가변 전압 프로세서를 위한 저전력 소프트웨어 설계 기법

서울대학교 김운석 · 김지홍* · 민상렬*

1. 서 론

최근 무선 인터넷 환경의 확산과 더불어 스마트폰 및 PDA(Personal Digital Assistant)와 같은 이동 내장형 시스템들의 이용과 응용이 확대되고 있다. 이들 시스템들은 대부분 제한된 전원 용량을 가지는 베터리를 기반으로 운용되기 때문에, 이들 시스템에서는 주어진 전원 용량으로 얼마나 오래 동안 동작할 수 있느냐의 문제가 제품들의 시장 경쟁력을 결정하는 중요한 요소가 된다. 따라서, 이동 내장형 시스템에서 저전력 요건은 시스템 설계의 주요 요구사항이라 할 수 있다.

다양한 저전력 설계 기법들 중 동적전압조절 기법(DVS : Dynamic Voltage Scaling)[1]은 CMOS 회로로 구성되는 프로세서의 공급 전압을 온라인 상태에서 조절하여 에너지 소모량을 줄이는 기법이다. 일반적으로 CMOS 회로의 전력(P) 소모는 공급 전압 V_{DD}에 대해 $P \propto V_{DD}^2$인 관계를 가지므로, 공급 전압의 감소는 에너지 소모량 측면에 있어서 매우 효과적이다. 하지만, 공급 전압이 낮아질 때에는 회로의 최대 동작 주파수(f_{clk}) 또한 함께 늦추어져야 하기 때문에 프로세서의 처리량은 줄어들게 된다. 이러한 프로세서 성능과 에너지 소모량 간의 이해득실 관계를 이용하여 주어진 작업에 대한 최적의 전압 및 클락 속도를 설정하는 것을 동적 전압 스케줄링이라 한다.

최근 저전력 이동 내장형 시스템을 위하여 동적전압조절 기능을 활용한 가변 전압 프로세서들이 다수 출시되고 있는데, Crusoe[2], K6III+[3], 그리고 XScale 프로세서[4] 등을 예로 들 수 있다. (이들 프로세서들의 사양은 표 1에 요약되어 있다.) 이와 같이 동적전압조절이 가능한 내장형 시스템을 위한 프로세서들이 다수 출시됨에 따라, 최근 들어 저전력 시스템에 특화된 다양한 동적전압조절 알고리즘들이 제안되고 있다.

동적전압조절 알고리즘은 컴퓨팅 환경에 따라 달리 구현되는데, 주로 대상 응용 프로그램의 주기성 및 시간 제약성 여부에 의해 구별된다. 본 고에서는 응용 프로그램의 시간 제약성에 따라 컴퓨팅 환경을 비실시간 시스템, 연성 실시간 시스템(soft real-time system), 그리고 경성 실시간 시스템(hard real-time system)으로 구분하고, 각 환경에 특화되어 제안된 대표적인 동적전압조절 기법들에 대해 설명한다. 특히, 경성 실시간 시스템을 대상으로 제안된 기법들에 대해서는 에너지 소모량 측면에서의 효율성을 실험을 통하여 비교 평가한다.

본 고의 구성은 다음과 같다. 제2장에서는 비실시간 시스템에서의 동적전압조절 기법들에 대해 살펴보고, 제3장과 제4장에서는 각각 연성 및 경성 실시간 시스템에서의 동적전압조절 기법들에 대해 살펴보도록 한다. 끝으로 제5장에서 결론을 맺도록 한다.

2. 비실시간 시스템에서의 동적전압조절

시스템에 대해 주어진 작업들이 엄격한 시간 제약성을 가지지 않는 비실시간 작업들의 경우에는 프로세서의 성능이 다소 낮아진다고 하여도 시스템 운영에 큰 지장을 초래하지는 않는다. 그러한 작업들의 예로는 워드프로세싱, 계산기, 그리고 기타 후면 처리 작업들을 들 수 있다. 하지만, 이러한 작업들의 경우에도 에너지 소모량을 줄이기 위해 지나치게 공급 전압을 낮게 하여 클록 속도를 줄일 경우, 시스템에 대해서는 과부하를 유발 할 수 있으며, 사용자에 대해서는 서비스 품질의 저하를 가져올 수 있다. 따라

* 종신회원

서, 이러한 작업들을 대상으로 하는 시스템의 경우에도 동적전압조절에는 제약이 따르게 된다.

