국방 센서 네트워크를 위한 초소형 운영체제

한국전자통신연구원 마평수

1. 서 론

센서 네트워크 기술은 군사, 물류, 교통, 환경 감시, 방범, 방재, 홈 오토메이션 등 다양한 분야에 적용될 수 있어 그 중요성이 갈수록 높아지고 있다. 최근에는 전장감시, 목표물 추적 등 유비쿼터스 센서 네트워크 기술을 적용한 국방 분야의 기술개발을 위해 국방부와 정보통신부가 공동으로 민군겸용 기술 개발 사업을 추진하고 있다. 센서 네트워크 기술 중에서 센서 노드에 탑재되는 초소형 운영체제는 센싱 정보를 처리하고 센서 노드 간의 통신을 지원하는 핵심 요소기술이다. 제한된 메모리와 CPU 자원을 가진 센서 노드의 특성상 센서 노드용 운영체제는 초소형 크기를 가져야만 하므로 기존의 RTOS 보다 더 많은 제약사항을 가진다[1]. 대표적인 센서 노드용 초소형 운영체제로는 TinyOS, SOS, MANTIS 등이 있으며, 국내에서는 한국전자통신연구원에서 개발한 Nano Qplus가 있다.

TinyOS는 UC 버클리에서 진행해온 Smart Dust 프로젝트에 이용하기 위해 개발된 Event-driven 방식의 센서 네트워크용 초소형 운영체제이다[2]. TinyOS는 극도로 제한된 자원을 가지는 센서 노드에 사용하기 위해 멀티쓰레드를 지원하지 않고, NesC(network Embedded System C)라는 동적 메모리를 할당하지 않는 고유한 프로그래밍 언어를 사용하여야 하므로 멀티쓰레드를 이용하여 C 프로그래밍 하던 기존의 개발자에게는 다소 불편할 수 있다.

UCLA에서 개발한 SOS는 Event-driven 방식의 초소형 OS 이며 동적 메모리 할당을 지원하고 C 프로그래밍 언어를 지원한다[3]. 콜로라도 대학에서 개발한 MANTIS[4]와 한국전자통신연구원에서 개발한 Nano Qplus[5, 6]는 멀티쓰레드를 지원하고 C 프로그래밍 언어를 지원한다는 공통점이 있다. 이러한 센서 노드용 초소형 OS 들의 기능적 특성과 차이점은 표 1에 요약하였다.

무선 센서 네트워크 시스템은 적게는 수십 개에서 많을 경우 수백 개의 자율적인 노드들로 구성되므로 저렴한 하드웨어로 노드를 구성해야 하고, 센서노드용 운영체제는 필수적으로 자동구성과 재구성이 가능한 통신 기법을 지원해야 한다. 그리고 이러한 센서 네트워크는 모니터링 프로그램을 통해 전체 시스템의 효율적인 관리가 가능해야 한다.

그림 1은 이러한 센서 네트워크 시스템의 구성 예를 보여준다.

본 논문에서는 한국전자통신연구원에서 개발하고 있으며, 국방 분야에 적용하기 위해 국방과학연구소와 협

표 1 Nano Qplus와 타 운영체제의 기능 비교

	TinyOS	MANTIS OS	SOS	Nano Qplus
Multi-tasking based on priority	불가	가능	가능	가능
Language	nesC	C	C	C
Low-power mode	없음	없음	있음	있음
Dynamic Memory Allocation	없음	있음	있음	있음
Execution model	Event-driven	Thread	Event-driven	Thread
IPC	Command & Event	Message Queue	Message Passing	Message Queue, Mail Box

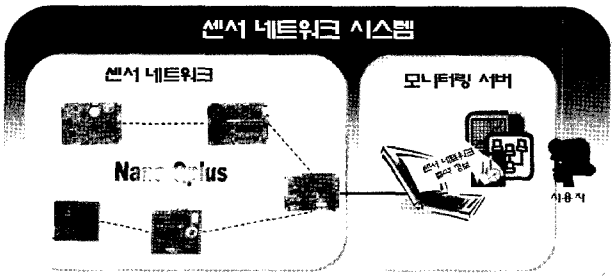


그림 1 센서 네트워크 시스템

의 중인 센서 네트워크용 초소형 운영체제인 Nano Qplus의 하드웨어와 커널 기술에 대하여 소개한다.

