DVFS에 기반한 안드로이드 스마트폰의 CPU 소모 전력 절감 기법의 한계

유종훈[1], 허승주[2], 홍성수[1),2),3)]
[1)]서울대학교 공과대학 전기·컴퓨터공학부
[2)]서울대학교 융학과학기술대학원 지능형융합시스템학과
[3)]차세대융합기술연구원 스마트시스템연구소
(jhyoo, sjhuh, sshong)@redwood.snu.ac.kr

1. 서론

2009년 스마트폰이 국내에 처음 도입된 지 불과 2년(11년 10월 기준) 만에 2000만대가 보급되어, 스마트폰 사용자의 비율이 휴대폰 사용자의 38%를 넘어섰다. 이런 경향은 더욱 심화될 것이며, 2015년에는 그 보급률이 90%를 돌파할 것으로 전망된다. 이와 같은 스마트폰 대중화로 경제, 사회 전반에 걸쳐 스마트 비즈니스, 스마트 라이프, 스마트 워크, 스마트 정부 등 이른바 '스마트 혁명'이 진행되고 있으며, 스마트폰은 일상생활에서 뗄 수 없는 필수 기기가 되었다.

이러한 스마트폰에서도 사용성의 측면에서 아직 개선되어야 할 점들이 여러 가지 존재한다. 많은 조사에 따르면, 스마트폰 사용자들이 가장 불편하게 느끼는 점은 '배터리의 빠른 소모'이다. 휴대폰이 다용도로 사용되어 그 사용 시간이 증가함에 따라, 배터리 용량의 부족을 호소하는 것이다. 이런 이유로 스마트폰 제조회사에서는 스마트폰 동작시간을 마케팅 포인트로 적극 활용하고 있다. 아이폰4S의 경우, 아이폰4 보다 연속대기시간이 100시간이나 줄어들어 큰 논란이 일어나고 있다. 그 만큼, 저전력 문제는 소비자 입장에서 민감한 이슈라 할 수 있다.

스마트폰 동작시간을 늘리기 위해 다양한 측면에서 저전력 기법이 연구되고 있다. 이들은 크게 하드웨어 기법과 소프트웨어 기법으로 나뉜다. 일반적으로 하드웨어 분야에서 신기술은 개발기간이 길고, 현재 판매되는 제품에 즉시 적용하기 어려우며, 개발에도 많은 비용이 소요된다. 반면 소프트웨어 분야의 신기술은 개발 즉시 적용이 용이하며, 개발에도 상대적으로 적은 비용이 소요된다. 따라서 소프트웨어에 기반한 저전력 기술은 매우 매력적인 방안이다.

소프트웨어에 기반한 저전력 기술로 가장 대표적인 것이 DVFS(dynamic voltage frequency scaling) 메커니즘과 이를 활용하여 실제로 CPU의 동작주

파수를 조정하는 저전력 정책이다. 이 기법은 CPU의 응용부하가 높을 때는 최고 동작주파수로 CPU를 구동하고, 응용부하가 낮을 때에는 응용의 응답시간에 대한 희생 없이 낮은 동작주파수로 CPU를 구동한다. 실제로 많은 AP(application processor) 제조사들은 칩의 동작주파수를 소프트웨어적으로 조작할 수 있는 인터페이스를 BSP를 통해 제공하고 있다. 스마트폰에서 점유율이 가장 높은 운영체제인 Linux는, CPU 동작주파수를 선택하는 정책을 관장하고 BSP의 DVFS 인터페이스를 이용하여 CPU의 동작주파수를 변경하는 모듈인 governor를 제공한다. 이에 더 나아가 사용자 차원에서 다양한 동작상황에 특화된 사용자 수준의 governor를 구성할 수 있도록 한다.

이에 따라 Linux governor를 활용하여 CPU의 소모전력을 감소시키려는 연구가 학계와 산업계에서 활발하게 진행되어 왔다. 그러나 많은 개발자와 연구자들의 기대와는 다르게, 실제로는 DVFS에 기반한 Linux governor의 효과는 매우 제한적인 것으로 평가된다. 그 이유는 다음의 세 가지 사실에 기인한다. (1) 스마트폰 사용환경에서 CPU가 소모하는 전력의 양은 전체 소모량의 12-13% 정도에 불과하다. 따라서 CPU 소모전력을 10%까지 감소시킬 수 있다고 하더라도, 전체 시스템 소모전력의 1.2% 정도 만 줄일 수 있게 된다. 이는 스마트폰을 1시간 사용한다고 할 때, 배터리 수명을 고작 1분 정도 연장시키는데 불과하다. (2) Linux governor는 CPU의 응용부하가 큰 경우 그 효과가 매우 제한적이다. CPU 응용부하가 낮더라도 싱글코어 시스템에서 커널 스케쥴러의 개입여지가 낮아 governor의 효과가 매우 작다. 한편 멀티코어 시스템에서는 load balancing을 시도할 수 있으나, cache affinity로 인해 load balancing이 제한을 받기 때문에 governor의 효과가 작을 것이다. (3) DVFS의 최적 활용이 매우 어렵다. 앱의 메모리 사용 패턴에 따라서 동작주파수와 전력소모량의 linear한 관계가 깨질 수 있다. 이는 동작주파수가 작아져서 앱의 수행 시간이 길어지게 되면, 메모리가 active 상태로 대기하는 시간이 함께 길어져서 대기전력이 커질 수 있기 때문이다. 따라서 동작주파수가 어느 값 이하로 작아지면 오히려 전력소모량이 증가하는 경우가 발생한다. 그러나 그런 임계값은 CPU 제조공정, CPU의 구조, 그리고 수행되는 응용부하 등에 복합적으로 영향을 받는다. 따라서 최적동작주파수를 찾기 위해서는 상당한 overhead가 수반되는 연산을 스마트폰 사용 중에 수행하여야 한다.

