FPGA による産業用コントローラのための アーキテクチャと安全性に関する研究

Study on Architecture and Safety Integrity
for FPGA Based Industrial Controller

2018年 3月

長崎大学大学院工学研究科

森本 賢一

目次

はじめに

発電所や石油化学プラント、鉄道などの産業インフラ設備や生産設備においては、古くから CPU が用いられてきた[1],[2]。1990 年代後半より近年まで、CPU は半導体技術の発展に基づくクロック周波数の上昇により処理能力が向上したが[3]、CPU の基本構造が変わらないため既存プログラムを流用することができた。その結果、産業用コントローラは既存の資産を活かしながら新しく開発された高性能 CPU を用いることで、より複雑な制御機能に対応できた。新型 CPU の登場サイクルが数年という短期間でも、産業用途において重要な 20 年以上の長期保守や継承に問題は生じてこなかった。

近年、CPU のクロック周波数は上限に達し[4],[5]、性能向上の手段としてマルチコア化が主流である。マルチコアの CPU を使うためには分散処理型プログラムに作り直すことや、マルチコアを1つのコアに仮想化するミドルウェアの搭載などのシステムの再開発が必要である[6]-[12]。あるマルチコア CPU で再開発を実施しても、数年で新たな構造のマルチコア CPU に対して再開発を繰り返さなければならない。これは事業者の大きな負担やリスクとなり、制御システムが進化する上での障害となってきている。

この問題に対し FPGA（Field Programmable Gate Array）が注目されている[13]。FPGA は内部記述言語（HDL: Hardware Description Language）が機種に依存しないため、後継の FPGA 製品に容易に継承できる。しかし、次の2つの問題のため産業用コントローラの CPU の代替としての活用は進んでいない。

産業用コントローラでは、設備現場での試運転によって制御ロジックやパラメータの最適化を行う。FPGA は小さな変更でも内部回路全体の信号タイミングの変化が伴うため、事業者としては試運転段階での内部回路全体に及ぶ変更は許容し難い。制御回路やパラメータの部分的な変更が FPGA の回路全体に影響を及ぼすことが第一の課題である。

第二の課題は機能安全への対応である。LSI のパターンピッチは微細化し、地上においてもソフトエラーの影響を考慮する必要がある[14]。FPGA で高い SIL（Safety Integrity Level，安全完全性）[15]を達成する技術が明確でないことである。以上2つの課題を解決する FPGA を用いた産業用コントローラの実現は、きわめて早急に対応すべき重要なテーマである。

そこで本論文では、前述の課題を解決するため、FPGA を用いた産業用コントローラのための新しい動作原理 TDB(Two Dimensional Bus)を提案し、処理性能をシミュレーションにより明らかにし、実際の発電プラントへの応用を想定した能力評価を行った。また高い SIL の達成には不可欠な多重化した FPGA 回路の多様性を生み出すために新しい手法 DTM(Diverse Technology Mapping)を提案し、効果をシミュレーションにより明確にした。最後に TDB および DTM を備えた産業用コントローラの新しいアーキテクチャ SGC(Safety Green Controller)を提案し、国際規格に基づく評価を行い、実現性と信頼性を明らかにした。本論文にて提案する SGC の基本原理である TDB は前述の課題を解決するだけではなく、近年の制御システム設計には不可欠なモデル言語での複雑な制御回路設計に適している。さらに DTM は多様性のある FPGA 回路の作成に掛かる作業コストや製造コストを大幅に削減する効果がある。

第1章は緒論である。産業用コントローラの種類や、CPU とともに発展してきた歴史や背景を整理する。現代の産業用コントローラに必要な機能や課題について述べ、CPU による産業用コントローラの問題点や FPGA が有用な点について述べる。

第2章は TDB アーキテクチャについて述べる。TDB の基礎となる高位合成による FPGA 回路の生成について示し、モデル言語で記述された制御回路を並列演算するための条件を明らかにする。また TDB アーキテクチャを FPGA シミュレーションによって処理性能を評価し実用性について論じる。

第3章は DTM について述べる。試験および評価では、DTM によって生成した演算回路にエラーを発生させ、その効果を評価した。FPGA の動作周波数をオーバークロックすることで環境からのストレスによる回路エラーを模擬し、多様性の効果を評価する手法を採用した。この結果、DTM 手法によって共通要因故障が低減し、MTBF が改善することを明らかとなった。

第4章は TDB および DTM を組み合わせ、自己診断機能を持つ多重化産業用コントローラのアーキテクチャ SGC の提案とその評価について述べる。ここではシステムの潜在故障の低減がシステムの不動作確率(PFH)の低減に不可欠であることを示し、本論文で提案する多重自己診断の手法が高い安全完全性を達成することを論じる。

第5章はまとめである。本研究を総括し、今後の展望について述べる。

第 1 章 緒論

1.1 まえがき

様々なタイプの産業用コントローラが古くから CPU を用いて実現されてきている。本章では産業用コントローラにはどのような種類があり、どのような機能が求められているのか、CPU の能力の向上の歴史を踏まえて整理する。またそのうえで産業用コントローラと CPU が直面している問題を明確にし、FPGA が有利な点について述べ、本論文の目的を示す。

1.2 産業用コントローラの種類

産業用コントローラとは、プラント、工場設備、鉄道および自動車など、信頼性を重視する機械製品で搭載される制御システムである。産業用コントローラはハードウェアの特徴や制御を記述する言語によって下記 3 つに分類できる[16]-[27]。Ⅰ型Ⅱ型Ⅲ型とは、本論文での便宜的な呼称である。大規模な設備では制御対象の機械や設備の特性に応じ各型を組み合わせて使用する[1],[2],[28]-[33]。

Ⅰ型　機械制御コントローラ(自動車・ロボット・医療機器・エンジン)

特徴:専用ハードウェア、モデル言語、プログラム言語

Ⅱ型　シーケンス制御コントローラ(昇降設備、信号、輸送ライン)

特徴:汎用ハードウェア(PLC)、ラダー言語

Ⅲ型　プロセス制御コントローラ(石油化学・薬品・熱サイクルプラント)

特徴:汎用ハードウェア(DCS)、FBD 言語

PLC (Programmable Logic Controller)とは、生産設備や機械制御で使用される汎用の産業用コントローラ製品の総称である。DCS (Distributed Control System) とは、石油化学などプロセス制御で使用される産業用コントローラ製品の総称である。たとえば火力発電所では、ガスタービンやコンプレッサなどの回転機械はⅠ型の産業用コントローラにより回転数制御を行い[34]、熱エネルギーを蒸気に変換する熱サイクルプロセスはⅢ型を用いて温度や圧力流量の制御を行う[35]-[38]。 図 1-1 は発電プラントの制御システムの構成例である。ラダー言語および FBD(Function Block Diagram)とは、産業用制御装置の制御回路を記述するための専用言語である。

風力発電では、翼角度・方位はⅠ型の産業用コントローラによって絶えず変化する風速や風向に対したピッチ制御を行い[39),(40]、発電機と電力系統との接続など電気機器の制御にはⅡ型を用いる[41]。図 1-2 は風力発電設備の制御システムの構成例である。

再生可能エネルギーに不可欠なリチウムバッテリーでは、電力平滑化制御はⅠ型のコントローラで実施されるが、電力量や発熱を監視し冷却や蓄電を抑制する制御回路はⅢ型のコントローラが用いられる[42]。データセンターの冷却設備ではⅡ型およびⅢ型の産業用コントローラが用いられている[43),(44]。図 1-3 はデータセンターなど、建築物の空調設備の制御システム構成例である。

近年、より効率の良い電力管理や熱制御のために、データセンターの負荷をモデル化した最適制御が適用されることがある。このような場合では専用のハードウェアを用いたⅠ型の産業用コントローラを使用する[45)(46]。 パソコンが高機能化したため、SCADA (Supervisory Control And Data Acquisition)によって低価格な機材で制御システムを構築するパソコン計装という製品群があるが、機能としてはⅡ型またはⅢ型に類するものである[47]。SCADA とは、 PLC からのデータを表示し操作を行うパソコンソフトウェアである。これらの産業用コントローラは、用途の違いやハードウェアの違いはあれども、いずれも CPU を用いた制御演算ユニットを持つ[1),(2),(47]。

ここでCPUとは記憶メモリから命令や値を読み取り逐次実行しその結果を再び記憶メモリに保存することを繰り返すことで、既定の処理を実行する処理ユニットの一般をさす。Ⅱ型の製品はラダー言語を演算するための専用プロセッサを採用するケースもあるが、ラダープログラムを専用コードに変換し逐次実行する構造のため、CPU に類するものと考えてよい[32]。産業用コントローラは逐次実行型の CPU に大きく依存している。カルマンフィルタなど現代制御理論に基づく機械制御の適用分野の拡大は、CPU の処理能力向上に負うところが大きい。

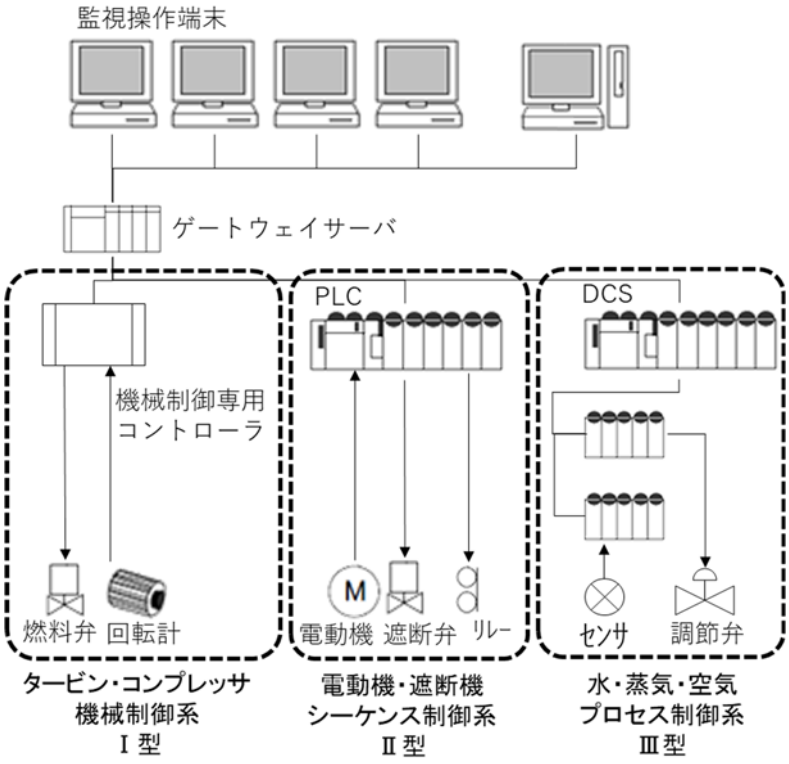


図 1-1. 発電プラントの制御システム構成例

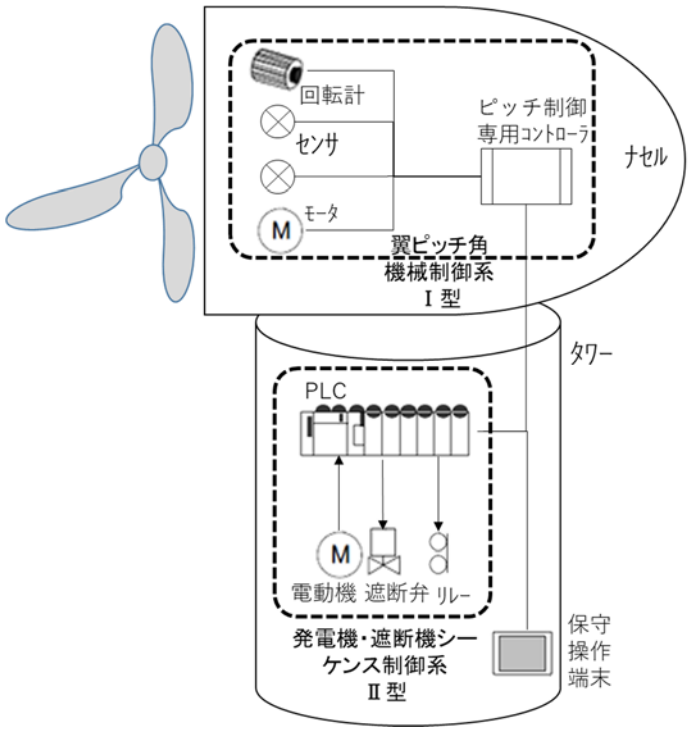


図 1-2. 風力発電設備の制御システム構成例

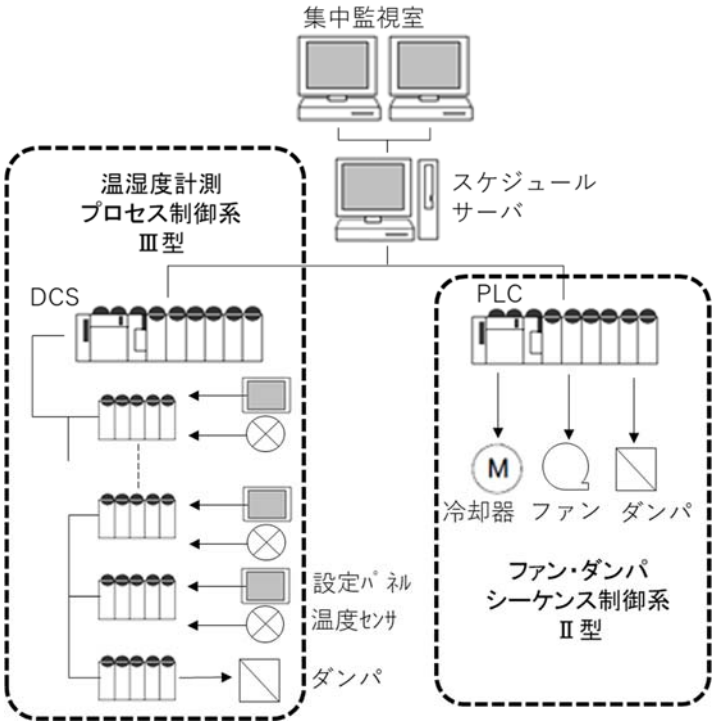


図 1-3. ビル空調設備の制御システム構成例

1.3 CPU の歴史と産業用コントローラの変遷

産業用途で使用に適した量産性と低価格を備えた CPU 製品の歴史と制御システムの発展への影響を示す。最初の CPU 製品として、1960 年代のインテル社の 4 ビット型 CPU の 4004 が代表的である。CPU が一度に処理できるビット長と命令実行のサイクルのクロック周波数（以下クロック）で CPU の性能を示したものが表 1-1 である。

表 1-1. 代表的な量産 CPU の処理データ長と動作周波数

発売年	型式	ﾃﾞｰﾀﾋﾞｯﾄ長	ｸﾛｯｸ	備考
1971	4004	4bit	0.7MHz	
1972	8008	8bit	0.8MHz	
1974	8080	8bit	2MHz	
1982	i286	16bit	12MHz	
1985	i386	32bit	40MHz	
1989	i486	32bit	100MHz	浮動小数点演算機能内蔵
1995	Pentium Pro	32bit	200MHz	
2000	Pentium 4	32bit	3.8GHz	
2004	Xeon	64bit	4.4GHz	
2006	Core 2 Series	64bit	3.3GHz	マルチコア

Intel® CPU Data Sheet

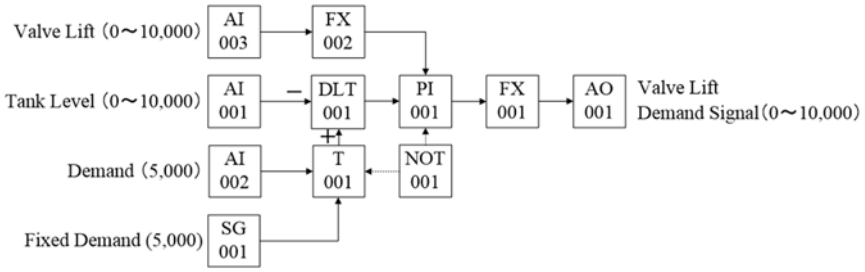
4 ビット～8ビットの CPU は主にディスクリート制御に用いられていた。ディスクリート制御とは、AND および OR などの論理演算で表現される制御手法である。ポンプやファンなど電動機械を起動停止する制御が代表的である。CPU が登場する以前は電磁リレーを用いて実現していた。1970 年代に 8 ビット CPU が商用化されたことが国内外で PLC が製品化されたことにつながる[1],[2],[47]。1980 年代に 16bit の CPU が登場したことにより、プロセス制御で必要な信号精度のアナログ信号処理ができるようになり、積分や微分を離散演算で行う産業用コントローラが製品化され始めた[1]。この段階ではまだ、1つまたは2つの計測信号で1つのアクチュエータを制御する単一のフィードバック制御（1ループ制御と呼称する）に適用されていた。1990 年代になり、制御システム同士が通信を介して結びつき、設備全体を分散制御（または協調制御）することが可能となった[48],[49],[50]。このような分散型制御システムを DCS と呼ぶ[2],[33]。

1.4 産業用コントローラの要件

産業用コントローラは対象機械や電力設備の動特性モデルに基づいた最適制御や、分散した産業用コントローラ同士の協調制御など高機能化を進めている[51]-[53]。現代の産業用コントローラは大容量の計算処理能力を必要としている。様々な機械製品や設備に使用されるため、求められる処理能力に大きなスケーラビリティが求められることは近年の特徴である。たとえば大規模な熱サイクルプラントでは、1つの制御システムが処理をする計測信がアナログで数千点に上る場合もある一方、内蔵機械ごとにコントローラを分散する自動車では、1つの産業用コントローラあたり 10 点未満の計測信号だけを扱うケースもある[2],[33]。様々なものが CPU で制御され通信で接続される生産現場では、すべてのセンサーや機械が互いに連携することを目指している[54],[55]。このため、産業用コントローラ用の CPU は大小さまざまなラインナップが必要になっている。例えばルネサス社の RX シリーズは処理能力メモリ容量などで数十の選択肢がある[56]。

また産業用制御システムは FBD 言語およびラダー言語など C 言語のような汎用のプログラム言語ではなく、PLC や DCS 製品ごとに搭載されている専用の制御記述言語をもちいて制御回路を設計することが一般的である。また近年ではモデリング言語である MATLAB や Simulink で記述するケースも増加している。これらも CPU の高性能化によって一般化してきた。図 1-4 は、(a)FBD 言語および(b)ラダー言語の記述例である。

(a)　FBD言語記述例



(b) ラダー言語記述例

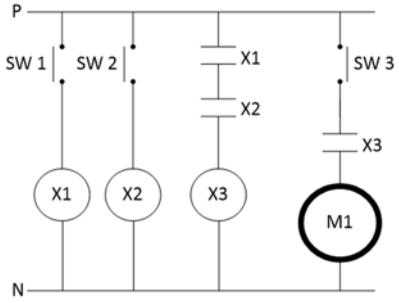


図 1-4.記述例

次に制御システムに必要な機能について整理する。1990 年代は産業用コントローラの制御回路は、Fortran 言語や C 言語など、アルゴリズムを直接プログラムで記述することが主流であった。制御論理の設計者はソフトウエアプログラマを兼ねていた。産業用コントローラの適用範囲の拡大やモデルベース設計の必要性から、FBD および MATLAB などのブロック図による制御記述一般化してきた。現代では制御論理を構築する技術者が直接プログラム言語を使用することは、事業の現場では極めてまれなケースである[32),(57)-(59)]。産業用コントローラを使用する現場で制御回路を変更する場合にはC言語のコンパイラを使用せず、制御記述言語による修正をオンラインで産業用コントローラにダウンロードする[(60)]。

1.5 国際規格機能安全と準拠の必要性

設備や機械が身体や環境へダメージを与えるリスクを低減するための準拠事項として、ISO 12100[(26)]が国際貿易の基本規格として採用されている。日本でも日本工業規格 JIS B 9700(機械類の安全性ー設計のための一般原則ーリスクアセスメント及びリスク低減)として制定されており、機械製品が準拠すべき基本規格(タイプ A 規格)として取り扱われてい

る。欧州ではCEマーキング機械指令(CE Marking Machinery Directive 2006/42/EC)において必須の規格として制定されており、欧州市場で流通するすべての機械製品が準拠する必要がある。

本規格は工業製品が及ぼす影響についてリスク分析し、その軽減対策を設計に組み込むことをもとめている。万一事故が発生した際には、その開発設計プロセスにおけるエビデンスを製造物責任法(Product Liability Law)に基づき裁判所に提示することが求められる。本規格では、物理現象や自然法則に基づく対策を第一に求める。本質安全(Inherent Safety またはIntrinsic Safety)および本質安全的安全(Inherently Safety)と呼ばれる安全対策である。前者は主に対象のエネルギーレベルの低減や、そもそもの危険減の除去などの対策を指す。列車衝突事故の回避のために踏切をなくして立体交差にするような対策が本質安全対策である。後者は危険な状態や災害の発生リスクを低減する方策である。たとえば湿式スプリンクラーのように熱源によって金属が溶融し止水弁が開くなどが本質的安全対策である。また発電用蒸気タービンでは、タービン回転数が危険な領域に至らないように、機械式オーバスピードトリップ装置を回転軸に取り付けることが国際的に取り決められている。これは、バネの力に抗した鉄製のピンが遠心力によって飛び出すことで、ある回転数以上になるとエネルギー源である蒸気弁を閉止する機械装置である。これは本質的安全対策である。本質的安全は、原理が確認された後は装置や材料の品質が適切であれば、製品単体個々の安全性は同水準で担保できると考える。このため、製作時の品質管理(Quality Control)が重視される。

一方で本質安全および本質的安全の対策は一般的に製品の製造コストや保守コストが大きくなる問題がある。また自動車の自動運転のように本質的安全対策が難しい製品も増加している。またコンピュータおよびセンサー、A/D コンバータなどの電子デバイスの高性能化、低価格化や小型化も理由となり、電子デバイス特にコンピュータによる安全保護や制御の仕組みが一般化してきた[21],[22],[26]。たとえばインバータモータなどの電気動力製品において、ヒューズによる電流制限の方法では、製品の大きさや定格によって数段階のヒューズを在庫として保有し組み合わせることが必要である。これが電子デバイスによる計測とソフトウェアによる保護の仕組みであれば、モータの定格や出力に大きな影響を受けず、共通の保護装置のソフトウェアパラメータの変更で適用が可能になる。物理的なヒュー

ズは設置せず、モデル計算によって推定されるデータに基づき産業用コントローラが判断し、運転動作に制限を加える手法が一般的になってきた[29),(61)]。これは共通部材による在庫圧縮の効果がある。前述の蒸気タービンでは、機械式オーバスピードトリップ装置は、動作確認試験を行った後は、一旦タービンを完全冷却し該当部分の適切な復帰を確認するなどの保守作業が必要である。一方、電子ピックアップによる電子式オーバスピードトリップ装置であれば、動作の確認後はすぐにシステムをリセットし、短時間の保守作業のあと直ちに運用に移ることができる。図 1-5 は機械式及び電子式のオーバスピードトリップ装置の構造を示している。鉄道の信号システムも、従来のような電気回路によるフェールセーフ構造だけではなく、通信など電子デバイスを介在させたシステムが不可欠になりつつある[62),(63)]。

このようなメリットの一方、コンピュータを用いた制御や保護の仕組みは本質安全および本質的安全の仕組みに比して、2つの観点で脆弱性を持つ。1つはシステマティック故障（系統故障）が発生する余地が大きいことであり、もう1つは電子デバイスのランダムハードウェア故障の発生の影響を確定的に論じることが難しいことである。

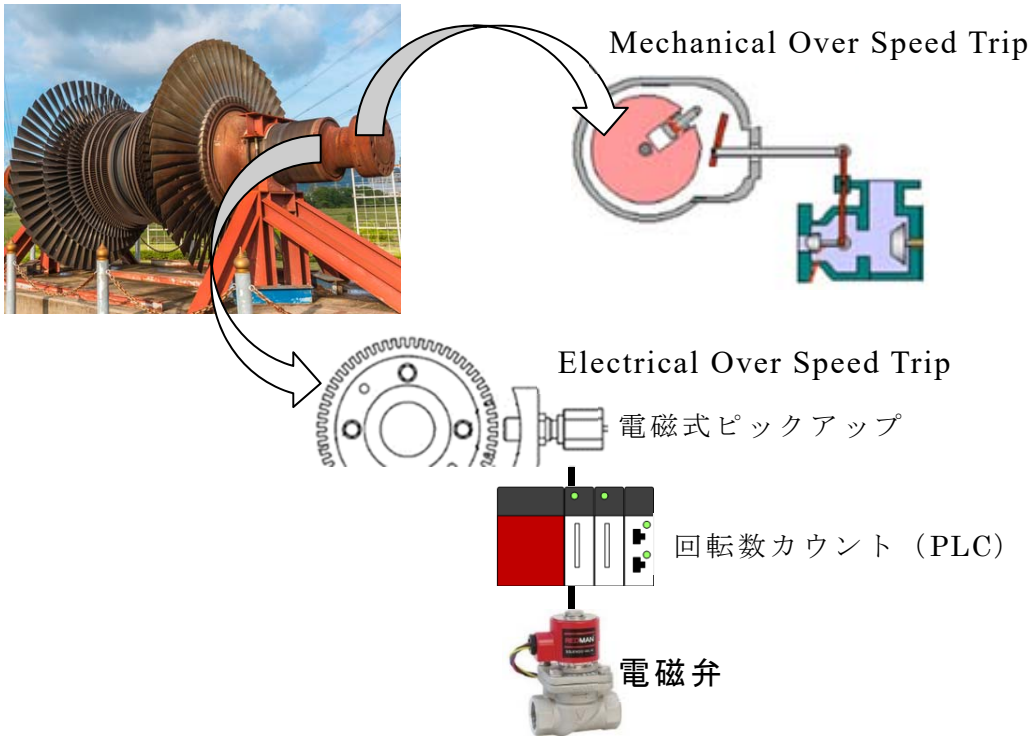


図 1-5. 蒸気タービンのオーバスピードトリップ装置

システマティック故障は人による作業の間違いが影響する。使用する電子デバイスの選定間違いなど設計作業上の間違いや、プログラムなどソフトウェアのバグなど記述間違いによって発生する故障が該当する。国内自動車メーカが米国で起こされた訴訟では 25 万行の C プログラムが対象となった[85]。エンジン制御やロボット制御では安全性にかかわる制御プログラムは大規模化しており、一旦制作したあとからプログラムを解析し問題点を発見することは極めて困難である。システマティック故障を低減するため、ソフトウェアだけではなくハードウェア回路の設計も含め、様々な開発手法や検証手法が用いられているが、いずれも様々な設計プロセスを経てのちプログラミングを実施する。そのほかプログラムに発生したバグの影響が全体に波及しないよう、メモリや CPU の処理時間を分離独立させるような構造的な方策も用いられる。いずれもプログラミング段階ではなく、開発開始時の設計のルールの明確化やその準拠、基本的な構造設計（アーキテクチャ設計）が、システマティック故障への主たる対策となる。

ランダムハードウェア故障は、ノイズや素子の劣化および半導体素子へのソフトエラーなどが要因となる。コイルや抵抗コンデンサのような部品では、ランダムに発生する故障といえども、故障発生時の影響が確定的である。たとえば炭素系抵抗素子であれば、ショートモード（短絡）、オープンモード（断裂）が主な故障モードであるため、それが発生した時の影響を回路で特定することが容易である。このような分析を FMEDA（Failure Modes Effects and Diagnostics Analysis）と呼び、この作業の結果を踏まえ、故障時の影響を局所化する回路や自己診断の仕組みを搭載する。故障時の対策は、あらかじめ回路設計やソフトウェア設計に組み込まれておく必要がある。プラント制御に用いる制御システムでは、CPU や FPGA のほか入出力装置や通信 I/F なども含め数千点の電子デバイスを使用する。一般に現実の商品開発においては基板サイズや CPU の処理能力に余裕は設けない。これらの故障の影響を極小化する機能は回路設計の初期段階で計画しておかなければ、製品量産の段階で、後から組み込むことは極めて困難である。

このようにコンピュータを用いた制御や保護の仕組みは、本質安全および本質的安全とは開発設計のプロセスが異なる。安全性を担保するための対策は製作段階や製造品質の管理だけではなく、構想設計など開発初期段階からの人的・組織的な準備や技術検討が重視される。このようなコンピュータを用いた制御や保護の仕組みの開発から製造や

将来の保守までの要件をまとめた国際規格が機能安全規格である。機能安全規格は、IEC 61508[15]を基本規格として様々な分野のセクター規格に広がっている。セクター規格とは基本規格に基づき、特定の製品分野に特化した条項が追加され整理された規格である。図 1-6 はセクター規格の例である。

機能安全規格 IEC 61508 は 2000 年に初版が発行されたが、航空宇宙分野の規格や、原子力発電所の規格、鉄道規格、燃焼や高温高圧を扱うプロセスの規格など、それ以前から欧米で蓄積されてきた産業分野の安全システムの規格を網羅し整理する形で制定された。例えば現代の欧米をはじめ関連の各国での鉄道建設では、RAMS 規格（Reliability, Availability, Maintainability and Safety）と呼ばれる規格群（EN 50129 など）への準拠が求められる。それは IEC 61508 のセクター規格である。また乗用車では ISO 26262 への準拠が義務づけられるが、これも IEC 61508 のセクター規格である。

IEC 61508 およびセクター規格は、前述の ISO 12100 と同様欧米市場に輸出する際には不可欠な規格になっている[15],[17]-[25]。たとえば冷凍食品の陳列ケースなどのような商用家電に使用されるインバータモータを例とすると、欧州への輸出で取得が必要な CE マーキングでは IEC 60335 という機能安全のセクター規格への準拠が該当する。また米国も UL マークがなければ米国に持ち込むことができないが、同様に IEC 60335 と同内容の UL 60335 への準拠がなければ UL マークを取得できない。すなわち税関を通過できない。

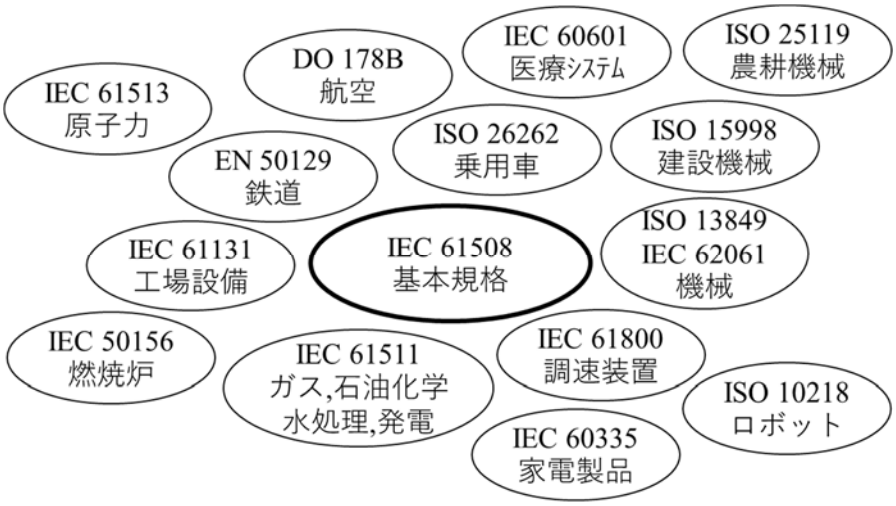


図 1-6. 機能安全基本規格 IEC 61508 とセクター規格

商品の機能や意匠デザインおよび価格における競争力は商品販売に不可欠であるが、一定規模の動力やエネルギー源を持つ機械製品では、安全保護の仕組みに機能安全規格への準拠がなければ、輸出そのものができない分野が広がっている。より効率的な機能安全への準拠の仕組みやアプローチが求められる。本論文で主たるテーマとして取り上げる背景である。

機能安全規格はいずれも、SIL（Safety Integrity Level）という指標を用いて、安全や保護の機能が持つべき信頼性の程度を表す。必要な SIL のレベルは設備や機械のリスク分析に基づくが、その評価方法は分野ごとに異なるためセクター規格の中で取り扱われる。たとえば石油化学プラント分野で適用される IEC 61511 では、図 1-7 のようなマトリクスと HAZOP（Hazard and Operability Study）という手法が用いられる。HAZOP は対象のプラントプロセスの温度や圧力、流量などに注目し、その値が正常な状態から極端に逸脱した状況を網羅的に想定し評価する手法である。ガイドワードと呼ばれる単語群を用いて異常な状況を洗い出してゆく。ガイドワードとは、「None」「Less」「More」「As Well as」「Reverse」などの言葉を、プロセスの物理量の異常状態に結び付けた連想を用いて、実際に発生しうる危険な事象を洗い出す手法である。その作業を繰り返し、災害に結びつく項目があれば何らかの設計変更や保護機能での対策を規定する。その危険な状況一つ一つに対して、過去の経験や同業種の事故事例などから、マトリクスのスコアを設定し必要な SIL の値を決定してゆく。ポンプやタンク、熱交換器などすべてのプラント機器に出入りするプロセス媒体の特性量を網羅的にチェックするため、その作業量は数カ月に及ぶ膨大な量になることもある。

被害程度	頻度指標合計 (F+P+W)					
	1-2	3-4	5-6	7-8	9-10	11-12
CF	NR	SIL1	SIL2	SIL3	SIL4	NO
CE	NR	NR	SIL1	SIL2	SIL3	SIL4
CD	OK	NR	NR	SIL1	SIL2	SIL3
CC	OK	OK	NR	NR	SIL1	SIL2
CB	OK	OK	OK	NR	NR	SIL1
CA	OK	OK	OK	OK	NR	NR

C：被害程度
CF（壊滅的）
CE（広範囲）
CD（深刻）
CC（重大）
CB（軽微）
CA（無視できる）

頻度指標
F：暴露の頻度（0~2）
恒常的：2
間隔的：1
偶発的：0
P：回避可能性（1~0）
回避不可：1
回避可能：0
W：危険状態の頻度
1回/1年以上：9
1回/3年 ：8
1回/10年：7
1回/30年：6
1回/100年：5
1回/300年：4
1回/1,000年：3
1回/10,000年：2
1回/100,000年：1

