실시간 운영체제 Q+를 위한 라이브러리 설계 및 구현

김 도 형† · 박 승 민†

요 약

본 논문에서는 정보가전용 실시간 운영체제 Q+에 탑재된 라이브러리의 설계 및 구현에 대해 기술한다. 실시간 운영체제에서의 라이브러리는 표준 인터페이스에 따라 정의되어야 하고, 실시간 운영체제의 응용 분야에 적합한 함수들을 제공하여야 한다. 구현된 Q+ 라이브러리는 응용 프로그램 간의 호환성을 보장하기 위해서, POSIX.1, ISO 7942 GKS 등의 업계 및 국제 표준에 따라 설계되었다. 그리고, Q+ 응용 분야에 적합한 C 표준 함수, 그래픽/윈도우 함수, 네트워크 관련 함수, 보안 지원 함수, 파일 시스템 관련 함수들을 제공한다. Q+ 라이브러리는 Q+ 커널과 디지털 TV용 셋탑박스, 그리고 디버깅 툴인 KBUG를 이용하여 구현되었다.

The Design and Implementation of Library for RTOS Q+

Do-Hyung Kim† · Sung-Min Park†

ABSTRACT

This paper describes the design and implementation of library for real-time operating system Q+, that was developed for the internet appliance. The library in the real-time operating system should be defined according to the standard interface and support the functions that are adequate to the real-time application. To ensure the compatibility between application programs, the Q+ library follows industrial and international standards, such as POSIX.1, ISO 7942 GKS. And, to support the Q+ application, library provides C standard functions, graphic/window functions, network functions, security support functions, file system functions. The Q+ library was implemented using the Q+ kernel, Digital TV set-top box, and KBUG debugging tool.

키워드 : 실시간 운영체제(RTOS), Q+, 라이브러리(Library)

1. 서 론

21세기 정보통신 분야에서 네트워크 기술을 탑재한 정보가전들의 정보 서비스가 우리의 일상 생활에 직접적으로 활용될 것이라는 예견은 이미 실현되고 있다. 현재 가정에서 흔히 사용되고 있는 TV, 냉장고, 전화기, 게임기 등이 지능을 가진 정보 단말기 역할을 수행하여 가정 내 뿐만 아니라 집 밖에서의 생활 양식까지도 바꾸어 놓고 있다. 그리고, 디지털 TV, 인터넷 셋탑박스(set-top box), 인터넷 전화기 등 디지털 및 네트워크 기술을 바탕으로 한 정보가전 제품이 속속 등장하면서 이들 제품의 기능을 제어하는데 필수적인 실시간 운영체제(RTOS) 시장이 크게 성장하고 있다[1-3].

컴퓨터 운영체제 업체들과 기존의 실시간 운영체제 전문업체들은 그 동안 통신, 산업전자, 항공 및 우주산업 분야의 제어시스템을 중심으로 형성되어온 실시간 운용체계 시장이 멀티미디어의 필요성에 따라 정보가전용으로 확대됨에 따라 이를 선점하고자 노력하고 있다. 정보가전 시장을 겨냥해 윈도우CE를 출시한 마이크로소프트는 휴대형 PC, 휴대용 정보단말기(PDA) 등에 윈도우CE를 보급시킨 데 이어 인터넷 셋탑박스, 디지털TV 시장도 선점하려 하고 있다. 선 마이크로시스템 사도 실시간 운영체제 전문업체인 Chorus 사를 인수하고, 자체 개발한 퍼스널자바(pJAVA) 등을 내세워 정보가전용 실시간 운영체제 시장에 도전하고 있다. 그리고, 그 동안 실시간 운영체제 VRTX, VxWorks, pSOS, QNX 등으로 실시간 운영체제 시장을 장악하고 있던 미국의 마이크로텍(Microtec), 윈드리버(WindRiver) 시스템, ISI사 및 캐나다의 QSSL사 등도 그 동안 축적된 노하우를 바탕으로 정보가전 시장을 위한 제품들을 지속적으로 개발하고 있다[7-9]. 이러한 해외 업체들의 활발한 시장 선점 경쟁에도 불구하고, 국내에서는 정보가전용 실시간 운영체제가 개발되지 않고 있으며 외국제품의 수입에 전적으로 의존하고 있는 실정이다[3].

21세기 최대의 황금 시장이 될 정보가전 제품에 필수적인 정보가전용 실시간 운영체제의 개발 필요성에 의해 한국전자

† 정 회 원 : 한국전자통신연구원 컴퓨터소프트웨어기술연구소
논문접수 : 2001년 10월 11일, 심사완료 : 2001년 12월 6일

통신연구원(ETRI)은 삼성전자, 대우 전자, LG 전자등과 공동으로 정보가전용 실시간 운영체제인 Q+(Q플러스)를 개발하였다. 실시간 운영체제 Q+는 커널, 라이브러리, 사용자 개발 도구, 그리고 응용 API 등으로 나누어지고, 라이브러리는 다시 C 표준 라이브러리, 그래픽/윈도우 라이브러리, 네트워크 라이브러리, 보안 지원 라이브러리, 파일 시스템으로 구성된다. 본 논문에서는 Q+에 탑재되는 라이브러리들의 설계 및 구현에 대해 다룬다. 라이브러리는 1차적으로 인터넷 셋탑박스에 탑재되어 디지털 TV용 응용 분야를 목표로 1999년 1월부터 2000년 12월까지 개발되었다.

본 논문은 다음과 같이 구성되어 있다. 2절에서는 라이브러리의 구조 및 기능에 대해 간략히 기술하고, 3절에서는 라이브러리의 개발 환경에 대해서 다룬다. 4절에서는 라이브러리의 구현과 수행 예제, 그리고 특징들을 기술하고, 마지막으로 5절에서는 결론 및 향후 과제에 대해서 다룬다.