본고에서는 Android의 기본 운영체제인 Linux의 governor에 대해서 살펴본다. 특히 Linux governor의 한계에 대해서 기술적 배경분석을 통해 구체적으로 살펴본다. 본 연구를 통해서 필자는 Linux governor를 활용하여 소

모전력 감소시키려는 노력에 대해 부정적인 견해를 밝힌다. 그 대신 (1) 스마트폰이 idle한 기간 동안에 DVFS 기술을 적용하는 장치들을 다양화시키는 방법, (2) LTE-TCP cross layer optimization, (3) WiFi MAC 계층을 최적화시키는 방법, (4) 동적 발열관리를 통한 소모저력 절감하는 기법에 대한 연구개발 방향을 제시한다. 본고의 나머지 장은 다음과 같이 구성된다. 먼저 2장에서는 Linux에서 저전력을 달성하기 위해 제공되는 기법들의 배경지식에 대해 알아본다. 3장에서는 저전력 기법들 중 최근 주목을 받고 있는 DVFS와 Linux governor의 한계점을 알아보고, 4장에서는 이런 DVFS의 한계를 극복하기 위한 연구방향을 제시한다. 끝으로 5장에서 본고의 결론을 맺는다.

2. Linux의 저전력 기법

학계와 산업계에서는 저전력 시스템을 구현할 수 있도록 하기 위해, 하드웨어 기반한 기법들을 개발하려는 노력을 경주해 왔다. 하지만 사용자의 요구에 부합될 만큼 기술개발의 속도가 빠르지 않았다. 그에 따라 추가로 소비전력을 줄이기 위한 소프트웨어 기술들이 함께 활발히 연구되어 왔다. 연구 초기에는 BIOS를 통해 전원관리를 수행하는 기법인 APM(Advanced Power Management)이 주로 연구되었다. 점차 운영체제에 의해 전원관리를 하는 필요성이 증대되었고, 이에 따라 ACPI(Advanced Configuration & Power Interface)가 발표되었다. Linux에서도 저전력 시스템에 대한 요구가 반영되어, 소비전력을 제어하기 위한 산업표준인 ACPI 가 도입되었다. 이 장에서는 Linux의 저전력 기법인 ACPI와 DVFS의 원리에 대해 살펴보도록 한다.

2.1 ACPI

Linux는 ACPI를 지원함으로써 저전력 시스템 설계를 지원한다. ACPI는 1996년 공개된 최초의 컴퓨터용 전력관리 산업표준이다. ACPI는 먼저 컴퓨터의 각 장치들의 전원 상태를 정의하도록 한다. 이에 더 나아가 운영체제가 상황에 맞게 장치들의 전원상태를 바꿀 수 있는 인터페이스를 제공한다. 시스템에서 운영체제는 이 인터페이스를 이용하여 절전 기능을 구현할 수 있다[1].

표 1 ACPI의 power states: (a) global power states, (b) CPU power states [1]

State	SW runs	Power consumption
G0	Yes	Large
G1	No	Smaller
G2	No	Very near 0
G3	No	RTC battery (mechanicall off)

(a)

State	Description
C0	Operating state (executing instructions)
C1	Halt (not executing instructions)
C2	Stop-clock (clock is stopped)
C3	Sleep (cache coherency disabled)

(b)

표 1 (a)는 ACPI가 정의하고 있는 global power state를 보여준다. Global power state란 컴퓨터 시스템 전체의 전원 상태를 의미한다[1]. G0 state부터 G3 state로 갈수록 시스템의 상태가 달라져 소비되는 전류가 더 작아진다. ACPI에서는 전체 global state 이외에도 각각의 장치들을 위한 별도의 power state가 제공된다. ACPI는 특히 프로세서를 위해서 표 1 (b)에 나타난 것과 같이 C0~C3 state를 제공하는데, C0에서 C3로 갈수록 소비전력이 작아진다. 하지만 이런 power state 들은 시스템이나 시스템에 연결된 장치, 혹은 그 장치들의 특정 부분의 전원을 off 시킴으로써 소비전력을 줄이는 방법을 제공하기 위한 것이다. 따라서 이 장치들이 동작하는 동안에 소비전력을 줄일 수 있는 추가적이고 고도화된 방법은 제공되지 않는다.

