社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2007－ＳＬＤＭ－132（１）
2007／11／20

遅延故障テスト容易化FF方式の下での縮退故障テストデータ圧縮法

加藤　健太郎†　　難波　一輝††　　伊藤　秀男††

† 千葉大学大学院自然科学研究科，千葉県千葉市稲毛区弥生町 1-33
† 千葉大学大学院融合科学研究科，千葉県千葉市稲毛区弥生町 1-33
E-mail: †katoh@office.chiba-u.jp, ††{namba,h.ito}@faculty.chiba-u.jp

あらまし　本論文では，遅延故障テスト容易化フリップフロップ（以下 FF と略記）方式の下での縮退故障テストデータ圧縮法を提案する．提案テスト圧縮法は，遅延故障テスト容易化 FF の構造を利用したテストデータ圧縮（１段階目圧縮）を行い，予めテストデータ量を削減する．その後削減されたデータにさらにテストデータ圧縮（２段階目圧縮）をかけることにより，２段階のテストデータ圧縮を行う．評価実験において，ハフマン符号によるテストデータ圧縮法を２段階目圧縮に用いて提案圧縮法を適用した場合，従来法の場合と比較して，ATE に格納するテストデータ量を最大で 37.1%，平均で 26.0%削減できることを確認した．

キーワード　遅延故障テスト容易化 FF，スキャンベクトル長圧縮法，テストデータ量削減，１回スキャンイン操作，２回スキャンイン操作．

Stuck-at Test Data Compression using Scan FFs with Delay Fault Testability

Kentaroh KATOH†, Kazuteru NAMBA††, and Hideo ITO††

† Graduate School of Science and Technology, Chiba University,
1-33, Yayoi-cho, Inage-ku, Chiba-shi, Chiba, 263-8522 JAPAN
† Graduate School of Advanced Integration Sciences, Chiba University,
1-33, Yayoi-cho, Inage-ku, Chiba-shi, Chiba, 263-8522 JAPAN
E-mail: †katoh@office.chiba-u.jp, ††{namba,h.ito}@faculty.chiba-u.jp

Abstract　This paper presents a stuck-at test data compression technique using the scan flip flops with delay fault testability. The propose technique consists of two-phase test data compression. First, the proposed technique compresses the test data volume utilizing the unique structure of the scan flip flop(1st compression process). Second, it compresses again the compresseded test data utilizing X data (2nd compression process). Evaluation shows that the proposed technique for huffman test data compression generates smaller compressed test data for ATE than conventional huffman data compression. The amount of compressed test data for ATE by the proposed technique is less 37.1% in maximum, 26.0% on average than the one by the conventional technique.

Key words　FFs with delay fault testability, scan vector length compression, one step scan-in operation, two step scan-in operation.

1. まえがき

近年のプロセスの微細化・高集積化に伴い，テストコスト削減は急務の課題となりつつある．このため設計段階でテスト容易化を考慮したテスト容易化設計（DFT）が不可欠となりつつある [1]．中でもスキャン法は，ディジタル論理回路において最も広く用いられているテスト容易化設計法の１つである．

チップに実装される回路の大規模・複雑化に伴いテストデータ量は今後ますます増加する傾向にある．このため効率的なテストデータ圧縮法の考案は急務の課題となっている．製造テストでは，論理故障を検出する縮退故障テスト，信号遅延を検出する遅延故障テストが重要である．近年，これらのテストのためのテストデータ圧縮法が提案されている．縮退故障テストデータ圧縮法に関しては，従来符号化法を用いた手法 [2]，Fan-out スキャンチェーンを用いた手法 [3]，Seed Encoding 法 [4]，Template-based Encoding 法 [5] など近年多くのスタンダードスキャン方式を仮定した手法が提案されている．また遅延故障テストパターン圧縮に関しては，エンハンストスキャン方式を仮定した効率的なハフマン圧縮法が提案されている [8]．

一方，難波らは遅延故障テスト容易化 FF（以下，千葉大スキャン FF と略記）[6] を提案した．この FF は冗長ラッチなし

にブロードサイドテストやスキュードロードテスト [7] などスタンダードスキャン FF を用いた手法では実現できなかった任意の 1 ビット遷移遅延故障パターンを入力できる．また，冗長ラッチが必要ないためエンハンストスキャンと比較し面積オーバヘッドが少ない．

本論文では，千葉大スキャン FF 方式を用いた縮退故障テストデータ圧縮法を提案する．提案手法は 2 段階に分けてデータを圧縮するという特徴を持つ．1 段階目の圧縮は，スキャンベクトル長圧縮法と呼ばれる圧縮法を用いる (スキャンベクトル長圧縮法については 3.2 において詳細を述べる)．これは千葉大スキャン FF の構造を利用したテストデータ圧縮であり圧縮後のデータも多くの X ビットを有するという特徴を持つ．このスキャンベクトル長圧縮法によるテストデータ圧縮をスキャンベクトル長圧縮と呼ぶこととする．次に 2 段階目として X ビットを利用したテスト圧縮を行う．このような 2 段階テストデータ圧縮を行う事により，従来圧縮法よりも高い圧縮率を実現する．評価実験において，ハフマン符号によるテストデータ圧縮を提案手法の 2 段階目圧縮として適用した場合，従来法の場合と比較して，ATE に格納するテストデータ量を最大で 37.1%，平均で 26.0%削減できることを確認した．

