특집

저전력 시스템과 저전력 소프트웨어

서울대학교 컴퓨터공학부 장 래 혁

차 례

Ⅰ. 개요

과거, 수행속도의 향상에 많은 노력을 들이던 디지털 시스템 분야에 최근 들어 전력 소모라는 매우 중요한 성능지표가 대두되었다. 적은 전력 소모는 시스템의 무게를 줄이고, 발열 문제를 완화시키므로 시스템의 동작의 안정성을 향상시키고 여러 면에서 제품 단가를 낮추는 역할을 할 뿐 아니라, 환경 오염을 줄이는 역할을 한다. 따라서 굳이 배터리를 사용하는 휴대용 기기가 아니라 하더라도, 전력 소모를 줄이는 노력은 매우 중요하다. 최근에는 여러 국가에서 휴대용이 아닌 기기의 대기상태의 전력 소모의 절감을 법적 규제로 묶는 것이 많이 시도되고 있다. 소비전력의 절감은 비교적 오래 전부터 반도체 소자를 개발하는 분야에서 매우 중요하게 다루어져 왔는데, 요즘에는 시스템 설계와 소프트웨어 설계에서도 저전력 소모의 관심이 크게 증폭되고 있다.

자동차에서 고효율의 연비를 실현하기 위해서는, 자동차 메이커에서 우선 연비가 좋은 자동차를 만들어야 하는 것이 우선적으로 이루어져야 한다. 하지만, 이미 연비를 충분히 고려하여 생산된 자동차라 하더라도, 자동차의 특성을 잘 이해하는 효율적인 운전방법의 개발과 이행이 연비에 큰 영향을 준다는 것은 널리 알려진 사실이다. 오늘날 시스템 수준이나 소프트웨어 수준에서 전력 소모를 줄이는 연구가 활성화된 이유도, 이미 충분히 전력 소모를 고려하여 생산된 소자를 이용한 시스템이라 하더라도 이를 어떻게 동작시키는 가에 따라서 전력 소모가 크게 달라지기 때문이다. 대표적인 디지털 시스템인 마이크로프로세서의 경우 매 클럭 사이클마다 소모되는 전력의 편차는 상당히 크며, 전력 소모의 편차는 해당 클럭에서 수행되는 명령어의 종류, 어드레싱 방법, 사용되는 레지스터의 번호, 레지스터에 저장된 데이터의 값, 프로그램 카운터의 변화 등에 의하여 생긴다. 마이크로프로세서를 기반으로 하는 디지털 시스템의 경우, 시스템을 구성하는 개개의 소자의 동작은 시스템 하드웨어를 구성하는 방법과 시스템을 구동하는 소프트웨어에 좌우된다. 얼마 전까지 소프트웨어의

최적화는 동작 속도, 코드의 크기 및 코드의 가독성에 주로 영향을 받았다. 하지만 요즘에는 같은 결과를 내놓는 소프트웨어라 하더라고 시스템의 전력 소모를 고려하여 작성하면 보다 적은 전력으로 동일한 결과를 얻을 수 있다는 것이 저전력 소프트웨어의 개념이다. 이와 같이 전력 소모를 줄이는 노력은, 반도체 설계, 시스템 하드웨어 설계, 컴파일러의 최적화 그리고 소프트웨어의 최적화 등 각 단계에서 모두 중요하며, 이중에서 자유도가 큰 부분에서 보다 더 큰 절감을 기대할 수 있다.

소프트웨어 수준에서 전력 소모를 절감하는 방법을 연구하려면, 소프트웨어의 변형에 따라 시스템의 전력 소모가 어떻게 변화하는가를 관찰하고, 이를 모델화하는 작업이 선행되어야 한다. 개개의 반도체 소자의 경우에는 소자의 전력 소모의 관찰과 모델링이 비교적 용이하나, 복잡도가 훨씬 큰 마이크로프로세서를 기반으로 하는 디지털 시스템의 경우에는 근본적으로 다른 방법을 취해야 한다. 일단 전력 소모의 모델링을 마치게 되면, 각종 최적화 방법이 모두 이 모델을 근거하여 설립되게 되므로, 소프트웨어 전력 소모 절감에서 전력 소모의 관찰과 모델링은 아주 중요하다.

본 글에서는 시스템 수준의 전력 소모 관찰 방법과, 모델링 방법 및 소프트웨어 최적화 방법에 대하여 소개한다. 최근 우리나라에서 강세를 보이는 휴대용 내장형 시스템에서의 전력 소모 최적화는 제품의 경쟁력을 높이는 매우 중요한 요소가 된다. 특히 소프트웨어를 사용한 전력 소모 최적화에는 생산 시에 추가적으로 제품단가를 높이지 않는다는 장점이 있다. 본 글을 통하여 시스템 설계자나 소프트웨어 설계자들이 또 다른 성능 지표인 전력 소모 절감이라는 과제를 이해하고, 이를 활용하여 보다 경쟁력 있는 연구 개발을 수행하는데 도움이 되었으면 한다.

Ⅱ. 디지털 시스템의 전력 소모

디지털 시스템을 구성하는 소자는 과거에는 TTL과 같은 Bipolar 소자를 많이 사용하였으나, 요즘에는 대규모 집적이 용이하며, 전기적으로 특성이 우수한 CMOS 소자를 널리 사용한다. 그런데, 최근 들어서는 고속으로 동작하는 시스템에서 신호의 무결성(signal integrity)을 고려하여 시스템 버스의 출력 단 등에는 Bipolar를 사용하여 CMOS 출력의 단점을 보완하는 것이 보편화 되었다. 그렇지만 디바이스 내부에서는 CMOS 소자를 사용하는 것이 일반적이므로, 보통 다음과 같이 간단한 전력 소모 모델을 적용하여도 큰 무리가 없다. CMOS로 구성된 시스템의 전력 소모는 $P=fC_LV_{DD}^2$로 계산할 수 있다. 따라서 CMOS로 구성된 디지털 시스템에서 전력 소모의 절감은 동작 주파수(f)를 낮추거나, 스위칭 캐패시턴스(C_L)를 줄이거나, 공급 전압()을 낮춤으로써 가능하다. 전력 소모의 절감 방법은 3장에서 소개한다. CMOS의 전력 소모 모델을 보면 CMOS 소자는 회로가 정적인 상태에 있는 경우에는 전력을 소모하지 않고, 신호의 천이가 있는 경우에만 전력을 소모한다는 것을 알 수 있다(동적 전력, dynamic energy).

