 HANKUK UNIVERSITY OF FOREIGN STUDIES	년월일 2012-06-08	문서번호 2.3	변경코드 2.3	수정회수 7	페이지 1(53)
	작성자/소속 김지훈/정보통신공학과	검토자 김지훈, 최한홍, 김연주		승인자 정일영	문서종류 상세설계서
제목 지능형 전력 제어 시스템 상세설계서		Title Smart Electric Power Control system Description Document			

지능형 전력 제어 시스템 상세설계서

(Smart Electric Power Control System)



Ver.2.3

2012.06.08

한국외국어대학교

정보통신공학과

2조(FBL)



HANKUK UNIVERSITY
OF FOREIGN STUDIES

문서 정보

구 분	소 속	성 명	날 짜	서 명
작성자	한국외국어대학교	김지훈	2012. 04. 30	
	한국외국어대학교	최한홍	2012. 04. 30	
	한국외국어대학교	김연주	2012. 04. 30	
검토자				
사용자				
승인자	한국외국어대학교	정 일 영		



년월일	문서번호	변경코드	수정회수	페이지
2012-06-08	2.3	2.3	7	3(53)

머리말

본 상세설계서를 작성하는데 대한 목적은 Kmote Electric Powermeter와 WSN-Multitab을 이용하여 실시간으로 전력을 측정하고 이를 기반으로 전력 제어를 할 수 있는 시스템 및 상세설계서를 기술한 것에 있다.





년월일 2012-06-08	문서번호 2.3	변경코드 2.3	수정회수 7	페이지 4(53)
-------------------	-------------	-------------	-----------	--------------

개정 이력

버전	작성자	개정일자	개정 내역	승인자
1.0	김지훈	2012.04.25	초안 작성	
	최한홍	2012.04.25	초안 작성	
	김연주	2012.04.25	초안 작성	
	검토자	김지훈		
1.1	김지훈	2012.04.28	초안 수정	
	최한홍	2012.04.28	초안 수정	
	김연주	2012.04.28	초안 수정	
	검토자	최한홍		
1.2	김지훈	2012.04.29	시스템 구성도 수정	
	최한홍	2012.04.29	하드웨어 부분 수정	
	김연주	2012.04.29	기능 설명 부분 수정	
	검토자	김연주		
1.3	김지훈	2012.04.30	시스템 구성도 재 수정	
	최한홍	2012.04.30	하드웨어 부분 재 수정	
	김연주	2012.04.30	기능 설명 부분 재 수정	
	검토자			
2.0	김지훈	2012.05.04	전체 검토 및 수정	
	최한홍	2012.05.04	시스템 구성도 재 수정	
	김연주	2012.05.04	하드웨어&소프트웨어 재 수정	
	검토자			
2.3	김지훈	2012.05.31	전체 검토 및 수정	
	최한홍	2012.05.31	시스템 구성도 재 수정	
	김연주	2012.05.31	APPENDIX & 자체시험방안 재 수정	
	검토자			



목 차

1. 일반	10
1.1 개요	10
1.1.1 목적	10
1.1.2 범위	12
1.1.3 관련자료	12
1.1.4 용어 및 약어	12
2. 시스템 구성도	13
2.1 시스템 기능 구성도	13
2.2 SOFTWARE 구성도	14
2.3 하드웨어 구성도	15
2.4 서버 구성도	16
2.5. 시스템 동작 구성	17
3. 기능 설명	18
3.1 모듈 설명	19
3.1.1 A Module(Sensor Module)	19
3.1.1.1 A-1 (데이터 측정 기능)	19
3.1.1.2 A-2 (데이터 송.수신 기능)	21
3.1.1.3 A-3 (전자기기 제어 기능)	22
3.1.1.3 A-3 (전자기기 제어 기능)	22
3.1.2 B Module(Sensor Module)	24
3.1.2.1 B-1 (데이터 송.수신 기능)	24
3.1.2.2 B-2 (전력량 추출 기능)	25
3.1.2.3 B-3 (16진수 데이터 전력량 수치 변환 기능)	25
3.1.2.4 B-4 (수집된 데이터 저장기능)	25
3.1.3 C Module(User Interface Module)	26
3.1.3.1 C-1 (전력 제어 기능)	26
3.1.3.2 C-2 (위험 수위 확인 자동 전력 차단기능)	27
3.1.3.3 C-3 (수집된 데이터 웹 출력 기능)	28
3.2 INTERFACE 기능 설명	29
3.2.1 IF-1(Module A-> Module B 간 Interface)	29
3.2.2 IF-2(Module A-> Module B 간 Interface)	30
3.2.3 IF-3(Module B-> Module C 간 Interface)	31
3.2.4 IF-4(Module C-> Module B 간 Interface)	32
4. 개발환경	32





년월일 2012-06-08	문서번호 2.3	변경코드 2.3	수정회수 7	페이지 6(53)
-------------------	-------------	-------------	-----------	--------------

4.1 OS.....	32
4.1.1 TinyOS.....	32
4.1.2 Linux	33
4.1.3 Tool & Utility	33
4.1.3.1 VMware	33
4.1.3.2 Eclipse	33
4.1.4 Database	33
4.1.4.1 MySQL.....	33
4.1.5 Languages.....	33
4.1.5.1 Servlet.....	33
4.1.5.2 JSP	33
4.1.5.3 nesC	33
4. 동작설명	34
4.1 SEQUENCE DIAGRAM	34
4.2 동작 설명	35
5. 자체시험 방안	35
5.1 시험 환경	35
5.2 KEP(K-MOTE ELECTRIC POWERMETER)	36
5.3 WSN-MULTITAB	37
5.4 SERVER(DATABASE)	37
5.5 USER INTERFACE	37
5.6 시연 시나리오.....	37
6. 향후 실제 적용방안.....	37
7. 기대 효과	38
8. 세부추진계획 및 일정	39
9. APPENDIX	40
9.1 TINYOS AND NESC	40
9.2 스마트 그리드.....	46
9.3 스마트 그리드 국내외 동향.....	48
10. 안드로이드(ANDROID).....	50
10.1 안드로이드의 개념	50
10.2 ANDROID APPLICATION	51
11. WEB	51
11.1 WEB PAGE.....	52



년월일 2012-06-08	문서번호 2.3	변경코드 2.3	수정회수 7	페이지 7(53)
-------------------	-------------	-------------	-----------	--------------

11.2 WEB SERVER	52
12. 원격제어(REMOTE CONTROL)	52



년월일 2012-06-08	문서번호 2.3	변경코드 2.3	수정회수 7	페이지 8(53)
-------------------	-------------	-------------	-----------	--------------

그림 목차

그림 1 스마트 그리드 개념도	10
그림 2 실시간 전력 사용량 측정	11
그림 3 시스템 기능 구성도	13
그림 4 소프트웨어 구성도	14
그림 5 하드웨어 구성도	15
그림 6 서버 구성도	16
그림 7 전체 시스템 동작 구성도(FLOW)	17
그림 8 전체 시스템 기능 구성도	18
그림 9 IF-1의 기능	29
그림 10 elecMeasurevalue API	30
그림 11 IF-2 의 기능	30
그림 12 Powerctrl API	31
그림 13 Electricvalue table	31
그림 14 IF-4 의 기능	32
그림 21 동작 설명 그림	35
그림 22 콘센트 연결중인 상태	36
그림 23 콘센트 연결 중에서 연결 해제 할 때	36
그림 24 전력 측정 제어 시스템 기대 효과	38
그림 25 컴포넌트의 연결	44



년월일 2012-06-08	문서번호 2.3	변경코드 2.3	수정회수 7	페이지 9(53)
-------------------	-------------	-------------	-----------	--------------

표 목차

<표 1> 전력 측정 제어 시스템 참고 문서 목록 표	12
<표 2> 용어 및 약어 풀이	13

1. 일반

1.1 개요

본 장에서는 Knote Electric Powermeter와 WSN-Multitab을 이용하여 실시간으로 전력을 측정하고 이를 기반으로 전력 제어를 할 수 있는 시스템 및 사용자 요구사항의 총괄 개요를 제공한다. 여기서는 Knote Electric Powermeter와 WSN-Multitab을 이용한 실시간 전력측정제어 시스템 구축을 위해 목적과 범위, 정의사항, 참고자료, 그리고 본 상세설계서의 개요를 소개한다. 전력 측정 제어 시스템은 기본적으로 스마트 그리드 기술을 지향 하고 있다.

스마트 그리드란 기존의 전력망에 정보기술(IT)을 접목하여 전력 공급자와 소비자가 양방향으로 실시간 정보를 교환함으로써 에너지 효율을 최적화하는 차세대 지능형 전력망이다. 스마트 그리드 중에서도 본 상세설계서는 WSN-Multitab에 꽂혀있는 여러 개의 KEP를 통해 들어오는 전력량 패킷을 저장하고 이 저장된 패킷 데이터를 이용하여 집에서 사용되는 전력량이나 문제가 되는 콘센트의 위치를 확인할 수 있다. 문제가 있는 콘센트는 소비자가 전원을 직접 내리거나, Web상에서의, 그리고 안드로이드 애플리케이션을 이용한 스마트폰 에서의 원격제어를 통하여 전원을 제어할 수 있다. 그리고 웹과 스마트폰 상에서 모두 전력량의 임계 값을 정한 후 적정 임계 값을 넘어가면 초과 위험수위를 사용자가 확인할 수 있으며 스마트폰의 경우 알람을 통하여 사용자에게 알릴 수 있다. 본 설계서는 기능을 도식화 하는 상세설계서이다.



그림 1 스마트 그리드 개념도

1.1.1 목적

본 장에서는 Knote Electric Powermeter와 WSN-Multitab을 이용하여 실시간으로 전력을 측정하고 이를 기반으로 전력 제어를 할 수 있는 시스템 및 사용자 요구사항 명세를 기술하는데 있다. 본 문서는 Knote Electric Powermeter와 WSN-Multitab을 이용한

실시간 전력측정제어 시스템에 대한 기본적 요구사항의 식별과 이해를 위하여 작성되었으며, 아래의 사항을 구체적으로 명시하는데 목적이 있다.

- Kmute Electric Powermeter 을 이용하여 실시간으로 전력량 측정이 가능하다.
- 실시간으로 측정한 전력량을 토대로 전기세의 추정치를 환산할 수 있다.
- Web page 와 스마트폰에서 WSN-Multitab 을 이용하여 문제가 되는 콘센트에 대한 원격 전원 제어가 가능하다.
- 전력량의 임계값을 정한 후 적정 임계값을 넘어가면 웹을 통해 초과위험수위를 사용자가 확인할 수 있고, 또한 안드로이드 애플리케이션을 통해 스마트폰의 알람으로 사용자에게 전력량에 대한 경보를 알릴 수 있다.

에너지사용의 실시간 조회



컴퓨터를 켜고
스마트그리드 사용자용 웹에 접속하면

그림 2 실시간 전력 사용량 측정

1.1.2 범위

본 요구사항 명세서에는 Kmote Electric Powermeter와 WSN-Multitab을 이용하고, Web을 통해 전력량을 그래프로 나타내고 불필요한 전력량 소모를 줄이기 위한 원격 제어 시스템 및 구현 기술에 대해 기술하고 있다. WSN-Multitab의 각 포트에서 들어오는 전력 값을 KEP를 통해서 확인한 후 DB에 저장하고, DB에 저장된 데이터를 토대로 서버에 각 포트의 전력량을 그래프로 도식화 하여 사용자로 하여금 전력의 흐름을 직관적으로 알 수 있게 한다. 또한 전력의 최대 임계 값을 설정하여 그 값을 넘으면 웹 상에서 초과 위험 메시지를 사용자에게 보내주고, 스마트폰의 경우 안드로이드 애플리케이션을 통하여 알람을 울리도록 하며, 웹과 스마트폰 상에서 모두 해당 포트의 기기에 대한 전원 제어를 가능하도록 한다. 본 프로젝트 개발 진행에 있어 다음과 같은 범위를 둔다.

- 전력량 측정 : KEP 를 통해 측정된 전력량은 센서를 통해 PC 로 전달된다.
- Web 으로 전송 : 전달된 전력량은 Web 에 있는 Database 에 저장된다.
- 통계치 저장 : KEP 에서 송신하는 전력 데이터를 일정한 주기로 받아 DB 에 저장한다.
- 전력 그래프: DB 에 저장된 전력량을 웹 페이지에 그래프로 표현해서 사용자가 쉽게 상태를 파악할 수 있다.
- 전기세 예측: DB 에 저장된 전력량의 정보를 통해, 전기세를 대략적으로 알아볼 수 있다.
- 초과 위험 알람 : 설정된 전력 임계 값이 넘어가게 되면 알람을 통하여 사용자에게 알려준다. 이 시스템은 웹과 스마트폰에 모두 적용이 된다.
- 원격 제어 : 소비자는 Web 과 안드로이드 애플리케이션을 이용한 스마트폰을 통해 원격으로 WSN-Multitab 을 통해 전자기기의 ON/OFF 명령을 내릴 수 있다.

1.1.3 관련자료

<표 1> 전력 측정 제어 시스템 참고 문서 목록 표

표준문서	문서 제목
아이애펀테크	Kmote_Menuals.pdf
아이애펀테크	WSN-Multitab 관련 서적,문서

1.1.4 용어 및 약어

본 요구사항 정의서에서 사용된 용어 및 그에 대한 설명은 다음과 같다.

- Sensor Network : 센서를 네트워크로 구성한 것을 말한다. 무선 네트워크(WSN : wireless sensor network), 유비쿼터스 센서 네트워크(USN : ubiquitous sensor network) 등으로도 불린다.

본 요구사항 정의서에서 사용된 약어 및 풀이는 다음과 같다.

<표 2> 용어 및 약어 풀이

용어 및 약어	풀이	비고
KEP	Kmote Electric Powermeter	
WSN	Wireless Sensor Network	
USN	Ubiquitous Sensor Network	
DB	DataBase	
JSP	Java Server Page	

2. 시스템 구성도

2.1 시스템 기능 구성도

전력 측정 제어 시스템의(Electronic energy measure system) 아래의 그림은 전체 시스템 구성도를 나타내고 있다.