本論文の構成を以下に示す．2. で準備として千葉大スキャン FF の構造を説明する．3. で提案手法の詳細を説明する．4. で提案手法のテストデータ量の削減について評価を行う．最後に 5. でまとめを述べる．

2. 千葉大スキャン

千葉大スキャンは，冗長ラッチを付加することなく，任意の 1 ビット遷移パターンを入力することができる．テスト時に，マスタラッチのみから構成されるスキャンパスと，独立に制御可能な 4 本のクロック線を用いて 1 ビット遷移 2 パターンベクトルを入力することができる．1 ビット遷移パターンにより，ロバストパス，ノンロバストパス遅延故障の高い検出率を得る事ができる．

図 1 に長さ 4 の 4 つの千葉大スキャン FF により構成されるスキャンチェーンを示す．このスキャンチェーンは，$FF_0 \sim FF_3$ から構成される．また *sdi* はスキャン入力，*sdo* はスキャン出力である．各 FF は，マスタスレーブ FF の構造を有する．図中 $M_0 \sim M_3$ はマスタラッチ，$S_0 \sim S_3$ はスレーブラッチである．各 FF は，通常入力 $d[i](0 \leqq i \leqq 3)$ とテスト入力の 2 つの入力線，通常出力 $q[i]$ とテスト出力の 2 つの出力線を有する．テスト時には，M_0 と M_1，M_2 と M_3 がそれぞれ対となりマスタスレーブ FF を構成し，*sdi* をスキャン入力，*sdo* をスキャン出力とするスキャンパスを構成する．一方通常動作時は，M_0 と S_0，M_1 と S_1，M_2 と S_2，M_3 と S_3 がそれぞれ対となりマスタスレーブ FF を構成する．偶数番目の FF のマスタラッチと奇数番目の FF のマスタラッチは，それぞれクロック線 CLK_{ME}，CLK_{MO} を操作する事により交互に開閉される．シフト動作の際，閉じているラッチの出力側に接続されているラッチは開くため，これら 2 つのラッチの値は同じ値となる．

例として，初期ベクトル $(FF_{S0}, FF_{S1}, FF_{S2}, FF_{S3}) = (0, 1, 0, 1)$ の状態で，FF_0 から 1 ビットの立ち上がり遷移を与える場合を取り上げる．この場合，まず初期ベクトルの偶数ビット $(0, 0)$ を CLK_{ME}，CLK_{MO} の交互の操作によりスキャンインした後，CLK_{SE} の操作を行い，$(S_0, S_2) = (0, 0)$ とする．次に奇数ビット $(1, 1)$ を同様の操作によりスキャンインした後，CLK_{SO} の操作を行い，$(S_1, S_3) = (1, 1)$ とする．これで初期ベクトルはセットされた．そして最後に遷移を発生させたいビットが奇数であれば遷移ベクトルの奇数ビット (偶数であれば偶数ビット) をスキャンインする．この場合は偶数であるので $(FF_0, FF_2) = (1, 0)$ をスキャンインする．そして遷移を発生させたいビットのスレーブラッチを制御するクロック (この例では CLK_{SE}) を制御し，遷移を発生させる．

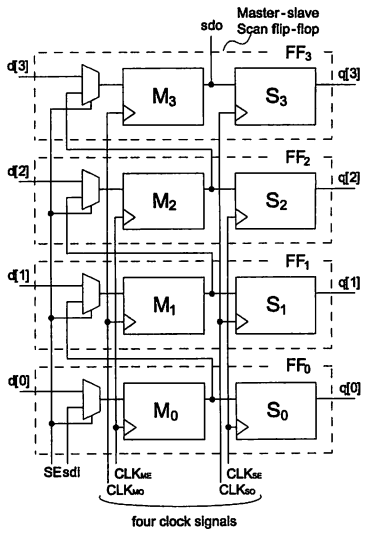

図 1 遅延故障容易化 FF [6].

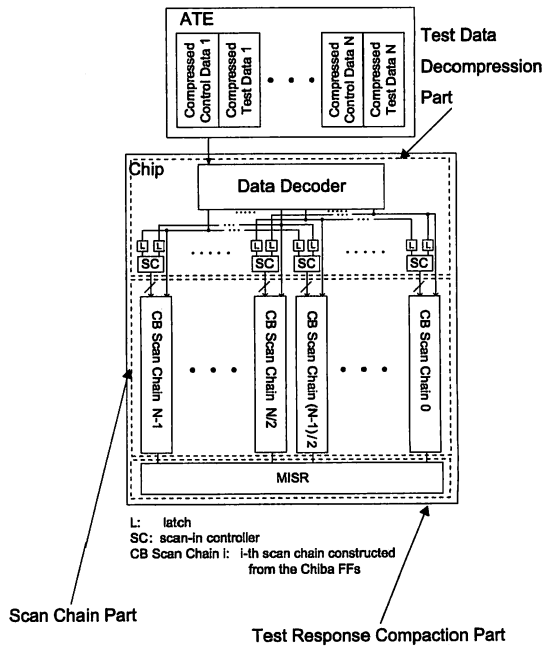

図 2 提案テスト圧縮アーキテクチャ.

