웨어러블 컴퓨터를 위한 저전력 실시간 운영체제 eRTOS 설계 및 구현

Design and Implementation of eRTOS Real-time Operating Systems for Wearable Computers

조문행, 최찬우, 이철훈
충남대학교 컴퓨터공학과

Moon-Haeng Cho(root4567@cnu.ac.kr), Chan-Woo Choi(cwchoi00@cnu.ac.kr),
Cheol-Hoon Lee(clee@cnu.ac.kr)

요약

오늘날의 내장형 시스템은 군사 무기체계, 로봇, 인공위성 등과 같이 전통적인 내장형 시스템과 휴대폰, PMP(Portable Multimedia Player), PDAs(Personal Digital Assistants)와 같이 통신과 멀티미디어 기기가 결합된 디지털 컨버전스 시스템에서 먹는 PC, 웨어러블 컴퓨터와 같은 차세대 PC 개념으로 진화하고 있다. 차세대 PC는 문서작성・인터넷 검색・데이터 관리 등에서 사용되었던 기존의 PC에서 분기된 네트워크 기반의 인간중심 디지털 정보기기이다. 웨어러블 컴퓨터는 극히 전력과 메모리 제한적인 시스템으로, 구성 하드웨어의 제약 사항을 극복하고 사용자 서비스의 QoS를 제공하기 위해 초소형이면서 저전력 기능을 갖춘 실시간 운영체제를 사용해야만 한다. 본 논문에서는 웨어러블 컴퓨터를 위한 저전력 실시간 운영체제 eRTOS를 설계 및 구현하였다. 본 논문에서 구현한 eRTOS는 18KB의 풋프린트(footprint)로 동적 전력 관리 기법(Dynamic Power Management)과 장치 전력 관리 기법(Device Power Management)의 저전력 기법이 구현되어 있다. 웨어러블 컴퓨터의 응용프로그램을 실험하여 47%의 전력 소모 감축효과를 확인하였다.

■ 중심어 : | 차세대 PC | 웨어러블 컴퓨터 | 저전력 | 실시간 운영체제 |

Abstract

In recent years, embedded systems have been expanding their application domains from traditional embedded systems such as military weapons, robots, satellites and digital convergence systems such as celluar phones, PMP(Portable Multimedia Player), PDAs(Personal Digital Assistants) to Next Generation Personal Computers(NGPCs) such as eating PCs, wearable computers. The NGPCs are network-based, human-centric digital information devices diverged from the traditional PCs used mainly for document writing, internet searching and database management. Wearable computers with battery capacity and memory size limitations have to use real-time operating systems with small footprints and low power management techniques to provide user's QoS in spite of hardware constraints. In this paper, we have designed and implemented a low-power RTOS (called eRTOS) for wearable computers. The implemented eRTOS has 18KB footprints and the dynamic power management and the device power management schemes are adapted in it. Experimental results with wearable computer applications show that the low power techniques could save energy up to 47 %.

■ keyword : | Next Generation PCs | Wearable Computers | Low Power | Real-time Operating Systems |

* 본 연구는 지식경제부의 지원으로 수행되었습니다.

접수번호 : #080520-005
접수일자 : 2008년 05월 20일
심사완료일 : 2008년 09월 05일
교신저자 : 이철훈, e-mail : clee@cnu.ac.kr

I. 서 론

과거 내장형 시스템은 군사 무기체계, 인공위성, 원자력 발전소의 제어시스템, 공장 자동화 로봇과 같은 시스템이 주를 이루었지만, 최근에는 휴대폰, PMP(Portable Multimedia Player), Navigation, 휴대용 게임기, DMB(Digital Multimedia Broadcasting) 등의 디지털 컨버전스 기기와 같은 내장형 이동 시스템으로 발전하였다. 또한, 멀지 않은 미래에는 U-city, U-health와 같은 유비쿼터스 컴퓨팅의 일반화로 의류 등 전통산업과 IT산업이 접목된 웨어러블 컴퓨터와 같은 차세대 PC 사용이 대세가 될 것이다.

차세대PC, 차세대 컴퓨팅의 한 분야인 웨어러블 컴퓨터는 개발초기에 전자전 장비 같은 군사용으로 시장이 형성되었지만, 최근에는 비행기 엔진의 유지·보수 같이 산업 현장뿐만 아니라 액세서리형 MP3 Player, 시계 타입의 스마트폰과 같이 일반 대중 용도로 많은 응용 제품들이 개발되고 있다[1][2].

이런 웨어러블 컴퓨터는 가격 경쟁력과 함께 휴대용이라는 점에서 제한적인 하드웨어 자원을 탑재한다. 제한된 CPU 구동주파수와 메모리(RAM) 자원을 효율적으로 사용· 관리해야 하며, 특히, 작은 메모리 크기로 인하여 커널 풋프린트(Kernel Footprint)가 큰 Linux나 Windows와 같은 범용 운영체제 보다는 풋프린트가 작은 실시간 운영체제를 탑재한다. 또한, 휴대용이라는 점에서 내장된 배터리 자원을 응용 프로그램의 구동에 따라 효율적으로 제어할 수 있는 저전력 기법이 필수적이다.

본 논문에서는 이처럼 하드웨어 제약사항을 갖는 차세대 PC인 웨어러블 컴퓨터에 탑재되어 실시간 성이 요구되는 응용 프로그램의 안정적 구동을 보장하고, 소모 전력을 관리하여 보다 긴 시스템 구동시간을 보장하기 위한 초소형의 저전력 기능을 갖춘 실시간 운영체제인 eRTOS에 대해 기술한다.

본 논문의 구성은, 관련연구에서 차세대 PC의 한축인 웨어러블 컴퓨팅에 대해 기술하고, 근래의 내장형 시스템에 많이 탑재되고 있는 Windows Embedded CE, Embedded Linux와 웨어러블 컴퓨터의 제한적인 하드웨어 자원을 효율적으로 관리할 수 있는 실시간 운영체제에 대해 소개한 후, 웨어러블 컴퓨터와 같이 배터리로 구동되는 시스템을 위해 그동안 연구되어온 저전력 기술에 대해 기술한다. 3장에서는 본 논문에서 구현한 초소형 저전력 실시간 운영체제인 eRTOS의 설계 및 구현 내용에 대해 논하고, 4장에서 구현한 eRTOS의 기능 실험 결과와 실시간 운영체제 성능 평가의 잣대인 스케줄링과 태스크간 통신 오버헤드를 측정한 값으로, 10.5μs의 스케줄링 오버헤드와 58μs의 태스크간 통신 오버헤드의 실험 결과를 소개한다. 또한, 저전력 기법 적용에 따라 47%의 소모 전력 감축을 확인하였다. 마지막으로, 5장에서 결론과 향후연구 과제에 대해 기술한다.

