

논문 2010-47SD-7-8

모바일 멀티코어 시스템을 위한 동적 전력관리 프레임워크

(Dynamic Power Management Framework for Mobile Multi-core System)

안영호*, 정기석*

(Youngho Ahn and Ki-Seok Chung)

요약

본 논문에서는 멀티코어 시스템을 위한 동적전력관리 프레임워크를 통하여 응용프로그램의 특성에 따라 멀티코어의 불필요한 전력소모를 줄일 수 있음을 Intel Centrino Duo를 사용한 경우와 ARM11 MPCore를 사용한 경우를 통하여 검증하였다. 프로세서의 종류에 따라 전력 소모를 줄이기 위하여 사용된 기술에 차이가 있으며 아직까지는 멀티코어 임베디드 프로세서에 동적 전압 관리와 같은 정밀한 제어가 이뤄지지 못하고 있다. 제안하는 동적전력관리 프레임워크를 이용하여 스마트폰과 같이 운영체제를 통한 멀티 프로세싱을 지원하는 환경에서는 다수의 프로세서가 소모하는 불필요한 전력을 효과적으로 줄일 수 있어야 한다. 필요한 만큼의 프로세서 성능을 결정하고 실시간으로 프로세서의 성능을 변경함으로써 각 응용프로그램의 동작을 위한 최소 요구사항을 만족시키면서 전력소모를 최소화 시킬 수 있다. 이를 위하여 본 논문에서는 응용프로그램의 실행과 종료에 따라 필요한 동작을 자동화시키고 시스템 성능을 분석하기 위한 기준을 정의하였다. 대표적인 임베디드 프로세서와 범용 프로세서에 제안하는 전력 관리 프레임워크를 적용하여 성능을 검증하였으며 본 논문이 제안한 동적전력관리 프레임워크가 응용프로그램의 최소 요구 성능을 만족시키면서 가능한 전력소모를 줄일 수 있는 인터페이스라는 것을 확인하였다.

Abstract

In this paper, we propose a dynamic power management framework for multi-core systems. We reduced the power consumption of multi-core processors such as Intel Centrino Duo and ARM11 MPCore, which have been used at the consumer electronics and personal computer market. Each processor uses a different technique to save its power usage, but there is no embedded multi-core processor which has a precise power control mechanism such as dynamic voltage scaling technique. The proposed dynamic power management framework is suitable for smart phones which have an operating system to provide multi-processing capability. Basically, our framework follows an intuitive idea that reducing the power consumption of idle cores is the most effective way to save the overall power consumption of a multi-core processor. We could minimize the energy consumption used by idle cores with application-targeted policies that reflect the characteristics of active workloads. We defined some properties of an application to analyze the performance requirement in real time and automated the management process to verify the result quickly. We tested the proposed framework with popular processors such as Intel Centrino Duo and ARM11 MPCore, and were able to find that our framework dynamically reduced the power consumption of multi-core processors and satisfied the performance requirement of each program.

Keywords: 멀티코어, 저전력, 동적전력관리, 전력관리정책

I. 서론

반도체 기술의 발달로 프로세서의 집적도 및 동작 속

도가 향상됨에 따라 컴퓨터 시스템의 성능이 지속적으로 개선되었다. 하지만 동작 속도 개선을 통한 성능 향상은 높은 전력 소모 및 발열 등의 문제로 인하여 한계에 이르게 되었다. 이를 극복하기 위한 새로운 프로세서 구조에 대한 연구가 활발하게 이루어졌으며, 최근에 멀티코어(multi-core) 및 다수코어(many-core)를 이용한 병렬 처리 기법이 대안으로 제시되었다.

본 논문에서는 8개 이하의 코어를 하나의 칩으로 설계한 것을 멀티코어로 정의하며 대표적인 예로는 Intel사의 Core2 Duo/Quad, AMD사의 Opteron과 Phenom,

* 정회원, 한양대학교 전자통신컴퓨터공학부
(Department of Electronics, Communication and Computer Engineering, Hanyang University)

※ “본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음” (NIPA-2010-C1090-1031-0009)

접수일자: 2010년5월6일, 수정완료일: 2010년7월5일

ARM사의 Cortex9, ARM11 MPCore, IBM의 CellBE, Xenon 등이 있다. ARM사의 제품들은 RISC(Reduced Instruction Set Computer) 멀티코어로 임베디드 플랫폼에서 많이 사용되며 Intel사나 AMD사의 제품들은 대표적인 CISC(Complex Instruction Set Computer) 멀티코어로 PC(Personal Computer)에 많이 사용된다. 멀티코어는 그것을 구성하는 각 코어가 동일한 경우 동종멀티코어(Homogeneous Multi-core)라 하며, 동일하지 않는 경우에는 이종멀티코어(Heterogeneous Multi-core)라고 한다. 이종멀티코어의 예로 IBM사의 CellBE 프로세서를 들 수 있다.

멀티코어의 처리능력은 그것을 사용하는 응용프로그램의 성능에 크게 영향을 주기 때문에 최근에 멀티코어의 성능을 극대화시키는 연구가^[1~5] 많이 진행되었으며 이는 다시 하드웨어적인 접근방법과^[1~2] 소프트웨어적인 접근방법으로^[3~5] 나뉜다. 초기 멀티코어에 대한 연구는 아키텍처를 제안하는 것에 초점이 맞춰져 있었지만 최근에는 Intel사나 IBM사의 멀티코어 플랫폼의 성능을 최대한 사용하기 위한 소프트웨어 중심의 연구가 주류를 이룬다. 본 논문에서는 이러한 성능중심의 연구에서 벗어나 멀티코어를 사용함으로써 발생하는 전력소모를 최대한 줄일 수 있는 방법에 대하여 연구를 진행하였다.

