

## 固有初期値伝播法を用いた 部分スキャン回路のテスト生成

村井真一<sup>†</sup> 萩原拓治<sup>††</sup> 米森玄一<sup>†</sup>

本論文では、故障の影響により記憶素子に値が設定できなくなる故障を検出するテスト生成および故障シミュレーション方式について述べる。故障の影響により初期状態が設定できなくなる記憶素子  $n$  それぞれに、故障回路での固有な初期値  $X_n$  を与え、故障の影響  $0/X_n$  (正常回路の値/故障回路の値) および  $1/X_n$  を観測することにより、従来検出できなかった記憶素子のクロックを OFF にする故障を検出する。本手法を非スキャン同期回路 5 品種および部分スキャン回路 7 品種に適用した結果、固有初期値  $X_n$  を使用することにより、故障検出率は非スキャン同期回路で平均 4.49%、部分スキャン回路で 1.78% 改善できた。また、固有初期値を使用することにより、実行時間は非スキャン同期回路で平均 41%、部分スキャン回路で 25% 削減できた。

### Test Generation for Partial Scan Circuits Using Individual Initial Value Propagation

SHINICHI MURAI,<sup>†</sup> TAKUJI OGIHARA<sup>††</sup> and GENICHI YONEMORI<sup>†</sup>

This paper describes a test generation and fault simulation method which detects faults in the clock control logic that could cause register initialization failures in sequential circuits. By assigning an individual initial value  $X_n$  to the inaccessible register  $n$  in the faulty circuit and observing both the fault effects  $0/X_n$  (good value/faulty value) and  $1/X_n$ , previously undetectable faults can be detectable. Application results of this method to 5 non-scan synchronous circuits and 7 partial scan circuits show that this method can increase fault coverage by 4.49% for non-scan synchronous circuits and by 1.78% for partial scan circuits compared with conventional methods. This method can also reduce execution time by 41% for non-scan synchronous circuits and by 25% for partial scan circuits.

#### 1. はじめに

スキャン設計<sup>1),2)</sup>の導入によりテスト生成コストは大幅に削減できる。しかし、ハードウェアオーバーヘッドおよび動作速度の低下という点で必ずしも全スキャン設計が適用されるとは限らず、部分スキャン設計の採用が望ましい場合も多い。全スキャン設計ではすべての記憶素子の値の設定・観測が可能のため、テスト対象回路を組合せ回路として取扱可能となり、テスト生成は容易である。一方部分スキャン設計では、対象回路を順序回路として取り扱う必要があることからくる困難に加えて、クロック信号およびクロック制御回

路から構成されるクロック系回路の故障の中には、非スキャン記憶素子への値の設定を不能にするものがあり、テスト生成は容易ではない。非スキャン記憶素子のクロック系回路の故障で、その故障の影響により故障回路の記憶素子の値を決めることができないとき、従来のテスト生成手法では次の 4 通りの扱いをしていた。

- (1) クロック系回路の故障はテスト生成対象外としてあらかじめ除外する<sup>3)</sup>。この場合、最初から意識してクロック系回路のテスト生成を行わないため、クロック系回路の故障を検出できる保証はない。
- (2) テスト生成を試みるが、テスト生成できないため未検出故障としてあきらめる。この場合、故障検出率は低くなる。さらにこのような故障が未検出故障として残ると、多くの故障イベントが発生するため、故障シミュレーション時間も増大する。

<sup>†</sup> 三菱電機株式会社 システム LSI 開発研究所  
System LSI Laboratory, Mitsubishi Electric Corporation

<sup>††</sup> 三菱電機株式会社 マイコン ASIC 事業統括部  
Microcomputer & ASIC Division, Mitsubishi Electric Corporation

- (3) すべての記憶素子に値を設定後、テスト生成を行う<sup>4)</sup>。この手法の場合、故障回路でも記憶素子に値が設定できると仮定してテスト生成している。しかし、実際には故障回路ではすべての記憶素子に値が設定できるとは限らないため、本来未検出故障であるものを検出故障と判定している可能性がある。
- (4) ポテンシャルディテクション法<sup>5)</sup>でテスト生成する。この手法は故障回路での外部出力の値が決まらなくても、正常回路で外部出力の値が決まれば故障検出可能とするものである。つまり、故障の影響  $0/X$  または  $1/X$  が外部出力に伝播するとき、その故障を検出可能とする。ここに  $0/X$  は、回路中のある信号の値が、ある時刻に正常回路では  $0$ 、故障回路では  $X$  となることを示す。 $1/X$  についても同様である。しかし、実際には故障回路での外部出力の値は不明(すなわち  $X$ )であり、本当に故障検出できるかどうか定かでない。

以上のように従来手法では、順序回路のクロック系回路のテスト生成に対して、故障検出率が下がったり、本来検出できない故障を検出可能と判定することがあった。そこで我々は、このような問題を解決するため、故障の影響により記憶素子に値が設定できなくなる故障を検出するテスト生成および故障シミュレーション方式を開発した。故障の影響により値が設定できなくなる記憶素子  $n$  に、故障回路での固有な初期値  $X_n$  を与え、故障の影響  $0/X_n$  および  $1/X_n$  の双方を観測することにより、従来検出できなかったクロックを OFF にする故障を検出する。