표 1 상용 가변 전압 프로세서들의 예

	제조사	가용 공급 전압 (V)	가용 클록 속도 (MHz)	공급 전압 단계
Crusoe	Transmeta	1.2-1.65	333-700	3
K6III+	AMD	1.5-1.9	400-550	7
XScale	Intel	0.7-1.75	200-1.000	9

비실시간 작업들을 대상으로 하는 동적전압조절에 관한 연구는 [5][6][7] 등에서 찾아볼 수 있는데, 이러한 연구에서의 중요한 목적은 수시로 변하는 시스템에 대한 작업 요구량, 즉 작업 부하량을 정확히 예측하여 신속히 클록 속도와, 공급 전압을 적응시키는 것이다.

미래의 작업 요구량을 예측하는 가장 일반적인 방법은 지금까지 시스템에 부과된 작업량 정보를 바탕으로 하여 미래를 예측하는 것이다. 즉, 과거의 특정 구간 동안 시스템의 프로세서 활용률 자료를 바탕으로 하여, 앞으로의 특정 구간 동안의 작업 부하량을 예측하는 것이다. 예를 들어, 샘플링 구간의 길이를 1초라 할 때, 과거 1초전에서부터 현재의 시점 사이의 프로세서 활용률이 50% 였다면, 현재의 시점에서 향후 1초 동안의 프로세서 활용률도 50% 정도가 될 것이라 가정하는 것이다. 이 때, 프로세서는 저부하 상태에 놓일 것이므로, 프로세서의 클록 속도와 공급 전압은 낮추어질 수 있다. 반대로 과거의 프로세서 활용률이 높은 경우에는 프로세서가 과부하 상태에 있다고 가정을 하고, 프로세서의 클록 속도와 공급 전압을 높이게 된다. 이러한 기법들은 구간기반 예측(interval-based prediction)기법이라 불리어진다.

Weiser 등에 의해 [5]에서 처음 제시된 구간기반 예측 방법은 차후 Govil 등에 의해 [6]에서 여러 방향으로 발전되었다. Govil 등은 미래의 작업 요구량을 예측하기 위하여 단순히 과거의 프로세서 활용률 정보를 사용하는 대신 다양한 형태의 작업 부하량 모델에 기반하여 개선된 방법들을 제시하였는데, 대표적인 것들로는 AGED-AVERAGE 방법과 PATTERN 방법을 들 수 있다. AGED-AVERAGE 방법에서는 1) 보다 먼 과거의 정보를 반영함으로써 프로세서의 클록 속도가 급격히 변화하지 않고 점진적으로 변화할 수 있도록 하였고, 2) 먼 과거에 비해 가까운 과거의 정보에 보다 높은 가중치를 둠으로써 작업 부하량 변화를 적절히 반영할 수 있도록 하였다. 이에 반해, PATTERN 방법에서는 시스템에 대한 작업 부하량 변화가 일정 유형을 가진다고 가정하고, 과거 작업 부하량 변화가 가지는 유형을 분석하여 미래의 작업 부하량 유형을 예측한다.

이외에도 과거의 정보를 바탕으로 하여 미래의 시스템 작업 부하량을 예측하는 방법들이 다수 제시되었으나, 일반적으로 시스템의 작업 부하량 변화는 매우 불규칙하기 때문에 [6]과 [7]에서 보여진 것과 같이 과거의 정보를 반영하는 어떤 특정 방법이 다른 방법들에 비해 일반적으로 더 나은 효과를 가져오기는 어렵다. 또한, 미래의 작업 부하량에 대한 예측의 정확성과 변화하는 작업 부하량에 대해 적응하는 민첩성은 이해득실 관계를 가지기 때문에 구간기반 동적전압조절 방법은 어느 정도 한계를 가진다고 할 수 있다. 이러한 문제는 Grunwald에 의해 [7]에서 잘 설명되었는데, 샘플링 구간의 길이를 길게 하여 긴 구간을 관찰할 경우, 작업 부하량 변화를 정확히 예측할 순 있지만, 예측된 결과를 클록 속도 및 공급 전압 조절에 반영하는 시점이 너무 늦어져 적응 민첩성이 떨어진다는 문제가 있으며, 샘플링 구간의 길이를 짧게 하여 클록 속도 및 공급 전압조절 시점을 조밀하게 하는 경우, 적응 민첩성은 높아지지만, 작업 부하량 변화를 정확히 예측하지 못한다는 문제가 있다. 이러한 문제를 해결하기 위해서는 시스템에 요구되는 작업량에 대해 보다 자세한 정보를 가지고 있는 운용체제 혹은 사용자에 의해 주어지는 정보를 바탕으로 프로세서에 대한 성능 요구조건을 분석하는 작업이 필요하다.

