PROTOCOLO DE COMUNICACIÓN
(D.T.P.)
Display Transfer Protocol
M&P ELECTRONICS

PARA MODELOS DE PANTALLA
REVISIÓN DEL MANUAL 23-07-02

ESPECIFICACIONES DEL SISTEMA

Las pantallas pueden mostrar información programada mediante un PC IBM o compatible, asociado al puerto de RS232 del que deberá disponer el ordenador. El protocolo de comunicaciones ha sido probada en ordenadores 386-25Mhz, 486-100Mhz y Pentium. El protocolo puede usarse tanto en RS485 como en RS232 siguiendo las debidas especificaciones para cada caso.

LA COMUNICACIÓN RS232/RS485 CON LAS PANTALLAS

El protocolo de transmisión:

Comprende un byte de sincronismo, la longitud del paquete, el nº de pantalla, la orden que se pretende dar a la pantalla, los datos y el checksum. Cada paquete se lanza vía RS232/RS485, y si se recibe correctamente, la pantalla devuelve una contestación. El paquete sólo debe reenviarse si se han producido errores en la transmisión (retornados por la pantalla) o bien pantalla no ha devuelto ACK**. Atención !!!** En caso de no recibir el ACK deberemos seguir el siguiente procedimiento para determinar si es necesario reenviar el paquete:

1. Enviar la orden GETNPAC después del fallo de comunicación: la pantalla enviará el número de paquete que toca transmitir

    1.1. Si éste coincide con el del paquete actual, no será necesario el reenvío. La pantalla ha transmitido el ACK pero no ha sido recibida por nuestro PC.

    1.2. Si éste no coincide con el del paquete actual, será necesario el reenvío, la pantalla no ha recibido la información o bien lo recibido no tiene sentido para ella.

El reenvío de información sin los debidos controles puede provocar que la información en un determinado programa quede duplicada provocando una ejecución no deseada por el usuario.

Identificación del nº de pantalla:

La pantalla no contestará en caso de que su número no coincida con el número de dirección en los paquetes. El número de pantalla por defecto es el **1**.

PROGRAMACIÓN AVANZADA

Parámetros de transmisión:

Los parámetros de transmisión son configurables :

9600 - valor por defecto-,pudiendose elegir 1200, 2400, 19600 y 38400 baudios.
NO PARITY
8 BITS
1 STOP BIT

Tipos de datos a transmitir a la pantalla:

- Programas a ser interpretados por la pantalla.
- Gráficos y animaciones, - no soportados por la serie D.I. -.
- Software para actualizar la pantalla, - no soportados por la serie D.I. -.
- Fonts - tipos de letra - de caracteres, - no soportados por la serie D.I. -.

Tipos de memoria en las pantallas:

Las pantallas D.I. tienen un sólo tipo de memoria, la memoria de programas, mientras que las series V-4 y M.L. tienen además el *heap*. La primera de ellas está gestionada con mucha eficacia, y sólo puede contener programas; se abren los archivos con la orden EDITAR. Mediante la orden EDITFILE se abre un archivo en el *heap*, y puede contener gráficos, animaciones y archivos binarios para actualizar el software de la pantalla , o los fonts de la pantalla. En los modelos V-4 y M..L., el *heap* posee 64Kb, mientras la memoria convencional es de entre 32K y 64K según el modelo de la pantalla.

Campos de los paquetes a transmitir:

¡Error! Marcador no definido.SYN (1 byte)	Byte 16H
NUM.B. (2 byte)	Indica el número de bytes del paquete (todos los bytes entre SYN y CHECKSUM, éstos inclusive)
Nº (1 byte)	Indica el número de identificación de la pantalla, (modificable). Por defecto y salvo casos especiales, todas las pantallas salen con el Nº 1 de fábrica.
COD (1 byte)	Indica el código de la orden a ejecutar.
DATOS (N bytes)	Puede ser el texto del un programa, con sus efectos, modos de aparición, etc., que va a ejecutarse en la pantalla, o bien software para actualizar la pantalla, gráficos o bien fonts (tipos de letra).
NOM.P. (8 bytes)	Nombre del programa. Puede tener 8 bytes como máximo. Los nombres son pasados a mayúsculas al ser recibidos por la pantalla, y sólo se acepta ASCII de 7 bits. En la orden EDITFILE es indispensable que el campo tenga 8 bytes exactos y el primero de ellos sea un asterisco '*' (ASCII 2AH). Ponga espacios a la derecha en caso necesario.
EditFile (5 bytes)	Estructura situada después del nombre del programa en EDITFILE, que indica el tipo de datos que se envían así como el tamaño del archivo. Descrita en el Apéndice B.
GETFILE (8 bytes)	Estructura situada después del nombre del archivo en GETFILE. Indica longitud y posición del paquete a transmitir. Descrita en el Apéndice B.
CHECKSUM (2 bytes)	Suma de todos los bytes del paquete menos los dos de checksum.

Los campos DATOS, NOM.P. EditFile y GETFILE deben colocarse solo en determinados tipos de paquete. Los otros campos son obligatorios y se colocan en todos los tipos de paquete.

Códigos de transmisión y paquetes:

RESET (COD = 0x01)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Inicializa la pantalla borrando todos los programas de la memoria sin inicializar el software interno de la pantalla.

WDRESET (COD = 0x02)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**) .Inicializa la pantalla borrando todos los programas de la memoria inicializando el software interno de la pantalla

STOP (COD = 0x03)

| SYN | NÚM. B. | Nº | COD | CHECKSUM |
Para la ejecución del programa, si hay alguno ejecutándose.

EJECUTAR (COD = 0x04)

| SYN | NÚM. B. | Nº | COD | **NOM.P**. | CHECKSUM |
Ejecuta el programa cuyo nombre viene dado en el campo siguiente del código.

BORRAR (COD = 0x05)

| SYN | NÚM. B. | Nº | COD| **NOM.P.** | CHECKSUM |
Borra un programa de la memoria de la pantalla, el nombre del cual está en el campo siguiente del código. En caso de no existir, la pantalla devuelve el error correspondiente.

EDITAR (COD = 0x06)

| SYN | NÚM. B. | Nº | COD| **NOM.P.** | CHECKSUM|
Editar programa. Prepara programa para la edición, poniéndolo en la zona de edición (si ya existía) o creándolo de nuevo.

CHECKSUM (COD = 0x07)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Solicita checksum del último paquete recibido . Se deberá recibir el **ACK** y el Byte alto del último checksum transmitido.

BAUDS (COD = 0x08)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Uso interno **M&P Electronics** no se responsabiliza de mal uso de esta orden

GETPROG (COD = 0x09)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(Función obsoleta)

SETHORA (COD = 0x0A)

| SYN | NÚM. B. | Nº | COD| **FECHA+HORA** | CHECKSUM|

Programa el RTC (Real Time Clock) con 6 bytes que indican año, mes, día, horas, minutos y segundos respectivamente.

GETHORA (COD = 0x0B)

| SYN | NÚM. B. | Nº | COD| **FECHA+HORA** | CHECKSUM|
Recibe 6 bytes que indican año, mes, día, horas, minutos y segundos respectivamente.

ENVIAR (COD = 0x0C)

| SYN | NÚM. B. | Nº | COD| **... DATOS ...** | CHECKSUM|
Se usa esta orden tanto si el PC envía algo a la pantalla como viceversa. Este paquete puede contener cualquier tipo de datos, tanto programa, como fonts o software. Estos datos se colocarán en distintos tipos de memoria, según se haya usado EDITAR o EDITFILE para abrir el archivo.

GETDIR (COD = 0x0D)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

PASSWORD (COD = 0x0E)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Uso interno **M&P Electronics** no se responsabiliza de mal uso de esta orden. El mal uso de este provocar bloqueos del control remoto por motivos de seguridad.

SETPASSW (COD = 0x0F)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Uso interno **M&P Electronics** no se responsabiliza de mal uso de esta orden. El mal uso de este provocar bloqueos del control remoto por motivos de seguridad.

SETTEMP (COD = 0x10)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Establece el Offset de temperatura en la pantalla.

SETTEMP (COD = 0x11)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Solicita la temperatura a la pantalla

GETVER (COD = 0x12)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Solicita a la pantalla el tipo de producto y su versión de software

ADDVAL (COD = 0x13)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

GETVIDEO (COD = 0x14)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

GETSTAT (COD = 0x15)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

EDITFILE (COD = 0x16)

| SYN | NÚM. B. | Nº | COD| **NOM.PR.** | **EditFile** | CHECKSUM |
Editar archivo de gráficos en el *heap*. Crea el archivo en el *heap* si el nombre del archivo no existía o no tenía el mismo número de bytes reservados. El tipo de datos enviado se distingue mediante una estructura del tipo EditFile, enviada después del nombre del programa, y se valida con otra estructura del tipo cap_MP al principio del archivo. Las estructuras EditFile y cap_MP están descritas en el Apéndice B. El campo de nombre del programa ha de ser de 8 bytes exactos, y el primero ha de ser el asterisco '*' (ASCII 2AH).

FLASHOTP (COD = 0x17)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Uso interno **M&P Electronics** no se responsabiliza de mal uso de esta orden. El mal uso de este provocar un mal funcionamiento el en producto.

GETVECT (COD = 0x18)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
Uso interno **M&P Electronics** no se responsabiliza de mal uso de esta orden. El mal uso de este provocar un mal funcionamiento el en producto.

NGETDIR (COD = 0x1E)

| SYN | NÚM. B. | Nº | COD | CHECKSUM |
Pide a la pantalla la estructura de directorio, que está constituida por estructuras del tipo nom_arx. El final del directorio viene determinado porque el campo *pos vale NULL. La estructura de directorio es enviada al ordenador (nº FEH) mediante la orden ENVIAR.

NEJECUTAR (COD = 0x1F)

| SYN | NÚM. B. | Nº | COD | **NOM.P**. | CHECKSUM |
Ejecuta el programa cuyo nombre viene dado en el campo siguiente del código.

GETFILE (COD = 0x20)

| SYN | NÚM. B. | Nº | COD | NOM.PR. | GetFile | CHECKSUM |
Permite recuperar de la pantalla cualquier archivo (en ram de programas o en el *heap*) en forma de paquetes de tamaño variable. Éstos serán enviados al ordenador (destino = MASTER 0xFE) mediante la orden ENVIAR. El origen del paquete, respecto a la posición inicial del archivo, y su longitud, son asignados en la estructura GetFile, que se coloca en el paquete después del campo 'nombre'. Si en la estructura GetFile - Apéndice B -, el campo denominado **pos** es superior al final del paquete inicial, la pantalla devuelve el error 22 (POS_DEMASIADO_GRANDE).