本論文では、まず2章で固有初期値を用いた故障検出の考え方を述べる。次に3章で部分スキャン回路を扱うためのテスト生成モデルについて述べ、4章ではテスト生成および故障シミュレーションで使用する信号値について述べる。5章および6章では固有初期値を用いたテスト生成および故障シミュレーション方式について述べる。7章で本手法を非スキャン同期回路5品種および部分スキャン回路7品種に適用した結果について述べる。

## 2. 固有初期値

順序回路では通常、クロックが存在し、さらにそのクロックがゲート信号(クロック制御回路)によりゲートされることがよくある。このような回路で、クロック系回路に故障が存在し、その故障の影響により記憶素子にクロックが入らなくなると、故障回路では記

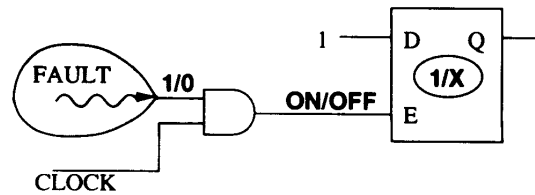


図1 クロック系回路の故障の影響  
Fig. 1 Fault effect in clock control logic.

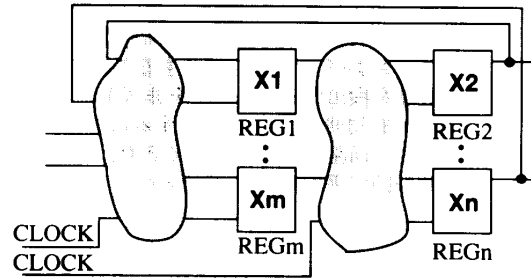


図2 固有初期値  
Fig. 2 Individual initial value.

憶素子の内部状態を永久に設定できず、その故障を検出するテストパターンが生成できないという問題があった(図1)。

この問題を解決するために導入したのが固有初期値という新しい信号値である<sup>6)~8)</sup>。故障の影響により記憶素子にクロックが入らなくても、実デバイスでの各記憶素子はそれぞれ固有な初期値を持っており、その値を固有初期値  $X_n$  と定義する。故障の影響でクロックが入らない記憶素子が  $n$  個あれば固有初期値は  $X_1, X_2, \dots, X_n$  までである(図2)。  $X_n$  は記憶素子  $n$  に固有な初期値であり、その初期値は  $0$  か  $1$  が分からないが、故障回路では、その値は永久に変わらないという性質を持っている。

この固有初期値を利用して、クロック系回路の故障検出を行う。次の4つの条件のうち1つでも満足されると故障は検出できると判定する。

- (1) 故障の影響  $0/X_n$  (正常回路の値/故障回路の値) および  $1/X_n$  の双方が外部出力に伝播するとき
- (2) 故障の影響  $0/\sim X_n$  ( $\sim X_n$  は  $X_n$  の反転値) および  $1/\sim X_n$  の双方が外部出力に伝播するとき
- (3) 故障の影響  $0/X_n$  および  $0/\sim X_n$  の双方が外部出力に伝播するとき
- (4) 故障の影響  $1/X_n$  および  $1/\sim X_n$  の双方が外部出力に伝播するとき

図3を用いて固有初期値を使った、故障検出の例を説明する。時刻  $i$  で故障の影響 ON/OFF が記憶素子

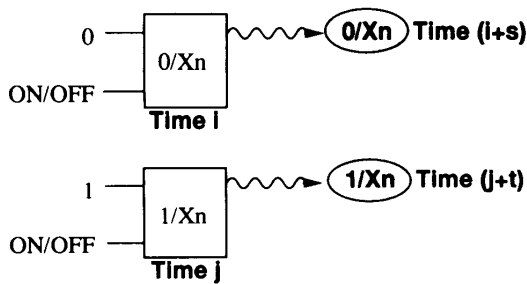


図3 固有初期値を利用した故障検出

Fig. 3 Fault detection using individual initial value.

$n$  のクロック端子に伝播している。そのためクロック印加後、正常回路では記憶素子  $n$  にクロックが入るが、故障回路ではクロックが入らない。記憶素子  $n$  のデータ入力  $0$  なので、クロック印加後、正常回路では値  $0$  が、故障回路では固有初期値  $X_n$  が記憶素子  $n$  の出力に伝播する。したがって、記憶素子  $n$  の出力には故障の影響  $0/X_n$  が伝播する。時刻  $j$  でも故障の影響 ON/OFF が記憶素子  $n$  のクロック端子に伝播しているが、記憶素子  $n$  のデータ入力は  $1$  なので、クロック印加後、正常回路では値  $1$  が、故障回路では固有初期値  $X_n$  が記憶素子  $n$  の出力に伝播する。したがって、記憶素子  $n$  の出力には故障の影響  $1/X_n$  が伝播する。時刻  $i$  での記憶素子  $n$  の出力値  $0/X_n$  を  $s$  タイムフレーム後外部出力で観測し、時刻  $j$  での記憶素子  $n$  の出力値  $1/X_n$  を  $t$  タイムフレーム後外部出力で観測することにより、記憶素子  $n$  の実デバイスでの初期値が  $1$  のとき、時刻  $(i+s)$  で故障の影響  $0/1$  を観測できる。記憶素子  $n$  の実デバイスでの初期値が  $0$  のとき、時刻  $(j+t)$  で故障の影響  $1/0$  を観測できる。したがって、記憶素子  $n$  の初期値が  $1$  であっても  $0$  であってもクロック制御回路のテストが可能となる。

