转载 默默的码农 2016-07-12 20:14:32 1763 收藏 6
分类专栏： 嵌入式 文章标签： framebuffer linux

所有的这些操作，都是在控制台界面下，root登录。
一，
$ cat /dev/fb0 > sreensnap /*获取一屏的数据*/
$ clear /*清楚屏幕的输出*/
$ cat sreensnap > /dev/fb0 /*将刚才的屏幕数据显示*/

二，操作/dev/fb0
1）查看/dev/fb0 的信息

```c
#include <unistd.h>
#include <stdio.h>
#include <fcntl.h>
#include <linux/fb.h>
#include <sys/mman.h>
#include <stdlib.h>

int main ()
{
     int fp=0;
     struct fb_var_screeninfo vinfo;
     struct fb_fix_screeninfo finfo;
     fp = open ("/dev/fb0",O_RDWR);

     if (fp < 0){
      printf("Error : Can not open framebuffer device/n");
      exit(1);
     }

     if (ioctl(fp,FBIOGET_FSCREENINFO,&finfo)){
      printf("Error reading fixed information/n");
      exit(2);
     }

     if (ioctl(fp,FBIOGET_VSCREENINFO,&vinfo)){
      printf("Error reading variable information/n");
      exit(3);
     }

     printf("The mem is :%d\n",finfo.smem_len);
     printf("The line_length is :%d\n",finfo.line_length);
     printf("The xres is :%d\n",vinfo.xres);
     printf("The yres is :%d\n",vinfo.yres);
     printf("bits_per_pixel is :%d\n",vinfo.bits_per_pixel);
     close (fp);
}
```

2）改变屏幕上某一个点的颜色

```c
#include <unistd.h>
```

分类专栏
Ubuntu 3篇
嵌入式 22篇
Android图形显示 13篇
图形图像 4篇
Linux 8篇
NTP 1篇

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <linux/fb.h>
#include <sys/mman.h>

int main ()
{
     int fp=0;
     struct fb_var_screeninfo  vinfo;
     struct fb_fix_screeninfo  finfo;
     long screensize=0;
     char *fbp = 0;
     int x = 0, y = 0;
     long location = 0;
     fp = open ("/dev/fb0",O_RDWR);

     if (fp < 0)
     {
          printf("Error : Can not open framebuffer device/n");
          exit(1);
     }

     if (ioctl(fp,FBIOGET_FSCREENINFO,&finfo))
     {
          printf("Error reading fixed information/n");
          exit(2);
     }

     if (ioctl(fp,FBIOGET_VSCREENINFO,&vinfo))
     {
          printf("Error reading variable information/n");
          exit(3);
     }

      screensize = vinfo.xres * vinfo.yres * vinfo.bits_per_pixel / 8;
     /*这就是把fp所指的文件中从开始到screensize大小的内容给映射出来，得到一个指向这块空间的指针*/
     fbp =(char *) mmap (0, screensize, PROT_READ | PROT_WRITE, MAP_SHARED, fp,0);
     if ((int) fbp == -1)
     {
        printf ("Error: failed to map framebuffer device to memory./n");
        exit (4);
     }
     /*这是你想画的点的位置坐标,(0，0)点在屏幕左上角*/
     x = 100;
     y = 100;
     location = x * (vinfo.bits_per_pixel / 8) + y  *  finfo.line_length;

     *(fbp + location) = 100;  /* 蓝色的色深 */  /*直接赋值来改变屏幕上某点的颜色*/
     *(fbp + location + 1) = 15; /* 绿色的色深*/
     *(fbp + location + 2) = 200; /* 红色的色深*/
     *(fbp + location + 3) = 0;  /* 是否透明*/

     munmap (fbp, screensize); /*解除映射*/
     close (fp);    /*关闭文件*/
     return 0;

}
```

三，framebuffer 内部结构

数据结构：framebuffer 设备很大程度上依靠了下面四个数据结构。这三个结构在fb.h 中声明。

Struct fb_var_screeninfo //用来描述图形卡的特性的。通常是被用户设置的。
Struct fb_fix_screeninfo // 定义了图形卡的硬件特性， 是不能改变的，用户选定了哪一个图形卡，那么它的硬件特性也就定下来了。
Struct fb_info //定义了当前图形卡framebuffer 设备的独立状态，一个图形卡可能有两个framebuffer， 在这种情况下，就需要两个fb_info 结构。这个结构是唯一在内核空间可见的。

```
1 ) fb_var_screeninfo解析
```

```
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 Graylevels instead of colors*/

    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; /* acceleration flags (hints) */

    /* 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 reserved[6]; /* Reserved for future compatibility*/

};
```

前几个成员决定了分辨率。
xres和yres是在屏幕上可见的实际分辨率，
xres-virtual决定了构建屏幕时视频卡读取屏幕内存的方式。
bits_per_pixel 设为1，2，4，8，16，24或32来改变颜色深度

