安全・安心のための半導体設計・テスト技術

福本　聡，新井　雅之，岩崎　一彦

半導体技術は，安全・安心な情報ネットワーク社会を実現するための基幹要素の役割を担っている．本稿では，現在の半導体の信頼性を保証している設計・テスト技術について概説し，さらに，今後の信頼性低下要因のひとつであるソフトエラーとその対策，半導体によるセキュアコンピューティングへのアプローチについて紹介する．

キーワード：設計検証技術，半導体テスト，ソフトエラー，セキュアプロセッサ

1. はじめに

半導体技術は，情報通信システムを構築するための最も重要な要素のひとつである．今日の，コスト・性能・信頼性のバランスに優れたコンピュータや携帯電話の普及は，さまざまな半導体技術の成果の賜物といっても過言ではない．とりわけ，ユーザに安定して製品を供給するための品質保証技術，すなわち厳しい設計・製造コストの制約下で一定の品質を保証するための技術が果たしてきた役割は大きい．その品質とは，個別の半導体に与えられた設計品質と製造品質である．設計品質の保証は，ユーザに提示された半導体の仕様を設計が正しく満たしているか審査することで実現される．また，製造品質の保証は，完成した半導体が設計通りに正しく機能するか評価することで達成される．

しかしながら，このような半導体技術に対して，我々はいま，より高度な役割を期待しようとしている．それは，安全・安心な情報ネットワーク社会を実現するための基幹要素としての役割である[1]．現在の大規模で複雑な情報通信システムは，設計・製造段階では検出し得ない故障や想定外の不具合，あるいは人的な悪意ある攻撃など，新たな脅威にさらされている．こうした脅威に対処するため，半導体技術に要求される信頼性の水準は，単に故障し難いという性質にとどまらず，未知の故障に耐え，それをユーザに意識させない性質を備えるところにまで高まっている．また，半導体がハードウェアとして，積極的にシステムのセキュリティを支援し，外部の攻撃から情報処理機能や通信機能あるいはデータを守る技術，セキュアコンピューティングの性質を備えることが期待されている．

本稿では，以上のような観点から，現在の半導体の信頼性を保証している設計・テスト技術について概説し，さらに今後顕在化が懸念される半導体の信頼性低下要因，半導体のセキュアコンピューティングへのアプローチについて述べる．具体的には，まず，半導体の論理を正しく設計し，それを検証するための設計検証技術，およびテスト容易化設計を中心としたテスト技術について解説する．つぎに，微細化と低電源電圧化に伴って発生の増加が予想されるソフトエラーとその対策について触れ，最後に，セキュリティをハードウェア側から支援するセキュアプロセッサの考え方について紹介する．

2. 設計検証技術

半導体の設計はさまざまな要素技術によって実現されるが，設計の信頼性を担保するという意味で重要となるのは，正しい動作が得られることを検査・保証する技術，すなわち設計検証技術である．

2.1 シミュレーションによる検証

現在のLSIには，数千万から数億個ものトランジスタが集積されるようになっている．このような大規模な集積回路を，人手で作図して設計することはもはや不可能である．そこで，ハードウェア記述言語（HDL：hardware description language）による回路記述と，論理合成に代表される合成技術によってハードウェアの自動生成が行われるようになった．

このようなEDA（electronic design automation）ツールによる設計自動化の大きな利点は，シミュレーションによって設計の正しさを比較的容易に検証できることである．デジタルシステムの各レベル（レジス

