **Bước 4:** Vẽ hình mạch giải mã

## Giải mã địa chỉ bộ nhớ sử dụng mạch logic cơ bản:

- ▶ Các mạch cơ bản bao gồm AND, OR, XOR, NOT hay NAND, NOR



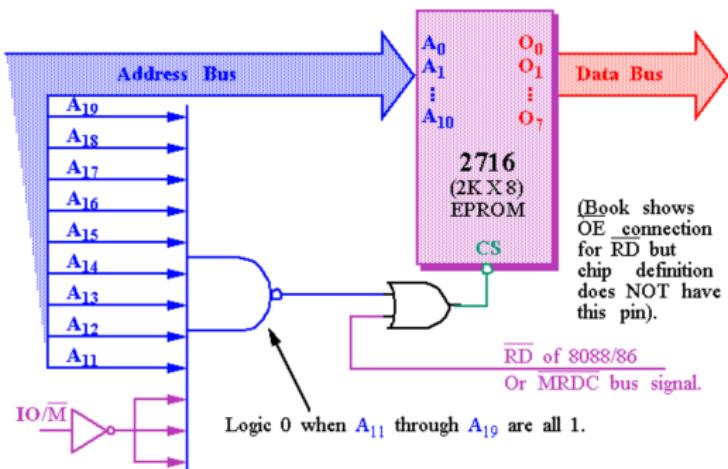
- ▶ Xây dựng mạch giải mã địa chỉ đơn giản với số đầu ra hạn chế
- ▶ Ưu điểm:
  - Cho phép tạo mạch giải mã đầy đủ
  - Tương đối đơn giản rẻ tiền khi chỉ cần 1 hoặc ít đầu ra.
- ▶ Nhược điểm:
  - Cồng kềnh khi cần giải mã cho nhiều đầu ra do số mạch tăng nhanh.

## Giải mã địa chỉ bộ nhớ sử dụng mạch logic cơ bản:

Xây dựng mạch giải mã địa chỉ cho 1 bộ nhớ ROM có dung lượng 4KB bằng phương pháp sử dụng mạch logic cơ bản;

Biết rằng kích thước 1 vi mạch nhớ là  $2K \times 8$  và địa chỉ cơ sở là  $03800H$ ;

$$C_{ROM} = 4KB; IC = 2K \times 8; DCCS = 03800H$$



**Giải mã địa chỉ bộ nhớ sử dụng mạch logic cơ bản:**  
 $C_{ROM} = 4KB$ ;  $IC = 2K \times 8$ ;  $DCCS = 03800H$

- ▶ **Bước 1:** Xác định số bit cho địa chỉ nội bộ chip và mạch giải mã  
Số bit cho nội bộ chíp là  $m = \log_2 IC \Rightarrow$  số bit cần cho mạch giải mã là  $20 - m$
- ▶ **Bước 2:** Phân giải địa chỉ cơ sở của các chíp  
Số lượng chíp nhớ là  $n = \frac{C_{IC}}{IC} \Rightarrow$  Dung lượng của 1 chíp nhớ là IC  
Một chíp nhớ có Địa chỉ cuối = Địa chỉ đầu + Dung lượng chíp - 1  
*Địa chỉ của IC 1:* Từ  $DCCS$  đến  $(DCCS + IC - 1H)$   
*Địa chỉ của IC 2:* Từ  $(DCCS + IC)$  đến  $(DCCS + 2 \times IC - 1H)$   
...
- ▶ **Bước 3:** Lập bảng bit
- ▶ **Bước 4:** Vẽ hình mạch giải mã

**Giải mã địa chỉ bộ nhớ sử dụng mạch logic cơ bản:**

$$C_{ROM} = 4KB; IC = 2Kx8; DCCS = 03800H$$

► **Bước 1:** Xác định số bit cho địa chỉ nội bộ chip và mạch giải mã

- Chíp nhớ IC 2Kx8 chiếm không gian  $2KB = 2^1 \times 2^{10} = 2^{11}B$   
⇒ 11 bit cho địa chỉ nội bộ chíp ( $A_0 - A_{10}$ )
- Vi xử lý 8086 có 20 bit địa chỉ nên số bit cho mạch giải mã là:  
⇒  $20 - 11 = 9$  bit cho mạch giải mã ( $A_{11} - A_{19}$ )

► **Bước 2:** Phân giải địa chỉ cơ sở của các chíp

- Bộ nhớ ROM có dung lượng  $C_{ROM} = 4KB$  và mỗi chíp nhớ có dung lượng 2KB  
⇒ Số lượng chíp nhớ 2Kx8 là  $\frac{4KB}{2KB} = 2$  tương ứng IC1 và IC2
- Đổi kích thước của một chíp nhớ sang hệ 16:  
 $2KB = 2^{11} = 0000\ 0000\ 1000\ 0000\ 0000B = 00800H$   
⇒ Dung lượng một chíp nhớ là 00800H
- Địa chỉ của một chíp nhớ:  
Địa chỉ cuối = Địa chỉ đầu + Dung lượng chíp nhớ - 1

- Dải địa chỉ của IC1: Từ 03800H đến (03800H + 00800H - 1)  
: Từ 03800H đến 03FFFFH
- Dải địa chỉ của IC2: Từ 04000H đến 047FFFFH

► **Bước 3:** Lập bảng bit từ địa chỉ cơ sở của các chip

		9 bit cho mạch giải mã ( $A_{19} - A_{11}$ )									11 bit địa chỉ nội bộ chip ( $A_{10} - A_0$ )														
IC1	03800H	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IC2	04000H	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

