Vol.2013-SE-179 No.32
2013/3/12

# モデルベース開発におけるブロック線図モデルの保守性向上手法の提案

弓倉陽介[†1]　川田秀司[†1]　川勝則孝[†1]

近年，特に車載分野の組込みソフトウェア設計において，ブロック線図に基づくモデルベース設計手法の利用が拡大してきている．しかし，モデル化の仕方によっては，状態遷移の情報がデータフローの表現としてブロック線図に表れることがあり，モデルの保守性を下げる要因となっている．ブロック線図はデータフローの表現には適しているが，状態遷移の表現に適しているとは言い難い．本研究では，ブロック線図から状態遷移とデータフローに関わる情報を抽出することで，モデルの保守性を向上させる手法を提案する．

# A Method to Increase Maintainability of Block Diagram Models in Model-Based Development

YOSUKE YUMIKURA[†1]　HIDEJI KAWATA[†1]
NORITAKA KAWAKATSU[†1]

Recently, block diagram approach is getting popularity in the domain of automotive embedded software development. Unfortunately, maintainability of block diagram models may decrease depending on modeling method. One possibility is that block diagram models may contain information about state transitions in the guise of data flows. A block diagram model is suitable for data flows, but is not suitable for modeling state transitions. In this report we will present a method that extract state transition information and data flow information from block diagrams, which increases maintainability of block diagram models.

## 1. はじめに

近年，組込みソフトウェアは開発規模が大規模化する一方で，製品に対して高い品質と短い納期が求められている[1]．モデルベース開発はこの高品質かつ短納期の要求に対応することができる技術として注目を集めている[2]．特に，車載分野のソフトウェア開発では，モデルベース開発手法が盛んに用いられている．

車載分野では組込みソフトウェアの設計においてSimulink [a]という開発ツールがよく利用される．このツールはモデルベース開発をサポートする機能を豊富に備えており，モデル編集機能，シミュレーション機能，トレーサビリティ機能，コード自動生成機能，テスト支援機能により開発の各工程の作業を支援する．この開発環境ではモデルの表現形式としてブロック線図が使用される．ブロック線図は制御工学の分野でよく利用されるダイアグラムである．システムのデータフローを表現するのに適しており，フィードバック制御やフィードフォワード制御といった連続系のシステムを表現することに優れたダイアグラムである．Simulink では，このブロック線図に独自の拡張を施し，条件分岐や繰り返し制御など離散系の制御ロジックの記述をサポートしている[3]．

しかし，ブロック線図で大規模なモデルを記述すると、モデルの保守性が低下してしまうことがある[4]．このことを引き起こす大きな要因の１つに、システムの持つ状態遷移の情報がブロック線図を用いて表現できることがある．

本研究では，上記に示すような保守性低下の要因を踏まえ，ブロック線図から状態遷移とデータフローに関する情報を抽出する手法を提案し，ブロック線図の保守性が改善されることを示す．

## 2. 状態遷移を含むブロック線図の保守性の課題

ブロック線図の保守性指標として以下の指標が定義されている[4]．

- ブロック数：ブロックの総数．
- 信号線数：信号線の総本数．
- 信号線交錯数：２本の信号線が結線を作らずに交差している総回数で，同じ２本の信号線でも複数の交差がありうる．
- レイアウト面積[b]：ブロックの配置された領域の面積．上端，下端，左端，右端のそれぞれのブロックの位置からこの面積を計算する．

ここでは，上記の指標を用いてブロック線図が状態遷移の

†1 (株)東芝　ソフトウェア技術センター
Toshiba Corporation, Corporate Software Engineering Center

a Simulink は，米国 The MathWorks, Inc.の登録商標です．

b 実際の指標では，レイアウト面積／表示可能領域だが，Simulink では表示可能領域を拡大・縮小することができるため，ここではレイアウト面積のみを考える．実際にブロック線図全体を収めるために縮小し過ぎると，図の可読性が低下するため，このように取り扱うことは妥当と思われる．

情報を含むことにより,保守性が低下することを説明する.

ブロック線図に状態遷移情報が含まれる場合，これらの指標値が悪くなる傾向がある．具体的には次のような２つの傾向が表れる．

一つ目は，状態遷移時に判定されるガード条件がブロック線図内の要素に分散されることで信号線数，信号線交錯数が増加する（図 1－左）．状態遷移のガード条件が増加する，または，複雑化する度に条件判定に利用するデータを参照するための信号線がブロック線図内に配線される．これにより，信号線が増加し配線が難しくなり，その交錯が増加する．

二つ目は，状態毎に異なる挙動を実現するためのブロック線図の記法により，信号線の交錯数，および，レイアウト面積が増加する（図 1－右）．この記述方法には，配線のループが用いられる．ループが構成されるためには，データフローの下流のブロックから上流のブロックへ信号線を接続しなければならず，ブロック線図のレイアウト領域を大きく外側に膨らませることになる．また，ループが形成された後にループの内側の配線を編集する場合，ループする信号線との交錯が生じないように配線されるため，内部での配線が難しくなり，それらの信号線の交錯が増加すると考えられる．