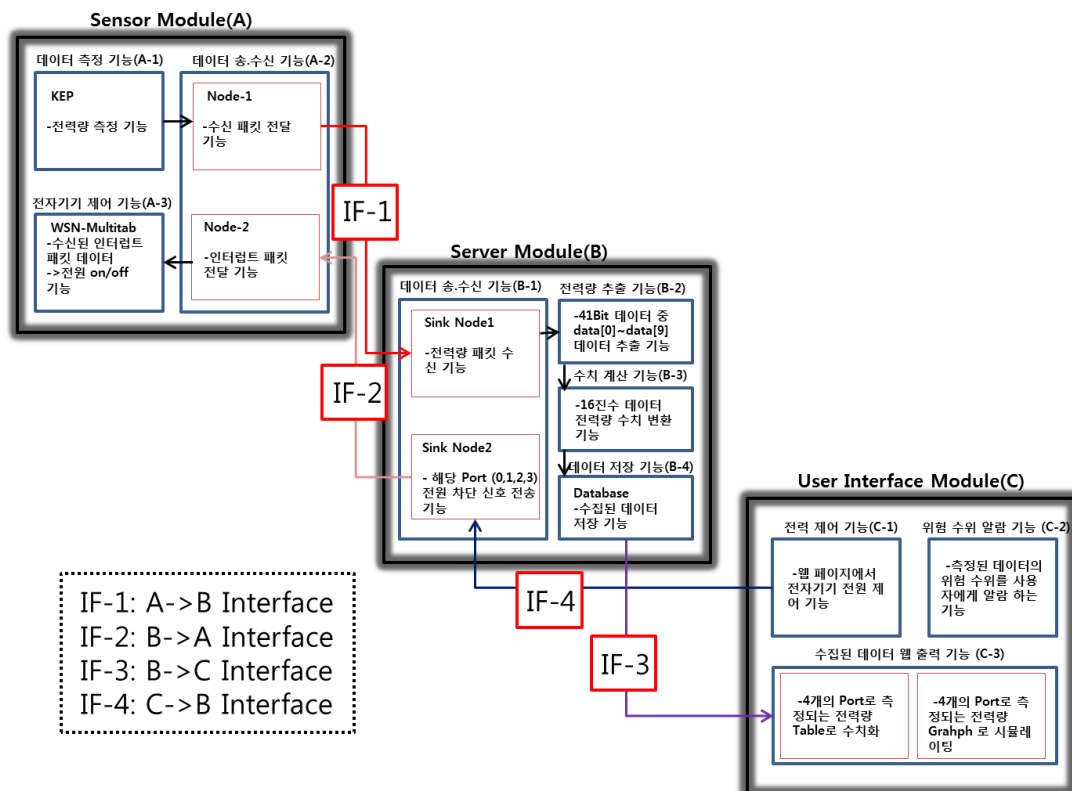


그림 3 시스템 기능 구성도

2.2 Software 구성도

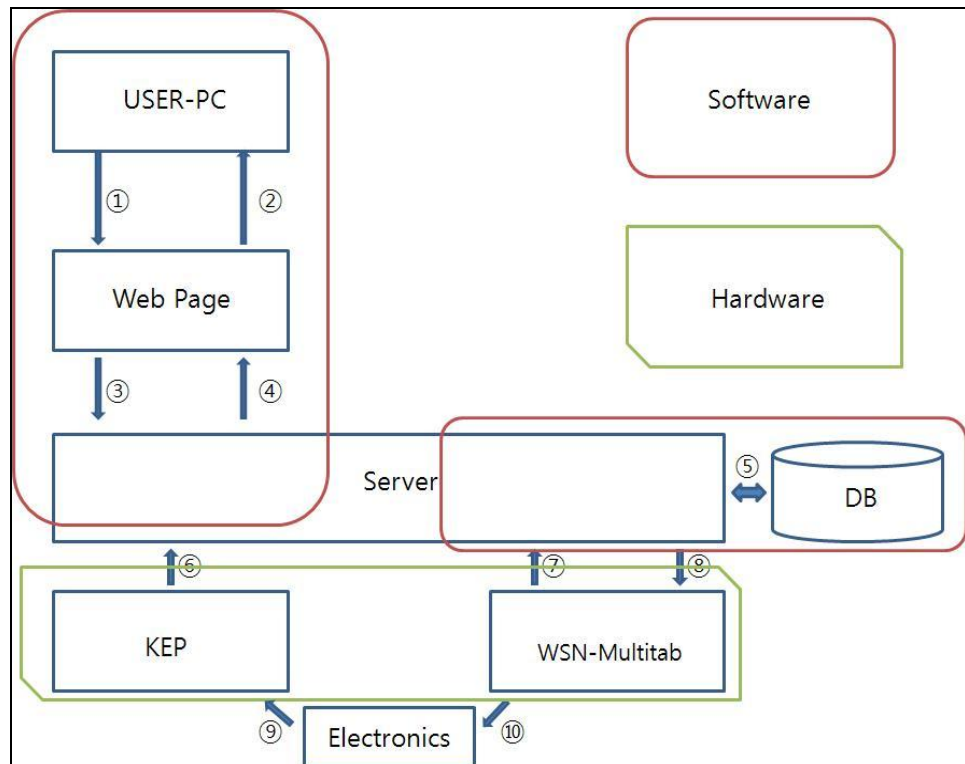


그림 4 소프트웨어 구성도

전력측정 및 제어 시스템에서 소프트웨어의 구성은 다음과 같다.

- USER PC (Web page)
 - Web page 접속을 통해 전력량을 측정하고 예측 전기세를 확인하며 원하는 전자 제품의 전원 컨트롤을 할 수 있다.
- Web Page
 - 서버가 올린 My SQL 데이터베이스 정보를 받아와서 Parsing을 통해 사용자에게 보여준다.
- Data Transmission
 - Sink Node가 보낸 정보를 MySQL로 Parsing을 통하여 데이터 베이스에 정보를 올린다.
 - JDBC를 이용하여 MySQL 데이터베이스를 구현한다..
- Smart Phone(Android Application)
 - 안드로이드 애플리케이션을 통해 스마트폰으로 전력량의 임계 값을 넘어선 기기 에 대한 경보의 의미로 알람을 울릴 수 있게 한다.

2.3 하드웨어 구성도

KEP 및 WSN-Multitab과 웹페이지 기반 전력 측정 및 제어 시스템에서는 전력량 측정, 전기세 환산, 전원컨트롤을 위해 다음과 같은 Hardware를 포함한다.

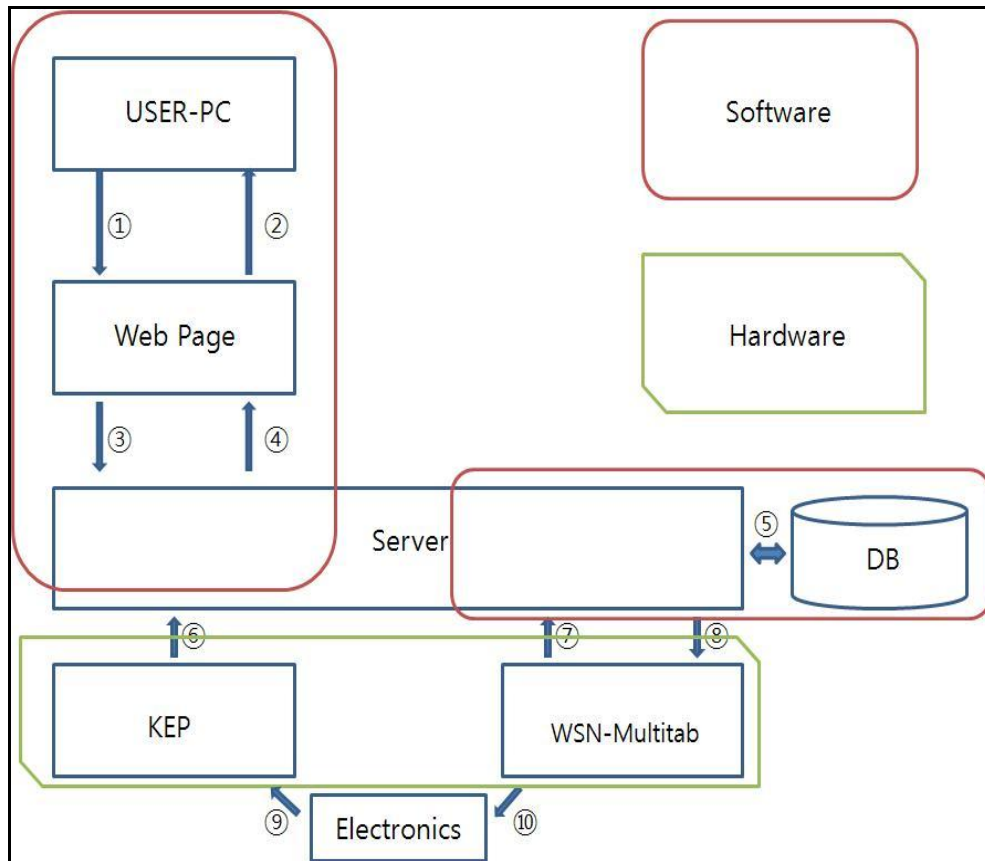


그림 5 하드웨어 구성도

- Knote Electric Powermeter
 - KEP를 통해 원하는 전자제품의 전력을 측정 할 수 있다. 또한 송신노드로 부터 받은 전력을 server에서 DB로 저장한다.
- WSN-Multitab
 - 사용자로부터 내려온 명령을 서버에서 받아 전자제품의 전원을 컨트롤한다.
- Server
 - MySQL을 통해 데이터베이스를 생성하고, 데이터베이스를 JDBC를 통해 제어한다.
- USER-PC
 - USER-PC를 통해 웹페이지에 접속한다. 웹페이지를 통해 전력량 측정 및 전

원 제어를 할 수 있다.

- Smart Phone
 - 스마트폰을 통해 전력량의 임계값이 넘어갈 경우 알람 신호를 주어 사용자가 위험을 알 수 있게 한다.

2.4 서버 구성도

다음 그림 5는 서버의 전체적인 구조를 나타내는 서버 구성도이다. 웹 페이지에 대한 접근은 JSP로 페이지 동적으로 뿌려주기 때문에 JDBC 드라이버를 사용하여 DB를 저장하는 역할을 수행하고 WAS는 Web Application Server로 WS 와 AS의 기능을 수행한다. WS는 Web Server로 아파치 웹 서버로 구현을 한다. 그리고 AS는 Application Server로 톰캣 서버로 구현을 수행한다.

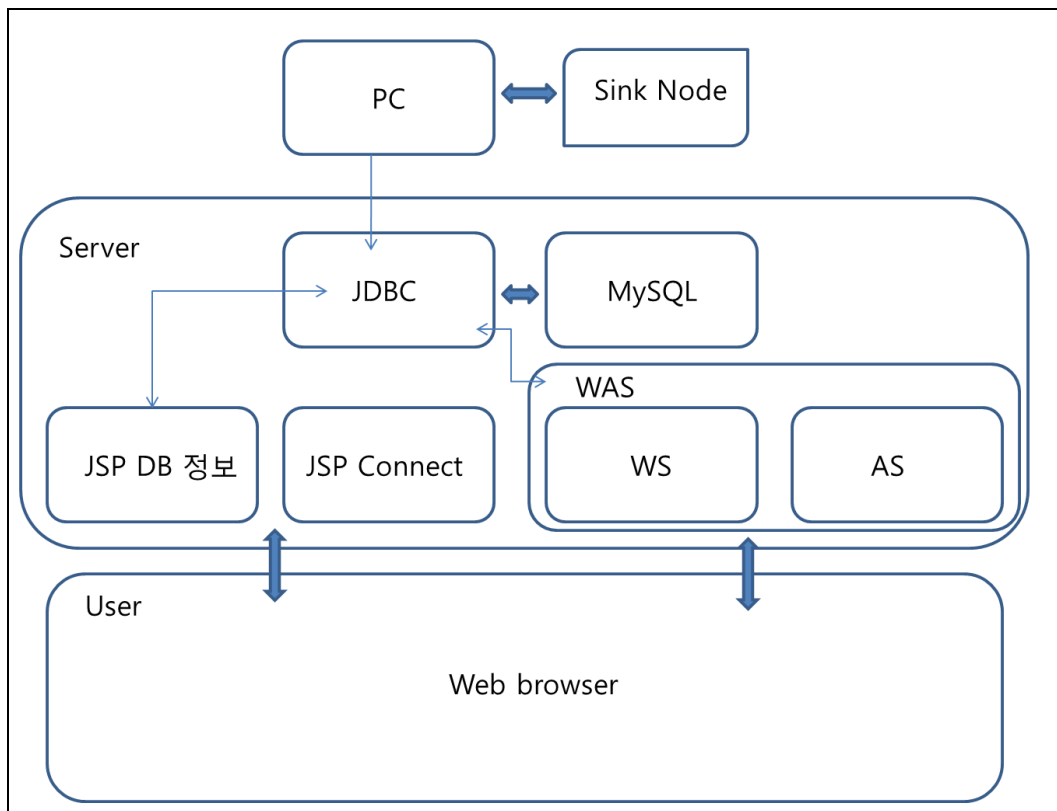


그림 6 서버 구성도

2.5. 시스템 동작 구성

동작 설명은 2.1절에서 제시한 시스템 전체 구성도에서 옆에 해당하는 IF_1~IF_4번호에 해당하는 큰 기능을 중심으로 설명을 도식화한다.

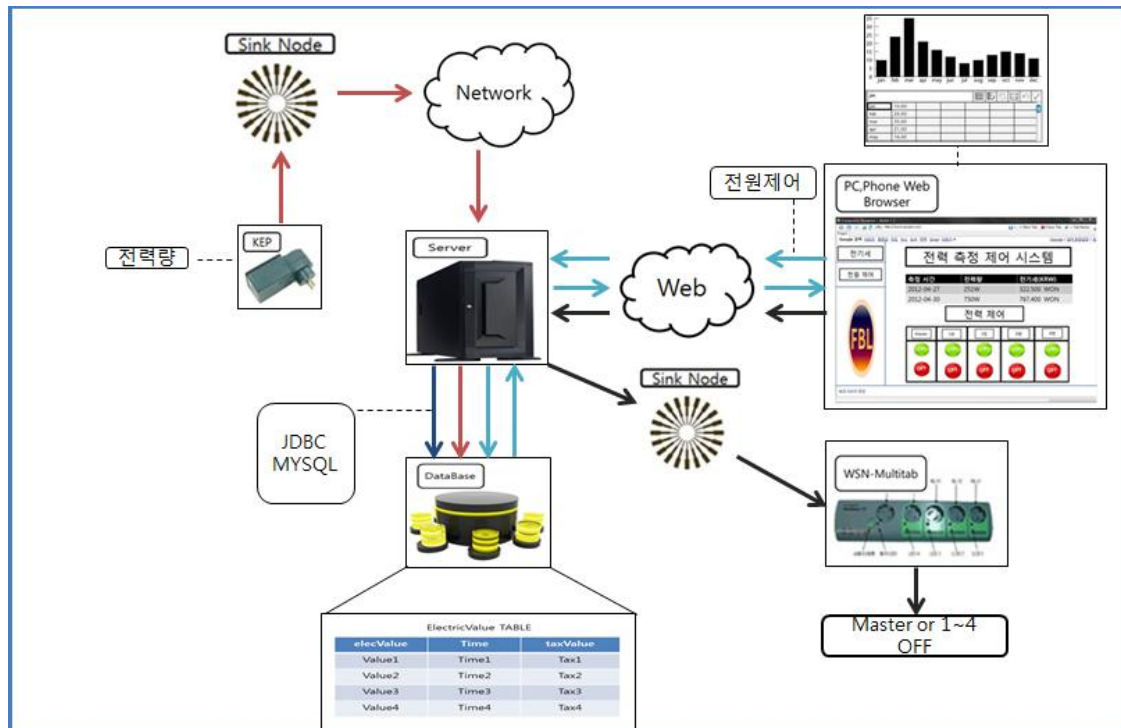


그림 7 전체 시스템 동작 구성도(FLOW)

3. 기능 설명

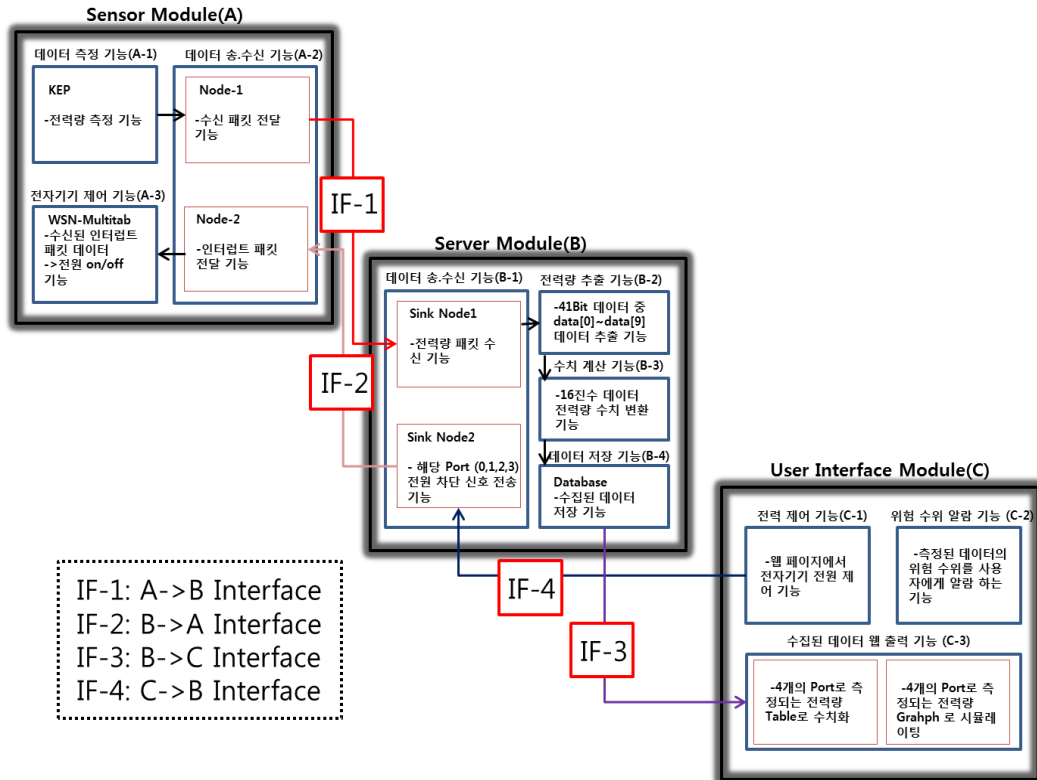


그림 8 전체 시스템 기능 구성도

3.1 모듈 설명

3.1.1 A Module(Sensor Module)

3.1.1.1 A-1 (데이터 측정 기능)

A모듈 중 A-1의 기능은 데이터 측정 기능이다. WSN-Multitab(4개의 Port 0~3)에 장착되어 있는 4개의 KEP(K-mote Electric Powermeter)를 통해 전력량을 측정하는 기능을 수행한다. 데이터의 측정은 다음과 같이 이루어진다. 그림 10은 총 3개의 KEP에서 전송 받은 전력량 데이터를 콘솔창에서 확인 하는 모습이다. 총 3개의 KEP에서 전송 받은 전력량 데이터는 하나의 싱크 노드로 얻어오기 때문에 구별해서 전송 받아야 한다. 이를 위해 그림 11처럼 nesC 프로그래밍을 통해 KEP에 각 KEP를 구별하는 코드 양식을 작성하였다. 이는 서버에서 각 들어오는 패킷을 검사하여 구별해내는 작업을 수행한다.