GETNPAC (COD = 0x21)

| SYN | NÚM. B. | Nº | COD | NOM.P. | CHECKSUM |
Devuelve el nº del paquete que toca recibir - empezando por el 0 -. Aconsejamos que se use este paquete para confirmar la recepción (en caso de no recibir ACK), en vez de usar la orden CHECKSUM. El nº que se devuelve es módulo 256.

TODED (COD = 0x22)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

FROMDED (COD = 0x23)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

GETDED (COD = 0x24)

| SYN | NÚM. B. | Nº | COD| CHECKSUM|
(**Función obsoleta**)

PUTCNT (COD = 0x25)

| SYN | NÚM. B. | Nº | COD| CONTADOR | CHECKSUM|
Actualiza un contador determinado en un D.E.D.

GETCNT (COD = 0x26)

| SYN | NÚM. B. | Nº | COD| CONTADOR | CHECKSUM|
Solicita el valor de un contador determinado(**Función obsoleta**)

FASTEXEC (COD = 0x27)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
Transmite el contenido de un programa que se ejecutara de forma automática en la pantalla, este paquete esta destinado para la actualización de datos en una pantalla de una forma muy rápida.

MP_PUTPORT (COD = 0x28)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
Esta orden permite el control de los reles de la pantalla a través del protocolo de comunicación. El campo DATOS incluira los cinco parametros siguientes:

1- Nº de puerto (0x01,0x02 ……)
2- Estado de inicio (0x01 – ON ó 0x02 – OFF)
3- Nº de repeticiones (0x01 – 0Xff)
4- Tiempo en ON (0x01 – 0Xff) ¼ Segundos
5- Tiempo en OFF (0x01 – 0Xff) ¼ Segundos

DCFESTAT (COD = 0x29)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
Solicita el estado del DCF Actual .El contenido datos contiene la estructura detallada a continuación. Los campos muestran el último paquede DCF recibido y el último recibido correctamente.

```
struct control_dcf  {
                unsigned char dia;
                unsigned char mes;
                unsigned char ann;
              unsigned char dif_hora;
                unsigned char dif_minuto;
                };

struct dcf_pack {
              struct control_dcf lastbo;
              struct control_dcf last;
              };
```

QUERYACTION (COD = 0x2A)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
Esta orden nos permite preguntar a un dispositivo TR40 configurado como gestor de simple si se ha producido una acción estado configurado como (gestor de colas simple).
En caso de no producirse ninguna acción el contenido del campo DATOS será 0.
Cuando el primer byte de este campo sea 1 (incrementar) ó 2 (decrementar) se recibirá a continuación el nº de contador y el nº de grupos a los cuales afecta la actión producida.
Cada grupo consta de 5 bytes con el siguiente signigicado.

1- Pantalla inicial del Grupo
2- Pantalla final del Grupo
3- Dígito inicial dentro de cada pantalla
4- Dígito final dentro de cada grupo
5- Byte de control especial para expresar el Nº de ventanilla después del último dígito.

Por último cuando el primer byte sea 3 significará que se ha establecido un valor manual en el contador. En este caso a continuación se mandará la siguiente la siguiente estructura seguida del nº de grupos con el formato anterior mencionado.

```
struct putcnt   {
              unsigned int  num_cnt;     Nº De contador
              unsigned long val_cnt;      Valor del contador  quese ha introducido manualmente.
              };
```

PUTCNT2 (COD = 0x2B)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
El comando putcnt2 permite actualizar el contenido de los contadores de un dispositivo utilizando la siguiente estructura en el campo DATOS

```
struct putcnt   {
              unsigned int  num_cnt;     Nº De contador
              unsigned long val_cnt;      Valor del contador a actualizar.
              };
```

QTRFAST (COD = 0x2C)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
La Orden funciona igual a QUERYACTION pero no envia la información sobre los grupos de pantallas afectados. Se limita a respoder 0 (Ningún evento) 1 (Incrementar contador) 2 (Decrementar Contador). Si se devuelve un 3 significa que se ha variado el contador manualmente y se transmite el número de contador (1 por defecto) y el nuevo contador.

SETROUTE (COD = 0x35)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
La orden SETROUTE permite transmitir una información a un puerto RS232 como por ejemplo una impresora utilizando como transporte una red de RS485 con protocolo DTP.
Este comando solo esta disponible para el Gestor de Red trabajando en modo ROUTER.
Utilizando este sistema se podrá mandar la información **DATOS** al gestor con el protocolo **DTP** y una vez recibida toda su trama se enviará esta información por el puerto RS232 al dispositivo que se conecte a este puerto, es posible utilizar equipos industriales, impresoras con interface serie, etc.

GETROUTE (COD = 0x36)

| SYN | NÚM. B. | Nº | COD| **DATOS** | CHECKSUM|
La orden GETROUTE efectua la operación contraria a SETROUTE, transmite al Master un buffer con la información recibida por el puerto RS232 en el campo **DATOS** dentro del protocolo **DTP**. Al igual que SETROUTE esta orden solo está disponible para el Gestor de Red operando en modo ROUTER.
Este último caso es ideal para conectar lectores de barras o bien otros dispositivos de este tipo a un red RS485 con protocolo **DTP.**

Respuesta general de la pantalla

| ACK | CÓDIGO ERROR |
Si la orden ha llegado correctamente, el DTE recibirá ésta confirmación. En ella hay un código de validación fijo (ACK) y un código de error de la pantalla. Éste normalmente será 0, pero puede darse el caso que se envíe una orden ilegal en determinado momento. El valor de los distintos códigos de error se muestra en el Apéndice A.
La orden CHECKS devuelve el byte bajo del último checksum recibido correctamente en el lugar del código de error.
La orden GETNPAC devuelve el nº de paquete que toca transmitir en el lugar del código de error.

Descripción del protocolo

Cada paquete debe ir separado al menos 16 ms. del paquete contestación, para dar tiempo a que las pantallas se pongan en recepción. En caso de que una pantalla reciba incorrectamente un paquete, se ha establecido un tiempo de reset de paquete de 100 ms. Es decir, al cabo de 100 ms. la pantalla entiende que el próximo byte que reciba será el primero de un nuevo paquete. En aplicaciones bajo **RS485** es recomendable dejar siempre 100 ms. Entre paquete y paquete ya que todas las pantallas de la red deben recibir correctamente el paquete de comunicación aunque este no sea para ellas provocando asi que el tiempo de 16 ms. Fuese insuficiente. Es necesario esperar el tiempo de reset si se ha recibido incorrectamente un paquete. El tiempo entre el último byte del paquete completo (que llega a la pantalla) y su contestación varia entre 16 ms. y 1 segundo (en caso que implique muchas operaciones internas).
Los paquetes empiezan por el byte de sincronismo (SYN), luego 2 bytes que indican la longitud del paquete. El checksum es una suma de control de 2 bytes que incluye todos los bytes menos a sí mismo.
La longitud y el checksum se envían primero el byte bajo y después el alto.
Cada programa se ha de enviar después de un paquete EDITAR, de manera que cuando se reciban las ordenes o efectos del programa de la pantalla, éste sepa en que programa colocarlos; si no, los pondría en el último programa que se editó.
Mientras se está ejecutando un programa no se puede editar otro, a menos que se envíe la orden EDITAR del nuevo programa antes de ejecutar el programa antiguo. Si no se hace así, y el nuevo programa no está en el buffer de edición, el Data Terminal Equipment - a partir de ahora DTE - recibirá el código de error NO_PUEDE_EDITAR.
Al enviar la orden EDITAR , se comprueba si el programa existe; si no existe, se crea uno de nuevo; si existe, el nuevo programa sobrescribe al anterior.
Si se está ejecutando el programa XX y se envía un 'EJECUTA YY', el programa 'YY' se empezará a ejecutar inmediatamente.
Si se envía 'STOP', el programa parará la ejecución inmediatamente.
Si un byte se ha enviado con más de 100 ms. de retardo respecto al último byte enviado, la pantalla entiende que ha empezado la transmisión de otro paquete, y por tanto se resetean en nº de bytes, checksum, etc.
Si la pantalla no recibe correctamente la orden, o el checksum estaba equivocado, etc; la pantalla se mantiene en silencio, sin contestar.
La forma correcta de operar sería:
Primeramente enviar la orden deseada, si no se recibe contestación antes de 2 seg. , enviar la orden GETNPAC para verificar si la pantalla recibió la primera orden (se sabrá comparando, el número de paquete transmitido). Si los dos números son iguales significa que se perdió el ACK. El usuario puede entonces escoger entre reenviarle la primera orden (para comprobar si se ha producido un error de otro tipo), o no, sabiendo que en el primer caso, y con la orden enviar, el paquete quedadá duplicado en la pantalla. En caso de que los dos números sean diferentes, indudablemente ha habido un error de transmisión en el sentido del DTE a la pantalla, de manera que es necesario verificarlo con un reenvío de la primera orden.

NOTA: Todas las órdenes que impliquen el envío de más de 1 byte de la pantalla al DTE, serán transmitidas mediante la orden ENVIAR, y con el nº de identificación FEH (con que consideramos el PC).

Edición de un programa para ejecutarlo en la pantalla

Una vez se ha creado el programa mediante la orden EDITAR, éste debe llenarse con los efectos, modos de aparición, etc. que más tarde se ejecutarán en la pantalla. Se procederá enviando uno o varios bloques del tamaño deseado, mediante la orden ENVIAR. Finalmente se ejecutará el programa mediante la orden NEJECUTAR.

Edición de un gráfico

Después de crear el archivo mediante la orden EDITFILE, mediante uno o varios paquetes ENVIAR se van enviando todos los bytes del archivo. El formato de un archivo GMP es el siguiente:

- 32 bytes de la estructura cap_MP descrita en el Apéndice B.
- Tantos fotogramas como los indicados en cap_MP.nframes.

