PC에서의 인터럽트 지연 측정을 통한 실시간성 분석

송정애° 황소영 김영호
부 산 대 학 교 전자계산학과
{jasong, youngox}@juno.cs.pusan.ac.kr, yhkim@hyowon.pusan.ac.kr

Real-time characteristic analysis through measure of interrupt latency on PC

Jung-Ae Song° Soyoung Hwang Youngho Kim
Dept. of Computer Science , Pusan National University

요 약

최근 들어 IBM 호환 PC의 성능이 급속도로 향상됨에 따라 이를 기반으로 하는 실시간 시스템의 필요성이 대두되고 있다. 이는 기존의 실시간 제어 시스템이 특별하게 설계된 하드웨어를 요구하는 것에 반해 널리 사용되는 하드웨어를 이용함으로써 응용 프로그램의 이식성을 높여 주는 큰 장점을 가지고 있다. 이러한 PC를 실시간 시스템으로 개발하게 위해서는, PC에서의 실시간 분석이 선행 되어야 한다.

실시간 시스템을 위한 중요한 요소는 인터럽트가 발생 하였을 때 빠른 시간에 인터럽트 핸들러를 불러 주는 것이다. 즉, 인터럽트 발생에서 인터럽트 핸들러가 불리기까지의 시간인 인터럽트지연(interrupt latency) 시간이 짧아야 한다. 따라서, 본 논문에서는 실시간 시스템의 성능에 영향을 주는 인터럽트 지연을 측정하고, 이를 통하여 PC에서의 인터럽트 특성 및 실시간성을 분석한다.

1. 서론

일반 PC의 성능이 향상됨에 따라 이를 기반으로 하는 실시간 시스템의 필요성이 대두되고 있다. 이는 기존의 실시간 제어 시스템보다 PC가 가지고 있는 다음과 같은 장점 때문이다. 첫째, PC의 하드웨어와 소프트웨어 기술이 급속히 발전하여 PC의 안정성과 성능이 향상되고 가격은 낮아지고 있다. 둘째, PC는 다수의 주변기기와 소프트웨어들을 저가에 활용할 수 있다. 또한 사용환경이 실시간 전용 제어 시스템에 비해 우수하다는 것이다. 따라서 이를 실시간 제어 시스템으로 활용할 경우 종래의 다른 전용 시스템에 비해 고급의 기능과 경제적 효과를 기대 할 수 있다. 그러므로, 일반 PC는 본래 실시간 제어를 목적으로 개발되어지지 않았음에도 불구하고, 이와 같은 잇점들 때문에 PC를 실시간 제어 시스템으로 활용 하려는 연구가 진행 중이다. 그리고, PC를 실시간 시스템으로 개발하게 위해서는 앞서, PC에서의 실시간 분석이 요구된다. 따라서 본 논문에서는 실시간성을 평가하는 주요 요소인 인터럽트 지연 측정을 통하여 PC에서의 인터럽트 특성 및 실시간성을 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 실시간 시스템과 PC 운영체제에 대해서 알아보고, 3장에서는 시스템의 실시간성을 평가하는 요소 및 방법에 대하서 기술한다. 4장에서는 인터럽트 지연 측정 및 결과 분석, 그리고 마지막 5장에서는 결론 및 향후 계획을 기술한다.

2. 실시간 시스템과 PC 운영체제

2.1 실시간 시스템

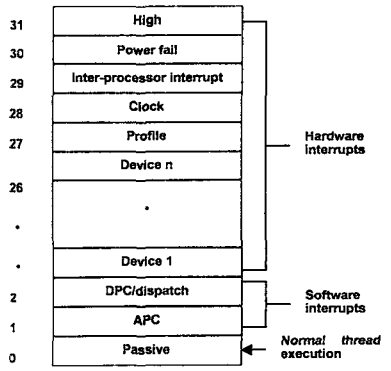
실시간 시스템은 인터럽트가 발생했을 때 정해진 시간 이내에 처리되는 것을 보장하는 시스템이다. 즉 이벤트에 대한 반응이 빠르고, 높은 우선순위의 이벤트가 그보다 낮은 우선순위 이벤트보다 먼저 수행되며, 이벤트를 놓치는 일이 결코 일어나서는 안 되는 시스템을 말한다. 그러기 위해서 인터럽트가 발생하였을 때 빠른 시간에 인터럽트 핸들러를 불러주어야한다. 즉, 인터럽트 지연 시간이 짧아야 한다. 그리고 우선순위가 보다 높은 이벤트가 발생하면 이를 먼저 처리하도록 한다. 그래서 대부분의 실시간 시스템은 우선순위를 기반으로한 선점 스케줄링(priority-based preemptive scheduling)을 한다.

