(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2014-509106
(P2014-509106A)
(43)公表日　平成26年4月10日(2014.4.10)

(51)Int.Cl.		F I			テーマコード（参考）
H03K 19/177	(2006.01)	H03K	19/177		5B013
G06F 9/38	(2006.01)	G06F	9/38	370C	5B376
G06F 11/00	(2006.01)	G06F	9/06	630A	5J042

審査請求 未請求　予備審査請求 有　（全 39 頁）

(21)出願番号　特願2013-549414 (P2013-549414)
(86)(22)出願日　平成23年12月8日 (2011.12.8)
(85)翻訳文提出日　平成25年8月27日 (2013.8.27)
(86)国際出願番号　PCT/US2011/064038
(87)国際公開番号　WO2012/096735
(87)国際公開日　平成24年7月19日 (2012.7.19)
(31)優先権主張番号　13/005,962
(32)優先日　平成23年1月13日 (2011.1.13)
(33)優先権主張国　米国 (US)

(71)出願人　591025439
ザイリンクス　インコーポレイテッド
ＸＩＬＩＮＸ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国　カリフォルニア州　９５１２４－３４００　サン　ホセ　ロジック　ドライブ　２１００
(74)代理人　110001195
特許業務法人深見特許事務所
(72)発明者　テイラー，ブラッドリー・エル
アメリカ合衆国、９５１２４　カリフォルニア州、サン・ホセ、ロジック・ドライブ、２１００

最終頁に続く

(54)【発明の名称】集積回路におけるプロセッサシステムの拡張

(57)【要約】
集積回路（２００，３００、４００）の実施例が述べられる。集積回路は、プログラムコード（３１５，４１５）を実行するように構成されるプロセッサシステム（２０２，３０２，４０２）と、集積回路のプログラマブル回路（２０４，３０４，４０４）内に実現されるプロセス特定回路（２８２，３２０，４４０，４４５）とを備える。プロセス特定回路はプロセッサシステムに結合され、プロセッサシステムによってオフロードされるプロセスを実行するように構成される。プロセッサシステムは、プロセスを実行するプログラムコードを実行する代わりにプロセス特定回路にプロセスをオフロードするように構成される。

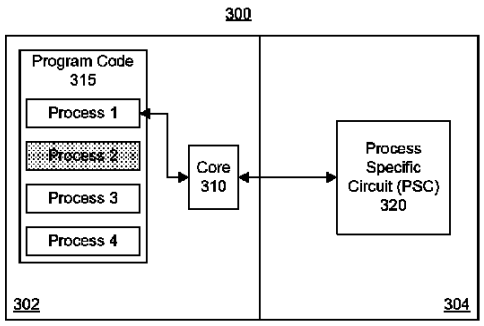

FIG. 3

【特許請求の範囲】
【請求項１】
集積回路であって、
プログラムコードを実行するように構成されるプロセッサシステムと、
前記集積回路のプログラマブル回路内に実現されるプロセス特定回路とを備え、前記プロセス特定回路は、前記プロセッサシステムに結合されかつ前記プロセッサシステムによってオフロードされるプロセスを実行するように構成され、
前記プロセッサシステムは、前記プロセスを実行するためのプログラムコードの実行に代えて前記プロセス特定回路にプロセスをオフロードするように構成される、集積回路。
【請求項２】
前記プロセッサシステムは、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される電力消費の低減に従って、前記プロセス特定回路に前記プロセスをオフロードするか否か決定するようにさらに構成される、請求項１に記載の集積回路。
【請求項３】
前記プロセッサシステムは、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善に従って、前記プロセス特定回路に前記プロセスをオフロードするか否か決定するようにさらに構成される、請求項１に記載の集積回路。
【請求項４】
前記プロセッサシステムは、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される時間遅れの低減に従って、プロセス特定回路に前記プロセスをオフロードするか否か決定するようにさらに構成される、請求項１に記載の集積回路。
【請求項５】
前記プログラマブル回路はプログラマブルファブリックであり、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善は、前記プロセス特定回路を実現するために前記プログラマブルファブリックの少なくとも一部を動的に再構成するために必要とされる時間の測定をさらに備える、請求項３に記載の集積回路。
【請求項６】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善は、複数のメモリのどのメモリが前記プロセス特定回路に対する前記プロセスに必要とされるソースデータを提供するために使用されるかに依存する、請求項３に記載の集積回路。
【請求項７】
前記プロセッサシステムは、前記プロセス特定回路を実現するために前記プログラマブルファブリック内の利用可能なスペースの量が存在するか否か決定するようにさらに構成される、請求項１から請求項６のいずれか１項に記載の集積回路。
【請求項８】
前記プログラマブル回路はプログラマブルファブリックであり、前記プロセッサシステムは、前記プロセス特定回路を実現するための前記プログラマブルファブリックの少なくとも一部の動的再構成を開始するように構成される、請求項１から請求項７のいずれか１項に記載の集積回路。
【請求項９】
集積回路におけるプロセッサシステムの拡張の方法であって、
前記集積回路内に実現される前記プロセッサシステム内のプログラムコードを実行するステップを備え、前記集積回路はプログラマブル回路を含み、前記プロセッサシステムは前記プログラマブル回路に結合され、
前記方法は、前記プロセッサシステムでのプロセスを実行するためのプログラムコード

の実行に代えて前記プログラマブル回路内に実現されるプロセス特定回路を使用するプロセスを実行するステップと、
前記プロセッサシステムに対して利用可能な前記プロセス特定回路から前記プロセスの結果を作成するステップとをさらに備える、方法。
【請求項１０】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用するプロセスの実現を通して達成される電力消費の低減に従って、前記プロセス特定回路を使用する前記プロセスを実行するか否か決定するステップをさらに備える、請求項９に記載の方法。
【請求項１１】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される時間遅れの低減に従って、前記プロセス特定回路を使用する前記プロセスを実行するか否か決定するステップをさらに備える、請求項９に記載の方法。
【請求項１２】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の低減に従って、前記プロセス特定回路を使用する前記プロセスを実行するか否か決定するステップをさらに備える、請求項９に記載の方法。
【請求項１３】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の低減は、前記プロセス特定回路を実現するための前記プログラマブル回路の少なくとも一部を動的に再構成するために必要とされる時間の測定をさらに含む、請求項１２に記載の方法。
【請求項１４】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善の測定は、前記プロセス特定回路による前記プロセスの実行のために必要とされるソースデータを提供するために複数のメモリのどのメモリが用いられるかに依存する、請求項１２に記載の方法。
【請求項１５】
前記プログラムコードによって特定される複数のプロセスから前記プロセスを選択するステップと、
ハードウェアにおいて前記選択されるプロセスを実現する前記プロセス特定回路を特定する設定データを選択するステップと、
前記集積回路の前記プログラマブル回路内に前記プロセス特定回路を実現するために選択された設定データをロードするステップとをさらに備える、請求項９から請求項１４のいずれか１項に記載の方法。
【発明の詳細な説明】
【技術分野】
【０００１】
この明細書中に開示される１つ以上の実施例は、集積回路（ＩＣ）に関する。より特定的には、１つ以上の実施例は、ＩＣ内で実現されるプロセッサシステムの拡張に関する。
【背景技術】
【０００２】
集積回路（ＩＣｓ）は、特定の機能を実行するために提供される。１つのタイプのＩＣは、たとえば、フィールドプログラマブルゲートアレイ（ＦＰＧＡ：field programmable gate array）のようなプログラマブルＩＣである。ＦＰＧＡは、典型的には、プログラマブルタイルの配列を含む。これらのプログラマブルタイルは、たとえば、入力／出力ブロック（ＩＯＢ）、論理ブロック配列（ＣＬＢ）、専用ランダムアクセスメモリブロック（ＢＲＡＭ）、乗算器、ディジタル信号処理ブロック（ＤＳＰ）、プロセッサ、クロックマネージャ、遅延ロックループ（ＤＬＬ）などを含み得る。

【０００３】
プログラマブルタイルの各々は、典型的には、プログラマブル相互接続回路と、プログラマブル論理回路とを含む。プログラマブル相互接続回路は、典型的には、プログラマブル相互接続ポイント（ＰＩＰ）によって相互接続される可変長の多数の相互接続ラインを含む。プログラマブル論理回路は、たとえば、関数生成器、レジスタ、論理演算などを含み得るプログラマブル素子を使用するユーザ設計の論理を提供する。
【０００４】
プログラマブル相互接続回路とプログラマブル論理回路とは、典型的には、プログラマブル要素がどのように設定されるかを定める内部設定メモリセル内に設定データのストリームをローディングすることによりプログラムされる。設定データは、メモリから（たとえば外部のＰＲＯＭから）読み出され、または、外部デバイスによってＦＰＧＡ内に書き込まれ得る。そして、個々のメモリセルの集合状態は、ＦＰＧＡの機能を決定する。
【０００５】
別のタイプのプログラマブルＩＣは、複合プログラマブル論理デバイス、つまりＣＰＬＤである。ＣＰＬＤは、相互接続スイッチマトリクスによって共にかつ入力／出力（Ｉ／Ｏ）リソースに接続された、２つ以上の「機能ブロック」を含む。ＣＰＬＤの機能ブロックの各々は、プログラマブル論理アレイ（ＰＬＡ）およびプログラマブルアレイ論理（ＰＡＬ）デバイスで使用されるものに類似した、２レベルのＡＮＤ／ＯＲ構造を含む。ＣＰＬＤにおいて、設定データは、典型的には、オンチップ不揮発性メモリに記憶される。幾つかのＣＰＬＤにおいて、設定データは、オンチップ不揮発性メモリに記憶され、そして、初期設定（プログラミング）シーケンスの一部として、不揮発性メモリにダウンロードされる。
【０００６】
これらの全てのプログラマブルＩＣについて、デバイスの機能性は、デバイスに供給されるデータビットによってその目的のために制御される。データビットは、揮発性メモリ（ＦＰＧＡおよび幾つかのＣＰＬＤのように、たとえばスタティックメモリセル）、不揮発性メモリ（幾つかのＣＰＬＤのように、たとえばフラッシュメモリ）、または他のタイプのメモリセルに記憶され得る。
【０００７】
他のプログラマブルＩＣは、デバイスのさまざまな素子をプログラム可能に相互接続する、メタル層のような処理層を適用することによってプログラムされる。これらのプログラマブルＩＣは、マスクプログラマブルデバイスとして知られる。プログラマブルＩＣは、たとえばヒューズあるいはアンチヒューズ技術を使用する他の方法によっても実現され得る。「プログラマブルＩＣ」は、これらのデバイスを含み得る、ただし、これらのデバイスに限定されず、たとえば特定用途向け集積回路（ＡＳＩＣ）を含む部分的にのみプログラム可能なデバイスを包含し得る。たとえば、プログラマブルＩＣの別のタイプは、ハードコーディングされたトランジスタ論理とプログラム可能にハードコーディングされたトランジスタ論理を相互接続するプログラマブルスイッチファブリックとの組み合わせを含む。
【発明の概要】
【発明が解決しようとする課題】
【０００８】
上述の多様なＩＣを幾つか含む近代の幾つかのＩＣは、プログラムコードを実行できる組み込みプロセッサを含む。プロセッサは、まとめてＩＣの「プログラマブル回路」として称され、プログラマブル論理回路とプログラマブル相互接続回路とを含む同じダイの一部として製造される。プロセッサ内のプログラムコードの実行は、ＩＣにおけるプログラマブル回路の「プログラミング」または「設定」から区別されると理解されるべきである。ＩＣのプログラマブルファブリックのプログラミングまたは設定の行為は、プログラマブルファブリック内の設定データによって特定される異なる物理回路の実現をもたらす。
【課題を解決するための手段】

【０００９】
この明細書中の１つ以上の実施例は、集積回路（ＩＣ）に関し、より特定的には、ＩＣ内で実現されるプロセッサシステムの拡張に関する。
【００１０】
集積回路の実施例は、プログラムコードを実行するように設定されるプロセッサシステムと、集積回路のプログラマブル回路内に実現されるプロセス特定回路とを含み得る。プロセス特定回路は、プロセッサシステムに結合され、プロセッサシステムによりオフロードされるプロセスを実行するように構成され得る。プロセッサシステムは、プロセスを実行するためのプログラムコードを実行する代わりに、プロセス特定回路にプロセスをオフロードするように構成され得る。
【００１１】
この実施例の１つの局面において、プロセッサシステムは、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される電力消費の低減に従って、プロセス特定回路にプロセスをオフロードするか否か決定するようにさらに構成され得る。
【００１２】
この実施例の別の局面において、プロセッサシステムは、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される完了時間における改善に従って、プロセス特定回路にプロセスをオフロードするか否か決定するようにさらに構成され得る。
【００１３】
さらにこの実施例の別の局面において、プロセッサシステムは、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセス特定回路を使用するプロセスの実現を通して達成される時間遅れの低減に従って、プロセス特定回路にプロセスをオフロードするか否か決定するようにさらに構成され得る。
【００１４】
この実施例の別の局面において、プログラマブル回路は、プログラマブル回路であり得る。プロセッサシステムを使用する代わりにプロセス特定回路を使用するプロセスの実現を通して達成される完了時間の改善は、プロセス特定回路を実現するために、少なくともプログラマブル回路の一部を動的に再構成するのに必要とされる時間の測定をさらに含む。
【００１５】
さらにこの実施例の別の局面において、プロセッサシステムを使用する代わりにプロセス特定回路を使用するプロセスの実現を通して達成される完了時間の改善は、複数のメモリのどのメモリがプロセス特定回路に対するプロセスに必要とされるソースデータを提供するために使用され得るかに依存する。
【００１６】
この実施例の別の局面において、プロセッサシステムは、プロセス特定回路を実現するためのプログラマブルファブリック内の利用可能なスペースの量が存在するか否か決定するようにさらに構成され得る。
【００１７】
さらにこの実施例の別の局面において、プログラマブル回路は、プログラマブルファブリックであり得る。プロセッサシステムは、プロセス特定回路を実現するためのプログラマブルファブリックの少なくとも一部の動的再構成を開始するようにさらに構成され得る。
【００１８】
集積回路内のプロセッサシステムの拡張の方法の実施例は、集積回路内に実現されるプロセッサシステム内のプログラムコードを実行するステップを含み得る。集積回路は、プログラマブル回路を含み得る。プロセッサシステムは、プログラマブル回路に結合され得る。方法は、プロセッサシステムのプロセスを実行するためのプログラムコードの実行に

代えてプログラマブル回路内で実現されるプロセス特定回路を使用するプロセスを実行するステップを含み得る。方法は、プロセッサシステムに対して利用可能なプロセス特定回路からのプロセスの結果を作成するステップを含み得る。

【００１９】

この実施例の別の局面において、方法は、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される電力消費の低減に対応して、プロセス特定回路を使用するプロセスを実行するか否か決定するステップをさらに含み得る。

【００２０】

この実施例の別の局面において、方法は、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される時間遅れの低減に従って、プロセス特定回路を使用するプロセスを実行するか否か決定するステップをさらに含み得る。

【００２１】

さらにこの実施例の別の局面において、方法は、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される完了時間における低減に従って、プロセス特定回路を使用するプロセスを実行するか否か決定するステップをさらに含み得る。

【００２２】

この実施例の別の局面において、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される完了時間の低減は、プロセス特定回路を実現するためのプログラマブル回路の少なくとも一部を動的に再構成するために必要とされる時間の測定をさらに含み得る。

【００２３】

さらにこの実施例の別の局面において、プロセッサシステムを用いる代わりにプロセス特定回路を使用するプロセスの実現を通して達成される完了時間の改善は、プロセス特定回路によるプロセスを実行するのに必要とされるソースデータを提供するために複数のメモリのどのメモリが使用され得るかに依存する。

【００２４】

この実施例の別の局面において、方法は、プロセス特定回路を実現するためのプログラマブルファブリック内の利用可能なスペースの量が存在するか否か決定するステップをさらに含み得る。

【００２５】

この実施例の別の局面において、方法は、プロセス特定回路を実現するために、プログラマブルファブリックの少なくとも一部を動的に再構成するステップをさらに含み得る。

【００２６】

この実施例の別の局面において、方法は、プログラムコードにより特定される複数のプロセスからプロセスを選択するステップをさらに含み得る。方法は、ハードウェアにおいて選択されるプロセスを実現するプロセス特定回路を特定する設定データを選択するステップをさらに含み得る。方法は、集積回路のプログラマブル回路内にプロセス特定回路を実現するために選択された設定データをロードするステップをさらに含む。

