(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2015-170367
(P2015-170367A)
(43)公開日　平成27年9月28日(2015.9.28)

(51)Int.Cl.			F I			テーマコード（参考）
G05B	19/042	(2006.01)	G05B	19/042		5B081
G06F	9/45	(2006.01)	G06F	9/44	322F	5H220

審査請求 未請求　請求項の数 14　ＯＬ　（全 18 頁）

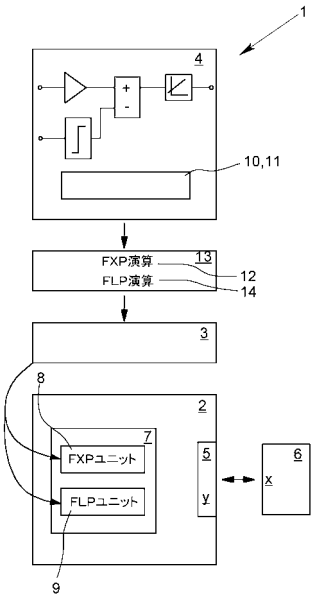
(21)出願番号　特願2015-44745(P2015-44745)
(22)出願日　平成27年3月6日(2015.3.6)
(31)優先権主張番号　14158000.1
(32)優先日　平成26年3月6日(2014.3.6)
(33)優先権主張国　欧州特許庁(EP)

(71)出願人　506012213
ディスペース　デジタル　シグナル　プロセッシング　アンド　コントロール　エンジニアリング　ゲゼルシャフト　ミット　ベシュレンクテル　ハフツング
ｄｓｐａｃｅ　ｄｉｇｉｔａｌ　ｓｉｇｎａｌ　ｐｒｏｃｅｓｓｉｎｇ　ａｎｄ　ｃｏｎｔｒｏｌ　ｅｎｇｉｎｅｅｒｉｎｇ　ＧｍｂＨ
ドイツ連邦共和国　パデルボルン　ラーテナウシュトラーセ　２６
Ｒａｔｈｅｎａｕｓｔｒ．２６，Ｄ－３３１０２　Ｐａｄｅｒｂｏｒｎ，　Ｇｅｒｍａｎｙ

最終頁に続く

(54)【発明の名称】グラフィック制御モデルから制御システム上で実行可能な制御プログラムを形成するためにコンピュータ上で実行される方法およびコンピュータプログラム

(57)【要約】
【課題】グラフィック制御モデルから、実行可能な制御プログラムを生成するための方法を提供して、この方法により、上記の制御システムの資源が一層良好に利用できるようにすること。
【解決手段】グラフィック制御モデルをプログラムコードに翻訳して、生成したこのプログラムコードが少なくとも１つのＦＸＰ演算および少なくとも１つのＦＬＰ演算を有しており、かつ、生成したこのプログラムコードをつぎに上記の実行可能な制御プログラムに翻訳して、上記の制御システム上でこの制御プログラムを実行する際にこの制御プログラムの一部分が上記のＦＸＰユニット上で実行され、この制御プログラムの他の一部分が上記のＦＬＰユニット上で実行されるようにする。
【選択図】図２



【特許請求の範囲】
【請求項１】
グラフィック制御モデル（４）から、制御システム（２）上で実行可能な制御プログラム（３）を形成するため、コンピュータ上で実現される方法（１）であって、
前記制御システム（２）は、プロセスインタフェースを有しており、
前記制御システム（２）は、前記プロセスインタフェース（５）を介して、物理プロセス（６）の少なくとも１つのプロセス変量を前記制御システム（２）によって検出し、および／または、前記物理プロセス（６）に影響を及ぼすための出力量を前記制御システム（２）によって送出することができるように構成されており、
前記制御システム（２）は、少なくとも１つの固定小数点数演算実行ユニット（ＦＸＰユニット，８）と、少なくとも１つの浮動小数点数演算実行ユニット（ＦＬＰユニット，９）とを有する少なくとも１つの電子計算ユニット（７）を有しており、
前記グラフィック制御モデル（４）は複数のモデル演算（１０，１１）を有している、コンピュータ上で実現される方法（１）において、
前記グラフィック制御モデル（４）をプログラムコード（１３）に翻訳して、生成される当該プログラムコード（１３）が少なくとも１つのＦＸＰ演算（１２）と少なくとも１つのＦＬＰ演算（１４）を有するようにし、
生成された前記プログラムコード（１３）を前記実行可能な制御プログラム（３）に翻訳して、前記制御システム（２）上で前記制御プログラム（３）を実行する際に当該制御プログラム（３）の一部分が前記ＦＸＰユニット（８）上で実行され、当該制御プログラム（３）の別の一部分が前記ＦＬＰユニット（９）で実行されるようにした、
ことを特徴とする方法（１）。
【請求項２】
請求項１に記載の、コンピュータ上で実現される方法（１）において、
時間平均において前記制御システム（２）の前記計算ユニット（７）の前記ＦＸＰユニット（８）および前記ＦＬＰユニット（９）の負荷が同程度になるような個数の前記制御モデル（４）のモデル演算（１０，１１）を、前記制御プログラムコード（１３）の前記ＦＸＰ演算（１２）および前記ＦＬＰ演算（１４）に翻訳する、
ことを特徴とする方法（１）。
【請求項３】
請求項１または２に記載の、コンピュータ上で実現される方法（１）において、
前記制御システム（２）の前記計算ユニット（７）上で前記制御プログラム（３）を実行する際に、前記プログラムコード（１３）が、前記計算ユニット（７）の前記ＦＸＰユニット（８）および前記ＦＬＰユニット（９）上で同時に実行されるように、前記プログラムコード（１３）のＦＸＰ演算（１２）と、前記プログラムコード（１３）のＦＬＰ演算（１４）とを混合する、
ことを特徴とする方法（１）。
【請求項４】
請求項１から３までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記プログラムコード（１３）のＦＬＰ演算（１４）の個数に対するＦＸＰ演算（１２）の個数の比は、単位時間当たりに前記ＦＸＰユニット（８）上で実行可能なＦＸＰ演算と、前記ＦＬＰユニット（９）上で実行可能なＦＸＰ演算との比に対応する、
ことを特徴とする方法（１）。
【請求項５】
請求項１から４までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記ＦＸＰユニット（８）上で計算するため、前記制御モデル（４）の整数のモデル演算を前記プログラムコード（１３）のＦＸＰ演算（１２）に翻訳し、例えば、ループ、switch-case演算またはビット演算に対するインクリメントまたはデクリメントカウンタを

