실시간 운영체제에서 작업량 관찰에 기반한 저전력 기법의 설계 및 구현

Design and Implementation of Low-Power Technique based on Monitoring Workload on Real-Time Operating Systems

조문행, 정명조, 김용희, 이철훈
충남대학교 컴퓨터공학과

Moon-Haeng Cho(root4567@cnu.ac.kr), Myoung-Jo Jung(mjjung@cnu.ac.kr),
Yong-Hee Kim(yonghee@cnu.ac.kr), Cheol-Hoon Lee(clee@cnu.ac.kr)

요약

오늘날의 내장형 이동 시스템은 MP3플레이어나 디지털 캠코더와 같이 하나의 기능만을 지원했던 단일 응용프로그램 시스템에서 PMP(Portable Multimedia Player), PDA(Personal Digital Assistants)와 같이 MPEG, MP3플레이어, 전자사전, DMB(Digital Multimedia Broadcasting), 게임, 통신 기능 등을 모두 포함하는 하나의 디지털 컨버전스 기기로 변화해 가고 있다. 이러한 변화는 CPU 성능 향상과 메모리, 저장장치의 증가, LCD의 크기 증가와 같은 하드웨어적인 요구사항의 증가로 이어졌고, 이로 인해 단말기에서 소모하는 전력이 그에 비례하여 증가하였다. 소모 전력의 증가에 따른 배터리 용량의 증가는 더딘 상황이며, 이를 해결하기 위해 소프트웨어적으로 소모 전력을 감축시키는 연구들이 많이 진행되고 있다. 본 논문에서는 소모 전력 감축 기법이 적용된 실시간 운영체제 UbiFOSTM을 ARM9계열의 MBA2440에 탑재하였다. 전력 감축 기법으로 시스템 작업량 관찰에 기반한 동적 전력 관리 기법과 장치 전력 관리 기법을 적용하였다. 본 논문의 저전력 기법을 통해 작업량에 따라 24% 이상의 소모 전력 감축효과가 있음을 확인하였다.

■ 중심어 : | 저 전력 | 전력 관리 기법 | 실시간운영체제 | 작업량 관찰 |

Abstract

In recent years, embedded mobile systems have been expanding their application domains from embedded portable devices which only execute a specialized application such as MP3 player or digital camcoder to digital convergence devices which execute more complicated applications converged various functionalities such as video and audio play, digital dictionary, DMB, games, phone, etc. As it requires the increasing hardware performance such as more faster CPU and more larger RAM, display, disk size, it has brought about a corresponding increase in power consumption. However, coupled with relatively small gains in battery capacity over recent years, the importance of software architecture including intelligent power management has become paramount. In this paper, we have ported UbiFOSTM with energy saving techniques on the ARM9-based MBA2440 platform. For energy savings, we adapted the dynamic power management and the device power management schemes based on monitoring workload. Experimental results with some well-known applications show that proposed low power technique could save energy up to 24 %.

■ keyword : | Low-Power | Power Management Techniques | Real-Time Operating Systems | Monitoring Workload |

* 본 연구는 정보통신부의 선도기반기술개발사업의 지원으로 수행되었습니다.

접수번호 : #070326-002
접수일자 : 2007년 03월 26일
심사완료일 : 2007년 04월 13일
교신저자 : 이철훈, e-mail : clee@cnu.ac.kr

I. 서 론

수행속도 향상에 초점을 맞춰왔던 과거의 시스템과는 달리, 근래의 내장형 이동 시스템은 수행속도 뿐만 아니라 전력 소모라는 매우 중요한 성능지표가 대두 되었다[1].

휴대폰, PMP(Portable Multimedia Player), Navigation, DMB(Digital Multimedia Broadcasting) 등과 같은 내장형 이동 시스템은 사용자의 다양한 요구에 부합하기 위해 응용프로그램이 고도화되고 서로 간의 기능이 융합되면서 높은 성능의 CPU 구동주파수, 메모리(RAM)와 저장장치(Flash Memory)의 용량 증가, LCD의 크기 증가와 같은 하드웨어적인 요구와 이를 관리하기 위한 실시간 운영체제에 대한 요구가 증대되었다.

배터리로 동작하는 내장형 시스템의 성능 요구사항에 대한 증가는 시스템에서 소모하는 전력의 양을 증가시켰으며, 소모 전력의 증가는 내장형 시스템의 사용시간 감소로 이어지고 있다. 배터리로 동작하는 내장형 이동 시스템의 경우, 시스템에서 소모하는 전력은 시스템 하드웨어를 구성하는 방법과 시스템을 구동하는 소프트웨어에 의해 좌우 된다. 종전의 소프트웨어 최적화는 동작 속도, 코드의 크기 및 코드의 가독성에 주로 영향을 받아 왔지만, 최근에는 같은 결과를 내놓는 소프트웨어라 하더라도 시스템의 전력 소모를 고려하여 보다 적은 전력으로 동일한 결과를 얻을 수 있는 저전력 소프트웨어에 대한 연구가 활발히 진행되고 있다. 이와 같이 전력 소모를 줄이는 노력은 반도체 설계, 시스템 하드웨어 설계와 같은 하드웨어 부분과 컴파일러의 최적화, 소프트웨어의 최적화, 저전력 소프트웨어와 같은 소프트웨어 부분 모두에서 다루어져야 한다.