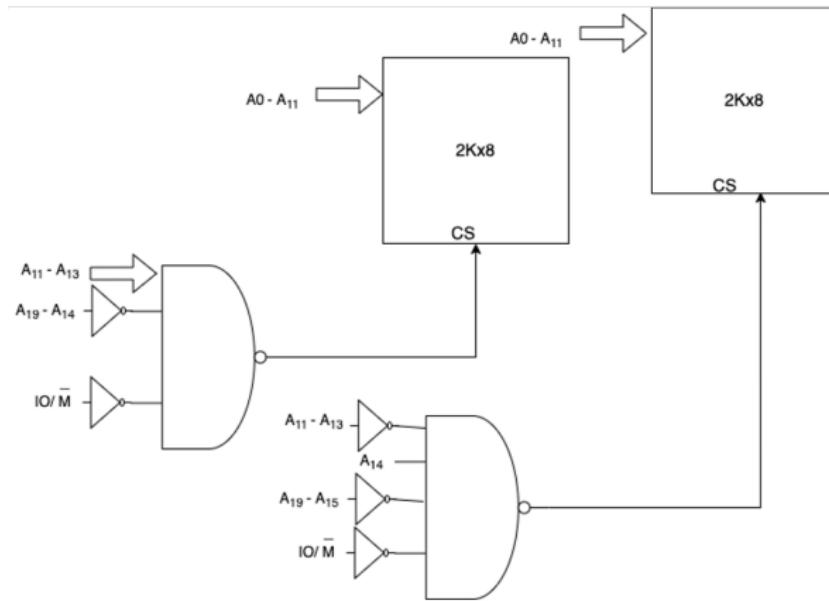
► **Bước 4:** Vẽ mạch giải mã địa chỉ:

		9 bit cho mạch giải mã ( $A_{19} - A_{11}$ )									11 bit địa chỉ nội bộ chip ( $A_{10} - A_0$ )														
IC1	03800H	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IC2	04000H	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		$A_{19}$	$A_{18}$	$A_{17}$	$A_{16}$	$A_{15}$	$A_{14}$	$A_{13}$	$A_{12}$	$A_{11}$	$A_{10}$	$A_9$	$A_8$	$A_7$	$A_6$	$A_5$	$A_4$	$A_3$	$A_2$	$A_1$	$A_0$				

# Phối ghép CPU với bộ nhớ (cont.)

► Xét 9 bit cao (A14 – A19) :

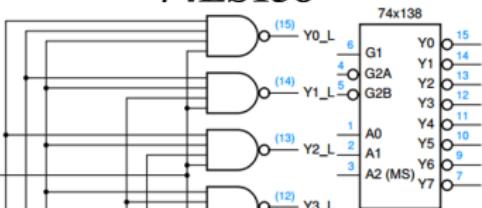
- Một cho 03800H
- Một cho 04000H



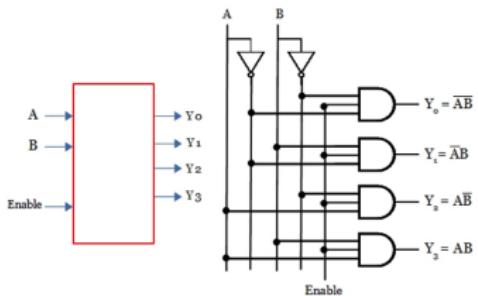
## Giải mã địa chỉ bộ nhớ sử dụng mạch tích hợp:

- 74LS138 mạch giải mã  $3 \rightarrow 8$
- 74LS139 mạch giải mã  $2 \rightarrow 4$

**74LS138**



**74LS139**



## Giải mã địa chỉ bộ nhớ sử dụng mạch tích hợp 74LS139:

Xây dựng bộ giải mã địa chỉ bộ nhớ có dung lượng 8KB có địa chỉ bắt đầu là  $0F800H$  với các chíp nhớ có dung lượng  $2K \times 8$ . Chỉ sử dụng các chíp giải mã địa chỉ 74LS139 (các chíp giải mã có 2 đầu vào và 4 đầu ra).

$$C_{ROM} = 8KB; IC = 2K \times 8; DCCS = 0F800H$$

### ► Bước 1: Xác định số bit cho địa chỉ nội bộ chíp và mạch giải mã

- Chíp nhớ IC  $2K \times 8$  chiếm không gian  $2KB = 2^1 \times 2^{10} = 2^{11}B$   
 $\Rightarrow$  11 bit cho địa chỉ nội bộ chíp ( $A_0 - A_{10}$ )
- Vi xử lý 8086 có 20 bit địa chỉ nên số bit cho mạch giải mã là:  
 $\Rightarrow 20 - 11 = 9$  bit cho mạch giải mã ( $A_{11} - A_{19}$ )

### ► Bước 2: Phân giải địa chỉ cơ sở của các chíp

- Bộ nhớ ROM có dung lượng  $C_{ROM} = 4KB$  và mỗi chíp nhớ có dung lượng  $2KB$   
 $\Rightarrow$  Số lượng chíp nhớ  $2K \times 8$  là  $\frac{8KB}{2KB} = 4$  tương ứng IC1, IC2, IC3 và IC4

# Phối ghép CPU với bộ nhớ (cont.)

- Đổi kích thước của một chíp nhớ sang hệ 16:  
 $2KB = 2^{11} = 0000\ 0000\ 1000\ 0000\ 0000B = 00800H$   
 $\Rightarrow$  Dung lượng một chíp nhớ là 00800H
- Địa chỉ của một chíp nhớ:  
 Địa chỉ cuối = Địa chỉ đầu + Dung lượng chíp nhớ - 1  
 Dải địa chỉ của IC1: Từ 0F800H đến 0FFFFH  
 Dải địa chỉ của IC2: Từ 10000H đến 107FFFH  
 Dải địa chỉ của IC3: Từ 10800H đến 10FFFFH  
 Dải địa chỉ của IC4: Từ 11000H đến 107FFFH

▶ **Bước 3:** Lập bảng bit từ địa chỉ cơ sở của các chíp 74LS139 là chíp giải mã có 2 đầu vào và 4 đầu ra:

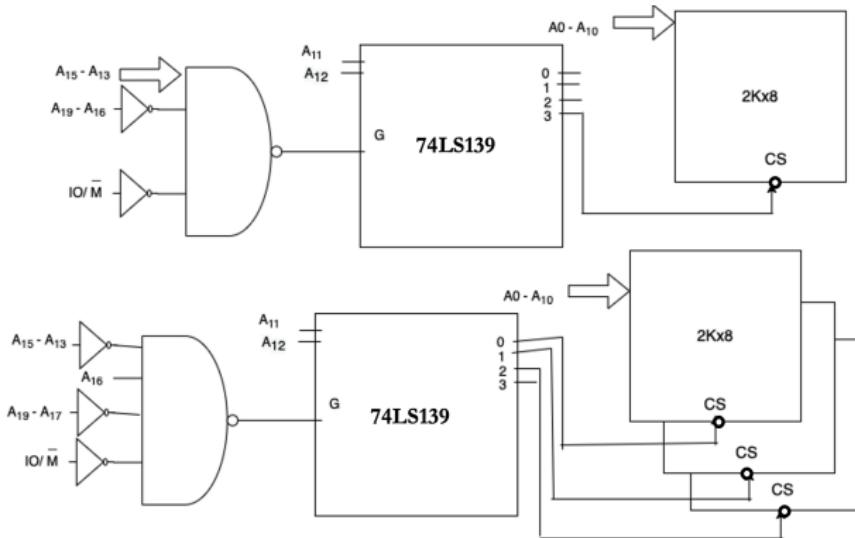
		7 bit cao							2 bit vào		11 bit địa chỉ nội bộ chíp ( $A_0 - A_{10}$ )												
IC1	0F800H	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
IC2	10000H	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IC3	10800H	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
IC4	11000H	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

# Phối ghép CPU với bộ nhớ (cont.)

## ► Bước 4: Vẽ mạch giải mã địa chỉ:

Xét 7 bit cao ( $A_{13} - A_{19}$ ) cần 2 chip giải mã 74LS139:

- 1 cho IC1: 0F800H
- 1 cho IC2, IC3 và IC4: 10000H, 10800H, 11000H (7 bit giống nhau)



## Giải mã địa chỉ bộ nhớ sử dụng mạch tích hợp 74LS138:

Xây dựng bộ giải mã địa chỉ bộ nhớ có dung lượng 8KB có địa chỉ bắt đầu là  $0F800H$  với các chíp nhớ có dung lượng  $2K \times 8$ . Chỉ sử dụng các chíp giải mã địa chỉ 74LS138 (các chíp giải mã có 3 đầu vào và 8 đầu ra).

$$C_{ROM} = 8KB; IC = 2K \times 8; DCCS = 0F800H$$

### ► Bước 1: Xác định số bit cho địa chỉ nội bộ chíp và mạch giải mã

- Chíp nhớ IC  $2K \times 8$  chiếm không gian  $2KB = 2^1 \times 2^{10} = 2^{11}B$   
 $\Rightarrow$  11 bit cho địa chỉ nội bộ chíp ( $A_0 - A_{10}$ )
- Vi xử lý 8086 có 20 bit địa chỉ nên số bit cho mạch giải mã là:  
 $\Rightarrow 20 - 11 = 9$  bit cho mạch giải mã ( $A_{11} - A_{19}$ )

### ► Bước 2: Phân giải địa chỉ cơ sở của các chíp

- Bộ nhớ ROM có dung lượng  $C_{ROM} = 4KB$  và mỗi chíp nhớ có dung lượng  $2KB$   
 $\Rightarrow$  Số lượng chíp nhớ  $2K \times 8$  là  $\frac{8KB}{2KB} = 4$  tương ứng IC1, IC2, IC3 và IC4

# Phối ghép CPU với bộ nhớ (cont.)

- Đổi kích thước của một chíp nhớ sang hệ 16:  
 $2KB = 2^{11} = 0000\ 0000\ 1000\ 0000\ 0000B = 00800H$   
 $\Rightarrow$  Dung lượng một chíp nhớ là 00800H
- Địa chỉ của một chíp nhớ:  
 Địa chỉ cuối = Địa chỉ đầu + Dung lượng chíp nhớ - 1  
 Dải địa chỉ của IC1: Từ 0F800H đến 0FFFFH  
 Dải địa chỉ của IC2: Từ 10000H đến 107FFFH  
 Dải địa chỉ của IC3: Từ 10800H đến 10FFFFH  
 Dải địa chỉ của IC4: Từ 11000H đến 107FFFH

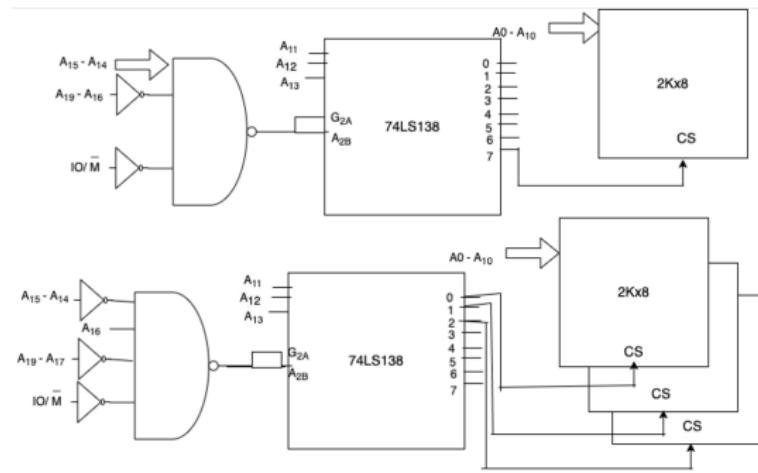
► **Bước 3:** Lập bảng bit từ địa chỉ cơ sở của các chíp  
 74LS138 là chíp giải mã có 3 đầu vào và 8 đầu ra:

		6 bit mạch giải mã					3 bit vào			11 bit địa chỉ nội bộ chíp ( $A_0 - A_{10}$ )										
IC1	<b>OF800H</b>	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0		
IC2	<b>10000H</b>	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0		
IC3	<b>10800H</b>	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0		
IC4	<b>11000H</b>	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0		

