

1장. 개요 (Overview)

6년 전, 가상 디바이스 드라이버(virtual device driver)를 만들 때 이런 책이 정말 필요했었다. 윈도우즈 3.0이 막 출시되었을 때, 필자는 윈도우즈의 시작과 종료에 때를 맞추어 동작하는 도스 확장자에 가까운 TSR 프로그램을 작성하는 업무를 맡고 있었다. 이 문제를 풀기 위해서는 가상 디바이스 드라이버(VxD) 작성이 필요했지만, 구할 수 있는 문서는 고작 가상 디바이스 드라이버 설치 안내서(Virtual Device Driver Adaptation Guide) 베타 초안의 복사본 밖에 없었다. 그래서 정작 필자가 원했던 VxD의 디자인법, 작성법, 디버깅 방법에 대해서 아무 도움이 되지 못했다. 어쨌든 그 프로젝트와 씨름했고, 그렇게 하는 동안 항상, 이렇게 어렵게 배운 모든 것을 누가 벌써 써 놓았을 것이라고 생각했다.

다른 프로그래머 역시 그들이 맡은 일을 끝낼 정도로 완벽히 윈도우즈 95의 시스템 인터페이스를 이해하기 못하고 있다는 것을 알게 되었다. 지난 많은 해를 컴퓨서브(CompuServe)의 WINSDK 포럼에서 많은 시간을 보냈다. 여기서 만난 사람들은 옛날 내가 궁금하게 생각했던 질문과 같은 기본적인 질문을 계속 해대는 것이었다. 래리 닉벤(Larry Niven)과 제리 포넬(Jerry Pournelle)은 *신의 눈속에 든 티끌(The Mote in God's Eye, Simon and Schuster, 1974)*이란 책에서, 외계 종족 같이 엔지니어가 실제적인 문제를 푸는데 있어 유별나게 잘하고, 대화에 있어 유별나게 못하는 양태에 대하여 썼다. 가끔 마이크로소프트가 이런 티끌 같은 엔지니어에 의해 책을 출판해 온 것이 아닌가하고 생각하며, 그들도 아마 이 말을 공경할 것이다.

프로그래머가 그들의 일을 잘 수행할 수 없을 때 사용자는 고생하게 된다. 마이크로소프트와 윈도우즈는 비난받을 것을 잘 되새겨 보아야 한다. 왜냐하면 사용자들은 어떤 버그에 대하여 이것이 운영체제가 나쁜 것인지(그렇지 않다), 소프트웨어 프로그래머가 나쁜 것인지(그렇지 않다), 불안전하고 생략 적인 문서 때문인지(음, 맞다) 말할 수 없기 때문이다. 따라서 필자는 이 책이 불완전하지 않도록 하기 위해 최선을 다했으며, 무엇을 하지 말라고 희미하게 말하는 대신, 무엇을 해야하고 왜 해야하는지를 말하기 위해 최선을 다했다. 마이크로소프트 출판사는 필자에게 내부 정보에 대하여 특별한 접근을 허용하지 않았다. 특히 필자는 윈도우즈 95의 소스 코드를 보길 원했는데, 왜냐하면 마이크로소프트는 우리가 그 운영체제를 사용하는 방법을 잘 알기를 원할 것이라고 생각했으며, 그렇다면 이에 대한 정확한 정보를 알려주기 위한 것이었다. 그러나 필자의 경우 홈페이지는 고사하고 1루도 밟지 못했다. 필자의 최선과 이 책을 읽는 많은 개발자들의 관심으로 우연히 범한 어떤 실수를 여기에서 발견해 주면 기쁘겠다.

1.1 누가 이 책을 읽어야 하나

(Who Should Read This Book)

만약 일을 하기 위해 윈도우즈 95의 저 수준 운영체제 기능의 사용법을 배우길 원한다면 이 책을 읽어야 한다. 이 책은 다음과 같은 일에 도움이 될 것이다.