특히, 저전력 소프트웨어를 사용한 전력 소모 최적화는 생산 시에 추가적으로 제품단가를 높이지 않는 장점이 있다[2].

저전력 소프트웨어에 대한 연구로는 시스템에서 CPU가 소모하는 전력이 가장 크다는 점에 착안하여 CPU에서 소모하는 전력을 줄이려는 연구로써 CPU의 구동 주파수와 그에 따른 전압을 조절하여 저전력을 실현하는 동적 전압 조절기법(Dynamic Voltage Scaling)과 동적 전력 관리기법(Dynamic Power Management)이 있으며[3-6], 최근 들어 LCD와 저장장치들의 용량 증가와 이에 따른 소모 전력의 증가를 관리하기 위해 시스템을 구성하는 장치들의 상태에 따라 전원을 관리하는 장치 전력 관리기법(Device Power Management)이 있다[7-10].

본 논문의 구성은, 관련연구에서 저전력 기법을 탑재하기 위한 실시간 운영체제인 UbiFOS™과 본 논문 구현과 관련된 저전력 기법인 동적 전력 관리 기법과 장치 전력 관리 기법에 대해 소개한다. 3장에서는 본 논문에서 제시하고자 하는 실시간 운영체제에서 작업량 관찰에 기반한 저전력 기법의 설계 및 구현 내용에 대해 논하고, 4장에서 그에 따른 실험 결과에 대해 기술한다. 5장에서 결론과 향후연구 과제에 대해 기술한다.

II. 관련연구

1. 실시간 운영체제 UbiFOS™

실시간 운영체제인 UbiFOS™ 은 멀티태스킹 환경을 지원하고, 256단계 우선순위 기반의 선점형 실시간 스케줄러로 동일한 우선순위에 대해 FIFO와 라운드로빈 스케줄링을 제공한다. 또한 태스크에 관련된 태스크 관리기능과 메모리를 동적으로 관리하기 위한 가변 크기 메모리 할당과 고정 크기 메모리 할당 메커니즘을 가지고 있으며, 태스크 간 통신을 위해 메시지 메일 박스(Message Mail Box)와 메시지 큐(Message Queue), 메시지 포트(Message Port), 태스크 포트(Task Port)를, 태스크 간 동기화를 위해 세마포(Semaphore)와 이벤트 플래그(Event Flags)를 제공한다. 또한, 저전력 지원을 위한 동적 전력 관리 기법을 지원한다[5]. [그림 1]은 실시간 운영체제 UbiFOS™의 전체 구성을 나타낸 것이다[11].

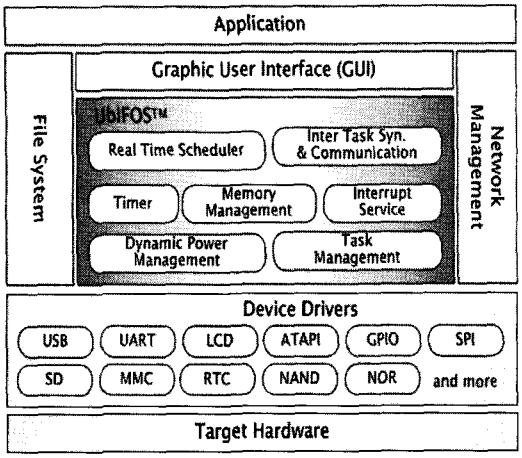


그림 1. 실시간 운영체제 UbiFOS™ 구성도

2. 저 전력 기법

2.1 동적 전력 관리 기법

동적 전력 관리 기법(Dynamic Power Management, DPM)은 IBM 과 MotaVista Software에서 소비전력이 가장 큰 CPU의 frequency와 Bus Clock의 속도를 조절하여 전체 시스템의 소비전력을 감소시키는 전력 관리 기법으로, 시스템에 탑재되는 응용프로그램과 각 디바이스에서 요구하는 Bus Clock 까지 고려하여 소비전력을 감소시키는 융통성을 갖춘 전력 관리 기법이다[5].

시스템 디자이너는 정책 관리자(Policy Manager)를 통해 시스템에서 제공하는 CPU Frequency와 Bus Clock에 맞춰 각 응용프로그램이 요구하는 CPU Frequency와 Bus Clock의 집합을 정의한 정책들(Policies)을 생성한다. 생성된 정책 하에서 정책 관리자는 응용프로그램을 구성하는 각 태스크들에게 태스크가 요구하는 CPU Frequency와 Bus Clock을 할당한다.

비숍 브록과 카딕 라자마니는 CPU 구동 주파수와 전압 조절이 가능한 IBM의 405LP시스템에 동적 전력 관리 기법을 적용한 PowerPC Linux 2.4 운영체제를 탑재하여 MPEG과 MP3 구동 시 26~58%의 소모전력 감축 효과가 있음을 확인하였다[6].