この例は上記 (1) の条件を満足するものであるが、他の場合も同様である。

### 3. テスト生成モデル

テスト生成は部分スキャン設計を扱う。外部入力端子群 PI, 外部出力端子群 PO, システムクロック端子群 CLOCK, 非スキャンレジスタ群  $FF_1, \dots, FF_n$ , スキャンレジスタ群  $SRL_1, \dots, SRL_n$  を持つ部分スキャン設計回路は、テスト生成に際して図 4 のようにモデル化される。回路中のスキャンレジスタ SRL はシステムデータ入力端子 D, システムクロック端子 CLK, スキャン入力端子 SI, スキャン出力端子 SO を持った TSRL<sup>9)</sup> に変換される。各 TSRL のスキャン入

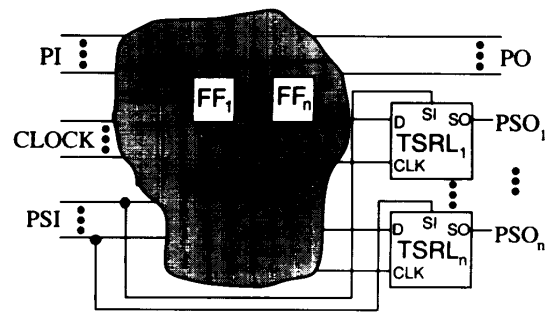


図4 テスト生成モデル

Fig. 4 Test generation model.

表1 信号値  
Table 1 Signal values.

0	: 論理値 0
1	: 論理値 1
Z	: ハイインピーダンス
X	: 不定値
$X_n$	: 記憶素子 $n$ の固有初期値
$\sim X_n$	: 記憶素子 $n$ の固有初期値の反転値
C	: ポジティブクロックパルス
$\sim C$	: ネガティブクロックパルス

力端子 SI は疑似入力端子群 PSI に接続され、スキャン出力端子 SO は疑似出力端子群 PSO に接続される。すなわち、テスト生成は入力端子群 PI, PSI, 出力端子群 PO, PSO, クロック端子群 CLOCK を持つ同期順序回路を対象に行われる。ここで、クロック端子群 CLOCK は 1 相あるいは多相クロックであり、システムデータによるクロック抑止を許し、FF および TSRL のクロック端子に供給される。

### 4. 信号値

表 1 にテスト生成および故障シミュレーションで使用する信号値を示す。信号値  $0, 1, X, Z$  は正常回路および故障回路で使用される。固有初期値  $X_n, \sim X_n$  は故障回路でのみ使用される。クロック信号  $C, \sim C$  は正常回路での外部クロック入力端子にのみ使用される。テスト生成で外部クロック入力端子に割り付けられた信号値  $1$  は  $C$  に変換され、外部クロック入力端子に割り付けられた信号値  $0$  は  $\sim C$  に変換される。

表 2 に固有初期値  $X_m, \sim X_m, X_n, \sim X_n$  を含む 2 入力 NAND ゲートの真理値表を示す。

### 5. テスト生成

テスト生成は以下の手順で行う。

#### STEP1: 初期化

正常回路および故障回路でのすべての信号の値を不定値  $X$  にする。

表2 2入力NANDゲートの真値表  
Table 2 Truth table of 2 input NAND gate.

	0	1	Z	X	$X_m$	$\sim X_m$	$X_n$	$\sim X_n$
0	1	1	1	1	1	1	1	1
1	1	0	X	X	$\sim X_m$	$X_m$	$\sim X_n$	$X_n$
Z	1	X	X	X	X	X	X	X
X	1	X	X	X	X	X	X	X
$X_m$	1	$\sim X_m$	X	X	$\sim X_m$	1	X	X
$\sim X_m$	1	$X_m$	X	X	1	$X_m$	X	X
$X_n$	1	$\sim X_n$	X	X	X	X	$\sim X_n$	1
$\sim X_n$	1	$X_n$	X	X	X	X	1	$X_n$

### STEP2: 故障挿入

まだテスト生成の対象としていない未検出故障を1つ選び、それを故障回路に挿入し、故障信号値で含意操作を行う。含意操作の結果、故障回路で非スキャン記憶素子のクロックがOFFになればその記憶素子  $n$  に固有初期値  $X_n$  を割り付け、さらに固有初期値  $X_n$  で含意操作を行う。

