

mSROS : ATM 교환기 장치 제어계를 위한 실시간 운영체제

° 김형환, 정부금, 조주현, 차영준, 임동선

한국전자통신연구원 교환전송기술연구소 교환시스템연구부 실시간 OS 팀

Tel: 042-860-6381, Fax: 042-860-5410, hhkim@etri.re.kr

mSROS : Real-Time Operating System for Device Controller System in ATM Switching Systems

° Hyung-Hwan Kim, Bu-Geum Jung, Ju-Hyun Cho, Sung-Ik Jun, Young-Jun Cha, Dong-Sun Lim
Real-Time OS team, Switching System Department, Switching and Transmission Tech. Labs, ETRI
161 Kajong-Dong, Yusong-Gu, Taejon, 305-350, KOREA
Tel: 042-860-6381, Fax: 042-860-5410, hhkim@etri.re.kr

Abstract

In this paper, we present mSROS(Micro-Scalable Real-time Operating System) to be applied commonly to the device controller systems in the HANbit ACE256 system. The device controller systems in HANbit ACE256 system are organized as many kinds of device controllers. Applying modified PPOS(Peripheral Processor Operating System) which is an operating system for devices of the TDX-10 switching system to the firmwares for them, the inefficiency in development and maintenance exists inherently. To remove the inefficiency and to improve the performance of firmwares, we build a common operating system platform that including multi-tasking micro-kernel so that the firmwares among devices can acquire convenient development and cheap cost of maintenance. Especially, building a virtual machine as a development methodology, it is possible to remove dependency from the kernel so that any kinds of commercial real-time kernels can be used in mSROS as a basic kernel. The virtual machine in mSROS is compatible with the API of SROS(Scalable Real-time Operating System), PPOS, and CROS(Concurrent Real-time Operating System).

1. 서론

ATM 교환기인 HANbit ACE64 의 장치 제어계는 10여종의 장치 제어기들로 구성되는데 각각의 장치제어기별로 다른 CPU를 적용하여 순차 프로그래밍 방식의 PPOS(Peripheral Processor Operating System, TDX-10 교환기의 주변 장치용 운영체제)를 적용된 CPU들에 맞도록 일부 변형하고 이를 기반으로 탑재될 휠웨어들을 개발했다. 이로 인해 두 가지 관점에서의 비효율성을 내재하게 되었는데 그 각각은 시스템의 관점과 응용프로그램 개발자의 관점이다. 전자의 경우는 변형된 PPOS가 장치 제어기의 종류만큼 존재하게 되어 유지보수의 관점에서 중복된 투자가 이루어졌고 후자의 경우에는 빈번한 휠웨어의 변경 시에 print 문에만 의존하는 디버깅과 ROM(Read Only Memory) 교체에 의한 시험방법으로 시험 효율이 매우 낮아 장치 제어계 개발기간의 대부분이 휠웨어의 시험에 소요됐다는 것

이다.

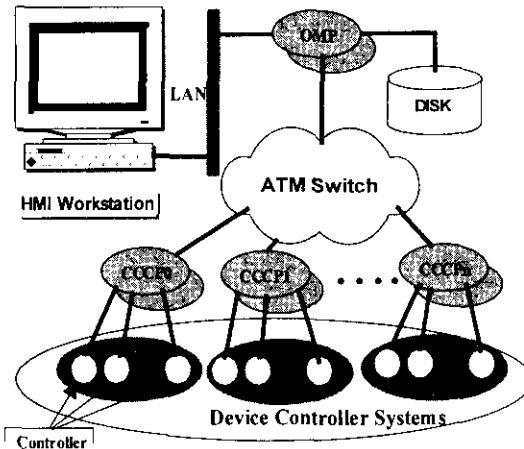
HANbit ACE64 의 다음 기종으로 개발중인 HANbit ACE256 시스템에서는 이러한 비효율성을 제거하여 장치 제어계의 개발을 효율적으로 이끌기 위해 가상기계 개념을 적용한 새로운 개발방법을 도출하고 이에 따라 장치 제어계에 공통으로 적용될 운영체제(mSROS)를 개발하게 되었다. 2장에서 mSROS 가 적용될 ATM 교환시스템의 개념적인 모델을 설명하고 3장에서 도출된 개발방법에 따른 mSROS 의 구조를 설명하며 4장에서는 mSROS 의 중요 부분인 가상기계에 대하여 그 구성요소와 정의된 인터페이스인 API(Application Program Interface)에 대해서 서술한다. 마지막으로 5장에서 결론을 맺는다.