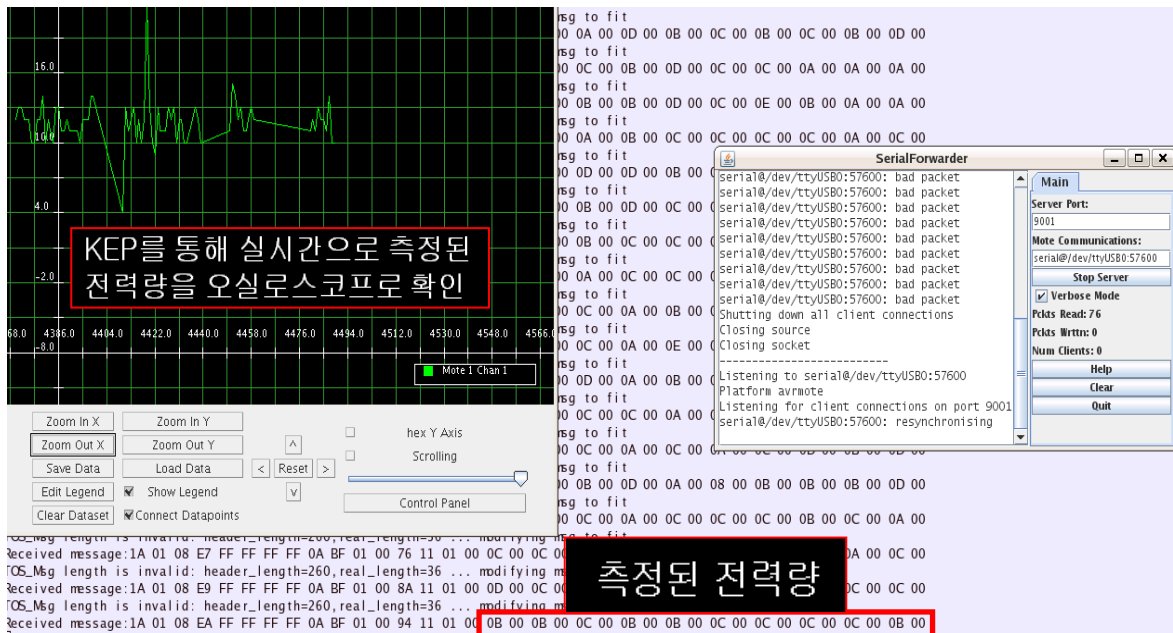
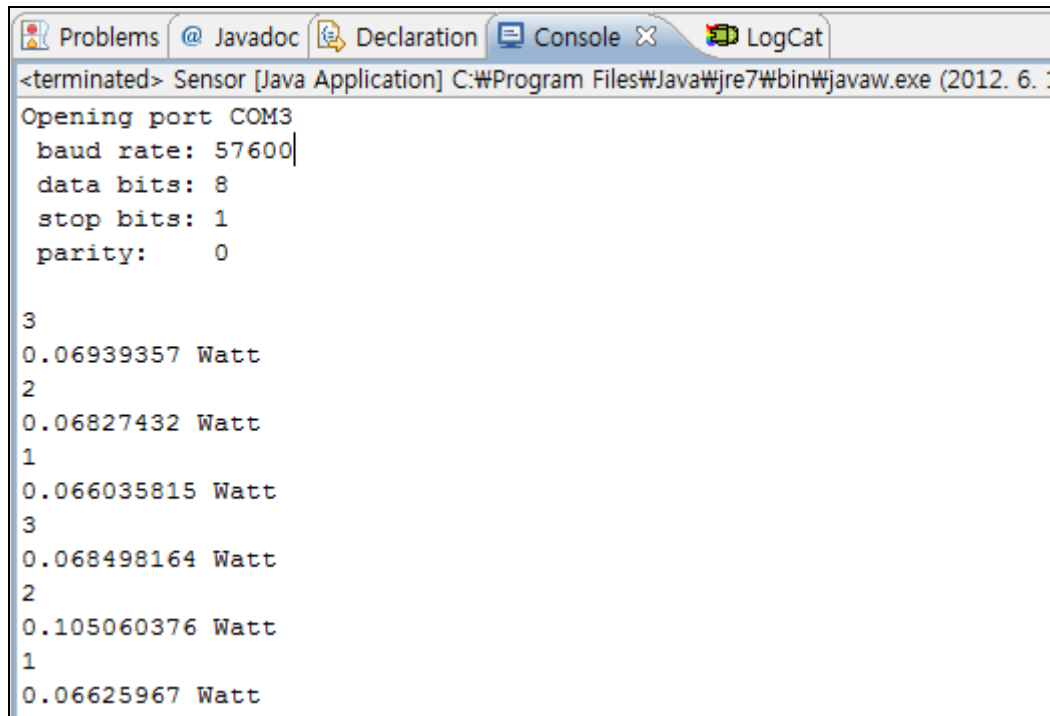


그림 9 데이터 측정 기능

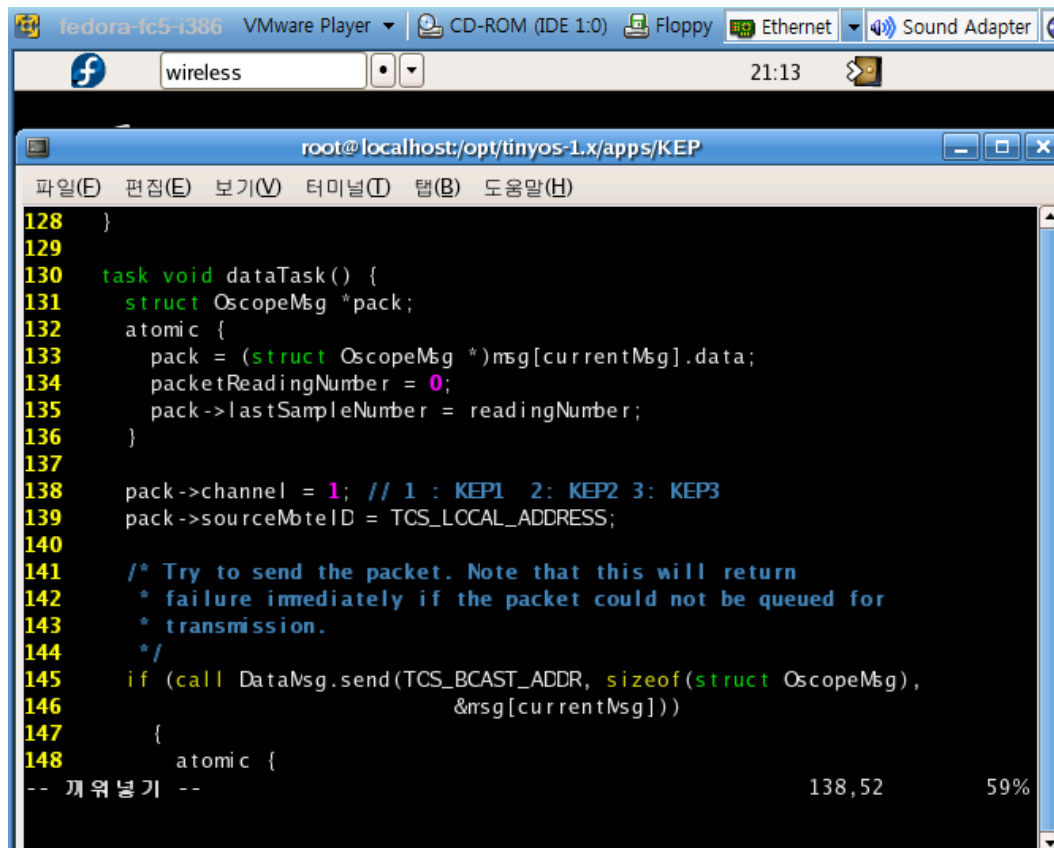


The screenshot shows a Java IDE console window with the title bar containing 'Problems', '@ Javadoc', 'Declaration', 'Console', and 'LogCat'. The console output is as follows:

```
<terminated> Sensor [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (2012. 6. 1)
Opening port COM3
  baud rate: 57600
  data bits: 8
  stop bits: 1
  parity: 0

3
0.06939357 Watt
2
0.06827432 Watt
1
0.066035815 Watt
3
0.068498164 Watt
2
0.105060376 Watt
1
0.06625967 Watt
```

그림 10 데이터 측정 기능2



The screenshot shows a terminal window titled 'root@localhost:/opt/tinyos-1.x/apps/KEP'. The terminal output is as follows:

```
128 }
129
130 task void dataTask() {
131     struct OscopeMsg *pack;
132     atomic {
133         pack = (struct OscopeMsg *)msg[currentMsg].data;
134         packetReadingNumber = 0;
135         pack->lastSampleNumber = readingNumber;
136     }
137
138     pack->channel = 1; // 1: KEP1 2: KEP2 3: KEP3
139     pack->sourceMteID = TCS_LOCAL_ADDRESS;
140
141     /* Try to send the packet. Note that this will return
142      * failure immediately if the packet could not be queued for
143      * transmission.
144      */
145     if (call DataMsg.send(TCS_BCAST_ADDR, sizeof(struct OscopeMsg),
146                          &msg[currentMsg]))
147     {
148         atomic {
149             -- 패워 넣기 --
150         }
151     }
152     138,52 59%
```

그림 11 데이터 측정 기능3

3.1.1.2 A-2 (데이터 송.수신 기능)

A모듈 중 A-2의 기능은 A-1으로부터 측정된 데이터를 B모듈인 Server 모듈로 보내기 위한 Node-1 의 기능이다. 수신된 패킷 데이터는 다음과 같은 메시지 포맷을 가진다.



그림 12 KEP message Format

그림 12에서 보듯이 KEP로 측정한 KEP message Format은 다음 그림과 같이 총 41바이트로 구성되어 있으며 그 중 그림에 표시된 부분이 전력량 수치에 해당되는 부분이다. 총 41바이트로 구성되어 있으며 그 중에 데이터로 표시된 부분이 전력량에 해당하는 데이터 부분이다. 해당 16진수의 데이터 패킷은 B모듈에서 전력량으로 환산하는 기능을 통해 전력량으로 데이터가 가공된다. 나머지 25바이트의 데이터는 다음과 같은 정보를 가지고 있다.

7e	시작
42	Serial Data
1A	length
01	fcfhi
08	fcflo
EC	dsn
FF FF	destpan
FF FF	addr
0A	type
7C	Group ID
03 00	source Mote ID
58 1B	lastSampleNumber
01 00	channel
F7 0F	Data [0]
~	~
F7 0F	Data [9]
35 4B	CRC
7E	끝

그림 13 패킷 데이터 정보

그리고 A-2의 기능 중 Node-2에 수신되는 데이터는 B 모듈로부터 전송 받은 WSN-Multitab 전원 on/off 제어 신호를 A-3(전자기기 제어)모듈로 전달하는 기능을 수행한다.

```
public void open() throws NoSuchPortException, PortInUseException,
    IOException, UnsupportedOperationException {
    System.out.println("Opening port " + portName);
    portId = CommPortIdentifier.getPortIdentifier(portName);
    port = (SerialPort) portId.open("CLASS_NAME", 0);
    in = port.getInputStream();
    port.setFlowControlMode(SerialPort.FLOWCONTROL_NONE);
    port.disableReceiveFraming();
    port.setSerialPortParams(portSpeed, SerialPort.DATABITS_8,
        SerialPort.STOPBITS_1, SerialPort.PARITY_NONE); // port setting
    printPortStatus();
    System.out.println();
}
```

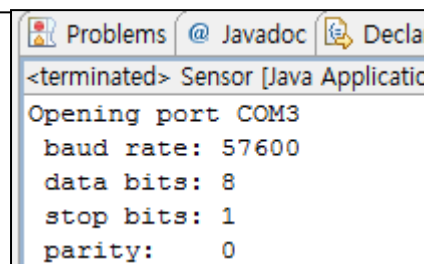


그림 14 데이터 수신 기능

3.1.1.3 A-3 (전자기기 제어 기능)

A모듈 중 A-3의 기능은 서버로부터 전달 받은 전원 제어 패킷으로 전원을 제어하는 기능을 수행한다. 그림 15는 전자기기를 제어하기 위한 패킷을 생성하는 java코드이다. 그림 16은 A-2 모듈의 Node-2로부터 전달받은 데이터를 WSN-Multitab 0~3 전원을 제어하기 위한 nesC코드이다. Switch문에 해당하는 부분은 KEP로 측정된 데이터를 사용자가 확인하여 전원을 내리는 event 처리하기 위한 Switch문이다. 최악의 위험 수위에 대한 정보를 수신 받았을 때 모든 전원을 내리는 부분은 Switch문을 제외한 나머지 부분이다.

```
byte[] On1 = { (byte) 0x7E, (byte) 0x42, (byte) 0xFF, (byte) 0xFF,
    (byte) 0x0A, (byte) 0x7D, (byte) 0x5D, (byte) 0x1C, (byte) 0x31,
    (byte) 0x32, (byte) 0x33, (byte) 0x34, (byte) 0x35, (byte) 0x36,
    (byte) 0x37, (byte) 0x38, (byte) 0x39, (byte) 0x31, (byte) 0x32,
    (byte) 0x33, (byte) 0x34, (byte) 0x35, (byte) 0x36, (byte) 0x37,
    (byte) 0x38, (byte) 0x39, (byte) 0x31, (byte) 0x32, (byte) 0x33,
    (byte) 0x34, (byte) 0x35, (byte) 0x36, (byte) 0x37, (byte) 0x38,
    (byte) 0x39, (byte) 0x00, (byte) 0x2D, (byte) 0xD8, (byte) 0x7E };
```