2. 센서노드 하드웨어 플랫폼

무선 센서노드의 하드웨어는 제한된 배터리 전원을 효율적으로 사용할 수 있도록 에너지 효율성 및 저전력 관리 기능을 지원해야 하며, 센서 네트워크 응용에 따른 호환성 확보를 위해 주요 기능별 모듈을 구분하고 모듈간 인터페이스를 확장 가능한 구조로 설계하여 다양한 조합이 가능해야 한다.

위와 같은 요구사항을 충족시키기 위해 ETRI에서 개발한 센서노드인 ETRI-Smart Sensor Node (ETRI-SSN)는 저전력 에너지 관리 기능과 확장 인터페이스를 통한 모듈화 기능을 중요시하여 설계되었으며, 그림 2와 같이 메인(Main), 베이스(Base), 센서(Sensor), 액츄에이터(Actuator) 모듈로 구분된다.

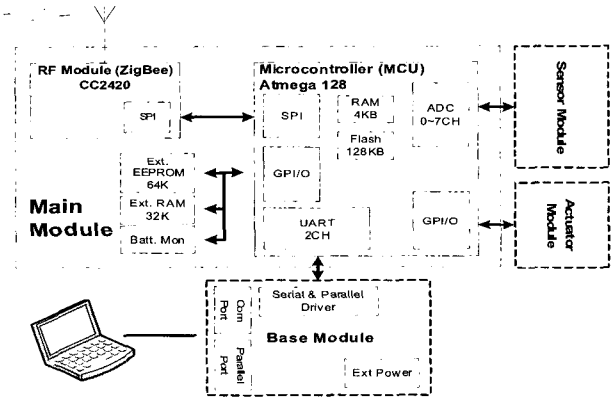


그림 2 센서노드 H/W 구조

메인 모듈은 8bit Microcontroller인 ATmega128과 무선 통신을 위해 2.4GHz 대역의 IEEE 802.15.4 PHY와 H/W MAC 기능을 지원하는 Chipcon사의 CC2420 RF 송수신기를 가지고 있다. ATmega128은 8개의 ADC 채널과 시리얼 통신을 위한 UART 인터페이스, 프로그램 저장을 위한 128KB의 메모리가 내장되어 있다. 배터리를 공급 전원으로 사용하는 경우 3.3V의 입력 전원을 안정화 회로를 통해 전체 시스템을 위한 3.3V와 RF Transceiver를 위한 1.8V의 전원을 안정적으로 제공하며, 저전력 관리 지원을 위해 슬립 모드를 지원한다. CC2420은 2.4GHz 대역의 무선 주파수를 사용하며 최대 250Kbps의 전송률을 지원한다.

센서 모듈에서 지원되는 센서는 온도, 조도, 습도, 적외선, 가스, 초음파 등을 측정하며 각각의 센서들은 구성 방법에 따라 다양하게 사용될 수 있다. 초음파 센서는 위치 인식 또는 거리 측정을 위해 사용되며 별도의 독립된 모듈로 제작된다. 센서 및 Amp 구동을 위한 전원은 메인 모듈의 확장 인터페이스를 통해 연결되며, 메인 모듈과 결합하여 센서 기능을 갖춘 노드로서의 역할을 수행한다.

액츄에이터 모듈은 DC/AC Relay를 통한 On/Off 제어기능을 제공한다. 일반 적인 경우 AA 1.5V 건전지 2개를 사용하여 동작된다.

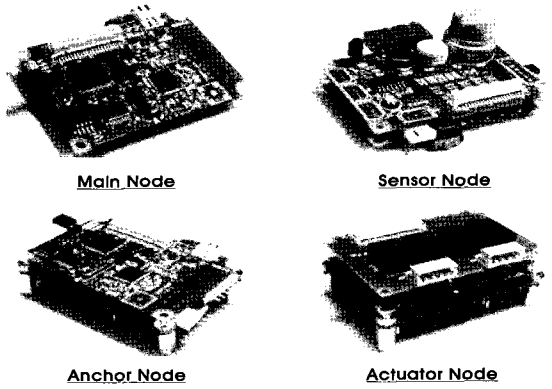


그림 3 센서노드의 구성 예

이러한 메인, 베이스, 센서, 액츄에이터 모듈들은 각각 구성하고자 하는 센서노드의 응용에 따라 표 2와 같이 결합되어 그림 3과 같이 각각의 노드를 구성할 수 있다. 메인 모듈은 모든 노드에 공통으로 포함되며, 메인 모듈과 센싱 모듈이 결합되면 센서 노드, 메인모듈과 엑츄에이터 모듈이 결합되면 엑츄에이터 노드, 메인 모듈과 베이스 모듈이 결합되면 싱크 기능을 담당하는 앵커 노드가 된다.