```
    2)  fb_fix_screeninfo
```

```c
struct fb_fix_screeninfo {

    char id[16]; /* identification string eg "TT Builtin" */

    unsigned long smem_start; /* Start of frame buffer mem */

    /* (physical address) */

    __u32 smem_len; /* Length of frame buffer mem */

    __u32 type; /* see FB_TYPE_* */

    __u32 type_aux; /* Interleave for interleaved Planes */

    __u32 visual; /* see FB_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; /* Start of Memory Mapped I/O */

    /* (physical address) */

    __u32 mmio_len; /* Length of Memory Mapped I/O */

    __u32 accel; /* Type of acceleration available */

    __u16 reserved[3]; /* Reserved for future compatibility */

};
```

3) 显示说明

【双显示器例子】
一个例子，可能就是双显示，最近刚刚看到实际某开发者的系统，就是两个显示器，鼠标移动超过单个显示器，到最右边的时候，就跑到另一个显示器了。对于常常用多系统或者需要打开很多东西的开发人员，这个功能很实用。
帧缓冲可以用于 页面交换page flipping（也常叫做 双缓冲double buffering），许多游戏都是采用此技术，以实现更流畅的视频输出，以便用户获得更好的游戏体验。此技术也被用于3D图形加速。
【双缓冲的主要实现原理】
假如你的显示器是VGA模式，640×400，也就是虚拟的分辨率是640X800，也就是800线（每一行的数据，称为一条线，也就是640X1的数据了）。800线的数据存储于Framebuffer，而实际的显示内容，只是400线，
Linux内核中的Framebuffer模型中，对应有个变量yoffset，就是表示的这个具体的纵坐标，默认是0，所以显示的内容就是，0 - 399线，由于和实际显示 页面大小等同，所以此处可以简称为第一帧。
如果yoffset改变了，比如此例中变为400，那就是显示剩余的部分，400 - 799线。此处简称为第二帧。
在系统显示第一帧的时候，系统在后台悄悄地准备第二帧的数据，所以，等第一帧显示完成，多数时候，第二帧的数据也准备好了，就可以直接显示，同时系统又在准备接下来的一帧的数据，这样就可以大大提高显示效率。

【平滑地滚动页面的实现原理】
同上，在显示完第一帧数据的时候，也就是0 - 399线的时候，将yoffset设置为1，就可以显示1 - 400线的数据了，显示完成后，再设置yoffset为2，就显示2 - 401线的数据，这样，就可以一点点地，平滑地显示整个滚动画面了。其实也就是画面在垂直方向的滚动。其中yoffset的增加，可以使用定时器，各个一段时间，比如10us，增加1，系统自动会更新显示对应的内容，这样我们所看到的内容就是滚动的画面了。

此外，Linux中的Framebuffer模型中，提供了一些ioctl功能，给定一些参数，然后系统可以实现对应的功能，其中有个参数就是FBIOPAN_DISPLAY。具体也就是类似如下调用：

ioctl (framebuffer_handler, FBIOPAN_DISPLAY, &variable_info);

而这个调用，如果显示不支持framebuffer的双缓冲的话，那么其framebuffer的缓冲大小，就是和物理上的显示器大小等同，那么对应的yoffset也就不会像双缓冲那样变化了。

也就是说，如果显卡/显示屏控制器不支持双缓冲，那么yoffset就应该一直为0，并且在运行时候，也不应该改变，也不应该去给FBIOPAN_DISPLAY的参数调用ioctl。

转载：http://blog.csdn.net/tianshuai1111/article/details/8502613

linux framebuffer打开图片 10-24

linux framebuffer编程，显示BMP图片

/dev/fb0入门练习(linux FrameBuffer)_pk_20140716的专栏 10-27

X-Window-System是Unix/Linux上的图形系统,它是通过X-Server来控制硬件的。但有一些Linux的发行版在引导的时候就会在屏幕上出现图形,这时的图形是…

Linux操作系统下使用FrameBuffer直接写屏_vrix的专栏 9-29

因为Linux是工作在保护模式下,所以用户态进程是无法象DOS那样使用显卡BIOS里提供的中断调用来实现直接写屏,故Linux抽象出FrameBuffer这个设备来…

linux图像显示（一）framebuffer操作 JT同学的博客 5034

linux图像显示 linux图像显示（一）framebuffer操作 linux图像显示（二）bmp图片 linux图像显示（三）使用libjpg处理jpg图片 linux图像显示（四）使用libp…

【嵌入式Linux驱动程序-基础篇】 - FrameBuffer架构 santapasserby的博客 1036

