Journal of the Korea Convergence Society
Vol. 8. No. 2, pp. 15-20, 2017 https://doi.org/10.15207/JKCS.2017.8.2.015

Esp8266모듈을 이용한 소형 데이터 통합 및 융합장치 설계

이동석, 임중수*
백석대학교 정보통신학부

Design of Compact Data Integration and Convergence Device Using Esp8266 Module

Dong-Seok Lee, Joong-Soo Lim*
Division of Information & Communication Eng., Baekseok University

요 약 본 논문에서는 Esp8266 모듈과 Node.js, TCP/IP 소켓 통신을 이용하여 소형경량 데이터 통합장치를 설계하였다. 본 장치는 Wifi연결 기능을 지원하는 Esp8266모듈을 사용하여 서버와 클라이언트를 구성하고, TCP/IP의 소켓 통신을 사용하여 양방향 데이터 전송을 지원하도록 구성하였다. 서버는 Node.js 운영체제를 사용하여 구성하고 Mysql을 사용하여 데이터를 통합할 수 있게 하였으며, 네트워크는 홈 네트워크와 같이 공유기를 중심으로 사설 IP를 부여하여 Esp8266이 각각 독립적인 IP를 가지게 설계하였다. 본 장치는 양방향으로 데이터를 전송 할 수 있고, 서버 측에서 각각의 클라이언트 데이터들을 저장할 수 있으며 또한 Wire-Shark를 통해 양방향으로 전송되어지는 데이터의 흐름을 확인할 수 있어서 소형의 실시간 데이터 통합 및 융합 장치로 활용할 수 있다.

• **주제어** : 사물인터넷, 소켓 통신, 네트워크, 센서, 통합, 융합

Abstract In this paper, Esp8266, Node.js, and TCP / IP socket communication are used to design a compact data integration device. This device is designed to configure server and client using Esp8266 module that supports Wifi connection and to support bidirectional data transmission using TCP / IP socket communication. The server is configured using the Node.js operating system, and the database is integrated using Mysql. The network is designed to have a separate IP address by assigning a private IP address to the router, such as a home network. This device can transmit data bidirectionally, store individual client data on the server side, and can check the flow of data transmitted bidirectionally through wire-shark, so that it can be used as a compact real-time data integration and convergence device.

• **Key Words** : IoT, Socket Communication, Network, Sensor, Fusion, Convergence

1. 서론

정보통신기술의 발전으로 스마트 폰, 웨어러블 컴퓨터, 지능형 로봇, Smart Home Service와 같은 지능적이고 사람들에게 편의를 제공하는 수많은 제품이 출시되어 일상생활에 많이 사용되고 있다. 이러한 기기들은 대부분 여러 개의 센서로 부터 데이터를 받아 정보를 저장하

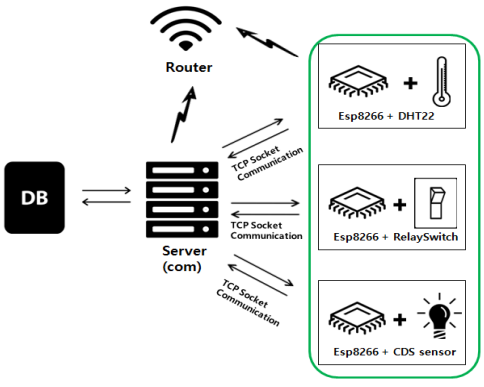
*Corresponding Author : 임중수 (jslim@bu.ac.kr)
Received January 11, 2017 Revised February 6, 2017
Accepted February 20, 2017 Published February 28, 2017

고, 정리된 각종 정보를 사용자에게 실시간으로 제공한다[1,2,3]. 이런 장치들은 기존의 스탠드-온 컴퓨터(PC)를 이용하여 센서 데이터를 수집하는 방법에서 범용 프로세서나 암(ARM) 프로세서를 이용하여 데이터를 수집하는 단계로 소형화되어가는 추세이다. 본 연구에서 이러한 센서와 네트워크의 결합을 Esp8266 모듈과 DHT22 센서 등을 이용하여 소형.염가로 구현하고, Wifi 연결과 TCP/IP (Transmission Control Protocol/Internet Protocol)의 소켓 통신을 이용하여 데이터의 양방향 통신 및 데이터 흐름을 실시간 감시할 수 있도록 하였다[4,5]. 또한 서버에 Mysql을 사용하여 각종 센서의 데이터를 저장하고 저장된 센서 데이터를 표준데이터 또는 과거 데이터와 비교하여 정상값과 경계값에 대한 상관값을 추출하여 데이터를 융합할 수 있도록 하여 실시간으로 현장의 상황을 판단을 할 수 있도록 설계하였다.

