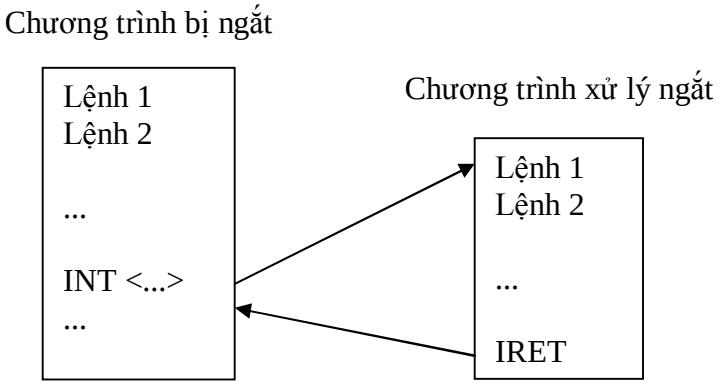
CHƯƠNG VIII: SỬ DỤNG NGẮT TRONG HỢP NGỮ

8.1 Khái niệm ngắt

Trong chương II ta đã đề cập tới khái niệm chương trình ngắt nhưng chưa giải thích ngắt là gì. Có thể tóm tắt về khái niệm ngắt như sau:

Ngắt là hành động dừng chương trình đang chạy để thực hiện một chương trình khác (chương trình này được gọi là *chương trình xử lý ngắt*). Bộ vi xử lý sẽ dừng các công việc đang thực hiện khi nó nhận được một tín hiệu yêu cầu ngắt rồi trao quyền điều khiển lại cho chương trình xử lý ngắt. Tín hiệu yêu cầu ngắt có thể do một thiết bị phần cứng hoặc do một lệnh INT trong chương trình sinh ra. Quá trình ngắt được mô tả trong hình dưới đây:



Chương trình xử lý ngắt cần được kết thúc bằng lệnh IRET để sau khi thực hiện xong có thể quay trở về thực hiện tiếp chương trình bị ngắt trước đó.

Có nhiều loại ngắt khác nhau, để phân biệt các ngắt cần dựa vào số hiệu của chúng. Bộ vi xử lý 8086 có thể quản lý 256 ngắt, được đánh số lần lượt từ 0, 1, 2,..., FFh. Dưới đây là bảng danh sách các ngắt:

Số hiệu ngắt	Chức năng
0 – 1Fh	Ngắt của BIOS
20h – 3Fh	Ngắt của DOS
40h – 7Fh	Dự trữ
80h – F0h	Dùng cho chương trình BASIC trong ROM
F1h – FFh	Không sử dụng

Giải thích:

Chương trình xử lý ngắt có thể là một bộ phận của BIOS hay của DOS, cũng có thể do người sử dụng tự viết. Ta cần phân biệt rõ hai khái niệm: "Ngắt" và "Chương trình xử lý ngắt". Không phải số hiệu ngắt nào cũng có có chương trình xử lý ngắt tương ứng.

Khi một ngắt có số hiệu từ 0 – 1Fh xuất hiện thì chúng sẽ được xử lý bởi các chương trình viết sẵn nằm trong ROM BIOS (chủ yếu là giải quyết các yêu cầu vào/ ra cơ bản). Còn nếu ngắt có số hiệu từ 20h – 3Fh thì sẽ do hệ điều hành DOS xử lý.

8.2 Phân loại ngắt

Để phân loại cần dựa trên một tiêu chí nào đó, ở đây ta sẽ phân loại ngắt dựa trên cách thức phát sinh ngắt, tạm chia làm hai loại sau: Ngắt mềm và Ngắt cứng.

8.2.1 Ngắt mềm

Ta gọi một ngắt là ngắt mềm nếu nó được phát sinh khi có lời gọi ngắt bằng lệnh INT trong chương trình. Cú pháp của lệnh INT là:

INT <Số hiệu ngắt>

Ví dụ:

```
INT   21h      ;Gọi ngắt 21h của DOS
INT   13h      ;Gọi ngắt 13h của BIOS
```

8.2.2 Ngắt cứng

Một ngắt cứng phát sinh khi có một thiết bị phần cứng gửi tín hiệu yêu cầu ngắt tới bộ vi xử lý.