그림 15 데이터 수신 기능

```
// event occurs when the radio message is received
event TOS_MsgPtr RadioReceiveMsg.receive(TOS_MsgPtr m) {
    TOS_MsgPtr tmp;
    struct stLedNumData* pack;
    tmp = pmsg;
    pmsg = m;
    pack = (struct stLedNumData*)m->data;

    switch(pack->data) {
        case 1:
            call RLY01.setLow();    //ON
            call RLY02.setHigh();   //OFF
            call RLY03.setHigh();   //OFF
            call RLY04.setHigh();   //OFF
            break;
        case 2:
            call RLY01.setLow();
            call RLY02.setLow();
            call RLY03.setHigh();
            call RLY04.setHigh();
            break;
        case 3:
            call RLY01.setLow();
            call RLY02.setLow();
            call RLY03.setLow();
            call RLY04.setHigh();
            break;
        case 4:
            call RLY01.setLow();
            call RLY02.setLow();
            call RLY03.setLow();
            call RLY04.setLow();
            break;
    }
    return tmp;
}

async event void port27int.fired() {
    gINTCtlCnt++;
    if(gINTCtlCnt > 100)
        gINTCtlCnt = 0;

    if(gINTCtlCnt%2) {
        call RLY01.setLow();
        call RLY02.setLow();
        call RLY03.setLow();
        call RLY04.setLow();
    }
    else {
        call RLY01.setHigh();
        call RLY02.setHigh();
        call RLY03.setHigh();
        call RLY04.setHigh();
    }
    call port27int.clear();
}
}
```

그림 16 A-2중 Node-2로 수신된 인터럽트 처리 function

3.1.2 B Module(Sensor Module)

3.1.2.1 B-1 (데이터 송.수신 기능)

A모듈의 A-2기능으로부터 전달받은 데이터를 수신 하거나 C모듈의 C-1기능으로부터 전달받은 데이터를 송.수신하는 기능을 수행한다 Sink Node-1은 수신된 41바이트의 전력량을 전력량 추출 기능인 B-2로 송신하는 기능을 수행하고 Sink Node-2는 전력제어 명령을 A-2로 전달하는 기능을 수행한다.

```
public void read() throws IOException {  
    int[] packet = new int[MAX_MSG_SIZE];  
    int check = 0, count = 0, i;  
  
    while ((i = in.read()) != -1) {  
        if (i == 0x7e && i == check) {  
            send_packet = make_Packet(packet); // make packet  
            count = 0;  
        }  
        packet[count] = i;  
        count++;  
        check = i;  
    }  
}
```

그림 17 데이터 수신 기능(전달 받은 패킷 분석하여 전력량 패킷 확인)

```
SocketServer s2;  
int port = 20222;  
  
if (args.length > 0) {  
    port = Integer.parseInt(args[0]);  
}  
  
while (true) {  
    try {  
        s2 = new SocketServer(port);  
        num = s2.startSocket();  
        s.turnONOFF(num);  
    } catch (IOException e) {  
        System.out.println("소켓 생성에 실패했습니다.");  
    }  
}
```

그림 18 WEB에서 전송 받은 전원 원격제어 명령 전달 기능

3.1.2.2 B-2 (전력량 추출 기능)

B-1의 Sink Node-1로부터 41바이트 패킷 데이터를 전송 받으면 그림 10에서 처럼 F7로 시작하는 부분부터 총 20바이트인 data[0]~data[9]부분을 파싱 하는 기능을 수행한다. 그림 12에 파싱 하는 부분에 대한 그림이다.

3.1.2.3 B-3 (16진수 데이터 전력량 수치 변환 기능)

B-2를 통해 측정된 전력량은 16진수의 데이터 전력량이다. 이 16진수 데이터를 전력량 데이터로 변환하는 기능이 필요하다. 그림 19는 다음 코드는 추출한 16진수 데이터를 전력량으로 변환하는 코드이다.

```
private String make_Packet(int[] packet) {  
    String sensor_info;  
    String protocol = "116/";  
    int roomnum=1;  
    voltage = 0;  
    voltagebuf = 0;  
  
    for (int i = 18; i <= 36; i++) {  
        voltagebuf = voltagebuf + packet[i];  
    }  
    voltage = voltagebuf / 10; // data average  
    voltage = 10 * voltage / 4095; // 전류(A) = 10 * data / 4095  
    voltage = voltage * 220; // P=v*i => electronic power  
    node_number = packet[12];  
  
    sensor_info = protocol + roomnum + Integer.toString(node_number) + "/" + voltage;  
    System.out.println(sensor_info);  
  
    return sensor_info;  
}
```