II. 관련연구

본 절에서는 차세대 PC인 웨어러블 컴퓨터에 대해 소개하고, 컴퓨터의 하드웨어 자원을 관리하는 내장형 운영체제에 대해 기술한다. 또한, 웨어러블 컴퓨터를 비롯해 기존의 내장형 이동 시스템의 배터리 자원을 효율적으로 관리하기 위한 저전력 기법들에 대해 소개한다.

1. 웨어러블 컴퓨터

유비쿼터스 컴퓨팅은 일상생활에서 매일 사용하는 시계, 옷, 신발 등과 같은 물건들에 프로세서와 무선통신 인터페이스, S/W 등이 탑재되어 사람이 원하는 것을 사람의 해석이나 간섭 없이 처리하고 제공하는 차세대 컴퓨팅 시스템이다. 이 유비쿼터스 컴퓨팅은 일상생활 속의 사물을 대상으로 하는 내장형 컴퓨팅과 사람의 신체를 대상으로 하는 웨어러블 컴퓨팅, 주변의 환경을 대상으로 하는 환경지각 컴퓨팅 등으로 구분할 수 있다[1].

웨어러블 컴퓨터와 같은 차세대 PC는 착용성, 저전력, 소형화 기술에 의한 스마트 웨어 등과 같은 플랫폼 분야와 재래식 키보드, 마우스, 모니터를 대체할 초소형 키보드 및 디스플레이 등을 포함하여 양손의 사용을 자유롭게 하는 입출력 장치와 음성, 시각, 촉각, 후각, 미각 등 오감 정보처리 기술을 위한 차세대 사용자 인터

페이스, 그리고 데이터 송수신을 위한 BAN(Body Area Network), WPAN(Wireless Personal Area Network) 기술 등을 주요 대상으로 한다. 또한, 차세대 PC는 기술의 융합화와 정보기기의 소형, 경량화 추세로 차세대 PC의 진화방향과 발전요인에 따라 PDA, 전자지갑 등 지니고 다니는 전자비서 형태에서 손목시계와 같은 액세서리형, 신체 착용형 입는 컴퓨터, 향후 신체 내장형 컴퓨터인 먹는 컴퓨터까지 나아가 차세대 PC는 종래의 컴퓨터에서 보다 진화된 형태의 미래형 컴퓨터의 모습을 가질 것이다[2].

앞으로 입는 컴퓨터는 의복이나 액세서리 개념의 컴퓨터에서 인간의 오감 메커니즘을 모방하여 자연스럽고 편리하게 컴퓨터와 때와 장소의 제한없이 대화할 수 있도록 기기와 인간사이의 상호작용을 개선시키는 인간 중심의 휴먼 인터페이스 기술로 발전될 것이다.

2. 내장형 운영체제

내장형 시스템에는 정보가전 시장과 여타의 내장형 응용 산업을 주도하던 전통 내장형 실시간 운영체제인 VxWorks, pSOS, QNX, Nuclues, µC/OS 등과 하드웨어의 양적, 질적 발전으로 CPU 구동속도와 RAM 용량이 늘어남에 따라 내장형 응용이 데스크탑화 되면서 Windows Embedded CE와 임베디드 리눅스(Embedded Linux)와 같은 기존의 데스크탑 운영체제를 내장형 시스템에 탑재될 수 있도록 개조한 운영체제들이 탑재되고 있다[3].

2.1 실시간 운영체제

실시간 운영체제는 그 실행환경의 특성상 MS-DOS Windows, Unix, Linux 등과 같이 컴퓨터에 쓰이는 범용 운영체제와는 달리, 태스크 수행의 데드라인(Deadline)을 초과하지 않도록 시간결정성(determinism)을 보장하는 안정된 스케줄링 기능을 갖춘 운영 체제이다. 또한 실시간 운영체제는 범용 운영체제에 비해 실행 크기가 작아서 주로 내장형 시스템(Embedded System)에 탑재되어 사용된다. 실시간 운영체제는 EDF(Earliest Deadline Frist), RM(Rate Monotonic) 기반의 경성 실시간 운영체제와 우선순위 기반의 연성 실시간 운영체제로 구분한다. VxWorks, pSOS, QNX, Nuclues, µC/OS 등 대부분의 상용 실시간 운영체제는 연성 기법을 사용한다[4-8].

2.2 Windows Embedded CE & 임베디드 리눅스

Windows Embedded CE는 기존의 데스크 톱 사용자들에게 이미 익숙해진 윈도 인터페이스를 내장형 환경에서도 동일하게 사용할 수 있다는 점과 임베디드 리눅스와 비교하여 기술적인 측면에서의 우위로 임베디드 리눅스가 적용되는 다양한 분야에서 사용하고 있다. 하지만, 임베디드 리눅스나 일부 상용 RTOS에 비해 상대적으로 로얄티가 비싸다는 단점이 있다[9].

임베디드 리눅스는 특정 응용프로그램에 맞도록 리눅스 커널의 크기와 성능을 최소, 최적화 시켜 만들어낸 커널로, 검증된 견고성과 보안 기술, POSIX 호환과 GNU 개발환경지원, 광범위한 응용과 하드웨어 지원, 다양한 종류의 네트워킹 프로토콜과 파일 시스템을 지원, 공개소스, 낮은 로얄티 가격 등의 장점으로 기존 상용 RTOS 시장인 백색가전, 셋톱박스와 같은 정보가전 분야와 휴대폰, 스마트폰과 같은 휴대 통신 분야, 라우터 장비와 같은 네트워킹 분야 등 다양한 분야에서 사용하고 있다[10].

3. 저전력 기법

3.1 동적 전력 관리 기법

동적 전력 관리 기법(Dynamic Power Management, DPM)은 IBM 과 MontaVista 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[11].