このようにブロック線図が状態遷移を含む場合，ブロック線図の保守性に関する指標が低下する．

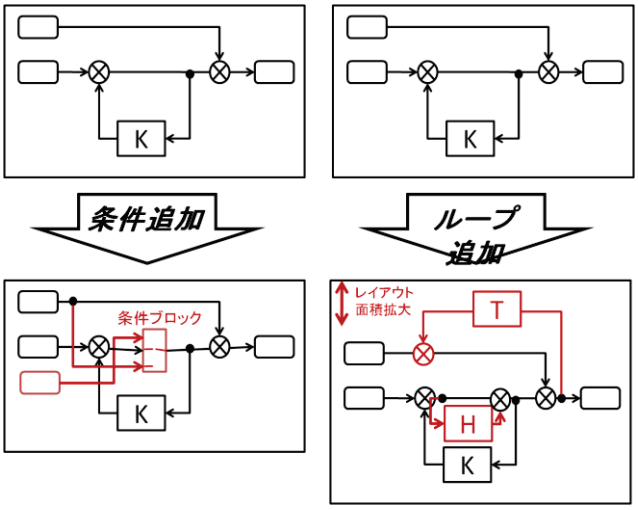


図 1：状態遷移情報が含まれるブロック線図

## 3. ブロック線図からの状態遷移／データフロー情報の抽出手法

まず，本手法で利用する用語について説明する．その後，本手法の概要と各手順の詳細を述べる．

### 3.1 用語

以下では，本手法の説明内に表れる用語を定義する．

- ロジック選択ブロック：データフロー選択端子に入力されたデータ値を利用して，出力するデータフローを決定するブロックのことを指す（Simulink における Multiport Switch, Switch, If, Switch Case ブロックが該当）．
- データ記憶ブロック：入力されたデータ値を保存するブロック（Simulink における Unit Delay ブロックなどが該当）．
- 共有変数ブロック：ブロック線図のいかなる場所からでもデータの読出し，書出しができるブロック（Simulink における Memory ブロックが該当）．
- 定数値ブロック：常に同じ数値を出力し続けるブロック（Simulink における Constant ブロックが該当）．
- 定性値ブロック：真偽値や列挙値を出力するブロック（Simulink における Relational Operator ブロックが該当）．
- 定性値出力ブロック：入力を受け付ける端子を持たず，ブロック毎に決められた定性値の範囲内のいずれかの値を出力するブロックのこと．
- サンプル時間：実際の時間に対するブロックの実行タイミングを表す値．

### 3.2 概要

本手法の手順は次の３つのステップに分けられる．なお，本手法を適用するブロック線図を BL，その部分として状態遷移を含有する領域を BLp と定義する．

Step 1.状態遷移情報を含むブロック線図の抽出

ブロック線図 BL から領域 BLp を抽出する．この抽出領域を決めるための起点となるブロックとして，ロジック選択ブロックを入力する必要がある．

Step 2.状態遷移情報の抽出

1 で抽出した BLp から状態遷移情報を抽出する．ブロック線図 BLp の状態変数が取りうる全ての状態をシミュレーションによって探査する．

Step 3.データフロー情報の抽出

2 で抽出した状態遷移情報を用いて，BLp からデータフロー情報を抽出する．ある状態遷移が発生した時にデータ出力に関与したブロック線図をデータフローとして抽出する．

この手法により，入力されたモデルを以下の 3 つの階層のモデルへと再構成することができる（図 2）．このように一つのブロック線図が複数のブロック線図に分割されるため，１画面あたりのブロック数，信号線数，レイアウト面積が減少し，信号線の交錯数が減少する．これにより，モデルの保守性を高めることができる．

- システムレイヤー：入力されたモデルで外界とのインタフェースに関する情報が現れる層．
- 状態遷移レイヤー：入力されたモデルで状態遷移に関する情報が現れる層．
- データフローレイヤー：入力されたモデルで特定の状態の時に出力されるデータを生成するデータフロ

ーに関する情報が現れる層.

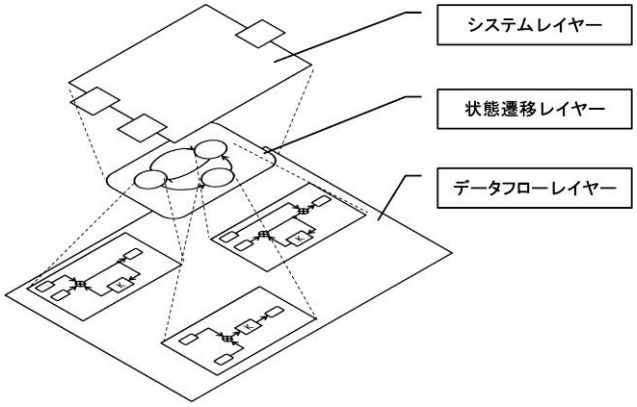


図 2：再構築されたモデル

### 3.3 Step1.状態遷移情報を含むブロック線図の抽出

ブロック線図 BL に含まれるロジック選択ブロック群 LSs を用いて， BL の部分領域 BLp を，状態遷移の情報を含有するように抽出する．具体的には，BL に含まれる LSs を入力として，以下の手順で BLp を抽出する．

