ソフトウェア工学 115－3
（1997．7．31）

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

中島 毅† 別所 雄三† 山中 弘† 広田 和洋‡
†三菱電機(株) 情報技術総合研究所
‡三菱電機セミコンダクタソフトウエア株式会社

本論文では、シングルチップマイコン用S/W開発の特徴と問題点を、対顧客関係とS/Wの性質と既存ツールの面からまとめ、その結果に基づく新しい支援方式を提案する。本支援方式は、状態遷移表で記述された仕様からテスト手順を自動生成し、生成された手順を用いてマイコンS/Wのテストを自動化する。これは、マイコンS/Wが仕様とプログラムの両面にわたって入出力デバイスとロジック部分に分かれるという性質を利用し、仕様記述に現れるイベントとアクションを、それぞれメモリ上の擬似入力発生とメモリ－期待値の照合に置き換えることにより可能となる。これによって、テスト作業量の大幅削減と製品品質の向上を達成することができる。

Problems in Microcomputer Software Developments and a Solution to Them

Tsuyoshi NAKAJIMA†, Yuzo BESSYO†, Hiroshi YAMANAKA†, and Kazumi HIROTA‡
†Information Technology R & D Center
Mitsubishi Electric Corporation
‡Mitsubishi Electric Semiconductor Software Corporation

In this paper, we discuss characteristics and problems in microcomputer software developments from the aspects of relationship with customers, software artifacts, and existing tools, and provide a new way of supporting environment based on this discussion. This environment generates test data from state transition table, and uses the test data to automate functional testing, by taking advantage of the typical structure of micro computer software, i.e., separation to input/output device drivers and logical body in both specifications and program codes. The environment allows us to dramatically reduce testing cost and to increase the quality of software products.

1．はじめに

シングルチップマイコンは、元来家電品などの価格競争と新製品開発競争の激しい分野の製品にとりれられるため、そのS/W開発では短納期かつ高品質・低コストを求められることが多い。

我々は、シングルチップマイコンS/W開発における要求定義とテストを支援する方式とその支援環境(仮称：testCASE)を研究開発している。その開発にあたり、まず対象ドメインの特徴と問題点を洗い出すための現状調査と既存支援ツールの分析を行った。

本論文では、まずその分析結果について報告し、さらに一解決法としてtestCASEのアプローチを提案し、その効果と課題について議論する。

2．マイコンS/W開発における問題点

現状調査は、汎用マイコン用のS/W開発を行う特約店SEのグループを対象に行った。調査方法は、まず要求定義段階とテスト段階に絞って問題意識を調べるアンケートを行い、さらにアンケートに基づきヒアリングを実施した。以下に分析結果を示す。

2.1 顧客との関係と要求定義の現状

シングルチップマイコンS/W開発は、基本的に特定顧客を対象とした受注型のS/W開発である。マイコンの特約店にとっての顧客とは、家電品や通信機器などのH/Wメーカである。顧客はマイコンを数量ベースで購入する。そのマイコンは、顧客が製造するH/Wに組込んで要求通りに機能することが求められる。そのため、S/W自体が要求通りであることよりも、「顧客のH/Wが顧客の要求通りに動く」ことを保証することを強く求められる。

顧客の中には、マイコンS/Wを熟知しS/W仕様を詳細かつ正確に記述できる顧客もいれば、マイコンS/Wがどんなものか知らない顧客もいる。後者の顧客は開発製品全体の要求機能を記述できるが、それを実現するために組込まれるマイコンS/Wの仕様を記述できない。開発者は、後者の顧客の場合の要求分析工程（およびその後工程）を問題と考えている。

後者の場合も、本来が顧客から文書によって要求仕様を受けるべきであるが、ときとして、S/W開発者が、顧客の口頭あるいはメモ書きの要求を基に、S/Wの仕様として外部仕様書を作成する場合がある。要求仕様の正しさは、この仕様書を顧客の参加の下にレビューを行うことで確認している。開発者は、現状の外部仕様書の記述が顧客にとって理解し難く、顧客レビューには不十分な形式であると考えている(仕様記述については後述)。