図 1-7. IEC 61511 のおける SIL の決定マトリクス

リスクそのものの数学的な意味はあまり重要ではない。産業用インフラの設備では、災害リスクは甚大なため SIL2 または SIL3 の領域となることが一般的だからである。重要な点は SIL が設備全体に一律に与えられるものではなく、膨大な作業量のリスク分析の結果に基づき設定された保護機能個々に与えられるというプローチにある。抽出したリスク分析とそこから得られる保護機能項目の網羅性が低ければ、設備や機械の安全性を高めることができない。このため、国際規格の評価においてはリスク分析に関するエビデンスの提出を求めることがある。この初期のリスク分析の網羅性が低ければ、もしくはそのリスク分析のエビデンスが残っていなければ、そのあとの安全性設計のすべての評価が停止されるほど重要な前提となる作業である。ここで求められるエビデンスとは、リスク分析の結果「クリティカルだ」と評価した項目だけの一覧ではなく、抽出過程で網羅したイベントや想定事象のすべての項目とその評価記録である。

設備や機械のリスク分析と同様、保護システムそのものの障害解析も網羅的に行うことが求められる。抵抗器やコンデンサなど安全機能に関係するすべての電子デバイスの故障モードを列記し、すべての故障モードについてシステムが担う安全や保護の機能に与える影響を分析する作業 FMEDA の実施が必要である。その結果に基づき、システムの安全機能が動作しない確率（平均的な危険側不動作確率）を計算する。

この網羅的なリスク分析のアプローチは過去の経験や事故事例の対策の蓄積や、技術者個人の改善活動に基づく日本の安全設計アプローチと異なるが、現在の国際的な取り決めのトレンドである。機能安全規格のみならず、制御セキュリティ規格などでも同様なアプローチを規定している。これは変化や進化が急激で激しい電子デバイスを用いたシステムによって安全を担保するためには、過去の経験を踏まえたうえで、対象のシステムを網羅的に解析することが大切であると考えるからである。このようなアプローチが日本の従来のアプローチよりも優れているかどうかは別として、国際貿易においては、このようなアプローチと実施した証明（エビデンス）がなければビジネスができないのが現実である。これが鉄道や航空機のような設備から、ロボットなどの産業用機器や、商用家電、医療機器にまで広がっている。スケーラブルな機能安全対策の産業用コントローラのアーキテクチャを本論文にてとりあげる背景である。

以上の現代の産業用コントローラに求められる要件を整理すると以下の4つの観点にまとめることができる。

（1） 高度な制御理論の実現のための、大容量の計算処理能力。

（2） さまざまな規模に適用できるスケーラビリティ。

（3） モデル言語での制御回路設計保守。

（4） 機能安全への対応。

1.6 産業用コントローラにおける CPU の問題

上記のような要件を満たす産業用コントローラの演算処理装置として CPU が現在も使われている。本節では現代の CPU が産業用コントローラの演算処理装置として大きな問題を抱えている点を指摘する。

1980 年以降 30 年間以上、主に動作クロックの周波数を上昇させることで CPU の性能は向上してきた。これは産業用コントローラの基礎的なプログラムは変更する必要なく、新たな機能が追加できることを意味する。これまでの CPU 性能向上のアプローチは、産業用コントローラにとってとても望ましいアプローチである。なぜならば、産業用コントローラは長期の保守保全や維持が必要であるが、1980 年代の CPU のプログラムも、2010 年に登場した CPU で容易に動作させることができる。ソフトウェアの継承性の良さが産業用コントローラの分野で CPU の適用が拡大した大きな要因である。

近年の CPU のコア当たりの動作クロック周波数は数 GHz で上限に達している[3),(4)]。CPU の性能向上は動作クロックの周波数ではなく、1 つの CPU に搭載されるコアの数の拡張によってもたらされている[3),(4)]。図 1-8 は CPU の周波数の向上のカーブとコア数の推移である。コア数による CPU の機能向上の戦略は産業用制御システムのコントローラに使用するうえで、いくつかの問題を発生させている。第一は発熱の問題である。能力の高いマルチコア CPU の発熱は極めて高い。CPU の電力消費について、様々な解決方法が提案されてはいるものの冷却機構が不可欠である[43)-(45)]。産業用制御システムでは長期のメンテナンスフリーが必要だが、エアコンなどの冷却機構は定期的な保守や部品交換が必要でありメンテナンスフリーの障害となる。

第二にマルチコア化による産業用コントローラの複雑化である[6),(7)]。たとえば、風力発電設備の場合、落雷や系統事故によって系統と切り離されるタイミングと、回転トルクを遮断

するタイミングと同一にする必要がある。このタイミングのずれは機械本体に大きな物理的なストレスとなり機械機構の破壊につながる可能性がある[8]。このため、制御システムは、ハードウェア、OS およびアプリケーションにわたって、詳細なタイミング設計を行うことが不可欠である。マルチコア CPU の場合、マルチコアを正確に動作させるプログラムの記述は極めて難しい[8]-[12],[64],[65],[66]。マルチコアのアーキテクチャ毎に処理の優先順序や内部バスの使用の優先順位の決定アルゴリズムが異なるため、専用の命令を使用することや、それぞれの構造に基づいたプログラム設計が必要だからである。このため複数のコアを仮想的に 1 つのコアにみなせる仮想化ミドルウェアを搭載したり、専用のリアルタイム OS を搭載したりすることで、制御システム設計者の負担を軽減する方策を取らざるを得ない[67]-[69]。

システム内部に複数のミドルウェアが介在する構造は開発にかかる負担を軽減する一方で、システムの動作を詳細に解析することを困難にしている。産業用コントローラでは、品質と信頼性および長期の連続稼働性の確保ため、可能な限りホワイトボックス化を求める。ミドルウェアが多く存在し構造が複雑なマルチコアは、産業用コントローラとして扱うには制御関連事業者にとってとても負担が大きい。

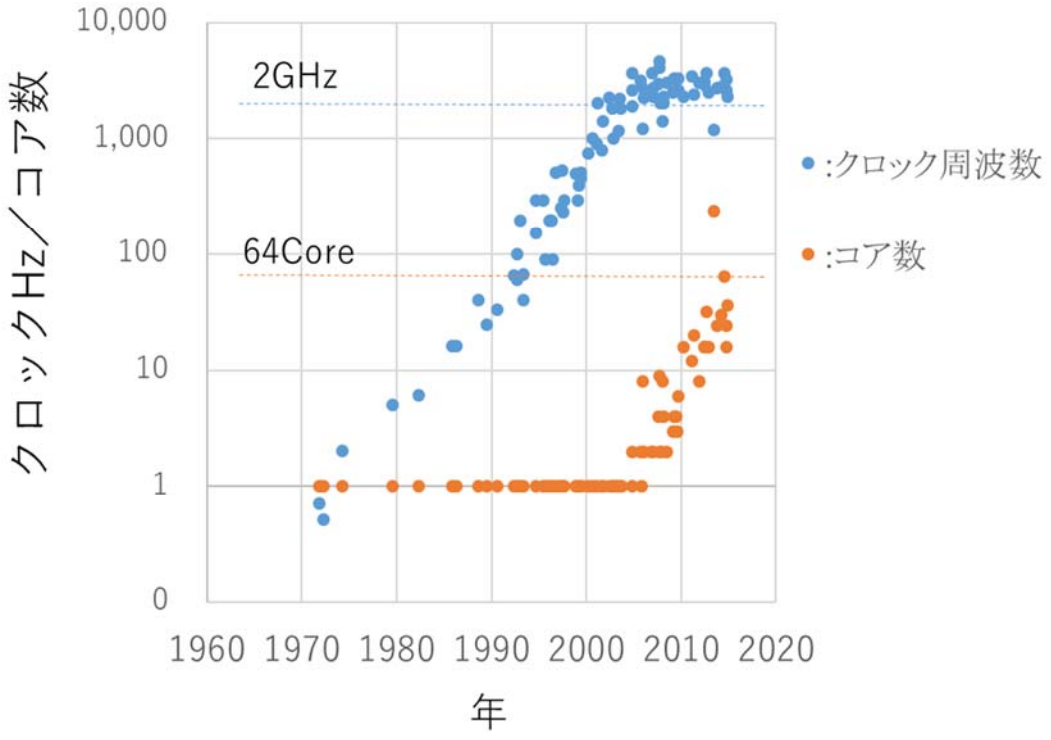


図 1-8. CPU 動作クロック周波数とコア数の推移

第三にマルチコア CPU の製品アーキテクチャ変化のサイクルが短いことである。マルチコアのアーキテクチャは製品ブランドによって大きく異なる[10]。 CPU 型式を変更する都度、マルチコア相互の連携を CPU の構造に沿ってソフトウェアを再設計しプログラムコードを生成しなければならない。またマルチコア CPU の実現方法には多様性があるため、CPU 内部のアーキテクチャには様々なものが製品化されている。さらに一つのアーキテクチャが継続して製品化される期間が極めて短い。CPU の上で動作するプログラムコードが短期間で再利用できず再設計を余儀なくされること、すなわちソフトウェア資産が継承できないことは、産業用コントローラを扱う現場では極めて大きな問題である。

以上のことをまとめる。産業用コントローラに今後も CPU を適用し続けた場合、以下の問題が避けられない。

（1） 発熱が極めて高い。サーバ向け CPU が主流。冷却機構の維持が必要。

（2） 周波数限界によるマルチコア化。仮想化等複雑さ増。解析が困難。

（3） 製品のアーキテクチャ変更サイクルの短期化。ソフトの継承が困難。

今後、自動車の自動運転など、さらに多くの分野で産業用コントローラに役割が広がり、高性能な制御や保護の仕組みが必要となる。CPU に依存せず、発熱が低くスケーラビリティに富み、そして長期のアーキテクチャの継承が可能な産業用コントローラを実現することは、制御対象の分野を超えた緊急かつ重大な課題である。

1.7 FPGA の可能性

近年、産業用電子デバイスに FPGA の適用が広がっている[70]-[72]。FPGA に次のような特徴があるからである[73]-[75]。

（1） 発熱が小さい。単位ロジックあたりの消費電力が低下しつづけている。

（2） 構造が単純。大小さまざまな製品がある（スケーラビリティが高い）。

（3） 製品間の言語継承性が高い。長期の保守や維持が可能である。

たとえば Xilinx 社の FPGA は単位ロジックあたりの消費電力だけではなく、単位ロジック当たりの価格も低下している。図 1-9 は Xilinx 社が公開している自社製品の性能や消費電力などの推移である。縦軸は 1990 年時点の性能数値を1とした相対指標である。

図から明らかのように、ロジック回路の搭載量と処理速度は増加している。今後も性能向上を期待できる。消費電力量は低下している。産業用制御システムにとって熱処理は極めて大きな問題である。発熱が低ければ、ファンや冷却設備など定期的なメンテナンスが必要な機械機構を備える必要がなく、長期の運用や稼働率の向上に有利だからである。

さらに FPGA は HDL という言語を用いることで、製品タイプや製造メーカに依存しないシステム開発が可能である。HDL (Hardware Description Language)とは、 FPGA の接続回路の設定データを作成するための記述言語である。この特徴により FPGA は前節で示した CPU の問題を解決する可能性がある。FPGA で産業用コントローラを構築できれば、CPU に依存せず長期的な製品維持と処理能力の向上を両立することができると期待できる。

なお、FPGA の一部の製品では FPGA の論理ブロックの中に、CPU コアと同様な機能ブロックを搭載したものがある。そのような内蔵演算コアは、CPU 同様に性能向上をマルチコア化により実現しているため、第 1.5 節に示したマルチコア CPU に関する課題は解決されない。これに類する FPGA の利用形態はここでは CPU と類似の問題を持つため、前述の課題の解決策とはならない。

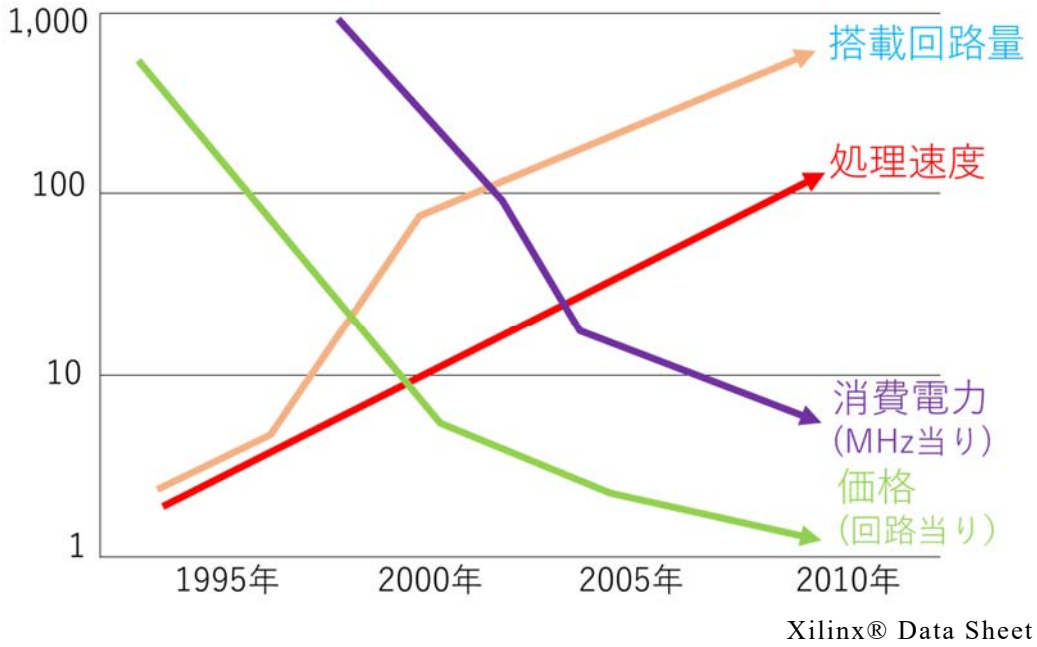


図 1-9. FPGA の性能や消費電力等の推移

1.8 FPGA を産業用コントローラに適用するための課題

前節では CPU の代替として FPGA が有望であることを述べたが、FPGA を産業用コントローラに適用するにはいくつかの課題がある。産業用コントローラに必要な要件は第 1.4 節より以下の4点である。

（1） 高度な制御理論の実現のための、大容量の計算処理能力。

（2） さまざまな規模に適用できるスケーラビリティ。

（3） モデル言語での制御回路設計保守。

（4） 機能安全への対応。

第1項および第2項は FPGA としての基本的な特徴で解決することは前節より明らかである。第3項および第4項は FPGA にとって課題となる要件である。初めに、第3項モデル言語での制御回路設計とオンラインでの修正における FPGA の課題を示す。

MATLAB などのモデル言語では、作成した制御演算アルゴリズムやモデルアルゴリズムを変換ツールによって、直接 FPGA 言語（HDL）に変換することが可能である。このような変換機能は、すでにソフトウェア商品として利用できる[76]-[81]。このためすでにモータや電源装置などの小型の製品では、モデリング言語により制御アルゴリズムが記述された FPGA による制御コントローラが採用されている。しかし産業用コントローラ一般では、かなり小型製品での適用にとどまっている。

FPGA の直接の配線回路として FPGA が起動時に読み込む不揮発メモリに記録される。制御演算アルゴリズムの修正は不揮発メモリの書き換えと FPGA の再起動を伴う。このような専門性の高い作業のために、産業用コントローラを設置しているそれぞれの現場に FPGA 回路の技術者を派遣し実施することは現実的な解決策ではない。半年など長期間を要する試運転の期間中プラント現地に FPGA 回路技術者を拘束しておくことは経済的にも妥当ではない。また試運転の途中に制御アルゴリズムそのものの変更が必要な場面もある。FPGA は、わずかな回路修正であっても合成時に配線が大きく変化し、回路のタイミングが変化する可能性がある。設備の試運転段階では変更の影響を極小化することが不可欠であり、そのような作業を実施することは困難である。

FPGA の回路構成を動的に変更させる研究がなされており、一部で製品化されている[80]。この対策は FPGA の不揮発メモリに書き込む作業や再起動の時間を短くする効果が

あるものの、やはり論理合成による変更を伴う。このため設備現場での使用は困難である。モデル言語のブロック図で記載した制御アルゴリズムが直接 FPGA の回路となるのではなく、アプリケーションとして動作する構造が必要である。

次に、第4項に示した機能安全への対応が困難な点について述べる。機能安全とは、現代の産業用コントローラにおいて準拠が不可欠な国際規格である[15](付録 1-1 参照)。これは CPU や FPGA などの LSI 内部パターンピッチが微細化したことにより、ソフトエラーの発生が無視できなくなってきたからである[82)-(84)]。エネルギーの高い荷電粒子の衝突は宇宙のような特殊な環境下での問題であったが、近年では地表においても発生が疑われるようになった。自動車のシステム異常など人命にかかわるケースも発生している[14),(85)]。CPU は、メモリ、レジスタ、キャッシュ、スタック、プログラムカウンタおよび演算コアという、機能的な構造が自明であり、プログラムコード内にこれらの機能的構造ごとへの対策を搭載することで、信頼性への対策が可能となっている[15),(20),(30),(31)]。プログラムコードがあるメモリであれば、単位領域ごとに CRC を計算し付加することで意図しない変更を検知する[15),(20),(30),(31), (86)-(88)]。プログラムカウンタの障害を検知するためには、サブルーチンなどの処理の単位領域ごとに固有の ID 番号を比較することで、意図したプログラムの順序で演算しているのかを確認する手法がとられる[15),(20),(30),(31)]。

FPGA は生成される内部配線接続が様々であり、画一的な信頼性アプローチを決定することが難しい。一部の FPGA 製品で機能安全対応と称するものが発売されている[89)]。これは内部に CPU コアを搭載し CPU への機能安全対応を構造的に組み込んだものである。FPGA の論理回路に対する機能安全対策ではない。

機能安全規格への準拠を最も確実に主張する手法として、FPGA回路を多重化する手法があげられるが、単純な多重化は共通要因故障の影響のため、全体の信頼性の向上には大きくは貢献しないとされている[90)-(94)]。FPGA 回路の多重化に加え多様化が必要である[95)]。しかし回路の多様化は開発コストの増加が著しく、産業用コントローラを扱う事業体として現実的には許容できるアプローチではない。

以上のように、FPGA には CPU が持つ課題を解決できる特徴があるが、産業用コントローラとして使用するうえで、「モデル言語による制御回路記述とオンライン修正」と「機能安全対応」の2つの観点で課題があり、適用が進んでいない。

1.9 むすび

マルチコアのコア数によって CPU の性能が向上することは、産業用コントローラにとって問題が多いことを示した。FPGA はそれらの問題を解決する可能性を持つが、産業用コントローラに必要な機能を実現するためには、いくつかの課題があることが明らかとなった。「モデル言語による制御回路記述とオンライン修正」と「機能安全対応」の2つである。本論文では、制御システムにおける今後の一層の発展や進化に貢献することを目指し、これらの課題を解決するFPGAによる制御演算コントローラのアーキテクチャを提案しその有効性を確認する。

次章ではモデルベースの制御設計に適した産業用コントローラのための新しい動作原理 TDB(Two Dimensional Bus)を提案する。第3章では高い機能安全の達成には不可欠な多重化した FPGA 回路の多様性を生み出すために新しい手法 DTM(Diverse Technology Mapping)を提案し、効果をシミュレーションにより明確する。最後に第4章にて TDB および DTM を備えた制御コントローラの新しいアーキテクチャ SGC(Safety Green Controller)を提案し、国際規格に基づく評価を行い、実現性と信頼性を明らかにする。

付録 1-1 機能安全規格の準拠の要件

リスク分析によって必要な安全や保護の機能とその機能が持つべき SIL の値が決定することで、その機能を搭載するシステムに求められる SIL の値が決定する。たとえば火力発電所では、30 個前後の保護機能が抽出される。それらを1つの保護システムで実現する場合、その中の最も高い SIL の値が保護システムに必要な SIL と考える。SIL が高い保護機能とは、それだけリスクの高い事象に対抗する機能のため、より信頼性の高いシステムが求められる。このため規格では SIL の値に応じて、システマティック故障およびランダムハードウェア故障のそれぞれに対して低減するための方策が定まっている。

システマティック故障に対しての要件は、開発プロセスだけではなく製品の検討段階から廃棄までのライフサイクル全体で定まっている。初回の製品開発が適切でも、電子部品の製造中止による変更がシステムの安全性を損なう可能性もあるからである。IEC 61508 では、図 A-1.1 のような活動それぞれについて、詳細に取り組むべき要件がある。本論文の対象ではないため詳細は割愛する。後述するランダムハードウェア故障に対する故障確率計算などの技術的な要件は、このような管理プロセスがあらかじめ厳格に定まった上での作業結果であることが前提である。基本的に「設計完了した」後に、規格への準拠を主張することは極めて難しく、海外の規格認証機関ではそのような場合は審査を拒否することも多々ある。この点は日本の製造メーカは注意しておく必要がある。

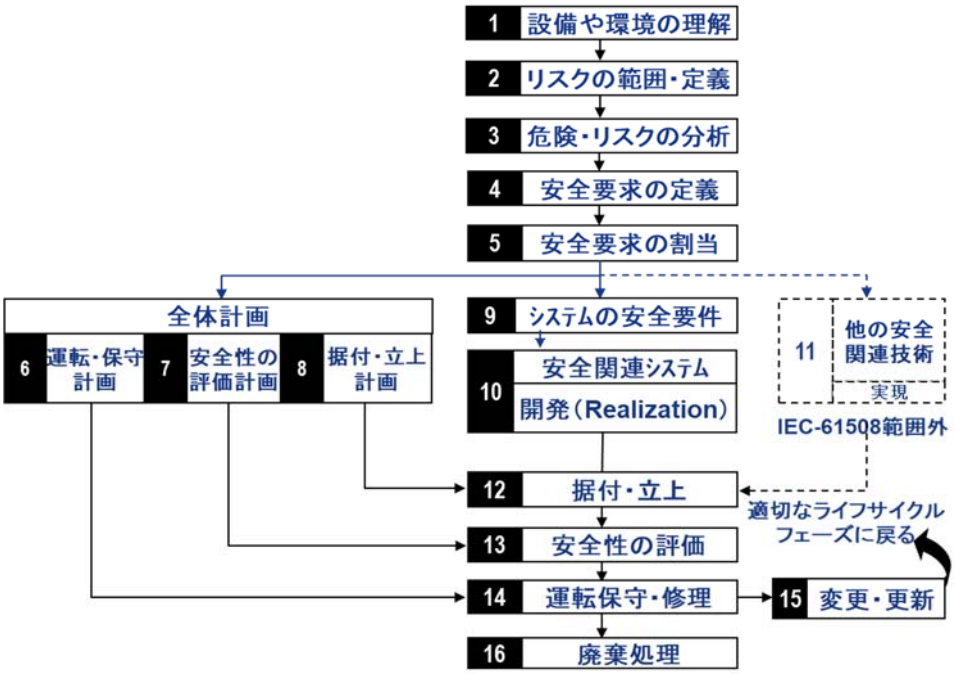


図 A-1.1. IEC 61508 がカバーする製品ライフサイクル

ランダムハードウェア故障に対する要件は、システムの構造的な評価と関連する電子デバイスの故障率を用いたシステムの故障確率による評価である。SIL が高い保護機能とは、それだけリスクの高い事象に対抗する機能のため、許容される不動作確率が、SIL のレベルに応じて決められる。たとえば機能安全規格では表 A-1.1 のような基準を設けている。石油化学プラント分野で適用される IEC 61511 では、危険な事象は頻度が少ないため、要求あたりの平均的な不動作確率 PFDavg を用いた要件が設定されている。

表 A-1.1. IEC 61511 における SIL と PFD の関係

Safety Integrity Level (SIL)	Average Probability of a Dangerous Failure on Demand of the Safety Function (PFDavg)
4	$\geq 10^{-5}$ to $<10^{-4}$
3	$\geq 10^{-4}$ to $<10^{-3}$
2	$\geq 10^{-3}$ to $<10^{-2}$
1	$\geq 10^{-2}$ to $<10^{-1}$

この基準はとても厳しい。電子システムは機械システムとは異なり、定期的な現場での検査（定期検査）では、内部の電子デバイスすべての健全性を完全に確認することは困難だからである。電子デバイスの故障率が一定と考えれば、システムの不動作確率の時間関数 $PFD_{(t)}$は使用期間に亘って増加してゆく。たとえば一般的なパワートランジスタ 1 個あたりの故障率は 1 時間当たり 10^{-9} 程度である。これはとても小さな値であるが、このトランジスタの故障モードのすべてが安全機能の喪失（不動作）につながる場合、システムの使用期間を 30 年とすると、その期間の平均的故障確率 PFDavg≒故障率×30 年＝$(1.0\times10^{-9})\times(2.6\times10^{5})=2.6\times10^{-4}$ すなわち SIL3 となる。これでは数千個の電子デバイスを使用する制御システムは構築できない。このため、故障モードが不動作につながらないような回路を考案する、またはソフトウェアによる自己診断機能を搭載し故障時にシステムを安全状態に推移するようなメカニズムを搭載するなどし、機能安全規格の基準を満たすようシステムの構想設計を行うことが必要となる。多くの場合、SIL3 など高い SIL レベルを達成するためには、センサーや演算処理部分を多重化するような構造的な対策（アーキテクチャ設計）を伴わなければ、目標の PFDavg を達成することは難しい。

システムを多重化することによって、ランダムに発生する部品故障のようなエラーの影響は軽減されるが、機能安全規格ではその軽減の効果は、2 つの観点での評価を要件としている。1つは多重化システムに共通する共通要因故障であり、もう1つは多重化したサブシステムに潜在する検知できない危険側故障の割合である。

共通要因の故障とは、設計プロセスの共通性や、使用する電子デバイスの共通性、ソフトウェアの共通性のほか、バスや電源などの電気的な共通性も要因となる。たとえば同じプログラムを搭載したサブシステムで多重化した場合、同じバグが潜在すると予想できるため、そのバグが顕在化した場合に両システムが同じ誤った計算結果を出力することで、多重化の意味をなさない可能性が高い。これはプログラムのバグに限ることではなく、人の作業に起因するシステマティック故障すべてに当てはまる。開発の根拠となる仕様設定が共通でかつ間違っていれば、両方のサブシステムが同時に危険側不動作故障となりえる。

ランダムハードウェア故障はランダムに発生するため、サブシステムすべてに同時に発生する可能性が低いように思われるが、たとえば電源ラインが共通であれば同じ雷サージによって両サブシステムが同時に故障する可能性が高い。通常同時には発生しない多重化した電子デバイスの劣化による故障も、腐食性ガスの中では他方で発生した後の修復時間の間にもう一方も故障を発生する可能性がある。極端な環境の中では、ランダムに発生する電子デバイスの故障も共通要因故障となる。

機能安全規格では、共通要因故障の低減のためにサブシステム相互の多様化を求めている。相互に多様化すべき要件は多岐にわたり規格の中で規定されている。しかしその評価方法には明確な数学的な根拠はない。対策しておくことが望ましいとあるが、それをした場合どのぐらい多様性として効果があるのか、共通要因故障がどのぐらい低減できるのかは審査官や評価機関の判断となるのが実際である。しかし過剰な多様性をもつ開発は事業者に極めて大きな負担となる。使用する電子デバイスを2種類使用すれば、製品化後の初期不良のトラブルを 2 倍抱える可能性がある。設計工数も 2 倍かかり、部品在庫にかかわる経済的なリスクも 2 倍になる。このようなことから、作業工数を増加させず、かつ定量的に効果が主張できる共通要因故障対策が求められる。本論文のテーマとして取り上げる背景である。

多重化したシステムを設計した場合、さらに評価が求められる要件がある。危険側の潜在故障である。危険側とは、そのシステムが担う安全や保護の機能が働かない（不動作）の方向に影響するような故障である。潜在とは故障が発生しているにもかかわらず、その状態が観測できず、故障を保守する機会を逸するような故障である。一般に λ_{DU} と表記する。

電子デバイスの故障は、FMEDA という作業で回路全体に対して網羅的に故障モードをチェックし影響を分析しなければならない。その中で「発生時に検知できるか」「安全機能の喪失に至るか」を評価し、抽出した λ_{DU} をサブシステム全体で合計する。これを $\Sigma\lambda_{DU}$ とする。サブシステム全体の故障率の合計を $\Sigma\lambda_{total}$ とした場合、$\Sigma\lambda_{total}$ と $\Sigma\lambda_{DU}$ の割合を SFF（Safe Failure Fraction）という指標で示し、SFF と多重化の度合いによる SIL との関係に表 A-1.2 のような制約を設けている。

$$SFF = 1 - \frac{\Sigma\lambda_{DU}}{\Sigma\lambda_{total}} \qquad (A\text{-}1.1)$$

表 A-1.2. IEC 61508 における SIL と SFF／HFTとの制約

SFF (=Safe Failure Fraction)	HFT (=Hardware Fault Tolerance)		
	0	1	2
<60%	許容されません	SIL1	SIL2
60%-<90%	SIL1	SIL2	SIL3
90%-<99%	SIL2	SIL3	SIL4
≧99%	SIL3	SIL4	SIL4

HFT(Hardware Fault Tolerance)とは多重化の度合いである。HFT＝0とは多重化していないシステムを示す。たとえば HFT＝0で、かつSFFが60%より小さい、すなわち危険側潜在故障 $\Sigma\lambda_{DU}$ が大きい場合は、機能安全が必要となるリスク低減のシステムとしては使用してはならないことが示されている。HFT＝0のシステムによって産業用インフラで必要な SIL3 を達成するためには、SFF が 99%すなわち、回路に使用する電子デバイスの故障モードの 99%が発生しても、安全保護機能に障害を与えないことが明確であるか、または故障発生を検知する仕組みがあることが必要であることが示されている。HFT＝1とは 2 重化したシステムが該当する。この場合は SFF が 90%程度でも SIL3 を主張してよいことと

なっている。現実のシステム設計では SFF90%が実現性の限界である。SIL3 では多重化はほぼ不可欠な要件である。

以上のように、機能安全規格ではシステマティック故障とランダムハードウェア故障による安全や保護の仕組みの不動作確率を低減するための様々な要件が規定されている。機能安全規格への準拠をまとめると図 A-1.2 のようになる。機能安全規格とは、このような開発プロセスを踏むことで、システマティック故障およびランダムハードウェア故障に基づくシステムの不動作リスクを低減することを目指す規格である。

このうち、ランダムハードウェア故障に対する要件を満たすためには、多様化による多重化によって共通要因故障を低減し、検知できない故障（$\Sigma \lambda_{DU}$）を少なくすることが求められる。また規格に準拠した開発作業には多大な労力がかかるため、システムの規模や長期的な維持に影響しない効率的な設計アプローチが求められる。これらの課題を背景とし、本論文では FPGA による制御コントローラのアーキテクチャ SGC を提案し、機能安全規格に基づく評価を行う。

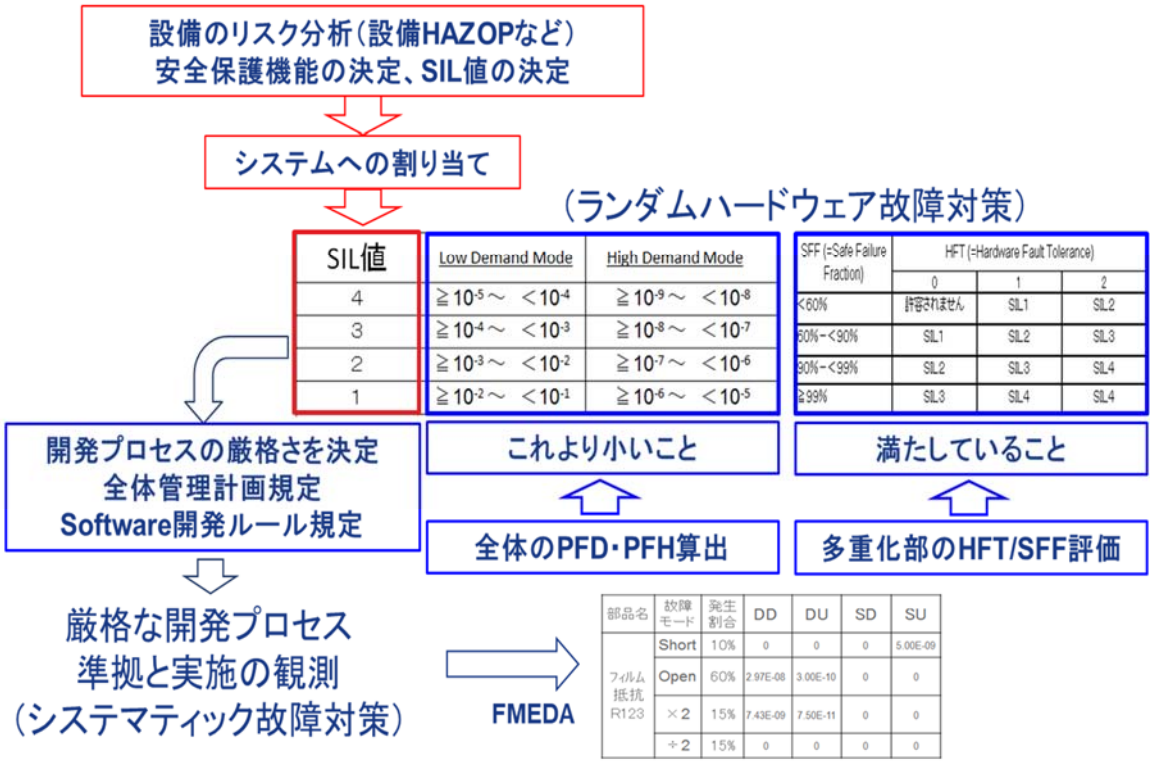


図 A-1.2. 機能安全規格への準拠フロー

第 2 章 修正可能な制御回路処理構造の実証と評価

2.1 まえがき

大規模なスマートグリッドシステムでは、電力系統制御において、バッテリと組み合わせたモデリングベースの最適制御が必要である[42]。複雑な制御を実現するためには高性能な CPU が必要である。再生可能エネルギーの大規模化、適切な系統オペレーションにも高性能な CPU による制御システムが必要である[79]。発電用ガスタービンや石炭燃焼ボイラなどでは、従来から高性能な CPU が使われてきた。発電プラントを構成する主要機械が複雑な制御を必要とするからである。

従来の再生可能エネルギー設備ではシンプルな制御が行われてきた[41]。このため制御システムは、PID 制御(Proportional-Integral-Derivative Control)とよばれる P(比例)I(積分)D(微分)により特性を調整する制御方式を実行できるように拡張したラダー言語や、簡単なファンクションブロック回路が演算できる PLC で実現されてきた。これらは古くから専用の ASIC で構成されている[32],[39]。このため長期の製品維持に十分に対応できてきた。しかしながら、再生可能エネルギー設備の制御は複雑化している。たとえば洋上風車では、MATLAB や Simulink によって翼の物理モデルに基づいたピッチ制御や油圧変速器制御を行う[40]。複雑な制御を実現するためには高性能な CPU が不可欠である。

