**졸업 프로젝트**

**최종 보고서 (1차)**

-아두이노와 웹서버를 이용한 스마트홈 오토메이션-

고지혜 201311193

김예찬 201311200

**1. 개요**

**1.1 프로젝트 기획 배경**

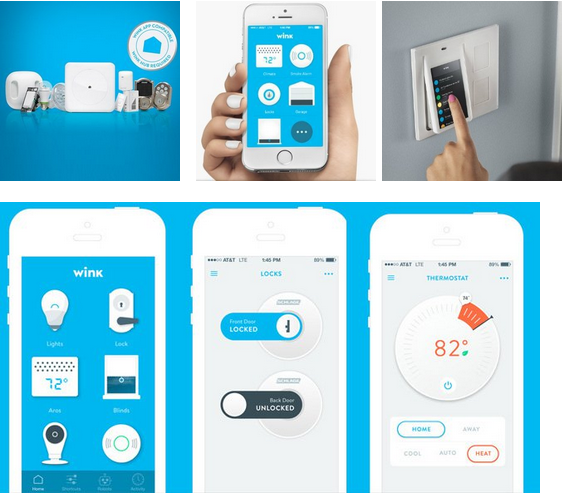
프로젝트의 주제에 대해서 교수님과 토론하는 중에 교수님의 연구실에 있는 아두이노를 발견하고, 이에 대해서 교수님과 다시 얘기를 해보면서 흥미 있는 주제라고 생각을 하게 되었다.

또한 현재 IT분야에서 스마트홈에 대한 주제는 미래 가망성이 높다고 평가되고 있어 프로젝트 의 주제로 적절하다고 생각하여 이번 프로젝트의 주제로 선정하였다.

**1.2 기술 동향**

최근 들어 미국, 중국 등의 해외 시장을 중심으로 스마트홈에 대한 소비자들의 관심이 높아지고 있다. 시장 조사업체 마켓앤마켓은 전 세계 스마트홈 시장이 2015년부터 연평균 17%씩 성장해 2020년에는 거의 600억 달러에 달할 것으로 내다보고 있다.

이렇게 많은 해외 시장에서 스마트홈에 대한 연구는 많이 진행되고 있다고 한다. 예를 들어, 미국의 스타트업 퀼키(Quirky)가 스마트홈 시장 진출을 위해 윙크(Wink)를 설립했는데, 퀼키는 단돈 50달러짜리 윙크 허브를 통해 다양한 브랜드의 스마트홈 제품을 연동해 제어할 수 있는 스마트홈 플랫폼을 만들었다. 이 윙크 허브를 통해 조명, 전원관리, IP 카메라, 화재감지, 도어락, 가전, 블라인드 등을 관리한다고 한다.



국내 기업에서도 스마트홈에 대한 주제로 연구가 활발하고, 대표적인 기업으로는 LG를 들을 수 있다. LG에서는 스마트싱큐(SmartThinQ) 라는 메인 센서를 가전제품에 설치하여 스마트폰과 태블릿으로 제어를 할 수 있고, 그 가전제품의 상태를 확인할 수도 있다고 한다. 또한 그 센서에는 진동을 포함한 4개의 센서를 가지고 있는데, 이를 통해 냉장고, 세탁기등의 진동, 온도를 확인하여 식품의 상태, 세탁의 완료 여부 등 더욱 세밀한 상태를 알 수 있다고 한다.



**1.3 프로젝트 주요 기능 및 특징**

**1.3.1 프로젝트 주요 기능**

1) 안드로이드 기기 등록

처음으로 사용하는 안드로이드 기기의 경우 등록 과정을 거친다. 해당 기기의 아이디를 홈 오토메이션 시스템의 데이터베이스에 저장하여, 등록된 아이디를 가진 기기가 명령을 보낼 때만 서버에서 명령을 아두이노로 보내게 된다.

2) 집안의 사물 상태 확인

집 안의 전자제품과 조명의 On/Off상태, 커튼의 열림/닫힌 상태 등 집 안의 사물들의 상태를 안드로이드 앱에서 보여준다. 사물들의 상태는 변경이 있을 때 마다 서브 아두이노에서 웹 서버로 보내어 안드로이드 앱으로 보낸다.