디바이스 밖으로 나와, 시스템 수준에서 전력 소모를 보면 단순한 CMOS의 모델로 표현하기에는 적절하지 못하다는 것을 알 수 있다. 시스템 수준에서의 신호의 전달은 디바이스 내부에서 일어나는 여러 동작 중에서 극히 일부분만이 디바이스 밖으로 표출되어 발생하는 것이므로, 그 횟수나 개수에서는 상대적으로 적다. 하지만 디바이스 밖에서의 신호 구동에는 수십, 수백 배 이상의 전력이 소모되므로 실제로 차지하는 전력 소모의 양은 상당하다. 앞에서 말한 것처럼 CMOS 소자는 출력 임피이던스(impedance)가 높고, 로직 값의 0과 1의 전압차(logic swing)가 큰 점 등, 신호의 무결성 면에서 불리하다. 따라서 시스템을 구축할 때 CMOS 소자보다는 Bipolar 소자를 사용하고, 임피이던스를 정합(matching)하기 위하여 여러 형태로 신호를 터미네이션(termination)

하게 된다. 이러한 경우 Bipolar 소자와 터미네이션 저항들은 상당한 양의 정적 전력(static energy)을 소모한다. 따라서 시스템 수준의 전력 소모 모델은 정적 전력과 동적 전력을 모두 표현할 수 있어야 한다.

정적 전력 소모와 동적 전력 소모를 모두 고려한 전력 소모 모델은 다음과 같다.

$$P_T(d,f) = p_B(f) + p_{Cp}(f) + p_D(f) + p_I(d) + p_D(d)$$

$p_B(f)$: 바이어스 중첩 시 흐르는 전류에 의한 전력 소모

$p_{Cp}(f)$: 내부 기생(parasitic) 캐패시턴스에 의한 전력 소모

$p_D(f)$: 드라이브단의 동적 정류에 의한 전력 소모

$p_I(d)$: 내부의 정적 전류에 의한 전력 소모

$p_D(d)$: 드라이브단의 정적 전류에 의한 전력 소모

d : 듀티 사이클

f : 동작 주파수

그림 1은 최근에 마이크로컴퓨터에서 많이 사용하는 버스 시스템의 전력 소모를 실제로 측정하여 도시한 것이다. GTL+ 버스는 인텔사의 펜티엄 프로세서의 입출력에 사용되는 버스 규격이며, SSTL2는 DDR(Double data rate) SDRAM의 입출력에 사용되는 버스 규약이다. ABT는 5볼트 BiCMOS, LVT는 3.3볼트 BiCMOS 버스 드라이버로, 일반적인 디지털 시스템에서 널리 사용되는 대표적인 버스 규격이다. 그래프에서는 이들의 전력 소모가 동작주파수에 따라서 어떻게 변화하는가를 나타내고 있다. 동작주파수에 따라서 동적 전력 소모의 변화와, 전체 전력 소모에서 동적 전력 소모와 정적 전력 소모가 차지하는 비중을 한번에 알 수 있도록 도시하였다.

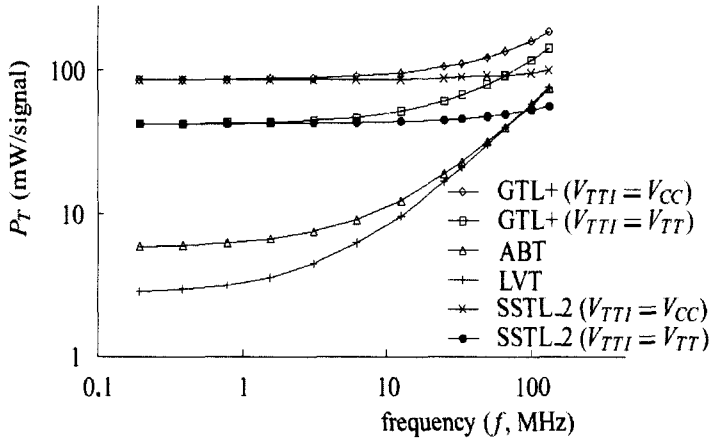


그림 1. GTLP16616, 74ABT16245, 74LVT16245, SSTL18837의 전력 소모 (d=0.5)

결과를 보면 ABT와 LVT는 정적 전력 소모가 거의 0에 가깝기 때문에 다른 디바이스에 비교해서 적은 전력을 소모하게 된다. SSTL2와 GTL/GTL+은 둘 다 많은 정적 전력 소모를 나타내지만 SSTL2는 출력의 상태에 상관없이 유사하게 정적으로 전력을 소비한다. 전체 전력 소모 PT는 정적 전력 소모 요소와 주파수에 비례해서 증가하는 동적인 요소에 의해 다음과 같은 식으로 나타낼 수 있다.

$$P_T(f) = c_q(d) + c_a f$$

다음 그림은 각 버스 규격에 따른 드라이버가 소모하는 전력 소모를 구성하는 정적 소모(C_W), 공통 정적소모(C_S), 동적 소모(C_H)를 나타낸 것이다. 간단히 살펴보면 GTL+는 0인 상태에서 소모되는 정적 소비전력이 매우 크며, 동적 소비 전력 또한 빠르게 증가하고, LVT는 정적 소비전력은 거의 무시할 수 있으며 동적 소비전력은 GTL+와 마찬가지로 빠르게 증가한다. SSTL_2는 동적 소비전력과 정적 소비전력이 적은 반면 공통 정적 소모 전력이 상대적으로 매우 크다.