前記プログラムコード（１３）のＦＸＰ演算（１２）に翻訳する、
ことを特徴とする方法（１）。
【請求項６】
請求項１から５までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記プログラムコード（１３）におけるＦＸＰ演算に翻訳する際に、固定小数点数リスケーリングが行われ得る、前記制御モデル（４）のモデル演算（１０，１１）を、前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項７】
請求項１から６までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記計算ユニット（７）がｍビットＦＬＰユニット（９）を有しているがｍビットＦＸＰユニット（８）を有していない場合、前記プログラムコード（１３）におけるＦＸＰ演算として翻訳するときに、当該プログラムコードにおいて１つまたは複数のｍビットＦＸＰ演算が生じ得る、前記制御モデル（４）のモデル演算（１０）を、その代わりに前記プログラムコードにおける１つまたは複数のｍビットＦＬＰ演算に変換する、
ことを特徴とする方法（１）。
【請求項８】
請求項１から７までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記制御モデル（４）が複数の並列の矢印線を有する場合、前記制御モデル（４）の複数のモデル演算（１０，１１）の矢印線のうちで、ＦＸＰ演算に翻訳するときに前記プログラムコード（１３）おいて最も多くのＦＸＰリスケーリングを有し得る矢印線を、前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項９】
請求項１から８までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記制御モデル（４）における少なくとも１つのモデル演算を、ＦＬＰ演算（１１）としかつ前記プログラムコード（１３）のＦＸＰ演算（１２）に変換し、および／または、
前記制御モデル（４）における少なくとも１つのモデル演算を、ＦＸＰ演算（１０）としかつ前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項１０】
請求項９に記載の、コンピュータ上で実現される方法（１）において、
変換によって生じる、前記プログラムコード（１３）における固定小数点データ型から浮動小数点データ型への、および／または、前記プログラムコード（１３）における浮動小数点データ型から固定小数点データ型への型変換の個数が少なくとも局所的に最小になるように、前記制御モデル（４）のＦＬＰ演算（１１）として決定され選択されたモデル演算を前記プログラムコード（１３）のＦＸＰ演算（１２）に変換し、および／または、前記制御モデル（４）のＦＸＰ演算（１２）として決定され選択されたモデル演算を前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項１１】
請求項１から１０までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記グラフィック制御モデル（４）から形成したプログラムコード（１３）は、ハードウェア記述言語のプログラムコードであり、
前記プログラムコード（１３）は、少なくとも１つのＦＸＰユニット（８）および少な

くとも１つのＦＬＰユニット（９）を有する制御システム（２）の前記計算ユニット（７）を表し、
当該計算ユニット（７）により、前記制御モデル（４）の機能が実現される、
ことを特徴とする方法（１）。
【請求項１２】
請求項１１に記載の、コンピュータ上で実現される方法（１）において、
前記ハードウェア記述言語は、ＦＰＧＡを表しかつこれをプログラムする、
ことを特徴とする方法（１）。
【請求項１３】
コンピュータプログラムを有するコンピュータプログラム製品において、
前記コンピュータプログラムがコンピュータによって実行される場合、前記コンピュータプログラムは、請求項１から１２までのいずれか１項に記載の方法を実施するためのソフトウェア手段を有する、
ことを特徴とするコンピュータプログラム製品。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、グラフィック制御モデルから、制御システム上で実行可能な制御プログラムを形成するための方法であって、コンピュータ上で実現される方法に関する。この制御システムは、プロセスインタフェースを有しており、またこの制御システムは、このプロセスインタフェースを介して、物理プロセスの少なくとも１つのプロセス変量を上記の制御システムによって検出し、および／または、物理プロセスに影響を及ぼすための出力量を上記の制御システムによって送出することができるように構成されている。この制御システムは、少なくとも１つの固定小数点演算実行ユニット（ＦＸＰユニット）と、少なくとも１つの浮動小数点演算実行ユニット（ＦＬＰユニット）とを備えた少なくとも１つの電子計算ユニットを有しており、上記のグラフィック制御モデルは、複数のモデル演算を有する。
【背景技術】
【０００２】
グラフィック制御モデルから、実行可能な制御プログラムをコンピュータ支援によって（すなわち計算機によって自動化されて）形成する上記のような方法は、かなり以前から公知であり、また例えば制御装置のような制御システム用のプログラムを形成するための開発環境の基礎的な機能に属している。上記のグラフィック制御モデルは、ほとんどの場合にブロック線図の形態をしており、このブロック線図により、例えば、制御アルゴリズムの数学的な機能がモデル化されて表される。産業分野における業務においてこのような制御システムは、上位の技術プロセスに埋め込まれる小型計算機であることが多く、自動車分野ではこのような制御システムは制御装置と称される。
【０００３】
グラフィック制御モデルから、実行可能な制御プログラムを形成するための方法は具体的にはコードジェネレータと称される。このコードジェネレータにより、抽象的な機能記述（グラフィック制御モデル）がターゲットシステム（制御システム）用のハードウェア寄りのプログラムに確実かつ誤りなく翻訳されることが保証される。制御プログラムの形成方法は、例えばDE 10347891 A1により詳細に記載されている。
【０００４】
制御システムの計算ユニットは一般的に、英語ではInteger-Execution-Unit（ＩＥＵ）と称される整数計算ユニットを有しており、以下ではこれをＦＸＰユニット（Fixed-Point-Executionユニット）と称する。この整数または固定小数点演算ユニットを用いれば、整数も固定小数点数（Fixed-Point-Number）も計算することができる。なぜならば、固定小数点数演算は基本的に整数演算に相応するからである。以下ではFixed-Point-Operation（かつ整数演算ではない演算）およびFloating-Point-Operation（浮動小数点数演算、

ＦＬＰ演算）について扱うため、ＦＬＰユニット（Floating-Point-Execution-Unit）称されることが一般的である浮動小数点計算ユニットに対し、上記の整数計算ユニットをＦＸＰユニット（Fixed-Point-Execution-Unit）と称する。このような理解によれば、固定小数点演算（Fixed-Point-Operation）には整数演算（Integer-Operation）が含まれる。

【０００５】

殊に組み込み式制御システムの電子計算ユニットはかつてただ１つのＦＸＰユニットしか有しないことが多かったため、従来技術から公知の方法では、グラフィック制御モデルのすべてのモデル演算はもっぱら、上記の生成されるプログラムコードのFixed-Point-Operation（ＦＸＰ演算）に翻訳されていた。この手法によって確かに、適用された上記の方法によって生成されるプログラムコードおよびこれから翻訳した制御プログラムはその原理にしたがい、実質的に任意の計算ユニット上で動作可能であることが保証されたが、制御システムの電子計算ユニットがＦＸＰユニットの他にＦＬＰユニットを有する場合には、上記のハードウェア資源は部分的にだけしか使用されなかったのである。ここで考察している制御システムは一般的にリアルタイム条件下で動作するサンプリングシステムを扱っており、つまり上記の制御プログラムの実行に対して任意の多くの時間を費やすことができないため、上記のＦＸＰユニットがフルに使われ、あらかじめ設定された時間インターバルでさらなる計算ステップをもはや実行できない場合には、一層性能のよい制御システムを使用しなければならないのである。

