فصل یازدهم

مقیم کردن برنامه در حافظه، یا برنامه های TSR

(Terminate and Stay Resident)

۱۸-۱- مقدمه

معمولاً برنامه‌های کامپیوتر بر روی دیسک قرار دارند و هر برنامه‌ای که از روی دیسک می‌خواهد اجرا شود، سیستم عامل، محلی از حافظه را برای آن تخصیص می‌دهد و سپس برنامه را از دیسک به حافظه بارگذاری می‌کند. و بعد از اجرای برنامه، سیستم عامل، محلی از حافظه که برنامه کاربر آن را اشغال کرده بود را، آزاد می‌نماید، که بتواند در اختیار برنامه دیگری قرار دهد. در غیر این‌صورت بعد از اجرای چند برنامه، حافظه کاملاً پُر می‌شود، و کامپیوتر نمی‌تواند برنامهٔ دیگری را اجرا نماید. لذا هر برنامه‌ای که از روی دیسک می‌خواهد اجرا شود به‌طور موقت وارد حافظه می‌شود، و پس از اجرا، حافظه مذکور در اختیار برنامه دیگری قرار می‌گیرد، به عبارت دیگر مانند این است که برنامه کاربر از حافظه خارج شده است.

فلسفه برنامه‌های TSR یا مقیم کردن برنامه در حافظه، این است که پس از اجرای برنامه، برنامه مذکور در حافظه باقی بماند، به عبارت دیگر سیستم عامل برنامه مذکور را در حافظه مقیم کند که در این صورت نیازی نیست برای اجرای برنامه آنرا از دیسک فراخوانی کرد، چون برنامه در حافظه قرار دارد، لذا بسیار سریع‌تر اجرا می‌گردد. البته اشکال آن این است که برنامه مقیم در حافظه، قسمتی از حافظه را اشغال می‌نماید، و جای کمتری در حافظه برای برنامه‌های دیگر باقی خواهد ماند. لذا هرچه تعداد بیشتری برنامه در حافظه مقیم شوند، جای کمتری برای بقیه برنامه‌های کاربر باقی خواهد ماند. به عنوان نمونه برنامه‌های ساعت گوشه مانیتور، یا ماشین حساب کامپیوتر، . . . در حافظه مقیم می‌باشند و بقیه برنامه‌ها، بر روی دیسک قرار دارند. برنامه‌های مقیم یا TSR تا موقعی‌که کامپیوتر روشن است در حافظه قرار دارند.

۱۸-۲- روش مقیم کردن برنامه در حافظه

برای مقیم کردن برنامه کاربر در حافظه کافیست، در انتهای برنامه، به‌جای برگشت به سیستم عامل (با سرویس 4CH، دستور INT 21H) از سرویس 31H دستور INT 21H استفاده نمود، به‌شرطی‌که اندازه برنامه کاربر، برحسب تعداد پاراگراف (هر پاراگراف 16 بایت است) قبلاً در ثبات DX قرار داده شود.

به عنوان مثال اگر برنامه‌ای دارای ۱۲ پاراگراف باشد، دستورات زیر برای مقیم کردن آن در حافظه در انتهای برنامه کاربر، باید گذارده شود:

```
MOV AH,31H      ;  شماره سرویس 31H را، برای مقیم کردن برنامه در ثبات AH قرار بده
MOV DX,12       ;  ۱۲ پاراگراف (۱۹۲ بایت) در حافظه مقیم کُن
INT 21H         ;  دستور INT 21H را اجرا کُن
```

۱۸-۳- فراخوانی یا فعال کردن برنامه مقیم

روش کلی برای فعال کردن برنامه مقیم در حافظه این است که آدرس برنامه مقیم در حافظه را، جایگزین آدرس یا بُردار روتین وقفه سخت افزاری نمود. لذا هر موقع که وقفه مذکور فعال شود، به‌جای روتین وقفه سخت افزاری، برنامه مقیم در حافظه اجرا می‌گردد. و برای اینکه وقفه سخت‌افزاری نیز دچار

اشکال نشود، آدرس روتین وقفه سخت افزاری در محل دیگری در حافظه ذخیره می گردد، که در انتهای برنامه مقیم، به آن مراجعه می شود. وقفه های سخت افزاری که به کار می روند، عبارتند از وقفه 09 با دستور INT 09 که با فشار دادن کلید صفحه کلید فعال می شود و یا تایمر که با دستور INT 08H فعال می گردد. حال بررسی می نماییم که چطور می توان آدرس برنامه مقیم در حافظه را، جایگزین آدرس روتین وقفه سخت افزاری کرد.

