12.7. - EL DISCO DURO DEL AT (IDE, MFM, BUS LOCAL).

La información aquí vertida se aplica al tradicional controlador de disco duro ST506, que ha equipado a los discos duros MFM/RLL de los AT, con el que es compatible en líneas generales tanto el interface de los ESDI como el de los IDE (ISA, PCI o Bus Local). Sin embargo, los discos SCSI no son compatibles con la información que aquí se expone, ni tampoco la controladora de los PC/XT.

12.7.1 - EL INTERFACE.

El disco duro se conecta a la controladora a través de dos cables: uno con las señales de control y otro con las de datos. El de señales de control consta de 34 conectores, y el de datos de 20.

Nombre señal	Pin señal	Pin masa
- HEAD SELECT 3	2	1
- HEAD SELECT 2	4	3
- WRITE GATE	6	5
- SEEK COMPLETE	8	7
- TRACK 000	10	9
- WRITE FAULT	12	11
- HEAD SELECT 0	14	13
RESERVADO	16	15
- HEAD SELECT 1	18	17
- INDEX	20	19
- READY	22	21
- STEP	24	23
- DRIVE SELECT 1	26	25
- DRIVE SELECT 2	28	27
- DRIVE SELECT 3	30	29
- DRIVE SELECT 4	32	31
- DIRECTION IN	34	33

SEÑALES DE CONTROL

Nombre señal	Pin señal
-Unidad seleccionada	1
+MFM Escribir datos	13
-MFM Escribir datos	14
+MFM Leer datos	17
-MFM Leer datos	18
Masa	2, 4, 6, 8, 11, 12, 15, 16, 19

SEÑALES PARA TRANSFERENCIA DE DATOS

Significado de las señales de control (entrada):

-Write Gate: Un nivel activo permite a los datos ser escritos en disco. Inactivo implica una lectura y permite mover los cabezales con el pulso de Step.

-Head Select: Estas señales codifican un número de 4 bits para referenciar a un cabezal.

-Direction In: Esta señal define la dirección en que se mueven los cabezales con la señal Step. Un valor inactivo indica dirección Out y los pulsos Step alejan los cabezales del centro del disco; un valor activo se interpreta como In y dichos pulsos acercan los cabezales al centro.

-Step: Esta señal provoca un movimiento de los cabezales en la dirección que indica la anterior.

-Drive Select: Líneas de selección de unidad. Las unidades poseen jumpers para dejarse seleccionar con ciertas combinaciones.

Significado de las señales de control (salida):

-Seek Complete: Informan del final del posicionamiento de los cabezales. Si esta señal no indica que está terminada dicha operación, no se puede leer ni escribir.

-Track 000: Indica que los cabezales están sobre la pista 0.

-Write Fault: Señala una condición que está provocando una operación no correcta del disco.

-Index: La unidad indica aquí al exterior el comienzo de una pista (a cada revolución).

-Ready: Señal necesaria junto con Seek Complete para poder leer/escribir/posicionar cabezales.

12.7.2 - PROGRAMACIÓN DE LA CONTROLADORA.

El disco duro trabaja con sectores de 512 bytes; soporta corrección de errores ECC, operaciones multisector rebasando fronteras de pista y cilindro y funciones de autodiagnóstico. El límite de capacidad está en 1024 cilindros y 16 cabezales. Los registros de operación son los mostrados en la figura, estando la controladora ubicada normalmente en la localización E/S primaria.

Dirección E/S hex.		Significado	
Primaria	Secund.	Lectura	Escritura
1F0	170	Data registers	Data register
1F1	171	Error register	Write precomp
1F2	172	Sector count	Sector count
1F3	173	Sector number	Sector number
1F4	174	Cylinder low	Cylinder low
1F5	175	Cylinder high	Cylinder high
1F6	176	Drive/Head	Drive/Head
1F7	177	Status register	Command register

Significado de los registros:

Data Register: Permite acceder al buffer donde está almacenado el sector para leer y escribir en el modo PIO (esto es, sin DMA). No debería ser accedido a menos que haya una operación de lectura o escritura en curso. Implementa una dirección de 16 bits dentro del buffer de la controladora que contiene al sector para las operaciones de lectura y escritura normales. Para una lectura/escritura *largas* 4 bytes ECC son transferidos por byte con al menos 2 microsegundos entre transferencias (la línea DRQ debe estar activa antes de transferir los bytes ECC).

Error Register: De sólo lectura, contiene información sobre el comando previo. El dato es válido sólo cuando el bit de error en el registro de estado está activo.

- Tras conectar el disco duro a la corriente o tras enviar el comando apropiado, se encuentra en modo diagnóstico: en esos casos, el registro debe ser comprobado diga lo que diga el bit del registro de estado (con el significado en estos casos de 01-No hay error, 02-Fallo del controlador, 03-Error en el buffer del sector, 04-Error en el dispositivo ECC, 05-Error en el procesador de control).
- Cuando no está en modo diagnóstico, caso más común, significado de sus bits:

bit 0: Data Address Mark (DAM) no encontrada en los 16 bytes del campo ID.
bit 1: Error TR 000. Se activa si tras un comando *Restore*, la señal Track 000 no se activa después de 1023 pulsos de retroceso.
bit 2: Comando abortado. En estos casos, se debe mirar los registros de Status y Error para determinar con precisión la causa (que estar en Write Fault, Seek Complete, Drive ready -- o comando inválido en otro caso).
bit 3: No usado.
bit 4: ID no encontrada. La marca ID que identifica al cilindro, cabezal y sector no ha sido encontrada. Si están activos los reintentos, el controlador lo reintenta 16 veces antes de dar error, en caso contrario sólo explora la pista como mucho 2 veces antes de dar el error.
bit 5: No usado.
bit 6: Error ECC. Indica si se ha producido un error ECC incorregible durante una lectura.
bit 7: Bad Block detected. Indica que se ha encontrado un sector marcado como defectuoso en la ID; no se intentarán en él ni lecturas ni escrituras.

Write Precompensation: El valor almacenado es el cilindro de comienzo para la escritura precompensada dividido por 4.

Sector Count: Indica el número de sectores a transferir durante la lectura, escritura, verificación o formateo. En las operaciones multisector, este registro se decrementa y el *Sector Number* se incrementa; al formatear, antes de enviar cada comando de formateo debe cargarse aquí el número de sectores en la pista. Se soportan operaciones multisector que crucen fronteras de pista y cilindro. Las características de la unidad deben establecerse con el comando *Set Parameters* antes de una transferencia multisector. Este registro debe cargarse con el número de sectores antes de cualquier comando relacionado con datos. Un valor 0 representa 256 sectores.

Sector Number: Número de sector para la lectura, escritura y verificación. El sector inicial se carga aquí en las operaciones multisector.

Cylinder Number: Número de cilindro para los comandos de lectura, escritura, verificación y posicionamiento de cabezales. Entre el registro que almacena la parte baja y el de la parte alta (low y high respectivamente) se guarda un número entre 0 y 1023.

Drive/Head: Bits 7 y 5 puestos a 1, el 6 puesto a 0. El bit 4 indica la unidad seleccionada (0 el primer disco duro y 1 el segundo) y los bits 0-3 el número de cabezal de lectura/escritura deseado. Para acceder a las cabezas 8-15, es necesario además activar el bit 3 del puerto 3F6h. Importante: este registro debe cargarse con el número máximo de cabezales antes de enviar un comando *Set Parameters*.

`Status register:` Se actualiza tras ejecutar los comandos. El programa debe mirar este registro para conocer el resultado. Si el bit *busy* (7) está activo, los demás bits no son válidos. Una lectura de este registro borra la petición de interrupción IRQ 14. Si *write-fault* (bit 5) o *error* (bit 0) están activos, o si *seek-complete* (bit 4) o drive-*ready* (bit 6) están inactivos, la operación multisector es abortada. Significado de los bits:

- bit 7: Busy. Un 1 indica que el controlador está ejecutando un comando; por tanto, este bit debe ser examinado antes de leer cualquier registro.
- bit 6: Drive-ready. Un 0 indica que la lectura, escritura y seek están inhibidas; para poder ejecutarlas debe estar a 1 junto con el bit seek-complete (4).
- bit 5: Write-fault. Un 1 indica funcionamiento incorrecto de la unidad; la lectura, escritura o seek están inhibidos.
- bit 4: Seek-complete. Un 1 indica que los cabezales han terminado el seek.
- bit 3: Data-request. Este bit indica que el buffer del sector necesita ser atendido en un comando de lectura o escritura: si este bit o el busy (7) están activos, hay un comando en ejecución. Hasta recibir algún comando, este bit está a 0.
- bit 2: Corrected-data. Un 1 indica que los datos leídos del disco fueron corregidos de error ECC con éxito. Errores *suaves* no abortan la operación multisector.
- bit 1: Index. Este bit se pone a 1 tras cada revolución del disco.
- bit 0: Error. Un 1 indica que el comando previo terminó en error, y uno o más bits del *Error register* están activos. El próximo comando enviado al controlador borra este bit. Si este bit se activa, la operación multisector es abortada.

`Command Register:` Acepta 8 diferentes comandos. Los comandos se programan cargando primero los demás registros necesarios y escribiendo después el comando en éste mientras el registro de estado devuelve una condición de no *busy*. Un comando no legal provoca un error de comando abortado. La solicitud de interrupción IRQ 14 se borra al escribir un comando. Los comandos soportados son:

Comando	bit	7	6	5	4	3	2	1	0
Restore		0	0	0	1	R3	R2	R1	R0
Seek		0	1	1	1	R3	R2	R1	R0
Read sector		0	0	1	0	0	0	L	T
Write sector		0	0	1	1	0	0	L	T
Format track		0	1	0	1	0	0	0	0
Read verify		0	1	0	0	0	0	0	T
Diagnose		1	0	0	1	0	0	0	0
Set Parameters		1	0	0	1	0	0	0	1

R3	R2	R1	R0	Stepping rate
0	0	0	0	35 µs
0	0	0	1	0.5 ms
0	0	1	0	1.0 ms
0	0	1	1	1.5 ms
0	1	0	0	2.0 ms
0	1	0	1	2.5 ms
0	1	1	0	3.0 ms
0	1	1	1	3.5 ms
1	0	0	0	4.0 ms
1	0	0	1	4.5 ms
1	0	1	0	5.0 ms
1	0	1	1	5.5 ms
1	1	0	0	6.0 ms
1	1	0	1	6.5 ms
1	1	1	0	7.0 ms
1	1	1	1	7.5 ms

Bit		0	1
L	Modo de datos	Sólo datos	Datos y 4 bytes ECC
T	Modo de reintentos	Reintentos habilitados	Reintentos inhibidos

Nota: Después de un reset o un comando Diagnose, el step rate queda en 7.5 ms. Por otro lado, el sistema verifica la operación ECC leyendo y escribiendo estos bytes: cuando los reintentos están deshabilitados, los reintentos de ECC e ID están limitados a menos de dos vueltas completas del disco.

Explicación de los comandos.

- Restore: Envía los cabezales a la pista 0 (hasta que la señal Track 000 es activa). Si Track 000 no se activa tras 1023 pulsos de *step* activa el bit de error en el registro de estado y deja el error TR 000 en el registro *error*. El step rate es establecido por el propio comando.
- Seek: Mueve los cabezales al cilindro indicado. Está soportado un seek simultáneo en dos unidades. Al final del comando se produce una interrupción.
- Read sector: Cierto número de sectores (1-256) pueden ser leídos del disco duro con o sin el campo ECC añadido, en el modo PIO (entrada-salida programada, sin DMA). Si los cabezales no están sobre la pista necesaria, el controlador envía pulsos *step* para posicionarlo, utilizando el step rate del último *seek* o *restore*. Los errores de datos de hasta 5 bits son corregidos automáticamente en los comandos de lectura *corta*. Si un error no corregible tiene lugar, se continúa leyendo el sector donde apareció pero ya no se leen más sectores en el caso de los accesos multisector. Se produce una interrupción por cada sector cuando está preparado para ser transferido, pero no al final del comando.
- Write sector: Cierto número de sectores (1-256) pueden ser escritos a disco duro con o sin el campo ECC añadido, en el modo PIO (entrada-salida programada, sin DMA). Realiza los seeks que sea necesario hacer. Las interrupciones suceden cada vez que es transferido un sector al buffer (salvo el primero) y al final del comando. El primer sector debería ser escrito en el buffer inmediatamente después de que el comando ha sido enviado y "Data-request" es activo.
- Format track: Se formatea la pista indicada según la tabla de interleave que se transfiere. Hay 2 bytes por cada sector: 0, Nº sector. Así se puede elegir la numeración deseada. Hay que enviar 512 bytes con independencia de que sean menos en la tabla (por ej. 34 bytes para 17 sectores). El *sector count* debe cargarse con el nº de sectores por pista antes de cada comando de estos. Se genera una interrupción al final del comando de formateo. Los sectores defectuosos se marcan sustituyendo el 0 que les precede por 80. Cuando se conmuta entre dos unidades, antes de formatear hay que hacer un *restore*.
- Read Verify: Similar al comando *read sector* con la diferencia de que no se envían datos al ordenador; de esta manera simplemente se verifica la integridad de los mismos. Una única interrupción se genera al completarse el comando o en caso de error.
- Diagnose: El adaptador ejecuta su auto-test y devuelve el resultado en el *error register*. Se produce una interrupción cuando completa el comando.
- Set Parameters: Establece los parámetros de la unidad: máximo número de cabezales y sectores/pista. El registro *drive/head* indica qué unidad es afectada. Hay que actualizar los registros *sector count* y *drive/head* antes de enviar este comando. Estos parámetros serán empleados para cruzar los cilindros en las operaciones multisector. Se genera una interrupción cuando se completa el comando. Este comando debe ser enviado antes de intentar alguna operación multisector. Se soportan dos discos duros, con diferentes características cada uno, definidas por este comando.

Registro del controlador de disco duro (3F6h) y Registro de entrada digital (3F7h).

