Salida por pantalla y tarjetas de vídeo

Como elemento de salida primario la pantalla representa la conexión entre el usuario y los diferentes programas, que envían sus salidas a través de las llamadas tarjetas de vídeo a la pantalla. Una parte importante del esfuerzo de desarrollo se va al crear la mayoría de programas en la creación de rutinas, con las cuales se puedan hacer visibles informaciones en la pantalla. No siempre uno tiene que trabajar directamente con la tarjeta de vídeo, ya que en el ROM-BIOS existen toda una serie de rutinas preparadas para el acceso a la pantalla. Pero a la mayoría de programadores, estas rutinas les resultan demasiado lentas, y por ello prefieren programarlas ellos mismos, controlando directamente el hardware de vídeo. No es una tarea fácil, si se piensa que en el ámbito del PC hoy por hoy hay seis estándares de vídeo, con diferentes características de capacidad, y nunca se puede saber qué tipo de tarjeta de vídeo esté presente en el ordenador a la hora de ejecutar el programa.

Este capítulo trae un poco de luz a la penumbra de los diferentes estándares de vídeo, describe sus características, las rutinas BIOS para la salida por pantalla y la programación directa de las diferentes tarjetas de vídeo. Hemos puesto un énfasis especial en la programación de las tarjetas Super VGA, ya que hoy por hoy siguen siendo las más difundidas entre todos los tipos de tarjetas gráficas. En relación a estas tarjetas no solo se tratan todos los secretos de la programación gráfica, sino también el tema del tratamiento de la programación de «Sprites», que debería se muy interesante para todos los aficionados a las animaciones gráficas.

Pero todos los estándares gráficos anteriores como MDA, CGA y Hercules (HGC) también se tratan, al igual que las nuevas tarjetas Super-VGA. Pero para comenzar, hablaremos de la historia y las características de los diferentes estándares de vídeo en el ámbito del PC.

Historia y características

Aparte de las velocidades de ordenador cada vez crecientes, el progreso de la tecnología de hardware no se actualiza en ningún campo tan clara y rápidamente como en las tarjetas de vídeo. Pero aquí, el desarrollo no solo trae tarjetas de vídeo más rápidas, sino sobre todo más capaces, que superan a sus antecesores claramente en número de colores representables y resolución de pantalla.
Mientras que hasta ahora se han conseguido mejoras cuantitativas primarias, basándose en tecnologías existentes, la moda en los últimos tiempos va en otra dirección. Cada vez más tarjetas gráficas se equipan con procesadores inteligentes, que le quitan al procesador principal el trabajo de dibujar líneas, círculos y superficies. Por ello le queda más tiempo para su tarea principal, que es la de preparar informaciones y la interacción con el usuario. Especialmente ahora, en el umbral de la época de los entornos gráficos, esta tecnología tiene una gran importancia, ya que estos sistemas requieren grandes esfuerzos del procesador. Si se quiere que las aplicaciones no se ralenticen a ojos del usuario, se ha de descargar de trabajo al procesador, y precisamente esto lo realizan la mayoría de las tarjetas gráficas con el popular chip S3 u otro procesador gráfico.

Los siguientes apartados aclararán el desarrollo en este sector del hardware, apoyándose en las modificaciones históricas, y dan una visión de las características de las diferentes clases de tarjetas de vídeo. Además de ello, se muestran las perspectivas de futuro, ya que en el punto tarjetas de vídeo las cosas no han llegado aún a su final..

MDA

El IBM Monochrome Display Adapter, abreviado MDA, representa, junto con la tarjeta CGA, a uno de los adaptadores gráficos más antiguo que está disponible para PC. En 1981 se presentó al público junto con el primer ordenador personal de IBM, y durante varios años se consideró como el estándar en tarjetas de vídeo monocromas.

La tarjeta MDA soporta simplemente un modo de funcionamiento, en el que aparece 25 líneas con 80 columnas cada una en pantalla. Al contrario que muchos otros adaptadores de vídeo dispone de muy poca RAM de vídeo, de modo que sólo se puede tener una página de pantalla en memoria.
A pesar de que con esta tarjeta no se pueden crear gráficos, muchos usuarios la prefirieron a la tarjeta CGA, que en aquellos años representaba la única alternativa. Comparada con la tarjeta CGA, la tarjeta MDA tiene una resolución mucho mayor, de modo que el trabajo con esta tarjeta cansa mucho menos los ojos de los usuarios, que las tarjetas CGA.

Hoy en día apenas quedan tarjetas MDA puras en funcionamiento, sobre todo porque hace cierto tiempo que ya no se producen. En el ámbito de las tarjetas de pantalla monocromas ya han hecho lugar a las tarjetas Hercules, que disponen de todos los atributos de las tarjetas MDA, y que además de ello pueden representar gráficos monocromos. Por ello se presentan como una alternativa real a los diferentes tipos de tarjetas gráficas en color.

CGA

Igualmente del año 1981 es el estándar CGA, cuyo acrónimo viene de Color Graphics Adapter. Como alternativa a las tarjetas MDA, esta tarjeta le ofrecía al usuario, ya en los tempranos días de los PC, la posibilidad de crear gráficos; aunque, comparando con las condiciones actuales, a un precio increíble.

Quien gastó sus ahorros comprando una tarjeta CGA, al menos podía prescindir de comprarse un monitor, ya que las tarjetas CGA disponen de una salida especial, que permitía su conexión a una televisión normal. Además también disponen de una salida RGB, es decir una línea, en la aparte de las señales de sincronismo, se dividía el color de un punto de pantalla en sus partes proporcionales de Rojo, Azul y Verde. La imagen creada, comparada con las tarjetas MDA, tiene una calidad inferior, lo que no solo se debe a la resolución menor, sino a que la distancia entre puntos del monitor CGA era mayor.

La tarjeta CGA, al igual que la tarjeta MDA representa en modo texto 25 líneas y 80 columnas en la pantalla, pero los diferentes caracteres se basan en una matriz de puntos más pequeña que en el caso de las tarjetas MDA. Pero a cambio se pueden representar gráficos de 320 × 200 puntos, donde la posibilidad de elección de color queda muy limitada, con tan solo cuatro. En el modo de mayor resolución incluso sólo quedan 2 colores de los que disponemos para construir la pantalla.

A pesar de que las características de una tarjeta CGA y MDA se diferencian claramente, se basan en el mismo controlador de vídeo, el MC6845 de Motorola. Aunque este fabricante de chips, con respecto a Intel, se llevó la parte pequeña del pastel, en este campo pudo mantener su fuerte durante años.

Hercules Graphics Card

Un año después de la introducción en el mercado del PC, apareció la hasta entonces totalmente desconocida empresa Hercules, con una tarjeta gráfica para el PC, y consiguió un éxito devastador. Esta tarjeta también se basaba en el mencionado controlador de Motorola, y era prácticamente compatible con la tarjeta MDA e IBM. Pero Hercules fue mucho más allá de las posibilidades de esta tarjeta, ya que aparte del modo de texto, la tarjeta HGC puede gestionar dos páginas gráficas con una resolución de 720 × 348 puntos en pantalla. Con ello combina la estupenda legibilidad de la tarjeta MDA con las capacidades gráficas de la tarjeta CGA, ampliando incluso la resolución.

Aún hoy en día, esta tarjeta se considera el estándar en cuanto a monitores monocromos se refiere, aunque las tarjetas monocromas, con respecto a las de color, pierden cada vez más importancia. Mientras que las tarjetas CGA y MDA hoy en día prácticamente ya no se producen, la tarjeta Hercules se puede encontrar en las ofertas de muchos fabricantes. Sobre todo los ordenadores del lejano oriente están equipados frecuentemente con este tipo de tarjeta, que en realidad no procede de la empresa Hercules, sino de un tercer fabricante cualquiera.

Tanto el original como sus imitaciones tienen el fallo de que les falta soporte del BIOS, ya que IBM siempre se ha negado a soportar tarjetas de vídeo externas por su BIOS. La Hercules Graphics Card sin embargo sufre poco bajo ello, ya que es compatible a la tarjeta MDA de IBM, y al menos en este aspecto se puede direccionar en el modo de texto a través del ROM-BIOS.
En lo que se refiere al modo gráfico de esta tarjeta, el programador realmente no se ve soportado por el BIOS en este aspecto, y esto es cierto para la inicialización del modo gráfico, así como para el acceso a diferentes puntos de pantalla. Pero esto no representa un gran problema, ya que las rutinas del BIOS correspondientes, a causa de su deficiente velocidad, normalmente no se emplean nunca, y el acceso a las informaciones de punto de una HGC se realiza de forma bastante sencilla.

Como que la Hercules Graphics Card representa una tarjeta «incombustible» en el mercado del PC, que por otra parte es de evolución muy dinámica, se puede demostrar muy bien la miniaturización de los diferentes componentes del PC tomándola como ejemplo. Mientras que las primeras tarjetas Hercules aún tenían la longitud completa, y contenían más de 40 circuitos integrados, las tarjetas Hercules modernas se montan en una tarjeta corta, y habitualmente tienen menos de diez CI, pudiéndose encontrar además de la tarjeta gráfica un puerto paralelo en ellas.

Aparte de la Hercules Graphics Card la empresa Hercules también ha comercializado otras tarjetas gráficas, que sin embargo no tuvieron tanto éxito, y que por ello ya no tienen ningún papel importante en el mercado de los PC. Se pueden nombrar: la Hercules Graphics Card Plus y la tarjeta Hercules InColor Card.
EGA
Después de que la Hercules Graphics Card entrara por primera vez en el mercado hasta entonces dominado por IBM, en IBM se dieron prisa para desarrollar el sucesor de la tarjeta CGA, que también tuviera la potencia como para desplazar la Hercules Graphics Card del mercado. Como resultado de estos esfuerzos, en 1985 se presentó el Enhanced Graphics Adapter, abreviado EGA.

Gracias a los grandes adelantos que se consiguieron entre los años 1981 y 1985 en el ámbito del hardware, la tarjeta EGA fue mucho más allá de las posibilidades de las tarjetas CGA y MDA, representando una pequeña revolución en el sector. Por consiguiente también resultó bastante cara. Se tardó un buen tiempo hasta que se convirtió en asequible para la mayoría de los usuarios, avanzando así hasta estándar.
Aparte de sus modos de vídeo propios, la tarjeta EGA es completamente compatible con las tarjetas CGA y MDA, y también se puede utilizar en programas que sólo soportan este tipo de tarjetas de vídeo. Además de ello la tarjeta EGA, semejante a la Hercules Graphics Card, dispone de la posibilidad de producir gráficos monocromos en un monitor monocromo, representando con ello la primera tarjeta gráfica que se podía utilizar tanto en monitores monocromos como en los de color.

Pero la EGA desplegaba todo su esplendor en unión de un monitor EGA especial, que va mucho más allá de las características de un monitor CGA. A pesar de que su resolución en el modo gráfico más alto, 640 × 350 puntos, no era mucho más alto que el de la CGA, se podían representar 16 colores diferentes, de una paleta de 64 colores. Con la representación de un mayor número de colores también viene la ampliación de la RAM de vídeo, que en las tarjetas EGA puede llegar a ser de hasta 256 KBytes, para tener espacio para varias páginas gráficas.

Pero todo esto sólo se consiguió abjurando de la controladora de vídeo MC6845, que no es capaz de este tipo de prestaciones. En vez de ello, la tarjeta EGA se basa en varios circuitos VLSI altamente integrados, que se encargan de todas las tareas en el marco de la generación de imagen. VLSI viene de «very large scale integration» que hace referencia a la tecnología de fabricación que posibilita la alta escala de integración lograda en estos circuitos

Este estándar se diferencia dramáticamente de los procedimientos anteriores especialmente en lo que se refiere a como se guardan las informaciones de imagen en la RAM de vídeo, y en cuanto a la programación de tarjetas de vídeo representa un importante inciso.

Eligiendo una distancia menor entre los puntos en los monitores EGA, la tarjeta EGA brilla con respecto a la tarjeta CGA con una imagen mucho más nítida, destacándose de su antecesora también en otros aspectos. Por ejemplo representa la primera tarjeta de vídeo que es capaz de trabajar con juegos de caracteres variables, ya que por software se pueden cargar juegos de caracteres cualesquiera. No sin razón, los primeros juegos de acción realmente vistosos no aparecieron hasta los comienzos de la tarjeta EGA.

Como las características ampliadas de una tarjeta EGA, al contrario que las tarjetas CGA y MDA, no se soportan por el ROM-BIOS, las tarjetas EGA disponen de un ROM-BIOS propia, que se encuentra en la misma tarjeta. Sustituye al BIOS original en cuanto a la salida de vídeo, y por ello pone a disposición todas las características de las tarjetas EGA.
Con la amplia difusión de las tarjetas EGA también comenzó la pelea entre los fabricantes de tarjetas compatibles, sobre quién hacía tarjetas cada vez más potentes con modos de vídeo cada vez más raros, que finalmente apenas son soportados por ningún programa. Mientras que IBM en un principio denunciaba a todos los fabricantes de una tarjeta compatible IBM, pronto esto no era posible debido a la inundación de tarjetas compatibles EGA que venían sobre todo del lejano oriente. Durante un tiempo incluso ocurrió la absurda situación de que diferentes fabricantes ofrecieron tarjetas EGA que con sus características iban mucho más allá del estándar VGA, que en realidad estaba pensado como sucesor natural del estándar EGA.

Hoy en día, las tarjetas EGA siguen estando muy difundidas, pero ya no se pueden considerar como estándar, ya que son desplazadas cada vez más por las tarjetas VGA. Pero son compatibles en gran medida a estas, ya que el estándar EGA en realidad no es otra cosa que un subconjunto del estándar VGA.

VGA
La tarjeta VGA, junto con los primeros sistemas PS/2 de IBM, se presentó en la primavera de 1987, empalma perfectamente a la tradición de la tarjeta EGA, es decir: Compatibilidad a todos sus antecesores, más colores, más resolución y mejor representación de texto.

En unión con los precios a la baja, no sólo para tarjetas VGA sino también para los monitores necesarios, estas características se convirtieron rápidamente en estándar del ámbito del PC. Quien hoy en día no quiera trabajar exclusivamente en monocromo, al comprar un ordenador apenas tendrá otra elección que la de VGA.
Y en realidad, el estándar VGA sólo estaba pensado para los sistemas PS/2 de IBM, y con ellos para el nuevo Micro Channel que hasta hoy no ha podido imponerse. Rápidamente aparecieron muchos fabricantes con tarjetas VGA para el bus ISA en el mercado, de modo que también los sistemas convencionales se podían equipar con tarjetas VGA.

Las tarjetas VGA se diferencias de las tarjetas EGA por su densidad de integración mayor, que hace posible la colocación de toda la lógica de control en un solo circuito. Pero también se diferencian de las tarjetas EGA, en que ya no envían señales digitales de color al monitor, sino analógicas. Con ello se hacen posible más de 260.000 colores, de los que puede haber activos, según modo, 2, 4, 16 o 256.
El modo de mayor resolución en las tarjetas VGA es de 640 × 480 puntos, donde este modo se puede representar en 2, 4, o 16 colores. Por otra parte, existe el modo de 320 × 200 puntos, que nos ofrece la magnífica cantidad de 256 colores simultáneos. Esta claro que en vista de las altas resoluciones y la multiplicidad de los colores se necesita una buena cantidad de RAM de vídeo para

acoger toda la información de imagen. Por ello, las tarjetas VGA se suministran habitualmente con un mínimo de 256 KBytes de RAM de vídeo, que se puede ampliar fácilmente a 512 KBytes.

Al igual que las tarjetas EGA, las tarjetas VGA también disponen de su BIOS propia, que sustituye el BIOS estándar en las salidas de vídeo. Análogamente al hardware de una tarjeta VGA, también su BIOS es compatible hacia abajo al BIOS de la EGA, de modo que todos los programas que fueron creados para el BIOS de la EGA, también se pueden ejecutar sin problemas bajo el BIOS de la VGA.
La gran competencia en el mercado de los PC ha creado una situación en el mercado, referida a las tarjetas VGA, que es mucho mayor que la que se origino con las EGA. Hay muchos fabricantes, y que intentan superarse cada vez más con diferentes modos de vídeo, y cada vez más colores. Hasta ahora, cada fabricante tenía sus propias maneras, de modo que los programadores habían de ajustar sus programas individualmente a cada una de las diferentes tarjetas de vídeo, de las que se habían de soportar modos ampliados. Esta es también la razón por la cual la mayoría de programas, a pesar de la gran difusión de las tarjetas VGA, sólo soportan los modos VGA estándar.

Pero parece que con una determinación de un estándar VGA ampliado se va abriendo un camino, para estandarizar el acceso a los modos ampliados de las VGA, para hacerlos accesibles a todos los programas.

Super-VGA
Las tarjetas Super-VGA corresponden, en lo que al hardware básico se refiere, con las tarjetas VGA normales, pero trabajan más rápido, para poder visualizar más puntos en pantalla en el mismo tiempo, y con ello obtener una resolución más alta. Además soportan todos los modos VGA normales, pero si se quiere están en disposición de mostrar sensiblemente más colores en pantalla, de los que serían posible en una tarjeta VGA normal.
Mientras que la representación de 256 colores en tarjetas VGA normales sólo es posible en el modo de 320 × 200 puntos, las tarjetas Super-VGA amplían esta posibilidad a los demás modos (640 × 200, 640 × 350 o 640 × 480 puntos). Además le permiten al usuario resoluciones de 800 × 600 o 1024 × 768 puntos, que sólo se pueden representar en conexión con los potentes monitores Multiscan, y que además consumen una enorme cantidad de RAM de vídeo.
Como siempre, entre los diferentes fabricantes de este tipo de tarjetas no existe unanimidad en cuanto a como se han de inicializar o direccionar este tipo de modos gráficos a través de los registros de la tarjeta. Por ello, los fabricantes más importantes de juegos de chips compatibles VGA (Tseng, Paradise y Video Seven) han formado un consorcio, que lleva el nombre de Video Electronic Standard Association (VESA). Juntos han determinado un estándar para el acceso a los modos ampliados de la Super VGA a través del BIOS, y que en un futuro se empleará en los BIOS sobre las que se basan las tarjetas de estos fabricantes. Para tarjetas Super-VGA se ofrecen programas TSR, que amplían el BIOS existente en unas cuantas funciones.

Por desgracia, estas convenciones se tomaron durante el año 1990, de modo que aún se tardará un tiempo en que estén totalmente difundidas. Hasta entonces, los programas tendrán que acceder directamente al hardware de las diferentes tarjetas Super-VGA, si quieren emplear los modos ampliados de la VGA.
MCGA
Mientras que IBM prevé para sus modelos altos PS/2 una tarjeta VGA, los modelos más pequeños se ofrecen con una tarjeta MCGA. Se trata de una mezcla de casi todos los estándares anteriores, que hasta ahora no ha entusiasmado a nadie. El nombre MCGA viene de «Memory Controller Gate Array», un término que ni siquiera suena a tarjeta de vídeo.

En lo que se refiere al modo de texto, estas tarjetas se comportan igual que una tarjeta CGA, es decir representan 80 × 20 caracteres en el modo texto, donde se puede elegir el color de texto y de

fondo de una paleta de 16 colores. Pero al contrario que en las tarjetas CGA, estos colores no están predeterminados, sino que, al igual que en una tarjeta VGA, se pueden elegir libremente del total de más de 262.000 colores. Y al contrario que en la tarjeta CGA, su resolución horizontal no es de 200 líneas, sino de 400 líneas, por lo que la definición de los caracteres es bastante mejor.

La tarjeta MCGA también aparece como hermafrodita en el aspecto de los diferentes modos gráficos. Aparte de los modos compatibles VGA también se soportan los dos modos de la CGA con 320 × 200 puntos y 640 × 200 puntos. Pero como esta tarjeta siempre ha de trabajar con una resolución horizontal de 400 puntos, las diferentes líneas de puntos de estos modos se duplican, lo que no mejora la imagen, pero a cambio da la mitad de la resolución.
Aún peor es el asunto en los modos VGA, que alcanzan la resolución VGA normal, pero se encuentran limitados en cuanto a colores. La razón de ello es que las tarjetas MCGA sólo vienen equipadas con 64 KBytes de RAM de vídeo, con los que, dada la resolución, solo se pueden sacar una cantidad limitada de colores. Esto me recuerda al propietario de una cara limousine, que es demasiado tacaño como para comprar gasolina para su coche.

Al contrario que las tarjetas VGA, las tarjetas MCGA no han conseguido, a causa de las limitaciones antes mencionadas (parcialmente incomprensibles) el salto del Micro Channel al gran mundo de los PC, y por ello han permanecido en un cierto anonimato. Y en esto no queremos cambiar nada en este libro, de modo que no hablaremos detalladamente de la programación de las tarjetas MCGA a lo largo de este capítulo.
8514/A
Para no permitir que alguien le robara la batuta en el campo de las tarjetas de vídeo, IBM presentó ya en el año 1987 un heredero para su estándar VGA. 8514/A era su nombre, un tanto críptico. Detrás de él se esconde nada menos que una revolución en el ámbito de las tarjetas de vídeo. Mientras que los controladores de vídeo, comparados con el procesador, hasta ahora no eran otra cosa que controladores tontos, ahora la tarjeta de vídeo misma se equipa con un procesador, al que se le pueden comunicar ordenes externas. La ventaja está clara, ya no es el procesador el que ha de calcular los puntos de las líneas o los círculos, sino que puede delegar esta tarea al procesador gráfico, que procesa, paralelamente a la ejecución del resto del programa, la línea deseada u otro objeto gráfico. De esta forma no se sobrecarga al procesador. Hasta aquí la teoría.

En la práctica, el nuevo caballo de batalla de IBM aún no se ha podido imponer, lo que seguro que tiene que ver con que el estándar VGA ahora está conquistando los corazones y las carteras de los usuarios. La razón para su fracaso actual también se puede buscar en algunos movimientos erróneos que IBM tiene a su cargo.
Por una parte, este estándar se pensó exclusivamente para la gama alta de los modelos PS/2, y con ello para el Micro Channel, que hasta ahora no se ha podido imponer, y tampoco lo hará en el futuro. Por otra, las características técnicas de esta tarjeta se han mantenido en riguroso secreto por IBM, para que ningún otro fabricante pudiera imitar una tarjeta similar. Pero precisamente esta es una estrategia que en el mercado del PC no cae en suelo fructífero, ya que el PC basa su éxito precisamente, comparándolo por ejemplo con el Apple Macintosh, en que todo está documentado abiertamente y es accesible para todo el mundo.

Pero hay una tercera razón que habla contra estos adaptadores de vídeo, ya que a la interfaz de software, que IBM pensó para esta tarjeta, los expertos le dedican una sonrisa condescendiente. A pesar de que se le ha de dejar a la interfaz de software que tiene una clara orientación al futuro, precisamente esto nos lleva a que el rendimiento den hardware en cuanto a velocidad se queda muy por detrás de sus posibilidades. Si además se considera el precio comparativamente alto de esta tarjeta, no es ningún milagro que la tarjeta 8514/A hasta ahora haya encontrado muy pocos adeptos.

El BIOS de vídeo

El ROM BIOS del PC contiene toda una serie de servicios, que se dedican a las diferentes tareas, que se crean en el marco de la salida por pantalla. Estos servicios se resumen bajo el techo de la interrupción 10h, que por ello se llama interrupción de vídeo. Para delimitar los diferentes servicios de otras funciones del ROM-BIOS, aquí también se habla del BIOS de vídeo, refiriéndose a todas las funciones que se pueden alcanzar a través de la interrupción 10h.
En este capítulo aprenderá entre otras cosas:

Qué funciones contiene el BIOS de vídeo, y como se emplean,
Por qué a veces es preferible el acceso directo al hardware de vídeo, en vez del empleo de estas funciones.

El BIOS de vídeo y sus extensiones

En su forma original, la interrupción 10h sólo ofrece funciones para el trabajo con tarjetas MDA y CGA. Por ello se soportan automáticamente las tarjetas Hercules, al menos para trabajar en el modo de texto, ya que son compatibles con el estándar MDA.
Sin embargo no se soportan de ninguna forma las tarjetas EGA y VGA con sus posibilidades ampliadas, por el BIOS original. Por ello, las tarjetas EGA y VGA contienen un circuito ROM con una ampliación del BIOS, que se enlaza automáticamente en la interrupción 10h durante el arranque del sistema, para poder adaptar algunas funciones a las necesidades de las tarjetas EGA y VGA. El BIOS de la VGA contiene las mismas extensiones que el BIOS EGA, pero va un poco más allá. A pesar de que existen tarjetas EGA y VGA de una cantidad inmensa de fabricantes, todas se ciñen a las ampliaciones del BIOS que IBM colocó en sus tarjetas EGA y VGA, ya que precisamente esto es lo que les da el sello de «compatible EGA/VGA», sin el cual un comprador difícilmente adquirirá la tarjetas.

Sobre todo los PC de alto rendimiento se equipan cada vez más con tarjetas VGA cuyo hardware se encuentra directamente en la placa madre, para garantizar una comunicación más rápida entre la tarjeta de vídeo y la CPU. Ya que estos sistemas están fijados para trabajar con una tarjeta VGA, las funciones ampliadas del BIOS EGA y VGA se adoptan directamente en el ROM-BIOS de la placa madre, de modo que en realidad ya no representan una ampliación. Pero para el trabajo con estas funciones, esto no hace ninguna diferencia.

Velocidad de las funciones BIOS

Las funciones del BIOS no representan la única posibilidad de llevar caracteres a la pantalla, o de situar el cursor. Además de ello, en la interfaz de funciones del DOS se encuentran otras funciones para la salida por pantalla, y naturalmente, todos los servicios del BIOS se pueden realizar mediante programación directa del hardware de vídeo. Los servicios del BIOS se han de colocar entre las funciones del DOS, y la programación directa del hardware. Ante todo se han de tener en cuenta aspectos como la velocidad de ejecución, flexibilidad e independencia del dispositivo, que no se pueden unir en consonancia.

Así por ejemplo el DOS es el que ofrece la independencia de dispositivo mayor, ya que las salidas se pueden redireccionar sin problemas a una impresora o un archivo. Pero a cambio, estas funciones son muy lentas y muy poco flexibles. La programación directa del hardware por lo contrario permite mayor velocidad y flexibilidad por que le permite al programador un control total de todos los procesos. Pero a cambio depende extremadamente del hardware, y las rutinas que por ejemplo se han programado para la salida de caracteres en tarjetas CGA, ni siquiera funcionan con tarjetas MDA.

Las funciones del BIOS por lo contrario no son ni mucho menos tan rápidas como rutinas que puedan acceder directamente al hardware, pero en cualquier caso trabajan con la tarjeta de vídeo instalada. En este caso el programador no ha de preocuparse de las diferencias de tarjetas CGA y MDA, ya que esta tarea la realiza el BIOS.
Si uno se pregunta por qué las funciones del BIOS son sensiblemente más lentas que el acceso directo al hardware, se han de nombrar dos razones en particular. Por una parte el mecanismo con el que se llaman las funciones del BIOS. Al fin y al cabo una llamada de interrupción cuesta mucho más tiempo que la llamada de una rutina dentro de un programa. Pero sobre todo la posición de las rutinas del BIOS es la que causa su lenta velocidad de ejecución. En la mayoría de tarjetas de vídeo se sigue tratando de tarjetas de 8 bits, de modo que los accesos al ROM-BIOS es mucho más lenta que por ejemplo los accesos a la memoria RAM, en PC modernos se puede direccionar en unidades de 16 o 32 canales. Si se tiene en cuenta que la CPU ha de pasar cada una de las instrucciones de lenguaje máquina de las rutinas del BIOS a través de este cuello de botella de 8 bits, se puede entender el porqué estas rutinas son tan lentas.

No en vano, en muchos PC, existe hoy en día la posibilidad de copiar el ROM-BIOS a una zona de RAM la llamada Shadow ROM que se encuentra entre la RAM de vídeo y el límite de memoria de 1 MByte. A pesar de que las diferentes rutinas del BIOS allí se ejecutan bastante más rápidamente, por que son posibles los accesos de 16 y 32 bits, pero las desventajas generales con respecto a la programación directa del hardware tampoco se pueden evitar.
Pero como las funciones del BIOS contienen muchos servicios útiles, la mayoría de aplicaciones de PC trabajan con las funciones del BIOS. Aunque también emplean rutinas, que acceden directamente al hardware de la tarjeta de vídeo. Esto sobre todo es así cuando se trata de salidas rápidas por pantalla en el modo texto y gráfico. Las funciones para la salida de texto aún se pueden utilizar menos que las rutinas gráficas del BIOS. Funciones de control, como pueden ser la inicialización de un modo de vídeo, la selección de una página de pantalla, o la colocación del cursor de texto, se le dejan al BIOS. Esto se llama división de poderes.

Los servicios del BIOS de vídeo
En este capítulo no se describen todos los servicios del BIOS de vídeo, sino solamente las funciones de control más importantes, y los servicios que sirven para la salida de pantalla en modo de texto. Pero puede encontrar más funciones en los siguientes capítulos, como por ejemplo en el apartado 9.8.2, donde de trata la programación de juegos de caracteres propios en tarjetas EGA y VGA. Además puede encontrar una recopilación de todas las funciones del BIOS de vídeo en la parte de referencia bajo «Funciones del BIOS de vídeo EGA/VGA».

Pero en este lugar queremos incluir un resumen de los diferentes servicios que se esconden detrás de las diferentes funciones del BIOS de vídeo y que frecuentemente disponen de subfunciones propias, para poder realizar diferentes tareas. Véase la tabla 49.
Para la llamada de las diferentes funciones sirve como base que antes de la llamada de la interrupción de vídeo del BIOS se ha de cargar el registro AH con el número de la función. Si se quiere llamar una subfunción, el número de subfunción se la de emplazar normalmente en el registro AL. Aunque existen excepciones, que se indican en este libro. Una de las excepciones son habitualmente las funciones que devuelven diferentes registros de procesador con contenidos modificados a su invocador, ya que normalmente no se modifica el contenido de los diferentes registros por las funciones.

Selección del color de carácter y fondo
Algunas de las funciones para la salida de caracteres en la pantalla esperan de su invocador la indicación de un color de texto y de fondo, ya que los caracteres de las diferentes tarjetas de vídeo de PC se pueden representar con colores separados de texto y fondo. Por desgracia, las tarjetas de vídeo de color y monocromas codifican estos colores de forma totalmente distinta, lo que

simplemente refleja sus diferentes características con respecto a los colores. En ambos casos sin embargo, se retiene para cada carácter un byte de color y otro de atributo, que está dividido en dos Nibbles. El Nibble bajo, es decir, los bits 0 a 3, definen el color de texto, mientras que los bits 4 a 7, que hacen el Nibble alto, codifican el color de fondo.

En las tarjetas monocromas, en los dos Nibbles se incluye un bit para la intensidad del color de texto, y la intermitencia del carácter, como muestra la siguiente figura.
Si no se tiene en cuenta el bit de intensidad para el color de texto y fondo, a los colores de texto y fondo se le pueden asignar diferentes combinaciones de bits, como muestra la siguiente figura.
Colores frecuentemente seleccionados en tarjetas de pantalla monocromas son los colores 07h, que resultan en texto claro sobre fondo oscuro, y 70h, que representa texto oscuro sobre fondo claro. Estos códigos también tienen sentido en conexión con las tarjetas de color, aunque no aprovechen las posibilidades de estas tarjetas, ya que resultan en letras gris claro sobre fondo negro, o letras negras sobre fondo gris claro.

Seleccionando los valor 0 o 7 como color de texto y fondo, se evita intencionadamente el poner a uno el bit 7 del byte de atributos, con lo que los caracteres no hacen intermitencia, siempre que la tarjeta de vídeo esté ajustada de modo, que los caracteres con este bit activado hagan intermitencia, y no que se representen con un color de fondo intensivo.
Los bytes de color no tienen una estructura muy diferente si se los compara con la tarjeta monocroma. Pero aquí se puede seleccionar un color real para texto y fondo, que procede de una paleta de 16 colores. Las siguiente dos figuras muestran la estructura del byte de color en tarjetas de color, y la paleta de colores disponible. Véase la tabla 50.

El juego de caracteres ASCII del PC
El PC trabaja con un juego de caracteres, que comprende 256 caracteres, y que tiene sitio para números, letras, caracteres especiales, y otro tipo de caracteres. A ellos pertenecen, aparte de los signos de idiomas extranjeros, todos los símbolos de marcos y matemáticos.
Con respecto a las diferentes funciones del BIOS de vídeo, los diferentes caracteres de este juego de caracteres se representan por su código ASCII.
El sistema de coordenadas de la pantalla

Muchas funciones del BIOS esperan como parámetro de entrada coordenadas de pantalla, con cuya ayuda se puede direccionar una posición de pantalla determinada. Por ello, el conocimiento del sistema de coordenadas empleado es básico para la llamada de muchas funciones.
Independientemente de si está activo el modo de texto o gráfico, el punto de origen de coordenadas se encuentra en la esquina superior izquierda. Lleva las coordenadas (0/0). Hacia la derecha o hacia abajo se incrementas desde este punto las ordenadas X e Y. En el modo de texto de 80 × 25 caracteres, la coordenada de la esquina inferior derecha de la pantalla es por ello (79/24), mientras que en modo gráfico de 640 × 200 puntos de la tarjeta CGA es (639 × 199).

Inicialización de un modo de vídeo
Una función muy importante, que muy pocas veces se sustituye por programación directa del hardware, es la función 00h. Indicando el código de modo en el registro AL se pueden activar con su ayuda todos los modos de vídeo estándar de las tarjetas MDA hasta VGA. Esto sirve tanto para modos de texto, como para modos gráficos. La única excepción la representa el modo gráfico de la tarjeta Hercules.
Condición indispensable para la inicialización de un modo de vídeo es naturalmente que existe la tarjeta de vídeo correspondiente. Pero esto no se puede averiguar con ayuda de esta función, ya que no se le señaliza al invocador si la operación ha fallado.

Al llamar la función 00h no solo se inicializa el modo de vídeo deseado, sino que simultáneamente se borra el contenido de la RAM de vídeo. Al inicializar los modos de EGA y VGA esto se puede suprimir si se desea, añadiendo 128 al valor del número de modo, es decir activando el bit 7 del número de modo. En ese caso, el contenido actual de la RAM de vídeo permanece intacto, y se visualiza en pantalla después de la inicalización del modo deseado.
Al ejecutar un programa se puede partir de que hay activo un modo de texto de 80 × 25 caracteres. Al utilizar tarjetas de pantalla monocromas, estará activo el modo 7, si se trabaja con tarjetas CGA, el modo 3. Por ello, la función 00h se ha de llamar ineludiblemente, si su programa ha de funcionar en el modo de texto de 80 × 25 caracteres.

Lo contrario a la función 00h lo representa la función 0Fh, con la que se puede obtener el modo de vídeo actual. Esta función se llama con el valor 0Fh en el registro AH. Como resultado devuelve el valor del modo de vídeo, en correspondencia con la tabla anterior, en el registro AL. Aparte del modo de vídeo, esta función también devuelve el número de columnas por línea de pantalla del modo actual, si se trata de un modo de texto. Esta información se puede encontrar en el registro AH después de la llamada. Además también se devuelve el número de la página de pantalla actual. Se puede encontrar en el registro BH. Véase la tabla 52.

Programación del cursor de texto
En el modo de texto, todas las tarjetas gráficas desde MDA hasta VGA, muestran un cursor intermitente de texto, que marca el punto de inserción o salida actual. Tanto el aspecto de este cursor, así como su posición en la pantalla se pueden ajustar con ayuda del BIOS de vídeo.
El posicionamiento se convierte en asunto de la función 02h. Espera durante su llamada el número de función 02h en el registro AH, la línea en el registro DH, y la columna en el registro DL, a la que se quiera desplazar el cursor. Además. el registro BH ha de contener el número de la página de pantalla, en la que se quiere posicionar el cursor. Se ha de tener en cuenta, que cada página de pantalla dispone de un cursor propio. Por ello, el desplazamiento de este cursor intermitente con ayuda de esta función, sólo se hace visibles, cuando el valor en el registro BH se corresponda con la página de pantalla actual. Sin embargo, mediante la llamada de esta función no solo se coloca el cursor intermitente de pantalla en el lugar deseado, sino que también se determina la posición de pantalla, en la que comenzará la salida de caracteres con la función destinada a ello. Esto quedará más claro, cuando describamos esta función más adelante.

Lo contrario a la función 02h lo es la función 03h. Lee la posición actual del cursor de una página de pantalla determinada, y la devuelve al programa invocador. Al llamar esta función, el registro AL ha de contener el número de función 03h, y el registro BH el número de la página de pantalla, cuya posición de cursor se quiere averiguar. Además, esta función devuelve en los registros CH y CL dos valores adicionales, que tienen una particularidad: indican la línea inicial y final del cursor, que definen su aspecto, y no su posición.

Para entender el significado de estos parámetros, se ha de saber que en una tarjeta en color cada carácter (de texto) se compone de 8 líneas de puntos y en una tarjeta monocroma de 14 (no se confundan con las líneas de pantalla). El programador tiene aquí la posibilidad de decidir en qué línea ha de comenzar el cursor, y en que línea ha de acabar. Con ello tiene la posibilidad de definir cursores pequeños y grandes, lo que normalmente se utiliza en la entrada de informaciones para indicar determinados estados de funcionamiento.

Al determinar estos valores, la altura de la matriz de caracteres en la que se definen los caracteres naturalmente tiene mucha importancia. En una tarjeta CGA esto son 8 líneas de puntos, de modo que se pueden determinar valores entre 0 y 7 para las líneas inicial y final del cursor. En una tarjeta MDA o Hercules, la altura de la matriz es de 14 líneas, de modo que las líneas iniciales y finales del cursor se han de encontrar entre 0 y 13. Aún más alta es la matriz de caracteres en tarjetas EGA y

VGA, pero aquí, el BIOS trabaja con la escala de la CGA de 0 a 7, y luego convierte las indicaciones de forma automática para que quepan en la altura real de la matriz de caracteres.

Indicando valores mayores para la línea inicial y final, se puede hacer desaparecer el cursor de pantalla, lo que a veces es necesario.
Mientras que la línea inicial y final del cursor se puede leer con ayuda de la función 03h, la función 01h sirve únicamente para definirlas. A este fin, se ha de cargar el registro AH con 1, antes de llamar a la interrupción 10h, el registro CH con la línea inicial, y el registro CL con la línea final del cursor intermitente. Por favor tenga en cuenta que la línea inicial ha de ser menor o igual que la línea final, ya que sino el cursor no es visible.

Selección de la página de pantalla
Hasta ahora se ha hablado varias veces de la «página de pantalla actual», sin haber explicado cómo se activa una página de pantalla. La respuesta está en la función 05h del BIOS de vídeo. Se ha de llamar con el valor 05h en el registro AH, y el número de la página de pantalla activar el en registro AL. El número de la página a activar naturalmente también depende de cuantas páginas de pantalla se disponga en el modo de vídeo actual en la tarjeta de vídeo. Ya que la tarjeta MDA sólo ofrece una pantalla, la llamada de esta función al utilizar una tarjeta MDA no tiene sentido. Los siguientes valores se permiten para otras tarjetas de vídeo, aunque también se ha de tener en cuenta el modo actual de vídeo. Véase la tabla 53.

Las diferentes páginas de pantalla se cuentan siempre desde 0, de modo que por ejemplo en las tarjetas EGA y VGA en el modo 2 se pueden direccionar las páginas de pantalla 0 a 7.

Salida de caracteres con el BIOS
El BIOS de vídeo dedica toda una serie de funciones a la salida de caracteres. Entre los modos de funcionamiento de estas funciones hay algunas diferencias, que se notan por ejemplo en la forma en que se tratan determinados códigos de control. Este nombre corresponde a los códigos ASCII 7, 8, 10 y 13. En el fondo representan caracteres normales, del juego de caracteres del IBM (ver antes), pero a lo largo de la historia de la informática se les ha asignado un significado determinado, que aún viene de la época de los terminales de impresión. Véase la tabla 54.

Algunas funciones consideran estos códigos como caracteres ASCII normales, y los muestran como tales, mientras que otras los evalúan como códigos de control, que por ejemplo en el caso del código 7, generan un pitido. Según si desea este procesamiento de códigos de control o no, debería decidir, qué función debe utilizar.
En todas las funciones para la salida de texto se ha de tener en cuenta que se pueden aplicar tanto en el modo texto, como en el modo gráfico. A pesar de que en el modo gráfico no es posible una salida de texto sin más, ya que aquí no hay disponible un juego de caracteres, pero este Handicap se evita automáticamente por el BIOS, pasando los patrones de carácter de los códigos ASCII a puntos gráficos. Mientras que los patrones de carácter ASCII 0 a 127 ya están guardados en la ROM, los patrones de carácter de los códigos 128 a 255 se leen de una tabla en la RAM, que primero se ha de instalar mediante el comando del DOS, GRAFTABL.

La dirección de esta tabla, el BIOS la toma de un puntero FAR, que se puede encontrar en la posición de memoria 0000:007C. Estas posiciones de memoria se encuentran en la tabla de interrupciones, pero se pueden utilizar para este propósito, ya que la interrupción 1Fh, cuya dirección se encuentra normalmente allí, no se utiliza.
La circunstancia, de que esta tabla se guarde en la RAM, permite definir una tabla propia, de modo que se pueden visualizar caracteres especiales que no están contenidos en el juego de caracteres estándar. como cada carácter queda definido por 8 bytes, los primeros 8 bytes de la tabla se refieren al código ASCII 128, los siguientes 8 al código 129, etc. Cada byte representa el patrón de bits para

una de las 8 líneas de las que se compone un carácter. El bit 0 representa en cada byte el punto del borde derecho de la matriz de caracteres, y el bit 7 el punto en el borde izquierdo. Si un bit está a 1, esto significa, que el punto correspondiente de la pantalla está encendido.

Para la salida de caracteres dispone de las dos funciones 09h y 0Ah. Sólo se diferencian en que la función 0Ah representa los caracteres en el color que ya está determinado para la posición de pantalla, mientras que la función 09h también determina el color (el atributo) del carácter a visualizar. Ambas funciones no desplazan el cursor a la siguiente posición de pantalla, de modo que en una llamada posterior la salida de caracteres ocurre en la misma posición.
Por eso, para visualizar un texto coherente, después de la llamada de la función se ha de mover el cursor con ayuda de la función 02h a la siguiente posición de pantalla.

