윈도우즈 서비스팩(Service Pack)에서 실시간성 지원 방안 연구

A Study to support real-time on Windows Service Pack

박 지 윤, 조 아 라, 이 철 훈
충남대학교 컴퓨터공학과

Park Ji-Yoon, Jo Ah-Ra, Lee Cheol-Hoon
Chungnam National University

요약

범용 운영체제 윈도우즈는 라운드로빈(round-robin)스케줄링 방식을 사용하고 있다. 라운드로빈 스케줄링 방식은 높은 우선순위 태스크에 대한 빠른 응답성을 보장할 수 없어 실시간성을 제공하지 못한다. 이를 해결하기 위해 윈도우즈에서 실시간성을 제공하기 위한 RTiK-MP(Real Time implant Kernel - Multi Processor)가 개발되었다. 기존 RTiK-MP는 멀티프로세서 윈도우즈 환경에서 정상동작하게 되어있으나 서비스팩(Service Pack)환경에서는 Local APIC 타이머 레지스터의 값이 초기화되면서 윈도우즈에 실시간성을 제공하지 못하는 문제점이 있다. 본 논문에서는 윈도우즈 서비스팩에서 발생하는 RTiK-MP의 문제점에 대해 설명하고 이에 대한 해결방안을 기술하여 윈도우즈 서비스팩에서 실시간성을 제공하기 위한 방안을 연구하였다.

I. 서론

실시간성이란 어떤 수행에 대한 연산이 예측 가능한 시간 내에 완료되는 것을 말한다. 일반적으로 사용하는 범용 운영체제인 윈도우즈의 경우 실시간성을 제공하지 못하므로 RTX나 INtime같은 고가의 서드파티를 구입하여 사용해야 한다. 이러한 경우 개발 비용이 증가하는 문제점이 있어 이를 해결하기 위해 디바이스 드라이버 형태의 RTiK-MP가 개발되었다. RTiK-MP는 x86 기반의 하드웨어 자원인 Local APIC (Advanced Programmable Interrupt Controller) 타이머 레지스터를 이용하여 윈도우와 독립적인 타이머 인터럽트를 발생시켜 실시간성을 제공하고 있으며 현재 멀티코어 윈도우즈 환경에서 정상동작하도록 되어있다. 최근 윈도우즈 서비스팩으로 업데이트되면서 Local APIC의 타이머 레지스터의 값이 0으로 초기화 되는 문제점이 발견되어 이를 해결하기 위해 RTiK-MP를 수정할 필요가 있다.

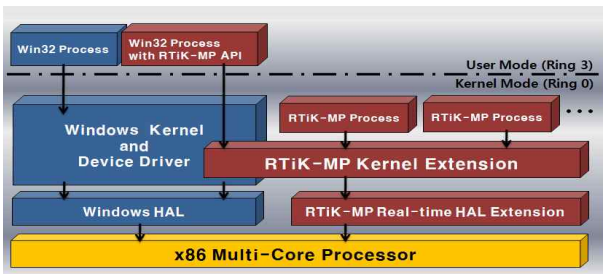
본 논문에서는 윈도우즈 서비스팩에서 실시간성을 제공하기 위한 RTiK-MP를 연구 하였다.

본 논문의 구성으로는 먼저 2장에서 관련 연구로 기존 RTiK-MP와 Local APIC에 대하여 설명하고, 3장에서는 윈도우즈 서비스팩에서 실시간성을 제공하기 위한 방법과 실험 환경 및 결과를 기술하였으며, 마지막 4장에서는 결론을 맺는다.

Ⅱ. 관련 연구

2.1 RTiK-MP

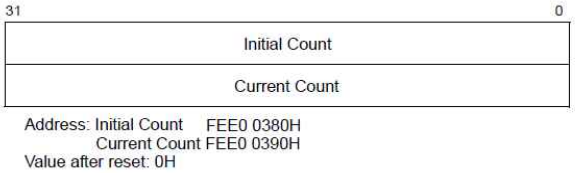
멀티프로세서 기반 윈도우즈에 실시간성을 제공하기 위한 RTiK-MP는 [그림 1]과 같이 디바이스 드라이버 형태로 구현되었다. 윈도우즈의 커널 및 하드웨어 자원을 이용하도록 윈도우즈와는 별개의 HAL(Hardware Abstraction Layer)을 가지는 형태로 설계되었다[1].



▶▶ 그림 1. RTiK-MP의 구조

2.2 Local APIC

Local APIC는 프로그램 가능한 인터럽트 컨트롤러로, x86 기반의 아키텍처에서 제공되며 입력된 인터럽트 신호를 CPU 코어에 전달한다. [그림 2]는 Local APIC 타이머 관련 레지스터를 나타낸 것이다. 타이머의 주기는 Initial Count 값에 의해 결정되며, Initial Count 값이 Current Count에 등록되어 주기적인 동작을 수행한다 [2][3].