【先行技術文献】

【特許文献】

【０００６】

【特許文献１】DE 10347891 A1

【発明の概要】

【発明が解決しようとする課題】

【０００７】

本発明の課題は、グラフィック制御モデルから、実行可能な制御プログラムを形成するための方法を提供して、この方法により、上記の制御システムの資源が一層良好に利用できるようにすることである。

【課題を解決するための手段】

【０００８】

上記の課題は、実行可能な制御プログラムを形成する上記の方法においてまず、上記のグラフィック制御モデルをプログラムコードに翻訳して、生成したこのプログラムコードが少なくとも１つのＦＸＰ演算および少なくとも１つのＦＬＰ演算を有しており、かつ、生成したこのプログラムコードをつぎに上記の実行可能な制御プログラムに翻訳して、上記の制御システム上でこの制御プログラムを実行する際にこの制御プログラムの一部分が上記のＦＸＰユニット上で実行され、この制御プログラムの別の一部分が上記のＦＬＰユニット上で実行されるようにすることによって解決される。

【図面の簡単な説明】

【０００９】

【図１】制御システム上で実行可能な制御プログラムを形成するための従来技術から公知の方法を示す図である。

【図２】制御システム用の制御プログラムを形成するための本発明の方法を略示する図である。

【図３】制御プログラムを形成するための本発明の方法の別の実施形態を略示する図である。

【図４】制御プログラムを形成するための本発明の方法の、図３に基づく別の実施形態を略示する図である。

【図５】制御プログラムに翻訳すべきグラフィックモデルを示す図である。

【図６】ＦＬＰ演算によって実現される特徴的なモデル部分を有する、図５のグラフィック制御モデルを示す図である。

【図７】ＦＬＰ演算に翻訳される別の特徴的なモデル部分を有する、図５の制御モデルを示す図である。
【発明を実施するための形態】
【００１０】
本発明による上記の方法によって回避されるのは、上記のグラフィック制御モジュールをプログラムコードに翻訳する際に、上記のＦＸＰユニットだけで使用されるプログラムコードだけが形成されるか、または、上記のＦＬＰユニットだけで使用されるプログラムコードだけが生成されることである。すなわち、本発明による方法において強制されるのは、ターゲットシステム上の上記のグラフィック制御モデルが、ＦＸＰユニット上でも、またＦＬＰユニット上でも計算されることであり、これによって全体として一層良好に資源が利用されることになる。公知の方法に比べ、同じ１つの制御システム上で、従来公知のコード生成方法によって形成される制御プログラムよりも格段に大量の制御プログラムを計算することができる。これによって可能になるのは、同じ１つの制御システムにより、上記のプロセスインタフェースを介して、一層高いサンプリングレートで、および／または、一層多くの個数のプロセス変量を交換して、接続された物理プロセスに影響を及ぼすことができる。
【００１１】
上記のコンピュータ上で実現される方法の一発展形態の特徴は、時間平均において上記の制御システムの計算ユニットのＦＸＰユニットおよびＦＬＰユニットの負荷が同程度になるような個数の制御モデルのモデル演算を、上記のプログラムコードのＦＸＰ演算およびＦＬＰ演算に変換することである。ＦＸＰユニットおよびＦＬＰユニットに同じように負荷を加えることにより、平均的にはこれらの２つのユニットの負荷は可能な限りに少なくなるため、これらのユニットにおける実行時間変動によって複数のリアルタイム条件が損なわれてしまう確率が小さくなる。
【００１２】
上記の有利な実施形態によれば、上記のプログラムコードのＦＬＰ演算の個数に対するＦＸＰ演算の個数の比は、単位時間当たりにＦＸＰユニット上で実行可能なＦＸＰ演算と、ＦＬＰユニット上で実行可能なＦＬＰ演算との比に対応する。この結果は、上記の制御システムの計算ユニットないしはこの計算ユニットの種々異なるユニットが、所定のＦＸＰおよびＦＬＰ演算にどのくらいのクロックサイクルが必要であるか考慮すれば、上記の方法により、容易に自動化して得ることができる。上記の制御モデルの変換すべきモデル演算の選択は、ランダムに行うことができるかまたは以下で検討する所定の選択判定基準にしたがって行うこともできる。
【００１３】
上記の制御プログラムによる上記の計算ユニットのＦＸＰユニットおよびＦＬＰユニットの利用の上記のただ１つの状況は、上記の計算の時間的な配位を考慮しなくても、それ自体ですでに有利である。殊に大きな効果は、コンピュータ上で実現される方法の別の有利な実施形態において、つぎのような場合に得られる。すなわち、制御システムの計算ユニット上で制御プログラムを実行する際に、上記のプログラムコードないしは制御プログラムが、計算ユニットのＦＸＰユニットおよびＦＬＰユニット上で同時に実行されるように、プログラムコードのＦＸＰ演算と、プログラムコードのＦＬＰ演算とを混合する。すなわち、ＦＸＰ演算とＦＬＰ演算とを必要に応じて混合することにより、ＦＸＰユニットおよびＦＬＰユニットは実際に、交互だけではなく同時にも計算を実行するようにＦＸＰユニットおよびＦＬＰユニットに命令が供給されることになり、これにより、実行時に最大限の効果が得られる。
【００１４】
本発明による方法において有利な作用は、例えば、制御モデルのモデル演算からランダムに選択したモデル演算を、プログラムコードの対応するＦＬＰ演算ないしはＦＸＰ演算に変換することによってすでに得られるが、この方法の有利な実施形態によれば、所期のように適用可能な判定基準があり、この判定基準によれば、制御モデルのどの部分が変換

に殊に適しているかを決定することできる。

【００１５】

コンピュータ上で実現される上記の方法の有利な実施形態では、ＦＸＰユニット上で計算するため、制御モデルの整数のモデル演算を、プログラムコードのＦＸＰ演算に翻訳する。例えば、ループ、switch-case演算用のインクリメントまたはデクリメントカウンタまたはビット演算のような上記の整数のモデル演算は、浮動小数点データ型に変換することの利点があると思われない。

