MICROPROCESADORES 2003/2004

RELOJ CRONÓMETRO DIGITAL

Jaime Pérez Crespo
Rubén Seijas Valverde

Introducción:

El objetivo de esta práctica es la construcción de un reloj y cronómetro digital bajo el procesador 8051 de Intel.

El funcionamiento del programa es sencillo. Cargaremos y arrancaremos el programa a través del puerto serie mediante la interfaz de desarrollo y su simulador, y el programa comenzará a ejecutar. En un principio se nos preguntará si deseamos que el sistema funcione en modo reloj o modo cronómetro. Se le pedirá al usuario que pulse las teclas R o C (sin distinguir entre mayúsculas y minúsculas) para realizar esta selección.

En caso de seleccionar el usuario el modo reloj, se le pedirán respectivamente las horas y los minutos a partir de los que comenzar a contar. Se le seguirá preguntando al usuario siempre y cuando no introduzca una entrada válida, esto es, dígitos numéricos que representen números localizados entre 0 y 23 para las horas, y entre 0 y 59 para los minutos. Una vez el usuario introduzca una entrada correcta, el programa comenzará a contar a partir de dicha hora, empezando con 0 segundos, a modo de un reloj convencional.

Por contra, si se selecciona el modo de funcionamiento cronómetro, directamente el programa comenzará a ejecutar a partir de 0 minutos, 0 segundos, 0 centésimas. Cuando el usuario accione el pulsador incluído en el circuito se producirá una señal de lapsus, durante la cual el cronómetro seguirá en funcionamiento, pero el tiempo mostrado en pantalla estará congelado en aquel que transcurría cuando se accionó el pulsador. Si el usuario acciona de nuevo el pulsador, el cronómetro se parará definitivamente, mostrando el tiempo en ese instante, provocando un reset. Una última pulsación provocará una señal de stop que pondrá todos los contadores a 0 de nuevo y reinicializará la rutina de cronómetro.

Listado:

A continuación se muestra el listado en ensamblador del programa construído para el este reloj y cronómetro digital.

```
A51 MACRO ASSEMBLER  RELOJ                                                    09/12/03
06:37:31 PAGE     1


DOS MACRO ASSEMBLER A51 V5.28m
OBJECT MODULE PLACED IN RELOJ.OBJ
ASSEMBLER INVOKED BY: C:\C51EVAL\BIN\A51.EXE RELOJ.A51 DB EP

LOC  OBJ            LINE     SOURCE

                       1     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                       2     ;Practica 2 de la asignatura de;
                       3     ; microprocesadores 2003/2004  ;
                       4     ;Autores:                      ;
                       5     ; Jaime Perez Crespo           ;
                       6     ; Ruben Seijas Valverde        ;
                       7     ;Ultima modificacion: 8-12-03  ;
                       8     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                       9     NAME    RELOJ
                      10
                      11     ;Definicion de retorno de carro y nueva linea (CR y LF)
  000D                12     CR      EQU     0DH             ; Carriage Return
  000A                13     LF      EQU     0AH             ; Line Feed
```

```
                     14
                     15     PROG    SEGMENT CODE
                     16     STACK   SEGMENT IDATA
                     17
----                 18             RSEG    STACK
0000                 19             DS      10H             ; 16 bytes de pila
                     20
----                 21             CSEG    AT      0
                     22             USING   0               ; Banco 0 de registros
                     23
0003                 24             ORG     03H
0003 020000   F      25             LJMP    PULSACION       ; Rutina de la interrucion de pulso del
boton
                     26
000B                 27             ORG     0BH
000B 020000   F      28             LJMP    DESBORD         ; Rutina de la interrupcion de
desbordamiento del timer
                     29
                     30     ;Tras reset se comienza a ejecutar sobre la direccion 0
0000                 31             ORG     0H
0000 020000   F      32             JMP     START           ; Saltamos al comienzo del programa
----                 33             RSEG    PROG
                     34
                     35
                     36
                     37
                     38
                     39     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     40     ;Rutina que escribe un caracter en el puerto serie;
                     41     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     42     ;El caracter esta almacenado en A
0000                 43     PUTCHAR:
0000 C299            44             CLR     TI              ; TI = 0
0002 F599            45             MOV     SBUF,A          ; Enviamos el caracter
0004 3099FD          46             JNB     TI,$            ; Si TI == 0 entonces linea
ocupada,esperamos a TI == 1
0007 22              47             RET
                     48
                     49     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     50     ;Rutina que lee un caracter por el puerto serie;
                     51     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     52     ; El caracter leido se almacena en A
0008                 53     GETCHAR:
0008 3098FD          54             JNB     RI,$            ; Si RI==0 entonces no se ha
recibido,esperamos a RI==1
000B E599            55             MOV     A,SBUF          ; Leemos el caracter por el puerto
000D C298            56             CLR     RI
                     57             ; Como mon51 envia 11H de vez en cuando lo ignoramos
000F F5F0            58             MOV     B,A             ; Salvaguardamos A

A51 MACRO ASSEMBLER  RELOJ                                                              09/12/03
06:37:31 PAGE     2

0011 24EF            59             ADD     A,#-11H         ; Restamos 11H a A
0013 60F3            60             JZ      GETCHAR         ; Si es cero leemos otro caracter
0015 E5F0            61             MOV     A,B             ; Recuperamos A desde B
0017 22              62             RET
                     63
                     64
                     65     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     66     ;Rutina que escribe una cadena por el puerto serie;
                     67     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     68     ; Escribe la cadena apuntada en DPTR, y MODIFICA dicho puntero
0018                 69     PUTSTRING:
0018 E4              70             CLR     A
0019 93              71             MOVC    A,@A+DPTR       ; Lee del segmento de codigo
001A 6006            72             JZ      EXIT            ; Sale si el caracter es 00H
001C 120000   F      73             CALL    PUTCHAR         ; Escribimos el caracter
001F A3              74             INC     DPTR            ; Incrementamos el contador
0020 80F6            75             SJMP    PUTSTRING
0022 22              76     EXIT:   RET
                     77
                     78     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     79     ;Rutina que escribe un caracter ':' por el puerto serie;
                     80     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0023                 81     PUTSEPARATOR:
0023 743A            82             MOV     A,#58D          ; Cargamos un : en A
0025 120000   F      83             CALL    PUTCHAR         ; Lo escribimos
```

```
0028 22                      RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina que obtiene la equivalencia entre digitos ASCII y numeros;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ; #'0' == #48D, ... , #'9' == #57D
0029                 CONVERT:
0029 F5F0                    MOV     B,A               ; Guardamos A en B
002B B43002                  CJNE    A,#48D,UNO        ;Si es X,no saltamos,ponemos X en A.Si
no,saltamos a X+1
002E 7400                    MOV     A,#00D
0030 B43102          UNO:    CJNE    A,#49D,DOS
0033 7401                    MOV     A,#01D
0035 B43202          DOS:    CJNE    A,#50D,TRES
0038 7402                    MOV     A,#02D
003A B43302          TRES:   CJNE    A,#51D,CUATRO
003D 7403                    MOV     A,#03D
003F B43402          CUATRO: CJNE    A,#52D,CINCO
0042 7404                    MOV     A,#04D
0044 B43502          CINCO:  CJNE    A,#53D,SEIS
0047 7405                    MOV     A,#05D
0049 B43602          SEIS:   CJNE    A,#54D,SIETE
004C 7406                    MOV     A,#06D
004E B43702          SIETE:  CJNE    A,#55D,OCHO
0051 7407                    MOV     A,#07D
0053 B43802          OCHO:   CJNE    A,#56D,NUEVE
0056 7408                    MOV     A,#08D
0058 B43902          NUEVE:  CJNE    A,#57D,FAIL       ;Si no es 9,comprobamos si fue alguno
de los anteriores
005B 7409                    MOV     A,#09D
005D B5F006          FAIL:   CJNE    A,B,OK            ; Si A es distinto de B, se encontro un
digito
0060 75F00A                  MOV     B,#10D            ; Si no es un digito, B == 10D
0063 020000   F              JMP     OK2               ; Salimos directamente
0066 F5F0            OK:     MOV     B,A               ; Devolvemos el digito en A y B
0068 22              OK2:    RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina que pasa de numerico a ASCII;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ; Soporta un numero de dos digitos almacenado en A.
                     ; El primer digito se guarda en A, el segundo en B.
0069                 GETASCII:
0069 F8                      MOV     R0,A              ; Guardamos el numero en R0
```



```
006A 75F00A                  MOV     B,#10D            ; Cargamos 10 en B
006D 84                      DIV     AB                ; Dividimos el numero por 10
006E F5F0                    MOV     B,A               ; Guardamos en B una copia de las
decenas
0070 2430                    ADD     A,#48D            ; Le sumamos #48 (ASCII '0') al primer
digito
0072 F9                      MOV     R1,A              ; Guardamos ese primer digito
0073 E5F0                    MOV     A,B               ; Restauramos las decenas
0075 75F00A                  MOV     B,#10D            ; Cargamos un 10 en B
0078 A4                      MUL     AB                ; Obtenemos las decenas
0079 F5F0                    MOV     B,A               ; Guardamos las decenas en B
007B E8                      MOV     A,R0              ; Restauramos el numero original en A
007C 95F0                    SUBB    A,B               ; Le restamos las decenas al original
007E 2430                    ADD     A,#48D            ; Le sumamos #48 (ASCII '0') al segundo
digito
0080 F5F0                    MOV     B,A               ; Dejamos en B el segundo digito
0082 E9                      MOV     A,R1              ; y en A el primero
0083 22                      RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina que escribe la hora por el puerto serie;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ; Escribe la hora actual guardada de R4 a R6
0084                 PRINT_R:
0084 EC                      MOV     A,R4              ; Cargamos las horas
0085 120000   F              CALL    GETASCII          ; Las convertimos a digitos
0088 120000   F              CALL    PUTCHAR           ; Escribimos el primer digito
008B E5F0                    MOV     A,B
008D 120000   F              CALL    PUTCHAR           ; Escribimos el segundo digito
0090 120000   F              CALL    PUTSEPARATOR      ; Escribimos un :
```

```