### STEP3: 故障の影響を発生

故障箇所には故障の影響 0/1 あるいは 1/0 を発生させるためのパターンを生成する。このとき、故障箇所を目標信号、故障の影響 0/1 あるいは 1/0 を目標値として回路をバックトレースする<sup>10)</sup>。テスト生成対象回路が部分スキャン回路であるため、非スキャン記憶素子を経由した回路のバックトレースを行い、目標値を設定する。以下、回路のバックトレースの過程で非スキャン記憶素子に達したとき、次の目標値設定をどのようにするか、3つの場合に分けて説明する。

- (1) バックトレースした非スキャン記憶素子のクロックが  $X$  のとき (図5(a))、クロックにはまだ何も設定されていないため、クロックを1 (ON) にすることを新しい目標とする。
- (2) バックトレースした非スキャン記憶素子のクロックが1 (ON) のとき (図5(b))、すでにクロックが印加されているため、記憶素子のデータ入力に記憶素子の出力値を設定することを新しい目標とする。
- (3) バックトレースした非スキャン記憶素子のクロックが0 (OFF) のとき (図5(c))、現在の時刻  $T_i$  ではクロックが入らず記憶素子に値が設定できないため、時刻  $T_i - 1$  で値の設定を試みることを新しい目標とする。さらに、時刻  $T_i - 1$  でのクロックの状態によりその後の目標が設定される。

### STEP4: 故障の伝播

故障の影響を外部出力に伝播させるためのパターンを生成する。対象回路が部分スキャン回路であるため、

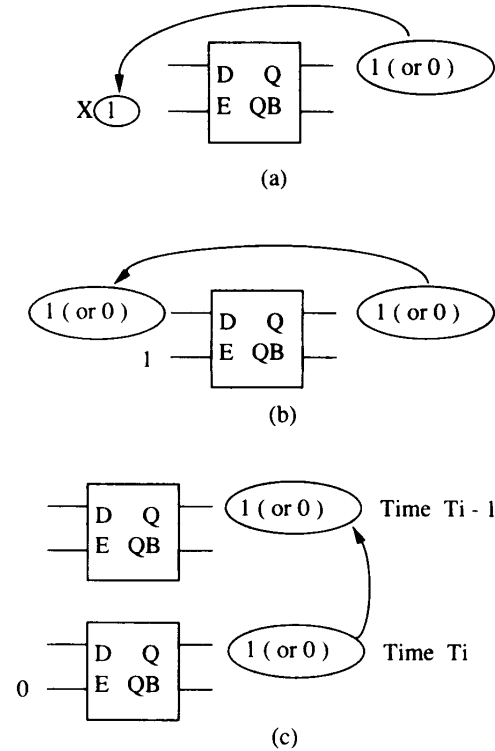


図5 非スキャン記憶素子のバックトレース  
Fig. 5 Backtrace through non-scan registers.

クロックの扱いを考慮する必要がある。故障の影響が非スキャン記憶素子に伝播してきたとき、それをさらに出力側に伝播させるにはどうすればよいか、3つの場合に分けて説明する。

- (1) 故障の影響が非スキャン記憶素子のデータ入力に伝播したとき (図6(a))、クロックを印加し、故障の影響を記憶素子の出力に伝播させる。もし、この時刻内で故障信号が外部出力または疑似外部出力まで伝播できないとき、記憶素子に取り込まれた故障の影響を次の時刻で保持するように試みる。そして、保持された故障の影響を次の時刻以降で外部出力または疑似外部出力まで伝播させることを試みる。
- (2) 故障の影響 0/1 が非スキャン記憶素子のクロック

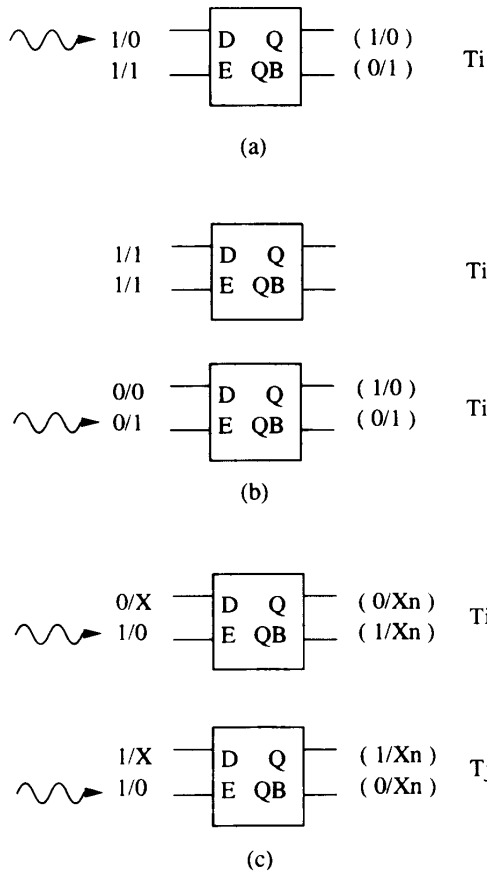


図6 非スキャン記憶素子の故障伝播

Fig. 6 Fault propagation through non-scan registers.

