

전력소모량 및 실행시간 추정이 가능한 센서 네트워크 시뮬레이터의 개발

김방현^{1*} · 김태규¹ · 정용덕¹ · 김종현¹

Development of Sensor Network Simulator for Estimating Power Consumption and Execution Time

Bang-Hyun Kim · Tae-Kyu Kim · Yong-Doc Jung · Jong-Hyun Kim

ABSTRACT

Sensor network, that is an infrastructure of ubiquitous computing, consists of a number of sensor nodes of which hardware is very small. The network topology and routing scheme of the network should be determined according to its purpose, and its hardware and software may have to be changed as needed from time to time. Thus, the sensor network simulator being capable of verifying its behavior and estimating performance is required for better design.

Sensor network simulators currently existing have been developed for specific hardwares or operating systems, so that they can only be used for such systems and do not provide any means to estimate the amount of power consumption and program execution time which are major issues for system design. In this study, we develop the sensor network simulator that can be used to design and verify various sensor networks without regarding to types of applications or operating systems, and also has the capability of predicting the amount of power consumption and program execution time. For this purpose, the simulator is developed by using machine instruction-level discrete-event simulation scheme. As a result, the simulator can be used to analyze program execution timings and related system behaviors in the actual sensor nodes in detail. Instruction traces used as workload for simulations are executable images produced by the cross-compiler for ATmega128L microcontroller.

Key words : sensor network simulator, ubiquitous computing, power consumption, execution time

요 약

유비쿼터스 컴퓨팅의 인프라가 되는 센서 네트워크는 매우 작은 하드웨어로 이루어지는 많은 수의 센서 노드들로 구성된다. 이 네트워크의 토폴로지와 라우팅 방식은 그 목적에 따라 결정되어야 하며, 하드웨어 및 소프트웨어도 필요한 경우에는 변경되어야 한다. 따라서 그러한 네트워크를 최적으로 설계하기 위해서는 시스템 동작을 확인하고 성능을 예측할 수 있는 센서 네트워크 시뮬레이터가 필요하다. 현존하는 몇몇 센서 네트워크 시뮬레이터들은 특정 하드웨어나 운영체제에 맞추어 개발되었기 때문에, 그러한 특정 시스템들을 위해서만 사용될 수 있다. 그리고 시스템 설계 상의 주요 이슈가 되는 전력소모량 및 프로그램 실행 시간을 추정하기 위한 어떤 수단도 지원하지 못하고 있다. 이 연구에서는 응용이나 운영체제의 종류에 상관없이 다양한 센서 네트워크들의 설계 및 검증에 사용될 수 있는 시뮬레이터를 개발하였다. 이를 위하여, 시뮬레이터는 기계어-레벨 이산-사건 시뮬레이션 방식을 사용하였다. 따라서 이 시뮬레이터는 프로그램 실행 타이밍 뿐 아니라 실제 센서 노드 내부의 동작들도 세부적으로 분석하는 데도 사용될 수 있다. 이 연구에서 시뮬레이션의 작업부하인 명령어 트race으로는 ATmega128L 마이크로컨트롤러용 크로스컴파일러에 의해 생성된 실행 이미지를 사용하였다.

주요어 : 센서 네트워크, 시뮬레이터, 유비쿼터스 컴퓨팅, 전력소모량, 실행 시간

1. 서 론

미국의 'Business Week'지는 미래의 기술이라는 특집

2005년 12월 30일 접수, 2006년 2월 27일 채택

¹⁾ 연세대학교 문리대학 컴퓨터정보통신공학부

주 저 자 : 김방현

교신저자 : 김종현

E-mail; jhkim34@yonsei.ac.kr

기사에서 비즈니스 관점에서 주목해야 할 네 가지 기술에 센서 네트워크(sensor network)를 포함시켰고^[4], MIT의 'Technology Review'는 세상을 바꿀 10가지 미래 기술들 중의 하나로 "Brain-Wireless Sensor Networks"를 제시하기도 하였다^[8]. 이와 같이 미래의 주요 기술로 주목받고 있는 센서 네트워크는, 최근 일상생활에 산재한 사물과 물리적 대상이 점차 정보의 대상으로 확대됨에 따라

인간과 컴퓨터 및 사물들이 유기적으로 연계되어 다양하고 편리한 새로운 서비스를 제공해 주는 유비쿼터스 컴퓨팅(ubiquitous computing)의 핵심 인프라가 될 것으로 예상되고 있다.

센서 네트워크는 제한된 전력량을 가진 많은 수의 작은 센서 노드들로 구성되어 있고, 센서 노드들 간의 통신은 무선으로 이루어진다. 이 네트워크는 구축 목적에 따라 네트워크 토폴로지 및 라우팅 방식이 결정되어야 하며, 이와 더불어 센서 노드의 하드웨어와 소프트웨어도 필요에 따라 다양하게 변경되어야 한다. 따라서 센서 네트워크가 구현되기 전에 시스템 동작과 성능을 예측할 수 있고 소프트웨어 개발 환경도 제공해주는 시뮬레이터가 필요하다.