Ambas funciones interpretan los códigos de control anteriormente descritos como caracteres normales, y los visualizan en correspondencia. Se llaman, cargando el registro AH con el número de función y el registro AL con el código ASCII del carácter a visualizar. La página de pantalla en la que se ha de visualizar el carácter la acoge como siempre en registro BH. En el registro CX hay un número, que indica cuantas veces se debe realizar la salida. Con ello es posible el visualizar un carácter varias veces consecutivas con una sola llamada de función, lo que ahorra mucho tiempo. Si el carácter en el registro AL sólo se ha de visualizar una vez, el registro CX se ha de cargar con un 1 antes de la llamada de la función.

A causa de un error en el BIOS, el factor de repetición en la llamada de esta función durante el modo gráfico sólo puede ser tan grande, como para que quepan todos los caracteres a visualizar en una sola línea.
Hasta aquí la ocupación de registros de la función 0Ah. Ya que con la función 09h también se puede determinar el color del carácter a visualizar, aquí también se emplea el registro BL, que está diseñado para acoger el color de los caracteres.
Una gran desventaja de estas dos funciones, en concreto el hecho de que no se desplace la posición del cursor después de la llamada de función, se compensa con la función 0Eh, Simula a un terminal, o un teletipo, y por ello en inglés frecuentemente se llama como rutina TTY, donde TTY está por Teletype. En su llamada, después de visualizar un carácter, se desplaza el cursor a la siguiente posición de pantalla. Si ya se encuentra al final de la línea de pantalla, se realiza un salto al inicio de la siguiente línea.

Si ya no hay más líneas, porque el cursor ya se encuentra en la esquina inferior derecha de la pantalla (línea 24, columna 79), el contenido completo de la pantalla se desplaza una línea hacia arriba, con lo que la primera línea de pantalla desaparece de la zona visual. A continuación se borra la línea 24, y el cursor se desplaza al inicio de esta línea, para seguir visualizando caracteres.
Al contrario que en las funciones 09h y 0Ah los códigos de control no son interpretados por la función TTY como códigos ASCII normales, sino que se procesan según su función. Análogamente a la función 0Ah la función TTY toma durante la salida de un carácter el color, que hasta entonces se encontraba en aquella posición de pantalla. Pero esto sólo es cierto para los modos de texto. En los modos gráficos, el color de texto del carácter se le ha de pasar a la función TTY en el registro BL.

Además, esta función espera durante su llamada en el registro AH el número de función 0Eh, en el registro AL el código del carácter a visualizar, y en el registro BH la página de pantalla, en la que debe aparecer el carácter.
Salida de cadenas (strings)
Con la introducción del los AT, el BIOS de vídeo se amplió en una función, que también se puede encontrar en las versiones del BIOS de las tarjetas EGA y VGA. Por fin ofrece la posibilidad de

visualizar en la pantalla una cadena de caracteres con una sola llamada de función, sin tener que llamar para cada carácter la función de salida. Esta función tiene el número de función 13h.

En su llamada se han de pasar toda una serie de argumentos, aparte del número de función en el registro AH. El registro BH acoge el número de la página de pantalla en la que se ha de visualizar la cadena string. Naturalmente no se ha de tratar obligatoriamente de la página actual. La posición inicial de la cadena en la pantalla encuentra espacio en el registro DH (línea) y DL (columna). La salida es independiente de la posición actual de cursor. El registro CX acoge el número de caracteres a visualizar de la cadena.
El contenido del registro AL define uno de los 4 posible modos, en los que se puede visualizar una cadena de caracteres. El formato de la cadena a visualizar se distingue por una parte en los modos 0 y 1, y por otra en los modos 2 y 3. Mientras que en los modos 2 y 3 a cada códigos ASCII le ha de seguir el byte de atributos correspondiente, en los modos 0 y 1 se enfilan directamente los diferentes caracteres de la cadena. En estos modos, se determina el byte de atributos para todos los caracteres mediante el contenido del registro BL. Independientemente de la estructura del buffer, en cualquier caso se ha de pasar un puntero FAR al buffer en la pareja de registros ES:BP.

Aunque en los modos 2 y 3 para cada carácter se depositan 2 bytes en la cadena de caracteres, y por ejemplo una cadena de 4 caracteres obtiene una longitud de 8 bytes, en el registro CX sólo se ha de indicar el número de caracteres a visualizar. En el ejemplo mencionado serían 4. Existe otra diferencia entre los modos 0 y 2 y los modos 1 y 3. Después de la salida de la cadena de caracteres en los modos 1 y 3, el cursor se coloca en la posición de pantalla, que le sigue a la del último carácter de la cadena. La siguiente salida de carácter mediante una función del BIOS se realizará entonces en esta posición de pantalla. la actualización de la posición del cursor no se realiza en los modos 0 y 2.

Obtener caracteres de la pantalla
Mientras que las funciones 09h, 0Ah y 0Eh sirven para visualizar caracteres en la pantalla, la función 08h permite leer caracteres de la pantalla. Es decir, averigua qué carácter está en una posición de pantalla determinada, y con que atributo se está representando. Antes de la llamada de esta función se ha de cargar el valor 8 en el registro AH, y el número de la página de pantalla direccionada en el registro BH. La posición de pantalla, de la que se lee el carácter, es la posición actual del cursor en esa página.

Mientras que en el modo texto el código del carácter se puede leer directamente de la RAM de vídeo, en el modo gráfico sólo se puede obtener, comparando el patrón del carácter en la posición actual del cursor con los patrones de carácter de todos los caracteres. Pero esto no funciona siempre, de modo que no se ha de esperar demasiado de esta función en el modo gráfico.
Como resultado se devuelve después de la llamada de la función el atributo (el color) en el registro AH, y el código del carácter leído en el registro AL.

Desplazamiento (scrolling) de pantalla
En relación con la función 0Eh (Salida TTY) se habló de la capacidad de desplazar la pantalla una línea hacia arriba en caso necesario, o hacer scroll, como se dice en inglés.
Esta capacidad la obtiene la función 0Eh mediante una llamada interna de la función 06h, que también está a disposición del programador. Con su ayuda se puede desplazar una zona determinada de la pantalla una o varias líneas hacia arriba, o ser rellenada con espacios. Esta operación sólo se puede realizar con la página de pantalla representada actualmente. Para llamar a esta función, se ha de cargar el número de función 06h en el registro AH, y el número de líneas que se quiere desplazar la zona de pantalla en el registro AL. El valor 0 le señala al BIOS, que la zona de pantalla no se quiere desplazar, sino que debe ser llenada con espacios. En el registro BH, excepcionalmente, no

se indica la página de pantalla, sin el color para los blancos. Mediante el registro CH, CL, DH y DL se define la zona de pantalla, que ha de ser procesado por la función. Véase la tabla 55.

Aparte de la función 06h, el BIOS también conoce la función 07h, que realiza la misma tarea, pero que desplaza la ventana de pantalla indicada hacia abajo. La ocupación de los registros es la misma que en la función 06h, amén de los distintos números de función.
Ejemplos de programa
Para terminar este capítulo queremos presentarles dos programas, que hacen accesibles algunas funciones de la interrupción de vídeo del BIOS desde lenguajes de alto nivel. Esto sirve especialmente para las diferentes funciones de control y las funciones para acceder a la pantalla en el modo de texto.

La ventaja de estos programas es que la salida por pantalla en Pascal y C mediante las funciones del BIOS es parcialmente más rápida que a través de las funciones estándar de los lenguajes que hacen servir las lentas funciones del DOS para la salida por pantalla.
Pero no queremos ocultar una desventaja importante de la salida por pantalla mediante las funciones del BIOS: a los comandos de lenguaje de alto nivel para la salida por pantalla se le pueden pasar variables numéricas, las que incluso teniendo en cuenta una cierta cantidad de cifras y decimales han de ser convertidas primero en caracteres ASCII, antes de ser visualizadas. Esto no es posible con las funciones del BIOS. Si a pesar de todo quiere visualizar variables numéricas con las funciones del BIOS, ha de convertirlas primero a una cadena de caracteres mediante la función apropiada, y enviar después esta cadena de caracteres a las funciones de la salida del BIOS. Un proceso que naturalmente necesita tiempo...

Ambos programas trabajan en su parte de demostración según el mismo principio. Primero rellenan la pantalla con caracteres sucesivos del juego de caracteres del PC, y después abren dos ventanas, en las que se mueven dos flechas hacia arriba y abajo. Otra cosa en común entre los dos programas es que por razones de compatibilidad entre tarjetas monocromas y color sólo acceden a una página de pantalla, y por ello con contienen subprogramas o funciones y procedimientos para la llamada de funciones gráficas del BIOS.

Si ha comprendido este capítulo, no debería resultarle difícil el completar el programa correspondiente con las funciones que le faltan, y escribir un pequeño programa de demostración propio. En realidad estos programas deberían animarle al «bricolaje», ya que por una parte es imposible «colgar» su ordenador ya que se va a través de las funciones del BIOS, y por otra es imposible que le ocurre algo negativo a la máquina. Precisamente por ello no se imponen límites a su fantasía. Pero antes de comenzar con el trabajo, observemos con detalle los diferentes programas.

Averiguar las tarjetas de vídeo instaladas
Siempre que un programa quiera acceder directamente al hardware de una tarjeta de vídeo, o a una determinada función del BIOS, que sólo se puede encontrar en determinadas versiones del BIOS, como por ejemplo un BIOS EGA/VGA, debería asegurarse primero de que la tarjeta direccionada realmente está instalada. Si no se realiza semejante test, puede ocurrir fácilmente que se escriban los caracteres en la RAM de vídeo, pero que la pantalla permanezca oscura, ya que el PC dispone de una tarjeta CGA en vez de la MDA esperada.

Sobre todo cuando un programa ha de trabajar con diferentes tipos de tarjetas de vídeo, y a pesar de todo ha de acceder directamente al hardware de estas tarjetas, el conocimiento del tipo de la tarjeta de vídeo instalada es especialmente importante. Sólo esta información permite ajustar las rutinas de salida de manera óptima a los parámetros de las diferentes tarjetas. En este capítulo puede aprender:
Cómo se puede averiguar el tipo de tarjeta de vídeo instalada en tiempo de ejecución

Qué tarjetas de vídeo se puedenhacer funcionar simultáneamente en el PC

Funcionamiento simultáneo de varias tarjetas de vídeo

Si en tiempo de ejecución del programa se quiere averiguar el tipo de la tarjeta de vídeo instalada, no se puede olvidar que el PC puede disponer simultáneamente de una tarjeta de pantalla monocroma (MDA, HGC o EGA en un monitor MDA) y una tarjeta de pantalla en color (CGA, EGA o VGA), aunque sólo pueda haber una activa simultáneamente. Las diferentes posibilidades de combinación que resultan de esto se pueden ver en la siguiente tabla. Véase la tabla 56.

Se trata de averiguar el tipo de la tarjeta de vídeo instalada. Por desgracia, no se pueden hacer servir una llamada de función del DOS del BIOS o alguna variable. En vez de ello se ha de desarrollar una rutina en ensamblador, que en función de determinados test compruebe la disponibilidad de los diferentes tipos de tarjetas de vídeo. Puede recurrir a las indicaciones de los fabricantes de hardware, ya que casi todos los fabricantes de tarjetas de vídeo nombran en la documentación técnica de su tarjeta algún método con cuya ayuda se puede verificar la existencia de esta tarjeta en el sistema del PC. Naturalmente también es importante, que este test sea inconfundible, es decir que sólo de un resultado positivo con una tarjeta determinada. Aquí sobre todo las tarjetas EGA y VGA presentan dificultades, ya que, según el monitor conectado, emulan una tarjeta CGA o MDA, y no se pueden distinguir de ellas.

Tests para averiguar la tarjeta de vídeo activa

Todos los tests, que se describen en las siguientes páginas se encuentran al final de este apartado en forma de dos pequeños programas en ensamblador, que se concibieron para incluirlos en programas C y Pascal. En un vector (Array), cuya dirección se les pasa por la función invocadora, escriben el tipo de tarjeta de vídeo y el tipo del monitor conectado. Si hay dos tarjetas de vídeo instaladas, del orden en que aparecen en el vector es determinante para saber cual de las dos tarjetas está activa en este momento.

Las siguientes tarjetas se cubren con la rutina en ensamblador

- Tarjetas MDA
- Tarjetas CGA
- Tarjetas HGC
- Tarjetas EGA
- Tarjetas VGA (también Super-VGA)

Ya que dentro de la rutina de ensamblador se busca selectivamente la existencia de una determinada tarjeta de vídeo, en el listado en ensamblador se encuentra un subprograma para cada familia de tarjetas de vídeo, que realiza esta tarea. Lleva el nombre de la tarjeta de vídeo, a cuyo descubrimiento ha de llegar. Por ello, estas rutinas tienen los nombre TEST_EGA, TEST_VGA, etc. A pesar de que los tests se llaman secuencialmente, algunos se pueden excluir, si dan un resultado negativo.

Esto se puede aplicar por ejemplo al test de la CGA, si anteriormente se ha detectado una tarjeta EGA o VGA, que está conectada a un monitor de alta resolución. A pesar de que junto a esta tarjeta no se puede instalar una tarjeta CGA, pero esto no detendría al test de la CGA, de demostrar que esta existe, aunque en realidad se trate de la tarjeta EGA o VGA ya detectada anteriormente.

Para no dejar que aparezcan contradicciones, para cada test existe una bandera, que decide sobre si el test se ha de realizar o no. Antes del primero, el test de VGA, todas las banderas se ponen a 1, de modo que los diferentes tests se realizan cuando les toca. A lo largo de los tests, las banderas nombradas se pueden poner a cero a causa de las razones indicadas, evitando así la llamada de la rutina de test asociada.

Estos tests se introducen con el test de la VGA. Es muy simple. Mediante una función especial del BIOS de la VGA, la subfunción 00h de la función 1Ah, se pueden obtener exactamente las informaciones, que debe suministrar la rutina en ensamblador. Sin embargo, sólo están disponibles después de la llamada de la función, si hay instalada una tarjeta VGA, y con ello un BIOS VGA.
Este es el caso, si después de la llamada de la función se encuentra el valor 1Ah en el registro AL. Si de lo contrario, la rutina de test encuentra otro valor allí, el test se interrumpe sin éxito, y se continua con otro test. En este caso no hay instalada una tarjeta VGA.

Después de una llamada con éxito de esta función, en el registro BL se encuentra un código especial de dispositivo para la tarjeta de vídeo activa, y en el registro BH para la inactiva. Pueden aparecer los siguientes códigos:

Códigos de retorno después de la llamada de la subfunción 00h de la función 1Ah del BIOS de la VGA.

Código	Significado
00h	no hay tarjeta de vídeo
01h	Tarjeta MDA en monitor MDA
02h	Tarjeta CGA en monitor CGA
03h	reservado
04h	Tarjeta EGA en monitor EGA
05h	Tarjeta EGA en monitor MDA
06h	reservado
07h	Tarjeta VGA en monitor VGA monocromo analógico

Detrás de los diferentes tipos de monitor se pueden esconder monitores Multisync, que emulan un determinado tipo de monitor.

Estos códigos se dividen por la rutina de test de la VGA en valores separados para la tarjeta de vídeo y el monitor conectado, y se cargan en el vector, cuya dirección se le ha pasado por el invocador. A causa de que esta rutina ya devuelve informaciones sobre ambas tarjetas de vídeo, los siguientes tests ya no se han de ejecutar. Simplemente el test Mono se ejecutará, siempre y cuando la función haya informado de la existencia de una tarjeta de pantalla monocroma, ya que no está en disposición de distinguir entre una tarjeta MDA y una HGC.

Dentro de la rutina principal del programa en ensamblador, al test de la VGA le sigue el test de EGA, que sólo se ejecuta, si el test de la VGA no tuvo éxito, y por ello no se puso a cero la bandera EGA. También el hace servir una función que sólo se encuentra en el BIOS de la EGA: la subfunción 10h de la función 12h. Si no hay ninguna tarjeta EGA instalada, esta función no está disponible, y por ello el valor 10h no se encuentra en el registro BL después de la llamada de la función, adónde se cargó para la llamada de la función. En este caso de finaliza el test de la EGA.

En caso de éxito, el registro CL contiene después de la llamada de función la posición de los interruptores DIP de la tarjeta EGA, que informan sobre el tipo de monitor conectado. Se convierten en los códigos de monitor de la rutina en ensamblador y se graban en el vector, junto con el código de la tarjeta EGA. Según el tipo del monitor conectado, se pone a cero la bandera CGA o Mono, ya que el test asociado ya no se ha de realizar a causa del descubrimiento de una tarjeta EGA. A continuación se finaliza la rutina EGA.

Si la bandera CGA no se ha puesto a cero durante alguno de los tests anteriores, a continuación del test de EGA, viene el test de CGA. Al igual que el test de Mono, no puede hacer servir ninguna función del BIOS, sino que ha de comprobar directamente la presencia del hardware asociado. Esto ocurre en ambas rutinas mediante la llamada de la rutina TEST_6845, que determina si la controladora de vídeo 6845 de la tarjeta de vídeo correspondiente se encuentra en la dirección de

puerto indicada. En el caso de la tarjeta CGA, esta es la dirección de puerto 3D4H, que se le pasa a la rutina TEST_6845.

El único camino para demostrar la presencia del CRTC en una dirección de puerto determinada consiste en escribir un valor cualquiera (distinto de cero) en uno de los registro de la controladora CRTC, y volver a leerlo inmediatamente. Si coincide con el valor escrito, el CRTC, y con ello la tarjeta de vídeo existen en la dirección de puerto especificada. Pero antes de escribir alegremente un valor en uno de los registros CRTC, debería uno acordarse de que tiene un significado central en la construcción de las señales de vídeo y de que un acceso a la ligera no sólo podría confundir completamente al CRTC, sino que además, en el peor de los casos, se podría causar un daño al monitor. Los registros 0 a 9 no se pueden utilizar para el test, a causa de esto. Quedan los registros 10 a 15, cuya modificación también tiene influencia al contenido de la pantalla. Pero no todos los registros pueden ser escritos y leidos en este caso. Pero con los registros 14 y 15, que determinan la dirección del cursor intermitente de pantalla, no hay problemas en este sentido.

La rutina en ensamblador lee para comenzar el contenido del registro 14, antes de escribir un valor seleccionado al azar. Después de una breve pausa, en la que el CRTC puede reaccionar a este cambio, se vuelve a leer el contenido de este registro. Pero antes de que se compare con el valor escrito, primero se vuelve a escribir al valor antiguo en el registro, para que este test no tenga consecuencias permanentes sobre la pantalla. Si la comparación que viene a continuación determina la identidad del valor escrito y del leido, hay presente un CRTC, y con ello una tarjeta de vídeo (en este caso una CGA). La rutina CGA reacciona a ello, cargando el código correspondiente en el vector, donde indica como monitor un monitor en color, que es el único que se puede hacer funcionar en conjunto con una tarjeta CGA.

Como último test se llama al test Mono, que igualmente comprueba la disponibilidad de un CRTC 6845 en este caso en la dirección de puerto 3B4h . Si allí se encuentra un CRTC, en cualquier caso hay instalada una tarjeta de pantalla monocroma, aunque falta por aclarar si se trata de una tarjeta MDA o HGC. Para decidir esto, se utiliza el registro de estado de los dos tarjetas de vídeo, que se puede direccionar a través de puerto 3BAh. Mientras que el bit 7 en este registro no tiene ninguna importancia en el caso de una tarjeta MDA, y su contenido por ello no está definido, en el caso de la tarjeta HGC siempre es 1, cuando el haz de electrones del tubo de pantalla está en su retrazado vertical. Como esto no es permanente, sino que ocurre cada 2 milisegundos mas o menos, el contenido de este bit alterna constantemente entre 0 y 1.

Este cambio, que no se puede observar en la tarjeta MDA, es el que la rutina de test aprovecha, leyendo primeramente el contenido de este registro, y apagando los bits 0 a 6. El resultado así obtenido se guarda como valor comparativo para las como máximo 32768 pasadas por el bucle, en el que se lee continuamente el valor del registro de estado. Si se descubre una modificación con respecto al valor comparativo es decir, se ha modificado el estado del bit 7 se ha de tratar de una tarjeta HGC. Si de lo contrario, el contenido de este bit no se modifica durante los 32768 bucles, se trata de una tarjeta MDA.

También aquí se carga de nuevo el código correspondiente para la tarjeta de vídeo encontrada en el vector. Por norma se indica como código de monitor el código del monitor monocromo, ya que es el único que se puede conectar a las tarjetas MDA y HGC.
Las pruebas en sí se han terminado con ello, y las tarjetas de vídeo se han averiguado. Lo que no queda es saber cual es la tarjeta de vídeo activa (primaria) y la inactiva (secundaria). Si el test de la VGA tuvo éxito, esta tarea es innecesaria, ya que la diferenciación entre tarjeta de vídeo activa e inactiva se toma directamente en esta rutina.

En todos los demás casos, la tarjeta de vídeo activa se puede averiguar mediante el modo de vídeo actual, que se puede obtener con ayuda de la función 0Fh de la interrupción de vídeo del BIOS. Si como resultado devuelve el valor 7, está activo el modo de texto de 80 × 25 caracteres de la tarjeta monocroma. Cualquier otro modo permite suponer, que hay una tarjeta CGA, EGA o VGA activa. Partiendo de esta información se intercambia el orden de las dos entradas en el vector en el último paso de la rutina de ensamblador, si no coincide con las circunstancias reales.

Con ello ha terminado la tarea de la rutina de ensamblador. Devuelve el control a la función invocadora.
Después de estas extensas explicaciones no le debería resultar difícil hacerse una idea del trabajo de los dos programas de ensamblador que hay a continuación. Les precede un programa en lenguaje de alto nivel, que llama la función GETVIOS desde el módulo en ensamblador, y con ello demuestra su forma de aplicación.

Estructura fundamental y funcionamiento de una tarjeta de vídeo
La una domina la representación de texto, la otra no pasa de un pobre modo gráfico, y la tercera genera imágenes de calidad fotográfica las diferencias entre los diferentes estándares de vídeo son exagerados. Y a pesar de ello: cada tipo de tarjeta de vídeo, sea una simple MDA o una Super-VGA moderna, trabaja según un principio unificado. Por ello, y antes de comenzar a estudiar los diferentes estándares de vídeo en los siguientes capítulos, este capítulo le quiere suministrar algunas informaciones sobre la estructura fundamental y el modo de funcionamiento de tarjetas de vídeo. Podrá aprender:
- Cómo genera un monitor la imagen de vídeo,
- Cómo controla la construcción de pantalla la controladora CRT,
- Qué tareas tienen los registros de la controladora CRT, y que es lo que pertenece además a una tarjeta de vídeo.

En la segunda parte del capítulo trataremos de la RAM de vídeo, que es de importancia primordial para la creación de la imagen de vídeo y la programación de tarjetas de vídeo.
Construcción de la pantalla mediante el monitor
El monitor representa el final de la cadena, que ha de recorrer una imagen en su camino a la visibilidad. Si no se tienen en cuenta los monitores Multisync, en realidad de trata de un dispositivo «tonto», que no dispone de ningún tipo de inteligencia, y que no puede ser programado. Los monitores que se emplean en el ámbito del PC son todos ellos los llamados «raster-scan-devices», en los cuales la pantalla se compone de multitud de puntos pequeños, ordenados en una retícula rectangular.

Al construir la imagen, el haz de electrones de la pantalla recorre cada punto de ella, y lo enciende si es que ha de estar activa. En la práctica, esto se realiza activando el haz de electrones en este punto, lo que hace que las partículas de fósforo del tubo de pantalla se iluminen.
Mientras que un monitor monocromo tiene suficiente con un haz de electrones, un monitor en color ya necesita tres haces de electrones que recorran la pantalla. Pero aquí el punto de pantalla no se compone sólo de una partícula de fósforo, sino de tres, en los colores básicos rojo, verde y azul. Cada uno de los haces de electrones es el responsable para uno de los colores, por lo que estos monitores también son conocidos como RGB (Red-Green-Blue). Mediante la combinación de estos colores y las diferentes intensidades, se pueden obtener todos los demás colores, siempre y cuando la tarjeta de vídeo lo permita.

Pero como una partícula de fósforo ionizada sólo brilla durante muy poco tiempo, la pantalla completa se ha de recorrer muchas veces por segundo, para sugerirle al ojo una imagen fija. En muchos monitores de PC esto ocurre entre 50 y 70 veces por segundo, y se puede decir que la calidad de imagen aumenta proporcionalmente con la velocidad de refresco, ya que la imagen tiene un aspecto más definido. Cada construcción de pantalla comienza en la esquina superior izquierda

de la pantalla. Desde allí el haz de electrones se mueve por la primera línea de pantalla horizontalmente hacia la derecha. Al final de esta línea salta de nuevo al principio de la línea, y además una hacia abajo. Este proceso se llama retrazado horizontal o «horizontal retrace».
Después de uno de estos retrazados, el haz de electrones comienza con la construcción de la siguiente línea de pantalla, al final de la cual el haz se desplaza según el mismo esquema al inicio de la siguiente línea. Si de esta forma alcanza al final de la pantalla, salta de nuevo a la esquina superior izquierda de la pantalla, donde comienza inmediatamente otra construcción de pantalla. Este proceso, análogamente al retrazado horizontal se denomina retrazado vertical, que en inglés lleva la denominación «vertical retrace».

Sin embargo se ha de tener en cuenta que el desplazamiento del haz de electrones no se controla por el monitor en sí, sino por la controladora de la tarjeta de vídeo mediante unas líneas de señal especiales, como veremos a continuación. Así que todo el proceso carece de algún automatismo que tenga su origen en el monitor.
De la cantidad de líneas y columnas de retícula que el haz de electrones recorre, depende directamente la resolución del monitor. Un monitor que sólo disponga de 200 líneas a 640 columnas cada una no puede procesar las altas resoluciones de duna tarjeta EGA o VGA de hasta 640 × 480 puntos. Los diferentes tipos de monitor que se emplean en el ámbito del PC disponen habitualmente de las siguientes resoluciones. Véase la tabla 58.

La controladora CRT
El camino del haz de electrones por la pantalla se controla por el llamado controlador CRT, que representa el corazón de la tarjeta de vídeo. Su nombre se lo debe a la expresión inglesa para pantalla cathode-ray tube. Además también se le conoce bajo el nombre de controlador de vídeo o pantalla, o abreviado: CRTC.
Controla los procesos en la tarjeta de vídeo y genera las señales de tiempo que necesita el monitor para construir la pantalla. Entre sus tareas se cuenta, además del controlar el lápiz óptico (si existe), de generar el cursor de pantalla intermitente en el modo de texto y de controlar la RAM de vídeo, en la que se guardan las informaciones de pantalla.
Para iniciar la construcción de un línea de pantalla en el monitor, el CRTC al principio de cada línea, emite una señal llamada «display-enable», que activa el haz de electrones. Mientras se va moviendo de izquierda a derecha por las diferentes columnas de pantalla de la línea, el CRTC controla las diferentes señales para el o los haces de electrones, de modo que los puntos de pantalla aparecen en pantalla en el color deseado. Al llegar al final de la línea, el haz de electrones se lleva al borde izquierdo de la siguiente línea de pantalla mediante una señal de sincronismo horizontal. Pero antes se vuelve a anular la señal de display-enable, para que al retroceder el haz de electrones no se genere una línea visible en la pantalla. Una vez llegado al inicio de la siguiente línea de pantalla, esta señal se vuelve a conectar, y se comienza con la construcción de la siguiente línea.

Ya que el tiempo que necesita el haz de electrones para el retrazado es más corto que el tiempo que necesita el CRTC para obtener y preparar nuevas informaciones de la RAM de vídeo, se crea una pequeña pausa. Pero como el haz de electrones no se puede detener, se crea el llamado Overscan, que se puede ver como borde izquierdo y derecho al lado del contenido de la pantalla en si. En realidad no deseado, este Overscan tiene además un desagradable efecto secundario, que es un borde de pantalla a la izquierda y la derecha de la imagen en sí. Si el haz de electrones se deja activado durante el recorrido de este borde, se puede visualizar un marco de pantalla con color.

Si el haz de electrones ha alcanzado la última línea de pantalla, es hora de volver a moverlo a la esquina superior izquierda de la pantalla. También aquí primero se ha de anular la señal de display-enable, y después se envía una señal de sincronismo vertical. Una vez alcanzada la esquina superior izquierda, se vuelve a conectar la señal de dsplay-enable, y se inicia una nueva construcción de pantalla.

Durante el retrazado horizontal del haz de electrones se crea, al igual que durante el retrazado vertical, una especie de agujero temporal, que se ha de rellenar. Por ello, aparte del Overscan horizontal se crea también un Overscan vertical, que se hace visible en un borde de pantalla vertical.

Los registros de la controladora CRT

La secuencia temporal de las diferentes señales de sincronización y el marco temporal para la construcción de la pantalla no están determinadas estáticamente, sino que varían entre las diferentes tarjetas de vídeo y sus diferentes modos de vídeo. Por esta razón el CRTC dispone de una serie de registros, que describen la salida de señal y su secuencia temporal.

En las siguientes páginas se describe la estructura de estos registros y su programación. Se toman como referencia los registros de la controladora de vídeo 6845 de Motorola. Se puede encontrar en las tarjetas gráficas MDA, CGA y Hercules. Las tarjetas EGA y VGA disponen como CRTC de un chip especial VLSI (very large scale integration), cuyos registros, condicionados por las mayores posibilidades de estas tarjetas, tienen una estructura ligeramente diferente. Sin embargo, los registros más importantes del 6845 también se pueden encontrar en estos controladores. Véase la tabla 59.

El acceso a los registros de la controladora CRT se realiza mediante los llamados puertos (ports) con ayuda de las instrucciones de lenguaje máquina IN y OUT. y esto también es aplicable para todos los demás registros de una tarjeta de vídeo. Los diferentes registros del CRTC no se pueden direccionar directamente, sino que se han de direccionar primero con ayuda de un registros de direcciones especial. Para ello se ha de enviar el número del registros CRTC direccionado al puerto del registros de direcciones. Después se puede obtener el contenido del registro con ayuda de la instrucción de lenguaje máquina IN, en un registros de datos especial. Si se quiere escribir un valor al registro direccionado, se ha de transferir al registro de datos mediante la instrucción OUT, de donde el CRTC lo toma automáticamente al registros direccionado.

A pesar de los registros de datos y direcciones se encuentran siempre en direcciones de puerto consecutivas, estas direcciones difieren en las distintas tarjetas de vídeo. En el caso de las tarjetas monocromas son los ports 3B4h/3B5h, los que se pueden direccionar a través de estos dos registros, mientras que las direcciones de puerto en tarjetas de color son 3D4h y 3D5h.

Ya que durante la discusión sobre las diferentes tarjetas de vídeo siempre aparecerán tablas, en las que se describe el contenido de los diferentes registros CRTC en los diferentes modos de vídeo, en el siguiente ejemplo queremos mostrar como se calcula el contenido de estos registros, y como están conectados los diferentes registros entre si. Si quiere hacer estos cálculos con su calculadora o PC, se dará cuenta de que algunos cálculos no salen exactamente, es decir no devuelven resultados enteros. Pero como que los registros del CRTC sólo pueden acoger valores enteros, siempre se redondea hacia arriba o abajo.

La base para los diferentes cálculos es la anchura de banda para la tasa de desplazamiento (scan) vertical y horizontal de un monitor. Véase la tabla 60.

La anchura de banda de la representación anterior indica la cantidad de puntos que puede desplazarse el haz de electrones de un tubo de pantalla en un segundo, y por ello también se denomina como tasa de puntos (Dot-Rate). Ha de crecer con resoluciones de pantalla mayores y también depende de la tasa de desplazamiento vertical, que representa el número de repeticiones de imagen por segundo. Cuantos más cuadros por segundo se generen, más puntos se ha de desplazar el haz de electrones, por segundo.

Como tercer componente de la tabla anterior, la tasa de desplazamiento horizontal se refiere al número de líneas de pantalla que recorre el haz de electrones por segundo. Depende de la anchura de banda y de la cantidad de puntos de pantalla por línea.

Como muestra la tabla 60, las tarjetas MDA, CGA y Hercules trabajan con sólo una anchura de banda, mientras que en las tarjetas EGA y VGA ya son dos las anchuras de banda. En las tarjetas Super-VGA se les añade a las anchuras de banda normales otras tres anchuras de banda adicionales.

Habitualmente se puede ver en la placa de la tarjeta cuántas anchuras de banda soporta una tarjeta de vídeo. Al fin y al cabo, para cada anchura de banda se necesita un cuarzo propio, que oscile al ritmo de la anchura de banda, y que alimente al controlador CRT. Los cristales de cuarzo se pueden identificar fácilmente en su sitio, ya que al contrario que los demás componentes no son negros, sino metálicos, y llevan su frecuencia impresa sobre ellos.

Cálculo de ejemplo

Si no se tiene en cuenta el modo gráfico de la tarjeta Hercules, que sólo se puede activar mediante programación directa de los diferentes registros CRTC, es normalmente el BIOS el que hace la inicialización de estos registros, cuando activa un modo de vídeo mediante la función 00h del BIOS de vídeo. A pesar de ello es interesante el realizar los cálculos para los diferentes valores de los registros del CRTC, lo que queremos demostrar tomando como ejemplo el modo de texto de 80 × 25 caracteres de la tarjeta CGA.

Sin embargo buscará en vano una entrada para la resolución de 80 × 25 caracteres en la tabla anterior. La controladora CRT y también el monitor no conocen caracteres, sino sólo puntos de pantalla individuales. Pero los caracteres se componen de puntos de pantalla, que se guardan en una matriz rectangular. Si la controladora CRT quiere llevar un carácter a la pantalla, desmonta por ello la matriz de caracteres en puntos individuales.

En una tarjeta CGA la matriz de puntos es de 8 × 8 puntos, de modo que la resolución de 80 × 25 caracteres corresponde a una «resolución gráfica» de 640 × 200 puntos. Para este modo la anchura de banda es de 14,3168 MHz con una frecuencia de repetición de imagen de 60 Hz y una tasa de desplazamiento horizontal de 15,740 KHz.

Si se divide la anchura de banda entre la tasa de desplazamiento horizontal, se obtienen la cantidad de puntos de pantalla por línea de pantalla, que es mayor que la resolución esperada, ya que aquí se incluye el Overscan.

Anchura de banda 14,318 MHz
÷ tasa de despl. horizontal 15,750 KHZ

Puntos de pantalla por línea 909

Ya que los registros CRTC en gran parte no se refieren a los puntos de pantalla, sino a la cantidad de caracteres, este valor se ha de convertir en caracteres por línea. Esto se puede hacer con una división por la anchura de la matriz de caracteres, que como ya hemos mencionado, en la tarjeta CGA es de 8 puntos de pantalla.

Puntos de pantalla por línea 909
÷ Puntos por línea 8

Caracteres por línea 114

Este valor se ha de reducir en 1, y se guarda en el registro del CRTC, indicando así la cantidad total de caracteres por línea. En el segundo registro se carga la cantidad de caracteres por línea que han de aparecer en la pantalla. En el modo de 80 × 25 caracteres de texto, esto son 80 caracteres.

De la diferencia entre el número total de caracteres por línea y el mostrado, se obtiene el número de caracteres que se podrían visualizar durante la duración del retrazado horizontal y del Overscan. Son aquí 34 caracteres.

En el cuarto registro del CRTC se ha de grabar la duración del retrazado horizontal del haz de electrones. El registro en sí no acoge la duración, sino el número de caracteres que se podrían haber visualizado durante este tiempo. No se puede definir libremente por la tarjeta de vídeo, sino que habitualmente viene dada por las características del monitor. Habitualmente está entre el 5 y el 15% del número total de caracteres por línea. En un monitor en color son exactamente 10 caracteres.

Con ello quedan 24 caracteres para el Overscan, es decir, para el marco horizontal de pantalla. Cómo se distribuyen estos caracteres en el borde derecho e izquierdo, lo decide el tercer registro del CRTC, que indica después de cuántos caracteres se inicia el retrazado horizontal del haz de electrones. El BIOS especifica aquí el valor 90, de modo que después de los caracteres mostrados aún quedan 10 caracteres para el marco de pantalla. Los 14 caracteres que quedan se encuentran por ello antes del principio de la siguiente línea y forman el borde izquierdo de la pantalla.

Similar a los cálculos para las características horizontales, se realizan los cálculos para los datos verticales, es decir la cantidad de líneas verticales, la posición de la señal de sincronización vertical, etc.
Los cálculos comienzan con el cálculo de la cantidad de líneas de pantalla por pantalla. Resulta de la división de la cantidad de líneas que se pueden construir por segundo, y de las repeticiones de imagen por segundo.

tasa de despl. horizontal	15,750 KHz
÷ repeticiones de imagen	60 Hz
---	---
líneas de pantalla	262

Dado que los caracteres en el modo de texto de la CGA no solo tienen 8 puntos de anchura, sino también 8 puntos de altura, de nuevo se ha de realizar una división entre 8, que devuelve la cantidad de líneas de texto por pantalla.

Líneas de pantalla	262
÷ puntos por carácter	8
---	---
Líneas por pantalla	32

Este resultado se reduce en 1, y después se carga en el quinto registro del CRTC. En el séptimo registro se guarda la cantidad de líneas mostradas (25) por pantalla. Como en la pantalla se muestran 7 líneas menos de las que en realidad existen, estas líneas se utilizan de nuevo para el retrazado vertical y el Overscan, donde el retrazado vertical del haz de electrones se inicia en la línea 28.
La altura de los diferentes caracteres, que en estos cálculos tiene un papel fundamental, se ha de reducir en 1 (aquí 7) y guardarla en el registro 9 del CRTC. Cierra hacia arriba el margen en que se pueden mover los valores en los registros 10 y 11. Indican la primera y la última línea de retícula, entre la que el cursor intermitente de pantalla se superpone al carácter que tiene bajo él. De qué carácter se trata, lo define el contenido de los registros 14 y 15 que indican la posición del cursor intermitente de pantalla. Sin embargo no se refieren a línea y columna, sino a la distancia del carácter del borde superior izquierdo de la pantalla. Este valor se calcula, multiplicando la línea que

ha de contener al cursor con el número de columnas por línea, y se suma a la columna del cursor. El byte alto del resultado se ha de cargar en el registro 14 y el byte bajo en el registro 15.

Estructura de una tarjeta de vídeo

A pesar de que la controladora CRT realiza toda una serie de tareas en el ámbito de la construcción de la pantalla, no es capaz de realizar todo el trabajo él solo. Una tarjeta de vídeo se compone además de varias unidades funcionales, que trabajan con la controladora CRT, y que están conectadas entre si. La siguiente figura muestra la representación simplificada de una tarjeta de vídeo con ayuda de un diagrama de bloques.

El punto de partida más importante para la creación de una imagen es la RAM de vídeo, una zona de memoria, que está contenida directamente en la tarjeta de vídeo y que trataremos con más detalles en el resto del capítulo. En el modo de texto acoge los códigos ASCII de los caracteres y su color, mientras que en el modo gráfico dispone de algunos bits para reflejar el color de cada punto.

A la RAM de vídeo acceden en el modo de texto tanto el generador de caracteres como el llamado controlador de atributos. El primero carga, a petición de la controladora CRT, un carácter de la RAM de vídeo, y lo divide en puntos individuales. Para ello hace servir un circuito ROM, en el que se retienen todos los patrones de caracteres de todos los caracteres disponibles del juego de caracteres ASCII, en forma de una tabla. El resultado de esta transformación, el generador de caracteres lo envía directamente al controlador de señal, que es controlado por la controladora CRT, y que representa el puerto a las diferentes líneas del cable del monitor.

Al controlador de señal también le llegan las indicaciones del controlador de atributos. En el modo de texto tiene la tarea según ordenes de la controladora CRT de obtener el color de un carácter y de enviarlo al controlador de señal. Tampoco es otra su tarea en el modo gráfico, sólo que ha de tener en cuenta la estructura cambiada de la RAM de vídeo.

En las primeras tarjetas de vídeo aún se puede reconocer muy bien la división de las diferentes unidades funcionales, por que estaban incluidas en diferentes circuitos integrados (IC). Con la miniaturización que avanza, casi todos los componentes se colocan hoy en día en un solo circuito integrado. Esto también se aplica a tarjetas EGA y VGA, que tienen una estructura más compleja de lo que permite suponer el diagrama de bloques anterior.

En los siguientes capítulos aprenderá más sobre la estructura concreta de las diferentes tarjetas de vídeo y las diferencias que existen entre ellas. Pero primero vamos a dedicarnos a la RAM de vídeo, que es muy importante para la programación de tarjetas de vídeo, y que en el modo de texto es el único nexo entre todos los tipos de estándares de vídeo.

La RAM de vídeo

La RAM de vídeo es el lugar, en el que todas las tarjetas de vídeo desde MDA hasta Super-VGA guardan las informaciones de la pantalla. Esto se aplica tanto al modo de texto, como al modo gráfico. Para la programación directa de una tarjeta gráfica, es decir, evitando la ROM BIOS, es fundamental el conocimiento de la estructura y la posición de la RAM de vídeo. Mientras que la RAM de vídeo se organiza de modo totalmente distinto en los diferentes modos gráficos de las tarjetas de vídeo, en los modos de texto de todas las tarjetas se presenta una estructura homogénea.

Por ello, este capítulo le describe como se puede acceder a la RAM de vídeo desde lenguajes de alto nivel, y además responde las siguientes preguntas:

- ¿Dónde está la RAM de vídeo en el espacio de direcciones del PC?
- ¿Cómo y cuándo se puede acceder a la RAM de vídeo?
- Estructura de la RAM de vídeo en el modo texto

Si quiere saber más sobre la organización de la RAM de vídeo en los diferentes modos gráficos, por favor consulte los siguientes capítulos. Allí también se explica en conjunto con la descripción de la

tarjeta de vídeo correspondiente como se puede poner y obtener puntos individuales en los diferentes modos gráficos.

