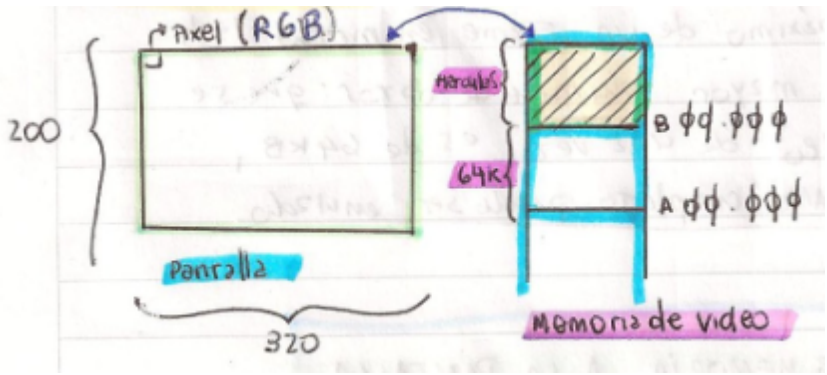
Tarjeta de video - Modo gráfico

- 320x200: 2 colores / 4 colores / 16 colores / 256 colores
- 640x480: 2 colores / 4 colores / 16 colores

Modo 320x200 = 256 colores - Modo 13h / Interrupción 10h - VGA



En la pantalla tenemos 64000 puntos distribuidos en 200 filas con 320 puntos por fila. Cada pixel corresponde con una posición de memoria (1 byte). Se crea una imagen de lo que se quiere reproducir en la memoria y eso se reproduce automáticamente bit a bit, es decir, habrá una relación 1 a 1 donde cada pixel corresponde con una posición de memoria (mapeo 1-1).

Según Wikipedia:

El funcionamiento de la interrupción 13h de video, es el modo de video gráfico más utilizado para los videojuegos de DOS y algunos de Linux debido a su facilidad de uso y excelentes tiempos de respuesta.

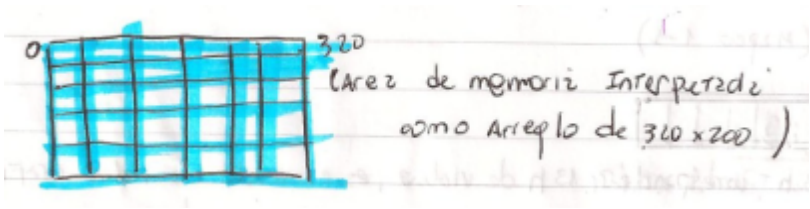
El modo 13h es un estándar de las placas VGA que sobresale del resto por sus tiempos de respuesta. Posee una resolución de 320x200 píxeles y una paleta de colores de 8 bits.

Sabemos que las placas de video poseen un área de memoria virtual (000A:0000 a 000A:FFFF) la cual utilizan para recibir los pixeles que serán vistos en la pantalla. Este área de memoria posee un tamaño de memoria de 64 bytes, lo que indica que el mayor volumen de datos que se puede enviar a la placa de video de una vez es de 64 kb.

Ahora podemos explicar el porqué de los tiempos de respuesta: como dijimos, el modo 13h posee una resolución de 320x200 píxeles con una paleta de 8 bits (1 byte), de aquí podemos averiguar el tamaño máximo de un frame para el modo 13h que resulta ser (320x200x1) = 64000 bytes o 64 kB. Sabiendo que el tamaño máximo de un frame en modo 13h es de 625 kB y que el mayor volumen de datos que se puede enviar a la placa de video de una vez es de 64 kB, podemos decir que *el frame completo puede ser enviado a la placa de una sola vez*.

Mapeo del área virtual de memoria a la pantalla

Como sabemos, la memoria de video ocupa 64kB, cuando la placa se encuentra trabajando en el modo 13h, interpreta los datos que se encuentran en dicha zona de memoria como una matriz de 320x200:

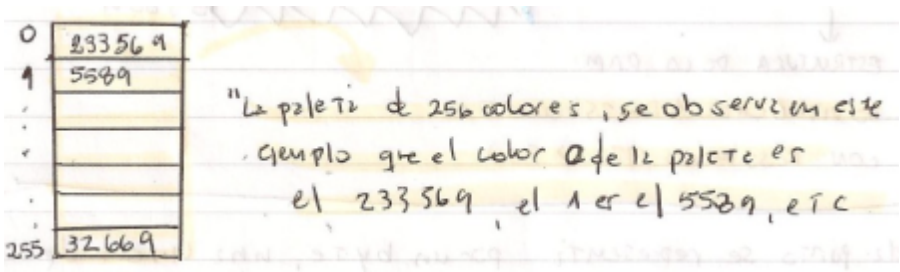


Luego, habiendo interpretado el área de memoria como una matriz se puede dibujar sobre la pantalla haciendo corresponder las posiciones en la matriz con los pixeles del monitor.

Paleta de colores

Este modo posee una paleta de colores de 8 bits, lo que brinda un máximo de 256 colores simultáneos. Los colores que se permiten elegir para la paleta son colores de 18 bits (lo que brinda un máximo de 262144 colores), es decir, este modo permite elegir 256 dentro de 262144 colores.

Los colores de 8 bits de la paleta poseen una representación numérica (de 0 a 255) y la placa luego los traduce a colores de 18 bits mediante el acceso a una tabla donde posee almacenada la paleta de colores. En dicha tabla se almacenan los colores secuencialmente (el color 0 en la posición 0, el 1 en la posición 1, etc):

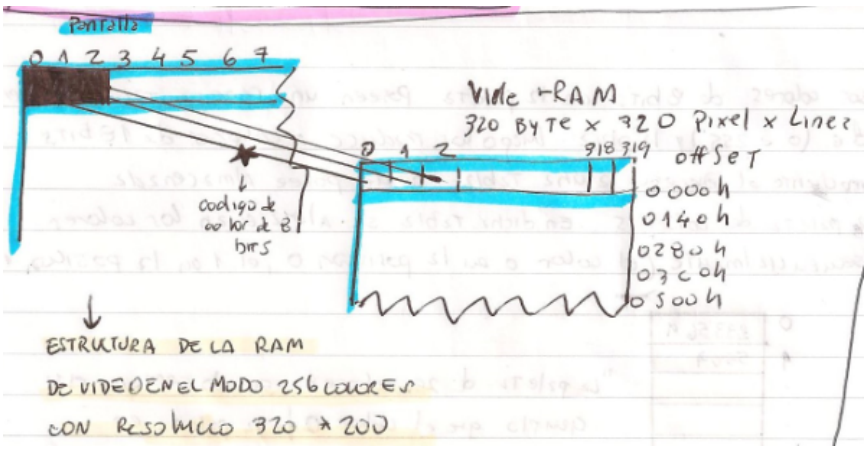


Los colores de 18 bits que se obtienen mediante el acceso a la paleta, representan valores RGB (Red, Green, Blue) de los 18 bits, 6 representan los rojos, 6 los verdes y 6 los azules, con esta representación, un color cualquiera se puede representar en sus 3 componentes (R, G, B) para poder ser utilizado para dibujar un pixel en pantalla.