2.2 장치 전력 관리 기법

장치 전력 관리 기법(Device Power Management, DPM)은 시스템의 성능감소 없이 전력 소비를 줄이는 효과적인 접근 방법으로, 서비스요구가 없는 디바이스는 유휴모드(Idle Mode)로 동작하다가, 태스크로부터 디바이스 사용요청이 들어왔을 때 요구한 디바이스를 가동시키는 방식이다. 즉, 프로그램 구동 중에 구현되는 전력관리 알고리즘으로 운영체제나 디바이스 드라이버가 외부 디바이스 중 구동될 필요가 있는 디바이스에 전력을 보내고, 그렇지 않은 디바이스에는 전력을 보내지 않는(Shut Down) 알고리즘이다. 장치 전력 관리 기법을 적용하기 위해서는 태스크가 수행되는 동안 다양한 작업량을 가지며, 작업량의 변동이 예측 가능하다는 가정이 필요하다. 따라서 작업량 관찰을 기반으로 하는 제어정책을 가지고 있다. 가장 간단한 정책으로는 데스크 탑에서 사용되고 있는 시간제한(timeout) 정책이 있다. 이 정책은 일정시간 동안 디바이스 사용 요청이 없는 경우 디바이스를 가동하지 않다가, 서비스 요청이 있을 때 가동되는 전력 관리 기법이다[10].

2.3 동적 전압 조절 기법

동적 전압 조절 기법(Dynamic Voltage Scaling, DVS)은 실시간 시스템에서 요구되는 마감시간이라는 제약 조건(deadline constraint)을 만족시키는 범위 내에서 CPU 구동주파수와 그에 따른 공급 전압을 조절하는 기법이다. 이 기법은 대부분의 실시간 태스크들이 그들의 최악 실행시간(Worst Case Excution Time)전에 수행이 완료된다는 점에서 착안하여 전력 감축을 위한 유휴시간(Slack Time) 계산을 통해 실시간 성을 유지하면서 전압과 주파수를 조절하는 것이다. 실시간 태스크들에 대한 동적 전압 조절 기법 알고리즘들은 태스크들에 대한 전압 및 CPU 클럭 속도가 언제 결정되는지에 따라 크게 오프라인 알고리즘과 온라인 알고리즘으로 구분될 수 있다. 오프라인 알고리즘의 경우, 태스크들이 최악 실행시간을 요구하는 최악 실행 시나리오에 대해 각 태스크에 대한 최적의 공급 전압과 클락 속도를 계산한다.

일반적으로 태스크들의 실제 실행시간은 오프라인에서 분석된 이들의 최악 실행시간보다 적으므로, 온라인 상태에서는 태스크들의 실행시간 변화에 따른 유휴시

간 측정이 필요하다. 오프라인 동적 전압 조절기법에서의 유휴시간은 오프라인 상에서 시스템에 제공되는 시작시간과 데드라인을 근거로 하여 태스크의 수행이 데드라인보다 일찍 완료되어 프로세서가 유휴상태로 있는 시간을 측정하여 알 수 있다.

온라인 동적 전압 조절기법에서의 유휴시간은 태스크의 실제 수행시간을 측정하고 이 시간과 데드라인을 근거로 하여 유휴 시간을 측정한다. 이렇게 측정된 유휴시간은 태스크의 수행시간 이전에 값을 알 수 있으므로 유휴시간을 기준으로 태스크를 구동시키는 CPU의 주파수와 그에 따른 시작 전압 값(starting voltage)을 조절하여 CPU에서 소비되는 전력을 감축시키는 방법이다[1].

동적 전압 조절기법은 공급 전압 조절을 어느 수준에서 하느냐에 따라 크게 태스크 간 전압 조절 기법(InterDVS)과 태스크 내 전압 조절 기법(IntraDVS)로 구분 한다[12].

III. 작업량 관찰에 기반한 저전력 기법의 설계 및 구현

1. 저전력 기법 설계 고려 사항

본 논문의 저전력 기법을 구현하기 위한 고려 사항은 다음과 같다.

- 저전력 기법을 적용하기 위해 기존 실시간 운영체제 커널을 변경해야 하는지 여부
- 하드웨어 시스템에서 구동 주파수에 따라 전압 조절이 가능한지 여부
- 장치 전력 관리 기법 적용을 위해 시스템에서 구동되는 응용프로그램의 작업량에 따른 시간제한(timeout) 값의 크기 설정

본 논문의 저전력 기법을 구현하기 위해 우선순위 기반의 선점형 방식을 사용하는 기존 연성 실시간 운영체제 UbiFOS™의 커널 수정 없이 저전력 기법을 적용할 수 있는지가 가장 큰 고려 대상이었다. 저전력 기법으로 많은 연구들이 진행되고 있는 동적 전압 조절 분야는 경성 실시간 운영체제에 사용되는 스케줄링 기법인 EDF(Earliest Deadline First), RM(Rate Monotonic) 등을 사용하며, CPU 구동 주파수와 그에 따른 전압조절 메커니즘이 하드웨어적인 제약을 고려하지 않은 이상적인(ideal) 상황을 가정하여 연구되어 왔다. 이에 실제 하드웨어적인 제약사항을 고려하여 저전력 기법으로 설계된 동적 전력 관리 기법을 기존 실시간 운영체제에 모듈 형태로 탑재하는 방식을 채택하게 되었다[5][6]. 그러나, 본 논문을 통해 구현한 저전력 기법이 탑재된 실시간 운영체제는 소형 내장형 이동 시스템(embedded mobile system)을 목표로 개발된 것으로 이런 내장형 시스템은 대부분 단일 태스크에서 구동되는 하나의 응용프로그램 모듈로 구성되는 것을 고려하면, 태스크 간에서 구동정책에 따라 태스크에 CPU와 Bus Clock을 할당하는 기존의 동적 전력 관리 기법을 적용하기에는 걸림돌이 되며, 다중 태스크 환경에서도 만약 하나의 태스크가 전체 실행 시간의 대부분을 차지하는 경우에는 태스크 간 동적 전력 관리를 하는 저전력 기법으로는 소모 전력을 감축시키는 데에 한계가 있다. 또한, 근래에 들어 내장형 이동 시스템이 사용자 요구에 따라 응용프로그램의 복잡도가 증가하면서 CPU성능과 메모리 용량, LCD의 크기 증가 등 시스템에서 소모하는 전력의 대부분을 차지했던 디바이스 장치들이 점점 커지면서 이에 따른 전체 시스템의 소비전력 또한 증가하고 있다. 이로 인해 각 디바이스 장치들에서 소비하는 전력을 감축시키는 장치 전력 관리 기법의 적용이 중요 이슈로 고려되었다. 이를 위해 작업량 관찰을 위한 시간제한(timeout) 정책과 예측(predictive) 정책, 확률 모델(stochastic modeling) 정책 중 구현이 용이한 시간제한(timeout) 정책을 고려하였다[10].