La RAM de vídeo en el espacio de direcciones del PC

Como tal vez sepa, la memoria RAM del PC no necesariamente ha de estar en la placa madre del ordenador, sino que también se puede encontrar en una de las tarjetas de expansión. De eso se aprovechaban todos los estándares actuales en el ámbito del PC, «visualizando» una zona de RAM variable en la zona de direcciones del PC, y utilizándola como RAM de vídeo.

A pesar de que las diferentes tarjetas de vídeo determinan por ellas mismas donde se ha de «insertar» esta RAM de vídeo en la zona de direcciones del PC, no son totalmente libres en su decisión. Ya que en el estrecho corsé que los primeros desarrolladores le pusieron al PC, no queda mucho espacio. Esto es así, porque estaban obligados a incluir RAM, ROM y las extensiones del sistema por debajo de un Megabyte. Para las tarjetas de vídeo sólo quedaron libres los segmentos de memoria A y B, que comienzan en las direcciones de segmento A000h y B000h, y que tienen 64 KBytes cada uno.

Para darle la posibilidad al usuario de poder colocar en su ordenador una tarjeta de vídeo en color y una monocroma simultáneamente, estos dos tipos de tarjetas de vídeo se dividen el segmento B en partes iguales. Las tarjetas de vídeo monocromas, es decir las tarjetas MDA y Hercules, toman para su RAM de vídeo la zona de memoria desde B000:0000 hasta B000:7FFF. A partir de B8000:0000 (es lo mismo que B000:8000) la RAM de vídeo pertenece a las tarjetas de color, es decir CGA, EGA y VGA.

Pero en las tarjetas EGA y VGA esta posición no siempre está clara, ya que las tarjetas EGA se pueden conectar también a un monitor MDA, y se comportan como una tarjeta MDA. En ese caso, su RAM de vídeo no comienza en B800h sino en B000, como es normal en las tarjetas de vídeo monocromas. Algo parecido ocurre con las tarjetas VGA, que no se conectan a un monitor MDA, pero que a cambio pueden emular tarjetas MDA en un monitor VGA normal. También entonces su RAM de vídeo comienza en B000h.

Sin embargo, no siempre se emplean los 32 KBytes de los que dispone una tarjeta de vídeo en su totalidad. En las tarjetas MDA sólo son los primeros 4 KBytes de la RAM de vídeo los que se ocupan, mientras que la tarjeta CGA ya emplea los primeros 16 KBytes, es decir justo la mitad de la zona de direcciones disponible. Las tarjetas Hercules ya utilizan los 32 Kbytes completamente, y además están en disposición de utilizar la segunda mitad del segmento B, es decir la zona de memoria a partir de B800:0000, para una segunda página gráfica. Esta característica se puede desconectar habitualmente a través de un interruptor DIP de la tarjeta, para que esta deje libre la zona a partir de B800:0000 para una tarjeta en color.

La RAM de vídeo también se aprovecha del todo en las tarjetas EGA y VGA, que habitualmente disponen de 256 KBytes de RAM de vídeo, de los cuales sólo se puede acceder a los primeros 32 KBytes a través de su parte del segmento B. El apartado 9.8.2 mostrará, como se llega al resto de RAM de vídeo en estas tarjetas.

Direccionamiento de la RAM de vídeo

Ya que la RAM de vídeo se activa en la zona de direcciones normal del PC, se puede direccionar como memoria RAM normal, lo que sin embargo no debe confundir su especial posición. Al fin y al cabo, la RAM de vídeo completa se lee, dependiendo de la frecuencia de repetición de imagen, hasta 70 veces por segundo de los diferentes componentes de la tarjeta de vídeo, para crear la imagen basada en ella.

Durante este acceso, no es posible otro acceso simultáneo por un programa al ROM BIOS, por lo cual la mayoría de las tarjetas ya toman medidas en el hardware, para evitar colisiones durante el

acceso de la RAM de vídeo. La única excepción la forman los dos primeros representantes de la tarjeta CGA original de IBM, en los cuales el programa era el encargado de evitar posibles colisiones. Más sobre ello en el apartado 9.7.

Estructura de la RAM de vídeo en el modo de texto

La organización de la RAM de vídeo en el modo de texto representa un consenso realmente débil entre los diferentes tipos de tarjetas de vídeo. Ya que la mayoría de los programas trabajan en el modo de texto, esto precisamente les favorece a los programadores, ya que existe la posibilidad de escribir directamente a la RAM de vídeo con una sola rutina de salida, sin tener en cuenta el tipo de la tarjeta de vídeo instalada. Todos los programas con éxito, desde Lotus 1-2-3, hasta dBASE III hacen uso de esta posibilidad, ya que la velocidad de estas rutinas es mucho más elevada que las funciones equivalentes del BIOS.

Al final del capítulo queremos presentarle tres programas en los lenguajes BASIC, C y Pascal, que contienen una rutina para el acceso directo a la RAM de vídeo en el modo de texto. Naturalmente se podría aumentar la velocidad de esta rutina, realizándola directamente en ensamblador, pero aún así, estas rutinas son extremadamente rápidas.

Para poder comprender el funcionamiento de estas rutinas, se ha de imaginar la estructura de la RAM de vídeo, tal y como lo muestra la siguiente figura.

Cada posición de pantalla, como muestra la figura anterior, se representa por dos bytes en la RAM de vídeo. En el primero de los dos bytes, en la dirección de offset par, se guarda el código ASCII del carácter a visualizar. Utilizando un código de 8 bits, se pueden representar un máximo de 256 caracteres diferentes. De aquí es de donde viene el juego de caracteres ASCII de 256 caracteres del PC, que ya presentamos en el apartado 9.2.

A continuación del código ASCII, y con ello siempre en la dirección impar de offset, viene el byte de atributos, que define el aspecto del carácter en la pantalla. Se divide en dos grupos de cuatro bits (nibbles) por el controlador de atributos, donde el cuarteto alto (bits 4 a 7) describe el fondo del carácter, y el nibble bajo (bits 0 a 3) el color de texto. Así resultan dos valores en el rango de 0 a 15, que se interpretan dependiendo del monitor conectado. En una tarjeta de color (tarjeta EGA o CGA), estos dos valores seleccionan uno de 16 colores. Por ello, cada carácter de la pantalla puede disponer de un color individual de fondo y texto.

Con un monitor monocromo (en una tarjeta gráfica MDA, EGA o Hercules) naturalmente no se pueden visualizar colores. Aquí se dispone de la posibilidad adicional de representar los caracteres de forma intensa, inversa o subrayados. Por ello aquí no se habla de colores sino de atributos.

Si se quiere acceder a los caracteres en la RAM de vídeo, naturalmente se ha de conocer su orden dentro de esta zona de memoria. No se eligió aleatoriamente, sino que sigue la ordenación de los caracteres en la pantalla. El primer carácter de la pantalla, el carácter en la esquina superior izquierda, se representa por los dos primeros bytes del inicio de la RAM de vídeo. Así que se encuentra en la dirección de offset 0000h. En la posición de offset 0002h de la RAM de vídeo viene el carácter de al lado. De esta misma forma, vienen los siguientes 78 caracteres.

Ya que cada carácter ocupa 2 bytes, la primera línea de pantalla ocupa los primeros 160 bytes de la RAM de vídeo. Inmediatamente a continuación viene el primer carácter de la segunda línea de pantalla. Esta también ocupa 160 bytes de la RAM de vídeo, al igual que todas las demás líneas, que siguen a continuación.

A causa de esta ordenación estricta, se puede calcular la dirección de inicio de una línea en la RAM de vídeo multiplicando simplemente el número de línea por 160. Sin embargo es importante recordar, que los números de línea no comienzan en 1, sino en 0, con lo que se obtiene la dirección 0000h para la primera línea. Para llegar a un carácter cualquiera desde el inicio de una línea, se ha de sumar la distancia del carácter del inicio de la línea. Como que cada carácter ocupa 2 bytes (también el número de columna se cuenta desde 0), esto se puede realizar simplemente multiplicando el número de columna por 2. Si se suman los dos productos, se obtiene la dirección de offset del carácter en la RAM de vídeo.

Estos cálculos se pueden resumir en una sola fórmula de forma sencilla:

Posición de offset (Columna, Línea) = Línea × 160 + Columna × 2

En la dirección de offset descrita de esta forma, se encuentra el código ASCII del carácter deseado; un byte más allá se encuentra su byte de color o atributo.

Modos de texto ampliado

Pero las tarjetas CGA, EGA y VGA no solo conocen el modo de texto de 80 × 25 caracteres, que es el que define la fórmula anterior. Aparte del modo de texto de 40 × 25 caracteres de las tarjetas CGA, las tarjetas EGA y VGA también disponen de los modos de texto con más de 25 líneas. En la EGA son 43 líneas, y en la VGA incluso 50, aunque con ayuda de unos pequeños trucos se puede conmutar también a 43 líneas. Pero no basta con ello: las tarjetas Super-VGA conocen otra serie de modos de texto, con resoluciones de hasta 132 × 80 caracteres.

También en estos modos existe la estructura anteriormente descrita de la RAM de vídeo sin cambios. Pero naturalmente se ha de adaptar la fórmula para calcular la dirección de offset de un carácter, ya que las diferentes líneas ocupan más memoria en la RAM de vídeo. Pero es suficiente con adaptar el factor 160, y sustituirlo por el doble de la anchura de la líneas en caracteres. Así que en el modo de texto 40 × 25 de la tarjeta CGA debería ser de 80.

Sin embargo, no es necesario que tenga en cuenta estos modos cuando desarrolle programas propios, si no los conecta explícitamente mediante la programación de la tarjeta de vídeo. Habitualmente, los programas se ejecutan desde la línea de comandos del DOS, desde el modo de texto de 80 × 25 caracteres, sólo es necesario soportar este modo.

Acceso a las diferentes páginas de pantalla

Hasta ahora, en la fórmula del offset no se tuvo en cuenta que las tarjetas Hercules, CGA, EGA y VGA soportan varias páginas de pantalla, ya que en la RAM de vídeo hay mucho más espacio del que necesita una página de pantalla individual. Para acceder a varias páginas de pantalla, sólo es necesario completar la fórmula y añadir el offset inicial de la página a la dirección de offset.

Se ha de tener en cuenta, que las diferentes páginas de pantalla en el modo de texto de 80 × 25 caracteres se ordenan en la RAM de vídeo en distancias de 4 KBytes (4096 bytes). Entre las diferentes páginas de pantalla, que ocupan 4000 bytes (80 × 25 × 2), quedan 96 bytes sin utilizar, pero que puede utilizar para sus propósitos.

Para el modo de texto de 80 × 25 caracteres la fórmula del offset, teniendo en cuenta la página de pantalla:

Posición de offset (Columna, Línea) = Línea × 160 + Columna × 2 + Página × 4096

La primera página de pantalla lleva el número 0

Para el modo de texto de 40 × 25 caracteres, se ha de sustituir el factor 4096 por 2048, ya que las diferentes páginas de pantalla se siguen en distancias de 2 KBytes. De manera semejante ocurre con los modos de texto de EGA y VGA, en los que aparecen 43 o 50 líneas en la pantalla. Sin embargo, en este caso el factor es de 8 KBytes.

La tarjeta monocroma de IBM (MDA)

La tarjeta MDA, que se presentó junto con el PC en el año 1981, hoy en día es difícil de encontrar. Los estándares que ha fijado, siguen viviendo en forma de Hercules Graphics Card, que mayormente es compatible con la tarjeta MDA. En los días iniciales de PC la tarjeta MDA era muy cotizada, al menos más cotizada que la tarjeta CGA, y no sólo porque fuera mucho más barata, sino

porque en la representación de texto ofrecía una imagen mucho mejor. Quien no necesitaba color y gráficos, empleaba una tarjeta MDA.

Este apartado analiza las capacidades de la tarjeta MDA en detalle, y echa un poco de luz sobre las tareas y la programación de sus diferentes registros. Se tratarán los siguientes temas:
o Representación de texto en la tarjeta MDA
o Tamaño y posición de la RAM de vídeo
o Estructura del byte de atributos en el modo de texto
o Tarea del registro de control y estado
o Acceso al controlador CRT
Representación en la tarjeta MDA
La mejor representación de texto con respecto a la tarjeta CGA resulta del empleo de una matriz de 9 × 14 puntos, que permite una alta resolución para la representación de los diferentes caracteres. El formato de esta matriz es tanto más desacostumbrado, cuanto con ayuda del generador de caracteres que contiene los patrones de bits de cada carácter sólo se pueden generar caracteres de una anchura de 9 puntos. Pero esto tendría como consecuencia en el juego de caracteres de IBM, que por ejemplo los signos de marco en dirección horizontal ya no estuvieran conectados.

Por ello se ha implementado un circuito en la tarjeta, que salva esta desventaja. Simplemente copia en todos los caracteres que hay entre los códigos ASCII B0h y DFh (la gran mayoría son los caracteres de marco) el octavo punto al noveno. ya que en los caracteres de marco el octavo punto está a uno de todas formas, el noveno también lo estará.
El generador de caracteres necesita, a causa de este modo de proceder, para la codificación de una línea de puntos de un carácter, un byte, ya que codifica cada uno de los ocho puntos con ayuda de un bit. El patrón completo de un carácter con sus 14 líneas de puntos ocupa por ello 14 bytes. Para el juego completo de caracteres resulta de ello una necesidad de memoria de apenas 4 KBytes, que están guardados en un circuito ROM en la tarjeta. No es de entender entonces, que las tarjetas MDA originales de IBM no contengan 4 sino 8 KBytes de ROM, donde los segundos 4 KBytes permanecen sin utilizar.

Posición y tamaño de la RAM de vídeo
La RAM de vídeo de la tarjeta MDA comienza fundamentalmente en la dirección B000:0000 y comprende 4 KBytes (4096 bytes). Con ello sólo se puede gestionar en memoria una página de pantalla con la resolución de 80 × 25 caracteres de texto. Además, esta tarjeta desconoce otros modos de representación. En referencia a la función de inicialización del BIOS de vídeo (función 00h) este modo lleva el número 7.
Ya que una página de pantalla, con sus 2000 caracteres sólo ocupa 4000 bytes en la RAM de vídeo, los 96 bytes sin utilizar quedan disponibles para su libre utilización.

Estructura del byte de atributos en el modo de texto
Como muestra la siguiente figura, los bits 0 a 2 y 4 a 6 del byte de atributo de un carácter el color de texto y de fondo de la tarjeta MDA.
A pesar de que se pueda cargar una combinación de bits cualquiera en estos bits, la tarjeta MDA, al contrario de la tarjeta CGA, no reconoce códigos de color entre 0 y 15, sino sólo las siguientes combinaciones de atributos determinadas de antemano.
Independientemente de la combinación de atributos elegida, se pueden activar el bit 3 y 7 del byte de atributos. El bit 3 define la intensidad de la representación de texto. Si está a uno, el carácter se representa de forma especialmente intensiva. El significado del bit 7 depende del contenido del llamado registro de control, del que hablaremos enseguida. Según el ajuste que aquí se haya realizado, puede tener como consecuencia la intermitencia del carácter o una representación en alta intensidad.

Tarea del registro de estado y control

Aparte de los registros de la controladora CRT 6845 presentados en el apartado 9.4, la tarjeta monocroma de pantalla dispone de dos registros más: el registro de control y el de estado.

El registro de control, que se encuentra en el puerto 3B8h, se puede escribir y leer, y sirve para el control de diferentes funciones de la tarjeta de pantalla monocroma. Como se puede ver en la figura anterior, sólo los bits 0, 3 y 5 tienen significado. El bit 0 controla la resolución de la pantalla. A pesar de que esta tarjeta sólo soporta una resolución (25 líneas a 80 columnas), este bit siempre ha de estar a 1, ya que de lo contrario el ordenador se queda «colgado».

El bit 3 controla la generación de la señal de vídeo y con ello la construcción de la imagen visible. Si se pone a 0, la pantalla aparece en negro y el cursor intermitente desaparece. Si se pone de nuevo a 1, la controladora CRT comienza de nuevo con su trabajo, y recupera la imagen original.

Una función totalmente distinta es la del bit 5 del registro de control. Decide sobre la intermitencia de los caracteres cuyo bit 7 del byte de atributos está a uno. Si el bit 5 del registro de control está a 0, estos caracteres no hacen intermitencia, sino que se representan con un color de fondo intenso. El bit 7 del byte de atributos funciona entonces como bit de intensidad para el fondo. El valor por defecto de este registro, que se fija durante la inicialización del modo de texto por el BIOS, es 29 h, de modo que todos los bits están a 1.

El registro de estado no influencia de modo muy diferente al registro de control el modo de funcionamiento de la tarjeta MDA, sino que refleja su estado actual de trabajo. Por ello sólo se puede leer, pero no escribir.

Como muestra la figura anterior, el bit 0 indica, si en este momento hay activa una señal de sincronización. Esta señal se activa por la tarjeta de vídeo después de dibujar una línea de puntos, para que el haz de electrones del tubo de pantalla salte al inicio de la siguiente línea de puntos. Durante este proceso, este bit contiene el valor 1.

El bit 3 refleja el valor del pixel que es dibujado en este momento por el haz de electrones. Un 1 señaliza que el pixel se puede ver en pantalla, y 0 significa que la pantalla permanece negra en este punto.

Acceso al controlador CRT

El registro de direcciones del 6845 se encuentra en la dirección de puerto 3B4h en la tarjeta MDA, el registro de datos en la dirección de puerto 3B5h. A pesar de que se encuentran en direcciones de puerto consecutivas, aquí no se puede enviar directamente el número del registro a direccionar y el nuevo contenido del registro con ayuda del comando de 16 bits OUT, al puerto 3B4h, como es habitual en otras tarjetas de vídeo. Sin embargo, la salida se ha de realizar con ayuda de dos comandos de 8 bits, OUT, entre los que se ha de realizar una pequeña pausa (5 o 6 ciclos de reloj bastan), para que el CRTC pueda reaccionar a la salida del registro de direcciones. En el programa en ensamblador esta pausa se realiza normalmente con la instrucción de salto

```
JMP $+2
```

detrás del comando OUT. Este comando de salto transfiere la ejecución del programa a la siguiente instrucción, lo que cuesta algo de tiempo, pero no modifica en nada la ejecución del programa. Este tiempo puede ser utilizado por el CRTC, para preparar internamente el acceso al registro deseado.

La programación del registro CRTC en esta tarjeta sólo tiene sentido en lo que se refiere a la línea inicial y final del cursor intermitente de pantalla y su posición de pantalla. Ambas tareas se pueden realizar de forma más sencilla con ayuda de las funciones de la interrupción del BIOS 10h con poco trabajo, lo que además tiene la ventaja de no estar limitado a un hardware determinado.

Para todos los aficionados que juegan con los diferentes registros del CRTC, y que alguna vez quieren ver una pantalla con 81 columnas o 26 líneas, aquí está la ocupación de registros de los

diferentes registros del CRTC en el modo de texto de 80 × 25 caracteres de la tarjeta MDA. Véase la tabla 61.

Programación de tarjetas EGA y VGA

Las tarjetas EGA y VGA sobrepasan con mucho las capacidades de sus antecesores en el ámbito de los gráficos y de texto. A pesar de que ya hace tiempo que existen sistemas gráficos de potencia similar en otros campos de los ordenadores (Workstations, aplicaciones CAD/CAM, etc.) pero ahora, estos sistemas están introduciéndose en unos precios, que los hacen asequibles para el usuario normal. Se ve claramente, que la tarjeta VGA ha adelantado a la tarjeta EGA en mucho, y que los nuevos ordenadores apenas se suministran con EGA. No es de extrañar, ya que las tarjetas VGA se puedan conseguir hoy en día por unas 20.000 Ptas.

Característica para esta nueva generación de tarjetas gráficas no es sólo una resolución no alcanzada hasta ahora, la capacidad de representar un número casi cualquiera de líneas de texto en la pantalla, sino sobre todo la flexibilidad con la que emulan otras tarjetas de vídeo. Sin embargo, esto no debe confundirnos de que la funcionalidad aumentada también trae consigo una mayor complejidad de las estructuras organizativas. Esto tiene como consecuencia que las características de una tarjeta EGA y VGA ya no se puedan realizar con la controladora de vídeo tradicional del ámbito del PC, el Motorola 6845.

En vez de ello, en las tarjetas EGA y VGA se encuentran un total de cuatro controladores, que se dividen entre sí las tareas que se generan al crear la imagen de vídeo. Son la controladora de vídeo, el secuenciador (Sequencer), el controlador de gráficos (Graphics Controller) y el controlador de atributos (Attribute Controller). En las tarjetas VGA además se puede encontrar el DAC (abreviatura de: digital to analog converter), una unidad funcional que es capaz de convertir señales digitales de color en analógicas.

La mayor complejidad de la tarjeta EGA/VGA también trae consigo una mayor complejidad de la programación directa de los diferentes controladores y registros, aumentado por la circunstancia de que no todos los fabricantes de tarjetas de vídeo compatibles EGA/VGA se mantienen en las normas de IBM. Como mostrarán los diferentes subcapítulos de este capítulo, muchas veces no es posible evitar una programación directa de los diferentes registros, si se quiere aprovechar a fondo las características de la tarjeta. A pesar de que en muchos puntos existe soporte de las funciones del BIOS especial de EGA y VGA, que ha sido ampliado sustancialmente con respecto al BIOS de vídeo normal, pero siempre hay situaciones en las que sus servicios no son suficiente.

Los subcapítulos de este capítulo se ocupan de las diferentes características de las tarjetas EGA y VGA, con las cuales se destacan de sus antecesoras. En el ámbito del texto se puede contar sobre todo su capacidad de trabajar con juegos de caracteres cualesquiera, que se describe en el apartado 9.8.2. En el ámbito gráfico se tratará de los nuevos modos gráficos de 16 colores, así como del modo de 256 colores de la VGA. Se descubrirán algunos trucos, como por ejemplo, como sacar el doble de resolución de una tarjeta VGA en el modo de 256 colores.

A propósito de VGA original: a pesar de que IBM fijó los estándares con sus tarjetas EGA y VGA, en los que se basan todos los imitadores, las capacidades de estos han deja do muy atrás a las de los originales. Por desgracia se ha olvidado el fijar un estándar para los modos ampliados con sus resoluciones de hasta 1024 × 768, de modo que cada fabricante va a la suya. El apartado 9.9 intenta buscar cosas comunes y como sacar partido de ellas.

En el orden del día de este apartado también se encuentra el tema «Sprites» (Sprites), ya que en las tarjetas EGA y VGA es la primera vez que merece la pena, en el ámbito de los PC, el invertir esfuerzos en un programa de animación. El apartado 9.8.9 muestra como se hace.

En muchos subcapítulos se habla de la programación de diferentes registros y de su papel en el marco de determinadas problemáticas. En el último apartado se vuelven a listar todos los registros del estándar EGA/VGA, y se describen sus funciones en detalle.

A este apartado también pertenece la parte de consulta de las funciones del BIOS ampliada de tarjetas EGA y VGA en la parte de referencia de este libro. Muchas de las funciones que allí se

mencionan también se describen en los diferentes subcapítulos de este capítulo, pero es imposible el citar aquí a todas las funciones en detalle. Para ello debería consultar primero allí, si no está seguro de si un servicio determinado es ofrecido por el BIOS ampliada de EGA/VGA, antes de comenzar con la programación directa de los registros correspondientes de hardware.

Del monitor depende...

Que los monitores de las tarjetas EGA y VGA son importantes, lo sabe todo el mundo que alguna vez haya disfrutado de la incesante verborrea de un vendedor de PC, y ya se ve en una jungla llena de colores con 800 × 600 puntos o más. Ya que no es la tarjeta de vídeo ella sola la que se encarga de la imagen, sino sobre todo el monitor. Y ahí es donde hay que introducir un poco más la mano en el bolsillo, si se quieren ver 800 × 600 puntos o más, a ser posible con 256 colores diferentes.

Pero no sólo para los modos gráficos que están más allá de los estándares convencionales de la EGA y VGA el monitor conectado tiene importancia, sino que todo el modo de funcionamiento de una tarjeta EGA/VGA queda influenciado por él. Este apartado se ocupa por ello de:

o Cómo influencia la conexión de un monitor determinado la capacidad de una tarjeta EGA y VGA

o Cómo se puede reconocer en tiempo de ejecución de un programa si hay activa una tarjeta EGA o VGA, y de qué tipo es el monitor conectado.

Ignorando si es o no Multiscan, para comenzar es muy importante si hay conectado un monitor monocromo o de color. En las tarjetas EGA y VGA existen varias posibilidades.

Los monitores de las tarjetas EGA

Las tarjetas EGA se pueden conectar a un monitor CGA, EGA o Multisync, así como a un monitor monocromo MDA, y todo esto simultáneamente. Según qué monitor esté activo, se comportará como una tarjeta MDA ampliada, o como una tarjeta EGA sencilla. La conmutación no sólo se reflejará en la imagen de vídeo, sino sobre todo en los registros internos de la tarjeta EGA y la RAM de vídeo. Así por ejemplo la RAM de vídeo en el modo monocromo no se encuentra, como siempre, en B800h, sino en B000h. Además, los bytes de atributos de los caracteres en la RAM de vídeo se interpretan como en la tarjeta MDA, e incluso en registro de datos e índice modifica su posición, para que sea conforme al estándar MDA.

Así que como la tarjeta en un monitor monocromo se comporta mayormente como una tarjeta MDA, a esta configuración no se le prestará mayor atención en los siguientes capítulos. Así que cuando en este libro se hable de tarjeta EGA, siempre se referirá a una tarjeta EGA en un monitor de alta resolución. Esto es tanto más cierto, cuanto muchas tarjetas compatibles EGA, al contrario que el original del IBM, ya no se pueden conectar a un monitor monocromo, y sólo pueden trabajar junto con monitores EGA o Multisync.

Del mercado han desaparecido sobre todo las tarjetas EGA de IBM que sólo disponían de 64 KBytes de RAM de vídeo, y que después se podían ampliar a 128, 192 y 256 KBytes con caros módulos de memoria RAM. Las modernas tarjetas EGA se suministran con 256 KBytes de RAM, al igual que las tarjetas CGA, y éste es el estándar del que se parte en el siguiente apartado.

Los monitores de las tarjetas VGA

La cosa es ligeramente diferente en la tarjeta VGA, que se puede conectar a un monitor monocromo, pero en el que no se puede tratar en absoluto de un monitor MDA. La tarjeta VGA pide un monitor VGA monocromo analógico, que es bastante más barato que un monitor VGA en color, pero que a pesar de todo puede procesar las altas resoluciones de la tarjeta VGA.

Sin embargo, uno de estos monitores no puede cubrir el espectro completo de 256 colores simultáneos, ya que sólo existen 64 escalas de grises en total. Pero esto, habitualmente no se nota si no se trabaja con demasiados colores distintos, cuyos tonos de verde difieran muy poco.

Una de las ventajas del monitor VGA monocromo con respecto a su rival en color consiste en que una tarjeta VGA, en unión con este monitor, se puede conmutar al modo monocromo, en el que básicamente se comporta como una tarjeta MDA muy potente. Al igual que en la tarjeta EGA, la

RAM de vídeo ya no se encuentra en B800h, sino en B000h, y también las direcciones de diferentes registros se adaptan al estándar MDA.

La tarjeta se conmuta mediante los comandos DOS MODE MONO y MODE CO80, o simplemente con ayuda de la función 00h de la interrupción de vídeo del BIOS, en la que se ajusta un determinado modo de vídeo. Exclusivamente el modo de vídeo 07h, que también se emplea en las tarjetas MDA, simboliza el modo de funcionamiento monocromo, mientras que cualquier otro modo convierte de nuevo a la tarjeta VGA en una auténtica tarjeta de color.
La siguiente tabla muestra qué monitores se necesitan para la representación de los diferentes modos de vídeo de las tarjetas EGA y VGA. Véase la tabla 67.

Ya que las capacidades de una tarjeta VGA en el modo monocromo, tiene sus posibilidades técnicas muy limitadas, en el transcurso de este apartado partimos implícitamente de que su tarjeta VGA ya ha sido conmutada al modo color antes de ejecutar los programas de ejemplo. También se hace una suposición semejante en cuanto a la tarjeta EGA, en la que se parte de un equipamiento RAM de 256 KBytes y una conexión a un monitor EGA o Multisync.
Si quiere ir sobre seguro en sus programas, y en cualquier caso quiere estar seguro de que realmente haya instalada una tarjeta EGA o VGA, el siguiente apartado le muestra como se puede integrar un control semejante en sus programas.

Identificar tarjetas EGA y VGA
Si recurre en sus programas a las diferentes características de las tarjetas EGA y VGA, que son propias de éstas, y que posiblemente presuponen la conexión de un monitor determinado, debería colocar al principio de su programa una llamada a una rutina de control correspondiente. Con su ayuda puede diferenciar, en tiempo de ejecución de su programa, las tarjetas EGA de las VGA, y de sus predecesores. Esta rutina es la que queremos presentarles en el siguiente apartado.

Lleva el nombre IsEgaVga y se realiza como una función en una versión en Pascal y otra en C. Su base la forman la llamada de dos funciones del BIOS, que sólo están contenidas en el BIOS ampliado de las tarjetas EGA y VGA. Sólo con ello se puede realizar una diferenciación con respecto a los modelos anteriores, MDA hasta CGA, donde estas llamadas de función han de fracasar sin remedio.
Una de las dos funciones nombradas sólo está disponible en la tarjeta VGA, y así ayuda en la distinción entre tarjetas EGA y VGA. Se trata de la función 1Ah, que dispone de dos subfunciones. Para nuestros propósitos es relevante la subfunción 00h, que devuelve informaciones sobre la tarjeta de vídeo activa y pasiva.

En relación a todo esto se habla de una tarjeta de vídeo activa y pasiva a causa de que en muchos PC aparte de la tarjeta VGA se puede emplear otra tarjeta adicional, en la que se ha de tratar de una tarjeta MDA o Hercules. En la mayoría de los modelos PS/2 esta posibilidad sin embargo no existe, ya que no existen tarjetas MCA y Hercules para el bus MCA de estas máquinas.
Si se llama a la función 1Ah con el número de función y subfunción en los registros AH y AL, se podrá determinar inmediatamente después de la llamada de la función en el contenido del registro AL, si la función fue soportada por el BIOS, es decir, si hay presenta una tarjeta VGA y un BIOS VGA. El BIOS VGA hace servir la circunstancia, de que el BIOS normal vuelve inmediatamente al invocador en el caso de llamar a una función inexistente, dejando el contenido de todos los registros sin modificar.

En el registro AL el BIOS normal devuelve sin modificaciones el valor 00h, mientras que el BIOS VGA carga intencionadamente el número de función 1Ah en este registro, para documentar la ejecución con éxito de la llamada de función. El código en el registro BL para la tarjeta activa y en el registro BH para la tarjeta pasiva se puede interpretar según la siguiente tabla: Véase la tabla 68.

Si la llamada de la función 1Ah falla, se evaporan todos los sueños de las maravillosas posibilidades de una tarjeta VGA, pero por ello no se ha de renunciar. A lo mejor hay instalada una tarjeta EGA. Su presencia se comprueba en el siguiente paso. A ello ayuda la subfunción 10h de la función 12h, que sólo se soporta por las tarjetas EGA. Al contrario a todas las otras muchas funciones del BIOS, en su llamada no se ha de pasar el número de subfunción en el registro AL y no en el registro BL. Si el número de subfunción 10h después de la llamada de la función todavía se encuentra allí, se puede estar seguro de que no hay presente una tarjeta EGA.

Si no es así, el contenido del registro BH nos informa sobre si la tarjeta EGA está conectada a un monitor MDA o EGA en color. Para esto último está el valor 0, mientras que el monitor MDA se simboliza con el valor 1. Además, a través del registro BL se puede averiguar algo sobre el equipamiento RAM de la tarjeta, tal y como muestra la siguiente tabla:
Véase la tabla 69.
Aquí ahora los dos programas de ejemplo ISEVP.PAS y ISEVC.C, que contienen la función IsEgaVga, y que simultáneamente demuestran su empleo en el marco de un programa. En una forma ligeramente modificada podrá encontrar esta función en los diferentes programas de ejemplo que se presentarán a lo largo del apartado.

Selección y programación de juegos de caracteres

Las tarjetas EGA y VGA, ya se mencionó al principio del capítulo, al contrario que sus antecesoras no están unidas a diferentes juegos de caracteres, que haya grabados en un circuito ROM en la placa correspondiente. De lo contrario disponen de un potente generador de caracteres, que coge las informaciones sobre el aspecto de los diferentes caracteres de una zona determinada de la RAM de vídeo, en la que los patrones de bits están guardados según un esquema determinado e invariable. Este apartado le quiere mostrar como se construyen y se trabaja con juegos de caracteres y estas tablas, y enseñarle además que es lo que se puede hacer con ayuda de todo esto. Hablaremos de los siguientes puntos:

- Cargar y definir juegos de caracteres con el BIOS
- Conmutar de la representación de 8 puntos a la de 9 puntos
- Creación de logos de caracteres compuestos
- Estructura y posición de las diferentes tablas de caracteres
- Conmutación entre los diferentes juegos de caracteres o: 512 caracteres diferentes simultáneamente en pantalla
- Una ventana gráfica en modo texto
- Juegos de caracteres para el modo gráfico

Cargar y definir juegos de caracteres con el BIOS

El BIOS ampliada de las tarjetas VGA o EGA dispone de algunas funciones, con cuya ayuda se pueden modificar las diferentes tablas de caracteres. En total hay algo más de una docena de subfunciones disponibles, que se pueden llamar a través de la función 11h de la interrupción de vídeo del BIOS. Como ya es costumbre, el número de función se ha de pasar en AH, y el número de subfunción en el registro AL.
Aparte de las tablas de caracteres existentes, con ayuda de esta función se puede cargar tablas de caracteres propias, llevarla a la pantalla, o conmutar entre las diferentes tablas de caracteres. En las tablas de caracteres existentes se trata de juegos de caracteres que se encuentran grabados en un circuito ROM en las tarjetas EGA o VGA, y que desde allí se pueden copiar a una parte de la RAM de vídeo, de la que el generador de caracteres obtiene las informaciones sobre el aspecto de los diferentes caracteres.

En este aspecto, las tarjetas EGA y VGA se diferencian, ya que mientras en una tarjeta EGA sólo se dispone de dos juegos de caracteres diferentes, la tarjeta VGA dispone además de un tercer juego de caracteres, que también se puede activar con ayuda del BIOS. La diferenciación entre los diferentes juegos de caracteres no se refiere al tipo de letra, como en un impresora. De lo contrario, estos juegos de caracteres se diferencian en el tamaño de los diferentes caracteres, es decir en la matriz en la que se basan los caracteres.

Mientras que la tarjeta EGA trabaja normalmente en la matriz arriba representada de 8 × 14, contiene un segundo juego de caracteres en la ROM, cuyos caracteres se han de ceñir un poco el corsé, hasta 8 × 8 puntos. Ya que, independientemente del juego de caracteres seleccionado, el tamaño de l punto de pantalla se mantiene constante, estos caracteres naturalmente necesitan mucho menos espacio. Pero a causa de su resolución menor no se pueden distinguir con tanta facilidad. ¿Así que por qué emplear este juego de caracteres?

La respuesta a esta pregunta se encuentra en la otra cara de la medalla, ya que a causa del menor tamaño de los caracteres individuales, caben más en la pantalla, ya que la resolución total de la pantalla por supuesto se mantiene invariable. Al utilizar en juego de caracteres pequeño de 8 × 8 se pueden representar 43 línea de texto, en vez de las 25 líneas convencionales, de forma que se le pueden mostrar muchas más informaciones al usuario, simultáneamente.
La resolución de 25 o 43 líneas no se seleccionó aleatoriamente, sino que resulta de la resolución vertical de la tarjeta EGA, que en el modo de texto siempre es de 350 puntos. Como muestra la siguiente relación, al utilizar el juego de caracteres de 8 × 8 en realidad deberían aparecer 43,75 líneas en pantalla, pero como esto naturalmente no tiene sentido, se queda limitado a 43 líneas.

Resolución EGA vertical:	350 puntos	350 puntos
Altura de la matriz de caracteres:	14 puntos	8 puntos
Número de líneas de texto:	25 líneas	43,75 líneas

Lo que es cierto para la resolución vertical, también lo es para la resolución horizontal. Aquí, la tarjeta EGA dispone de 640 puntos. Si se divide esta cifra por la anchura de 8 puntos de ambos juegos de caracteres, resulta el ya conocido número de 80 columnas por línea de texto.
El caso de la tarjeta VGA es ligeramente diferente, tanto en el aspecto vertical, como en el horizontal. Aquí, la resolución vertical en el modo de texto no es de 350, sino de 400 puntos, por lo que en el modo de texto normal de 80 × 25 caracteres no se emplea el juego de caracteres de la EGA, con sus 8 × 14 puntos, sino un juego de caracteres especial de la VGA, con 8 × 16 caracteres. A pesar de todo, los juegos de caracteres de la EGA siguen estando disponibles en la VGA, en la cual, a causa de su mayor resolución, resultan en 28 o 50 líneas de texto.

La resolución de la tarjeta VGA también es mayor en el aspecto vertical, ya que no se representan 640 puntos por línea, sino 720. A pesar de ello, sólo caben 80 caracteres en una línea, ya que los diferentes caracteres miden horizontalmente 9, y no 8 puntos. Del punto noveno especial hablaremos en breve.
Bien, ahora dediquémonos a las funciones con cuya ayuda se pueden cargar los diferentes juegos de caracteres, con lo que se ajusta automáticamente el número de líneas de pantalla. Llevan los números 11h, 12h, 14h. Aparte de los dos números de función en el registro AX, esperan otro argumento de función en el registro BL. Refleja el número de la tabla de caracteres, en la que se ha de cargar el juego de caracteres, para activarlo después. Si no quiere trabajar con diferentes tablas de caracteres simultáneamente, debería indicar aquí el valor 0 para la primera tabla de caracteres. Sino, en las tarjetas EGA puede indicar valores de 0 a 3, y en tarjetas VGA valores de 0 a 7. En los siguientes apartados aprenderá más sobre las diferentes tablas de caracteres.

Véase la tabla 70.
Mientras que con estas funciones se puede cargar y activar un juego de caracteres determinado y programar simultáneamente los registros de la tarjeta de vídeo de forma que se muestre este juego de caracteres y una cantidad correspondiente de líneas de pantalla, los mencionados juegos de caracteres simplemente sólo se pueden cargar en una tabla de caracteres. Si por casualidad no se trata de la tabla de caracteres activa actualmente, esto no tiene ninguna influencia sobre la representación en pantalla, ya que ni se activa el juego de caracteres ni el número adecuado de líneas de pantalla. Su llamada, con respecto a la ocupación de registros, es idéntica con las tres funciones ya descritas.

Véase la tabla 71.
Pero no sólo las tablas de caracteres grabadas en la ROM se pueden cargar en la RAM de vídeo con ayuda del BIOS, sino que también existen las funciones adecuadas para los juegos de caracteres definidos por el usuario. Con ello por ejemplo se pueden redefinir caracteres individuales, o elegir otro tipo de fuente, como también es posible en la impresora.
Como truco especial se puede elegir libremente la altura de caracteres entre 1 y 32 puntos, con lo que naturalmente se adaptan las líneas que aparecen en pantalla adecuadamente. Sin embargo, a su fantasía se le ponen límites ópticos, ya que los caracteres que emplean menos de seis líneas de puntos apenas se pueden leer. Por otra parte, los caracteres que pasan de 16 líneas de puntos son especialmente nítidos y definidos, pero a cambio se reduce la cantidad de líneas en pantalla. Por ello se recomienda como norma el seleccionar la altura de los diferentes caracteres en el margen que utilizal BIOS, entre 8 y 16 líneas de puntos. Esto tiene otra ventaja, y es que el usuario no se ha de acostumbrar a una resolución de pantalla completamente diferente.

El BIOS le deja mano libre en este aspecto, ya que durante la llamada de la función dedicada a la carga de juegos de caracteres propios, la función 10h, se ha de pasar la altura de los caracteres en el registro BH. La anchura sin embargo, no se puede influenciar, de modo que en la EGA permanece constante en 8 puntos y en la VGA en 9 puntos.
Pero a cambio de ello puede seleccionar la tabla de caracteres libremente, en la que se ha de cargar su juego de caracteres, indicando el número de esta tabla antes de la llamada de la función en el registro BL. Además de ello, la función espera en el registro CX la cantidad de caracteres que se han de cargar, ya que en determinadas circunstancias no es necesario cargar la tabla completa, sino sólo algunos caracteres determinados en una tabla que ya exista. Para soportar esto, también ha de indicar el número (el código ASCII) del primer carácter. Se espera en el registro DX, y se puede mover en el rango de 0 a 255.

Las informaciones sobre los patrones de puntos de los diferentes caracteres, el BIOS las obtiene de un buffer, que se crea del invocador de la función y que ha de ser inicializado. Su dirección, la función la espera como puntero FAR en la pareja de registros ES:BP.
Como muestra la figura anterior, el buffer pasado ha de contener una entrada para cada uno de los caracteres a definir, cuyo tamaño en bytes a he corresponder a la altura de caracteres. Si por ejemplo se definen caracteres en una matriz de 8 × 12, cada entrada se compone por ello de 12 bytes. La longitud total del buffer se calcula del producto de 12 y de la cantidad de caracteres a definir, ya que las entradas individuales del buffer son secuenciales. La primera entrada en la dirección de offset 0000h acoge las informaciones del primer carácter a definir, y a continuación siguen en orden ascendente las entradas para todos los demás caracteres.

Dentro que cada una de las diferentes entradas, cada byte representa el patrón de bits para una línea de puntos del carácter, concretamente en orden ascendente, de la primera línea de puntos hasta la última. En un cada uno de estos bytes, los diferentes bits reflejan el estado de los puntos individuales de una de estas líneas de puntos, de izquierda a derecha. Si el bit está a uno, el punto de

pantalla correspondiente se representa en el color de texto fijado para el carácter. Si el bit por lo contrario está a cero, el punto aparece en el color de fondo correspondiente.