ふくもと　さとし，あらい　まさゆき，いわさき　かずひこ
首都大学東京 システムデザイン学部
〒191-0065 日野市旭が丘6-6

タ転送レベル，ゲートレベル，トランジスタレベルなど）において，極めて正確に信号値やタイミングに関する動作検証が可能である．

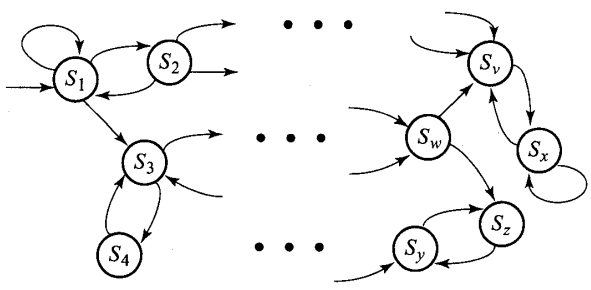
しかしこの検証では，特定の入力に対するシステムの挙動・状態を見るのが基本である．したがって，すべての入力とすべてのシステム状態の組合せについて動作を確認することは，多くの場合に不可能である．また，確認する動作の選択自体を人手によって行う必要があり，そこに誤りを見逃す余地が生ずる．

2.2 形式的設計検証技術

そこで，最近では設計検証の新しい手法として，形式的設計検証技術（フォーマルアプローチ）が注目されている[2]．これは，システムの仕様を数理論理学的手法で形式的に表現して，システムがそれを満たしているかどうか検証する手法である．ハードウェアだけではなくソフトウェアの設計検証にも応用されている．検証の手法は，モデル検査と定理証明型検査に大別される．

モデル検査は，システムの挙動をモデル化し，それが形式的に記述された仕様を満足するかどうかを判定する．判定結果はYesまたはNoである．例えば，図1(a)のシステムの挙動を，同(b)のような状態グラフによってモデル化できるとき，検査したい仕様の形式的記述に対応するグラフの状態探索から，システムがその仕様を満たしているかどうかを判定する．モデル検査は，ハードウェアの設計検証を自動化できるという点で非常に有用である．しかし，システムの挙動が複雑になりモデルが大規模になると，いわゆる状態爆発が起こるという問題がある．

(a) あるデジタルシステム



(b) 状態グラフ

図1 システムの挙動のモデル化

一方，定理証明型検査は，検査する性質を定理で記述し，それを帰納法などを用いて証明する．この手法には，大きなモデルについても状態爆発が起こらないという特長があるが，検証の自動化という観点からはやはり万能ではない．ただし，計算機上で機能する定理証明支援システムによって記述言語の定義や定理証明ツールが提供されており，ある程度の定理については自動証明が可能となっている．

3. テスト技術

設計に誤りがないと判断できるとき，つぎに問題となるのは製造工程での信頼性の保証である．テスト技術は，半導体の製造上の欠陥を検出する技術である．

3.1 半導体テストの概要

半導体テストは，LSIの機能ごとに，論理回路テスト，メモリテスト，アナログ・ミクストシグナルテストなどに分類される[3]．いずれも，ウェハ完成後，モールドパッケージ封入後などの，いくつかの製造工程の節目に実行される．

テストは，回路に対して何らかの入力や命令を与え，それに対する応答を観測することで実現される．論理回路テストであれば，入力テストパターンを回路に印加して出力値を見る．テストでは，物理的欠陥の挙動を論理的な故障モデルで単純化する．例えば，縮退故障（stuck-at fault）と呼ばれる故障モデルでは，信号線のグランドまたは電源への短絡は，論理値0または1にモデル化される．組合せ回路における縮退故障については，それが回路中にひとつだけ存在するという前提であれば，基本的に決定論的なアルゴリズムによって検出に必要な入出力パターンを特定できる．また，メモリテストでは，メモリの各アドレスに対応するセルへの書込み・読込みテストを実行する．テストするアドレス空間の設定や命令の順序には独特のノウハウがあり，“マーチ”や“チェッカーボード”と呼ばれるメモリテストのアルゴリズムに反映されている[3]．

半導体テストのテスト品質は，設定した故障モデルにおける故障の検出率で評価され，当然ながら高い検出率が望ましい．特に，縮退故障モデルでは，100%検出率が原則とされる．そのような高いテスト品質を実現するには高いテストコストが必要である．しかし，高性能プロセッサなどの一部の高価格品を除けば，多くのLSIの製造は厳しいテストコストの制約下にあ