3. 연성 실시간 시스템에서의 동적전압조절

MPEG 동화상 재생, 문자 및 음성 인식, 그리고 통신 응용물과 같이 사용자와의 직접적인 상호작용을 하는 응용물들의 경우 프로세서 성능에 따라 사용자에 대한 응용물의 서비스의 질이 달라지게 되므로, 이러한 응용물들에 의한 작업들은 어느 정도 시간 제약성을 가진다고 할 수 있다. 여기서, 사용자에 대한 서비스의 질과 시스템의 에너지 소모량간에는 이해득실 관계가 존재하는데, 즉 사용자에 대한 서비스의

질을 높이기 위해 클록 속도를 높여 시스템을 운영할 경우, 시스템의 에너지 소모량은 늘어나게 되고, 반대로 에너지 소모량을 줄이기 위해 시스템을 낮은 클록 속도로 운영할 경우, 사용자에 대한 서비스의 질이 낮아지게 된다.

Yuan 등은 [8][9]에서 연성 실시간 응용물들에 대해 요구되는 서비스의 질을 유지하면서 에너지 소모량을 줄이기 위한 방법으로 기존의 고정 대역폭 서버(constant bandwidth server) 알고리즘을 이용한 동적전압조절 기법을 제시하였다. 이들의 방법에서는 각 연성 실시간 응용물을 하나의 주기적 태스크로 보고, 이들에 대해 주기적으로 프로세서 대역폭을 할당하여 이들의 서비스의 질을 유지할 수 있도록 하였으며, 할당 후 남는 대역폭을 이용하여 프로세서의 클록 속도와 공급 전압을 낮출 수 있도록 하고 있다. 또한, 이들은 비실시간 작업들에 대해서도 하나의 고정 대역폭 서버를 할당 할 수 있도록 하여 연성 실시간 작업들과 비실시간 작업들이 함께 스케줄될 수 있도록 하였다.

이들의 SRT-DVS(soft real-time dynamic voltage scaling) 알고리즘을 좀더 자세히 살펴보면, 연성 실시간 작업 τ_i이 도착할 때, 이에 대해 하나의 고정 대역폭 서버 SR_i를 할당한다. 이때, 서버가 가지는 대역폭은 사용자에 의해 주어진 주기 (T_i)와 처리량 (Q_i)에 의해 정해지는데, 주기 T_i마다 Q_i만큼의 프로세서 싸이클이 할당된 서버를 통해 해당 응용물에 제공된다. τ_i에 의한 프로세서 활용률은 $U_i=Q_i/T_i$ 와 같이 계산될 수 있는데, 만일 시스템에 총 N 개의 서버가 할당되어 있다면, 이들에 의한 프로세서 활용률은 $U_T=\sum_{j=1}^{N} U_j$가 된다. 이와 같이 계산된 프로세서 활용률을 바탕으로 하여 프로세서의 클록 속도를 $f_{CLK}=U_T \cdot f_{MAX}$로 설정하는 경우 각 응용물에 대해서는 요구된 프로세서 대역폭을 제공할 수 있으며 프로세서는 항상 100%의 활용률을 가지는 상태에 놓이게 되고, 낮아진 클록 속도 및 공급 전압에 의해 프로세서의 에너지 소모량은 줄어들게 된다. (f_{MAX}는 프로세서가 지원하는 가장 빠른 클록 주파수를 나타낸다.)

SRT-DVS 알고리즘에서는 응용물의 서비스의 질과 에너지 소모량간의 이해득실을 각 서버에 할당되는 대역폭을 통해 조절할 있는데, 예를 들어, 각 서버의 대역폭을 낮출 경우, 응용물들의 서비스의 질은 낮아지는 반면 전체적인 프로세서 활용률이 낮아져 에너지 소모량은 줄어들게 된다. Yuan 등은 [9]에서 이러한 이해득실 관계를 실험을 통해 비교하고 있다.

4. 경성 실시간 시스템에서의 동적전압조절

비실시간 혹은 연성 실시간 시스템과는 달리 경성 실시간 시스템에서는 주어진 작업들이 엄격한 시간 제약성을 가진다. 이러한 경성 실시간 시스템에서는 주어진 작업들이 종료 시한을 위배할 경우 시스템의 기능성이 저해되기 때문에 클록 속도의 조절은 작업들의 시간 제약성에 의해 엄격히 제한된다. 경성 실시간 시스템에서의 동적전압조절 문제는 크게 둘로 나뉘는데, 하나는 슬랙 측정(slack estimation) 문제로 각 스케줄 시점에서 실시간 작업들이 가지는 슬랙을 계산하는 것이고, 다른 하나는 슬랙 배분(slack distribution) 문제로 계산된 슬랙을 바탕으로 어느 정도의 클록 및 전압 수준으로 설정하느냐 하는 것이다.

