



MICROPROCESADORES 2003/2004

PROGRAMACIÓN DE LCD 'S

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, todo ello controlado mediante un LCD.

El funcionamiento del programa es idéntico al de la anterior práctica, con la salvedad de que los mensajes mostrados únicamente por el puerto serie en dicha práctica, ahora se ven duplicados en el LCD conectado a la pequeña placa del 8051. Para información más detallada sobre el funcionamiento del programa y las rutinas no relacionadas con el manejo del LCD, consultar la memoria correspondiente a la mencionada práctica y el anexo incluido al final de este mismo documento, respectivamente.

Como sencilla indicación, decir que el paso de la anterior práctica a la presente es muy sencillo, ya que tan sólo se requiere programar las rutinas para escribir en el LCD, y utilizarlas a lo largo del programa en las mismas situaciones en las que se escriba algo en el puerto serie, con la salvedad de que los mensajes han de estar adaptados a la longitud del display (por lo que usualmente deberán ser más escuetos) y de que los retornos de carro no son reconocidos por el mismo.

Así mismo, se ha construido un pequeño circuito sobre la placa de inserción que conecta el LCD al 8051, siendo P5 los puertos por los que se envía el dato al LCD, y P4.5, P4.6 y P4.7 para las señales de control. El pulsador, al igual que en la anterior ocasión, emite sus interrupciones a través del evento externo 0.

Listado:

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

DOS MACRO ASSEMBLER A51 V5.28m
 OBJECT MODULE PLACED IN RELOJ2.OBJ
 ASSEMBLER INVOKED BY: C:\C51EVAL\BIN\A51.EXE RELOJ2.A51 DB EP

LOC	OBJ	LINE	SOURCE
		1	;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
		2	;Practica 3 de la asignatura de;
		3	; microprocesadores 2003/2004 ;
		4	;Autores:
		5	; Jaime Perez Crespo ;
		6	; Ruben Seijas Valverde ;
		7	;Ultima modificacion: 16-12-03 ;
		8	;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
		9	NAME RELOJ
		10	
		11	
		12	
		13	;Definicion de retorno de carro y nueva linea (CR y LF)
000D		14	CR EQU 0DH ; Carriage Return
000A		15	LF EQU 0AH ; Line Feed
0020		16	FLAG EQU 20H ; Flag para comprobar si los segundos
han cambiado			
		17	
		18	PROG SEGMENT CODE
		19	STACK SEGMENT IDATA
		20	
----		21	RSEG STACK
0000		22	DS 10H ; 16 bytes de pila
		23	
----		24	CSEG AT 0
		25	USING 0 ; Banco 0 de registros
		26	
0003		27	ORG 03h
0003 020000 F		28	LJMP PULSACION ; Rutina de la interrupcion de pulso del
boton			
		29	
000B		30	ORG 0Bh
000B 020000 F		31	LJMP DESBORD ; Rutina de la interrupcion de
desbordamiento del timer			
		32	
		33	;Tras reset se comienza a ejecutar sobre la direccion 0
0000		34	ORG 0H
0000 020000 F		35	JMP START ; Saltamos al comienzo del programa
----		36	RSEG PROG
		37	
00DB		38	P7 DATA 0DBh
00DD		39	P8 DATA 0DDh
00E8		40	P4 DATA 0E8h
00F8		41	P5 DATA 0F8h
00FA		42	P6 DATA 0FAh
		43	
00F8		44	DB0 EQU P5.0
00F9		45	DB1 EQU P5.1
00FA		46	DB2 EQU P5.2
00FB		47	DB3 EQU P5.3
00FC		48	DB4 EQU P5.4
00FD		49	DB5 EQU P5.5
00FE		50	DB6 EQU P5.6
00FF		51	DB7 EQU P5.7
		52	
00EF		53	EN EQU P4.7
00EE		54	RS EQU P4.6
00ED		55	RW EQU P4.5
00F8		56	DATO EQU P5
		57	
		58	;Inicialización

```

59
0000 60 WAIT_LCD:
0000 D2EF 61 SETB EN
0002 C2EE 62 CLR RS
0004 D2ED 63 SETB RW
0006 75F8FF 64 MOV DATO,#0FFh
0009 E5F8 65 MOV A,DATO
000B 20E7F2 66 JB ACC.7,WAIT_LCD
000E C2EF 67 CLR EN
0010 C2ED 68 CLR RW
0012 22 69 RET
0013 70
0013 D2EF 71 INIT_LCD:
0015 C2EE 72 SETB EN
0017 75F838 73 CLR RS
001A C2EF 74 MOV DATO,#38h
001C 120000 F 75 CLR EN
001F D2EF 76 LCALL WAIT_LCD
0021 C2EE 77 SETB EN
0023 75F80E 78 CLR RS
0026 C2EF 79 MOV DATO,#0Eh
0028 120000 F 80 CLR EN
002B D2EF 81 LCALL WAIT_LCD
002D C2EE 82 SETB EN
002F 75F806 83 CLR RS
0032 C2EF 84 MOV DATO,#06h
0034 120000 F 85 CLR EN
0037 22 86 LCALL WAIT_LCD
0038 87 RET
0038 C0F0 88 CLEAR_LCD:
003A F5F0 89 PUSH B
003C C0F0 90 MOV B,A
003E D2EF 91 PUSH B
0040 C2EE 92 SETB EN
0042 75F801 93 CLR RS
0045 C2EF 94 MOV DATO,#01h
0047 120000 F 95 CLR EN
004A D0F0 96 LCALL WAIT_LCD
004C E5F0 97 POP B
004E D0F0 98 MOV A,B
0050 22 99 POP B
0051 100 RET
0051 D2EF 101 WRITE_TEXT:
0053 D2EE 102 SETB EN
0055 F5F8 103 SETB RS
0057 C2EF 104 MOV DATO,A
0059 120000 F 105 CLR EN
005C 22 106 LCALL WAIT_LCD
106 RET
107
108
109 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
110 ;Rutina que escribe un caracter en el puerto serie;
111 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
112 ;El caracter esta almacenado en A
113 PUTCHAR:
005D 114 CLR TI ; TI = 0
005F F599 115 MOV SBUF,A ; Enviamos el caracter
0061 3099FD 116 JNB TI,$ ; Si TI == 0 entonces linea ocupada,
esperamos a TI == 1
117 ; CALL WRITE_TEXT
0064 22 118 RET
119
120 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
121 ;Rutina que escribe una cadena por el puerto serie;
122 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
123 ; Escribe la cadena apuntada en DPTR, y MODIFICA dicho puntero
0065 124 PUTSTRING:
A51 MACRO ASSEMBLER RELOJ2

