Driver for PXA25x LCD controller

The driver supports the following options, either via options=<OPTIONS> when modular or video=pxafb:<OPTIONS> when built in.

For example:

```
modprobe pxafb options=vmem:2M,mode:640x480-8,passive
```

or on the kernel command line:

```
video=pxafb:vmem:2M,mode:640x480-8,passive
```

vmem: VIDEO_MEM_SIZE

> Amount of video memory to allocate (can be suffixed with K or M for kilobytes or megabytes)

mode:XRESxYRES[-BPP]

> XRES == LCCR1_PPL + 1
> YRES == LLCR2_LPP + 1
>
> > The resolution of the display in pixels
>
> BPP == The bit depth. Valid values are 1, 2, 4, 8 and 16.

pixclock:PIXCLOCK

> Pixel clock in picoseconds

left:LEFT == LCCR1_BLW + 1
right:RIGHT == LCCR1_ELW + 1
hsynclen:HSYNC == LCCR1_HSW + 1
upper:UPPER == LCCR2_BFW
lower:LOWER == LCCR2_EFR
vsynclen:VSYNC == LCCR2_VSW + 1

> Display margins and sync times

color | mono => LCCR0_CMS

> umm...

active | passive => LCCR0_PAS

> Active (TFT) or Passive (STN) display

single | dual => LCCR0_SDS

> Single or dual panel passive display

4pix | 8pix => LCCR0_DPD

> 4 or 8 pixel monochrome single panel data

hsync:HSYNC, vsync:VSYNC

> Horizontal and vertical sync. 0 => active low, 1 => active high.

dpc:DPC

> Double pixel clock. 1=>true, 0=>false

outputen:POLARITY

> Output Enable Polarity. 0 => active low, 1 => active high

pixclockpol:POLARITY

pixel clock polarity 0 => falling edge, 1 => rising edge

Overlay Support for PXA27x and later LCD controllers

PXA27x and later processors support overlay1 and overlay2 on-top of the base framebuffer (although under-neath the base is also possible). They support palette and no-palette RGB formats, as well as YUV formats (only available on overlay2). These overlays have dedicated DMA channels and behave in a similar way as a framebuffer.

However, there are some differences between these overlay framebuffers and normal framebuffers, as listed below:

1. overlay can start at a 32-bit word aligned position within the base framebuffer, which means they have a start (x, y). This information is encoded into var->nonstd (no, var->xoffset and var->yoffset are not for such purpose).
2. overlay framebuffer is allocated dynamically according to specified 'struct fb_var_screeninfo', the amount is decided by:

   ```
   var->xres_virtual * var->yres_virtual * bpp
   ```

   bpp = 16 -- for RGB565 or RGBT555

   bpp = 24 -- for YUV444 packed

   bpp = 24 -- for YUV444 planar

   bpp = 16 -- for YUV422 planar (1 pixel = 1 Y + 1/2 Cb + 1/2 Cr)

   bpp = 12 -- for YUV420 planar (1 pixel = 1 Y + 1/4 Cb + 1/4 Cr)

   NOTE:

   a. overlay does not support panning in x-direction, thus var->xres_virtual will always be equal to var->xres
   b. line length of overlay(s) must be on a 32-bit word boundary, for YUV planar modes, it is a requirement for the component with minimum bits per pixel, e.g. for YUV420, Cr component for one pixel is actually 2-bits, it means the line length should be a multiple of 16-pixels
   c. starting horizontal position (XPOS) should start on a 32-bit word boundary, otherwise the fb_check_var() will just fail.
   d. the rectangle of the overlay should be within the base plane, otherwise fail

   Applications should follow the sequence below to operate an overlay framebuffer:

   a. open("/dev/fb[1-2]", ...)
   b. ioctl(fd, FBIOGET_VSCREENINFO, ...)
   c. modify 'var' with desired parameters:
      1. var->xres and var->yres
      2. larger var->yres_virtual if more memory is required, usually for double-buffering
      3. var->nonstd for starting (x, y) and color format
      4. var->{red, green, blue, transp} if RGB mode is to be used
   d. ioctl(fd, FBIOPUT_VSCREENINFO, ...)
   e. ioctl(fd, FBIOGET_FSCREENINFO, ...)
   f. mmap
   g. ...

3. for YUV planar formats, these are actually not supported within the framebuffer framework, application has to take care of the offsets and lengths of each component within the framebuffer.
4. var->nonstd is used to pass starting (x, y) position and color format, the detailed bit fields are shown below:

   ```
   31                23  20         10          0
    +-----------------+---+----------+----------+
    |  ... unused ... |FOR|   XPOS   |   YPOS   |
    +-----------------+---+----------+----------+
   ```

   FOR - color format, as defined by OVERLAY_FORMAT_* in pxafb.h

   - 0 - RGB
   - 1 - YUV444 PACKED
   - 2 - YUV444 PLANAR
   - 3 - YUV422 PLANAR
   - 4 - YUR420 PLANAR

   XPOS - starting horizontal position

   YPOS - starting vertical position