3. 提案手法

3.1 概 要

本論文では，千葉大スキャン FF 方式の下での縮退故障テストデータ量圧縮法を提案する．提案手法では，スキャンベクトル長圧縮法による 1 段階目圧縮を行った後に再度 2 段階目圧縮を行う．このような 2 段階の圧縮によるテストデータ量削減を行うことにより従来圧縮法よりも ATE に格納するデータ量を削減する事ができる．

ここで図 2 に提案手法で用いるテストアーキテクチャを示す．この構造は大きく分けて ATE，テストデータ展開部，スキャ

ンチェーン部, テスト応答圧縮部と 4 つの部位に分けることができる．スキャンチェーン部は千葉大スキャン FF から構成される．テスト応答圧縮部は MISR を用いて実装される．テストデータ展開部は，大きく上部のデコーダモジュールと下部のスキャンイン操作制御器から構成される．ATE より圧縮されたデータがテストデータ展開部に入力され，解凍されたテストデータがスキャンチェーンに入力される．

提案テストデータ圧縮法の全体の流れを述べる．まず初めにスキャンチェーン数を決定し，テストデータ集合を分割したスキャンチェーン毎のサブテストデータ集合に分ける．各テストセット毎にスキャンベクトル長圧縮法を適用する．スキャンベクトル長圧縮法の適用により，スキャンインモード制御データが構成される．そして，そのテストパターンとスキャンイン操作制御データの圧縮を行い，ATE に格納するテストデータを構成する．

ここで圧縮前の n 個のテストパターンからなるテストデータ集合を $TP = \{tp_i | 0 \leqq i \leqq n-1\}$，スキャンベクトル長圧縮後のテストデータ集合を $TP' = \{tp'_i | 0 \leqq i \leqq n-1\}$ と定義する．tp_i, tp'_i は，各スキャンチェーン毎のテストデータ集合を含む．またスキャンベクトル長圧縮適用後の各テストパターン tp_i のスキャンイン操作制御データ tc'_i の集合を $TC' = \{tc'_i | 0 \leqq i \leqq n-1\}$ と定義する．ここで $TD' = \{td'_i | td'_i = (tp'_i, tc'_i), 0 \leqq i \leqq n-1\}$ を定義する．また TD'' を ATE へ格納するテストデータとする．そして TP から TD' への関数を $g : TP \longrightarrow TD'$ で定義し，TD' から TD'' への関数を $f : TD' \longrightarrow TD''$ で定義するものとすると，テストデータ圧縮手順は

$$TD'' = f(TD') = f \circ g(TP).$$

と記述される．関数 g で行われる処理が 2 段階圧縮の 1 段階目の圧縮に，関数 f で行われる処理が 2 段階目圧縮に該当する．一方，テストデータ展開手順は，この逆変換

$$TP = g^{-1} \circ f^{-1}(TD'').$$

で表現される．

f^{-1} の処理をテストデータ展開器のデータデコーダ部が，g^{-1} の処理をスキャンイン操作制御器が行う．

つまりデータデコーダ部 f^{-1} は TD'' からのデータを入力とし TD' を出力とする．またスキャンイン操作制御器は，TD' を入力とし TP を出力とする．

3.2 でスキャンベクトル長圧縮法の原理を説明する． 3.3 でスキャンベクトル長圧縮法のデータ削減効率を上げる FF 並び替え法について説明する．3.4 でスキャンインモード制御データの構成法を述べる．最後に 3.5 でスキャンイン操作制御器について説明する．

3.2 スキャンベクトル長圧縮法

スキャンベクトル長圧縮法は，1 回スキャンイン操作，2 回スキャンイン操作と呼ばれる 2 つのスキャンイン法を用いて行う．これらのスキャンイン操作を用いて故障検出率を維持しながら必要なテストデータ量を削減する．1 回スキャンイン操作は，入力するテストデータ量がスタンダードスキャン方式と比較して少ないという長所を有する．この 1 回スキャンイン操作を出来るだけ多くのテストパターンに適用しテストに必要なデータ量を削減する．しかしながら，千葉大スキャンパスは，閉じているラッチの値とその出力側に接続されているラッチの値は常に同じであるため 1 回スキャンインでは任意の 1 パターンベクトルを入力することができない．そのため，故障検出率がスタンダードスキャン方式と比較して少ないという欠点を有する．一方，2 回スキャンイン操作では任意のテストパターンを入力することができる．そこで 1 回スキャンイン操作では入力できないテストパターンを 2 回スキャンイン操作で入力する．よって本手法はスキャンイン操作をテストデータ量削減のため，故障検出率に影響の無い限り 1 回スキャンイン操作と呼ぶ方法でテストを行い，1 回スキャンイン操作を用いたテストでは検出不可能な故障を 2 回スキャンイン操作と呼ぶ方法を用いたテストで検出する．まず 3.2.1 において提案手法で用いるスキャンイン操作操作の説明を行う．次に 3.2.2 においてスキャンベクトル長圧縮法の説明を行う．

