

(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2007-522584

(P2007-522584A)

(43) 公表日 平成19年8月9日(2007.8.9)

(51) Int. Cl.

G06F 9/44 (2006.01)

F I

G06F 9/44 530P

テーマコード (参考)

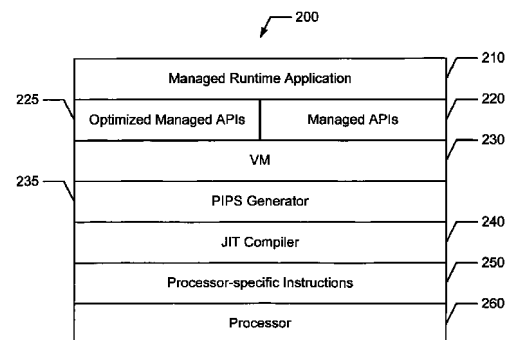
審査請求 有 予備審査請求 未請求 (全 17 頁)

(21) 出願番号	特願2006-553152 (P2006-553152)	(71) 出願人	591003943 インテル・コーポレーション
(86) (22) 出願日	平成17年2月2日 (2005.2.2)		アメリカ合衆国 95052 カリフォル ニア州・サンタクララ・ミッション カレ ッジ プーレバード・2200
(85) 翻訳文提出日	平成18年8月11日 (2006.8.11)	(74) 代理人	100104156 弁理士 龍華 明裕
(86) 国際出願番号	PCT/US2005/002989	(72) 発明者	テイラー、スチュアート アメリカ合衆国、94025 カリフォル ニア州、メンロ パーク、ナンバー69、 シャロン パーク ドライブ 1202
(87) 国際公開番号	W02005/083564	(72) 発明者	ナジェンドラ、グルラジ アメリカ合衆国、97124 オレゴン州 、ヒルズバロ、ナンバー94、ノースイ ースト カーラビィ ウェイ 1441
(87) 国際公開日	平成17年9月9日 (2005.9.9)		最終頁に続く
(31) 優先権主張番号	10/783,840		
(32) 優先日	平成16年2月20日 (2004.2.20)		
(33) 優先権主張国	米国 (US)		

(54) 【発明の名称】 仮想マシン環境におけるアプリケーション・プログラム・インタフェースの最適化の方法および装置

## (57) 【要約】

仮想マシン (VM) 環境における最適化アプリケーション・プログラム・インタフェース (API) に関する方法および装置。まず、1つ以上のプロセッサ命令に関連するプロセッサ命令スタブ (PIPS) が生成される。この「スタブ」は、プログラムの実行時に種々のタスクを実行するために提供される、動的に生成されるコードの一部を指す。そして、生成されたPIPSに基づいて、1つ以上のプロセッサ命令を実行するために最適化アプリケーション・プログラム・インタフェースが生成される。この最適化は、VM環境においてJITコンパイラがプロセッサ特有の最適化を行うことができない問題、すなわち、VM環境においてJITコンパイラが下層のプロセッサによって提供される機能を使用することができない問題を克服する。



**【特許請求の範囲】****【請求項 1】**

1 つ以上のプロセッサ命令に関連するプロセッサ命令プロキシ・スタブを生成するステップと、

前記 1 つ以上のプロセッサ命令に関連する 1 つ以上のプロセッサの機能を使用可能にするために、前記プロセッサ命令プロキシ・スタブに基づいて最適化マネージド・アプリケーション・プログラム・インタフェースを生成するステップと

を備える方法。

**【請求項 2】**

前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成するステップは、マネージド・ランタイム環境の仮想マシンに関連するレイヤにおいて前記プロセッサ命令プロキシ・スタブを生成するステップを備える請求項 1 に記載の方法。 10

**【請求項 3】**

前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成するステップは、マネージド・ランタイム・アプリケーションのインストール時に前記プロセッサ命令プロキシ・スタブを生成するステップを備える請求項 1 に記載の方法。

**【請求項 4】**

前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成するステップは、ストリーミング SIMD 拡張 ( SSE ) 命令、 SSE 2 命令、およびマルチメディア拡張命令のうちの 1 つに関連する前記プロセッサ命令プロキシ・スタブを生成するステップを備える請求項 1 に記載の方法。 20

**【請求項 5】**

前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成するステップは、仮想マシンのマージング言語コードによって前記プロセッサ命令プロキシ・スタブを生成するステップを備える請求項 1 に記載の方法。

**【請求項 6】**

前記プロセッサの前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成するステップは、前記 1 つ以上のプロセッサ命令に関連する前記プロセッサの識別に応じて、前記プロセッサ命令プロキシ・スタブを生成するステップを備える請求項 1 に記載の方法。 30

**【請求項 7】**

前記最適化マネージド・アプリケーション・プログラム・インタフェースに基づいて、マネージド・ランタイム・アプリケーションの実行中に、前記 1 つ以上のプロセッサ命令に関連する機能を使用可能にするステップを更に備える請求項 1 に記載の方法。

**【請求項 8】**

命令群を記録したマシンアクセス可能な媒体であって、

前記命令群は、マシン上で実行されることにより、

1 つ以上のプロセッサ命令に関連するプロセッサ命令プロキシ・スタブを生成し、

前記 1 つ以上のプロセッサ命令に関連する 1 つ以上のプロセッサの機能を使用可能にするために前記プロセッサ命令プロキシ・スタブに基づいて、最適化マネージド・アプリケーションプログラムインタフェースを生成する 40

媒体。

**【請求項 9】**

前記命令は、前記マシンに、マネージド・ランタイム環境の仮想マシンに関連するレイヤにおいて前記プロセッサ命令プロキシ・スタブを生成させることによって、前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成させる請求項 8 に記載の媒体。

**【請求項 10】**

前記マシンに前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成させる前記命令は、前記マシンに、マネージド・ランタイム・アプリケーション 50

ョンのインストール中に前記プロセッサ命令プロキシ・スタブを生成させることを備える請求項 8 に記載の媒体。

【請求項 1 1】

前記マシンに前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成させる前記命令は、前記マシンに、ストリーミング SIMD 拡張 ( SSE ) 命令、 SSE 2 命令、およびマルチメディア拡張命令のうちの 1 つに関連する前記プロセッサ命令プロキシ・スタブを生成させることを備える請求項 8 に記載の媒体。

【請求項 1 2】

前記命令は、前記マシンに、仮想マシンのマーシャリング言語コードによって前記プロセッサ命令プロキシ・スタブを生成させることによって、1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成させる請求項 8 に記載の媒体。 10