【００２７】

集積回路内のプロセッサシステムの拡張の方法の別の実施例は、プロセッサシステム内のプログラムコードを実行するステップをさらに含み得る。プログラムコードは、複数のプロセスを特定する。方法は、プログラムコードにより特定される複数のプロセスからプロセスを選択するステップを含み得る。方法は、ハードウェアにおいて選択されるプロセスを実現するプロセス特定回路を特定する設定データを選択するステップを含み得る。方法は、集積回路のプログラマブルファブリック内にプロセス特定回路を実現するために選択されるデータをローディングするステップと、プロセッサシステムの代わりにプロセス特定回路を使用する選択されるプロセスを実行するステップとを含み得る。

【００２８】

この実施例の別の局面において、方法は、プロセス特定回路によりアクセス可能なプロセッサシステム内のメモリ内のプロセスディスクリプタを記憶するプロセッサシステムをさらに含み得る。プロセスディスクリプタは、選択されるプロセスのソースデータを特定する。

【００２９】

この実施例の別の局面において、方法は、プロセッサシステム内のメモリからのプロセスディスクリプタにアクセスすることによりソースデータを決定するプロセス特定回路をさらに含み得る。

【００３０】

さらにこの実施例の別の局面において、方法は、プロセッサシステムに選択されるプロセスの結果を与えるプロセス特定回路をさらに含み得る。

【図面の簡単な説明】

【００３１】

【図１】この明細書中に開示される実施例に従う集積回路のアーキテクチャを説明する第１ブロック図である。

【図２】この明細書中に開示される別の実施例に従うＩＣを説明するための第２ブロック図である。

【図３】この明細書中に開示される別の実施例に従うプロセッサシステムの拡張のために構成されるＩＣを説明するための第３ブロック図である。

【図４】この明細書中に開示される別の実施例に従うプロセッサシステムの拡張のために構成されるＩＣを説明するための第４ブロック図である。

【図５】この明細書中に開示される別の実施例に従うＩＣ内のプロセッサシステムの拡張の方法を説明するための第１フローチャートである。

【図６】この明細書中に開示される別の実施例に従うＩＣのプログラマブルファブリック内に実現される回路へのプロセスのオフローディングの方法を説明するための第２フローチャートである。

【図７】この明細書中に開示される別の実施例に従うＩＣのプログラマブルファブリック内に実現される回路へのプロセスのオフローディングの方法を説明するための第３フローチャートである。

【図８】この明細書中に開示される別の実施例に従うＩＣのプログラマブルファブリック内に実現される回路へのプロセスのオフローディングの方法を説明するための第４フローチャートである。

【発明を実施するための形態】

【００３２】

この明細書は新規とみなされる１つ以上の実施例の特徴を定める請求項で終えるが、１つ以上の実施例は、図面と併せた記載を考慮することによってさらによく理解されると思われる。要求されるように、１つ以上の実施例は、この明細書中に開示される。しかし、１つ以上の実施例は、単に例示であると理解されるべきである。そのため、この明細書中に開示される特定の構造および機能的な詳細は、限定解釈されず、単に請求項の根拠として、かつ、実質的に任意の適切な詳細構成における１つ以上の実施例をさまざまに採用するように当業者に教示するための代表的な根拠として解釈される。さらに、ここで使用される用語および語句は、限定することでなく、ここに開示される１つ以上の実施例の理解可能な説明を与えることを意図する。

【００３３】

この明細書中に開示される１つ以上の実施例は、集積回路に関し、より特定的には、ＩＣ内で実現されるプロセッサシステムの拡張に関する。ＩＣは、プログラマブルファブリックの一部に結合されるプロセッサシステムを含むように実現され得る。プロセッサシステムの機能性は、プロセッサシステムと連動して、たとえば相補的に、プログラマブルファブリックの能力を活用することによって拡張され得る。プロセッサシステムによって実行されるあるいはされ得る１つ以上のプロセスは、たとえば、プログラムコードの実行を

通して、プログラマブルファブリック内で実現される回路、たとえばハードウェアにオフロードされ得る。場合によって、回路においてソフトウェアプロセスを実現することは、プロセッサシステム内のプログラムコードの実行を通したプロセスを実現することよりも、完了のために少ない時間を必要とし、少ない時間遅れを有し、および／または少ないエネルギーを使用する。

【００３４】

したがって、１つ以上のプロセスが、選択され、かつプログラマブルファブリック内で実現される回路にオフロードされ得る。回路は、特に、オフロードプロセスを実現するために設計される。たとえば、プロセッサシステムによって実行されるプログラムコードにおいて特定される特定のアルゴリズムまたは手順は、プログラマブルファブリック内のハードウェアにおいて実現され得る。プロセッサシステムは、回路、たとえばプロセス特定回路にプロセスを実行するように指示する。そのため、プロセスは、プロセッサシステム内のプログラムコードの実行を介してよりも、プログラマブルファブリック内のプロセス特定回路によって実行される。

【００３５】

オフロードされるべきプロセスの選択と、いつプロセスをオフロードするかについての決定とは、さまざまな異なる技術を使用してプロセッサシステムによって実行され得る。たとえば、インジケータは、これらのプロセスがオフロードされることを特定するプロセッサシステムによって実行されるプログラムコード内に配置され得る。別の局面において、プロセッサシステムは、プロセス選択および選択されるプロセスがいつプログラマブルファブリックにオフロードされるかに関する、より動的かつ知的な決定をする。

【００３６】

図１は、この明細書中に開示される実施例に従うＩＣのアーキテクチャ１００を説明する第１ブロックダイアグラムである。アーキテクチャ１００は、フィールドプログラマブルゲートアレイ（ＦＰＧＡ：field programmable gate array）タイプのＩＣ内に実現され得る。図示されるように、アーキテクチャ１００は複数の異なるタイプのプログラマブル回路、たとえば論理、のブロックを含む。たとえば、アーキテクチャ１００は、マルチギガビットトランシーバ（ＭＧＴ）１０１と、設定論理ブロック（ＣＬＢ)１０２と、ランダムアクセスメモリブロック（ＢＲＡＭ）１０３と、入力／出力ブロック（ＩＯＢ）１０４と、設定およびクロッキング論理（ＣＯＮＦＩＧ／ＣＬＯＣＫ）１０５と、ディジタル信号処理ブロック（ＤＳＰ)１０６と、専用入力／出力ブロック（Ｉ／Ｏ）１０７（たとえば設定ポートおよびクロックポート）と、ディジタルクロックマネージャ、アナログ－ディジタル変換器、およびシステムモニタリング論理などの他のプログラマブル論理１０８とを含む複数の異なるプログラマブルタイルを含む。

【００３７】

幾つかのプログラマブルＩＣにおいて、プログラマブルタイルの各々は、プログラマブル相互接続素子（ＩＮＴ)１１１を含む。プログラマブル相互接続素子（ＩＮＴ)１１１は、隣接の各々のタイルの対応するＩＮＴ１１１へ／からの、標準化された接続を有する。そのため、ＩＮＴ１１１は、ともに、図示されるＩＣのプログラマブル相互接続構造を実現する。各ＩＮＴ１１１は、図１の上部に含まれる例に示されるように、同じタイル内のプログラマブル論理素子へ／からの、接続も含む。

【００３８】

たとえば、ＣＬＢ１０２は、ユーザ論理に加えて単一のＩＮＴ１１１を実現するようにプログラムされ得るコンフィギュラブル論理素子（ＣＬＥ）１１２を含み得る。ＢＲＡＭ１０３は、１つ以上のＩＮＴ１１１に加え、ＢＲＡＭ論理素子（ＢＲＬ）１１３を含み得る。典型的には、タイル内に含まれるＩＮＴ１１１の数は、タイルの高さに依存する。図示される実施例において、ＢＲＡＭタイルは、ＣＬＶの５倍の高さを有するが、他の数（たとえば４）が使用されてもよい。ＤＳＰタイル１０６は、適切な数のＩＮＴ１１１に加え、ＤＳＰ論理素子（ＤＳＰＬ）１１４を含み得る。ＩＯＢ１０４は、たとえば、１つのＩＮＴ１１１のインスタンスに２つの入力／出力論理素子（ＩＯＬ）１１５を加えたもの

を含み得る。当業者に明確なように、実際にたとえばＩＯＬ１１５へ接続されるＩ／Ｏパッドは、典型的には、ＩＯＬ１１５に限られない。

【００３９】

図１に示される例において、ダイ中央付近のコラム状の領域（図１のハッチングで示される）は、設定と、クロックと、他の制御論理とに使用される。このコラムから延びる水平領域１０９は、プログラマブルＩＣの幅を横切るクロックおよび設定信号を分配する。

【００４０】

図１に示されるアーキテクチャを利用する幾つかのプログラマブルＩＣは、プログラマブルＩＣの大部分を構成する規則的なコラム状の構造を中断させる追加の論理ブロックを含み得る。追加の論理ブロックは、プログラマブルブロックおよび／または専用回路とすることができる。たとえば、ＰＲＯＣ１１０として示されるプロセッサブロックは、ＣＬＢとＢＲＡＭとの複数のコラムにおよぶ。

【００４１】

ＰＲＯＣ１１０は、ＩＣのプログラマブルファブリックを実現するダイの一部として製造され、ハードウェアにより実現されるプロセッサとして実現され得る。ＰＲＯＣ１１０は、ことなるさまざまなプロセッサタイプ、および／または、個別のプロセッサ、たとえばプログラムコードを実行可能なシングルコア、から、１つ以上の、コア、モジュール、コプロセッサ、インタフェイス、あるいは同様のものを有するプロセッサシステム全体のような複雑なものに至るまでを表わし得る。

【００４２】

より複雑な配列において、たとえば中央処理ユニット、キャッシュメモリ、メモリコントローラ、プログラマブルＩＣのＩ／Ｏピンに直接結合するおよび／またはプログラマブルＩＣのプログラマブル回路に結合するように構成可能な一方向および／または双方向インタフェイスなどの１つ以上のコアを含み得る。「プログラマブル回路」は、ここで記載されるさまざまなプログラム可能なまたは設定可能な回路ブロックまたはタイルを参照し得る。ＩＣにロードされる設定データに従って、さまざまな回路ブロック、タイル、および／または素子を選択的に結合する相互接続回路についても同様である。

【００４３】

ＰＲＯＣ１１０内で利用可能なさまざまなインタフェイスを用いることで、ＰＲＯＣ１１０によって実行されるプログラマブルコードにより定められるプロセスは、プログラマブルファブリック内に実現される回路にオフロードされ得る。述べられたようなアーキテクチャを有するＩＣ内にロードされる設定データは、たとえば、プログラマブルファブリック内の１つ以上のプロセス特定回路（ＰＳＣ）を実現し得る。ＰＲＯＣ１１０は、ハードウェアの実施、すなわちＰＳＣまたはＰＳＣを特定する設定データ、に関連づけられる１つ以上のプロセスを選択し、実現のためのプロセスをプログラマブルファブリック内にオフロードし得る。

【００４４】

図１は、プログラマブルＩＣを実現するのに使用されるアーキテクチャの例を単に説明することが意図されたものである。たとえば、コラム内の論理ブロックの数と、コラムの相対幅と、コラムの順序と、コラムに含まれる論理ブロックのタイプと、論理ブロックの相対サイズと、図１の上部に含まれる相互接続／論理の実行例は、単に例示にすぎない。実際のＩＣにおいて、たとえば、ユーザ回路デザインの効率的な実現を促進するために、ＣＬＢの１より多い隣接コラムは、典型的には、ＣＬＢが現れるところのどこにでも含まれる。しかし、隣接ＣＬＢコラムの数は、ＩＣ全体のサイズで変わり得る。

【００４５】

図２は、この明細書中に開示される別の実施例に従って構成されるＩＣ２００を説明する第２ブロックダイアグラムである。ＩＣ２００は、たとえばプログラマブルファブリック２０４のようなプログラマブル回路に結合されるプロセッサシステム（ＰＳ）２０２を含む任意のさまざまな異なる構成を使用して実現することができる。たとえば、ＩＣ２００は、図１のアーキテクチャ１００と同じあるいは類似したアーキテクチャを使用して実

現され得るが、その場合に限られない。一般に、ＩＣ２００は、ＰＳ２０２からハードウェアへのソフトウェアベースプロセスのオフロードを促進するために、ＰＳ２０２をプログラマブルファブリック２０４内に実現される回路に結合するために使用されるさまざまなインタフェイスをより詳細に説明する。

【００４６】

図２に示される例において、ＰＳ２０２は、ＩＣ２００のダイのおよそ三分の二を占めるように図示され、一方でプログラマブルファブリック２０４は、同じダイのおよそ三分の一を占めると説明される。図２は、しかし、ＩＣ２００の代表的なスケールを意図しているのではない。むしろ、図２は、例示を目的として与えられ、この明細書内に開示される１つ以上の実施例の限定を意図していない。

【００４７】

一般に、ＰＳ２０２は、ＩＣ２００内にハードウェアにより構成されるシステムとして実現される。ＰＳ２０２内のさまざまな部品やモジュールは、ある程度、矢印を有するライン、たとえば信号または通信のリンクなど、によって結合される。そのような矢印は、制御のフローの方向を説明することを意図する。この点で、方向矢印を有するラインとして図示される信号は、一般に、信号の制御が、ターゲット部品よりも矢印を発するソース部品によってもたらされることを示している。矢印は、一般に、データまたは信号の方向を示すことを意図するものではない。この点で、信号は、方向矢印の存在にかかわらず、双方向信号または通信リンクとして実現され得る。

【００４８】

この明細書中において、同じ参照符号は、端子と、信号ラインと、ワイヤと、それらの一致する信号とを参照するために使用される。この点で、「信号」と、「ワイヤ」と、「接続」と、「端子」と、「ピン」との用語は、随時、この明細書中において、同義に使用されてもよい。「信号」や「ワイヤ」などは、シングルワイヤを介するシングルビットの伝達または複数の並列ワイヤを介する複数の並列ビットの伝達などの１つ以上の信号を表わし得ると認められるべきである。さらに、ワイヤまたは信号の各々は、別掲のとおり、信号またはワイヤによって接続された２つ以上の要素間の双方向通信を表わす場合もあってよい。

【００４９】

示されるように、ＰＳ２０２は、コア複合体２０６を含む。コア複合体２０６は、コア２０８および２１０と、ＤＳＰユニット２１２および２１４と、割り込み要求ユニット（ＩＲＱ）２１６と、スヌープ制御ユニット（ＳＣＵ）２１８とを含む。コア２０８および２１０の各々は、そこに組み込まれたレベル１（Ｌ１）キャッシュ（図示しない）を含み得る。任意のさまざまな異なるタイプのプロセッサコアおよび／またはＤＳＰユニットが使用され得るが、ここに開示される実施例において、コア２０８および２１０の各々は、３２ＫＢの命令キャッシュと３２ＫＢのデータキャッシュとを有するＡＲＭ Ｃｏｒｔｅｘ（登録商標）－Ａ９タイプのプロセッサコアとして実現され得る。ＤＳＰユニット２１２および２１４は、ＮＥＯＮ（登録商標）メディアおよび／または浮動小数点処理エンジンの形で実現され得る。ＤＳＰユニット２１２および２１４の各々は、１２８ビットのベクトルベースのＤＳＰ機能を与え得る。ＡＲＭ Ｃｏｒｔｅｘ（登録商標）－Ａ９プロセッサおよびＮＥＯＮ（登録商標）メディアおよび／または浮動小数点処理エンジンは、英国ケンブリッジのＡＲＭホールディングス（ＡＲＭ）から利用可能である。

【００５０】

ＰＳ２０２内において、コア複合体２０６は、レベル２（Ｌ２）キャッシュ２２０とオンチップメモリ（ＯＣＭ）２２２とに結合される。Ｌ２キャッシュ２２０は、２５６ＫＢメモリとして実現され得る。ＯＣＭ２２２も、２５６ＫＢメモリとして実現され得る。コア２０８および２１０とＤＳＰユニット２１２および２１４とは、Ｌ２キャッシュ２２０およびＯＣＭ２２２に直接アクセスし得る。一般に、ＯＣＭ２２２は、ＰＳ２０２および／またはプログラマブルファブリック２０４たとえばプログラマブルファブリック２０４内に実現される回路、に対して利用可能なローカルメモリを実現する。比較により、メモ

リでもあるＬ２キャッシュ２２０は、ＰＳ２０２のキャッシュとして機能する。したがって、Ｌ２キャッシュ２２０は、オフチップ実行メモリなどのＲＡＭ内に記憶されたデータビットを効率的なコピーであるデータの小さいブロックまたは一部を記憶し得る。もし、たとえば、Ｌ２キャッシュ２２０内に記憶されたデータの読み出し要求が発行されると、ＲＡＭから取り戻されるのと反対に、データはＬ２キャッシュ２２０から読まれる。

