組込みシステム向け診断機能のモデル段階での組込みと部品化

中野 真吾[†] 渋田 達也[†] 新井 正敏[†,††]
松本 倫子[†] 吉田 紀彦[†]

組込みシステムの動作診断のために，モデルベース設計と連携させて，モデル段階から診断機能を組み込む手法を提案する．これにより，ハードウェア・ソフトウェア未分化の段階，各種バリエーション派生以前の段階から，診断機能の組込みを可能にする．そのため，まず，ハードウェア・ソフトウェアのどちらでも実装できる単純な監視機能を JTAG に基づいて実現する．また，多くのコンポーネントによって構成される組込みシステムでは，あるコンポーネントの障害が他に波及することから，診断がより困難になるが，人工知能でいうモデルベース診断に基づく沓名らの枠組みを取り込むことで対応する．本研究では，モデル記述には MATLAB/Simulink を用いる．そして，診断機能の組込みにプログラミング言語論の成果であるアスペクトを用いることで，診断機能の部品化と再利用を可能にしている．

Reusable Modeling of Diagnosis Functions for Embedded Systems

Shingo Nakano,[†] Tatsuya Shibuta,[†] Masatoshi Arai,[†,††]
Noriko Matsumoto[†] and Norihiko Yoshida[†]

This paper presents a technique to embed diagnosis functions in model-based design. The technique enables designers to embed such functions at early design stages before separation of hardware and software implementations and derivation of several variations. First, we develop some simple monitoring functions suitable for both HW and SW based on JTAG. Then, to identify faulty components in a complex embedded system where a fault in a component affects others, we employ a method proposed by Kutsuma et al., which is based on "model-based diagnosis" studied in Artificial Intelligence. This paper uses MATLAB/Simulink as a modeling framework, and employs Aspects, which is studied in Programming Methodorogies, for diagnosis description to promote reuse of diagnosis function models.

1. はじめに

近年，組込みシステムは大規模化・複雑化・分散化が進んだことにより多くのハードウェア（以下 HW）とソフトウェア（以下 SW）が協調的に動作している．システムの障害発生時には，多数の構成要素が関連付けられることから根本原因の特定が難しくなった．したがって，システム側で障害を検知し，根本原因を特定する診断機能が実現できれば，原因特定の労力・コストの低下，平均修理時間の短縮によるシステムの稼働率の上昇が期待できる．

組込みシステムの診断についての研究は行われており，様々な手法が提案されている．また，組込みシステムのモデルベース開発に合わせて，モデル段階から診断機能を組み込んでいくことで，効率的な開発を行えることが知られている．しかし，実際の組込みシステム開発において，これらの技術をどのように組み合わせて，適用していくかについては考慮されていない．

そこで，本研究では，既存の技術を組み合わせることで，異常箇所を特定する診断機能を実現し，診断機能をモデル段階から組み込む手法を提案する．診断機能については，HW での実装がしやすく，消費リソースの少ない方法をとることによって，組込みシステムに適した機能としての実現を目指す．本研究では，モデルの記述には MATLAB/Simulink を用いる．

診断機能の実現には，まず，診断に必要となるコンポーネントの入出力を取得するため，文献 1) で提案された組込みシステムの IC チップの入出力を監視する手法を参考に，コンポーネントの入出力の取得機能を HW/SW の区別のないモデルの段階で記述する．そして，文献 2) で提案された異常箇所を特定する診断手法を実装し，モデル化を行う．

† 埼玉大学
Saitama University
†† カルソニックカンセイ株式会社
Calsonic Kansei Corp.

また，プログラミング言語論の成果であるアスペクトを用いることで，モデル段階から診断機能を組み込む．アスペクトを用いることによって，部品化した機能の追加が容易となり，効率的な設計やバリエーションへの対応が可能となる．

以下，第 2 章では，コンポーネントの入出力の取得方法について考案し，MATLAB/Siumulink を用いたモデル化について述べる．第 3 章では，文献 2) で提案された診断手法について紹介し，その診断手法を MATLAB/Siumulink を用いてモデル化する方法について説明する．第 4 章では，診断機能の部品化と再利用を可能にするため，アスペクトを用いて診断機能を診断対象のモデルに組み込む方法について述べる．第 5 章では，実験として，簡単なモデルで，提案する診断機能のシミュレーションを行うことで，本研究の実現性を確認する．最後に，第 6 章では，本研究のまとめと今後の課題について述べる．

2. コンポーネントの入出力の取得

2.1 方　　針

診断には，組込みシステムの構成要素として各機能を実現するコンポーネントの入出力を用いる．本研究では，IC チップの品質検査方式標準である JTAG[3] を抽象化して用いることで，入出力の取得を可能にする．

