Nano-Q+에서 MCU 및 센서의 자동 슬립을 지원하는 전력 관리 기법

소선섭* · 최복동** · 은성배** · 김병호***

A Power Management Scheme for Sensors with MCU in Sleep Mode in Nano-Q+

Sun-Sup So* · Bok-Dong Choi** · Seong-Bae Eun** · Byung-Ho Kim***

이 논문은 2009학년도 한남대학교 학술연구조성비 지원을 일부 받아 연구되었음

요 약

본 논문에서는 센서 투명성을 지원하는 센서노드 운영체제에서 MCU가 슬립모드일 때 센서의 전원도 같이 차단할 수 있는 전력관리 기법을 제안한다. 전원 차단이 가능한 센서인지를 구별하기 위해 센서의 종류를 이벤트 센서와 폴링 센서로 구분하고 이들을 지원하기 위한 스케줄러를 설계하였다. 성능 분석을 위해 기존 센서네트워크 운영체제인 Nano-Q+에서 센서에 대한 표준 인터페이스를 지원하는 센서 투명성 기능을 구현하고, MCU와 함께 센서를 자동으로 슬립모드로 바꿀 수 있는 전력 관리자를 설계, 구현하여 성능을 분석하였다.

ABSTRACT

This paper proposes a power management scheme for sensor nodes in wireless sensor networks based on sensor node operating system supporting the sensor transparency, which can turn off the sensors when the MCU is in sleep mode. We classify the sensors in two types, that is, event sensors and polling sensors, to be able to decide whether the sensor is a type of sensors whose power supply can be turned off or not, and we design a new scheduler to support recognition of those different types of sensors. Implementing and evaluation of the scheduler and the power manager supporting sensor transparency are shown based on Nano-Q+.

키워드

무선센서네트워크, 센서노드 운영체제, 저전력, Nano-Q+, 센서 투명성

* 공주대학교 컴퓨터공학부
** 한남대학교 정보통신공학과
*** 경성대학교 컴퓨터공학과

접수일자 2009. 06. 29
심사완료일자 2009. 07. 20

Ⅰ. 서 론

무선센서네트워크는 주변 환경의 상태와 변화를 감지하여 사용자 시스템과 소통하는 근미래 유비쿼터스 컴퓨팅의 핵심 기술이다. 센서노드는 센서네트워크의 기본 구성 요소로써 환경에 따라 다양한 형태의 센서들이 장착된다. 센서노드는 운용되는 환경의 특성상 배터리를 사용하기 때문에 전력 소모를 줄이기 위한 많은 연구가 진행되었으며 이를 크게 두 가지로 구분하면 모듈의 전력 소모를 줄이는 하드웨어 접근 방식과 라우팅 알고리즘이나 모듈 전력 관리자를 사용하는 소프트웨어 접근 방식으로 나눌 수 있다[1][2].

소프트웨어 기반 전력관리의 기본 주체는 곧 센서노드 운영체제이다. 대부분의 센서노드 운영체제들은 전력관리를 위하여 MCU(Microcontroller Unit)의 자동슬립 기능을 제공한다[3]. 센서노드 운영체제는 태스크 큐에 들어있는 함수들을 순차적으로 수행하고 태스크 큐에 실행할 함수가 남아있지 않으면 MCU를 슬립모드로 전환하여 MCU의 전력소모를 줄일 수 있다.

MCU가 슬립 모드일 때에는 센서노드의 모든 작업 수행이 기본적으로 중단되기 때문에 센서 모듈의 기능도 처리할 수 없다. 따라서 이 때 센서 모듈의 동작은 불필요한데도 기존의 센서노드 운영체제는 센서의 전원을 제어하지 못하고 있다. 그 이유는 응용 환경에 따라 센서의 종류도 매우 많고 같은 종류의 센서라도 그 특성이 서로 달라 센서노드 운용체제가 각 센서 디바이스들을 개별적으로 관리하지 못하기 때문이다. 결과적으로 MCU가 슬립모드인 동안에는 센서 동작이 무의미함에도 센서는 전력을 사용하게 됨으로써 불필요한 전력 낭비가 발생한다.