► **Bước 4:** Vẽ mạch giải mã địa chỉ:

Xét 6 bit cao (A14 – A19) cần 2 chip giải mã 74LS138:

- 1 cho IC1: 0F800H
  - 1 cho IC2, IC3 và IC4: 10000H, 10800H, 11000H (7 bit giống nhau)

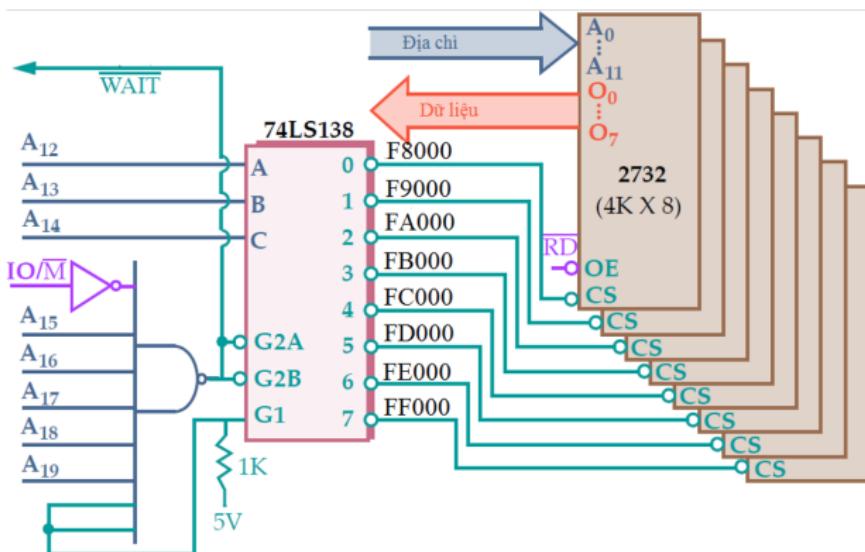


## Giải mã địa chỉ bộ nhớ sử dụng Chip nhớ EPROM:

- ▶ Chip nhớ EPROM 4Kx8
- ▶ Khoảng địa chỉ cấp: F8000 - FFFFFH (32KB) tương ứng với cần 8 chip nhớ
- ▶ Số đường tín hiệu nội bộ một chip nhớ EPROM tương ứng 4KB =  $2^{12}$  là 12 bit ( $A_0 - A_{11}$ ).
- ▶ Đường tín hiệu để cho mạch giải mã / tín hiệu chọn chip là 20 -12 = 8 :  $A_{19}...A_{16}A_{15}...A_{12}$
- ▶ 8 chip nhớ EPROM 4Kx8 có địa chỉ cơ sở lần lượt là F8000H, F9000H, ..., FE000H, FF000H.
  - Các chip này có địa chỉ cơ sở 5 tín hiệu  $A_{19}A_{18}A_{17}A_{16}A_{15}$  không đổi và luôn bằng 1.
  - Còn 3 tín hiệu  $A_{14}A_{13}A_{12}$  khác nhau.

# Phối ghép CPU với bộ nhớ (cont.)

- $A_{14}A_{13}A_{12}$  đưa vào các đầu vào A, B, C của mạch giải mã, còn các tín hiệu địa chỉ còn lại  $A_{19}A_{18}A_{17}A_{16}A_{15}$  và tín hiệu điều khiển IO/M được nối vào tín hiệu điều khiển của 74LS138 (G2A, G2B). Tín hiệu G1 luôn ở mức lô-gíc 1. Các đầu ra của 74LS138 được nối lần lượt với các mạch nhớ ứng với dải địa chỉ gán trước.



# Phối ghép CPU với bộ nhớ (cont.)

## Giải mã địa chỉ bộ nhớ sử dụng Chíp nhớ EPROM:

► Ưu điểm:

- Cho phép tạo mạch giải mã đầy đủ
- Cho phép tạo mạch giải mã chấp nhận một số hạn chế đầu vào và tạo ra một số hạn chế tín hiệu chọn mạch đầu ra.

► Nhược điểm:

- Không thích hợp với mạch giải mã cần chấp nhận một số lượng lớn tín hiệu đầu vào và sinh ra nhiều tín hiệu đầu ra.
- Cần sử dụng bổ sung mạch logic phụ thì mạch tích hợp mới có thể cho phép giải mã đầy đủ.

## Giải mã địa chỉ bộ nhớ sử dụng Chíp nhớ PROM:

- ▶ Bộ nhớ ROM/PROM có thể được sử dụng làm bộ giải mã do:
  - Chấp nhận một nhóm tín hiệu địa chỉ và điều khiển đầu vào
  - Sinh ra một nhóm các tín hiệu dữ liệu đầu ra; Trạng thái của các tín hiệu dữ liệu này tùy thuộc vào giá trị được lưu vào trong ROM trước đó.
  - Nếu các tín hiệu dữ liệu đầu ra loại trừ lẫn nhau thì chúng có thể được dùng làm các tín hiệu chọn vi mạch nhớ.
- ▶ **Ví dụ:** sử dụng PROM 256 byte để làm bộ giải mã cho các chíp nhớ 2732 4Kx8 vào không gian địa chỉ F8000-FFFFF.