2.2 DVFS의 원리

위에서 설명된 것처럼, 장치들이 동작하는 동안에는 소비전력을 줄일 수 없는 한계를 극복하기 위해 나온 기술이 바로 DVFS(dynamic voltage frequency scaling)이다. 컴퓨터에서 수행되는 작업들 모두가 최고의 CPU의 성능을 요구하는 것은 아니다. 또한 사용자의 입장에서 더 빠른 작업속도 보다는 더 긴 배터리 지속시간을 원하는 경우도 있다. 이러한 인식에 기초하여, 부품의 동작속도를 줄이는 대신 소비전력에서 이득을 취하고자 하는 기술이 DVFS이다. DVFS 기술은 특히 CPU에서 많이 사용되는데, 이는 성능에 영향을 미치는 다른 부품대비, CPU의 소비전력이 월등히 높기 때문이다.

Linux에서는 DVFS 기술을 CPU를 통해서만 구현하고 있다. 즉 Linux의 DVFS는 CPU의 clock frequency를 줄여 소비전력을 줄이는 기술로서 존재한다. 표 1 (b)의 C0 state는 P0~Pn (n<16) state로 더 세부적으로 나뉘는데, 이는 ACPI 내에서 DVFS 기술을 지원하기 위함이다. P0에서 Pn으로 갈수록

clock frequency가 낮아지고 소비전력은 줄게 된다[1].

CPU clock frequency를 낮추면 소비전력이 감소하는 원리는 다음과 같다. CPU를 구성하는 반도체 소자인 CMOS가 소비하는 에너지는 다음과 같이 구성된다[2, 3, 5].

$$E = E_{static} + E_{dynamic} + E_{short}$$

E_{static}은 CMOS 회로의 bias 전류와 누설 전류에 의해 소비되는 에너지이고, $E_{dynamic}$은 회로의 capacitor에 전하가 충·방전됨에 따라 소비되는 에너지이다. E_{short}은 CMOS의 PMOS와 NMOS가 동시에 on 될 경우 소비되는 에너지를 말한다. 고전적인 CPU 구조에서는 $E_{dynamic}$이 다른 것들에 비해 우세하므로, 위의 수식을 다음과 같이 근사화할 수 있다.

$$E \cong E_{dynamic} = C \bullet f \bullet V^{2} \bullet {}^{t}(C\text{는상수})$$

여기서 f는 주파수이고, V는 소자에 공급되는 전압이며, t는 동작된 시간이다. 여기서 $V \propto f$, $t \propto \frac{1}{f}$ 이므로 CPU에서 소모되는 에너지와 전압 간에는 다음과 같은 관계식이 성립한다.

$$CPU\text{에서소비되는에너지} E \propto f^{2}$$

즉 CPU의 clock frequency를 조금만 줄이더라도 많은 에너지를 줄일 수 있게 되는 것이다. 그림 1에 기술된 StrongARM processor의 경우를 보면, clock frequency를 200MHz에서 161MHz로 20% 줄일 때, 소비전력은 44%, 에너지로 환산하면 30% 절감 효과가 발생한다는 것을 알 수 있다. 이는 이론적 계산치인 36% 에너지 절감효과가 실제로도 유효함을 증명하는 것이다[4].

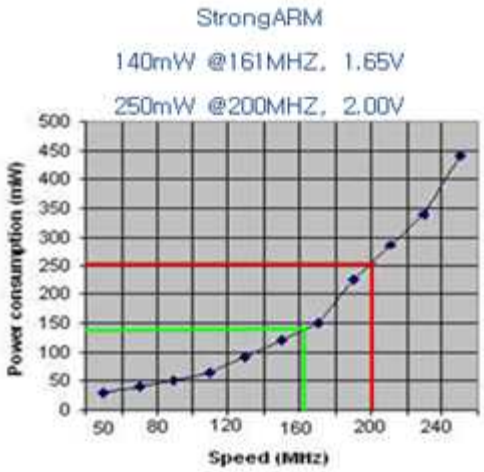


그림 1 StrongARM processor에서 CPU clock frequency에 따른 소비전력 [4]