2. ATM 교환시스템 모델

ATM 교환시스템[1]은 분산 실시간 시스템으로서 고유의 기능을 수행하는 노드(여기서는 서브 시스템이라 함)들이 ATM 교환 스위치에 연결되어 IPC(Inter Process Communication)를 통한 동기 및 통신을 통한 상호작용으로 교환기의 목적을 달성하도록 구성되어 있다.

각 서브시스템에는 SROS(Scalable Real-Time Operating System)[2]가 적용되어 실행되고 장치 제어계(Device Controller Systems)를 구성하는 각 제어기(controller)들에는 mSROS 가 탑재되어 실행되는데 이에 대한 ATM 교환 시스템의 개념적 구조는 <그림 1>에 명시되어 있다. 각 서브 시스템들로는 교환기 시스템 전체의 운용과 유지 보수를 위한 기능을 수행하는 OMP(Operation and Maintenance Processor)와 가입자들을 수용하고 호처리 기능을 수행하는 CCP(Call and Connection Control Processor)들이 있으며 각 서브 시스템들은 서비스의 연속성을 위해서 이중화[3]되어 있다. OMP는 운용자 터미널(HMI Workstation)에 LAN으로 연결되어 고속의 통신을 통해 교환 시스템내의 상태 정보를 운용자 터미널로 출력하거나 운용자 터미널로부터 지시되는 운용자의 명령어를 처리하며 과금 정보 등을 보조기억장치에 저장한다. 장치 제어계는 호

처리 기능을 수행하는 CCCP에 가입자들을 수용하는 각 장치 제어기들이 연결되어 구성되는데 각 장치 제어기는 자신이 속한 그룹내의 상위 프로세서인 CCCP들의 제어를 받으며 따라서, 제어를 위한 통신은 자신이 속한 그룹내의 CCCP와 이루어지며 고신뢰 통신을 위하여 자체 프로토콜[4]을 갖고 있다.



<그림 1> ATM 교환 시스템 구조도

ATM 교환 시스템의 제어기는 상위 제어기와 하위 제어기로 구성되며 ATM 교환 스위치에 연결[5]된 각 서브 시스템(OMP, CCCPs)들이 상위 제어기를 구성하고 장치 제어기들이 하위 제어기를 구성한다.

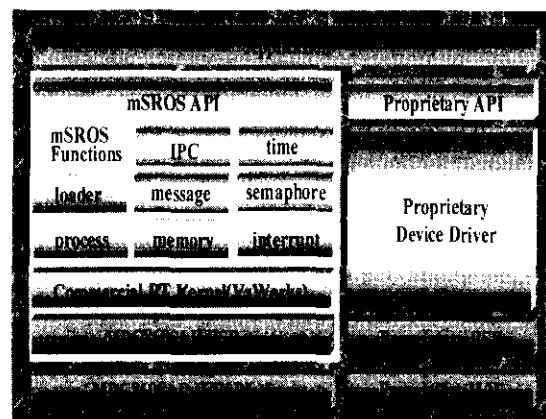
3. mSROS 의 구조

HANbit ACE64 시스템의 장치 제어기를 개발한 경험은 순차 프로그램 특성을 갖는 PPOS를 적용하여 개발된 휠웨어의 성능 개선, 시스템 관점에서의 소프트웨어 유지 보수 비용의 절감, 그리고 응용 프로그램 개발의 용이함에 대한 필요성을 제기하게 되었으며 이를 만족시키기 위해 멀티타스킹 기능을 제공하는 실시간 커널과 다양한 실시간 요구사항을 위한 API를 제공하는 가상기계로 구성되는 mSROS와 다양한 개발 도구들을 포함하는 상용 개발환경을 적용한 공동 운영체제 플랫폼을 제공하게 되었다.

