윈도우즈 시스템에서의 시각 조정 방안

이성진, 김영호
부산대학교, 전자계산학과

A Clock Adjustment Method of Windows System

Seong Jin Lee, Youngho Kim
Department of Computer Science, Pusan National University

컴퓨터 시스템에 내재된 시각 장치는 제반 동작의 기본이 되며, 시간과 관련된 연산들에 직접적인 영향을 준다. 시각 장치는 온도나 습도와 같은 주위 환경 요인에 의해 오차를 가지며, 보다 높은 시각 정확도를 얻기 위해서는 시각 장치를 동기시키는 알고리즘이 필요하다. 본 논문에서는 윈도우즈 시스템 시각장치의 정확도를 향상시키기 위해 동기 알고리즘을 적용한 시각 조정 방안과, 제시된 조정 방안을 적용한 알고리즘의 실험과 결과에 대해 제시한다 사용된 알고리즘은 피드백 제어를 사용하여, 분주된 오실레이터의 주파수를 GPS로부터의 1PPS 신호에 동기시키는 FLL로 구성하였다. 조정방안의 구현 및 실험은 Windows 95 상에서 실행되었다

1. 서론

병렬 처리 시스템이나 분산처리 시스템 상에서 서로 동기된 시각들은 시각 측정, 프로세스간 동기 등 여러 목적을 위해 중요하게 사용된다.[5][6][7] 분산 처리 시스템 상에서는 분산된 어플리케이션이 올바른 결과를 만들이 내기 위해서는 각 노드에서 유지하고 있는 시각을 동기 시키기 위한 방법론이 필요하다.[8]

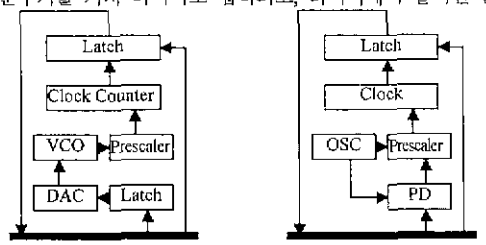
병렬 처리 시스템상에서 서로 동기된 시각은 시스템 성능 측정, 프로세스간 스케줄링, 프로그램 추적, 디버깅 등에 사용될 수 있다 IBM의 SP2와 같은 시스템에서는 시각 동기를 위해 sptimed라는 프로그램이 구현되어 실험 된 바 있으며, NTP 대한 성능 측정이 이루어진 바 있다. [4]

이러한 시각 동기의 가장 중심이 되는 것은 외부 시각원과 그 시각을 유지하고 있는 시스템 내 시각간의 동기이다. 본 논문에서는, 표준 PC에서 동작하는 윈도우즈 시스템상에서 외부 시각원과 동기시키기 위한 알고리즘과 이에 대한 실험 결과, 결과 분석에 대해 논의한다 논문의 전반부에서는 두 시각 장치에 있어서 동기에 관한 일반적에 문제에 대해 기술하고 기존의 시각 동기 모델에 대해 논의한다 후반부에서는, 일반적 시각 장치 모델을 포함하는 윈도우즈 시스템에서의 시각 동기 문제와 동기 알고리즘에 대해 논의한다

2. 일반적 컴퓨터 시각 동기 문제

2.1. 일반적 모델 및 문제점

컴퓨터 시각장치는 사용되는 오실레이터에 따라 두가지로 나뉜다. [그림 1]에서 보인바와 같이, 주파수를 가변시킬 수 있는 VCO를 사용한 것과, 가변이 불가능한 오실레이터로 나눌 수 있다 두 모델 모두 분주기를 거쳐 타이머로 입력되고, 타이머에서 출력된 클럭은



[그림 1] 시각 장치 모델

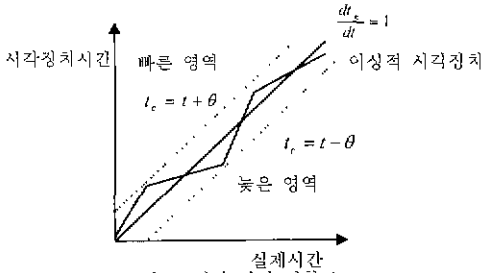
소프트웨어의 클럭틱수를 증가시킨다.[2][3]

시각장치는 환경요인, 특히 온도, 습도로 인해 항상 오차를 가진다. 최대 drift 율이 ρ, 시각 옵셋이 0 인 경우 <식 1>과 같이 표현할 수 있다.

$$1-\rho \le \frac{dt_c}{dt} \le 1+\rho \qquad \text{<식 1>}$$

다음 [그림 2]에서 보인 바와 같이, 완전히 동기된 시각을 나타내는 직선에 비해 빠른 시각은 이보다 큰 기울기의 직선, 늦은 시각은 작은 기울기의 직선으로 그려진다.