- 새로운 하드웨어에 대한 디바이스 드라이버를 작성하는 것. 윈도우즈 95는 하드웨어 제조업자가 MS-DOS 리얼모드 드라이버나 16비트 윈도우즈 DLL 드라이버를 제공하는 것보다는 32비트 보호모드 드라이버를 제공할 때 더 잘 작동한다. 이 책은 윈도우즈 95의 시스템 아키텍처에 알맞은 드라이버 작성법을 알려 줄 것이다.
- 디바이스 드라이버를 업그레이드하는 것. 만약, 윈도우즈 3.x 가상 디바이스 드라이버 작성을 시작했다면, 깊이 들어가지 않도록 해야 할 것이다. 플러그 앤 플레이에 관계된 내용(11장, 12장)을 읽기 바란다. 현재 코드에 플러그 앤 플레이를 덧붙여 넣거나 말거나 꼭 보길 바란다. 그러나 16비트 드라이버 작성을 시작했다면 이것은 새로운 드라이버를 작성해야 하는 프로그래머와 한배를 탄 것이다.
- 시스템 툴 지원 서비스를 제공하는 것. 디버거, 시스템 모니터, 고장 해결 툴, 그 외 마이크로소프트가 제공하지 않는 운영체제 지원 툴 등등. 이에 대한 가상 디바이스 드라이버를 작성하는 것은 문서화된 인터페이스보다는 문서화되지 않는 뒷구멍을 이용하여 운영체제의 저 수준으로 접근해야 한다.

- 내부적으로 윈도우즈가 어떻게 동작하는지 문서화되지 않은 것. 윈도우즈 95가 파일 운영을 수행하는 INT 21h 인터페이스에 확장이 있다는 것을 읽었을 것이다. 어떤 한 시스템에서 호출하는 INT 21h 파일 시스템은 다른 곳에 있는 네트워크상의 다른 컴퓨터의 가상 디바이스 드라이버에 의해 서비스 될 수 있다는 것을 알고 있는가? 이 책은 그 프로세스 하부의 신비스러움과 또 다른 윈도우즈 95 프로세스를 설명한다.

시스템 프로그래밍은 필연적으로 하드웨어와 어셈블리어에 대한 자세한 지식을 필요로 한다. 그러나 여기서는 이 책을 읽는데 필요한 지식의 양을 최소로 하려했다. 아마 이미 하고 있는 일을 통해 이 영역에 대해 이해하고 있을지도 모르겠지만, 그것 이상으로 인텔 x86 프로세서의 어셈블리어로 작성된 프로그램을 읽을 수 있어야 한다. 비록 그런 프로그램을 작성할 필요가 없어도 말이다. 또한 이 책의 예제는 대부분 C나 C++로 되어 있기 때문에 C/C++ 프로그래밍 언어의 사용에 능숙해야 한다.

1.2 이 책은 어떻게 구성되었는가.

(How This Book Is Organized)

이 책은 4부로 되어 있다. 1장부터 5장으로 구성된 1부는 윈도우즈 95 시스템의 아키텍처와 윈도우즈 95 시스템 프로그래밍에 포함된 중요한 개념의 개요를 나타내었다.

- 1장은 이 책의 전체적인 구성을 나타내었으며, 이 책과 부록 CD에서 필요한 것을 찾는 방법을 써 놓았다.
- 2장은 모든 마이크로소프트 운영체제로서의 윈도우즈 95와 다양한 디바이스 드라이버를 설명한다.
- 3장은 시스템 프로그래머의 시각으로 본 윈도우즈 95의 구조를 종합적으로 설명한다.
- 4장은 가상 머신 관리자(Virtual Machine Manager; VMM)와 기타 윈도우즈 95 운영체제의 중요 요소들과의 상호 작용을 사용한 프로그래밍 인터페이스를 설명한다.
- 5장은 윈도우즈 95가 실행되는 인텔 32비트 프로세서의 특별한 구성을 종합적으로 설명한다. 이 장은 독자들이 벌써 16비트 프로그래밍에 대해 많이 알고 있다고 가정한다. 만약 그렇지 않다면 이 장을 그냥 넘기고 싶을지도 모르겠다. 그러나 이러한 것에 대한 이해가 없더라도 이 책의 나머지 부분이 여기에 대해 충분히 설명해 줄 것이다. (역자 주 : 꼭 그렇지 않다는 것을 명심하기 바란다)

이 책의 2부는 가상 디바이스 드라이버 프로그래밍의 필수 요소에 대한 내용을 담고 있다. 독자들이 계획하고 있는 시스템 프로그래밍 프로젝트가 무엇이든 간에 다음의 내용에 대해 알고 있어야 할 것이다.