La subfunción 00h trabaja del mismo modo que la subfunción 10h, pero que sólo carga las definiciones de caracteres indicadas en la tabla de caracteres determinada, pero no la activa, ni adapta la cantidad de líneas de texto. También aquí encontramos de nuevo un dualismo, que ya era característico para la relación entre las funciones 11h, 12h y 14h por una parte, y las subfunciones 01h, 02h y 04 por la otra.
Por favor, nunca olvide durante la definición de caracteres y juegos de caracteres propios, que sus definiciones sólo tienen validez en pantalla, pero que no afectan a la impresora, ya que este no tiene ninguna posibilidad de enterarse del aspecto modificado de los caracteres en pantalla. Así que no se asombre si al hacer un hardcopy de la pantalla, la impresora imprima unos caracteres totalmente diferentes, que los que se pueden ver en el monitor.

Conmutación de la representación de 9 puntos a la de 8 puntos
Como ejemplo para la programación de juegos de caracteres individuales les queremos presentar un pequeño programa en el siguiente apartado, con cuya ayuda puede visualizar en pantalla logos (de empresa), que se pueden componer de diferentes caracteres. Pero en las tarjetas VGA esto trae consigo un problema, que tiene que ver con la extensión horizontal de los diferentes caracteres. Mientras que un carácter de una tarjeta EGA sólo tiene 8 puntos, en una tarjeta VGA tiene 9 puntos. Esta diferencia resulta de la diferente resolución horizontal de la pantalla, que en una VGA no es de 640, sino de 720 puntos.

¿Pero de dónde tomar el noveno punto, si tanto en los juegos de caracteres fijos de la ROM, así como en los juegos de caracteres que se pueden cargar individualmente con ayuda del BIOS, sólo se captan 8 puntos? La respuesta no es difícil, el noveno punto habitualmente se queda vacío. única excepción: los caracteres con el código ASCII entre C0h y DFh. En esta zona se encuentran los diferentes caracteres de marco, que han de contactar físicamente, para que se cree la impresión de una línea horizontal ininterrumpida. En estos caracteres el octavo punto se copia simplemente al noveno punto, de modo que se consigue el paso al siguiente caracteres.

Mediante el empleo del noveno punto, cada carácter de la tarjeta VGA dispone potencialmente de una resolución mayor, aunque en las tarjetas EGA y VGA sólo se codifiquen 8 puntos en la ROM. Pero como que ha de quedar algo de espacio entre los diferentes caracteres, en las tarjetas EGA sólo se pueden utilizar los primeros siete puntos, para que el octavo punto, libre, cree el espacio entre caracteres. En tarjetas VGA se puede utilizar este octavo punto, ya que el espacio al vecino se crea artificialmente con el noveno punto.

Este procedimiento de la tarjeta VGA normalmente no tiene efectos molestos, siempre que sólo se definan caracteres individuales, que no tengan conexión con los caracteres adyacentes. Pero si se quiere como en logotipos crear una especie de mosaico de caracteres individuales no se puede conseguir un paso continuo entre las diferentes partes del mosaico, ya que no se tiene control sobre el noveno punto. Por ello se ha de suprimir el noveno punto en este caso, y se ha de conmutar a la representación de 8 puntos, como se emplea en la tarjeta EGA.

En esta conmutación se involucran diferentes registros de la tarjeta VGA, que por desgracia no se pueden direccionar directamente a través del BIOS, y que por ello se han de programar directamente. Primero vamos a nombrar el Miscellaneous Output Register, que se puede leer a través de la dirección de puerto 3CCh, y que se puede escribir a través de la dirección de puerto 3C2h. Contiene dos bits (los bits 2 y 3), con cuya ayuda se puede ajustar el temporizador de la tarjeta VGA, y con ello la resolución horizontal.

Mientras que la tarjeta VGA en el modo de texto funciona normalmente con 28,322 MHz, y lleva 720 puntos por línea de puntos a la pantalla, con una frecuencia de 25,175 MHz sólo son 640 puntos. Esta es la resolución deseada, ya que con una anchura de caracteres de 8 puntos se obtienen exactamente 80 caracteres por línea de pantalla. Leyendo el registro nombrado, modificando los bits 2 y 3 en correspondencia, y devolviendo el valor al registro, la tarjeta se puede conmutar a la resolución horizontal deseada. Pero esto sólo es la mitad, ya que la VGA sigue trabajando con 9 puntos por carácter, y desencadenará rápidamente una verdadera tormenta de nieve en la pantalla, sino se la conmuta rápidamente a 8 puntos.

Esto ocurre en el siguiente paso, modificando el contenido del Clocking Mode Register. Este registro es parte del llamado Sequencer Controller, y no se puede direccionar directamente como el Miscellaneous Output Register. Antes y después del acceso a los registros del Sequencer Controller se ha de realizar un reset de este controlador mediante su registro de reset. Solo después se puede modificar un registro del Sequencer, como el Clocking Mode Register, cuyo bit 0 es responsable de la anchura de caracteres.

Pero no es suficiente con ello. Como último paso se ha de modificar el Horizontal Pel Panning Register, que ya se presentó en relación al llamado Smooth Scrolling. Más exactamente, después de la conmutación a la representación de 8 puntos se ha de cargar el valor 0 en este registro, para que la imagen de vídeo se desplace un punto a la izquierda con respecto a la representación normal. Si esto no ocurre, el usuario verá en la primera columna de puntos de la pantalla un constante titilar, que es muy molesto. Pero aquí no es necesario un acceso directo al registro mencionado, ya que existe una función del BIOS adecuada.

Este proceso, que en tres pasos provoca la conmutación de una anchura de caracteres de 9 a 8 puntos, naturalmente es reversible, de modo que de manera similar se puede retornar a la representación de 9 puntos. Para ello se han restaurar la resolución horizontal de 720 puntos en el Miscellaneous Output Register, y la representación de nueve puntos en el Clocking Mode Register. Pero también el Horizontal Pel Panning se ha de poner a 8 de nuevo, para que la primera línea de puntos de la pantalla se vuelva a hacer visible.

Si se interesa por la estructura concreta de los registros mencionados, por favor consulte al final de este capítulo. Allí puede encontrar una descripción detallada de todos los registros de la EGA y VGA. Puede encontrar una rutina que realice la conmutación según las reglas establecidas, la puede encontrar en los programas de ejemplo del siguiente apartado.

Creación de logotipos de caracteres compuestos

Pero vayamos ahora a los logotiposs ya mencionados, por los cuales se ha de reducir la anchura de caracteres de la VGA a 8 puntos. Estos logos normalmente quedan mucho mejor en la pantalla que un aburrido mensaje de Copyright, que se encuentra en casi cualquier programa. Por desgracia, un logotipo sólo se puede componer en los menos de los casos con los caracteres establecidos del juego de caracteres ASCII. Esto naturalmente también es cierto para las tarjetas EGA y VGA, pero estas tarjetas ofrecen la posibilidad de determinar el aspecto de los diferentes caracteres individualmente.

En los casos menos numerosos será suficiente con un solo carácter para la construcción de un logotipo, ya que en una tarjeta VGA contiene hay 128 puntos (8 × 16), y en una tarjeta EGA sólo 112, de los que dispone el diseñador del logo. Además de eso, un solo carácter hace poca apariencia en la pantalla. ¿Que menos, que construir un logo de varios caracteres, ordenados en rectángulo, como un mosaico.

No sólo que con ello se aumenta la resolución del logo, sino que además se hace más visible en la pantalla, porque simplemente es más grande. Sin embargo tampoco se debería exagerar con el tamaño del logo, ya que de lo contrario se gastarían demasiados caracteres del juego de caracteres ASCII. Al fin y al cabo, además del logo deben seguir apareciendo en la pantalla todas las letras y

los números, para que pueda haber una interacción con el usuario. Así que no se trata de redefinir un juego de caracteres completamente nuevo. Mas bien se redefinen algunos caracteres de un juego de caracteres existente, para obtener el efecto deseado.

Para la definición del logotipo recomendamos aquellos caracteres del juego de caracteres ASCII, que no necesita en su programa. Esto son por ejemplo los caracteres con el código ASCII menor que 32, los caracteres extranjeros, y los caracteres con los códigos ASCII mayores que 224. Según qué caracteres se necesiten realmente en su carácter, se pueden liberar hasta 100 caracteres para la utilización en el logo. De ello resulta un logotipo con una extensión de respetable 10 × 10 caracteres, con una resolución total de 12.000 puntos.

Para ayudarles en el desarrollo de su logotipo personal, queremos presentarles el programa LOGO, que existe en una versión en Pascal y otra en C. Los dos programas se llaman LOGOP.PAS y LOGOC.C. En el centro de los dos programas se encuentra el procedimiento o función con el nombre BuildLogo, que controla completamente la construcción del logo y su visualización.
Como informaciones simplemente se necesitan para ello la posición deseada del logo en la pantalla, su altura en líneas de puntos, su color y una matriz de cadenas, con cuya ayuda se determina el aspecto del logotipo.

En esta matriz, cada cadena corresponde a una línea de puntos del logo, donde cada carácter en una de estas cadenas responde a un punto en la columna de puntos correspondiente. Si el carácter correspondiente es un espacio, el punto correspondiente permanece vacío en el logo, de lo contrario se activa. Con esta forma de codificación se gasta un montón de espacio en la memoria para la plantilla del logo, pero a cambio, es fácilmente editable en el programa fuente, sin que se haya de desarrollar un editor específico. Además, la anchura del logo se puede obtener directamente de la array de cadenas, y no se le ha de pasar al procedimiento BuildLogo como parámetro adicional.

Según el tamaño del logo pasado, y de la tarjeta de vídeo instalada (EGA o VGA), BuildLogo calcula automáticamente la cantidad de caracteres necesitados, y define los patrones de puntos de los diferentes caracteres del logo con ayuda de la ya descrita subfunción 00h de la función 10h. Si el logo no rellena completamente el rectángulo que le sirve como base, se centra en él.
Para la definición del logo se utilizan los caracteres extranjeros, que en el juego de caracteres ASCII se encuentran entre los códigos 128 y 166 y se hacen servir algunos que se emplean, para simplificar las cosas un poco. Pero BuildLogo se puede modificar con facilidad, de modo que se pueden emplear más caracteres del juego de caracteres ASCII para la construcción del logo.

También es BuildLogo el que conmuta la anchura de caracteres en una tarjeta VGA a 8 puntos, cuando detecta la presencia de esta tarjeta con ayuda de la función IsEgaVga. La conmutación se realiza con el procedimiento o función SetCharWidth, que realiza los diferentes pasos para la conmutación de la anchura de caracteres, tal y como se describió en el apartado anterior.
En cuanto a la altura de los diferentes caracteres, BuildLogo toma la suposición de que un carácter en la tarjeta EGA se compone de 14 y en una tarjeta VGA de 16 líneas de pantalla. Así que se parte de los juegos de caracteres normales, que resultan de una resolución de pantalla de 25 líneas de texto por 80 columnas de texto. Si quiere utilizar BuildLogo en un programa que active un juego de caracteres más pequeño, y que por ello aumente el número de líneas de pantalla ha de adaptar la variable CharHeigh en ambos programas LOGO. En ese caso naturalmente se han de redefinir más caracteres en un logo del mismo tamaño, ya que hay menos líneas de puntos por carácter.

Si ya no necesita el logo, puede volver a borrarlo con ayuda del procedimiento ResetLogo, para que se vuelva a instalar el juego de caracteres antiguo, y en las tarjetas VGA se vuelva a conmutar a una representación de 9 puntos.

Como ejemplo para el trabajo con BuildLogo se define en los dos programa LOGO un pequeño logotipo, y se representa en el borde inferior de la pantalla. En la parte superior de la pantalla se listan los caracteres del juego de caracteres ASCII, donde resaltan en color los caracteres que se pueden redefinir por BuildLogo. De esta forma se pueden reconocer muy bien los caracteres redefinidos, para ver una parte del mosaico, que finalmente formará el logotipo.

Mientras que el programa Pascal LOGOP.PAS puede pasar sin rutinas en ensamblador, el programa LOGOC.C es apoyado por el módulo LOGOCA.ASM. Se necesita, por que la función del BIOS que se llama para la definición de los caracteres espera informaciones en el registro Bit-Plane, y este registro no se puede direccionar mediante la función de interrupción normal int86(). Por ello, la rutina DefChar en el módulo de ensamblador se encarga de la definición de los caracteres.

Estructura y posición de las diferentes tablas de caracteres

Mientras sólo acceda a las diferentes tablas de caracteres con el BIOS, no ha de preocuparse de la posición de estas tablas. Pero esto se modifica inmediatamente, si ha de influenciar directamente las diferentes tablas de caracteres; situación que se describe en uno de los siguientes apartados. En ese caso, el conocimiento de la posición y la estructura de la tabla de caracteres le ofrece la posibilidad del acceso directo a todas o solo una de las definiciones de caracteres. Entonces es posible hacer muchas cosas que con el BIOS son muy difíciles o imposibles. Así que, ¿dónde están estas tablas y cómo están estructuradas?

Para aclarar esta pregunta, primero hemos de hacer una breve excursión a la organización de la RAM de vídeo de las tarjetas EGA y VGA. Como mostrará en detalle el apartado 9.8.5, las tarjetas EGA y VGA dividen su RAM de vídeo en cuatro zonas iguales, que se denominan bit-planes. Realizan diferentes tareas en el modo gráfico y de texto, pero los cuatro bit-planes de una tarjeta de vídeo equipada con 256 KBytes siempre ocupan 64 KBytes.

Sin que el programador lo note, los dos primeros bit-planes se utilizan en el modo de texto para acoger los códigos de carácter y el byte de atributos. Los códigos ASCII de los caracteres se guardan en el Bit-Plane #0, ya que todos los accesos de memoria a la RAM de vídeo a partir de B800h, que se refieren a la dirección de offset par, se redireccionan automáticamente a este Bit-Plane. Por otra parte, todos los accesos de memoria bajo direccionen impares, en las que se guarda el byte de atributos, van al Bit-Plane #1.

También utilizan los otros bit-planes en el modo de texto, ya que por norma, las tablas de caracteres de las tarjetas EGA y VGA se guardan en el tercer Bit-Plane, que sirve únicamente para esto. Como este Bit-Plane en una tarjeta de 256 KBytes ocupa 64 KBytes, pero cada tabla de caracteres sólo ocupa 8 KBytes, en el Bit-Plane #2 se pueden guardar un total de 8 tablas de caracteres diferentes. Como muestra la siguiente figura, sólo las tarjetas VGA lo emplean. Las tarjetas EGA se conforman con sólo cuatro tablas de caracteres diferentes, de modo que se desaprovechan 32 KBytes en este Bit-Plane.

Cada tabla de caracteres necesita 8 KBytes por que necesita 32 Bytes para cada carácter de los 256 diferentes de que se compone. 32 seguro que se acuerda es el tamaño máximo de altura de los caracteres en las tarjetas EGA y VGA es de 32, de modo que en las tablas de caracteres siempre se prevé el peor de los casos. Si la tabla de caracteres ha de contener un juego de caracteres con una altura inferior a 32 líneas de puntos, quedan libres una cantidad correspondiente de bytes al final de cada entrada.

Los demás bytes se codifican de la forma que ya se explicó en relación a las diferentes funciones del BIOS para la definición de caracteres. Así que cada byte acoge el patrón de puntos de una línea del carácter, donde los diferentes bits representan el estado de los puntos en este línea de puntos. Con esta ordenación se puede calcular sencillamente la dirección de inicio de un carácter, o de una

línea de puntos en él, si se conoce la dirección inicial de la tabla de caracteres correspondiente. Multiplique el código ASCII del carácter por 32, súmele el número de la línea de puntos deseada, y además la dirección de inicio de la tabla de caracteres. Ya tiene la dirección de offset de la línea de puntos.

Pero tan sencillo tampoco es el acceder a una tabla de caracteres, ya de momento no es posible acceder al Bit-Plane #2. Para acceder a un Bit-Plane en las tarjetas EGA y VGA, primero se ha de «visualizar» éste en la memoria, para que pueda ser direccionado a través de la dirección de segmento A000h. Por desgracia, no existen funciones del BIOS para este propósito, de modo que se necesita recurrir a la programación directa de los registros de las tarjetas EGA y VGA.

En concreto son los diferentes registros del Sequencer Controller y del Graphics Controller, cuya programación libera el acceso al Bit-Plane #2. En lo que se refiere al Sequencer Controller, son los registros 2 y 4 los que son responsables del acceso a los diferentes bit-planes. Llevan el nombre de Map Mask Register y Memory Mode Register. El primero determina a qué Bit-Plane se puede acceder actualmente. Hay un bit por cada Bit-Plane, de modo que en nuestro caso se ha de activar el bit del Bit-Plane #2, borrando todos los demás. Al fin y al cabo no queremos afectar a los demás bit-planes con nuestros accesos.

Pero para que no se acceda simultáneamente a todos los bit-planes, sino sólo a los bit-planes direccionados en el Map Mask Register, se ha de cargar el Memory Mode Register con el valor 7. Normalmente contiene 3 en el modo de texto, que en unión con el Map Mask Register provoca que durante un acceso a la RAM de vídeo en B800h todos los accesos a direcciones de memoria pares (códigos ASCII) vayan a parar el Bit-Plane #0, y todos los accesos a posiciones de memoria impares (Códigos de atributo) al Bit-Plane #1. Ahora se suprime esta división de las direcciones de memoria.

Al igual que durante la conmutación de la representación de 9 a 8 puntos, también en este acceso a los registros del Sequencer se ha de volver a realizar una reset a través del registro de Reset del Sequencer. Para ello se escribe el valor 1 antes del acceso al Map Mask Register y el Memory Mode Register, y después el valor 3.
No se ha de realizar un reset para acceder a los registros del Graphics Controller. Aquí son los registros 4, 5 y 6, que se han de manipular, para poder direccionar el Bit-Plane #2. Se trata del Read Map Select Register, el Graphics Mode Register y el Miscellaneous Register.

En el Read Map Select Register se ha de escribir el número del Bit-Plane a direccionar; aquí es el valor 2. Con ello se consigue que incluso en accesos de lectura a la RAM de vídeo se direcciones el Bit-Plane #2. En el Graphics Mode Register se ha de escribir el valor 0, tratándose simplemente de borrar el bit 4, ya que todos los demás bits sólo tienen importancia en el modo gráfico. Mediante el borrado de este bit se confirma de nuevo que no se ha de realizar la división en posiciones de memoria pares e impares de los diferentes bit-planes.

Esto ocurre de nuevo mediante el Miscellaneous Register, en el que, aparte del bit en cuestión, el 1, también los bits 2 y 3 tienen su importancia. Ellos determinan donde está la RAM de vídeo, donde se han de «visualizar» los bit-planes en la zona de direcciones del procesador. Mientras que la RAM de vídeo en el modo de texto aparece en la posición B800h, extendiéndose a lo largo de 32 KBytes, al Bit-Plane #2 se le hace accesible la dirección de segmento A000h, en la que se pueden acceder a los 64 KBytes completos del Bit-Plane.

Con los ajustes descritos se destruyen los valores para el modo de texto, así que después de terminar las diferentes operaciones de registro es posible el acceso al Bit-Plane #2, pero ya no el acceso a la RAM de vídeo normal en B800h. Salidas de caracteres durante un acceso al Bit-Plane #2

desaparecen sin efecto alguno en el Nirvana de los bits, y no se reflejan en la RAM de vídeo. La imagen de vídeo sigue apareciendo sin cambios en el monitor, ya que la tarjeta de vídeo puede seguir accediendo internamente a la RAM de vídeo.

Para que su programa, o el BIOS pueda volver a acceder a la RAM de vídeo, las modificaciones realizadas en los diferentes registros se han de deshacer en pro de los ajustes anteriores. La siguiente tabla muestra los valores que han de contener los diferentes registros para acceder al Bit-Plane #2, o para acceder a la RAM de vídeo en B800h
Véase la tabla 72.
Puede encontrar dos rutinas con cuya ayuda se puede entre el acceso a la RAM de vídeo y el Bit-Plane #2, en los programas de ejemplo MIKADO, en los siguientes apartados. También puede aprender más sobre los diferentes registros que hay envueltos en la conmutación al final de este capítulo, en la descripción de todos los registros EGA y VGA.

Conmutación entre diferentes juegos de caracteres o: cómo tener 512 caracteres diferentes en pantalla
Después de haber visto que en el Bit-Plane #2 de las tarjetas EGA se podían colocar 4 juegos de caracteres diferentes, y en la VGA incluso 8, y que el BIOS no está fijado para el manejo del primer juego de caracteres, falta por resolver la pregunta de como se conmuta entre los diferentes juegos de caracteres, y cómo se trae a pantalla un juego de caracteres determinado. La respuesta la tiene el Character Map Select Register, que es una parte del Sequencer Controller, y que se puede direccionar ahí con el número 4. Es exclusivo para la selección de la tabla actual de caracteres, como muestra la siguiente figura.

Lo que destaca en la ocupación del Character Map Select Register a primera vista es la circunstancia de que aquí no se habla de un juego de caracteres, sino de dos. Enseguida comentaremos qué es lo que ocurre con esto.
La ordenación de los diferentes bits, que indican los números de los dos juegos de caracteres, es también algo irritante. En vez de emplear tres bits adyacentes para la codificación de un número, los dos números se componen de un grupo de 2 bits y un bit adicional, que está separado de ambos, y que es el bit de más peso del grupo de tres. La razón de esto se encuentra en la historia de las tarjetas EGA y VGA. En las tarjetas EGA sólo se emplean dos bits para la codificación de un número de juego de caracteres, ya que aquí sólo se dispone de 4 juegos de caracteres diferentes, es decir se han de codificar los números entre 0 y 3.

En las tarjetas VGA es diferente, ya que permiten el acceso a ocho juegos de caracteres diferentes. Aquí se necesitan tres bits para la codificación de los números de juego de caracteres 0 a 7. Para garantizar una máxima compatibilidad de registros entre las tarjetas EGA y VGA se ha mantenido la ocupación de la EGA, y se han añadido dos bits a los bits ya existentes. Ya que las posiciones de bits correspondientes de la tarjeta EGA no se utilizaban hasta ahora, esto no molesta.

Pero ahora dediquémonos a la pregunta de por qué en este registro se acogen dos números de registro diferentes. En la figura se habla del segundo juego de caracteres, lo que no refleja un orden jerárquico entre los dos juegos de caracteres. Al contrario que sus antecesores, las tarjetas EGA y VGA están en disposición de mostrar simultáneamente 512 caracteres en la pantalla, en vez de los 256 de siempre. Aunque esta característica no es muy conocida, es altamente interesante, ya que, naturalmente aparece la pregunta de cómo se selecciona entre los dos juegos de caracteres. Al fin y al cabo, el código ASCII de un carácter sólo refleja un número entre 0 y 255 en la RAM de vídeo, de modo que el criterio de diferenciación no se ha de buscar aquí.

En el byte de atributos de un carácter no queda espacio para estas informaciones, se podría pensar a primera vista. Al fin y al cabo se emplean los dos cuartetos (nibbles) de este byte, para seleccionar

uno de los 16 colores de texto y fondo para el carácter correspondiente. Pero en realidad, precisamente aquí está la llave para el trabajo con 512 caracteres diferentes, ya que es el byte de más peso del color de texto; es decir el bit 3 en el byte de atributos mediante el cual se selecciona el primer o el segundo juego de caracteres.

Si este bit contiene el valor 0, y se selecciona un color de texto menor que 8, la tarjeta de vídeo toma el aspecto para el carácter asociado el primer juego de caracteres. Si el bit #3 está activado, se utiliza el segundo juego de caracteres, donde se emplea automáticamente el color de texto, que es mayor o igual que 8.
Esta separación entre el primer juego de caracteres y el segundo se realiza por norma, aunque no se pueda distinguir por el usuario. Pero tiene su razón, y es que en el Character Map Select Register se indica en mismo número para el primer juego de caracteres y el segundo. Pero en el momento en el que los dos números difieren, la diferencia se puede ver claramente en pantalla.

El acceso a este registro se recomienda siempre que aparezcan dos juegos de caracteres diferentes en la pantalla, o siempre que se quiera seleccionar un juego de caracteres diferente del empleado actualmente. De este modo se puede conmutar rápidamente entre dos juegos de caracteres, para obtener interesantes efectos ópticos. pero no es necesario programar directamente el Character Map Select Register, ya que el BIOS dispone de una subfunción de la función 11h en el BIOS de vídeo.

La mencionada función es la subfunción 03h, que durante su llamada sólo espera otro parámetro aparte de los dos números de función en los registros AH y AL, que se ha de pasar en el registro BL. Representa un valor, que se ha de cargar el en Character Map Select Register y que determina el juego de caracteres mostrado. En el siguiente apartado puede encontrar un ejemplo de aplicación de esta función, en el que con ayuda del segundo juego de caracteres se construye una ventana gráfica en la pantalla de texto.

El hecho de que los caracteres del segundo juego de caracteres aparezcan más claros que los del primero es molesto, ya que al fin y al cabo tienen activado el bit 3 del color de texto. Pero esto se puede solucionar fácilmente, igualando simplemente los contenidos de los registros de paleta con el número 0 a los que tienen un número entre 0 y 7. Ya que al BIOS no le faltan estas funciones, esto no es ningún problema. Sin embargo, el número de colores de texto se reduce con ello a 8, pero en realidad ningún programa emplea realmente ocho colores, y finalmente puede seguir seleccionando los colores disponible mediante la programación de los registros de paletas.

Una ventana gráfica en el modo de texto
Quien no necesite precisamente 512 caracteres diferentes en la pantalla, porque trabaja con caracteres especiales matemáticos, símbolos, o lo que sea, tiene una alternativa interesante en el Character Map Select Register: una ventana gráfica en el modo de texto. Con su ayuda se pueden incluir pequeños gráficos o dibujos en el modo de texto, sin conmutar expresamente al modo gráfico, cuya programación es más pesada y complicada.

La técnica que se esconde detrás de esta ventana gráfica se parece a la de la creación de un logo. También aquí se juntan los caracteres esta vez del segundo juego de caracteres y no del primero en un bloque, formando como mosaico una imagen completa. Mediante este procedimiento cada uno de los caracteres del segundo juego de caracteres está representado exactamente una vez en la ventana, y esto es muy importante. Por ello la ventana gráfica sólo puede tener un tamaño de 256 o menos caracteres. Un bloque así se puede formar, por ejemplo, por un cuadrado de 16 por 16 caracteres, aunque se permite cualquier ordenación.

Para la salida gráfica en la ventana se considera el segundo juego de caracteres como una gran matriz de puntos individuales, que se han de representar en las coordenadas de puntos en la ventana

gráfica. Si se quiere activar o desactivar un punto en la ventana gráfica, se calcula por ello el carácter en el cual se encuentra el punto. Después se averigua la posición del punto con respecto a la esquina superior izquierda del carácter y el bit correspondiente en el patrón de puntos del carácter en la tabla de caracteres se borra o activa. para ello, un ejemplo práctico.

En las tarjetas VGA los diferentes caracteres se componen de 16 líneas de puntos, que acogen a 8 puntos. Si se determina como origen de coordenadas para la ventana gráfica la esquina inferior izquierda, todos los puntos con una coordenada X menor que 8 y una coordenada Y menor que 16 caerán en la zona de influencia del carácter de texto que se encuentra en la esquina inferior de la ventana gráfica. Como muestra la figura, esto siempre es el carácter con el código ASCII 0.

Con ello, el número del carácter a manipular en el juego de caracteres queda determinado. En el siguiente paso se calcula la línea de puntos que se ha de modificar en el carácter. Se trata de respetar la estructura de los diferentes caracteres en la tabla de caracteres. Las diferentes líneas de puntos de los caracteres se guardan aquí de arriba a abajo. Un punto con la coordenada Y 15 se encuentra por ello en la línea de puntos superior del carácter con el código ASCII 0, mientras que los puntos con la coordenada 0 ocupan la línea de puntos inferior. Gracias a esta norma se puede calcular fácilmente las línea de puntos fácilmente.

Sólo queda averiguar la posición de bit del punto gráfico direccionado en la línea de puntos correspondiente. Pero esto no es difícil, ya que, partiendo del borde izquierdo del carácter los diferentes puntos ocupan las posiciones de bit 7, 6, 5 etc. hasta 0. Con ayuda de las tres informaciones de número de carácter, línea de puntos y número de bit se puede borrar o fijar el punto direccionado, proyectando esta información simplemente a la estructura de la tabla de caracteres en la RAM de vídeo.

Naturalmente es algo complicado redefinir el carácter completo para fijar o borrar un punto individual, como es necesario utilizando las funciones del BIOS. Además, estas funciones no son demasiado rápidas, lo que se nota especialmente cuando se han de visualizar diferentes puntos en secuencia (por ejemplo en una línea). Por esta razón los dos programas que se presentan en este capítulo acceden directamente al Bit-Plane #2, obteniendo una velocidad aceptable durante la manipulación de la ventana gráfica.

Sin embargo, en las tarjetas VGA se encuentra de nuevo un problema del que ya se habló en relación con el programa LOGO: el noveno punto. Para que en las tarjetas VGA en la ventana gráfica se proyecte automáticamente el octavo punto sobre el noveno, o permanezca vacío, se ha de reducir la representación de caracteres a ocho, como ya era el caso con LOGO. Los pasos necesarios para ello ya se han descrito en el apartado anterior.
Los programas que presentamos aquí llevan el nombre MIKADO y existen en una versión en Pascal (MIKADOP.PAS) y otra en C (MIKADOC.C). En el centro de estos programas está la rutina con el nombre InitGraphArea, con cuya ayuda se puede instalar una ventana gráfica. Además hay una rutina con cuya ayuda se pueden activar o desactivar diferentes puntos. Lleva el nombre SetPixel, y sirve como base para la función o procedimiento Line, que dibuja una línea en la ventana gráfica, según el conocido algoritmos de Bresenham.

Al procedimiento InitGraphArea el usuario le puede pasar tanto la posición de la ventana gráfica en pantalla, así como su extensión y color para los diferentes caracteres. Además se puede seleccionar el número del juego de caracteres que se ha de utilizar para la formación de la ventana gráfica. En base a las indicaciones sobre la extensión de la ventana, en unión con la resolución de los diferentes caracteres se calculan las coordenadas máximas de X e y, y se guardan en las variables globales xmax e ymax.

El acceso al Bit-Plane #2 se realiza según el principio ya anteriormente descrito, mediante una rutina con el nombre GetFontAccess. Como oposición existe la función o procedimiento ReleaseFontAccess que vuelve a hacer posible el acceso a la RAM de vídeo en B800h. Las dos rutinas intencionadamente no se llaman con demasiada frecuencia en el programa, ya que su ejecución siempre está ligada a la aparición de un poco de nieve en la pantalla.
Para el trabajo del programa MIKADO también es importante el procedimiento SelectMaps. Se encarga de la tarea de colocar los juegos de caracteres deseados en el Character Map Select Register. Para ello utiliza la función del BIOS anteriormente descrita. Con ello se evita un acceso directo al hardware de la tarjeta de vídeo, lo que le viene muy bien al programa en lo que se refiere a la diferente ocupación de los registros de las tarjetas VGA de diferentes fabricantes.

La programación de los registros de paleta sirven en los dos programas MIKADO a diferentes procedimientos, definen de nuevo un elemento de la paleta de 16 colores con ayuda del BIOS, o que bien definen de nuevo la paleta completa, o averiguan su contenido. Llevan los nombres SetPalCol, SetPalAry, GetPalCol y GetPalAry y se utilizan premeditadamente, para hacer disponible diferentes colores para la ventana de texto y gráficos.
A propósito de colores: los colores son un asunto complicado en las ventanas gráficas, ya que naturalmente no se puede asociar un color a un punto determinado, sino sólo a un carácter entero. Se afectan directamente 14 × 8 Puntos en una tarjeta EGA, y 16 × 8 puntos en una tarjeta VGA. Por ello es recomendable el equipar todos los caracteres que forman la ventana gráfica con un color homogéneo de texto y fondo.

En los dos programas MIKADO, esto no se ha hecho premeditadamente, sino que los diferentes caracteres se equiparon con un código de color continuo. Esto es lo que le da el nombre al programa, ya que dibujando líneas en la ventana gráfica normalmente seccionan diferentes caracteres, que muestran a causa de ello un color siempre cambiante. Esto hace aparecer estas líneas como Mikados, los pequeños palitos del juego japonés de habilidad del mismo nombre.
Los dos programas MIKADO también muestran el orden en que se ha de realizar la llamada de las diferentes rutinas: Primero debería asegurarse con ayuda de la función IsEgaVga que realmente hay instalada una tarjeta EGA o VGA. Después se puede abrir la ventana gráfica mediante la llamada de InitGraphArea. Antes de acceder a esta ventana se ha de realizar una llamada de GetFontAccess, después de la cual ya puede dibujar los diferentes puntos con ayuda de SetPixel o Line.

Si entretanto se quiere visualizar texto en la pantalla, primero se ha de llamar ReleaseFontAccess y a continuación de nuevo GetFontAccess. La ventana gráfica se puede borrar mediante una llamada del procedimiento ClearGraphArea. Libera automáticamente el acceso a la RAM de vídeo, y vuelve a ajustar un juego de caracteres homogéneo (el juego de caracteres estándar con el número 0).
Que, a pesar de la ventana gráfica, siguen estando disponibles todos los caracteres del juego de caracteres ASCII a través de primer juego de caracteres, lo demuestra el programa MIKADO, llenando la pantalla alrededor de la ventana gráfica con ese mismo carácter.

Todo lo que necesita para el trabajo con una ventana gráfica en el modo ya lo puede encontrar en los dos programas MIKADO. Después de un breve vistazo a los dos listados, seguramente no le resultará difícil el emplear las rutinas que vienen, para sus propios propósitos. Es fascinante cuando en medio de la pantalla de texto se abre de pronto una ventana con gráficos. Su fantasía no tendrá límites.

Juegos de caracteres para el modo gráfico
También en el modo gráfico de las tarjetas EGA y VGA se utilizan tablas de caracteres, en cuanto se quieren visualizar caracteres en la pantalla con ayuda de las funciones correspondientes del

BIOS. El Bit-Plane #2 ya no está disponible para acoger las tablas de caracteres, ya que ha de acoger, junto con los demás BPS las informaciones de puntos para la pantalla gráfica.

Por ello el BIOS ha de tomar los patrones de puntos de los diferentes caracteres en este modo o bien directamente del circuito ROM correspondiente o de un buffer RAM. Ambas cosas son posibles, pero al igual que en el modo de texto se plantea la pregunta de qué juego de caracteres se ha de utilizar.
Sin que le comunique al BIOS qué juego de caracteres ha de utilizar, en el modo gráfico automáticamente emplea el juego de caracteres que está activo en el modo de texto de 80 × 25 caracteres. En las tarjetas EGA es el juego de caracteres de 8 × 14, y en las VGA el de 8 × 16. Al igual que en los demás casos, la cantidad de caracteres representables en pantalla corresponde al cociente de la resolución vertical y la altura de los caracteres. Lo mismo es cierto para el número de caracteres de texto que caben en una línea de pantalla, donde el dividendo es ocho.

Para la salida de texto en el modo gráfico no ha de acceder al juego de caracteres determinado, sino que con ayuda del BIOS puede seleccionar cualquiera de los juegos de caracteres de la ROM. A ello le ayudan las subfunciones 22h, 23h, 24h de la función 11h del BIOS de vídeo. En la literatura de sistema dedicada se puede leer que durante la llamada de estas funciones se han de pasar dos informaciones adicionales al número de función en los registros AH y AL, en los registros BL y DL. Según nuestras experiencias incluso con el original del IBM esto no es cierto. Normalmente, estos parámetros no tienen ningún efecto sobre la siguiente salida de texto en el modo gráfico, y por ello se pueden omitir, Así que para la llamada de esta función sólo necesita el número de función.

Véase la tabla 73.
Como alternativa a ello puede cargar mediante la subfunción 21h una tabla de caracteres propia para el modo gráfico, pero que en ese caso debería ser de 256 caracteres. Para ello se ha de pasar el número de función en los registros AH y Al, la altura de los diferentes caracteres en el registro CX. Al igual que en las demás funciones del BIOS, con ayuda de las cuales se pueden definir caracteres, esta función espera además, durante su llamada, en el registro ES:BP un puntero a la tabla de caracteres. Su estructura ha de corresponder con las tablas de caracteres tal y como se esperan de las subfunciones 00h y 10h.

Desplazamiento lento de pantalla (smooth-scrolling)
Si se le quiere dar al usuario la sugestión de una imagen en movimiento, en los antecesores de la EGA y VGA, se tenían pocas posibilidades. En el modo texto sólo se podía desplazar la pantalla un mínimo de un carácter, y en el modo gráfico se había de copiar todas la RAM de vídeo, si se quería desplazar la pantalla tan sólo un punto. En cuanto a animación, el PC se quedó muy atrás con respecto a la competencia.
Con las tarjetas EGA y VGA esto no es así. Ya disponen de una posibilidad de hardware, para desplazar la pantalla punto a punto, y realizar con ello un scroll suave, llamado Smooth-Scrolling. Sólo con ello se hizo posible la programación de juegos de acción emocionantes, que casi siempre se basan en un fondo en movimiento. Pero también en el modo de texto se pueden conseguir efectos interesantes, como mostrará este apartado. Hablaremos de los siguientes temas:

Smooth-scrolling mediante los registros de Pel-Panning
Desplazamiento del inicio de la pantalla y modificación de la longitud de línea en la RAM de vídeo
Sincronización del desplazamiento con la controladora CRT
Texto en movimiento en el modo de texto

Desplazamiento lento de pantalla mediante los registros de Pel-Panning

La posibilidad del desplazamiento por puntos de la zona de pantalla visible se oculta detrás de dos registros en las tarjetas EGA y VGA, que pertenecen a dos unidades funcionales completamente diferentes del hardware de vídeo. En principio, ambos tienen la misma tarea, en concreto, determinar el punto inicial de la pantalla, en cuanto al aspecto horizontal y vertical.

El porqué el Vertical-Pel-Panning-Register es una parte de la controladora CRT, mientras que el Horizontal-Pel-Panning-Register se colocó en el controlador de atributos (Attribute-Controller), sólo lo sabrán probablemente los desarrolladores de la primera tarjeta EGA.
Comencemos nuestras consideraciones con el Horizontal-Pel-Panning-Register. En estado normal contiene, según la anchura de los caracteres en el modo texto, el valor 0 u 8. Ambos se encargan de que los caracteres aparezcan en pantalla de la forma adecuada, en cuanto a su orientación horizontal, tal y como estamos acostumbrados.

El valor ocho representa la posición normal de todos los modos de vídeo, en los que los caracteres tienen una anchura de nueve puntos. Esto sirve básicamente para la tarjeta VGA así la como tarjeta EGA, si está conectada a un monitor monocromo, emulando así una tarjeta MDA. Para la tarjeta EGA, en un monitor color, el valor para la posición básica es por lo contrario, cero.
Partiendo de la posición básica, puede hacer que la pantalla se desplace punto a punto hacia la izquierda, aumentante el valor en el Horizontal-Pel-Panning-Register paso a paso. En una tarjeta EGA en un monitor color, se colocan los valores 1, 2, 3, etc. y cuanto mayor es el valor, tanto más se desplaza la pantalla a la izquierda. En el valor 7, ya se ha llegado al final del desplazamiento posible, ya que un valor mayor tendría como consecuencia un desplazamiento de más de un carácter. Pero esto no se puede conseguir con el Horizontal-Pel-Panning-Register.

El procedimiento es ligeramente diferente en las tarjetas VGA y EGA conectadas a un monitor monocromo. Allí, después del valor inicial de ocho no viene el nueve, sino el valor 0, que desplaza la pantalla un punto a la izquierda. Igual que en las tarjetas EGA, a este le siguen los valores 1, 2, 3, etc. hasta que, también aquí, se llega al desplazamiento máximo con el valor 7.
Ya que el desplazamiento a través del Horizontal-Pel-Panning-Register se realiza siempre hacia la izquierda, queda por resolver la pregunta de cómo se puede realizar un desplazamiento hacia la derecha. Pero esto no es difícil, si simplemente comienza con un valor de siete, y retrocede paso a paso hasta el valor cero. En las tarjetas VGA y las tarjetas EGA en monitores MDA, puede pasar del valor 0 al valor 8, para llevar la posición de la pantalla de nuevo a la posición básica.

Pero el Horizontal-Pel-Panning-Register no sólo se puede emplear en el modo de texto, sino sobre todo en el modo gráfico. Se diferencia entre los modos de 256 colores de una tarjeta VGA, y todos los demás modos. En el modo de 256 colores la posición básica es igualmente cero, pero la pantalla sólo se puede desplazar tres puntos hacia la izquierda. Pero para ello no se han de cargar los valores 1, 2 y 3 en el Horizontal-Pel-Panning-Register, sino los valores 2, 4 y 6.
En todos los demás modos gráficos, la posición inicial es cero, y se pueden cargar los valores 1 a 7 en el Horizontal-Pel-Panning-Register, para desplazar la zona visible de pantalla un máximo de siete puntos a la izquierda.
El Horizontal-Pel-Panning-Register se carga bien con programación directa del controlador de atributos, o con ayuda de una función del BIOS, que en realidad se dedica a otra tareas totalmente distintas. Se habla de la subfunción 00h de la función 10 de la interrupción del BIOS de vídeo. En realidad, con su ayuda se debería cargar un registro de paletas individual determinado, para lo que se ha de indicar su número.

Pero como los registros de paletas, al igual que el Horizontal-Pel-Panning-Register se encuentran en el controlador de atributos, a través de esta función se puede direccionar también el Horizontal-Pel-Panning-Register, no indicando ningún número de registro de paletas, sino el número de este

registro. Para ello, aparte de los dos números de función simplemente se ha de cargar el número de registro, 13h, en el registro BL, y en el registro BH se ha de colocar el valor a pasar al registro.

Si no es necesario que todo vaya especialmente rápido, este método es preferible al acceso directo al controlador de atributos, ya que esta es la forma de aislar un programa de las diferencias entre el hardware de las diferentes tarjetas EGA y VGA. Por otra parte, la mayoría de fabricantes de tarjetas EGA y VGA mantienen la ocupación de registros descrita por IBM, de forma que aún con la programación directa del Horizontal-Pel-Panning-Register no se puede uno equivocar en mucho.

Si elige este camino directo, primero se debe escribir el número del registro (13h) en el registro combinado de datos e índice del controlador de atributos en la dirección de puerto 3C0h. No se olvide de colocar a uno el bit 5 de este registro, que en realidad no tiene nada que ver con este registro, pero que representa, por así decirlo, el interruptor de conexión del controlador de atributos. Si se borra, el controlador de atributos deja de trabajar, y la pantalla se queda negra.