범용 운용체제, 특히 임베디드 운영체제에서 사용되는 대표적인 전력관리 기법은 ACPI(Advanced Configuration & Power Interface)[4]이다. ACPI는 소프트웨어와 하드웨어 관한 인터페이스를 제공하여 입출력 장치의 전원을 직접 관리할 수 있게 함으로써 전력의 낭비를 줄일 수 있다.

센서노드 운영체제 분야에서 ACPI와 유사한 연구로는 다양한 센서들의 입출력 인터페이스를 표준화한 센서 투명성 연구가 있다[5]. 센서 투명성은 다양한 센서를 일관되게 관리할 수 있는 표준 인터페이스를 제공하는 것이다. 즉, 응용 개발자에게는 표준화된 API를 제공하고, 센서 부품 공급자에게는 표준화된 하드웨어 인터페이스와 디바이스 드라이버 인터페이스가 제공된다. 이러한 표준 인터페이스는 그림 1과 같이 3개 계층으로 구성되는 센서노드 플랫폼을 따르는데 아래가 하드웨어 계층이고 중간이 운영체제 계층, 위가 API(Application Programming Interface) 계층이다.

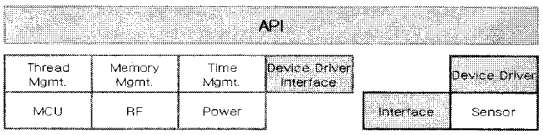

그림 1. 센서노드 플랫폼 구조
Fig. 1 Sensor Node Platform Architecture

본 논문에서는 센서 투명성을 지원하는 센서노드 플랫폼을 통해 MCU가 슬립모드일 때 센서의 전원도 같이 차단할 수 있는 전력관리 기법을 제안한다. 센서 종류의 다양성으로 인한 인터페이스 문제는 센서 투명성을 보완하여 해결하였다. 성능 분석을 위해 기존 센서 네트워크 운영체제인 Nano-Q+[6]에서 센서 투명성 지원 기능을 구현하고, MCU와 함께 센서를 자동으로 슬립 모드로 바꿀 수 있는 전력 관리자를 설계, 구현하여 성능을 분석하였다.

Ⅱ. 배 경

2.1 센서의 다양성

무선센서네트워크에서는 응용 환경에 따라 사용되는 센서들의 종류가 매우 다양하다. 동일한 목적의 센서라도 응용 시스템의 용도, 변환될 값의 범위, 동작 환경에 따라 다르다. 온도 센서를 예를 들면 측정값의 범위에 따라 상온에서 동작하는 반도체 온도 센서, 수온 측정용 막대형 온도 센서, 1천도 이상의 고온 측정용 온도 센서 등으로 다양하다. 센서의 출력 형식도 전압, 전류, 주파수 등으로 다양하며, 증폭이 필요한 센서, ADC(Analog Digital Converter)에 바로 붙일 수 있는 센서 등으로 다양하다.

이러한 센서의 다양성은 센서 하드웨어는 물론 소프트웨어 개발을 어렵게 만든다. 즉 하드웨어적으로는 공통된 센서 인터페이스를 설계하기가 어렵고, 센서노드

운영체제 입장에서는 응용 프로그래머에게 통일된 API를 제공하기가 쉽지 않다.

2.2 센서의 분류

센서의 동작 방식은 크게 두 가지로 나눌 수 있다. 첫째는 일정 주기로 데이터를 측정하는 폴링 방식이고, 둘째는 이벤트가 발생했을 때 측정된 데이터를 전송하는 이벤트 방식이다. 폴링형 센서는 주기적으로 측정된 센서값을 평균하여 처리하는 응용에 주로 사용되며 일부 데이터 손실이 발생하더라도 문제가 되지 않는다. 폴링 주기 사이에는 전원을 차단하여 전력의 낭비를 막을 수 있다. 온도, 조도, 습도를 측정하는 각종 환경 모니터링, 위치인식서비스 등에 활용된다.

