

パス遅延故障の過剰テストを削減するためのテストパターン生成法

A Scheme of Test Pattern Generation to Reduce Over-testing of Path Delay Faults

古本 圭†
Kei Furumoto

吉川 祐樹†
Yuki Yoshikawa

1. まえがき

近年、VLSI の動作速度や性能はますます向上する中で、回路のタイミング不良の原因となる遅延故障のテストは重要な課題となっている。しかし、順序回路において遅延故障を検出するためのテストパターン生成は一般に難しく、膨大な時間を要しても高い故障検出率は得られない。そこでテストパターン生成を容易にするために、FF の値を外部から制御、観測できるスキャン設計が用いられる。スキャン設計された回路に対する遅延故障のアプローチとして、ブロードサイドテスト[1]が提案されている。

ブロードサイドテストは、検出困難な故障に対するテストを容易にすることで、高い故障検出率を達成することができる。しかし、スキャン動作によって通常動作では起こり得ない回路状態にも設定できてしまうため、回路の機能に影響しない故障(以下、機能的冗長故障と呼ぶ)を検出してしまふ可能性がある。テストにおいて回路機能に影響しない機能的冗長故障を検出することは、検出すべきでない故障まで過剰に検出してしまうという意味で過剰テスト[2]とよばれており、歩留まりの低下を招く。

本研究では、ブロードサイドテストにおけるパス遅延故障の過剰テスト削減を目的としたテストパターン生成法を提案する。パス遅延故障に対するテストは、回路の微小遅延を検出できる高品質なテストである。ただし、パス遅延故障には機能的冗長故障が多く存在する。そのため過剰テストの削減は重要な課題である。

2. パス遅延故障のテスト

2.1 パス遅延故障

回路におけるパスとは、記憶素子であるフリップフロップ(FF)を始点とし AND, OR などの論理ゲートを通り FF を終点とする経路のことを言う。パス遅延故障とはパス上の累積遅延がある遅延値(一般にはクロック周期)を超える故障である。

パス遅延故障のテストパターンは、回路を初期化するための初期化パターン v_1 と、パスの始点に信号遷移(0→1 もしくは 1→0 の変化)を起こしパスの終点までその信号遷移を伝搬する活性化パターン v_2 からなる。例えば、図 1 の回路のパス(FF2, g6, g7, FF1)について、始点での立ち上がり遷移(0→1)が遅れるパス遅延故障 f が存在する場合を考える。ここでの記号 X はドントケアであり、その信号線の値は 0, 1 どちらでも良いことを表現している。ある時刻 t に(P11, P12, P13, P14, FF1, FF2)の値がそれぞれ(X , X , X , X , 0, 0)で、時刻 $t+1$ で(X , X , 0, 1, 0, 1)とすると、時刻 $t+1$ で FF2 の出力に 0 から 1 の立ち上がりの遷移が起こり、その遷移はパスの終点の FF1 に取り込まれる。

図 2 に示すように、故障がない正常時には遷移が遅れることなく伝搬し、FF1 は 1 を取り込むが、故障時には遷移が遅れて伝搬し、FF1 に 0 が取り込まれる。

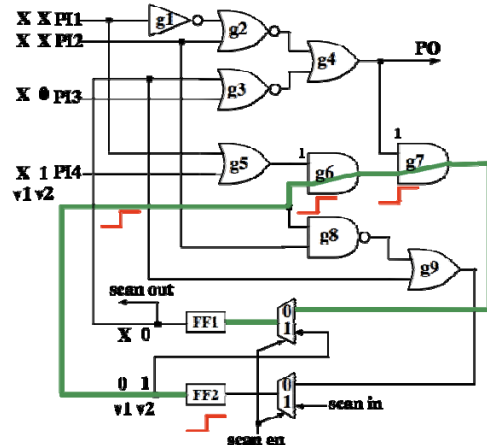


図 1 スキャン設計した順序回路

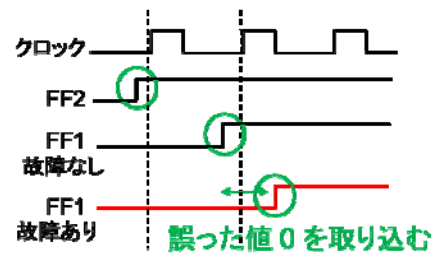


図 2 故障ありと故障なしの信号伝搬波形

2.2 ブロードサイドテスト[1]