표 1. 정적 전력 소모(m/Wsignal)

Power	GTL+		ABT	LVT	SSTL	
	VCC	VTTI (VTT)	VCC	VCC	VDD	VTTI (VTT)
Cq(0)	5.2	162.5 (73.9)	11.2	5.4	6.7	85.1 (38.7)
Cq(1)	5.3	0.0 (0.0)	0.5	0.0	5.6	74.6 (33.9)

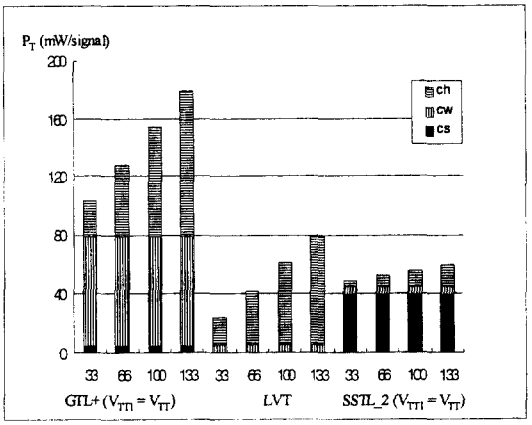


그림 2. 드라이버 별 정적 전력 소모, 공통 정적 전력 소모, 동적 전력 소모

Ⅲ. 상위 수준에서의 전력 소모 절감에 필요한 전력 소모의 관찰

이제 상위 수준 또는 소프트웨어 수준에서 전력 소모를 절감하는 방법에 대해서 소개한다. 소프트웨어 수준에서 전력 소모를 줄이기 위해서는, 우선 그 대상이 되는 반도체 소자의 전력 소모에 대한 정보를 가지고 있어야만 한다. 이러한 정보를 얻는 방법은 여러 가지가 있을 수 있겠지만, 여기서는 크게 시뮬레이션에 의한 방법과 실제 측정에 의한 방법 두 가지로 나누고 이들에 대해 설명한다.

1. 시뮬레이션을 통한 방법

관찰의 대상이 되는 소자의 전력 소모 모델이 존재하는 경우, 시뮬레이션을 통해 대상 소자의 전력 소모를 알아낼 수 있다. 시뮬레이션을 통한 전력 소모 관찰 방법은 적용되는 모델의 추상화 정도에 따라서 여러 가지 단계로 나눌 수 있다. 더 높은 수준의 추상화가 이루어진 모델을 사용할수록 시뮬레이션의 속도는 더 빨라지게 되고, 더 넓은 범위에 적용이 가능해지지만, 상대적으로 시뮬레이션의 정확도는 낮아지게 된다. 사용자는 원하는 정확도와 수행 시간 등을 고려하여, 어느 수준의 시뮬레이션을 수행할 것인지를 결정해야 한다. 현재 트랜지스터 수준에서는 Hspice, PowerMill 등이, 아키텍쳐 수준에서는 PennState의 SimplePower, Intel의 Tempest, Princeton의 Wattch, IBM의 PowerTimer 등이 시뮬레이션 도구로 많이 사용되고 있다.

2. 상용 도구를 사용한 측정에 의한 방법

전력 소모를 알아내는 다른 방법은 실제로 대상 소자의 전력을 측정하는 것이다. 전력 소모를 측정하는 데에는 여러 가지 방법이 존재한다. 일반적으로 일정한 전압에서 동작하는 대상 반도체 소자에 흐르는 전류를 측정함으로써 소모되는 전력을 알아내는 방법이 많이 사용된다. 측정을 통한 전력 소모 관찰 방법을 이용하면 시뮬레이션의 단점인 느린 속도 문제를 해결할 수 있다. 하지만, 정확하게 측정을 하기 위해서 장치를 구성할 경우 많은 어려움이 있기 때문에 새로운 장치를 측정하기 위해서는 많은 시간과 노력이 필요하다. 또 일반적인 전류 측정 방법을 사용하는 경우, 원하는 지점에서의 정확한 전력 소모 측정을 위해서는 상당히 많은 회수의 전류 측정을 필요로 하며 빠른 속도로 동작하는 마이크로프로세서와 같은 경우 이러한 방법으로는 전력 소모를 측정하기가 어렵다는 단점이 있다.

3. 실시간 사이클 단위의 전력 측정에 의한 방법

시뮬레이션의 느린 속도와 일반적인 전력 소모 측정 방법의 번거로움을 해결하기 위해서, 최근 새로

운 전력 소모 측정 방법을 제시되었다. 제시한 방법의 기본적인 동작 원리는 다음 그림에서 보는 바와 같이 2개의 캐패시터를 4개의 스위치를 이용하여 원하는 측정 주기에 맞추어 교대로 충, 방전을 시키는 구조이다. 방전된 캐패시터의 전압을 보고 대상 소자의 전력 소모를 알아낼 수 있으며, 다시 다음 주기에는 완전히 충전된 다른 캐패시터를 사용하여 측정하고, 현재의 캐패시터는 다시 충전시키는 방식으로 측정이 이루어진다.

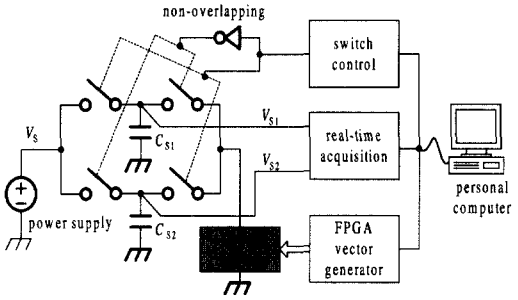


그림 3. 실시간 사이클 단위 전력측정 방법

이 측정 방법은 일반적인 전류 측정 방식과는 달리 원하는 주기의 처음과 마지막의 전압만을 측정하는 것으로 적은 회수의 측정으로도 정확한 전력 소모 계산이 가능하다는 장점이 있다.

현재 이와 같은 방법으로 마이크로프로세서, 메모리 소자 등에 대해서 측정을 수행한 연구가 소개되었다. 이 연구에서는 ARM7TDMI와 같은 마이크로프로세서의 경우 한 클럭 사이클 마다 전력 소모를 측정하여, 명령어 별 전력 소모와 데이터, 주소 값들이 전력 소모에 미치는 영향 등을 분석하였다. 또 SRAM, DRAM, SDRAM 과 같은 여러 가지 메모리 소자들의 경우는, 데이터, 주소, 또는 각종 제어 신호들에 따라서 달라지는 전력 소모 양상을 관찰, 분석하였다.

측정된 메모리의 전력 소모 경향에 대해서 간단히 살펴보면, SRAM의 경우는 대부분의 전력 소모가 주소나 데이터의 값에 따라 달라지는 동적 전력 소모로 이루어져 있으며, 일반적으로 다른 메모리 장치들에 비해 적은 전력을 소모한다. 이와는 달리 DRAM과 SDRAM은 제어 신호의 변화 시 일어나는 동적 전력 소모와, 시간에 비례하는 정적 전력 소모가 대부분을 차지하고 있어, 메모리 장치의 종류에 따라 서로 다른 전력 소모 절감 방안이 적용되어야 함을 알 수 있다.

