HỌC PHẦN
VI XỬ LÝ – VI ĐIỀU KHIỂN

LỚP: TĐH, CN ô tô

GV: Đỗ Văn Cần

ĐT: 0356906275

Web: gpktqn.com

HỌC PHẦN VI XỬ LÝ – VI ĐIỀU KHIỂN

Chương 1. Mở đầu (2 tiết)
Chương 2. Bộ Vi xử lý 8086/8088 của Intel
(Cấu trúc, thanh ghi - Câu 1 2đ)
Chương 3. Vi điều khiển học 8051 (
Chương 4. Lập trình hợp ngữ
Chương 5. Chức năng mở rộng MSC-51 (Time, 232, ngắt) Câu 2đ
Chương 6. Thiết kế giao tiếp ngoại vi
(Nút, LEd đơn, 7 đoạn, Matrix, LCD, ADC)
Câu 3 4 là 3đ phần cứng, phần mềm)
Phần mềm MIDE, Protues

Chương 1. Mở đầu

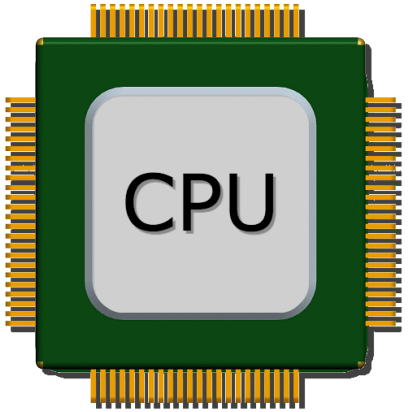
Chương 1. Mở đầu

1.1. Lịch sử hình thành bộ vi xử lý
1.2. Cấu trúc của hệ vi xử lý
1.3. Các cơ số tính toán
1.4. Các phép toán nhị phân

Chương 1. Mở đầu

1.1. Lịch sử hình thành bộ vi xử lý

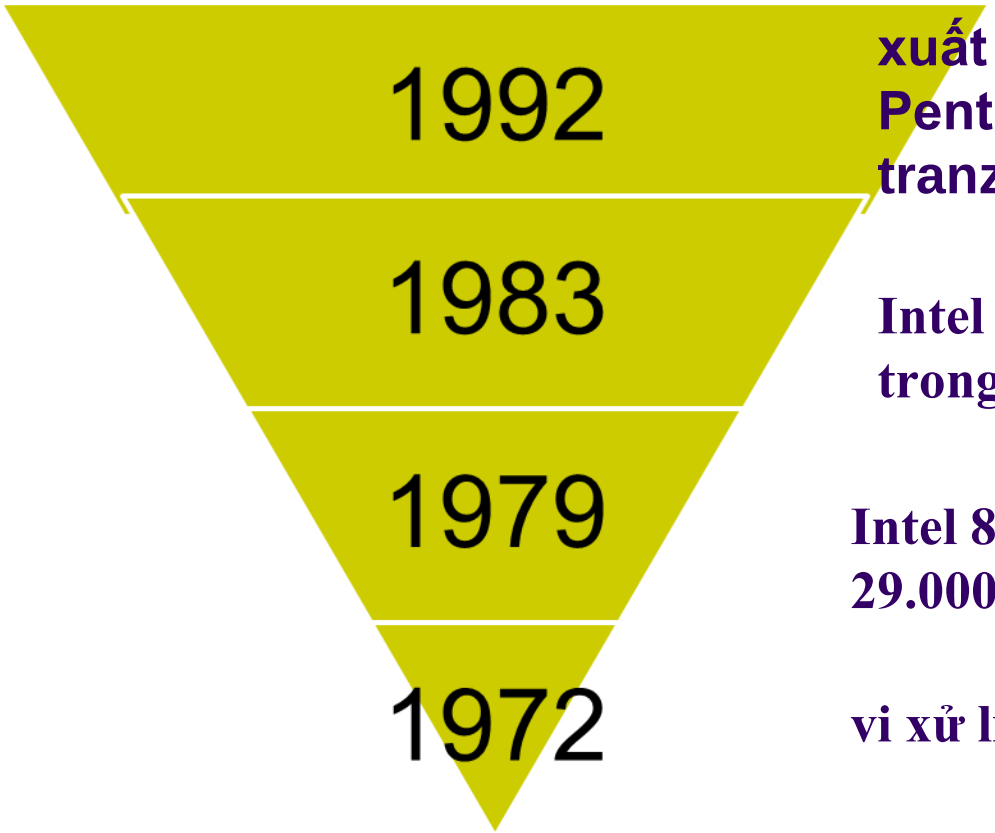
CPU viết tắt của chữ ***Central Processing Unit*** (tiếng Anh), tạm dịch là **Bộ xử lý trung tâm**, là mạch điện tử, là mạch điện tử thực hiện các câu lệnh, là mạch điện tử thực hiện các câu lệnh của chương trình máy tính, là mạch điện tử thực hiện các câu lệnh của chương trình máy tính bằng cách thực hiện các phép tính số học, logic, so sánh và các hoạt động nhập/xuất dữ liệu (Input/Output) cơ bản từ mã lệnh được định sẵn trong một máy tính. Thuật ngữ này đã được sử dụng trong ngành công nghiệp máy tính kể từ đầu những năm 1960.[1] Theo truyền thống, thuật ngữ "CPU" chỉ một bộ xử lý, cụ thể là bộ phận xử lý và điều khiển (Control Unit) của nó, phân biệt với những yếu tố cốt lõi khác của một máy tính nằm bên ngoài như bộ nhớ và mạch điều khiển xuất/nhập dữ liệu.[2]



Chương 1. Mở đầu

1.1. Lịch sử hình thành bộ vi xử lý

Intel Core i9 : Cung cấp tới 10 lõi đã unlock cho video 4K Ultra HD
CPU Intel Xeon với 3 dòng Xeon E3, Xeon E5 và Xeon E7



xuất hiện Intel 80586 còn gọi là Pentium 64 bit chứa 4 triệu tranzito

Intel đưa ra bộ vi xử lý 80286 dùng trong các máy vi tính họ AT

Intel 8086 là loại bộ xi xử lý 16 bit với 29.000 tranzito

vi xử lí 8-bit đầu tiên với tên Intel-8008

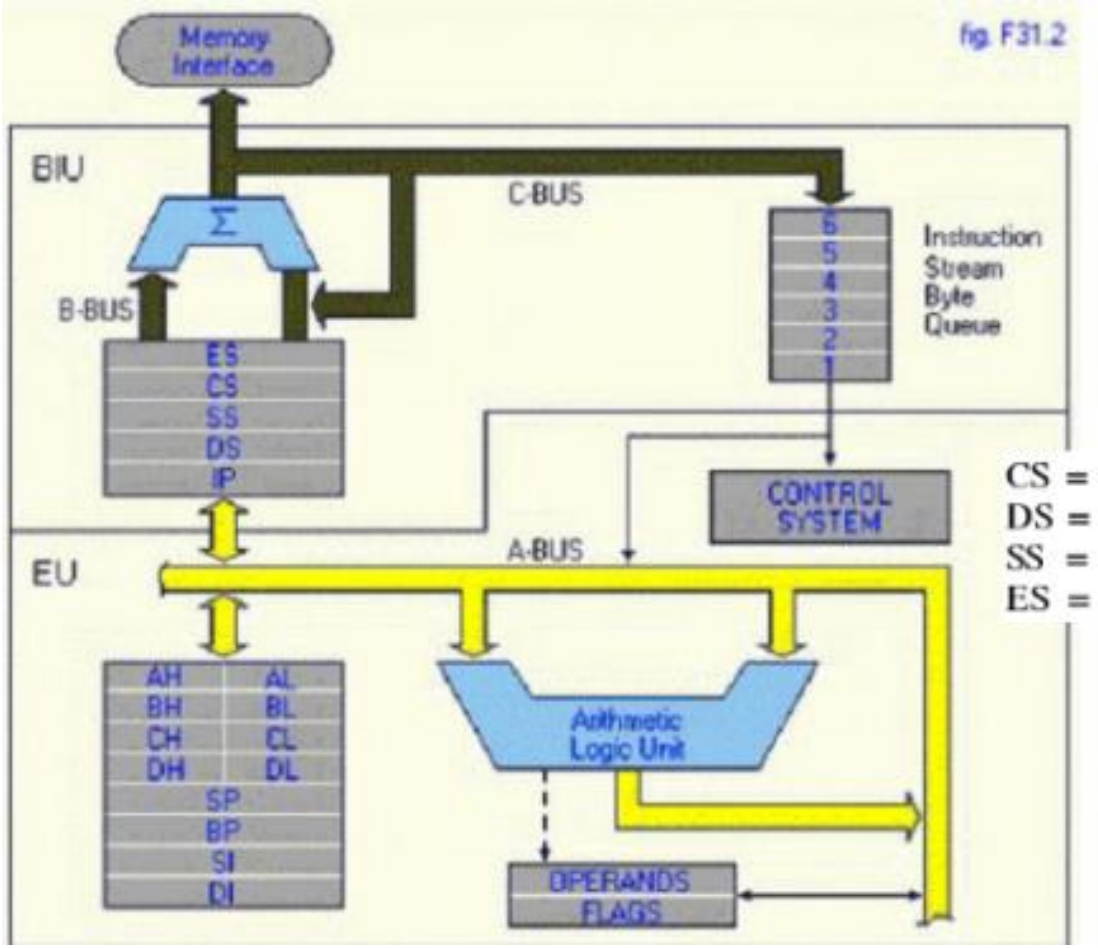
Chương 1. Mở đầu

1.2. Cấu trúc bộ vi xử lý

- The unit of data size
  - *Bit* : a binary digit that can have the value 0 or 1
  - *Byte* : 8 bits
  - *Nibble* : half of a bye, or 4 bits
  - *Word* : two bytes, or 16 bits
- The terms used to describe amounts of memory in IBM PCs and compatibles
  - *Kilobyte* (K): 2^{10} bytes
  - *Megabyte* (M) : 2^{20} bytes, over 1 million
  - *Gigabyte* (G) : 2^{30} bytes, over 1 billion
  - *Terabyte* (T) : 2^{40} bytes, over 1 trillion

Chương 1. Mở đầu

1.2. Cấu trúc bộ vi xử lý



BIU (Bus Interface Unit)
EU (Execution Unit)
CS (Control System)
ALU (Arithmetic/Logical Unit)
F (Register of Flag)

AX = Primary accumulator
BX = Accumulator and base register
CX = Accumulator and counter
DX = Accumulator and I/O addresser

CS = Program instruction
DS = General data and string source
SS = Stack operation SP is used as based register
ES = Destination string

Chương 1. Mở đầu

1.2. Cấu trúc bộ vi xử lý

Khối thực thi EU

- Khối điều khiển (Control System - CS): có mạch giải mã lệnh. Mã lệnh đọc vào từ bộ nhớ đưa đến đầu của bộ giải mã, các thông tin thu được từ đầu ra cục nó sẽ được đưa đến mạch tạo xung điều khiển, kết quả là thu được các dãy xung khác nhau (tùy theo mã lệnh) để điều khiển hoạt động của các bộ phận bên trong và bên ngoài CPU.
- Khối logic và số học (Arithmetic and Logic Unit ALU): dùng để thực hiện các thao tác khác nhau với các toán hạng cuả lệnh.

Chương 1. Mở đầu

1.2. Cấu trúc bộ vi xử lý

Khối phối ghép bus BIU

- Khối BIU có nhiệm vụ đưa ra địa chỉ, đọc mã lệnh từ bộ nhớ. Nói cách khác BIU chiu trách nhiệm đưa địa chỉ ra bus và trao đổi dữ liệu với bus.
- Trong BIU còn có bộ nhớ đệm lệnh với dung lượng 4 byte dùng để đưa các mã lệnh đọc được nằm sẵn sàng chờ EU xử lý (bộ lệnh này còn được gọi là hàng đợi lệnh)
- Tóm lại: khi CPU hoạt động EU sẽ cung cấp thông tin về địa chỉ cho BIU để khối này đọc lệnh và dữ liệu, còn bản thân nó thì giải mã lệnh và thực hiện lệnh

Chương 1. Mở đầu

1.2. Cấu trúc bộ vi xử lý

- Khi CPU 8088 hoạt động, EU sẽ cung cấp các thông tin về địa chỉ cho BIU đọc lệnh và dữ liệu đưa về giải mã lệnh và thực hiện lệnh. Khối EU gồm các bộ tính toán số học ALU (Arithmetical Logical Unit), bộ điều khiển CS (Control System) và các thanh ghi (Register). Tại CU có mạch giải mã lệnh sẽ nhận mã lệnh đọc vào từ bộ nhớ, xử lý và đưa ra mạch tạo xung điều khiển. Tuỳ theo mã lệnh ta sẽ thu được các dãy xung khác nhau để điều khiển các hoạt động khác nhau bên trong và bên ngoài của CPU. Khối ALU có nhiệm vụ thực hiện các thao tác khác nhau với các toán hạng của lệnh. Tóm lại EU duy trì trạng thái của CPU, kiểm soát các thanh ghi đa năng và toán hạng lệnh, tất cả các thanh ghi và đường truyền dữ liệu trong EU có dung lượng 16 bit.

Bộ vi xử lý thực hiện các lệnh theo các bước sau:

- Lấy lệnh từ bộ nhớ.
- Đọc toán hạng (nếu lệnh yêu cầu).
- Thực hiện lệnh.
- Ghi kết quả.

Chương 1. Mở đầu

1.3. Các cơ số tính toán

- Human beings use base 10 (*decimal*) arithmetic
  - There are 10 distinct symbols, 0, 1, 2, …, 9
- Computers use base 2 (*binary*) system
  - There are only 0 and 1
  - These two binary digits are commonly referred to as *bits*

Trong Tin học, ngoài hệ cơ số 10, người ta còn sử dụng hai loại hệ đếm sau:

- Hệ cơ số 2 (Hệ nhị phân): Chỉ sử dụng hai kí hiệu 0 và 1. Lấy ví dụ, $1011_2 = 1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 = 11_{10}$.
- Hệ cơ số 16 (Hệ thập lục phân hay hệ Hexa): Sử dụng các chữ số từ 0 tới 9 và 6 chữ cái A, B, C, D, E, F; trong đó A, B, C, D, E, F có giá trị lần lượt là $10, 11, 12, 13, 14, 15$. Lấy ví dụ, $16A = 10 \times 16^0 + 6 \times 16^1 + 1 \times 16^2 = 362_{10}$.

Chương 1. Mở đầu

1.3. Các cơ số tính toán

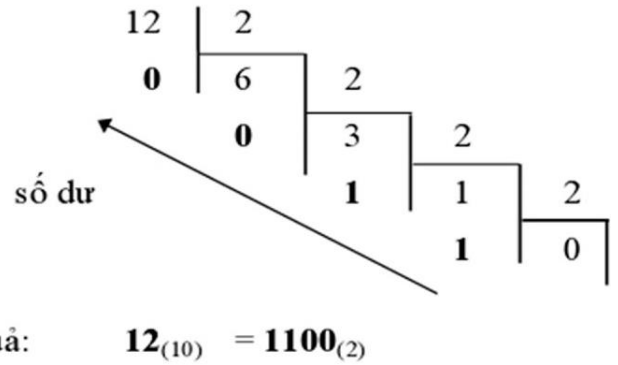
- Divide the decimal number by 2 repeatedly
- Keep track of the remainders
- Continue this process until the quotient becomes zero
- Write the remainders in reverse order to obtain the binary number

Ex. Convert 25_{10} to binary

	Quotient	Remainder	
25/2 =	12	1	LSB (least significant bit)
12/2 =	6	0	
6/2 =	3	0	
3/2 =	1	1	
1/2 =	0	1	MSB (most significant bit)

Therefore $25_{10} = 11001_2$

Chuyển 12_{10} sang hệ 2



Kết quả: $\mathbf{12_{(10)} = 1100_{(2)}}$

Bài tập: 1234_{10} = ???$_2$

Chương 1. Mở đầu

1.3. Các cơ số tính toán

- Know the weight of each bit in a binary number
- Add them together to get its decimal equivalent

Ex. Convert 11001_2 to decimal

Weight:	2^4	2^3	2^2	2^1	2^0
Digits:	1	1	0	0	1
Sum:	16 +	8 +	0 +	0 +	$1 = 25_{10}$

- Use the concept of weight to convert a decimal number to a binary directly

Ex. Convert 39_{10} to binary

$32 + 0 + 0 + 4 + 2 + 1 = 39$

Therefore, $39_{10} = 100111_2$

Department of Computer Science and Information Engineering
National Cheng Kung University, TAIWAN 6

Bài tập: $10101_2 = ???_{10}$

$1100_2 = ?_{10}$	$7AD_{16} = ?_{10}$
$1111_2 = ?_{10}$	$5DE_{16} = ?_{10}$
$1000_2 = ?_{10}$	$8F_{16} = ?_{10}$
$1001_2 = ?_{10}$	$25D_{16} = ?_{10}$

Chương 1. Mở đầu

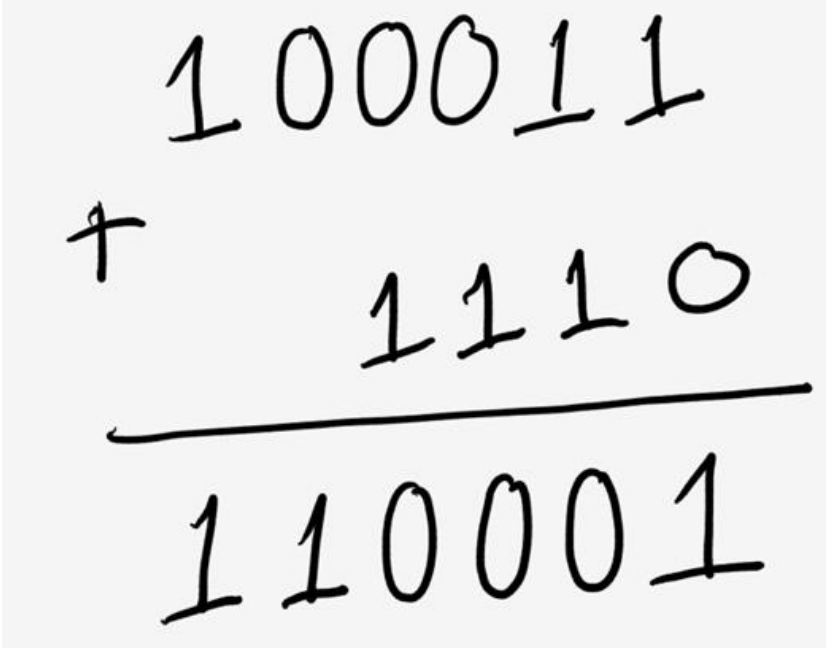
1.3. Các cơ số tính toán

Thập phân (Decimal)	Nhị phân (Binary)	Thập lục phân (Hexadecimal)	Bát phân (Octal)
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

Chương 1. Mở đầu

1.4. Các phép toán nhị phân

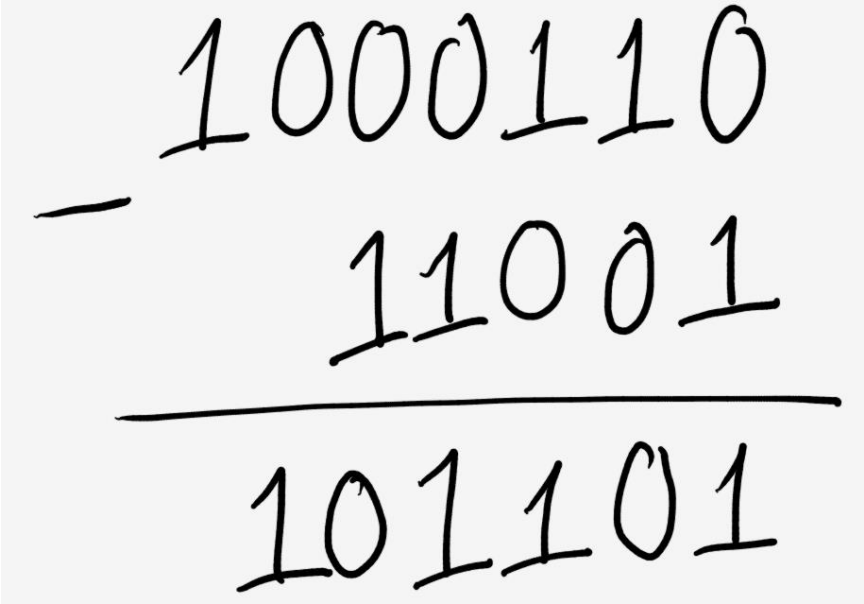
Quy tắc:
0+0=0
0+1=1+0=1
1+1=10



Chương 1. Mở đầu

1.4. Các phép toán nhị phân

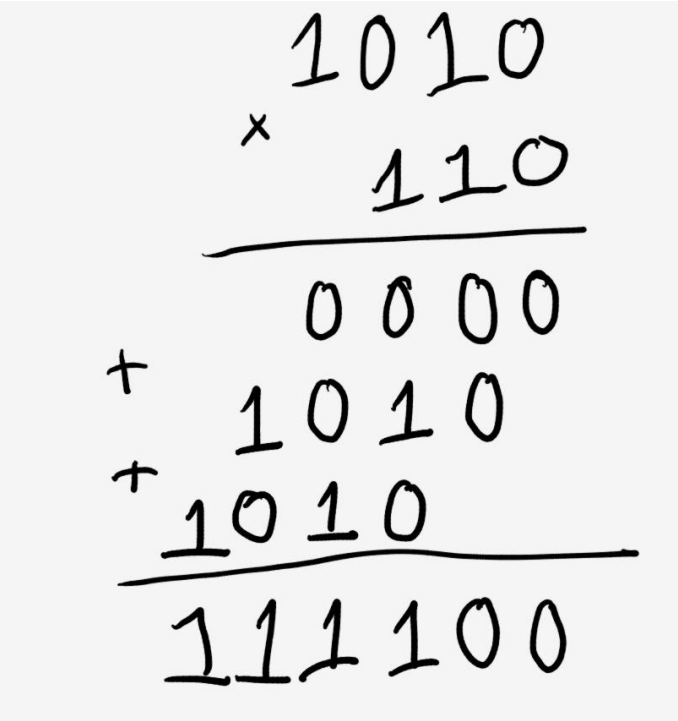
0-0=0
0-1=-1(“mượn”, vì trong phép tính số nhị phân không được số âm)
1-0=1
1-1=0
-1-1=-10



Chương 1. Mở đầu

1.4. Các phép toán nhị phân

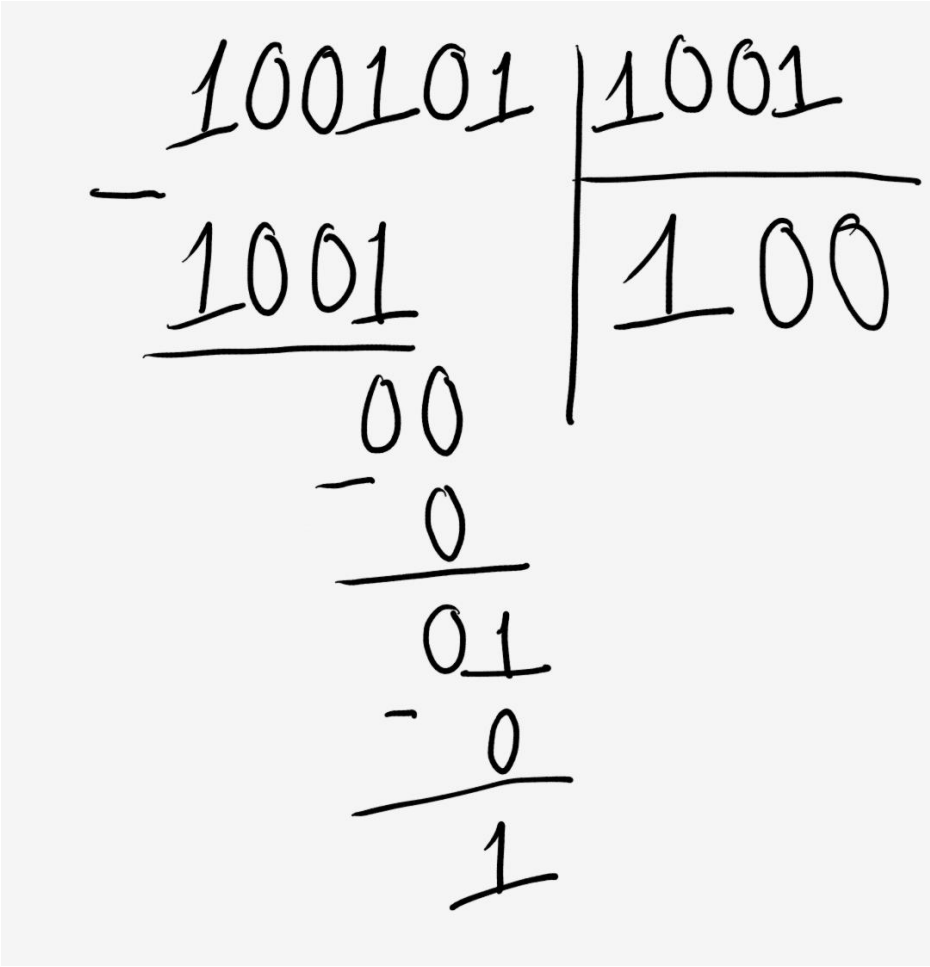
Bài tập: 1011x1010




```
   1011   (A)
 × 1010   (B)
 --------
   0000
  1011
 0000
1011
---------------
```

Chương 1. Mở đầu

1.4. Các phép toán nhị phân



Bài tập: 10011111:1100

```
10011111 | 1100
 1100    | 1101
 1111
 1100
   1111
   1100
     11
```

Chương 1. Mở đầu

1.4. Các phép toán nhị phân

Bài tập về nhà

110101011 + 10101=
1010011 - 1010=
10110 x 110010 =
11010011 : 1001 =

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.1. Kiến trúc tổ chức

2.1. Thanh ghi

2.3. Tập lệnh

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.1. Kiến trúc tổ chức

Các thông số của 8086 như sau:

- Năm sản xuất: 6/1978
- fclkmax (đồng hô nhịp): 10MHz
- MIPS (triệu lệnh/s): 0, 33
- Số tranzitor: 29000
- Bus số liệu: 16 bit
- Bus địa chỉ: 20 bit
- Khả năng địa chỉ: 1 MB
- Số chân: 40
- Độ dài bộ nhớ đệm lệnh (hàng đợi): 6 byte
- Có thể thao tác với bit, byte, từ, từ khối.
- Có khả năng thực hiện phép tính với các số 8 và 16 bit có dấu hoặc không có dấu dạng nhị phân hoặc thập phân, bao gồm cả phép chia và nhân

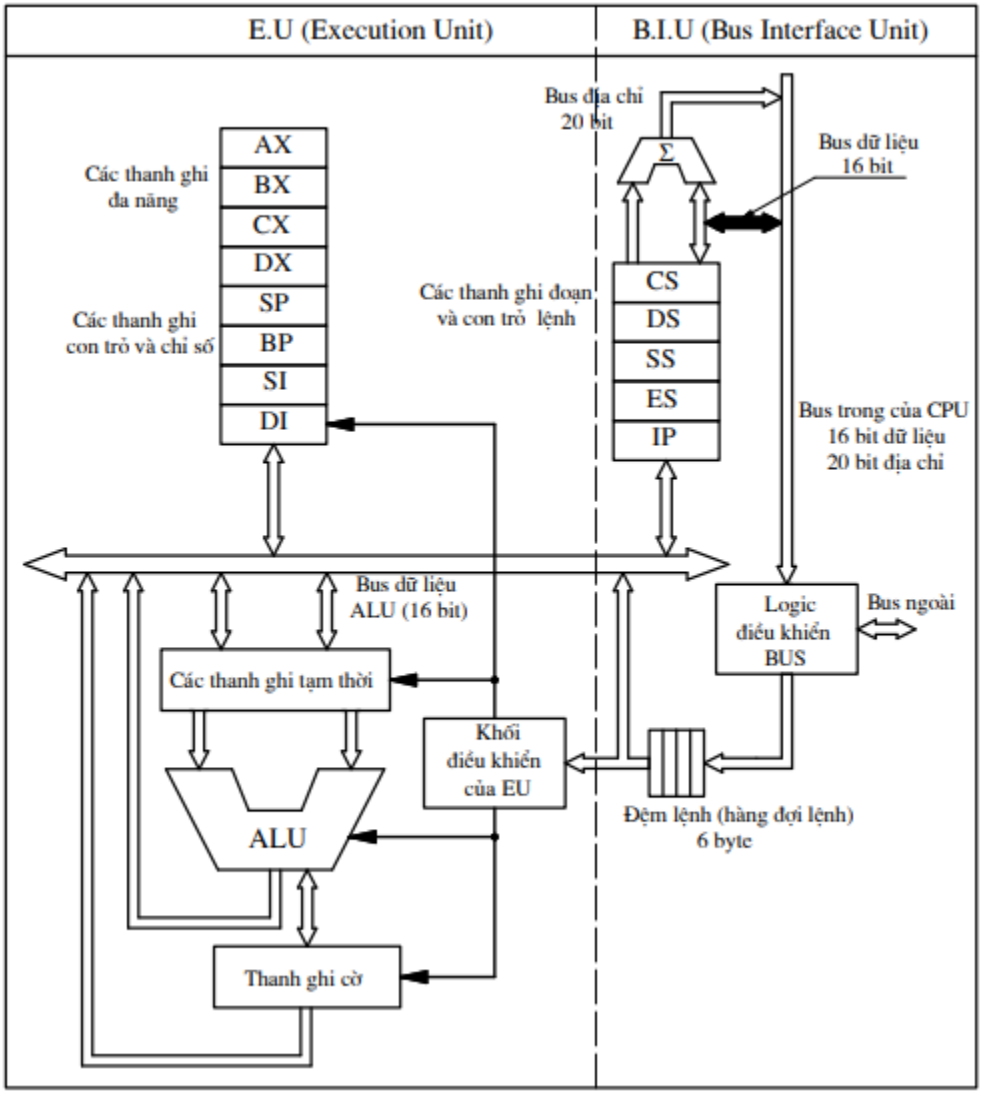
Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.1. Kiến trúc tổ chức

EU: Execution Unit, khối thực hiện lệnh.

BIU: Bus Interface Unit, khối phối ghép bus.

ALU: Arithmetic and Logic Unit, khối số học và logic.



Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.1. Kiến trúc tổ chức

BIU: Cung cấp các chức năng liên quan đến việc nhận lệnh và xếp hàng lệnh, lưu trữ các toán hạng và định vị các địa chỉ. Khối này cũng cung cấp các chức năng điều khiển BUS cơ sở. Trong hầu hết các trường hợp thời gian thực hiện lệnh và lấy lệnh và thực hiện lệnh là trùng nhau. Chính điều này làm tăng khả năng hoạt động của vi xử lý thông qua việc cải thiện Bus. Trong khi khối thực hiện lệnh đang bận rộn với lệnh hiện thời thì BIU đã có thể bắt đầu việc lấy các lệnh kế tiếp từ bộ nhớ và phần cuối của chúng được đặt trong một RAM nội bộ tốc độ cao được gọi là hàng đợi. Độ dài của hàng đợi này với vi xử lý 8086 là 6byte. Kỹ thuật hàng đợi lệnh cho phép BIU sử dụng bộ nhớ rất hiệu quả. BIU sẽ lấy mã lệnh trong bộ nhớ rồi đưa vào hàng đợi. Theo cách này BIU có thể cung cấp các lệnh một cách liên tục mà không độc chiếm BIU. Điều này làm giảm đáng kể thời gian chết trên Bus. Hàng đợi lệnh làm việc như một bộ đệm lệnh FIFO (First In First Out, vào trước ra trước).

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.1. Kiến trúc tổ chức

EU: Nhận các lệnh được lấy ra trước từ hàng đợi lệnh và cung cấp các toán hạng, các địa chỉ cho BIU để khối này đọc lệnh và dữ liệu. Trong khi đó bản thân EU sẽ giải mã lệnh, thực hiện, rồi lại chuyển các kết quả tới BIU để lưu trữ. Thao tác được thực hiện trước tiên của EU là việc giải mã lệnh và khoảng thời gian này có vẻ như là lãng phí đối với CPU khi mà dường như chẳng có một hoạt động về mặt điện nào diễn ra ở trên Bus. Nhưng trong thực tế, chính khoảng thời gian này là khoảng thời gian được BIU khai thác để lấy trước các câu lệnh tiếp theo như đã được mô tả ở trên.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.1. Kiến trúc tổ chức

ALU: Đây chỉ là một tập con của EU, nhưng trong thực tế nó giống như một phần có cấu trúc độc lập, chịu trách nhiệm thực hiện các thao tác số học và các thao tác logic. Các toán hạng có thể là dữ liệu tức thì, dữ liệu từ các thanh ghi hoặc dữ liệu được lưu trữ trong bộ nhớ. Trong khi đó kết quả lại được định vị trong một thanh ghi hoặc trong bộ nhớ và 6 cờ trạng thái được cập nhật dựa trên kết quả của các thao tác này

HỌC PHẦN VI XỬ LÝ – VI ĐIỀU KHIỂN

Vẽ sơ kiến trúc VXL 8086 và chức năng từng khối (2đ)

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

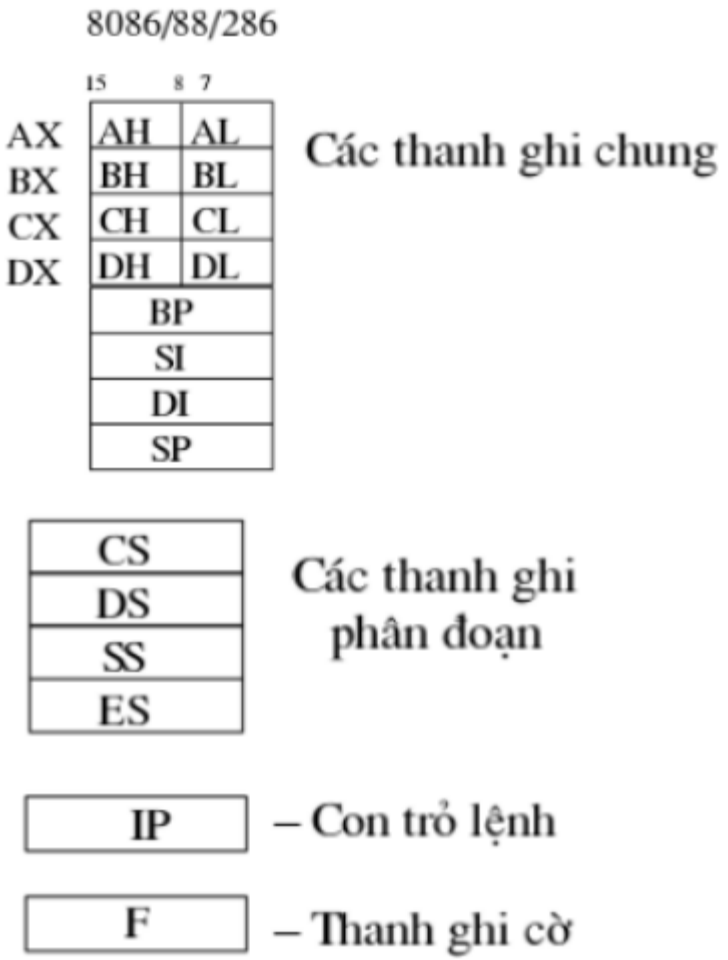
Các thanh ghi có thể được chia làm 4 nhóm lần lượt có tên là:

- Các thanh ghi đoạn: CS, DS, SS, ES.

- Các thanh ghi đa năng: AX, BX, CX, DX.