```

```

0065 E4          125          CLR      A
0066 93          126          MOVC    A,@A+DPTR      ; Lee del segmento de codigo
0067 6006        127          JZ      EXIT          ; Sale si el caracter es 00H
0069 120000     F 128          CALL    PUTCHAR       ; Escribimos el caracter en el serie
006C A3          129          INC     DPTR          ; Incrementamos el contador
006D 80F6        130          SJMP   PUTSTRING
006F 22          131          EXIT:   RET
132
133          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
134          ;Rutina que escribe una cadena por el LCD;
135          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
136          ; Escribe la cadena apuntada en DPTR, y MODIFICA dicho puntero
0070          PUTLCD:
0070 E4          138          CLR      A
0071 93          139          MOVC    A,@A+DPTR      ; Lee del segmento de codigo
0072 6006        140          JZ      EXIT2         ; Sale si el caracter es 00H
0074 120000     F 141          CALL    WRITE_TEXT    ; Escribimos el caracter en el LCD
0077 A3          142          INC     DPTR          ; Incrementamos el contador
0078 80F6        143          SJMP   PUTLCD
007A 22          144          EXIT2:  RET
145
146          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
147          ;Rutina que escribe un caracter ':' por el puerto serie;
148          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
007B          PUTSEPARATOR:
007B 743A        150          MOV     A,#58D         ; Cargamos un : en A
007D 120000     F 151          CALL    PUTCHAR       ; Lo escribimos
0080 120000     F 152          CALL    WRITE_TEXT
0083 22          153          RET
154
155          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
156          ;Rutina que escribe la hora por el puerto serie;
157          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
158          ; Escribe la hora actual guardada de R4 a R6
0084          PRINT_R:
0084 EC          160          MOV     A,R4          ; Cargamos las horas
0085 120000     F 161          CALL    GETASCII     ; Las convertimos a digitos
0088 120000     F 162          CALL    PUTCHAR       ; Escribimos el primer digito
008B 120000     F 163          CALL    WRITE_TEXT    ; Escribimos en el LCD
008E E5F0        164          MOV     A,B
0090 120000     F 165          CALL    PUTCHAR       ; Escribimos el segundo digito
0093 120000     F 166          CALL    WRITE_TEXT    ; Escribimos en el LCD
0096 120000     F 167          CALL    PUTSEPARATOR  ; Escribimos un :
0099 ED          168          MOV     A,R5          ; Cargamos los minutos
009A 120000     F 169          CALL    GETASCII     ; Los convertimos a digitos
009D 120000     F 170          CALL    PUTCHAR       ; Escribimos el primer digito
00A0 120000     F 171          CALL    WRITE_TEXT    ; Escribimos en el LCD
00A3 E5F0        172          MOV     A,B
00A5 120000     F 173          CALL    PUTCHAR       ; Escribimos el segundo digito
00A8 120000     F 174          CALL    WRITE_TEXT    ; Escribimos en el LCD
00AB 120000     F 175          CALL    PUTSEPARATOR  ; Escribimos un :
00AE EE          176          MOV     A,R6          ; Cargamos los segundos
00AF 120000     F 177          CALL    GETASCII     ; Los convertimos a digitos
00B2 120000     F 178          CALL    PUTCHAR       ; Escribimos el primer digito
00B5 120000     F 179          CALL    WRITE_TEXT    ; Escribimos en el LCD
00B8 E5F0        180          MOV     A,B
00BA 120000     F 181          CALL    PUTCHAR       ; Escribimos el segundo digito
00BD 120000     F 182          CALL    WRITE_TEXT    ; Escribimos en el LCD
00C0 22          183          RET
184
185
186          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
187          ;Rutina que escribe el cronometro por el puerto serie;
188          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
189          ; Escribe el instante actual guardado de R4 a R7
00C1          PRINT_C:
A51 MACRO ASSEMBLER RELOJ2