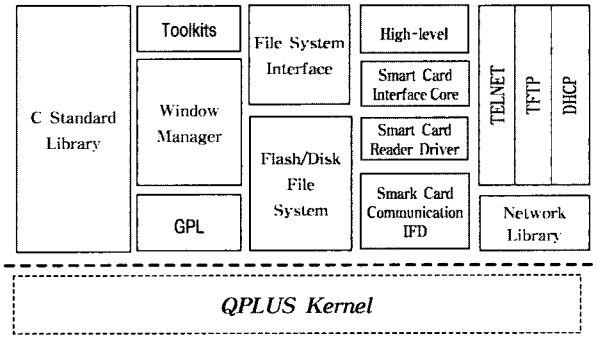
2. 라이브러리 구조 및 기능

이 절에서는 라이브러리들의 구조 및 기능에 대해서 기술한다.

2.1 라이브러리 구조

실시간 운영체제 Q+에 탑재되는 라이브러리는 크게 C 표준 라이브러리, 그래픽/윈도우 라이브러리, 네트워크 라이브러리, 보안 지원 라이브러리, 그리고 파일 시스템으로 구성된다. (그림 1)은 라이브러리의 전체 구조를 보여준다.

(그림 1)에서 C 표준 라이브러리는 응용 프로그램 개발에 공통적으로 사용할 수 있는 함수들을 제공하고, 그래픽/윈도우 라이브러리는 Q+ 응용을 위한 그래픽 프리미티브 라이브러리(GPL)와 윈도우 매니저 등을 구현한다. 네트워크 라이브러리는 네트웍 공용 라이브러리, 텔넷(TELNET), TFTP, DHCP 등을 제공하고, 보안 지원 라이브러리는 스마트카드 관련 함수들을 구현한다. 마지막으로, 파일 시스템은 플래시 파일 시스템과 디스크 파일 시스템, 그리고 파일 시스템 관련 함수들을 제공한다.



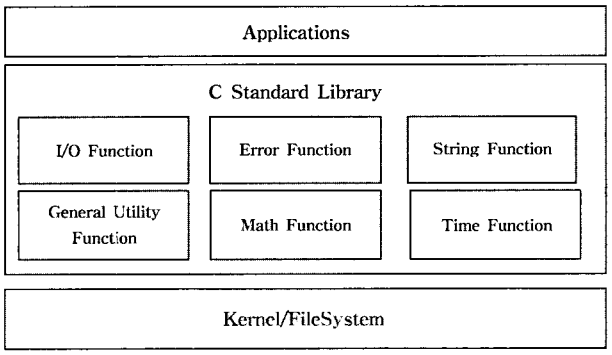
(그림 1) 라이브러리 구조

2.2 라이브러리 기능

이 절에서는 각 라이브러리의 세부 구조 및 기능에 대해 기술한다.

2.2.1 C 표준 라이브러리

C 표준 라이브러리는 POSIX.1[5, 6]에 정의된 표준 인터페이스를 따르도록 설계함으로써 운영체제 간 호환성을 높일 수 있도록 하였고, 표준 인터페이스 중에서 Q+ 응용 분야에 적합한 함수들을 선택하여 라이브러리를 경량화 하였다. 그리고, 응용 프로그램들의 반응 시간을 증가시키기 위해 쓰레드들이 라이브러리 함수를 공유할 수 있도록 재진입 가능하도록 구현되었다. (그림 2)는 C 표준 라이브러리의 구조를 보여준다.



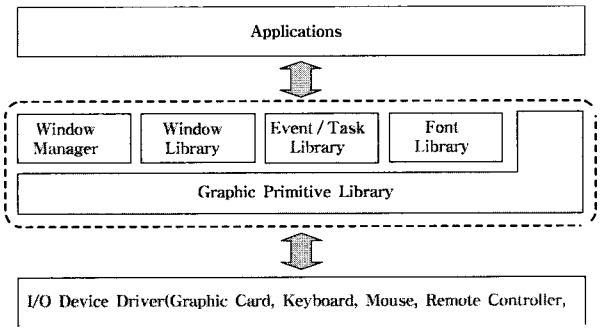
(그림 2) C 표준 라이브러리 구조

(그림 2)에서 보듯이 C 표준 라이브러리는 크게 입출력 함수, 문자열처리 함수, 일반 유틸리티 함수, 수학 함수, 오류처리 함수, 그리고 시간관련 함수들로 구성된다. 입출력 함수에서는 파일 입출력과 형식화 입출력에 관련된 기능들을 제공하고, 오류 처리 함수에서는 파일에 관련된 오류 검사와 발생한 오류에 대한 메시지 출력 등의 기능들을 지원한다. 문자열처리 함수는 문자열의 복사, 결합, 비교 등의 기능들을 제공하고, 일반 유틸리티 함수는 문자와 숫자의 상호 변환, 메모리 할당 및 해제 등의 기능들을 지원한다. 수학 함수는 모듈로 연산, 로그 함수, 삼각 함수 등의 기능들을 제공하고, 시간처리 함수는 시간과 날짜의 설정 등에 관련된 기능들을 제공한다.

2.2.2 그래픽/윈도우 라이브러리

그래픽/윈도우 라이브러리는 사용자에게 디스플레이 화면을 통하여 다양한 정보를 제공하고, 사용자가 이 정보를 쉽게 처리할 수 있도록 지원해야 한다. 이를 위해 Q+ 그래픽/윈도우 라이브러리는 응용 프로그램 개발이 용이하도록 다양한 사용자 인터페이스를 제공하고, 표준 VGA 이상(예를 들어, 640×480×256칼라)의 그래픽 해상도를 제공함으로써 모니터, TV 화면 등에 모두 사용 가능하도록 설계되었다. 그리고, 윈도우 및 리눅스 환경에서 그래픽/윈도우 시스템을 위한 시뮬

레이션 환경을 구축할 수 있도록 하여, 응용 프로그램 개발 환경을 단순화하고 실제 타겟(target) 시스템이 준비되지 않아도 그래픽 응용 프로그램을 개발할 수 있도록 하였다[1, 2, 12, 13]. 그래픽/윈도우 라이브러리의 구조는 (그림 3)과 같다.



