教育用無線センサネットワークの基盤技術開発*

鹿間 敏弘*1

Development of Wireless Sensor Network Platform for Educational Use

Toshihiro SHIKAMA

*1 Department of Electrical, Electronic and Computer Engineering

This paper describes the architecture and implementation of a wireless sensor network for educational use. The purpose of this development is intended to support students to develop original IoT applications using the wireless sensor network. The system consists of Raspberry Pi as a sensor server and XBee modules, which employ ZigBee as a wireless interface. All the software developed in this study is written in Python and is open to students. Students are allowed to develop their own IoT applications based on their own ideas. This paper first introduces the outline of ZigBee and XBee, then it describes the detail of XBee control by Python and its interface with CGI program. It also reports the experiments of multi-hop communication by XBee modules and the exclusive control, which is needed to keep consistency of XBee control when multiple users try to access the sensor server simultaneously. Basic functions of the sensor network have been confirmed through the experiments.

Key Words : Sensor Networks, IoT, XBee, Python, Raspberry Pi

1. 緒　　言

近年，IoT (Internet of Things) [(1)]が注目を集めている．従来のインターネットは全世界のコンピュータをネットワークにより結ぶことにより仮想的な世界（サイバー空間）構築を可能とするものであるが，IoT ではコンピュータだけではなく様々な「もの」に埋め込まれた極めて多数のコンピュータによるネットワークの拡大を目指している．ここでは，温度や湿度や振動など現実の世界に密着した情報がインターネットを介して配信され活用される．これにより，様々なアクチュエータやロボットなどを介して現実の世界に働きかけることができる．

このような IoT を実現する上では，様々な「もの」にコンピュータを埋め込み，ネットワークに接続することが必要となる．ここで，従来のコンピュータと異なり，電池で長時間自律動作することが重要となる．すなわち，「もの」が電源コードの束縛を離れて自由に設置され，また移動できることが重要で，電池や環境エネルギーの利用が可能な低消費電力のネットワーク技術が必要となる．このようなネットワーク技術は，これまで注目されてきたユビキタスネットワーク [(2)] や M2M (Machine to Machine) [(3)] のキーワードの延長上にあると言える．

ユビキタスネットワークに対応するネットワーク技術として，これまでセンサネットワークの技術開発が行われてきており[(4)(5)]，標準化されたものとして ZigBee[(6)(7)]が有名である．最近，広く使われている Bluetooth においても，バージョン 4 では通信の高速化よりも低消費電力化を目指して開発されている．IoT のキーワードとともに従来から開発されてきたセンサネットワーク技術に再び注目が集まっていると言える．

本研究のねらいは，上記のような IoT に注目が集まる状況を踏まえ，センサネットワークや IoT アプリケーション開発のための教育用のプラットフォームを開発することにある．IoT では「もの」と「もの」または「もの」とコンピュータをつなぐネットワークにより，新鮮な発想で新しいアプリケーションが生まれる可能性がある．

最近，学生が開発目標やスケジュールを含むプロジェクトを設定し，自律的に開発を進める PBL 教育が注目さ

* 原稿受付　2015 年 2 月 26 日
*1 電気電子情報工学科
E-mail: shikama@fukui-ut.ac.jp

れている．本研究では，学生が自由な発想で新しいIoTアプリケーションを開発するための基盤となる無線センサネットワークを構築することを目指し，PBL教育の材料提供を目的としている．

具体的にはZigBeeによる無線センサネットワークを実現し，名刺サイズの小型コンピュータであるラズベリー・パイ[8][9]によりセンサネットワークとインターネットを接続するセンササーバを実現する．ユーザはこのセンササーバを介して，インターネットから「もの」へのWEBによるアクセスが可能となる．本研究で開発するすべてのソフトウェアはPython[11]により記述し，オープンとする．これにより学生がアプリケーション開発において自由に工夫を加えることを可能とすることを狙う．

2. 基本構成と技術課題

2.1 基本構成

本研究では，ユーザがPCからWEBを介してセンサネットワークの先にあるデバイスにアクセスして，出力する値の設定やセンサ値の読み取りを実現する．具体的な構成として，Fig. 1 のようなシステムの基本構成を想定している．

ユーザはPCのブラウザから学内ネットワークを介してセンササーバにアクセスする．センササーバにはUSBインタフェースにより XBee モジュールが接続されており，このサーバ側 XBee モジュールと離れた場所に置かれた遠隔の XBee モジュールとが ZigBee 規格に従った無線通信を行う．サーバ側 XBee モジュールは USB により給電されるが，遠隔の XBee モジュールは電池で動作し，温度センサや LED などが接続されている．センササーバは名刺サイズの小型 Linux コンピュータであるラズベリー・パイにより構成され，WEB サーバ機能は Linux サーバにおいて標準的に用いられる Apach2 をインストールして実現する．

リモート XBee モジュールに LED が接続されている場合，ユーザが PC を操作して LED を点灯する処理の流れは次のようになる．ユーザは PC のブラウザを操作して，センササーバの CGI (Common Gateway Interface) を介して XBee 制御プログラムを呼び出す．CGI は WEB サーバがネットワークから要求を受信して，サーバ内部の処理プログラムを起動し結果を受け取るための標準的なインタフェースである．CGI を介して起動された XBee 制御プログラムは USB インタフェースを介してサーバ側の XBee モジュールに AT コマンドを送る．AT コマンドは XBee で定義されたコマンドで，これによりリモート XBee モジュールの特定のポートを ON にすることができる．これにより，このポートに接続された LED が点灯する．