- 6장은 가상 머신 관리자가 메모리와 프로세서 시간을 어떻게 관리하는지 설명한다.
- 7장은 어셈블리어나 C/C++로 가상 디바이스 드라이버를 만드는 방법에 대하여 자세히 설명한다. 마이크로소프트 툴이나 서드파티 툴은 드라이버 작성에 쉽겠다고 선택한 언어가 무엇이든 간에 이를 통합할 수 있는 수준에 도달했다. 그리고 거기에 대한 어려운 점들도 설명한다.
- 8장은 디바이스 드라이버의 로드와 초기화에 대하여 논의한다. 단지 CONFIG.SYS 파일에 device= 이라는 문장만 더해 주면 된다거나 모든 것이 MS-DOS에 의해 관리된다고만 아는 시대는 지나갔다. 윈도우즈 95의 가상 디바이스 드라이버를 정적으로 혹은 동적으로 로드 하는 방법과 초기화하기 위해서는 무엇을 해야 하는가에 대하여 이 장에서 배우게 될 것이다.
- 9장은 가상 디바이스 드라이버 제작자에게 유용한 기술을 담고 있다. 이 장은 한 드라이버에서 다른 드라이버를 호출하는 방법, 메모리를 할당하는 방법, 비동기 콜백을 제공하는 방법, 다른 드라이버 프로그램과 동기를 하는 방법 등을 설명한다.
- 10장은 가상 머신에서 실행되는 프로그램에게 유익한 하드웨어와 소프트웨어를 가상화(virtualize)하는 방법을 설명한다. 이전 버전의 윈도우즈에서 이 가상화가 가상 디바이스 드라이버를 작성하는 주요 이유였다. 또한 이 장에서는 지금까지도 중요한 것으로 남아 있는 기술을 설명한다.

이런 기초 지식을 바탕으로 하여 3부에서는 입출력 프로그래밍을 논의한다. 하드웨어 드라이버를 작성하는

프로그래머에게는 이 장이 제일 관심 있는 부분이 될 것이다.

- 11장과 12장은 구성 관리자(Configuration Manager)와 윈도우즈 95의 플러그 앤 플레이 아키텍처를 종합적으로 설명한다. 이 장을 통해서 윈도우즈 95가 하드웨어를 식별하는 방법, 리소스를 로드하는 방법, 드라이버를 로드하고 초기화하는 방법을 배우길 바란다.
- 13장은 드라이버가 로드 되고 인터럽트, DMA(Direct memory access) 채널이나 혹은 하드웨어에서 사용하는 다른 리소스 등이 확인되고 난 후에는 무엇을 할 수 있는지를 설명한다. 이 장에서는 하드웨어 인터럽트를 처리하는 방법, I/O 포트의 내용을 관리하고 디바이스를 시뮬레이트하기 위해 이를 “트랩”하는 방법, 메모리에 맵핑된 하드웨어를 액세스하는 방법, DMA를 다루는 방법 등을 설명한다.
- 14장은 직렬포트와 병렬포트를 처리하는 마이크로소프트의 VCOMM 드라이버를 가지고 만드는 통신 드라이버를 설명한다. 이 장에서는 C++로 작성된 직렬 포트 드라이버의 좋은 예를 보여줄 것이며, 이를 통해 자기만의 포트 드라이버 만드는 방법을 배우게 될 것이다.
- 15장은 디스크 디바이스나 기타의 “블럭” 디바이스들을 처리하는 입출력 수퍼바이저(Input/Output Supervisor; IOS)에 대하여 논의한다. 이 장은 램 디스크 운용과 IOS를 필요로 하는 모니터 프로그램에서 포트에 기록하는 방법이나 제조업체에서 제공한 드라이버 사용법 등으로 주로 다루었다. 이 두 종류의 블럭 디바이스는 일반 사람들이 가장 많이 사용할 만한 것이다.

마지막으로 4부에서는 윈도우즈 95의 추가적인 구성 요소에 대하여 설명한다. 이것은 시스템 프로그래밍을 하는데 있어 폭넓은 기능을 사용할 수 있도록 해 줄 것이다.