본 논문에서는 기존의 태스크 간 동적 전력 관리기법을 태스크 내 동적 전력 관리기법으로 변경하여 실시간 운영체제 UbiFOS™에 모듈형태로 탑재하였으며, 내장형 이동 시스템에 탑재되는 각 디바이스들의 소모 전력을 감축시킬 수 있는 장치 전력 관리기법 또한 모듈 형태로 추가하였다.

2. 실시간 운영체제 UbiFOS™에서 태스크 내 동적 전력 관리 기법(Intra Dynamic Power Management)의 구현

본 논문에서 작업량 관찰에 기반한 저전력 관리 기법을 실시간 운영체제 UbiFOS™에 구현하기 위해 태스크 간 동적 전력 관리 기법을 태스크 내 동적 전력 관리 기법으로 변경하여 탑재하였으며, 배터리로 구동되는 내장형 이동 시스템을 고려하여 시간제한(timeout) 정책에 따라 작업량을 구분하고, 배터리 잔량에 따라 [표 1]과 같이 태스크 내 동적 전력 관리를 위한 구동전략과 그에 따른 구동정책을 설계 및 구현하였다.

[표 1]에서 Highest, High, Middle, Low, Lowest는 CPU 구동 주파수의 속도를 기준으로 구분한 것으로 단위는 MHz이다.

표 1. IntraDPM 구동 전략과 정책

구동정책 / 구동전략	Workload High	Workload Low	Workload suspend
Battery High	Highest	High	Middle
Battery Low	High	Middle	Low
Battery Critical	Middle	Low	Lowest

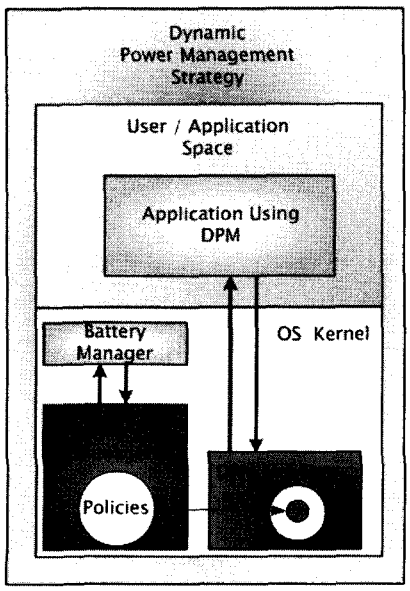


그림 2. 동적 전력 관리 구조모델

[그림 2]는 커널에서 제공하는 태스크 내 동적 전력 관리의 API를 통해 구동전략을 결정하고, 각 응용프로그램에 미리 정의된 구동정책을 할당하여 수행하는 동적 전력 관리 구조모델이다.

[그림 3]의 동적 전력 관리자는 응용프로그램 구동을 위해 요구되는 작업량과 현재 시스템의 배터리 잔량을 확인하여 응용프로그램 내에서 구동될 CPU 구동 주파수와 그에 따른 전압을 조절한다.

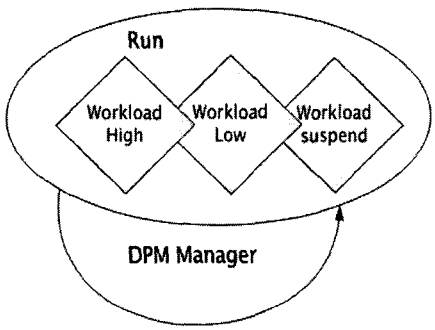


그림 3. 태스크 내 동적 전력 관리

표 2. 태스크 내 동적 전력 관리 관련 API

함 수	설 명
IntraDPM_Init()	태스크 내 동적 전력 조절 관련 초기화 함수
IntraDPM_Set_Policies()	구동전략 설정 함수
IntraDPM_Get_Policies()	구동전략 얻어오는 함수
IntraDPM_Frequency_Set()	구동전략에 따라 구동주파수와 전압을 조절하는 함수
BatteryLevel_Check()	배터리량을 얻어오는 함수

[표 2]는 본 논문에서 구현한 태스크 내 전력 관리 기법과 관련된 저전력 API로 응용프로그램 프로그래머는 구동정책을 설정하는 IntraDPM_Set_Policies() 함수를 통해 구동정책을 설정 및 변경할 수 있으며, IntraDPM_Get_Policies() 함수를 통해 현재 설정된 구동정책을 얻어 올 수 있다. 한편, 커널 내부의 태스크 내 동적 전력 관리자는 설정된 구동정책 하에서 BatterLevel_Check() 함수를 통해 배터리 잔량을 확인하고 시간제한(timeout) 정책에 따라 IntraDPM_Frequency _Set() 함수를 통해 CPU 구동

주파수와 전압을 조절한다.