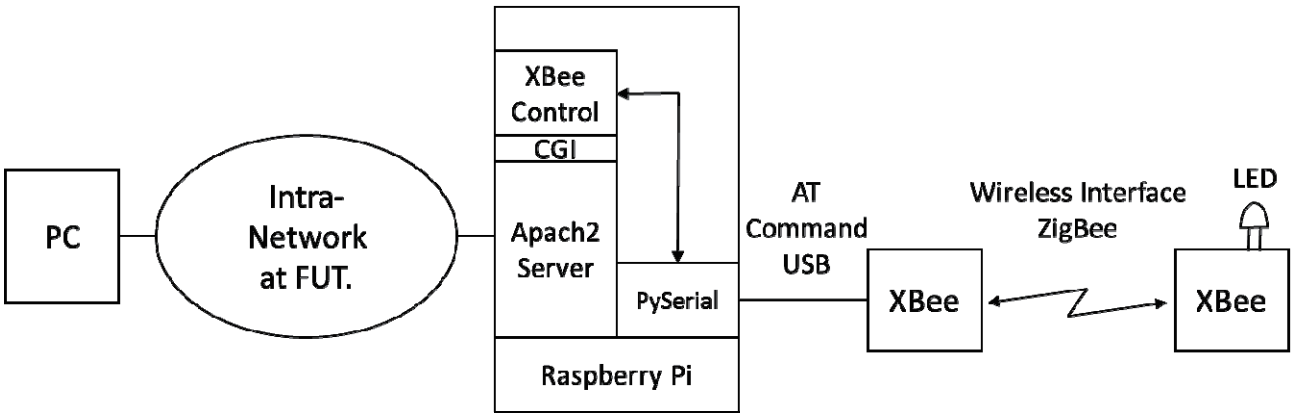

Fig. 1 Basic System Configuration

2.2 プログラミング言語

本研究は学生が自由にセンサネットワークを構築する上で必要となる基盤技術開発を目的としている．センサネットワークのアプリケーションを開発するためにはプログラミングを行わねばならない．基本となるプログラミング言語の選択が重要となるが，本研究では文法がシンプルでコンパイルすることなくプログラムを実行できる Python を採用した．Python は本研究に先行して行ったロボットを使用した教材開発[10]において実績がある．ま

た, XBee モジュールを制御するためには USB インタフェースを介してシリアル通信を行うことが必要であるが, Python では比較的容易にこれを実現できる.

2.3 技術課題

教育用のセンサネットワークシステムを開発する上では以下の課題がある. 以下, 最初に 3 章で ZigBee と XBee の概要を紹介し，以降でこれらの技術課題について述べる.

- Python プログラムによる XBee モジュールの制御
- CGI と XBee 制御のインタフェース
- マルチホップ通信
- センササーバの排他制御

3. 無線センサネットワーク用の標準規格 ZigBee と無線モジュール XBee

3.1 ZigBee とは

本研究ではセンサからの情報受信やセンサへの指示を ZigBee と呼ばれる標準規格の無線通信により行う. ZigBee のレイヤモデルを Fig. 2 に示す. ZigBee の物理層 (PHY) とメディアアクセス制御層(MAC)は IEEE802.15.4 として規定されている．この上位にネットワーク層とアプリケーションサポート層が存在し，これらは ZigBee アライアンスにより規定されている．最上位のアプリケーション層(APL)はユーザにより定義される.

ZigBee は比較的短距離の無線通信で省電力が大きな特徴である．携帯などで普及している Bluetooth に近い無線通信規格でどちらも 2.4GHz 帯の ISM バンドを使用する．どちらも比較的近距離をカバーするネットワークであるため PAN (Personal Area Network)と呼ばれる．両者の比較を Table 1 に示す．ZigBee の方が低速であるが，ネットワーク内のノード数が格段に大きく，省電力であることが分かる.

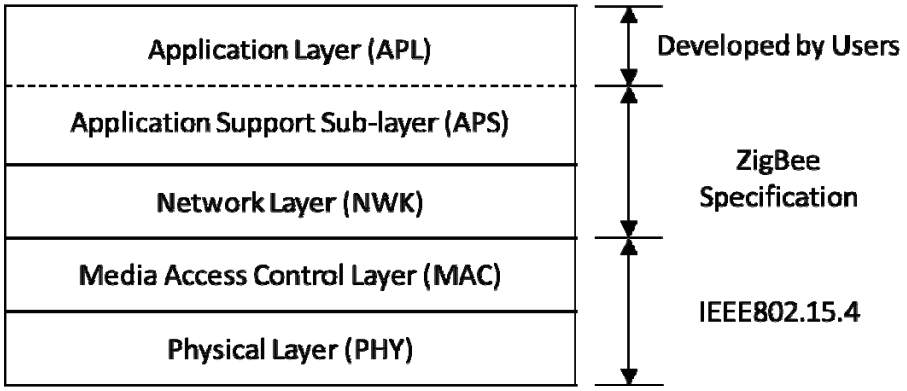

Fig. 2 Layer model of ZigBee

Table 1 Comparison of ZigBee and Bluetooth

名称	ZigBee	Bluetooth
IEEE 規格	802.15.4	802.15.1
周波数	2.4GHz	2.4GHz
変調方式	Offset QPSK	GFSK
拡散方式	DSSS	FHSS
通信距離	30m	10m
通信速度	250kpbs	1Mbps
ネットワーク内最大ノード数	65,536	7
電池寿命	数年	数日
アプリケーション	計測制御	ハンズフリー通信（音声）など

ZigBee によるネットワークでは論理的なノードとして，ZigBee コーディネータ，ZigBee ルータ，ZigBee エンドデバイスが存在し，これらのノードは IEEE の拡張 MAC アドレス(64bit)と，ネットワークにノードが参入する際にコーディネータが割り当てる 16 ビットのショートアドレスにより識別される．ZigBee によるネットワークは Fig. 3 に示すようにスター，メッシュ，クラスタツリーのトポロジーが可能である．どのトポロジーにおいてもコーディネータが 1 ノード存在し，親ノードとして管理やコンピュータ等とのインタフェースを行う．ルータは他のノードからのデータを中継する機能を有し，これによりネットワークのカバー範囲を拡大することができる．エンドデバイスには中継機能は無いが省電力動作が可能である．