(그림 3) 그래픽/윈도우 라이브러리 구조

(그림 3)에서 보듯이, 그래픽/윈도우 라이브러리는 입출력 장치와 통신하는 그래픽 프리미티브 라이브러리와 상위 개념의 윈도우 매니저, 윈도우 라이브러리, 이벤트/태스크 라이브러리, 그리고 폰트 라이브러리로 구성된다. 그래픽 프리미티브 라이브러리는 펜 색상, 크기 등과 같이 펜의 속성을 설정할 수 있는 기능과 좌표 이동, 점 및 선 그리기 등의 기본적인 그래픽 함수들을 제공하고, 윈도우 매니저는 여러 개의 윈도우를 통하여 윈도우 생성과 종료 등을 지원하는 사용자 인터페이스를 제공한다. 윈도우 라이브러리는 상위 레벨의 응용 프로그램 인터페이스를 제공하는 그래픽 사용자 인터페이스(GUI) 요소들을 포함하고, 이벤트/태스크 라이브러리는 사용자의 입력과 다른 응용 프로그램으로부터의 이벤트 및 태스크 관리를 위한 인터페이스를 제공한다. 마지막으로, 폰트 라이브러리는 기본적으로 문자열을 출력시키고 문자를 축소, 확대하는 기능을 제공한다.

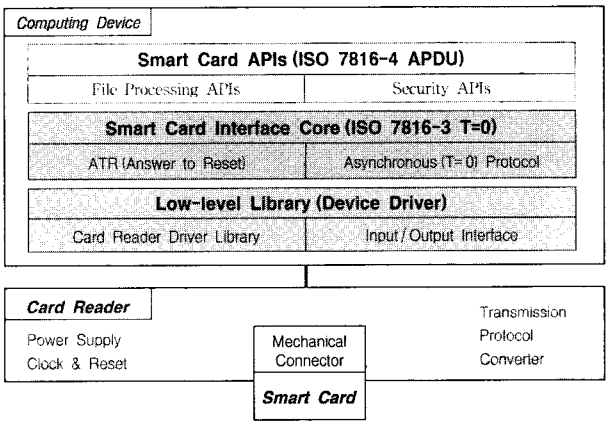
2.2.3 네트워크 라이브러리

네트워크 라이브러리는 응용 프로그램의 TCP/IP 사용, 파일 전송, 원격 로그인 및 자동 환경 설정을 지원하도록 설계되었다. 네트워크 라이브러리는 프로그램 수행을 위한 그래픽 사용자 인터페이스를 제공하며, 인터넷 표준을 준수하여 타 시스템과의 호환성을 유지하도록 설계되었다[1, 2, 14-16]. 네트워크 라이브러리의 구조는 (그림 1)의 네트워크 부분과 동일하다.

(그림 1)에서 보듯이 네트워크 라이브러리는 네트워크 공용 라이브러리, DHCP, TFTP, 그리고 텔넷으로 구성된다. 네트워크 공용 라이브러리는 네트워크 라이브러리에서 공통적으로 사용되는 기능들을 모아 놓은 것으로, 호스트 파일과 프로토콜, 서비스 파일에서 필요한 정보를 검색하거나, 문자열 주소를 실제 숫자 주소 등으로 변환하는 기능 등을 제공한다. DHCP 모듈은 DHCP 클라이언트와 서버로 구성되며, 클라이언트가 동적으로 서버로부터 새 주소를 할당받을 수 있는 기능을 제공한다. TFTP는 TFTP 서버와 클라이언트 간에 512 바이트의 고정 길이 블록을 이용하여 파일을 송수신 할 수 있는 기능을 제공하고, 텔넷은 인터넷 상의 다른 호스트로 로그 인을 할 수 있는 기능과 터미널 에뮬레이션을 제공한다.

2.2.4 보안 지원 라이브러리

보안 지원 라이브러리는 디지털 TV셋탑박스에 선택적으로 장착되는 스마트카드 리더기를 위한 인터페이스를 ISO 7816 기반의 스마트카드 리더기 측면에서 지원한다[1, 2, 17-20]. 스마트카드의 보안 지원은 카드에 있는 정보를 직접 액세스할 수 없도록 한다. 즉, 카드로부터 외부 장치로의 정보는 카드 마이크로 프로세서에 의해서만 접근되며, 보내기 전에 정보를 암호화 하고 외부 장치는 정보를 복호화 한 후 카드를 인식할 수 있다. 보안 지원 라이브러리의 구조는 (그림 4)와 같다.