【請求項 1 3】

前記命令は、前記マシンに、前記 1 つ以上のプロセッサ命令に関連する前記プロセッサを識別したことに応じて、前記プロセッサ命令プロキシ・スタブを生成させることによって、前記プロセッサの前記 1 つ以上のプロセッサ命令に関連する前記プロセッサ命令プロキシ・スタブを生成させる請求項 8 に記載の媒体。

【請求項 1 4】

前記命令は、前記マシンに、前記最適化アプリケーション・プログラム・インタフェースに基づいてマネージド・ランタイム・アプリケーションを実行させるために、前記 1 つ以上のプロセッサ命令に関連する機能を使用可能にさせる請求項 8 に記載の媒体。 20

【請求項 1 5】

前記マシンアクセス可能な媒体は、プログラマブル・ゲート・アレイ、特定用途向け集積回路、消去可能プログラマブル・リード・オンリー・メモリ、リード・オンリー・メモリ、ランダム・アクセス・メモリ、磁気メディア、および光メディアのうちの 1 つからなる請求項 8 に記載の媒体。

【請求項 1 6】

1 つ以上のプロセッサ命令に関連するプロセッサ命令プロキシ・スタブを生成し、前記プロセッサ命令プロキシ・スタブに基づいて最適化マネージド・アプリケーション・プログラム・インタフェースを生成するプロセッサ命令プロキシ・スタブ・ジェネレータと、前記 1 つ以上のプロセッサ命令に関連するプロセッサの 1 つ以上の機能を使用可能にするために前記最適化マネージド・アプリケーション・プログラム・インタフェースをコンパイルするコンパイラとを備える装置。 30

【請求項 1 7】

前記プロセッサ命令プロキシ・スタブ・ジェネレータは、仮想マシンおよびコンパイラのうちの 1 つに組み込まれている請求項 1 6 に記載の装置。

【請求項 1 8】

前記プロセッサ命令プロキシ・スタブ・ジェネレータは、前記プロセッサ命令プロキシ・スタブを生成するために、前記 1 つ以上のプロセッサ命令に関連するプロセッサを識別する請求項 1 6 に記載の装置。 40

【請求項 1 9】

1 つ以上のプロセッサ命令は、ストリーミング SIMD 拡張 ( SSE ) 命令、 SSE 2 命令、およびマルチメディア拡張命令のうちの 1 つを有する請求項 1 6 に記載の装置。

【請求項 2 0】

前記コンパイラは、ジャスト・イン・タイム・コンパイラを有する請求項 1 6 に記載の装置。

【請求項 2 1】

前記プロセッサ命令プロキシ・スタブは、マネージド・ランタイム環境の仮想マシンに関連するレイヤにおいて生成される請求項 1 6 に記載の装置。

【請求項 2 2】

前記最適化マネージド・アプリケーション・インタフェース・プログラムは、マネージド・ランタイム・アプリケーションの実行に対して前記１つ以上のプロセッサ命令に関連する機能を使用可能にする請求項１６に記載の装置。

【請求項２３】

１つ以上の最適化マネージド・アプリケーション・プログラム・インタフェースを保存するダイナミック・ランダム・メモリ（ＤＲＡＭ）と、

１つ以上のプロセッサ命令に関連するプロセッサ命令プロキシ・スタブを生成し、前記１つ以上のプロセッサ命令に関連する前記プロセッサの１つ以上の機能を使用可能にするために前記プロセッサ命令プロキシ・スタブに基づいて最適化アプリケーション・プログラム・インタフェースを生成する、前記ＤＲＡＭに接続されたプロセッサと

10

を備えるプロセッサシステム。

【請求項２４】

１つ以上のプロセッサ命令は、ストリーミングＳＩＭＤ拡張（ＳＳＥ）命令、ＳＳＥ２命令、およびマルチメディア拡張命令のうちの１つを有する請求項２３に記載のプロセッサシステム。

【請求項２５】

前記プロセッサ命令プロキシ・スタブは、マネージド・ランタイム環境の仮想マシンに関連するレイヤにおいて生成される請求項２３に記載のプロセッサシステム。

【請求項２６】

前記プロセッサ命令プロキシ・スタブは、マネージド・ランタイム・アプリケーションのインストール時に生成される請求項２３に記載のプロセッサシステム。

20

【請求項２７】

前記最適化マネージド・アプリケーション・プログラム・インタフェースは、マネージド・ランタイム・アプリケーションの実行時に前記１つ以上のプロセッサ命令に関連する機能を使用可能にする請求項２３に記載のシステム。

【発明の詳細な説明】

【技術分野】

【０００１】

本開示内容は、概して、マネージド・ランタイム環境に関する。本開示内容は、特に、マネージド・アプリケーション・プログラム・インタフェース（ＡＰＩｓ）の最適化の方法および装置に関する。

30

【背景技術】

【０００２】

マネージド・コードは、マネージド・ランタイム環境（ＭＲＴＥ）（例：Ｍｉｃｒｏｓｏｆｔ（登録商標）社のＣ＃（「Ｃシャープ」）またはＶｉｓｕａｌ Ｂａｓｉｃ .Ｎｅｔによって書かれた全てのコード）の制御下で実行するコードである。これに対して、アンマネージド・コードは、ＭＲＴＥの外側（例：ＣＯＭコンポーネントおよびＷＩＮ３２ＡＰＩ関数）において実行するコードである。概して、マネージド・コードは、ランタイム時にコンポーネントおよびアプリケーションをサポートするために使用され、アンマネージド・コードは、プラットフォーム（すなわち、プロセッサ）との下位レベルの相互作用をサポートするために使用される。アプリケーションがＪａｖａ（登録商標）仮想マシン（ＪＶＭ）およびＭｉｃｒｏｓｏｆｔ .Ｎｅｔによって提供される共通言語ランタイム（ＣＬＲ）のようなＭＲＴＥ上における動作に移行するにつれて、仮想マシンは、アプリケーションを抽象化してプロセッサへの依存を取り除いていく（すなわち、マネージド・ランタイム・アプリケーションは、仮想マシンにより大きく依存するようになり、プロセッサに対する依存は小さくなる）。

40

【０００３】

現在、Ｉｎｔｅｌ（登録商標）インテグレートッド・パフォーマンス・プリミティブ（ＩＰＰ）のようなアンマネージド・ソフトウェア・ライブラリ関数は、概して、Ｉｎｔｅｌ Ｐｅｎｔｉｕｍ（登録商標）テクノロジーおよび／またはＩｎｔｅｌ Ｉｔａｎｉｕｍ

