WLAN と ZigBee の共存に向けた

AA（Access Point-Assisted） CTS-Blocking に関する研究

佐伯 良光

平成 27 年 2 月

修士課程

情報知能工学専攻

社会情報システム工学コース

第1章

システムの設計と実装

本章では，提案する AA CTS-Blocking の設計，実装について説明する．まず，ZigBee 通信の実装に向けて UML を用いた設計を行った．次に，この設計を基に ZigBee 通信の実装を行ったので，説明する．その後，制御 PC を含めたシステム全体の実装について述べる．

1.1　ZigBee 通信の設計

本システムでは ZigBee ノード，ZigBee 基地局に MICA を利用する．MICA において，ZigBee 通信の実装は TinyOS によるイベント駆動型プログラミングとなる．イベント駆動型プログラミングでは，起動すると共にイベントを待機し，起こったイベントに従って処理を行う．イベントとは，プログラムの実行に際し，データを受信した，起動していたタイマが完了した等，何らかのアクションが発生した際にプログラムに発信される信号である．イベントを待機している間，MICA は何らかの状態を持つ．このような状態遷移のフローをを記述するのに最適なのが，UML のステートマシン図である．UML(Unified Modeling Language) は，抽象化したシステムをグラフィカルな記述でモデル化し，汎用的なプログラム設計図を与える．本システムの設計では，システムを構成する ZigBee ノード，ZigBee 基地局の状態遷移を記述するためにステートマシン図を利用した．以下で，詳細を説明する．

1.1.1　ステートマシン図

ステートマシン図は無償 UML モデリングツールである astah* community を利用して作成した．ステートマシン図の遷移は矢印で表されており，説明はイベント［ガード条件］／アクションで記述される．ガード条件は直前に発生したイベントの評価をするための条件である．評価値は真もしくは偽の値を持ち，ガード条件に対して真であるときのみ遷移が許される．アクションは遷移が起こると同時に実行される動作である．

ZigBee ノード

ステートマシン図は状態と状態遷移によって表されることを前項で説明した．状態を Sn（n は添字），イベントを En，ガード条件を Gn，アクションを An と表記すると，ZigBee ノードの状態遷移は以下の通りになる．

- S1：受信待機状態
- E1：ZigBee 基地局から送信要求メッセージを受信する
- G1：送信要求メッセージの受信成功
- A1：スロット時間待機するためのタイマを起動する
- S2：送信準備状態
- E2：タイマが終了する
- A2：データを送信する
- S3：送信完了状態
- A3：タイマをリセットする

本システムでは，ZigBee 通信時間は CTS-Blocking が有効である時間に等しい．この時間はわずかであり，送信失敗した場合にタイムアウトによる再送の仕組みを設ける余裕が無い．従って，送信成功かどうかの判定は行わない．

これらを総合すると，ZigBee ノードのステートマシン図は図??の様に描画できる．

ZigBee 基地局

ノードの場合と同様に，状態を Sn，イベントを En，ガード条件を Gn，アクションを An，と表記すると ZigBee 基地局の遷移は以下の通りになる．

- S1：シリアル信号待機状態
- E1：制御 PC から信号を受信する
- G1：信号の受信成功
- A1：CTS-Blocking 有効時間を測るためのタイマを起動する
- S2：ブロードキャスト準備状態
- A2：ZigBee ノードへブロードキャストする
- S3：受信待機状態
- E3：ZigBee ノードからデータを受信する
- G3：データの受信成功
- A3：コンソールにデータを表示する
- S4：受信完了状態
- E3’，E4’：タイマが完了する
- A3’，A4’：タイマをリセットする

本システムでは，ZigBee 通信時間は CTS-Blocking が有効である時間に等しい．この時間はわずかであり，タイムアウトによる再送の仕組みを設ける余裕が無い．従って，送信成功かどうかの判定は行わない．よって，S3 状態は S2 状態から A2 の動作を伴い自動遷移するよ

うにした．他方で，S4 状態に遷移した後，他のノードからのデータを受信して表示する必要がある．そのため，S4 状態に入った際すぐに S3 状態へと自動遷移するようにした．そして，A1 で起動させたタイマが終了した際は，S3/S4 状態に関わらず S1 状態に遷移する事とした．

これらを総合すると，ZigBee 基地局のステートマシン図は図??の様に描画できる．

1.2　ZigBee 通信の実装

前節で作成したステートマシン図を基に，ZigBee ノード及び ZigBee 基地局への実装を行う．プラットフォームは MICA に対応する TinyOS を利用した．TinyOS はオープンソースで開発がされているセンサネットワークノード向けのイベント駆動 OS である．TinyOS 上で利用されるイベント駆動型プログラミング言語が nesC であり，C 言語の拡張である．ZigBee 通信のコーディングはこの nesC を利用して行った．nesC は一般的なプログラミング言語ではないため，次項で nesC の概念について説明し，その後 ZigBee ノード及び ZigBee 基地局への実装を説明する．