2.2 JTAG

関連研究として，組込みシステムの誤動作の防止のために，IC チップの品質検査方式標準である JTAG を用いて，組込みシステムの入出力を監視するという手法が提案されている[1]．JTAG によって，IC チップ内部の各入出力ピンに対応したセルに外部から入出力を行うことが可能となる．JTAG のテストを行うためのコントローラは TAP（Test Access Port）コントローラとして標準化されており，最低でも以下の 4 つのシリアルインタフェースを持つ．

- TCK（Test Clock）
- TMS（Test Mode Select）
- TDI（Test Data In）
- TDO（Test Data Out）

JTAG に対応した IC チップの構造は**図 1** のようになっている．IC の入出力ピンと内部ロジックの間にセルを置き，通過した入出力の値を保存させる．セルは数珠つなぎのシフトレジスタとなっているので，TDI から入力，TDO へ出力が可能となっている．

2.3 JTAG の抽象化

JTAG は HW で実装できるため，複雑な計算を必要としない簡便な手法であり，機能やリソースに制限のある組込みシステムでも実現可能な手法である．しかし，JTAG は HW で実装されるため，HW で行われる入出力のみしか取得できない．本研究では，HW/SW に依存せず，診断を行えるようにするため，診断に必要となる入出力の取得を HW/SW の区別のないモデルの段階で記述する．そのため，以下のように JTAG を抽象化する．

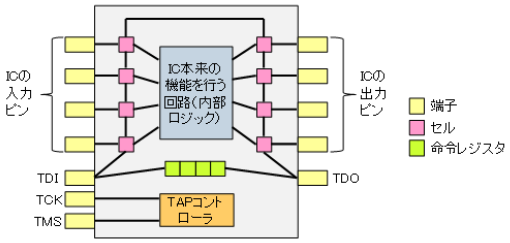

図 1　JTAG に対応した IC チップの構造
Fig. 1　IC chip construction based on JTAG

- IC の入出力にセルを介さない．
- 命令と命令レジスタを必要としない．
- TAP コントローラのかわりに，TMS の値のみで制御．
- 1 ビットずつではなく，入出力の値を 1 つずつ取得．
- TDI の値は入出力の値のあとにシフトされるのみ．

まず，診断には，IC の入出力を取得できればよく，変更を行う必要はないので，IC の入出力にセルを介したり，行う動作を決定する命令や命令レジスタを用いる必要がない．

また，JTAG では動作を制御するため，多くの状態を用いて様々な制御を行う TAP コントローラを必要とするが，入出力の取得には，次節に示す 2 つの状態が実現できればよいので，TAP コントローラは必要なく，TMS の値のみで状態を切り替える．

JTAG では，値の保存にシフトレジスタを用いているので，TCK が 1 クロックの間に 1 ビットの値しか取得できないが，データ型に関係なく入出力の値を 1 つずつ取得できるように抽象化する．

TDI への入力値は，入出力の変更や命令のセットなどにも用いるが，入出力取得のために抽象化すると，TDI への入力値はシフトをセルの個数より多く行った時に TDO から出力されるだけの値で，入出力取得の処理には影響しない．

2.4 抽象化した JTAG による入出力の取得

抽象化した JTAG では，TMS の値によって切り替わる以下の 2 状態を用いて入出力の取得を行う．

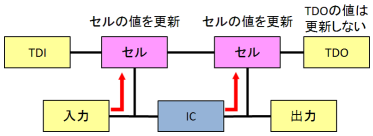

図 2 入出力を保存（TMS=0，TCK の立ち上がり）
Fig. 2 Store Input/Output（TMS=0，rise TCK）

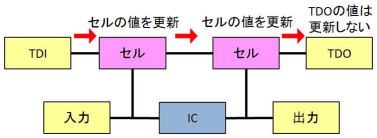

図 3 各セルに保存されていた値をシフト（TMS=1，TCK の立ち上がり）
Fig. 3 Shift value stored in cell（TMS=1，rise TCK）

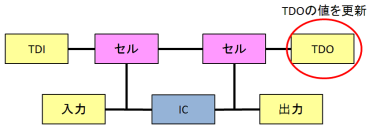

図 4 最後のセルから送られていた値を TDO から出力（TMS=1，TCK の立ち下がり）
Fig. 4 Output value from last cell to TDO（TMS=1，fall TCK）