جایگزینی آدرس برنامه مقیم در حافظه، با آدرس یا بردار روتین وقفه

برای این کار از سرویس 35H و 25H دستور INT 21H استفاده می کنیم، که در ذیل هر یک از آنها شرح داده می شود.

الف: استخراج مقادیر CS:IP یک روتین وقفه از جدول بردار وقفه توسط سرویس 35H دستور INT 21H:

در این حالت با قرار دادن عدد 35H در ثبات AH و شماره وقفه در ثبات AL و اجرای دستور INT 21 ، آدرس سگمنت کد CS روتین وقفه در ثبات ES قرار می گیرد، و مقدار افست آدرس IP نیز در ثبات BX قرار می گیرد. یعنی آدرس CS:IP روتین وقفه موردنظر، در ثبات های ES:BX گذارده می شوند. به عنوان مثال اگر بخواهیم مقدار CS:IP روتین وقفه INT 08H را پیدا می کنیم و در محل های حافظه OLDVECT و OLDVECT+2 قرار دهیم دستورات زیر را می نویسیم:

```
در سگمنت داده:                                    OLDVECT DD ?
در سگمنت کد:                                      ---
شماره سرویس روتین وقفه                  ;        MOV  AH,35H
مقدار CS:IP روتین وقفه                   ;        MOV  AL,08H
08 را استخراج کن                         ;        INT 21H
IP یا BX را در محل OLDVECT ذخیره کن      ;        MOV  OLDVECT,BX
CS یا ES را در محل OLDVECT+2 ذخیره کن    ;        MOV  OLDVECT+2,ES
```

به این ترتیب آدرس CS:IP سرویس روتین وقفه INT 08H استخراج می شود و در خانه های حافظه OLDVECT و OLDVECT+2 قرار می گیرند.

ب: جایگزین کردن آدرس برنامه مقیم در جدول بردار وقفه، به جای آدرس روتین وقفه

برای این منظور از سرویس 25H دستور INT 21H استفاده می کنیم. در این حالت با قرار دادن عدد 25H در ثبات AH و شماره وقفه در ثبات AL و همچنین گذارندن افست آدرس روتین جدید وقفه در ثبات DX، بالاخره اجرای دستور INT 21H، آدرس روتین وقفه جدید، در محل بردار وقفه مربوطه، در جدول برداری وقفه قرار می گیرند.

مثال: می خواهیم آدرس برنامه یا روتین خودمان به نام NEWISR را در محل آدرس دستور وقفه INT 08H، در جدول بردار وقفه قرار دهیم.

حل:

```
MOV   AH,25H                ;   شماره سرویس 25H را در AH قرار بده
MOV   AL,08H                ;   آدرس بُردار وقفه 08H را انتخاب کن
MOV   DX,OFFSET NEWISR      ;   افست آدرس روتین جدید را در ثبات DX قرار بده
INT 21H                     ;   وقفه 21H را فعال کُن
```

به این ترتیب آدرس سرویس روتین جدید NEWISR، در محل جدول بُردار وقفه، به جای آدرس روتین وقفه 08H قرار می‌گیرد.

۱۸-۵- ساختار برنامه‌های مقیم در حافظه یا TSR

برنامه‌های مقیم در حافظه باید به صورت COM نوشته شوند، و از دو قسمت تشکیل می‌شوند:

الف: در قسمت اوّل روتین جدید NEWISR در ابتدای برنامه گذارده می‌شود و در حافظه مقیم می‌شود و تا موقعی‌که کامپیوتر روشن است در حافظه مقیم می‌ماند، و هر موقع لازم بود اجرا می‌گردد.