開発者は、要求仕様が頻繁に変更される点も問題であると考えている。要求定義段階で仕様が固定しない理由としては、顧客側で並行開発を行っているH/Wが仕様変更したり、実際にテスト段階になってから顧客の好みに直接結びつく表示系・操作系部分の仕様を変更することが多い。こうした変更は、後工程であればあるほど改修に時間を要しかつ品質へ悪影響を及ぼしている。

2.2 テストの現状

シングルチップマイコンS/Wの開発コード規模は、5K～100KLOC程度であり中小規模であるが、その納期は3カ月～1年程度である。しかもその間に、テストパターン（ROM焼きした試験版）を製作するため、一般のS/W開発から見れば、非常に開発期間は短い。

マイコンS/Wの評価テストでは、周辺装置を同時開発している場合が多くS/Wだけのテストというより H/Wも含めたシステムテストの意味合いをもつことが多い。そこでは、H/Wの多少の問題はS/W側で吸収することが常識となっている。S/Wの単体テストは工程に組込まれていないが、これは短納期ゆえにその余裕がないからである。開発者は、評価テストだけでは十分ではないと考えており、可能ならば単体テストを実施したいと考えている。

マイコンS/Wの評価テストは、周辺装置をつないだ状態で行われるため、入力装置へ手入力を行い出力装置に表示される結果を目で確認する作業が基本になる。このため評価テストは効率化を図ることが困難な労働集約的な作業となっている。さらに、前述のように顧客からの仕様変更が生じるため、短納期の中でテストおよび再テストを行う時間を確保できない。このテストの不十分さゆえに、出荷後に不具合が現れるケースが増えてきている。

3．シングルチップマイコン用S/Wの特徴

ここでは、シングルチップマイコン用S/Wを生産物の視点から見た特徴についてまとめる。

3.1 一般的特徴

マイコンは、周辺装置と直接（あるいはI/Fボードを介して）ポートに接続されている。入力装置はキーイン操作などを入力信号としてポートへ送り、出力装置はポートの信号を読み、出力に変換する。

マイコン用S/Wは、マイコンに組込まれて、入力装置からの入力信号を、出力装置を制御する出力信号へ変換する役割を持ったS/Wである。

マイコンプログラムは、この構造を反映して次の3つの部分に分かれている場合が多い(図1)。

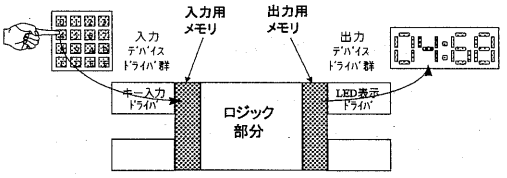

図1. マイコンS/Wの典型的構造

入力デバイスドライバ: ポート上の入力信号を処理し入力用のメモリ上に値を書き込む部分

ロジック部分:そのメモリ上の変化をイベント発生とみなし要求される出力を出すために出力用のメモリ上に値を書き込む部分

出力デバイスドライバ: 出力用のメモリの値を読みポートへの信号として書き込む部分

3.2 仕様記述の分類

マイコンS/Wの仕様記述の現状を知るために、異なる顧客に対する3編の外部仕様書を調査した。いずれの仕様書も、仕様記述の内容は、大きく次の4つの部分に分類できることが分かった。

1)入出力機器とマイコンポートとの接続

2)入出力装置の仕様（キーマトリックスならキー入力の解釈方法、発信機なら出力波形）

3)ふるまい仕様

4)タイマ割付と割り込み制御

1)は、S/Wの仕様ではなくS/Wを作る上でのH/W条件を記述している。4)は、要求仕様というよりプログラム設計に通じる。これらはプログラムのアーキテクチャを決め性能へ大きな影響を与える。2)と3)がS/Wの機能仕様にあたり、それぞれ図1に示すプログラム構造における入出力デバイスドライバとロジック部分とに対応している。

