ソフトウェア工学 118－16
（1998．3．11）

ソフトウェア機能試験手順の状態遷移表に基づいた生成法

山中 弘† 中島 毅† 別所 雄三† 広田 和洋‡
†三菱電機株式会社 情報技術総合研究所
‡三菱電機セミコンダクタシステム株式会社

本稿では，状態遷移表で記述されたマンマシン系のマイコン組込みソフトウェアの仕様に基づき，ソフトウェア機能試験手順を自動生成する方法について記述する．我々は，マイコン組込みソフトウェアのシステム試験工程を目的別に，機能適合試験，信頼性向上試験，再試験の３つに分類し，それぞれの性格にあった試験手順生成法について考察した．機能適合試験向けの試験手順生成法としては，従来から知られている手法を採用したが，信頼性向上試験，および再試験向けの試験手順生成法としては，それぞれ試験手順を部分手順毎に生成し結合する手法，および状態遷移のレイヤを指定して手順を生成する手法を新しく提案した．

Methods for generating software functional test sequences from a state transition table

Hiroshi Yamanaka† Tsuyoshi Nakajima† Yuzo Bessho† Kazumi Hirota‡
†Information Technology R & D Center, Mitsubishi Electric Corporation
‡Mitsubishi Electric Semiconductor System Corporation

In this paper, we describe methods for generating test sequences from a state transition table, in which the specification of the man machine interface for microcomputer software is described. We classify the system testing for microcomputer software into three types in terms of its purpose: functional conformance, reliance, and regression testing. We considered the method that generates suitable test sequences for each test type, and established two new methods for reliance, and regression testing. The method for reliance testing is that can create partial test sequences and combine them. And the method for regression testing is that creates test sequences by the layer of state transition table.

1. はじめに

家電品など価格/新製品開発競争の激しい分野を対象とするマイコン組込みソフトウェアの開発では，短納期，高品質，かつ低コストが求められている．また，マンマシン仕様に対する顧客の好みの変化に従って仕様変更が頻繁に発生し，開発工程が圧迫される傾向がある．

マイコン組込みソフトウェア開発の現場においては，この短納期，仕様変更という要因と，システム試験作業自体が手入力と目視確認による労働集約的な作業であるという理由から，試験および再試験に充分な時間をかけることができないという問題がある．このため，出荷後に不具合が見つかるケースが増えている．

我々は，こうした問題に対処するために，マイコン組込みソフトウェア開発における要求分析工程とシステム試験工程を支援する新しい開発支援方式とその支援ツール環境testCASEの研究開発を行っている[1]．testCASEは，状態遷移表で記述された要求分析結果のソフトウェア仕様を基に，試験仕様を自動生成し，かつその試験仕様に基づいた試験を自動化する．

本稿では，testCASEの一機能である，状態遷移表に基づいたソフトウェア機能試験手順の生成法について記述する．

2. testCASEにおける試験実施形態

まず，testCASE[1]における試験実施の考え方について，ここで説明しておく．

2.1. 試験自動実施の手段

一般にマイコン組込みソフトウェアの典型的な構造を図1に示す．

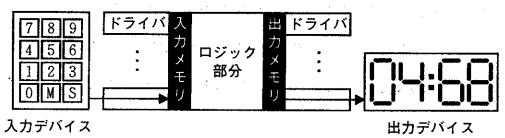


図1．マイコン組込みソフトウェアの構造

図1において，ユーザが入力デバイスに与えた操作は，ドライバを介してある値としてマイコンのメモリに設定される．マイコン組込みソフトウェアはこの値をメモリから取得して応答する処理を行う．処理結果は再びマイコンのメモリに設定され，ドライバを介して出力デバイスに送られる．

このような動作構造に従い，testCASEでは次のような手段を採り，ロジック部分のシステム試験を自動実施する．

まず，各入力操作が発生したときの入力値の情報，およびその入力に対する応答処理が終了した後の出力期待値の情報を入力させる．そして，これらの情報と，ICE(In Circuit Emulator)を用いることによって，

(1) ソフトウェアの動作をある状態で停止させ，
(2) 疑似入力としてある入力操作に相当する値をメモリに設定し，
(3) 応答処理が完了するまでソフトウェアを動作させ，
(4) その結果としてメモリに設定された値を期待値と照合する，