경성 실시간 시스템에서의 슬랙 계산 문제는 기존의 실시간 스케줄링 연구에서 많이 다루어져 왔다. 하지만, 기존 연구에서의 최적화된 방법들은 시간 및 공간 복잡도가 높아 제한된 컴퓨팅 자원을 가지는 이동 내장형 시스템에서는 일반적으로 적용에 어려움이 따른다. 따라서, 대부분의 동적전압조절 알고리즘들에서는 시간 및 공간 복잡도가 높지 않은 경험적 방법(heuristic)을 이용하고 있다. 슬랙 배분 문제의 경우에 있어서도 그 계산 복잡도에 근거하여 대부분 경험적 방법들이 이용되고 있는데, 주로 각 스케줄링 시점에서 계산된 슬랙을 그 시점에서 스케줄 되는 태스크 인스턴스가 모두 이용할 수 있도록 하는 방법을 택하고 있다.

태스크들이 가지는 슬랙은 크게 두 가지로 나뉠 수 있는데, 하나는 정적 슬랙으로 주어진 태스크 집합의 명세상에 들어나는 슬랙을 말하며, 다른 하나는 동적 슬랙으로 태스크들의 실제 실행시간과 이들의 명세상의 최악실행시간과의 차이에서 발생하는 슬랙을 말한다. 예를 들어, 주기가 5초이고 최악 실행시간이 3초인 태스크가 주어졌다고 하자. 주어진 작업은 최악의 경우에도 3초의 시간만을 소모하므로, 매 5초마다 2초의 정적 슬랙이 주어진다고 할 수 있다. 만일, 주어진 작업이 실제 실행에 있어서 1초만을 소모하였다고 하였을 때, 시스템은 최악 실행에 비해 2초의 시간이 줄어들었으므로 2초의 추가적인 동적 슬

랙을 얻었다고 할 수 있다.

경성 실시간 시스템에서, 위와 같은 슬랙을 오프라인 및 온라인에서 계산하여 클록 및 전압 수준을 결정하는 동적전압조절 알고리즘들은 전압조절 시점의 단위에 따라 크게 두 가지로 나눌 수 있는데, 하나는 태스크간 조절 방법(InterDVS)이고 다른 하나는 태스크내(IntraDVS) 조절 방법이다[10]. 태스크간 조절 방법은 각 스케줄링 시점에서 태스크 단위로 클록 및 전압을 조절하는 것을 말하며, 태스크내 조절 방법은 한 태스크의 실행 내에서 클록 및 전압을 조절하는 것을 말한다. 이들간의 가장 큰 차이점은 전자의 경우에는 현재 실행되고 있는 태스크에 의해 발생한 동적 슬랙을 다음 실행될 태스크가 이용하는 반면 후자의 경우에는 현재 실행되고 있는 태스크가 자신으로부터 비롯되고 있는 슬랙을 자신의 실행 중 이용한다는 점이다. 지면상의 제약으로 본고에서는 태스크간 조절 방법만 다루도록 한다. 일반적으로 많이 이용되는 경성 실시간 시스템을 위한 동적전압조절 기법들이 표 2에 정리되어 있다.

표 2 동적전압조절 기법들의 분류

	동적전압조절 기법	결정 시점
IntraDVS	(1) 경로 기반 기법	오프라인
	(2) 통계 기반 기법	
InterDVS	(3) 최대 고정 속도 기법	오프라인
	(4) Stretching to NTA 기법	온라인
	(5) 우선순위기반 슬랙측정 기법	
	(6) 프로세서 활용률 갱신 기법	

4.1 최대고정속도 기법

가장 일반적으로 이용되는 정적 슬랙 측정 방법 중에 하나는 최대고정속도를 계산하는 것이다[11]. 최대고정속도란 주어진 태스크 집합에 의한 모든 태스크 인스턴스들이 최악실행시간을 요구하는 경우에도 스케줄 가능성이 보장될 수 있는 최저 가능 클록 속도를 의미한다. 주로 기존의 실시간 스케줄링에서 이용되는 스케줄 가능성 검사식을 이용한다. 예를 들어, EDF(Earliest Deadline First) 스케줄링의 경우, 주어진 태스크 집합을 최대 클록 속도 f_{MAX}로 수행하는 경우 프로세서 활용률 U가 1보다 작거나 같으면 주어진 태스크 집합은 항상 스케줄 가능하다. 이 경우, 클록 속도가 $f_{MCS}=U \cdot f_{MAX}$ 로 낮추어져 클록 속도 f_{MCS}하에서 태스크 인스턴스들의 최악실행시간이 $\frac{1}{U}$만큼 늘어난다 하더라도, 주어진 태스크 집합의 최악 프로세서 활용률은 1 이 되며, 여전히 스케줄 가능하다. EDF 스케줄링의 경우, 이와 같이 계산된 클록 속도 f_{MCS}는 주어진 태스크 집합의 스케줄 가능성을 보장하는 한도 내에서 이용될 수 있는 최저 클록 속도가 되며 이를 최대고정속도라 한다. RM (Rate Monotonic) 스케줄링의 경우에도, 다소 복잡하지만, 기존의 스케줄 가능성 분석방법을 이용하여 비슷한 방식으로 최대고정속도를 계산할 수 있다 [11][12].