2. 소형 데이터 통합장치 설계

2.1 서버와 클라이언트 구성

소형 데이터 통합 장치는 [Fig. 1]과 같이 하나의 서버에 내부 공유기를 중심으로 여러 개의 센서들이 연결되어 서로 연관되는 동작을 할 수 있게 구성되어 있다. 서버는 Esp8266 모듈과 구글의 V8엔진을 기반으로 개발된 Node.js를 이용하여 구축하였다. Node.js는 여러 모듈을 사용하여 서버구축이 가능한데, 이중 'net'모듈을 사용하여 TCP서버를 구축하였다. 클라이언트는 Esp8266 모듈과 각종 센서로 구축하였으며 ESPlorer 툴을 사용하여 운용 프로그램을 업로드 하였다.



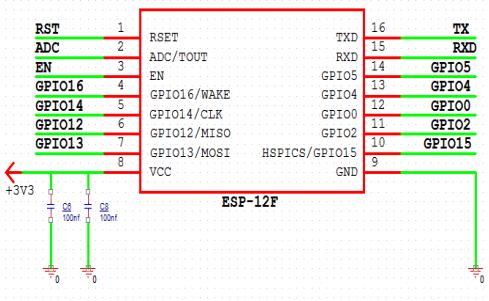
[Fig. 1] Server and client configuration of small data fusion system

TCP/IP 네트워을 통해서 서버에 도달한 센서 데이터는 Mysql에 모두 저장되어지며, 저장된 데이터를 상황을 판단할 수 있도록 통합하면 화재, 지진, 홍수 등의 유용한 정보를 추출할 수 있도록 설계되어 있다.

통합장치에서 UDP(user datagram protocl)로 데이터를 전송하지 않는 이유는 일방적인 데이터 전송이 아닌 특정 값이나 조건에 도달하면 서버에서 클라이언트로 제어용 데이터를 전송할 수 있도록 하기 위한 양방향 통신이 필요해서 UDP가 아닌 TCP를 사용하였다[6,7].

2.2 Esp8266 모듈 규격과 구성

Espressif사에서 개발한 Esp8266 모듈은 Wifi 통신 기능을 독립적으로 지원하고 모바일 플랫폼 개발자를 위한 무선 SoC(System-on-Chip)로서 Esp8266EX을 기반으로 크기가 매우 작고 저 전력을 소모하므로 소형.염가로 IoT(Internet of Things) 장비를 설계할 수 있는 장점이 있으며 모듈의 중요 입출력 핀 구성은 [Fig. 2]과 같다[8].



[Fig. 2] Pin diagram of Esp8266 and ESP-12F

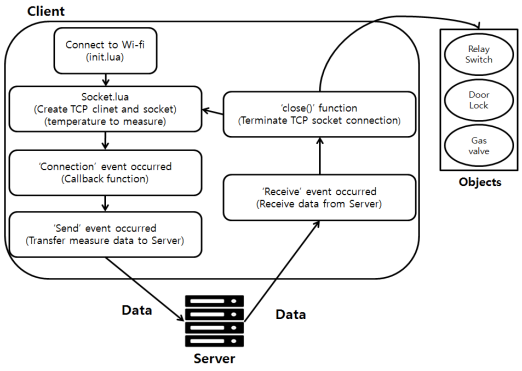
〈Table 1〉 Physical Specifications of Esp8266

Feature	
Microcontroller	ESP-8266EX
Operating Voltage	3.3V
Digital I/O Pins	11
Analog Input Pins	1(Max input: 3.2V)
Clock Speed	80MHz/160MHz
Flash	4M bytes
Length	34.2mm
Width	25.6mm
Weight	10g
• 802.11 b/g/n	
• Integrated low power 32-bit MCU	
• Integrated 10-bit ADC	
• Support STA/AP/STA+AP operation modes	
• Integrated TCP/IP protocol stack	
• Support Smart Link Function for both Android and iOS devices	
• WiFi 2.4 GHz, support WPA/WPA2	
• Operating temperature range -40C ~ 125C	