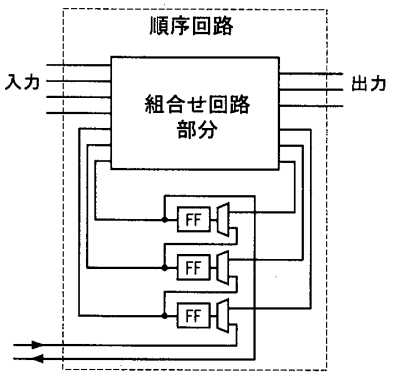
る．とりわけ，テスト時間は直接LSIの単価に影響する要素として常に問題視されている．そこで現在，テストコスト削減のために不可欠となっているのが，テスト容易化設計（DFT：design for testability）の考え方である．

テスト容易化設計は，LSIの設計とテストとを完全に独立した要素とは考えず，設計の段階でその回路のテストが容易に実現できるように支援しようとするものである．典型的なテスト容易化設計の例としては，スキャン設計（scan design）と組込み自己テスト（BIST：built-in self test）が挙げられる．以下，それらについて具体的に述べる．

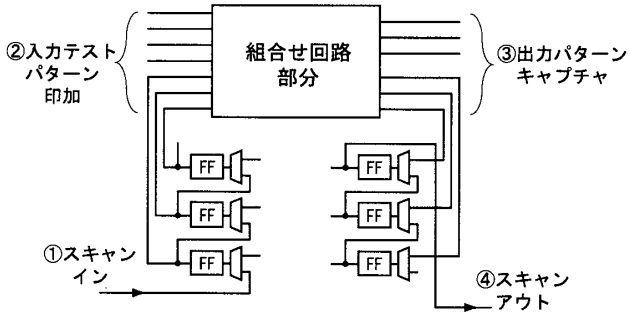
3.2 スキャン設計

スキャン設計は，順序回路におけるテストの困難さを克服するために導入されたテスト技術である．順序回路では，通常，回路の内部状態（フリップフロップの値）が過去の入力パターン履歴に依存し，効果的なテストパターンの生成が非常に困難である．これは内部のフリップフロップの値を外部から直接制御・観測できないためである[2]．

そこで図2(a)のように，回路内部のフリップフロップにシフトレジスタ機構を追加し，回路外部からの可制御性と可観測性を改善しようというのが，スキャン設計の基本的なアイディアである．スキャン設計された順序回路では，通常時はフリップフロップが本来の機能で使用され，テスト時にはシフトレジスタとして動作する．まず，外部からテストパターンをシフトインして，各フリップフロップに値をセットする．続いて回路本来の入力からの値とフリップフロップの値を，組合せ回路部分に印加する．その応答結果が回路本来の出力とフリップフロップまで伝播する．フリップフロップに到達した値についてはキャプチャした後，シフトアウトする．このような機構によって，結局，順序回路のテストを図2(b)のように，等価的に，組合せ回路のテストに帰着することができる．



(a) シフトレジスタ機構の追加



(b) 組合せ回路部分のテスト

図2 スキャン設計

3.3 組込み自己テスト

組込み自己テストは，図3に示すようにテスタの機能をテスト対象であるチップ内部に組込み，チップに自らをテストさせるというテスト技術である．組込んだテストパターン発生器で大量のテストパターンを生成し，高速に被テスト回路に印加することができる．これによって，高性能・高機能テスタを導入するためのコストを削減することができる．また，実動作テスト（at-speed testing）が比較的容易であるという利点もある．

なお，印加した大量の擬似ランダムパターンに対する応答結果もまた大量になる．それらの結果は，MISR（multiple input signature register）と呼ばれる圧縮機で圧縮されてから外部に出力され，あらかじめ計算された期待値と比較される（図3参照）．これによって，非常に効率的な故障検出が可能となる．

しかし一方で，組込み自己テストには，外部からテスタでテストパターンを印加する従来のテスト方式とは異なる問題点もある．組込まれるパターン発生器には，通常，線形帰還レジスタ（LFSR：linear feed-