4.2 Stretching to NTA 기법

비록 주어진 태스크 집합이 최대고정속도로 스케줄 되더라도, 태스크들의 실제 실행시간은 일반적으로 그들의 최악실행시간보다 작기 때문에, 태스크들은 대부분 동적 슬랙을 가진다. 동적 슬랙을 계산하는 가장 간단한 방법 중에 하나는 다음 태스크의 도착 시간을 이용하는 것이다[12]. 즉, 다음 태스크의 도착 시점이 충분히 멀다고 했을 때, 현재 스케줄된 태스크가 그 도착 시점 이전에 가까스로 종료될 수 있도록 클록 속도를 낮추어 태스크의 실행시간을 늘이는 것이다.

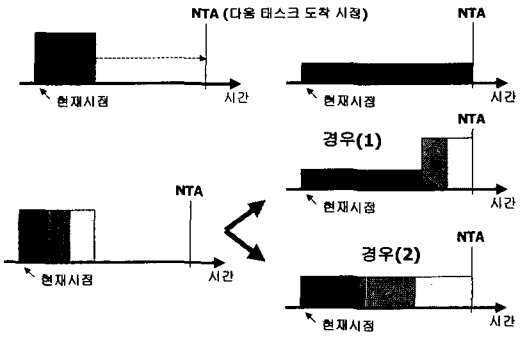


그림 1 Stretching to NTA 기법들의 예

그림 1은 이 방법의 한 예를 보여주고 있다. 그림 1의 상단 그림처럼 단일 태스크만이 활성화된 경우에는 단순히 다음 태스크의 도착 시점까지 이의 실행을 늘일 수 있다. 그림 1의 다른 그림들과 같이 다수의 태스크가 활성화되어있는 경우에는, 몇가지 선택들이 있을 수 있는데, 그림 1의 경우(1)과 같이 현재 스케줄 되는 태스크에 대한 클록 속도만을 낮출 수도

있으며, 경우(2)와 같이 다른 활성화된 태스크들까지 고려하여 클록 속도를 조절 할 수도 있다[13].

4.3 우선순위기반 슬랙 측정 기법

이 방법도 EDF 나 RM 같은 우선순위기반 스케줄링의 기본적인 특성을 이용한다. 기본 개념은, 높은 우선순위의 태스크가 이의 최악실행시간보다 일찍 종료하는 경우, 잇따라 스케줄 되는 낮은 우선순위의 태스크들이 상위 태스크가 남긴 시간을 이용한다는 것이다. 물론, 상위 태스크 또한 하위 태스크가 남긴 시간을 이용할 수도 있다. 하지만, 이의 경우 계산 복잡도가 다소 높아진다는 단점이 있다. 따라서, 대부분 하위 태스크가 남긴 시간을 계산하는 과정에서는 간단한 경험적 방법들을 이용한다[10][14].

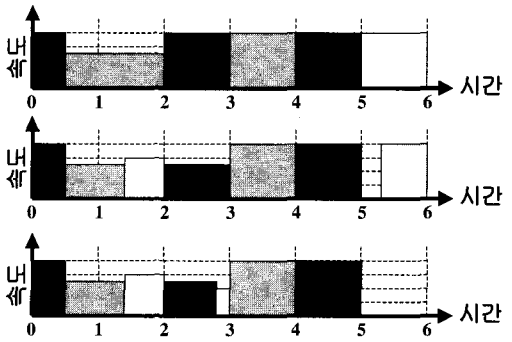


그림 2 우선순위기반 슬랙측정 기법에 의한 동적전압조절의 예

그림 2는 우선순위기반 슬랙측정 방법에 근거한 동적전압조절의 예를 보여주고 있다. 모든 태스크들의 최악 실행시간이 1이라고 하고, 주기가 2, 3, 그리고 6인 태스크들이 주어졌다고 하자. 최초 스케줄되는 태스크 인스턴스가 0.5초 만에 종료하는 경우, 다음 스케줄되는 태스크는 첫 번째 태스크 인스턴스가 남긴 0.5초의 시간을 자신의 슬랙으로 이용할 수 있다. 따라서, 자신이 이용할 수 있는 시간 1초와 얻어진 슬랙 0.5초를 활용할 수 있으므로, 그림 2의 상단 그림에서와 같이 t=0.5 인 시점에서 클록 속도는 1/1.5 의 비율로 낮추어질 수 있다. 나머지 태스크 인스턴스들도 이와 유사한 방식으로 낮은 클록 속도로 스케줄 될 수 있는데, 그림 2의 중간과 하단 그림은 이를 보여주고 있다.