센서 네트워크 시뮬레이터에 관한 연구로는 U.C. Berkeley 대학 중심의 NEST(network embedded software technology) 프로젝트^[12]에서 개발된 TOSSIM (TinyOS simulator), Prowler(probabilistic wireless network simulator), Siesta(simple NEST application simulator), Ashut(acoustic simulator for urban terrain), 그리고 Rmase(routing modeling application simulation environment) 등이 있으며, 이들은 NEST 프로젝트에서 개발된 TinyOS가 탑재되어 있는 하드웨어 플랫폼(mote 또는 sensor node) 기반에서 동작하는 시뮬레이터들이다. TOSSIM^[13]은 센서 노드에 탑재된 TinyOS와 응용프로그램의 동작과 상호작용을 시뮬레이션 하는 운영체제 시뮬레이터이며, Prowler^[7]는 무선 네트워크 알고리즘을 평가할 수 있는 네트워크 시뮬레이터이고, Siesta^[3]는 NEST 응용 프로그램 및 미들웨어의 기능을 검증할 수 있는 미들웨어 시뮬레이터이다. 그리고 Ashut^[12]는 주어진 환경에서 직접 전달되는 음향과 반사 전달되는 음향의 도착 시간을 계산할 수 있는 시뮬레이터이고, Rmase^[12]는 네트워크 토폴로지 및 응용 시나리오를 생성하여 센서 네트워크의 라우팅 기능을 평가하는 Prowler 기반의 라우팅 시뮬레이터이다.

그러나 이 시뮬레이터들은 TinyOS^[14]에서만 동작하며, 전력소모량 및 실행시간에 대한 부분은 고려되지 않았다. 특히 TOSSIM을 구동하기 위해서는 TinyOS에서 TOSSIM용으로 컴파일된 작업부하가 필요하고, Prowler의 경우에는 MATLAB 소프트웨어가 필요하다. 또한 Siesta는 응용 소프트웨어가 시뮬레이션을 위한 특정 API 기능을 포함하지 않을 경우에는 시뮬레이션이 수행되지 않는다.

따라서 본 연구에서는 특정한 프로그램이나 운영체제

에 의존하지 않으면서 다양한 센서 네트워크 환경을 설계 및 검증할 수 있고, 더불어 전력소모량과 실행시간 추정도 할 수 있는 범용 센서 네트워크 시뮬레이터를 개발하는 것을 목표로 하였다. 본 연구에서 개발한 기계명령어-레벨의 센서 네트워크 시뮬레이터(machine instruction-level sensor network simulator: 이하 MISS라 함)는 이산-사건 시뮬레이션(discrete-event simulation) 기법을 이용함으로써 센서 노드 내부 모듈 및 센서 노드들 간 동작의 시간 동기화를 가능하게 하였다. 또한 기계명령어-레벨의 시뮬레이션 작업부하를 사용하여 시스템 클록 수준으로 시뮬레이션의 정밀도를 높임으로써 전력소모량 및 실행시간 추정을 가능하게 하였다.

본 논문의 구성은 2장에서 시뮬레이터의 설계 방법에 대하여 설명하였고, 3장에서는 시뮬레이션을 실제 사용한 예를 통하여 검증한 결과를 보여주고 있다. 마지막으로 4장에서는 연구에 대한 전반적인 결론을 요약하였다.

2. 시뮬레이터의 설계 및 구현

MISS는 기계명령어-레벨의 시뮬레이터이기 때문에 실제 구현에 있어서는 센서 보드의 하드웨어 특성에 의존성을 갖는다. 따라서 본 연구에서는 현재 센서 네트워크 연구에서 가장 널리 사용되고 있는 MICAz 형식의 센서 보드인 CrossBow 사의 MPR2400 보드^[6]와 옥타컴 사의 NANO-24 보드^[10]를 기본 모델로 하여 MISS를 구현하였다. MICAz 형식의 센서 보드에는 Atmel 사의 ATmega128L 마이크로컨트롤러^[1]와 ChipCon 사의 CC2420 RF 트랜시버^[5]가 장착되어 있다. 그리고 MISS의 작업부하인 명령어 트래이스로는 ATmega128L 마이크로컨트롤러용으로 크로스 컴파일되어 실제 센서 보드에 적재되는 실행이미지를 사용한다.