[그림 2]에서 기울기가 1인 직선과 겹쳐 그려진 그래프는 주기적인 보정을 통해 동기되고 있는 한 시각 장치의 시각을 표현한 것이다.. 증가된 주파수의 의해 참조 시각보다 빨라졌을 경우 주파수를 감소시키고, 이로 인해 느려졌을 경우 다시 주파수를 증가시켜, 시각 옵셋이 일정한 경계를 넘지 않도록 계속해서 보정을 하고 있음을 보이고 있다.

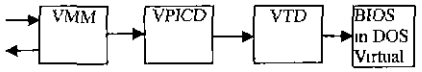


[그림 2] 시각 장치 정확도

2.2. 윈도우즈 동작

표준 PC 상에서 운영되는 윈도우즈 시스템의 시각 장치는 오실레이터, 분주기, 타이머, 소프트웨어 클럭으로 구성되며, 72.8Hz의 기본 주파수의 타이머 인터럽트로 시간을 유지한다. 전체 구성은 [그림 3]과 같다

윈도우즈 시스템은 이건의 도스 시스템보다 복잡한 과정을 거쳐 하드웨어 인터럽트를 처리한다. 우선 버스상에 인터럽트 요청되면 먼저 VMM(Vritual Machine Manager)에 의해 처리된다.[10][11][12] 여기서 VMM 은 인터럽트 처리의 시작을 위해 스택과 데이터 세그먼트의 초기화를 한후 VPICD(Virutal Programmable Interrupt Controler Device)로 처리를 넘긴다. VPICD 는 VMM 으로부터 제어권을 넘겨받은 후 PIC 에 대한 처리를 후, 각 하드웨어의 처리를 담당하는 Virtual Device Driver 에 제어권을 넘긴다



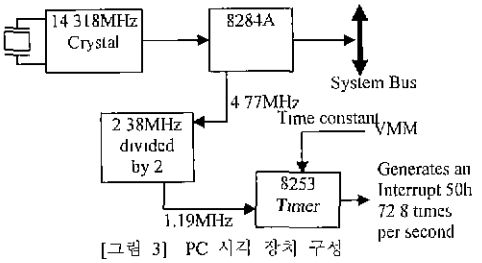
[그림 4] 윈도우 시각 신호 처리 과정

타이머 인터럽트 50h 는 다른 하드웨어 인터럽트의 처리과정과는 달리 VMM 에서 스케줄러에 의한 과정이 추가된다. 스케줄러는 스케줄정책을 위해 타이머를 사용기 때문에, 타이머의 시정수를 조정한다 [그림 4]에서 보인바와 같이, VMM VPICD 의 처리가 끝난후, VTD(Virtual Timer Device)가 제어권을 넘겨받아 가상 타이머의 값을 조정한다 VTD 는 윈도우즈 시스템의 시각을 유지하면서, 55ms 마다 DOS 인터럽트 8h 를 호출한다 [9]

이렇게 처리되는 인터럽트 50h 는 VTD 내 클릭틱 계수를 증가시키고 윈도우즈 시스템 시각을 갱신한다 클릭틱의 계수는, 매 클릭틱 발생시 1 증가시키는 도스에서의 방식과는 달리 타이미에 설정되는 시정수 자체를 더한다 즉, 클럭틱 계수 자체는 시스템이 시작한 이래 누적된 시정수의 값이다 시스템 시각은 이 계수값을 1193 으로 나눈 값으로 설정된다 이렇게 구해진 시스템 시각은 시스템이 시작한 이후 현재까지의 시간이 된다

2.3. 시각원

시각 장치 동기 문제에 있어서 동기 되는 내부 시각 장치의 정확도를 결정하는 요인 중 하나는 참조되는 외부 시각 장치의 정확도이다 내부 시각 장치의 정확도를 향상시키기 위해서는 대략 수십 내지 수백배 이상의 정확도를 가진 외부 시각원을 사용해야 한다 이러한 시각원들 중 일반적으로 사용되는 외부 시각원으로서 GPS.

