시스템 수준에서의 전력관리 기법

한국전자통신연구원 손동환 · 김선자 · 김흥남

1. 머리말

시스템에서 운영체제의 두 가지 중요한 역할은 추상화의 제공과 자원의 관리이다. 운영체제는 시스템 하드웨어의 기능을 추상화하여 응용프로그램이 하드웨어에 대한 정보 없이 시스템 콜을 통하여 제어를 할 수 있도록 한다. 또한 운영체제는 프로세서, 디스크, 메모리 등의 자원을 관리한다.

전력은 시스템의 하나의 자원이지만 전원이 연결되어 있는 시스템에서는 거의 무한한 자원이라고 볼 수 있기 때문에 최근까지 운영체제에서의 전력관리에 대한 고려가 없었으나 이동 기기의 증가와 이에 비하여 배터리의 느린 발전속도로 인하여 점차 중요한 자원으로 인식되어지고 있으며 운영체제의 성능을 가름하는 주요 요인으로 자리를 잡게 되었다.

전력관리는 배터리로 동작하는 이동 기기뿐 아니라 데스크탑이나 서버에 있어서도 시스템의 발열을 감소시켜 안정성의 증대와 팬에 의한 소음 감소 등의 효과를 가져올 수 있다. 특히 클러스터나 SMP와 같이 많은 컴퓨팅 파워를 요구하는 시스템에서는 컴포넌트의 발열을 처리하기 위하여 많은 팬을 필요로 하는데 이를 효과적으로 최소화 할 수 있다.

전력관리는 기본적으로 시스템이나 기기가 사용되지 않을 때 전력 상태를 저전력 상태로 천이시킴으로써 불필요한 전력소모의 낭비를 막는 것과 시스템이나 기기의 작업의 변화를 통해 저전력 상태로 존재하는 시간을 최대화하는 것을 목적으로 한다. 하지만 전력관리는 단순히 시스템이나 기기의 전력 소모를 줄이기 위한 것이 아니다. 전력 소모의 감소를 위해서는 성능의 감소가 필수적이며 이 둘 사이의 tradeoff를 적절히 조절하는 전략을 구현하여야 한다.

본 논문에서는 운영체제를 중심으로 시스템 수준에서 사용되는 전력관리 기법들을 소개한다. 본 논문의 구성은 다음과 같다. 먼저 2장에서 전력관리의 주요한 두 가지 요소에 대하여 알아본 뒤 3장과 4장에서 두 요소별, 즉 기기 및 시스템의 전력관리를 각각 소개한다. 5장에서는 전력관리 전략 구현에 있어서의 고려사항을 살펴본다. 6장에서는 전력관리 인터페이스를 ACPI를 중심으로 알아본 다음 7장에서 결론으로 마무리를 짓는다.

2. 전력관리의 정의 및 두 가지 측면

전력관리란 시스템의 성능의 저하를 최소화하면서 소모되는 전력을 최소화하기 위하여 전력을 소모하는 기기들을 제어하고 관리하는 것을 의미하며 두 가지 측면을 가지고 있다.

2.1 상태 천이

기기는 프로세스에 의해 사용이 되는 busy 상태와 사용되지 않는 idle의 상태로 존재한다. 기기가 idle 상태인 경우 이 상태가 지속되면 전력 관리자는 기기가 당분간 사용되지 않을 것이라고 예측하고 기기의 전력 상태를 저전력 상태로 천이 시키는데 언제 천이가 이루어져야 하는지에 대한 문제가 전력관리의 첫 번째 측면이다.

만약 기기의 상태가 저전력 상태에서 busy 상태로 돌아오는데 걸리는 시간과 전력소모를 무시할 수 있다면 기기가 idle 상태일 때마다 기기를 저전력 상태로 천이 시키고 기기의 사용과 동시에 busy 상태로 되돌아오는 것이 전력소모를 가장 줄이는 방법일 것이다. 하지만 대부분의 기기들은 저전력 상태에서 busy 상태로 wakeup하는데 시간과 전력을 소모하므로 기기의 사용 빈도가 클수록 이러한 방법은 시스템의 성능 면이나 전력소모의 측면에서 비효율적일 수 있다.