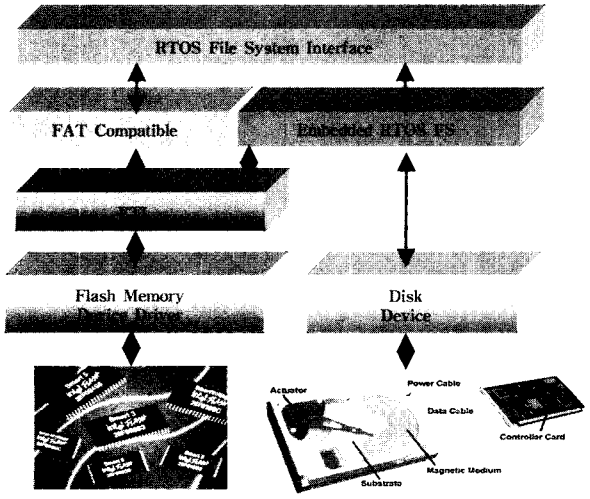
(그림 4) 보안 지원 라이브러리 구조

보안 지원 라이브러리는 프로세스 기반의 ISO 7816 표준 사양을 따르는 스마트카드 리더기와 호스트 시스템 사이의 인터페이스를 제공한다. 보안 지원 라이브러리는 카드 리더기와 호스트 시스템 인터페이스를 위해 RS232C직렬 통신을 사용하고, 이를 제어하기 위한 기능들을 포함한다. 스마트카드 리더기와 스마트카드 간의 통신은 ISO 7816-3의 비동기식 통신 프로토콜인 T=0를 포함하고, ISO 7816 표준 프로토콜과 호환성을 유지한다.

2.2.5 파일 시스템

파일 시스템은 데이터 저장을 위해 파일과 디렉터리를 관리하는 기능을 제공하고, 플래시 파일 시스템과 디스크 파일 시스템으로 구성된다. 플래시 파일 시스템은 실시간 운영체제에서 빠른 접근 시간을 요구하는 데이터를 처리하기 위해 저장 장치로 플래시 메모리를 이용하고, 디스크 파일 시스템은 멀티미디어 등의 대용량 데이터를 저장하기 위하여 하드 디스크를 이용한다[1, 2, 21-23]. 파일 시스템의 대략적인 구

조는 (그림 5)와 같다.



(그림 5) 파일 시스템 구조

(그림 5)에서 보듯이 파일 시스템은 크게 사용자 인터페이스, 파일 시스템 관리자, 플래시 매핑 레이어(FML), 그리고 디바이스 드라이버로 구성된다. 사용자 인터페이스는 기존의 유닉스 운영체제와 유사한 사용자 인터페이스를 제공하여 응용 프로그램 개발을 용이하도록 하고, FAT을 이용하여 사용자가 물리적인 저장 매체에 있는 파일을 편리하게 읽고 쓰는 것이 가능하도록 지원한다. 그리고, 플래시 매핑 레이어는 플래시 메모리에 파일을 저장하기 위하여 플래시 메모리를 섹터 형태로 보이도록 하는 기능을 수행하고, 하위 장치 드라이버는 플래시 메모리와 디스크에 대한 장치 드라이버를 제공한다.

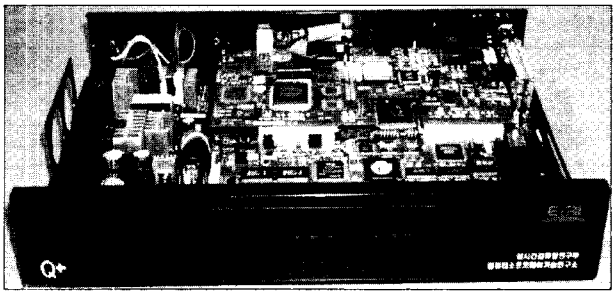
3. 라이브러리 개발 환경

Q+ 라이브러리 개발 환경은 커널과 셋탑박스의 개발 진도에 따라 크게 세 단계로 나누어진다. 먼저 1999년 1월부터 1999년 10월까지는 아직 커널과 셋탑박스가 개발되지 않았기 때문에, 상용화된 실시간 운영체제인 VxWorks와 ebsa-285 보드 상에서 라이브러리의 1차적인 구현 작업이 진행되었다. 1차 구현에 사용된 하드웨어와 개발 도구는 다음과 같다.

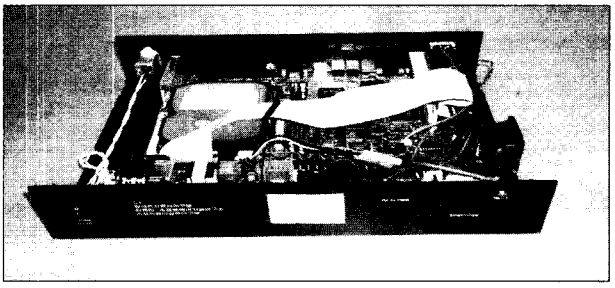
- 보드 : ebsa-285 보드(스트롱 암 SA-110)
- 운영체제 : VxWorks
- 개발 환경 : 토네이도(Tornado) 1.0
- 호스트 : PC (펜티엄 II)
- 컴파일러 : 토네이도 스트롱 암 크로스 컴파일러

개발 방식은 먼저 호스트에 설치된 토네이도를 이용하여 소스를 개발한 다음, 스트롱 암 크로스 컴파일러를 이용하여 실행 파일을 생성한다. 그런 다음, ebsa-285 보드에서 동작하는 토네이도 타겟 쉘에서 랜(LAN)을 통해 호스트로부터 실행 파일을 다운로드하여 수행하게 된다. 수행 결과는 ebsa-285 보드와 시리얼로 연결된 호스트의 터미널로 출력되게 된다.

2차 구현 작업은 1999년 11월부터 2000년 6월까지 커널 0.5와 I-TV(Intelligent Television) 셋탑박스 상에서 이루어졌고, 3차 구현은 2000년 7월부터 2000년 12월까지 커널 1.0과 디지털 TV 셋탑박스 상에서 진행되었다. (그림 6)은 2차와 3차 개발에 사용된 I-TV와 디지털 TV 셋탑박스를 보여준다.