El refresco del modo

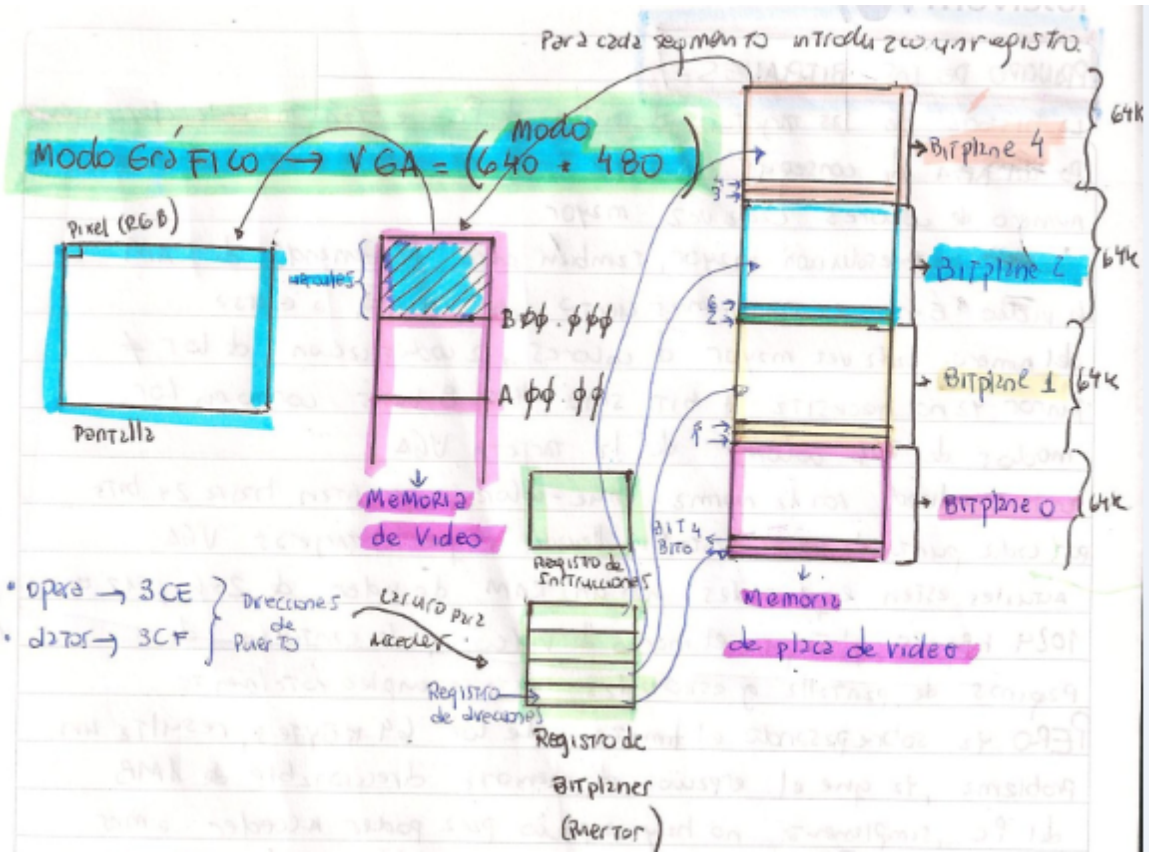
Este modo posee una tasa de refresco de 60hz. Esto significa que el modo de video hace que el monitor se refresque o redibuje la imagen 60 veces por segundo.

320x200 (Segun PC Interno)



Para cada punto se representa por un byte, una línea de pantalla en la RAM de video consta exactamente de 320 bytes consecutivos que representan los puntos de una línea de izquierda a derecha.
Dado que las líneas se suceden una tras otras, la dirección offset de un punto resulta muy fácil de calcular: $offset = y \cdot 320 + x$
Desde el nivel del programa, todos los puntos se encuentran dentro del segmento de 64 kB, donde se forma la RAM de video en ese modo, a partir de la dirección de segmento A000h y son por tanto más fáciles de direccionar debido a la sencilla estructura de la RAM de video, en este modo se han de dejar 3/4 partes de la ram de video sin utilizar. Entonces para una página de 320x200 puntos no se necesitan más de 320x200 puntos, es decir un total de 64000 bytes.

Modo gráfico - 640x480



(Giovanini en clases)
¿Cómo acceso a 256kB sabiendo que puedo direccionar físicamente sólo 64 kB?
¿Cómo organizo la memoria física para accederla desde el punto de vista lógico y pueda encajar los 4 bloques de 64 kB en uno de 64 kB?
Lo primero que se hizo es dividir los 256 kB en 4 bloques de 64 kB, cada bloque se llama BitPlane.
¿Cómo accedo a cada BitPlane? (no tiene acceso directo como en modo texto)
Entonces por cada BitPlane yo tengo un registro, entonces cuando quiero hacer una operación sobre el BitPlane no la hago SOBRE el BitPlane, sino que la hago *SOBRE LOS REGISTROS*, desde el punto de vista de la BIOS si yo escribo en el primer registro, esa información va al primer BitPlane. Por ejemplo, el primer pixel está en el BitPlane 0, el 2do

pixel está en el BitPlane 1, etc. La posición en el BitPlane se redefine con un registro de direcciones.

Principio de los BitPlanes

La historia de las tarjetas de video 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 video. Esto es tanto más cierto, en cuanto, a causa del número cada vez mayor de colores, la codificación de los distintos puntos ya no necesita 1 bit sino 4 u 8 bits como en los modos de 256 colores de la tarjeta 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 VGA actuales esten equipadas con una RAM de video de 256, 512 o 1024 kB, que, según el modo de video y la cantidad de páginas de pantalla gestionadas, hasta se emplea totalmente, PERO ya sobrepasando el límite de los 64 kB, resulta un problema, ya que el espacio de memoria direccionable del PC es de 1MB, simplemente no hay espacio para poder acceder a más RAM de video. Por ello, a continuación describiremos como "se encaja" la RAM de video de 256 kB de la tarjeta VGA en la zona direccionable del PC.

La división de la RAM de video en BitPlanes

Para las tarjetas de video, el PC solo 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 video de tarjetas MDA (6A y Hercules). Para las tarjetas VGA solo queda el segmento A, entonces el problema es *cómo administrar 256 kB de RAM de video a través de una ventana de sólo 64 kB?*.