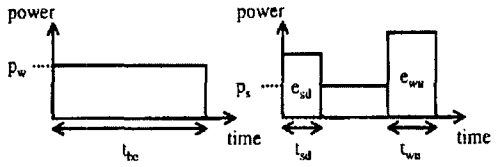


그림 1 기기의 상태 천이에 의한 전력 소모 비교

상태 천이에 의하여 전력소모를 줄이기 위해서는 상태 천이로 인한 에너지 소모보다 저전력상태로 인하여 절약된 에너지가 더 많아야 한다. 즉 idle 상태인 시간이 어떤 시간값(tbe)보다 커서 그 시간동안 저전력 상태로 존재함으로써 절약된 에너지가 상태 천이에 의한 에너지소모보다 클 경우에만 상태 천이를 해야 하는데 이 때 tbe를 break even time이라고 한다. tbe는 기기에 따라 다르다. 그림 1은 상태 천이가 일어나지 않은 경우와 상태가 천이된 경우를 비교한 것이다. 여기에서 Pw는 동작상태의 전력이고 Ps는 저전력상태의 전력이며 tsd와 twu은 각각 shut-down과 wakeup 시간을 나타낸다. 이 때 idle 시간이 tbe이라면 두 경우의 전력소모량은 같아진다.

따라서 언제 천이가 이루어져야 할 것인지에 대한 결정은 예측된 idle 시간과 tbe 중 어느 쪽이 크냐에 따라서 이루어져야 한다. 이러한 판단에 대한 전략은 크게 정적인 것과 동적인 것으로 나뉜다. 정적인 것은 현재 대부분의 운영체제에서 구현되어 있는 전략으로 일정한 시간(timeout)을 정해놓고 그 시간동안 idle한 상태로 지속되면 앞으로 idle 상태인 시간이 tbe보다 클 것으로 예측하고 저전력상태로의 천이를 수행하는 것이다. 이 전략은 구현이 간단한 장점이 있으나 timeout 동안의 전력소모로 인하여 동적인 전략에 비해 전력소모 감소의 효과는 떨어진다.

동적인 전략은 기기의 사용 패턴에 따라서 앞으로의 기기의 사용을 예측하는 것인데 크게 history에 의존하는 heuristics와 확률이론에 의한 전략으로 나뉜다. History에 기반을 둔 전략은 과거의 기기 사용에 대한 정보를 기반으로 앞으로의 idle 시간을 예측하고 이 값이 tbe보다 크면 바로 저전력 상태로 천이시키는 것이다. 한 예는 기기가 과거에 idle 상태로 오래 있었을 수록 미래에는 더 오랫동안 idle 상태로 있을 것이라고 예측하는 전략이다. 이 전략의 가장 큰 문제는 예측의 정확도가 응용프로그램에 의존적이라는 것이다. 확률적 전략은 기기의 사용을 stochastic 모델링을 통한 stochastic 최적화 문제로 전력관리를 푸는 방법이다. 이러한 stochastic 전략은 전력 감소와 성능간에 tradeoff를 할 수 있는 반면 작업에 대한 특성을 요구하고 있고 작업의 특성이 변하는 경우에 대처할 방법이 없다는 단점을 가지고 있다.

2.2 작업의 변화

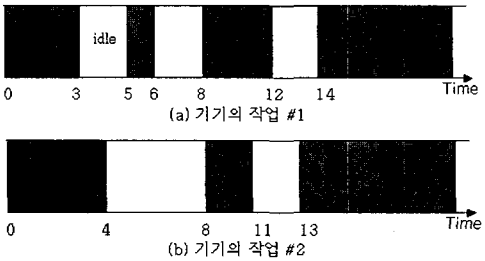


그림 2 기기 사용의 두 가지 예

전력관리에 의한 전력 소모의 감소는 기기의 idle 상태의 절대적인 시간에 비례하지 않는다. 예를 들면 그림 2의 (a) 및 (b)의 경우에 전체 idle 시간은 6으로 동일하지만 (a)의 경우에는 3번의 idle이 있는 반면에 (b)는 2번의 idle이 있고 idle 때 마다 상태의 천이가 일어난다면 wakeup에 의한 전력소모가 큰 (a)의 경우에 더 많은 에너지가 소모된다. 또한 tbe가 2보다 클 경우에는 (a)의 경우에 저전력 상태로의 천이가 한번도 발생하지 않을 수 있다. 즉 다수의 짧은 idle 상태보다는 소수의 긴 idle 시간이 전력소모 면에서 유리하다고 할 수 있다.