3) 집안의 사물 제어

안드로이드 앱에서 LTE및 Wifi를 이용하여 get 방식을 통해 서버로 정보를 보내면 그 정보를 데이터 베이스에 저장한다. 중앙 아두이노에서는 이더넷 쉴드를 통해 인터넷에 연결하여 서버에서 php 파일을 요청하여 읽어와 디비에 저장되어 있는 명령 중 최근 명령을 가져온다. 명령을 분석하여 해당 되는 물체의 서브 아두이노에 신호를 보내어 제어한다.

집과 멀리 떨어진 외부에서도 와이파이나 핸드폰 데이터 통신을 이용하여 서버에 접근할 수 있으므로 아두이노로 명령을 보내는 것이 가능해진다.

4) 자동 제어

4-1) GPS 체크

사용자가 외출한 후 집에 돌아올 때 에어컨, 보일러, 컴퓨터(부팅이 오래 걸리는) 등 사용자가 집에 도착하기 전에 미리 틀어놓으면 좋은 전자제품들은 사용자의 GPS의 위도 경도와 중앙 아두이노 의 GPS의 위도 경도의 거리를 계산하여 적정 값 이하일 때 에어컨을 틀도록 신호를 보낸다.

외출 시에 계속해서 서버로 GPS의 위도 경도를 보내게 되면 와이파이 환경이 아닌 이상 데이터 통신비가 많이 들게 되므로 GPS계산은 안드로이드 앱에서 한다. 외출 모드로 전환하면 서버의 데이터베이스에서 중앙 아두이노의 GPS값을 한 번 읽어온 후에 안드로이드 앱에서 현재 GPS위치와 중앙 아두이노의 GPS값을 비교하여 적정 값 이하일 때 서버로 명령을 보내도록 한다.

4-2) 행동 패턴 분석

데이터 베이스에 저장되어 있는 사용자가 보내온 명령들의 내용, 시간, 빈도 등을 바탕으로 행동 패턴을 분석하여 서버에서 적절한 시간에 적절한 명령을 내려 스스로 사물들을 제어할 수 있도록 한다.

**1.3.2 프로젝트 특징**

1) 데이터 처리가 많은 작업

대부분의 단일 보드 마이크로컨트롤러를 이용하여 스마트홈을 구현한 프로젝트 들은 메인 허브 기능을 하는 서버를 아두이노 자체에 서버를 두어 구현한 경우들이 많다. 그러나 이런 경우엔 아두이노의 용량 문제로 데이터 처리가 많은 작업들을 원활하게 할 수 없게 된다.

따라서 본 프로젝트에서는 허브 기능을 하는 서버를 아두이노가 아닌 컴퓨터에 웹서버를 두어 이용함으로써 데이터를 지속적으로 저장할 필요가 있는 행동 패턴 분석 서비스를 제공하는 작업들을 원활하게 할 수 있다.

2) 원거리 조종 가능

대부분의 아두이노를 이용한 무선 통신 프로젝트들은 와이파이 혹은 블루투스를 이용한 근거리 통신들이 대부분이다. 그러나 본 프로젝트에서는 아두이노와 사용자 사이에 웹서버를 두어 이용함으로써 사용자가 안드로이드 기기를 통해 원거리에서 아두이노를 제어할 수 있다.