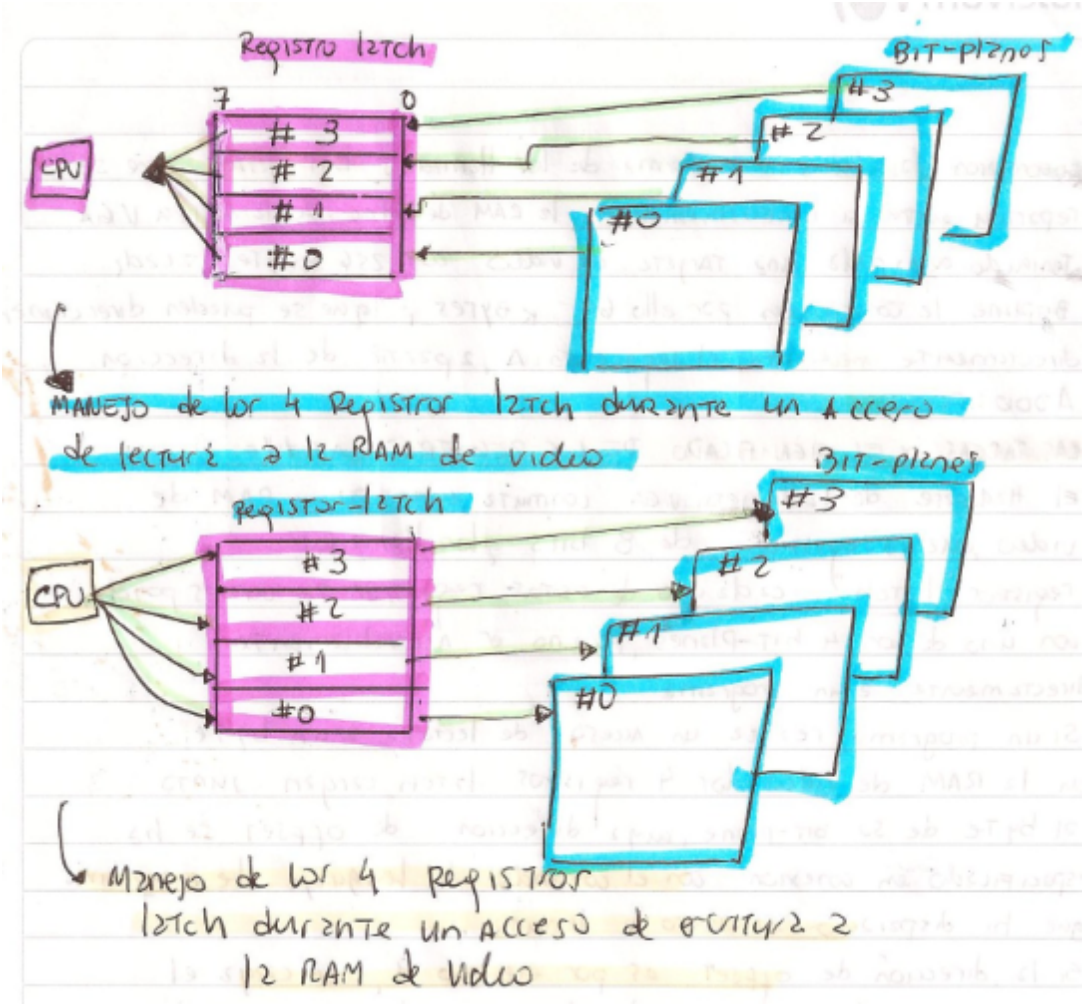
Solución:

Encontraron la solución en forma de los llamados BitPlanes, que se reparten entre si uniformemente en la RAM de video de la tarjeta VGA.
Teniendo equipada una tarjeta de video con 256 kB, a cada BitPlane le corresponde por ello 64 kB, que se pueden direccionar directamente mediante el segmento A, a partir de la dirección A0000:0000.

Las tareas y el significado de los registros Latch

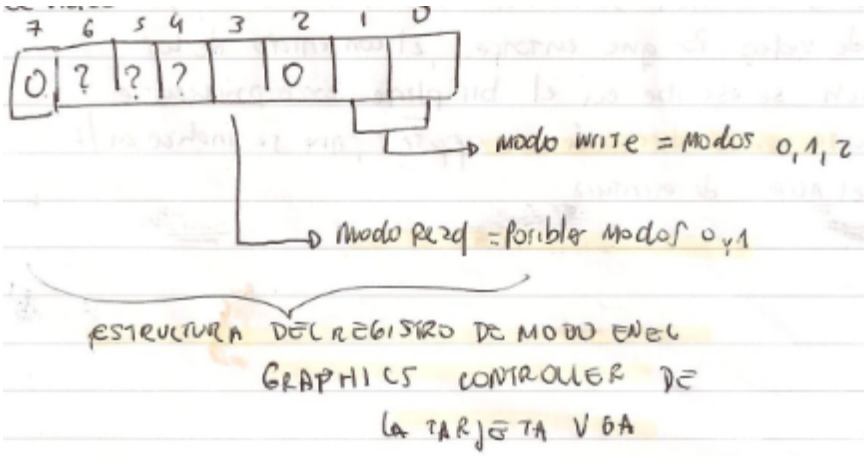
El hardware de la tarjeta VGA conmuta entre CPU y RAM de video, cuatro registros de 8 bits, los llamados registros *latch*. Cada uno de estos registros se corresponde con uno de los 4 BitPlanes, pero no es accesible directamente a un programa.
Si un programa realiza un acceso de lectura a un byte en la RAM de video, los 4 registros latch cargan junto al byte de su BitPlane, cuya dirección de offset se ha especificado en conexión con el comando de lenguaje de máquina que ha disparado el acceso de lectura.
Si la dirección de offset es por ejemplo 9, se carga el décimo byte (o es el primero) del primer BitPlane en el primer registro latch, el décimo byte del segundo BitPlane en el segundo registro latch, y correspondientemente se procede con el tercer y el cuarto registro latch.
El mismo proceso también se realiza en el acceso de escritura en la RAM de video. Porque entonces, el contenido de los registros latch se escribe en el BitPlane correspondiente y

concretamente en la dirección de offset que se indicó en la conexión con el acceso de escritura.



El papel del Controlador de Gráficos

Controlador Gráfico: Determina el desde dónde, a dónde y el cómo en todos los accesos de lectura y escritura a la RAM de video.



Modo Read 0

Le ofrece a un programa la posibilidad de leer bytes de un BitPlane determinado. Ej: cuando se quiere guardar una parte de la RAM de video con un bucle se pueden recorrer

sucesivamente los 4 BitPlanes cargando la zona deseada de cada BitPlane y colocándolo en la memoria principal.
En este modo se cargan todos los registros latch del byte direccionado en su plano, pero sólo uno de estos 4 bytes llega hasta la cpu a través de su registro latch.

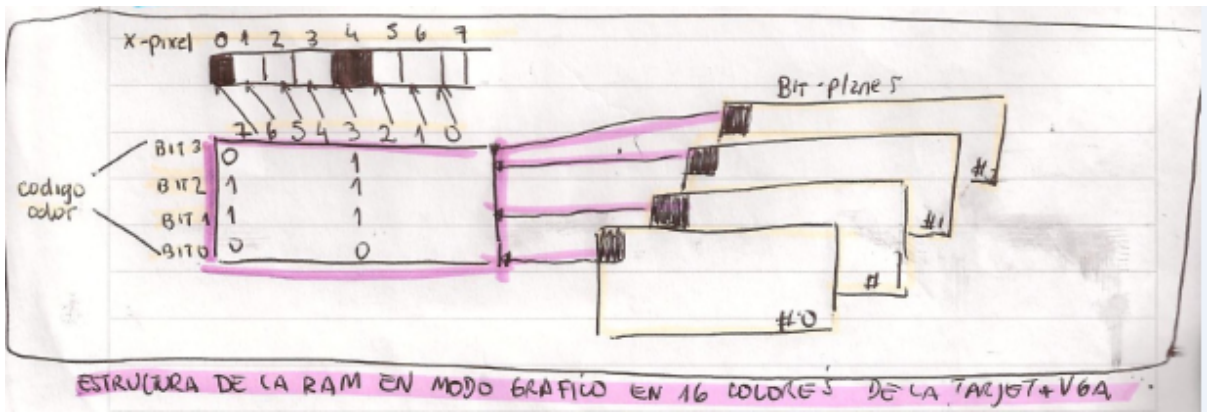
Modo Read 1
La tarea de este modo es verificar si los bits de los cuatro registros latch tienen un valor determinado. Ej: buscar puntos de un color determinado.