```

```

00C1 120000 F 191 CALL CLEAR_LCD ; Limpiamos el LCD
00C4 120000 F 192 CALL PRINT_R ; Escribimos la hora (h:m:s)
00C7 7422 193 MOV A,#'"' ; Escribimos "
00C9 120000 F 194 CALL PUTCHAR
00CC 120000 F 195 CALL WRITE_TEXT ; Escribimos en el LCD
00CF EF 196 MOV A,R7 ; Cargamos las centesimas
00D0 120000 F 197 CALL GETASCII ; Los convertimos a digitos
00D3 120000 F 198 CALL PUTCHAR ; Escribimos el primer digito
00D6 120000 F 199 CALL WRITE_TEXT ; Escribimos en el LCD
00D9 E5F0 200 MOV A,B
00DB 120000 F 201 CALL PUTCHAR ; Escribimos el segundo digito
00DE 120000 F 202 CALL WRITE_TEXT ; Escribimos en el LCD
00E1 22 203 RET
204
205
206 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
207 ;Rutina que habilita la interrupcion del timer 0 y el evento externo 0;
208 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
209 ; Escribe el instante actual guardado de R4 a R7
00E2 210 INTCONF:
00E2 D2A8 211 SETB EX0 ; Habilitamos la interrupcion externa 0
00E4 D288 212 SETB IT0
00E6 D2A9 213 SETB ET0 ; Habilitamos la interrupcion de del
timer 0
00E8 D2AF 214 SETB EA
00EA 22 215 RET
216
217
218 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
219 ;Rutina que lee un caracter por el puerto serie;
220 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
221 ; El caracter leido se almacena en A
00EB 222 GETCHAR:
00EB 3098FD 223 JNB RI,$ ; Si RI == 0 entonces no se ha
recibido, esperamos a RI ==
1
00EE E599 224 MOV A,SBUF ; Leemos el caracter por el puerto
00F0 C298 225 CLR RI
226 ; Como mon51 envia 11H de vez en cuando lo ignoramos
00F2 F5F0 227 MOV B,A ; Salvaguardamos A
00F4 24EF 228 ADD A,#-11H ; Restamos 11H a A
00F6 60F3 229 JZ GETCHAR ; Si es cero leemos otro caracter
00F8 E5F0 230 MOV A,B ; Recuperamos A desde B
00FA 22 231 RET
232
233 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
234 ;Rutina que obtiene la equivalencia entre digitos ASCII y numeros;
235 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
236 ; #'0' == #48D, ... , #'9' == #57D
00FB 237 CONVERT:
00FB F5F0 238 MOV B,A ; Guardamos A en B
00FD B43002 239 CJNE A,#48D,UNO ; Si es X, no saltamos, ponemos X en A.
Si no, saltamos a X
+1
0100 7400 240 MOV A,#00D
0102 B43102 241 UNO: CJNE A,#49D,DOS
0105 7401 242 MOV A,#01D
0107 B43202 243 DOS: CJNE A,#50D,TRES
010A 7402 244 MOV A,#02D
010C B43302 245 TRES: CJNE A,#51D,CUATRO
010F 7403 246 MOV A,#03D
0111 B43402 247 CUATRO: CJNE A,#52D,CINCO
0114 7404 248 MOV A,#04D
0116 B43502 249 CINCO: CJNE A,#53D,SEIS
0119 7405 250 MOV A,#05D
011B B43602 251 SEIS: CJNE A,#54D,SIETE
011E 7406 252 MOV A,#06D
0120 B43702 253 SIETE: CJNE A,#55D,OCHO
0123 7407 254 MOV A,#07D
A51 MACRO ASSEMBLER RELOJ2

```

```

0125 B43802      255      OCHO:  CJNE   A,#56D,NUEVE
0128 7408        256              MOV    A,#08D
012A B43902      257      NUEVE:  CJNE   A,#57D,FAIL      ; Si no es 9, comprobamos si fue alguno
de los anteriores
012D 7409        258              MOV    A,#09D
012F B5F006      259      FAIL:  CJNE   A,B,OK          ; Si A es distinto de B, se encontro un
digito
0132 75F00A      260              MOV    B,#10D      ; Si no es un digito, B == 10D
0135 020000      261      F          JMP    OK2          ; Salimos directamente
0138 F5F0        262      OK:    MOV    B,A          ; Devolvemos el digito en A y B
013A 22          263      OK2:   RET
264
265      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
266      ;Rutina que pasa de numerico a ASCII;
267      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
268      ; Soporta un numero de dos digitos almacenado en A.
269      ; El primer digito se guarda en A, el segundo en B.
013B          270      GETASCII:
013B F8          271              MOV    R0,A          ; Guardamos el numero en R0
013C 75F00A      272              MOV    B,#10D      ; Cargamos 10 en B
013F 84          273              DIV   AB          ; Dividimos el numero por 10
0140 F5F0        274              MOV    B,A          ; Guardamos en B una copia de las
decenas
0142 2430        275              ADD   A,#48D      ; Le sumamos #48 (ASCII '0') al primer
digito
0144 F9          276              MOV    R1,A          ; Guardamos ese primer digito
0145 E5F0        277              MOV    A,B          ; Restauramos las decenas
0147 75F00A      278              MOV    B,#10D      ; Cargamos un 10 en B
014A A4          279              MUL   AB          ; Obtenemos las decenas
014B F5F0        280              MOV    B,A          ; Guardamos las decenas en B
014D E8          281              MOV    A,R0          ; Restauramos el numero original en A
014E 95F0        282              SUBB  A,B          ; Le restamos las decenas al original
0150 2430        283              ADD   A,#48D      ; Le sumamos #48 (ASCII '0') al segundo
digito
0152 F5F0        284              MOV    B,A          ; Dejamos en B el segundo digito
0154 E9          285              MOV    A,R1          ; y en A el primero
0155 22          286              RET
287
288      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
289      ;Rutina que configura el timer0 para que se desborde cada centesima;
290      ;de segundo
291      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0156          292      TIMECONF:
293      ; Configuramos el Timer0:
0156 758901      294              MOV    TMOD,#00000001B ; C/T = 0, Mode = 1
0159 758CD8      295              MOV    TH0,#11011000B ; Configuramos el timer0 para que se
desborde cada
015C 758AF0      296              MOV    TL0,#11110000B ; centesima de segundo
015F D28C        297              SETB  TR0          ; Arrancamos el timer 0
0161 7E00        298              MOV    R6,#0D      ; Segundos = 0
0163 7F00        299              MOV    R7,#0D      ; Centesimas = 0
0165 22          300              RET
301
302      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
303      ;Rutina de inicializacion del timer0;
304      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0166          305      BLANK_TIMER:
0166 758CD8      306              MOV    TH0,#11011000B ; Configuramos el timer0 para que se
desborde cada
0169 758AF0      307              MOV    TL0,#11110000B ; centesima de segundo
016C 22          308              RET
309
310      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
311      ;Rutina que trata las interrupciones del timer;
312      ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
016D          313      DESBORD:
016D C0D0        314              PUSH  PSW          ; metemos dentro de la pila la palabra
de estado
016F C083        315              PUSH  DPH
0171 C082        316              PUSH  DPL
0173 F5F0        317              MOV    B,A
0175 C0F0        318              PUSH  B
0177 C28D        319              CLR   TF0
0179 120000      320      F          CALL  BLANK_TIMER    ; Reiniciamos el timer 0
A51 MACRO ASSEMBLER RELOJ2