IV. 전력 소모 절감 방법

1. 회로 수준

회로 수준에서의 전력 소모 절감 방법으로는 전압 조절 기법(voltage manipulation techniques), 전력 차단 기법(power shutdown techniques) 등이 있다.

전압 조절 기법은 반도체 소자의 전압을 조절하여 전력 소모 절감 효과를 얻는 방법이다. 반도체 소자의 전력 소모는 전압의 제곱에 비례하기 때문에 공급 전압을 낮출 수 있다면 효과적으로 전력 소모를 줄일 수 있다. 하지만 공급 전압을 낮추는 것은 반도체 소자의 신호 전달 시간을 증가시키기 때문에 성능의 저하를 초래한다. 이를 극복하기 위해서 반도체 소자 내부에 주요 회로를 중복 배치하여 병렬적으로 동작 가능하게 하거나 파이프라인 (pipeline)을 도입하여 성능 저하 문제를 해결하고 있다. 소자의 공급 전압을 낮추는 대신 다양한 임계 전압(threshold voltage)을 사용하는 기법도 시도되었다. VTCMOS(Variable Threshold CMOS) 기법은 반도체 소자를 구성하는 트랜지스터의 임계 전압을 동적으로 조절하여 전력 소모를 줄인다. 반도체 소자가 활성화 모드에 있을 경우에는 임계 전압을 낮추어 높은 속도로 동작하게 하고, 비활성화 모드에 있을 경우에는 임계 전압을 높여 속도를 포기하는 대신 누설 (leakage) 전류를 줄여 전력 소모 절감 효

과를 얻는다.

전력 차단 기법의 대표적인 예로는 클럭 차단(clock gating) 기법이 있는데 이 기법은 반도체 소자 내부의 여러 회로 중에서 유용한 계산을 수행하지 않는 부분을 동적으로 찾아내어 클럭 공급을 차단하는 방법이다. 또 선행 계산(pre-computation)이란 기법은 전력 소모 최적화 대상 회로의 결과를 미리 선행하는 소규모 회로를 추가하여 선행 계산의 결과 만으로 원래 회로가 동작하는 것을 방지함으로써 전체 회로의 상태 변화를 최소화하고자 하는 방법이다. 이를 위해 사용되는 선행 계산용 회로는 제한된 자원으로 원래 대상 회로의 결과 선행 계산의 성공률을 최대한 높이는 것을 목표로 한다. 전력 소모 절감 효과를 얻기 위해서는 예측 계산용 회로 구현 시에 속도, 크기 등의 까다로운 제약 조건을 만족시켜야 하기 때문에 선행 계산의 성공률 향상에는 어느 정도 한계가 있다. 계산 제한(guarded evaluation) 기법은 전력 소모 최적화 대상 회로의 입력 단을 제어하여 전력 소모를 방지하는 기법이다. 대상 회로의 입력단에 래치(latch)를 추가하여 해당 회로가 유용한 연산을 하는 경우에만 입력 신호를 전달해 주고 그 외의 경우에는 입력을 차단하여 대상 회로의 상태 변화를 막아 전력 소모를 줄이게 된다.

2. 시스템 하드웨어 수준

시스템 하드웨어 수준에서의 대표적인 전력 소모 절감 방법의 예로는 버스 인코딩 방법과 동적 전력 관리(dynamic power management) 방법이 있다.

먼저 버스 인코딩 방법에 대해서 알아보도록 하자. 디지털 시스템에서 버스의 캐패시턴스가 칩 내부의 캐패시턴스보다 훨씬 크다는 사실은 잘 알려져 있다. 따라서 시스템 버스의 전력 소모를 줄이기 위해서 버스의 로직 값의 변화 횟수를 줄일 수 있도록 버스를 인코딩 하는 방법이 연구되었다. 예를 들어 버스 인코딩 방법은 버스를 통해 전달되는 데이터를 인코딩하여 버스의 상태 변화를 최소화함으로써 전력 소모를 줄이고자 하는 방법이다. 버스 인코딩 방법의 대표적인 예인 버스 인버트(bus-invert) 방법은 INV라는 버스 신호를 추가하여 전송할 데이터와 이전 데이터 사이의 해밍 디스턴스(Hamming distance)가 버스 폭의 절반보다 클 경우 데이터를 0과 1을 반대로 전송하는 방법이다. 최근에는 시스템 수준의 버스에 버스 인코딩 기법을 적용할 때에, 단순한 CMOS 소자의 전력 소모 모델을 사용하지 않고, BiCMOS와 터미네이션 저항을 모두 고려하여 버스에서 소비되는 정적 전력 소모와 동적 전력 소모 를 모두 고려한 버스인코딩에 관한 연구도 발표되었다. 예를 들면, GTL+버스 규격의 경우 0의 개수를 줄이는 코딩을 하는 것이 좋은 반면 버스 홀드(bus-hold) 기능이 있는 LVT 버스 규격의 경우는 스위칭 빈도를 줄이는 것이 더 중요하다.

동적 전력 관리 방법은 컴퓨터 시스템을 구성하고 있는 각각의 장치들이 항상 의미 있는 활동을 하는 것이 아니고 비활성화 상태로 있는 경우가 상당히 많다는 사실에 착안한 방법이다. 각각의 장치들에 대한 활성화/비활성화 패턴을 예측하여 어떠한 장치의 비활성화 상태가 충분히 오래 지속될 것이라고 판단되는 경우 해당 장치가 지원하는 수면 모드(sleep mode)와 같은 소비 전력 감소 모드를 활용하여 시스템 전력 소모를 줄이는 것을 목표로 한다.

3. 소프트웨어 수준

소프트웨어 수준에서의 전력 소모 절감 방법은 하드웨어 변경이 없이 소프트웨어 최적화만으로 전력 소모 절감 효과를 얻을 수 있다는 점에서 의의가 있다. 또한 알고리즘 차원에서 문제를 다시 정리하면, 효과적으로 전력 소모를 절감할 수 있는 기회를 제공하므로 현재 상당히 활발하게 연구되고 있는 분야이다.

