

Xenix 에서의 이미지 서브 시스템의 구현에 관한 연구

김 원, 박 용진

(한양대학교 전자공학과)

(A Study on Development of Image Subsystem on Xenix)

Weon Kim, Yong-jin Park
Dept. of Electronic Eng. Hanyang Univ.

ABSTRACT

This paper proposes the model of Image Display on the Xenix System V and describes the method of writing device driver which can drive the graphics hardware and image scanner. The interface between user program and device driver in Kernel is also presented.

디스플레이 할수있는 기능을 추가할 필요성은 자명하며 또한 시급한 과제이다. 본 논문에서는 IBM PC/AT 상에서 실행되는 SCO(Santa Cruze Operation)의 Xenix Sytem V 의 커널 내에 이미지 디스플레이를 위한 이미지 서브시스템을 설계하고 구현한 결과 및 아울러 일반 사용자와 커널 내의 이미지 서브시스템간의 인터페이스를 수행한 결과를 기술한다.

1. 서론

현재 퍼스널 컴퓨터 상의 UNIX 호환의 오퍼레이팅 시스템을 이식한 제품이 여러종류 발표 되어 가고 있다. 특히 IBM PC/AT 상에서는 종래의 UNIX Sytem V 를 개량한 Xenix 또는 System V/AT 등이 이미 발표되어 널리 사용되고 있으며 국내에서도 퍼스널 컴퓨터에서 UNIX를 사용하는 것이 보편화되고 있다.

이와같은 추세는 현재의 퍼스널 컴퓨터가 앞으로의 퍼스널 워크스테이션으로 발전할 전망을 보다 확실하게 비추어 주고 있다. 또한 UNIX 오퍼레이팅 시스템의 다양한 네트워크 유틸리티의 개발과 보급으로 컴퓨터 네트워크에 연결된 다른 컴퓨터 상의 사용자에게 자신의 메시지를 전달하는 것이 가능하게 되었다. 그럼에도 불구하고 이러한 대부분의 오퍼레이팅 시스템들은 그래픽과 이미지를 디스플레이 할수있는 유틸리티를 제공하고 있지않는 것이 일반적이며 따라서 이는 퍼스널 컴퓨터에서 텍스트 이외의 이미지를 처리하는데 있어서 장애가 되고 있다.

2. 이미지 디스플레이를 위한 모델

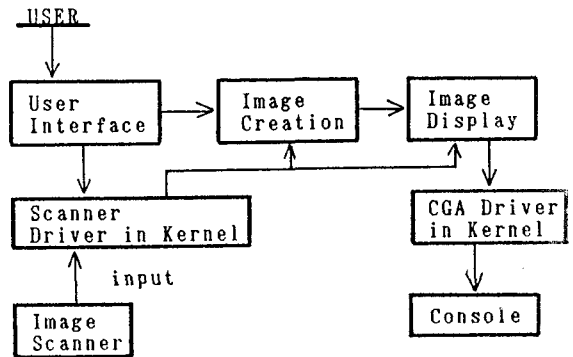


그림 1. 이미지 디스플레이를 위한 모델

그림 1 은 이미지 디스플레이를 위한 모델 [1], [2] 로서 일반 사용자가 이미지를 화면에 디스플레이 하기 위해서는 이미지 스캐너 [3] 를 구동하는 커널 내의 디바이스 드라이버와 이미지 데이터를 화면에 디스플레이 하기 위한 CGA(Color Graphics Adapter) 디바이스 드라이버로 구성된다. 그리고 사용자의 요구에 따라 유저 인터페이스의 처리를 통해 이미지 스캐너로 부터 이미지를 입력하여 화일로 저장한 이미지 크리에이션(Image Creation) 과 화면에 직접

따라서 이러한 오퍼레이팅 시스템 상에서 이미지를

디스플레이 하는 이미지 디스플레이(Image Display)로 구성된다.

