11.Последовательный интерфейс RS-232

11.1. Общие сведения

В сотав IBM PC могут входить до четырех последовательных интерфейсов, работающих в стандарте RS-232 (отечественный аналог - стык С2) и именуемых COM1 - COM4. Им выделены следующие адреса в области портов ввода-вывода:

COM1: 3F8h-3FFh COM3: 338h-33Fh
COM2: 278h-2FFh COM4: 238h-23Fh

(интерфейсы COM3 и COM4 поддерживаются только в моделях PS/2).

Каждый интерфейс связан с определенным уровнем контроллера прерываний:

COM1 вызывает прерывание IRQ4 (Int 0Ch)
COM2 вызывает прерывание IRQ3 (Int 0Bh)
COM3 и COM4 не имеют стандартных векторов прерываний.

Каждое из устройств RS-232 представляет собой контроллер 8250, оснащенный 25- или 9- штырьковым разъемом на задней стенке корпуса ПЭВМ. Этот разъем может использоваться для подключения мыши, графопостроителя или организации связи между ПЭВМ. Контакты стыка RS-232 имеют следующие наименования:

Название сигнала	Имя цепи		Номер контакта		Назначение	Направление
	EIA	CCITT	9-шт.	25-шт.		
DCD	CF	109	1	8	Связь модемов установлена	В комп.
RX	BB	104	2	3	Приём данных	В комп.
TX	BA	103	3	2	Передача данных	Из комп.
DTR	CD	108/2	4	20	Готовность комп.к работе	Из комп.
SG	AB	102	5	7	Сигнальная земля	
DSR	CC	107	6	6	Готовность модема к работе	В комп.
RTS	CA	105	7	4	Запрос на передачу	Из комп.
CTS	CB	106	8	5	Готовность модема к передаче	В комп.
RI	CE	125	9	22	Индикатор вызова	В комп.
FG	AA	101		1	Защитная земля	

Контроллер стыка RS-232 является полностью программируемым

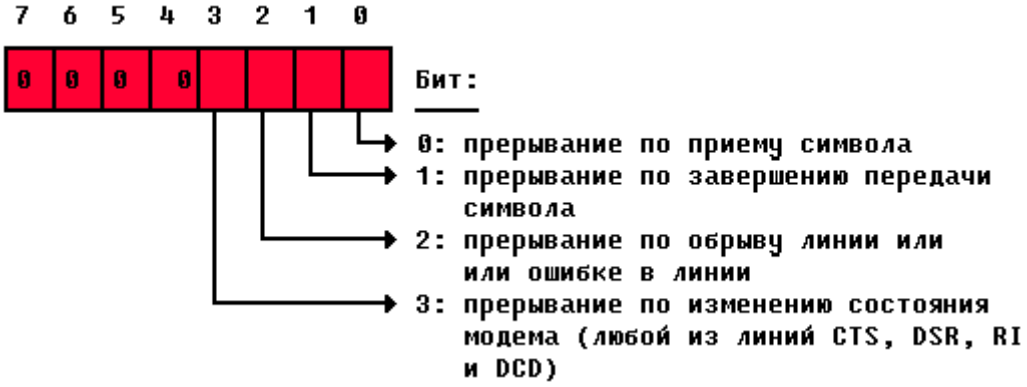
устройством; вы можете задать следующие параметры обмена: количество битов данных и стоп-битов, вид четности и скорость обмена в бодах (бит/с).