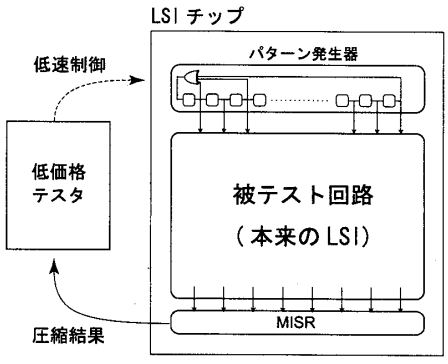


図3 組込み自己テスト

back shift register）が使用され，生成されるテストパターンは擬似ランダムパターンである．そのため，大量のテストパターンを印加するにもかかわらず，アルゴリズムで生成されたテストパターンを用いる場合よりも故障検出率が低い．この欠点を改善するため，検出率を高めるためのさまざまな手法が研究されている．例えば，検出率向上の妨げとなる検出困難な故障を検出するテストパターンをランダムパターンから作り出す各種の手法や，回路にテストポイントを挿入して可制御性と可観測性を改善する手法がそれである．

以上は，論理回路における組込み自己テストの説明であるが，メモリテストにおける組込み自己テストも広く実用化されている．それはメモリBISTと呼ばれる．テスト時には，組込まれたメモリテストのアルゴリズムが起動して自動的にメモリセルのテストが実行される．また，歩留まりの向上を意識して，あらかじめ冗長なメモリセルを用意し，故障検出したセルと自動的に交換する機構を備える手法もある．これは冗長救済技術（BISR：built-in self repair）と呼ばれている．

4. ソフトエラーとその対策

4.1 懸念される信頼性低下要因

上述のような設計・テスト技術の貢献もあり，今日まで，半導体産業は安価で高信頼なLSIを供給し続けてきた．しかし，次世代および次々世代の半導体設計・製造の現場では，LSIの信頼性を大きく揺るがすさまざまな要因が顕在化すると予想されている．その最大の原因は，半導体加工技術の微細化と設計マージンの減少にある．

18ヶ月で半導体の集積度が2倍になるという経験則，ムーアの法則が今後も成り立つとすれば，集積回路の最小加工寸法を表すプロセスルールは，現在の主流の90 nmから数年ごとに次世代65 nmおよび次々世代45 nmへ移行するものと思われる．これにともなってクロック周波数が上昇する一方で，動作電圧の低電圧化などが進み，全体的に安定な動作を確保するための設計余裕がほとんど失われる可能性がある．

プロセスの微細化は，各寸法パラメータの相対的ばらつき，ひいてはトランジスタ特性のばらつきの増大をまねく．その結果，LSIの動作速度のばらつきなどを制御することが困難になる．電圧の変動，温度の変動についても同様であり，これらはPVT（process, voltage, temperature）変動の問題と呼ばれている．また高集積化によって，単位面積当たりのトランジスタ数が増大し，配線幅および配線間隔が縮小する．宇宙線中性子やα線によって発生する過渡的な故障であるソフトエラーや，配線間の信号の飛びつきによるクロストーク遅延や誤動作といった信頼性低下要因が懸念されている[3]．

こうした問題の中から，以下ではソフトエラーに注目して，その影響と対策について具体的に述べる．

4.2 ソフトエラーの概要

ソフトエラーの存在そのものは比較的古くから知られており，1978年にはインテルによってそのメカニズムも明らかにされている．一時期は，放射線シールドによってその対策は完了したと考えられていたが，プロセスの微細化や動作電圧の低電圧化にともない，近年，再び深刻な問題となっている．2000年には，サン・マイクロシステムズから，ソフトエラーのために月に一度の頻度でサーバクラッシュが発生しているとの調査結果が報告された[4]．

