タブレット型端末による無線センサネットワークの管理に関する研究

油井 誠志・宮本 博永・布施 嘉裕・中込 広幸・古屋 雅章

Study on Wireless Sensor Network Management by Tablet Terminal

Seishi YUI, Hironaga MIYAMOTO, Yoshihiro FUSE, Hiroyuki NAKAGOMI and Masaaki FURUYA

要　約

センサネットワークの普及にともない注目を浴びつつある通信規格 IEEE802.15.4 は，メッシュ型のネットワークを構築することが可能である．メッシュ型のネットワークは複数の通信経路が確保できるため障害に強いことが長所として挙げられるが，複数の通信モジュール同士がつながるため障害箇所の特定が難しい．そこで，ZigBee を通信プロトコルに使用してメッシュ型ネットワークを構築し，シングルボードコンピュータでネットワーク情報を収集，タブレット型端末で表示・管理できるシステムを開発することにし，今年度はメッシュ型ネットワークの形状（接続状態）を確認できるシステムを開発した．

1. 緒　言

近年，東日本大震災の影響もあって省エネルギー技術が脚光を浴びている．その 1 つであるエネルギー管理システムは，センサネットワークから入手した情報から無駄なエネルギー消費を抑えようという試みであり，そのハードウェア基盤となるセンサネットワークも重要性を増している．

例えば，工場等でエネルギー管理システムを導入する目的でセンサネットワークを新たに構築する場合，センサを有線で接続しようとすると工事が必要になる場合が多い．センサを壁面に埋め込みケーブルを壁面内で配線しようとすると，一時的に操業を停止しなければならない場合もあり，多大な費用と時間が発生してしまう．安価に抑えようとするとセンサやケーブルが露出して見た目も悪く，ケーブルが運搬等の障害になることも少なくなかった．これらの理由から，今後は無線でセンサネットワークを構築する場合が多くなると考えられる．

無線センサネットワークの構築に適しているとされる無線通信規格が IEEE802.15.4 である．低消費電力と低コストを実現しており，メッシュ型ネットワークを構築できることが特徴となっている．しかし，メッシュ型ネットワークは管理が難しく，管理用のツールも多くはない．

そこで工場等のエネルギーマネジメントを実施する場合に活用できるタブレット型端末からネットワークが管理できる無線ネットワークシステムの開発を目的とし，今年度はメッシュ型ネットワークの形状を把握することを目的とした．

2. 仕　様

IEEE802.15.4 をベースとしたメッシュ型ネットワークを構築し，センサネットワークとして使用するために次のように仕様を決定する．また，その構成を図にすると図 1 のようになる．

- ○ 通信プロトコルとして ZigBee を使用する．
- ○ IEEE802.15.4 対応の通信モジュールをシングルボードコンピュータに USB 接続し，シングルボードコンピュータ上でデータを収集する．
- ○ シングルボードコンピュータと接続する通信モジュールは Coordinator とする．
- ○ シングルボードコンピュータでは，Web サーバおよびデータベースサーバを構築する．
- ○ タブレット型端末とシングルボードコンピュータは同一 LAN 上に存在するものとし，シングルボードコンピュータと LAN は有線で，タブレット型端末と LAN は無線で接続する．
- ○ 集計されたデータを閲覧するアプリケーションは Web アプリケーションとし，タブレット型端末から Web ブラウザを介してデータを閲覧する．
- ○ Web アプリケーションは，次の 4 つの機能を有する．
  - ● ネットワークの形状確認
  - ● 通信モジュールの状態出力
  - ● 通信モジュールへのコマンド送信および応答出力
  - ● センサの監視