3. 실시간 운영체제 UbiFOS™에서 장치 전력 관리 기법(Device Power Management)의 구현

본 논문의 저전력 기법을 적용하기 위해 내장형 이동 시스템에서 소모 전력이 큰 디바이스인 CPU와 RAM, LCD 등을 시간제한(timeout) 정책에 따라 작업량을 구분하고, shutdown 정책을 적용하여 소모 전력을 감축시키는 장치 전력 관리 기법을 구현하였다.

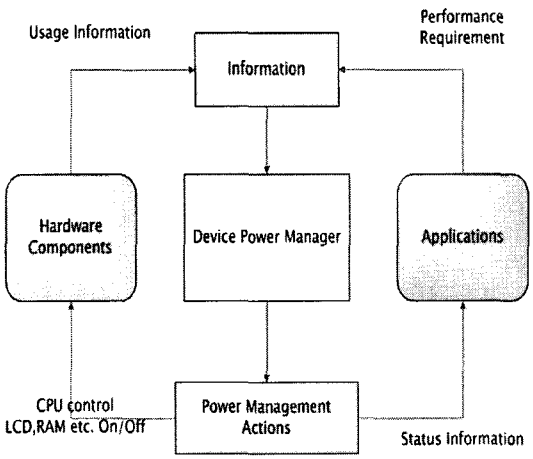


그림 4. 장치 전력 관리 구조모델

[그림 4]는 장치 전력 관리 구조모델로 장치 전력 관리자(Device Power Manager)는 각 디바이스에 따라 시간제한(timeout)을 두어 시스템 컴포넌트를 On/Off 하여 소비전력을 감소시킨다.

표 3. 장치 전력 관리 관련 API

함 수	설 명
DevPM_Init()	장치 전력 관리 관련 초기화 함수
DevPM_Register()	전력 관리자에 장치 등록 함수
DevPM_Unregister()	전력 관리자에 장치 해제 함수
DevPM_Suspend()	장치 전원 Off함수
DevPM_Wakeup()	장치 전원 On함수

표 4. LCD 장치 제어 관련 API

함 수	설 명
LCD_Init()	LCD 장치 초기화 함수
LCD_Suspend()	LCD 장치 Off 함수
LCD_Wakeup()	LCD 장치 On 함수
LCD_Set_Brightness()	LCD 후면 밝기 조절 함수

[표 3]은 장치 전력 관리 관련 API 함수이고, [표 4]는 LCD 장치를 제어하는 API 함수이다. 커널 내부에서 시스템 초기화 시에 DevPM_Init()함수를 통해 장치 전력 관리자를 초기화하고 DevPM_Register() 함수를 통해 전원 관리가 필요한 장치들을 등록한다. LCD 장치를 예로 들어 설명하면, 함수 내부에서 LCD장치를 등록하고 등록 시에 실제 장치들을 On/Off하는 LCD_Suspend() 함수와 LCD_Wakeup() 함수를 DevPM_Suspend() 함수와 DevPM_Wakeup() 함수에 등록한다.

장치 전력 관리자는 등록된 장치에 대한 시간제한(timeout) 정책에 따라 각 장치들을 제어하는 함수를 통해 On/Off 시킨다. 그 외 DevPM_Unregister() 함수는 장치 전력 관리자에 등록된 장치를 해제하는 함수이고, LCD_Set_Brightness() 함수는 LCD의 후면 밝기를 조절하는 함수이다.

4. 작업량 관찰에 기반한 저전력 기법의 구현

본 논문에서 구현하는 작업량에 기반한 저전력 기법은 작업량의 구분을 위한 시간제한(timeout) 정책 하에서 태스크 내 동적 전력 관리 기법과 장치 전력 관리 기법을 적용하여 설계 및 구현하였다.

[표 5]는 시간제한(timeout) 정책과 관련된 API로, Timeout_Init() 함수는 시스템에서 사용될 timeout 값을 디폴트(default) 값으로 설정하는 함수이다. 응용프로그래머는 응용프로그램 구동 시 Timeout_Set() 함수를 통해 시간제한(timeout) 값을 설정 및 변경할 수 있으며, Timeout_Get() 함수를 통해 값을 얻어 올 수 있다.