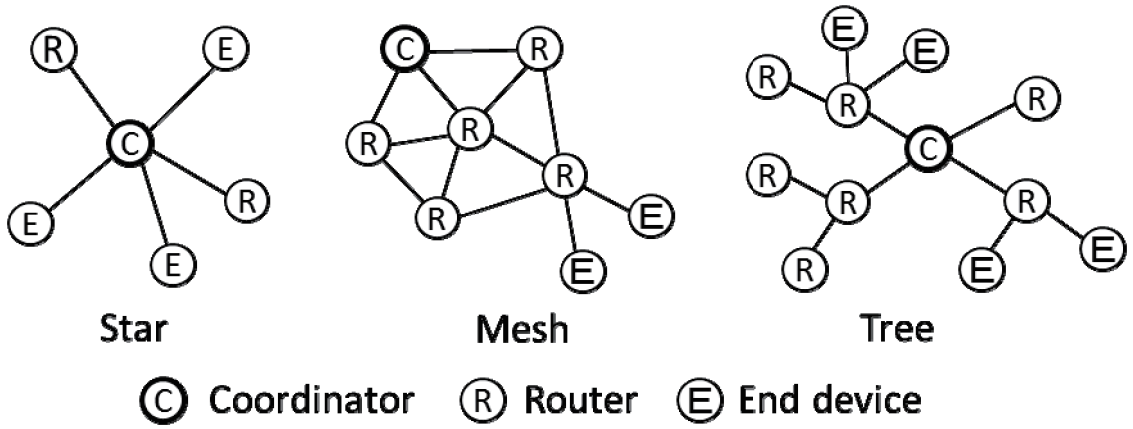

Fig. 3 Possible Network Topologies by ZigBee

3.2 XBee とは

XBee[12][13]は Digi International 社が販売しているマイコンを内蔵した小型の無線モジュールで，ZigBee の他に WiFi に対応したモジュールも販売されている．Fig. 4 に本研究で使用する XBee モジュールの外観を示す．Fig. 4 で上は表面, 下は裏面を示している. 各 XBee モジュールには予め 64 ビットの拡張 MAC アドレスが割当てられ，写真に示すように裏面に印刷されている．Fig.4 で下のモジュールの場合，拡張 MAC アドレスは 16 桁の 16 進数で“0013A200409BB65B”となる．XBee モジュールは 20 本のピンがあり，ピンの割当てを Table 2 に示す．これらのピンに電源やグランドそしてセンサなどを接続する. モジュールには ZigBee 規格の無線送受信機能の他にマイコンが内蔵され，このマイコンにより Fig. 2 に示す上位レイヤ機能が実現されている．

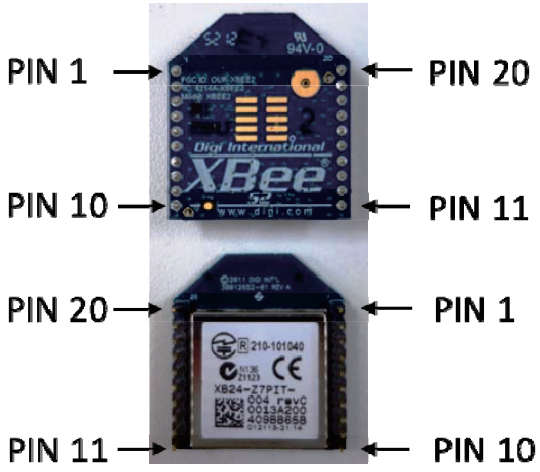

Fig. 4 Picture of XBee Module

Table 2 Pin Assignment

Pin #	Name	Direction	Default State	Description
1	VCC	–	–	Power supply
2	DOUT	Output	Output	UART Data Out
3	DIN / CONFIG	Input	Input	UART Data In
4	DIO12	Both	Disabled	DI/O 12
5	RESET	Both	Open-Collector	Module Reset
6	RSSI PWM / DIO10	Both	Output	RSSI / DI/O10
7	DIO11	Both	Input	DI/O 11
8	[reserved]	–	Disabled	Do not connect
9	DTR / SLEEP RQ/ DIO8	Both	Input	Sleep Control Line or DI/O 8
10	GND	–	–	Ground
11	DIO4	Both	Disabled	DI/O 4
12	CTS / DIO7	Both	Output	CTS or DI/O 7
13	ON / SLEEP	Output	Output	Module Status Indicator or DI/O 9
14	VREF	Input	–	Voltage Reference
15	Associate / DIO5	Both	Output	Associated Indicator, DI/O 5
16	RTS / DIO6	Both	Input	RTS, DI/O 6
17	AD3 / DIO3	Both	Disabled	Analog Input 3 or DI/O 3
18	AD2 / DIO2	Both	Disabled	Analog Input 2 or DI/O 2
19	AD1 / DIO1	Both	Disabled	Analog Input 1 or DI/O 1
20	AD0 / DIO0 / Commissioning Button	Both	Disabled	Analog Input 0, DI/O 0, or Commissioning Button

RSSI: Receive Signal Strength Indicator

CTS: Clear To Send

RTS: Request To Send

XBee モジュール間では ZigBee 規格に従った無線通信を行うが，これは基本的には XBee を使うユーザ側には隠蔽されており，ユーザが意識する必要はない．XBee モジュールとコンピュータと接続には USB インタフェー

スが用いられ，シリアル通信が行われる．キャラクタベースのシリアル通信で独自の AT コマンドが定義されており，コンピュータは AT コマンドにより遠隔の XBee モジュールに接続されたセンサの情報を読み取ることができる．

XBee モジュールは多くの機能を有し，予め動作や機能を定義して使用する．大きく分けて以下の動作モードが存在し，これは ZigBee の規格で説明した論理的ノードと整合するものであるが，規定の範囲を超える部分は独自の機能が定義されている．

(1) コーディネータ

ネットワークの親ノードとして動作しコンピュータと接続する．コンピュータからの AT コマンドによる指示を受信してルータやエンドデバイスと通信し，結果をコンピュータに返す．

(2) ルータ

センサなどのデバイスを接続してその値を読み取ってコーディネータに送信したり，デバイスに値を設定したりする．また，他のルータまたはエンドデバイスへの無線パケットを受信してこれを中継する．

(3) エンドデバイス

センサなどのデバイスを接続してその値を読み取ってコーディネータに送信したり，デバイスに値を設定したりする．ルータとは異なり，他のルータやエンドデバイスへの無線パケットの中継は行わないが，スリープにより省電力で動作できる．

XBee のモジュールは使用する前に動作モードの詳細設定や内臓マイコンのファームウェア更新を行う必要がある．このために Digi International 社から X-CTU と呼ぶソフトウェアが提供され，Fig. 5 に示すように Windows パソコンに XBee モジュールを接続して X-CTU により GUI ベースで詳細な設定を行うことができる．ここで，XBee モジュールをパソコンに接続するためには，XBee-USB インタフェース基板が必要で，この基板を介すことによりパソコンの X-CTU はシリアル通信により XBee モジュールの設定を行う．一旦，設定を行えば XBee モジュールはパソコンから切り離して使用できる．Fig. 6 は X-CTU の画面例を示している．同一の PAN に属するノードは全て同じ PAN ID を設定する必要があり，Fig. 6 では“1”を設定している．また，この画面では対象となる XBee モジュールはルータに設定されている．