50

(登録商標)テクノロジーのうちの1つ以上を使用して実装されたプロセッサ上のアンマネージド環境における実行に最適化されている。アンマネージド・ソフトウェア・ライブラリ関数は、Intelプロセッサによって提供されるストリーミング・シングル・インストラクション/マルチプル・データ(SIMD)拡張(SSE)命令、SSE2命令、および/またはマルチメディア拡張(MMX)命令のようなプロセッサ特有の命令群を使用したカスタムハンドな最適化コードを書くことによって、特定のプロセッサアーキテクチャ上における動作に対して更に最適化されうる。例えば、文字列比較関数は、アンマネージド・コードによって実装され、SSE2命令を使用したカスタムハンドな最適化コードによって最適化されうる。アンマネージド・コードとは対照的に、マネージド・コードをカスタムハンドに最適化する方法が存在しないため、マネージド・コードは、アンマネージド・コードと同様の方法では特定のプロセッサアーキテクチャに対して最適化されない。例えば、一般的に、マネージドAPIsは、最適化に関してジャスト・イン・タイム(JIT)コンパイラに完全に依存している。その結果、マネージド・ランタイム・アプリケーションは、下層のプロセッサ上での実行において、オーディオ処理、ビデオ処理、画像処理、音声認識、暗号などの機能を使用可能にし、かつ、最適化する、プロセッサ特有の最適化命令群を使用することができない。

10

【図面の簡単な説明】

【0004】

【図1】既存のシステムに構成されたマネージド・ランタイム環境(MRTE)システムのアーキテクチャ階層の一例を示すブロック図である。

20

【0005】

【図2】本明細書で開示される発明の一実施形態により構成されるプロセッサ命令プロキシ・スタブ(PIPS)システムを含む、MRTEシステムの一例におけるアーキテクチャ階層の一例を示すブロック図である。

【0006】

【図3】プロセッサ命令プロキシ・スタブ(PIPS)システムの一例を示すブロック図である。

【0007】

【図4】図3に示すPIPSシステムの一例によって最適化されうるアンマネージド・コードの一例の高級プログラミング言語による表現を示す。

30

【0008】

【図5】図4に示す高級プログラミング言語に相当し、ネイティブ・アセンブリ・コードを最適化するPIPSを含む、ネイティブ・アセンブリ・コードの一例のコード表現を示す。

【0009】

【図6】図3に示すPIPSシステムの一例を実装するために実行されうる、マシンアクセス可能な命令群の一例を示す第1のフロー図である。

【図7】図3に示すPIPSシステムの一例を実装するために実行されうる、マシンアクセス可能な命令群の一例を示す第2のフロー図である。

【0010】

40

【図8】図3に示すPIPSシステムの一例を実装するために使用されうる、プロセッサシステムの一例を示すブロック図である。

【発明を実施するための最良の形態】

【0011】

図1を参照すると、マネージド・ランタイム環境(MRTE)システム100のアーキテクチャ階層は、概して、マネージド・ランタイム・アプリケーション110と、1つ以上のマネージド・アプリケーション・プログラム・インタフェース(APIs)120と、仮想マシン(VM)130と、コンパイラ140と、プロセッサ特有の命令群150と、プロセッサ160とを有する。本明細書で使用される「アプリケーション」は、データの操作を行う1つ以上の方法、プログラム、関数、ルーチン、またはサブルーチンを指す

50

。

## 【0012】

概して、マネージド・ランタイム・アプリケーション110は、MRT Eにおいて種々のサービスを提供するためにプログラマによって書かれる。マネージド・ランタイム・アプリケーション110のソースコードは、例えば、C#、Visual Basic .Net、および/またはその他の適用可能ないかなるオブジェクト指向プログラミング言語によって書かれてもよい。

## 【0013】

Microsoft .Netフレームワーク・クラス・ライブラリまたはJavaクラス・ライブラリのようなマネージドAPIs120は、マネージド・ランタイム・アプリケーション110のソースコードを、それぞれMicrosoft中間言語(MSIL)またはJavaバイトコードに変換(すなわち、コンパイル)する。マネージドAPIs120は、マネージド・ランタイム・アプリケーション110とVM130の間のインタフェースとして動作する。

## 【0014】

VM130は、ガーベッジコレクション、メモリ管理、およびコードとロールベースのセキュリティのようなサービスをマネージドAPIs120に提供することによって、マネージド・ランタイム・アプリケーション110を管理する抽象的なプロセッサを動作させる。例えば、プロセッサに関する知識を持たないVM130は、Microsoft共通言語ランタイムまたはJava仮想マシンでもよい。マネージドAPIs120およびVM130は、MSILコードまたはJavaバイトコードが特定のプロセッサをターゲットとしないために、すべての特定のプラットフォームに対して依存せずに動作する。それに従って、ジャスト・イン・タイム(JIT)コンパイラのようなコンパイラ140は、MSILコードまたはJavaバイトコードを、マネージドAPIs120からプロセッサ160によって実行されうるネイティブ・アセンブリ・コードに変換(すなわち、再コンパイル)する。

## 【0015】

プロセッサ160は、Intel Pentiumテクノロジ、Intel Itaniumテクノロジ、および/またはIntelパーソナル・インターネット・クライアント・アーキテクチャ(PCA)テクノロジのうちの1つ以上を使用して実装されてもよい。プロセッサ160は、暗号、マルチメディア、オーディオコーデック、ビデオコーデック、画像コーディング、画像処理、信号処理、文字列処理、音声圧縮、コンピュータビジョンなどのソフトウェア・ライブラリ関数をMRT Eシステム100に提供する、SSE命令群、SSE2命令群、MMX命令群、および/またはその他の適用可能な命令群のような、プロセッサ特有の命令群150を実行することが可能であってもよい。

## 【0016】

しかし、上述のように、アンマネージド・ソフトウェア・ライブラリ関数(すなわち、プロセッサ特有の命令群150)は、プロセッサ160に対して最適化されうるが、マネージド・コード(すなわち、マネージドAPIs120)は、これまでマネージド・コード関数をカスタムハンドで最適化する方法が存在しないため、アンマネージド・ソフトウェアと同様の方法では、ある特定のプロセッサアーキテクチャに対して最適化されない。すなわち、マネージド・ランタイム・アプリケーション110に対応するマネージドAPIs120は、最適化に関して完全にJITコンパイラ140に依存しており、また、JITコンパイラ140は、プロセッサ特有の最適化を行うことができなかった。このように、従来のシステムでは、VM130が下層のプロセッサ160の一定のプロセッサ特有の命令群150をサポートしていなかったために、下層のプロセッサ160は、VM130によって提供されるサービスを使用することができず、また、マネージド・ランタイム・アプリケーション110は、下層のプロセッサ160によって提供される機能を使用することができなかった。