ク入力に伝播したとき (図6(b)), 現在の時刻  $T_i$  で, 正常回路ではクロックが入らないが故障回路ではクロックが入っている. このようとき通常, その前の時刻  $T_i - 1$  で正常回路, 故障回路ともにクロックを印加し, 記憶素子のある値に初期化する. そして時刻  $T_i$  で初期化した値の反転値をデータ入力に設定する. そうすれば, 時刻  $T_i$  で正常回路ではクロックが入らないため時刻  $T_i - 1$  の値を保持し, 故障回路ではクロックが入るため時刻  $T_i$  の値を出力する. 時刻  $T_i - 1$  の値と時刻  $T_i$  の値は異なるため, 故障の影響が発生でき, そしてそれを伝播させればよい.

- (3) 故障の影響  $1/0$  が非スキャン記憶素子のクロック入力に伝播したとき (図6(c)), 故障回路では記憶素子にはクロックを印加することができない. そこで, 故障の影響によって内部状態を初期化できなくなる記憶素子  $n$  に対し, 故障回路での初期値として固有初期値  $X_n$  を割り付ける. その後, 正常回路での信号値  $0$  と故障回路での  $X_n$  を表す故障の影響  $0/X_n$  および

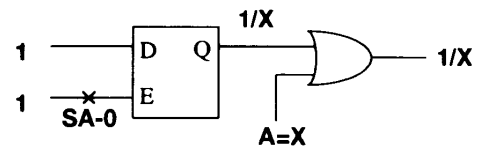


図7 ポテンシャルディテクション

Fig. 7 Potential detection.

正常回路での信号値  $1$  と故障回路での  $X_n$  を表す故障の影響  $1/X_n$  を外部出力または疑似外部出力まで伝播させることを試みる. 故障検出を可能にするための固有初期値を含む故障の影響を出力に伝播させる4つの組合せは, 2章で述べたとおりで,  $0/X_n$  と  $1/X_n$ ,  $0/\sim X_n$  と  $1/\sim X_n$ ,  $0/X_n$  と  $0/\sim X_n$  および  $1/X_n$  と  $1/\sim X_n$  である.

## 6. 故障シミュレーション

従来の多くの故障シミュレータは  $0/X$  あるいは  $1/X$  が外部出力端子に伝播したとき, その故障を検出可能としていた. このポテンシャルディテクション法<sup>5)</sup>は検出不可能な故障を検出可能と判定するかもしれない. 図7のケースで, 非スキャン記憶素子のクロック信号上の0縮退故障は, 記憶素子の初期状態が1のとき, 検出できない. また, シミュレーションで不定値  $X$  になっている信号線  $A$  の実デバイスでの値が1のとき, 故障の影響はマスクされるため検出できない. 一方, ポテンシャルディテクション法が使われないとき, 非スキャン記憶素子のクロックをOFFにする故障は永久に検出できない. そのため, そのような故障がいつまでも故障シミュレーションの対象として残っており, 故障シミュレーション時間の増大にもつながっていた.

故障シミュレーションにおける固有初期値伝播法は,  $0/X_n$  と  $1/X_n$  (ほかに  $0/\sim X_n$  と  $1/\sim X_n$ ,  $0/X_n$  と  $0/\sim X_n$  あるいは  $1/X_n$  と  $1/\sim X_n$ ) の2つの故障の影響を観測したときのみ故障検出と判定するため, 正確な故障検出率の評価が可能となる.

## 7. 実行結果

情報処理装置用開発されたLSI群のうち非スキャン同期回路5品種, 部分スキャン回路7品種, 全スキャン回路2品種に対し, 固有初期値を使うモードと固有初期値を使わないモードでテスト生成を行い, 両者の比較を行った. この評価では, まず未試行未検出故障を1つ選びテスト生成し, 生成されたパターンで

表3 テスト生成結果  
Table 3 Execution results.