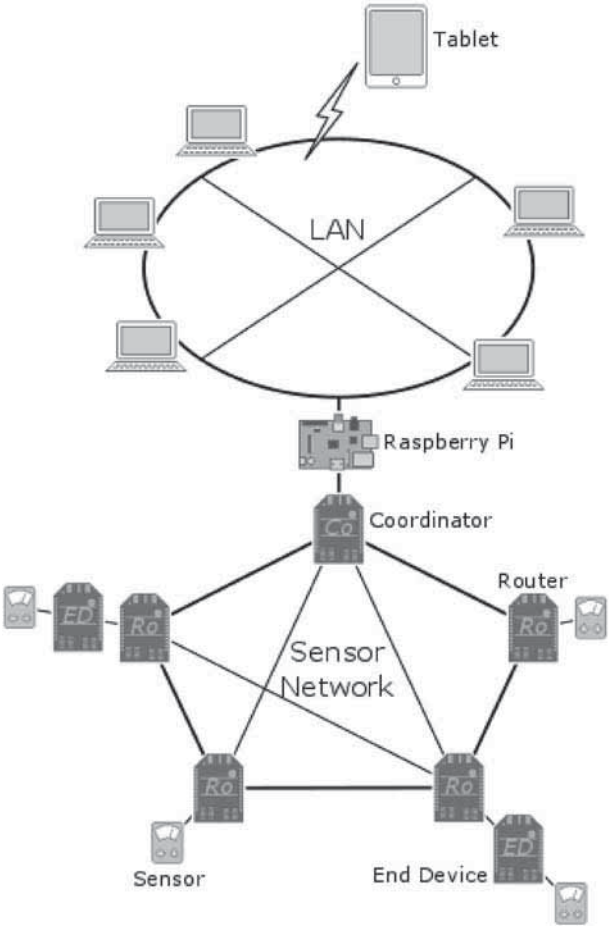

図 1　システム構成図

3. 結　果

3−1　使用機器

データ収集サーバとして，ラズベリーパイ財団が開発しているシングルボードコンピュータ Raspberry Pi Model B を使用し，OS として同財団が配布しているリナックスディストリビューション Raspbian をデフォルトインストールした．OS インストール後に，リポジトリより oracle-java8-jdk パッケージおよび librxtx-java パッケージを追加インストールした．また，Google Code から xbee-api をダウンロードし展開している．

通信モジュールとしてディジ インターナショナル社の XBee-PRO ZB S2B モジュールを 8 台使用した．ZigBee の仕様に合わせて 1 台を Coordinator に，残りの 7 台を Router もしくは End Device として API モードでファームウェアを更新する．また Coordinator は Raspberry Pi と USB ケーブルで接続した（図 2）．

タブレット型端末には Google 社の Nexus 7(2013)を使用しており，OS のバージョンは Android 5.0.2 となっている．

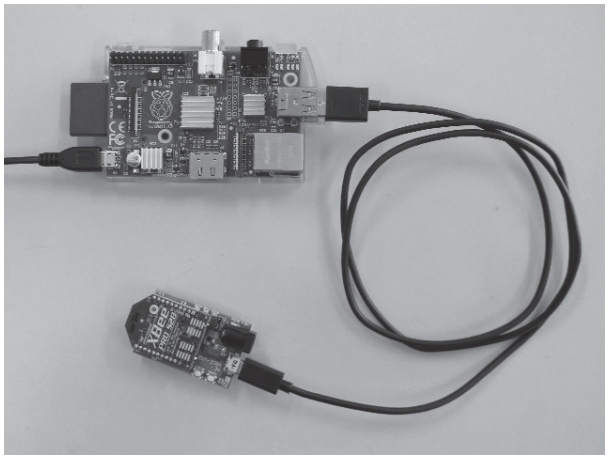

図 2　USB 接続された Raspberry Pi と XBee

3−2　ネットワーク形状の把握

ZigBee では，最初に Coordinator に電源が投入されていれば，Router および End Device は電源投入時に自動的にネットワークアドレスが割り振られ，メッシュ型のネットワークを形成する．

しかし，ネットワーク全体の接続を一括で把握できるコマンドが用意されていないため，一つ一つのモジュールに接続問い合わせを行い，ネットワークの形状を把握しなくてはならない．そのための手順は以下の通り．

1. Raspberry Pi に接続されている Coordinator に接続問い合わせを行う．
2. 応答から取り出した接続モジュールのアドレスを問い合わせキューにエンキューし，問い合わせ済みセットに Coordinator を登録する．
3. 問い合わせキューからアドレスをデキューし，問い合わせ済みセットと照合する．セットに登録されていなければ，該当するモジュールに接続問い合わせを行う．既に問い合わせ済みだった場合は，次のアドレスをデキューする．
4. 応答からアドレスを取り出し問い合わせキューにエンキューする．問い合わせを済ませたアドレスは問い合わせ済みセットに登録する．
5. 問い合わせキューが空になるまで 3〜5 を繰り返す．