(a) I-TV 셋탑박스



(b) 디지털 TV 셋탑박스

(그림 6) 셋탑박스

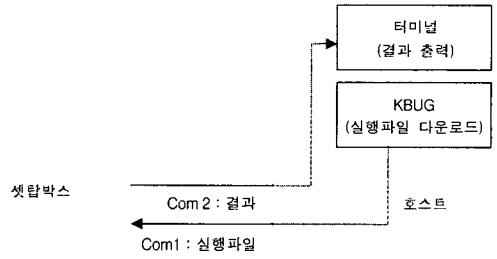
2차 구현 환경에 사용된 하드웨어와 개발도구는 다음과 같다.

- 하드웨어 : I-TV 셋탑박스(스트롱 암 : SA-110, 플래시 메모리 : 8 메가 바이트 , 메모리 : 32 메가 바이트)
- 운영체제 : 커널 0.5
- 개발환경 : KBUG(Kernel Debug)
- 호스트 : PC (펜티엄 II)
- 컴파일러 : GNU 스트롱 암 크로스 컴파일러

3차 구현 환경에 사용된 하드웨어와 개발 도구는 다음과 같다.

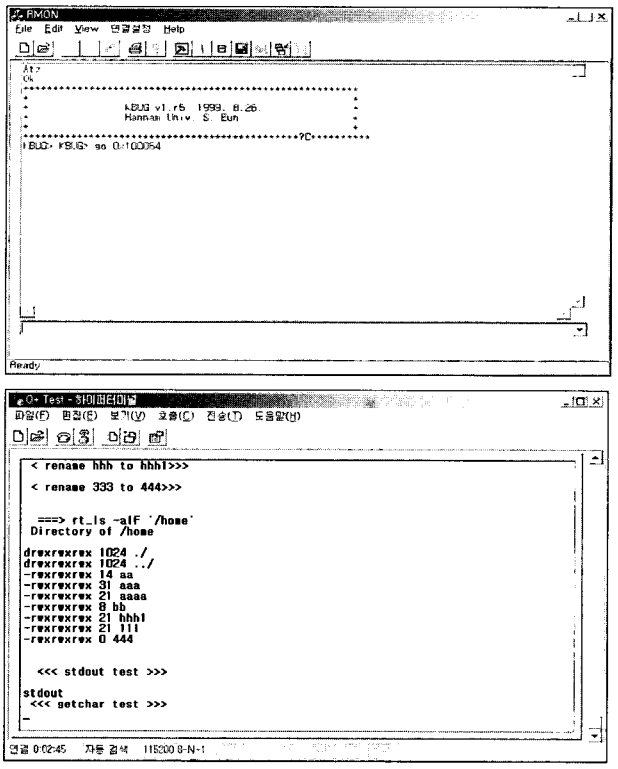
- 하드웨어 : 디지털 TV 셋탑박스(스트롱 암 : SA-110, 플래시 메모리 : 8 메가 바이트, 메모리 : 32 메가 바이트, 하드 디스크 : 8기가 바이트)
- 운영체제 : 커널 1.0
- 개발환경 : KBUG
- 호스트 : PC(펜티엄 II)
- 컴파일러 : GNU 스트롱 암 크로스 컴파일러

(그림 6)에서 보듯이 I-TV와 디지털 TV 셋탑박스는 하드웨어 사양 면에서 거의 동일하고, 대용량의 데이터를 저장하기 위해 하드 디스크가 디지털 TV 셋탑박스에 추가되었다. (그림 7)은 셋탑박스와 호스트를 연결한 구현 환경을 보여준다.



(그림 7) 셋탑 개발 환경

(그림 7)에서 보듯이 KBUG는 셋탑박스와 호스트를 시리얼을 통해 서로 연결하여 실행 파일을 호스트에서 셋탑박스로 다운로드할 수 있도록 한다. 실행 결과는 셋탑박스와 시리얼로 연결된 호스트 상의 터미널로 출력된다.



(그림 8) KBUG 수행 화면과 출력 결과

개발 방식은 먼저 스트롱 암용 크로스 컴파일러가 설치된 유닉스 기계에서 필요한 코드를 개발한 다음, .bin 형태의 실행 파일을 생성한다. 구현 작업에 사용된 유닉스 기계는 SunOS 5.5.1을 사용하는 Ultra-I 워크스테이션이다. 실행 파일을 생성한 다음, KBUG가 설치된 호스트로 실행 파일을 옮기고, 다시 KBUG를 이용하여 실행파일을 셋탑박스로 다운로드 한다. 셋탑박스로 실행 파일을 다운로드 한 다음, KBUG에서 시작 메모리 주소를 입력하게 되면 파일이 실행되고, 출력은 셋탑박스와 시리얼로 연결된 터미널을 통해 호스트로 출력된다. 이 때, 실행파일을 셋탑박스로 다운로드하는 주소(0×100000)와 실행 주소(0×100064)는 항상 일정하고, 하나의 실행파일을 수행한 다음에는 항상 셋탑박스를 리셋(reset)해야 한다. (그림 8)은 KBUG 수행 화면과 터미널을 통해 출력된 수행 결과를 보여준다.

4. 라이브러리 구현

이 절에서는 Q+ 라이브러리 구현과 수행 예제, 그리고 라이브러리의 특징 등에 대해 기술한다.