なお，この節では特に断らない限りスキャンチェーンは一本であると仮定する．

3.2.1 スキャンイン操作

ここでは，スキャンベクトル長圧縮法で用いる 1 回スキャンイン操作と，2 回スキャンイン操作について説明する．

1 回スキャンイン操作は，千葉大スキャンにおいてスキャンイン操作を 1 回だけ行う．スキャンインを行った後，CLK_{SE}, CLK_{SO} を操作することにより，各マスタラッチに格納されているデータをすべて対応するスレーブラッチへ転送する．スキャンイン操作終了後，閉じているラッチとそのラッチの出力に接続する隣接 FF のマスタラッチは同じ値となるので，スレーブに転送された値もこのような依存関係を有する．この依存関係は， 偶数番目のラッチが閉じた状態でスキャンイン操作を終えた場合と奇数番目のラッチが閉じた状態でスキャンイン操作を終えた場合で異なる．偶数番目のラッチが閉じた状態で終える 1 回スキャンイン操作を一回偶数ビットスキャンイン操作，一方奇数番目のラッチが閉じた状態で終えるスキャンイン操作を奇数ビットスキャンイン操作と呼ぶ事にする．またスキャンインされるデータ量は通常のスキャンイン操作の約半分である．例えば，長さ 4 のスキャンチェーンにスキャンインベクトル (0, 1) を 1 回スキャンイン操作を行う場合, (FF_0, FF_1, FF_2, FF_3) = (0, 0, 1, 1) となる．

一方，2 回スキャンイン操作とは，千葉大スキャンにおいてスキャンイン操作を 2 回連続して行う事により，任意のパターンをスレーブラッチに設定するスキャンイン操作である．最初のスキャンインで偶数番目のビットを設定し，次のスキャンインで奇数番目のビットを設定する．シフトインされるデータ量は通常のスキャンイン操作と同様である．例えば (FF_0, FF_1, FF_2, FF_3) = (0, 1, 0, 1) を設定する場合は，最初に偶数ビット (FF_0, FF_2) = (0, 0) を設定を行い， 次に奇数ビット (FF_1, FF_3) = (1, 1) を設定を行う．

また千葉大スキャン FF は 1 回のスキャンアウトで取り出せるテスト応答は偶数番目，若しくは奇数番目の FF のテスト応答のみである．よってすべての FF のテスト応答を取り出すためには，2 回スキャンアウトを実行する必要がある．提案手法ではスキャンインしたテストデータをスレーブラッチに保持した状態を維持しつつ，マスタラッチのキャプャ動作とスキャンアウトを 2 回連続して行う．

3.2.2 スキャンベクトル長圧縮の流れ

図 3 を用いて提案圧縮法を説明する．これはスキャン長が $bit_0 \sim bit_5$ の 6，テストパターン数が $tp_0 \sim tp_4$ の 5 のテスト集合へ提案手法を適用したものである．左側が圧縮前．右側が圧縮後のデータである．テストデータの前に 1 Step と書いてあるものは 1 回スキャンイン操作適用可能テストベクトルである事を意味する．この例では，tp_1 と tp_3 が該当する．tp_1 は，(bit_0, bit_1, bit_2, bit_3, bit_4, bit_5) = (X, 1, 1, 0, 0, 0) であり，$bit_1 = bit_2$, $bit_3 = bit_4$ の条件を満たしているため 1 回奇数ビットスキャンイン操作を適用できる．一方 tp_3 は，(bit_0, bit_1, bit_2, bit_3, bit_4, bit_5) = (1, X, X, X, 0, X) であり，1 回偶数ビットスキャンイン操作の適用条件 $bit_0 = bit_1$, $bit_2 = bit_3$, $bit_4 = bit_5$，と 1 回奇数ビットスキャンイン操作の適用条件 $bit_1 = bit_2$, $bit_3 = bit_4$ 双方を満たしている．よってどちらのスキャンイン操作を適用しても問題ない．tp_3 を 1 回偶数ビットスキャンイン操作を適用した場合，圧縮後のテストデータ量は，26 ビットとなる．

スキャンベクトル長圧縮前のテストデータ TP のデータ量

を $|TP|$，圧縮後のテストデータ TP' のデータ量 $|TP'|$ とする．また圧縮前のスキャン長を l とすると，スキャンベクトル長圧縮前のテストデータは，$|TP| = nl$ と記述できる．スキャンベクトル長圧縮によって各テストベクトルは，最大で半分まで圧縮できることから $|TP'| \geq nl/2$ となる．