mSROS의 개발은 단일화된 하드웨어 및 운영체제라는 개념을 그 출발점으로 하여 모든 장치 제어기의 제어기들에서 하드웨어 공통 부분을 추출하고 이 부분에 mSROS를 공통으로 적용함으로써 공동 운영체제 플랫폼이 적용될 수 있도록 이루어졌다. 하드웨어는 CPU(MPC860-SAR)를 공동으로 사용하여 하드웨어의 공동부분을 구성하도록 하고 운영체제는 상용 개발환경을 변경 없이 사용할 수 있는 상용 실시간 커널(VxWorks)을 기반으로 교환기 응용 프로그램들의 요구사항을 수용하도록 하부커널을 추상화(abstraction)한 가상기계를 구축함으로써 하드웨어의 공동 부분에 공동으로 적용되도록 했다. 특히, 가상기계의 구축으

로 동작 환경에 영향이 없이 주후 자체 개발한 실시간 커널로 완전 교체가 가능함은 물론 다른 종류의 상용 커널로의 대체도 가능하여 장치 제어계용 응용 프로그램들의 이식성을 크게 향상시키는 구조를 갖는 공동 운영체제 플랫폼을 구성했다. 이에 따른 mSROS의 구조는 <그림 2>에 명시했다.

CCCP에 연결된 장치 제어기들에 탑재되는 mSROS는 <그림 2>에서와 같이 다양한 개발도구를 포함하는 상용 개발환경과 연동이 가능한 상용 실시간 커널을 기반으로 하고 mSROS 기능-교환기 응용을 위해 요구되는 고성능 고신뢰 IPC 통신을 위해 mSROS에서 사용되는 전용 IPC 프로토콜[4] 처리보드를 포함하는 IPC 기능, 다양한 실시간 관련 요구사항을 지원하기 위한 시간관리를 위한 time 기능, 장치 제어보드 고유의 기능을 실현하기 위해 필요한 각종 응용 프로그램들을 주기억 장치로 로딩하여 실행시키거나 이를 플래시 메모리에 쓸 수 있도록 하는 loader 기능 등으로 구성된 가상기계로 구성되어 응용 프로그래머들에게 가상기계에 기반한 인터페이스(mSROS API)를 제공한다.



<그림 2> mSROS의 구조

장치 제어기는 사용되는 CPU를 통일함으로써 구축된 공동 하드웨어와는 별도로 가입자의 특성 및 각 응용을 위해 각기 자신들만의 하드웨어(proprietary H/W)를 구성할 수 있는데 mSROS는 이 부분을 수용하기 위한 인터페이스를 제공하고 있어서 각 장치 제어기별 응용 프로그램 개발자들은 자신들만의 요구사항에 따른 부분들(proprietary BSP, proprietary device driver, proprietary API)을 간편하게 추가할 수 있다. 즉, 필요에 따라 하부 커널을 추상화한 가상기계가 제공하는 인터페이스가 사용자에 의해서 손쉽게 확장될 수 있도록 했다.

다음 장에서 가상기계의 구성 요소들과 이들의 API에 관해 기술한다.

4. 가상기계의 구성요소

가상기계는 교환기 응용 프로그램들이 필요에 따라 다양한 응용 프로그램들을 제작할 수 있도록 IPC, time,

loader, message, semaphore, process, memory, interrupt 기능들로 구성되어 있다. 가상기계를 구성하는 기능들 중에서 교환기 응용에 특히 중요한 IPC, time 및 loader 기능들은 독자적으로 개발했으며 나머지 기능들은 하부의 삼용 실시간 커널에서 제공되는 기본 기능들을 대체로 사용자의 요구사항을 수용하도록 추상화한 것이다. 가상기계는 동작환경에 영향 없이 하부 커널을 변경하거나 주후 자체 개발한 커널로의 대체가 손쉽게 이루어질 수 있게 한다.

가상기계의 구성 요소인 mSROS 기능들과 이들의 API에 대해 설명한다.