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 태스크

가 요구하는 CPU Frequency와 Bus Clock을 할당하여, 응용프로그램 구동의 QoS를 보장하면서 소모전력을 줄이는 기법이다[11].

3.2 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[12].

3.3 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 점에 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 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 조절 기법에서의 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[13-16].

Ⅲ. 초소형 저전력 실시간 운영체제 - eRTOS

1. eRTOS 설계 고려 사항

본 논문에서 타켓으로 하고 있는 하드웨어 플랫폼은 시계타입의 웨어러블 컴퓨터로써, 프리스케일사의 i.MX21 프로세서(266MHz)와 1MByte의 SDRAM, OLED 디스플레이 장치가 탑재된 WPGB(Wearable Personal Gateway Board)라 명명된 플랫폼이다.

WPGB와 같은 웨어러블 컴퓨터는 제한된 메모리 공간에 운영체제와 응용프로그램이 차지하는 메모리에 대해서 고려해야 하기 때문에 Windows Embedded CE나 임베디드 리눅스와 같은 내장형 운영체제와 VxWorks, QNX, Nucleus와 같이 수십 ~ 수백KB의 실시간 운영체제를 사용하기에는 적합하지 않다. 한편, 대략 12KB의 µC/OS 실시간 운영체제는 메모리 공간만 고려하면 탑재가 가능하지만 저전력 기법을 제공하지 않기 때문에 웨어러블 컴퓨터에는 적합하지 않다.

또한, 제한적인 CPU성능도 고려하여야 한다. 최신 버전의 Windows Embedded CE나 임베디드 리눅스는 실시간 성을 지원하지만, 실시간 운영체제 보다는 스케줄링 타임과 태스크간 통신, 인터럽트 지연시간 등 성능면에 있어서 제약사항을 갖는다. 또한, VxWorks, pSOS, QNX, Nuclues, µC/OS와 같은 실시간 운영체제는 CPU 성능만 고려하면 웨어러블 컴퓨터에 적합하지만, WPGB가 배터리로 동작한다는 점을 고려하면 웨어

러블 컴퓨터에는 적합하지 않다. 이는 기존 상용 실시간 운영체제들이 저전력 기법을 제공하지 않기 때문이다.

본 논문에서는 웨어러블 컴퓨터의 비교적 낮은 CPU 성능과 작은 메모리 용량, 휴대용의 배터리로 구동된다는 점 등을 고려하여, 초소형 저전력 실시간 운영체제인 eRTOS를 설계 및 구현하였다.

2. eRTOS 구현

본 논문의 eRTOS는 WPGB의 하드웨어 제약사항을 극복하고 WPGB 응용 프로그램 구동의 QoS를 보장하는 실시간 성 지원과 저전력 기능, 초소형 풋프린트를 고려하여 20KB 이내의 연성 실시간 운영체제 개발을 목표로 구현 되었다.

[그림 1]의 eRTOS는 WPGB응용에 맞추어 28단계의 우선순위를 관리하는 실시간 스케줄러와 태스크 관리, 가변길이 메모리 관리, 메시지 포트와 태스크 포트 기능의 태스크간 통신 메커니즘, 인터럽트 서비스루틴, 저전력 기법인 동적 및 장치 전력 관리로 구성되어 있다.

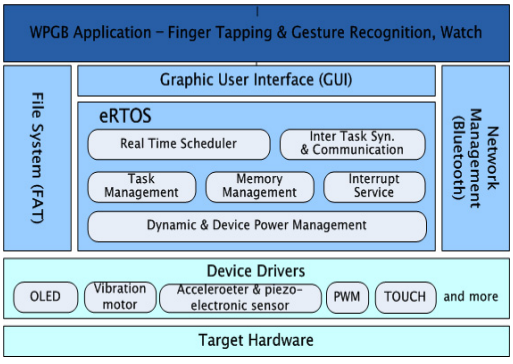


그림 1. eRTOS 구성도

2.1 태스크 관리(Task Management)

eRTOS는 멀티태스킹 환경을 지원하며, 이를 위해 각 태스크는 자신의 문맥(Context), 우선순위, 스택 영역, 실행함수, 타임슬라이스, 저전력 구동정책에 대한 정보를 태스크 자료 구조(TCB, Task Control Block)에 저장한다. 또한, 여러 태스크들이 CPU, 메모리 등과 같은 시스템의 자원을 서로 공유해야 하므로 공유자원에 대한 상호 배제 접근(Mutual Exclusion Access) 및 동기화(Synchronizing) 기법을 제공한다.

eRTOS는 태스크의 상태를 관리하기 위해 SUSPEND, READY, RUNNING, DELAYED, PENDING 5가지 상태를 정의한다. 태스크 생성(LP_TaskCreate()) 혹은 삭제(LP_TaskDelete()), 실행중단(LP_TaskSuspend()) 시 SUSPEND 상태가 되고, 생성된 태스크나 CPU 권한을 얻고자 하는 태스크는 LP_Resume()를 통해 READY 상태가 된다. CPU 권한을 얻은 태스크는 문맥교환(Context Switch)을 통해 RUNNING 상태로 천이하고, 실행 중인 태스크가 자원을 요청하여 자원 할당을 대기할 경우 PENDING 상태로 천이하는데, 폴링(Polling)으로 인한 무한 대기를 방지하기 위해 대기 시간을 설정하고 DELAYED 상태가 추가된다. 대기 시간 설정은 LP_SemPend()에 인자로 넘겨준다. [그림 2]는 각 이벤트에 따른 태스크의 상태 천이도를 나타낸 것이다.

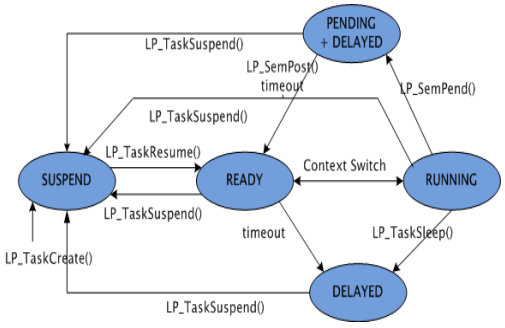


그림 2. 태스크 상태 천이도

eRTOS의 스케줄러는 0부터 27까지 28단계의 우선순위를 지원한다. 이 스케줄러는 스케줄러 자체의 메모리 사용량을 최소화하기 위해 비트맵 방식을 변형하여, 4Bytes(32bit) 크기의 변수에 28단계의 우선순위를 4그룹으로 분할하여 관리하는 경량 스케줄러이다.