표 5. 시간제한(timeout) 제어 관련 API

함 수	설 명
Timeout_Init()	timeout 값 초기화 함수
Timeout_Set()	timeout 값 설정 함수
Timeout_Get()	timeout 값 얻어오는 함수

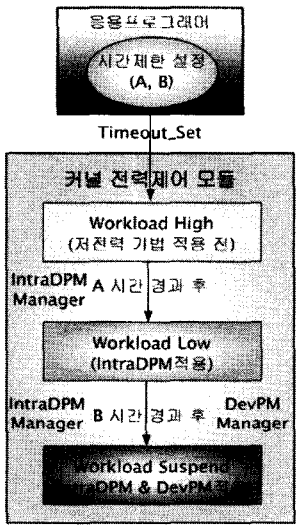


그림 5. 작업량에 기반한 저전력 기법

[그림 5]는 본 논문에서 구현한 작업량에 기반한 저전력 기법을 도시한 것이다.

저전력 기법의 수행은 다음과 같다. 우선 응용프로그래머는 태스크 내 동적 전력 관리를 위한 구동 전략과 구동 정책에 따른 CPU 구동 주파수와 전압을 설정하고, 전력 관리가 필요한 장치를 장치 전력 관리자에 등록하며, Timeout_Set() 함수를 통해 시간제한 기법을 위한 A, B시간을 설정한다.

실행 시나리오는 다음과 같다.

하나의 태스크 내에서 수행되는 응용프로그램 내부에서 시스템 초기화 시에는 Workload High 구동 정책으로 수행하고, A라는 시간이 경과한 후에 사용자 입력과 같은 특정 작업 요청이 없을 경우 태스크 내 동적 전력 관리자는 시스템 초기의 구동 전략과 Workload Low 구동 정책에 설정한 CPU 구동 주파수와 전압으로 조절한다. 만약 그 시간 내에 작업 요청이 있을 경우는 Workload High 상태로 환원하고, 특정 작업 요청이 없고 추가적으로 B라는 시간이 경과한 후에도 특정 작업 요청이 없는 경우 Workload Suspend 구동 정책에 따라 CPU 구동 주파수와 전압을 조절하며, 장치 전력 관리자는 해당 장치에 대한 전원을 오프(Off)한다.

Ⅳ. 실험 환경 및 결과

본 논문에서 구현한 작업량에 기반한 저전력 기법의 소모 전력 감축 비율을 측정하기 위해 ARM920T 기반의 MBA2440(S3C2440)[11] 보드에 J2ME(Java 2 Platform Micro Edition)플랫폼의 CLDC(Connected Limited Device Configuration)와 MIDP(Mobile Informaiton Device Profile) 기반의 KVM(Kilobyte Virtual Machine)[13-15] 상에서 구동되는 푸쉬푸쉬(PushPush) 게임 응용프로그램을 실험 대상으로 하였다. 그 이유는 내장형 이동 시스템의 한 예인 휴대폰에서는 VM(Virtual Machine)을 응용프로그램 구동을 위해 많이 사용하고 있으며, 휴대폰에서 사용되는 VM은 대부분이 하나의 태스크 위에 하나의 응용프로그램이 동작하는 형태로 구성되기 때문이다. 또한, 내장형 이동 시스템에서 전력 소모가 큰 CPU와 LCD 디바이스를 대상으로 본 논문에서 구현한 저전력 기법을 적용하여 실험하였다.

그러나 MBA2440보드는 평가보드(Evaluation Board)로써 배터리로 동작하지 않고 전압 조절이 불가능하다는 제약사항을 고려하여 [표 6]과 같은 구동전략과 정책을 구성하여 실험을 수행하였다. 또한, 실험한 MBA2440 보드는 ARM920T 기반의 S3C2440MCU와 32MB SDRAM, 320 * 240 크기의 LCD, 64MB의 NandFlash, 16MB NorFlash로 구성되어 있다. CPU의 구동 주파수는 399MHz ~ 96MHz까지 조절이 가능하지만 그에 따른 전압 조절은 불가능하며(5V고정전압), 각 디바이스 장치들의 On/Off가 가능하다. 그러나 LCD의 후면 밝기 조절은 불가능하다.

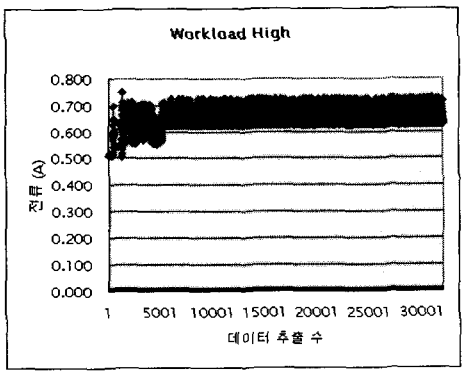
실험을 위한 시간제한 값은 푸쉬푸쉬 게임이 시작되고 1초 동안 키 입력이 없을 경우 Workload Low로, 그 이후 3초 동안 키 입력이 없을 경우 Workload Suspend

로 동작하도록 구성하였다.

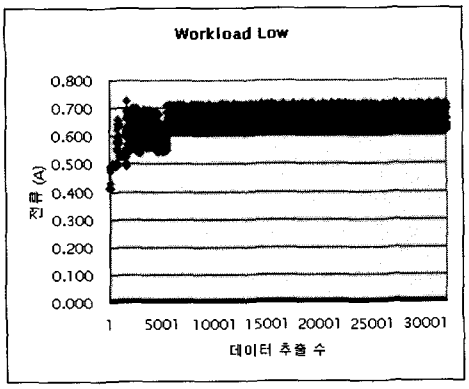
소모 전력에 대한 측정은 에이질런트사의 34411A 디지털 멀티미터를 사용하여 일정 시간 간격으로 30000회 이상의 순간 전류 변화 자료를 수집하여 평균값을 계산하는 방식을 사용하였다. 고정 전압을 갖는 시스템에서 소모되는 전력은 각 순간에 흐르는 전류량에 비례하여 증가하기 때문에 본 논문에서 구현한 저전력 기법을 적용하였을 경우 소모 전력 감축 비율을 계산할 수 있다.