4.1 라이브러리 구현

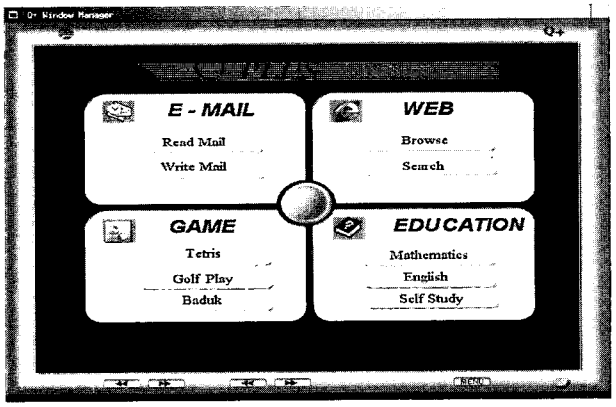
Q+ 라이브러리들은 응용 프로그램 간의 호환성을 최대한 보장하기 위해서, POSIX.1(C언어), ISO 7942 GKS(그래픽), RFC 2131(DHCP), RFC 1350(TFTP), RFC 854~861(텔넷), ISO 7816(스마트 카드), Flash File, FAT File 등의 업계 표준 및 국제 표준에 따라 구현되었다. 각 라이브러리의 구현은 기존의 공개 소스들 중에서 Q+의 개발 목표에 적합한 소스를 이용하여 이루어졌다. 예를 들어, C 라이브러리는 GNU glibc -2.0.x와 시그너스(cygnus) 사의 newlib가 주로 참조되었고, 네트워크 라이브러리는 ISC(Internet Software Consortium)에서 공개된 DHCP 소스와 BSD에서 공개된 텔넷, TFTP 소스들을 이용하였다. 그리고, 커널과 라이브러리의 개발이 일정상 동시에 진행되었기 때문에, 라이브러리에서 필요한 시스템 호출과 기타 커널 기능들이 커널에 포함되어 있는지 조사하고, 구현 중 발생한 문제점들은 해당 라이브러리 및 커널에 지속적으로 요구하는 방식으로 해결되었다.

라이브러리의 구현 작업은 2000년 5월 말까지 I-TV 셋탑박스에서 완료를 하고, 한 달 동안에 걸쳐 커널과 통합한 후, Q+ 버전 1.0을 2000년 6월 말에 릴리스 하였다. Q+ 버전 1.0은 플래시 파일 시스템만 포함되고, 디스크 파일 시스템은 포함되지 않았다. 2000년 7월부터 2000년 10월까지는 Q+ 버전 1.0을 디지털 TV셋탑박스로 이식하는 작업과 Q+ 응용 프로그램들의 요구 사항이 주로 반영되어, 응용 프로그램에서 필요한 기능들이 라이브러리에 새롭게 추가되었다. 예를 들어, 그래픽/윈도우 라이브러리는 응용 프로그램에서 필요로 하는 추가적인 위젯들(예를 들어, 작업 진행 표시 바, 화면 스크롤)이 포함되었고, C 라이브러리는 유니코드 문자열을 처리하기 위한 함수들이 추가되었다. 그리고, 디스크 파일 시스템도 디지털 TV셋탑박스에서 구현되어, 기존의 플래시 파일 시스템과 통합되었다. Q+ 라이브러리 구현은 2000년 10월말까지 완료되었고, 최종 데모는 2000년 11월에 실시되었다.

4.2 라이브러리 수행 예제

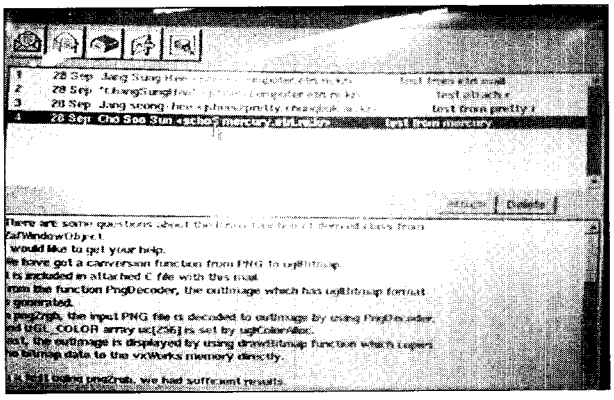
라이브러리는 특성상 다른 응용 프로그램에 포함되어 수행되므로, 이 절에서는 라이브러리 만의 수행 예제 뿐만 아니라 다른 응용 프로그램들의 수행 예제를 함께 보여준다.

(그림 8)은 C 표준 라이브러리의 테스트 프로그램 수행 결과를 나타내고, (그림 9)는 그래픽/윈도우 라이브러리의 초기 화면을 보여준다.



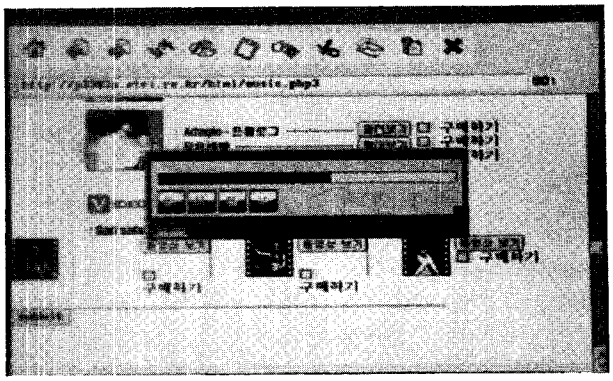
(그림 9) 그래픽/윈도우 초기 화면

(그림 9)에서 보듯이 그래픽/윈도우 초기 화면은 다수의 버턴들로 구성되고, 각 서브 윈도우에 특정 응용 프로그램을 연결하여 수행할 수 있다. 즉, 특정 응용 프로그램을 윈도우 초기화면의 적절한 버턴에 연결시킨 후, 마우스로 그 버턴을 선택하게 되면 응용 프로그램을 수행시킬 수 있게 된다. (그림 10)은 사용자가 E-MAIL 서브 윈도우의 Read Mail을 선택했을 때 수행되는 메일 클라이언트의 수행 화면을 보여준다.



(그림 10) 메일 클라이언트 수행 화면

라이브러리를 이용한 응용 프로그램 중에는 미디어 플레이어와 웹 브라우저 등이 있다. Q+에서 동작하는 미디어 플레이어에는 MPEG-1, MP3, MPEG-4들이 있는데, (그림 11)은 MP3 미디어 플레이어의 수행 화면을 보여준다.