1. 入力端子の探索
   LSs に含まれる各ロジック選択ブロックのデータ選択端子から信号線を逆方向に辿り，システムの入力端子までの領域 $P$ を取り出す.
2. 定性値ブロックの定性値入力ブロックへの置換
   領域 $P$ の入力端子から順方向に信号線を辿り定性値ブロックに到達するまで探索する．到達したそのブロックを定性値出力ブロックとして置き換える．この作業の結果作成される定性値出力ブロック群を以下のように定義する.

$$\mathbf{B} = \{B_1, B_2, \dots , B_s\}$$

   また，**B** に属する定性値出力ブロックそれぞれが出力しうる値の範囲を以下の記号でまとめる，

$$\mathbf{A} = \{ \boldsymbol{v}_1, \boldsymbol{v}_2, \dots , \boldsymbol{v}_u \}$$

$$v_i = \{ b_{i1}, b_{i2}, \dots , b_{is} \}$$

$$(b_{ij} \text{ は} B_j \text{の値 } (0<i<u+1, 0<j<s+1))$$

   なお，この定性値出力ブロック群に接続している信号線が以下で抽出される状態遷移モデルと外部とのインタフェースとなる.
3. BLp の出力
   領域 $P$ の全ての入力端子から定性値ブロックの探索を行ったか？
   - Yes → 残っている BLp を結果とし，終了.
   - No → 2 に戻って残りの入力端子から探索する.

### 3.4 Step2.状態遷移情報の抽出

ブロック線図 BLp に含まれる状態遷移の情報を抽出する．本手法で抽出する状態は次の三つの値の組みで表現される.

- ブロック線図の実行時刻の状態 T
- データフローの状態 **F**
- データ記憶ブロックの状態 **M**

状態を構成する各値の取得方法について以下で説明した後，状態遷移の抽出手順を記述する.

#### 3.4.1 ブロック線図の実行時刻の状態 T

ブロック線図 BLp を構成する全ブロックのサンプル時間について最小公倍数 $L$ と最大公約数 $G$ を計算する．$L$ は全ブロックの実行タイミングが一致するまでの時間，$G$ はシミュレーション時間の量子化単位となる．ブロック線図の実行時刻の状態 $T$ の範囲は以下のようになる.

$$T = \{ 0, G, 2G, 3G, \dots , L\text{-}2G, L\text{-}G \}$$

ただし，$L$=$G$ の時には $T$ は 0 しか取り得ないため，ブロック線図の実行時刻の状態は，常に 0 として取り扱う.

#### 3.4.2 データフローの状態 F

ブロック線図 BLp を，ロジック選択ブロックのデータ選択端子から信号線を逆方向に辿ることで通過する全ての信号線について，信号線のソース側を $B_{pre}$，ターゲット側を $B_{post}$ と定義する．$B_{pre}$ のサンプル時間を $S_{pre}$ と，$B_{post}$ のサンプル時間 $S_{post}$ と定義する．もし，$S_{pre}$ と $S_{post}$ の最大公約数が $S_{pre}$ でないなら，$B_{pre}$ と $B_{post}$ の間の信号線は値を一定期間保持する必要があるため，これを状態として扱う．$B_{pre}$ の識別子を $i$，$B_{post}$ の識別子を $j$ とし，この状態を $D_{ij}$ と定義する.

$n$ 個の信号線の状態 $D_{ij}$ があるとき，以下の記号で，これらの値を取り扱う.

$$\mathbf{F} = \{ F_1, F_2, F_3, \dots , F_n \}$$

この $F_k$ のサンプル時間を $C_k$ という記号で扱い，$F_k$ の初期状態は $I_k$ という記号で取り扱う.

#### 3.4.3 データ記憶ブロックの状態 M

ブロック線図 BLp 内のデータ記憶ブロックがあればこれをデータ記憶ブロックの状態の項として扱う． BLp 内に，データ記憶ブロックが $m$ 個あるとき，データ記憶ブロックは次の記号で取り扱う.

$$\mathbf{M} = \{ M_1, M_2, M_3, \dots , M_m \}$$

状態の中の一つの項を表す記号は $M_i$，この初期状態を $J_i$ という記号で取り扱う.

#### 3.4.4 状態遷移情報の抽出

データフローの状態の項目数が $n$ 個，データ記憶ブロックの状態の項目数が $m$ 個存在するときに，抽出しようとする状態遷移の状態変数は以下の形式で表される.

$$S = ( T, F_1, F_2, \dots , F_n, M_1, M_2, M_3, \dots , M_m )$$

また，$S$ の初期値は以下のように表される.

$$S_0 = ( 0, I_1, I_2, I_3, \dots , I_n, J_1, J_2, J_3, \dots , J_m )$$

以下では，この状態を網羅的に探索し，状態遷移の情報 **E** の抽出手順を説明する（図 3）．ここで、手順の中に現れる変数 **PL** と **E** について説明する．状態プール **PL** の元の第 1 項は状態変数 $S$ の取りうる値，第 2 項は状態が既に到達したことがあるかどうかを判定するフラグで End であれば