명령어 스케줄링 및 코드 재생성 방법은 동일한 고수준 프로그램 코드가 다양한 순서를 가진 기계어로

번역될 수 있다는 점에 기반하고 있다. 명령어의 콜드 스케줄링 (cold scheduling)이란 방법은 프로그램 코드를 번역하는 단계에서 명령어들을 재배치하여 명령어 버스의 상태 변화를 최소화하려는 시도를 보여준다. 또한 DSP 프로세서 시스템에서 명령어 압축(instruction packing), 오퍼랜드 교환(operand swapping) 등의 기법을 통해 최적화된 코드를 생성하는 방법도 제안되었다. 또한 명령어의 인코딩을 바꾸어 메모리에서 명령어를 가져오는데 필요한 스위칭 동작을 줄이는 방법과 각 명령어에서 레지스터의 번호를 바꾸어 스위칭 동작을 줄이는 방법도 제안되었다. 다른 한편으로는 어드레스 버스에서의 스위칭 활동(switching activity) 최소화를 통해 메모리 접근 비용을 줄이는 기법 등도 제시되었다.

보다 상위 수준에서 직관적이면서 효과적인 방법으로 전력 소모를 절감하는 방법 알고리즘을 개선하여 프로그램이 자원을 접근하는 횟수를 줄이는 것이다. 일예로, 메모리 접근 비용 최적화 방법은 앞에서 언급한 방법을 통하여 접근에 필요한 단위 에너지를 줄이는 방법을 쓰는 것도 좋지만 근본적으로 알고리즘을 수정하여 메모리 읽기/쓰기 연산 회수를 줄이는 것이다.

```
for (i=0; i<n; i++)
        b[i]=f(a[i]);
for (i=0; i<n; i++)
        c[i]=g(b[i]);
```

(a)

```
for (i=0; i<n; i++) {
        b[i]=f(a[i]);
        c[i]=g(b[i]);
}
```

(b)

그림 4. 메모리 접근 횟수의 최적화

중간 계산 결과인 배열 b가 CPU의 레지스터에 다 적재될 수 없을 정도로 크다고 가정하면 그림 4 (a)에서는 최대 2n번에 가까운 메모리 읽기/쓰기 연산이 일어나게 된다. 하지만 코드를 그림 4 (b)와 같이 수정하면 배열 b의 각 요소가 최종적으로 함수 g()에 의해 사용될 때까지 그대로 CPU의 레지스터에 존재할 수 있게 되어 총 n번의 메모리 읽기/쓰기 연산만이 필요하다. 위에서 예로 든 메모리 읽기/쓰기 연산 회수를 줄이는 방법은 본래 프로그램 실행 속도 향상을 위해서 계속 개발되어 온 방법이기 때문에 별다른 노력 없이도 전력 소모 절감을 위해 사용될 수 있다.

요즘 활발히 연구되고 있는 소프트웨어 수준의 전력 소모 절감 방법 중에는 CMOS 소자의 전력 소모가 공급 전압의 제곱에 비례하는 반면, 동작속도는 공급 전압에 비례한다는 관계에 근거한 것이 있다. 이 이론에 의하면 공급 전압을 낮출수록 전력 소모에서 이득을 얻을 수 있다. 공급 전압을 낮추면 디바이스의 동작 속도가 늦어지므로 주기적으로 동작하는 소프트웨어에서 주기마다 수행해야 할 계산량에 따라 전압을 조절하는 방법으로 태스크의 수행 기한(deadline)을 만족하면서 전력 소모를 줄이는 것이 가능하다. 하지만 공급 전압을 극단적으로 낮추면 정적 전력 소모가 늘어 오히려 전력 소모가 증가한다. 이와 같이 동적으로 공급 전압을 스케줄(dynamic voltage scheduling)하는 것은 대부분 CMOS 마이크로프로세서의 전력 소모 절감에는 매우 효과적이나, 공급전압의 변경이 어려운 시스템 버스와 메모리 등에서는 오히려 동적 전력 관리를 사용해서 전력 차단을 할 수 있는 기회가 줄어든다는 점에서 서로 상반되는 어려움이 있다.

V. 시스템 수준 전력 소모 분석 도구

앞장에서 설명한 것과 같이 소프트웨어 기법을 사용하여 전력 소모를 절감하려면 내장형 시스템의 에너지 소모를 측정하고 분석하는 도구가 필요하다. 이러한 도구에 대한 연구는 기존에 여러 가지 접근 방법에 의해서 시도되었다.

시스템 수준의 전력 소모 정보를 알 수 있는 소프트웨어 도구로 내장형 시스템을 구성하는 중요한 요소인 마이크로프로세서와 캐쉬 메모리, 온칩 버스와

같은 시스템-온-칩(system-on-chip)의 전력 소모에 대한 연구를 위하여 Wattch, SimplePowr와 같은 도구들이 제작 되었다.이 두 가지 도구는 전력 소모를 시뮬레이션을 통하여 추정하는 방법을 사용하며 모두 SimpleScalar라는 고성능 파이프라인 마이크로프로세서의 동작을 시뮬레이션하는 도구를 기반으로 하고 있다. 이 도구는 사이클별로 마이크로프로세서에서 현재 사용되고 있는 구성요소들, 예를 들어 레지스터나 기능별 유닛들의 에너지 소모를 계산함으로써 시스템-온-칩에서 소비되는 에너지 소모를 예측할 수 있도록 하였다. 하지만 이와 같은 시도는 실제 시스템이 아닌 시뮬레이션에서 동작하는 시스템에 기반으로 할 수 밖에 없다는 한계점이 있다. 다시 말해서, 시뮬레이션을 통하여 얻은 에너지 값은 가상 기계에서 소모되는 에너지 값이며, 이 값이 실제 전력 소모 절감 기법을 적용하려는 해당 시스템에 같은 형태로 적용이 된다고 볼 수 없다는 것이다.

하드웨어를 사용하는 도구로는 시뮬레이션과는 대조적으로 직접 측정을 통해서 에너지 소모를 분석하고자 하는 PowerScope와 같은 도구를 들 수 있다. PowerScope는 측정 대상 시스템의 전력 공급이 멀티미터를 거쳐서 이루어지도록 함으로써 측정 대상 시스템이 얼마나 많은 전류를 소비하는지를 측정하였다. 또한 Itsy라는 포켓 컴퓨터에서 자바 가상 기계가 동작할 때 얼마나 많은 에너지를 소모하는지를 측정하기 위해서 데이터 수집 장비 (DAQ)를 통한 측정을 수행하였다. 하지만 이러한 기존에 이루어진 측정을 통한 전력 소모 분석은 전력 소모를 분석하는 도구가 측정에 적합하지 않거나 정확한 결과를 얻어내기가 매우 어렵다는 단점이 있다. 예를 들어 PowerScope에서 사용한 멀티미터의 경우 측정 샘플링 주파수가 수십 KHz를 넘기가 어렵기 때문에 현재의 수백 MHz에서 동작하는 마이크로 프로세서와 같은 시스템 구성 요소들의 전력 소모를 정확하게 측정하기가 거의 불가능하다.

