HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÀI GIẢNG MÔN

Kiến trúc máy tính

CHƯƠNG 5 – PHỐI GHÉP VỚI BỘ NHỚ VÀ THIẾT BỊ VÀO RA

Giảng viên:

Điện thoại/E-mail:

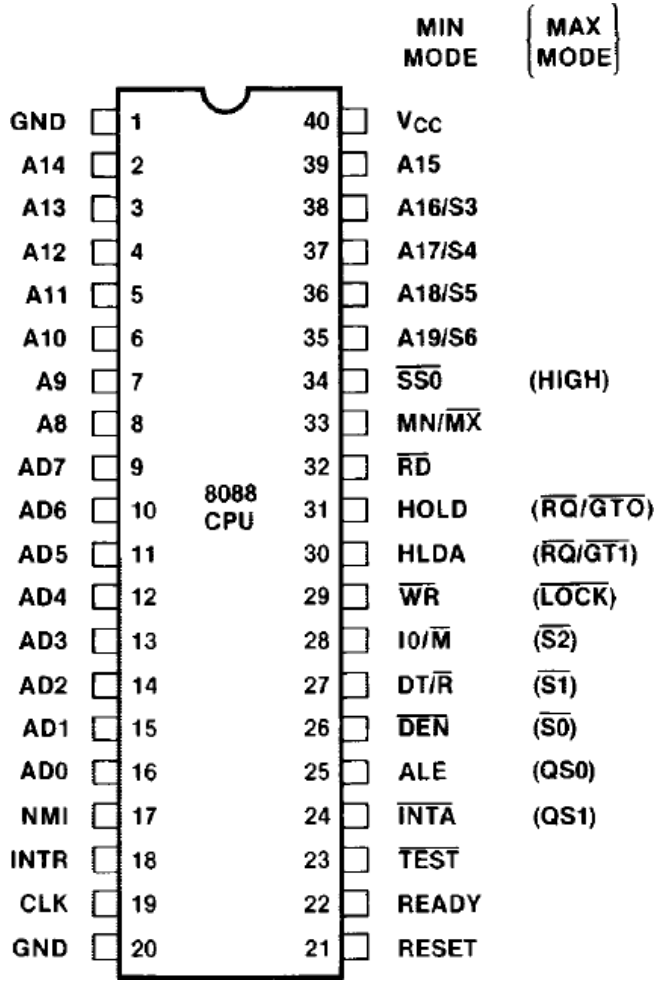
Bộ môn: Khoa học máy tính - Khoa CNTT1

NỘI DUNG

1. Các tín hiệu của CPU
2. Các tín hiệu của các mạch phụ trợ
3. Phối ghép CPU với bộ nhớ
4. Phối ghép CPU với thiết bị vào ra
5. Giới thiệu một số mạch hỗ trợ vào ra

4.1. Các tín hiệu của 8088

- VXL 8088 có có 40 chân tín hiệu, gồm các nhóm:
  - Nhóm tín hiệu địa chỉ:
    - AD_0-AD_7: 8 chân dồn kênh cho phần thấp bus A và bus D ;
    - A_8-A_{15}: 8 chân tín hiệu phân cao bus A
    - A_{16}/S_3-A_{19}/S_6: 4 chân dồn kênh cho phần cao bus A và bus C;
  - Nhóm tín hiệu dữ liệu
    - AD_0-AD_7: 8 chân dồn kênh cho phần thấp bus A và bus D;
    - Khi chân chốt ALE=0→ tín hiệu dữ liệu, ALE=1→ tín hiệu địa chỉ.



4.1. Các tín hiệu của 8088

- 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.

4.1. Các tín hiệu của 8088

- Nhóm tín hiệu điều khiển hệ thống:
  - $\overline{SS}_0$: Tín hiệu trạng thái được sử dụng kết hợp với $IO/\overline{M}$ và $DT/\overline{R}$ 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.

4.1. Các tín hiệu của 8088

- Nhóm tín hiệu điều khiển bus:
  - 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 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.

4.1. Các tín hiệu của 8088

- 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 IF=0. 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.

4.1. Các tín hiệu của 8088

- Nhóm tín hiệu điều khiển CPU:

  - 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; còn 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.

4.1. Các tín hiệu của 8088

- Nhóm tín hiệu đồng hồ và nguồn:
  - CLK: Xung nhịp đồng hồ cung cấp nhịp làm việc cho CPU.
  - Vcc: chân cung cấp nguồn nuôi 5V.
  - GND: Chân nối đất.
  - 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 ngăn 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

4.1. Các tín hiệu của 8088 – Chu kỳ bus

IO/M	DT/R	SS0	
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

4.1. Các tín hiệu của 8088 – 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 thống
  - 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

4.1. Các tín hiệu của 8088 – Chế độ Max

❖ Các tín hiệu riêng của chế độ Max

- $\overline{RQ}$ / $\overline{GT_0}$ và $\overline{RQ}$ / $\overline{GT_1}$: 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. $\overline{RQ}$ / $\overline{GT_0}$ có mức ưu tiên hơn $\overline{RQ}$ / $\overline{GT_1}$.
- $\overline{LOCK}$: 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).

4.1. Các tín hiệu của 8088 – Chế độ Max

- Các tín hiệu riêng của chế độ Max
  - $\bar{S}_2$,$\bar{S}_1$và $\bar{S}_0$: 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.

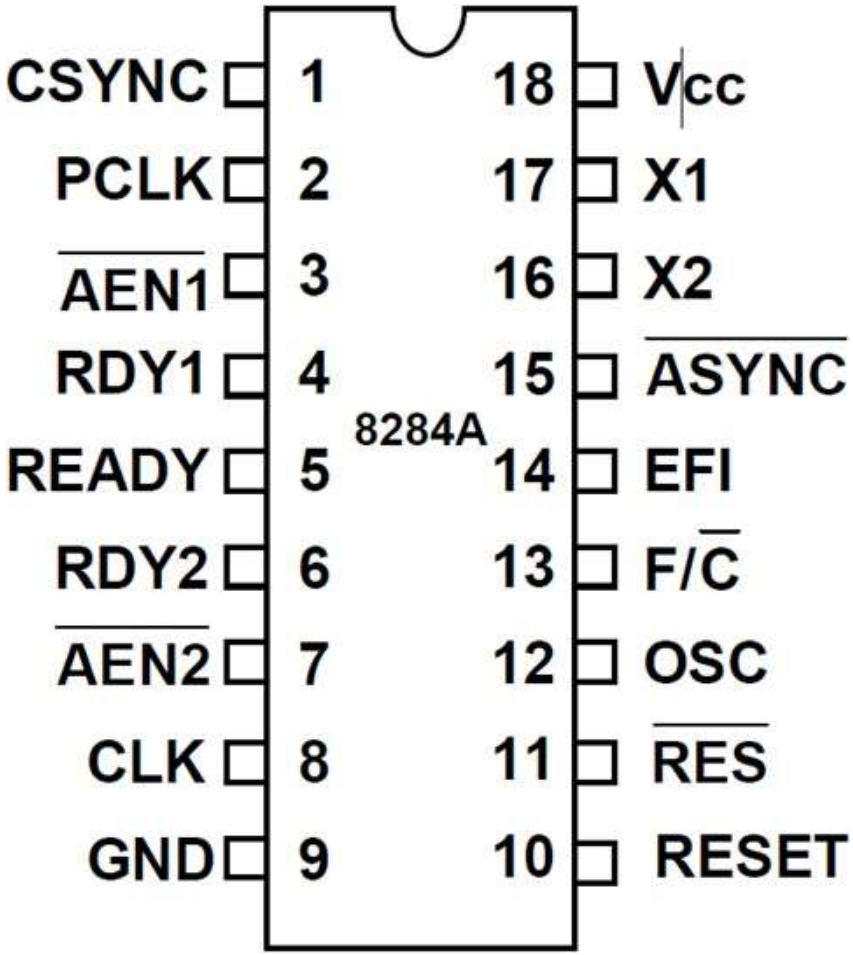
4.1. Các tín hiệu của 8088 – Chế độ Max

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	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 (nghĩ)	Không

4.2 Các mạch phụ trợ

- Là các mạch phụ trợ 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

4.2.1 Mạch tạo xung nhịp 8284



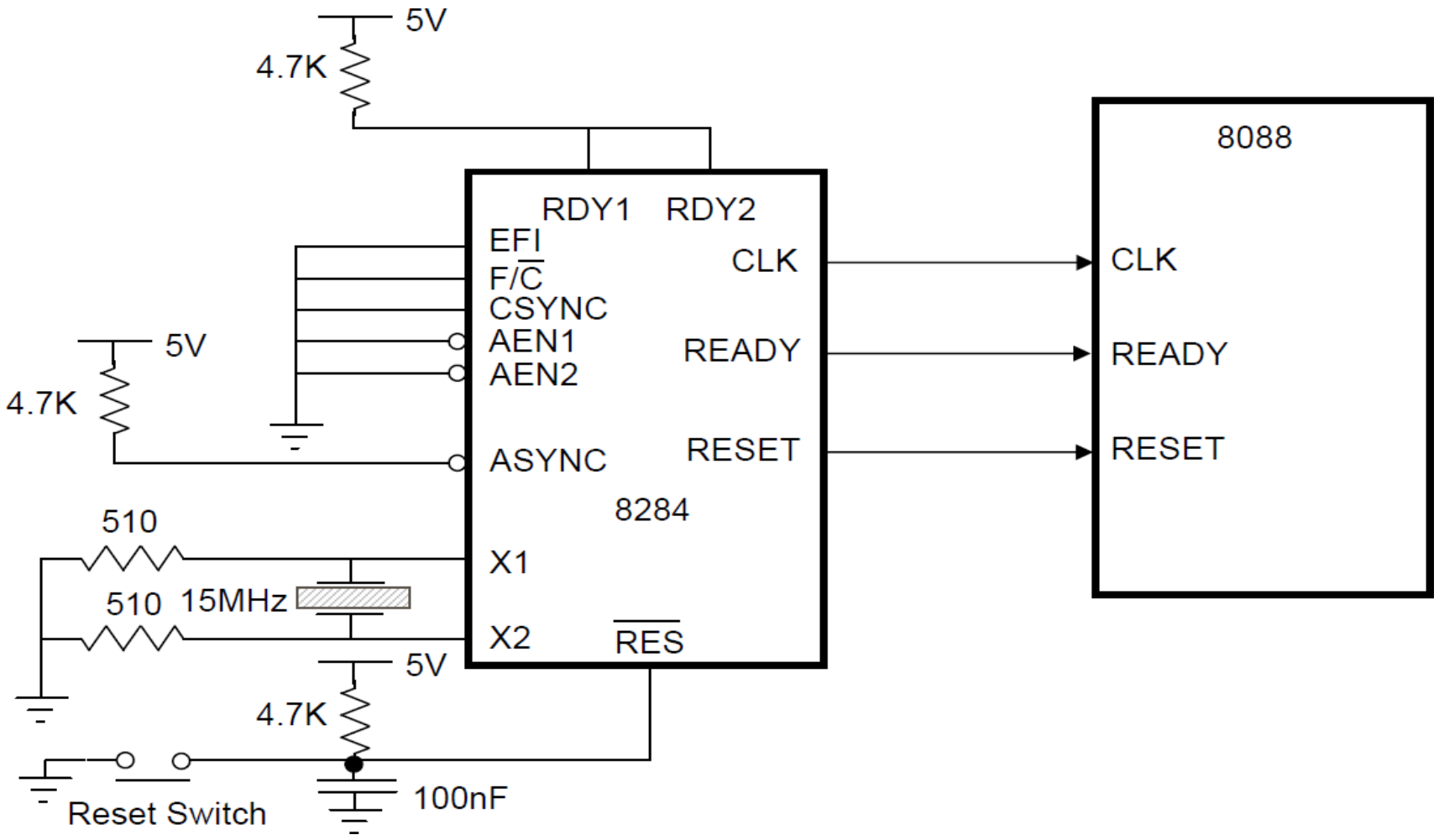
4.2.1 Mạch tạo xung nhịp 8284

- 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 fx, 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.

4.2.1 Mạch tạo xung nhịp 8284

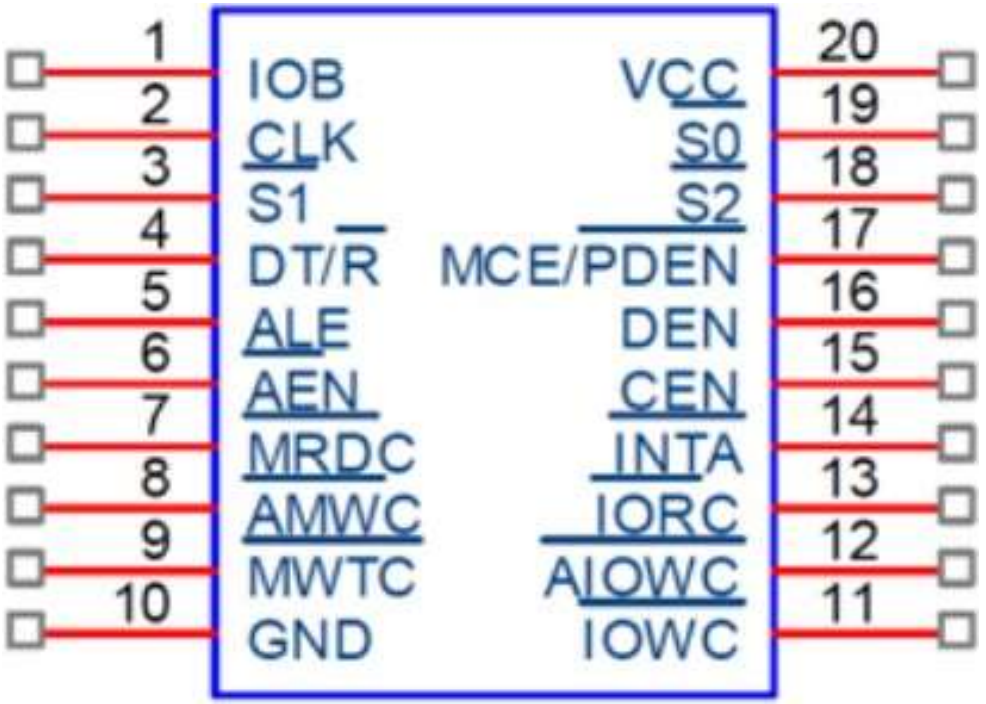
- $\overline{AEN_1}$,$\overline{AEN_2}$: Tín hiệu cho phép chọn đầu vào tương ứng RDY1, RDY2 làm tín hiệu báo tình trạng sẵn sàng của bộ nhớ hoặc thiết bị ngoại vi.
- RDY1, RDY2: cùng với $\overline{AEN_1}$,$\overline{AEN_2}$dùng để tạo ra các chu kỳ đợi ở CPU.
- F/ $\overline{C}$: Dùng để chọn nguồn tín hiệu chuẩn cho 8284. Khi chân này ở mức cao thì xung đồng hồ bên ngoài sẽ được dùng làm xung nhịp cho 8284, ngược lại thì xung đồng hồ của mạch dao động bên trong dùng thạch anh sẽ được chọn để làm xung nhịp.

4.2.1 Mạch tạo xung nhịp 8284 ghép nối với CPU



4.2.2 Mạch điều khiển bus 8288

- Mạch điều khiển bus 8288 nhận các tín hiệu trạng thái ($\bar{S}_2$ $\bar{S}_1$, và $\bar{S}_0$) từ CPU và sinh các tín hiệu điều khiển bus thay cho CPU.
- 8288 chỉ được sử dụng trong chế độ MAX.



4.2.2 Mạch điều khiển bus 8288

❖ Các chân tín hiệu:

- $\overline{S}_2$,$\overline{S}_1$và $\overline{S}_0$: 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.
- IOB: tín hiệu để điều khiển mạch 8288 làm việc ở các chế độ bus khác nhau. Khi IOB = 1 8288 làm việc ở chế độ bus vào/ ra, khi 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.