順序回路は、記憶素子である FF の値を外部から直接制御することや観測することはできない。そこで回路のテストを容易にするための 1 つの方法として、図 1 に示すようなスキャン設計がある。スキャン設計を行った回路では、スキャン信号(scanenable)を 1 にすることでスキャンインから各 FF へ値を設定することができる。また、スキャンアウトでは各 FF の値を観測することができる。このようなスキャン設計された回路を利用した遅延故障のテスト方法としてブロードサイドテストが提案されている。先のパス(FF2, g6, g7, FF1)について、始点の遷移が立ち上がりのパス遅延故障 f を例にブロードサイドテストの説明を行う。図 3 は図 1 の順序回路を 2 時刻展開した回路を示している。故障 f に対するブロードサイドテストでは、時刻 t の(FF1, FF2)に対する初期化パターン(X , 0)をスキャンインからスキャンシフトによって設定する。

†呉工業高等専門学校, Kure National College of Technology

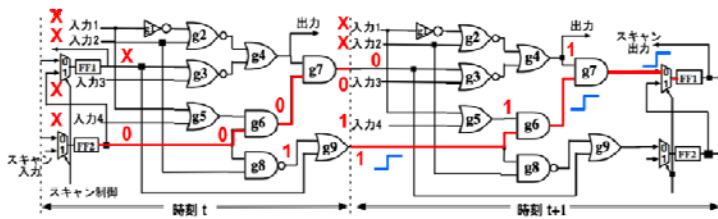


図3 ブロードサイドテストの例

この例では、初期化パタンは(X, X, X, X, X, 0)となる。時刻 t+1 の FF1, FF2 の活性化パタンは、初期化パタンの出力(時刻 t のゲート g7 と g9 の出力 0 と 1)を利用する。この例では、活性化パタンは(X, X, 0, 1, 0, 1)となる。そして、時刻 t+1 で FF1 と FF2 に取り込まれた値をスキャンシフトすることで外部へ出力する。これにより故障 f の影響を観測することができ、故障を検出することができる。この初期化パタンと活性パタンのペア(v1, v2)を故障 f のテストパタンという。

3. 過剰テスト

回路にはたとえ存在したとしてもその回路動作に影響しない故障が存在する。このような故障を機能的冗長故障という。機能的冗長故障は通常動作の範囲では回路の出力に影響しないため、故障として検出されることはない。しかし、スキャン設計などのテスト容易化設計を行った回路では、通常動作では起こらない回路状態に設定できるため、回路動作に影響しない機能的冗長故障を検出してしまふ可能性がある。これを過剰テストという。

例えば、図1におけるパス(FF1, g9, FF2)上のパス遅延故障gを機能的冗長故障とする。故障fを見つけるためのテストパタンv1=(X, X, X, X, 1, 0), v2=(X, X, 0, 1, 0, 1)でブロードサイドテストを行うとき、同時に故障gも検出される。つまり、この回路において故障gは存在しても正常に動作するにも関わらず、gを検出することによりこのチップは不良品として判定される。

4. 過剰テスト削減のためのテストパタン生成

4.1 テストパタンと故障検出の関係

ある故障を検出するためのテストパタンは、その故障だけではなく他の故障も検出する可能性がある。また、故障を検出するテストパタンは1つではなく複数存在する。

例えば表 1 に示すように、各テストパタン t1, t2, ..., t6 とそれぞれの故障が検出する故障の関係があるとする。この表ではテスト対象故障集合を $F_t = \{f1, f2, f3, f4, f5, f6\}$ とし、機能的冗長故障の集合を $F_u = \{f7, f8, f9, f10\}$ とする。ただしこれ以降、機能的冗長故障は非テスト対象故障と表現する。ここで、テスト対象故障集合 F_t に属する全ての故障を検出するテストパタン集合 $T1 = \{t1, t3, t6\}$ と $T2 = \{t1, t4, t5\}$ を考える。それぞれのパタン集合 T1 と T2 が検出する非テスト対象故障 F_u は以下ようになる。集合 T1 が検出する非テスト対象故障は f7, f8, f9, f10 の 4 つであるのに対し、集合 T2 は f7, f8 の 2 つのみを検出する。