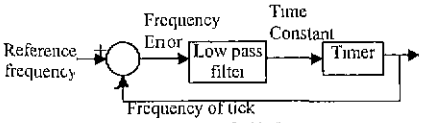


[그림 3] PC 시각 장치 구성

GOES, Radio Clock, ACTS(Automated Computer Time Service)에 의해 서비스되는 시각 신호를 사용할 수 있다 [1] 이러한 시각원들은 보통 10^{-7}~10^{-12}s 의 정확도를 가진다 본 연구에서는 GPS 시각원을 사용하였고, 이의 정밀도는 UTC 대비 ±150ns 이다

3. 시각 동기

3.1. 동기 알고리즘



[그림 5] FLL 의 블록도

시스템내 시각 장치의 동기는, [그림 5]에서 보인 바와 같이, FLL(Frequency Locked Loop)에 의해 이루어진다 FLL 은 피드백 제어 루프의 일종으로서 비교되는 두 주파수 간의 차이를 최소화시킨다 두 주파수의 오차는 저대역 필터를 거쳐 오실레이터의 주파수를 제어하기위한 신호로 입력된다. 이 알고리즘은 NTP Version 4 에서 Clock Discipline 알고리즘의 일부로 사용된 바 있다. [3][4]

$$
\begin{aligned}
e_{phase} &= t_{ext} - t_{clock} \\
e_{freq} &= f_{ref} - f \\
e_{k+1} &= e_k + e_{freq} \\
f_{k+1} &= K_{p1} e_{freq} + K_{p2} e_{phase} + K_I e_{k+1}
\end{aligned}
\qquad \text{<식 2>}
$$

매번의 갱신은 n PPS 마다 이루어지며, 식<2>에서 보인 바와 같이, k+1 번째의 갱신에서 주파수 오차 e_{freq} 가 계산된다. 이 오차는 이전의 오차와 합해져 계속 누적되며, 타이머에 설정될 주파수의 계산에 사용된다. 알고리즘의 최종 출력 f_{k+1} 은 현재 주파수 오차에 비례 상수 K_P 를 곱한 값과 누적 오차에 누적 상수 K_I 를 곱한 값의 합으로 구해진다 이 값이 타이머의 주기를 조정하기 위한 시정수 계산에 사용되는 값이다 시정수는 다음과 같이 구해진다

이러한 동기 알고리즘을 윈도우즈 시스템에 적용함에 있어 문제점들은 다음과 같다

첫째, 시정수를 설정하기 위해서는 하드웨어 포트에 직접 접근해야 한다는 것이다 윈도우즈 시스템에서 하드웨어 포트는 어플리케이션 계층으로부터 분리된다 Virtual Device Driver 에 의해 에뮬레이션 되기 때문에, 이플리케이션 계층에서 접근하는 포트는 논리적인 장치일 가능성이 높다 따라서, 실제 물리적인 장치에 직접 접근하기 위해서는 Virtual Device 계층에서 행해져야 한다

두번째 문제는 스케줄러에 의한 시정수 변경이다

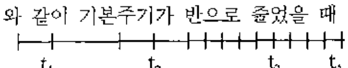
스케줄러는 다음 발생할 INT 50h 의 주기를 결정하여 타이머의 시

정수를 설정한다 이 주기는 필요에 따라 기본적으로 설정된 13.7ms의 배수로 설정되거나 그 이하로 설정된다. 이 때문에 스케쥴러에 의해 조정되는 동안 시각보정은 불가능하다. 이러한 현상을 막기위해서는 현재 설정된 시정수로 계속하여 타이머에 설정해야한다.

세번째, 다른 장치에 의해, 시스템에 재설정 가능한 기본 주기가 바뀌는 경우이다.

시스템 시작시 기본적으로 설정되는 이 주기는 다른 VxD에 의해 필요에 따라 원 주기가 변경될 수 있다 이로 인해, 보정을 위한 알고리즘 외에 주기의 변화에 따라 기본 시정수를 바꾸는 알고리즘이 필요하다

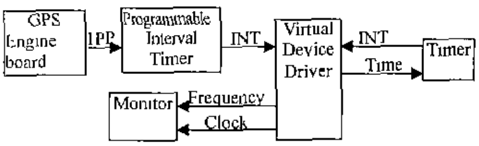
[그림 6]에서 보인 바와 같이 주기의 변화를 보인다고 할 때, 매 인터럽트 발생시 INT 50h에 삽입된 루틴은 인터럽트 주기를 측정한다 만일, t_3와 같이 기본주기가 반으로 줄었을 때 이 후의 시각 보



