Framebuffer显示开发

Framebuffer(帧缓冲)是Linux为显示设备提供的一个接口，把显存抽象后的一种设备，它允许上层应用程序在图形模式下直接对显示缓冲区进行读写操作。Framebuffer是LCD对应的一种件抽象层，提供抽象统一的操作接口，用户不必关心硬件层是如何实现。这些都是由Framebuffer设备驱动来完成的。

帧缓冲设备对应的设备文件为/dev/fb*，如果系统有多个显示卡，Linux下还可支持多个帧缓冲设备，最多可达32个，分别为/dev/fb0到 /dev/fb31，而/dev/fb则为当前缺省的帧缓冲设备，通常指向/dev/fb0，在嵌入式系统中支持一个显示设备就够了。帧缓冲设备为标准字符设备，主设备号为29，次设备号则从0到31。分别对应/dev/fb0-/dev/fb31。

在`linux/fd.h`目录下定义了linux framebuffer的数据结构体。

```
struct fb_info {
        atomic_t count;
        int node;
        int flags;
        struct mutex lock;               /* Lock for open/release/ioctl funcs */
        struct mutex mm_lock;               /* Lock for fb_mmap and smem_* fields */
        struct fb_var_screeninfo var;        /*  */
        struct fb_fix_screeninfo fix;        /*  */
        struct fb_monspecs monspecs;        /* */
        struct work_struct queue;        /* Framebuffer event queue */
        struct fb_pixmap pixmap;        /* Image hardware mapper */
        struct fb_pixmap sprite;        /* Cursor hardware mapper */
        struct fb_cmap cmap;                /*  */
        struct list_head modelist;      /* mode list */
        struct fb_videomode *mode;        /* current mode */

#ifdef CONFIG_FB_BACKLIGHT
        /* assigned backlight device */
        /* set before framebuffer registration,
           remove after unregister */
        struct backlight_device *bl_dev;

        /* Backlight level curve */
        struct mutex bl_curve_mutex;
        u8 bl_curve[FB_BACKLIGHT_LEVELS];
#endif
#ifdef CONFIG_FB_DEFERRED_IO
        struct delayed_work deferred_work;
        struct fb_deferred_io *fbdefio;
#endif

        struct fb_ops *fbops;        /*,ioctl()*/
        struct device *device;                /* This is the parent */
        struct device *dev;                /* This is this fb device */
        int class_flag;                    /* private sysfs flags */
#ifdef CONFIG_FB_TILEBLITTING
        struct fb_tile_ops *tileops;    /* Tile Blitting */
#endif
        char __iomem *screen_base;        /* Virtual address */
        unsigned long screen_size;        /* Amount of ioremapped VRAM or 0 */
        void *pseudo_palette;               /* Fake palette of 16 colors */
#define FBINFO_STATE_RUNNING        0
#define FBINFO_STATE_SUSPENDED        1
        u32 state;                          /* Hardware state i.e suspend */
        void *fbcon_par;                /* fbcon use-only private area */
        /* From here on everything is device dependent */
        void *par;
        /* we need the PCI or similar aperture base/size not
           smem_start/size as smem_start may just be an object
           allocated inside the aperture so may not actually overlap */
        struct apertures_struct {
                unsigned int count;
                struct aperture {
                        resource_size_t base;
                        resource_size_t size;
                } ranges[0];
        } *apertures;
};
```

`struct fb_fix_screeninfo` 这个结构体用来描述设备无关，不可变更的信息。用户可以使用FBIOGET_FSCREENINFO命令来获得这些信息。

```
struct fb_fix_screeninfo {
        char id[16];                       /* identification string eg "TT Builtin" */
        unsigned long smem_start;        /* () */
        __u32 smem_len;                       /* */
        __u32 type;                         /* fbTFTSTN*               */
        __u32 type_aux;                       /* Interleave for interleaved Planes */
        __u32 visual;                        /*,         */
        __u16 xpanstep;                       /* zero if no hardware panning  */
        __u16 ypanstep;                       /* zero if no hardware panning  */
        __u16 ywrapstep;                /* zero if no hardware ywrap    */
        __u32 line_length;               /* length of a line in bytes    */
        unsigned long mmio_start;        /* I/O, */
        __u32 mmio_len;                       /* Length of Memory Mapped I/O  */
        __u32 accel;                        /* Indicate to driver which        */
                                        /*  specific chip/card we have        */
        __u16 capabilities;                /* see FB_CAP_*                        */
        __u16 reserved[2];                /* Reserved for future compatibility */
};
```

struct fb_var_screeninfo该结构描述设备无关的，可以更改的配置信息。应用程序可以使用FBIOGET_VSCREENINFO命令来获得这些信息，使用FBIOPUT_VSCREENIFO命令写入这些信息

```
struct fb_var_screeninfo {
        __u32 xres;                         /* visible resolution               */
        __u32 yres;
        __u32 xres_virtual;                /* virtual resolution               */
        __u32 yres_virtual;
        __u32 xoffset;                        /* offset from virtual to visible */
        __u32 yoffset;                        /* resolution                         */

        __u32 bits_per_pixel;                /* guess what                         */
        __u32 grayscale;                /* 0 = color, 1 = grayscale,        */
                                        /* >1 = FOURCC                          */
        struct fb_bitfield red;                /* bitfield in fb mem if true color, */
        struct fb_bitfield green;        /* else only length is significant */
        struct fb_bitfield blue;
        struct fb_bitfield transp;        /* transparency                        */

        __u32 nonstd;                        /* != 0 Non standard pixel format */

        __u32 activate;                        /* see FB_ACTIVATE_*                */

        __u32 height;                        /* height of picture in mm    */
        __u32 width;                        /* width of picture in mm     */

        __u32 accel_flags;                /* (OBSOLETE) see fb_info.flags */

        /* Timing: All values in pixclocks, except pixclock (of course) */
        __u32 pixclock;                        /* pixel clock in ps (pico seconds) */
        __u32 left_margin;                /* time from sync to picture        */
        __u32 right_margin;                /* time from picture to sync        */
        __u32 upper_margin;                /* time from sync to picture        */
        __u32 lower_margin;
        __u32 hsync_len;                /* length of horizontal sync        */
        __u32 vsync_len;                /* length of vertical sync        */
        __u32 sync;                        /* see FB_SYNC_*                */
        __u32 vmode;                        /* see FB_VMODE_*                */
        __u32 rotate;                        /* angle we rotate counter clockwise */
        __u32 colorspace;                /* colorspace for FOURCC-based modes */
        __u32 reserved[4];                /* Reserved for future compatibility */
};
```