ソフトエラーは，宇宙線を原因とする中性子が半導体内部のシリコン原子と衝突し，核反応を起こしたときに生成されるイオン化粒子によって発生する．あるいは，α線の進入でシリコン材料中に発生する電子・正孔対によっても引き起こされる．いずれも，浮遊したノイズ電荷が素子の臨界電荷量（値を保持するための最小電荷量）を超えたときにエラーとなる[5]．エラーは過渡的なものであり，素子の値をリセットした段階で正常機能が回復される．

実際のソフトエラーの影響は，メモリと論理回路とで異なる．メモリの場合は，通常1ビットのメモリセルの値が反転する誤りとして観測される．誤りがシステムレベルに波及するまでにマスクする方法としては，誤り訂正符号（ECC：error correcting code）が有効であり，実際にメモリチップで使用されている．しかし今後，微細化・低電圧化が進むと複数ビットのセルで誤りが発生することが予想され，さらに追加的な対策が必要になるものと考えられる．

一方，論理回路で発生するソフトエラーは，フリップフロップ（FF）の値の反転として観測される．ただし，この反転は発生のメカニズムによって2通りに分かれる．ひとつは，フリップフロップ内に発生したノイズ電荷による直接の反転であり，もうひとつは，組合せ回路内で発生したノイズ電荷によって瞬間的にゲートの値が反転し，それがフリップフロップまで伝播してキャプチャされる反転である（図4参照）．前

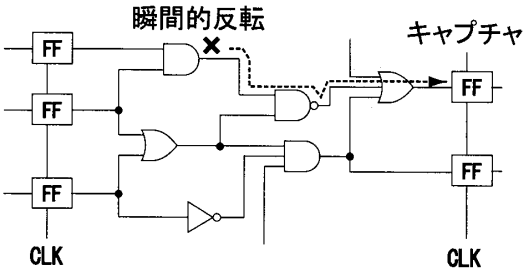


図4　SETのメカニズム

者の場合のエラーをSEU（single event upset），後者のそれをSET（single event transient）と呼ぶ．SETでは信号値をキャプチャするタイミングによって，値の反転が起こるか否かが決まるため，クロック周波数とノイズ信号の持続性がエラー発生率に影響する．現在のところ，論理回路のソフトエラーではSETよりもSEUの発生率が高いが，今後，微細化が進みクロック周波数が上昇するとともに，SETの発生率が増加してSEUのそれを上回るものと予想されている．

4.3　ソフトエラーの評価技術

実際のソフトエラーのメカニズムや発生率を評価するには，中性子を生成してLSIへ照射する実験を行う必要がある．現在，米国・ロスアラモス国立研究所にある中性子散乱センターで世界標準テストが行われている．日本国内でも，大阪大学核物理研究センターなどで照射実験用中性子源が開発されている．

シミュレーションレベルでは，ソフトエラーがメモリや論理回路に与える影響を評価するためのツールが半導体メーカやEDAベンダによって開発されている[6]．また，最近の半導体関係の国際会議でも，オンラインまたはオフラインでソフトエラーを検出・訂正する方法や，エラーに脆弱な回路部分の特定手法[7]に関する研究などについて活発な議論が行われている．

4.4　ソフトエラー対策

ソフトエラーへの対策は，プロセスレベルからシステムレベルまでさまざまに研究されている．メモリにおけるソフトエラー対策としては，前述のとおり，ECCによる誤り訂正が従来から主要なアプローチになっている．これは，システムレベルでの対策であるが，プロセスレベルでも，ソフトエラーの影響を低減する材料の検討や，プロセス変更による耐ソフトエラー構造を採用するなどの対策が考えられている．