## 【0017】

10

20

30

40

50

図 2 の例は、マネージド・ランタイム・アプリケーション 210、1つ以上の APIs 220、1つ以上の最適化マネージド APIs 225、VM 230、PIPS ジェネレータ 235、コンパイラ 240、プロセッサ特有の命令群 250、およびプロセッサ 260 を含むプロセッサ命令プロキシ・スタブ (PIPS) システム 200 を有する、MRTE のアーキテクチャ階層を示す。本明細書において使用される「スタブ」は、プログラムの実行中に種々のタスクを実行するために提供される動的に生成されるコードの一部を指す。

#### 【0018】

一般的に、PIPS ジェネレータ 235 は、下層のプロセッサ 260 上におけるマネージド・ランタイム・アプリケーション 210 の実行を最適化するために、PIPS (例：図 5 の PIPS 510) と呼ばれるコードの一部または命令のセットを生成する。マネージド・ランタイム・アプリケーション 210 がインストールされると、例えば、PIPS ジェネレータ 235 は、プロセッサ特有の命令群 250 に基づいて、PIPS を生成する。更に、PIPS ジェネレータ 235 は、マネージド・ランタイム・アプリケーション 210 によって使用される最適化マネージド APIs 225 を作成するために、PIPS があるマネージド APIs 220 に挿入する。下記に詳細に説明するように、マネージド・ランタイム・アプリケーション 210 が実行されている間、最適化マネージド APIs 225 は、アンマネージド・コード (すなわち、プロセッサ特有の命令群 250) をマネージド・コード (すなわち、マネージド・ランタイム・アプリケーション 210) に書き換えることなく、下層のプロセッサ 260 の性能を最適化する。最適化マネージド APIs 225 は、メモリ (例：図 8 のメモリ 1030) に保存され、MRTE におけるマネージド・ランタイム・アプリケーション 210 の実行時にリコールされる。その結果、下層のプロセッサ 260 上におけるマネージド・ランタイム・アプリケーション 210 の性能を最適化するために、下層のプロセッサ 260 の機能が使用可能にされうる。

#### 【0019】

図 2 の PIPS ジェネレータ 235 は PIPS システム 200 内の独立したブロックとして図示されるが、PIPS ジェネレータ 235 によって実行される機能は、VM 230 および / または JIT コンパイラ 240 内に組み込まれてもよい。

#### 【0020】

図 3 を参照すると、PIPS システム 300 の例は、MRTE においてマネージド・ランタイム・アプリケーション 310 を実行するために、マネージド・ランタイム・アプリケーション 310、1つ以上の最適化マネージド APIs 325、VM 330、JIT コンパイラ 340、ネイティブ・アセンブリ・コード 350、およびプロセッサ 360 を有する。VM 330 は、マネージド・ランタイム・アプリケーション 310 を実行するために、複数の異なるプロセッサに対して互換性があるプロセッサ命令群を実行してもよい。しかし、一般的に、VM 330 は、最適化マネージド APIs 325 なしでは使用することができない機能を使用可能にするために下層のプロセッサ 360 の特定のプロセッサ特有の命令群を実行することはしない。これに対して、PIPS システム 300 によってマネージド・ランタイム・アプリケーション 310 が実行されている間、例えば、JIT コンパイラ 340 は、ネイティブ・アセンブリ・コード 350 (例：図 5 のネイティブ・アセンブリ・コード 500) を生成するために、最適化マネージド APIs 325 をコンパイルする。特に、マネージド・ランタイム・アプリケーション 310 のインストール時に、最適化マネージド APIs 325 を生成するために PIPS ジェネレータ 235 が PIPS を挿入したため、JIT コンパイラ 340 は、ネイティブ・アセンブリ・コード 350 を更に最適化することなく、ネイティブ・アセンブリ・コード 350 を単純にコンパイルして実行する。換言すると、PIPS は、下層のプロセッサ 360 上においてマネージド・ランタイム・アプリケーション 310 を実行するために、マネージド・ランタイム・アプリケーション 310 のマネージド APIs を予め最適化 (すなわち、最適化マネージド APIs 325) した。これにより、JIT コンパイラ 340 がアンマネージド・コード (例：図 2 のプロセッサ特有の命令群 250) をマネージド・コード (すなわち、マネ

10

20

30

40

50

ージド・ランタイム・アプリケーション 310) に書き換えることなく、最適化マネージド APIs 325 は、下層のプロセッサ 360 の性能を最適化する。その結果、ネイティブ・アセンブリ・コード 350 は、下層のプロセッサ 360 におけるマネージド・ランタイム・アプリケーション 310 の性能を最適化するように、カスタマイズされる。

#### 【0021】

図 4 の例では、文字列比較関数 400 は、アンマネージドの高級言語のコードによって実装される。概して、文字列比較関数 400 は、上述の Intel プロセッシング・テクノロジーのうちの 1 つ以上を使用して実装されたプロセッサにおける SSE2 命令群のようなプロセッサ特有の命令群を使用したカスタムハンドな最適化コーディングによって、C 言語ルーチンとして最適化される。しかし、C# または Java の比較関数コードのようなマネージド・コードを特定のプロセッシング・アーキテクチャに対してカスタムハンドに最適化する方法は存在しない。

#### 【0022】

図 2 および図 3 と関連して説明されたように、下層のプロセッサ 360 上における文字列比較関数 400 の性能を最適化する PIPS 510 を含む、ネイティブ・アセンブリ・コード 500 の一部の例が、図 5 に示される。特に、ネイティブ・アセンブリ・コード 500 は、PIPS ジェネレータ 235 によって生成された PIPS 510 を有する。例えば、PIPS ジェネレータ 235 は、文字列比較関数 400 のインストール時に PIPS 510 を生成するために、Microsoft .NET によって提供されるネイティブ・マーシャリング言語 (ML) コードを使用してもよい。PIPS 510 に基づいて、PIPS ジェネレータ 235 は、マネージド・ランタイム・アプリケーション 310 に相当する最適化マネージド APIs 325 を生成する。JIT コンパイラ 340 は、図 5 に示すように、実行する下層のプロセッサ 360 に対する PIPS 510 を含む、文字列比較関数に対応するネイティブ・アセンブリ・コード 500 をコンパイルする。文字列比較関数 400 がランタイム時に開始されたとき、VM 330 は、ネイティブ・アセンブリ・コード 500 を生成するために、JIT コンパイラ 340 に対して、最適化マネージド APIs 325 を取り込む。PIPS ジェネレータ 235 が予め PIPS 510 を最適化マネージド APIs 325 に挿入したため、JIT コンパイラ 340 は、最適化マネージド APIs 325 を更に最適化することなく、最適化マネージド APIs 325 をコンパイルして実行する。その結果、下層のプロセッサ 360 のプロセッサ特有の命令群 250 (すなわち、アンマネージド・コード) が PIPS 510 によって VM レイヤにまで抽象化されたため、マネージド・ランタイム・アプリケーション 310 は、VM 330 によって提供されるサービス (例: ガベージコレクション、メモリ管理、および / またはコードとロールベースのセキュリティ)、および、下層のプロセッサ 360 の機能の両方の利益を得ることができる。換言すると、最適化マネージド APIs 325 は、マネージド・ランタイム・アプリケーション 310 を動作させるために、下層のプロセッサ 360 の機能を使用可能にするプロセッサ特有の命令群を使用可能にすることができる。

