SISTEMAS BASADOS EN MICROPROCESADORES

Grado en Ingeniería Informática

ENUNCIADO PROBLEMA 2

Implementar un programa en ensamblador de 80x86 que rote un carácter en la esquina superior derecha de la pantalla hasta que se pulse una tecla, usando las interrupciones del RTC como base de tiempos. La pulsación de la tecla se ha de detectar usando el DOS. La pantalla está previamente configurada en modo texto de 80x25.

Programa Principal

```
; ETIQUETAS DEL PROGRAMA
VIDEO EQU 0B800H   ; Buffer de vídeo en modo texto 80x25

; SEGMENTO DE PILA

stacksg SEGMENT STACK "stack"
        DB 256 DUP (0)
stacksg ENDS

; SEGMENTO DE DATOS

datossg SEGMENT
        OFFSET_O   DW 0        ; Vector original de la INT 70H
        SEGMEN_O   DW 0
        TABLA      DB  "/|\-"  ; Tabla de caracteres
        CONT       DW 0        ; Índice a la tabla de caracteres
datossg ENDS

; CODIGO DEL PROGRAMA

codesg  SEGMENT
        assume  CS:codesg, DS:datossg, SS:stacksg, ES:datossg

;****************************************
;* Programa principal                   *
;****************************************

rtc     proc far

        ; Configuración interna
        MOV  AX, datossg
        MOV  DS, AX
        MOV  ES, AX
        MOV  AX, stacksg
        MOV  SS, AX
        MOV  SP, 256

        ; Configuración inicial del teclado y RTC
```

```
        CALL vaciar_buffer
        CALL config_rtc
        CALL start_rtc

        ; Instala el vector de la INT 70H
        CLI
        MOV  AX, 0
        MOV  ES, AX

        ; guarda los valores originales
        MOV  AX, word ptr ES:[70H*4]
        MOV  OFFSET_O, AX
        MOV  AX, word ptr ES:[70H*4 + 2]
        MOV  SEGMEN_O, AX

        ; Apunta a la RSI del RTC: serv70_int
        MOV  word ptr ES:[70H*4], offset serv70_int
        MOV  word ptr ES:[70H*4 + 2], seg serv70_int
        STI

; El bucle principal espera la pulsación de una tecla
; para terminar
bucle:
        MOV  AH, 0BH   ; Lee el estado del teclado
        INT  21H
        CMP  AL, 0
        JE   bucle   ; No hay tecla -> sigue esperando

fin:
        ; Desactiva la interrupción del RTC
        CALL stop_rtc
        CLI

        ; Repone vector de interrupción original del RTC
        MOV  AX, 0
        MOV  ES, AX

        MOV  AX, OFFSET_O
        MOV  word ptr ES:[70H*4], AX
        MOV  AX, SEGMEN_O
        MOV  word ptr ES:[70H*4 + 2], AX
        STI

        ; Vacía buffer del teclado
        CALL vaciar_buffer

        ; Devuelve el control al DOS
        MOV  AX, 4C00H
        INT  21H
rtc     endp


; Vacía el buffer del teclado
vaciar_buffer:
        PUSH AX
        MOV  AH, 0CH
        MOV  AL,0
```

```
        INT  21H
        POP AX
        ret

; ..........................................
; . Funciones relacionadas  con el RTC       .
; ..........................................

; Función que configura el RTC
config_rtc proc near
        PUSH AX

        CLI
        ; Activa interrupciones en IMRs de PICs
        IN    AL, 21H         ; Lee IMR maestro
        AND   AL, 11111011b   ; Pone a 0 bit 2 IMR maestro
        OUT   21H, AL         ; Escribe IMR maestro
        IN    AL, 0A1H        ; Lee IMR esclavo
        AND   AL, 11111110b   ; Pone a 0 bit 0 IMR esclavo
        OUT   0A1H, AL        ; Escribe IMR esclavo

        ; Configura la frecuencia del RTC
        MOV   AL, 0AH
        OUT   70H, AL
        MOV   AL, 00101111b   ; DV = 32768Hz, RS = 2Hz
        OUT   71H, AL

        STI
        POP AX
        RET
config_rtc endp

; Activa las interrupciones del RTC
start_rtc proc near
        PUSH AX
        CLI

        ; Activa interrupción PIE y desactiva las demás
        MOV   AL, 0BH
        OUT   70H, AL
        IN    AL, 71H         ; lee registro B
        OR    AL, 01000000b   ; PIE = 1
        AND   AL, 01000111b   ; SET = AIE = UIE = SQWE = 0
        MOV   AH, AL
        MOV   AL, 0BH
        OUT   70H, AL
        MOV   AL, AH
        OUT   71H, AL         ; Escribe registro B

        MOV   AL, 0CH
        OUT   70H, AL
        IN    AL, 71H   ; Lee registro C: Pone a cero banderas

        STI
        POP   AX
        RET
start_rtc endp
```

```
; Desactiva las interrupciones del RTC
stop_rtc proc near
        PUSH AX
        CLI
        ; Desactiva interrupción PIE
        MOV   AL, 0BH
        OUT   70H, AL
        IN    AL, 71H         ; Lee registro B
        AND   AL, 10111111b   ; PIE = 0
        MOV   AH, AL
        MOV   AL, 0BH
        OUT   70H, AL
        MOV   AL, AH
        OUT   71H, AL         ; Escribe registro B

        MOV   AL, 0CH
        OUT   70H, AL
        IN    AL, 71H   ; Lee registro C: Pone a cero banderas

        STI
        POP   AX
        RET
stop_rtc endp

;.............................................
;. Rutina de servicio de la interrupción 70H   .
;.............................................
serv70_int proc far

         STI
         PUSH AX BX ES DS

         MOV  AX, datossg
         MOV  DS, AX

        ; Comprueba que ha sido el RTC-PIE quien ha interrumpido
        MOV   AL, 0CH
        OUT   70H, AL
        IN    AL, 71H         ; lee registro C
        AND   AL, 01000000b   ; PF = bit 6 de registro C
        JNZ  pi_int
        JMP  salir

pi_int:  ; Interrupción periódica
         MOV  AX, VIDEO
         MOV  ES, AX     ; ES apunta al buffer de vídeo
         MOV  BX, CONT   ; BX := índice a la tabla de caracteres
         INC  BX
         CMP  BX, 4      ; Superado final de tabla de caracteres?
         JNE  sigue      ; NO -> Se imprime el carácter apuntado por BX
         MOV  BX, 0      ; SI -> BX apunta al primer carácter

; Muestra el carácter en la esquina superior derecha (col 79)
sigue:   MOV  AL, TABLA[BX]
         MOV  byte ptr ES:[79*2], AL
```

```
        ; Actualiza variable de índice a tabla de caracteres
        MOV  word ptr CONT, BX

salir:  ;  Manda los EOIs
        MOV  al, 20H   ; EOI no específico
        OUT  20H, al   ; manda EOI al PIC maestro
        OUT  A0H, al   ; manda EOI al PIC esclavo

        POP  DS ES BX AX

        IRET
serv70_int endp


codesg  ends
end     rtc
```