ب: قسمت دوّم دستورات LOAD1 که در ادامه قسمت اوّل قرار می‌گیرد، فقط یکبار در موقع مقیم کردن قسمت اوّل NEWISR برنامه اجرا می‌شوند، سپس بلااستفاده می‌ماند. به این ترتیب ساختار برنامه مقیم در حافظه مطابق شکل (۱۸-۱) می‌باشد. این برنامه به شکل COM نوشته شده است، و قسمت‌های مختلف برنامه به شرح زیر می‌باشند:

- متغیر OLDVECT به‌صورت DD تعریف شده، که دو کلمه برای ذخیره CS:IP روتین سرویس وقفه 08H یا 09H در نظر می‌گیرد. البته بعد از آن در صورت لزوم می‌توان متغیرهای مورد نیاز برنامه را نیز تعریف نمود.
- در قسمت اوّل برنامه PART1: روتین جدید وقفه NEWISR (که ما می‌خواهیم به‌جای روتین قدیم وقفه 08H یا 09H اجرا گردد) نوشته می‌شود.
- قسمت دوّم برنامه PART2: که فقط در اولین بار اجرای برنامه و مقیم شدن قسمت اوّل فعال می‌شود،

```
        PAGE 110,100
TITLE   'tsr_exa.asm'  structure of TSR program
;-----------------------------------------------
;
;               Defining Segment of Program
;               ---------------------------
CODESG  SEGMENT 'CODE'
        ASSUME CS:CODESG
        ORG 100H
MAIN:  JMP LOAD1          ; Jump to instuctions
;-----------------------------------------------
OLDVECT DD ?     ;Four byte to save CS:IP of
;       origin ISR
;
```

```
;                          تعریف بقیه داده‌ها
;
;-----------------------------------------------
;                  PART1:                    قسمت اول
;This part of program reside in memory as new ISR
;
NEWISR  PROC NEAR
        PUSH AX
;            بقیه دستورات برنامه مقیم در حافظه
;
;
        POP AX
        JMP CS:OLDVECT   ;Perform original ISR
NEWISR  ENDP
;-----------------------------------------------
;                  PART2:                    قسمت دوم
;This part run once only, during initialization
LOAD1   PROC NEAR
;
;   1-Get vector of ISR & save on OLDVECT
;    ------------------------------------
        MOV AH,35H       ;Get the
        MOV AL,- -       ; vector
        INT 21H          ; of OLDISR
        MOV WORD PTR OLDVECT,BX     ;Save them
        MOV WORD PTR OLDVECT+2,ES   ; on OLDVECT
;
;   2-Set the offset of new ISR in vector table
;    ------------------------------------------
        MOV AH,25H       ;Set vector of
        MOV AL,- -       ; new ISR in vector table
        MOV DX,OFFSET NEWISR  ;DX=IP,DS=CS (Set by
        INT 21H          ; COM program)
;
;   3-Make resident of PART1 of program (NEWISR)
;    -------------------------------------------
        MOV DX,(OFFSET LOAD1-OFFSET CODESG) ;Find
;                how many byte resident
        ADD DX,15        ;Make it
        MOV CL,4         ; multiple of
        SHR DX,CL        ; 16 byte
;
        MOV AH,31H       ;Make it
        INT 21H          ; resident
LOAD1   ENDP
CODESG  ENDS
        END MAIN
```

شکل (۱۸-۱) ساختار اصولی برنامه اسمبلی مقیم در حافظه

شامل قسمت‌های زیر می‌باشد:

۱- بُردار یا آدرس سرویس روتین قبلی مربوط به وقفه 08H یا 09H را می‌گیرد، و در محل OLDVECT قرار می‌دهد.

۲- آدرس روتین وقفه جدید NEWVECT را در محل بُردار روتین قبلی قرار می‌دهد.

۳- روتین جدید وقفه NEWISR را در حافظه مقیم می‌کند.

همان‌طوری که قبلاً اشاره شد، برای این کار از سرویس 31H دستور INT 21H استفاده می‌شود و اندازه سرویس روتین وقفه جدید، باید برحسب پاراگراف در ثبات DX قرار گیرد.

برای این کار اختلاف آدرس CODESG و LOAD1، اندازه سرویس روتین وقفه جدید NEWISR را برحسب بایت تعیین می‌نماید، که این مقدار توسط دستور:

```
MOV DX,(OFFSET LOAD1-OFFSET CODESG)
```

در ثبات DX قرار می‌گیرد ولی باید این مقدار را تقسیم بر ۱۶ نمود که برحسب پاراگراف گردد، لذا توسط دستورات:

```
MOV  CL,4
SHR  DX,CL
```

محتوای ثبات DX را چهار بار به طرف راست شیفت می‌دهیم که تقسیم بر ۱۶ گردد. امّا ممکن است اندازه روتین وقفه جدید، دقیقاً مضارب ۱۶ نباشد، و تعداد پاراگراف در DX، کمتر از مقدار واقعی گردد، لذا برای اطمینان بیشتر قبل از دستورات فوق توسط دستور:

```
ADD  DX,15
```

۱۵ بایت به DX اضافه می‌شود، که پس از تقسیم DX بر ۱۶، تعداد پاراگراف کمتر از مقدار واقعی مورد نیاز نگردد.