2.1 시뮬레이터의 구조

MISS는 <그림 1>에서 보는 바와 같이 센서 네트워크 시뮬레이션을 수행하는 시뮬레이션 엔진(Simulation Engine)과 인터페이스 역할을 하는 GUI로 구성된다. 셸 명령어 기반으로 동작하는 시뮬레이션 엔진과 GUI는 서로 독립적인 모듈이며, 이들 사이의 연동은 TCP/IP 통신을 이용한 메시지 교환에 따라 이루어진다. GUI에서 시뮬레이션 엔진으로 전달되는 메시지의 내용은 셸 명령어이고, 반대로 전달되는 메시지의 내용은 LCD의 변화나 RF 통신 발생 등을 포함하는 시뮬레이션 진행 상황이다. 이러한 구조이기 때문에 MISS는 시뮬레이션 엔진만으로도

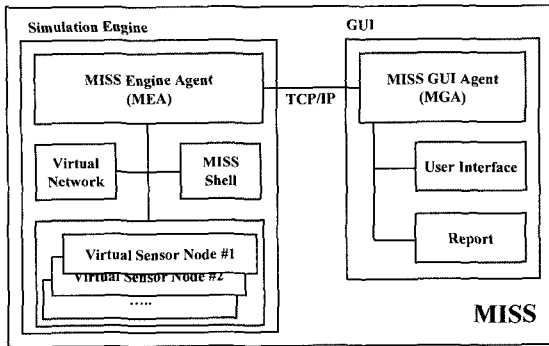


그림 1. MISS의 구조

시뮬레이션을 할 수 있으며, GUI와 연동할 경우에는 사용자가 시뮬레이션 상황을 한눈에 파악할 수 있고 MISS를 손쉽게 제어할 수 있는 그래픽 환경의 인터페이스를 제공한다.

시뮬레이션 엔진은 각각의 센서 보드를 시뮬레이션 하는 가상 센서 노드(virtual sensor node) 모듈, 센서 보드 사이의 통신을 시뮬레이션 하는 가상 네트워크(virtual network) 모듈, 사용자가 입력한 셸 명령어를 해석하여 처리하는 MISS 셸 모듈, 그리고 GUI와 연결 기능을 하는 MISS 엔진 에이전트(MEA) 모듈로 구성된다. 그리고 GUI는 시뮬레이션 엔진의 MEA와 연결 기능을 하는 MISS GUI 에이전트(MGA) 모듈, 사용자가 입출력을 통하여 MISS를 제어할 수 있도록 하는 사용자 인터페이스(user interface) 모듈, 그리고 시뮬레이션 결과를 생성하는 리포트(report) 모듈로 구성된다.

시뮬레이션 엔진은 ANSI C로 구현되었고, GUI는 Java로 구현되었다. 그리고 시뮬레이션 엔진에서 이산-사건 시뮬레이션을 위한 라이브러리는 ANSI C로 구현된 `smp1`^[9]을 사용한다. 따라서 MISS는 ANSI C와 Java를 지원하는 모든 운영체제(MS 윈도우, 유닉스, 리눅스, 솔라리스 등)에서 사용이 가능하다.

2.2 기계명령어-레벨의 시뮬레이션

MISS가 사용하는 시뮬레이션 작업 부하는 실제 센서 보드에 적재되는 실행 이미지로서, ATmega128L 마이크로컨트롤러용으로 크로스 컴파일되어 실제 센서 보드에 적재되는 인텔 헥스-레코드 형식(Intel's hex-record format: .hex), 롬 이미지 형식(ROM image format: .rom), 그리고 S-레코드 형식(S-record format: .srec)의 파일을 사용한다. 기계명령어-레벨의 시뮬레이션 방법은 이 실행 이미지를 시뮬레이션 엔진의 가상 센서 노드에 적재한다

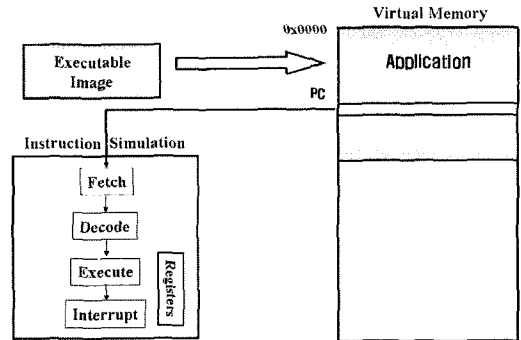


그림 2. 기계명령어-레벨의 시뮬레이션

음에, 실제 센서 노드에서 프로세서 명령어가 실행되는 방법과 거의 동일하게 이루어진다. 즉, 시뮬레이션은 실제 센서 노드에서와 같이 가상 플래시 메모리에서 현재 프로그램 카운터(PC)가 가리키는 위치의 기계명령어를 인출하여, 기계명령어의 비트 패턴을 분석하고 실행함으로써 이루어진다. <그림 2>는 <그림 1>의 가상 센서 노드에서 프로세서에 의한 기계명령어 실행 동작을 시뮬레이션 하는 개념도를 보여주고 있다.

