(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第5911952号
(P5911952)

(45)発行日　平成28年4月27日(2016.4.27)　(24)登録日　平成28年4月8日(2016.4.8)

(51)Int.Cl.		F I		
G06F 9/318	(2006.01)	G06F	9/30	320C
G06F 12/08	(2016.01)	G06F	12/08	509Z
G06F 12/10	(2016.01)	G06F	12/10	501Z
G06F 9/30	(2006.01)	G06F	9/30	310A
		G06F	9/30	310C

請求項の数 17　(全 31 頁)　最終頁に続く

(21)出願番号	特願2014-509420 (P2014-509420)
(86)(22)出願日	平成24年5月2日(2012.5.2)
(65)公表番号	特表2014-513366 (P2014-513366A)
(43)公表日	平成26年5月29日(2014.5.29)
(86)国際出願番号	PCT/US2012/036199
(87)国際公開番号	WO2012/151334
(87)国際公開日	平成24年11月8日(2012.11.8)
審査請求日	平成25年11月25日(2013.11.25)
(31)優先権主張番号	13/099,463
(32)優先日	平成23年5月3日(2011.5.3)
(33)優先権主張国	米国(US)

(73)特許権者　507364838
クアルコム，インコーポレイテッド
アメリカ合衆国　カリフォルニア　９２１２１　サン　ディエゴ　モアハウス　ドライブ　５７７５
(74)代理人　100108453
弁理士　村山　靖彦
(74)代理人　100163522
弁理士　黒田　晋平
(72)発明者　セルゲイ・ラリン
アメリカ合衆国・カリフォルニア・９２１２１・サン・ディエゴ・モアハウス・ドライヴ・５７７５

最終頁に続く

(54)【発明の名称】メモリ階層内に組み込まれたエントロピーエンコーディングされたソフトウェアの記憶および変換のための方法および装置

(57)【特許請求の範囲】

【請求項１】

圧縮命令を解凍するための方法であって、

メモリ(542、544)から圧縮命令をフェッチするステップであって、各々の圧縮命令(554)がX-インデックスおよびY-インデックスを含む、ステップと、

各々の圧縮命令のために、２行アクセスのためにマスクされた前記X-インデックス(558)を使用してXメモリ(562)からX-ビットパターン(506、576)の２行をフェッチし、２行アクセスのためにマスクされた前記Y-インデックス(560)を使用してYメモリ(564)からY-ビットパターン(508、578)の２行をフェッチするステップであって、パターンの前記２行は、２行にまたがるパターンにアクセスするためにフェッチされ、

前記X-インデックスに従って前記フェッチされたX-ビットパターンの２行からX-ビットパターンを選択するステップ、および前記Y-インデックスに従って前記フェッチされたY-ビットパターンの２行からY-ビットパターンを選択するステップと、

前記X-ビットパターンのビットを、各々の圧縮命令と関連付けられるミックスマスク(504、574)に基づいて前記Y-ビットパターンのビットと組み合わせて、解凍命令(502)にするステップであって、前記解凍命令のフォーマット長が、前記関連付けられたミックスマスクによって決定され、各々の圧縮命令に関連付けられた前記ミックスマスクが、各々の圧縮命令のアドレスで変換索引バッファ（TLB）エントリーに記憶された属性から識別され、前記TLBエントリーは、仮想アドレスを物理アドレスに変換するために使用される、ステップと、を特徴とする、方法。