프로세서가 소모하는 전력을 최소화하기 위한 연구는 멀티코어가 일반화되기 이전에는 활발히 연구되어^[6~8] 왔다. 하지만 싱글코어에서 멀티코어나 다수코어로 연구 방향이 전환되었음에도 불구하고 멀티코어에 대한 저전력 이슈는 아직까지 널리 연구되지 못하고 있다. 최근까지는 다중 코어를 이용하여 고성능을 이루는 것을 목표로 하였기 때문에 저전력은 중요한 이슈가 아니었지만 멀티코어를 모바일 플랫폼에 사용하기 위해서는 저전력 문제는 반드시 해결되어야 한다. 최근 멀티코어에서 소프트웨어를 이용한 저전력 연구들은 특정 전력관리정책을^[9~10] 제안하거나 추상적인 전력관리 프레임워크를^[11~13] 제안하고 시뮬레이션 결과만으로 결론을 짓는다는 점에서 실효성을 검증할 수 없을 뿐만 아니라 2개 이상의 전력 관리 정책을 적용하는 방법에 대한 논의가 없다.

본 논문에서는 동종멀티코어의 동적 전력 소모를 줄이기 위하여 동적전력관리 프레임워크를 제안하였다. 이는 멀티코어가 소모하는 전력을 줄이기 위하여 시스템 소프트웨어 수준에서 솔루션을 제안한 것으로 기존

연구의^[14] 결과를 보완하였다는 점에서 의의가 있다. 우선 Intel Centrino Duo 멀티코어 플랫폼에서의 실험을 통해서 온도가 프로세서의 전력소모와 직접적으로 관계가 있음을 확인을 하였다. 또한 ARM11 MPCore에서 실험이 가지는 한계를 분석하였다. ARM11 MPCore는 임베디드 멀티코어의 대표적인 예로 사용되었지만 실제로 이를 이용한 전력관리가 현실적으로 거의 불가능함을 확인하였다. 마지막으로 동적전력관리 프레임워크가 프로세서를 제어하는데 있어 acpi-cpufreq CPU 드라이버를 이용함으로써 동작 주파수에 따른 전력소모를 예측할 수 있었다.

동적전력관리 프레임워크의 동작과 원리 그리고 시스템 성능을 분석을 위한 기준에 대하여 본문에서 설명하고 Intel Centrino Duo와 ARM11 MPCore에서 본 논문이 제안하는 동적전력관리 프레임워크를 동작시켜 실험함으로써 제안하는 방법의 유효성과 실제 전력절감 효과를 확인하였다.

II. 본 론

1. 동적전력관리 프레임워크의 구성

가. 동적전력관리를 위한 시스템 환경

동적전력관리는 응용프로그램이 동작하는 동안에 발생하는 전력소모를 줄이기 위한 것이다. 본 논문에서는 최근에 주목받고 있는 멀티코어 아키텍처 기반의 리눅스 환경에서 동적전력관리를 하기위한 일련의 과정을 프레임워크로 제안하였다. 제안한 동적전력관리 프레임워크가 적용된 하드웨어 플랫폼은 ARM11 MPCore 와 Intel Centrino Duo 프로세서이며 ARM사와 Intel사의 저전력 기술이 적용된 대표적인 멀티코어 프로세서이다.

ARM11 MPCore는 동일한 4개의 ARM11 임베디드 프로세서를 하나의 칩으로 구성한 멀티코어 프로세서이다. 각 ARM11 코어는 전력소모를 줄이기 위하여 WFI(Wait For Interrupt) 모드^[15] 지원한다. WFI 모드 상태에서는 해당 ARM11 코어에 들어가는 클럭이 차단되기 때문에 클럭에 의한 해당 ARM11 코어의 동적 전력 소모를 줄일 수 있다.

Intel Centrino Duo 프로세서는 Intel Core Duo 프로세서와 Mobile Intel 945 Express Chipset 그리고 Intel PRO/Wireless 3945ABG를 하나로 통합한 플랫폼으로 저전력과 고성능을 동시에 만족시키기 위한 모바일 프

로세서이다. Intel Core Duo 프로세서는 2개의 프로세싱 코어로 구성되며 각 코어는 1Mhz에서 1.83Mhz 사이의 클럭 속도로 동작가능하다. 그리고 Intel Centrino Duo 프로세서는 전력관리를 위하여 ACPI^[2] 구조를 따르고 있어 프로세서의 동작 주파수가 낮을수록 소모되는 전력은 줄어든다. 프로세서를 구성하는 각 코어에 공급되는 전원을 공유하기 때문에 프로세서의 전체 전력소모는 두 코어 중에 높은 클럭 속도로 동작하는 코어를 기준으로 프로세서의 전체 공급 전압이 결정된다. 앞서 언급한 ARM11 MPCore와 Intel Centrino Duo 프로세서는 공통적으로 L2 캐시를 공유하며 각 코어는 독립적으로 L1 캐시를 가지며 외부 메모리는 모든 코어가 공유한다.