Ví dụ:

Khi ta gõ một phím trên bàn phím hay bấm chuột, sẽ có tín hiệu ngắt gửi tới bộ vi xử lý để yêu cầu xử lý hành động vừa thực hiện.

Các ngắt được kích hoạt từ thiết bị ngoài (bàn phím, chuột...) giống như ví dụ trên được gọi là *Ngắt cứng ngoài*. Còn nếu ngắt phát sinh bởi các kinh kiện bổ trợ nằm trên mainboard thì được gọi là *Ngắt cứng trong* (hay *ngắt cứng nội bộ*).

8.3 Một số ngắt thông dụng

Trong phần này ta sẽ tìm hiểu tác dụng của một số ngắt và cách thức sử dụng chúng trong chương trình. Nói chung mỗi ngắt thường có nhiều chức năng khác nhau, số hiệu chức năng cần được đặt vào thanh ghi AH trước khi gọi ngắt bằng lệnh INT.

8.3.1 Ngắt 17h – Vào/ra máy in

Đây là ngắt do BIOS quản lý, nó có ba chức năng:

a) Chức năng số 0: Đưa một kí tự ra máy in

Các tham số: AH = 0
AL = Mã ASCII của kí tự cần in
DX = Số hiệu máy in

Ví dụ:

Gửi kí tự ‘A’ ra máy in.

Giải:

```
MOV   AH, 0
MOV   AL, ‘A’    ;Kí tự cần in
MOV   DX, 0      ;Máy in số 0
INT    17h

MOV   AH, 0
MOV   AL, 0Ah    ;in tiếp kí tự xuống dòng
INT    17h
```

b) Chức năng số 1: Khởi tạo cổng máy in

Các tham số: AH = 1
DX = Số hiệu máy in

c) Chức năng số 2: Kiểm tra trạng thái máy in

Vào: AH = 2
DX = Số hiệu máy in
Ra: AH chứa trạng thái máy in

Các trạng thái của máy in được liệt kê trong bảng sau:

Các bít của AH	Giá trị bít	Ý nghĩa
Bít 0	1	Máy in quá thời gian
Bít 1		Không sử dụng
Bít 2		Không sử dụng
Bít 3	1	Lỗi vào ra
Bít 4	1	Máy in được chọn
Bít 5	1	Không có giấy
Bít 6	1	Chấp nhận in
Bít 7	1	Máy in không bận

8.3.2 Ngắt 21h

Đây là ngắt hay dùng nhất của DOS, nó có rất nhiều chức năng. Ở các chương trước ta đã sử dụng bốn chức năng của ngắt này (chức năng số 1, 2, 9 và 4Ch). Trong phần này ta sẽ tìm hiểu thêm một số chức năng khác.

a) Chức năng số 39h: Tạo một thư mục trên ổ đĩa

Các tham số: AH = 39h
DS:DX = Địa chỉ của chuỗi chứa đường dẫn
(chuỗi này phải có tận cùng là 0)

Ví dụ:

Tạo thư mục ASM trong ổ đĩa C.

Giải:

```
TITLE  Tao thu muc
.MODEL  SMALL
.STACK  100H
.DATA
    DuongDan   DB   'C:\ASM', 0
.CODE
 MAIN   PROC
     MOV   AX, @DATA
     MOV   DS, AX

     MOV   AH, 39h          ;Chức năng số 39h: tạo thư mục
     LEA   DX, DuongDan     ;Lấy địa chỉ offset của chuỗi đặt
                            ;vào DX
```

```
      INT   21h             ;Gọi ngắt

      MOV   AH, 4Ch         ;Kết thúc
      INT   21h

 MAIN ENDP
END MAIN
```

b) Chức năng số 3Ah: Xoá một thư mục

Các tham số: AH = 3Ah
DS:DX = Địa chỉ của chuỗi chứa đường dẫn
(chuỗi này phải có tận cùng là 0)

Cách sử dụng tương tự như chức năng 39h ở trên.

c) Chức năng số 41h: Xoá file

Các tham số: AH = 41h
DS:DX = Địa chỉ của chuỗi chứa đường dẫn
(chuỗi này phải có tận cùng là 0)

Ví dụ:

Xoá file Vidu.txt trong thư mục gốc của ổ đĩa C.