Además de informar de la línea de cambio de disco en los disquetes, los bits 0-5 del registro de entrada digital (3F7h) están relacionados con el disco duro.

```
3F6h  -  bits 7-4:Reservados
         bit 3:   0 - Reduce Write Current)
                  1 - Head 3 select enable)
         bit 2:   1 - Disk reset enable
                  0 - Disk reset disable
         bit 1:   0 - Disk initialization enable
                  1 - Disk initialization disable
         bit 0:   Reservado

3F7h  -  bit 7:   Línea de cambio de disquetes.
         bit 6:   Write gate
         bit 5:   Head select 3 / Reduced Write Current
         bit 4:   Head select 2
         bit 3:   Head select 1
         bit 2:   Head select 0
         bit 1:   Drive select 1
         bit 0:   Drive select 0
```

12.7.3 - EJEMPLO PRACTICO DE PROGRAMACIÓN.

En los AT la interrupción de disco duro es la IRQ 14 (INT 76h). La BIOS, en caso de producirse esta interrupción, almacena un valor 0FFh en 40h:8Eh con el gestor que tiene por defecto. Las transferencias

con el disco duro tienen lugar sin DMA por regla general. Esto se comprende mejor teniendo en cuenta que la controladora tiene un buffer interno con capacidad para algún sector y, por tanto, cuando hay que transferirlo, no hay que esperar a que venga del disco mientras este gira *lentamente* (como en el caso de los disquetes): una transferencia con el DMA ordinario aquí sería más lenta que a través de la CPU.

Parte de la documentación vista con anterioridad es sólo *oficial*. Por ejemplo, los discos IDE suelen venir formateados de fábrica a bajo nivel e ignoran el comando de formateo: estas unidades son bastante inteligentes y llevan su propia gestión de sectores defectuosos (reemplazándolos por otros que tienen libres para simular que todo está correcto) así como de interleaves (generalmente 1:1, valores peores se deben a controladoras obsoletas que no tenían un buffer con capacidad para una pista) y skews óptimos.

El programa de ejemplo es el embrión de una utilidad para acceder directamente a la controladora de disco duro. La función **operahd()** solo implementa realmente el comando de leer sector. En el ejemplo, se lee el primer sector absoluto del disco, donde suele ser fácil reconocer la tabla de particiones. En nuestro caso, en lectura, justo antes de enviar el comando se borra el flag de interrupción. Una vez enviado, cuando llegue la interrupción se procede a leer el sector. Para ello se utiliza la rápida instrucción REP INSW del 286 y procesadores superiores, para E/S repetitiva, que lo trae a gran velocidad desde la memoria de la controladora a la del sistema.

Un acceso directo a bajo nivel puede tener mucho interés para ciertas aplicaciones. Por ejemplo, un antivirus puede asegurarse de que ha reparado la tabla de particiones (o cualquier otra zona del disco) sin temor a que en su llamada a INT 13h el virus residente le haya estropeado el trabajo (aunque si el virus trabaja en modo protegido y controla el acceso a los puertos E/S del disco duro...).

HDIRECT.C

```
/*********************************************************************
*                                                                    *
*     ACCESO A DISCO DURO ESTANDAR AT (IDE, MFM, BUS LOCAL, ETC)     *
*              PROGRAMANDO DIRECTAMENTE LA CONTROLADORA              *
*                                                                    *
*     - Compilar en modelo Large.                                    *
*     - Este programa sólo implementa la función de leer sector.     *
*     - No soportadas controladoras de XT, SCSI u otras.             *
*                                                                    *
*********************************************************************/


#include <dos.h>
#include <alloc.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>


#define HD_RESTORE    0x10     /* comandos del controlador */
#define HD_SEEK       0x70
#define HD_READ       0x20
#define HD_WRITE      0x30
#define HD_FORMAT     0x50
#define HD_READVERIFY 0x40
#define HD_DIAGNOSE   0x90
#define HD_SETPARAM   0x91

#define HDR_MAIN      0x3F6
#define HDR_DATA      0x1F0    /* registros del controlador */
#define HDR_ERROR     0x1F1
#define HDR_WRITEP    0x1F1
#define HDR_SECNT     0x1F2
#define HDR_SEC       0x1F3
#define HDR_LCYL      0x1F4
#define HDR_HCYL      0x1F5
#define HDR_DRVHD     0x1F6
#define HDR_STATUS    0x1F7
#define HDR_CMD       0x1F7

#define HD_ECC         2
#define HD_NORETRY     1

#define HD_BUSY       0x80
#define HD_DATA_REQ    8


int operahd (int unidad, int cabeza, int cilindro, int sector,
             int operacion, char huge *direccion, int numsect)
{
  int i;

  if (operacion==HD_SETPARAM) {
    }
  else if (operacion==HD_DIAGNOSE) {
    }
  else if (operacion==HD_FORMAT) {
    }
  else if ((operacion & 0xFE) == HD_READVERIFY) {
    }
  else if ((operacion & 0xFC) == HD_READ) {
    outportb (HDR_SECNT, numsect);        /* nº sectores */
    outportb (HDR_SEC, sector);           /* primer sector */
    outportb (HDR_LCYL, cilindro & 0xFF); /* nº cilindro 0..7 */
    outportb (HDR_HCYL, cilindro >> 8);   /* nº cilindro 8..9 */
    outportb (HDR_DRVHD, unidad << 4 | cabeza | 0xC0);
    outportb (HDR_MAIN, cabeza & 8);

    pokeb (0x40, 0x8e, 0);         /* flag de interrupción a 0 */

    outportb (HDR_CMD, operacion); /* comando */

    while (!peekb(0x40, 0x8e));    /* esperar interrupción 76h */
                                   /* (convendría poner un timeout) */

    /* por eficiencia, el siguiente código está en ensamblador */

    asm {
      push es              /* máxima lectura soportada: casi 64 Kb */
      push cx
      push dx
      push di
      mov  cx,numsect
      xchg ch,cl           /* CX = numsect * 256 = nº palabras */
      les  di,direccion
      cld
      mov  dx,HDR_DATA
      db   0F3h, 6Dh       /* instrucción 286+ 'rep insw' */
      pop  di
      pop  dx
      pop  cx
      pop  es
      }
    }
  else if ((operacion & 0xFC) == HD_WRITE) {
    }
  else if ((operacion & 0xF0) == HD_SEEK) {
    }
  else if ((operacion & 0xF0) == HD_RESTORE) {
    }
}


void main()
{
  /* el puntero huge comienza en XXXX:0004 */

  unsigned char huge *buffer, huge *p;
  unsigned i, j, k;

  if ((buffer=farmalloc(0xFFFC))==NULL) {
    printf("\nMemoria insuficiente.\n");
    exit(1);
    }

  /* leer sector de tabla de partición */

  operahd (0, 0, 0, 1, HD_READ | HD_NORETRY, buffer, 1);

  /* imprimir sector de 512 bytes */

  p=buffer;
  for (i=0; i<2; i++) {
    clrscr();
    for (j=0; j<256; j+=16) {
      for (k=0; k<16; k++) printf("%02X ", *p++); p-=16;
      printf("   ");
      for (k=0; k<16; k++) {
        if (*p<' ') printf("."); else printf("%c", *p);
        p++;
        }
      printf("\n");
      }
    printf("\n- Estás viendo 256 bytes del sector.\n");
    printf("- Pulsa una tecla para continuar.");
    getch();
  }
}
```

12.8. - EL CONTROLADOR DEL TECLADO: 8042.

En este apartado se estudiará a fondo el funcionamiento a bajo nivel del teclado en los ordenadores compatibles, si bien es poco frecuente que sea necesario acceder al mismo de esta manera.

12.8.1 - EL 8042.

Un microordenador llamado teclado.

El teclado se conecta al ordenador por medio de un cable que contiene 4 hilos hábiles: dos que conducen la corriente, uno para datos y otro para reloj. El teclado es en realidad un pequeño microordenador; de hecho muchos teclados llevan en su interior el chip 8049 de Intel (el microprocesador esclavo del viejo QL de Sinclair) que consta de unos 2 Kb de memoria ROM y 128 bytes de RAM (las 8 primeras posiciones son empleadas como registros). Este procesador se encarga de detectar la pulsación de las teclas, generando unos bytes que las identifican y enviándolos a continuación por el cable a través de un protocolo de comunicación en serie que en el AT consta de 11 bits por cada dato (1 de inicio, 8 de datos, 1 de paridad y otro de *stop*) y 9 en los XT (entre otras razones, porque no se controla la paridad). Los teclados de AT y de XT generan códigos diferentes para las mismas teclas. Además, al soltar una tecla, los teclados de XT generan el mismo código que al pulsarla pero con el bit 7 activo; sin embargo, en AT se generan dos códigos que se envían consecutivamente (0F0h y después el mismo código que al pulsarla). El teclado se encarga de repetir los códigos de una tecla cuando ésta lleva cierto tiempo pulsada, en el conocido mecanismo *autorepeat* de la mayoría de los teclados. Muchos teclados tienen debajo un interruptor que permite seleccionar su modo de funcionamiento (XT o AT).

El teclado en los PC y XT.

Los datos, cuando llegan al ordenador, reciben un tratamiento diferente en función de si el ordenador es un XT o un AT, mucho más sencillo en el primero. En los XT se van colocando los bits que llegan en un simple registro de desplazamiento conectado al puerto 60h; al completarse los 8 se produce una interrupción de tipo IRQ 1 (INT 9), la segunda de mayor prioridad después de la del temporizador. No obstante, el teclado es capaz de memorizar hasta 8 pulsaciones cuando la CPU no tiene tiempo para atenderle. Después de leer el código de la tecla, el programa que la gestione habrá de enviar una señal de reconocimiento a la circuitería del ordenador para permitir que continúe la recepción de datos.

El controlador del teclado del AT: el 8042.

En los AT hay un circuito integrado encargado de interpretar los datos procedentes del teclado y, después de traducirles adecuadamente para compatibilizar con los XT si así ha sido programado, enviarles a la CPU: el 8042 de Intel. También sirve de intermediario a las transmisiones de datos de la CPU al teclado, que en el AT es un periférico bidireccional que puede recibir comandos para configurar los LEDs, entre otras tareas. Cuando el 8042 recibe un byte entero del teclado, inhibe la comunicación hasta que la CPU lo acepta. Si el dato se recibe con error de paridad, automáticamente el 8042 lo solicita de nuevo al teclado enviando un comando de reenvío al mismo y un byte 0FFh a la CPU indicando esta circunstancia, activando también el bit 7 del registro de estado del 8042. Además, chequea que no pasen más de 2 milisegundos durante la recepción: si se excede este límite se envía también un 0FFh a la CPU y se activa el bit 6 en el registro de estado. Cuando la CPU envía algo al teclado, el 8042 inserta el bit de paridad automáticamente. Si el teclado no empieza la comunicación en menos de 15 milisegundos o tarda en recibir el dato más de 2 milisegundos, se envía un 0FEh a la CPU y se activa el bit 5 en el registro de estado. Además, el teclado ha de responder a todas las transmisiones con un byte de reconocimiento, si en esta operación hay un error de paridad se activarán los bits 5 y 7 en el registro de estado; si tarda más de 25 milisegundos en responder también se envía el byte 0FEh a la CPU y se activan los bits 5 y 6 del registro de estado.

La comunicación teclado-CPU puede ser inhibida por hardware por medio de la llave que incorpora la unidad central, aunque la comunicación CPU-teclado sigue habilitada. El 8042 se apoya en tres registros

básicos: uno de estado, uno de salida y otro de entrada. El registro de estado, del que ya se ha explicado parte de su funcionalidad, se encuentra en el puerto de E/S 64h y puede ser leído en cualquier momento. El significado de sus bits se explica en el cuadro 1.

El registro de salida está ubicado en el puerto 60h y es de sólo lectura; el 8042 lo usa para enviar los códigos de las teclas a la CPU y los bytes de datos de los comandos que los soliciten. Debería ser leído sólo cuando el bit 0 del registro de estado está activo.

El registro de entrada del 8042 es de sólo escritura y puede ser accedido por los puertos 60h y 64h según que lo que se quieran enviar sean datos o comandos al 8042, respectivamente; los datos serán reenviados por el 8042 hacia el teclado a menos que el propio 8042 esté esperando un dato de la CPU a consecuencia de un comando previo enviado por ésta. Los datos deben ser escritos en este registro sólo cuando el bit 1 del registro de estado esté inactivo. En el cuadro 2 se listan los comandos que admite el 8042 (enviados al puerto 64h). Debe darse cuenta el lector de la particularidad de que los registros de salida y entrada son accedidos por el mismo puerto (60h), siendo la lectura y escritura las que seleccionan el acceso a uno u otro respectivamente.

BIT	SIGNIFICADO
0	Registro de salida lleno. Un 1 indica que el 8042 ha colocado un dato en el registro de salida y la CPU aún no lo ha leído. Este bit se pone a 0 cuando la CPU lee el puerto 60h.
1	Registro de entrada lleno. Un 1 significa que ha sido colocado un dato en el registro de entrada y el 8042 aún no lo ha leído.
2	Banderín del sistema: asignado con un comando del 8042. 0 al arrancar.
3	Comando/dato. Se pone a 1 o a 0 al enviar algo al puerto 60h o al 64h respectivamente: de esta manera, el 8042 sabe si lo que se le envía son órdenes o datos (órdenes= 1). Ambos puertos conectan con el registro de entrada.
4	Bit de inhibición. Este bit se actualiza siempre que se coloca un dato en el registro de salida, un 0 indica teclado inhibido.
5	Transmisión fuera de tiempo. Indica que la transmisión de un dato hacia el teclado no ha sido respondida en los márgenes de tiempo adecuados.
6	Recepción fuera de tiempo. Indica si el teclado ha enviado un dato y sigue enviando más después del tiempo esperado.
7	Error de paridad. Indica la paridad del dato recibido: 0 la correcta.

CUADRO 1: REGISTRO DE ESTADO

12.8.2 - EL TECLADO DEL AT

Como se dijo en el apartado anterior, el teclado del AT es bidireccional y admite comandos por parte del ordenador. Estudiaremos ahora cuáles son esos comandos. En primer lugar, tras el arranque del ordenador y al recibir la alimentación el teclado, éste realiza un autotest denominado BAT (Basic Assurance Test) donde chequea su ROM, RAM y enciende y apaga todos los LED. Esta operación emplea entre 600 y 900 milisegundos; al acabar el BAT y cuando sea posible establecer la comunicación con el ordenador (líneas de reloj y datos en alto) envía un byte 0AAh si todo ha ido bien y un 0FCh si ha habido fallos; inicializando después los parámetros de autorepetición de las teclas.

