논문 2008-4-1

TinyOS 기반의 센서 노드 제어 알고리즘

Sensor Node Control Algorithm Based on TinyOS

부준필*, 양현규**, 김도현***

Jun-Pil Boo*, Hyeon-Gyu Yang, Do-Hyeon Kim*****

요 약 최근에 대표적인 센서 노드 운영체계인 TinyOS를 이용하여 센서 네트워크를 개발하여 다양한 유비쿼터스 응용 서비스를 개발하고 있다. 이들 TinyOS 기반의 센서 네트워크에서는 상황 정보를 획득하기 위해 센서로부터 센싱된 정보의 전달과 수집을 집중적으로 수행한다. 이에 본 논문에서는 센서 노드의 전력 상태를 파악하여 이를 토대로 센서 노드를 수면, 활동, power off 모드로 전환하는 센서 노드 제어 알고리즘을 제시한다. 그리고 이 알고리즘을 토대로 센서 네트워크의 센서 노드, 싱크, 서버에서 센서 제어 모듈을 설계하고 구현한다. 이를 위하여 센서 노드의 센서 전력제어 모듈과, USN 서버의 센싱 데이터 수신 및 도시 모듈과 센서 제어 모듈을 설계하고 TinyOS와 자바 언어를 이용하여 구현한다. 이를 통하여 센서 노드의 전력 상태를 확인하여 데이터 수집이 어려울 경우 수면이나 power off 모드로 전환하여 전력 손실을 방지하고, 주변 환경이 정상적일 경우 활동 모드로 변경함으로써 효과적으로 센서 노드의 전력을 제어할 수 있을 것으로 사료된다.

Abstract Recently, there is developing various ubiquitous application services using sensor networks based on TinyOS represented the operating system of sensor node. These sensor networks perform the collection and the transmission of sensing data from sensor node to get the context information. In this paper, we proposes the sensor node control algorithm which converts a sensor node to sleep, active, power off mode according to monitoring result of the voltage state of sensor node. Also, we designs and implement the sensor control module on server, sink, sensor node of sensor networks using this algorithm. It designs a sensor voltage control module of sensor node, data receive and display module of USN server using a java language and TinyOS. And, it checks the voltage state of sensor node, and it changes one of the sleep or power off modes in case of high voltage loss. Accordingly, we effectively use the power of sensor nodes as changing control modes of sensor nodes.

Keywords : Sensor Control Module, Wireless Sensor Network, TinyOS.

Ⅰ. 서 론

최근 유비쿼터스 사회에 대한 관심이 점점 증가하고, 이에 따른 무선 센서 네트워크에 대한 관심도 높아지고 있다. 현재도 많은 분야에서 무선 센서 네트워크가 활용되고 이러한 무선 센서 네트워크를 효율적으로 구현하기 위한 연구가 활발히 진행되고 있다.

무선 센서 네트워크는 센서로부터 수집한 데이터를 가공하고 이를 무선 송수신 장치를 이용하여 이를 외부로 전달하는 일련의 시스템이라고 할 수 있다. 더 넓은 의미로는 이와 관련된 모든 기술을 일컫는데 센서 노드 하드웨어에 들어가는 운영체제와 미들웨어, 싱크 노드를 통해 전달된 정보를 모니터링 하는 시스템 등도 여기에 포함된다. 무선 센서 네트워크를 구성하는 센서노드는 외부에서 무한정의 에너지를 공급 받는 것이 아니기 때문에 저전력으로 설계되어야 한다. 또한 다양한 응용에

*정회원, 제주대학교 통신컴퓨터공학부
**정회원, 제주대학교 통신컴퓨터공학부
***종신회원, 제주대학교 통신컴퓨터공학부(교신저자)
접수일자: 2008.7.15, 최종수정일자: 2008.8.2

맞게 효과적으로 활용될 수 있도록 유성성과 모듈성을 지녀야 하며, 대규모 응용을 위해 가격이 저렴해야 한다. 특히, 무선 센서 네트워크의 에너지를 효과적으로 사용하기 위해 에너지 관리 및 센서 노드 제어에 많은 관심이 집중되고 있다[1].