#### 【0023】

フロー図 600 および 700 は、図 6 および図 7 にそれぞれ図示されるマネージド API を最適化するためにプロセッサによって実行されうる、マシンアクセス可能な命令群を示す。通常の技術を有する当業者は、この命令群が揮発性または非揮発性のメモリまたはその他の大容量ストレージデバイス (例: フロッピーディスク、CD、および DVD) のようなコンピュータによってアクセス可能な全ての媒体に保存された全てのプログラミングコードの使用による多くの異なる方法によって実装されうることを、理解するであろう。例えば、マシンアクセス可能な命令群は、消去可能プログラマブル・リード・オンリー・メモリ (EPROM)、リード・オンリー・メモリ (ROM)、ランダム・アクセス・メモリ (RAM)、磁気メディア、光メディア、および / またはその他の適用可能なタイプの媒体のような、マシンアクセス可能な媒体に組み込まれてもよい。あるいは、マシンアクセス可能な命令は、プログラマブル・ゲート・アレイおよび / または特定用途向け集積回路 (ASIC) に組み込まれてもよい。更に、図 6 および図 7 には動作の特定の順番

10

20

30

40

50



が図示されているが、通常の技術を有する当業者は、これらの動作が他の順番において実行されうることを理解するであろう。また、フロー図 600 および 700 は、図 2 および図 5 に関連して、マネージド APIs を最適化する方法の一例として提供および説明されるのみである。

#### 【0024】

図 6 の例では、フロー図 600 は、P I P S ジェネレータ 235 が下層のプロセッサ 260 のプロセッサ特有の命令群 250 に関連する P I P S 510 を生成する、ブロック 610 から開始する。例えば、P I P S ジェネレータ 235 は、マネージド・ランタイム・アプリケーション 210 のインストール中に、下層のプロセッサ 260 に相当するプロセッサ識別子に基づいて、P I P S 510 を生成してもよい。上述の通り、プロセッサ特有の命令群 250 は、下層のプロセッサ 260 上におけるマネージド・ランタイム・アプリケーション 210 の性能を最適化するために、プロセッサ特有の命令を使用することなくして使用できない、オーディオ処理、ビデオ処理、画像処理、音声認識、暗号などのような、下層のプロセッサ 260 の機能を使用可能にする。ブロック 620 において、P I P S ジェネレータ 235 は、P I P S 510 に基づいて、最適化マネージド APIs 225 を生成する。特に、P I P S ジェネレータ 235 は、P I P S 510 をマネージド・ランタイム・アプリケーション 210 に相当するあるマネージド APIs 220 に挿入する。P I P S ジェネレータ 235 は、下層のプロセッサ 260 においてマネージド・ランタイム・アプリケーション 210 が実行されている間、最適化マネージド APIs 225 が J I T コンパイラ 240 から使用可能となるように、最適化マネージド APIs 225 を保

#### 【0025】

図 7 の例では、フロー図 700 は、J I T コンパイラ 240 がマネージド・ランタイム・アプリケーション 210 に相当する最適化マネージド APIs 225 をコンパイルして実行する、ブロック 710 から開始する。上述のように、P I P S ジェネレータ 235 がプロセッサ特有の命令群 250 に関連する P I P S 510 を最適化マネージド APIs 225 に予め挿入したため、J I T コンパイラ 240 は、最適化マネージド APIs 225 を更に最適化することなく、最適化マネージド APIs 225 をコンパイルしうる。すなわち、P I P S 510 は、最適化マネージド APIs 225 によって、下層のプロセッサ 260 上で実行するためにマネージド・ランタイム・アプリケーション 210 をカスタムハンドで最適化する。ブロック 640 において、J I T コンパイラ 240 は、プロセッサ特有の命令群 250 に相当する下層のプロセッサ 260 の機能を使用可能にする。V M 230 によって提供されるガーベッジコレクション、メモリ管理、およびコードとロールベースのセキュリティのようなサービスに加えて、マネージド・ランタイム・アプリケーション 210 は、下層のプロセッサ 260 上で実行される間、暗号、マルチメディア、オーディオコーデック、ビデオコーデック、画像コーディング、画像処理、信号処理、音声圧縮、コンピュータビジョンなどの最適化マネージド APIs 225 によってマネージド・ランタイム・アプリケーション 210 に提供される、ソフトウェア・ライブラリ関数を使用しうる。その結果、最適化マネージド APIs 225 は、マネージド・ランタイム・アプリケーション 210 が、プロセッサ特有の命令を使用することなくして使用できない、または、他のプロセッサにおいては非効率的である下層のプロセッサ 260 の機能を使用可能にする、プロセッサ特有の命令群 250 を実行することを可能にする。更に、最適化マネージド APIs 225 は、ネイティブ・アセンブリ・コード 500 によって、下層のプロセッサ 260 におけるマネージド・ランタイム・アプリケーション 210 の性能をカスタムハンドに最適化する。

#### 【0026】

本明細書において開示される方法および装置は、欧州コンピュータ・マネージメント・アソシエーション (E C M A) 共通言語インフラストラクチャ (C L I) (second edition, December 2002) および E C M A C # 言語仕様 (second edition, December 2002) の実装のソースコードに適する。しかし、通常の技術を有する当業者は、開示内容が他のラン

10

20

30

40

50

タイム環境におけるソースコードに対しても適用可能であることを理解するであろう。

【0027】

図8は、本明細書において開示される方法および装置の実装に適合するプロセッサシステム1000の一例を示すブロック図である。プロセッサシステム1000は、デスクトップコンピュータ、ラップトップコンピュータ、ノートブックコンピュータ、パーソナルデジタルアシスタント(PDA)、サーバ、インターネットアプライアンス、またはその他いかなるタイプのコンピュータデバイスでもよい。

