第3卷 第1期 应用气象学报 Vol. 3, №1
1992年2月 QUARTERLY JOURNAL OF APPLIED METEOROLOGY February 1992

XENIX 驱动程序设计及高分辨大屏幕显示开发

陈少林 罗建国

(武汉中心气象台)

提 要

本文结合一个简单的显示驱动程序例子讨论了 XENIX/386 系统下驱动程序设计的重要概念和关键技术,并介绍了开发大屏幕高分辨率彩色图形(图象)显示驱动软件的一些思路和技巧;而这正是目前 XENIX 系统显示软件所缺少的,并且为加强 XENIX 系统 I/O 能力使之支持新的外部设备提供了有效途径。

一、引 言

XENIX/386 系统的推出,为微机开辟了新的应用环境,特别是由于它的多用户多任务(多进程)特点,先进的系统管理,以及丰富的开发工具,给用户带来了很多方便。随着 XENIX 应用广泛深入的发展,各种 XENIX 环境下的应用课题已经提出,其中对系统 I/O 能力的新要求,例如在 XENIX 下使用高分辨彩色显示卡(Enhanced VGA)和大屏幕彩色监示器(支持 1024×768 以上分辨率)显示彩色图形(图象)或彩色汉字等,就是其中一个很困难的问题。在目前国内较新的 Rel 2.2.3 和 Rel 2.3.1 版本中,虽然可通过 SCO CGI[1] (Computer Graphics Interface)1.0 或 1.1 版本图形驱动软件包显示彩色图形,但由于其功能和速度的限制,用户很难真正利用,并且也不支持 1024×768 以上高分辨显示,这使得 XENIX 系统在图形(图象)显示方面受到很大限制。

目前,国内已有人开始在这方面做研究[2],其方法是利用现有版本中的驱动程序并做少量改动,可支持 MC6845 CRT 控制器在 640×450 分辨率下显示图形,但由于现有驱动程序 I/O 能力的局限,这种方法不能完全解决问题,也解决不了驱动高分辨显示问题。我们的做法与之不同,而是为系统设计新的驱动程序,并将新驱动程序装入 XENIX 内核;并且利用 DOS 系统的开放性,使 DOS 下已有的 I/O 功能移入于 XENIX 之中。这种方法虽然难度大,但它不仅解决了目前 XENIX 不能驱动 Enhanced VGA 卡的问题,而且也为 XENIX 驱动其它各种特殊外设创造了一种通用的方法,以便用户根据自己的需要对 XENIX 进一步开发,扩展 I/O 功能。

本文 1990 年 7 月 9 日收到,1991 年 4 月 11 日收到再改稿。

二、XENIX 系统设备驱动程序的结构原理

所谓设备驱动程序就是系统与外设的软件接口,它由一系列形式统一的驱动程序例程(Driver Defined Routines)组成,其功能是管理用户程序与外设之间的数据交换及设备运行控制。在 XENIX 环境下对外设的管理采用文件系统的统一形式,即每台外设对应一个(或几个)设备文件(Special File),用户的 I/O 请求通过对这些设备文件的打开、读/写与控制、关闭等操作,即使用与普通文件类似的系统调用来启动和运行驱动程序,完成 I/O 功能。在 XENIX 环境下用户程序、系统内核(Kernel)与设备驱动程序三者的关系如图 1 所示[3]。

XENIX 支持两种类型的外设,即字符设备与块设备,本文重点探讨字符型设备驱动程序的设计。下面结合图 1 对一些关键性的概念进行说明。

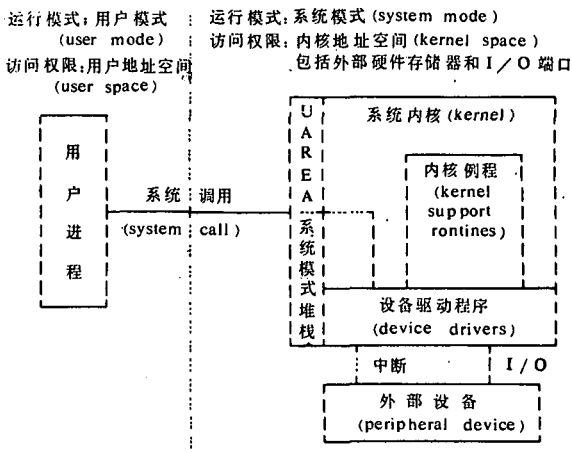

图 1　XENIX 系统设备 I/O 驱动程序运行示意图

• 驱动程序与设备:在 XENIX 下,一个驱动程序可同时管理多台同类外设(例如多台终端共享一个终端驱动程序);反过来,一台外设则可对应几个 I/O 驱动程序(当然,这些程序一般不可同时使用),这一点就为用户开发自己特色的驱动程序提供了机会。