3.3 ふるまい仕様のための記述方式の現状

ふるまい仕様では、仕様記述方式として、状態遷移図、状態遷移表、イベント・アクション対応表、あるいはタイミングチャートが使われている。

状態遷移図あるいは表は、表示系や操作系のふるまい仕様を記述するために多く用いられている。これは、一般にマイコン応用機器の表示系・操作系が少ない入力手段に対して機能が豊富であるため、複雑なモード遷移を必要とするためであろう。状態遷移表は、評価テストのための基データとして使うことが可能なため、ふるまいの記述法として比較的受け入れられてきている。イベント・アクション対応表は、ボタンを押すとある信号コードが出力されるような比較的単純な仕様に用いられている。状態遷移図は、遷移表ほど市民権を得ておらず、一部の開発者がマンマシン仕様の最初の大まかなモードを洗い出すために用いている程度である。イベント・アクション対応表は信号変換を行う非常に単純なケースに用いられている。

タイミングチャートは機器制御の時間的な流れを記述するために用いられている。この種の仕様は状態遷移図（または表）にタイムアウト仕様を書けるように拡張することによってより形式的に記述することができるが、このようにして用いている例はなかった。これは、顧客レビューのしやすさを重視しているためである。

ふるまいの仕様が顧客要求に直接的に関わっているため、その記述方式は顧客がレビューしやすいように考慮されている。例えば、状態遷移図は状態を表すのに、状態名を書いた箱を用いるのではなく、表示パネルそのものの絵を使うなど工夫を凝らしている。

3.4 ふるまい仕様の分類とその割合

前節に現れたふるまい仕様を性質の違いから、仕様を次の6つのクラスに分けた。

(1) イベント・アクション：イベントに対するアクションの形式で記述される仕様クラス。主に表示系や操作系の動作仕様を記述するのに使う。

(2) タイミング：一連のアクションがタイムスケジュールに沿って起動されることを記述する仕様クラス。主に機器制御仕様を記述するのに使う。

(3) 安全性：あってはならない／してはいけないこと。

(4) 性能：ある時間幅内に特定の機能が収まるかどうか。

(5) フィードバック：連続系の制御を系の中に含むもので、多くは数学モデルやグラフなどで記述される仕様クラス（炊飯器における炊飯曲線のようなもの）。

(6) 快適度：表示上の見栄え(表示のちらつき等)、操作上の快適さ(微妙なレスポンス等)など、あらかじめ記述できないが最も厳しく評価される仕様クラス。

表1に、それぞれの仕様クラスがふるまい仕様の中でどれくらいの割合を占めるかをマイコン S/W が組込まれる代表的なターゲット別にその仕様を示した（この表は、完全な統計に基づくものではなく各分野の開発者に感覚的に割合を答えてもらったものである）。

表1. 仕様の種別と分野別割合

	PHS	TV	VTR	ファンヒータ	炊飯器	コードレス電話	PPC	FAX
イベント・アクション	6	7	7	3	4	7	5	4
タイミング	2	1	1	2	1	1	1	3
快適度	1	1	0.5	1	1	1	3	1
安全性	1	1	0.5	3	3	1	1	2
性能	0	0	0	1	0	0	0	0
フィードバック	0	0	1	0	1	0	0	0

表1より、イベント・アクション仕様がふるまい仕様全体の 5 割以上を占めていることが分かる。性能仕様の記述は少なく、比較的タイミング仕様が多い。

3.4 プログラム実装での特殊性

要求する機能を実現するプログラムを、顧客が選択したマイコンのメモリ容量以内に実装することが要求される。シングルチップマイコン S/W の主なターゲットである家電品は、コスト競争が激しく、顧客は製品に組み込まれるマイコンもなるべく低価格のものを使いたいと考えるため、メモリ容量が要求機能に対してぎりぎりのマイコンが選択される傾向がある。

低価格帯のマイコンでは未だに言語の主力はアセンブリ言語であり、メモリ上にプログラムを詰め込むための最終調整を人手によって行っている（これを ROM 圧縮と呼んでいる）。OS は存在せずプログラムの実行制御は開発者が作り込む。

中価格帯では S/W の規模が大きくなるので開発効率が多少考慮される。従って C 言語が使われることが多い。分野別に標準 OS や特定目的の自社製 OS などが用いられている。