ساختار برنامه شکل (۱۸-۱)، ساختار کلی برنامه‌های مقیم در حافظه می‌باشد، که با این روش می‌توان هر برنامه‌ای را در حافظه مقیم نمود. برای روشن شدن مطلب مثال‌های زیر را در نظر می‌گیریم.

مثال برنامه‌ای بنویسید که در حافظه مقیم شود، به‌طوری‌که هر لحظه که کلیدهای Alt و F10 با هم* فشار داده شوند، کامپیوتر برای مدتی بوق بزند.

حل: این برنامه به‌نام Alt_F10.asm (شکل ۱۸-۲) و مطابق با ساختار برنامه اسمبلی مقیم در حافظه شکل (۱۸-۱) می‌باشد. و برای فعال کردن برنامه مقیم در حافظه از امکانات وقفه 09H استفاده می‌شود، یعنی آدرس برنامه مقیم، جانشین آدرس وقفه 09H در حافظه می‌شود.

لذا بعد از دستور ۱، متغیر OLDINT9 DD ? برای ذخیره بُردار وقفه 09H تعریف می‌شود.

قسمت اول برنامه:

روتین **جدید وقفه** NEWISR: از دستورات ۲ تا ۱۶ می‌باشد.

- دستور ۲ : محتوای ثبات AX را ذخیره می‌کند که مقدار قبلی آن از بین نرود.
- دستورات ۳ و ۴ : سرویس 2 دستور INT 16H، فشار دادن کلیدهای Alt، Ctrl، ... را تشخیص می‌دهد، که با فشار دادن کلید Alt، بیت ۳ ثبات AL یک می‌شود، در غیر این‌صورت این بیت صفر می‌باشد.
- دستور ۵ : باعث می‌شود، که اگر بیت ۳ ثبات AL یک باشد، بیت پرچم تشخیص صفر ZF = 0، و در صورتی‌که صفر باشد ZF = 1 گردد.
- دستور ۶ : بررسی می‌کند که اگر Alt کلید فشار داده نشده، یعنی ZF = 1 است، پس دستور با برچسب EXIT یعنی دستور ۱۵ اجرا می‌گردد، در غیراین صورت دستور بعدی آن یعنی دستور ۷ اجرا می‌شود.
- دستور ۷ : باعث می‌شود که اگر کلیدی از صفحه کلید فشار داده شود، کُد اسکن* آن در ثبات AL قرار گیرد، لذا اگر کلید F10 فشار داده شود، عدد 44H که کُد اسکن F10 است در ثبات AL قرار می‌گیرد.
- دستورات ۸ و ۹ : کُد اسکن ثبات AL را با عدد 44H که کُد اسکن کلید F10 است مقایسه می‌کنند. اگر مساوی نبودند، یعنی کلید F10 فشار داده نشده، به دستور ۱۵ می‌رود، در غیر این‌صورت دستور بعدی یعنی دستور ۱۰ اجرا می‌گردد.
- دستور ۱۰ : این دستور موقعی اجرا می‌گردد، که کلیدهای Alt و F10 هر دو با هم فشار داده شده‌اند، که در این صورت عدد FFH در ثبات CX قرار می‌گیرد، که تعداد تکرار حلقه دستور ۱۴، یعنی دستور LOOP را تعیین می‌نماید.
- دستور ۱۱ تا ۱۳ : با استفاده از سرویس 0EH دستور INT 10H بلندگو کامپیوتر بوق می‌زند.
- دستور ۱۴ : باعث می‌گردد که دستورات ۱۱ تا ۱۴ به اندازه CX = FFH بار تکرار گردد، یعنی بوق کامپیوتر در این مدت به صدا درآید، که کاملاً شنیده شود.
- دستور ۱۵ : محتوای قبلی ثبات AX را در آن قرار می‌دهد.
- دستور ۱۶ : به مقدار قبلی بُردار وقفه 09H، مراجعه می‌کند، که وقفه 09H کار عادی خود را ادامه دهد.

```
        PAGE 110,100
TITLE   'alt_f10.asm'  a TSR program with alt_f10
;-----------------------------------------
;
;               Defining Segment of Program
;               ---------------------------
CODESG  SEGMENT 'CODE'
        ASSUME CS:CODESG
        ORG 100H
MAIN:  JMP LOAD1        ;1-Jump to instuctions
;-----------------------------------------------
OLDINT9 DD ?    ;Four byte to save CS:IP of INT 9H
;-----------------------------------------------
;               PART1:
;This part of program reside in memory as new ISR