Esp8266은 내부는 Tensilica의 L106 Diamond 시리즈 32-bit 프로세스와 SRAM을 탑재하여 보다 저전력에서 고성능 동작이 가능하다. 센서와 같은 외부적인 데이터를 GPIO(General Purpose Input/Output)를 통해 제공 받고, 아날로그 데이터도 1 채널 받을 수 있다. 또한 내부 Flash memory에 특정 펌웨어를 업로드 하여 보다 쉬운 MCU(Micro Controller Unit) 컨트롤이 가능하다[9].

2.3 클라이언트 동작 절차

클라이언트의 동작 절차는 [Fig.3]과 같다. 제일 먼저 Wifi연결을 시도하여 서버와 연결 설정에 들어간 후 Socket.lua를 통해 TCP 클라이언트를 생성한 후에 센서(온도, 습도 등)를 측정한다. 그리고 센서 데이터는 TCP segment data로 전송되어지는데 TCP 헤더의 Push flag에 의해 바로 응용프로그램으로 전달된다. TCP는 연결형 서비스를 제공하기에 “연결설정 -> 데이터 전송 -> 연결 해제”를 계속 반복하고, 이러한 반복적인 데이터 전송은 소켓을 통해 실시간으로 수행된다.



[Fig. 3] Esp8266 Operation Flowchart

3. 소켓 프로그래밍

3.1 서버 프로그래밍

소형 데이터 통합장치는 TCP/IP의 소켓 통신을 이용하여 양방향 통신을 할 수 있도록 구성되어 있으며, 일반적으로 소켓(Socket)은 컴퓨터가 네트워크에서 데이터를 전송하기 위해 사용하는 도구라고 할 수 있다. 즉, 물리적인 연결이 아닌 논리적인 연결로서 네트워크상에서 서버와 클라이언트의 데이터 교환을 위해서 TCP/IP 통신 방식을 더 쉽게 구현하는 방법 중의 하나이다. 소켓 통신은 제공자인 Server와 요청자인 Client로 나뉘는데 Client가 Server에 접속하는 형태로 3-Way Hand Shake를 통해서 연결되어 진다.

Node.js에서는 socket이라는 객체를 사용하여서 이벤트를 동작 방식으로 연결, 데이터 전송, 연결해제, 에러제어 등을 수행할 수 있다. 또한 서버는 listen()함수를 통하여 특정 포트번호로 접속이 가능하도록 서버를 대기하도록 한다. 서버에 접근한 클라이언트는 지정된 IP번호와 연결을 시도하게 된다.

[Fig. 4]는 Nodejs의 소켓 구성 코드이다. 처음에 server라는 변수에 net모듈을 사용하여 서버를 생성하고 내부 Callback 함수에 socket 객체를 매개변수로 전달한다. 그 후 socket를 가지고 각각의 이벤트를 발생시키는데, socket.on()은 이벤트를 연결하는 메서드이다. 이러한 메서드는 내부에 ‘data’, 'end', 'error','timeout' 등의 이벤트를 지정하여 동작하게 된다[10,11].

```
var server=net.createServer(function(socket){
  console.log('socket connection(연결)');
  console.log('local(서버측 IP,PORT) = %s:%s',
  socket.localAddress, socket.localPort);
  console.log('remote(socket측 IP,PORT) = %s:%s',
  socket.remoteAddress, socket.remotePort);
socket.on('data', function(data) {

        //생략

else if(socket.remoteAddress == "::ffff:192.168.0.11"){
console.log("받은 데이터: "+data.toString());
    socket.write("hello");
    socket.on('end', function() {
    console.log('socket disconnected(연결 종료)');
    });
    socket.on('error', function(err) {
      console.log('Socket Error: ', JSON.stringify(err));
    });
    socket.on('timeout', function() {
      console.log('Socket Timed out');
    });
  });
});

server.listen(80,function(){
console.log("서버가 80포트에 대기합니다.");
});
```

[Fig. 4] A socket implementation of Node.js

3.2 클라이언트 프로그래밍

소켓 통신을 위해서는 클라이언트도 소켓을 생성하여야 한다. Esp8266은 lua script을 기반으로 코드가 진행되어지지만 Node.js 스타일을 따라가기 때문에 서버 측에 서소켓을 구성한 것과 비슷한 구조를 지닌다. 기존에 업로드 한 NodeMCU 펌웨어에는 ‘net‘ Module이 정의되어 있으며 Node.js에서 사용하던 코드와 구성이 유사하다.