본 논문에서는 무선 센서 네트워크에서 센서 노드들을 효과적으로 활용할 수 있도록 하기 위하여 센서 노드의 전력 상태를 기반으로 센서 노드를 수면, 활동, power off 모드로 전환하는 센서 노드 제어 알고리즘을 제시한다. 그리고 서버(단말 컴퓨터)에서 센서 노드들로부터 수집된 전압을 감시하고, 이 전압에 따라 센서 노드를 제어하는 모듈을 설계하고 구현한다.

서론에 이어 2장에서는 TinyOS를 비롯한 센서 네트워크 기반의 기존 초소형 운영체제를 살펴본다. 3장에서는 센서 노드를 제어하기 위해 제안된 센서 노드, 서버 및 싱크 노드에서 제어를 위한 순서도를 설명하다. 그리고 4장에서는 TinyOS 기반의 센서 노드, 싱크 및 서버의 데이터 수신 및 제어 알고리즘을 구현하고 실험한다. 마지막으로 결론을 맺는다.

II. 관련연구

센서 네트워크에 사용되는 운영체제는 센서 네트워크를 지원하기 위하여 필수적으로 만족시켜야 하는 조건이 있다. 운영체제는 저전력, 적은 코드 사이즈, 최소한의 하드웨어 리소스를 사용해야 한다. TinyOS는 이러한 조건을 만족하면서 여러 가지 많은 특징을 가짐으로써 임베디드 운영체제로 주목받고 있다. TinyOS는 매우 작은 용량이라는 점이 가장 중요한 장점이다. 핵심 커널은 4000 바이트 이하이고 데이터 메모리는 256바이트 이하인 초소형 운영체제이다. TinyOS는 매우 적은 메모리로 멀티태스킹을 지원할 수 있는 이벤트 기반 모델을 선택했다. 이러한 이벤트 기반 멀티 태스크 방식은 초당 4만 번의 컨텍스트 스위칭을 지원한다. 이러한 이벤트 기반 멀티 태스크 방식을 사용함으로써 초소형 모트의 중요한 특징인 저전력 소모가 가능하게 된다. 이 방식은 이벤트의 발생이 없는 시간 동안 CPU를 수면 모드(sleep mode)로 전환함으로써 효율적인 CPU의 사용을 통해 저전력을 실현할 수 있게 된다. TinyOS의 특징 중 하나는 모듈형태로 프로그래밍을 한다는 것이다. 모듈은 객체지향프로그래밍의 클래스와 비슷한 개념으로서 모듈은 이용하여 모듈 내에 정의된 함수들을 가져다 사용할 수가 있다. 네트워크를 통해 데이터를 전송하거나 센서노드에서 데이터를 읽어오기 위한 모듈들이 시스템라이브러리로 제공된다.

MANTIS OS는 MultimodAl NeTworks of Insitu Sensor의 약자로 콜로라도 대학에서 개발된 최고의 멀티스레드를 지원하는 센서네트워크용 임베드디 운영체제이다. TinyOS와 달리 초소형 스레드 기반의 멀티 스레드 구조를 채택하여 실제 리눅스 환경과 비슷한 프로그래밍 환경을 제공하기 때문에 센서 네트워크를 처음 접하는 사용자도 쉽게 개발이 가능하다.

Contiki의 가장 큰 특징은 동적 모듈 재할당 기능을 제공한다는 것이다. Contiki는 기본적으로 TinyOS, SOS와 같이 이벤트 기반모델을 따르고 있지만 프로토스레드라는 기술을 통해 제한적이기는 하지만 스택 지정 없이 멀티스레드 같은 코드 간의 동기화를 제공한다. 하지만 지역 변수를 사용할 수 없고 지정된 구간에서만 블로킹이 가능하다는 제약이 있어 멀티스레드 센서 OS라고는 할 수 없다.