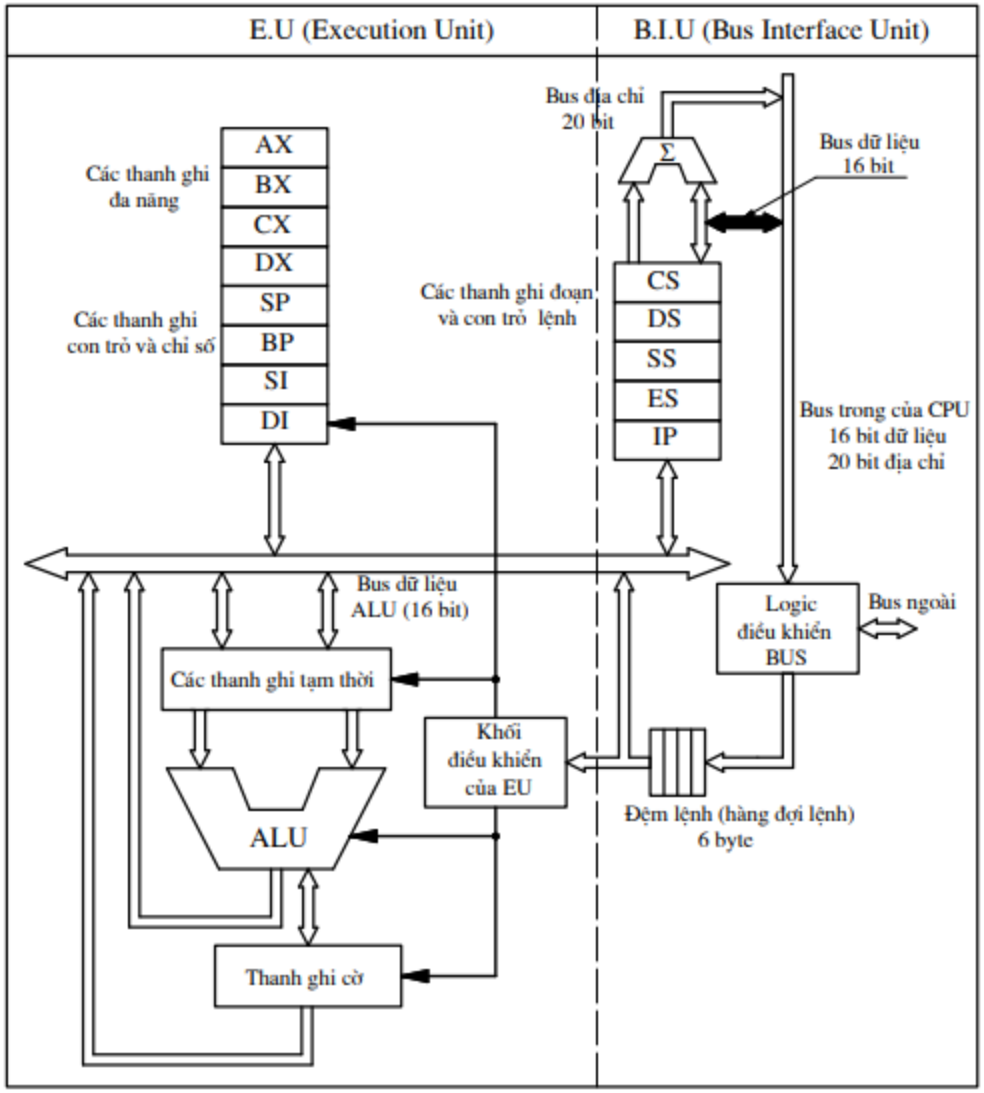
- Các thanh ghi con trỏ và chỉ số: IP, BP, SP, SI, DI.

- Thanh ghi cờ. FR (Flag).



Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086



Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Thanh ghi đoạn

Khối BIU đưa ra trên BUS địa chỉ 20 bit địa chỉ. Như vậy 8086 có khả năng phân biệt được 2^{20} = 1.048.576 =1M ô nhớ hay 1MB. Trong không gian 1MB này bộ nhớ cần được chia ra thành các vùng khác nhau dành riêng để:

- Chứa mã chương trình.
- Chứa dữ liệu và kết quả trung gian của chương trình.
- Tạo ra một vùng nhớ đặc biệt gọi là ngăn xếp (stack) dùng vào việc quản lý các thông số của bộ vi xử lý khi gọi chương trình con hoặc trở về từ chương trình con.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Thanh ghi đoạn

Vi xử lý 8086/8088 có các thanh ghi 16 bit:

- CS (Code Segment): Thang ghi đoạn mã, chứa địa chỉ bắt đầu của đoạn chương trình (đoạn mã) ROM.
- DS (Data Segment): Thanh ghi đoạn dữ liệu, chứa địa chỉ bắt đầu của đoạn dữ liệu, bao gồm các tham số, các biến, các mảng số liệu…
- SS (Stack Segment): Thanh ghi đoạn ngăn xếp, chứa địa chỉ bắt đầu của mảng stack. Đây là một mảng của RAM, nơi mà dữ liệu tồn tại trong các thanh ghi được lưu trữ trong suốt quá trình ngắt.
- ES (Extra Segment): Thanh ghi đoạn dữ liệu phụ, chứa địa chỉ bắt đầu của vùng nhớ bổ sung.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Thanh ghi đoạn

Mọi sự trao đổi thông tin trong hệ thống vi xử lý đều dùng địa chỉ vật lý, còn địa chỉ được tạo bởi thanh ghi đoạn và thanh ghi lệch như trên được gọi là địa chỉ logic và được ký hiệu như sau:

Địa chỉ logic = Thanh ghi đoạn: Thanh ghi lệch

Địa chỉ logic tồn tại dưới dạng giá trị các thanh ghi cụ thể bên trong CPU và khi cần thiết truy nhập ô nhớ nào đó thì nó phải được đổi ra địa chỉ vật lý để rồi đưa lên bus địa chỉ . Việc chuyển đổi này do một bộ tạo địa chỉ thực hiện. Địa chỉ vật lý của ô nhớ được tính theo công thức sau:

20 bit địa chỉ vật lý = Thanh ghi đoạn x 16 + Thanh ghi lệch

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Ví dụ: địa chỉ vật lý của 32412H có thể được tạo ra từ các giá trị.

Thanh ghi đoạn	Thanh ghi lệch
3000H	2412H
3200H	0412H
3240H	0012H
...	

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Các thanh ghi đa năng

Trong khối EU có 4 thanh ghi đa năng AX, BX, CX, DX. Điều đặc biệt là khi cần chứa dữ liệu 8 bit thì mỗi thanh ghi này có thể tách ra làm 2 thanh ghi 8 bit cao và thấp làm việc độc lập nhau, đó là các thanh ghi AH và AL, BH và BL, CH và CL, DH và DL

	7 … 0	7 … 0	
Accumulator	AH	AL	AX
Base	BH	BL	BX
Counter	CH	CL	CX
Data	DH	DL	DX

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Các thanh ghi đa năng

- AX (Accumulator, Acc): Thanh chứa, các kết quả của các thao tác thường được chứa ở đây, nếu kết quả là 8 bit thì thanh ghi AL được gọi là Acc.
- BX (Base): Thanh ghi cơ sở, thường chứa địa chỉ cơ sở của một bảng trong bộ nhớ.
- CX (Count): Thanh ghi đếm, thường dùng để chứa số lần lặp của lệnh lặp LOOP, còn CL thường dùng chứa só lần dịch hoặc quay trong các lệnh dịch hoặc quay.
- DX (Data): Thanh ghi dữ liệu. DX và AX tham gia vào thao tác của các phép nhân hoặc chia 16 bit, DX còn dùng để chứa địa chỉ của các cổng trong các lệnh vào/ra dữ liệu trực tiếp (IN/OUT)

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Các thanh ghi con trỏ và chỉ số

- IP (Instruction Pointer): Con trỏ lệnh, IP luôn trỏ vào lệnh tiếp theo sẽ được thực hiện nằm trong đoạn mã CS. Địa chỉ đầy đủ của lệnh tiếp theo này ứng với CS:IP và được xác định theo cách đã nói ở trên.
- BP (Base Pointer): Con trỏ cơ sở, BP luôn trỏ vào một dữ liệu nằm trong đoạn ngăn xếp SS. Địa chỉ đầy đủ của một phần tử trong đoạn ngăn xếp ứng với SS:BP và được xác định theo cách đã nói ở trên.
- SP (Stack Pointer): Con trỏ ngăn xếp, luôn trỏ vào đỉnh hiện thời của ngăn xếp nằm trong đoạn ngăn xếp SS. Địa chỉ đầy đủ của đỉnh ngăn xếp ứng với SS:SP và được xác định theo cách đã nói ở trên.
- SI (Source Index): Chỉ số nguồn, SI chỉ vào dữ liệu trong đoạn dữ liệu DS mà địa chỉ đầy đủ tương ứng với DS:SI và được xác định theo cách đã nói ở trên.
- DI (Destination Index): Chỉ số đích, DI chỉ vào dữ liệu trong đoạn dữ liệu DS mà địa chỉ đầy đủ tương ứng với DS:DI và được xác định theo cách đã nói ở trên.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Thanh ghi cờ FR (Flag Register)

Đây là thanh ghi khá đặc biệt trong CPU mỗi bit của nó để phản ánh một trạng thái nhất định của kết qủa phép toán do ALU thực hiện hoặc một hoạt động của EU. Dựa vào các cờ này mà người lập trình có thể đưa ra các lệnh thích hợp tiếp theo cho vi xử lý (các lệnh nhảy có điều kiện). Thanh ghi cờ có 16 bit nhưng chỉ sử dụng 9 bit làm bit cờ

X	X	X	X	O	D	I	T	S	Z	X	A	X	P	X	C

x: Không được định nghĩa

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Thanh ghi cờ FR (Flag Register)

Các cờ trạng thái

- C hoặc CF (Carry Flag): Cờ nhớ CF = 1 khi có nhớ hoặc mượn từ MSB.
- F hoặc P (Parity Flag): Cờ chẵn lẻ, phản ánh tính chẵn lẻ của tổng số bit 1 có trong kết quả. CF = 1 khi tổng số bit 1 trong kết quả là chẵn.
- A hoặc AF (Auxiliary carry Flag): cờ nhớ phụ, rất có ý nghĩa khi ta làm việc với các số BCD, AF = 1 khi có nhớ hoặc mượn từ một số BCD thấp (4 bit thấp) sang một số BCD cao (4bit cao).
- Z hoặc ZF (Zero Flag): Cờ rỗng, ZF = 1 khi kết qủa bằng 0.
- S hoặc SF (Sign Flag): Cờ dấu, SF = 1 khi kết quả âm.
- O hoặc OF (Overflow Flag): Cờ tràn, OF = 1 khi kết quả là số bù hai vượt ra ngoài giá trị biểu diễn của nó.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.2. Thanh ghi trong 8086

Thanh ghi cờ FR (Flag Register)

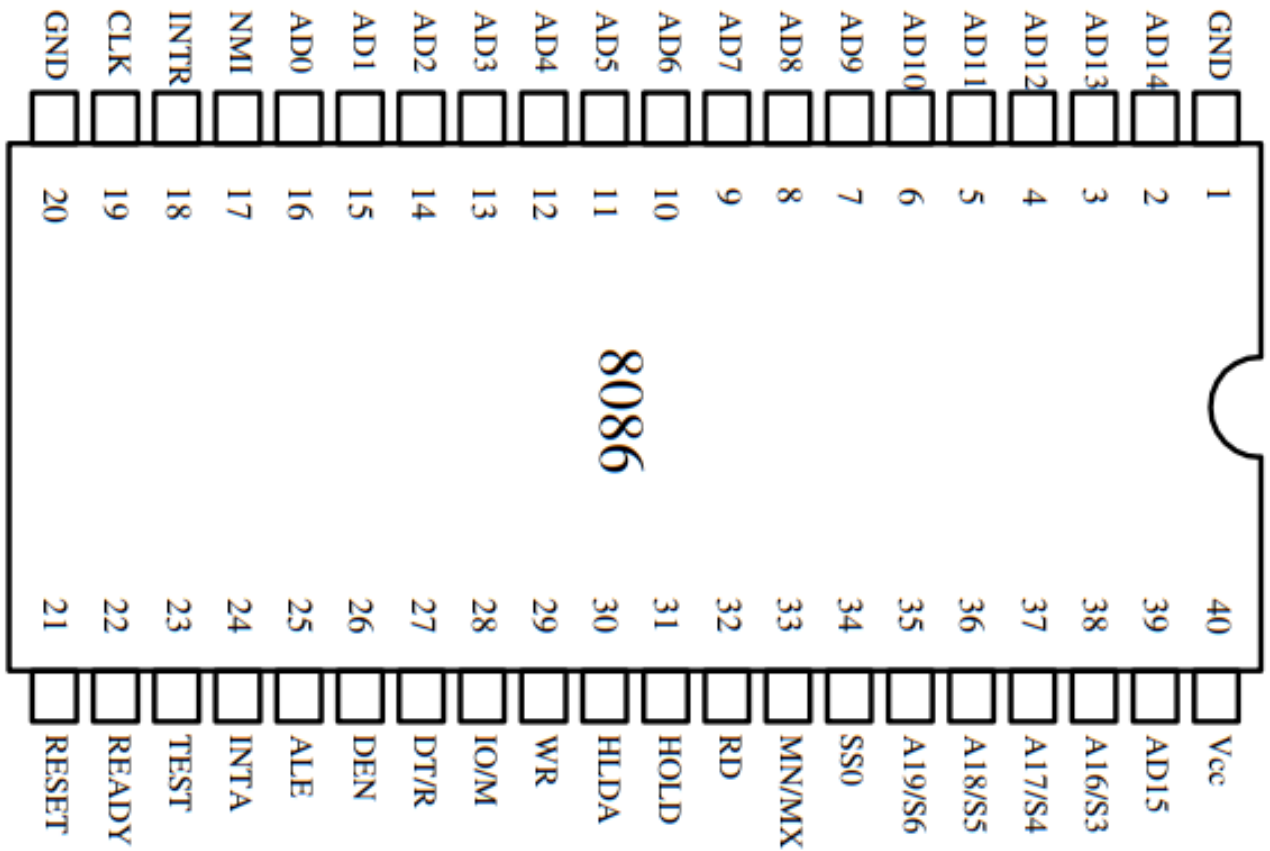
Các cờ điều khiển (có thể lập hoặc xoá bằng các lệnh riêng)

- T hoặc TF (Trap Flag): Cờ bẫy, TF = 1 thì CPU làm việc ở chế độ chạy từng lệnh (chế độ này dùng khi cần tìm lỗi chương trình)
- I hoặc IF (Interrupt enable Flag): Cờ cho phép ngắt, IF = 1 thì CPU cho phép các yêu cầu ngắt được tác động.
- D hoặc DF (Direction Flag): Cờ hướng, DF = 1 khi CPU làm việc với chuỗi ký tự theo ký tự từ phải sang trái (vì vậy D chính là cờ lùi).

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Mô tả chức năng các chân của vi xử lý 8086



Chương 2. Bộ Vi xử lý 8086/8088 của Intel

Câu 1: Nêu cấu tạo (hình vẽ) Sơ đồ cấu trúc của vi xử lý 8086/8088

Câu 2: Trình bày ngắn gọn chức năng các thanh ghi của vi xử lý 8086/8088

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

- AD0 ÷ AD15 [I, O]: Các chân dồn kênh cho các tín hiệu của bus dữ liệu và bus địa chỉ. Xung ALE sẽ báo cho mạch ngoài biết khi nào trên các đường đó có tín hiệu dữ liệu (ALE=0) hoặc địa chỉ (ALE=1). Tín hiệu này chuyển sang trạng thái trở kháng cao khi Bus nội bộ ghi nhận tín hiệu treo.
- A16/S3, A17/S4, A18/S5, A19/S6 [O]: Địa chỉ/trạng thái. Địa chỉ A16 – A19 sẽ có mặt tại các chân đó khi ALE=1 còn khi ALE=0 thì trên các chân đó có tín hiệu trạng thái S3 – S6. Bit S6=0 liên tục, bit S5 phản ánh giá trị bit IF của thanh ghi cờ, hai bit S3, S4 phối hợp với nhau để chỉ ra việc truy nhập các thanh ghi đoạn. Tín hiệu này chuyển sang trạng thái trở kháng cao khi Bus nội bộ ghi nhận tín hiệu treo

AD17/S4	AD16/S3	Truy nhập đến
0	0	Đoạn dữ liệu phụ (ES)
0	1	Đoạn ngăn xếp (SS)
1	0	Đoạn mã (CS) hoặc không đoạn nào
1 S6 luôn là 0	1	Đoạn dữ liệu (DS)

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

- RD [O]: Đọc. Tín hiệu đọc cho biết bộ vi xử lý đang thực hiện đọc bộ nhớ hay đọc I/O phụ thuộc vào trạng thái chân S2. RD=0 thì bus dữ liệu sẵn sàng nhận số liệu từ bộ nhớ hoặc thiết bị ngoại vi. Tín hiệu này chuyển sang trạng thái trở kháng cao khi Bus nội bộ ghi nhận tín hiệu treo.
- READY [I]: Tín hiệu báo cho CPU biết tình trạng sẵn sàng của thiết bị ngoại vi hay bộ nhớ. Khi READY=1 thì CPU thực hiện đọc/ghi mà không cần chèn thêm các chu kỳ đợi. Ngược lại khi thiết bị ngoại vi hay bộ nhớ có tốc độ hoạt động chậm, chúng có thể đưa tín hiệu READY=0 để báo cho CPU biết mà chờ chúng, lúc này CPU tự động kéo dài thời gian thực hiện lệnh đọc/ghi bằng cách chèn thêm các chu kỳ đợi.
- INTR [I]: tín hiệu yêu cầu ngắt che được. Khi có yêu cầu ngắt mà cờ cho phép ngắt IF=1 thì CPU kết thúc lệnh đang làm dở, sau đó đi vào chu kỳ chấp nhận ngắt và đưa ra bên ngoài tín hiệu INTA=0.
- TEST [I]: Tín hiệu tại chân này được kiểm tra bởi lệnh WAIT (xem phần tập lệnh). Khi CPU thực hiện lệnh WAIT mà lúc đó tín hiệu TEST=1 nó sẽ chờ cho đến khi TEST=0 thì nó mới thực hiện lệnh tiếp theo

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

- NMI [I]: Tín hiệu yêu cầu ngắt. Tín hiệu này không chịu sự khống chế của cờ IF và nó sẽ được CPU nhận biết bằng tác động của sườn lên của xung yêu cầu ngắt. Nhận được yêu cầu này CPU kết thúc lệnh đang làm dở sau đó nó chuyển sang chương trình phục vụ ngắt kiểu INT 2.
- RESET [I]: Tín hiệu khởi động lại 8086. Khi RESET=1 kéo dài ít nhất trong thời gian 4 chu kỳ đồng hồ thì 8086 buộc phải khởi động lại.
- CLK [I]: Tín hiệu đồng hồ (xung nhịp). Xung nhịp có độ rỗng là 77% và cung cấp nhịp làm việc cho CPU.
- Vcc [I]: Chân nguồn nuôi, tại đây CPU được cung cấp nguồn +5V±10%, 340mA.
- GND [O]: Hai chân nguồn để nối với điểm 0V của nguồn nuôi.
- MN/MX [I]: Chân điều khiển hoạt động của CPU theo chế độ MIN/MAX.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ MIN: Chân MN/MX được nối thẳng vào +5V mà không qua điện trở. Trong chế độ MIN tất cả các tín hiệu điều khiển liên quan đến thiết bị ngoại vi truyền thống và bộ nhớ đã có sẵn trong 8086, vì vậy việc phối ghép với các thiết bị đó rất dễ dàng và chính vì tận dụng được các phối ghép ngoại vi có sẵn nên có thể giảm giá thành hệ thống.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

- IO/M [O]: Tín hiệu này phân biệt trong thời điểm đã định phần tử nào trong các thiết bị vào/ra (IO) hoặc bộ nhớ (M) được chọn làm việc với CPU. Trên bus địa chỉ lúc đó sẽ có các địa chỉ tương ứng của các thiết bị đó. Chân này ở trạng thái trở kháng cao khi µP chấp nhận treo.
- WR [O]: Xung cho phép ghi. Khi CPU đưa ra WR=0 thì trên bus dữ liệu các dữ liệu đã ổn định và chúng sẽ được ghi vào bộ nhớ hoặc thiết bị ngoại vi tại thời điểm đột biến WR=1. Chân này ở trạng thái trở kháng cao khi µP chấp nhận treo.
- ALE [O]: Xung cho phép chốt địa chỉ. Khi ALE=1 có nghĩa là trên bus dồn kênh AD có các địa chỉ của thiết bị vào/ra hay của ô nhớ. ALE không bao giờ ở trạng thái trở kháng cao, khi CPU bị treo thì ALE=0.
- DT/R [O]: Tín hiệu điều khiển các đệm hai chiều của bus dữ liệu để chọn chiều chuyển trên bus D. Chân này ở trạng thái trở kháng cao khi µP chấp nhận treo.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

- DEN [O]: Tín hiệu báo cho bên ngoài biết là lúc này trên bus dồn kênh AD có dữ liệu ổn định. Chân này ở trạng thái trở kháng cao khi µP chấp nhận treo.
- HOLD [I]: Tín hiệu yêu cầu treo CPU để mạch ngoài thực hiện việc trao đổi dữ liệu với bộ nhớ bằng cách thâm nhập trực tiếp (Direct Memory Access, DMA). Khi HOLD=1, CPU sẽ tự tách ra khỏi hệ thống bằng cách treo bus A, bus D, bus C của nó (các bus ở trạng thái trở kháng cao) để bộ điều khiển DMA (DMA Controller, DMAC) có thể lấy được quyền điều khiển hệ thống để làm các công việc trao đổi dữ liệu.
- HLDA [O]: Báo tín hiệu cho bên ngoài biết yêu cầu treo CPU để dùng các bus đã được chấp nhận và CPU 8086 đã treo các bus A, bus D và một số tín hiệu của bus C.
- SS0 [O]: Tín hiệu trạng thái. Tín hiệu này giống như S0 ở chế độ MAX và được dùng kết hợp với IO/M, DT/R để giải mã các chu kỳ hoạt động của bus.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

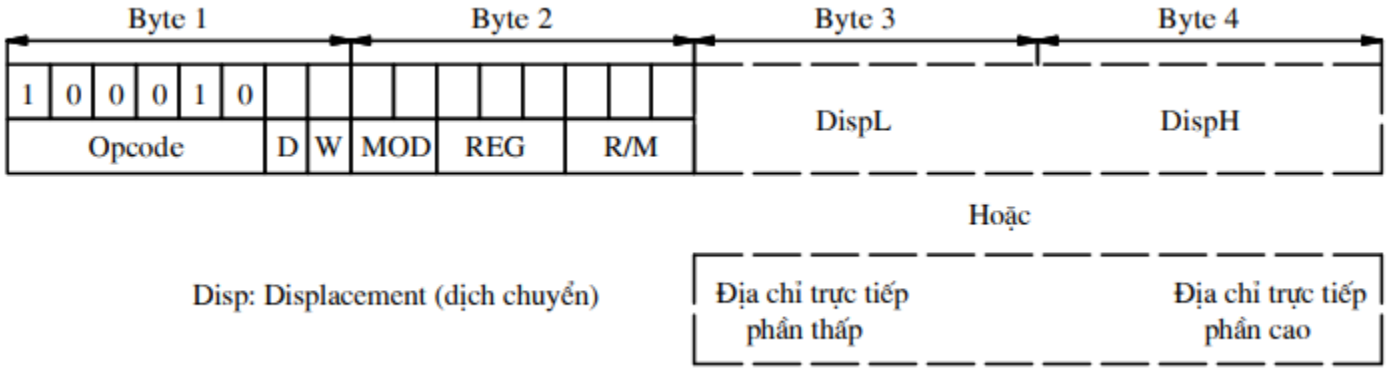
2.3. Tập lệnh

Định địa chỉ của bộ vi xư lý 8086

Lệnh của bộ vi xử lý được ghi bằng các ký tự dưới dạng gợi nhớ để người sử dụng dễ nhận biết. Đối với bản thân bộ vi xử lý thì lệnh cho nó được mã hoá dưới dạng các số 0 và 1 (còn gọi là mã máy) vì đó là dạng biểu diễn thông tin duy nhất mà máy có thể hiểu được. Vì lệnh cho bộ vi xử lý được cho dưới dạng mã nên sau khi nhận lệnh, bộ vi xử lý phải thực hiện giải mã lệnh rồi sau đó mới thực hiện lệnh Một lệnh có thể có độ dài một vài byte tuỳ theo bộ vi xử lý.

Đối với vi xử lý 8086 một lệnh có độ dài từ 1 đến 6 byte.

Ta sẽ dùng lệnh MOV để giải thích cách ghi lệnh nói chung của 8086.



Chương 2. Bộ Vi xử lý 8086/8088 của Intel

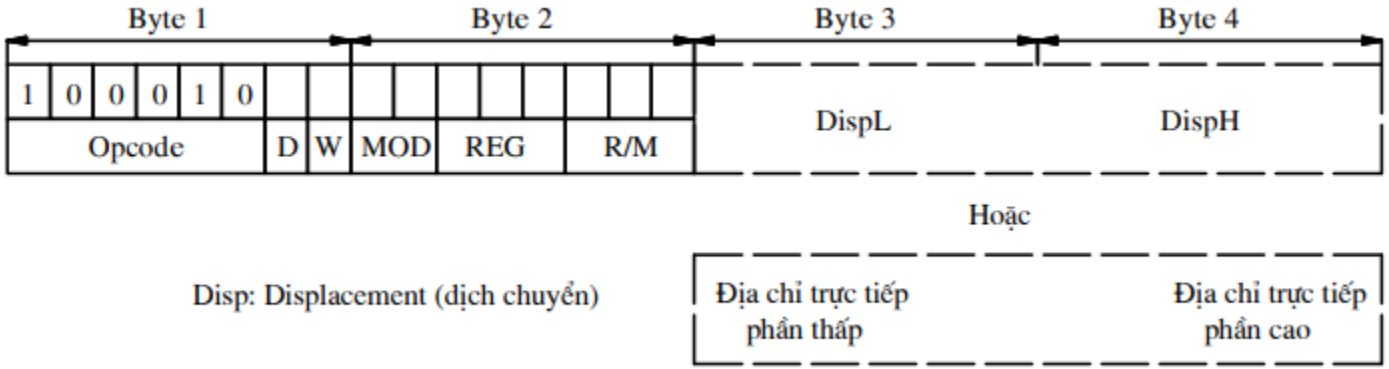
2.3. Tập lệnh

Định địa chỉ của bộ vi xư lý 8086

Lệnh của bộ vi xử lý được ghi bằng các ký tự dưới dạng gợi nhớ để người sử dụng dễ nhận biết. Đối với bản thân bộ vi xử lý thì lệnh cho nó được mã hoá dưới dạng các số 0 và 1 (còn gọi là mã máy) vì đó là dạng biểu diễn thông tin duy nhất mà máy có thể hiểu được. Vì lệnh cho bộ vi xử lý được cho dưới dạng mã nên sau khi nhận lệnh, bộ vi xử lý phải thực hiện giải mã lệnh rồi sau đó mới thực hiện lệnh Một lệnh có thể có độ dài một vài byte tuỳ theo bộ vi xử lý.

Đối với vi xử lý 8086 một lệnh có độ dài từ 1 đến 6 byte.

Ta sẽ dùng lệnh MOV để giải thích cách ghi lệnh nói chung của 8086.



Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Từ đây ta thấy để mã hoá lệnh MOV cần ít nhất 2 byte. Trong đó 6 bit đầu dùng để chứa mã lệnh, 6 bit này luôn là 100010. đối với các thanh ghi đoạn thì điều này lại khác. Bit W dùng để chỉ ra rằng một byte (W=0) hoặc một từ (W=1) sẽ được chuyền đi. Trong thao tác chuyển dữ liệu, một toán hạng luôn bắt buộc phải là thanh ghi. Bộ vi xử lý sử dụng 2 hoặc 3 bit (REG) để mã hoá các thanh ghi trong CPU như sau:

Bit D dùng để chỉ hướng đi của dữ liệu. D = 1 thì dữ liệu đến thanh ghi, D = 0 thì dữ liệu đi ra từ thanh ghi.

Thanh ghi		Mã
W = 1	W = 0	
AX	AL	000
BX	BL	011
CX	CL	001
DX	DL	010
SP	AH	100
DI	BH	111
BP	CH	101
SI	DH	110

Thanh ghi đoạn	Mã
CS	01
DS	11
ES	00
SS	10

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Hai bit MOD (chế độ) cùng với ba bit R/M (thanh ghi/bộ nhớ) tạo ra 5 bit dùng để chỉ ra chế độ địa chỉ cho các toán hạng của lệnh. Bảng sau cho ta thấy cách mã hoá các chế độ địa chỉ.

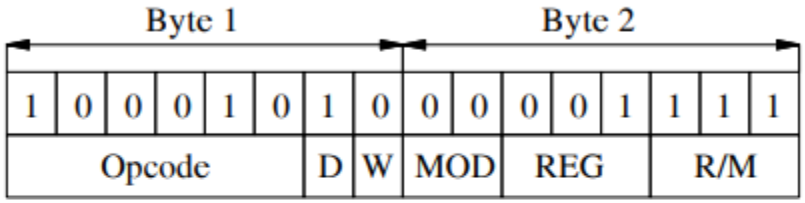
MOD / R/M	00	01	10	11	
				W=0	W=1
000	[BX+SI]	[BX+SI]+d8	[BX+SI]+d16	AL	AX
001	[BX+DI]	[BX+DI]+d8	[BX+DI]+d16	CL	CX
010	[BP+SI]	[BP+SI]+d8	[BP+SI]+d16	DL	DX
011	[BP+DI]	[BP+DI]+d8	[BP+DI]+d16	BL	BX
100	[SI]	[SI]+d8	[SI]+d16	AH	SP
101	[DI]	[DI]+d8	[DI]+d16	CH	BP
110	D16 (địa chỉ trực tiếp)	[BP]+d8	[BP]+d16	DH	SI
111	[BX]	[BX]+d8	[BX]+d16	BH	DI

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Hai bit MOD (chế độ) cùng với ba bit R/M (thanh ghi/bộ nhớ) tạo ra 5 bit dùng để chỉ ra chế độ địa chỉ cho các toán hạng của lệnh. Bảng sau cho ta thấy cách mã hoá các chế độ địa chỉ.

Ví dụ 1: MOV CL, [BX]



Mã lệnh MOV: 100010

D = 1: Chuyển tới thanh ghi

W = 0: Chuyển 1 byte

MOD: ở chế độ 00 và R/M là 111

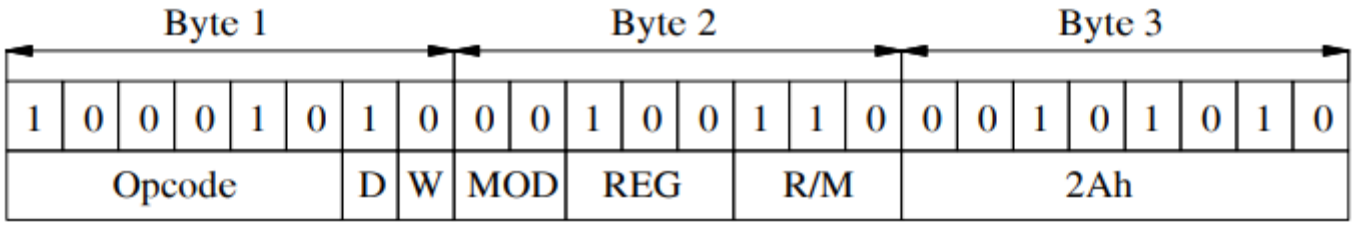
REG: 001 mã hoá CL

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Hai bit MOD (chế độ) cùng với ba bit R/M (thanh ghi/bộ nhớ) tạo ra 5 bit dùng để chỉ ra chế độ địa chỉ cho các toán hạng của lệnh. Bảng sau cho ta thấy cách mã hoá các chế độ địa chỉ.

Ví dụ 2: MOV AH, 2Ah



Mã lệnh MOV: 100010

D = 1: Chuyển tới thanh ghi

W = 0: Chuyển 1 byte

MOD: ở chế độ 00 và R/M là 110: Địa chỉ trực tiếp

REG: 100 mã hoá AH

2Ah = 00101010 dữ liệu cần chuyển tới AH

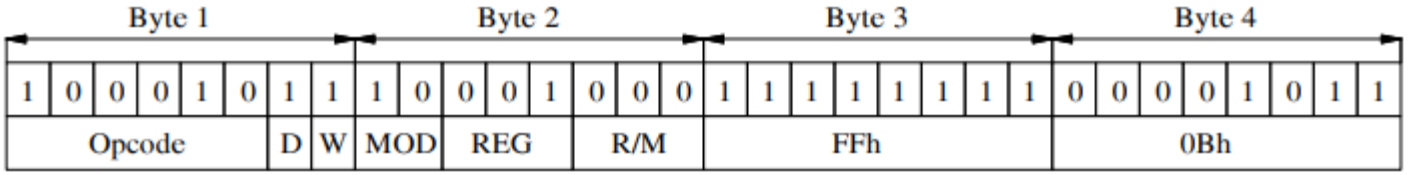
Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Ví dụ 3: MOV CX, [BX][SI]+DATA

DATA là một biến trong bộ nhớ, đó là địa chỉ lệch và là một hằng (ví dụ như 0BFF).

Lệnh này sẽ sử dụng 4 byte tổ chức như sau:



Mã lệnh MOV: 100010

D = 1: Chuyển tới thanh ghi

W =1: Chuyển 1 Word

MOD: ở chế độ 10 (offset 16 bit) và R/M là 000 (sử dụng thanh ghi cơ sở BX và thanh ghi chỉ số SI).

REG: 001 mã hoá thanh ghi CX.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ thanh ghi

Trong chế độ địa chỉ này người ta sử dụng các thanh ghi có sẵn trong CPU như là các toán hạng để chứa dữ liệu cần thao tác, vì vậy khi thực hiện có thể đạt tốc độ truy nhập cao hơn so với các lệnh truy nhập đến bộ nhớ.

Ví dụ:

MOV BX, DX ;copy noi dung DX vao BX

ADD AX, BX ;cong AX voi BX roi ghi ket qua vao AX

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ tức thì

Trong chế độ này toán hạng đích là một thanh ghi hay một ô nhớ, còn toán hạng nguồn là một hằng số. Ta có thể dùng chế độ địa chỉ này để nạp dữ liệu cần thao tác vào bất kỳ thanh ghi nào trừ (thanh ghi đoạn và thanh ghi cờ) và bất kỳ ô nhớ nào trong đoạn dữ liệu DS.

Ví dụ:

```
MOV CL, 100 ;chuyen 100 vao CL.
MOV AX, 0BC8h ;chuyen 0BC8h vao AX de roi
MOV DS, AX ;copy noi dung AX vao DS (vi khong duoc chuyen truc tiep vao thanh ghi doan)
MOV [BX], 20 ;chuyen 20 vao o nho tai ;dia chi DS:BX.
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ trực tiếp

Trong chế độ địa chỉ này một toán hạng chứa địa chỉ lệch của ô nhớ dùng chứa dữ liệu, còn toán hạng kia có thể là thanh ghi mà không được là ô nhớ.

Ví dụ:

MOV AL, [0243H] ;chuyen noi dung o nho DS:0243 vao AL
MOV [4320], CX ;chuyen noi dung CX vao hai o nho lien tiep DS:4320 va DS:4321

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ gián tiếp qua thanh ghi

Trong chế độ địa chỉ này một toán hạng là một thanh ghi được sử dụng để chứa địa chỉ lệch của ô nhớ dữ liệu, còn toán hạng kia chỉ có thể là thanh ghi mà không được là ô nhớ.

Ví dụ:

```
MOV AL, [BX] ;copy noi dung o nho co dia chi DS:BX
MOV [SI], CL ;copy noi dung CL vao o nho co dia chi DS:SI
MOV [DI], AX ;copy noi dung AX vao hai o nho lien tiep co dia chi DS:DI va DS:(DI+1)
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ tương đối cơ sở

Trong chế độ địa chỉ này các thanh ghi cơ sở như BX và BP và các hằng số biểu diễn các giá trị dịch chuyển được dùng để tính địa chỉ hiệu dụng của toán hạng trong các vùng nhớ DS và SS.

Ví dụ:

MOV CX, [BX]+10 ;copy noi dung hai o nho lien tiep co dia chi DS:BX+10 va DS:BX+11 vao CX

MOV CX, [BX+10] ;cach viet khac cua lenh tren

MOV CX, 10+[BX];cach viet khac cua lenh tren

MOV AL, [BP]+5 ;chuyen noi dung o nho co dia chi SS:BP+5 vao AL Quan sát trên ta thấy:10 và 5 là các dịch chuyển của các toán hạng tương ứng.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ tương đối chỉ số

Trong chế độ địa chỉ này các thanh ghi chỉ số như SI và DI và các hằng số biểu diễn các giá trị dịch chuyển được dùng để tính địa chỉ hiệu dụng của toán hạng trong các vùng nhớ DS.

Ví dụ

MOV CX, [SI]+10 ;copy noi dung hai o nho ;lien tiep co dia chi DS:SI+10 va DS:SI+11vao CX

MOV CX, [SI +10];cach viet khac cua lenh tren

MOV CX, 10+[SI];cach viet khac cua lenh tren

MOV AL, [DI]+5 ;chuyen noi dung o nho co dia chi DS:DI+5 vao AL

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ tương đối chỉ số cơ sở

Kết hợp hai chế độ địa chỉ chỉ số và cơ sở ta có chế độ địa chỉ chỉ số cơ sở. Trong chế độ này ta dùng cả hai thanh ghi cơ sở lẫn thanh ghi chỉ số để tính địa chỉ của toán hạng. Nếu ta dùng thêm cả thành phần biểu diễn sự dịch chuyển của địa chỉ thì ta có chế độ địa chỉ tổng hợp nhất: Chế độ địa chỉ tương đối chỉ số cơ sở.