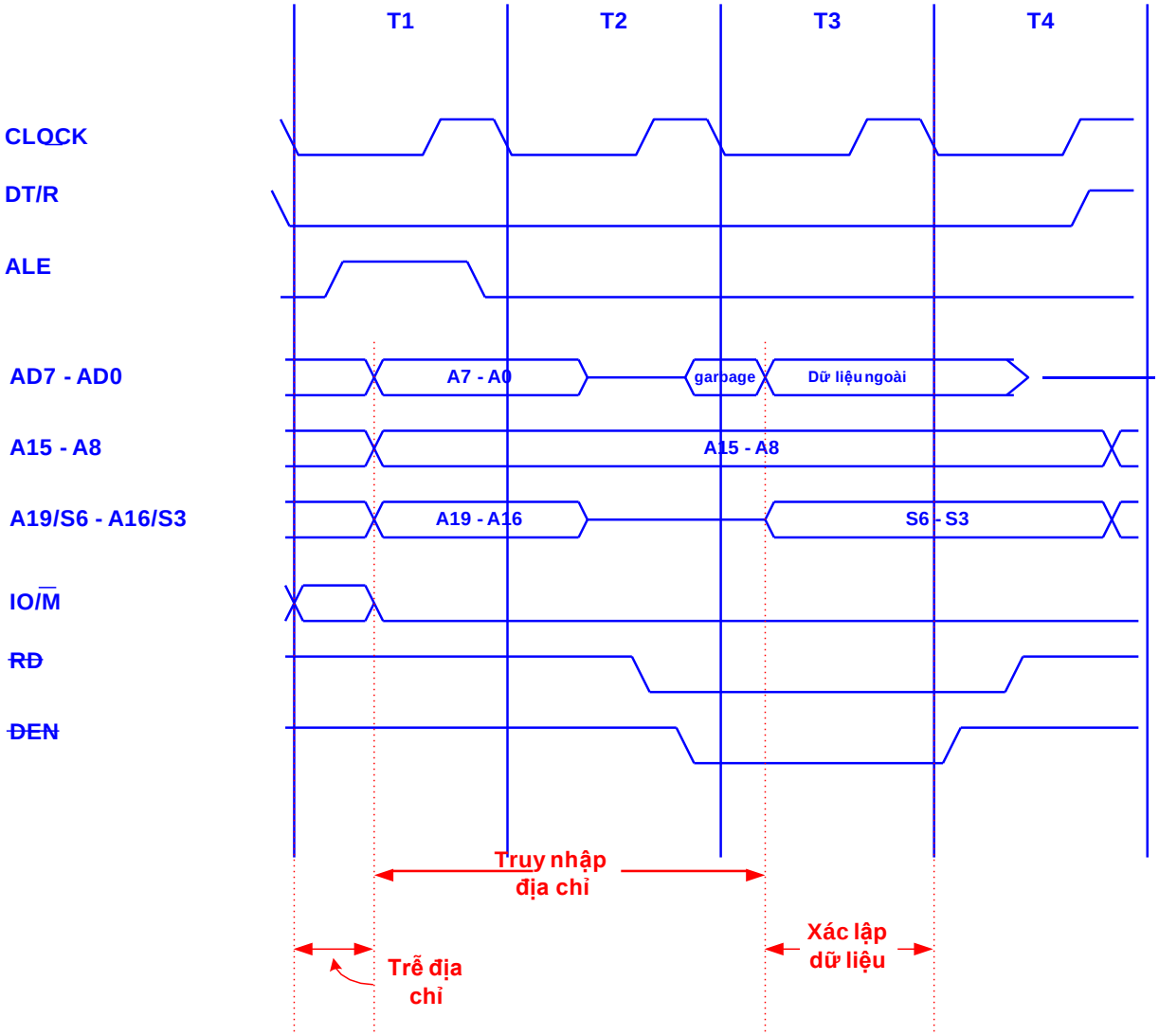
4.2.3 Định thời và chu trình đọc ghi bus

- 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, xuống, độ rộng)

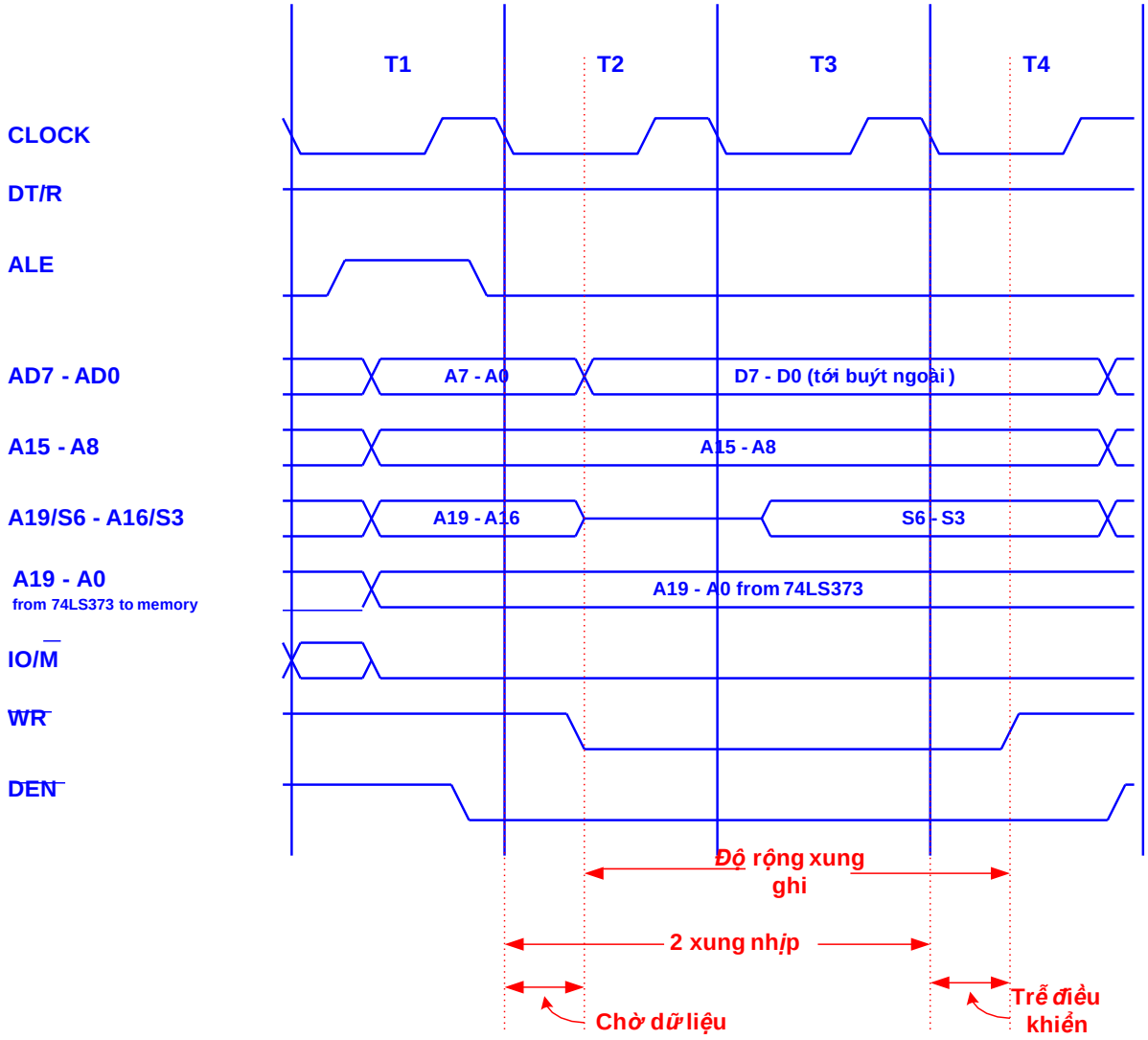
4.2.3 Định thời và chu trình đọc ghi bus

- T_1: khởi đầu chu trình. Sinh các tính hiệu điều khiển chốt, kiểu thao tác, hướng dữ liệu và địa chỉ
- 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

4.2.3 Chu trình đọc bus



4.2.3 Chu trình ghi bus

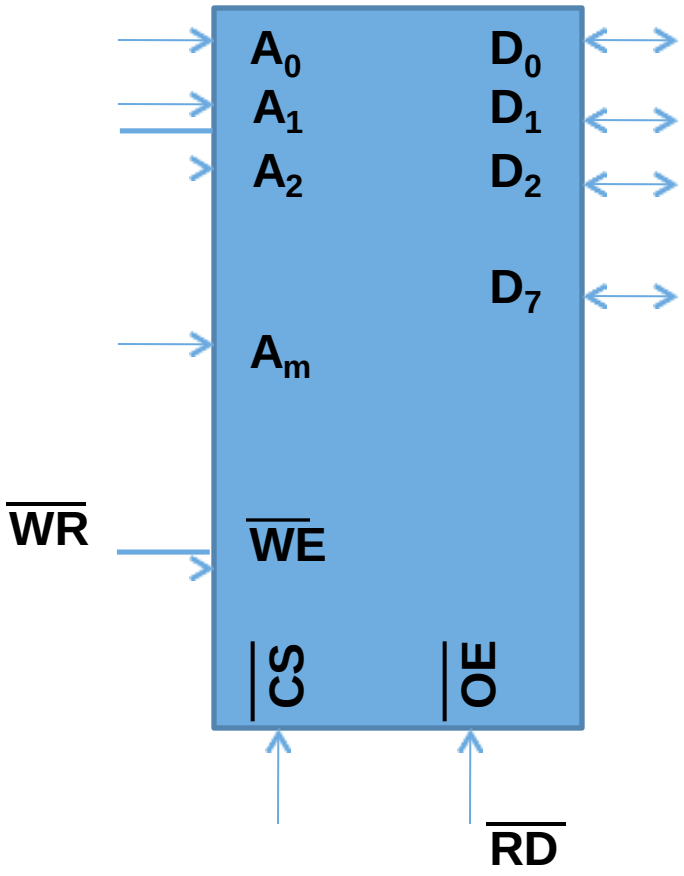


4.3 Phối ghép CPU với bộ nhớ

- Vai trò:
  - Chọn mạch nhớ cần đọc ghi
  - Chọn ô nhớ cần đọc ghi
- Đầu vào:
  - 20 bit địa chỉ vật lý
  - Các tín hiệu IO/M và RD (đọc) hoặc WR (ghi)
- Các loại mạch nhớ:
  - ROM/EPROM
  - SRAM
  - DRAM
- Mạch phối ghép: NAND, 74LS134, EPROM

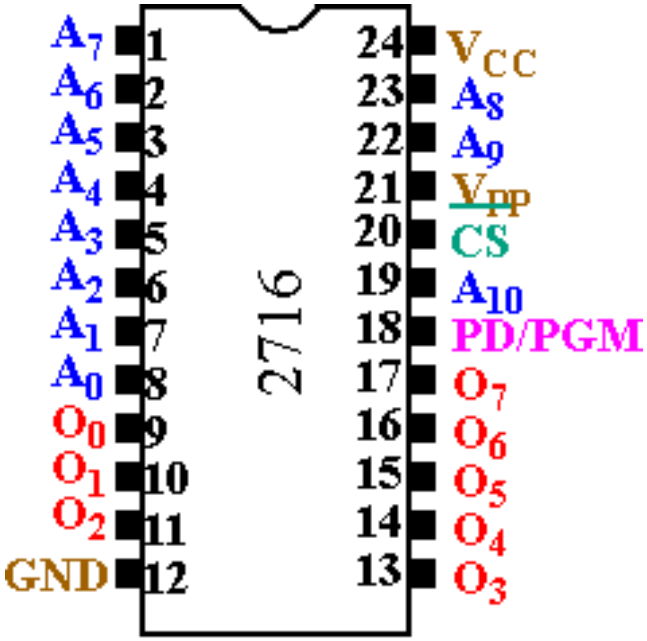
4.3.1 Cấu trúc mạch nhớ - tổng quát

- A_1-A_m: Địa chỉ
- D_0-D_7: Dữ liệu
- WE: Cho phép ghi
- OE: Cho phép ra
- CS: Kích hoạt

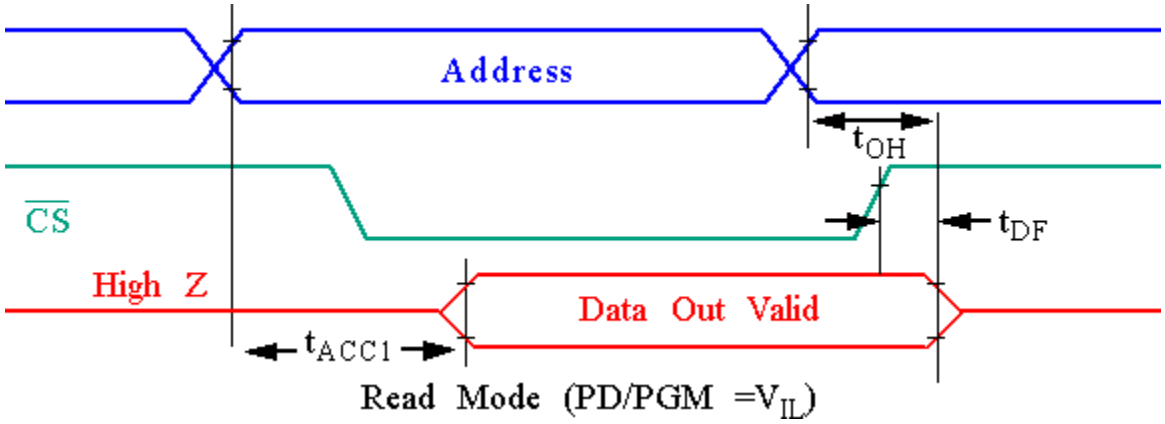


4.3.1 Cấu trúc mạch nhớ - EFROM Intel 2176(2Kx8)

- 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
  $V_{pp} = 25V$



2K x 8 EPROM

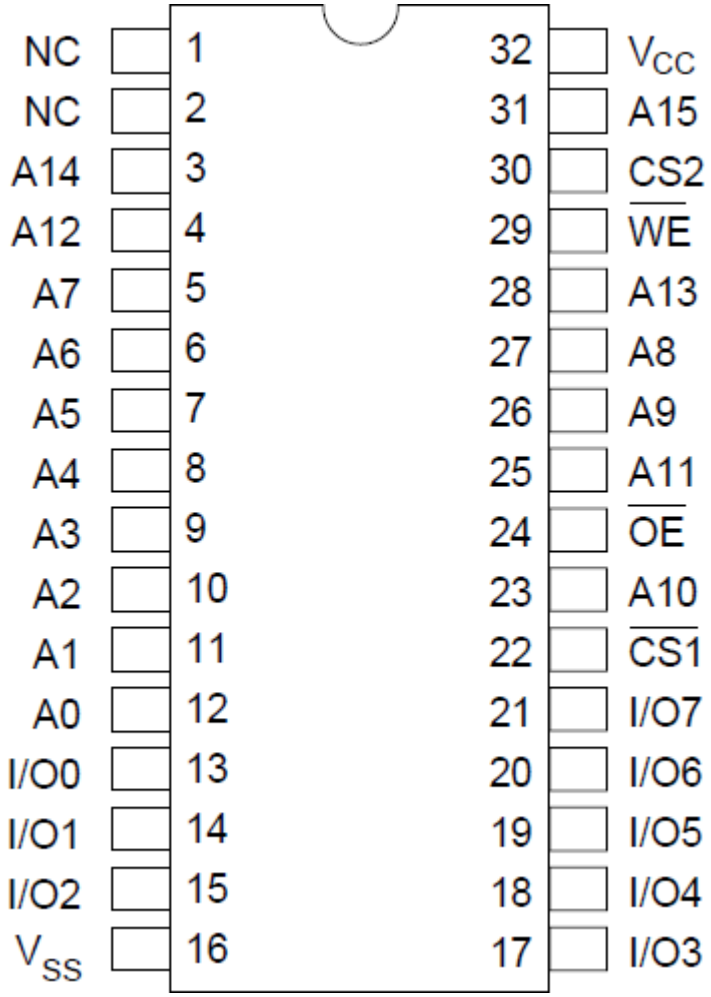


Read Mode (PD/PGM =V_{IL})