RETOS는 Resilient, Expandable and Threaded Operating System의 약자로 연세대학교 모바일 임베디드 시스템 연구실에서 개발된 운영체제이다. RETOS는 멀티스레드 프로그래밍 인터페이스를 제공하며 커널의 동적 재구성을 통한 확장이 가능하다는 특징을 가지고 있다. 또 다른 특징으로 스택 스위칭을 통한 듀얼모드를 지원하고 있다[2].

nesC는 TinyOS의 구조와 실행 모델에 적합하게 고안된 C의 확장된 언어이다. nesC의 컴포넌트는 state를 내부에 가지고 있고 잘 정의된 인터페이스를 통해 인터럽트 한다는 점에서 객체와 유사하다. 그러나 컴포넌트들과 컴포넌트 사이의 사용작용이 컴파일 시점에 정해진다는 점에서 큰 차이가 있다. nesC는 C기반의 언어로 2가지 특징을 가지고 있다. 하나는 컴포넌트들은 인터페이스를 통해 상호작용하는 프로그래밍 모델이라는 점이고 다른 하나는 run-to-completion 태스크와 인터럽트 핸들러의 이벤트 기반 동시 모델이라는 점이다.

III. 센서 노드 제어 알고리즘

센서 노드를 제어하기 위해서는 센서 노드 뿐만아니

라 서버 및 싱크 노드에서 제어를 위한 순서도가 요구된다. 센서 노드에서 수집된 센서 데이터들은 메시지의 형식으로 묶어서 지그비 통신 방식을 통하여 서버의 싱크 노드로 전송된다. 그리고 싱크 노드는 받아들인 데이터를 직렬통신을 이용하여 서버로 전송되고 있다. 여기서 싱크 노드로 들어오는 데이터의 메시지에는 각 센서노드로부터 보내지는 데이터를 식별할 수 있도록 각 센서 노드에 부여된 고유의 노드 ID를 포함한다. 그림 1에서는 서버와 센서노드 사이에서 센서 데이터와 제어 데이터의 흐름을 나타내고 있다.

제어 데이터는 서버에서 사용자로부터 데이터를 입력받아 직렬통신을 이용하여 서버의 싱크 노드로 전송되고 싱크 노드에서는 이 데이터를 지그비(Zigbee) 통신을 이용하여 센서노드에게 전송되고 있다. 그림 1에서는 센서노드와 서버간의 데이터 흐름을 보여주고 있다.

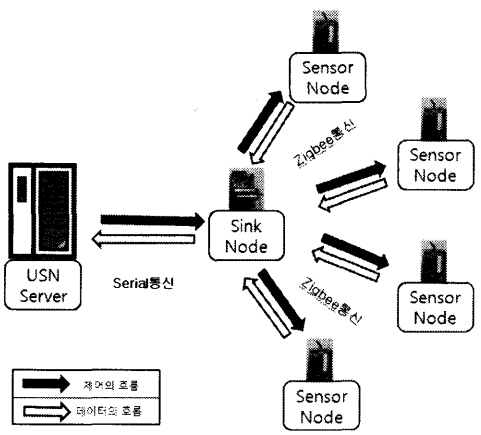


그림 1. 센서 노드와 서버간의 데이터 흐름
Fig 1. Data flow between server and sensor node

각 센서 노드는 노드ID를 할당하여서 서버에서 데이터를 받아들일 때 각 센서노드로부터 들어오는 정보를 노드 ID로 구분할 수 있게 한다.

처음 센서 노드는 수면 모드에서 시작하여서 서버로부터 제어 데이터를 수신하여 데이터안의 Count 값을 비교하여 활동 모드, 수면 모드, PowerOff 모드를 선택하여 센서가 작동할 수 있도록 제어를 할 수가 있다. 활동 모드일 경우 센서가 동작을 시작할 경우 수집하는 데이터는 온도, 습도, 조도, 전력량의 4가지이다. 이렇게 수집된 데이터는 각 노드 ID와 각 데이터를 포함하는 메시지의 형식으로 묶여 진다. 만들어진 메시지는 지그비 통신 방식을 이용하여 서버로 전송된다. 수면 모드일 경우에는 센서안에서 동작하는 타이머를 중지함으로써 데이터의 수집을 중단 시키고 있다. 그러므로 수집되는 정보가 없기 때문에 서버로 보내지는 데이터가 없다. 하지만 RF 통신은 유지하고 있어서 이벤트형식으로 동작하는 Receive가 항상 제어 데이터의 수신을 대기하고 있다. 따라서 수면 모드의 주안점은 데이터 수집함으로써 소모되는 전력량을 필요에 따라서 중지함으로써 전력량의 손실을 줄이는 것이다. PowerOff 모드의 경우에는 타이머뿐만 아니라 MCU(Micro Controller unit)를(을) 수면으로 바꾸어 수면 모드 보다 더 적은 전력으로 동작하도록 한다. 그림 2에서는 센서 노드에서 제어 순서도를 보여주고 있다.