2.3 이산-사건 시뮬레이션

명령어의 인출과 실행은 이산-사건 시뮬레이션 기법을 이용하여 사건(event) 단위로 처리된다. 이산-사건 시뮬레이션은 모든 시간의 진행이 사건의 발생에 의해서 이루어지게 하는 방법이다. 즉, 사건이 발생하면 시간이 진행되는 것이다. 그리고 이산-사건 시스템의 상태(state)는 사건 발생에 의해 불규칙한 시간 간격으로 천이되며, 구체적인 물리적 상황을 표현하는 상징적 의미를 가진다. 다시 말하면, 이산-사건 시뮬레이션은 모델의 상태 변화가 일어나는 경우에만 시간을 진행시키는 방법이다^[2]. 본 연구에서는 센서 노드 내부 모듈 및 센서 노드들 간 동작의 시간 동기화를 위하여 이산-사건 시뮬레이션 방식을 이용하였다. 특히 본 연구의 대상인 센서 네트워크는 규모가 커지는 경우에는 노드의 수가 수천 개에 이를 수도 있기 때문에 이러한 시간 동기화는 매우 중요하다.

<그림 3>은 ATmega128L 마이크로컨트롤러의 CPU 코어에 의한 명령어 실행, 인터럽트 처리, 타이머 동작, EEPROM 쓰기, SPI 동작, 그리고 SPI 동작에 따른 CC2420 RF 트랜시버의 RF 통신 등을 시뮬레이션 하기 위한 사건 루틴들 및 그들 간의 스케줄링 관계를 보여주고 있다. 이 사건 루틴들은 동작의 순차성에 따라 적절한 사건 시간(event time) 간격으로 관련 사건을 스케줄링 하

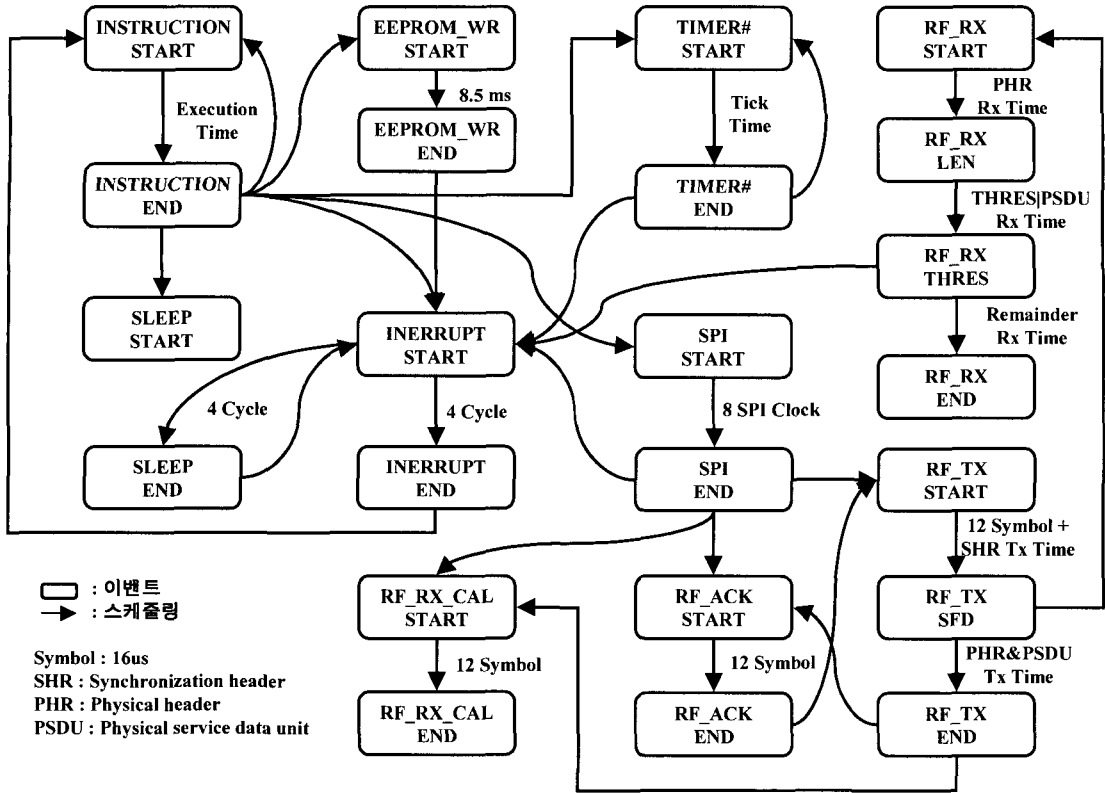


그림 3. 사건들 간의 스케줄링 관계

게 된다. 스케줄링 된 사건들은 시간 스탬프(time stamp)와 함께 사건 리스트(event list)에 저장되며, 순차적으로 호출되어 수행된다. <그림 3>의 사건 루틴들은 <그림 1>의 가상 센서 노드들과 가상 네트워크 모듈들의 상호동작 및 <그림 2>의 기계명령어 실행 동작을 시뮬레이션하기 위한 것들이다.