El teclado tiene un buffer interno con capacidad para 17 bytes (unas 8 teclas) con objeto de almacenar las últimas teclas pulsadas cuando no puede enviarlas al 8042. Cuando este buffer se llena, su última posición (17ª) se rellena con 0 y se ignoran las siguientes pulsaciones.

12.8.3 - COMUNICACIÓN CPU —▸ TECLADO

Los comandos al teclado pueden ser enviados en cualquier momento al puerto 60h: a menos que el 8042 esté esperando por un byte de datos en el registro de entrada, como consecuencia de un comando previo, redireccionará todo lo que se le envíe por el puerto 60h hacia el teclado. El teclado responderá en menos de 20 milisegundos, devolviendo una señal de reconocimiento por medio de un byte 0FAh. Los

principales comandos (diferenciados de los datos por tener el bit 7 activo) son:

- Reset (0FFh): Al recibirlo envía una señal de reconocimiento y se asegura de que la CPU se de por enterada poniendo en alto las líneas de reloj y datos un mínimo de 500 microsegundos; el teclado permanece inhibido hasta que la CPU acepta la señal de reconocimiento o envía otro comando que sobreescribe y anula éste. Llegados a este punto, el teclado ejecuta de nuevo el BAT, estableciendo valores por defecto para la autorepetición y limpiando su registro de salida.

- Reenvío (0FEh): El sistema puede enviar este comando al teclado cuando detecta un fallo en la recepción desde el teclado. Este comando sólo puede ser enviado después de una transmisión del teclado y antes de habilitar la comunicación para la siguiente recepción. El teclado responde enviando de nuevo el dato anterior (si ya era un 0FEh, el último dato que envió que no fuera 0FEh).

COMANDO	SIGNIFICADO
20h	Leer el byte de comando del 8042 (ver cuadro 3). Esta orden envía al registro de salida (en el puerto 60h) dicho byte para que sea leído.
60h	Escribir el byte de comando del 8042. El siguiente byte que se envíe al registro de entrada (puerto 60h) será el byte de comando del 8042.
AAh	Autotest. El 8042 realiza un diagnóstico interno y coloca un 55h en el registro de salida si todo va bien.
ABh	Test del interface. El controlador chequea las líneas de reloj y datos devolviendo: 0 si no hay errores; 1: el reloj está demasiado en bajo, 2: está demasiado en alto; 3: la línea de datos está demasiado en bajo y 4: la línea de datos está demasiado en alto.
ACh	Volcado de diagnóstico. Envía al registro de salida, sucesivamente, 16 bytes de la RAM del 8042, el estado de los registros de entrada y salida y la palabra de estado del controlador.
ADh	Inhibir teclado. Esto activa el bit 4 del byte de comando del 8042.
AEh	Habilitar teclado. Esto baja el bit 4 del byte de comando del 8042.
C0h	Leer el puerto de entrada (véase cuadro 4). Esto obliga al 8042 a leer el puerto de entrada y colocar lo que lee en el registro de salida; sólo ha de emplearse este comando cuando el registro de salida está vacío.
D0h	Leer el puerto de salida. El 8042 lee el puerto de salida y lo coloca en el registro de salida; sólo debe emplearse este comando si dicho registro está vacío.
D1h	Escribir el puerto de salida (ver cuadro 5). El siguiente byte que se envíe al registro de entrada (puerto 60h) se colocará en el puerto de salida.
E0h	Leer entradas de testeo. El 8042 coloca en el registro de salida los bits de reloj (bit 0) y datos (bit 1) para permitir la comunicación directa con el teclado.
Fxh	Los bits 0 al 3 de este comando (la parte baja de este mismo comando) se relacionan con los bits 0 al 3 del puerto de salida del 8042; un 0 indica bit pulsado durante 6 microsegundos (apróx.) y un 1 que el bit no resulta modificado; ¡cuidado con el reset!.

CUADRO 2: COMANDOS DEL 8042

- Establecer valores por defecto (0F6h): Devuelve la autorepetición a los valores habituales, limpia su registro de salida y continúa rastreando las teclas si no estaba inhibido; es una especie de *reset en caliente*.

- Establecer valores por defecto y parar (0F5h): Similar al comando anterior, pero dejando de rastrear las teclas y permaneciendo inhibido hasta recibir más instrucciones.

- Habilitar (0F4): Reanuda el funcionamiento interrumpido por el comando anterior o algún otro.

- Establecer ratio y retardo de autorepetición (0F3h): Tras este comando debe enviarse otro inmediatamente a continuación, que se interpretará como dato, estableciendo los valores de autorepetición. De este segundo byte, el bit 7 estará siempre a cero; el valor de los bits 5 y 6, sumándole una unidad, indica el tiempo que ha de pasar desde que se pulsa una tecla hasta que comience a autorepetirse, en unidades de 0,25 segundos (±20%). Los bits 2, 1 y 0 forman un número A; los bits 4 y 3 forman otro número B; por medio de la siguiente fórmula se obtiene la tasa o ratio de autorepetición en «teclas por segundo»:

$$\frac{1}{(8 + A) * (2 \wedge B) * 0.00417}$$

Una vez recibido este comando, el teclado envía la acostumbrada señal de reconocimiento, deja de rastrear las teclas y espera por el parámetro de autorepetición, respondiendo al mismo con otra señal de reconocimiento y volviendo a rastrear las teclas. Si en lugar de recibir el parámetro recibe otro comando (bit 7 activo) dejará inalterados los valores de autorepetición y procesará dicho comando, aunque ¡cuidado!: permanecerá inhibido hasta que se le habilite con el comando 0F4h. Por defecto, el sistema establece una tasa de 10 caracteres por segundo y 0,5 segundos de espera (parámetro 4Ch).

BIT	SIGNIFICADO
0	Activar la interrupción del registro de salida lleno: un 1 indica que el 8042 genere una IRQ1 (INT 9) tras colocar un dato en el registro de salida (esto es lo normal).
1	Reservado (escribir 0).
2	Banderín del sistema. Este bit define el bit 2 del registro de estado.
3	Ignorar inhibición: con 1 se ignorará la función de inhibir el teclado.
4	Deshabilitar el teclado: un 1 baja la línea de reloj inhibiendo la comunicación del 8042 con el teclado.
5	Modo IBM PC. Con 1 no se traducen los códigos del teclado ni se controla la paridad.
6	IBM PC compatibilidad. Un 1 selecciona la conversión de los códigos del teclado para emular los del PC y XT, traduciendo los códigos de rastreo y generando un único byte al soltar las teclas. Puesto a 1 por la BIOS antes de cargar el DOS (compatibilidad).
7	Reservado (escribir 0).

CUADRO 3: BYTE DE COMANDO DEL 8042

BIT	SIGNIFICADO
0-3	Indefinidos
4	RAM del sistema. A 1 si instalada la extensión de 256 Kb.
5	A 0 si presente el puente (o «jumper») del fabricante.
6	Tipo de pantalla. 0 si la pantalla principal es de color y 1 si es monocroma.
7	0: el teclado ha sido bloqueado con la llave externa de la unidad central.

CUADRO 4: BYTE RECIBIDO POR EL PUERTO DE ENTRADA

BIT	SIGNIFICADO
0	Reset del sistema (como Ctrl-Alt-Del).
1	Línea A20: 0 fuerza la línea A20 de la CPU a 0, con lo que se prohíbe acceder a la memoria por encima de 1 Mb lo cual emula el direccionamiento de los PC/XT; un 1 deja que A20 la controle la CPU aunque hay PC's en que esto no basta.
2-3	Indefinidos.
4	Registro de salida lleno.
5	Registro de entrada vacío.
6	Línea de reloj (comunicación directa con el teclado).
7	Línea de datos (comunicación directa con el teclado).

CUADRO 5: BYTE A ENVIAR AL PUERTO DE SALIDA

- No operación (0F7h a 0FDh y 0EFh al 0F2h): Son códigos reservados; el teclado al recibirlos envía la señal de reconocimiento de siempre y no realiza ninguna acción.

- Eco (0EEh): Si el teclado recibe este comando, lo reenvía a continuación. Es una ayuda al diagnóstico.

- Encender/apagar los LED (0EDh). Tras este comando se ha de enviar otro byte de datos, cuyos bits 0, 1 y 2 están ligados al estado de los LED de Scroll Lock, Num Lock y Caps Lock, respectivamente; los demás están reservados. Al recibir el comando envía la correspondiente señal de reconocimiento y deja de rastrear las teclas, esperando por el dato. Si en vez de un dato recibe otro comando, dejará intactos los LED, procesará dicho comando y continuará rastreando las teclas (sin quedar inhibido en esta ocasión). El siguiente ejemplo muestra cómo establecer los LED configurados en AH:

```
CLI
MOV     AL,0EDh
OUT     60h,AL          ; enviar comando
XOR     CX,CX
```

```
espera:     JMP     SHORT $+2       ; insertar estados de espera para AT obsoleto
            JMP     SHORT $+2
            IN      AL,64h
            TEST    AL,2
            LOOPNZ  espera          ; esperar que reciba comando
            MOV     AL,AH
            OUT     60h,AL          ; establecer los LED
            STI
```

En general, este será el procedimiento a seguir para cualquier comando que requiera parámetros: hay que esperar el momento adecuado para enviarlos; el LOOPNZ evita que la CPU se quede *colgada* si por cualquier motivo fallara el teclado o el 8042. Como se ve, se establecen los 3 LED a la vez, aunque si sólo se desea cambiar uno habrá que consultar el estado actual de los otros en las variables de la BIOS. No obstante, este cambio es sólo puntual ya que al pulsar las teclas que actúan sobre los LED, la BIOS o el KEYB los reajustarán anulando el cambio, siendo necesario reprogramar parcialmente la interrupción del teclado si se desea evitarlo.

12.8.4 - COMUNICACIÓN TECLADO —▸ CPU

Más bien cabría llamarla la comunicación teclado —▸ 8042: aunque muchos de estos códigos acaben siendo interpretados por la CPU, algunos *se los queda* el 8042 que siempre es el primero en enterarse. A continuación se listan los valores que el teclado puede enviar a la CPU o al 8042 en un momento dado.

- Reenvío (0FEh): El teclado puede enviar este comando a la CPU para solicitar el reenvío cuando detecta un fallo en la recepción (normalmente de paridad) o una entrada incorrecta.

- Reconocimiento ó *ACK* (0FAh): El teclado devuelte este valor cada vez que la CPU le envía algo, para indicar que lo ha recibido (excepto en el caso de los comandos Eco y Reenvío de la CPU).

- Desbordamiento (0): Cuando la CPU intenta leer el teclado directamente sin haber códigos en el buffer del teclado (el buffer interno del propio teclado, se entiende) accederá a la posición 17ª del mismo, encontrándose este valor.

- Fallo en el diagnóstico (0FDh): El teclado periódicamente se autochequea y envía este código si detecta algún fallo. Si el fallo sucede durante el BAT, dejará de rastrear las teclas en espera de un comando de la CPU; en cualquier otro momento continuará rastreando las teclas.

- Código de tecla soltada ó *break code* (0F0h): El teclado envía este código a la CPU para indicar que el siguiente código que enviará a continuación corresponderá a una tecla soltada. Bajo MS-DOS este código lo intercepta el 8042 y se lo oculta a la CPU, con objeto de emular el código de tecla soltada de los PC/XT.

- BAT completado (0AAh): Después de realizar el BAT el teclado envía un 0AAh para indicar que ha salido bien, o un 0FCh (u otro valor) si ha habido fallos.

- Respuesta al eco (0EEh): El teclado envía este valor a la CPU si ésta se lo ha enviado a él.

la comunicación directa CPU —▸ teclado.

Debido a la presencia del 8042, normalmente no será preciso que la CPU se comunique directamente con el teclado a través de las líneas de reloj y datos. No obstante, este capítulo está explicado en el manual de referencia técnico del IBM AT, al menos en la edición de 1984; por tanto, aquellos aficionados que estén pensando construirse su propio ordenador y acoplarle un teclado ordinario de PC podrían consultar ese libro. Por cierto, en los PC y XT no es preciso tampoco realizar esta tarea, ya que el teclado con el conmutador de selección de la parte inferior en modo XT no es realmente bidireccional (de hecho, lleva un control autónomo de los LED) por lo que no tiene sentido intentar enviar nada. Y a la hora de recibir, hay métodos mucho más cómodos...

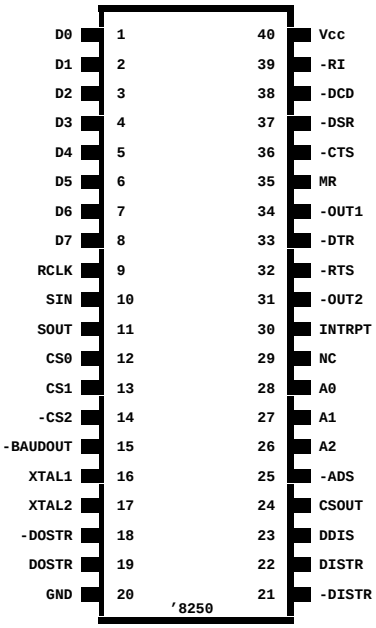
12.9. - EL PUERTO SERIE: UART 8250.

La transmisión de datos en serie es una de las más comunes para aquellas aplicaciones en las que la velocidad no es demasiado importante, o no es posible conseguirla (por ejemplo, vía red telefónica). Para simplificar el proceso de enviar los bits uno por uno han surgido circuitos integrados que realizan la función, teniendo en cuenta todos los tiempos necesarios para lograr una correcta comunicación y aliviando a la CPU de esta pesada tarea. El circuito que estudiaremos es el 8250 de National, fabricado también por Intel, aunque las diferencias respecto al 16550 serán brevemente señaladas. Esta última UART es más reciente y mucho más potente -aunque solo sea por unos pequeños detalles- y cada vez está más extendida, en particular en las actuales placas base.