그러나, 짧은 인터럽트 지연 시간이 실시간 시스템의 요구조건을 항상 만족시키는 것은 아니다. 일반적으로 이는 시스템의 평균 응답시간을 최소화하지만, 실시간 시스템에서 요구되는 예측가능성을 보장하지는 않는다. 예측가능성이란 시스템의 명세에 정의된 고장이나 작업 부하 조건에서 태스크의 종료시한 만족을 보장하는 것을 의미한다. 따라서 실시간 시스템을 위해서는 어떤 프로세서의 부하에서도 인터럽트 지연 시간이 안정적이어야 한다[1][2][11].

2.2 PC 운영체제

PC의 실시간성 분석에 있어서는 PC기반에서 운용 될 운영체제가 무엇보다 중요하다. 따라서, 데스크탑 컴퓨터에서 널리 사용되는 범용 운영체제의 하나인 윈도우 2000의 내부 구조에 대해서 간략히 알아본다.

윈도우즈 2000는 원래 실시간 운영체제로 잘 알려져 있는 VxWorks나 QNX와 같이 마이크로 커널 설계 기반이다.



[그림1] 우선순위 구조

[그림1]은 윈도우즈 2000에서 제공되는 우선순위 시스템의 전반적인 구조이다. 인터럽트 컨트롤러들은 인터럽트에 우선권의 수준을 부여하지만, 윈도우즈 2000도 자체적으로 인터럽트 우선권의 스키마를 가지고 있다. 이를 IRQL(Interrupt Request Level)이라고 한다. 커널은 IRQL을 0에서 31까지의 숫자로 표현하고, 높은 숫자는 더 높은 우선권을 의미하게 된다. 인터럽트는 우선권에 따라서 서비스되므로, 높은 우선권을 가진 인터럽트는 낮은 우선권의 인터럽트의 서비스를 선점(preempt)할 수 있다.

[그림1]에서 보는 바와 같이, 시스템 클럭이나 다른 하드웨어 장치들에 대한 인터럽트 서비스 루틴(ISR)이 가장 우선 순위가 높다. 그 다음은 디스패치 또는 DPC(Deferred Precedure Call) 그리고, APC(Asynchronos Procedure call) 순이다. 스레드가 더 이상 실행되지 않을 때, 예를 들어 스레드가 종료되었거나 자발적으로 대기 상태로 들어 간 경우 커널은 직접적으로 디스패치를 호출해서 컨텍스트 전환을 요청하게 된다. 하지만, 때때로 커널은 코드들의 많은 레이어 속에서 스케줄링을 다시 해야 한다. 이런 경우에 가장 이상적인 해결 방법은 커널이 현재 활동을 마칠 때까지 연기된 후에 실행될 디스패칭을 요청하는 것이다. 바로 이때 사용 된 것이 DPC 소프트웨어 인터럽트이다. 그리고 APC는 사용자 프로그램이나 시스템 코드가 특정 사용자 스레드의 컨텍스트에서 즉, 특정 프로세스 주소 공간에서 실행 될 수 있도록 해주는 해 주는 소프트웨어 인터럽트이다[3][4][5].