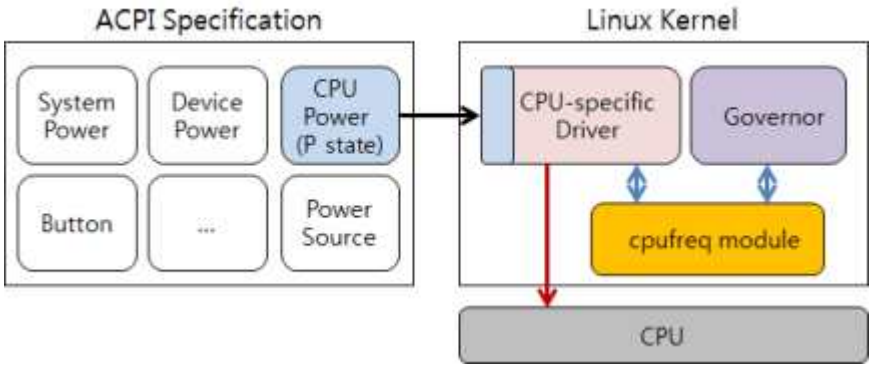


그림 2 Linux DVFS system

2.3 DVFS의 원리

Linux는 DVFS를 크게 세 부분으로 나누어 구현한다[1, 5, 6]. 이 세 부분은 각각, (1) CPU의 frequency를 직접적으로 변경하는 CPU specific driver 부분, (2) CPU의 frequency를 언제, 얼마나 바꿀지를 결정하는 governor 부분, 그리고 (3) driver와 governor가 각각 독립적으로 개발될 수 있도록 지원하는 공통 인터페이스인 CPUfreq module 부분이다. 그림 2의 오른쪽 Linux kernel에서는 이 세 가지 부분의 상호관계를 그림으로 보여준다. CPU specific driver는 ACPI specification에서 DVFS를 관장하는 CPU power state에 따라 구현된다. Governor는 CPUfreq module을 통해 CPU frequency를 간접적으로 바꾸는 역할을 하며, 어떤 상황에서 얼만큼 변경할지에 대한 정책을 가지고 있다. Linux는 다양한 종류의 governor를 구현하여 사용자 요구에 따라 바꿔 쓰도록 지원한다. Linux에서 일반적으로 쓰이는 다섯 개의 governor는 다음과 같다.

- Performance

  Performance governor는 CPU가 가능한 최고의 clock frequency를 항상 사용하도록 설정하는 governor이다. 따라서 소비전력 절약을 제공되지 않는다. 보통 데스크톱 시스템과 같이 전원에 직접 연결된 곳에서 사용되며, CPU가 idle 상태로 거의 가지 않는 경우 적합하다.

- Powersave

  Powersave governor는 CPU가 가능한 가장 낮은 clock frequency를 사용하도록 설정하는 governor이다. 이는 Performance governor와 반대되는 개념으로 동작한다. 이 governor는 전력 소모를 최고로 줄여 주지만, CPU의 성능은 가장 낮아진다.

- Ondemand

  Ondemand governor는 일반적인 DVFS 개념을 적용한 대표적인 governor이다. CPU 사용률이 상한 임계 치를 넘으면 최대 clock frequency를 사용하고, 하한 임계 치보다 낮아지면 clock frequency를 단계적으로 낮춘다. 시스템은 외부의 간섭 없이 CPU 부하에 따라 자동으로 clock frequency를 변경한다.

- Conservative

  Conservative governor는 Ondemand governor를 약간 변형한 것으로 CPU 사용률이 상한 임계 치를 넘으면 단계적으로 clock frequency를 높인다는 점에서 상이하다.

- Userspace

  위의 네 개의 governor는 운영체제 스스로 CPU의 clock frequency를 변경하는 것인데 반해, Userspace governor는 user space의 프로그램(또는 root로 실행중인 프로세스)에 의해 동작되는 것이다. 이는 사용자 혹은 프로그램이 임의로 CPU clock frequency 변경 정책을 만들 수 있도록 지원하기 위함이다.

3. 실사용 환경에서 Linux DVFS 기법의 한계점

앞 절에서 기술한 Linux governor를 활용하여 CPU의 소모전력을 감소시

키려는 연구가 학계와 산업계에서 활발하게 진행되어 왔다. 그러나 많은 개발자와 연구자들의 기대와는 다르게, 실제로는 DVFS에 기반한 Linux governor의 효과는 매우 제한적인 것으로 평가된다. 이 장에서는 Linux governor가 왜 실사용 환경에서는 기대와 다른 결과를 보이는지 세 가지 측면에서 살펴본다.