Modo Write 0
Con este modo se realizan una serie de operaciones que dependen del contenido de diferentes registros.

Modo Write 1
Se utiliza cuando una zona determinada de la RAM de video debe copiarse a otra.

Modo Write 2
Se asemeja a una combinación de los distintos modos en el modo write 0.
Este modo sirve especialmente para fijar el color de puntos individuales en los modos gráficos de 16 colores en las tarjetas VGA.

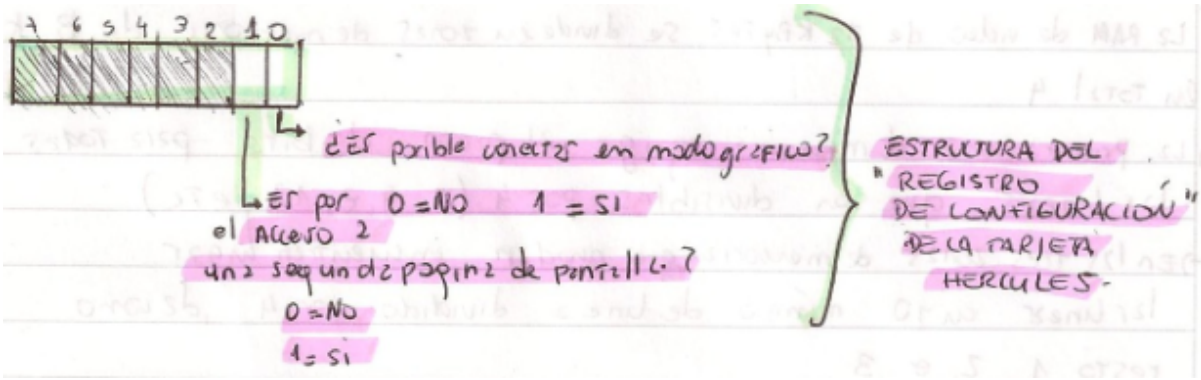
Modo Write 3



Los 8 primeros puntos gráficos del ángulo superior izquierdo de la pantalla se representan, por ejemplo, mediante los 4 bytes que se encuentran en los 4 BitPlanes de la dirección de offset 0000h.
La información de color para cada uno de los puntos se consigue agrupando en una determinada posición de bit los 4 bits de cada uno de los 4 bytes.
De este procedimiento resultan 8 grupos de 4, cada uno de los cuales se encarga del color de un punto de la pantalla.
El bit del BitPlane #0 constituye el bit 0 del código de color, el bit del BitPlane #1 el bit 1, y así sucesivamente para los bits 2 y 3 del código de color.

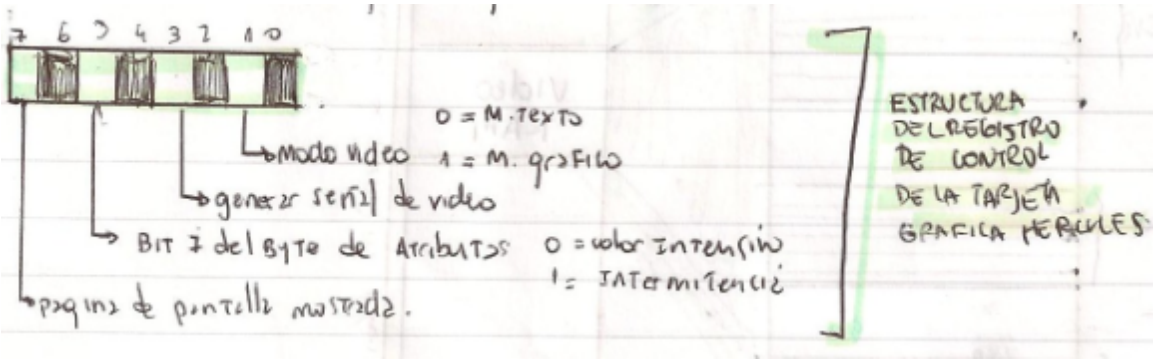
Tarjeta Hércules
Para poder trabajar en el modo gráfico, la tarjeta Hércules dispone de 64 kB de RAM en total, que se dividen en dos páginas de pantalla. Cada una de las páginas tiene 32 kB y puede acoger una página de texto que sólo emplea 4 kB.

La primera página de pantalla va desde la dirección B000:0000 hasta la B800:7FFF. A ella le sigue inmediatamente la segunda página de pantalla que cubre las porciones de memoria B000:8000 hasta B000:FFFF.



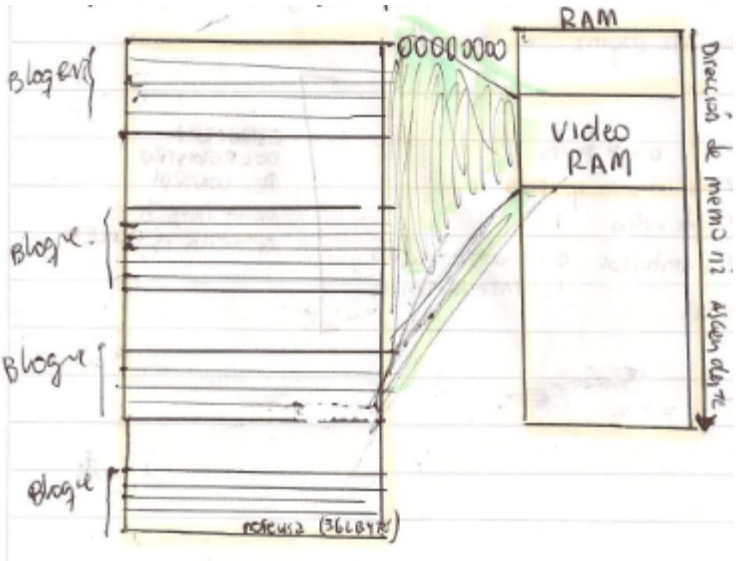
Resolución en modo gráfico 720x384 puntos en monocromo

Para evitar colisiones con otras tarjetas de video (con las de color por ejemplo) los 2 bits se ponen en 0 durante el arranque del sistema, de modo que para empezar no se pueden representar gráficos ni se puede utilizar la segunda página de pantalla.