論理回路においてもさまざまなソフトエラー対策が

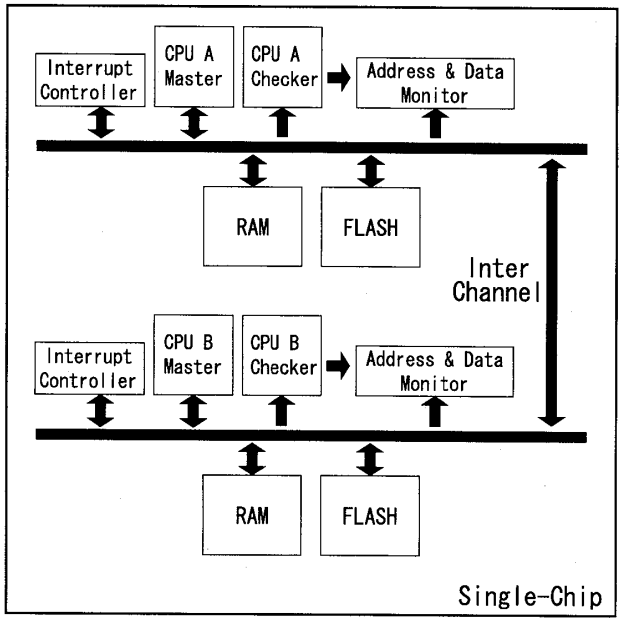


図5　耐過渡故障プロセッサアーキテクチャの例[9]

行われている．SEUの対策としては，各半導体メーカや研究機関から耐ソフトエラーラッチ・フリップフロップ回路が発表されている[5]．例えば，インテルは独自にスキャン設計されたフリップフロップを応用してハードウェアオーバヘッドが非常に小さい耐ソフトエラーラッチを提案している[8]．

また，SETについては，それに特化した対策という意味ではまだあまり議論されていないが，広い意味での過渡故障対策としてシステムレベルでのエラーマスク手法が多数研究されている．それらは，伝統的なフォールトトレラント技術（fault tolerant computing techniques）を応用することで実現できる．すなわち，チェックポインティングリスタート（checkpointing restart）などの時間冗長や，TMR（triple modular redundancy）に代表される空間冗長，あるいは符号（coding）などの情報冗長がベースになっている．

図5に過渡故障をマスクするために複数のコアを搭載したプロセッサアーキテクチャの例を示す．これは，マスタCPUとチェッカCPUからなる安全停止構成を2重化したものである．2重化ロックステップアーキテクチャ（dual lock-step architecture）と呼ばれる[9]．

5.　セキュアプロセッサ

最後に，半導体の新たな役割であるセキュアコンピューティング（secure computing）への支援につい

て簡単に触れる．

近年，暗号技術を応用した新しいプロセッサアーキテクチャで情報通信システムのセキュリティの問題を解決しようという試みがある．従来，セキュリティ管理はオペレーティングシステムの担当すべきものと考えられていたが，昨今の情報通信システムでは，そのオペレーティングシステム自体が悪意ある攻撃に対して安全ではないという認識が一般的になりつつある．オペレーティングシステムもソフトウェアである以上，不正な侵入や改ざんといった攻撃を受ける可能性がある．一方，命令セットアーキテクチャは，ひとたび半導体としてハードウェア実装されれば論理的に改ざんすることは不可能である．すなわち，正しく設計されていれば，プロセッサ自体は外部からの侵入・改ざんに対して安全であると考えられる．セキュリティ管理の役割をプロセッサに託そうという発想が生まれたのは当然の帰結といえる．

セキュリティ管理をつかさどるプロセッサ，セキュアプロセッサ（secure processor）が監視し，防止すべき攻撃には，不正プログラムの実行およびデータの書換えがある[10]．不正プログラムは，システムのセキュリティホールなどから侵入する．そして多くの場合，正常なプログラムと不正プログラムは識別が困難であると予想される．データの書換えについては，主記憶を標的とした攻撃が想定される．特定メモリ空間のデータ書換えや，プロセッサと主記憶間のデータバス値改ざんなどがそれである．

セキュアプロセッサは，プログラムやデータをチェックする機構をハードウェア実装する．これによって，プロセッサが認証したプログラムだけが実行される．またデータの改ざんは，そのデータがプロセッサに転送された時点で検出される．