[Fig. 5]은 클라이언트의 소켓 생성과 데이터 전송에

관련된 코드를 보여주고 있으며, Conn=net.createConnection (net.TCP)를 통해 TCP 클라이언트 객체가 생성된다. 연결할 대상(서버)의 IP와 Port번호를 기입하고 난 후 특정 이벤트가 발생하면 준비되어진 동작을 실행 한다. 이벤트는 대표적으로 "connection", "send", "receive", "close" 등이 있으며, 각각 연결 설정, 데이터 전송, 데이터 수신, 연결 해제의 의미를 가지고 있다[12,13].

```
tmr.alarm(0, 5000, 1, function()
pin=5
rev=4
disc=6
gpio.mode(rev,gpio.OUTPUT)
gpio.mode(disc,gpio.OUTPUT)
conn = net.createConnection(net.TCP)          // TCP 클라이언트 생성
conn:connect(80, "192.168.0.2")               // 연결할 서버의 IP와 PORT번호
conn:on("connection",function(sck,c)          // 연결 이벤트 연결
status, temp, humi, temp_dec, humi_dec = dht.read(pin)
    if status == dht.OK then
        temp_1 = math.floor(temp)
        humi_1 = math.floor(humi)
    elseif status == dht.ERROR_CHECKSUM then
        print( "DHT Checksum error.")
    elseif status == dht.ERROR_TIMEOUT then
        print( "DHT timed out." )
    end
    print("temp:"..temp)
    print("humi:"..humi)
    conn:send("temp:"..temp.." humi:"..humi)      //데이터 전송 이벤트
    end)
    conn:on("receive",function(conn,payload)      //데이터 수신 이벤트
    print(payload)
    if(payload=="hello") then
    gpio.write(rev,gpio.HIGH)
    tmr.delay(50000)
    gpio.write(rev,gpio.LOW)
    end
    conn:close()                                  // 연결 종료
    end)
end)
```

[Fig. 5] A socket creation and data transmission/reception code on the client side.

4. 데이터 통합 장치 구현

4.1 WireShark을 이용한 데이터 흐름 분석

소켓 구성을 통해 구성되어지는 TCP연결 흐름과 헤더를 확인하기 위해 WireShark를 사용한다. 먼저 Client(192.168.0.11)는 Server(192.168.0.2)에게 SYN(연결 요구) 플래그와 함께 TCP헤더를 전송한다. 이에 Server는 Client의 연결요구에 대한 응답의 의미로 ACK 플래그를 지정하여 응답한다. 그 다음 Client는 TCP segements data를 전송하는데 PSH(상위 계층으로 바로 적용) 플래그를 함께 지정하여 전송한다. 또한 Server도 Client에게 TCP segements data를 전송하는데 동일하게 PSH 플래그 비트를 지정해서 데이터를 전송한다. 데이터가 양방향으로 모두 전송되어진 후에는 Client가 연결 해제를 요구하는 의미로 FIN 플래그 비트를 지정하고, 수신한 데이터에 대한 ACK 응답으로 ACK플래그를 지정하여 응답한다. 이를 받은 Server는 ACK 응답을 한다. TCP에서는 연결해제 또한 점진적인 방식으로 진행 되므로 Server가 FIN 플래그를 지정하여 Client에게 전송하므로 연결이 해제된다.