11.2. Описание портов

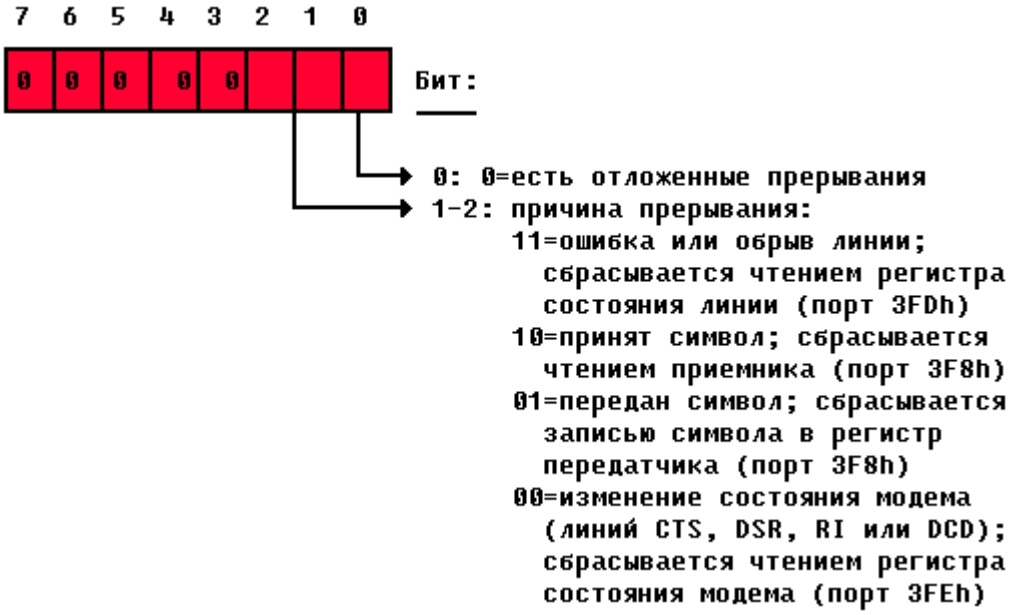
Ниже описаны порты ввода-вывода для COM1, имеющего базовый адрес 3F8h. Обратите внимание, что порты 3F8h и 3F9h имеют

разное назначение в зависимости от бита 7 порта 3F8h (т. н. бит DLAB - Divisor Latch Access Bit).

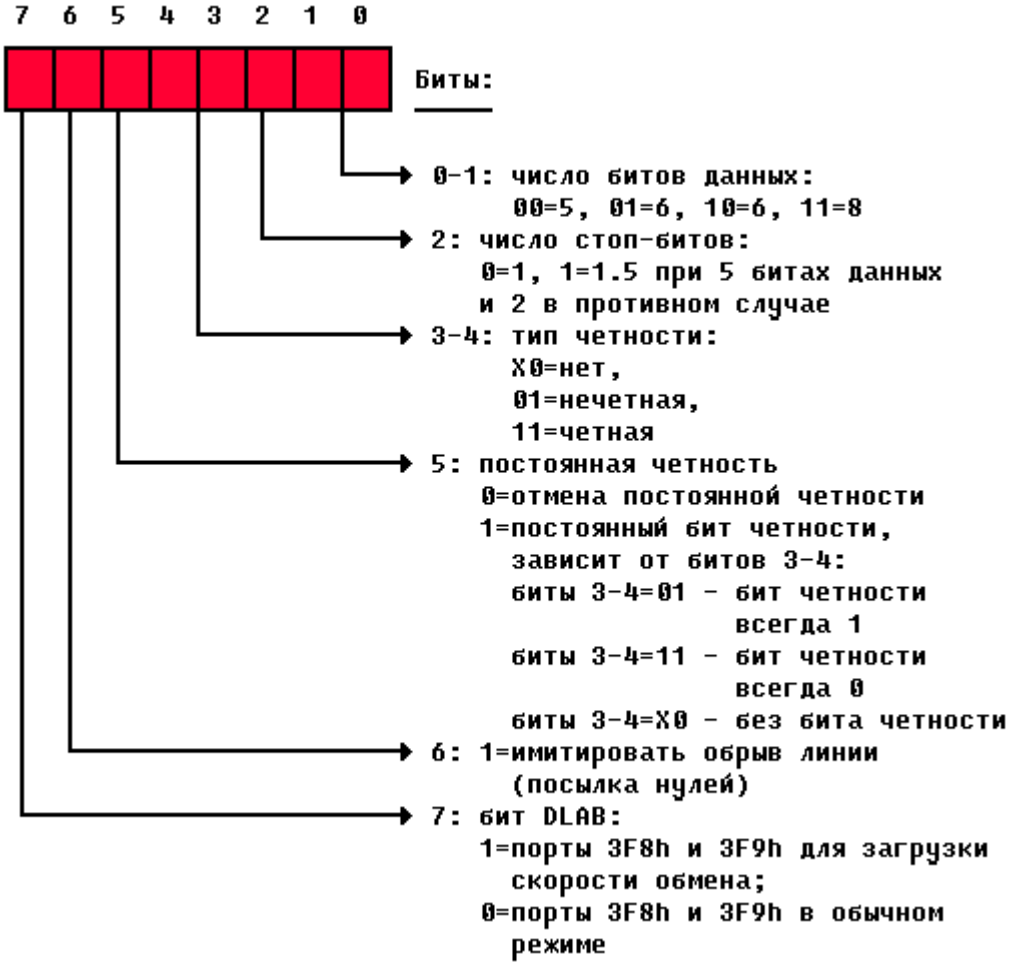
Порт	Операция	Описание
3F8h	Запись	Регистр передатчика - сюда засылается байт для передачи
	Чтение	Регистр приемника - отсюда извлекается принятый байт
	Запись	Если DLAB=1, то сюда засылается младший байт скорости обмена (см. порт 3F9h)
3F9h	Запись	Если DLAB=1, то сюда засылается старший байт скорости обмена. Скорость задается значением делителя, равным 115200/V, где V - скорость в бодах. Например, для скорости 9600 бод делитель равен 115200/9600=12=0Ch, поэтому нужно вывести 0Ch в порт 3F8h и 0 в порт 3F9h.
	Запись	Регистр управления прерываниями (1 = разрешить прерывание):