3.1 CPU 전력 소모량의 한계

실사용 환경에서 DVFS 기술이 가지고 있는 가장 큰 한계는 스마트폰이 소비하는 총 에너지 중 CPU로 줄일 수 있는 에너지의 비율이 작다는데 기인한다. 이것은 스마트폰에서 차지하는 CPU 소비전력 비중과 governor를 통해 줄일 수 있는 소비전력 비율 확인을 통해서 알 수 있다. 그림 3는 스마트폰에서 차지하는 CPU 소비전력 비중을 나타내다[7]. 이는 20명의 스마트폰 사용자를 대상으로 6개월 동안 사용 패턴을 조사하여 얻은 결과이다. 스마트폰 시스템 전체에서 CPU의 전력 소비율은 display, network에 이어 3위를 차지한다. 하지만 이를 전체적으로 비교해 보았을 때에는 단지 12.7%를 정도에 불과하다. 그림 4는 governor를 통해 줄일 수 있는 소비전력의 비율을 보여준다[5]. Linux의 Powersave governor를 살펴보면 Performance governor 대비 10%의 소비전력 감소 효과가 있음을 알 수 있다. Ondemand governor나 Userspace governor는 10% 이내의 소비전력 감소 효과가 있다. Performance governor가 최대 clock frequency를 사용하고, Powersave governor가 최소 clock frequency를 사용한다는 점에 근거하여 판단하면, Linux governor의 최대 효과는 10%가 된다는 결론을 얻을 수 있다. 결과적으로 그림 3과 그림 4의 결과를 종합해보면, CPU로 줄일 수 있는 총 전력은 스마트폰 소비전력의 고작 1.27% 밖에 되지 않는다. 이는 스마트폰을 1시간 사용한다고 했을 때, 배터리 수명을 겨우 1분 정도 연장시키는데 불과하다.

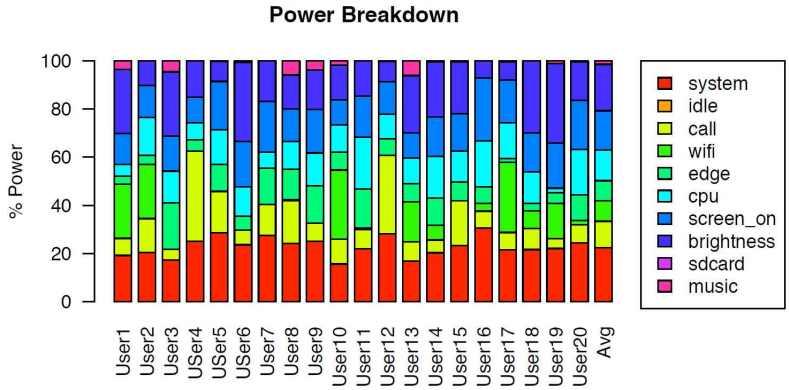


그림 3 스마트폰 각 파트별 소모전력 비율 [7]

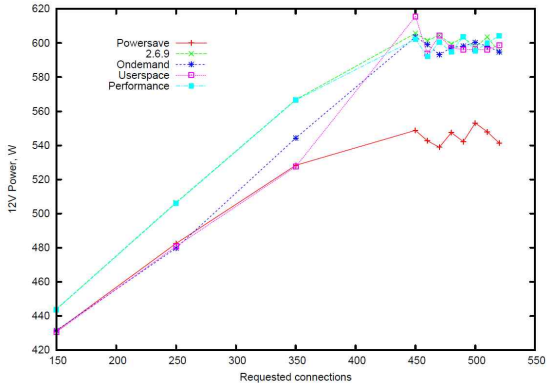


그림 4 Linux governor별 전력 소모량 [5]

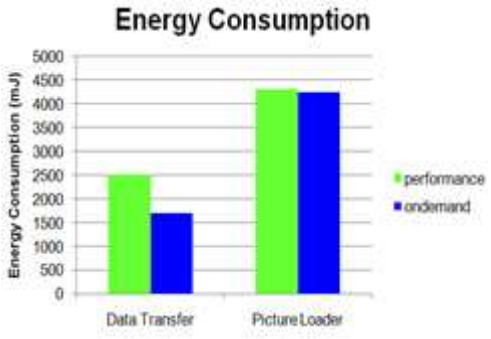


그림 5 CPU 동작 유형별 전력 소모량 [8]

3.2 Linux governor의 한계

Linux governor는 CPU의 응용부하가 큰 경우 그 효과가 매우 제한적이다. Governor는 CPU 사용률을 기반으로 동작한다. CPU 사용률이 높아지면 clock frequency를 높여 소비전력을 늘리고, 사용률이 낮아지면 clock frequency를 낮추어 소비전력을 줄이는 방식이다. 따라서 사용자가 스마트폰에서 앱을 많이 사용하여 CPU의 응용부하가 클 경우, DVFS로 인한 소비전력 감소의 이득이 없다. 그림 5를 보면 data transfer 같이 CPU 사용률이 적은 작업을 할 때는 Ondemand governor가 약 30% 에너지 감소 효과를 보임을 알 수 있다[8]. 그러나 picture loader처럼 CPU 사용률이 높은 작업을 할 때는 에너지 감소 효과가 거의 없음을 알 수 있다.

CPU 응용부하가 낮을 경우에도 문제가 있다. 싱글코어 시스템의 Linux에서는 커널 스케줄러와 governor가 따로 구현된다. 즉 커널 스케줄러와 governor는 개별적으로 동작하며, 커널 스케줄러가 에너지를 줄이기 위한 스케줄링을 할 수가 없기 때문에 governor의 효과가 매우 작다[9]. 한편 멀티코어 시스템에서는 load balancing을 시도할 수 있으나[6], cache affinity로 인해 load balancing이 제한을 받기 때문에 governor의 효과가 작아지게 된다.