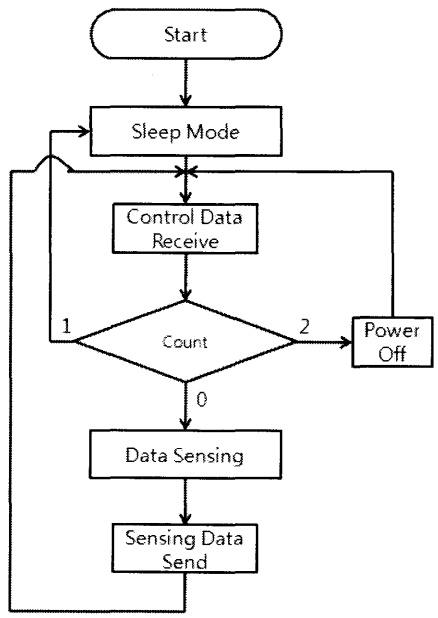


그림 2. 센서 노드에서 제어 순서도
Fig 2. Control flow diagram in sensor node

싱크 노드는 센서로부터 들어오는 데이터를 지그비통신을 이용하여 수신하여 직렬통신을 이용하여 서버로 송신하게 된다. 싱크 노드에서의 수신은 이벤트 형식의 메소드를 사용함으로 사용자로부터 어떠한 명령을 받지 않아도 계속적으로 들어오는 데이터를 수신하게 된다. 서버에서 직렬통신을 위한 포트를 열지 않아도 센서로부터 들어오는 데이터들을 계속해서 수신하고 있다. 사용자에

의해서 포트가 열리면 이벤트형식의 Send에 의하여 센서로부터 수신된 데이터를 시리얼 통신을 이용하여 서버로 송신한다. 싱크 노드에서는 두 개의 수신과 송신 메소드가 이벤트 형식으로 동작하고 있다. 그림 3에서는 서버의 싱크 노드에 대한 순서도를 나타내고 있다.

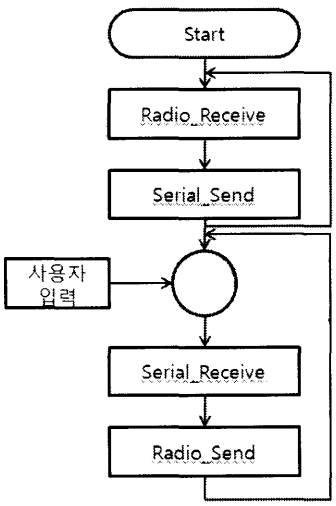


그림 3. 싱크 노드에서의 순서도
Fig 3. Flow diagram of sink node

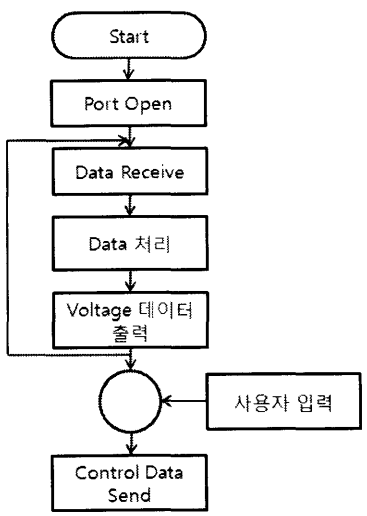


그림 4. 서버에서의 센서 제어 순서도
Fig 4. Control flow diagram of Server

센서 제어 모듈을 구동되면 우선은 센서노드에서 들어오는 데이터들을 수신하기 위하여 포트를 열고 싱크 노드를 통하여 센서로 부터의 데이터를 수신한다. 이때 수신되는 데이터들은 처리를 통하여 사용자가 알 수 있는 값으로 변환되어 화면에 출력된다. 센서 제어 모듈에서는 이 수신되는 값들 중에서 전력량의 값을 사용하는데 이 값을 사용자가 판단하여 센서로 제어 데이터를 전송할 수 있도록 한다. 그림 4에서는 서버의 알고리즘을 나타내고 있다.