우선순위기반 슬랙측정 기법은 EDF 와 같은 동적 우선순위 스케줄링에서는 태스크들의 우선순위가 동적으로 변화하기 때문에 거의 대부분의 태스크 인스턴스들이 낮은 클록 속도로 스케줄 될 수 있다는 장점을 가지는 반면, RM 과 같은 고정우선순위 스케줄링에서는 태스크들의 우선순위가 고정되어 일부 태스크들은 슬랙을 전혀 활용하지 못하게 되는 슬랙 이용의 불균형이 발생한다는 단점이 있다.

4.4 프로세서 활용률 갱신 기법

프로세서의 실제 활용률은 일반적으로 주어진 태스크 집합 명세에 의해 계산된 최악 프로세서 활용률보다 낮다. 프로세서 활용률 갱신 기법은, 태스크 인스턴스들이 조기 종료하는 경우, 이의 실제 실행시간을 기반으로 하여 새로운 최악 프로세서 활용률을 계산하고, 이를 바탕으로 하여 프로세서의 클록 속도를 조절한다[13].

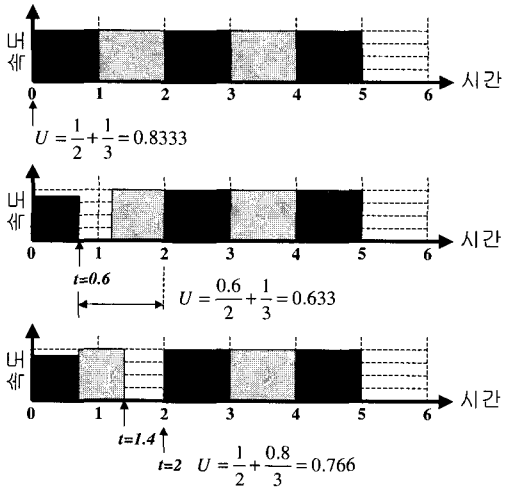


그림 3 프로세서 활용률 갱신 기법에 의한 동적전압조절의 예

그림 3은 그 예를 보여주고 있다. 주기가 2이고 3인 두 개의 주기적 태스크가 주어지고 이들의 최악 실행시간이 모두 1일 때, 최초 스케줄 되는 시점에서 프로세서 활용률은 0.8333 (=1/2+1/3) 이되고 클록 속도는 이에 기반하여 조절된다 (그림 3의 상단 그림). 만일 최초 스케줄된 태스크 인스턴스가 t=0.6 인 시점에서 조기 종료한다면, 첫 번째 스케줄 된 태스크의 실제 프로세서 활용률은 0.3(=0.6/2)이 되므로, 그 시점에서부터의 프로세서 활용률은 다시

0.6333(=0.6/2+1/3) 으로 갱신되고 클록 속도 또한 이를 바탕으로 다시 설정된다(가운데 그림). 물론, 주기적 태스크가 다시 활성화되는 시점에서는 이의 최악 실행시간을 기준으로 다시 프로세서 활용률이 갱신되고 클록 속도가 조절된다(하단 그림 참조).

이 방법의 가장 큰 장점은 구현이 매우 용이하다는 것인데, 매 스케줄링 시점에서 태스크들의 실제 실행시간을 기준으로 프로세서 활용률을 갱신하기만 하면 되기 때문이다.

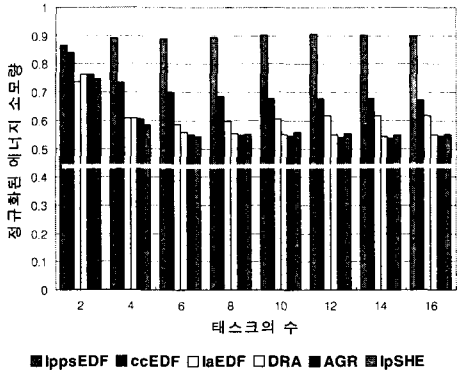
4.5 동적전압조절 알고리즘 및 성능 평가

실제 제안된 여러 동적전압조절 알고리즘들에서는 위에서 설명된 여러 동적전압조절 기법들이 혼용되어 사용된다. 예를 들어, Shin 등에 의해 제안된 lppsEDF 와 lppsRM 알고리즘들을 보면, 오프라인 상태에서 정적 슬랙을 바탕으로 하여 최대고정속도를 계산하며, 온라인 상태에서는 Stretching to NTA 기법을 이용하여 동적 슬랙을 계산한 후, 이 두 정보를 바탕으로 클록 속도 및 전압을 결정 및 설정한다[11]. 다른 알고리즘들에서도 다수의 기법들이 혼용되고 있는데, 이를 정리하여 보면 표 3과 같다.