• 访问权限(Access Privileges):XENIX 系统中的进程(Processes)在运行过程中分为两种模式,即用户和系统模式。系统提供了多进程保护,一个用户进程仅当通过进入系统模式执行内核代码时,才具备访问 I/O 端口和外部硬件存贮器(如显示卡的缓冲区 Video RAM)等特权,换句话说,任何用户进程只有通过调用安装在系统内核的设备驱动程序才能实现 I/O 操作(这一点不同于单用户 DOS,用户程序可直接使用 I/O 指令和访问物理地址)。

• u area:即系统模式堆栈(System mode stack),它的地址映射在内核地址空间,可用于用户进程与驱动程序之间的参数传送。

• clists:其数据结构为链表队列(Linked List Queue),是一种通用字符型数据缓冲机构,可作为用户进程与驱动程序之间的字符型数据 I/O 接口使用。

• 插入文件(Include Files):系统为开发 C 程序提供的工具,其中也包括了设计驱动程序有关的各种变量、常数和函数等定义,设计驱动程序时须仔细阅读,对应选用。

• 驱动程序与多进程环境:可采用系统提供的进程同步技术,例如使用信号灯(Semaphore)实现多进程环境下驱动程序的同步调用;若多个用户需同时使用同一驱动程序,则可采用排队的方法。

三、驱动程序的设计技术

下面我们结合附录中一个简单的显示驱动程序的例子，对于驱动程序的结构和设计要点进行说明。

1. 组成结构

字符型设备驱动程序通常由××init，××open，××close，××read，××write，××ioctl，××intr 等驱动程序例程组成，其中××init 例程设置在系统启动时运行，用于加电初始化。如果系统与外设交换的数据要求按中断方式进行，则中断处理部分可用××intr 实现，由于 XENIX 环境下每台外设的访问都统一使用文件的读/写方式，所以××open 和××close 是必要的，剩下的××read 和××write 以及××ioctl 等例程可根据驱动程序的功能要求和应用特点灵活处理，附录例中××read 和××write 用于用户数据区与显示缓冲区数据交换，作为整幅图形(图象)的缓存与输出显示，而××ioctl 则包括有设置 CGA 中分辨图形显示模式及在这个模式下画象素点等功能。

2. 设计要点

• I/O 端口访问：可使用 Kernel 提供的 out(b)，in(b)例程。

• 系统内核地址空间以外的存贮器访问：显示控制卡上的 Video RAM 空间不属于系统内核的地址空间，因此对这样的存贮器的读/写操作在 XENIX/386 环境下可直接使用 PTOK(x)函数[1]将 32 位物理地址变换成为系统内核的逻辑地址，提供给驱动程序使用。而在附录例中直接使用系统内核中已定义的逻辑地址指针(char *)B8000—sel，通过此地址可方便地访问启始地址在 B8000H 的 RAM 显示存贮器。

• 用户程序与驱动程序的参数传递：例中采用 copyioc( )例程，可在用户程序与驱动程序之间传递任何定义的结构数据，使用简便巧妙，参数的类型和个数不受限制(具体参见例中 grioctl( )例程)。

• 用户与设备数据传送：可采用 clists 结构，即在××read 中使用 getc；在××write 中使用 pute。附录例中采用 copyin 和 copyout 两个内核例程，并利用 u area 传送用户数据区地址和数据容量，目的在于提高 I/O 传送速度。

• 驱动程序设计好后，需要进行编译连接，具体步骤可参考文献[3]中有关内容。值得注意的是对于不同的 XENIX 版本，其编译与连接稍有差别，这可通过详细查看 usr/sys/conf 目录下的 c • c、makfile 和 link-xenix 等文件内容获得有用的信息。连接安装完成后，再通过 mknod 命令生成设备文件(Special file)。