Después de haber enviado el valor 33h (número de registro 13h más bit 5) al puerto 3C0h, se puede colocar allí el nuevo valor para el registro direccionado, en este caso, el nuevo contador de pixeles para el Horizontal-Pel-Panning-Register.
Hemos visto el desplazamiento de la zona de pantalla visible. Algo más sencillo es el desplazamiento vertical de la pantalla mediante el Vertical-Pel-Panning-Register, ya que, independientemente de la tarjeta de vídeo o del modo de representación, el valor 0 siempre marca la posición inicial. Todo valor mayor desplaza la pantalla una cantidad de pixeles correspondiente hacia arriba. El límite superior se marca en los modos de texto mediante la altura de los diferentes caracteres. En el modo de texto normal de 80×25 caracteres, en la EGA este valor es 13, mientras que en la tarjeta VGA puede llegar hasta 15.

En el modo gráfico, el Vertical-Pel-Panning-Register puede acoger valores entre 0 y 31, donde la posición inicial queda representada por el 31, y no por el 0. Cada valor inferior desplaza la zona de pantalla visible una cantidad de puntos correspondiente hacia abajo. Por ello, primero se ha de colocar el Vertical-Pel-Panning-Register en su posición mínima, para poder desplazar hacia arriba la zona de pantalla visible mediante el incrementado de este registro.

Para desplazar la zona de pantalla visible hacia abajo, se hace al igual que en el modo de texto con el desplazamiento horizontal hacia la derecha: se comienza simplemente con el valor más alto, y se va reduciendo en uno, hasta que se vuelve a alcanzar la posición básica.
A diferencia que en el Horizontal-Pel-Panning-Register, el Vertical-Pel-Panning-Register se ha de direccionar directamente a través de la controladora CRT ahí no ayuda ninguna función del BIOS. Para ello se ha de colocar primero el número de registro, aquí 08h, en el registro de índice de la controladora CRT. En qué dirección se encuentra este registro de índice, depende del modo de funcionamiento de la tarjeta de vídeo: en el modo de color, se encuentra en la dirección de puerto 3D4h, en el modo monocromo en la dirección 3B4h. Inmediatamente a continuación viene en cualquier caso el registro de datos, al que se le ha de transmitir el nuevo valor para el Vertical-Pel-Panning-Register. En vez de una operación de 8 bits, aquí se puede emplear una operación de 16 bits y se puede enviar simultáneamente el valor para el registro de datos e índice.

Pero a menudo no basta con la programación de los dos Pel-Panning-Register, ya que generalmente no se trata de desplazar la zona visible de la pantalla por espacio de un carácter. Más bien al contrario se pretende desplazar la pantalla continuadamente en una dirección determinada, y esto carácter a carácter.
A primera vista se ofrece el método de desplazar primero la zona de pantalla con ayuda de los dos Pel-Panning-Register un carácter en la dirección deseada, poner a cero el Pel-Panning-Register modificado, y desplazar simultáneamente el contenido de la RAM de vídeo de forma, que todos los

caracteres se desplacen una posición en la dirección deseada. Este proceso se puede repetir tantas veces como se desee, para darle al usuario la impresión de una pantalla en movimiento continuo.

En la práctica se podrá disfrutar muy poco de este procedimiento, ya que el desplazamiento de los caracteres en la RAM de vídeo cuesta simplemente demasiado tiempo, y además es visible para el usuario, si no se trabaja con dos páginas de pantalla diferentes, de las que se una siempre se procesa en el fondo, y después se hace visible de nuevo.
Por ello se utiliza normalmente otra técnica, que incluye la posibilidad de seleccionar libremente la dirección inicial de la pantalla y de modificar la longitud de líneas en la RAM de vídeo. Más sobre ello, en el siguiente apartado.

Desplazamiento del inicio de la pantalla en la RAM de vídeo y modificación de la longitud de línea.
Si se quiere desplazar el contenido de la pantalla en el modo de texto una línea hacia arriba, se puede desplazar para ello el contenido completo de la RAM de vídeo de la página de pantalla mostrada, 160 bytes hacia arriba, que es lo que mide una línea del modo de texto de 80×25 caracteres. Pero también se puede incrementar la posición inicial de la página de pantalla en 160 bytes, ya que de esta forma, la controladora CRT comienza directamente en la segunda línea de pantalla al construir la próxima pantalla. El resultado desde el punto de vista del usuario en ambos casos es el mismo, pero desde el punto de vista del programa en marcha ocurren dos cosas totalmente diferentes.

En vez copiar toda la RAM de vídeo, simplemente se reprograma un registro en la controladora CRT, que allí existe especialmente para acoger la dirección de inicio de la página de pantalla, y que a causa de ello es responsable para la conmutación entre las diferentes páginas pantalla. Esto no sólo es mucho más rápido, sino que además guarda la línea de pantalla desplazada fuera de la pantalla en la RAM de vídeo, por si después fuera necesario volver a mostrarla.
El desplazamiento de la dirección de inicio de la RAM de vídeo se parece con ello al movimiento del cursor en un programa de tratamiento de textos, con cuya ayuda se puede traer otro trozo del texto en la ventana de texto visible en la pantalla. En el lugar de las teclas de cursor se utiliza aquí el registro de inicio CRT, y en vez del texto, es el contenido de la RAM de vídeo el que se hace visible pantalla a pantalla.

En concreto se trata de los dos registros CRT 0Ch y 0Dh, en los que se retiene la dirección de inicio de la parte visible de la RAM de vídeo. Estos registros no se pueden alcanzar directamente a través del BIOS, de forma que se ha de acceder de nuevo directamente al controlador CRT. Hay dos cosas que se han de tener en cuenta: primero, la dirección de inicio se guarda en forma de una dirección de offset de 16 bits, cuyo byte alto estará en el registro 0Ch y cuyo byte bajo estará en el registro 0Dh. Al contrario que en el ordenamiento de words en la memoria, el orden entre byte bajo y byte alto han sido invertidos, ya que el byte alto se encuentra antes que el byte bajo. Segundo, el offset no se cuenta en bytes, sino en words, así que se ha de dividir en dos, antes de que se escriba en los registros.

Mientras que la zona visible de pantalla se puede desplazar en vertical fácilmente a través de docenas o incluso cientos de líneas, con la combinación de Pel-Panning y el desplazamiento del inicio de pantalla, este mecanismo muestra desagradables fallos en movimientos de dirección horizontal. Se tiene una desagradable sorpresa después de desplazar la pantalla un carácter a la izquierda con ayuda del Horizontal-Pel-Panning-Register, y cuando se quiere desplazar el inicio de pantalla un carácter a la izquierda y se incrementa para ello la dirección de inicio para que por ejemplo no comience en 0000h sino en 0002h.

Realmente, con ello aparece en la esquina superior de la pantalla el carácter que hasta ahora aparecía en la segunda columna de la primera línea, pero a cambio, todos los demás caracteres de la

primera columna han saltado a la última columna de la línea anterior. Esto se llama reiniciación cíclica de las líneas en la pantalla (ing: Wrap-Around).
Ya que este problema no se puede solucionar con ayuda del registro de inicio CRT a causa la estructura organizativa de la RAM de vídeo, se necesitaría copiar de nuevo la RAM de vídeo, lo que en realidad se quería evitar. Por suerte, la controladora CRT contiene un registro, que ayuda en este caso: el llamado registro de offset. Acoge la longitud de una línea en la RAM de vídeo y en el modo de texto contiene normalmente el valor 80, ya que se necesitan 80 words y con ello 160 bytes por línea.

Si se aumenta este valor, no se representan más caracteres por línea en la pantalla, sino que el contador de direcciones interno de la controladora CRT se incrementa en un valor mayor por línea. Así que si en vez del valor 80, se escribe 82 en este registro, la línea en la RAM de vídeo se compone de 82 caracteres, de los que aparecen, como siempre, 80 en pantalla. Los caracteres que sobran, se pueden traer a la zona de pantalla visible, aumentando simplemente el registro de inicio CRT en un valor de 1 o 2. Pero a cambio también desaparecen caracteres del borde izquierdo de la pantalla.

Ya que el registro de offset, que en la controladora CRT lleva el número 13h, tiene una anchura de ocho bits, una línea de la RAM de vídeo se puede expandir hasta un máximo de 255 caracteres. Aunque en ese caso ocupa 510 bytes en la RAM de vídeo, y eso se ha de tener en cuenta, cuando se escriban caracteres a la RAM de vídeo, para que aparezcan después en un punto determinado de la pantalla.
Mientras que en el cálculo de la dirección de offset de un carácter en la RAM de vídeo se ha de multiplicar siempre la línea indicada con 160, este factor ahora es 510, 320 o lo que sea. Pero en cualquier caso siempre se corresponde con el doble de la cantidad de caracteres que se retienen en la RAM de vídeo en una línea.

Una vez que se ha determinado una anchura interna mayor de las líneas, ya no hay ningún problema para el desplazamiento horizontal de la zona visible de pantalla mediante la combinación de Pel-Panning-Register y el inicio de pantalla. Sin embargo, al igual que para el desplazamiento vertical, se ha de pensar en la sincronización de los hechos, ya que sino la práctica puede tener un aspecto totalmente diferente que la teoría.
Sincronización del desplazamiento con la controladora CRT

Si se quiere conciliar el desplazamiento de la zona visible de pantalla mediante los Pel-Panning-Register y el inicio de pantalla, también se ha de considerar la controladora CRT, que en este asunto tiene mucho que decir.
Para comenzar, sólo se debería acceder a los Pel-Panning-Register durante el retrazado vertical del rayo de electrones, es decir, cuando no se están dibujando informaciones en la pantalla. Con ello se asegura que la modificación de estos registros no haga aparecer una parte de la pantalla de forma diferente a causa de que los registros ya están modificados, que la que ya está dibujada.

Si los Pel-Panning-Register y el registro de inicio CRT han de ser programados simultáneamente, para por ejemplo reponer los Pel-Panning-Register a su posición inicial y desplazar el inicio de pantalla, se ha de andar con cautela. Mientras que los contenidos modificados de los Pel-Panning-Register ya se emplean durante la siguiente reconstrucción de la pantalla, para el registro de inicio CRT ya es demasiado tarde.
Eso tiene que ver con que el programa sólo se entera del inicio del retrazado vertical con un poco de retraso, aunque naturalmente no se trata de segundos, sino fracciones de ellos. Esto tiene que ver con el control del retrazado vertical, que normalmente se realiza a través del bit 3 del llamado registro de estado de entrada (Input-Status-Register). Indica el estado del retrazado vertical, y durante este retrazado contiene siempre el valor 1, de lo contrario el valor 0. Este retraso no deseado

se genera por que este registro se ha de controlar una y otra vez en un bucle de programa, lo que genera un retraso temporal, ya que este control implica unos comandos de lenguaje máquina, cuya ejecución naturalmente necesita algo de tiempo. Y esto es tanto más cierto, si se programa en un lenguaje de alto nivel, ya que allí no se puede generar un código de control tan optimizado, como es posible en la programación directa en ensamblador.

Si un programa reconoce el retrazado vertical, puede que ya esté en marcha desde hace una fracción de segundo, y para entonces la controladora CRT ya ha cargado la dirección inicial para la siguiente construcción de pantalla, desde el registro de inicio CRT. Una modificación de este registro no se tendrá en cuenta durante la siguiente reconstrucción de la pantalla, sino durante la próxima.
Pero como las modificaciones en los Pel-Panning-Register sí que se tienen en cuenta durante la siguiente construcción de la pantalla, aquí se crean disonancias, que no le quedarán ocultas al usuario. Por ello, durante la modificación simultánea de los Pel-Panning-Register y del registro de inicio CRT, es necesario un procedimiento coordinado.

Se ha de esperar, a través del llamado registro de estado de entrada 1, la aparición de un retrazado vertical y a continuación se ha esperar su final. Después se carga el nuevo inicio de pantalla en los registros correspondientes de la controladora CRT. La construcción de la pantalla no se molesta por esto, ya que la controladora CRT ya ha cargado la dirección inicial antes de comenzar la nueva construcción de pantalla en su contador de direcciones interno, y por ello no necesita acceder de nuevo a ella.

A continuación se espera el final de la construcción de la pantalla, y con ello el inicio del retrazado vertical. Si este evento ocurre, se pueden programar rápidamente los Pel-Panning-Register, cuya modificación se muestra efectiva en la construcción de pantalla siguiente. Esto también es válido para la nueva dirección de inicio del registro de inicio CRT, que ahora se cargará con toda seguridad en el contador de direcciones antes de la nueva construcción de pantalla.

La espera al retrazado vertical del rayo de electrones tiene como consecuencia que la ejecución del programa se pueda sincronizar, independientemente de la velocidad del ordenador en cuestión, con la frecuencia de pantalla de la tarjeta de vídeo, lo que es muy ventajoso sobretodo en juegos. Si llama la rutina para ajustar los Pel-Panning-Register y el registro de inicio CRT una y otra vez, y de forma estrictamente secuencial, se puede desplazar la zona visible de pantalla uno o varios puntos con cada construcción de pantalla. Según si utiliza una tarjeta EGA o VGA, y según el modo de vídeo esto pueden ser entre 50 y 70 desplazamientos por segundo. El ojo humano naturalmente no es capaz de seguir este ritmo, y verá un movimiento continuo, incluso si se realizan desplazamientos sobre varios pixeles.

Letras en movimiento en el modo de texto
Como ejemplo para una programación combinada de los Pel-Panning-Register y del registro de inicio CRT, les queremos presentar un programa, con cuya ayuda puede hacer correr un texto en grandes letras, de la derecha a la izquierda de la pantalla. Según su tarea, los dos programas se llaman TXTMOVC.C y TXTMOVP.PAS, cuyas siglas nos vienen sugeridas por «texto en movimiento».
En el centro de ambos programas están las dos rutinas con los nombres ShowLaufText y SetOrigin. ShowLaufText tiene la tarea de recoger el texto a visualizar, y de construirlo en la RAM de vídeo, sin que por ello sea totalmente visible para el usuario. Para poder aprovechar al máximo la longitud completa de 32 KB de la RAM de vídeo, los dos programas la dividen en tres bandas, que tienen 216 caracteres de ancho y 25 líneas de alto. Cada banda ocupa por ello 10800 bytes en la RAM de vídeo, lo que da un total de 32400 bytes y sólo deja sin utilizar 368 bytes.

Después de haber construido el texto en la RAM de vídeo, ShowLaufText comienza con su visualización, donde el procedimiento SetOrigin tiene un papel fundamental. Con su ayuda se determina el inicio de pantalla correspondiente, donde se ha de indicar la banda a mostrar, la columna y línea inicial y el desplazamiento en puntos. Estas últimas indicaciones se guardan sin cambios en los Pel-Panning-Register, mientras que el número de banda se ha de operar con la línea y columna iniciales, para formar la dirección inicial de la zona visible de pantalla. Esta se escribe en el registro de inicio CRT.

Para darle al usuario la impresión de un texto en movimiento, se llama a SetOrigin en ShowLaufText con un inicio de columna y línea constante, donde sólo se incrementa paso a paso el contador de pixeles para el desplazamiento horizontal. Cuando se haya llegado al desplazamiento máximo de un carácter, se coloca el contador de pixeles a 0 (u 8), y se incrementa la columna de inicio.
El desplazamiento puede ocurrir en diferentes velocidades, y también se tiene en cuenta la diferencia de anchura de los caracteres de las tarjetas EGA y VGA. ShowLaufText espera por ello, aparte del texto a mostrar, dos parámetros que reflejen la velocidad deseada y la tarjeta de vídeo instalada. Para la velocidad se pueden indicar las constantes FAST, MEDIUM, y SLOW, mientras que para el tipo de tarjeta de vídeo se aceptan las dos constantes EGA y VGA.

Las dos constantes se emplean durante el desplazamiento del inicio de la pantalla en ShowLaufText, como índice al array StepTable, que se declara en el interior de esta rutina. Aquí se retienen, por separado para las tarjetas EGA y VGA, los valores que dependen de la velocidad seleccionado, y que SetOrigin coloca sucesivamente en el contador de pixeles horizontal. El final de esta serie lo marca el valor 255 en la matriz, que señala, que ahora se ha de continuar con la siguiente columna de pantalla.

Las diferentes velocidades, al mostrar el texto en movimiento, se consiguen mediante diferentes amplitudes de paso al aumentar el contador horizontal de pixeles. Mientras que en el modo SLOW recorre cada punto de 0 a 7 (o bien de 8 a 7), en el modo MEDIUM ya se salta uno de cada dos puntos, con lo que se desplaza todo el doble en la pantalla, en la misma cantidad de tiempo. En el modo FAST, la velocidad se duplica de nuevo, ya que cada carácter se desplaza de cuatro en cuatro pixeles.

El incremento de la columna inicial llega a su límite, en cuanto la última columna de la banda, la columna 216, se ha hecho visible en pantalla. En este caso se ha de cambiar la banda, y se ha de seguir con su primera columna. Pero como el contenido de la banda anterior no se pudo desplazar hacia afuera de la pantalla, en el principio de la siguiente banda se ha de comenzar con los mismos caracteres de la última pantalla de la banda anterior. Sólo con ello se puede obtener un cambio sin problemas entre las diferentes bandas, haciéndolo inapreciable para el usuario.

La cantidad de caracteres que se pueden mostrar en total en pantalla, y delante de los ojos del usuario, depende naturalmente de la anchura de los mismos. Para construir los caracteres en la RAM de vídeo, hay una rutina responsable de ello, con el nombre PrintChar. PrintChar espera como argumento, aparte del carácter en si, la columna y la banda, en la que ha de aparecer el carácter. La línea se especifica mediante una constante (STARTZ), y por ello no se ha de pasar específicamente. El patrón de puntos de los caracteres esta rutina lo toma del juego de caracteres de 8×14 puntos, que existe tanto en tarjetas EGA como VGA en un componente ROM en la tarjeta. Este juego de caracteres se puede leer, después de la su dirección se haya averiguado con ayuda de la subfunción 30h de la función 11h de la interrupción de vídeo del BIOS.

Esta rutina es la razón, por la que la versión en C del programa TextMov no pueda pasar sin un pequeño módulo en ensamblador, que tiene el nombre de TXTMOVCA.ASM. Ya que la función

del BIOS antes mencionada devuelve informaciones en el registro BP, las funciones normales en C no nos sirven para la llamada de la interrupción, y por ello se han de sustituir por esta rutina dedicada.
La matriz de caracteres de 8×14 se transfiere a la pantalla mediante PrintChar, visualizando cada uno de los puntos de esta matriz en una posición de pantalla. Cada carácter se compone por ello de 8 columnas de texto y 14 líneas de texto. En una banda caben en consecuencia 27 caracteres (216 / 8), de los cuales hay 10 visibles simultáneamente (80 / 8). Pero como en la segunda y tercera banda primero hay que repetir la última pantalla de la banda anterior, en vez de 27, sólo se añaden 17 caracteres. En total se pueden visualizar y guardar 61 caracteres (27 + 17 + 17) diferentes de esta forma en la RAM de vídeo.

Desconexión de la pantalla
Siempre hay situaciones en las que se ha de suprimir la visualización por pantalla de la tarjeta de vídeo, por ejemplo en un protector de pantalla, que evita que los tubos de rayos catódicos se quemen a causa de imágenes fijas. También las tarjetas EGA y VGA conocen varias posibilidades para suprimir la generación de una imagen de vídeo. De estas posibilidades hablaremos en este apartado. Trataremos las siguientes cuestiones:

- Qué es lo que el controlador de atributos (Attribute-Controller) tiene que ver con la generación de la imagen de vídeo y...

- Cómo se puede aprovechar su papel para apagar la visualización en pantalla.

El papel del controlador de atributos en la generación de imágenes de vídeo
En la cuadriga de CRT-Controller, Graphics-Controller, Attribute-Controller y Sequencer-Controller, que pilotan las tarjetas EGA y VGA, el Attribut-Controller tiene la responsabilidad de la conexión entre la información de imagen y color. Si se puentea su forma de trabajo, ya no se suministran informaciones de color, con lo que la pantalla se oscurece automáticamente. Esto es equiparable a una desconexión de la pantalla y con ello hace las funciones que deseamos. Mientras que los otros controladores no se pueden interrumpir con facilidad en su trabajo, en el Attribut-Controller es especialmente sencillo. Aquí hay un bit en el registro de índice de este controlador, mediante el que se le puede prácticamente desconectar. Se trata del bit 5, que simplemente se pone a cero para suprimir la preparación de informaciones de color mediante el controlador.

Sin embargo, la posición de este bit es un tanto extraña, ya que en todos los demás controladores, el registro de índice sirve sólo para aceptar el número de uno de los registros a direccionar. Que el registro de índice del *AC sea distinto tiene que ver con los registros de atributos, que probablemente son los más direccionados: los registros de paleta. Ya que antes de acceder a un registro de paleta para lo cual se ha de direccionar el registro de índice de todas formas se ha de interrumpir brevemente el trabajo del controlador, tiene sentido el colocar el bit responsable directamente en el registro de índice. Esto es tanto más cierto en cuanto el *AC dispone solamente de 21 registros diferentes, así que en el registro de índice sólo se necesitan los bits 0 a 4 para aceptar el número de registro.

Pero antes de enviar de vacaciones al *AC borrando el quinto bit, se ha de leer primero el registro de estado del Controlador CRT (CRT-Controller) ya que sólo entonces se hace efectivo el reset del *AC. Esto también es cierto si este bit se ha de poner de nuevo a 1, para que el *AC vuelva a realizar su trabajo.
Desconexión de la pantalla en la práctica.
Los dos programas VONOFFP.PAS y VONOFF.C demuestran en vivo y en directo la conexión y desconexión de la pantalla mediante al *AC. Pero esto sólo ocurre después de que se hayan cerciorado de que realmente hay presente una tarjeta EGA o VGA. De otro modo, el programa se termina después de un breve mensaje.

Si se descubre una tarjeta EGA o VGA, a continuación se visualiza un breve mensaje, y se le dan al usuario cinco segundos de tiempo, para leerlo. Sólo después se desconecta la pantalla con ayuda del procedimiento o función ScrOff.
Como ya hemos descrito, en ScrOff primero se lee el registro de estado del Controlador CRT. Para que tanto las tarjetas que están conectadas a monitores en color, como sus homólogas, conectadas a monitores monocromos puedan sacar partido de esta rutina, se realiza directamente un reset doble: una vez por la dirección del registro de estado monocromo (3BAh) y otra vez por la dirección en color (3DAh). Además, no puede ocurrir nada peligroso, ya que uno de los dos ports no estará ocupado, y su lectura no traerá consigo ninguna consecuencia.

Después se escribe el valor 0 en el registro de índice del controlador de atributos en la dirección de puerto 3C0h, se pone a cero el bit 5 con lo que se puentea el trabajo de este controlador. A causa de ello, la pantalla se pone negra inmediatamente.
No ocurre de forma muy diferente con el procedimiento ScrOn, que se llama en los dos procedimientos, después de que el usuario haya pulsado una tecla. Aquí, simplemente se escribe el valor 20h en el registro de índice del controlador de atributos, y con ello se vuelve a «despertar» el controlador. A continuación vuelve a aparecer una imagen de vídeo.

El principio de los bit-planes
La historia de las tarjetas de vídeo de PC está marcada claramente por un afán de conseguir cada vez mayores resoluciones con un número de colores cada vez mayor. Pero con una resolución mayor, también crece la demanda de RAM de vídeo. Esto es tanto más cierto, en cuanto, a causa del número cada vez mayor de colores, la codificación de los diferentes puntos ya no necesita 1 bit, sino 4 o 8 bits, como en los modos de 256 colores de las tarjetas VGA. En la actualidad, con la norma True-Color se necesitan hasta 24 bits, por cada punto de color. Esto ha llevado a que las tarjetas EGA y VGA actuales estén equipadas con una RAM de vídeo de 256, 512 o 1024 KBytes, que según el modo de vídeo y la cantidad de páginas de pantalla gestionadas hasta se emplea totalmente.

Pero ya sobrepasando el límite de los 64 KBytes, resulta un problema, ya que en el espacio de memoria direccionable de 1 MB del PC, simplemente no hay espacio para poder acceder a más RAM de vídeo. Por ello, este apartado se ocupa de la cuestión de como se encajó la RAM de vídeo de 256 KBytes de las tarjetas EGA y VGA en la zona direccionable del PC, y qué implicaciones resultan de ello para la programación de estas tarjetas. Se tratarán los siguientes temas uno a uno:
o La división de la RAM de vídeo en bit-planes

o La tarea y el significado de los registros de retención (latch)
o El papel del Graphics Controller para la programación de gráficos
o La programación y el modo de funcionamiento de los diferentes modos Read yWrite
La división de la RAM de vídeo en bit-planes
Para las tarjetas de vídeo, el PC sólo dispone, en su zona de direccionamiento, de los segmentos de memoria A (a partir de A000:0000) y B (a partir de B000:0000), donde el segmento B ya está reservado para la RAM de vídeo de tarjetas MDA, CGA y Hercules. Para las tarjetas EGA y VGA sólo queda el segmento A, que los primeros diseñadores de la tarjeta EGA pusieron ante la ingrata tarea de administrar 256 KBytes de RAM de vídeo a través de una ventana de sólo 64 KBytes.

Encontraron la solución en forma de los llamados bit-planes, que se reparten, entre sí uniformemente, la RAM de vídeo de las tarjetas EGA y VGA. Teniendo equipada una tarjeta de vídeo con 256 KBytes, a cada bit-plane le corresponden por ello 64 KBytes, que se pueden direccionar directamente mediante el segmento A, a partir de la dirección A000:0000. Si la memoria es menos, como por ejemplo en las primeras tarjetas EGA originales de IBM, con sólo 64 KBytes, a cada bit-plane le corresponde una pa rte proporcionalmente inferior de la RAM de vídeo.

Este principio de organización, que procede del año 1984, tanto ayer como hoy tiene validez para todos los tipos de tarjetas EGA y VGA, aunque en las tarjetas Super-VGA haya sido ampliado ligeramente, como mostrará el apartado 9.9.

Las tareas y el significado de los registros latch

Mientras que los diferentes bit-planes sólo son importantes para la programación del modo de texto, cuando se quiere acceder manualmente a diferentes juegos de caracteres, se encuentran como una espada de Damocles sobre todos los modos gráficos. Y no tiene ninguna importancia, como se distribuyen las informaciones de los puntos en los diferentes bit-planes, lo que depende exclusivamente del modo gráfico particular. En independencia total de ello, el hardware de la tarjeta EGA/VGA conmuta entre CPU y RAM de vídeo cuatro registros de ocho bits, los llamados registros latch (Latch-Register). Cada uno de estos registros se corresponde con uno de los cuatro bit-planes, pero no es accesible directamente a un programa.

Si un programa realiza un acceso de lectura a un byte en la RAM de vídeo, los cuatro registros latch se cargan junto al byte de su bit-plane, cuya dirección de offset se ha especificado en conexión con el comando de lenguaje máquina, que ha disparado el acceso de lectura. Si la dirección de offset es por ejemplo 9, se carga el décimo byte (0 es el primero) del primer bit-plane en el primer registro latch, el décimo byte del segundo bit-plane en el segundo registro latch, y correspondientemente se procede con el tercer y el cuarto registro latch. El mismo procedimiento también se emplea en un acceso de escritura en la RAM de vídeo. Por que entonces, el contenido de los registros latch se escribe en el bit-plane correspondiente, y concretamente en la dirección de offset, que se indicó en conexión con el acceso de escritura.

Ya que durante un acceso de lectura sólo se puede transferir un byte y no los cuatro bytes a la CPU, falta aclarar la cuestión de cual de los cuatro bytes es el que se envía a la CPU. Aparece una pregunta semejante, cuando se escribe en la RAM de vídeo, ya que seguramente no se quiere escribir el mismo valor en los cuatro registro latch y con ello un byte en todos los cuatro bit-planes. Ya que entonces, los cuatro bit-planes tendrían el mismo contenido, y sólo se podría acceder simultáneamente a 64 KBytes de la RAM de vídeo, y se podría ahorrar todo el lío de los bit-planes y los registros latch.

El papel del controlador de gráficos Graphics Controller para la programación

La respuesta a todas estas preguntas la dan los nueve registros del llamado Graphics Controller (Controlador Gráfico), que es una parte importante de las tarjetas EGA y VGA. Determinan el desde donde, el adónde y el cómo en todos los accesos de lectura y escritura a la RAM de vídeo. Esto, por cierto, no sólo es aplicable a todos los diferentes modos gráficos, sino también a los modos de texto. Mientras que en el ámbito de la programación de gráficos se ha de recurrir frecuentemente a estos registros, y se les han de asignar diferentes valores en función de la operación deseada, en el modo de texto no es así.

Aquí se ajustan los diferentes registros directamente durante la inicialización de uno de estos modos a través del BIOS, de manera que después no es necesario acceder a ellos. A pesar de todo, también es aplicable a los modos de texto que todas las entradas y salidas de la o a la RAM de vídeo mediante los cuatro registros latch, aunque este proceso sea transparente para el programa.

La programación de los registros del Graphics Controller se asemeja al acceso a los registros del CRTC de la tarjeta gráfica Hercules, pero también se emplea en otros controladores de las tarjetas EGA y VGA. En la dirección de puerto 3CEh existe un registro de direcciones, en el que primero se ha de cargar el número del registro del Graphics Controller, al que se quiere acceder. El valor para este registro se puede colocar a continuación en el registro de datos, que está a continuación del registro de direcciones y que lleva la dirección de puerto 3CFh.

El acceso a los dos ports no se ha de realizar por separado, sino que se puede realizar con ayuda de un comando OUT de 16 bits sobre el registro de direcciones, El registro AX, que en el marco de este comando OUT DX,AX se envía a este puerto, ha de contener el número de registro en el registro AL y el valor a cargar en este registro en el registro AH.
A pesar de que de esta forma se pueden cargar valores en los diferentes registros del Graphics Controller, pero un acceso de lectura sólo es posible en tarjetas VGA; en las tarjetas EGA estos registros no se pueden leer.

Es decisivo, sobre todo para el modo de proceder del Graphics Controller en accesos de lectura y escritura a la RAM de vídeo, el registro de modo (Mode-Register) que lleva el número 5. Determina uno de dos modos Read y uno de tres modos Write, que en un acceso de lectura y escritura a la RAM de vídeo. Todos los demás registros sólo son accesorios, que especifican determinados parámetros en dependencia del modo de lectura y escritura.
Los siguientes apartados se ocupan en detalle del modo de funcionamiento de los diferentes modos Read y Write, sus tareas y el significado de los otros registros del Graphics Controller para estos modos. Sin embargo se dará cuenta pronto, que para el trabajo práctico sólo se emplean pocos de estos modos, ya que el modo de funcionamiento de los demás es sencillamente demasiado retorcido.

Modo Read 0
El modo Read 0 le ofrece a un programa la posibilidad de leer bytes de un bit-plane determinado. Es útil, por ejemplo, cuando se quiere guardar una parte de la RAM de vídeo. Con un bucle se pueden recorrer sucesivamente los cuatro bit-planes, cargando la zona deseada de cada bit-plane y colocándola en la memoria principal.
A pesar de que en un acceso de lectura a la RAM de vídeo en el modo Read 0 se cargan todos los cuatro registros latch del byte direccionado en su plano (plane), pero sólo uno de estos cuatro bytes llega hasta la CPU a través de su registro latch. De qué bit-plane viene este byte, queda determinado por el contenido del Read-Map-Select-Register, que en el Graphics Controller lleva el número 4.

En este registro sólo se han ocupado los dos bits más bajos y ellos deciden sobre el número de registros latch, cuyo contenido deberá ser desplazado hasta la CPU. Tenga en cuenta, por favor, que en la programación de este registro el primer Bit-plane lleve el número 0, y no el número 1, si quiere acceder al primer Bit-plane.
La siguiente secuencia de comandos en ensamblador demuestra el empleo del modo Read 0. Para ello se cargan los primeros 8 KBytes del segundo bit-plane en la memoria principal. Primero se activa el modo Read 0, y a continuación se carga el valor 1 en el Read-Map-Select-Register, para poder leer el segundo bit-plane. Con ayuda del comando de ensamblador REP MOVSB se cargan a continuación los primeros 8 KBytes, byte a byte, desde la RAM de vídeo a un buffer.

Si se fija un poco en la secuencia anterior en ensamblador, a lo mejor objetaría, que se puede acelerar el proceso de copia, copiando, en vez de 8192 bytes, 4096 words, cambiando simplemente el comando REP MOVSB en REP MOVSW, y cargando en el registro CX el valor 4096 en vez de 8192. Al copiar zonas de memoria dentro de la memoria principal esto realmente estaría justificado, pero aquí tendría consecuencias nefastas.
Ya que no son posibles los accesos de 16 bits en la RAM de vídeo, la CPU realizaría en este caso dos accesos de lectura a nivel de byte, antes de realizar el acceso de escritura. Mediante el segundo acceso de lectura, sin embargo, se sobrescribe el resultado del primero acceso de lectura en el registro latch, antes de que llegue a la CPU. Como consecuencia, en el buffer destino se encontrarán palabras, cuyo byte bajo y alto tienen el mismo valor, pero donde sólo el byte alto corresponde al verdadero contenido del de la RAM de vídeo. Por ello, ¡se han de evitar accesos de lectura de 16 bits en tarjetas EGA y VGA a toda costa!

Modo Read 1

Si el sentido y el modo de funcionamiento del modo Read 0 aún era relativamente fácil de comprender, en el modo Read 1, la cosa se complica. No se trata de enviar el contenido de un registro latch a la CPU sin cambios, sino que se realizan numeras operaciones lógicas, que afectan el contenido de todos los cuatro registros latch.
La tarea de este modo es verificar si los bits de los cuatro registros latch tienen un valor determinado. Esto se puede utilizar después, en los modos gráficos de las tarjetas EGA y VGA, para buscar puntos de un color determinado. Pero en la práctica, este modo se emplea muy poco.

Después de cargar los cuatro registros latch, primero se forman ocho grupos, compuestos respectivamente de cuatro bits. En uno de estos grupos se encuentran los cuatro bits, que dentro de los diferentes registros latch ocupan la misma posición de bit. Así por ejemplo todos los bits de la posición de bit 0 se reúnen en un grupo, igual que ocurre con los bits de la posición de bit 1.
Cada uno de estos ocho grupos de cuatro se comparará con el valor del registro de comparación (Color Compare Register), que se cargó en este registro antes del acceso de lectura. Del resultado de esta comparación se obtiene el byte que se envía a la CPU como resultado de la operación de lectura.

En este byte todos los bits, cuyo grupo de cuatro correspondiente tenga el valor del Color Compare Register estarán a 1. Todos los demás contendrán el valor 0. Por ello, después del acceso de lectura se puede determinar con exactitud cual de los grupos de cuatro correspondía al valor del Color Compare Register.
Tan extraño como pueda parecer a primera vista; ni siquiera es la verdad completa. Ya que durante la operación completa también entra en juego el Color Compare Register, y solo mientras este registro 00001111b, los diferentes grupos de cuatro se confrontan directamente con el valor comparativo del Color Compare Register. Cada uno de los cuatro bits bajos del Color Compare Register está por uno de los cuatro bit-planes, el bit 0 por el primero, el bit 1 por el segundo, etc. Sólo cuando uno de estos bits contiene el valor 1, el bit-plane correspondiente se incluye en la comparación de los grupos de cuatro del Color Compare Register.

Si de lo contrario el valor es 0, se hace como si el valor del Bit-Plane en los ocho grupos de cuatro coincide en cualquier caso con el bit correspondiente del Color Compare Register. Esto lleva a que al indicar el valor 0 en el Color Compare Register, siempre se devuelve el valor 11111111b a la CPU, independientemente del contenido de los cuatro registros latch y del Color Compare Register, ya que ya no se realiza una comparación con los ocho grupos de cuatro con el valor de comparación, sino que todos los grupos de cuatro se consideran como adecuados.

La siguiente secuencia en ensamblador demuestra el uso del modo Read 1, aunque en un contexto muy teórico. Se trata de determinar cual de los grupos de cuatro del primer byte en los cuatro bit-planes de la RAM de vídeo tiene el valor 5. El Color-Don't-Care-Register no se programa explícitamente, ya que se presupone que contiene el valor por defecto 00001111b y con ello se incluyen todos los bit-planes en la comparación.
Modo Write 0
Durante el acceso a la RAM de vídeo con el modo Write 0 se realizan una serie de operaciones, que dependen del contenido de diferentes registros. Básicamente, el contenido del registro de representación de bits (Bit-Mask-Register) decide sobre si el contenido de un bit en los cuatro registros latch se transferirá sin cambios a los cuatro bit planes, o si ha de ser manipulado primero. Los diferentes bits del Bit-Mask-Register se corresponden con los bits individuales en los cuatro registros latch. Si un bit del Bit-Mask-Register contiene el valor 0, el bit correspondiente en los cuatro registros latch se transfiere sin cambios a los cuatro bit planes. Si de lo contrario el bit está a uno, se realiza una operación lógica, cuyo tipo queda definido mediante el contenido de este registro llamado Function-Select-Register. Como muestra la siguiente figura, los bits simplemente se pueden sustituir, o manipular con una operación lógica Y, O u O EXCLUSIVO.

Quién es el compañero de estos bits en la operación, lo decide el contenido del Enable Set/Reset Register. Cada uno de estos bits se emplea para la operación con los cuatro bits de una posición de bit de los cuatro registros latch, cuyo tipo se describe mediante el contenido del Function-Select-Register.
Todos los bits a manipular del registro latch 0 se operan con el bit 0 del registro de Set/Reset, mediante la operación elegida. De la misma forma, todos los bits a manipular de los registros 1, 2 y 3 se operan respectivamente con los bits 1, 2 y 3 del registro Set/Reset. El byte de CPU, que aquí se transfiere incondicionalmente al controlador de gráficos, aquí no tiene ninguna importancia el acceso de escritura se degrada a un momento de disparo, que no tiene influencia sobre el contenido de los registros latch y con ello de los bit planes.

La siguiente secuencia en ensamblador tiene la finalidad de darles el código 1011b a los grupos de cuatro del bit 2 del primer byte de la RAM de vídeo sin modificar el contenido de los demás grupos de cuatro. Como verá en uno de los siguientes capítulos, esta es una técnica que se emplea frecuentemente al visualizar puntos individuales en los modos gráficos de 16 colores de las tarjetas EGA y VGA.
Ya que el color de los otros grupos de cuatro no se quiere modificar, primero se carga su contenido mediante un acceso de lectura a la RAM de vídeo en los registros latch. Cual de los modos de lectura está activo, no tiene importancia. En realidad, el valor devuelto a la CPU no tiene importancia, sino que simplemente se trata de rellenar los registros latch.

Ya que sólo se quiere manipular el grupo de cuatro en la posición de bit 2, traspasando los demás sin cambios a los bit-planes, primeramente se carga el valor 00000100b (04h) en el Bit-Mask-Register. A continuación se escribe el valor 0 en el Function-Select-Register, ya que los bits a manipular se quieren sustituir por una nueva combinación de bits. El color, que debe adoptar el grupo de cuatro en la posición de bit 2 (1011b=0Bh), se carga a continuación en el registro Set/Reset.

Para que también pueda ser extraído de este registro al escribir la RAM de vídeo, como último acceso a los registros del controlador gráfico se ha de escribir el valor 1111b (0Fh) en el Enable-Set/Reset-Register. Inmediatamente después puede ocurrir el acceso de escritura a la RAM de vídeo, donde el byte de procesador pasado ya no tiene ninguna importancia.

```
mov ax,0A000h       ;Dirección de segmento de la RAM de vídeo
mov ds,ax     ;a DS
mov al,ds:[0] ;Cargar Byte 0 en los registros latch
mov dx,3CEh ;Direccionar el Graphics-Controller
mov ax,0005h;Read-Mode 0, Write-Mode 0
out dx,ax     ;al Mode-Register
mov al,03h    ;Escribir 0 en el Function-
out dx,ax     ;Select-Register
mov ax,0408h;Escribir máscara de bits
out dx,ax     ;Bit-Mask-Register
mov ax,0B00h;Escribir nuevo valor de color en el
out dx,ax     ;Set/Reset-Register

mov ax,0F01h;Escribir 1111b en el Enable-Set/Reset-
out dx,ax     ;Register
mov ds:[0],al ;Manipular y devolver registros Latch
```

Ocurre de otra forma, cuando el Enable Set/Reset Register contiene el valor 0. En este caso todos los bits a manipular de los cuatro registros latch se operan con el byte de CPU. También aquí, el tipo de operación depende del contenido del Function-Select-Register. Si por ejemplo está seleccionado el tipo de operación O y se deben manipular los bits 1, 2, 4 y 6, estos bits de todos los registros latch se operan individualmente con los bits 1, 2, 4, y 6 en el byte de CPU mediante un O lógico.

A causa de que en este modo las diferentes posiciones de bit en los diferentes registros latch se operan con el mismo valor del byte de CPU respectivamente, este modo se utiliza con menos frecuencia que la operación mediante el Set/Reset-Register, tal y como se describe anteriormente.
Figura 145. Acceso de escritura a la RAM de vídeo en el modo Write 0, cuando el Enable Set/Reset Register contiene el valor 00000000h

Modo Write 1

Comparando con las complexas conexiones del modo Write 0, el modo Write 1 parece sencillo y poco importante. El contenido de los diferentes registros del controlador gráfico y del byte de CPU pasado, aquí no tienen ninguna importancia, ya que el contenido de los cuatro registros latch se escribe sin modificaciones en la dirección de offset especificada, dentro de los cuatro bit-planes.