최근에 개발된 도구에는 SES(Seoul National University Energy Scanner)로 명명된 것이 있다. SES는 기존에 이루어진 이러한 연구들의 한계를 극복하고 보다 정확하게 에너지 소모를 측정하고 분석하는 것을 가능하게 하는 도구이다. 이 도구는 전하 이동 모델을 이용한 측정 방법을 통해서 요즘 실제로 많이 사용되고 있는 ARM7TDMI 마이크로 프로세서의 전력 소모를 측정한다. 그리고 동시에 메모리 트레이스와 제어 신호들을 프로파일하여 프로세서뿐만 아니라 메모리나 버스 시스템의 전력 소모까지 예측하는 것을 가능하게 한다. 이를 가능하게 하기 위해서 마이크로 프로세서에서 대상 프로그램이 동작할 때 사이클별로 메모리 트레이스를 프로파일하고 수집된 프로파일 데이터가 실제 측정을 통해서 검증된 메모리 에너지 모델, 버스에너지 모델 그리고 캐쉬 시뮬레이터의 입력으로 주어진다. 이는 Wattch나 SimplePower가 시스템-온-칩 요소인 마이크로프로세서 코어와 캐쉬 메모리 그리고 온-칩 버스의 전력 소모만을 고려했던 점에 비해서 SES는 실제 내장형 시스템에서 상당한 전력을 소모하는 시스템 버스, 오프-칩 메모리의 전력 소모까지 함께 고려한다는 점에서 실제 시스템의 전력 소모를 보다 정확하게 측정할 수 있게 된다.

먼저 SES의 기본이 되는 SES 하드웨어의 구조는 그림 5와 같다. SES 시스템은 PC의 PCI 버스 슬롯에 삽입되는 하드웨어(그림 5)와 PC로 구성된다. SES 하드웨어는 ARM7TDMI 마이크로 프로세서의 코어의 전력 소모를 측정한다. 이와 동시에 ARM7TDMI 마이크로 프로세서 코어의 메모리 트레이스와 각 제어 신호들을 프로파일한다. 이러한 프로파일 과정이 ARM7TDMI 마이크로 프로세서를 동작시키면서 실시간으로 수행될 수 있도록 프로파일 데이터가 고속으로 전송되어야 한다. 따라서 PC와의 인터페이스로 현재 이용할 수 있는 인터페이스 중에서 가장 속도가 우수한 PCI 로컬 버스를 사용하였다. 하드웨어적인 구현을 위해서 SES 하드웨어에는 측정 대상인 ARM7TDMI 마이크로 프로세서 코어

와 전하 이동 모델을 이용하는 전력 소모 측정회로 그리고 ARM7TDMI 마이크로 프로세서의 각 제어 신호들을 프로파일하기 위한 회로로 구성되어 있다. 실시간으로 이러한 동작이 가능하도록 하기 위해서 자일링스(Xilinx)사의 고성능 FPGA인 Spartan II 와 고성능 메모리인 SDRAM을 사용하였다.

그리고 SES 하드웨어가 PCI 로컬 버스를 통해서 PC와 연동되므로 SES 하드웨어가 리눅스 운영 체제 위에서 동작할 수 있도록 디바이스 드라이버를 작성하였다. 그리고 작성된 디바이스 드라이버를 이용