¡Error! Marcador no definido.b07 b17 b27 b37 b47 b57 b67 b77 b87 b97
b06 b16 b26 b36 b46 b56 b66 b76 b86 b96
b05 b15 b25 b35 b45 b55 b65 b75 b85 b95
b04 b14 b24 b34 b44 b54 b64 b74 b84 b94
b03 b13 b23 b33 b43 b53 b63 b73 b83 b93
b02 b12 b22 b32 b42 b52 b62 b72 b82 b92
b01 b11 b21 b31 b41 b51 b61 b71 b81 b91
b00 b10 b20 b30 b40 b50 b60 b70 b80 b90

Los fotogramas son monocromáticos, y cada uno de ellos está organizado de forma que los 8 primeros bits de la primera columna de la izquierda forman un byte, el siguiente lo forman los 8 primeros bits de la segunda columna, etc. El la ilustración adjunta, donde se muestran simbólicamente el número de los LED's del extremo superior izquierdo de una pantalla vista de frente. Cada LED se representa como bxy siendo 'x' el número del byte, y 'y' el número del bit.

La serie D.I. no soporta la edición de gráficos.

Creación de un gràfico *.GMP

Todo gráfico con extensión GMP, está estructurado en:

-Cabecera del fichero.
-Contenido del gráfico.

En el siguiente cuadro se muestra la cabecera de 32 bytes en forma de estructura en lenguaje **C**.

-Char tipo, es un carácter que determina el tipo de archivo que se envia - 1 byte -.

-00, equivale a gráficos.
-01, a programas.
-02, a fuentes.

-Char compresion (Reservado) -	1 byte -.
-Int x, (Reservado)	2 bytes.
-Int y, (Reservado)	2 byte.
-Int anchura	Define la longitud de pixels del gráfico - 2 bytes -.
-Int altura	Define en este caso, la altura en pixels del gráfico - 2 bytes -.
-Int nframes	Número de gráficos que tiene el programa - 2 bytes -.
-Is_conv:1	(Reservado) - 2 bytes -.
-Indeterminado[18]	(Reservado) -18 bytes -.

En el ejemplo siguiente, realizado sobre una pantalla de 128 pixels de anchura y 96 pixels de altura, se ilumina el pixel de la posición (22,6), siendo el punto (0,0), el vertice superior izquierdo.

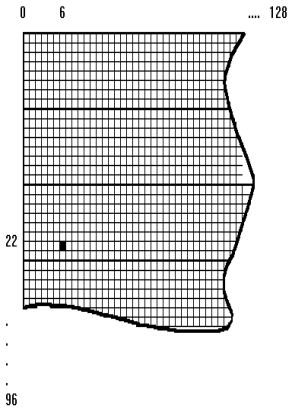
```
struct cap_gmp
        {
        unsigned char tipo;
```

```
unsigned char compresion;
unsigned int x;
unsigned int y;
unsigned int ancho;
unsigned int alto;
unsigned int frames;
unsigned int is_conv;
unsigned char ncolores;
unsigned char ntonos;
unsigned char peed;
unsigned char paginas;
unsigned char indeterminado[14];
} cap;
```

Creamos el archivo GRAFICO.GMP, cuyo volcado hexadecimal está reproducido a continuación.El gráfico de ejemplo esta realizado en monocromo pero el sistema de un gráfico de color es bastante parecido.
Las animaciones en color utilizan el campo de la estructura (páginas) para determinar cuantos fotogramas se unen para crear un único fotograma. En el caso de animaciónes en tricolor usaremos dos fotogramas:

Un primer fotograma de color Rojo.
Un segundo fotograma de color Verde.

Si ponemos a 1 un mismo bit en ambos fotogramas obtendremos el color amarillo.



Pixel encendido en la posición 22,6 (y,x)

```
0000H   ...   00 00 00 00 00 00 80 00-60 00 01 00 00 00 00 00
0010H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0020H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0030H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0040H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0050H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0060H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0070H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0080H   ...   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

```
0090H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00A0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00B0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00C0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00D0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00E0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
00F0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0100H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0110H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0120H  ...  00 00 00 00 00 00 02 00-00 00 00 00 00 00 00 00
0130H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0140H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0150H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0160H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0170H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0180H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0190H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
01A0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
01B0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
01C0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
01D0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
01E0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
01F0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0200H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0210H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0220H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0230H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0240H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0250H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0260H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0270H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0280H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0290H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
02A0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
02B0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
02C0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
02D0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
02E0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
02F0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0300H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0310H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0320H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0330H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0340H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0350H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0360H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0370H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0380H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0390H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
03A0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
03B0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
03C0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
03D0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
03E0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
03F0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0400H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0410H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0420H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0430H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0440H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0450H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0460H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0470H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0480H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

```
0490H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04A0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04B0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04C0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04D0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04E0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04F0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0500H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0510H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0520H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0530H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0540H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0550H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0560H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0570H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0580H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0590H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
05A0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
05B0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
05C0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
05D0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
05E0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
05F0H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0600H  ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0610H  ...  00 00 00 00 00 00 00 00-00
```

