# 임베디드 시스템을 위한 ISO-9660 파일 시스템 구현*

이호송⁰, 성영락†, 이철훈, 권택근
충남대학교 컴퓨터공학과, 국민대학교 전자정보통신공학부
(hslee⁰, chlee, tgkwon)@ce.cnu.ac.kr, † yeong@mail.kookmin.ac.kr

# Implementation of ISO-9660 File System for Embedded System

Ho-Song Lee⁰, Yeong Rak Seong†, Cheol-Hoon Lee, and Taeck-Geun Kwon
Dept. of Computer Engineering, Chungnam national Univ.
† School of Electrical Engineering, Kookmin Univ.

요 약

최근에 PDA 나 디지털 TV, 셋탑박스, 디지털 카메라, DVD 플레이어, IA(Internet Appliance)제품 등 일반 소비자 제품에도 임베디드 시스템이 탑재되면서 임베디드 시스템 시장이 급속도로 발전하고 있다. 이런 임베디드 시스템의 핵심은 실시간 운영체제(RTOS)와 시스템의 특성에 맞는 파일시스템이다. 본 논문에서는 임베디스 시스템에 탑재된 실시간 운영체제에서 동작할 수 있는 파일시스템 중에서 가상 파일 시스템과 통합하여 동작하는 ISO-9660 파일 시스템을 구현하였다. 본 논문에서 구현한 ISO-9660 파일 시스템은 ISO-9660 과 RRIP(Rock Ridge Interchange Protocol), Joliet 표준을 지원한다.

## 1. 서 론

초기에 군사용 장비나 산업용 반도체 제조 장비 분야에서 시작되었던 임베디드 시스템은, 최근 CPU 의 발전과 함께 비약적 성장을 하면서 스마트폰, 디지털 TV, 디지털 카메라, PDA, DVD 플레이어, IA(Internet Appliance) 제품 등의 일반 소비자 제품에도 독자적인 CPU 와 운영체제가 탑재되면서 현재 그 적용범위를 크게 넓혀가고 있다.

특정한 기능의 수행을 목적으로 하는 임베디드 시스템은 적용될 시스템의 특성에 맞는 운영체제와 파일 시스템으로 구성된다. 본 논문에서는 DVD 타이틀 뿐만 아니라 CD-ROM 에 저장된 데이터도 처리할 수 있는 DVD 플레이어를 위한 ISO-9660 파일 시스템을 구현하였다.

본 논문의 구성은 다음과 같다. 2 장에서는 파일 시스템의 개념과 구조를 소개한다. 3 장에서는 ISO-9660 파일 시스템의 데이터 저장 기법과 자료 구조에 대해 설명한다. 4 장에서는 ISO-9660 파일 시스템의 구현 내용을 기술하고 결론 및 향후 연구 과제는 5 장에서 제시한다.

## 2. 파일시스템

파일 시스템은 파일을 하드 디스크나 기타 저장 장치에 저장하는 방식과 그에 연관된 연산(operation) 및 자료 구조(structure)를 포괄적으로 포함한다. 일반적으로 시스템은 많은 파일 시스템을 지원하기 위해서 가상 파일 시스템(Virtual File System)을 사용한다. 이 장에서는 가상 파일 시스템의 개념과 구조에 대해서 살펴보자.

### 2.1 가상 파일 시스템 개념

일반적으로 파일 시스템은 여러 종류의 저장 장치를 지원한다. 파일 시스템 내에는 저장 장치의 특성에 따라 각기 다른 파일 시스템이 존재한다. 가상 파일 시스템은 각각의 파일 시스템을 통합하여 사용자 계층으로 동일한 인터페이스를 제공하고, 사용자가 저장 장치에 접근하고자 하는 경우, 각각의 파일 시스템의 연산을 호출한다. 즉, 가상 파일 시스템은 일관된 방법으로 각각의 파일 시스템을 접근할 수 있게 한다. 아래의 그림 1 은 사용자가 가상 파일 시스템을 통하여 서로 다른 저장 장치에 접근하는 구조를 보여준다.