그림 19 전력량 파싱과 전력량 변환 코드

3.1.2.4 B-4 (수집된 데이터 저장기능)

수신된 전력량은 총 4개의 KEP를 통해 전송 받은 전력량이기 때문에 병렬적으로 처리해야 한다. 이 전력량은 서버의 Database에 저장되어야 하기 때문에 다음과 같은 방법으로 데이터베이스에 저장한다.

```
if (packet[16] == 1)
    DBInit(voltage);
else if (packet[16] == 2)
    DBInit2(voltage);
else if (packet[16] == 3)
    DBInit3(voltage);
```

```
try {
    String jdbcUrl = "jdbc:mysql://127.0.0.1/colderos";
    String userId = "root";
    String userPass = "apmsetup";

    conn = (Connection) DriverManager.getConnection(jdbcUrl, userId,
        userPass); // get Connection Object
    stmt = (Statement) conn.createStatement(); // get Statement Object

    // System.out.println("Database Connection Success!");
    nowTime = dtm.getTime();
    stmt.executeUpdate("INSERT INTO voltage3 (`time`, `voltage`) VALUES('"
        + nowTime + "', " + voltage + ")");
    sum(voltage);
    rs = stmt.executeQuery("SELECT * FROM voltage3");

    rs.close();
    stmt.close();
    conn.close();
} catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

그림 20 수집된 데이터 저장하는 코드

3.1.3 C Module(User Interface Module)

3.1.3.1 C-1 (전력 제어 기능)

C-1 기능은 사용자가 웹 페이지에서 위험수위에 대한 신호를 받거나 최악의 상황의 경우에 대해서 모든 전원을 On/off 하거나 원하는(0~3번 Port)에 대해 전원을 컨트롤 하는 인터럽트 신호를 B Module에 전송하는 기능을 수행한다. 사용자는 WEB으로부터 어디서든지 전원을 컨트롤 할 수가 있다. 그림 21에서 처럼 총 4개의 전원을 자유롭게 ON/OFF 할 수 있다.

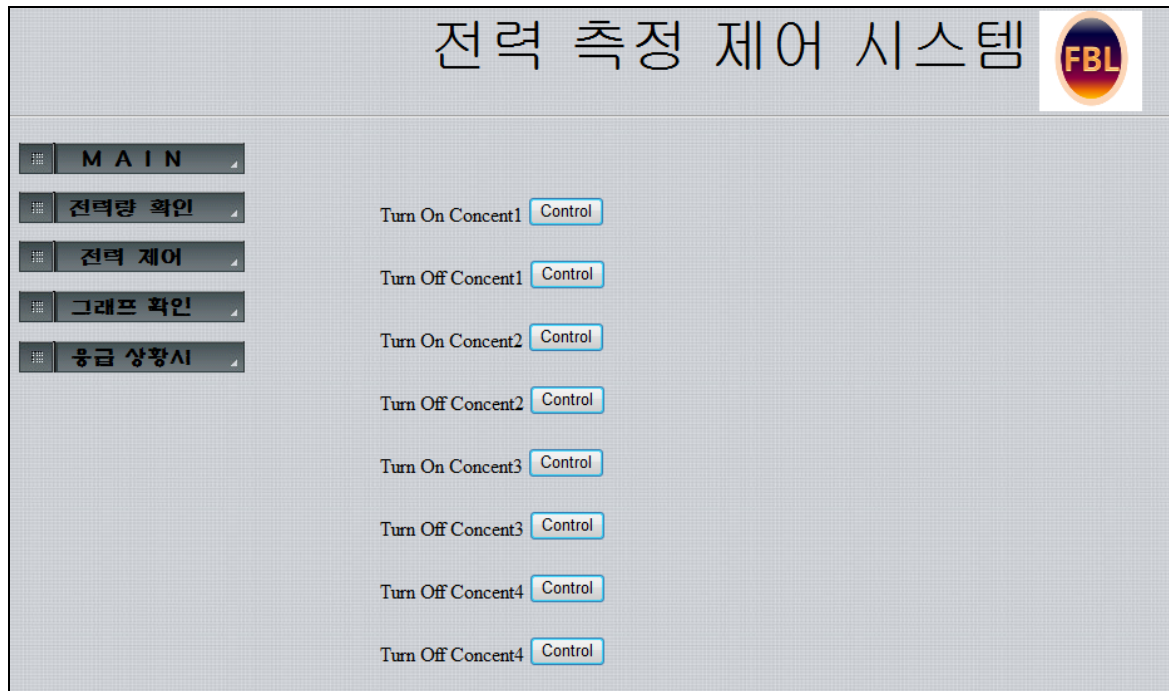


그림 21 전력 제어 기능

3.1.3.2 C-2 (위험 수위 확인 자동 전력 차단기능)

C-2 기능은 전력량의 피크 되어 혼전이 되거나 기타 최악의 상황에 사고에 대비하여 전원이 공급되는 WSN-Multitab에 자동으로 전력을 차단하는 기능을 제공한다.

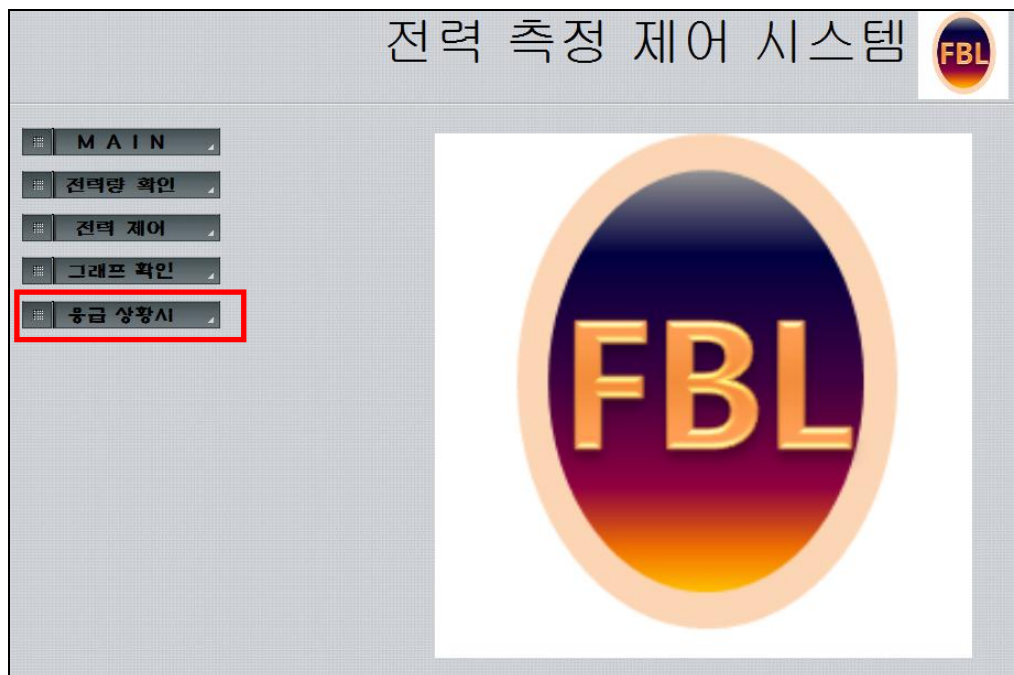


그림 22 위험 수위 확인 자동 전력 차단기능

3.1.3.3 C-3 (수집된 데이터 웹 출력 기능)

C-3 기능은 서버의 데이터베이스에 저장되어 있는 데이터를 Web을 통해 사용자에게 실시간으로 4개의 KEP로 측정되는 전력량을 시물레이팅하는 그래프 출력 기능과 도표로 도식화하는 표 출력 기능을 가지고 있다. 데이터 베이스에서는 측정되는 순간 전력으로 그래프를 통해 나타나게 되며 사용자는 이를 통해 위험을 감지하고 전원을 On/off를 유도하게 된다. 그리고 도표를 통해 사용자로 하여금 자신의 집에서 무의미하게 지출되는 대기전력과 실수로 켜두고 온 전자기기등을 off 시킬 수 있으며 순간적으로 피크되는 전압을 확인하여 위험요소를 사전에 방지할 수 있는 기능을 제공한다.



그림 23 수집된 데이터 확인 기능(표)

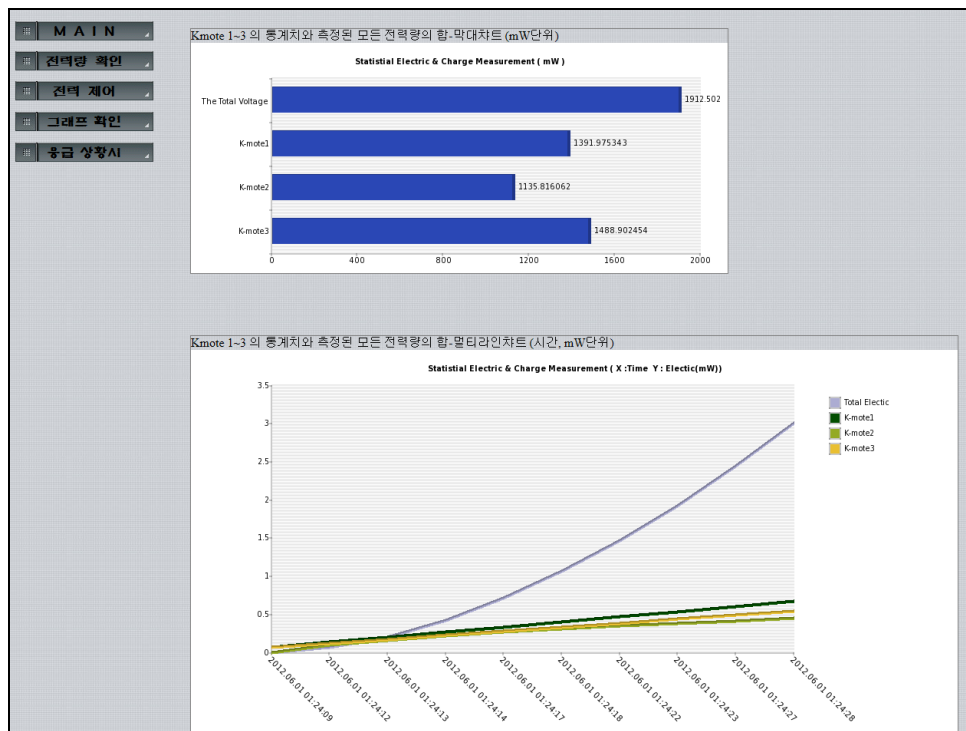


그림 24 수집된 데이터 확인 기능(그래프)

3.2 Interface 기능 설명

3.2.1 IF-1(Module A-> Module B 간 Interface)

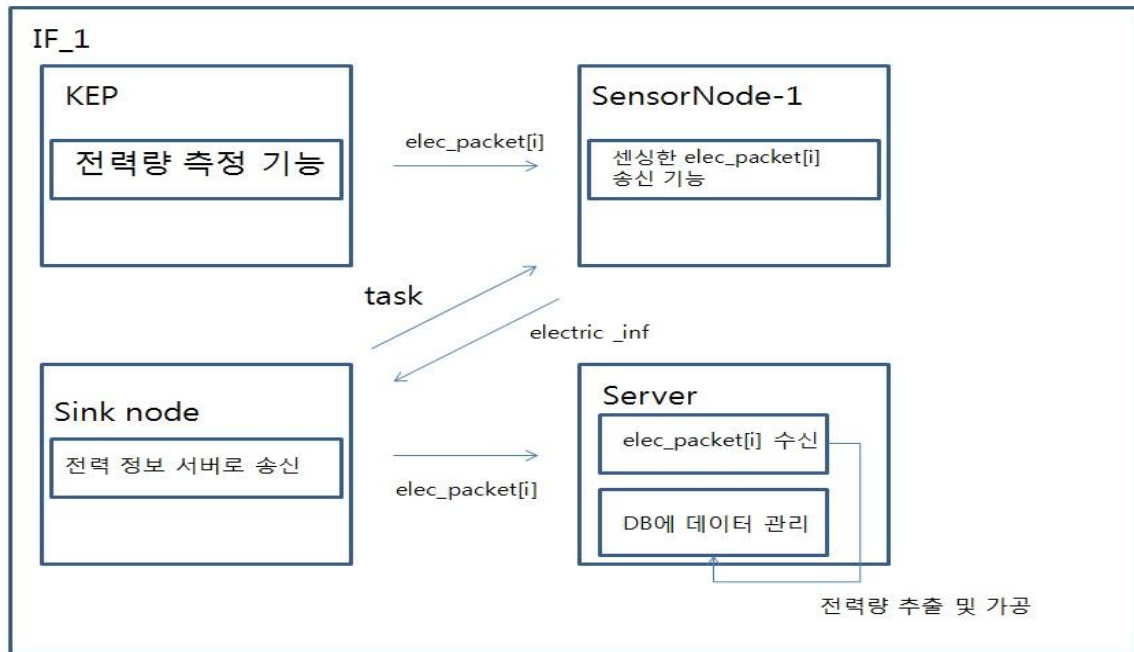


그림 25 IF-1의 기능

먼저 IF_1의 전체적인 시퀀스 및 동작 기능 설명이다. 먼저 KEP를 통해 해당 포트별 전력 값을 측정하고 이 정보를 센서노드가 KEP로부터 받아들인다. 싱크노드로부터 명령 받은 태스크를 수행하기 위해 센서노드는 싱크노드로 수신한 전력정보를 다시 송신한다. 싱크노드는 송신받은 전력 정보를 다시 서버로 전송하게 되고, 서버에서는 전송받은 41바이트 패킷 데이터 중 전력량에 해당하는 부분인 `elec_packet[18]~elec_packet[36]` 까지의 배열의 16진수 데이터를 추출하고, 이를 10진수로 변환한 후에 최종적으로 측정 시간과 함께 DB 테이블에 저장하게 된다. 다음은 IF_1에서 설명한 내용의 API 이다.

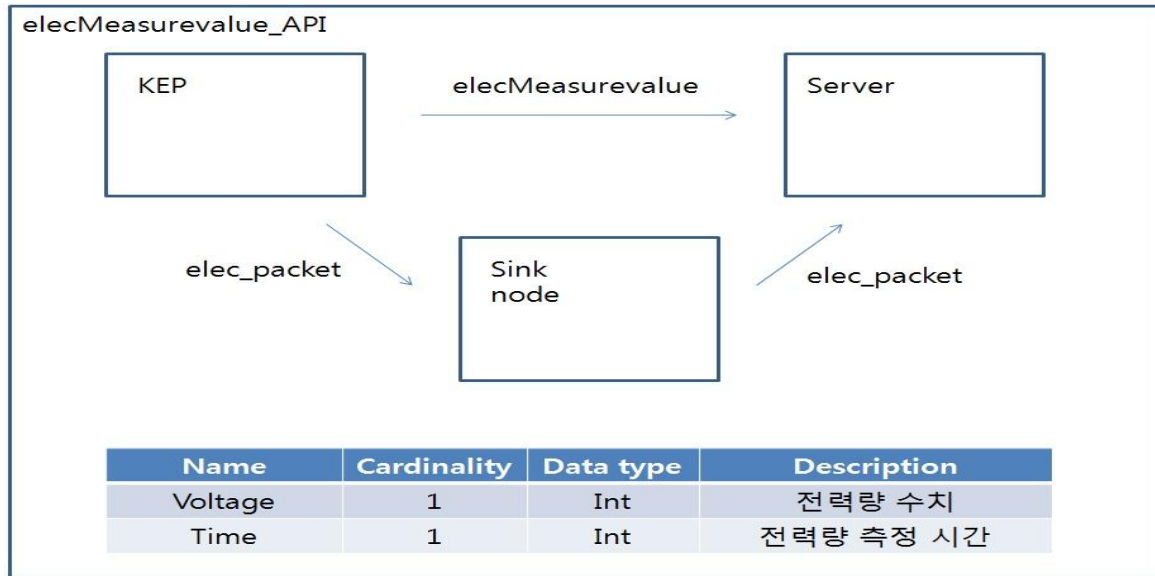


그림 26 elecMeasurevalue API

3.2.2 IF-2(Module A-> Module B 간 Interface)

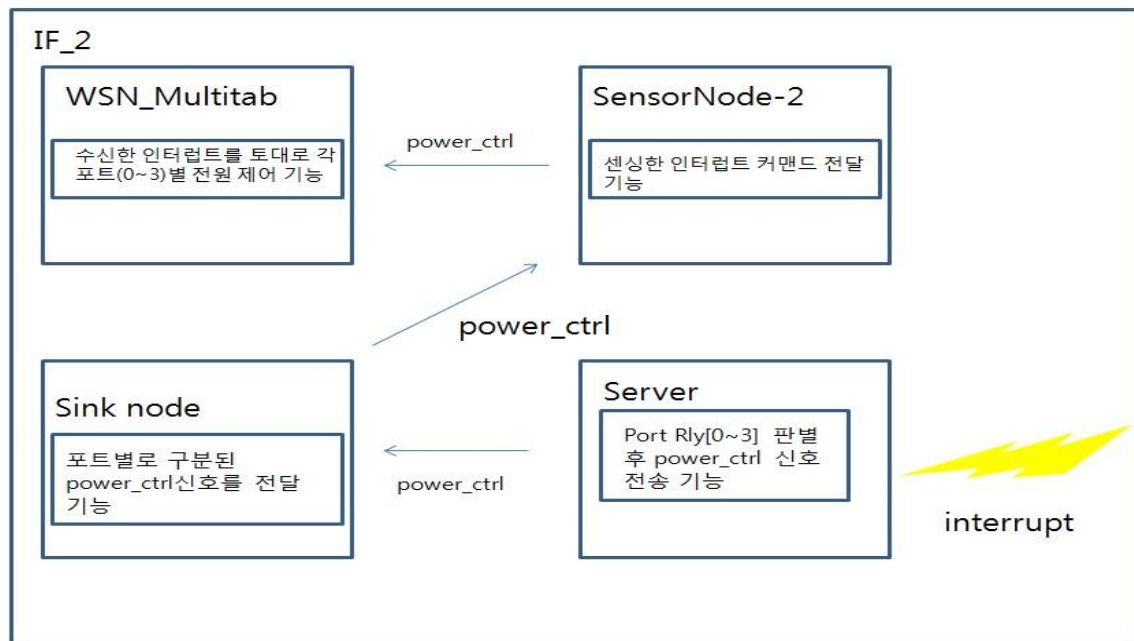


그림 27 IF-2 의 기능

웹페이지의 사용자, 혹은 threshold value로부터 오는 인터럽트로 서버는 전원 제어를 원하는 포트를 구별 후 그에 맞는 포트의 전원제어 신호를 싱크노드로 전달하게 된다. 전원 제어 신호를 송신받은 싱크노드는 센서노드로 그 패킷을 전달하게 되고, 그 정보를 다시 센서노드는 WSN-Multitab으로 최종 전달하게 되어 전달받는 WSN-Multitab은 포트를 구분하고 해당 포트의 전원을 제어하게 된다. 다음은 IF-2를 위한 powerctrl의 API이다.

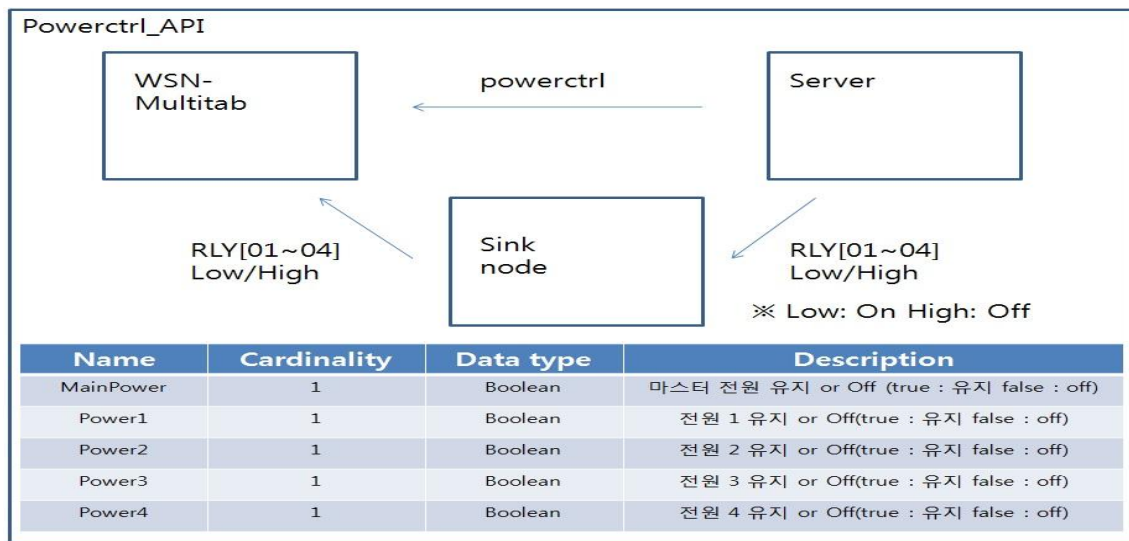


그림 98 Powerctrl API

3.2.3 IF-3(Module B-> Module C 간 Interface)

다음은 IF-3에 대한 설명이다. 앞서 B-3 에서 elec_packet에서 전력에 관한 데이터를 추출한 후 (packet[18]~packet[36]) 16진수인 이 데이터를 10진수로 변환하고 아울러 어느 노드에서 온 패킷인지 까지 식별하는 과정을 거친 후에 B-4에서 이 데이터를 DB에 저장하는 과정을 진행하게 된다. 그리고 난 후 웹 페이지에서 사용자가 원하는 포트의 전력량을 표로 수치화하여 보일 수 있으며, 또한 DB에 저장되어있는 전력량과 시간을 가지고 간단한 그래프로 도식화 할 수도 있다. 다음은 전력측정시스템에서 웹 사용자에게 표현되어야 할 DB 테이블이다.

Electricvalue Table			
Elecvalue	Type	Time	Type
elecvalue_1	Varchar(20)	Time1	Varchar(30)
elecvalue_2	Varchar(20)	Time2	Varchar(30)
elecvalue_3	Varchar(20)	Time3	Varchar(30)
elecvalue_4	Varchar(20)	Time4	Varchar(30)

그림 29 Electricvalue table

3.2.4 IF-4(Module C-> Module B 간 Interface)

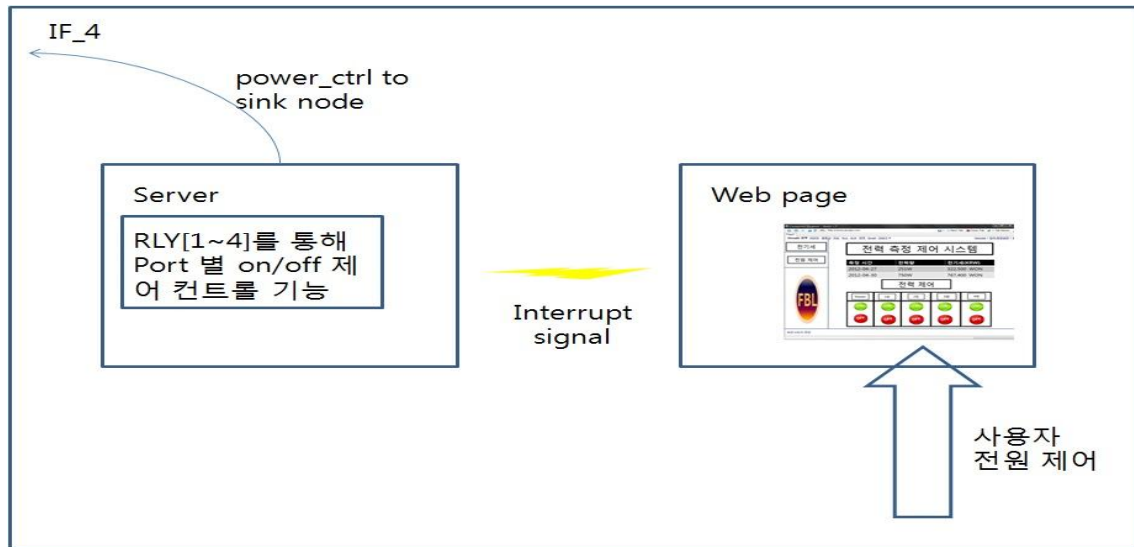


그림 30 IF-4 의 기능

다음은 IF-4에 대한 설명이다. 웹 페이지에서 사용자가 IF-3과정을 거쳐서 전력량을 확인 한 후에, 과도한 전력을 사용하고 있거나, 혹은 설정한 임계값의 수치를 근접하는 포트를 발견하게 될 경우 사용되는 인터페이스이다. 먼저 사용자로부터 전원제어 인터럽트가 들어오게 되고, 이 인터럽트 신호를 서버에 전송하게 된다. 서버는 RLY 수치를 통해 포트별 전원 on/off 제어를 결정하고, 해당 포트의 전원을 on 또는 off 하는 명령을 싱크노드로 보내게 되며, 최종적으로 WSN-multitab에서 전원제어를 하게끔 한다.

또한 IF-4의 기능은, 웹 페이지 뿐만 아니라 안드로이드 애플리케이션을 이용하여 스마트폰의 환경에서도 역시 수행이 가능하다. (TBD)

4. 개발환경

전력 측정 및 제어 시스템의 개발에서 OS로는 Linux, TinyOS가 쓰이고, Tool은 Eclipse를 사용한다. 언어로는 JSP, Servlet, nesC를 사용한다.

4.1 OS

4.1.1 TinyOS

네트워크 내장형 시스템을 위해 특별히 디자인된 초소형 OS 이다 . 핵심 OS 코드는 4000 바이트 이하이고, 데이터 메모리는 256 바이트 이하이며 이벤트 기반 멀티 태스킹을 지원한다. 센싱노드와 같은 초 저전력, 초소형, 저가의 노드에 저전력, 적은 코드 사이즈, 최소한의 하드웨어 리소스를 사용하는 내장형 OS를 목표로 하며, 내장형 네트워크를 위한 프로그래밍 언어로는 nesC 가 사용된다.

4.1.2 Linux

1991년 11월에 리누스 토르발즈(Linus Torvalds)가 버전 0.02을 공개한 유닉스 기반 개인컴퓨터용 공개 운영체제이다. 유닉스와 거의 유사한 환경을 제공하면서 무료라는 장점 때문에 프로그램 개발자 및 학교 등을 중심으로 급속히 사용이 확대되고 있다.

4.1.3 Tool & Utility

4.1.3.1 VMware

VMware는 하나의 OS 안에 여러가지 OS를 설치하여 운용하는 것을 말한다. 한 OS의 사용자가 다른 OS를 한 작업장에서 사용해야 하는 경우 필요로 한다.

4.1.3.2 Eclipse

Eclipse 는 Java의 원활한 개발을 위해 만들어진 개발 툴이다. 텍스트 에디팅 기능을 제공함과 동시에, 자동완성기능, 실시간 디버깅 기능 등 다양한 편의 기능을 제공한다. 또한 플러그 인이라는 아주 편리한 시스템을 도입하여, 플러그인 설치만 하면 여러 가지 언어와 환경을 구축하기 굉장히 편리하도록 준비가 되어있다.

4.1.4 Database

4.1.4.1 MySQL

MySQL은 임베딩하기 좋으며 깔끔한 SQL 인터페이스를 제공한다. 또한 메모리를 적게 사용하여 속도가 빠르다. 소스코드와 실행파일이 무료로 공개되어있다.