Seguidamente analizaremos de forma detallada la cabecera de un archivo *.GMP..

```
          1  2  3    4    5    6    7    8
         __ __ ____ ____ ____ ____ ____ ____ ____ ____
0000H ...  00 00 00 00 00 00 80 00-60 00 01 00 00 00 00 00   ........`.......

                                  9
          ________________________________________
0010H ...  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

1-	1 bytes de unsigned char tipo.		00H	0
2-	1 bytes de unsigned char compresion.		00H	0
3-	2 bytes de unsigned int x.	0000H	0	
4-	2 bytes de unsigned int y.	0000H	0	
5-	2 bytes de unsigned int anchura.		0080H	128
6-	2 bytes de unsigned int altura.		0060H	96
7-	2 bytes de unsigned int nframes.		01H	1
8-	2 bytes de unsigned is_conv:1.		0000H	0
9-	18 bytes de char indeterminado[18].			

Total de bytes empleados en la cabecera, 32.

El resto de las líneas del programa, definen el gráfico que debe aparecer, en este caso sólo se iluminará un pixel de la pantalla, dejando al resto apagados.
Es necesario conocer que la pantalla se divide comunmente en líneas. Una línea es una porción de la pantalla de 8 pixels de altura que empieza en el extremo izquierdo de la misma y termina en el derecho. En el ejemplo tenemos 96 pixels de altura que equivalen a doce líneas. Cada línea ocupa 128 bytes en el archivo, número equivalente a la anchura de la pantalla.

Podemos ver en el volcado hexadecimal que el bit encendido se encuentra en la posición 0126H. Como sabemos que cada línea ocupa 128 bytes y el pixel se encuentra en la tercera línea, hemos de sumar a los bytes que forman la cabecera (32 bytes), dos líneas completas (128 bytes cada una) y, 6 bytes debido a que la coordenada X es 6. La posición del bit puesto a 1 viene determinada por la fórmula 7 - (Y MOD 8), y en nuestro caso será b1, que es el segundo bit del byte, empezando por abajo.

0126H = 32 + 128 + 128 + 6

El archivo GRAFICO.GMP se ha segmentado en varios bloques de 255 bytes para reducir errores el porcentage de errores de transmisión. Seguidamente analizamos los distintos paquetes que componen el ejemplo de transmisión.

16 14 00 01 16 2A 47 52 41 46 49 43 4F 20 06 00 00 00 8C 02

16H	:SYN.
0014H	:Nº de bytes (20).
01H	:Nº de pantalla.
16H	:Código de EDITFILE
2A4752414649434F	:Nombre del archivo a editar ('*GRAFICO').
00000620H	:1568 bytes de longitud del archivo GRAFICO.GMP
00H	:Tipo de datos; en este caso un gráfico.
028C	:Checksum (todos los bytes sumados menos los 2 del checksum).

16 06 01 01 0C 00 00 00 00 00 00 80 00 60 00 01 00 00 00	Primer paquete de
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	transmisión del archivo
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	GRAFICO.GMP.
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 0B 01	

16 06 01 01 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Segundo paquete de
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	transmisión del
00 00 00 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00 00	archivo GRAFICO.GMP
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00 00 00 00 00 00 00 00 00 00 00 00 00 2C 00	

16 06 01 01 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Tercer paquete de
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	transmisión del

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2A 00	archivo GRAFICO:GMP
16 06 01 01 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2A 00	Cuarto paquete de transmisón del archivo GRAFICO.GMP
16 06 01 01 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2A 00	Quinto paquete de transmisión del archivo GRAFICO:GMP
16 06 01 01 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2A 00	Sexto paquete de transmisión del archivo GRAFICO.GMP

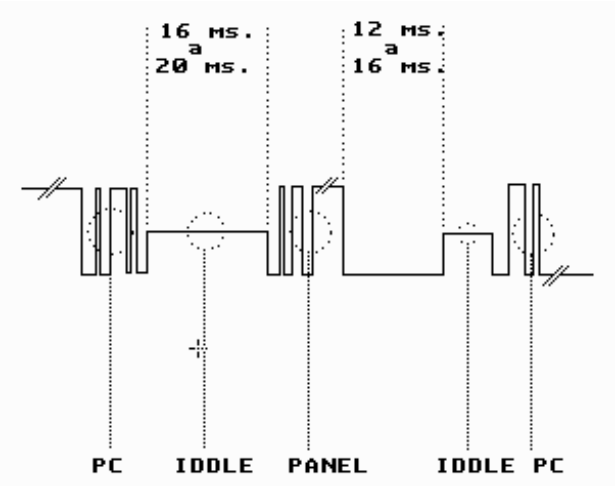
16 2D 00 01 0C 00 00 00 00 00 00 00 00 00 00 00 00 00 Septimo paquete de
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 transmisión del
00 00 00 00 00 50 00 archivo GRAFICO.GMP

16 0D 00 01 06 50 52 55 45 42 41 E9 01 EDITAR 'PRUEBA'

16 10 00 01 0C CE 2A 47 52 41 46 49 43 4F 26 03 ENVIAR ANIM
'*GRAFICO'

16 0D 00 01 1F 52 55 45 42 41 02 02 NEJECUTAR 'PRUEBA'

El sistema RS485



Se basa en la transmisión de datos mediante tensiones diferenciales. Está concebido para que se "cuelguen" en una misma línea de comunicaciones hasta 32 transmisores/ receptores RS485. El sistema consiste en que solamente haya un transmisor activo en un momento dado; los demás se mantienen en recepción, quedando en estado de alta impedancia. Para que no haya conflicto de líneas, un receptor se puede convertir en transmisor sólo si no hay ningún otro elemento transmisor. En el gráfico adjunto se muestra como debe establecerse la comunicación entre dos elementos de una línea RS485.

Primeramente aparecen la información transmitida por el DTE; vemos que cuando ésta termina, el driver del DTE se pone en recepción, quedando en alta impedancia o IDDLE LINE. Al cabo de entre 16 y 20 milisegundos, la pantalla activará su driver de transmisión, y empezará a transmitir si la recepción ha sido correcta (NOTA: en caso de órdenes que conlleven una gran cantidad de tiempo al ejecutarlas por la pantalla, como EDITAR, BORRAR o SETCONF, este tiempo puede retrasarse hasta 1 segundo). Cuando haya terminado, la pantalla se mantiene en modo transmisión durante 12 a 16 milisegundos. Al cabo de ese tiempo, se vuelve a poner en recepción, pudiendo activarse el driver del DTE para empezar a transmitir de nuevo otro paquete. Si no se cumplen estos tiempos, probablemente habrá algún conflicto de línea con los consiguientes errores de transmisión.

Ejemplos de transmisión

En los siguientes ejemplos se ha usado la pantalla nº 1 como destino (valor por defecto).

-EDITAR. Seguido el nombre del archivo en ASCII, de 8 caracteres como máximo.

P.e.:
16 0D 00 01 06 45 44 49 54 4F 52 F1 01

Desglosado:
16H : SYN.
000DH : nº de bytes (13).
01H : nº de la pantalla (1).
06H : Código de EDITAR.
454449544F52 : Nombre del fichero a editar (EDITOR).
01F1H : Checksum (todos los bytes sumados menos los dos de checksum).

-BORRAR. Seguido del nombre del archivo que se desea borrar.

P.e.:
16 0D 00 00 05 45 44 49 54 4F 52 F0 01

Desglosado:
16H : SYN.
000DH : nº de bytes (13).
01H : nº de la pantalla (1).
05H : Código de BORRAR.
454449544F52 : Nombre del fichero a borrar (EDITOR).
01F0H : Checksum (todos los bytes sumados menos los dos de checksum).

-EDITFILE. - No soportado en los pantallas de la serie D.I.-. Seguido el nombre del archivo en ASCII que debe ser de 8 caracteres exactamente (rellénese con espacios a la derecha en caso necesario), el primero de los cuales ha de ser un asterisco '*' ASCII 2AH. Seguidamente aparece una estructura del tipo EditFile que indica la cantidad de memoria a reservar para este archivo.

P.e.:
16 14 00 01 15 2A 50 52 4F 56 41 20 20 20 03 00 00 00 56 02

Desglosado:
16H : SYN.
0014H : nº de bytes (20).
01H : nº de la pantalla (1).
16H : Código de EDITFILE.
2A50524F56412020 : Nombre del fichero a editar ('*PROVA ').
00000320 : Longitud archivo (800 bytes).
00H : Tipo de datos; en este caso es un gráfico.
0256H : Checksum (todos los bytes sumados menos los dos de checksum).

-ENVIAR. Seguido del bloque de datos a enviar.

P.e.:
16 17 00 01 0C E0 50 52 55 45 42 41 20 44 45 20 54 45 58 54 4F 36 05

Desglosado:
16H : SYN.
0017H : nº de bytes (23).
01H : nº de la pantalla (1).
0CH : Código de ENVIAR.
E05052554542 : Modo CORRER seguido del texto:"PRUEBA DE TEXTO".
412044452054
4558544F
0536H : Checksum (todos los bytes sumados menos los dos de checksum).

-RESET PANTALLA. Inicializa la pantalla borrando toda la memoria.

P.e.:
16 07 00 01 01 1F 00
Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de la pantalla (1).
01H : Código de RESET PANTALLA.
001FH : Checksum.

-STOP PROGRAMA. Para la ejecución de un pantalla.

P.e.:
16 07 00 01 03 21 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de la pantalla (1).
03H : Código de STOP PROGRAMA.
0021H : Checksum.

-GETHORA. Pide la hora del reloj de la pantalla. La pantalla devuelve un paquete con la orden ENVÍAR, el buffer que contiene el campo DATOS tiene en siguiente formato

[0] Año
[1] Mes
[2] Dia
[3] Horas
[4] Minutos
[5] Segundos

P.e.:
16 07 00 01 0B 29 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de la pantalla (1).
0BH : Código de GETHORA.
0029H : Checksum.

-SETHORA. Configura la hora del reloj de la pantalla. Se envía en el formato año,mes,día,hora,minuto,segundo.

P.e.:
16 0D 00 01 0A 5F 04 1B 0F 1F 1B F5 00
Desglosado:
16H : SYN.
000DH : nº de bytes (13).
01H : nº de la pantalla (1).
0AH : Código de SETHORA.
5F041B0F1F1B : Año,mes,día,hora,min,seg (95,4,27,15,31,27) .
00F5H : Checksum.

-NEJECUTAR. Ejecuta un programa, si éste no se encuentra, devolverá error en el byte siguiente del ACK.

P.e.:
16 0D 00 01 1F 45 44 49 54 4F 52 0A 02

Desglosado:
16H : SYN.
000DH : nº de bytes (13).
01H : nº de pantalla (1).
1FH : Código de NEJECUTAR.
454449544F52 : Nombre del programa (EDITOR).
020AH : Checksum.

-CHECKSUM. Pide el byte bajo del checksum del último paquete recibido correctamente. Éste se enviará en el segundo byte del paquete ACK.

P.e.:
16 07 00 01 07 25 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de la pantalla (1).
07H : Código de CHECKSUM.
0025H : Checksum.

-PASSWORD. Da el password de 8 caracteres a la pantalla; si éste lo considera incorrecto, dará un error. Si el password que hay en la pantalla es el que se inicializó con un reset, o son 8 espacios, se considera que el password está desactivado, no dando nunca el error PASSWORD INCORRECTO.

P.e.:
16 0F 00 01 0E 20 20 20 20 20 20 20 20 34 01
Desglosado:
16H : SYN.
000FH : nº de bytes (15).
01H : nº de la pantalla (1).
0EH : Código de PASSWORD.
2020202020202020 : Password.
0134H : Checksum.

-SETPASSW. Inicializa el password de la pantalla mediante una palabra de hasta 8 caracteres.

P.e.:
16 09 00 01 0F 4D 50 CC 00

Desglosado:
16H : SYN.
0009H : nº de bytes (9).
01H : nº de la pantalla (1).
0FH : Código de SETPASSW.
4D50 : Nuevo password.
00CCH : Checksum.

-GETTEMP. Pide la temperatura y offset a la pantalla. Éste devuelve un paquete con un buffer de 2 bytes mediante la orden ENVIAR: la temperatura se encuentra en el primer byte (buf[0]), y el offset en el segundo byte (buf[1]) (Opcional).

P.e.:
16 07 00 01 11 2F 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de la pantalla (1).
11H : Código de GETTEMP.
002FH : Checksum.

-SETTEMP. Pone nuevo offset a la pantalla.

P.e.:
16 08 00 01 10 00 2F 00

Desglosado:
16H : SYN.
0008H : nº de bytes (8).
01H : nº de la pantalla (1).
10H : Código de SETTEMP.
00H : offset.
002FH : Checksum.