よって $|TP'|/|TP| \geq 0.5$ となるため，スキャンベクトル長圧縮法の最大データ削減率は，50%となる．

3.3 FF の並べ替えによる圧縮率向上

図 3，4 を用いて説明する．図 4 は図 3 の FF の並べ替え後のテストデータである．

図 3 は，bit_0, bit_1, bit_2, bit_3, bit_4, bit_5 の順にスキャンチェーンを構成する場合に相当する．このパターンのうち tp_1, tp_3 は 1 回スキャンイン操作が適用可能であるが，tp_0, tp_2, tp_4 はこれが適用できない．しかしながら図 4 のように，bit_1 と bit_2 及び bit_4 と bit_5 を入れ替え，スキャンチェーンを bit_0, bit_2, bit_1, bit_3, bit_5, bit_4 の順に配線することにより，tp_0, tp_2 が新たに 1 回スキャンイン操作が適用可能となる．tp_0, tp_2 は，1 回偶数ビットスキャンイン操作が適用可能である．

この例では並べ変えを行う事により，テストデータ量を 26 ビットから 19 ビットまで圧縮できる．このように適切に FF の並べ替えを行う事により，スキャンベクトル長圧縮法のテストデータ圧縮率の向上ができる．

3.4 スキャンイン操作制御データの構成

提案手法では，テストデータに加えスキャンイン操作の制御のためのデータの構成も行う必要がある．ここでは，3.2 の方法で圧縮されたテストデータからスキャンイン操作制御データを構成する方法を説明する．

表 1 にスキャンイン操作制御データのデータ変換表を示す．各スキャンイン操作は固定長 2 ビットで表現される．この表の 1 列目が提案手法で用いる各スキャンイン操作を示している．2 列目が各スキャンイン操作に対応する制御コードである．テストパターンの中には,1 回偶数ビットスキャン操作，1 回奇数ビットスキャン操作のどちらでもスキャンイン可能なテストパターンが存在する．このようなベクトルは"xx"と符号化する．

スキャンイン操作制御データの構成例を表 2 に示す．表 2 は，Scan Chain 1~4 の 4 スキャンチェーン構造でのスキャンイン操作制御データの構成例である．各スキャンチェーンの，スキャン長は 4 であるとする．2 列目は，スキャンベクトル長圧縮前のテストデータを示す．3 列目は，提案圧縮を適用した際のスキャンイン操作制御データである．

Scan Chain 1 のデータは，“0011”である．これは，1 回偶数ビットスキャンイン操作を適用可能である．表 1 より制御コードは “00”であるのでスキャンイン操作制御データに “|00|”を追加する．次に Scan Chain 2 のテストデータが “0101”であるが，これは 2 回スキャンイン操作しか適用できない．よって表 1 より制御コードは，“01”となり，これを追加してスキャンイン操作制御データを “|00|01|”とする．Scan Chain 3 のテストデータの “00xx”は，1 回偶数ビットスキャンイン操作でも 1 回奇数ビットスキャンイン操作でもスキャンインが可能である．よって制御コードは “xx”となる．Scan Chain 4 も同様である．よって最終的にスキャンイン操作制御データを，“|00|01|xx|xx|”とする．

この例に示すように構成されたスキャンイン操作制御データにも X(xx) ビットが存在する．よってスキャンイン操作制御データもテストデータと同様に X ビットを有効に用いる事により，効率的に圧縮する事ができる．

3.5 スキャンイン操作制御器

最後にスキャンイン操作制御器を説明する．提案手法は，各スキャンチェーン，各テストパターン毎にスキャンイン操作制御データで指定されたスキャンインを行う必要がある．この制御をスキャンイン操作制御器が行う．

スキャン長 4 のスキャンチェーン 4 つから構成されるテスト

表 1　スキャンインモード制御データのデータ変換表.

Scan-in Op.		Control Code
One Step	Even	00
	Odd	11
Two Step		01
Don't Care		xx

表 2　スキャンインモード制御データの構成例.

	Test Data	Scan-in Op.
Scan Chain 1	0011	00
Scan Chain 2	0101	01
Scan Chain 3	00xx	xx
Scan Chain 4	1x1x	xx
Scan Cont. Data	|00|01|xx|xx|	

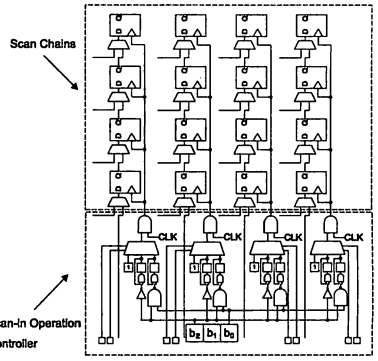

図 5　スキャンイン操作制御器のアーキテクチャ.