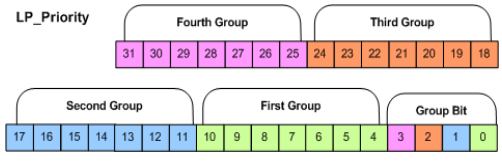


그림 3. 경량 실시간 스케줄러

[그림 3]은 경량 실시간 스케줄러를 도시한 것으로, 4Bytes 크기의 LP_Priority라는 전역 변수를 두고, 하위 4bit(0 ~ 3bit)를 그룹 비트로 해서 각 그룹에 7개의 bit를 할당한다. 특정 우선순위를 갖는 태스크가 수행대기 상태가 되면, 해당 태스크의 우선순위를 LP_Priority에 설정하고 그 우선순위에 해당하는 그룹 비트에도 1로 설정한다. 만약, 수행대기상태에 있는 태스크 중 최상위 우선순위를 갖는 태스크를 찾는 경우, 먼저 그룹 비트를 탐색하여 최상위 우선순위가 있는 그룹을 결정하고 그 그룹 내에서 7단계를 탐색하여 가장 높은 우선순위를 계산한다. 이때, 최상위 우선순위를 찾기 위한 탐색 오버헤드는 그룹 비트와 7단계의 우선순위만 검색하면 되므로 탐색시간 복잡도는 O(1)이다[17].

2.2 메모리 관리(Memory Management)

동적 메모리 할당 기법은 메모리 공간 효율성 극대화와 할당 시간의 최소화를 목표로 오랫동안 연구되어져 왔으며, 다양한 기법들이 소개되었다. 그 결과 주소 순서화된 최초 적합(Address Ordered First-Fit)과 최적 적합(Best-Fit)이 가장 효율적인 방법으로 다수의 논문에서 증명하고 있다[18-20]. 이에 eRTOS는 주소 순서화된 최초 적합 기법의 가변 길이 메모리 관리자를 구현하였다.

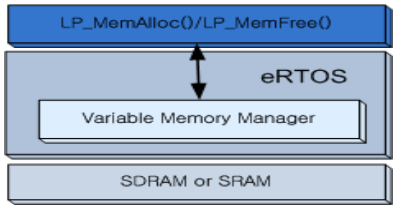


그림 4. 메모리 관리

[그림 4]의 가변 길이 메모리 관리자는 가변 크기의 메모리를 할당 및 해제하는 역할을 통해 동적으로 메모리를 관리한다.

2.3 세마포(Semaphore)

세마포는 공유자원에 대한 상호 배타적인 접근 방법 및 태스크간 동기화 기능을 제공한다. 여러 개의 태스크가 동시에 공유 자원을 접근하면 데이터의 충돌이나 잘못된 연산을 수행할 수 있기 때문에 세마포를 두어 이를 얻은 태스크만이 공유 자원을 접근하고, 수행을 마친 뒤 세마포를 반환하여 다른 태스크가 공유자원을 획득할 수 있도록 한다. 세마포가 관리하는 공유자원의 수에 따라 세마포 카운트 값을 0(바이너리 세마포) 또는 양의 정수(카운팅 세마포) 값으로 초기화한다.

2.4 태스크간 통신(Inter-Task Communication)

eRTOS의 응용프로그램은 여러 개의 독립적인 태스크가 동시에 실행되는 멀티태스킹 환경 하에서 구동된다. 이 태스크들 사이에 정보를 주고받기 위해서는 태스크간 통신 기법이 필요하다. eRTOS에서는 여러 태스크간 통신 기법 중 통신 오버헤드를 고려하여 메시지 포트(Message Port)와 태스크 포트(Task Port)를 구현하였다.

[그림 5]의 메시지 포트는 메시지 복사가 아닌 메시지가 저장된 메모리영역의 주소 정보를 통해 메시지를 하나의 태스크 혹은 여러 태스크에게 메시지가 저장된 주소를 전달한다.

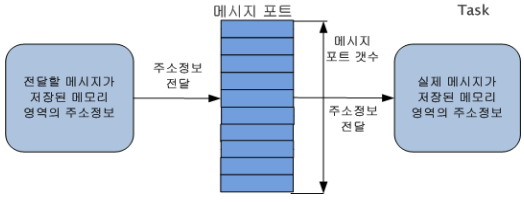


그림 5. 메시지 포트

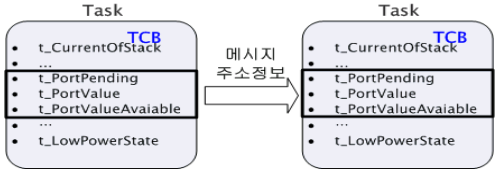


그림 6. 태스크 포트

[그림 6]의 태스크 포트는 메시지 포트와 달리 태스크와 태스크간 1:1 동기적인 통신을 제공한다. 태스크간 통신은 태스크 자료구조인 TCB의 필드를 통해 메시지가 저장된 메모리 영역의 주소 정보를 전달하는 방

식으로 가장 빠른 태스크간 통신 방식이다.

2.5 저전력 기법 – Dynamic & Device Power Management

2.5.1 동적 전력 관리

본 논문에서 구현한 저전력 기법은 IBM&Montavista의 동적 전력 관리 기법을 기반으로 WPGB 플랫폼에 맞게 수정하여 eRTOS상에 구현하였다. 동적 전력 관리 기법은 실시간 운영체제의 스케줄러에 대한 수정없이 탑재가 가능하다는 장점이 있다.

WPGB 응용프로그램들은 여러 태스크들로 구동되며, 그 중요도에 따라 각 태스크의 구동주파수와 버스클럭을 조절한다. CPU의 구동주파수는 공급 전압에 영향을 받는다. WPGB는 공급 전압을 조정하는 MAX계열의 칩이 탑재되어 있어 구동 주파수에 맞춰 공급 전압을 조정할 수 있다. CPU에서 소모하는 전력량은 공급전압의 제곱에 비례하기 때문에, 구동주파수를 조정하는 것도 중요하지만 그에 따른 공급전압을 조정하는 것이 소모 전력량의 감소에 더 큰 영향을 미친다.