(그림 11) MP3 미디어 플레이어 수행 화면

4.3 라이브러리의 특징

일반적으로 실시간 운영체제 Q+ 가 탑재되는 기기에서는 사용할 수 있는 하드웨어 자원과 응용 분야가 제한되어 있기 때문에, 불필요한 라이브러리 함수들의 탑재는 가용 자원의 크기를 줄여 응용 프로그램들의 반응 시간을 증가시킬 수 있다. 따라서, Q+ 라이브러리들은 불필요한 함수들을 제거하여 라이브러리의 크기를 줄이는 작업을 진행하였다. 예를 들어, C 표준 라이브러리는 함수 수를 줄이기 위해POSIX.1에 정의된 표준 함수들 중에서 커널과 디지털 TV 응용 분야에 적합하지 않는 함수들은 1차적으로 제거하였다. 그리고, 실시간 운영체제들인 VRTX[7], VxWorks[8], pSOS[9], Chorus[10]에서 제공하는 C 라이브러리 함수들을 비교하여, 이들 운영체제에서 공통적으로 제공하는 함수들을 우선적으로 선정하였다. 결과적으로, POSIX.1에 정의된 총 325개의 표준 함수들 중에서 VxWorks는 172개, pSOS는 147개, Chorus는 172개, VRTX는 175개를 구현하였지만, C 표준 라이브러리는 145개의 함수들만으로 Q+ 응용에 필요한 기능들을 제공할 수 있었다. 그리고, Q+ 라이브러리들은 응용 프로그램 간의 호환성을 최대한 보장하기 위해서, POSIX.1(C언어), ISO 7942 GKS(그래픽), RFC 2131(DHCP), RFC 1350(TFTP), RFC 854~861(텔넷), ISO 7816(스마트 카드), Flash File, FAT File 등의 업계 표준 및 국제 표준에 따라 구현되었다. 응용 프로그램에서 사용되는 라이브러리 함수들이 표준 인터페이스에 따라 정의되면, 응용 프로그램들이 라이브러리 함수 호출로 인한 추가적인 코드 변경 없이 다른 운영체제 상에서 쉽게 수행될 수 있는 장점이 있다.

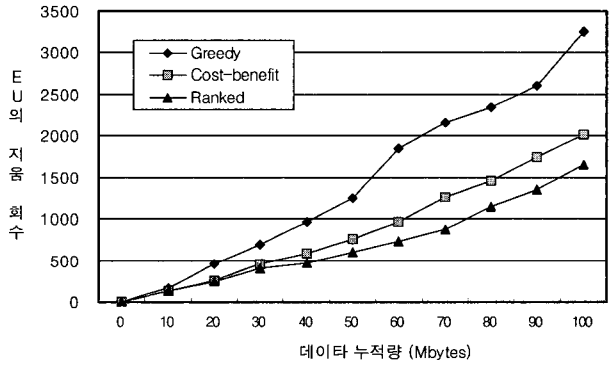
Q+ 라이브러리는 응용 분야의 특성에 따라 소량의 데이터를 빠르게 접근할 수 있도록 지원하는 플래시 파일 시스템과 대용량의 데이터를 저장할 수 있도록 하는 디스크 파일 시스템을 선택적으로 사용할 수 있도록 지원한다. 즉, 파일 시스템은 플래시 파일 시스템과 디스크 파일 시스템 중에서 하나의 파일 시스템만 사용하거나, 모두 사용할 수 있어 응용 분야의 특성에 따라 사용자가 선택적으로 파일 시스템을 구성할 수 있다. Q+ 플래시 파일 시스템은 기존의 플래시 파일

시스템들이 갑작스러운 전원 공급 중단 시 데이터를 복구하지 못하는 문제점을 해결하여, 전원 공급 중단 시에도 데이터를 복구하는 기능을 제공한다. 그리고, 제한된 용량의 플래시 메모리에서 쓰기 공간이 부족할 때 무효한 블록들을 지우는 새로운 등급별 지움 정책(Ranked Cleaning Policy)을 구현하였고, 주 메모리에 여러 개의 섹터를 저장한 뒤 주기적으로 플래시 메모리로 읽기/쓰기 연산을 수행함으로써 플래시 메모리의 참조 횟수를 감소시켰다. <표 1>는 Q+ 플래시 파일 시스템의 지움 정책과 기존의 Greedy 지움 정책 및 Cost-benefit 지움 정책과의 성능 비교를 위한 실험 환경을 보여준다. <표 1>에서 보듯이 성능 평가는 디지털 TV 셋탑박스 상에서 이루어졌고, 플래시 메모리의 지움 단위(Erasing Unit)는128 Kbytes 이다.

〈표 1〉 실험 환경

항 목	값
플래시 메모리	Intel 28F320S[8]×4개
EU 크기	128 *Kbytes*
EU 개수	32 *개/Chip*
쓰기 속도	5.76 μ *sec*
지우기 속도	0.5 *sec/EU*
CPU	SA 110S 233M RISC
메인 메모리	32 *Mbytes*
하한의 Free EU 개수	$N_1 = 5$, $N_2 = 10$
등급별 지움 정책 상수값	$A = 1/1000$

실험 방법은 100 Mbytes까지 누적하여 플래시 파일시스템에 데이터를 저장하면서 발생되는 지움 단위의 지움 회수를 조사하였다. 전체 플래시 파일 시스템의 데이터 양은 플래시 메모리 크기의 60% 정도에서 수행하였고, 등급별 지움 정책에 사용되는 상수 값 A는 1/1000로 정하였다. (그림 12)는 실험 결과를 보여준다. (그림 12)에서 보듯이 Q+ 플래시 파일 시스템에 적용된 지움 정책이 기존의 Greedy 지움 정책에 비해 약 50%, Cost-benefit 지움 정책에 비해 약 20%의 성능 개선을 보여주고 있다[4].