【００１６】

しかしながら有利であり得るのは、プログラムコードにおいてＦＸＰ演算として実現する際に固定小数点数リスケーリングが行われ得る、制御モデルのモデル演算をプログラムコードのＦＬＰ演算に変換することである。このような手段により、固定小数点リスケーリングに対する計算ステップを節約することができるため、元々の浮動小数点計算の時間が節約されるか、ないしはプログラムコードにおけるＦＸＰからＦＬＰへの必要な変換がもはや重要ではなくなる。この発明の別の実施形態によれば、計算ユニットがｍビットＦＬＰユニットを有しているがｍビットＦＸＰユニットを有していない場合、プログラムコードにおけるＦＸＰ演算として翻訳するときに、プログラムコードにおいて１つまたは複数のｍビットＦＸＰ演算が生じ得る、制御モデルのモデル演算を、その代わりにプログラムコードにおける１つまたは複数のｍビットＦＬＰ演算に変換する。例えば、モデルの設定により、プログラムコードにおいて６４ビットＦＸＰ演算が生じるが、３２ビットＦＸＰユニットしか存在しない場合、生成されたコードでは、所要の６４ビット演算を複数の別個の３２ビットＦＸＰ演算によって模造しなければならないが、これは極めて繁雑である。この場合には、計算ユニット上に６４ビットＦＬＰユニットが設けられている場合には、６４ビットＦＬＰ演算への変換を行うことは大いに手間をかける甲斐があるのである。

【００１７】

グラフィック制御モデルでは、直列および並列に接続された演算があり、直列接続された演算は時間的に必ず相前後して実施しなければならない。なぜならば１つの演算の結果が、後続の演算の入力量になるからである。制御モデルの並列の矢印線ないしは並列の演算は時間的に並列に処理することができる。なぜならば異なる矢印線において計算される値は、互いに影響し合わないからである。

【００１８】

本発明による方法の有利な実施形態によれば、制御モデルが複数の並列な矢印線が有する場合、制御モデルのモデル演算の矢印線のうちで、最も多くのＦＸＰリスケーリングを有するか、ないしはＦＸＰ演算に変換するときに、最も多くのＦＸＰリスケーリングを有し得る矢印線をプログラムコードのＦＬＰ演算に変換する。先行してＦＬＰ演算が使用される場合、プログラムコードにおけるこれらのＦＸＰリスケーリングは省略される。この場合に有利には上記の矢印線は、ＦＬＰ演算とＦＸＰ演算との間で交換されるのではなく、プログラムコードにおいてＦＬＰ演算にとどまる。この際にはプログラムコードにおける独立したモデル矢印線のコード部分は交互に配置することもでき、これによって（例えば、使用されるコンパイラにおいて場合によっては存在する検出欠陥を保証するため等のような）ＦＸＰおよびＦＬＰユニットの時間的な負荷にさらに有利に作用が及ぼされる。

【００１９】

ここでは、プログラムコードにおいてＦＸＰ演算ないしはＦＬＰ演算に翻訳されるようにモデル演算が定められているまたは設けられている制御モデルも考えられる。このような制御モデルでは、この方法の有利な一実施形態によれば、この制御モデルの少なくとも１つのＦＬＰ演算は、プログラムコードの１つのＦＸＰ演算に変換され、および／または、この制御モデルの少なくとも１つのＦＸＰ演算は、プログラムコードの１つのＦＬＰ演算に変換される。この手法によって可能になるのは、プログラムコード内でＦＸＰ演算およびＦＬＰ演算の実質的に任意の比を調整することであり、これにより、制御システムの計算ユニットのＦＸＰユニットおよびＦＬＰユニットの負荷も調整可能である。

【００２０】
別の有利な実施例によれば、変換によって生じる、プログラムコードにおける固定小数点データ型から浮動小数点データ型への、および／または、プログラムコードにおける浮動小数点データ型から固定小数点データ型への型変換の個数が少なくとも局所的に最小になるように、制御モデルのＦＬＰ演算の選択されたＦＬＰ演算をプログラムコードのＦＸＰ演算に変換し、および／または、制御モデルのＦＸＰ演算の選択されたＦＸＰ演算をプログラムコードのＦＬＰ演算に変換する。この点から「局所的な最小」とは，ＦＬＰ演算およびＦＸＰ演算の考えられ得るすべての選択が必ずしも評価されるのではなく、理論的に考えられ得る選択の一部だけが評価されることを意味し、考察される変化形態は、例えばランダムな選択とすることも可能である。

【００２１】
上記の方法の別の実施例では、上記のグラフィック制御モデルから形成されるプログラムコードは、ハードウェア記述言語のプログラムコードであり、このプログラムコードは、少なくとも１つのＦＸＰユニットおよび少なくとも１つのＦＬＰユニットを有する制御システムの計算ユニットを表し、付加的にはこの計算ユニットにより、上記の制御モデルの機能が実現される。すなわち、このプログラムコードは、所定のターゲットハードウェア上だけで動作できるのではなく、これはターゲットハードウェアをも表し得るのである。ここでは、例えば、上記のハードウェア記述言語により、ＦＰＧＡ（Field Programmable Gate Array）を表しかつプログラミングすることができる。すなわち、この場合にこのプログラムコードは、ＦＰＧＡに設けられる機能構造が互いにどのように接続されて協調動作するかを定めるのである。今日のＦＰＧＡの一部には、例えばＦＬＰ演算に特別に使用される固定のＨＷ構造（いわゆるＤＳＰユニット）が含まれているため、ＦＸＰプログラムコードとＦＬＰプログラムコードとを混合する比を考慮する際にこのようなハードウェア素子の負荷／利用度に着目することは有利である。

【００２２】
個別的にみれば、本発明にしたがってコンピュータ上で実現される方法を構成して発展させるのには多くの選択肢がある。これについては請求項１に従属する複数の請求項ならびに図面を参照して行われる本発明の有利な実施例の以下の詳細な説明を参照されたい。

【実施例】

【００２３】
図１には、グラフィック制御モデル４から、制御システム２上で実行可能な制御プログラム３を形成するための方法であって、コンピュータ上で実現される方法１が示されており、この制御システム２はプロセスインタフェース５を有する。制御システム２は、プロセスインタフェース５を介して、物理プロセス６の少なくとも１つのプロセス変量ｘを制御システム２によって検出し、および／または、物理プロセス６に影響を及ぼすための出力量ｙを制御システム２によって送出できるように構成されている。制御システム２は、ＦＸＰユニット（Fixed-Point-Execution Unit）８およびＦＬＰユニット（Floating-Point-Execution Unit）９を備えた電子計算ユニット７を有する。

【００２４】
グラフィック制御モデル４には、図示の実施例において複数のモデル演算１０，１１が含まれている。制御モデル４のモデル演算１０，１１は、データ型が特定されていない。シミュレーションを制御モデル４によって実行しようとする場合（「オフラインシミュレーション」）、これらの計算は一般的に完全に複数のＦＬＰ演算として実行される。しかしながら、このことは、ここで関心の対象となっており、かつ、以下の説明の対象である制御モデル４からのコード生成プロセスとは直接的には関係がない。