1 回の問い合わせに対して，応答は接続モジュール数と同数返ってくるため，応答回数で接続数がわかることになる．しかし IEEE802.15.4 の通信速度は遅いため，必ずしも問い合わせ直後に応答が返ってくるとは限らない．そのため，応答を 10 秒程度待つように設定した．結果，1 回の接続状態のスキャンにモジュール数×10 秒程度の時間がかかることになった．

3−3　スキャン結果

3-2 で示したスキャンを行うことで，ネットワークの接続状態を表した一覧が得られる（表 1）．

表 1　スキャン結果例（モジュール数 8）

モジュールアドレス	接続モジュールアドレス
0013A20040991D72 （Coordinator）	0013A2004091BDA1 0013A20040991ECF 0013A200409EB1DC
0013A2004091BDA1 （Router）	0013A20040991D72 0013A200409EB209 0013A20040991ECF 0013A200409EB1DC
0013A20040991ECF （Router）	0013A20040991D72 0013A20040991E99 0013A2004091BDA1 0013A2004091Bf3F 0013A200409EB1DC
0013A200409EB1DC （Router）	0013A200409EB249 0013A20040991D72 0013A2004091BDA1 0013A20040991ECF
0013A200409EB249 （End Device）	0013A200409EB1DC
0013A20040991E99 （End Device）	0013A20040991ECF
0013A200409EB209 （End Device）	0013A2004091BDA1
0013A2004091BF3F （End Device）	0013A20040991ECF

ZigBee には，個々の通信モジュールに ID として使用できるアドレスが 64bit アドレスと 16bit アドレスの 2 種類用意されている．表 1 は 64bit アドレスで記述している．64bit アドレスはメーカー出荷時に付与されたシリアル ID で，同じアドレスを持つ機器は存在しないため，容易に機器を特定できるようになる．もう一方の 16bit アドレスは，ネットワーク参加時に Coordinator が割り振るソフトウェアアドレスで，毎回ランダムに決定されるため機器の特定が難しい．どちらかのアドレスを指定するだけでコマンドを送信することが可能であるが，本研究では両アドレスを指定して通信するプログラムを作成している．

表 1 で得られたネットワークの接続状態をグラフ化したものが図 3 である．

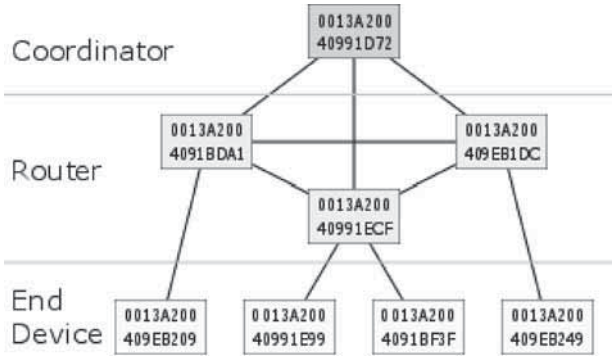

図 3　ネットワーク図

3−4　操作手順

管理システムは次の手順で操作する．

1. Coordinator に設定した通信モジュールを USB ケーブルで Raspberry Pi に接続し，通信モジュールの電源を投入する．
2. 残りの通信モジュールの電源を投入する．
3. Raspberry Pi の電源を投入し，システムを起動する．システムは Java で記述・コンパイルしてある．オプションとして java.library.path に /usr/lib/jni を指定し，またクラスパスに xbee-api-0.9.jar と log4j.jar，RXTXcomm.jar を追加する（図 4）．
4. モジュール数×10 秒程度でネットワークのスキャンが終了する．その後タブレット端末を起動し Web ブラウザを起動する．アドレスバーに Raspberry Pi の IP アドレスを指定すると，ネットワーク接続の表が出力される（図 5）．
5. 図 5 中のアドレスをクリックすると，個々の通信モジュールの状態が確認できる（図 6）．