到達したことを，Yet であれば未到達であることを示す．**E** の元の第１項は状態遷移の遷移元の状態，第２項は遷移時の定性値出力ブロックの出力値の組み合わせ，第３項は遷移先の状態である．

1　**PL** = {($S_0$, Yet)}, **E** = {}.
2　**PL** の元で第 2 項が Yet のものがあるか？
- True → $n$ という名前で取り出し，3 へ．
- False → 終了

3　$n$ の第１項を $s$ として，$s$ の次の状態 $s$'を生成する．
4　$s$ の $T = t$ の時，$s$'の $T = ((t+G) mod L)$ とする．
5　**A** の元を取り出せるか？
- True → $\boldsymbol{v}_k$ を取り出し，6 へ．
- False → 2 へ．

6　ブロック線図 BLp の $\boldsymbol{v}_k$ を入力．それにより次の状態 $s$' の値が決まる．ただし，$s$'の各 $F_p$ において，$s$'の $T$ $mod\ C_p = C_p$-1 ならば，$F_p$ の値を計算する．
7　**E** = **E** ∪ ($n$, $\boldsymbol{v}_i$, $n$'), **PL** = **PL** - ($n$, Yet)+ ($n$, End).
8　($n$', Yet)∉**PL** ∧ ($n$', End)∉**PL** の時，**PL** = **PL**+($n$', Yet).
9　2 へ．

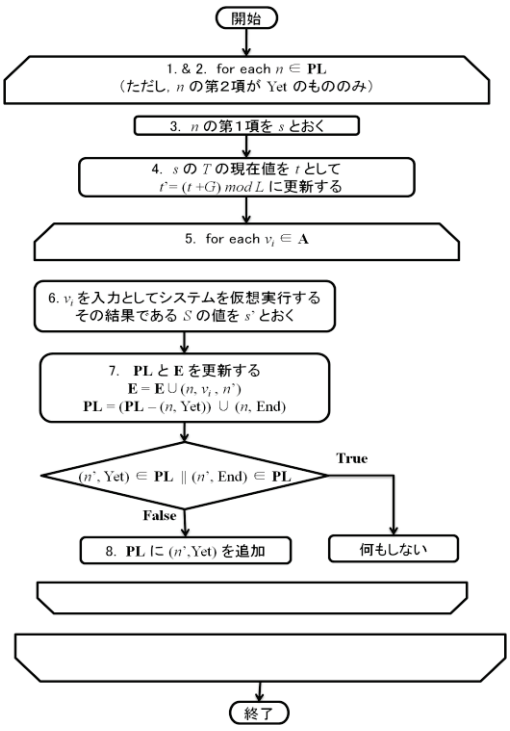


図 3：状態遷移情報の抽出のフローチャート

### 3.5 Step3.データフロー情報の抽出

Step2 で抽出された状態遷移を用いて，ブロック線図 BLp からデータフローの情報 **D** を抽出する．その手順を以下に記す．ここで、手順の中に出てくる変数 **D** について説明する．**D** の第１項は状態遷移の情報 **E** の元 $e$，第２項はデータフロー情報 $d$（状態遷移情報を含まないブロック線図）である．

1　状態遷移の情報 **E** から取り出せる元があるか？
- True → $e$ という名前で取り出し，2 へ．
- False → 終了

2　共有変数ブロックの作成
BLp に含まれる２つのブロック $B_i$, $B_j$ について，ブロック $B_i$ が $e$ の第一項 $n$ の持つ状態値 $s$ の $T$ の時刻に計算しないブロックで，かつ，そのブロックの出力先のブロック $B_j$ が同じ時刻で計算する場合，共有変数ブロック $MB_{ij}$ を作成する．

3　状態遷移情報の状態と遷移によるブロック線図の縮約
BLp に含まれるブロック B が e の第１項 $n$ と e の第２項 $\boldsymbol{v}$ の値を利用して出力値を計算できる場合，B を定数値ブロックに置換する

4　サンプル時間によるブロック線図の縮約
$e$ の第一項 $n$ の持つ状態値 $s$ の $T$ の時刻に動作しないブロック B に入力する信号線を逆方向に辿りブロックを消去していく．ただし，B が他のブロックの入力になっている場合は信号線のみ消去する．ブロック B を対応する共有変数ブロック置換する．

5　上記の処理を経たブロック線図 BLp を $d_i$ として，**D** = **D** ∪ {$d_i$}を計算して 1 へ

## 4. 適用事例

ここでは，サンプルとして二つのモデルを採り上げ，本手法を適用した結果を述べる．

### 4.1 充電池制御モデル

ここで取り扱うモデルは、充電池の充電方式を制御するための制御器モデルであり、次のような仕様であるものとする（表 1）．

表 1：充電池制御の仕様

| |
|---|
| **モード種別** |
| 無充電モード，電圧充電モード，電流充電モードの３つの制御モードを持つ． |
| **無充電モード時** |
| 充電電圧に 0 を設定する |
| コンセント接続状態が切断から接続へ変化した時<br>→電圧充電モードに遷移する． |
| **電圧充電モード時** |
| モード制御に利用する電圧値を元に計算される電圧値を充電電圧に設定する． |
| 電圧値が 1000V 以上となった時<br>→電流充電モードに遷移する． |
| **電流充電モード時** |
| モード制御に利用する電流値を元に計算される電圧値を充電電圧に設定する． |
| 電流値が 330A 以下となった時<br>→無充電モードに遷移する． |