표 6. 저전력 기법 실험을 위한 구동 전략과 정책

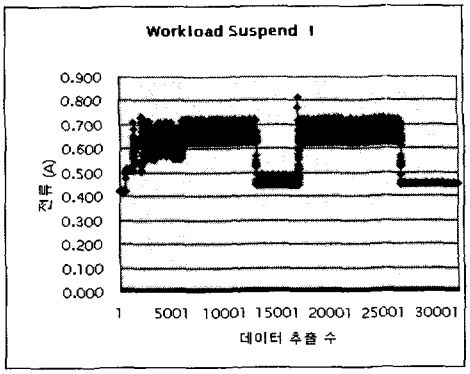
구 분	Workload High	Workload Low	Workload Suspend
작업량 시간제한(timeout)	초기상태	1초	3초
구동 전략	399MHz LCD On	266MHz LCD On	96MHz LCD off



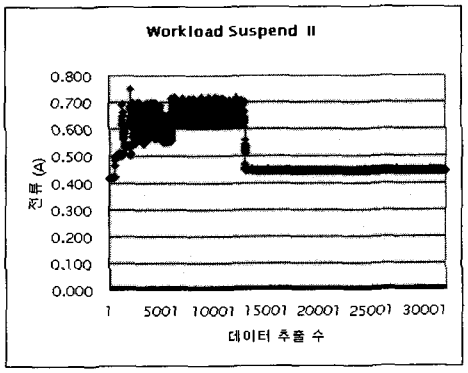
(a)



(b)



(c)



(d)

그림 6. 구동 전략과 정책에 따른 실험 결과

[그림 6]은 본 논문에서 구현한 저전력 기법을 적용하여 실험을 수행한 결과이다.

Workload High(a)와 Workload Low(b)의 경우는 CPU 구동 주파수만 변경한 것으로, 399MHz에서 266MHz로 변경했을 경우 실제 시스템에 흐르는 순간 전류의 변화가 미미한 것을 확인할 수 있다. Workload suspend의 경우 96MHz의 CPU구동 주파수와 LCD off로 인한 순간 전류의 변화를 확인할 수 있다. Suspend I (c) 그림에서 순간 전류가 높아진 곳은(16926부분) 키 입력을 가한 부분으로 Workload High상태로 환원되면서 LCD가 On되는 순간이다.

표 7. 구동 전략에 따른 평균 소모 전류 / 전력 감축율

구 분	Workload High	Workload Low	Workload Suspend
평균 소모 전류	0.636	0.624	0.528 / 0.486
전력 감축률	0%	2%	17% / 24%

[표 7]은 시간제한(timeout) 정책과 그에 기반한 구동 전략과 구동 정책 하에서 소모되는 평균전류 값을 나타낸다. 작업량이 많은 경우(Workload High) 즉, 시스템이 유휴 상태 없이 계속적인 작업요청이 있을 경우 소모되는 평균 전류는 0.636이고, 시간제한(timeout) 정책에 의해 1초 동안 작업량이 없을 경우(Workload Low) 평균 소모 전류는 0.624로 2%의 소모 전력이 감축되었으며, 그 이후 3초가 지난이후에도 작업량이 없을 경우(Workload Suspend) 소모되는 평균 전류는 0.528과 0.486으로 각각 17%(c)와 24%(d)의 소모 전력이 감축되었다.

0.528인 경우는 Workload Suspend상태에서 특정 시점에 작업요청이 있어 Workload High상태로 천이한 경우이고 0.486인 경우는 특정 시점에 작업요청이 없는 경우이다. 이를 통해 짧은 시간(1초) 동안 작업 요청이 없을 경우는 소모 전력 감축효과(2%)가 미미한 반면, 그 보다 긴 시간 동안 작업 요청이 없을 경우에는 소모 전력 감축효과(24%)를 확인할 수 있으며, 만약 3초보다 긴 시간동안 작업 요청이 없을 경우에는 그 이상의 소모 전력 감축효과를 기대할 수 있다.

V. 결 론

본 논문에서 복잡한 기능을 가진 내장형 이동 시스템의 소모 전력 증가에 따른 문제를 소프트웨어적인 방법으로 해결하기 위한 동적 전력 관리, 장치 전력 관리, 동적 전압 조절과 같은 저전력 기법을 소개하였으며, 그 중 실시간 운영체제의 커널 수정 없이 적용할 수 있는 동적 전력 관리 기법과 장치 전력 관리 기법을 융합하여 응용프로그램에서 직접 적용할 수 있는 작업량 관찰에 기반한 저전력 기법을 설계 및 구현하였다.