という制御を行うのである．この(1)および(2)を入力ブレーク，(3)および(4)を照合ブレークと呼ぶ．入力ブレークは状態遷移表のイベントに，照合ブレークは状態遷移表のアクションにそれぞれ対応づけられる．

testCASEでは，要求仕様の状態遷移表からイベントの系列と応答するアクションの系列を導き出し，その系列に従って入力ブレークと照合ブレークを実施して試験を自動実行する．testCASEでは，この入力ブレークと照合ブレークの系列のことを，試験ケースと呼んでいる．本稿で試験手順と呼んでいるのは，試験ケースの基データであるイベントとアクションの系列(言い換えると状態遷移の連続した系列)のことである．

2.2. 試験の性格付け

我々は，試験の目的を考慮し，(1)機能適合試験，(2)信頼性向上試験，(3)再試験の3つのアプローチからマイコン組込みソフトウェアのシステム試験を行うことを検討した．以下，この3つのアプローチの内容と，それらに即した試験手順生成に

求められる要件をまとめる．

(1) 機能適合試験

マイコン組込みソフトウェアの開発では，短納期であるためにソフトウェアの単体試験を行わずに，システム試験をいきなり行う場合が多い[1]．機能適合試験は，このような場合にソフトウェアの一通りの機能動作を確認すること．つまり，ソフトウェアの全ての状態において可能な入力操作を行ったときの動作を確認することを目的とする．

この試験を行うためには，状態遷移表の全ての状態遷移を網羅する試験手順を生成する必要がある．

(2) 信頼性向上試験

できる限り多くの試験手順によって試験を行い，高いレベルの信頼性を確認することを目的とする．

この試験を行うためには，状態遷移表からあらゆるパス(状態遷移の系列)を取り出して試験手順とすることが望ましい．しかし，このあらゆるパスを取り出すという要件は，3.2 節に後述するように実際上実現が不可能である．この信頼性向上試験に即した試験手順生成に求められる要件については，3.2節で詳しく考察する．

(3) 再試験

発見されたある障害に対処してソフトウェアの一部の処理を変更した場合，この修正が原因となって発生する別の障害が想定される．本稿でいう再試験とは，このような二次障害の確認を効果的に行うことを目的とし，修正したモジュールを中心に全ての関連モジュールとの組み合わせ動作を確認することをいう．ただし，ここでいう修正とは状態遷移表における仕様の修正ではなく，プログラムにおける処理実装の修正である．

例として，プログラムに修正が加えられ，影響を受ける状態遷移表内のオブジェクトが特定されたとする．それがアクションであった場合，そのアクション処理が終了して到達した状態において，全てのイベントを入力することにより組み合わせ動作の一部を確認することができる．この再試験では，そのアクションの発生直前状態に至る入力操作としても，可能な全てのイベントを入力として想定し組み合わせ確認を行う．

3. 従来の試験手順生成手法

ここでは，まず従来の試験手順生成手法を紹介し，次に 2.2 節の各試験目的に即した試験手順生成法として従来手法の適用を考えたときの問題点について考察する．

3.1. 従来手法

ここで取り上げる従来手法によって生成される試験手順は，以下の 4 種類である．

(1) 全状態遷移網羅手順

状態遷移表の全ての状態遷移を網羅して試験する試験手順である[2]．この試験手順により，ソフトウェアの全ての状態における全ての入力操作の動作結果を一通り確認することができる．

(2) 状態判定系列

状態判定系列を用いた試験は，通信ソフトウェアの分野において提案されている[3][4]．通信ソフトウェアの試験では，(1)に述べた全状態遷移を網羅する入力系列などを試験対象に与え，結果として得られる出力系列が状態遷移表の仕様通りであるかを確認することにより，応答動作試験の合否を決定する[3]．例えば，図 2-(a)の状態遷移表において，入力系列 E1-E1-E2-E2-E2-E1 を与えたときには出力系列 A2-A2-A2-A2-A2-A1 が得られることを確認する．

(a)

	S1	S2	S3
E1	S2|A2	S3|A2	S1|A1
E2	S3|A2	S1|A2	S2|A2

(b)