4．現状の支援ツールの研究

ここでは、マイコン用 S/W 開発のテストに関連して採られてきた3つのツール支援方式について分析する。

4.1 ソフトウェアシミュレーション法

ソフトウェアシミュレーション法とは、実機や特殊な H/W を使わずにマイコン及び外部の周辺機器をすべてソフト的に模擬し、開発したプログラムを実行する方法である[1]。

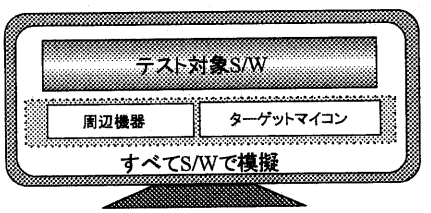

図2. ソフトウェアシミュレーション法

この方法は、マイコンと実機を必要としないので、比較的早期に S/W 自体の論理的なエラーを見つけることができる。しかし、評価テストとして使うことはできない。なぜなら、論理的に正しいことを確認しても、その S/W をマイコンの少ないメモリにのせるためにコード最適化をマニュアルで行う場合が多いこと、顧客は評価テストを H/W 込みで動作を確認したいこと、開

発者にとってH/Wを/S/Wで模擬するための設定が面倒であり設定自体に仕様の誤解が入る危険性があることなどによる。そのため、この方法は、シングルチップマイコン用S/W開発では普及していない。

4.2 完全ブラックボックス法

完全ブラックボックス法とは、マイコンとそれにのったS/Wを完全な機能ボックスと考え、マイコンのポートに対する入出力だけを用いて評価テストを行う方法である[2]。図3にこの方法の概念図を示す。試験環境は、実機環境（実機と周辺機器あるいは治具）そのままである。図3で中心的な役割を果たすのが信号設定・照合装置である。ユーザは、PC側から入力値と期待値を設定することができる（ウィンドウ擬似部品で行うサポートも一部ある）。信号設定・照合装置は、S/Wの載ったマイコンをポート接続を介して入力と照合を繰り返すことで自動試験を行う。

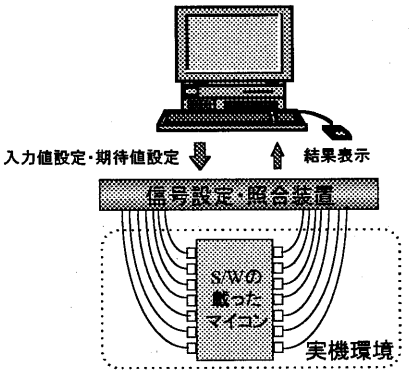

図3．完全ブラックボックス法

この方法の利点は、周辺機器を接続し実時間で動作させるため評価テストとして使えること、入力と期待値を一度記憶してしまえば例えプログラムの構造が変化しても何度でも再現できることである。しかし、特殊なH/Wが必要であることと、期待値の設定が非常に難しいことなどから短納期の開発では導入が事実上困難である。

この方法では、特に時間が絡む入出力を扱うと入力値と期待値の設定が非常にやっかいとなる。例えば、単純なロジック「KEY1～9を押したら1個目のLEDにその数字が表示されること」を評価したい場合を考える。この場合、ポートに現れる入出力信号は、複雑で抽象度が低い。例えば、入力であるキーには20msの2回一致で1つキーインされたと見なされるという仕様があったり、出力である6個のLEDには20ms毎に点灯するLEDを変える(ダイナミック点灯)信号が出力されていたりする。

4.3 コード生成法

コード生成法とは、仕様として記述した状態遷移図あるいは表から、状態遷移を管理する部分（アクションディスパッチテーブル）とアクションの雛形に相当するコード部分を生成する方法である(図4)。