La utilización de este modo Write sólo tiene sentido, por ejemplo, cuando una zona determinada de la RAM de vídeo debe copiarse a otra. En ese caso es suficiente el recorrer la RAM de vídeo, llenar el contenido de los cuatro latches mediante un acceso de lectura en un modo de lectura cualquiera, y escribir los registros latch a continuación mediante un acceso de escritura en el modo Write 1 en la RAM de vídeo. Con ello se pueden copiar cuatro bytes de una vez, lo que ahorra un montón de tiempo.

```
mov ax,0A000h       ;Dirección de segmento de la RAM de vídeo
mov ds,ax     ;a DS y ES
mov es,ax
mov si,0000h  ;Zona fuente comienza en 0000h
mov di,0200h  ;Zona destino en 0200h
mov cx,100h   ;Copiar 256 bytes
cld     ;contar hacia arriba en comandos de string
mov dx,3CEh ;direccionar Graphics-Controller
mov ax,0105h;Read-Mode 0, Write-Mode 1
out dx,ax     ;al Mode-Register
rep movsb     ;copiar los cuatro bit-planes de la zona
        ;de una sola vez
```

Modo Write 2

El modo Write 2 se parece a una combinación de los diferentes modos en el modo Write 0. También aquí, el registro Bit-Mask decide análogamente al modo Write 0 sobre que bits se aceptan sin modificación de los registros latch, y cuales se manipulan.
La forma en la que se manipulan los diferentes bits, se define por el modo de operación, que se determina en el Function-Select-Register. Independientemente del contenido del Enable-Set/Reset Register, en el modo Write 2 se operan los cuatro bits bajos del byte de CPU con los registros latch. Se opera el bit 0 del byte de CPU con todos los bits del registro latch 0, que deben ser manipulados. Lo mismo se aplica a los bits de CPU 1, 2 y 3, que se operan uno a uno con todos los bits de los registros latch 1, 2 y 3.

Este modo sirve especialmente para fijar el color de puntos individuales en los modos gráficos de 16 colores en tarjetas EGA y VGA. Esto también es posible en el modo Write 0, pero la secuencia

en ensamblador en el modo Write 2 es más corta, ya que no se han de programar ni el registro Enable-Set/Reset ni el registro Set/Reset.
El mismo ejemplo sirve de nuevo para el modo Write 2:

```
mov ax,0A000h        ;Dirección de segmento de la RAM de vídeo
mov ds,ax      ;a DS
mov al,ds:[0]  ;Cargar Byte 0 en los registros latch
mov dx,3CEh ;direccionar Graphics-Controller
mov ax,0205h;Read-Mode 0, Write-Mode 2
out dx,ax      ;al Mode-Register
mov ax,0003h;REPLACE-Mode (0) al Function-
out dx,ax      ;Select-Register
mov ax,0408h;Escribir máscara de bits al Bit-Mask-
out dx,ax      ;Register
mov byte ptr ds:[0],0Bh        ;nuevo valor de color a la RAM de vídeo
```

Modo Write 3
Sólo los tres modos Write de la tarjeta EGA ya le procuraban un rompecabezas a los desarrolladores de Software, desde la introducción de la tarjeta VGA se les ha añadido otro misterioso modo, que sólo está disponible en tarjetas VGA. Se activa igual que todos los demás modos, escribiendo su número en el campo de bits correspondiente del Mode-Register (registro de modo) del controlador gráfico.
Durante un modo de acceso de escritura a la RAM de vídeo en este modo primeramente se operan los cuatro bits bajos del Set/Reset Register con los diferentes bits de los cuatro registros latch. Al contrario que en el modo Write 0, el contenido del Enable Set/Reset Register no tiene ninguna importancia, pero el modo de operación se define, como ya acostumbrado, mediante el Function Select-Register.

El byte de CPU se combina en este modo, mediante una operación Y, con el contenido del registro de Bit Mask. El y cuáles resultan de la operación del registro set/reset con los diferentes bits latch. El byte de CPU hace el papel, por lo tanto, que realiza el registro Bit-Mask en el modo Write 0 y 2.
Hasta ahora, debo confesar, no se me ha ocurrido ninguna aplicación realmente práctica de este modo. Por eso no puedo presentarles en este punto ningún ejemplo práctico en ensamblador.

El registro Map-Mask
A continuación, permítanme mencionar un registro que pende, como una espada de Damocles sobre todos los modos Write: el registro Map-Mask de lo que se llama el secuenciador (Sequencer). Con su ayuda se pueden bloquear individualmente los diferentes bit planes de manera que queden protegidos ante accesos de escritura y tampoco se desvele su contenido en los procesos de lectura. Esto tiene sentido cuando, por ejemplo, sólo queremos manipular el contenido de un bit plane determinado.

El estado de los diferentes bit-planes se representa en este registro mediante los cuatro bits más bajos. A cada uno de estos bits le corresponde un bit-plane y se le ha de dar el valor 1 para hacer posible el acceso al bit plane correspondiente.
En los siguientes capítulos mostraremos la manera de utilizar los diferentes modos de lectura y escritura, por ejemplo para colocar o buscar puntos en los diferentes modos gráficos o para copiar áreas de pantalla dentro de la RAM de vídeo o entre la RAM de vídeo y la memoria principal.

Los modos gráficos en 16 colores de las tarjetas EGA y VGA

El logro más importante de la tarjeta EGA son, sin duda, los modos gráficos, con resoluciones que van entre los 320×200 puntos y los 640×350. Los 16 colores de la tarjeta EGA representan un verdadero adelanto frente a los cuatro colores a los que nos veíamos reducidos con la tarjeta CGA. El objetivo final, conseguir reproducciones con calidad de foto, queda todavía muy lejano, pues para eso se necesitan más de 16 millones de colores diferentes. Sin embargo, por algo se empieza.

En este apartado nos ocuparemos de los diferentes modos gráficos de las tarjetas EGA y VGA. Con estas tarjetas se pueden representar 16 colores a la vez. Las preguntas que intentaremos contestar son:
- Por qué diferentes modos gráficos?
- Qué estructura tiene la RAM de vídeo en el modo gráfico de 16 colores?
- Cómo se puede acceder a puntos concretos en estos modos?

¿Por qué diferentes modos gráficos de 16 colores?
Si consideramos la evolución de las tarjetas de vídeo para PC desde el punto de vista de la resolución, en perpetuo aumento, y de la gama de colores, cada vez más amplia, no podemos por menos de preguntarnos por qué la tarjeta EGA ofrece, además del modo de máxima resolución (640 × 350), otros dos modos gráficos de resoluciones más bajas en 16 colores. La respuesta la encontraremos en la tabla siguiente; la tabla nos muestra que una resolución menor requiere también menor espacio en memoria, por página, para los mismos modos. Eso permite la colocación de varias páginas a la vez en la RAM de vídeo, aspecto que es muy importante en según qué aplicaciones. Un buen ejemplo nos lo brinda la programación de sprites, una parte del tema más amplio de la animación que tratamos en el apartado 9.8.9. En este apartado se muestra igualmente la manera de aprovechar el «resto» de la RAM de vídeo para determinadas tareas, es decir, la manera de no dejarlo «en barbecho» forzosamente.

En la siguiente tabla podemos ver, además, que los modos gráficos en 16 colores no son exclusivos de la tarjeta EGA, ya que la tarjeta VGA también los soporta y, además, añade un modo más de resolución 640 × 480 puntos. Si bien este modo está superado por muchos de los modos de la tarjeta super-VGA, dentro de las tarjetas estándar VGA sigue siendo el modo gráfico de mayor resolución y se utiliza en gran cantidad de aplicaciones.
Véase la tabla 75.

Estructura de la RAM de vídeo en el modo gráfico de 16 colores
Si consideramos la tabla precedente con ojos de programador, se nos planteará irremediablemente la pregunta de por qué se mezclan diferentes modos con resoluciones tan diferentes unas de otras. La respuesta la encontramos en la forma en que está estructurada la RAM de vídeo en cada uno de esos modos. En todos los modos mencionados se codifica la información de los colores de cada uno de los puntos de la pantalla según un esquema uniforme, sin tener en cuenta las pequeñas diferencias que se producen a consecuencia de las diferentes resoluciones.

En cada byte de la RAM de vídeo se codifica la información del color para cada ocho puntos (unos junto a otros) de cada línea de puntos, es decir, que cada uno de estos grupos de ocho puntos viene representado por un sólo bit. Pero para la representación de 16 colores diferentes eso no basta, ya que en ese caso se necesitan cuatro bits diferentes para cada punto de la pantalla. Estos cuatro bits se pueden conseguir agrupando cuatro bytes sucesivos de los cuatro bit-planes, tal como nos muestra la ilustración siguiente.

Los ocho primeros puntos gráficos del ángulo superior izquierdo de la pantalla se representan, por ejemplo, mediante los cuatro bytes que se encuentran en los cuatro bit-planes de la dirección de offset 0000h. La información del color para cada uno de los puntos se consigue agrupando en una determinada posición de bit los cuatro bits de cada uno de los cuatro bytes. De este procedimiento

resultan ocho grupos de cuatro, cada uno de los cuales se encarga del color de un punto de la pantalla.

El bit del bit-plane #0 constituye el bit 0 del código de color, el bit del bit-plane #1, el bit 1, y así sucesivamente para los bits 2 y 3 del código de color. Este código se traduce entonces de la misma manera que en las tarjetas CGA: negro (0), azul (1), verde (2), etc.
Hay diferentes modos de lectura (Read) y escritura (Write) que operan con estos grupos de cuatro para hacer que los puntos sean accesibles mediante registros latch en los modos gráficos en 16 colores.

Además de la estructura de cada uno de los bytes, para la programación en el modo gráfico es también muy importante el orden en que estén colocados en la RAM de vídeo. Las tarjetas EGA y VGA funcionan en el modo de 16 colores según un esquema muy simple: partiendo de la dirección de offset de la RAM de vídeo 000h, cada línea de puntos cubre un determinado número de bytes sucesivos. En todos los modos gráficos con una resolución horizontal de 640 puntos eso nos dará un total de, por ejemplo, 80 Bytes; en modo 0D, en cambio, con sus 320 × 200 puntos, sólo obtendremos 40.

Cada una de las líneas de puntos entran en contacto entre sí en la memoria, de manera que la dirección de offset de un punto se puede calcular mediante la fórmula siguiente:

Offset = Y × (resolución horizontal / 8) + int (X / 8)

Hay que señalar que el byte de la dirección de offset calculada de esta manera contiene la información de ocho puntos consecutivos, de manera que hay que aislar cada vez un sólo bit en los cuatro bytes del bit-plane para llegar al código de color del punto deseado. El número de este bit se obtiene con la fórmula:

Bit = 7 - (X mod 8)

Las diferencias entre los diferentes modos gráficos de 16 colores no están basadas, por lo tanto, en la codificación de cada uno de los puntos, sino simplemente en la longitud de cada una de las líneas gráficas en la RAM de vídeo, en su número y, consecuentemente, en la longitud de cada una de las líneas de pantalla. No debemos olvidar este aspecto a la hora de programar estos modos gráficos.
Acceso a los puntos individuales en el modo gráfico de 16 colores
La manera de aplicar estas fórmulas a la práctica y de acceder a puntos concretos con ayuda de los diferentes modos Read y Write vamos a desarrollarla utilizando como ejemplo dos programas, uno de ellos escrito en Pascal y el otro en C. Los programas se llaman V16COLP.PAS y V16COLC.C y, al igual que todos los programas de este libro, se encuentran en los disquetes que los acompañan. No sólo muestran la programación de un modo gráfico determinado en 16 colores, sino que sirven para todos los modos gráficos de 16 colores, dando resultados perfectos en cualquiera de ellos.

Aquí se incluye, tal como veremos en el apartado 9.9, incluso el más sencillo de los modos gráficos de las tarjetas super-VGA. Este modo, con una resolución de 800 × 600 puntos, también trabaja con 16 colores. Pero ya hablaremos de ello en el capítulo mencionado.
A causa de la alta velocidad de ejecución, los dos programas de lenguaje de alto nivel que hemos mencionado vienen soportados, cada uno de ellos, por uno en ensamblador que contiene las rutinas más importantes para el acceso a puntos concretos y para trabajar con diferentes páginas de pantalla. Estos módulos se llaman V16COLCA.ASM, para el programa C, y V16COLPA.ASM, para la versión Pascal.

Puesto que los módulos en los dos lenguajes de alto nivel, así como sus módulos de ensamblador, son, respecto a los nombres de las variantes globales, los nombres de sus rutinas y sus tareas, prácticamente idénticos, podemos limitarnos a una consideración general de los programas sin necesidad de entrar en detalles específicos concernientes al lenguaje.
Empecemos por los módulos en ensamblador, ya que es en ellos donde se realiza verdaderamente el trabajo. En las declaraciones PUBLIC de ambos listados en ensamblador puede leer las rutinas que se ofrecen en ellos como soporte de los programas de lenguaje de alto nivel. Nos encontramos primeramente con cuatro rutinas denominadas INIT640480, INIT640350, INIT640200 e INIT320200. Con su ayuda se puede activar uno de los modos gráficos de 16 colores de las tarjetas EGA y VGA. A estas rutinas se añaden GETPIX y SETPIX, que sirven para poder activar o leer, respectivamente, un punto determinado en el modo gráfico de que se trate. Por último tenemos en los listados en ensamblador las rutinas SETPAGE y SHOWPAGE, con las que podemos, por un lado, definir las páginas de pantalla a las que tendrán acceso GETPIX y SETPIX, y, por otro, hacer visible una página de pantalla determinada.

En el módulo en ensamblador para el programa C encontramos, además, una rutina con el nombre GETFONTPTR, rutina que, en realidad, no tiene nada que ver con la programación gráfica, sino que devuelve un puntero del juego de caracteres de 8 × 8 que queda fijado en los chips ROM en las tarjetas EGA y VGA. Este puntero es necesario en los dos programas de lenguaje de alto nivel para, con ayuda del juego de caracteres apuntado por él, poder llevar a pantalla letras y números también en el modo gráfico. Esto es válido para el C y para el programa en Pascal; aunque también se puede averiguar el puntero necesario desde el programa de lenguajes de alto nivel. No se necesita, por lo tanto, ninguna rutina GETFONTPTR, y es por ello por lo que no se encuentra ninguna en el listado de ensamblador V16COLPA.ASM.

Las rutinas INIT para la inicialización del modo gráfico correspondiente tienen básicamente dos funciones: activar el modo de vídeo que deseemos con ayuda de la función 00h de las interrupciones del BIOS de vídeo y fijar tres variables globales que serán necesarias para el direccionamiento de los puntos individuales y el acceso a las diferentes páginas de pantalla. Al activar la función 00h del BIOS se borra a la vez la pantalla.
Las variables mencionadas llevan los nombres VIO_SEG, ZBREITE y PAGEOFS. La primera de estas variables, VIO_SEG, ocupa la dirección de segmento de la RAM de vídeo, pero representa a la vez la página actual de pantalla, ya que su dirección de offset queda calculada en la dirección de segmento. De esta manera, las rutinas SETPIX y GETPIX, al calcular la dirección de un punto no necesitan tener en cuenta también la dirección inicial de la página actual de pantalla; pueden atenerse simplemente a la dirección de segmento de VIO-SEG, en la que ya se refleja la dirección inicial de la página de pantalla. Veámoslo con un ejemplo:

En el modo gráfico de 640 × 200 puntos cada página de pantalla ocupa 640000 Bytes, repartidos entre los cuatro bit-planes. Por eso la distancia entre las diferentes páginas dentro de un bit-plane es de 8000 bytes. Transformada al sistema hexadecimal esta distancia es de 1F40h bytes.
La primera página de pantalla empieza en la dirección A000:0000, la segunda en A000:1F40, la tercera en A000:3E80, la cuarta en A000:5DC0, etc. Si estas direcciones se añaden a la dirección de segmento, entonces las páginas empiezan con la dirección de offset 0000h. Dicho más exactamente, la primera página empezará en A000:0000, la segunda en A1F4:0000, la tercera en A3E8:0000 y la cuarta en A5DC:0000.

Para que se pueda calcular el inicio de las diferentes páginas de pantalla de acuerdo con cada uno de los modos de vídeo, es necesario tener en cuenta la longitud de una página. Esta información la depositan las rutinas INIT en las variables globales PAGEOFS, dividiendo primeramente el offset entre 16 para poderlo aplicar después directamente a la dirección de segmento.

La variable global ZBREITE no sólo se necesita para calcular el inicio de una página de pantalla sino que es también necesaria para el cálculo de cada una de las direcciones de los puntos. Esta variable encierra el ancho de las líneas de puntos en bytes, es decir, el factor representado en la fórmula anterior por «(resolución horizontal / 8)». En base a las diferentes informaciones de las variables globales VIO_SEG y ZBREITE, las rutinas como SETPIX y SHOWPAGE pueden empezar a desarrollarse con la llamada de una rutina INIT. SETPIX y GETPIX transforman las coordenadas de pantalla dadas por el usuario en una dirección de offset para el acceso a la RAM de vídeo. Para ello se multiplica la coordenada Y por el valor de la variable global ZBREITE, la coordenada X se divide entre ocho, y finalmente se suman ambos resultados.

A continuación, lo que se hace en ambos casos es calcular la posición del bit del punto en cuestión a partir de la coordenada X. Frente al operando MOD, empleado en la fórmula anterior, estas dos rutinas utilizan la orden AND del ensamblador para desactivar todos los bits, excepto los tres más bajos, de la coordenada X. El resultado es el mismo que el de una operación MOD, pero resulta mucho más rápido (viejo truco de la programación gráfica).
A partir de este momento GETPIX y SETPIX se desarrollan de manera completamente distinta, por lo que es necesario considerarlos separadamente. En SETPIX la posición del bit del punto se transforma primeramente en una máscara de bits, la cual necesitaremos más adelante para el acceso al punto en el modo Write 2. Para ello se desplaza a la izquierda el valor 1 en tantas unidades como las del número de la posición de bit. Para la posición de bit 4, por ejemplo, se obtiene el valor 0010000b.

Este valor se carga a continuación en el registro del Bit-Mask del controlador gráfico y, desde aquí, se define para el Mode-Register del modo Write 2 y del modo real 0. Seguidamente se carga la dirección de segmento de la RAM de vídeo, incluida la dirección de offset de la página actual de pantalla, en el registro ES, para poder referirse a la RAM de vídeo mediante él.
Si bien un punto se ha de colocar, y no ser leído, a continuación se carga el byte en el que se encuentra el punto que se quiere direccionar. El valor transmitido en este caso a la CPU no tiene mayor importancia, pues el fin de esta operación no es sino cargar los cuatro registros latch con los cuatro bit-plane-bytes en los que está codificado el color del punto que se quiere direccionar. Este color se direcciona a continuación escribiendo simplemente el código correspondiente como valor de cuatro bits en la RAM de vídeo.

De acuerdo con el modo Write 2, este valor del bit, de la tarjeta EGA o VGA, se escribe en primer lugar en todos los grupos de cuatro bits del registro latch, cuya posición correspondiente en el registro de Bit-Mask contenga un 1. Como aquí eso sólo es el caso en la posición de bit del punto a direccionar, los demás grupos de cuatro no se verán afectados por el acceso de escritura. En las operaciones que automáticamente suceden a continuación, por tanto, sólo cambiarán los cuatro bits que forman el punto de pantalla a direccionar.

Con eso concluye, en realidad, la misión de SETPIX; sin embargo, antes de regresar al usuario, los registros modificados del controlador gráfico se convierten otra vez a su estado por defecto. Este paso es necesario siempre que se hayan manipulado estos registros, para que las rutinas, al ser activadas, puedan contar con que los registros contienen en sí su valor por defecto y sólo hay que modificar aquellos cuyo contenido ha de variar respecto a él.
Lo dicho sirve igualmente para GETPIX, pero teniendo en cuenta que aquí sólo se programa un registro del controlador gráfico, a saber, el registro Read-Map, el cual, sin embargo, volverá automáticamente a su valor por defecto al final de la rutina. Pues para GETPIX no es tan fácil leer un punto de la pantalla como para SETPIX. Los constructores de las tarjetas EGA y VGA parecen haber olvidado el correspondiente modo Read, y por eso tiene que contentarse GETPIX con el modo Read 0.

Pero este modo sólo remite un byte por cada acceso de lectura al vídeo RAM, byte que proviene del bit-plane cuyo número quedará fijado en el registro Read-Map del controlador gráfico. Por eso GETPIX recorre cuatro veces un bucle en el que se lee, cada vez, el byte de uno de los cuatro bit-planes donde se encuentra el punto a direccionar. Sin embargo, antes de leer el byte, GETPIX programa dentro del bucle el registro Read-Map correspondiente.

De esta manera se accede primero al bit-plane 3, después al bit-plane 2, al bit-plane 1 y, finalmente, en el último recorrido por el bucle, al bit-plane 0. Por eso, al final del bucle, el registro Read-Map recobra nuevamente su valor por defecto, a saber, 0.
A partir de los cuatro bytes leídos se forma ya en el mismo bucle el color del punto deseado. Para ello, con ayuda de una máscara de bits formada previamente, se desactivan, en el byte que se lee, todos los bits que no cuentan para el punto deseado. Como resultado obtenemos un byte cada vez en el que o bien hay un bit o ninguno. Este último caso (ningún bit) significa que el bit correspondiente del código del color del punto no estaba en el bit-plane recorrido.

El estado de este bit se averigua ejecutando el comando NEG después de haber desactivado todos los bits que no se necesitan. Este comando, por su naturaleza, hace que el bit 7 del registro pertinente corresponda exactamente al estado del único bit que no se desactivó en la operación precedente. Si el bit de color ya estaba puesto, después de la operación NEG sirve lo dicho hasta ahora también para el bit 7 de ese registro. Dicho de otra manera, el bit 7 se borra después de la operación, aunque el bit de color tenga el valor 0.

El bit 7 conseguido de esta manera se rota seguidamente del registro BH al BL con ayuda del comando ROL. Y, puesto que esta operación se realiza cuatro veces seguidas, al final tenemos en los últimos cuatro bits del registro BL el color del punto deseado, pues cada vez que se recorre el bucle queda desplazado un lugar a la izquierda el resultado que se tenía hasta el momento y colocado en la posición con el valor más bajo.
Por eso es importante recorrer los cuatro bit-planes desde el bit-plane 3 al bit-plane 0, y no en el sentido inverso. De lo contrario, los bits del código de color se verían prácticamente puestos del revés, con lo que se transmitiría una especie de código invertido. Siguiendo el orden correcto, GETPIX le puede transmitir a su invocador el código verdadero del color del punto deseado, aunque la operación sea algo más complicada y más larga que colocar simplemente un punto.

Las rutinas SETPAGE y SHOWPAGE son mucho más simples que SETPIX y GETPIX. Sobre todo en el caso de SETPAGE, con ayuda de la cual se definen las páginas de pantalla en las que operan SETPIX y GETPIX. Para definir la página se multiplica simplemente el número que nos dan de la página por la longitud de una página de la variable PAGEOFS, al resultado se le suma la dirección de segmento A000h y el resultado final se guarda en la variable global VIO_SEG.
Ahora, cada vez que llamemos las rutinas SETPIX y GETPIX, éstas accederán automáticamente a esa página. Hay que puntualizar que, llamando a SETPAGE, la página a la que accede no es automáticamente visible en pantalla.

El motivo es que, de esa manera, con SETPIX y GETPIX tenemos la posibilidad de procesar una página en el transfondo mientras tenemos otra visualizada en pantalla. La página así procesada podemos visualizarla después con SHOWPAGE.
SHOWPAGE transforma el número que le dan de la página en un offset dentro de la RAM de vídeo. Para ello se sirve de la variable PAGEOFS, y tiene que multiplicar por 16 el resultado de la multiplicación de las variables por el número de página, pues los valores en PAGEOFS se refieren a una dirección de segmento y no a una offset, que es lo que se pide aquí.

Para hacer visible en pantalla un página de pantalla se ha de cargar en dos registros del controlador CRT la dirección de inicio de pantalla. En principio se puede cargar cualquier valor en estos

registros, como vimos en el apartado 9.8.3. Para obtener una buena imagen de la página en pantalla es necesario ajustar el arranque de pantalla a las direcciones de arranque en las que SETPIX y GETPIX empiezan a procesar una página de pantalla. Por eso hay que multiplicar aquí también el número de página con PAGEOFS.

Desde las rutinas de ensamblador de los módulos V16COLPA.ASM y V16COLCA.ASM llegamos a los dos programas de lenguaje de alto nivel con los que son llamadas esas rutinas. Ambos programas pueden trabajar en todos los modos gráficos de 16 colores de las tarjetas EGA y VGA; pero siempre hay que definir previamente, antes de compilar el modo, cuál de ellos ha de estar activo.
En la parte de constantes de ambos programas encontramos una constante, en cada uno de ellos, llamada MODUS, a la que se le ha de asignar una de las constantes A320200, A640200, A640350 o A640480. Representan los diferentes modos gráficos.

En el programa principal, teniendo en cuenta la constante definida, con la función ISEGAVKA se comprueba en primer lugar si el modo elegido puede ser inicializado. Para el modo de 640 × 480 puntos eso significa que es imprescindible una tarjeta VGA, mientras que para todos los demás modos basta con una tarjeta EGA. Aunque también se aceptan tarjetas VGA, claro está.
Cuando la tarjeta vídeo ha superado el test, se cargan las variables globales MAXX, MAXY y PAGES teniendo en cuenta el tipo de modo de vídeo. Las variables toman los máximos valores X e Y de pantalla y el número de las páginas de pantalla representables.

Esta información es de gran importancia para la rutina DEMO, rutina que se activa a continuación para demostrar el empleo de las diferentes rutinas del módulo en ensamblador. DEMO recorre con un bucle las diferentes páginas de pantalla, mediante SETPAGE las define como medio de salida (emisor) para SETPIX y GETPIX y las trae a pantalla mediante SHOWPAGE. Después, dentro de ese mismo bucle, llama las rutinas COLORBOX, DRAWAXIS y GRAPRINT o GRAFPRINT (en la versión C), para llenar la pantalla.

COLORBOX, con ayuda de LINE, nos dibuja en pantalla un cuadro lleno de líneas. LINE ha sido realizado en el correspondiente programa de lenguaje de alto nivel y está basado en el conocido algoritmo de Bresenham, algoritmo que permite dibujar líneas sin necesidad de recurrir a complicadas operaciones de coma flotante. Los puntos de la línea se definen mediante SETPIX.
Con este procedimiento, desde luego, la rutina no gana en rapidez; no obstante ya era mi intención el no redactarla en ensamblador. Habría resultado demasiado complicado si hubiéramos querido que la rutina fuera verdaderamente rápida, lo que habría supuesto también una gran cantidad de trabajo en relación con el algoritmo de Bresenham para optimizar el código. Pero todo ese trabajo, en mi opinión, no está en proporción con el resultado, pues una rutina LINE muy raramente viene sola. Cuando se necesita esta rutina para alguna forma de programa gráfico, son necesarias igualmente rutinas para rellenar superficies, para dibujar círculos y polígonos y todo tipo de figuras geométricas.

Si nos dedicáramos a estudiar, con la bibliografía pertinente, el desarrollo y la optimización de estas rutinas, veríamos sin duda pasar los días y las semanas, al final de las cuales habríamos conseguido rutinas válidas para las tarjetas EGA y VGA, pero inservibles para todas las demás. En lugar de desarrollar personalmente esas rutinas, yo aconsejo utilizar rutinas ya preparadas, tal como las que se adquieren por paquetes especiales y las contenidas en los compiladores de la casa Borland y Microsoft. Con su ayuda tendremos acceso a un amplio espectro de rutinas gráficas muy potentes y que además, cargando los controladores adecuados, nos servirán para muchos tipos distintos de tarjetas vídeo.

El carácter de demostración de los programas S16COLP.PAS y S16COLC.C hace que sea suficiente con una rutina LINE en Pascal o en C. LINE no es la única rutina que se basa en SETPIX para la determinación de puntos, GRAFPRINT (o GRAFPRINTC) están en el mismo caso. Para traer letras y números a pantalla lo que hacen es leer en el juego de caracteres ROM 8 × 8 la muestra de puntos de cada uno de los caracteres y escribirlos punto por punto en la RAM de vídeo mediante SETPIX.
Si bien todo esto no resulta especialmente rápido, estas rutinas nos demuestran las capacidades de las diferentes rutinas en ensamblador, dándonos de esta manera, el punto de partida para poder aplicarlas en relación con los modos gráficos de 16 colores de las tarjetas EGA y VGA.

Los modos gráficos VGA con 256 colores
Uno de los mayores logros de los creadores de la tarjeta VGA, frente a los tiempos de la tarjeta EGA, es el modo gráfico ampliado, en el que se pueden representar a la vez 256 colores diferentes sacados de una paleta de más de 262.000 colores toda una novedad en la galería de ancestros del estándar gráfico para PC hasta la fecha.
No obstante, el único modo gráfico estándar de 256 colores sólo es reproducible con una resolución de 320 × 200 puntos, lo que, frente a los modos de 16 colores, con su resolución de 640 × 480 puntos, representa indudablemente un paso atrás. Pero la resolución no es el único criterio a la hora de juzgar la calidad de la imagen. En algunos campos de aplicación la cantidad y variedad de colores es un factor por lo menos tan importante como el anterior, pues, de lo contrario, el modo gráfico de 256 colores de la tarjeta VGA no habría provocado tanto entusiasmo como ha sido el caso.

En este apartado vamos a tratar la cuestión de cómo se puede activar este modo, cómo puede direccionar los puntos e incluso, con sus propios trucos, alcanzar el doble de resolución.
- Activar el modo de 256 colores y direccionar los puntos.
- Cuatro páginas gráficas en lugar de una.
- 320 × 400 puntos y dos páginas gráficas.

Activar el modo de 256 colores y direccionar los puntos
Desde el punto de vista del programador, el modo gráfico de la tarjeta VGA es, entre los diferentes modos gráficos, sin duda el menos complicado. Esto se demuestra, sobre todo, en el direccionamiento de los puntos de pantalla, que se parece más al modo de texto que no a los demás modos gráficos.
Antes de llegar hasta ahí, sin embargo, hay que activar, poner en funcionamiento, el modo; y para eso disponemos de la bien conocida función 00h de la interrupciones de vídeo del BIOS 10h. Como número distintivo de este modo se ha de introducir el valor 13h, junto con el número de función 00h en el registro AH. Esto es todo lo que se ha de hacer para la inicialización del modo, pues con esa misma operación quedan inicializados los modos Write y Read del controlador gráfico, así como los correspondientes registros, de manera que usted ya puede acceder libremente a los 320 × 200 puntos.
Los cuatro bit-planes, tan molestos en el trabajo con los diferentes modos gráficos de 16 colores, ya puede olvidarlos, pues ante usted se despliega la RAM de vídeo en el modo de 256 colores, organizados de manera muy clara. Al igual que en el modo de texto, en la RAM de vídeo los puntos de pantalla también están representados por un byte; este byte indica el color, entre 0 y 255, para el punto. Este valor del color es interpretado por la tarjeta VGA como índice directo en la tabla DAC de colores, de la cual se extraerá el color definitivo para el punto de que se trate.

Puesto que cada punto se representa por un byte, una línea de pantalla en la RAM de vídeo consta exactamente de 320 bytes consecutivos que representan los puntos de una línea de izquierda a derecha. Puesto que las líneas se suceden unas a otras como en el modo de texto, la dirección de offset de un punto resulta muy fácil de calcular con la fórmula siguiente:

offset= y × 320 + x

Desde el nivel del programa todos los puntos se encuentran dentro del segmento de 64 KB, donde se forma la RAM de vídeo en ese modo a partir de la dirección de segmento A000h, y son, por tanto, más fáciles de direccionar.
Por eso no me parece necesario ofrecerles ningún programa de ejemplo para el trabajo con este modo, pues el direccionamiento de los puntos según la fórmula precedente no ofrece ninguna dificultad, como para tener que ilustrarla con un listado de programa. Pero esa no es la única razón para no entrar aquí más en detalle sobre este modo. Debido a la sencilla estructura de la RAM de vídeo, en este modo se han de dejar tres cuartas partes de la RAM de vídeo sin utilizar. Finalmente, para una página de 320 × 200 puntos no se necesitan más que 320 × 200 puntos, es decir, un total de 64000 bytes.

En teoría se podrían utilizar los 3 × 64 KB restantes para la representación de otras tres páginas gráficas, sin embargo esta posibilidad queda impedida por el ordenamiento mismo de la RAM de vídeo. Por qué IBM lo ha permitido, es algo sólo abierto a las especulaciones. Si consideramos ahora la tarjeta VGA desde el punto de vista de IBM, nos vemos obligados a compararla con la tarjeta MCGA, considerada por IBM como la hermana pequeña de VGA. Si bien también dispone del mismo modo a color con 256 tonalidades, sólo posee 64 KB de RAM de vídeo, de modo que sólo puede administrar una página gráfica. Tal vez la idea de IBM era dejar que ambas tarjetas fueran compatibles y, por eso, ha renunciado a la posibilidad de administrar varias páginas con la tarjeta VGA.

En el próximo apartado, no obstante, mostraremos la manera de acceder a las otras tres páginas gráficas con la tarjeta VGA, y un par de trucos, y la manera de traer esas páginas a pantalla.
Cuatro páginas gráficas en lugar de una
La clave para la administración de cuatro páginas gráficas en el modo de 256 colores representa la inteligencia del orden estructurado de toda la información sobre el punto en los diferentes bit-planes. Al igual que en modo de texto, en el modo de 256 colores también es una ilusión la RAM de vídeo continuo en A000h, que es el que tendría que facilitar al programador el direccionamiento de los diferentes puntos de pantalla.

En la realidad, sin embargo, los puntos gráficos se siguen administrando en los cuatro bit-planes, a los que llegan con ayuda del modo denominado Chain4. Este modo supone una ampliación del modo odd/even utilizado por la tarjeta VGA en el modo de texto para llevar la información de la RAM de vídeo hasta los bit-planes 0 y 1.
En este modo el número del bit-plane al que irá a parar el byte indicado, viene determinado, en cada acceso a la RAM de vídeo, por los dos bits más bajos de la dirección de offset indicada. A estos dos bits se les da internamente el valor 0 y sirven así de dirección offset para el acceso al bit-plane seleccionado anteriormente. De esta manera, en los diferentes bit-planes quedan tres bytes sin utilizar por cada byte ocupado.

A diferencia del modo gráfico de 16 colores, ahora la información sobre el color de un punto de palla queda contenida en un byte dentro de un bit-plane, es decir, no queda repartida, como era lo habitual, entre los cuatro bit-planes. Siempre se encuentran cuatro puntos en la misma dirección de offset, pero en diferentes bit-planes.
Si consideramos la primera línea de pantalla vemos que toda la información sobre los cuatro puntos primeros de esta línea se encuentra en la dirección offset 0000h dentro de sus bit-planes. El punto de coordenada X = 0 tiene el bit plane 0, el de x=1, el bit-plane 1, etc. Siguiendo este esquema nos encontraremos los puntos con las coordenadas 4 a 7 en una misma dirección de offset, a saber, en el cuarto byte del correspondiente bit-plane. Pero también aquí están repartidos los puntos entre los cuatro bit-planes de la manera que ya hemos explicado anteriormente.

Puesto que los 64000 bytes necesarios para ocupar los 320 × 200 bytes quedan, de esa manera, repartidos en cuatro bit-planes diferentes, en realidad sólo quedan cubiertos los primeros 16 KB de un bit-plane. Y sin embargo no sobra lugar suficiente para otra página más, pues los bytes quedan repartidos por todo el bit-plane y entre ellos se forman permanentemente «agujeros» de tres bytes.
Si se quiere conseguir tener más de una página en los diferentes bit-planes, es necesario situar los bits en los bit-planes más cerca unos de otros para que no se pierdan tres bytes en torno a cada byte. Para ello es imprescindible reprogramar algunos registros VGA, pero eso es perfectamente posible.

Un pequeño inconveniente es que tenemos que renunciar al modo Chain4, esto es, hemos de escribir a la información sobre el color a mano en los diferentes bit-planes. Con el programa V3220, que he preparado para ustedes, esto no es ningún problema. La denominación algo críptica del programa hace referencia a «VGA 320 × 200 puntos». Puede encontrar el programa en el disquete adjunto en Pascal y versión C, a las que corresponde, respectivamente, un módulo en ensamblador. Los módulos C se denominan V3220C.C y V3220CA.ASM, en tanto que los módulos Pascal se llaman V3220P.PAS y V3220PA.ASM. Los módulos ensamblador contienen diferentes rutinas para la inicialización del modo de vídeo así como para el acceso a puntos concretos, y se han desarrollado en lenguaje de alto nivel atendiendo especialmente a su llamada.
Ambos programas utilizan la rutina init320200, del módulo en ensamblador, para hacer que, con el modo de 320 × 200 puntos con 256 colores, en la RAM de vídeo se puedan utilizar a la vez cuatro páginas gráficas diferentes.
Primero se activa con el BIOS, de la manera habitual, el modo de vídeo 13h, pues sería demasiado trabajoso programar los registros uno a uno. Pero después sí que se realizan algunas modificaciones en diferentes registros para adaptar la estructura de la RAM de vídeo a nuestros deseos.

Para ello se desactiva en primer lugar el modo Chain4 y, seguidamente, el odd/even. Eso tiene por efecto la activación de una especie de modo lineal, de manera que al acceder a la RAM de vídeo ya no se produce ninguna división ni transformación de las direcciones de offset en diferentes bit-planes. Desgraciadamente, esto sólo tiene efecto para los accesos de lectura y escritura en la RAM de vídeo por parte de la CPU. Desde el punto de vista del controlador CRT, responsable de la formación y estructura de la imagen del vídeo, esto no afecta para nada a la estructura de la RAM de vídeo.
Por eso es necesario comunicarle al controlador CRT en el paso siguiente que los bytes de color en los diferentes bit-planes no han de guardar una distancia entre sí de cuatro bytes sino que han de ir uno a continuación del otro. Para ello hay que cambiar de doubleword al modo byte, y es necesario desactivar el modo word. Los registros afectados por la conmutación de modos los obtendrá usted consultando el listado, o también al final de este apartado, donde le ofrecemos una descripción detallada de todos los registros EGA y VGA.
Puesto que ahora en cada bit-plane sólo están cubiertos los primeros 16 KB, ya podemos colocar en la RAM de vídeo tres páginas más de pantalla, depositando una cuarta parte de cada una de ellas en cada uno de los cuatro bit-planes. La primera página de pantalla empieza, dentro de cada bit-plane, en la dirección de offset 0000h, la segunda empieza en la dirección de offset 4000h, la tercera en 8000h y la cuarta, finalmente, en C000h.
Hay que observar, sin embargo, que el direccionamiento de los puntos se vuelve algo más difícil, pues ya no puede acceder a todos los puntos a través de la RAM de vídeo de 64 KB en A000h; sino que primero hay que direccionar cada bit-plane individualmente. Esta tarea la lleva a cabo una rutina del módulo en ensamblador. Su nombre es SETPIX y espera como argumento las coordenadas X e Y del punto considerado, así como su color.

SETPIX forma en primer lugar el offset necesario para el direccionamiento del punto en cuestión. Puesto que en la nueva estructura de la RAM de vídeo las líneas dentro de un bit-plane ya no representan 320 sino sólo 80 puntos, la rutina toma la coordenada Y multiplicada por 80. A

continuación divide la coordenada X entre cuatro, porque siempre se encuentran cuatro puntos consecutivos en la misma dirección de offset de los diferentes bit-planes. La suma de ambas operaciones nos da el offset definitivo para el acceso al bit-plane correspondiente.

El número del bit-plane al que se quiere acceder se deduce igualmente de la coordenada X, a saber, por los dos bits más bajos. Estos definen el número de bits que se ha de desplazar el valor 1 a la izquierda para poder crear la máscara de bits para el registro de Map-Mask. Una vez transmitido el número al registro mencionado, ya es seguro que en el próximo acceso de escritura sólo afectará realmente al bit-plane que deseábamos.
A través de la dirección de offset calculada anteriormente también se puede acceder a la RAM de vídeo y cubrir, de esta manera, el punto deseado con el color indicado. Pero seguramente se estará usted preguntando qué ha sido de la página de pantalla.

La página no queda definida por SETPIX, sino previamente con setpage. Por definición, setpage define todas las páginas de pantalla a las que se accederá cada vez que llamemos en lo sucesivo a SETPIX, hasta que setpage no defina otra página para la salida.

SETPIX retrocede a la página de pantalla definida utilizando el contenido de la variable VIO_SEG como dirección de segmento en el acceso al vídeo RAM. Pues ésta variable (VIO_SEG) toma, al activar setpage, la dirección de segmento de la página correspondiente en la RAM de vídeo, a saber, A000h para la primera, A400h para la segunda, A800h para la tercera, y Ac00h para la cuarta página. Como el arranque de pantalla ya está contenido en la dirección de segmento, no hace falta tenerlo en cuenta al calcular la dirección de offset en SETPIX.

Para poder averiguar el color de los puntos, además de situarlos, se ha incluido en ambos módulos ensamblador una rutina más junto a SETPIX. Esta nueva rutina se llama GETPIX y espera como argumento, al ser activada, una coordenada X y otra Y, y nos da como resultado de la función el color del punto apuntado.
El proceder de esta rutina es igual en gran medida, al comportamiento de SETPIX, pero aquí no es el registro de Map-Mask del Sequencer que se ha de programar para definir el bit-plane al que se quiere acceder. Sino es el Read-Map-Register, que se encuentra en el controlador gráfico, el que resulta de gran importancia, pues es él quien, en un acceso de lectura a la RAM de vídeo en el modo Read 0, define el registro latch y, por ende, el bit-plane cuyo contenido ha de transmitirse a la CPU. Señalemos que, a diferencia del registro Map-Mask, en este registro no se escribe ninguna máscara de bits, sino simplemente el valor del bit-plane que se quiere leer.

Así pues, una vez formada la dirección de offset del punto deseado y consecuentemente programado el registro Read-Map, ya se puede leer el color del punto en la RAM de vídeo. El contenido de la variable VIO_SEG hace a su vez la función de dirección de segmento, de manera que es tenida en cuenta la página de pantalla que tuvimos la última vez que se activó setpage.
Para, además de acceder a diferentes páginas, poder también mostrarlas, la rutina en ensamblador showpage completa el abanico de las diferentes rutinas en el modo de 320 × 200 puntos. El argumento para esta rutina es el número de la página que se quiere mostrar; una vez recibido el número mostrará la página cargando en el registro de inicio del controlador CRT la dirección de offset 0000h, 4000h, 8000h o C000h, según sea la página de que se trate.