0093 ED                     MOV     A,R5            ; Cargamos los minutos
0094 120000   F             CALL    GETASCII        ; Los convertimos a digitos
0097 120000   F             CALL    PUTCHAR         ; Escribimos el primer digito
009A E5F0                   MOV     A,B
009C 120000   F             CALL    PUTCHAR         ; Escribimos el segundo digito
009F 120000   F             CALL    PUTSEPARATOR    ; Escribimos un :
00A2 EE                     MOV     A,R6            ; Cargamos los segundos
00A3 120000   F             CALL    GETASCII        ; Los convertimos a digitos
00A6 120000   F             CALL    PUTCHAR         ; Escribimos el primer digito
00A9 E5F0                   MOV     A,B
00AB 120000   F             CALL    PUTCHAR         ; Escribimos el segundo digito
00AE 22                     RET


                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;Rutina que escribe el cronometro por el puerto serie;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ; Escribe el instante actual guardado de R4 a R7
00AF                PRINT_C:
00AF 120000   F             CALL    PRINT_R         ; Escribimos la hora (h:m:s)
00B2 7422                   MOV     A,#'"'          ; Escribimos "
00B4 120000   F             CALL    PUTCHAR
00B7 EF                     MOV     A,R7            ; Cargamos las centesimas
00B8 120000   F             CALL    GETASCII        ; Los convertimos a digitos
00BB 120000   F             CALL    PUTCHAR         ; Escribimos el primer digito
00BE E5F0                   MOV     A,B
00C0 120000   F             CALL    PUTCHAR         ; Escribimos el segundo digito
00C3 22                     RET

                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;Rutina que configura el timer0 para que se desborde cada centesima;
                    ;de segundo                                                        ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
00C4                TIMECONF:
                    ; Configuramos el Timer0:
00C4 758901                 MOV     TMOD,#00000001B ; C/T = 0, Mode = 1
00C7 758CD8                 MOV     TH0,#11011000B  ; Configuramos el timer0 para que se
desborde cada
00CA 758AF0                 MOV     TL0,#11110000B  ; centesima de segundo
00CD D28C                   SETB    TR0             ; Arrancamos el timer 0
```



```
00CF 7E00                   MOV     R6,#0D          ; Segundos = 0
00D1 7F00                   MOV     R7,#0D          ; Centesimas = 0
00D3 22                     RET

                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;Rutina que habilita la interrupcion del timer 0 y el evento externo 0;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ; Escribe el instante actual guardado de R4 a R7
00D4                INTCONF:
00D4 D2A8                   SETB    EX0             ; Habilitamos la interrupcion externa 0
00D6 D288                   SETB    IT0
00D8 D2A9                   SETB    ET0             ; Habilitamos la interrupcionde del
timer 0
00DA D2AF                   SETB    EA
00DC 22                     RET


                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;Rutina de inicializacion del timer0;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
00DD                BLANK_TIMER:
00DD 758CD8                 MOV     TH0,#11011000B  ; Configuramos el timer0 para que se
desborde cada
00E0 758AF0                 MOV     TL0,#11110000B  ; centesima de segundo
00E3 22                     RET

                    ;Rutinas para manejo del reloj

                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;Rutina para incremento de horas;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
00E4                INCR_H:
00E4 C0F0                   PUSH    B               ; Guardamos B
00E6 EC                     MOV     A,R4            ; Guardamos el valor para operar
```

```
00E7 2401                    ADD     A,#1D           ; Incrementamos en 1 los segundos
00E9 F5F0                    MOV     B,A             ; Salvaguardamos el nuevo numero
00EB 24E8                    ADD     A,#-24D         ; Le restamos 24
00ED B40003                  CJNE    A,#0D,OK_H      ; Si A == 0, reiniciamos cuenta
00F0 75F000                  MOV     B,#0D           ; Reiniciamos las horas
00F3 ACF0            OK_H:   MOV     R4,B            ; Restauramos los horas
00F5 D0F0                    POP     B               ; Recuperamos B
00F7 22                      RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina para incremento de minutos;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
00F8                 INCR_M:
00F8 C0F0                    PUSH    B               ; Guardamos B
00FA ED                      MOV     A,R5            ; Guardamos el valor para operar
00FB 2401                    ADD     A,#1D           ; Incrementamos en 1 los segundos
00FD F5F0                    MOV     B,A             ; Salvaguardamos el nuevo numero
00FF 24C4                    ADD     A,#-60D         ; Le restamos 60
0101 B40006                  CJNE    A,#0D,OK_M      ; Si A == 0, reiniciamos cuenta
0104 75F000                  MOV     B,#0D           ; Reiniciamos las centesimas
0107 120000   F              CALL    INCR_H          ; Incrementamos las horas
010A ADF0            OK_M:   MOV     R5,B            ; Restauramos los minutos
010C D0F0                    POP     B               ; Recuperamos B
010E 22                      RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina para incremento de segundos;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
010F                 INCR_S:
010F C0F0                    PUSH    B               ; Guardamos B
0111 EE                      MOV     A,R6            ; Guardamos el valor para operar
0112 2401                    ADD     A,#1D           ; Incrementamos en 1 los segundos
0114 F5F0                    MOV     B,A             ; Salvaguardamos el nuevo numero
0116 24C4                    ADD     A,#-60D         ; Le restamos 60
```



```
0118 B40006                  CJNE    A,#0D,OK_S      ; Si A == 0, reiniciamos cuenta
011B 75F000                  MOV     B,#0D           ; Reiniciamos las centesimas
011E 120000   F              CALL    INCR_M          ; Incrementamos los minutos
0121 AEF0            OK_S:   MOV     R6,B            ; Restauramos los segundos
0123 D0F0                    POP     B               ; Recuperamos B
0125 22                      RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina para incremento de centesimas de segundo;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0126                 INCR_C:
0126 C0F0                    PUSH    B               ; Guardamos B
0128 EF                      MOV     A,R7            ; Guardamos el valor para operar
0129 2401                    ADD     A,#1D           ; Incrementamos en 1 las centesimas
012B F5F0                    MOV     B,A             ; Salvaguardamos el nuevo numero
012D 249C                    ADD     A,#-100D        ; Le restamos 100
012F B40006                  CJNE    A,#0D,OK_C      ; Si A == 0, reiniciamos cuenta
0132 75F000                  MOV     B,#0D           ; Reiniciamos las centesimas
0135 120000   F              CALL    INCR_S          ; Incrementamos los segundos
0138 AFF0            OK_C:   MOV     R7,B            ; Restauramos las centesimas
013A D0F0                    POP     B               ; Recuperamos B
013C 22                      RET

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ;Rutina que trata las interrupciones del timer;
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
013D                 DESBORD:
013D C0D0                    PUSH    PSW             ; metemos dentro de la pila la palabra
de estado
013F C083                    PUSH    DPH
0141 C082                    PUSH    DPL
0143 F5F0                    MOV     B,A
0145 C0F0                    PUSH    B
0147 C28D                    CLR     TF0
0149 120000   F              CALL    BLANK_TIMER     ; Reiniciamos el timer 0
014C 120000   F              CALL    INCR_C          ; Incrementamos centesimas
014F EB                      MOV     A,R3            ; Movemos las pulsaciones a A
```

```
0150 75F001          296             MOV     B,#1H           ; Comprobamos si estamos en modo Lapsus
(una pulsacion)
0153 95F0            297             SUBB    A,B
0155 6009            298             JZ      LAPSUS          ; Si estamos en lapsus, no motramos el
nuevo instante
0157 120000   F      299             CALL    PRINT_C         ; Escribimos el tiempo actual
transcurrido
015A 900000   F      300             MOV     DPTR,#MSG9      ; Iniciamos el cursor
015D 120000   F      301             CALL    PUTSTRING
0160 D0F0            302     LAPSUS: POP     B
0162 E5F0            303             MOV     A,B
0164 D082            304             POP     DPL
0166 D083            305             POP     DPH
0168 D0D0            306             POP     PSW             ; sacamos de la pila la palabra de
estado
016A 32              307             RETI
                     308
                     309
                     310     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     311     ;Rutina que trata las interrupciones del pulsador;
                     312     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     313     ; Incrementa en 1 el registro de numero de pulsaciones (R3)
016B                 314     PULSACION:
016B C0D0            315             PUSH    PSW             ; Metemos dentro de la pila la palabra
de estado
016D C083            316             PUSH    DPH
016F C082            317             PUSH    DPL
0171 F5F0            318             MOV     B,A