본 논문이 제안하는 동적전력관리 프레임워크는 두 종류의 멀티코어에^[15~16] 적용되었으며 리눅스 커널을 통하여 시스템 성능에 대한 정보를 수집한다. 그리고 수집된 정보와 각 프로세서가 제공하는 저전력 기법을 이용하여 본 논문이 제안하는 동적전력관리 프레임워크가 응용프로그램이 요구하는 최소 성능을 만족시키면서 전력 소모를 가능한 줄일 수 있음을 확인하였다.

나. 동적전력관리 구조

본 논문에서 제안하는 동적전력관리 방법은 IBM과 MontaVista가 제안한 동적전력관리 구조에서 확장된 것이다. IBM과 MontaVista가 제안한 임베디드 시스템에서 프로세서 및 주변 장치에 대한 동적전력관리 기법은 임베디드 싱글코어에 저전력 기술이 적용됨에 따라 주변 장치에 대한 전력관리 또한 중요하기 때문에 시스템 관점에서 동적 전력관리가 필요함을 강조하였다.

프로세서에 대한 연구가 멀티코어로 변화됨에 따라

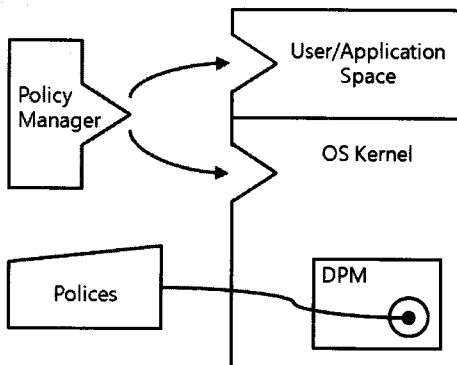


그림 1. IBM과 MontaVista의 동적전력관리 구조^[17]
 Fig. 1. A high-level view of dynamic power management of IBM and MontaVista.^[17]

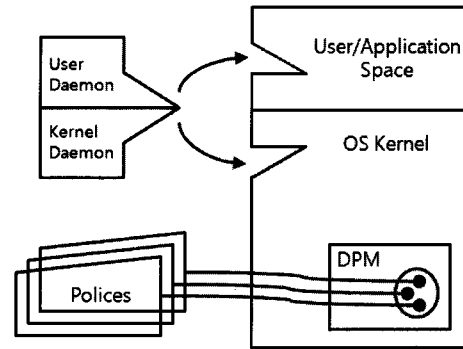


그림 2. 제안하는 멀티코어를 위한 동적전력관리 구조
 Fig. 2. A high-level view of the proposed dynamic power management for multi-core.

임베디드 프로세서로 멀티코어를 사용하고자 하는 노력이 이뤄지고 있으나 임베디드 싱글코어와 비교하여 전체적인 전력소모가 크며 임베디드 시스템 내에서 차지하는 에너지 소모 비중은 증가하였기 때문에 임베디드 시스템에 적용하기가 쉽지 않다. 그래서 본 논문에서는 [그림 2]와 같은 동적전력관리 기법을 제안하였다. IBM과 MontaVista가 제안한 구조를 더욱 세분화 하고 다수의 전력관리 정책이 동시에 적용될 수 있음을 가정하였다. 여러 전력관리정책들을 제어하기 위해서는 전력관리정책과 직접적으로 관련되는 응용프로그램을 연계하여 관리해야할 뿐 만 아니라 커널 내부에서 시스템자원을 프로파일링하고 커널에 로드된 전력관리 정책을 스케줄링해야 한다. 전력관리 정책에 대한 스케줄링은 우선순위가 높은 것을 먼저 적용하는 것을 의미하며 각 응용프로그램에 적합한 전력관리 정책이 개별적으로 존재하기 때문에 2개 이상의 응용프로그램이 동시에 수행할 때에는 시스템 성능에 민감한 응용프로그램에 맞추어 시스템 성능을 조절해야한다. 그래서 멀티프로세싱 환경에서 프로세서의 동적전력관리를 위해 사용자가 정의한 우선순위를 각 전력관리 정책에 부여하여 우선순

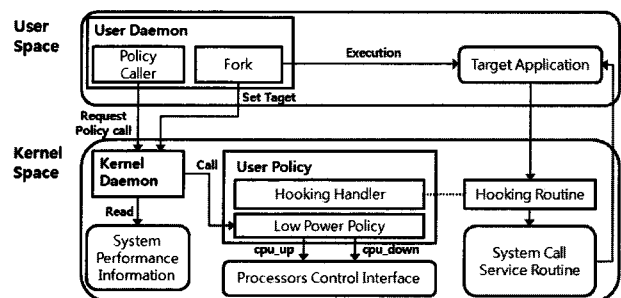


그림 3. 제안하는 동적전력관리 프레임워크
 Fig. 3. The proposed dynamic power management framework for ARM11 MPCore.

위가 높은 것을 먼저 적용해야한다.

본 논문에서는 앞서 제안한 멀티코어를 위한 동적전력관리 방법을 [그림 3]과 같이 구체화시켰으며 하나의 프레임워크로 구성하였다. 동적전력관리 프레임워크를 구성하는 요소는 크게 유저데몬, 커널데몬, 그리고 전력관리정책이다. 유저데몬은 유저영역에서 동작하는 응용프로그램으로 전력관리 대상 응용프로그램의 동작에 따라 필요한 정보를 커널데몬에 전달한다. 커널데몬은 유저데몬으로부터 받은 정보와 리눅스 커널이 수집한 시스템 정보를 이용하여 전력관리정책을 스케줄링한다.

