다중 센서값의 효율적인 전송을 위한 Periodic 데이터 전송 방법 구현

이민구, 강정훈, 임호정, 윤명현
전자부품연구원

The Implementation of periodic method to transmit multi-sensor value in USN

Min goo Lee, Jeong Hoon Kang, Ho Jung Lim, Myung Hyun Yoon
Korea Electronics Technology Institute

Abstract - 본 논문은 다중 센서를 채택한 무선 센서 노드의 효율적인 데이터 메시지 전송 방법에 대한 고찰에서 시작되었다. 이를 위해 TinyOS-1.x 기반의 SurgeTelos 어플리케이션을 기반으로 하여 다중 센서를 전송하는 어플리케이션을 구현하였다. 구현 과정에서, TinyOS에서 제공하는 SurgeTelos는 한 개의 센서 값 전송을 지원하도록 설계 되었는데, 이는 본 연구에서 수행하고자 하는 다중 센서값의 전송 어플리케이션에는 부적합하여, 다중 센서의 효율적인 전송을 위해 Timer 인터럽트에 의한 주기적 데이터 전송 방법을 적용한 신규 어플리케이션을 개발하게 되었다.
따라서 본 논문에서는 다중 센서값의 전송을 위한 Periodic 데이터 전송 방법의 효용성에 대해서 살펴보고, 추후 확장 가능한 센서의 수에 대해 알아보고자 한다.

Key Words -다중 센서, 주기적, 무선 센서 네트워크

1. 서 론

최근 국내/외 무선 센서 네트워크 기술의 발전과 더불어 다양한 센서들을 결합한 무선 센서 네트워크 플랫폼의 수가 꾸준히 증가하고 있다.

특히, 환경의 중요성이 점차 커짐에 따라 온/습도, 조도 등의 단순 환경값 센서뿐만 아니라 공기의 질(CO2, NH3, CO, NO, 포름알데히드, VOC 등)을 센싱할 수 있는 센서들과 무선 센서 플랫폼의 결합이 일부 진행되고 그 결과가 속속 보고 되고 있다.

이 같은 과정에서 환경 센서들이 대부분 고가로 제작/판매되어지다 보니, 무선 센서 노드 1개에 여러 개의 센서들을 부착하여 단가를 낮추는 효과가 있는 다중 센서 노드의 형태로 발전이 이루어지고 있다.

그러나 이때 발생되는 문제점은 무선 전송 어플리케이션의 아키텍처가 기존의 하나의 센싱값을 전송하는 것과는 크게 달라져야 한다는 점이다. 즉, 하나의 센서 노드에서 각각의 센서들의 특성에 따라 전송 프로세스를 특화해서 처리해야 한다.

그리고 추가적으로 고려할 것이 TinyOS 기반에서의 하나의 센서 노드에서 무선 전송을 통해 전송 가능한 센서들의 개수(Number)가 몇 개까지인지 이를 예측하고 실제 테스트를 통해 검증해야 할 필요가 발생한다.

이를 위해 본 논문에서는 다중 센서(조도, 디지털 온도/습도, 아날로그 온도, 마이크로폰, Gyro 등) 노드를 사용하여 동시에 여러 센서 값의 전송이 가능한 어플리케이션을 구현하였으며, 이에 대한 성능 검증을 수행하였다. 그리고 개발된 어플리케이션을 통해 TinyOS에서 하나의 센서 노드에서 무선 전송 가능한 동시 적용 가능한 최대의 센서 개수를 찾기 위한 테스트를 병행하였다.

2. 본 론

2.1 운영 체재

본 논문에서 사용된 OS는 U.C 버클리에서 배포한 TinyOS를 사용하였다. 현재 TinyOS-2.x가 배포되었으며, 본 논문에서는 TinyOS 1.1.14 버전을 사용해 개발 및 테스트를 진행하였다. TinyOS는 센서 네트워크와 같은 임베디드 네트워크 시스템들을 위해 특별히 고안된 아주 심플한 OS이며, 재사용이 가능한 컴포넌트 기반의 구조이다.

즉, 어플리케이션들이 구현에 필요한 각각의 컴포넌트들을 Wiring을 이용하여 연결한다. 이와 같이 컴포넌트 기반으로 이루어진 구조에서는, 다른 OS 서비스들로 구분된 컴포넌트들을 다른 어플리케이션에서 반복적으로 사용하지 않아도 되는 장점을 갖는다.