4.3.1 Cấu trúc mạch nhớ - SRAM

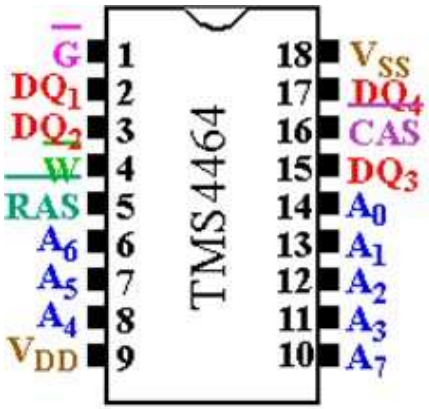
- Hitachi HM62864 - 64K×8
  - Tốc độ 50-85ns

Pin Name	Function
A0 to A15	Address
I/O0 to I/O7	Input/output
$\overline{CS1}$	Chip select 1
CS2	Chip select 2
$\overline{WE}$	Write enable
$\overline{OE}$	Output enable
NC	No connection
V_{CC}	Power supply
V_{SS}	Ground



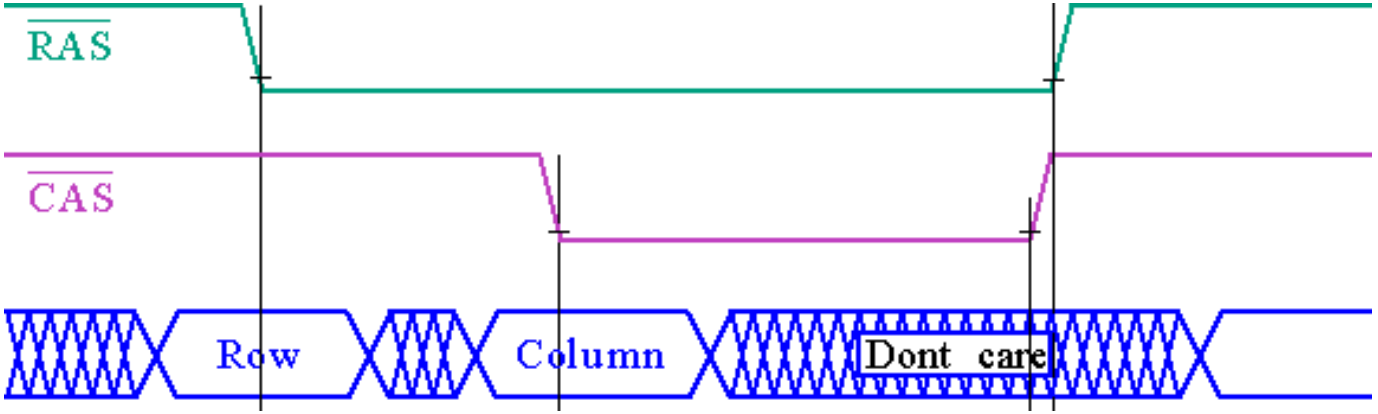
4.3.1 Cấu trúc mạch nhớ - DRAM

- TMS 4464
  - 64K×4
- 64K = {RA_0 - RA_7}+ {CA_0-CA_7}



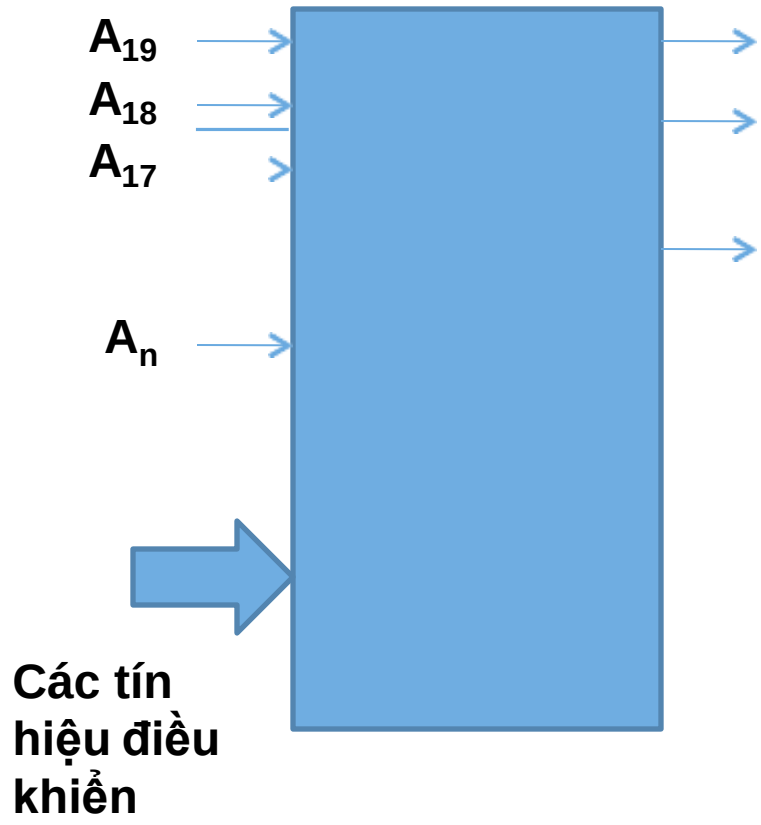
64K x 4 DRAM

Pin(s)	Function
A_0-A_7	Address
DQ_1-DQ_4	Data In/Data Out
RAS	Row Address Strobe
CAS	Column Address Strobe
G	Output Enable
W	Write Enable



4.3.2 Giải mã địa chỉ bộ nhớ

- Ánh xạ các tín hiệu địa chỉ thành tín hiệu chọn (kích hoạt) chíp nhớ
  - $A_{19}A_{18}..A_n \rightarrow CS_0,CS_1,\ldots,CS_n$
- Giải mã đầy đủ
  - Sử dụng $A_{19}A_{18}..A_n$
  - Tín hiệu đầu ra chọn duy nhất 1 mạch nhớ.
- Giải mã rút gọn
  - Sử dụng $A_{19}A_{18}..A_m;m>n$
  - Tín hiệu đầu ra có thể chọn nhiều hơn 1 mạch nhớ.



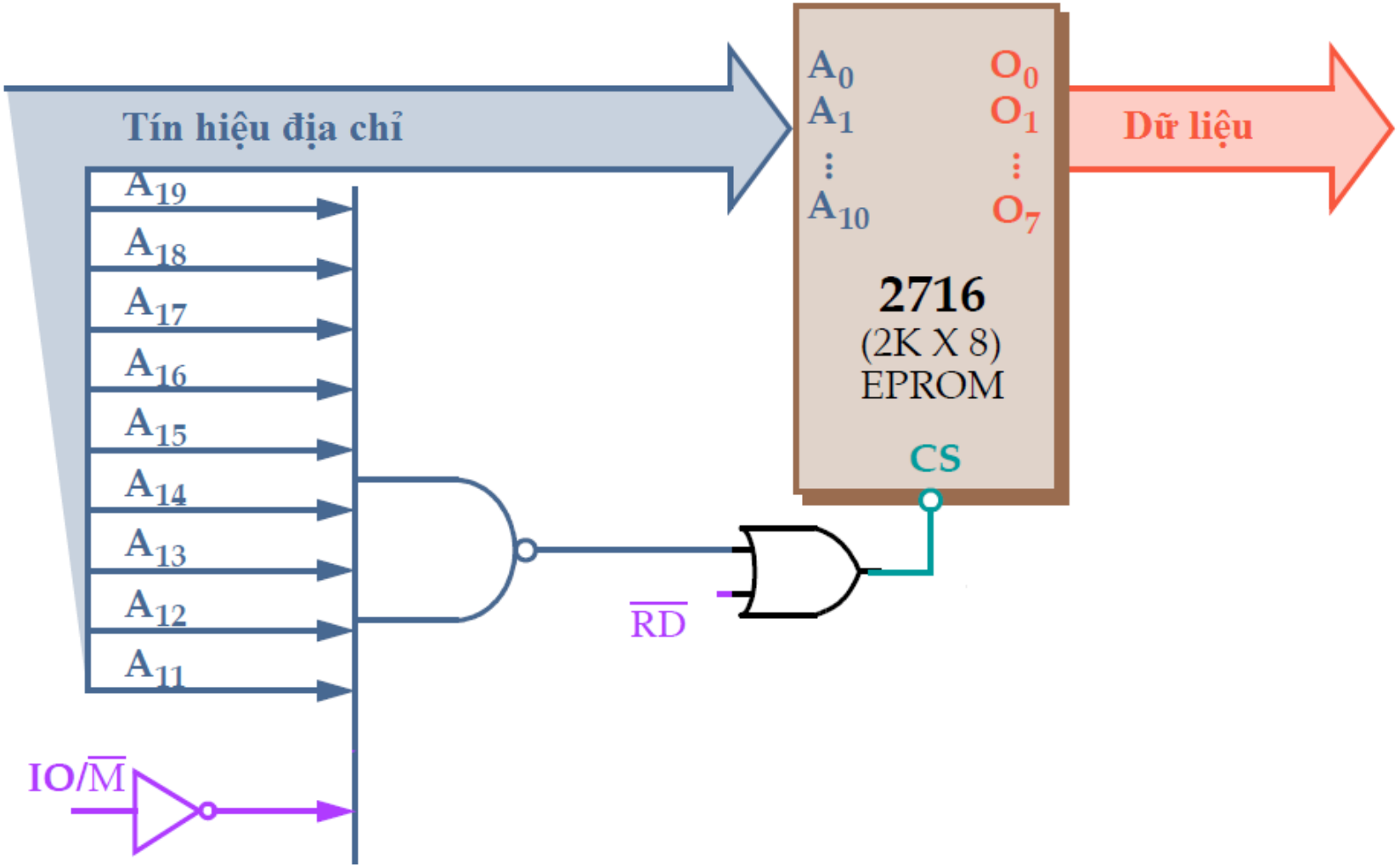
4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

- Chíp nhớ ROM 2K×8
- Khoảng địa chỉ cấp: FF800-FFFFF
- Tín hiệu địa chỉ dùng để địa chỉ hóa các ô nhớ trong chip ROM 2K: 11bit (A_0-A_{10}).
- Tín hiệu địa chỉ dùng để chọn chíp
  - $A_{19}…A_{16}A_{15}A_{12}A_{11}$
  - 1111 1111 1000 0000 0000 - 1111 1111 1111 1111 1111

NOT AND

- CS = RD OR NOT(($A_{19}…A_{16}A_{15}A_{12}A_{11}$)AND(IO/M))

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản



4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

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

Biết rằng kích thước 1 vi mạch nhớ là 2Kx8 và địa chỉ cơ sở là 03800H;

→ ***C_{ic} = 4KB; IC = 2Kx8; ĐCCS = 03800H***

- **Bước 1**: xác định số bit cho địa chỉ nội bộ chip và mạch giải mã
- **Bước 2**: Phân giải địa chỉ cơ sở của các chip
- **Bước 3**: Vẽ sơ đồ bit
- **Bước 4**: Vẽ hình mạch giải mã

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

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

$$C_{ic} = 4KB;\ IC = 2Kx8;\ ĐCCS = 03800H$$

Chíp nhớ IC 2Kx8 chiếm không gian 2KB = $2^1 x 2^{10} = 2^{11}$ B

→ Cần 11 bit địa chỉ nội bộ chip ($A_0 – A_{10}$)

Vi xử lý 8086 có 20 bit địa chỉ nên ta có 20 – 11 = 9

→ 9 bit cho mạch giải mã

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

- **Bước 2**: Phân giải địa chỉ cơ sở của các chip

$$C_{ic} = 4KB;\ IC = 2Kx8;\ ĐCCS = 03800H$$

Bộ nhớ có dung lượng 4KB và mỗi chip nhớ có dung lượng 2KB

→ Cần $\frac{4KB}{2KB} = 2$ chip nhớ

Mà 2KB = 2^{11} = 0000 0000 1000 0000 0000(B) = 00800(H)

→ Dung lượng của một chip nhớ là 00800(H)

Địa chỉ cuối = Địa chỉ đầu + Dung lượng - 1

- Địa chỉ của IC 1: Từ ***03800H*** đến (***03800H + 00800H – 1H***)

  : Từ ***03800H*** đến (***04000H – 1H***)

  : Từ ***03800H*** đến ***03FFFH***

- Địa chỉ của IC 2: Từ ***04000H*** đến ***047FFH***

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

- **Bước 3**: Vẽ sơ đồ bit

$$C_{ic} = 4KB;\ IC = 2Kx8;\ ĐCCS = 03800H$$

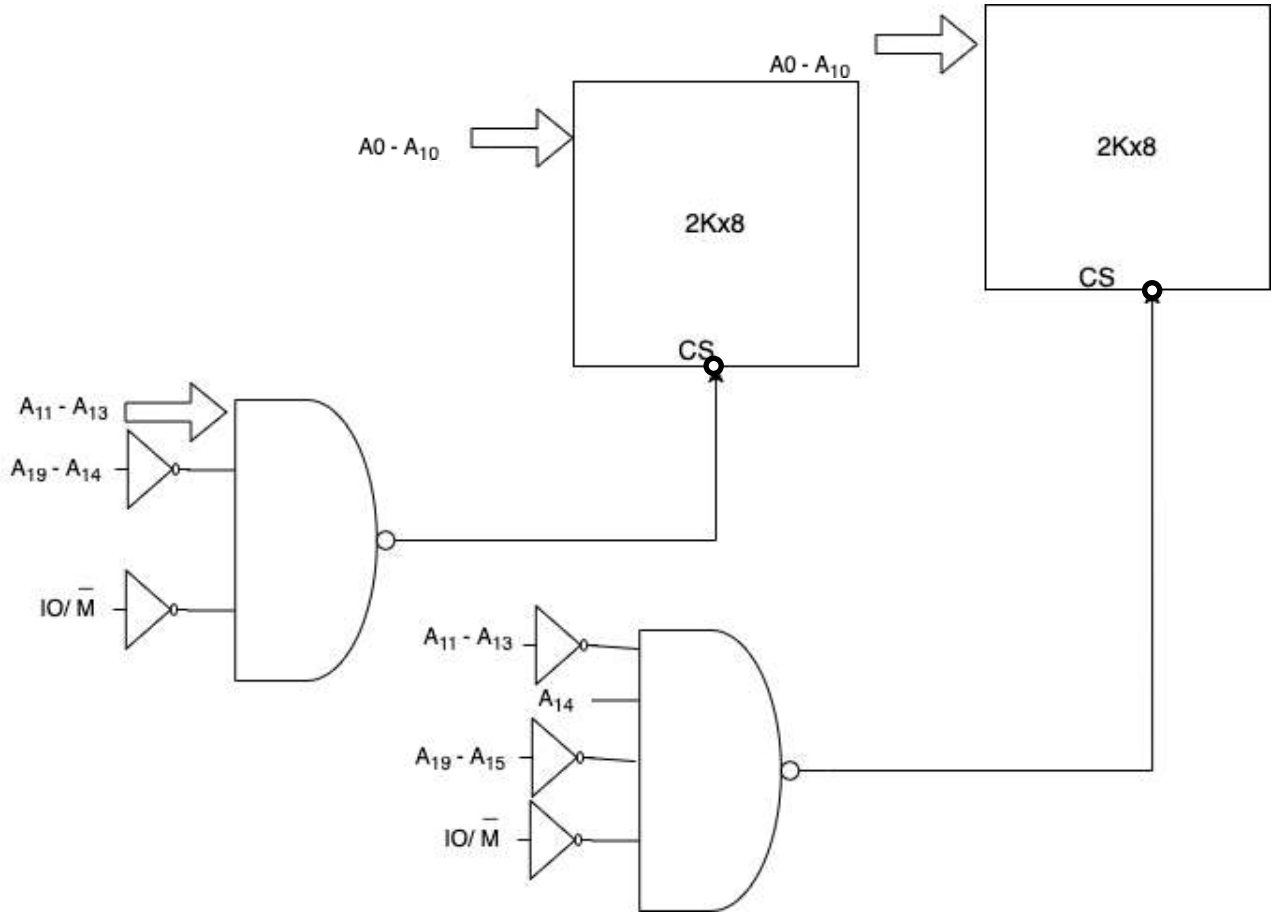
  - Địa chỉ của IC 1: Từ ***03800H*** đến ***03FFFH***
  - Địa chỉ của IC 2: Từ ***04000H*** đến ***047FFH***