La línea que transmite los datos en serie está inicialmente en estado alto. Al comenzar la transferencia, se envía un bit a 0 ó **bit de inicio**. Tras él irán los 8 **bits de datos** a transmitir (en ocasiones son 7, 6 ó 5): estos bits están espaciados con un intervalo temporal fijo y preciso, ligado a la velocidad de transmisión que se esté empleando. Tras ellos podría venir o no un bit de paridad generado automáticamente por la UART. Al final, aparecerá un bit (a veces un bit y medio ó dos bits) a 1, que son los bits de parada o **bits de stop**. Lo de medio bit significa que la señal correspondiente en el tiempo a un bit dura la mitad; realmente, en comunicaciones se utiliza el término **baudio** para hacer referencia a las velocidades, y normalmente un baudio equivale a un bit. La presencia de bits de inicio y parada permite sincronizar la estación emisora con la receptora, haciendo que los relojes de ambas vayan a la par. A la hora de transmitir los bytes de datos unos tras otros, existe flexibilidad en los tiempos, de ahí que este tipo de comunicaciones se consideren **asíncronas**. La transmisión de los 8 bits de datos de un byte realmente es síncrona, pero las comunicaciones en serie siempre han sido consideradas asíncronas.

Para una transmisión en serie básica bastan tres hilos. Sin embargo, el software que controla el puerto serie a través de la interfaz RS-232-C podría requerir más señales de control para establecer la comunicación, al igual que para controlar un modem telefónico pueden hacer falta más líneas (de control, no telefónicas...). Bromas aparte, sobre comunicaciones en serie existe todo un mundo; acerca de este tema se han escrito muchos libros completos. Lógicamente, aquí no vamos a dar ningún curso de comunicaciones en serie. Sin embargo, los menos introducidos en la materia no deben temer: ¿qué mejor manera de aprender sobre las comunicaciones en serie que examinar cómo funciona un chip que las soporta?. Desde luego, también se podría partir desde el punto de vista contrario, pero como entendido en sistemas digitales, el lector puede que tenga menos problemas con este interesante enfoque.

12.9.1. - DESCRIPCIÓN DEL INTEGRADO.



El ACE 8250 (Asynchronous Communication Element) integra en un solo chip una UART (Universal Asynchronous Receiver/Transmitter) y un BRG (Baud Rate Generator). Soporta velocidades de hasta 625000 baudios con relojes de hasta 10 MHz. El BRG incorporado divide la frecuencia base para conseguir las velocidades estándar de la RS-232-C.

SIGNIFICADO DE LAS LÍNEAS DEL 8250

DISTR: Data In Strobe. Línea de entrada que indica al 8250 que deje los datos en el bus (D0..D7), los datos dejados dependen del registro seleccionado con A0..A2. Son necesarias CS0..CS2 para habilitar DISTR. En vez de DISTR se puede usar -DISTR, pero sólo una de las dos.

DOSTR: Data Out Strobe. Idéntico a DISTR pero en salida.

D0..D7: Data Bits 0..7: Bus triestado bidireccional de 8 líneas para transmitir datos, información de control y de estado entre la CPU y el 8250. El primer bit enviado/recibido es D0.

A0..A2: Register Select. Líneas de entrada que indican el registro del 8250 usado en la operación.

XTALx: Crystal/Clock: Conexiones para el cristal del cuarzo del BRG. XTAL1 puede actuar como entrada de reloj externa, en cuyo caso XTAL2 debería quedar abierto.

SOUT: Serial Data Output: Salida de datos en serie del 8250. Una marca es un '1' y un espacio es un '0'. SOUT está en marca cuando el transmisor está inhibido, MR está a 1, el registro de transmisión está vacío o en el modo lazo (LOOP) del 8250. No es afectado por -CTS.

-CTS: Clear To Send: Línea de entrada. El estado lógico de esta señal puede consultarse en el bit CTS del Modem Status Register (MSR) -como el bit CTS es el bit 4 del MSR se

referencia MSR(4)-. Un cambio en el estado de -CTS desde la última lectura del MSR provoca que se active DCTS (bit MSR(0)). Cuando -CTS está activo (a 0) el modem indica que el dato en SOUT puede ser transmitido. -CTS no afecta al modo lazo (LOOP) del 8250.

-DSR: Data Set Ready: Línea de entrada. El estado lógico de esta señal puede consultarse en MSR(5). DDSR (bit MSR(1)) indica si -DSR ha cambiado desde la última lectura del MSR. Cuando -DSR está activo el modem indica que está listo para intercambiar datos con el 8250; ello depende del estado del DCE (Data Communications Equipment) local y no implica que haya comunicación con la estación remota.

-DTR: Data Terminal Ready. Línea de salida que puede activarse (poner a 0) escribiendo un 1 en MCR(0), y desactivarse escribiendo un 0 en dicho bit o ante la activación del pin MR. Con -DTR activo se indica al DCE que el 8250 puede recibir datos. En algunas circunstancias, esta señal se usa como LED de 'power on'. Si está inactivo, el DCE desconecta el modem del circuito de telecomunicaciones.

-RTS: Request To Send. Línea de salida que habilita el modem. Se activa (poner a 0) escribiendo un 1 en MCR(1). Esta señal se pone en alto en respuesta a MR. -RTS indica al DCE que el 8250 tiene un dato listo para transmitir. En la modalidad half-duplex, esta señal se utiliza para controlar la dirección de la línea.

-BAUDOUT: Esta línea de salida contiene una señal de reloj 16 veces mayor que la frecuencia usada para transmitir. Equivale a la frecuencia de entrada en el oscilador dividida por el BRG. La estación receptora podría emplear esta señal conectándola a RCLK (para compartir el mismo reloj).

-OUTx: Estas dos salidas de propósito general se pueden activar (poner a 0) escribiendo un 1 en MCR(2) y MCR(3). Son desactivadas por la señal MR. En el modo lazo (LOOP o bucle), están también inactivas.

-RI: Ring Indicator. Esta línea de entrada indica si el modem ha detectado que llaman por la línea y puede consultarse en MSR(6). El bit TERI (MSR(2)) indica si esta línea ha cambiado desde la última lectura del MSR. Si las interrupciones están habilitadas (IER(3) activo) esta patilla provoca una interrupción al activarse. -RI permanece activo durante el mismo intervalo de tiempo que la zona activa del ciclo de llamada e inactivo en los intervalos de la zona inactiva (o cuando el DCE no detecta la llamada). El circuito no se corta por culpa de -DTR.

-DCD: Data Carrier Detect. Línea de entrada que indica si el modem ha detectado portadora. Se puede consultar su estado lógico en MSR(7). El bit MSR(3) indica si esta línea ha cambiado desde la última lectura del MSR. Esta línea no tiene efecto sobre el receptor. Si las interrupciones están permitidas, una interrupción será generada ante el cambio de esta línea.

MR: Master Reset. Esta línea de entrada lleva el 8250 a un estado inactivo interrumpiendo su posible actividad. El MCR y las salidas ligadas al mismo son borradas. El LSR es borrado en todos sus bits salvo THRE y TEMT (que son activados). El 8250 permanece en este estado hasta volver a ser programado.

INTRPT: Interrupt Request. Línea de salida que se activa cuando se produce una interrupción de alguno de estos tipos y está permitida: Recepción de banderín de error, dato recibido disponible, registro de retención de transmisión vacío, y estado del modem. Esta línea se desactiva con el apropiado servicio de la interrupción o ante MR.

SIN: Serial Data Input. Es la línea de entrada de datos desde el modem. En el modo lazo (LOOP o bucle) están inhibidas las entradas en SIN.

CS0..2: Chip Select. Estas entradas actúan como líneas de habilitación para las señales de escritura (DOSTR, -DOSTR) y lectura (DISTR, -DISTR).

CSOUT: Chip Select Out. Esta línea de salida se activa cuando el chip ha sido seleccionado con CS0..2. No comenzará transferencia de datos alguna hasta que CSOUT se active.

DDIS: Driver Disable. Esta salida está inactiva cuando la CPU lee datos del 8250. Una salida activa puede emplearse para inhibir un transceiver externo cuando la CPU está leyendo datos.

-ADS: Address Strobe. Cuando esta línea de entrada está activa se enclavan las líneas A0..A2 y CS0..2; esto puede ser necesario si los pines de selección de registro no son estables durante la duración de la operación de lectura o escritura (modo multiplexado). Si esto no es preciso, esta señal se puede mantener inactiva (modo no-multiplexado).

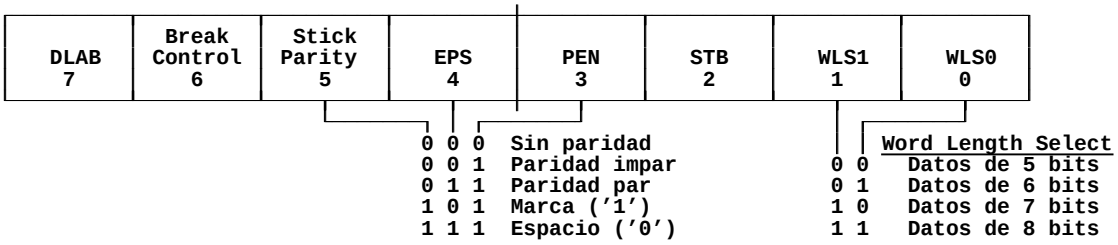
RCLK: Esta línea se corresponde con la entrada de reloj para la sección receptora, equivalente a 16 veces la frecuencia empleada en la transmisión y puede proceder del BAUDOUT de la estación remota o de un reloj externo.

REGISTROS DEL 8250

El 8250 dispone de 11 registros (uno más el 16550) pero sólo 3 líneas de dirección para seleccionarlos. Lo que permita distinguir unos de otros será, aparte de las líneas de direcciones, el sentido del acceso (en lectura o escritura) y el valor de un bit de uno de los registros: el bit DLAB del registro LCR, que es el bit 7 de dicho registro. La notación para hacer referencia a un bit de un registro se escribe REG(i); en este ejemplo, el bit DLAB sería LCR(7). Realmente, DLAB se emplea sólo puntualmente para poder acceder y programar los registros que almacenan el divisor de velocidad; el resto del tiempo, DLAB estará a 0 para acceder a otros registros más importantes.

A2	A1	A0	DLAB	MODO	NOMBRE	SIGNIFICADO
0	0	0	0	R	RBR	Receiver Buffer Register (Registro buffer de recepción)
0	0	0	1	R/W	DLL	Divisor Latch LSB (Divisor de velocidad, parte baja)
0	0	0	0	W	THR	Transmitter Holding Register (Registro de retención de transmisión)
0	0	1	0	R/W	IER	Interrupt Enable Register (Registro de habilitación de interrupciones)
0	0	1	1	R/W	DLM	Divisor latch MSB (Divisor de velocidad, parte alta)
0	1	0	X	R	IIR	Interrupt Identification Register (Registro de identificación de interrupciones)
0	1	0	X	W	FCR	FIFO Control Register (Registro de control FIFO) - SOLO 16550 -
0	1	1	X	R/W	LCR	Line Control Register (Registro de control de línea) **¡¡EL BIT 7 ES DLAB!!**
1	0	0	X	R/W	MCR	Modem Control Register (Registro de control del modem)
1	0	1	X	R/W	LSR	Line Status Register (Registro de estado de la línea)
1	1	0	X	R/W	MSR	Modem Status Register (Registro de estado del modem)
1	1	1	X	R/W	SCR	Scratch Register (Registro residual)

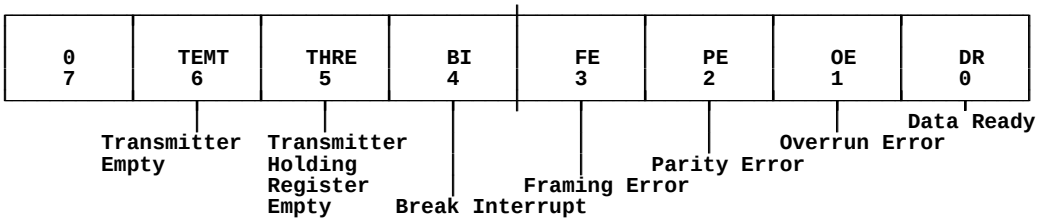
1) LCR (Line Control Register). Controla el formato del carácter de datos.



Los bits WLS seleccionan el tamaño del dato empleado. STB indica el número de bits de stop, que pueden ser 1 (STB=0) ó 2 (STB=1), al trabajar con datos de 5 bits STB=1 implica 1.5 bits de stop. PEN (Parity Enable) permite habilitar o no la generación de bit de paridad, EPS (Even Parity Select) selecciona paridad par si está a 1 (o impar en caso contrario). Stick Parity permite forzar el bit de paridad a un estado conocido según el valor de EPS. Cuando Break Control es puesto a 1, la salida SOUT se pone en estado espacio (a 0), sólo afecta a SOUT y no a la lógica de transmisión. Esto permite a la CPU alertar a un terminal del sistema sin transmitir caracteres erróneos o extraños si se siguen estas fases: 1) cargar un carácter 0 en respuesta a THRE, 2) activar Break Control en respuesta al próximo THRE, 3) esperar a que el transmisor esté inactivo (TEMT=1) y bajar Break Control. Durante el Break, el transmisor puede usarse como un preciso temporizador de carácter.

El bit DLAB (Divisor Latch Access Bit) puesto a 1 permite acceder a los Latches divisores DLL y DLM del BRG en lectura y escritura. Para acceder al RBR, THR y al IER debe ser puesto a 0.

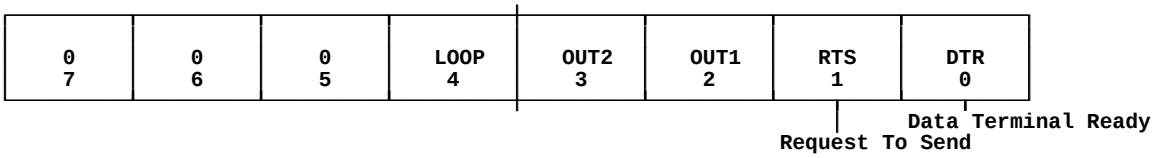
2) LSR (Line Status Register). Este suele ser el primer registro consultado tras una interrupción.