【請求項２】
前記解凍命令の前記フォーマット長が、複数の異なるフォーマット長の1つである、請求項1に記載の方法。
【請求項３】
前記メモリがメインメモリおよびレベル2キャッシュを含む、請求項1に記載の方法。
【請求項４】
レベル1命令キャッシュにおけるミスに応答して、レベル2キャッシュから前記圧縮命令をフェッチするステップをさらに含む、請求項1に記載の方法。
【請求項５】
前記レベル1命令キャッシュに、各々の解凍命令を記憶するステップをさらに含む、請求項4に記載の方法。
【請求項６】
各々の圧縮命令が、ミックスマスクインデックスと関連付けられる、請求項1に記載の方法。
【請求項７】
各々の圧縮命令と関連付けられる前記ミックスマスクが、前記ミックスマスクインデックスを使用してミックスマスクメモリから選択される、請求項6に記載の方法。
【請求項８】
仮想メモリシステムに記憶された圧縮命令を変換するための装置であって、
非圧縮命令(606)のページと混合された、異なるミックスマスクに基づいて圧縮されたミックスマスクに基づく圧縮命令(608)のページを記憶するように構成される、ページ化された命令キャッシュ(600)であって、ミックスマスクは、選択されたミックスマスクを使用して達成される圧縮の単位に基づいて、前記ページ化された命令キャッシュに圧縮される命令の、それぞれのページのための複数のミックスマスクから選択される、ページ化された命令キャッシュと、
仮想アドレスを物理アドレスに変換し、圧縮命令を記憶するものとして前記ページ化された命令キャッシュ中のページを識別する、ミックスマスクインデックス(616)を記憶し、前記識別されたページを圧縮するために使用される前記選択されたミックスマスクを識別するように構成され、前記識別されたミックスマスクは、圧縮命令の前記識別されたページからフェッチされる各々の圧縮命令を解凍するために使用される、命令変換索引バッファ(TLB)(602)と、を特徴とする、装置。
【請求項９】
1つまたは複数のミックスマスクを記憶し、前記ミックスマスクを選択するために前記ミックスマスクインデックスによってアドレス可能である、ミックスマスクメモリをさらに含む、請求項8に記載の装置。
【請求項１０】
各々のミックスマスクに基づく圧縮命令が、非圧縮命令を生成するための、前記識別されたミックスマスクに従って組み合わされる第1のビットパターンおよび第2のビットパターンからなる、請求項8に記載の装置。
【請求項１１】
圧縮命令を解凍するように構成される、変換デコーダであって、前記ページ化された命令キャッシュと、非圧縮命令および解凍命令を記憶するように構成されたより低水準の命令キャッシュとの間に配置される、変換デコーダをさらに含む、請求項8に記載の装置。
【請求項１２】
前記変換デコーダが、
圧縮命令を解凍するように構成される、圧縮命令デコーダと、
前記より低水準の命令キャッシュへの記憶に備えて、前記解凍命令を記憶するように構成される、解凍バッファとを含む、請求項11に記載の装置。
【請求項１３】
前記変換デコーダからの解凍命令をプロセッサパイプラインに直接渡すように構成され

る、バイパス回路をさらに含む、請求項11に記載の装置。
【請求項１４】
圧縮命令を実行可能なフォーマットに変換するためのシステムであって、
プログラムコードの各々の圧縮命令(554)のためのX-インデックスおよびY-インデックスを記憶するように構成される、レベル2（L2）キャッシュ(544)と、
複数のエントリーを含む変換索引バッファ（TLB）であって、各々のエントリーは、仮想アドレスを物理アドレスに変換し、命令のページが非圧縮命令または圧縮命令を含むか否かを識別し、次いで、前記命令のページが圧縮命令のページであるとき、複数のミックスマスクから選択されるミックスマスクを識別するために使用され、前記識別されたミックスマスクは、圧縮命令の前記ページを生成するための圧縮動作に使用される、変換索引バッファと、
前記L2キャッシュの圧縮命令のページから圧縮命令を受け取って、各々の受け取られた圧縮命令に対して、前記X-インデックス(558)および前記Y-インデックス(560)を使用して変換メモリ(562、564)からX-ビットパターン(506、576)およびY-ビットパターン(508、578)を選択し、前記ミックスマスクを識別する前記TLBから前記エントリーを使用してミックスマスクメモリから前記識別されたミックスマスク(504、574)をフェッチし、前記識別されたミックスマスクは、前記選択されたX-ビットパターンのビットと前記Y-ビットパターンのビットで組み合わせてネイティブな命令フォーマット(502)にするために使用されるように、構成される、変換ユニット(546)と、
各々の圧縮命令のための前記ネイティブな命令フォーマットを記憶するように構成される、レベル1キャッシュ(548)と、を特徴とする、システム。
【請求項１５】
前記レベル2(L2)キャッシュが、ネイティブな命令フォーマットで命令を記憶するページの第1のセットと、前記圧縮命令を記憶するページの第2のセットへと分割される、請求項14に記載のシステム。
【請求項１６】
圧縮命令を格納する前記L2キャッシュにおいてページを識別するエントリーを含む、命令索引バッファをさらに含む、請求項15に記載のシステム。
【請求項１７】
プログラムコードの各々の圧縮命令のための前記X-インデックスおよび前記Y-インデックスの前記L2キャッシュに記憶される前記プログラムコードが、前記プログラムコードの暗号化されたフォームである、請求項14に記載のシステム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は全般に、組込みソフトウェアにおけるコード密度を向上させるための圧縮命令セットを有するプロセッサに関し、より具体的には、圧縮命令を生成し、圧縮命令を記憶し、圧縮命令を変換するための技法に関する。
【背景技術】
【０００２】
携帯電話、ラップトップコンピュータ、携帯情報端末(PDA)などの多くのポータブル製品は、通信およびマルチメディアのアプリケーションをサポートするプログラムを実行するプロセッサの使用を必要とする。そのような製品の処理システムは、それぞれが命令、入力オペランド、および実行の結果のための記憶装置を有する、1つまたは複数のプロセッサを含む。たとえば、プロセッサのための命令、入力オペランド、および実行の結果は、汎用レジスタファイル、マルチレベル命令キャッシュ、データキャッシュ、およびシステムメモリからなる、階層メモリサブシステムに記憶され得る。
【発明の概要】
【発明が解決しようとする課題】
【０００３】