0173 C0F0            319             PUSH    B
0175 EB              320             MOV     A,R3            ; Movemos el numero de pulsaciones a A
0176 75F001          321             MOV     B,#1H           ; Movemos 1 al registro B
0179 25F0            322             ADD     A,B             ; Sumamos 1 al numero de pulsaciones
(R3)

A51 MACRO ASSEMBLER  RELOJ                                                              09/12/03
06:37:31 PAGE     6

017B FB              323             MOV     R3,A            ; Movemos el resultado a R3
017C D0F0            324             POP     B
017E E5F0            325             MOV     A,B
0180 D082            326             POP     DPL
0182 D083            327             POP     DPH
0184 D0D0            328             POP     PSW             ; Sacamos de la pila la palabra de
estado
0186 32              329             RETI
                     330
                     331     ;;;;;;;;;;;;;;;;;;;;;;;
                     332     ;Comienzo del programa;
                     333     ;;;;;;;;;;;;;;;;;;;;;;;
                     334     ; Selecciona el modo adecuado de operacion del programa
0187                 335     START:
0187 758100   F      336             MOV     SP,#STACK-1     ; Inicializamos el registro SP
                     337
                     338     ; Inicializamos el puerto serie
018A 759850          339             MOV     SCON,#01010000B
018D 758920          340             MOV     TMOD,#00100000B ; C/T = 0, Mode = 2
0190 758DF3          341             MOV     TH1,#0F3H
0193 D28E            342             SETB    TR1
0195 D299            343             SETB    TI
                     344
                     345     ; Preguntamos el modo de operacion (reloj o cronometro)
0197                 346     REP1:
0197 900000   F      347             MOV     DPTR,#MSG1      ; Cargamos el mensaje en DPTR
019A 120000   F      348             CALL    PUTSTRING       ; Escribimos la cadena de texto
019D 120000   F      349             CALL    GETCHAR         ; Leemos el modo de operacion en A
01A0 F5F0            350             MOV     B,A             ; Salvamos A en B
                     351     ; Comprobamos si entramos en modo crono
01A2 24BD            352             ADD     A,#-43H         ; Comprobamos si el caracter es 'C'
01A4 6016            353             JZ      CRONO           ; Si es 'C' saltamos a modo crono
01A6 E5F0            354             MOV     A,B             ; Restauramos A
01A8 249D            355             ADD     A,#-63H         ; Comprobamos si el caracter es 'c'
01AA 6010            356             JZ      CRONO           ; Si es 'c' saltamos a modo crono
01AC E5F0            357             MOV     A,B             ; Restauramos A
                     358     ; Comprobamos si entramos en modo reloj
01AE 24AE            359             ADD     A,#-52H         ; Comprobamos si el caracter es 'R'
01B0 6035            360             JZ      RELOJ           ; Si es 'R' saltamos a modo reloj
01B2 E5F0            361             MOV     A,B             ; Restauramos A
01B4 248E            362             ADD     A,#-72H         ; Comprobamos si el caracter es 'r'
```

```
01B6 602F            363             JZ      RELOJ           ; Si es 'r' saltamos a modo reloj
01B8 E5F0            364             MOV     A,B             ; Restauramos A
                     365     ; Si no es ninguna opcion de las anteriores volvemos a preguntar
01BA 80DB            366             SJMP    REP1            ; Si no indican una opcion valida,
repetimos
                     367
                     368     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     369     ; Rutina que simula el funcionamiento de un cronometro;
                     370     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     371     ;  R3 ->Numero de pulsaciones, R4 ->Horas, R5 ->Minutos, R6 ->Segundos,
R7 ->Centesimas
01BC                 372     CRONO:
01BC 900000   F      373             MOV     DPTR,#MSG3      ; Cargamos el modo en DPTR
01BF 120000   F      374             CALL    PUTSTRING       ; Lo escribimos
01C2 120000   F      375             CALL    INTCONF         ; Habilitamos interrupciones
01C5 120000   F      376     RESET:  CALL    TIMECONF        ; Configuramos el timer
01C8 7B00            377             MOV     R3,#0H          ; Pulsaciones = 0
01CA 7C00            378             MOV     R4,#0H          ; Horas = 0
01CC 7D00            379             MOV     R5,#0H          ; Minutos = 0
01CE                 380     CHECK:
01CE 8BF0            381             MOV     B,R3
01D0 20F102          382             JB      B.1,STOP
01D3 80F9            383             SJMP    CHECK           ; Si no estamos en el modo STOP
seguimos esperando
                     384
01D5                 385     STOP:
01D5 C28C            386             CLR     TR0             ; Paramos el Timer
01D7 120000   F      387             CALL    PRINT_C         ; Mostramos el instante actual
01DA 900000   F      388             MOV     DPTR,#MSG9      ; Iniciamos el carro

A51 MACRO ASSEMBLER  RELOJ                                                              09/12/03
06:37:31 PAGE     7

01DD 120000   F      389             CALL    PUTSTRING
01E0 8BF0            390     WAIT:   MOV     B,R3
01E2 20F0E0          391             JB      B.0, RESET
01E5 80F9            392             SJMP    WAIT            ; Si no estamos seguimos esperando
                     393
                     394
                     395     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     396     ; Rutina que simula el funcionamiento de un reloj;
                     397     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     398     ; R4 -> Horas, R5 -> Minutos, R6 -> Segundos, R7 -> Centesimas
01E7                 399     RELOJ:
01E7 900000   F      400             MOV     DPTR,#MSG4      ; Cargamos el modo en DPTR
01EA 120000   F      401             CALL    PUTSTRING       ; Lo escribimos
                     402     ; Pedimos al usuario que ponga el reloj en hora
                     403     ; HORAS
01ED                 404     REP2:
01ED 900000   F      405     BACK1:  MOV     DPTR,#MSG5      ; Cargamos el mensaje en DPTR
01F0 120000   F      406             CALL    PUTSTRING
01F3 120000   F      407             CALL    GETCHAR         ; Leemos el primer numero
01F6 FB              408             MOV     R3,A            ; Guardamos el caracter
01F7 120000   F      409             CALL    CONVERT         ; Lo convertimos a numero
01FA AAF0            410             MOV     R2,B            ; Cargamos en R2 el resultado de la
conversion
01FC BA0A02          411             CJNE    R2,#10D,D1OK    ; Si no es un digito, volvemos a
pedirlo
01FF 80EC            412             SJMP    BACK1
0201 75F00A          413     D1OK:   MOV     B,#10D          ; Cargamos el 10 en B
0204 A4              414             MUL     AB              ; Multiplicamos el primer digito por 10
0205 F9              415             MOV     R1,A            ; Cargamos el primer digito en R1
0206 EB              416             MOV     A,R3            ; Cargamos el digito para escribirlo
0207 120000   F      417             CALL    PUTCHAR         ; Escribimos el digito leido
                     418
020A 120000   F      419     BACK2:  CALL    GETCHAR         ; Leemos el segundo numero
020D FB              420             MOV     R3,A            ; Guardamos el caracter
020E 120000   F      421             CALL    CONVERT         ; Lo convertimos a numero
0211 FA              422             MOV     R2,A            ; Cargamos en R2 el resultado de la
conversion
0212 BA0A02          423             CJNE    R2,#10D,D2OK
0215 80D6            424             SJMP    BACK1           ; Si no es un digito, volvemos a
pedirlo
0217                 425     D2OK:
0217 EB              426             MOV     A,R3            ; Cargamos el digito para escribirlo
0218 120000   F      427             CALL    PUTCHAR         ; Escribimos el segundo digito de las
horas
021B 900000   F      428             MOV     DPTR,#MSG7      ; Saltamos de linea
```

```
021E 120000   F      429             CALL    PUTSTRING
0221 EA              430             MOV     A,R2            ; Restauramos el digito en A
0222 29              431             ADD     A,R1            ; Sumamos ambas cantidades, hemos
obtenido las horas
0223 F5F0            432             MOV     B,A             ; Salvaguardamos A en B
0225 54E0            433             ANL     A,#11100000B    ; Aplicamos mascara para ver que el
numero no sea muy grand
                                     e
0227 B400C3          434             CJNE    A,#0D,BACK1     ; Si el resultado es 0 OK, sino
releemos
022A E5F0            435             MOV     A,B             ; Restauramos A desde B
022C 5418            436             ANL     A,#00011000B    ; Aplicamos mascara para comprobar que
los dos bits no este
                                     n a 1 simultaneamente
022E B41802          437             CJNE    A,#24D,SAVE_H   ; Si el resultado es 0 OK
0231 80BA            438             SJMP    BACK1           ; No es correcto, releemos
0233 ACF0            439     SAVE_H: MOV     R4,B            ; Guardamos la hora valida en R4
                     440     ;Si la hora es correcta, pasamos a leer los minutos
                     441
                     442     ; MINUTOS
0235                 443     REP3:
0235 900000   F      444             MOV     DPTR,#MSG6      ; Cargamos el mensaje en DPTR
0238 120000   F      445             CALL    PUTSTRING