	elm	DL	SRL	fit	CLD(dis(forDL))	red	abort	cov(ex_red)	CPU	(succ fail fsim)	patt
回路01	10080	864	0	19781	2370(1140(1140))	114 114	1412 2601	92.29(92.82) 86.27(86.77)	15122 26668	(281 3333 11492) (292 3809 22548)	2165 2282
回路02	9959	392	0	18693	960( 480( 480))	202 202	436 897	96.59(97.64) 94.12(95.15)	4063 5130	(177 1099 2769) (173 1250 3689)	1547 1544
回路03	9013	358	0	18543	1304( 582(582))	18 18	554 827	96.92(97.01) 95.44(95.54)	4196 6742	(348 1377 2455) (320 1486 4920)	2529 2190
回路04	6750	960	0	15437	2658(1226(1226))	7 7	9 1139	99.90(99.94) 92.58(92.62)	6894 12398	(169 30 6682) (164 396 11825)	1564 1530
回路05	6188	490	0	11549	1558( 725( 725))	38 38	231 825	97.67(97.99) 92.53(92.83)	2192 5066	( 86 255 1841) ( 87 389 4579)	2302 2438
回路06	13681	1221	294	22901	7937(3309(1603))	27 27	58 1582	99.63(99.75) 92.97(93.08)	19876 40636	(323 137 19393) (320 839 39455)	3934 3983
回路07	4790	55	312	6338	424( 168( 69))	77 77	9 77	98.64(99.86) 97.57(98.77)	150 202	( 32 16 95) ( 32 26 136)	691 691
回路08	4757	33	279	6389	536( 205( 41))	56 56	6 47	99.03(99.91) 98.39(99.26)	198 247	( 86 6 100) ( 85 13 141)	784 784
回路09	4755	104	303	6838	1707( 736( 124))	45 45	20 143	99.05(99.71) 97.25(97.89)	213 296	( 29 33 144) ( 29 51 208)	663 682
回路10	4447	29	300	5645	498( 155( 36))	54 54	14 49	98.80(99.75) 98.18(99.12)	121 160	( 32 10 73) ( 32 15 106)	784 784
回路11	4195	75	162	6399	536( 229( 88))	27 27	21 108	99.25(99.67) 97.89(98.31)	192 242	( 24 74 88) ( 25 87 124)	512 512
回路12	4185	18	253	5934	834( 348( 22))	23 23	21 41	99.26(99.64) 98.92(99.31)	120 144	( 29 18 67) ( 26 23 88)	537 516
回路13	5755	0	428	7795	742( 371( 0))	739 739	3 3	90.72(99.96) 90.72(99.96)	99 97	( 13 4 74) ( 13 4 72)	331 331
回路14	3381	0	141	7097	321( 106( 0))	209 209	0 0	97.06(100.00) 97.06(100.00)	117 120	( 52 5 55) ( 53 5 57)	619 619

elm	: 素子数 (テスト生成プリミティブ数)
DL	: 非スキャン記憶素子数
SRL	: スキャンレジスタ数
fit	: 代表故障数
CLD	: クロック信号線およびクロック制御回路中の故障数
(dis)	: 上記のうち, クロックを OFF にする可能性がある故障数
(forDL)	: 上記のうち, 非スキャン記憶素子のみに入力して, スキャンレジスタには入力しないもの
red	: 冗長故障数
abort	: バックトラック数制約あるいはテスト周期長制約によりテスト生成に失敗した故障数
cov	: 冗長故障を含めた故障検出率 (単位: %)
(ex_red)	: 冗長故障を含めない故障検出率 (単位: %)
CPU	: 実行 CPU 時間 (単位: 秒 on SPARCstation 2 with SPARC Power uP (57 MIPS))
(succ)	: 上記のうち, 成功したテスト生成にかかった CPU 時間
(fail)	: 上記のうち, 失敗したテスト生成にかかった CPU 時間
(fsim)	: 上記のうち, 故障シミュレーションにかかった CPU 時間
patt	: 生成パターン数

故障シミュレーションを実行し, 検出故障をドロップするという手順を, 未試行あるいは未検出故障がなくなるまで繰り返す方法をとった. また, 1つの故障に対するテスト生成打ち切り条件は, バックトラック数100, テストシーケンス長20とした.

表3は全回路の実行結果で, 各回路の実行結果のうち上段は固有初期値を使うモード, 下段は固有初期値を使わないモードを示している. 表4は, 固有初期値を使うモードと固有初期値を使わないモードで実行

結果が大きく異なる4回路について, クロック信号線とクロック制御回路中の未検出故障の詳細解析結果を示している. クロック系の故障を, まず非スキャン記憶素子だけに伝播する故障とスキャンレジスタに伝播する故障とに分け, 次にクロックを OFF にする可能性がある故障とクロックを ON にする可能性がある故障とに分け, さらにクロック制御回路中の故障とクロック信号線上の故障とに分け, 未検出故障の解析を行った.

表4 クロック系回路の未検出故障  
Table 4 Undetected faults in clock control logic.

	for DL				for TSR			
	disable		enable		disable		enable	
	gate	clk	gate	clk	gate	clk	gate	clk
回路01 total	29	1111	104	1126	0	0	0	0
IIV undet	23	90	0	74	0	0	0	0
NOIIV undet	23	1111	0	90	0	0	0	0
回路04 total	91	1135	249	1183	0	0	0	0
IIV undet	3	0	0	0	0	0	0	0
NOIIV undet	3	1135	0	0	0	0	0	0
回路05 total	59	666	153	680	0	0	0	0
IIV undet	38	25	0	2	0	0	0	0
NOIIV undet	10	666	0	2	0	0	0	0
回路06 total	152	1451	984	1643	1629	77	1936	65
IIV undet	12	1	24	0	6	0	22	0
NOIIV undet	92	1451	24	0	5	0	17	0

total : クロック系論理の総故障数  
 IIV undet : 固有初期値を用いた場合の未検出故障数  
 NOIIV undet : 固有初期値を用いない場合の未検出故障数  
 for DL : 非スキャン記憶素子だけに伝播するクロック系の故障  
 for TSRL : スキャンレジスタのみか、または非スキャン記憶素子とスキャンレジスタの両方に伝播するクロック系の故障  
 disable : クロックを OFF にする可能性がある故障  
 enable : クロックを ON にする可能性がある故障  
 gate : クロック制御回路中の故障  
 clk : クロック信号線上の故障