표 1. 동적 전력 관리 구동 정책

구분	TASK+	TASK	TASK-
Battery High	Highest CPU frequency and voltage	Highest CPU frequency and voltage	Highest CPU frequency and voltage
Battery Low	High CPU frequency and voltage	Middle CPU frequency and voltage	Low CPU frequency and voltage
Battery Critical	Middle CPU frequency and voltage	Low CPU frequency and voltage	Lowest CPU frequency and voltage

본 논문에서 구현한 eRTOS의 저전력 기법은 동적으로 구동주파수와 공급 전압을 조정하는 DFS(Dynamic Frequency Scaling)와 DVS(Dynamic Voltage Scaling)가 가능한 WPGB 플랫폼에서 개발하여, [표 1]과 같은 구동정책을 구성하였다.

eRTOS의 동적 전력 관리 구동정책은 WPGB의 하드웨어 특성인 배터리로 동작한다는 점과 구동주파수와 버스클럭, 공급전압 레벨을 조정할 수 있다는 점을 고려하여, 총 4가지의 정책을 정의하여 구현하였다. 각 태스크는 실행 중에 동적 전력 관리자가 배터리 잔량을 측정하여 구동정책을 동적으로 변경할 수 있다.

Task+, Task, Task-, Idle 은 각 동적 전력 관리 정책 적용 시 동적 전력 관리자에 의해 동적으로 변경할 수 있는 CPU 구동주파수와 버스 클럭, 공급전압에 대한 정보들이다. 응용 프로그램 수행 시 CPU 구동주파수를 높이려면 Task+ 모드, 낮추려면 Task- 모드로 구동주파수와 버스클럭, 공급전압을 동적으로 변경할 수 있다. 또한, 응용프로그램 수행 중 태스크가 유휴(Idle) 상태가 되면 Idle 모드로 변경 된다. [표 2]는 본 논문에서 구현한 동적 전력 관리 기법의 저전력 관련 API이다.

표 2. 동적 전력 관리 API

함수	설명
LP_DPM_Init()	태스크 내 동적 전력 조절 초기화 함수
LP_Get_DPM_Policies()	구동전략 얻어오는 함수
LP_Set_DPM_Policies()	구동정책 설정 함수
LP_VolFrequency_Set()	구동전략에 따라 구동주파수와 전압을 조절하는 함수
LP_Cal_Frequency()	CPU 구동주파수 계산 함수
BatteryLevel_Check()	배터리량을 얻어오는 함수

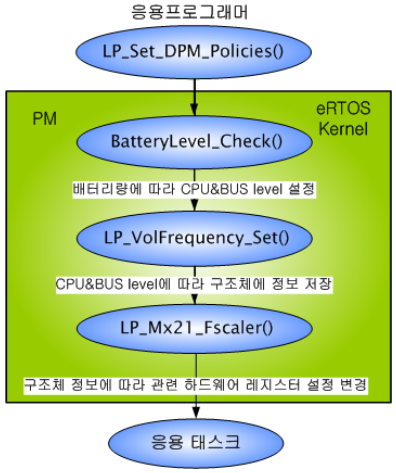


그림 7. 동적 전력 관리 개요도

[그림 7]은 동적 전력 관리의 전체 개요도 이다. 응용프로그래머는 LP_Set_DPM_Policies()를 통해 [표 1]과 같은 구동정책을 설정할 수 있다. eRTOS의 커널에서는 응용프로그래머가 설정한 구동정책과 BatteryLevel_Check()를 통해 얻어진 배터리 잔량에 대한 정보를 바탕으로 CPU와 BUS 구동주파수 레벨을 설정하고

LP_VolFrequency_Set()을 통해 CPU와 BUS 구동주파수, 전압에 대한 정보를 관리 구조체에 저장하며, LP_Mx21_Fscaler()를 통해 실제 관련 하드웨어 레지스터에 설정을 변경한다. 변경이 완료되면 응용태스크는 설정된 값으로 구동되게 되고, 그에 대한 확인은 LP_Cal_Frequency()를 통해 이루어진다. [그림 8]은 동적 전력 관리를 위해 설계된 구조체로 CPU, BUS 구동주파수, 전압 값과 레지스터 값을 관리하기 위한 필드로 구성되어 있고, [그림 9]는 [그림 8]의 구조체 정보에 따라 레지스터를 설정하는 함수이다.

```
typedef        struct LP_dpm_regs {
  LP_U32_t cscr; /* Clock Source Control Register */
  LP_U32_t cscr_mask;
     /* Clock Source Control Register mask */
  LP_U32_t mpctl0;    /* MCU PLL Control Register 0 */
  LP_U32_t freq_up_flag;
}dpm_regs; //동적 전력 관리를 위한 레지스터 정보 구조체
typedef        struct LP_dpm_md_opt {
  LP_U32_t v;          /* voltage value */
  LP_U32_t cpu_freq;    /* cpu fequency in Hz */
  LP_U32_t bus_freq;   /* bus fequency in Hz */
  dpm_regs regs;    /* Register values */
}dpm_md_opt
  ;// 전압, CPU와 버스 Frequency, 레지스터 정보를 포함하는 구조체
```

그림 8. 동적 전력 관리를 위한 구조체

```
LP_Status_t   LP_VolFrequency_Set(LP_U32_t cpulevel,
LP_U32_t bclklevel)
{
dpm_regs      *regs;
dpm_md_opt   *md_opt;
.....
switch (bclklevel) {
   // BUS Frequency 레벨에 대한 정보를 구조체에 저장하는 코드
  case BCLK_Level:
    md_opt->bus_freq = 16;
    regs->cscr   = CLK_CSCR_PRESC(0) |
      CLK_CSCR_BCLKDIV(15) | CLK_CSCR_IPDIV(1);
    break;
  case BCLK_Level1:
    md_opt->bus_freq = 33;
    regs->cscr   = CLK_CSCR_PRESC(0) |
     CLK_CSCR_BCLKDIV(7) | CLK_CSCR_IPDIV(1);
    break;
.....
 }
switch (cpulevel) {
   // CPU Frequency 레벨에 대한 정보를 구조체에 저장하는 코드
  case CPU_Level1:
    md_opt->cpu_freq = 266;
    regs->mpctl0 = /* not MPCTL0_CPLM | */
      CLK_MPCTL0_PD(0) | CLK_MPCTL0_MFD(123) |
      CLK_MPCTL0_MFI(7) | CLK_MPCTL0_MFN(115);
    break;
  case CPU_Level2:
    md_opt->cpu_freq = 133;
    regs->mpctl0 = /* not MPCTL0_CPLM | */
      CLK_MPCTL0_PD(1) | CLK_MPCTL0_MFD(123) |
      CLK_MPCTL0_MFI(7) | CLK_MPCTL0_MFN(115);
    break;
.....
}
   regs->freq_up_flag = 1;
   LP_Mx21_Fscaler(regs); // 설정값에 따라 전압, CPU&BUS
Frequency 레지스터값을 설정하는 함수
                    .....
}
```