3) GPS

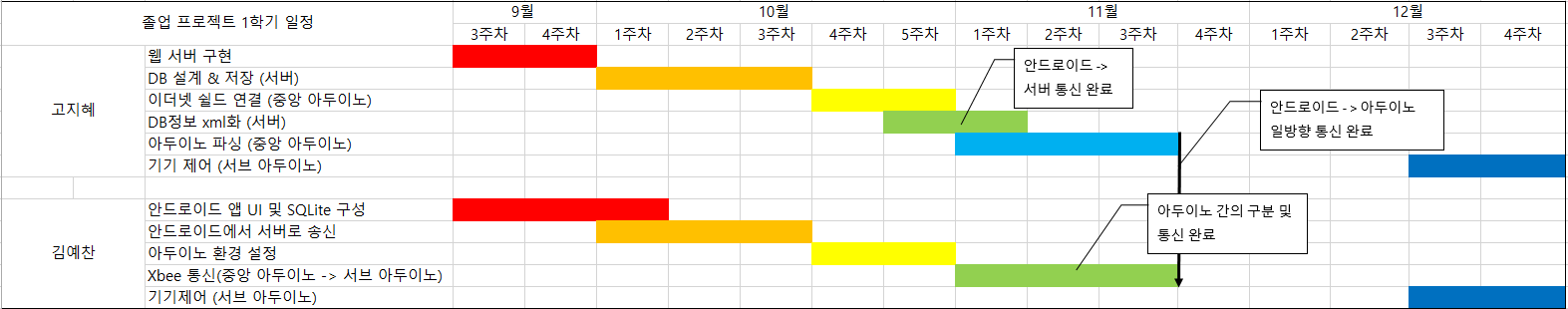
본 프로젝트에서는 스마트홈에 GPS 기능을 추가하였는데, 이는 사용자의 안드로이드 기기에서 중앙 아두이노까지의 거리를 계산해서 일정한 거리 이내에 오면 서버로 접근했다는 데이터를 전송하여 에어컨을 미리 켜놓는 것 같은 특정한 동작을 수행하도록 설정을 할 수 있다.

**1.4 조원 구성 및 역할 분담**

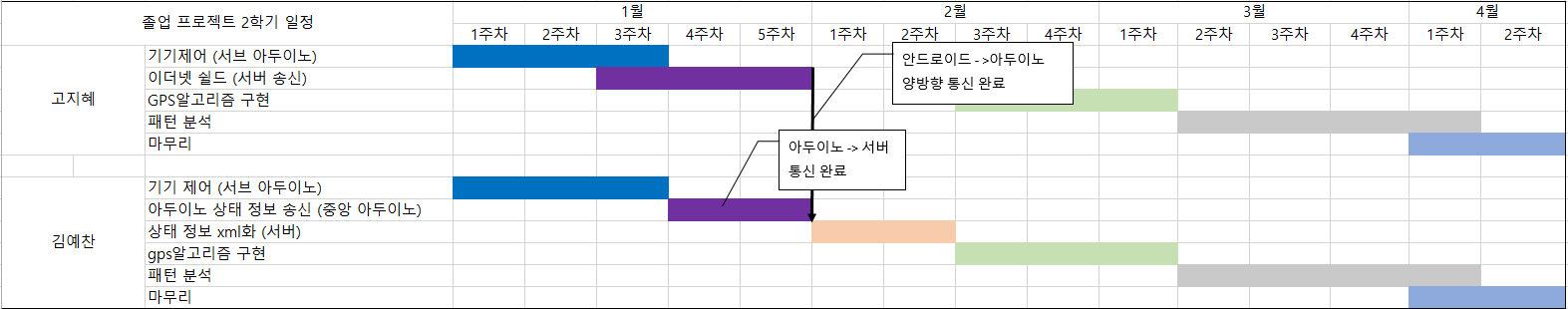
|  |  |
| --- | --- |
| 고지혜 | 김예찬 |
| 서버 사이드에서 명령 가공 및 DB저장  중앙 아두이노에서 서버로부터 명령 수신  중앙 아두이노에서 서버로 사물들의 상태 송신 | 안드로이드 앱에서 서버로 명령 송신  중앙 아두이노에서 서브 아두이노로 정보 송신  중앙에서 안드로이드 앱으로 사물 상태 송신 |
| 공통 | |
| 웹 서버 구축  사물 제어  GPS 알고리즘 | |

**1.5 일정**

1학기 일정



2학기 일정



**2. 진행 상황**

**2.1 DB설계**

웹 서버에 사용자가 안드로이드 폰에서 보낸 명령들, 등록된 아두이노 기기들을 저장할 DB를 만들았다.