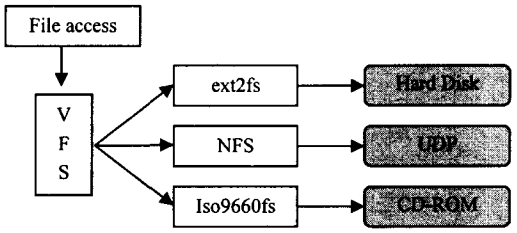


그림 1. 파일 시스템의 구조

### 2.2 가상 파일 시스템의 구조

파일 시스템은 저장 장치를 접근하기 위한 연산과 자료 구조를 정의하고 있다. 파일 시스템은 슈퍼블록과 디렉토리 엔트리, 아이노드, 파일 등의 자료 구조를 사용한다. 각각의 파일 시스템은 자신의 자료 구조를 제어하기 위한 각기 다른 연산을 정의한다. 즉, 일관성을 유지하기 위해서 동일한 자료 구조를 사용하지만, 그 연산은 각각의 파일 시스템의 특성에 맞게 재정의 된다. 가상 파일 시스템이 각각의 파일 시스템의 특성에 맞는 연산을 호출함으로써 사용자는 서로 다른 파일 시스템의 특성을 고려할 필요가 없다. 다음의 그림 2 는 파일 시스템의 내부 자료 구조의

* 본 논문은 BK21 대전, 충남 정보통신인력양성 사업단의 RA 연구비 지원에 의한 것임

연관 관계를 보여준다.

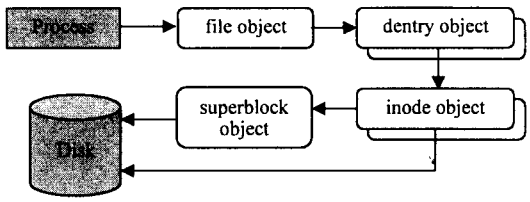


그림 2. 파일 시스템 객체의 상호 동작

## 3. ISO-9660 파일 시스템의 구조

ISO-9660 파일 시스템은 CD-ROM 에 저장된 데이터를 처리하기 위해서 구현된 것이다. 이 장에서는 ISO-9660 의 데이터 저장 구조를 살펴보고 ISO-9660 자료 구조를 사용하여 파일 시스템을 구성하는 방법을 기술한다.

### 3.1 ISO-9660 의 데이터 저장 구조

ISO-9660 의 물리적인 데이터 저장 구조를 살펴보자. ISO-9660 에서는 0~15 번째 섹터까지는 시스템을 위해서 예약된 영역이다. 실제로 데이터가 저장 되는 영역은 아래의 그림 3 에서 보는 것처럼 물리적으로 16 번째 섹터에서부터 이다. ISO-9660 의 섹터 크기는 2,048 bytes 이다.

16

| RFU | Volume Descriptor | Path Table | Directory Record | Files |
|---|---|---|---|---|

그림 3. ISO-9660 의 물리적인 데이터 저장 구조

### 3.2 ISO-9660 의 자료 구조

ISO-9660 의 자료 구조는 볼륨 디스크립터, 패스 테이블, 디렉토리 구조의 세가지로 분류된다.

볼륨 디스크립터는 볼륨에 대한 정보와 다른 자료 구조의 위치를 저장하고, 루트 디렉토리 레코드를 가지고 있다. 패스 테이블은 각 디렉토리의 위치와 부모 디렉토리의 위치를 저장하고 있다. 디렉토리 구조는 파일이나 디렉토리의 이름과 위치를 저장하고 있다. 디렉토리의 구조는 트리 구조로 구성 되어 있다. 그림 4 는 ISO-9660 의 자료 구조를 보여준다.

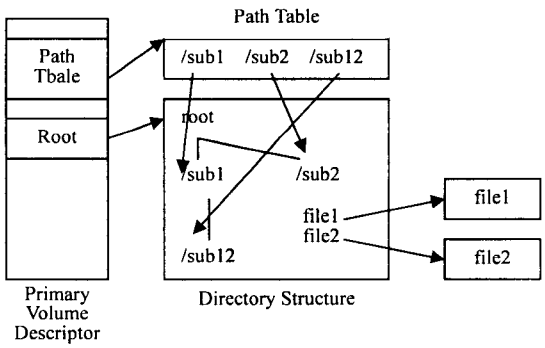


그림 4. ISO-9660 자료 구조

ISO-9660 에서는 CD-ROM 에 대한 몇 가지 제약사항을 정의하고 있다.

문자는 ISO-646 캐릭터 셋을 사용하며 파일명은 최대 8 자의 캐릭터 이름과 3 자의 확장자를 사용할 수 있도록 규정하였다. 디렉토리의 레벨도 8 레벨 이하로 내려가지 않도록 규정하고 있다. 이러한 제약을 해결하기 위해서 RRIP, Joliet 과 같은 표준이 제안되었다.

### 3.3 ISO-9660 의 디렉토리 레코드 구조

일반적으로 대부분의 운영체제는 디렉토리나 파일에 빠르게 접근하기 위하여 패스 테이블을 메모리에 적재하여 사용한다. 패스 테이블을 통하여 디렉토리 레코드에 접근할 경우, 한번의 저장 장치 접근을 통해서 원하는 레코드에 대한 정보를 얻어올 수 있다. ISO-9660 의 디렉토리 레코드 저장 구조는 물리적인 블록(2048 bytes) 단위로 디렉토리 내의 디렉토리나 파일에 대한 레코드를 저장하고 있다. 그림 5 는 디렉토리 레코드가 물리적인 블록에 저장되는 구조를 보여준다.