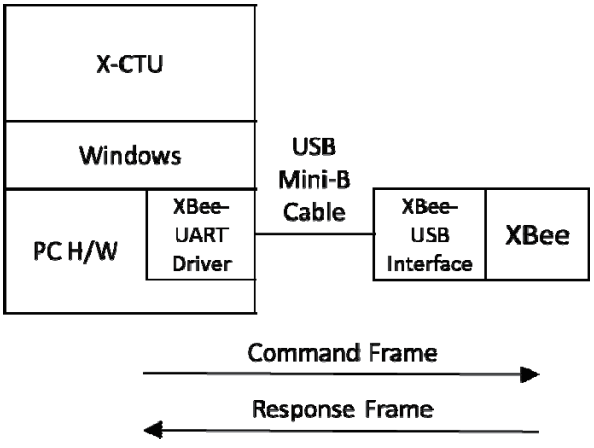

Fig. 5 Configuration of XBee by X-CTU

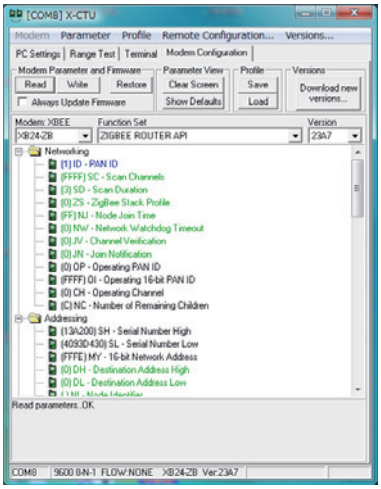

Fig. 6 Example of X-CTU Window

4. Python による XBee の制御

3.3 XBee とコンピュータとのインタフェース

XBee モジュールとコンピュータとは Fig. 5 に示すように XBee-USB インタフェース基板を介して USB ケーブル（USB Mini-B ケーブル）により接続される．インタフェース基板には USB-シリアル変換を行う IC が搭載されており，パソコンにはこの IC のドライバをインストールする必要がある．最終的にパソコンは XBee モジュール

を論理的なシリアルインタフェースとして扱い，シリアルインタフェースの上にXBee独自のATコマンドが定義されている．このATコマンドはローカル通信，リモート通信で形式が異なるが，無線通信で遠隔のXBeeモジュールにアクセスするためにはリモート通信の形式を用いる必要がある．リモート通信ではフレーム形式で通信を行う．

Fig. 7は受信強度を調べるためのATコマンド‘DB’を送るコマンドフレームとこのコマンドに対する応答フレームを示している．フレームは左側のバイトから順次送受信され，‘7E’の開始文字から始まり，2および3バイト目(B)は以降のデータのバイト数を示している．フレームの最後のバイト(I)は誤り検出用のチェックサムである．6バイト目から8バイト（64ビット）のリモートアドレスが設定され，その後に16ビットのショートアドレスが設定させる．送信時にショートアドレスが分からない場合は‘FFFE’を設定する．H:の‘4442’がDBのASCIIコードを16進数表示したものである．応答フレームにおいて(H)はDBコマンドで，(J)の‘00’は正常にコマンドが受け付けられたことを示す．(K)の‘20’が受信強度でこれは10進数で-32dbmの値を示している．応答フレームではF:のショートアドレスとして‘F78A’が設定されており，これがコーディネータにより割り当てられた実際の16ビットアドレスの値である．

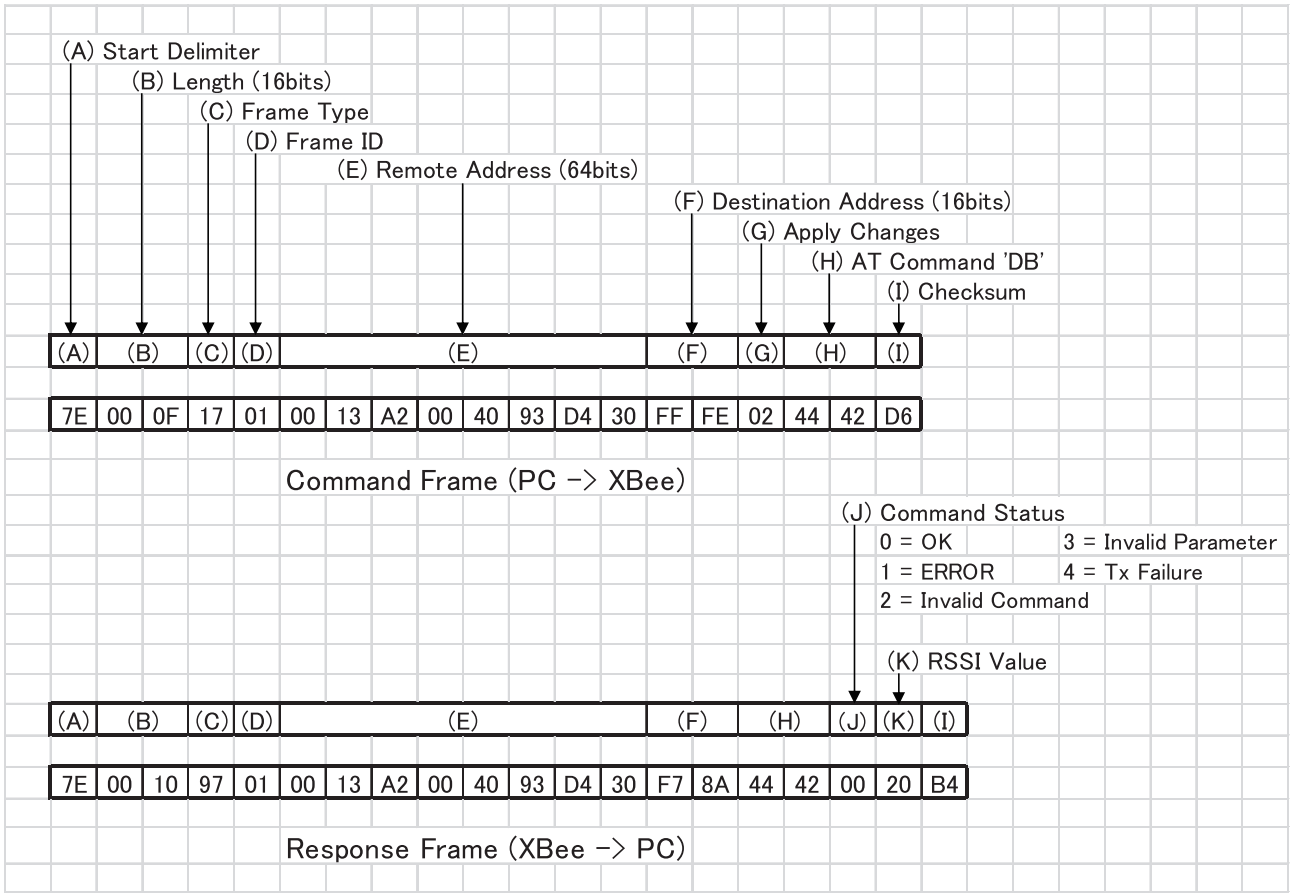

Fig. 7 ‘DB’ Command and Response Frame Structure

3.4 XBeeとコンピュータとのインタフェース

XBeeとのコマンドの送信およびレスポンスの受信はシリアル通信で行う．ラズベリー・パイでユーザはPythonによりプログラミングを行うが，シリアル通信で送受信を行う必要がある．このためにPyserialのモジュール[14]を使用する．