3FAh	Чтение	Регистр идентификации прерывания. Когда произошло прерывание, здесь содержится причина, вызвавшая его:



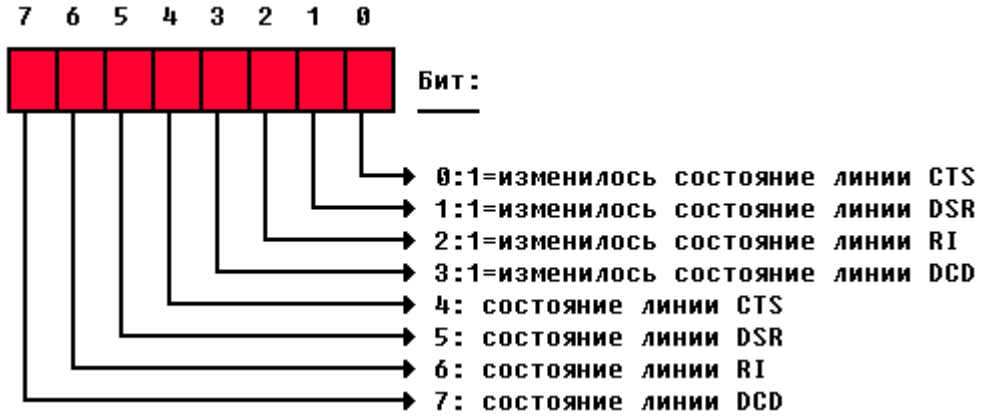
3FBh Чтение/ Регистр управления линией:
Запись



3FCh Запись Регистр управления модемом

3FDh Чтение Регистр состояния линии. Биты 1-4 вызывают прерывания по ошибке, если оно разрешено:

3FEh Чтение Регистр состояния модема. Биты 0-3 вызывают прерывание по изменению состояния модема, если оно разрешено:



Примечание: "Изменилось состояние линии..." означает, что данная линия стыка RS-232 изменила свое состояние по сравнению с последним чтением этого регистра.

11.3. Порядок инициализации 8250

Для подготовки контроллера 8250 к работе необходимо выполнить следующие шаги.

1. Установить бит DLAB порта 3FBh и заслать делитель, задающий скорость обмена, в порты 3F8h и 3F9h.

2. Инициализировать регистр управления линией (порт 3FBh); при этом сбросить бит DLAB.

3. Инициализировать регистр управления модемом (порт 3FCh).

4. Инициализировать регистр управления прерываниями (порт 3F9h) и, если прерывания разрешены, установить адрес программы обработки прерываний от стыка RS-232.