このように、故障集合の作り方によって非テスト対象故障の検出数が異なることから、個々のテストパタンの

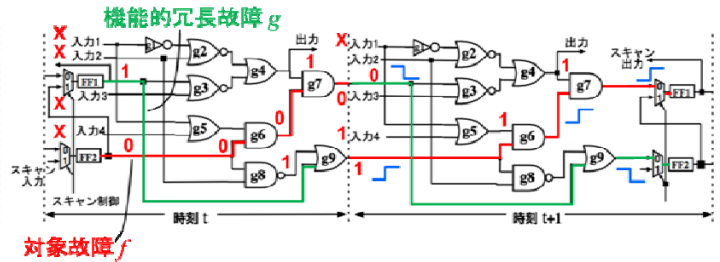


図4 過剰テストの例

表 1 各パタンの故障検出の関係

	テスト対象故障 F_t						非テスト対象故障 F_u			
	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10
t1	○	○					○	○		
t2		○		○		○			○	
t3			○	○				○		○
t4			○	○		○		○		
t5					○		○	○		
t6					○	○			○	○

検出故障だけを考えるのではなく、テスト集合全体でどの非テスト対象故障を検出するかを考慮する必要がある。

4.2 過剰テスト削減のためのテストパタン生成

以下に、対象とする問題を示す。

問題

入力：対象回路 C, 回路 C におけるテスト対象故障集合 F_t , 非テスト対象故障集合 F_u 。

出力：対象故障集合 F_t の中で検出可能な故障を全て検出するテストパタン集合 T。

最適化目標：非テスト対象故障集合の検出故障数最小化。

この問題に対するテストパタン生成のアプローチとして、大きく 2 つの方法が考えられる。1 つは、各故障に対してパタン生成を行う際、必須割当てをどのように入力側へトレースするかに着目し、ドントケアを含むパタンを生成するときに、非テスト対象故障の検出数が少ないパタンを生成するためのアプローチが考えられる。2 つ目は、既に生成されたドントケアを含むパタン集合について、非テスト対象故障の検出数を最小にするためのドントケアへの 0,1 の割当て方法である。この論文ではまず後者のドントケア割当てに着目し、非テスト対象故障の検出数を最小にする手法を提案する。

以下に、生成したテストパタン集合のドントケアの埋め方について、非テスト対象故障の検出数を最小化するための戦略を述べる。

各パタンはできるだけ非対象故障の検出数 $|F_{ud}|$ が少なく、かつ対象故障の検出数 $|F_{td}|$ が多くなるようにドントケアに 0, 1 を割り当てる。

あるパタンで検出された故障 f ($F_t \cup F_u$) は、それ以降のパタン生成時にドントケア割り当ての尺度計算から除くようにする。

1 つ目の戦略は、各パタンが検出する非対象故障 F_u の検出数削減だけではなく、全体のテストパタン数を減らすために各パタンの F_t の検出数を多くすることに注目している。各パタンが検出する対象故障数が多くなれば、テストパタ

ン集合 T のボタン数が少なくなり、結果として非対象故障の検出数を減らすことができると考えられる。

一方、2つ目の戦略は、あるボタンで1度でも検出された故障は、それ以外のボタンによる検出/非検出にかかわらず全体のボタン集合 T で検出されることに着目している。つまり、その時点でまた未検出の F_u の故障を非検出にすること、未検出の F_t の故障を検出することを重視した戦略である。

提案するテストボタン生成アルゴリズムの概要を以下に示す。

ステップ1. $F_t = \varnothing$ ならば手続きは終了し、そうでない場合、テスト対象故障集合 F_t から故障 f を1つ選択する。

ステップ2. 選択した故障 f に対してドントケアを埋めていないテストボタン t を生成する。

ステップ3. ボタン t のドントケアへの割り当てを評価する尺度にしたがって、ドントケアを埋めたボタン t' を生成する。

ステップ4. 生成したボタン t' を T に追加し、そのボタンが検出するテスト対象故障と非テスト対象故障を故障シミュレーションで求め、検出される故障を F_t と F_u から取り除き、ステップ1へ移る。