범용 운영체제로써 전반적인 시스템 성능에 뛰어나게 고안 설계된 윈도우즈 2000은 장치의 IRQ를 통제해서 우선권을 높이지도 못하며 사용자 모드 응용 프로그램은 항상 수동적 수준에서 실행되므로, 윈도우즈 2000은 실시간 운영체제로 항상 적당하다고는 말 할 수는 없다. 하지만, 실시간 운영체제 (RTOS, Real Time Operating System)같은 방법으로 우선순위를 기반으로한 선점 스케줄링(Priority-Based Preemptive Scheduling)을 할 뿐만 아니라 인터럽트 서비스 루틴(ISR, Interrupt Service Routine)이 매우 효율적인 방법으로 디자인 되어져 있다[1]. 이런 요소들을 가지고 있는 윈도우즈 2000이 어느 정도의 실시간성을 보장하는 지에 평가가 요구된다. 이 평가를 통해서 실시간 시스템으로의 발전 가능성을 가늠 할 수 있다.

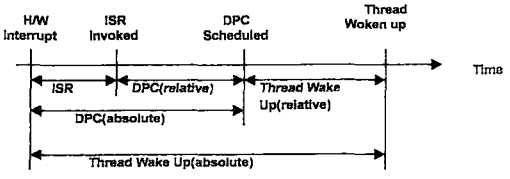
3. 인터럽트 지연 측정

실시간 시스템의 중요한 구성요소인 인터럽트는 어떤 외부 이벤트가 일어날 때 실시간 시스템에 알리는데 사용되는 메커니즘이다. 실시간 시스템은 이러한 이벤트에 빨리 응답할 수 있어야 한다. 인터럽트는 인터럽트가 일어날 때 운영체제에 의해 자동적으로 실행되는 소프트웨어 루틴인 인터럽트 서비스 루틴(ISR)에 의해 다루어지고, 이는 인터럽트에 응답함에 있어, 적합한 실행을 행할 책임이 있다.

실시간 성능의 중요 평가 요소는 하드웨어 인터럽트가 발생한 직후부터 서비스가 시작되기까지의 시간인 인터럽트 지연(Interrupt Latency)이다[5][6][7][9].

3.1 측정 요소

인터럽트 핸들을 특성짓는 기본적인 3가지의 지연에 대해서 측정한다.



[그림2] 인터럽트 지연

- 인터럽트 서비스 루틴 지연(ISR Latency)
- DPC 지연(DPC Latency)
- 스레드 웨이크업 지연(Thread Wake Up latecy)

3.2 측정 방법

대부분의 인터럽트 지연은 고해상도 타이머를 제공하는 하드웨어 보드의 도움으로 실행된다. 그러므로, 이러한 측정은 새로운 플랫폼에서 실행되지는 못 하고, 프로그래머블 타이머와 함께 로컬 APIC를 포함한 모든 P6 family의 펜티엄 프로세서에서 가능하다. 본 실험도 펜티엄 프로세서에 포함되어져 있는 두 개의 고해상도 성능 카운터를 통해서 시간 측정을 하였다[5][7][8].

4. 측정 및 결과 분석

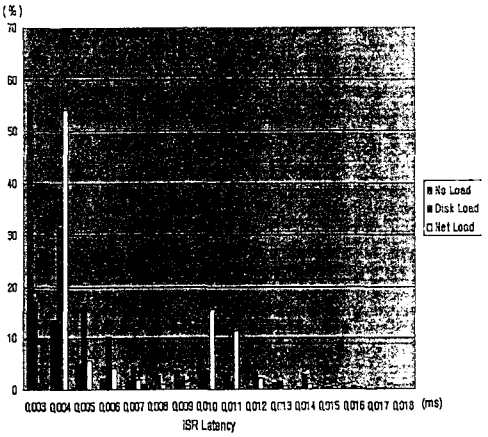
대상 시스템	Load 조건	실험 환경
· Intel Pentium III 550MHz · 256Mbyte Memory · IBM DPTA-37205020GB UDMA66 7200RPM · LG CRD-8402B 40X EIDE · Intel 100MHz PCI · Windows 2000 Professional	· Idle (No Load) : 유휴 상태 · Disk Load 디스크에서 디스크로의 파일 복사 · Net Load : 네트워크 활성	· 측정 시간 5분 · 10ms 인터럽트 주기 · 시간 임계적인 실시간 스케줄링 프라퍼티 (우선순위 레벨 31) · IRQL 29

4.1 ISR latency

다음은 실제 하드웨어 인터럽트가 발생한 직후부터 인터럽트 서비스 루틴이 수행되기까지의 지연시간을 측정값이다.