그래픽 드라이버는 커널 내에 위치하며 CGA를 텍스트에서 그래픽 모우드로 전환한 후 각 픽셀에 해당하는 메모리 상의 디스플레이 버퍼의 비트를 세트 또는 리세트함으로써 이미지를 디스플레이 한다.

3. Xenix Sytem V 의 커널

Xenix System V(이하 Xenix)의 커널 내에 위치하는 디바이스 드라이버는 그림 2에서 보는 바와 같이 커널과 I/O 디바이스간의 인터페이스를 담당한다 [4], [5].

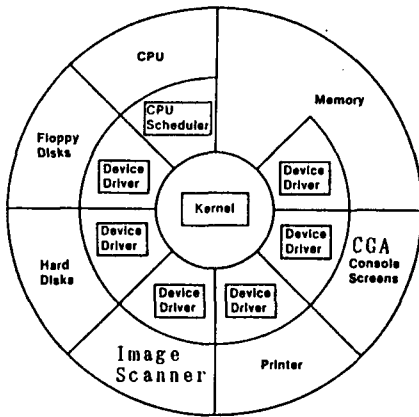


그림 2. 커널과 디바이스 드라이버

Xenix는 유저 어드레스 공간과 커널 프로그램 및 데이터 스트럭처가 80286 CPU의 구조와 밀접하게 관련되어 있다 [6]. 80286 CPU는 두가지의 실행 모우드를 가지고 있다. 하나는 리얼 모우드(real mode)로서 CPU에서 발생된 어드레스는 바로 실제 메모리의 어드레스 즉, 물리 어드레스(physical address)가 되는 경우이고 다른 하나는 프로텍트 모우드(protected mode)로서 이 경우 CPU가 발생하는 어드레스는 가상 어드레스(virtual address)가 된다. Xenix는 프로텍트 모우드에서 실행된다.

프로텍트 모우드의 경우, CPU에서 발생된 가상 어드레스는 두개의 디스크립터 테이블에 의해서 물리 어드레스로 전환된다. 글로벌 디스크립터 테이블(Global Descriptor Table:GDT)과 로컬 디스크립터 테이블(Local Descriptor Table:LDT)이 바로

그것이다. GDT는 전체 메모리 상의 64K 바이트 단위의 세그먼트의 위치 및 각 세그먼트에 대한 액세스의 제한 등에 관한 내용을 담고있다. LDT는 Xenix의 각 프로세스의 텍스트 세그먼트, 데이터 세그먼트, 그리고 스택 세그먼트 등의 메모리 상의 위치 및 이에 대한 액세스 관계에 대한 내용을 담고 있다. 이 두 테이블은 80286 CPU가 메모리를 액세스 하는 데에 개입하게 된다.

본 연구에서 이미지 스캐너로 부터 읽은 이미지 데이터를 CGA에 디스플레이 하기 위해서는 물리 어드레스 B8000~BBFFF(Hexa)를 디스플레이 버퍼로서 사용한다. 따라서 이미지를 화면에 나타내기 위해서는 이러한 메모리 내의 디스플레이 버퍼와 6845 컨트롤 레지스터를 액세스 할 수 있어야 한다. 80286 CPU는 I/O 맵드 I/O(I/O mapped I/O)를 채용하고 있어서 커널 내의 프로그램이 I/O 포트인 컨트롤 레지스터를 액세스 하는데 문제가 없으며 또한 Xenix는 커널 내의 루틴으로서 I/O 포트에 한 바이트 또는 한 워드(2 바이트)를 입출력하는 C 서버루틴을, 제공하고 있다. 하지만 디스플레이 버퍼는 메인 메모리의 한부분으로서 CPU에서 발생시킨 가상 어드레스로는 이를 액세스 할수 없으며 이러한 문제를 해결하기 위해서는 GDT에 디스플레이 버퍼에 대한 엔트리를 삽입하여야 한다.