다. 시스템 성능 프로파일링

전력관리정책 모듈은 응용프로그램이 실행되는 동안에 응용프로그램의 동작이 일정한 속도를 가지도록 시스템 성능을 결정하는 알고리즘을 정의하고 있다. 본 논문에서 제안한 동적전력관리 프레임워크는 시스템 성능을 빠르게 측정하기 위하여 프로세서 사용률, idle 코어의 수, 어플리케이션의 특징, 그리고 시스템함수 호출 패턴을 사용하였다. 리눅스 커널로부터 부팅이후 지금까지 프로세서의 사용량에 대한 정보를 얻을 수 있기 때문에 이를 이용하여 유저모드 사용률과 시스템모드 사용률을 계산할 수 있다. 유저모드 사용률은 응용프로그램의 프로세서 점유율을 의미하며 시스템모드 사용률은 커널의 프로세서 점유율을 의미한다.

Idle 코어의 개수는 저전력 상태로 동작 중인 코어의 수를 의미한다. 그리고 응용프로그램의 특징은 크게 cpu bounded, memory bounded, I/O bounded로 분류되며 순서대로 cpu, memory, I/O장치가 응용프로그램의 성능에 결정적으로 영향을 준다는 것을 의미한다. 아래 [그림4]와 같이 리눅스 커널과 유저데몬으로부터 받은 시스템 성능 데이터와 응용프로그램 특징 정보는 전력관리정책모듈이 커널데몬에 의하여 실행될 때 시스템 성능을 결정하기위한 척도로 사용된다.

응용프로그램은 프로세서가 실행이 가능한 명령어의 집합이며 명령어들은 응용프로그램 내부의 함수를 구성

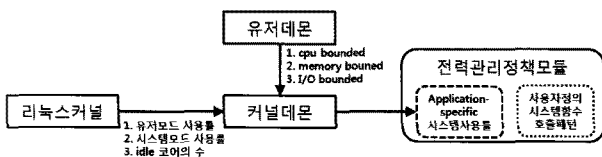


그림 4. 동적전력관리 프레임워크의 프로파일링 모델
Fig. 4. Profiling model of the proposed dynamic power management framework.

하거나 시스템함수를 구성한다. 즉, 응용프로그램을 함수 단위로 실행을 나눠서 볼 수 있다. 시스템함수가 실행되는 동안 프로세서는 커널모드로 동작하게 되며 I/O 및 메모리 할당과 해체, 그리고 타이머 및 스케줄링과 관련된 동작을 처리하게 된다. 프로세서의 동작이 유저모드에서 커널모드로 변경하는 과정에서 메모리 영역이 유저영역에서 커널 영역으로 변경되기 때문에 사용자가 정의한 함수와 비교하여 시스템함수가 가지는 오버헤드는 클 뿐 아니라 함수단위로 하나의 응용프로그램을 추적할 경우에는 셀 수 없이 많은 함수의 호출을 확인할 수 있기 때문에 그 동작을 분석하기가 어렵다. 그래서 본 논문에서는 응용프로그램의 동작에 따른 시스템 요구 성능을 분석하기 위하여 시스템함수의 호출 패턴만 분석하였다.

2. 동적전력관리 프레임워크의 동작

본 논문에서 제안한 동적전력관리 프레임워크는 유저데몬, 커널데몬, 전력관리정책으로 구성된다. 멀티코어 플랫폼에 리눅스가 포팅되었을 경우에 동적전력관리 프레임워크를 적용할 수 있으며 [그림5]과 같이 동작한다. 동적전력관리를 시작하기에 앞서 커널데몬과 유저데몬이 순차적으로 실행되며 유저데몬은 유저공간에서 동작하는 데몬 응용프로그램이다. 유저데몬은 명령어 실행 환경을 지원하며 사용자가 실행한 응용프로그램과 동일한 이름을 가진 전력관리정책을 지정된 디렉토리에서 검색하여 커널에 로드한다. 커널데몬은 커널공간에서 실행될 수 있는 코드로 전력관리의 대상 프로그램이 실행될 때 유저데몬의 요청으로 실행되며 커널에 로드된 전력관리정책을 실행한다. 응용프로그램의 실행과

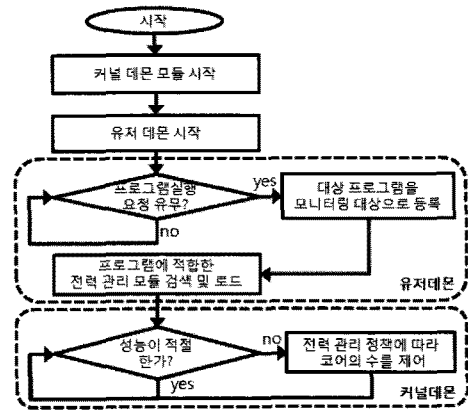


그림 5. 동적전력관리 프레임워크의 순서도
Fig. 5. Flow chart of the proposed dynamic power management framework.

종료에 따라 커널데몬이 실행할 수 있는 전력관리정책들이 변경되며 1초를 주기로 반복적으로 현재 시스템의 성능이 응용프로그램들이 요구하는 성능을 만족하는지 확인한다.

전력관리정책 모듈은 커널데몬과 마찬가지로 리눅스 커널이 제공하는 LKM(Loadable Kernel Module)를 사용하여 구현된다. LKM은 리눅스 커널에 코드를 추가하기 위한 방법으로 새롭게 커널 소스 트리를 생성하지 않고 컴파일되어있는 코드를 커널에 추가하는 방법이다. 커널코드와 독립적으로 관리할 수 있기 때문에 개발이 용이하고 실시간으로 커널에 적용가능하며 내부의 오류로 인한 시스템 오동작을 쉽게 찾을 수 있다.