ステップ1では、テスト対象故障集合 F_t からテストボタン生成の対象故障 f を1つ選択する。このとき、 F_t が $\varnothing$ であれば全てのテスト対象故障に対してボタン生成が終了したことを意味するため手続きを終了する。次にステップ2では、選択した故障 f に対してドントケアを埋めていない状態のテストボタン t を生成する。ステップ3では、生成したボタン t のドントケアへの割り当てを決めるために、できるだけ非テスト対象故障の検出数 $|F_u|$ が少なく、かつテスト対象故障の検出数 $|F_t|$ が多くなるドントケア割り当てのための尺度を計算し、尺度にしたがって t のドントケアに0または1を割り当てた t' を生成する。この尺度は、上記で述べた戦略の1つ目に対応する。具体的な尺度の説明は後ほど述べる。ステップ4は、上述の戦略の2つ目に対応する。

ステップ3で用いる、できるだけ非テスト対象故障の検出数 $|F_u|$ が少なく、かつテスト対象故障の検出数 $|F_t|$ が多くなるようにドントケアに0, 1を割り当てるための尺度の1つとして、ここでは、生成したボタン t' の非テスト対象故障の検出数 $|F_u(t')|$ とテスト対象故障の検出数 $|F_t(t')|$ のバランスをパラメータ α で調節する評価式を提案する。この評価式をステップ3に埋め込んだ時の具体的なヒューリスティックアルゴリズムを図5に示す。

この手続きではまずテスト対象故障集合 F_t からある故障 f を選択し、 f に対してテストボタン t を生成する。それから、生成したボタン t のドントケアをランダムに埋めた N 個のボタン $t'_1, t'_2, \dots, t'_N$ を生成し、以下の評価式で評価値を計算する。

$$e(t'_i) = \alpha \times |F_u(t'_i)| - (1-\alpha) \times |F_t(t'_i)|$$

この評価式は α を係数とし、ある対象故障 f に対して生成したボタン t のドントケアをランダムに埋めたボタン t'_i ($i=0, 1, \dots, N$)で検出される非テスト対象故障数 $|F_u(t'_i)|$ と、検出されるテスト対象故障数 $|F_t(t'_i)|$ が

```
# Define N 3
# Define  $\alpha$  0.5
# Define eval( $F_a, F_b$ ) ( $\alpha \times |F_a| - (1-\alpha) \times |F_b|$ )
tpg( $F_t, F_u$ ) {
  //  $F_t$  is a set of faults to be tested
  //  $F_u$  is a set of faults to be undetected

   $T = \varnothing$ ;
  While( $|F_t| = \varnothing$ ) {
     $f = \text{target\_from}(F_t)$ ; //  $F_t = F_t - \{f\}$ 
     $t = \text{test\_generation\_for}(f)$ ;
    // if  $f$  is redundant,  $t$  will be nil
    if( $t \neq \text{nil}$ ) {
       $tb = \text{nil}$ ;  $e = |F_u|$ ;
       $F_{tb} = F_{ub} = \varnothing$ ;
      for ( $i=0$ ;  $i < N$ ;  $i++$ ) {
         $tt = \text{randomfill}(t)$ 
        //  $X_s$  in  $t$  are specified randomly
        ( $F_{td}, F_{ud}$ ) =  $\text{fault\_sim\_by\_for}(tt, F_t, F_u)$ ;
        //  $F_{td}$  and  $F_{ud}$  are sets of faults detected
        // by  $tt$  in  $F_t$  and  $F_u$ , respectively.
         $v = \text{eval}(F_{ud}, F_{td})$ ;
        if( $v < e$ ) {
           $tb = tt$ ;  $e = v$ ;
           $F_{tb} = F_{td}$ ;  $F_{ub} = F_{ud}$ ;
        }
      }
       $T = T \cup \{tb\}$ ;
       $F_t = F_t - F_{tb}$ ;  $F_u = F_u - F_{ub}$ ;
    } else {
       $F_t = F_t - \{f\}$ ;
    }
  }
  return  $T$ ;
}
```

図 5 提案するヒューリスティックアルゴリズム

ら計算する。検出される非テスト対象故障数 $|F_u(t'_i)|$ が少なく、また検出されるテスト対象故障数 $|F_t(t'_i)|$ が多いほど評価値 e は小さくなる。また、係数 α は $0 \leq \alpha \leq 1$ であり、 α が大きいときはボタン t'_i で検出される非テスト対象故障数 $|F_u|$ の重みが大きくなり、 $|F_u|$ が少ない方が評価値 e はより小さくなる。つまり、 $|F_u|$ の検出数が少ないボタンを優先して選択する。一方、 α が小さいときは、ボタン t'_i で検出されるテスト対象故障数 $|F_t|$ の重みが大きくなり、 $|F_t|$ が多い方が評価値 e はより小さくなる。つまり、 $|F_t|$ の検出数が多いボタンを優先して選択する。このように、 α によって $|F_u|$ の検出数を減らすことを優先するのか、 $|F_t|$ の検出数を増やすことを優先するのかを調整する。ここでは $\alpha=0.5$ としている。

