Introduction.

GTMOUSE is a Borland Pascal 6.0 & 7.0 Unit for mouse handling. GTMouse is not a mouse driver, it works over any standard mouse driver. If included it converts standard block mouse cursor in smoothly moving across the screen arrow-like mouse cursor in TEXT MODE of video adapter. This is similar to Norton Utilities 6.0, PCTools 7.0 and some others programs. GTMouse will make your application program much more attractive and professionally looking. This is possible only for EGA or VGA adapters, it don't do anything on others adapters.

Initially developed for Borland's *Turbo Vision* package, it may be used with various TP mouse drivers, such as TurboPower Software's *Object Professional,* IdSoft D&M's *MouseLib* and others.

News in version 1.3.

- The support of BP7 protected mode;
- Internal Sleeper is included in unit.

File List.

This package contains the following files.

GTMOUSE6.TPU	- TP 6.0 unit (rename to GTMOUSE.TPU)
GTMOUSE.TPU	- TP 7.0 unit
GTMOUSE.TPP	- BP 7.0 protected mode unit
GTMOUSE.DOC	- this file
GTMOUSE.WRI	- the same as GTMOUSE.DOC but in Windows 3.1 Write format
GTMOUSE.TXT	- interface section of GTMouse.TPU
GTMOUSE.REG	- registration File.
TVBDEMO.PAS	- simple Turbo Vision sample program with
TVBDEMO.EXE	GTMouse included. Slow modification of Borland's TVGUID14.PAS.
LIBDEMO.PAS	- MouseLib 6.0 sample with GTMouse included
LIBDEMO.EXE	
OPDEMO.PAS	- Object Professional Mouse sample with GTMouse included
OPDEMO.EXE	

1. Usage of GTMouse.

Simply copy GTMOUSE.TPU file in your TPU direction and include GTMouse in **use** clause of your program

```
  uses GTMouse,...{any others TPUs};
```

ATTENTION. The GTMouse must be the FIRST used TPU of program. More accuratly it must precede any TP mouse driver (such as Drivers of Turbo Vision package).

Now recompile and run your program. You will see the arrow-like mouse cursor which is smoothly moving over the screen and changing with pushing of any mouse button. Besides this the behavior of your program doesn't have to change.

You can use GTMouse without any additional calls, but they can improve your program.

1.1 Mouse Cursor Form Changing.

GTMouse contains some predefined mouse cursor images.There are two procedures to select mouse cursor image

```
  SelectNotPressedImage(nmImage:ImageNames);
  SelectPressedImage(nmImage:ImageNames);
```

for notpressed and pressed mouse button respectevly. Moreover you can make your own image and connect it with any ImageName by procedure

```
  LinkUserImageWith(nmImage:ImageNames,MyArrow:ImageArray);
```

where ImageArray = array[0..15] of byte. Example of Cursor Image:

```
Byte      BitMap        MyArrow
 0       00000000         $00
 1       01000000         $40
 2       01100000         $60
 3       01110000         $70
 4       01111000         $78
 5       01111100         $7c
 6       01111110         $7e
 7       01111000         $78
 8       01101100         $6c
 9       01001100         $4c
10       00001100         $0c
11       00001100         $0c
12       00000000         $00
13       00000000         $00
14       00000000         $00
15       00000000         $00
```

<u>**1.2 Activate and deactivate GTMouse.**</u>

There are two procedures

```
      DoneGTMouse;
      InitGTMouse;
```

to deactivate and reactivate GTMouse respectively.

If your program use COMMAND.COM to start other programs or run DOS shell you must deactivate GTMouse before this and reactivate it after:

```
  . . . .
  DoneGTMouse; {Deactivate GTMouse}
  SwapVectors;
  Exec(GetEnv('COMSPEC'),...);
  SwapVectors;
  InitGTMouse; {Reactivate GTMouse}
  . . . .
```

At the program begining GTMouse activates automatically.

<u>**1.3 Reserved symbols.**</u>

If you are using GTMouse there are eight symbols (from 256 possible) which cann't be used to screen output. These symbols reserved by GTMouse to dynamically redefine their views to form graphics mouse cursor. By default they are (#01, #02, #03, #04, #208, #209, #215, #216). To replace them with any others use

```
     ChangeCursorArray(var mArray);
```

where parameter is array[1..8] of chars. Last four characters must be in pseudo graphics range ($c0..$df), first four must not be in this range. Last demand connected with VGA peculiarities in text mode.

1.4 Internal Sleeper.

Sleeper activates automatically with default params. You can reactivate it at any moment with

```
  InitSleeper(nWaitTime,xUpLeft,yUpLeft,xDnRight,yDnRight);
```

where
 nWaitTime is a no-event waiting time to sleep (in seconds), other params - screen region in which the occurance of mouse cursor cause the "sleeping" of program.
The default is

```
     InitSleeper(30,80,1,80,1);
```

To deactivate sleeper simply set variable *Sleeper* to **false**.

2. About samples.

TVBDEMO.EXE is not a special GTMouse sample, it is slightly modified Turbo Vision Guide example TVGUID14.PAS. Besides GTMouse we include only the possibility to change mouse cursor form (with Alt-H key or from status line). You can look also the modified windows and buttons views, to compare compile and run TVBDEMO.PAS with your compiler and Turbo Vision TPU's. This TVBDEMO.EXE has been compiled with our own version of TurboVision TPU's, all registered users of GTMouse can obtain them on special request.

Others samples illustrate the technique of GTMouse usage with Object Professional and MouseLib packages. The main advise is to avoid any screen SCROLLs and screen SAVING/RESTORING with active mouse cursor, so your code must be something about

```
  . . . . .
  HideMouse; {call to hide mouse in your mouse driver}
  {Any screen changing code}
  ShowMouse;
  . . . .
```

3. Some recomendations.

- deactivate GTMouse when use internal debugger of Turbo Pascal IDE;
- deactivate GTMouse before any videomode change. GTMouse works with various screen text modes, simply reactivate it after mode changing.
Important. Deactivate GTMouse in graphics modes.

4. Registration.

The Graphic mouse cursor in Text mode unit (GTMouse) is provided on 'as is' basis without warranty of any kind, expessed or implied. The person using the software bears all risk as to the quality and performance of the software.

We will try to extend our support to unregistrered users during their evaluation period, however we reserve the right to limit our support for unregistered users if their requests become taxing for us. For registered users the support is unlimited. Questions and any comments on this unit and source code can be sended to us via E-Mail addresses.

evsikov@lcta5.jinr.dubna.su or
sychov@jinr.dubna.su

or via usual mail to

Igor Evsikov
LCTA, Joint Institute for Nuclear Research, (Dubna)
PO Box 79, Head Post Office
101000 Moscow Russia

For registration details look file GTMOUSE.REG.

Russia, Dubna, June 30 1993.
▯