▶▶ 그림 2. Local APIC 타이머 관련 레지스터 구조

Ⅲ. 본론

3.1 Local APIC 접근을 위한 코드 구현

윈도우즈에 실시간성 제공을 위한 기존 RTiK-MP의 동작 방식은 앞서 언급한 Local APIC 타이머 관련 레지스터를 이용한다. 하지만 서비스팩으로 업데이트 후 타이머 관련 레지스터의 Initial Count 값이 0으로 초기화되어 타이머 인터럽트 발생이 불가능해 진다. 이를 해결하기 위해 서비스팩 업데이트 전의 Initial Count 값을 외부 시스템 파일에 저장하고 서비스팩 업데이트 후 RTiK-MP를 실행하여 저장해둔 외부 시스템 파일에 값을 읽어 다시 Initial Count를 세팅하게 하였다. 업데이트 이전의 Initial Count 값을 얻으려면 먼저 유저 영역에서 커널 영역에 접근 할 수 있어야 한다. 즉, 디바이스 드라이버와 통신이 가능하여야 하는데 이를 위해 [그림 3]과 같은 미리 정의된 IOCTL(I/O Control Code) 코드를 이용 할 수 있다.

```
#define IOCTL_RTiK_MP_WRITE_APIC_VALUE CTL_CODE
(FILE_DEVICE_WDFRTIK, 2056, METHOD_OUT_DIRECT, FILE_ANY_ACCESS)
```

▶▶ 그림 3. Local APIC에 접근을 위한 IOCTL 코드

[그림 4]처럼 디바이스 드라이버와 통신하는 함수 호출 시 미리 정의한 IOCTL 코드와 커널로부터 받을 Initial Count 값이 저장 될 변수의 주소를 인자로 넣는다.

```
DeviceIoControl(DriverHandle,
    IOCTL_RTiK_MP_WRITE_APIC_VALUE,
    NULL,
    0,
    &apicValue,
    sizeof(apicValue),
    &dwOut,
    NULL);
```

▶▶ 그림 4. 디바이스 드라이버와 통신을 위한 함수 호출

Ⅳ. 실험 환경 및 결과

4.1 실험 환경

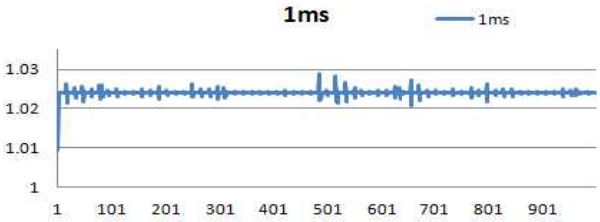
윈도우즈 서비스팩 환경에서 실시간성을 제공하기 위한 RTiK-MP의 실험환경은 표-1과 같다.

표 1. 실험 환경

	HOST
CPU	Intel Core2 CPU 6300 @1.86GHz
OS	Windows 7 Ultimate K
Development Tool	MS Visual Studio 2008

4.2 실험 결과

윈도우즈 서비스팩 환경에서 RTiK-MP의 주기 측정을 위해 HOST에 RTiK-MP를 이식하고 1ms의 주기를 설정하였으며, [그림 5]는 측정된 주기를 그래프로 나타낸 것이다. [표 2]는 그 결과의 최대, 최소값을 계산한 것이다. [표 2]에서 볼 수 있듯이 최소주기의 오차가 약 2% 이내로 정상동작 함을 확인 할 수 있다.



▶▶ 그림 5. 1ms 주기 실험결과

표 2. 1ms 주기 최대값/최소값

	1ms 주기 테스트 결과
최대값	1.009495 ms
최소값	1.028844 ms

Ⅴ. 결론

본 논문에서는 윈도우즈에서 서비스팩으로 업데이트할 경우 Local APIC 타이머 레지스터 값이 초기화되어 RTiK-MP를 정상적으로 실행시키지 못하는 문제점을 해결하기 위한 연구를 하였다. 업데이트되기 전 타이머 레지스터의 Initial Count 값을 외부 시스템 파일에 저장해 놓고, 업데이트 이 후 RTiK-MP 실행 시 외부 파일을 참조하여 Initial Count 값을 다시 설정하여 실시간성을 제공하도록 하였다. 이 과정에서 타이머 레지스터에 접근이 필요하게 되는데, 유저 영역에서 커널 영역으로의 접근은 IOCTL코드를 통한 디바이스 드라이버와의 통신으로 가능하게 하였다.

향후 연구과제로는 기존에 연구 개발 되었던 RTiK-MP를 이용한 미들웨어나 TCP/IP 및 시리얼 통신이 본 논문에 제시된 방안을 통하여 윈도우즈 서비스팩 환경에서 정상동작 하는지에 대한 성능 검증이 이루어져야 할 것이다.

■ 참 고 문 헌 ■

[1] 조아라, 이승훈, 이철훈, “RTiK-MP의 Export Driver 지원을 위한 방안 연구”, 한국콘텐츠학회, 제1권, 제1호, pp. 55-56, 2012.

[2] 송창인, 이승훈, 이철훈, “멀티프로세서 윈도우즈 XP 상에서 실시간성 지원”, 한국컴퓨터정보학회, 제20권, 제1호, pp.21-24, 2012.

[3] Intel, “Intel 64 and IA-32 Architecture Software Developer's Manual Volume 3B : Systems Programming Guide,” 2012.