이벤트형 센서는 이벤트가 발생했을 때에만 데이터를 전송하며 신뢰성 있는 데이터 전달이 필수적으로 요구된다. 이벤트 발생 시기를 미리 예측할 수 없어 센서의 전력을 함부로 차단할 수 없기 때문에 저전력이 요구되지만 효율적인 전력관리 기법이 어렵다.

2.3 기존 센서노드 운영체제

센서노드에서 저전력 소모를 위한 전력관리는 중요한 이슈 중의 하나이다. 센서노드 운영체제들은 저전력 통신 알고리즘이나 MCU의 전력관리 기법을 제공하지만 센서 모듈에 대한 전력관리 방안은 아직 없다.

기존 센서노드 운영체제 가운데 가장 먼저 개발된 Tiny-OS[7][8]는 하드웨어의 제약을 고려하여 매우 소형으로 개발되었다. 저전력 기능을 위주로 개발되었으며 컴포넌트 기반의 프로그래밍 패러다임을 지원한다. Tiny-OS는 프로세스 개념을 지원하지 않기 때문에 프로그래밍이 어렵다는 단점이 있고, 특히 센서나 구동기 디바이스의 복잡성에 대한 고려는 없다.

Nano-Q+ 운영체제는 한국전자통신연구원(ETRI)에서 개발된 센서노드 운영제제이다. 전력 소모를 줄이기 위해 저전력 슬립모드를 제공하지만 역시 센서 모듈에 대한 별도의 전력관리 기법은 없다. Nano-HAL이라는 디바이스 드라이버 영역이 있어 하드웨어의 투명성을 지원하기는 하지만 프로그래머가 각각의 API를 알고 있어야 한다는 단점이 있다.

2.4 소형 임베디드 시스템의 전력관리

배터리를 전원으로 사용하는 소형 임베디드 시스템은 사용할 수 있는 전력의 양과 시간이 제한적이다. 따라서 제한된 전력을 효과적으로 사용하기 위한 전력관리는 소형 임베디드 시스템에서도 중요한 이슈이다.

이러한 임베디드 시스템을 위한 다양한 전력관리 기법들이 연구되었는데[9][10], 회로 설계 단계에서부터 장치를 저전력으로 동작하도록 설계하는 하드웨어 측면과 운영체제 혹은 시스템 소프트웨어 측면에서 전력을 관리하는 방법이 있다. 소프트웨어 방식으로는 프로세서의 공급 전압을 조절하여 전력 소모량을 줄이는 동적 전압조절 기법(DVS), 장치의 상태에 따라 적절한 전원 상태로 전이하여 전력 소모를 줄이는 동적 전력관리 기법(DPM) 등이 있다.

Ⅲ. 시스템 설계

3.1 센서의 전력 소모

센서노드는 프로세싱을 위한 MCU, 통신을 위한 RF 모듈 및 센서 자체로 구성된다. 따라서 센서노드의 전력 소모는 이들 세 가지 구성요소에서 비롯됨으로 각각의 전력 소모량을 계산해 볼 필요가 있다. 표 1은 대기 모니터링 응용에서 주기적으로 온도를 측정하는 센서노드의 전력 소모량이다. 측정 주기는 5분이고 데이터 전송을 위한 RF 동작 시간은 1초이며, 대기 시간에는 MCU와 RF 모두 슬립 모드에 있어 전력 소모는 없다고 가정한다.

표 1. 실제 전력 소모
Table 1. Actual Power Consumption

MCU	5.5mA × 1/300 = 18μA
RF	12mA × 1/300 = 40μA
센서	5mA × 298/300 = 4.97mA

표에서 보는 것처럼 매 5분 주기마다 소모되는 전력의 합, 5.028mA 중에 센서의 전력 소모는 4.97mA로써 전체의 98%를 차지한다. 결과적으로 측정 시에 센서가 필요로 하는 전력은 작아도 전원이 관리되지 않으면 전력 낭비가 크다는 것을 알 수 있다.