Ví dụ:

```
MOV BX, [BX][SI]+10 ;chuyen noi dung ;hai o nho lien tiep co dia chi
                                     DS:BX+SI+10 va DS:BX+SI+11 vao CX
MOV AL, [BP+DI+5] ;chuyen noi dung o ;nho co dia chi DS:BP+DI+5 vao AL
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ chuỗi (string) – mảng

Một chuỗi (string) là một dãy các byte hoặc word liên tiếp trong bộ nhớ. Các lệnh thao tác với chuỗi không sử dụng bất kỳ một chế độ địa chỉ nào ở trên. Một chuỗi có thể có độ dài tối đa lên tới 64K-bytes (một segments). Chế độ địa chỉ chuỗi sử dụng các thanh ghi SI, DI, DS và ES. Với tất cả các lệnh thao tác chuỗi đều sử dụng SI để trỏ vào byte đầu tiên của chuỗi nguồn và DI trỏ vào byte đầu tiên của chuỗi đích.

Ví dụ: Giả sử: DS=1000h, ES=2000h, SI=10h, DI=20h)
MOVSB ;Sao chép chuỗi từ 10010h đến 20020

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Chế độ địa chỉ cổng (port)

Trong họ vi xử lý 80x86 của Intel có không gian địa chỉ cho bộ nhớ và cổng vào/ra là tách biệt nhau. Không gian địa chỉ cổng có thể lên đến 65536 cổng (64K-ports). Địa chỉ của một cổng có thể được xác định bởi một hằng giá trị kiểu byte (phạm vi = 0..255)

Ví dụ:

IN AL, 40h ;Đọc cổng – sao chép nội dung tại ;cổng có địa chỉ 40h vào thanh ghi AL

OUT 80h, AL ;Ghi cổng – gửi dữ liệu trong ;thanh ghi AL tới cổng có địa chỉ 80h. Địa chỉ của cổng cũng có thể được xác định gián tiếp qua thanh ghi (Khi này phạm vi tối đã sẽ là 65536 cổng).

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Tập lệnh của vi xử lý 8086

- Nhóm các lệnh vận chuyển (sao chép) dữ liệu.
- Nhóm các lệnh tính toán số học.
- Nhóm các lệnh tính toán logic.
- Nhóm các lệnh dịch, quay toán hạng.
- Nhóm các lệnh nhảy (rẽ nhánh).
- Nhóm các lệnh lặp.
- Nhóm các lệnh điều khiển, đặc biệt khác.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh vận chuyển (sao chép) dữ liệu.

LDS – Load register and DS with words from memory (nạp một từ (từ bộ nhớ) vào thanh ghi cho trong lệnh và một từ tiếp theo vào DS).

Dạng lệnh:

LDS Đích, Nguồn

Trong đó:

- Đích là một trong các thanh ghi: AX, BX, CX, DX, SP, BP, SI, DI.
- Nguồn là ô nhớ trong đoạn DS được chỉ ra trong lệnh.

Ví dụ:

LDS SI, STR_PTR

Nạp vào thanh ghi SI nội dung 2 ô nhớ STR_PTR và STR_PTR+1 và nạp vào DS nội dung 2 ô nhớ STR_PTR+3 và STR_PTR+4. các ô nhớ này đều nằm trong đoạn dữ liệu DS và chứa địa chỉ của chuỗi Nguồn. Do vậy sau đó DS:SI chỉ vào đầu chuỗi Nguồn cần thao tác

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh vận chuyển (sao chép) dữ liệu.

LEA – Load Effective Address (nạp địa chỉ hiệu dụng vào thanh ghi).

Dạng lệnh:

LEA Đích, Nguồn

Trong đó:

- Đích là một trong các thanh ghi: BX, CX, DX, BP, SI, DI.
- Nguồn là tên biến trong đoạn DS được chỉ rõ trong lệnh hoặc ô nhớ cụ thể.

Ví dụ:

LEA DX, Label ;nap dia chi lech cua Label vao DX

LEA CX, [BX][DI] ;nap vao CX dia chi hieu dung do BX va DI chi ra EA=BX+DI

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh vận chuyển (sao chép) dữ liệu.

LES – Load register and ES with words from memory (nạp một từ (từ bộ nhớ) vào thanh ghi cho trong lệnh và một từ tiếp theo vào ES).

Dạng lệnh:

LES Đích, Nguồn

Trong đó:

- Đích là một trong các thanh ghi: AX, BX, CX, DX, SP, BP, SI, DI.
- Nguồn là ô nhớ trong đoạn DS được chỉ ra trong lệnh.

Ví dụ: LDS DI, [BX]

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh vận chuyển (sao chép) dữ liệu.

MOV – Mov a byte or word (chuyển một byte hay từ)

Dạng lệnh:

MOV Đích, Nguồn

Mô tả: Đích←Nguồn Trong đó toán hạng đích và Nguồn có thể tìm được theo các chế độ địa chỉ khác nhau, nhưng phải có cùng độ dài và không được phép đồng thời là hai ô nhớ hoặc hai thanh ghi đoạn.

Ví dụ:

```
MOV AL, AH     ;AL←AH
MOV CX, 50     ;CX←50
MOV DL, [SI]   ;DL←{DS:SI}
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh vận chuyển (sao chép) dữ liệu.

MOVS/MOVSB/MOVSW – Move String byte or String word

Dạng lệnh:

MOVS Chuỗi_đích, Chuỗi_Nguồn

MOVSB

MOVSW

Mô tả: Phần tử chuỗi_đích←phần tử chuỗi_Nguồn

OUT – Output a byte or a work to a port.

Dạng lệnh: s

OUT Port, Acc

Mô tả: Acc→{Port}

Ví dụ:

```
OUT 45H, AL ;dua du lieu tu AL ra cong 45H
MOV DX, 0 ;xoa DX
MOV DX, 00FFH ;nap dia chi cong vao DX
OUT DX, AX ;dua du lieu tu AX ra 00FFH
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh vận chuyển (sao chép) dữ liệu.

POP – Pop word from top of Stack (lấy lại 1 từ vào thanh ghi từ đỉnh ngăn xếp)
Dạng lệnh: *POP Đích*

POPF – Pop word from top of Stack to Flag register (lấy 1 từ vào thanh ghi cờ từ đỉnh ngăn xếp).
Dạng lệnh: *POPF*

PUSH – Push word on the Stack (cất 1 từ vào ngăn xếp)
Dạng lệnh: *PUSH Nguồn*

PUSHF – Push Flag register to the Stack (cất thanh ghi cờ vào ngăn xếp)
Dạng lệnh: *PUSHF*

XCHG – Exchange (hoán đổi nội dung hai toán hạng)
Dạng lệnh: *XCHG Đích, Nguồn*

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

ADC – Add with Carry (cộng có nhớ)

Dạng lệnh:

ADC Đích, Nguồn

Mô tả: Đích ← Đích + Nguồn + CF

Cộng hai toán hạng Đích và Nguồn với cờ CF kết quả lưu vào Đích. Các cờ bị thay đổi: AF, CF, OF, PF, SF, ZF.

Ví dụ:

ADC AL, 74H ;AL←AL+74+CF

ADC CL, BL ;CL←CL+BL+CF

ADC DL, [SI] ;DL←DL+(DS:SI)+CF

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

ADD – Add (cộng hai toán hạng)

Dạng lệnh:

ADD Đích, Nguồn

Mô tả: Đích ← Đích + Nguồn Cộng hai toán hạng đích và Nguồn kết quả lưu vào đích. Các cờ bị thay đổi: AF, CF, OF, PF, SF, ZF.

Ví dụ:

ADD DX, CX ;DX←DX+CX

ADD AX, 400 ;AX←AX+400

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

SUB – Substract (trừ hai toán hạng)

Dạng lệnh:

SUB Đích, Nguồn

Mô tả: Đích←Đích - Nguồn Toán hạng đích vào Nguồn phải chứa cùng một loại dữ liệu và không được đồng thời là hai ô nhớ, cũng không được là thanh ghi đoạn. Các cờ bị thay đổi: AF, CF, OF, PF, SF, ZF.

Ví dụ:

```
SUB AL, 78H      ;AL←AL-78H
SUB BL, CL       ;BL←BL-CL
SUB DL, [SI]     ;DL←DL-{DS:SI}
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

SBB – Substract with Borrow (trừ có mượn).

Dạng lệnh:

SBB Đích, Nguồn

Mô tả: Đích←Đích-Nguồn-CF Toán hạng đích vào Nguồn phải chứa cùng một loại dữ liệu và không được đồng thời là hai ô nhớ, cũng không được là thanh ghi đoạn. Các cờ bị thay đổi: AF, CF, OF, PF, SF, ZF.

Ví dụ:

SBB AL, 78H ;AL←AL-78H-CF

SBB BL, CL ;BL←BL-CL-CF

SBB DL, [SI] ;DL←DL-{DS:SI}-CF

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

DEC – Decrement (giảm byte hay word đi một giá trị)

Dạng lệnh:

DEC Đích

DEC trừ toán hạng Đích đi 1. Toán hạng Đích có thể là byte hay word. Các cờ bị thay đổi: AF, OF, PF, SF, ZF.

Ví dụ:

MOV BX, 1200H ;chuyen 1200H vao BX

DEC BX ;BX=11FFH

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

MUL – Multiply unsigned byte or word (nhân số không dấu)

Dạng lệnh:

MUL Nguồn

Thực hiện phép nhân không dấu với toán hạng Nguồn (ô nhớ hoặc thanh ghi) với thanh ghi tổng.

- Nếu Nguồn là số 8 bit: AL*Nguồn. Số bị nhân phải là số 8 bit đặt trong AL, sau khi nhân tích lưu vào AX
- Nếu Nguồn là số 16 bit: AX*Nguồn. Số bị nhân phải là số 16 bit đặt trong AX, sau khi nhân tích lưu vào DXAX. Nếu byte cao (hoặc 16 bit cao) của 16 (hoặc 32) bit kết quả chứa 0 thì CF=OF=0. Các cờ bị thay đổi: CF, OF.

Ví dụ:

```
MUL CX          ;AXxCX → DXAX
MUL BL          ;ALxBL → AX
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

DIV – Division (chia không dấu)

Dạng lệnh:

DIV Nguồn

Toán hạng Nguồn là số chia. Tuỳ theo độ dài toán hạng Nguồn ta có hai trường hợp bố trí phép chia.

- Nếu Nguồn là là số 8 bit: AX/Nguồn, thương để vào AL, số dư để vào AH
- Nếu Nguồn là số 16 bit: DXAX/Nguồn, thương để vào AX, số dư để vào DX

Nếu thương không phải là số nguyên nó được làm tròn theo số nguyên sát dưới. Nếu Nguồn bằng 0 hoặc thương thu được lớn hơn FFH hoặc FFFFH (tuỳ theo độ dài của toán hạng Nguồn) thì 8086 thực hiện lệnh ngắt INT 0.

Ví dụ:

- MOV AX, 0033H ;chuyen 0033H vao AX
- MOV BL, 25 DIV BL ;AL=02H va AH=01H

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

IDIV – Integer Division (chia có dấu)

Dạng lệnh:

IDIV Nguồn

Thực hiện một phép chia có dấu thanh ghi tổng (và phần mở rộng của nó) cho toán hạng Nguồn.

- Sau phép chia AL (AX) chứa thương số (số có dấu), AH (DX) chứa số dư (số có dấu).
- Dấu của số dư trùng với dấu của số bị chia.
- Nếu Nguồn = 0 hoặc thương nằm ngoài dải -128…+127 hoặc - 32768…32767 (tuỳ theo độ dài Nguồn) thì 8086 thực hiện lệnh ngắt INT 0. Các cờ bị thay đổi: không.

Ví dụ:

IDIV CL

IDIV BX

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

IMUL – Integer Multiplication (nhân có dấu)

Dạng lệnh:

IMUL Nguồn

Tuỳ theo độ dài toán hạng Nguồn ta có 2 trường hợp bố trí phép nhân, chỗ để ngầm định cho số bị nhân và kết quả.

- Nếu Nguồn là số có dấu 8 bit: ALxNguồn. Sau khi nhân AX←tích.
- Nếu Nguồn là số có dấu 16 bit: AXxNguồn. Sau khi nhân DXAX←tích.

Ví dụ:

IMUL CL

IN – Input data from a port (đọc dữ liệu từ cổng vào thanh ghi Acc).

Dạng lệnh:

IN Acc, địa_chỉ_cổng

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

IN – Input data from a port (đọc dữ liệu từ cổng vào thanh ghi Acc).

Dạng lệnh:

IN Acc, địa_chỉ_cổng

Ví dụ:

IN AL, 45H ;doc mot byte tu mot cong duoc xac dinh trong che do tuc thi IN AX, 0046H ;doc hai byte tu mot cong duoc xac dinh trong che do tuc thi IN AX, DX ;doc mot tu tu mot cong dang bien

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

INC – Increment (tăng toán hạng lên 1)

Dạng lệnh:

INC đích

Mô tả: Đích ← Đích + 1 Lệnh này tăng đích lên 1, tương đương với việc ADD đích, 1 nhưng chạy nhanh hơn. Các cờ bị thay đổi: AF, OF, PF, SF, ZF.

Ví dụ:

INC AL

INC BX

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh tính toán số học.

NEG – Negation (lấy bù hai của một toán hạng, đảo dấu của một toán hạng)

Dạng lệnh:

NEG Đích

Mô tả: Đích←0-Đích NEG lấy 0 trừ cho đích (có thể là 1 byte hoặc 1 từ) và trả lại kết quả cho toán hạng đích, nếu ta lấy bù hai của -128 hoặc -32768 ta sẽ được kết quả không đổi nhưng OF=1 để báo là kết quả bị tràn vì số dương lớn nhất biểu diễn được là +127 và +32767. Các cờ bị thay đổi: AF, CF, OF, PF, SF, ZF

Ví dụ:

```
NEG AL                    ;AL←0-(AL)
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh Logic.

AND (phép và logic)

Dạng lệnh:

AND Đích, Nguồn

Mô tả: Đích ← Đích ^ Nguồn Thực hiện phép và logic hai toán hạng và lưu kết quả vào toán hạng đích. Người ta thường sử dụng để che đi/giữ lại một vài bit nào đó của một toán hạng bằng cách nhân logic toán hạng đó với toán hạng tức thì có các bit 0/1 ở các vị trí cần che đi/giữ lại tương ứng. Các cờ bị thay đổi: CF, OF, PF, SF, ZF.

Ví dụ:

```
AND DX, CX ;DX←DX AND CX theo tung bit
AND AL, 0FH ;che 4 bit cao cua AL
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh Logic.

NOT – Logical Negation (phủ định logic)

Dạng lệnh:

NOT Đích

NOT đảo các giá trị của các bit của toán hạng đích. Các cờ bị thay đổi: không.

Ví dụ:

```
MOV AL, 02H     ;AL=(0000 0010)B
NOT AL          ;AL=(1111 1101)B
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh Logic.

OR – Logic OR (phép hoặc logic)

Dạng lệnh:

OR Đích, Nguồn

Mô tả: Đích = Đích ∨ Nguồn Toán hạng Đích và Nguồn phải chứa dữ liệu cùng độ dài và không được phép đồng thời là hai ô nhớ và cũng không được là thanh ghi đoạn. Phép OR thường dùng để lập một vài bit nào đó của toán hạng bằng cách cộng logic toán hạng đó với các toán hạng tức thời có các bit 1 tại vị trí tương ứng cần thiết lập. Các cờ bị thay đổi: CF, OF, PF, SF, ZF.

Ví dụ:

```
OR AX, BX       ;AX←AX∨BX theo tung bit
OR CL, 30H      ;lap bit b4 va b5 cua CL len
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

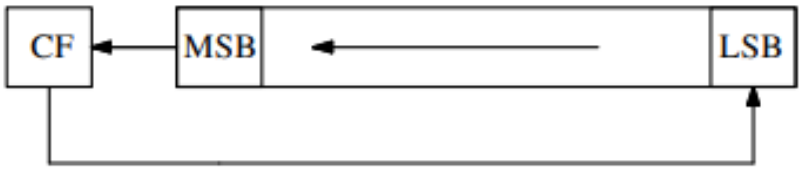
Nhóm các lệnh dịch, quay toán hạng.

RCL – Rotate though CF to the Left (quay trái thông qua cờ nhớ)

Dạng lệnh:

RCL Đích, CL

Mô tả:



Ví dụ:

MOV CL, 3 ;so lan quay la 3

RCL AL, CL

Trước khi thực hiện lệnh: AL = 01011110, CF = 0.

Sau khi thực hiện lệnh: AL = 11110001, CF = 0.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh dịch, quay toán hạng.

RCR – Rotate though CF to the Right (quay phải thông qua cờ nhớ)

Dạng lệnh:

RCR Đích, CL

Ví dụ:

MOV CL, 2 ;so lan quay la 2

RCR AL, CL

Trước khi thực hiện lệnh: AL = 11000010, CF = 1.

Sau khi thực hiện lệnh: AL = 01110000, CF = 1

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

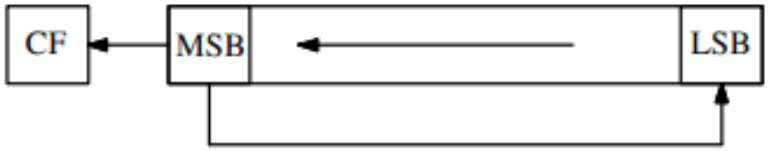
Nhóm các lệnh dịch, quay toán hạng.

ROL – Rotate all bit to the Left (quay vòng sang trái)

Dạng lệnh:

ROL Đích, CL.

Mô tả:



Ví dụ:

MOV CL, 2 ;so lan quay la 2

ROL AL, CL

Trước khi thực hiện lệnh: AL = 11001100, CF = 1

Sau khi thực hiện lệnh: AL = 00110011, CF = 1

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh dịch, quay toán hạng.

ROR – Rotate all bit to the Left (quay vòng sang phải).

Dạng lệnh: *ROR Đích, CL*

SAL/SHL - Shift Arithmetically Left (dịch trái số học)/Shift Logically Left (dịch trái logic).

Dạng lệnh: *SAL Đích, CL*

SHL Đích, CL

SAR - Shift Arithmetically Right (dịch phải số học).

Dạng lệnh: *SAR Đích, CL*

SHR – Shift logically Right (dịch phải logic)

Dạng lệnh: *SHR Đích, CL*

TEST – Logic Camparison (lệnh so sánh logic)

Dạng lệnh: *TEST Đích, Nguồn*

XOR – Exclusive OR (lệnh logic XOR (hoặc đảo)).

Dạng lệnh: *XOR Đích, Nguồn*

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh so sánh

CMP – Compare (so sánh)

Dạng lệnh:

CMP đích, Nguồn

CMP trừ toán hạng đích cho toán hạng Nguồn, chúng có thể là các byte hoặc các từ, nhưng không lưu trữ kết quả. Các toán hạng không bị thay đổi. Kết quả của lệnh này dùng để cập nhật các cờ và có thể được dùng để làm điều kiện cho các lệnh nhảy có điều kiện tiếp theo.

CMPS/CMPSB/CMPSW – Compare String Bytes or String Words (so sánh hai chuỗi byte hay hai chuỗi từ).

Dạng lệnh:

CMPS Chuỗi_đích, Chuỗi_Nguồn

CMPSB

CMPSW

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh nhảy (rẽ nhánh)

JA/JNBE – Jump if Above/Jump if Not Below or Equal (nhảy nếu cao hơn/nhảy nếu không thấp hơn hoặc bằng).

Dạng lệnh:

JA NHAN

JNBE NHAN

Mô tả: IP←IP+dịch chuyển Hai lệnh trên biểu diễn cùng một thao tác nhảy có điều kiện tới NHAN nếu CF + ZF = 0. Quan hệ cao hơn/thấp là quan hệ dành cho việc so sánh (do lệnh CMP thực hiện) độ lớn hai số không dấu. NHAN phải nằm cách xa một khoảng -128…+127 byte so với lệnh tiếp theo sau lệnh JA/JNBE. Chương trình sẽ căn cứ vào vị trí NHAN để xác định giá trị dịch chuyển.

Ví dụ:

```
CMP AX, 12ABH          ;so sanh AX voi 12ABH
JA THOI                ;nhay den THOI neu AX cao hon 12ABH
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh nhảy (rẽ nhánh)

JB/JC/JNAE – Jump if Below/Jump if Carry/Jump if Not Above or Equal (nhảy nếu thấp hơn/nhảy nếu có nhớ/nhảy nếu không cao hơn hoặc bằng).

Dạng lệnh:

JB NHAN
JC NHAN
JNAE NHAN

Mô tả: IP←IP+dịch chuyển Ba lệnh trên biểu diễn cùng một thao tác nhảy có điều kiện tới NHAN nếu CF = 1. Quan hệ cao hơn/thấp là quan hệ dành cho việc so sánh (do lệnh CMP thực hiện) độ lớn hai số không dấu. NHAN phải nằm cách xa một khoảng -128…+127 byte so với lệnh tiếp theo sau lệnh JB/JC/JNAE. Chương trình sẽ căn cứ vào vị trí NHAN để xác định giá trị dịch chuyển. Các cờ bị thay đổi: không.

Ví dụ:

```
CMP AL, 10H     ;so sanh AL voi 10H
JB THOI         ;nhay den THOI neu AL thap  hon 10H
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh nhảy (rẽ nhánh)

JBE/JNA – Jump if Below or Equal/Jump if Not Above (nhảy nếu thấp hơn hoặc bằng/nhảy nếu không cao hơn).

Dạng lệnh: *JBE NHAN*

JNA NHAN

JCXZ – Jump if CX register is Zero (nhảy nếu nội dung thanh đếm CX rỗng).

Dạng lệnh:

JCXZ NHAN

JE/JZ – Jump if Equal/Jump if Zero (nhảy nếu bằng nhau/nhảy nếu kết quả bằng không)

Dạng lệnh:

JE NHAN

JZ NHAN

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh nhảy (rẽ nhánh)

JG/JNLE – Jump if Greater than/Jump if Not Less than or Equal (nhảy nếu lớn hơn/nhảy nếu không bé hơn hoặc bằng)

Dạng lệnh:

JG NHAN

JNLE NHAN

JGE/JNL – Jump if Greater than or Equal/Jump if Not Less than (nhảy nếu lớn hơn hoặc bằng/nhảy nếu không nhỏ hơn)

Dạng lệnh:

JGE NHAN

JNL NHAN

JL/JNGE – Jump if Less than/Jump if Not Greater than or Equal (nhảy nếu bé hơn/nhảy nếu không lớn hơn hoặc bằng).

Dạng lệnh:

JL NHAN

JNGE NHAN

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh nhảy (rẽ nhánh)

JLE/JNG – Jump if Less than or Equal/Jump if Not Greater than (nhảy nếu nhỏ hơn hoặc bằng/nhảy nếu không lớn hơn)

Dạng lệnh:

JLE NHAN

JNG NHAN

JMP – Unconditional Jump (lệnh nhảy không điều kiện). JMP trao quyền điều khiển cho vùng mục tiêu một cách không điều kiện. Lệnh này có các chế độ giống như lệnh CALL và nó cũng phân biệt nhảy gần, nhảy xa.

Dạng lệnh: Sau đây là những cách viết lệnh không điều kiện.

JMP NHAN

JNE/JNZ – Jump if Not Equal/Jump if Not Zero (nhảy nếu không bằng nhau/nhảy nếu kết quả không rỗng).

Dạng lệnh:

JNE NHAN

JNZ NHAN

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh nhảy (rẽ nhánh)

JNO – Jump if Not Overflow (nhảy nếu không tràn)

Dạng lệnh: *JNO NHAN*

JNP/JPO – Jump if Not Parity/Jump if Parity Odd (nhảy nếu parity lẻ).

Dạng lệnh: *JNP NHAN JPO NHAN*

JNS - Jump Not Signed (nhảy nếu kết quả dương).

Dạng lệnh: *NS NHAN*

JO – Jump if Overflow (nhảy nếu tràn) D

ạng lệnh: *JO NHAN*

JP/JPE – Jump if Parity/Jump if Parity Even (nhảy nếu parity chẵn)

Dạng lệnh: *JP NHAN*

JPE NHAN

JS – Jump if Sign (nhảy nếu âm)

Dạng lệnh: *JS NHAN*

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh lặp

LOOP – Loop if CX is not 0 (lặp nếu CX ≠ 0)

Dạng lệnh: *LOOP NHAN*

Mô tả: Lệnh này dùng để lặp lại đoạn chương trình (gồm các lệnh nằm trong khoảng từ NHAN đến hết lệnh LOOP NHAN) cho đến khi số lần lặp CX=0. Điều này có nghĩa là trước khi vào vòng lặp ta phải đưa số lần lặp mong muốn vào CX, và sau mỗi lần lặp thì CX tự động giảm đi 1. NHAN phải nằm cách xa (dịch đi một khoảng) tối đa -128 byte so với lệnh tiếp theo sau lệnh LOOP.

Các cờ bị thay đổi: không.

Ví dụ:

```
MOV AL, 0       ;xoa AL
MOV CX, 10      ;nap so lan lap vao CX
LAP:INC AL      ;tang AL len 1
LOOP LAP        ;lap lai 10 lan, AL=10
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh lặp

LOOPE/LOOPZ – Loop while CX=0 or ZF=0 (lặp lại đoạn chương trình cho đến khi CX=0 hoặc ZF=0).

Dạng lệnh: *LOOPE NHAN*
LOOPZ NHAN

Mô tả: Lệnh này dùng để lặp lại đoạn chương trình (gồm các lệnh nằm trong khoảng từ NHAN đến hết lệnh LOOPE NHAN hoặc LOOPZ NHAN) cho đến khi số lần lặp CX=0 hoặc cờ ZF=0. Điều này có nghĩa là trước khi vào vòng lặp ta phải đưa số lần lặp mong muốn vào CX, và sau mỗi lần lặp thì CX tự động giảm đi 1. NHAN phải nằm cách xa (dịch đi một khoảng) tối đa -128 byte so với lệnh tiếp theo sau lệnh LOOPE/LOOPZ. Các cờ bị thay đổi: không.

Ví dụ:
```
MOV AL, AH     ;AL=AH
MOV CX, 50     ;nap so lan lap vao CX
LAP:INC AL     ;tang AL
COMP AL, 16    ;so sanh AL voi 16
LOOPE LAP      ;lap lai cho den khi AL≠16 hoac CX=0
```

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh lặp

LOOPNE/LOOPNZ – Loop while CX=0 or ZF=1 (lặp lại đoạn chương trình cho đến khi CX=0 hoặc ZF=1).

Dạng lệnh:

LOOPNE NHAN
LOOPNZ NHAN

Mô tả: Lệnh này dùng để lặp lại đoạn chương trình (gồm các lệnh nằm trong khoảng từ NHAN đến hết lệnh LOOPNE NHAN hoặc LOOPNZ NHAN) cho đến khi số lần lặp CX=0 hoặc cờ ZF=1. Điều này có nghĩa là trước khi vào vòng lặp ta phải đưa số lần lặp mong muốn vào CX, và sau mỗi lần lặp thì CX tự động giảm đi 1.

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh lặp

REP – Repeat String Instruction until CX=0 (lặp lại lệnh viết sau đó cho tới khi CX=0). Đây là tiếp đầu ngữ dùng để viết trước các lệnh thao tác với chuỗi đữ liệu mà ta muốn lặp lại một số lần. Số lần lặp phải để trước trong CX. Khi các lệnh này được thực hiện thì CX tự động giảm đi 1. Quá trình lặp kết thúc khi CX=0.

REPE/REPZ – Repeat String Instruction until CX=0 or ZF=0 (lặp lại lệnh viết sau đó cho tới khi CX=0 hoặc ZF=0).

REPNE/REPNZ – Repeat String Instruction until CX=0 or ZF=1 (lặp lại lệnh viết sau đó cho tới khi CX=0 hoặc ZF=1)

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh điều khiển, đặc biệt khác

CALL – Call a procedure (gọi chương trình con)

Dạng lệnh:

CALL Thủ_tục

Mô tả: Lệnh này dùng để chuyển hoạt động của vi xử lý từ chương trình chính (CTC) sang chương trình con (ctc). Nếu ctc nằm trong cùng một đoạn mã với CTC ta có gọi gần (near call). Nếu ctc và CTC nằm ở hai đoạn mã khác nhau ta có gọi xa (far call).

CLC – Clear the Carry flag (xoá cờ nhớ)

Dạng lệnh:

CLC

Xoá cờ nhớ CF và không làm ảnh hưởng đến các cờ khác.

Các cờ bị thay đổi: CF

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh điều khiển, đặc biệt khác

CLD – Clear the Direction flag (xoá cờ hướng)

Dạng lệnh: *CLD*

Xoá cờ hướng DF và không làm ảnh hưởng đến các cờ khác. Các cờ bị thay đổi: DF.

CLI – Clear the Interrupt flag (xoá cờ ngắt)

Dạng lệnh: *CLD*

Xoá cờ ngắt IF và không làm ảnh hưởng đến các cờ khác. Các yêu cầu ngắt che được sẽ bị che Các cờ bị thay đổi: IF.

CMC – Complement the Carry flag (đảo cờ nhớ).

Dạng lệnh: *CMC*

Mô tả: CF = CF. Đảo cờ nhớ CF Các cờ bị thay đổi: CF

HLT – Halt processing (dừng)

Dạng lệnh: *HLT*

INT – Interrupt (lệnh gọi ngắt)

Dạng lệnh: *INT N (N=0…FFH)*

Chương 2. Bộ Vi xử lý 8086/8088 của Intel

2.3. Tập lệnh

Nhóm các lệnh điều khiển, đặc biệt khác

IRET – Interrupt Return (trở về CTC từ ctc phục vụ ngắt)

Dạng lệnh: *IRET*

NOP – No Operation (CPU không làm gì)

Dạng lệnh: *NOP*

RET – Return from Procedure to Calling Program (trở về chương trình chính từ chương trình con).

Dạng lệnh: *RET hoặc RET N*

STC – Set the Carry Flag (lập cờ nhớ)

Dạng lệnh: *STC*

STD – Set the Direction Flag (lập cờ hướng).

Dạng lệnh: *STD*

STI – Set the Interrupt Flag (lập cờ cho phép ngắt)

Dạng lệnh: *STI*

WAIT – Wait for TEST or INTR Signal

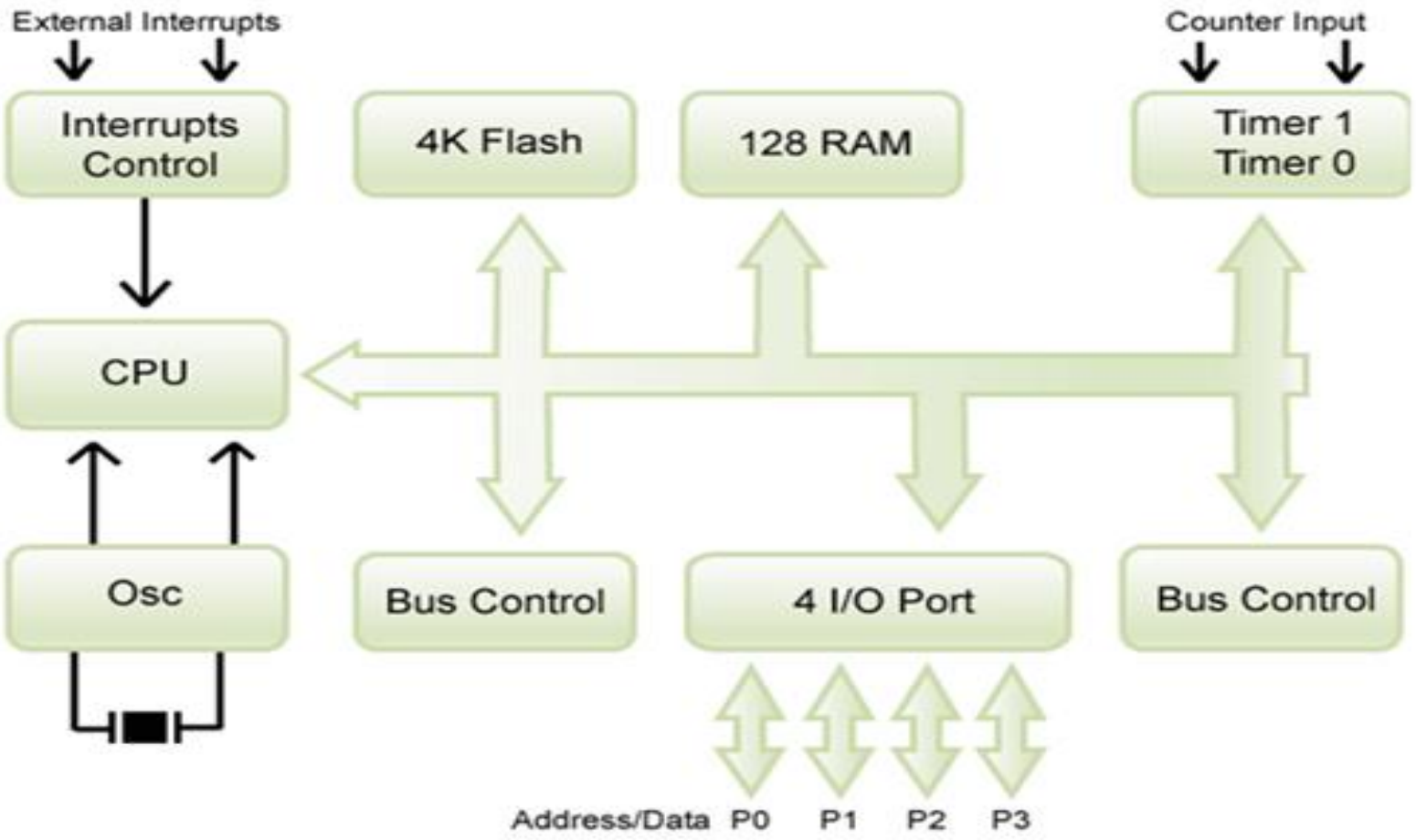
Dạng lệnh: *WAIT*

Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển
3.2. Tổ chức bộ nhớ
3.3. Tập lệnh vi điều khiển 8051

Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển



Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển

Các IC của họ MCS-51™ có các đặc trưng chung như sau:

- 4 port I/O x 8 bit
- 1 giao tiếp nối tiếp
- 4k không gian bộ nhớ chương trình mở rộng
- 128byte không gian bộ nhớ dữ liệu mở rộng
- Một bộ xử lý luận lý (thao tác trên các bit đơn)
- 210 bit được địa chỉ hóa
- Bộ nhân/chia 4 µs.

Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển

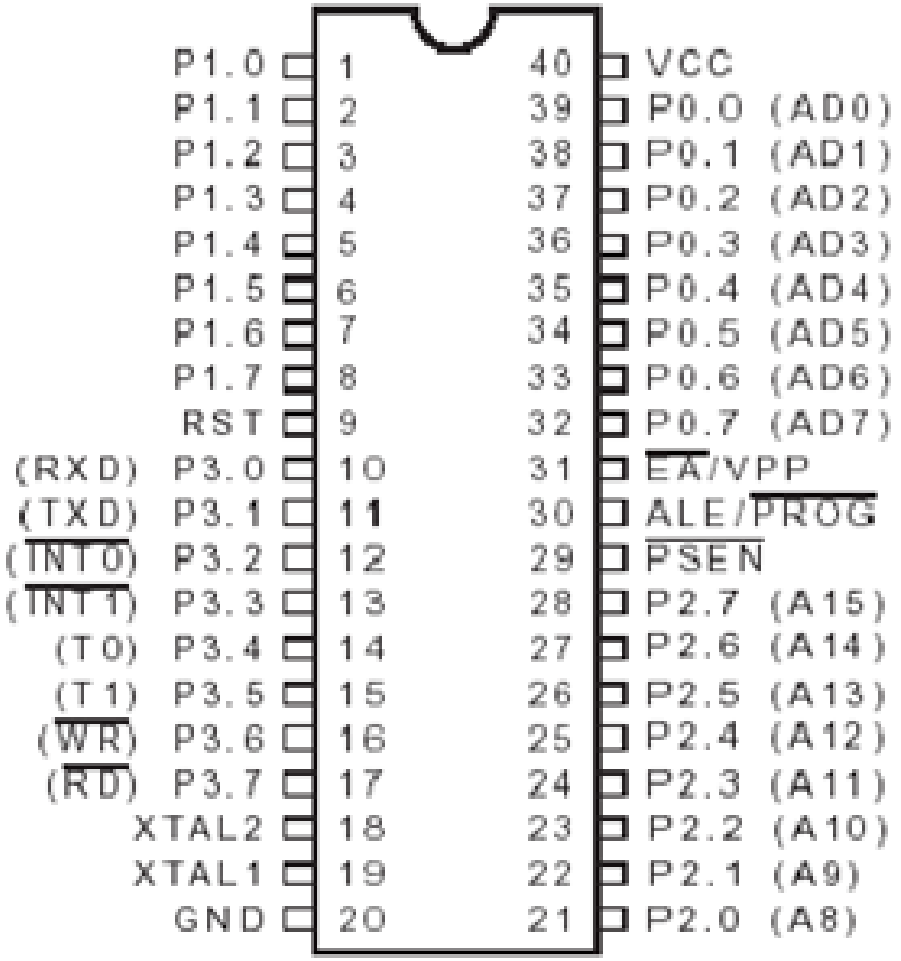
Các IC của họ MCS-51™ có các đặc trưng chung như sau:

Dòng SP	ROM	RAM	Số bộ định thời
8051	4K ROM	128 byte	2
8751	4K EPROM	128 byte	2
8951	4K FLASH	128 byte	2
8032	0K	256 byte	3
8052	8K ROM	256 byte	3
8752	8K EPROM	256 byte	3
8952	8K FLASH	256 byte	3

Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển 89x51

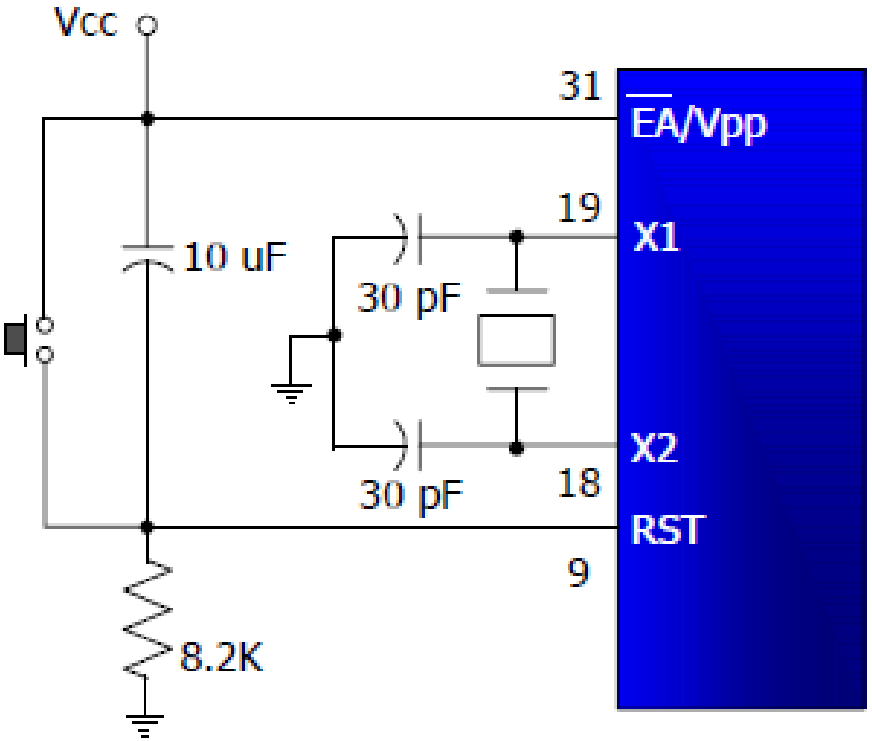
Nguồn cấp



Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển 89x51

Reset, Thạch anh, điều khiển bộ nhớ



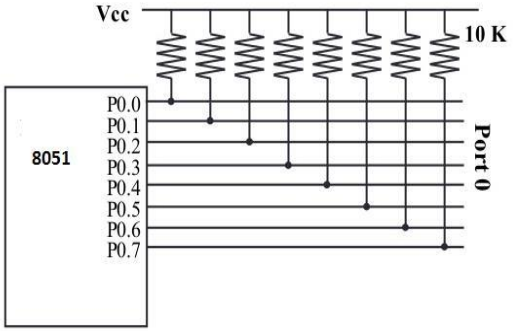
Chương 3. Vi điều khiển họ 8051

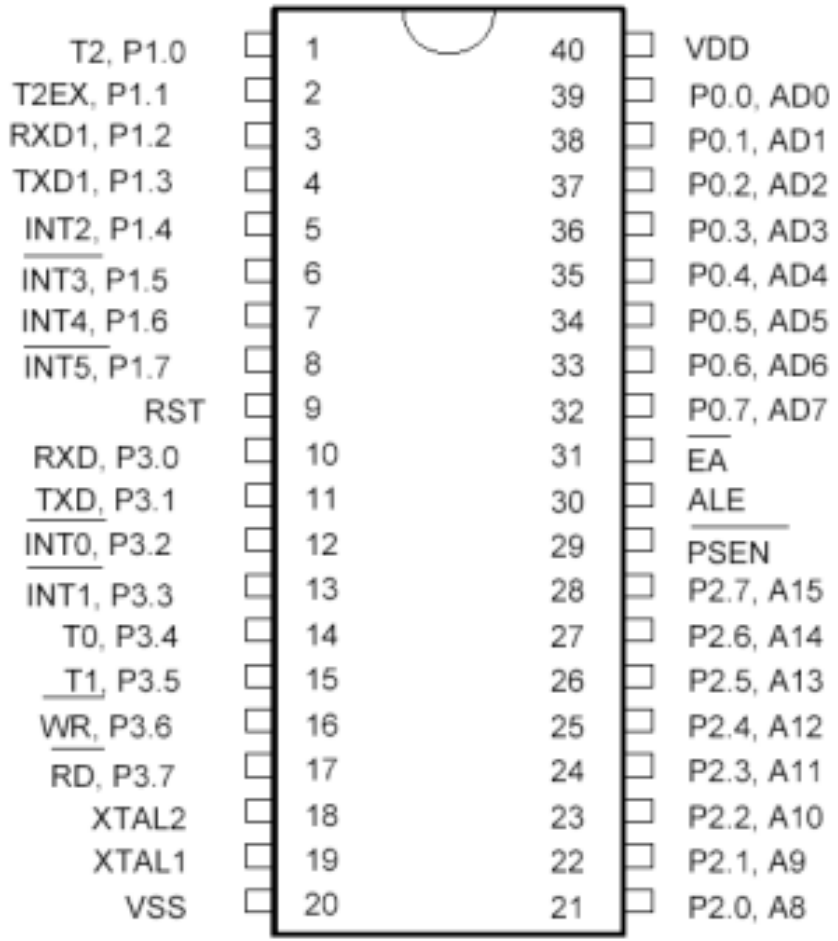
3.1. Cấu trúc vi điều khiển 89x51

Port 0:

Port 0 là một port xuất/nhập song hướng cực máng hở 8 bit. Nếu được sử dụng như là một ngõ xuất thì mỗi chân có thể kéo 8 ngõ vào TTL. Cần có các điện trở pullup (5k-10k) bên ngoài.

Ngoài ra còn có chức năng AD0- AD7





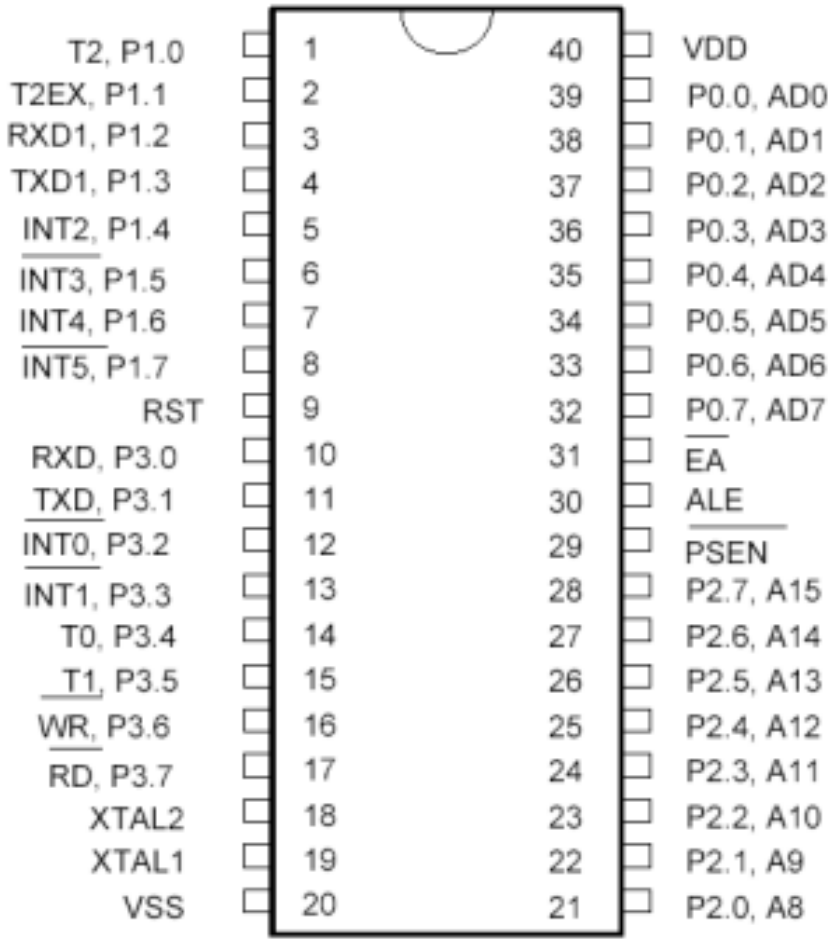
Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển 89x51

Port 1:

Port 1 là một port xuất/nhập song hướng 8 bit có các điện trở pullup bên trong.

Ngoài ra với họ 89x52 có thể các chức năng phụ



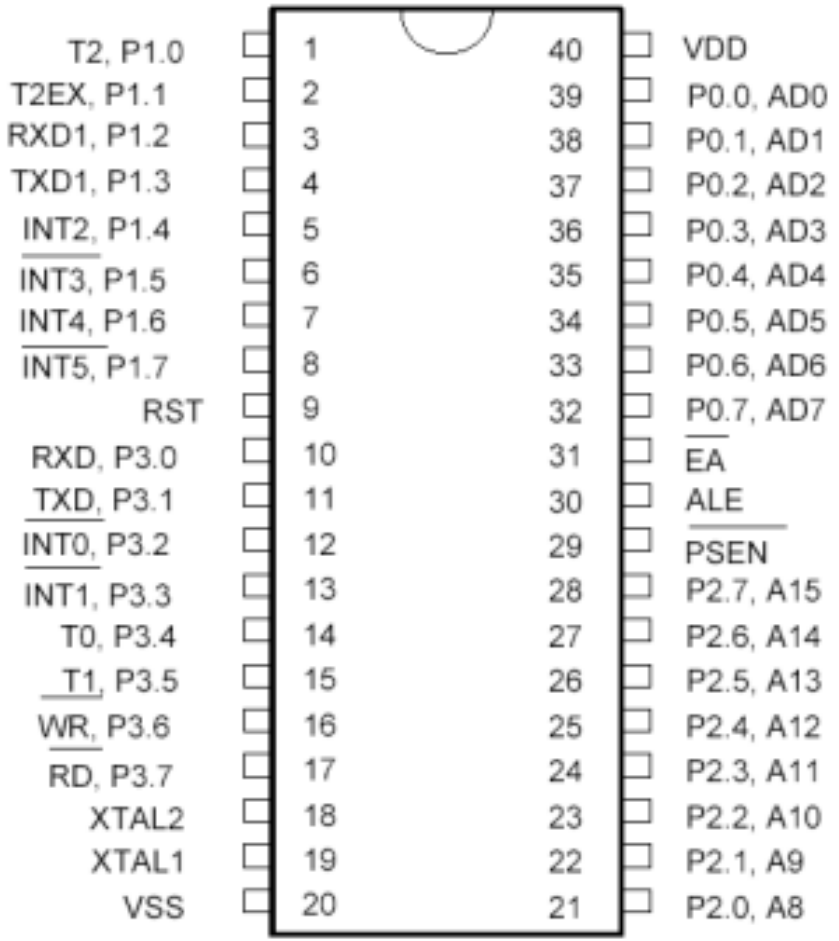
Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển 89x51

Port 2:

Port 2 là một port xuất/nhập song hướng 8 bit có các điện trở pullup bên trong.

Ngoài ra còn là A8-A15 trong lệnh MOVX @DPTR.



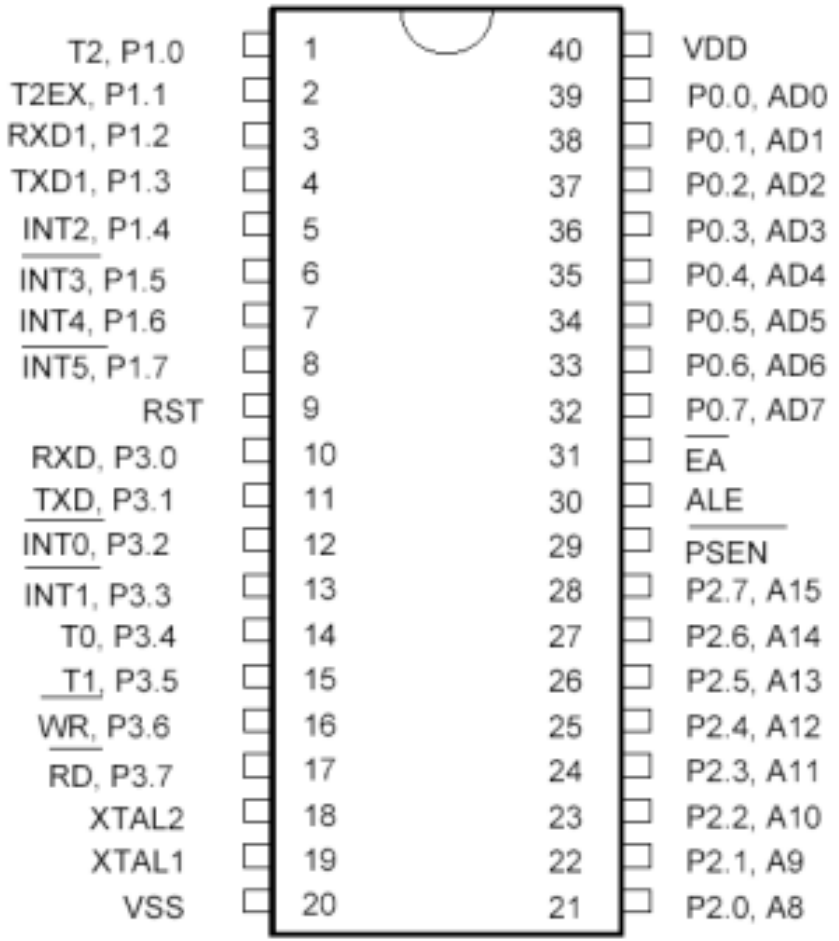
Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển 89x51

Port 3:

Port 3 là một port xuất - nhập song hướng 8 bit có điện trở pullup nội bên trong.

Ngoài ra Port 3 cũng cung cấp các chức năng của các đặc trưng đặc biệt như được liệt kê dưới đây:



Chương 3. Vi điều khiển họ 8051

3.1. Cấu trúc vi điều khiển 89x51

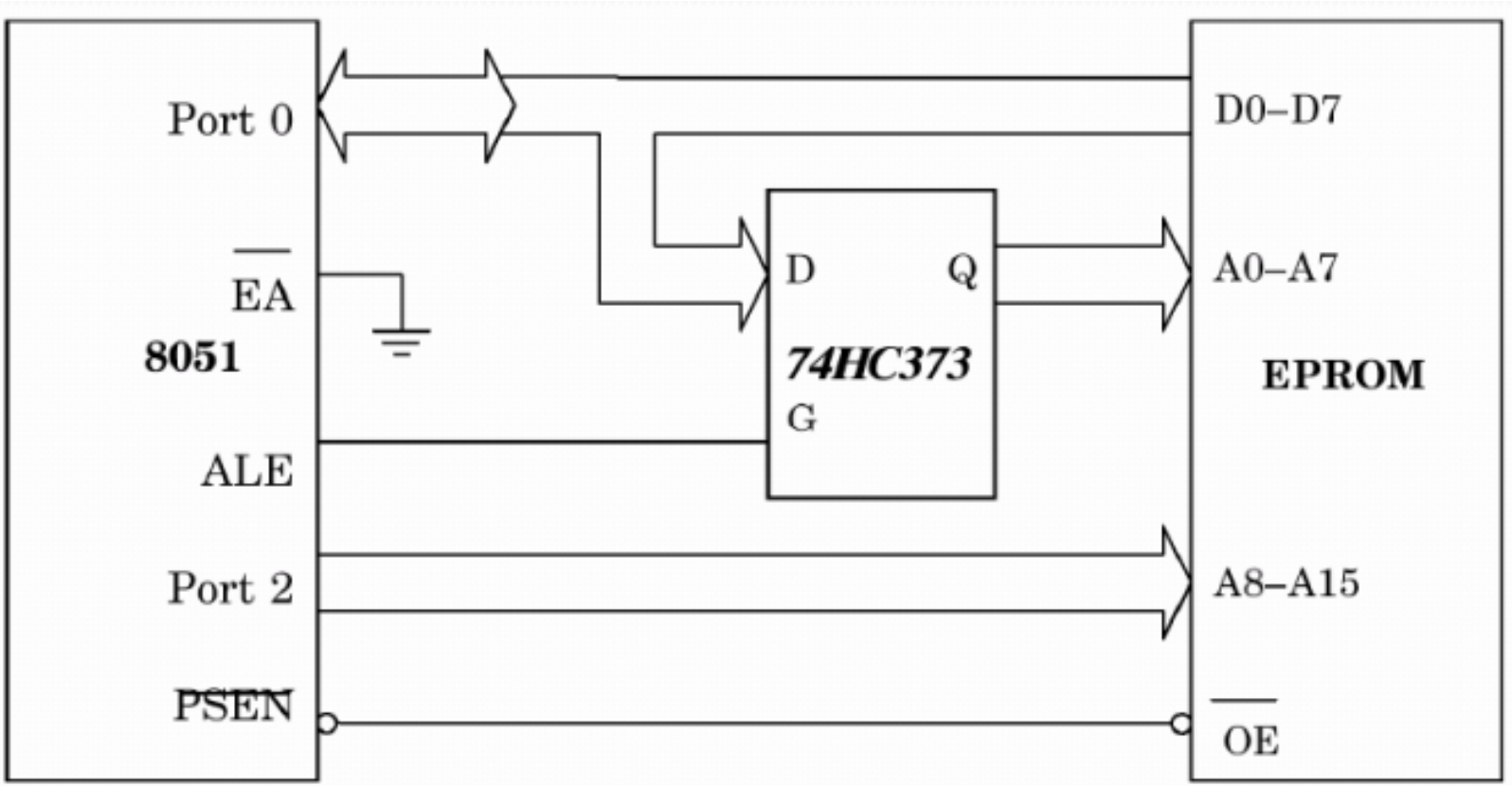
Port 3:

Bit	Tên	Địa chỉ bit	Chức năng chuyển đổi
P3.0	RXD	B0H	Dữ liệu nhận cho port nối tiếp.
P3.1	TXD	B1H	Dữ liệu phát cho port nối tiếp.
P3.2	$\overline{\text{INT0}}$	B2H	Ngắt 0 bên ngoài.
P3.3	$\overline{\text{INT1}}$	B3H	Ngắt 1 bên ngoài.
P3.4	T0	B4H	Ngõ vào của timer/counter 0.
P3.5	T1	B5H	Ngõ vào của timer/counter 1.
P3.6	$\overline{\text{WR}}$	B6H	Xung ghi bộ nhớ dữ liệu ngoài.
P3.7	$\overline{\text{RD}}$	B7H	Xung đọc bộ nhớ dữ liệu ngoài.

Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Bộ nhớ ngoài

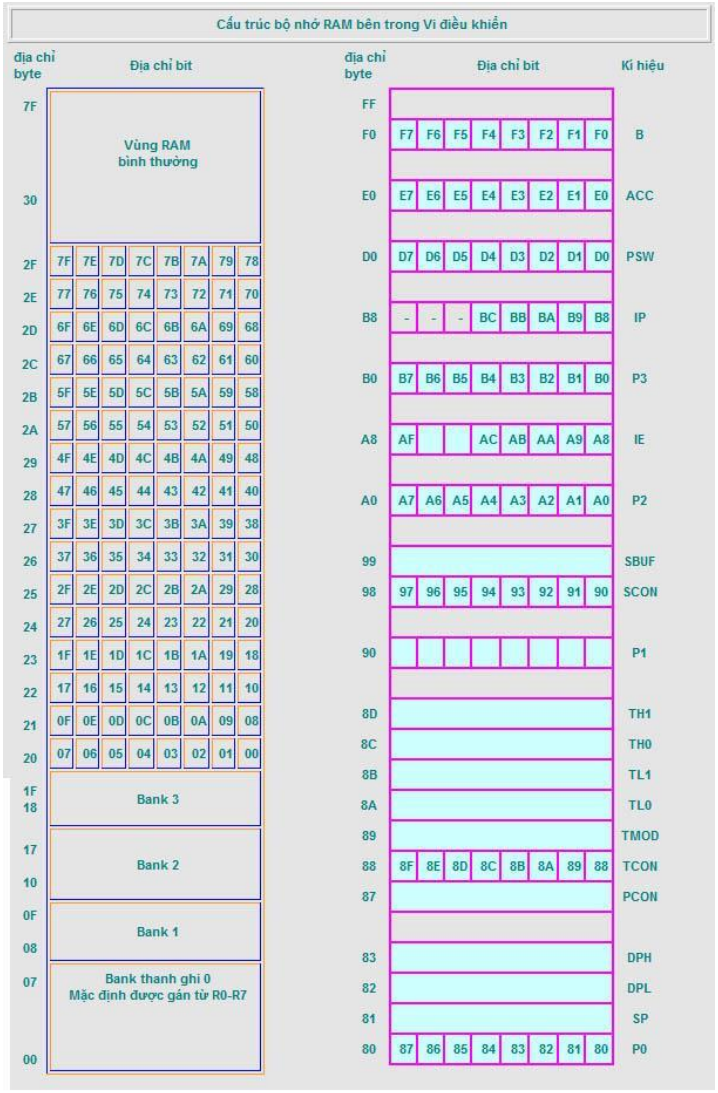


Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

128byte

00H-FFH

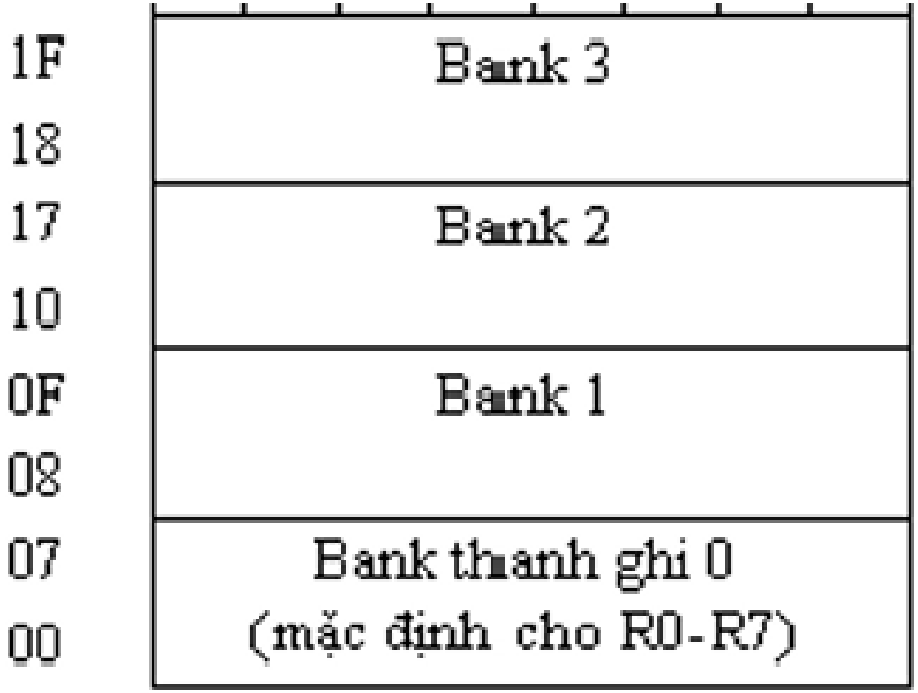


Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Ram Thanh ghi

00H-1FH



Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Ví dụ: R7 = R2 + R5

R2=	0	1	1	0	0	1	1	0
R5=	1	0	1	0	1	0	0	1
R7=								

Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

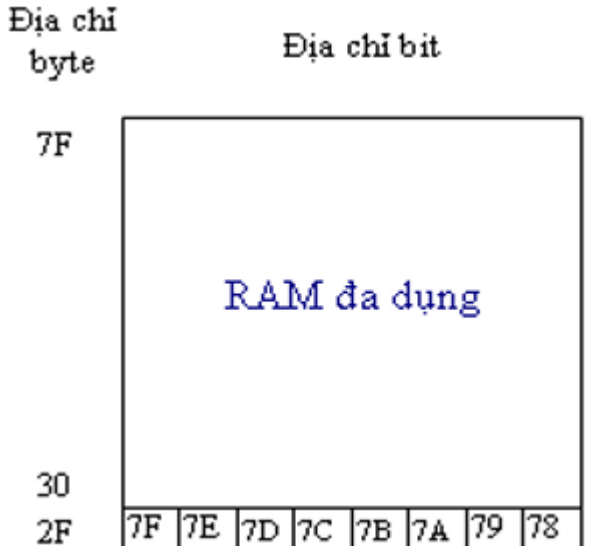
Ram địa chỉ hoá bit

20H-2FH

Ram đa dụng

30H-7FH

30								
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
2D	6F	6E	6D	6C	6B	6A	69	68
2C	67	66	65	64	63	62	61	60
2B	5F	5E	5D	5C	5B	5A	59	58
2A	57	56	55	54	53	52	51	50
29	4F	4E	4D	4C	4B	4A	49	48
28	47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38
26	37	36	35	34	33	32	31	30
25	2F	2E	2D	2C	2B	2A	29	28
24	27	26	25	24	23	22	21	20
23	1F	1E	1D	1C	1B	1A	19	18
22	17	16	15	14	13	12	11	10
21	0F	0E	0D	0C	0B	0A	09	08
20	07	06	05	04	03	02	07	00
1F	Bank 3							



Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Ví dụ: Xác định giá trị thanh ghi 2C = thanh ghi 99H-thanh ghi 7FH

Biết : Thanh ghi 99H = #32H, Thanh ghi 7FH = #25H

99H = # 00110010

7FH = #00100101

a) Thanh ghi 2CH = #??

Thanh ghi 37H = thanh ghi 99H: #00000101

a) Thanh ghi 37H = #??

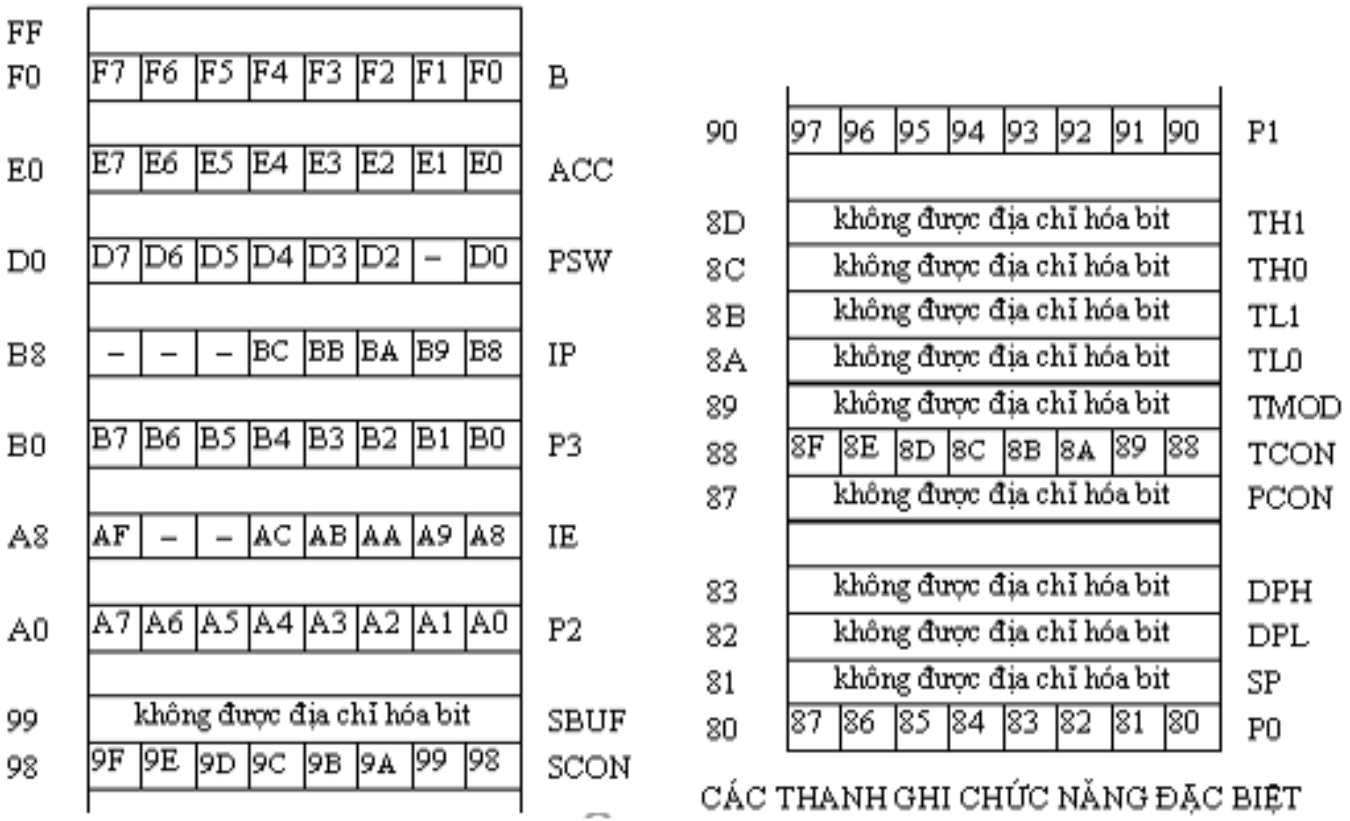
b) Thanh ghi R5 = thanh ghi 2CH AND thanh ghi 37H

Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Thanh ghi đặc biệt

80H - FFH



CÁC THANH GHI CHỨC NĂNG ĐẶC BIỆT

Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Ví dụ: Thiết lập Bít thứ thứ 5 trong 4F = “1”

4FH= 10101001

Cách 1: Chuyển vào vùng địa chỉ hóa từng bít → lập trình Bít tương ứng = “1” rồi trả về vị trí 4F.

Cách 2: Chúng ta thực hiện phép toán OR (nếu Set lên “1”), Phép toán AND (Nếu Reset xuống “0”)

Chương 3. Vi điều khiển họ 8051

3.2. Tổ chức bộ nhớ

Thanh ghi
PSW

Bit	Symbol	Address	Description
PSW.7	CY	D7H	Carry flag
PSW.6	AC	D6H	Auxiliary carry flag
PSW.5	F0	D5H	Flag 0
PSW.4	RS1	D4H	Register bank select 1
PSW.3	RS0	D3H	Register bank select 0
PSW.2	OV	D2H	Overflow flag
PSW.1	--	D1H	Reserved
PSW.0	P	D0H	Even parity flag

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh dịch chuyển dữ liệu

MOV <destination>, <source>

```
MOV     A, Rn          ; (A)  (Rn)
MOV     A, direct      ; (A)  (direct)
MOV     A, @ Ri        ; (A)  ((Ri))
MOV     A, # data      ; (A)  # data
MOV     Rn, A          ;  (Rn)  (A)
MOV     Rn, direct     ; (Rn)  (direct)
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh dịch chuyển dữ liệu

MOV <destination>, <source>

MOV Rn, # data ; (Rn) # d

MOV direct, A ; (direct) (

MOV direct, Rn ; (direct) (

MOV direct, direct ; (direct) (direc

MOV direct, @ Ri ; (direct) ((Ri)

```
1   org 00h
2   MOV P1, #00h
3   MOV P2, #00h
4   MOV 0B0H, #00h
5
6   mov 27H,#15h
7   Mov R1,#27H
8   MOv A, R1 ;A=27H
9   MOv A, @R1 ;A=15H
10  Mov P1, A ;P1 = #15h
11  mov P2, #01010101b
12  end
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh dịch chuyển dữ liệu

MOV <destination>, <source>

MOV direct, # data ; (direct) data

MOV @ Ri, A ; ((Ri)) (A)

MOV @ Ri, direct ; ((Ri)) (direct)

MOV @ Ri, # data ; ((Ri)) # data

MOV DPTR, # data16 ; (DPTR) # data16

MOVC A, @ A + DPTR ; (A) (A) + (DPTR)

MOVC @ A + PC ; (PC) (PC) + 1; (A) (A) + (PC)

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh dịch chuyển dữ liệu

MOV <destination>, <source>

MOVX	*A, @ Ri*	*; (A) ((Ri))*
MOVX	*A, @ DPTR*	*; (A) ((DPTR))*
MOVX	*@ Ri, A*	*; ((Ri)) (A)*
MOVX	*@ DPTR, A*	*; ((DPTR)) (A)*

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh dịch chuyển dữ liệu

MOV <destination>, <source>

PUSH direct ; Cất dữ liệu vào Ngăn xếp;(SP)(SP) + 1; (SP) (Drirect)

POP direct ; Lấy từ Ngăn xếp ra direct; (direct) ((SP));(SP) (SP) - 1

XCH A, Rn ; Đổi chổ nội dung của A với Rn; (A) (Rn)

XCH A, direct ; (A) (direct)

XCH A, @ Ri ; (A) ((Ri))

XCHD A, @ Ri ; Đổi chổ 4 bit thấp của (A) với ((Ri));(A3A0) ((Ri3Ri0))

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

```
org 00h
Mov 25h,#0AAH
PUSH 25H
POP 25H
Mov A, #55H
XCH A, 25h

end
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh số học (Arithmetic Instrustion)

ADD A, <src, byte>

ADD A, Rn ; (A) (A) + (Rn)

ADD A, direct ; (A) (A) + (direct)

ADD A, @ Ri ; (A) (A) + ((Ri))

ADD A, # data ; (A) (A) + # data

ADDC A, Rn ; (A) (A) + (C) + (Rn)

ADDC A, direct ; (A) (A) + (C) + (direct)

ADDC A, @ Ri ; (A) (A) + (C) + ((Ri))

ADDC A, # data ; (A) (A) + (C) + # data

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh số học (Arithmetic Instrustion)

SUBB A, <src, byte>

SUBB A, Rn *; (A) (A) - (Rn)*

SUBB A, direct *; (A) (A) - (direct)*

SUBB A, @ Ri *;(A) (A) - ((Ri))*

SUBB A, # data *; (A) (A) -- # data*

Ví dụ: mov a, #38h; số thập phân của 38h là 56 và nhị phân là 0011 1000b

Add a, #2fh ; số thập phân 2f là 47 và nhị phân là 0010 1111b

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh số học (Arithmetic Instrustion)

MUL AB ;

A chứa byte thấp

B chứa byte cao

```
1  org 00h
2  mov a, #02h ; a=02
3  add a,#0feh ; a= 02+fe = 00
4  mov r1,#15h ;
5  addc a, r1 ; A=16H
6  subb a, #06h ;A=10H
7  Mov b,#3AH ; b=30
8  Mul AB ; 10h*3Ah = 03A0h --> A=A0H, B =03H
9  DIV AB ; A0:03 = 35H du 01H
10 end
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh số học (Arithmetic Instrustion)

DIV AB ; (A) Integer Result of [(A)/(B)]; cờ OV

; (B) Remainder of [(A)/(B)]; cờ Carry xóa

DA A ; Điều chỉnh thanh ghi A thành số BCD đúng trong phép cộng BCD (thường DA A đi kèm với ADD, ADDC)

Ví dụ: mov a, #47h ; *nạp con số BCD 47h vào thanh ghi A*

Mov b, #25h ; *nạp con số BCD 25h vào thanh ghi B*

Add a, b ; *cộng nội dung có trong thanh a và thanh b kết quả ghi trong a*

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh số học (Arithmetic Instrustion)

INC <byte>

INC	*A*	*; (A) (A) + 1*
INC	*direct*	*; (direct) (direct) + 1*
INC	*Ri*	*; ((Ri)) ((Ri)) + 1*
INC	*Rn*	*; (Rn) (Rn) + 1*
INC	*DPTR*	*; (DPTR) (DPTR) + 1*

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh số học (Arithmetic Instrustion)

DEC <byte>

```
DEC    A          ; (A)  (A) - 1
DEC    direct     ; (direct)  (direct) - 1
DEC    @Ri        ; ((Ri))  ((Ri)) - 1
```

```
MULL   AB    ; (A)  LOW [(A) x (B)];có ảnh hưởng cờ OV
             ; (B) HIGH [(A) x (B)];cờ Cary được xóa.
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh logic (Logic Operation)

ANL <dest - byte> <src - byte>

ANL A, Rn ; (A) (A) AND (Rn).

ANL A, direct ; (A) (A) AND (direct).

ANL A,@ Ri ; (A) (A) AND ((Ri)).

ANL A, # data ; (A) (A) AND (# data).

ANL direct, A ; (direct) (direct) AND (A).

ANL direct, # data ; (direct) (direct) AND # data.

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh logic (Logic Operation)

ORL <dest - byte> <src - byte>

ORL A, Rn ; (A) (A) OR (Rn).

ORL A, direct ; (A) (A) OR (direct).

ORL A,@ Ri ; (A) (A) OR ((Ri)).

ORL A, # data ; (A) (A) OR # data.

ORL direct, A ; (direct) (direct) OR (A).

ORL direct, # data ; (direct) (direct) OR # data.

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh logic (Logic Operation)

XRL <dest - byte> <src - byte>

XRL	*A, Rn*	*; (A) (A) (Rn).*
XRL	*A, direct*	*; (A) (A) (direct).*
XRL	*A,@ Ri*	*; (A) (A) ((Ri)).*
XRL	*A, # data*	*; (A) (A) # data.*
XRL	*direct, A*	*; (direct) (direct) (A).*
XRL	*direct, # data*	*; (direct) (direct) # data.*

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

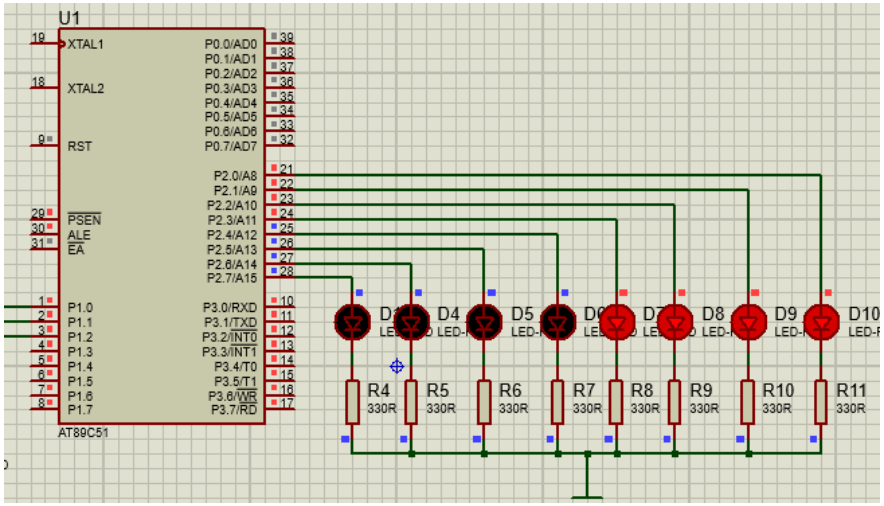
Các lệnh logic (Logic Operation)

```
CLR     A                ; (A)  0
CLR     C                ; (C)  0
CLR     Bit              ; (Bit)  0
RL      A ;Quay vòng thanh ghi A qua trái 1 bit (An + 1)(An); n = 06
RLC     A   ; Quay vòng thanh ghi A qua trái 1 bit có cờ Carry
RR      A   ; Quay vòng thanh ghi A qua phải 1 bit
RRC  A      ; Quay vòng thanh ghi A qua phải 1 bit có cờ Carry
SWAP A ; Đổi chỗ 4 bit thấp và 4 bit cao của A cho nhau
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

```
MOv  P2,#00h
CALL delay
mov  A,#00000001b
mov  P2,A ; 1
CALL delay
SETB C
RLC  A ;A=00000011
mov  P2,A ;2
CALL delay
SETB C
```



Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh rẽ nhánh

JC rel ; Nhảy đến "rel" nếu cờ Carry C = 1.