3.3 DVFS 최적화의 한계

위와 같은 문제 외에도 DVFS를 적용하여 최적화 시키는 데도 두 가지 한계가 있다.

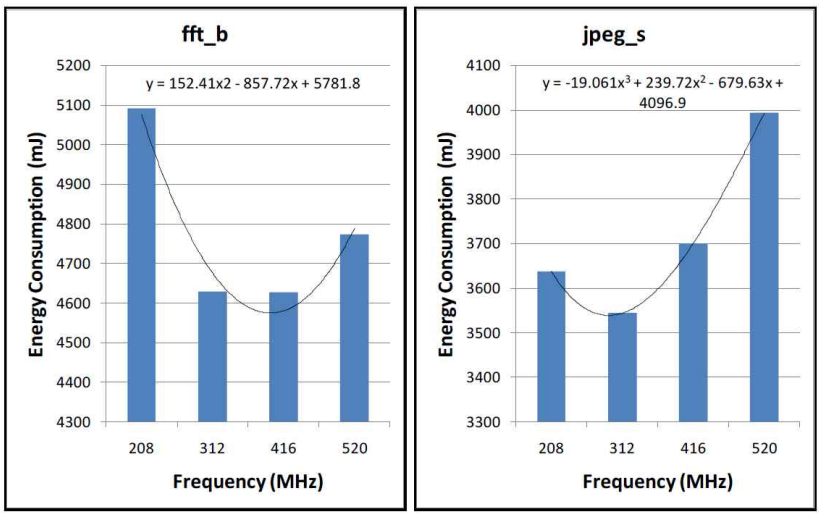
3.3.1 AP별 DVFS 최적화 포인트 상이

DVFS의 효과가 모든 스마트폰에서 유효하지 않다. 표 2는 설정할 수 있는 최소 frequency에서 소모되는 에너지와 최대 frequency에서 소모되는 에너지의 비율을 G1 단말기와 넥서스원 단말기에서 각각 측정한 결과를 보여준다[10]. G1의 경우에는 소모전력의 비율이 120%에 달해, 최소 주파수에서 소비되는 에너지가 최대 주파수에서 소비되는 에너지 보다 오히려 20%가 높아지는 역효과가 발생하였다. 반면에 넥서스원에서는 주파수를 줄임으로써 약 25% 소비되는 에너지 절감효과가 있었다.

또한 그림 6은 CPU clock frequency를 520MHz에서 208MHz로 줄이며 에너지를 측정한 결과를 보여준다. 동작주파수와 전력소모량은 U자형 curve 관계를 가지며 주파수가 어느 값 이하로 작아지면 오히려 전력소모량이 증가함을 알 수 있다[11]. 즉 주파수를 낮추는 것이 무조건적으로 소비되는 에너지를 낮추는 것을 의미하는 것이 아님을 알 수 있다.

표 2 G1과 N1의 최소주파수-최대주파수 에너지 소모 비율 [10]

Benchmark	% Energy		
	Freerunner	G1	N1
equake	95.5	126.0	75.6
vpr	95.8	124.5	75.9
gzip	95.8	120.1	77.7
crafty	95.5	115.6	77.3
mcf	94.9	105.3	65.9



(a) fft_b (b) jpeg_b

그림 6 CPU 실행 작업과 주파수에 따른 소비에너지 변화 [11]

3.3.1 실행 작업 별 DVFS 최적화 포인트 상이

실행 작업 별로 DVFS 최적화 포인트가 상이하다는 사실이 최적 동작 주파수 탐색을 어렵게 한다. 그림 6 (a)의 fft_b 작업의 경우 최소 에너지 소비 주파수가 400MHz임을 알 수 있다. 한편 그림 6 (b)의 jpeg_b 작업의 경우 그 주파수가 300MHz로 나타난다. 즉 실행되는 작업에 따라 에너지 소비가 최저가 되는 주파수인 critical speed가 달라진다[11]. 이와 같은 현상이 발생하는 이유는 다음과 같다. 앱의 메모리 사용 패턴에 따라서 동작주파수와 전력소모량의 linear한 관계가 깨질 수 있다. 이는 동작주파수가 작아져서 앱의 수행 시간이 길어지게 되면, 메모리가 active 상태로 대기하는 시간이 함께 길어져서 대기전력이 커질 수 있기 때문이다. 따라서 동작주파수가 어느 값 이하로 작아지면 오히려 전력소모량이 증가하는 경우가 발생한다. 그림 6의 경우, 동작주파수와 소비전력의 관계는 U자형 함수 관계를 가짐을 알 수 있다.

위의 두 가지 문제를 살펴보면, DVFS의 효율성이 최대가 되는 critical speed는 CPU 제조공정, CPU의 구조, 그리고 수행되는 응용부하에 영향을 받음을 알 수 있다. 이런 최적 주파수를 찾기 위해서는 상당한 overhead가 수반되는 연산을 동적으로 수행하여야 한다.