```

```

017C 120000 F 321 CALL INCR_C ; Incrementamos centesimas
017F EB 322 MOV A,R3 ; Movemos las pulsaciones a A
0180 75F001 323 MOV B,#1H ; Comprobamos si estamos en modo Lapsus
(una pulsacion)
0183 95F0 324 SUBB A,B
0185 6009 325 JZ LAPSUS ; Si estamos en lapsus, no motramos el
nuevo instante
0187 120000 F 326 CALL PRINT_C ; Escribimos el tiempo actual
transcurrido
018A 900000 F 327 MOV DPTR,#MSG9 ; Iniciamos el cursor
018D 120000 F 328 CALL PUTSTRING
0190 D0F0 329 LAPSUS: POP B
0192 E5F0 330 MOV A,B
0194 D082 331 POP DPL
0196 D083 332 POP DPH
0198 D0D0 333 POP PSW ; sacamos de la pila la palabra de
estado
019A 32 334 RETI
335
336
337 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
338 ;Rutina que trata las interrupciones del pulsador;
339 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
340 ; Incrementa en 1 el registro de numero de pulsaciones (R3)
019B 341 PULSACION:
019B C0D0 342 PUSH PSW ; Metemos dentro de la pila la palabra
de estado
019D C083 343 PUSH DPH
019F C082 344 PUSH DPL
01A1 F5F0 345 MOV B,A
01A3 C0F0 346 PUSH B
01A5 EB 347 MOV A,R3 ; Movemos el numero de pulsaciones a A
01A6 75F001 348 MOV B,#1H ; Movemos 1 al registro B
01A9 25F0 349 ADD A,B ; Sumamos 1 al numero de pulsaciones
(R3)
01AB FB 350 MOV R3,A ; Movemos el resultado a R3
01AC D0F0 351 POP B
01AE E5F0 352 MOV A,B
01B0 D082 353 POP DPL
01B2 D083 354 POP DPH
01B4 D0D0 355 POP PSW ; Sacamos de la pila la palabra de
estado
01B6 32 356 RETI
357
358 ;Rutinas para manejo del reloj
359 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
360 ;Rutina para incremento de centesimas de segundo;
361 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
01B7 362 INCR_C:
01B7 C0F0 363 PUSH B ; Guardamos B
01B9 EF 364 MOV A,R7 ; Guardamos el valor para operar
01BA 2401 365 ADD A,#1D ; Incrementamos en 1 las centesimas
01BC F5F0 366 MOV B,A ; Salvaguardamos el nuevo numero
01BE 249C 367 ADD A,#-100D ; Le restamos 100
01C0 B40006 368 CJNE A,#0D,OK_C ; Si A == 0, reiniciamos cuenta
01C3 75F000 369 MOV B,#0D ; Reiniciamos las centesimas
01C6 120000 F 370 CALL INCR_S ; Incrementamos los segundos
01C9 AFF0 371 OK_C: MOV R7,B ; Restauramos los segundos
01CB D0F0 372 POP B ; Recuperamos B
01CD 22 373 RET
374
375 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
376 ;Rutina para incremento de segundos;
377 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
01CE 378 INCR_S:
01CE C0F0 379 PUSH B ; Guardamos B
01D0 D220 380 SETB FLAG ; Ponemos el flag a 1 para indicar que
han cambiado los seg
undos
01D2 EE 381 MOV A,R6 ; Guardamos el valor para operar
01D3 2401 382 ADD A,#1D ; Incrementamos en 1 los segundos
01D5 F5F0 383 MOV B,A ; Salvaguardamos el nuevo numero
01D7 24C4 384 ADD A,#-60D ; Le restamos 60
01D9 B40006 385 CJNE A,#0D,OK_S ; Si A == 0, reiniciamos cuenta
A51 MACRO ASSEMBLER RELOJ2

```



```

01DC 75F000      386          MOV     B,#0D          ; Reiniciamos las centesimas
01DF 120000      F 387          CALL    INCR_M        ; Incrementamos los segundos
01E2 AEF0        388          OK_S:   MOV     R6,B          ; Restauramos los segundos
01E4 D0F0        389          POP     B              ; Recuperamos B
01E6 22          390          RET
391
392          ;;;;;;;;;;;;;;
393          ;Rutina para incremento de minutos;
394          ;;;;;;;;;;;;;;
01E7          395          INCR_M:
01E7 C0F0        396          PUSH   B              ; Guardamos B
01E9 ED          397          MOV     A,R5          ; Guardamos el valor para operar
01EA 2401        398          ADD     A,#1D         ; Incrementamos en 1 los segundos
01EC F5F0        399          MOV     B,A          ; Salvaguardamos el nuevo numero
01EE 24C4        400          ADD     A,#-60D       ; Le restamos 60
01F0 B40006      401          CJNE   A,#0D,OK_M    ; Si A == 0, reiniciamos cuenta
01F3 75F000      402          MOV     B,#0D        ; Reiniciamos las centesimas
01F6 120000      F 403          CALL    INCR_H        ; Incrementamos los segundos
01F9 ADF0        404          OK_M:   MOV     R5,B          ; Restauramos los segundos
01FB D0F0        405          POP     B              ; Recuperamos B
01FD 22          406          RET
407
408          ;;;;;;;;;;;;;;
409          ;Rutina para incremento de horas;
410          ;;;;;;;;;;;;;;
01FE          411          INCR_H:
01FE C0F0        412          PUSH   B              ; Guardamos B
0200 EC          413          MOV     A,R4          ; Guardamos el valor para operar
0201 2401        414          ADD     A,#1D         ; Incrementamos en 1 los segundos
0203 F5F0        415          MOV     B,A          ; Salvaguardamos el nuevo numero
0205 24E8        416          ADD     A,#-24D       ; Le restamos 60
0207 B40003      417          CJNE   A,#0D,OK_H    ; Si A == 0, reiniciamos cuenta
020A 75F000      418          MOV     B,#0D        ; Reiniciamos las centesimas
020D ACF0        419          OK_H:   MOV     R4,B          ; Restauramos los segundos
020F D0F0        420          POP     B              ; Recuperamos B
0211 22          421          RET
422
423          ;;;;;;;;;;;;;;
424          ;Comienzo del programa;
425          ;;;;;;;;;;;;;;
0212          426          START:
0212 758100      F 427          MOV     SP,#STACK-1   ; Inicializamos el registro SP
428
429          ; Inicializamos el puerto serie
0215 759850      430          MOV     SCON,#01010000B
0218 758920      431          MOV     TMOD,#00100000B ; C/T = 0, Mode = 2
021B 758DF3      432          MOV     TH1,#0F3H
021E D28E        433          SETB   TR1
0220 D299        434          SETB   TI
435
436          ; Inicializaoms el LCD
0222 120000      F 437          CALL    INIT_LCD
0225 120000      F 438          CALL    CLEAR_LCD
439
440          ; Preguntamos el modo de operacion (reloj o cronometro)
0228          441          REPl:
0228 900000      F 442          MOV     DPTR,#MSG1    ; Cargamos el mensaje en DPTR
022B 120000      F 443          CALL    PUTSTRING     ; Escribimos la cadena de texto
022E 900000      F 444          MOV     DPTR,#MSG1
0231 120000      F 445          CALL    PUTLCD        ; Escribimos en el LCD
0234 900000      F 446          MOV     DPTR,#MSG7    ; Escribimos un CRLF
0237 120000      F 447          CALL    PUTSTRING
023A 120000      F 448          CALL    GETCHAR       ; Leemos el modo de operacion en A
023D F5F0        449          MOV     B,A          ; Salvamos A en B
450          ; Comprobamos si entramos en modo crono
023F 24BD        451          ADD     A,#-43H       ; Comprobamos si el caracter es 'C'
A51 MACRO ASSEMBLER RELOJ2