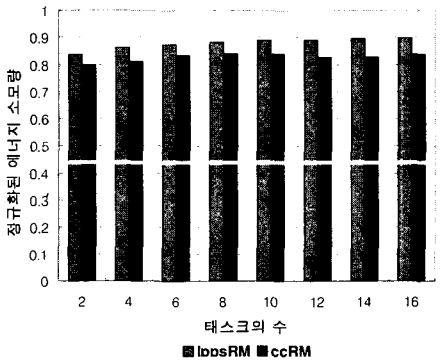
표 3 동적 전압 조절 알고리즘들

분류	스케줄정책	DVS 정책	사용된 기법들
InterDVS	EDF	lppsEDF[11]	(3)+(4)
		ccEDF[13]	(6)
		laEDF[13]	(6)
		DRA[14]	(3)+(4)+(5)
		AGR[14]	(4) +(5)
		lpSHE[10]	(3)+(4)+(5)
	RM	lppsRM[11]	(3)+(4)
		ccRM[13]	(3)+(4)
IntraDVS	스케줄 정책에 독립적	intraShin[16]	(1)
		intraGruian[12]	(2)

표 3에서 괄호 안에 표시된 번호들은 표 2에서의 각 기법들의 번호에 해당되는 것으로, lppsEDF을 예로 들면 최대고정속도 기법((3))와 Stretching to NTA 기법((4))을 이용하므로 (3)+(4)와 같이 표시된다. (a)*와 같이 표시된 경우는, 동일한 기법이 다소 변형된 (향상된) 형태로 이용되는 경우를 의미한다. 예를 들어, ccRM 의 경우 Stretching to NTA 기법을 이용하지만 단일 태스크 만이 활성화 된 경우에 슬랙 측정이 가능한 lppsRM 과는 달리 다수의 태스크가 활성화 된 경우에도 가능하도록 확장하였다. 이러한 경우 (4)*와 같이 표시하였다.



(a) EDF 스케줄링



(b) EDF 스케줄링

그림 4 태스크간 동적전압조절 알고리즘들의 에너지 효율성 비교

이들 제안된 알고리즘들의 에너지 효율성은 [10]에서 자세히 비교되었는데, 그림 4는 그중 일부 결과를 보여준다. 그림에서 흰선은 이론적으로 가능한 최저 에너지 소모량을 나타낸다. 그림에서 보여지듯이, DRA, AGR, 그리고 lpSHE 알고리즘들은 상당히 좋은 에너지 효율성을 보이는데, 이론적 하한선과 비교하여 10 % 정도의 차이만을 보인다. 이는 여러 동적전압조절 기법들 중 우선순위기반 슬랙 측정 방법이 가장 효율적임을 보여준다. 또한, RM 스케줄링 정책에 대한 알고리즘들의 경우, 이론적 하한선과 비교해 상당히 많은 에너지 소모량 차이를 보이고 있다. RM 스케줄링 정책의 경우, 이의 스케줄링 가능성 분석의 복잡도 때문에 많은 알고리즘들이 제시되지 못해왔기

때문에 기존 알고리즘들의 성능이 아직 저조함을 말한다. 따라서, RM 스케줄링 정책에 대한 적합한 동적 전압 알고리즘의 개발이 요구된다고 할 수 있다.

5. 결 론

본 고에서는 저전력 시스템을 설계하는 가장 효율적인 기법으로 대두되고 있는 가변 전압 프로세서를 위한 동적전압조절 기법들에 대해 살펴보고, 이들의 장단점, 그리고 에너지 효율성에 대해 살펴보았다. 현재 제안되어 있는 다양한 동적전압조절 알고리즘들은 운용환경에 따라 약간의 차이는 있으나 시스템이 유휴상태에 있는 슬랙 구간을 효율적으로 찾아내는데 초점이 맞추어져 있다. 제안된 다양한 기법들 중 어떤 기법이 주어진 시스템에 가장 높은 에너지 효율성을 가져다줄 수 있는지는 시스템이 수행하는 작업들의 특성(예: 시간 제약성, 작업부하량의 분포)에 따라 달라진다. 작업의 특성에 따른 최적 전압조절기법에 대해서는 향후 좀 더 연구되어야 할 부분이라 생각된다.

본 고에서는 제안된 기법들을 구현하는데 있어서의 부대비용이나 구현상의 어려움 등은 고려하지 않았으나, 저전력 시스템 설계자들에게 실질적으로 유용한 정보를 제공해 주기 위해 비교 분석된 알고리즘들의 구현상의 복잡도(예: 시간 및 에너지 소모) 이해는 중요한 향후 과제로 남아있다.