표 2 센서노드 조합

Node Type	Hardware Module				OS Module				
	Main Module	Base Module	Sensor Module	Actuator Module	Task Mgmt	PWR Mgmt	RF	Sensing (ADC)	UART
Anchor Node	M	M	O	O	M	O	M		M
Sensing Node	M	O	M	O	M	M	M	M	
Actuator Node	M	O	O	M	M	O	M	M	
Moving Tag(with PDA)	M	M	O	O	M	M	M	O	O

M : Mandatory, O : Option

3. 센서노드용 운영 체제(Nano Qplus)

센서노드용 운영체제에서는 하드웨어적인 제약에 의해 다른 범용 운영체제와는 다른 이슈들이 발생하게 되므로 설계부터 이에 대하여 신중히 고려하여 이러한 제약을 극복할 수 있도록 하는 것이 중요하다.

이러한 하드웨어적인 제약으로는 먼저 에너지 제약을 들 수 있다. 센서 네트워크에 사용되는 대부분의 센서노드는 외부에서 무한정의 에너지를 공급 받을 수

없고, 배터리를 이용해 동작한다. 센서노드의 주기적인 배터리 교체를 통한 센서 네트워크의 유지가 필요한 경우도 있지만, 경우에 따라서는 배터리 교체가 불가능할 수도 있다. 따라서 가능한 각 센서노드의 에너지 소모를 최소화 하고, 전체적인 네트워크 측면에서는 노드들의 에너지 소모를 균등하게 이루어질 수 있도록 하는 것이 중요한 이슈이다.

다음으로는 하드웨어 자원과 관련된 제약이 있다. 센서 네트워크를 이용한 응용 시스템들은 기본적으로 수백, 수천 개의 센서노드들이 네트워크를 이루는 것을 가정하고 있어서, 센서노드가 매우 저렴한 가격에 공급될 수 있어야 한다. 따라서 센서노드에 사용되는 대부분의 하드웨어는 자원이 매우 제한되어 있고 성능이 낮은 것이 대부분이다. 현재 ETRI-SSN (ETRI Smart Sensor Node) 노드는 RAM의 크기가 4KB에 불과하다. 따라서 Nano Qplus의 꼭 필요한 기능만을 추가하여 커널 사이즈를 줄이고 프로세서의 메모리 영역을 효율적으로 관리함으로써 많은 메모리를 확보하는 일과 낮은 컴퓨팅 파워를 고려해 코드를 최적화하여 가능한 CPU 부하를 줄이는 일이 중요하다.

ETRI에서는 이러한 센서노드용 운영체제의 제약을 극복하기 위한 최대한의 노력을 기울여 센서노드를 위한 운영체제를 개발하였다. Nano Qplus는 크게 운영체제의 핵심이 되는 커널 부분과 각종 센서들을 위한 드라이버, 하드웨어들을 제어하고 추상화하는 HAL (Hardware Abstraction Layer) 부분, 그리고 센서 네트워크의 핵심이 되는 통신 프로토콜 스택으로 이루어져 있다.

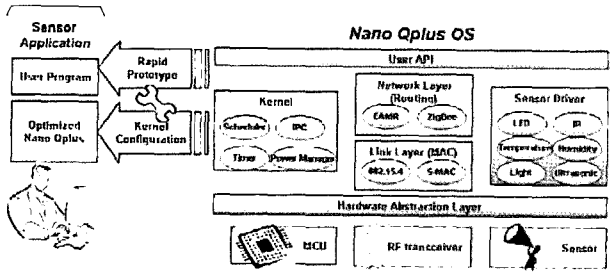


그림 4 Nano Qplus 구조

3.1 Nano Qplus 커널

Nano Qplus 커널의 첫 번째 특징은 멀티 쓰레드를 지원한다는 점이다. 이벤트 드리븐(Event-driven) 형식으로 동작하는 다른 센서 운영체제와는 달리 Nano Qplus는 POSIX 기반의 표준 인터페이스 중에서 멀티 쓰레드와 관련된 POSIX.4a 규격의 서브셋을 지원하고 있다. 또한 실시간 지원의 핵심이라 할 수 있는 태스크 간의 선점(preemption)을 지원하고 응답성을 높여, 실시간 운영체제로서의 면모도 갖추고 있다.