【0028】

図8に示されるプロセッサシステム1000は、メモリコントローラ1012および入力/出力(I/O)コントローラ1014を含む、チップセット1010を有する。よく知られているように、チップセットは、一般的に、プロセッサ1020からアクセス可能である、または、プロセッサ1020によって使用されるメモリおよびI/O管理機能を提供し、一般的な目的および/または特定の目的のレジスタ、タイマーなども提供する。プロセッサ1020は、1つ以上のプロセッサを使用して実装される。例えば、プロセッサ1020は、Intel Pentiumテクノロジー、Intel Itaniumテクノロジー、Intel Centrino(登録商標)テクノロジーおよび/またはIntel XScale(登録商標)テクノロジーのうちの1つ以上を使用して実装されてもよい。あるいは、他のプロセッシング技術がプロセッサ1020を実装するために使用されてもよい。プロセッサ1020は、第1レベル・ユニファイド・キャッシュ(L1)、第2レベル・ユニファイド・キャッシュ(L2)、第3レベル・ユニファイド・キャッシュ(L3)、および/または通常の技術を有する当業者が認識するその他の適用しうる全ての構造を使用して実装されうる、キャッシュ1022を有する。

10

20

【0029】

従来技術のように、メモリコントローラ1012は、プロセッサ1020がバス1040によって揮発性メモリ1032および非揮発性メモリ1034を含むメインメモリ1030にアクセスして通信できるようにする機能を実行する。揮発性メモリ1032は、同期型ダイナミック・ランダム・アクセス・メモリ(SDRAM)、ダイナミック・ランダム・アクセス・メモリ(DRAM)、RAMBUSダイナミック・ランダム・アクセス・メモリ(RDRAM)、および/またはその他全てのタイプのランダム・アクセス・メモリ・デバイスによって実装されてもよい。非揮発性メモリ1034は、フラッシュメモリ、リード・オンリー・メモリ(ROM)、電氣的消去可能リード・オンリー・メモリ(EEPROM)、および/またはその他全てのタイプのメモリデバイスを使用して実装されてもよい。

30

【0030】

プロセッサシステム1000は、バス1040に接続されたインタフェース回路1050も有する。インタフェース回路1050は、イーサネット・インタフェース、ユニバーサル・シリアル・バス(USB)、第3世代入力/出力インタフェース(3GIO)インタフェース、および/またはその他適用可能な全てのタイプのインタフェースのような、いかなる従来のインタフェースの標準を使用して実装されてもよい。

【0031】

1つ以上の入力デバイス1060は、インタフェース回路1050に接続される。1つまたは複数の入力デバイス1060は、ユーザによるプロセッサ1020に対するデータおよびコマンドの入力を可能にする。例えば、1つまたは複数の入力デバイス1060は、キーボード、マウス、タッチセンシティブディスプレイ、トラックパッド、トラックボール、isopoint、および/または音声認識システムによって実装されてもよい。

40

【0032】

1つ以上の出力デバイス1070もまた、インタフェース回路1050に接続される。例えば、1つまたは複数の出力デバイス1070は、ディスプレイデバイス(例:発光ディスプレイ(LED)、液晶ディスプレイ(LCD)、陰極線管(CRT)ディスプレイ、プリンタ、および/またはスピーカ)によって実装されてもよい。このように、インタ

50

フェース回路 1050 は、一般的には、他のデバイスの中でも特に、グラフィックスドライバカードを有する。

【0033】

プロセッサシステム 1000 は、ソフトウェアおよびデータを保存する 1 つ以上の大容量ストレージデバイス 1080 も有する。そのような 1 つまたは複数の大容量ストレージデバイス 1080 の例は、フロッピーディスクとドライブ、ハードディスクドライブ、コンパクトディスクとドライブ、およびデジタル多用途ディスク (DVD) とドライブを含む。

【0034】

インタフェース回路 1050 は、ネットワークによる外部コンピュータとのデータの交換を促進するモデムまたはネットワークインタフェースカードのような通信デバイスも含む。プロセッサシステム 1000 とネットワーク間の通信リンクは、イーサネット接続、デジタル加入者回線 (DSL)、電話線、携帯電話システム、同軸ケーブルなどのような、いかなるタイプのネットワーク接続でもよい。

【0035】

1 つまたは複数の入力デバイス 1060、1 つまたは複数の出力デバイス 1070、1 つまたは複数の大容量ストレージデバイス 1080、および / またはネットワークへのアクセスは、概して、従来技術の手法に沿って I/O コントローラ 1014 によってコントロールされる。特に、I/O コントローラ 1014 は、プロセッサ 1020 がバス 1040 およびインタフェース回路 1050 によって 1 つまたは複数の入力デバイス 1060、1 つまたは複数の出力デバイス 1070、1 つまたは複数の大容量ストレージデバイス 1080、および / またはネットワークにアクセスできるようにする機能を実行する。

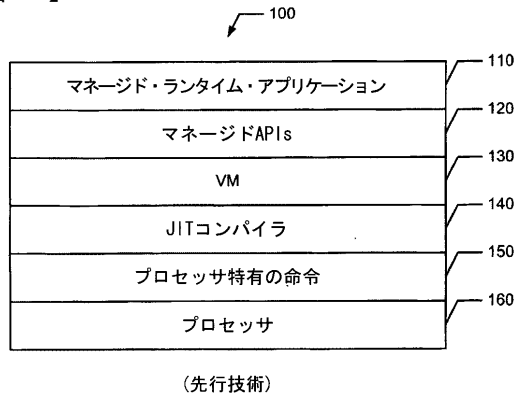
【0036】

図 8 に示されるコンポーネントはプロセッサシステム 1000 内の独立したブロックとして図示されるが、これらのブロックのうちのいくつかによって実行される機能は、1 つの半導体回路内に組み込まれてもよく、または、2 つ以上の独立した集積回路を使用して実装されてもよい。例えば、メモリコントローラ 1012 および I/O コントローラ 1014 は、チップセット 1010 内の独立したブロックとして図示されるが、通常の技術を有する当業者は、メモリコントローラ 1012 および I/O コントローラ 1014 は 1 つの半導体回路に組み込まれうることを理解するであろう。