```

```

0241 6016          452          JZ      CRONO          ; Si es 'C' saltamos a modo crono
0243 E5F0          453          MOV     A,B           ; Restauramos A
0245 249D          454          ADD     A,#-63H       ; Comprobamos si el caracter es 'c'
0247 6010          455          JZ      CRONO          ; Si es 'c' saltamos a modo crono
0249 E5F0          456          MOV     A,B           ; Restauramos A
                                457          ; Comprobamos si entramos en modo reloj
024B 24AE          458          ADD     A,#-52H       ; Comprobamos si el caracter es 'R'
024D 6041          459          JZ      RELOJ         ; Si es 'R' saltamos a modo reloj
024F E5F0          460          MOV     A,B           ; Restauramos A
0251 248E          461          ADD     A,#-72H       ; Comprobamos si el caracter es 'r'
0253 603B          462          JZ      RELOJ         ; Si es 'r' saltamos a modo reloj
0255 E5F0          463          MOV     A,B           ; Restauramos A
                                464          ; Si no es ninguna opcion de las anteriores volvemos a preguntar
0257 80CF          465          SJMP   REP1          ; Si no indican una opcion valida,
repetimos
                                466
0259              467          CRONO:
                                468          ; Rutina que simula el funcionamiento de un cronometro
                                469          ; R3 -> Numero de pulsaciones, R4 -> Horas, R5 -> Minutos, R6 ->
Segundos, R7 -> Centesima
s
0259 900000        F      470          MOV     DPTR,#MSG3    ; Cargamos el modo en DPTR
025C 120000        F      471          CALL   PUTSTRING     ; Lo escribimos
025F 900000        F      472          MOV     DPTR,#MSG3    ;
0262 120000        F      473          CALL   PUTLCD        ; Escribimos en el LCD
0265 900000        F      474          MOV     DPTR,#MSG7    ; Escribimos un CRLF
0268 120000        F      475          CALL   PUTSTRING     ;
026B 120000        F      476          CALL   INTCONF       ; Habilitamos interrupciones
026E 120000        F      477          RESET: CALL   TIMECONF  ; Configuramos el timer
0271 7B00          478          MOV     R3,#0H        ; Pulsaciones = 0
0273 7C00          479          MOV     R4,#0H        ; Horas = 0
0275 7D00          480          MOV     R5,#0H        ; Minutos = 0
0277              481          CHECK:
0277 8BF0          482          MOV     B,R3
0279 20F102        483          JB     B.1,STOP
027C 80F9          484          SJMP   CHECK         ; Si no estamos seguimos esperando
                                485
027E              486          STOP:
027E C28C          487          CLR     TR0           ; Paramos el Timer
0280 120000        F      488          CALL   PRINT_C       ; Mostramos el instante actual
0283 900000        F      489          MOV     DPTR,#MSG9    ; Iniciamos el carro
0286 120000        F      490          CALL   PUTSTRING     ;
0289 8BF0          491          WAIT:  MOV     B,R3
028B 20F0E0        492          JB     B.0, RESET
028E 80F9          493          SJMP   WAIT          ; Si no estamos seguimos esperando
0290              494          RELOJ:
                                495          ; Rutina que simula el funcionamiento de un reloj
                                496          ; R4 -> Horas, R5 -> Minutos, R6 -> Segundos, R7 -> Centesimas
0290 900000        F      497          MOV     DPTR,#MSG4    ; Cargamos el modo en DPTR
0293 120000        F      498          CALL   PUTSTRING     ; Lo escribimos
0296 900000        F      499          MOV     DPTR,#MSG4    ;
0299 120000        F      500          CALL   PUTLCD        ; Escribimos en el LCD
029C 900000        F      501          MOV     DPTR,#MSG7    ; Escribimos un CRLF
029F 120000        F      502          CALL   PUTSTRING     ;
                                503          ; Pedimos al usuario que ponga el reloj en hora
                                504          ; HORAS
02A2              505          REP2:
02A2              506          BACK1:
02A2 120000        F      507          CALL   CLEAR_LCD     ; Limpiamos el LCD
02A5 900000        F      508          MOV     DPTR,#MSG5    ; Cargamos el mensaje en DPTR
02A8 120000        F      509          CALL   PUTSTRING     ;
02AB 900000        F      510          MOV     DPTR,#MSG5    ;
02AE 120000        F      511          CALL   PUTLCD        ; Escribimos en el LCD
02B1 900000        F      512          MOV     DPTR,#MSG7    ; Escribimos un CRLF
02B4 120000        F      513          CALL   PUTSTRING     ;
02B7 120000        F      514          CALL   GETCHAR       ; Leemos el primer numero
02BA FB            515          MOV     R3,A          ; Guardamos el caracter
02BB 120000        F      516          CALL   CONVERT       ; Lo convertimos a numero
A51 MACRO ASSEMBLER RELOJ2