DR está activo cuando hay un carácter listo en el RBR y es puesto a 0 cuando se lee el RBR. Los bits 1 al 4 de este registro (OE, PE, FE y BI) son puestos a 0 al consultarlos -cuando se lee el LSR- y al activarse pueden generar una interrupción de prioridad 1 si ésta interrupción está habilitada. OE se activa para indicar que el dato en el RBR no ha sido leído por la CPU y acaba de llegar otro que lo ha sobreescrito. PE indica si hay un error de paridad. FE indica si el carácter recibido no tiene los bit de stop correctos. BI se activa cuando la entrada de datos es mantenida en espacio (a 0) durante un tiempo superior al de transmisión de un carácter (bit de inicio + bits de datos + bit de paridad + bit de parada).

THRE indica que el 8250 puede aceptar un nuevo carácter para la transmisión: este bit se activa cuando el THR queda libre y se desactiva escribiendo un nuevo carácter en el THR. Se puede producir, si está habilitada; la interrupción THRE (prioridad 3); INTRPT se borra leyendo el IIR. El 8250 emplea un registro interno para ir desplazando los bit y mandarles en serie (el Transmitter Shift Register), dicho registro se carga desde el THR. Cuando ambos registros (THR y el Transmitter Shift) están vacíos, TEMT se activa; volverá a desactivarse cuando se deje otro dato en el THR hasta que el último bit salga por SOUT.

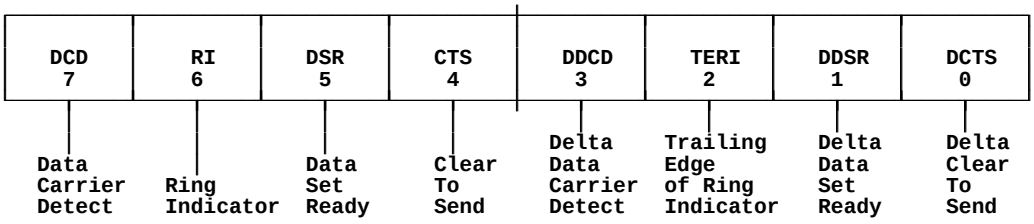
3) MCR (Modem Control Register). Controla el interface con el modem.



Las líneas de salida -DTR, -RTS, -OUT1 y -OUT2 están directamente controladas por estos bits; como se activan a nivel bajo, son puestas a 0 escribiendo un 1 en estos bits y viceversa. Estas líneas sirven para establecer diversos protocolos de comunicaciones.

El bit LOOP introduce el 8250 en un modo lazo (o bucle) de autodiagnóstico. Con LOOP activo, SOUT pasa a estado de marca (a 1) y la entrada SIN es desconectada. Los registros de desplazamiento empleados en la transmisión y la recepción son conectados entre sí. Las cuatro entradas de control del modem (-CTS, -DSR, DC y -RI) son desconectadas y en su lugar son internamente conectadas las cuatro salidas de control del modem (-DTR, -RTS, -OUT1 y -OUT2) cuyos pines son puestos en estado inactivo (alto). En esta modalidad de operación (modo lazo o bucle), los datos transmitidos son inmediatamente recibidos, lo que permite comprobar el correcto funcionamiento del integrado. Las interrupciones son completamente operativas en este modo, pero la fuente de estas interrupciones son ahora los 4 bits bajos del MCR en lugar de las cuatro entradas de control. Estas interrupciones están aún controladas por el IER.

4) MSR (Modem Status Register).



Además de la información de estado del modem, los 4 bits bajos (DDCD, TERI, DDSR, DCTS) indican si la línea correspondiente, en los 4 bits superiores, ha cambiado de estado desde la última lectura del MSR; en el caso de TERI sólo indica transiciones bajo-▸alto en -RI (y no las de sentido contrario). La línea CTS del modem indica si está listo para recibir datos del 8250 a través de SOUT (en el modo lazo este bit equivale al bit RTS del MCR). La línea DSR del modem indica que está listo para dar datos al 8250 (en el modo lazo -o LOOP- equivale al bit DTR del MCR). RI y DCD indican el estado de ambas líneas (en el modo lazo se corresponden con OUT1 y OUT2 respectivamente). Al leer el MSR, **se borran** los 4 bits inferiores (que en una lectura posterior estarían a 0) pero no los bits de estado (los 4 más significativos).

Los bits de estado (DCD, RI, DSR y CTS) reflejan siempre la situación de los pines físicos respectivos (estado del modem). Si DDCD, TERI, DDSR ó DCTS están a 1 y se produce un cambio de estado durante la lectura, dicho cambio no será reflejado en el MSR; pero si están a 0 el cambio será reflejado después de la lectura. Tanto en el LSR como en el MSR, la asignación de bits de estado está inhibida durante la lectura del registro: si se produce un cambio de estado durante la lectura, el bit correspondiente será activado después de la misma; pero si el bit ya estaba activado y la misma condición se produce, el bit será borrado tras la lectura en lugar de volver a ser activado.

5) y 6) BRSR (Baud Rate Select Register). Son los registros DLL (parte baja) y DLM (parte alta).

Estos dos registros de 8 bits constituyen un valor de 16 bits que será el divisor que se aplicará a la frecuencia base para seleccionar la velocidad a emplear. Dicha frecuencia base (por ejemplo, 1.8432 MHz) será dividida por 16 veces el valor almacenado aquí. Por ejemplo, para obtener 2400 baudios:

$$\frac{1843200}{16 * 2400} = 48 \text{ -▸ } \text{DLL=48, DLM=0}$$

7) RBR (Receiver Buffer Register).

El circuito receptor del 8250 es programable para 5, 6, 7 u 8 bits de datos. En el caso de emplear menos de 8, los bits superiores de este registro quedan a 0. Los datos entran en serie por SIN (comenzando por el bit D0) en un registro de desplazamiento gobernado por el reloj de RCLK, sincronizado con el bit de

inicio. Cuando un carácter completa el registro de desplazamiento de recepción, sus bits son volcados al RBR y el bit DR del LSR es activado para indicar a la CPU que puede leer el RBR. El diseño del 8250 permite la recepción continua de datos sin pérdidas: el RBR almacena siempre el último carácter recibido dando tiempo suficiente a la CPU para leerlo mientras simultáneamente está cargando el registro de desplazamiento con el siguiente; si la CPU tarda demasiado un nuevo dato podría aparecer en el RBR antes de haber leído el anterior (condición de overrun, bit OE del LSR).

8) THR (Transmitter Holding Register).

El registro de retención de transmisión almacena el siguiente carácter que va a ser transmitido en serie mientras el registro de desplazamiento de transmisión está enviando el carácter actual. Cuando el registro de desplazamiento se vacíe, será cargado desde el THR para transmitir el nuevo carácter. Al quedar vacío THR, el bit THRE del LSR se activa. Cuando estén vacíos tanto el THR como el registro de desplazamiento de transmisión, el bit TEMT del LSR se activa.

9) SCR (Scratchpad Register).

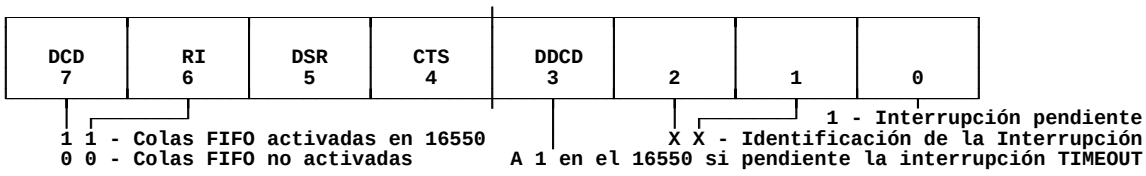
Este registro no es empleado por el 8250, y de hecho no existía en las primeras versiones del integrado. Puede ser empleado por el programador como una celdilla de memoria.

10) IIR (Interrupt Identification Register).

Existen 4 niveles de prioridad en las interrupciones generables por el 8250, por este orden:

1) Estado de la línea de recepción.
2) Dato recibido disponible.
3) Registro de retención de transmisión vacío.
4) Estado del modem.

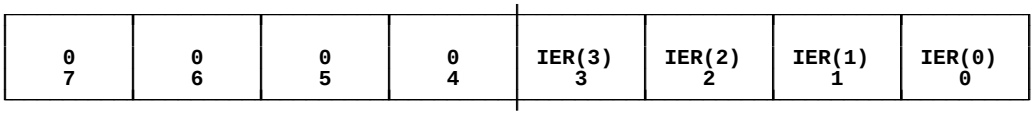
La información que indica que hay una interrupción pendiente y el tipo de la misma es almacenada en el IIR. El IIR indica la interrupción de mayor prioridad pendiente. No serán reconocidas otras interrupciones hasta que la CPU envíe la señal de reconocimiento apropiada. En el registro IIR, el bit 0 indica si hay una interrupción pendiente (bit 0=0) o si no la hay (bit 0=1), esto permite tratar las interrupciones en modo *polled* consultando este bit. Los bits 1 y 2 indican el tipo de interrupción. Los restantes están a 0 en el 8250, pero el 16550 utiliza alguno más.



IDENTIFICACIÓN DE LA INTERRUPCIÓN				ACTIVACIÓN / RECONOCIMIENTO (RESET) DE LA INTERRUPCIÓN		
Bit 2	Bit 1	Bit 0	Prioridad	Flag	Fuente	Reconocimiento
X	X	1		Ninguno	Ninguna	
1	1	0	Primera	Línea de estado del receptor	OE, PE, FE ó BI	Leer LSR
1	0	0	Segunda	Recibido dato disponible	Recibido dato disponible	Leer RBR
0	1	0	Tercera	THRE	THRE	Leer IIR si es la fuente de interrupción, o escribir THR
0	0	0	Cuarta	Estado del modem	-CTS, -DSR -RI, -DCD	Leer MSR

11) IER (Interrupt Enable Register).

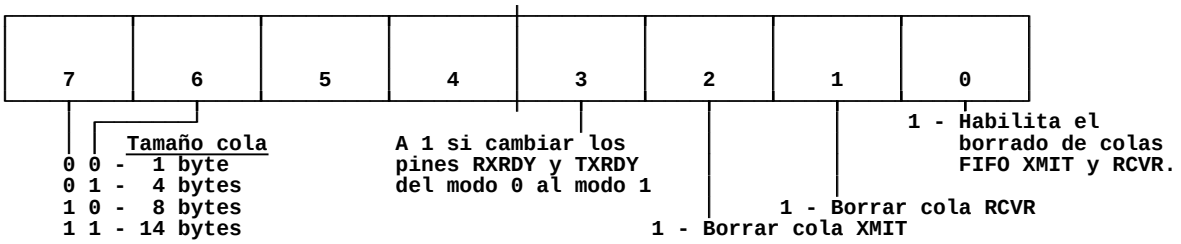
Este registro de escritura se utiliza para seleccionar qué interrupciones activan INTRPT y, por consiguiente, van a ser solicitadas a la CPU. Deshabilitar el sistema de interrupciones inhibe el IIR y desactiva la salida INTRPT.



IER0: A 1 si habilitar interrupción de dato disponible.
IER1: A 1 si habilitar interrupción de registro de retención de transmisión vacío.
IER2: A 1 si habilitar interrupción de error de recepción (bits 1 al 4 del LSR).
IER3: A 1 si habilitar interrupción ante el cambio del MSR (Registro de estado del modem).

El 16550 genera también una interrupción de TIMEOUT (prioridad 1) si hay datos en la cola FIFO y no son leídos dentro del tiempo que dura la recepción de 4 bytes o si no se reciben datos durante el tiempo que tomaría recibir 4 bytes.

12) FCR (FIFO Control Register). Sólo disponible en el 16550, no en el 8250.



El bit 0 debe estar a 1 para escribir los bits 1 ó 2. Cuando el bit 1 ó el 2 son activados, la cola afectada es borrada y el bit es devuelto a 0. Los registros de desplazamiento de la transmisión y la recepción, en cada caso, no resultan afectados.

LA TRANSMISIÓN Y LA RECEPCIÓN EN EL 8250

La sección de transmisión del 8250 consiste en el Registro de Retención de transmisión (THR), el Registro de Desplazamiento de la Transmisión (TSR) y en la lógica de control asociada. Dos bits en el LSR indican si está vacío el THR (bit THRE) o el TSR (bit TEMT). El carácter de 5-8 bits a ser transmitido es escrito en el THR; la CPU debería realizar esta operación sólo si THRE está activo: este bit es activado cuando el carácter es copiado del THR al TSR durante la transmisión del bit de inicio.

Cuando el transmisor está inactivo, tanto THRE como TEMT están activos. El primer carácter escrito provoca que THRE baje; tras completarse la transferencia vuelve a subir aunque TEMT permanecerá bajo mientras dure la transferencia en serie del carácter a través de TSR. Si un segundo carácter es escrito en THR, THRE vuelve a bajar y permanecerá bajo hasta que el TSR termine la transmisión, porque no es posible volcar el contenido de THR en TSR hasta que este último no acabe con el carácter que estaba transmitiendo. Cuando el último carácter ha sido transmitido fuera del TSR, TEMT vuelve a activarse y THRE también lo hará tras un cierto tiempo (el que tarda en escribirse THR en TSR).

En la recepción, los datos en serie asíncronos entran por la patilla SIN. El estado inactivo de la línea se considera el ’1’ lógico. Un circuito de detección de bit de inicio está continuamente buscando una transición alto—▸bajo que interrumpa el estado inactivo. Cuando la detecta, se resetea un contador interno y cuenta 7½ pulsos de reloj (tener en cuenta que la frecuencia base es dividida por 16), posicionándose en el centro del bit de inicio. El bit de inicio se considera válido si SIN continúa aún bajo en ese momento. La validación del bit de inicio evita que un ruido espúreo en la línea sea confundido con un nuevo carácter.

El LCR tiene toda la información necesaria para la recepción: tamaño del carácter (5-8 bits), número de bits de stop, si hay paridad o no... la información de estado que se genere será depositada en el LSR. Cuando un carácter es transmitido desde el Registro de Desplazamiento de la Recepción (RSR) al Registro Buffer de Recepción (RBR), el bit DR del LSR se activa. La CPU lee entonces el RBR, lo que hace bajar de nuevo DR. Si el carácter no es leído antes de que el siguiente carácter que se está formando pase del RSR al RBR, el bit OE (overrun) del LSR se activa. También se puede activar PE en el LSR si hay un error de paridad. Finalmente, la circuitería que chequea la validez del bit de stop podría activar el bit FE del LSR en caso de error.