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

- **Bước 4**: Vẽ mạch giải mã

		9 bit cao									11 bit địa chỉ nội bộ chip ($A_0 - A_{10}$)										
IC1	***03800H***	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**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**
		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**

- Xét 9 bit cao ($A_{14} - A_{19}$) :
  - Một cho 03800H
  - Một cho 04000H

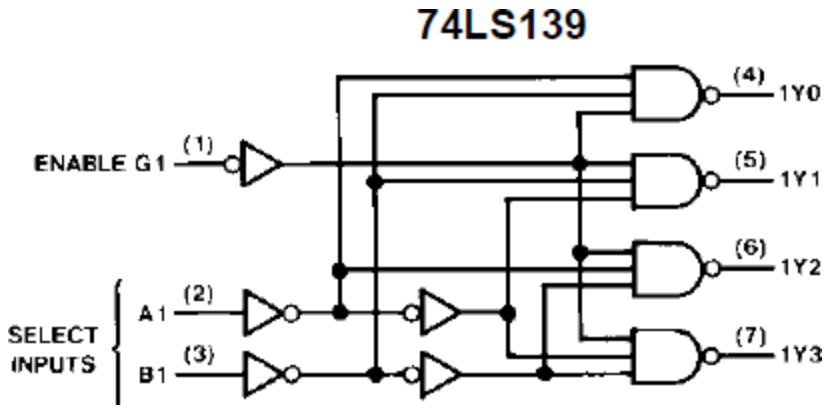


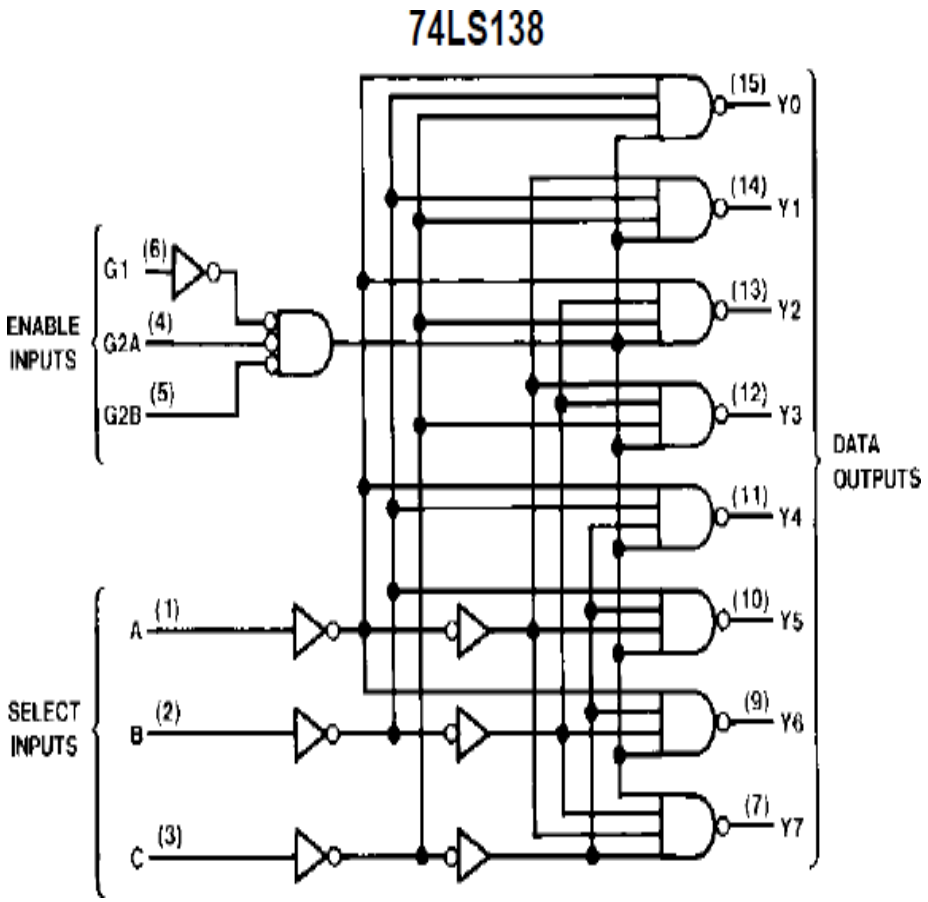
4.3.2 Giải mã đ.c b.nhớ sử dụng mạch lôgic cơ bản

- 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.

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- 74-138 mạch giải mã 3→8
- 74-139 mạch giải mã 2→4





4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

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 2Kx8. Chỉ được sử dụng các chip giải mã địa chỉ ***74LS139*** *(Là các chip giải mã có 2 đầu vào và 4 đầu ra).*

→ ***C_{ic} = 8KB; IC = 2Kx8; ĐCCS = 0F800H***

- **Bước 1**: xác định số bit cho địa chỉ nội bộ chip và mạch giải mã
- **Bước 2**: Phân giải địa chỉ cơ sở của các chip
- **Bước 3**: Vẽ sơ đồ bit
- **Bước 4**: Vẽ hình mạch giải mã

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

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

$$C_{ic} = 8KB;\ IC = 2Kx8;\ ĐCCS = 0F800H$$

Chíp nhớ IC 2Kx8 chiếm không gian $2KB = 2^1 x2^{10} = 2^{11}$ B

→ Cần 11 bit địa chỉ nội bộ chip ($A_0 – A_{10}$)

Vi xử lý 8086 có 20 bit địa chỉ nên ta có 20 – 11 = 9

→ 9 bit cho mạch giải mã

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- **Bước 2**: Phân giải địa chỉ cơ sở của các chip

$$C_{ic} = 8KB;\ IC = 2Kx8;\ ĐCCS = 0F800H$$

Bộ nhớ có dung lượng 8KB và mỗi chip nhớ có dung lượng 2KB

→ Cần $\frac{8KB}{2KB} = 4$ chip nhớ

Mà 2KB = 2^{11} = 0000 0000 1000 0000 0000(B) = 00800(H)

→ Dung lượng của một chip nhớ là 00800(H)

Địa chỉ cuối = Địa chỉ đầu + Dung lượng - 1

- Địa chỉ của IC 1: Từ ***0F800H*** đến ***0FFFFH***
- Địa chỉ của IC 2: Từ ***10000H*** đến ***107FFH***
- Địa chỉ của IC 3: Từ ***10800H*** đến ***10FFFH***
- Địa chỉ của IC 4: Từ ***11000H*** đến ***117FFH***

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- **Bước 3**: Vẽ sơ đồ bit

$$C_{ic} = 8KB;\ IC = 2Kx8;\ ĐCCS = 0F800H$$

  - Địa chỉ của IC 1: Từ ***0F800H*** đến ***0FFFFH***
  - Địa chỉ của IC 2: Từ ***10000H*** đến ***107FFH***
  - Địa chỉ của IC 3: Từ ***10800H*** đến ***10FFFH***
  - Địa chỉ của IC 4: Từ ***11000H*** đến ***117FFH***

74LS139 *(Là các chip giải mã có 2 đầu vào và 4 đầu ra).*

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

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- **Bước 4**: Vẽ mạch giải mã

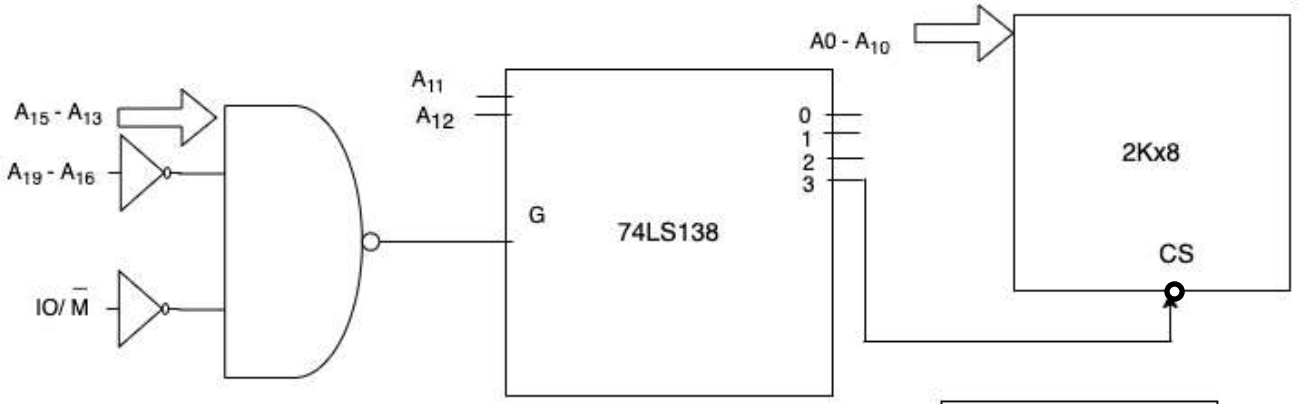
		7 bit cao							2 bit vào		11 bit địa chỉ nội bộ chip ($A_0 - A_{10}$)										
IC1	*0F800H*	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
IC2	*10000H*	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IC3	*10800H*	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
IC4	*11000H*	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
		A_{19}	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

- Xét 7 bit cao ($A_{13} - A_{19}$) cần 2 chip giải mã 74LS139:
  - Một cho 0F800H
  - Một cho 10000H, 10800H, 11000H (vì 7 bit giống nhau)

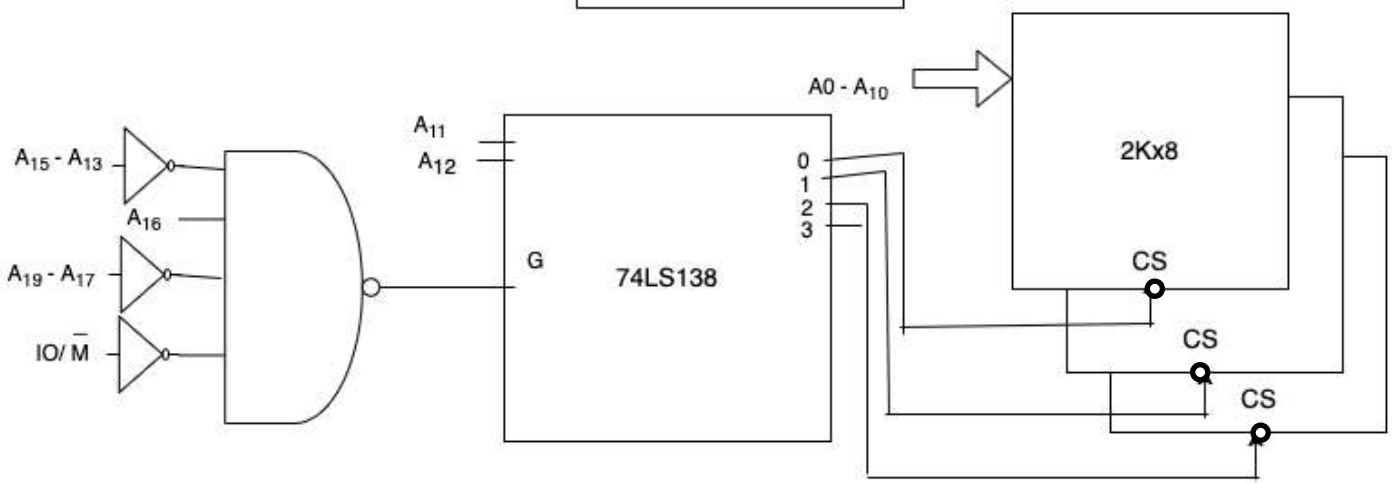
- **Bước 4**: Vẽ mạch giải mã

		7 bit cao							2 bit vào		11 bit địa chỉ nội bộ chip (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
IC2	*10000H*	0	0	0	1	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
IC4	*11000H*	0	0	0	1	0	0	0	1	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_1 0	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0

- Một cho 0F800H



- Một cho 10000H, 10800H, 11000H



4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

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 2Kx8. Chỉ được sử dụng các chip giải mã địa chỉ ***74LS138*** *(Là các chip giải mã có 3 đầu vào và 8 đầu ra).*

→ C_{ic} = 8KB; IC = 2Kx8; ĐCCS = 0F800H

- **Bước 1**: xác định số bit cho địa chỉ nội bộ chip và mạch giải mã
- **Bước 2**: Phân giải địa chỉ cơ sở của các chip
- **Bước 3**: Vẽ sơ đồ bit
- **Bước 4**: Vẽ hình mạch giải mã

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

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

$$C_{ic} = 8KB;\ IC = 2Kx8;\ ĐCCS = 0F800H$$

Chíp nhớ IC 2Kx8 chiếm không gian $2KB = 2^1 x2^{10} = 2^{11}$ B

→ Cần 11 bit địa chỉ nội bộ chip ($A_0 – A_{10}$)

Vi xử lý 8086 có 20 bit địa chỉ nên ta có 20 – 11 = 9

→ 9 bit cho mạch giải mã

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- **Bước 2**: Phân giải địa chỉ cơ sở của các chip

$$C_{ic} = 8KB;\ IC = 2Kx8;\ ĐCCS = 0F800H$$

Bộ nhớ có dung lượng 8KB và mỗi chip nhớ có dung lượng 2KB

→ Cần $\frac{8KB}{2KB} = 4$ chip nhớ

Mà 2KB = 2^{11} = 0000 0000 1000 0000 0000(B) = 00800(H)

→ Dung lượng của một chip nhớ là 00800(H)

Địa chỉ cuối = Địa chỉ đầu + Dung lượng - 1

- Địa chỉ của IC 1: Từ ***0F800H*** đến ***0FFFFH***
- Địa chỉ của IC 2: Từ ***10000H*** đến ***107FFH***
- Địa chỉ của IC 3: Từ ***10800H*** đến ***10FFFH***
- Địa chỉ của IC 4: Từ ***11000H*** đến ***117FFH***

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- **Bước 3**: Vẽ sơ đồ bit

$$C_{ic} = 8KB;\ IC = 2Kx8;\ ĐCCS = 0F800H$$

  - Địa chỉ của IC 1: Từ ***0F800H*** đến ***0FFFFH***
  - Địa chỉ của IC 2: Từ ***10000H*** đến ***107FFH***
  - Địa chỉ của IC 3: Từ ***10800H*** đến ***10FFFH***
  - Địa chỉ của IC 4: Từ ***11000H*** đến ***117FFH***

74LS138 *(Là các chip giải mã có 3 đầu vào và 8 đầu ra).*

		6 bit cao						3 bit vào			11 bit địa chỉ nội bộ chip ($A_0 - A_{10}$)										
IC1	***0F800H***	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
IC2	***10000H***	0	0	0	1	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
IC4	***11000H***	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- **Bước 4**: Vẽ mạch giải mã