```

```

02BE AAF0          517          MOV      R2,B          ; Cargamos en R2 el resultado de la
conversion
02C0 BA0A02       518          CJNE     R2,#10D,D1OK  ; Si no es un digito, volvemos a
pedirlo
02C3 80DD          519          SJMP    BACK1
02C5 75F00A       520          D1OK:   MOV      B,#10D  ; Cargamos el 10 en B
02C8 A4           521          MUL     AB            ; Multiplicamos el primer digito por 10
02C9 F9           522          MOV     R1,A         ; Cargamos el primer digito en R1
02CA EB           523          MOV     A,R3         ; Cargamos el digito para escribirlo
02CB 120000       F 524          CALL   PUTCHAR       ; Escribimos el digito leído
02CE 120000       F 525          CALL   WRITE_TEXT    ; Escribimos en el LCD
526
02D1 120000       F 527          BACK2:  CALL   GETCHAR       ; Leemos el segundo numero
02D4 FB           528          MOV     R3,A         ; Guardamos el caracter
02D5 120000       F 529          CALL   CONVERT        ; Lo convertimos a numero
02D8 FA           530          MOV     R2,A         ; Cargamos en R2 el resultado de la
conversion
02D9 BA0A02       531          CJNE     R2,#10D,D2OK  ; Si no es un digito, volvemos a
pedirlo
02DE             533          D2OK:
02DE EB           534          MOV     A,R3         ; Cargamos el digito para escribirlo
02DF 120000       F 535          CALL   PUTCHAR       ; Escribimos el segundo digito de las
horas
02E2 120000       F 536          CALL   WRITE_TEXT    ; Escribimos en el LCD
02E5 900000       F 537          MOV     DPTR,#MSG7   ; Saltamos de linea
02E8 120000       F 538          CALL   PUTSTRING     ; Restauramos el digito en A
02EB EA           539          MOV     A,R2         ; Sumamos ambas cantidades, hemos
02EC 29           540          ADD     A,R1         ; obtenido las horas
02ED F5F0         541          MOV     B,A          ; Salvaguardamos A en B
02EF 54E0         542          ANL     A,#1110000B  ; Aplicamos mascara para ver que el
numero no sea muy grand
e
02F1 B400AE       543          CJNE     A,#0D,BACK1  ; Si el resultado es 0 OK, sino
releemos
02F4 E5F0         544          MOV     A,B          ; Restauramos A desde B
02F6 5418         545          ANL     A,#00011000B ; Aplicamos mascara para comprobar que
los dos bits no este
n a 1 simultaneamente
02F8 B41802       546          CJNE     A,#24D,SAVE_H ; Si el resultado es 0 OK
02FB 80A5         547          SJMP    BACK1       ; No es correcto, releemos
02FD ACF0         548          SAVE_H: MOV     R4,B   ; Guardamos la hora valida en R4
549          ;Si la hora es correcta, pasamos a leer los minutos
550
551          ; MINUTOS
552          REP3:
02FF             553          CALL   CLEAR_LCD    ; Limpiamos el LCD
0302 900000       F 554          MOV     DPTR,#MSG6   ; Cargamos el mensaje en DPTR
0305 120000       F 555          CALL   PUTSTRING     ; Escribimos en el LCD
0308 900000       F 556          MOV     DPTR,#MSG6   ; Escribimos un CRLF
030B 120000       F 557          CALL   PUTLCD        ; Escribimos el primer numero
030E 900000       F 558          MOV     DPTR,#MSG7   ; Escribimos un CRLF
0311 120000       F 559          CALL   PUTSTRING     ; Leemos el primer numero
0314 120000       F 560          CALL   GETCHAR       ; Guardamos el caracter
0317 F8           561          MOV     R0,A         ; Lo convertimos a numero
0318 120000       F 562          CALL   CONVERT        ; Cargamos en R2 el resultado de la
031B AAF0         563          MOV     R2,B         ; conversion
031D BA0A02       564          CJNE     R2,#10D,D3OK  ; Si no es un digito, volvemos a
pedirlo
0320 80DD          565          SJMP    REP3
0322 75F00A       566          D3OK:   MOV     B,#10D  ; Cargamos el 10 en B
0325 A4           567          MUL     AB            ; Multiplicamos el primer digito por 10
0326 F9           568          MOV     R1,A         ; Cargamos el primer digito en R1
0327 E8           569          MOV     A,R0         ; Cargamos el digito para escribirlo
0328 120000       F 570          CALL   PUTCHAR       ; Escribimos el digito leído
032B 120000       F 571          CALL   WRITE_TEXT    ; Escribimos en el LCD
572
032E 120000       F 573          CALL   GETCHAR       ; Leemos el segundo numero
0331 F8           574          MOV     R0,A         ; Guardamos el caracter
0332 120000       F 575          CALL   CONVERT        ; Lo convertimos a numero
0335 FA           576          MOV     R2,A         ; Cargamos en R2 el resultado de la
conversion
0336 BA0A02       577          CJNE     R2,#10D,D4OK  ; Si no es un digito, volvemos a
0339 80C4         578          SJMP    REP3