;
NEWISR  PROC NEAR
```

```
        PUSH AX          ;2-Store AX
        MOV AH,02        ;3-Get keyboard
        INT 16H          ;4- status
        TEST AL,00001000B ;5-Check for ALT
        JZ EXIT          ;6-If no ALT key then exit
        IN AL,60H        ;7-Get scan code
        CMP AL,44H       ;8-See if it is F10
        JNE EXIT         ;9-If no then EXIT
        MOV CX,00FFH     ;10-CX=00FFH for delay
AGAIN:  MOV AH,0EH       ;11-If yes
        MOV AL,07        ;12- beep
        INT 10H          ;13-the speaker for
        LOOP AGAIN       ;14- a delay
EXIT:   POP AX           ;15-Restore register
        JMP CS:OLDINT9   ;16- and performe INT 09
NEWISR  ENDP
;-----------------------------------------
;               PART2:
;This part run once only, during initialization
LOAD1   PROC NEAR
;
;   1-Get vector of ISR & save on OLDVECT
;    ------------------------------------
        MOV AH,35H       ;17-Get the
        MOV AL,09H       ;18- vector
        INT 21H          ;19- of OLDISR INT 9H
        MOV WORD PTR OLDINT9,BX    ;20-Save them
        MOV WORD PTR OLDINT9+2,ES  ;21- on OLDINT9
;
;   2-Set the offset of new ISR in vector table
;    ------------------------------------------
        MOV AH,25H       ;22-Set vector of
        MOV AL,09H       ;23- new ISR in vector table
        MOV DX,OFFSET NEWISR  ;24-DX=IP,DS=CS (Set by
        INT 21H          ;25- COM program)
;
;   3-Make resident of PART1 of program (NEWISR)
;    ------------------------------------------
        MOV DX,(OFFSET LOAD1-OFFSET CODESG) ;26-
;       Find how many byte resident
        ADD DX,15        ;27-Make it
        MOV CL,4         ;28- multiple of
        SHR DX,CL        ;29-  16 byte
;
        MOV AH,31H       ;30-Make it
        INT 21H          ;31- resident
LOAD1   ENDP             ;End of procedure LOAD1
CODESG  ENDS             ;End of CODESG
        END MAIN         ;End of program
```

شکل (۱۸-۲) نمونه‌ای از برنامه مقیم در حافظه با استفاده از وقفه 09H

قسمت دوم برنامه:

همان‌طوری‌که در ساختار اصولی برنامه‌های مقیم در حافظه بحث شد، در این قسمت عملیات زیر انجام می‌شود:

الف: توسط دستورات ۱۷ تا ۲۱ آدرس روتین وقفه 09H، با استفاده از سرویس 35H دستور INT 21H گرفته می‌شود و در محل OLDINT9 ذخیره می‌گردد.

ب: توسط دستورات ۲۲ تا ۲۵، افست آدرس روتین جدید وقفه NEWISR در محل آدرس روتین وقفه 09H قرار می‌گیرد.

ج: توسط دستورات ۲۶ تا ۳۱، روتین وقفه NEWISR در حافظه مقیم می‌گردد.

با ایجاد فایل com برنامه شکل (۱۸-۲) و اجرای آن، برنامه مذکور در حافظه مقیم می‌شود، و با هر بار فشار دادن هم‌زمان کلیدهای Alt و F10 صدای بوق کامپیوتر شنیده می‌شود.

بدیهی است به جای دستورات بوق‌زدن کامپیوتر (دستورات ۱۰ تا ۱۴) می‌توان هر دستورات دلخواهی را برای برنامه مقیم در حافظه نوشت، که با فشار دادن کلیدهای مذکور، برنامه مورد نظر اجرا گردد.