【００５１】

ＰＳ２０２はさらにリセットユニット２２４と、クロックユニット２２６と、メモリコントローラ２２８とを含む。リセットユニット２２４は、信号２３０などのＩＣ２００の外部のソースを起源とする１つ以上の信号を受ける。信号２３０は、ＰＳ２０２および／またはＰＳ２０２内の１つ以上の全ての部品をリセットするように、リセットユニット２２４に指示する。クロックユニット２２６は、信号２３２などのＩＣ２００の外部のソースを起源とする１つ以上の基準信号を受け得る。クロックユニット２２６は、たとえば、受信された信号２３２と同期できる位相同期回路として実現され得、あるいはそれを含み得る。クロックユニット２２６は、１つ以上の異なる周波数のクロック信号を生成し得、そのクロック信号はＰＳ２０２のいたるところに分配される（図示しない）。さらに、クロックユニット２２６は、１つ以上の異なる周波数の１つ以上のクロック信号を生成し得、そのクロック信号はプログラマブルファブリック２０４内に実現される回路による使用のためにプログラマブルファブリック２０４に分配され得る。

【００５２】

メモリコントローラ２２８は、「オフチップ」などのＩＣ２００の外部に配置される１つ以上の異なるタイプのＲＡＭと通信するように実現され得る。たとえば、メモリコントローラ２２８は限定されないが、デュアルデータレート（ＤＤＲ)２、ＤＤＲ３、ロー電力（ＬＰ）ＤＤＲ２タイプのメモリ、１６ビット、３２ビット、ＥＣＣを有する１６ビットなどを含む、さまざまなタイプのメモリにアクセス、たとえば読みおよび／または書きなど、するように実現され得る。メモリコントローラ２２８が通信可能である異なるメモリタイプのリストは、説明のためのみ与えられ、限定や網羅的であることを意図しているのではない。

【００５３】

ＰＳ２０２は、コアスイッチ２３６とプログラマブルファブリック２０４とに結合されるダイレクトメモリアクセス（ＤＭＡ）インタフェイス２３４も含み得る。ＰＳ２０２は、この明細中により詳細に記載されるインタフェイス２５６の１つ、すなわちインタフェイス２５６Ｄ、と、ＯＣＭ２２２と、メモリコントローラ２２８とに結合するメモリスイッチ２３８をさらに含む。

【００５４】

コアスイッチ２３６は、示されるように、ＰＳ２０２のさまざまな部品の間に信号を送る。実施例において、コアスイッチ２３６は、ＰＳ２０２の内部バス（図示しない）に直接結合され得る。そのような実施例において、コアスイッチ２３６と接続するＰＳ２０２内の他の部品の各々は、内部バスを介してコアスイッチ２３６に結合され得る。たとえば、インタフェイス２４０，２４２，２４６，および２４８の各々は、内部バスを経由してコアスイッチ２３６に結合され得る。内部バスは、たとえばアドバンストペリフェラスバス（ＡＰＢ）のような任意のさまざまな異なるバスとして実現され得る。

【００５５】

一般に、ＰＳ２０２は、およそ４つのＩ／Ｏのカテゴリーを含み得る。ＰＳ２０２には、フラッシュメモリタイプのインタフェイスと、より高いパフォーマンスのインタフェイスと、より低いパフォーマンスのインタフェイスと、デバッギングインタフェイスとが設けられる。Ｉ／Ｏの第１カテゴリに関して、ＰＳ２０２は、２４０Ａおよび２４０Ｂとして説明される１つ以上のフラッシュメモリインタフェイス２４０を含み得る。たとえば、１つ以上のフラッシュメモリインタフェイス２４０は、４ビット通信用に構成されるクワッドシリアルペリフェラスインタフェイス（ＱＳＰＩ）として実現され得る。１つ以上のフラッシュメモリインタフェイス２４０は、パラレル８ビットＮＯＲ／ＳＲＡＭタイプの

インタフェイスとして実現されてもよい。１つ以上のフラッシュメモリインタフェイス２４０は、８ビットおよび／または１６ビット通信用に構成されるＮＡＮＤインタフェイスとして実現され得る。述べられる特定のインタフェイスは、説明の目的のためであり、限定を目的とするものでないと理解されるべきである。異なるビット幅を有する他のインタフェイスも使用され得る。

【００５６】

Ｉ／Ｏの第２カテゴリに関して、ＰＳ２０２は、Ｉ／Ｏの第１カテゴリよりもより高いレベルの性能を与える１つ以上のインタフェイス２４２を含み得る。インタフェイス２４２Ａ－２４２Ｃの各々は、ＤＭＡコントローラ２４４Ａ－２４４Ｃにそれぞれ結合され得る。たとえば、１つ以上のインタフェイス２４２は、ユニバーサルシリアルバス（ＵＳＢ）タイプのインタフェイスとして実現され得る。１つ以上のインタフェイス２４２は、ギガビットイーサネット（登録商標）タイプのインタフェイスとして実現されてもよい。１つ以上のインタフェイス２４２は、セキュアディジタル（ＳＤ）タイプのインタフェイスとして実現されてもよい。

【００５７】

Ｉ／Ｏの第３カテゴリに関して、ＰＳ２０２は、Ｉ／Ｏの第２カテゴリよりも低いレベルの性能を与えるインタフェイス２４６Ａ－２４６Ｄのような１つ以上のインタフェイス２４６を含み得る。たとえば、１つ以上のインタフェイス２４６は、汎用Ｉ／Ｏ（ＧＰＩＯ）タイプのインタフェイスとして実現され得る。１つ以上のインタフェイス２４６は、ユニバーサル非同期レシーバ／トランスミッタ（ＵＡＲＴ）タイプのインタフェイスとして実現されてもよい。１つ以上のインタフェイス２４６は、シリアルペリフェラルインタフェイス（ＳＰＩ）バスタイプのインタフェイスの形で実現されてもよい。１つ以上のインタフェイス２４６は、コントローラエリアネットワーク（ＣＡＮ）タイプのインタフェイスの形で実現されてもよい。１つ以上のインタフェイス２４６は、トリプルタイマカウンタ（ＴＴＣ）および／またはウォッチドッグタイマ（ＷＤＴ）タイプのインタフェイスの形で実現されてもよい。

【００５８】

Ｉ／Ｏの第４カテゴリに関して、ＰＳ２０２は、プロセッサＪＴＡＧ（ＰＪＴＡＧ）ポートまたはインタフェイス２４８Ａ、およびトレースインタフェイス２４８Ｂのような１つ以上のデバッグインタフェイス２４８を含み得る。ＰＪＴＡＧポート２４８Ａは、ＰＳ２０２のための外部のデバッグインタフェイスを与え得る。トレースインタフェイス２４８Ｂは、トレースなどのデバッグ情報をプログラマブルファブリック２０４から受けるためのポートと、ＰＳ２０２のデバッギングデータをプログラマブルファブリック２０４に送り出すためのポートと、クロストリガポートとを与え得る。クロストリガポートは、プログラマブルファブリック２０４内の回路にＰＳ２０２内のトレースのようなデバッグ機能を起動することができる。同様に、ＰＳ２０２は、プログラマブルファブリック２０４内に実現される回路内のデバッグ機能を開始させ得る。

【００５９】

示されるように、インタフェイス２４０，２４２，２４６，および２４８の各々は、マルチプレクサ２５０に結合され得る。マルチプレクサ２５０は、内部にＩＣ２００が配置されるパッケージのボールなどのＩＣ２００の外部のピンに直接送られまたは結合される複数の出力を与える。たとえば、ＩＣ２００の複数のＩ／Ｏピン、たとえば５３個のピン、は、インタフェイス２４０，２４２，２４６，および２４８の間で共有され得る。ユーザは、ＰＳ２０２の一部としてマルチプレクサ２５０をインタフェイス２４０－２４８のいずれが使用されるか、したがって、マルチプレクサ２５０を経由してＩＣ２００のＩ／Ｏピンに結合されるかを選択するように構成することができる。

【００６０】

示されるように、インタフェイス２４２－２４８は、ファブリックマルチプレクサ入力／出力（ＦＭＩＯ）インタフェイス２５２にも選択的に結合され得る。したがって、ＩＣ２００、より具体的にはＰＳ２０２のユーザ設定に基づいて、カテゴリ２，３，４、すな

わちインタフェイス２４２－２４８の任意の１つは、ＦＭＩＯインタフェイス２５２を経由してＩＣ２００のプログラマブルファブリック２０４に結合され得る。このことは、さらなる処理または／およびモニタリングのために、インタフェイス２４２－２４８のいずれか１つを経由して通知されるデータがプログラマブルファブリック２０４内の回路に送られるようにすることができる。

【００６１】

制御レジスタ２５４は、ほとんどでなくとも、さまざまなＰＳ２０２のアスペクトを制御するように設定され得る。１つ以上の指令は、制御レジスタ２５４に書き込まれてＰＳ２０２のオペレーションを制御または調整する。たとえば、プログラマブルファブリック２０４内の回路は、ここでさらに詳細に記載されるインタフェイス２５６Ｂのようなインタフェイスを介して制御レジスタ２５４に書き込まれ得る。制御レジスタ２５４は、知的所有権（ＩＰ）イネーブルリセットの制御と、クロックユニット２２６によって生成されるクロック周波数の設定、Ｉ／Ｏのドライブ強度の特定および、他のシステムレベル機能のような機能を制御または調整することができる。制御レジスタ２５４は、ＰＳ２０２のパワーダウン、ＰＳ２０２の特定のインタフェイスを個別にパワーダウンまたは非活性化するような追加の機能を調整することができる。制御レジスタ２５４は、たとえば、制御レジスタ２５４をコアスイッチ２３６に結合させるＡＰＢ（図示しない）のようなバスを介してアクセスされ得る。

【００６２】

ＰＳ２０２は、プログラマブルファブリック２０４に直接結合するインタフェイス２５６Ａ－２５６Ｄとして示される１つ以上のインタフェイス２５６も含み得る。実施例において、１つ以上の全てのインタフェイス２５６は、ＡＲＭによって発行されるＡＭＢＡ　ＡＸＩプロトコル仕様（ＡＸＩ)に従って実現され得る。たとえば、インタフェイス２５６の各々は、ＡＭＢＡ　ＡＸＩプロトコル仕様Ｖ．２．０に準拠して実現され得る。一般に、ＡＸＩは、サブミクロン相互接続に適した高い性能、高い周波数のインタフェイスである。

【００６３】

図２を再び参照して、インタフェイス２５６Ａおよび２５６Ｂの各々は、たとえば、プログラマブルファブリック２０４をコアスイッチ２３６に結合する２つの３２ビットチャネルを与えるように実現され得る。インタフェイス２５６Ａは、汎用マスタインタフェースとして実現され得る。インタフェイス２５６Ａは、たとえば、ＰＳ２０２および／またはＤＭＡコントローラからプログラマブルファブリック２０４へのデータの汎用転送を実行するために使用され得る。たとえば、インタフェイス２５６Ｂは、ＰＳ２０２とプログラマブルファブリック２０４との間の汎用データ転送のために使用され得る。

【００６４】

インタフェイス２５６Ａ－２５６Ｂとコアスイッチ２３６とを介して、プログラマブルファブリック２０４内に実現される回路は、さまざまなインタフェイス２４０，２４２，２４６，および２４８の１つにアクセスし得る。インタフェイス２５６Ａおよび／または２５６Ｂを介して、コアスイッチ２３６との組み合わせにより、プログラマブルファブリック２０４内の回路はＯＣＭ２２２に直接、そしてメモリコントローラ２２８などを介してオフチップメモリに、さらにアクセルし得る。

【００６５】

インタフェイス２５６Ｃは、プログラマブルファブリック２０４をコア複合体２０６、より具体的にはＳＣＵ２１８、に直接結合する６４ビットスレーブインタフェイスとして実現され得る。インタフェイス２５６ＣとＳＣＵ２１８とを介して、プログラマブルファブリック２０４内に実現される回路は、コア２０８および２１０、ＩＲＱ２１６、Ｌ２キャッシュ２２０およびＯＣＭ２２２の各々の中にあるＬ１キャッシュへの直接のアクセスを与える。したがって、プログラマブルファブリック２０４内の回路は、そのようなメモリに対する読出しおよび／または書き込みと、コア複合体２０６内で生成されるまたはアサートされる割り込みの検出とを行ない得る。加えて、信号２９０は、ポートまたは信号

としてＩＲＱ２１６に与えられ得る、プログラマブルファブリック２０４からの１つ以上の割り込みおよび／またはＰＳ２０２から、より具体的にはコア複合体２０６からの、ポートまたは信号しとしてプログラマブルファブリック２０４へ提供され得る、１以上の割り込みのコピー、を表わし得る。より具体的には、プログラマブルファブリック２０４にポートまたは信号として与えられ得るコア複合体２０６からポートまたは信号としてプログラマブルファブリック２０４に与えられ得る。別の実施例において、インタフェイス２５６Ｃは、コプロセッサとしての回路機能による使用に適するコア複合体２０６へのコヒーレントアクセスを与える。たとえば、プログラマブルファブリック２０４内にＰＳＣたとえばＰＳＣ２８２の形で実現されるソフトプロセッサは、インタフェイス２５６Ｃを経由してＰＳ２０２と通信し得る。

【００６６】

インタフェイス２５６Ｄは、複数、たとえば４つの、６４ビットスレーブインタフェイスを与えるように実現され得る。インタフェイス２５６Ｄは、ＰＳ２０２とプログラマブルファブリック２０４内に実現される回路との間の大量のデータを効率良く交換するために使用され得る。示されるように、インタフェイス２５６Ｄは、メモリスイッチ２３８を経由するＯＣＭ２２２へのアクセスと、メモリスイッチ２３８およびメモリコントローラ２２８を経由するオフチップメモリへのアクセスを有する、プログラマブルファブリック２０４内に実現される回路を提供する。

【００６７】

ＰＳ２０２は、プロセッサ設定アクセスポート（ＰＣＡＰ）２５８をさらに含む。示されるように、ＰＣＡＰ２５８は、設定コントローラ２６０とプログラマブルファブリック２０４内に配置されるシステムモニタブロック２６２とに結合され得る。設定コントローラ２６０とシステムモニタブロック２６２とは、ハードウェアにより実現される回路の形で実現され得る。設定コントローラ２６０は、設定データを設定メモリセルへ書き込むことを担当し、それによって、プログラマブルファブリック２０４内の設定データにより特定される回路を物理的に実現する。システムモニタブロック２６２は、アナログディジタル（ＡＤ）変換、電圧モニタリング、電流モニタリング、および／または温度モニタリングのような機能を実行し得る。

【００６８】

プログラマブルファブリック２０４は、プログラマブル相互接続回路を使用してともに結合され得る１つ以上のプログラマブル回路ブロックを含むように実現され得る。プログラマブル回路ブロックとプログラマブル相互接続回路とは、ＩＣ２００内にロードされた設定データに基づいて、１つ以上の異なる物理回路、たとえばユーザ回路２８０実現するように構成され得る。プログラマブルファブリック２０４、その中に実現されるハードウェアにより実現されるさまざまな回路を除く、は、設定データが設定メモリ内にロードされて、プログラマブルファブリック２０４内に物理回路が実現されるまで、動作ユニットまたは機能ユニットでないと理解されるべきである。

【００６９】

プログラマブルファブリック２０４は、ハードウェアにより実現される回路の形の１つ以上のインタフェイスを実現するようにも構成され得る。たとえば、ＪＴＡＧインタフェイス２６２と、１つ以上のＭＧＴ２６６Ａ－２６６Ｄと、周辺部品相互接続エクスプレス（ＰＣＬｅ）インタフェイス２６８と、相互設定アクセスポート（ＩＣＡＰ）２７０と、セキュリティポート２７２とは、ＩＣ２００の一部のプログラマブルファブリック内にハードウェアにより実現される回路として含まれ得る。プログラマブルファブリック２０４を参照して述べられるさまざまなインタフェイスは、実現され得るインタフェイスの例を説明するのであって、この明細書中に開示される１つ以上の実施例について制限または限定することを意図するものではない。

【００７０】

たとえば、設定データは、ＩＣ２００内にロードされ、設定コントローラ２６０によって受信され得る。実施例において、設定データは、ＩＣ２００の設定プロセスを制御し得

るＰＳ２０２を介して受信され得る。プログラマブルファブリック２０４内にハードウェアにより実現される回路として実現され得る設定コントローラ２６０は、ＰＣＡＰ２５８を経由してＰＳ２０２から受信される設定データをＩＣ２００の設定メモリ（図示しない）内にロードし得る。ユーザ回路２８０のような異なる物理回路は、ＩＣ２００の設定メモリ内にロードされる特定の設定データによって特定されるプログラマブルファブリック２０４内に実現または形成され得る。ハードウェアにより実現される回路によるこのような態様における設定データのローディングは、プログラマブルファブリック２０４の初期設定を要求しないと理解されるべきである。設定データのローディングの結果によってプログラマブルファブリック２０４内に実現される回路は、物理回路であるか、その回路が、ハード回路あるいはそうでなければＩＣ２００内に固定されるというよりはむしろプログラマブル回路内に形成されるという点で、典型的に「ソフト」と称される。