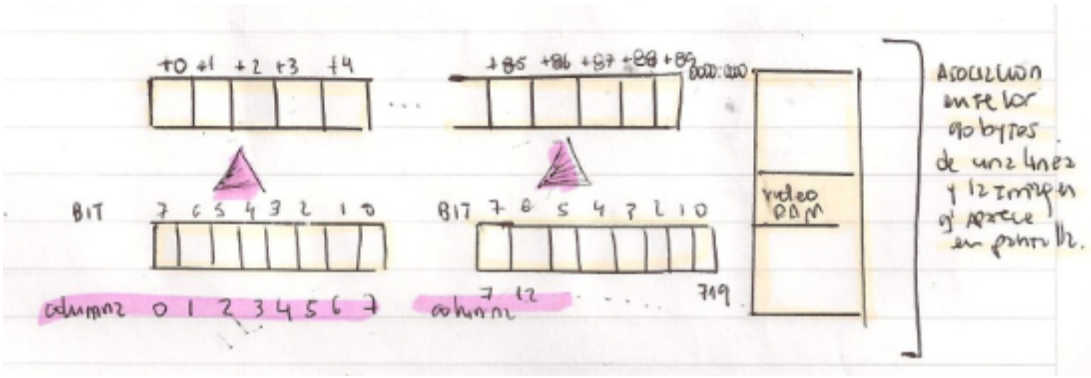


Modo gráfico de Hércules

Mientras que los registros al arrancar el sistema se ajustan automáticamente con ayuda del BIOS a los valores para el modo texto de 80x25 caracteres, el ajuste del modo gráfico no es posible con el BIOS, entonces no se puede evitar la programación directa de estos registros.
Como dijimos, la tarjeta Hércules representa 348 líneas a 720 puntos cada una. Cada punto en la pantalla corresponde con un bit en la RAM de video, si el bit correspondiente contiene el valor 1, el punto de la pantalla es visible, sino permanece oscuro.
La RAM de video de 32 kB se divide en 2 zonas de memoria de 8 kB, en total 4.
La primer zona de memoria acoge al patrón de bits para todas las líneas que son divisibles por 4 (0, 4, 8, 12, etc).
En las tres zonas de memoria que quedan, encuentran lugar las líneas cuyo número de línea dividido por 4 da como resto 1, 2 o 3.
Para la segunda zona de memoria estas son las líneas 1, 5, 9, 13.
La tercer zona de memoria contiene los patrones para las líneas 2, 6, 10, 14, etc y la última zona de memoria aloja las líneas 3, 7, 11, 15, etc.



Enlace entre los diferentes bloques de la RAM de video y la imagen que se forma tomando como ejemplo la primera pantalla.
En los diferentes bloques de datos, cada línea ocupa 90 bytes, ya que un punto ocupa un bit (720 puntos entre 8 bits = 90 bytes), así los primeros 90 bytes del primer bloque de memoria contienen los patrones de bits para la línea de pantalla cero, y los siguientes 90 bytes el patrón de bits de la cuarta línea de pantalla.
El primer byte de esta secuencia de 90 bytes representa las primeras 8 columnas de una línea de pantalla, el segundo byte las columnas 8 al 15, etc.
En los bytes individuales, el bit 7 se corresponde con el punto izquierdo de pantalla y el bit 0 con el punto derecho.



Si se numeran los puntos de pantalla de una línea de 0 a 719 y los de una columna de 0 a 347, de lo dicho resulta la siguiente fórmula que devuelve la dirección del byte en la que hay codificado un punto (x;y): $Dirección = 2000h \cdot (y \bmod 4) + 90 \cdot int(y \div 4) + int(x \div 8)$
Para obtener el n° de los bits deseados en este byte: $N° de bits = 7 - (x \bmod 8)$

Mouse

Un ratón de PC normalmente posee dos botones, con ayuda de estos botones, el usuario puede transmitir instrucciones a un programa.
Además hay otra información importante que es la posición del ratón o, mejor dicho, del cursor del ratón en pantalla. Este cursor le confiere al ratón el carácter de un instrumento para apuntar a algo, el hecho de pulsar el botón izquierdo del ratón siempre se hace servir con la información de fondo de la posición del cursor en la pantalla, esta posición indica

sobre qué objeto de pantalla que ha seleccionado el usuario se encuentra el puntero del ratón.

La pantalla virtual del ratón

Las posiciones del ratón se interpretan en el interior del driver del ratón siempre en relación a una pantalla gráfica virtual, cuya resolución depende del modo de video actual y por ello de la tarjeta de video de la que se dispone. Esta pantalla virtual también se utiliza en los diferentes modos texto para determinar la posición del cursor del ratón y representan la base para la comunicación con la interface del ratón, se ha de realizar cada vez el cálculo de correspondencia entre estas coordenadas gráficas y la coordenada de línea y columna del cursor del ratón, ya que cada línea y columna corresponden a 8 puntos, las coordenadas gráficas se dividen entre 8, es decir, se desplazan 3 posiciones de bit hacia la derecha.

El cursor del ratón

En la pantalla, el ratón se representa por un cursor de ratón, que sigue los movimientos del ratón en el escritorio.

En el modo texto puede estar representado por el cursor de hardware intermitente.

¿Cómo funciona?

Al desplazar el ratón sobre una superficie, la bola o sensor mueve los rodillos que están en contacto con ella. Un rodillo se encarga de los movimientos laterales y otro de los verticales. Los rodillos están conectados a unas ruedas llamadas codificadores que están situados enfrente de unos pequeños sensores de luz.

Estas ruedas poseen unas ranuras que permiten el paso de luz hasta unos dispositivos fotosensibles que detectan los destellos y los traducen en información codificada que el ordenador es capaz de interpretar.

Al pulsar algún botón del ratón, se genera otro tipo de señal que el ordenador distinguirá del anterior y que dependiendo del programa que se esté utilizando, permitirá realizar operaciones cuando este se desplaza. El movimiento de la bolita que está en su parte inferior se descompone en 2 movimientos según dos ruedas con ejes perpendiculares entre si (en correspondencia con los ejes de coordenadas x e y) que un conversor analógico/digital traduce en pulsos eléctricos, la cantidad de pulsos generados para cada representa la distancia recorrida por la bolita respecto de ese eje y en relación con la última posición en el que el mouse estuvo quieta. Dichos pulsos se van guardando en contadores, uno para cada eje, pudiendo ser la cuenta progresiva o regresiva, según el sentido del movimiento del mouse respecto de dichos ejes.

Los circuitos envían por un cable que va hacia un puerto serie del computador, el valor de la cuenta de los contadores como 2 números de 8 bits. Se envían 3 bytes cuando se pulsa o libera una tecla del mouse, aunque este no se mueva. Cuando el puerto recibe el primero de los 3 bytes, la plaqueta con la interfaz del buffer que contiene el circuito de dicho puerto solicita al procesador que interrumpa el programa en ejecución y pare a ejecutar la subrutina (driver mouse) que maneja la información del mouse.

Puertos - Serie y Paralelo

Son elementos materiales del equipo que permiten que el sistema se comunique con los elementos exteriores. Permiten el intercambio de datos.