高性能な CPU で産業用制御システムを実現することにおいて問題が顕在化している。現在の CPU の性能向上は、動作周波数ではなくマルチコアに依存している[11],[12]。このためリアルタイム OS や仮想化など、複雑なミドルウェア[63],[66]の搭載が必要になっている。また製品の世代交代により大きく CPU 内部のアーキテクチャや命令セットが変わり、プログラムコードの互換性がなく、過去のソフトウェアをそのまま新しい CPU で使用することは困難である。産業用コントローラにとって、長期のシステム維持が難しいことは大きな問題である。

そこで、構造がシンプルで長期の互換性に有利な FPGA が注目されている。すでに小規模な制御には FPGA は使用されている[13],[70]-[72],[76],[77]。またモデリング言語から FPGA 回路データの直接生成も市販ソフトウェアによって実現している[78]。しかし前述のような大規模な発電システムや、再生可能エネルギーシステムなどのインフラ設備に FPGA を適用するためには様々なハードルがあり実現されていない。

たとえば運転特性を左右するパラメータの変更や制御回路の改善は発電システムを運転しながら行う必要がある。これは発電システムの制御システムに FPGA を使用するにあたって数々のハードルのうちの一例である。さらに、大規模で複雑な制御コントローラには適応されてない理由がある。ソフトコアのようなアーキテクチャでは、性能向上を図るアプローチは CPU と同じアナロジーとなる。デバイスの動作クロックの向上によって性能向上を図るアーキテクチャでは FPGA の並列性を生かせない。大規模な制御システムを実現するためには FPGA の持つ特性を活かす必要がある。重要な点は並列処理である[78]。モデリング言語で記述された制御アプリケーションを FPGA 上で並列処理する特徴を持つ、新しいコントローラアーキテクチャが必要である。

本章では FPGA 内部に配置したプロセスエレメントをネットワークで接続したアーキテクチャを提案し、それにより FPGA が制御システムに必要な要件を満たすことができることを示す。また提案したアーキテクチャを実際に構築した試験体を用い、処理能力や必要になる FPGA 資源について評価し、本構想の妥当性を証明する。

2.2 発電プラント分野の制御システムの要件

2.2.1 CPU による従来型制御システム

制御アプリケーションは柔軟に作成できることが望ましい。柔軟性とともに、十分な処理速度で大量のロジックを処理できることが必要である。実際の発電向け制御システムは、多入力多出力が相互に関係する複雑な制御回路を持つが、アクチュエータの数に基づくフィードバック回路の数が全体の演算処理量の規模を概ね示すと考えることができる。処理性能や使用する処理資源の量を評価するために、1入力1出力に対する PI 制御コントローラを評価単位として使用する。このようなロジックを1ループと呼称する。

発電用の大型ガスタービンにおいて、実際の制御回路は1ループ制御ではないが、1ループ制御回路を単位とすると、必要な処理量はおよそ 1,000 ループから 2,000 ループ相当の規模である。これは調節弁だけではなく遮断弁なども出力に含めている。大型の石炭ボイラの燃焼制御装置も同様である。その演算周期は一般的に 50ms が必要とされる。将来の演算量の増大に備え、現在の 10 倍の処理能力を想定しておくべきである。従って、50ms で、およそ 2.0×10^4 ループを演算することが、発電設備分野の産業用制御システムに求められる要件である。高性能な CPU が求められる理由である。

再生可能エネルギー設備では、長期間の試運転を通し環境とのマッチングをはかり、性能の最大化を図ることが不可欠である[60]。石炭などを燃料にする発電システムでは、石炭の性質の違いによって長期間の試運転を行い、制御パラメータを最適化することが必要である[51]。たとえば実際に使用する石炭のカロリーや水分含有量によって燃焼系の動特性が変化するため、温度制御の積分や比例のパラメータを変化させることが必要である。最新の再生可能エネルギー、たとえば洋上風車などの大型の風車では、制御対象の機械の物理モデルを用いた最適制御を行う必要がある[40]。これらは MATLAB や Simulink のようなモデリング言語を用いて記述され、シミュレーションを経て実際の制御システムに適用される。これらの演算には ASIC ベースの PLC ではなく、高い能力をもつ CPU が必要である[52],[53]。またパラメータ調整とオンラインでの監視・変更の観点でも CPU には利点がある。高性能な CPU ならば、必要な演算周期の間に制御パラメータなどをオンラインで変更する機能を実現することが容易であるからである。

しかしながら高性能な CPU を産業用コントローラの演算処理装置として適用することに問題が生じている。これまでの CPU の進化はクロック周波数によって性能が向上してきたが、近年、シングルコアあたりのクロック周波数は 2GHz～6GHz が上限となり、マルチコアによる性能向上が一般的になっている[11],[12]。マルチコアアーキテクチャは複雑であり、製造メーカごとに構造が異なるばかりか、同じメーカのラインナップであっても数年の製品世代の変更によって大きく構造が変化し、障害解析や防止を困難にする。CPU の仮想化などの余分な CPU 能力が必要となる。余剰な CPU 能力の消費はより大きな発熱に繋がり、冷却設備など設備のコストを上昇させる[37],[43]-[45]。このように、発電システムの制御システムを CPU で実現することには長期的に大きなリスクがある。

2.2.2 FPGA による制御コントローラの実現

これらの問題を解決するため、産業分野で FPGA が注目されている[13],[73]-[78]。性能や搭載容量の増大、そして単位演算辺りの商品電力の低下や低価格化などのほかに、画像処理など従来 CPU で演算していたプログラムを FPGA の論理回路に合成する HLS が活用範囲を拡大しているからである[96],[97]。HLS (High-Level Synthesis: 高位合成)とは、C 言語などのプログラムコードを HDL データに変換する技術である。FPGA は製品の世代進化においても HDL の互換を維持するケースが多い。このことは長期の運用における互

換品供給にきわめて有利である。FPGA は構造がシンプルでありシミュレーションによって動作を完全に再現することができるため、機能障害が発生した場合、その原因解析や防止措置を行うことが容易である。

FPGA を制御コントローラとして用いるアプローチには2つのアプローチがある。1 つ目は制御回路を HLS によって用いて直接 FPGA の回路に変換する方法である。このアプローチは、画像処理や音響処理などの製品で広く実現している。ただし FPGA の回路資源を大量に消費することが予想される。もう1つの方法は、FPGA の回路資源の消費量を低減する制御コントローラ専用の演算処理アーキテクチャを構築し、そのうえで制御回路を実行する手法である。次節より、この2つのアプローチをシミュレーションにより比較評価する。

2.3 アーキテクチャの提案と評価

2.3.1 テストモデルと制御回路

本節では、モデルのシミュレーションによって、FPGA の制御コントローラとしての性能や消費する FPGA 資源を検証する。実験で使用する、タンクの水面制御のテストモデルと制御ロジックは、図 2-1 および図 2-2 のとおりである。1つの流量調節弁経由の水供給ラインと1つの排水ラインをもつタンクである。ここでは簡単のため、タンクの排出水量は排出弁にて一定の流量であるとする。タンクの水位制御は、水位に対する PI 制御によって水供給の流量調節弁を開閉することで実現する。この系の閉ループ伝達関数は次のようになる。

$$H(s) = \frac{Gc(S)\cdot Gp(S)}{1-Gc(s)\cdot Gp(s)} = \frac{\alpha kp\cdot S+Ki}{S^2+\alpha Kp\cdot S+Ki} \qquad (2.1)$$

発電プラントの制御回路を表現するために、古くから MATLAB や Simulink のような特殊な FBD 言語が用いられるケースがある。ここでは一般的なプラント向け制御システムの FBD 記述言語を評価する[32]。C 言語コードで表現できる機能ブロックを接続によって表現できる制御記述言語であれば、本アプローチは適用できる。図 2-3 は式(2.1)を実現する PI 制御回路を一般的な FBD 言語で記載したものである。

タンク水位モデル

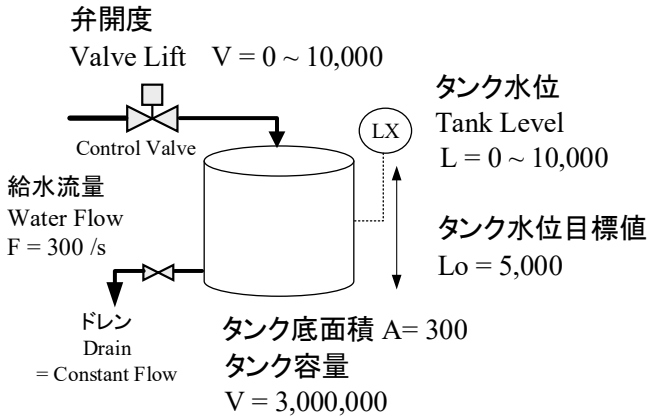


図 2-1．テストモデル／タンク水位モデル

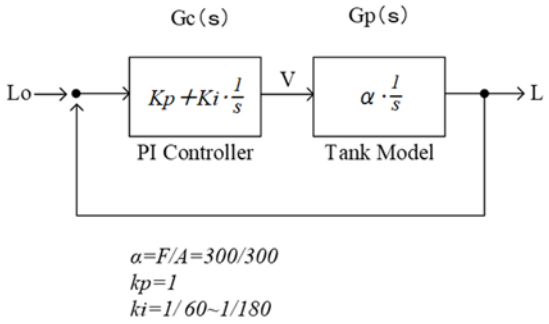


図 2-2．テストモデル／タンク制御モデル

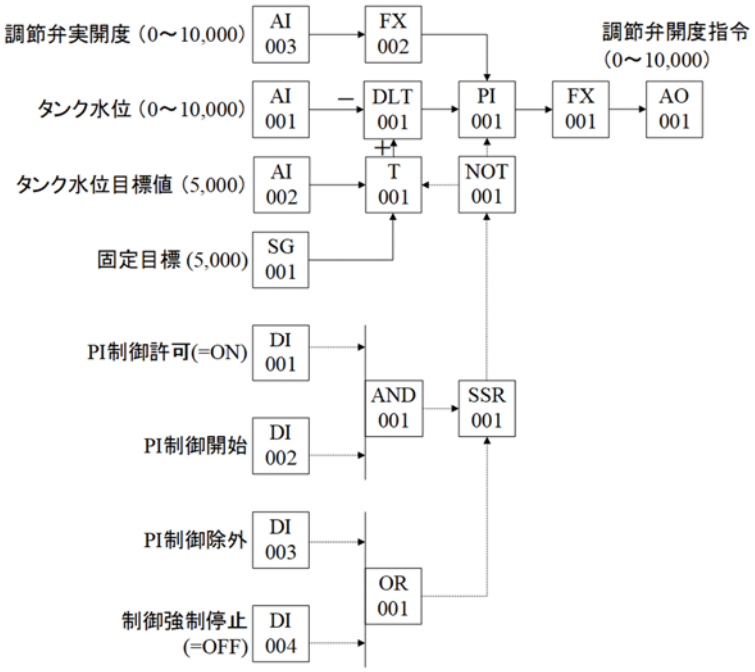


図 2-3．FBD 言語で記述した PI 制御回路

2.3.2 HLS による直接合成(Direct Synthesis)

制御アプリケーションの記述のために、一般に算術演算専用のモデリング言語を用いる。そのようなモデリング言語は MATLAB や Simulink のように、機能ブロックの接続で構成されている[76]。モデリング言語においては、接続線が関数の入力または出力信号を表現する。CPU 型の制御システムでは、この機能ブロックの連結をC言語のサブルーチンで表現し逐次実行する。発電システムの制御で使用する機能ブロックの C 言語コードを HLS で FPGA 回路に変換したものを FPGA に配置し、機能ブロック相互の接続を FPGA 内部の配線で実現する。このイメージが図 2-4 である。

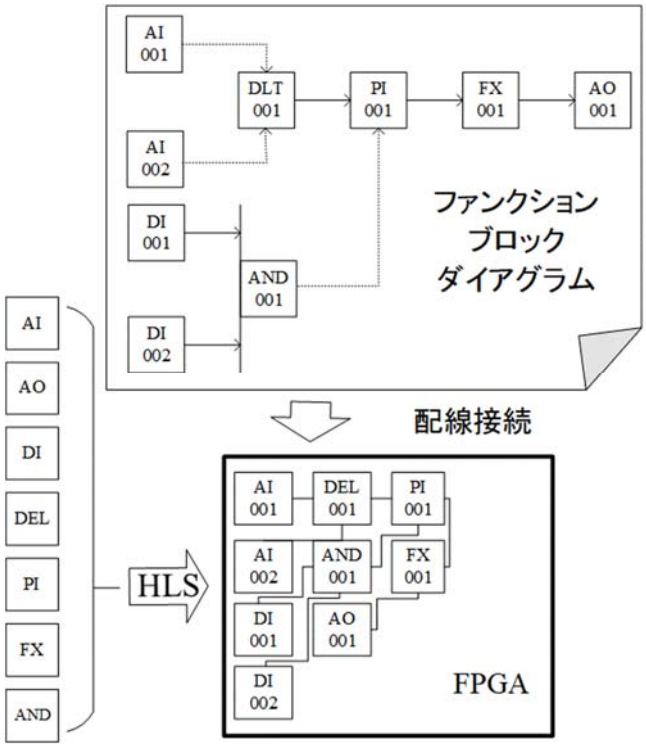


図 2-4. 直接接続による制御回路の実現

このアーキテクチャによる処理能力と FPGA 資源の使用量をシミュレーションにより確認した。処理能力は 1 ループあたり 0.88μs であり、資源消費量は 783 スライスであった。この結果を実際の制御コントローラに置き換える。上位クラスの FPGA(Xilinx 社 Virtex-7 XC7V200T)は 305,400 スライスを持つため、1 ループコントローラを 390 ループ搭載することができることが分かる。この資源消費量では要求される制御コントローラの処理容量を満たすことができない。省資源のアーキテクチャが必要である。

2.3.3 Two Dimensional Bus (TDB)アーキテクチャ

TDB は FPGA の省資源を達成するための制御コントローラ専用の回路アーキテクチャである。TDB の構造の特徴は次の3つで示すことができる。なお PE (Process Element)とは、FPGA 内部で特定の演算を処理するための、ひとかたまりの演算ブロックを示す。

（1） 機能ブロックのみを HLS にて HDL に変換し合成後、FPGA に配置する。

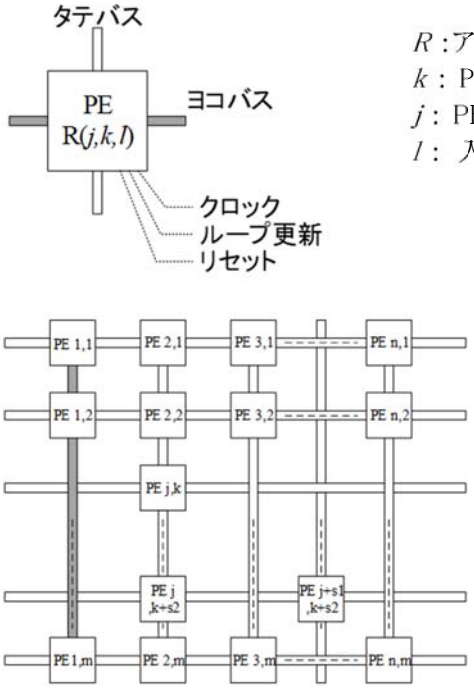
（2） PE は実行する順番をメモリに保持し同期信号のカウントにより演算を開始する。

（3） 各機能ブロックは FBD の接続情報から生成されたアドレス配列 *R* を保持する。

機能ブロックは特定の演算順序で演算し、その計算結果は通信によって、その出力値を入力値として使用する機能ブロックに伝達する。計算結果の伝達先の機能ブロックの ID をアドレスと呼称し、配列 *R* に保持する。この配列 *R* によって、FBD の機能ブロック間の接続を機能ブロック間の通信として実現する。TDB アーキテクチャの演算の順番と演算ブロックの接続は FPGA 内部の配線に依存しない。 それらは FPGA の合成なしに変更可能である。図 2-5 は TDB の構造を示している。各 PE はタテ方向のバスとヨコ方向のバスの二次元バスのマトリクスに配置される。ひとつのループの中の同じ種類の演算ブロックはタテに配置している。種類を示す番号を *j* とする。演算ブロックは種類ごとに ID 番号を持つ。その番号を *k* とする。またそれぞれの演算ブロックの入力信号の順列を *l* とする。つまり FBD に記述された演算ブロックの入力信号は、*j,k,l* にて特定することができる。*j,k* は使用する FPGA の資源にあわせて最大値を決定する。

クロックパルスのカウントにより、各機能ブロックに割り当てられたカウント番号で特定の機能ブロックが演算を行う。各機能ブロックを演算する PE は、データパケットのルーティングの処理の回路を持つ。演算結果はタテのバスに送信される。受け取ったデータの宛先が自分と同じ *j* のデータを受信し、他の PE はデータを破棄する。受け取った PE が自分あてではない場合、ヨコバスに対して転送を行う。受け取ったデータの宛先が自分と同じ *j,k* の PE はデータを採用し自身の入力値 *l* に値をセットする。

図 2-6 は信号伝達の例である。図 2-7 は TDB アーキテクチャを FPGA に展開した際の例である。



R :アドレス配列テーブル
k : PE 番号
j : PE 種類番号
l : 入力信号番号

図 2-5. TDB の構造

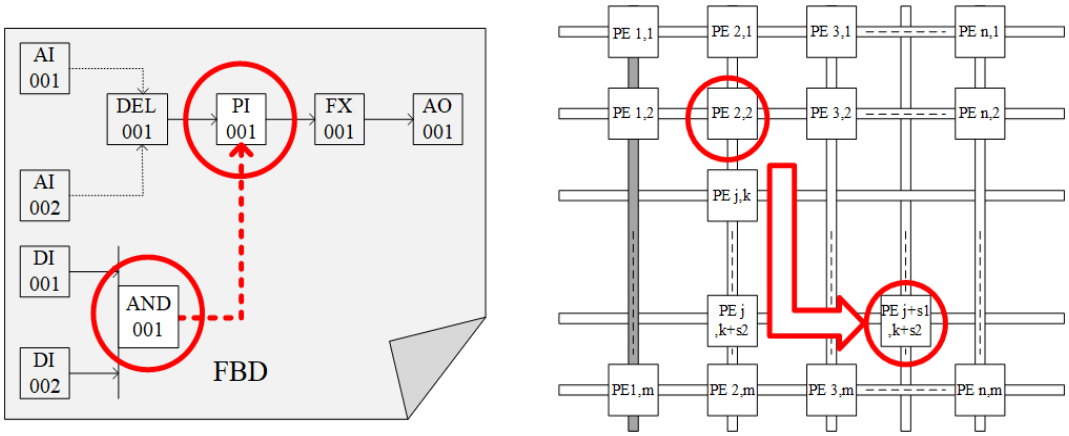


図 2.6. 信号伝達の例

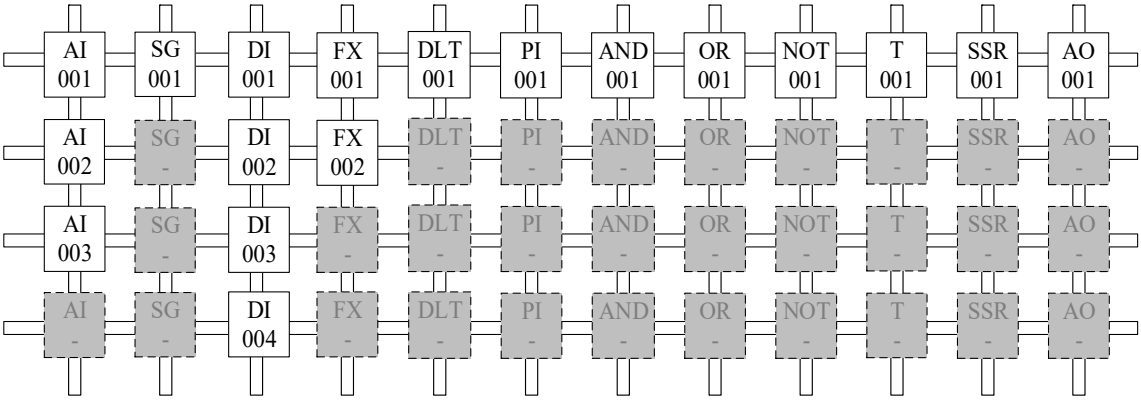


図 2-7. FBD ブロック図の FPGA 回路への展開

機能ブロックの接続情報は、配列 R の通信アドレスのテーブルによって保持されているため、配列 R を変更することにより、制御回路の変更が可能になる。FBD ロジックの1ループ分を演算完了したのち、機能ブロックのアドレス配列Rを次のループの接続情報に切り替え演算を行うことにより、同じ FPGA 資源を用いて異なるループの演算を実行することが可能となる。また次の演算周期の開始までの時間で配列 R を変更することにより、オンラインでの制御回路の変更も実現できる。図 2-8 はシミュレーション試験における制御動作である。図 2-1 に示すモデルに基づき、タンクレベル 5,000 を目標値として制御を行い、およそ 3,000 サイクルで整定した状態を示している。

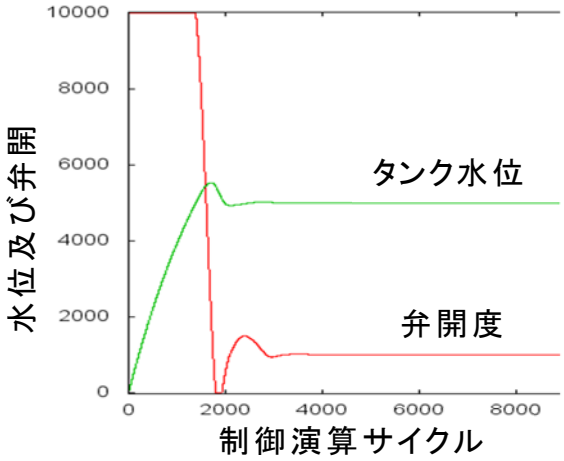


図 2-8. 試験モデルでのステップ応答

この通信に関する回路のオーバヘッドが演算処理のパフォーマンスにどの程度影響するのか、またその影響が実際の発電システムの制御システムの実現可能性を阻害しないかを評価する必要がある。このシミュレーション試験における TDB の処理能力および FPGA の消費資源を評価したところ、1ループあたりの演算処理時間は 2.36 μ s であった。また、使用した FPGA 資源は 4,402 スライスである。ある制御演算回路から別な制御演算回路に配列 R を切り替えるための必要時間は 10ns であった。したがって、配列 R の切り替え時間を含む演算処理時間は 1 ループ当たり 2.37 ms である。この結果から、50ms の演算周期が必要な制御演算では、2.0×10^4 ループに相当する演算を 4,402 スライスの FPGA 資源を再利用しながら実行することができることを意味する。また、FPGA 内に TDB アーキテクチャを複数配置し並列処理することにより、制御コントローラとしての処理能力はさらに向上する。制御演算を分割し並列処理するための条件については本節の付録に示す。

図 2-9 および図 2-10 は、HLS による直接合成の手法と TDB 手法における、FPGA の資源消費量を表している。図 2-9 より、HLS による直接合成の手法は、1ループを演算するためだけであれば使用する資源は少ない。一方で、図 2-10 より、TDB は FPGA 資源を再利用するため、4,096 ループを演算する際の消費量は極めて少なくなっている。

図 2-11 は、HLS 直接合成手法と TDB の演算実行時間の比較である。この結果から、TDB は HLS による直接合成の手法に比べ、およそ 3 倍の演算処理時間が必要である。これは PE 間の通信オーバヘッドによるものである。しかしながら、いずれのアプローチも 1 ループの実行時間について 50ms の要求を満たしている。

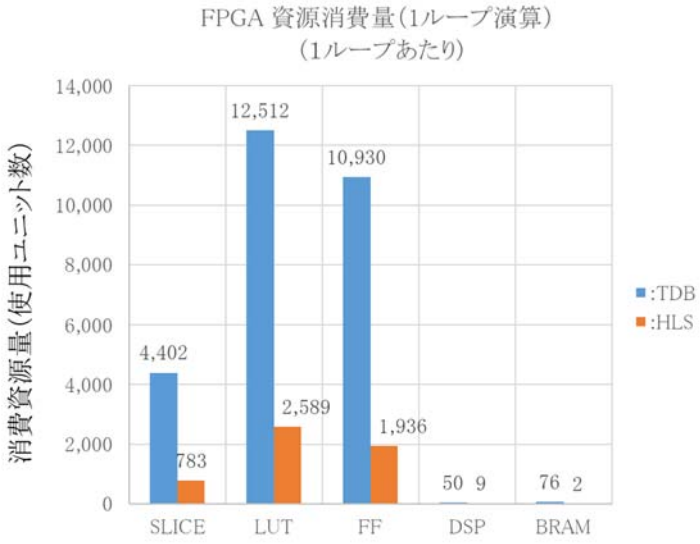


図 2-9. 1 ループでの FPGA 資源消費量

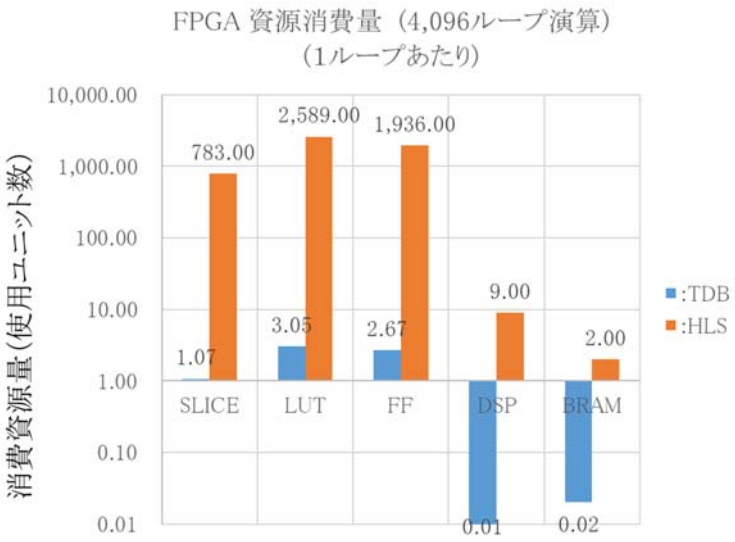


図 2-10. 4,096 ループでの FPGA 資源消費量
（1 ループあたり。縦軸は対数表示）

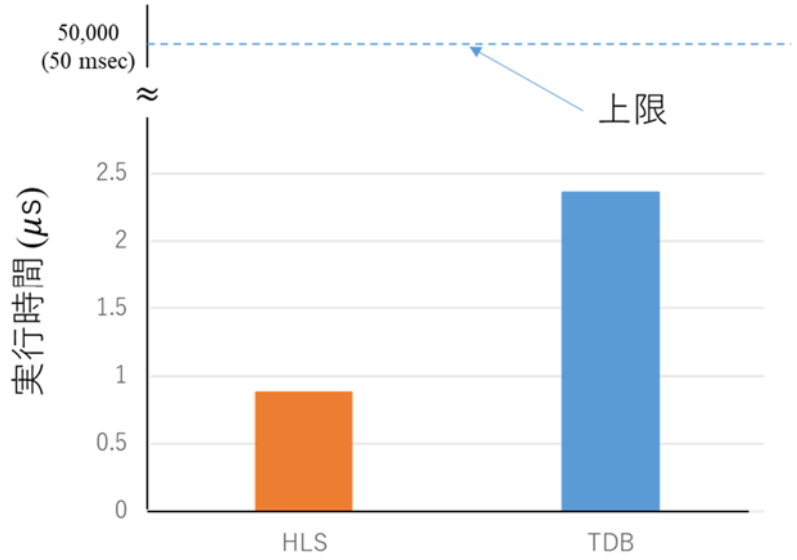


図 2-11. HLS 直接合成手法と TDB の演算実行時間の比較

2.3.4 TDB アーキテクチャの拡張についての考察

TDB アーキテクチャはバス型接続とマルチキャスト型通信を用いるため、1つの信号伝達シーケンスの期間はその信号がバスを占有する。効率が低い通信方式である。前節の通り1ループ制御回路を基準とした制御回路であれば、火力発電用ガスタービンを想定した規模の機能ブロック数でも通信オーバヘッドは無視できる。しかし伝達効率が低いバス型通信は、連結する機能ブロック間の信号数が大規模な制御演算には適さない。たとえばロボットの動力学計算などベクトルを変数とする行列演算が必要なケースでは通信オーバヘッドは顕在化する。

また機能ブロック数の規模の増加に伴い、バスに接続するノードが信号を送受信する同期タイミングの余裕が小さくなる(クリティカルパスになる)欠点がある。タイミング余裕が極端に小さなクリティカルパスは次章で示す通り、ノイズや電圧低下、ソフトエラーなどの環境からの影響で信号伝達にエラーを生じさせ、結果として演算エラーを生じさせる原因となりうる。バス構造では、発生した1つのソフトエラーが、複数の機能ブロック間通信に影響するため、エラーの影響が広くなる。

このような問題には、すべての機能ブロックを相互に 1 対1(P2P:Peer to Peer)接続し信号を伝達するアプローチが有効である。機能ブロック間の接続数が極めて大きくなることから、バス型の信号伝達ではなくシリアル型通信によって配線数を低減させる必要がある。

図 2-12 は構造のイメージである。ここでは PE は 1 つの機能ブロックと同等である。n機能ブロックのすべてが相互に接続するために必要な接続数を Cn とすると

$$\mathrm{Cn} = \sum_{k=1}^{n-1} k = \frac{1}{2} \cdot (n-1) \cdot n \qquad (2.2)$$

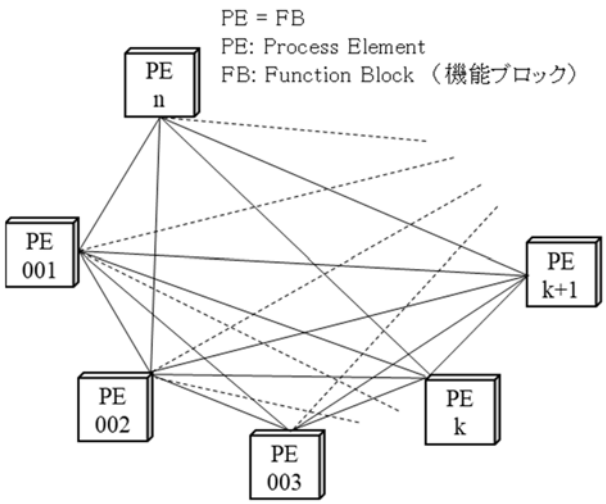


図 2-12. 機能ブロックの P2P 接続のイメージ

図 2-3 では 1 ループあたりの連結する機能ブロック数は 20 程度を想定している、ここではより複雑な制御演算回路を想定し、連結する機能ブロック数を 1.0×10^3 と仮定する。この場合、機能ブロック間の接続数C_{10^3}は、

$$C_{10^3} = \sum_{k=1}^{(10^3-1)} k \approx 1.0 \times 10^6 \qquad (2.3)$$

機能ブロック間をシリアル通信接続とすれば、大規模な FPGA 製品では制限となる配線リソース数ではない。しかし 999 個の接続先から 1 つの接続先を選択するマルチプレクサ回路のブロックが 1,000 個必要となることから、現在販売されている上位製品でも実装が難しい可能性がある。このため、本論文で示した TDB アーキテクチャを単位ノードとした演算グループを、P2P 接続したアーキテクチャで実現することが有効である。

図 2-13 はそのイメージである。PG(Process Group)とは TDB アーキテクチャで実現した演算グループの単位である。PE に相当する機能ブロックは TDB アーキテクチャで接続され1つの PG を構成し、PG が相互に P2P 接続されている。各 PG は並行演算することも可能であるが、TDB アーキテクチャにおける演算順序クロックシグナルを共有することで、異なる PG 間の機能ブロックを特定の演算順序で演算することも可能になる。

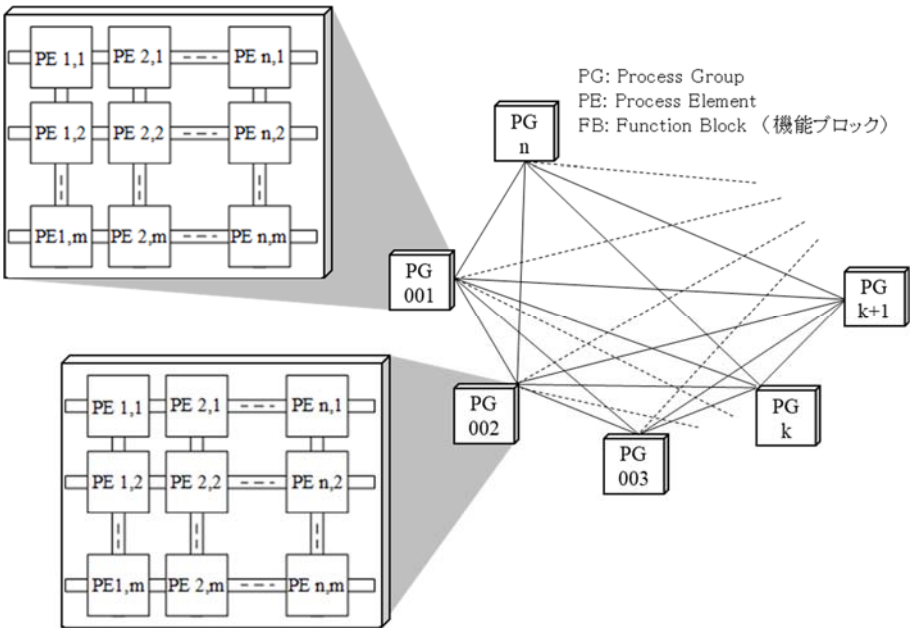


図 2-13. TDB 構造を単位とした P2P 接続のイメージ

FPGA は並列処理が可能であるため、P2P 型の通信効率は高い。また PG 間の通信経路にバスに類する共通部位がないため、前述のような環境からの影響による通信エラーが発生した場合も、影響を受ける通信経路が限定される。このように TDB アーキテクチャの欠点を補う効果が期待できる。