Ⅳ. 구현 및 실험

센서 노드와 서버 간에 데이터 및 제어를 하기 위한 구조는 그림 5와 같다. 여기서 센서와 서버는 제어 알고리즘을 갖고 센서 노드를 전력에 따라 활동 모드, 수면 모드, PowerOff 모드로 동작시킨다.

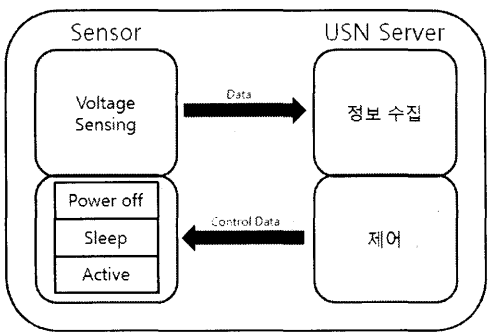


그림 5. 시스템 구조
Fig 5. Architecture

그림 6에서는 사용한 TIP700CM의 전원 특징에 대해 보여주고 있다. 여기서 활동 모드의 경우 1MHz의 동작 성능일 때 300㎂의 전력과 통신을 위해 19.7㎃의 전력이 소모되고 수면모드에서는 1.1㎂의 전력이 소모됨을 알 수가 있다. 수치적으로 계산해 본다면 수면모드에서는 활동모드 일 때 보다 수백 배 이상의 전력이득을 취함을 알 수 있다. 그리고 제어 알고리즘을 사용하는데 있어 모드의 전환 기준이 되는 전력량은 하드웨어의 최소 전력 2.1V과 MCU의 최소 전력인 1.8V로 결정하였으며 1.8V 이하에서는 MCU가 정상적인 동작을 못하여 불확실한 이상 데이터를 보냄을 확인하였다. 본 연구에서 사용한 TIP700CM에 해당 프로그램을 적재하여 측정한 결과 방향에 상관없이 약 23m 이내에서는 정상적인 동작을 하였

으며 24m 이상의 거리에서는 데이터의 송수신이 정상적으로 동작하지 않았다.

TIP700CM (TI MSP430F1611)
- Voltage Range : 2.1V ~ 3.6V (hardware)
  1.8V ~ 3.6V (MCU)
- Active mode : 300μA at 1MHz, 2.2V
- Sleep mode : 1.1μA
- Radio : RX-19.7mA, TX-17.4mA

그림 6. TIP700CM의 전원 특징
Fig 6. Power characteristic of TIP700CM

센서 노드에서의 통신구조는 이벤트 형식의 수신함수와 송신함수가 있어서 물리적으로 켜져 있으면 항상 이벤트가 실행되고 있다고 생각하면 된다. 송신 이벤트의 경우에는 센서가 데이터를 수집하여서 데이터가 발생하였을 때 이벤트가 발생하여서 수집 데이터를 지그비 통신방식을 이용하여 전송한다. 수신 이벤트의 경우에는 싱크노드로부터 지그비 통신방식을 이용하여 들어오는 제어 데이터를 수신한다. 그림 7에서는 센서 노드에서의 통신 구조를 보여주고 있다. 여기서 센서노드는 TinyOS를 운영체제로 사용하고 그 위에서 동작하는 nesC 프로그램으로 작성된다.

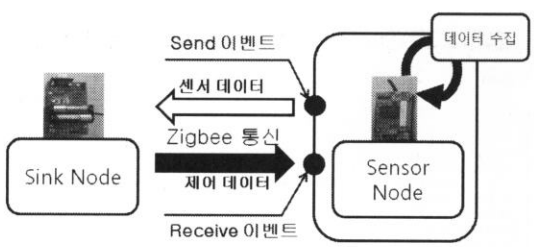


그림 7. 센서 노드에서의 통신 구조
Fig. 7. Communication architecture of sensor node

싱크 노드는 센서 네트워크 와 서버를 연결해 주는 다리 역할을 할 수 있도록 RF통신과 UART 통신을 모두 구현가능 하도록 설계된다. RF통신 방법으로는 지그비를 사용하였고 UART통신에서는 시리얼을 이용하는 RS-232를 사용하여 설계된다. 싱크 노드는 TinyOS를 운영체제로 사용하고 그 위에서 동작하는 nesC 프로그램으로 작성된다.