그림 9. CPU, BUS, 전압 설정 함수 Pseudo 코드

2.5.2 동적 장치 관리

동적 장치 관리는 내장형 이동 시스템에서 소모 전력이 큰 장치인 CPU와 RAM, OLED와 같은 디스플레이 장치 등을 제어하여 소모전력을 감축시키는 기법이다. 이 기법을 eRTOS에 적용하기 위해 구현이 용이한 시간제한(timout) 기반의 shutdown 정책(장치 On/Off)을 구현하였다. eRTOS의 장치 전력 관리자는 [표 3]의 장치 전력 관리 API를 통해 장치들을 관리한다.

표 3. 장치 전력 관리 API

함수	설명
LP_DevPM_Init()	장치 전력 관리자 초기화 함수
LP_Dev_Register()	장치 관리자에 장치 등록 함수
LP_Dev_Unregister()	장치 관리자에 장치 해제 함수
LP_DevOff()	장치 전원 Off함수
LP_DevOn()	장치 전원 On함수

Ⅳ. 실험 환경 및 결과

본 논문에서 구현한 eRTOS는 웨어러블 컴퓨터의 일종인 시계타입의 WPGB 보드를 타겟으로 개발되었다. WPGB는 프리스케일사의 ARM926EJ core 기반의 i.MX21(MAX : 266MHz, Min : 66MHz)[21]의 MCU와

1MBytes의 SDRAM, OLED 디스플레이, 진동기, 가속도센서, 동작인식엔진 등으로 구성되어 있다. [그림 10]은 eRTOS에 대한 기능과 성능, 저전력 기능 검증을 위한 실험환경이다. 기능 실험은 WPGB보드의 UART를 통해 터미널 창에 출력된 결과로 실험 응용프로그램의 정상동작 여부를 확인하였다. 성능 측정 실험은 WPGB 보드의 GPIO를 설정하고, 이를 텍트로닉스(Tektronix) TDS3054B 오실러스코프를 이용하여 μS 단위까지 측정하였다. 또한, 소모 전력에 대한 측정은 에이질런트사의 34411A 디지털 멀티미터를 사용하여 일정 시간 간격으로 14000회의 순간 전류 변화 자료를 수집하여 평균값을 계산하는 방식을 사용하였다.

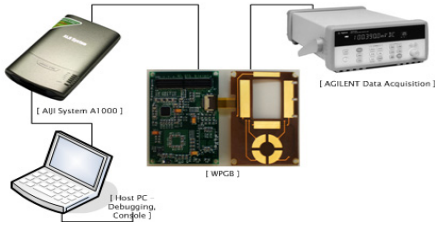


그림 10. 실험환경

1.1 eRTOS 기능 및 성능 분석

본 논문에서 구현한 eRTOS의 실시간 운영체제가 갖추어야 할 기본적인 태스크간 문맥교환과 메모리 할당 기법, 태스크간 통신과 같은 기능 검증 실험과 성능 측정 실험을 WPGB 보드에서 수행하였다.

기능과 성능 검증을 위해 28단계 우선순위를 갖는 28개의 태스크를 생성하여, 스케줄링 실험을 진행하였다. 또한, 메모리 관리 기능에 대한 실험은 태스크 생성시 태스크 스택영역을 가변길이 관리자로부터 할당 받아 그에 대한 기능과 성능 검증을 같이 수행하였다. 또한, 태스크간 통신은 메시지 포트, 태스크 포트의 기능 및 성능 비교를 위해 1:1 통신 실험을 수행하였다. 세마포에 대한 기능 실험은 가변길이 메시지 포트의 메시지 자원을 관리하기 위해 카운팅 세마포를 사용하므로 별도의 기능 실험은 하지 않는다.

[그림 11]의 좌측부터 eRTOS의 스케줄러, 메시지포트, 태스크포트 ITC 기능에 대한 실험결과이다.

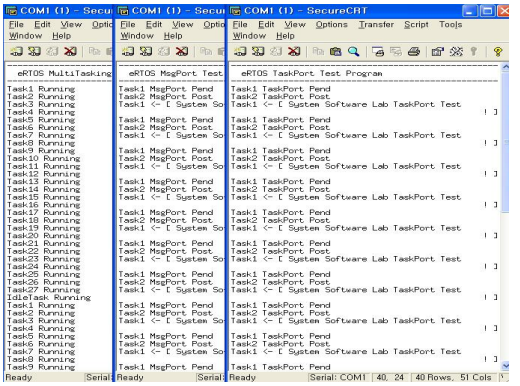


그림 11. eRTOS 기능 실험 결과

표 5. 성능 측정 결과

구 분	스케줄링 오버헤드	태스크간 통신	
		메시지포트	태스크포트
성능	10.504μ S	357.8μ S	48.53μ S

[표 5]는 eRTOS에 대한 성능 측정결과이다. 태스크간 통신 도구인 메시지포트와 태스크 포트의 성능을 비교해 보면, 메시지 포트는 메시지에 대한 주소정보를 저장하기 위해 메모리공간을 세마포로 관리해야 하는 오버헤드가 있는데 반해, 태스크포트는 태스크 제어블록내의 필드를 사용하여 1:1 통신을 하므로 성능이 우수하다. 하지만 메시지포트는 다수 간의 통신이 가능하다는 장점이 있다.