따라서 좋은 전력관리는 미래의 작업을 정확히 예측하여 상태의 천이를 결정하는 것 뿐만 아니라 미래의 작업을 적절히 배치시켜서 절대적인 idle 시간을 증가시키거나 또는 여러 idle 상태를 묶어서 하나의 긴 idle 상태를 구성할 수 있어야 한다. 절대적인 idle 시간을 줄이는 방법의 한 예로는 cache의 적절한 사용을 통하여 disk의 사용을 줄임으로써 disk의 idle 시간을 늘이는 것이 있다.

프로세스 작업의 순서를 바꿈으로써 idle 상태를 묶는 것은 전력관리를 운영체제에서 해야 하는 가장 큰 이유 중 하나이다. 작업 순서의 변경은 운영체제를 통해서만 가능하기 때문이다. 하지만 스케줄러 자체는 각 프로세스가 어떤 기기에 대한 사용을 요청할 것인지에 대한 정보를 알지 못한다. 따라서 운영체제는 이러한 정보를 응용프로그램으로부터 받을 수 있도록 시스템 콜을 제공해야 한다. 응용프로그램은 이

러한 시스템 콜을 통하여 자신이 언제 어떤 기기의 사용을 요청할 것인지를 운영체제에 알리고 스케줄러는 이러한 요청을 토대로 같은 기기를 사용할 프로세스를 묶어서 스케줄링함으로써 전력 상태의 천이를 최소화하고 idle 시간을 묶어 전력소모를 최소화 할 수 있다.

3. 기기별 전력관리

전력관리는 시스템 전체 및 각 기기 별로 구현이 될 수 있다. 시스템 전체의 전력관리는 기기별 전력관리로부터 이루어진다. 최근의 하드웨어들은 여러 기기들이 하나의 칩에 집적되는 SOC(System On a Chip)로 제작되는 경우가 많은데 이 경우에는 SOC 내에 전력관리를 위한 모듈을 따로 두어서 SOC 내의 각 기기별 전력상태를 제어할 수 있게 한다. 이러한 설계는 기기별 전력관리뿐만 아니라 시스템 전체의 전력관리를 쉽게 구현할 수 있는 장점이 있다.

3.1 프로세서

프로세서는 가장 빈번히 사용되며 그만큼 많은 전력을 소모하는 기기 중 하나이다. 따라서 전력관리를 통해 많은 에너지를 절약할 여지가 있으며 하드웨어적으로도 저전력을 위한 기능의 제공 및 저전력 설계를 위하여 가장 많은 노력이 이루어지고 있다. 예전에는 프로세서 제조사들이 경쟁이 주로 clock speed에서 이루어졌으나 최근에는 그 축이 저전력 지원으로 옮겨가고 있음을 볼 수 있다. 그 예로 저전력을 강점으로 하는 ARM core의 프로세서들이 영역을 넓혀가고 있으며 Intel과 AMD가 각각 SpeedStep과 PowerNow 기술을 통하여 전력관리를 위한 여러 기능을 제공하고 있고, Transmeta의 Crusoe 프로세서도 저전력을 최대 강점으로 내세우며 시장을 확장해 나가고 있다.