続いて、ドントケアをランダムに埋めて生成した N 個のボタンの中から $e(t'_i)$ が最小となるものを選択し、それを対象故障 f のテストボタンとして記憶し、故障シミュレーションにより各故障集合 F_t, F_u から検出されるものを取り除く。これにより、今後はすでに検出されたものを取り除いた F_t と F_u に対して、ボタン t' で検出される非テスト対象故障数 $|F_u(t')|$ と、検出されるテスト対象故障数

$|Ftd(t')|$ を故障シミュレーションで求めて評価値 e を求めるため、未検出の非テスト対象故障をより非検出に、未検出のテスト対象故障をより検出するようになる。 $|Fud(t')|$ が小さいということは最適化目標である Fu の検出数最小に直接関係する。また $|Ftd(t')|$ が多いことは生成するパターン数 $|T|$ を減らすことになり、結果として非対象故障の検出数を少なくできる。

上記のアルゴリズムの実行例を示す。図3を用いて説明する。まず、テスト対象故障集合 Ft と、非テスト対象故障集合 Fu が与えられたとき、 Ft の中からある故障 f を選択する。故障 f に対するテストパターン t を生成する。生成されたパターン t は初期化ベクトルが $(PI1, PI2, PI3, PI4, FF1, FF2)=(X, X, X, X, X, 0)$ 、活性化ベクトルが $(PI1, PI2, PI3, PI4, FF1, FF2)=(X, X, 0, 1, 0, 1)$ である。このパターン t のドントケアをランダムに埋めたパターンを N 個生成する。ここでは、 N は3とする。3つのパターン $t1, t2, t3$ はそれぞれ、 $t1$ の初期化ベクトルが $(PI1, PI2, PI3, PI4, FF1, FF2)=(0, 1, 1, 0, 1, 0)$ 、活性化ベクトルは $(PI1, PI2, PI3, PI4, FF1, FF2)=(0, 0, 0, 1, 0, 1)$ 、 $t2$ の初期化ベクトルが $(PI1, PI2, PI3, PI4, FF1, FF2)=(0, 1, 1, 0, 1, 0)$ 、活性化ベクトルは $(PI1, PI2, PI3, PI4, FF1, FF2)=(0, 1, 0, 1, 0, 1)$ 、 $t3$ の初期化ベクトルが $(PI1, PI2, PI3, PI4, FF1, FF2)=(0, 0, 351, 0, 1, 0)$ 、活性化ベクトルは $(PI1, PI2, PI3, PI4, FF1, FF2)=(0, 0, 0, 1, 0, 1)$ である。これらのパターンが検出するテスト対象故障数と非テスト対象故障数を故障シミュレーションで求める。パターンごとの検出するテスト対象故障数と非テスト対象故障数は、 $t1$ の $|Ftd(t1)|$ は4、 $|Fud(t1)|$ は1、 $t2$ の $|Ftd(t1)|$ は4、 $|Fud(t1)|$ は2、 $t3$ の $|Ftd(t1)|$ は4、 $|Fud(t1)|$ は0となる。それぞれのパタンの検出する故障から評価式を用いて、それぞれのパタンの評価値 e を求める。ここでは、 α は0.5とする。3つのパターン $t1, t2, t3$ について評価値 e を求める。まず $t1$ の評価値 e は、以下のように1.5となる。

$$\begin{aligned} e(t1) &= \alpha \times |Fud(t1)| - (1 - \alpha) \times |Ftd(t1)| \\ &= 0.5 \times 1 - (1 - 0.5) \times 4 \\ &= 0.5 - 2 = -1.5 \end{aligned}$$

同様に、 $t2$ の評価値 e は-1、 $t3$ の評価値 e は-2となる。この3つの中で評価値が最も小さいものを選択するため、この例の場合、 $t3$ が選択される。パターン $t3$ によって検出される故障をテスト対象故障集合と非テスト対象故障集合から取り除き、これをテスト対象故障集合 Ft が ϕ になるまで繰り返す。