023B 120000   F      446             CALL    GETCHAR         ; Leemos el primer numero
023E F8              447             MOV     R0,A            ; Guardamos el caracter
023F 120000   F      448             CALL    CONVERT         ; Lo convertimos a numero
0242 AAF0            449             MOV     R2,B            ; Cargamos en R2 el resultado de la
conversion
0244 BA0A02          450             CJNE    R2,#10D,D3OK    ; Si no es un digito, volvemos a
pedirlo
0247 80EC            451             SJMP    REP3
0249 75F00A          452     D3OK:   MOV     B,#10D          ; Cargamos el 10 en B

A51 MACRO ASSEMBLER  RELOJ                                                          09/12/03
06:37:31 PAGE     8

024C A4              453             MUL     AB              ; Multiplicamos el primer digito por 10
024D F9              454             MOV     R1,A            ; Cargamos el primer digito en R1
024E E8              455             MOV     A,R0            ; Cargamos el digito para escribirlo
024F 120000   F      456             CALL    PUTCHAR         ; Escribimos el digito leido
                     457
0252 120000   F      458             CALL    GETCHAR         ; Leemos el segundo numero
0255 F8              459             MOV     R0,A            ; Guardamos el caracter
0256 120000   F      460             CALL    CONVERT         ; Lo convertimos a numero
0259 FA              461             MOV     R2,A            ; Cargamos en R2 el resultado de la
conversion
025A BA0A02          462             CJNE    R2,#10D,D4OK
025D 80D6            463             SJMP    REP3            ; Si no es un digito, volvemos a
pedirlo
025F                 464     D4OK:
025F E8              465             MOV     A,R0            ; Cargamos el digito para escribirlo
0260 120000   F      466             CALL    PUTCHAR         ; Escribimos el segundo digito de los
minutos
0263 900000   F      467             MOV     DPTR,#MSG7      ; Saltamos de linea
0266 120000   F      468             CALL    PUTSTRING
0269 EA              469             MOV     A,R2            ; Restauramos el digito en A
026A 29              470             ADD     A,R1            ; Sumamos ambas cantidades, hemos
obtenido los minutos
026B F5F0            471             MOV     B,A             ; Salvaguardamos A en B
026D 54C0            472             ANL     A,#11000000B    ; Aplicamos mascara para ver que el
numero no sea muy grand
                                     e
026F B400C3          473             CJNE    A,#0D,REP3      ; Si el resultado es 0 OK, sino
releemos
0272 E5F0            474             MOV     A,B             ; Restauramos A desde B
0274 543C            475             ANL     A,#00111100B    ; Aplicamos mascara para comprobar que
los cuatro bits no e
                                     sten a 1 simultaneamente
0276 B43C02          476             CJNE    A,#60D,SAVE_M   ; Si el resultado es 0 OK
0279 80BA            477             SJMP    REP3            ; Si los minutos no son correctos,
reintentamos
027B ADF0            478     SAVE_M: MOV     R5,B            ; Guardamos los minutos validos en R5
027D 120000   F      479             CALL    TIMECONF        ; Configuramos el timer
                     480     ; Nos metemos en un bucle, comprobamos desbordamiento del timer
0280                 481     REP4:
0280 308DFD          482             JNB     TF0,REP4        ; Comprobamos desbordamiento en Timer0
0283 C28D            483             CLR     TF0
0285 120000   F      484             CALL    BLANK_TIMER     ; Reiniciamos el timer 0
```

```
0288 120000   F      485             CALL    INCR_C          ; Incrementamos centesimas
028B 900000   F      486             MOV     DPTR,#MSG8      ; Escribimos la cadena de texto inicial
028E 120000   F      487             CALL    PUTSTRING
0291 120000   F      488             CALL    PRINT_R         ; Imprimimos el reloj
0294 900000   F      489             MOV     DPTR,#MSG9      ; Iniciamos el carro
0297 120000   F      490             CALL    PUTSTRING
029A 80E4            491             SJMP    REP4
                     492
                     493
029C BF4D4F44        494     MSG1:   DB      '¿MODO RELOJ O CRONOMETRO? (R/C):',CR,LF,00H
02A0 4F205245
02A4 4C4F4A20
02A8 4F204352
02AC 4F4E4F4D
02B0 4554524F
02B4 3F202852
02B8 2F43293A
02BC 0D0A00
02BF 4D4F444F        495     MSG2:   DB      'MODO: ',00H
02C3 3A2000
02C6 4D4F444F        496     MSG3:   DB      'MODO CRONOMETRO',CR,LF,00H
02CA 2043524F
02CE 4E4F4D45
02D2 54524F0D
02D6 0A00
02D8 4D4F444F        497     MSG4:   DB      'MODO RELOJ',CR,LF,00H
02DC 2052454C
02E0 4F4A0D0A
02E4 00
02E5 494E4449        498     MSG5:   DB      'INDIQUE LA HORA (HH):',CR,LF,00H
02E9 51554520
02ED 4C412048

A51 MACRO ASSEMBLER  RELOJ                                                      09/12/03
06:37:31 PAGE     9

02F1 4F524120
02F5 28484829
02F9 3A0D0A00
02FD 494E4449        499     MSG6:   DB      'INDIQUE LOS MINUTOS (MM):',CR,LF,00H
0301 51554520
0305 4C4F5320
0309 4D494E55
030D 544F5320
0311 284D4D29
0315 3A0D0A00
0319 0D0A00          500     MSG7:   DB      CR,LF,00H
031C 484F5241        501     MSG8:   DB      'HORA: ',00H
0320 3A2000
0323 0D00            502     MSG9:   DB      CR,00H
                     503
0325                 504     FIN:
                     505             END

A51 MACRO ASSEMBLER  RELOJ                                                      09/12/03
06:37:31 PAGE    10

SYMBOL TABLE LISTING
------ ----- -------


N A M E             T Y P E  V A L U E   ATTRIBUTES

B. . . . . .  D ADDR   00F0H   A
BACK1. . . . . .  C ADDR   01EDH   R   SEG=PROG
BACK2. . . . . .  C ADDR   020AH   R   SEG=PROG
BLANK_TIMER. . . .  C ADDR   00DDH   R   SEG=PROG
CHECK. . . . . .  C ADDR   01CEH   R   SEG=PROG
CINCO. . . . . .  C ADDR   0044H   R   SEG=PROG
CONVERT. . . . . .  C ADDR   0029H   R   SEG=PROG
CR . . . . . .  N NUMB   000DH   A
CRONO. . . . . .  C ADDR   01BCH   R   SEG=PROG
CUATRO . . . . . .  C ADDR   003FH   R   SEG=PROG
D1OK . . . . . .  C ADDR   0201H   R   SEG=PROG
D2OK . . . . . .  C ADDR   0217H   R   SEG=PROG
D3OK . . . . . .  C ADDR   0249H   R   SEG=PROG
D4OK . . . . . .  C ADDR   025FH   R   SEG=PROG
DESBORD. . . . . .  C ADDR   013DH   R   SEG=PROG
```

```
DOS. . . . . .  C ADDR   0035H   R   SEG=PROG
DPH. . . . . .  D ADDR   0083H   A
DPL. . . . . .  D ADDR   0082H   A
EA . . . . . .  B ADDR   00A8H.7 A
ET0. . . . . .  B ADDR   00A8H.1 A
EX0. . . . . .  B ADDR   00A8H.0 A
EXIT . . . . .  C ADDR   0022H   R   SEG=PROG
FAIL . . . . .  C ADDR   005DH   R   SEG=PROG
FIN. . . . . .  C ADDR   0325H   R   SEG=PROG
GETASCII . . . .  C ADDR   0069H   R   SEG=PROG
GETCHAR. . . . .  C ADDR   0008H   R   SEG=PROG
INCR_C . . . . .  C ADDR   0126H   R   SEG=PROG
INCR_H . . . . .  C ADDR   00E4H   R   SEG=PROG
INCR_M . . . . .  C ADDR   00F8H   R   SEG=PROG
INCR_S . . . . .  C ADDR   010FH   R   SEG=PROG
INTCONF. . . . .  C ADDR   00D4H   R   SEG=PROG
IT0. . . . . .  B ADDR   0088H.0 A
LAPSUS . . . . .  C ADDR   0160H   R   SEG=PROG
LF . . . . . .  N NUMB   000AH   A
MSG1 . . . . . .  C ADDR   029CH   R   SEG=PROG
MSG2 . . . . . .  C ADDR   02BFH   R   SEG=PROG
MSG3 . . . . . .  C ADDR   02C6H   R   SEG=PROG
MSG4 . . . . . .  C ADDR   02D8H   R   SEG=PROG
MSG5 . . . . . .  C ADDR   02E5H   R   SEG=PROG
MSG6 . . . . . .  C ADDR   02FDH   R   SEG=PROG
MSG7 . . . . . .  C ADDR   0319H   R   SEG=PROG
MSG8 . . . . . .  C ADDR   031CH   R   SEG=PROG
MSG9 . . . . . .  C ADDR   0323H   R   SEG=PROG
NUEVE. . . . . .  C ADDR   0058H   R   SEG=PROG
OCHO . . . . . .  C ADDR   0053H   R   SEG=PROG
OK . . . . . .  C ADDR   0066H   R   SEG=PROG
OK2. . . . . .  C ADDR   0068H   R   SEG=PROG
OK_C . . . . . .  C ADDR   0138H   R   SEG=PROG
OK_H . . . . . .  C ADDR   00F3H   R   SEG=PROG
OK_M . . . . . .  C ADDR   010AH   R   SEG=PROG
OK_S . . . . . .  C ADDR   0121H   R   SEG=PROG
PRINT_C. . . . .  C ADDR   00AFH   R   SEG=PROG
PRINT_R. . . . .  C ADDR   0084H   R   SEG=PROG
PROG . . . . . .  C SEG    0325H       REL=UNIT
PSW. . . . . .  D ADDR   00D0H   A
PULSACION. . . .  C ADDR   016BH   R   SEG=PROG
PUTCHAR. . . . .  C ADDR   0000H   R   SEG=PROG
PUTSEPARATOR . .  C ADDR   0023H   R   SEG=PROG
PUTSTRING. . . .  C ADDR   0018H   R   SEG=PROG
RELOJ. . . . . .  C ADDR   01E7H   R   SEG=PROG

A51 MACRO ASSEMBLER  RELOJ                                                    09/12/03
06:37:31 PAGE    11

REP1 . . . . . .  C ADDR   0197H   R   SEG=PROG
REP2 . . . . . .  C ADDR   01EDH   R   SEG=PROG
REP3 . . . . . .  C ADDR   0235H   R   SEG=PROG
REP4 . . . . . .  C ADDR   0280H   R   SEG=PROG
RESET. . . . . .  C ADDR   01C5H   R   SEG=PROG
RI . . . . . .  B ADDR   0098H.0 A
SAVE_H . . . . .  C ADDR   0233H   R   SEG=PROG
SAVE_M . . . . .  C ADDR   027BH   R   SEG=PROG
SBUF . . . . . .  D ADDR   0099H   A
SCON . . . . . .  D ADDR   0098H   A
SEIS . . . . . .  C ADDR   0049H   R   SEG=PROG
SIETE. . . . . .  C ADDR   004EH   R   SEG=PROG
SP . . . . . .  D ADDR   0081H   A
STACK. . . . . .  I SEG    0010H       REL=UNIT
START. . . . . .  C ADDR   0187H   R   SEG=PROG
STOP . . . . . .  C ADDR   01D5H   R   SEG=PROG
TF0. . . . . .  B ADDR   0088H.5 A
TH0. . . . . .  D ADDR   008CH   A
TH1. . . . . .  D ADDR   008DH   A
TI . . . . . .  B ADDR   0098H.1 A
TIMECONF . . . .  C ADDR   00C4H   R   SEG=PROG
TL0. . . . . .  D ADDR   008AH   A
TMOD . . . . . .  D ADDR   0089H   A
TR0. . . . . .  B ADDR   0088H.4 A
TR1. . . . . .  B ADDR   0088H.6 A
TRES . . . . . .  C ADDR   003AH   R   SEG=PROG
UNO. . . . . .  C ADDR   0030H   R   SEG=PROG
WAIT . . . . . .  C ADDR   01E0H   R   SEG=PROG
```

```
REGISTER BANK(S) USED: 0

ASSEMBLY COMPLETE.  0 WARNING(S), 0 ERROR(S)
```