[그림 6] 인터럽트 주기

정은 이전 시정수의 반을 기본값으로 보정을 계속해야 한다

3.2. 실험 및 결과



[그림 7] 시스템 구성

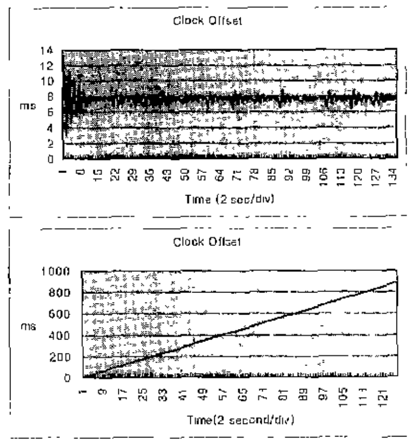
실험을 위한 시스템은 [그림 7]과 같이 구성되었다.

외부 시각원으로서 GPS Engine board 에서 출력되는 1PPS를 선택하였고, 이 것을 다시 분주하여 INT 59h의 하드웨어 인터럽트원으로 사용하였다 시각 동기 알고리즘은 이 인터럽트 핸들러 내부에 구현되었으며, INT 50h 핸들러에는 주기를 측정하기 위한 알고리즘을 구현하였다 모니터 _로그램을 통해서는 클럭옵셋과 주파수를 일정 주기 마다 폴링을 통해서 관찰하였다.

[그림 8]은 시각 동기를 하기 전과 한 후의 시각옵셋을 나타낸 것이다 아래 그래_에서는 매 초마다 7ms 이상의 시각 오차를 보이면서 시간이 지날수록 옵셋이 점점 커지는 것을 보여주고 있다 이때, 시스템에서 설정한 시정수는 16384로서, 이상적인 경우 인터럽트 주기는 16384/1193000 = 13.7334ms이 되어야한다. 실험된 결과로서 측정을 시작한 시점부터 4분후 시각옵셋이 838.6404ms으로 측정되었다 즉, 매번의 인터럽트마다 평균 47.9893μs의 오차가 발생한 것으로 나타났다. 이러한 시스템에 시각 조정을 통해 동기 알고리즘을 적용시킨 후 위 그래_와 같은 결과를 얻었다. 보정을 시작한 후 대략 20초간의 과도 현상을 거쳐 평균 8ms 옵셋을 보이고 있다. 결과적으로, 시스템은 일정한 시각옵셋을 보이면서 외부 클럭에 동기되어 누적 오차가 제거되는 것을 보여주고 있다

4. 결론

본 논문에서는 윈도우즈 시스템의 시각장치에 동기 알고리즘을 적용하기 위한 시각 조정 방안을 제시하였다 윈도우즈 시스템의 시



[그림 8] 시각옵셋 측정

각장치 모델, 특성을 제시하고 이러한 특성에 적응하기 위한 방법론을 제시하였다 제어 알고리즘으로서 NTP clock discipline 알고리즘에 적용된 바 있는 FLL을 선택하였고, 제시된 조정방안을 사용하여 적용한 알고리즘의 실험결과 일정한 시각 옵셋을 보이며, 외부 시각원에 동기됨을 제시하였다. 보다 정확한 동기를 얻기위한 제어 구조 [illegible] 로 보인다

참조문헌

[1] Elliott D Kaplan, Understanding GPS, principles and applications, Artech House Publishers 1996

[2] David L. Mills, Modeling and Analysis of Computer Network Clocks, Technical Report 92-5-2, University of Delaware, May 1992

[3] David L Mills, Clock Discipline Algorithms for the Network Time Protocol Version 4

[4] David L. Mills and Ajit Thyagarajan, Network Time Protocol Version 4 Proposed Changes, Technical Report 94-10-2, University of Delaware, October 1994

[5] Michael Lombardi, Computer Time Synchronization, Time and Frequency Division NIST

[6] Bulent Abali, Craig B. Stunkel, Clock Synchronization on a Multicomputer, IBM Research Report, March 1996

[7] Daniel L Palumbo and R Lynn Graham, Experimental Validation of Clock Synchronization Algorithms, NASA Langley Research Center June 1992

[8] Pradeep K Sinha, Distributed Operating Systems : Concepts and Design, IEEE PRESS, 1997

[9] Andrew Schulman, Unauthorized Windows 95:Developer's Resource Kit, Programmers Press, October 1994

[10] Adrian King, Inside Windows 95 한국어판, 교학사, 1995

[11] Walter Oney, Systems Programming for Windows 95, Microsoft Press, 1996

[12] Karen Hazzah, Writing Windows VxDs and Device Drivers, Programming Secrets for Virtual device Drivers, R&D Books, 1997