3.2 센서 전원 제어 하드웨어

센서의 전원을 관리하기 위해서는 그림 2와 같이 수

정된 센서 플랫폼이 필요하다. 기존의 플랫폼은 센서의 전원 관리를 장치가 따로 없지만 그림 2의 플랫폼에서는 각각의 센서에 부착된 전원 스위치를 통해 전원 공급을 제어할 수 있다.

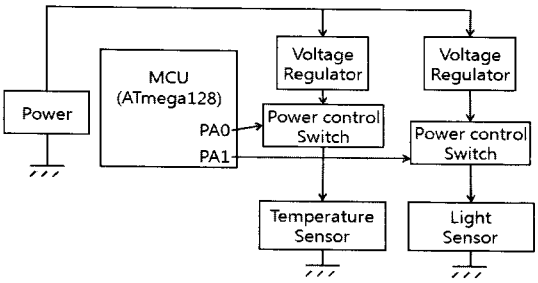

그림 2. 센서 전원 제어 하드웨어
Fig. 2 Power Control Hardware for Sensors

센서의 전원 관리를 위해 추가된 스위치 또한 약간의 전력을 필요로 한다. 표 2는 스위치를 켜고(On) 끌(Off) 때의 전원 스위치의 소모 전류를 나타낸다. 표에서 보는 바와 같이 스위치 자체의 전력 소모는 상대적으로 작지만 기존 플랫폼에 추가되는 하드웨어이므로 점검할 필요가 있다.

표 2. 스위치 특성
Table 2. Switch Specifications

모델명	공급 전류	전류량 (On)	전류량 (Off)
ADG719	30mA	6nA	1nA
ADG819, ADG829	200mA	21nA	10nA
MAX1607	500mA	14uA	1uA

3.3 센서의 지정

센서의 전원 관리를 위해 본 논문에서는 센서를 두 가지 형태로 분류한다. 첫째는 지속적으로 전원이 켜져 있어야 하는 이벤트 센서이고 둘째는 주기적으로 동작하여 각 주기 사이에는 전원이 꺼져 있어도 되는 폴링 센서이다.

이벤트 센서는 언제 이벤트가 발생할 지 알 수 없기 때문에 전원 관리가 어렵다. 반면에 폴링 센서는 측정 주기를 통해 전원을 관리할 수 있다.

센서 투명성이 보장되면 모든 종류의 센서에 대한 추상화된 인터페이스를 제공할 수 있다. 따라서 센서 종류에 상관없이 동일한 API로 센서 디바이스에 대한 프로그램이 가능하다. 즉, 응용 프로그램은 device_connect() 함수를 이용하여 표 3과 같이 센서와 하드웨어 연결 포트(hardware Mapping Table)를 구성한다. 이 때 소프트웨어 개발자는 디바이스 드라이버를 제공하고 하드웨어 개발자는 하드웨어의 포트 정보를 제공해야 한다.

표 3. 하드웨어 연결 포트
Table 3. Hardware Mapping Table

dev_name	pport	dport0	dport1	dport2	dport3
adcv_v1_0	FA0	FP0	-	-	-
intr_v2_0	PA1	PD0	-	-	-
actr_v0_0	-	PA3	-	-	-

기존 센서 투명성 연구[5]에서는 센서의 형태를 구분하지 않는다. 본 논문에서는 이벤트 센서와 폴링 센서를 구분하여 지정하기 위해 센서 형태 지정 기능을 추가하여 전원을 관리하도록 하였다. 즉, 센서의 전원 관리가 필요한 폴링 타입과 전원 관리가 불가능한 이벤트 타입의 두 가지 방식으로 지정할 수 있도록 하였다. 센서의 타입을 추가한 센서 투명성 API는 표 4와 같다.