- 16장은 IFSM(Installable File System Manager)를 담고 있는데, 이것은 앞서도 언급했듯이 결국 INT 21h의 여러 서비스가 그 대상이다. 이 장은 로컬 디스크 디바이스에 새로운 파일 시스템을 지원하는 드라이버를 작성하는 방법을 설명한다.
- 17장은 이 책의 마지막 장으로써, DPMS(Dos Protected Mode Interface)를 설명한다. 비록 새로운 프로그램에게는 DPMS가 필요 없다고 하더라도, 소위 말하는 “구식” 응용 프로그램이나 윈도우즈 3.1에 호환 가능하게 하려는 16비트 응용 프로그램에게는 필요할 수 있다.

1.3 샘플 코드에 대하여 (About the Code Samples)

샘플은 설명과 다를 수 있다. 이러한 이유로 이 책에서 완전한 프로그램 리스트를 찾기 힘들 것이다. 대신 일반적인데 초점을 맞추어 예를 든 토막 코드를 사용한다. 그리고 이 책에는 토막 코드가 대단히 많은데, 왜냐하면 이 책을 보는 대부분의 프로그래머가 컴퓨터 언어로 생각하기를 더 좋아할 것이기 때문이다.

이 책에서 긴 프로그램을 올려놓지 않은 또 다른 이유는, 솔직히 이런 코드를 전문적으로 다룬 책이 많이 있다고 생각하기 때문이다. 필자가 말하는 이런 종류의 책이란 제작자(글쓰기가 아니라 코드를 만든 사람)가 이러한 코드를 산뜻하게 만들어 놓았으며, 간단히 이들을 책으로 복사한 것이기 때문에 독자에게는 더 적당할 것이다. 물론, 필자도 역시 제작자이며 내 코드도 매우 산뜻하다고 생각하지만, 독자들은 자화자찬하는 필자의 코드보다는 자체적인 프로그램을 만드는데 더 관심이 있으리라 생각한다.

앞에서 그 이유에 대하여 말했음에도 불구하고, 책 내용을 보지 않고 문장 여기저기 널려진 코드 토막을 본다면 허탈한 느낌이 들지 모르겠다. 이러한 이유로 해서 부록 CD가 있다. 부록 CD에는 몇 개의 응용 프로그램과 특정한 컴퓨터에서라면 컴파일하고 실행할 수 있는 매우 많은 샘플이 들어 있다. CD에는 README 파일이 있는데 여기에는 샘플간의 차이점을 설명과 이들을 실행해 볼 수 있는 방법을 설명해 놓았다. 어떤 것은 Vireo Software의 VToolsD를 설명하기 위한 샘플도 있기는 하지만 이 샘플들은 대부분 마이크로소프트 MASM 6.11(이것은 마이크로소프트 윈도우즈 95용 DDK를 위한 특별 버전이다.)과 마이크로소프트 비주얼 C++ 버전 4.0에 적당하다. 이 샘플들은 펜티엄 90, 메모리 64M, ISA 버스, PCI 로컬 버스, 3.5 인치 플로피 디스크 드라이버가 있는 컴퓨터에서 컴파일 되고 테스트되었다.

CD에 있는 나머지 것들은 윈도우즈 95 DDK에 있는 공식적인 WinHelp 파일과 이 항목들에 대한 필자의 설명이 담긴 설명파일이다. 이 설명은 필자가 관찰한 것이나 다른 프로그래머들이 CompuServ에 전자메일로 보내온 것을 기초로 하였다. (다음의 출판 본에 독자들이 넣었으면 하는 것이 있다면 필자의 전자메일 주소가 waltoney@oneysoft.com이니 여기로 메일을 보내주면 된다.)

설명 (Note)

부록 CD에 대하여 좀더 자세한 정보를 원한다면 서문을 보기 바란다.

1.4 문서화되지 않은 것에 대한 프로그래밍 기법

(About All the Undocumented Hacks)

1.5 문서화된 것에 대한 프로그래밍 기법

(About All the Documented Hacks)

위의 절을 빈줄로 남겨 놓은 것은 다음과 같은 점을 말하기 위함이다. 즉, 이 책은 문서화되지 않은 것에 대한 책이 아니라는 것이다. 이 책은 마이크로소프트가 제시하는 시스템 인터페이스를 통해 어떻게 하면 독자들의 일을 마칠 수 있는가에 대한 것이다. 이 인터페이스는 모두 문서화된 것(혹은 실수로 빠뜨리지만 않았다면 문서화되었을 것)이다. 즉, 모르고 있었다면 비난받을 만한 것이다. 필자가 콘트롤 블록을 설명한 부분에서 각 필드를 다

분해해서 설명하고 문서화되지 않은 필드를 언급한 곳은 단지 이 곳 뿐이며 이렇게 하지 않고서는 독자들이 충분히 사용할 정도로 이 인터페이스를 설명할 방법이 없었기 때문이다.