В качестве примера приведем набор подпрограмм, обеспечивающих обмен через порт COM1 в дуплексном режиме со скоростью 1200 бод.

```
          title     RS232
_DATA     segment   para public
Buf_Size  equ  1024           ; размер буфера

Source    db   Buf_Size+2 dup (0) ; буфер приема символов
Src_ptr   dw   Source         ; указатель позиции в буфере
Count     dw   0              ; количество символов в буфере
Ser_ip    dw   0              ; старый адрес Int 0Ch
Ser_cs    dw   0
Save_ds   dw   0              ; служебные переменные
Int_sts   db   0
Overrun   db   0
_DATA     ends

_TEXT     segment   para public
          assume    cs:_TEXT, ds:_DATA
          public    Ser_Ini:near, Ser_Rst:near
          public    Get_Chr:near, Out_Chr:near
;+-------------------------------------------------------------+
;|        Подпрограмма инициализации стыка COM1.               |
;+-------------------------------------------------------------+
Ser_Ini   proc near
          push ax        ; сохранить регистры
          push dx
          push bx
          push es
          in   al,21h    ; IMR 1-го контролера прерываний
          or   al,10h    ; запретить прерывание IRQ4 от COM1
          out  21h,al
          mov  al,0Ch
          mov  ah,35h
          int  21h       ; взять вектор Int 0Ch в es:bx
          mov  Ser_ip,bx ; и сохранить его
          mov  Ser_cs,es
          mov  al,0Ch
          mov  dx,offset Ser_int
          push ds
          mov  bx,cs
          mov  ds,bx
          mov  ah,25h
          int  21h       ; установить Int 0Ch = ds:dx
          pop  ds
          pop  es
          pop  bx
          cli            ; запретить прерывания
          in   al,21h    ; IMR 1-го контроллера прерываний
          and  al,not 10h
          out  21h,al    ; разрешить прерывания от COM1
          mov  dx,3FBh   ; регистр управления линией
          in   al,dx
          or   al,80h    ; установить бит DLAB
          out  dx,al
          mov  dx,3F8h
          mov  al,60h
```

```
          out  dx,al     ; младший байт для скорости 1200 бод
          inc  dx
          mov  al,0
          out  dx,al     ; старший байт скорости
          mov  dx,3FBh   ; регистр управления линией
          mov  al,00000011b ; 8 бит, 2 стоп-бита, без четности
          out  dx,al
          mov  dx,3F9h   ; регистр разрешения прерываний
          mov  al,1      ; разрешить прерывания по приему
          out  dx,al
          nop            ; и чуть-чуть подождать
          nop
          mov  dx,3FCh   ; регистр управления модемом
          mov  al,00001011b ; установить DTR, RTS и OUT2
          out  dx,al
          sti            ; разрешить прерывания
          mov  dx,3F8h   ; регистр данных
          in   al,dx     ; сбросить буфер приема
          pop  dx
          pop  ax
          ret
Ser_Ini   endp
;+-------------------------------------------------------------+
;|       Подпрограмма отключения стыка COM1.                   |
;+-------------------------------------------------------------+
Ser_Rst   proc near
          push ax        ; сохранить регистры
          push dx
Wait_Free:
          mov  dx,3FDh   ; регистр состояния линии
          in   al,dx
          jmp  short $+2 ; короткая задержка
          test al,60h    ; передача окончена?
          jz   Wait_Free ; ждем, если нет
          mov  dx,3F9h   ; регистр разрешения прерываний
          mov  al,0      ; запретить прерывания
          out  dx,al
          jmp  short $+2 ; еще подождем...
          jmp  short $+2
          mov  dx,3FCh   ; регистр управления модемом
          mov  al,00000011b ; активировать DTR и RTS
          out  dx,al
          jmp  short $+2
          jmp  short $+2
          push bx
          mov  al,0Ch
          mov  dx,Ser_ip
          push ds
          mov  bx,Ser_cs
          mov  ds,bx
          mov  ah,25h
          int  21h       ; восстановить вектор Int 0Ch
          pop  ds
          pop  bx
          cli            ; запрет прерываний
          in   al,21h    ; читать маску прерываний
          jmp  short $+2
          or   al,10h    ; запретить IRQ4
```

```
          out  21h,al
          sti            ; разрешение прерываний
          pop  dx
          pop  ax
          ret
Ser_Rst   endp
;+-------------------------------------------------------------+