【００２５】
従来技術からは図１のように、グラフィック制御モデル４のすべてのモデル演算１０，１１をプログラムコード１３のＦＸＰ演算に翻訳して、生成したプログラムコード１３をほとんど「汎用的に」使用できるようにすることが公知である。なぜならば一般的な制御システム２は、整数ないしは固定小数点数を計算するための少なくとも１つのＦＸＰユニ

ット８を有するが、浮動小数点演算を計算するためのＦＬＰユニット９は必ずしも有する必要がないからである。この手法により、ＦＬＰユニット９によって使用されないため、制御システム２には部分的にしか負荷が加わらないことになる。

【００２６】

これに対し、図２に示した本発明によるコンピュータ上で実現される方法では、上記のグラフィック制御モデル４がプログラムコード１３に翻訳されて、生成したプログラムコード１３が少なくとも１つのＦＸＰ演算１２と、少なくとも１つのＦＬＰ演算１４とを有しており、生成したこのプログラムコード１３が、実行可能な制御プログラム３に翻訳されて、制御システム２上で制御プログラム３を実行する際に制御プログラム３の一部分がＦＸＰユニット８上で実行され、制御プログラム３の別の一部分がＦＬＰユニット９で実行されるようにする。このことは、制御システム２の計算ユニット７の双方のユニット８，９が使用されることを意味する。

【００２７】

制御プログラム３を制御システム２のＦＸＰユニット８にもＦＬＰユニット９にも翻訳することは、図２において制御プログラム３と計算ユニット７との間の矢印によって示されている。生成したプログラム１３が少なくとも１つのＦＸＰ演算１２および少なくとも１つのＦＬＰ演算１４を有するというこのような手法により、制御システム２の計算ユニット７のＦＸＰユニット８もＦＬＰユニット９も共に使用されることが強制され、これによってランタイム時の挙動が、図１に示した実施例に比べて改善される。

【００２８】

図３において一層詳細に再度示されているのは、制御モデル４のいくつかのモデル演算１０，１１が、プログラムコード１３の複数のＦＸＰ演算１２に翻訳され、制御モデル４のいくつかのモデル演算が複数のＦＬＰ演算１４に変換されることである。ここでこの変換はつぎのような条件で行われる。すなわち、時間平均において計算ユニット７のＦＸＰユニット８およびＦＬＰユニット９の負荷が同程度になるように、制御モデル４のモデル演算１０，１１が、プログラムコード１３のＦＸＰ演算１２に変換される個数と、制御モデル４のモデル演算１０，１１が、プログラムコード１３のＦＬＰ演算１４に変換される個数とが決められるのである。

【００２９】

さらに、制御モデル４の変換すべきモデル演算１０，１１を選択する際に考慮されるのは、プログラムコード１３のＦＸＰ演算１２と、プログラムコード１３のＦＬＰ演算１４とが混合されて、制御システム２の計算ユニット７上で制御プログラム３を実行する際に、プロクラムコード１３が計算ユニット７のＦＸＰユニット８およびＦＬＰユニット９上で同時に実行されるようにする。これらの演算を時間的に同時に実行しまた計算ユニット７のユニット８，９が時間的に同時に酷使されることにより、制御プログラム３の最適な実行時間について最大限の効果が得られる。

【００３０】

図３においてさらにわかるのは、上記の手法により、（制御プログラム３と、計算ユニット７のユニット８，９との間に示した）命令ストリームにおいてもＦＬＰ命令およびＦＸＰ命令の混合が得られるため、処理ユニット８，９の供給時にも、これらの２つのユニット８，９に同時に負荷が加えられることが保証されることである。

【００３１】

図４に示した方法のフローチャート１は実質的に、図３に示した方法のフローチャート１をベースにしている。実質的な違いは、上記のコード生成に対し、すでに制御モデル４においていつかのモデル演算がＦＸＰ演算１０として、また別のいつかのモデル演算がＦＬＰ演算１１として決定されているかないしは予定されていることにある。上記の方法では、制御モデル４のＦＸＰ演算１０がプログラムコード１３のＦＸＰ演算１２に、また制御モデル４のＦＬＰ演算１１がプログラムコード１３のＦＬＰ演算１４に単純に翻訳されることはない。むしろここでは、制御モデル４の少なくとも１つのＦＬＰ演算１１がプログラムコード１３のＦＸＰ演算１２に変換され、および／または、制御モデル４において

ＦＸＰ演算１０として設けられている少なくとも１つのモデル演算がプログラムコード１３のＦＬＰ演算１４に変換されるようにする。この際に有利にはつぎのようにする。すなわち、プログラムコード１３において変換によって発生する固定小数点数データ型から浮動小数点数データ型へのデータ型変換、および／または、プログラムコード１３において変換によって発生した浮動小数点データ型から固定小数点データ型へのデータ型変換の個数が少なくとも局所的に最小になるように、制御モデル４のＦＬＰ演算１１として決定された、モデル演算の選定された演算が、プログラムコード１３のＦＸＰ演算１２に変換され、および／または、制御モデル４のＦＸＰ演算１２として決定されたないしは予定された、モデル演算の選定された演算が、プログラムコード１３のＦＬＰ演算１４に変換されるようにするのである。

【００３２】

図５，６および７にはそれぞれ、複数の伝達素子を有するブロック線図の形態で同じグラフィック制御モデル４が示されている。まず２つの入力量から制御偏差ｅが形成される。この制御偏差ｅは、上側の矢印線において係数Ｋｐが乗算され、また下側の矢印線ではフィードバックむだ時間素子（Unit Delay）の後、係数Ｋｉが乗算される。後続の加算素子ｓＵでは２つの矢印線の結果が加算されて出力量Ｕが得られる。

【００３３】

以下では図４を考慮し、制御モデル４をプログラムコードに翻訳した結果を示す。このプログラムコードにはＦＸＰ演算だけが含まれている。この例は、（複数のログラムコードの以下の例も同様に）プログラミング言語Ｃで実現されている。ＦＸＰ演算は整数データ型の変数だけで動作するため、このプログラムコードにはint変数だけが含まれている。種々異なるスケーリングを行った整数ないしは固定小数点数を互いに計算できるようにするため、対応するシフト演算（「＜＜」，「＞＞」）を行わなければならない。それ自体公知の固定小数点数四則演算の詳細はここでは重要ではなく、重要であるのは、ここで示すプログラムコード例が結果的に、図１ないし４のようにＦＸＰユニット８上だけ実行することができるという状況であり、これにより、ハードウェア資源が部分的にしか使用されないことである。

```
int e /* LSB: 2^-10 OFF: 0 MIN/MAX: -32 .. 31.9990234375 */;
int x /* LSB: 2^-6 OFF: 0 MIN/MAX: -512 .. 511.984375 */
static unsigned int Unit_Delay = 0
 /* LSB: 2^-6 OFF: 0 MIN/MAX: 0 .. 1023.984375 */;
e = (int) ((((int) REF) - ((int) Y)) >> 1);
x = (int) (((unsigned int) (int) (e >> 4)) + Unit_Delay);
Unit _Delay = (unsigned int) x;
sU = (int) (((int) ((((int) e) * ((int) P_Kp)) >> 7)) + ((int) ((((int) x) * ((int) P_Ki)) >> 6)));
U = sU;。
```