نکته: در قسمت اول برنامه، یعنی بخشی از برنامه که در حافظه مقیم می‌شود، نمی‌توان از وقفه‌های DOS استفاده نمود. به این دلیل در برنامه شکل (۱۸-۲) از وقفه BIOS، سرویس 0EH دستور INT 10H استفاده کردیم.

مثال برنامه‌ای بنویسید که در حافظه مقیم شود، و هر ۳۰ ثانیه کامپیوتر بوق بزند.

حل: برای این منظور از وقفه زمانی INT 08H استفاده می‌نمائیم. همانطوری‌که در فصل ۱۵ بحث شد، وقفه INT 08H در هر ۵۵ میلی‌ثانیه فعال می‌شود (دقیقاً ۵۴/۹۴ میلی‌ثانیه، یا در هر ثانیه ۱۸/۲ بار). بنابراین اگر بخواهیم کامپیوتر هر ۳۰ ثانیه بوق بزند، باید ۵۵۰ بار وقفه فعال شود (چون ثانیه ۳۰ = میلی‌ثانیه ۳۰۰۰۰ = ۵۴/۹۴ × ۵۵۰).

برای این کار در برنامه شکل (۱۸-۳) متغیر COUNT را با مقدار اولیه ۵۵۰ به‌صورت زیر معرفی می‌کنیم:

```
COUNT DW 550
```

```
        PAGE 110,100
TITLE   'timer_8.asm'  a TSR program with timer 8
;-----------------------------------------------
;
;               Defining Segment of Program
;               ---------------------------
CODESG  SEGMENT 'CODE'
        ASSUME CS:CODESG
        ORG 100H
MAIN:  JMP LOAD1          ;1-Jump to instuctions
;-----------------------------------------------
OLDINT8 DD ?    ;Four byte to save CS:IP of INT 8H
```

```
COUNT   DW 550  ;550 *55 ms=30 seconds
;-----------------------------------------------
;               PART1:
;This part of program reside in memory as new ISR
;
NEWISR  PROC NEAR
        DEC CS:COUNT      ;2-Is time
        JNZ EXIT          ;3- over?
        MOV CS:COUNT,550 ;4-If yes intialize COUNT
        MOV CX,00FFH      ;5- make delay
AGAIN:  MOV AH,0EH        ;6- and beep
        MOV AL,07         ;7- the speaker
        INT 10H           ;8- for
        LOOP AGAIN        ;9- a delay
EXIT:   JMP CS:OLDINT8  ;10- take care INT 08
NEWISR  ENDP
;-----------------------------------------------
;               PART2:
;This part run once only, during initialization
LOAD1   PROC NEAR
;
;  1-Get vector of ISR & save on OLDVECT
;   ------------------------------------
        MOV AH,35H        ;11-Get the
        MOV AL,08H        ;12- vector
        INT 21H           ;13- of OLDISR INT 8H
        MOV WORD PTR OLDINT8,BX    ;14-Save them
        MOV WORD PTR OLDINT8+2,ES  ;15- on OLDINT8
;
;  2-Set the offset of new ISR in vector table
;   ------------------------------------------
        MOV AH,25H        ;16-Set vector of
        MOV AL,08H        ;17- new ISR in vector table
        MOV DX,OFFSET NEWISR  ;18-DX=IP,DS=CS (Set by
        INT 21H           ;19- COM program)
;
;  3-Make resident of PART1 of program (NEWISR)
;   ------------------------------------------
        MOV DX,(OFFSET LOAD1-OFFSET CODESG) ;20-
;       Find how many byte resident
        ADD DX,15         ;21-Make it
        MOV CL,4          ;22- multiple of
        SHR DX,CL         ;23-  16 byte
;
        MOV AH,31H        ;24-Make it
        INT 21H           ;25- resident
LOAD1   ENDP              ;End of procedure LOAD1
CODESG  ENDS              ;End of CODESG
        END MAIN          ;End of program
```

شکل (۱۸-۳) نمونه‌ای از برنامه مقیم در حافظه با استفاده از وقفه 08

و هر بار که وقفه 08 فعال شود، یک واحد از آن کم می‌گردد، تا COUNT صفر شود، که در این صورت مدت ۳۰ ثانیه گذشته است.