무엇보다도 Nano Qplus 커널의 특징은 센서노드용 운영체제인 만큼 앞서 언급된 제한된 에너지 문제를 해결하기 위하여 저전력 지원을 고려해 작성되었다는 점에 있다. 각 태스크들은 일단 생성되면 태스크가 종료되기 전까지 대기-준비-실행 세가지 상태중의 한 가지 상태에 놓이게 된다. 여기까지는 전통적인 LINUX 스타일의 태스크 스케줄러와 유사하나, Nano Qplus는 준비 상태의 태스크가 없을 경우, 새로운 태스크가 준비 리스트에 들어오기 전까지 sleep mode로 하드웨어를 조절하여 전력 소모를 최소화 할 수 있도록 한다.

또한, Nano Qplus 운영체제는 센서노드의 하드웨어적인 제약, 즉 적은 메모리와 낮은 CPU 성능도 고려하여 제작되었다. 적은 메모리 제약을 극복하기 위해 선택적으로 커널의 모듈들을 선택하여 각각의 응용에 최적화된 커널 이미지를 생성할 수 있도록 하였다. 이러한 과정은 사용자 편의를 위해 menuconfig를 이용해 구성함으로써 가시적인 환경에서 쉽게 선택적으로 각각의 모듈을 더하고 뺄 수 있다.

3.2 Hardware Abstraction Layer(HAL)

Nano Qplus는 센서노드에 사용되는 하드웨어를 제어하고 이를 추상화하여 사용자에게 API(Application Program Interface)의 형태로 제공함으로써, 사용자는 Nano Qplus에서 제공하는 간단한 API를 이용하는 것만으로도 센서노드의 모든 하드웨어를 원하는 대로 제어할 수 있다. 이러한 기능을 제공하는 부분이 HAL부분이다.

기본적으로는 ETRI-SSN 보드에서 지원하는 가스 센서, 초음파 센서, 온도 센서, 습도 센서, 적외선 센서, LED 등의 동작을 제어할 수 있는 드라이버가 작성되어 있다. 이러한 드라이버는 Nano Qplus에서 추상화 되어 사용자에겐 API의 형태로 제공된다. 따라서 사용자는 간단한 API를 통해, 각 센서들을 간단하게 응용 프로그램 내에서 제어할 수 있다. 물론 각각의 센서는 모듈화 되어 있으므로 커널 이미지를 생성할 경우에 포함된 모듈의 센서에 대해서만 조작이 가능하다.

HAL 부분에도 센서노드의 에너지 소비를 최소화 시킬 수 있도록 하기 위하여 저전력 모드와 관련한 사용자 API를 제공한다. ETRI-SSN 보드의 경우 32KHz로 동작하는 비동기적인 클럭을 추가하여 설계함으로써, 최대 8초까지 사용자가 원하는 시간 동안에 sleep mode로 센서노드의 에너지 소모를 최소화 하여 동작시킬 수 있도록 하는 API를 지원한다.

각 하드웨어 모듈들로부터 데이터를 수집할 때는 가능한 한 폴링(polling)방식을 지양하고 인터럽트 방식을 사용하여 CPU가 busy-waiting 상태로 실행되지 않고 대기모드로 들어가게 하였다. 이러한 방식을 통해 각 태스크가 가능한 한 CPU를 적게 사용하게 하고 준비 상태의 태스크 수를 최소화 함으로써, 낮은 성능의 CPU하에서도 Nano Qplus의 강점인 멀티 쓰레드 기능이 활용 가능하고 이벤트에 대한 응답도 빠른 시간에 이루어질 수 있게 하였다. 또한 준비상태의 태스크 수를 최소화하는 것은 CPU가 가능한 많은 시간을 sleep mode로 동작하게 함으로써 에너지 측면에서도 많은 이득을 볼 수 있다. 예를 들어 UART를 통한 시리얼 통신의 경우 사용자로부터 키보드를 통해 데이터를 입력 받을 때 폴링 방식을 통해 CPU가 busy-waiting 하면서 블록된 채로 사용자의 입력을 기다리도록 하는 것이 아니라, 키보드 입력을 인터럽트로 처리하여 인터럽트 서비스 루틴 내에서 사용자 입력 플래그 비트를 이용하여 사용자 입력요청의 유무를 판단하고 입력을 받을 수 있는 상황에는 그에 맞는 적절한 실행이 이루어지도록 처리하였다.