トポロジーの異なるネットワークを構成するためには、信号伝達のためのルーティングテーブルの構築や制御が必要である。ここで提案する構造は、IP(Internet Protocol)ネットワークにおけるレイヤー2とレイヤー3の階層構造と類似している。経路決定のためのルーティングプロトコルに、RIP（Routing Information Protocol）や、BGP（Border Gateway Protocol)を使用することで解決することが可能であろう。そのほか通信データのエラー時のリトライ処理の追加や、冗長な経路の障害時の切り替えには、TCP/IP のフロー制御やスパニングツリープロトコルなどの IP ネットワークの技術を適用することができるだろう。

P2P 接続は、ハードエラー（FPGA 素子にダメージが発生するような恒久的な故障）時に予備の PG に切り替えるなどのフォールトトレラント機能（待機冗長構造）を構成することがバス接続と比して容易である。これらの構造や機能の実装は、TDB アーキテクチャの成果を踏まえた今後の検討テーマである。

2.4 むすび

本章では、新しい FPGA の制御コントローラのアーキテクチャ TDB を提案し、その性能や FPGA 資源の消費量をシンプルなタンクモデルシミュレーションに結果にも続き評価した。その結果、TDB が 1 ループ制御の演算に使用する時間は 2.37ms であり、使用する FPGA 資源は 4,402 スライスであることが判った。TDB は同じ FPGA 資源を用いながら、配列 R を切り替えることで複数の制御演算を実行することができる。

切り替えにかかる処理時間のオーバヘッドを考慮しても、2.1×10^4ループに相当する規模の制御演算回路を 50ms で処理できる能力である。これは大規模な発電プラントの制御コントローラとして使用するうえで、十分な演算処理性能である。この演算処理性能は、FPGA 内部に TDB アーキテクチャを複数配置し並列処理をすることにより、さらに高めることができる。またさらに TDB がシステムの再起動をせずに、オンラインで制御回路の変更ができることも確認できた。

本章で提案した TDB アーキテクチャは、処理速度の観点、処理できる容量の観点、そして運用保守の観点など、産業用制御システムに FPGA を適用する際に解決すべき要件をすべて満たすことが明らかとなった。また TDB アーキテクチャは、処理する制御回路の機能ブロックの量に相応した FPGA デバイスを選択することで、小さな処理から複雑で大規模な制御まで広い範囲の制御回路に適用できることも大きな長所である。

またバス型アーキテクチャの問題点の整理と、それを補う P2P 型のアーキテクチャについて効果や実現性を考察し、本研究の今後の展開について述べた。

付録 2-1 FBD ブロック図の並列演算の条件

FPGA の並列構造を活用するため、適切な演算順序の条件について示す。機能ブロックは順番に演算し、おのおのの結果を接続先の機能ブロックに伝達する。その演算順番を演算順序と呼び $1\sim k\sim n$ で示す。ある機能ブロックに入力信号として使用できるすべての信号をベクトル表現したものを全信号ベクトルと呼称する。FBD ロジックでは機能ブロックを演算するごとに、次の機能ブロックの演算で入力値として使用することができる全信号ベクトルの要素数が増加する。演算前のロジックに存在する全信号ベクトルをx_0とする。ロジックの初期状態は、そのロジックの入力信号しか確定的ではないため、x_0は入力信号をベクトル表現したものである。同様にそのロジックの出力信号ベクトルを$\mathbf{x}_n$と表すこととする。

ある全信号ベクトルx_mを入力信号として演算順序 k の機能ブロックの演算後の全信号ベクトルをx_kとし、機能ブロックにより新たに確定し生成された信号だけで構成されるベクトルを$f_k(x_m)$とすると下記になる。

$$x_k = \begin{pmatrix} x_m \\ f_k(x_m) \end{pmatrix}, \quad 1 < \mathrm{k} < \mathrm{n}, \quad 1 < \mathrm{m} < \mathrm{n}, \quad n,m \text{ は整数} \tag{A-2.1}$$

$$x_k = \begin{pmatrix} x_m \\ f_k(x_m) \end{pmatrix}, \quad 1 < \mathrm{k} < \mathrm{n}, \quad 1 < \mathrm{m} < \mathrm{n}, \text{ かつ } n \text{ と} m \text{ は自然数} \tag{A-2.2}$$

この処理を次のような演算子$\tilde{F}_k$で表すこととする。

$$x_k = \begin{pmatrix} x_m \\ f_i(x_m) \end{pmatrix} = \tilde{F}_k * x_m \tag{A-2.3}$$

このとき、演算順序が i,j の2つの機能ブロックが順序jの機能ブロックの結果が順序iの機能ブロックの入力に使用されている場合は次のように書ける。

$$x_i = \begin{pmatrix} x_j \\ f_i(x_j) \end{pmatrix} = \tilde{F}_i * x_j \tag{A-2.4}$$

このときすべての機能ブロックで、$0 < \mathrm{j} < i \leq n$であるならば、常に$x_j \subseteq x_{i-1}$ であるため、

$$x_i = \tilde{F}_i * x_j = \tilde{F}_i * x_{i-1} \tag{A-2.5}$$

これは、演算順序1からnまでのすべの機能ブロックの順序演算において、不定となる入力値が存在しないことを示している。

$$x_n = \tilde{F}_n * \tilde{F}_{n-1} * \tilde{F}_{n-2} \cdots \tilde{F}_i \cdots \tilde{\mathrm{F}}_j \cdots \tilde{F}_2 * \tilde{F}_1 * x_o \tag{A-2.6}$$

上記を満たす演算順序が存在する。これを満たすロジックを1つの完全系列と呼ぶこととする。適切な演算順序によって演算すれば、演算後に計算結果を接続先の機能ブロックの入力値に送信することにより、適切な演算が実行できることを示している。

つぎに$0 < i < j \leq n$となる機能ブロックが存在する場合について考える。$x_j \supseteq x_{i-1}$であり、$\tilde{F}_i * x_j$を演算するための信号が不足する。つまり不定な入力信号がある機能ブロックの演算が存在することを意味する。このような場合には次の操作を行う。前回のプロセス演算サイクル完了後の全信号ベクトルを$x_{(t-1)}$とし、x_mを拡張したベクトルを$\tilde{x}_m$とする。

$$\tilde{x}_m = \begin{pmatrix} \mathrm{x}_m \\ x_{(t-1)} \end{pmatrix}, \text{ m は自然数であり、} 1<m<n \tag{A-2.7}$$

信号が不足する場合、該当する信号の前回値を用いることで、不定な信号を補完できる。補完した変数を用いた演算を$\bar{\tilde{F}}_i$とすれば、次の式を満たす演算順序が存在する。

$$x_n = \tilde{F}_n * \tilde{F}_{n-1} * \tilde{F}_{n-2} \cdots \tilde{F}_j \cdots \bar{\tilde{F}}_i \cdots \tilde{F}_2 * \tilde{F}_1 * x_o \tag{A-2.8}$$

該当するケースは、フィードバックなどの帰還回路である。このケースは式(A-2.6)と対比し不完全系列と呼ぶこととする。再生可能エネルギーなどのエネルギー設備において、制

御に用いるプロセス量の変化はそれほど急激ではなく、前回サイクルの結果で補完しても問題ではない。以上のことより、並列処理に関して次のことがいえる。

ある2つの演算順序により計算される系列を$\tilde{F}(x_p, x_{p-i})$、$\tilde{F}(x_q, x_{q-j})$とする。

$\tilde{F}(x_p, x_{p-i}) \Leftrightarrow \{x_p = \tilde{F}_p * \tilde{F}_{p-1} * \tilde{F}_{p-2} \cdots * \tilde{F}_{p-i} * x_{p-i-1}\}$ ，p、iは自然数　　(A-2.9)

$\tilde{F}(x_q, x_{q-j}) \Leftrightarrow \{x_q = \tilde{F}_q * \tilde{F}_{q-1} * \tilde{F}_{q-2} \cdots * \tilde{F}_{q-j} * x_{q-j-1}\}$ ，p、iは自然数　　(A-2.10)

(ケース 1) その全信号ベクトルが互いに独立な場合、互いの演算順序は計算結果に影響しない。即ち、FA と FB は並列演算が可能である。

$\tilde{F}(x_p, x_{p-i}) \cap \tilde{F}(x_q, x_{q-j}) = \emptyset,\ x_p \cap x_q = \emptyset$　　(A-2.11)

言い換えれば、$\tilde{F}(x_p, x_{p-i})$ と $\tilde{F}(x_q, x_{q-j})$は並列演算が可能である。

(ケース 2) $\tilde{F}(x_p, x_{p-i})$ と $\tilde{F}(x_q, x_{q-j})$の全信号ベクトルに共通因子がある場合でも、制御性を逸脱しなければ、前回の値を使用することができる。

$\tilde{F}(x_p, x_{p-i}) \cap \tilde{F}(x_q, x_{q-j}) \neq \emptyset,$

ここで $x_p \cap x_q = X = \{X_{1,} X_{2,} \cdots X_{r,}\} \neq \emptyset$, rは自然数　　(A-2.12)

$\tilde{X} = \{X_{1(t-1),} X_{2(t-1),} \cdots X_{r(t-1),}\},$

X(t-1)は前回の演算サイクルにおける値とする。　　(A-2.13)

$\bar{\tilde{F}}(x_q, \tilde{X}, x_{q-j}) \Leftrightarrow \{x_q = \tilde{F}_q * \tilde{F}_{q-1} * \tilde{F}_{q-2} \cdots \bar{\tilde{F}} \cdots \bar{\tilde{F}} * \tilde{F}_{q-j} * x_{q-j-1}\}$, q,j は自然数　(A-2.14)

$\tilde{F}(x_p, x_{p-i}) \cap \bar{\tilde{F}}(x_q, \tilde{X}, x_{q-j}) \neq \emptyset$　　(A-2.15)

言い換えれば、$\tilde{F}(x_p, x_{p-i})$ と $\bar{\tilde{F}}(x_q, \tilde{X}, x_{q-j})$ は並列演算が可能である。

これは、複雑な制御回路であっても、式(A-2.15)を満たすいくつかの系列に分割し、その系列ごとに並列演算をすることが可能であることを示している。さらに、単純な1ループコントローラも図 A-2.1 のように回路を系列化することができる。

このとき、系列 F_C と系列 F_D は独立しているため並列に演算することができる。この場合は式(A-2.15)を逸脱しない。つまり本アーキテクチャは、演算のパフォーマンスの向上を系列の分割で調整することができることを示している。この構造により、FPGA の並列性を活用しながら FPGA の再合成をせずに、オンラインで適切な制御演算性能を探索することが可能である。本件は複雑な回路を分割する手法について、さらに議論をすることができるが、本章の実験では、シンプルな1ループ制御を 1 つの完全系列として処理した場合の基本的な動作の確認を行った。

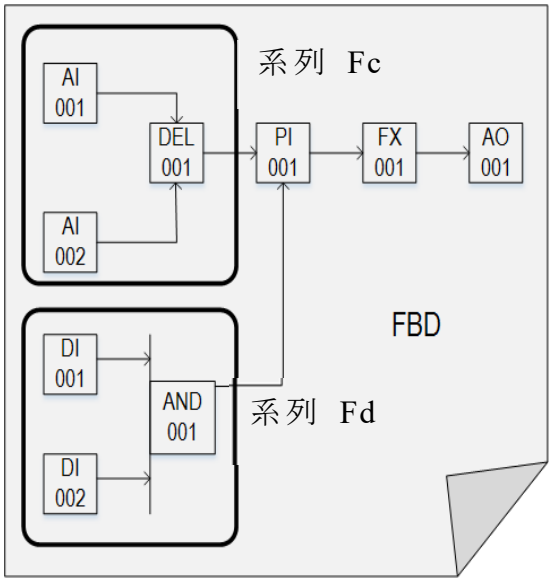


図 A-2.1. 2つの並列演算への分割例

第 3 章　多様性を持つ FPGA 回路の冗長化手法

3.1 まえがき

発電量に対して再生可能エネルギーの割合を拡大するため、現在よりもさらにアクセスが困難な場所に設置することが必要である。たとえば風力発電であれば洋上風力発電が該当する[60]。そのような場所では保守作業員が近づくことが困難であるため、制御システムには長期間のメンテナンスフリーが求められる。

近年の再生可能エネルギーの制御では高度な演算処理が必要になっている[40],[41]。CPU コア動作クロックはおよそ数 GHz で上限であり、それ以上の性能向上はマルチコアによってもたらされる。このため単純な PLC[32],[33],[39]ではなく、高速なマルチコア CPU を再生可能エネルギーの制御システムに採用しはじめている[11],[12]。マルチコア CPU をリアルタイムの制御システムに使用するためには、マルチコアに対応するリアルタイム OS や仮想化など、複雑なミドルウェアが必要である[65],[66]。このため、さらに高い性能が必要となっている。高い性能の CPU は発熱が大きいため、システムの多重化はさらに大きな発熱の原因となる。CPU の低消費電力化の試みも行われているが[98]-[100]、多くの場合高性能な CPU をフィールドで使用するためには専用の冷却設備が必要となる。冷却設備は定期的な保守が必要であり、メンテナンスフリーの障害となる。

以上のような CPU の問題は、FPGA を採用することによって解決することができる。たとえば FPGA は単位周波数当たりの発熱量が年々改善している。FPGA は処理能力も改善がつづいており、構造もシンプルである。OS のような複雑なミドルウェアを必要としない。さらに、デバイスの世代交代によっても、過去の設計結果を継承しやすい利点がある。エネルギー分野や鉄道など、長期運用するインフラの制御システムには大きなメリットである。そこで FPGA による制御システムの構築が注目されている[13][70]-[81]。制御システムは、ノイズや温度変化や電源電圧の変化などのストレスによって、ランダムハードウェア故障が発生する可能性が高い。

システムを不信頼に導く故障は2つの要因に分類できる[15]。1つは確率的な故障であり、主に物理的にハードウェア・素子に発生する故障である。電磁ノイズの影響や、ソフトエラーはこのカテゴリに属する。素子の劣化もこのカテゴリに属する。他方は確定的な故障である。主に設計間違いやプログラムのバグがこのカテゴリに属する。

長期の信頼性が必要な設備においては多重化構成とし、フォールトトレラントシステムを構築する[82],[90]-[92]。FPGA は並列処理構造のため、多重化による信頼性向上のアプローチは効果的である[90]。多重化したシステムの信頼性を評価する際は、多重化したサブシステムに共通に発生する故障を想定しなければならない。共通要因故障は多重化したサブシステムの両方に同時に発生するため、1つの故障が全体のシステムを故障に至らせるからである。たとえば各サブシステムのグランドが共通であれば、ノイズの影響で両方のサブシステムが異常になる可能性がある。このような原因の故障を共通要因故障と称する[15]。共通要因故障は、ランダムハードウェア故障にもシステマティック故障にもどちらにも起因する。十分に品質検査がされたサブシステムであれば、システマティック故障による共通要因故障は低減することができる。環境からのストレスが大きい再生可能エネルギーの制御システムにおいては、ランダムハードウェア故障に起因する共通要因故障がシステムにとって大きなリスクである。

多重化による信頼性向上では、共通要因故障の低減ため多様化手法が用いられる[83],[84]。多様化には3つのアプローチがある。アルゴリズムの多様化、コードの多様化そしてハードウェアの多様化である[95]。いずれのアプローチも開発コストや製造コストを大幅に増加させるため、再生可能エネルギーのシステムでは採用が難しい。アルゴリズム共通、コード共通で、さらにハードウェア(FPGA)も共通でありながら、FPGA のランダム故障に効果的で低コストを実現する多様化の手法を実現することが必要である。

そこで本書では、FPGA の TM(Technology Mapping)に注目した、多様性を持つ FPGA コントローラを生み出す新しい手法を提案する。この手法により、アルゴリズムやコードを多様化せずに FPGA 回路の多様化を実現できる。ハードウェアのランダム故障の発生に多様性を持たせることができ、多重化した FPGA の共通要因故障を小さくできる。この結果 CPU の代替となる FPGA による新しい制御演算コントローラを実現し、再生可能エネルギーの高度化や大規模化、メンテナンスフリーを実現する。さらにこの手法によれば、従来使用している実績のある”Proven-in-use”の合成ツールを使用することができるため、多くの使用実績が必要な再生可能エネルギーの制御システムに効果的である。この手法により、アルゴリズムやコードを多様化せずに FPGA 回路の多様化を実現できる。

次節では、多様化が必要となる理由として、二重化システムの MTBF と共通要因故障について説明する。第 3.3 節では本論文にて提案する多様化を実現する手法について説明する。第 3.4 節ではこの手法をシミュレーションにより評価する。またこの手法による MTBF への影響を考察する。第 3.5 節はまとめである。

3.2 共通要因故障と MTBF

3.2.1 共通要因故障

発電設備などの産業制御コントローラは、修復可能な多重化システムで構成される。図 3-1 は一般的な二重化した産業用コントローラの信頼性モデルである[15]。サブシステム(コントローラ-A および B)のうち、他方だけが異常となる場合は、もう片方が制御コントローラの機能を維持するため、サブシステムに固有の故障の影響は並列の関係で示している。共通要因故障とは両システムの結果を比較する回路の故障や、雷サージや電源変動など環境からの共通のストレスによってサブシステムが同時に異常になる故障、およびサブシステムに共通するプログラムコードのバグなどを想定している。共通要因故障は単独の発生で制御コントローラ全体を異常にするため、図では直列の関係となっている。

図 3-1 では、サブシステムの故障率をλ_{sub_sys}とし、その中に含まれる共通要因故障の仮想的な割合をβとした。βを小さくしなければ、多重化してもコントローラ全体の信頼性を大きく向上させることができない。このことを MTBF(Mean Time Between Failure)を用いて確認する。

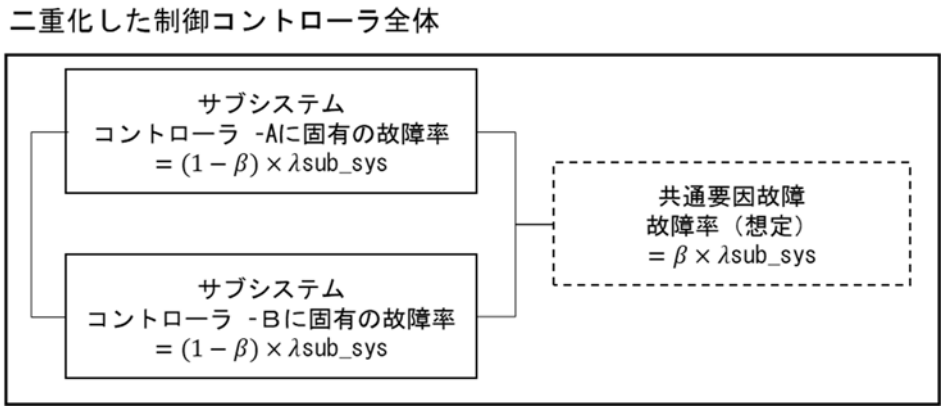


図 3-1. 二重化した制御コントローラの信頼性モデル

3.2.2 共通要因故障を考慮しない二重化システムの MTBF

MTBF(Mean Time Between Failure)とは、そのシステムが機能不全になるまでの平均時間間隔である。再生可能エネルギーの制御システムにおいて、稼働率を長くしメンテナンスが不要な期間を長くするということは MTBF を長くするということと等しい。図 3-2 は図 3-1 に示した修復系二重化システムにおいて、共通要因故障を考慮しない場合のマルコフモデルである。

このモデルにおける MTBF は次のとおりである(付録 3-2 を参照)。

$$\mathrm{MTBF} = \frac{1}{2\times\lambda_{sub\_sys}{}^{2}} \tag{3.1}$$

λ_{sub_sys} はサブシステムの故障率(1/hr)である。

インフラ設備のサブシステムのλ_{sub_sys} は一般的に、10^{-4}などの小さな値である[15]。共通要因故障を考慮しなければ、MTBF は 10^{8}時間(10,000 年)などのきわめて長い保守インターバルとなると期待できる。

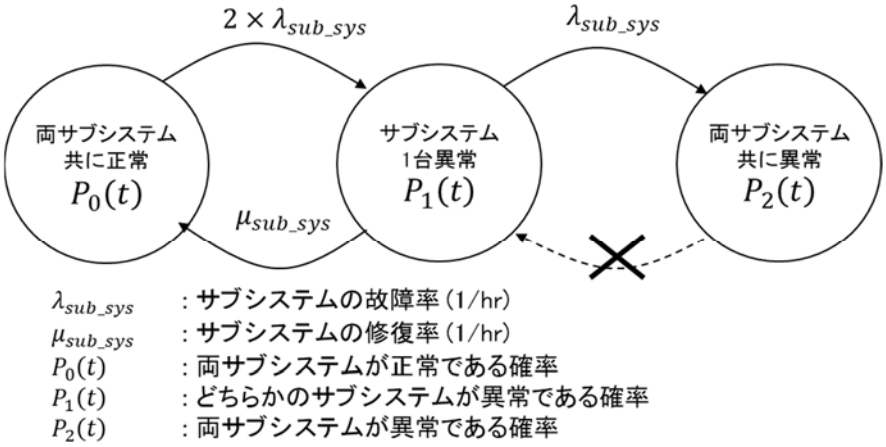


図 3-2. 共通要因故障を考慮しない二重化システムのマルコフモデル

3.2.3 共通要因故障を考慮した二重化システムの MTBF

共通の FPGA によって二重系システムを構築する場合の共通要因故障には、共通のクロックのドリフト、共通の供給電源の低下などの品質劣化、および素子材質の化学的な劣化などが考えられる。FPGA はすべての演算が内部の配線の信号伝達で実現されている。たとえば 32 ビットの数値を伝達する回路では 32 本の信号線の同期によって実現される。したがって配線上の信号波形の乱れは信号線間の同期の乱れとなり、想定したタイミング

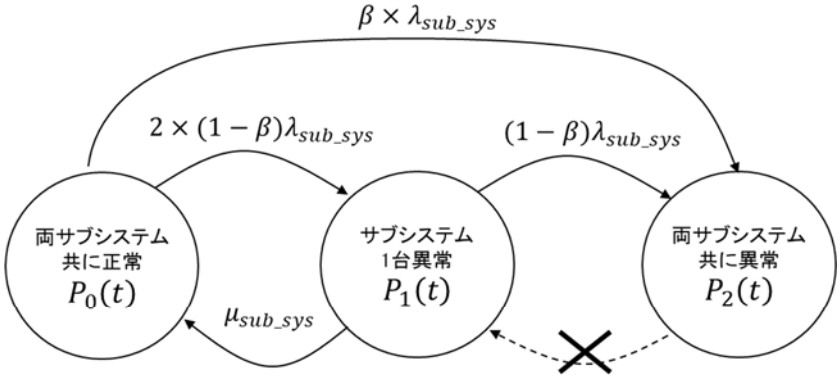
で正しい信号伝達ができない。この結果、演算にエラーが発生する。再生可能エネルギーは自然環境の中に設置するため、雷などの電磁波の影響は電源電位の変動などに現れるため、FPGA 内部の回路に共通に影響を与えやすい[8]。

FPGA 内部の回路における信号伝達のタイミングのマージンは、余裕のある場所もあれば余裕のない場所もある。タイミングのマージンが大きければ外乱による影響を受けにくい。タイミングマージンが小さければ外乱の影響を受けやすく演算のエラーの原因となる。同じ特性を持つ FPGA 回路の場合、配線経路やタイミングマージンが同じであるため、同様な演算エラーを発生させると考えられる。エラー結果が共通しているということは、多重化したシステムであっても、エラーが発生したことが検知できないことを示している。仮に FPGA の内部配線が大きく異なれば、エラーが発生するタイミングやエラーの演算結果が異なることが期待できる。共通要因故障への対策を考慮した多重化設計が不可欠である。

図 3-3 は共通要因故障を考慮した修復系二重化システムのマルコフモデルである。多重化したサブシステムに共通する故障の割合を β として表現している。このモデルにおける MTBF は次のとおりである(付録 3-2 を参照)。

$$\mathrm{MTBF} = \frac{1}{\beta\lambda_{sub\_sys}} \tag{3.2}$$

λ_{sub_sys} はサブシステムの故障率(1/hr)である。



λ_{sub_sys} ：サブシステムの故障率(1/hr)
μ_{sub_sys} ：サブシステムの修復率(1/hr)
β ：共通要因故障割合($0 < \beta < 1$)
$P_0(t)$ ：両サブシステムが正常である確率
$P_1(t)$ ：どちらかのサブシステムが異常である確率
$P_2(t)$ ：両サブシステムが異常である確率

図 3-3. 共通要因故障を考慮した二重化システムのマルコフモデル

共通要因故障ファクタ β の値は、機能安全規格 IEC 61508-6-Annex D [15]において一般的な取り扱い数値が規定されている。その値は 0.5%～5%の範囲である。インフラ設備のサブシステムに想定できる故障率（λ_{sub_sys}）は 10^{-4}などの小さな値であっても、βは 10^{-2} 程度であるため、共通通要因故障を想定した二重化システムの MTBF は 10^{6} 時間（＝100 年）程度にとどまる。風車など再生可能エネルギーに限らず、発電設備は多くの多重化した制御コントローラによって構成される。その数は 10 セットから 100 セットである。仮に 20 セットのコントローラで構成されていれば、設備全体のシステムの MTBF（年）は 5 年程度となる。これは発電設備の稼働率として十分ではない。特に保守が困難な場所に建設される再生可能エネルギー発電では不十分である。多重化によって、FPGA による制御コントローラの信頼性を実現するのであれば、共通要因故障割合 β をより小さくしなければならない。

同時に同じ演算エラーを発生させないためには、FPGA の内部配線が異なるサブシステムを組み合わせることが有効であると予想できる。FPGA の内部配線が異なるサブシステムの組み合わせを本論文では多様であると称する。

3.3 Diversity Technology Mapping （DTM）手法

一般的な FPGA の回路設計においては、はじめに HDL で記述されたコードを解析合成し論理的なネットリストに変換する。その後 FPGA 回路のための物理的なゲートの資源を割り当てることにより、FPGA の構成データをネットリストから生成する。後者の設計ステップをテクノロジマッピング（Technology Mapping）と呼称する。多様性を持つ FPGA 回路を生成実現するために、このテクノロジマッピングフェーズに注目する。

FPGA コンパイラはとても多くの回路パターンを生成する。配線長や処理時間などを鑑み、いくつかの配線パターンを評価し最適な1つのパターンを選別する。1つの論理的な回路に対して使用可能な数多くの異なる実際の配線パターンがある。様々な実際の配線パターンの組み合わせには大きな多様性がある。冗長化システムを構築する上で大きな利点である。FPGA の内部の資源の配置はコンパイラソフトウェアによって制御できる。図 3-4 は同じ HDL 記述から異なる資源配置を取ることによって、異なる FPGA の構成データが生成される様子を示している。DSP（Digital Signal Processor）とは FPGA の内部の資源の一つである。掛け算のような算術演算のための専用論理回路ブロックである。BRAM（Block

Random access memory)とは、論理回路の一部として使用することができる FPGA 内部のメモリ資源である。LUT とは小さなメモリ資源であり、通常は回路の構成情報そのものである。LUT 自身をメモリとして使用することもできる。DSP や BRAM の機能は LUT だけでも実現することができる。たとえば Xilinx ISE FPGA デザインツールは、命令によってそれらの資源の使用割り当てを指定することができる。次の記述がその例である。

(*use_dsp48 = “no”*) (3.3)

この記述によって生成される FPGA の物理回路には DSP 資源が含まれない。回路の中では、LUT だけによって演算が実行される。FPGA におけるこの資源割り当ての制限は HDL ソースコードの変更をせずに、生成される物理回路に多様性を生み出す。表 3-1 は Xilinx Kintex 7 FPGA にて掛け算のテクノロジマッピングをした例である。この結果のとおり、1つの HDL コードから異なる物理回路が生成されていることがわかる。

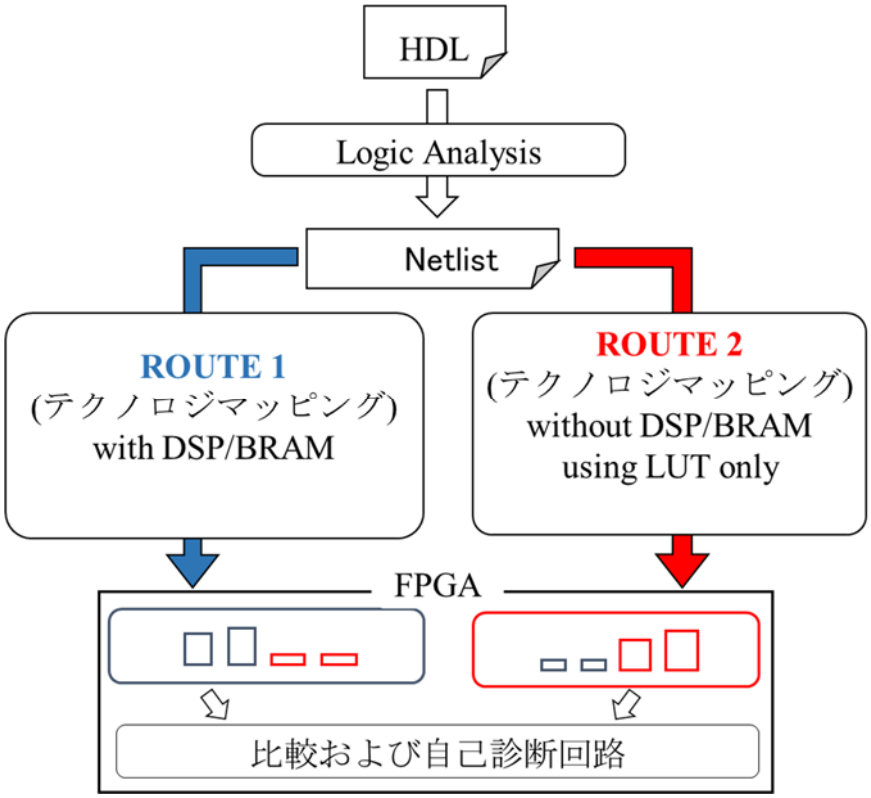


図 3-4. DTM のイメージ(多様性の生成)

表 3-1.生成された乗算回路の物理特性の比較

FPGAの物理パラメータ	掛け算 (MUL)		比例積分 (PI)	
	ROUTE 1 (DSP)	ROUTE 2 (LUT)	ROUTE 1 (DSP)	ROUTE 2 (LUT)
Slice Logic Utilization				
Number of Slice Registers	0	0	65	96
Number used as Flip Flops	0	0	65	96
Number of Slice LUTs	38	1203	126	3049
Number used as logic	38	1199	126	3041
Number using O6 output only	38	590	110	2243
Number using O5 output only	0	51	0	102
Number using O5 and O6	0	558	16	696
Number used exclusively as route-thrus	0	4	0	8
Number with same-slice carry load	0	4	0	8
Slice Logic Distribution				
Number of occupied Slices	24	316	43	877
Number of LUT Flip Flop pairs used	38	1203	127	3049
Number with an unused Flip Flop	38	1203	63	2968
Number with an unused LUT	0	0	1	0
Number of fully used LUT-FF pairs	0	0	63	81
Specific Feature Utilization				
Number of DSP48E1s	4	0	9	0
Average Fanout of Non-Clock Nets	1.61	2.68	1.75	3.18

このように生成された FPGA 回路は、異なる FPGA 資源を使うことによって、配線パターンが異なる回路が生成された。この多様性設計の手法によって、共通要因故障の影響が軽減されることが期待できる。この手法を DTM(Diversity Technology Mapping)と呼称する。DTM のもう一つの利点は、“Proven-in-use”であるコンパイラなどのソフトウェア製品にもとづくことである。“Proven-in-Use”とは、産業界において、多くの実績により信頼できる製品や技術を指す[15],[17]。再生可能エネルギーや発電などのインフラ設備において、とても重要な利点である[17]。

3.4 実験と評価

3.4.1 共通要因故障の模擬方法

多様性の評価の為の、環境ストレスの模擬手法とDTMの評価について述べる。共通要因故障の発生原理として FPGA 内部のクリティカルパスでのエラーに注目する。クリティカルパスとは、タイミングマージンが小さい配線経路を示す。たとえば電源電圧の低下は FPGA 内部の信号波形の劣化となり、信号伝達効率を低下させ信号伝達が遅延する。信

号の遅延は回路の動作タイミングをひっ迫させる。想定の時間内で信号伝達ができず意図した動作ができなくなる。この結果エラーを発生させる。また FPGA 素子内部の素材劣化による信号伝達効率の低下も内部の信号伝達の遅延をつながり、前述の電源低下と同じくエラーを発生させる。電磁ノイズの影響も信号波形のみだれとなり、信号伝達の遅延として回路の動作タイミングをひっ迫させる。これらはいずれも意図した回路の動作タイミングを乱すため演算結果のエラーとして顕在化する。

図 3-5 は共通要因による FPGA 内の信号伝達効率の低下がクリティカルパス部分でエラーを発生する様子を示している。またLUTのみで構成された回路とDSPを用いる回路では、異なる場所やタイミングがエラーを発生させる様子も併せて示している。ハッチングは外的ストレスにさらされている期間を表している。この期間では FPGA 内の信号伝達が乱され、信号が予想以上の遅延を発生させる。それにより演算回路のクリティカルパスにおいてエラーが発生しやすくなる。発生する共通要因故障の影響は、LUT のみで構成された回路と DSP を用いる回路とで異なると考えられる。LUT のみで構成された回路が 2 つある場合でも、配線ルートや配線長が異なる回路であれば、両者のエラーの発生状況やその影響は異なると予想できるが、その違いは DTM の差ほどではないと期待できる。