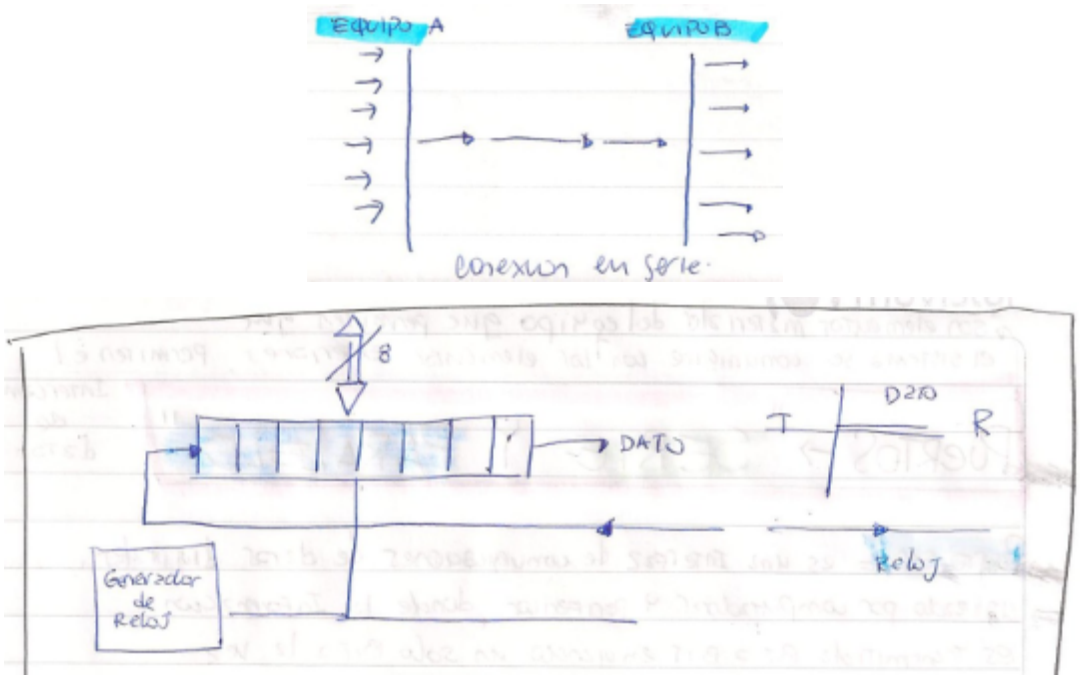
Puerto Serie

Es una interfaz de comunicaciones de datos digitales, utilizado por computadoras y periféricos donde la información es transmitida bit a bit enviando un sólo bit a la vez.

Trabaja mucho más lento que un puerto paralelo ya que debe enviar cada bit aislado a través de la línea, uno detrás de otro. Para el mismo ratio de bits (de pulsos), un puerto serie es forzosamente ocho veces más lento que el paralelo. Si bien desde el punto de vista de la velocidad presenta inconvenientes, desde el punto de vista del cableado ofrece una ventaja decisiva.

La conexión en serie trabaja igualmente en “serie” ya que no tiene ocho líneas de datos más una de control, sino que ÚNICAMENTE UNA LÍNEA DE MASA Y UNA PARA DATOS (si se tienen que transmitir datos en las dos direcciones se precisan dos líneas, lo que de todos modos hace un total de 3 líneas). Por este motivo, los cables de los puertos paralelos son más pesados y caros.

En los casos en los que los diferentes dispositivos se encuentren relativamente cerca, el cableado no es un factor importante, pero cuando se encuentran alejados unos de otros y el ordenador del que deben conectarse, el coste de los cables juega un papel importante. Un ejemplo: hay cables de transmisión serie en cualquier parte donde haya un cable de teléfono.



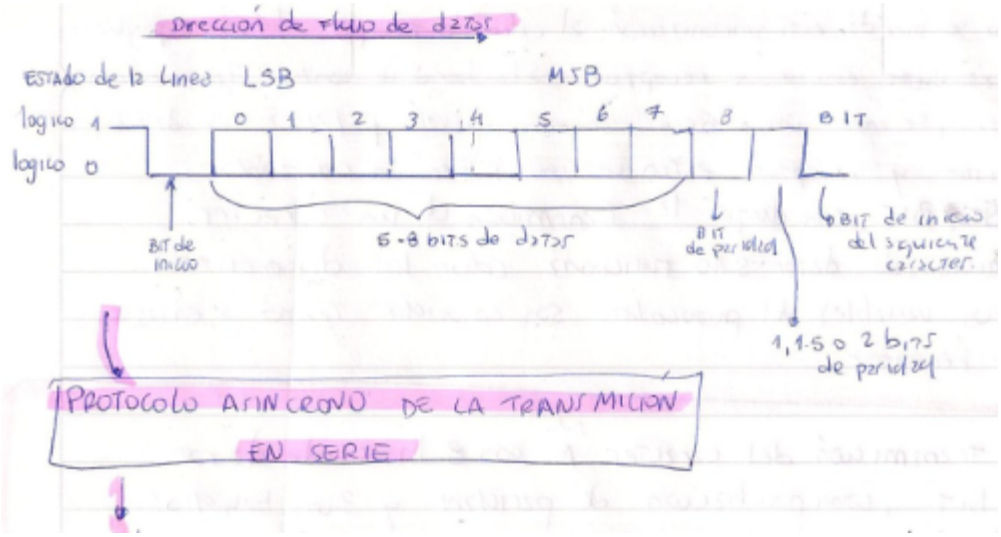
Comunicación Asíncrona/Síncrona

En las comunicaciones a través de líneas eléctricas cabe distinguir entre comunicación síncrona y asíncrona.

Comunicación síncrona

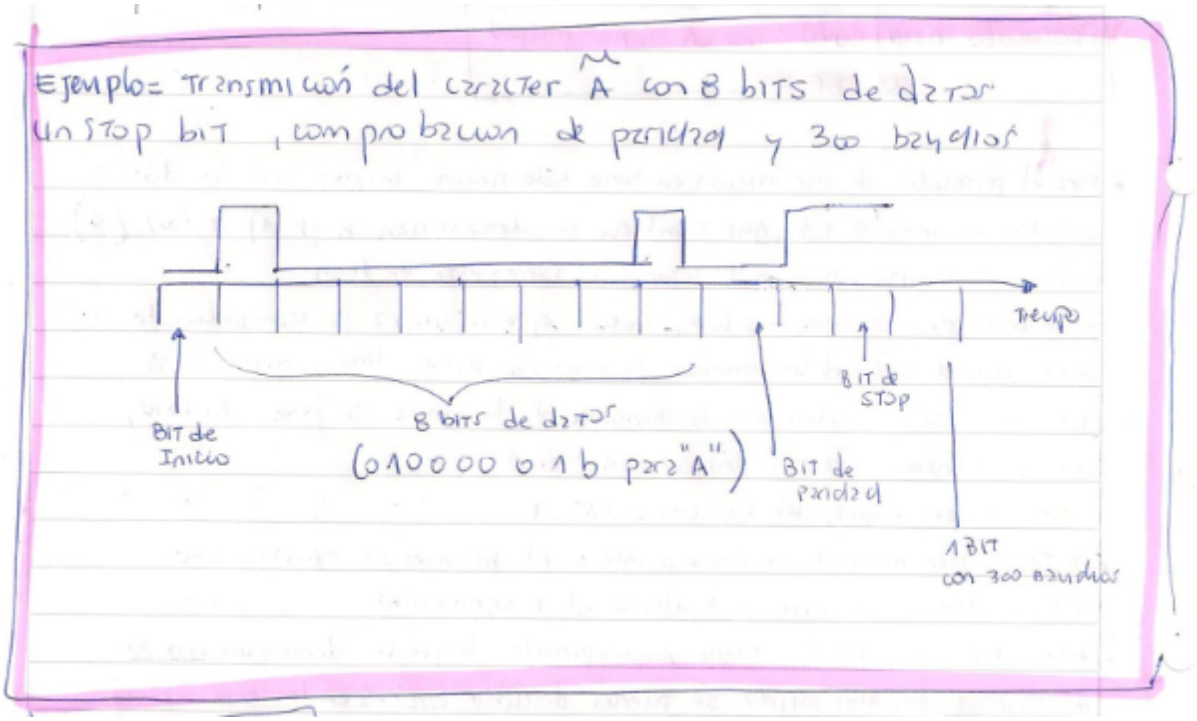
Son cuando el emisor y el receptor disponen de una señal de pulsos común que ayuda a coordinar todas sus acciones. Por ejemplo, que el emisor antes de un nuevo pulso envíe el

siguiente bit a la línea y el receptor sepa que con la llegada del pulso puede recibirlo, en este caso el emisor y receptor están sincronizados.
Para la sincronización se precisa siempre una línea adicional a través de la cual el emisor y receptor intercambian la señal de pulso PERO en la transmisión en serie a través de un cable de dos líneas esto NO es posible ya que ambas están ocupadas por los datos y la masa.
La sincronización debe llevarse a cabo a través de las líneas de datos, con datos mismos, por este motivo se intercalan palabras antes y después de los datos, que pueden consistir en palabras de entre 5 y 8 bits.