	S1	S2	S3
E1	S2|A2	S3|A2	S1|A1
E2	S3|A2	S1|A2	S3|A2

図 2. 初期仕様(a)と誤った実装仕様(b)

しかし，この応答動作試験だけでは，入力系列を与えることによって到達した状態が，状態遷移表の仕様通りであることを確認できない．例えば，図 2-(a)が正しい仕様であり，(b)が状態 S3 で入力

E2 が発生したときの遷移先状態を誤って実装したソフトウェアであるとする．このときに入力系列 E1-E1-E2-E2-E2-E1 を与えると，共に出力系列 A2-A2-A2-A2-A2-A1 が得られることになる．

この入力系列を実行した後に到達した状態の確認を行うための試験手順が状態判定系列である．例えば，図 2-(a)の状態遷移表においては，入力系列 E1-E1-E1 を与えその出力系列を照合することによって，今ソフトウェアがどの状態にあるかを識別することができる．出力系列が A2-A2-A1 であれば現在状態は S1 であり，A2-A1-A2 であれば現在状態 S2，A1-A2-A2 であれば現在状態 S3 であることが判る．

状態判定系列としては，DS 系列，W 系列，UIO 系列などが考案されている．DS 系列，および W 系列は，それぞれ 1 つ，または複数の入力系列に対する出力系列を照合することによって，今ソフトウェアがどの状態にあるかを知ることができる．先述した E1-E1-E1 は図 2-(a)の DS 系列である．UIO 系列は，状態遷移表の各状態に対して生成される入力系列であり，現在状態が Si であることを保証するものである．例えば，図 2-(a)の入力系列 E1 は状態 S3 の UIO 系列である．出力系列が A1 であれば現在状態が S3 であることがわかるからである．多くの状態遷移表について UIO 系列は存在し，かつ手順も比較的短いことなどから，実用的な手法として UIO 系列が注目されている．

(3) 全パス網羅手順

全パス網羅手順は，有向グラフの簡約アルゴリズム[5]によって生成される手順である．有向グラフ簡約アルゴリズムは，図 3 に示す簡約基本手順を，これ以上簡約できなくなるまで対象の有向グラフに適用するものである．この手法では，ループ回数を度外視すればあらゆる手順を網羅して生成することができる．状態遷移表にこの手法を用いるときには，状態遷移表に対応づけられる有向グラフを生成し，その有向グラフに簡約アルゴリズムを適用する．

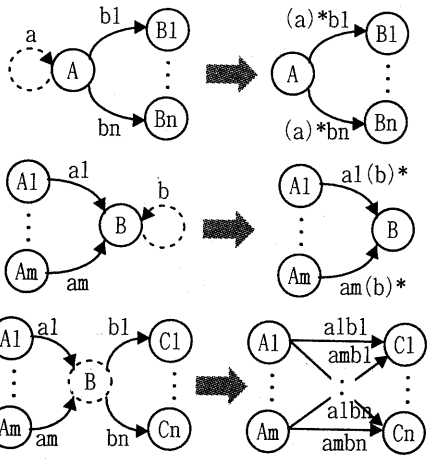


図 3. 有向グラフ簡約アルゴリズム基本手順

(4) 利用モデルに基づく試験手順

後述するように，一般にソフトウェアのあらゆる試験手順を集めると，その数は膨大なものとなる．限られた期間内で試験を実施するためには，できる限り「有効な試験手順」を数多く生成して試験する必要がある．

「有効な試験手順」に対する考え方としては，試験対象のソフトウェアや，試験の目的などによっていろいろな尺度が考えられる．例えばマンマシン系のソフトウェアであれば，実際にユーザがソフトウェアを使用するときに発生しやすい手順ということが一つの尺度である．

この考え方に基づく試験手順の一つが，利用モデルに基づく試験法である[6]．この方法では，各状態遷移が発生する確率を設定し，確率が高い手順から順に試験を行う．

3.2. 従来手法の問題点

以上に述べた従来手法の，2.2 節の各試験目的に即した試験手順生成法への適用と問題点について考察する．

(1) 機能適合試験向きの従来手法

機能適合試験の試験手順として，全状態遷移網羅手順が要件「状態遷移表の全ての状態遷移を網羅する試験手順」を満たす．全パス網羅手順も要件を満たすが，後述するように手順数が膨大になるため不適切である．機能適合試験について，全状