假设附录中驱动程序所对应的设备文件取名为/dev/cga，则用户的应用程序可按如下形式编写。

```
/ * 变量和数据缓冲区定义 * /
int cgafil;
struct {
```

```
        short x;
        char y;
        char color;
          } point;
    char img-buf[16384];
    {/* 打开设备文件并设置 CGA 图形模式 */
    cgafil=open(/dev/cga,2);
/* 清屏 */
    ioctl(cgafil,' a' ,o);
/* 设置画点所需参数 */
    point. x=160;
    point. y=100;
    point. color=3;
/* 在屏幕坐标(160,100)处画一个点 */
    ioctl(cgafil,' b' ,&point);
/* 显示缓冲区读操作 */
    read(cgafil,&img-bnf,sizeof(img-buf));
/* 显示缓冲区写操作 */
    write(cgafil,&img-buf,size of(img-buf);}
```

四、大屏幕彩色图形(图象)显示驱动软件

1. 硬件配置

为提高 XENIX 下图形(图象)显示功能,我们在 386 微机系统上配置了 EVA(Enhanced VGA)扩展卡及 19 英吋大屏幕彩色监示器作为主控台,其最高显示分辨率达 1024×768 象素。显示卡采用 ET3000—AP/AF 视频图形控制 VLST 芯片,除在 I/O 寄存器上与 IBM VGA/EGA 模式全兼容外,支持 800×600 分辨率 256 色,1024×768,16 色,多窗口特点包括:分裂屏幕(Split screen)可作为第 2 窗口,硬件开窗口放大(Zoom window)可作为第三窗口,以及垂直方向滚动(Rolling)和水平方向移动(Panning)等一系列硬件功能,卡上还配备有支持 DOS 系统的 EVA/1024 ROM BIOS[4][5]。

2. 驱动软件结构

为在 XENIX 下驱动大屏幕彩色图形(图象)显示,我们开发了针对 EVA 卡的显示软件,其主要功能可参考《大屏幕显示驱动软件功能一览表》,这个软件完全弥补了 XENIX 目前版本这方面的不足,使得所有 DOS 下能够实现或者卡上硬件提供的功能在 XENIX 下都能应用,使用效果很好。

大屏幕显示驱动软件功能一览表

组成部份	程 序 与 功 能	说 明
显示驱动程序	××open,××close:打开和关闭设备 I/O 文件 ××read,××write:用户数据区与显示缓冲区交换 ××oictl:显示卡 I/O 寄存器设置,显示模式设置,选择显示缓冲区首址,画点,清屏	属于驱动程序例程,并且安装在系统内核之中
显示子程序库	设置模式子程序,清屏子程序 画点与线子程序,画圆和椭圆子程序 画矩形子程序,区域填色子程序 显示(16×16 或 24×24)彩色汉字子程序(字形大小可变) 显示专用图形字符子程序 显示缓冲区读/写子程序,I/O 端口直接访问子程序	按 XENIX 系统函数的形式建立和使用
专用显示子程序	图象存取程序:可进行显示缓冲区或磁盘之间的整幅图象数据拷贝 多窗口程序:使用显示卡硬件功能,包括硬件放大(Zoom)和开窗漫游显示等 动画显示程序:可按一定的分辨率进行 4～16 幅图形图象高速动画显示	

此软件划分为三个层次(请参见图 2),其中最低层即为安装在 XENIX 内核的设备驱动程序,它包含了基本的设备 I/O 驱动功能,如显示模式设置,显示缓冲区读/写操作,I/O 端口访问,各种分辨率下画点等等;而中间一层则作为建立在显示驱动程序之上的应用子程序库,它综合实现了一些基本、常用的图形(图象)显示功能,并按 XENIX 库函数的形式安放在系统中,供用户程序连接调用,为协调驱动程序的调用,我们这里使用了一个信号灯,以保证各种调用同步进行。

第三个层次,也就是最高层次,是用户的程序或一些有用的图形(图象)实用程序,开发这些实用程序的目的是为用户提供一些专用高效工具,如高速动画,高速缓存,多窗口放大,漫游显示等等。

3. DOS 显示程序的移植

EVA 显示卡的 I/O 控制寄存器多达几十个[4],并且对于各种显示模式设置,这些寄存器的值以及访问顺序有很大差别,一般地讲,用户手里的资料是很有限的,在实际的程序设计中,一个寄存器设置不对,功能就无法实现,程序中的错误也很难查出。