본 논문의 저전력 기법을 실험하기 위해 실시간 운영체제인 UbiFOS™에 모듈 형태로 추가하고 MBA2440 하드웨어 플랫폼에 탑재하여 내장형 이동 시스템의 응용프로그램의 하나인 게임 응용프로그램을 통해 검증하였다. 작업량에 따라 유동적일 수 있지만, 대략 24% 이상의 소비전력 감축효과를 확인할 수 있었으며, 실시간 운영체제의 커널 수정 없이 모듈형태로 추가하여 저전력 구현이 가능함을 보였다.

향후 연구 과제는 전압 조절 및 후면 밝기 조절이 가능한 플랫폼에 본 논문의 저전력 기법을 탑재하여 실험하고 동적 후면 밝기 조절 기법(Dynamic Backlight Luminance Scaling)을 적용하여 소모 전력 감축에 대해 연구하는 것이다.

참 고 문 헌

[1] T. S. Marcus, M. Bashir, and A. Hashimi, *System-Level Design Techniques for Energy-Efficient Embedded Systems*, Kluwer academic publishers, Boston, 2004.

[2] 장래혁, "저전력 시스템과 저전력 소프트웨어", 한국통신학회지 (정보통신), 제18권, 제12호, pp.59-71, 2001.

[3] K. Govil, E. Chan, and H. Wasserman, "Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU," in Proc. ACM Int'l Conf. on Mobile Computing and Networking, pp.13-25, 1995.

[4] T. Pering, T. Burd, and R. Brodersen, "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms," in International Symposium on Low Power Electronics and Design (ISLPED), pp.76-81, 1998.

[5] http://www.research.ibm.com /arl/projects/dpm.html

[6] B. Brock and K. Rajamani, "Dynamic Power Management for Embedded Systems," Proc. of IEEE Int'l SoC Conf. (SoCC 2003), pp.416-419, Sep. 2003.

[7] A. Karlin, et al., "Competitive randomized nonuniform problems," Algorithmica, Vol.11, No.6, pp.130-142, June 1994.

[8] R. Golding, P. Bosch, and J. Wilkes, "Iddleness is not Sloth," USENIX Winter Conference, pp.201-212, 1995.

[9] E. Y. Chung, L. Benini, and G. D. Micheli, "Dynamic power management using adaptive learning tree," Proceedings of the International Conference on Computer Aided Design, pp.274-279, 1999.

[10] Wang Yue, Zhao Xia, and Chen Xiangqun, "A Task-Specific Approach to Dynamic Device Power Management for Embedded System," in ICESS'05, Vol.00, pp.158-165, 2005.

[11] http://www.aijisystem.com..

[12] 신동군, 김지홍, "실행 시간 프로파일을 이용한 저전력 경성 실시간 프로그램용 동적 전압 조절 알고리즘", 한국정보과학회논문지 (시스템 및 이론), 제29권, 제11권, 제12호, pp.601-610, 2002.

[13] Kim Topley, *J2ME In a nutshell, O'Reilly & Associates*, 1st Edition, 2002.

[14] Sun Microsystems, *JSR-139, Connected Limited Device Configuration(CLDC) Specification*, Version 1.1, 2003.

[15] Sun Microsystems, *JSR-118, Mobile Information Device Profile Specification*, Version 2.0, 2002.

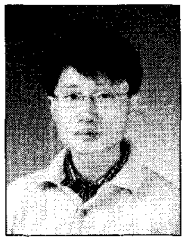
저 자 소 개

조 문 행(Moon-Haeng Cho) 정회원

- 2004년 2월 : 충남대학교 컴퓨터공학과 (공학사)
- 2006년 2월 : 충남대학교 컴퓨터공학과 (공학석사)
- 2006년 3월 ~ 현재 : 충남대학교 컴퓨터공학과 박사과정 재학

<관심분야> : 실시간 컴퓨팅, 실시간 운영체제, 초소형 초절전 실시간 운영체제

정 명 조(Myoung-Jo Jung) 정회원



- 2003년 2월 : 충남대학교 컴퓨터공학과 (공학석사)
- 2003년 3월 ~ 현재 : 충남대학교 컴퓨터공학과 박사과정 재학

<관심분야> : 실시간 컴퓨팅, 실시간 운영체제, 초소형 초절전 실시간 운영체제, 자바가상머신

김 용 희(Yong-Hee Kim) 정회원

- 2003년 2월 : 충남대학교 컴퓨터공학과 (공학석사)
- 2003년 3월 ~ 현재 : 충남대학교 컴퓨터공학과 박사과정 재학

<관심분야> : 실시간 시스템, 실시간 운영체제, 고장허용 컴퓨팅

이 철 훈(Cheol-Hoon Lee) 정회원

- 1983년 2월 : 서울대학교 전자공학과 (공학사)
- 1988년 2월 : 한국과학기술원 전기및전자공학과 (공학석사)
- 1992년 2월 : 한국과학기술원 전기및전자공학과 (공학박사)
- 1983년 3월 ~ 1986년 2월 : 삼성전자 컴퓨터사업부 연구원
- 1992년 3월 ~ 1994년 2월 : 삼성전자 컴퓨터사업부 선임연구원
- 1994년 2월 ~ 1995년 2월 : Univ. of Michigan 객원연구원
- 1995년 2월 ~ 현재 : 충남대학교 컴퓨터공학과 교수
- 2004년 2월 ~ 2005년 2월 : Univ. of Michigan 초빙연구원

<관심분야> : 실시간시스템, 운영체제, 고장허용 컴퓨팅