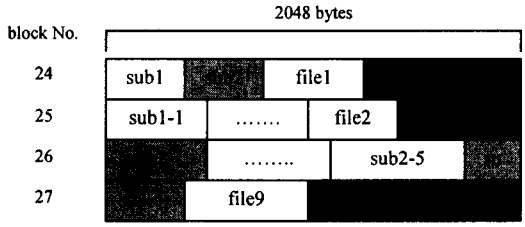


그림 5. 디렉토리 레코드의 물리적인 저장 구조

그림 5 에서 24, 25, 26~27 번째 블록은 각각 하나의 디렉토리 내에 디렉토리나 파일에 대한 레코드를 저장하고 있다. ISO-9660 은 패스 테이블을 거치지 않더라도 해당 블록을 검색함으로써 디렉토리나 파일에 대한 정보를 알 수 있다. 구현된 ISO-9660 파일 시스템은 패스 테이블을 사용하지 않고 디렉토리 레코드에 접근 할 수 있도록 설계되었다. 그러므로 패스 테이블을 메모리에 적재하여 사용하는 시스템 보다 속도는 느리지만, 한정된 메모리 공간을 효율적으로 사용할 수 있다.

임베디드 시스템은 상대적으로 적은 메모리를 사용하기 때문에 모든 디렉토리 엔트리 캐쉬를 메모리에 구성하면 메모리를 효과적으로 사용할 수 없다. 따라서 구현된 파일 시스템은 루트 디렉토리로부터 필요한 디렉토리 엔트리를 찾기 위한 최소한의 디렉토리 엔트리 캐쉬를 유지하며, 유지하는 디렉토리 엔트리와 연관된 아이노드만을 메모리에 저장한다.

### 3.4 ISO-9660 의 확장

일반적으로 UNIX 시스템에서는 긴 파일명을 사용하고, 디렉토리의 레벨도 8 레벨이 넘는 경우가 빈번하였다. ISO-9660 의 제약사항을 지키기 힘든 UNIX 시스템의 필요에 의해서 RRIP 가 제안되게 되었다.

RRIP 는 ISO-9660 의 디렉토리 구조를 변경시키지 않고, 디렉토리 레코드 내의 System Use 필드를 사용하여 ISO-9660 의 제약사항을 해결할 방법을 제시하였다. 또한 POSIX 파일과 디렉토리를 지원할 수 있는 방법을 제시하였다.

Joliet 표준은 MS(Microsoft)사가 Windows 95 용으로 개발한 표준이다. 현재 MS Windows 계열에서 사용되는 모든 CD 는 Joliet 을 사용하고 있다. Joliet 표준 역시 파일명의 길이와 디렉토리의 레벨의 제약을 개선하는 방법과 유니코드를 지원하는 방법을 제시하고 있다. Joliet 은 ISO-9660 에서는 사용하지 않았던 SVD(Supplementary Volume Descriptor)를 사용하여 유니코드를 지원한다.

## 4. ISO-9660 파일 시스템 구현

본 논문에서 구현한 ISO-9660 파일 시스템은 C 언어로 개발되었으며, 컴파일러는 gcc 버전 2.96 을 사용하였다. 개발 및 테스트 환경은 인텔 CPU 를 사용하는 Redhat Linux Kernel 버전 2.4.18 이다. 차후에 ARM920T 를 기반으로 한 삼성 S3C2800™ 32-bit RISC Micro Processor 에서 테스트할 예정이다.

### 4.1 ISO-9660 구현

ISO-9660 파일 시스템은 가상 파일 시스템과 통합되어 동작해야만 한다. ISO-9660 파일 시스템의 함수들은 가상 파일 시스템의 연산을 위한 함수와 동일한 매개변수와 기능을 수행하도록 정의하고 있다.

CD-ROM 의 특징은 다른 저장 장치와 달리 읽기 전용이라는 것이다. 구현 단계에서는 쓰기에 관련된 연산을 정의할 필요가 없다. ISO-9660 파일 시스템은 CD-ROM 의 정보를 읽어서 슈퍼블럭을 생성하고, 그에 따른 디렉토리 엔트리, 아이노드, 파일의 연산을 정의한다. ISO-9660 파일 시스템은 가상 파일 시스템과 연관되어 동작한다. 따라서 가상 파일 시스템을 위한 표준 인터페이스가 필요하다. 아래의 표 2 는 ISO-9660 파일 시스템이 가상 파일 시스템에 제공하는 기본 인터페이스이다.