```

```

pedirlo
033B          579      D4OK:
033B E8       580      MOV      A,R0          ; Cargamos el digito para escribirlo
A51 MACRO ASSEMBLER RELOJ2                                16/12/03
18:53:43 PAGE 10

033C 120000   F      581      CALL    PUTCHAR       ; Escribimos el segundo digito de los
minutos
033F 120000   F      582      CALL    WRITE_TEXT    ; Escribimos en el LCD
0342 900000   F      583      MOV     DPTR,#MSG7    ; Saltamos de linea
0345 120000   F      584      CALL    PUTSTRING     ;
0348 EA       585      MOV     A,R2          ; Restauramos el digito en A
0349 29       586      ADD     A,R1          ; Sumamos ambas cantidades, hemos
obtenido los minutos
034A F5F0     587      MOV     B,A           ; Salvaguardamos A en B
034C 54C0     588      ANL    A,#11000000B  ; Aplicamos mascara para ver que el
numero no sea muy grand
e
034E B400AE   589      CJNE   A,#0D,REP3    ; Si el resultado es 0 OK, sino
releemos
0351 E5F0     590      MOV     A,B           ; Restauramos A desde B
0353 543C     591      ANL    A,#00111100B  ; Aplicamos mascara para comprobar que
los cuatro bits no e
sten a 1 simultaneamente
0355 B43C02   592      CJNE   A,#60D,SAVE_M ; Si el resultado es 0 OK
0358 80A5     593      SJMP   REP3          ; Si los minutos no son correctos,
reintentamos
035A ADF0     594      SAVE_M: MOV    R5,B    ; Guardamos los minutos validos en R5
035C 120000   F      595      CALL    TMECONF      ; Configuramos el timer
035F D220     596      SETB   FLAG          ; Ponemos el FLAG a 1
597      ; Nos metemos en un bucle, comprobamos desbordamiento del timer
0361          598      REP4:
0361 308DFD   599      JNB    TF0,REP4      ; Comprobamos desbordamiento en Timer0
0364 C28D     600      CLR    TF0
0366 120000   F      601      CALL    BLANK_TIMER   ; Reiniciamos el timer 0
0369 120000   F      602      CALL    INCR_C        ; Incrementamos centesimas
036C 102002   603      JBC    FLAG,P_LCD    ;
036F 80F0     604      SJMP   REP4          ; Sino ha cambiado el segundo no lo
mostramos
0371 120000   F      605      P_LCD: CALL   CLEAR_LCD ; Limpiamos el LCD
0374 900000   F      606      MOV     DPTR,#MSG8    ; Escribimos la cadena de texto inicial
0377 120000   F      607      CALL    PUTSTRING     ;
037A 120000   F      608      CALL    PRINT_R       ; Imprimimos el reloj
037D 900000   F      609      MOV     DPTR,#MSG9    ; Iniciamos el carro
0380 120000   F      610      CALL    PUTSTRING     ;
0383 80DC     611      SJMP   REP4
612
613
614
0385 020000   F      615      LJMP   FIN
616
0388 52454C4F 617      MSG1:  DB      'RELOJ/CRONOMETRO (R/C):',00H
038C 4A2F4352
0390 4F4E4F4D
0394 4554524F
0398 2028522F
039C 43293A00
03A0 4D4F444F 618      MSG2:  DB      'MODO: ',00H
03A4 3A2000
03A7 4D4F444F 619      MSG3:  DB      'MODO CRONOMETRO',00H
03AB 2043524F
03AF 4E4F4D45
03B3 54524F00
03B7 4D4F444F 620      MSG4:  DB      'MODO RELOJ',00H
03BB 2052454C
03BF 4F4A00
03C2 484F5241 621      MSG5:  DB      'HORAS (HH):',00H
03C6 53202848
03CA 48293A00
03CE 4D494E55 622      MSG6:  DB      'MINUTOS (MM):',00H
03D2 544F5320
03D6 284D4D29
03DA 3A00
03DC 0D0A00    623      MSG7:  DB      CR,LF,00H
03DF 484F5241 624      MSG8:  DB      'HORA: ',00H
03E3 3A2000
03E6 0D00     625      MSG9:  DB      CR,00H
626

```

03E8 627 FIN:
 A51 MACRO ASSEMBLER RELOJ2
 18:53:43 PAGE 11

628
 A51 MACRO ASSEMBLER RELOJ2
 18:53:43 PAGE 12

16/12/03

END

16/12/03

SYMBOL TABLE LISTING

N A M E	T Y P E	V A L U E	A T T R I B U T E S	
ACC.	D ADDR	00E0H	A	
B.	D ADDR	00F0H	A	
BACK1.	C ADDR	02A2H	R	SEG=PROG
BACK2.	C ADDR	02D1H	R	SEG=PROG
BLANK_TIMER.	C ADDR	0166H	R	SEG=PROG
CHECK.	C ADDR	0277H	R	SEG=PROG
CINCO.	C ADDR	0116H	R	SEG=PROG
CLEAR_LCD.	C ADDR	0038H	R	SEG=PROG
CONVERT.	C ADDR	00FBH	R	SEG=PROG
CR	N NUMB	000DH	A	
CRONO.	C ADDR	0259H	R	SEG=PROG
CUATRO	C ADDR	0111H	R	SEG=PROG
D1OK	C ADDR	02C5H	R	SEG=PROG
D2OK	C ADDR	02DEH	R	SEG=PROG
D3OK	C ADDR	0322H	R	SEG=PROG
D4OK	C ADDR	033BH	R	SEG=PROG
DATO	D ADDR	00F8H	A	
DB0.	B ADDR	00F8H.0	A	
DB1.	B ADDR	00F8H.1	A	
DB2.	B ADDR	00F8H.2	A	
DB3.	B ADDR	00F8H.3	A	
DB4.	B ADDR	00F8H.4	A	
DB5.	B ADDR	00F8H.5	A	
DB6.	B ADDR	00F8H.6	A	
DB7.	B ADDR	00F8H.7	A	
DESBORD.	C ADDR	016DH	R	SEG=PROG
DOS.	C ADDR	0107H	R	SEG=PROG
DPH.	D ADDR	0083H	A	
DPL.	D ADDR	0082H	A	
EA	B ADDR	00A8H.7	A	
EN	B ADDR	00E8H.7	A	
ET0.	B ADDR	00A8H.1	A	
EX0.	B ADDR	00A8H.0	A	
EXIT	C ADDR	006FH	R	SEG=PROG
EXIT2.	C ADDR	007AH	R	SEG=PROG
FAIL	C ADDR	012FH	R	SEG=PROG
FIN.	C ADDR	03E8H	R	SEG=PROG
FLAG	N NUMB	0020H	A	
GETASCII	C ADDR	013BH	R	SEG=PROG
GETCHAR.	C ADDR	00EBH	R	SEG=PROG
INCR_C	C ADDR	01B7H	R	SEG=PROG
INCR_H	C ADDR	01FEH	R	SEG=PROG
INCR_M	C ADDR	01E7H	R	SEG=PROG
INCR_S	C ADDR	01CEH	R	SEG=PROG
INIT_LCD	C ADDR	0013H	R	SEG=PROG
INTCONF.	C ADDR	00E2H	R	SEG=PROG
IT0.	B ADDR	0088H.0	A	
LAPSUS	C ADDR	0190H	R	SEG=PROG
LF	N NUMB	000AH	A	
MSG1	C ADDR	0388H	R	SEG=PROG
MSG2	C ADDR	03A0H	R	SEG=PROG
MSG3	C ADDR	03A7H	R	SEG=PROG
MSG4	C ADDR	03B7H	R	SEG=PROG
MSG5	C ADDR	03C2H	R	SEG=PROG
MSG6	C ADDR	03CEH	R	SEG=PROG
MSG7	C ADDR	03DCH	R	SEG=PROG
MSG8	C ADDR	03DFH	R	SEG=PROG
MSG9	C ADDR	03E6H	R	SEG=PROG
NUEVE.	C ADDR	012AH	R	SEG=PROG
OCHO	C ADDR	0125H	R	SEG=PROG
A51 MACRO ASSEMBLER RELOJ2				
18:53:43 PAGE 13				
OK	C ADDR	0138H	R	SEG=PROG
OK2.	C ADDR	013AH	R	SEG=PROG