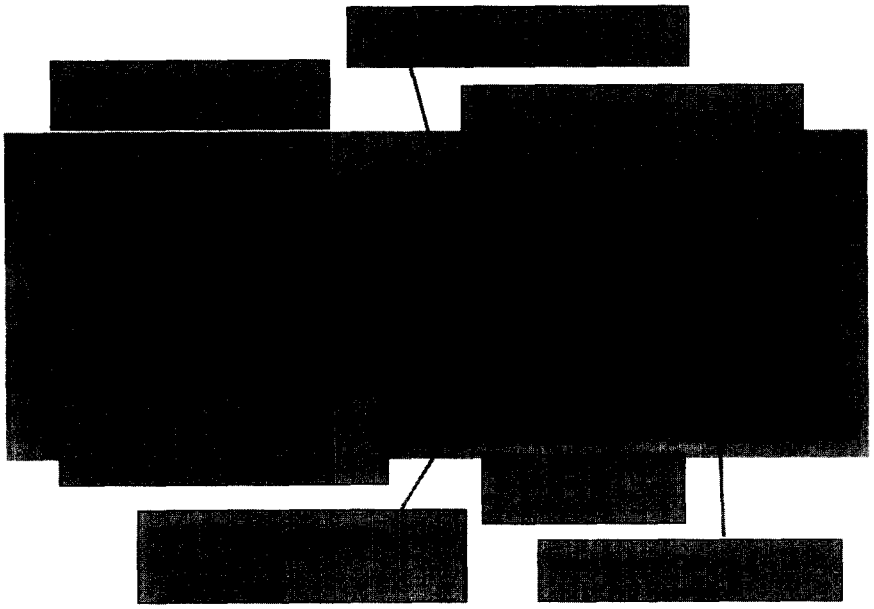


그림 5. SES의 하드웨어 구조

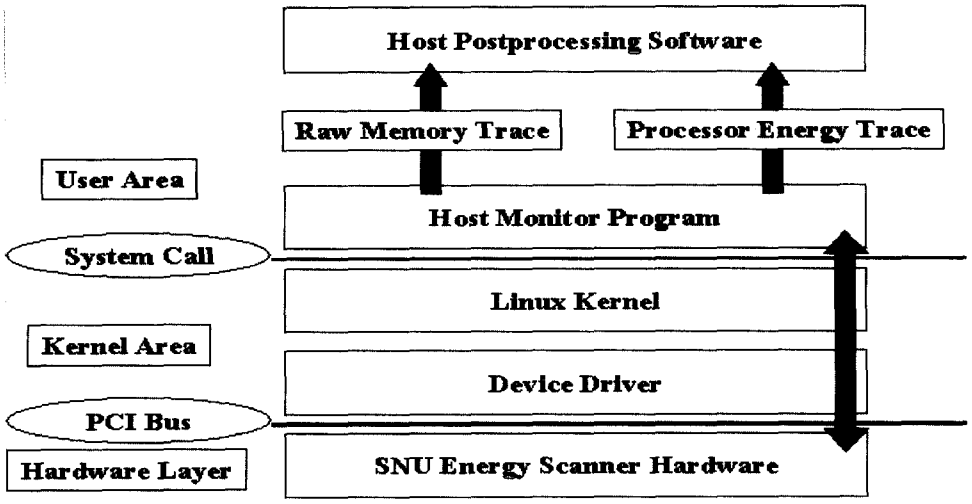


그림 6. SES의 소프트웨어 구조

해서 SES 하드웨어에 있는 ARM7TDMI 마이크로 프로세서를 구동하고 측정 대상 소프트웨어를 다운로드하고 그에 해당하는 프로파일 데이터를 실시간으로 가져올 수 있는 모니터 소프트웨어를 작성하였다. 이러한 모니터 소프트웨어를 이용해서 우리는 ARM7TDMI 마이크로 프로세서에서 동작하는 측정 대상 소프트웨어가 발생하는 에너지 트레이스와 메모리 참조 트레이스를 빠르고 효율적으로 얻어낼 수 있다.

위에서 설명한 SES 하드웨어와 모니터 소프트웨어를 사용하면 ARM7TDMI 마이크로 프로세서에서 측정 대상 소프트웨어를 구동할 때에 마이크로 프로세서 코어가 소비하는 에너지에 대한 트레이스와 측정 대상 소프트웨어가 발생시키는 메모리 참조 트레이스를 얻을 수 있게 된다. 이를 바탕으로 마이크로 프로세서 코어뿐만 아니라 메모리 시스템의 전력 소모까지 예측하기 위해서 후처리 과정(Postprocessing)이 수행되며 이는 아래의 그림과 같이 수행된다. 먼저 메모리 참조 트레이스를 캐쉬 시뮬레이터를 통해서 측정 대상 소프트웨어에 의해서 발생하는 메모리 참조를 시뮬레이션한다. 이는 발생하는 메모리 참조들이 메모리 시스템과 프로세서 코어의 동작과 전력 소모에 어떠한 영향을 줄 것인가를 결정하기 위해서 수행된다. 예를 들어 캐쉬 미스를 발생시킨 메모리 참조의 경우 마이크로 프로세서 코어가 그 사이클에 대해서 파이프라인이 스톨(stall) 되어야 한다. 따라서 이미 측정된 프로세서 코어에서 소비되는 에너지가 이를 반영하도록 재계산이 되어야 한다. 캐쉬나 메모리 그리고 버스 시스템과 같은 메모리 시스템에 대해서도 마찬가지로 이러한 정보들을 바탕으로 에너지 모델을 통해서 소모되는 에너지를 계산하게 된다. 마지막으로 위에서 계산된 각각의 시스템 구성 요소들에서 소모되는 에너지들을 모두 더함으로써 실제 내장형 시스템에서 소모되는 전력을 정확하게 예상할 수 있다. 여기서 사용하는 버스나 메모리의 에너지 모델은 실제 측정과 실험을 통해서

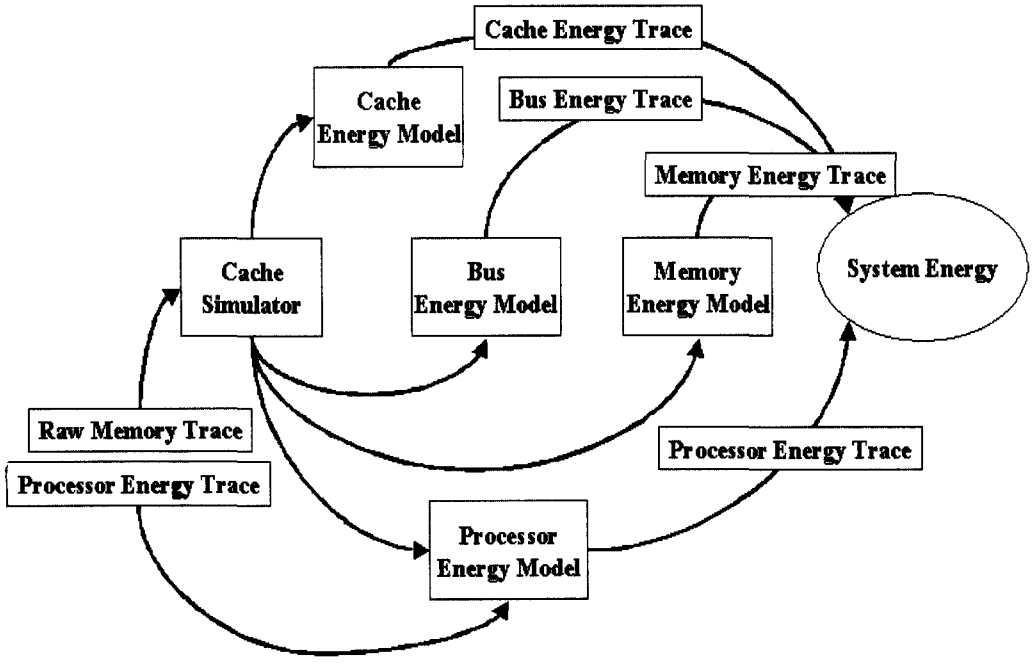


그림 7. SES의 시스템 수준 전력 소모 분석의 구조

얻어진 모델을 사용하기 때문에 매우 정확하게 에너지 소모를 예측할 수 있다. 캐쉬의 에너지 모델은 캐쉬 자체의 규칙적인 구조로 인해서 많은 관련 연구들이 이미 수행되어 있으며 이를 참조해서 정확한 에너지 모델을 수립할 수 있다.

SES는 일반적인 내장형 시스템을 이루는 가장 기본적인 요소인 마이크로프로세서와 캐쉬, 메모리 그리고 버스 시스템에 이르기까지 전체적인 전력 소모와 각 시스템 구성 요소들의 전력 소모를 분석할 수 있다.