4. 実験結果

この評価実験では、ベンチマーク回路としてPaulin, Lwf, Tseng, Jwf の4つを対象とし、どのような α のときに非テスト対象故障の検出数が少なくなるのかを調べるために、 $\alpha=0, 0.25, 0.5, 0.75, 1$ のそれぞれについて実験を行った。また、与えられる故障集合によって、同じ回路でも検出する Fu の故障数 $|Fud|$ が最小となる α が同じなのかを調べるために、各回路についてそれぞれ3種類の故障集合を用いて実験を行った。

実験結果の表2より、各パターンについて、非テスト対象故障の検出数を最小($\alpha=1$)にするだけでは、全体の非対象故障の検出数 $|Fud|$ は最小にならない。また、各パターンについて、対象故障の検出数を最大($\alpha=0$)にするだけで

表2 実験結果

回路	対象故障 $ Ft $	非対象故障 $ Fu $	非対象故障の検出数 $ Fud $				
			$\alpha=0$	$\alpha=0.25$	$\alpha=0.5$	$\alpha=0.75$	$\alpha=1$
Paulin	532	1000	187	183	177	167	173
	532	532	167	165	108	108	135
	1000	1998	936	918	888	889	906
Lwf	1458	2918	1821	1721	1756	1777	1793
	1458	1458	605	597	588	598	605
	2918	1458	1403	1359	1353	1341	1366
Tseng	900	1786	239	185	184	183	188
	1343	1343	492	451	450	458	469
	1786	900	343	345	345	339	347
Jwf	1266	2500	440	439	439	438	438
	1883	1883	1292	1273	1296	1304	1314
	2500	1266	897	859	867	867	871

も、やはり全体の非対象故障の検出数 $|Fud|$ は最小にならない。これは、非対象故障の検出数を最小($\alpha=1$)にすると1パターン当たりの非対象故障の検出数は減少するが、テスト対象故障に対するテスト集合が増加するため、テスト集合全体では非対象故障の検出数が増加してしまったものと考えられる。一方、対象故障の検出数を最大($\alpha=0$)にする場合は、テスト集合のパターン数自体は減少することで、非対象故障の検出数を抑えようとしているが、非対象故障の検出数のことを考えていないため、非対象故障の検出数が増加してしまったものと考えられる。

以上のことより、各パタンの非対象故障の検出数と対象故障の検出数のバランス(α)を考えることで、全体の非対象故障の検出数を減らせることが分かる。今回の実験から、 $\alpha=0.5 \sim 0.75$ で Fu の検出数が最小になることが多いことが分かる。このことから、非対象故障の検出数を減らすことに重みを置きながら、対象故障の検出数を増やすことにも配慮する必要がある。また回路の特性などから前もって最適な α をある程度予測する手法が必要であり、これは今後の課題とする。

5. まとめ

本研究では、ブロードサイドテストにおけるパス遅延故障の過剰テスト削減を指向したテストパターン生成法の1つとして、すでに生成されたテストパタンのドントケアの埋め方に着目し、ドントケアをランダムに埋めて、テスト集合全体で非対象故障の検出数 Fud が最小となるようなパターンを選択するための評価式を提案し、実験を行った。実験により、同じ回路でも故障集合が異なると非対象故障の検出数が最小となる α がばらつくことが分かった。

今後の課題としては、より大規模な回路での実験や、回路ごとに最適な係数 α を推測する尺度の提案などが挙げられる。また、パターン生成時に必須割当てをどのように入力側ヘトレースするかという問題にも今後取り組む。

参考文献

- [1] J.Savir and S.Patel "Broad-side delay test," IEEE Trans. on CAD, Vol.13, No.8, pp.1057-1064, Aug.,1994.
- [2] J.Rearick "Too much delay fault coverage is a bad thing," Proc. IEEE International Test Conf., pp.624-632, 2001.
- [3] I.Pomeranz and S. M.Reddy, "On generation tests that avoid the detection of redundant faults in synchronous sequential circuits with fulls can," IEEE Trans. on Comp, Vol.55, No.4, pp.491-495, April, 2006.
- [4] 藤原秀雄, デジタルシステムの設計とテスト, 工学図書株式会社, ISBN4-7692-0459-0, 2004.