■ IPC 기능

정의된 교환기용 프로세서간 통신 인터페이스에 따라 시그널의 송수신을 담당하는 기능으로서 프로세스별 IPC 송신 및 수신 큐 관리, IPC 프로토콜 및 하드웨어 드라이버와의 인터페이스 기능 등을 갖는다. 특히, IPC는 실시간 시스템의 응용에 적합하도록 논-블록킹 송신 기능을 갖고 수신은 선택적 대기(selective waiting)[6]를 기반으로 블록킹 수신 및 논-블록킹 수신이 가능하다.

API	기능
ipc_link()	Initial signal 을 등록
ipc_rcvcase()	다수 개의 시그널 수신
ipc_rcvcase2()	특정 포트로부터 다수 개의 시그널을 수신
ipc_receive()	시그널 수신
ipc_send()	시그널 송신
ipc_send2()	특정 포트로 시그널 송신

■ time 기능

교환기와 같은 대형 실시간 시스템에서는 일정시간 후에 재처리, 일정한 주기로 계속적인 반복처리, 그리고 입출력 등 외부장치의 응답 최대시간을 정하고 그 시간 내에 정상적인 처리가 안될 경우의 장애처리 등이 필요하다. 이를 위해 프로세스의 요구에 따라 다수의 타이머를 동작시키고 일회 혹은 계속적인 타임아웃을 발생시키며 타임아웃 발생시에는 일정한 프로세스 혹은 프로시저를 수행시켜주는 기능을 제공한다.

API	기능
time_get()	시스템 시작 조사
time_set()	시스템 시작 설정
time_tod_get()	시스템 시작을 TOD 형태로 조사
time_tod_set()	시스템 시작을 TOD 형태로 설정
timer_cancel()	등록된 타이머를 삭제
timer_check()	타이머 프로세스의 상태 조사
timer_cycle()	일정 주기로 수행될 프로세스 등록
timer_set()	일정 시간 후에 수행될 프로세스 등록

timer_sleep()	프로세스의 수행을 일정시간 동안 멈춤
timer_sig()	타임아웃 시 그널 등록

*TOD : Time-of-Day

■ loader 기능

교환기 시스템의 고유한 기능은 각 서브 시스템별 응용 프로그램들로 이루어진다 각 서브 시스템들의 기능 수행에 필요한 프로그램들을 해당 시스템의 주기억장치에 적재하고 실행시켜 주는 기능을 갖는다.

API	기능
user_alloc()	사용자 블록의 적재
user_run()	사용자 블록의 실행
user_free()	사용자 블록의 unloading
user_load_done()	사용자 블록의 적재 완료
user_move()	사용자 블록의 이름 변경
user_printf()	사용자 블록의 출력문 제어
user_all_list()	적재된 모든 사용자 블록 조회

■ message 기능

동일 프로세서내에서 프로세스간의 동기 및 통신을 위한 메시지 통신 기능 등에 대한 사용자 정합 기능을 갖는다.

API	기능
message_create()	메시지 큐 생성
message_delete()	메시지 큐 삭제
message_receive()	메시지 수신
message_send()	메시지 송신

■ semaphore 기능

공유 데이터 접근시의 상호배제 영역(critical region) 처리나 프로세스간 동기를 위한 기능 등에 대한 사용자 정합 기능을 갖는다.

API	기능
sema_create()	세마포 생성
sema_delete()	세마포 삭제
sema_take()	세마포 획득
sema_give()	세마포 양도

■ process 기능

실시간 커널에서 독립적인 수행 모듈인 프로세스를 생성, 소멸하고 필요 시 임의로 상태를 변경하는 등의 프로세스 관리[7]에 관한 사용자 정합 기능을 갖는다.

API	기능
process_start()	프로세스의 생성
process_delete()	프로세스의 삭제
process_status()	프로세스의 상태 조사

process_exit()	프로세스의 소멸
process_delay()	프로세스의 지연
process_suspend()	프로세스 실행 정지
process_resume()	프로세스의 실행 재개
process_myid()	프로세스의 ID 조사
process_oirid()	프로세서의 IPC 주소 조사
process_isexist()	프로세스의 유효성 조사
process_name()	프로세스의 이름 조사
process_name2id()	프로세스의 이름에 해당하는 ID 조사
process_priority_get()	프로세스의 우선순위 조사
process_priority_set()	프로세스의 우선순위 변경
process_errno()	프리미티브 수행 시 에러 번호 조사

■ memory 기능

시스템내의 메모리 자원을 관리하는 기능으로서 프로그램 로딩 및 로딩 취소에서부터 프로그램의 수행 중에 할당 받고 되돌려주는 동적 메모리 관리에 대한 사용자 정합 기능을 갖는다.