1.2 eRTOS 저전력 기법의 소모전력 분석

본 논문에서 구현한 저전력 기법의 소모 전력 측정을 위해 WPGB보드 상에 WPGB응용프로그램을 탑재하여 측정하였다. WPGB응용프로그램은 가속도 센서와 동작인식 엔진을 통해 사용자 입력을 받아 서버로 전송하는 역할을 한다. 사용자 입력 대기 상태인 것을 사용자에게 알리기 위해 OLED창에 입력대기 상태라는 정보를 디스플레이하고 진동기를 구동시킨다. 또한, OLED 디스플레이를 통해 현재 시간을 출력한다.

WPGB응용프로그램을 구성하는 각 태스크들에게 [표 4]와 같은 동적 전력 관리 기법의 구동정책을 설정한다. GESTURE를 처리하는 태스크는 다른 태스크에 비해 작업 처리량이 많아 비교적 빠른 TASK+ 정책을

표 4. 동적 전력 관리 기법 구동 정책

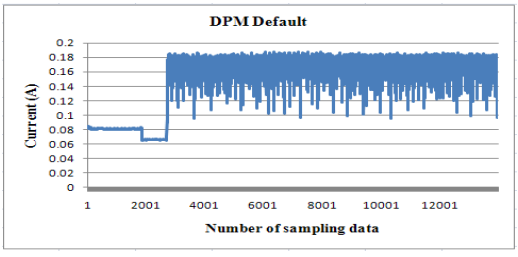
구분	GESTURE TASK	FINGERTAP TASK	CLOCK TASK	IDLE TASK
	TASK+	TASK	TASK-	Idle
Default	266/133(1.6V)	266/133(1.6V)	266/133(1.6V)	266/133(1.6V)
Idle Scaling	266/133(1.6V)	266/133(1.6V)	266/133(1.6V)	133/66(1.5V)
Load Scaling1	133/88(1.525V)	133/66(1.5V)	133/33(1.45V)	133/16(1.45V)
Load Scaling2	133/33(1.45V)	133/16(1.45V)	88/33(1.425V)	66/33(1.425V)

설정하고 인식 준비를 처리하는 FINGERTAP 태스크는 TASK 정책으로, CLOCK 태스크는 TASK- 정책으로 구동시킨다.

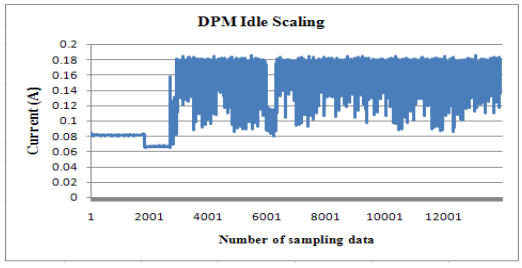
[표 6]은 [그림 12]의 소모 전류에 대한 평균전류 값이다. DPM Default는 [표 1]의 Battery High와 대응되는 구동 정책으로 저전력 기법을 적용하지 않은 경우이고, DPM Idle Scaling은 Battery Midium과 대응되는 정책으로 유휴 태스크에게 낮은 구동속도와 버스클럭, 그에 따른 전압을 인가한 경우이다. DPM Load Scaling Ⅰ은 Battery Low에 대응되는 경우이며, DPM Load Scaling Ⅱ는 Battery Critical인 경우이다. 각 구동정책에 따라 WPGB 응용프로그램이 정상동작하는 것을 확인하였으며, 소모전력 측정 결과를 통해 저전력 기법을 적용하지 않은 경우에 비해 42(c)~47(d)%의 소모 전력 감축을 확인하였다. 소모 전력에 대한 평가를 위해 전류 값을 측정한 이유는 순간적으로 공급되는 전력(Power)은 그 순간의 전류 값에 비례하므로, 주어진 시간구간 동안 소모되는 전체 소비전력은 그 주어진 시간구간 동안의 전류 값에 비례한다 할 수 있기 때문이다.

표 6. 구동 정책에 따른 평균 소모 전류 / 전력 감축율

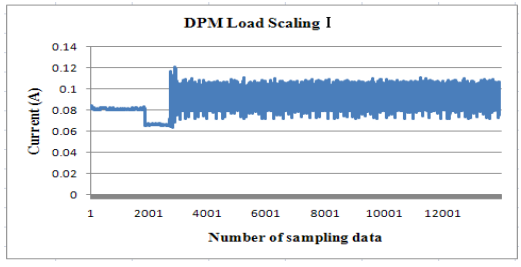
구분	Default	Idle Scaling	Load Scaling Ⅰ, Ⅱ
평균 소모 전류	0.146	0.140	0.085 / 0.077
전력 감축률	0%	4%	42% / 47%



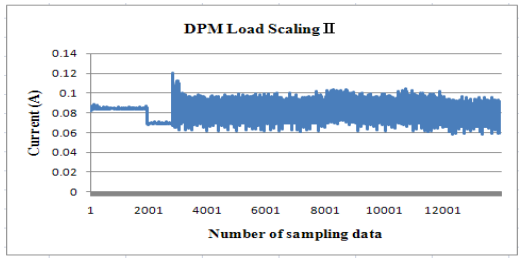
(a)



(b)



(c)



(d)

그림 12. 구동정책에 따른 소모 전류

[표 7]은 내장형 운영체제와 본 논문에서 구현한 RTOS의 풋프린트 크기와 저전력 기능이 포함되어 있는지에 대해 비교한 것이다[4-8]. [표 7]에서 보는 바와 같이 대부분의 상용 RTOS는 비교적 적은 수십KB 이하의 풋프린트 크기로 구현되어 있으나, 저전력 기능이 포함되어 있지 않다. 리눅스에서는 B.Brock과 K.

표 7. 내장형 운영체제의 풋프린트와 저전력 기능 포함여부 비교표

구분	eRTOS	µC/OS	pSOS	QNX	Nucleus	VxWorks	EmbeddedLinux	WinCE
풋프린트	18KB	12KB	49.5KB	101KB	74.2~92.5KB	286KB	1~2MB	1MB
저전력기능 포함여부	O	X	X	X	X	X	O	O

Rajamani[14]가 리눅스 2.4버전에 IBM&Montavista의 동적 전력 관리 기법을 탑재하여 MP3와 MPEG4 응용을 대상으로 실험하여 26~58%의 소모전력 감축효과를 확인하였다. 하지만 커널 크기가 1.1MB로 WPGB에 탑재하기에는 적합하지 않다. 또한, WinCE의 경우도 저전력 기법이 포함되어 있지만, 수MB의 풋프린트로 역시 적합하지 않다.