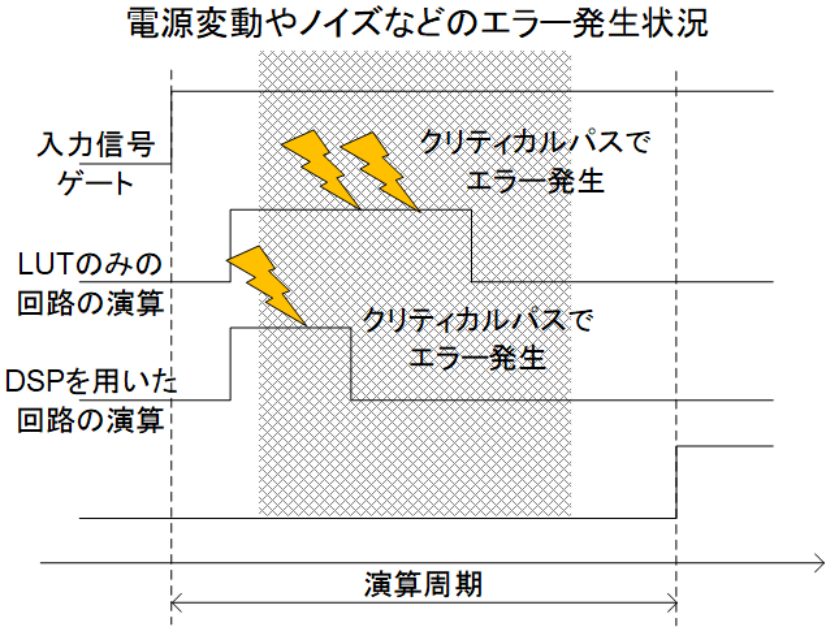


図 3-5. 共通の外部ストレスによる影響と DTM の効果

前述のような信号伝達のタイミングエラーは、FPGA の動作周波数を過剰に上昇させること(オーバークロック)によって発生させることができることを示す。図 3-6 は図 3-5 で示した回路をオーバークロックすることで、タイミングエラーを発生させている様子を示している。オーバークロックすることで、もともとの演算周期で行われるべき回路の信号伝達の時間的余裕が少なくなる。ノイズなどによる影響とは異なり、この場合では信号波形そのものが乱れることはない。しかし信号伝達の時間的マージンが不足することで、信号伝達エラーが発生しやすくなる。このようなエラーは、前述の影響と同様に回路のクリティカルパスにおいて発生しやすいと考えられる。外的な要因による信号波形の乱れによる演算エラーと、オーバークロックによる演算エラーは同様な性質を持つと考えた。これに基づき、本論文では、DTM の評価方法として、オーバークロックをさせた際のエラーの発生を用いて評価する手法を採用する。図にある通り、オーバークロックによるエラーにおいても、LUT のみを用いた回路と DSP や BRAM を用いた回路とではクリティカルパスが異なるため、エラー発生場所やタイミングが異なることが予想される。DTM の効果が期待できる。

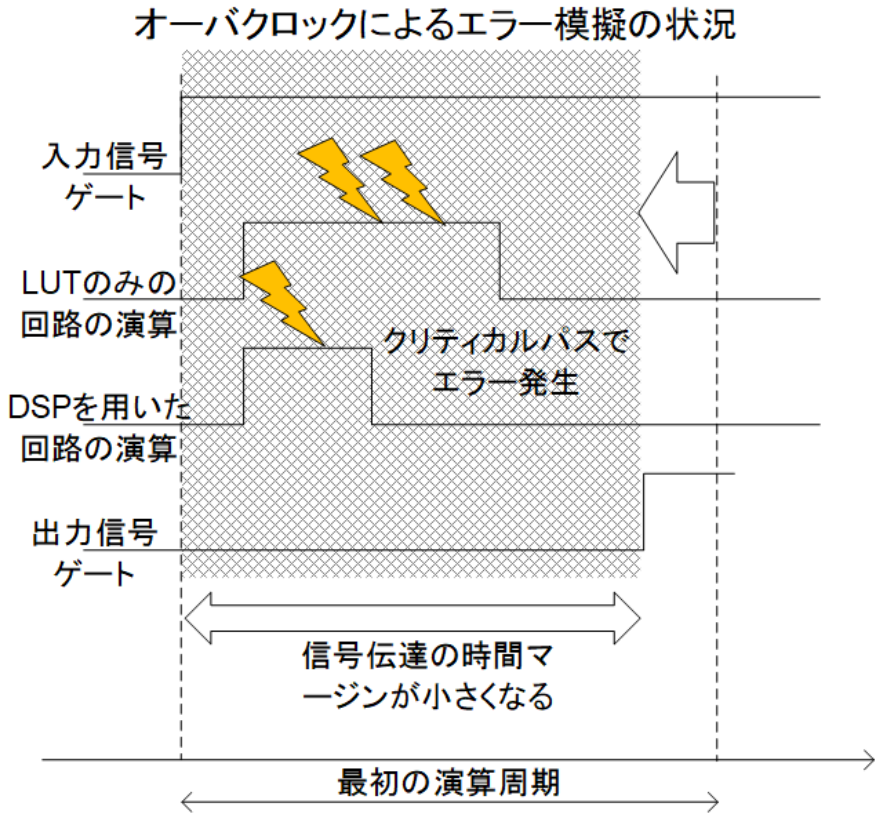


図 3-6. オーバークロックによるエラーの発生状況

3.4.2 テスト環境

テストのターゲットデバイスは Xilinx 社 XC7K325T-2(Kintex-7)である。使用する開発ツールは Xilinx ISE 14.7 である。表2のテストケースを用いて、シミュレーションにおける動作クロックをステップ的に変更する。2MHz ごとの各ステップの周波数で、ランダムな入力値を用いた演算を 10,000 回実施し、その結果を記録する。

各演算結果は別に用意した正しい計算結果(真値)と比較することで、10、000 回の演算結果のどれが演算エラー結果なのかを検証できるように準備する。多様性のある回路の計算結果と真値をそれぞれ比較し DTM の効果を評価する。

3.4.3 テストケースと結果

実験では DSP を使用する代表的な関数として、乗算(Multiple)、三乗(Cube)および積和(MAC)を選んだ。BRAM を用いる代表的な関数として、10 点ポイントの折れ線関数(PLI)を選んだ。それぞれの関数において、LUT のみで構成した回路とペアを組み、各ステップの周波数で演算した結果を記録する。その計算結果を用いて、それぞれの関数に対して下記の観点で演算結果を評価する。結果を図 3-7～図 3-18 に示す。

(a) エラー出現率(真値と異なる計算結果が出現した回数の割合)
(b) エラー検知率(エラー出現時に不一致によりエラーを検知できた回数の割合)
(c) エラー検知確率(エラー出現時の計算結果のペア間の検知の能力)

表 3-2. テストケース

テストケース No	関数	クロック周波数 (MHz)	非多様性ペア		非多様性ペア		多様性ペア (DTM)		評価観点		
			Mark	Route1/Route2	Mark	Route1/Route2	Mark	Route1/Route2	a	b	c
									図	図	図
1	Multiple	160~258	▲	DSP/DSP	■	LUT/LUT	●	DSP/LUT	3-7	3-11	3-15
2	PLI	180~248	▲	BRAM/BRAM	■	LUT/LUT	●	BRAM/LUT	3-8	3-12	3-16
3	Cube	80~298	▲	DSP/DSP	■	LUT/LUT	●	DSP/LUT	3-9	3-13	3-17
4	MAC	120~298	▲	DSP/DSP	■	LUT/LUT	●	DSP/LUT	3-10	3-14	3-18

図 3-7～3-10 は、各テストナンバーにおいて、a)の観点すなわちエラー出現率（%）でまとめたグラフである。DSP または BRAM を用いた回路は特定周波数より急激にエラー出現率が上昇し、LUT だけの回路はクロックの上昇にあわせてエラーの出現率が高くなっている。DSP や BRM は特定の機能のためのブロックである。それらの信号伝搬の遅れは入力信号の値には依存しない。

一方で、LUT にもとづく回路のクリティカルパスは入力信号の値に依存して変化する。本試験ではランダムな入力値を用いて各周波数で 10,000 回の演算を行っている。このためエラー出現率は周波数ごとに平滑化された値となる。このため出現率のカーブ配線や構造に根差した特定の形状ではなく、オーバークロック周波数に応じたなだらかなカーブを形成したものと考えられる。

このように FPGA 内部のメカニズムにより DSP、LUT、BRAM を用いる回路に、特徴的で異なるエラーカーブが出現している。

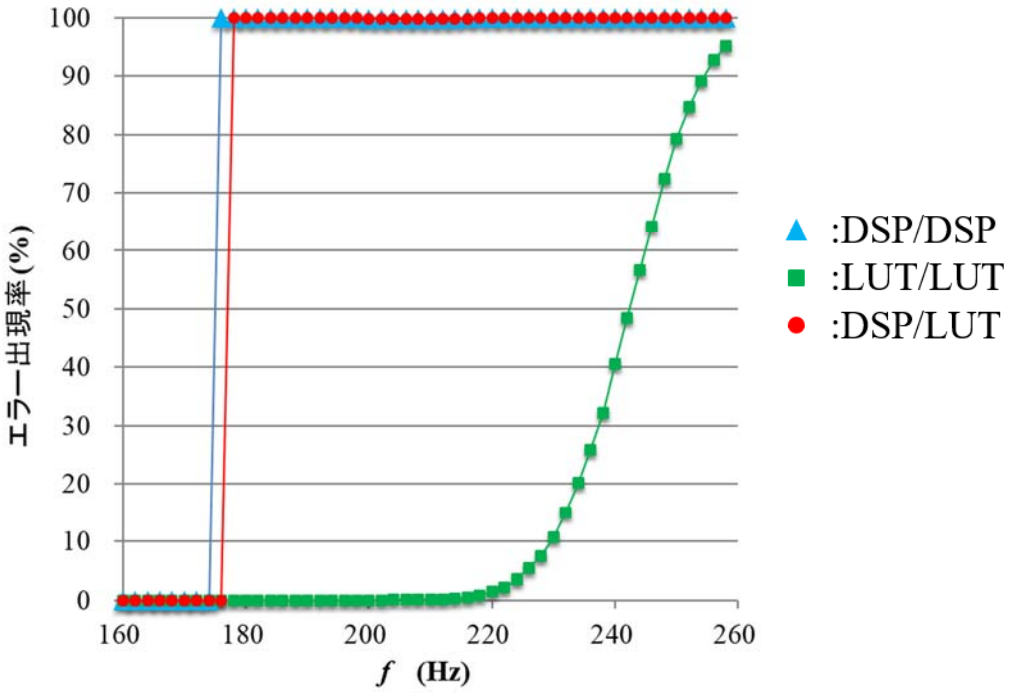


図 3-7. エラー出現率（テストケース1）

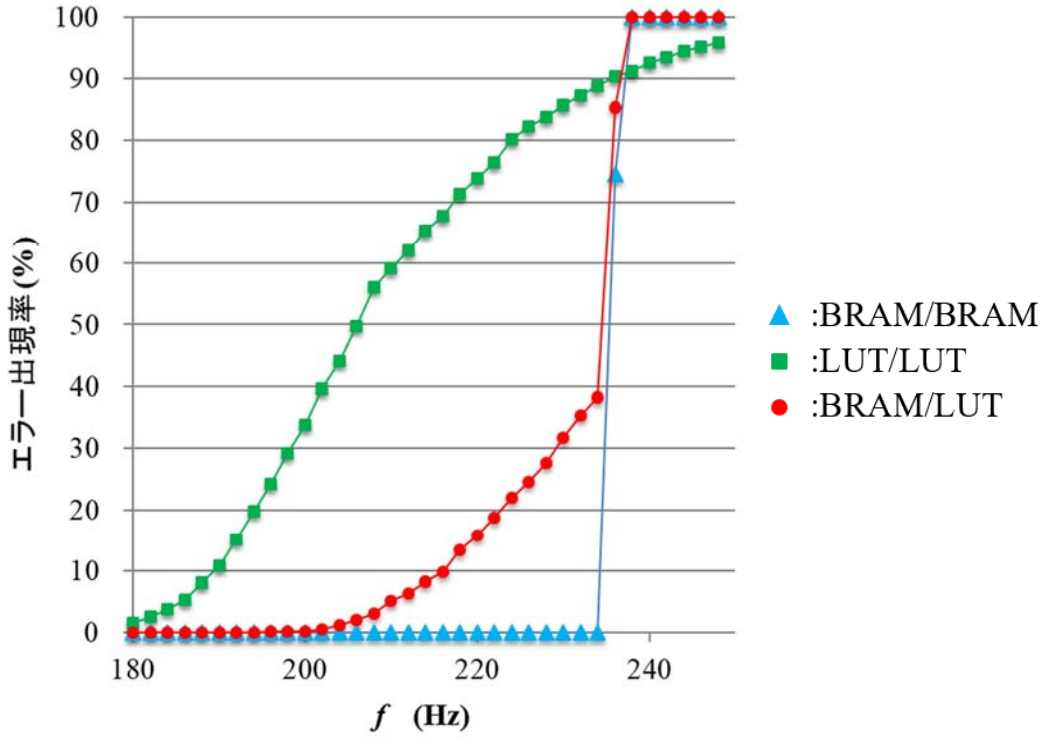


図 3-8. エラー出現率（テストケース2）

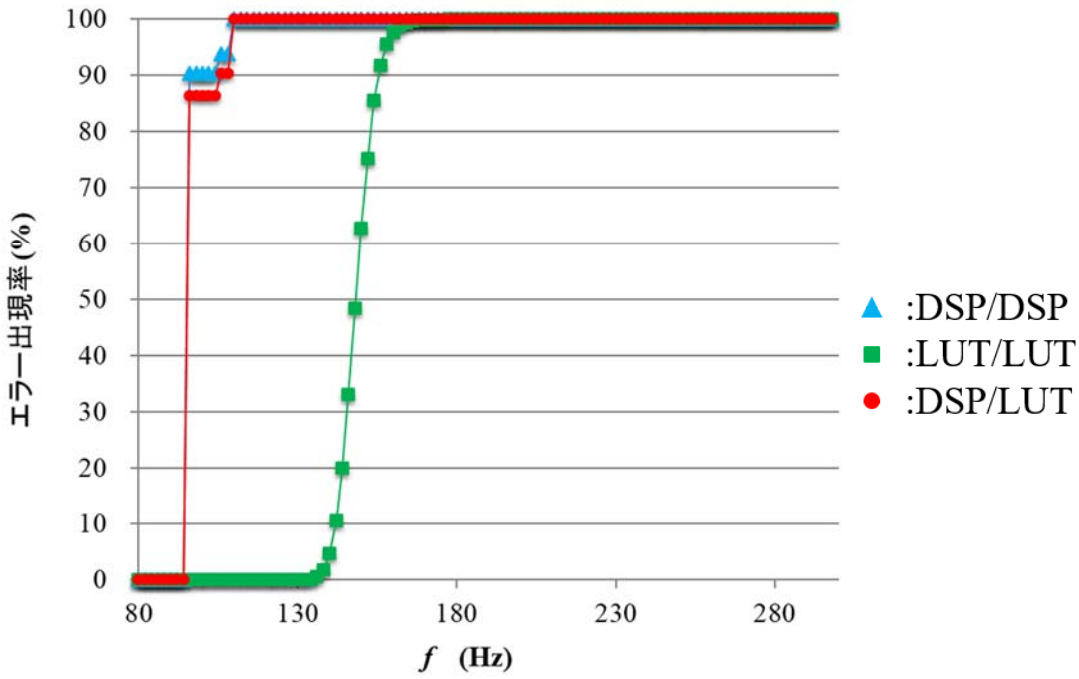


図 3-9. エラー出現率（テストケース3）

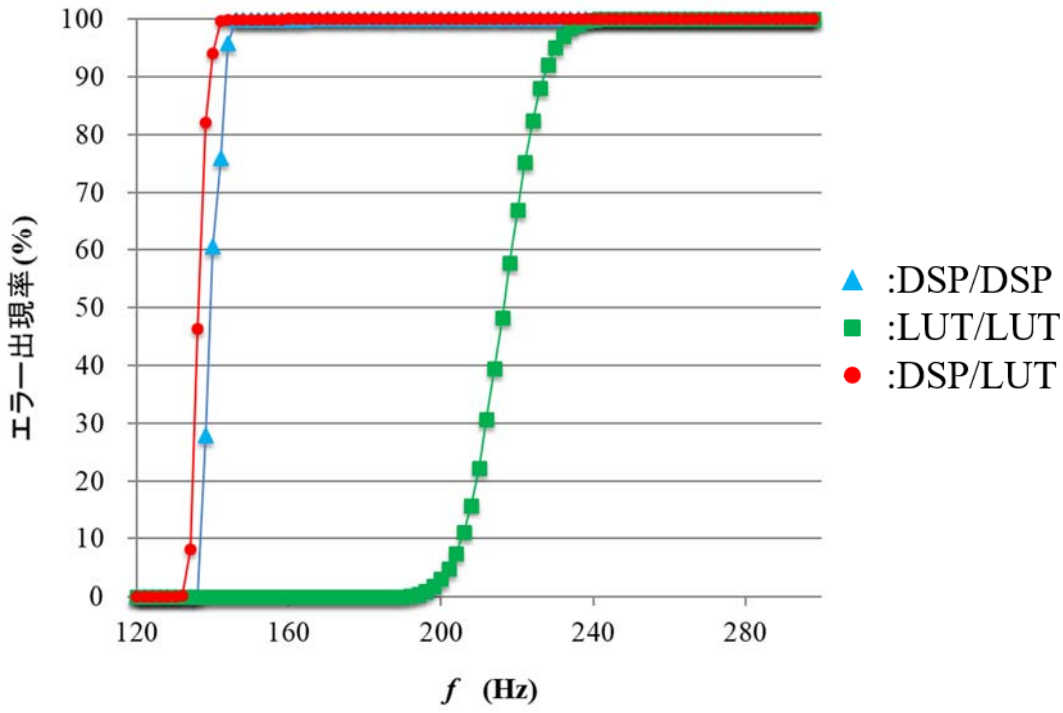


図 3-10. エラー出現率(テストケース4)

次に、図 3-11～3-14 は、各テストナンバーにおいて、b)の観点すなわちエラーの検知率(%)でまとめたグラフである。エラーが発生したとしても、ペアの両方に同じ計算結果のエラーが出現した場合、エラーが発生したことを検知することができない。すなわち、「他方のみエラーが発生」と「両方にエラーが発生したがエラー結果がペアで異なる」場合に限りエラーが検知できたと考え、周波数ごとにエラー出現回数と検知できた回数の比をエラー検知率(%)とした。

DSP だけのペアおよび BRAM だけのペアは、ほとんどすべてのテストケースにおいて、エラーの検知率が0である。これは、エラー出現時の両者の計算結果が完全に一致するからである。片方だけがエラーというケースもない。これは冗長設計が機能していないことを意味している。LUT だけのペアではエラー検知率が向上している。これはデータの遅延の影響が入力信号の値に依存しているからである。DTM を用いたペアは更によいエラー検知率を示している。テクノロジマッピングに多様性が生み出されたため、計算結果が異なるからである。

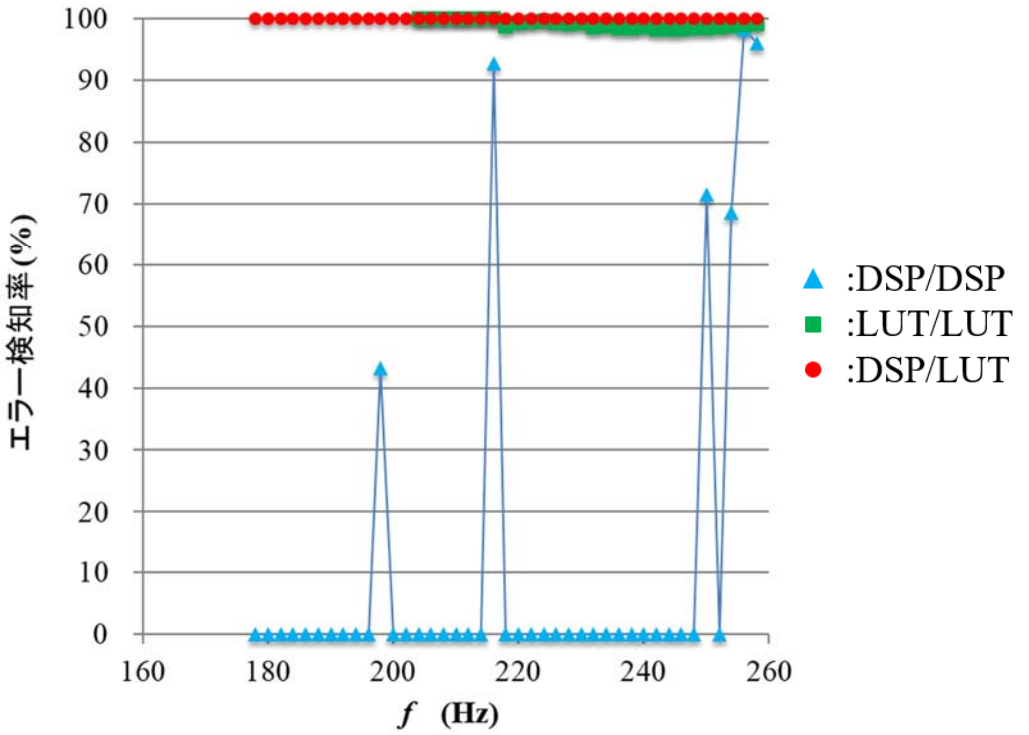


図 3-11. エラー検知率（テストケース1）

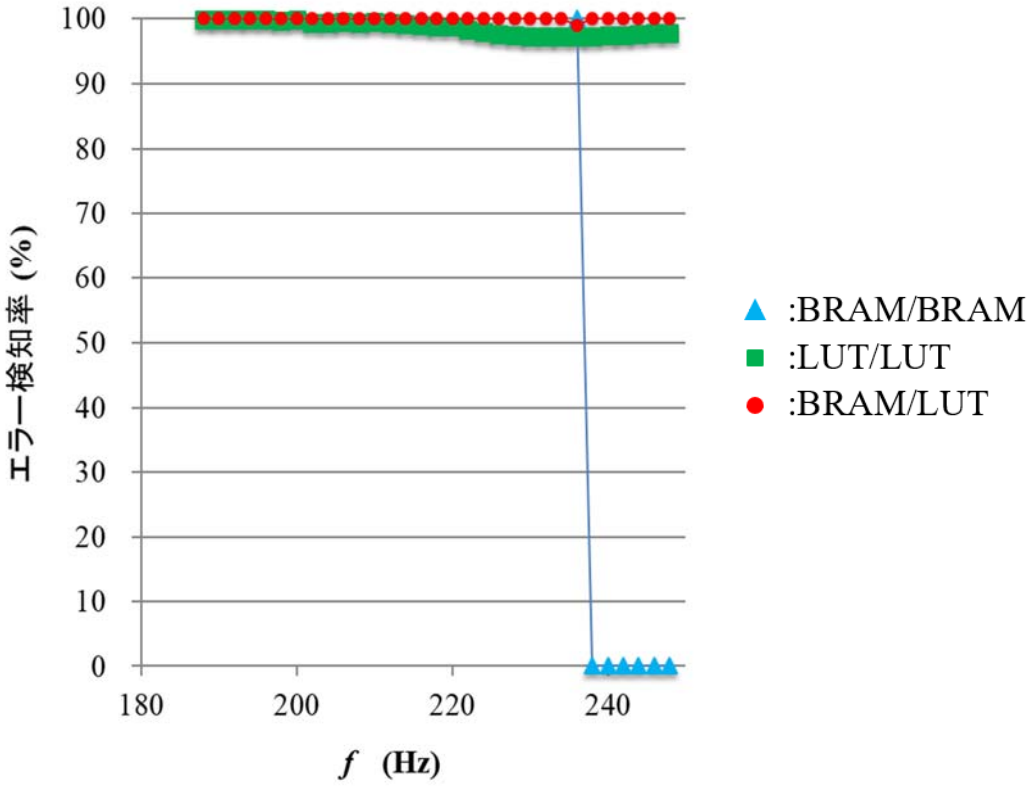


図 3-12. エラー検知率（テストケース2）

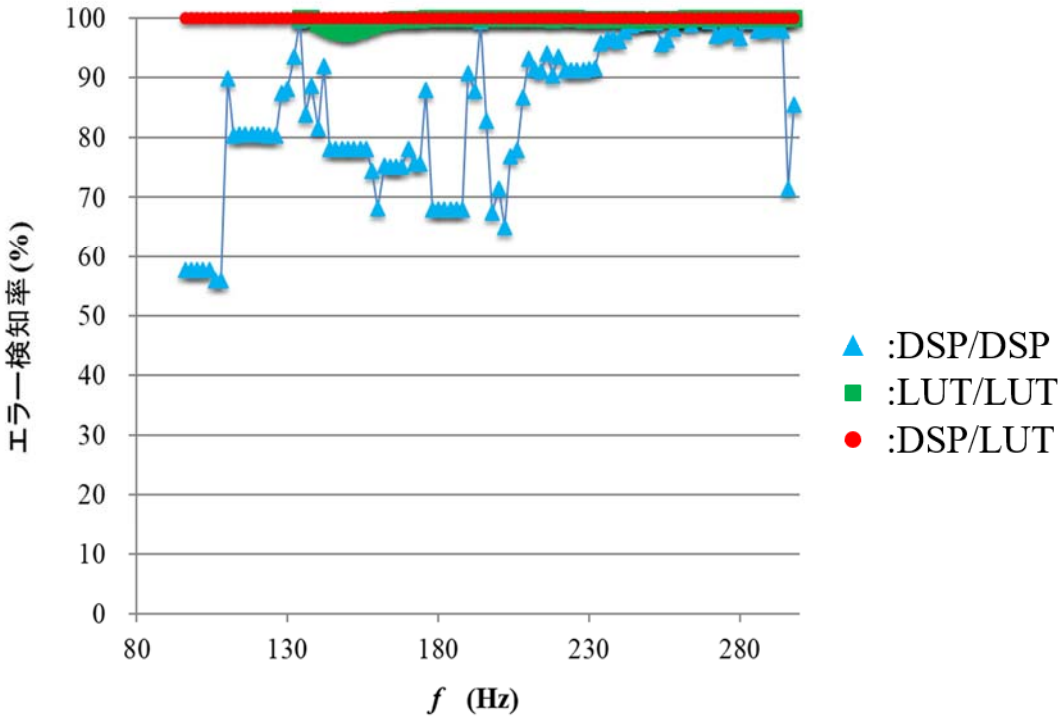


図 3-13．エラー検知率（テストケース3）

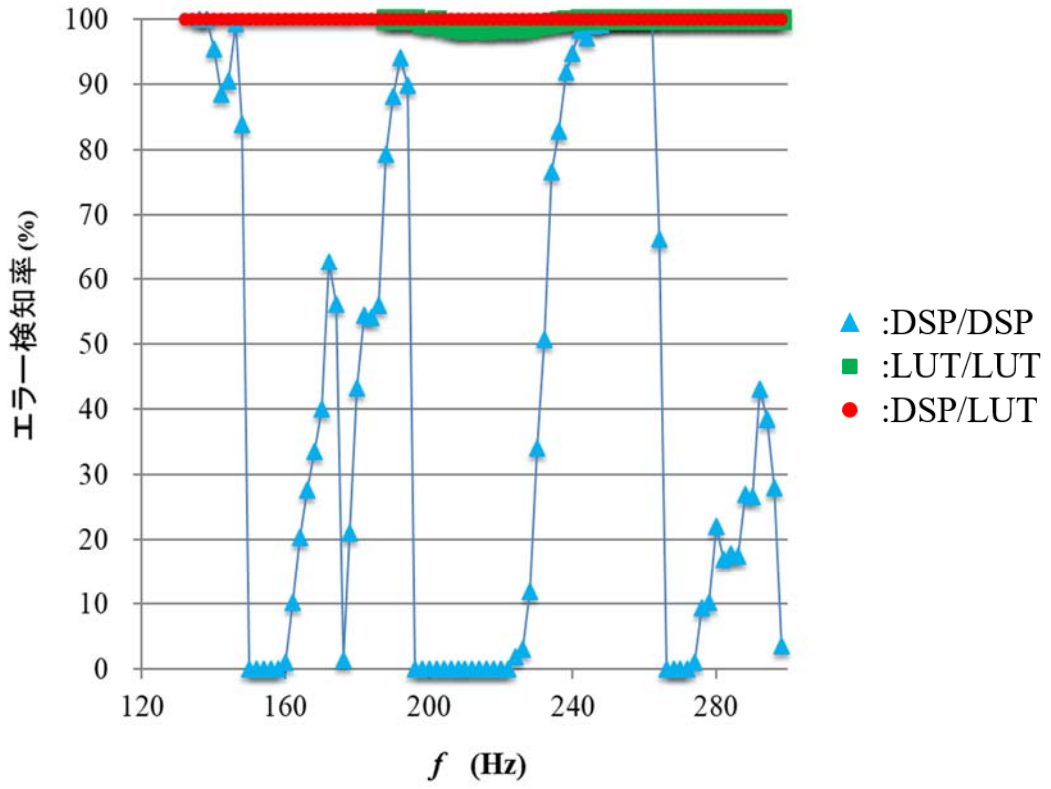


図 3-14．エラー検知率（テストケース4）

次に、図 3-15～3-18 は、各テストナンバーにおいて、c)の観点すなわち検知確率でまとめたグラフである。ペアで結果が異なることでエラーの発生を検知できたとしても、計算結果の違いが大きく異なる場合とあまり違わない場合では検知性能が異なるといえる。計算結果が大きく異なるすなわち、計算結果をビット列で表した場合、より多くのビットに違いがあれば、そのペアの検知確率がより高いと考えることができる。

エラー検知確率$P_{(f)}$を次のように定義する。

$$P_{(f)} = \frac{\sum_{i=1}^{10^4} E(i,f)}{32 \times N(f)} \qquad (3.4)$$

fはクロック周波数である。$E_{(i,f)}$は周波数fにおけるi番目のテストのペア間の不一致ビットのビット数である。$N_{(f)}$は周波数fにおけるエラーの発生回数である。32 ビットのデータ長で正規化している。

$P_{(f)}$= 0 とは 32 ビットすべての結果が一致していることを意味し、$P_{(f)}$= 1 とは両者の値が完全に異なっていることを意味する。$P_{(f)}$は、周波数fにおいて、エラー出現時に検知できる能力を示すため、$P_{(f)}$= 0 とは発生したエラーを検知できないことを示す。