전력관리정책 모듈은 초기화 부분, 종료 부분, 알고리즘 부분의 세 부분으로 나뉜다. 초기화 부분과 종료 부분에서는 전력관리정책 모듈이 커널에 로드되고 삭제됨에 따라 스케줄링에 필요한 자료구조가 안정적으로 유지될 수 있도록 데이터의 삽입과 삭제가 이뤄진다. 알고리즘 부분은 커널데몬의 전력관리정책 스케줄러가 호출하는 함수를 구성하는 것으로 커널데몬으로부터 현재 유저모드 사용자, 시스템 모드 사용자, idle 코어의 수, 그리고 어플리케이션의 특징에 대한 정보를 전달 받아 필요한 시스템 성능을 예측한다. 시스템 함수의 호출 패턴이 시스템 성능 변화와 일정한 관계를 가질 경우에는 시스템 성능을 예측하는 정보로 사용할 수 있다. 다수의 전력관리정책 모듈은 동시에 커널에 로드될 수 있으며 사용자가 정의한 우선순위에 따라 실행된다.

커널데몬이 전력관리정책 모듈을 스케줄링할 수 있

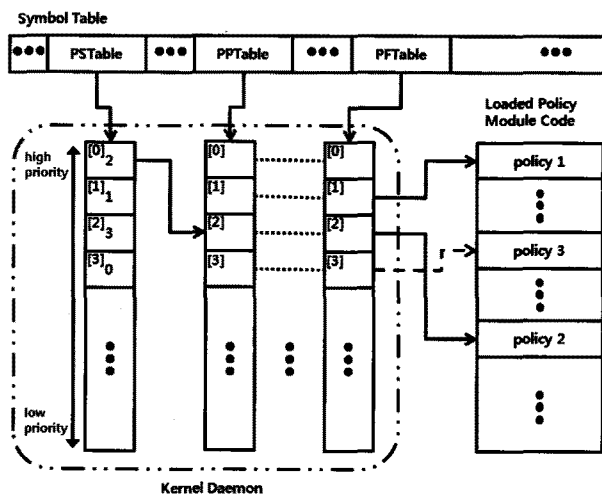


그림 6. 전력관리정책 모듈 스케줄링을 위한 자료구조
Fig. 6. Data structure for the scheduling of power management policies.

도록 하기 위하여 [그림 6]과 같은 자료구조를 사용하였다. 3개의 배열된 구성되며 각 배열은 전력관리정책 모듈의 우선순위(PPTable)와 커널데몬에 의하여 호출되는 순서(PSTable) 그리고 실제 코드가 커널 내부에서 위치 정보(PFTable)를 담고 있다. 전력관리정책의 삽입과 삭제에 따른 오버헤드는 각 전력관리정책 모듈로 분산을 시킴으로써 커널데몬이 가질 수 있는 오버헤드를 줄였다. 예를 들어 커널데몬은 PSTable이 참조하는 배열에 저장된 전력관리정책의 순서에 따라 높은 우선순위의 전력관리 정책을 먼저 실행하며 PFTable을 통하여 커널에 로드된 전력관리정책을 호출한다. 그리고 PSTable은 PPTable을 참조하여 변경되며 전력관리정책 모듈이 삽입 또는 삭제를 하는 경우에만 3개의 배열이 변경된다.

III. 실험

1. 실험환경

가. ARM11 MPCore EB

ARM11 MPCore EB에는 [표1]에 나타난 것과 같이 시스템 전압 레지스터가 FPGA 상에 설계되어 있다. 시스템 전압 레지스터를 통하여 ARM11 MPCore에 공급되는 전압과 전류를 확인할 수 있기 때문에 코어 전체의 전력소모를 수식(1)을 통하여 계산할 수 있다.

$$\begin{aligned}
 P(W) &= VI \\
 V(V) &= SYSVOLTAGE0[19:8] * (610 * 10^{-6}) \\
 I(A) &= SYSVOLTAGE2[19:8] * (610 * 10^{-6}) / (R_{sense} * 100) \quad (1) \\
 R_{sense} (\Omega) &= 0.025
 \end{aligned}$$

ARM11 MPCore가 WFI(Wait For Interrupt) 모드라는 저전력 기술을 사용하고 전력소모 값을 모니터링할 수 있는 인터페이스를 제공함에도 불구하고 ARM11 MPCore의 사용률이 100%일 경우에만 코어 수에 따른 전력 절감효과가 발생한다는 것을 [표2]에서 확인할 수

표 1. EB System FPGA Register^[17]
Table 1. EB System FPGA Register^[17].

Name	Address	Readable
SYS_VOLTAGE0	0x100000A0	VDDCORE voltage
SYS_VOLTAGE1	0x100000A4	AVDD(PLL) voltage
SYS_VOLTAGE2	0x100000A8	VDDCORE current
SYS_VOLTAGE3	0x100000AC	AVDD(PLL) current

표 2. ARM11 MPCore에서 워크로드와 활성화된 코어의 수에 따른 전력소모 변화

Table 2. Power consumption of ARM11 MPCore with changing the number of its enabled cores.