4. 이미지 디스플레이를 위한 디바이스 드라이버

4.1 구현된 시스템의 환경

본 연구에서 사용된 하드웨어 및 소프트웨어는 표 1과 같다.

표 1. 사용된 하드웨어 및 소프트웨어

컴퓨터	IBM PC/AT
OS	Xenix System V
그래픽 보드	Color Graphics Adapter
이미지 스캐너	NEC PC-IN502
메인 메모리	3.2 메가 바이트

4.2 이미지 디바이스 드라이버

디바이스 드라이버는 커널 내의 프로그램으로서 특정한 하드웨어를 구동하기 위해서 작성되는 프로그램이다. 본 연구에서 작성한 디바이스 드라이버는

크게 두가지로 나눈다. CGA를 구동하기 위한 디바이스 드라이버와 이미지 스캐너를 구동하는 디바이스 드라이버가 바로 그것이다.

CGA를 구동하기 위해서는 우선 6845의 사양에 의거한 모우드의 전환 [7], [8], 즉 6845의 컨트롤 레지스터 조작과 디스플레이 버퍼 역세스의 두가지 문제가 우선 해결되어야 한다. 텍스트 디스플레이 모우드에 있는 6845를 그래픽 모우드로 전환 한다는지 그래픽 모우드에서 다시 텍스트 모우드로 전환하는 것은 커널의 서브루틴 out(), outb()등을 사용하여 6845의 컨트롤 레지스터에 특정 값의 데이터를 기입하면 된다 [9]. 디스플레이 버퍼에 대한 역세스는 메모리 역세시므로 CPU에서 발생하는 가상 어드레스를 물리 어드레스로 전환하기 위해서 GDT 엔트리를 추가하여야 한다. 이를 위하여 커널 내에서 제공하는 서브루틴으로서 dscralloc() 과 dscrfree() 가 있다. 예로서 CGA의 디스플레이 버퍼를 GDT에 삽입하기 위해서는 dscralloc() 으로 GDT 슬롯을 하나 할당하고 여기에 CGA를 사용할수 있는 각종 인자들을 mmudescr() 에 추가하여 초기화 시키면 된다. 다음 코드는 위의 실행을 보여 준다.

```
main()
{
  unsigned short selector = dscralloc();
  mmudescr(selector, 0xb8000, 0x3fff, 0x92);
  :
  :
  dscrfree(selector);
}
```

이미지 스캐너를 구동하기 위해서는 우선 8250 UART의 초기화가 필요하다 [10]. 이때 초기화 파라미터는 이미지 스캐너의 사양에 따른다. 디바이스 드라이버 루틴은 유저와의 인터페이스를 고려하여 기능별로 몇 개의 그룹으로 나뉜다. 본 연구에서는 캐릭터 디바이스로 가정하였다. 캐릭터 디바이스 드라이브의 루틴은 open, close, read, write 그리고 ioctl의 다섯가지 이다. 이중에 ioctl은 사용하지 않았으며 이미지 스캐너 드라이브를 위해 사용한 루틴은 편의상 imgopen, imgclose, imgread, 그리고 imgwrite로 가정 하였다.

imgopen()은 8250의 초기화, 6845의 모우드 전환, 그리고 GDT 엔트리의 할당 및 초기화 등의 기능을

한다. imgclose()은 8250의 파라미터 복귀, 6845의 텍스트 모우드의 복귀, 그리고 할당된 GDT 엔트리의 원상 복귀 등의 기능을 제공한다. imgwrite()은 실제적으로 이미지 스캐너로 부터 이미지 데이터를 읽어, 그래픽 모우드로 변환된 CGA 버퍼에 한 픽셀 씩 세트 또는 리세트하여 이때 화면에서 이미지를 볼 수 있게 한다. imgread()은 CGA 버퍼에 존재하는 데이터를 파일로 저장하는 기능을 담당 한다. 이때 사용되는 커널의 서브루틴은 다음과 같다.

```
imgread(dev)
int dev;
{
  char data;
  :
  while ( passc(data) == 0 )
  :
}
```

위의 커널 내의 프로그램들 중 이미지 스캐너로 부터 데이터를 읽는 부분은 속도들 요함으로 기계어로 작성되었으며, 그 외의 부분은 C로 작성되었다.