【００３４】

つぎのコード例は、図５に示した制御モデル４のプログラムコードへの完全な翻訳を示しており、このプログラムコードは、ＦＬＰ演算だけで実現されており、その結果、浮動小数点数データ型の変数だけが含まれている。ここでは浮動小数点数演算が扱われるため、ＦＸＰ演算の場合のような絶え間ないリスケーリングは不要である。制御プログラムに翻訳されるこのようなプログラムコードは、制御システム２の計算ユニット７のＦＬＰユニット上だけで実行されることになるため、ＦＸＰユニット８は利用されないままになる。

```
float e;
float x;
static float Unit_Delay = 0.F;
e = REF - Y;
x = e + Unit_Delay;
```

```
Unit_Delay = x;
sU = (e * P_Kp) + (x * P_Ki)
U = sU;。
```

【００３５】

図６および７に示した実施例においてそれぞれ出発点としているのは、グラフィック制御モデル４ないしはそこに示された複数の演算が、通常のように、上記のプログラムコードにおけるＦＸＰ演算に翻訳されることである。破線でマークした複数の領域はそれぞれ、プログラムコードのＦＬＰ演算に翻訳される制御モデル４の部分を示している。

【００３６】

以下で説明するプログラムコードは、図６に示した制御モデルの翻訳を表している。ここで容易にわかるのは、形成したプログラムコードが、浮動小数点数演算も固定小数点数演算も（ＦＬＰ演算およびＦＸＰ演算）共に有することであり、これらの演算は、ここから形成される制御プログラムが、図１ないし４に示したＦＸＰユニット８上で実行されると共にＦＬＰユニット９上でも実行されることである。

```
float Ki;
float x;
int Kp /* LSB: 2^-6 OFF: 0 MIN/MAX: -512 .. 511.984375 */;
int e /* LSB: 2^-10 OFF: 0 MIN/MAX: -32 .. 31.9990234375 */;
e =(int) ((((int) REF) - ((int) Y)) >> 1);
Kp = (int) ((((int) e) * ((int) P_Kp)) >> 7);
x = (((float) e) * 0.0009765625F) + Unit_Delay;
Ki = x * P_Ki;
sU = (int) (((((float) Kp) * 0.015625F) + Ki) * 64.F);
U = sU;
Unit Delay = x;。
```

【００３７】

以下のプログラムコードは、図７に示した制御モデル４の翻訳を表しており、制御偏差ｅと係数Ｋｐとの乗算だけが浮動小数点数四則演算で実行される。この結果、変数Ｋｐだけが浮動小数点データ型であり、他のすべての変数および計算は整数データ型であり、ないしはＦＸＰ演算である。

```
float Kp;
int Ki /* LSB: 2^-6 OFF: 0 MIN/MAX: -512 .. 511.984375 */;
int e  /* LSB: 2^-10 OFF: 0 MIN/MAX: -32 .. 31.9990234375 */;
int x  /* LSB: 2^-6 OFF: 0 MIN/MAX: -512 .. 511.984375 */;
e = (int) ((((int) REF) - ((int) Y)) >> 1);
Kp =((float) e) * 0.0009765625F * P_Kp;
x = (int) (((unsigned int) (int) (e >> 4)) + Unit _Delay);
Ki = (int) ((((int) x) * ((int) P_Ki)) >> 6);
sU = (int) ((Kp + (((float) Ki) * 0.015625F)) * 64.F);
U = sU;
Unit_Delay = (unsigned int) x;。
```

【００３８】

上記のプログラムコード例を比較することによって明らかになったのは、プログラムコードのＦＸＰ演算ないしはＦＬＰ演算に変換されるグラフィック制御モデル４の部分の選択が、異なる２つの演算タイプについてのプログラムコードの混合に影響を及ぼすことである。一般的な説明部分で述べた判定基準により、このような混合を所期のように行うことができる。

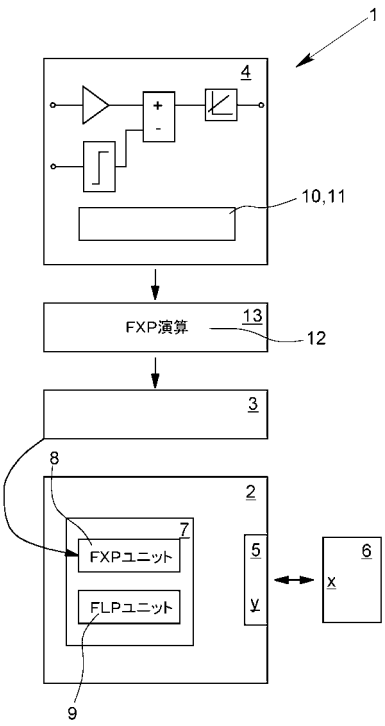
【符号の説明】

【００３９】

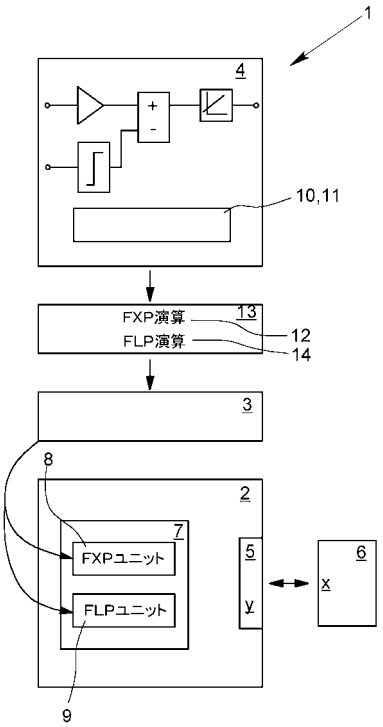
２　制御システム、　３　制御プログラム、　４　グラフィック制御モデル、　５　プ

ロセスインタフェース、　６　物理プロセス、　７　電子計算ユニット、　８　ＦＸＰユニット、　９　ＦＬＰユニット、　１０，１１　モデル演算、　１２　ＦＸＰ演算、　１３　プログラムコード、　１４　ＦＬＰ演算