먼저 스마트폰에서 보낸 명령들을 저장할 control테이블이 있다. Primary key인 num과 제어할 기계 이름인 cname, 제어 명령(기계를 On할 것인지, Off할 것인지 등)을 저장하는 corder, 날짜와 시간을 저장하는 date, 위도와 경도인 lat, lng, 마지막으로 명령을 보낸 스마트폰의 아이디를 저장할 aid칼럼이 있다.



<control 테이블>

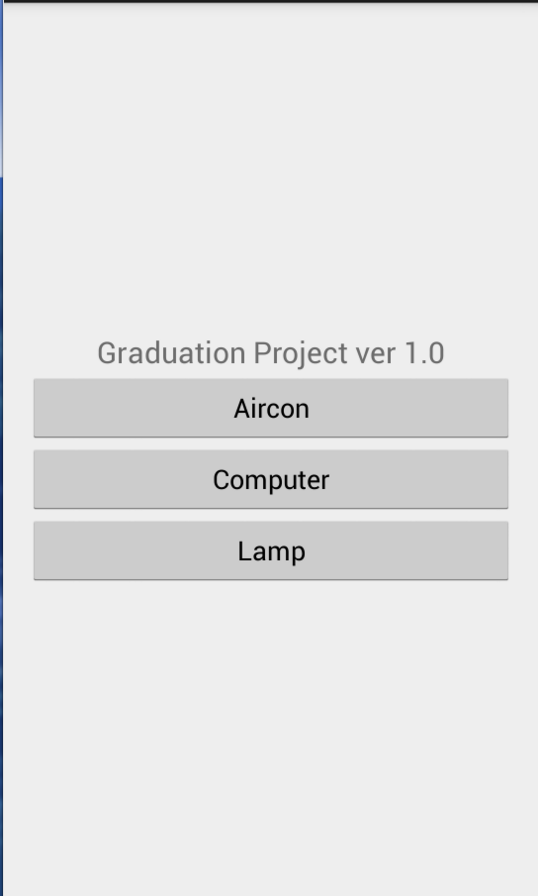
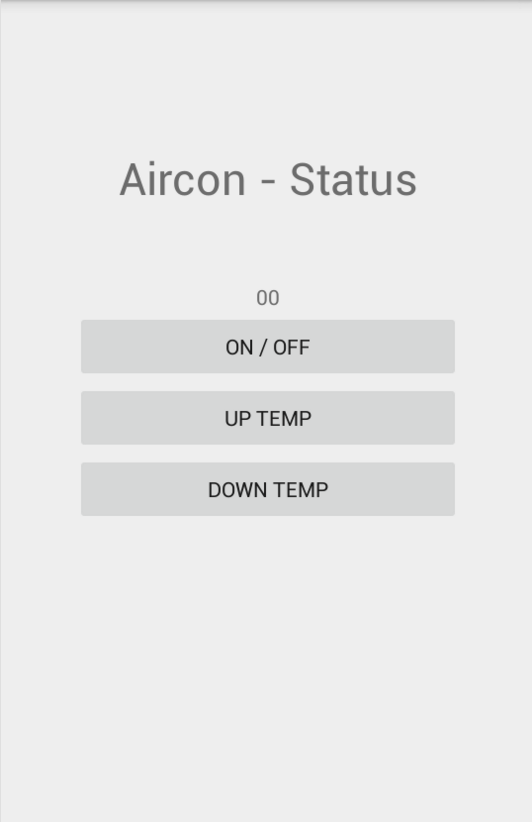
두 번째로는 현재 웹 서버에 등록된 안드로이드 기기들을 저장하는 registered 테이블이 있다. 단순히 아이디를 저장하는 aid칼럼이 있다.



<registered 테이블>

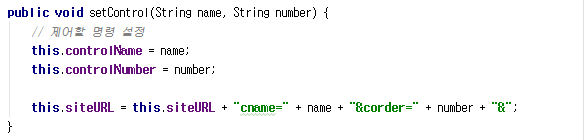
**2.2 안드로이드에서 웹서버로 데이터 전송**

사용자가 안드로이드 GUI에서 제어를 하고 싶은 제품과 제어를 할 명령을 선택을 하게 하고, 대략적인 GUI는 테스트용 버전으로 아래와 같이 만들었다.