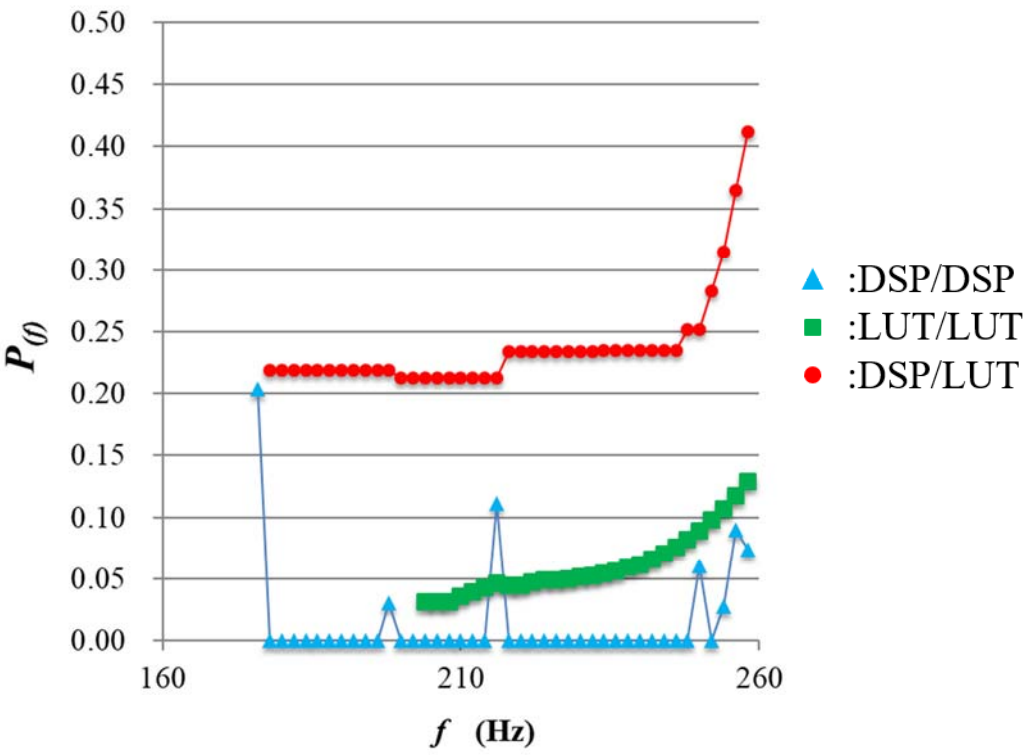


図 3-15. エラー検知確率（テストケース1）

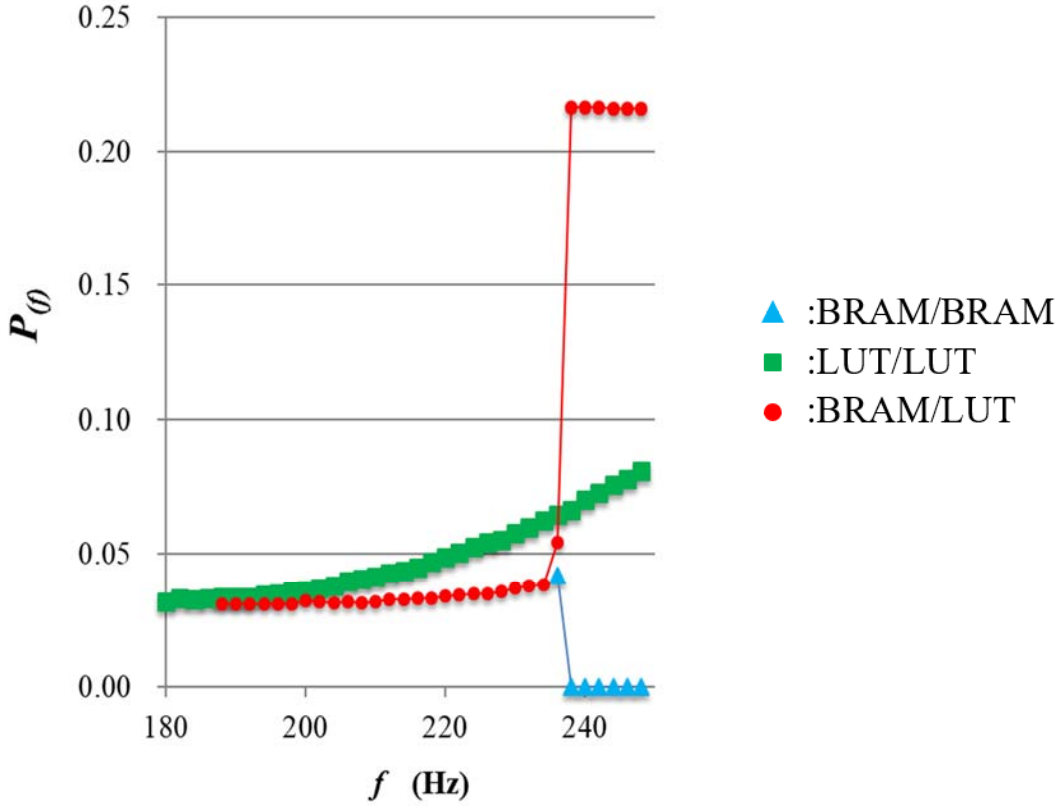


図 3-16. エラー検知確率（テストケース２）

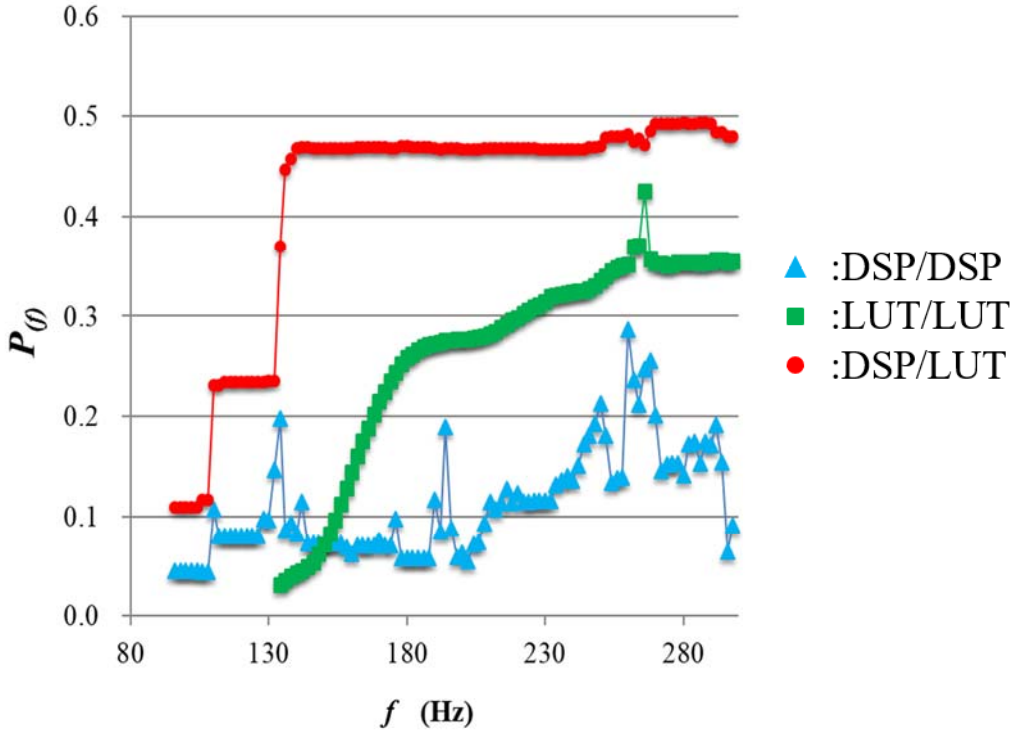


図 3-17. エラー検知確率（テストケース3）

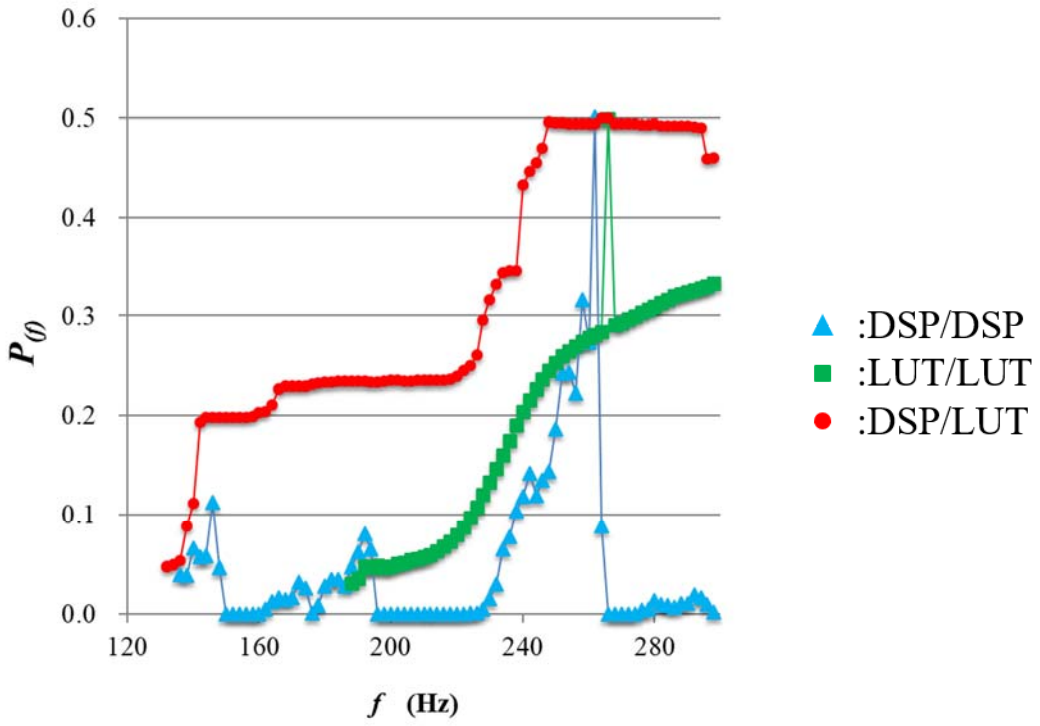


図 3-18. エラー検知確率(テストケース4)

3.4.4 エラー検知確率とβファクタおよび MTBF との関係

関数単位の多様化の効果が制御の演算全体の多様性に対してどのように影響するのか考察する。制御演算は関数の連結で示すことができる[32),(101)]。代表的な例として FBD がある。ここでは、FBD で実現する制御システムを用いて DTM を評価する。図 3-19 は FBD と DTM の関係を示している。またこの図は機能ブロックのエラー検知確率とシステム全体としてのエラー検知確率との関係も示している。

n 個の関数の連結で構成する制御演算があり、その k 番目の関数のエラー検知確率を P_kとする。エラーとは、冗長化した制御システムのすくなくともどちらかにエラーがあることを両者の間の不一致によって確認できる確率を意味する。P_kは、$P_{(f)}$の平均的な確率であると考えることができる。よって、$(1-P_k)$は、どちらにもエラーがあるにもかかわらず、結果が一致するためエラーを検知できないケースの確率である。$\prod_1^n(1-P_k)$は、システムが何らかのエラーを含んでいる時に冗長なペアの間の機能ブロックのすべての計算結果が、全体のプロセスにわたり不一致を全然持っていない確率を意味している。この確率は 3.2.2 節にて示した、共通要因故障割合 βと等しいと考えることができる。

$$\beta = \prod_1^n(1-P_k) \tag{3.5}$$

P_kは機能ブロックの種類によって異なる。ここでは、P_kの分散は小さいと仮定し、$\bar{P}$を次のように定義する。

$$\bar{P} = \frac{1}{n}\sum_{k=1}^{n} P_k. \tag{3.6}$$

するとβは次のように考えることができる。

$$\beta \approx (1-\bar{P})^n \tag{3.7}$$

原理的には、βには比較回路が発生するエラーの確率も含まれている。このシステムはn個の比較回路を持つ。すべての比較回路が同時にエラーを発生させる確率は、無視できるほど小さいと考えることができる。

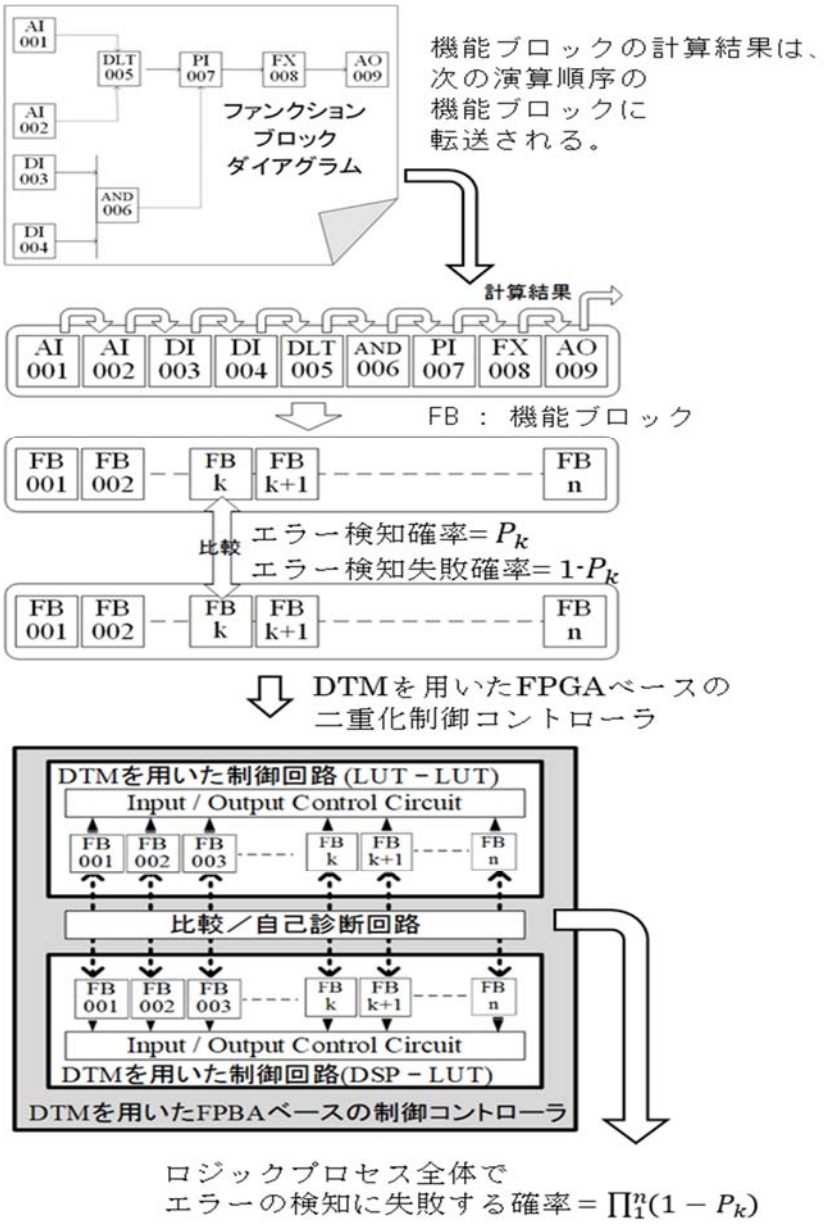


図 3-19. FBD ベースの制御コントローラと DTM

図 3-20 は、β とnに対する多様性の効果を示している。比較データは、表 3.2 におけるテストケース1：関数は「掛け算」、ペアは「DSP/LUT（赤）」と「LUT/LUT（緑）」である。オーバークロック周波数が 210MHz から 258MHz の値をサンプルとし、そのサンプルのエラー検知確率$P_{(f)}$の平均を$\bar{P}$とする。LUT と LUT のペアと DSP と LUT のペアのエラー検知確率$\bar{P}$の差は小さいため、n=1 や n=5 のように、n が小さい時には βは大きくは違わない。n=10 や n=20 のように、n が大きくなると DSP と LUT の β は大きく減少するため、LUT と LUT のペアと DSP と LUT のペアの β の差は大きくなる。n＝20 では差は 100 倍以上となる

図 3-21 は、β とサブシステムの故障率λ_{sub_sys}によって、MTBF にどのような影響があるのかを示している。MTBF は 3.2.3 節の式（3.2）より、

$$\mathrm{MTBF} = \frac{1}{\beta \times \lambda_{sub\_sys}} \qquad (3.2)$$

また、MTBF は年を単位としている。使用したテストケースは前述と同じである。 サブシステムの故障率は、10^{-4}/hr、10^{-5}/hr、および 10^{-6}/hr を適用した。サブシステムの故障率にかかわらず、共通要因故障割合 β が小さくなることにより、MTBF は 10 倍～100 倍改善していることが確認できる。

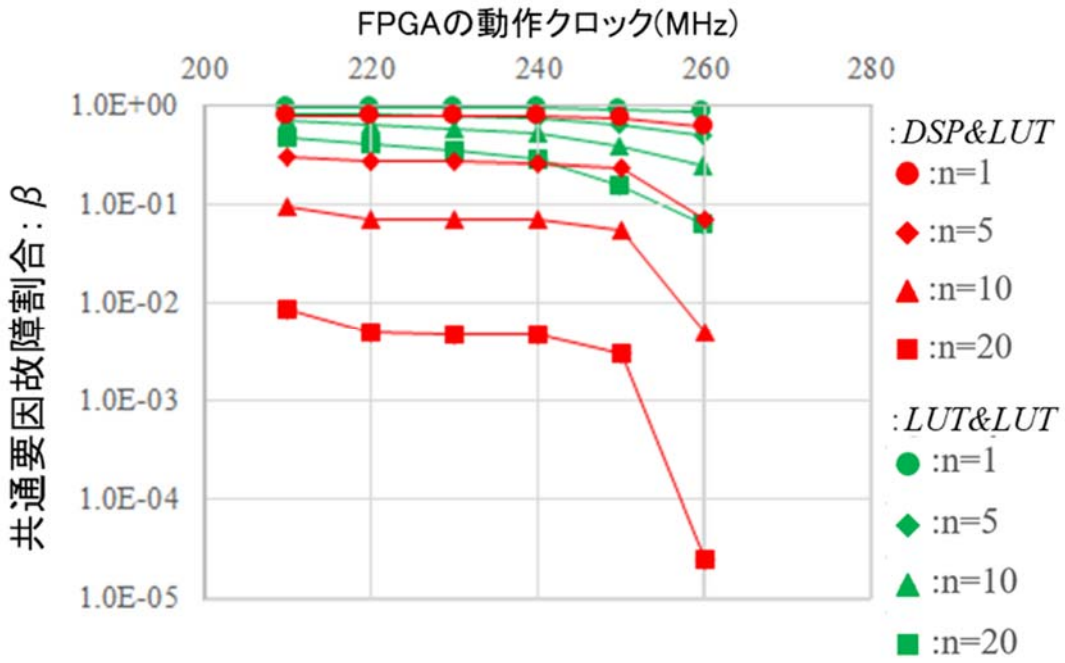


図 3-20. β とクロック周波数および n と多様性の効果

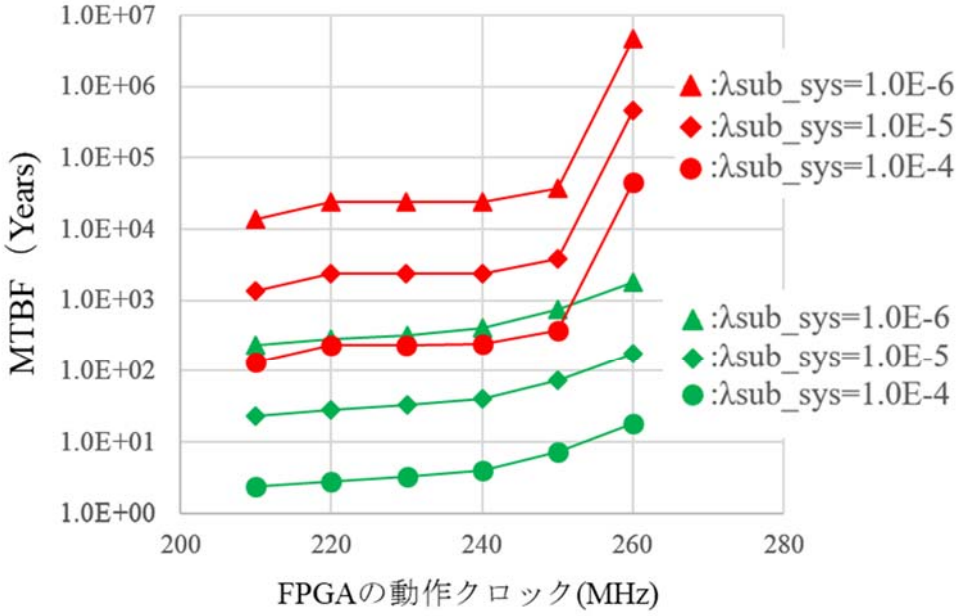


図 3-21. MTBF とクロックおよびサブシステムの故障率と多様性の効果

3.5 むすび

本章では、アルゴリズムやコードおよびハードウェアが共通であっても、冗長化した FPGA 回路に多様性を生み出す新しいアプローチを提案した。これを DTM と名付けた。DTM は意図的に FPGA の資源配分を変化させることにより、多様性を生み出す手法である。その手法によって生成した FPGA 回路をシミュレーションによって評価した。その結果、多様化した FPGA 回路のペアは、ハードウェアへの環境ストレスによってエラーが発生するときに、異なる計算結果を発生しやすくなり、エラーを効率的に検知することが明らかになった。FBD で示した制御回路の機能ブロック数の連結数が大きくなるほど多様化の効果は大きくなり共通要因故障割合 β の値は小さくなる。一般的な制御回路である連結数 20 では、多様化したペアと多様化をしていないペアの β は 100 倍以上となる。これにより、システムの MTBF は多様化することにより、10 倍から 100 倍改善することが判った。DTM が共通要因故障や MTBF に効果がある手法であることが明らかとなった。

また DTM は十分に使用実績のあるツールソフトウェアを使用するため、産業用設備の制御システムに適している。特に高い信頼性をもち、メンテナンスフリー、低コスト、低発熱であるため、再生可能エネルギーに最適である。また DTM はシステム内部の潜在故障を低減する効果も期待できると考えられる。本件については次章で明らかにする。

付録 3-1 共通要因故障を考慮しない二重化システムの MTBF の導出

図 3-3 のマルコフモデルにおいて、$P_0(t)$は両サブシステムが正常な状態に存在する確率の時間関数である。$P_1(t)$ は一方のサブシステムが異常状態である確率である。$P_2(t)$は両サブシステムが異常状態である確率である。$P_0(t)$ から $P_1(t)$ に、サブシステム 1 台あたりの故障率λ_{sub_sys}によって推移する。トータルの故障率は$2 \times \lambda_{sub_sys}$である。このシステムは修復可能システムのため、$P_1(t)$から$P_0(t)$ に、修復率μ_{sub_sys}で推移する。発電設備などでは、バックアップの予備システムへの切り替えを行うため、μ_{sub_sys}は一般的に1である。$P_0(t)$, $P_1(t)$ および$P_2(t)$ は、λ_{sub_sys} と μ_{sub_sys} を用いて次のように表すことができる。

$$\frac{\mathrm{d}}{\mathrm{dt}}P_0(t) = -2\lambda_{sub\_sys}P_0(t) + \mu_{sub\_sys}P_1(t) \quad \text{(A-3.1)}$$

$$\frac{\mathrm{d}}{\mathrm{dt}}P_1(t) = 2\lambda_{sub\_sys}P_0(t) - \mu_{sub\_sys}P_1(t) - \lambda_{sub\_sys}P_1(t) \quad \text{(A-3.2)}$$

$$\frac{\mathrm{d}}{\mathrm{dt}}P_2(t) = \lambda_{sub\_sys}P_1(t) \quad \text{(A-3.3)}$$

稼働開始時、両サブシステムは正常であるため、初期値は下記の通りとなる。

$$P_0(0) = 1,\ P_1(0) = P_2(0) = 0 \quad \text{(A-3.4)}$$

MTBF とはシステムが稼働できる状態の存在確率$R(t)$ の時間積分である。

$R(t)$ は$P_0(t)$ または $P_1(t)$ どちらかの状態であるため、MTBF は次のように定義できる。

$$\mathrm{MTBF} = \int_0^\infty R(t)\,dt = \int_0^\infty (P_0(t) + P_1(t))\,dt \quad \text{(A-3.5)}$$

これを（A-3.1）から（A-3.4)を用いて解く。

$$\mathrm{MTBF} = \frac{\mu_{sub\_system} + 3\lambda_{sub\_sys}}{2\times\lambda_{sub\_sys}{}^2} \quad \text{(A-3.6)}$$

前述の通り μ_{sub_system}＝1>>λ_{sub_sys}

$$\mathrm{MTBF} = \frac{1}{2\times\lambda_{sub\_sys}{}^2} \quad \text{(A-3.7)}$$

付録 3-2 共通要因故障を考慮した二重化システムの MTBF の導出

図 3-4 のマルコフモデルにおいて、$P_0(t)$は両サブシステムが正常な状態に存在する確率の時間関数である。$P_1(t)$ は一方のサブシステムが異常状態である確率である。$P_2(t)$は両サブシステムが異常状態である確率である。サブシステムの故障率λ_{sub_sys}をサブシステム間の共通要因故障とそうではない故障に分類する。

$$\lambda_{sub\_sys} = \ (1-\beta) \times \lambda_{sub\_sys} + \beta \times \lambda_{sub\_sys} \quad \text{(A-3.8)}$$

$\beta \times \lambda_{sub\_sys}$ は共通要因故障を示し、システムを$P_1(t)$ から$P_2(t)$ に直接遷移させる。$P_0(t)$ から $P_1(t)$ に、サブシステム 1 台あたりの故障率 $(1-\beta) \times \lambda_{sub\_sys}$ によって推移する。トータルの故障率は $2 \times (1-\beta) \times \lambda_{sub_sys}$である。このシステムは修復可能システムのため、$P_1(t)$から$P_0(t)$ に、修復率μ_{sub_sys}で推移する。発電設備などではバックアップの予備システムへの切り替えを行うため、μ_{sub_sys}は一般的に1である。$P_0(t)$, $P_1(t)$ および$P_2(t)$ は、λ_{sub_sys} と μ_{sub_sys} を用いて次のように表すことができる。

$$\frac{\mathrm{d}}{\mathrm{dt}}P_0(t) = -\left(2 \times (1-\beta)\lambda_{sub\_sys} + \beta \times \lambda_{sub\_sys}\right)P_0(t) + \mu_{sub\_sys}P_1(t) \quad \text{(A-3.9)}$$

$$\frac{\mathrm{d}}{\mathrm{dt}}P_1(t) = 2 \times (1-\beta)\lambda_{sub\_sys}P_0(t) - \mu_{sub\_sys}P_1(t) - (1-\beta)\lambda_{sub\_sys}P_1(t) \quad \text{(A-3.10)}$$

$$\frac{\mathrm{d}}{\mathrm{dt}}P_2(t) = \beta \times \lambda_{sub\_sys}P_0(t) + (1-\beta)\lambda_{sub\_sys}P_1(t) \quad \text{(A-3.11)}$$

$$P_0(0) = 1, \quad P_1(0) = P_2(0) = 0 \quad \text{(A-3.12)}$$

これを用いて、MTBF を計算すると次のようになる。

$$\mathrm{MTBF} = \int_0^\infty R(t)\,dt = \int_0^\infty (P_0(t) + P_1(t))\,dt \quad \text{(A-3.13)}$$

$$\mathrm{MTBF} = \frac{\mu_{sub\_sys} + 3(1-\beta)\lambda_{sub\_sys}}{2(1-\beta)^2{\lambda_{sub\_sys}}^2 + \mu_{sub\_sys}\beta\lambda_{sub\_sys} + \beta{\lambda_{sub\_sys}}^2} \quad \text{(A-3.14)}$$

前述の通り $\mu_{sub\_system} = 1 >> \lambda_{sub\_sys}$

$$\mathrm{MTBF} = \frac{\mu + 3(1-\beta)\lambda}{2(1-\beta)^2\lambda^2 + \mu\beta\lambda + \beta\lambda^2} \approx \frac{1}{\beta \times \lambda_{sub\_sys}} \quad \text{(A-3.15)}$$

第 4 章 FPGA による産業用コントローラの実現と評価

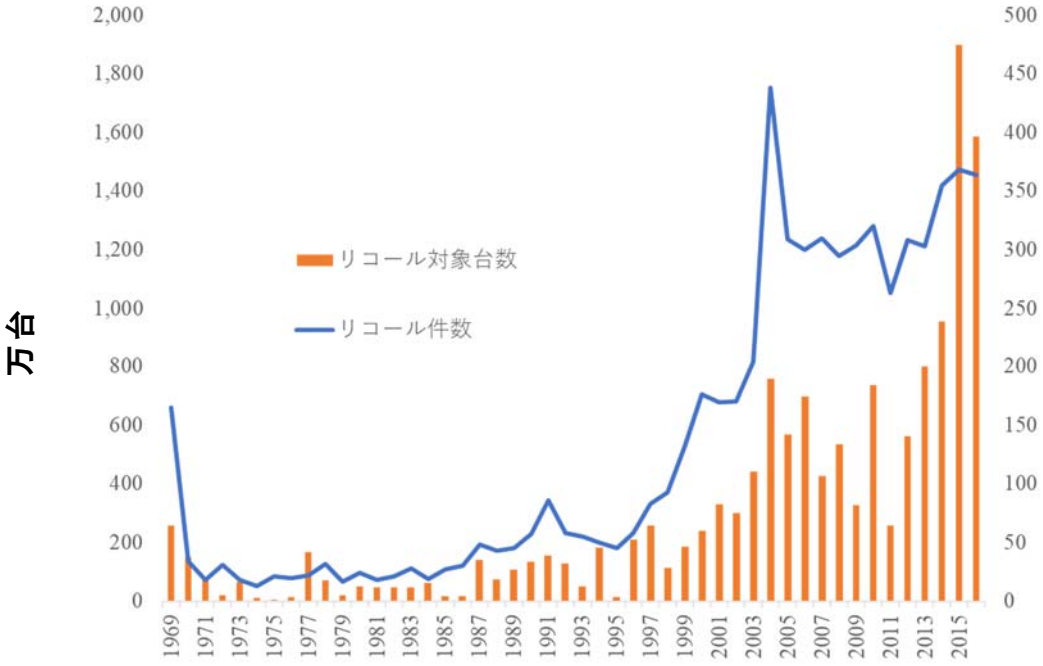
4.1 まえがき

トヨタ自動車の米国における急加速[85]、サムソンの携帯電話の爆発[103]など、製造過程の品質管理が原因ではなく、製品の構造や内部の仕組みに起因する事故や、そのような事故に対する製造物責任(Product Liability)に対する裁判やリコールが増加している。図 4-1 および図 4-2 は、日本での自動車のリコールと家電製品のリコールの件数の推移である。これらの件数の増加は、自動車では車体や部品の車種を跨いだ共通化のためにリコール範囲が広くなることや製造物責任に対する一般の認知度が上がったことなども起因している。事業者としては更に慎重な製品開発と製造が重要である。

自動車の自動運転やドローンによる輸送、介護や医療の現場でのロボットの導入など、今後はさらに複雑で危険性の高い装置が身近に増加する時代が予想されている[104]。制御システムや保護システムによる安全装置の信頼性が強く求められる。

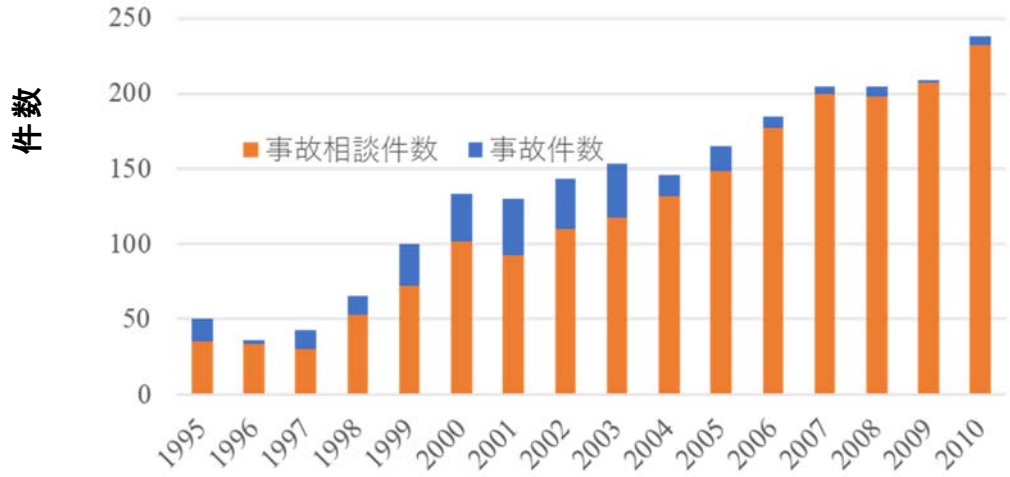
従来の制御と保護の関係は国際規格 ISO 12100[26]にまとめられている。本質的安全設計による保護機構を設置することでシステム全体の安全性の達成を図る。表 4-1 は、ISO 12100 にある本質安全による安全達成ための設計要件である。たとえば産業用ロボットや工場内の搬送機械では、図 4-3 のように機械の周囲に柵を設け、人の立ち入りを物理的に制限する仕組みが本質的安全対策に相当する[23]。自動車では車両統合システムとは独立した機械式のハンドブレーキを設置することで危険な場面で確実に動作する保護機構を用意している。これも本質的安全対策に相当する[20]。

本質安全的なアプローチで安全や保護を達成するのではなく、近年では保護の仕組みをコンピュータが担うケースが増えてきた。たとえば図 4-4 のように、人と共同作業をするロボットでは安全柵による安全距離をとることはできず、ロボットに搭載された力センサーにより、人との干渉を検知して自律的に保護動作を行う[25]。自動運転を行う自動車では、レーダや画像処理装置により車両周囲の障害物を検知し緊急停止するシステムが不可欠である[102]。



出典：H28 年度国土交通省 自動車局 審査・リコール課発表資料[108]
http://www.mlit.go.jp/jidosha/carinf/rcl/data_sub/data004.html

図 4-1. 国産自動車のリコール件数の推移



出典：H28 年度 一般財団法人家電製品協会家電製品 PL センター年次報告書[109]
http://www.aeha.or.jp/plc/houkoku/nenji/nenji_2016.pdf

図 4-2. 家電製品のリコール件数の推移

表 4-1.　ISO 12100 本質的安全における設計アプローチ

ISO12100 6.2 Inherently safe design measure	
6.2.2.1	幾何学的要因
6.2.2.2	物理的側面の考慮
6.2.3	機械の一般的技術知識
6.2.4	適切な技術の選択
6.2.5	機械作用原理の利用
6.2.6	安定性
6.2.7	保守性
6.2.8	人間工学的考察
6.2.9	電気的な危険源
6.2.10	空液圧の危険源
6.2.11	制御装置の本質的安全設計
6.2.12	安全機能の故障確率最小化
6.2.13	機器信頼性による危険源への暴露機会の制限
6.2.14	搬入搬出作業での危険源への暴露機会の制限
6.2.15	作業区画を危険源外とすることによる暴露機会の制限

図 4-3. 安全柵による防護

図 4-4. ロボットと人の共同作業

本質的安全対策は重力や電磁気力による保護の仕組みに基づくため、構造の原理評価に加え、その保護機構の製造品質が安全性を評価する重要な観点である。コンピュータを用いたしくみはコンピュータ自身の製造品質に加え、ソフトウェアのバグなど人が原因となる障害の可能性が高くなる。これを系統故障（システマティック故障）と称する[15]。また大気中の荷電粒子によるメモリなどの半導体素子へのソフトエラーやノイズの影響による誤動作も重大な故障を誘発する原因となる[82]。集積回路の中で発生するこのようなエラーが、プログラムのどのような機能に影響を及ぼすのかを確定的に分析することは極めて難しい。コンピュータの誤動作による安全性の喪失について評価が必要である。たとえば欧州への輸出にはその評価がなければ上市できない。このためコンピュータを使用した保護の仕組みを評価する基準が国際的に取り決められている。IEC 61508（機能安全）である[15]。

機能安全では、システムの安全性を SIL という指標で設定し、ランダム故障の観点では危険側故障確率を算出し、システマティック故障の観点では厳格な設計プロセスを要求し第三者認証を経ることを求めている。故障確率の評価においては高い SIL レベルでは多重化が不可欠である。機能安全評価における多重化では、共通要因故障、潜在故障（検知できない危険側故障）の存在による影響度を評価する必要がある。共通要因故障については本論文第3章に述べた通りである。多様化が重要である。

システムに潜在故障が多く存在することは、多様化・多重化の効果を低くする恐れがあり小さくする必要がある。潜在故障を小さくするためには、自己診断の仕組みをシステムにあらかじめ組み込む必要がある。すなわち、システムのアーキテクチャに大きく依存する。一

般的に自己診断は CPU に大きな負荷をかける。たとえば表 4-2 は機能安全の国際規格に挙げられている自己診断機能の例である[15],[30],[31]。

一方、FPGA では並列化した回路によって制御演算と自己診断を同時に実行する構造にできるため、限られた演算資源を効率よく産業用コントローラに搭載することができる。本章では、第 2 章および第 3 章を踏まえ、高い安全完全性を備え、かつインフラなどの産業用途の制御システムに使用できる FPGA 二重化コントローラのアーキテクチャを提案し、その安全完全性について国際規格 IEC 6158[15]に基づき評価を行う。

次節では、二重化したシステムに影響を及ぼす潜在故障について述べ、その影響を小さくするための対策としての自己診断が CPU を用いた産業用制御システムに大きな問題になっていることを示し、FPGA による実現性の優位性について述べる。第 3 節は、前述の課題を解決する FPGA に基づく産業用コントローラのアーキテクチャを提案し、産業用途に適している点を確認する。第 4 節は、提案したシステムを IEC 61508 に基づき評価し、SIL レベルの評価を行う。第 5 節はまとめである。

表 4-2. IEC 61508 における CPU の自己診断機能の抜粋

参照先	項目	備考
	不揮発メモリ	
	複数ビットによる冗長化	例)データの CRC 保護
IEC 61508-7 A.4.4	データエリアの署名	
IEC 61508-7 A.4.5	データエリアの複製	例)多重保存、読み出し時の比較
	可変メモリ	
IEC 61508-7 A.5.1	RAM テスト	例)ウォークビットまたはガルパット法
IEC 61508-7 A.5.6	訂正コードによる検知	
IEC 61508-7 A.5.7	冗長メモリによるリードライトテスト	例)冗長化した RAM の内容を比較する
	割り込みなどの I/O	
IEC 61508-7 A.6.1	テストパターン	特定の周期で特定信号を強制的に入力(出力)させて健全性を確認する。
IEC 61508-7 A.6.3	マルチチャンネルパラレルアウトプットによる監視	例)同一出力に一方をリードバックして出力状態を確認する。
IEC 61508-7 A.6.5	多数決	
IEC 61508-7 A.11.4	相反信号	反転した信号を同時入力(出力)する。
	プログラムシーケンス処理部	
IEC 61508-7 A.9.1	ウォッチドッグタイマ	処理時間の規定超過の外部観測
IEC 61508-7 A.9.2	ウィンドウ付きウォッチドッグタイマ	超過と遅延の両方の外部観測
IEC 61508-7 A.9.3	プログラム順序の論理的な観測	例)処理終了毎に特の CRC を計算する
IEC 61508-7 A.9.4	プログラム順序の時間的な観測	例)処理終了毎に処理時間を評価する
	クロック	
IEC 61508-7 A.9.5	多重クロックによる周波数監視	