표 4. 센서 타입 지정
Table 4. Type Assignment of Sensors

HAL Device Connect
device_connect("Motion-Event", "intr_v2_0", TE100_drv_pt, event)
device_connect("Temp-sensor", "adcv_v1.0", lm64_drv_pt, polling)
device_connect("Relay-Acto", "actr_v0_0", relay_drv_pt, polling)

Nano-Q+는 쓰레드 생성 시 사용하는 센서의 타입을 확인하고 폴링 시에만 저전력 관리자를 사용한다.

3.4 전력 관리자 구조

먼저 Nano-Q+에서 3개의 쓰레드를 생성한다. 생성된 쓰레드는 태스크를 실행하고 각 태스크가 센서를 연결

할 수 있도록 하였다. 쓰레드 생성 과정을 포함한 전력 관리자의 구조는 그림 3과 같다.

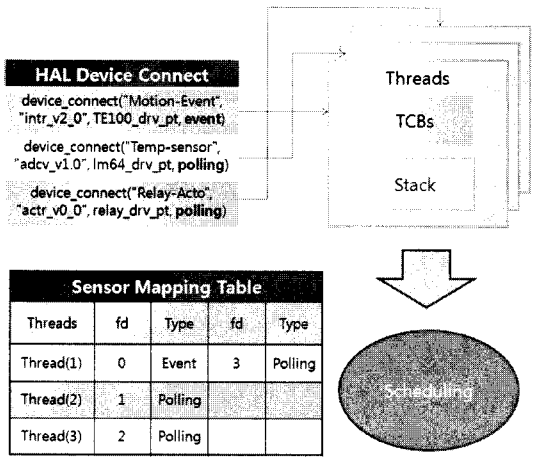

그림 3. 전력 관리자
Fig. 3 Power Manager

메인 함수가 쓰레드를 생성할 때에는 아래 코드와 같이 센서의 타입을 이벤트와 폴링으로 구분하여 생성한다. 이는 응용 프로그래머가 사용할 센서의 타입을 정하고 연결을 해 주어야 한다.

```
int main(void)
{
   ...
   init_nos();
   uart_send_str("\n Power Manager\n");
   thread_create(1, task1, &args1, sen1.type, ...);
   thread_create(2, task2, &args2, sen2.type, ...);
   thread_create(3, task3, &args2, sen3.type, ...);
   start_sched();
   return 0;
}
```

테스크 함수는 센서 투명성을 지원하는 함수이다. 센서의 이름, 함수, 타입 등을 사용하여 센서 디바이스를 연결한다. 센서의 타입을 지정하지 않으면 기존의 스케줄러가 사용된다.

```
task1(void *args)
{
   ...
   device_connect("light-sensor", "adcv_v1.0",
         tsl2561_drv_pt, polling);
}
```

디바이스 연결 시에 센서의 타입을 정해줌으로써 센서의 전원을 관리할 수 있다. 스케줄러는 쓰레드가 슬립모드로 전환될 때 쓰레드에 연결된 센서의 타입을 확인하고 센서의 타입이 폴링이면 자동으로 센서의 전원을 차단한다. 쓰레드가 모두 슬립모드 상태이면 MCU를 슬립모드로 전환한다.

기존의 센서 운영체제에서도 센서의 전원 차단이 가능하기는 하지만 센서의 타입과 투명성이 제공되지 않아 자동으로 수행할 수는 없었다. 센서 투명성이 제공되면 운영체제는 센서를 fd값으로 받아들여 특정 센서에 관계없이 fd값만으로 센서의 타입을 구분하여 전력을 관리한다.

3.5 저전력 스케줄러

기존의 Nano-Q+도 하드웨어 추상화를 제공하지만 프로그래머가 모든 API를 알고 있어야 한다는 단점이 있다. 센서 투명성은 표준화된 API와 인터페이스를 제공하기 때문에 센서에 대한 프로그래밍이 보다 쉽다.