사용자가 제어 신호를 보내면 M2S PACK이라는 서버로 보내는 클래스의 객체를 만들어 전송을 하였는데, 이 클래스는 멤버 변수로 제어할 기기, 제어 명령, 사용자의 위치 등을 가지고 있고 네트워크 통신을 할 때, main thread에서 작업을 하게 되면 사용자의 안드로이드 기기가 그 작업중에는 Block이 되는 상황이 일어나게 되어 재정의된 thread를 따로 가지고 있다.

네트워크 통신을 위한 thread에서는 유저가 선택한 정보(제어 기기,명령)를 본 프로젝트의 웹서버로 보내는 역할을 하고 단순하게 url connection을 하여 전송을 하였다.



< 사용자가 선택한 데이터를 url에 업데이트 하는 부분 >



< 네트워크 thread에서 url을 open하는 부분 >

**2.3 DB에 저장하는 php**

명령을 DB에 저장하기 위해 안드로이드 기기에서 리퀘스트할 php페이지를 만들었다.



<android.php>

안드로이드에서 GET방식으로 리퀘스트 보낸 것을 받아 변수에 저장하고, 테이블에 넣어주는 php코드이다.

**2.4 DB에서 꺼내오는 php**

DB에서 가장 최근의 명령을 가져와서 페이지로 작성하기 위해 select 쿼리를 보내는 php페이지를 만들었다.



<getControl.php>

DB를 연결한 후 테이블을 num순으로 정렬했을 때 가장 위에 있는 명령, 즉 가장 최근에 들어온 명령을 가져온 후 xml형식으로 출력하여 중앙 아두이노에서의 파싱이 쉽게 이루어지도록 하였다

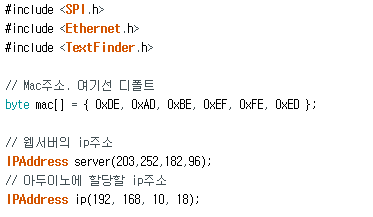
**2.5 중앙 아두이노에서 웹페이지 파싱**

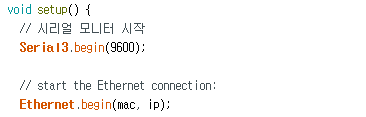
**2.5.1 이더넷 쉴드 적층**



이더넷 쉴드는 아두이노 보드를 인터넷에 연결할 수 있게 해준다. 쉴드 같은 하드웨어들은 조금만 연결이 잘못되어도 제대로 동작하지 않았기 때문에 신중하게 적층시켜야했다.

이더넷 쉴드를 이용하여 중앙 아두이노를 인터넷과 연결시키기 위해 SPI.h와 Ethernet.h의 라이브러리를 추가하여준다. 중앙 아두이노에 ip주소를 할당해 주고 Ethernet.begin으로 이더넷 연결을 시작한다. 이 때 할당해주는 ip는 현재 사용되지 않는 ip주소여야한다.





<ParsingFromServer.ino 일부분>

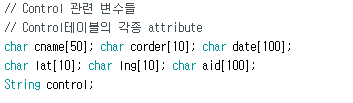
**2.5.2 웹 페이지 리퀘스트**

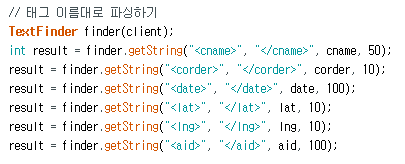
이더넷 연결이 되면 반복문을 돌면서 클라이언트를 생성하여 웹 서버에 연결한다. 연결이 되면 HTTP request를 보내 getControl.php 페이지를 요청한다. 연결지향적인 통신이 아니기 때문에 계속해서 client를 connect해야했다.



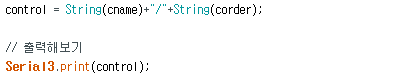
**2.5.3 TextFinder로 파싱**

파싱을 쉽게 만들어 주는 TextFinder라이브러리를 추가한다. Client와 연결된 TextFinder객체를 만들고 태그 이름 사이의 내용을 getString으로 읽어온다. 읽은 값들은 해당되는 변수에 저장한다.