또한 TinyOS는 동시성을 확보하기 위해 테스크(Task)와 이벤트(Event) 개념을 사용한다. 이 두 가지 요소들의 차이는 "선점"에 대한 가능 여부이다. 즉, 테스크들은 서로를 선점하지 않는 반면, 이벤트들은 테스크들이나 이벤트들의 실행에 대한 선점이 가능한 차이점이 있다.

이처럼 센서 네트워크에서 사용하기 위해 고안되어진 TinyOS는 Sleep 상태로 유지하는 비율이 전체 시간에서 99% 이상으로 동작하여 장시간 노드의 지속 시간을 유지할 수 있도록 설계되어 있다.

2.2 다중 센서 플랫폼

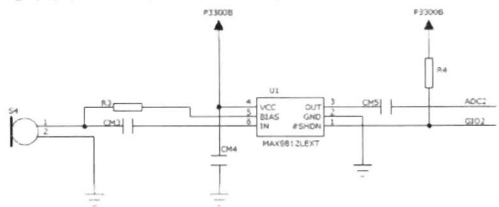
본 논문에서 구현하고자 하는 다중 센서 관련 어플리케이션의 성능 검증을 위해 RF 플랫폼은 Kmote를 사용하였고, 센서 플랫폼으로는 (주)유니온사에서 개발한 다중 센서 플랫폼을 사용하였으며, <그림 1>은 본 논문에서 사용된 Kmote와 (주)유니온사의 다중 센서 플랫폼의 사진이다.

<그림 1> Kmote와 다중 센서 플랫폼

본 논문에서 사용한 RF 플랫폼은 <그림 1>처럼 전자부품연구원에서 개발한 Kmote를 사용하였는데, Kmote는 2.4GHz 무선 RF 대역을 사용하고 있으며, 무선 통신 칩셋으로는 TI사의 CC2420 모듈을 적용하고 있다.

본 논문에서 사용한 다중 센서 플랫폼은 <그림 1>처럼 (주)유니온사의 조도, 디지털 온도/습도, 아날로그 온도, 마이크로폰, Gyro 센서 등이 하나의 센서 노드에 부착되어져 있는 다중 센서 플랫폼을 사용하였다.

<그림 2>는 다중 센서 노드에서 채택된 여러 개의 센서 가운데, 마이크로폰 센서에 대한 회로도이다. 그림처럼 마이크로폰 센서의 출력값을 ADC2 인터페이스를 통해 MCU에서 인식/처리하도록 설계되었다.



Microphone Sensor

<그림 2> 다중 센서 회로도

<표 1>처럼 4개의 센서(조도, 온도, 마이크로폰, Gyro)는 MCU의 ADC0~ADC3 Pin에서 ADC 인터페이스를 이용하여 연결되도록 설계 되었고, 디지털 온/습도 센서는 MCU의 1번 Port 5번 Pin을 통해 센싱값을 인식하도록 설계되었다.

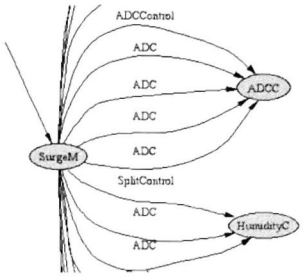
<표 1> 센서들의 MCU 연결 Pin

센서 종류	MCU Pin
조도 센서	ADC0
온/습도(D) 센서	P1_5
온도(A) 센서	ADC1
마이크로폰 센서	ADC2
Gyro 센서	ADC3

2.3 어플리케이션 구현

<그림 3>은 본 논문에서 구현한 다중 센싱값 전송 어플리케이션에 대한 소프트웨어 연결도이다. TinyOS에서 제공하는 SurgeTelos를 기반으로 구현되었으며, 기존의 DemoSensorC 컴포넌트를 사용하고 있는 SurgeTelos는 다양한 센서를 손쉽게 추가 연결할 수 있는 중간 컴포넌트(DemoSensorC)를 가지고 있기 때문에 초보자도 매우 쉽게 사용이 가능하다. 하지만 이 같은 편리성은 여러 센서를 동시에 부착할 수 없는 약점을 동시에 가지고 있다.