圧縮アーキテクチャのスキャンイン操作制御器の構成を図 5 に示す．図の上部がスキャンチェーン部，下部がクロック制御部である．各スキャンチェーンのクロック制御部は大きく分けて上から順にクロック線を制御する AND ゲート，3 入力 1 出力セレクタ，セレクタの各入力線に接続されているラッチ，組み合わせ回路部，8 進カウンタから構成されている．

クロック制御部の動作クロックは通常テストクロックの 1/2 周期となる．8 進カウンタは上位ビットから順番に，b_2, b_1, b_0 となる．3 つのラッチの初期値はすべて 1 である．またカウンタ初期値は，0 である．

3 入力セレクタを制御することによりスキャンイン操作を切り替える．3 入力セレクタの左側の入力が 2 回スキャンイン操作，中央の入力が 1 回偶数ビットスキャンイン操作，右側の入力が 1 回奇数ビットスキャンイン操作のクロック制御に用いられる．左側の入力は常に 1 であるため常にクロックが各 FF に入力される．

スキャン長が 4 であるため，1 回偶数ビットスキャンイン操作では 2 クロック，1 回奇数ビットスキャンイン操作では 2.5 クロックとなる．これを 8 進カウンタを用いて制御する．1 回偶数ビットスキャンイン操作，1 回奇数ビットスキャンイン操作完了時はそれぞれ 8 進カウンタが，4，5 の時である．8 進カウンタが 4 の時にセレクタの 3 入力のうち中央の入力に接続されているラッチに 0 がラッチされ，これ以降クロックがオフとなる．同様に 8 進カウンタが 5 の時にセレクタの 3 入力のうち右の入力に接続されているラッチに 0 がラッチされる．このようにして各スキャンイン操作のクロック制御を行う．

4. 評　　価

提案手法の有効性を示すため，4 つの評価実験を行う．4.1 においてスキャンベクトル長圧縮法による 1 段階目圧縮の効果を評価する．また 1 段階目圧縮により圧縮されたデータを展開するにはスキャンイン操作制御データが必要となる．そこで 4.2

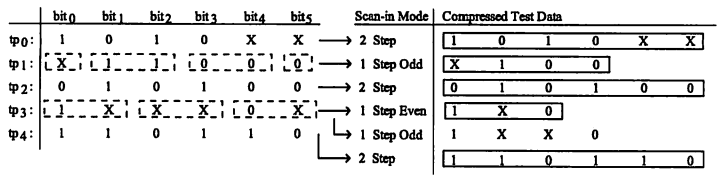

図 3 スキャンベクトル長圧縮.

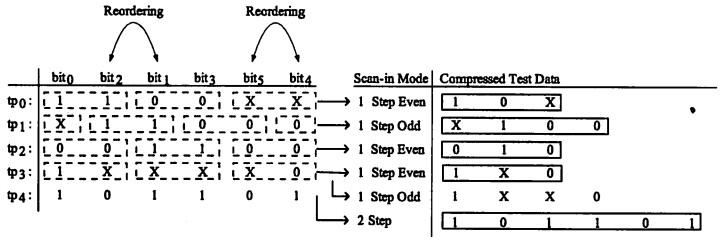

図 4 FF の並べ替えによるスキャンベクトル長圧縮率向上.

においてスキャンイン操作制御データの評価を行う．次に 2 段階圧縮の効果を評価するため，4.3 において ハフマンテストデータ圧縮を 2 段階目圧縮とした場合のテストデータ量の評価を行う．提案テストアーキテクチャの面積オーバヘッドは，スタンダードスキャン方式を用いた時と比較した場合，スキャンイン制御器とその入力部に付加されているラッチ回路である．4.4 においてこれらの面積オーバヘッドの評価を行う．評価はすべて ISCAS 89 ベンチマーク回路を用いて行う．

4.1 スキャンベクトル長圧縮法によるテストデータ量の圧縮

ここでは，スキャンベクトル長圧縮法の圧縮率の評価を行う．(ATE へ格納するデータ量について直接評価を行っていない事に注意されたい．) 評価は，スキャンチェーン数をパラメータとして行う．スキャンチェーン数が，1，8，16，32，64 の各場合で提案手法の 2 段階目の圧縮法にハフマンテストデータ圧縮を適用する．そして ATE へ格納されるデータ量が最小となるものを探索する．スキャンベクトル長圧縮法のデータ削減率向上のために行う並べ替え処理は，20,000 回のランダム検索で最適化する．提案手法を適用する際，スキャンイン操作制御データには “xx” ビットが存在する．X ビットをすべて “00(偶数ビットスキャンイン操作)” 埋めする場合と，“11(奇数ビットスキャンイン操作)” 埋めする場合の二通りで圧縮を行う．評価結果を表 3 に示す．表の 1 列目はベンチマーク回路名，2〜3 列目 TP, CBR は，順に圧縮前のテストデータ量 (ビット数), ケアビット率 (%) である．4〜7 列目は圧縮後の結果である．4 列目 F は，偶数，奇数ビットスキャンイン操作どちらも可能なテストパターンに関して，すべて偶数ビットスキャンイン操作で行った場合を E と示し，すべて奇数ビットスキャンイン操作で行った場合を O と示している．5 列目#S はスキャンチェーン数である．6 列目 TP' が圧縮後のテストデータ量，7 列目 CBR が圧縮後のケアビット率 (%)，そして 8 列目 CR が提案手法による圧縮率である．これは，圧縮前のテストデータ量を N，圧縮後のテストデータ量を M で表した場合，$(1 - M/N) \times 100(\%)$ で定義される．