Rutinas utilizadas:

A continuación se explican detalladamente todas y cada una de las rutinas utilizadas en este programa.

- PUTCHAR: rutina que escribe el carácter almacenado en el registro A, en el puerto serie. Para ello movemos el contenido de A, a SBUF y esperamos hasta que la línea no esté ocupada.
- GETCHAR: rutina que lee un carácter ASCII del puerto serie, espera a que se escriba algo en el puerto, lee del puerto, comprueba que el carácter no es 11H, ya que a veces es enviado por el mon51, y por último lo almacena en A.
- PUTSTRING: escribe una cadena de caracteres por el puerto serie. La cadena de caracteres debe estar apuntada por DPTR. La rutina lee carácter a carácter desde DPTR y llama a la rutina PUTCHAR, hasta que llegue un carácter nulo (00H).
- PUTSEPARATOR: escribe el carácter ":" en el puerto serie, moviendo el equivalente ASCII de dicho carácter al registro A y llamando a PUTCHAR.
- CONVERT: rutina que convierte un carácter ASCII a su equivalente decimal, si el carácter no es un número devuelve un 10 en el registro B, en caso contrario, devuelve el valor de dicho carácter, tanto en el registro B, como en el A. Para ello comprobamos el valor del carácter, con el valor ASCII de los dígitos del 0 al 9, y si coincide con alguno, lo devolvemos en A y en B.
- GETASCII: rutina que obtiene de A un numero de dos dígitos, y obtiene el primer dígito dividiendo A entre 10 y el segundo dígito restando el primer dígito multiplicado por 10, al número total. Devuelve las decenas en A y las unidades en B.
- PRINT_R: escribe la hora actual en el puerto serie. Obtiene la hora actual del registro R4, llama a la rutina GETASCII y luego escribe las decenas almacenadas en A y las unidades en B, escribe ":" llamando a PUTSEPARATOR. Realiza la misma acción para los minutos (R5) y los segundos (R6).
- PRINT_C: escribe el instante actual del cronómetro en el puerto serie y en el siguiente formato hh:mm:ss″cc. Para ello llamamos a la rutina PRINT_R que envia hh:mm:ss, y a continuación escribimos " con PUTCHAR y las centésimas que se encuentran en el registro R7 con GETASCII y PUTCHAR.
- TIMECONF: rutina que configura el Timer 0. Elegimos el modo 1 en TMOD, configuramos el Timer 0 para que se desborde cada centésima de segundo, lo arrancamos e inicializamos los segundos y las centésimas.
- INTCONF: rutina que habilita la interrupción del Timer 0 y del evento externo 0, poniendo a 1 los bits ET0, IT0, EX0, y EA.
- BLANK_TIMER: configura el Timer 0 para que se desborde cada centésima, del mismo modo que lo hace TIMECONF.
- INCR_H: rutina que incrementa en una unidad el número de horas almacenadas en R4, suma 1 al número de horas, le resta 24 y si el resultado es cero inicializa a cero las horas.
- INCR_M: rutina que incrementa en una unidad el número de minutos almacenados en R5, suma 1 al número de minutos, le resta 60 y si el resultado