4.2 潜在故障とCPUの課題およびFPGAの有用性

多重化したシステムでは、両方に共通する原因による共通要因故障と潜在故障が多重化の効果を低下させる要因となる。共通要因故障については第 3 章で示した。ここでは潜在故障について述べる。

故障は図 4-5 に示す通り4つに分類することができる。発生した故障によって、そのシステムが危険な状態になる原因となるものを危険側故障と称する。発生の結果システムが安全な状態になるものを安全側故障と称する。たとえば火災を消火するための装置(スプリンクラー)において、火災ではないときに水が噴射する故障を発生した場合、結果的に火災には至らないため安全側故障と考える。一方止水弁が腐食などの要因で固着し、火災が発生しても水が噴射できない場合、その故障は危険側故障と考える。これらの故障は発生と同時に故障であることを運用者が知ることができる故障と、故障の発生を知ることができず必要な動作をする際に動作できないことによって、はじめて故障していることを知ることができる故障がある。前者を検知できる故障、後者を検知できない故障と称する。

危険側で、かつ検知できない故障をここでは潜在故障と称することとする。図 4-5 では λ_{DU} と表している。潜在故障が二重化したシステムに及ぼす影響をマルコフモデルで示したものが図 4-6である。簡略化のため、サブシステムの故障すべてが潜在故障である場合を示す。

潜在故障は発生してもシステムまたは運用者が認知できないため、発生したサブシステムを修理する機会が失われる。すなわち修復率は0である。時間の経過とともに、他方のサブシステムにも危険側故障が発生することにより、二重化したシステムの両方が異常となり運用ができない状態になる。このような事象を潜在故障の累積と称する。潜在故障はシステムの多重化の効果を著しく低下させることが分かる。このため多重化したシステムでは、潜在故障を低減するため自己診断(Diagnostic または Self-test)の機能を搭載する。自己診断の周期がその保護システムに求められる応答時間よりも短ければ、自己診断の結果検知できた潜在故障は、検知可能な故障として扱うことができる。

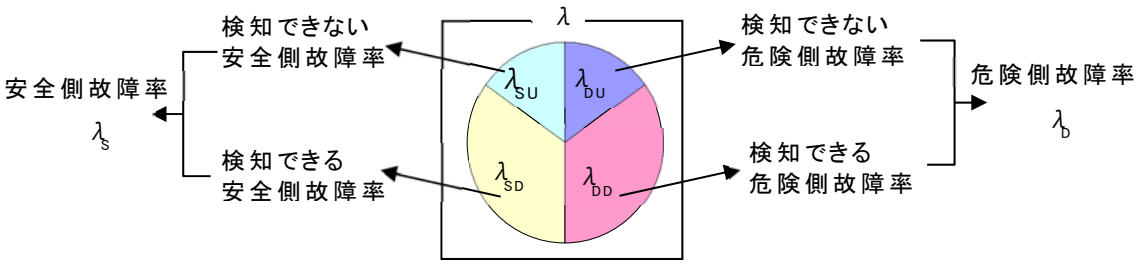


図 4-5. 故障の分類

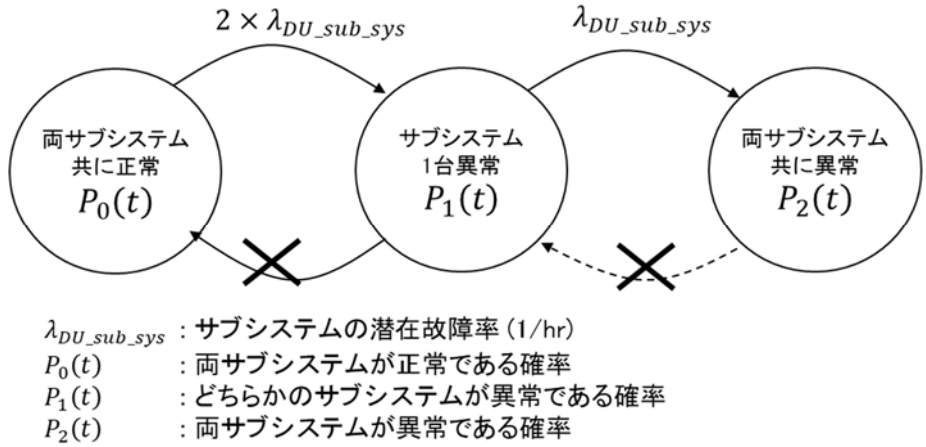


図 4-6. 潜在故障の影響を示す二重化システムのマルコフモデル

CPU を用いる演算装置の自己診断機能は表 4-2 のように一般化されている[15]。潜在故障を自己診断によって検知できる故障としてみなすためには、自己診断を実施する周期が、そのシステムが担う安全保護機能の安全時間以内に実施することが必要である。安全時間とは、安全や保護のシステムが異常状態を計測したのち安全動作が完了するまでに許容することができる時間である。言い換えれば、異常状態が実際に災害につながる状態(ハザード)となるまでに想定される時間である。

たとえば、ガスタービンなどの大型の発電用原動機では、燃焼不安定などの異常状態が、タービン翼が溶けるなどの災害につながると想定できる状態に至るまでの時間は 1 秒未満である。このため、異常状態をセンサーが観測した時点から燃料弁を閉止する電磁弁を励磁するまでに 0.1 秒未満であることが求められる。機械構造物では、弁やアームな

どを動作させる動力が起動し動作を完了させるまで、物理的な遅延時間がさけられない。上記の例であれば、およそ 0.4 秒は弁が動作するための時間に割り当てられる。この場合、異常を判断し弁を閉止する演算結果を出力する産業用コントローラには 0.1 秒未満の動作時間しか割り当てることができない。潜在故障を検知できる故障と見なすためには、その制御システムの自己診断を 0.1 秒周期の安全時間以内で常時働かせておく必要がある。

これは CPU の演算処理の時間的資源を大きく消費する。たとえば、搭載する可変メモリを表 4-2 に基づいてビットテストした場合、DRAM の使用エリアの内容を一旦退避させ変化させたのち、固着をチェックし元の記録内容を復元する。復元時にも CRC などの符号をつけることによってデータのビット化けを確認する。プログラムシーケンスのチェックではライブラリなどの処理単位ごとに特定の CRC 演算を行い、最終の処理完了後、予定の処理フローを経たかどうか CRC の演算結果を用いて判定する。CPU が担うべき自己診断機能は一般化されているが、極めて処理量が多い。さらに CPU が演算に使用するメモリなどの資源が大型することで、自己診断に必要な CPU 能力は増大している。加えて自動車の自動運転など安全保護装置に求められる安全時間も短くなっているため、自己診断を完了させなければならない時間要求が短くなってきている。

このため CPU が本来の制御や保護のための演算に必要な処理時間に加え、自己診断にかかる処理時間も大きな割合を占めるため、さらに高速で大容量な CPU が必要になってきている。産業用制御システムには発熱などの制約も多い。CPU に負担のかかる潜在故障の検知処理を目的とした自己診断機能は今後さらに大きな問題となる。

産業用制御システムの演算フローは図 4-7 のようになる。制御演算は厳格な一定間隔の演算周期を実現する必要があるため、自己診断処理は演算周期ごとに分割して実施する。このような処理の切り替えをタスクスイッチ、コンテキストスイッチと呼ぶ。コンテキストスイッチを行う際 CPU は、処理中のレジスタの値を保存しスタックに退避し、次の処理が退避していたレジスタを読み込み、中断していた処理を継続する。CPU にとってコンテキストスイッチは、それ自身が CPU の処理時間を消費する作業であり、数多くのコンテキストスイッチは望ましくない

FPGA は素子の中の演算回路が並行処理する点が特徴である。多くの自己診断機能が必要であっても、制御演算と並行して行うことができる。たとえば大容量化するメモリのチ

ェックも短時間で完了する。CPU では1ブロックごとに検査する処理が必要であるが、FPGA ではブロック RAM ごとに同時的にメモリチェックを実施できる。また自己診断機能を並列処理することができれば、図 4-8 のように、システムが保証できる安全時間を短くすることができることも大きなメリットである。潜在故障が検知できる故障とみなせる安全時間を短くすることができることは、より高速な安全保護を必要とする移動体の自動運転や、人と共同で動作する産業用ロボットなどで有用性が高い。

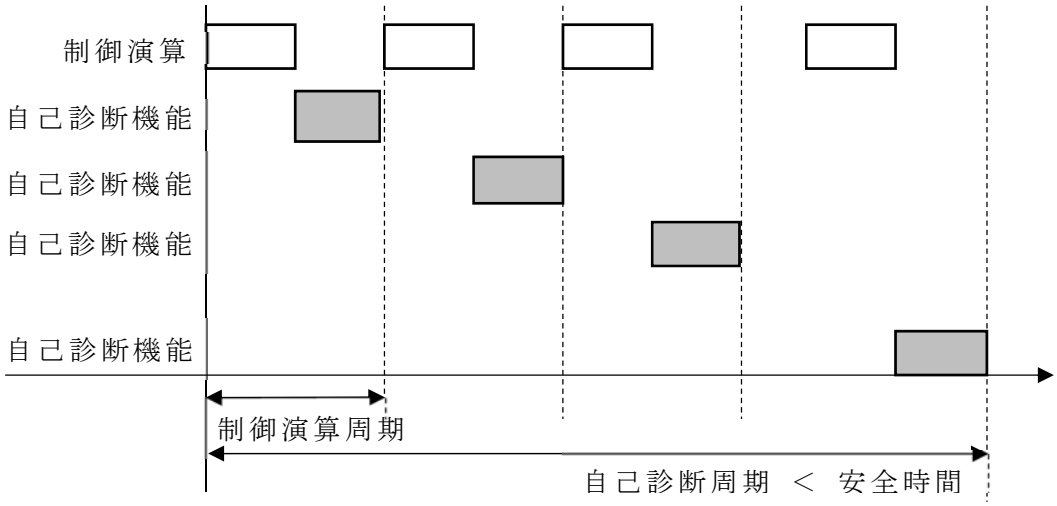


図 4-7. CPU における制御演算と自己診断機能

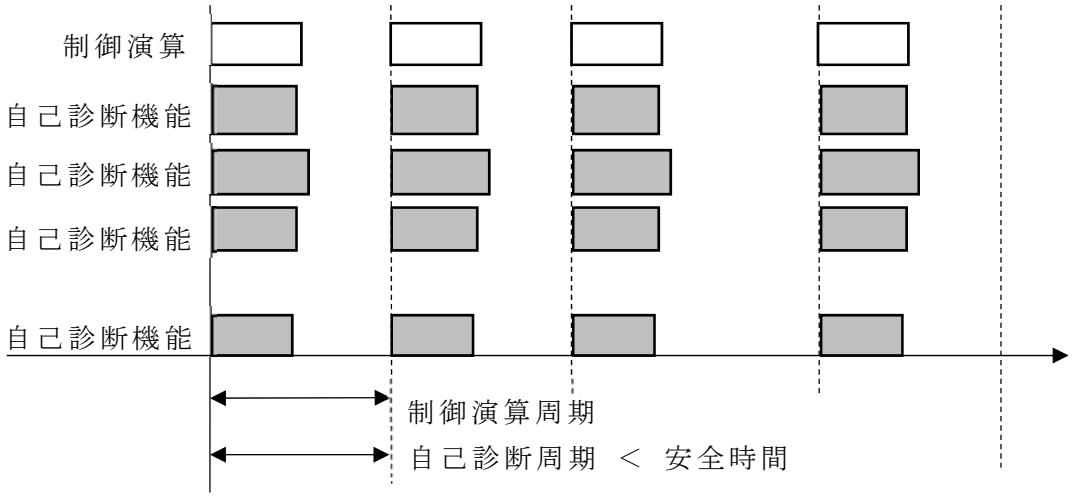


図 4-8. FPGA における制御演算と自己診断機能

4.3 Safety Green Controller (SGC)アーキテクチャの提案

FPGA の並列処理によるメリットの一方、産業用制御システムとしては課題がある。第 2 章で示した通り、産業用制御システムとして活用するためには、モデリング言語からの制御アルゴリズムの生成に加え、設備現場でのオンラインでの変更が必要である。FPGA の内部で構成する回路に応じて必要となる自己診断回路も変化するため、オンラインでの変更によって必要となる自己診断回路も変化することが予想される。このため、モデリング言語を用いることができ、かつ制御アルゴリズムの変更で FPGA の配線回路の再構成が必要とならない自己診断の仕組みが必要である。この解決アプローチとして TDB を用いる。また機能安全への対応のため、DTM アプローチを用いることが有効である。本章では、第 2 章 TDB および第 3 章 DTM の成果を踏まえ、FPGA による自己診断付きの産業用コントローラのアーキテクチャを提案しその安全性を国際規格に基づき評価する。

図 4-9 のように第 2 章で提案評価した TDB アーキテクチャの演算処理部を拡張し、各機能ブロックの演算結果をバス経由で次の機能ブロックに伝達する際に、自己診断用演算結果メモリにも計算結果を転送する。TDB アーキテクチャではバスの転送はマルチキャスト型通信を用いるため、自己診断演算結果用メモリへの転送による演算処理の遅延は発生しない。このように、個々の機能ブロックの結果を自己診断用演算結果メモリに一時保管する。次に図 4-10 のように、演算処理回路を 2 系統用意し互いに演算結果を比較する構造を構成する。この比較回路が相互監視機能を実現し、システムの自己診断機能としての役割を果たす。

機能ブロックの演算結果の比較の結果、結果が不一致であれば、どちらかのサブシステムにエラーが発生したことを検知することができる。この検知によって、設備全体をフェールセーフ状態(たとえば退避処理など)に移行し、保護システムが無効化することを防ぐ。

このように機能ブロックごとに比較するエラー検知方法は、前章の結果より、演算の最終結果だけを比較する手法よりも共通要因故障の影響や潜在故障の影響を小さくできる。比較回路そのものの故障も共通要因故障の要因であるが、本方式では連結する機能ブロック数の数だけの比較回路が存在する。システム全体として比較回路が共通要因故障となるためには、多くの比較回路が同時に故障し無効となる必要がある。なぜならば、ある機能ブロックの計算結果はその次の機能ブロックの計算結果に影響を与えるため、後段

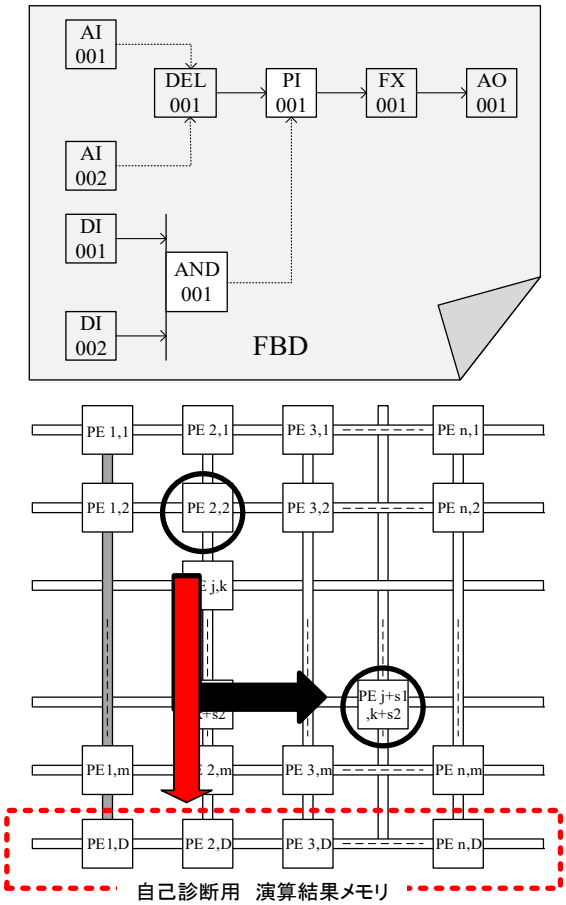


図 4-9. TDB アーキテクチャの拡張

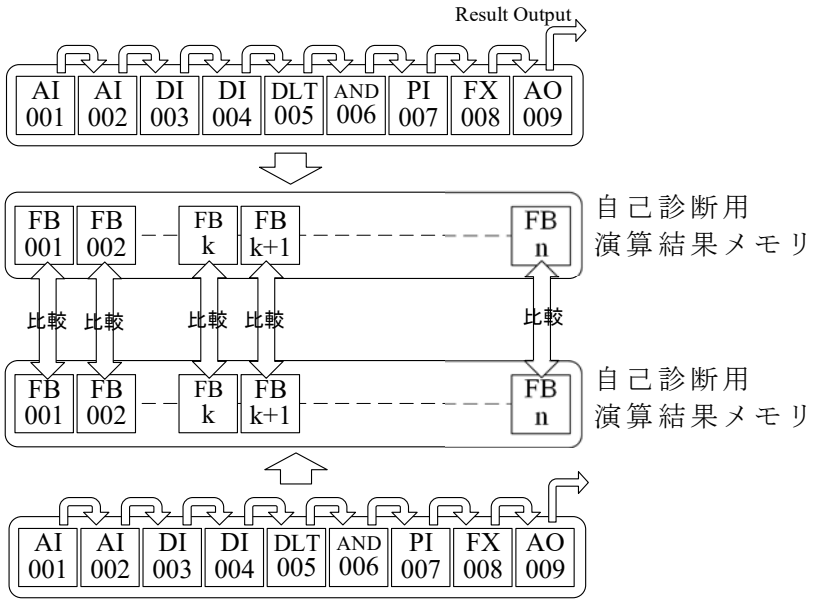


図 4-10. 二系統 TDB の相互監視による自己診

の機能ブロックの計算結果も不一致になる可能性が高い。比較回路の故障ですべて検知できないためには、後段の比較回路もすべて故障し無効化することが必要だからである。このような確率は極めて小さいと考えられるため、比較回路が共通要因故障として与える影響は無視することができると考える。

第 2 章の結果から、MATLAB や Simulink、FBD で記載された制御ロジックは、TDB アーキテクチャによって、連結された機能ブロックの順序演算で実現することができる[101]。制御演算を機能ブロックの連結で表せることにはメリットがある。この特性から各機能ブロックの演算ごとに結果を比較することができる。比較回路は機能ブロックを実行する回路とは別な回路が並行して行うため、演算処理の遅延は発生しない。

二系統の FPGA 回路は同一タイミングで同じ制御演算アルゴリズムを演算するため、共通要因故障が発生すると二系統の機能ブロックは同じエラーを発生させるため、異常を検知することができない。そこで第 3 章の結果から、2 系統の FPGA 回路は同じ制御アルゴリズムであっても、DTM アプローチを適用することにより FPGA の配線回路に多様化を持たせ共通要因故障を低減することが可能である[105]。このように第 2 章の TDB アーキテクチャ、第 3 章の DTM アプローチを用いた制御演算の構造を図 4-11 に示す。このアプローチを用いて、1 ループ制御を実現すると図 4-12 のようになる。機能ブロックごとの比較回路はシステムの自己診断回路として機能する。最終的な演算結果に出現しない回路の中に隠れたエラーを検知することを目的とする。

この FPGA による産業用コントローラの新しいアーキテクチャを SGC（Safety Green Controller）と呼称する。SGC は TDB の特徴と DTM の特徴を共に備える。モデル言語による制御演算設計を可能にするほか、第 2 章の結果よりオンラインでの変更や FPGA の資源利用効率の点で、大型の火力発電所でも使用できる処理能力を持つ。産業用コントローラとして有用である。また第 3 章の結果より、多様性を生み出す DTM アプローチは既存の FPGA のコンパイルソフトウェアなどの使用実績のある”Proven-in-use”ツールを用いることができる点も大きな利点である。さらに、制御演算の規模や複雑さに応じ FPGA の大きさを自由に選ぶことができる。処理が増加した場合、より大きな FPGA に変更する際も、構造を継承することができる。このことも産業用コントローラとして有用である。次節にて、機能安全におけるコントローラとしての信頼性評価を行う。

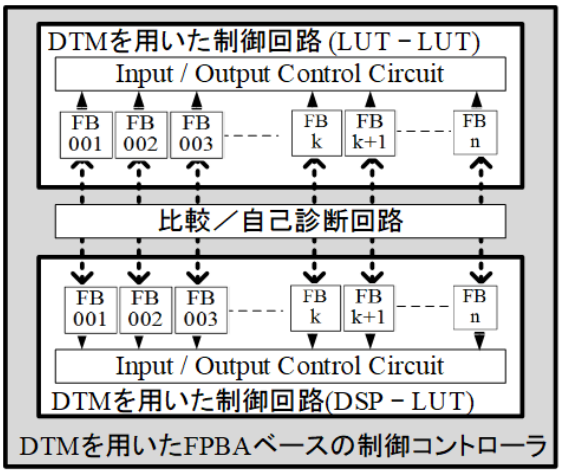


図 4-11. DTM アプローチによる二系統演算の多様化

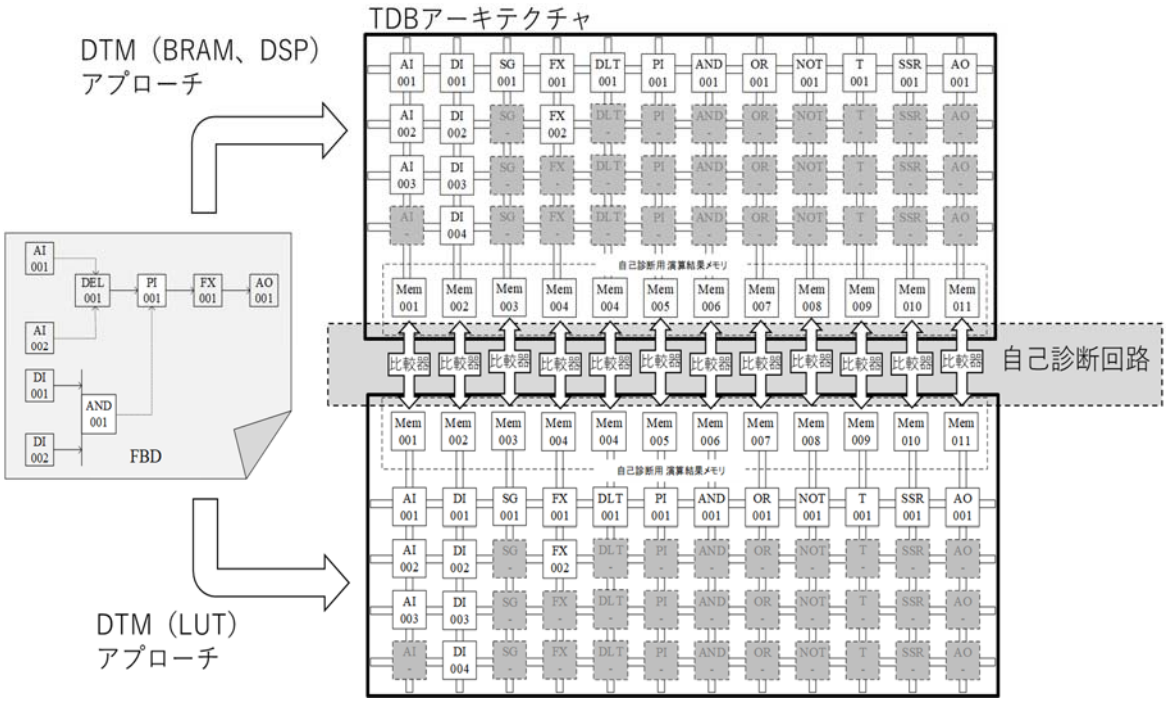


図 4-12 FPGA による自己診断付き二重化コントローラと 1 ループ制御

4.4 SGC の機能安全における評価

SGC を制御システムまたは保護システムとして用いた場合の、信頼性について評価する。DTM アプローチによる多様化による共通要因故障への効果は第 3 章で示した通りである。ここでは、国際規格に基づき、SGC の機能安全における安全完全性レベル（SIL）を評価する[15),(28)]。ロボットや自動車の自動運転のような連続稼働を行う制御システムにおける安

全完全性の評価にはPFH(Probability of Failure per Hour (h^{-1}))を用いる。PFHは、システムに発生するランダムハードウェア故障のうち、システムを安全ではない状態に導く故障(危険側故障)が発生する単位時間当たりの確率である。安全完全性レベルとPFHは表4-3のように定義されている[15]。

たとえば人と協調して作業するような産業用ロボットでは、安全保護システム全体でSIL2の達成が必要である。これは産業用ロボットの安全規格で指標として規定されている[23]-[25]。この場合、これは産業用コントローラがSIL2のレベルにあればよいということでない。図4-13に一般的な保護システムとPFHの割り当てとSILの関係を示す。

図4-13に示す通り、$PFH\_s = PFH\_i = PFH\_c = PFH\_o = PFH\_a = 2.0 \times 10^{-7}$(SIL2)と仮定した場合、保護システム全体は PFH＝1.0×10^{-6}(SIL1)となり、システム全体としてはSIL2を達成できない。産業用コントローラにはSIL3レベル以上の信頼性が要求される。従って、SGCを用いた産業用コントローラもSIL3レベルの信頼性が不可欠である。

表4-3. SILとPFHの定義(IEC 61508)

Safety Integrity Level (SIL)	Average frequency of a dangerous failure of the safety function [h^{-1}] (PFH)
4	$\geq 10^{-9}$ to $<10^{-8}$
3	$\geq 10^{8-}$ to $<10^{-7}$
2	$\geq 10^{-7}$ to $<10^{-6}$
1	$\geq 10^{-6}$ to $<10^{-5}$

自己診断機能を持つ二重化したシステムのPFHは次の式で求められる[15],[28]。

$$PFH = (1-\beta)^2\{[\lambda_D{}^2 \times 2 \times DC] \times T_2 / 2 + [\lambda_D{}^2 \times 2 \times (1-DC)] \times T_1 / 2\} + \beta \times \lambda_D \quad (4.1)$$

PFH : 二重化システムの危険側故障の時間あたりの発生確率
λ_D : サブシステムの危険側検知できない故障率
DC : 自己診断カバー率(Diagnostic Coverage)
β : 共通要因故障割合
T_1 : 定期検査時間間隔
T_2 : 自己診断時間間隔

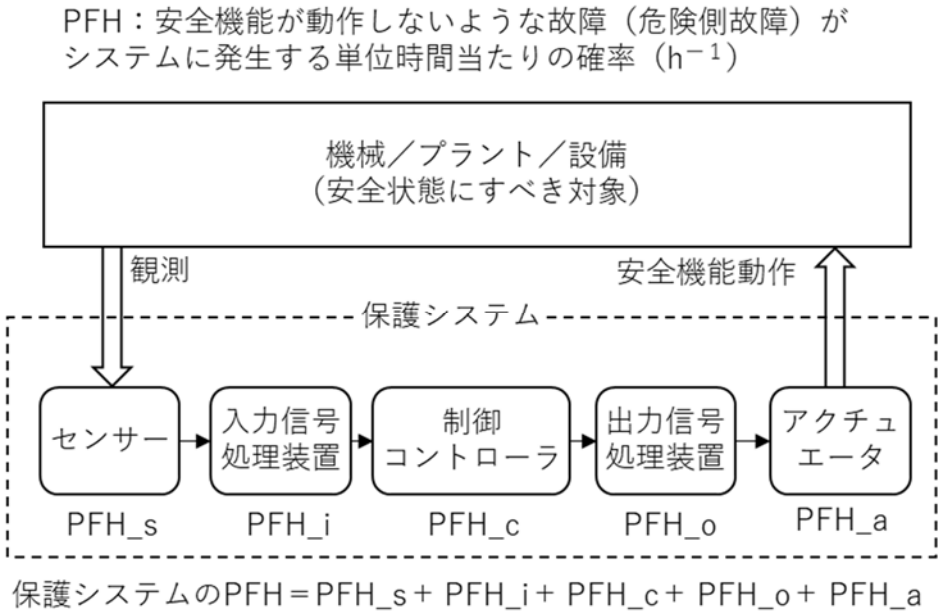


図 4-13. 保護システム全体の PFH 計算と SIL

サブシステムの危険側検知できない故障率は、その部位に含まれる電子部品の危険側検知できない故障率を積算することで求める。電子部品の故障率は、国際規格 IEC 62830[106]や産業分野で公開されているデータベースを用いることが一般的である。ここでは、日本の産業用システムの開発で広く使われている、シーメンス社が提供する電子部品の故障率データベース SN29500[107]を用いる。

このデータベースに基づき、本評価ではサブシステムを構成する FPGA の λ_D を 1,138 FIT と仮定した。大規模な FPGA であり、機械や発電プラント制御を十分に実施できるスケールである。FIT とは、Failure in time と呼び単位時間当たりの故障率である。1 FIT＝1.0×10^{-9} (h^{-1})である。

共通要因故障割合 β は、第3章の 3.4.3 のテストケースのうち、Multiple 関数（テストケース1）における DTM を用いたペア（DSP/LUT）と非 DTM のペア（DSP/DSP）のオーバークロック周波数 240MHz の試験結果を用いた。β は連結する機能ブロックの個数によって変化する。表 4-4 にまとめる。

表 4-4 演算要素 Multiple （DTM 使用:DSP/LUT）の β

	n＝1	n＝5	n＝10	n＝20
DTM 使用（DSP/LUT）	0.7652	0.2623	0.0688	0.0047
DTM 不使用（DSP/DSP）	0.9383	0.7273	0.5290	0.2798

DC（Diagnostic Coverage）とは、潜在故障を検知可能な故障として扱うための自己診断機能のエラー発見効率である。本論文では、第3章の Error Detection Rate（%）と同義である。DC の値も、前述の β と同じく、本論文第3章 3.4.3 テストケース1のデータにおける DTM を用いたペア（DSP/LUT）と非 DTM のペア（DSP/DSP）のオーバークロック周波数 240MHz の試験結果に基づくものとする。DTM を用いたペア（DSP／LUT）は、ほぼ検知率 99%以上であるが、DTM 不使用のペア（DSP/DSP）は検知率がおおむね0%である。これは非 DTM の FPGA 回路ではエラー時の結果が同一であり、エラーの発生が検知できないためである。IEC 61508 に基づく PFH 算出の際の DC は、様々な自己診断機能に対する設計パラメータとして、おおむね Low（60%）、Middle（90%）、High（99%）を設定することとなっている。ここでは、DTM 使用（DSP/LUT）ペアの DC を 99%、DTM 不使用（DSP/DSP）ペアの DC を下限値の 60%として設定することとした。

T1（定期検査時間間隔）とは、内部の故障を発見するような、システムの定期検査のインターバルを指す。例えば火力発電所の蒸気タービンでは、弁の固着を確認するために 3 カ月程度に 1 度タービンに流入する蒸気を緊急遮断するための油圧バルブを実開閉させ固着していないことを確認する。このような定期的な試験によって、特定の故障モードの発生率を一旦リセットするような試験の試験期間が T1 である。コンピュータシステムでは設備に据え付けた電子基板内部の電子デバイスの故障モードをつぶさに分解点検するような運用はあり得ない。このため産業用コントローラの想定運用期間 30 年（8,760 時間×30 年）を T1 のパラメータとして使用することが一般的である。

T2（自己診断時間間隔）とは、自己診断を実施する周期である。SGC においては、演算プロセスの中で機能ブロックの演算ごとに結果を比較し異常を検知する仕組みである。このため産業用コントローラの演算周期と同一と考え、一般的なプラント制御において使用される 100 ms をパラメータとして使用する。以上のパラメータを表 4-5 にまとめる。

表 4-5. 設定パラメーター一覧

DTM 使用(DSP/LUT)			
記号	名称	値	備考
λ_D	危険側検知できない故障率	1,138 FIT	
β	共通要因故障割合	0.7652～0.0047	n=20 (表 4-3)参照
DC	自己診断カバー率	99%	
T1	定期検査時間間隔	2.628×10^{5}	30 年
T2	自己診断時間間隔	2.77778×10^{-5}	100 ms
DTM 不使用(DSP/DSP)			
記号	名称	値	備考
λ_D	危険側検知できない故障率	1,138 FIT	
β	共通要因故障割合	0.7652～0.0047	n=20 (表 4-3)参照
DC	自己診断カバー率	99%	
T1	定期検査時間間隔	2.628×10^{5}	30 年
T2	自己診断時間間隔	2.77778×10^{-5}	100 ms

このパラメータを用い、式(4.1)に基づき、n＝1、5、10、20についてプロットしたものが図 4-14 である。SGC は機能ブロックの連結数 n が多くなるほど、PFH の値が小さくなる傾向があり、DTM を使用する場合に顕著となる。機械制御などモデルベースの制御システムでは、演算ブロックの連結数は少なくとも10を超えるため、SGC は SIL3 相当の安全完全性が達成できると評価することができる。SGC は FPGA の規模が小さなものから大きなものまで容易に適用できるため、人間と共同作業をするような産業用ロボットや、自動運転の自動車、さらには化学プラントや発電プラントなど、様々な分野の高い SIL レベルが必要な機能安全システムに適用することができることを意味している。