FrameBuffer架构 Linux内核中，FrameBuferr设备驱动的源码主要分布在linux/include/fb.h和linux/drivers/video/fbmem.c，它们处于FrameBuffer驱动…

Linux framebuffer简介及操作 欢迎光临 2346

Framebuffer简介 ： 帧缓冲（framebuffer）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，他允许上层应用程序在图形模式下直接…

浅析linux2.6 framebuffer wu7585535的专栏 1203

在Linux内核中，Framebuffer(帖缓冲)驱动是显示驱动的标准，Framebuffer 将显示设备抽象为帖缓冲区，用户通过内存映射到进程地址空间之后，就可以…

framebuffer驱动详解-linux驱动开发第7部分 10-18

本课程是linux驱动开发的第7个课程，主要内容是linux的framebuffer驱动详解，本课程带大家分析fb驱动的框架、构成以及一些代码细节，目标是让大家彻…

嵌入式linux操作framebuffer显示bmp图片 小飞的博客 2518

编译后拷贝进开发板即可使用 使用方法 ./fb_show_bmp test.bmp 显示的图片由参数指定，上面指令中test.bmp为测试用的bmp格式的图片 效果 源码说明 …

linux下实现对framebuffer(/dev/fb0)的截屏操作 gp_scorpius 8867

在linux系统中,使用framebuffer来提供用户态进程直接操作显示屏的功能. 在嵌入式系统开发中,需要对显示屏的内容进行截取,实现一个lcd截屏工具实现对…

linux framebuffer 测试程序 06-25

嵌入式应用中测试framebuffer。支持RGB565.彩虹色，七色转换。

linux framebuffer 02-13

基于Linux平台的 framebuffer开发。包括基本的点，线，圆，以及中文显示，bmp位图显示等。

framebuffer的入门介绍-实现程序分析 liuzijiang1123的专栏 1万+

如想想对lcd屏进行操作（例如在lcd屏幕上画线，或者显示视频数据），我们就必须得了framebuffer（帧缓冲），网上各种百度，大多都说的很官方，至…

对FrameBuffer的简单解释和用法示例 1847

大家都知道Unix/Linux系统是由命令驱动的。那么最基本的系统是命令行的（就是想DOS一样的界面）。X - Window - System是Unix/Linux上的图形系统，…

一、 FrameBuffer 原理、实现与应用 写屏（转） 热门推荐 或许 1万+

一、FrameBuffer 原理、实现与应用 一、FrameBuffer的原理 FrameBuffer 是出现在 2.2.xx 内核当中的一种驱动程序接口。 Linux是工作在保护模式…

默默的码农

码龄8年 暂无认证

17	84万+	114万+	11万+	等级
原创	周排名	总排名	访问	

1322	38	16	9	72
积分	粉丝	获赞	评论	收藏

私信 关注

LuckyToMeet: 大佬。这些文件的版本必须一样吗？我make libminigui时报错了。

RAW格式

weixin_45118391: ！

您愿意向朋友推荐“博客详情页”吗？

强烈不推荐 不推荐 一般般 推荐 强烈推荐

最新文章

dmalloc检测程序内存泄漏

bmp转jpeg文件

NTP时间同步

2017年 3篇　　2016年 43篇

2013年 1篇

全面的framebuffer详解一　　weixin_34235457的博客　75

转：http://blog.chinaunix.net/uid-20628575-id-72534.html 一、FrameBuffer的原理 FrameBuffer 是出现在 2.2.xx 内核当中的一种驱动程序接口。 …

从终端操作framebuffer　　sinc00的专栏　4813

参考http://www.armadeus.com/wiki/index.php?title=FrameBuffer， 总结一下 LCD进入睡眠 echo "0" > /sys/class/graphics/fb0/blank LCD显示退出睡眠…

保存framebuffer数据的两种方法(gsnap和T32 d.image命令)及lookat工具使用　　kentas_dxpc的博客　1463

1.save framebuffer (1) method1: we can dump mem,and use t32 to resume framebuffer. Kernel space Framebuffer address: setup_fb_mem ph vr sc…

FrameBuffer操作　　感雨韵的专栏　488

FrameBuffer是Linux下对于显存操作的抽象层，一般作为一个驱动文件，位置：/dev/fb0。 用C++封装了一下，便于使用： fbhelper.h #ifndef FBHELPER…

android下操作FrameBuffer　　wwater的专栏　265

一、framebuffer使用基础： 1. Linux是工作在保护模式下，所以用户态进程是无法象DOS那样使用显卡BIOS里提供的中断调用来实现直接写屏， Linux抽…

Android驱动深度开发视频教程　　05-23

也许是中国第一个讲解android驱动的课程，涵盖: bootloader，内核移植，INIT进程，框架(BINDER IPC,SERVICE FRAMEWORK Activity Manager Serive…

默默的码农 关注

1 0 6

专栏目录

举报