1.2.1　nesC の概念

本稿では nesC の基本概念について，簡単なチュートリアルを交えながら説明する．

1. コンポーネント

   コンポーネントとは，カプセル化されたソフトウェアの部品であり，同等の機能をもつものと交換可能である．nesC において，アプリケーションのプログラムはコンポーネントで作られ，その組み合わせによりプログラム全体が形成される．コンポーネントは 2 つのスコープ（ある変数や関数が特定の名前で参照される範囲）を定義する．1 つはコンポーネントの規定部スコープ（コンポーネントのインタフェースインスタンスの名前を含む）であり，もう 1 つはコンポーネントの実装部スコープである．コンポーネントはタスク（task）という形によって内部的に並列動作性を持つ．制御の流れはコン

ポーネントのインタフェースを通じて他のコンポーネントへと流れる．これらの制御の流れはタスクやハードウェア割り込みから生じている．

2. 一連のインタフェース（interface）によるコンポーネントの動作の規定

   – インタフェースはコンポーネントによって提供（provide）されるか利用（use）される．

   – 提供インタフェースは、コンポーネントがそのユーザに提供する機能をあらわす．

   – 利用インタフェースはコンポーネントが処理を行うために必要とする機能をあらわしている．

- インタフェースは双方向性を持つ

  – インタフェースはインタフェースの提供者が実装すべき関数群（コマンド）とインタフェースの利用者が実装すべき関数群（イベント）を示す．

  – これにより一つのインタフェースでコンポーネント間の複雑なやりとりを表現することができる（例えば興味のあるイベントに登録しておけば、そのイベントが起こったときにコールバックされる）．

  – TinyOS ではすべての時間のかかる処理（例 : パケット送信）はノンブロッキングであるため，このことは重要である．時間のかかる処理の完了はイベント（例 : 送信完了）を通じて知らされる．

  – 概してコマンドは呼び下げ，すなわちアプリケーションコンポーネントからハードウエアに近いところへ向かって順に呼ばれていく．一方，イベントは呼び上げられていく．

  – いくつかの低レベルイベントはハードウエア割り込みに結びつけられている（どのように結びつけられるかは基本的にはシステム依存である)．

- コンポーネントはインタフェースを通じて互いに静的にリンクされる．

  - これは実行時の効率を向上させ，堅牢性を増し，プログラムのよりよい静的解析を可能とする．

- nesC はコードがすべてコンパイラによって生成されるようにデザインされている．
  - これによってよりよいコード生成と解析が可能となる．例として，nesC のコンパイル時のデータ競合検出があげられる．
- nesC の並列処理モデルは処理完了まで走りきる（run-to-completion）タスクと割り込みハンドラに基づいている．
  - 割り込みハンドラはタスクや他の割り込みハンドラに割り込むことができる．
  - nesC コンパイラは割り込みハンドラが起こしうる潜在的なデータ競合を通知してくれる．

コンポーネント図

前項で説明した nesC では，

♣ ”プログラムは コンポーネントで作られ，コンポーネントによりプログラム全体が形作られるよう組み立てられる．”

とある．そこで，コンポーネント毎の機能を明確にするために，コンポーネント図を作成した．図**??**にそれを示す.（コンポーネント図の説明）

ZigBee ノード

ZigBee 基地局

♣ ：シリアル信号待機状態

- E1：制御 PC から信号を受信する
- G1：信号の受信成功

- A1：CTS-Blocking 有効時間を測るためのタイマを起動する
- S2：ブロードキャスト準備状態
- A2：ZigBee ノードへブロードキャストする
- S3：受信待機状態
- E3：ZigBee ノードからデータを受信する
- G3：データの受信成功
- A3：コンソールにデータを表示する
- S4：受信完了状態
- E3’，E4’：タイマが完了する
- A3’，A4’：タイマをリセットする

1.3　シーケンス図

図**??**にシステム全体のシーケンス図を示す．シーケンス図は，オブジェクト間の相互作用を時系列に沿って表現するダイアグラムである．シーケンス図での時間は，ライフラインに沿って上から下に進む．ここでは制御 PC を含めた，システム全体の処理の流れが確認できる．

制御 PC

同様に，図**??**に，ZigBee ノードのステートマシン図を示す．ZigBee ノードに起こるイベント Ev 及びイベントハンドラ EvH （イベントが発生した際に実行すべきサブルーチン）は以下の通りである．

- A

- B
- C

従って，状態は

- A
- B
- C

の 3 つが存在すれば良いため，図??の様に作成した．