Nano-Q+에서 쓰레드 문맥 교환은 현재 쓰레드 문맥의 저장, 쓰레드 스케줄링, 다음 실행될 쓰레드의 문맥 복구로 이루어진다. 이 과정에서 현재 쓰레드는 준비 큐에 들어가고 스케줄러를 통해 준비 큐에 들어있는 쓰레드들 중 다음에 실행될 특정 쓰레드를 선택하게 된다.

쓰레드가 슬립모드 상태에 들어가면 사용하고 있던 센서의 전원을 차단하여 전력 관리를 한다. 준비 큐에 들어있는 쓰레드가 없을 때에는 MCU도 슬립모드로 전환시킨다.

Ⅳ. 구현 및 성능 평가

4.1 Nano-Q+에서의 구현

구현은 한국전자통신연구원(ETRI)에서 개발한 센서 노드 운영체제인 Nano-Q+를 기반으로 하였다. Nano-Q+

는 ATmega128L을 사용하는 MCU 부분과 무선통신을 위한 CC2420 RF모듈, 센서와 구동기를 지원한다. 쓰레드 기반 스케줄러를 지원하며 RF 메시지 핸들링을 지원하는 네트워크 프로토콜 스택을 지원한다. 응용 프로그램은 시스템 API를 통해 운영체제 부분과 상호작용할 수 있 있다.

4.2 램 용량

센서 투명성을 Nano-Q+에 적용하면 약간의 오버헤드가 발생한다. 가스 센서를 사용하여 응용 프로그램을 작성하면 기존의 Nano-Q+에 비해 약 19Kbyte 정도의 코드가 추가된다. 또한 추가된 데이터형의 사용량도 늘어나게 된다.

여기에서는 순수하게 전력 관리자를 구현하기 위한 오버헤드를 계산해 보았다. 하나의 모듈에 센서가 5개있다고 가정하면 결과는 아래와 같다.

$$\begin{aligned}\text{Overhead} &= (\text{H/W Mapping Table} + \\ &\quad \text{Sensor Mapping Table}) \times 5\text{개} \\ &= (5\text{bytes} + 2\text{bytes}) \times 5\text{개} = 35\text{bytes}\end{aligned}$$

센서 투명성이 추가된 Nano-Q+를 사용하면 전력 관리자의 오버헤드는 크게 늘어나지 않는다. 센서 투명성의 Hardware Mapping Table과 Sensor Mapping Table을 사용하고 센서의 타입을 추가하면 된다.

4.3 센서노드 전류량 측정

실험 환경은 Nano-24 센서 노드에서 MCU만 슬립모드로 전환하는 경우와 MCU와 함께 센서까지 슬립모드로 전환하는 경우를 비교하였다. 전류 소모량 측정은 디지털 멀티미터를 사용하였고 센서는 조도센서를 사용하였다.

표 5에서 보는 바와 같이 먼저 MCU가 조도센서를 사용하여 정상적으로 동작하는 경우에 노드에서 소모되는 전류량은 9.05mA이다. 다음으로 MCU가 슬립모드 상태일 때 소모되는 전류량은 5.20mA이며, 끝으로MCU와 센서가 모드 슬립모드 상태일 때 소모되는 전류량은 2.24mA이다. 결과적으로 MCU만 슬립모드 상태와 비교하여 센서를 같이 슬립모드 상태로 전환하면 2.78mA의 전력소모가 줄어드는 것을 확인할 수 있으며 이는 46%에 해당한다.

표 5. 전류량 측정
Table 5. Current Measurement

	동작상태	MCU sleep	MCU & Sensor sleep
전류량	9.05mA	5.20mA	2.42mA

V. 결론 및 향후 연구방향

본 논문에서는 무선센서네트워크의 저전력 소모를 위해 센서노드의 센서의 전원을 자동으로 차단할 수 있는 전력 관리자를 설계하고 구현을 통해 성능을 평가하였다. 제안한 방법은 MCU가 슬립모드 상태일 때 센서노드 자체의 정상적인 작동도 중단됨으로 이 때 센서 모듈의 전원을 자동으로 차단하는 것으로써 이를 통해 센서노드의 불필요한 전력 낭비를 줄일 수 있다. 구현을 통한 실제 측정을 통해 기존의 방법보다 46%의 전력 소모를 줄일 수 있음을 확인하였다.