하드에어 제약사항을 극복하고 사용자의 QoS를 보장하기 위해서 웨어러블 컴퓨터는 풋프린트 크기가 작고 저전력 기능이 포함된 실시간 운영체제를 사용해야만 한다.

본 논문에서 구현한 eRTOS는 동적 전력 관리와 장치 전력 관리 기법의 저전력 기법과 18KB 풋프린트의 초소형 실시간 운영체제로 웨어러블 컴퓨터와 같은 차세대 PC에 적합하다.

V. 결 론

본 논문에서는 차세대 컴퓨팅의 하나인 웨어러블 컴퓨터를 위한 초소형 저전력 실시간 운영체제를 설계 및 구현하였다. 본 논문에서의 웨어러블 컴퓨터는 시계타입의 하드웨어 플랫폼으로, 휴대폰, PMP, PDA와 같은 기존의 내장형 시스템들보다 CPU 구동속도와 메모리 용량 등 하드웨어 자원이 보다 제한적인 시스템이다. 이와같은 하드웨어 제약을 극복하고 사용자가 요구하는 응용프로그램 구동의 QoS를 보장하기 위해서는 초소형의 실시간 운영체제 탑재가 필수적이다. 또한, 웨어러블 컴퓨터는 배터리로 동작하는 시스템으로, 소모 전력 감축과 발열량을 감소시킬수 있는 저전력 기법이 탑재되어야 한다.

본 논문에서 구현한 초소형 저전력 실시간 운영체제 eRTOS는 웨어러블 응용프로그램 구동에 필요한 멀티태스킹기능, 28단계의 우선순위 지원, 태스크 관리, 가변길이 메모리 관리, 메시지포트와 태스크포트를 통한 향상된 실시간 성 보장이 가능한 태스크간 통신 기능을 포함하고, 저전력 기능으로 동적 전력 관리와 장치 전력 관리 기능이 탑재되어 있다.

본 논문에서 eRTOS의 기능과 성능, 소모 전력 감축 효과에 대한 검증을 위해, 시계타입의 WPGB하드웨어 플랫폼에 eRTOS를 탑재하고 실험하였다. eRTOS의 풋프린트는 18KB로 초소형이면서 실시간 운영체제의 기능에 대한 기능 실험을 통해 검증하였으며, 10.504µS의 스케줄링 오버헤드와 357.8µS(메시지포트), 48.53µS(태스크포트)의 태스크간 통신 오버헤드를 보였다. 또한, 저전력 기법을 적용하지 않은 경우와 비교하여 42~47%의 소모 전력 감축효과를 확인하였다.

향후 연구 과제는 eRTOS에 대한 최적화 및 안정화 작업, WPGB이외의 웨어러블 컴퓨팅 하드웨어 플랫폼 탑재, 메모리를 포함하여 기타 하드웨어 장치에 대한 전력 관리 기능에 대해 연구하는 것이다.

참 고 문 헌

[1] 한동원, 박준석, “입고 다니는 차세대 PC”, ETRI CEO Information, 제19호, pp.1-12, 2005.

[2] 한동원, “차세대 PC”, TTA저널, 제95호, pp.132-137, 2004.

[3] ETRI, “정보 가전용 실시간 OS 컨퍼런스 : RTOS 2000 자료집”, 한국정보처리학회, 2000.

[4] http://www.windriver.com

[5] http://www.qnx.com

[6] http://www.atinucleus.com

[7] J. Jean, Labrosse, µC/OS, The Real-time Kernel, R&D Publications, 1993.

[8] http://www.realtime-info.be
[9] http:\\www.micorsoft.com
[10] 정갑주, 이민석, 최건, “내장형 리눅스”, 한국정보과학회지, 제18권, pp.18-25, 2000.
[11] http://www.research.ibm.com/arl/projects/dpm.html, 2002(11).
[12] W. Y. Xia and C. Xiangqun, "A Task-Specific Approach to Dynamic Device Power Management for Embedded System," in ICESS'05, Vol.00, pp.158-165, 2005.
[13] M. T. Schmitz and B. M. Hashimi, System-Level Design Techniques for Energy-Efficient Embedded Systems, Kluwer academic publishers, Boston, 2004.
[14] B. Brock and K. Rajamani, "Dynamic Power Management for Embedded Systems," Proc. of IEEE Int'l SoC Conf. (SoCC 2003), pp.416-419, 2003(9).
[15] 장래혁, “저전력 시스템과 저전력 소프트웨어”, 한국통신학회지(정보통신), 제18권, 제12호, pp.59-71, 2001.
[16] 조문행, 정명조, 김용희, 이철훈, “실시간 운영체제에서 작업량 관찰에 기반한 저전력 기법의 설계 및 구현”, 한국콘텐츠학회논문지, 제7권, pp.69-78, 2007.
[17] 조문행, 이철훈, “내장형 시스템을 위한 경량 실시간 스케줄링 기법의 설계 및 구현”, 한국차세대컴퓨팅학회, pp.5-12, 2007.
[18] T. Ogasawara, "An algorithm with constant execution time for dynamic storage allocation," 2nd International Workshop on Real-Time Computing Systems and Applications, pp.21-27, 1995.
[19] I. Puaut, "Real-Time Performance of Dynamic Memory Allocation Algorithms," 14th Euromicro Conference on Real-Time Systems(ECRTS'02), pp.41-48, 2002.
[20] P. R. Wilson, M. S. Johnstone, and M. Neely, "Dynamic Storage Allocation:A Survey and Critical Review," IWMM'95, pp.1-116, 1995.
[21] i.MX21 Application Processor Reference Manual, 2, Freescale, 2005.

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

최 찬 우(Chan-Woo Choi) 준회원

- 2007년 2월 : 충남대학교 컴퓨터공학과(공학사)
- 2007년 3월 ~ 현재 : 충남대학교 컴퓨터공학과 석사과정 재학

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

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 객원 연구원
▪ 1995년 2월 ~ 현재 : 충남대학교 컴퓨터공학과 교수
▪ 2004년 2월 ~ 2005년 2월 : Univ. of Michigan 초빙 연구원
<관심분야> : 실시간시스템, 운영체제, 고장허용 컴퓨팅