Interface RS-232 para AMSTRAD

Diseñado y fabricado por MHT ingenieros

INDICE

1. Información general

El interface RS-232-C se utiliza en aplicaciones en las que sea necesario enviar y/o recibir datos en forma serie. Aplicaciones típicas son, por ejemplo, el manejo de impresoras y plotters con interface serie, comunicaciones con otros ordenadores, transmisión a través de modems, etc.

Este interface se conecta de forma simple al bus de expansión de cualquiera de los ordenadores AMSTRAD (CPC 464, CPC 664 o CPC 6128). Prolongando este por la parte posterior de forma que se puede conectar a continuación cualquier otro periférico. El software incluido es válido también en cualquiera de estos ordenadores.

La conexión serie se realiza a través de un conector estándar de 25 pines. La disposición de las señales dentro del conector es así mismo estándar, por lo que no hay dificultades para su conexión con cualquier otro equipo (incluso de marcas diferentes).

2. Introducción

Atención: Nunca debe enchufar ni desenchufar el interface del bus de expansión sin antes desconectar la alimentación del ordenador.

Una vez conectada la alimentación del ordenador aparecerá en pantalla un mensaje de reconocimiento del periférico que indicará que éste se ha inicializado correctamente.

En este momento el interface se encuentra configurado de la siguiente manera:

Velocidad de transmisión	9.600 baudios
Velocidad de recepción	9.600 baudios
Control de paridad	ninguno.
N° de bits de stop	1 bit
Longitud de palabra	8 bits.

Si esta configuración es distinta de la que usted va a utilizar, no hay problema, más adelante se indicará como se puede reprogramar para cualquier otra configuración.

Este interface, por el simple hecho de estar conectado al ordenador, genera una serie de nuevos comandos para simplificar el manejo desde el BASIC. También incorpora un software para su utilización desde CP/M.

Estos comandos permiten realizar las siguientes funciones:

— Reprogramación de los parámetros de la comunicación (velocidad de transmisión y recepción, control de paridad, n° de bits de stop y longitud de palabra).

- Transmisión/ recepción de caracteres simples.
- Transmisión/ recepción de bloques de caracteres.
- Examinar la palabra de estado del periférico.
- Redireccionar la salida de impresora a través de la RS-232.
- Volver a direccionar la impresora a través del port centronics.

Estos comandos no utilizan en absoluto la ram libre de usuario por lo que no suponen ni merma de esta ni incompatibilidad con programa alguno y se pueden utilizar como si fueran nuevas instrucciones de basic tanto dentro de una línea de programa, como en modo inmediato. Siempre deben ir precedidos del símbolo "|" que se genera con [shift] @

3. Funcionamiento

3.1 Utilización desde Basic:

En este apartado se explican detalladamente los comandos que se pueden utilizar desde el basic de Amstrad.

Los parámetros entre corchetes ([]) son opcionales.

3.1.1 | Sform.

Este comando se ha creado para poder modificar de una forma simple los parámetros de la transmisión. Su sintaxis es la siguiente:

| Sform, <velocidad de transmisión>, <velocidad de recepción>, <paridad>, <bits stop>, <bits dato>

<velocidad de transmisión> y <velocidad de recepción> son números o valores de variables que indican las respectivas tasas de baudios.

Las velocidades disponibles son:

50	baudios	1800	baudios
75	baudios	2000	baudios
110	baudios	2400	baudios
134,5	baudios	3600	baudios
150	baudios	4800	baudios
300	baudios	7200	baudios
600	baudios	9600	baudios
1200	baudios	19200	baudios

<paridad> es un número (o valor de variable) que puede tomar cualquiera de los 3 valores siguientes con el significado que a continuación se expresa.

0 ..... No paridad.
1 ..... Paridad impar.
2 ..... Paridad par.

<bits stop> es un número que puede tomar los siguientes valores:

0 ..... 1 bit de parada
1 ..... 1,5 bits de parada
2 ..... 2 bits de parada

<bits de dato> es un número que expresa la longitud del dato. Puede tomar los valores:

5 ..... 5 bits de dato.
6 ..... 6 bits de dato.
7 ..... 7 bits de dato.
8 ..... 8 bits de dato.

Ejemplo:

| sform,9600,4800,2,0,7

dá como resultado:

velocidad de transmisión	9600 baudios.
velocidad de recepción	4800 baudios.
Paridad	par
nº de bits de parada	1 bit.
longitud de dato	7 bits.