この仕様に従って作成されたブロック線図モデルが図 4 である．ブロック数が 24 個，信号線数が 34 本のシステムである．

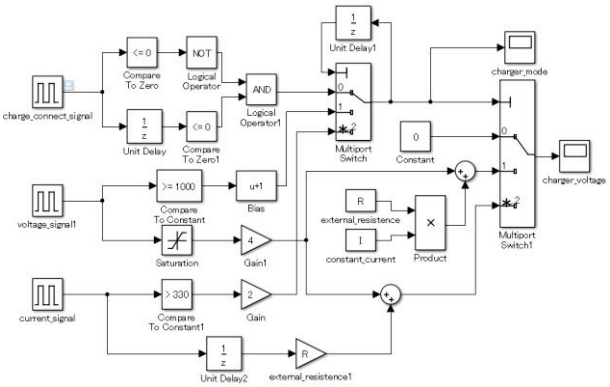


図 4：充電池制御モデル

以下では、このブロック線図に対して本手法を適用し，再構築されたモデルについて説明する．システムレイヤーは，６入力，２出力のシステムとなった（図 5）．６つの入力は以下のとおり：

- 現在のコンセント接続状態：システムの計算を行うときのコンセントの接続／切断の状態．
- 前のコンセント接続状態：システムが前回計算を行ったときのコンセント接続／切断の状態．
- 電圧充電終了判定：電圧充電モードを終了するかどうかの判定値．
- モード制御用電圧値：モード制御の判定に利用した電圧値．出力となる充電電圧の計算にも利用される．
- 電流充電完了判定：電流充電モードを終了するかどうかの判定値．
- モード制御用電流値：モード制御の判定に利用した電流値．出力となる充電電圧の計算値も利用される．

２つの出力値は以下のとおり：

- 現在のモード：現在の充電制御モード
- 充電電圧：充電池の充電を行う充電電圧値

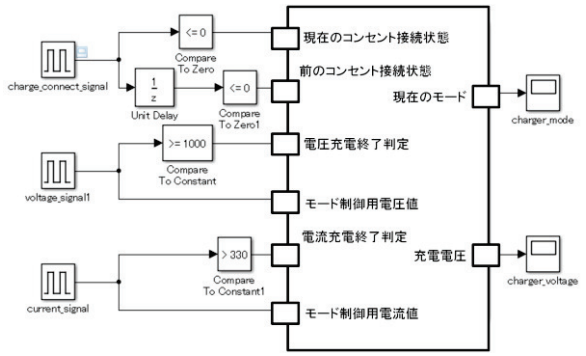


図 5：充電池制御モデル（システムレイヤー）

状態遷移レイヤーは 3 つの状態と 5 つの遷移を持つ状態遷移となった（図 6）．ここでは，仕様で記述されていたモードが過不足なく抽出されている．ただし，仕様には制御モードの初期値が記述されておらず，作成されたブロック線図から状態遷移の情報を抽出したときに，状態遷移の初期状態が二通り存在することが明らかとなった．

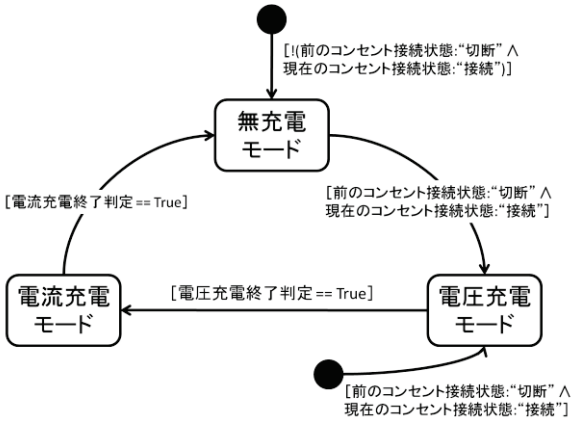


図 6：充電池制御モデル（状態遷移レイヤー）

データフローレイヤーは状態遷移レイヤーに存在した３つの状態に対して，それぞれ異なるデータフローとなった（図 7）．各データフローは以下のとおり：

- 無充電モードのデータフロー：充電電圧に 0 を出力する．
- 電圧充電モードのデータフロー：充電電圧にモード制御用電圧値から計算された値を出力する．
- 電流充電モードのデータフロー：充電電圧がモード制御用電圧値とモード制御用電流値から計算されることが分かる．