Fig. 8はPyserialによるフレームの送信処理で，左側のsend()関数は64ビットアドレスの下位32ビットとコマンドおよびパラメータを与えて送信するフレームのリストを作成し，右側のsend_frame()関数がこのリスト型オブジェクトを受け取って，チェックサムを計算し，リスト型オブジェクトから1バイトづつ取り出して送信を行

う．これらの関数を呼び出すプログラムでは以下のように，シリアル通信用のオブジェクト ser を生成する必要がある．ここではシリアル通信の速度を 9600bit/s，最大受信待ち時間を決めるタイマーを 2 秒に設定している．

```
ser = serial.Serial('/dev/ttyUSB0', 9600, timeout = 2)
```

Fig. 9 は Python と Pyserial によるフレームの受信処理を行う関数 rcv_frame() のプログラムである．先頭から 2 バイト目と 3 バイト目の値で，後続データのバイト数を把握し，このバイト数分のデータを受信する．ここで，ser.read()が 1 文字受信を行う関数であるが，最大受信待ち時間を 2 秒に設定しているため，2 秒間受信しない場合は文字受信無しで制御が戻り，制限なく受信待ちとならないようにしている．この rcv_frame()関数は，受信したフレームの中で先頭の 3 バイトを除く後続のバイト列をリスト型のオブジェクトとして呼び出し側に返す．

上記で説明した送信用および受信用関数を利用することにより，ユーザはラズベリー・パイ上で動作する Python のプログラムにより自由にセンサデータを送受信できる．ここで比較的面倒な送信および受信におけるチェックサムの計算はこれらの関数で行っており，ユーザが意識する必要はない．

```
import time
import serial

def send_frame(sd):    # Send a frame
    sum = 0
    for byte in sd:
        sum = sum + int(byte)
    chk = (255 - sum) % 256
    length = len(sd)
    frame = [0x7e, length / 256, length % 256] + sd + [chk]
    for byte in frame:
        char = chr(byte)
        ser.write(char)
    return
```

```
def send(addr1, addr2, cmd, par):
    Ftype = 0x17    # Remote AT command request
    Fid = 0x01
    Raddr1 = [0x00, 0x13, 0xa2, 0x00]
    Raddr2 = [addr1 / 256, addr1 % 256, addr2 / 256, addr2 % 256]
    DA = [0xFF, 0xFE, 0x02]
    sd = [Ftype, Fid] + Raddr1 + Raddr2 + DA
    ATcom = [ord(cmd[0]), ord(cmd[1])] + par
    sd = sd + ATcom
    send_frame(sd)
    rd = []
    rd = rcv_frame()
    return rd
```

Fig. 8 Python Program for Sending a Frame

```
def rcv_frame():    # Receive a frame
    for i in range(3):
        char = ser.read()
        if len(char) > 0:
            byte = ord(char)
            if i == 0:
                if byte != 0x7e:
                    break
            elif i == 1:
                length = 256 * byte
            else:
                length = length + byte
        else :
            print 'receive timeout'
            length = -1
            break
    s = 0
    rdata = []
    for i in range(length+1):
        char = ser.read()
        if len(char) > 0:
            byte = ord(char)
            rdata = rdata + [byte]
            s = (s + byte) % 256
        else :
            print 'receive timeout'
            break
    if s != 255:
        print 'receive check error'
        rdata = []
    return rdata
```

Fig. 9 Python Program for Receiving a Frame

5. マルチホップ通信

ZigBee のネットワーク層ではマルチホップ通信が規定されており，XBee においてもモジュールの構成設定により可能となる．マルチホップ通信では Fig. 10 に示すようにルータ機能が定義されたモジュールが他のルータま

たはエンドデバイスとして定義されたモジュールのパケットを蓄積交換する．これにより，30m 程度とされている ZigBee の無線通信範囲を超えてセンサネットワークの範囲を拡大できる．

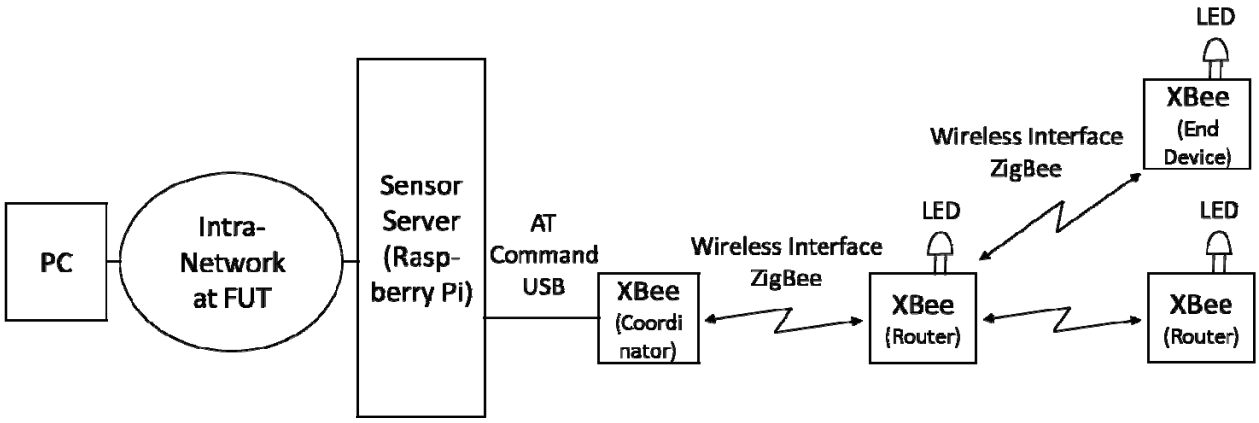

Fig. 10 Example of Multi-hop Communication

マルチホップ通信が行われていることを確認するために 3 個の XBee モジュールを Fig. 10 のフロアに配置して通信実験を行った．Fig. 11 は使用した建物のフロア図で，A から F は XBee モジュールを配置した場所を示している．コーディネータノードは常に A の位置に配置し，他の 2 個はルータとして機能し位置を A から F に変えてコーディネータとの通信を確認した．以下，これら 2 個の XBee モジュールを XBee1 および XBee2 と呼ぶ．これら 2 個の XBee モジュールは機能的には同じであるが，それぞれの位置を変化させることによりマルチホップ通信が行われるか否かを試験した．通信はコーディネータに接続されたセンササーバから 2 秒ごとにルータに接続された LED の点滅させるメッセージを送信し，点滅すれば通信が行われていると判断した．また，センササーバは LED の点滅とともに DB コマンドによりルータでの受信強度(RSSI)読取りも併せて行った．