4.1.5 Languages

4.1.5.1 Servlet

서블릿은 Java 플랫폼에서 컴포넌트를 기반으로 한 웹 애플리케이션을 개발할 때 활용하는 중요 기술이다. 웹 애플리케이션 연동 시 서버에 존재하는 DB에서 애플리케이션 정보를 가져오기 때문에 페이지에 동적으로 생성할 필요가 있다. 이때 필요한 기술이 서블릿이다.

4.1.5.2 JSP

서블릿은 HTML코드가 자바 코드 안에 들어가기 때문에 프로그래밍 작업의 효율성을 떨어뜨리는 큰 단점이 존재한다. 이를 보완하기 위해 본 프로젝트에서는 서블릿 소스에서 처리하던 HTML을 JSP 스크립팅 기술로서 처리 하였다. JSP를 사용하면 컴파일 과정이 없어지며 개발 및 관리가 수월해 진다는 장점을 가진다.

4.1.5.3 nesC

NesC는 컴포넌트 모델언어로 여러 개의 컴포넌트를 묶어 하나의 애플리케이션 형태로 조

합하고, 센서노드로 다운로드 할 때에는 꼭 필요한 라이브러리 및 커널 컴포넌트들만 선택해 컴파일 하기 때문에 코드크기가 작다는 장점이 있다. NesC로 작성된 코드는 NesC 컴파일러에 의해 C언어로 변환된 후 GCC에 의해 바이너리 파일로 컴파일 된다.

4. 동작설명

4.1 Sequence Diagram

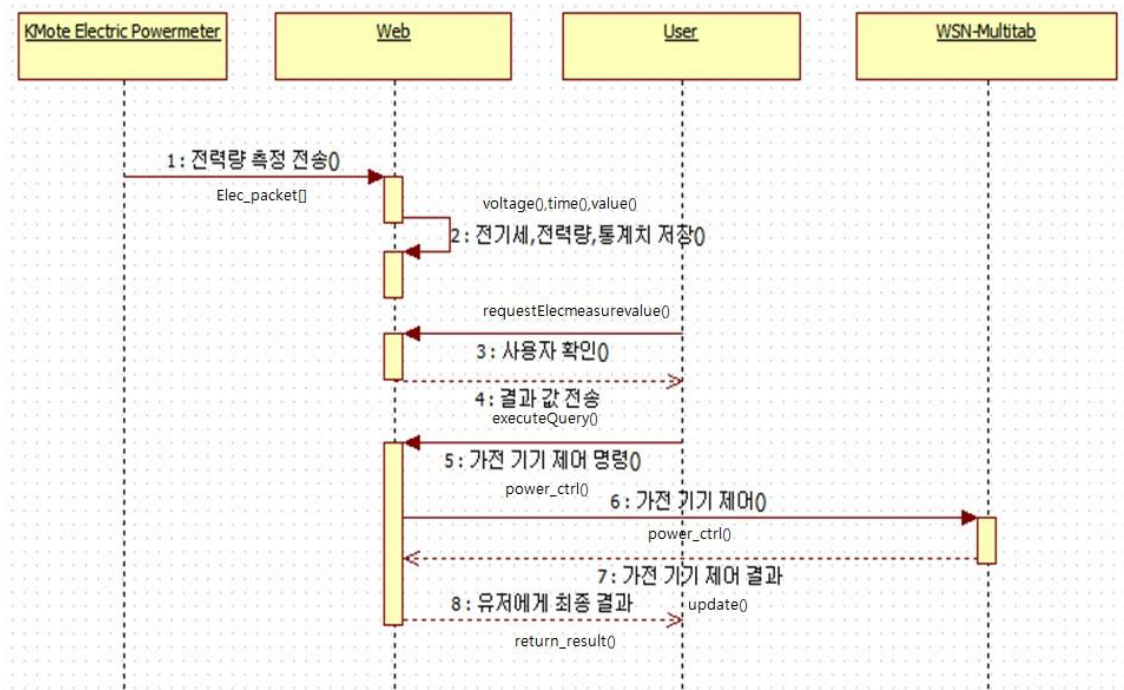


그림 31 전력 측정 제어 시스템 Sequence Diagram

다음은 전력 측정 제어 시스템의 기본적인 시퀀스 다이어그램이다. (1) 에서 Knode Electric Powermeter는 실시간으로 전력량을 측정해 41바이트의 Elec_packet[] 패킷 데이터로 변환시켜 전송한다. 이 때, Elec_packet은 16진수로 이루어져 있으며, Elec_packet[18]~[36]까지가 전력량을 나타낸다. 웹에서 이 패킷을 수신받아 파싱한 후 각각의 데이터들을 데이터베이스에 저장한다.(2) 사용자가 전력량이나 전기세를 확인하기 위한 명령을 내리는데, 이때 사용되는 함수는 requestElecmeasurevalue()이다.(3) 명령을 받으면 Web은 명령에 따른 결과 값을 전송한다.(4) 웹페이지나 스마트폰에서 유저가 가전 기기 제어 명령을 내리면(5) power_ctrl() 함수를 통해 Web에서 WSN-Multitab까지 명령이 전달되어 명령을 수행하고,(6) 가전기기 제어 결과를 다시 업데이트 시킨다.(7) 최종적으로 가전기기 제어의 결과를 유저에게 알려준다.(8)

4.2 동작 설명

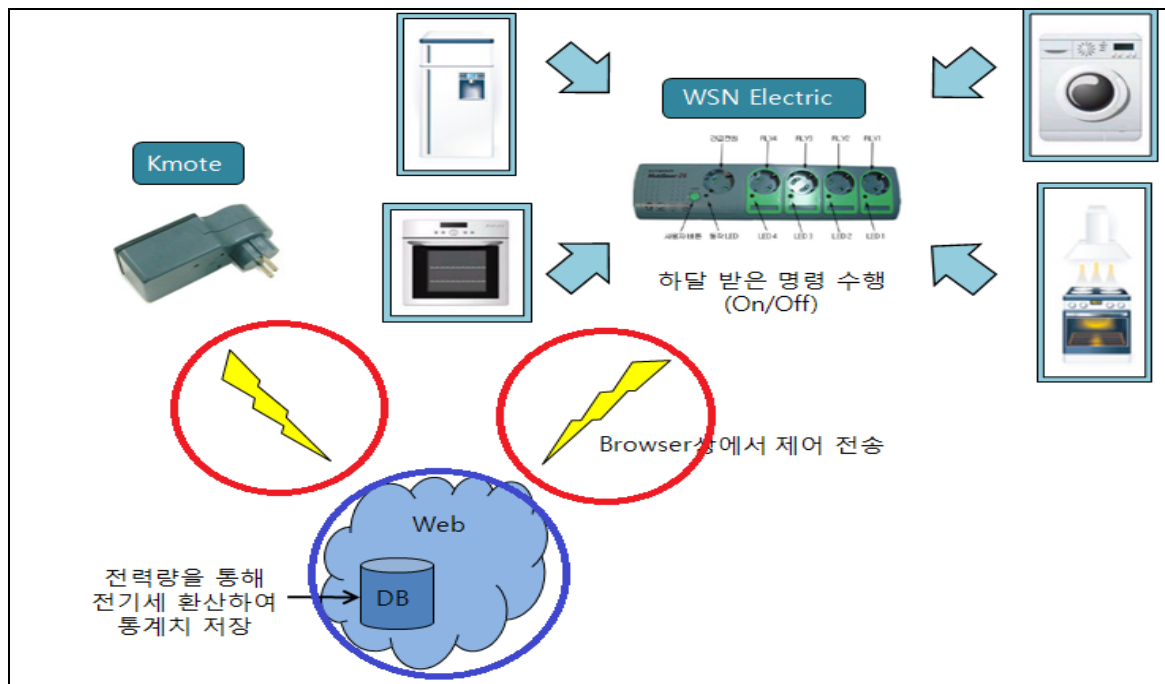


그림 32 동작 설명 그림

위 그림에서 가장 중요한 부분은 바로 원 표시로 표시한 부분이다. 2.1에서 시스템 기능 구성도에서 두 가지의 API가 빨간색 원 부분의 API(elecMeasureValue, power_ctrl)이다. KEP와 WSN-Multitab과 서버와의 통신을 위한 API이기 때문에 실시간으로 웹 서버에 각 정보들을 보내줘야 하고 서버에서는 이를 업데이트 하기 위해 중요한 부분을 담당한다. 파란색 원 부분 또한 서버 내부적으로 계산하는 기능을 가지는 함수를 가져야 한다. 또 이정보들[전력량(voltage), 전기세(value), 시간(time)]은 웹 서버내의 DB에 저장되어야 한다(ElectricValue Table).

- elecMeasureValue API : K-mote Electric Powermeter에서 측정한 전력량과 그 측정시간을 서버로 전송한다
- power_ctrl API : 웹 에서 사용자가 실시간으로 WSN-Multitab으로 전원 Off메시지를 전달하면 그에 해당하는 Master전원 혹은 slave전원의 Boolean 정보로 유지 혹은 전원 off정보를 전달한다.

5. 자체시험 방안

5.1 시험 환경

본 프로젝트에서 가장 중요한 점은 전력량 측정하는 것과 전력제어 하는 것을 각 포트별로 분리하여 시행할 수 있느냐는 것이다. 프로젝트의 취지가 바로 전자기기별로 전력량을 측

정하고 측정한 전력량의 정보를 토대로 각 기기별 전기세 정보를 얻어낼 수 있고, 각 기기별로 임계치를 통한 전원제어를 할 수 있어야 하기 때문이다. 따라서 프로젝트의 시험 환경을 위해 하나의 포트에 대한 전력량을 측정할 수 있는 KEP의 특성 상 여러대의 KEP가 필요하고, 전원제어를 위한 WSN-Multitab이 필요하다. 본 프로젝트에서는 3대의 KEP와 1대의 WSN-Multitab을 통해서 구현을 진행한다.

5.2 KEP(K-mote Electric Powermeter)

실시간으로 전력이 측정되는지 오실로스코프 프로그램으로 확인한다.

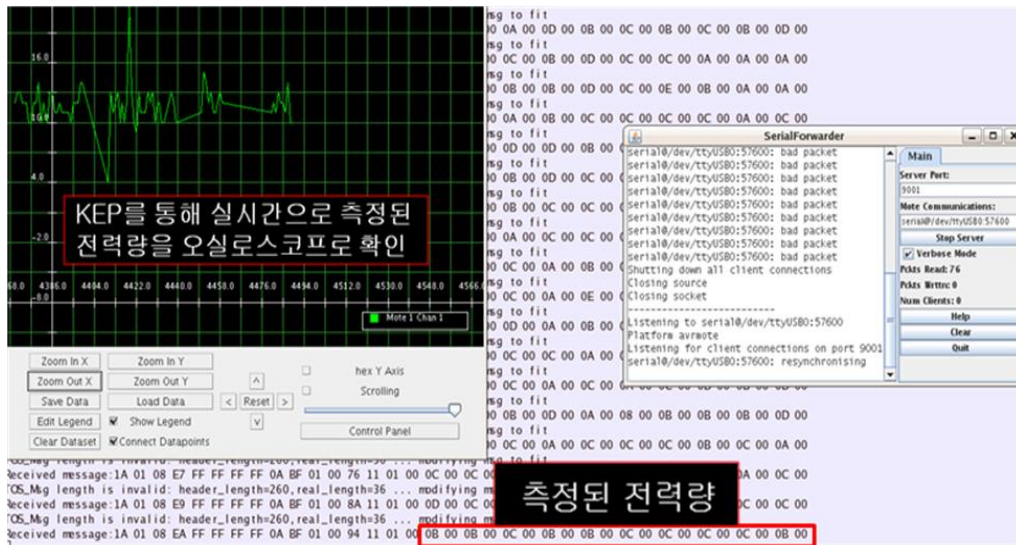


그림 33 콘센트 연결 중인 상태

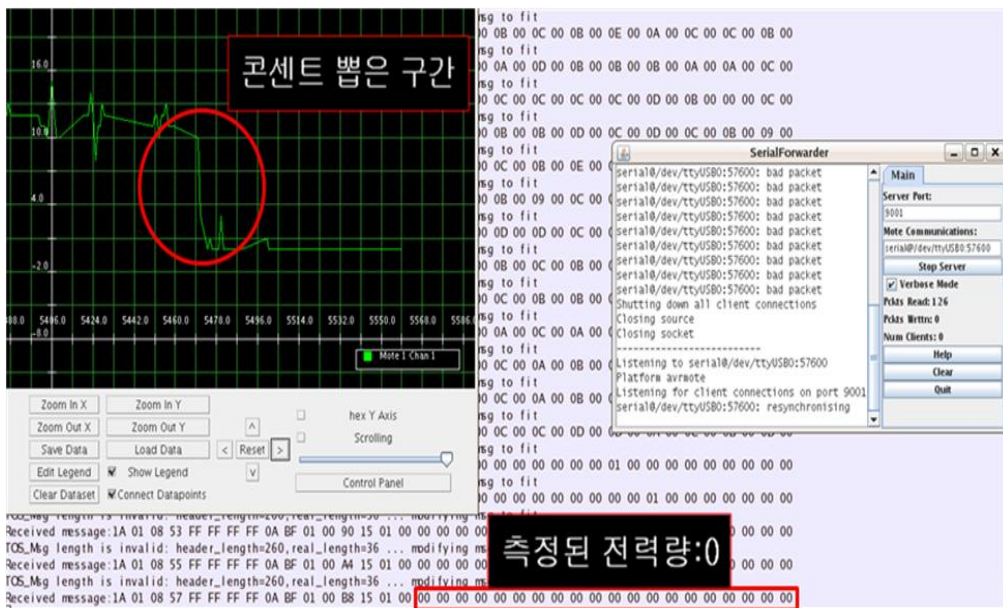


그림 34 콘센트 연결 중에서 연결 해제 할 때

5.3 WSN-Multitab

- RLY[1~4]를 통해 각 포트별로 전원 제어가 되는지 확인한다.
- 임계 값 초과에 따른 자동적인 전원 제어가 되는지 확인한다.

5.4 Server(Database)

- KEP를 통해 들어오는 전력량 패킷을 저장한다.
- 서버로 전송되는 정보가 웹 서버의 DB에 저장이 되는지 확인한다.
- 이 저장된 패킷 데이터를 이용하여 문제가 되는 전력량이나, 콘센트의 위치를 확인할 수 있다.
- 전력량의 최대 적정 임계값을 정하여 DB에 저장한다.

5.5 User Interface

- 적정 임계값을 넘어가면 Web상에서 초과위험수위를 확인할 수 있고, 알람을 통하여 사용자에게 알릴 수 있다.
- 웹 페이지를 통해 전자기기의 전원을 제어할 수 있다.
- 또한 안드로이드 애플리케이션을 통한 스마트폰 으로도 같은 기능(알람, 전원제어)을 수행할 수 있다.
- 웹 페이지 상에서 직접 전원을 껐을 때 전원을 연결한 전자기기가 꺼지는지 확인한다.

5.6 시연 시나리오

WSN-Multitab 각각의 포트에 KEP를 연결하고, 그 위에 전자기기의 콘센트를 연결한다. 먼저 전원 컨트롤에 대한 시나리오이다. 임계치를 임의로 설정하고, 임계치가 넘어가면 자동적으로 전원이 제어가 되는 것을 보여준다. 또한, 사용자의 필요로 인한 전원제어의 시연을 위해 웹페이지나 스마트폰을 통하여 강제 전원제어가 가능한지 보여준다. 두 번째로 실시간 전력량 측정에 대한 시나리오이다. 데이터베이스에 저장된 전력량을 토대로 그래프화시켜 웹페이지에 보여준다. 마지막으로 전기세 예측에 대한 시나리오이다. 서버에서 가공한 예상 전기세를 웹페이지에 일정한 시간대별로 업데이트 시켜 사용자에게 보여준다.

6. 향후 실제 적용방안

- 일반 가정의 전자제품들의 전력량을 각각 측정하며, 특정 전자제품이 전력의 위험수위(임계값)에 다다르게 될 경우 해당 기기에 대한 전원 제어를 통해 불의의 사고를 사전에 방지할 수 있다.
- 데이터베이스에 저장되어있는 전력량을 토대로 전기세를 예측해낼 수 있다.
- 또한 일반 가정에서 전원을 끄지 못하고 외부로 나왔을 때 전자제품의 전원을 원격으로 컨트롤 할 수 있다.
- 빌딩이나 공장과 같은 범위가 넓은 지역에서는 그 중앙 통제실에서 각각의 전자제품의 전원을 웹을 통해 쉽게 컨트롤 할 수 있다.

- 대기전력 소모량이 심한 곳에서 원격제어 On/Off 컨트롤을 통해 대기전력량을 줄일 수 있다.

7. 기대 효과

요즘 가정이나 빌딩, 혹은 공장 내에서 매 순간 얼마만큼의 과도한 전력을 사용하는지, 그에 따라 어떠한 위험을 초래할 것인지 인식 하지 못한 채 과도한 전력을 사용하고 있다. 적정 임계 값을 넘어선 과도한 전류의 흐름은 기계를 고장나게 하고, 심지어는 화재도 발생시켜 인명 피해의 원인이 되기도 한다. 이러한 일을 방지하기 위하여 전력 측정 시스템을 이용하여 전력을 측정함으로써 어떤 전자제품에서 순간적으로 얼마만큼의 전력이 사용되는지 도식화된 그래프와 위험 알람 등을 통하여 미리 알 수 있다. 또한 측정된 전력량을 바탕으로 예측 전기세를 산출해 낼 수 있다. 앞서 말한 그래프 혹은 위험 수위 알람을 통해 전력량을 확인한 후 과도하게 전력이 흐르는 전자제품에 대한 전원 제어를 할 수 있다. 추가적으로 만약 사용 후 전원을 꺼야하는 전자제품을 잊고 외부로 나갔을 때에는 Web을 통한 원격제어로 전자제품의 On/Off를 제어할 수 있다. 이러한 시스템을 이용하여 불필요하게 소비되는 전력량을 줄일 수 있다. 요약하면, 과도한 전류의 흐름을 사전에, 혹은 직후에 확인하여 해당 기기에 대한 전원 제어를 할 수 있을 뿐만 아니라, 바쁜 현대인들의 생활에서 다시 내부 안으로 가서 전원을 제어할 필요 없이 전자기기의 전원을 제어할 수 있다면, 또한 납부해야할 전기세를 예측할 수 있다면 더 편리하고 효과적으로 전력량을 줄일 수 있음을 기대할 수 있다.



그림 35 전력 측정 제어 시스템 기대 효과



8. 세부추진계획 및 일정

세부 내역 \ 월	3 월			4 월			5 월			6 월		
주제 설정 및 구상	→											
주제 확정 및 발표	→											
사용 장비 프로세서의 이해			→	→								
웹 개발을 위한 이해			→	→								
개인 임무별 구현					→	→	→	→				
통합 구현							→	→	→	→		
결과 발표 및 최종 시연											→	

9. APPENDIX

9.1 TinyOS and nesC

TinyOS

TinyOS의 정의

TinyOS는 우선 오픈소스이며 BCD-license형의 저전력 시스템에 최적화 시킨 운영체제이다. 주로 센서 네트워크에 쓰이는데, 유비쿼터스 컴퓨팅이나 PAN환경, 스마트빌딩 등을 구축하는데 일조하는 센서네트워크 환경에 사용한다. 현재 전세계에서 그 목적이 맞게 널리 사용이 되고 있다.

1. The simplest tinyOS programming

목적 : tinyOS programming에 적응하기 위해 가장 간단한 방법을 예시로 들어 설명을 하고 이 틀에 맞추어서 어떻게 구체적인 프로그래밍으로 확장시켜 나갈 것인가에 대해서 서술한다.

There is some value in knowing the simplest code that can be compiled without errors. The C equivalent is