3.3 통신 프로토콜 스택

일반 범용 운영체제의 경우엔 통신 프로토콜 스택을 운영 체제의 일부로 포함시키지 않는다. 운영체제에 관계없이 IEEE, IETF에서 채택한 표준에 의해 정해진 Ethernet, 802.11 등의 MAC 프로토콜, 그리고 TCP/IP, UDP 등의 네트워크 프로토콜을 사용하고 있기 때문이다. 하지만 센서 네트워크의 경우엔 아직까지 정해진 표준이 없고, 독립적인 네트워크 구성을 가지게 된다. 센서 노드가 필요에 의해 기존 인터넷 망에 연결된다 하더라도 일반적으로 이는 센서노드용 운영체제에서 지원하지 않고, 별개의 하드웨어와 소프트웨어를 두게 된다. 따라서 센서노드용 운영체제는 각 운영체제마다의 고유한 MAC 프로토콜, 라우팅 프로토콜을 지니게 되며 이러한 통신 프로토콜 스택 자체가 각 운영체제의 고유한 특성이 된다.

현재 센서 네트워크를 위한 새로운 라우팅 프로토콜이 학계에서 많은 논문을 통해 계속해서 발표되고 있다. 이러한 라우팅 프로토콜은 수백~수천개의 고유 ID를 지니지 않은 노드들로 이루어진 네트워크를 모델로 하여 flooding 방식의 통신을 기반으로 한 것도 있고, 기존 Ad-hoc 네트워크와 비슷하게 적게는 수십에서 많게는 수백개의 각각 고유한 ID를 지닌 노드들이 점대점 방식으로 통신하는 모델을 기반으로 제안된 프로토콜도 있다. 현재 Nano Qplus의 센서 네트워크 모델은 노드수가 수천개까지 이르는 것이 아니기 때문에 후자의 방법으로 통신하는 프로토콜을 탑재하고 있다.

실제로 Nano Qplus 에서 지원하는 MAC 프로토콜로는 Zigbee 프로토콜 스택의 표준이기도 한 IEEE 802.15.4 표준과 S-MAC이 있다. 이는 C 코드로 구현되었으며 사용자가 커널 이미지 생성시에 선택적으로 사용할 수 있도록 하고 있다.

라우팅 프로토콜의 경우엔 Ad-hoc 네트워크에서 초창기에 제안되고 가장 널리 알려진 Perkins가 제안한 AODV(Ad hoc On Demand Distance Vector)와 유사한 라우팅 기법을 사용한다. EAMR(Energy Aware Multi-path Routing)이라고 명명된 Nano Qplus에 탑재된 새로운 라우팅 프로토콜은 기존의 AODV 기법을 기본으로 하여 여기에 disjoint multi-path 기능과 각 노드의 에너지 잔량을 경로 설정시 참고하도록 하는 기능을 추가한 것이다[7]. EAMR 프로토콜 역시 앞서 언급한 센서노드의 에너지 제약을 극복하기 위한 방편으로 고안된 프로토콜이다. 센서 네트워크에서 각 센서노드의 에너지 잔량을 고려하지 않을 경우엔 데이터 통신과 관련된 부하가 분산되지 않고 특정 센서노드에 집중되어 결국 네트워크의 수명이 짧아지는 결과를 가져오게 된다. 하지만 EAMR은 각 센서노드의 에너지 잔량을 고려함으로써 이러한 각 센서노드의 수명의 불균형을 최소화하였고, 이를 통해 획기적으로 센서 네트워크 수명을 늘리게 되었다. 또한 multi-path를 유지함으로써 센서노드의 예상치 못한 고장 혹은 수명이 다한 경우에 대해서도 데이터 전송에 있어서의 신뢰성을 높일 수 있었다.