对于这样一个工作量大且复杂的问题,我们的对策是解剖 EVA 卡支持 DOS 的 EVA ROM BIOS 系统,此 BIOS 系统是固化在显示卡 ROM 中,由设置各种图形显示模式和画点、线等一系列基本图形显示子程序构成,并可根据寄存器的值使用 DOS INT10H 调用,下面我们以设置图形模式为例,说明移植处理方法, 因设置模式功能主要涉及访问显

示卡上的各种 I/O 寄存器,具体移植步骤描述如下:

(1)在 DOS 下设计一个调用 BIOS 实现设置某种图形显示模式(如 1024×768 分辨率)的汇编子程序。

(2)使用 DOS 下的 debug 调用该程序,并按单步方式执行,将进入 ROM BIOS 后所执行的指令存入一个中间文件 tmp. txt。

(3)设计一个识别程序,去自动"识别"tmp. txt 文件中的指令,若为 IN 或 OUT(输入或输出)指令,则将其按顺序存入结果文件 result. txt。

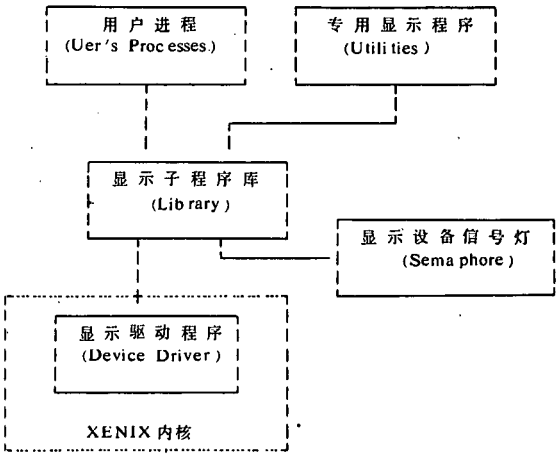

图 2　显示软件结构示意图

(4)调整 debug 中的指令地址,转入(2),继续下一条指令的识别。

上述过程是利用 DOS 下的批处理、管道操作等所构成的自动处理过程,直至所需移植功能执行结束,最后所形成的 result. txt 文件就是设置该模式所需的全部 IN 和 OUT 指令。

接下来可利用编辑程序,将 DOS 下 I/O 指令 IN 和 OUT 改成 XENIX 内核支持的 INB 和 OUTB 例程,并将循环操作用循环语句代替,移植到 XENIX 中去,作为驱动程序对应功能的 I/O 端口访问部份。

对于卡上 ROM BIOS 中没有支持的一些硬件特点的功能,如多窗口放大,水平和垂直移动等,则可通过解剖 DOS 下的一些表演程序,方法与上述类似,这些做法非常成功,在技术资料不全的情况下大大缩短了驱动程序的开发周期,提高了工作效率。

五、应用效果

我们开发的大屏幕显示驱动程序以及由此所构成的气象部门雷达和卫星图象显示软件系统已经于 1990 年 4 月份投入试运行,并在 1990 年 6 月 5 日—8 月 15 日汛期连续运行,为短期天气预报和科研工作积累了大量资料,通过对雷达和卫星图象的各种拼图、放大、叠套显示、加等值线、动画等操作,揭示了很多有意义的天气现象。以下列出了部分图象显示产品(图 3—8),供参考。

随着 XENIX 系统的应用广泛的发展,计算机生产厂家也会不断地推出支持新设备的驱动程序或软件包,但是新的外设总会不断的涌现,而用户的特殊要求也会不断提出。所以上述所介绍的方法为提高 XENIX I/O 能力以及实现用户特色的 I/O 要求提供了一种完全可行的途径。

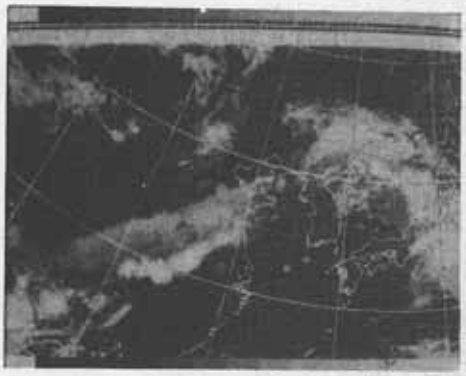

图3 GMS低分辨红外云图(640×480模式)