이를 개선하기 위해 본 논문에서의 다중 센서 어플리케이션의 구현은 기존의 SurgeTelos 어플리케이션이 사용하고 있는 DemoSensorC 컴포넌트를 제거하고, <그림 3>과 같이 ADCC 컴포넌트와 HumidityC 컴포넌트를 사용하여 각각의 센서들과 ADC 인터페이스를 통해 해당 센싱 컴포넌트와 직접적인 Wiring을 취하였고, <그림 4>와 같이 MCU에서 ADC Pin으로 센싱값을 정상 인식하도록, 센서별로 ADC 인터페이스와 MCU의 가상 ADC 포트를 연결 해주었다.

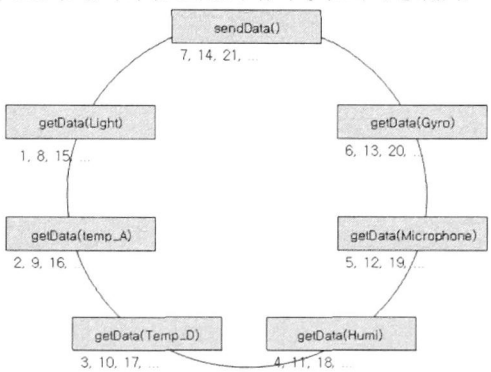


〈그림 3〉 다중 센서 어플리케이션의 핵심 컴포넌트 연결도

```
enum
{
  TOS_ADC_light_PORT = unique("ADCPort"),
  TOSH_ACTUAL_ADC_light_PORT = ASSOCIATE_ADC_CHANNEL(
    INPUT_CHANNEL_A0,
    REFERENCE_VREFplus_AVss,
    REFVOLT_LEVEL_1_5
  )
};
```

〈그림 4〉 ADC 가상 포트와의 연결 구현

<그림 5>는 본 논문에서 구현한 다중 센서 어플리케이션의 주기적인 전송 순서도이다. <그림 5>처럼 TIMER 인터럽트 발생에 의해서 1초에 조도 센서값을 읽고, 2초에 아날로그 온도값, 3초에 디지털 온도값, 4초에 디지털 습도값, 5초에 마이크로폰 센서값, 6초에 Gyro 센서값, 7초에 무선 전송 등의 프로세스를 주기적으로(7초마다) 수행하도록 구현하였다.



〈그림 5〉 다중 센서 어플리케이션의 주기적 전송 순서도

<그림 6>은 다중 센서 어플리케이션의 센서 값들을 무선으로 전송하기 위해 사용된 메시지 구조이다. 총 6개의 센서 값을 동시에 전송할 수 있도록 메시지의 구조를 설계하였으며, 각각의 센서 값들은 2바이트로 데이터를 전송할 수 있다.

이때 주의할 점은 TinyOS에서 제공하고 있는 무선 전송 기본 메시지의 데이터 Payload의 Default 값이 '28'로 설정되어져 있는데, 본 논문에서처럼 6개의 복합 센서의 값들을 동시에 전송하기 위해서는 반드시 메시지 데이터 Payload 값을 28이상으로 확장해야 한다. 본 논문에서는 '32'로 설정하여 테스트를 수행하였다.

```
typedef struct SurgeMsg {
  //Message Length 10+10+12 = 32 bytes
  uint16_t type;
  uint16_t reading;
  uint16_t parentaddr;
  uint32_t seq_no;

  uint16_t Light;
  uint16_t Temp_ANAL;
  uint16_t Temp_DIGI;
  uint16_t Humi_DIGI;
  uint16_t Microphone;
  uint16_t Gyro;
} SurgeMsg;
```

〈그림 6〉 무선 전송에 사용된 다중센서 메시지 구조

2.4 실험 환경 및 결과

본 논문에서 구현된 다중 센싱값 전송 어플리케이션의 성능 검증을 위해 다중 센서(조도, 디지털 온도/습도, 아날로그 온도, 마이크로폰, Gyro 등)의 센싱값 변화 이벤트 발생에 따른 수신측에서의 센싱값 변화를 확인하고, 하나의 센서 노드에 확장 적용 가능한 센서의 개수를 검증하기 위해 테스트하였다.

2.4.1 실험 환경

이를 테스트하기 위한 환경으로는 다중 센서 플랫폼이 결합되어 있는 센서 노드(A)와 무선으로 센싱 데이터를 수신하기 위한 센서 노드(B), 그리고 센서 노드(B)와 USB로 연결되어 센싱 데이터를 분석하기 위한 노트북(VAIO VGN-T27LP/L)으로 구성하였다.