API	기능
free()	메모리 회수
malloc()	메모리 할당

■ interrupt 기능

인터럽트 벡터 값 및 인터럽트 발생시 처리되어야 하는 프로시저들의 등록 및 취소 등을 관리에 대한 사용자 정합 기능을 갖는다.

API	기능
intr_cancel()	인터럽트 처리기 등록 취소
intr_establish()	인터럽트 처리기 등록

상기 기능 이외에도 ATM 시그널링을 수용할 수 있도록 다중 포트 UTOPIA 인터페이스를 갖는 하드웨어를 위한 시그널링 전용 송수신 기능을 별도로 지원하고 있다.

5. 결론

본 논문에서는 순차프로그램 특성을 갖는 PPOS를 장치 제어계의 각 하드웨어별로 일부 수정하여 휠웨이브를 개발함으로써 HANbit ACE64 시스템에서는 시스템 개발과 유지보수에 있어서 상당한 비효율성을 내포하고 있었다. 이러한 비효율성을 하드웨어 및 운영체제에 공통 개념을 적용한 공통 운영체제 플랫폼을 통해 해결했다. 특히, 다양한 개발 도구들을 포함한 상용 개발환경과의 연동이 가능한 멀티타스킹 실시간 커널을 기반으로 하여 교환기 응용 프로그램들의 요구사항을 수용하도록 하부커널을 추상화(abstraction)한 가상기계를 구축하여 동작 환경에 영향이 없이 추후 자체 개발한 실시간 커널로 완전 교체가 가능함은 물론 다른 종류의 상용 커널로의 대체도 가능하여 장치

제어계용 응용 프로그램들의 이식성을 크게 향상시키는 구조를 갖는 공통 운영체제 플랫폼을 구성했다. 가상기계는 교환기를 위해 요구되는 고성능 고신뢰 IPC 관리기능, 다양한 실시간 관련 요구사항을 지원하기 위한 시간관리 기능, 장치제어보드의 기능 실현을 위한 각종 응용프로그램들을 주기억 장치로 적재하는 적재기(loader) 등을 포함한다. 이런 접근방법을 통하여 HANbit ACE256 시스템에서는 기존의 비효율성 및 성능을 효과적으로 개선할 수 있었다.

향후 현재의 상용 하부 커널을 대체할 자체 커널을 개발할 예정이다.

참고 문헌

- [1] Kim YB, "An Architecture of Scalable ATM Switching and Its Call Processing Capacity Estimation", ETRI Journal, v.18, n.3, pp. 23-36, Oct. 1996.
- [2] 정부금, 차영준, 김형환, 전성익, 조주현, "실시간 마이크로 커널의 설계", 대만전자공학회 추계학술 발표대회 논문집, Vol. 17, No. 2, pp. 514-517, Nov. 19, 1994.
- [3] 전성익, 조주현, 한치문, "TDX-ATM 용 프로세서 이종화 제어 방식에 관한 연구", 전자공학회추계학술 대회, 제 19 권 2 호, Nov.23, 1996.
- [4] 김형환, 전성익, 차영준, 조주현, "HLRP: A High-performance Light-weight Real-time Protocol", IEICE'95, pp.467-470, Aug. 7-12, 1995.
- [5] Park HS, "ATM Interface Technologies for an ATM Switching System", ETRI Journal, v.18, n.4, pp. 229-244, Jan. 1997.
- [6] Alan Burns, Andy Wellings, "Real-Time Systems and Programming Languages", Addison Wesley, 1996.
- [7] 정부금, 차영준, 전성익, 조주현, "분산 실시간 시스템을 위한 멀티 유저 프로그래밍의 설계 및 구현", 전자공학회 충청지부 학계학술대회, pp. 40-43, May 30, 1997.