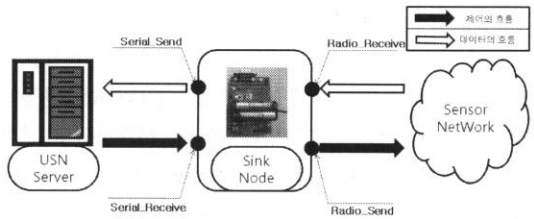


그림 8. 싱크 노드에서의 통신구조
Fig 8. Communication Architecture of Sink node

그림 8에서 보면 싱크 노드에서의 통신 구조를 도시한 그림이다. 여기서 데이터의 흐름과 제어의 흐름이 싱크 노드를 중심으로 어떻게 이동되는가에 대해서 확실히 알 수가 있다. 싱크 노드와 센서 네트워크 사이의 데이터 교환에서 보면 지그비 통신을 이용하기 때문에 Radio_Receive와 Radio_Send가 있고 USN 서버와 싱크 노드의 데이터 교환에서 보면 시리얼 통신을 이용하기 때문에 Serial_Receive와 Serial_Send가 있다. 센서에서 보내지는 데이터는 Radio_Receive를 통하여 싱크 노드로 전해지고 이 데이터는 싱크 노드에서 Serial_Send를 통하여 USN 서버로 전송된다. 그리고 사용자의 입력에 의해 발생하는 제어 데이터는 USN 서버에서 싱크 노드로 Serial_Receive를 통하여 전송되고 이 데이터는 싱크 노드에서 Radio_Send를 통하여 센서 네트워크로 전송된다.

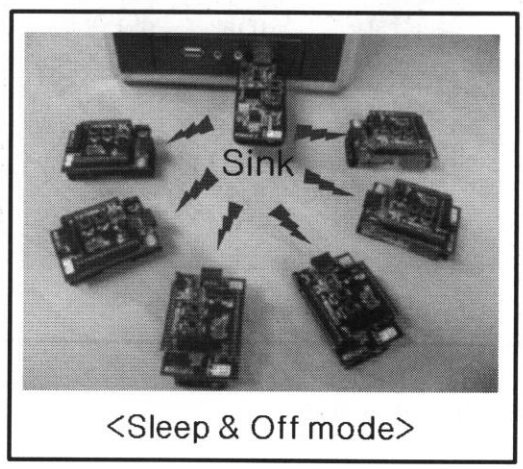


그림 9. 수면 모드와 파워 오프 모드
Fig 9. Sleep mode and power off mode

이러한 구조에서 센서 노드, 싱크 노드, 서버로 전송되는 데이터 안에는 메시지 형식의 데이터가 포함되어 있으며 서버, 싱크 노드, 센서 노드로 전송되는 데이터 안에

는 Counter값 데이터가 포함되어 있다.

센서 노드 제어 모듈을 구현하기 위하여 센서노드와 싱크노드에서 동작하는 운영체제로는 TinyOS를 사용하였으며 센서 보드로는 TIP700SM-a를 사용하였고 센서 보드 위에서 동작하는 센서로는 Hamamatsu사의 S1087, S1087-1과 Sensirion사의 SHT11(Humidity & Temperature)를 사용하고 있다. 센서에서 TinyOS에서 동작하는 프로그램을 작성하기 위해서는 nesC언어를 사용한다.

그림 9는 최초의 수면 모드와 사용자의 제어에 의한 Power Off 모드를 나타내고 있다. 이때 지그비 통신 방식은 중단되지 않고 서로를 연결하고 있다. 그럼으로써 사용자의 제어에 따라서 활성화 시킬 수 있는 것이다.

그림 10에서 센서노드가 사용자의 제어에 의하여 활성화된 상태에서 수집한 데이터들을 지그비 통신을 통하여 USN 서버의 싱크 노드로 전송한다. 센서노드가 데이터를 수집하는 시간 간격을 3초로 한다. 이에 따라 센서노드에서 싱크 노드로 전송하는 시간 간격도 3초로 설정한다.