## 7.1 故障検出率

すべての非スキャン同期回路および部分スキャン回路において、固有初期値を使うモードの故障検出率の方が、固有初期値を使わないモードの故障検出率を上回っている。固有初期値による故障検出率の改善は、非スキャン同期回路で平均 4.49%、部分スキャン回路で 1.78% である。最も効果があった非スキャン同期回路は回路 04 で、故障検出率が 92.58% から 99.90% に改善された。部分スキャン回路では回路 06 の故障検出率が 92.97% から 99.63% に改善された。故障検出率の改善効果は非スキャン記憶素子を駆動するクロック系論理が多いものほど大きい。また、スキャン化率（回路 06 のスキャン化率は 19.4%）の小さいものほど故障検出率の改善効果が大きいことが分かる。

表 4 は、固有初期値を使うモードと固有初期値を使わないモードで、クロック系の未検出故障がどのように違うかを示している。クロック系の故障がスキャンレジスタに伝播するとき、そのスキャンレジスタにはスキャンパス経由で初期値を設定できるため、その故障がクロックを OFF させるものであっても、故障を検出できる可能性が高い。表 4 に示すように回路 06 でスキャンレジスタのクロックに伝播する故障は固有初期値を使わなくても、ほとんど検出できている。非スキャン記憶素子のクロックに伝播する故障のうち、

クロックを ON にするものも固有初期値を使わなくても、ほとんど検出できている。一方、非スキャン記憶素子のクロックに伝播する故障のうち、クロックを OFF にするクロック信号自身の故障は固有初期値を使わなければ検出できないことを示している。

## 7.2 実行時間

すべての非スキャン同期回路および部分スキャン回路に対し、固有初期値を使うモードの実行時間の方が、固有初期値を使わないモードの実行時間より少ない。固有初期値による実行時間の削減は、非スキャン同期回路で平均 41%、部分スキャン回路で 25% である。最も効果があった非スキャン同期回路は回路 05 で、実行時間が 5066 秒から 2192 秒に、57% 削減された。部分スキャン回路では回路 06 の実行時間が 40636 秒から 19876 秒に、51% 削減された。非スキャン記憶素子を駆動するクロック系論理が多いものほど実行時間の削減効果が大きい。また、スキャン化率の小さいものほど実行時間の改善効果が大きいことが分かる。実行時間を、テスト生成が成功したときに費やした時間、テスト生成が失敗したときに費やした時間、故障シミュレーションに費やした時間に分けて解析した。テスト生成が成功したときに費やした時間は固有初期値を使う場合と使わない場合とで大差はない。一方、テスト生成が失敗したときに費やした時間は固有初期値

を使わないモードの方が圧倒的に大きい。これは固有初期値を使わなければ本来検出できない故障に対し、固有初期値を使わないモードで何回もテスト生成を試み、最終的にテスト生成に失敗するためである。しかし、実行時間全体からみれば、あまり影響を及ぼしていない。実行時間で最も影響を及ぼしているのは故障シミュレーション時間である。故障シミュレーションに費やした時間は、固有初期値を使わないモードの方が固有初期値を使うモードの約2倍もの実行時間を費やしている。これは固有初期値を使わなければ本来検出できないクロック系の故障が固有初期値を使わないため、未検出故障として実行終了まで故障シミュレーションの対象として残るためである。

### 7.3 テストパターン数

固有初期値を使うモードと固有初期値を使わないモードでのテストパターン数はほとんど変わらない。

### 7.4 固有初期値伝播法によるオーバーヘッド

固有初期値を導入することによるオーバーヘッドについて述べる。使用メモリーに関しては、従来の信号値(0, 1, Z, X)も固有初期値も1ワードで表現しているため使用メモリーのオーバーヘッドはない。実行時間に関しては、信号値が固有初期値であるかどうかを判定する部分でやや増加するが、それは全体から見ればごくわずかなので実行時間が大きく増加することはないと考えられる。クロック系論理の少ない回路(回路08, 回路10, 回路12)に対しても固有初期値を使うモードの実行時間が、固有初期値を使わないモードより少ないこと、および固有初期値を使う意味がない全スキャン回路(回路13, 回路14)に対して固有初期値を使うモードと使わないモードで実行時間にほとんど差がないことにより、固有初期値の実行時間に対するオーバーヘッドはほとんどないといえる。

以上の実行結果より、固有初期値を使ったテスト生成は、非スキャン同期回路および部分スキャン回路に対し、故障検出率の向上および実行時間の短縮に非常に効果的であることが分かる。

## 8. おわりに

本論文では、非スキャン同期回路および部分スキャン回路のクロックをOFFにする故障に対するテスト生成手法および故障シミュレーション手法について述べた。クロック系の故障の影響により初期化できなくなる非スキャン記憶素子に、それぞれ固有な初期値を割り当てることにより、従来検出不可能であった故障を検出できることを示した。非スキャン同期回路5品種、部分スキャン回路7品種に対する評価により、本