주요 사건 루틴들에 대하여 좀 더 상세히 설명하면 다음과 같다.

- **INSTRUCTION_START** 루틴: AVR 마이크로컨트롤러의 CPU 코어가 명령어를 인출하고 실행을 준비하는 루틴으로서, 가상 센서 노드의 플래시 메모리에 적재되어 있는 실행 이미지에서 PC가 가리키는 위치의 명령어 하나를 읽어서 해독하고 해당 명령어의 실행 시간 후에 **INSTRUCTION_END** 사건이 수행되도록 스케줄 한다. 만약 AVR 마이크로컨트롤러의 시스템 클럭이 8 Mhz이고, 명령어가 1 사이클의 실행시간을 갖는 “add” 명령어라면 125 ns 후에 **INSTRUCTION_END** 사건이 수행되도록 스케

줄 한다.

- **INSTRUCTION_END** 루틴: 명령어의 실행이 실제 수행되는 루틴으로서, 해당 명령어의 실행 결과에 따라 변경된 상태를 가상 센서 노드의 레지스터에 반영한다. 이 루틴에서는 명령어의 실행 결과나 센서 노드의 상태에 따라서 여섯 가지의 다른 사건들을 스케줄 한다.
- **RF_TX_START** 루틴: RF 송신을 시작하는 루틴으로서, 12 심벌(symbol) 시간 지연과 SHR(synchronization header) 전송 시간 후에 **RF_TX_SFD** 사건이 발생하도록 스케줄 한다. CC2420에서 1 심벌은 16 μ s 이고, 1 비트의 전송 시간은 4 μ s 이다.
- **RF_TX_SFD** 루틴: 송신 노드에서 SHR까지 송신한 시점에 호출되어 실행되는 루틴으로서, MISS에서는 이 시점에서 송신 노드의 송신 버퍼에 있는 패킷 내용을 수신 노드의 수신 버퍼로 복사한다. <그림 4>에서 보는 바와 같이 패킷의 내용이 수신되는 시간에 따라 수신 노드의 CC2420 핀에 변화가 발생하기 때문에, 이 때 RF 수신 동작을 수행하기 위하여 수

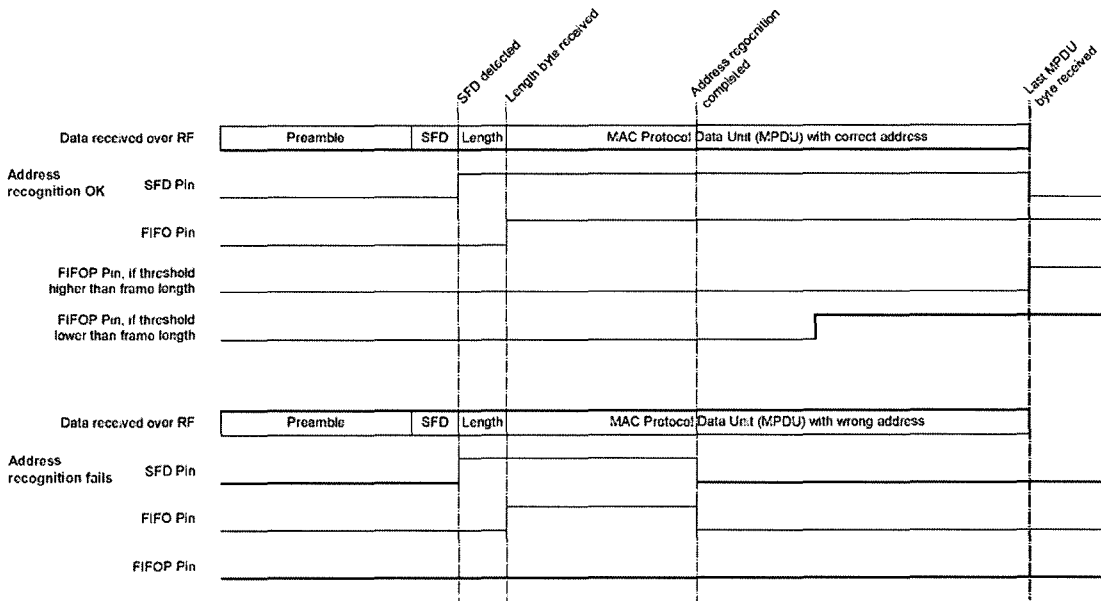


그림 4. CC2420에서의 RF 수신 동작

신 노드에 RF_RX_START 사건을 스케줄 한다. 이와 동시에 송신 노드에는 송신 완료 동작을 수행하기 위한 RF_TX_END 사건이 송신 패킷의 PHR (physical header)과 PSDU (physical service data unit)의 전송이 끝나는 시간 후에 발생하도록 스케줄 한다.