GETFILE. Permite recuperar de la pantalla cualquier archivo (en ram de programas o en el *heap*) en forma de paquetes de tamaño variable. Éstos serán eviados al ordenador (destino = MASTER 0xFE) mediante la orden ENVIAR. El origen del paquete, respecto a la posición inicial del archivo, y su longitud, son asignados en la estructura GetFile, que se coloca en el paquete después del campo 'nombre'. Si en la estructura GetFile, el campo **pos** es superior al final del paquete inicial, la pantalla devuelve el error 22 (POS_DEMASIADO_GRANDE)

P.e.:
16 17 00 01 20 45 44 49 54 4F 52 20 20 00 00 00 00 F9 00 00 00 4E 03

Desglosado:
16H : SYN.
0017H : nº de bytes (23).
01H : nº de pantalla (1).
20H : Código de GETFILE.
4544 9544F522020 : *Nombre del archivo.*
00000000 : *posición inicial del paquete solicitado.*
000000F9 : *tamaño de la sección del archivo solicitado.*
034EH : Checksum.

GETNPAC. Devuelve el nº de paquete que toca recibir la pantalla, empezando por el número 0.

P.e.:
16 07 00 01 21 3F 00

Desplosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de pantalla (1).
21H : Código de GETNPAC.
003FH : Checksum.

NGETDIR. Pide el directorio de la pantalla.

P.e.
16 07 00 01 1E 3C 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de pantalla (1).
1EH : Código de NGETDIR.
003CH: Checksum.

GETCNT. Solicita el contador Nº 1 a un D.E.D.

P.e.
16 07 00 01 1E 3C 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de pantalla (1).
26H : Código de GETCNT
0001H : Nº Contador
0081H : Checksum.

Despues de ACK + 0h Recibirá un paquete del tipo ENVIAR en el cual encontrará la siguiente estructura en la zona de Datos

unsigned int Contador;
unsigned long valor_contador;

PUTCNT. Establece el valor del contador Nº1 a el D.E.D. Nº 1

P.e.
16 07 00 01 1E 3C 00

Desglosado:
16H : SYN.
0007H : nº de bytes (7).
01H : nº de pantalla (1).
25H : Código de PUTCNT
0001H : Nº Contador
00000001H : Valor contador
0081H : Checksum.

El dispositivo devolverá ACK+Nº de dispositivo.

-FASTEXEC.

P.e.:
16 17 00 01 27 E0 50 52 55 45 42 41 20 44 45 20 54 45 58 54 4F 51 05

Desglosado:
16H : SYN.
1700H : nº de bytes (23).
01H : nº de la pantalla (1).
27H : Código de FASTEXEC.
E05052554542 : Modo CORRER seguido del texto:"PRUEBA DE TEXTO".
412044452054
4558544F
5105H : Checksum (todos los bytes sumados menos los dos de checksum).

- SETROUTER

DATOS:					P	R	U	E	B	A		R	O	U	T	E	R	
16	14	00	01	**35**	50	52	55	45	42	41	20	52	4F	55	54	45	52	20	04

-GETROUTER

16 07 00 01 **36** 54 00

Ejemplo de envío de un programa

CÓDIGO ENVIADO	PROGRAMA
16 0D 00 01 06 50 52 55 45 42 41 E9 01	EDITAR 'PRUEBA'
16 61 00 01 0C C4 35 30	ENVIAR: VelModo 50
C1 30	TipoLetra 0
E0 50 52 55 45 42 41 20 44 45 20 54 45	Correr 'PRUEBA DE TEXTO
58 54 4F 20 4C 45 54 52 41 20 30	LETRA 0'
C1 31	TipoLetra 1
E0 50 52 55 45 42 41 20 44 45 20 54 45	Correr 'PRUEBA DE TEXTO
58 54 4F 20 4C 45 54 52 41 20 31	LETRA 1'
C1 32	TipoLetra 2
E0 50 52 55 45 42 41 20 44 45 20 54 45	Correr 'PRUEBA DE TEXTO
58 54 4F 20 4C 45 54 52 41 20 32	LETRA 2'
F0 50 52 55 45 42 41	Inmediato 'PRUEBA'
F0 A7 78 1D	Inmediato Hor:Min
16 0D 00 01 1F 50 52 55 45 42 41 02 02	NEJECUTAR 'PRUEBA'

Códigos de edición

Sirven para dar órdenes a la pantalla, o pedirle datos.

ORDEN	COD	EXPLICACIÓN
ACK	06H	-Se ha recibido correctamente la orden.
SYN	16H	-Byte de sincronismo de paquete.

Tabla de códigos para la creación de programas

Estos códigos son utilizados para la creación de programas que seran transmitidos con la orden EDITAR + ENVIAR o bien con FASTEXEC siendo su ejecución inmediata.

CODIGO	FONT	PANT	DESCRIPCIÓN
Año	0x80	0x96	Dos cifras que indican el año
Número de mes	0x81	0x97	Dos cifras que indican el mes
Mes	0x82	0x98	Nombre del mes
Número de día	0x83	0x99	Dos cifras que indican el dia
Día	0x84	0x9A	Nombre del dia
Horas	0x85	0x9B	2 cifras que indican la hora del día
Minutos	0x86	0x9C	2 cifras que indican el minuto del dia
Segundos	0x87	0x9D	2 cifras que indican los segundos de la hora
Temperatura	0x88	0x9F	Temperatura
Blink	0x89	0xA0	Parpadea el texto que se encuentre entre 2 BLINKA
Grafico < n >	0x8A	0xA3	Aparece un gráfico
Diferencia dias	0x8B	0xA4	Dias para la fecha del evento
Diferencia semanas	0x8C	0xA5	Semanas para la fecha del evento
Flash < n >	0x8E	0xB0	Se producen n parpadeos en la pantalla
Negativo	0x8F	0xB1	El contenido de la pantalla se invierte de color
Borrado	0x90	0xB2	Se borra la línea donde estamos trabajando
Espera < n >	0x91	0xB3	Espera n/4 segundos hasta el ejecutar el próximo efecto
Fecha evento	0x92	0xCC	Fecha para DifDia, DifSem o DifMes
Grosor < n >	0x93	0xC0	Cada columna ocupará n columnas de grosor
Tito de letra < n >	0x94	0xC1	Tipos: de 0 a 10 (dependiendo del modelo)
Velocidad Modo < n >	0x95	0xC4	Velocidad del correr 1-127
Espera Modo < n >	0x96	0xC5	Tiempo de espera entre modos
Ciclos	0x97	0xC6	Se pone en marcha un programa en una fecha y hora
Linea < n >	0x98	0xC7	Nos situamos en la línea n de la pantalla
Programa < nombre >	0x99	0xC8	Ejecuta un programa dentro de otro programa
Sincronismo	0x9A	0xC9	Al principio de grupo de líneas (ejecución en paralelo)
No sincronismo	0x9B	0xCA	Al principio de un grupo de líneas (para ejecutarlas secuencialmente). Pone línea 0.
Correr	0x9C	0xE0	El texto se desplaza de derecha a izquierda
Centro	0x9D	0xE1	Modo de aparición
Disminuido	0x9E	0xE2	Modo de aparición
Apilado	0x9F	0xE3	Modo de aparición
Rodar	0xA0	0xE4	Modo de aparición
Baja	0xA2	0xE6	Modo de aparición
Color	0xA3	0xA1	Color del Texto 0 – Apagado 1 – Rojo 2 – Verde 3 – Ambar
Fondo	0xA4	0xA2	Fondo del Texto 0 – Apagado 1 – Rojo 2 – Verde 3 – Ambar
Nieve	0xA5	0xEF	Modo de aparición
Inmediato	0xA6	0xF0	Modo de aparición
Horas:MInutos	0xA7	0xA7	Muestra la hora en formato HH:MM
Temperatura ºC	0xA8	0xA8	Muestra la Temperatura con ºC
Inverso	0xA9	0xC2	A partir de aquí, todo sale invertido de color
Normal	0xAA	0xC3	Restablece si estaba invertido
Decimas	0xAB	0x9E	Decimas
Rotación EXT-CEN V	0xAC	0xE7	Rotación del texto Vertical
Rotación CEN-EXT V	0xAD	0xE8	Rotación del texto Vertical

Rotación EXT-CEN H	0xAE	0xE9	Rotación del texto Horizontal
Rotación CEN-EXT H	0xAF	0xEA	Rotación del texto Horizontal
Aparición derecha	0xB0	0xEC	Modo de aparición
Aparición izquierda	0xB1	0xEB	Modo de aparición
Aparición centro	0xB2	0xED	Modo de aparición
Aparición extremos	0xB3	0xEE	Modo de aparición
Idioma < n >	0xB4	0xCB	0 (Castellano) 1(Catalán) 2(Vasco) 3 (Gallego) 4(Francés) 5 (Inglés) 6(Portugués)
Deslizar	0xB5	0xF1	Modo de aparición
Girar	0xB6	0xF2	Futura ampliación
Sube	0xB8	0xE5	Modo de aparición
Animación <*nombre>	0xD0	0xCE	Ejecuta una animación
Luminosidad < n >	0xD8	0xD0	Ajusta la luminosidad de la pantalla del 1 al 100%.
No animación	0xDD	0xCF	Función obsoleta
No centro	0xDE	0xCD	Futura ampliación
Dia abreviado	0xE3	0xA9	Muestra el nombre día de forma abreviada.
Mes abreviado	0xE6	0xAA	Muestra el nombre del mes de forma abreviada.
Beep < n,ton,toff >	0xBA	0xB4	Produce n pulsos de <ton> tiempo activo y <toff > timpo de inactivo
Automático	0xFF	0xF4	Ejucuta un programa con el texto combinando efectos y colores de forma aleatoria
Iris	0xFD	0xF5	Realiza un efecto de cambio de color por letras
DibAnim	0xF0	0xF6	Aparece el texto seguido deun dibujo animado
Dibug	0xFB	0xD2	Tipo de dibujo para dibanim
Diferencia de horas* (se envian 2 bytes)	0xC9	0xAE,0x9A	Horas para la fecha evento
Diferencia de minutos	0xC8	0xAC	Minutos para la fecha evento
Diferencia de segundos	0xCD	0xAD	Segundos para la fecha evento
Resto de dias * (se envian 2 bytes)	0xCC	0xAE, 0x99	Dias para la fecha evento
Resto de horas * (se envian 2 bytes)	0xDA	0xAE, 0x98	Resto de horas para la fecha evento
Resto de minutos * (se envian 2 bytes)	0xD9	0xAE, 0x97	Resto de minutos para la fecha evento
Resto de segundos * (se envian 2 bytes)	0xF7	0XaE, 0x96	Resto de minutos para la fecha evento
Variable	0xDF	0x75	Variable alfanumerica
Ventana	0xD7	0xd3	Definir una ventana en la pantalla

(*) Los códigos para los tiempos Resto de días, Resto de horas, Resto de minutos y Resto de segundos se tienen que enviar después del código 0xAE.

- **Tabla de códigos del protocolo D.T.P.**

CODIGO	PANT	DESCRIPCIÓN
RESET	0x01	Inicializa el panel borrándolo todo.
WDRESET	0x02	Inicializa el panel borrándolo todo.
STOP	0x03	Para la ejecución de cualquier programa
EJECUTAR	0x04	Ejecutar un programa.
BORRAR	0x05	Borra un programa
EDITAR	0x06	Editar un programa
CHECKSUM	0x07	Pide el byte bajo del último checksum correcto
BAUDS	0x08	
GETPROG	0x09	Pide el programa (le siguen 8 caracteres)

SETHORA	0x0A	Pide el RTC en hora real (6 bytes: año, mes, día, hora, min, seg).
GETHORA	0x0B	Pide la fecha y hora del RTC, devuelve 6 bytes:año, mes, día, hora, min, seg.
ENVIAR txt	0x0C	Envía bloque de datos
GETDIR	0x0D	Pide el directorio del panel (**Obsoleta**)
PASSWORD	0x0E	Da el password (8 caracteres) - sólo en la versi´ñon con modem
SETPASSW	0x0F	Pone nuevo password (8 caracteres)
SETTEMP	0x10	Pone nuevo offset a la temperatura (opcional)
GETTEMP	0x11	Pone al panel offset y temperatura (Opcional)
GETVER	0x12	Pide la versión del software, tipo de panel, etc
GETEXEC	0x13	Pide el nombre del programa en ejecución
GETVIDEO	0x14	Pide el bitmap que sale actualmente por el panel (sólo para verificar la ejecución ya que el bitmap se modifica durante la transmisión).
GETSTAT	0x15	
EDITFILE	0x16	Permite enviar gráficos, animaciones y actualizar el software del panel (futura ampliación).
FLASHOTP	0x17	
GETVECT	0x18	
NGETDIR	0x1E	Pide el directorio del panel
NEJECUTAR	0x1F	
GETFILE	0x20	-Permite recuperar del panel cualquier archivo en forma de paquetes de tamaño variable, .- no soportado en la serie D.I.-.
GETNPAC	0x21	Devuelve el nº del paquete que toca recibir.
TODED	0x22	Reservada
FROMDED	0x23	Reservada
GETDED	0x24	Reservada
PUTCNT	0x25	Establece un contador
GETCNT	0x26	Solicita un contador determinado
FASTEXEC txt	0x27	Ejecuta el programa que se envia de forma inmediata

Códigos de error.

¡Error! Marcador no definido. 0	OK (operación efectuada correctamente)
1	Archivo no encontrado
2	No puede editar (se está ejecutando un prog.)
3	Archivo no creado (inexistente)
4	No queda memoria de programas
5	Archivo en ejecución
6	No permitido en ejecución
7	Orden desconocida
8	Archivo sin contenido (no se puede ejecutar)
9	Password incorrecto
10	Nombre demasiado largo
11	Formato de fecha incorrecto
12	Formato de EDITFILE incorrecto
13	No hay memoria en el *heap* para 'files'
14	'file' mayor de FFF0H bytes
15	No está permitido ejecutar 'file'
16	Productos diferentes
17	EDITFILE no soportado
18	Configuración incorrecta
19	Configuración del checksum
20	Uso interno
21	Uso interno
22	**pos** de GETFILE superior a tamaño archivo
23	No permitido en OTP
24	Baudios incorrectos

25	Datos no válidos
26	Realizando test de Pixel
27	El proceso de test no ha terminado
28	Dispositivo ocupado

APÉNDICE B: ESTRUCTURAS Y CONSTANTES

ESTRUCTURA USADA EN LA ORDEN NGETDIR