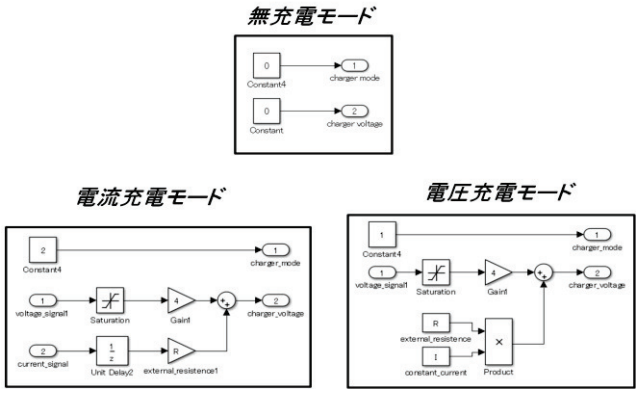


図 7：充電池制御モデル（データフローレイヤー）

### 4.2 エアコン制御モデル

ここで取り扱うモデルは、エアコンの冷温風切り替えを自動制御する制御器のモデルであり、以下のような仕様であるものとする（表 2）．

表 2：エアコン制御の仕様

| 制御モードが停止状態の時 |
|---|
| 室温と湿度への影響はない |
| 室温が冷房温度（冷房時のの目標温度）以上<br>→制御モードが冷房状態に遷移する． |

| |
|---|
| 室温が暖房温度（暖房時の目標温度）以下<br>→制御モードが暖房状態に遷移する． |
| 室温が適正温度（冷房温度－遊び値以上，暖房温度＋遊び値以下）<br>AND 湿度が除湿湿度（除湿時の目標湿度）<br>→制御モードが除湿状態に遷移する． |
| **制御モードが冷房状態の時** |
| 室温を低下させるための信号を出力する．<br>湿度への影響はない． |
| 室温が冷房温度－遊び値以下<br>AND 湿度が除湿湿度＋遊び値以下<br>→制御モードが停止状態に遷移する． |
| **制御モードが除湿状態の時** |
| 湿度を低下させるための信号を出力する．<br>室温への影響はない． |
| 室温が冷房温度以上<br>→制御モードが冷房状態に遷移する |
| 室温が冷房温度－遊び値以下<br>AND 湿度が除湿湿度＋遊び値以下<br>→制御モードが停止状態に遷移する． |
| **制御モードが暖房状態の時** |
| 室温を上昇させるための信号を出力する．<br>湿度への影響はない． |
| 室温が暖房温度＋遊び値以上<br>→制御モードが停止状態に遷移する． |
| 室温が暖房温度以下<br>AND 湿度が加湿湿度（加湿時の目標湿度）<br>→制御モードが加湿暖房状態に遷移する． |
| **制御モードが加湿暖房状態の時** |
| 室温と湿度を上昇させるための信号を出力する． |
| 湿度が上限湿度＋遊び値以上<br>→制御モードが暖房状態に遷移する |

この仕様に従って作成されたブロック線図モデルが図 8 である．このブロック線図はブロック数が 100 個，信号線数が 180 本のシステムである．なお，室温と湿度を変化させるための出力データフローは簡易化して取り扱う（各値を上昇させる場合は 1 を出力，下降させる場合は－1 を出力，変化させない場合は 0）．

以下では、このブロック線図に対して本手法を適用し，再構築されたモデルについて説明する．システムレイヤーは，１０入力，２出力のシステムとなった（図 9）．１０個の入力は以下の式の真偽値となる．なお、式中の「遊び値」は、目標室温や目標湿度の値付近で状態遷移が繰り返されることを防ぐためのマージンの値を表す．

- 室温＞冷房温度＋遊び値
- 室温＜冷房温度
- 適正範囲上限判定（室温を目標温度帯域に収めるための判定．目標温度帯域の高温側の閾値）
- 適正範囲下限判定（室温を目標温度帯域に収めるための判定．目標温度帯域の低温側の閾値）
- 室温＜暖房温度＋遊び値
- 室温＞暖房温度
- 湿度＞下限湿度＋遊び値
- 湿度＜下限湿度
- 湿度＜上限湿度＋遊び値
- 湿度＞上限湿度

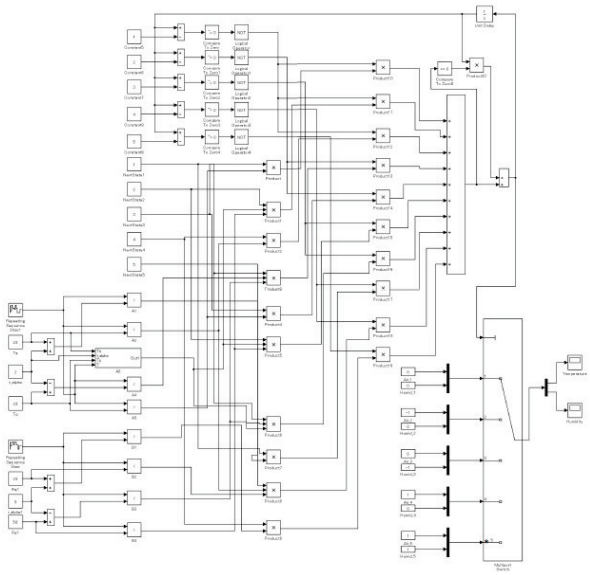

図 8：エアコン制御モデル

２つの出力値は以下のとおり：

- 室温制御信号：室温をコントロールするため，暖房の稼働（1），冷房の稼働（-1），停止（0）のいずれかの値を出力する．
- 湿度制御信号：湿度をコントロールするため，加湿の稼働（1），除湿の稼働（-1），停止（0）のいずれかの値を出力する．