## Giải mã địa chỉ bộ nhớ sử dụng Chip nhớ PROM:

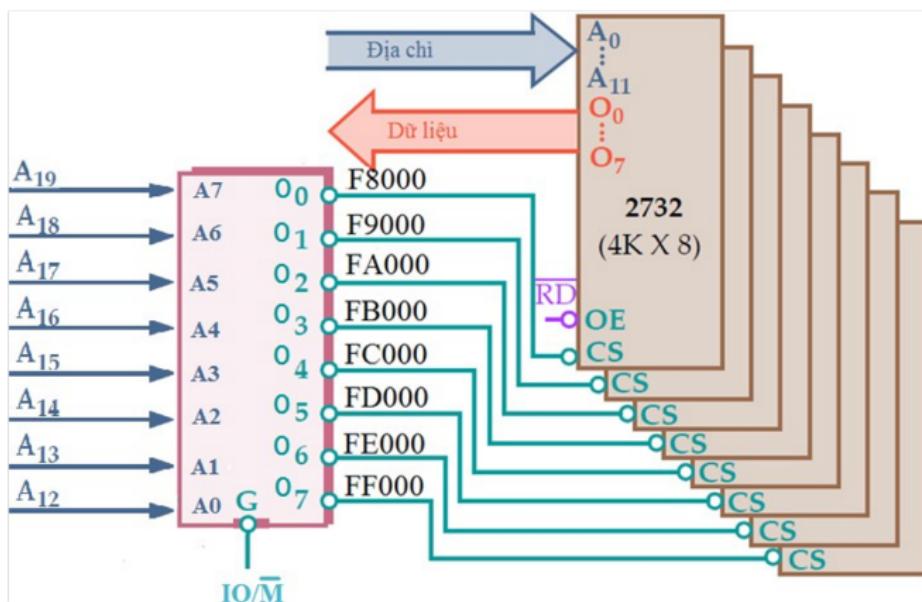
► Mẫu dữ liệu ghi vào PROM 256 bytes:

- Chỉ 8 ô nhớ (nằm trên đường chéo) lưu giá trị ở mức thấp (00)
- Còn tất cả các ô nhớ khác giá trị ở mức cao (FF)

$A_7$	$A_6$	$A_5$	$A_4$	$A_3$	$A_2$	$A_1$	$A_0$	$-O_0$	$-O_1$	$-O_2$	$-O_3$	$-O_4$	$-O_5$	$-O_6$	$-O_7$
1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1
1	1	1	1	1	0	0	1	1	0	1	1	1	1	1	1
1	1	1	1	1	1	0	1	0	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1
1	1	1	1	1	1	0	0	1	1	1	1	0	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1
1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

## Giải mã địa chỉ bộ nhớ sử dụng Chíp nhớ PROM:

- Sử dụng PROM 256 byte để làm bộ giải mã cho các chíp nhớ 2732 4Kx8 vào không gian địa chỉ F8000-FFFFF.

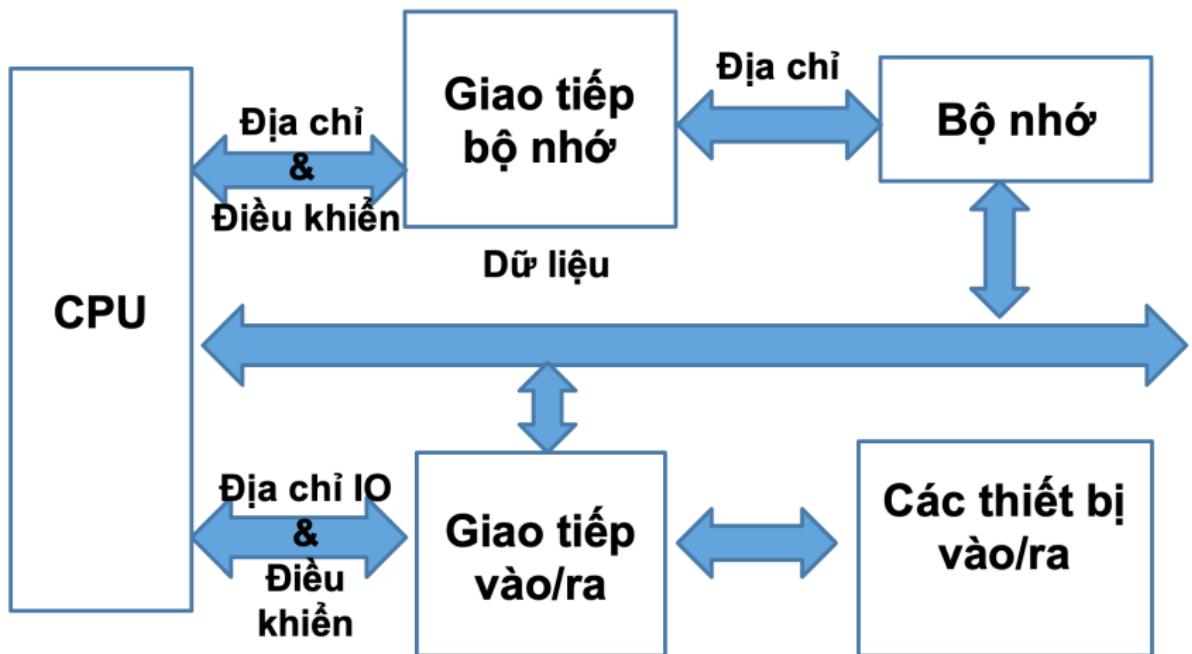


## Giải mã địa chỉ bộ nhớ sử dụng Chíp nhớ PROM:

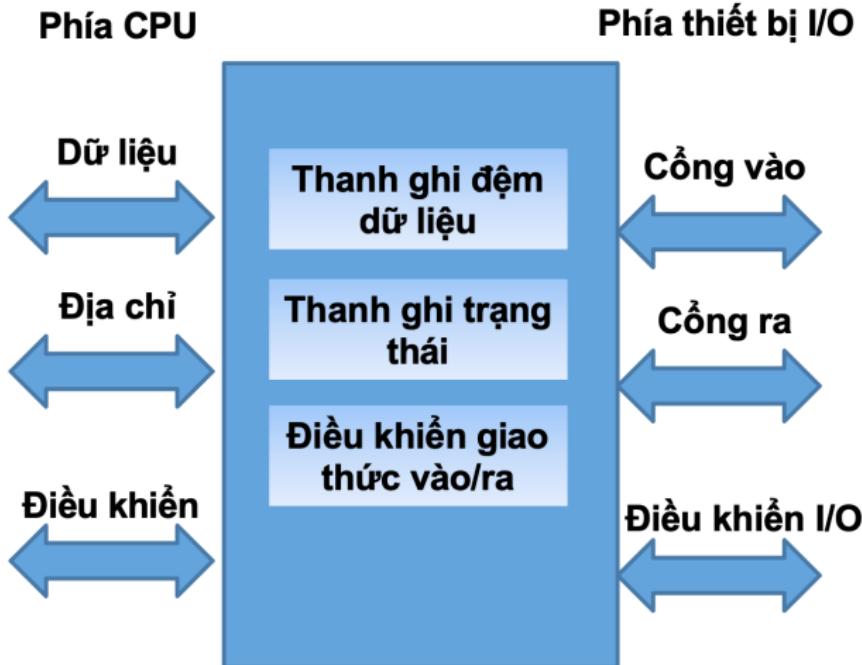
► Ưu điểm của giải mã địa chỉ sử dụng chíp nhớ PROM:

- Cho phép tạo mạch giải mã đầy đủ mà không cần phải sử dụng mạch phụ trợ  $\Rightarrow$  giảm kích thước bộ giải mã.
- Cho phép tạo mạch giải mã chấp nhận nhiều tín hiệu đầu vào và tạo ra một lớn tín hiệu chọn mạch đầu ra.
- Trạng thái của các tín hiệu tùy thuộc vào giá trị được lưu trữ trong ROM, nếu các tín hiệu này loại trừ lẫn nhau thì các tín hiệu có thể được dùng để làm các tín hiệu chọn vi mạch nhớ.
- Dễ dàng thay đổi địa chỉ của các mạch nhớ bằng cách thay đổi vị trí và giá trị dữ liệu trong mạch nhớ giải mã ROM.

## Phối ghép CPU với thiết bị vào ra



## Phối ghép CPU với thiết bị vào ra

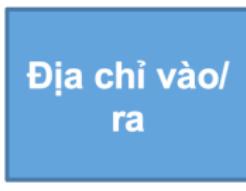


## Phân loại thiết bị vào ra theo không gian địa chỉ:

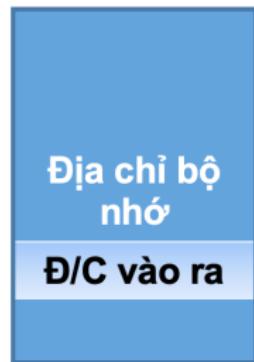
00000



0000



00000



FFFFF

FFFFF

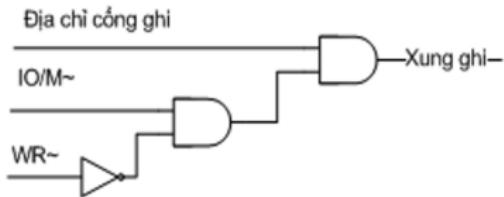
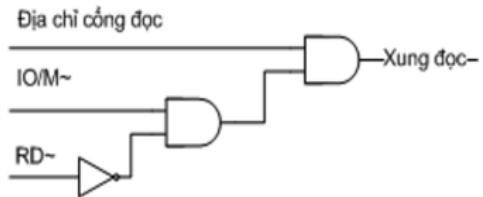
- Thiết bị vào/ra có không gian địa chỉ tách biệt
- Thiết bị vào/ra dùng chung không gian địa chỉ với bộ nhớ

## Phân loại thiết bị vào ra theo không gian địa chỉ:

- ▶ Thiết bị vào/ra có không gian địa chỉ tách biệt:
  - IN AX, [Địa chỉ cổng]
  - OUT [Địa chỉ cổng], AX
  - Địa chỉ cổng vào/ra: 0000-FFFF được lưu trong DX;  
00-FF lưu địa chỉ trực tiếp
- ▶ Thiết bị vào/ra dùng chung không gian địa chỉ với bộ nhớ
  - MOV [Địa chỉ cổng], AX
  - Đọc: MOV AX, [Địa chỉ cổng]
  - Địa chỉ cổng vào/ra 00000-FFFFF

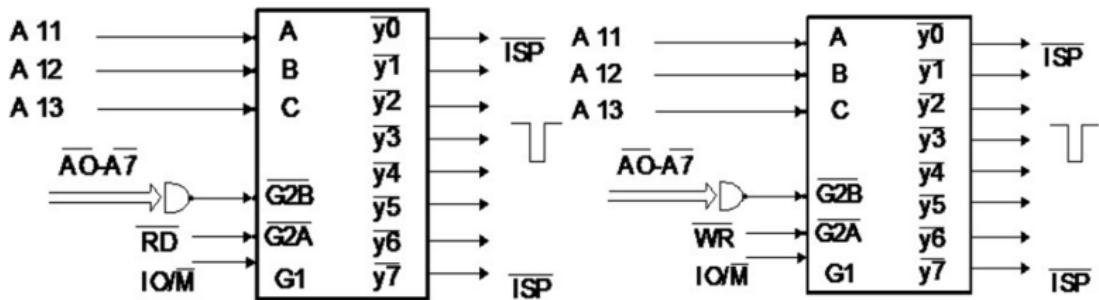
## Giải mã địa chỉ thiết bị vào ra sử dụng cổng logic:

- ▶ Giải mã địa chỉ thiết bị vào ra cũng tương tự như giải mã địa chỉ cho mạch nhớ.
- ▶ Thông thường các cổng có địa chỉ 8 bit ( $A_0 - A_7$ ), trong một số bộ vi xử lý các cổng có thể là 16 bit ( $A_0 - A_{15}$ ).
- ▶ Tổ hợp các tín hiệu địa chỉ và điều khiển thành xung đọc/ghi.
- ▶ Các mạch giải mã đơn giản được tạo bởi các cổng logic như sau:



## Một số mạch cổng đơn giản:

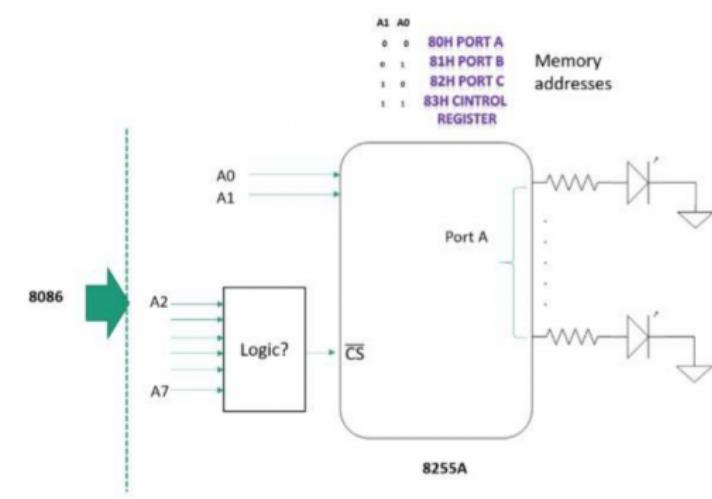
- ▶ Có thể sử dụng các mạch tích hợp cỡ vừa để làm cổng phối ghép với vi xử lý để vào/ra dữ liệu. Các mạch này thường được cấu tạo từ:
  - Các mạch chốt 8 bit có đầu ra 3 trạng thái ( $74LS373, 74LS374$ )
  - Các mạch khuếch đại đếm 2 chiều 8 bit đầu ra 3 trạng thái ( $74LS245$ )



Giải mã địa chỉ cổng dùng 74LS138

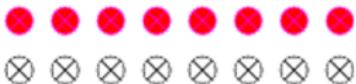
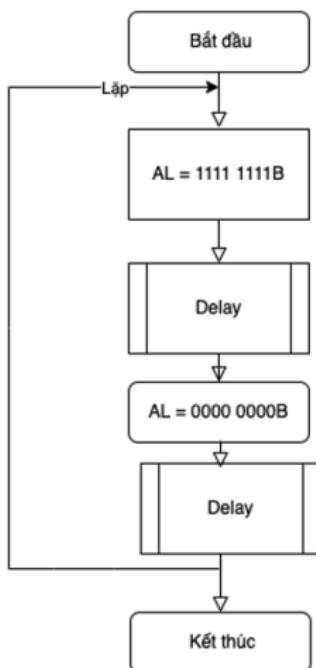
## Chương trình điều khiển LED sáng:

Chương trình 8 LED nối với cổng ra 2C0H sáng rồi tắt. Đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn tắt. Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP.



# Một số mạch cổng đơn giản (cont.)

## Chương trình điều khiển LED sáng:



Đặt giá trị cổng vào = FFH đèn tắt

Làm trễ 64 chu kỳ

Đặt giá trị cổng vào = 00H đèn sáng

Làm trễ 64 chu kỳ

# Một số mạch cổng đơn giản (cont.)

## Chương trình điều khiển LED sáng:

*Chương trình 8 LED nối với cổng ra 2COH sáng rồi tắt. Đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn tắt. Tất cả các đèn đều bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP.*

```

1   DK_LED EQU 2COH ; Cong dieu khien den LED
2
3 LAP: MOV AL, FFH ; Tat tat ca cac den LED
4     MOV DX, DK_LED
5     OUT DX, AL
6     MOV CX, 64 ; Tre 64 chu ky NOP
7 TRE_1: NOP
8     LOOP TRE_1
9     XOR AL, AL ; Dat AL = 0
10    OUT DX, AL ; Bat tat ca cac den
11    MOV CX, 64
12 TRE_2: NOP
13    LOOP TRE_2
14    JMP LAP

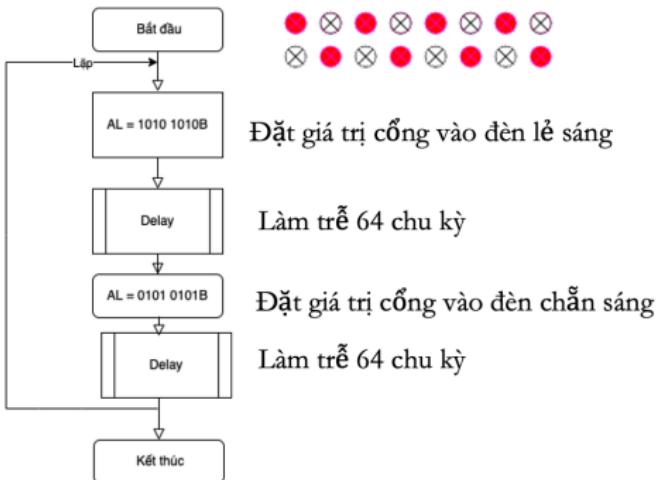
```



# Một số mạch cỗng đơn giản (cont.)

## Chương trình điều khiển LED sáng:

*Chương trình để 8 LED (D0 – D7) nối với cỗng ra 2C0H sáng tắt xen kẽ nhau. Đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn tắt. Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP.*



# Một số mạch cỗng đơn giản (cont.)

## Chương trình điều khiển LED sáng:

Chương trình để 8 LED ( $D0 - D7$ ) nối với cỗng ra  $2COH$  sáng tắt xen kẽ nhau. Đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn tắt. Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP.

```
1      DK_LED EQU 2COH ; Cong dieu khien den LED
2      LAP: MOV AL, 0AAH ; Chi bat cac den D1, D3, D5, D7
3          MOV DX, DK_LED
4          OUT DX, AL
5          MOV CX, 64 ; Tre 64 chu ky NOP
6      TRE_1: NOP
7          LOOP TRE_1
8          MOV AL, 55H ; Dat AL = 0
9          OUT DX, AL ; Chi bat cac den D0, D2, D4, D6
10         MOV CX, 64
11     TRE_2: NOP
12         LOOP TRE_2
13     JMP LAP
```