;|        Подпрограмма обработки прерываний от COM1.           |
;+-------------------------------------------------------------+
Ser_Int   proc far
          push ax
          push dx
          push ds
          mov  ax,seg _DATA
          mov  ds,ax
          mov  dx,3FAh   ; регистр идентификации прерываний
          in   al,dx
          mov  Int_Sts,al; сохраним его содержимое
          test al,1      ; есть отложенные прерывания?
          jz   Is_Int    ; да
          pop  Save_ds   ; нет, передаем управление
          pop  dx        ; старому обработчику Int 0Ch
          pop  ax
          push Ser_cs
          push Ser_ip
          push Save_ds
          pop  ds
          ret            ; длинный переход
Is_Int:
          mov  al,64h    ; послать EOI для IRQ4
          out  20h,al    ; в 1-й контроллер прерываний
          test Int_Sts,4 ; прерывание по приему?
          jnz  Read_Char ; да
No_Char:
          sti            ; нет, разрешить прерывания
          jmp  Int_Ret   ; и закончить обработку Int 0Ch
Read_Char:
          mov  dx,3FDh   ; регистр состояния линии
          in   al,dx
          and  al,2
          mov  Overrun,al; ovvrrun<>0, если была потеря символа
          mov  dx,3F8h   ; регистр данных
          in   al,dx     ; вводим символ
          or   al,al     ; если принят нуль,
          jz   No_Char   ; то игнорируем его
          push bx
          mov  ah,Overrun
          or   ah,ah     ; предыдущий символ потерян?
          jz   Save_Char ; нет
          mov  ah,al     ; да,
          mov  al,7      ; заменяем его на звонок (07h)
Save_Char:
          mov  bx,Src_ptr; заносим символ в буфер
          mov  [bx],al
          inc  Src_ptr   ; и обновляем счетчики
          inc  bx
          cmp  bx,offset Src_ptr-2 ; если конец буфера
          jb   Ser_Int_1
```

```
          mov  Src_ptr,offset Source ; то "зацикливаем" на начало
Ser_Int_1:
          cmp  Count,Buf_Size ; буфер полон?
          jae  Ser_Int_2 ; да
          inc  Count     ; нет, учесть символ
Ser_Int_2:
          or   ah,ah     ; если была потеря символа
          jz   Ser_Int_3
          mov  al,ah     ; то занести в буфер сам символ
          xor  ah,ah
          jmp  short Save_Char
Ser_Int_3:
          pop  bx
          sti            ; разрешить прерывания
Int_Ret:
          pop  ds
          pop  dx
          pop  ax
          iret
Ser_Int   endp
;+-------------------------------------------------------------+
;| Подпрограмма вывода символа AL в порт.                      |
;| При ошибке возвращает CF=1, иначе CF=0.                     |
;+-------------------------------------------------------------+
Out_Chr   proc near
          push ax
          push cx
          push dx
          mov  ah,al
          sub  cx,cx
Wait_Line:
          mov  dx,3FDh   ; регистр состояния линии
          in   al,dx
          test al,20h    ; стык готов к передаче?
          jnz  Output    ; да
          jmp  short $+2
          jmp  short $+2
          loop Wait_Line ; нет, ждем
          pop  dx
          pop  cx
          pop  ax
          stc            ; нет готовности порта
          ret
Output:
          mov  al,ah
          mov  dx,3F8h   ; регистр данных
          jmp  short $+2
          out  dx,al     ; вывести символ
          pop  dx
          pop  cx
          pop  ax
          clc            ; нормальный возврат
          ret
Out_Chr   endp
;+-------------------------------------------------------------+
;| Подпрограмма ввода символа из порта в AL.                   |
;| Если буфер пуст, возвращает CF=1, иначе CF=0.               |
;+-------------------------------------------------------------+
```

```
Get_Chr   proc near
          cmp  Count,0    ; буфер пуст?
          jne  loc_1729   ; нет
          stc             ; да, возврат по ошибке
          ret
loc_1729:
          push si
          cli             ; запретим прерывания
          mov  si,Src_ptr
          sub  si,Count
          cmp  si,offset Source
          jae  loc_1730
          add  si,Buf_Size
loc_1730:
          mov  al,[si]    ; выберем символ
          dec  Count      ; и уменьшим счечик
          sti             ; разрешение прерываний
          pop  si
          clc             ; и нормальный возврат
          ret
Get_Chr   endp
_TEXT     ends
          end
```

Взято с:http://www.ournet.md/~nihona
E-mail :nihona@mail.md