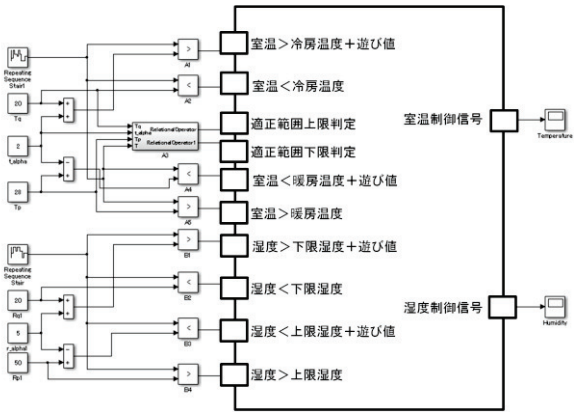


図 9：エアコン制御モデル（システムレイヤー）

状態遷移レイヤーは 8 つの状態と 24 の遷移を持つ状態遷移

となる（図 10）．仕様よりも状態が３つ多く抽出されており，また，起こり得ない遷移も抽出されている．しかし，各ガード条件の排他性を考慮すると１０個の遷移は起こり得ない遷移であることが判明し，８つのうち３つの余分な状態には状態遷移で到達することはない事が判明した（起こり得ない遷移および到達しない状態は図 9 において赤く記している）．

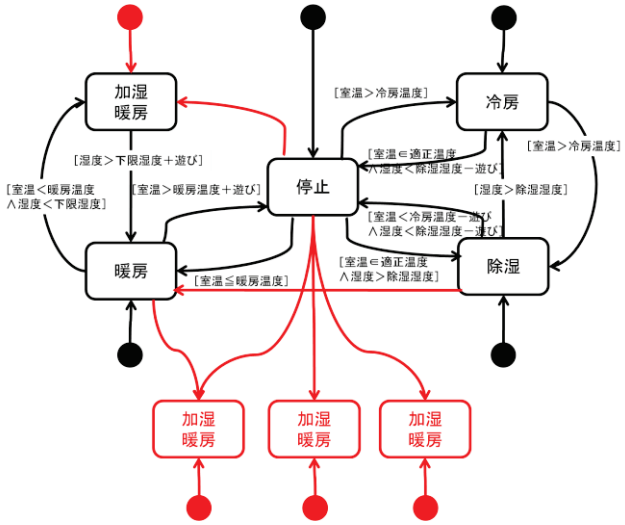


図 10：エアコン制御モデル（状態遷移レイヤー）

データフローレイヤーは上記の仕様で記述したとおり，各制御モードが出力すべきデータフローの情報を抽出できた（図 11）．各データフローは以下のとおり：

- 停止状態のデータフロー：室温制御信号の出力が停止（0），湿度制御信号の出力が停止（0）．
- 冷房状態のデータフロー：室温制御信号の出力が冷房の稼働（-1），湿度制御信号の出力が停止（0）．
- 除湿状態のデータフロー：室温制御信号の出力が停止（0），湿度制御信号の出力が除湿の稼働（-1）．
- 暖房状態のデータフロー：室温制御信号の出力が暖房の稼働（1），湿度制御信号の出力が停止（0）．
- 加湿暖房状態のデータフロー：室温制御信号の出力が暖房の稼働（1），湿度制御信号の出力が加湿の稼働（1）．

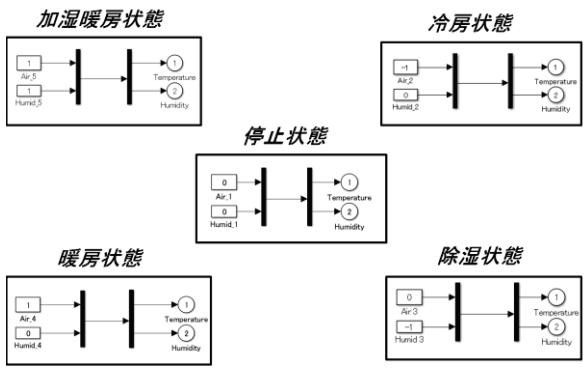


図 11：エアコン制御モデル（データフローレイヤー）

## 5. 評価と考察

本手法のサンプルに対する適用結果の評価と考察を行う．まず，本手法の評価として，手法の適用前後での各保守性指標値を比較する（表 3）．システムとしてのブロック数と信号線数はエアコン制御モデルでは双方とも約 50%に減少しているが，充電池制御モデルではブロック数が増加，信号線は変化しなかった．この原因は，ブロック線図モデルの階層化では，階層間を繋ぐために用いられるポートブロックを用いるためだと考えられる．ポートブロックは階層化前に１本の信号線で済んでいた情報に１つのブロックを付け加えてしまう．このためブロック数が増加したと思われる．次に，１ブロック線図あたりのブロック数（平均ブロック数）と信号線数（平均信号線数）は充電池制御モデルでは約 30%，エアコン制御モデルでは，10%以下になっている．特に，平均ブロック数は双方で１ブロック線図あたり約 9 個となっており，データフロー図の複雑性を抑える指針である 7±2 理論が示す値の範囲内にある[8]．信号線の交錯数に関しては，エアコン制御モデルで著しく減少している．しかし，このことは配線が容易になったことが理由ではなく，配線が複雑な部分が状態遷移に変換されたために起きていると考えられる．レイアウト面積についても同様の理由により減少していると考えられる．抽出された状態遷移の情報には，仕様で定義された状態と遷移が全て含まれており，本手法により，抽出された状態遷移の情報には，仕様で定義された状態と遷移が全て含まれている．このように本手法はブロック線図が含む状態遷移の情報を抽出し，ブロック線図を再構築し，保守性を向上させるために有効であると考えられる．