JNC rel ; Nhảy đến "rel" nếu cờ Carry C = 0.

JB bit, rel ; Nhảy đến "rel" nếu (bit) = 1.

JNB bit, rel ; Nhảy đến "rel" nếu (bit) = 0.

JBC bit, rel ; Nhảy đến "rel" nếu bit = 1 và xóa bit.

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh rẽ nhánh

ACALL addr11; Lệnh gọi tuyệt đối trong page 2K;

LCALL addr16 ; Lệnh gọi dài chương trình con trong 64K.

RET ; Kết thúc chương trình con trở về chương trình chính.

RETI ; Kết thúc thủ tục phục vụ ngắt quay về chương trình chính

AJMP Addr11 ; Nhảy tuyệt đối không điều kiện trong 2K.

LJMP Addr16 ; Nhảy dài không điều kiện trong 64K

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh rẽ nhánh

JMP @ A + DPTR;Nhảy không điều kiện đến địa chỉ (A) + (DPTR)

JZ rel ; Nhảy đến A = 0. Thực hành lệnh kế nếu A 0.

JNZ rel ; Nhảy đến A 0. Thực hành lệnh kế nếu A = 0.

CJNE A, direct, rel ; So sánh và nhảy đến A direct

CJNE A, # data, rel ; Tương tự lệnh CJNE A, direct, rel.

CJNE Rn, # data, rel ; Tương tự lệnh CJNE A, direct, rel.

CJNE @ Ri, # data, rel ; Tương tự lệnh CJNE A, direct, rel.

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh rẽ nhánh

Ví dụ: sử dụng lệnh rẽ nhánh để tạo delay:

```
Delay:
Mov r5, #10
Vong1: mov      r6, #50
Djnz r6,$
Djnz r5, vong1
ret
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh luận lý hay BIT(Boolean Instruction)

```
CLR     C     ; Xóa cờ Carry xuống 0. Có ảnh hưởng cờ Carry.
CLR     BIT   ; Xóa bit xuống 0. Không ảnh hưởng cờ Carry
SETB    C     ; Set cờ Carry lên 1. Có ảnh hưởng cờ Carry.
SETB    BIT   ; Set bit lên 1. Không ảnh hưởng cờ Carry.
CPL     C     ; Đảo bit cờ Carry. Có ảnh hưởng cờ Carry.
CPL     BIT   ; Đảo bit. Không ảnh hưởng cờ Carry.
```

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh luận lý hay BIT(Boolean Instruction)

ANL C, BIT ; (C) (C) AND (BIT) : Có ảnh hưởng cờ Carry.

ANL C, /BIT ; (C) (C) AND NOT (BIT):Không ảnh hưởng cờ Carry.

ORL C, BIT ; (C) (C) OR (BIT) : Tác động cờ Carry.

ORL C, /BIT ; (C) (C) OR NOT (BIT) : Tác động cờ Carry.

MOV C, BIT ; (C) (BIT) Cờ Carry bị tác động.

MOV BIT, C ; (BIT) (C) Không ảnh hưởng cờ Carry

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Các lệnh luận lý hay BIT(Boolean Instruction)

ANL C, BIT ; (C) (C) AND (BIT) : Có ảnh hưởng cờ Carry.

ANL C, /BIT ; (C) (C) AND NOT (BIT):Không ảnh hưởng cờ Carry.

ORL C, BIT ; (C) (C) OR (BIT) : Tác động cờ Carry.

ORL C, /BIT ; (C) (C) OR NOT (BIT) : Tác động cờ Carry.

MOV C, BIT ; (C) (BIT) Cờ Carry bị tác động.

MOV BIT, C ; (BIT) (C) Không ảnh hưởng cờ Carry

Chương 3. Vi điều khiển họ 8051

3.3. Tập lệnh vi điều khiển 8051

Bài tập

CHƯƠNG 4 LẬP TRÌNH HỢP NGỮ

4.1. Cấu trúc chương trình
4.2. Phần mềm lập trình
4.3. Phần mềm mô phỏng

CHƯƠNG 4 LẬP TRÌNH HỢP NGỮ

4.1. Cấu trúc chương trình

Định địa chỉ

Địa chỉ gián tiếp (Indirect Address): MOV A, @R0
Dữ liệu tức thời (Immediate Data): ORL 40H, # CONSTANT
Địa chỉ dữ liệu (Data Address): MOV A, 45H
Địa chỉ Bit (Bit Address) : SETB ACC, 7
Các cơ sở số (Number Basses):
MOV A, # 15 ; Thập phân
MOV A , #1111B ; Nhị phân
MOV A , #30H ; Hex
MOV A , #315D ; Thập phân
MOV A , #317Q ; Octal
CJNZ A , # 'Q', AGAIN; chuỗi ký tự

CHƯƠNG 4 LẬP TRÌNH HỢP NGỮ

4.1. Cấu trúc chương trình

Định nghĩa

Chỉ lẫn EQU gán giá trrị số cho tên của ký hiêu được định nghĩa. Symbol EQU Expression (biểu thức). Ví dụ: EPROM SEGMENT CODE cho biết EPROM của một segment kiểu code. Dạng chỉ thị EQU: symbol EQU Expression MESSAGE DB ‘hello’ Dạng chỉ thị BIT: symbol BIT Expression

Khai báo lưu trữ DS (Define Storage)

Dạng phát biểu DS là: [label:] DS Expression

Khai báo DBIT (Define Bit)

BSEG ; segment bit truyệt đối KBFLAG DBIT 1 ; trạng thái của bàn phím

PRFLAG DBIT 1 ; trạng thái của máy in

DKFLAG DBIT 1 ; trạng thái của đĩa

CHƯƠNG 4 LẬP TRÌNH HỢP NGỮ

4.1. Cấu trúc chương trình

Khai báo

Sự thành lập chỉ dẫn DB: [label:] DB Expression, [Expression], [...]

CSEG AT 0100H SQUARES: DB 0, 1, 4, 9, 16, 25 ;Bình phương từ 0-5

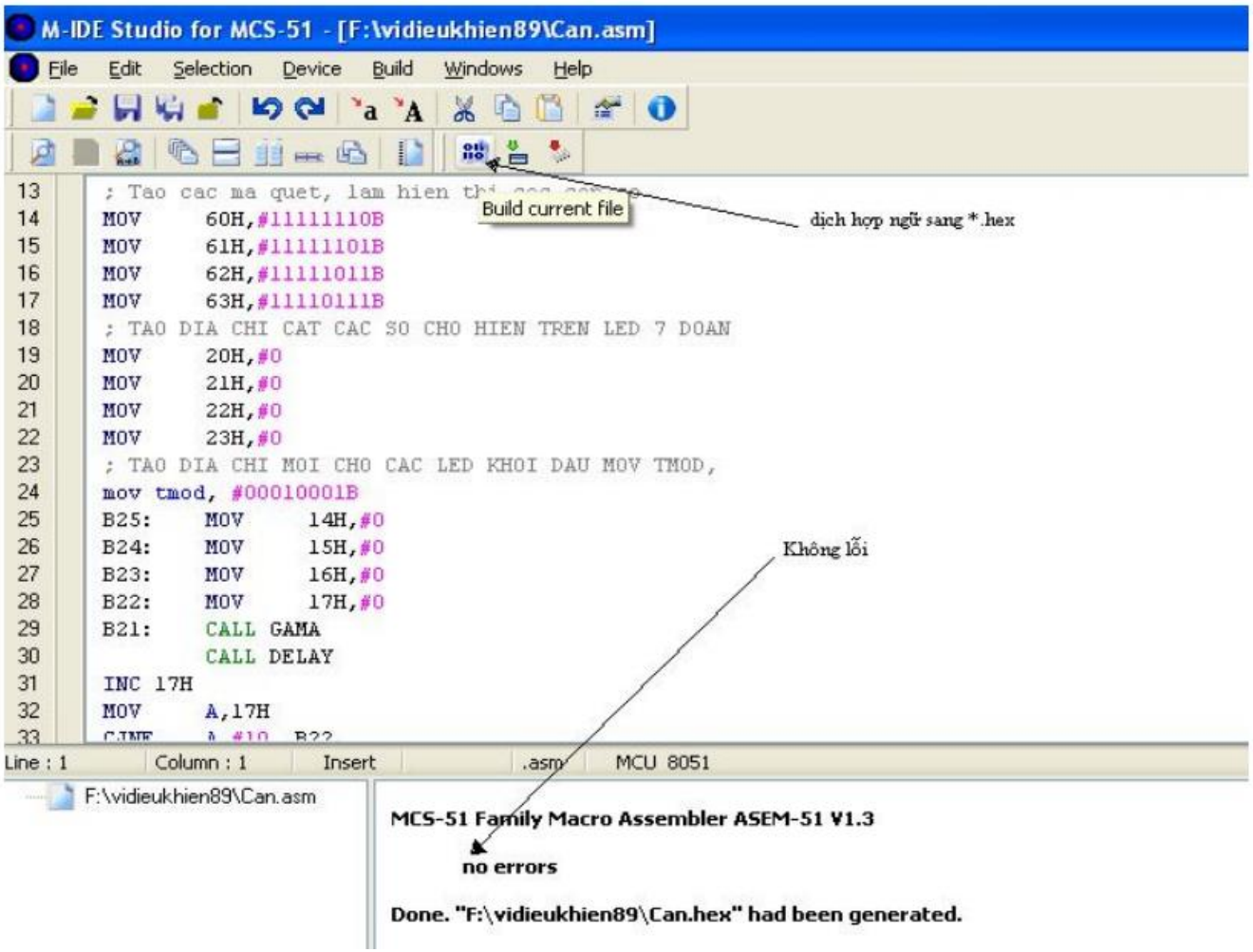
MESSAGE: DB ‘login’, 0 ; chuỗi ký tự kết thúc bởi 0

Khai báo từ DW (Define Word) Sự thành lập: [label:] DW Expression [,Expression], [...]

Chỉ dẫn PUBLIC Dạng chỉ dẫn PUBLIC symbol [, symbol]…

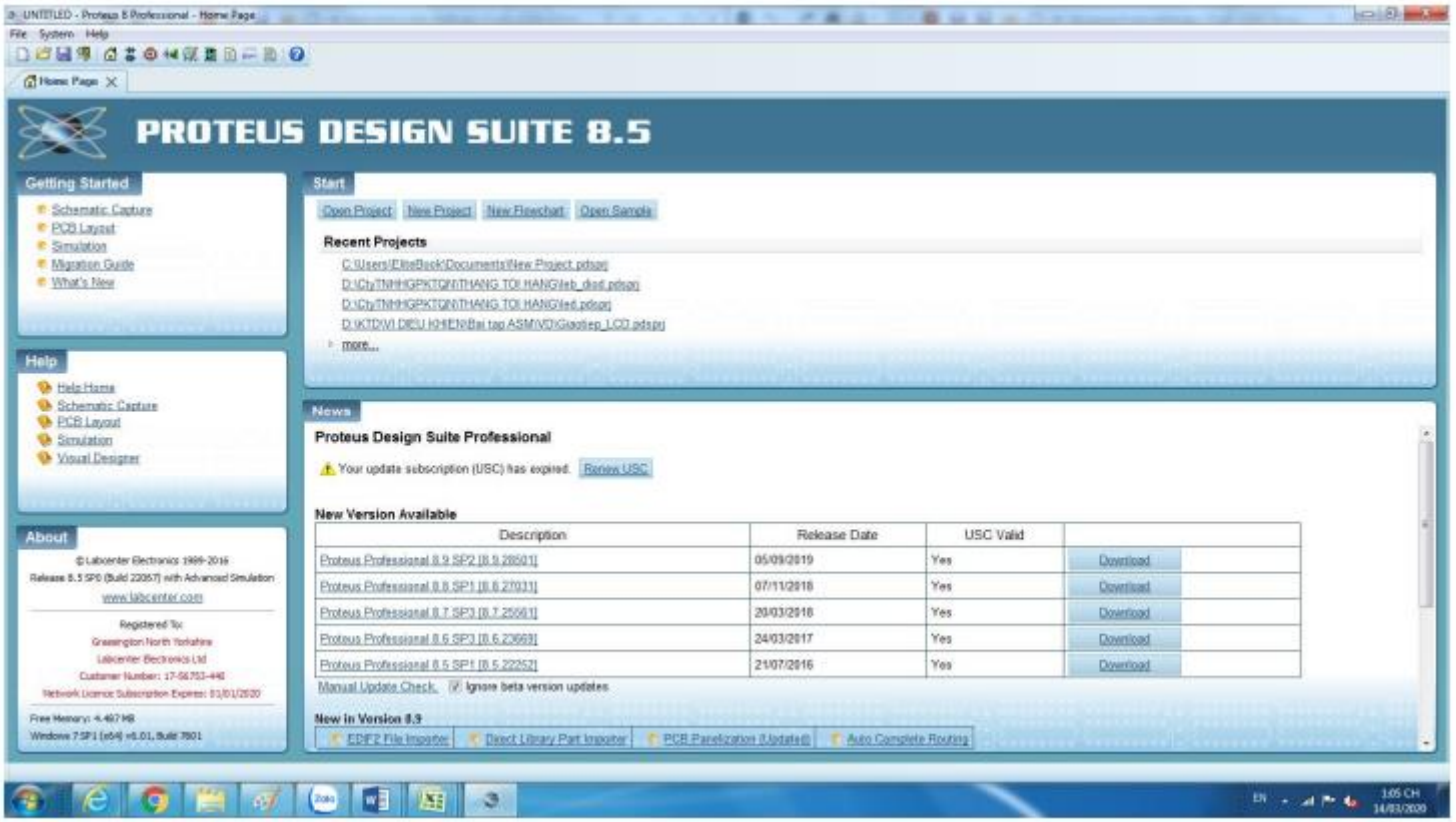
CHƯƠNG 4 LẬP TRÌNH HỢP NGỮ

4.2. Phần mềm lập trình



CHƯƠNG 4 LẬP TRÌNH HỢP NGỮ

4.3. Phần mềm mô phỏng



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

5.2. Truyền thông nối tiếp

5.3. Ngắt

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

Trong 8051 có 4 chế độ Timer/counter có 2 bít C/T thanh ghi TMOD ở địa chỉ byte 89H quyết định chế độ Timer/counter.

GATE1	C/T	M1	M0	GATE0	C/T	M1	M0
Timer/Counter 1				Timer/Counter 0			

M1	M0	Modul	Chế độ hoạt động
0	0	0	Bộ định thời 13 bit (8 bitC/T và 5 bit đặt trước)
0	1	1	Bộ định thời 16 bit không đặt trước không tự nạp lại
1	0	2	Bộ định thời 8 bit tự nạp lại
1	1	3	Chế độ định thời chia tách

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

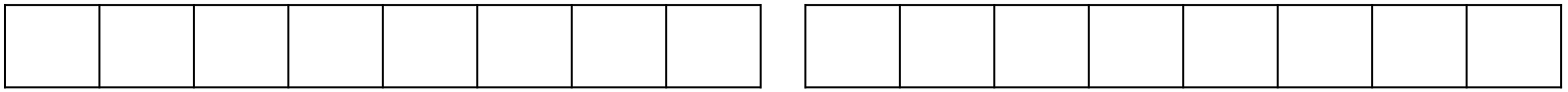
5.1. Timer/counter

GATE1	C/T	M1	M0	GATE0	C/T	M1	M0
Timer/Counter 1				Timer/Counter 0			

GATE = 0: Chế độ Timer/Counter; GATE = 1: Chế độ Ngắt;
C/T = 0: Lựa chọn Timer; GATE = 1: Chế độ NgắC/T = 0: Lựa chọn Counter;

M1=0:
M0=1 :
Chế độ chế độ 1 (16 bit đếm)

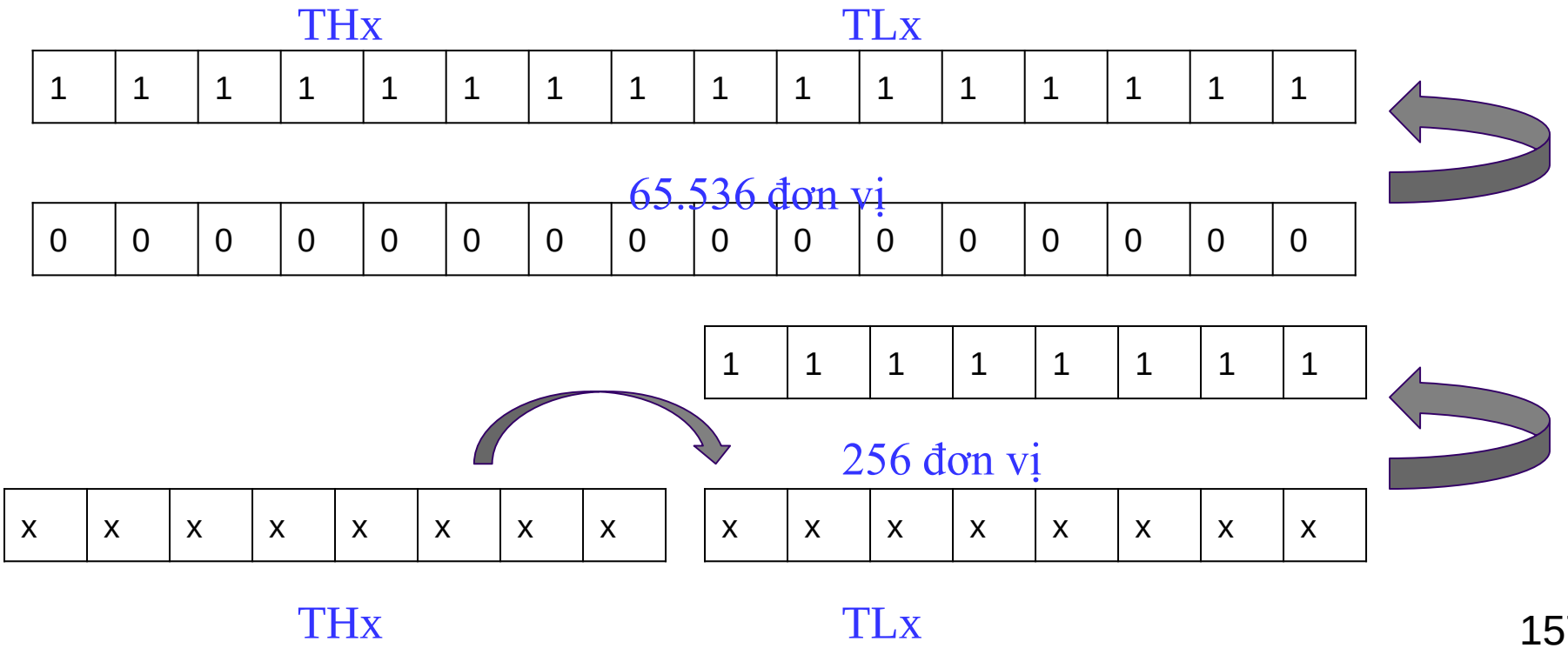
M1=1:
M0=0 :
Chế độ chế độ 2 (8 bit đếm)



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

GATE1	C/T	M1	M0	GATE0	C/T	M1	M0
Timer/Counter 1				Timer/Counter 0			



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

1. Hoạt động Timer 0, chế độ 1 (16 bit) → TMOD=01H
2. Hoạt động Counter 0, chế độ 1 (16 bit) → TMOD=05H
3. Hoạt động Counter 1, chế độ 1 (16 bit) → TMOD=xxH
4. Hoạt động Counter 1, chế độ 2 (8 bit) → TMOD=xxH
5. Hoạt động Counter 0, chế độ 2 (8 bit) → TMOD=xxH
6. Hoạt động Timer 0, chế độ 2 (8 bit) → TMOD=xxH
7. Hoạt động Timer 1, chế độ 2 (8 bit) → TMOD=xxH
8. Hoạt động Timer 1, chế độ 1 (16 bit) → TMOD=xxH
9. Hoạt động Timer 1, chế độ 1 (16 bit);Timer 0, chế độ 1 (16 bit); → TMOD=xxH
10. …..
11. ….

Bài tập:

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

thanh ghi TCON ở địa chỉ byte 88H quyết định chế độ Timer/counter.

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Counter1/Timer1		Counter0/Timer0					

```
MOV  TMOD, #01H
MOV  TH0, #20H
MOV TL0, #0C0H
SETB TR0
JNB TF0,$
CLR TF0
CLR TR0
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

```
ORG 00H
LL:
SETB P1.0
CALL DELAY
CLR P1.0
CALL DELAY
JMP LL
```

```
DELAY:
MOV TMOD, #01H ; Timer 0 che do 1 (16 bit)
MOV TH0, #00h ; nap gia tri ban dau
MOV TL0,#00H ; nap gia tri ban dau
JNB TF0,$ ; cho co tran tf=1
CLR TR0          ;dung timer
CLR TF0          ; xoa co tran
RET      ; ket thuc ct con
END
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

```
11  DELAY:; TRE 65.535-50.000 = 3CAFH ;
12  MOV TMOD, #01H ; Timer 0 che do 1 (16 bit)
13  MOV R1, #20 ; 20X50.000= 1S
14  VUNG1:
15  MOV TH0, #3CH ; nap gia tri ban dau
16  MOV TL0,#0AFH ; nap gia tri ban dau
17  setb TR0
18  JNB TF0,$ ; cho co tran tf=1
19  CLR TR0 ;dung timer
20  CLR TF0 ; xoa co tran
21  DJNZ R1, VUNG1
22  RET       ; ket thuc ct con
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.1. Timer/counter

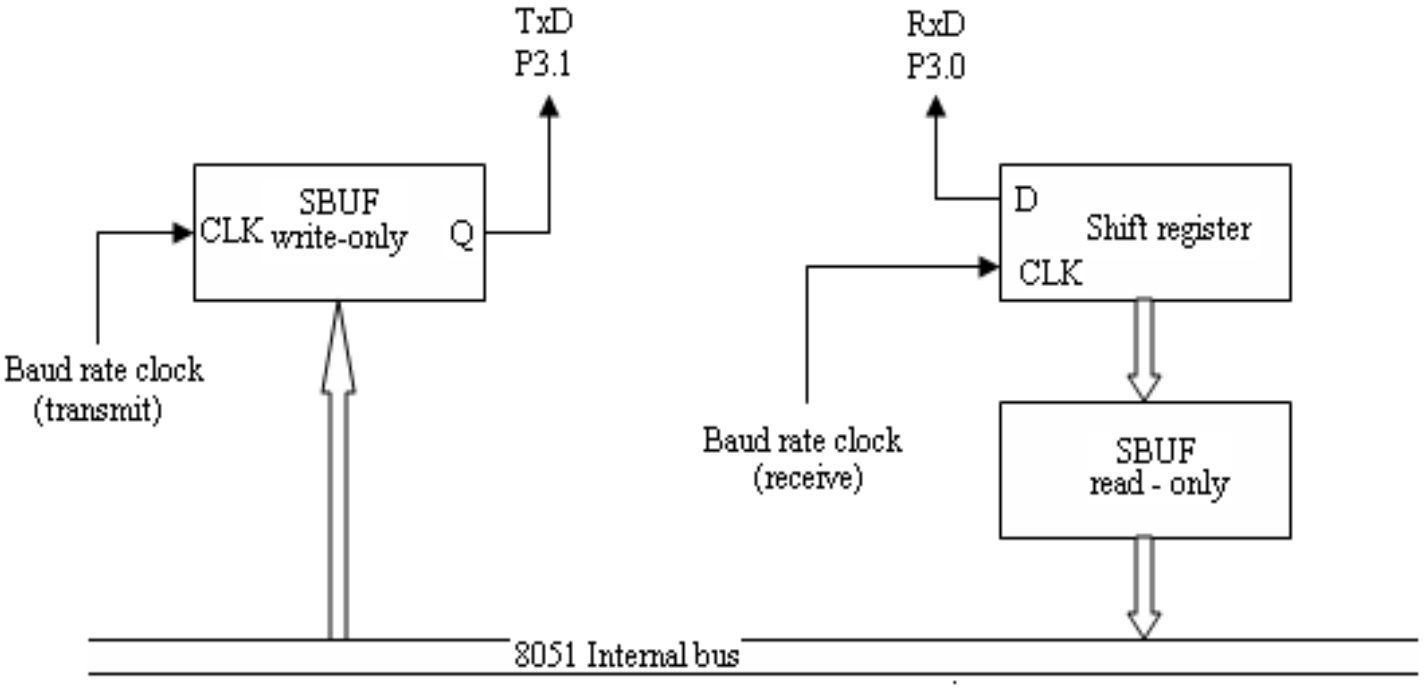
Tạo sóng xung vuông 10kHz trên chân số 10

```
; TAO SONG XUNG VUONG 10KHZ tren chan 10
ORG 0H
LL:
setb p3.0
call delay50us
clr p3.0
call delay50us
jmp LL
; tao thoi gian 50us
delay50us:
mov TMOD,#20h ; time 1 che do 8 bit
mov th1, #205
setb tr1
cho: jnb tf1,cho
clr tr1
clr tf1
ret
END
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

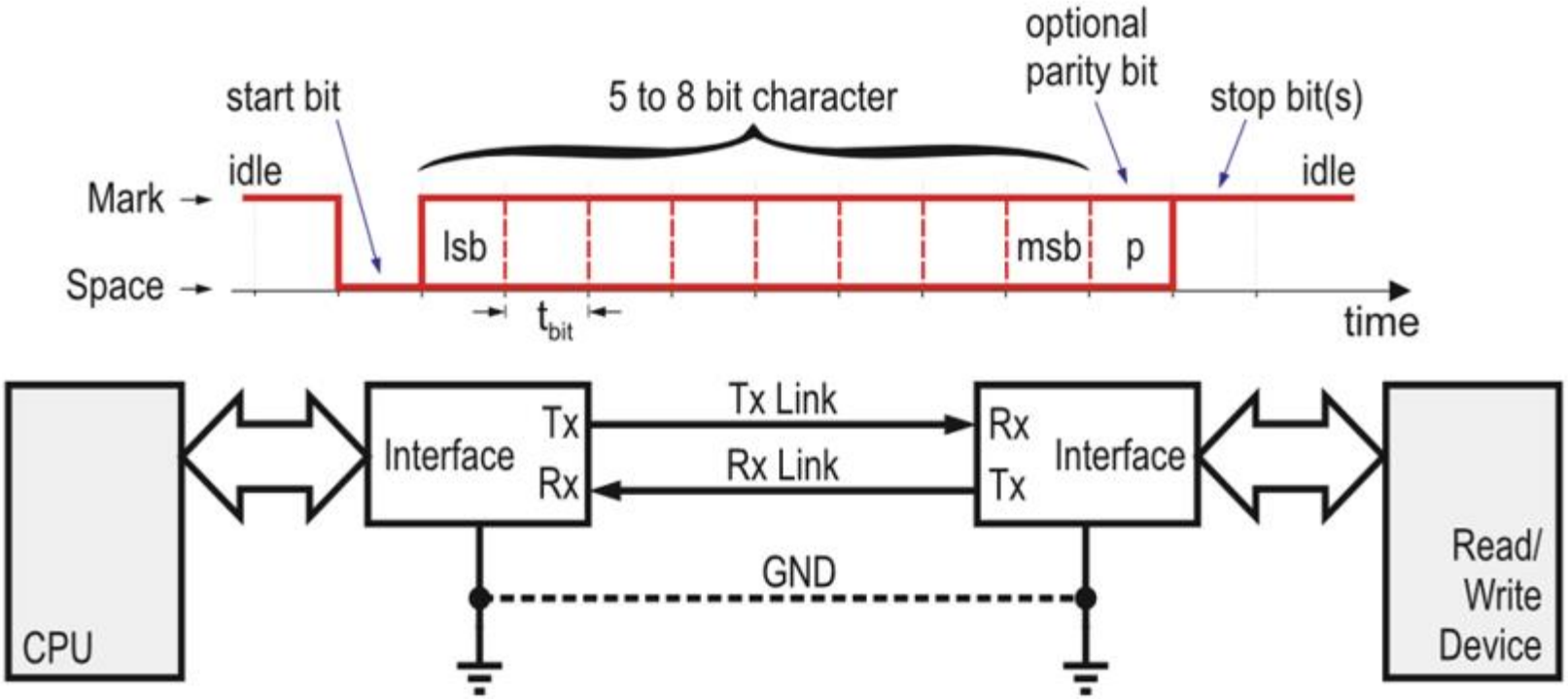
5.2 Truyền thông nối tiếp

Chức năng của port nối tiếp là thực hiện việc chuyển đổi dữ liệu song song thành nối tiếp khi phát và chuyển đổi dữ liệu nối tiếp thành song song khi thu. Các mạch phần cứng bên ngoài truy xuất thông qua chân TxD (P3.1 phát dữ liệu) và RxD (P3.0 thu dữ liệu)



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

Thanh ghi chọn chế độ SCON của port nối tiếp ở địa chỉ 98H, trước khi sử dụng port nối tiếp, thanh ghi SCON phải được khởi động đúng chế độ yêu cầu

Bit	Ký hiệu	Địa chỉ	Mô tả
SCON.7	SM0	9FH	**Bit 0 chọn chế độ port nối tiếp**
SCON.6	SM1	9EH	**Bit 1 chọn chế độ port nối tiếp**
SCON.5	SM2	9DH	**Bit này cho phép truyền thông đa xử lý ở chế độ 2 và 3, bit RI sẽ không được tích cực nếu bit thứ 9 nhận được 0**
SCON.4	REN	9CH	**Cho phép thu, phải được set để nhận ký tự**
SCON.3	TB8	9BH	**Bit phát 8. bit thứ 9 dược phát ở chế độ 2 và 3, set và xóa bởi phần mềm**
SCON.2	RB8	9AH	**Bit thu 8. bit thứ 9 nhận được**
SCON.1	TI	99H	**Cờ ngắt phát, cờ này được set ngay khi kết thúc việc phát 1 ký tự, xóa bởi phần mềm**
SCON.0	**RI**	**98H**	**Cờ ngắt nhận, cờ này được set ngay khi kết thúc việc thu 1 ký tự, xóa bởi phần mềm**

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

Bằng cách ghi giá trị 0 vào SM0, SM1. Dữ liệu nối tiếp được thu và phát thông qua chân RxD, TxD xuất xung clock dịch bit.

SM0	SM1	Chế độ	Mô tả	Tốc độ baud
0	0	0	Thanh ghi dịch	**Cố định (tần số dao động/12)**
0	1	1	UART 8 - bit	**Thay đổi (thiết lập bộ định thời)**
1	0	2	UART 9 - bit	**Cố định (tần số dao động/12 hoặc/64)**
1	**1**	**3**	**UART 9 - bit**	**Thay đổi bởi bộ định thời**

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

Thiết lập truyền dữ liệu UART 8 bit (cố định)

SCON = 40H; //0100 0000

Tương tự thiết lập Nhận UART 9 bít (thay đổi
Tương tự thiết lập Nhận UART 9 bít (Cố định)
Tương tự thiết lập TRUYỀN UART 9 bít (thay đổi
Tương tự thiết lập TRUYỀN UART 9 bít (Cố định)

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

Để lập trình 8051 truyền các byte ký tự nối tiếp thì cần phải thực hiện các bước sau đây:

1. Nạp thanh ghi **SCON** chế độ UART.
2. Nạp thanh ghi **TMOD** Timer 1 chế 2
3. Nạp thanh ghi **TH1** giá trị tạo tốc độ baud
4. Cho giá trị truyền vào **SBUF (truyền)**
5. Bật **TR1=1** để khởi động Timer1.
6. Đợi **TI=1, xóa TI**.
7. Lấy dữ liệu ở **SBUF**. **(nhận)**

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

Dùng Timer1 tạo Baud rate:

- Tốc độ (Baud) của cổng 8051 phải phù hợp với máy tính: 100, 150, 300, 600, 1200, 2400, 4800, 9600, 19200. Khuyến cáo dùng thạch anh có tần số 11.0592Mhz để có thể tạo ra tốc độ Baud với sai số thấp (gần như là 0).
- Timer 1 được dùng để tạo tốc độ baud cho truyền nối tiếp ,cần phải đặt Timer 1 ở Mode 2 – 8 bit tự nạp lại, nếu đặt ở chế độ khác sẽ tốn vài chu kỳ máy cho quá trình thiết lập lại giá trị cho TH1,TL1.
- Đặt TH1 theo bảng sao để có tốc độ Baud tương ứng với SMOD = 0 (bit D7 trong thanh ghi PCON, XTAL = 11.0592 MHz) :

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

Tốc độ Baud	TH1 (Thập phân)	TH1 (Hexa)
9600	-3	FD
4800	-6	FA
2400	-12	F4
1200	-24	E8

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

```
1   ;Tao chuong trinh truyen du lieu
2   org 00h
3   Truyen:
4   mov scon, #40h ; truyen UART 8 thay doi
5   mov tmod, #20h; time1 che do 8 tu nap
6   mov th1, #0FDh; 9600 baud
7   mov sbuf, 'B'; truyen ky tu B
8   ;(ASCII là 42H  STOP+01000010+START)
9   setb tr1; kich hoat truyen
10  jnb ti, $ ; cho co xong
11  clr ti
12  ret
13  end
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

```
15  Nhan:
16  mov scon, #0D0h ; nha UART 9 thay doi
17  mov tmod, #20h; time1 che do 8 tu nap
18  mov th1, #0FAh; 4800 baud
19  setb tr1; kich hoat nhan
20  jnb ri, $ ; nhan xong
21  clr ri
22  mov A, sbuf;
23  ret
```