[표1] ISR latency

Load	Avg (*ms*)	Max (*ms*)
No Load	0.005	0.018
Disk Load	0.006	0.018
Net Load	0.006	0.018
Average	0.0057	0.018



[그림3] ISR Latency Sample

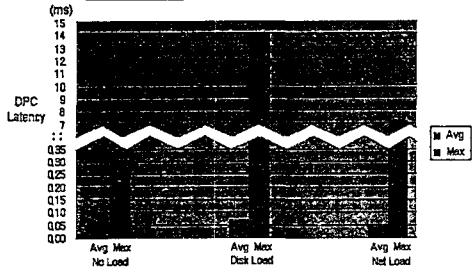
[표1]과 [그림3]에서 보는 바와 같이, ISR 지연은 서로 다른 로드에서도 비교적 일정한 지연 시간을 보여주고 있다. 이 결과만 가지고는 ISR지연 시간이 폐 결정적(determinic)이고, 예측 가능(predicitible)하다고 볼 수 있다. 하지만, ISR에서 모든 경성 실시간 계산(hard Real-time computations)을 유지하는 것에는 제한이 있다. 예를 들어, ISR은 하드 드라이브를 액세스를 할 수 없고, 뿐 만 아니라, GUI code를 실행 할 수도 없다. 따라서, ISR latency만으로 실시간성을 얘기 하기는 어렵다.

4.2 DPC latency

다음은 실제 하드웨어 인터럽트가 발생한 직후부터 DPC 루틴이 수행되기까지의 지연시간 값이다.

[표2] DPC Latency

Load	Avg (ms)	Max (ms)
No Load	0.042	0.304
Disk Load	0.067	14.018
Net Load	0.042	7.048
Average	0.050	7.123



[그림4] DPC Latency

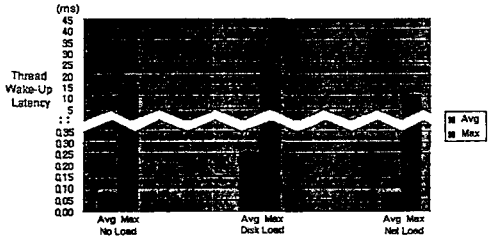
[표2]와 [그림4]를 통해 평균 지연 시간값에 비해 최대 지연 시간값이 200배가 넘는 경우도 있다. 그리고 각 로드에 따라서, 최대 지연 시간에도 엄청난 차이가 나타난다. 이는 DPC 계층에서 이슈되는 일들은 예측 가능성이 힘듬을 보여주는 것이다. 모든 사용자 우선순위들은 DPC 계층 아래이므로, 이 것은 또한 사용자 프로그램등에도 영향을 줄 것이다.

4.3 Thread Wake-up Latency

Thread Wake up Latency은 실제 하드웨어 인터럽트가 발생한 직후부터 스레드 핸들링이 수행되기까지의 지연시간 값이다.

[표3] Thread Wake-Up Latency

Load	Avg (ms)	Max (ms)
No Load	0.078	15.282
Disk Load	0.255	29.245
Net Load	0.078	10.343
Average	0.137	18.29



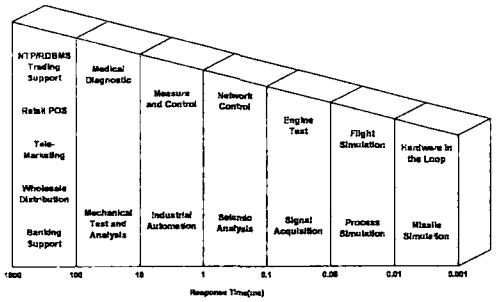
[그림5] Thread Wake-Up Latency

위 실험 결과들을 통해서, 평균 인터럽트 지연값이 수십 마이크로 세컨드로 빠르다는 것을 알 수 있다. 이는 PC가 범용 시스템으로써 전반적인 안정성과 신뢰성을 제공해 준다는 것을 알 수 있다. 하지만, 간헐적으로 발생하는 최대 지연으로 실시간 어플리케이션에는 다소 문제가 따른다는 것을 알 수 있다. 따라서, 서로 다른 환경에서의 지속적인 측정을 통해서, 인터럽트 지연을 야기 하는 요소에 대한 분석이 계속 되어야 한다.