El centro del bit de inicio se define como 7½ pulsos de reloj; si los datos que entran por SIN constituyen una onda cuadrada simétrica, el centro de las celdas que contienen los bits se desviará a lo sumo un ±3.125% del centro real, lo que deja un margen de error del 46.875%; el bit de inicio puede comenzar, como mucho, 1 ciclo de reloj (de los 16) antes de ser detectado.

EL B.R.G. (BAUD RATE GENERATOR)

El BRG genera las señales de reloj para el funcionamiento de la UART, permitiendo los ratios de transferencia del estándar ANSI/CCITT. Se puede conectar un cristal a XTAL1 y XTAL2 ó una señal de reloj a XTAL1. La salida -BAUDOUT puede excitar la línea XTAL1 de otro 8250.

La velocidad es determinada por los registros DLL y DLM almacenando un valor divisor de la frecuencia del reloj conectado al 8250. El resultado debe ser 16 veces mayor que la frecuencia en baudios deseada, ya que el 8250 utiliza 16 pulsos de reloj para cada bit. El siguiente cuadro resume los valores que hay que asignar al divisor para lograr las frecuencias más usuales con los cristales más comunes.

	Cristal de 1.8432 MHz		Cristal de 2.4576 MHz		Cristal de 3.072 MHz	
Baudios finales	Divisor usado para 16xReloj	% error si lo hay	Divisor usado para 16xReloj	% error si lo hay	Divisor usado para 16xReloj	% error si lo hay
50	2304		3072		3840	
75	1536		2048		2560	
110	1047	0.026	1396	0.026	1745	0.026
134.5	857	0.058	1142	0.0007	1428	0.034
150	768		1024		1280	
300	384		512		640	
600	192		256		320	
1200	96		128		160	
1800	64		85	0.392	107	0.315
2000	58	0.69	77	0.260	96	
2400	48		64		80	
3600	32		43	0.775	53	0.628
4800	24		32		40	
7200	16		21	1.587	27	1.23
9600	12		16		20	
19200	6		8		10	
38400	3		4		5	
56000	2	2.86	-		-	

RESET DEL 8250

Tras dar corriente al 8250 hay que tenerlo unos 500 ns con MR alto para resetearlo. Un nivel alto en MR provoca:

1) Se inicializan los contadores internos de transmisión y recepción.
2) Se limpia el LSR salvo en sus bits TEMT y THRE (que son puestos a 1).
   MCR, todas las líneas discretas, elementos de memoria y demás son puestos a 0.
   DLL y DLM, RBR y THR no son afectados.

Tras el reset (MR llevado a estado bajo) el 8250 permanece en estado inactivo hasta ser programado. Un reset hardware activa THRE y TEMT: cuando las interrupciones sean habilitadas, THRE provocará una.

Por software se puede forzar al 8250 a retornar a un estado totalmente conocido. Dicho reset consiste en escribir el LCR, DLL y DLM, así como MCR. LSR y RBR deberían ser leídos antes de habilitar las interrupciones para borrar cualquier información residual (datos o estado) de las operaciones anteriores.

REGISTRO / SEÑAL	CONTROL DEL RESET	EFECTO DEL RESET EN EL 8250
IER	MR	Todos los bits a 0 (4..7 ya lo estaban)
IIR	MR	Bit 0 a 1, Bits 1 y 2 a 0, demás siempre a 0
LCR	MR	Todos los bits a 0
MCR	MR	Todos los bits a 0
LSR	MR	Todos los bits a 0, salvo el 5 y el 6 (a 1)
MSR	MR	Bits 0..3 a 0, bits 4..7 señal de entrada
SOUT	MR	En alto
INTRPT (RCVR error)	Leer LSR / MR	En bajo
INTRPT (RCVR dato listo)	Leer RBR / MR	En bajo
INTRPT (THRE)	Leer IIR / Escribir THR / MR	En bajo
INTRPT (Cambios en el estado del modem)	Leer MSR / MR	En bajo
-OUT2	MR	En alto
-RTS	MR	En alto
-DTR	MR	En alto
-OUT1	MR	En alto

PROGRAMACIÓN DEL 8250

El 8250 se programa a través de los registros de control LCR, IER, DLL, DLM y MCR. Aunque los registros de control pueden ser escritos en cualquier orden, IER debe ser escrito al final porque controla la habilitación de las interrupciones. Una vez que el 8250 ha sido programado, los registros pueden ser actualizados en cualquier momento en que el 8250 no se encuentre enviando o recibiendo datos.

12.9.2. - EL 8250 EN EL ORDENADOR.

Los ordenadores compatibles pueden tener conectados, de manera normal, hasta 4 puertos serie, nombrados COM1-COM4. En el área de datos de la BIOS (segmento 40h) y justo al principio de la misma, hay 4 palabras con la dirección de memoria base de los puertos serie. A esta dirección de memoria base habrá que sumar el desplazamiento relativo del número de registro a ser accedido.

El principal problema reside en que sólo están previstas 2 interrupciones para los puertos serie. Ello implica que generalmente sólo 2 de los puertos podrán emplear interrupciones a un tiempo, debido a la arquitectura del bus ISA. Generalmente COM1 y COM3 compartirán la IRQ4 (INT 0Ch) y COM2/COM4 la IRQ3 (INT 0Bh). Estas asignaciones pueden ser cambiadas por el usuario actuando sobre los switches de configuración de las tarjetas (que en ocasiones permiten incluso elegir la IRQ5). Por tanto, no está de más tener cuidado en los programas y permitir un cierto grado de configuración en estas cuestiones.

OFFSET	DLAB	MODO	NOMBRE	SIGNIFICADO
0	0	R	RBR	Receiver Buffer Register (Registro buffer de recepción)
0	1	R/W	DLL	Divisor Latch LSB (Divisor de velocidad, parte baja)
0	0	W	THR	Transmitter Holding Register (Registro de retención de transmisión)
1	0	R/W	IER	Interrupt Enable Register (Registro de habilitación de interrupciones)
1	1	R/W	DLM	Divisor latch MSB (Divisor de velocidad, parte alta)
2	X	R	IIR	Interrupt Identification Register (Registro de identificación de interrupciones)
2	X	W	FCR	FIFO Control Register (Registro de control FIFO) - SOLO 16550 -
3	X	R/W	LCR	Line Control Register (Registro de control de línea) **¡¡EL BIT 7 ES DLAB!!**
4	X	R/W	MCR	Modem Control Register (Registro de control del modem)
5	X	R/W	LSR	Line Status Register (Registro de estado de la línea)
6	X	R/W	MSR	Modem Status Register (Registro de estado del modem)
7	X	R/W	SCR	Scratch Register (Registro residual)

El cuadro superior muestra los desplazamientos (offsets) que hay que sumar a la dirección E/S base del puerto serie para acceder a sus registros. COM1 suele estar en 3F8h, COM2 en 2F8h, COM3 en 3E8h y COM4 en 2E8h. Sin embargo, es mejor acceder a las variables de la BIOS para obtener la dirección.

La INT 14h de la BIOS se encarga de controlar el puerto serie. El trabajo del DOS a través de los dispositivos COM1: (conocido también como AUX:) al COM4: se realiza también apoyándose en esta interrupción. El comando MODE del sistema permite inicializar el puerto serie a *alto nivel*. Sin embargo, tanto el DOS como la BIOS no permiten exceder los 9600 baudios, velocidad excesivamente baja para la transmisión de datos entre dos ordenadores cercanos o el trabajo con un modem.

El cristal que gobierna el 8250 oscila a 1.8432 MHz. Nosotros debemos considerar esta frecuencia dividida por 16 de cara a calcular el valor para el divisor. Por tanto, la velocidad máxima que puede alcanzar el puerto serie de los PC es de 1843200/16 = 115200 baudios. Con datos de 8 bit se pueden empaquetar los bytes en *10 baudios* (1 bit de inicio, 8 de datos, 1 de stop), lo que permite alcanzar 11520 bytes/seg (11.25 Kb/seg). Para distancias de pocos metros (no decenas ni centenas) no habrá problemas, incluso para distancias algo mayores si los cables se diseñan con cuidado. La programación del puerto serie en el PC a nivel de hardware es necesaria a menudo por dos razones de mucho peso: poder utilizar interrupciones y emplear velocidades superiores a 9600 baudios. Por supuesto, en estas transferencias los paquetes deberían llevar algún control de errores, aunque no precisamente basado en la paridad.

Baudios más comunes	Divisor a emplear en el PC		
	Divisor	DLM	DLL
50	2304	9	0
110	1047	4	23
150	768	3	0
300	384	1	128
1200	96	0	96
2400	48	0	48
4800	24	0	24
9600	12	0	12
14400	8	0	8
19200	6	0	6
28800	4	0	4
38400	3	0	3
57600	2	0	2
115200	1	0	1

Nota: El bit OUT2 del MCR controla en los PC la salida de la línea INTRPT. Esto significa que si dicho bit, por defecto inicializado a 0, es puesto a 1, las interrupciones del puerto serie quedan inhibidas. El bit OUT1, por el contrario, debe estar a 1 por motivos no muy claros. También se podría inhibir la INTRPT a través del 8259, por lo que este dato no es muy importante, con la excepción de evitar que una involuntaria e incorrecta asignación de OUT1 y OUT2 inhiba las interrupciones. La ventaja de inhibir las interrupciones en el 8250 radica en la posibilidad de utilizar plenamente todas sus funciones incluso en el modo de no interrupciones: el olvido del diseñador de incluir esta característica obligó a IBM a utilizar para este fin OUT2. Realmente, el 8250 está concebido para ser utilizado por medio de interrupciones, y hay quien duda incluso de la veracidad de la afirmación del fabricante acerca del *double buffering* (buffers duplicados) que son muy aconsejables al trabajar sin interrupciones.

12.9.3. - EJEMPLO: AUTODIAGNÓSTICO DEL 8250.

El siguiente programa de ejemplo coloca el 8250 en modo lazo (LOOP) y seguidamente comienza a transmitir datos de 8 bits (desde 0 hasta 255) comprobando que le llegan los mismos datos que envía y sin que se produzcan errores. Se permite elegir el puerto deseado así como la velocidad de transmisión.

```
/*********************************************************************
*                                                                    *
*  8250T.C  1.0  -  UTILIDAD DE AUTODIAGNOSTICO DEL 8250 EN TURBO C  *
*                                                                    *
*                   (c) 1993 Ciriaco García de Celis.                *
*                                                                    *
*********************************************************************/

#include <dos.h>
#include <conio.h>

#define LCR (base+3)    /* registro de control de línea */
#define IER (base+1)    /* registro de activación de interrupciones */
#define DLL (base+0)    /* parte baja del divisor */
#define DLM (base+1)    /* parte alta del divisor */
#define MCR (base+4)    /* registro de control del modem */
#define LSR (base+5)    /* registro de estado de línea */
#define RBR (base+0)    /* registro buffer de recepción */
#define THR (base+0)    /* registro de retención de transmisión */

#define DR   1          /* bit dato disponible del LSR */
#define OE   2          /* bit de error de overrun del LSR */
#define PE   4          /* bit de error de paridad del LSR */
#define FE   8          /* bit de error en bits de stop del LSR */
#define BI   0x10       /* bit de error de break en el LSR */
#define THRE 0x20       /* bit de THR vacío */


void error()
{
  printf ("\r    ¡¡Fallo del puerto serie!!\n");
  exit (2);
}


void main()
{
  unsigned com, base, divisor, dato, entrada, lsr;

  printf("\n8250 Test v1.0 - (c) 1993 Ciriaco García de Celis.\n");

  printf("- Elige COM (1, 2, ...):   "); scanf ("%d", &com);
  base=peek(0x40, (com-1)*2);

  if (base==0) {
    printf("\n  ¡El COM elegido no existe para la BIOS!.\n");
    exit (1);
    }

  printf("- Elige divisor (1-65535): ");
  scanf ("%d", &divisor);  if (!divisor) divisor=1;

  printf("\nComprobando 8250 en %03Xh a %lu baudios.\nEspera...",
    base, 1843200L/divisor/16);

  outportb (LCR, 0x83);  /* DLAB=1, 8 bits, 1 stop, sin paridad */
  outportb (IER, 0);
  outportb (DLL, divisor % 256);
  outportb (DLM, divisor >> 8);
  outportb (MCR, 8+16);         /* modo LOOP */
  outportb (LCR, 0x03);  /* DLAB=0, 8 bits, 1 stop, sin paridad */

  for (dato=0; (dato<0x100) && !kbhit(); dato++) {

    do {                                /* esperar por THR vacío */
      lsr=inportb(LSR);
      if (lsr & (OE|PE|FE|BI)) error();
    } while (!(lsr & THRE));

    outportb (THR, dato);               /* enviar carácter */

    do {                                /* esperar por RBR lleno */
      lsr=inportb(LSR);
      if (lsr & (OE|PE|FE|BI)) error();
    } while (!(lsr & DR));

    entrada=inportb (RBR);              /* recibir carácter */

    if (dato!=entrada) error();
    printf ("\rEnviado y recibido byte %d",dato);
    }

  if (!kbhit())
      printf("\rAutodiagnóstico del 8250 en COM%d superado.\n", com);
    else
      { getch(); printf("\rTecla pulsada - prueba abortada.\n");}
}
```

12.10. - EL PUERTO DE LA IMPRESORA.

La impresora se controla desde el DOS referenciándola como dispositivo LPT1 (PRN) ó LPT2. La BIOS utiliza la INT 17h para los servicios de impresora. En ambos casos, el funcionamiento es realmente trivial y la dificultad estriba en el modelo de impresora que se trate (IBM, Epson, HP-III, PostScript, etc.) de cara al lenguaje que soporta. Eso no lo trataremos aquí, ya que todas las impresoras vienen acompañadas de un manual técnico de programación (o en su defecto se puede adquirir opcionalmente). Lo que veremos a continuación son los registros a bajo nivel del puerto paralelo, así como pistas para una utilización algo más allá de la impresora: la comunicación entre ordenadores.