برنامه Timer-8.asm شکل (۱۸-۳) مطابق با ساختار برنامه‌های مقیم در حافظه شکل (۱۸-۱) می‌باشد.

برای فعال کردن برنامه مقیم در حافظه از امکانات وقفه 08H استفاده شده است یعنی آدرس برنامه مقیم، جانشین آدرس وقفه 08H در حافظه می‌شود. لذا بعد از دستور ۱، متغیر OLDINT8 DD ? برای ذخیره آدرس وقفه 08H تعریف می‌شود، و مقدار COUNT نیز مانند فوق تعریف می‌گردد.

قسمت اول برنامه

روتین جدید وقفه NEWISR : از دستورات ۲ تا ۱۰ می‌باشد.

- **دستورات ۲ و ۳** : یک واحد از مقدار COUNT کم می‌کند، اگر COUNT صفر نشده بود، به دستور ۱۰ می‌رود، و کار وقفه 08H ادامه می‌یابد. و با فعال شدن وقفه 08H، دوباره دستورات ۲ و ۳ و ۱۰ اجرا می‌شوند، و این حلقه آنقدر اجرا می‌گردد تا COUNT صفر شود، یعنی وقتی که ۵۵۰ بار وقفه 08H اجرا شد که معادل ۳۰ ثانیه است، دستور ۴ اجرا می‌گردد.
- **دستور ۴** : مقدار اولیه به متغیر COUNT می‌دهد.
- **دستورات ۵ تا ۹** : باعث می‌شوند مطابق مثال قبل کامپیوتر بوق بزند.
- **دستور ۱۰** : دوباره کنترل به وقفه 08H برمی‌گردد، و کار عادی وقفه مذکور ادامه می‌یابد.

قسمت دوّم برنامه:

همانطوری‌که در ساختار اصولی برنامه‌های مقیم در حافظه بحث شد، در این قسمت عملیات زیر انجام می‌شود:

الف : توسط دستورات ۱۱ تا ۱۵ آدرس روتین وقفه 08H با استفاده از سرویس 35H دستور INT 21H گرفته می‌شود، و در محل OLDINT8 ذخیره می‌گردد.

ب : توسط دستورات ۱۶ تا ۱۹، افست آدرس روتین جدید وقفه NEWISR در محل آدرس روتین وقفه 08H قرار می‌گیرد.

ج : توسط دستورات ۲۰ تا ۲۵ روتین وقفه NEWISR در حافظه مقیم می‌گردد.

با ایجاد فایل com برنامه شکل (۱۸-۳)، و اجرای آن، برنامه مذکور در حافظه مقیم شده است، و هر ۳۰ ثانیه یکبار بوق کامپیوتر شنیده می‌شود.

البته به جای دستورات زدن بوق کامپیوتر (دستورات ۵ تا ۹)، می‌توان هر تعداد و هر نوع دستوراتی را برای برنامه مقیم در حافظه نوشت، که هر ۳۰ ثانیه برنامه مذکور اجرا گردد.

۱۸-۶- خلاصه

در این فصل فلسفه برنامه‌های مقیم در حافظه و ویژگی‌های آن، روش مقیم نمودن، فراخوانی یا فعال کردن برنامه مقیم با استفاده از وقفه‌های 08H و 09H، جایگزینی آدرس برنامه در حافظه، با آدرس یا بُردار روتین وقفه، و بالاخره ساختار برنامه‌های مقیم TSR در حافظه، با ذکر مثال‌ها و برنامه‌های اجرا شده روی کامپیوتر مورد بحث قرار گرفتند.

۱۸-۷- تمرین

۱۸-۱- برنامه‌ای بنویسید که در حافظه مقیم شود، به‌طوری‌که هر لحظه که کلیدهای Ctrl و F10 با هم فشار داده شوند، کامپیوتر برای مدتی بوق بزند.

۱۸-۲- برنامه‌ای بنویسید که در حافظه مقیم شود، به طوری‌که هر لحظه که کلیدهای Alt و F7 با هم فشار داده شوند، کامپیوتر برای مدتی بوق بزند.

۱۸-۳- برنامه‌ای بنویسید که در حافظه مقیم شود، و هر ۱۰ ثانیه یکبار بوق بزند.

۱۸-۴- برنامه‌ای بنویسید که در حافظه مقیم شود، و با فشار دادن کلیدهای Ctrl و F7 کار به‌خصوصی را برای شما انجام دهد.