참고문헌

[1] T. Sakurai and A. Newton. Alpha-power Law MOSFET Model and Its Application to CMOS Inverter Dealy and Other Formulars. IEEE Journal of Solid State Circuits, 25(2):584-594, 1990.

[2] Transmeta Corporation. Crusoe Processor. http://www.transmeta.com, June 2000.

[3] AMD Corporation. PowerNow! Technology. http://www.amd.com, December 2000.

[4] Intel Corporation. Intel XScale Technology. http://developer.intel.com/design/intelxscale, November 2001.

[5] M. Weiser, B. Welch, A. Demers and S. Shenker, Scheduling for Reduced CPU Energy. In Proceedings of the Symposium on Operating Systems Design and Implementation, pages 13-23, 1994.

[6] K. Govil, E. Chan and G. Wasserman, Comparing Algorithm for Dynamic Speed-Setting of a Lower-Power CPU. In Proceedings of the First Annual International Conference on Mobile Computing and Networking, pages 13-25, November 1995.

[7] D. Grunwald, P. Levis, and K. I. Farkas. Policies for Dynamic Clock Scheduling. In Proceedings of the 4th Symposium on Operating Systems Design and Implementation, pages 73-86, October 2000.

[8] W. Yuan and K. Nahrstedt, A Middleware Framework Coordinating Processor/Power Resource Management for Multimedia Applications. In Proceedings of IEEE Globecom 2001, pages 1984-1988, November 2001.

[9] W. Yuan and K. Nahrstedt, Integration of Dynamic Voltage Scaling and Soft Real-Time Scheduling for Open Mobile Systems. In Proceedings the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '02), pages 105-114, May 2002.

[10] W. Kim, J. Kim, and S. L. Min, A Dynamic Voltage Scaling Algorithm for Dynamic-Priority Hard Real-Time Systems Using Slack Time Analysis. In Proceedings of Design Automation and Test in Europe (DATE'02), pages 788-794, March 2002.

[11] Y. Shin, K. Choi, and T. Sakurai. Power Optimization of Real-Time Embedded Systems on Variable Speed Processors. In Proceedings of the International Conference on Computer-Aided Design, pages 365-368, November 2000.

[12] F. Gruian. Hard Real-Time Scheduling Using Stochastic Data and DVS Processors. In Proceedings of the International Symposium on Low Power Electronics and Design, pages 46-51, August 2001.

[13] P. Pillai and K. G. Shin. Real-Time Dynamic

Voltage Scaling for Low-Power Embedded Operating Systems. In Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP' 01), pages 89-102, October 2001.

[14] H. Aydin, R. Melhem, D. Mosse, and P. M. Alvarez. Dynamic and Aggressive Scheduling Techniques for Power-Aware Real-Time Systems. In Proceedings of IEEE Real-Time Systems Symposium, December 2001.

[15] S. Lee and T. Sakurai. Run-time Voltage Hopping for Lowpower Real-Time Systems. In Proceedings of the 37th Design Automation Conference, pages 806~809, June 2000.

[16] D. Shin, J. Kim, and S. Lee. Intra-Task Voltage Scheduling for Low-Energy Hard Real-Time Applications. IEEE Design and Test of Computers, 18(2):20-30, March 2001.

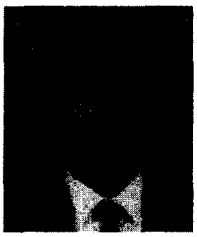
김 운 석

1997 홍익대학교 컴퓨터공학과 학사
1999 서울대학교 컴퓨터공학과 석사
2000~현재 서울대학교 전기컴퓨터공학부 박사과정 재학중
관심분야:실시간 시스템, 저전력 시스템, 멀티미디어 시스템임
E-mail: wskim@archi.snu.ac.kr

김 지 홍

1986 서울대학교 계산통계학과 학사
1988 University of Washington 컴퓨터과학과 석사
1995 University of Washinton 컴퓨터과학 및 공학과 박사
1995~97 미국 Texas Instruments 사 선임연구원.
1997~현재 서울대학교 전기컴퓨터공학부 부교수
관심분야:컴퓨터구조, 내장형 시스템, 저전력 시스템, 멀티미디어 시스템임
E-mail: jihong@davinci.snu.ac.kr

민 상 렬



1983 서울대학교 컴퓨터공학과 졸업
1985 서울대학교 컴퓨터공학과 석사
1989 University of Washington 전산학 박사
1989~90 IBM T.J. Watson Research Center 객원 연구원.
1990~92 부산대학교 컴퓨터공학과 조교수
1992~현재 서울대학교 전기컴퓨터공학부 교수
관심분야:Computer Architecture, Parallel Processing, Computer Performance Evaluation임
E-mail:symin@dandelion.snu.ac.kr