```
struct nom_arx{
                    char *pos;              */ Posición escritura archivo          */
                    unsigner int size;  */ Tamaño del archivo                      */
                    char nom[9];            */ Nombre del archivo terminado en 0   */
                    char *posbuf;           */ Posición escritura EDITFILE         */
                    unsigned int sizeact;
                    unsigned int sizebuf:   */ Tamaño del malloc EDITFILE          */
                };
```

ESTRUCTURAS USADAS EN LA ORDEN EDITFILE

```
struct EditFile {
                    unsigned long len;      /* Longitud archivo (menor de FFF0H )     */
                    unsigned char tipo;             /* Tipo del archivo (ver tipos)
                                                                                    *
                                                                                  /
                };
        struct cap_gmp
                {
                unsigned char tipo;
                unsigned char compresion;
                unsigned int x;
                unsigned int y;
                unsigned int ancho;
                unsigned int alto;
                unsigned int frames;
                unsigned int is_conv;
                unsigned char ncolores;
                unsigned char ntonos;
                unsigned char peed;
                unsigned char paginas;
                unsigned char indeterminado[14];
                } cap;
```

ESTRUCTURAS USADAS EN LA ORDEN GETFILE

```
/* Estructura de la cabecera de adquisiciones de ficheros y programas */
struct GetFile {
                    unsigned long pos;              /* Posición respecto al inicio    */
                    unsigned long len;      /* Longitud  solicitada                   */
            };
```

TIPOS DE ARCHIVOS

```
#define GRAF    0x00    /* Archivo gráfico                    */
#define PROG    0x01    /* Archivo de actualización software */
#define FONTS   0x02    /* Archivo de fonts (tipos de letra)  */
```

APÉNDICE C: SET DE CARACTERES

La gran mayoría coinciden con el código ASCII.

¡Error! Marcador no definido.COD	CAR	COD	CAR	COD	CAR	COD	CAR	COD	CAR	COD	CAR	COD	CAR
SP	20H	0	30H	º	40H	P	50H	ó	60H	p	70H	à	80H
!	21H	1	31H	A	41h	Q	51H	a	61H	q	71H	è	81H
"	22H	2	32H	B	42H	R	52H	b	62H	r	72H	i	82H
Ñ	23H	3	33H	C	43H	S	53H	c	63H	s	73H	ò	83H
$	24H	4	34H	D	44H	T	54H	d	64H	t	74H	ù	84H
%	25H	5	35H	E	45H	U	55H	e	65H	u	75H	ä	85H
&	26H	6	36H	F	46H	V	56H	f	66H	v	76H	ë	86H
'	27H	7	37H	G	47H	W	57H	g	67H	w	77H	ï	87H
(	28H	8	38H	H	48H	X	58H	h	68H	x	78H	ö	88H
)	29H	9	39H	I	49H	Y	59H	i	69H	y	79H	ü	89H
*	2AH	:	3AH	J	4AH	Z	5AH	j	6AH	z	7AH	↑	8AH
+	2BH	;	3BH	K	4BH	Ç	5BH	k	6BH	é	7BH	¡	8BH
'	2CH	←	3CH	L	4CH	\	5CH	l	6CH	ú	7CH		
-	2DH	=	3DH	M	4DH	ç	5DH	m	6DH	í	7DH		
.	2EH	→	3EH	N	4EH	↓	5EH	n	6EH	ñ	7EH		
/	2FH	?	3FH	O	4FH	¿	5FH	o	6FH	á	7FH		

APÉNDICE D: EJEMPLOS DE PROGRAMACIÓN

Sincronizar líneas con multilineas

SYNC
LINEA1
CORRER TEXTTO1
LINEA2
CORRER TEXTTO2
NOSYNC

Siempre que se inicie un bloque SYNC deberá finalizarse con nosync

Cambio de color en una línea

CORRERCOLOR POR DEFECTO**COLOR**2 VERDE **COLOR**3 AMARILLO
INMEDIATO COLOR3 ATENCIÓN**COLOR**1 100% DTO.

Ejecución de una animación en una pantalla gráfica

CORRER A continuación ejecutamos una animación
ANIM*hands

APÉNDICE E: LA ORDEN FASTEXEC

La función FASTEXEC se ha implementado en los diferentes productos en diferentes versiones. Para determinar si su producto soporta esta propiedad deberá comprobar el modelo y versión de su producto y contrastarlo en la siguiente tabla

Tabla compatibilidad para FASTEXEC

Serie Pantalla	Placa Control	Versión	Fecha
COL	CO31MUL	7.0	2-6-97
DI	CO31MUL	5.4	30-3-99
ED	CO31	5.4	5-2-99
ONLY	CO31AL	5.3	30-1-99
MN	CO31	**TODOS**	**TODOS**
TR40	TEC	1.9	15-3-99
V4	DIAH8	7.6	8-5-97
ML	H8MULTI	7.4	6-5-97
MLC	MULTICOL	8.2	19-11-97
PGM	PGM	**NO**	**NO**

APÉNDICE E: LA INSTRUCCIÓN VENTANA

DESCRIPCIÓN:
Mediante el dato VENTANA se pueden configurar porciones de la pantalla que ocupen menos que el tamaño total de la misma. Por ejemplo una pantalla de 160 pixels de ancho puede partirse en dos de 80.

Ventana A,X1,Y1,X2,Y2. A, es el identificador de la ventana, y ha de ser un carácter en mayúsculas entre A y P (sin la Ñ), lo que permite un máximo de 16 ventanas. X1,Y1 es la posición de la esquina superior izquierda Y1 de la pantalla medido en líneas, X1 medido en pixels, X2,Y2 es la posición de la esquina inferior derecha de la pantalla medida de la misma forma.

Una vez se ha declarado una ventana, todas las instrucciones que haya a continuación se referirán a la ventana en concreto menos el dato LINEA, que mantendrá su independencia y se conservará como hasta ahora (refiriéndose a líneas de la totalidad de la pantalla). Esta situación se mantendrá hasta que se declare otra ventana o se ponga LINEA 0, que indica que vamos a trabajar con toda la pantalla (sin ventanas).

Una vez se haya declarado una ventana, será posible referirse a ella indicando solo el indicador de la misma, sin incluir las posiciones de inicio y final. Ejemplo:

Ventana A

Observaciones

Una instrucción LINEA después de una instrucción VENTANA, nunca se referirá a una línea de la ventana, sino a una línea de la pantalla. Para direccionar partes de una ventana declare otra ventana.
Se considera ilegal que dentro de un sincronismo coincidan instrucciones LINEA referidas a VENTANAS y instrucciones LINEA normales que comparten la misma región de la pantalla.
También se considera ilegal que dos VENTANAS compartan un área de la pantalla.
Al igual que sucedía con las líneas hasta ahora, las ventanas no pueden sobrepasar físicamente el tamaño de la pantalla, en cuyo caso la instrucción será ignorada.
Siempre que no estén dentro del mismo grupo de instrucciones SYNC/NOSYNC, el identificador de las ventanas puede repetirse para facilitar la edición. Provocando como resultado que un mismo identificador se pueda usar para redeclarar una o varias ventanas.

Ejemplo 1

Supongamos que tenemos una pantalla GTM 160-96 (160 pixels de ancho y 96 de alto, 12 líneas), y queremos partirla en 2 ventanas de 80 pixels de ancho y 8 líneas de alto (en este caso cada línea tiene 8 pixels ya que la pantalla está constituida por matrices de LEDs de 8 pixels de altura), y otras dos ventanas de 80 pixels por 4 líneas. Cada una de las ventanas ha de mostrar distintos textos
Para la instrucción VENTANA, los pixels y líneas empiezan por 1, y la posición 1,1 es la esquina superior izquierda de la pantalla; la posición de la esquina inferior derecha es la 160,12 (pixels 160, línea 12). En la ilustración se puede apreciar la distribución de las ventanas en pixels, para poder comparar con la descripción de las 4 ventanas.

EJEMPLO WinEdit:

```
Ventana A,1,1,80,8                  Ventana A, de 80 píxels x 8 líneas
Correr PRUEBA VENTANA A
Ventana B,81,1,160,8      Ventana B, de 80 píxels x 8 líneas
Correr PRUEBA VENTANA B
Ventana C,1,9,80,12                 Ventana C, de 80 píxels x 4 líneas
Correr PRUEBA VENTANA C
Ventana D,81,9,160,12     Ventana D, de 80 píxels x 4 líneas
Correr PRUEBA VENTANA D
Ventana A                                   Ventana A, no es necesario dar la resolución si no
                                            se desea cambiar la ventana.
Correr VENTANA A YA DECLARADA
```

EL DATO VENTANA CON EL DATO SYNC

Aunque es teóricamente posible declarar las ventanas en la cabecera del programa y luego referirse a ellas mediante su identificador, dentro de las instrucciones SYNC/NOSYNC, en la práctica se ha visto que hay un pequeño retraso que afecta principalmente a los parpadeos, y que puede subsanarse poniendo la declaración de cada ventana dentro de la zona SYNC/NOSYNC.

Ejemplo 2

Partiendo de loas datos del ejemplo 1, supongamos que se desea mostrar un texto en cada una de las líneas de la ventana B simultáneamente, mediante el dato SYNC. Para ello hay que crear una nueva ventana para cada línea, no siendo necesaria la declaración de la ventana B, a menos que se desee mostrar alguna cosa en la totalidad de B. Para evitar confusiones, en el siguiente ejemplo no hemos usado el identificador B para ninguna ventana, aunque podría ponerse sin problemas.

EJEMPLO WinEdit:

```
SYNC
Ventana E,81,1,160,1                Primera línea ventana B
Correr PRUEBA LINEA 1 VENTANA B
Ventana F,81,2,160,2                Segunda línea ventana B
Correr PRUEBA LINEA 2 VENTANA B
Ventana G,81,3,160,3                Tercera línea ventana B
Correr PRUEBA LINEA 3 VENTANA B
Ventana H,81,4,160,4                Cuarta línea ventana B
Correr PRUEBA LINEA 4 VENTANA B
Ventana I,81,5,160,5                Quinta línea ventana B
Correr PRUEBA LINEA 5 VENTANA B
Ventana J,81,6,160,6                Sexta línea ventana B
Correr PRUEBA LINEA 6 VENTANA B
Ventana K,81,7,160,7                Séptima línea ventana B
Correr PRUEBA LINEA 7 VENTANA B
Ventana L,81,8,160,8                Octava línea ventana B
Correr PRUEBA LINEA 8 VENTANA B
NOSYNC
```

Tabla compatibilidad para VENTANA producto standard

Serie Pantalla	Placa Control	Versión	Fecha
COL	NO	NO	NO
DI	NO	NO	NO
ED	NO	NO	NO
ONLY	NO	NO	NO
V4	DIAH82	9.6	1999
ML	H8EXMUL	--	2000
MLC	H8EXMUL	--	2000
PGM	H8EXMUL	--	2000
GTM	MULTICOL2	--	2002
GTM	H8EXMUL	--	2000
GTI	H8EXMUL	--	2000

LA INSTRUCCIÓN VENTANA EN LA PLACA MULTICOL2

Tiene las siguientes diferencias respecto a los otros productos:

Soporta 224 ventanas mientras que como máximo los otros productos soportan 16
Debido a que 224 ventanas no se pueden seleccionar de la 'A' a la 'Z', las ventanas se pueden direccionar numéricamente de 001 a 224 o por el método standard de 'A' a 'Z'.

Ejemplo Ventana 003,81,2,160,2.

Si se usan los dos sistemas al mismo tiempo se ha de tener en cuenta que Ventana A equivale a Ventana 001, y en caso que coexistan es posible que el contenido de una machaque al de la otra porque las dos numeraciones son equivalentes. Por esta razón se recomienda usar un solo sistema de numeración dentro del mismo sincronismo. En dos sincronismos diferentes pueden haber los 2 sistemas.

DESCRIPCION TECNICA MPBASIC VERSION 1.0

COMPATIBILIDAD PRODUCTOS																
ED	V4	PGM	CO	VT	DI	ML	COL	MLC	GTI	GTM	MGTM	R100CO	PCM	DED COM	DED DCF	MN
						V8.4										

TIPOS DE PAQUETES

GETVER2 (30H)

Da como contestación un paquete ENVIAR que contiene una estructura que determina:

byte 1 ..... Versión del software
byte 2 ..... Modelo de hard de la placa de control
byte 3 ..... Longitud línea en pixels (L)
byte 4 ..... Longitud línea en pixels (H)
byte 5 ..... Altura de la línea de caracteres (1)
byte 6 ..... Número de líneas de caracteres
byte 7 ..... Versión del FONT de caracteres
byte 8 ..... Versión del MPBASIC
byte 9 ..... Versión de PROGRAMAS
.....
byte n

¡ATENCION!, esta estructura no es de longitud fija sino que se puede ampliar en futuras versiones de software. Como en una misma línea de comuniaciones pueden coexistir varios modelos, es indispensable que se ponga a 0 el buffer donde se va a alojar la estructura, con lo que las estructuras más cortas tendrán a 0 (que es el valor por defecto) los campos aún no implementados. Si la versión del MPBASIC es 0, significa que no está implementado en la pantalla. El valor actual de la versión del MPBASIC es 10 (1.0)

EX GETVER2:

16 07 00 01 30 4E 00

PUTVARS (2EH)

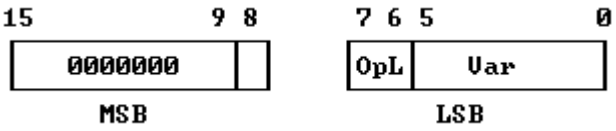
Pone variables en la memoria de la pantalla para que se puedan mostrar con el DATO 'VAR'. El paquete se puede enviar en cualquier momento, esté la pantalla en ejecución o no, y está formado por un número de 1 a 26 estructuras del tipo *PUTVARS*, que contienen 10 bytes cada una. Aunque en la siguiente explicación los bytes se describan de la forma habitual (el más significativo a al izquierda), hay que tener en cuenta que para enviarlos a la pantalla se transmite primero el byte menos significativo (LSB).

bytes [0,1]

6 bits ..... Número de variable [0,25] (26 variables, sin la 'Ñ'). 0='A', 1='B',.., 25='Z'
3 bits ..... Operación a realizar con el valor en coma flotante o string de caracteres.

0 = Asignación de string de chars
1 = Asignación del valor en coma flotante
2 = Sumar el valor a la var y asignar a la var (var = var + valor)
3 = Restar el valor a la var y asignar a la var (var = var - valor)

7 bits ..... Ha de ser 0



bytes [2-9]

valor ..... Valor en coma flotante a asignar u operar con la variable. El número tiene el formato IEEE de 64 bits. O bien string de caracteres terminado en 0 si es menor de 8 (opcional).

RESTRICCIONES

Aunque el sistema es muy versátil, no está permitida la mezcla de números y strings. Así se considerará ilegal y no se responderá del resultado en los siguientes casos:

- Sumar o restar un número a un string almacenado en la pantalla, en estos casos el string se considera un 0, por lo que por ejemplo si se incrementa en 1 un string, el resultado será el número 1.
- Los strings no pueden sobrepasar los 8 bytes, aunque pueden ser más cortos, poniendo un 0 (cero hexadecimal) al final del mismo, aunque no es necesario, pero puede servir para formatear la salida en la pantalla.
- Es ilegal enviar el paquete sin ninguna estructura.
- Es ilegal enviar un paquete con cualquier variable repetida (varias estructuras que se refieran a la misma variable).

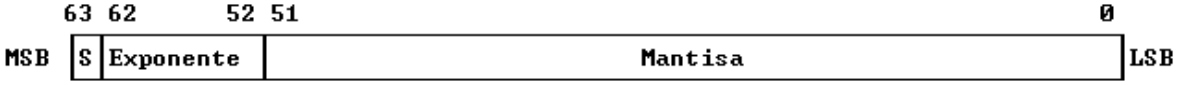
CONSEJOS

Para aumentar el número de transmisiones por segundo, actualice solo las variables imprescindibles, con ello reducirá el tamaño de los paquetes *PUTVARS* y evitará que se retrase mucho el *ACK* debido al tiempo de proceso en coma flotante de 64 bits. Pueden inicializarse todas las variables de una vez, al principio y luego actualizar variable a variable.

NÚMEROS EN COMA FLOTANTE DE 64 BITS

Los números de coma flotante con el formato IEEE de 64 bits, permite una resolución de 16 dígitos (sumando los de antes y después de la coma) sin pérdida de precisión, y están formados por una mantisa que comprende los 52 bits menos significativos, los siguientes 11 son el exponente y el último al signo. Representándose un número como:

Nº = -1^S * 2^(E-1023) * 1.MANTISA



El número 0 se representa con 8 bytes a 0.
Para calcular el valor de un número en coma flotante de 64 bits como los que hay en los ejemplos de este documento, a menudo nos encontramos con que el número de dígitos que soportan las calculadoras hexadecimales es menor que lo que necesitamos, por lo que expondré una forma alternativa de calcularlo. Ahora bien, otra forma de representar el número en coma flotante es:

N = -1^S * 2^(E-1023) * (1 + MANTISA*2^(-52))

Supongamos que enviamos a la pantalla el número 00 00 00 00 74 5D 40 41, que ya está en el formato de salida, es decir, primero el byte menos significativo. El exponente con el signo (en este caso positivo) lo forman los 3 nibbles más altos: 414H. Basándonos en que el valor 1023 es 3FFH, e^(E-1023) da un resultado de 2^(414H-3FF) = 2^21.
La mantisa tendrá el valor 05D7400000000H, lo que es igual a decir 5D74H * 2^32. Por tanto:

N = 2^21 * (1 + 5D74H*2^32*2^(-52))
N = 2^21 + 2^21*5D74H*2^32*2^(-52)
N = 2^21 + 5D74H*2^1
N = 2145000

Para calcular un número en coma flotante a partir de un entero se debe hacer la mantisa cuanto más grande, mejor y luego ajustar el exponente.

EJEMPLO:

En el siguiente ejemplo, asignamos:

1. El string 'PRODUCTO' en la variable A
2. El número 1 sumado a la variable B
3. El número 2145000 en la variable C
4. El número 13406.25 en la variable D
5. El número 1 restado a la variable E
6. Ponemos las demás variables a 0 (no es necesario ya que está permitido enviar desde 1 a 26 estructuras, pero sirve para el ejemplo)

