Pfair 멀티코어 스케줄러에서 CPU 유휴시간 기반의 인터럽트 처리 기법의 지연시간 평가

박상수
이화여자대학교 컴퓨터공학과
e-mail:sangsoo.park@ewha.ac.kr

Latency Evaluation of CPU Idle Time Based Interrupt Processing on Pfair Multi-Core Scheduler

Sangsoo Park
Dept. of Computer Science & Engineering, Ewha Womans University

요 약

다중의 명령어를 동시에 수행할 수 있는 멀티코어 시스템의 특성으로 하나의 시스템 내에서 태스크를 수행하면서 외부 이벤트의 발생에 의한 인터럽트를 동시에 처리할 수 있다. 각 태스크가 처리되어야 하는 시간에 제약성을 갖는 실시간 시스템에서는 스케줄러에 의해 CPU 코어에서의 수행이 제어되어야한다. 본 논문에서는 최적이라고 알려진 Pfair 멀티코어 스케줄러의 각 코어별 유휴시간을 정량적으로 평가함으로써 인터럽트 처리의 지연시간을 분석한다.

1. 서론

최근 자동차, 항공기 등에서 인포테인먼트, 네비게이션 등 사용자 편의에 대한 요구가 점점 높아지는 한편 연비 절감을 위한 고도의 제어 기능, 그리고 ABS (Anti-Lock Breaking System), ESC (Electronic Stability Control) 기능의 의무 탑재 등 사용자 안전에 대한 정부 규제가 강화됨에 따라 전통적인 내장형 실시간 시스템이 제어해야할 소프트웨어 컴포넌트의 양이 기하급수적으로 증가하고 있다 [1].

멀티코어 시스템은 여러개의 CPU 코어를 하나의 칩에 내장함으로써 낮은 비용에 보다 많은 소프트웨어 컴포넌트를 처리할 수 있는 장점을 갖고 있는 동시에 다중의 명령어를 동시에 수행할 수 있는 장점을 갖고 있다. 내장형 실시간 시스템에 탑재되는 소프트웨어 컴포넌트에는 컴포넌트 단독으로 수행 혹은 또다른 컴포넌트와의 정보 전달을 통해 수행하는 경우 외에도 센서를 통한 입력에 기반하여 수행되는 컴포넌트가 존재한다 [2].

이러한 센서를 통한 입력 값은 소프트웨어 컴포넌트가 센서 값을 폴링 (polling)하여 입력을 받는 경우도 있으나 폴링 주기가 적절히 설정되지 않는 경우 너무 많은 CPU 시간이 낭비 되거나 혹은 센서 값의 처리 지연시간이 너무 길어지게 되어 이에 대응되는 소프트웨어 컴포넌트의 시간 제약성을 어기게 되는 경우가 발생할 수 있다. 반면 인터럽트에 의한 센서 값의 입력 방식은 센서에 이벤트가 발생했을 때 즉시 처리될 수 있으며, 이벤트가 발생하지 않을 때는 CPU 시간을 사용하지 않는 장점이 있다 [3].

각 태스크가 처리되어야 하는 시간에 제약성을 갖는 실시간 시스템에서는 스케줄러에 의해 각 태스크 혹은 인터럽트가 CPU 코어에서 수행되는 시간이 제어되어야한다. 싱글코어 시스템에서는 많은 수의 최적화된 실시간 스케줄러가 존재하나 하나의 시스템에서 태스크의 수행 중에 한 태스크가 한 코어에서 다른 코어로 이동할 수 있는 멀티코어 시스템에는 현재까지 Pfair 스케줄러가 CPU 이용률면에서 최적이면서 구현이 가능한 유일한 방법으로 알려져 있다 [4].

내장형 시스템에 사용되는 멀티코어 프로세서는 외부의 다양한 이벤트를 인터럽트를 이용하여 효율적으로 처리하기 위한 프로그램 가능한 고수준의 인터럽트 제어기(APIC: advanced programmable interrupt controller)를 탑재하고 있다. 예를 들어 Intel의 APIC은 소프트웨어에 의해 외부 이벤트에 의해 발생된 인터럽트가 전달될 특정 코어를 선택할 수 있다 [5]. 이러한 APIC의 프로그램 가능한 인터럽트 매커니즘을 이용하여 유휴 상태인 코어로 인터럽트를 전달함으로써 시간 제약성을 갖는 태스크와 인터럽트에 의한 이벤트의 처리를 할 수 있다.