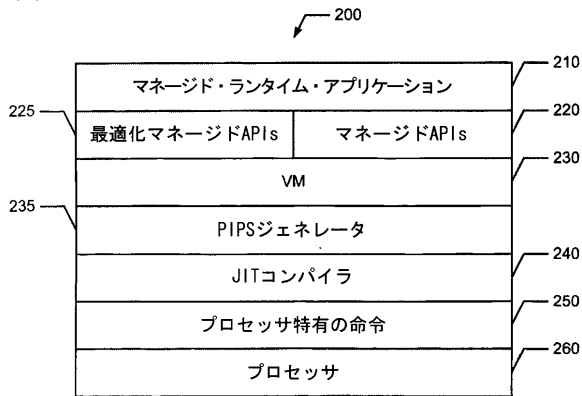
【0037】

本明細書ではある一定の方法、装置、および製品の例が説明されるが、本発明の特許請求の範囲は、これらに限定されない。本発明の範囲は、逐語的にまたは均等論によって添付の特許請求の範囲に収まる全ての方法、装置、製品に及ぶ。例えば、本明細書は、他のコンポーネントの中でもハードウェア上で実行されるソフトウェアまたはファームウェアを含むシステム例を開示するが、これらのシステムは、単に本発明を説明するためのものであって本発明を限定するものではない。特に、全ての開示されたハードウェア、ソフトウェア、および / またはファームウェアコンポーネントは、ハードウェア内に独占的に組み込まれてもよく、ソフトウェアに独占的に組み込まれてもよく、ファームウェアに独占的に組み込まれてもよく、または、ハードウェア、ソフトウェア、および / またはファームウェアの組み合わせに組み込まれてもよいことが意図されている。

【図 1】



【図 2】



【図 4】

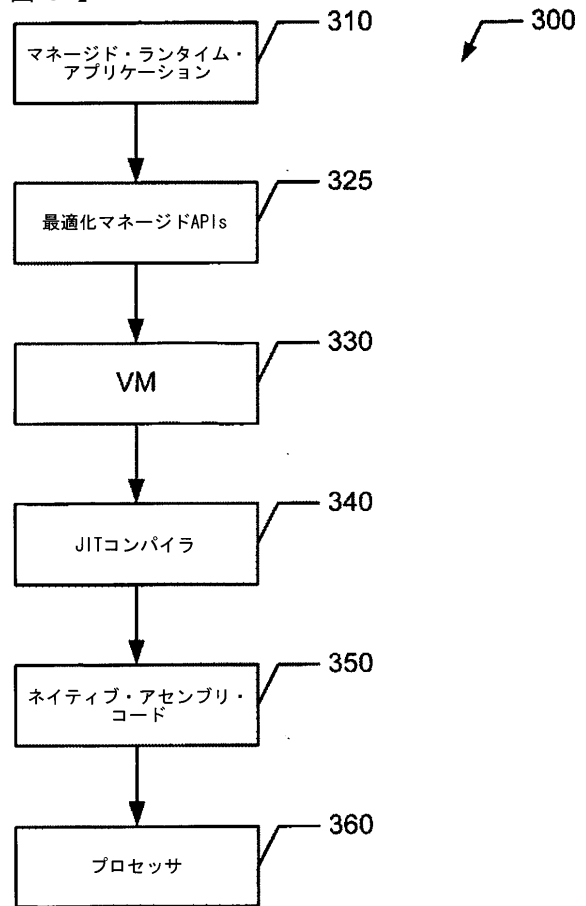
400

```

LppStatus ownsCompare_16u (const lpp8u* pSrc1, const lpp8u* pSrc2, int len, int *pResult
{
    int i;
    for (i = 0; i < len; i++){
        if (pSrc1[i] != pSrc2[i]) break;
    } // for
    *pResult = (i < len) ? pSrc1[i] - pSrc2[i] : 0;
    return lppStsNoErr;
} // end of function

```

【図 3】



【図 5】

500

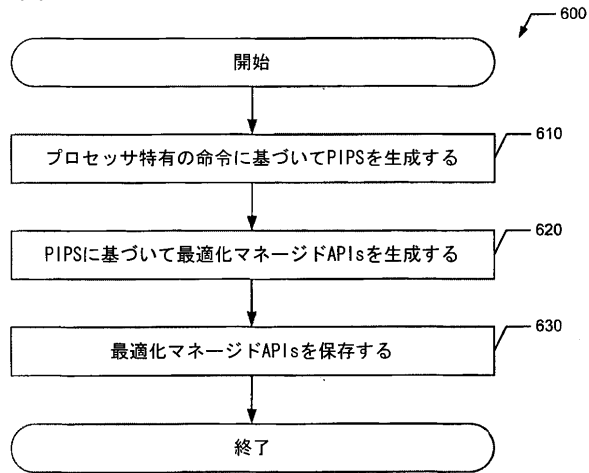
```

Public ownsCompare_16u
_TEXT SEGMENT
pSrc1 EQU 12[esp]
pSrc2 EQU 16[esp]
len EQU 20[esp]
pResult EQU 24[esp]
ALIGN 16
; Lib = W7 (code name for P4 optimization)
; Caller = ippsCompare_16u function
ownsCompare_16u PROC NEAR
    push esi
    push edi
    .....
    mov esi, pSrc1
    mov eax, pSrc2
    mov edi, pSrc2
    mov ecx, len
    test ecx, ecx
    jz ResultCmp16u00
    .....
    xor eax, edi
    and eax, 03h
    jnz ShortLoop4Cmp16u00
    test edi, 01h
    jnz ShortLoop4Cmp16u00
    cmp ecx, 8
    jg Align16Cmp16u00
    .....
    SSE2ResultCmp16u00:
        xor eax, 0ffffh
        bsf edx, eax
        lea esi, [esi + edx]
        movzx eax, WORD PTR [esi]
        movzx edx, WORD PTR [esi + edi]
        sub eax, edx
        jmp ResultCmp16u01
    .....
    AlignResultCmp16u00:
        sub edi, esi
        jmp SSE2ResultCmp16u00
    Result16Cmp16u01:
        add esi, 16
        jmp SSE2ResultCmp16u00
    Result16Cmp16u02:
        add esi, 32
        jmp SSE2ResultCmp16u00
    Result16Cmp16u03:
        add esi, 48
        jmp SSE2ResultCmp16u00
    Result16Cmp16u04:
        add esi, 64
        jmp SSE2ResultCmp16u00
    Result16Cmp16u05:
        add esi, 80
        jmp SSE2ResultCmp16u00
    ownsCompare_16u ENDP
_TEXT ENDS

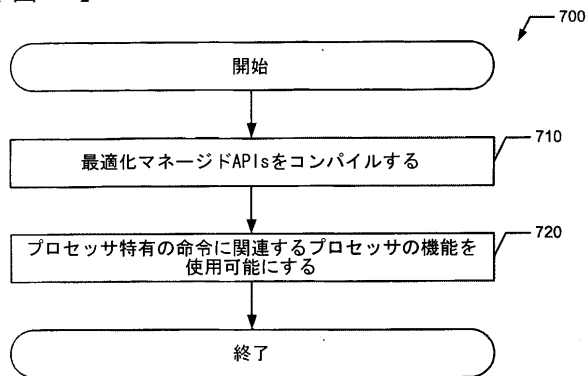
```