【００７１】

ＰＳＣ２８２は、ユーザ回路２８０同様、上述された任意のさまざまなインタフェイスを介してＰＳ２０２に結合され得る。直接アクセスは、インタフェイス２５６を経由して与えられ得、一方、ＰＳ２０２へのさらなるアクセスは、ＦＭＩＯインタフェイス２５２を介して容易にされ得る。ＰＳＣ２８２にオフロードされる特定の機能性およびプロセスは、一般に、ＰＳ２０２との通信に必要なインタフェイスのタイプを決定すると理解されるべきである。

【００７２】

図３は、この明細書中に開示される別の実施例に従うプロセッサシステムの拡張のために構成されるＩＣ３００を説明するための第３ブロック図である。ＩＣ３００は、図２のＩＣ２００を参照して述べられるものとして実質的に実現され得る。図３は、しかし、ＰＳ３０２からプログラマブルファブリック３０４内のＰＳＣ３２０のような回路へのソフトウェアベースプロセスのオフロードの摘要図を示す。

【００７３】

図３は、プログラムコード３１５を実行するコア３１０を示す。プログラムコード３１５は、１つ以上の異なるプロセスを含み、または特定し得る。説明のため、プログラムコード３１５は、プロセス１と、プロセス２と、プロセス３と、プロセス４とを含むように示される。プロセス２は、ＰＳＣ、この例ではＰＳＣ３２０、に関連づけられることを説明するために、ハッチングされている。示されるように、ＰＳＣ３２０は、プログラマブルファブリック３０４内に実現される。したがって、コア３１０は、ＰＳＣがプロセス２のために存在することを決定するようにプログラムされ得る。プロセス２を実行するよりも、コア３１０は、プロセッシングをオフロードし得る、そうでなければプロセッシングは、ＰＳ３０２によるプログラマブルファブリック３０４へのプログラムコードの実行を通して実行される。コア３１０は、たとえば、ＰＳＣ３２０に対して入力として利用可能なプロセス２の実行に必要な任意のソースデータを作るようにアレンジし得る。一度ＰＳＣ３２０によるプロセスが完了すると、ＰＳＣ３２０によって生成される任意のデータ結果は、コア３１０によるプロセスまたは使用のために、ＰＳ３０２に与え返され得る。

【００７４】

実施例において、プロセス２は、プログラマブルファブリック３０４内の実現のためにプログラムコード３１５内で識別され、または、そうでなければマークされ得る。たとえば、コア３１５は、プロセス２が実行されておらず、ＰＳＣによるハードウェアにおいて実行されていることを特定するインジケータを識別し得る。別の実施例において、プロセス２は、単に、コア３１５がＰＳＣ３２０に特定のプロセスをオフロードするまたはＰＳＣ３２０を使用するプロセスを実行することを知らせる指令であり得る。その場合、プロセス２は、実行されるプロセスを特定する実際のプログラムコードを含む必要はない。むしろ、進行中、ソフトウェアにおいてよりも、ハードウェア内にプロセス２を実現するように決定がされ得る。プロセス２は、プロセスをＰＳＣ３２０にオフロードするようにＰＳ３０２に指示するインジケータと効率的に置き換えられ得る。さらに別の実施例において、ＰＳＣ３２０にプロセスをオフロードするための決定は、動的に、たとえばフィール

ドにおけるＩＣ３００のオペレーションの間にされ得る。
【００７５】
ＰＳＣの形で特定のプロセスを実行するための決定が、ＩＣ３００内に実現されるシステムの設計サイクルの間に実行されようが、ＰＳ３０２によってフィールドにおいてダイナミックに実行されようが、決定は、１つ以上の異なるコスト測定あるいは単に「コスト」に基づいてなされる。コストは、ＰＳ３０２内のソフトウェア内のプロセスの実行に関連づけられる１つ以上のコストを反映する実行コストを決定するために評価され得る。１つ以上のコストは、プログラマブルファブリック３０４内のＰＳＣ３２０にプロセスをオフロードするための実現コストを決定するためにさらに評価され得る。一般に、実現コストが、実行コストよりも少ない場合、あるいは、実行コストよりも所定の量またはパーセンテージ少ない場合は、プロセスは、対応するＰＳＣにオフロードされ得る。
【００７６】
実施例において、実行コストと実現コストとの各々は、時間要素、電力要素、または時間要素およびパワー要素の両方の幾つかの組み合わせを含み得る。場合によって、ＰＳ３０２からプログラマブルファブリック３０４へのプロセスのオフローディングは、有益である。なぜなら、ＰＳＣは、ＰＳ２０３がプロセスを実行し得るよりも早くプロセスを実行し得るためである。他の場合においては、ＰＳＣは、プロセスを実行するためにＰＳ３０２によって使用されるエネルギーよりも少ないエネルギーでプロセスを実行し得る。さらに他の場合には、ＰＳＣは、ＰＳ３０２よりも少ない時間および少ないエネルギーでプロセスを実行し得る。実行コストおよび実現コストは、プロセスのオフローディングが、省時間または省電力あるいは省時間と省電力の両方をもたらす状況を識別するために、算出され比較され得る。
【００７７】
実施例において、時間要素は、時間遅れの面で測定され得る。たとえば、プロセスをオフロードするか否かの決定は、ＰＳＣが、ＰＳ２０３がプロセスを実行するよりも少ない時間遅れでプロセスを実行し得るか否かに基づく。一例において、各々が図２のＩＲＱ２１６に与えられる割り込みを扱うまたは処理するように構成され得る１つ以上のＰＳＣは、プログラマブルファブリック３０４内に実現され得る。１つ以上のＰＳ３０２の割り込みは、ＩＲＱを介してプログラマブルファブリック３０４内の回路、たとえば１つ以上のＰＳＣにさらされる。割り込みを検出すると、割り込みは、ＰＳ３０２内のプログラムコードの扱いの除外の実行を通して扱われるのと対照的に、プログラマブルファブリック３０４内のＰＳＣによって扱われ得る。ＰＳＣを使用する割り込みの扱いは、ＰＳ３０２に他のタスクまたは要求に応じることによって、システム時間遅れを低減し得る。
【００７８】
図４は、この明細書中に開示される別の実施例に従うプロセッサシステムの拡張のために構成されるＩＣ４００を説明するための第４ブロック図である。ＩＣ４００は、図２のＩＣ２００を参照して述べられたように、実質的に実現され得る。図４は、図３同様に、ＰＳ４０２からプログラマブルファブリック４０４内の回路へのソフトウェアベースプロセスのオフロードを説明するための、ＩＣ４００の摘要図を示す。
【００７９】
図４は、プログラムコード４１５を実行するコア４１０を示す。プログラムコード４１５は、１つ以上の異なるプロセスを含みまたは特定し得る。説明のため、プログラムコード４１５は、プロセス１と、プロセス２と、プロセス３とプロセス４とを含むように示される。プロセス２と４とは、両プロセスがＰＳＣを特定する設定データに対応して関連づけられることを説明するために、ハッチングされている。プロセス２は、プロセス２設定データに関連づけられる。プロセス４は、プロセス４設定データに関連づけられる。
【００８０】
たとえば、プロセス２は、プログラムコード内に実現されさらに「プロセス２設定データ」として示される設定データに関連づけられる。プロセス２設定データがＩＣ４００内にロードされるとき、ＰＳＣ４４０は、プログラマブルファブリック４０４内に実現され

る。プロセス４は、プログラムコード内に実現され、さらに「プロセス４設定データ」として示される設定データに関連付けられる。プロセス４設定データがＩＣ４００内にロードされるとき、ＰＳＣ４４５はプログラマブルファブリック４０４内に実現される。

【００８１】

ＰＳＣ４４０とＰＳＣ４４５とは、プログラマブルファブリック４０４内にそれぞれ実現されるエリアを説明するために、ブロックとして示される。ＰＳＣ４４０と４４５とを実現するよう要求されるエリアは、プログラムコード４１５の一部として知られかつ記憶され、または要求されるときの使用のために、ＩＣ４００内の他のメモリ内に記憶され得る。

【００８２】

実施例において、ＰＳＣの各々は、ＰＳＣのサイズが見積もられまたは決定され得るように、進行の間、プログラマブルファブリック内に提供され得る。別の実施例において、ＰＳＣのサイズは、プログラムコードのライン（ＬＯＣ）の１つ、またはＬＯＣまたは指令を実現するために必要とされるプログラマブルファブリックの平均部品数に対する指令との関係に基づいて見積もられまたは算出され得る。たとえば、指令の各々は、実現のために特定の数のＬＵＴを要求し得る。そのため、所与のプロセスのためのＰＳＣのサイズは、ＬＯＣの各々を実行するために必要とされるハードウェアユニットの数で乗算されるプロセスのためのＬＯＣとして決定され得る。

【００８３】

示されるように、プログラマブルファブリック４０４は、プログラマブルファブリック４０４内にすでに提供実現されユーザ回路設計を表わすユーザ回路４２０を含む。ＩＣ４００は、プログラマブルファブリック４０４内の回路を実現するために設定データを設定メモリセル内にロードするために構成される設定コントローラ４２５をさらに含む。利用可能スペース４３０は、プログラマブルファブリック４０４の使用されていない部分を表わす。利用可能スペース４３０のサイズは、さらに、知られまたは決定されかつコア４１０に利用可能とされ得る。たとえば、設定コントローラ４２５は、利用可能スペース４３０を算出し、標準特定用途プログラミングインタフェイス（ＡＰＩ）を介してＰＳ４０２がその情報を利用できるようにする。

【００８４】

したがって、実施例において、プロセス２の実行の代わりにプログラマブルファブリック４０４内にプロセス２のような特定のプロセスを提供するか否か決定するとき、コア４１０は、プログラマブルファブリック４０４内の利用可能スペース４３０がＰＳＣ４４０を実現するために十分であるか否かを決定し得る。説明されるように、利用可能スペース４３０は、ＰＳＣ４４５でなく、ＰＳＣ４４０を実現するために十分広い。

【００８５】

利用可能スペース４３０の量は、ＩＣ４００のオペレーションの間、たとえばＩＣ４００がフィールドにおいて時々動的部分的再構成、完全な再構成などを受けるときに変化し得ると理解されるべきである。したがって、所与のプロセスがハードウェア内に実現され得るか否かは、プロセスのオフローディングが望まれ得る特定の時間での利用可能スペース４３０の量と同様に、すでに手短に議論されたさまざまな要素に基づき得る。

【００８６】

プログラマブルファブリックに特定のプロセスをオフロードするか否か決定するとき、別掲のとおり、実行コストは、実現コストと比較され得る。一般に、実現コストおよび実行コストの各々は、時間、たとえばプロセスが実行され得るスピード、時間遅れ、電力および、たとえばプロセスを実行するのに消費されるまたは必要とされる電力の量、あるいは時間および時間遅れの両方、および／または電力との幾つかの組み合わせに依存する。

【００８７】

実行コストは、異なるさまざまな技術を使用することによって算出され得る。一局面において、たとえば、実行コストは、実行されるプロセスのＬＯＣの数に対応して定められ得る。別の局面において、実行コストは、実行されるオペレーションの数および／または

それぞれのオペレーションのタイプに対応して定められ得る。たとえば、追加オペレーションは、実行するための第１時間量と第１電力量とを必要とすることが知られ得る。特定のＤＳＰオペレーションは、実行するための第２時間量と第２電力量とを必要とすることが知られ得る。電力消費と実行時間とに関する情報は、一般に、プロセッサまたはＰＳの、メーカまたはプロバイダから利用可能である。

【００８８】

そのため、所与のプロセッサまたはＰＳに対し、ＬＯＣの実行または特定の動作の実行に必要とされる時間は、一般に知られている。同様に、ＬＯＣを実行または所与のタイプの動作の実行に必要とされる電力量も、一般に知られている。ＰＳによって実行されるプログラムコードの各々は、オフロードされ、実行コストを特定し得るプロファイルに関連づけられ得る。実行コストは、電力消費、時間遅れ、待機時間、あるいは、電力消費、実行する時間、および／または時間遅れのうちの２つ以上の組み合わせを反映し得る。

【００８９】

実現コストは、さらに、さまざまな異なるテクニックを使用することによって決定され得る。たとえば、実現コストは、プロセスのオフローディングをセットアップするためにＰＳによって実行されなければならない１つ以上の動作に依存し得る。オフローディングをセットアップするために、たとえば、ＰＳは、最初に、ＰＳＣによって必要とされる入力としてのソースデータが利用可能であることを保証する必要があり得る。そのため、ＰＳは、ソースメモリからデータを読み出し、ソースデータを計算し、ソースデータをＰＳＣがソースデータにアクセス可能なデスティネーションメモリに書き込み、および／または１つ以上のアドレス変換を実行するように要求され得る。

【００９０】

加えて、メモリにアクセス、たとえば読み出しおよび／または書き込みするために必要とされる時間の量は、メモリの配置に依存する。電力消費に関しても同様である。一般に、ＰＳ内部のメモリは、ＩＣ外部のメモリ、たとえばメモリコントローラを介してアクセスされるＲＡＭ、よりも短い時間および少ない電力消費においてアクセスされ得る。さらに、Ｌ１キャッシュは、Ｌ２キャッシュより短い時間においてアクセスされ得、Ｌ１またはＬ２キャッシュのいずれかは、ＯＣＭよりも短い時間においてアクセスされ得る。

【００９１】

ＰＳＣが未だプログラマブルファブリック内に実現されていない場合、ＰＳＣを設定するための設定データが設定メモリ内にロードされるための追加時間が必要とされる。ＰＳＣを実現するために必要とされる時間は、たとえば、ロードされる設定データと設定が外部ソースからロードされ得るスピードとに依存し得る。プログラマブルファブリック内にＰＳＣを実現するために必要とされる時間および電力とが、見積もられ、実現コストに寄与し得る。

【００９２】

ＰＳＣによって実行される実際の処理の観点から、ＰＳＣは、電力消費と、実行される動作の数および／またはタイプや、動作を実行するための特定の回路などのような要因に対応するプロセスのオフロードを完了するのに必要な時間のためにプロファイルされ得る。典型的に、電力消費とスピートとは、ＰＳＣによって実行される動作数と、ＰＳＣによって実行される動作タイプと、オフロードプロセスの動作を実現するためのＰＳＣの特定の回路素子とに基づいて決定され得る。

【００９３】

実現コストは、さらに、オフロードプロセスの結果がＰＳＣからＰＳに返される態様に依存し得る。別掲のとおり、結果が記憶される特定のメモリと結果がＰＳに送られるために介される通信チャネルとは、時間とパワーとの両方に影響し得、そのため、実現コストに影響し得る。

【００９４】

時間と電力とに加え、図４に示されるように、プログラマブルファブリック内の十分なスペースは、ＰＳＣを実現するために利用可能でなければならない。たとえば、プロセス

の各々は、実行コストと、実現コストと、プロセスのためのＰＳＣを特定する設定データとに関連付けられ得る。実現コストは、プログラマブルファブリック内にＰＳＣを実現するために必要とされるエリアの推定を含みまたは特定し得る。そのため、一局面において、プロセスの各々のための実行コストは記憶され得る。プロセスの実現コストの１つ以上の要素は、記憶され得る。しかし、別掲のとおり、１つ以上の他の要因は、ＩＣ内の状態の変化、たとえばＰＳＣがすでに実現されているか否か、プログラマブル回路内の利用可能スペースが足りるか否かなどによって、動的に、たとえばフィールドにおける動作の間、算出される必要があり得る。

【００９５】

一般に、実現コストおよび／または実行コストは、述べられたさまざまな個別のコスト要素の合算によって算出され得る。別掲のとおり、１つ以上のコスト要素は、重みづけたとえば提供および／または実行算出の全体のコスト内の、特定のコスト要素の重要性を増加させるか、または特定のコスト要素の重要性を低減させる係数と掛けあわされ得る。実行コストと実現コストとの決定のための異なる方法および技術は、説明のために与えられ、この明細書中に開示される１つ以上の実施例を制限することを意図するものではない。