Table 3 は XBee2 モジュールの位置は A に固定し，XBee1 モジュールの位置を A から F に順次変更した場合に LED 制御が可能な否かを調べたものである．XBee1 モジュールの位置が A から E までは LED の制御が可能であるが，F の位置では電波が受信限界を超え，制御できない結果となっている．Table 4 は XBee1 モジュールの位置を F に固定し，XBee2 モジュールの位置を A から F に順次変更した場合に LED 制御が可能な否かを調べたものである．XBee2 モジュールの位置が A の場合，XBee1 モジュールの LED を制御できないが，XBee2 モジュールの位置が B から E の場合, XBee2 モジュールとともに XBee1 モジュールの LED も制御可能である. これは, XBee2 モジュールが中継を行うことによりコーディネータから XBee1 モジュールへの通信が可能となっていると考えられる．これからマルチホップ通信が行われていると言える．

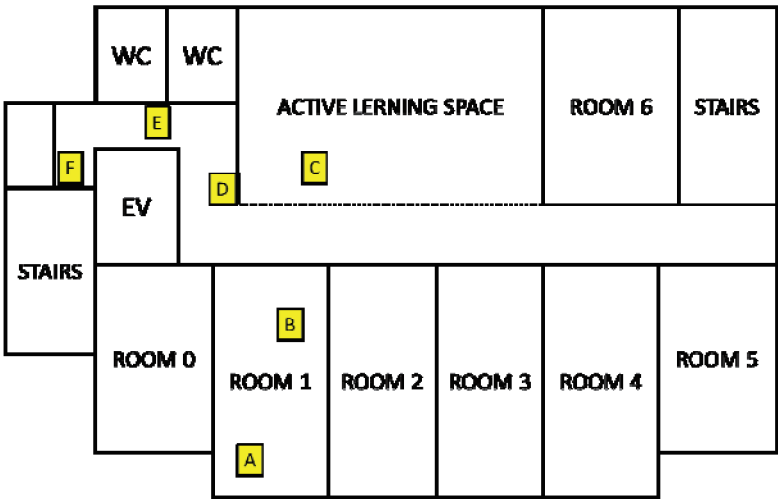

Fig. 11 Location of XBee modules

Table 3 ではコーディネータから XBee1 ノードへの距離が大きくなるに従い受信信号強度(RSSI)の値が小さくなっており予想される変化と言える．しかし，Table 4 では XBee2 モジュールの位置が C～E で受信信号強度(RSSI)の値が変化しておらず，説明できない現象が見られる．XBee モジュールにおける受信信号強度(RSSI)の定義や測定方法を確認する必要があり，この値をアプリケーションで使用することは避けるべきと考えられる．

Table 3 Location and Availability of LED Control

XBee1	Location	A	B	C	D	E	F
	LED Control	○	○	○	○	○	×
	RSSI (dbm)	−29	−45	−77	−73	−80	NA
XBee2	Location	A	A	A	A	A	A
	LED Control	○	○	○	○	○	○
	RSSI (dbm)	−28	−28	−28	−28	−28	−28

NA: Not Available

Table 4 Location and Availability of LED Control

XBee1	Location	F	F	F	F	F	F
	LED Control	×	○	○	○	○	×
	RSSI (dbm)	NA	−89	−217	−217	−217	NA
XBee2	Location	A	B	C	D	E	F
	LED Control	○	○	○	○	○	×
	RSSI (dbm)	−28	−49	−39	−39	−39	NA

NA: Not Available

6. インターネットからの制御

6.1 当初の検討した CGI 方式の問題点

本研究では，ユーザが PC から WEB を介してセンサネットワークの先にあるデバイスにアクセスする．具体的な構成として，Fig. 1 に示したプログラム構成を考えていた．ユーザの PC から CGI (Common Gateway Interface) プログラムにアクセスして，この CGI プログラムで XBee を制御する．この CGI プログラムは Python で記述するが，実際に CGI プログラムで XBee のシリアルインタフェースをオープンしようとすると，権限がないためオープンできないとのエラーメッセージが出力される．基本的に，ラズベリー・パイの OS では CGI のプログラムでデバイスファイルをオープンできず，管理者権限が必要と考えられる．

6.2 改良した方式

この問題を解決するため，XBee 制御を CGI プログラムから分離し，ユーザのアプリケーションプログラムとして常時起動させておく方式とした．Fig. 11 にこの方式の構成を示す．この場合，CGI のプログラムは XBee を制御するプログラムと通信を行う必要があるが，両方のプログラムは完全に別プロセスとして動作する．

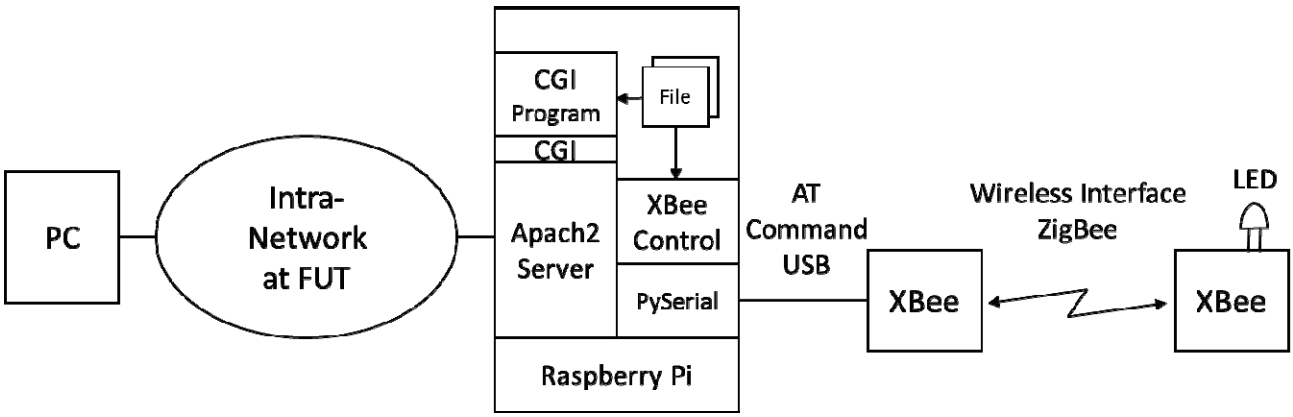

Fig. 12 Improved System Configuration

6.3 CGI プログラムと XBee 制御プログラムの通信方式

CGI プログラムと XBee 制御プログラムは別プロセスとして動作するため，プロセス間通信が必要となる．今回は簡単化のため，Fig. 13 に示すようにファイルを介して通信を行う方式とした．CGI プログラムと XBee 制御プログラムは，Fig. 14 に示すシーケンスで通信を行うことにより処理を実現する．