表 3：2 つの適用事例の指標値の手法適用前後の変化

| 保守性指標 | 充電池制御モデル | | エアコン制御モデル | |
|---|---|---|---|---|
| | 適用前 | 適用後 | 適用前 | 適用後 |
| ブロック数 | 24 | 35 | 100 | 54 |
| 平均ブロック数 | 24 | 8.75 | 100 | 9 |
| 信号線数 | 34 | 34 | 180 | 91 |
| 平均信号線数 | 34 | 8.5 | 180 | 15.2 |
| 信号線交錯数 | 1 | 0 | 205 | 15 |
| 平均信号線交錯数 | 1 | 0 | 205 | 2.5 |
| レイアウト面積 | 177520 | 221620 | 1351635 | 328925 |
| 平均レイアウト面積 | 177520 | 55405 | 1351635 | 54821 |
| 状態数 | 0 | 3 | 0 | 8 |
| 遷移数 | 0 | 5 | 0 | 24 |

一方で，本手法では状態遷移抽出時に仕様で定義されていない状態が抽出されてしまうことがあった．エアコン制御モデルから抽出された状態遷移情報には，仕様で定義している状態よりも多くの状態が含まれていた．これらの状態について，そこに至る遷移のガード条件を分析すると，１つのガード条件を構成する複数の条件式の中に排他的な条件が含まれていた．つまり，絶対に起こらない遷移がモ

デルから抽出されていることが分かった．

## 6. 関連研究

ブロック線図モデルの保守性を取り扱った研究では，ISO/IEC9126 を基にして Simulink のブロック線図モデルの保守性に関する品質評価手法が提案されている[4]．ブロック線図モデルにおいて，コピー＆ペーストによるクローンと呼ばれる重複箇所が増加することに起因する保守性の低下に対して，そのクローン情報を抽出する手法がある[7]．モデルの変換という観点では，クラスタリング手法を利用して Simulink モデルを UML モデルに変換する手法[6]やハイブリッドオートマトンに変換する手法[5]が提案されている．これらに対して，本研究では，Simulink のブロック線図モデルが陰に含む状態遷移情報を抽出し，元のブロック線図モデルを状態遷移モデルとデータフローモデルに分離し階層構造化する．これにより，元のモデル記述と比較して保守性指標が改善され，モデルの保守性が高まることが期待される．

## 7. おわりに

本研究では，ブロック線図に状態遷移の情報が含まれることで保守性が低下する課題を解決するために，ブロック線図から状態遷移とデータフローの情報を抽出する手法を提案した．これにより，状態遷移の情報を含むことで保守性が低下したブロック線図を，状態遷移，データフローの情報が抽出・分離されたブロック線図へと再構築でき，ブロック線図の保守性を向上することが可能である．

今後の課題として，不要な状態遷移情報が抽出されることがあり，その分析作業が余分に必要となる場合がある．これに対しては，抽出された状態遷移を最適化する手法の開発により対応していく予定である．

## 参考文献

1) ソフトウェア産業の実態把握に関する調査 調査報告書 -速報版-, 情報処理推進機構 (2012)
2) 組込みシステムの先端的モデルベース開発実態調査 調査報告書, 情報処理推進機構 (2012)
3) Simulink, http://www.mathworks.co.jp/products/simulink/
4) W. Hu. T. Loeffler, and J. Wegener: Quality model based on ISO/IEC 9126 for internal quality of MATLAB/Simulink/Stateflow models, IEEE, ICIT, Athens, Greece, pp.19-21 (2012)
5) A. Agrawal, G. Simon, and G. Karsai: Semantic translation of Simulink/Stateflow models to hybrid automata using graph transformations, Electron. Notes Theor. Comput. Sci., Vol.109, pp.43-56 (2004).
6) 小澤貴之, 鷲崎弘宜, 深澤良彰: Simulink モデルの保守性向上に向けたクラスタリングおよび UML モデルとの双方向変換, 信学技報, vol. 112, no. 373, SS2012-54, pp. 49-53 (2013).
7) 鷲崎弘宜, 村上真一, 深澤良彰: Simulink モデルにおけるグラフに基づく非完全一致モデルクローン検出, 信学技報, vol. 112, no. 164, SS2012-11, pp. 7-12 (2012).
8) G. A. Miller: The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information, Psycol. Rev., Vol. 63(1956), pp. 81-97