- 診断対象コンポーネントの入出力を保存する状態（TMS=0）
- 保存した値をシフトさせる状態（TMS=1）

それぞれの状態での動作を**図 2**～ **図 4** に示す．

TMS によって切り替えられる 2 つの状態を用いて，入出力の取得を以下のように行う．

step1. TMS が 0 の状態で TCK を立ち上げて，セルに入出力の値を保存．

step2. TMS が 0 の状態で TCK を立ち下げて，TCK を LOW にする（処理は行わない）

step3. TMS が 1 の状態で TCK を立ち上げて，セルに保存した値をシフト．

step4. TMS が 1 の状態で TCK を立ち下げて，step3 で最後のセルから受け取った入出力の値で TDO の値を更新．(step1 で保存した入出力の値を 1 つ取得)

step5. step3，step4 をコンポーネントの入出力の数だけ繰り返す．

2.5 MATLAB/Simulink によるモデル化

本研究では，組込みシステム設計に広く用いられている MATLAB/Simulink によって，抽象化した JTAG を以下のようにモデル化する．

1 ステップだけ入力を保存して出力する Memory ブロックを用いて，セルへの値の保存を実現する．また，2 つのデータ入力のうち，どちらを出力するかを制御入力によって切り替える Switch ブロックを用いて，TMS の値による状態変化を実現し，外部信号によってその実行をトリガーする Trigger ブロックを用いて，TCK のクロックによる同期を行う．

このモデルから，HW 実装を行う場合，JTAG またはそれに近い形になる．一方，SW 実装を行う場合，メモリの動的な取得は必要なく，実装を行う際に，入出力の数によって定められる一定量のメモリを用いての代入演算の繰返しと制御文による制御を行う．

3. モデルベース診断による異常箇所の特定

3.1 方　　針

組込みシステムにおいて，1 つのコンポーネントが故障して異常な出力を行った場合，その異常な出力を入力として受け取ったコンポーネントも異常な出力を行ってしまい，外部からは複数のコンポーネントが故障しているように見えてしまう．この問題を異常伝播問題と言う．異常伝播問題によって，大規模化したシステムで異常箇所を特定するのは困難となっている．

この問題を解決するため，本研究では，どのコンポーネントが異常となっているかを抽象モデルベース診断[2] によって特定する．

3.2 抽象モデルベース診断

モデルベース診断とは，システム診断のためのフレームワークで，各コンポーネントの振る舞いを定義して，システムの構成から導かれる論理的な関係とシステム内を流れたデータの観測結果から各コンポーネントが正常か異常かを求める．そのために，以下の式を用いる．

SD（System Description） システムの構成から導かれる論理的な関係を示す式．

OBS（Observations） システム内を流れるデータの観測結果を表す式．

DIAG（Diagnosis） システム内の各コンポーネントが正常か異常かを表した式．

通常のモデルベース診断では，コンポーネントの振る舞いを記述する必要があるが，ソフトウェアの振る舞いを正確に書き下すことは難しい．この問題に対処するため，抽象化したモデルを用いた抽象モデルベース診断が提案されている[2]．抽象モデルベース診断では，「コンポーネントが正常で入力が正常ならば出力は正常」という関係を用いてシステムの構成から導かれる論理的な関係を求める．コンポーネントの振る舞いを書き下す必要がないため，ソフトウェアを含んだコ

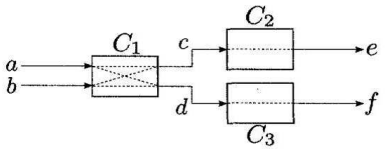

図 5 抽象モデルの例
Fig. 5 Example of abstract model

ンポーネントも容易に扱うことができる．

例として，文献 2) で例題に用いられている**図 5** のシステムに対して，抽象モデルベース診断を行う手順を示す．まず，SD を以下のように記述する．

$$\begin{aligned}\mathbf{SD} \equiv \{&ok(C_1) \wedge ok(a) \wedge ok(b)\\ &\rightarrow ok(c) \wedge ok(d)\}\\ \wedge\{&ok(C_2) \wedge ok(c)\\ &\rightarrow ok(e)\}\\ \wedge\{&ok(C_3) \wedge ok(d)\\ &\rightarrow ok(f)\}\end{aligned} \quad (1)$$

(1) 式の 1，2 行目は，コンポーネント C_1 が正常で，かつ，その入力 a, b が正常ならば，その出力 c, d は正常という関係を表している．

抽象モデルを用いる場合，OBS として，各データに対して何らかの基準で正常か異常かを判断した結果を用いる．例えば，図 5 において c, e は異常，その他のデータは正常と判断した場合，OBS は次のように記述する．

$$\begin{aligned}\mathbf{OBS} \equiv &ok(a) \wedge ok(b) \wedge \neg ok(c)\\ \wedge&ok(d) \wedge \neg ok(e) \wedge ok(f)\end{aligned} \quad (2)$$

データが正常か異常かを判断するための方法として，次の二つが考えられる．

a) 設計者の知識に基づく方法

データが取り得る範囲を規定し，その範囲を外れた場合に異常とする方法や，データの発生周期を規定し，規定周期で発生しない場合に異常とする方法．

b) 知識以外に基づく方法

蓄積データから，統計的学習やデータマイニング手法などで正常か異常かを判断する方法や，コンポーネントの中身を別途モデル化し，モデル検査などの形式手法を用いて正常か異常かの判断基準を求める方法など．