es cero inicializa a cero los minutos e incrementa las horas.

- INCR_S: rutina que incrementa en una unidad el número de segundos almacenadas en R6, suma 1 al número de segundos, le resta 60 y si el resultado es cero inicializa a cero los segundos, e incrementa los minutos.
- INCR_C: rutina que incrementa en una unidad el número de centésimas almacenadas en R7, suma 1 al número de centésimas, le resta 100 y si el resultado es cero inicializa a cero las centésimas e incrementa los segundos.
- DESBORD: rutina que trata la interrupción de desbordamiento del Timer 0 cuando estamos en modo cronómetro. La rutina guarda la palabra de estado en la pila y el registro B (que puede estar siendo usado por el programa principal), reinicia el Timer 0 para que se desborde cada centésima, e incrementa las centésimas. A continuación envía al puerto serie el instante del crono llamando a PRINT_C, salvo en el caso de que el numero de pulsaciones del botón (R3) sea igual a 1, ya que estaríamos en el modo LAPSUS, en el que el programa no actualizará la hora en el puerto serie, aunque el timer seguirá funcionando y actualizando la hora correctamente. Por último restaura el registro B y la palabra de estado.
- PULSACION: rutina que trata la interrupción generada por el evento externo 0. Al saltar la interrupción se guarda en la pila la palabra de estado y el registro B (que puede estar siendo usado por el programa principal), y a continuación incrementa en 1 en número de pulsaciones almacenado en R3, por último restaura los elementos de la pila.
- RELOJ: segmento de código que simula el funcionamiento de un reloj. Se salta a este segmento si el modo elegido para el programa es el modo reloj. Lo primero que hace es mostrar un mensaje indicando el modo en el que estamos por el puerto serie, y a continuación envía al puerto serie una cadena de texto indicando que se introduzca la hora. La rutina lee el primer digito de la hora por el puerto serie y lo convierte a número decimal llamando a CONVERT, haciendo lo mismo para el segundo dígito, en caso de que alguno de los dos caracteres no sea equivalente a un número decimal, se volverá a enviar al serie un mensaje con PUTSTRING indicando que se vuelva a introducir la hora. En caso de que la hora esté formada por dos dígitos se pasa a la comprobación de si la hora es una hora correcta para el formato del reloj, es decir, que esté entre 0 y 23. Para ello aplicamos una máscara de bits sobre el número introducido y comprobamos si supera el límite (23), en caso de que lo supere volvemos a pedirlo por el serie, en caso contrario lo almacenamos en R4, y pasamos a la lectura de los minutos. La comprobación de los minutos funciona del mismo modo que la de las horas, salvo que la máscara que aplicamos es para comprobar que no se supera el número 60, en caso de que el formato sea correcto lo almacenaremos en R5.