제안한 방식에서는 전력 관리가 가능한 센서 여부를 구분하기 위해 센서 투명성을 사용하였으며, 센서 투명성과 전력 관리자 추가로 인한 오버헤드는 큰 영향이 없음을 보였다.

향 후 다양한 특성의 센서들을 제어하는 방법이 보다 표준화된다면 하드웨어 및 소프트웨어의 저전력화가 가능하여 유비쿼터스 센서네트워크 산업의 생산성과 기술력 향상에도 기여할 수 있을 것이다.

참고문헌

[1] 엄흥식, 김건욱, "저전력 마이크로 컨트롤러를 위한 명령어 레벨의 소모전류 모델링 및 최적화에 대한 연구," 전자공학회논문지, 43권, CI편, 5호, 2006, pp. 1-7.

[2] 조문행, 이철훈, "DPM 기법을 적용한 저전력 실시간 운영체제 설계 및 구현," 한국정보과학회학술대회, 33권, 2호(A), 2006, pp. 281-286.

[3] 박승민, "센서네트워크 노드 플랫폼 및 운영체제 기술 동향," ETRI 전자통신동향분석, 21권, 1호, 2006,

pp. 14-24.
[4] 이형석, 정영준, "임베디드 운영체제 커널 기술 동향," ETRI 전자통신동향분석, 21권, 1호, 2006, pp. 1-13.
[5] 은성배, 소선섭, 김병호, "센서투명성을 지원하는 센서노드 운영체제 구조," 한국컴퓨터종합학술대회(KCC 2008), 2008, pp. 311-312.
[6] S. Park, J. Kim, K. and D. Kim, "Embedded Sensor Networked Operating System" Proc. of 9th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, 2006, pp. 117-124.
[7] T. Schmid, H. Dubois-Ferriere, and M. Vetterli, "Sensorscope: experiences with a wireless building monitoring sensor network," Proc. of Workshop on Real-World Wireless Sensor Networks, 2005.
[8] P. Volgyesi, A. Ledeczi, "Component-based development of networked embedded applications," Proc. of 28th Euromicro Conference, 2002.
[9] 이재동, 허정연, "태스크 동기화가 필요한 임베디드 실시간 시스템에 대한 효율적인 전압 스케쥴링," 정보과학회논문지: 시스템 및 이론, 35권, 5-6호, 2008, pp. 273-283.
[10] 황영시, 정기석, "E-ACPI: 임베디드 시스템에서 적극적 전력 관리를 위한 전력관리 인터페이스 구현," 전자공학회논문지, 45권, SD편, 3호, 2008, pp. 36-43.

저자소개

소선섭(Sun-Sup So)

1986년: 이화여대 전산학과 학사
2001년: KAIST 전산학과 석박사
988년~1995년: 국방과학연구소 연구원
1995년-현재: 공주대학교 컴퓨터공학부 부교수
※관심분야: 소프트웨어 테스팅, 임베디드 소프트웨어, 센서네트워크

최복동(Bok Dong Choi)

2009년: 한남대학교 정보통신공학과 학사/석사
2009년~현재: 한남대학교 정보통신공학과 박사과정
※관심분야 : USN, 센서노드 운영체제

은성배(Seong Bae Eun)

1985년: 서울대학교 전산학과 학사
1995년: KAIST 전산학과 석박사
1995년~현재: 한남대학교 정보통신공학과 교수
※관심분야: 실시간시스템, 유비쿼터스 센서네트워크

김병호(Byungho Kim)

1990년: 연세대학교 전산과학과 학사
1997년: KAIST 전산학과 석박사
2007년~현재: 경성대학교 컴퓨터공학과 조교수
※관심분야: 컴퓨터구조, 센서네트워크