DIAG は，各コンポーネントが正常か異常かを示したものであり，例えばコンポーネント C_1, C_3 が異常で，コンポーネント C_2 が正常の場合，DIAG を次のように記述する．

$$\mathbf{DIAG} \equiv \neg ok(C_1) \wedge ok(C_2) \wedge \neg ok(C_3) \quad (3)$$

$\neg ok(C)$ はコンポーネント C が異常であることを意味する．また，以下では，異常なコンポーネントを括弧で括って DIAG を表現する．例えば，(3) 式の DIAG を $\{C_1, C_3\}$ と表わす．

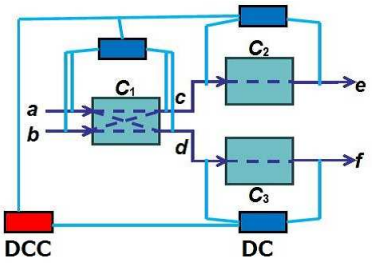

図 6 診断システムの配置
Fig. 6 Placement of diagnosis system

モデルベース診断では，SD と OBS から，適切な DIAG を求める．具体的には，「どのように DIAG を仮定すれば，SD と OBS の間に矛盾が生じないか」という考え方に基づき DIAG を求める．つまり，次式を満たすような DIAG を求める．

$$\mathbf{SD} \wedge \mathbf{OBS} \wedge \mathbf{DIAG} = \mathbf{True} \quad (4)$$

SD が (1) 式，OBS が (2) 式で与えられる場合，DIAG は $\{C_1\}$，$\{C_1, C_2\}$，$\{C_1, C_3\}$，$\{C_1, C_2, C_3\}$ となる．DIAG のうち，$\{C_1\}$ のように，異常コンポーネントを 1 つでも減らすと (4) 式が不成立になるものを Minimal Diagnosis と呼ぶ．

上記の例のように，モデルベース診断では，1 組の SD と OBS に対して，複数の DIAG が求まることがある．このような場合，複数のコンポーネントが同時に異常になる確率は小さいと考え，異常コンポーネントが少ない DIAG を診断結果として出力する．したがって，この例では，C_1 が異常という診断結果を出力する．

3.3 MATLAB/Simulink によるモデル化

HW/SW に依存せずに診断を行えるようにするため，2.5 節で述べた入出力の取得のモデル化と同様に，抽象モデルベース診断による診断システムを，特に OBS の扱いについて MATLAB/Simulink でモデル化する方法について述べていく．

3.3.1 診断システムの配置

例として，図 5 のモデルを対象に診断システムを配置すると，**図 6** のようになる．各コンポーネントの入出力を 2 章の抽象化した JTAG から受け取り，正常異常を判断するコンポーネントを DC（Diagnosis Component），DC からの結果をもとに，OBS を出力するコンポーネントを DCC（Diagnosis Component Center）と名付ける．

$$\begin{array}{cccccccccc} C & \text{入力 1} & \cdots & \text{入力 } m & \text{出力 1} & \cdots & \text{出力 } n & \text{入力と出力の境目} & \text{判断基準} \\ \left(\begin{array}{c} x \\ \vdots \\ x \end{array} \right. & \begin{array}{c} a_1 \\ \vdots \\ c_1 \end{array} & \begin{array}{c} \cdots \\ \ddots \\ \cdots \end{array} & \begin{array}{c} a_m \\ \vdots \\ c_m \end{array} & \begin{array}{c} b_1 \\ \vdots \\ d_1 \end{array} & \begin{array}{c} \cdots \\ \ddots \\ \cdots \end{array} & \begin{array}{c} b_n \\ \vdots \\ d_n \end{array} & \begin{array}{c} m+1 \\ \vdots \\ m+1 \end{array} & \left. \begin{array}{c} y \\ \vdots \\ y \end{array} \right) \end{array}$$

図 7 正常データ行列の記述法
Fig. 7 Notation of normal data matrix

3.3.2 データの正常異常の判断基準

3.2 節で述べたように，抽象モデルベース診断において，データの正常異常の判断基準は，設計者の知識に基づく方法と知識以外に基づく方法の 2 種類がある．本研究では，設計者の知識に基づく方法を扱い，以下の 2 つの判断基準を用いる．

1. 入力と出力の組み合わせが一意に定まる
2. 入力と出力が一定範囲内に収まればよい場合

各コンポーネントの入出力が正常か異常かを判断するための情報について，文献 2) では記述形式が定義されていないが，本研究では行列で記述するものとした．この行列を正常データ行列と名付け，DC において，各コンポーネントの入出力の正常異常判断に用いる．