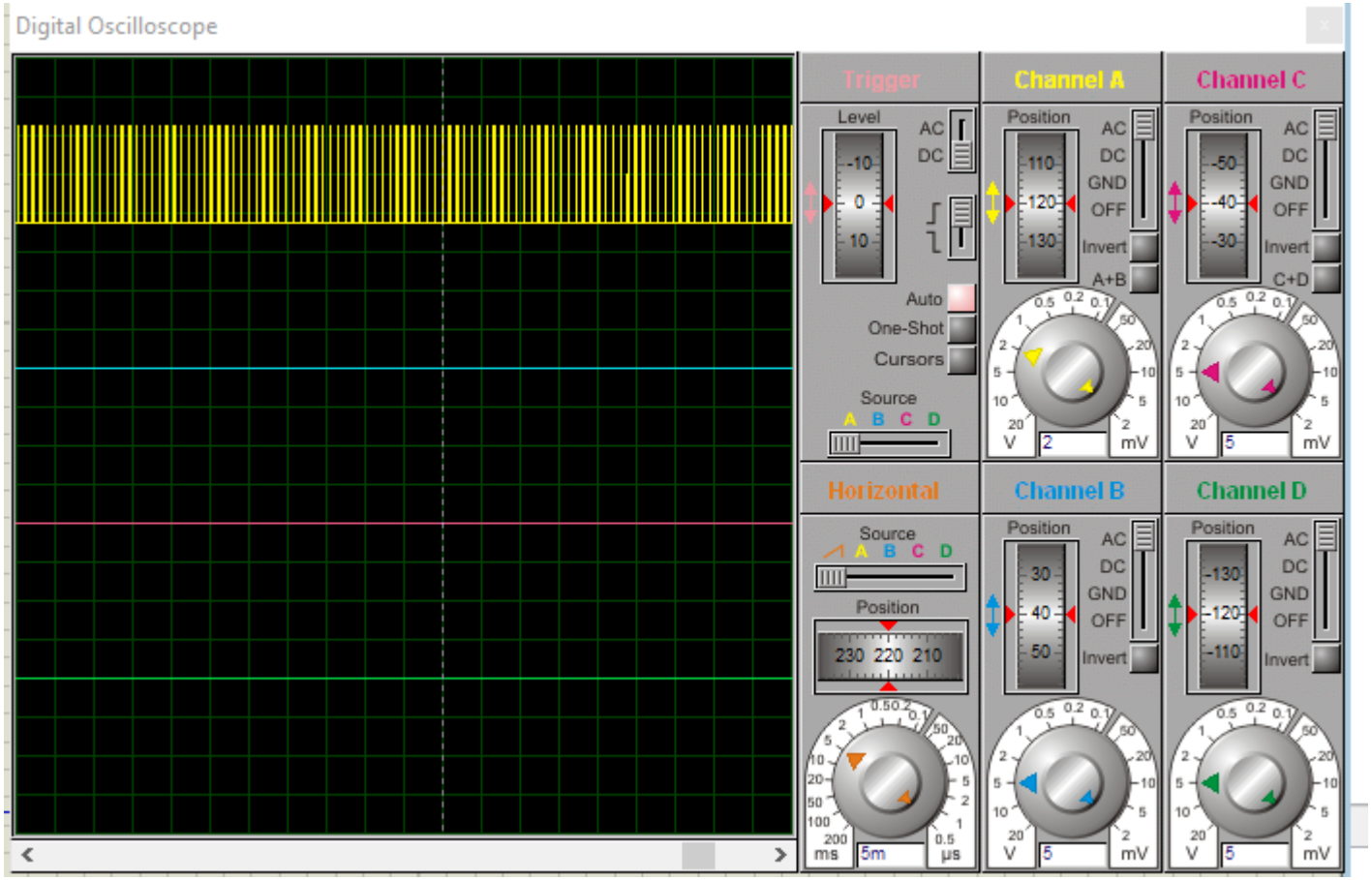
CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.2 Truyền thông nối tiếp

```
;Tao chuong trinh truyen du lieu
org 00h
Laplai:
mov A,'T'
call truyen
mov A,'D'
call truyen
mov A,'H'
call truyen
jmp Laplai
Truyen:
mov scon, #40h ; truyen UART 8 thay dc
mov tmod, #20h; time1 che do 8 tu nap
mov th1, #0FDh; 9600 baud
mov sbuf, A; truyen ky tu B
;(ASCII là 42H  STOP+01000010+START)
setb tr1; kich hoat truyen
jnb ti, $ ; cho co xong
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

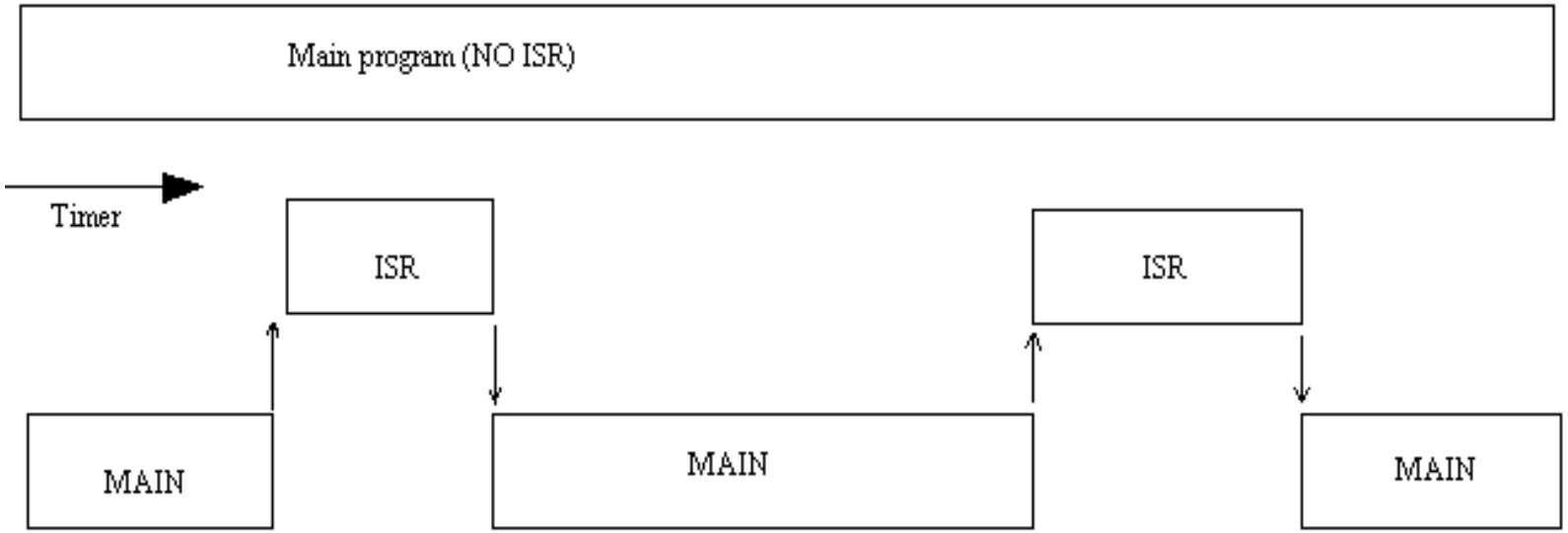
5.2 Truyền thông nối tiếp



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Ngắt

8051 có 5 nguyên nhân ngắt: 2 do bên ngoài, 2 ngắt do bộ định thời và 1 ngắt do port nối tiếp. 8052 có thêm 1 ngắt do bộ định thời thứ 3. Khi thiết lập trạng thái ban đầu cho hệ thống (cấp nguồn, Reset) tất cả các ngắt đều bị vô hiệu hóa và sau đó chúng cho phép riêng rẽ theo chương trình



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Ngắt

8051 có 5 nguyên nhân ngắt: 2 do bên ngoài, 2 ngắt do bộ định thời và 1 ngắt do port nối tiếp. 8052 có thêm 1 ngắt do bộ định thời thứ 3. Khi thiết lập trạng thái ban đầu cho hệ thống (cấp nguồn, Reset)

Bit	Ký hiệu	Địa chỉ	Mô tả
IE.7	EA	AFH	**Cho phép 1/ không cho phép 0 toàn cục**
IE.6	-	AEH	**None**
IE.5	ET2	ADH	**Cho phép ngắt do bộ định thời 2**
IE.4	ES	ACH	**Cho phép ngắt do port nối tiếp**
IE.3	ET1	ABH	**Cho phép ngắt do bộ định thời 1**
IE.2	EX1	AAH	**Ngắt ngoài 1 (P3.3)**
IE.1	ET0	A9H	**Cho phép ngắt do bộ định thời 0**

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Ngắt

Ví dụ Kích hoạt ngắt ngoài 1 thì

EA=1
EX1=1

hoặc

IE.7=1
IE.2=1
tương tự thiết lập các ngắt sau

- Kích hoạt 2 ngắt là timer 0 và truyền thông nối tiếp
- Kích hoạt 3 ngắt là Ngoài 0, timer 0 và truyền thông nối tiếp

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Ngắt

Mỗi 1 nguyên nhân ngắt được lập trình riêng để có 1 trong 2 mức ưu tiên thông qua thanh ghi ưu tiên ngắt IP (interrup priority) có địa chỉ byte 0B8H

Bit	Ký hiệu	Địa chỉ bit	Mô tả (1 mức cao, 0 thấp)
IP.7	-	-	**Không sử dụng**
IP.6	-	-	**Không sử dụng**
IP.5	PT2	0BDH	**Ưu tiên cho ngắt do bộ định thời 2**
IP.4	PS	0BCH	**Ưu tiên cho ngắt do port nối tiếp**
IP.3	PT1	0BBH	**Ưu tiên cho ngắt do bộ định thời 1**
IP.2	PX1	0BAH	**Ngắt ngoài 1**
IP.1	PT0	0B9H	**Ưu tiên cho ngắt do bộ định thời 0**
IP.0	**PX0**	**0B8H**	**Ngắt ngoài 0**

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Ngắt

Mỗi 1 nguyên nhân ngắt được lập trình riêng để có 1 trong 2 mức ưu tiên thông qua thanh ghi ưu tiên ngắt IP (interrup priority) có địa chỉ byte 0B8H

Ngắt	Cờ	Thanh ghi SFR và vị trí bit
Ngắt ngoài 0	IE0	TCON.1
Ngắt ngoài 1	IE1	TCON.3
Ngắt do bộ định thời 1	TF1	TCON.7
Ngắt do bộ định thời 0	TF0	TCON.5
Ngắt do port nối tiếp	TI	SCON.1
Ngắt do port nối tiếp	RI	SCON.0
Ngắt do bộ định thời 2	TF2	T2CON.7 (8052)
Ngắt do bộ định thời 2	EXF2	T2CON.6 (8052)

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Ngắt

Vecto reset hệ thống (RST ở địa chỉ 0000H) được chứa trong bảng này vì vậy cũng được xem như 1 ngắt: chương trình chính bị ngắt và PC được nạp giá trị mới

Ngắt do	Cờ	Địa chỉ vecto
Reset hệ thống	RST	0000H
Ngắt ngoài 0	IE0	0003H
Bộ định thời 0	TF0	000BH
Ngắt ngoài 1	IE1	0013H
Bộ định thời 1	TF1	001BH
Port nối tiếp	RI hoặc TI	0023H
Bộ định thời 2	TF2 hoặc EXF2	002BH

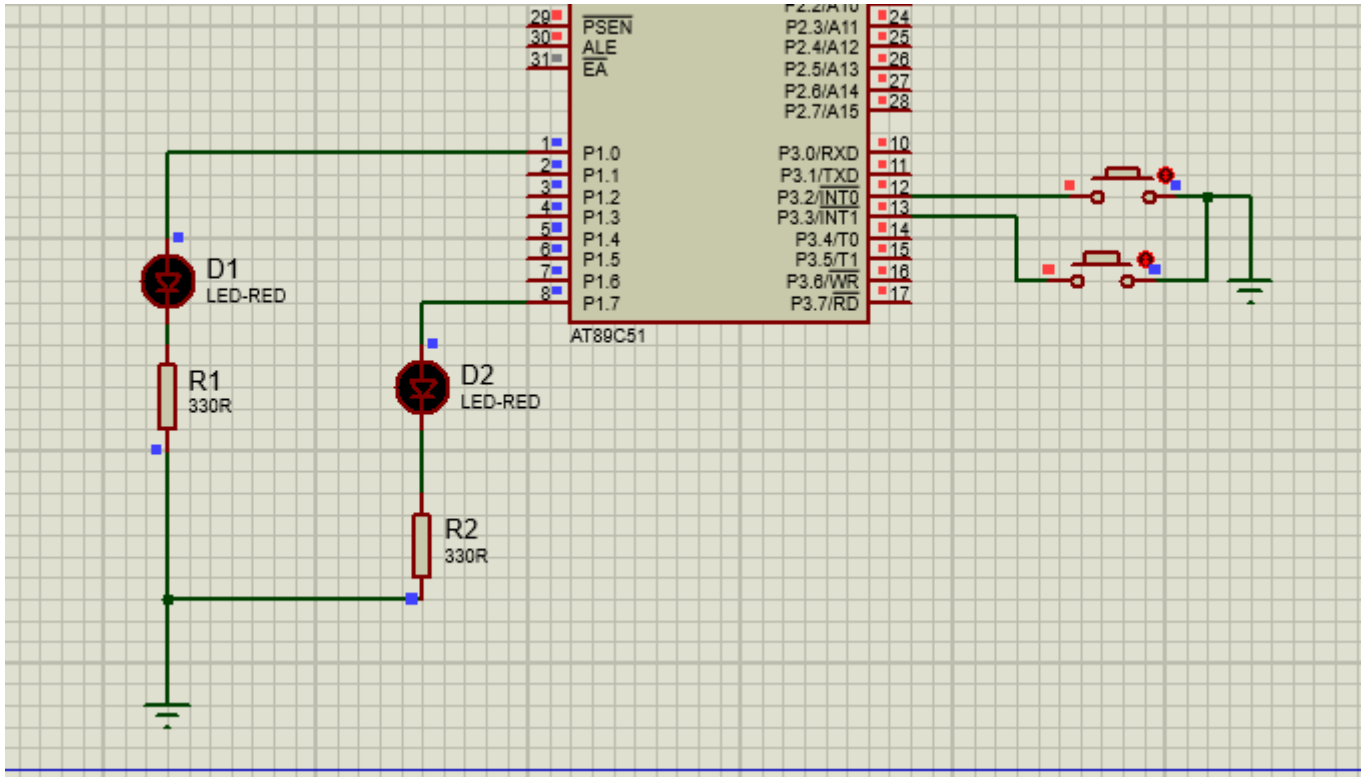
CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Lập trình ngắt

```
1  org 03h
2  setb P3.0
3
4  org 30h ; Kich hoat ngat ngoài 0
5  mov p3,#00h
6  mov IE, #10000001b
7
8  end
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Lập trình ngắt



CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Lập trình ngắt

```
1   org 00h
2   jmp CTchinh
3   org 03h
4   TenCTngat0:
5   setb P1.0
6   reti
7   org 13h
8   TenCTngat1:
9   clr P1.0
10  reti
```

```
12  org 30h ; Kich hoat ngat ngoài 0, 1
13  CTchinh:
14  mov p1,#00h
15  setb EA
16  setb EX0
17  setb ex1
18  LL:
19  Setb P1.7
20  call delay
21  clr p1.7
22  call delay
23  jmp LL
```

```
25  delay:
26  mov tmod, #01h
27  mov R0,#20
28  V1:
29  mov th0,#3Bh ;high(-50000)
30  mov tl0,#0FCh ; low(-50000)
31  setb tr0
32  jnb tf0,$
33  clr tf0
34  clr tr0
35  djnz R0, V1
36  ret
37  end
```

CHƯƠNG 5 CHỨC NĂNG MỞ RỘNG MCS51

5.3 Lập trình ngắt

```
1   ; sáng tat LED có ngat
2   org 00h
3   jmp main
4   org 1Bh
5   ct1:
6   clr tf1
7   cpl p1.0
8   mov r1,#200
9   v2:
10  mov r2, #200
11  djnz r2,$
12  djnz r1, v2
13  reti
```

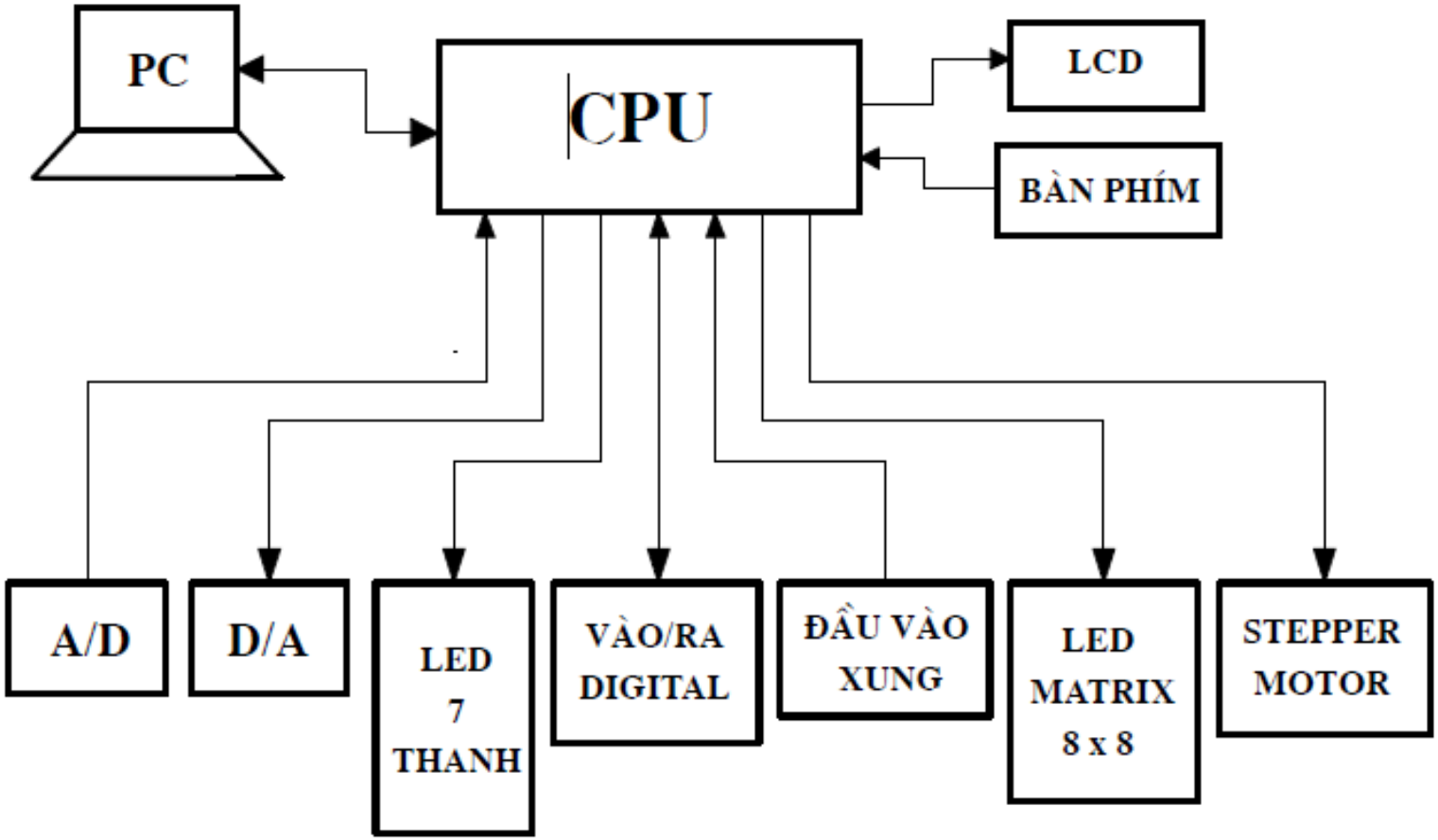
```
15  org 05h
16  main:
17  setb EA
18  setb et1
19  mov tmod, #20h
20  mov th1,#56
21  setb tr1
22  jmp $
23
24  end
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít
6.2. Giao tiếp mã quét
6.3. Giao tiếp dữ liệu
6.4. Giao tiếp truyền thông

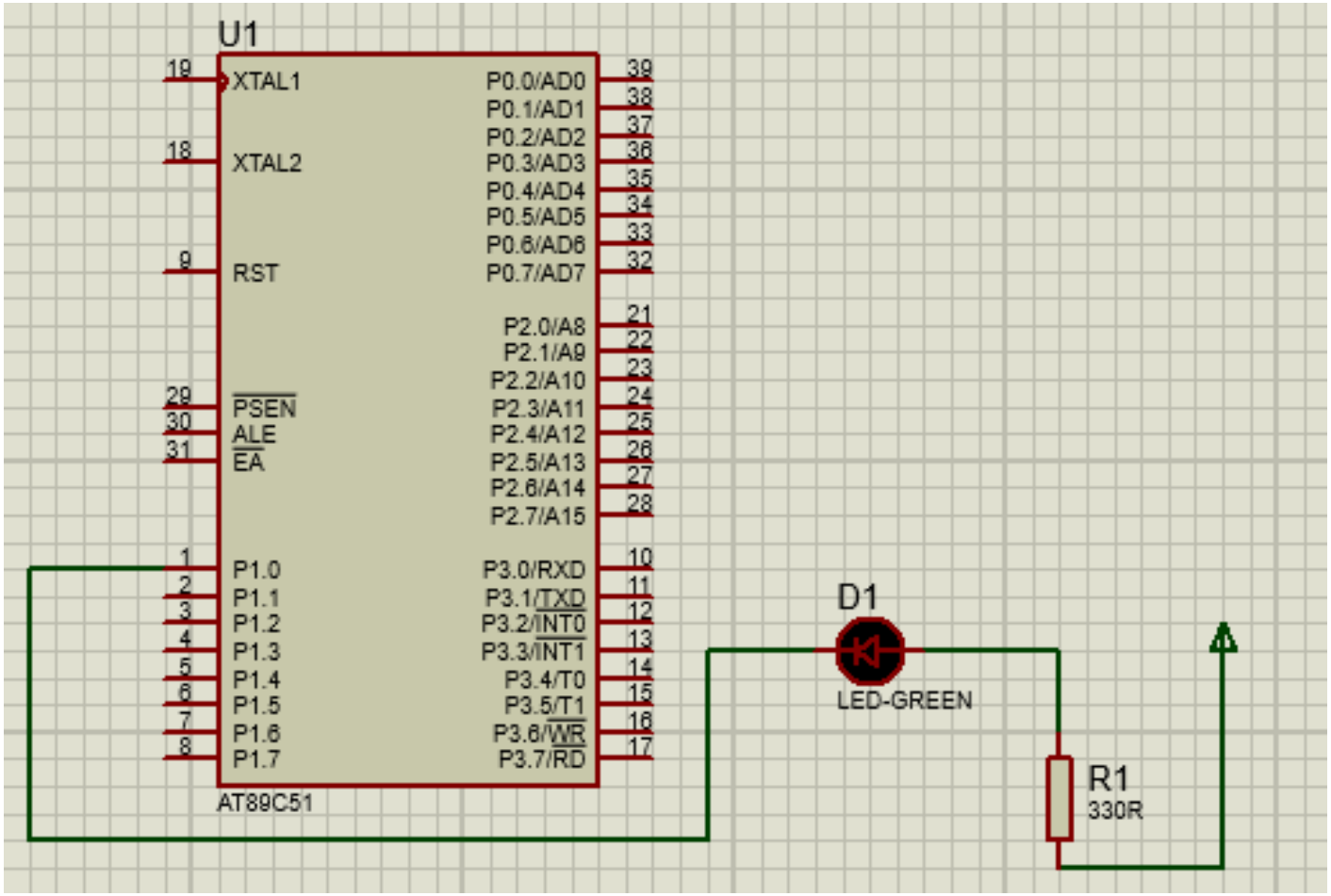
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



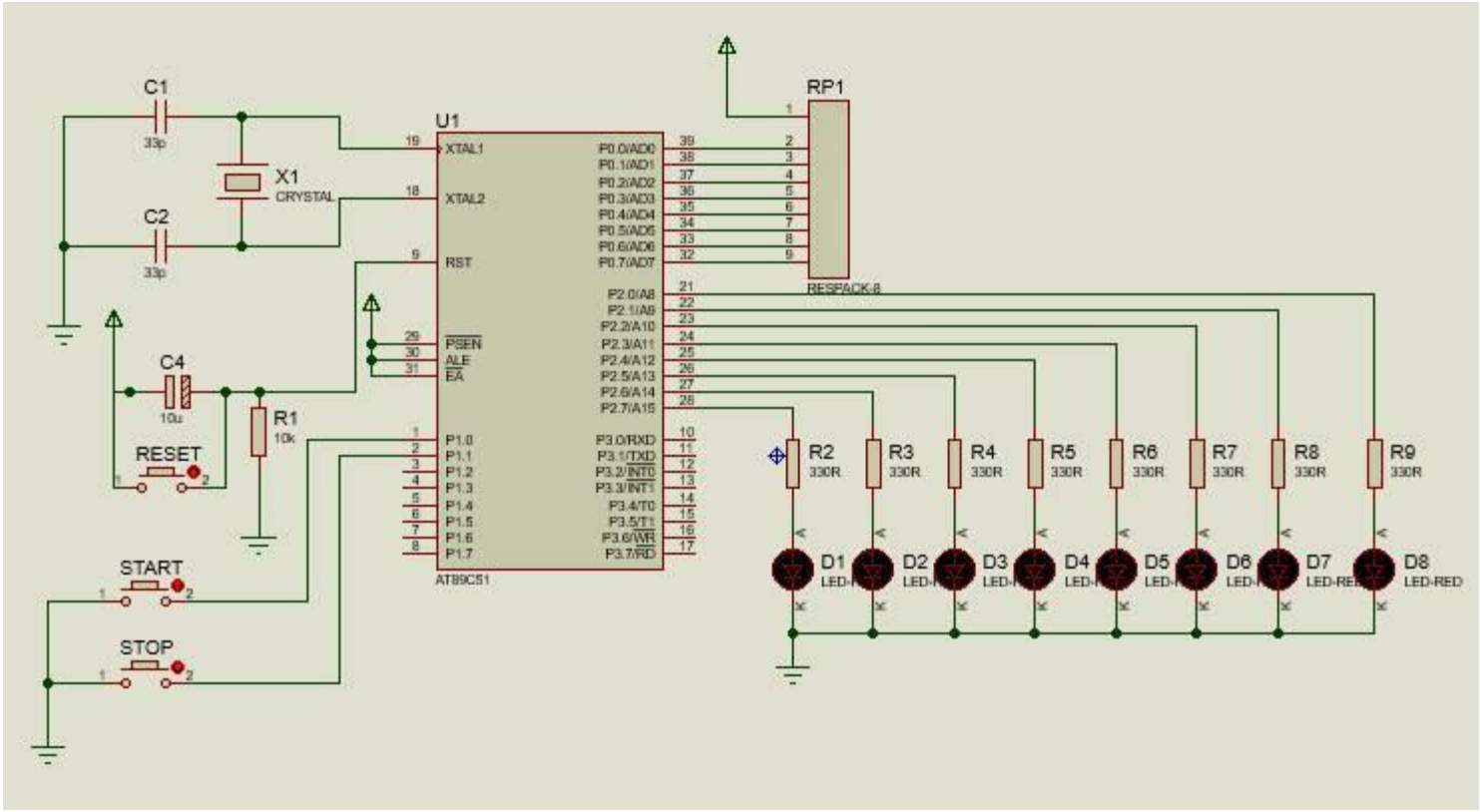
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



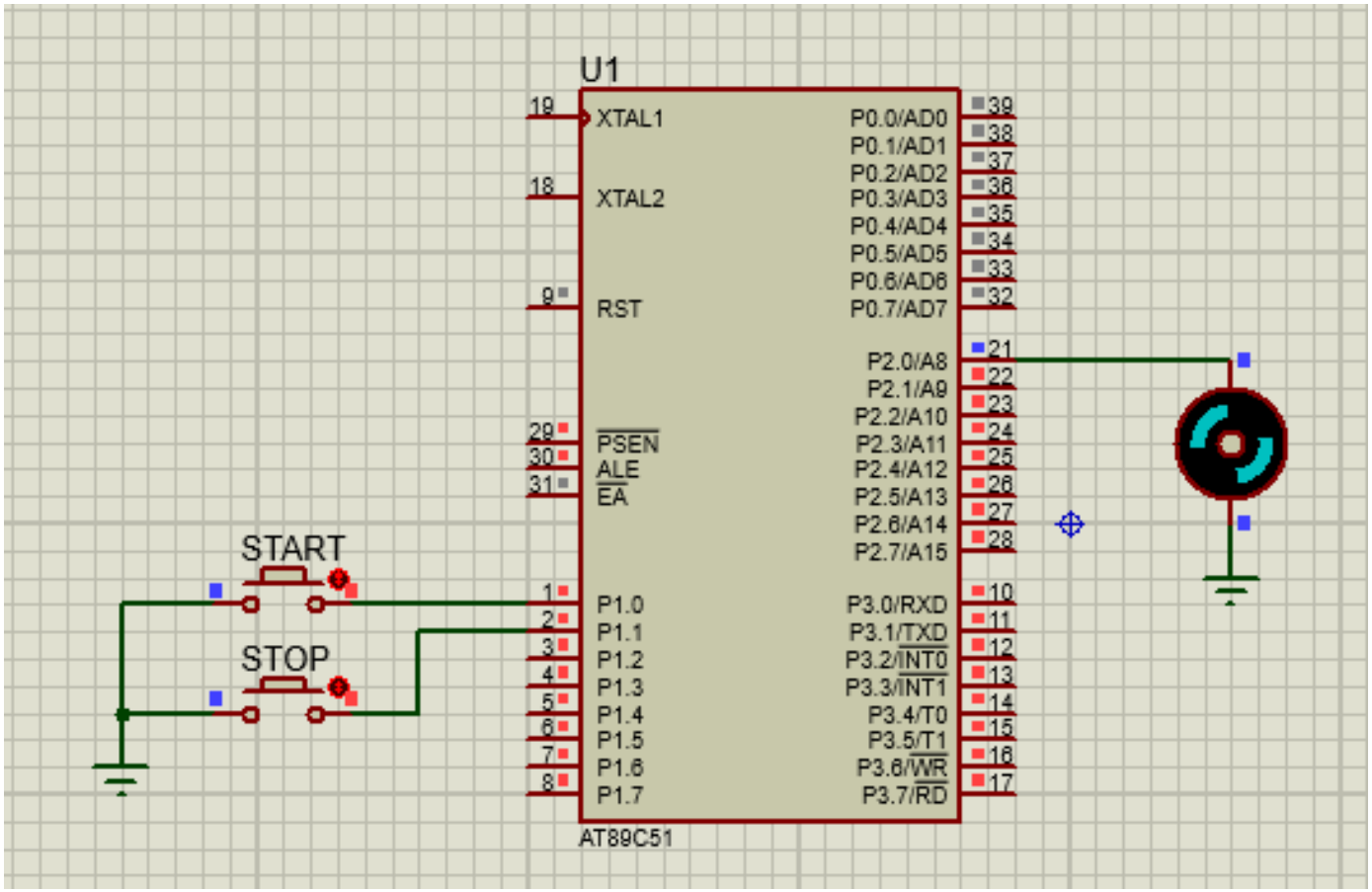
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



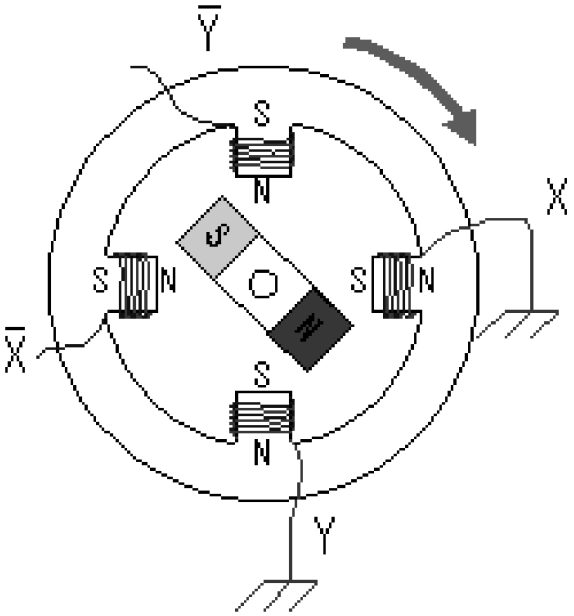
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

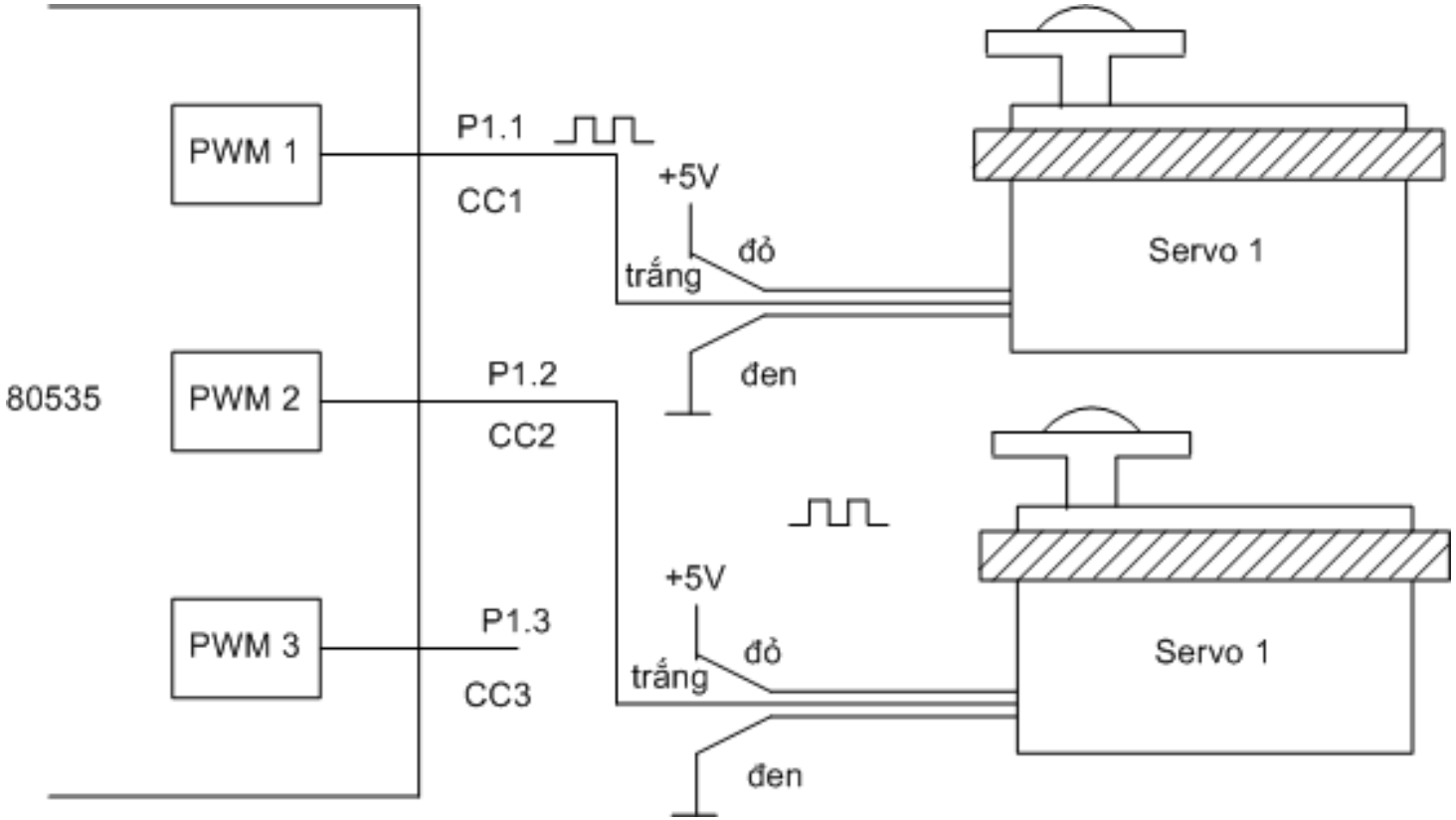
6.1 Giao tiếp vào ra đơn bít



X	X̄	Y	Ȳ
0	1	0	1
1	0	0	1
1	0	1	0
0	1	1	0

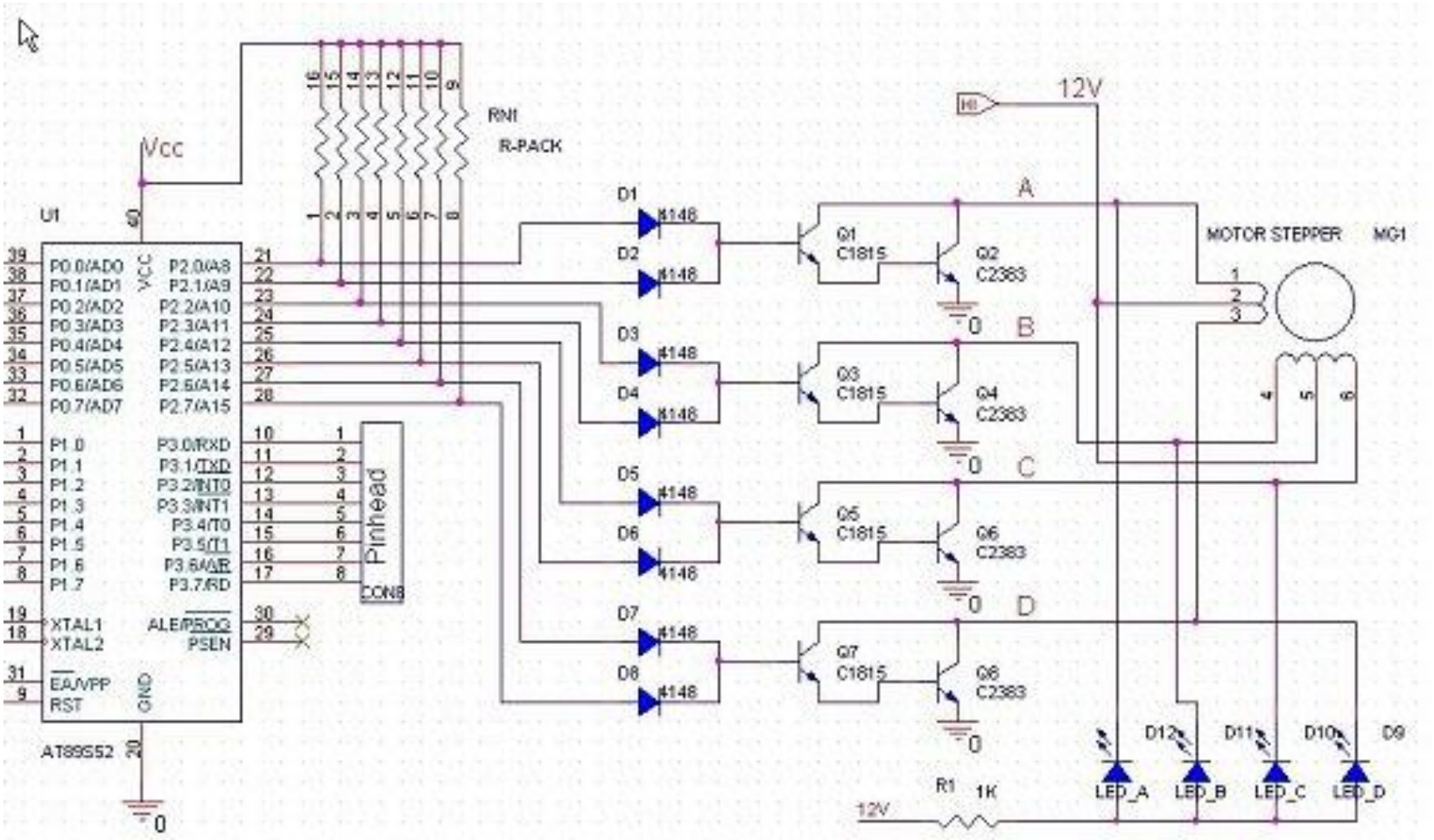
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



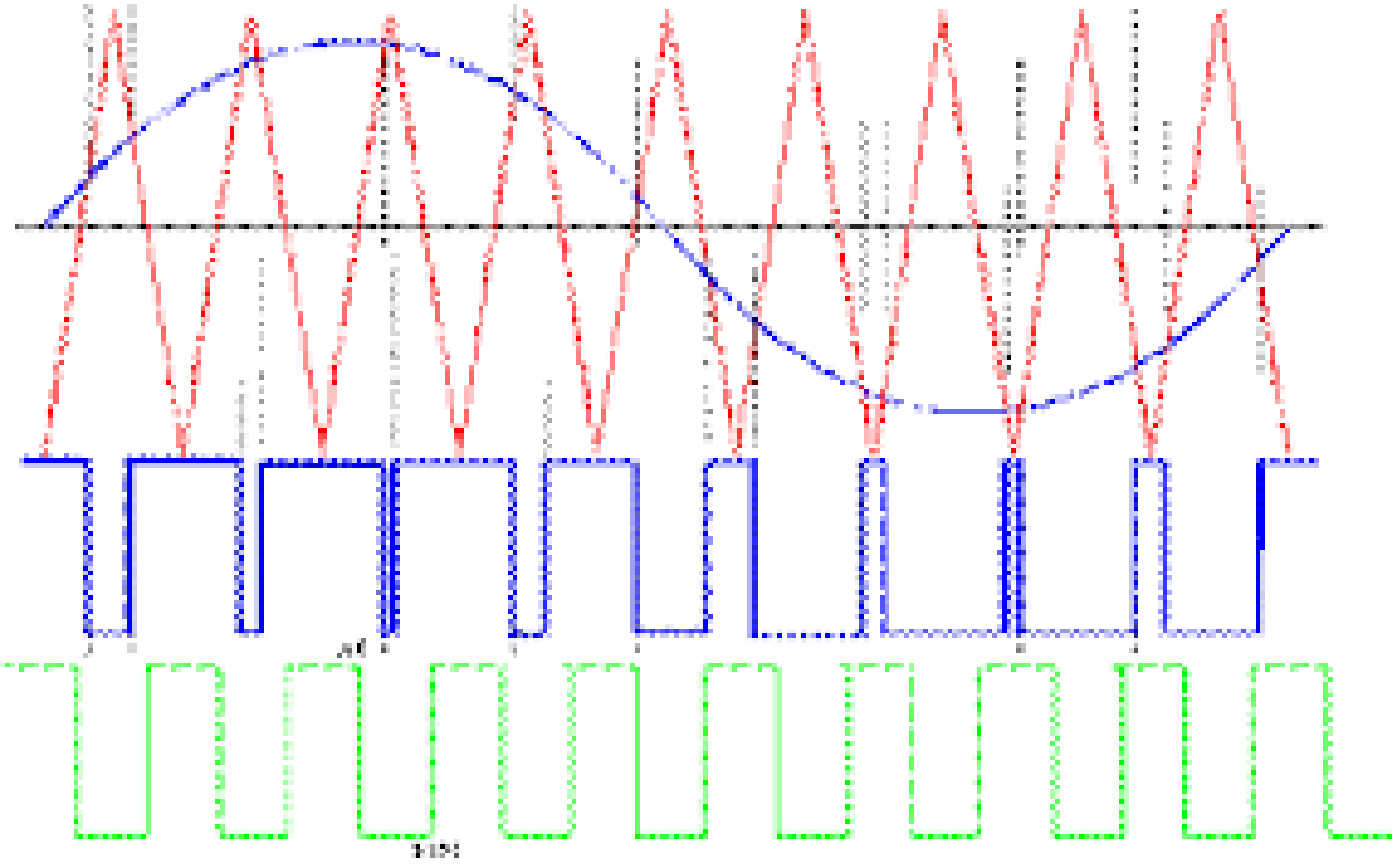
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít

```
1   org 00h
2   mov p2,#00
3   LL:
4   mov p2,#00000001b
5   call delay
6   mov p2,#00000010b
7   call delay
8   mov p2,#00000100b
9   call delay
10  mov p2,#00001000b
11  call delay
12  jmp LL
13  delay:
14  mov tmod, #01h
15  mov th0,#0
16  mov tl0,#0
17  setb tr0
18  jnb tf0, $
```

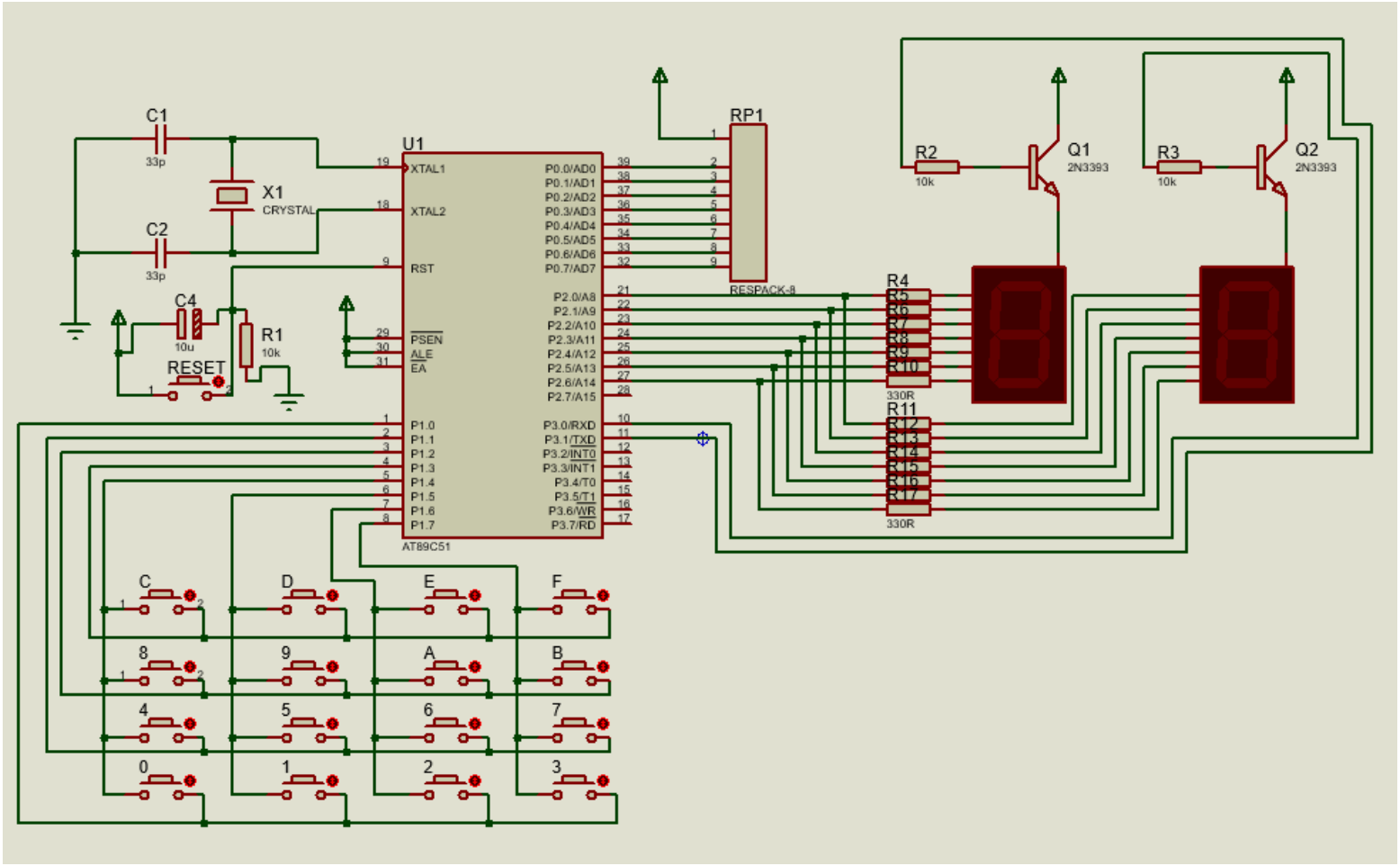
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.1 Giao tiếp vào ra đơn bít



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét

```
org 0h
LL:
mov p2,#0C0H ;0
call delay
mov p2,#0F9H ;1
call delay
mov p2,#0A4H;2
call delay
mov p2,#0B0H;3
call delay
mov p2,#99H;4
call delay
mov p2,#92H;5
call delay
mov p2,#82H;6
call delay
```

```
mov p2,#90H;9
call delay
jmp LL

delay:
mov TMOD, #10H ; timer
mov R0,#20
V1:
mov TH1, #high(-50000)
mov TL1, #low(-50000)
setb TR1
jnb TF1,$
clr TR1
clr TF1
djnz R0, V1
ret
end
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét

```
1   org 00h
2   mov dptr, #maled ; giai ma chi chi ROM
3   ct_phu:
4   mov r7, #0
5   ct:
6   mov A, r7
7   movc A, @A+dptr ; cho dia chi bo nho ROM
8   mov p2, A
9   call delay
10  inc r7
11  cjne r7, #10, ct
12  jmp ct_phu
13
14  delay:
15  mov r1, #10
16  V1:
17  mov tmod, #01h
18  mov th0, #3Ch
```

Line : 6 | Column : 10 | Insert | .asm | MCU 8051

```
14  delay:
15  mov r1, #10
16  V1:
17  mov tmod, #01h
18  mov th0, #3Ch
19  mov tl0, #0AFh
20  setb tr0
21  jnb tf0, $
22  clr tr0
23  clr tf0
24  djnz r1, V1
25  ret
26
27  maled:;
28      db 0C0h, 0F9h,
29  end
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét

```
1   org 00h
2   mov p3,#00
3
4   mov p2,#0F9h; 1
5   setb p3.2
6   clr p3.1
7   call delay
8   mov p2,#0C0h; 1
9   setb p3.1
10  clr p3.2
11  call delay
12
13
14
15  delay:
```

```
14  delay:
15  mov r1, #10
16  V1:
17  mov tmod, #01h
18  mov th0, #3Ch
19  mov tl0, #0AFh
20  setb tr0
21  jnb tf0, $
22  clr tr0
23  clr tf0
24  djnz r1, V1
25  ret
26
27  maled:;
28      db 0C0h, 0F9h,
29  end
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét

```
1   org 00h
2   mov DPTR, #maled
3   ct_chinh:
4   mov r0, #0 ; R0=xy
5
```

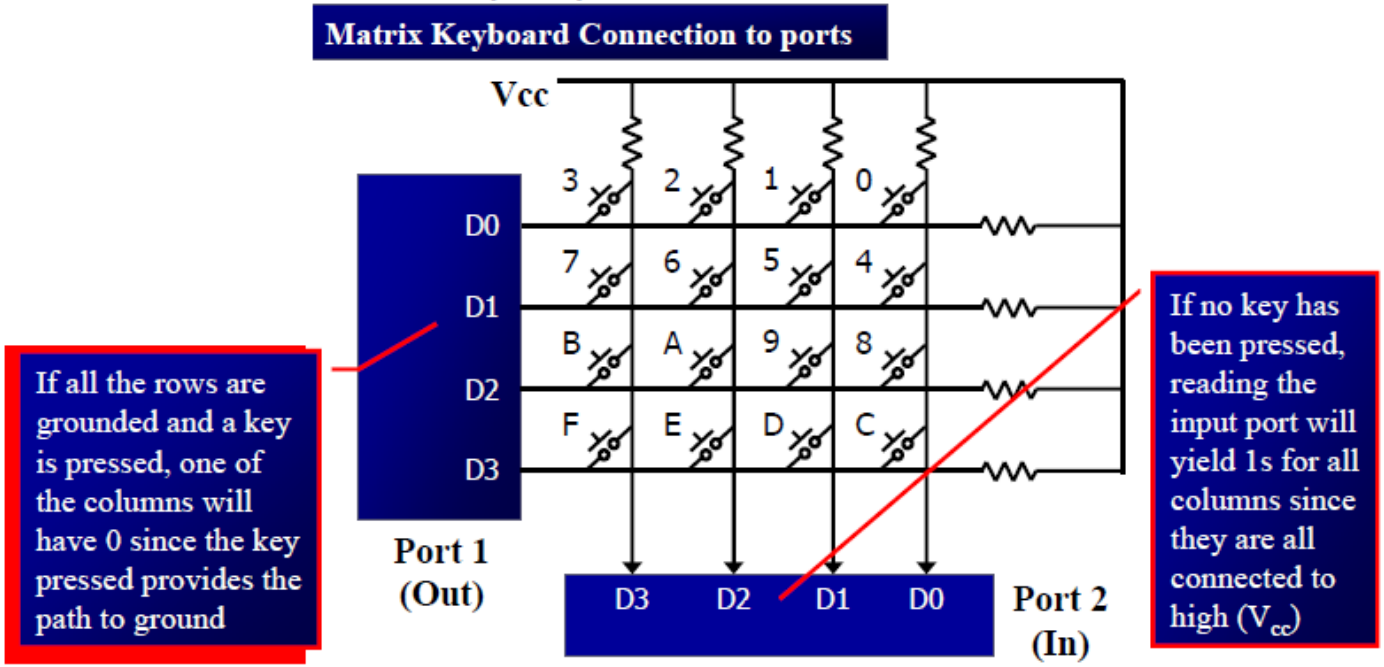
```
25  inc r0
26  cjne r0, #100, main
27  jmp ct_chinh
28
29
30  delay:
31  mov TMOD, #01h
32  mov TH0, #high(-1000(
33  mov TL0, #low(-10000)
34  setb TR0
35  jnb TF0, $
36  clr TR0
37  clr TF0
38  ret
39  org 1000
40  maled:
41      db 0C0h, 0F9h, 0A
42  end
```

```
6   main:
7   mov R1, #100
8   LL:
9   mov A, r0
10  mov B, #10
11  DIV AB ; A=x, B=y
12  setb p3.1 ; hien so hang chuc
13  movc A, @A+DPTR
14  mov p2, A
15  call delay
16  clr p3.1; tat so hang chuc
17  mov A, B
18  setb p3.0 ;hien so hang dv
19  movc A, @A+DPTR
20  mov p2, A
21  call delay
22  clr p3.0 ;tat so hang dv
23  Djnz r1, LL
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét

- A 4x4 matrix connected to two ports
  - The rows are connected to an output port and the columns are connected to an input port



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét bàn phím matrix

```
ORG 00H
LL:
MOV P1,#0FFH
CLR P1.0
MOV A,P1
CJNE A,#11101110B,TIEP; Cot 1
MOV P2,#0C0H
TIEP:
CJNE A,#11011110B,TIEP1 ;cot 2
MOV P2,#0F9H
TIEP1:
CJNE A,#10111110B,TIEP2
MOV P2,#0A4H
TIEP2:
CJNE A,#01111110B,TIEP3
MOV P2,#0B0H
TIEP3:
MOV P1,#0FFH
CLR P1.1
MOV A,P1
CJNE A,#11101101B,TIEP4
MOV P2,#99H
TIEP4:
JMP LL
END
```

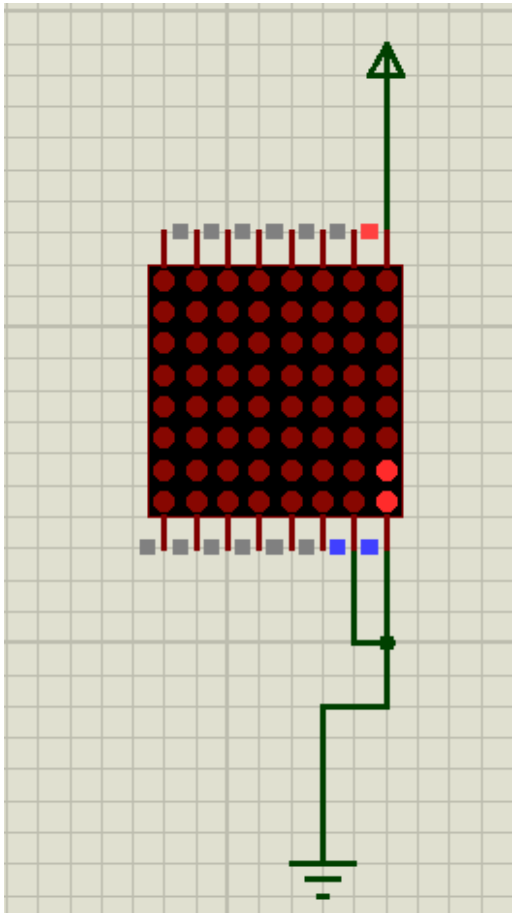
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét bàn phím matrix

```
1  ORG 00H
2  MOV DPTR,#MALED
3  SO0:
4  MOV P1,#11111111B
5  CLR P1.0
6  JB P1.4,SO1
7  MOV A,#0
8  CALL HIENTHI
9  SO1:
10 JB P1.5,SO2
11 MOV A,#1
12 CALL HIENTHI
13 SO2:
14 JB P1.6,SO3
15 MOV A,#2
16 CALL HIENTHI
17 SO3:
18 JB P1.7,SO4
19 MOV A,#3
20 CALL HIENTHI
21 SO4:
22 SETB P1.0
23 CLR P1.1
24 JB P1.4,SO5
25 MOV A,#4
26 CALL HIENTHI
27 SO5:
28 JB P1.5,SO6
29 MOV A,#5
30 CALL HIENTHI
31 SO6:
32 JB P1.6,SO7
33 MOV A,#6
34 CALL HIENTHI
35 SO7:
36 JB P1.7,SO8
37 MOV A,#7
38 CALL HIENTHI
39 SO8:
40 SETB P1.1
41 CLR P1.2
42 JB P1.4,SO9
43 MOV A,#8
44 CALL HIENTHI
45 SO9:
46 JB P1.5,so0
47 MOV A,#9
48 CALL HIENTHI
49 JMP SO0
50
51 HIENTHI:
52 MOVC A,@A+DPTR
53 MOV P2,A
54 RET
55
56 MALED:
57 DB 0C0h, 0F9h,
   0A4h, 0B0h, 099h,
   092h, 082h, 0F8h,
   080h, 090h
58 END
```

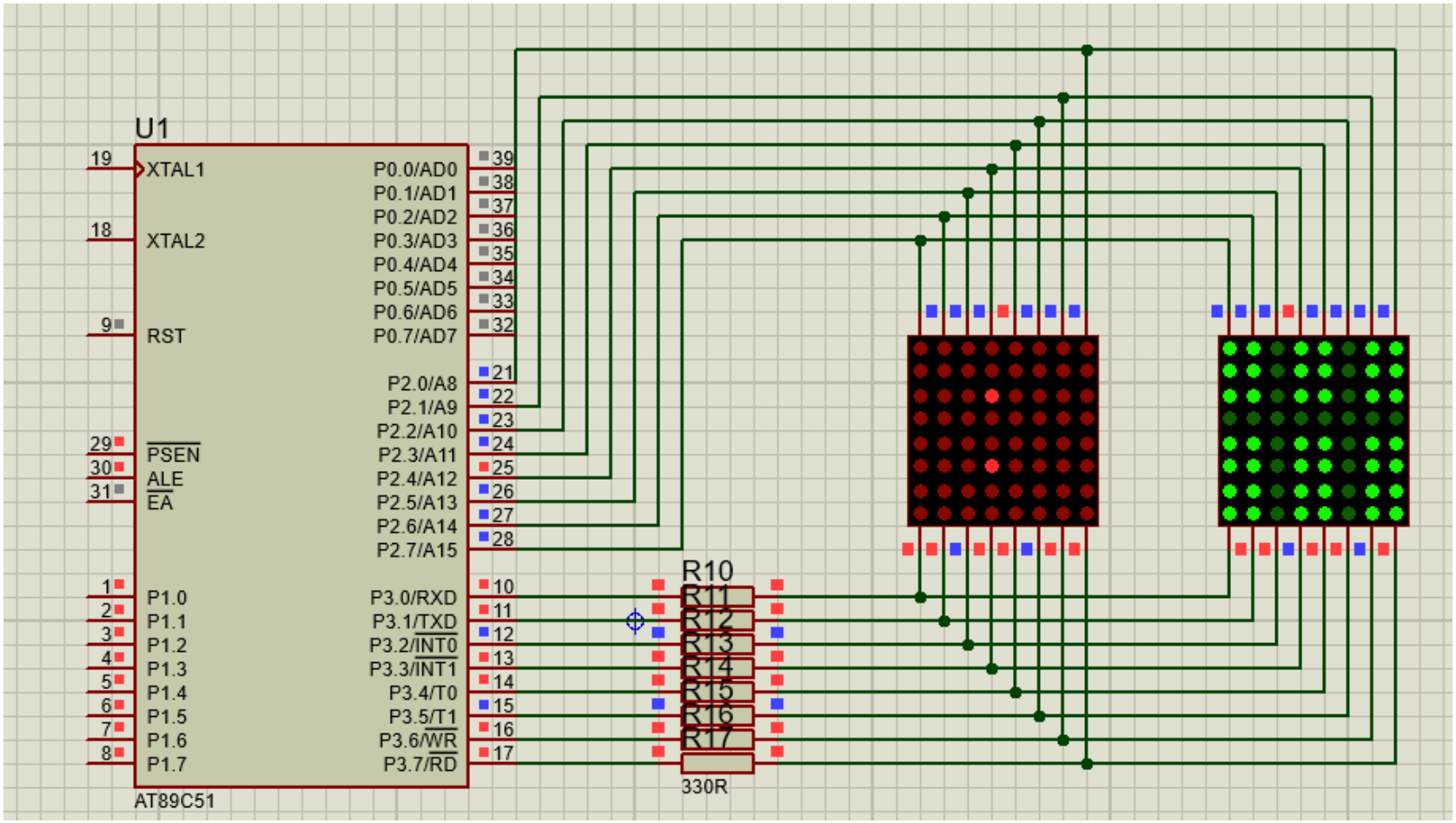
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét



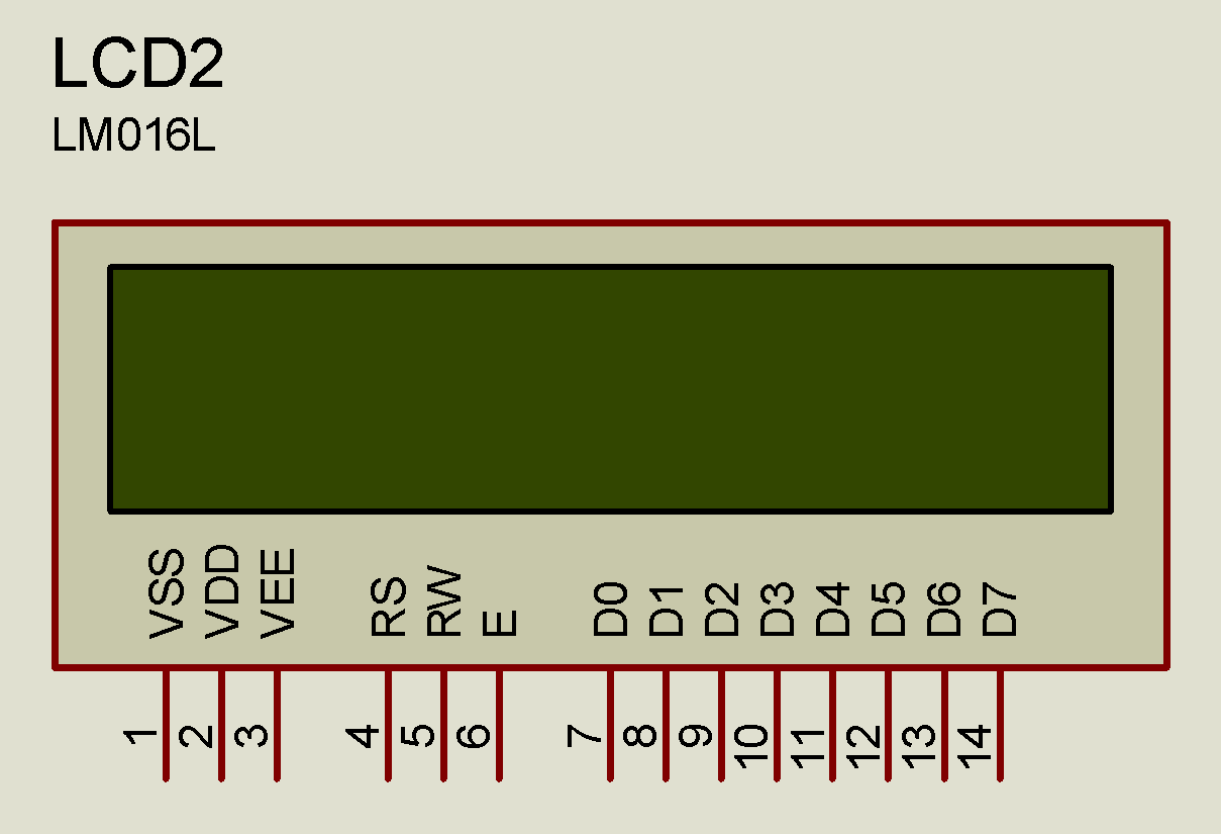
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.2. Giao tiếp mã quét



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

Màn tinh thể lỏng LCD

Màn hình tinh thể lỏng LCD 02_16

Chân RS: “Register Select” =1/0 □ hiển thị dữ liệu/tín hiệu đk

Chân R/W: “Read/Write” =1/0 □ đọc/ghi lên LCD

Chân E: “Enable” =1/0 □ cho phép/chốt dữ liệu lên LCD

Màn hình tinh thể lỏng LCD 08_24

Tương tự như LCD 04_16 có 2 chân E1, E2 dùng nửa trên và nửa dưới của màn hình.

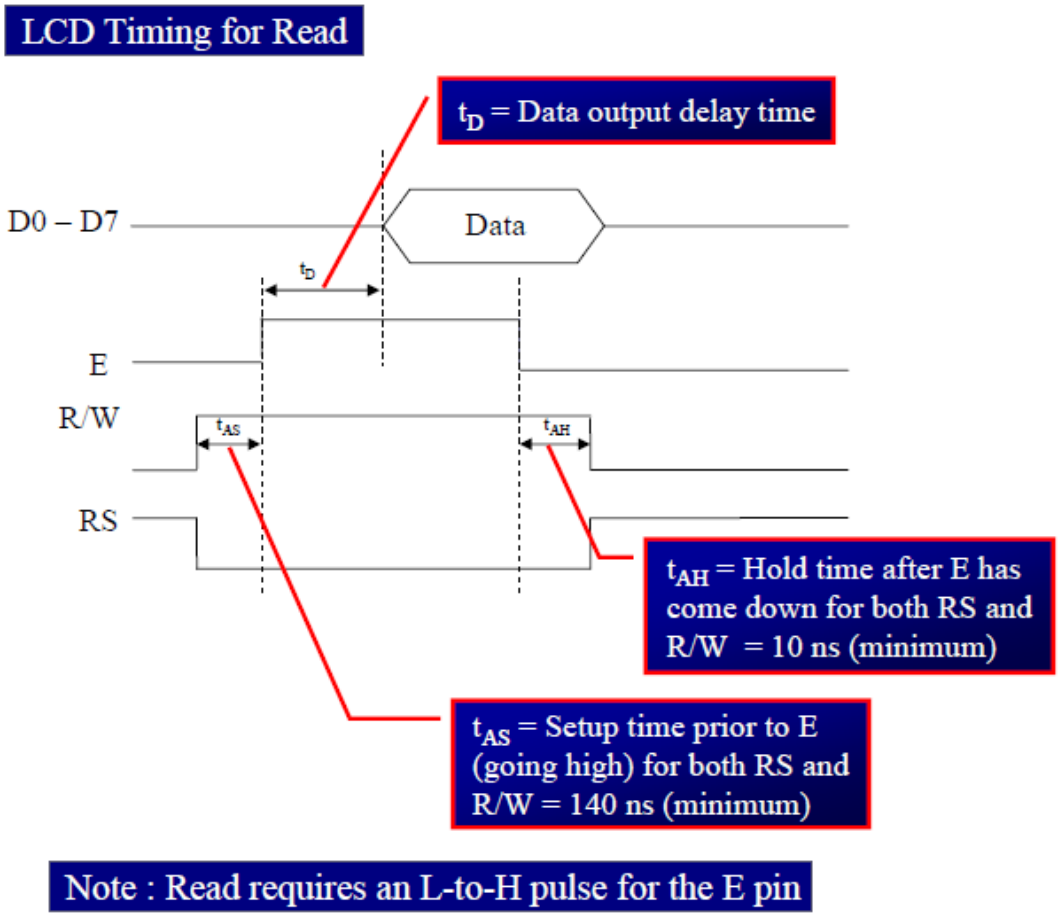
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

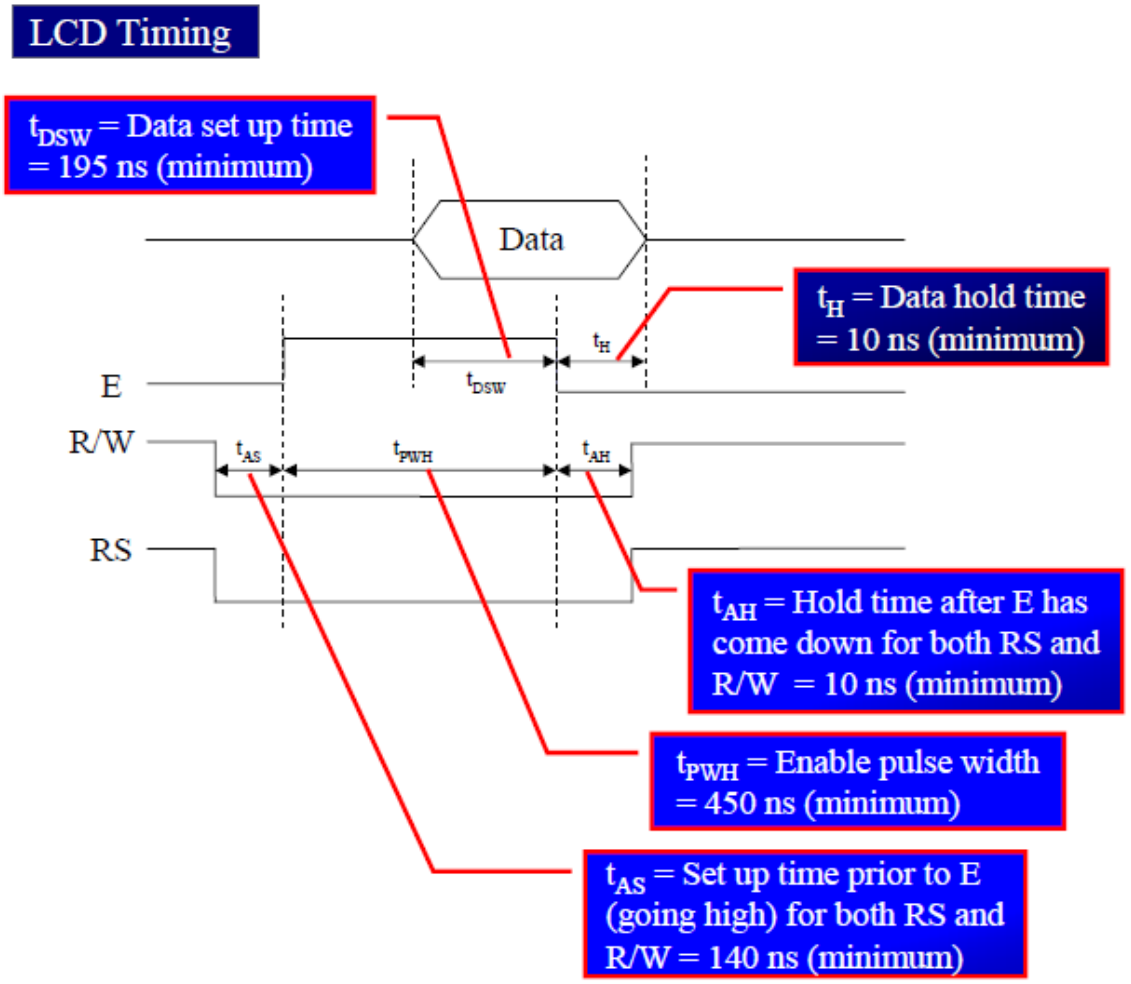
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



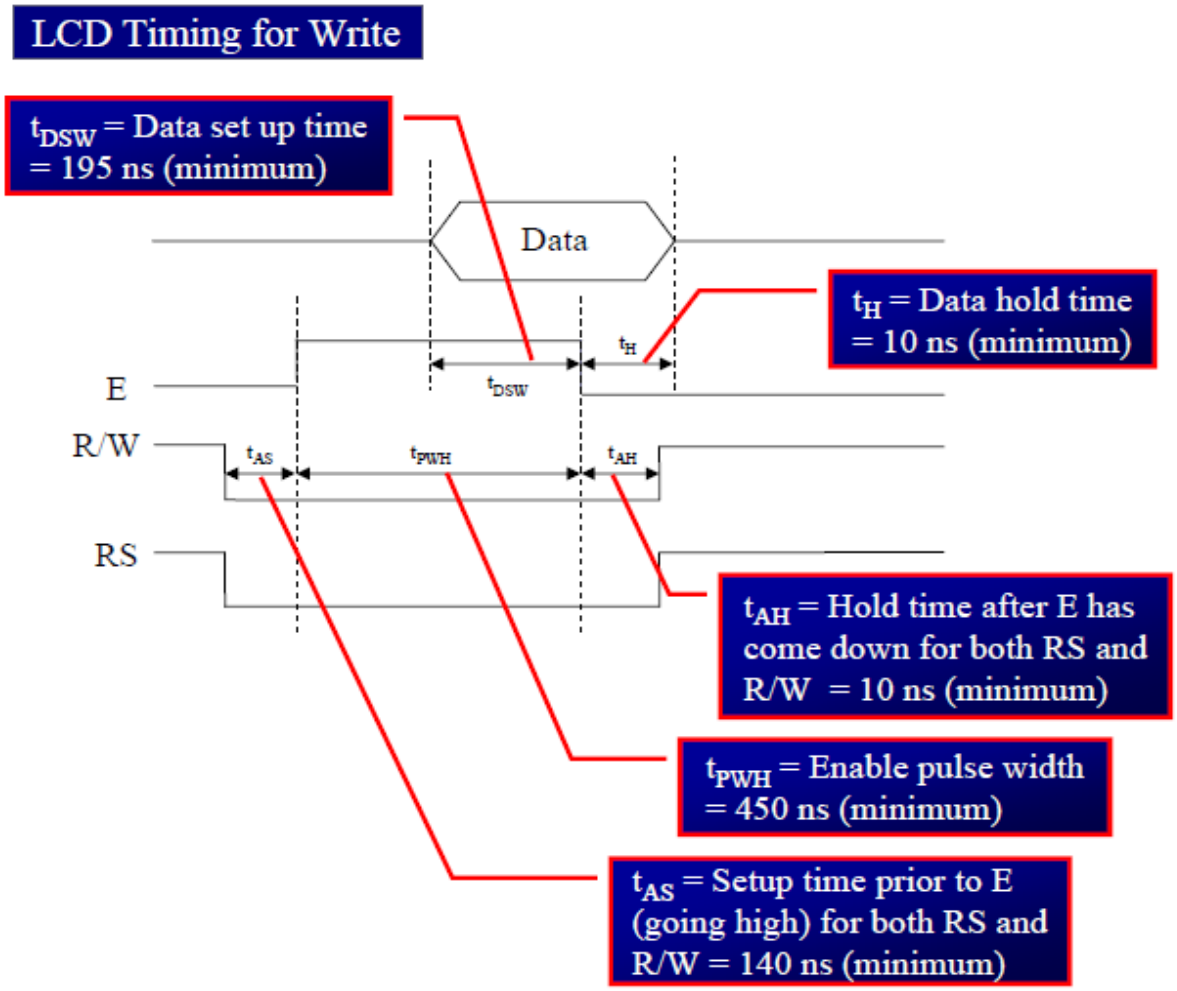
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



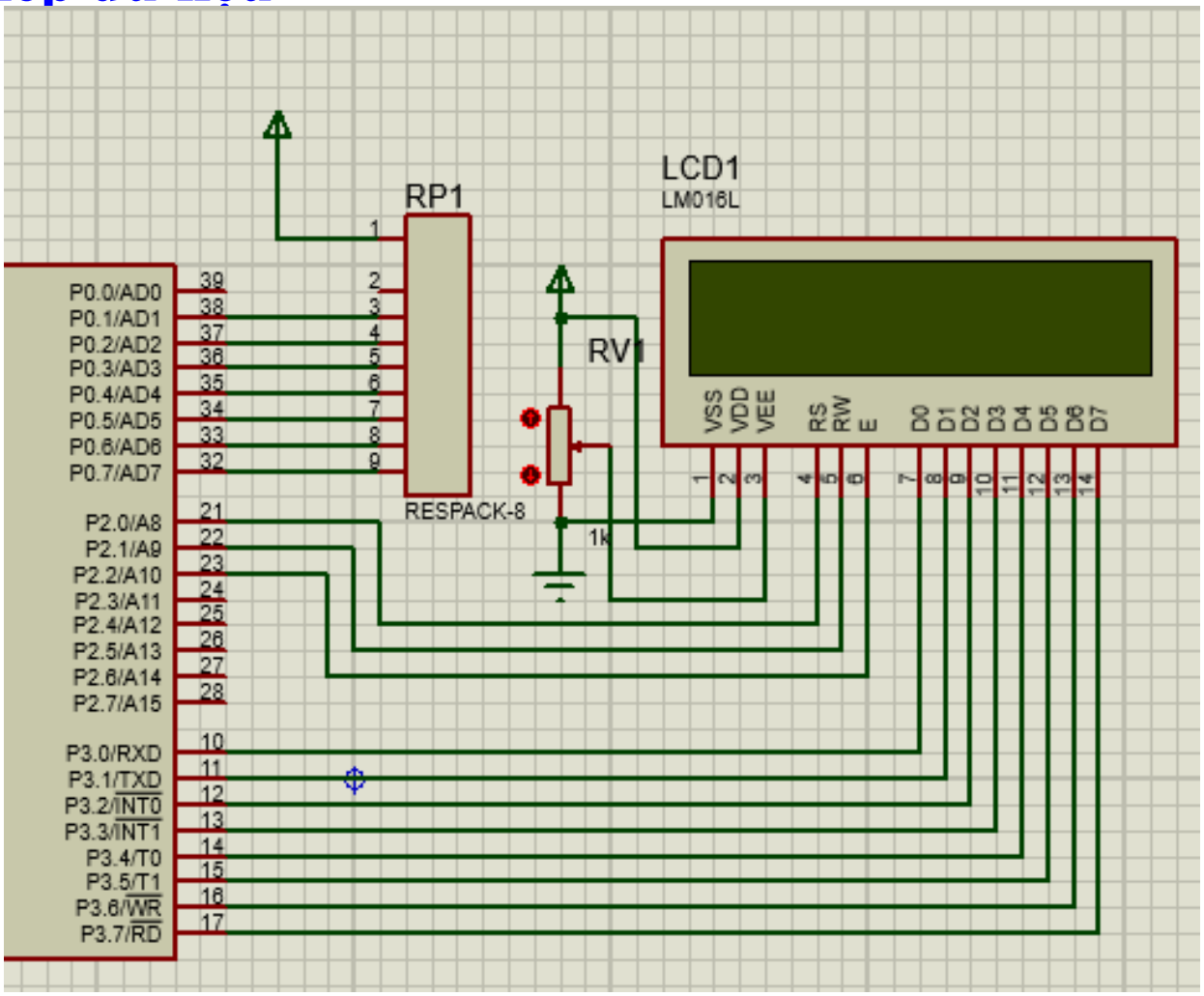
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

Dec	Hx	Char	Dec	Hx	HTML	Char	Dec	Hx	HTML	Char	Dec	Hx	HTML	Char
0	0	NUL (null)	32	20	&#32;	Space	64	40	&#64;	@	96	60	&#96;	`
1	1	SOH (Start of heading)	33	21	&#33;	!	65	41	&#65;	A	97	61	&#97;	a
2	2	STX (Start of text)	34	22	&#34;	"	66	42	&#66;	B	98	62	&#98;	b
3	3	ETX (End of text)	35	23	&#35;	#	67	43	&#67;	C	99	63	&#99;	c
4	4	EOT (End of transmission)	36	24	&#36;	$	68	44	&#68;	D	100	64	&#100;	d
5	5	ENQ (Enquiry)	37	25	&#37;	%	69	45	&#69;	E	101	65	&#101;	e
6	6	ACK (Acknowledge)	38	26	&#38;	&	70	46	&#70;	F	102	66	&#102;	f
7	7	BEL (Bell)	39	27	&#39;	'	71	47	&#71;	G	103	67	&#103;	g
8	8	BS (Backspace)	40	28	&#40;	(	72	48	&#72;	H	104	68	&#104;	h
9	9	TAB (Horizontal tab)	41	29	&#41;	)	73	49	&#73;	I	105	69	&#105;	i
10	A	LF (NL line fd, new line)	42	2A	&#42;	*	74	4A	&#74;	J	106	6A	&#106;	j
11	B	VT (Vertical tab)	43	2B	&#43;	+	75	4B	&#75;	K	107	6B	&#107;	k
12	C	FF (NP form fd, new page)	44	2C	&#44;	,	76	4C	&#76;	L	108	6C	&#108;	l
13	D	CR (Carriage return)	45	2D	&#45;	-	77	4D	&#77;	M	109	6D	&#109;	m
14	E	SO (Shift out)	46	2E	&#46;	.	78	4E	&#78;	N	110	6E	&#110;	n
15	F	SI (Shift in)	47	2F	&#47;	/	79	4F	&#79;	O	111	6F	&#111;	o
16	10	DLE (Data link escape)	48	30	&#48;	0	80	50	&#80;	P	112	70	&#112;	p
17	11	DC1 (Device control 1)	49	31	&#49;	1	81	51	&#81;	Q	113	71	&#113;	q
18	12	DC2 (Device control 2)	50	32	&#50;	2	82	52	&#82;	R	114	72	&#114;	r
19	13	DC3 (Device control 3)	51	33	&#51;	3	83	53	&#83;	S	115	73	&#115;	s
20	14	DC4 (Device control 4)	52	34	&#52;	4	84	54	&#84;	T	116	74	&#116;	t
21	15	NAK (Negative acknowledge)	53	35	&#53;	5	85	55	&#85;	U	117	75	&#117;	u
22	16	SYN (Synchronous idle)	54	36	&#54;	6	86	56	&#86;	V	118	76	&#118;	v
23	17	ETB (End of trans. block)	55	37	&#55;	7	87	57	&#87;	W	119	77	&#119;	w
24	18	CAN (Cancel)	56	38	&#56;	8	88	58	&#88;	X	120	78	&#120;	x
25	19	EM (End of medium)	57	39	&#57;	9	89	59	&#89;	Y	121	79	&#121;	y
26	1A	SUB (Substitute)	58	3A	&#58;	:	90	5A	&#90;	Z	122	7A	&#122;	z
27	1B	ESC (Escape)	59	3B	&#59;	;	91	5B	&#91;	[	123	7B	&#123;	{
28	1C	FS (File separator)	60	3C	&#60;	<	92	5C	&#92;	\	124	7C	&#124;	|
29	1D	GS (Group separator)	61	3D	&#61;	=	93	5D	&#93;	]	125	7D	&#125;	}
30	1E	RS (Record separator)	62	3E	&#62;	>	94	5E	&#94;	^	126	7E	&#126;	~
31	1F	US (Unit separator)	63	3F	&#63;	?	95	5F	&#95;	_	127	7F	&#127;	DEL

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