態遷移網羅手順とこの欠点を補う状態判定系列を適用した．

(2) 信頼性向上試験向きの従来手法

信頼性向上試験の理想的要件「状態遷移表からあらゆるパスを取り出して試験手順とする」を満たすのは，全パス網羅手順である．これを生成する有向グラフ簡約アルゴリズムでは，ループ回数を度外視すればあらゆるパスを網羅して生成することができる．ただし，簡約アルゴリズムの対象とする有向グラフが大きくなると，試験手順数が膨大になってしまう．例えば，ノード数28，エッジ数280の有向グラフに対して，簡約アルゴリズムで試験手順を生成しようとすると，手順数が32ビット整数値の上限をオーバーする例が存在した．そこで，試験手順生成法として実用的な手法とはいえず，信頼性向上試験の要件の見直しを行い，他の手法を用いる必要がある．

「ある限度数に抑えたできるだけ多くのパスを取り出して試験手順とする」と要件を見直すと，開始状態を始点とした状態遷移ツリー構造から，逐次探索や乱数を利用して試験手順を生成する手法も考えられる．しかし，これらの機械的な手法では，ユーザがソフトウェアを正しく動作させるための正常手順が含まれない可能性がある．いくら試験手順を数多く生成してそれを試験したとしても，その中にマンマシン系のソフトウェアで最低限試験しておくべき正常手順が含まれていなければ，充分な試験をしたとは言い難い．

手順数と正常手順両方の要件を満たすのが，利用モデルに基づく試験法である．この試験法によれば，実際にユーザがソフトウェアを使用するときに発生しやすい手順を，効率よく試験することができる．しかし，実際には開発時に各状態遷移の発生確率を正しく求めることに困難がある．発生確率の設定を行わなくても，正常手順を生成できるような手法が求められる．

我々は，以上の従来手法の問題点に基づき，

(a) 試験手順数を抑えることが可能であり，

(b) 状態遷移の発生確率に基づかずに正常手順を生成することが可能である，

の2点を信頼性向上試験に即した試験手順生成法の要件とし，これに即した試験手順生成法を検討する．

(3) 再試験に関わる従来手法

3.1節(1)～(4)の試験手順は，修正箇所を中心とした前後との組み合わせを意図して作成されておらず，再試験向けの試験手順とはいえない[1]．

4. 機能適合試験の試験手順生成法

機能適合試験の試験手順生成法としては，3.1節に述べた全遷移網羅試験手順[2]と状態判定系列[3, 4]を採用した．以下にその実現手段について記述する．

まず，全遷移網羅試験手順の実現手段について述べる．全遷移網羅試験手順を採用したのは，これが機能適合試験の要件「ソフトウェアの動作を一通り確認する」を満たすのに必要だからである．全状態遷移を網羅するための試験手順生成法としては，次の実現手段を採った．

(1) ある状態遷移に着目し，試験開始状態からその遷移元状態に至るまでの最小の長さの手順を1つ求める(図4-(b))．

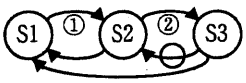
(2) (1)で求めた遷移元状態に至るまでの手順と，対象となった状態遷移を組み合わせたものを試験手順とする(図4-(c))．

(3) (1)と(2)の操作を状態遷移表内の各状態遷移について実施する．

(4) (3)で求められた状態遷移と同数の試験手順を最適化する．ここでいう最適化とは，他の試験手順に完全に含まれている試験手順を破棄することである．

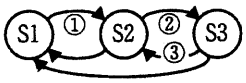
(a)

[1] 再試験に応用できそうな従来手法は発見できなかった．



S3に至る最短手順を求める

(b)



対象の状態遷移と組み合わせる

(c)

図4. 全遷移網羅試験手順生成の手段

次に，状態判定系列の実現手段について述べる．状態判定系列は，全遷移網羅試験手順を実施した後に，ソフトウェアが正しい状態に遷移しているかどうかを確認するために必要であった．状態判定系列のうちUIO系列は，DS系列やW系列と比較して長さも短く，機能適合試験の要件にも合っていることから，UIO系列を生成して試験を行うこととした．