(그림 12) 지움 정책에 대한 성능 평가

5. 결론 및 향후 연구

본 논문에서는 정보가전용 실시간 운영체제 Q+에 탑재되는 라이브러리의 설계 및 구현에 대해 기술하였다. Q+는 1999년 1월부터 2000년 12월까지 1차적으로 디지털 TV용 인터넷 셋탑박스에 탑재되는 것을 목표로 개발되었으며, 크게 커널, 라이브러리, 개발 도구, 응용 API로 구성된다. 라이브러리는 다시 C 표준 라이브러리, 그래픽/윈도우 라이브러리, 네트워크 라이브러리, 보안 지원 라이브러리, 파일 시스템으로 구성된다. Q+ 라이브러리들의 구현은 POSIX.1, ISO 7942 GKS 등의 업계 및 국제 표준에 따라 구현되었으며, 주로 공개된 오픈 소스를 참고로 하여 커널과 디지털 TV용 셋탑박스, 그리고 개발 도구인 KBUG 등을 이용하여 구현되었다.

Q+ 라이브러리는 Q+ 응용 분야에 적합하지 않은 함수들을 제거하여 라이브러리 크기를 줄였고, 응용 분야의 특성에 따라 소량의 데이터를 빠르게 접근할 수 있도록 하는 플래시 파일 시스템과 대용량의 데이터를 저장할 수 있도록 하는 디스크 파일 시스템을 선택적으로 사용할 수 있도록 하였다. 그리고, Q+ 플래시 파일 시스템은 기존의 플래시 파일 시스템들이 갑작스러운 전원 공급 중단 시 데이터를 복구하지 못하는 문제점을 해결하여, 전원 공급 중단 시에도 데이터를 복구하는 기능을 제공한다.

2001년 1월부터 2001년 10월까지 Q+ 기능 테스트와 기능 보완 작업이 진행되고 있으며, 2001년 11월에 Q+ 소스를 무료로 공개하여 개발자가 응용 분야에 적합한 기능들을 자유롭게 추가할 수 있도록 지원할 예정이다.

참 고 문 헌

[1] 한국전자통신연구원, "정보가전용 실시간 OS 컨퍼런스", RTOS '99 자료집, 1999.

[2] 한국전자통신연구원, "정보가전용 실시간 OS 컨퍼런스", RTOS 2000 자료집, 2000.

[3] 김도형, "실시간 운영체제 Q+를 위한 C 표준 라이브러리의 설계 및 구현", 정보처리학회논문지 A, Vol.8-A, No.1, pp.1-8, March, 2001.

[4] 김정기, "정보가전을 위한 플래시 파일시스템에서 등급별 지움 정책과 오류 복구 방법", NCS 2001 차세대 통신소프트웨어 학술대회, Dec. 2001.

[5] ISO/IEC 9945-1, "C 언어를 위한 시스템 응용 프로그래밍 인터페이스(API) 표준", 1993.

[6] 체신부, "개방형 운영체제 인터페이스(POSIX.1) 표준", 1993.

[7] "VTRX Reference Guide," Mentor Graphics Corporation, 1997.

[8] "VxWorks 5.3.1 Programmer's Guide Edition *1*," Wind River Systems, 1997.

[9] "pSOSystem Programmer's Reference," Integrated Systems,

1997.
[10] "Standard C Library Function," Chorus OS man page, 1994.
[11] ISO/IEC 9636-1, "정보 기술-그래픽 장치와의 대화를 위한 인터페이스 기법(CGI)-기능 규격", 1991.
[12] "Nucleus GRAFIX System and Library Reference Manual," Accelerated Technology, 1998.
[13] Richard F. Ferraro, "Programmer's Guide to the EGA, VGA, and Super VGA Cards," Addison Wesley, 1994.
[14] R. Droms and K. Fong, "Netware/IP Domain Name and Information," RFC 2242, November 1997.
[15] D. Provan, "DHCP Options for Novell Directory Services," RFC 2241, Novell Inc., November, 1997.
[16] G. Malkin and A. Harkin, "TFTP Timeout Interval and Transfer Size Options," RFC 2349, May, 1998.
[17] ISO 7816-1, "Physical Characteristics," 1987.
[18] ISO 7816-2, "Dimensions and Location of Contacts," 1988.
[19] ISO/IEC 7816-3, "Electronic Signals and Transmission Protocols," 1989.
[20] ISO/IEC 7816-4, "Interindustry Commands for Interchange," 1995.
[21] Atsuo Kawaguchi, Shingo Nishioka, and Hiroshi Motoda, "A Flash-Memory Based File System," In Proceedings of the Winter USENIX Technical Conference, 1995.
[22] "Flash File System Selection Guide," Intel Corporation, Technical Paper, December, 1997.
[23] "Software Concerns of Implementing a Resident Flash Disk," Intel Corporation, Technical Paper, December, 1995.

김 도 형

e-mail : dhkim@chinmi.etri.re.kr
1993년 경북대학교 컴퓨터공학과(공학사)
1995년 포항공과대학 전자계산학과 (이학석사)
1995년~현재 한국전자통신연구원 컴퓨터 소프트웨어기술연구소 선임연구원
관심분야 : 결함포용, 실시간시스템, 성능감시, 성능평가

박 승 민

e-mail : minpark@etri.re.kr
1981년 울산대학교 전자공학과(학사)
1983년 홍익대학교 전자공학과(석사)
1983년~1984년 LG 전자 연구원
1984년~현재 한국전자통신연구원 컴퓨터 소프트웨어기술연구소 책임연구원
관심분야 : 실시간 운영체제, 네트워크 컴퓨팅