正常データ行列は**図 7** のように記述する．C の列には対応するコンポーネントの番号を記述し，入力 $1,\cdots,$入力 m の列には入力が満たすべき条件を記述する．出力 $1,\cdots,$出力 n の列には出力が満たすべき条件を記述し，最後から 2 列目には入力データと出力データの境目の位置を，最後の列には判断基準の番号を記述する．

現在は，上で述べたように 2 つの判断基準のみを用いているが，今後，判断基準が増える場合にも対応できるように，判断基準を示す値を正常データ行列に含めることで，拡張性を持たせている．

DC は対応する判断基準によって，以下のように，入出力の正常異常を判断する．

1) 判断基準が 1 の場合

step1. 診断対象コンポーネントの正常な入出力の組合せを 1 行ごとに記述した正常データ行列を取得.
step2. 診断対象コンポーネントの入出力の観測値と正常データ行列に記述された正常な入出力の組合せを比較して，一致する値の個数が最も多い入出力の組合せを正常データ行列から 1 つ選択.
step3. 観測値と選択した入出力の組合せで，一致した入出力は正常，異なる入出力は異常と判断する.

2) 判断基準が 2 の場合

step1. 診断対象コンポーネントの各入出力が取り得る値の最小値を 1 行目に，最大値を 2 行目に記述した正常データ行列を取得.
step2. 診断対象コンポーネントの入出力の観測値が範囲内に収まっている入出力は正常，収まっていない入出力は異常と判断する.

3.3.3 正常データ行列の記述例

具体例として，図 6 のモデルを用いて，正常データ行列を記述する．

まず，各コンポーネントの入出力が正常か異常かの判断を，上で挙げた 2 つの判断基準を用いて，以下のように行うと定めたとする．

C_1**:入力 a** $-1 \leq a \leq 1$ ならば正常.（判断基準 2）
C_1**:入力 b** $-2 \leq b \leq 2$ ならば正常.（判断基準 2）
C_1**:出力 c** $0 \leq c \leq 1$ ならば正常.（判断基準 2）
C_1**:出力 d** $0 \leq d \leq 1$ ならば正常.（判断基準 2）
C_2**:入力 c，出力 e** (c,e) の値が (0,0)，(0,1)，(1,0)，(1,1) のどれかならば正常.（判断基準 1）
C_3**:入力 d，出力 f** (d,f) の値が (0,−1)，(0,1)，(1,−1)，(1,−1) のどれかならば正常.（判断基準 1）

このときの C_1，C_2，C_3 の正常データ行列は以下のように記述される．

C_1 **の正常データ行列**

$$\left(\begin{array}{ccccccc} 1 & -1 & -2 & 0 & 0 & 3 & 2 \\ 1 & 1 & 2 & 1 & 1 & 3 & 2 \end{array} \right)$$

C_2 **の正常データ行列**

$$\left(\begin{array}{ccccc} 2 & 0 & 0 & 2 & 1 \\ 2 & 0 & 1 & 2 & 1 \\ 2 & 1 & 0 & 2 & 1 \\ 2 & 1 & 1 & 2 & 1 \end{array} \right)$$

C_3 **の正常データ行列**

$$\left(\begin{array}{ccccc} 3 & 0 & -1 & 2 & 1 \\ 3 & 0 & 1 & 2 & 1 \\ 3 & 1 & -1 & 2 & 1 \\ 3 & 1 & 1 & 2 & 1 \end{array} \right)$$

3.3.4 正常データ行列の記述内容の爆発

判断基準 2 では各入出力の最大値，最小値の条件を記述するだけでよいので正常データ行列は 2 行で記述できる．それに対して，判断基準 1 では正常なデー

タの組合せを全て書き下しておく必要があるため，コンポーネントの入出力数や取り得る値の組み合わせ数によって，記述する内容が爆発的に増加するという問題がある．この問題への対策としては，正常な入出力の組み合わせを記述するプログラムを作成する方法，判断基準 2 を用いるという方法が挙げられる．これらの対策を用いることができない状況に対応するためには，扱えるデータの正常異常の判断基準を追加していく必要がある．

3.3.5 DC と DCC の処理

正常データ行列を用いて，DC と DCC は以下のように処理を行っていく．

DC の処理

step1. 診断対象のコンポーネントの入出力の観測値を抽象化した JTAG から取得．正常データ行列を取得．

step2. 正常データ行列から判断型を読み取り，判断方法を選択．

step3. 選択された判断方法に従い，正常異常を判断．

step4. 各入出力が正常か異常かを DCC に伝達．

DCC の処理

step1. 全ての DC からの伝達情報を取得．

step2. 伝達情報から OBS を出力．

3.4 モデル化方法

モデル化する際は，MATLAB/Simulink のユーザ定義関数ブロックを用いる．このブロックの処理はユーザーがプログラムで書くことができるので，このブロックに DC と DCC の処理を記述することで，診断システムのモデル化を行う．