프로세서의 전력관리는 크게 두 가지 측면으로 나눌 수 있다. 첫번째 측면은 일반적인 기기와 마찬가지로 idle 상태일 때 sleep 또는 suspend 상태로 천이하는 것이다. 대부분의 프로세서들은 idle 상태일 때에는 즉시 idle process를 실행시킴으로써 하드웨어적으로 전력소모를 줄이도록 설계되어지므로 이 모드를 위한 전력관리는 필요하지 않다. idle process보다 좀 더 깊은 저전력 상태를 지원하는 경우에는 프로세서에 clock이 들어가지 않을 수 있다. 대부분의 운영체제는 내부 clock tick에 의해 시스템 시간을 유지하므로 이러한 저전력 상태 동안에는 시스템의 관점에서 시간이 멈춘 것으로 보일 수 있고 따라서 외부의 clock에 의하여 wakeup 후에 시간을 동기화해야 한다.

두 번째 측면은 프로세서의 clock speed 및 구동전압의 변동을 통해 프로세서가 busy 상태에서의 전력소모를 감소시키는 전략으로 동적 전압 조절(dynamic voltage scaling, DVS)이라고 한다. 이것은 프로세서에서의 전력소모가 clock speed에 비례하고 또한 구동전압의 제곱에 비례한다는 사실과 clock speed와 구동전압은 비례관계에 있다는 사실을 바탕으로 한다. 즉 프로세서의 전압을 낮추면 그 제곱에 비례하여 전력소모를 줄일 수 있지만 clock speed를 전압에 비례하여 낮춰줘야 하기 때문에 프로세서의 성능의 저하를 가져온다. 하지만 많은 작업의 경우에 항상 빠른 스피드 및 응답시간을 요구하는 것은 아니므로 여러 작업들에서 프로세서의 스피드의 적절하고 동적인 조절을 통해 사용자가 성능의 저하를 거의 못 느끼게 하면서 전력소모를 상당히 줄일 수 있다.

이 전략은 특히 작업량과 deadline에 대한 정보를 가진 real-time 응용프로그램에 있어서 큰 효과를 볼 수 있다. 예를 들면 soft real-time 응용프로그램인 MPEG 디코딩의 경우, 하나의 프레임을 디코딩할 때 작업량은 프레임의 종류와 크기에 따라 변하고 deadline은 초당 디코딩 해야 하는 프레임수(frames per second, fps)에 의해 정해진다. 이러한 정보를 바탕으로 각 프레임의 디코딩의 시작 전에 그 프레임이 정해진 시간 내에 디코딩되기 위한 최소 clock frequency와 구동전압을 설정함으로써 많은 전력소모를 줄일 수 있다.

3.2 디스크

디스크는 모터의 동작을 필요로 하는 기계적인 기기로 많은 에너지를 소모함과 동시에 wakeup을 하는데 많은 시간이 걸리는 특성을 가지고 있다. 이러한 latency로 인하여 디스크 자체에서의 에너지 소모가 큰 것 외에도 디스크 I/O를 기다리는 다른 기기들에서도 에너지 소모가 일어나기 때문에 디스크 작업의 잘못된 예측에 의한 전력관리는 시스템 전체의 성능을 저하시키고 에너지소모를 증가시킬 수 있으므로 가장 보수적인 전력관리가 요구된다.

따라서 상태 천이의 측면에서는 주로 고정된

timeout에 의한 shutdown 기법을 사용하는 것이 일반적이며, 디스크 작업을 줄이기 위하여 buffer cache의 효율적인 사용이나 pre-fetching과 같은 작업의 변화 측면에서의 고려가 더 효과적이다. 전력소모가 중요한 시스템에서는 아예 디스크를 전력소모가 적은 flash memory나 NFS로 대체하는 것이 최근의 추세이다.

일반적인 디스크가 가지는 전력모드는 주로 아래와 같다.

- active : 데이터를 읽고 쓰고 찾는 상태
- idle : spinning만 하고 있는 상태
- standby : controller만 active한 상태
- sleep : 몇몇 제어만이 active하고 host 인터페이스가 off인 상태
- off : 디스크 동작이 완전히 멈춘 상태

3.3 디스플레이 및 backlight

디스플레이 혹은 backlight 부문에서의 전형적인 전력 관리 기법은 정해진 임계시간동안 사용자의 입력이 없을 경우 디스플레이 및 backlight를 끄는 것이다. 이 때 사용자가 스크린을 보고 있을 경우에는 바로 사용자가 wakeup을 시키게 되고 이 경우에는 임계시간 값을 늘리는 전략이 효과적이다. 또한 주위 환경의 밝기 정도를 인지하여 어두울 경우에는 전원을 끌 수도 있고, 궁극적으로는 eye-tracking 기술을 이용하여 사용자의 시야가 디스플레이를 벗어나면 꺼지고 사용자의 시선을 감지하면 wakeup하는 기술의 적용도 가능하다.