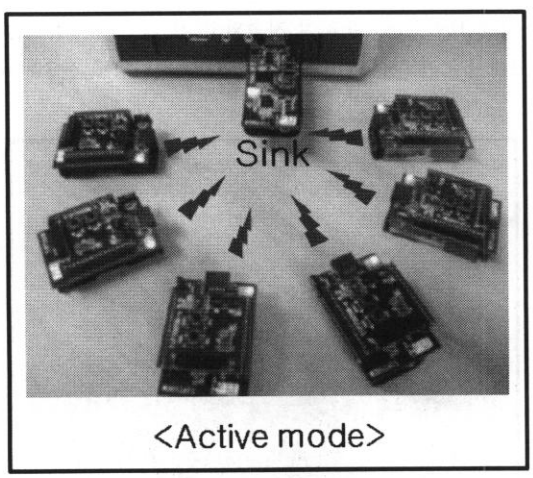


그림 10. 활동 모드
Fig 10. Active mode

센서노드에서 TinyOS 위에서 동작하는 프로그램 중에 센서노드 제어에 관한 코드만 위의 그림으로 표현한다. 위의 그림에서 서버에서 보내진 제어 데이터 내부의 Counter 값을 비교하여 어떠한 동작을 할 것인지를 결정할 수가 있다. 위의 코드에서 오른쪽 코드를 보면 startTimer(), stopTimer(), powerOff() 함수가 보이는데 이 함수가 각각의 모드를 동작시키는 함수이다. 그림 11에서는 센서 노드의 제어 함수를 보여주고 있다.

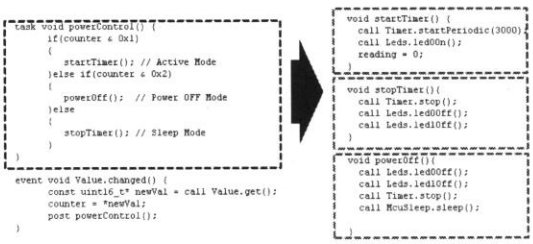


그림 11. 센서노드 제어 함수
Fig 11. Sensor Control Function

USN 서버에서는 서버에서 싱크 노드를 거쳐 수신되는 데이터를 처리하여 사용자가 확인할 수 있게 화면에 표시한다. 그리하여 사용자가 어떠한 기준을 가지고 있으면 그 기준에 맞추어 센서를 제어 할 수 있게 한다. 위의 그림 12에서 보듯이 수신된 데이터들을 출력하고 있는데 출력되는 데이터를 확인해 보면 각 노드의 노드ID와 온도, 습도, 조도, 전력량 값들이 있음을 알 수 있다. 이번 연구에서는 수신되는 데이터들 중에서도 전력량 값을 사용자가 판단하여 각각의 센서를 제어할 수 있도록 한다. 사용자의 제어를 통하여 센서는 소비되는 전력을 줄일 수 있다.