(1) CGI プログラムはネットワークから例えば LED の点灯要求を受け付けると，file-A を新規作成し，これに LED 点灯要求のコマンドを書き込む．

(2) XBee 制御プログラムは周期的に file-A が存在するか否かを監視する．

(3) file-A が存在することを検出すると，file-A を読出し，コマンドに従って LED の点灯フレームを XBee に送信する．
(4) XBee 制御プログラムはコマンドの処理が完了すると，file-B を新規作成し，これに完了したことを通知するレスポンスを書き込む．
(5) CGI プログラムは周期的に file-B が存在するか否かを監視する．
(6) CGI プログラムは file-B が存在することを検出すると，最初に file-A を消去する．次に file-B を読出し，レスポンスに従ってネットワークの先にある要求元に完了した旨のメッセージを返す．
(7) XBee 制御プログラムは周期的に file-A が存在するか否かを監視する．削除されたことを検出すると，(2)に戻り周期的に file-A が存在するか否かを監視する．この際，file-B を削除する．

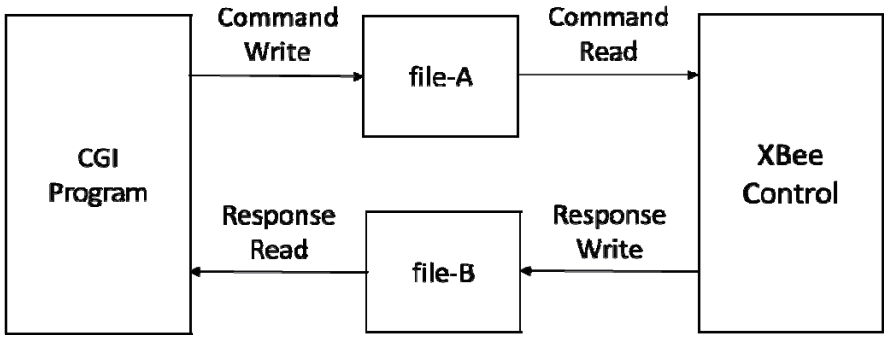

Fig. 13 Communication between CGI Program and XB Control through Files

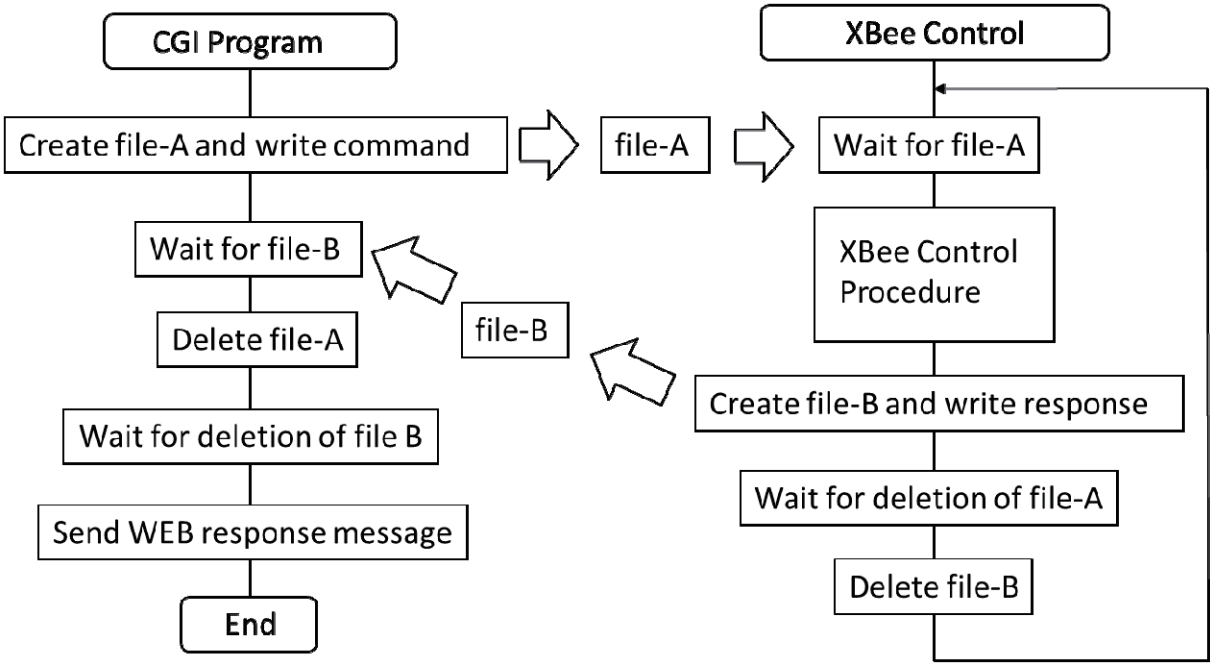

Fig. 14 Protocol between CGI Program and XBee Control

6.2 排他制御

ラズベリー・パイによるセンササーバでは以下の 2 種類の排他制御が必要となる．

(1) CGI プログラムと XBee 制御プログラム間でのファイルの排他制御
(2) CGI プログラム実行中にさらに CGI プログラムが実行された場合の排他制御（シリアライズ）

(1)に関してはファイルの書き込みを行うプロセスと読み出すプロセスを分け，同時に複数のプロセスが同じファイルの書込みを行わないようにしている．(2)に関しては，イントラネットを介して複数のパソコンが同時にセンササーバにアクセスする場合，対策が必要である．CGI では，センササーバが WEB によりアクセスされ CGI プログラムが起動されると，起動ごとに独立したプロセスが生成される．CGI プログラム実行中に，他のパソコンが CGI プログラムを起動すると同時に複数のプロセスが同じ処理を行うことになり，1 台のみの XBee のコーディネータに処理要求が重なる恐れがある．これを解決するために，CGI の受付で同時には 1 つの処理要求のみ受け付け，他の要求は待たせる排他制御を行うこととした．このために専用のファイルを設けてこのファイルにロックを掛けることにより排他制御を実現した．

排他制御の効果を確認するためセンササーバが送受信した HTTP メッセージを Wireshark[15] により記録した．Fig. 15 は Wireshark の画面を示している．この確認は Fig. 16 の試験構成で行い，2 台の PC (PC1，PC2)から手動で同時にセンササーバに要求を送っている．2 台の PC とセンササーバはそれぞれ Fig. 16 に示す IP アドレスが割り当てられている．

Fig. 17 は HTTP メッセージのシーケンスを図示したもので，先頭のメッセージの受信時刻を 0 として各メッセージの受信または送信時刻を示している．これから分かるように，PC1 (172.16.34.15) と PC2 (172.16.34.66) からほぼ同時に HTTP の GET メッセージがセンササーバ (172.16.34.93) に送られているが，PC1 と PC2 への応答メッセージ送信は約 2.5 秒の時間をおいて行われている．これから，PC1 からの GET メッセージが先に処理され，それが終了した後に PC2 からの GET メッセージが処理されたと考えられ，排他制御が行われていることが確認できる．