La ejecución del siguiente programa daría el mismo resultado:

```
10 | sform,9600,4800,2,0,7
```

Y también:

```
10 a=9600:b=4800:c=2:d=0:e=7
20 | sform, a,b,c,d,e
```

3.1.2 | Ssend.

Su misión es el envío de un carácter simple. Su sintaxis es:

| ssend, [<tiempo de espera>], <status>, <dato>

<Tiempo de espera> es un número que indica en ms el tiempo en que se intentará transmitir el carácter. Si al cabo de este tiempo no se ha logrado enviar el carácter, por no estar listo el receptor, se devuelve el control al basic. Este parámetro puede tomar valores comprendidos entre **0** y 65535; el valor **0** corresponde a un tiempo de 65536 ms. Si este parámetro no se especifica, la rutina esperará por un tiempo indefinido hasta que pueda enviar el dato. <Status> es una variable entera que previamente hay que definir. El comando modifica el valor de esta variable. Los valores que devolverá serán:

0 x 256 El carácter ha sido enviado correctamente.
1 x 256 Error de sintaxis.
2 x 256 El carácter no ha sido enviado en el tiempo de espera especificado.

<dato> es un número comprendido entre 0 y 255. Puede ser el código ASCII de un carácter.

Ejemplos:

s%=0: | ssend,1000,@ s%,&41

Intenta enviar el carácter hexadecimal 41; si al cabo de 1000 ms o sea 1 seg no lo ha conseguido retornará con S%=512. Si en lugar de &41 se pone 65 ó ASC("A") el resultado será el mismo.

s%=0: | ssend,@ s%,66

Intenta enviar el carácter decimal 66. No retornará hasta que el carácter haya sido enviado.

3.1.3 | Srecb

Su función es recibir un carácter simple. Su sintaxis es:

| srecb, [<tiempo de espera>], <status>, <dato>

<tiempo de espera> tiene el mismo significado que en el caso anterior. Si en el tiempo especificado no ha llegado ningún carácter se devuelve control al basic.

<status> también es análogo al caso anterior. Difiere únicamente en el significado del valor que retorna, siendo éste el siguiente:

0 x 256 carácter recibido correctamente
1 x 256 error de sintaxis
2 x 256 carácter no recibido en el tiempo especificado
3 x 256 Detectado break en la transmisión
4 x 256 error de trama
5 x 256 error de atropellamiento (overrun en inglés). Sucede cuando llega un nuevo carácter sin haber leído todavía el anterior.
6 x 256 error de paridad.

<dato> es en este caso una variable entera que se debe definir previamente Esta variable tomará el valor del carácter recibido.

Ejemplos:

s%=0:d%=0: | srecb,1000,@ s%,@ d%

Si se recibe por ejemplo D%=65

print chr $ (d%) = print "A"

Análogamente al caso anterior si no se especifica tiempo de espera, el comando aguardará indefinidamente hasta que reciba un carácter.

s%=0:d%=0: | srecb,@ s%,@ d%

3.1.4 | Breaksend

Este comando envía un break por la linea de transmisión.
Su sintaxis es:

| breaksend, <status>, <tiempo de break>, <tiempo de espera>

<status> es una variable entera definida previamente. Puede tomar los valores siguientes:

0 break transmitido
255 break no transmitido en el tiempo de espera especificado.

<tiempo de break> valor en ms del tiempo que la línea de transmisión va a permanecer en break.
<tiempo de espera> tiene el mismo significado que en los casos anteriores pero en este comando no es opcional. Nótese que no va entre corchetes.

3.1.5 | Outstring

Este comando permite enviar de una sola vez un bloque de caracteres, definidos previamente en una variable alfanumérica.
Su sintaxis es la siguiente:

| outstring,[<tiempo de espera>], <status>, <string>, <fin de string>

<tiempo de espera> desempeña un papel análogo al ya explicado anteriormente.
Este parámetro es opcional.
<status> es también análogo a los casos ya explicados. Los valores que puede tomar en función de los diferentes casos son los siguientes:

0 x 256 + **0** string enviado normalmente.
1 x 256 + **0** error sintáctico o string de longitud nula.
2 x 256 + **N** carácter **N** no enviado en el tiempo de espera especificado.

<string> es la variable alfanumérica que se desea enviar.
<fin de string> es un valor entre **0** y 255 (carácter). Indica fin de string.

Ejemplo:

s%=0:a$="MHT ingenieros": | outstring, 1000, @ s%, @ a$,&0d

envía la cadena : "MHT Ingenieros"
seguida de CR (CR=13 = &**0D**)

Si en el tiempo de 1 sg no ha conseguido enviar nada, retorna con el valor de status correspondiente.

s%=0:A$="MHT ingenieros": | outstring, @ s%, @ a$,&0d

realiza la misma función pero esperando todo el tiempo que sea necesario.