スキャンベクトル長圧縮法は，理論的には最大で 50 %の圧縮率となる．評価ではどの回路も 40 %以上のデータ圧縮率となっており，これは多くの TP が 1 回スキャンイン操作を適用可能である事を示している．

また ATE へ格納するデータ量が最小となるスキャンチェーン数は回路によって異なる．これは圧縮前のテストデータのケアビット率，ビット列の配置，及びスキャンインモード制御データ量に依存する．(スキャンイン操作制御データ量の影響については後述する．) ケアビット率については，1〜12%ほど増加する．偶数，“xx” ビットを “00” 埋めする場合と “11” 埋めする場合で，スキャンインベクトル長圧縮法の圧縮率に，最小で 0.6%，最大で 4.6%差が発生する．

表 3 スキャンベクトル長圧縮法によるテストデータ圧縮率とケアビット率の変化.

Circuit	Org		PS				CR
	TP	CBR	F	#S	TP'	CBR	
s13207	165,200	6.9	E	8	85,805	12.9	48.1
			O	8	86,666	12.9	47.5
s15850	76,986	16.4	E	16	41,419	28.9	46.2
			O	16	42,689	28.1	44.5
s35932	28,208	64.7	E	64	15,419	70.2	48.5
			O	64	15,880	69.8	43.9
s38584	199,104	17.7	E	32	110,140	30.7	44.7
			O	32	112,536	30.1	43.5

4.2 スキャンイン操作制御データのケアビット率

次に提案手法で用いるスキャンイン操作制御データのケアビット率について評価を行う．評価は，スキャンチェーン数をパラメータとして行う．図 6 に評価結果をグラフに示す．グラフの横軸はスキャンチェーン数である．これは前節の評価と同様に 1，8，16，32，64 について評価を行っている．縦軸は，ケアビット率である．

スキャンチェーンの増加に伴いケアビット率が減少しており，どの回路も特性は総じて右下がりの傾向がある．これはスキャンの多重化を行う事により，ビット間の依存性が分散されるため，1 回偶数ビットスキャンイン操作でも 1 回奇数ビットスキャンイン操作でもスキャンイン可能なテストパターンが増加することを意味している．これによりスキャンイン操作制御データの圧縮率は，スキャンチェーン数が増加するほど向上する．

しかしながら，スキャンイン操作制御データ量はスキャンチェーン数に比例して増加するため，圧縮後による削減されるデータの絶対量はスキャンチェーン数の増加に比例しない．

4.3 2 段階圧縮によるテストデータ量の削減

4.1，4.2 ではそれぞれ，提案 2 段階圧縮の 1 段階目圧縮の評価，及びスキャンイン操作制御データ量の評価をそれぞれ行ったが，本節では提案 2 段階圧縮の評価を行う．ハフマン符号によるテストデータ圧縮法を 2 段階目圧縮として提案圧縮法を適用した場合，ATE に格納する最終的なデータ量を従来法を適用した場合と比較を行う．LSI テストデータ圧縮のためのハフマン圧縮法にはさまざまなものが提案されているが [9]，本評価では典型的な二元ハフマン符号構成法を用いて評価を行う．ハ

表 4　スタンダードスキャンとの ATE に格納するテストデータ量の比較.

Circuit	TP	Std. Scan	Chiba					BR(%)
		TD''	Len	#S	TP'	TC'	TD''	
s13207	165,200	53,118	4	8	85,805	3,776	34,773	34.61
			8	8	85,805	3,776	27,687	24.99
s15850	76,986	32,062	4	16	41,419	4,032	24,510	23.55
			8	16	41,419	4,032	22,533	16.05
s35932	28,208	22,928	4	64	15,491	2,048	14,420	37.11
			8	64	15,491	2,048	13,004	34.82
s38584	199,104	89,268	4	32	110,140	8,704	69,698	22.03
			8	32	110,140	8,704	69,698	14.52

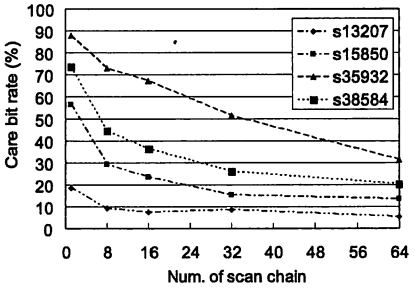

図 6　スキャンインモード制御データのケアビット率.