이 책이 마이크로소프트에서 출판했다고 해서 마이크로소프트가 그어 놓은 선에 필자의 발을 맞추고 있다고 상상하지 마라. 누구든지 컬러로 색칠된 라인에 설 수 있다고 생각한다. 마이크로소프트는 윈도우즈가 발전함에 따라 여기에 맞춰 문서화된 인터페이스의 기능도 고쳐야할 의무가 있다. 그 외에도 독자나 필자도 윈도우즈 시스템 프로그래머를 위해 마이크로소프트가 제공한 문서를 읽어야 한다. 만약 가상 통신 디바이스가 직렬포트가 충돌하지 않도록 관리하는 것과 같은 방법으로 윈도우즈 95가 시스템 구성 요소와 밀접하게 연관되어 있다면, 우리는 아마 이 구성 요소들이 오랫동안 매우 안정적인 것이라고 확신할 수 있다. 다른 한편으로 프로그램이 시스템을 속이는 교묘한 방법을 사용하고 있다면 (마이크로소프트는 우리들이 *VxDCall* 인터페이스를 호출하는 것이 힘들도록 그 호출 범위를 축소 시켰음에도 불구하고 KERNEL32.DLL 모듈에서 여전히 *VxDCall* 인터페이스를 호출하고 있는 것), 그 인터페이스가 갑자기 바뀌었을 때 만약 같은 결과를 얻기 위한 문서화된 방법이 있다면 적절한 대처를 할 수 있어서 겉으로 보기에는 항상 같은 것처럼 보이게 된다. (역자 주 : 윈도우즈 95가 출시되고 얼마 되지 않아서 "Unauthorized Windows 95"라는 책을 산 일이 있다. 거기에 보면 우리가 DDK를 작성하지 않고 3순위 권한 Win32 응용 프로그램에서 인터럽트를 사용하는 방법이 설명되고 있었는데, 그 가운데에 *VxDCall*이라는 함수가 있었다. 역자는 수십 번 이 샘플 코드를 컴파일하고 실행해 보았지만 결과는 "잘못된 연산으로 종료"한다는 메시지 뿐이었다. 나중에 알아낸 일이었지만 그 책은 윈도우즈 95의 마지막 베타판에 대한 설명이었으며, 불행히도 마이크로소프트에서 정식버전을 출시하기 전에 이 책을 입수 할 수 있었으며 이 책에 설명된 이렇게 할 수 있는 방법을 막아 버렸다고 한다. 하지만 더 속상한 것은 마이크로소프트 내부적으로는 교묘히 이 함수를 계속 사용하고 있다는 것이다. 확인하고 싶다면 앞에서 필자가 언급한 KERNEL32.DLL을 윈도우즈 탐색기의 간략히 보기로 들여다보아라. 여기에는 *VxDCall*이라는 함수의 흔적이 아주 진하게 남아 있을 것이다.)

필자가 문서화되지 않은 것은 사용하지 말라고 한 또 다른 이유는 스스로가 찾자고 하는 것이다. 필자는 어떤 장애물이나 극히 복잡한 문제는 자기 자신 스스로가 문제를 풀기 위한 적당한 방법을 찾으라는 신호라는 것을 터득했다. 예를 들어, 만약 어떤 일을 하는 리얼모드 프로그램을 실행시킬 수 있게 하기 위해 히든 가상머신을 생성하는 것이 쉽지 않다는 것에 초조해 진다면, 가능하면 내 자신에 질문하는 것이 더 쉽고 더 강인한 방법을 찾을 수 있다는 것을 터득했다는 것이다. (역자 주 : Visual C/C++의 컴파일 시에 이 히든 가상 머신을 만드는 것 같다)

그러나 디바이스 드라이버 프로그래밍을 시작했을 때 모든 것이 장애물이고 모든 것이 복잡한 것처럼 보인다. 필자는 이 책을 통해 지난 몇 해 동안 윈도우즈 시스템 프로그래머를 괴롭혀 온 많은 모호한 것을 명료하게 하기를 희망한다. 필자는 또한 이로써 몇 주나 몇 달간의 헛된 노력이 절약되기를 희망한다.