Giải:

```
TITLE   Xoa file
.MODEL  SMALL
.STACK  100H
.DATA
    DuongDan   DB   'C:\Vidu.txt', 0
.CODE
 MAIN   PROC
     MOV   AX, @DATA
     MOV   DS, AX

     MOV   AH, 41h            ;Chức năng số 41h: xoá file
     LEA   DX, DuongDan       ;Lấy địa chỉ offset của chuỗi
                              ;đặt vào DX
     INT   21h                ;Gọi ngắt

     MOV   AH, 4Ch            ;Kết thúc
     INT   21h

 MAIN ENDP
END MAIN
```

d) Chức năng số 2Ch: Lấy thời gian từ đồng hồ hệ thống

Vào: AH = 2Ch
Ra: CH = giờ ($0 \le CH \le 23$)
CL = phút ($0 \le CL \le 59$)
DH = giây ($0 \le DH \le 59$)

DL = % giây $(0 \le DL \le 99)$

Ví dụ:

Viết chương trình hiện ra màn hình giờ hiện tại của hệ thống.

Giải:

```
TITLE   Hien thoi gian
.MODEL  SMALL
.STACK  100H
.DATA
    Time_Buf   DB   '00:00:00$'
.CODE
 MAIN   PROC
     MOV   AX, @DATA
     MOV   DS, AX

     MOV   AH, 2Ch        ;Chức năng số 2Ch: đọc thời gian
     INT   21h            ;Gọi ngắt

     ;Đổi Giờ (trong CH) từ số thập phân sang mã ASCII rồi
     ;cất vào Time_Buf
     MOV   AL, CH         ;Chuyển Giờ vào AX
     MOV   AH, 0
     MOV   DL, 10         ;Chia AX cho 10
     DIV   DL             ;AL = Thương = Số hàng chục
                          ;AH = Số dư = Số hàng đơn vị
     ADD   AL, 30h        ;Đổi số hàng chục sang mã ASCII
     ADD   AH, 30h        ;Đổi số hàng đơn vị sang mã ASCII
     MOV   Time_Buf, AL   ;Cất vào chuỗi
     MOV   Time_Buf+1, AH

     ;Đổi Phút (trong CL) từ số thập phân sang mã ASCII rồi
     ;cất vào Time_Buf
     MOV   AL, CL         ;AX chứa Phút
     MOV   AH, 0
     MOV   DL, 10         ;Chia AX cho 10
     DIV   DL             ;AL chứa số hàng chục của Phút
                          ;AH chứa số hàng đơn vị của Phút
     ADD   AL, 30h        ;Đổi sang mã ASCII
     ADD   AH, 30h
     MOV   Time_Buf+3, AL ;Cất vào chuỗi (sau dấu hai chấm)
     MOV   Time_Buf+4, AH

     ;Đổi Giây (trong DH) từ số thập phân sang mã ASCII rồi
     ;cất vào Time_Buf
     MOV   AL, DH         ;AX chứa Giây
     MOV   AH, 0
     MOV   DL, 10         ;Chia AX cho 10
```

```
    DIV   DL              ;AL chứa số hàng chục của Giây
                          ;AH chứa số hàng đơn vị của Giây
    OR    AX, 3030h       ;Đổi sang mã ASCII
    MOV   Time_Buf+6, AL
    MOV   Time_Buf+7, AH

    ;Hiện chuỗi chứa thời gian (Time_Buf) ra màn hình.
    MOV   AH, 9           ;Chức năng số 9
    LEA   DX, Time_Buf    ;Lấy địa chỉ chuỗi kí tự đặt vào
                          ;thanh ghi DX
    INT   21h             ;Gọi ngắt


    MOV   AH, 4Ch         ;Kết thúc
    INT   21h

 MAIN ENDP
END MAIN
```

Giải thích:

- Chương trình trên sẽ hiện ra màn hình giờ của dồng hồ hệ thống dưới dạng HH:MM:SS (Giờ:Phút:Giây). Các giá trị thời gian được đổi sang dạng kí tự rồi cất vào chuỗi Time_Buf. Chuỗi này được kết thúc bởi '$' để có thể hiển thị bằng chúrac năng số 9 của ngắt 21h.
- Sau khi gọi ngắt, các giá trị giờ, phút và giây được chứa trong các thanh ghi CH, CL, DH. Chúng là các số thập phân có 2 chữ số, muốn hiển thị chúng cần tách riêng chữ số hàng chục và hàng đơn vị ra, sau đó chuyển đổi sang mã ASCII tương ứng.