3.1.6 | Instring

Es el complementario de Outstring. Permite recibir de una vez una cadena completa.

Su sintaxis es:

| instring, [<tiempo de espera>], <status>, <string>, <fin de string>

Es análogo al caso anterior salvo en el status retornado que en este caso es:

0 x 256 + **0** string recibido normalmente
1 x 256 + **0** string de longitud nula o error sintáctico
2 x 256 + **N** carácter **N** no recibido en el tiempo de espera especificado.
3 x 256 + **N** recibido break en carácter **N**
4 x 256 + **N** carácter **N** recibido con error de trama
5 x 256 + **N** carácter **N** recibido con error de atropellamiento
6 x 256 + **N** carácter **N** recibido con error de paridad.

3.1.7 | sstat

Sirve para leer la palabra de estado del interface.

Su sintaxis es:

| sstat, <status>

<status> es una variable entera definida previamente. Su significado es:

bit 7 DSR: a 1 indica que la línea DSR está activa.
bit 6 Break: a 1 indica que se está produciendo un break en la línea
bit 5 error de trama.
bit 4 error de atropellamiento
bit 3 error de paridad
bit 2 buffer de transmisión vacío
bit 1 recibido un carácter
bit **0** listo para transmitir

Nota: Si <status> es S%
Bit n es análogo a S% and 2^n

Ejemplo:

s%=0: | sstat, @ s%

Si S% AND 2^3=1 entonces se ha producido error de paridad

3.1.8 | Sprnt

Sirve para redireccionar la impresora (# 8) hacia el interface RS232.

Su sintaxis es:

| sprnt, <protocolo>

<protocolo> puede tomar los valores siguientes:
0: No se utiliza protocolo de comunicación con la impresora serie
1: Se utiliza protocolo XON/XOFF

3.1.9 | Pprnt

No utiliza ningún argumento. Su misión es devolver el canal de impresora (# 8) al port centronics.

3.2 Utilización desde CP/M 2.2

Este interface se puede utilizar sin ningún problema con todos los programas CP/M (PIP, STAT, etc)

El interface funciona como el periférico lógico TTY.

Para ello y dado que el sistema operativo no está inicialmente adaptado para este interface, se ha previsto una rutina incluida en el software residente en el interface, que basta con llamar para modificar el BIOS y adaptar perfectamente el periférico al sistema operativo.

Esta rutina se encarga también de programar las distintas configuraciones de la comunicación.

Veamos con detalle el procedimiento a seguir:

La primera vez que use usted este periférico con el sistema operativo CP/M 2.2 escriba lo siguiente, teniendo en cuenta que los caracteres en inversa son las respuestas del ordenador.

paso 1

```
A>DDT
DDT VERS 2.2
-S100
0100 01 DF
0101 BC 03
0102 0F 01
```

```
0103 C3 7E
0104 8B D1
0105 01 01
0106 43 .
-^C
```

paso 2 A> **SAVE 1 SIO.COM**

paso 3
```
A> ^C
A> SIO
```

A partir de este momento el ordenador le irá pidiendo que seleccione las diferentes opciones. Cuando termina vuelve nuevamente al sistema.

paso 4 A> **SAVE 1 SIO.COM**

En este punto el sistema operativo está perfectamente adaptado al interface y una zona del programa creado SIO.COM contiene la información de las opciones seleccionadas. A partir de este momento usted ya puede trabajar normalmente con el sistema operativo. Si hiciera nuevamente:

paso 5
```
A> ^C
A> SIO
```

Aparecerá en pantalla la opción elegida. Si hace:

```
A> ^C
A> SIO I
```

El programa le pedirá nuevamente que seleccione opciones. Si a continuación hace:

paso 6 A> **SAVE 1 SIO.COM**

La nueva configuración elegida se grabará en el disco; si no ejecuta el SAVE el periférico se configura con la nueva opción, pero al hacer nuevamente:

paso 7
```
A> ^C
A> SIO
```

Volverá a tener la opción anterior.

Las sucesivas veces que quiera usted trabajar con el periférico y el sistema operativo CP/M basta con hacer

A> **SIO**

Siguiendo, si lo desea, los pasos a partir del 5

3.3 Utilización desde CP/M plus (Sólo para CPC 6128)

Lo mismo que sucedía en el caso del CP/M 2.2 sucede también en el CP/M plus, siendo necesario igualmente adaptar el sistema. Sin embargo en este caso basta con modificar la 1.ª vez algunos segmentos del programa **C10CPM3.EMS.** Las siguientes veces ya no ha de hacer absolutamente nada.