- RF_TX_END 루틴: RF 송신이 완료되었을 때 발생하는 동작을 시뮬레이션 하는 루틴으로서, 송신을 완료한 송신 노드가 수신 대기 모드로 전환하도록 RF_RX_CAL_START 사건을 스케줄 한다.

3. 시뮬레이션 결과

<그림 5>는 TinyOS[11]에 포함되어 있는“CntToRfm.srec”과 “RfmToLeds.srec” 예제를 MISS에서 시뮬레이션 한 결과를 보여준다. 센서 노드의 종류는 MICAz 형식을 갖는 CrossBow 사의 MPR2400 보드[6]로 설정하였고, AVR 마이크로컨트롤러의 시스템 클록은 7.37 MHz으로 설정하였다. “CntToRfm.srec”은 250ms 간격으로 증가되는 카운트 값을 RF 통신을 이용하여 주변 노드로 전송하는 예제이고, “RfmToLeds.srec”은 수신된 카운터 값의 하위 세 비트를 LED로 디스플레이 하는 예제이다. 실행 화면에서 보는 바와 같이 예제가 정확하게 시뮬

```

C:\D:\Project\Wmias\Wmias.exe
$ nodetype MICAz
[Msg] Current node type is MICAz.
$ add 2
[Msg] ID of created node : 0 1
$ load CntToRfm.srec
[Msg] <CntToRfm.srec> loaded in node 0's flash memory.
$ nch 1
[Msg] Current node is changed : 0 => 1
$ load RfmToLeds.srec
[Msg] <RfmToLeds.srec> loaded in node 1's flash memory.
$ siminit
[Msg] Environment values is loaded from 'miss.cfg'
[Msg] Initialize simulation parameter.
$ simrun 1000
[Msg] Node-0 : RP-Tx, Time= 249.371474 ns, TxCnt=14, FREQ=2405
[Msg] Node-1 : RP-Rx (1 < 0), Time= 249.755474 ns
    Packet : 0f 01 08 01 ff ff ff ff 04 7d 01 00 01 00
[Msg] Node-1 : LED = _R, Time= 249.958110 ns
[Msg] Node-0 : RP-Tx, Time= 489.120825 ns, TxCnt=14, FREQ=2405
[Msg] Node-1 : RP-Rx (1 < 0), Time= 489.512825 ns
    Packet : 0f 01 08 02 ff ff ff ff 04 7d 02 00 01 00
[Msg] Node-1 : LED = __, Time= 489.715733 ns
[Msg] Node-1 : LED = _G, Time= 489.717767 ns
[Msg] Node-0 : RP-Tx, Time= 732.003716 ns, TxCnt=14, FREQ=2405
[Msg] Node-1 : RP-Rx (1 < 0), Time= 732.387716 ns
    Packet : 0f 01 08 03 ff ff ff ff 04 7d 03 00 01 00
[Msg] Node-1 : LED = __GR, Time= 732.590353 ns
    
```

그림 5. 실행시간 추정 결과 화면

레이션 되는 것을 확인할 수 있으며, 이것은 MISS가 정밀한 실행시간 추정이 가능하다는 것을 보여준다.

이 예제들을 1분 동안 시뮬레이션 하여 평균 전력소모량을 추정한 결과가 <그림 6>에서 나타나 있다. 소모 전력량은 CrossBow사에서 제공하는 전력소모량[6]을 이용하여 계산하였다. 결과에서 보면, MISS가 정밀도 높은 시뮬레이션을 수행함에도 불구하고 시뮬레이션 시간이 약 6.5초 밖에 걸리지 않았다는 것을 알 수 있다. 이것은 대기 모드가 대부분의 시간을 차지하는 센서 네트워크의 특성을 최대한 활용할 수 있는 이산-사건 시뮬레이션 방법