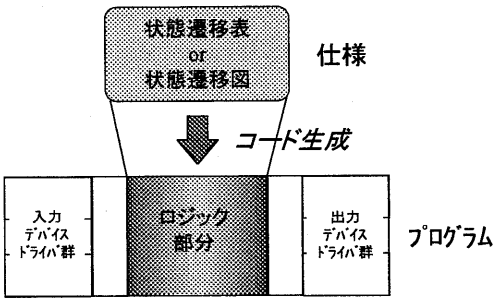

図4．コード生成法

コード生成法は、評価テストを直接行うわけではないが、仕様とコードが等価であることを保証する。市販ツールの中には、状態遷移表を記述することによって、仕様レベルのシミュレーション実行を行えるものもある[3]。

状態遷移図あるいは表からコードを生成する試みは、今まで幾度となく繰り返されてきているが、開発の現場に適合してこなかった。これには、以下に述べるいくつかの理由がある。

- マンマシン系の仕様以外、例えば機器制御等の時間要素を含む仕様などは状態遷移では書かない場合が多い。コード生成法は、その生成コードがマニュアルコーディングで書いたコードとアーキテクチャミスマッチを起こしやすい。
- アクションディスパッチのためのプログラム規模はセル数(イベント数×状態数)に比例する。状態数はモード要因の排他的状態数の積となるので、これを

1枚の状態遷移表にする場合、状態数は非常に大きくなる。プログラム設計では、プログラム規模を抑えるため、1枚の表ではなく互いに関連が疎な複数の表に分割し、その表を実装する方法を採っている。

並列状態を持つStatechart[4]のような記述法を用いれば後者の問題はカバーできるが、並列状態の難解さと評価テストのしにくさのために普及はしていない。

5．我々のアプローチ

前節で述べたように、完全ブラックボックス法は、マイコンS/Wの評価テスト法として優れた特長をもっているが、期待値設定に手間がかかるため短納期の開発では受け入れにくいという欠点を持っている。この問題は、次の2つの問題に帰着する。

P1. 機能要求を（せっかく仕様書として書くにも関わらず）テストデータに結び付けるすべがない。

P2. 機能要求の記述レベルと期待値の設定の抽象レベルがかけ離れている。

我々は、上記の2点を解決する方法を提案する。

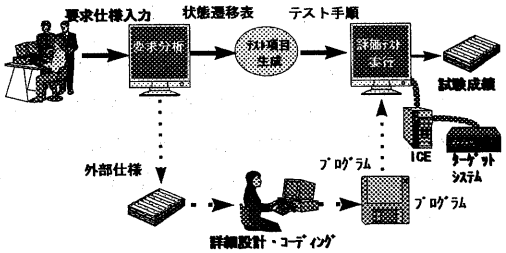

図5．提案する方法

5.1 提案する方法のアウトライン

P1の問題を解決するために、我々の方法では、次のステップにしたがって、仕様入力に基づきテスト自動実行を行う(図5)。

S1. 要求仕様を状態遷移表によって記述する

S2. 要求仕様からテストケースを生成する

状態遷移表からテスト手順を生成する。このアルゴリズムとしては、状態遷移をエッジとして全パス展開する方法[5]などのように自動展開する方法やウィンドウ部品を用いた手入力による方法などがある。

S3. テストケースを自動実行し結果をレポートする

PCからICE（In Circuit Emulator）を制御してテストの自動実行をする。テストケースに含まれる入力イベントと出力アクションを、それぞれプログラム上の擬似入力の発生と期待値照合に置き換える。照合結果は試験報告書の形式で出力する。

5.2 入力・照合ブレーク

P2の問題に対して次のようにアプローチする。

まず、ヒアリング調査の結果から、機能要求として単純な形式で記述される仕様は、プログラム中でもロジック部分として分離して実現されることが多いという事実に着目した。つまり、マイコンS/Wの構造は、ポート上に現れる低レベルの信号を処理する入出力デバイスドライバモジュールと、それらによって単純化された入出力を扱うロジック部分とに分離されている場合が多い(図1)。入出力デバイスドライバとロジック部分は、メモリを用いてその単純化された入出力をやり取りする。

例えば、先ほどの例では、KEY1～9を押したという入力があった状態は、入力デバイスドライバモジュールを通過した後有効な入力があることを示すフラグを立てそのキー値をあるメモリに格納するという状況に相当する。さらにKEY1～9が押された結果1つ目のLEDの表示がそのキー値になるという出力仕様は、出力デバイスドライバ内でLEDの値を格納するメモリ配列の1番目にキー値が設定されることに相当する。