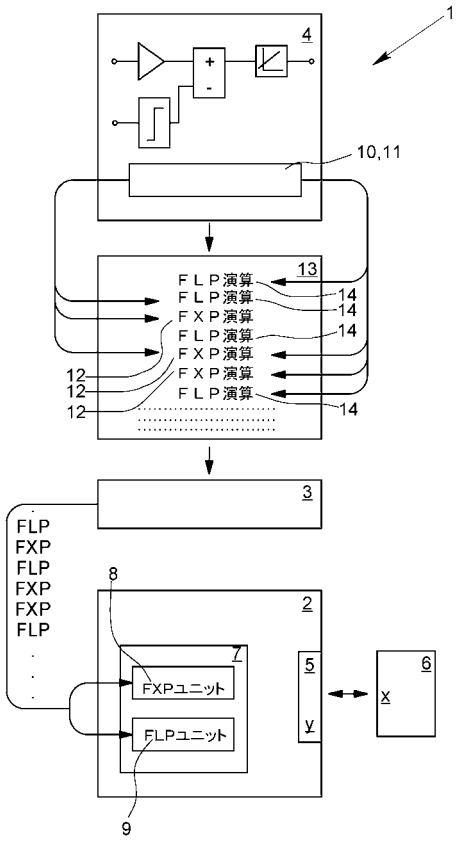
【図１】



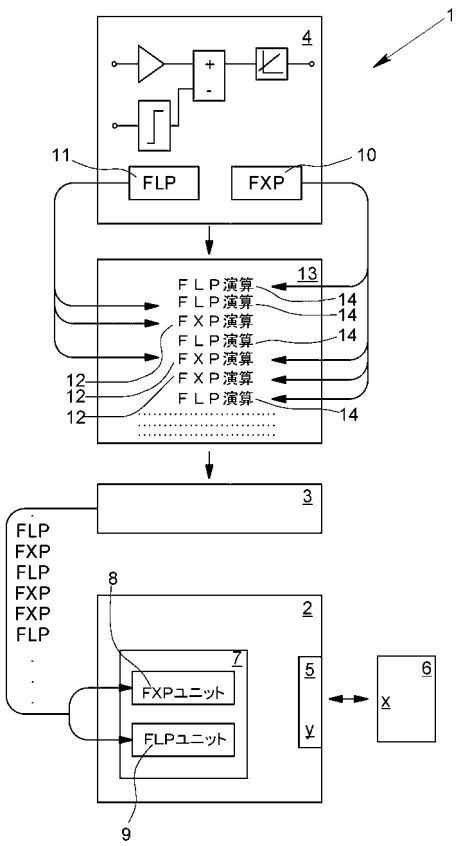
【図２】



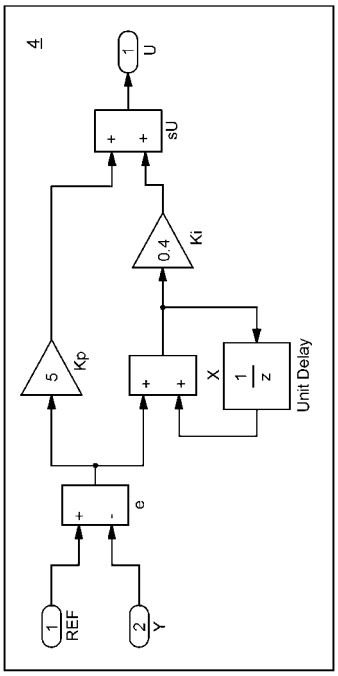
【図３】


1
4
10,11
13
ＦＬＰ演算
ＦＬＰ演算
ＦＸＰ演算
ＦＬＰ演算
ＦＸＰ演算
ＦＸＰ演算
ＦＬＰ演算
14
12
3
FLP
FXP
FLP
FXP
FXP
FLP
8
2
7
FXPユニット
FLPユニット
5
y
6
x
9

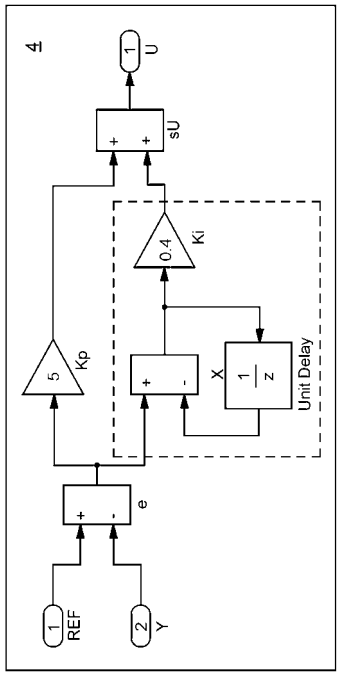
【図４】


1
4
11
FLP
FXP
10
13
ＦＬＰ演算
ＦＬＰ演算
ＦＸＰ演算
ＦＬＰ演算
ＦＸＰ演算
ＦＸＰ演算
ＦＬＰ演算
14
12
3
FLP
FXP
FLP
FXP
FXP
FLP
8
2
7
FXPユニット
FLPユニット
5
y
6
x
9

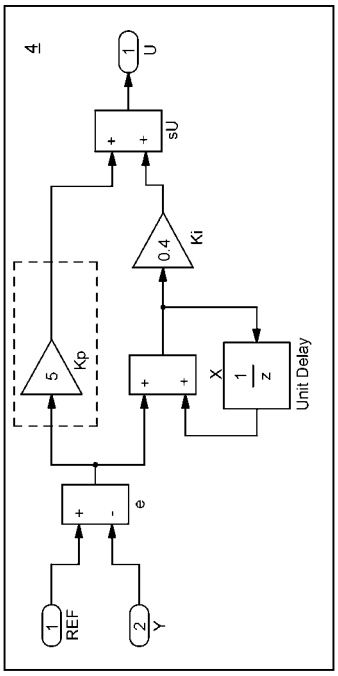
【図５】


4
1
U
sU
0.4
Ki
5
Kp
X
1/z
Unit Delay
e
1
REF
2
Y

【図６】


4
1
U
sU
0.4
Ki
5
Kp
X
1/z
Unit Delay
e
1
REF
2
Y

【図７】



【手続補正書】
【提出日】平成27年3月24日(2015.3.24)
【手続補正２】
【補正対象書類名】特許請求の範囲
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【特許請求の範囲】
【請求項１】
グラフィック制御モデル（４）から、制御システム（２）上で実行可能な制御プログラム（３）を形成するため、コンピュータ上で実現される方法（１）であって、
前記制御システム（２）は、プロセスインタフェースを有しており、
前記制御システム（２）は、前記プロセスインタフェース（５）を介して、物理プロセス（６）の少なくとも１つのプロセス変量を前記制御システム（２）によって検出し、および／または、前記物理プロセス（６）に影響を及ぼすための出力量を前記制御システム（２）によって送出することができるように構成されており、
前記制御システム（２）は、少なくとも１つの固定小数点数演算実行ユニット（ＦＸＰユニット）（８）と、少なくとも１つの浮動小数点数演算実行ユニット（ＦＬＰユニット）（９）とを有する少なくとも１つの電子計算ユニット（７）を有しており、
前記グラフィック制御モデル（４）は複数のモデル演算（１０，１１）を有している、コンピュータ上で実現される方法（１）において、
前記グラフィック制御モデル（４）をプログラムコード（１３）に翻訳して、生成される当該プログラムコード（１３）が少なくとも１つのＦＸＰ演算（１２）と少なくとも１つのＦＬＰ演算（１４）を有するようにし、
生成された前記プログラムコード（１３）を前記実行可能な制御プログラム（３）に翻