센서 노드(A)에 다중 센서 어플리케이션을 프로그래밍하여 퓨징하였으며, 센서 노드(B)에는 무선으로 메시지를 수신할 수 있는 TOSBase 어플리케이션을 퓨징하여 테스트하였다.

센서 노드(A)와 센서 노드(B)간의 무선 통신 환경으로는, GroupID(0x7d), RF Power(16), Channel(25)을 사용하였다.

센서 노드(B)와 결합되어 사용된 노트북에 설치된 메시지 분석 Tool은 TinyOS에서 배포한 'ListenRaw' Java Tool을 시그윈(Cygwin) 상에서 사용하였다.

다중 센서 값의 무선 전송을 통해 수신된 메시지의 구조는 <그림 7>과 같다.

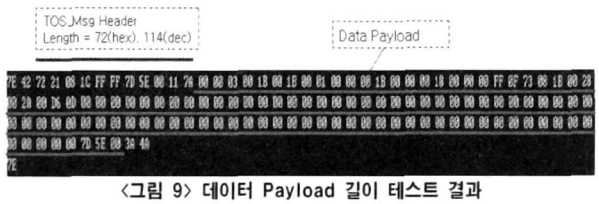
〈그림 7〉 수신된 메시지 구조

2.4.2 실험 결과

<그림 8>은 다중 센싱값 전송 어플리케이션에 대한 메시지 분석을 ListenRaw Tool 상에서 살펴본 결과이다. <그림 7>에서 살펴본 바와 같이 수신된 메시지의 구조에 의거하여 살펴보면, 다중 센서 6개의 값이 센싱 이벤트 발생에 따라 정상적으로 변화하고 있음을 확인할 수 있다.

〈그림 8〉 다중 센싱값 전송 어플리케이션 결과값

<그림 9>는 하나의 센서 노드에 확장 적용 가능한 센서의 개수를 유추하기 위해 다중센서 어플리케이션의 데이터 Payload에 실장 가능한 데이터의 개수를 검증하는 테스트 결과이다. 테스트 결과상 다중센서 어플리케이션의 데이터 Payload에 실장 가능한 데이터 사이즈는 114(Dec)임을 검증하였다. 즉, 이 보다 더 큰 데이터 사이즈는 TinyOS에서 한 번에 전송이 불가능하다. 물론 멀티-홉 헤더, 어플리케이션 헤더 등에 대한 고려시 실제 데이터 Payload의 사이즈는 114(Dec)보다 20이 작은 94(Dec)가 된다.



〈그림 9〉 데이터 Payload 길이 테스트 결과

3. 결 론

본 논문에서는 구현된 다중 센서값 전송 어플리케이션의 성능 검증을 위해 다중 센서(조도, 디지털 온도/습도, 아날로그 온도, 마이크로폰, Gyro 등)의 센싱값 변화 이벤트 발생에 따른 수신측에서의 센싱값 변화를 주기적으로 확인할 수 있었다.

그리고 다중센서 어플리케이션의 데이터 Payload에 실장 가능한 데이터의 개수를 검증할 수 있었다. 하지만, 이 개수가 곧 확장 적용 가능한 센서의 개수는 아니다. 즉, 센서 플랫폼의 MCU가 지원 가능한 물리적인 ADC Port가 8개이고, 기타 4개의 GIO 포트들을 활용할 수 있기 때문에 물리적으로 확장 가능한 센서의 개수는 일단 제약적이다.

본 논문의 결과는 추후 이미지(Image) 혹은 음성(Voice) 데이터처럼 사이즈가 큰 데이터의 전송을 요하는 어플리케이션을 개발하는 경우, 본 논문에서 검증한 실험 결과를 참고하여 개발하면 도움이 될 것으로 기대한다.

[참 고 문 헌]

[1] David Gay, Philip Levis, David Culler, "Software Design Pattern for TinyOS", LCTES'05, 2005
[2] V. Handziski, J.Polastre, C.Sharp, "Flexible Hardware Abstraction for Wireless Sensor Networks", EWSN 2005, 2005
[3] http://docs.tinyos.net/index.php/Main_Page
[4] http://www.tinyos.net/tinyos-1.x/doc/tutorial/index.html