Number of workloads	Properties	Number of Cores			
		4	3	2	1
1	utilization(%)	24	33	49	99
	power(mW)	401	399	399	400
2	utilization(%)	49	66	99	99
	power(mW)	584	583	584	400
3	utilization(%)	74	99	99	99
	power(mW)	766	766	584	400
4	utilization(%)	99	99	99	99
	power(mW)	940	766	584	400

있다. 하지만 ARM11 MPCore를 이용한 실험을 통하여 임베디드 멀티코어 환경에서 본 논문이 제안한 동적전력관리 프레임워크가 멀티미디어 응용프로그램의 QoS를 만족시키면서 가능한 코어의 수를 줄일 수 있음을 확인했다는 점에서 의미가 있다.

나. Intel Centrino Duo

Intel Centrino Duo 프로세서에는 각 코어 마다 DTS(Digital Thermal Sensor)가 장착되어 있다. DTS는 프로세서의 TCC(Thermal Control Circuit) Activation Temperature에 대한 상대 값으로 코어의 온도를 측정한다. 예를 들어 DTS 값이 0일 경우 TCC값이 현재 온도가 되고 1일 경우 TCC-1값이 현재 온도가 된다. Intel Centrino Duo의 T junction(TCC)은 100이며 수식(2)에 따라 각 코어별로 측정할 수 있다.

$$Thermal(C) = TCC - IA32THERMSTATUS[22:16] \quad (2)$$

Intel Centrino Duo 프로세서에서 본 논문이 제안한 전력관리 프레임워크를 검증하기 위하여 DTS값을 사용하였으며 프로세서의 온도 변화와 동작 주파수 변화 사이의 상관관계가 있음을 실험을 통하여 [표 3]과 같

표 3. 코어별 동작 주파수에 따른 최대 온도

Table 3. Maximum thermal of each core to its frequency.

Step	Frequency (Ghz)		Active Max (Celsius)	
	Core0	Core1	Core0	Core1
0	1.00	1.00	58	56
1	1.00	1.33	63	60
2	1.33	1.33	64	62
3	1.33	1.83	70	69
4	1.83	1.83	72	70

이 확인하였다. 이는 에너지 소모 증가에 따른 발열 증가라는 일반적인 원리와 일치하는 결과이다.

[표 3]의 값을 측정하기위하여 Intel Centrino Duo 프로세서의 사용률을 100%로 고정하였으며 최대 동작 주파수에서 한 단계씩 낮추면서 측정하였다. [표3]의 값을 코어별 동작 주파수의 변화에 따른 안정된 값이다.

다. Benchmark Program 및 기타 환경

실시간 성능 변화에 대한 동적전력관리를 실험하기 위하여 FFPlay라는 오픈소스 멀티 쓰레딩 미디어 프로그램을 사용하였다. FFPlay가 재생하는 영상의 변화가 많거나 적어짐에 따라 시스템 사용률이 변화되며 일정한 프로세서 성능을 만족하지 못하는 경우 영상이 느려진다. 멀티코어에서 워크로드는 크게 병렬화된 워크로드와 여러 개의 독립된 워크로드로 나뉜다. FFPlay는 threads 옵션을 사용하여 디코딩 쓰레드의 수를 조절할 수 있기 때문에 다양한 워크로드를 실험할 수 있다.

ARM11 MPCore와 Intel Centrino Duo를 대상으로 한 실험에서 운영체제는 각각 리눅스 커널 2.6.7과 2.6.26이 사용되었으며 컴파일러는 각각 arm-gcc-3.4.2와 gcc-4.1.2가 사용되었다. 그리고 실험에 사용된 영상은 ARM11 MPCore에서는 320*184의 해상도를 가진 24fps 영상을 사용하였으며 Intel Centrino Duo에서는 1280*544의 해상도를 가진 24fps 영상을 사용하였다.

2. 실험 결과

FFPlay와 같이 QoS(Quality of Service)가 중요한 실시간 응용프로그램을 대상으로 전력관리를 수행 할 때에는 ARM11 MPCore의 WFI(Wait For Interrupt) 모드의 전력절감 효과를 기대할 수 없다. 하지만 본 논

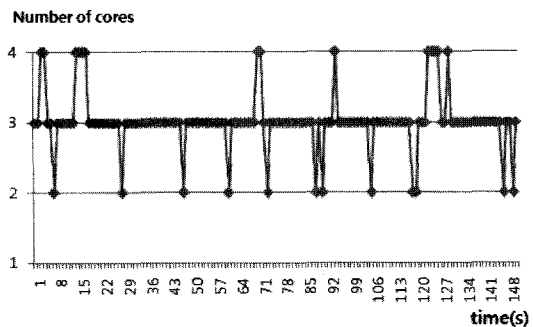


그림 7. ARM11 MPCore에서 FFPlay 실행시 활성화된 코어 수의 변화

Fig. 7. Trace of the number of cores at ARM11 MPCore while running FFPlay.

Pattern A	Pattern B	System Call Number	System Call Name
78	78	3	sys_read
4	4	4	sys_write
3	3	78	sys_gettimeofday
4	168	162	sys_nanosleep
240	3	168	sys_poll
162	4	240	sys_futex
	240		
	162		

그림 8. FFPlay의 시스템 함수 호출 패턴
 Fig. 8. Patterns of system function calls of FFPlay.