状態判定系列は，ソフトウェアに対しある入力系列を与えたときの出力系列の違いによって状態を判別するものである．そこで，出力系列の各出力要素が明確に識別できる必要がある．我々は，出力要素として状態遷移表のアクションを用いる．そして，ある入力系列を与えたときに，対応するアクションの照合ブレーク系列が全て正しく確認されれば，試験合格とする．

各状態のUIO系列生成法としては，次の実現手段を採った．

(5) UIO系列の長さの最大値をユーザに指定させる．

(6) 各状態について，探索法により最大長以内のUIO系列を求める．ただし，存在しない場合も有り得る．

求めたUIO系列は，他の試験手順生成法で求めた試験手順の末尾に結合する．ただし，ユーザに(a)UIO系列は追加しない，(b)全ての試験手順にUIO系列を追加する，という2つのオプション指定を行わせることとした．これは，次の理由による．マイコン組込みソフトウェアにおける状態は，実装形態によってはいくつかの変数値で識別することがあり，照合ブレークにおいてその値を確認することがある．このような場合には，UIO系列を追加する必要性はないと考えられるからである．

5. 信頼性向上試験のための試験手順生成法

3.2節で述べた要件「試験手順数を抑えることが可能」，「状態遷移の発生確率に基づかずに正常手順を生成することが可能」を満たす信頼性向上試験の試験手順生成法について，我々は以下の実現手段を採った．

(1) 試験手順の部分手順を生成する．部分手順の生成手段として，次の2つがある．

(a) 部分手順の開始状態，終了状態，手順の長さL，および手順の最大数Mを指定させ，その状態間を結ぶ長さLの手順をM以内の数求める．このときに途中で通過してはならない状態，発生してはならないイベント，含めてはならない状態遷移を指定することができる．

(b) 複数の状態とイベントを指定させ，その交差部の部分状態遷移表から生成することのできる有向グラフについて，有向グラフ簡約アルゴリズムを適用し試験手順を生成する．

(2) 生成した各部分手順を結合する．結合手段として，和結合と積結合の2つを提供し，手順数の調整を可能としている．

2つの部分手順集合
$A=\{Ai \mid 0\leq i<m\}$, $B=\{Bj \mid 0\leq j<n\}$ があるとき

和結合によって得られる試験手順
$\{AiBi' \mid Ai\in A,\ Bi'\in B,\ 0\leq i<m,\ i'=i \bmod n\}$ $(m>n)$
$\{Aj'Bj \mid Aj'\in A,\ Bj\in B,\ 0\leq j<n,\ j'=j \bmod m\}$ $(m\leq n)$

積結合によって得られる試験手順
$\{AiBj \mid Ai\in A,\ Bi'\in B,\ 0\leq i<m,\ 0\leq j<n\}$

上述の手段によれば，まず(1)の(a)，(b)の2つの部分手順生成手段を選択できるようにしたことと，(2)の結合法を選択できるようにしたことで，試験手順数を抑えることが可能となる．(1)の(b)が部分状態遷移表に対するパス試験法であり，このときに手順数が大きくなった場合は(a)の逐次探索法を選択して数を減らすことが可能となる．

正常手順を状態遷移の発生確率に基づかずに生成する要件については，試験手順を部分毎に分けて生成し，試験者が認識している状態遷移の意味的なつながりの強さを生成時の情報として採り入れたこと，(1)-(a)の逐次探索法において試験手順に対する制約を設けたこと，により実現が可能になった．(1)-(a)の制約は，マイコン組込みソフトウェアの正常手順においては，途中に重要な意味のを持つ状態が存在する場合が多いこと，「電源OFF」など正常手順の途中に出現してはならないイベントがあること，を考慮して設定した．

6.再試験のための試験手順生成法

再試験のための試験手順生成法として，我々はレイヤ指定による試験手順生成を検討した．

レイヤの概念について説明する．ある状態遷移の第 N 上位(下位)レイヤ手順とは，その状態遷移の遷移先状態からの(遷移元状態に至る)長さ N の手順のことをいう(図5)．つまり，ある状態遷移を中心にして，有向グラフを 1 つずつ手繰っていくのである．