5. 유저와 디바이스 드라이버 간의 인터페이스

유저와 디바이스 드라이버간의 인터페이스는 스페셜 디바이스 파일(special device file)을 사용하여 이루어 지고 있다. 이미지 스캐너에 관한 드라이버는 캐릭터 디바이스 드라이버로서 작성되었으므로 디바이스 파일 역시 캐릭터 디바이스 파일이다. 이미지 디바이스 드라이버와 인터페이스 하고자 하는 유저 영역의 프로그램은 이 디바이스 파일을 open, close, read, 그리고 write 시스템 콜을 사용하여야 한다.

```
main()
{
  char buffer[];
  file_pointer = open("/dev/img", O_RDWR);
  write(file_pointer, buffer, sizeof(buffer));
  :
  :
}
```

위의 유저 프로그램과 같이 시스템 콜을 하면 buffer의 내용이 스트럭처 구분인 u-area에 격납되고 그대로 커널 내의 디바이스 드라이버에 전달된다. 커널 내의 디바이스 드라이버는 커널 루틴인 cpass(), passc() 등을 사용하여 u-area 내의 인자 들을 읽을 수 있다. 유저 프로그램이 디바이스 파일을 open 하고 여기에 write 시스템 콜을 수행하는 경우 디바이스 드라이버의

imgwrite() 루틴은 u-area를 통하여 전달되어 온 인자를 해독하여 해당되는 루틴을 호출하게 된다.

6. 결 론

본 연구는 Xenix 상에서 이미지 디스플레이를 위한 이미지 서브 시스템을 구현하였다. 즉 CGA를 텍스트 모우드에서 그래픽 모우드로 전환하여 이미지 스캐너로 부터 입은 데이터를 화면에 디스플레이 할 수 있었다. 그러나 현재 구현된 시스템에서는 이미지 데이터를 축소, 확대하여 디스플레이 하는 기능이 없다. 또한 앞으로 텍스트, 그래픽 그리고 이미지를 혼합한 문서를 다룰 수 있는 다중 매체 에디터가 필요하다. 이는 앞으로 장차 해결되어야 할 문제이다.

참고 문헌

- [1] F.R.A Hopgood, R.J.Hubbold, and D.A.Duce, "Advances in Computer Graphics II", Erographic Seminars, Springer-Verlag, pp.153-186, 1986.
- [2] C.Bathe, and H.P.Godberson, "The INCA Workstation", Proceedings of the 4th Annual ESPRIT Conference Brussels, North-Holland, pp.1459-1471, Sep. 1987.
- [3] NEC, PC-IN502 Image Scanner - User's Guide
- [4] Christopher L. Morgan, Inside XENIX, Howard W. Sams &Co., pp.227-268, 1986.
- [5] Maurice J. Bach, The design of the UNIX Operating Sytem, Prentice Hall, pp.312-354, 1986.
- [6] Edmund Strauss, Inside the 80286, Prentice Hall, pp.77-91, 1986.
- [7] IBM, "IBM Color/Graphics Monitor Adapter", Technical Reference Personal Computer XT, pp.1-123~1-147, 1983.
- [8] Chul-min Lim and Yong-jin Park, "Design and Implementation of Graphics System on Xenix", KISS Conference Proceeding, Vol. 14, Sep. 1987.
- [9] The Santa Cruz Operation, The XENIX V Development System: Programmer's Guide System Development Tools, pp.9-1~10-28, 1986.
- [10] IBM, "IBM Asynchronous Communication Adapter", Technical Reference Personal Computer XT, pp.1-215~1-243, 1983.