4. 시스템 전체의 전력관리

시스템의 전력관리는 시스템 전체를 하나의 기기로 생각할 때 기기의 전력관리와 크게 다르지 않다. 하지만 구현상에 있어 몇 가지 고려사항이 있다.

시스템이 sleep할 때에 시스템의 각 기기들이 정해진 순서에 따라서 sleep 모드로 전환되어야 한다. 그렇지 않다면 기기 α가 sleep을 하기 위해 기기 β로부터 정보를 받아야 할 경우에 만약 기기 β가 이미 sleep 상태로 들어갔다면 다시 wakeup을 해야 하는 경우가 발생할 수 있다. 따라서 기기 β는 기기 α가 sleep한 후에 sleep하여야 한다. 이 순서는 시스템 설계시 정해진다.

시스템이 sleep 상태로 들어가기 위해서는 sleep하기 전의 상태에 대한 정보를 주로 램이나 디스크에 저장해야 한다. 램에 저장할 경우에는 램 자신은 sleep 상태로 천이하지 않고 self-refresh 모드로 동작하면서 시스템의 데이터를 유지하여야 한다. 이 경우에 sleep 상태로 들어가는 코드는 롬 영역에 있어야 한다. 만약 이 코드가 램 영역에 있다면 램이 self-refresh로 들어가는 인스트럭션을 수행함과 동시에 더 이상 램으로부터 코드를 읽을 수 없으므로 시스템이 정지하게 된다. 마찬가지로 wakeup하는 코드 역시 롬 영역에 있어야 한다.

시스템이 정보를 디스크에 저장을 하고 sleep을 하는 경우는 랜카드를 제외한 모든 기기에 전원 공급이 중단된 상태이고 wakeup은 부트로더로부터 시작된다. 따라서 wakeup을 하기 위해서는 부트로더의 수정을 필요로 한다. 즉 wakeup을 하는 것은 시스템 부팅을 하는 것과 일부 같은 절차를 필요로한다. 부트로더는 부팅 중에 power-off로부터 새로 부팅을 하는지 아니면 sleep 상태에서 wakeup을 하는지에 대하여 판단한 후 wakeup일 경우에는 미리 지정된 디스크의 위치로부터 sleep하기 전에 저장된 시스템의 정보를 읽어서 wakeup을 하게 된다.

5. 전력관리 구현시의 고려사항

아직까지는 대부분의 시스템에서의 전력관리는 사용자의 설정에 의한 timeout에 의존하는 초보 단계에 머물러 있다. 즉 노트북이나 PDA에서 사용자가 정해준 시간동안 작업이 없으면 시스템을 sleep 상태로 천이시키는 것이 전부이며 동적으로 전력관리를 하는 것은 실험단계에 머물러 있다. 그 이유는 여러 가지가 있겠지만 전력관리의 구현이 쉽지 않다는 것이 하나의 이유가 될 수 있다. 구현을 위해서는 각 플랫폼 별로 하드웨어의 전력 특성이 분석되어야 하고 사용자의 사용 패턴에 대한 분석도 있어야 한다. 특히 범용 시스템인 경우에는 사용자에게 여러 가지 선택을 제공해야 하며 표준 인터페이스를 제공하여야 한다. 그 외에 전력관리를 위하여 고려해야 할 사항은 다음과 같다.

- *시스템 내의 전력 bottleneck들에 대한 정보가 있어야 한다.*

즉 전력 소모가 큰 기기의 사용을 최대한 줄여야 한다. 예를 들어 전력소모가 적은 기기의 상태천이로 인하여 전력소모가 큰 기기의 사용시간이 늘어나는

것은 비효율적이다.

-- 배터리 시간이 아니라 수행된 작업의 양을 최대화해야 한다.