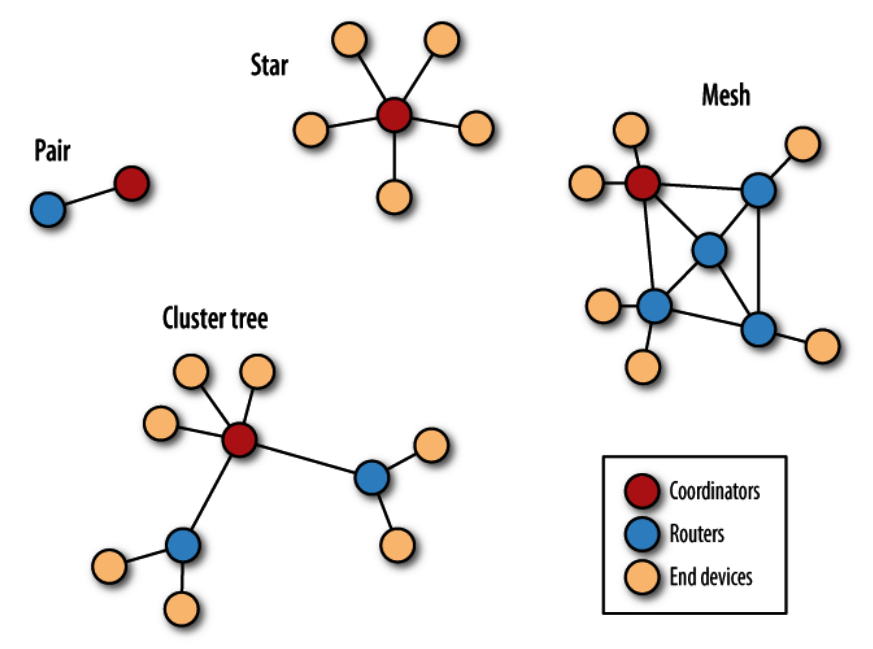


그 후엔 변수의 값은 Xbee 통신을 사용하여 Serial3에 write해주어 서브 아두이노들로 보내준다.