4. アスペクトによる機能の部品化と再利用

4.1 方　　針

本研究では，診断機能の部品化と再利用を可能にするため，アスペクト適用によるモデル変換によって診断機能を組み込む方法を考案する．アスペクトの適用は，対象モデルの完成後に行えるので，モデルの変更や部品化した機能の追加が容易となり，診断機能のみを外した実装などのバリエーションにも対応できる．

本研究では，このアスペクト適用によるモデル変換を，診断機能の組込みに特化させたものとしてではなく，一般的なモデルの変換にも用いることができるものとして実現する．

4.2 アスペクト適用によるモデル変換

アスペクトとは，複数のモデル，あるいはプログラム等に共通して現れる処理を分離し，部品化するために考案された技術である[4]．複数のモデルやプログラムに共通して現れる処理を横断的関心事と呼び，横断的関心事はモジュール化技術を用いて部品化することが困難である．アスペクトは，この横断的関心事を分離し，部品化することを可能にする．

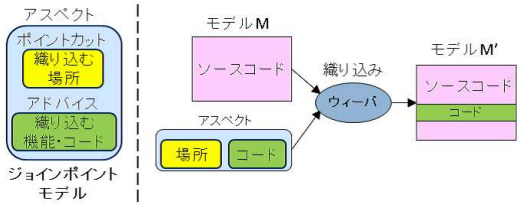

図 8 アスペクトによるモデル変換
Fig. 8 Model transformation by aspect

アスペクトの代表的なモデルであるジョインポイントモデルはジョインポイント，ポイントカット，アドバイスの 3 つから構成される．ジョインポイントはアスペクトを適用可能な「コード上の箇所」であり，1 つ以上のジョインポイントをまとめた定義がポイントカットである．アドバイスは実際に織り込むアスペクトの処理定義であり,「どのような処理を行うか」が記述される．アスペクトは，プログラムのコンパイル時，あるいは実行時に対象プログラムに対して適用される．ポイントカットで指定されたジョインポイントにアドバイスの機能を追加する．この作用のことを織り込みと呼び，また織り込みを行う処理系のことをウィーバと呼ぶ．

MATLAB/Simulink で作成したモデルは階層化された構造化文書のコードとして保存される．そこで，本研究では，アスペクト適用によってコードの内容を変更することで，モデル変換を行う．具体的には，ポイントカットとしてコードの特定の場所を指定し，アドバイスによりコードの内容を変更することでモデルの内部構造を変化させる（**図 8** 参照）．MATLAB/Simulink でモデル化した診断機能の組込みには，診断を行うために必要なブロック，ライン，ラインの分岐を追加するアスペクトを適用できればよい．ただし，現状は，適用に用いることができるウィーバが未完成なため，実際の適用は手動で行っている．

診断対象のモデルによって，コンポーネントの数，各コンポーネントの入出力の数や名前が異なるため，診断機能の組込み方法は，診断対象のモデルごとに異なり，設計は困難となる．そこで，診断機能を，どんな診断対象モデルにも同様に追加する共通部分と診断対象モデルごとに異なる部分に分け，共通部分については，まとめて部品化する．これによって，診断機能の共通部分を組み込む際には，追加する場所を変える

```
<aspect name="共通ブロック追加アスペクト">
  <pointcut name="ポイントカット1">
    診断対象のコンポーネントが配置されているシステムを指定(ジョインポイントの指定)
  </pointcut>
  <advice name="アドバイス1" Type="ブロックの追加">
    共通して用いるブロックの記述(アドバイスの内容)
  </advice>
</aspect>
```

図 9 アスペクト記述の概要
Fig. 9 Outline of aspect description

だけで，使いまわすことができ，効率的な設計が行えるようになる．

本研究では，MATLAB/simulink モデルを xml 記述するツールを利用することにして，アスペクトを xml 形式で記述する．記述の方法について説明するため，「共通ブロック追加アスペクト」というアスペクトの記述の概要を**図 9** に示す．「共通ブロック追加アスペクト」は上で説明した共通部分のブロックを追加するアスペクトである．図 9 のようにアスペクト記述には aspect というタグで全体を囲み，ポイントカット，アドバイスの記述についてはそれぞれ pointcut，advice というタグで囲む．ブロック，ラインを追加する際には，ジョインポイントとして追加するシステムを指定し，ラインの分岐を追加する際には，ジョインポイントとして分岐を追加するラインを指定する．また，アドバイスの動作は，Type によって，どのような処理を行うかを選択する．本研究の診断機能の組込みでは，ブロックの追加，ラインの追加，ラインの分岐の追加の 3 種類から Type を選択する．