.- Para el protocolo de transmisión en serie sólo tienen importancia los dos estados de línea 0 y 1, que también se denominan high (1) y low (0).
Si no se transmite ningún carácter, la línea está en high, si se modifica su estado a low, indica que se inicia la transmisión de datos. Dependiendo de los convenios se enviarán por la línea entre 5 y 8 bits de datos. Si durante la transmisión, la línea se pone en low, se está transmitiendo un bit con valor cero, si se pone en high, un bit con valor 1.
.- En toda transmisión, se envía primero el bit menos significativo del carácter a transferir y por último el más significativo.
.- A los bits de datos puede acompañarle lo que se denomina un bit de paridad, con cuya ayuda se puede descubrir los errores en la transmisión.
.- El *stop bit* es opcional, ya que indica la finalización de la transmisión de una palabra de datos. El protocolo de transmisión de datos permite 1, 1.5 y 2 stop bit, dependiendo de la longitud de la palabra. La velocidad de transmisión es medida en baudio (unidad de medida que representa el número de símbolos por segundo en un medio de transmisión digital, bit por segundo) que de debe estar configurada al mismo valor tanto para el emisor como para el receptor si se pretende que la transmisión funcione.
.- *Start bit*: cuando el receptor detecta el start bit, sabe que la transmisión ha comenzado y es a partir de entonces que debe leer las señales de la línea a distancias concretas de tiempo en función del ratio de baudios determinado para que no se pierda esta sincronización al cabo de un tiempo debido a pequeñas diferencias entre emisor y receptor a la hora de contar las fracciones de tiempo. Se envía un número start bit con cada palabra de datos así el emisor y receptor están continuamente sincronizados.
Con el stop bit concluye la transmisión de un caracter.

La transmisión de datos sólo funciona cuando los diferentes parámetros variables del protocolo son conocidos tanto por el emisor como por el receptor.



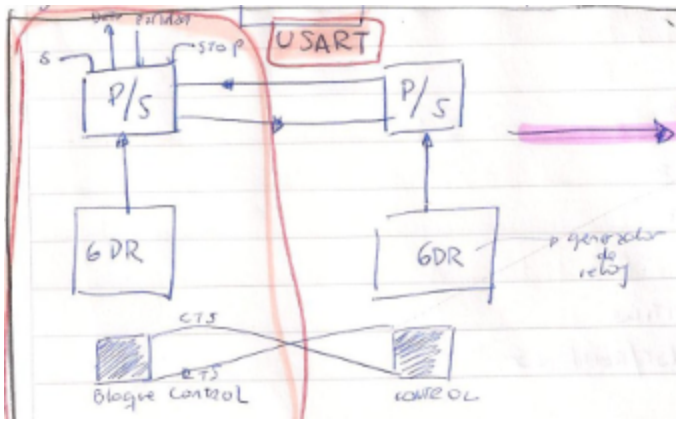
Norma RS232

Al abandonar el puerto serie, los bits entran en un mundo regido por la norma RS232.
Esta norma determina tanto las dimensiones del conector como el número de clavijas y su posición o los diferentes parámetros eléctricos.
Exactamente cuando se habla de puerto serie nos referimos a RS-232-C.
Para la transmisión serie en un sentido se precisan únicamente dos líneas (masa y datos), para la transmisión en los dos sentidos, sólo 3 (una masa para la línea de datos adicional en el sentido contrario).

La vida interior de un puerto serie

- UART - Asíncrono
- USART - Asíncrono/síncrono

En el interior del puerto serie hay un chip especial para la entrada y salida de caracteres y sobre todo para la conversión de palabras de datos en las correspondientes señales del puerto serie, lo que se denomina ***UART*** (Universal Asynchronous Receiver Transmitter / Emisor y Receptor Asíncrono Universal).



Este controla los puertos serie. Se encuentra integrado en la placa base. Funciones principales: Manejar las interrupciones de los dispositivos conectados al puerto serie y de convertir los datos en formato paralelo, transmitirlos al bus del sistema a datos en forma serie para que puedan ser transmitidos por los puertos.
El UART dispone en total de 10 registros desde el exterior. Si el UART recibe un caracter, los diferentes bits que se van recibiendo se “amontonan” primero en el RECEIVER SHIFT REGISTER hasta que se completa una palabra de datos. Si no aparece ningún fallo, el byte es transferido al Receiver Data Register desde donde puede ser leído via software.
En el sentido contrario, el software primero escribe la palabra de datos a enviar en el Transmitter Holding Register, de allí, el UART lo traslada al registro interno para desde ahí transmitir los diferentes bits, uno tras otro a través de la línea. Además también hay diferentes registros para inicializar el UART.
La comunicación entre el software y el UART puede darse tanto en modo Polling como en modo Interrupción.
En polling es responsabilidad del software en consultar el estado del UART en espacios de tiempos regulares, sólo así se puede determinar si se ha recibido un nuevo caracter o si el último caracter enviado se encuentra realmente en camino.
Desventaja: la cpu está todo el tiempo ocupada con el dispositivo, entonces los caracteres son enviados/recibidos de forma muy lenta, por este motivo es mejor utilizar Interrupciones, aunquesea más costoso de programar, ofrece todas las ventajas conocidas por su uso. La cpu sólo tiene que usar el puerto serie cuando realmente llega un caracter o tiene que enviarse.

Resumen UART

El controlador UART es el componente clave del subsistema de comunicaciones serie de decenas de computadoras. El UART toma bytes de datos y transmite los bits individuales de forma secuencial, en el destino, un segundo UART reensambla los bits en bytes completos. Cada UART contiene un registro de desplazamiento que es el método fundamental de conversión entre serie y paralelo.

Comunicación Serial Asíncrona mediante el módulo USART

Es un microchip que proporciona al ordenador una interfaz necesaria para la comunicación con modems y otros dispositivos serie. A diferencia de un UART, el USART ofrece la opcion del modo síncrono. En la comunicación de programa a programa, el modo síncrono requiere que cada extremo de un intercambio responda a su vez sin iniciar una nueva comunicación.

Diferencias entre el modo síncrono (USART) y modo asíncrono (USART - UART)

Modo Síncrono	*Modo Asíncrono*
Los datos se transmiten a una tasa fija	Los datos no tienen que ser transmitidos a una tasa fija
El dato síncrono se transmite en forma de bloques	Los datos asíncronos se transmiten normalmente un byte a la vez
Permite una mayor velocidad de transferencia de datos.	