16/12/03

OK_C	C ADDR	01C9H	R	SEG=PROG
OK_H	C ADDR	020DH	R	SEG=PROG
OK_M	C ADDR	01F9H	R	SEG=PROG
OK_S	C ADDR	01E2H	R	SEG=PROG
P4	D ADDR	00E8H	A	
P5	D ADDR	00F8H	A	
P6	D ADDR	00FAH	A	
P7	D ADDR	00DBH	A	
P8	D ADDR	00DDH	A	
PRINT_C.	C ADDR	00C1H	R	SEG=PROG
PRINT_R.	C ADDR	0084H	R	SEG=PROG
PROG	C SEG	03E8H		REL=UNIT
PSW.	D ADDR	00D0H	A	
PULSACION.	C ADDR	019BH	R	SEG=PROG
PUTCHAR.	C ADDR	005DH	R	SEG=PROG
PUTLCD	C ADDR	0070H	R	SEG=PROG
PUTSEPARATOR	C ADDR	007BH	R	SEG=PROG
PUTSTRING.	C ADDR	0065H	R	SEG=PROG
P_LCD.	C ADDR	0371H	R	SEG=PROG
RELOJ.	C ADDR	0290H	R	SEG=PROG
REP1	C ADDR	0228H	R	SEG=PROG
REP2	C ADDR	02A2H	R	SEG=PROG
REP3	C ADDR	02FFH	R	SEG=PROG
REP4	C ADDR	0361H	R	SEG=PROG
RESET.	C ADDR	026EH	R	SEG=PROG
RI	B ADDR	0098H.0	A	
RS	B ADDR	00E8H.6	A	
RW	B ADDR	00E8H.5	A	
SAVE_H	C ADDR	02FDH	R	SEG=PROG
SAVE_M	C ADDR	035AH	R	SEG=PROG
SBUF	D ADDR	0099H	A	
SCON	D ADDR	0098H	A	
SEIS	C ADDR	011BH	R	SEG=PROG
SIETE.	C ADDR	0120H	R	SEG=PROG
SP	D ADDR	0081H	A	
STACK.	I SEG	0010H		REL=UNIT
START.	C ADDR	0212H	R	SEG=PROG
STOP	C ADDR	027EH	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	0156H	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	010CH	R	SEG=PROG
UNO.	C ADDR	0102H	R	SEG=PROG
WAIT	C ADDR	0289H	R	SEG=PROG
WAIT_LCD	C ADDR	0000H	R	SEG=PROG
WRITE_TEXT	C ADDR	0051H	R	SEG=PROG

REGISTER BANK(S) USED: 0

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

Rutinas utilizadas:

A continuación se explican detalladamente las rutinas relacionadas con el manejo del LCD en este programa:

- `WAIT_LCD`: rutina que espera a que el display esté listo para ejecutar una nueva instrucción, la rutina entra en un bucle mientras el flag *busy* `ACC.7` se mantenga a 1, cuando el flag tenga el valor 0, la rutina finaliza, indicando así que el display está listo para ejecutar una nueva instrucción.
- `INIT_LCD`: rutina que inicializa el display, necesaria para su funcionamiento, coloca el cursor al comienzo del mismo y lo configura para que el cursor se reposicione con cada nueva entrada.
- `CLEAR_LCD`: sitúa el cursor en la posición inicial del display, borrando el contenido anterior, y prepara el cursor para que se incremente.
- `WRITE_TEXT`: escribe el carácter almacenado en `A` en la posición actual del cursor del display.
- `PUTLCD`: Escribe una cadena de caracteres por el display. La cadena de caracteres debe estar apuntada por `DPTR`. La rutina lee carácter a carácter desde `DPTR` y llama a la rutina `WRITE_TEXT`, hasta que llegue un carácter nulo (`00H`).

Problemas encontrados:

Durante el desarrollo de esta sencilla práctica hemos tenido fundamentalmente un inconveniente, el mal funcionamiento del LCD disponible. Es por ello que debimos recurrir a otros LCD prestados por compañeros para probar la práctica. En el lado negativo, destacar la cantidad de tiempo que nos hizo perder este problema buscando una posible solución en nuestro propio programa.

Anexo: rutinas generales.

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 número 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 envía `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 número 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 dígito 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 número 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 número 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.