【００９６】

実施例において、ＰＳＣにオフロードされるプロセスは、ＩＣおよびＰＳへのデータの入力と、ＩＣおよび／またはＰＳからのデータの出力とを含むものであり得る。ＰＳＣは、効率的に、ＰＳのＩ／Ｏ周辺機器を提供し得る。たとえば、ＰＳＣは、入来データを受信し、データの初期プロセッシングを実行し、ＰＳによってアクセス可能なＲＡＭ内にデータを記憶するように構成され得る。したがって、ＰＳは、必要なときに、ＲＡＭからのデータにアクセスし得る。ＲＡＭ内のデータの収容は、事前処理と同様、ＰＳにかかわらず、ＰＳＣによって実行され得る。同様に、ＲＡＭのようなメモリに記憶されるデータは、ＰＳにかかわらず、ＰＳＣによって出力され得る。ＰＳＣは、ＰＳにかかわらず、指定されるメモリからデータを読み出し、必要な場合任意のプロセスも実行し、データ結果を出力し得る。ＰＳは、たとえば、この明細書中に述べられた１つ以上の技術を使用することによって、データが出力されるようにＰＳＣに通知する。上述のＩ／Ｏ周辺機器としてのＰＳＣの使用を説明する一例は、ビデオプロセッシングの状況中にある。ＰＳＣは、１つ以上のプリプロセッシングタスクを実行し、ＰＳによる使用のためメモリ内にデータ結果を記憶し得る。

【００９７】

図５は、この明細書中に開示される別の実施例に従うＩＣ内のプロセッサシステムの拡張の方法５００を説明するための第１フローチャートである。方法５００は、この明細書中に開示されるようなＩＣ、たとえばプログラマブルファブリックに結合されるＰＳを含むもの、によって実現され得る。方法５００は、一般に、フィールドにおいて適応され得る動的な態様における動作中のＰＳの拡張のための技術を説明する。しかし、述べられたさまざまなコストおよびプログラマブルファブリック内の実現のために別のプロセスを経て１つのプロセスを選択するためのテクニックは、さらに、システムの進行の間適用され得る。これは、ＰＳによって実行されるプログラムコード内へのそのような決定を「ハードコーディング（hard-code）」するためおよび／またはどのプロセスがＰＳによって扱われるか、およびどのプロセスがＰＳＣ、たとえばＩ／Ｏ周辺機器、インタラプトハンドラなどとして提供されるべきかを決定するためである。

【００９８】

ＰＳがどのプロセスを動的に、たとえば動作の間およびフィールドにおける実行時間で、実現するかを決定する実施例において、ＰＳによって実行されるプログラムコードは、図５を参照して述べられる知能と決定とを含み得る。さらに、ソフトウェアプロセスのプロファイリングの観点から使用されるさまざまな量およびソフトウェアプロセスを実現するＰＳＣは、ＰＳ内で実行するプログラムコードの一部として記憶され、プログラムコードの実行の間ＰＳによってアクセス可能な態様でＩＣ内の他の場所に記憶され、または実行時間あるいはその間に算出され得る。

【００９９】
したがって、方法５００は、ＩＣのＰＳがプログラムコードを実行するステップ５０５において開始し得る。別掲のとおり、プログラムコードは、複数の異なるプロセスを含み得る。１つ以上のプロセスは、プログラマブルファブリックにオフロードされ得る。ＰＳによって実行されるプログラムコードは、さらに、オフローディングのためのプロセスの選択とそのようなプロセスをいつオフロードするかの決定において、図５を参照して述べられる機能性を含み得る。
【０１００】
ステップ５１０において、ＰＳは、ＰＳＣとしてプログラマブルファブリックにオフロードされるために実行されるプログラムコードの候補プロセスを決定し得る。実施例において、ＰＳは、予め定められた量の時間内、たとえば次のプロセスが実行される時間内に実行されると予測される１つ以上のプロセス、次のＮプロセス内のプロセス、Ｎは予め定められた整数値、または将来における予め定められた数のクロックサイクル数内に実行されると推定されるプロセスを識別し得る。たとえば、ＰＳは、候補プロセスを選択する先行（look-ahead）機能を活用し得る。
【０１０１】
ステップ５１５において、ＰＳは、候補プロセスがＰＳＣに関連付けられるか否か決定し得る。候補プロセスがＰＳＣに関連付けられる場合、方法５００は継続する。候補プロセスがＰＳＣに関連付けられない場合、方法５００は、異なる候補プロセスを選択するためにループバックし得る。
【０１０２】
ステップ５２０において、ＰＳは、ＰＳＣがすでにプログラマブルファブリック内に提供されたか否か決定し得る。１つ以上のＰＳＣは、すでにプログラマブルファブリック内に実現され得る。ＰＳＣがすでにプログラマブル回路内に実現されている場合、実現コストの時間要素および電力要素は、オフローディングに先行してプログラマブルファブリック内にＰＳＣの実現を要求するよりも少なくなる。ＰＳＣがまだプログラマブルファブリック内に実現されていない場合、実現コストの時間要素とコスト要素の各々は、プログラマブルファブリック内にすでにＰＳＣが実現されたものよりも大きくなる。さらに、ＰＳは、ＰＳＣの実現のためにプログラマブルファブリック内に十分な利用可能スペースが存在するか否か決定しなければならない。したがって、ＰＳＣが、プログラマブルファブリック内に実現される場合は、方法５００は、ステップ５３５に処理を進める。ＰＳＣがプログラマブルファブリック内に実現されない場合は、方法５００はステップ５２５に処理を進める。
【０１０３】
ステップ５２５において、ＰＳは、プログラマブルファブリック内の利用可能スペースの量を決定し得る。別掲のとおり、利用可能スペースの量は、ＰＳＣのようなさらなる回路設計を実現することが可能な使用されていないプログラマブルファブリックの量であり得る。ステップ５３０において、ＰＳは、プログラマブルファブリック内の利用可能スペースの量がＰＳＣを実現するのに十分であるか否かを決定し得る。ＰＳＣを実現するために十分な利用可能スペースがプログラマブルファブリック内にある場合は、方法５００は、ステップ５３５に処理を進める。ＰＳＣを実現するための十分な利用可能スペースがプログラマブルファブリック内にないと、方法５００は、オフローディングのための異なる候補プロセスがあればそれを選択するために、ステップ５１０にループバックし得る。
【０１０４】
ステップ５３５において、ＰＳは、ＰＳＣの実現コストとＰＳ内のプロセスの実行コストとを算出または決定し得る。別掲のとおり、必要に応じて、実行コストと実現コストとは、１つ以上の時間要素、１つ以上の電力要素、または１つ以上の時間要素および１つ以上の電力要素の組み合わせを反映させ得る。時間、電力、または両方が考慮されるか否かについて、機能は、実現コストの算出に組み込まれるさまざまな要素を重みづけするために使用される。たとえば、実現コストの算出において、時間は、より重く、たとえば電力

よりも大きな重要性を与え、重みづけられる。あるいは、電力は、時間より重く重みづけられる。実行コストの算出に組み込まれるさまざまなコスト要素を重みづけするために異なる機能が使用されてもよい。

【０１０５】

別の実施例において、実現コストおよび／または実行コストの計算のための複数の機能は、時間、電力またはそれらの組み合わせの値が異なって記憶され得る。実現コストおよび／または実行コストの計算のための特定の機能は、動作状態、動作状況、またはフィールドにおいて動作の間にＩＣに与えられる他の指令に従ってＰＳによって動的に選択され得る。そのため、フィールドにおける実現コストの算出および／または実行コストの算出を変えることによって、所与のプロセスをオフロードするための決定が行なわれる方法は、随時動的に変えられ得る。

【０１０６】

さらに別の実施例において、利用可能スペースの決定は、実現コストの決定ステップに組み込まれ得ると理解されるべきである。たとえば、プログラマブルファブリックが十分な利用可能スペースを含むか否かの決定は、実現コストの算出によって生み出されるいかなる結果に優先し得るバイナリタイプの変数として実現コストに直接組み込まれ得る。別の例において、ＰＳＣを実現するためにプログラマブルファブリック内に十分でないスペースが利用可能な場合は、実現コストが確実に実行コストよりも大きいためにＰＳＣがプログラマブルファブリック内に実現されていないことを保証するのに十分大きい乗算器が使用され得る。

【０１０７】

別掲のとおり、実現コストと実行コストとは、一般に、デザインプロセスの間に知られまたは決定され得、プログラムコード内で特定される。しかしながら、１つ以上のコスト要素は、限定されないが、ＩＣのプログラマブルファブリック内の利用可能スペースの量および評価される特定のプロセスがすでにＰＳＣとしてプログラマブルファブリック内に実現されているか否かを含むＩＣの電流オペレーティング状態、に従って変わり得る。別掲のとおり、ＰＳは、さらに、実現コストおよび／または実行コストの決定のための特定の態様あるいは機能を選択し得る。

【０１０８】

ステップ５４０において、ＰＳは、実現コストと実行コストとを比較する。ステップ５４５において、ＰＳは、プログラマブルファブリックにプロセスをオフロードするか否かを決定する。実行コストが実現コストを超える、あるいは実現コストがあるパーセンテージだけまたは予め定められた量だけ超える場合、ＰＳは、プログラマブルファブリックへのプロセスのオフローディングが無駄でないと決定し実行されるように決定し得る。プロセスがプログラマブルファブリックにオフロードされない場合、方法５００は、ステップ５１０にループバックして実行されるプログラムコードの一部である１つ以上の追加プロセスを評価する。プロセスがオフロードされる場合、方法５００は、ステップ５５０に処理を進める。

【０１０９】

ステップ５５０において、ＰＳは、ＰＳＣがプログラマブルファブリック内に実現されているか否か決定し得る。ステップ５５０は、明確性および説明目的のために示されているが、分離されたプロセスステップとして実行される必要はない。情報はすでにＰＳ、たとえばステップ５２０、から知られているためである。ステップ５５０は、たとえば、ＰＳＣがプログラマブルファブリック内に実現されない場合は、ＰＳＣを実現するために設定プロセスが実行されなければならないことを説明するために示される。

【０１１０】

いずれの場合も、プロセスがＰＳＣとしてすでにプログラマブル回路内に実現されている場合は、方法５００は、ステップ５６０に処理を進める。プロセスがＰＳＣとしてプログラマブルファブリック内にいまだ実現されていないと、方法５００は、ステップ５５５に処理を進める。ステップ５５５において、ＰＳＣは、プログラマブルファブリック内に

実現される。ＰＳＣは、実現されるＰＳＣを特定する設定データのローディングを介して実現され得る。実施例において、プログラマブルファブリックの設定は、ＰＳによって制御、たとえば開始され得る。たとえば、ＰＳは、外部ソースからＩＣ内に設定データをロードし、設定コントローラに設定データを提供し得る。別掲のとおり、プログラマブルファブリックは、完全に再構成され得、またはプログラマブルファブリックの一部は、動的部分再構成を経ることができる。一度プログラマブルファブリックの設定が完了すると、ＰＳＣは、物理的にそこに実現される。

【０１１１】

ステップ５６０において、ＰＳは、ＰＳＣにプロセスをオフロードし得る。ステップ５６５において、ＰＳＣは、オフロードされたプロセスを実行し得る。したがって、ステップ５７０において、オフロードされたプロセスの実行においてＰＳＣによって生じるいかなる結果データも、ＰＳに対して利用可能になり得る。

【０１１２】

図６～図８は、ＩＣのプログラマブルファブリック内の回路へのプロセスのオフロードの実現のためのさまざまな技術を説明する。図６～図８の各々は、明細書中に述べられたＩＣを使用することによって実現され得る。図６～図８の各々は、プログラマブルファブリックへのプロセスのオフロードの決定がされ、かつ、プログラマブルファブリック内のＰＳＣが実現された状態において開始し得る。

【０１１３】

図６は、この明細書中に開示される別の実施例に従うＩＣのプログラマブルファブリック内に実現される回路へのプロセスのオフローディングの方法６００を説明するための第２フローチャートである。図６は、オフロードが、ＰＳによって主に駆動され、完全でない場合のケースを説明する。方法６００は、オフロードされるプロセスが非常に複雑で、かつ比較的小さいデータのセットで動作するときに適用され得る使用例を説明する。図６で説明される例において、データは、ＰＳからプログラマブルファブリックにプッシュされ、結果は、プログラマブルファブリックからＰＳによって獲得される。

【０１１４】

ステップ６０５において、ＰＳは、プロセス特定回路に入力として与えられるべきソースデータを選択的に計算し得る。ステップ６１０において、ＰＳは、ソースデータをＰＳＣに直接プッシュし得る。たとえば、ＰＳは、インタフェイス２５６Ｂのようなスレーブインタフェイスを経由してソースデータをプッシュし、ＰＳＣにソースデータをプッシュするようにＤＭＡを設定し、または、ＦＭＩＯインタフェイス２５２を経由してＰＳＣにデータを送り得る。ステップ６１５において、ＰＳＣは、ソースデータを処理し得る。ＰＳＣがソースデータの処理を終えると、ＰＳＣは、ＰＳによる読み出しのために、メモリ位置で生成された任意の結果データをプログラマブルファブリック内に記憶し得る。

【０１１５】

ステップ６２０において、ＰＳは、ＰＳＣがいつソースデータの処理を完了したかを決定するために、プログラマブルファブリック内の回路をポーリングし得る。たとえば、プロセッサは、ＰＳＣの状態がビジーまたはフリーであることを示すプログラマブルファブリック内の特定のレジスタをポーリングし得る。レジスタは、ＰＳＣの一部であり得、またはＰＳＣの外部にあり得るが、未だプログラマブルファブリック内に配置される。ビジー状態は、ＰＳＣがソースデータの処理を終えていないことを示す。フリー状態は、ＰＳＣがソースデータの処理を完了したことおよび／または結果データが利用可能であることを示す。

【０１１６】

ステップ６２５において、ＰＳは、ＰＳＣの状態を決定し得る。完了している場合、結果データは利用可能であり、方法６００はステップ６３０に処理を進める。完了していない場合は、ＰＳＣはソースデータ処理の処理を終えておらず、方法６００は、ステップ６２０にループバックしてＰＳＣ状態のポーリングを継続し得る。ステップ６３０に処理が進むと、結果データが利用可能な場合は、ＰＳは、プログラマブルファブリック内のメモ

リから結果データを読み出し得る。データは、たとえばＤＭＡ転送などを介して読み出され得る。

【０１１７】

図７は、この明細書中に開示される別の実施例に従うＩＣのプログラマブルファブリック内に実現される回路へのプロセスのオフローディングの方法７００を説明するための第３フローチャートである。図６がＰＳによって駆動されるオフローディング技術を説明するのに対して、図７は、ＰＳとＰＳＣとの間に存在するよりタイトなレベルの統合における例を説明する。したがって、ステップ７０５において、ＰＳは、計算のための入力としてＰＳＣに与えられるいかなるソースデータを任意的に計算し得る。実施例において、オフロードされるべき処理がソースデータを入力として要求するとき、ＰＳは、そのソースデータを計算または取得し得る。プログラマブルファブリックのタイトな統合を容易にするために、ＰＳは、ＰＳの内部のメモリ内のメモリ位置にソースデータを記憶し得る。たとえば、一度計算されると、ソースデータは、ＯＣＭ、Ｌ１キャッシュ、またはＬ２キャッシュ内のアドレスに記憶され得る。

【０１１８】

ステップ７１０において、ＰＳは、ディスクリプタを生成し、記憶し得る。ディスクリプタは、オフロードされるプロセスを実現または実行するために、ＰＳＣのためのリファレンスおよび／または指令を含むメモリのブロックまたはセグメントであり得る。ソースデータ同様、ディスクリプタは、ＯＣＭ、Ｌ１キャッシュまたはＬ２キャッシュ内に記憶され得る。ディスクリプタは、ソースデータに対するポインタまたは基準、結果データをどこに、たとえばどのアドレスまたはメモリに記憶するかに関する指令、などを含み得る。

【０１１９】

実施例において、ディスクリプタを生成すると、ＰＳは、アドレス変換を実行し、仮想アドレスを物理アドレスにあるいはその逆に変換し得る。一般に、ＰＳは、仮想アドレスを理解しまたは解釈し得る。しかし、そこに実現される任意の回路を包括するプログラマブルファブリックは、物理アドレスを仮想アドレスにまたはその逆に変換できない。プログラマブルファブリックは、物理アドレスのみを理解する。このようにディスクリプタ内で特定される任意のアドレスは、プログラマブルファブリックによって用いられる物理アドレスとして特定されなければならない。