12.10.1. - LOS REGISTROS DEL PUERTO PARALELO.

La dirección base del puerto paralelo en los ordenadores compatibles depende del tipo de adaptador que incorporen. Las primeras máquinas traían un puerto paralelo en el adaptador de vídeo monocromo, cuya dirección base es 3BCh. Sin embargo, otros adaptadores utilizan la dirección base 378h para LPT1 y 278h para LPT2. Por fortuna, la BIOS tiene en el área de datos una tabla con las direcciones base de los 4 posibles puertos paralelos. Dicha tabla comienza en 40h:8 y consta de 1 palabra por puerto (a 0 si ese puerto no existe). La asignación que realizan diversas BIOS puede ser un tanto discutible, pero si el usuario no ve salir los datos por la impresora que desea, siempre puede cambiar los cables o configurar su programa...

Los registros de que consta el puerto paralelo son 3: el primero es el **registro de datos**, de 8 bits, ubicado en la dirección base (3BCh, 378h, 278h, etc.). Este registro es de sólo escritura, para enviar los caracteres a la impresora. El siguiente registro, de sólo lectura, es el **registro de estado**, inmediatamente a continuación del anterior (3BDh, 379h, 279h). Finalmente, tras ellos hay un registro de sólo escritura, el **registro de control** (en 3BEh, 37Ah, 27Ah). Aunque en los tres casos he indicado la dirección, hay que tener en cuenta que lo correcto es consultar la variable de la BIOS y tomarla como punto de partida.

Los registros de estado y control están asociados a unas líneas físicas del puerto paralelo estándar, y poseen un significado concreto que resumimos a continuación. En el valor pin se hace referencia al pin del puerto paralelo del ordenador y al correspondiente en la impresora (ordenador/impresora). Las líneas o pines que no aparecen aquí son las de datos (líneas 2 a la 9, conectadas también con las líneas 2 a la 9 del lado de la impresora; las restantes están a masa).

- **Registro de estado:**
  - Bits 0-2: no utilizados.
  - Bit 3: pin 15/32 (-ERROR). A 0 si hay un error gordo (a revisar los cables).
  - Bit 4: pin 13/13 (SLCT). A 1 si la impresora está ON LINE.
  - Bit 5: pin 12/12 (PE). A 1 si la impresora no tiene papel (PAPER ERROR).
  - Bit 6: pin 10/10 (-ACK). A 0 si la impresora confirma la recepción del carácter.
  - Bit 7: pin 11/11 (-BUSY). A 0 si la impresora está ocupada.
- **Registro de control:**
  - Bit 0: pin 1/1 (-STROBE). A 0 si hay un carácter en el registro de datos.
  - Bit 1: pin 14/14 (-AUTO FEED). A 1 si la impresora debe saltar línea tras cada código 13 (CR).
  - Bit 2: pin 16/31 (-INIT). A 0 para resetear la impresora.
  - Bit 3: pin 17/36 (SLCT IN). A 1 para seleccionar la impresora (0 para OFF-LINE).
  - Bit 4: no conectado al puerto de impresora. A 1 activa la interrupción de la impresora.
  - Bits 5-7: no utilizados.

La posibilidad de emplear interrupciones es realmente interesante: cuando la señal -ACK se pone a nivel 0 (esto es, se activa) viene una IRQ7 ó una IRQ5 (según cómo esté configurada la tarjeta). De todos modos, habrá que mandar primero un carácter por el método tradicional para iniciar la transmisión. La BIOS, sin embargo, no utiliza la interrupción de la impresora.

12.10.2. - ENVÍO DE CARACTERES.

Ante todo dejar claro que cuando digamos 0 ó 1 nos referimos al valor del bit en el registro del PC, olvidando ya cuestiones como el nivel al que son activas las señales, para evitar lios: los nombres de las

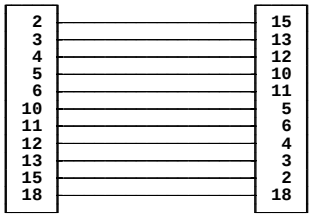
señales les tomaremos como referencia, sin considerar su polaridad. Para enviar un carácter, primero se le coloca en el registro de datos. A continuación se pone a 0 en el registro de control el bit de STROBE. Este bit debe estar muy poco tiempo activo, para evitar que la impresora lea dos veces el mismo carácter (del orden de un microsegundo). Como la impresora no tiene una capacidad de aguante ilimitada, se puede defender poniendo el bit de BUSY en el registro de estado a 0 para poder leer con tranquilidad el STROBE que le llega. Cuando lo haya leído, pondrá un 0 en ACK para indicar que ya ha recibido el carácter.

Este es el esquema básico del envío de caracteres. Sin embargo, hay que tener en cuenta que la impresora puede devolver ciertas condiciones de error, tanto leves (falta de papel) como más graves, como el caso de ERROR. También el ordenador puede provocar ciertos efectos en la impresora, a través del registro de control, como vimos anteriormente. Quizá el más curioso es el del AUTO FEED: ya se podían haber puesto de acuerdo el primer día, resulta triste que además de perder horas configurando impresoras y programas, hasta el propio puerto pueda meter las narices en el control del salto de línea...

12.10.3. - CABLE NULL-MODEM PARA CONECTAR DOS ORDENADORES.

Anteriormente hemos visto una descripción de patillas del puerto paralelo suficiente para que cualquiera se pueda construir su propio cable centronics. De todas formas, estos cables afortunadamente se venden ya construidos por un precio poco aceptable. Los que no se venden, aunque sí acompañan a ciertas aplicaciones software e incluso hardware (como disqueteras externas vía puerto de impresora) permiten una comunicación bidireccional. El truco consiste en utilizar las líneas del registro de estado para recibir datos, aunque esto limita la transferencia a 5 bits (realmente 4, más otro para el protocolo de transferencia).

Se toman dos conectores centronic 25-pin machos. Se unen los pins de la siguiente forma:



El motivo de emplear esta asignación y no otra se debe a que es la ya utilizada por ciertas aplicaciones comerciales, como LAPLINK. Es por razones de compatibilidad, para que no pase como con los saltos de línea. La línea común (18) es masa, aunque valdría cualquier patilla entre la 18 y la 25; si se emplea un cable de 10 hilos más malla, esta última es la más adecuada para hacer de masa.

Con este cable, para enviar datos se utilizan las líneas D0 a D4 del registro de datos y para recibirlos las 5 líneas útiles del registro de estado. Como D0-D1-D2-D3-D4 están conectados en este mismo orden a ERROR-SLCT-PE-ACK-BUSY, lo ideal es utilizar D0-D3 para transmitir datos y ERROR-SLCT-PE-ACK para recibirlos. Las señales BUSY y D4 sirven para establecer el protocolo de transmisión. La transferencia puede ser bidireccional y además de forma simultánea. En realidad, cuando se mande un dato y el ordenador remoto indique con BUSY que ya lo tiene (a través de su línea D4), de paso nos puede haber reenviado el dato en D0-D3 para que veamos si es correcto: un control de errores bastante fiable y rápido. Sin embargo, se podría aprovechar quizá para enviar otro medio byte en sentido contrario en el caso de que las dos máquinas se estén pasando información simultáneamente la una a la otra; el control de errores ya se haría de otra manera, a nivel de bloques con checksum, etc. Conviene aprovechar y mandar otros 4 bits de datos cada vez que se envía un reconocimiento (al informar al receptor de que ya se ha recibido su señal de "dato recibido"), lo que permite transferir un byte completo en cada ciclo del protocolo de transferencia. Ah, no hay que olvidar la polaridad de las líneas: al poner un 0 en D4 aparece un 1 en el -BUSY del otro extremo...

Si el cable no rebasa los 3 metros o poco más la transmisión será fiable, y además bastante rápida: 4 bits en paralelo, a la velocidad que pueda alcanzar la CPU del ordenador más lento. No emplear el ensamblador sería un acto imperdonable.

12.11. - EL RATÓN.

El ratón se controla normalmente a través de llamadas a la INT 33h. Existen toda suerte de funciones para controlar su posición, el estado de los botones, el puntero que se visualiza... todas ellas son bastante intuitivas y aptas para un programador en lenguajes de alto nivel. Aquí estudiaremos, sin embargo, el funcionamiento a bajo nivel del ratón. En concreto, del ratón de Microsoft, el más extendido y con el que son compatibles casi todos los demás (aunque sea accionando el correspondiente conmutador).

La mayoría de los ratones se conectan vía puerto serie a 1200 baudios, 7 bits y sin paridad. Para detectar la presencia del ratón, hay que poner la línea DTR del puerto serie a 1. Al cabo de un rato, el ratón devuelve el código ASCII de la letra M (¿será por lo de Mouse o por Microsoft?). Los controladores de Microsoft son un poco estrictos en esta comprobación, y si el ratón no responde en unos márgenes de tiempo muy concretos consideran que no existe, de ahí que en ocasiones haya que emplear otro controlador un poco más flexible.

Llegados a este punto, el funcionamiento se establece a partir de interrupciones de puerto serie. Se trasmiten 3 bytes cada vez que hay un envío: en ellos se indica cuánto se ha movido el ratón en los ejes X e Y desde la última vez, así como el estado de los botones. La unidad de medida, cómo no, son los *Mickeys*, que según la resolución del aparato serán 1/200 ó 1/400 pulgadas.

Los desplazamientos se toman en complemento a dos; como hay 8 bits por cada eje, el movimiento puede oscilar en el rango +128 a -127. Hay además un bit por cada botón. De los 7 bits recibidos en cada interrupción, el más significativo (bit 6) está a 1 en el primer envío y a 0 en los restantes, con objeto de evitar malas interpretaciones de la secuencia si se pierde alguna interrupción por cualquier motivo. El formato empleado para codificar la información es el siguiente:

1	L	R	Y7	Y6	X7	X6

0	X5	X4	X3	X2	X1	X0

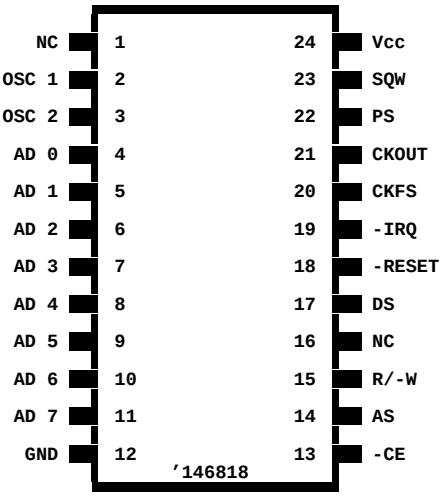
0	Y5	Y4	Y3	Y2	Y1	Y0

El otro gran estándar de ratón, el Mouse Systems, permite trabajar hasta con tres botones. Estos ratones envían (cuando están en modo Mouse) 5 bytes por cada evento. En el primero hay información sobre el estado de los botones; los 4 siguientes parecen contener el desplazamiento relativo en los ejes X e Y. El funcionamiento es, por tanto, similar, y al parecer quizá todavía con 7 bits. Curiosamente, al conmutar el selector de modo (Microsoft-Mouse) aparece una secuencia de bytes un tanto especial, distinta según el sentido de la conmutación, para ayudar al controlador de ratón a detectar el paso al nuevo protocolo con objeto de poder adaptarse al mismo.

12.12. - EL RELOJ DE TIEMPO REAL DEL AT: MOTOROLA MC146818.

12.12.1. - DESCRIPCIÓN DEL INTEGRADO.

El MC146818 incorpora un completo reloj con alarma, calendario, interrupción periódica programable, generador de onda cuadrada y 64 bytes libres de RAM estática de bajo consumo. Los primeros 10 bytes de esta RAM son empleados para gestionar la fecha y la hora y los 4 siguientes son registros (A, B, C y D); los 50 restantes quedan a disposición del usuario.



La línea OSC1 (de entrada) puede conectarse a señales cuadradas de 4.194304 Mhz, 1.048576 Mhz y 32768 Hz. La frecuencia de esta base de tiempos, como se verá, ha de indicarse en el registro A (bits DV0 a DV2). El chip provee una útil salida de reloj en CKOUT dependiente del nivel de la entrada CKFS, según la siguiente tabla:

Señal en OSC1	Nivel de CKFS	Señal en CKOUT
4,194304 Mhz	1	4,194304 Mhz
4,194304 Mhz	0	1,048576 Mhz
1,048576 Mhz	1	1,048576 Mhz
1,048576 Mhz	0	262,144 KHz
32,768 Khz	1	32,768 Khz
32,768 Khz	0	8,192 Khz

La salida SQW genera una onda cuadrada, cuya frecuencia es programable (útil para alarmas). La línea -IRQ se encarga de solicitar las interrupciones periódicas si están habilitadas. La línea de entrada -RESET reinicializa el integrado asignando valores por defecto a ciertos bits de los registros B y C, aunque no afecta a la fecha/hora ni a la memoria. La entrada PS debe mantenerse a nivel bajo cuando se alimenta el chip hasta que la tensión se estabilice, poniéndose después en alto; esta entrada está asociada al bit VRT del registro D que indica si el integrado está en condiciones de operar. El bus bidireccional de direcciones y datos está multiplexado (líneas AD0..AD7): en los flancos de bajada de la entrada de validación de direcciones (línea AS) contiene direcciones, y datos en los flancos de subida de la entrada de validación de datos (línea DS). La línea -R/-W indica si la operación es de entrada o salida; -CE permite habilitar el chip o desconectarlo de los buses.

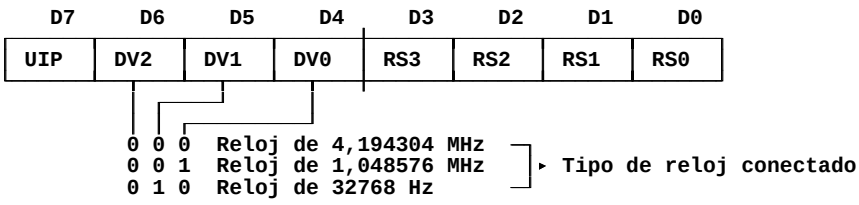
El cuadro de la derecha refleja la estructura de la memoria del MC146818. Los primeros 14 bytes son empleados para la fecha y hora.