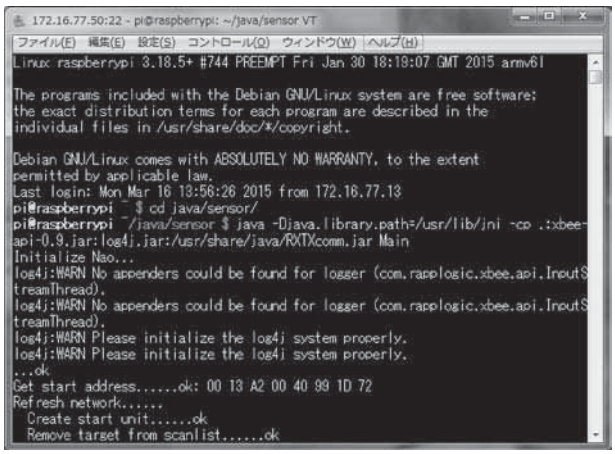

図 4　起動画面

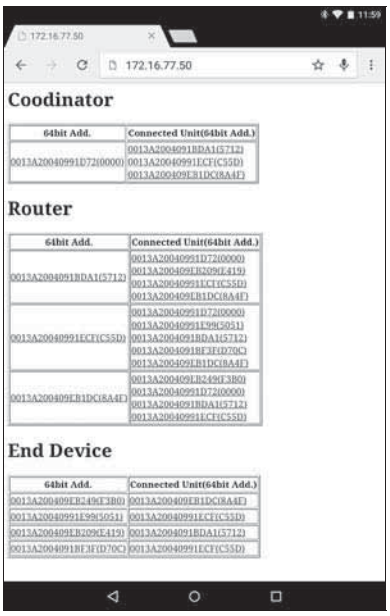

図 5　ネットワーク接続の表

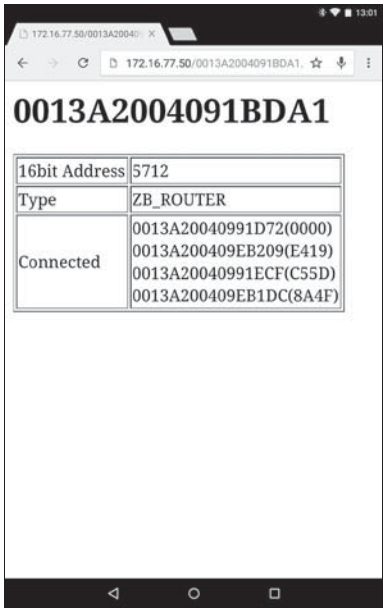

図 6　モジュールの状態

3－5　課題

本研究は，2 年計画の 1 年目が終了したところである．ネットワーク接続の表が出力できるようになった．この表を図 3 のように階層型にネットワーク図を出力したところ，Router の数が多いときに接続線の本数が増えてしまい，中央付近は接続状況がかえってわかりづらくなってしまったため，デザインを含めて図の表示は今後の課題である．

4.　結　言

IEEE802.15.4 をベースとし ZigBee を通信プロトコルとしたメッシュ型ネットワークの構築を試みたところ，Web アプリケーションを通じてネットワークの形状が把握できることを確認した．

来年度は，通信モジュールへのコマンド送受信，センサーデータの受信に関するソフトウェア開発を行う予定である．また，通信モジュールの状態についても，出力される項目を増やす考えである．

プログラムに関しては，現プログラムでは設計が甘く，オブジェクト指向が生かしきれていない．早期に設計を見直して，再コーディングしなくてはならない．

今年度，実現できなかったネットワーク図のグラフ出力に関しても，より見やすいグラフをデザインし出力することを目指す．

参考文献

1)　xbee-api：https://code.google.com/p/xbee-api/（2014/10/16 - 2015/02/13）

2)　Linux 工作室：http://penguin.tantin.jp/hard/XBee.html（2014/10/16 - 2015/02/13）

3)　SONY AROUJE：Connecting XBee to Raspberry Pi：http://java.dzone.com/articles/connecting-xbee-raspberry-pi　（2014/10/27 - 2014/11/13）

4)　XBee on the Raspberry Pi：http://rapplogic. blogspot.jp/2013/06/xbee-on-raspberry-pi.html（2014/10/27 - 2015/01/30）