Standard Video Modes

Setting Normal VGA Resolutions

When setting normal (non-VESA) resolutions, we can easily use a BIOS function call. All we have to do is load the appropriate values into the CPU registers and call the video interrupt! It can look something like this:

```
void StandardVideoMode(int mode)
{asm{mov ax,mode
        int 10h
       }
}
```

or if you prefer straight CPP then try

```
void StandardVideoMode(int mode)
{union REGS regs;
 regs.x.ax=mode;
 int86(0x10,&regs,&regs);
}
```

That's all there is too it! I honestly only use this function to switch to text mode (3) or good old 320x200x256 mode (19). Here's a good way to structure our function:

```
 void Video::StandardVideoMode(short mode)
 { union REGS regs;
   regs.x.ax=mode;
   if(!ClearMem)
     {regs.x.ax+=128; }
   if(mode == 0x13)
    {Screen_Width=320; Screen_Height=200; Screen_Size=64000;
     }
   if(mode == 3)
    {Screen_Width=80;
     Screen_Height=25;
     Screen_Size=4000;
     }
   int86(0x10,&regs,&regs);
   CurrentVideoMode=mode;
   Move2Vid = &Video::MoveMemStandard;
   XCenter = Screen_Width >>1;
   YCenter = Screen_Height >>1;
 }
```

Here we've added a couple of new twists. Firstly this function is contained in my Video class where I have a couple of new variables. We use a new variable called ClearMem that is a flag telling the code wether it is ok to clear the contents of Video RAM. This flag must be set before our function is called. We set several screen variables, Screen_Width, Screen_Height, Screen_Size according to which mode we are setting. I'm only testing my favorite two video modes in the code, but I think you can add the appropriate changes for other modes. CurrentVideoMode is simply a class variable that holds, amazingly, the current video mode. Lastly, notice we set a new variable called Move2Vid. Previously it was a function that moved our offscreen buffer to Video RAM. Now it is a function pointer. MoveMemStandard is the function that will move our video buffer to VRAM. For more explanation on how these work, take a look at the MasterMove2Vid tutorial! Without further delay, here's the much anticipated mode listing!

Mode Listing

If bit 7 of the mode is set when our StandardVideoMode function is called, contents of Video Ram won't be cleared, otherwise changing video modes will automatically clear the screen. Some of the modes are duplicated because and EGA/ VGA card can't differ the output of the color signal in Modes 0,1,2,3,4 and 5. As you will see below, these modes really aren't very useful! As I noted previously, I only use two modes out of the list. I suggest mastering these two modes if you are a beginning programming. I know that it is really tempting to try at higher resolutions and color depths, but you must go through the steps to really appreciate and fully understand what is going on.

Mode Number	Description
0	40x25 Text Mode, 16 Colors
1	40x25 Text Mode, 16 Colors
2	80x25 Text Mode, 16 Colors
3	80x25 Text Mode, 16 Colors
4	320x200 Graphics Mode, 4 Colors
5	320x200 Graphics Mode, 4 Colors
6	640x200 Graphics Mode, 2 Colors
7	80x25 Text Mode, Mono
8	Undocumented
9	Undocumented
10	Undocumented
11	Undocumented
12	Undocumented
13	320x200 Graphics Mode, 16 Colors
14	640x200 Graphics Mode, 16 Colors
15	640x350 Graphics Mode, Mono
16	640x350 Graphics Mode, 16 Colors
17	640x480 Graphics Mode, 2 Colors
18	640x480 Graphics Mode, 16 Colors
19	320x200 Graphics Mode, 256 Colors

Contact Information

I just wanted to mention that everything here is copyrighted, feel free to distribute this document to anyone you want, just don't modify it! You can get a hold of me through my website or direct email. Please feel free to email me about anything. I can't guarantee that I'll be of ANY help, but I'll sure give it a try :-)

Email : Justin Deltener <deltener@mindtremors.com>
Webpage : http://www.inversereality.org

Inverse Reality