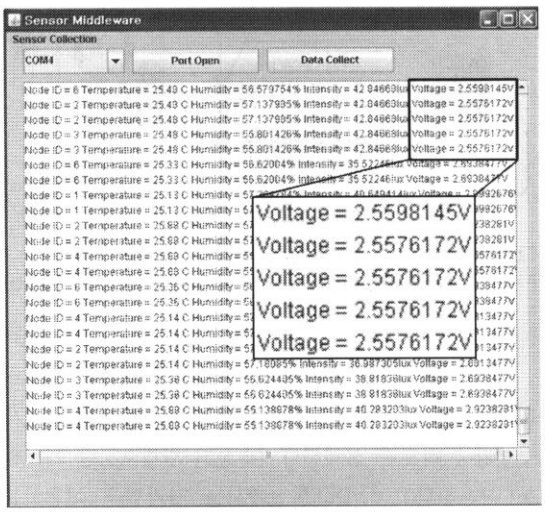


그림 12. 데이터 수신 및 도시
Fig 12. Data receive and display

V. 결 론

무선 센서 네트워크를 구성하는 센서 노드들의 외부로부터 무한정의 에너지를 공급받는 것이 아니기 때문에 에너지 관리 및 센서 노드 제어에 많은 관심이 집중되어 있다. 본 논문에서는 센서 노드의 전력에 따라 활동 모드, 수면 모드, PowerOff 모드로 동작하는 센서 노드 제어 알고리즘을 제시한다. 그리고 TinyOS 기반의 센서 노드, 서버 및 싱크 노드에서 제어 모듈을 설계 및 구현한다. 이를 위하여 센서 노드의 내부 상태 정보로 활용할 수 있는 전력량을 수집하여 전송하는 프로그램과 제어 모듈로부터 보내어지는 제어 데이터에 의하여 제어 할 수 있도록 프로그램 내부의 조건을 변경한다. 싱크노드는 무선 센서 네트워크와 서버의 제어 모듈 간의 다리 역할을 할 수 있도록 하였으며 서버의 제어 모듈은 무선 센서 네트워크로부터 수신하는 데이터를 도시하고 이를 바탕으로 사용자가 판단할 수 있게 도시 모듈을 개발 하였으며, 사용자가 무선 센서 네트워크로 제어 데이터를 보낼 수 있도록 센서 노드 제어 모듈을 개발할 수 있다.

참 고 문 헌

[1] 이상훈 , 문승진,"TinyDB와 라인트레이서를 활용한 TinyOS기반의 센서 데이터 처리 모듈 설계 및 구현", 한국통신학회, 한국통신학회논문지 제31권 제10B호, 2006. 10, pp. 883 ~ 890

[2] 송준근 , 마평수 , 박승민,"센서 네트워크용 초소형 OS",한국통신학회, 한국통신학회지 (정보와통신) 제24권 제7호, 2007. 7, pp. 26 ~ 35

[3] 최용식, 김성선, 신승호,"유비쿼터스 환경에서 센서 노드의 관리와 망 구성을 위한 RFID 미들웨어 프로토콜에 관한 연구", 한국컴퓨터정보학회, 한국컴퓨터정보학회 논문지, 2007. 7, pp. 155 ~ 163

[4] 김영성, 김영환, 석정봉,"TinyOS 메시지 길이에 따른 에너지 절약 연구",한국정보과학회, 한국정보과학회 학술발표논문집 한국정보과학회 2006 한국컴퓨터종합학술대회 논문집(D), 2006. 6, pp. 343 ~ 345

[5] David Patnode, Joseph Dunne, Aleksander Malinowski, Donald Schertz, "WISENET-TinyOS Based Wiress Network of Sensors"

[6] Wong Kiing-Ing, "A Light-weighted, Low-cost and Wiress ECG Monitor Design based on TinyOS Operating System" 6th International Special Topic Conference on ITAR, 2007

[7] Masatuki Nakamura, Atsushi Sakurai, Shizuo Furubo, Hiroshi Ban, "Collaborative Processing in Sensor/Actuator Networks for Environment Control"

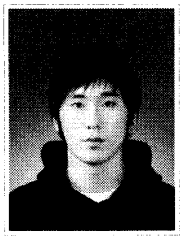
[8] Jaspal S. Sandhu, Alice M.Agogino, Adrial K. Agogino, "Wirelss Sensor Networks for Commercial Lighting Control"

[9] TinyOS , www.TinyOS.net

Acknowledgement : This work was supported by the Korea Research Foundation Grant (KRF -2006-311-D00147)

저자 소개

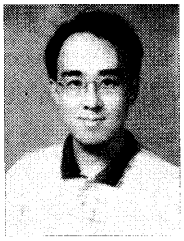
부 준 필(정회원)



- 2008년 현재 제주대학교 컴퓨터공학과 학사.

<주관심분야 : TinyOS, 네트워크, RFID, 센서>

김 도 현(정회원)



- 1988년 경북대학교 전자공학과(공학석사)
- 1990년 경북대학교 전자공학과(공학석사)
- 2000년 경북대학교 전자공학과(공학박사)
- 1990년 ~ 1995년 국방과학연구소 연구원
- 1999년 ~ 2004년 천안대학교 조교수
- 2004년 ~ 현재 제주대학교 부교수

<주관심분야 : 유비쿼터스 서비스, 센서 네트워크, 이동 컴퓨팅>

양 현 규(정회원)

- 2008년 제주대학교 컴퓨터공학과 학사 졸업.
- 2008년 현재 NHN 사원.

<주관심분야 : TinyOS, Java, Web, 검색>