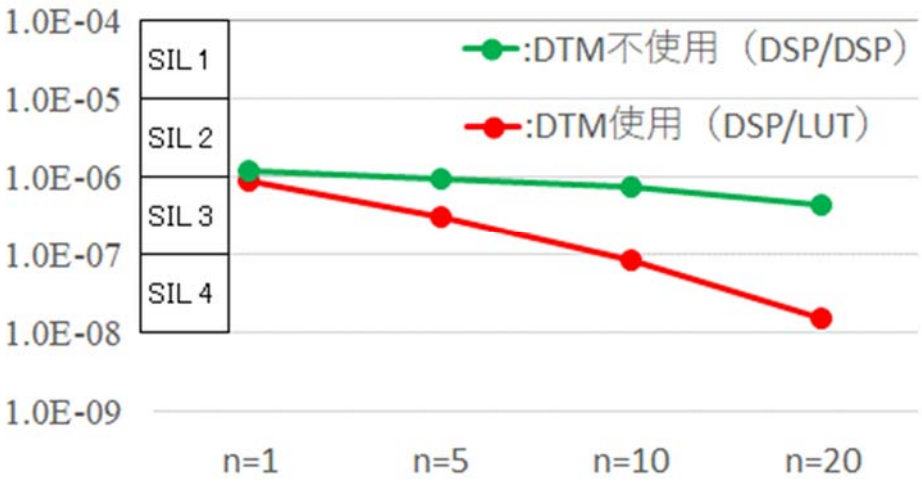


図 4-14. SGC の PFH による評価および DTM の効果

機能安全では、SIL の目標に応じた不動作確率（PFD／PFH）の要件に加え、危険側潜在故障の割合と多重化の効果についても要件が規定されている。図 A-1.6 にある通り、高い SIL を達成するためには多重化の度合い（HFT）が大きいことと、危険側潜在故障率（$\Sigma \lambda_{DU}$）の割合が低いことが求められる。潜在故障すなわち故障が発生しても検知できない故障は、修復する機会を逸するため多重化の効果を低下させるからである。

SGC では機能ブロック毎にエラーを検知することができ、そのエラー検知率（DC）は前述の通り 99%を超える。これにより SFF も 99%以上を見込むことができるため、二重化（HFT=1）で実現した SGC は、図 A-1.6 より SIL4 の要件も満たすことが判る。SGC アーキテクチャは、機能安全規格の要件をすべて満たすことが明らかとなった。

4.5 むすび

本章では、第 2 章および第 3 章を踏まえ、高い安全完全性を備えた FPGA の新しい産業用コントローラのアーキテクチャ SGC を提案した。TDB を備えたサブシステムとすることで、産業用制御システムが必要なモデリング言語を用いた設計や、オンラインでの回路の修正が可能であることを示した。TDB を2つ備えた多重化システムを TDB のバスに演算結果を保持するメモリエリアを設けることで、制御演算の処理を阻害することなく、両サブシステムの機能ブロックの演算結果を比較できる構造が実現できることを示した。この2つのサブステムを DTM の手法を用いて多様化し、制御演算プロセスの中の機能ブロックごとに演算結果を比較する自己診断機能を考案した。比較回路が機能ブロックごとに存在するため、演算プロセスの中に発生する潜在故障を効率よく検知できる。

最後に機能安全の国際規格 IEC 61508 に基づき、多重化システムの PFH を算出した。計算には、第2章および第3章のシミュレーション試験結果を用いた。その結果、FBD で示された制御回路において、一般的な機能ブロック数10を連結する回路の PFH は、1.0×10^{-7} となり、SIL3 相当の安全完全性を持つことが明らかとなった。加えて、危険側潜在故障の割合と多重化の要件も機能安全規格に基づき評価し、機能ブロック数20を連結する回路の場合、SIL4 の制約も満たすことが可能であることを示した。

SCG は、FBD などの記述言語を使用し、産業用コントローラに不可欠なオンラインでの修正機能を持ち、かつ国際規格に基づいた評価でも高い SIL 値を実現することができるアーキテクチャであることが明確となった。

付録 4-1 実際のシステム開発における β および DC の決定

本節では SGC の効果を評価するため、第3章 3.4.3 テストケース1のデータにおける DTM を用いたペア(DSP/LUT)と非 DTM のペア(DSP/DSP)のオーバークロック周波数 240MHz の試験結果を用いた。実際には演算関数や関数の論理合成作業ごとに検知確率が異なる回路が生成されるため、個々の機能ブロックごとに β および DC を設定することが必要となる。

実際の制御システムの開発では、機能ブロックの関数ごとに FPGA の回路を繰り返し生成し、β の値が低くエラー検知効率の高い回路を取捨選択し SCG の基本回路として組み込むこととなろう。その際に選択した個々の機能ブロックごとに β と DC を実験データとともにエビデンスとして残すこととなる。多様化は開発工数だけではなく、製造コストや製品出荷後の保守においても事業者に大きな負担を強いる。開発者側はできるだけ簡便な手法を採用したいが、負担の少ない多様化には効果の点で評価が分かれる手法が多い。このため多様化の手法とその効果は、現実のシステムの開発プロセスにおいて、審査機関と最も議論が活発に行われる領域である。

本論文で提案する SGC および多様化手法の DTM は、前述の通り実際の使用に際しては、関数ごと論理合成作業ごとに実験を行い β と DC の値を規定する必要がある。この点では確定的なアプローチではない。しかし現状の β ファクタや DC の値を決める開発審査において、実験に基づく定量的なパラメータを提示することができるため、機能安全の対応が求められるシステムの開発作業や審査プロセスを大きく効率化できる。このような開発プロセスや審査プロセスの効率化も SGC および DTM の効果である。

第 5 章　結論

産業用コントローラに用いる CPU はコアの高速化が限界に達しつつある。マルチコアによる性能向上が主流であるが、マルチコア CPU は大きな発熱に加え、仮想化などのミドルウェアを必要し、制御システムの基盤を複雑化する傾向がある。またマルチコアのアーキテクチャは製品世代による変化が大きく、ミドルウェアの変更やアプリケーションの変更など、製品の長期供給性や継承性に課題があり、事業者への負担も大きい。この問題は産業用制御コントローラが CPU アーキテクチャに依存する限り深刻化する。

本研究は、CPU に代わる制御演算デバイスとして FPGA に着目し、FPGA が持つ課題を解決し、かつ自動車の自動運転や産業用ロボットなどに不可欠な機能安全における高い安全完全性を実現する、新しい産業用コントローラのアーキテクチャの原理確立と実現を目指した。

近年の制御演算回路の設計は、モデル言語やファンクションブロック図による設計が主流である。このため本研究では、まず MATLAB などのモデル言語から FPGA 回路に容易に展開できる TDB アーキテクチャを提案した。TDB が産業用コントローラとして不可欠な、オンラインでの制御回路やパラメータの変更に対応できること、また FPGA の並列処理構造を活かすことで、火力発電などの大規模で複雑な制御回路であっても、十分な処理能力で実現できることをシミュレーションにより明確にした。

次に FPGA 回路を多重化した際のエラー検知確率を向上させるための、DTM アプローチを提案し評価した。DTM アプローチによって生成された1組の FPGA 回路は、従来方法で生成された FPGA 回路よりも、ペア間に大きな多様化が生じ、環境からのストレスによるランダムハードウェアエラーの発生時の検知効率を向上させうることを確認した。特に DTM アプローチは、モデル言語やファンクションブロック図によって記述された制御回路において、機能ブロックの連結数が多いほど効果があることが確認でき、システムの MTBF 拡大に効果があること明らかにした。

以上2つのアプローチの効果を踏まえ、TDB とDTMを組み合わせた、FPGA ベースの新しい産業用コントローラのアーキテクチャ SGC を提案し、国際規格 IEC 61508 に基づき、産業用コントローラとしての安全完全性レベル(SIL)を評価した。その結果、SGC は大型で故障レートが大きな FPGA を用いた場合においても、MATLAB などのモデリング言語

やファンクションブロック図で表現された連結数 10 以上の制御回路において、SIL3 相当の信頼性を達成できることを証明した。以上のことから、TDB および DTM という2つのアプローチを融合した SGC は、CPU の代替としての FPGA を使用した産業用コントローラのアーキテクチャとして極めて有効であることが示せた。その特徴をまとめると以下のようになる。

（1） FPGA ベースのため、低商品電力低発熱、小型から大型までのスケーラビリティ、製品世代を超えた長期の設計継承性を持つ。

（2） TDB アーキテクチャにより、モデル言語や FBD 言語を用いて制御回路を記述でき、パラメータや制御回路の修正の際に、FPGA の論理合成を必要としない。

（3） TDB アーキテクチャは制御ループごとに並列処理を行うことができ、FPGA の回路資源を有効活用するため、大規模で複雑な制御回路でも、FPGA が大型化せず、かつ十分な処理性能を実現する。

（4） SGC は FPGA の並列処理の特徴を活かした自己診断回路を持つため、自己診断にかかる時間が制御演算周期と同一であり安全時間が大幅に短い。

（5） SGC は DTM アプローチによる高い多様性を持つ二重化した演算回路を持ち、高い自己診断能力により、SIL3 相当の安全完全性を実現する。

上記に加え、SGC は産業用途で十分な使用実績のある開発ツールを使用するアプローチのため、発電や鉄道などシステム全体に信頼性が必要な用途に最適な産業用コントローラであることを証明できた。SGC アーキテクチャを採用することによって、FPGA を CPU の代替とすることができ、産業用コントローラが直面する CPU のマルチコア化の問題を解決できることが明らかとなった。

小さな家電製品から巨大な発電プラントまで、様々な制御システムが使用されている。水素動力などの新しいエネルギーや、介護ロボットのような機械も増えることだろう。そのような将来において、安全や保護を担うシステムの産業用コントローラは重要である。どのような便利な機械やサービスも、長期的な安全・安心の上に成り立つ。本研究で提案し実用を評価した産業用コントローラは、制御技術の進化を支える重要な基盤となるものと考える。

また今後は高速な無線ネットワーク網と人工知能の活用により、自動車やドローンの自動運転・搬送サービスなど、私たちの生活を大きく変えてゆくことだろう。このような未来で

は、産業用コントローラが地球規模のコンピュータネットワークに常時直接接続することが一般的となる。ハッキングやウィルス感染による障害などを鑑みれば、安全や保護を担うシステムのリスクは高まる一方である。CPU が行う機能を FPGA で置き換えることは、本論文で取り上げた発熱や継承性、および機能安全対応の観点のみならず、制御セキュリティの面でも大きな意味を持つと考えられる。

SGC は OS のような多機能で脆弱なミドルウェアを必要とせず、ハードウェアによる機能実現である点、そして並列処理が可能である点から、制御セキュリティ装置の基盤としても注目されている。本研究が安全な制御とセキュリティの礎としても広がることを期待する。

付録 5-1 SGC が役に立つ分野および制御セキュリティへの展開

SGC が活用できる分野について補足する。本論文では主に発電プラントなどのインフラ設備の産業用コントローラを対象に論じた。SGC の効果は、表 A-1.1 のような分野においても効果が期待できる。特に身体補助型のロボットや医療機器においては、逆動力学計算のような負荷の大きな処理を省電力で高速に処理することが必要であるため、SGC の効果が期待できる。機能ブロックを演算する PE を十分に冗長に搭載しておくことで、FPGA の配線接続を変えることなくダメージを受けた PE を変更することができる。宇宙空間や原子力設備のような過酷な環境で使用するフォールトトレラントコントローラにも適している。

表 A-1.1. SGC が有効な分野及びその効果、部位

分野	部位、効果
電力インフラ設備	再生可能エネ、スマートグリッド、保守性向上、メンテフリー
自動車、輸送機械	自動運転の安全性向上、小型軽量化、省電力
医療、介護、補助	応答性能向上、小型軽量化、省電力、低価格化
航空宇宙、原子力	フォールトトレラントコントローラ信頼性向上、小型化、省電力

機能安全規格には CPU に対するランダムハードウェア故障への対策メカニズムは明確になっているが、FPGA での多様化の要件は発展途上である。機能安全が必要となる分野の拡大とともに、本研究で示したアプローチが規格成熟の一助になることを期待したい。

さらに SGC は制御ネットワークのセキュリティ対策においても有効である。近年 IoT や第四次産業革命と称し、様々な制御システムがネットワークに接続し、より効率よく設備を稼働させる取り組みが進んでいる。このような取り組みは、従来孤立して動作することが前提であった制御システムが、ネットワークからの障害やサイバー攻撃にさらされる危険性が指摘されており、各国で対策の法令化やガイドラインが整備されている[110]。たとえば米国原子力発電所で発生した設備故障に起因するパケットストームによる重要設備のコントローラの機能不能のような事故[111]は、制御演算と通信処理を分離並行処理できる SGC であれば発生を防ぐことができる。SGCは脆弱性の大きな根源となるようなOSを必要としないため、ネットワーク経由のサイバー攻撃にも有効である。スマートグリッドのように広域に分散設置される配電設備ではOSのアップグレードが不要になる。港湾や空港などの公安施設の警備システムのコントローラとしても活用が期待できる。

謝辞

本論文は、元 長崎大学大学院工学研究科 電気・情報科学部門 黒川不二雄教授（現長崎総合科学大学大学院 新技術創成研究所 特命教授）、長崎大学大学院工学研究科 電気・情報科学部門 柴田裕一郎准教授から、懇切なご指導及びご鞭撻を賜りました。丸田英徳准教授には、学会への論文提出にあたり多大な労を煩わせましたこと、お礼申し上げるとともにお詫びいたします。また本論文のとりまとめにあたり、樋口剛教授、辻峰男教授には、貴重なご支援を賜りました。長崎大学大学院工学研究科未来志向制御計測学講座 江藤春日教授のお誘いと励ましのおかげで、このような機会と貴重な体験を得ることができました。田中雅晴准教授には、三菱重工業勤務時より FPGA について様々な知見をご教授賜りました。ありがとうございました。

長崎総合科学大学工学部工学科電気電子工学コース 松井信正教授には、論文編さんのご指導をいただきました。株式会社 NTT データ青木聡氏には論文校正にご支援いただきました。ありがとうございました。シミュレーション試験、実験データ整理では、柴田研究室白倉雄大氏に多大なご協力いただきました。

最後に、起業独立と同時の大学院入学という事態にも、慌てず変わらず日常を支えてくれた妻 和子に感謝します。

ここに記し、皆様に深甚なる感謝の意を表します。

参考文献

(1) S. Bennett, "Control and the digital computer: the early years," Proceedings of International Federation of Automatic Control (IFAC), vol. 35, no. 1, pp. 237-242, July 2002.

(2) 千本資、花渕太, "計装システムの基礎と応用," オーム社／横河電機株式会社, Sept. 1987.

(3) A. Danowitz, K. Kelley, J. Mao, J. P. Stevenson and M. Horowitz, "CPU DB recording microprocessor history," ACM Queue-Processors, vol. 10, no. 4, pp. 1-10, Apr. 2012.

(4) P. E. Ross, "Why CPU frequency stalled," IEEE Spectrum, vol. 45, no. 4, Apr. 2008.

(5) C. Moore, "Data processing in exascale-class computer systems," Proceedings of Salishan Conference on High Speed Computing, pp. 1-18, Apr. 2011.

(6) Intel Corporation, "Applying multi-core and virtualization to industrial and safety-related applications," White Paper of Wind River® Hypervisor and Operating Systems Intel® Processors for Embedded Computing, 2009.

(7) J. Wilhite, T, Yeh, "Divide and conquer: multi-core processors and automation applications," White Paper of Advantech Corporation, 2009.

(8) T. Shindo, "Lightning strike fault risk on wind power generation system," Proceedings of International Symposium on Electromagnetic Compatibility (EMC), pp. 593-596, May 2014.

(9) Intel Corporation, "Optimizing software for multi-core processors," White Paper of Intel Corporation Prefetching, 2007.

(10) J. Turley and Intel Corporation, "Introduction to Intel® architecture," White Paper of Introduction to Intel® Architecture, 2014.

(11) D. Patterson and J. L. Hennessy, "Computer organization and design 5th Edition," Elsevier, pp. 40, Sept. 2013.

(12) J. Cownie, "Multicore: the software view," Proceedings of Intel EMEA ACADEMIC FORUM, 2007.

(13) E. Irmak, I. Colak and H. I. Bulbul, "FPGA based parallel connection system of separate voltage sources," Proceedings of International Conference on Renewable Energy Research and Applications (ICRERA), pp. 1-3, Nov. 2012.

(14) 小林和淑, "半導体の耐性試験-加速器によるシングルイベント耐性の実測評価," 日本加速器学会「加速器」 vol.13, no. 4, pp. 1-6, 2016.

(15) IEC, "IEC 61508 Functional safety of electrical/electronic/programmable electronic safety-related systems," Part 1-7, Apr. 2010.

(16) IPA 独立合成法人 情報処理推進機構, "重要インフラの制御システムセキュリティとIT サービス継続に関する調査," Mar. 2009.

(17) IEC, "IEC 61511 Functional safety - safety instrumented systems for the process industry sector," Part 1-3, Feb. 2016.

(18) IEC, "IEC 60601 Medical electrical equipment - general requirements for basic safety and essential performance," Aug. 2012.

(19) CENELEC, "EN 50126 Railway applications the specification and demonstration of reliability, availability, maintainability and safety (RAMS)," Dec. 2009.

(20) ISO, "ISO/DIS 26262 Road vehicles functional safety," Part 1-10, Nov. 2011.

(21) ISO, "ISO 13849 Safety of machinery -safety-related parts of control systems," Dec. 2015.

(22) ISO, "ISO/TR 23849 Guidance on the application of ISO 13849-1 and IEC 62061 in the design of safety-related control systems for machinery," Jul. 2010.

(23) ISO, "ISO 10218 Robots and robotic devices - safety requirements for industrial robots," Jul. 2011.

(24) ISO, "ISO 13482 Robots and robotic devices - safety requirements for personal care robots," Feb. 2014.

(25) ISO, "ISO/TS 15066 Robots and robotic devices - collaborative robots," Feb. 2016.

(26) ISO, "ISO 12100 Safety of machinery - general principles for design - risk assessment and risk reduction," Nov. 2010.

(27) ISO, "ISO 15998 Earth-moving machinery - machine control systems (MCS) using electronic components," Apr. 2008.

(28) IEC, "IEC 62061 Safety of machinery - functional safety of safety-related electrical, electronic and programmable electronic control systems," Nov. 2011.

(29) IEC, "IEC 61800 Adjustable speed electrical power drive systems part 5-2: safety requirements," Apr. 2016.

(30) IEC, "IEC 60730 Automatic electrical controls part 1: general requirements," Nov. 2013.

(31) IEC, "IEC 60335 Household and similar electrical appliances - safety," Dec. 2013.

(32) IEC, "IEC 61131 Programmable controllers," Part 1-6, Oct. 2012.

(33) 川村貞夫、石川洋次郎, "工業計測と制御の基礎," 工業技術社, Apr. 2003.

(34) M. Assante and T. Conway, "An abbreviated history of automation & industrial controls systems and cybersecurity," White paper of SANS Institute Analyst, Aug. 2014.

(35) K. hoshmandi and M. montazeri, "Long range predictive PID control for nonlinear boiler-turbine dynamics," Proceedings of International Conference on Industrial Technology (ICIT), pp. 1-6, Feb. 2009.

(36) A. Kamiya, A. Kakei, K. Kawai and S. Kobayashi, "Advanced power plant start-up automation based on the integration of soft computing and hard computing techniques," Proceedings of International Conference on Systems, Man and Cybernetics (SMC), pp. 380-385, Oct. 1999.

(37) H. F. Hamann, V. Lopez and A. Stepanchuk, "Thermal zones for more efficient data center energy management," Proceedings of Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), pp. 1-6, June 2010.

(38) R. P. Mandi and U. R. Yaragatti, "Energy efficiency improvement of auxiliary power equipment in thermal power plant through operational optimization," Proceedings of International Conference on Power Electronics, Drives and Energy Systems (PEDES), pp. 1-8, Dec. 2012.

(39) K. H. John and M. Tiegelkamp, "Programming industrial automation systems chapter 7 innovative PLC programming systems," Springer International Publishing AG., pp. 243-282, 2001.

(40) M. K. Paul, B. Hamane, M. L. Doumbia and A. Cheriti, "Pitch control of a wind energy conversion system based on permanent magnet synchronous generator (PMSG)," Proceedings of International Conference on Ecological Vehicles and Renewable Energies (EVER), pp. 1-7, Mar. 2015.

(41) N. Luo, Y. Vidal and L. Acho, "Wind turbine control and monitoring," Springer International Publishing AG., 2014.

(42) V. Prema, S. Dutta, K. U. Rao, S. Shekhar and B. S. Kariyappa, "Effective battery usage strategies for hybrid power management," Proceedings of International Conference on Power and Advanced Control Engineering (ICPACE), pp. 95-98, Aug. 2015.

(43) H. Chen, P. C. Xiong, K. Schwan, A. Gavrilovska and C. Z. Xu, "A cyber-physical Integrated system for application performance and energy management in data centers," Proceedings of International Green Computing Conference (IGCC), pp. 1-10, June. 2012.

(44) Y. Chen, D. Gmach, C. Hyser, W. Zhikui, C. Bash, C. Hoover and S. Singhal, "Integrated management of application performance, power and cooling in data centers," Proceedings of Network Operations and Management Symposium (NOMS), pp. 615-622, Apr. 2010.

(45) P. Paschke, M. Plonczak, P. Klis and M. Grunt, "Perspectives of development of integrated monitoring system of power supply and air conditioning equipment towards technical environment equipment monitoring system of the operator," Proceedings of International Telecommunications Energy Conference (INTELEC), pp. 1-6, Sept. 2008.

(46) T. Walker, J. Bojarski and F. Kling, "An “open” DCS solution for cement process control," Proceedings of Cement Industry Technical Conference, pp. 377-392, Apr. 1996.

(47) S. Bennett, "A brief history of automatic control," IEEE Control Systems, vol. 16, no. 3, pp. 17-25, June 1996.

(48) IEC, "IEC 62280 Railway applications - communication, signaling and processing systems - safety related communication in transmission systems," Feb. 2014

(49) IEC, "IEC 60950 Information technology equipment - safety," May 2013.

(50) IEC, "IEC 62443 Industrial communication networks -network and system security," Nov. 2011.

(51) S. Li, H. Liu, W. J. Cai, Y. C. Soh and Li. H. Xie, "A new coordinated control strategy for boiler-turbine system of coal-fired power plant," IEEE Transactions on Control Systems Technology vol.13, pp. 943-954, Nov. 2005.

(52) I. Yousefi, M. Yari and M. A. Shoorehdeli, "Modeling, identification and control of a heavy duty industrial gas turbine," Proceedings of International Conference on Mechatronics and Automation (ICMA), pp. 611-615, Aug. 2013.

(53) G.P.Lim and H.H.Lee, "Development and application of boiler combustion air flow control algorithm for coal-fired power plant," Proceedings of International Conference on Industrial Electronics and Applications (ICIEA), pp. 662-667, June 2013.

(54) Acatech (National Academy of science and Engineering) Germany, "Recommendations for implementing the strategic initiative INDUSTRIE 4.0," Apr. 2013.

(55) Acatech (National Academy of science and Engineering) Germany, "Implementation strategy industrie 4.0 report on the results of the industrie 4.0 platform," Jan. 2016.

(56) Renesas Electronics Corporation, "https://www.renesas.com/ja-jp/products/microcontrollers-microprocessors/rx.html," May 2017.

(57) J. Figueiredo, V. Carvalho, J. Machado and F. Soares, "Modelica modeling language as a tool on control engineering education: simulation of a two-tank system," Proceedings of International Conference on Teaching, Assessment and Learning (TALE), pp 6 − 11, Dec. 2014.

(58) Z. Song and D. Li, "Decomposition and composition mechanism applied in control system modeling language," Proceedings of International Conference on Circuits, Communications and System (PACCS), pp. 453-455, Aug. 2010.

(59) H. Asadipooya and A. A. Safavi, "Enhancement of model predictive control implementation on a DCS PCS7," Proceedings of International Conference on Electrical Engineering (ICEE), pp. 1272-1277, May 2016.

(60) Y. Si, H. R. Karimi and H. Gao, "Parameter tuning for nacelle-based passive structural control of a spar-type floating wind turbine," Proceedings of Industrial Electronics Society (IECON), pp. 7687-7691, Nov. 2013.

(61) A. H. Dida, M. Bourahla, H. B. Ertan, M. Benghanem and M. E. Benzina, "Three phase inverter speed control of AC drives motor using DSPic microcontroller," Proceedings of International Aegean Conference on Electrical Machines & Power Electronics (ACEMP), pp. 93-101, Sept. 2015.

(62) CENELEC, "EN 50128 Railway applications -communication, signaling and processing systems -software for railway control and protection systems," Jun. 2011.

(63) CENELEC, "EN 50159 Railway applications -communication, signaling and processing systems part 2: safety related communication in open transmission systems," Mar. 2001

(64) N. Kurd, P. Mosalikanti, M. Neidengard, J. Douglas and R. Kumar, "Next generation Intel® Core™ micro-architecture (Nehalem) clocking," IEEE Journal of Solid-State Circuits, vol. 44, no. 4, pp. 1121-1129, Apr. 2009.

(65) H. Kim, A.K.lu and R. Rajkumar, "A coordinated approach for practical OS-level cache management in multi-core real-time systems," Proceedings of Euromicro Conference on Real-Time Systems (ECRTS), pp. 80-89, July 2013.

(66) K. Gilles, S. Groesbrink, D. Baldin and T. Kerstan, "Proteus hypervisor: full virtualization and paravirtualization for multi-core embedded systems," Proceedings of International Embedded Systems Symposium (IESS), pp. 293-305, June 2013.

(67) C. Main and TenAsys Corporation, "Virtualization on multi-core for industrial RTOS systems," 2008.

(68) J. Ralston and Freescale Semiconductor, Inc., "Managing machine safety and productivity with qoriq multicore processors," 2013.

(69) Intel Corporation, "Improving real-time performance by utilizing cache allocation technology," Apr. 2015.

(70) J. T. Welch and J. Carletta, "A direct mapping FPGA architecture for industrial process control application," Proceedings of International Conference on Computer Design (ICCD), pp. 595-598, Sept. 2000.

(71) D.A. Lee, E.S. Kim, J. Yoo, J.S. Lee and J. G. Choi, "FBD to Verilog 2.0: an automatic translation of FBD into Verilog to develop FPGA," Proceedings of International Conference on Information Science and Application (IcISA), pp. 1-4, May 2014.

(72) M. Kocur, S. Kozak and B. Dvorscak, "Design and implementation of FPGA - digital based PID controller," Proceedings of International Carpathian Control Conference (ICCCC), pp. 233-236, May 2014.

(73) E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems-a review," IEEE Transactions on Industrial Electronics, vol. 54, Issue:4, p.p. 1824-1842, Aug. 2007.

(74) G. Singh and S. S. Gill, "Design and Implementation of process controller for direct digital control on FPGA," Proceedings of International Conference on Machine Intelligence and Research Advancement (ICMIRA), pp. 326-330, Dec. 2013.

(75) S. W. A. Hashmi, M. Rehan, M. Aamir, H. Kumar and F. Liaquat, "Distributed process monitoring and control using FPGA," Proceedings of International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronics System (VITAE), pp. 1-5, May 2014.

(76) I. Moussa, A. Bouallegue and A. Khedher, "3 kW wind turbine emulator implementation on FPGA using Matlab/Simulink," International Journal of Renewable Energy Research (IJRER), vol 5, no. 4, pp. 1154-1163, 2015.

(77) A. M. Colak, Y. Shibata and F. Kurokawa, "FPGA implementation of the automatic multiscale based peak detection for real-time signal analysis on renewable energy systems," Proceedings of International Conference on Renewable Energy Research and Applications (ICRERA), pp. 379 - 384, Nov. 2016.

(78) Y.Y.Tzou and T.S.Kuo, "Design and implementation of all FPGA-based motor control IC for permanent magnet AC servo motors," Proceedings of International Conference on Industrial Electronics Society, Control and Instrumentation (IECON) vol.2, pp. 943-947, Nov. 1997.

(79) C. Wang, W. Li and J. Belanger, "Real-time and faster-than-real-time simulation of Modular Multilevel Converters using standard multi-core CPU and FPGA chips," Proceedings of Industrial Electronics Society (IECON), pp. 5405-5411, Nov. 2013.

(80) J. Khalifat, A. Ebrahim, A. Adetomi and T. Arslan, "A dynamic partial reconfiguration design for camera systems," Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems (AHS), pp. 1-7, June 2015.

(81) L. Ngalamou and L. Myers, "An industrial process control studio (IPCS) development tool: the IEC 61131-3 model capture features," Proceedings of International Joint Conference SIE-ICASE, pp.2874-2878, Oct. 2006

(82) F. Kastensmidt and R. Reis, "Soft error rate and fault tolerance techniques for FPGAs," Circuit Design for Reliability Springer Science+Business Media New York, pp. 207-221, 2015.

(83) R. A. Ashraf, O. Mouri, R. Jadaa and R. F. Demara, "Design-for-diversity for improved fault-tolerance of TMR systems on FPGAs," Proceedings of International Conference on Reconfigurable Computing and FPGAs (ReConFig), pp. 99-104, Nov. 2011.

(84) L. A. Tambara, F. L. Kastensmidt, J. R. Azambuja, E. Chielle, F. Almeida, G. Nazar, P. Rech, C. Frost and M. S. Lubaszewski, "Evaluating the effectiveness of a diversity TMR scheme under neutrons," Proceedings of European Conference on Radiation and Its Effects on Components and Systems (RADECS), pp. 1-5, Sept. 2013.

(85) P. Koopman, "A case study of toyota unintended acceleration and software Safety," Presentation of International Symposium on Trust, Security and Privacy for Emerging Applications (TSP), keynote, No.8, Nov. 2014.

(86) P. Koopman, "32-Bit cyclic redundancy codes for internet applications," Proceedings of International Conference on Dependable Systems and Networks (DSN), pp. 459-468, Feb. 2002.

(87) P. Koopman and T. Chakravarty, "Cyclic redundancy code (CRC) polynomial selection for embedded networks," Proceedings of International Conference on Dependable Systems and Networks (DSN), pp. 145-154, Jan. 2004.

(88) IEC, "IEC 61784 industrial communication networks - profiles - part 3-18: functional safety fieldbuses," Aug. 2009.

(89) E. Hallett, G. Corradi and S. McNeil, "Xilinx reduces risk and increases efficiency for IEC61508 and ISO26262 certified safety applications," Xilinx white paper WP461 (v1.0), Apr. 2015.

(90) W. Wei, Y. Jun and and Z. Mei-jie, "The research of FPGA reliability based on redundancy methods," Proceedings of International Conference of Computer Science and Network Technology (ICCSNT), pp. 1608-1611, Dec. 2011.

(91) M. Radu, "Reliability and fault tolerance analysis of FPGA platforms," Proceedings of Long Island Systems Applications and Technology Conference (LISAT), pp. 1-4, May 2014.

(92) A. Hayek, M AI-Bokhaiti and J. Borcsok, "Design and implementation of an FPGA-based 1004-architecture for safety-related system-on-chips," Proceedings of International Conference of Microelectronics (ICM), pp. 1-4, Dec. 2013.

(93) R. Noji, S. Fujie, Y. Yoshikawa, H. Ichihara and T. Inoue, "Reliability and performance analysis of FPGA-based fault tolerant system," Proceedings of International Symposium of Defect and Fault Tolerance in VLSI Systems (DFT), pp. 245-253, Oct. 2009.

(94) B. Harikrishna and S. Ravi, "A survey on fault tolerance in FPGAs," Proceedings of International Conference on Intelligent Systems and Control (ISCO) 2013, pp. 265-270, Jan. 2013.

(95) T. Lovric, "Systematic and design diversity - Software techniques for hardware fault detection," Dependable Computing - EDCC-1 Lecture Notes, in Computer Science, vol. 852, pp. 307-326, Oct. 1994.

(96) J. Cong, B. Liu, S. Neuendorffer, J. Noguera, K. Vissers and Z. Zhang, "High-level synthesis for FPGAs: from prototyping to deployment," IEEE Transactions on Computer-Aided Design of Integrated Circuits and System, pp. 473-491, Apr. 2011.

(97) K. Wakabayashi, "Use of high-level synthesis to generate hardware from software," IEICE Fundamentals Review, vol.6, no.1, pp. 37-50, July 2012.

(98) T. Fischer, J. Desai, B. Doyle, S. Naffziger and B. Patella, "A 90-nm variable frequency clock system for a power-managed itanium architecture processor," IEEE Journal of Solid-State Circuits, vol. 41, no. 1, pp. 218-228, Jan. 2006.

(99) T.D. Burd, T.A. Pering, A.J. Stratakos and R.W. Brodersen, "A dynamic voltage scaled microprocessor system," IEEE Journal of Solid-State Circuits, vol. 36, no. 11, pp. 1571-1580, Nov. 2000.

(100) S. Tam, S. Rusu, U. Nagarji Desai, R. Kim, Ji Zhang and I. Young, "Clock generation and distribution for the first IA-64 microprocessor," IEEE Journal of Solid-State Circuits, vol. 35, no. 11, pp. 1545-1552, Nov. 2000.

(101) K. Morimoto, Y. Shibata, Y. Shirakura, H. Maruta, F. Kurokawa, M. Nobe and M. Tanaka, "A new FPGA based green controller using modeling language," International Journal of Renewable Energy Research (IJRER), vol. 6, no. 2, pp. 715-722, 2016.

(102) S. Dabral, S. Kamath and V. Appia, "Trends in camera based automotive driver assistance systems (ADAS)," Proceedings of International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1110-1115, Aug. 2014.

(103) K. Stephan, "Exploding galaxies: how to do recalls right," IEEE Consumer Electronics Magazine, vol. 6, Issue:2, pp. 99-100, Apr. 2017.

(104) A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding and B. Matthias, "Safety in human-robot collaborative manufacturing environments: metrics and control," IEEE Transactions on Automation Science and Engineering, vol. 13, Issue:2, pp. 882-893, Apr. 2015.

(105) K. Morimoto, Y. Shibata, Y. Shirakura, H. Maruta, F. Kurokawa and M. Tanaka, "Diversity diagnostic for new FPGA based controller of renewable energy power plant," International Journal of Renewable Energy Research (IJRER), vol.7, no 3, pp. 1403-1412, 2017.

(106) IEC, "IEC/TR 62380 Reliability data handbook –universal model for reliability prediction of electronics components, PCBs and equipment," Aug. 2004.

(107) Siemens AG CT TIM IR Munich Germany, “SN29500 failure rates of components part 1-16 Siemens norm,” Apr. 2015.

(108) 国土交通省自動車局 審査・リコール課, “各年度のリコール届出件数及び対象台数,” 自動車のリコール・不具合情報, 2017.

(109) 一般財団法人家電製品協会家電製品 PL センター,”2016 年度家電製品 PL センター事業報告”, 2016 年度年次報告書, July 2017.

(110) U.S. department of commerce National Institute of Standard and Technology (NIST), “NIST special publication 800-82 guide to industrial control systems (ICS) security,” revision 2, May 2015.

(111) United States nuclear regulatory commission office of Nuclear Reactor Regulation, “NRC information notice:2007-15 effects of ethernet-based, non-safe and continued operation of nuclear power stations,” Apr. 2007.