アスペクトの例として，「共通ブロック追加アスペクト」を実際に記述した CommonBlockAdd-aspect のコードを**図 10** に示す．このアスペクトは，Wave というシステムに DCC という SubSystem ブロックを追加するというものになっている．

5. 実　　験

5.1 実験の内容

実験として，簡単なテストモデルを用いて，提案手法の実現性を確認する．テストモデルとして，**図 11** のような Wave.mdl を用いる．

Wave.mdl について説明する．Twice_Component は，振幅 1 の正弦波 a を入力として持つ．このコンポーネントを C_1 とする．C_1 の機能は，正弦波 a を 2 倍に増幅させて正弦波 b として出力するものである．Add_Component は，振幅 1 の正弦波 c と C_1 の出力結果の正弦波 b を入力として持つ．このコンポーネントを C_2 とする．C_2 の機能は，2 つの正弦波 b，c を加算して d を出力するものである．また，入出力を行う間隔である時間ステップは 0.1 とし，シミュレーションは 0.0 から 6.0 まで行う．

```
<aspect name="CommonBlockAdd-aspect ">
  <pointcut name="pointcut1">
    systemname="Wave"
  </pointcut>
  <advice name="advice1" Type="add-block">
    Block {
      BlockType   SubSystem
      Name    "DCC"
              :
    }
  </advice>
</aspect>
```

図 10 アスペクトの例
Fig. 10 Example of aspect

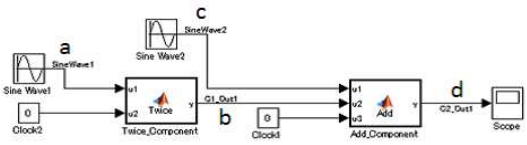

図 11 テストモデル:Wave.mdl
Fig. 11 Test model:Wave.mdl

この Wave.mdl に対して，診断機能を組み込み，コンポーネント C_1 とコンポーネント C_2 の入出力の取得と各コンポーネントの入出力の異常検出を行うことで，異常の原因となっているコンポーネントを特定する．

診断機能を組み込んだモデルとして Wave_DCC_JTAG.mdl を作成し，**図 12** のようになった．点線内の要素が診断機能の組込みによって追加されている．

各コンポーネントの入出力が正常か異常かの判断基準として，以下の基準を用いる．

C_1**:入力 a**　$-1 \leq a \leq 1$ ならば正常．(判断基準 2)
C_1**:出力 b**　$-2 \leq b \leq 2$ ならば正常．(判断基準 2)
C_2**:入力 b**　$-2 \leq b \leq 2$ ならば正常．(判断基準 2)
C_2**:入力 c**　$-1 \leq c \leq 1$ ならば正常．(判断基準 2)
C_2**:出力 d**　$-3 \leq d \leq 3$ ならば正常．(判断基準 2)

シミュレーション時間ステップでの時刻（以下，単に時刻と呼ぶ）0.0，1.0，2.0，3.0，4.0，5.0 で診断を行う．また，**表 1** のようなエラーを発生させて，診断できるかを検証する．