Ⅵ. 결론

디지털 시스템의 소비전력의 절감은 매우 중요한 최적화 문제로 자리잡았다. 여기서는 디지털 시스템의 전력 소모와 전력 소모 모델을 기초로 회로 소자 수준부터 소프트웨어 수준까지의 전력 소모 절감의 원리 및 방법에 대한 소개를 하였다. 그리고, 소프트웨어 수준에서 전력 소모 최적화를 하기 위한 소프트웨어 및 하드웨어 도구를 간단히 살펴보았다. 본 글이 최근 디지털 시스템에서 큰 변수로 떠오른 전력 소모 절감이라는 최적화 과제의 이해와 활용에 기초가 되었으면 한다.

※참고문헌

[1] Naehyuck Chang, Kwanho Kim, and Hyun Gyu Lee, "Cycle-Accurate Energy Consumption Measurement and Analysis : Case Study of ARM7TDMI," accepted for publication in *IEEE Transactions on VLSI Systems*, 2001.

[2] Naehyuck Chang, Kwan-Ho Kim, and Heonshik Shin, "Dual-mode bus encoding for high-performance bus drivers," *in proceedings of the IEEE Region 10 Conference*, September 1999, pp. 876-879

[3] Pradip Bose, Margaret Martonosi, and David Brooks "Modeling and Analyzing CPU Power and Performance: Metrics, Methods and Abstractions", *SIGMETRICS 2001/ Performance 2001 Tutorial.*

[4] Naehyuck Chang and Kwan-Ho Kim, "Real-time per-cycle energy consumption measurement of digital systems," *IEE Electronics Letters*, vol. 36, no. 13, pp. 1169-1170, June 2000.

[5] Naehyuck Chang, Kwanho Kim, and Hyun Gyu Lee, "Cycle-Accurate Energy Consumption Measurement and Analysis : Case Study of ARM7TDMI," in *Proceedings of ACM/IEEE International Symposium on Low Power Electronics and Design (ISPLED00)*, pp. 185 - 190, July, 2000.

[6] Naehyuck Chang, Kwanho Kim, Heonshik Shin, and Jinsung Cho, "Bus Encoding for Low-Power High-Performance Memory Systems," in *Proceedings of 37th ACM/IEEE Design Automation Conference (DAC2000)*, pp. 800 - 805, June 5 - 9, 2000, Los Angeles, USA

[7] Sheayun Lee, Andreas Ermedahl, Sang Lyul Min, and Naehyuck Chang, "An Accurate Instruction-Level Energy Consumption Model for Embedded RISC Processors," to appear *in* Proceedings of ACM SIGPLAN 1999 Workshop on Languages, Compilers and Tools for Embedded Systems, 2001

[8] Kuroda, et al., "A high-speed, low-power 0.3um CMOS gate array with variable threshold voltage (VT) scheme", *Proc. CICC*, May 1996, pp. 53-56.

[9] M. Alidina, J. Monteiro, S. Devadas, A. Ghosh, and M. Papaefthymiou, "Precomputation-based sequential logic optimization for low power," *IEEE Trans. VLSI Syst.*, vol. 2, no. 4, pp. 426-436, 1994.

[10] J. Monteiro, J. Rinderknecht, S. Devadas, and A. Ghosh, "Optimization of combinational and sequential circuits for low power using precom-putation," *in Proc. 1995 Chapel Hill Conf. Advanced Research in VLSI, Chapel Hill*, NC, Mar. 1995, pp. 430-444.

[11] V. Tiwari, S. Malik, and P. Ashar, "Guarded evaluation: Pushing power management to logic synthesis/design," *in Proc. ISLPD-95 : ACM/IEEE Int. Symp. Low Power Design*, Dana Point, CA, Apr. 1995, pp. 221-226.

[12] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," *IEEE Trans. VLSI* Syst., vol. 3, no. 1, pp. 49-58, 1995.

[13] C.-L. Su, C.-Y. Tsui, and A. M. Despain, "Low power architecture design and compilation techniques for high-performance processors," *in Proc. IEEE CompCon '94*, Feb. 1994, pp. 489-498.

[14] M. T.-C. Lee, V. Tiwari, S. Malik, and M. Fujita, "Power analysis and minimization techniques for embedded DSP software," *IEEE Trans. VLSI Syst.*, vol. 5, no. 1, pp. 123-135, 1997.

[15] V. Tiwari, S. Malik, A. Wolfe, and M. T.-C. Lee, "Instruction level power analysis and optimization of software," *J. VLSI Signal Process.*, pp. 1-18, 1996.

[16] S. Wuytack, F. Catthoor, L. Nachtergaele, and H. De Man, "Global communication and memory optimizing transformations for low power design," *in Proc. IWLPD-94: ACM/IEEE Int. Workshop on Low Power Design*, Napa Valley, CA, Apr. 1994, pp. 203-208.

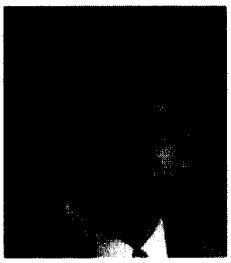
[17] Ye, W.; Vijaykrishnan, N.; Kandemir, M.; Irwin, M.J. "The design and use of SimplePower: a cycle- accurate energy estimation tool," *in Proceeding of 37th ACM / IEEE Design Automation Conference*, pp. 340 -345, 2000.

[18] Brooks, D.; Tiwari, V.; Martonosi, M., "Wattch: a framework for architectural-level power analysis and optimizations," *in Proceedings of the 27th International Symposium on Computer Architecture*, pp. 83 -94, 2000.

[19] J.~Flinn and M.~Satyanarayanan, "Powerscope: a Tool for Profiling the Energy Usage of Mobile Applications," *in Proceedings of IEEE Workshop on Mobile Computing Systems and Applicaions*, pp. 2-10, 1999.

[20] K.I.~Farkas, J.~Flinn, G.~Back, D.~Grunwald, and J.M.~Anderson, "Quantifying the Energy Consumption of a Pocket Computer and a Java

Virtual Machine," *in Proceedings of SIGMETRICS*, pp. 252-263, 2000.



장 래 혁

1985년-1989년 서울대학교 제어계측공학과(학사), 1989년-1992년 서울대학교 제어계측공학과(석사), 1992년-1996년 서울대학교 제어계측공학과(박사), 1996년-1997년 서울대학교 제어계측신기술 연구센터(연수연구원), 1997년 University of Michigan (Research Fellow)