본 논문에서는 실시간 태스크와 외부 이벤트의 발생을 동시에 처리하기 위한 방법으로 인터럽트를 사용하는 멀티코어 시스템에서 Pfair 스케줄러에 의해 태스크가 CPU를 점유하는 수행시간을 제외하고 남는 CPU 유휴시간에 인터럽트를 처리하는 환경에서 인터럽트 처리의 지연시간을 정량적으로 평가해본다.

본 논문은 다음과 같이 구성된다. 처리시간의 평가를 위한 시스템 모델은 2절에서 실험 환경과 실험 결과는 3절에서 기술하고 4절은 본 논문의 결과를 내린다.

2. Pfair 스케줄러와 인터럽트 처리 모델

시스템의 각 코어는 같은 주기를 갖는 타이머에 의해 동기화가되어 있으며 태스크는 타이머의 주기에 의해 서브 태스크로 분할되어 분할된 태스크가 하나의 단위로 CPU 코어

에 할당되어 수행된다고 가정한다. 즉, M개의 코어를 갖는 Pfair 스케줄러는 [t, t+1]의 범위를 갖는 시간 유닛 단위로 준비된 서브 태스크 중에서 M개를 선택하여 각 코어에 할당하도록 하여 모든 태스크의 시간 제약성을 만족하도록 한다. 이를 위해 Pfair 스케줄러는 스케줄링이 수행 시점에 반드시 수행되는 서브 태스크 비율을 만족하도록 각 서브태스트에 동적우선 순위를 PD2 알고리즘 [6]에 기반하여 할당하여 그 중에서 상위 M개의 서브태스크를 수행한다. 이때 각 태스크의 수행 비율을 보장하기 위해서 PD2 알고리즘은 'lag'라고 정의된 비율에 해당하는 만큼 수행되야하는 서브 태스크의 수와 실제 수행된 서브 태스크의 차이를 계산하여 우선순위를 할당한다. lag가 1이면 수행되어야 하는 서브태스크가 하나 적음을 lag가 -1이면 하나 많음을 의미한다.

인터럽트의 처리를 위해서는 유휴 코어가 있으며 $|lag| \leq 1$을 만족하면, 즉, 수행 준비된 태스크의 수가 M개 미만이고 수행된 서브 태스크 수의 오차가 1 이하를 만족하면 해당 코어로 인터럽트를 전달하여 처리하도록 한다. 이때 lag의 합을 각 코어별 인터럽트 처리를 위한 우선순위로 할당하여 APIC에서 제공하는 최저 인터럽트 전달 방식 [5]를 사용한다. 이러한 인터럽트 처리 모델에서는 인터럽트가 유휴상태인 CPU 코어로 전달되도록 한다.

3. 실험 결과

본 절에서는 2절에서 기술된 Pfair 멀티코어 스케줄러에서 CPU 유휴시간에 의해 계산된 lag 기반의 인터럽트 처리 기법의 지연시간을 평가한다. 본 논문의 인터럽트 처리 방법을 기술하기 위해 T_i=(주기, 수행시간)의 예제에 대해 다음과 같은 시간 제약성을 갖는 태스크 집합을 두 개의 코어를 갖는 시스템에서 Pfair 스케줄러로 스케줄링되었을 때 각 스케줄링 시점의 태스크의 코어 할당과 lag을 <표1>에서 기술한다.