Para modificar el fichero **C10CPM3.EMS** siga atentamente los siguientes pasos:

Primeramente cargue el sistema operativo CP/M 2.2 y siga los pasos siguientes:

```
A>DDT
DDT VERS 2.2
-S100
0100 01 DF
0101 BC 03
0102 0F 01
0103 C3 0F
0104 8B C6
0105 01 01
0106 43 .
-G0
A>SAVE 1 FILE.COM
A>^C
A>FILE
A>SAVE 1 FILE.COM
```

En este punto tiene usted un fichero (FILE.COM) que contiene todas las diferencias que hay que introducir en el **C10CPM3.EMS.**

Ahora apague el ordenador y cargue el sistema operativo CP/M plus.

En la otra cara del disco de sistema tiene, entre otros, dos programas: SAVE.COM y SID.COM.

Por tanto debe dar la vuelta al disco y hacer:

```
A>SAVE
A>SID
CP/M 3 SID - Versión 3.0
#
```

Ahora extraiga el disco CP/M plus e introduzca el CP/M 2.2 donde generó el archivo FILE.COM y haga:

```
#RFILE.COM
NEXT MSZE PC END
0200 0200 0100 D2FF
#M100,200,7000
#
```

Ahora introduzca nuevamente el disco del CP/M plus (por la cara A) donde se encuentra el fichero **C10CPM3.EMS** y haga lo siguiente:

```
#RC10CPM3.EMS
NEXT MSZE PC END
6500 6500 0100 D2FF
#M7000,7057,46E
#M7058,70C8,4E9
#M70C9,70E8,584
#^C
CP/M 3 SAVE — Versión 3.0
Enter file (type RETURN to exit): C10CPM3.EMS
Delete C10CPM3.EMS? Y
Beginning hex address 100
Ending hex address 64FF
A>
```

En este momento ya tiene introducidas todas las modificaciones del fichero **C10CPM3.EMS.**

La siguiente vez que cargue usted el sistema CPM plus aparecerá en pantalla el siguiente mensaje:

CP/M Plus Amstrad Consumer Electronics plc
v 1.0,61k TPA,1 disc drive,1 serial port

Esto indicará que todo ha ido perfectamente y por tanto ya tiene acceso al periférico desde el sistema operativo.

Todos los programas funcionarán con normalidad incluidos SETSIO.COM, DEVICE.COM etc.

Si lo desea puede borrar el fichero FILE.COM generado en el disco de CP/M 2.2 puesto que ya no tiene ninguna utilidad.

NOTA IMPORTANTE: Con el fin de no modificar los discos del sistema originales le recomendamos que todas las operaciones descritas anteriormente las realice sobre copias de estos.

4. APLICACIONES

— Cable para comunicarse con un QL Sinclair

QL		AMSTRAD
1	GND	7
2	TxD	2
3	RxD	3
4	DTR	20
5	CTS	5
6	RTS	6

Existe en el mercado el cable comercializado por L.S.B, S.A. referencia 110006 que se adapta perfectamente a estas especificaciones.

— Cable para comunicarse con un Spectrum Sinclair.

Si utiliza el Interface I

Interface I	AMSTRAD
2	2
3	3
4	20
5	5
7	7
9	6

Si utiliza el Interface RS232-centronics INDESCOMP.

Interface RS232	AMSTRAD
2	2
5	3
3	20
6	5
4	7

5. Especificaciones de Hardware

En el conector D-25 normalizado se encuentran las siguientes señales

PIN	FUNCION	
1	GND	Masa.
2	TxD	Transmisión de datos (salida).
3	RxD	Recepción de datos (entrada).
4	RTS	(Request to send) indica que está preparado para transmitir datos (salida).
5	CTS	(Clear to send) habilita la recepción de datos (entrada).
6	DSR	(data set ready) habilita la transmisión de datos (entrada).
7	GND	Masa.
20	DTR	(data terminal ready) indica que está preparado para recibir datos (salida).

Los ports utilizados son los siguientes:

FADC	UART 8251 datos
FADD	UART 8251 control/status
FBDC	contador **0** del 8253 reloj de transmisión.
FBDD	contador **1** del 8253 reloj de recepción.
FBDE	contador **2** del 8253 temporizador de 1 ms.
FBDF	Palabra de modo del 8253.

El software en ROM se encuentra en la ROM nº 1.