00	Segundos
01	Segundos Alarma
02	Minutos
03	Minutos alarma
04	Horas
05	Horas alarma
06	Dia de la semana
07	Dia del mes
08	Mes
09	Año
0A	Registro A
0B	Registro B
0C	Registro C
0D	Registro D
0E..3F	50 bytes libres

REGISTROS DEL MC146818

REGISTRO A (lectura/escritura, excepto UIP).

Este registro sirve para indicar al integrado qué tipo de reloj lo gobierna, así como elegir la frecuencia de la interrupción periódica programable y la de la salida SQW. También contiene un bit que indica si hay una actualización del reloj en curso, lo que sucede una vez cada segundo, ya que en ese preciso instante no se pueden leer los registros con objeto de evitar lecturas incorrectas.



El bit **UIP** (Update In Progress), de sólo lectura, se pone a 1 mientras se actualizan los primeros 14 bytes de la memoria y poco tiempo antes de que comience dicha actualización. Antes de acceder a estos bytes, hay que esperar a que el bit UIP se ponga a cero (si no lo estaba ya): con el bit UIP a 0, es seguro que en un intervalo de al menos 244 microsegundos no se va a producir ninguna actualización, por lo que hay tiempo suficiente para acceder (sin prisas, pero tampoco con pausas). La actualización dura 248 microsegundos (1984 con relojes de 32768 Hz).

Los bits **RS0**..**RS3**, de selección de velocidad, definen la frecuencia de la onda cuadrada generada en SQW y/o la de la interrupción periódica, como indica esta tabla:

				Reloj 1,048576 ó 4,194304 Mhz		Reloj de 32768 Hz	
RS3	RS2	RS1	RS0	Velocidad INT	Frecuencia SQW	Velocidad INT	Frecuencia SQW
0	0	0	0	(no actúa)	(nula)	(no actúa)	(nula)
0	0	0	1	30,517 µs	32768 Hz	3,90625 ms	256 Hz
0	0	1	0	61,035 µs	16384 Hz	7,81250 ms	128 Hz
0	0	1	1	122,070 µs	8192 Hz	122,070 µs	8192 Hz
0	1	0	0	244,141 µs	4096 Hz	244,141 µs	4096 Hz
0	1	0	1	488,281 µs	2048 Hz	488,281 µs	2048 Hz
0	1	1	0	976,562 µs	1024 Hz	976,562 µs	1024 Hz
0	1	1	1	1,953125 ms	512 Hz	1,953125 ms	512 Hz
1	0	0	0	3,90625 ms	256 Hz	3,90625 ms	256 Hz
1	0	0	1	7,8125 ms	128 Hz	7,8125 ms	128 Hz
1	0	1	0	15,625 ms	64 Hz	15,625 ms	64 Hz
1	0	1	1	31,25 ms	32 Hz	31,25 ms	32 Hz
1	1	0	0	62,5 ms	16 Hz	62,5 ms	16 Hz
1	1	0	1	125 ms	8 Hz	125 ms	8 Hz
1	1	1	0	250 ms	4 Hz	250 ms	4 Hz
1	1	1	1	500 ms	2 Hz	500 ms	2 Hz

REGISTRO B (lectura/escritura).

En este registro hay bits útiles, entre otros, para controlar la inicialización de la fecha y hora, para habilitar o inhibir las diversas interrupciones y para establecer ciertas características de operación.

D7	D6	D5	D4	D3	D2	D1	D0
SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE

El bit **SET** puede ser establecido a 1, con lo que cualquier ciclo de actualización de los primeros 14 bytes de la RAM resulta abortado: de este modo, es factible proceder a inicializar la fecha y la hora sin el riesgo de que se produzca en medio una actualización. Este bit no se ve afectado por la señal -RESET.

El bit **PIE** (Periodic Interrupt Enable) sirve para permitir la interrupción periódica cuando es puesto a 1; tras una señal -RESET es puesto a 0. El bit **AIE** (Alarm Interrupt Enable) ha de estar a 1 para habilitar la interrupción de alarma; también es puesto a cero tras un -RESET. El bit **UIE** (Update Interrupt Enable) sirve para habilitar o inhibir la interrupción de fin de actualización, que se produciría tras cada actualización del reloj; la señal -RESET baja el bit UIE. Por último, el bit **SQWE** (Square Wave Enable) permite habilitar o inhibir la señal de onda cuadrada de la salida SQW; también es borrado ante una señal -RESET.

El bit **DM** (Data Mode) permite seleccionar datos en binario (1) o BCD (0) en los bytes de fecha y hora; la señal -RESET no afecta a este bit. El bit **24/12** sirve para elegir entre el modo 12 horas del reloj (bit a 0) o el de 24 (bit a 1): en el modo de 12 horas, el bit más significativo del byte de la hora estará activo para indicar "PM". Si bit **DSE** está activo, el último domingo de abril la hora pasa de 1:59:59 AM a 3:00:00 AM; en el último domingo de octubre pasa de 1:59:59 AM a 1:00:00 AM (sólo la primera vez, claro) para ajustarse al cambio de hora oficial; este bit no es afectado por -RESET.

REGISTRO C (sólo lectura).

Este registro contiene bits que informan de las interrupciones que se producen. Permite identificar al ordenador qué o cuáles interrupción(es) se ha(n) producido.

D7	D6	D5	D4	D3	D2	D1	D0
IRQF	PF	AF	UF	0	0	0	0

El bit **IRQF** (Interrupt ReQuest Flag) se activa cuando el bit PF y el PIE (registro B) están activos, o bien cuando el bit AF y el AIE (registro B) están activos, o bien cuando UF y el bit UIE (registro B) están activos. Es decir, IRQF se pone en alto cuando es necesario que se produzca una interrupción: la línea -IRQ se encarga de pedirla entonces. Por su parte: **PF** (Periodic Flag), **AF** (Alarm Flag) y **UF** (Update Flag) indican si es necesario que se produzca la interrupción correspondiente. Todos los bits de este registro son borrados ante una señal -RESET, pero también ante una lectura por software del registro C.

REGISTRO D (sólo lectura).

Este registro contiene sólo el bit **VRT** (Valid RAM and Time). Este bit está a cero cuando la patilla PS está a cero (PS se eleva a 1 cuando la tensión de alimentación es correcta). Por software, el bit VRT puede ser puesto a 1 mediante una simple lectura del registro D (si la patilla PS=1), con objeto de indicar que la fecha y hora establecidas son correctas; si fallara la alimentación, al caer la tensión en la patilla PS este bit pasaría de nuevo a cero. VRT no es afectado por -RESET.

D7	D6	D5	D4	D3	D2	D1	D0
VRT	0	0	0	0	0	0	0

FUNCIONAMIENTO DE LA ALARMA

La interrupción de alarma se produce todos los días cuando llega la hora en que ha sido programada y el bit que permite esta interrupción está habilitado. Existe un método alternativo para programar la alarma, basado en los *códigos indiferentes* almacenables en los bytes de la alarma. Un código indiferente es cualquier valor comprendido entre 0C0h y 0FFh. Si la hora de alarma es un código indiferente, la alarma se producirá cada hora. Si la hora y minuto de alarma son códigos indiferentes, ésta se producirá cada minuto. Si tanto la hora como el minuto y segundo de la alarma son códigos indiferentes, la alarma se producirá cada segundo.

12.12.2. - EL MC146818 DENTRO DEL ORDENADOR.

El MC146818 es por lo general exclusivo de los AT y PS/2. En muchos ordenadores, la implementación física se realiza con circuitos totalmente compatibles que incluyen 128 bytes de RAM en lugar de 64. En la RAM que sobra por encima de los primeros 14 bytes se almacenan parámetros de la configuración del sistema, modificables con el programa SETUP durante el arranque.

Por defecto, la BIOS inicializa el chip para trabajar con un reloj de 32768 Hz y a un ritmo de 1024 interrupciones periódicas por segundo (cuando están habilitadas), al escribir el valor 26h en el registro A. De la misma manera, el registro B se carga con 2 (modo 24 horas, datos en BCD y sin horario verano/invierno).

El MC146818 está diseñado para ser conectado a un bus multiplexado, por lo que la circuitería de apoyo de los AT se encarga de gestionar la comunicación con el microprocesador, estableciendo dos puertos de entrada/salida en las direcciones 70h y 71h. Para leer o escribir cualquier registro de la RAM CMOS, basta con enviar al puerto 70h el número de registro y, a continuación, leer o escribir del puerto 71h. Entre los accesos a ambos puertos debe mediar un tiempo mínimo; de lo contrario la operación fallará. En particular, las últimas versiones de los compiladores de Borland no permiten acceder al reloj de tiempo real en la mayoría de las máquinas a través de las funciones *outportb()* e *inportb()*. La razón es que esas funciones están en una librería y es preciso llamarlas con paso de parámetros a través de la pila, lo que ralentiza excesivamente el proceso. Desde el lenguaje ensamblador, nunca hay problemas, aunque como es costumbre

es conveniente insertar algún estado de espera (JMP SHORT $+2) entre dos operaciones E/S consecutivas, precaución necesaria en los ordenadores más antiguos.

A nivel de interrupciones, la salida -IRQ del MC146818 está conectada a IRQ8 (INT 70h) a través del segundo controlador de interrupciones (véase la documentación del mismo).

Desde la interrupción 1Ah, la BIOS implementa una serie de servicios para acceder al reloj de tiempo real, incluyendo la posibilidad de programar la alarma (que invoque una INT 4Ah cuando llegue la hora). Las funciones de retardo de la INT 15h se apoyan también en el reloj de tiempo real.

Conviene tener presente que es de vital importancia acceder a los primeros 14 bytes de la CMOS sólo si el bit UIP del registro A (bit 7) está a cero. También es necesario poner a 1 el bit SET del registro B (bit 7) antes de modificar dichos bytes, devolviéndolo a 1 después. No respetar este principio puede provocar la lectura de fechas u horas incorrectas o una errónea asignación de valores. Para los demás bytes de la CMOS no es necesario tomar esta precaución.

12.12.3. - UN MÉTODO PARA AVERIGUAR LA CONFIGURACIÓN DEL AT Y PS/2.

Como se dijo antes, los AT y superiores almacenan en los 50 ó 114 últimos bytes de RAM libres de la CMOS información relativa a la configuración del sistema. Los bytes más importantes y comunes a todas las máquinas se muestran a continuación.

Byte 0Eh:	**Diagnostics Status Byte**. El bit 7 indica (si vale 1) que el MC146818 tiene un déficit de corriente eléctrica. El bit 6 indica (si es 1) que el chechsum o suma de comprobación de la CMOS ha fallado. El bit 5 indica (si vale 1) que la configuración del sistema es incorrecta (no hay al menos una disquetera presente o el modo de vídeo de la configuración no coincide con el detectado en el hardware). El bit 4 es puesto a 1 si el tamaño de la memoria detectado no coincide con el indicado en la configuración. El bit 3 activo indica que el adaptador o el disco fijo C: falló en la inicialización, siendo imposible botar desde él. El bit 2 activo indica que la hora del reloj es incorrecta. Los bits 1 y 0 están reservados.
Byte 0Fh:	**Shutdown Status Byte**. Los bits de este byte son asignados durante la inicialización del sistema por parte de la BIOS, informando de su desarrollo (véase listado de la BIOS).
Byte 10h:	**Diskette Drive Type Byte**. Los bits 7..4 indican el tipo de la disquetera A y los bits 3..0 el tipo de la disquetera B. Los valores posibles son 0 (no existe esa disquetera), 1 (5¼-360K), 2 (5¼-1.2M), 3 (3½-720K), 4 (3½-1.44M) y 5 (3½-2.88M en BIOS AMI) ó 6 (3½-2.88M en BIOS IBM).
Byte 11h:	**Reservado**.
Byte 12h:	**Fixed Disk Type Byte**. Los bits 7..4 indican el tipo del primer disco fijo y los bits 3..0 el tipo del segundo. Existe una tabla definida por IBM cuando lanzó el AT con 14 tipos de disco; ninguno que se vende hoy en dia está en la tabla, por lo que es frecuente que estos campos estén inicializados con el valor 1111b (ó 0 si no hay disco duro instalado) para indicar simplemente la presencia de disco duro.
Byte 13h:	**Reservado**.
Byte 14h:	**Equipment Byte**. Los bits 7 y 6 indican el número de disquetes instalados; los bits 5 y 4 el tipo de adaptador de vídeo primario (00: EGA/VGA, 01: CGA-80, 10: CGA-40, 11: MDA); los bits 3 y 2 no se emplean. El bit 1 indica si hay coprocesador aritmético y el bit 0 está activo para confirmar que hay disqueteras.
Byte 15h-16h:	**Low and High Base Memory Bytes**. El 15h es el bajo y el 16h el alto. Entre ambos forman una palabra de 16 bits que indica la cantidad de memoria convencional (típicamente 640 Kb).
Byte 17h-18h:	**Low and High Memory Expansion Bytes**. El 17h es el bajo y el 18h el alto. Entre ambos forman una palabra de 16 bits que indica la cantidad de memoria extendida, en Kbytes.
Byte 19h:	**Número del primer disco duro**. Número de identificación que la BIOS asigna al primer disco duro instalado.
Byte 1Ah-2Dh:	**Reservados**.
Byte 2Eh-2Fh:	**Checksum**. El 2Eh es el alto y el 2Fh el bajo. Entre ambos forman una palabra de 16 bytes que constituye el checksum o suma de comprobación de los bytes 10h-20h.
Byte 30h-31h:	**Low and High Memory Expansion Bytes**. Habitualmente es el mismo valor que el almacenado en los bytes 17h y 18h; esta variable refleja sólo la memoria extendida ubicada por encima del primer megabyte que detecta la BIOS en el momento de arrancar.
Byte 32h:	**Date Century Byte**. Valor BCD del siglo actual-1. Para 1992, por ejemplo, es 19h.
Byte 33h:	**Information Flag**. El bit 7 indica si está instalada la vieja opción de ampliación de 128 Kb (hasta los 640 Kb) del IBM AT original: hoy en día suele estar siempre activo. El bit 6 es empleado por el programa SETUP para eliminar el mensaje inicial al usuario tras el primer SETUP. Los demás bits están reservados.
Byte 34h-3Fh:	**Reservados**.