단지 느리게 동작하는 것이나 잦은 shutdown은 배터리의 시간을 길게 할지 모르나 동시에 작업이 끝나는 시간을 지연시킴으로써 원하는 전력소모의 감소를 얻지 못하는 것 뿐만 아니라 시스템의 성능을 감소시킬 수 있다. 프로세서의 구동전압을 낮추는 과정 없이 clock speed만을 낮추는 것이 한 예이다.

– 다른 기기들에의 영향을 고려해야 한다.

프로세서의 clock speed가 변화하면 이에 따라서 통신의 baud rate이나 디스플레이 refresh rate과 같은 값을 재설정해야 한다. 이런 작업은 하드웨어적으로 자동으로 이뤄지도록 설계될 수도 있다.

– 배터리의 용량은 일정하지 않다.

배터리는 사용패턴 및 프로파일에 따라서 소모량이 다르다. 같은 시간을 사용하더라도 연속적으로 사용하는 것이 간헐적으로 사용하는 것보다 많은 에너지를 소모한다. 이와 같은 배터리 특성을 고려한 전력관리가 이루어져야 한다.

6. 전력관리를 위한 OS의 인터페이스

전력관리를 위한 인터페이스는 하드웨어와 소프트웨어 양쪽에서 모두 필요하다. 운영체제는 응용프로그램이 전력 상태의 천이를 요청하거나 상태에 대한 정보를 요청하는 인터페이스를 제공하고, 더 나아가 응용프로그램이 앞으로 사용할 기기에 대한 정보를 운영체제에 전달할 수 있도록 인터페이스를 제공하여야 한다. 하드웨어는 운영체제에서 전력 상태의 천이를 하기 위한 표준화된 인터페이스를 제공하여야 한다. 이런 표준화를 통하여 플랫폼에 따른 전력관리 구현에 관련된 작업을 줄일 수 있다.

APM(Advanced Power Management)은 이러한 표준을 위한 인텔 계열의 첫 번째 스펙으로 BIOS에서의 인터페이스를 제공한다. BIOS에서 전력관리가 이루어지므로 전력관리가 각 플랫폼 별로 재구현되어야 하고 기능이 제한적이다. 이에 비해 ACPI(Advanced Configuration and Power Interface)는 인텔, 마이크로소프트, 도시바에 의해 개발된 전력관리 및 설정에 대한 개방형 업계 표준으로서 하드웨어와 운영체제의 적절한 협력으로 인해 동작되도록 설계되어 있다. ACPI는 APM에 비해 전력 상태를 세분화하였고 시스템, 기기 및 프로세서 별로 전력관리를 할 수 있도록 하였다.

그러나 인텔 구조의 복잡함과 많은 서로 다른 기기들을 지원해야 하는 관계로 ACPI의 스펙은 복잡하고 방대하며 구현을 위하여 시스템의 많은 자원을 필요로 하므로 소형 시스템에는 적합하지 않다. 또한 BIOS에의 의존성은 없어졌지만 아직도 하드웨어 벤더들이 제공하는 코드에 의존하여야 하기 때문에 시스템의 안정성을 보장할 수 없다는 단점이 있다.

비 인텔 계열의 플랫폼에서는 ACPI와 같은 표준이 아직 없으며 플랫폼별로 전력관리가 이루어지고 있다.

7. 맺음말

본 논문에서는 운영체제에서의 전력관리에 대한 개요와 기술 동향 및 구현을 위한 요구사항에 대하여 기술하였다. 전력관리는 운영체제에 있어서 선택사항이 아닌 필수사항이며 성능의 중요한 요소이다. 하지만 지금까지 전력관리에 관한 많은 연구가 이루어진 것에 비해 현재 운영체제에서 지원하는 기능은 초보적인 것이 현실이다. 이는 복잡한 전력관리의 구현과 운영을 위해서 드는 비용에 비해 전력 소모의 감소 효과가 별로 크지 않기 때문이다. 따라서 이러한 비용을 줄이는 노력이 필수적이다. 이를 위해서 운영체제와 응용프로그램간에 전력 관련 정보를 주고받을 수 있는 프레임워크의 개발과 하드웨어 및 소프트웨어의 인터페이스의 표준화가 이루어져야 할 것이다.