[Fig. 6]은 WireShark을 이용하여 클라이언트로부터 받은 데이터를 서버에서 보여주는 화면이다. 화면에서는 [Fig. 1]의 통합장치에서 클라이언트에서 측정한 온도, 습도 정보와 각각의 IP주소 및 Port번호를 보여준다.

```
###(시작)#########################################################################
#########
socket connection(연결)
   local(서버측 IP,PORT) = ::ffff:192.168.0.2:80
   remote(socket측 IP,PORT) = ::ffff:192.168.0.11:33388
받은 데이터: temp:21 humi:43
temp:21
humi:43

Received data from socket on port(받은 데이터 내용) ::ffff:192.168.0.11:33388 => temp:21 humi:43

  Bytes received(받은 바이트 수): 15
  Bytes sent(보낸 바이트 수): 5

socket disconnected(연결 종료)
Remaining Connections(연결 중인 개체 수): 0
###(종료)#########################################################################
```

[Fig. 6] The data on the server received from the client.

4.2 데이터베이스와 데이터 융합

서버로 전송되어지는 데이터는 데이터 베이스에 축적되어진다. 데이터베이스는 MySql을 사용하였으며, Node.js에서 Query 명령을 통해 온도, 습도, 측정시간을 담았으며, 위상과 진폭 등 더 많은 데이터를 담을 수 있다. Node.js에서는 DB와 관련된 모듈 'mysql'을 통해서 Mysql과의 연결을 설정 할 수 있다[14,15].

[Fig. 7]은 MySQL을 사용할 때 사용된 변수 값들을 보여주고 있다. 변수 데이터베이스에 mysql에 대한 유저 이름, 비밀번호, 데이터베이스 이름 등을 설정하여 저장하고 데이터를 수신 받으면 query명령을 통해 특정 데이터 값을 DB에 저장한다. 저장된 데이터는 단독으로도 사용될 수 있지만 온도, 습도, 강수량, 풍속, 진동, 충격 등의 자료를 유사환경에서 측정한 표준데이터 또는 예년 데이터와 비교하여 각 항목에 대한 정상값과 경고값 기준에 따라서 메트릭스 형태로 데이터를 융합하면 화재, 장마, 지진, 홍수, 태풍 등 환경변화에 대한 중요한 정보를 도출할 수 있으며 이것을 각종 재해대책에 활용할 수 있다.

```
mysql> select *from data;
+----+------+------+---------------------+
| id | temp | humi | created             |
+----+------+------+---------------------+
|  1 |   21 |   45 | 2016-11-11 04:05:11 |
|  2 |   23 |   47 | 2016-11-11 04:05:16 |
|  3 |   21 |   45 | 2016-11-11 04:05:21 |
|  4 |   21 |   45 | 2016-11-11 04:05:26 |
|  5 |   23 |   47 | 2016-11-11 04:05:31 |
|  6 |   21 |   45 | 2016-11-11 04:05:36 |
|  7 |   21 |   45 | 2016-11-11 04:05:41 |
|  8 |   21 |   45 | 2016-11-11 04:05:46 |
|  9 |   21 |   45 | 2016-11-11 04:05:51 |
| 10 |   21 |   45 | 2016-11-11 04:05:56 |
+----+------+------+---------------------+
10 rows in set (0.00 sec)
```

[Fig. 7] Table values in MySQL

5. 결 론

본 연구에서는 하나의 공유기로 연결되어지는 하나의 서버에 다수의 클라이언트들의 양방향 데이터 전송이 사용자의 개입이 없이도 사물간의 제어가 가능하다는 것이 소형의 Esp8266모듈을 이용하여서 설계가 가능 하다는 것을 확인 하였다.

이러한 데이터 통합장치는 APP이나 Web을 통해서 사용자에게 실시간으로 전기사용량, 수도사용량 등을 전달해 줄 수 있으며, 집안 내부의 각종 데이터(온도, 습도, 조도 등)를 서버에 전달함으로써 하나의 플랫폼으로 여러 데이터를 통합하는데 유용하게 사용될 수 있어서 소형 데이터 통합장치로도 활용할 수 있을 것으로 판단된다.

또한 저장된 환경 데이터를 유사환경에서 측정한 표준데이터 또는 과거 데이터와 비교하여 각 항목에 대한 정상값과 경고값을 기준으로 메트릭스 형태로 데이터를 융합하면 화재, 장마 등 환경변화에 대한 중요한 정보도 도출할 수 있도록 설계되어 있다. 그러나 TCP/IP의 보안적인 요소와 데이터를 좀 더 가치 있는 정보로 융합하는 방법을 더욱 연구해야 할 것으로 사료된다.