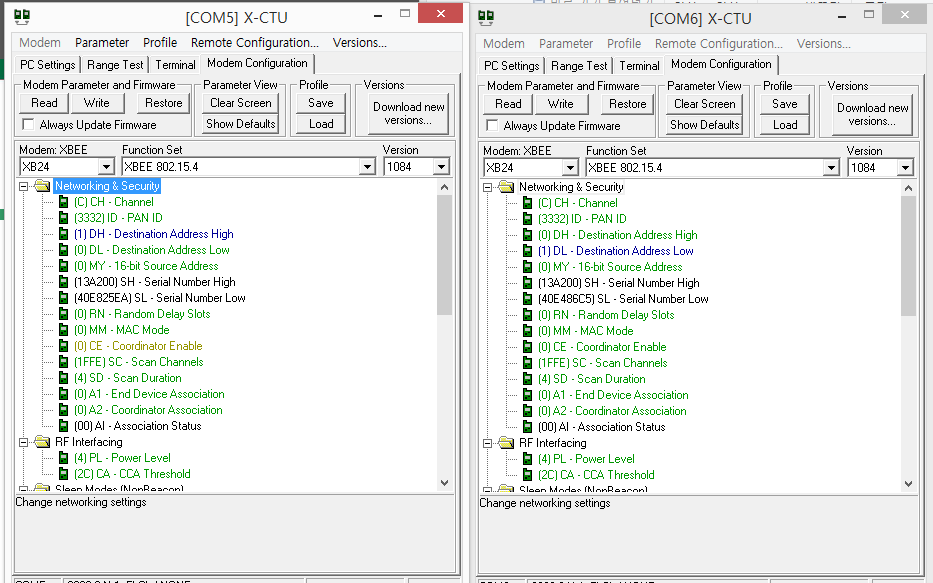


**2.6 Xbee 모듈 설정**

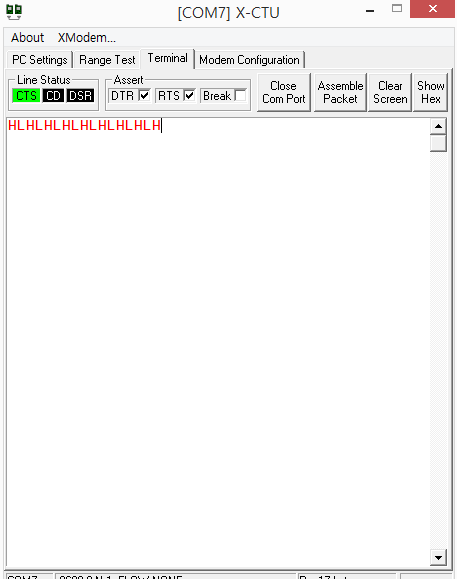
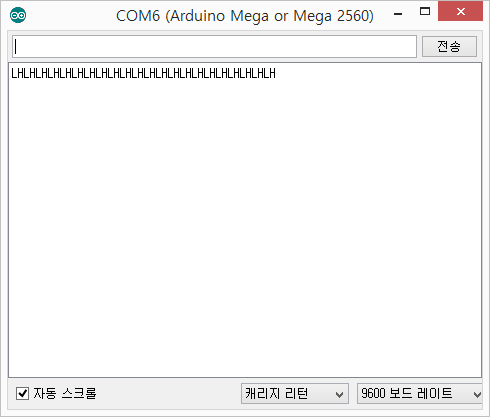
Xbee간의 코딩을 하기 위해선 Xbee 쉴드에 올라가는 Xbee 모듈들을 설정을 해주어야 한다. 즉, 보드에 올라가는 Xbee 모듈이 통신에 있어서 어떠한 역할을 할 것 인지를 설정해주어야 하는데 일반적인 Xbee 네트워크 형태는 아래와 같다.



본 프로젝트에서는 이더넷 기능을 하는 중앙 아두이노에서 다른 서브 아두이노와 통신을 하는 것이기 때문에 Star 형태의 네트워크를 갖도록 중앙 아두이노는 Coordinator로 나머지 서브 아두이노는 End device로 설정하기 위해서 각각의 모듈의 Channel, PIN 코드, 통신속도와 목적을 설정하였다.



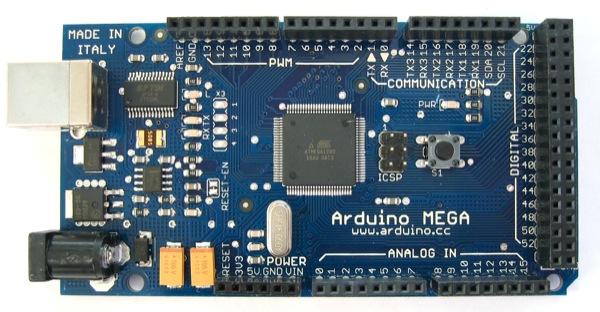
< X-CTU를 이용한 Xbee 모듈 설정 >

  
< X-bee 모듈 시리얼 테스트 >

**2.7 Xbee 통신**

아두이노에서는 다른 아두이노와 통신을 할 때 시리얼을 이용한 통신을 사용하고, 이 시리얼 통신은 크게 두 가지로 나뉜다. 하나는 하드웨어 시리얼이고 다른 하나는 소프트웨어 시리얼이다. 시리얼 통신의 경우 비 동기 전송방식을 취하기 때문에 ‘수신 버퍼’ 라는 공간과 그에 따른 제어가 필요하다. 하드웨어 시리얼은 이에 대한 하드웨어가 준비되어 있기 때문에 시스템에 영향을 주지 않고 안정적으로 수신이 가능하다. 반면 소프트웨어 시리얼의 경우엔 시스템의 자원을 할당해서 시리얼 통신이 들어오는지, 또 시리얼 통신이 들어왔다면 이것을 따로 지정한 버퍼에 저장하는 과정 등을 수행하게 된다. 따라서 소프트웨어 방식은 하드웨어 방식과 비교해서 안정성, 효율성 등이 매우 떨어지게 된다.