이상에서 살펴본 바와 같이 DVFS 기술을 실사용 환경에 적용하였을 경우 그 한계는 명확하다. DVFS로 최대의 효과를 얻기 위해서는 각 AP마다 최적의 효율성을 찾기 위한 작업을 해야 하며, 또 스마트폰이 수행하는 작업에 따른 최적화 주파수를 찾아야만 한다. 그리고 이러한 작업들을 완벽히 수행하더라도 스마트폰 CPU에 부하가 적게 걸리는 작업을 할 때만 그 효과를 볼 수 있다. 뿐만 아니라 최대의 효율성을 낼 환경을 갖추더라도 그 비율은 스마트폰 소비전력의 불과 1.27% 밖에 되지 않는다. 이렇듯 DVFS의 원리는 매력적임에 분명하지만 실제로 기술을 사용할 그 효과의 한계성은 명확하다.

4. DVFS의 한계를 극복하기 위한 방향

실사용 환경에서 DVFS의 효과가 매우 제한적인 것으로 분석됨에 따라, 이를 극복하기 위해 다음의 조사 결과를 분석하였다. 그림 7은 20명의 사용자가 6개월간 스마트폰을 사용한 패턴을 도식화한 내용이다[7]. 이를 통해, idle 기간에 소비되는 에너지가 전체 소모되는 에너지의 대략 50%를 차지함을 알 수 있다. 이는 idle 기간 동안에 효과적으로 스마트폰의 소모전력을 줄일 수 있다면, 스마트폰이 소비하는 전체 에너지의 상당량을 절약 할 수 있음을 시사한다. 이렇게 idle 기간에 에너지 소비를 줄이는 방향으로 향후 연

구개발을 수행할 필요가 있다.

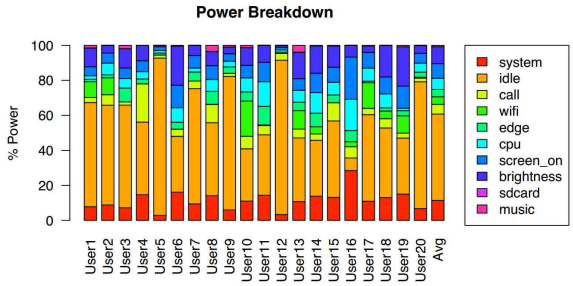


그림 7 Idle이 차지하는 에너지 비율 [7]

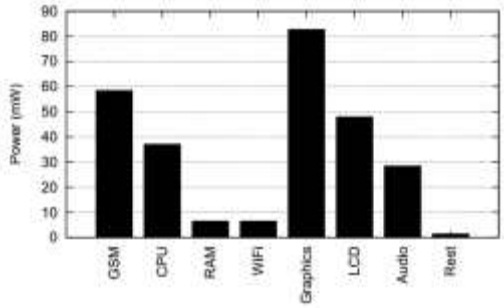


그림 8 Idle 기간 중 CPU의 소비전력 [10]

2장에서 설명하였듯이 DVFS는 주파수를 줄임으로써 에너지 소비를 줄이는 기술이다. 다만 지금까지는 이 기술이 CPU에 유일하게 적용되었다. 그림 8은 idle 기간에 각 장치들이 소모하는 전력의 비율을 보여준다. 여기에서 graphics 장치가 1위, GSM 장치가 2위, CPU가 3위를 차지함을 알 수 있다. 한편 이 세 장치가 소비하는 전력의 양은 idle 소비전력의 약60%를 차지한다. 따라서 CPU 외에도 무선네트워크 장치에 대해 DVFS 적용할 수 있다면 새롭게 소모전력 절감의 가능성을 발견할 수 있다. GSM과 같은 무선네트워크 장치도 CPU와 같은 clock frequency를 갖기 때문에, 무선네트워크 장치 역시 DVFS 기술의 적용대상이 될 수 있다. 일례로 2011년 9월 미시건 대학의 신 강근 교수 연구팀은 idle 기간 동안 WiFi 장치에 DVFS 기술을 적용하여 배터리 지속시간을 최대 54% 연장시킬 수 있다는 연구결과를 발표하였다[12].

또 다른 측면으로 LTE와 TCP 간의 cross layer optimization을 통해 idle 상태에서 소모전력을 감소시킬 수 있을 것이며, WiFi MAC 계층을 최적화시키는 방법을 고려할 수 있다. 또한 AP에서 동적 발열관리를 통해 소모전력을 감소시키는 새로운 방안을 기대할 수 있다.

5. 결론

향후 스마트폰 시장에서의 경쟁은 더욱 치열해질 것으로 예상되며, 소비자들에 대한 차별화 포인트로서 저전력 동작은 매우 중요한 이슈가 될 것이다. 반면 스마트폰의 AP가 듀얼코어를 넘어 쿼드코어로 진화하면서, 소모전력의 문제는 더욱 심각한 이슈가 되었다.