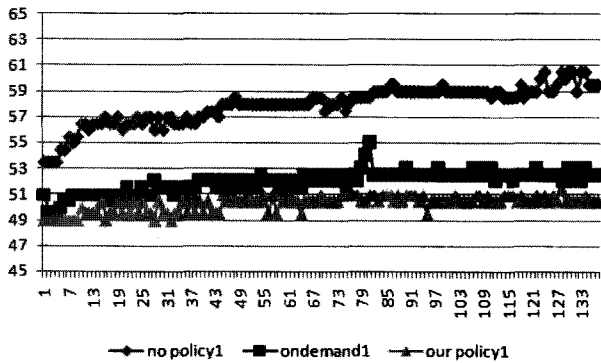


그림 9. 전력관리정책별 평균온도 변화(1 thread)
 Fig. 9. Trace of the average temperature of Intel Centrino Duo Processor while one thread is running.

문이 제안한 동적전력관리 프레임워크가 [그림 7]과 같이 실시간으로 코어의 수를 변화시킬 수 있으며 전체 재생시간에 영향을 주지 않음을 실험을 통하여 확인할 수 있다. 임베디드 시스템 내에서 사용이 가능한 전력 관리 하드웨어 기술이 프로세서에 적용된다면 본 논문이 제안한 동적전력관리 프레임워크를 적용할 수 있을 것이다.

본 논문이 제안한 동적전력관리 프레임워크의 동작을 검증하기 위하여 ARM11 MPCore이외에 Intel Centrino Duo 프로세서를 대상으로 실험을 수행하였다. 실험에 앞서 Intel Centrino Duo에서 FFPlay가 실행되는 동안 호출되는 시스템함수의 패턴을 관찰하였다. 아래 [그림 8]과 같이 FFPlay가 리눅스 커널 2.6.26에서 실행되는 동안 크게 2가지 패턴이 존재하며 이 패턴들은 반복해서 나타난다. 두 패턴에서 공통적으로 162번 시스템 함수인 sys_nanosleep이 각 패턴 동안 한번만 호출된다는 특성을 이용하여 전력관리정책을 정의할 수 있다. 시스템함수의 호출 빈도가 일정 시간동안 낮아지게 되면 FFPlay가 재생하는 영상이 느려짐을 의미하기 때문에 최소한의 QoS(Quality of Service)가 충족되는 지를 확인할 수 있다.

FFPlay가 동작하는 동안 sys_nanosleep 함수가 1초 동안 50 내외로 호출될 때 영상이 느려지지 않고 재생이 되며 40번 이하로 호출될 때 영상을 재생하는 속도는 급격히 떨어진다. 시스템 함수 호출 패턴은 응용프로그램의 동작과 직접적으로 관련되기 때문에 시스템 성능을 예측하는데 있어 시스템 사용률 보다 빠르고 정확하다. 본 논문이 제안한 전력관리 프레임워크에서 FFPlay의 동작 특성에 맞는 전력관리정책을 적용할 때 워크로드의 특성에 따라 [그림 9], [그림 10]과 같은 Intel Centrino Duo 프로세서의 온도변화를 관찰할 수 있다.

[그림 9]는 FFPlay에서 디코딩 쓰레드를 하나만 쓸 때의 결과로 전력관리정책을 쓰지 않을 때, ondemand 전력관리 정책과 cpufreq 서브시스템을 사용할 때, 그리고 본 논문에서 정의한 동적전력관리 프레임워크와 시스템 함수를 이용한 전력관리정책을 사용할 때를 비교하였다. [그림 9]에서 가로축은 초단위의 시간 변화를 나타낸 것이고 세로축은 Celsius 단위를 온도를 나타낸 것이다. 디코딩 쓰레드가 하나일 경우에는 프로세서의 평균온도가 1.6도 낮으며 가능한 낮은 동작 주파수로 프로세서가 동작하였음을 확인할 수 있으며 실험에서 사용된 Intel Centrino Duo 프로세서가 동작할 수 있는 주파수 범위인 1.83Ghz, 1.33Ghz, 1Ghz 이다. 각 동작 주파수는 ACPI P-state (Performance State)에서 P0, P1, P2에 해당하며 P-state에 따라 프로세서의 공급 전압이 낮아진다.^[2]

[그림 10]은 FFPlay의 디코딩 쓰레드를 2개 사용한 결과이다. Ondemand 및 cpufreq 서브시스템을 사용할 때와 본 논문이 제안한 동적전력관리 프레임워크와 전

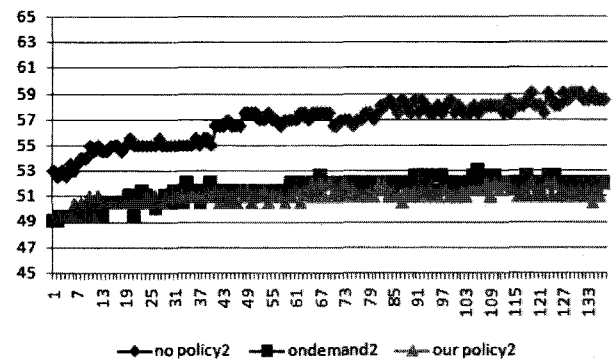


그림 10. 전력관리정책별 평균온도 변화(2 thread)
 Fig. 10. Trace of the average temperature of Intel Centrino Duo Processor while two threads are running.

력관리정책을 사용할 때의 평균온도 차이가 0.4도 밖에 되지 않는다. 이는 2개의 디코딩 쓰레드가 동작함에 따라 프로세서 사용량의 분포가 각 코어별 일정해져서 ondemand 방식이 효과적으로 적용된 결과이다. 하지만 본 논문이 제안한 동적전력관리 프레임워크와 시스템합수를 이용한 전력관리정책은 워크로드의 특성에 관계없이 프로세서의 온도를 일정하게 유지시킬 수 있다는 점에서 보다 안정되었다고 할 수 있다.