```
1   org 00h
2   LL:
3   mov a,#38H ; dinh dang co chu 5x7
4   call dieukhien
5   call delay
6   mov a,#01H ; xoa man hinh
7   call dieukhien
8   call delay
9   mov a,#0Fh ; bat hien thi, nhap nhay con tro
10  call dieukhien
11  call delay
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

```
13  mov a,#'A' ;  Hien A
14  call hienthi
15  call delay
16  call delay
17
18  mov a,#0C0h ; xuong dòng 2
19  call dieukhien
20  call delay
21  mov a,#'B' ; Hien B
22  call hienthi
23  call delay
24  call delay
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

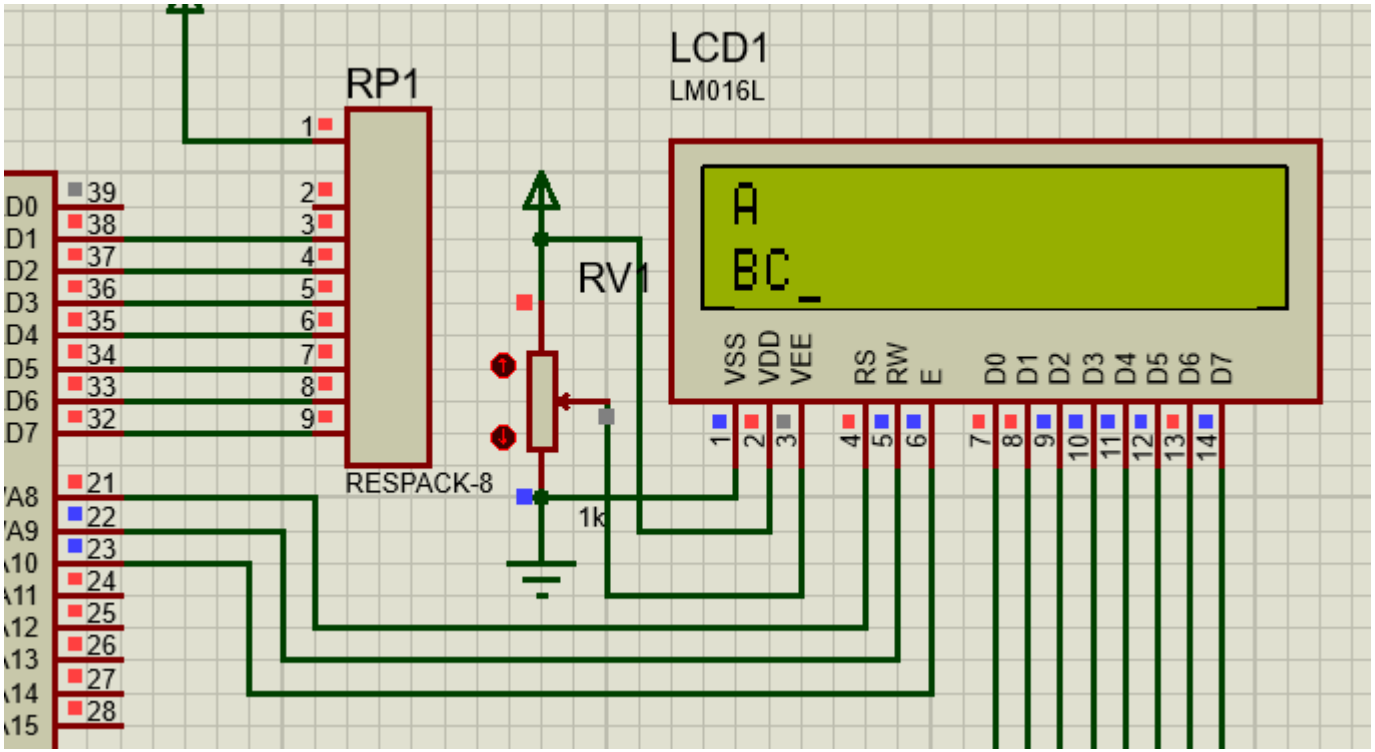
6.3. Giao tiếp dữ liệu

```
34  Hienthi:
35  mov p3,a ; d0-d7
36  setb P2.0; RS =1
37  clr P2.1; RW
38  setb P2.2; E
39  clr P2.2; E
40  call delay
41  ret
42
43  Dieukhien:
44  mov p3,a ; d0-d7
45  clr P2.0; RS =0
46  clr P2.1; RW
47  setb P2.2; E
48  call delay
49  clr P2.2; E
50  call delay
51  ret
```

```
53  delay:
54  mov r0, #50
55  v1:
56  mov tmod, #01h
57  mov th0, #high(-20000)
58  mov tl0, #low(-20000)
59  setb tr0
60  jnb tf0,$
61  clr tr0
62  clr tf0
63  djnz r0, v1
64  ret
65  end
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

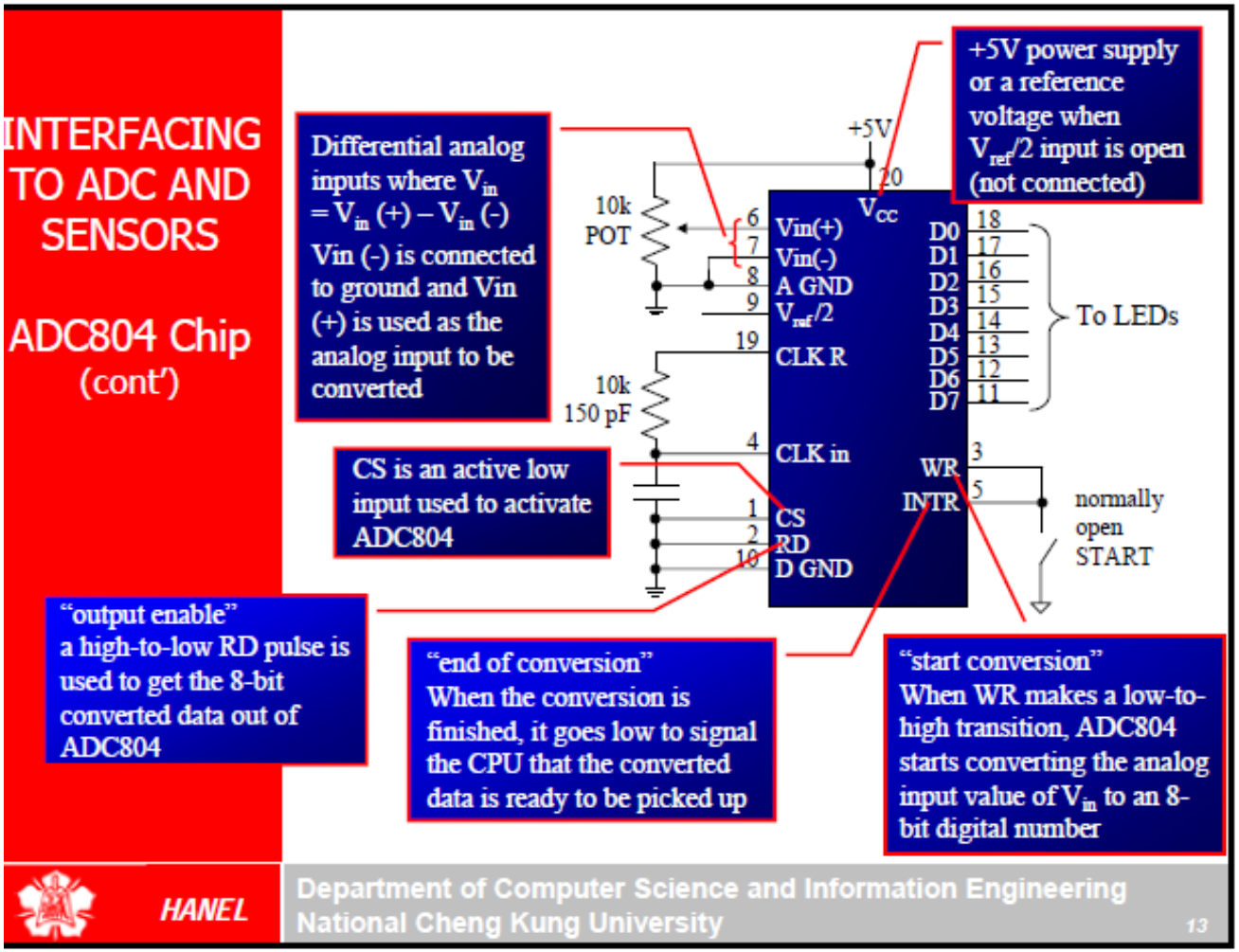
6.3. Giao tiếp dữ liệu

Dịch toàn bộ hiển thị

```
26
27    mov r2, #16
28    V2:
29    MOV A,#1Ch
30    CALL Dieukhien
31    CALL DELAY
32    djnz r2, V2
33
```

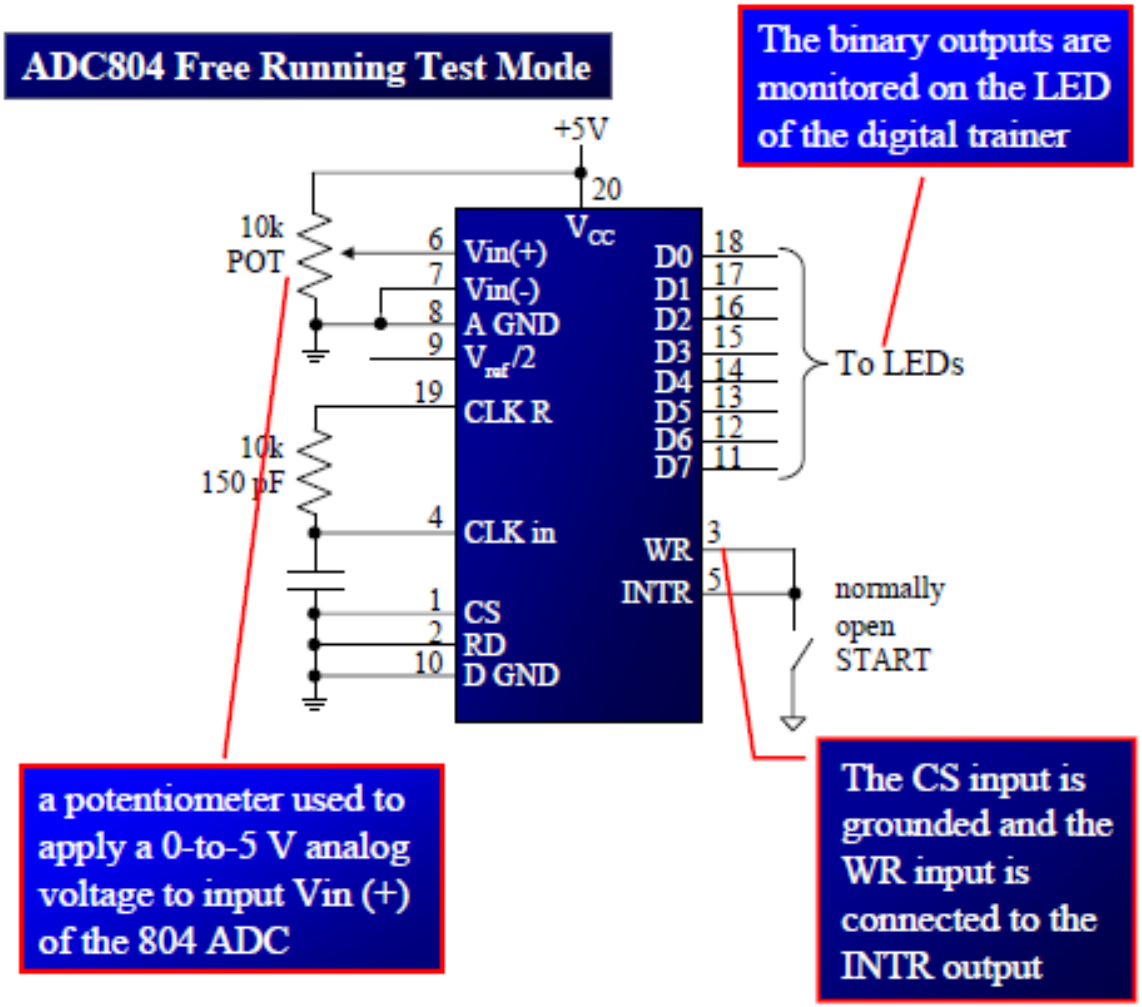
CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

- $V_{ref}/2$
  - It is used for the reference voltage
    - If this pin is open (not connected), the analog input voltage is in the range of 0 to 5 volts (the same as the Vcc pin)
    - If the analog input range needs to be 0 to 4 volts, $V_{ref}/2$ is connected to 2 volts

$V_{ref}/2$ Relation to V_{in} Range

$V_{ref}/2$(v)	V_{in}(V)	Step Size (mV)
Not connected*	0 to 5	5/256=19.53
2.0	0 to 4	4/255=15.62
1.5	0 to 3	3/256=11.71
1.28	0 to 2.56	2.56/256=10
1.0	0 to 2	2/256=7.81
0.5	0 to 1	1/256=3.90

Step size is the smallest change can be discerned by an ADC

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

- $V_{ref}/2$
  - It is used for the reference voltage
    - If this pin is open (not connected), the analog input voltage is in the range of 0 to 5 volts (the same as the Vcc pin)
    - If the analog input range needs to be 0 to 4 volts, $V_{ref}/2$ is connected to 2 volts

$V_{ref}/2$ Relation to V_{in} Range

$V_{ref}/2$(v)	V_{in}(V)	Step Size (mV)
Not connected*	0 to 5	5/256=19.53
2.0	0 to 4	4/255=15.62
1.5	0 to 3	3/256=11.71
1.28	0 to 2.56	2.56/256=10
1.0	0 to 2	2/256=7.81
0.5	0 to 1	1/256=3.90

Step size is the smallest change can be discerned by an ADC

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

- D0-D7
  - The digital data output pins
  - These are tri-state buffered
    - The converted data is accessed only when CS = 0 and RD is forced low
  - To calculate the output voltage, use the following formula

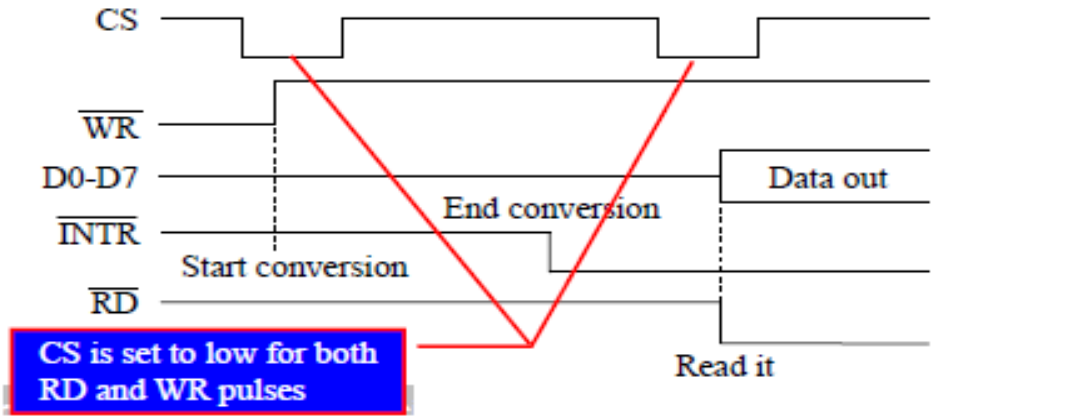
$$D_{out} = \frac{V_{in}}{step\ size}$$

    - *Dout* = digital data output (in decimal),
    - *Vin* = analog voltage, and
    - *step size* (resolution) is the smallest change

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

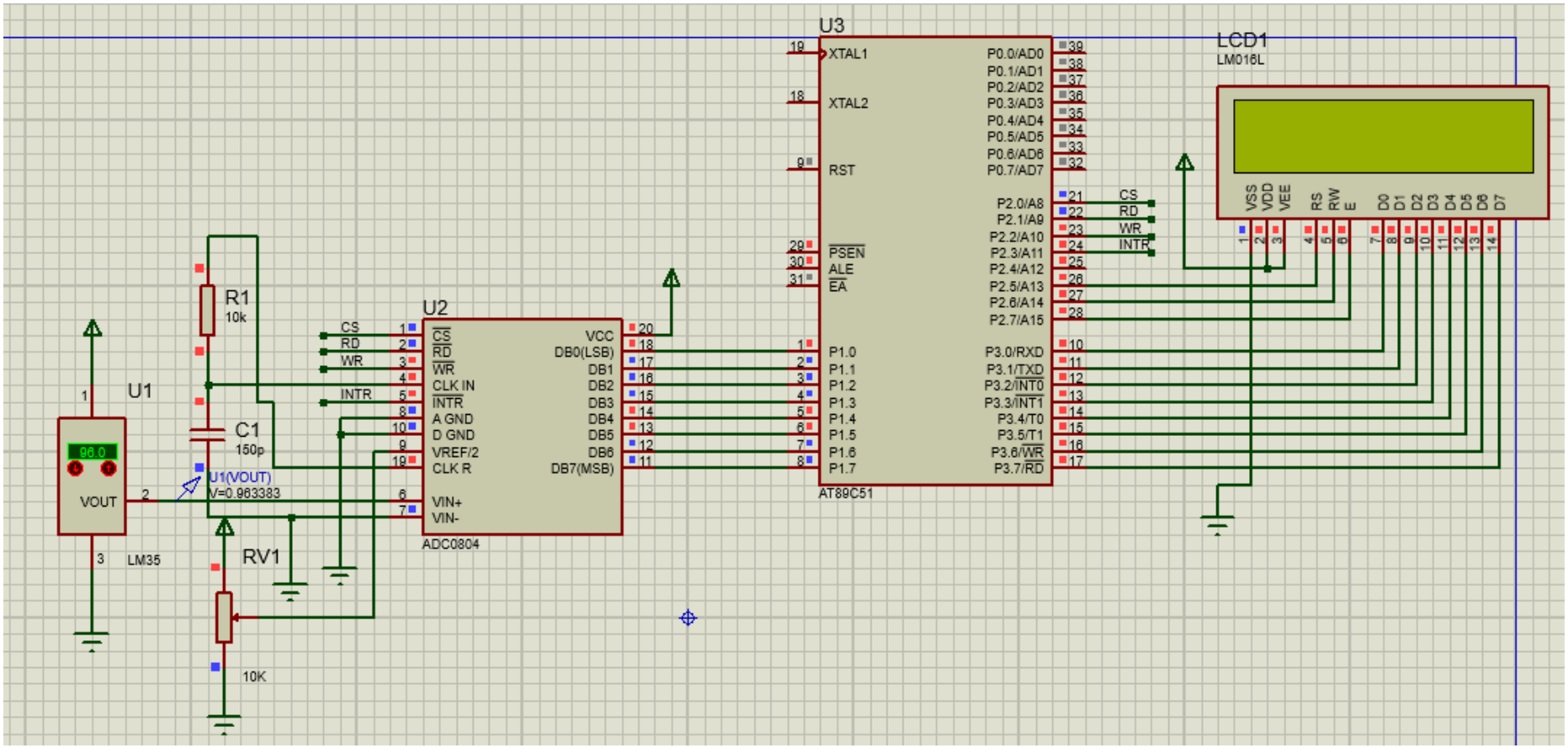
6.3. Giao tiếp dữ liệu

- The following steps must be followed for data conversion by the ADC804 chip
  - Make CS = 0 and send a low-to-high pulse to pin WR to start conversion
  - Keep monitoring the INTR pin
    - If INTR is low, the conversion is finished
    - If the INTR is high, keep polling until it goes low
  - After the INTR has become low, we make CS = 0 and send a high-to-low pulse to the RD pin to get the data out of the ADC804



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

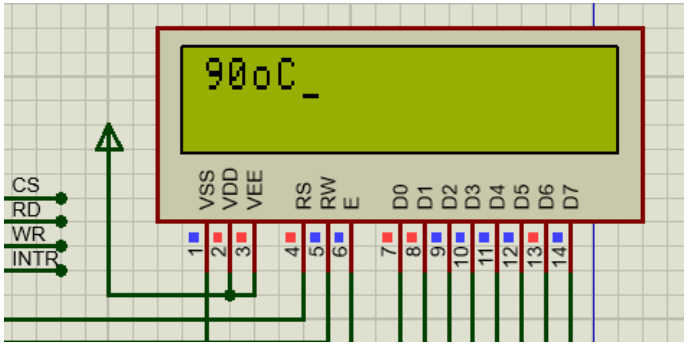
```
1   org 00h
2
3   CTCHINH:
4   Call Doc_ADC
5   call delay
6   Call Tinh_toan
7   Call Hien_thi
8   jmp CTCHINH
9
```

```
10  Doc_ADC:
11  CLR P2.0
12  CLR P2.2
13  CPL P2.2
14  JNB P2.3,$
15  CLR P2.1
16  MOV A,P1
17  ret
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

```
26  Hien_thi:
27  MOV A,#38H;DINHJ DANGJ 5*7
28  CALL DIEUKHIEN
29  CALL DELAY
30  MOV A,#01H
31  CALL DIEUKHIEN
32  CALL DELAY
33  MOV A,#0EH
34  CALL DIEUKHIEN
35  CALL DELAY
```




```
38  MOV A,#'9'
39  CALL HIENTHI
40  CALL DELAY
41  MOV A,#'0'
42  CALL HIENTHI
43  CALL DELAY
44  MOV A,#'o'
45  CALL HIENTHI
46  CALL DELAY
47  MOV A,#'C'
48  CALL HIENTHI
49  CALL DELAY
50
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

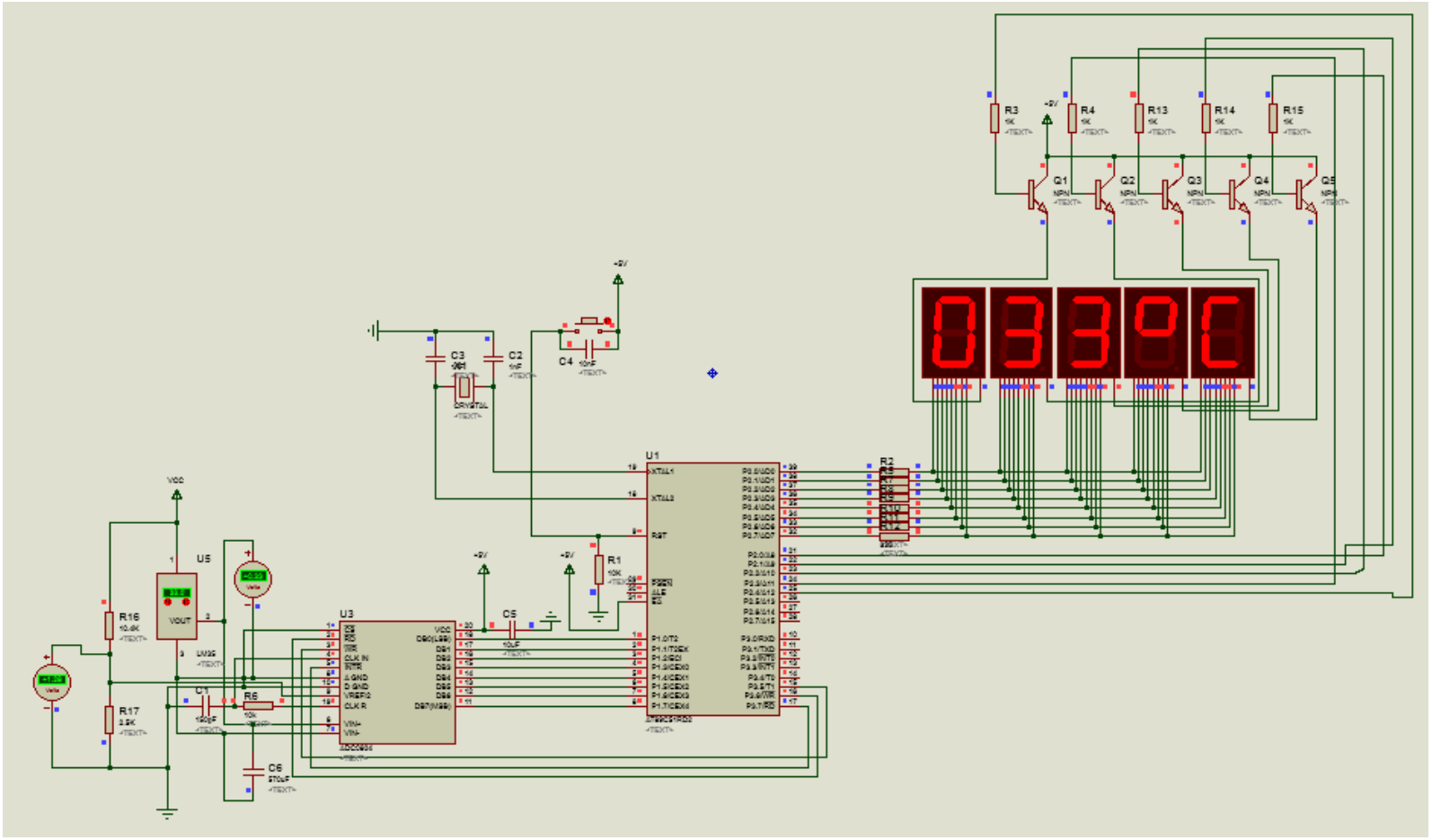
6.3. Giao tiếp dữ liệu

```
18  Tinh_toan:
19  ; doc giá tri P1 cát A=82
20  MoV B, #10
21  DIV AB ; A=8, B=2
22  Mov R6, A
23  Mov R7, B
24  MoV B, #16
25  MUL AB ; 8*16
26  MoV B, R7
27  ADD AB
```

```
37  ; R3 chua so hang chuc
38  ; R2 chua so hang dv
39  MOV A, #48
40  ADD A,R3
41  CALL HIENTHI
42  CALL DELAY
43  MOV A, #48
44  ADD A,R2
45  CALL HIENTHI
46  CALL DELAY
47  MOV A,#'o'
48  CALL HIENTHI
49  CALL DELAY
50  MOV A,#'C'
51  CALL HIENTHI
52  CALL DELAY
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu



CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

```
LED EQU P0
DATA_ADC EQU P1
;CAC BIT GIAO TIEP
ADC
INTR BIT P3.7
WR_ADC BIT P3.5
RD_ADC BIT P3.6
;VUNG NHO NHIET DO
DONVI EQU 40H
CHUC EQU 41H
TRAM EQU 42H
```

```
L1 BIT P2.4
L2 BIT P2.3
L3 BIT P2.2
L4 BIT P2.1
L5 BIT P2.0
ORG 0000H
   MOV
DATA_ADC,#0FFH    ;P1
DAU VAO DU LIEU
   MOV DPTR,#MALED7
   MOV LED,#00H
   MOV DONVI,#00H
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

```
    MOV CHUC,#00H
    MOV TRAM,#00H
MAIN:
    lcall READ_ADC
    CALL DISPLAY
    JMP MAIN
  READ_ADC:
    CLR WR_ADC
    SETB WR_ADC
HERE: JB INTR,HERE
CLR RD_ADC
    SETB RD_ADC
    MOV A,DATA_ADC
  CALL BIN2BCD
    RET
BIN2BCD:
    MOV B,#10
    DIV AB
    MOV DONVI,B
    MOV B,#10
    DIV AB
    MOV CHUC,B
    MOV TRAM,A
    RET
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.3. Giao tiếp dữ liệu

```
DISPLAY:
MOV R3, #5
LL:
   MOV LED,#0C6H    ;'C' RA
   SETB L5          ;CHO LED
   CALL DELAY       ;TRE
   CLR L5           ;TAT LED 5
 MOV LED,#9CH    ;DUA KY
      HIEU 'DO' RA LED7
            SETB L4
            CALL DELAY
            CLR L4
            MOV A,DONVI
           MOVC A,@A+DPTR
;
            MOV LED,A       ;
LED7
            SETB L3         ;
            CALL DELAY
            CLR L3
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

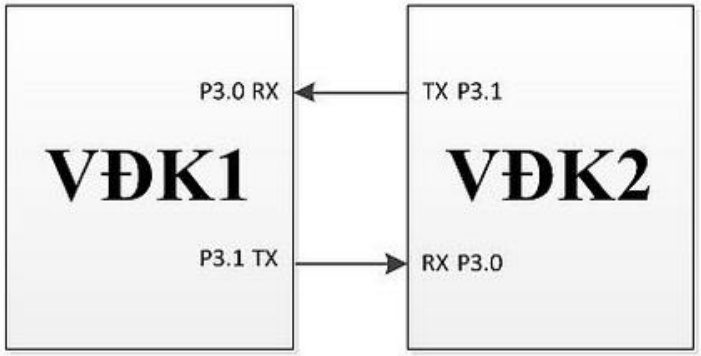
6.3. Giao tiếp dữ liệu

```
 MOV A,CHUC   ;
   MOVC A,@A+DPTR    ;
   MOV LED,A
   SETB L2
   CALL DELAY
   CLR L2
   MOV A,TRAM   ;
   MOVC A,@A+DPTR
   MOV LED,A
   SETB L1
   CALL DELAY
   CLR L1
DJNZ R3, LL
   RET
```

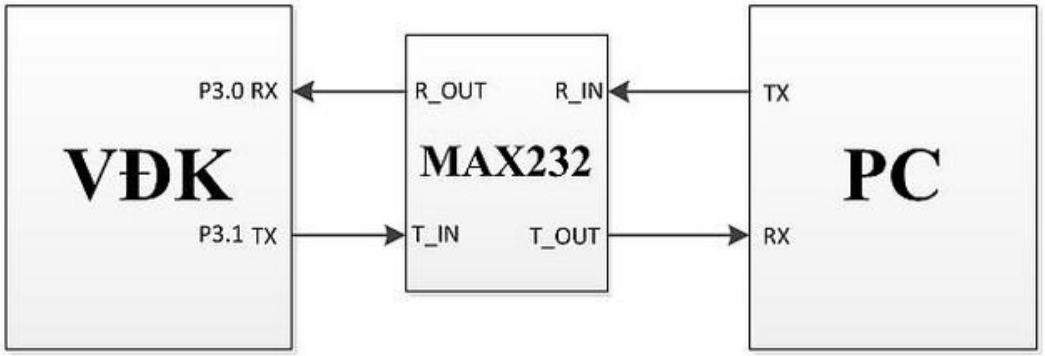
```
DELAY:
   MOV R7,#150
   DJNZ R7,$
   RET

MALED7:
DB
0C0H,0F9H,0A4H,0B0H,99H
,92H,82H,0F8H,80H,90H
; KY HIEU 'D0'    KY HIEU
'C'
;DB      9CH           0C6H
END
```

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.4. Giao tiếp truyền thông



Kết nối VĐK với VĐK



Kết nối VĐK với PC

CHƯƠNG 6 GIAO TIẾP NGOẠI VI

6.4. Giao tiếp truyền thông