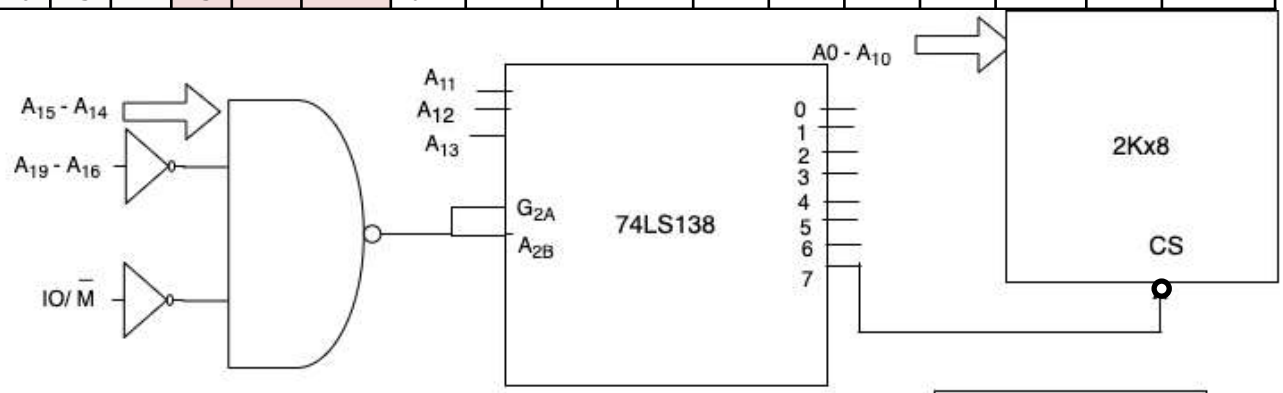
		6 bit cao						3 bit vào			11 bit địa chỉ nội bộ chip ($A_0 - A_{10}$)										
IC1	*0F800H*	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
IC2	*10000H*	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IC3	*10800H*	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
IC4	*11000H*	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
		A_{19}	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

- Xét 6 bit cao ($A_{14} - A_{19}$) cần 2 chip giải mã 74LS138:
  - Một cho 0F800H
  - Một cho 10000H, 10800H, 11000H (vì 6 bit giống nhau)

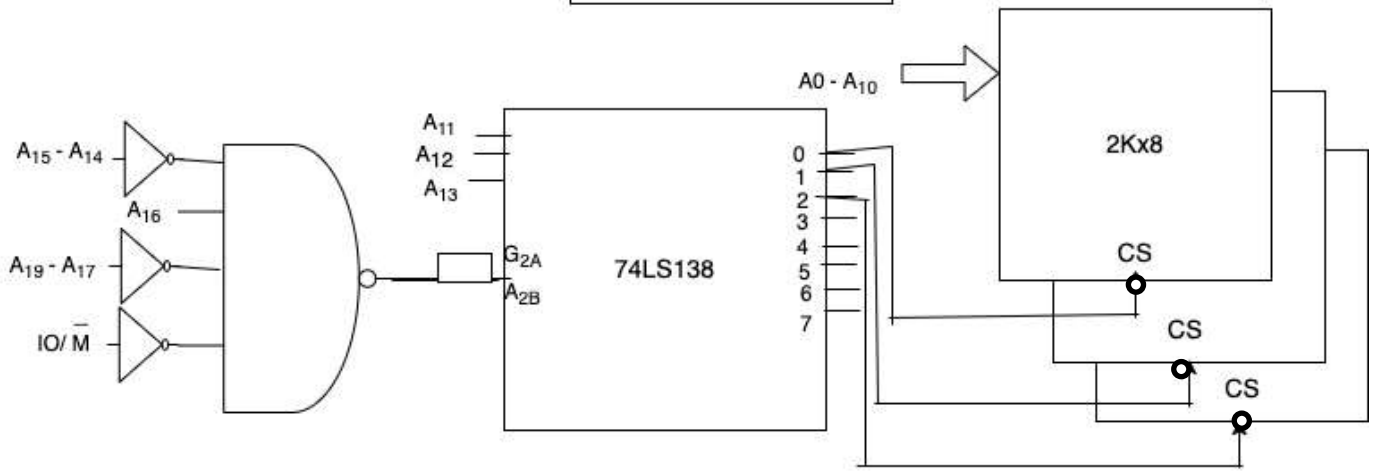
- **Bước 4**: Vẽ mạch giải mã

		6 bit cao						3 bit vào			11 bit địa chỉ nội bộ chip ($A_0 - A_{10}$)										
IC1	*0F800H*	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
IC2	*10000H*	0	0	0	1	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
IC4	*11000H*	0	0	0	1	0	0	0	1	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_1 0	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0

- Một cho 0F800H

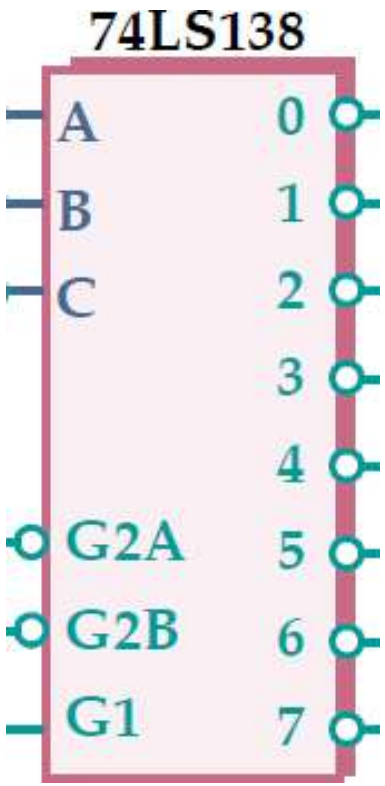


- Một cho 10000H, 10800H, 11000H



4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- Bảng dữ liệu mạch giải mã 74LS138



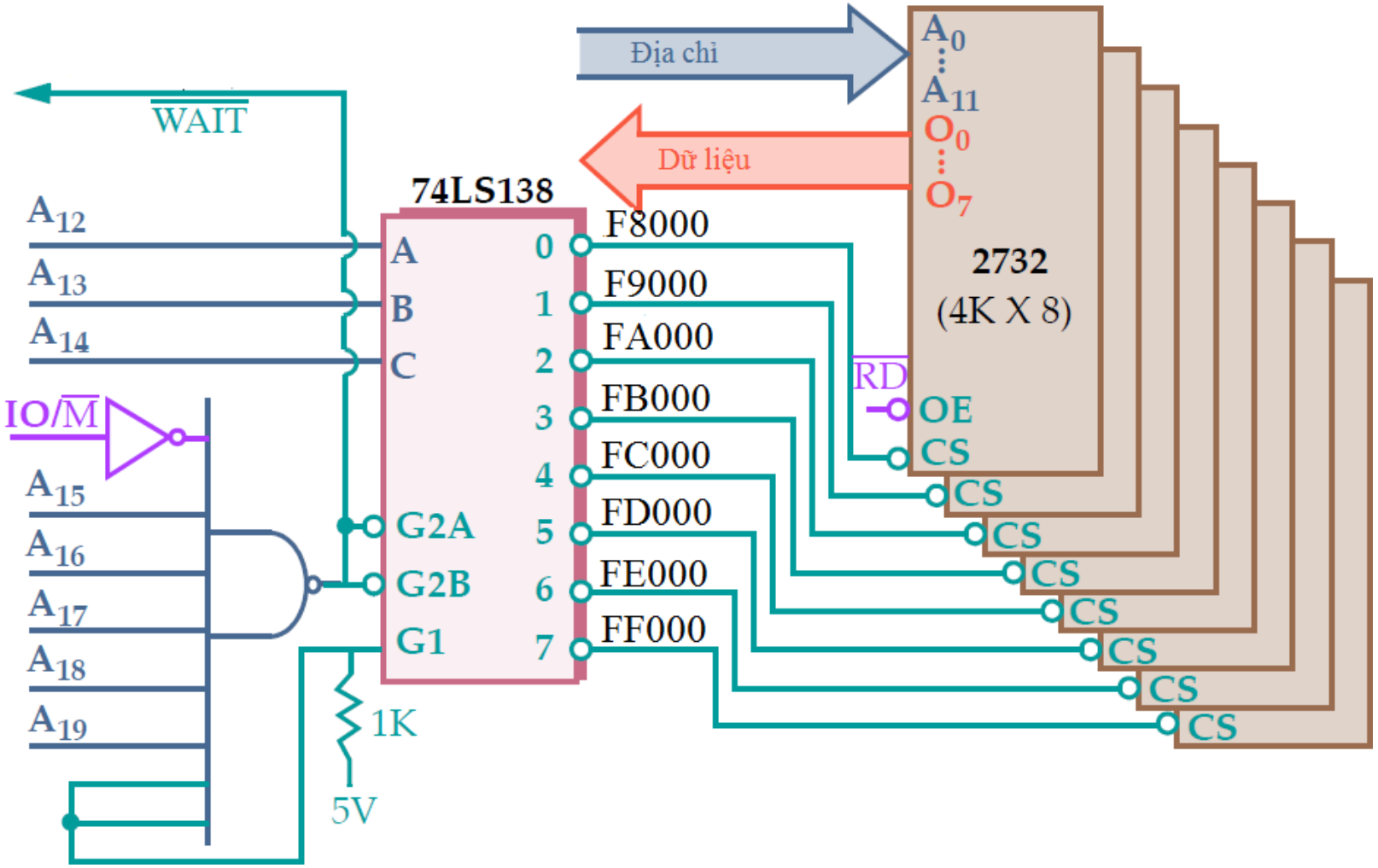
Inputs						Output							
Enable			Select										
G2A	G2B	G1	C	B	A	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$	$\overline{7}$
1	X	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	X	1	1	1	1	1	1	1	1
X	X	0	X	X	X	1	1	1	1	1	1	1	1
0	0	1	0	0	0	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	0	1	1	1	1	1
0	0	1	0	1	1	1	1	1	0	1	1	1	1
0	0	1	1	0	0	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	0	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	0

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- Chíp nhớ EPROM 4K×8
- Khoảng địa chỉ cấp: F8000-FFFFF (32KB)
- Tín hiệu địa chỉ dùng để địa chỉ hóa các ô nhớ trong chip EPROM 4K: 12bit (A_0-A_{11}).
- Tín hiệu địa chỉ dùng để chọn chíp: A_{19}…$A_{16}A_{15}$…..A_{12}
  - Các tín hiệu địa chỉ $A_{12}A_{13}A_{14}$ thay đổi, còn các tín hiệu A_{15}-A_{19} không thay đổi và luôn bằng 1.

➔ $A_{12}A_{13}A_{14}$ đưa vào các đầu vào A, B, C của mạch giải mã, còn các tín hiệu địa chỉ còn lại A_{15}-A_{19} 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.

4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp



4.3.2 Giải mã đ.c b.nhớ sử dụng mạch tích hợp

- 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.

4.3.2 Giải mã đ.c b.nhớ sử dụng 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.

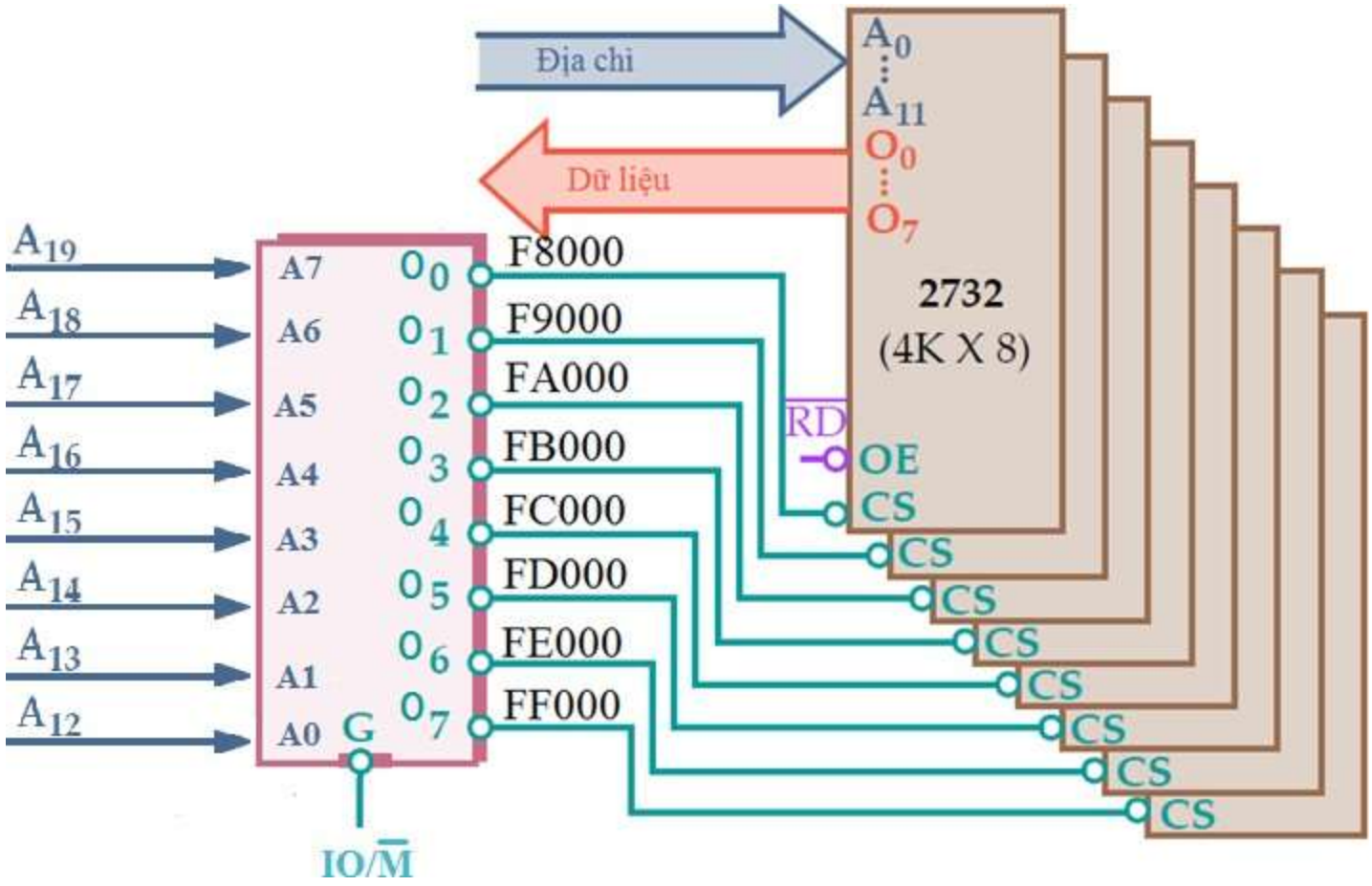
4.3.2 Giải mã đ.c b.nhớ sử dụng PROM

❖ 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	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	1	0	0	1	1	1	1	0	1	1	1
1	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