En líneas generales, el interés principal de ambos programas es mostrarle a usted cómo se trabaja con las diferentes rutinas en ensamblador. Para ello se cubren las cuatro páginas de pantalla con un patrón de puntos muy parecido en todos los casos, a saber, una cruz de coordenadas, un mensaje de copyright y un objeto que recuerda vagamente la cara posterior de un sobre. Este objeto está formado por muchas líneas distintas que llevan un color continuo entre 0 y n. Este objeto se dibuja mediante la función ColorBox.

La variable n del color toma el valor 16 en la página 0, el valor 64 en la página 1, 128 en la 2 y, finalmente, 256 en la 3. Para dibujar las líneas ColorBox se sirve de una rutina llamada Line, la cual, basada en la rutina SETPIX del módulo en ensamblador, traza las líneas según el conocido algoritmo de Bresenham.
Puesto que las diferentes páginas de pantalla no se diferencian, por lo demás, en nada, esta muestra «como punteada» permite lograr un efecto óptico muy interesante al cambiar rápidamente de una página a otra. Y nuestros dos programas en lenguaje de alto nivel se sirven de este efecto.

320 × 400 puntos y dos páginas gráficas

A pesar de lo impresionantes que resultan los 256 colores del modo en 320 × 200 puntos, este modo, sin embargo, no tiene punto de comparación con los modos vídeo de la tarjeta VGA. En el modo 12h, el modo VGA estándar de máxima resolución, cuenta con casi cinco veces más puntos. Si bien podemos alabar el modo de 320 × 200 puntos, ya que, gracias al elevado número de colores, produce la sensación, como el televisor , de una resolución mucho mayor a la real, en realidad, es natural que deseemos una resolución más elevada que la de este modo.

Tanto más, cuanto que en el último apartado hemos visto que una página en este modo ocupa 16 KB; es decir, que todavía queda espacio para páginas más grandes. A esto hay que añadir que la tarjeta VGA puede representar por pantalla, en este modo, 400 líneas de puntos sin ninguna dificultad, ya que esta tarjeta ni siquiera dispone de un modo de 200 puntos.
Efectivamente, la tarjeta VGA crea 400 líneas de pantalla incluso con el modo de 320 × 200 puntos; si bien sólo aparecen 200 líneas diferentes que se duplican. Pero entonces nada hay más fácil que desconectar la duplicación de líneas y dejar que la tarjeta VGA nos muestre, en lugar de las líneas duplicadas, 400 líneas de puntos en la pantalla. Increíble, pero funciona.

Ni si quiera hace falta reprogramar los registros CRT responsables del timing vertical y horizontal para convertir el modo de 320 × 200 puntos en uno de 320 × 400. La relación de puntos entre los ejes verticales y horizontales resulta un tanto desacostumbrada, pero un resolución dos veces más alta como la que tenemos ahora es aquí el argumento decisivo, tal como muestra la pantalla.
Los 128000 puntos conseguidos con este modo no pueden ser accedidos ya de la manera tradicional, ya que para ello se necesita una RAM de vídeo de una longitud de 128 KB, y el modo Chain4 no llega a tanto. Para no tener que renunciar a la página siguiente, nada más adecuado que emplear directamente el tipo direccionamiento que nos ha proporcionado cuatro páginas en el último apartado. Pero, como en el modo de 320 × 400 puntos cada página necesita 128 KB en lugar de 64, en este modo sólo disponemos de dos páginas, cantidad suficiente, por otra parte, para la gran mayoría de las aplicaciones.

Ahora me gustaría mostrarles la programación gráfica en este modo utilizando para ellos dos programas como ejemplo. Los programas se llaman V3240C.C y V3240P.PAS y han sido realizados a partir de los programas de demostración que conocimos en el apartado anterior. También aquí tenemos dos listados en ensamblador, el V3240CA.ASM y el V3240PA.ASM, que presentan ligeras modificaciones respecto a sus modelos del apartado anterior. Estas modificaciones son precisamente las que ponen de relieve las diferencias entre el modo de 320 × 200 puntos y el de 320 × 400, y que serán explicadas a continuación.

Una rápida ojeada a los listados en ensamblador nos muestra ya la primera diferencia: la rutina init320200 ahora se llama init320400. Aparte de eso, poco ha cambiado en las rutinas: el primer paso, al igual que antes, es inicializar el modo de vídeo mediante el BIOS. A continuación se cambia al direccionamiento lineal de la RAM de vídeo y se renuncia al modo doubleword en favor del modo byte.

Algo que sí es nuevo es el acceso que se realiza a continuación al registro Maximum-Scan-Line del controlador CRT. Aquí se modifican dos datos a la vez: el bit que controla la duplicación de las líneas de puntos, y la altura de los «caracteres». La altura se ha de cambiar de 1 a 0 porque ya no son dos las líneas de puntos que se han de representar en el espacio lineal sino solamente una. Además hay que desactivar el bit de 200 líneas para que no se produzca ninguna duplicación.

Estos dos cambios son todo lo que hace falta para cambiar del modo de 320 × 200 al modo de 320 × 400 puntos. Así de fácil resulta doblar la resolución en pantalla.
Pero esta nueva resolución de pantalla tiene sus repercusiones también sobre las rutinas en ensamblador setpage y showpage, las cuales han de ser adaptadas al nuevo tamaño de la página de pantalla. Para finalizar, la segunda página de pantalla que ahora es también la última ya no empieza en 4000h sino en 8000h.

Tal vez se asombrará al saber que las rutinas SETPIX y GETPIX no necesita ser modificadas. Al fin y al cabo, el ancho de las líneas y su distribución en la RAM de vídeo siguen siendo las mismas. Todo lo que se ha hecho ha sido duplicar su número, pero eso se suple dando la coordenada Y conveniente al nuevo número.
Que la conmutación a este modo funciona realmente, eso se lo demostrarán los dos programas de alto nivel. Al igual que sus predecesores del apartado anterior, ambos dibujan un eje de coordinadas en las dos páginas de pantalla y un cuadro lleno de líneas de colores, y escriben, además, un mensaje de copyright en la RAM de vídeo.

El origen de coordenadas ya le permite ver que ahora hay 400 líneas de puntos, pues el eje vertical se prolonga hasta el punto 399. Por el cambio tan rápido de un página a otra sabrá usted que dispone de más de una (de dos concretamente), a la vez que podrá apreciar el interesante efecto de color que se produce entre ambas.
Y llegados ya al final de este apartado no quiero ocultarles por más tiempo que, además del modo de 320 × 400 puntos, todavía existe otro modo con 256 colores, utilizable con muchas aunque por desgracia no con todas tarjetas VGA. Este modo tiene una resolución de 360 × 480 puntos, lo que significa que reproduce muchos más puntos en pantalla que el modo de 320 × 400. Por desgracia, esta resolución tan alta hace que las páginas de pantalla ocupen más de 128 KB, lo que se traduce en que la RAM de vídeo ya no puede administrar dos páginas a la vez.

En mi modesta opinión, estos dos argumentos son razón suficiente para contentarse con el modo, suficientemente probado, de 320 × 400 puntos, y que, comparado con el modo normal de 256 colores, incrementa el número de puntos en pantalla en más del 100%. No obstante, si lo que desea son más puntos en pantalla a toda costa, entonces puede acudir a la tarjeta Super-VGA y atenerse a su programación, ya que aquí encontrará resoluciones muy por encima de las estándares.

Libre elección de colores
Una diferencia importante entre la generación de las tarjetas EGA/VGA y sus predecesoras viene dada por la capacidad de las primeras de reproducir más de los 16 colores alcanzados por las últimas. Si la tarjeta EGA pone ya 64 colores a disposición del programador, la tarjeta VGA abarca un espectro de más 262000 colores. Aunque no todos los colores representables pueden aparecer a la vez en pantalla; según se trate de texto o de gráfico, dispondremos de 16 o de 256 colores elegidos entre toda la gama.

En este apartado le explicaremos cómo se seleccionan los colores que queremos hacer visibles y cómo se aplican con ayuda del BIOS. Los temas que tocaremos son los siguientes:
- Tareas y programación de los registros de la paleta.
- Tareas y programación de la tabla cromática DAC.
- Representación a color en modo de texto.

- Representación a color en modo gráfico con 16 y 256 colores.
- Definición del color del marco de la pantalla.
- Transformación de tonos grises en la tarjeta VGA.

Tareas de los registros de paleta

Decisivos para la representación a color en todos los modos de texto y los modos gráficos con 16 o menos colores, son, en las tarjetas EGA y VGA, los 16 registros denominados de paleta y que forman parte del controlador de atributos. Una vez ha averiguado el controlador CRT, durante la formación de la pantalla, el color de fondo o de realce de la pantalla de un carácter de texto, o bien el color de un punto gráfico, entonces utiliza este valor entre 0 y 15 como índice en la tabla de los registros de paleta. Entonces extrae el color deseado del registro de paleta con el subíndice y lo transmite al monitor, en las tarjetas EGA, directamente a través de las líneas del cable del monitor.
Con este procedimiento se pone fin a la difícil comunicación característica de todas las tarjetas anteriores, entre los diferentes códigos de color y los colores que aparecen en pantalla. El programador tiene, de esta manera, la posibilidad de seleccionar los colores por sí mismo y, además, la de poder realizar cambios globales de color.
Para ello no será necesario modificar el contenido de la RAM de vídeo. Por ejemplo, si quiere usted cambiar todos los puntos negros del modo gráfico de 640 × 350 puntos por puntos de otro color, basta con que cambie el contenido del registro de paleta 0.

Mientras no se modifique ninguno de los registros de paleta, no se apreciará tampoco ninguna diferencia entre las tarjetas EGA/VGA y la tarjeta CGA. El motivo es simplemente que, en la inicialización de un modo de vídeo a través de la función 00h de la interrupción del BIOS de vídeo, los diferentes registros de paleta quedan cargados con los valores que corresponden a los colores de la tarjeta CGA. Este proceso, sin embargo, se puede cancelar con ayuda de una subfunción de la función 12h para que los registro de paleta (así como los registros de color DAC de la tarjeta VGA) no sufran ninguna modificación durante la inicialización de un modo de vídeo con la función 00h.

Para llamar esta función es preciso cargar primero el número de función 12h en el registro AH y la subfunción 31h en el registro BL. El registro AL, que recoge normalmente los números de las subfunciones, es decisivo para el bloqueo de la inicialización. Si este registro se carga con el valor 1 antes de llamar la función, la inicialización de los registros de paleta no se produce en ninguna de las siguientes llamadas de la función 00h. En cambio, con el valor 0 para el registro AL, se vuelve a poner en marcha ese mecanismo.

Si está activada la inicialización automática de los registros de paleta, para aprovechar exhaustivamente las posibilidades cromáticas de las tarjetas EGA y VGA es necesario programar los registros de paleta. Pero para ello hay que conocer primero la estructura de los diferentes registros. Suponiendo que la tarjeta EGA esté conectada a un monitor EGA o un monitor Multisync, los bits de cada uno de los registros de paleta se corresponden entonces con las líneas de conexión del monitor que sirven para los códigos de color. Los tres colores básicos, rojo, verde y azul (RGB) disponen de dos líneas cada uno de ellos. Una de ellas corresponde a una representación en colores más intensos y la otra en colores algo más suaves. En total son seis los bits que entran en la codificación del color, y con esos seis bits se puede crear un máximo de 64 colores (2 elevado a 6 = 64), si bien sólo 16 de ellos, de entre los 16 registros de paleta, pueden ser representados a la vez en pantalla.

Además de los registros de paleta, para dar color son igualmente importantes los registros Color-Plane-Enable del controlador de atributos. Pues antes de acceder a uno cualquiera de los 16 registros de paleta, el controlador de vídeo de la tarjeta EGA/VGA lleva a cabo una operación lógica Y entre el subíndice del color y los cuatro bits más bajos del registro. Este proceso es, por lo general, bastante transparente, pues el registro Color-Plane-Enable contiene el valor 1111b, como

valor por defecto, en los cuatro bits más bajos. Por eso la operación Y con el índice de color no modifica este valor, y, así, en la pantalla aparece el color deseado de la manera acostumbrada.
Muy al contrario, si modificamos el valor en el registro Color-Plane-Enable y damos el valor 0 a algunos bits determinados de este registro. Pues entonces el subíndice del color sale con un valor modificado de la operación Y con el registro Color-Plane-Enable, de tal modo que los subíndices de color ya no corresponden de la manera habitual con los registros de paleta. Si el valor en el registro Color-Plane-Enable es 0111b, por ejemplo, en lugar de 1111b, entonces se desactiva el subíndice del bit más alto y, como consecuencia, los puntos de la pantalla (o los caracteres) con un código de color entre ocho y 15 se reproducen en los colores de los registros de paleta 0 a 7. Pero en la práctica apenas si se utiliza esta posibilidad, y el valor por defecto en el registro Color-Plane-Enable no sufre alteraciones en toda la ejecución del programa.

Tareas de la tabla de colores DAC
En la tarjeta EGA el contenido de los registros de paleta se transmitía directamente al monitor gracias a las seis líneas del cable del monitor. Pues bien, con la tarjeta VGA eso ya no es posible. Al fin y al cabo, lo que distingue a la VGA estándar de la EGA estándar es precisamente el empleo de señales analógicas, señales que no pueden ser emitidas con el sistema de líneas de la EGA.
Por eso, entre los 16 registros de paleta, existentes también en la tarjeta VGA, y el monitor, se activa la tabla de colores denominada DAC y que consta de 256 registros. Las siglas DAC se han formado a partir de la función de esta tabla, que es, a saber, la de convertir valores digitales para el color en señales análogas de color. En inglés: «Digital to Analog Converter».

Cada uno de los 256 registros de esta tabla representa uno de los 256 colores posibles a elegir entre más de 262.000. Esta imponente cifra es el resultado de la codificación de colores con 18 bits (2 elevado a 18= 262.144), pues cada registro de la tabla de colores DAC consta de tres entradas, cada una de ellas de 6 bits, y que determinan la proporción de rojo, verde y azul de cada uno de los colores.
Para seleccionar un registro de la tabla DAC el controlador de vídeo interpreta el contenido de los diferentes registros de paleta en la tarjeta VGA, no como un código de color sino como un subíndice en la tabla de colores DAC. Tal como muestra la siguiente ilustración, existen todavía otros muchos registros a tener en cuenta, ya que nos ofrecen diferentes posibilidades para dividir los registros DAC en grupos.
El registro Mode-Control del controlador de vídeo tiene una función muy importante. Si este registro contiene el valor 0, entonces el índice para la tabla DAC se forma con los bits 0 a 5 en el correspondiente registro de paleta, y con los bits 2 y 3 del registro Color-Select. La consecuencia práctica de ello es que la tabla de colores DAC queda subdividida en cuatro grupos de 64 registros consecutivos cada uno. El valor en el registro de paleta representa el índice de cada grupo, y el grupo activo se selecciona mediante el contenido de los bits 2 y 3 del registro Color-Select.

Algo diferente sucede cuando el bit 7 tiene el valor 1 en el registro Mode-Control. En este caso la tabla de colores DAC queda subdividida en 16 grupos de 16 registros consecutivos cada uno de ellos. En esta tabla el subíndice se forma con los bits 0 a 3 en el registro de paleta correspondiente y con los bits 0 a 3 en el registro Color-Select. De este modo, este registro selecciona el grupo de color activo dentro de la tabla de colores DAC, y el contenido del registro de paleta representa el subíndice en este grupo.

Una de las aplicaciones de este tipo de codificación es crear una transición de colores rápida y continuada, como, por ejemplo, cuando queremos resaltar en pantalla todo un grupo de caracteres (y no uno solo) mediante una especie de efecto de destello intermitente. Para ello, en los diferentes grupos de la tabla DAC, se han de guardar exclusivamente colores de la misma serie, con intensidad creciente o decreciente, y a continuación cambiar en sucesión muy rápida el grupo de color activo mediante el registro Color-Select.

Siguiendo el mismo principio de la tarjeta EGA en la simulación de colores de la tarjeta CGA, los 16 registros de paleta se inicializan en la tarjeta VGA de tal manera que nos remiten a los 16 primeros registros de la tabla de colores DAC. Estos, a su vez, se activan de manera que reflejen los colores normales en CGA, del negro (0) al blanco (15). Todos los demás registros de la tabla DAC permanecen inactivos al inicializar un modo de texto mediante la función 00h del BIOS de vídeo. Por el contrario, al inicializar un modo gráfico, se inicializan a la vez los 256 registros de la tabla de colores DAC, siempre que la inicialización no haya sido interrumpida mediante la subfunción 31h de la función 12h.

La ilustración 155 nos muestra el esquema que se sigue en la inicialización de los diferentes registros DAC. Para que se pueda hacer también una idea práctica de todo ello, hemos incluido un pequeño programa al final del capítulo que, además de mostrarnos el ajuste de todos los registros DAC en la pantalla, nos permite igualmente realizar modificaciones en los diferentes registros.

Colores en el modo gráfico de 256 colores de la tarjeta VGA

En los modos de texto y gráficos de 16 colores los registros de paleta son el punto de partida para dar color. En el modo gráfico de 256 colores de la tarjeta VGA este sistema ya no tiene sentido, pues harían falta 256 registros diferentes que, sin embargo, no están disponibles. Por eso se evita el rodeo por los registros de paleta y se trata directamente el código de color de cada uno de los puntos gráficos como si fuera un subíndice de la tabla de colores DAC.

Las 256 entradas de esta tabla en el modo gráfico de 256 colores reflejan de manera inmediata los 256 colores que pueden aparecer a la vez en pantalla en este modo. Por eso la programación de los registros de paleta no tiene ningún efecto en este modo.

Ajuste de color mediante el BIOS

El BIOS ampliado EGA/VGA tiene contiene una serie de funciones que nos permite ajustar tanto el contenido de los registros de paleta y DAC como el de otros registros del controlador de atributos. Estas funciones se han añadido como subfunciones de la función 10h de la interrupción del BIOS de vídeo. Al activarlas, por tanto, siempre debe estar presente el número de función 10h en el registro AH y el número de subfunción correspondiente en el registro AL.

Como primera función, la subfunción 00h le ofrece la posibilidad de cargar uno de los 16 registros de paleta con un valor a elegir libremente por usted. Además del número de función en el registro AH y el de la subfunción en el registro AL, es necesario transmitirle a esta función el número de registro de paleta (0 a 15) correspondiente, en el registro BH, y el nuevo valor del color para ese registro en el registro BL.

Como la subfunción no comprueba el número de registro que le transmitimos, con él podemos alcanzar no sólo un registro de paleta sino también el registro denominado Overscan, el cual se enlaza directamente al último registro de paleta. Como este registro nos da el color del marco de la pantalla y, además, el color de fondo en los modos gráficos compatibles CGA, existe una subfunción propia para este fin: la subfunción 01h.

Pero ajustar el color negro como color de fondo no tiene demasiado sentido, por lo menos en los modos de texto de la tarjeta EGA, ya que aquí la representación del texto ocupa prácticamente toda la pantalla y no deja más que dos o tres líneas tramadas para la creación de un marco. Aparte de eso, el contenido del registro Overscan no desempeña ninguna otra función si conectamos un monitor monocromo.

Para activar la función de acceso al Overscan hay que cargar en primer lugar el número de función 10h en el registro AH y el número de subfunción 01h en el registro AL. El registro BH se encarga del color del marco, que quedará cargado al activar la función en el registro Overscan.

Para cargar los registros de paleta en un suspiro, y no uno por uno, y cargar inclusive el registro Overscan, disponemos de la función 02h. Al llamar esta funció n, además de los números de función y subfunción en la pareja de registros ES:DX, hay que transmitir la dirección de una tabla que contiene los valores de los 17 registros (16 registros de paleta más el consecuente registro Overscan) en forma de 17 bytes. Al ejecutar esta función el contenido de la tabla se copia entero en los 17 registros, lo que trae consigo un reajuste momentáneo de colores.

Si bien disponemos de dos funciones para modificar el contenido de los registros de paleta, nos encontramos, por otra parte, con que el EGA-BIOS no contiene ninguna función que nos permita leer los registros. En el caso de la tarjeta EGA eso resulta imposible por principio, pues casi todos los registros de la tarjeta EGA, y no sólo los del controlador de atributos, ya no se desprenden de su contenido una vez escritos. Esto resulta especialmente problemático en los programas TSR, los cuales, al ser activados, les dan a los registros de paleta el estado por defecto, pero, al retomar el programa interrumpido anteriormente, ya no son capaces de devolver los registros de paleta a su estado original.

Estos programas se ayudan a menudo trayendo las subfunciones 00h y 02h de la función 10h a alguna de sus propias rutinas, la cual empieza por «apuntarse» el código del color antes de escribirlo en los registros de paleta. Este procedimiento, sin embargo, no surte efecto cuando el programa accede directamente a los diferentes registros de paleta, es decir, cuando evita las funciones BIOS previstas al efecto. Por eso yo recomiendo recurrir siempre a estas funciones, aunque en el apartado 9.8.10 se explique la manera de operar directamente con los registros de paleta.

La última subfunción de la función 10h en el BIOS ampliado de la EGA nos da el significado del bit 7 en el byte de atributo de un carácter dentro de un modo de texto. Al igual que en las tarjetas CGA y MDA este bit puede representar los caracteres, con las tarjetas EGA/VGA, con un color de fondo intenso o bien haciendo destellar el carácter (definiéndolo previamente). Mientras que en las tarjetas CGA y MDA este bit sólo puede influir en la programación directa del hardware del vídeo, el BIOS de la EGA/VGA pone a nuestra disposición, afortunadamente, la subfunción 03h de la función 10h.

Además de los registros AH y AL con los números de función y subfunción otro elemento de importancia al activar esta subfunción es el contenido del registro BL. El valor 0 define el color de fondo intenso, mientras que el valor 1 define el destello momentáneo de los caracteres en pantalla (el bit 7 se sitúa en el byte de atributo.

Otras subfunciones del BIOS ampliado EGA

La función 10h de la interrupción del BIOS de vídeo en el BIOS de la VGA fue ampliada, frente al BIOS de la EGA, con algunas funciones relacionadas con la tabla de colores DAC y la lectura de los registros de paleta. Para la tarjeta VGA, a diferencia de la EGA, esto ya no representa ningún problema, y se pueden leer muchos otros registros VGA que, en el estándar EGA, todavía eran «Read-Only».

El contenido de un registro de color DAC puede ser modificado con ayuda de la subfunción 10h. Además del número de función y de subfunción, esta función espera recibir en el registro BX el número del registro DAC al que se quiere acceder (0 a 255) y el código del color que se quiere cargar en los registros CH, CL y DH. Cada uno de ellos toma seis bits de los 18 que existen en total en el código de colores. Al igual que otras subfunciones sucesoras, la subfunción 10h espera la componente rojo en el registro DH, la verde en el CH y la azul, finalmente, en el DL. No hay que olvidar que sólo son tenidos en cuenta los bits del 0 al 5.

Los registros DH, DL y CL son los que recogen también, de la función 15h, el contenido de los registros DAC. Para ello sólo es necesario que, antes de llamar la función, estén cargados el registro BH con el número del registro DAC y los registros AH y AL con los números de función y subfunción.
La subfunción 12h nos permite cargar no sólo uno, sino un número cualquiera de registros de color DAC a la vez. Para ello introduzca en el registro BX el número del primer registro DAC a cargar, y en el registro CX el número total de registros que quiere cargar. El nuevo contenido de los registros DAC se transmitirá a un buffer (dirección en ES:DX), y no a los registros del procesador. Dentro de este buffer, cada uno de los registros DAC toma para sí tres bytes consecutivos, representando el primero de ellos la componente de color verde, el segundo la roja y el tercero la azul del código de color.

Como complemento de la subfunción 12h tenemos la subfunción 17h, la cual elabora una lista con el contenido de una serie de registros DAC. También aquí debe introducir el número del primer registro DAC en el registro BX y el número de registros DAC en el CX. El contenido de estos registros es copiado por el BIOS de la VGA en un buffer, que espera recibir las direcciones de segmento y offset en la pareja de registros ES:DX. Su estructura es idéntica a la del buffer de la subfunción 12h. Compruebe, por lo tanto, que cada registro de la tabla de color DAC toma para sí tres bytes (y no uno), y deje disponible un buffer lo suficientemente grande.

La distribución de la tabla DAC y el grupo activo de color pueden ser elegidos con ayuda de la función 13h, la cual, a su vez, dispone de otras dos subfunciones (es decir, subsubfunciones). Si la función recibe el valor 0 en el registro BL, entonces copia el bit 0 del registro BH en el bit 7 del registro Mode-Control del controlador VGA y distribuye la tabla DAC en 4, o 16 grupos fijos. Si, al contrario, el registro BL contiene el valor 1 al activar la subfunción, entonces ésta copia el contenido del registro BH en el registro Color-Select y selecciona, así, el grupo de color activo.

El contenido de estos registros lo obtenemos llamando la subfunción 1Ah. El registro BL, después de llamar la subfunción, obtiene el contenido del bit 7 en el registro Mode-Control, y el BH obtiene el contenido del registro Color-Select.
La subfunción 1Bh sirve para transformar el código de color de la tabla DAC en tonalidades grises. Este cambio resulta conveniente siempre que tengamos que reproducir una imagen en tonos grises en un monitor VGA color, pero no tiene ningún efecto en monitores VGA monocromos, ya que entonces el cambio a tonalidades grises lo realiza el mismo monitor.

La subfunción 1Bh recoge el número del primer registro que se quiere transformar en el registro BX, y el número total de los registros a procesar lo recoge en el registro CX. La transformación, lo que en inglés se llama «gray scale summing», se realiza mediante la ponderación de los colores componentes para obtener un valor entre 0 (negro) y 1 (blanco). De acuerdo con la intensidad de su representación en pantalla, la componente rojo representa un 30% del tono gris, la verde un 59% y la azul un 11%.

Además de la transformación selectiva de los diferentes registros DAC, se puede realizar una transformación global que será practicada automáticamente cada vez que se acceda a la tabla de color DAC mediante el BIOS. Se puede activar con ayuda de la subfunción 33h de la función 12h. Para ello, y de manera excepcional, hay que cargar la subfunción en el registro BL, y no en el AL, antes de llamar la interrupción del BIOS de vídeo; el número de función se introduce, como de costumbre, en el registro AH. En lugar del número de la subfunción, el registro AL recoge el argumento mismo de la función, el cual decide si la transformación de los registros DAC ha de ser automática o no. El valor 0 significa luz verde paral BIOS de la VGA, mientras que el valor 1 impide el proceso.

Además de las funciones de acceso a la tabla de color DAC, en el BIOS de la VGA encontramos otras subfunciones, dentro de la función 10h, que nos permiten la lectura de los registros de paleta. Así, la subfunción 07h nos facilita la lectura del contenido de cualquier registro de paleta. Facilitándole el número del registro de paleta deseado, además del número de función en el registro BL, la subfunción 07h transmite el contenido del registro de paleta al registro BH. De esta manera se puede obtener el contenido, incluso, del registro Overscan (color de marco 16 en el registro de paleta), aunque para esta operación se dispone de una subfunción especial, la subfunción 08h. Esta subfunción, al igual que la 07h, transmite su resultado al registro BH.

Con la subfunción 09h podemos obtener una copia de toda la tabla de paleta, es decir, de los 16 registros de paleta y del registro Overscan. La subfunción escribe el contenido de la tabla en un buffer de 17 bytes. La dirección de segmento del buffer le es transmitida a la subfunción en el registro ES al ser activada, y la dirección de offset en el registro DX.

Programas ejemplo

Más difícil que ajustar los colores, relativamente fácil gracias a las funciones BIOS que hemos visto, es seleccionarlos. La selección de colores es especialmente difícil en las tarjetas VGA, ya que, mientras que en las tarjetas EGA el espectro no abarca más que 64 colores, en las tarjetas VGA este número se ve ampliamente rebasado. Por eso hemos incluido al final del capítulo un programa a modo de ejemplo, el VDAC, que se encuentra en los disquetes que acompañan el libro bajo el nombre de VDACC.C en su versión C y VDACP.PAS en su versión Pascal.

Ambos programas trabajan en el modo gráfico de 256 colores de la tarjeta VGA, con una resolución de 320 × 400 puntos, y funcionan con los módulos listados en ensamblador V3240PA.ASM y V3240CA.ASM que vimos en el apartado 9.8.7. Ambos programas contienen muchas rutinas, entre ellas ISVGA, LINE y GRAFPRINT, que usted ya ha oído nombrar en capítulos anteriores.

Como punto central de estos dos programas encontramos las rutinas SETDAC, GETDAC y DEMO. SETDAC y GETDAC representan puntos de enlace con las subfunciones 12h y 17h de la función 10h; con su ayuda se pueden leer y escribir todos los registros DAC que se desee. DEMO aprovecha a fondo esta posibilidad, pues no se limita a cargar en memoria el contenido de todos los registros DAC sino que le ofrece la posibilidad de considerar y modificar el contenido de los diferentes registros.

Para ello aparecen 256 bloques de color en el centro de la pantalla, ordenados en un cuadrado. Cada uno de estos bloques se compone de puntos de un color determinado, empezando con el valor 0 en el ángulo superior izquierdo. El bloque junto a él tiene el valor 1, su vecino el 2, y así sucesivamente hasta llegar al último bloque de color en el ángulo inferior derecho, cuyos puntos tienen el código de color 255.

Con este orden tenemos visibles en pantalla los 256 colores que el BIOS ajusta automáticamente en los diferentes registros DAC al inicializar un modo gráfico. Sin embargo, para transmitir algo más que una impresión óptica de cada color, la línea de estado en el margen inferior de la pantalla nos indica los valores numéricos del porcentaje de rojo-verde y azul del color. Estos valores se refieren en principio al color situado en el ángulo superior izquierdo, pero, con ayuda del cursor puede usted obtener los valores de todos los demás colores.

Comprobará que, en esta última operación, el campo de colores queda resaltado por un marco blanco que hace las funciones de una especie de cursor. En el color del campo actual aparece, en la parte superior de la pantalla, el mensaje de copyright. Al principio no verá el mensaje, ya que se encuentra usted en un campo de color negro.

Para que el mensaje de copyright correspondiera siempre al del color actual, se le dio el código de color número 255, que es, en principio, el valor del color negro. Pero, cada vez que se mueve el cursor, en el registro DAC del color 255 se copia el color del campo actual, el mensaje de copyright

aparece entonces en ese color y, además, el campo de color aparece en el ángulo inferior derecho del bloque.

Este programa, además de considerar las proporciones de rojo, verde y azul que forman cada uno de los colores, le permite a usted modificar estas proporciones. Para ello disponemos de las teclas R, G, B. Pulsándolas en su forma minúscula se incrementa la intensidad (la proporción) del color correspondiente, lo que notará enseguida por el campo mismo de color, por el mensaje de copyright y por el contenido de la línea de estado. Para disminuir la proporción de uno de los tres colores hay que pulsar la tecla correspondiente pero en su forma mayúscula, es decir, a la vez que la tecla Shift. Para devolver un campo de color a su estado original sólo hay que pulsar la barra espaciadora.

Con la pulsación de <Return> se finaliza la ejecución del programa, se instala otra vez la tabla de colores que teníamos al principio y se vuelve al modo de texto.

Los registros de las tarjetas EGA y VGA

Las tarjetas EGA y VGA están basadas fundamentalmente en cuatro controladores diferentes que se reparten entre sí el trabajo necesario para la generación de las señales de vídeo. Estos controladores son concretamente:

- Controlador CRT
- Controlador de atributos
- Controlador gráfico
- Controlador de secuencias y
- Conversor digital-analógico (DAC), presente sólo en una de las tarjetas VGA.

Las tarjetas EGA y VGA disponen, además, de algunos registros comunes que también serán tratados en este apartado.

En un principio era posible separar ópticamente los diferentes portadores de funciones en la placa de estas tarjetas, porque estaban alojados en piezas o componentes diferentes. Con el tiempo, sin embargo, la integración ha hecho que en una de estas piezas se alojen uno o dos controladores ampliados que toman a su cargo todas las funciones.

A menudo estos controladores han implementado gran cantidad de funciones de modos gráficos que sobrepasan con mucho el estándar EGA/VGA; pero, por motivos de compatibilidad, por lo general se han esforzado en mantener la estructura de registros IBM-EGA e IBM-VGA. Los registros que describiremos aquí se encuentran, exceptuando variaciones mínimas, en todas las tarjetas EGA y VGA que se consiguen en el mercado y las cuales son compatibles con el estándar EGA/VGA. Los registros cuya estructura varía en las tarjetas EGA y VGA y que sólo se encuentran en una de las tarjetas, serán mencionados de manera explícita.

En este apartado hemos omitido todos los registros que quedan fuera del marco del estándar EGA/VGA y que sirven para definir capacidades ampliadas. Por desgracia, más allá de estos estándares no existe ningún otro para los registros de las tarjetas Super-EGA, Super-VGA y otras ampliaciones, incluso los tres modelos de Super-VGA existentes difieren considerablemente en este aspecto.

No obstante, incluso sin esos registros, los interesados encontrarán muchas cosas nuevas en este apartado, pues además de entrar en el funcionamiento interno de las tarjetas EGA y VGA, la descripción de los diferentes registros les descubrirán muchas posibilidades nuevas, ignoradas hasta ahora por el BIOS de EGA y VGA ampliada.

Existe también, naturalmente, toda un serie de registros cuya manipulación arbitraria puede resultar «peligrosa». Nos referimos, sobre todo, a los registros del controlador CRT, los cuales dirigen la creación de las señales de vídeo y sincronización para el retrazado horizontal y vertical del haz de electrones. Los enlaces entre los diferentes registros son muy complejos y, debido a su misma naturaleza, pueden incluso llegar a ocasionar daños en el monitor si no son programados con

estricta profesionalidad. Hay muchos otros bits y registros que usted puede manipular tranquilamente siempre que desee realizar un viaje de exploración por el mundo interior de las tarjetas EGA y VGA. Pero, por favor, si lo hace, no deje de tener en cuenta las diferentes estructuras de algunos registros y bits en las tarjetas EGA y VGA, pues aquí se encontrará con pequeñas diferencias a las que, no obstante, siempre se alude por separado.

Registros generales

Además de los registros especiales de los controladores, las tarjetas EGA y VGA disponen de otros registros que nos dan información sobre el funcionamiento general de estas tarjetas y que, por eso, se llaman «registros generales».

Miscellaneous output (Registros generales, escribir: Puerto 3CCh)

Direcciones de puerto

0= monocromo: 3B4h/3B5h y 3BAh

1= color: 3D4h/3D5h y 3DAh

Acceso CPU a la RAM de vídeo

0= prohibido

1= permitido

Resolución horizontal

00b = 640 puntos horizontales (25.175 MHz)

01b = 720 puntos horizontales (28.322 MHz)

10b = tasa externa dot-clock

11b = reservado (0 MHz)

Reservado

Bit de página para el modo Odd/even

VGA: número de líneas de puntos verticales

00b = reservado

01b = 350 líneas de puntos

10b = 400 líneas de puntos

11b = reservado (0 MHz)

EGA: número de líneas de puntos verticales

00b = reservado

01b = 200 líneas de puntos

10b =350 líneas de puntos

11b = sin definir

0 Para la simulación de tarjetas MDA se puede definir la dirección de puerto de los registros de datos CRT e índice, así como del Input-Status-Register 1, mediante este bit. Mientras que estos registros, normalmente cubren los puertos 3D4h/3D5hy 3DAh, con este bit se pueden cambiar a las direcciones de puerto 3B4h/3B5h y 3BAh.

3+2 Este campo de bits selecciona el generador de frecuencia activo y designa una línea de puntos responsable de la resolución horizontal, ya que la tasa dot-clock y el número de puntos representables están directamente relacionados.

La tasa dot-clock de 0 MHz está reservada, pues sólo pueda entrar en acción durante un reset de la tarjeta VGA.

Cualquier modificación de estos registros debe ir acompañada de un reset sincrónico a través del registro reset del secuenciador.

5 En los modos Odd/even (modos vídeo 0, 1, 2, 3 y 7), este bit hace las funciones de bit de más bajo valor del acceso a memoria y decide, por lo tanto, si dentro de los bit-planes se accederá a direcciones sólo pares o sólo impares. Si el bit contiene el valor 1 (por defecto), se accede a todos

los bytes de direcciones offset pares, mientras que el valor 0 señala la posibilidad de acceder a todas las direcciones impares.

El bit pierde su significado cuando a través del bit 1 del registro 6 del controlador gráfico se activa el modo chain, o bien cuando se activa el modo chain 4 a través del bit 3 del registro 4 del controlador de secuencias.

6+7 Estos dos bits, efectivamente, no seleccionan exactamente la resolución vertical, sino más bien la polaridad de las señales de retrazado horizontales y verticales. En la práctica, sin embargo, la resolución vertical indicada es el resultado de esta polaridad.

Tenga presente que el modo simulador de 200 puntos en las tarjetas VGA no puede ser definido para este campo de bits, pues en realidad se trata de un modo de 400 puntos en el que sólo se pueden representar 200 líneas duplicadas.

Input Status 0 (Registros generales, Puerto 3C2h, sólo lectura)

7 Este bit indica que se está produciendo un retrazado vertical del haz de electrones, siempre que ello comporte una interrupción. Después de este retroceso mantiene el valor 1 hasta que, mediante el bit 4, vuelva a ser situado en el registro vertical-retrace-end del controlador CRT.

Input Status 1 (Registros generales, Puerto 3BAh, sólo lectura)

1+2 En la tarjeta EGA, este bit representa el punto de comunicación con el lápiz óptico, el cual no recibe ningún soporte en las tarjetas VGA. El bit 2 sirve para comprobar la existencia del lápiz óptico; con el bit 1 se puede averiguar si el botón de arranque está apretado momentáneamente en el lápiz óptico. En ese caso se puede averiguar la posición del lápiz óptico con ayuda de los registros 10h y 11h del controlador CRT.

3 El interés de este bit para la mayoría de programas reside en que en él se realiza la lectura del estado del retrazado vertical, es decir, se puede determinar en qué momento se pueden realizar determinadas modificaciones en los registros; modificaciones que, en otro caso, producirían una imagen borrosa en pantalla.

Controlador CRT

El controlador CRT es el responsable autorizado de la formación de pantalla, pues es quien crea las señales para el monitor con ayuda de las cuales se dirige el haz de electrones de los tubos de pantalla. Consecuentemente, encontramos aquí gran cantidad de registros destinados, sobre todo, al timing del retrazado horizontal y vertical del haz de electrones.

Estos registros, por lo general, no resultan de interés para el programador, pues su actividad conjunta es bastante complicada y, por eso, es mejor dejar que sea el BIOS quien los programe, el cual se encargará de realizarlo pertinentemente cuando cambie el modo de vídeo.

La programación de registros como, por ejemplo, los de offset o el Line-compare, ya es mucho más prometedora. Estos registros se utilizan para determinados efectos vídeos, pero no se puede acceder a ellos a través del BIOS. Para empezar, echemos un vistazo a la lista de los 25 registros del controlador CRT.

Véase la tabla 76.

A los registros del controlador CRT se accede mediante un registro índice y un registro de datos, los cuales se encuentran en las direcciones 3D4h y 3D5h, respectivamente, cuando la tarjeta VGA o EGA se hacen funcionar en el modo color. En el modo monocromo estos registros tienen las direcciones 3B4h y 3B5h.

Siguiendo el proceso habitual, antes de acceder a un registro hay que escribir su número en el registro índice. Mediante un acceso de lectura se puede averiguar seguidamente, mediante el registro de datos, el contenido del registro en cuestión. Aunque esto sólo es posible sin restricciones con la tarjeta VGA. En las tarjetas EGA, por el contrario, sólo se pueden leer los dos registros de

lápiz óptico; todos los demás sólo pueden ser escritos. En los accesos de escritura se pueden escribir simultáneamente el registro de índice y el de datos mediante una operación de 16 bits. El valor para el registro índice se ha de transmitir al byte bajo, y el valor para el registro de datos al byte alto.

Una vez escrita la dirección del registro deseado en el registro de índice, no resulta invalidada por ningún acceso de lectura o escritura al registro de datos.
Más bien permite el libre acceso al registro deseado, de modo que en el caso de acceder varias veces consecutivas a un registro CRT, sólo hace falta cargar el número del registro en el registro índice la primera vez. Todos los accesos posteriores al primero se pueden desarrollar directamente a través del registro de datos.

No olvide que los ocho primeros registros del controlador CRT, con tarjetas VGA, sólo pueden ser escritos cuando el bit 7 del registro 11h tiene el valor 0. Pero, debido al BIOS, este bit suele ser cargado primeramente con el valor 1, lo que significa que, de entrada no es posible el acceso a los ocho primeros registros CRT.
Horizontal Total Controlador CRT, Registro 00h
0-7 Aquí se guarda la cantidad total de caracteres que se procesan por línea de pantalla. El término carácter en el caso del modo de texto coincide realmente con un carácter ASCII, pero en el modo gráfico corresponde a un grupo de ocho puntos gráficos. La cantidad total de caracteres se calcula del cociente de la anchura de banda y la tasa de muestreo horizontal, dividido por la cantidad de puntos por carácter.

En el caso de una tarjeta EGA, el valor fijado en este registro se ha de disminuir en 2, en el caso de una tarjeta VGA en 5.
Horizontal Total Controlador CRT, Registro 01h
0-7 Aquí se divide el número total de «caracteres» por el número de puntos por «carácter». En una tarjeta EGA, el valor definido en este registro se ha de disminuir en 2 unidades respecto al número total efectivo, y en una tarjeta VGA se ha de disminuir en 5.
Start Horizontal Blanking Controlador CRT, Registro 02h

0-7 Con el arranque de la señal blanking horizontal ya no se visualizan más caracteres porque el haz de electrones del tubo de pantalla queda desconectado. Como unidad se vuelve a tomar el carácter, y el primer carácter que se visualiza en el borde izquierdo de la pantalla se marca con el número 0.
End Horizontal Blanking Controlador CRT, Registro 03h
0-7 Para el final del blanking horizontal se toma al igual que al inicio, como base la unidad «carácter», y el primer carácter visualizado en el borde izquierdo de la pantalla se cuenta con el número 0. Ya que el final del blanking horizontal siempre se encuentra antes del inicio, aquí no se necesitan 8, sino un máximo de 6 bits para la codificación del final del blanking. El sexto bit se encuentra en el registro End-Horizontal-Retrace, que lleva el número 05h en el controlador de CRT. El valor en este registro se calcula de la suma del valor del Horizontal-Blanking-Register y de la anchura del blanking horizontal en caracteres.