【０１２０】

ステップ７１５において、ＰＳは、プロセスがオフローディングの準備ができていることをＰＳＣに通知し得る。通知は、たとえば、ソースデータがＰＳＣによるプロセスに利用可能であることを示し得る。ＰＳは、ＰＳＣに任意のさまざまな異なるメカニズムを用いて通知する。たとえば、ＰＳは、インタフェイス２５６あるいはＦＭＩＯインタフェイス２５２を経由して、ＰＳＣによってモニタされるプログラマブルファブリック内のレジスタ、またはＰＳＣ自身の内部のレジスタに書き込み得る。ステップ７２０において、レジスタが書き込まれたまたは特定の値に書き込まれたことのＰＳＣの決定に応じて、ＰＳＣはプロセッサ内のメモリからディスクリプタを読み出し得る。たとえば、ＰＳＣは、通知目的のためにレジスタが書き込まれると、ディスクリプタが記憶または配置されるＰＳ内の特定のアドレスから読み出すように構成され得る。

【０１２１】

ステップ７２５において、ＰＳＣは、ディスクリプタによって特定されるソースデータを取り戻しまたは読み出し得る。ステップ７３０において、ＰＳＣは、ソースデータを処理しかつ結果データを生成または出力し得る。ステップ７３５において、ＰＳＣは、ディスクリプタによって特定される場所に結果データを記憶し得る。結果データは、ＰＳの内部メモリ内、たとえばＯＣＭ，Ｌ１キャッシュまたはＬ２キャッシュにも記憶され得る。

【０１２２】

ステップ７４０において、ＰＳＣは、結果データが利用可能であることをＰＳに通知する。たとえば、ＰＳＣは、ＰＳによってモニタされる内部メモリのアドレス、たとえばＯ

ＣＭ，Ｌ２キャッシュまたは Ｌ２キャッシュ内に書き込み得る。実施例において、通知が書き込まれるまたは提供されるアドレスは、ディスクリプタによって特定もされ得る。メモリアドレスがたとえば結果データが準備できていることを示すＰＳＣからの値とともに書き込まれたことをＰＳが決定すると、ＰＳは、ディスクリプタによって特定されたようにＰＳＣが結果データを記憶した場所から結果データを読み出し得る。特定のポート、たとえばインタフェイス２５６Ｃが使用されると、キャッシュの一貫が達成され得ると理解されるべきである。他のインタフェイス、たとえばインタフェイス２５６Ａ，２５６Ｂおよび／または２５６Ｄは、ＰＳのキャッシュまたは内部メモリへのアクセスを保持しなくてもよい。ステップ７４５において、ＰＳは結果データを取り戻しまたは読み出し得る。

【０１２３】

ディスクリプタの使用は、毎回同じＰＳＣが呼び出される場合であっても、ソースデータおよび任意の結果データが記憶される場所を変えることができると理解されるべきである。ＰＳＣは、ディスクリプタによって特定される物理アドレスを読み出し、列挙された場所からソースデータを取得し、かつ列挙された場所に結果データを記憶する。そのような場所は、同じＰＳＣまたは異なるＰＳＣのために意図された複数のディスクリプタの各々の間で異なり得る。

【０１２４】

図８は、この明細書中に開示される別の実施例に従うＩＣのプログラマブルファブリック内に実現される回路へのプロセスのオフローディングの方法８００を説明するための第４フローチャートである。図８は、ＰＳがＰＳＣを活用して、ＰＳＣとＰＳとの間の接続手順のようなトランザクション設定および作業によって負担をかけられることなくプロセスの複数の細かい動作を実現するオフローディングメカニズムを説明する。図８は、ディスクリプタの使用を活用もする、図７に対する代替のメカニズムを示す。一般に、複数のディスクリプタが、ＰＳＣに提供され得る。ＰＳＣは、以下に述べられるような割り込まれない連続した態様でにおけるディスクリプタの各々を実行し得る。

【０１２５】

ステップ８０５において、ＰＳは、計算のための入力としてＰＳＣに提供される任意のソースデータを任意的にに計算し得る。別掲のとおり、一旦計算されると、ソースデータは、ＯＣＭ，Ｌ１キャッシュまたは Ｌ２キャッシュ内に記憶され得る。別の実施例において、ソースデータは、ＩＣのプログラマブルファブリック内に配置されるメモリ内に記憶され得る。

【０１２６】

ステップ８１０において、ＰＳは、複数のディスクリプタを生成しかつ記憶し得る。ディスクリプタの各々は、ＰＳＣのための参照および／または指令を含むメモリのブロックまたはセグメントであり、プロセスがオフロードされるように実現または実行し得る。実施例において、ＰＳは、ＰＳの内部メモリの１つ内に複数のディスクリプタを記憶し得る。別の実施例において、ＰＳは、複数のディスクリプタを、ＰＳＣによってアクセス可能なプログラマブルファブリック内またはＰＳＣ内のキューのようなメモリ内に記憶し得る。

【０１２７】

別掲のとおり、ディスクリプタは、そこから入力として使用される任意のソースデータを取得する場所、および任意の結果データが記憶される場所、たとえば物理メモリアドレスを特定し得る。複数のディスクリプタの処理を実施するために、前述の情報に加え、ディスクリプタの各々は、リンクされたディスクリプタの連鎖を形成する複数のディスクリプタ内の次のディスクリプタに対するポインタを含むまたは特定し得る。ディスクリプタの鎖中の最後のディスクリプタは、次のディスクリプタに対するポインタの欠如によって特徴づけられる。

【０１２８】

ディスクリプタ内で特定され得る追加の情報は、たとえばハードウェアにおいて実行される計算ステップのためのインジケータ、データを取得するためまたはデータが記憶され

るバッファポインタ、ＰＳへの通知または通信のためのメールボックスまたはレジスタアドレスを含み得る。述べられた複数のディスクリプタを使用することによって、ＰＳＣは、それ以上ディスクリプタが残らないようになるまで処理を続け得る。たとえば、ＰＳＣは、ディスクリプタがいつ特定のメモリ内またはキュー内に記憶されるかを検知するように構成され得る。ＰＳＣは、最後のディスクリプタが処理されるまで、ＰＳからのさらなる指令を伴うことなく自動的に複数のディスクリプタの各々を処理し続け得る。この配置は、複数のディスクリプタ内の第１ディスクリプタの処理のための設定を要求する。しかし、それに続くディスクリプタの設定は、最小化または省略される。

【０１２９】

ステップ８１５において、ＰＳＣは、複数のディスクリプタのディスクリプタを取り込み得る。たとえば、ＰＳＣは、ＰＳによって生成または記憶される複数のディスクリプタの第１ディスクリプタの存在のための所定のメモリ場所をチェックするように構成され得る。ＰＳＣは、処理のためのディスクリプタを現在のディスクリプタとして取り出し得る。ステップ８２０において、ＰＳＣは、現在のディスクリプタを処理し得る。たとえば、ＰＳＣは、任意の特定されるソースデータを取り戻し、結果データを生成するためのソースデータを処理し、現在のディスクリプタとして示される結果データを記憶し得る。

【０１３０】

ステップ８２５において、ＰＳＣは、処理されるべきさらなるディスクリプタが残っているか否か決定し得る。たとえば、ＰＳＣは、現在のディスクリプタが次のディスクリプタに対するポインタを特定するか否か決定する。現在のディスクリプタが次のディスクリプタに対するポインタを特定する場合、ＰＳＣは、ステップ８３０に処理を進める。ステップ８３０において、ＰＳＣは、ポインタによって特定された次のディスクリプタを選択し、ステップ８１５にループバックする。現在のディスクリプタが次のディスクリプタに対するポインタを特定しない場合、ＰＳＣは、現在のディスクリプタが最後のディスクリプタであると決定する。従って、ＰＳＣは、ディスクリプタの処理を中止し、ステップ８３５に処理を進める。

【０１３１】

ステップ８３５において、ＰＳＣが複数のディスクリプタの各々の処理を完了すると、ＰＳＣは、結果データが利用可能であることをＰＳに通知する。実施例において、ＰＳＣは、ディスクリプタの処理の完了に応じて、ＰＳがそこから結果データを取り戻し得る特定の場所内に結果データを記憶し得る。その場所は、１つ以上のディスクリプタの個々のものからの結果データが記憶され得る中間の場所と異なっていてもよい。

【０１３２】

指示の受信に応じて、ＰＳは、結果データを取得する。たとえば、ＰＳは、ＰＳＣからの結果データの利用可能性を待ちながら別のスレッドを実行しそれによってＰＳに他のタスクを実行させる。別の実施例において、ＰＳは、メールボックス、セマフォまたは割り込みを活用してＰＳに通知し得る。

【０１３３】

一般に、プロセスのオフローディングのために不適切な技術の使用は、プログラマブルファブリックまたは定期的なポーリングに対するソースデータの提供のような論理動作によって負担を強いられる時間および／または電力の任意の節約をもたらし得る。この点で、実現のコストを計算すると、実現のコストは、プログラマブルファブリックへのプロセスのオフロードに使用される特定の技術によって変動すると理解されるべきである。そのため、たとえば、選択されるプロセスの実現のコストは、オフロードが図６、図７または図８の方法を使用して実行されるか否かに従って変動する。

【０１３４】

図中のフローチャートは、この明細書中に開示される１つ以上の実施例に従うシステム、方法およびコンピュータプログラム製品の可能性のある実施例のアーキテクチャと、機能と、動作とを示す。この点で、フローチャートのブロックの各々は、特定される論理機能を実現する実行可能なプログラムコードの１つ以上の部分を備えるモジュール、セグメ

ント、またはコードを表わし得る。

【０１３５】

幾つかの代替的な実行例において、ブロック内に示される機能は、図に示される順序と異なって生じてもよい。たとえば、連続して示される２つのブロックは、実際に、含まれる機能によって、実質的に同時に実行されてもよいし、あるいは、ときにはブロックは逆の順序で実行されてもよい。フローチャートの例のブロックの各々と、フローチャートの例のブロックの組み合わせとは、特定の機能、または作用、あるいは特別な目的のハードウェアと実行可能な指令との組み合わせを実行する特別な目的のハードウェアベースのシステムによって実現され得る。

【０１３６】

１つ以上の実施例は、ハードウェアまたはハードウェアとソフトウェアとの組み合わせとして実現され得る。１つ以上の実施例は、１つのシステムに集中される形、または異なる要素がいくつかの相互接続されたシステムをまたがって分配される形で実現され得る。本明細書で述べられた方法の少なくとも一部を実行するように適合される任意の種類のデータ処理システムまたは他の装置が好適である。

【０１３７】

１つ以上の実施例は、さらに、ここで述べられた方法の実行例をイネーブルする全ての特徴を含むコンピュータプログラム製品のようなデバイスに組み込まれ得る。デバイスは、プログラムコードを記憶する、非一時的データ記録媒体、たとえばコンピュータが使用可能媒体またはコンピュータが読み取り可能媒体な、を含み得る。プログラムコードは、メモリとプロセッサとを備えるシステム内にロードされ実行されると、システムにこの明細書中に述べられた機能の少なくとも一部を実行させる。非一時的データ記録媒体の例は、光媒体、磁気媒体、光磁気媒体、ランダムアクセスメモリまたはハードディスクのようなコンピュータメモリなどを含み得るが、それらに限定されない。施例の範囲を示す、以下の請求項が参照されるべきである。

【０１３８】

「コンピュータプログラム」、「ソフトウェア」、「アプリケーション」、「コンピュータが使用可能なプログラムコード」、「プログラムコード」、「実行可能なコード」、それらの変形および／または組み合わせ、この文脈において、いかなるの言語、コードまたは表記は、情報処理能力を有するシステムに直接または以下のいずれかあるいは両方の後に特定の機能を実行させることを意図した一組の指令の任意の表現を意味する。ａ）別の言語、コードまたは表記への変換、ｂ）異なる材料形態での再生。たとえば、プログラムコードは、サブルーチン、機能、処理、オブジェクトメソッド、オブジェクトインプリメンテーション、実行可能なアプリケーション、アプレット、サーブレット、ソースコード、オブジェクトコード、共有ライブラリ／ダイナミックロードライブラリおよび／またはコンピュータシステムで実行するために設計される他の指令のシーケンスを含み得る。

【０１３９】

「ａ」および「ａｎ」の用語は、ここで使用されるように、１つ以上として定められる。「複数」の用語は、ここで使用されるように、２つ以上として定められる。「別の」の用語は、ここで使用されるように、少なくとも第２またはそれ以上のものとして定められる。「含む」および／または「有する」の用語は、ここで使用されるように、備える、すなわちオープンランゲージとして定められる。「結合され」の用語は、ここで使用されるように、いかなる要素も介在せずに直接か１つ以上の要素を介在して間接的に接続されるものとして定められる。２つの要素は、機械的に、電気的に結合されるか、または通信チャネル、配線、ネットワークあるいはシステムを介して通信可能にリンクされ得る。

【０１４０】

この明細書中に開示される１つ以上の実施例は、その中の精神または本質的属性から逸脱することなく他の形で実施され得る。したがって、前述の明細書よりも、１つ以上の実施例の範囲を示す、以下の請求項が参照されるべきである。

【図１】

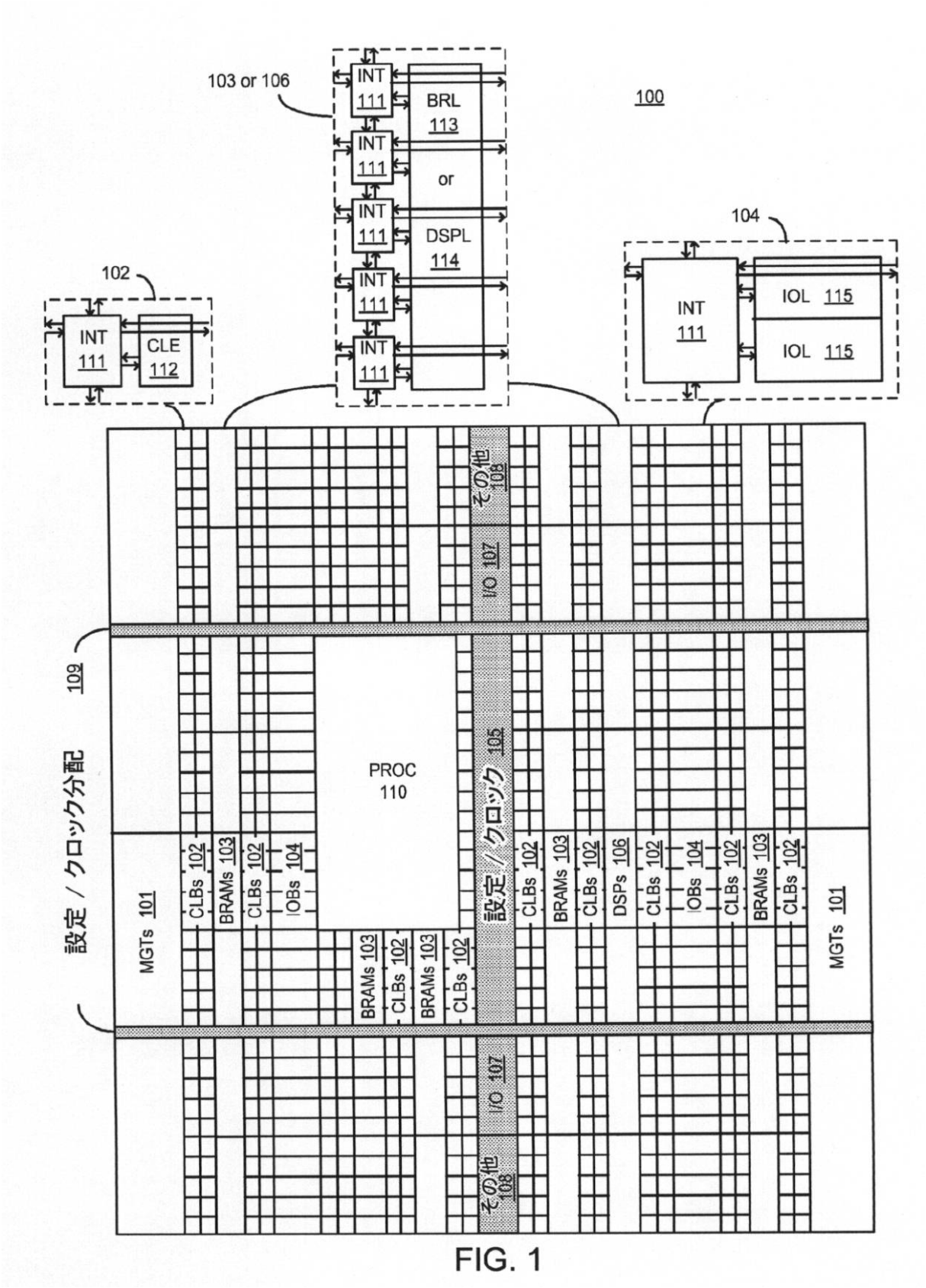
103 or 106
100
INT 111
BRL 113
or
DSPL 114
102
CLE 112
104
IOL 115
その他 108
I/O 107
109
設定／クロック分配
PROC 110
MGTs 101
CLBs 102
BRAMs 103
IOBs 104
DSPs 106