5. 결론 및 향후 계획

하드웨어와 소프트웨어 기술이 급속히 발전하여 PC의 안정성과 성능이 향상되고 가격은 낮아지고, 다수의 주변기기와 소프트웨어들을 저가에 활용 할 수 있다는 장점 때문에 PC가 본래 실시간 제어를 목적으로 개발되어지지 않았음에도 실시간 시스템으로 활용하려는 연구가 진행되고 있다. 본 논문에서는 윈도우즈 2000을 운용하는 일반 PC에서의 실시간성을 평가하는 기본적인 요소인 인터럽트 지연(interrupt latency)에 대해서 측정을 하였다. 이를 통해서 어느 정도까지 실시간성이 보장이 되는 지를 알 수 있다. 즉, 어느 정도까지의 실시간 어플리케이션에 적용 가능한지를 알 수 있다.

아래 [그림6]은 서로 다른 형태의 어플리케이션들의 전형적인 실시간 제약성 요구를 보여준다[12].



[그림 6] Sample Real-Time Requirements

[그림6]에서 보는 바와 같이 어플리케이션마다 서로 다른 시간 제약성을 가지고 있다. 예를 들어, 미사일 시뮬레이션과 같은 마이크로세컨드(μs)정도의 실시간을 요구하는 어플리케이션이 있는 반면에 은행 업무에 사용되는 어플리케이션과 같이 초단위의 정밀도를 요구하는 어플리케이션도 많이 있다. 후자의 경우에는 실시간 전용 제어 시스템이 아니더라고, 일반 PC에서도 충분히 실시간성을 보장해 줄 수 있다는 것을 실험 결과를 통해서 알 수 있다.

앞으로의 계획은 보다 정밀한 측정과 아울러, 장시간 동안에 여러 다른 로드 하에서의 인터럽트 지연을 측정해 봄으로써 인터럽트 지연을 야기 시키는 요소, 즉 실시간성을 저해하는 요소를 추출하고, 이를 바탕으로 PC 기반의 실시간 시스템을 디자인하는데 도움을 줄 것이다.

6. 참고 문헌

[1] Microsoft Corporation, "Real-Time Systems and Microsoft Windows NT", Microsoft Developer Network, June 29, 1995, http://premium.microsoft.com/msdn/library/bkgrnd/html/realtime.htm

[2] Phillip A. Laplante, "Real-time systems design and analysis",1992

[3] David A. Solomon, "Inside Windows 2000", Microsoft Press, 2000

[4] Water Oney," Programming the Microsoft Wodnows Drover Model", 2000

[5] Allan Baril,"Using Wondows NT in Real-time Systems"

[6] J. Bradley, "The measured Performance of Personal Computer Operating Systems", ACM Transactions on Computer Systems, Vol.14, No.1, February 1996, pages3-40.

[7] RadiSys Corporation, "INtime Interrupt Latency Report", Technical Paper, November, 1997, http://www.radisys.com/cgi-bin/filelib/showdir/INtime/White_Papers#here

[8] Intel Architecture Software Developer's manual Volume 3: System Programming

[9] 황소영, "임베디드 시스템에서의 인터럽트 처리 특성 분석", 부산대학교 이학석사 학위 논문, 2001

[10] 송정애, 김종성, 황소영, 김영호," 정밀 시각을 위한 오류요소 분석 및 일반 PC에서의 측정 방법", 정보 과학회 추계 학술대회, 2001

[11] http://linuxkernel.net

[12] Sam Dekey, "Making Windows NT a Real-Time Solution-A Technical Overview", National Instruments