入力デバイスドライバとロジック部分の間のやり取りを入力イベント、ロジック部分と出力デバイスドライバの間のやり取りを出力アクションとして捕らえたならば、上の例は仕様と次のような対応をなしている。

・入力イベント： KEY1～9を押した

- WHEN　入力デバイスドライバモジュール通過後
- 設定値　[有効入力であることを示すフラグ]=ON
  [キー値保存メモリ]=キー値

・出力アクション： LEDの表示がそのキー値になる

- WHEN　出力デバイスドライバモジュールの出口

■　期待値　[LED値を格納する配列1番目]=キー値

一般に、メモリを介して行われるロジック部分への入出力は、1)ポートへの入出力に比較して、時間的な要素が少なく擬似入力と期待値を設定しやすく、2)仕様に近い抽象レベルにあるため、仕様から直接テストデータを作りやすい。

我々の方法の基本的アイデアは、状態遷移表から導かれたテスト手順を、プログラムのメモリ上における擬似入力設定と期待値照合の系列に変換することである。これは、状態遷移表上に現れるイベントとアクションを、プログラム上でICEからコントロールしてメモリに値を設定する**入力ブレーク**と期待値と照合するための**照合ブレーク**とに対応付けて定義することによって可能になる(図6)。

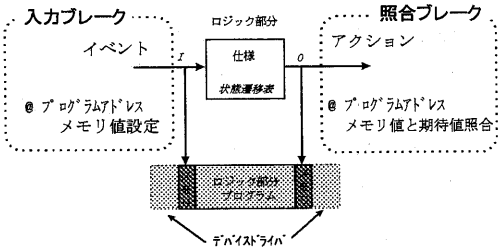

図6．入力・照合ブレーク

これによって、仕様のロジック部分に対応するプログラム部分に対して、仕様に基づくテスト自動化が可能になる。

1)　ロジック部分に対応する仕様が実行可能ならば、入力イベント系列*I*に対する出力アクション系列の期待値*O*を生成する。

2)　ロジック部分に対応するプログラムをテストする。この部分の内部的な実現方法に対して何も仮定をおかずに、その外部的なふるまい、すなわち入力ブレーク（あたかも*I*の各入力イベントが生じたようにする）に対して照合ブレーク（*O*の各出力イベントに対応）が満たされることをチェックする。

6．効果

testCASEのアプローチの利点は、評価テストの自動化による作業効率と品質の向上にある。この点について評価する。

家電品向けマイコンS/Wの機能仕様を書いた状態遷移表の大きさは、イベント数100×状態数100以上となる。通常の評価テスト作業では、テスト手順の作成を行い、実際にテストを実施し、その結果を記録する。一方、testCASEを利用した場合では、ブレーク情報の設定を行う作業の他は、テスト手順の生成とテスト実行・レポート作成が自動的に行われる。

まず、テスト実施時間に注目する。120×120の表で単純に14400個のテスト項目があるとする。通常の評価テスト作業では、1項目あたり20秒で人手で行うとすると、80時間（8時間×10日間）の作業量となる。testCASEを利用した場合では、1項目の処理時間は、タイムアウトで照合エラーになる場合(2秒)も含めて平均1秒かかるものとすると、4時間でテストが完了する。しかも、この間作業者の時間を拘束しない。

次にテスト手順生成について考える。通常の評価テスト作業では、テスト手順作成作業は大変時間がかかるため、通常は改めて作らず、状態遷移表で代用している。このため、テスト手順の効率が悪かったりテスト項目を網羅できなかったりすることが多い。この問題はtestCASEが自動生成することで解決する。

testCASEを利用した場合の再試験の容易さを評価する。表2は、状態遷移表内で記述されている仕様に対する変更がブレーク定義等のテスト設定用記述にどのように影響するかを示したものである。