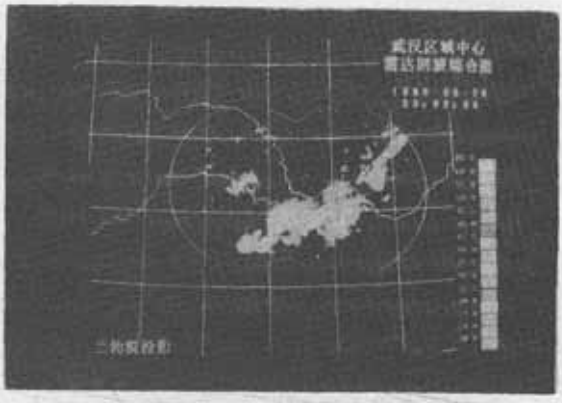

图4 武汉 宜昌雷达拼图(640×480模式)

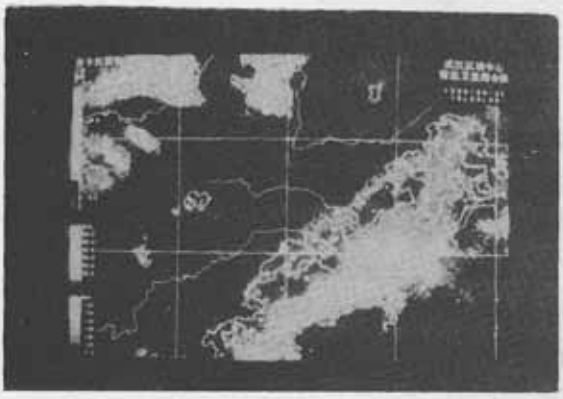

图5 红外云图叠加可见光等值线(1024×768模式)

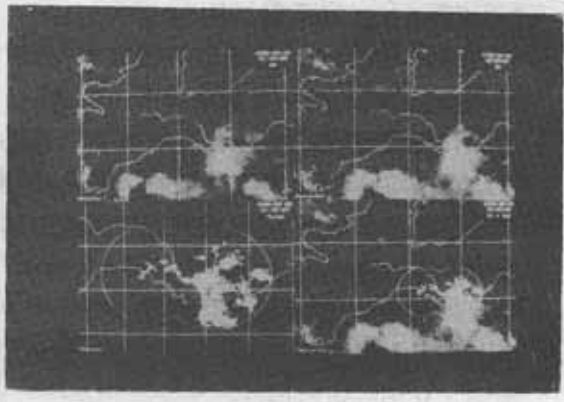

图6 雷达+红外云图叠套显示(1024×768模式)

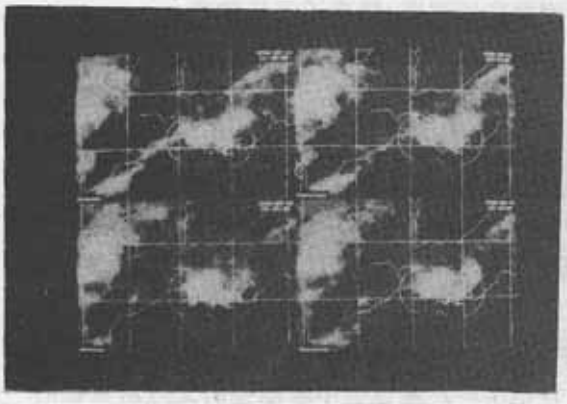

图7 可见光+红外+雷达拼图+雷达与红外叠套等四种资料综合显示(1024×768模式)

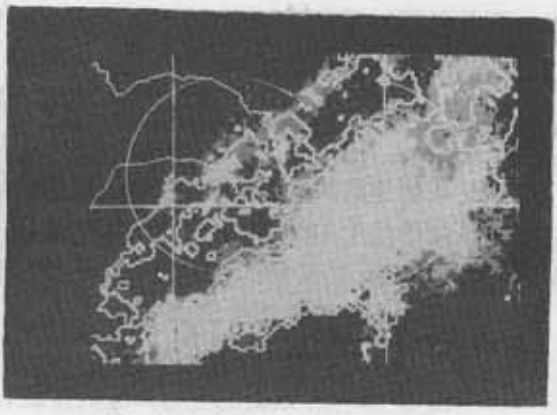

图8 雷达+红外云图+可见光等值线叠套并放大显示(1024×768模式)

致谢:感谢武汉中心气象台金鸿祥高级工程师的技术总负责和工作指导,以及中国科学院软件研究所李有志老师技术指导和咨询。

参 考 文 献

[1] The Santa Cruz Operation Inc. ,The SCO CGI Development System Version 1. 0,1987.
[2] 朱风明、徐少桐、沈　旭,XENIX 系统 V 中 C 语言作图方法,《计算机世界月刊》,第 3 期,1990 年。
[3] The Santa Cruz Operation Inc. ,XENIX System V Development System C user' s Guide,1987.
[4] Tseng Laboratories Inc. ,EVA Technical Handbook,1988.
[5] Tseng Laboratories Inc. ,EVA/1024 BIOS Guide,1987.