본 고에서는 이러한 상황에 맞추어 Android의 기본 운영체제인 Linux의 저전력 메커니즘을 다각적으로 분석하고, 현재 사용되고 있는 DVFS 메커니즘과 Linux governor의 한계성을 파악하였다. 그리고 이를 통해 저전력 시

스템 설계 문제에 대한 해결책을 모색하였다. 도출된 Linux governor의 한계점은 (1) CPU 전력 소모량의 한계, (2) Linux governor의 한계, (3) DVFS 최적화의 한계이었다. 이러한 문제들을 극복하기 위한 방안으로, DVFS 기술을 스마트폰의 다양한 장치들에 idle한 기간 동안 사용할 수 있도록 하여 기술의 효과를 높이는 연구 방향을 제시하였다.

CPU의 특성을 활용한 DVFS 메커니즘의 진보가 스마트폰의 전력소비 문제를 해결하는데 어느 정도의 도움이 된다는 것은 주지의 사실이다. 하지만 스마트폰에서 차지하는 CPU의 전력소비 비중을 보았을 때, 근본적인 한계를 극복하기는 어렵다. 따라서 이런 문제들을 극복하기 위한 방안으로, (1) 스마트폰이 idle한 기간 동안에 DVFS 기술을 적용하는 장치들을 다양화시키는 방법, (2) LTE-TCP cross layer optimization, (3) WiFi MAC 계층을 최적화시키는 방법, (4) 동적 발열관리를 통한 소모저력 절감하는 기법에 대한 연구개발과 그에 따른 메커니즘에 대한 연구가 있어야 할 것이다.

참고문헌

[1] “ACPI Specification 5.0”, http://www.acpi.info/spec50.htm.

[2] L. Niu, “Energy Efficient Scheduling for Hard Real-Time Systems with Fixed Priority Assignment”, IEEE Performance Computing and Communications Conference (IPCCC), 2010.

[3] K. Choi et al, “Dynamic Voltage and Frequency Scaling for Energy-Efficient System Design”, Doctoral Dissertation, University of Southern California Los Angeles, 2005.

[4] N. Min-Allah et al, “Towards Dynamic Voltage Scaling in Real-Time Systems - A Survey”, International Journal of Computer Sciences and Engineering Systems, Vol.1, No.2, 2007.

[5] V. Pallipadi and A. Starikovskiy, “The Ondemand Governor”, Linux Symposium, 2006.

[6] J. Hopper et al, “Using the Linux CPUFreq Subsystem for Energy Management”, IBM blueprints, 2009.

[7] A. Shye et al. “Into the Wild: Studying Real User Activity Patterns to Guide Power Optimizations for Mobile Architectures”, IEEE/ACM International Symposium on Microarchitecture, 2009.

[8] W. Liang et al, “Design and Implementation of a Critical Speed-based DVFS Mechanism for the Android Operation System”,

Embedded and Multimedia Computing (EMC), 2010.
[9] Linux kernel source. Ver. 2.6.39.4.
[10] A. Carroll et al, “An Analysis of Power Consumption in a Smartphone”, USENIX conference, 2010.
[11] W. Liang et al, “An Energy Conservation DVFS Algorithm for Android Operating System”, Journal of Convergence, 2010.
[12] X. Zhang et al, “E-Mili : Energy-Minimizing Idle Listening in Wireless Networks”, 17th ACM Annual international Conference on MobiCom, 2011.

유 종 훈

2001년 한국과학기술원 전기 및 전자공학과(학사)
2001년～2004년 아이피원(주) 연구원
2005년～현재 서울대학교 전기・컴퓨터공학부 석박사 통합과정

관심분야: 내장형 시스템 소프트웨어, 실시간운영체제, 시스템 가상화

허 승 주

2007년 건국대학교 컴퓨터공학(학사)
2009년 고려대학교 컴퓨터・전파통신공학부(공학석사)
2009년～현재 서울대학교 융합과학기술대학원 지능형 융합시스템학과 박사과정

관심분야: 내장형 시스템 소프트웨어, 운영체제, 멀티코어 스케줄링

홍 성 수

1986년 서울대학교 컴퓨터공학과(학사)
1988년 서울대학교 컴퓨터공학과(공학석사).
1994년 University of Maryland, Department of Computer Science(공학박사).
1996년～1997년 서울대학교 전기공학부 전임강사
1997년～2001년 서울대학교 전기공학부 조교수
2001년～2006년 서울대학교 전기・컴퓨터공학부 부교수
2006년～현재 서울대학교 전기・컴퓨터공학부 정교수
2006년～현재 서울대학교 융합과학기술대학원 지능형융합시스템학과 학과장
2008년～현재 가헌과학기술재단 석좌교수
2009년～현재 차세대융합기술연구원 스마트시스템연구소장

관심분야 : 임베디드 실시간 시스템 설계, 실시간 운영체제, 내장형 미들웨어, 소프트웨어 플랫폼, Linux 커널, 소프트웨어 공학