  Ya tenemos la hora en R4 y el minuto en R5, en R6 se almacenarán los segundos que en un principio comienzan a 0. En el modo reloj no tiene activadas las interrupciones del Timer 0 (el modo cronómetro si las usa), por lo que para comprobar si se desborda el timer 0, estaremos continuamente observando el bit TF0. Cuando se active sabemos que ha transcurrido una centésima de segundo, volvemos a poner manualmente el bit TF0 a 0, incrementamos el numero de centésimas llamando a INCR_C, y mostramos la hora con el siguiente formato (“HORA: ” ,h, “:”,m,“:”,s), retornando el carro pero sin realizar el salto de línea, volvemos a comprobar el bit TF0, y así sucesivamente.

- CRONO: éste es el segmento de código análogo al reloj que simula el funcionamiento de un cronómetro. Comienza mostrando el modo en que nos encontramos, a continuación activa las interrupciones del Timer 0 y del evento externo 0 al que se conectará el pulsador, configura el Timer 0 para que se desborde cada centésima de segundo, inicializa en numero de pulsaciones (R3), y el instante actual (0:0:0″0). A continuación permanece en un bucle comprobando si se ha accionado el pulsador 2 veces (en caso de que se haya pulsado una vez, modo LAPSUS) la rutina de interrupciones del Timer 0 se encarga de que el número no se actualice en el puerto serie, y si no se ha pulsado ninguna vez funcionará como si se tratara de un reloj, pero mostrando las centésimas de segundo en este formato h:m:s″c). Para ver si se ha pulsado 2 veces el botón, comprobamos si el bit de la posición 1 del registro que guarda el número de pulsaciones esta activo (en caso de que fuera mayor de 2 no estaríamos ejecutando esta parte del programa), si lo esta, entramos en el modo STOP, en el que paramos el Timer 0, mostramos el instante actual del crono, y permanecemos a la espera de la siguiente pulsación (tercera) del botón que nos llevaría de nuevo al inicio del modo CRONO, para esto comprobamos el bit 0 del registro que almacena las pulsaciones (al entrar en este modo el registro de pulsaciones tiene el valor 2), si está a 1 reseteamos el crono, y el numero de pulsaciones y comienza de nuevo la cuenta desde cero, a la espera de una nueva pulsación para volver a entrar en el modo LAPSUS, y así sucesivamente.

Problemas encontrados:

Los mayores problemas encontrados a la hora de la realización de la práctica han sido los siguientes:

Bug en mon51: a causa de un bug en el módulo mon51 utilizado para simular el funcionamiento del programa directamente en el microprocesador, periódicamente se recibe por el puerto serie un caracter 11H.

Falta de material: a la hora de utilizar el material para comprobar la práctica nos hemos encontrado con la falta tanto de una fuente de alimentación (la que se nos proporcionó no funcionaba) como de un cable para conectar el 8051 al puerto serie de un ordenador.

Tiempo de ejecución: dado que el módulo del reloj se realizó sin utilizar interrupciones por desbordamiento del timer, su ejecución en el simulador era prácticamente igual en cuanto a la temporización que ejecutando directamente sobre el microprocesador. Sin embargo, el cronómetro se codificó utilizado interrupciones por desbordamiento del timer para medir el tiempo, de modo que la ejecución en el simulador era mucho más rápida.