Ví dụ:

Giả sử AX đang chứa số thập phân 46.

Để tách riêng hàng chục và hàng đơn vị, ta đem chia AX cho 10. Như vậy Thương = 4 = AL và Số dư = 6 = AH.

Để đổi một số sang mã ASCII ta đem số đó cộng với 30h (xem lại chương III):

Chữ số hàng chục: AL + 30h = 34h

Chữ số hàng đơn vị: AH + 30h = 36h

Một cách khác để đổi sang mã ASCII là thực hiện phép toán OR với 30h:

4 OR 30h = 34h

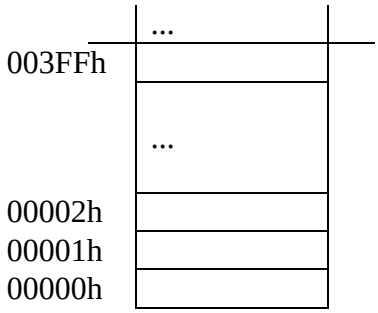
6 OR 30h = 36h

Vậy số thập phân 46 ứng với 2 kí tự có mã ASCII là 34h và 36h.

8.4 Bảng vector ngắt

Các chương trình xử lý ngắt nằm trong bộ nhớ ở những vị trí khác nhau, địa chỉ của chúng được gọi là vector ngắt. Bảng vector ngắt là nơi lưu giữ những địa chỉ này.

Bảng vector ngắt có kích thước 1024 byte (= 400h byte) và nằm ở ngay đầu bộ nhớ:



Mỗi địa chỉ bao gồm 4 byte (2 byte thấp cho offset 2 byte cao cho segment). Có tất cả 256 ngắt, do đó kích thước của bảng = 256 x 4 byte = 1024 byte.

Nếu người sử dụng muốn viết các chương trình xử lý ngắt của riêng mình (không muốn dùng các chương trình có sẵn của DOS hay BIOS) thì cần tác động tới bảng vector ngắt này. Các bước thực hiện như sau:
Bước 1: Viết chương trình xử lý ngắt mới.
Bước 2: Nạp chương trình xử lý ngắt mới vào bộ nhớ.
Bước 3: Truy nhập vào bảng vector ngắt để thay thế địa chỉ của chương trình xử lý ngắt cũ bằng địa chỉ của chương trình tự viết.

Người sử dụng có thể tự lập trình truy nhập vào bảng vector để tìm kiếm địa chỉ của các ngắt (cứ mỗi ngắt ứng với 4 byte), hoặc đơn giản hơn là sử dụng các chức năng có sẵn của ngắt 21h:

- Chức năng số 35h: Xác định địa chỉ của chương trình xử lý ngắt
  Vào: AH = 35h
  AL = Số hiệu ngắt
  Ra: ES:BX chứa địa chỉ của chương trình xử lý ngắt

Ví dụ:
Hãy xác định địa chỉ của chương trình xử lý ngắt 13h.
Giải:

```
MOV   AH, 35h
MOV   AL, 13h
INT   21h
```

- Chức năng số 25h: Thay thế địa chỉ của chương trình xử lý ngắt
  Vào: AH = 25h
  AL = Số hiệu ngắt
  DS:DX = Địa chỉ của chương trình xử lý ngắt mới

Ví dụ:
Hãy thay thế chương trình xử lý ngắt 13h bằng một thủ tục do người dùng tự viết.
Giải:
Giả sử thủ tục do người dùng tự viết có tên là NewProc.

```
MOV   AH, 25h
MOV   AL, 13h
```

```
MOV   BX, SEG NewProc
MOV   DS, BX                    ;DS chứa segment của thủ tục tự viết
MOV   DX, OFFSET NewProc        ;DX chứa offset của thủ tục tự viết
INT   21h
```

Ở đây ta sử dụng các toán tử giả SEG và OFFSET để lấy địa chỉ của thủ tục NewProc.