フマン符号を適用する際ブロック長は，4，8 とする．各ベンチマークにおいて，スキャンチェーン数，1，8，16，32，64 の各場合に提案手法を適用し，圧縮率が最大となるスキャンチェーン数を探索する．最終的に ATE に格納するテストデータ量を従来法の場合と比較を行う．表 4 に評価結果を示す．評価では，スキャンチェーン数を 1~64 で評価を行っているが，本表にはその圧縮率が最大のものを示している．1 列目はベンチマーク回路名，2 列目 TP は，圧縮前のテストデータ量である．3 列目 TD'' は従来圧縮法での圧縮後の ATE へ格納するデータ量である．4~7 列目は提案手法の評価結果を示す．4 列目 Len は，ハフマン圧縮時のブロック長を示す．5 列目#S は，最大圧縮率を得るスキャンチェーン数を示す．6 列目 TP' はスキャンベクトル長圧縮後のテストデータ量である．7 列目 TC' はそのスキャンイン制御データ量である．8 列目 TD'' は，2 段階圧縮後の ATE へ格納するデータ量である．最後に 9 列目 BR は，2 段階圧縮後のデータ量の従来圧縮適用後のデータ量に対するデータ削減率 (%) である．従来ハフマン圧縮による圧縮後のデータを M，提案 2 段階圧縮を用いたハフマン圧縮後のデータを N とすると，$(1 - M/N) \times 100(\%)$ で定義される．

テストデータ量の削減率 BR は，最大で，37.1%，平均で 26.0%である．削減率はブロック長に依存する．ブロック長が長いほうが提案手法の効果が少なくなる．回路によって削減率の低下の度合いは異なるが，2~9%低下する．また最大の削減率を得るスキャンチェーン数は回路に依存する．スキャンチェーン数を多くし，ビット間の依存関係を少なくすれば，提案データ削減法でのデータ削減率は向上するが逆に制御データ量が増加する．一方逆に制御データ量の削減のためスキャンチェーン数が少ない場合は，提案手法でのデータ削減率が悪化する．

4.4　スキャンイン操作制御器とその入力部の D ラッチの面積オーバヘッド

最後にスキャンイン操作制御器とその入力部の D ラッチの面積オーバヘッドの評価を行う．表 5 に評価結果を示す．表の各列は左から順に，回路名，スキャンチェーン数，クロックコントローラを構成する素子数，面積オーバヘッドである．素子数と面積オーバヘッドは Cell (D ラッチも含む)，FF それぞれ別々に算出している．クロックコントローラの構造からも推測

表 5　スキャンイン操作制御器と
その入力線ラッチの面積オーバヘッド.

Circuit	#S	Num.		Overhead(%)	
		Cell	FF	Cell	FF
s13207	8	230	7	4.5	1.1
s15850	16	422	6	7.1	1.2
s35932	64	1,521	5	6.6	0.3
s38584	32	783	7	4.0	0.5

がつくようにスキャンチェーン数とセル数に相関がある．

FF は，すべてカウンタを構成するために用いられるためスキャン長を N とすると $\log N$ のオーダとなる．またそのオーバヘッドの実装対象回路に対する割合はどの回路も 10 %以内となる．

5. むすび

本論文では，千葉大スキャン FF 方式の下での縮退故障テストデータの圧縮手法を提案した．提案手法は，スキャンベクトル長圧縮法を用いた 2 段階圧縮法を用いる事により，スキャンベクトル長圧縮法を用いない従来圧縮法の場合と比較して，テストデータ量の削減が可能となる．評価において，提案手法を用いた場合，37.1%，平均で 26.0%テストデータ量の圧縮が可能であることを確認した．

文　献

[1] L. Wang, C. Wu, X. Weng, "VLSI Test Principles And Architectures: Design for Testability," Morgan Kaufmann Pub, 2006.

[2] A. Chandra and K. Chakrabarty, "Test data compression for system-on-a-chip using Golomb codes," Proc. 18th VLSI Test Symposium, pp.113-120, 2000.

[3] I. Hamazaoglu and J.H. Patel, "Reducing test application time for full scan embedded cores," Proc. 29th International symposium on fault tolerant computing," pp.260-267, 1999.

[4] K.J. Balakrishnan and N.A. Touba, "Improving encoding efficiency for linear decompressors using scan inversion," Proc. 35th International Test Conference, pp.936-944, 2004.

[5] B. Arslan and A. Orailoglu, "Circular scan: a scan architecture for test cost reduction," Proc. 7th Design Automation and Test in Europe, pp.1290-1295, 2004.

[6] K. Namba, H. Ito, "Scan Design for Two-Pattern Test without Extra Latches," IEICE Trans. Inf. & Syst., Vol.E88-D, No.12 Dec, pp.2777-2785, 2005.

[7] A. Krstic and K.-T. Cheng, Delay Fault Testing for VLSI Circuits, Kluwer Academic Publishers, 1998.

[8] K. Namba, H. Ito, "Interleaving of delay fault test data for efficient test compression with statistical coding," Proc. 15th Asian Test Symposium, pp.389-394, 2006.

[9] A. Jas, J. Ghosh-Dastidar, M. Ng, and N.A. Touba, "An efficient test vector compression scheme using selective huffman coding," IEEE Trans. Comput.-Aided Des. of Integr. Circuits Syst., Vol.22, No.6. pp.797-806, 2003.