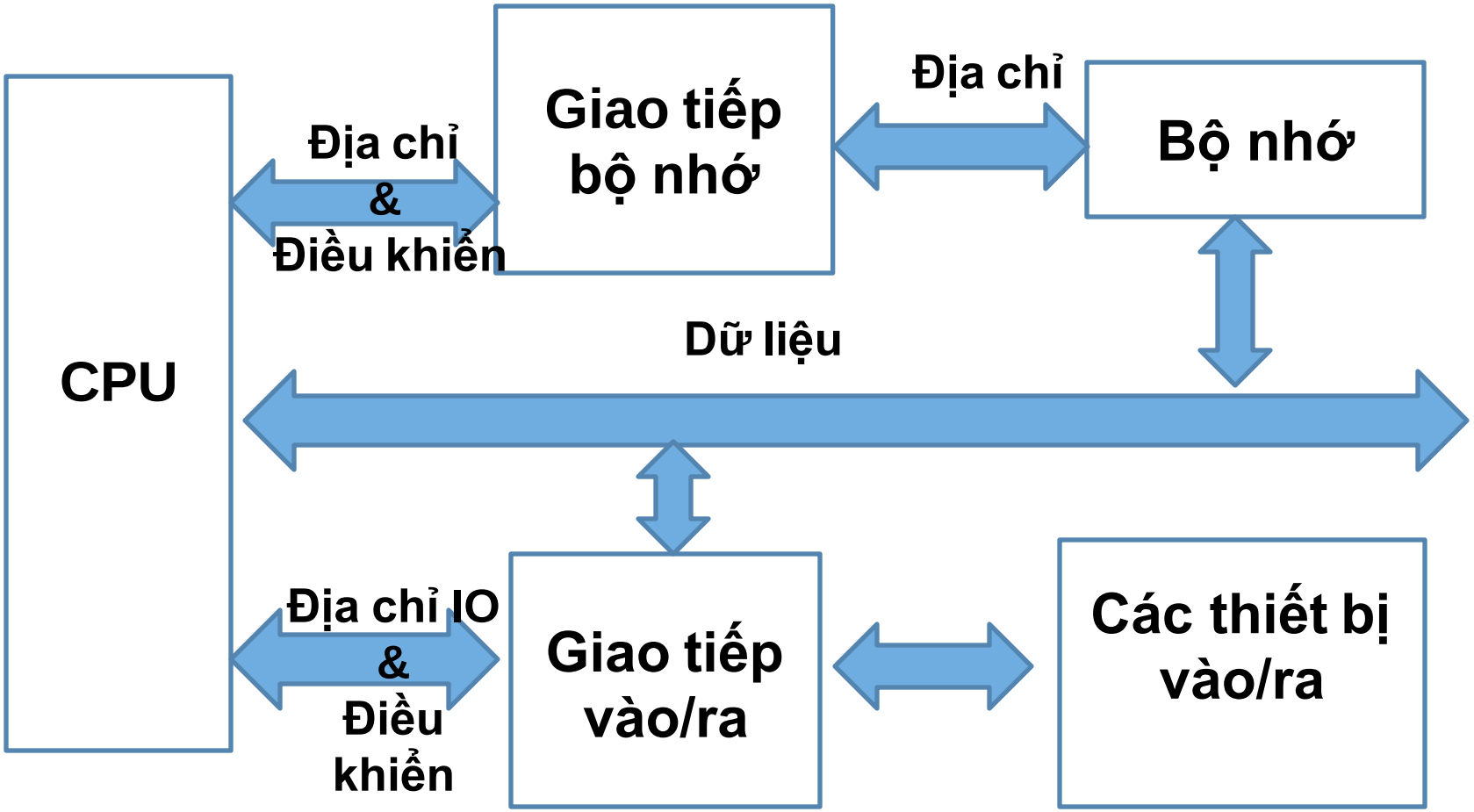
4.3.2 Giải mã đ.c b.nhớ sử dụng PROM



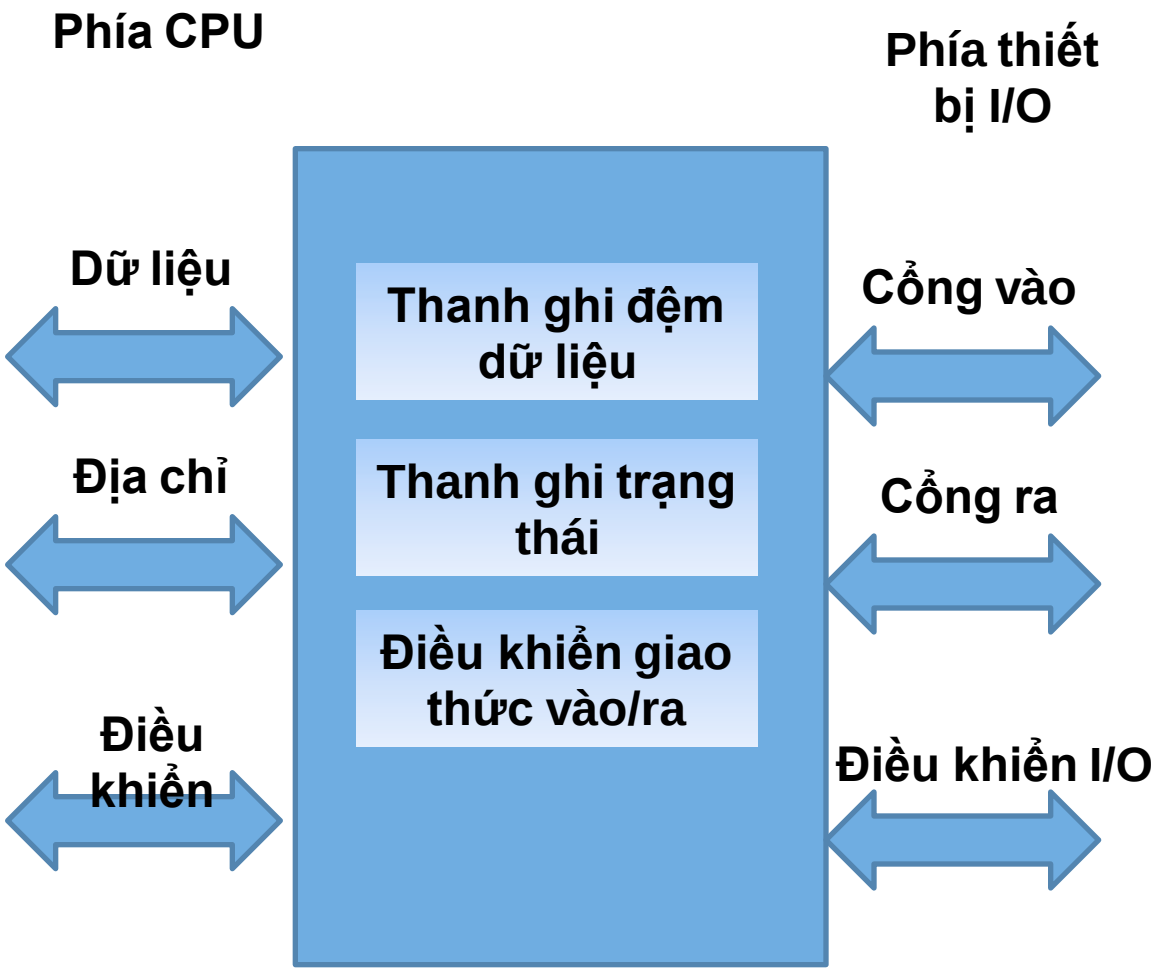
4.3.2 Giải mã đ.c b.nhớ sử dụng PROM

- Ưu điểm
  - Cho phép tạo mạch giải mã đầy đủ mà không cần phải sử dụng mạch phụ trợ → 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.
  - 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.
- Nhược điểm:

4.4. Phối ghép CPU với thiết bị vào ra

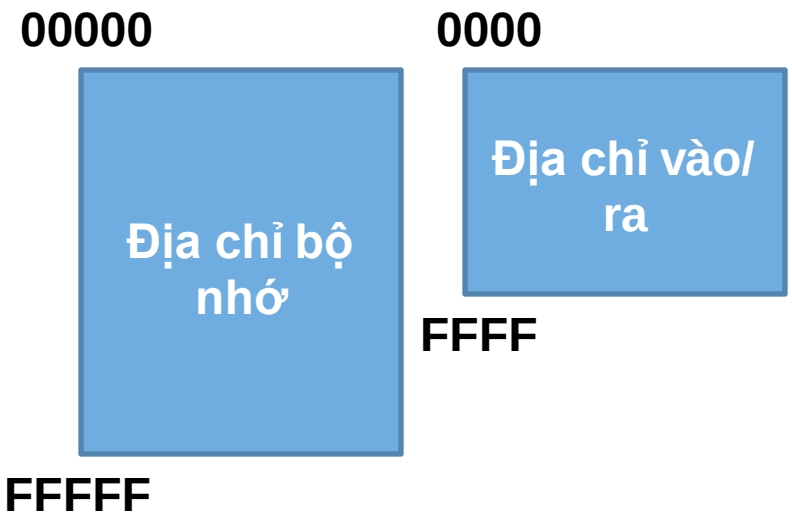


4.4. Phối ghép CPU với thiết bị vào ra



4.4.1 Phân loại thiết bị vào ra theo không gian địa chỉ

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

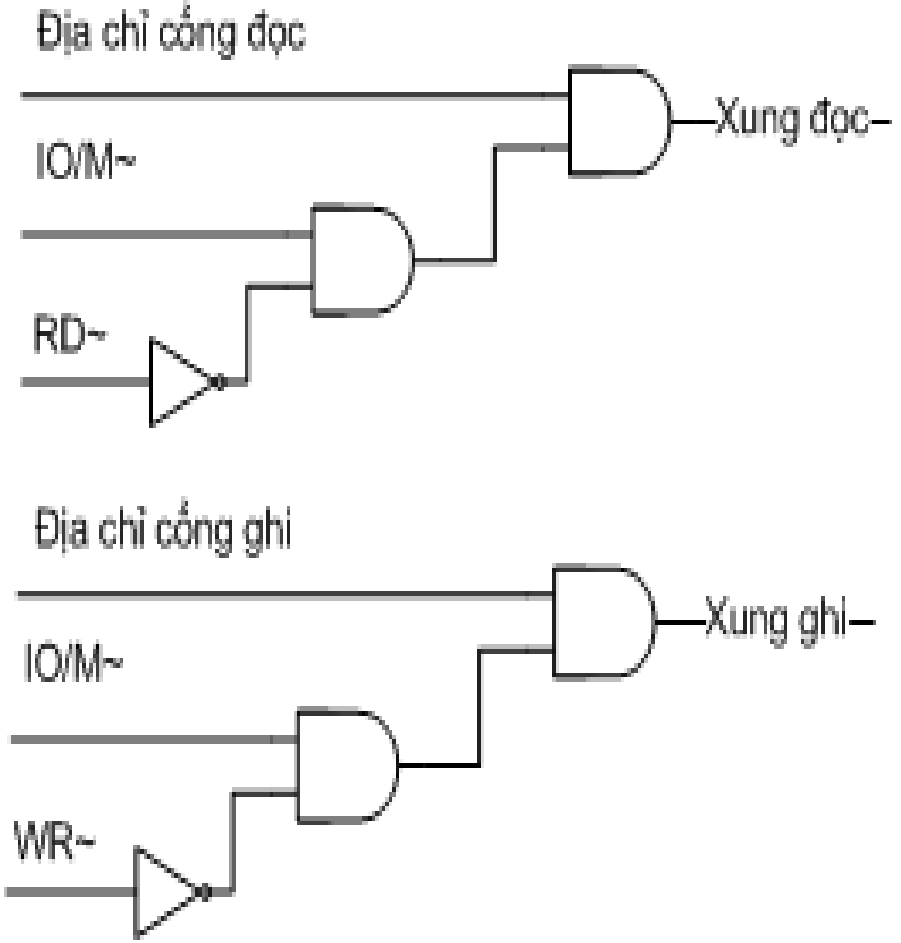


4.4.1 Phân loại thiết bị vào ra theo không gian địa chỉ

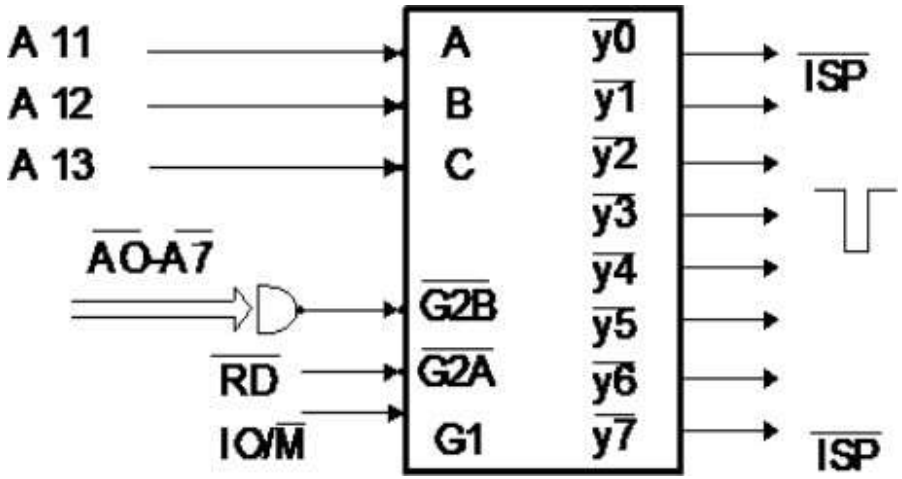
- Thao tác đọc/ghi dữ liệu với không gian địa chỉ tách biệt:
  - IN AX, [Địa chỉ cổng]
  - OUT [Địa chỉ cổng], AX
  - Địa chỉ cổng vào/ra
    - 0000-FFFF: Lưu trong DX
    - 00-FF: địa chỉ trực tiếp
- Thao tác đọc/ghi dữ liệu với không gian địa chỉ dùng chung:
  - MOV [Địa chỉ cổng], AX
  - Đọc: MOV AX, [Địa chỉ cổng]
  - Địa chỉ cổng vào/ra
    - 00000-FFFFF

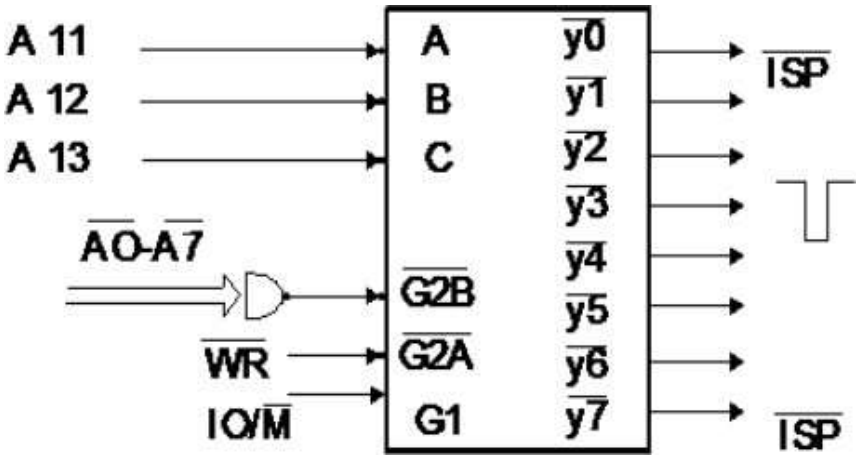
4.4.2 Giải mã đ.chỉ t.b vào ra sử dụng cổng logic

- Tổ hợp các tín hiệu địa chỉ và điều khiển thành xung đọc/ghi
  - Địa chỉ riêng
    - IO + RD~ + A_i…A_j = IN
    - IO + WR~ + A_i…A_j = OUT
  - Địa chỉ chung với bộ nhớ
    - M~ + RD~ + A_i…A_j = IN
    - M~ + WR~ + A_i…A_j = OUT