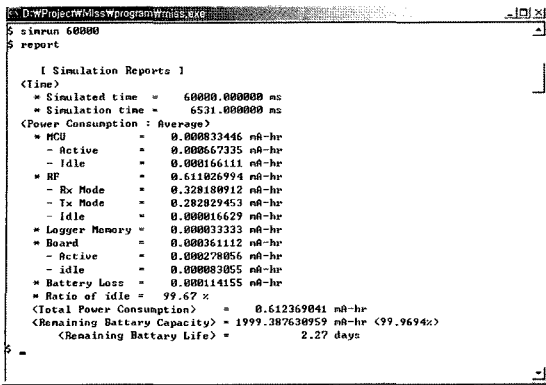


그림 6. 전력소모량 추정 결과 화면

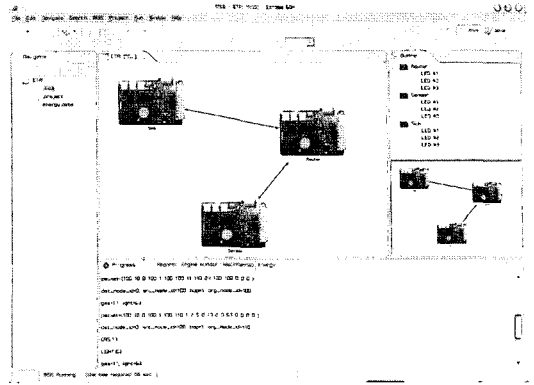


그림 7. GUI 연동 화면

을 사용하였기 때문이다. 전력소모량 추정 결과에서 보듯이 예제를 계속 실행시켰을 경우에 2.27 일 동안 동작이 가능하다는 것을 알 수 있다.

<그림 7>은 한국전자통신연구원(ETRI)의 NANO-Q⁺[11]에 포함되어 있는 ‘detect_net4’ 예제를 GUI 환경과 연동하여 실행한 것이다. 센서 노드의 종류는 MICAz 형식을 갖는 옥타컴 사의 NANO-24 보드[13]로 설정하였고, AVR 마이크로컨트롤러의 시스템 클록은 8 MHz로 설정하였다. 이 예제는 802.15.4 MAC 프로토콜을 사용하여 2초 간격으로 센싱된 가스와 조도 값이, 라우팅 노드를 거쳐 싱크 노드로 전달되어 시리얼 모니터를 통해 출력되는 예제이다. 센싱을 하는 노드에는 “sensor.rom”이 적재되고, 라우팅을 하는 노드에는 “router.rom”이 적재되며, 싱크 노드에는 “sink.rom”이 적재된다. 실행 화면에서 보는 바와 같이 GUI를 통해 사용자가 시뮬레이션 상황을 쉽게 알 수 있으며, 시뮬레이션 동작이 정상적으로 이루어짐을 할 수 있다.

MISS는 <그림 8>과 같은 디버깅 기능을 제공하여, 센서 네트워크 응용 프로그램 개발도구로 활용될 수 있다. 이는 MISS가 높은 정밀도를 갖는 시뮬레이터를 나타내는 하나의 예이다. 여기서 'n'은 하나의 기계명령어를 실행한 후에 ATmega128L 마이크로컨트롤러 내부 레지스터들의 내용을 포함한 여러 정보를 보여주는 쉘 명령어이고, 'ci'는 CC2420 내부 레지스터들의 내용과 RF 모듈에 관련된 정보들을 보여주는 쉘 명령어이다. 이렇게 시뮬레이션의 제어와 디버깅을 위해 MISS는 40개의 쉘 명령어를 제공한다. 특히 마이크로컨트롤러와 RF 모듈에 대한 세밀한 정보를 동시에 사용자에게 제공하는 센서 네트워크 시뮬레이터는 현재 MISS가 유일하다.

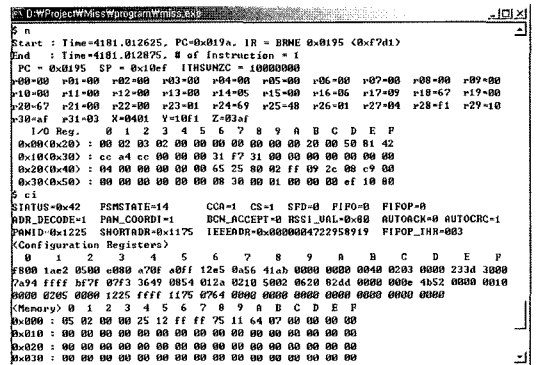


그림 8. 프로그램 디버깅 화면

4. 결론 및 향후 연구 과제

본 연구에서 개발한 센서 네트워크 시뮬레이터(MISS)는 센서 노드 하드웨어나 소프트웨어 개발 이전에 센서 네트워크 환경을 설계하고 검증할 수 있는 유용한 도구이다. 특히 MISS의 구현에서는 기계명령어-레벨의 이산-사건 시뮬레이션 기법을 이용하였기 때문에, 라이브러리 함수 레벨을 이용하는 기존의 센서 네트워크 시뮬레이터들에 비하여 더 높은 정밀도의 시뮬레이션이 가능해졌다. 즉, 프로그램 실행을 포함한 시스템 동작 시간을 실제 하드웨어에서의 시간에 근접하게 추정할 수 있고, 전력소모량 추정 기능을 제공하기 때문에 센서 네트워크의 수명도 예측할 수 있게 해준다. 또한 MISS는 실제 센서 노드에 적재되는 실행이미지를 작업부하로 사용하기 때문에, 운영체제나 라이브러리와 같은 소프트웨어에 독립적인 범

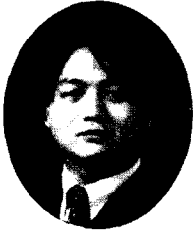
용 시뮬레이터이다. 이것은 기존의 센서 네트워크 시뮬레이터들이 가지고 있는 가장 큰 문제점인 소프트웨어 의존성 문제를 해결하였다는 것을 의미한다.