これまでに，上記のような考え方に基づくセキュアプロセッサが多数提案されている．例えば，Drinicらは，メッセージ認証コード（MAC: message authentication code）を使ったプラットホームを提案した[11]．プログラムを認証すべきメッセージとして考え，それに対する認証コードによって不正プログラムを検出する．具体的には，ソースプログラムをコンパイルしたときに暗号アルゴリズムで計算したMACをオブジェクトプログラムに付加し，そのプログラムを実行するときに再びプロセッサがMACを計算して値が一致するかどうかを確認する（図6参照）．

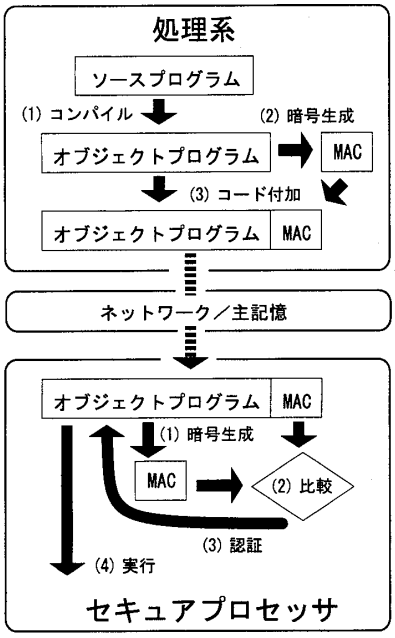


図6　セキュアプロセッサにおける実行プログラムの認証

6. まとめ

本稿では，現在の半導体の信頼性を保証している設計・テスト技術について概説し，さらに，今後の信頼性低下要因のひとつであるソフトエラーとその対策，半導体によるセキュアコンピューティングへのアプローチについて述べた．

半導体の信頼性保証技術に関連して，今後予想される動向のひとつに，冗長技術による実質的な歩留まりやスペックの向上が挙げられる．すなわち，冗長技術による障害回復を前提とした半導体テストの故障検出率100%制約の緩和や，同じく障害回復を前提とした低消費電力化・高速化に関する設計マージンの設定などである．また，今後は特定用途向けカスタムIC（ASIC: application specific IC）の再構成可能デバイス（FPGA: field programmable gate array）による置き換えが急速に進むものと予想されている．FPGA向けの信頼性保証技術の開発・研究が期待される．

参考文献

[1] 南谷崇：“ディペンダブルコンピューティングの過去・現在・未来,” 電子情報通信学会CPSY研究会，同DC研究会，CPSY 2006-7, DC 2006-7, pp. 37-42（2006, 4）．

[2] 米田友洋，梶原誠司，土屋達弘：ディペンダブルシステム（共立出版，2005）．

[3] http://strj-jeita.elisasp.net/strj/ITRS05/pdf/04_2005Test.pdf
[4] David Hamilton: The Wall Street Journal Online, November 7, 2000, 4:00 PM PT.
[5] http://strj-jeita.elisasp.net/pdf_ws_2003nendo/4C_ishibashi.pdf
[6] http://pr.fujitsu.com/jp/news/2005/09/15.html
[7] C. Zhao, X. Bai and S. Dey: "Noise Impact Analysis: Evaluating Transient Error Effects in VLSI Circuits," *ITC2005*, S 40.2 (2005).
[8] S. Mitra, M. Zhang 1, TM Mak, N. Seifert, V. Zia and K. S. Kim: "Logic Soft Errors a Major Barrier to Robust Platform Design," *ITC2005*, P 28.3 (2005).
[9] M. Baleani, A. Ferrari, L. Mangeruca, M. Peri and S. Pezzini: "FaultTolerant Platforms for Automotive Safety-Critical Applications," *CASES'03*, pp. 170-177 (2003).
[10] 井上弘士: "信頼性・安全性とプロセッサ," 情報処理学会誌, Vol. 46, No. 11, pp. 1218-1223 (2005).
[11] M. Drinic and D. Kirovski: "A Hardware-Software Platform for Intrusion Prevention," *MICRO37-'04*, pp. 233-242 (2004).