```
int main () {  
    return 0;  
}
```

which is written in a file, say, simple.c and compiled with the command

```
$ gcc test.c
```

which produces the executable file a.out, which of course does nothing!

In TinyOS, to get the same thing, **you need to create three files**. Suppose the program we create is called Simple.

0. Create a new directory to put the files. We can name this directory *Simple*:

```
$ mkdir Simple
```



```
$ cd Simple
```

1. You need to create a *Configuration* file SimpleAppC.nc (following the suggested naming convention).

```
configuration SimpleAppC{  
}  
implementation{  
    components SimpleC, MainC;  
  
    SimpleC.Boot -> MainC.Boot;  
}
```

There are two components in this program: your component called SimpleC and the *Main* component MainC. The MainC component provides the Boot.booted signal which essentially is the entry point of the application.

2. You need to create the *Component* file SimpleC.nc. This has definition (implementation) of the component SimpleC.

```
module SimpleC{  
    uses interface Boot;  
}  
  
implementation{  
    event void Boot.booted()  
    {  
        //The entry point of the program  
    }  
}
```

3. Now you need to create a *Makefile* so that the compiler can compile it. Create the file called Makefile with the following two lines:

```
COMPONENT=SimpleAppC  
include $(MAKERULES)
```

I.e., you put the name of the top level configuration in the *COMPONENT* field.

Now you are ready to compile:

```
$ make micaz
```

which should work successfully provided you have set up the environment properly.

Now you can start extending this skeleton code by adding more components.

nesC Programming Hints, Condensed

다음은 TinyOS에 쓰이는 언어인 nesC언어의 프로그래밍에 있어서 주의해야 할 점, 혹은 힌트 등을 정리해 놓은 것이다. 아무래도 저전력, 저사양의 하드웨어에 적용되는 TinyOS와 nesC 언어이기 때문에 경제성과 효율성을 동시에 고려해야 할 필요성이 있다. 따라서 다음에 보일 내용은 nesC를 활용하는데 있어서 앞서 말한 것과 같은 목적을 달성하기 위한 유용한 정보들을 간단히 정리 해 놓았다.

Programming Hint 1: It's dangerous to signal events from commands, as you might cause a very long call loop, corrupt memory and crash your program.

Programming Hint 2: Keep tasks short.

Programming Hint 3: Keep code synchronous when you can. Code should be async only if its timing is very important or if it might be used by something whose timing is important.

Programming Hint 4: Keep atomic sections short, and have as few of them as possible. Be careful about calling out to other components from within an atomic section.

Programming Hint 5: Only one component should be able to modify a pointer's data at any time. In the best case, only one component should be storing the pointer at any time.

Programming Hint 6: Allocate all state in components. If your application requirements necessitate a dynamic memory pool, encapsulate it in a component and try to limit the set of users.

Programming Hint 7: Conserve memory by using enums rather than const variables for integer constants, and don't declare variables with an enum type.

Programming Hint 8: In the top-level configuration of a software abstraction, auto-wire Init to MainC. This removes the burden of wiring Init from the programmer, which removes unnecessary work from the boot sequence and removes the possibility of bugs

from forgetting to wire.

Programming Hint 9: If a component is a usable abstraction by itself, its name should end with C. If it is intended to be an internal and private part of a larger abstraction, its name should end with P. Never wire to P components from outside your package (directory).

Programming Hint 10: Use the `as` keyword liberally.

Programming Hint 11: Never ignore combine warnings.

Programming Hint 12: If a function has an argument which is one of a small number of constants, consider defining it as a few separate functions to prevent bugs. If the functions of an interface all have an argument that's almost always a constant within a large range, consider using a parameterized interface to save code space. If the functions of an interface all have an argument that's a constant within a large range but only certain valid values, implement it as a parameterized interface but expose it as individual interfaces, to both minimize code size and prevent bugs.

Programming Hint 13: If a component depends on unique, then `#define` a string to use in a header file, to prevent bugs from string typos.

Programming Hint 14: Never, ever use the "packed" attribute.

Programming Hint 15: Always use platform independent types when defining message formats.

Programming Hint 16: If you have to perform significant computation on a platform independent type or access it many (hundreds or more) times, then temporarily copying it to a native type can be a good idea.

nesC의 특징

위의 힌트들을 기반으로 nesC 프로그래밍의 특징에 대해서 간단히 서술하고자 한다. 힌트들을 보면 알 수 있듯이, nesC는 component(컴포넌트)기반의 프로그래밍 언어인데, 이에 따른 여러가지 장점들이 있다. Component들을 말 그래도 어떤 제품을 만들기 위한 부품이라고 상상을 한다면 그 장점이 쉽게 이해가 되는데, 부품들을 조립을 하면 제품이 만들어 지듯이, nesC의 component들을 잘 조립하면 하나의 애플리케이션 소프트웨어로 탄생시킬 수 있다는 점이다. 또한 이러한 component들을 재사용성이 굉장히 높기 때문에, 이미 만들어 놓은 부품들을 다른 방식으로 조립하여 이전 목적과는 다른 다양한 애플리케이션 소프트웨어로 재탄생 시킬 수 있다.

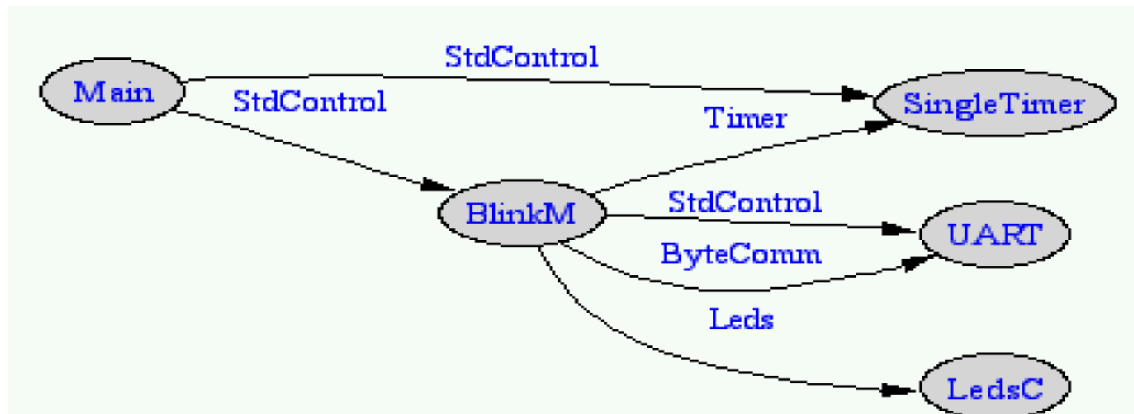


그림 10 컴포넌트의 연결

nesC의 함수는 command와 event라는 두 가지 함수를 사용한다. Command는 일반적인 c 언어의 함수와 비슷하고, 명령의 방향이 상위 소프트웨어에서 하위 하드웨어로 호출된다. Event는 하드웨어 동작에 대한 이해가 필요한데, command(명령)에 따른 기능 수행 후에 발생하는 이벤트를 호출하는 함수이다.

다음은 2-level structure 구조이다.

2-Level Structure

Command (Bottom direction)

Non-time critical

Long running operations

Cannot preempt & can be preempted

Background computation

Events (Top direction)

Time critical

Small running operations

Cannot be preempted

Able to interrupt running Tasks

nesC의 특징은 아래와 같으며 제약적인 프로그램 메모리와 램 메모리, 즉 저사양의 하드웨어에 적용되기 때문에 대부분의 경우 static(정적)메모리 할당을 하며, 컴파일 시에 플랫폼에서 사용될 메모리의 크기를 결정한다. 또한 nesC는 C와 동일한 문법을 따르고 있고, nesC에 추가된 개념이 Component(컴포넌트)와 Interface(인터페이스)이다.

Network Embedded System C Language

Supports TinyOS

Make applications for Network Embedded system

No dynamic memory allocation
Extension of C programming language
Efficient code for micro-controllers
Able to interact with old C code
Many C Programmer
C is little helpful for safe code & structuring applications

다음은 콤포넌트의 구성에 대해 알아보고자 하는데, 콤포넌트는 세부적으로 Module, Configuration으로 구성되며 인터페이스들을 통해 해당함수들이 구현된다.

Module

xxxM.nc
Code file

Configuration

xxx.nc/oooC.nc
Wiring of components
Define wiring of Modules

Interfaces

xxx.nc
Contains only definition
Commands/Events

Reference

- [1] TinyOS programming. Phillip Levis. June 28,2006
- [2] nesC 기초 프로그래밍 분석. 고원식,강정훈, 2008
- [3] TinyOS 프로그래밍 KETI Ubiquitous Technology Reserch Center, 강정훈, 유준재, 윤명현, 이민구, 임호정. 2007.
- [4] Wikipedia for TinyOS

9.2 스마트 그리드

만약 꼭 필요한 만큼 전기를 생산하거나 생산량에 맞춰 전기를 사용할 수 있다면 전기를 더 효율적으로 사용하면서 지구온난화를 막을 수 있을 것이다. 이러한 것을 가능하게 해주는 것이 스마트그리드이다. 스마트 그리드란 지능형 전력망'을 뜻하는 용어로, 기존 전력망(발전→송배전→판매)에 정보기술(IT)을 접목하여, 전력공급자와 소비자가 양방향으로 실시간 정보를 교환하고 에너지효율을 최적화하는 차세대 전력망을 말한다. IT기술이 발전하면서 에너지 부문에서도 양방향 통신 접목이 가능해지고, 태양·풍력 등 출력이 불규칙한 신재생전원의 보급을 확대시킬 수 있다는 점에서 많은 관심을 끌고 있다. 지능형 전력망의 가장 큰 장점은 에너지를 효율적으로 사용할 수 있다는 것이다. 예를 들면 집안 세탁기는 가장 싼 전기 요금 시간대에 맞춰 작동하고, 전기 자동차는 주간에 주차해도 심야에 맞춰 싼 요금으로 충전한다. 또 소비자 전력관리장치를 통해 전기사용 행태나 전기요금 등을 실시간으로 살펴볼 수 있어 소비자의 자발적인 에너지절약에도 도움이 된다.



▲ 지능형 전력망 ‘스마트그리드’ 개념도

폐쇄적이고 획일적이었던 공급자 중심의 기존 전력망에 정보통신 기술을 접목해 실시간 정보 교환을 함으로써 개방적이고 양방향성을 가진 수요자 중심의 다양한 서비스가 가능해 지는 것이다. 또한 신재생에너지, 전기차 등 청정 녹색기술의 접목, 확장이 용이한 개방형 시스템으로 산업간 융, 복합을 통한 신 비즈니스 창출이 가능해 진다.

스마트 그리드의 주요 역할



소비자에게 다양한 전력 정보를 전달해 효율적인 전력소비 유도
전기자동차가 전력을 충전하고 방전하는 시스템 구축
신재생에너지에서 생산된 전력을 안정적으로 공급
정전의 경우 관련 전력망을 부분적으로 단절시켜 정전구간 최소화
남북한 간 또는 동북아시아 국가 간 전력망 연계 가능
직류 송전이 가능해 가전제품의 전기효율 향상

스마트 그리드의 필요성

전기 공급량은 정확하게 예측할 수 없기 때문에 수요가 뚜렷하게 나타나는 계절을 기준으로 전기를 공급할 수밖에 없다는 단점이 있다. 언제까지 예측만으로 전력을 공급할 수는 없기 때문에 스마트 그리드는 이러한 문제를 해결해줄 수 있다는 점에서 에너지 공급의 긍정적인 방향이 될 것이다. 지금까지의 전력이 공급자 중심이었던 폐쇄형이었다면 스마트 그리드를 통해 수요자 중심인 개방형으로 바뀌게 된다.

에너지의 효율

신 . 재생 분산형 전원의 보급 확대

무정전, 고품질 전력서비스 제공

에너지 . 환경 문제의 주요 솔루션

신성장 동력으로서의 먹거리 창출

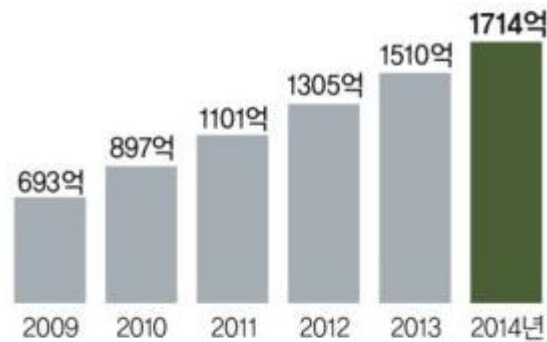
국민 녹색생활 혁명 가속화

스마트 그리드의 장점

스마트 그리드가 상용화된다면 전력 사용 현황을 실시간으로 파악할 수 있게 된다. 즉, 공급자는 실시간 사용 현황을 보며 공급량을 조절하는 것이 가능하게 되고 소비자는 전력 요금이 싼 시간대에 전력을 사용하는 것이 가능해진다.

스마트 그리드 시장규모

■ 스마트 그리드 시장 규모(단위: 달러)



■ 스마트 그리드 시장 분야별 규모
(단위: 달러)



자료: 삼성경제연구소

시장이 겨우 형성되는 초기 단계이지만 2014년에 시장이 1700억 달러 대로 급팽창할 것으로 예상돼 기업들이 앞 다투어 관련 사업에 뛰어들고 있다고 한다. 스마트 그리드는 기업이나 가정의 효율적인 전력망 구축 외에도 전기자동차 충전기, 스마트 가전제품 등 다양한 분야에서 이용 가능할 것이라 하니 새로운 사업 영토를 구축하려는 기업들의 무한 경쟁을 예상할 수 있다.

9.3 스마트 그리드 국내외 동향

스마트 그리드 해외 동향

가. 미국

세계 각국은 국가적 차원에서 차세대 전력망에 대한 검토를 이루어지고 있으며, 미국은 2000년 전력대란, 2001년 대규모 정전사태 및 2003년 중동부지역 정전사태를 경험하고 이와 관련된 법안들을 마련하였다. 미국에서의 스마트 그리드는 미국 전력중앙연구소(ERPI)의 인텔리 그리드, Modern Grid Initiative, GridWise 등 10여 개가 넘는 기관과 단체에서 전력시스템의 지능화 및 선진화에 대해 지속적인 연구의 결과물로 탄생한 것이다.

나. 유럽

유럽 각국은 신 재생에너지 중심의 분산형 전원의 보급확대, 환경 보전, EU국가간 전력 거래에 중점을 두고 스마트 그리드를 추진하고 있으며, 특히 풍력 등 신 재생 에너지의 보급 확산에 적극적이다. 미국과 달리 공통 아키텍처와 다양한 통신 기술의 상호 운용성을 보장하는 표준이 마련되어 있지 않으나 최근 스마트 그리드 표준에 대한 움직임이 나타나고 있는데, 프레임워크 연구 프로그램을 통하여 스마트 그리드 플랫폼 개발을 착수하였다.

다. 일본

일본은 Cool Earth-21을 통해 주요 에너지 혁신 기술 개발을 추진 중이며, 2050년까지 온실가스 감축 50% 달성을 위해 21개 핵심기술을 선정한 바 있다. 에너지 안정공급의 확보, 재생가능 에너지의 원활한 도입과 효과적인 활용, 수용 가측과 일체화된 에너지 절약 및 에너지 유효 이용을 실현하기 위한 시스템으로서 스마트 그리드 TIPS(Triple "I" Power System)를 추진 중이다.

라. 기타 국가

중국 최대 전력사인 'State Grid Corp'는 2009년 기술표준을 시작으로 2020년까지 스마트 그리드를 구축할 계획이며, 인텔과 GE 등과 협력하여 스마트 그리드 프로젝트를 진행 중이다. 6.6%의 송전과정 손실률을 감소시키고 약 11백만 킬로미터의 노후화된 송전선 문제를 해결하기 위해 스마트 그리드 추진 계획을 국가전망공사 주관으로 추진 중이다.

호주는 ISG(Industry Strategy Group)를 설립하여 AMI(Advanced Metering Infrastructure) 규격을 제정하였으며 2009년 5월부터 Country Energy와 IBM은 약 10만 가구를 대상으로 파일럿 프로그램을 추진 중이다. 현재까지 시장은 스마트 계량기 차원으로 국한되어 있었으나, 빅토리아주의 스마트 그리드 인프라 개발투자와 함께 점차 스마트 그리드 구축 사업으로 확장되고 있다.

이와 같이 각국의 정책이나 기술 동향을 종합/분석해 보면, 미국은 노후화된 전력 인프라의 교체와 스마트 홈과 같은 수용가 서비스 제공, 유럽은 신 재생에너지 중심의 분산형 전원의 보급 확대 및 EU 국가간 전력거래, 그리고 일본은 에너지 관리 측면에서 초점이 맞추어져 있는 것으로 판단된다.

스마트 그리드 국내 동향

2009년 2월 대통령 주재 녹색성장위원회 1차 보고에서 '세계 최초 국가 단위의 지능형 전

력망(Smart Grid) 구축'에 대한 국가 비전 발표를 시작으로 다양한 정책들이 수립되었으며 국내외 스마트 그리드 기술에 대한 관심이 고조되고 있는 상황이다.

Reference

- [1] 스마트 그리드 기술 동향. 이일우, 박완기, 박광로, 송승원, 한국전자통신연구원. Sep. 2009
- [2] Wikipedia for Smart Grid

10. 안드로이드(Android)

10.1 안드로이드의 개념

세계 각국의 이동통신 관련 회사 연합체인 '오픈 핸드셋 얼라이언스(OHA ; Open Handset Alliance)'가 2007년 11월에 공개하였다. 실질적으로는 세계적 검색엔진 업체인 구글(Google)사가 작은 회사인 안드로이드사를 인수하여 개발하였으며, 따라서 '구글 안드로이드'라고도 한다.

안드로이드는 리눅스(Linux) 2.6 커널을 기반으로 강력한 운영체제(OS ; operating system)와 포괄적 라이브러리 세트, 풍부한 멀티미디어 사용자 인터페이스, 폰 애플리케이션 등을 제공한다. 컴퓨터에서 소프트웨어와 하드웨어를 제어하는 운영체제인 '윈도'에 비유할 수 있는데, 휴대폰에 안드로이드를 탑재하여 인터넷과 메신저 등을 이용할 수 있으며, 휴대폰뿐 아니라 다양한 정보 가전 기기에 적용할 수 있는 연동성도 갖추고 있다.

안드로이드가 기존의 휴대폰 운영체제인 마이크로소프트의 '윈도 모바일'이나 노키아의 '심비안'과 차별화되는 것은 완전 개방형 플랫폼이라는 점이다. 종전에는 휴대폰 제조업체와 서비스업체마다 운영체제가 달라 개별적으로 응용프로그램을 만들어야 하였다.

이에 비하여 안드로이드는 기반 기술인 '소스 코드'를 모두 공개함으로써 누구라도 이를 이용하여 소프트웨어와 기기를 만들어 판매할 수 있도록 하였다. 개발자들은 이를 확장, 대체 또는 재사용하여 사용자들에게 풍부하고 통합된 모바일 서비스를 제공할 수 있게 된 것이다. 안드로이드를 탑재한 휴대폰 단말기를 안드로이드폰이라고 하며, 이 플랫폼에서 응용할 수 있는 애플리케이션을 거래하는 온라인 공간을 '안드로이드 마켓'이라고 한다. 미국의 시사 주간지 《타임》은 모토로라의 안드로이드폰 '드로이드(Droid)'를 2009년 최고의 디지털 기기로 선정하였다. 한국에서 처음 선보인 안드로이드폰은 2010년 1월에 출시된 모토로라의 '모토로이(Motoroi)'이다.

10.2 Android Application

보통 JAVA 언어로 개발하며 Android Software Development Kit을 이용한다. 또한 C나 C++을 이용한 Native Development Kit, Google App Inventor, 초보 프로그래머를 위한 visual environment, 또는 다양한 cross platform mobile web application frameworks등의 다른 Development kit을 이용해서도 개발할 수 있다.

Reference

[1]Wikipedia

11. Web

웹의 원래 의미는 「거미집」으로 하나의 사이트나 또는 다른 사이트와의 관계가 거미집 처럼 복잡하게 얽혀 있기 때문에 웹이라고 부른다. 이와 같이 월드 와이드 웹(WWW)은 세계 규모의 거미집 또는 거미집 모양의 망이라는 뜻으로, 하이퍼텍스트 기능에 의해 인터넷상에 분산되어 존재하는 온갖 종류의 정보를 통일된 방법으로 찾아볼 수 있게 하는 광역 정보 서비스 및 소프트웨어. WWW 또는 웹(web)이라고 부른다. 1989 년 스위스 제네바에 있는 유럽 원자핵 공동 연구소(CERN)의 버너스 리(Tim Berners-Lee)가 제안한 것으로, 인터넷을 이용하기 쉽게 만들어 크게 활성화한 주역으로 각광받고 있다. 그 이유는 웹이 문자 정보가 대부분이었던 이때까지의 통신에 의한 정보 전달 방법과는 달리 문자, 화상, 음성에 더하여 다양한 표현 방법을 가능하게 하였기 때문이다. 웹에서는 정보가 웹 서버라고 하는 컴퓨터 내에서 하이퍼텍스트 형식으로 작성되어 홈 페이지라는 단위로 관리되며, 링크라고 하는 정보에 의해 인터넷상에 분산되어 있는 세계 각지의 하이퍼텍스트와 연결될 수 있다. 현재 열려 있는 하이퍼텍스트 문서에 잘 모르는 단어가 있거나 그에 관련된 정보가 더 필요하면 링크에 의해 다른 하이퍼텍스트(홈 페이지)를 차례로 불러서 읽을 수 있다. 전 세계의 하이퍼텍스트가 이리저리로 연결된 모습이 마치 거미가 집을 지은 것처럼 보이기 때문에 월드 와이드 웹(WWW)이라는 이름이 붙여졌다. 하이퍼텍스트를 작성할 때는 하이퍼텍스트 생성 언어(HTML)를 사용한다. 클라이언트와 웹 서버의 통신 규약으로는 하이퍼텍스트 전송 규약(HTTP)을 사용한다. 웹 서버에는 CERN 판 이외에 전미 슈퍼컴퓨터 응용 연구소(NCSA)에서 개발한 것 등 여러 종류가 있다. 웹 서버에 있는 하이퍼텍스트를 볼 수 있게 하는 응용 소프트웨어가 브라우저인데, NCSA 에서 개발한 모자이크, 넷스케이프사에서 개발한 넷스케이프 내비게이터, 마이크로소프트사에서 개발한 인터넷 익스플로러 등이 있다. 이중에서 넷스케이프는 1994 년에 등장한 이래 단기간 내에 웹의 이용을 폭발적으로 증가시키고 인터넷을 크게 활성화한 주역으로 평가된다.

11.1 Web page

인터넷 상의 웹 문서들을 총칭한 말로 이 문서 속에는 다양한 텍스트는 물론 그림, 소리, 동영상 파일도 내장할 수 있다. 다른 인터넷 상의 문서와 서로 연결할 수 있게 해주는 강조된(highlighted) 글자나 그림 등이 있다는 것이 특징이다. 이러한 부분들은 주로 밑줄로 구분할 수 있으며, 파란색의 바탕선이 그려져 있기도 하다.

11.2 Web Server

웹서버는 클라이언트/서버 모델과 웹의 HTTP를 사용하여 웹 페이지가 들어 있는 파일을 사용자들에게 제공하는 프로그램이다.

가장 보편적인 웹서버로는 32 비트 윈도우와 유닉스 기반의 운영체제에서 모두 쓸 수 있는 아파치와, 윈도우 NT에 달려 나오는 IIS (Internet Information Server), 그리고 넷스케이프의 엔터프라이즈 서버 등이 있다.

1995년 NCSA 웹서버를 기반으로 탄생한 아파치는 "open source" 라이선스에 의거하여 배포되는 마음대로 쓸 수 있으며 현재 세계 웹서버 시장의 70%를 석권하고 있다. IIS(Internet Information Server)는 마이크로소프트의 서버 운영체제인 Windows NT에 포함되어 나오고 있으며 NT에서만 사용가능하다.

Reference

[1]Wikipedia

[2]네이버 지식사전

12. 원격제어(Remote Control)

기계 등을 직접 사람의 손이나 발로 조작 하지 않고 어떤 장치를 사용하여 간접적으로 조작하는 일. 원방제어라고 한다. 제어소(制御所)에서 전송기(傳送器) 또는 연락전송선 · 마이크로파 등을 통해서 원격조작을 하는 방식이다. 이 제어방식은 전력, 철강, 화학, 수송 등의 전체 산업분야에서 사용되고 있다. 이 방식으로는 보통 제어소에서 제어에 대하여 확실히 기기가 움직였는가, 그 상태는 어떤가 하는 반신(半信)을 받을 필요가 있어 무슨 방법으로든지 통신설비를 필요로 하는 경우가 많다. 따라서 이 방식을 원방감시제어라고 한다. 전송회로는 전송거리, 제어기기수, 지리적 조건 등에 의해서 다른데, 가공(架空) 또는 수중제어(水中制御) 케이블에 의한 직접연락, 전력선 통신선에 의한 반송연락(搬送連絡), 마이크로파 등에 의한 연락이 사용되고 있다. 사용하는 케이블의 수, 또는 사용하는 채널의 수가 한정된 전송회로로 많은 기기에 신호를 보내기 위한 경제적인 전송방식에 대해서는 여러 방식이 연구되고 있다. 예를 들어 제어소와 피제어소의 위치 릴레이가 보조를 맞추어, 선택된 위치까지 차례로 변환하게 한 동기방식(同期方式)이나 펄스의 수, 펄스의 극성(極性), 폭 등의 조합을 사용

하여 기기에 하나씩의 부호를 주어, 그 부호를 보내는 펄스 코드방식 등이 있다. 한편 원격 제어를 리모트콘트롤이라고 하는데 모형비행기나 모형선박을 떨어진 곳에서 무선조종하는 것은 그 한 예이다. 기계가공의 자동기계인 트랜스퍼 머신을 조종반으로 조작하거나 석유정 제공장에 따로 설치된 제어실에서 정제의 공정을 조작하는 것도 원격제어이다. 또한 사람이 직접 조작하기에는 위험한 경우에 원격제어가 응용된다. 방사선물질을 취급할 때 밀폐된 별실에서 조종기를 조작해서 다루거나, 제철공장에서 높은 온도로 가열한 동괴(銅塊)를 취급할 때에 멀리 떨어진 제어실에서 조작하는 것도 그 예이다.

Reference

[1]Wikipedia

[2]네이버 백과사전