# Một số mạch cổng đơn giản (cont.)

## Chương trình điều khiển LED sáng:

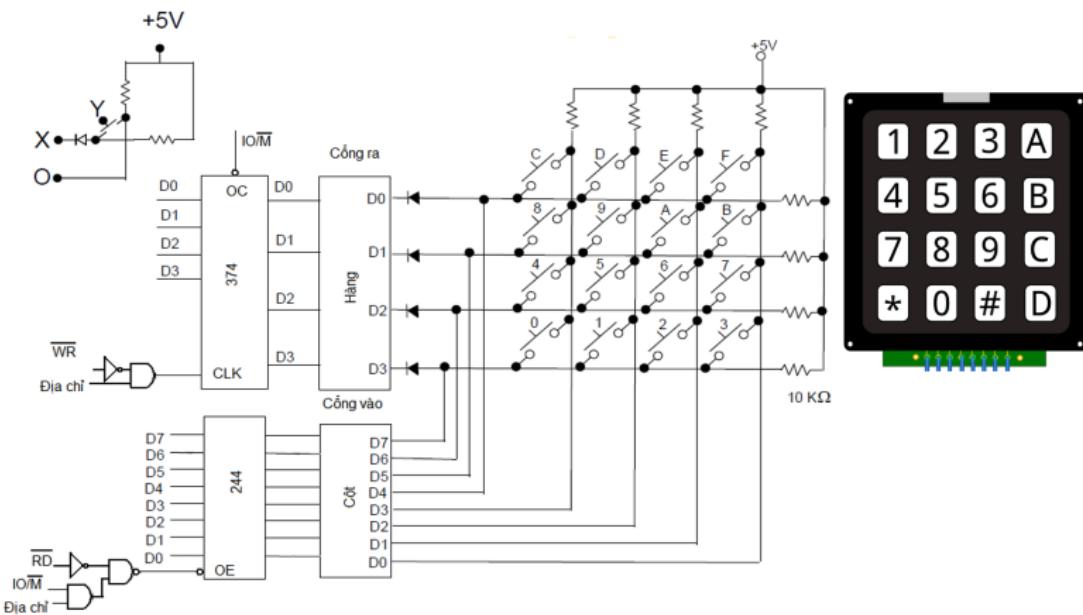
**Bài tập tương tự:** Viết giải thuật và chương trình để 8 LED nối với cổng ra 2C0H sáng tắt. Đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn sẽ tắt.

Minh họa trong hình phía dưới đây, biết tắt cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP

1. Đèn sáng tắt dần      
2. Sâu bòとり                
3. Sáng lặp lượt            

### Một số mạch cổng đơn giản (cont.)

## Ghép nối bàn phím



- ▶ Cổng ghép nối bàn phím 16 số dạng tiếp điểm sử dụng mạch tích hợp:

# Một số mạch cổng đơn giản (cont.)

- Vi mạch 74LS374 được dùng để điều khiển các tín hiệu hàng.
  - Vi mạch 74LS244 được dùng để điều khiển các tín hiệu cột.
- ▶ Nguyên lý hoạt động:
- Tín hiệu X ở mức cao (logic bằng 1) thì Diod khoá lại, tiếp điểm Y tại đầu O thu được điện áp 5V (không có dòng điện).
  - Tín hiệu X ở mức thấp (logic bằng 0) thì Diod mở, khi tiếp điểm Y đóng xuống tại đầu O thu được điện áp 0V.
- ▶ **Xác định được phím ấn:** quét tuần tự các dòng và các cột, nếu một phím được nhấn thì các bit tại cổng ra  $D_i = 0$ .

# Một số mạch cổng đơn giản (cont.)

## Chương trình kiểm tra một phím

Đoạn chương trình kiểm tra xem phím C (hàng D0, cột D3) có được bấm hay không. Biết địa chỉ cổng hàng là 0Ah và địa chỉ cổng cột là 0Bh.

```
1 HANG EQU 0AH ; Dia chi Cong hang
2 COT EQU 0BH ; Dia chi Cong cot
3     MOV AL, 1111 1110 ; Chi co D0 = 0 - hang thu nhat
4     OUT HANG, AL
5 KIEMTRA: IN AL, COT ; Doc tin hieu cot
6     AND AL, 0000 1000B ; Giu lai bit D3 tuong ung voi ban phim C
7     JMP KIEMTRA ; Khong bam
8     ... ; Phim C duoc bam
```

# Một số mạch cổng đơn giản (cont.)

## Ghép nối hiển thị số

► Ghép nối hiển thị số sử dụng mạch tích hợp 7447 và LED bảy đoạn:

- Cổng A điều khiển (bật/tắt) các thanh led thông qua 7 transistor Q1-Q7. Địa chỉ cổng A là 0Ah.
- Cổng B nhận dữ liệu số hiển thị - thông qua mạch 7447 giải mã số đầu vào dạng BCD ở cổng B (A-D) sinh ra các tín hiệu kích hoạt (a-g) các thanh led của LED bảy đoạn. Địa chỉ cổng B là 0Bh.
  - ▶ Bật đèn led thứ i: gửi bit  $D_i = 0$  ra cổng A
  - ▶ Tắt đèn led thứ i: gửi bit  $D_i = 1$  ra cổng A
- Hiển thị số: Gửi số cần hiện thị ra cổng B, bật đèn led i bằng cách gửi bit  $D_i = 0$  ra cổng A.

## Chương 5

- ▶ Các tín hiệu của CPU
- ▶ Các tín hiệu của các mạch phụ trợ
- ▶ Phối ghép CPU với bộ nhớ
- ▶ Phối ghép CPU với thiết bị vào ra
- ▶ Giới thiệu một số mạch hỗ trợ vào ra

## Tiếp theo Chương 6

- ▶ Các phương pháp vào ra dữ liệu
  - Vào ra bằng thăm dò
  - Vào ra bằng ngắt
  - Vào ra bằng truy nhập trực tiếp bộ nhớ DMA
- ▶ Giới thiệu một số hệ vi xử lý tiên tiến

# Câu hỏi và bài tập