510

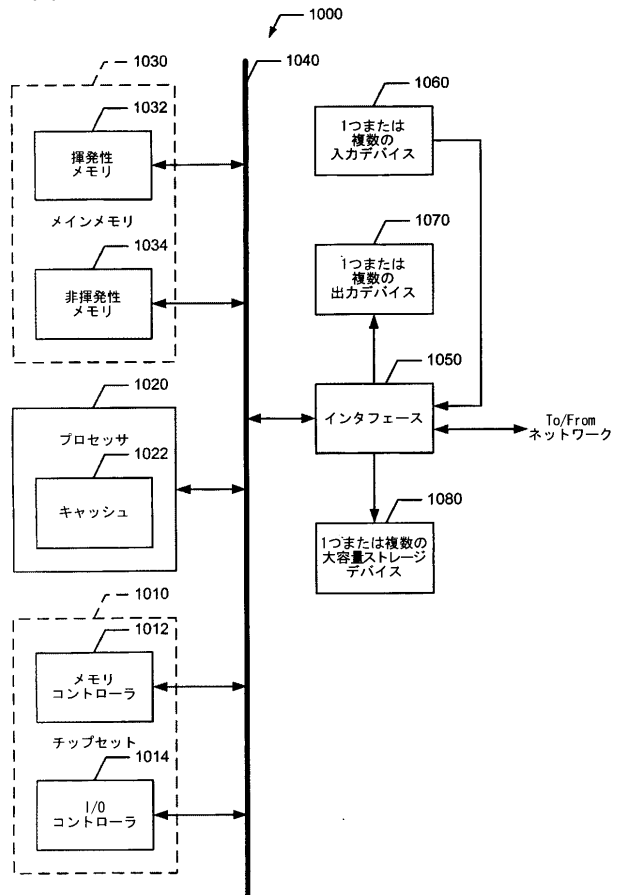
【図 6】



【図 7】



【図 8】



## 【国際調査報告】

## INTERNATIONAL SEARCH REPORT

International Application No.  
PCT/US2005/002989

<b>A. CLASSIFICATION OF SUBJECT MATTER</b> IPC 7 G06F9/45 G06F9/455		
According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) IPC 7 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, WPI Data, PAJ, IBM-TDB, INSPEC, COMPENDEX		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 1 313 012 A (TEXAS INSTRUMENTS FRANCE; TEXAS INSTRUMENTS INCORPORATED) 21 May 2003 (2003-05-21) page 2, paragraph 1 - page 4, paragraph 23 page 4, paragraph 26 - page 5, paragraph 32 page 5, paragraph 36 page 6, paragraph 43 - paragraph 44 page 10, paragraph 51 - paragraph 53 page 10, paragraph 55 - page 11, paragraph 59	1-27
X	EP 0 945 791 A (SUN MICROSYSTEMS, INC) 29 September 1999 (1999-09-29) column 1, paragraph 1 - column 3, paragraph 13 column 2, paragraph 9 column 2, paragraph 11 - column 3	1-27
-/--		
<input checked="" type="checkbox"/> Further documents are listed in the continuation of box C.		
<input checked="" type="checkbox"/> Patent family members are listed in annex.		
<b>* Special categories of cited documents:</b> <div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <p>*A* document defining the general state of the art which is not considered to be of particular relevance</p> <p>*E* earlier document but published on or after the international filing date</p> <p>*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)</p> <p>*O* document referring to an oral disclosure, use, exhibition or other means</p> <p>*P* document published prior to the international filing date but later than the priority date claimed</p> </div> <div style="width: 48%;"> <p>*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</p> <p>*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</p> <p>*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.</p> <p>*G* document member of the same patent family</p> </div> </div>		
Date of the actual completion of the international search		Date of mailing of the international search report
30 June 2005		02/08/2005
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016		Authorized officer  Lo Turco, S

## INTERNATIONAL SEARCH REPORT

International Application No  
PCT/US2005/002989

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2003/070161 A1 (WONG HINKMOND ET AL) 10 April 2003 (2003-04-10) page 1, paragraph 3 - page 2, paragraph 17 -----	1-27
A	US 6 332 215 B1 (PATEL MUKESH K ET AL) 18 December 2001 (2001-12-18) column 1, line 1 - column 2, line 27 -----	1-27
A	WO 02/41145 A (ZUCOTTO WIRELESS INC; BOTTOMLEY, THOMAS, MARK, WALTER) 23 May 2002 (2002-05-23) page 1, line 1 - page 5, line 13 -----	1-27

## INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No.

PCT/US2005/002989

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 1313012 A	21-05-2003	EP 1313012 A1	21-05-2003
		US 2003101208 A1	29-05-2003
EP 0945791 A	29-09-1999	US 5999732 A	07-12-1999
		CN 1234553 A ,C	10-11-1999
		EP 0945791 A2	29-09-1999
		JP 11327916 A	30-11-1999
US 2003070161 A1	10-04-2003	EP 1451682 A2	01-09-2004
		WO 03032155 A2	17-04-2003
US 6332215 B1	18-12-2001	AU 2165400 A	26-06-2000
		EP 1157323 A2	28-11-2001
		JP 2002532772 T	02-10-2002
		WO 0034844 A2	15-06-2000
		US 6338160 B1	08-01-2002
		US 2002019976 A1	14-02-2002
		US 2002066083 A1	30-05-2002
WO 0241145 A	23-05-2002	AU 2039502 A	27-05-2002
		AU 3044502 A	03-06-2002
		AU 3047302 A	03-06-2002
		AU 4150502 A	11-06-2002
		AU 4150702 A	24-06-2002
		AU 7832901 A	27-05-2002
		WO 0241145 A2	23-05-2002
		WO 0241143 A2	23-05-2002
		WO 0242898 A2	30-05-2002
		WO 0245385 A2	06-06-2002
		WO 0248864 A2	20-06-2002
		WO 02071211 A2	12-09-2002
		WO 0242911 A2	30-05-2002
		US 2004015912 A1	22-01-2004



---

フロントページの続き

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

(特許庁注：以下のものは登録商標)

- 1．フロッピー
- 2．イーサネット