FIG. 1

【図２】

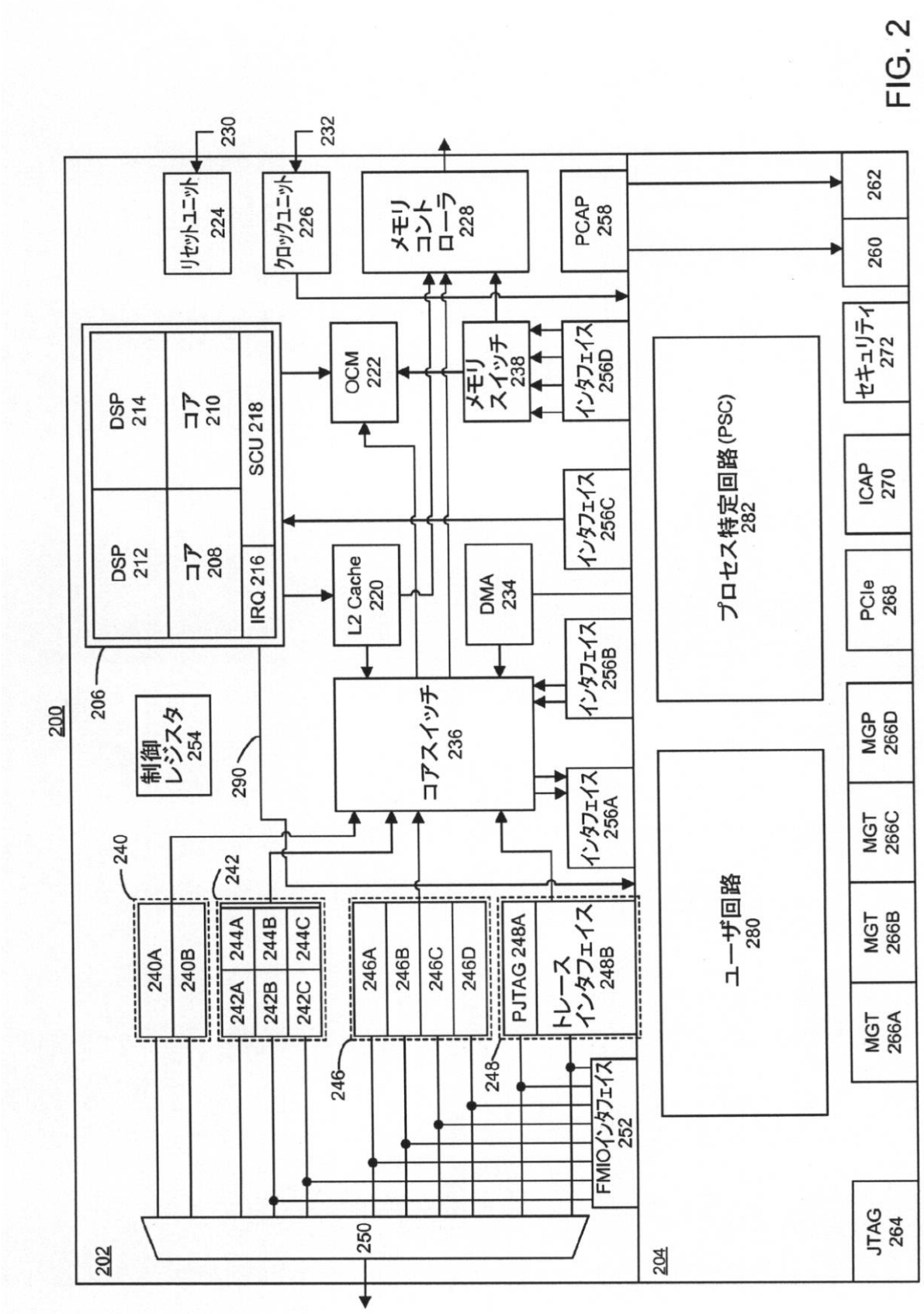
200
202
204
206
リセットユニット 224
230
クロックユニット 226
232
メモリコントローラ 228
PCAP 258
262
260
DSP 214
コア 210
SCU 218
DSP 212
コア 208
IRQ 216
OCM 222
メモリスイッチ 238
インタフェイス 256D
セキュリティ 272
L2 Cache 220
インタフェイス 256C
ICAP 270
DMA 234
プロセス特定回路 (PSC) 282
PCIe 268
制御レジスタ 254
290
コアスイッチ 236
インタフェイス 256B
MGP 266D
インタフェイス 256A
240
242
240A
240B
242A
242B
242C
244A
244B
244C
246A
246B
246C
246D
PJTAG 248A
トレースインタフェイス 248B
ユーザ回路 280
MGT 266C
MGT 266B
MGT 266A
246
248
FMIOインタフェイス 252
250
JTAG 264

FIG. 2

【図３】

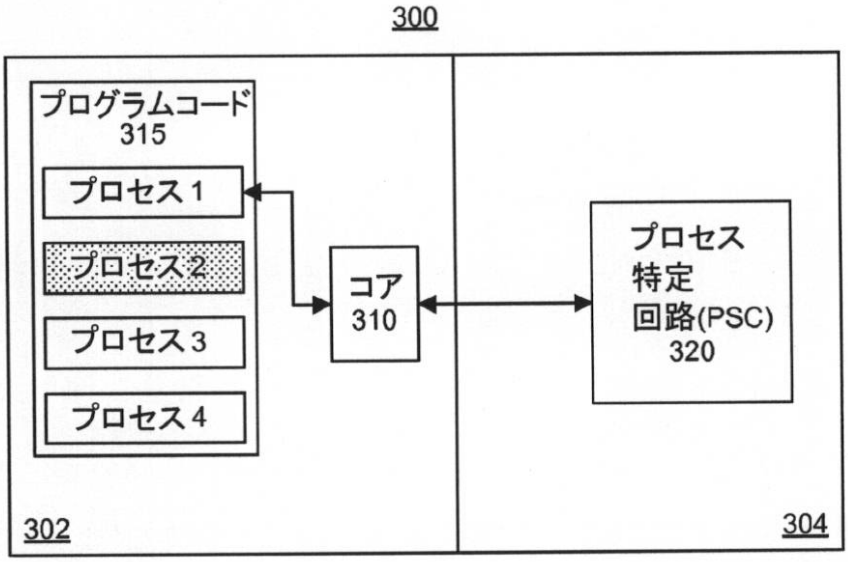

FIG. 3

【図４】

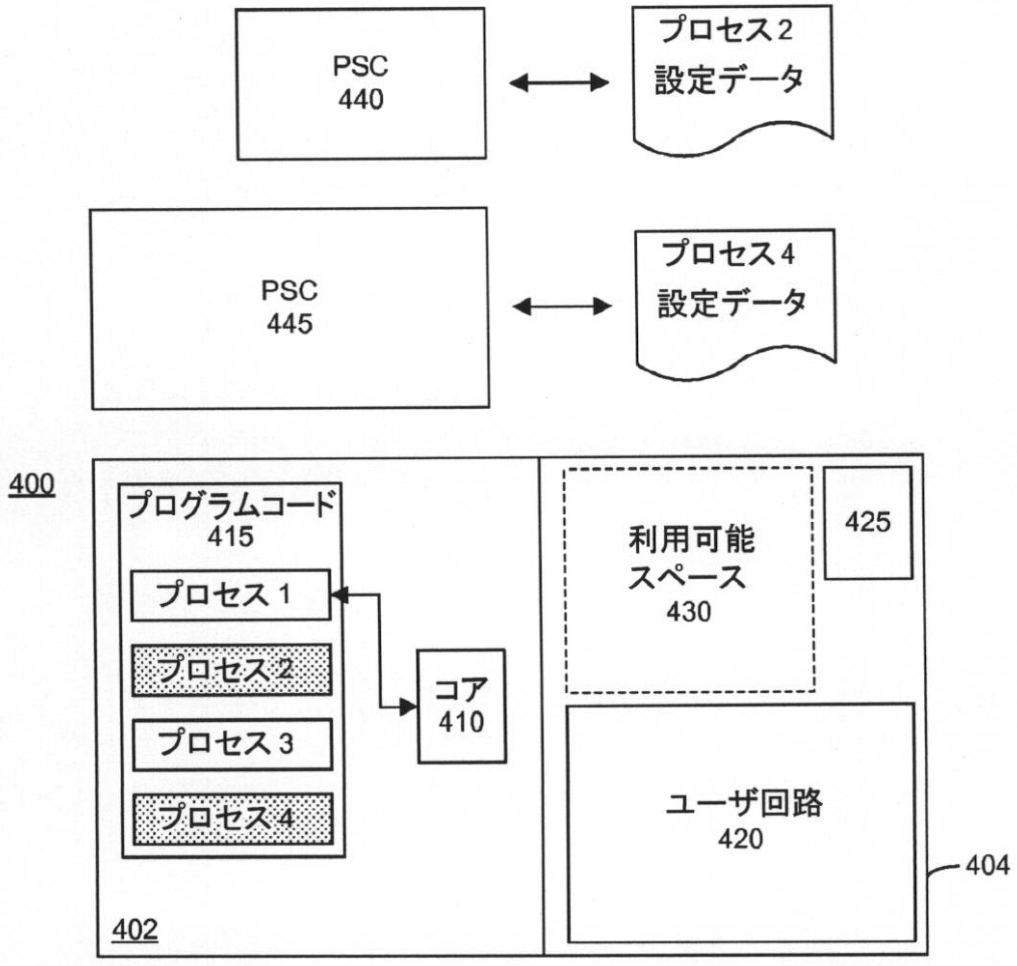

FIG. 4

【図５】

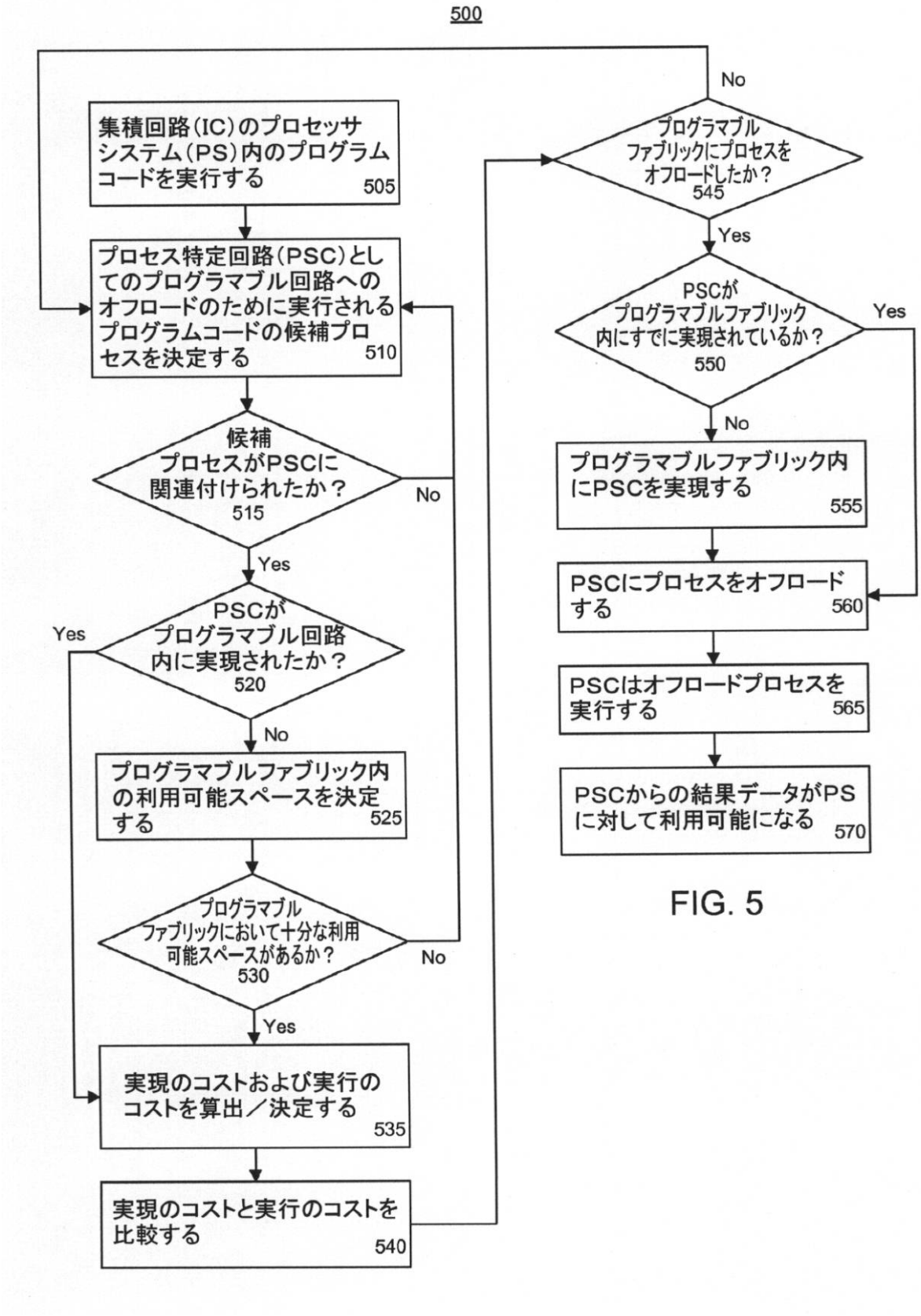
500
集積回路(IC)のプロセッサシステム(PS)内のプログラムコードを実行する 505
プロセス特定回路(PSC)としてのプログラマブル回路へのオフロードのために実行されるプログラムコードの候補プロセスを決定する 510
候補プロセスがPSCに関連付けられたか？ 515
No
Yes
PSCがプログラマブル回路内に実現されたか？ 520
Yes
No
プログラマブルファブリック内の利用可能スペースを決定する 525
プログラマブルファブリックにおいて十分な利用可能スペースがあるか？ 530
No
Yes
実現のコストおよび実行のコストを算出／決定する 535
実現のコストと実行のコストを比較する 540
プログラマブルファブリックにプロセスをオフロードしたか？ 545
No
Yes
PSCがプログラマブルファブリック内にすでに実現されているか？ 550
Yes
No
プログラマブルファブリック内にPSCを実現する 555
PSCにプロセスをオフロードする 560
PSCはオフロードプロセスを実行する 565
PSCからの結果データがPSに対して利用可能になる 570

FIG. 5

【図６】

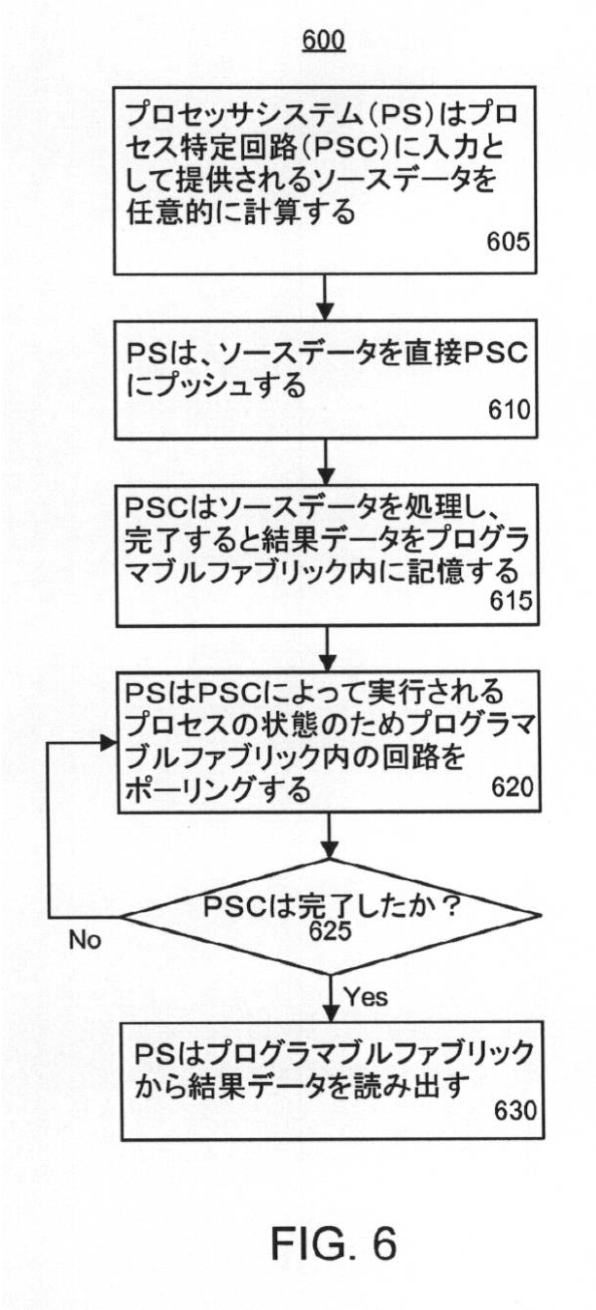
600
プロセッサシステム(PS)はプロセス特定回路(PSC)に入力として提供されるソースデータを任意的に計算する
605
PSは、ソースデータを直接PSCにプッシュする
610
PSCはソースデータを処理し、完了すると結果データをプログラマブルファブリック内に記憶する
615
PSはPSCによって実行されるプロセスの状態のためプログラマブルファブリック内の回路をポーリングする
620
PSCは完了したか？
625
No
Yes
PSはプログラマブルファブリックから結果データを読み出す
630

