Función 00h

Entrada: AX = 0000h Reseteo del driver del ratón

Salida: AX = FFFFh Driver del ratón instalado
0000h Driver no instalado
BX = Número de botones que tiene el ratón

Mediante una llamada a esta función podremos comprobar si está instalado el driver del ratón. Un ejemplo podría ser el siguiente:

```
char HayMouse(void)
{
   asm xor ax, ax
   asm int 33h

   return _AX;
}
```

Función 01h

Entrada: AX = 0001h Muestra el cursor del ratón

Salida: No devuelve nada

Cuando estamos en modo texto vemos el típico cuadrado, en modo gráfico la famosa flecha, pero hay una función que nos da la opción de cambiarlo si lo deseamos, tal y como veremos más adelante. Para los modos SVGA no nos queda más remedio que ir dibujando nosotros el ratón mediante un controlador, ya que en estos modos no nos muestra el cursor (al igual que pasaría con los modos x, por ejemplo).

```
void ShowMouse(void)
{
   asm mov ax, 01h
   asm int 33h
}
```

Función 02h

Entrada: AX = 0002h Oculta el cursor del ratón

Salida: No devuelve nada

Cuando el cursor está oculto, se sigue actualizando, así que nosotros podremos seguir leyendo su posición si lo deseamos. Una cosa a tener en cuenta es que si por ejemplo llamamos a esta función, pues no llamarla más ya que si lo hacemos así, al llamar a la función de poner ratón (01h) lo habremos de hacer dos veces también, si no, no se nos mostrará el cursor.

```
void HideMouse(void)
{
   asm mov ax, 02h
   asm int 33h
}
```

Función 03h

Entrada: AX = 0003h Leer posición (x, y) y estado de los botones

Salida: CX = Posición horizontal
DX = Posición vertical
BX = Estado de los botones

Bit 0 1 = Pulsado el botón izquierdo
Bit 1 1 = Pulsado el botón derecho
Bit 2 1 = Pulsado el botón central

Cuando estamos en el modo gráfico 13h (320x200) las coordenadas no corresponden con las que nos devuelve la función y tendremos que dividir entre dos el eje horizontal (x).

Eje X: de 0 a 639 (dividimos entre 2)
Eje Y: de 0 a 199

En modo texto tendremos que multiplicar las coordenadas por ocho, tanto en eje horizontal (x) como en vertical (y). La función siguiente solos nos sirve en modo 13h.

```
void ReadMouse(int *MouseX, int
*MouseY, int *MouseB)
{
   int x, y, b;

   asm mov ax, 03h
   asm int 33h
   asm shr cx, 1        // cx/2
   asm mov x, cx
   asm mov y, dx
   asm mov b, bx

   *MouseX=x;
   *MouseY=y;
   *MouseB=b;
}
```

Función 04h

Entrada: AX = 0004h Posiciona el cursor del ratón
CX = Posición horizontal
DX = Posición vertical

Salida: No devuelve nada

Debemos tener en cuenta lo dicho anteriormente, y transformar las coordenadas para el modo en el que estemos trabajando.

```
void SetMouseXY(int x, int y)
{
   asm mov ax, 04h
   asm shl x, 1          // x*2
   asm mov cx, x
   asm mov dx, y
   asm int 33h
}
```

Funciones 07h y 08h

Entrada: AX = 0007h Limita movimiento horizontal del ratón.
CX = Mínima posición eje horizontal (x)
DX = Máxima posición eje horizontal (x)

Salida: No devuelve nada

Entrada: AX = 0008h Limita movimiento vertical del ratón
CX = Mínima posición eje vertical (y)
DX = Máxima posición eje vertical (y)

Salida: No devuelve nada

Estas funciones hacen que se limite el ratón a un área. Esto nos puede servir por ejemplo, para cuando sale

un mensaje en nuestro programa, el usuario centre la atención en esa zona restringiendo las coordenadas a las de donde está el mensaje. Acuérdense de que también debemos convertir las coordenadas a las del modo que estemos usando.

```
void SetMouseLimit(int x1, int y1, int x2,
int y2)
{
   asm shl x1, 1          // x1*2
   asm shl x2, 1          // x2*2
   asm mov ax, 07h
   asm mov cx, x1
   asm mov dx, x2
   asm int 33h

   asm mov ax, 08h
   asm mov cx, y1
   asm mov dx, y2
   asm int 33h
}
```

Función 09h

Entrada: AX = 0009h Cambia apariencia del cursor.
DX = Offset bitmap donde esta el cursor del ratón
ES = Segmento bitmap donde esta el cursor del ratón.

Salida: No devuelve nada

Esta funcion cambia la apariencia del cursor de nuestro mouse. En la librería mouse.h podrás encontrar varios cursores. La función que se encarga de esto es la siguiente:

```
void SetNewForm(unsigned int *form)
{
   asm mov ax, 09h
   asm xor bx, bx
   asm xor cx, cx
   asm les dx, [form]
   asm int 33h
}
```

Función 0Ch

Entrada: AX = 000Ch Instalar el controlador de eventos
CX = Máscara de eventos que hará que el controlador sea llamado.

Bit 0 Movimiento del ratón
Bit 1 Pulsado el botón izquierdo
Bit 2 Soltado el botón izquierdo
Bit 3 Pulsado el botón derecho
Bit 4 Soltado el botón derecho
Bit 5 Pulsado el botón central
Bit 6 Soltado el botón central

ES:DX = Dirección del controlador que queremos instalar

Salida: No devuelve nada

Esta función es la que hace que un controlador nos vaya actualizando una serie de variables que cada una nos indicará una cosa. Aquí por controlador entendemos una función que vaya pasando los valores de CX, DX, etc. que serán los que actualizará el driver a nuestras variables para que después nosotros podamos leer, un ejemplo de un

controlador podría ser el siguiente:

```
void far MyEventHand()
{
   asm shr cx, 3
   asm inc dx
   asm mov [Mx], cx        // Mx: Posición x
   asm shr dx, 3
   asm inc dx
   asm mov [My], dx        // My: Posición y
   asm mov [Mbot], bl      // Mbot: Botón
pulsado
   asm ret
}
```

No hace falta decir que las variables tiene que ser de tipo global.

La información que el driver del ratón pasa a nuestro controlador de eventos es la siguiente:

BX = Estado actual de los botones
- Bit 0 Pulsado botón izquierdo
- Bit 1 Pulsado botón derecho
- Bit 2 Pulsado botón central

CX = Coordenada eje horizontal.
DX = Coordenada eje vertical.

Los siguiente registros (SI y DI) vienen en unas unidades de medida del ratón (los famosos mickeys), y que corresponden a 1/200, 1/400 pulgadas.

SI = Longitud del último movimiento en eje horizontal.
DI = Longitud del último movimiento en eje vertical.

En fin, esto es todo lo que hay que saber para poder programar mínimamente bien el mouse