ACKNOWLEDGMENTS

This work was supported by 2016 Baekseok University research fund.

REFERENCES

[1] Jeong Lae Lee "Real time remote management for home network system using bio-physical sensor" Journal of the Korea Computer Information Society, Vol. 16, No. 1, pp. 117-124, 2011.

[2] Joong-Soo Lim, "Design of High Speed Data Acquisition and Fusion System with STM32 Processor", Journal of Korea convergence Society, Vol. 7, No. 1, pp.9-15, 2016

[3] Hong Gyu Kim, Seung Jin Moon, Jong Dae Lee "Cattle Shed Management System Based on Wireless Sensor Network with Bio and Environmental Sensors" The Institute of Electronics Engineers of Korea Information and Communications Society (ETRI) Vol. 38 No. 7(fusion technology), pp. 573-586, 2013.

[4] Kwang Sin Shin, Joon Dal Kwon, Young Dong Lee, Wan Young Chung, "Ad-hoc home network system using wireless sensor network technology", Journal of the Korean Sensors Society, Vol. 16, No. 2, pp.142-149, 2007.

[5] Joong-Soo Lim, "Design of Fusion Multilabeling System Controlled by Wi-Fi Signals", Journal of Korea convergence Society, Vol .6, No. 1, pp.1-5, 2015.

[6] Dong Jo Kang, Hyun Ju Park, "A design and implementation of transmit/receive model to speed up the transmission of large string-data sets in TCP/IP socket communication" The Journal of the Korea Information Science and Technology Association, Vol. 17, No. 4, pp. 885-892, 2013.

[7] Sung Jun Lee, Geun Taek Kang, Won Chang Lee, "Intelligent Refrigerated Container Monitoring System using TCP/IP", 2011 The 26th ICROS Conference, pp. 494-496, 2011.

[8] "Esp8266 application guide" http://www.hardcopyworld.com/gnuboard5/bbs/board.php?bo_table=lecture_esp&wr_id=1&page=1

[9] "Node.js v4.6.2 Documentation"https://nodejs.org/dist/latest-v4.x/docs/api/net.html#net_new_net_socket_options

[10] "Node MCU Documentation"https://nodemcu.readthedocs.io/en/master/en/

[11] "Node.js official page" https://nodejs.org/ko/

[12] Young Tae Jo, Jin Sup Choi, In Bum Jung "Traffic

Information Acquisition System with Small Ultrasonic Sensors based on Wireless Sensor Networks", Journal of KISS: Computing Practices and Letters, Vol. 20, No. 7, pp. 408-424, 2014.

[13] Chang Hyeon Lee, Ho-Guen Song, Hee-Dong Park, Do-Hyeon Kim "Implementation of Sensor Node Control Based on Sensed Data and Power Monitoring", Korea Multimedia Society Fall Conference Proceedings, Vol. 13, No. 2, 2010

[14] Yong-Wook Nam, Yong-Hyuk Kim, "Speed estimation of sound-emitted objects through convergence of sound information analysis and smart device technology", Journal of Korea convergence Society, Vol. 6, No. 5, pp. 233-240, 2015.

[15] Kwon Yoon, Hyung-Wook Bang, Data Base Language SQL, N1, 2016. 3

저자소개

이 동 석(Dong-Seok Lee) **[학생회원]**

- 2012년 3월 : 백석대학교 정보통신학부 입학
- 2017년 1월 : 백석대학교 정보통신학부 재학중

<관심분야> : IT융합, 정보통신, 전자기기, IoT 시스템

임 중 수(Joong-Soo Lim) **[종신회원]**

- 1978년 2월 : 경북대학교 공과대학 전자공학과 (학사)
- 1994년 3월 : Auburn대학교 전자공학과 (공학박사)
- 1994년 3월 ~ 2003년 2월 : 국방과학연구소 책임연구원, 전자전연구실장
- 2003년 3월 ~ 현재 : 백석대학교 정보통신학부 교수
- 2016년 7월 ~ 현재 : 백석대학교 전산정보원장

<관심분야> : IT융합, 정보통신, 전자기기, 전자전 시스템, 레이더