5.2 実 験 結 果

診断の結果は**表 2** のようになった．

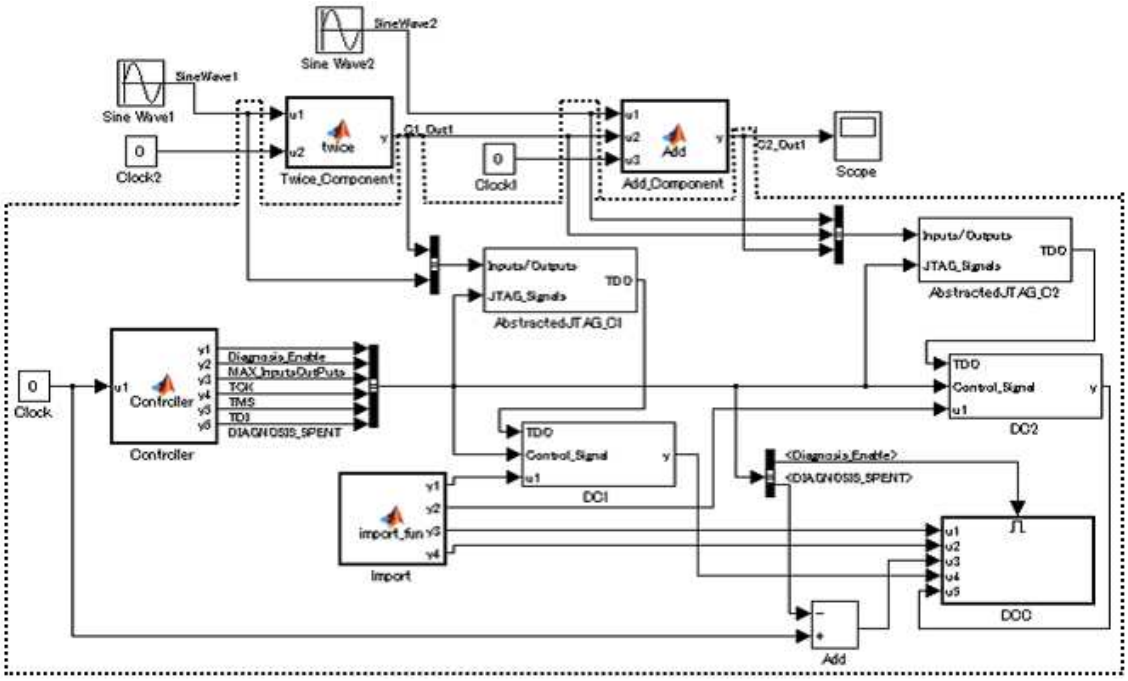

図 12 モデル:Wave_DCC_JTAG.mdl
Fig. 12 Model:Wave_DCC_JTAG.mdl

表 1 発生させるエラーの内容
Table 1 The details of causing errors

時刻	箇所	エラーの内容
1.0	C_1	入力 a を 5 倍にする.
3.0	C_2	入力 b を -1 倍してから加算を行う.
4.0 ~ 5.0	C_2	入力 c を 3 倍にする.
5.0	C_1	入力 a を -4 倍にする.

表 2 診断結果
Table 2 Diagnosis result

時刻	入出力	コンポーネント
0.0	全て正常	全て正常
1.0	C_1 の出力 b, C_2 の入力 b と出力 d が異常	C_1 が異常
2.0	全て正常	全て正常
3.0	全て正常	全て正常
4.0	C_2 の出力 d が異常	C_2 が異常
5.0	C_1 の出力 b, C_2 の入力 b と出力 d が異常	C_1 が異常

時刻 0.0，1.0，2.0，4.0 の結果については満足したものが得られた．特に時刻 1.0 では，C_1 の異常によって，異常伝播問題が発生し，正常な C_2 の入出力も異常な値となったが，異常発生の原因である C_1 の特定を行えた．

しかし，時刻 3.0，5.0 の結果は妥当なものとは言えない．

まず，時刻 3.0 では，C_2 にエラーを発生させたため，出力 d が異常なものとなっているはずだが，判断基準で用いている正常な値の範囲外には出なかったため，異常の検出が行えていない．入力と出力が一定範囲内に収まればよいという判断基準を用いた場合は，定めた範囲を超えさせるほどの大きな異常でなければ，異常検出ができないと言える．

また，時刻 5.0 では，C_1 と C_2 の両方にエラーを発生させたが，異常箇所は C_1 のみと特定されている．これは，異常コンポーネントが少ない DIAG を診断結果とするため，異常伝播問題の可能性のあるコンポーネントは異常ではないと判断するためである．つまり，連続したコンポーネントが同時に異常となってしまう場合は，起点となるコンポーネントのみが異常と特定されてしまう．

6. おわりに

組込みシステムの動作診断のために，本稿では，モデル段階から診断機能を組み込む手法を提案し，アスペクトによって診断機能の部品化と再利用を可能にした．そして，実験によって，MATLAB/Simulink 上でのシミュレーションで診断が行えることを示した．

今後は，MATLAB/Simulink モデルから実際に HW，SW を生成して，診断機能の組込みによって生じ

るオーバヘッドやコードサイズの増加などの評価を行うことで，本手法の実用性について確認していく．また，現在，DIAG については，SD と出力された OBS とを比較して，手でシミュレートすることで求めているため，DIAG もシステムによって求められるようにする必要がある．

最後に，今後の課題を以下にまとめる．

- MATLAB/Simulink モデルから実際に HW，SW を生成しての検証．
- システムによる DIAG の計算．
- データの正常異常についての判断基準の追加．
- ウィーバの開発，およびアスペクト適用の自動化．

参 考 文 献

1) 入月康晴, 大原衛, 坂巻佳壽美: JTAG を用いた組込みシステムのオンライン自己監視手法, 日本信頼性学会誌, Vol,32, No.3, pp.185–190 (2010)
2) 沓名拓郎, 佐藤守一, 中條直也: 抽象モデルベース診断による協調システムの異常原因特定, 組込技術とネットワークに関するワークショップ ETNET2009 論文集, pp.43–48 (2009)
3) http://www.xjtag.com/jp/support-jtag/jtag-introduction.php
4) 千葉滋: アスペクト指向入門, 技術評論社 (2005)
5) 中野真吾: 自律コンピューティングに基づく組込みシステムの動的自己監視, 埼玉大学卒業論文 (2013)
6) 渋田達也: 組込みシステムネットワークへのモデルベース診断の適用, 埼玉大学卒業論文 (2013)