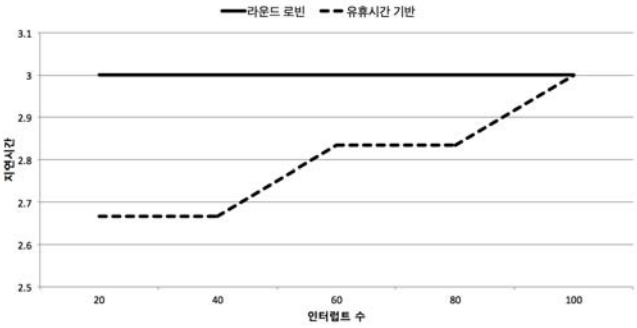
$T_0=(5,2),\ T_1=(15,3),\ T_2=(15,4),\ T_3=(30,6)$

<표1> 태스크 집합의 스케줄링과 유휴시간

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
C1	0	2	0			0	2	0			0	2	0			0	1	0			0	1	0			0	1	0		
lag	0	0	0			0	0	0			0	0	0			0	0	0			0	0	0			0	0	0		
C2	1	3				1	3				1	3				3	2				3	2				3	2			
lag	0	0				0	0				0	0				0	0				0	0				0	0			

<표1>의 각 열은 스케줄링 시간을 나타내며 C1은 코어 1을 C2는 코어 2를 의미하며 lag은 각 코어에서의 lag를 나타낸다. 스케줄링 시간 4,5,9,10,14,15,19,20,24,25,29,30에서는 수행 준비가된 서브 태스크가 없기 때문에 모든 코어가 인터런트를 처리할 수 있음을 나타내지만, 스케줄링 시간 3,8,13,18,23,28에는 코어 1은 서브 태스크를 수행하지만 코어 2는 유휴상태로 있기 때문에 코어 2의 이 시점들 역시 인터럽트의 처리가 가능하여 지연시간의 단축이 가능하다.

본 논문에서는 Pfair 스케줄러에 의한 태스크 스케줄링과 이벤트 발생에 따른 인터럽트 처리의 지연시간을 평가하기 위해 이산시간 기반의 멀티코어 시뮬레이터 환경에서 Pfair 스케줄러를 구현하고 가상의 이벤트를 발생하였을 때 인터럽트 처리 지연시간을 측정하였다. 실험에서는 APIC에서 기본으로 제공하는 라운드 로빈 방식 인터럽트 처리 방식 [5]과 본 논문의 유휴시간 기반 처리 방식과의 비교 하였다. 라운드 로빈 방식에서는 코어의 상태에 관계 없이 인터럽트를 각 코어에 순서대로 전달하며, 코어가 유휴 상태가 아닐 때는 태스크의 수행이 끝날때가지 인터럽트의 처리가 지연된다.



(그림 1) 인터럽트 처리의 최대 지연시간

(그림 1)은 임의의 이벤트를 발생하도록 하고, 처리하는 인터럽트의 수를 증가시키면서 측정한 최대 지연시간을 나타낸다. 그림에서 볼 수 있듯이 CPU 유휴시간에 의해 계산된 lag를 인터럽트 전달 코어를 선정하는데 활용함으로써 <표1>에서 C2가 유휴인 시점에 C2를 인터럽트 처리에 활용함으로써 최대 지연시간을 단축할 수 있음을 확인할 수 있다. 단, 인터럽트의 수가 어느 임계치를 넘어감에 따라 최대 지연시간이 라운드 로빈 방식에 수렴하는 것을 확인할 수 있다.

4. 결론

본 논문은 Pfair 멀티코어 스케줄러에서 lag에 기반하여 발생된 인터럽트를 전달할 코어를 선택하게 함으로써 각 코어의 유휴시간을 최대한 활용하게 하여 인터럽트가 처리되는 지연시간을 단축하였으며, 본 논문의 실험 사례에서 최대 12.5% 단축하는 것을 확인하였다.

Acknowledgement

본 논문은 한국연구재단의 기초연구사업 (과제번호: 2011-0013422)의 지원을 받아 수행된 것임.

참고문헌

[1] H. Kopetz, et. al., "Automotive software development for a multi-core system-on-a-chip," in Proc. Intl. Workshop on Software engineering for Automotive Systems, 2007.

[2] M. Ishikawa, D. McCune, G. Saikalis, "CPU model-based hardware/software co-design for real-time embedded control systems," in Proc. SAE World Congress, 2007.

[3] J. Regehr, and U. Duongsaa, "Preventing interrupt overload," in Proc. ACM LCTES, 2005.

[4] Holman and J. Anderson, "Implementing Pfairness on a symmetric multiprocessor," in IEEE RTAS, 2004.

[5] Intel, "Intel 64 and ia-32 architecture software developer's manuals"