```
16 0C 01 01 2E 00 00 50 52 4F 44 55 43 54 4F 81 00 00 00 00 00 00 00 F0 3F
42 00 00 00 00 00 74 5D 40 41 43 00 00 00 00 00 20 2F CA 40 C4 00 00 00 00
00 00 00 F0 3F 45 00 00 00 00 00 00 00 00 00 46 00 00 00 00 00 00 00 00 00
47 00 00 00 00 00 00 00 00 00 48 00 00 00 00 00 00 00 00 00 49 00 00 00 00
00 00 00 00 00 4A 00 00 00 00 00 00 00 00 00 4B 00 00 00 00 00 00 00 00 00
4C 00 00 00 00 00 00 00 00 00 4D 00 00 00 00 00 00 00 00 00 4E 00 00 00 00
00 00 00 00 00 4F 00 00 00 00 00 00 00 00 00 50 00 00 00 00 00 00 00 00 00
51 00 00 00 00 00 00 00 00 00 52 00 00 00 00 00 00 00 00 00 53 00 00 00 00
00 00 00 00 00 54 00 00 00 00 00 00 00 00 00 55 00 00 00 00 00 00 00 00 00
56 00 00 00 00 00 00 00 00 00 57 00 00 00 00 00 00 00 00 00 58 00 00 00 00
00 00 00 00 00 59 00 00 00 00 00 00 00 00 00 15 25 10
```

16	SYN
0C 01	Longitud (010CH)
01	Nº pantalla
2E	*PUTVAR*
00 00	Variable A string
50 52 4F 44 55 43 54 4F.....	'PRODUCTO'
81 00	Variable B suma
00 00 00 00 00 00 F0 3F	1

42 00 ..... Variable C asignación
00 00 00 00 74 5D 40 41 2145000
43 00 ..... Variable D asignación
00 00 00 00 20 2F CA 40 ..... 13406.25
C4 00 ..... Variable E resta
00 00 00 00 00 00 F0 3F 1
45 00 ..... Variable F asignación
00 00 00 00 00 00 00 00
46 00 ..... Variable G asignación
00 00 00 00 00 00 00 00
47 00 ..... Variable H asignación
00 00 00 00 00 00 00 00
48 00 ..... Variable I asignación
00 00 00 00 00 00 00 00
49 00 ..... Variable J asignación
00 00 00 00 00 00 00 00
4A 00 ..... Variable K asignación
00 00 00 00 00 00 00 00
4B 00 ..... Variable L asignación
00 00 00 00 00 00 00 00
4C 00 ..... Variable M asignación
00 00 00 00 00 00 00 00
4D 00 ..... Variable N asignación
00 00 00 00 00 00 00 00
4E 00 ..... Variable O asignación
00 00 00 00 00 00 00 00
4F 00 ..... Variable P asignación
00 00 00 00 00 00 00 00
50 00 ..... Variable Q asignación
00 00 00 00 00 00 00 00
51 00 ..... Variable R asignación
00 00 00 00 00 00 00 00
52 00 ..... Variable S asignación
00 00 00 00 00 00 00 00
53 00 ..... Variable T asignación
00 00 00 00 00 00 00 00
54 00 ..... Variable U asignación
00 00 00 00 00 00 00 00
55 00 ..... Variable V asignación
00 00 00 00 00 00 00 00
56 00 ..... Variable W asignación
00 00 00 00 00 00 00 00
57 00 ..... Variable X asignación
00 00 00 00 00 00 00 00
58 00 ..... Variable Y asignación
00 00 00 00 00 00 00 00
59 00 ..... Variable Z asignación
00 00 00 00 00 00 00 00
15..... Control transmisión
25 10..... Checksum 1025H

TRANSMISION

Hay que tener en cuenta que siempre transmitiremos primero el byte menos significativo (LSB). Al final de las estructuras hay un byte de control que realmente no se lee, sino que solo se usa para reforzar el sistema de comunicaciones, ya que en caso de que un paquete corto no se recibiera correctamente, el PC debería preguntar a la pantalla el checksum del último paquete correcto recibido, y si fueran diferentes, intentar un reenvío. Si el último paquete con ACK recibido fuera idéntico al que no fue recibido (cosa que suele suceder), el PC, al pedir el último checksum, comprobaría que es igual al del paquete no recibido, y por lo tanto consideraría que el paquete fue recibido correctamente pero no así el ACK, con lo que no se produciría el reenvío aún cuando el paquete fue recibido incorrectamente por la pantalla.
Este efecto no tiene importancia en operaciones de asignación, pero sí en todas las otras.
Para evitar estos caso se ha previsto que al final de la/s la estructura/s haya un byte cuyo valor debería ser cambiado en cada nueva transmisión. Se recomienda un contador que se incremente cada vez en 15H respecto el valor que tenía en la última transmisión.
Con el byte de control cambiando de un paquete al siguiente, ya no habría posibilidad que se confundieran los checksum de dos paquetes idénticos y consecutivos.

Por otra parte es imprescindible tener en cuenta la recepción del ACK y el sistema de reenvío descrito en el DTP (ver el manual del DTP), ya que en caso de una mala recepción del ACK, podría suceder que una misma operación se realizara tantas veces como reenvíos haya en una pantalla, mientras que debería haberse hecho una sola vez.

GETVARS (2FH)

Pide variables de la memoria de la pantalla que le son enviadas mediante un paquete ENVIAR que contiene 26 estructuras, las cuales corresponden a las variables ordenadas de la 'A' a la 'Z'. Cada una de ellas tiene 10 bytes con el siguiente formato:

bytes [0,1]

1 bit ..... Indica si es string.
15 bits ..... reservado

bytes [2-9]

valor ..... Valor que tiene la variable. Puede ser un número con el formato IEEE de 64 bits o bien string de 8 caracteres o menos, que puede no terminar en 0.

EX GETVARS:

16 07 00 01 2F 4D 00

DATOs IMPLICADOS

VAR (ABH)

Para representar las variables hace falta insertar el DATO VAR y después el nombre de una de las 26 variables [A,Z] (en mayúscula y sin incluir la Ñ) sin ningún espacio en medio. Las variables se ponen a 0 si se les asigna este valor o cuando se inicializa la pantalla. En caso de apagar, o desconectar la corriente,

las variables conservarán su valor mientras dure la batería interna (supercap). Tampoco se alterarán si se para o vuelve a empezar la ejecución. Las variables tienen una precisión de 16 dígitos, es decir, se pueden mostrar 16 dígitos (sumando los de antes y después de la coma) sin pérdida de precisión, en caso que se muestren más de 16, los dígitos menos significativos diferirán del valor real. Esto último es inherente al formato de coma flotante IEEE de 64 bits.

EX:

SYNC
LINEA 1
INMEDIATO VARA VARB
LINEA 2
INMEDIATO VARC PTAS.
LINEA 3
INMEDIATO VARD EUROS
NOSYNC

Los paquetes del DTP que se usan para editar el programa que hemos llamado *PRUEBA* son:
Editar 'PRUEBA':

```
16 0D 00 01 06 50 52 55 45 42 41 E9 01
```

Enviar el contenido del archivo:

```
16 27 00 01 0C C9 C7 31 F0 AB 41 20 AB 42 C7 32 F0 AB 43 20 50 54 41 53 2E C7
33 F0 AB 44 20 45 55 52 4F 53 CA A2 0E
```

PROGRAMACION AVANZADA

Como el formateo por defecto de una variable es con 6 dígitos después de la coma, puede ser incómodo representar números enteros con tantos decimales. Se puede formatear la variable añadiendo el número de dígitos totales y el número de dígitos después de la coma, de la forma:

EX:

VAR6.2B ..... Si la variable vale por ejemplo 1, el resultado será: __1.00
(los dos underscores representan espacios)

VAR9.0 ..... _____1

VAR09.0 ..... Se puede ajustar a la izquierda con ceros en vez de espacios,
colocando un cero 000000001

VAR+9.0 ..... También se puede indicar la representación con signo _____+1

VAR-9.0 ..... Poniendo el signo menos se ajusta por la izquierda 1_____

VAR9B ..... Puede ponerse solo el ajuste de antes de la coma _____1

VAR.9B ..... También puede ponerse solo el ajuste de después de la coma
1.000000000

En principio, una variable a la que se le ha asignado un número, puede ser substituida por un string mediante un paquete *PUTVARS*. Si el formateo es correcto, no tiene porque haber corrimiento.
Por ejemplo: a la variable X se le ha asignado el número 456342 mediante un paquete *PUTVARS*. INMEDIATO VAR8.0Z mostrará el número __456342 (con dos espacios de justificación a la izquierda que suman los 8 caracteres). Si mediante otro paquete *PUTVARS* asignamos el string 'PARO' a la variable Z, en la pantalla se mostrará ____PARO (con 4 espacios de justificación a la izquierda que suman los 8 caracteres) con lo que no habrá variaciones de justificación y tanto el número como el string se mostrarán en el mismo lugar de la pantalla. Se debe ir con mucho cuidado si se usa este sistema ya que si se envía un paquete *PUTVARS* que por ejemplo decremente en 1 un string (acción ilegal), el string será considerado como el número 0, con lo que el resultado será el número -1.

Los números se redondean a partir de la primera cifra visible; es decir, si internamente la variable vale 3.141592 y nosotros mostramos cuatro decimales (EX: VAR.4X), como el primer número invisible (el 9) es superior a 4, entonces se redondea el 5 a 6, mostrándose en pantalla 3.1416.

RESTRICCIONES

- Es ilegal representar números de más de 16 caracteres sin contar los símbolos (como la coma, etc.) ya que habría pérdida de precisión.
- Es ilegal formatear la variable para más de 22 caracteres, contando todos los símbolos que haya, el signo positivo, la coma, etc.
- Si se colocan más de 8 caracteres de formateo (entre el DATO VAR y la variable), éstos son despreciados, en la pantalla no se muestra la variable, sino el símbolo '---' (tres guiones), también se muestran los caracteres que sobran del formateo.

CONSEJOS

Como que se trata de mostrar una información en coma flotante de alta precisión en la pantalla, cuanto más variables se escriban, más tiempo tardará la pantalla en completar la información, como resultado de lo cual, puede darse el caso que se retrasen algunos modos de aparición o se produzcan brusquedades. Procure utilizar siempre el modo *INMEDIATO* o cualquier otro que deje la información estática en pantalla. Evite utilizar *HORMIN* o *BLINK* , ya que el carácter que parpadea puede quedar estático más tiempo de lo normal.
En caso que sea imprescindible mostrar *HORMIN* o cualquier otra información sensible al tiempo de proceso, considere la opción de mostrar alguna de las variables en formato string, es decir, calculando los caracteres que han de aparecer en pantalla antes de ser enviados en un paquete *PUTVARS*.

ERRORES

Cuando se usan tantas opciones, a veces es fácil equivocarse. Si la pantalla no reconoce la variable que debe mostrarse, entonces aparecerán tres guiones '---' en el sitio donde debería verse la variable. También podría ser que hayan más de 8 caracteres de formateo. Vuelva a editar el programa corrigiendo el problema y ejecútelo.

CARACTERES DE SELECCION PARA EL TECLADO

VARIABLE (DATO)

Idioma	**Texto**	**Carácter selección**
CASTELLANO	Var	V
FRANCES	Var	V
INGLES	Var	V
PORTUGES	Var	V

CARACTERES DE SELECCION PARA IBM1050

No es aconsejable mezclar los dos protocolos con paquetes de distintos tipos
Consultar el manual del protocolo IBM1050, donde se describen las funciones F1, F2, F3, F4 y sus valores en hexadecimal. F4 tiene el valor 1FH.

F4+'V'

EX:

LINEA 1
INMEDIATO VARA
LINEA 2
INMEDIATO VARB
LINEA 3
INMEDIATO VARC

```
04 41 31 02 1F 6C 20 20 31 1D 49 1F 56 41 1F
6C 20 20 32 1D 49 1F 56 42 1F 6C 20 20 33 1D
49 1F 56 43 17 36 37 02 17 31 39
```

EX:

SYNC
LINEA 1
INMEDIATO VARA VARB
LINEA 2
INMEDIATO VARC PTAS.
LINEA 3
INMEDIATO VARD EUROS
NOSYNC

```
04 41 31 02 1F 53 1F 6C 31 1D 49 1F 56 41 20
1F 56 42 1F 6C 32 1D 49 1F 56 43 20 50 54 41
53 2E 1F 6C 33 1D 49 1F 56 44 20 45 55 52 4F
53 1F 6E 17 42 33 02 17 31 39
```