아두이노의 보드 종류는 다양한데, 본 프로젝트에서는 아두이노 MEGA 2560(이하 메가)을 사용하였다. 메가는 다른 아두이노와 시리얼 통신이 가능하도록 시리얼 포트가 0번에서 3번까지 제공이 된다. 우리는 위에서 언급한 소프트웨어 방식의 단점을 생각하여 하드웨어 방식을 사용하였고, 3번 포트를 중앙 아두이노를 기준으로 송신하는 포트로 사용하였고, 수신하는 포트로는 2번 포트를 사용하였다.



< 아두이노 MEGA >

이에 대한 모듈 테스트는 간단하게 아두이노 보드 3개 (중앙 아두이노, 2개의 서브 아두이노) 로 구현하였다. 중앙 아두이노에서 이더넷 쉴드를 통해서 서버로부터 읽은 데이터를 String 타입으로 / 로 나눠서 Serial 3 번 포트로 Xbee 모듈에 전달 하였다. 그러면 Xbee 모듈에 저장된 목적지 주소로 데이터를 전송하게 되고, 서브 아두이노에서는 전달 받은 String 타입의 데이터를 읽고 String 을 ‘/’ 로 분할하여, 제어 명령과 제어 기기를 구분하도록 하였다. 그리고 제어 명령에 따라서 8번 포트에 연결되어 있는 led를 digitalWrite하였다.



< 중앙 아두이노에서 서브 아두이노로 데이터를 전송하는 부분 >



< 서브 아두이노에서 중앙 아두이노로부터 데이터를 받는 부분>

**3. 보완할 점**

**3.1 서버 – 중앙 아두이노 통신 방식**

현재는 중앙 아두이노에서 서버로 2초정도의 딜레이를 두고 계속해서 리퀘스트를 보내는 pulling방식으로 명령을 가져온다. 따라서 불필요한 연결이 많았고 배터리도 빨리 닳았다.

다른 방식을 찾아본 결과 long-pulling방식이나 WebSocket방식이 있었다. 이러한 방식을 아두이노에서 지원한다면 써보기로 하였는데, 오히려 연결을 계속 하고있어야 한다는 점 때문에 마찬가지로 아두이노에 부담이 갈 수 있다는 의견도 있었다. 따라서 위의 방식들과 다른 부담이 가지 않는 적절한 방식을 연구하여 가장 최적의 방법을 찾아보기로 하였다.

**3.2 Xbee 통신 방식**

현재 프로젝트는 중앙 아두이노에서 지정한 Destination Low 로 데이터를 전송하는데, 이렇게 하게 되면 제어를 받지 않는 아두이노까지 데이터를 읽는 수행을 하게 된다. 이러한 방식으로 통신을 하면 제어를 받지 않는 아두이노가 배터리를 낭비를 하게 된다. 그래서 이후엔 좀 더 효율적인 통신 방법을 모색할 예정이다.

또한 필요한 데이터를 전송하는 과정에서 몇 개의 데이터는 잘 보내졌지만, 손실이 되는 경우가 존재한다. 그래서 Selective Repeat ARQ 방식처럼 데이터를 전송할 때 cumulative Ack를 받는 방식으로, 데이터 손실을 최소화 해볼 예정이다.

**4. 향후 일정**

겨울 방학을 포함하여 다음 학기부터는 위에서 언급한 보완할 점들을 보완한 후 본격적으로 기기제어에 들어갈 것이다. 현재로서는 제어 명령이 해당하는 사물에 장착될 아두이노로 가는지 led를 켜거나 끄는 것으로 확인을 했지만 실제로 모터나 센서 등을 달아서 사물을 제어해 볼 것이다.

그 후 사물의 상태를 알려주는 현재까지 구현한 것의 역방향의 통신을 구현할 것이다. 그 부분이 완료 되면 GPS 알고리즘과 패턴 분석 알고리즘을 구현하는 데에 많은 노력을 들일 것이다. 그 후 마지막으로 안드로이드 기기를 프로젝트 만의 key값을 만들어 등록하는 등 보안 쪽을 점검하고 마무리를 할 예정이다.