MISS는 현재 널리 사용되는 MICAz 형식의 센서 보드를 기본 모델로 하여 구현되었지만, 다른 형식의 센서 보드에 대해서도 본 연구에서 사용한 방법으로 가상 센서 노드를 구현함으로써 시뮬레이터에서 사용 가능한 센서 보드의 종류에 대한 확장이 가능하다. 또한 LCD와 같이 센서 보드에 연결 가능한 다양한 주변 장치들의 동작도 마이크로컨트롤러의 입출력 핀의 시뮬레이션 부분에 해당 기능을 용이하게 추가할 수 있기 때문에 다양한 실행 결과를 사용자에게 제공할 수 있다.

그러나 실제 센서 네트워크에서 센서 노드의 수가 수백 혹은 수천 개가 되는 경우도 있으며, 그러한 대규모 센서 네트워크를 단일 컴퓨터에서 시뮬레이션 하는 경우에는 시뮬레이션 시간이 매우 길어지기 때문에, 그 시간을 단축시키기 위하여 병렬 시뮬레이션 방법을 찾는 것이 앞으로 수행되어야 할 과제이다.

참 고 문 헌

1. Atmel, ATmega128(L) Complete, May 2004.
2. A. Alan B. Pritsker, Jean J. O'Reilly, David K. LaVal, Simulation with Visual SLAM and AweSim, John Wiley & Sons, March 1999.
3. A. Ledeczi, M. Maroti, I. Bartok, "Simple Nest Application Simulator," Draft, Institute for Software Integrated Systems, Vanderbilt University, October 2001.
4. Business week, Aug. 2003.
5. ChipCon, SmartRF CC2420 Preliminary Datasheet 1.2, June 2004.
6. CrossBow, "MPR/MIB Users Manual," April 2005.
7. G. Simon, P. Volgyesi, M. Maroti, A. Ledeczi, "Simulation-based Optimization of Communication Protocols for Large-scale Wireless Sensor Networks," Proc. of the IEEE Aerospace Conference, March 2003.
8. MIT Enterprise Technical Review, Feb. 2003.
9. M. H. MacDougall, Simulating Computer Systems, MIT Press, July 1987.
10. NANO-24, <http://www.octacomm.net>.
11. NANO-Q+, <http://qplus.or.kr>.
12. NEST Project, <http://webs.cs.berkeley.edu>.
13. P. Levis, N. Lee, M. Welsh, D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," Proc. of 1st ACM Conference on Embedded Networked Sensor Systems, 2003.
14. TinyOS, <http://www.tinyos.net>



김 방 현 (legnamai@chol.com)

1996 연세대학교 전산학과 학사
2001 연세대학교 전산학과 석사
2001 ~ 현재 연세대학교 컴퓨터정보통신공학부 박사과정

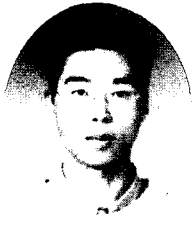
관심분야 : 센서 네트워크, 임베디드시스템, 병렬처리, 시뮬레이션



김 태 규 (windcry@korea.com)

2002 연세대학교 전산학과 학사
2004 연세대학교 전산학과 석사
2004 ~ 현재 연세대학교 컴퓨터정보통신공학부 박사과정

관심분야 : 센서 네트워크, 임베디드시스템, 병렬처리, 정보보안



정 용 덕 (sofe21@gmail.com)

2004 연세대학교 전산학과 학사
2004 ~ 현재 연세대학교 컴퓨터정보통신공학부 석사과정

관심분야 : 센서 네트워크, 임베디드시스템



김 종 현 (jhkim34@yonsei.ac.kr)

1976 연세대학교 전기공학과 학사
1981 연세대학교 전기공학과 석사
1988 Arizona State University 전기 및 컴퓨터공학과 박사
1976 ~ 1982 국방과학연구소 연구원
1988 ~ 1990 한국전자통신연구소 실장
1990 ~ 현재 연세대학교 컴퓨터정보통신공학부 교수
1996 Oregon State University 전산학과 방문교수
2003 Florida State University 전산학과 방문교수

관심분야 : 컴퓨터구조, 병렬처리, 시뮬레이션, 센서 네트워크