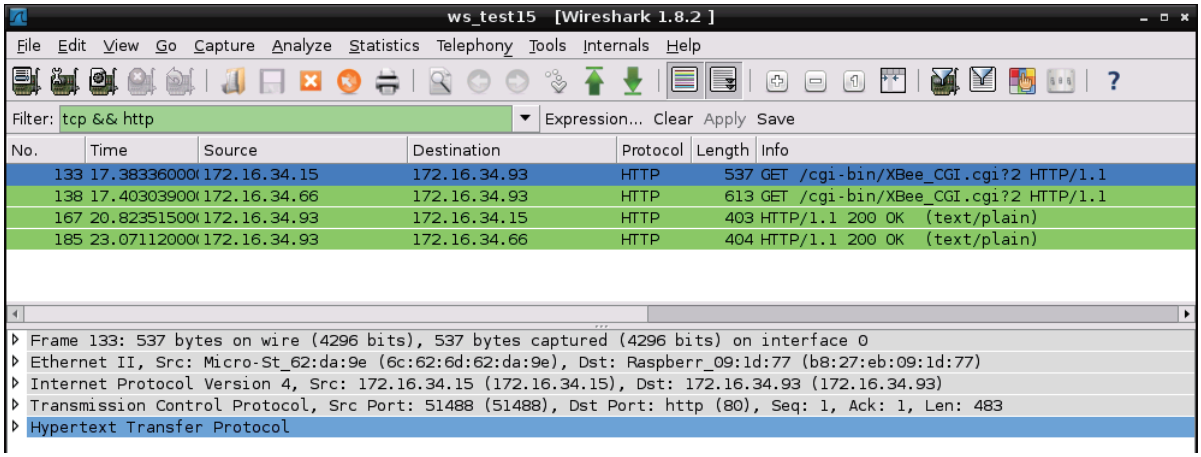

Fig. 15 Packet Capture Window by Wireshark

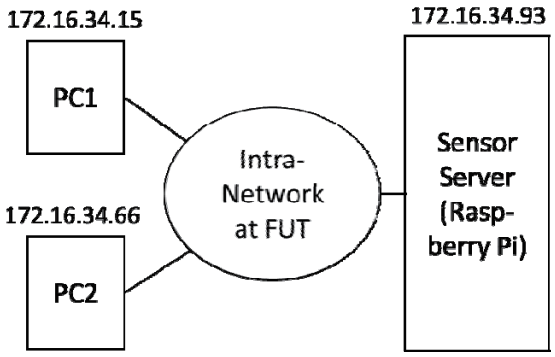

Fig. 16 Test Configuration

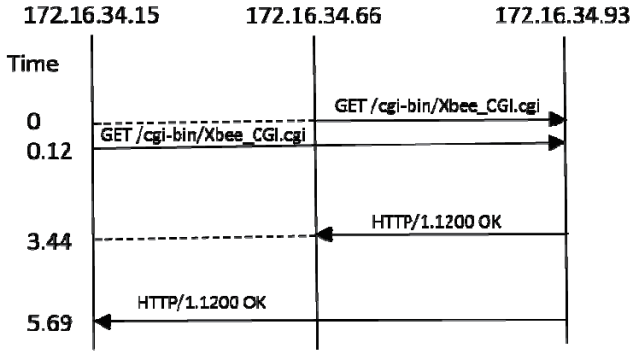

Fig. 17 HTTP Message Time Sequence

7. 結　　言

教育用センサネットワークの基盤技術開発について説明した．Python によりオープンなソフトウェアで実現しており，学生が自由に工夫を加えることができる．教育用の教材として開発したセンサネットワークは多くの技術要素を含んでおり，WEB サーバ，インターネットプロトコル，インターネットアプリケーション，ラズベリー・パイによる Linux 系 OS，Python によるプログラミング，CGI プログラミングなどの学習に展開できる．今後，開発成果を PBL 教育へ展開を図る予定である．また，センサネットワーク規模の拡大やエンドデバイスによる XBee モジュールの省電力動作とその評価が課題として残っており，卒業研究や PBL 教育における学生の課題として実施する予定である．

謝　　辞

本研究は，福井工業大学特別研究費の助成を受けたものである．ここに記して謝意を表す．

文　　献

(1) SERCOM, “IoT（Internet of Things）とは”, http://www.sercomm.co.jp/solution/survice/iot.html (参照日 2014 年 12 月 12 日).
(2) 総務省, “平成 16 年版 情報通信白書”, http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h16/html/G1401000.html (参照日 2014 年 12 月 12 日).
(3) 大宮知己, 織毛直美, “M2M を取り巻く標準化動向”, http://www.ntt.co.jp/journal/1204/files/jn201204063.pdf (参照日 2014 年 12 月 12 日).
(4) 阪田史郎編著, センサネットワーク, (2006), pp.20-25, オーム社.
(5) 間瀬憲一，阪田史郎，アドホック・メッシュネットワーク, (2007), pp. 22-25, コロナ社.
(6) 鄭　立, 実践入門ネットワーク ZigBee 開発ハンドブック, (2006), pp. 29-45, リックテレコム.
(7) ZigBee SIG ジャパン, “ZigBee SIG-J”, http://www.zbsigj.org/ (参照日 2014 年 12 月 12 日).
(8) Raspberry Pi Foundation, “WHAT IS A RASPBERRY PI?”, http://www.raspberrypi.org/ (参照日 2015 年 2 月 9 日).
(9) インターフェース編集部, お手軽 ARM コンピュータ ラズベリー・パイで I/O, (2013), CQ 出版社.
(10) 鹿間敏弘, “ET ロボコンのロボットを利用したプログラミング教育用教材の開発”, 福井工業大学研究紀要, Vol.43, (2013), pp. 9-20.
(11) 日本 Python ユーザ会, “プログラミング言語 Python”, http://www.python.jp/ (参照日 2014 年 12 月 12 日).
(12) Digi International Inc., “XBee®/XBee-PRO® ZB RF Modules”, http://www.adafruit.com/datasheets/ (参照日 2015 年 2 月 11 日).
(13) 濱原和明, 佐藤 尚一ほか, 超お手軽無線モジュール XBee, (2012), CQ 出版社.
(14) Chris Liechti , “pySerial API”, http://pyserial.sourceforge.net/pyserial_api.html (参照日 2015 年 2 月 17 日).
(15) Wireshark プロジェクト, “Wireshark”, http://sourceforge.jp/projects/wireshark/ (参照日 2015 年 2 月 24 日).

（平成 27 年 3 月 31 日受理）