表2．仕様変更に伴うブレーク定義

仕様変更部分		ブレーク設定変更部分
アクション内容修正		照合ブレーク定義の変更
状態変更	増加(アクション追加)	照合ブレーク定義の追加
	削る(アクション削除)	変更なし
イベント変更	増加(イベント+アクション追加)	入力+照合ブレーク定義の追加
	削る(アクション削除)	変更なし
遷移先変更		変更なし
条件分岐変更	条件式のみ	変更なし
	増加(アクション追加)	照合ブレーク定義の追加
	削る(アクション削除)	変更なし

ブレーク定義情報は、プログラム制御構造に大きく依存する。その制御構造自体が大きく変わった場合には、ほとんど再設定しなければならないが、一般に状

態遷移表で記述される範囲の仕様変更に対してテスト設定用記述の変更量は小さい。

仕様変更に伴うテスト設定変更を行えば、再テストは初回のテストと同じように自動的に行えるので、修正に伴う再テストにおけるテスト項目の漏れはない。これは、現状再テストを十分に行っていない点を考えると、非常に品質の向上に貢献するものと考える。

7．適用範囲と課題

7.1 評価テストとしての有効性

評価テストはマイコンが実機に組込まれた状態で、実機が機能することを確かめなければならない。我々の方法は、実機に載った状態でテストを行うことができるので、S/Wシミュレーション法のような検査の2度手間はない。しかし、最終的なポート入出力を検査するわけではなく、プログラム内部の変数状態を照合するので、評価テストとして有効性を保証するためには、入力デバイスドライバと出力デバイスドライバがそれぞれ正常に機能していることを別途検査する必要がある。

7.2 対象仕様種別

我々の方法は、主にイベント・アクション仕様をカバーするが、この部分はふるまい仕様全体の50%強である(表1)。タイミング仕様と性能仕様（2時点間の時間に関する言明）については、ICEでのエミュレーション実行のため、正確に実時間での実行はできないので、これを満たしているかどうかを検査することはできない。ただしカウンタ変数を用いてメモリ上に時間値を記録することで、間接的にこうした仕様の検査は可能である。また、安全性・快適度・フィードバック仕様も対象外である。

また、モータ制御など動力系と接続した状態でテストを行うと動力系に悪影響が出る危険性がある。

7.3 状態遷移表を用いることの問題点

状態遷移表は、記述の単純さと機能的なテスト空間（イベント×状態）の網羅性ゆえに、現場への導入に対する抵抗感は少ない。一方で、その分析過程は難しいという評価がある。特に、開発者は状態の洗い出しに困難を感じるため、状態の分析が行いやすい抽象状態をもつ状態遷移図を併用することなどが必要である。

設計時は、実装のコンパクトさを求めて複数の並列な小さな表を使い、テスト時は機能テスト空間を網羅するマージした大きな1枚の表を使う傾向がある。こうした記述内容のギャップを埋める工夫も必要である。

8．おわりに

シングルチップマイコンS/W開発ドメインの特徴と問題点を洗い出すために現状調査を行い、その結果を要求定義とテスト段階の問題と、生産物の面からの特徴としてまとめた。次に、既存支援ツールの特長と問題点の分析を行った。さらに、状態遷移表で記述した機能仕様から、テスト手順を生成し、そのデータに基づいてICEを利用して評価テストを自動的に行う方式を提案し、その効果、適用範囲、課題について評価を行った。同方式によれば、テストの自動化によって、同工程における大幅な作業量の削減と、製品品質向上への大きな効果がある。今後は、実プロジェクトへの適用を通じて得た評価データを分析していく。

参考文献

[1]三木他、マイコンプログラムの仮想実行方式、第53回情処全大7D-10

[2]橋本、ADO(自動テスト実行ツール)のテストレポート、MITECH会第8会SE FORUM分科会、1996.

[3]渡辺、状態遷移表を中心としたマイコン組み込みシステム用開発環境ZIPC、第53回情処全大7D-9

[4]D. Harel, On Visual Formalisms, CACM, Vol. 31, No. 5, pp. 514-530, 1988.

[5]B. Beizer（小野間他訳）, ソフトウェアテスト技法, 日経BP, pp. 206-211, 1990.