What is vesafb?

This is a generic driver for a graphic framebuffer on intel boxes.

The idea is simple: Turn on graphics mode at boot time with the help
of the BIOS, and use this as framebuffer device /dev/fb0, like the m68k
(and other) ports do.

This means we decide at boot time whenever we want to run in text or
graphics mode. Switching mode later on (in protected mode) is
impossible; BIOS calls work in real mode only. VESA BIOS Extensions
Version 2.0 are required, because we need a linear frame buffer.

Advantages:

* It provides a nice large console (128 cols + 48 lines with 1024x768)
  without using tiny, unreadable fonts.
* You can run XF68_FBDev on top of /dev/fb0 (=> non-accelerated X11
  support for every VBE 2.0 compliant graphics board).
* Most important: boot logo :-)

Disadvantages:

* graphic mode is slower than text mode...

How to use it?

Switching modes is done using the vga=... boot parameter. Read
Documentation/svga.txt for details.

You should compile in both vgacon (for text mode) and vesafb (for
graphics mode). Which of them takes over the console depends on
whenever the specified mode is text or graphics.

The graphic modes are NOT in the list which you get if you boot with
vga=ask and hit return. The mode you wish to use is derived from the
VESA mode number. Here are those VESA mode numbers:

```
    | 640x480  800x600  1024x768 1280x1024
----+-------------------------------------
256 |  0x101    0x103    0x105    0x107
32k |  0x110    0x113    0x116    0x119
64k |  0x111    0x114    0x117    0x11A
16M |  0x112    0x115    0x118    0x11B
```

The video mode number of the Linux kernel is the VESA mode number plus
0x200.

```
 Linux_kernel_mode_number = VESA_mode_number + 0x200
```

So the table for the Kernel mode numbers are:

	640x480	800x600	1024x768	1280x1024
256	0x301	0x303	0x305	0x307
32k	0x310	0x313	0x316	0x319
64k	0x311	0x314	0x317	0x31A
16M	0x312	0x315	0x318	0x31B

To enable one of those modes you have to specify "vga=ask" in the
lilo.conf file and rerun LILO. Then you can type in the desired
mode at the "vga=ask" prompt. For example if you like to use
1024x768x256 colors you have to say "305" at this prompt.

If this does not work, this might be because your BIOS does not support
linear framebuffers or because it does not support this mode at all.
Even if your board does, it might be the BIOS which does not. VESA BIOS
Extensions v2.0 are required, 1.2 is NOT sufficient. You will get a
"bad mode number" message if something goes wrong.

1. Note: LILO cannot handle hex, for booting directly with
   "vga=mode-number" you have to transform the numbers to decimal.
2. Note: Some newer versions of LILO appear to work with those hex values,
   if you set the 0x in front of the numbers.

X11

XF68_FBDev should work just fine, but it is non-accelerated. Running
another (accelerated) X-Server like XF86_SVGA might or might not work.
It depends on X-Server and graphics board.

The X-Server must restore the video mode correctly, else you end up
with a broken console (and vesafb cannot do anything about this).

Refresh rates

There is no way to change the vesafb video mode and/or timings after
booting linux. If you are not happy with the 60 Hz refresh rate, you
have these options:

* configure and load the DOS-Tools for the graphics board (if
  available) and boot linux with loadlin.
* use a native driver (matroxfb/atyfb) instead if vesafb. If none
  is available, write a new one!
* VBE 3.0 might work too. I have neither a gfx board with VBE 3.0
  support nor the specs, so I have not checked this yet.

Configuration

The VESA BIOS provides protected mode interface for changing
some parameters. vesafb can use it for palette changes and
to pan the display. It is turned off by default because it
seems not to work with some BIOS versions, but there are options

```
to turn it on.

You can pass options to vesafb using "video=vesafb:option" on
the kernel command line.  Multiple options should be separated
by comma, like this: "video=vesafb:ypan,invers"

Accepted options:

invers  no comment...

ypan    enable display panning using the VESA protected mode
        interface.  The visible screen is just a window of the
        video memory, console scrolling is done by changing the
        start of the window.
        pro:    * scrolling (fullscreen) is fast, because there is
                  no need to copy around data.
                * You'll get scrollback (the Shift-PgUp thing),
                  the video memory can be used as scrollback buffer
        kontra: * scrolling only parts of the screen causes some
                  ugly flicker effects (boot logo flickers for
                  example).

ywrap   Same as ypan, but assumes your gfx board can wrap-around
        the video memory (i.e. starts reading from top if it
        reaches the end of video memory).  Faster than ypan.

redraw  scroll by redrawing the affected part of the screen, this
        is the safe (and slow) default.


vgapal  Use the standard vga registers for palette changes.
        This is the default.
pmipal  Use the protected mode interface for palette changes.

mtrr:n  setup memory type range registers for the vesafb framebuffer
        where n:
              0 - disabled (equivalent to nomtrr) (default)
              1 - uncachable
              2 - write-back
              3 - write-combining
              4 - write-through

        If you see the following in dmesg, choose the type that matches the
        old one. In this example, use "mtrr:2".
...
mtrr: type mismatch for e0000000,8000000 old: write-back new: write-combining
...

nomtrr  disable mtrr

vremap:n
        remap 'n' MiB of video RAM. If 0 or not specified, remap memory
        according to video mode. (2.5.66 patch/idea by Antonino Daplas
        reversed to give override possibility (allocate more fb memory
        than the kernel would) to 2.4 by tmb@iki.fi)
```

```
vtotal:n
        if the video BIOS of your card incorrectly determines the total
        amount of video RAM, use this option to override the BIOS (in MiB).

Have fun!

  Gerd

--
Gerd Knorr <kraxel@goldbach.in-berlin.de>

Minor (mostly typo) changes
by Nico Schmoigl <schmoigl@rumms.uni-mannheim.de>
```