5+6 Puede ser aconsejable, retrasar la visualización en pantalla para que el controlador CRT disponga de tiempo suficiente para cargar un carácter y su atributo desde la RAM de vídeo y, seguidamente, conseguir el patrón de puntos correspondiente mediante el generador de caracteres. Mientras que con las tarjetas EGA todavía se necesita un retraso o desfase (skew) del carácter, con las tarjetas VGA esto, por lo general, ya no resulta necesario.
Start Horizontal Retrace Controlador CRT, Registro 04h

0-7 Mediante este registro se determina el carácter, y después de su procesamiento se comienza con el retrazado del haz de electrones. Con ayuda de este registro se puede centrar horizontalmente la pantalla

End Horizontal Retrace Controlador CRT, Registro 05h

0-4 Aquí se fija el final del retrazado horizontal, y como el final de este retrazado siempre se halla antes del arranque, sólo se necesitan 5 bits para su codificación. La unidad aquí vuelve a ser el «carácter»

5+6 Igual que para el final del blanking horizontal, para el final del retrazado horizontal también se puede definir un skew. Pero éste cambia en cada tarjeta, de manera que es mejor no manipular nunca estos bits.

7 Este bit sirve de ampliación a los bits 0 a 4 en el registro End-Horizontal-Blanking y representa su bit de más alto valor.

Vertical Total Register Controlador CRT, Registro 06h

0-7 En este registro se define el número total de líneas de puntos recorridas en la formación de una pantalla. Además de las líneas de puntos señaladas, cuentan también las líneas que podrían ser recorridas durante el retrazado vertical.

Como el número total de líneas de puntos, tanto en las tarjetas VGA como en las EGA, va mucho más allá de 256, en este registro sólo se retienen los 8 bits más bajos. El octavo bit se encuentra en el registro de rebasamiento, que es el registro que lleva el número 07h. En este registro se retiene un décimo bit, exclusivo para la tarjeta VGA, emplazado en el bit 5.

Frente al valor efectivo, el contenido de este registro siempre debe ser rebajado en 2 unidades, tanto en las tarjetas EGA como en las VGA.

Overflow-Register Controlador CRT, Registro 07h

En las tarjetas EGA y VGA se necesita un registro de rebasamiento (Overflow), pues el número de las líneas de puntos verticales es muy superior a 256 y, por eso, en cada registro implicado en el movimiento vertical del haz de electrones falta, por lo menos, un bit. Y los bits altos que faltan se agrupan, precisamente, en el registro de rebasamiento.

Otros bits de rebasamiento se depositan, en la tarjeta VGA, en el registro Maximum Scan Line, cuyo número de registro es el 09h.

Vertical Pel Panning Controlador CRT, Registro 08h

0-4 Con ayuda de estos bits se consigue verticalmente lo que se llama un desplazamiento lento de pantalla (smooth-scrolling), y que consiste en que la pantalla queda desplazada hacia arriba en un número determinado de puntos.

El valor 0 representa la posición normal, y los valores mayores representan el desplazamiento correspondiente hacia arriba, pues el controlador CRT ya no arranca al formarse la pantalla, con la línea de puntos cero, sino con la que se indique en este registro.

Maximum Scan Line Controlador CRT, Registro 09h

0-4 La altura de los caracteres en el modo texto viene controlada por este bit, y se mide en líneas de punto. El valor anotado aquí debe ser inferior en una unidad a la altura real de la línea.

En modo gráfico este campo de bits contiene normalmente el valor 0. A no ser que esté activo un modo gráfico VGA de 200 líneas de puntos en el que se han de duplicar las líneas para mostrarlas. En ese caso, el registro contiene el valor 1.

5 Si bien este bit no está ocupado en la tarjeta EGA, en la tarjeta VGA toma el bit 9 del registro Vertical-Blank-Start.

6 Igualmente sin ocupar en la tarjeta EGA. En la tarjeta VGA se encuentra aquí el bit 9 del registro Line-compare.

7 Mediante este bit, carente de significado en la EGA, se realiza la duplicación de las líneas de puntos en los modos gráficos de la tarjeta VGA basados, en realidad, en 200 líneas. Gracias a este bit se duplica cada una de las líneas para conseguir la resolución normal en VGA de 400 líneas de puntos.

No olvide que la conmutación de este bit debe producirse al margen de cualquier reprogramación de los diferentes registros implicados en el timing vertical de la formación de pantalla.

Cursor-Start Controlador CRT, Registro 0Ah

0-4 La línea de inicio del cursor sobre el carácter, se cuenta a partir de la línea de puntos 0 y puede moverse en un área entre 0 y 31. Si la línea sobrepasa la altura real de los caracteres, no aparece ningún cursor en pantalla.

Si la línea de inicio del cursor es mayor que la línea final (registro CRT 0Bh), en la tarjeta EGA aparece un cursor dividido en dos, y en la tarjeta VGA no aparece absolutamente ninguno.

5 Sólo con la tarjeta VGA se puede desactivar de manera explícita la creación de un cursor a través de este bit 5. En la tarjeta EGA este bit no se halla cubierto.

Cursor-End Controlador CRT, Registro 0Bh

0-4 En este campo de bits se define la línea final del cursor a través del carácter que se encuentra bajo él. Este valor se puede encontrar en un área comprendida entre 0 y 31, pero no puede sobrepasar la altura de un carácter en las líneas de puntos.

Si la línea final del cursor es más pequeña que la de arranque (registro-CRT 0Ah), en la tarjeta EGA aparece un cursor dividido en dos, y en la VGA no aparece absolutamente ninguno.

Start Adress High Controlador CRT, Registro 0Ch

0-7 En contacto con el registro 0Dh, este registro define la dirección de offset a partir de la cual el controlador CRT toma la información sobre el contenido de la pantalla desde la RAM de vídeo. De esta manera define el arranque de la página actual de pantalla en la RAM de vídeo; el valor definido ha de ser divido entre dos (frente al offset real) en el modo de paridad par/impar (odd/even), y entre cuatro en el modo chain4.

Start Adress Low Controlador CRT, Registro 0Ch

0-7 Aquí se define el byte bajo de la dirección inicial de la página actual de pantalla en la RAM de vídeo. Véase «registro 0Ch».

Cursor Location High Controlador CRT, Registro 0Dh

0-7 Este registro define la posición actual del cursor como offset en la página actual de pantalla. La dirección indicada se ha de dividir entre dos respecto a la dirección real de los caracteres. Mientras que el byte alto de este offset se encuentra en este registro, el byte bajo queda definido en el registro siguiente.

Cursor Location Low Controlador CRT, Registro 0Eh

0-7 Aquí se administra el byte bajo de la posición del cursor en la página actual de pantalla. Para más información ver «registro 0Dh».

Start Vertical Retrace Controlador CRT, Registro 10h

0-7 Con el contenido de este registro se define la línea de puntos a partir de la cual empieza el retrazado vertical del haz de electrones. Pero como las tarjetas EGA y VGA trabajan con más de 256 líneas de puntos, los 8 bits de que disponemos aquí para recibir esa información no son suficientes. Por eso se deposita un noveno bit en el registro de rebasamiento que tomará el registro 07h. Y como las tarjetas VGA necesitan un bit más para la codificación de las líneas de puntos, en el registro de desbordamiento encontraremos un décimo bit.

Vertical Retrace End Controlador CRT, Registro 11h

0-3 Aquí se define la línea de puntos que al alcanzarla se desactiva la señal de sincronización vertical y se forma una nueva pantalla. Como sólo se dispone de 4 bits para recibir esta información, la información se recibe a lo más tardar en la línea de puntos número quince de la pantalla.

4 Al producirse una interrupción vertical, el bit 7 del Input-Status-Register toma el valor 1 para indicar el inicio de un retrazado vertical. El bit de interrupción permanece activo hasta que se vuelve a emitir el valor 1 para este bit; durante ese tiempo el bit impide que se produzca una nueva interrupción vertical.

5 Con este bit se puede provocar la interrupción 2 cada vez que se inicia un retrazado vertical; pero no hay que olvidar que muchas tarjetas VGA (como las IBM para los modelos PC) no pueden producir ninguna interrupción vertical.

6 Sólo las tarjetas VGA permiten aumentar de 3 a 5 el número de ciclos de refresco por línea de puntos. Como entonces se necesita más tiempo para el refresco de la RAM de vídeo, la tarjeta VGA trabaja con una frecuencia de línea menor, lo que también le permite funcionar con monitores que no soportan la frecuencia normal de líneas VGA.

7 Mediante este bit se puede bloquear el acceso a los ocho primeros registros del controlador CRT en las tarjetas VGA. Si el bit está activo, los registros se pueden leer pero no se puede escribir en ellos.

Light Pen Low (sólo EGA. en accesos de lectura) Controlador CRT, Registro 10h

0-7 La tarjeta EGA es la única que utiliza el registro 10h, que ya conocemos, de diferente manera según se trate de un acceso de lectura o de uno de escritura: mientras que en un acceso de escritura toma el inicio del retrazado vertical, en el acceso de lectura envía al byte bajo la posición actual del lápiz óptico. Este valor, al igual que la posición del cursor, refleja la posición offset del carácter sobre el que se encuentra el lápiz óptico; este valor ha sido dividido entre dos.

En las tarjetas VGA que no soportan el lápiz óptico, este registro también nos da en un acceso de lectura, el contenido que acabamos de describir del registro Vertical-Retrace-Start.

Light Pen High (sólo EGA. en accesos de lectura) Controlador CRT, Registro 11h

0-7 En los accesos de lectura este registro envía el byte alto a la posición del lápiz óptico (con tarjetas EGA). Véase para más información la descripción del registro 10h.

End Vertical Display Controlador CRT, Registro 12h

0-7 En este registro se define el número de líneas de puntos con el que termina la formación de la pantalla. Con una tarjeta EGA necesita 9 bits para su codificación, y con una tarjeta VGA 10, y puesto que aquí sólo se pueden definir 8 bits, en el registro de rebasamiento (número 07h) encontraremos los bits que nos faltan.

Offset Register Controlador CRT, Registro 13h

0-7 En este registro se define el offset que el controlador CRT suma a la dirección de offset de la línea anterior cada vez que inicia una línea. Este valor no coincide con el offset efectivo, pues se ha de dividir siempre por un factor que varía según el modo de direccionamiento. En el modo par/impar (odd/even) este valor es el 2, en el chain4 el 4, y en el modo de byte normal el 1.

Normalmente, el valor en este registro corresponde a la longitud de una línea del vídeo RAM, contenido en el modo texto que, internamente, es tratado como modo odd/even; es decir, el valor 80 (por ejemplo), puesto que una línea de texto ocupa 160 bytes y, por lo tanto, 80 words.

No obstante, también se pueden indicar valores más grandes o más pequeños, pero teniéndolo siempre en cuenta al introducir en la RAM de vídeo informaciones de pantalla.

Underline Location Controlador CRT, Registro 14h

0-4 Cuando las tarjetas EGA y VGA trabajan en modo monocromo, tienen la posibilidad de subrayar los caracteres en pantalla. Este campo de bits sirve, precisamente, para definir la línea de puntos en la que se producirá el subrayado.

5 Este bit decide, aunque solamente en las tarjetas VGA, si el contador de direcciones interno ha de ser incrementado cada cuatro pulsaciones del character-clock. Para ello es necesario que este bit tenga el valor 1 y el bit 3 del CRT-Mode-Register (count by two) el valor 0. Si, por el contrario, este campo de bits tiene el valor 1, el modo doubleword es completamente ignorado.

6 Con tarjetas VGA, este bit sirve para activar el modo doubleword. En ese caso se ignora el contenido de los Access-Mode-Bits del CRT-Mode-Register.

En modo doubleword, la dirección del contador de direcciones interno, se desplaza en dos unidades hacia arriba durante la formación de pantalla en el acceso al vídeo RAM, a la vez que los bits 14 y 15 pasan a tomar las posiciones 0 y 1. Siempre que el contador de direcciones sea menor de 4000h, se accede a todos los lugares de memoria entre 0001h y FFFDh cuya dirección módulo cuatro de el valor 0.

Si el contador de direcciones interno alcanza la zona entre 4000h y 7FFFh, se direccionan todos las posiciones de memoria entre 001h y FFFDh, cuya dirección módulo cuatro dé el valor 1. Algo parecido sucede con las áreas entre 8000h y BFFFh, o C000h y FFFFh. En este caso, se accede a los lugares de memoria cuya dirección módulo cuatro dé los valores dos o tres.

Start Vertical Blanking Controlador CRT, Registro 15h

0-7 En este registro se define el número de la última línea de puntos visible aumentado en 1. El correspondiente bit 9 para las tarjetas EGA y VGA se administra en el registro de rebasamiento (registro 07h).

Con las tarjetas VGA se necesita un bit más, el bit décimo. Este bit lo encontraremos en el registro Maximum Scan Line, que lleva el número 09h.

End Vertical Blanking Controlador CRT, Registro 16h

0-7 En este registro se define la primera línea de puntos a partir de la cual se desactiva la señal vertical blanking y, por lo tanto, se inicia una nueva formación de pantalla.

Mode Control Controlador CRT, Registro 17h

0 Si un programa le da a este bit el valor 0, entonces se puede simular la estructura de la RAM de vídeo de una tarjeta CGA a partir de la emulación, en relación con otros registros, del modo CGA de 320 × 200 puntos y 4 colores. En este modo precisamente, la tarjeta CGA divide la memoria de vídeo en dos bloques. Estos bloques empiezan en las direcciones offset 0000h y 2000h de la RAM de vídeo. Mientras que en el primer bloque se definen todas las líneas con números pares, en el segundo se encuentran todas las líneas con numeración impar.

Para simular este procedimiento, lo que hace este bit es transmitir, durante la formación de pantalla, el bit 0 del Row-Scan-Counter interno al bit 13 del asimismo contador de direcciones interno. En este último se definen las direcciones en las que el CRT lee los datos de pantalla desde la RAM de vídeo.

Como el bit 13 tiene el peso 2000h, el CRT oscila permanentemente entre los dos bloques de dirección 0000h y 2000h, ya que el bit 0 del Row-Scan-Counter interno también oscila permanentemente entre 0 y 1. Condición imprescindible para ello es que la altura de los caracteres sea superior en 2 unidades al registro Maximum-Scan-Line, para lo cual es necesario cargar el valor 1 en este registro.

1 Este bit representa una ampliación del bit 0 y se le ha de activar, es decir, poner a 0 siempre que se trate de imitar un modo vídeo desconocido por el cual quede dividido la RAM de vídeo en cuatro bloques. Este es el caso, por ejemplo, de la tarjeta gráfica Hércules, y otros adaptadores similares a la tarjeta CGA, con una resolución de 320 × 400 puntos y 16 colores.

En un modo de este tipo, la RAM de vídeo queda dividida en cuatro bloques. Estos bloques empiezan en las direcciones offset: 0000h, 2000h, 4000h y 6000h. El bloque que contendrá las diferentes líneas se calcula mediante el número de líneas más cuatro. La línea 0 se encuentra en el bloque 1, la segunda línea en el segundo bloque, la tercera en el tercero y la cuarta en el cuarto.

Para reproducir esta ordenación se copia, de manera parecida al procedimiento con el bit 0, el bit 1 del Row-Scan-Counter interno en el bit 14 del contador de direcciones interno. Para ello hay que asegurarse antes de que la altura de los caracteres supera en cuatro unidades al registro Maximum-Scan-Line, y que, por lo tanto, el bit 1 del Row-Scan-Counter puede alcanzar el valor 1 y, con él, el bit 14 del contador de direcciones. En ese caso el valor del contador de direcciones superará los 4000h, lo que significa que se pueden direccionar los bloques dos y tres.

3 Si este bit contiene el valor 0, la dirección interna del contador va avanzando de bit en bit con cada pulsación del character-clock. Si este bit, por el contrario, contiene el valor 1, el contador sólo avanza cada dos pulsaciones del character-clock.
5 Con las tarjetas EGA que sólo poseen 64 KB de RAM de vídeo, este bit ha de tener el valor 0 para que no se produzca ningún rebase en el modo word (véase bit 6). En ese caso el bit 0 de la línea de direcciones no recibiría el bit 15 del contador de direcciones interno sino el bit 13.

6 El modo de byte activo es, por lo general, aquel en que el valor del contador de direcciones interno es aplicado, invariable, a las 16 líneas de dirección que determinan el byte direccionado en la RAM de vídeo.

Si por el contrario, este bit se carga con el valor 0, se pasa automáticamente al modo word. Entonces los bits de dirección del contador de direcciones son desplazados un bit a la izquierda y el bit más alto se coloca en la línea de dirección con el valor más bajo, la A0. Mientras el contador de direcciones no supere el valor 8000h, los bytes pares se direccionan entre 0000h y FFFEh. Cuando el valor sea superior a 8000h, se direccionará uno de los bytes impares entre 0001h y FFFFh.

Line Compare Controlador CRT, Registro 18h
0-7 Mediante este registro la pantalla puede ser repartida entre dos áreas diferentes de la RAM de vídeo. Este registro debe incluir entonces el número de la línea de puntos en la que se produce el paso de la primera a la segunda área. Al dar con esta línea de puntos, el controlador CRT le da nuevamente el valor 0 a la dirección de offset interna para obtener la información de pantalla desde la RAM de vídeo; de este modo el contenido de la pantalla se obtiene del principio de la RAM de vídeo.

Cada vez que se inicia un nuevo recorrido por la pantalla se indica el área de la misma cuya dirección aparece en los registros CRT correspondientes (registros 0Ch y 0Dh).

Puesto que las tarjetas EGA y VGA representan en pantalla más de 256 líneas de puntos, los 8 bits de que se dispone para tomar la información necesaria no son siempre suficientes. Por eso se ha dispuesto en el registro de rebasamiento (registro 07) un noveno bit, y en el registro Maximum-Scan-Line (registro 09) de las tarjetas VGA un bit más, el décimo.

Sequencer Controller

El acceso a los registros del Sequencer Controller se realiza a través de dos registros separados: uno registro de datos y un registro índice. Este último se encuentra en la dirección puerto 3C4h y le sigue inmediatamente el registro de datos en la dirección 3C5h.

A diferencia de los otros controladores de una tarjeta EGA y VGA, el Sequencer Controller no es tan fácil de delimitar, pues las tareas que lleva a cabo son de naturaleza muy diversa. Una de ellas es, por ejemplo, la de ejecutar los accesos de memoria a la RAM de vídeo y desviarlos a los diferentes bit-planes, así como seleccionar las tablas de caracteres activas. Además es el

responsable del refresco de la RAM de vídeo que se produce en cada línea de punto al iniciarse el retrazado horizontal.

El Sequencer Controller dispone en total, en las tarjetas EGA y VGA, de cinco registros que mostramos en la siguiente tabla. Tenga en cuenta que estos registros sólo pueden leerse con una tarjeta VGA, lo que resulta imposible con una tarjeta EGA.
Véase la tabla 77.
0 Este bit tiene normalmente el valor 1, pero puede tomar el valor 0 para provocar un reset en el Sequencer Controller, reset que hará que el Sequencer Controller finalice su trabajo. De esta manera no sólo dejan de producirse señales verticales y horizontales de sincronización (la pantalla se queda oscura, sin imagen), sino que el registro Character-Map-Select se carga con el valor 0 y se desactiva el refresco de la RAM de vídeo. Para que no se pierda el contenido de este último, es necesario darle a este bit, tras 20 ó 30 microsegundos lo más tardar, otra vez el valor 1 para poder así reanimar nuevamente el Sequencer Controller.

Antes de reprogramar los bits 0 y 3 en el Clocking-Mode-Register del Sequencer Controller, y de los bits 2 y 3 en el Miscellaneous-Output-Register, es necesario un reset del Sequencer Controller.
1 Este bit funciona exactamente como el bit 0, con la única diferencia de que, al iniciarse el reset, prescinde de darle al registro Character-Map-Select su valor anterior. De ahí que, para provocar un reset, sea posible conectarlo, como alternativa, al bit 0. En este caso, sin embargo, ambos bits deben tener el valor 1 para que el Sequencer Controller pueda proseguir su trabajo.

Clocking Mode Sequencer Controller, Registro 01h
0 Este bit sirve para definir el número de puntos horizontales que creará el controlador CRT por ciclo de reloj. Mientras que en los modos gráficos y los modos de texto a color de la tarjeta EGA siempre son ocho puntos, en los modos de texto de la tarjeta VGA y en un monitor MDA conectado a una tarjeta EGA, a este bit se le da el valor 1 para crear así 9 puntos por carácter.
3 Dándole a este bit el valor 1, la tasa Dot-Clock queda dividida entre dos. Esta división se produce automáticamente al cambiar el modo de vídeo, con ayuda del BIOS, al modo de 320 × 200 puntos con el que se simula la tarjeta gráfica CGA. En todos los demás modos (incluido el de 320 × 200 puntos con 256 colores), este bit tiene el valor 0.

5 Dándole a este bit el valor 1 se desactiva la producción de señales de vídeo en la tarjeta VGA. La consecuencia es que la pantalla se queda sin imagen (negra) y la CPU puede acceder igualmente, sin restricciones, a la RAM de vídeo.
Map Mask Sequencer Controller, Registro 02h
0-3 Cada uno de estos bits sirve para facilitar o bloquear el acceso a un bit-plane. Esto tiene especial importancia en el caso del acceso a la RAM de vídeo en conexión con los diferentes modos de lectura y escritura (read y write). El bit que corresponda en cada caso decidirá, en un acceso de escritura, si a un byte de un bit-plane se le ha de dar el contenido del correspondiente registro latch o no. De manera parecida, en cada acceso de lectura será el bit quien decida si el contenido del byte de que se trate ha de pasar del bit-plane en que se halle al registro latch correspondiente, o si, por el contrario, el contenido del registro latch se ha de mantener invariable.

Character Map Select Sequencer Controller, Registro 03h
0+1+4 Estos bits nos dan el número de la tabla de caracteres que se utilizará en todos los casos y cuyo bit 3 no está contenido en el byte de atributos. En la tarjeta EGA el bit 4 no desempeña aquí ningún papel, de manera que sólo se puede elegir entre las tablas de caracteres 0 a 3, mientras que con la tarjeta VGA el campo de elección, más amplio, incluye los juegos de caracteres 0 a 7.
2+3+5 Aquí se fija el número de la tabla de caracteres que definirá la imagen de todos los caracteres y cuyo bit 3 se halla en el byte de atributo. En este caso es el bit 5 el que no tiene ninguna

importancia en la tarjeta EGA, aunque en la tarjeta VGA es el bit de más peso del número de juego de caracteres.

Memory Mode Sequencer Controller, Registro 04h

1 Este bit sólo es importante en las tarjetas EGA, pues las tarjetas VGA se entregan con 256 KB de RAM de vídeo. Pero debemos tener en cuenta, por lo que hace a las tarjetas EGA, que, actualmente, ya no se utilizan prácticamente tarjetas con sólo 64 KB.

2 Con este bit se puede definir la división de las direcciones de memoria en pares e impares en los diferentes bit-planes de los modos denominados par-impar (odd/even). En este modo los accesos a las direcciones de memoria pares en la RAM de vídeo se transmiten automáticamente a los bit-planes 0 y 2, mientras que los accesos de memoria a las direcciones impares alcanzan a los bit-planes 1 y 3. En ambos casos se amplia previamente el bit más bajo de la dirección de offset con el page-bit (bit 5 del registro Miscellaneous-Output). Pero también en este caso se puede impedir el acceso a los diferentes bit-planes mediante el registro Map-Select del Sequencer Controller.

Si este bit tiene el valor 1 no se produce ninguna división de las direcciones de memoria, siendo posible en ese caso un procesamiento lineal de los diferentes bit-planes.

Compruebe siempre que el contenido de este bit se corresponda siempre con el bit de paridad (odd/even) del registro 5 del controlador gráfico, incluso cuando este registro parezca mostrar una lógica contraria.

3 Para la tarjeta VGA el modo chain4 es como una ampliación, una especie de doble, del modo odd/even. Se utiliza sobre todo en modos gráficos de 256 colores a fin de crear una RAM de vídeo lineal en A000h, lineal desde el punto de vista del programa, ya que, en realidad, queda repartido entre los 4 bit-planes. Esto último se refiere solamente al acceso a la RAM de vídeo mediante la CPU, pero no ejerce ninguna influencia en el direccionamiento de la RAM de vídeo mediante el controlador CRT.

Al igual que en el modo odd/even, los accesos de memoria a la RAM de vídeo en este modo se transmiten, a uno u otro de los cuatro bit-planes, según sea su dirección, y al hacerlo se desactivan los dos bits inferiores de la dirección de offset, o bien se les da el valor 0; de esta manera sólo se accede a lugares de memoria cuya dirección de offset sea múltiplo de cuatro. El bit-plane al que se accede en cada caso viene determinado por los dos bits de la dirección de offset que han sido desactivados, esto es, por la dirección de offset de módulo con el valor 4.

El acceso a los diferentes bit-planes sólo puede producirse cuando el bit-plane correspondiente ha sido liberado mediante el registro Map-Mask del Sequencer Controller.

Si el modo chain4 está activo, se ignoran los bits que controlan el modo odd/even. Estos bits no cobrarán su validez de nuevo hasta que no se desactive el modo chain4.

Controlador de atributos

El controlador de atributos es el encargado de preparar las señales de color para la pantalla, tanto en tarjetas EGA como VGA. Para ello, administra todos los registros de paleta y algunos otros registros de importancia para la generación de las señales de color.

El acceso a los registros se realiza mediante un registro mixto de direcciones y datos. La dirección puerto de este registro para los accesos de escritura es la 3C0h; la dirección puerto para los accesos de lectura (realizables sólo con una tarjeta VGA) es la 3C1h.

Para acceder a uno de los registros de atributos es necesario escribir previamente el número del registro en cuestión en el puerto 3C0h o en el 3C1h. En un acceso de lectura, el contenido del registro se puede obtener del puerto 3C1h a continuación, mientras que el nuevo contenido del registro, en un acceso de escritura, tiene que ser enviado después al puerto 3C0h para poder alcanzar el registro deseado.

Mientras que en el registro mixto, índice y de datos, con dirección puerto 3C1h, sólo hay que escribir el número del registro al que se quiere acceder para una lectura, su paralelo contiene otros datos decisivos para el estado del byte de atributos. Si este bit toma el valor 0, queda interrumpida la conexión entre el controlador de atributos y el controlador CRT y la pantalla se ve negra (o del color que se haya definido en el registro Overscan del controlador de atributos). Es decir, ya no se produce ninguna señal aislada de color para los caracteres o los puntos en pantalla.

El controlador de atributos, con una tarjeta EGA, tiene a su disposición un total de 20 registros, número al que se añade un registro más en la tarjeta VGA. La estructura de algunos de estos registros difiere de un tipo a otro de tarjeta, de forma que en la tarjeta VGA se ha de tener en cuenta la tabla DAC de colores a la hora de emitir señales de color, mientras que esa misma tabla no existe en la tarjeta EGA.
Los registros, tal como sucedía con el Sequencer Controller, sólo pueden ser leídos con tarjetas VGA; con las tarjetas EGA son, por lo tanto, «write-only».

Véase la tabla 78.
Adress Attribute Controller, Registro de direcciones
0-4 Mediante estos bits se direccionan los registros del controlador de atributos, como ya es habitual, con los otros controladores de las tarjetas EGA y VGA.
5 Con este bit activado el controlador de atributos funciona normalmente. Pero si se desactiva (se borra), el controlador de atributos se separa del controlador CRT y la pantalla aparece momentáneamente negra, o del color que se haya definido en el registro Overscan.

Este representa también el único momento en que se puede acceder desde la CPU a los registros de paleta, para cargarlos o para escribirlos.
Palette Attribute Controller, Registros 00h hasta 0Fh, en EGA
0-5 En la tarjeta EGA los 16 registros de paleta existentes toman un código de color determinado. Este código se forma a partir de 6 bits, los cuales permiten un total de 64 combinaciones diferentes.
Palette Attribute Controller, Registros 00h hasta 0Fh, en VGA
0-5 A diferencia de la tarjeta EGA, en la tarjeta VGA los registros de paleta no obtienen ningún código directo de color, sino simplemente un índice en la tabla DAC de colores. Este índice le permite al controlador de atributos cargar el color de un carácter desde la tabla DAC y enviar al monitor las señales de color correspondientes.

Mode Control Attribute Controller, Registro 10h
0 Con este bit se le comunica al controlador de atributos si el modo activado actualmente es un modo de texto o gráfico.
2 Este bit sólo tiene importancia en los modos de texto donde los caracteres vienen representados con una anchura de 9 puntos. Este bit decide entonces sobre la procedencia del noveno punto, ya que en las tablas de caracteres de la ROM sólo se pueden definir ocho puntos por línea.
Si el valor de este bit es 0, el punto número nueve permanece vacío, de manera que se crea un espacio (columna de un punto) tocando al carácter siguiente. Si este bit, por el contrario, está activado, el proceso depende entonces del carácter. Si se trata de un carácter de marco (código ASCII entre C0h y DFh), el punto noveno se copia del octavo. Pero, para todos los demás caracteres, este noveno punto permanece, por definición, libre.

3 Activando este bit se crea un cursor en el modo texto. En este proceso el cursor aparece en 16 imágenes de pantalla sucesivas y permanece invisible en las 16 siguientes. Según la frecuencia con que se produzcan las imágenes en pantalla, el tiempo requerido para ello oscila entre 1/3 y 1/4 de segundo.

5 Si se divide la pantalla en dos mitades mediante el registro Line-compare, este bit es el encargado de decidir en la tarjeta VGA si el panning vertical repercute en la primera mitad de la pantalla o bien a ambas mitades. Esto último se consigue dándole a este bit el valor 0.

6 Con ayuda de este bit se pone el controlador de atributos en un modo gráfico de 256 colores, en el que la información sobre el color ya no se lee de los registros de paleta sino directamente de los registros de color DAC.
7 En un modo color de 16 colores como máximo, este bit es el encargado de decidir si las señales A4 y A5 para el direccionamiento de los registros de color se han de tomar juntamente como las señales A0 a A3 del registro de color pertinente, o si se han de tomar de los bits 0 y 1 del registro Color-Select del controlador de atributos.

Overscan Attribute Controller, Registro 11h, EGA
0-5 Aquí es donde se fijan, en la tarjeta EGA, los colores para los márgenes de pantalla en el mismo formato utilizado en los registros de paleta.
Overscan Attribute Controller, Registro 11h, EGA
0-7 En la tarjeta VGA los colores del borde se seleccionan de la tabla DAC de colores, pudiéndose acceder mediante este registro a las 256 diferentes entradas.
Color Plane Enable Attribute Controller, Registro 12h

0-3 Estos bits permiten activar o desactivar, a voluntad, los bit-planes con que se transmite la información del color desde la RAM de vídeo hasta el controlador de atributos. Programando convenientemente los registros de paleta, o de color DAC, se pueden excluir de la visualización puntos determinados.
Horizontal Pel Panning Attribute Controller, Registro 13h
0-3h Aquí se fija el contador para el Horizontal-Pel-Panning, el cual define a su vez el número de bits que la parte visible de la pantalla debe desplazarse a la izquierda. Tenga en cuenta en este punto lo referido en el apartado 9.8.3.

Color Select Attribute Controller, Registro 14, sólo VGA
0+1 Estos dos bits se utilizan como bits 4 y 5 del índice DAC siempre que está activo el bit 7 en el Mode Control Register del controlador de atributos. En ese caso sustituyen a los bits 4 y 5 del correspondiente registro de paleta.
2+3 Independientemente de cualquier otro registro o campo de bits, estos dos bits determinan, en todos los modos gráficos con un máximo de 256 colores, los bits 6 y 7 para el índice de la tabla DAC de colores.

Controlador gráfico
El controlador gráfico toma parte en todos los accesos de lectura y escritura a la RAM de vídeo; él es el encargado de supervisar la transmisión de la CPU a la RAM de vídeo, pasando por los registros latch, y viceversa. Por eso entre sus registros no encontramos sólo aquellos que determinan los modos de lectura y escritura sino también aquéllos en los que se recoge la información sobre los parámetros de los diferentes modos.
El acceso a los nueve registros del controlador gráfico puede realizarse mediante un registro índice y otro de datos que se encuentran, respectivamente, en las direcciones puerto 3CEh y 3CFh. Como de costumbre, primero hay que enviar al registro índice el número del registro al que se quiere acceder (3CEh). Así el acceso se desvía al registro deseado, que debe ir a parar después, en un acceso de lectura o escritura, al registro de datos (3CFh). Le recordamos nuevamente que el acceso a los registros de este controlador sólo puede ser de lectura con una tarjeta VGA. Las tarjetas EGA no permiten la lectura de estos registros.

A continuación reproducimos una tabla con los diferentes registros del controlador gráfico:

Véase la tabla 79.

Set/Reset Graphics Controller, Registro 00h

0-3 Estos bits son de relevancia en conexión con el acceso a la RAM de vídeo en el modo write 0. Son ellos quienes definen, para cada bit-plane el valor que se ha de escribir en los 8 bits del plane correspondiente, cuando el origen de los datos transferidos es definido por el registro enable set/reset (registro 02h) como el registro set/reset.

Enable Set/Reset Graphics Controller, Registro 01h

0-3 Con estos bits se determina, trabajando en el modo write 0, si el valor que se inscribe en los 8 bits del byte al que se quiere acceder se extrae del byte CPU o del registro Set/Reset (número de registro 00h).

Color Compare Graphics Controller, Registro 02h

0-3 El contenido de estos bits tiene una importancia decisiva para los accesos de lectura en la RAM de vídeo, en el modo read 1 de las tarjetas EGA y VGA. En este modo se agrupan los cuatro bytes leídos, uno de cada bit-plane de la RAM de vídeo, en ocho grupos distintos de cuatro bits cada uno, representando cada cual el color de un punto. Los ocho códigos de color resultantes de este agrupamiento se comparan entonces uno a uno con el contenido del registro y el resultado de la comparación se transmite a la CPU.

Function Select Graphics Controller, Registro 03h

0-2 Este campo de bits define el número de posiciones (bits), para los accesos de escritura en los modos write 0 y 3, que hay que desplazar a la izquierda el byte CPU antes de ponerlo en contacto con el contenido de los registros latch.

3+4 Este campo de bits define, en los modos write 0 y 2, una operación lógica entre un byte CPU y los datos de los datos de los cuatro registros de cerrojo (latch), que puede ser aplicada antes de escribir los datos en los cuatro bit-planes.

Que esta conexión se produzca o no, es cosa que deciden los cuatro bits del registro de Bit-Mask en correspondencia, cada uno de ellos, con un registro latch. La conexión sólo se producirá cuando el bit correspondiente tenga el valor 1, y en ese caso la naturaleza de la conexión vendrá determinada por este campo de bits.

Read Map Select Graphics Controller, Registro 04h

0+1 En estos dos bits se fija el número del bit-plane y, por tanto, el del registro latch cuyo contenido debe ser transmitido a la CPU, para el acceso de escritura en el modo read 0. Esto no afecta en absoluto ni a los accesos de lectura en el modo read 1 ni a los accesos de lectura en el modo chain4, el cual puede ser activado mediante el bit 3 del registro Memory Mode del Sequencer.

Graphics Mode Graphics Controller, Registro 05h

0+1 Aquí se define el número del modo write que interviene en los accesos de escritura a la RAM de vídeo, independientemente del modo de vídeo actual.

2 En este bit se define el número del modo de vídeo actual.

3 También se le ha de comunicar al controlador gráfico si hay activado un modo odd/even o si se ha de realizar un direccionamiento lineal de los bit-planes. Para ello es necesario cargar este bit con el mismo estado definido en el bit 2 del registro Memory Mode del Sequencer. Aquí es preciso tener en cuenta que estos dos bits son de polaridad opuesta.

5 En este bit se define, especialmente para la tarjeta VGA, si el modo color activo es de 256 colores o no, ya que, según sea el caso, el controlador gráfico deberá transmitir la información sobre el color al controlador de atributos de una manera o de otra.

Miscellaneous Graphics Controller, Registro 06h

0 Aquí se ha de definir si el modo activo es un modo texto o gráfico. De esta manera el controlador gráfico no tendrá que recurrir a los registros latch internos encargados de convertir el código ASCII en muestras de puntos en el modo texto.

1 Este bit sirve para definir, nuevamente, si actualmente hay activado un modo odd/even o si los bit-planes se pueden direccionar linealmente.
2+3 Este campo de bits decide en qué lugar del área de direccionamiento del procesador hay que activar la RAM de vídeo. Este campo de bits es puesto en funcionamiento por el BIOS cuando se activa un modo de vídeo concreto, teniendo en cuenta asimismo el tipo de monitor conectado.
Color Don't Care Graphics Controller, Registro 07h

0-3 Estos bits ejercen una influencia decisiva en el procedimiento que sigue la tarjeta vídeo en los accesos de lectura a la RAM de vídeo en el modo read 1. Aquí son estos bits los que deciden qué bit-planes se han de tener en cuenta para la comparación del color y cuáles se han de ignorar.
Bit Mask Graphics Controller, Registro 08h
0-7 En el modo write 0 y 2 son los bits quienes deciden, en el acceso de escritura a la RAM de vídeo, qué bits de los registros latch han de ser escritos sin variación en el bit-plane correspondientes y cuáles se han de operar previamente con una operación lógica con otros datos (de la CPU o del registro Set/Reset) mediante el registro Function-Select. Esto es así, en los cuatro registros latch, para todos aquellos bits cuyo bit correspondiente esté activado en este registro.

Digital to Analog Converter (DAC)
Sólo las tarjetas VGA disponen de un DAC, esto es, de un convertidor que transforma las señales de color digitales en señales análogas para el monitor. El DAC representa el último nivel en la transformación del color y en las tarjetas VGA viene a continuación de los registros de paleta, los cuales, a su vez, remiten a uno de los 256 registros de color que contiene el DAC.
Estos registros de color constan de 18 bits cada uno: 6 bits para cada uno de los colores básicos rojo, verde y azul. En total se dispone de 262.144 colores (256 KB).

El acceso a los registros de color DAC se realiza mediante otros registros (que presentamos en la tabla siguiente). A diferencia de otros controladores existentes en las tarjetas EGA y VGA, para entrar en estos registros de acceso no se utilizan registros de datos o índice sino que cada uno tiene una dirección puerto individual.
Véase la tabla 80.
Para escribir en un registro DAC hay que introducir primero su número, entre 0 y 255, en el registro Pel-Write-Address. A continuación se ha de enviar al registro Pel-Data el color nuevo para el registro DAC en cuestión. Como este registro sólo tiene una anchura de 8 bits, y por lo tanto no puede asumir de una vez el código de color de 18 bits, hay que enviar este último por partes al registro pel-data. En primer lugar se envían los 6 bits del color rojo, a continuación los 6 bits del verde y, finalmente, los otros 6 del color azul.

Dentro de un programa lo normal es que, además de cargar de nuevo un registro DAC, tengamos que acceder a toda la paleta, con sus 256 registros, o a una gran parte de ellos. Por eso el valor del registro Pel-Write-Address sufre automáticamente un incremento una vez se han enviado las tres componentes de color al registro Pel-Data. De esta manera las tres componentes de rojo, verde y azul para el próximo registro DAC pueden ser cargadas directamente en el registro pel-data sin necesidad de acceder de nuevo al registro Pel-Write-Address.

Este ciclo no se cierra hasta que no se introduce un nuevo número de registro en el registro Pel-Address-Write y se carga, así, un nuevo registro. Por cierto, al registro Pel-Address podemos acceder en todo momento para leer en él el número del registro DAC que se está procesando.

Lo dicho anteriormente significa que podemos acceder a los registros DAC no sólo para escribirlos sino también para leerlos. Para ello es necesario enviar primero al registro Pel-Read-Address el número del primer registro que queremos leer. Seguidamente ya se pueden leer en orden sucesivo, mediante el registro Pel-Data, las componentes de rojo, verde y azul en el registro deseado. Al realizar estos tres accesos de lectura es importante dejar que el registro pel-data haga una pequeña pausa entre uno y otro. La mejor manera de realizar esta pausa, dentro de un programa ensamblador, es saltando al comando inmediatamente a continuación.

También en este caso se incrementa el contenido del registro Pel-Adress una vez finalizada la lectura; de esa manera se puede leer inmediatamente el siguiente registro de color DAC.
Durante la lectura y escritura de los diferentes registros DAC, el controlador de atributos no tiene ningún acceso a ellos; por eso, es mejor desconectar el monitor para evitar que se produzca una imagen borrosa (con nieve) en pantalla.
A continuación describimos nuevamente los registros DAC.

Pel Write Access DAC, Puerto 3C8h, sólo VGA
0-7 En este registro se define el número del registro DAC en el que se ha de escribir mediante un acceso al registro pel-data. Este valor se incrementa automáticamente después de cada proceso de escritura para que se pueda escribir en el siguiente registro de color DAC.
Pel Read Access DAC, Puerto 3C7h, sólo VGA
0-7 Aquí se guarda el número del registro DAC cuyo contenido ha de ser leído en el siguiente acceso de lectura al registro pel-data. Este nú mero de registro queda incrementado automáticamente tras cada proceso de lectura para que se pueda leer el siguiente registro DAC.

DAC State DAC, Puerto 3C7h, sólo VGA
1+2 Aquí se dirime si el DAC se encuentra actualmente en modo de escritura o en modo de lectura.
Pel Data DAC, Puerto 3C9h, sólo VGA
Durante la lectura de uno de los registros DAC es en este registro donde hay que escribir la componente de color rojo primeramente, a continuación la verde y, finalmente, la azul, siendo siempre los 6 bits más bajos los únicos que llegan efectivamente al registro DAC.
En la lectura de un registro de color DAC, este registro prepara siempre en primer lugar la componente roja del color, después la verde y seguidamente la azul.

Pel Mask DAC, Puerto 3C9h, sólo VGA
0-7 Si el controlador de atributos quiere leer, durante la creación de un color, el contenido de un registro DAC mientras se forma la pantalla, entonces el contenido de este registro se operará mediante un Y lógico con el número del registro en cuestión. De esta manera se pueden reproducir, en el nivel más bajo, grupos enteros de colores en un registro DAC.

El valor de este registro es normalmente el FFH, porque así se pueden aceptar sin modificación todos los accesos a los registros DAC