멀티코어의 온도변화를 ondemand라는 일반적인 전력관리 정책을 적용할 때와 동적전력관리 프레임워크를 통하여 응용프로그램에 최적화된 전력관리정책을 적용할 때를 비교함으로써 후자가 효과적임을 확인할 수 있다. 하지만 각 응용프로그램별 존재하는 전력관리정책에 대한 관리가 필요하다. 이러한 문제를 본 논문에서는 동적전력관리 프레임워크를 제안하여 해결하였다.

IV. 결 론

모바일 환경에서 멀티코어를 사용하기위하여 멀티코어가 소모하는 불필요한 전력을 줄여야한다. 이를 위해 본 논문은 동적전력관리 프레임워크를 통한 응용프로그램별 전력관리정책을 적용하는 것이 기존에 사용되던 Timeout이나 Utilization 등을 모든 응용프로그램에 동일하게 적용하는 것과 비교하여 안정적으로 전력소모를 줄일 수 있음을 보여주었다.

참 고 문 헌

- [1] X.-H. Sun, Y. Chen and S. Byna, "Scalable Computing in Multicore Era", PAAP, Sep. 2008.
- [2] Alon N., Efran R., Avi M., Simcha G., Rajshree C., Karthik K., Arun K., "Power and Thermal Management in the Intel Core Duo Processor", *Intel Technology Journal*, Volume 10, Issue 02, May 15, 2006.
- [3] J.A. Winter and D.H. Albonesi, "Scheduling Algorithms for Unpredictably Heterogeneous CMP Architectures", 38th International Conference on Dependable Systems and Networks, June 2008.
- [4] Hongzhong Zheng, Jiang Lin, Zhao Zhang, and Zhichun Zhu, "Memory Access Scheduling Schemes for Systems with Multi-Core Processors", ICPP'08, Portland, Oregon, September 8-12, 2008.
- [5] Svetislav Momcilovic, and Leonel Sousa, "A Parallel Algorithm for Advanced Video Motion Estimation on Multicore Architectures", MuCoCoS, 2008.
- [6] Kinshuk Govil, Edwin Chant, Hal Wasserman, "Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU", International Conference on Mobile Computing and Networking, November 1995.
- [7] Zhang Yuhua, Qian Longhua, Lv Qiang, Zhao Lei, "A dynamic frequency scaling solution to DPM in embedded linux systems", Proceedings of the 10th IEEE international conference on Information Reuse & Integration, August 2009.
- [8] Jacob R. Lorch, Alan Jay Smith, "Operating System Modifications for Task-Based Speed and Voltage", International Conference On Mobile Systems, Applications And Service, 2003.
- [9] Hrishikesh Amur, Ripal Nathuji, Mrinmoy Ghosh, Karsten Schwan, Hsien-Hsin S. Lee, "IdlePower: Application-Aware Management of Processor Idle States", MMCS, 2008.
- [10] Allan Porterfield, Rob Fowler, Mark Neyer, "MAESTRO: Dynamic Runtime Power and Concurrency Adaptation", MMCS, 2008.
- [11] Chen Tianzhou, Huang Jiangwei, Xiang Lingxiang, Shi Qingsong, "Dynamic power management framework for multi-core portable embedded system", IFMT, 2008.
- [12] Canturk Isci, Alper Buyuktosunoglu, Chen-Yong Cher, Pradip Bose, "An Analysis of Efficient Multi-Core Global Power Management Policies: Maximizing Performance for a Given Power Budget", International Symposium on Microarchitecture, 2006.
- [13] W. Lloyd Bircher, Lizy K. John, "Analysis of dynamic power management on multi-core processors", International Conference on Supercomputing, 2008.
- [14] Youngho Ahn, Young-Si Hwang, Ki-Seok Chung, "Flexible Framework for Dynamic Management of Multi-Core Systems", ISOC, November 2009.
- [15] *ARM11 MPCore Processor Technical Reference Manual*
- [16] *Intel64 and IA-32 Architectures Software Developer's Manual, Volume3A: System Programming Guide, Part1*
- [17] *Dynamic Power Management for Embedded Systems, IBM and MotaVista Software, Version1.1, November 19, 2002.*

 저 자 소 개



안 영 호(정회원)

2008년 한양대학교 컴퓨터학과
학사 졸업.

2010년 한양대학교 전자컴퓨터
통신공학과 석사 졸업.

2010년 한양대학교 전자컴퓨터
통신공학과 박사 재학중.

<주관심분야 : 저전력 멀티코어 시스템, 고성능
멀티코어 시스템, SoC>



정 기 석(정회원)

1989년 서울대학교 컴퓨터공학과
학사 졸업.

1997년 University of Illinois at
Urbana-Champaign
전산학 박사 졸업.

1998년 University of Illinois at
Urbana-Champaign
강의 전담 교수.

2000년 Synopsys, Inc. Sr. R&D Engineer.

2001년 Intel Corp. Staff Engineer.

2001년 홍익대학교 컴퓨터공학과 조교수.

2004년~현재 한양대학교 융합전자공학부
부교수.

<주관심분야 : 멀티코어, 임베디드시스템, SoC,
저전력 시스템 설계, VLSI/CAD>