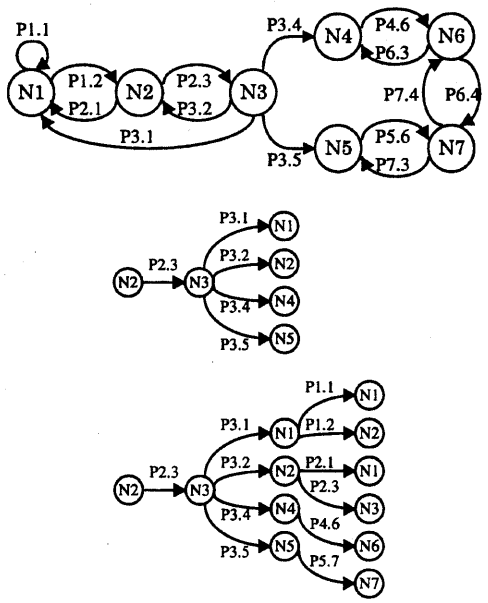


図 5. P2.3 の第 1 上位レイヤ手順と
第 2 レイヤ手順

このレイヤ指定による試験手順生成法は，以下の実現手段である．

(1) 試験対象となる状態遷移をユーザに指定させる．

(2) 上位レイヤ長(0 以上)，および下位レイヤ長(0 以上)をユーザに指定させる．

(3) 指定されたレイヤ長の全ての下位レイヤ手順と，試験対象となる状態遷移と，全ての上位レイヤ手順を積結合する．例えば，下位レイヤ手順数が N1，上位レイヤ手順数が N2 であれば，作成される手順の合計は N1*N2 である．

(4) (3)で求めた各組み合わせ手順を，試験開始状態からの試験手順として生成し直す．例えば，組み合わせ手順 T の先頭の状態が Si であれば，試験開始状態 S1 から Si に至る最短長の手順 T'を 1 つ求め，T'と T を組み合わせた手順を試験手順とする．

上述のレイヤ指定による試験手順生成法によれば，修正の影響が予想される状態遷移と前後の状態遷移との組み合わせを試験することができる．

7.おわりに

マイコン組込みソフトウェア開発支援ツール環境 testCASE の一機能である，状態遷移表に基づいたソフトウェア機能試験手順の生成法について説明した．

testCASE では，試験の目的によって機能適合試験，信頼性向上試験，再試験の 3 つのアプローチからシステム試験を行うこととし，それらの性格付けにあった試験手順生成法を実現した．

機能適合試験を行うための試験手順としては，従来の全状態遷移網羅手順と UIO 状態判定系列とを生成する手法を採用した．この手法により，ソフトウェアの一通りの動作確認を実施するための試験手順が生成できる．

また，信頼性向上試験を行うための試験手順としては，制約付けられた部分試験手順を生成して結合する手段を提案した．この手法により，試験期間に見合った数の試験手順で，状態遷移の発生

確率に基づかずに正常手順を生成することが可能となった.

また，再試験を行うための試験手順としては，レイヤ指定による試験手順生成法を提案した．この手法により，修正モジュールと関連しあうモジュールとの組み合わせ再試験を実施するための試験手順が生成できる.

今後は実プロジェクトでの結果を踏まえ，各試験手順生成法の定量的データの分析を行う予定である.

参考文献

[1] 中島毅，別所雄三，山中弘，広田和洋，「シングルチップマイコン用 S/W 開発における問題点と一解決法」，情報処理学会研究報告，Vol. 97, No. 74, pp. 17-24, 1997.

[2] 古川善吾，野木兼六，徳永兼司，「AGENT：機能テストのためのテスト項目作成の一手法」，情報処理学会論文誌，Vol. 25, No. 5, pp. 736-744, 1984.

[3] 「新版 情報処理ハンドブック」，オーム社，pp. 1030-1034, 1995.

[4] Barry S. Bosik, M. Umit Uyar, "Finite state machine based formal methods in protocol conformance testing:from theory to implementation, " Computer Networks and ISDN Systems, Vol. 22, pp. 7-33, 1991.

[5] Boris Beizer(小野間彰，山浦恒央訳)，「ソフトウェアテスト技法」，日経 BP 出版センター，1990.

[6] James A. Whittaker, J. H. Poore, "Markov analysis of software specifications, " ACM Transactions on Software Engineering and Methodology, Vol. 2, No. 1, pp. 93-106, 1993.