Ejemplo USART: MODEM
El protocolo para la comunicación con un Modem:
Antes que la PC pueda enviar datos al modem debe enviarle diversas señales y aguardar a la esperada reacción del mismo.
Primero pone la línea DTR (Data Terminal Ready) en high para indicar al modem que el DTE (PC) está preparado para la comunicación con el modem. Como respuesta, el modem pone la línea DSR (Data Set Ready) en high, indicando con ello su disponibilidad para el establecimiento de la comunicación. Ambas líneas deben permanecer en high mientras dure la comunicación.
Además hay dos líneas de saludo que coordinan en conjunto proceder de PC y mode: RTS y CTS.
El pc es el primero en poner la línea RTS en high y con ello articular su predisposición a enviar datos. Si el modem está preparado para recibirlos, lo indica poniendo en high la línea CTS. Es como si CTS y RTS se ponen de acuerdo para enviar un byte.
Si el pc quiere enviar más datos, sólo deja la línea RTS en high y observa el estado de la línea CTS. Si ésta continúa en high, puede enviarse más bytes. Si por el contrario vuelve a estar en low, el pc debe esperar a que el modem vuelva a mostrar su disponibilidad a recibir datos poniendo de nuevo la línea en high antes de transmitir más información.
Además existen dos líneas más, RLSD y RI. RI se necesita cuando el pc actúa como correo electrónico y por tanto quiere captar las llamadas que recibe. A través de la línea RI, el modem indica al ordenador que esta sonando el teléfono (RI = Ring) y el pc puede entonces enviar al modem la interrupción correspondiente para descolgar y abrir con ello la línea RLSD como “Carrier Detect”. Como carrier se expresa el sonido que se recibe tan pronto como el modem opuesto ha descolgado. La recepción de este carrier indica que en el otro extremo de la línea hay un modem.

Resumen de Líneas
TxD (Transmitted Data): Los datos se transmiten a través de esta línea. El DTE (PC) sólo puede empezar a enviar cuando las 4 líneas de gobierno RTS, CTS, DSR y DTR tengan un uno lógico.
RxD (Received Data): Línea de datos del DCE (modem) al DTE (PC).
RTS (Request To Send): Poniendo esta línea en high, el DTE (PC) pregunta al DCE (modem) si está preparado para recibir datos.

DSR (Data Set Ready): Poniendo esta línea a 1, el DCE (Modem) indica al DTE (PC) que puede establecerse contacto con la parte contraria.
DTR (Data Terminal Ready): El DTE (PC) pone esta línea en 1 tan pronto como está preparado para comunicarse con el DCE (modem) con ello el modem sabe que esta conectado a un DTE activo.
RI (Ring Indicator): A través de RI, el DCE indica al DTE que hay una llamada en la línea telefónica a la que está conectada el modem.
RLSD (Received Line Signal Detector): A través de esta línea, el DCE indica al DTE que ha recibido una señal (carrier) desde el otro extremo de la línea, esto no implica que se haya establecido un verdadero contacto, la que puede darse que ambos DCE no encuentren un protocolo de transmisión común en cuanto a la modulación/demodulación.

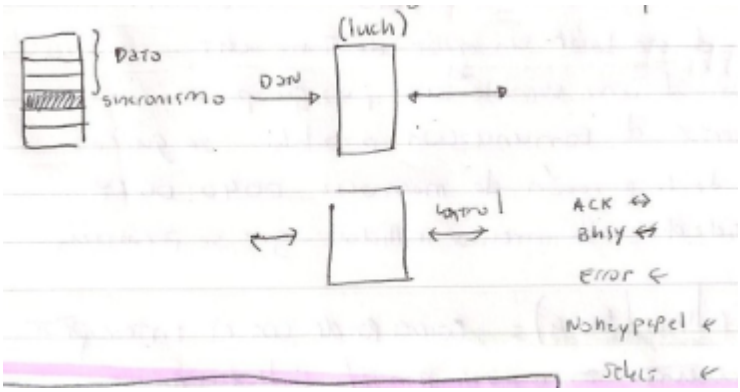
Puerto Paralelo

Es una interfaz entre una computadora y un periférico cuya principal característica es que los bits de datos viajan juntos enviado un paquete de bytes a la vez, es decir, se implementa un cable o una vía física para cada bit de datos formando un bus.
Para imprimir existen muchos caminos:

1. Programación directa de hardware.
2. Camino a través del ROM-BIOS
3. Acceso a las diferentes funciones del DOS.

El puerto paralelo es más simple que el puerto serie. Físicamente es un registro en donde yo escribo un dato y la impresora lo lee.
Para imprimir se usa la interrupción 17h y sus modos:

- 00h: enviar
- 01h: inicializar impresora
- 02h: estado impresora



Acceso a la impresora con la BIOS

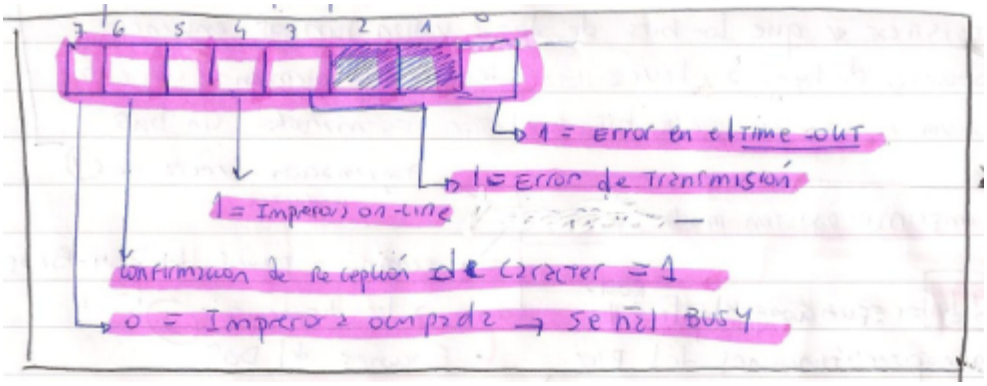
La interrupción *17h* está exclusivamente reservada para la comunicación con el puerto paralelo. En el PC se pueden conectar un máximo de 3 puertos paralelos diferentes

Interrupción 17h - Servicios

AH = 00h: Enviar caracter a la impresora. En AL el caracter a transmitir.
AH = 01h: Inicializar impresora / puerto paralelo. Siempre se debe ejecutar antes de enviar los datos a la impresora.
AH = 02h: Preguntar estado de impresora, leer el byte de estado en el registro AH.

Estado de la impresora:

Los 3 servicios de la interrupción 17h tienen en común que después de su llamada, cada uno de ellos devuelve el estado actual de la impresora en el registro AH.



.- Error en el timeout: El bit 0 del byte de estado aparece siempre que el BIOS intenta, durante un cierto período de tiempo, transmitir datos a la impresora pero esta no acepta los datos o avisa que está ocupada.

La cantidad de intentos que realiza el BIOS al transmitir un caracter, antes de señalizar un error de timeout, depende del contenido de una variable del BIOS, ya que para cada uno de los puertos de comunicación paralelos se guarda un byte a partir de la posición de memoria 0040:0078 que indica la cantidad de intentos fallidos que se permiten.

.- Impresora online/offline (bit 4): corresponde con el interruptor y el led correspondiente a la parte frontal de la impresora.

.- bit 7: señal busy, indica que la impresora está ocupada y por ello no puede aceptar más caracteres. Este bit es importante para el timeout porque es esta señal la que provoca que el ROM-BIOS repita una vez más el bucle porque no se puede enviar ningún otro caracter a la impresora. Este bir es el único que nos ofrece una lógica negativa, es decir, que tiene valor 0 cuando la impresora está ocupada. El valor nos indica la disponibilidad de la impresora.