최근에는 Zigbee Aliance에서 제안한 Zigbee 프로토콜 스택이 센서 네트워크에 많이 사용되고 있으며 ETRI에서도 이러한 경향에 발맞추어 내년 초까지 Zigbee 표준을 따르는 라우팅 프로토콜을 구현하여 완성된 Zigbee 프로토콜 스택을 포함한 새로운 버전을 배포하려는 목표를 세우고 구현 중에 있다.

4. 결 론

무선 센서네트워크 기술이 다양한 분야에 이용되기 시작함에 따라 유비쿼터스 사회가 현실로 다가 오고 있다. 이러한 유비쿼터스 서비스 구축을 실현하기 위해 한국전자통신연구원에서는 광범위한 응용 분야에 센서 운영체제 기술이 사용될 수 있도록 Nano Qplus의 기능 보완과 안정화에 주력하고 있으며, 국방 분야에도 사용될 수 있도록 실시간성 지원과 다양한 통신 프로토콜 지원 등의 기능을 확장하고 있다.

본 논문에서는 한국전자통신연구원에서 개발하고 있으며, 국방 분야에 적용하기 위해 협의 중인 센서 네트

워크용 초소형 운영체제인 Nano Qplus의 하드웨어와 커널을 소개하였다. TinyOS의 경우 nesC 라는 독자적인 언어로 프로그래밍하여야 하므로 개발자가 새로운 언어를 배워야 한다는 부담이 있으나, Nano Qplus는 프로그래머들에게 익숙한 C 언어의 기본적인 라이브러리들과 Nano Qplus에서 제공하는 간단한 API를 이용하는 것만으로 원하는 센서 네트워크용 프로그램을 쉽게 작성할 수 있다는 장점이 있다.

국방부와 정보통신부에서 공동으로 개발하려고 추진중에 있는 민군겸용기술 개발 사업에서는 감시정찰 분야에 활용할 수 있는 센서노드용 OS를 개발할 계획이다. 이 센서 노드용 OS는 Nano Qplus 개발 과정에서 습득한 경험을 바탕으로 국방과학연구소와 한국전자통신연구원이 공동으로 설계를 진행하고, 관련 응용 소프트웨어들이 호환될 수 있도록 초소형 OS API를 Nano Qplus의 API와 동일하게 개발하기로 하였다. 이번 민군겸용 기술개발 사업을 계기로 민간에서 개발하여 사용되고 있는 Nano Qplus기술이 국방 분야에도 적용되어 센서 네트워크 분야의 국내 기술력 확보와 시장 창출의 전기가 마련되기를 기대한다.

참고문헌

[1] I.F.Akyildiz, W. Su et al., "A Survery on Sensor Networks", IEEE Communication Magazine, August 2002.

[2] TinyOS Korea Community Forum, http://www.tinyos.or.kr/

[3] SOS operating system, http://nesl.ee.ucla.edu/projects/sos-1.x/tutorial/

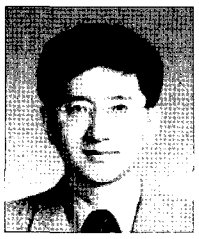
[4] MANTIS operating system, http://mantis.cs.colorado.edu/index.php/tiki-index.php

[5] Nano Qplus operating system, http://qplus.or.kr

[6] 박승민, "센서 네트워크 노드 플랫폼 및 운영체제 동향", 전자통신동향분석 제20권 제5호, 2005.

[7] 신기영 외 3인, "임베디드 무선 센서 네트워크에서의 경량 운영체제와 신뢰성을 고려한 에너지 인지 라우팅 프로토콜", 대한임베디드공학회 국제학술심포지엄 및 춘계학술대회, 2006.

마 평 수



1985 서울대학교 식물병리학 학사
1992 City University of New York, NY, USA 전산학 석사
1995 Wright State University, OH, USA 전산학 박사
1985~1990 시스템공학연구소 연구원
1996~현재 한국전자통신연구원 임베디드SW연구단 책임연구원
1998~현재 ETRI Journal 편집위원

관심분야 : 센서 네트워크, 운영체제, 라우팅 기술, 멀티미디어 스트리밍
E-mail : pmah@etri.re.kr