固有初期値伝播法を採用したテスト生成および故障シミュレーションは故障検出率の向上および実行時間の短縮に非常に効果的であることを示した。

著者等は、本論文で報告した手法について1986年より逐次報告してきたが<sup>(6)~(8)</sup>、1991年に、正常回路の初期化が不可能な場合をも考慮した多重観測時刻法が報告された<sup>(11),(12)</sup>。産業界で使われる実回路でこのような場合を考慮する必要があるとは考えにくいだが、機会があれば両手法の性能評価を行って報告したい。

**謝辞** 本手法を用いた初版のプログラムの開発・評価を担当いただいた故猿山秀一氏に感謝します。本論文をまとめるにあたり、多重観測時刻法等についての適切なコメントをいただいた査読委員および終始励ましをいただいた三菱電機システムLSI開発研究所長堀場康孝博士に感謝します。また、テスト生成手法/システムの研究・開発にあたり継続的にご指導いただいた大阪大学樹下三教授に感謝いたします。

## 参考文献

- 1) Funatsu, S., Wakatsuki, N. and Arima, T.: Test Generation System in Japan, *Proc. 12th Design Automation Symposium*, pp.114-122 (1975).
- 2) Eichelberger, E.B. and Williams, T.W.: A Logic Design Structure for LSI Testing, *Proc. 14th Design Automation Conference*, pp.462-468 (1977).
- 3) 辺保 久, 牛久保政憲, 菊地原秀行, 村上道郎: テスト不能故障抽出のための一手法, 情報処理学会設計自動化研究会報告, 87-DA-40, pp.79-84 (1987).
- 4) Shteingart, S., Nagle, A.W. and Grason, J.: RTG: Automatic Register Level Test Generator, *Proc. 22nd Design Automation Conference*, pp.803-807 (1985).
- 5) Thompson, E.W. and Szygenda, S.A.: Digital Logic Simulation in a Time-Based, Table-Driven Environment - Part2, Parallel Fault Simulation, *Computer*, Vol.8, No.3, pp.38-49 (1975).
- 6) 猿山秀一, 荻原拓治, 村井真一: レジスタファイルを含む論理回路のテストパターン自動生成, 第33回情報処理学会全国大会論文集, pp.2289-2290 (1986).
- 7) 猿山秀一, 荻原拓治, 村井真一: ALTES/RA: レジスタファイルを含む論理回路のテストパターン自動生成プログラム, 情報処理学会設計自動化研究会報告, 86-DA-35-3 (1986).
- 8) Ogihara, T., Saruyama, S. and Murai, S.: Test Generation for Sequential Circuits Using Individual Initial Value Propagation, *Digest of*



Technical Papers, 1988 International Conference on Computer Aided Design, pp.424-427 (1988).

- 9) Ogihara, T., Murai, S., Takamatsu, Y., Kinoshita, K. and Fujiwara, H.: Test Generation for Scan Design Circuits with Tri-State Modules and Bidirectional Terminals, *Proc. 20th Design Automation Conference*, pp.71-78 (1983).
- 10) Goel, P.: An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits, Digest of Papers, *The 10th International Symposium on Fault-Tolerant Computing*, pp.145-148 (1980).
- 11) Pomeranz, I. and Reddy, S.M.: Test Generation for Synchronous Sequential Circuits Using Multiple Observation Times, Digest of Papers, *The 21st International Symposium on Fault-Tolerant Computing*, pp.52-59 (1991).
- 12) Pomeranz, I. and Reddy, S.M.: The Multiple Observation Time Test Strategy, *IEEE Trans. Comp.*, Vol.41, No.5, pp.627-637 (1992).

(平成7年11月17日受付)

(平成8年1月10日採録)



村井 真一 (正会員)

1940年生。1963年京都大学工学部電気工学科卒業。1973年イリノイ大学計算機科学科修士課程修了。1963年三菱電機(株)入社。計算機の設計・開発・企画を経て論理LSI CADツールの研究・開発に従事。現在同社システムLSI開発研究所主管技師長。著書:“Advances in Computer Aided Design”(共著, North Holland)等。電子情報通信学会, IEEE各会員。



荻原 拓治 (正会員)

1951年生。1974年成蹊大学工学部電気工学科卒業。同年三菱電機(株)入社。1976年よりLSI CADの研究・開発に従事。主な研究テーマはテスト容易化設計, テスト生成。現在, 三菱電機(株)マイコンASIC事業統括部にてアナ/デジ混在LSI対応CAD開発に従事。IEEE会員。



米森 玄一 (正会員)

1963年生。1986年東北大学工学部通信工学科卒業。同年三菱電機(株)入社。以来, 同社カスタムLSI設計技術開発センターおよびシステムLSI開発研究所において, LSIのテストデータ生成プログラムおよびテストデータ検証システムの開発と適用に従事, 現在に至る。LSIの故障解析およびマイクロプロセッサの設計にも興味を持つ。