4.4.2 Giải mã đ.chỉ tb vào ra sử dụng mạch tích hợp



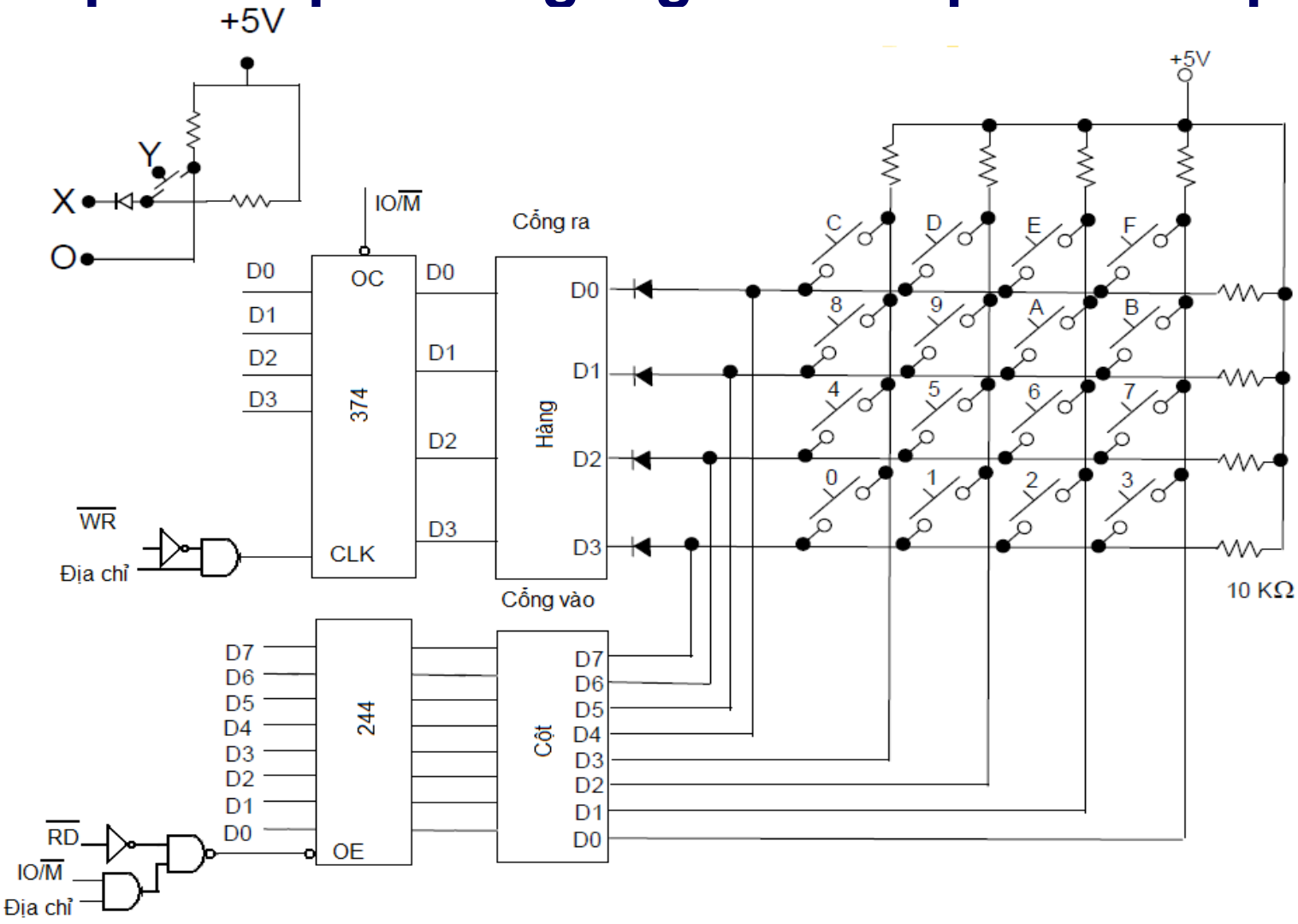


Giải mã địa chỉ cổng dùng 74LS138

4.4.3 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 bít có đầu ra 3 trạng thái (74LS373,74LS374)
  - Các mạch khuếch đại đệm 2 chiều 8 bít đầu ra 3 trạng thái (74LS245)

4.4.3 Một số mạch cổng đ.giản–Ghép nối bàn phím



4.4.3 Một số mạch cổng đ.giản–Ghép nối bàn phím

- Cổng ghép nối bàn phím 16 số dạng tiếp điểm sử dụng các mạch tích hợp:
  - Vi mạch 74LS374 được dùng để điều khiển các tín hiệu hàng và mạch 74LS244 dùng để điều khiển các tín hiệu cột;
- Nguyên tắc hoạt động:
  - Nếu tín hiệu X ở mức cao (lô-gíc 1) thì đi-ốt sẽ khóa lại, vậy nên tiếp điểm Y có đóng xuống hay không thì tại đầu O ta luôn thu được điện áp 5V (không có dòng điện).
  - Nếu tín hiệu X ở mức thấp (lô-gíc 0), thì đi-ốt mở và khi tiếp điểm Y đóng xuống tại đầu O ta thu được điện áp 0V.
- Xác định phím được ấn: quét tuần các dòng và đọc các cột. Nếu một phím được ấn thì bit tương ứng ở cổng ra Di = 0.

4.4.3 Chương trình kiểm tra một phím

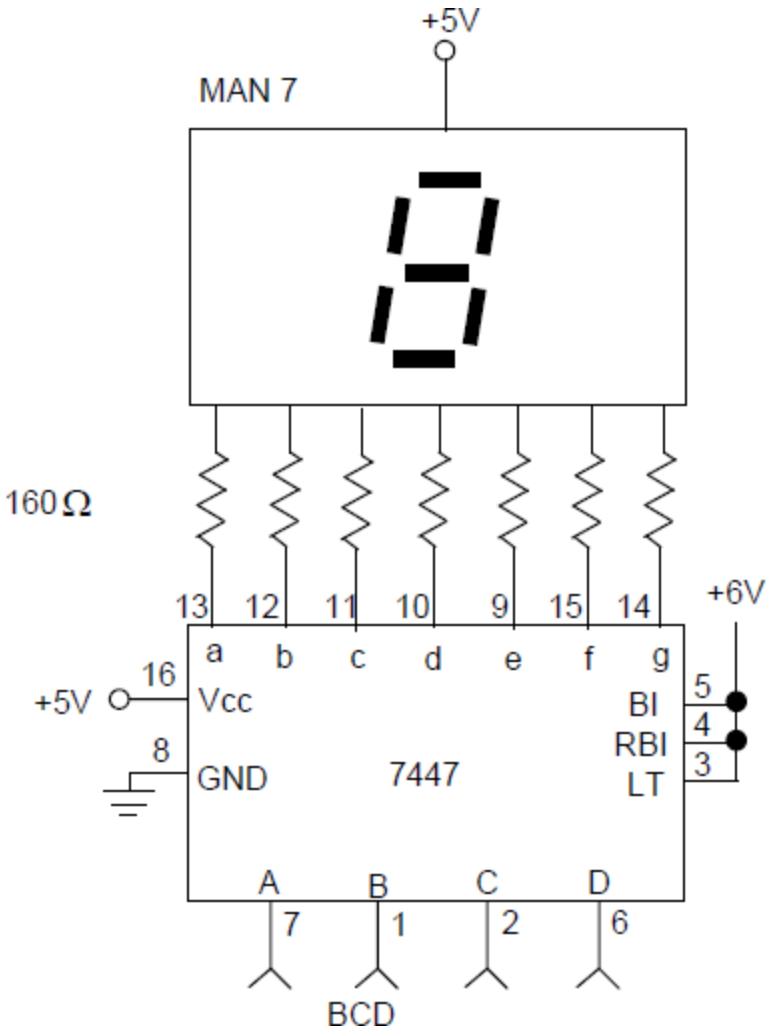
- Đoạn chương trình kiểm tra xem phím C (hàng D0, cột D3) có được bấm hay không. Biết địa chỉ cổng hàng là 0Ah và địa chỉ cổng cột là 0Bh.

```
Hang    EQU 0AH                 ; Địa chỉ cổng hàng
Cot     EQU 0BH                 ; Địa chỉ cổng cột
        MOV AL,11111110b        ; Chỉ có D0=0 – hàng thứ nhất
        OUT Hang, AL
Ktra:   IN AL, Cot              ; Đọc tín hiệu cột
        AND AL,00001000b        ; Giữ lại bít D3 ứng với phím C
        JNZ Ktra                ; Không bấm
        ...                     ; Phím C được bấm
```

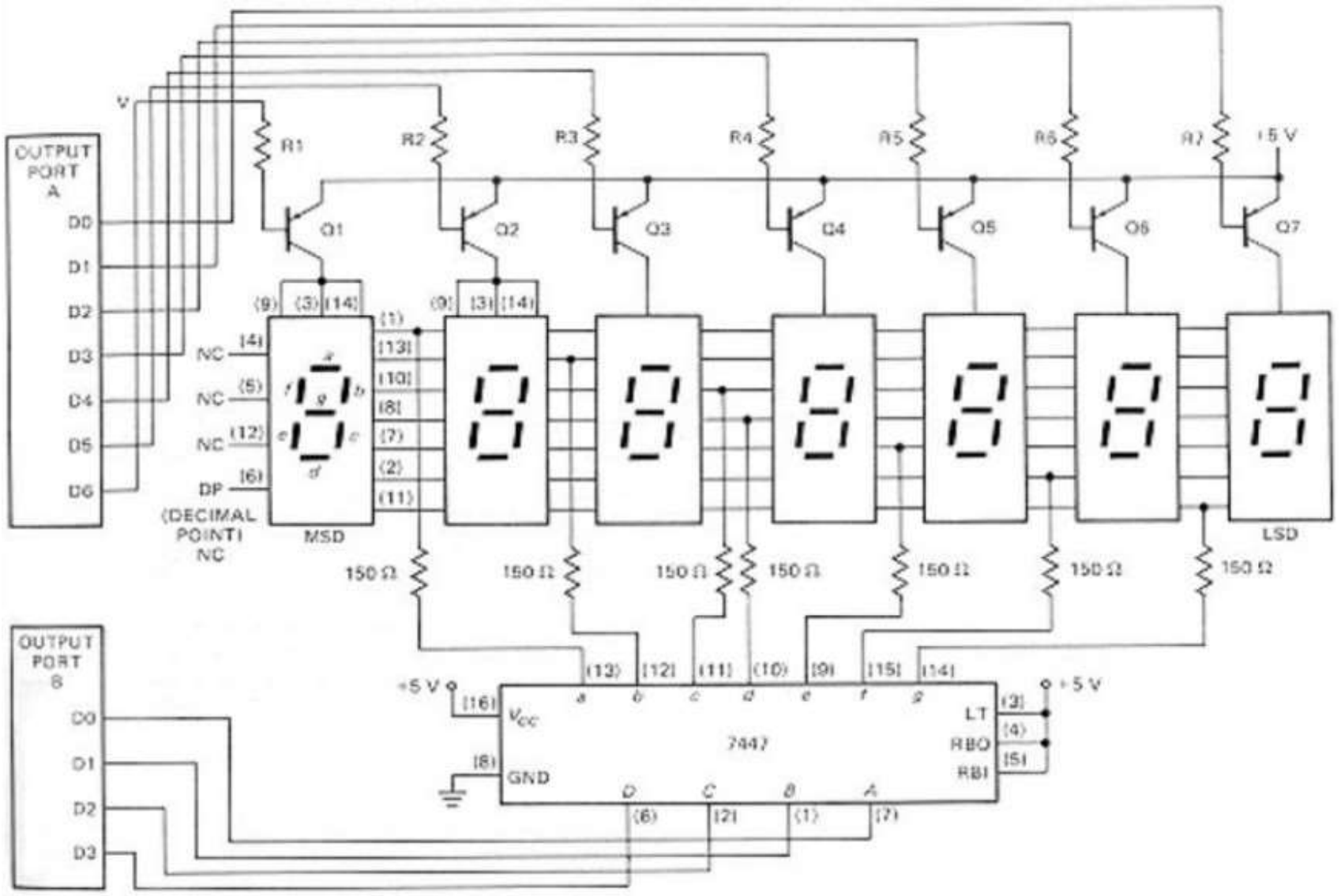
4.4.3 Một số mạch cổng–Ghép nối hiển thị số

- Ghép nối hiển thị số sử dụng mạch tích hợp 7447 và LED bảy đoạn:
  - Cổng A nhận thông tin điều khiển (bật/tắt) các thanh led thông qua 7 transitor 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.

4.4.3 Một số mạch cổng–Ghép nối hiển thị số



4.4.3 Một số mạch cổng–Ghép nối hiển thị số

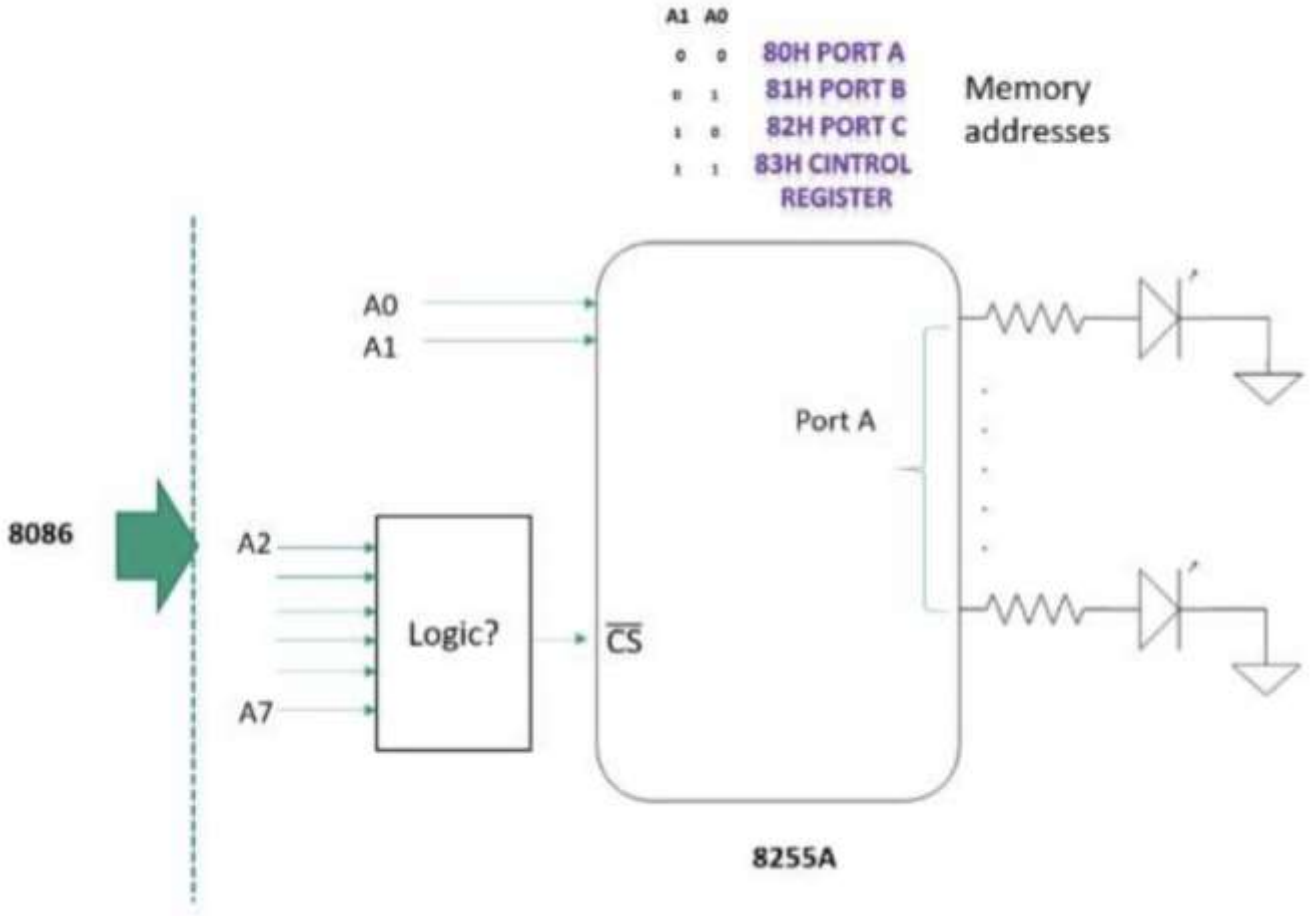


4.4.3 Chương trình hiển thị số trên LED

- Đoạn chương trình kiểm tra hệ thống LED bằng cách hiển thị trên cả 7 LED số 8. Biết địa chỉ cổng điều khiển LED là 0Ah và địa chỉ cổng dữ liệu hiển thị là 0Bh.