附录:简单显示驱动程序例子

```
/* 插入文件 */
#include<sys/types.h>
#include<sys/param.h>
#include<sysmacros.h>
#include<sys/dir.h>
#include<signal.h>
#include<sys/mmu.h>
#include<sys/file.h>
#include<sys/seg.h>
#include<sys/page.h>
#include<sys/user.h>
#include<errno.h>
/* 定义有关的数据 */
#define INDEX 0x3d4
#define DATA 0x3d5
#define MODE 0x3d8
#define COLOR 0x3d9
#define SCN—GRF 0x0a
#define SIZE 0x4000
#define FLAG—R 0
#define FLAG—W 1
/* 设置 CGA(320 * 200)图形模式所需数据 */
char grgraf—mode[]={0x38,0x28,0x2d,0x0a,0x7f,0x06,0x64,0x70,0x02,0x01,0x06,0x07,0x00,0x00};
/* Video RAM 启始地址指针 */
extern B8000—sel;
/* OPEN 例程 */
gropen ()
{
register int i;
/* 设置 CGA 图形模式 */
for(i=0;i<0x0d;i++)
    {
    outb(INDEX,i);
    outb(DATA,grgraf—mode[i]);
    }
outb(MODE,SCN—GRF);
}
/* CLOSE 例程 */
grclose()
{
}
```

```
/* READ 例程 */
grread()
{
gr—rw(FREAD);
}
{
struct {
     unsigned short x;
     unsigned char y;
     unsigned char color;
     } point;
int tmp;
unsigned char mask;
switch(cmd)
     {
     case('a'):/* 清屏功能 */
       grclear(B8000—sel,SIZE);
       break;
     case('b'):/* 画点功能 */
       /* 使用内核例程 copyio()实现画点参数传递 */
       copyio(ktop(&point),arg,sizeof(point),U—WUD);
       tmp=(point.y & 1)? 0x2000 : 0;
       tmp+=(point.y)>>1) * 80+(point.x)>>2);
       mask=~(0x03<<(((point.x & 0x03)<<1);
       point.color<<=((point.x & 0x03)<<1);
        * (faddr—t)(B8000—sel+tmp)=
( * (faddr—t)(B8000—sel+tmp)& mask)|point.color;
       break;
     }
}
/* WRITE 例程 */
grwrite()
{
gr—rw(FWRITE);
}
/* R/W 子程序 */
gr—rw(mode)
{
int count;
/* 使用提供的用户数据区首地 u.u—base */
/* 使用 u area 提供的 I/O 数据容量参数 u.u-count */
if(SIZE<u.u—count)
     count=SIZE;
else
     count=u.u—count;
if(mode==FREAD)
     grcopy(B8000—sel,u.u—base,count,FLAG—R);
else
     grcopy(u.u—base,B8000—sel,count,FLAG—W);
u.u—count—=count;
```

```
}
/ * IOCTL 例程 * /
grioctl(dev,cmd,arg,mode)
int dev,cmd,mode;
faddr—t arg;
/ * 数据块 I/O 子程序 * /
grcopy(from,to,count,flag)
register faddr—t from;
int count,flag;
{
    if(! flag)
      copyout(from,to,count);
    else
      copyin(from,to,count);
}
grclear(to,count)
register faddr—t to;
int count;
{
    do{
      * to++='\0';
    }while(--count);
}
```

DEVICE DRIVER DESIGN AND HIGH RESOLUTION GRAPHICS DISPLAY DEVELOPMENT IN XENIX/386 SYSTEM

Chen Shaolin　　Luo Jianguo

(Wuhan Central Meteorological Observatory)

Abstract

This paper through an example deals with the important concepts and software techniques of the device driver designing in the XENIX/386 operating system on microcomputer, and introduces some ideas for development of high resolution color graphics (or images) display. These research results overcome deficiency of the current version of the XENIX/386. In addition, it provides an efficient way for users to develop the XENIX to support new peripheral devices.