訳して、前記制御システム（２）上で前記制御プログラム（３）を実行する際に当該制御プログラム（３）の一部分が前記ＦＸＰユニット（８）上で実行され、当該制御プログラム（３）の別の一部分が前記ＦＬＰユニット（９）で実行されるようにした、
ことを特徴とする方法（１）。
【請求項２】
請求項１に記載の、コンピュータ上で実現される方法（１）において、
時間平均において前記制御システム（２）の前記計算ユニット（７）の前記ＦＸＰユニット（８）および前記ＦＬＰユニット（９）の負荷が同程度になるような個数の前記制御モデル（４）のモデル演算（１０，１１）を、前記プログラムコード（１３）の前記ＦＸＰ演算（１２）および前記ＦＬＰ演算（１４）に翻訳する、
ことを特徴とする方法（１）。
【請求項３】
請求項１または２に記載の、コンピュータ上で実現される方法（１）において、
前記制御システム（２）の前記計算ユニット（７）上で前記制御プログラム（３）を実行する際に、前記プログラムコード（１３）が、前記計算ユニット（７）の前記ＦＸＰユニット（８）および前記ＦＬＰユニット（９）上で同時に実行されるように、前記プログラムコード（１３）のＦＸＰ演算（１２）と、前記プログラムコード（１３）のＦＬＰ演算（１４）とを混合する、
ことを特徴とする方法（１）。
【請求項４】
請求項１から３までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記プログラムコード（１３）のＦＬＰ演算（１４）の個数に対するＦＸＰ演算（１２）の個数の比は、単位時間当たりに前記ＦＸＰユニット（８）上で実行可能なＦＸＰ演算と、前記ＦＬＰユニット（９）上で実行可能なＦＸＰ演算との比に対応する、
ことを特徴とする方法（１）。
【請求項５】
請求項１から４までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記ＦＸＰユニット（８）上で計算するため、前記制御モデル（４）の整数のモデル演算を前記プログラムコード（１３）のＦＸＰ演算（１２）に翻訳する、
ことを特徴とする方法（１）。
【請求項６】
請求項１から５までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記ＦＸＰユニット（８）上で計算するため、ループ、switch-case演算またはビット演算に対するインクリメントまたはデクリメントカウンタを前記プログラムコード（１３）のＦＸＰ演算（１２）に翻訳する、
ことを特徴とする方法（１）。
【請求項７】
請求項１から６までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記プログラムコード（１３）におけるＦＸＰ演算に翻訳する際に、固定小数点数リスケーリングが行われ得る、前記制御モデル（４）のモデル演算（１０，１１）を、前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項８】
請求項１から７までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記計算ユニット（７）がｍビットＦＬＰユニット（９）を有しているがｍビットＦＸ

Ｐユニット（８）を有していない場合、前記プログラムコード（１３）におけるＦＸＰ演算として翻訳するときに、当該プログラムコードにおいて１つまたは複数のｍビットＦＸＰ演算が生じ得る、前記制御モデル（４）のモデル演算（１０）を、その代わりに前記プログラムコード（１３）における１つまたは複数のｍビットＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項９】
請求項１から８までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記制御モデル（４）が複数の並列の矢印線を有する場合、前記制御モデル（４）の複数のモデル演算（１０，１１）の矢印線のうちで、ＦＸＰ演算に翻訳するときに前記プログラムコード（１３）において最も多くのＦＸＰリスケーリングを有し得る矢印線を、前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項１０】
請求項１から９までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記制御モデル（４）における少なくとも１つのモデル演算を、ＦＬＰ演算（１１）としかつ前記プログラムコード（１３）のＦＸＰ演算（１２）に変換し、および／または、
前記制御モデル（４）における少なくとも１つのモデル演算を、ＦＸＰ演算（１０）としかつ前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項１１】
請求項１０に記載の、コンピュータ上で実現される方法（１）において、
変換によって生じる、前記プログラムコード（１３）における固定小数点データ型から浮動小数点データ型への、および／または、前記プログラムコード（１３）における浮動小数点データ型から固定小数点データ型への型変換の個数が少なくとも局所的に最小になるように、前記制御モデル（４）のＦＬＰ演算（１１）として決定され選択されたモデル演算を前記プログラムコード（１３）のＦＸＰ演算（１２）に変換し、および／または、前記制御モデル（４）のＦＸＰ演算（１２）として決定され選択されたモデル演算を前記プログラムコード（１３）のＦＬＰ演算（１４）に変換する、
ことを特徴とする方法（１）。
【請求項１２】
請求項１から１１までのいずれか１項に記載の、コンピュータ上で実現される方法（１）において、
前記グラフィック制御モデル（４）から形成したプログラムコード（１３）は、ハードウェア記述言語のプログラムコードであり、
前記プログラムコード（１３）は、少なくとも１つのＦＸＰユニット（８）および少なくとも１つのＦＬＰユニット（９）を有する制御システム（２）の前記計算ユニット（７）を表し、
当該計算ユニット（７）により、前記制御モデル（４）の機能が実現される、
ことを特徴とする方法（１）。
【請求項１３】
請求項１２に記載の、コンピュータ上で実現される方法（１）において、
前記ハードウェア記述言語は、ＦＰＧＡを表しかつこれをプログラムする、
ことを特徴とする方法（１）。
【請求項１４】
コンピュータプログラムにおいて、
前記コンピュータプログラムがコンピュータによって実行される場合、前記コンピュータプログラムは、請求項１から１３までのいずれか１項に記載の方法を実施するためのソ

フトウェア手段を有する、
ことを特徴とするコンピュータプログラム。

フロントページの続き

(74)代理人 100114890
弁理士 アインゼル・フェリックス＝ラインハルト
(74)代理人 100099483
弁理士 久野 琢也
(72)発明者 ゼバスティアン ヒレブラント
ドイツ連邦共和国 マースベルク フェルスベルクシュトラーセ ５２
(72)発明者 カーステン フィッシャー
ドイツ連邦共和国 パデルボルン フォン－フィンケ－ヴェーク ５

Ｆターム(参考) 5B081 CC21 CC32
5H220 AA01 BB03 CC08 CX01 EE14 JJ12 JJ16 JJ48