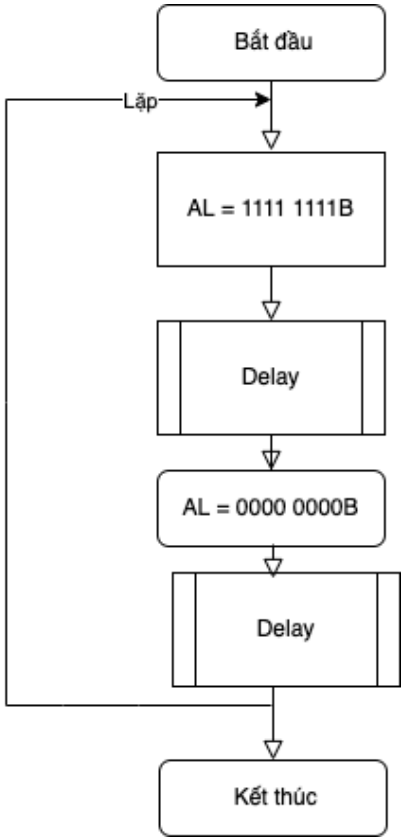
```
DK_LEDEQU 0AH          ; Cổng điều khiển LED
DL_LED EQU 0BH         ; Cổng dữ liệu hiển thị
        MOV AL,FFH     ; Tắt tất cả các LED
        OUT DK_LED, AL
        MOV CX,64      ; Trễ bằng 64 lệnh NOP
Tre:    NOP
        LOOP Tre
        MOV AL,8       ; Đưa số 8 ra 7447
        OUT DL_LED,AL
        XOR AL,AL      ; Đặt AL=0
        OUT DK_LED,AL ; Bật tất cả các LED
```

Một số mạch cổng–Ghép nối đèn LED



Chương trình điều khiển LED sáng

Chương trình để 8 LED nối với cổng ra 2C0H sáng rồi tắt led. 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 Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP



Đặt giá trị cổng vào = FFH đèn tắt

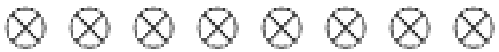
Làm trễ 64 chu kỳ

Đặt giá trị cổng vào = 00H đèn sáng

Làm trễ 64 chu kỳ

Chương trình điều khiển LED sáng

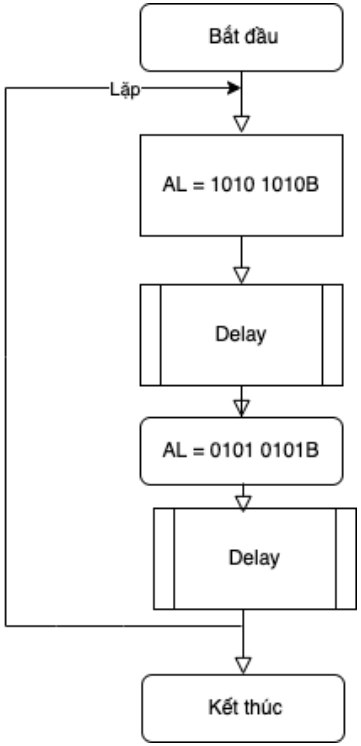
Chương trình để 8 LED nối với cổng ra 2C0H sáng rồi tắt led. đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn sẽ tắt. Minh họa trong hình phía dưới. Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh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



```
DK_LED EQU 2C0H          ; Cổng điều khiển LED
Lap:    MOV AL, FFH      ; Tắt tất cả các LED
        OUT DK_LED, AL
        MOV CX,64        ; Trễ bằng 64 lệnh NOP
Tre_1:  NOP
        LOOP Tre_1
        XOR AL,AL        ; Đặt AL=0
        OUT DK_LED,AL ; Bật tất cả các LED
        MOV CX,64
Tre_2:  NOP
                         ; Trễ bằng 64 lệnh NOP
        LOOP Tre_2
        JMP Lap
```

Chương trình điều khiển LED sáng

Chương trình để 8 LED (L1 – L8) nối với cổng ra 2C0H sáng rồi tắt xen kẽ nhau (đèn 1,3,5,7 sáng, đèn 2,4,6,8 tắt , sau đó đèn 1,3,5,7 tắt, đèn 2,4,6,8 sáng). 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 Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP



Đặt giá trị cổng vào =1 đèn sáng

Làm trễ 64 chu kỳ

Đặt giá trị cổng vào = 0 đèn tắt

Làm trễ 64 chu kỳ

Chương trình điều khiển LED sáng

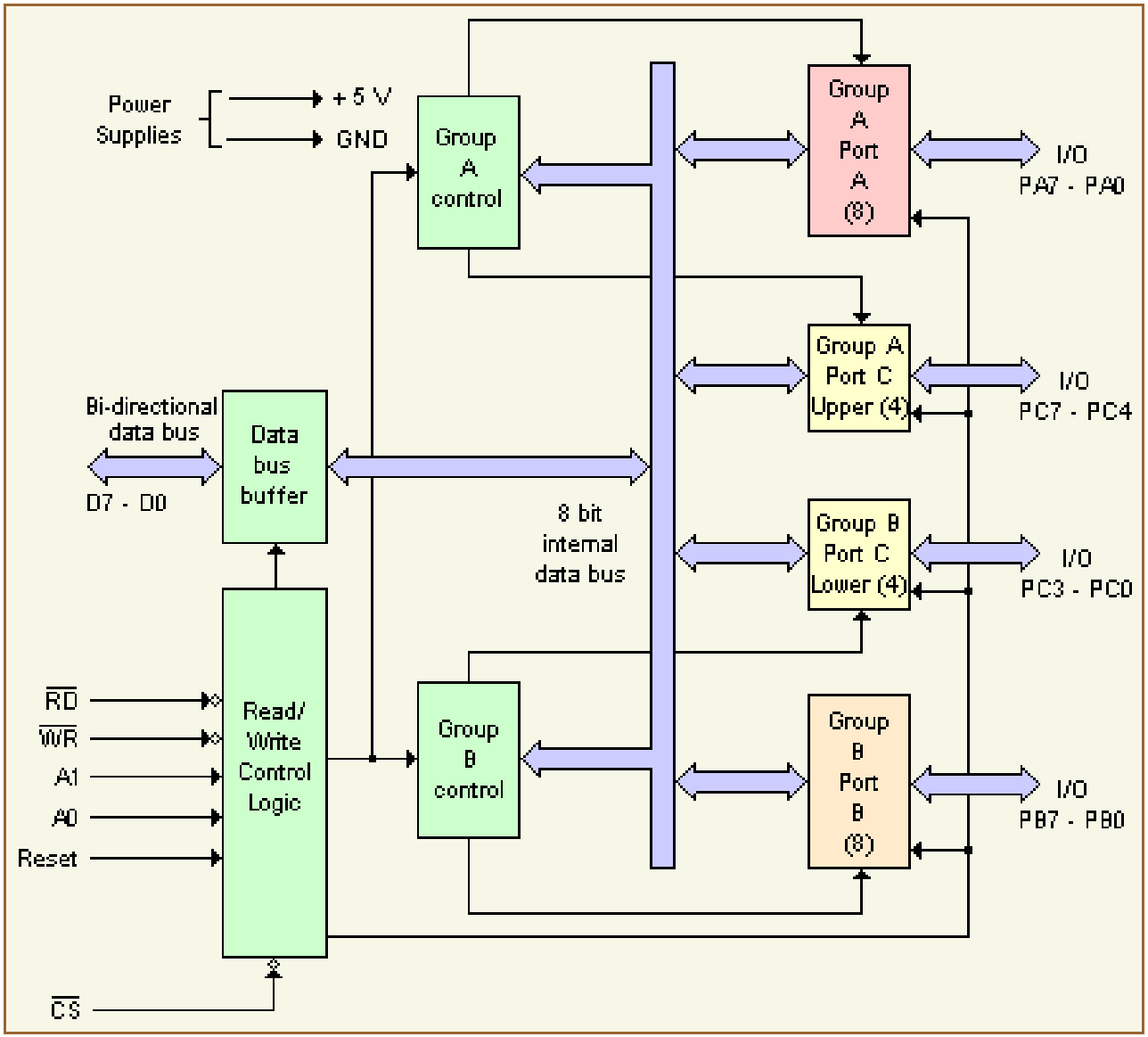
Chương trình để 8 LED nối với cổng ra 2C0H sáng rồi tắt xen kẽ nhau (đèn 1,3,5,7 sáng, đèn 2,4,6,8 tắt , sau đó đèn 1,3,5,7 tắt, đèn 2,4,6,8 sáng). Đèn được bật sáng nếu bít điều khiển tương ứng nhận giá trị 0. Ngược lại khi bít điều khiển bằng 1 thì đèn sẽ tắt. Minh họa trong hình phía dưới Tất cả các đèn đều cùng bật tắt liên tục với khoảng trễ giữa hai lần bật tắt là 64 chu kỳ lệnh NOP

```
DK_LED EQU 2C0H          ; Cổng điều khiển LED
Lap:    MOV AL, AAH      ; Tắt tất  các LED  1,3,5,7 và mở 2,4,6,8
        OUT DK_LED, AL
        MOV CX,64        ; Trễ bằng 64 lệnh NOP
Tre_1:  NOP
        LOOP Tre_1
        MOV AL, 55H      ; Đặt AL=0
        OUT DK_LED,AL ; Bật tất cả các LED 1,3,5,7 và tắt 2,4,6,8
        MOV CX,64
Tre_2:  NOP
        LOOP Tre_2       ; Trễ bằng 64 lệnh NOP
        JMP Lap
```

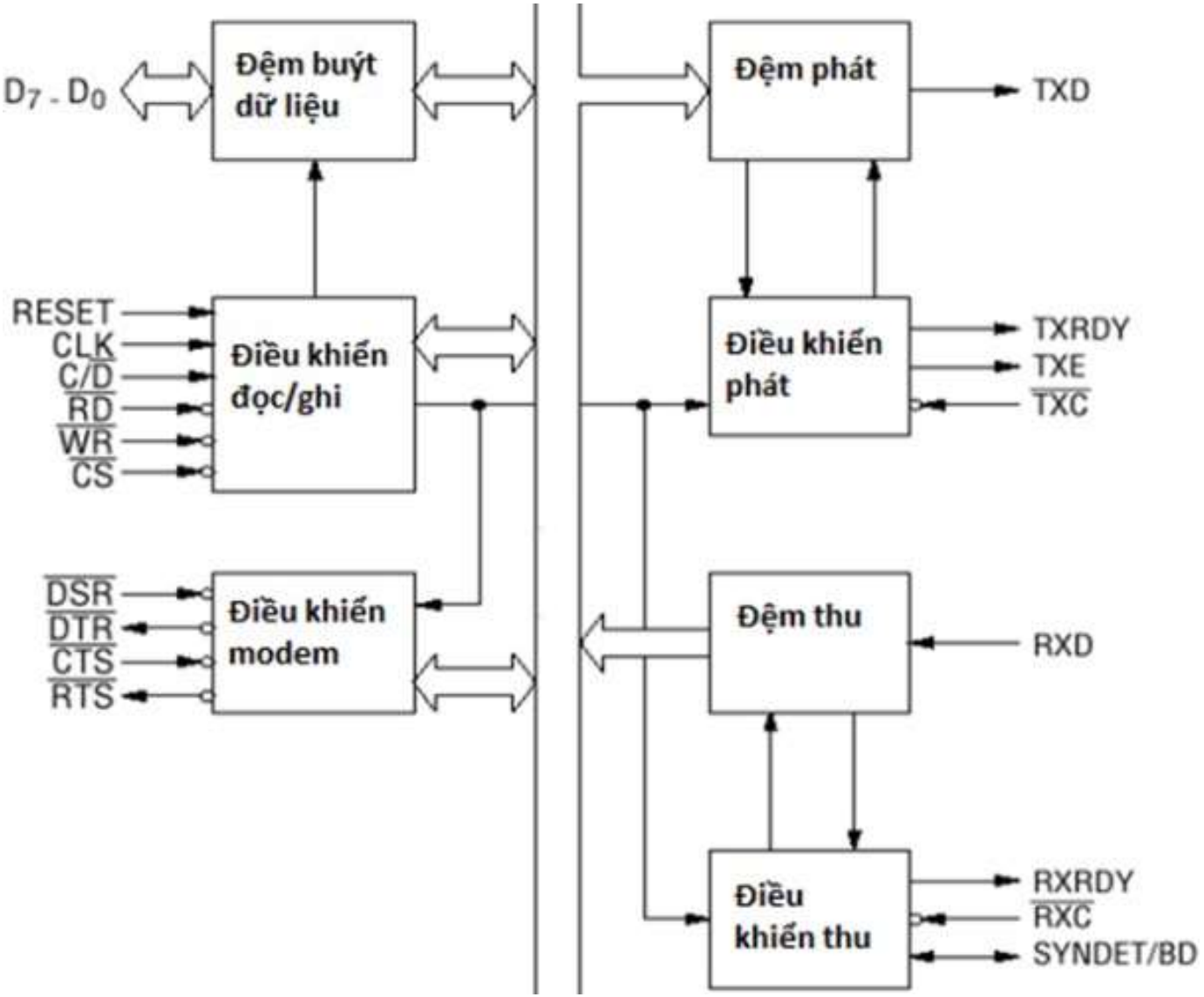
4.5. Giới thiệu một số mạch ghép nối vào ra

- Vi xử lý sử dụng một số mạch chuyên dụng phục vụ trao đổi dữ liệu với các thiết bị ngoại vi theo 2 phương pháp chính:
  - Trao đổi dữ liệu kiểu song song: sử dụng mạch ghép nối 8255A.
    - Cho phép trao đổi dữ liệu nhiều bit dữ liệu đồng thời → tốc độ cao
    - Không yêu cầu phải biến đổi dữ liệu
    - Hạn chế về khoảng cách do cần nhiều dây tín hiệu
  - Trao đổi dữ liệu kiểu nối tiếp: sử dụng mạch ghép nối 8250 hoặc 8251.
    - Có thể tăng khoảng cách truyền dữ liệu do cần ít dây tín hiệu
    - Tốc độ chậm
    - Cần biến đổi dữ liệu khi gửi đi (song song → nối tiếp) và khi nhận về (nối tiếp → song song).

4.5.1 Mạch vào ra song song 8255A

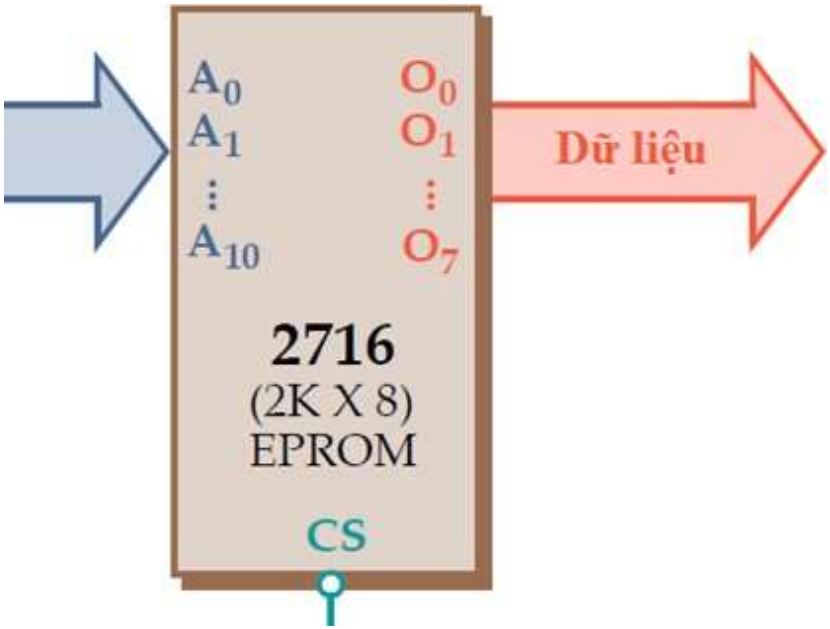


4.5.2 Mạch vào ra nối tiếp 8251A



Bài tập bổ sung – Xây dựng mạch giải mã địa chỉ

1. Xây dựng mạch giải mã địa chỉ dùng các mạch lô-gíc cơ bản cho bộ nhớ ROM dung lượng 4KB có địa chỉ cơ sở 05800H dùng vi mạch nhớ 2Kx8.



Bài tập bổ sung – Xây dựng mạch giải mã địa chỉ

2. Xây dựng mạch giải mã địa chỉ cho cổng vào có địa chỉ 8000h. Biết không gian cổng có địa chỉ tách biết với không gian bộ nhớ.
3. Xây dựng mạch giải mã địa chỉ cho cổng ra có địa chỉ 03F8h. Biết không gian cổng có địa chỉ tách biết với không gian bộ nhớ.