표 1. ISO-9660 파일 시스템 인터페이스

| | |
|---|---|
| iso9660_read_super | CD-ROM 의 볼륨 디스크립터를 접근하여 슈퍼블럭 구조체와 루트 디렉토리 엔트리와 루트 아이노드를 생성한다. |
| iso9660_read_inode | 아이노드 번호가 채워진 구조체를 매개변수로 받아서 아이노드 구조체의 다른 데이터를 채운다. |
| iso9660_lookup | 현재 디렉토리내에서 해당 파일명의 디렉토리 엔트리를 찾는다. |
| iso9660_readdir | 디렉토리의 내용을 읽는다. |
| iso9660_readpage | 아이노드 구조체를 사용하여 실제 파일을 내용을 버퍼에 저장한다. |
| iso9660_block_read | 실제로 CD-ROM 에서 해당 블록을 버퍼에 저장한다. |
| iso9660_init | 가상 파일 시스템에 ISO-9660 파일 시스템을 등록한다. |

### 4.2 확장 ISO-9660 구현

ISO-9660 의 확장인 RRIP 와 Joliet 을 지원하는 함수를 구현한다. RRIP 와 Joliet 은 ISO-9660 과 동일한 자료 구조를 사용한다. 단지 긴 파일명을 제공하기 위해서 디렉토리 레코드의 System Use 필드를 사용하고 파일명에 유니코드를 사용할 수 있다. 각각의 표준에 맞는 파일명 변환 함수를 제공함으로써 확장 ISO-9660 을 구현한다. 표 3 은 각각의 표준에 맞는 파일명 변환 함수이다.

Joliet 은 유니코드를 지원하기 위해서 SVD(Supplementary Volume Descriptor)를 사용한다. 파일 시스템이 Joliet 을 지원하기 위해서는 iso9660_read_super()에서 PVD(Primary Volume Descriptor)를 대신하여 SVD 를 사용하여 슈퍼블럭, 루트 아이노드와 루트 디렉토리 엔트리를 구성한다.

표 2. 파일명 변환 함수

| 표준 | 파일명 변환 함수 |
|---|---|
| ISO-9660 | iso9660_filename_translate() |
| RRIP | rock_ridge_filename_translate() |
| Joliet | joliet_filename_translate() |

### 4.3 가상 파일 시스템과 통합

임베디드 시스템은 경우에 따라서 여러 종류의 파일 시스템이 필요할 수 있다. 서로 다른 파일 시스템이 동시에 사용되기 위해서 가상 파일 시스템과의 통합은 필수적이다. 구현된 ISO-9660 파일 시스템은 4.1 에서 살펴본 인터페이스를 통해서 가상 파일 시스템과 통합하여 동작한다. 가상 파일 시스템에 ISO-9660 파일 시스템을 등록한 후 사용자의 마운트(mount) 요청이 있을 때 iso9660_read_super() 를 호출하여 ISO-9660 파일 시스템을 구성한다. 파일 시스템 구성과 동시에 가상 파일 시스템의 연산을 ISO-9660 파일 시스템의 연산으로 대체함으로써 사용자는 가상 파일 시스템의 인터페이스를 통하여 ISO-9660 파일 시스템을 사용하게 된다.

## 5. 결론 및 향후 연구 과제

본 논문에서는 임베디드 시스템에 적용 가능한 ISO-9660 파일 시스템을 구현하였다. 이 파일 시스템은 ISO-9660 과 RRIP(Rock Ridge Interchange Protocol), Joliet 표준을 지원한다. ISO-9660 파일 시스템은 가상 파일 시스템, UDF 파일 시스템과 통합되어 iRTOS 에 적용되었다. 구현된 ISO-9660 파일 시스템은 다중 볼륨(multi-volume)을 지원하지 않는다. 향후에 다중 볼륨을 지원할 수 있도록 개선이 필요하다.

### 참고 문헌


[1] Nancy Klocko, *Introduction to ISO 9660*, Disc Manufacturing Inc., 1995.

[2] IEEE P1282, "ROCK RIDGE INTERCHANGE PROTOCOL", 1994.

[3] Microsoft Developer Relations Group, "Joliet Specification", 1998.

[4] ECMA-119, "Volume and File Structure of CDROM for Information Interchange", 1987.

[5] ECMA-168, "Volume and File Structure of Read-Only and Write-Once Compact Disk Media for Information Interchange", 1994.

[6] Daniel P. Bovet and Marco Cesati, *Understanding the LINUX KERNEL*, O'REILLY, 2002.

[7] Craig Hollabaugh, *Embedded Linux*, O'REILLY, 2002.

[8] 강석민, 송재영, 조정철, 권택근 "임베디드 시스템을 위한 파일 시스템 구현" *한국정보과학회 논문집*, 제 29 권 1 호, pp.61~63, 2002.