FIG. 6

【図７】

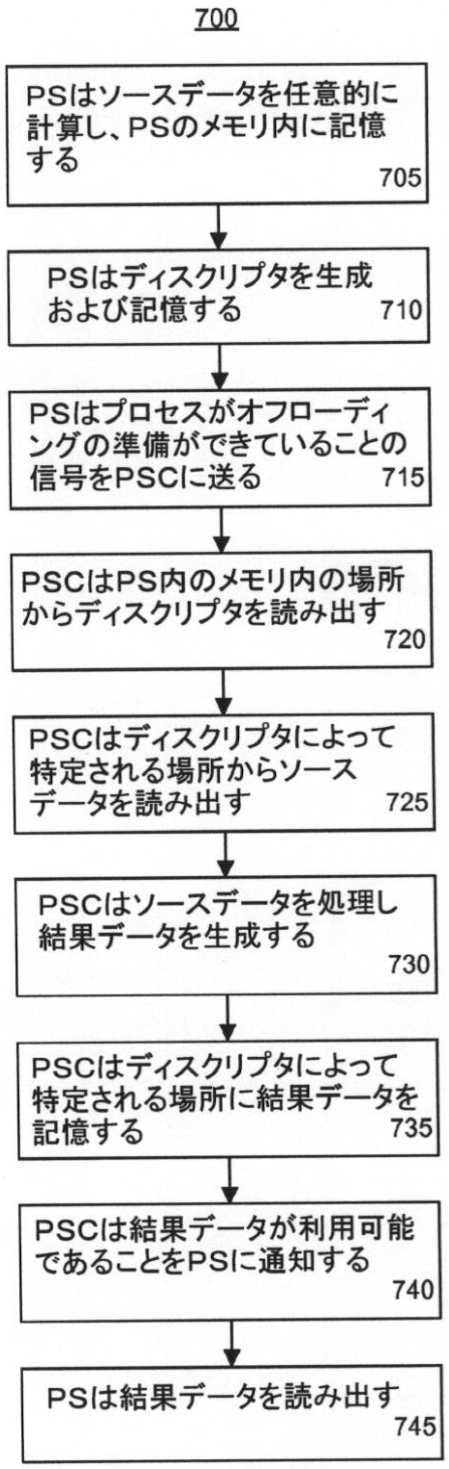
700
PSはソースデータを任意的に計算し、PSのメモリ内に記憶する
705
PSはディスクリプタを生成および記憶する
710
PSはプロセスがオフローディングの準備ができていることの信号をPSCに送る
715
PSCはPS内のメモリ内の場所からディスクリプタを読み出す
720
PSCはディスクリプタによって特定される場所からソースデータを読み出す
725
PSCはソースデータを処理し結果データを生成する
730
PSCはディスクリプタによって特定される場所に結果データを記憶する
735
PSCは結果データが利用可能であることをPSに通知する
740
PSは結果データを読み出す
745

FIG. 7

【図８】

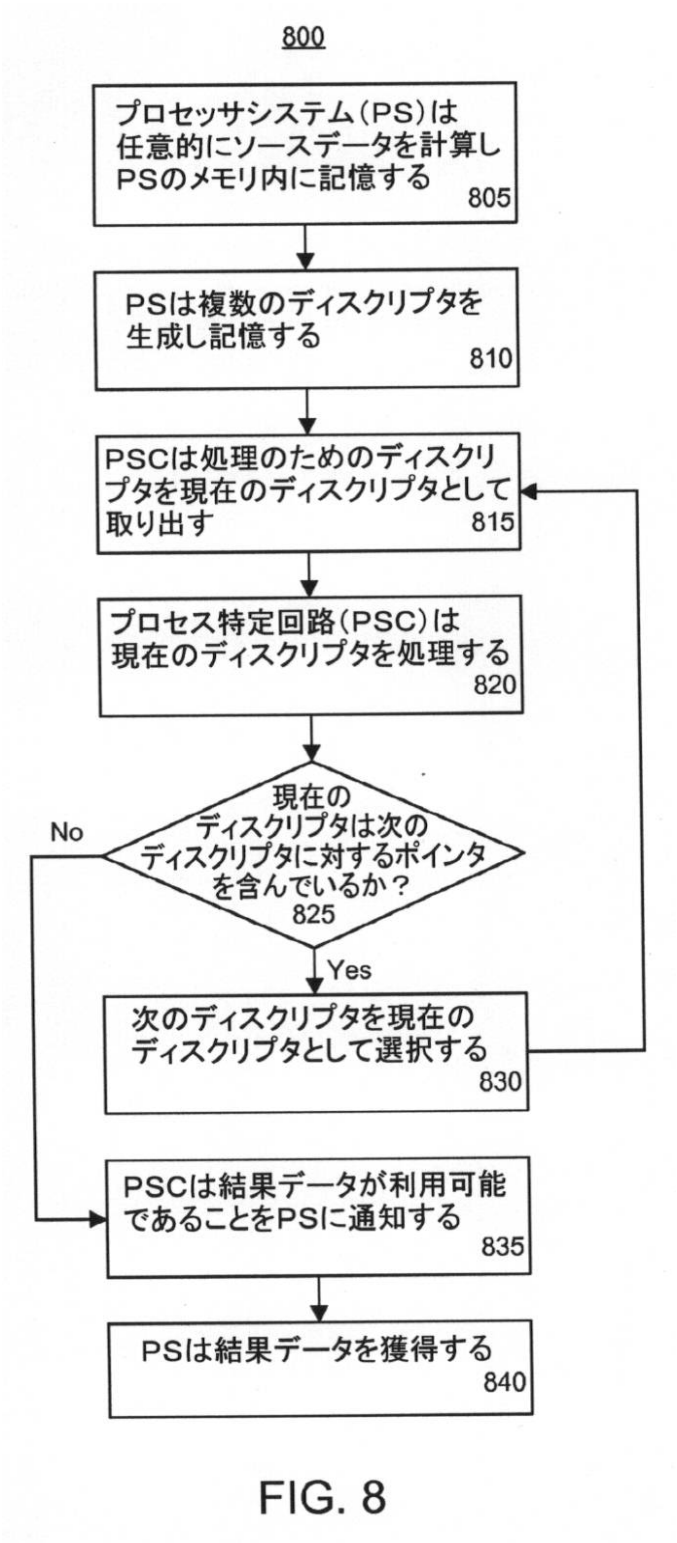

FIG. 8

【手続補正書】
【提出日】平成24年9月29日(2012.9.29)
【手続補正１】
【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】

集積回路（２００）であって、
プログラムコードを実行するように構成されるプロセッサシステム（２０２）と、
前記集積回路のプログラマブル回路（２０４）内に実現されるプロセス特定回路とを備え、前記プロセス特定回路（２８２）は、前記プロセッサシステムに結合されかつ前記プロセッサシステムによってオフロードされるプロセスを実行するように構成され、
前記プロセッサシステムは、フィールドにおいて前記集積回路の動作の間、動的になされる決定に従って、前記プロセスを実行するためのプログラムコードの実行に代えて前記プロセス特定回路にプロセスをオフロードするように、または前記プロセスを実行するためのプログラムコードを実行するように構成される（５１０－５４５）、集積回路。
【請求項２】
前記プロセッサシステムは、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される電力消費の低減に従って、前記プロセス特定回路に前記プロセスをオフロードするか否か決定するようにさらに構成される（５３５，５４０）、請求項１に記載の集積回路。
【請求項３】
前記プロセッサシステムは、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善に従って、前記プロセス特定回路に前記プロセスをオフロードするか否か決定するようにさらに構成される（５３５，５４０）、請求項１に記載の集積回路。
【請求項４】
前記プロセッサシステムは、前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される時間遅れの低減に従って、プロセス特定回路に前記プロセスをオフロードするか否か決定するようにさらに構成される（５３５，５４０）、請求項１に記載の集積回路。
【請求項５】
前記プロセッサシステムは、前記プロセスをオフロードするか否かの前記決定を行なうために使用される第１機能から、前記フィールドにおいて前記集積回路の動作の間、動的に前記プロセスをオフロードするか否かの前記決定を行なうために使用される第２機能へ切り替わるように構成される、請求項１に記載の集積回路
【請求項６】
前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善は、複数のメモリのどのメモリが前記プロセス特定回路に対する前記プロセスに必要とされるソースデータを提供するために使用されるかに依存する（５２５，５３０）、請求項３に記載の集積回路。
【請求項７】
前記プロセッサシステムは、前記プロセス特定回路を実現するために前記プログラマブルファブリック内の利用可能なスペースの量が存在するか否か決定するようにさらに構成される（５２５，５３０）、請求項１から請求項６のいずれか１項に記載の集積回路。
【請求項８】
前記プログラマブル回路はプログラマブルファブリックであり、前記プロセッサシステムは、前記プロセス特定回路を実現するための前記プログラマブルファブリックの少なくとも一部の動的再構成を開始するように構成される（５５５）、請求項１から請求項７のいずれか１項に記載の集積回路。
【請求項９】
集積回路（２００）におけるプロセッサシステムの拡張の方法であって、
前記集積回路内に実現される前記プロセッサシステム（２０２）内のプログラムコード（２０４）を実行するステップ（５０５）を備え、前記集積回路はプログラマブル回路を含み、前記プロセッサシステムは前記プログラマブル回路に結合され、
前記方法は、前記フィールドにおいて前記集積回路の動作の間、動的になされる決定に従って、前記プロセッサシステムでのプロセスを実行するためのプログラムコードの実行

に代えて前記プログラマブル回路内に実現されるプロセス特定回路（２８２）にプロセスをオフロードする、または前記プロセスを実行するための前記プロセッサシステム内の前記プログラムコードを実行するステップ（５１０－５４５）と、

オフロードされると、前記プロセッサシステムに対して利用可能な前記プロセス特定回路から前記プロセスの結果を作成するステップ（５７０）とをさらに備える、方法。

【請求項１０】

前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用するプロセスの実現を通して達成される電力消費の低減に従って、前記プロセス特定回路を使用する前記プロセスを実行するか否か決定するステップ（５３５，５４０）をさらに備える、請求項９に記載の方法。

【請求項１１】

前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される時間遅れの低減に従って、前記プロセス特定回路を使用する前記プロセスを実行するか否か決定するステップ（５３５，５４０）をさらに備える、請求項９に記載の方法。

【請求項１２】

前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の低減に従って、前記プロセス特定回路を使用する前記プロセスを実行するか否か決定するステップ（５３５，５４０）をさらに備える、請求項９に記載の方法。

【請求項１３】

前記プロセッサシステムは、前記プロセスをオフロードするか否かの前記決定を行なうために使用される第１機能から、前記フィールドにおいて前記集積回路の動作の間動的に前記プロセスをオフロードするか否かの前記決定を行なうために使用される第２機能へ切り替わるように構成される、請求項９に記載の方法。

【請求項１４】

前記プロセッサシステムを用いる代わりに前記プロセス特定回路を使用する前記プロセスの実現を通して達成される完了時間の改善の測定は、前記プロセス特定回路による前記プロセスの実行のために必要とされるソースデータを提供するために複数のメモリのどのメモリが用いられるかに依存する（５３５，５４０）、請求項１２に記載の方法。

【請求項１５】

前記プログラムコードによって特定される複数のプロセスから前記プロセスを選択するステップ（５１０）と、

ハードウェアにおいて前記選択されるプロセスを実現する前記プロセス特定回路を特定する設定データを選択するステップ（５１５）と、

前記集積回路の前記プログラマブル回路内に前記プロセス特定回路を実現するために選択された設定データをロードするステップ（５５５）とをさらに備える、請求項９から請求項１４のいずれか１項に記載の方法。

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2011/064038

A. CLASSIFICATION OF SUBJECT MATTER
INV. H03K19/177 G06F9/50
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H03K G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	MICHALIS D GALANIS ET AL: "Speedups and Energy Savings of Microprocessor Platforms with a Coarse-Grained Reconfigurable Data-Path", PARALLEL AND DISTRIBUTED PROCESSING SYMPOSIUM, 2007. IPDPS 2007. IEEE INTERNATIONAL, IEEE, PI, 1 March 2007 (2007-03-01), pages 1-8, XP031175434, ISBN: 978-1-4244-0909-9 figure 1 page 2, right-hand column, line 30 - line 33 page 4, right-hand column, line 8 - line 17 page 6, right-hand column, line 10 - line 27 figure 7 table 3 -/--	1-15

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 February 2012	22/02/2012

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Loiseau, Ludovic

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2011/064038

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
	page 3, left-hand column, line 33 - right-hand column, line 8	
X	JEFFREY M ARNOLD ED - JUN TANG ET AL: "Software Configurable Processors", APPLICATION-SPECIFIC SYSTEMS, ARCHITECTURES AND PROCESSORS, 2006. ASAP '06. INTERNATIONAL CONFERENCE ON, IEEE, PISCATAWAY, NJ, USA, 1 September 2006 (2006-09-01), pages 45-49, XP031021335, ISBN: 978-0-7695-2682-9 paragraph [02.1]; figure 2 paragraph [2.2.1]	1,9
X	HAUSER J R ET AL: "Garp: a MIPS processor with a reconfigurable coprocessor", FIELD-PROGRAMMABLE CUSTOM COMPUTING MACHINES, 1997. PROCEEDINGS., THE 5TH ANNUAL IEEE SYMPOSIUM ON NAPA VALLEY, CA, USA 16-18 APRIL 1997, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, US, 16 April 1997 (1997-04-16), pages 12-21, XP010247463, DOI: 10.1109/FPGA.1997.624600 ISBN: 978-0-8186-8159-2 abstract; figure 1 page 12, right-hand column, line 24 - line 26	1,9
X	EP 1 615 141 A2 (HARMAN BECKER AUTOMOTIVE SYS [DE]) 11 January 2006 (2006-01-11) abstract; figure 3	1,9
X	CONNER D: "RECONFIGURABLE LOGIC", EDN ELECTRICAL DESIGN NEWS, REED BUSINESS INFORMATION, HIGHLANDS RANCH, CO, US, vol. 41, no. 7, 28 March 1996 (1996-03-28) , pages 53-56,58,60, XP000592126, ISSN: 0012-7515 page 54, left-hand column, line 16 - line 21	1,9

3

Form PCT/ISA/210 (continuation of second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2011/064038

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 1615141 A2	11-01-2006	NONE	

Form PCT/ISA/210 (patent family annex) (April 2005)

フロントページの続き

(81)指定国 AP(BW,GH,GM,KE,LR,LS,MW,MZ,NA,RW,SD,SL,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,MD,RU,TJ,TM),EP(AL,AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HR,HU,IE,IS,IT,LT,LU,LV,MC,MK,MT,NL,NO,PL,PT,RO,RS,SE,SI,SK,SM,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AO,AT,AU,AZ,BA,BB,BG,BH,BR,BW,BY,BZ,CA,CH,CL,CN,CO,CR,CU,CZ,DE,DK,DM,DO,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,GT,HN,HR,HU,ID,IL,IN,IS,JP,KE,KG,KM,KN,KP,KR,KZ,LA,LC,LK,LR,LS,LT,LU,LY,MA,MD,ME,MG,MK,MN,MW,MX,MY,MZ,NA,NG,NI,NO,NZ,OM,PE,PG,PH,PL,PT,QA,RO,RS,RU,RW,SC,SD,SE,SG,SK,SL,SM,ST,SV,SY,TH,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN

(72)発明者 ルー，ティン
アメリカ合衆国、９５１２４ カリフォルニア州、サン・ホセ、ロジック・ドライブ、２１００

Ｆターム(参考) 5B013 DD03
5B376 CA41 CA68 EA06
5J042 BA01 BA02 CA19 CA20 DA02 DA04