참고문헌

[1] C.-H. Hwang and A. C. -H. Wu, "A Predictive System Shutdown Method for Energy Saving of Event-Driven Computation," in Proc. IEEE/ACM Int. Conf. Computer Aided Design, Nov. 1997, pp. 28-32.

[2] D. Ramanthan, S. Irani, and Rajesh K. Gupta, "An Analysis of System Level Power Management Algorithms and Their Effects on Latency," IEEE Trans. on Computer-Aided Design of Integrated Circuits and Syst., vol. 21, no. 3, Mar. 2002.

[3] Q. Qiu and M. Pedram, "Dynamic Power Management Based on Continuous-time

Markov Decision Processes," in Proc. the 36th ACM/IEEE Conf. on Design Automation Conf., pp. 555-561, 1999.

[4] T. Simunic, L. Benini, P. Glynn and G. De Micheli, "Dynamic Power Management for Portable Systems," in Proc. the sixth Annual Int'l Conf. on Mobile Computing and Networking, pp. 11-19, 2000.

[5] K. Govil, E. Chan, and H. Wasserman, "Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU," Technical Report TR-95-017, ICSI Berkeley, Apr. 1995.

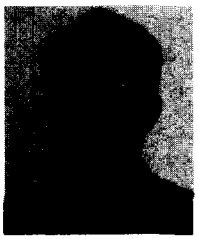
[6] T. Pering, T. Burd, and R. Brodersen, "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms," in Proc. Int'l Symp. on Low Power Electronics and Design, Aug. 1998.

[7] D. Son, C. Yu, and H. Kim, "Dynamic Voltage Scaling on MPEG Decoding," in Proc. Int'l Conf. on Parallel and Distributed Systems, June, 2001.

[8] Y.-H. Lu, L. Benini, and G. De Micheli, "Power-aware Operating Systems for Interactive Systems," IEEE Trans. on VLSI Syst. vol. 10, no. 2, Apr. 2002.

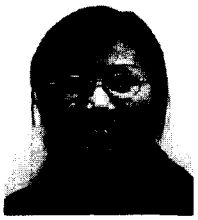
[9] ACPI. http://developer.intel.com/technology/iapc/acpi/downloads.htm

손 동 환



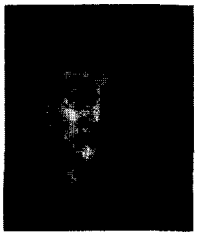
1996 고려대학교 전기공학과
1996~1999 현대정보기술
2001 한국정보통신대학원 정보공학과 (공학석사)
2001~현재 ETRI 컴퓨터·소프트웨어기술연구소 인터넷정보가전연구부 연구원
관심분야:embedded system, 전력관리, 운영체제
E-mail:dhson@etri.re.kr

김 선 자



1985 숙명여자대학교 수학과(이학사)
1995 충남대학교 대학원 컴퓨터 공학과 (공학석사)
1987~현재 ETRI 컴퓨터·소프트웨어기술연구소 인터넷정보가전연구부 선임연구원
관심분야:운영체제, DSM, 인터넷 서버
E-mail:sunjakim@etri.re.kr

김 홍 남



1980 서울대학교 전자공학과 학사
1989 미국 Ball State University 전산학 석사
1996 미국 Pennsylvania State University 전산학 박사
1983~현재 ETRI 컴퓨터·소프트웨어기술연구소 인터넷정보가전연구부 책임연구원(내장형 S/W 연구팀장)
관심분야:실시간 운영체제, 비디오 압축 알고리즘, 분산 멀티미디어 시스템
E-mail:hnkim@etri.re.kr

• IT Frontier Forum •

- 개최일자 : 2002년 10월 11일(금)
- 개최장소 : 코엑스 컨벤션센터 330호(서울 삼성동)
- 상세정보 및 참가등록 : www.kiss.or.kr
- 문 의 처 : 학회 사무국 Tel. 02-588-9246/7