高いコード密度を実現するために、32ビット命令フォーマットのサブセットである16ビット命令フォーマットのような、2つの命令フォーマットを有するネイティブな命令セットアーキテクチャ(ISA)が使用され得る。多くの場合、フェッチされた16ビット命令は、復号プロセスの前に、または復号プロセスにおいて、32ビット命令へとプロセッサによって変換され、これによって、実行ハードウェアが、32ビット命令フォーマットのみをサポートするように設計されることが可能になる。32ビット命令のサブセットである16ビット命令を使用することは、16ビットフォーマットへとエンコーディングされ得る情報の量を制限する、制約である。たとえば、16ビット命令フォーマットは、アドレス可能なソースオペランドレジスタと、指定され得る宛先レジスタとの数を制限し得る。たとえば、16ビット命令フォーマットは、3ビットまたは4ビットのレジスタファイルアドレスフィールドを使用し得るが、32ビット命令は5ビットのフィールドを使用し得る。2つのフォーマットがプログラムにおいて混合すると、一部には、16ビット命令および32ビット命令への分岐のような命令アドレッシングの制約が原因で、プロセッサパイプラインの複雑さも増し得る。また、コード圧縮の要件は、プログラムに応じて変わり、あるプログラムに対して選ばれた固定の16ビット命令フォーマットが、異なるプログラムによる使用においてはより不利になる。この点で、既存のプロセッサのレガシーコードは、2つの命令フォーマットを効果的に利用し、コード密度を大きく向上させることが不可能であり、多くの場合、リアルタイムの要件を満たすことが不可能であり得る。これらの制約および他の制約が、標準的なサイズの命令において使用されるフィールドのサブセットであるフィールドを有する、サイズが低減された命令の有効性を制限する。

【課題を解決するための手段】

【０００４】

いくつかの態様のうち、本発明は、ネイティブな命令セットを使用したプログラムを、圧縮命令セットを使用した圧縮されたプログラムから分離する必要性に対処する。本明細書で取り上げられる技法は、ソフトウェアの実行時間および効率に影響を与えることなく、組込みソフトウェアのための記憶装置および送信経路の高度に効率的な利用を可能にする。

【０００５】

そのような目標のために、本発明の実施形態は、圧縮可変長命令を解凍するための方法を適用する。圧縮固定長命令は、圧縮可変長命令を記憶するように構成されたシステムメモリからフェッチされ、各々の圧縮固定長命令は、X-インデックスおよびY-インデックスを含む。各々の圧縮固定長命令のために、X-ビットパターンがX-インデックスを使用してXメモリからフェッチされ、Y-ビットパターンがY-インデックスを使用してYメモリからフェッチされる。X-ビットパターンは、各々の圧縮固定長命令と関連付けられるミックスマスクに基づいてY-ビットパターンと組み合わされて、解凍可変長命令となり、解凍可変長命令のフォーマット長は、関連付けられたミックスマスクによって決定される。

【０００６】

本発明の別の実施形態は、仮想メモリシステムに記憶された、圧縮命令を変換するための装置を取り上げる。ページ化された命令キャッシュは、非圧縮命令のページと混合された、圧縮命令のページを記憶するように構成される。命令変換索引バッファ(TLB)は、圧縮命令を記憶するものとしてページ化されたキャッシュ中のページを識別する、アドレス変換エントリーを記憶するように構成される。

【０００７】

本発明の別の実施形態は、圧縮命令を実行可能なフォーマットを変換するためのシステムを取り上げる。レベル2キャッシュは、各々の圧縮命令のためのX-インデックスおよびY-インデックスを記憶するように構成される。変換ユニットは、レベル2キャッシュから圧縮命令を受け取って、各々の受け取られた圧縮命令に対して、X-インデックスおよびY-インデックスを使用して変換メモリからX-ビットパターンおよびY-ビットパターンを選択し、選択されたX-ビットパターンとY-ビットパターンを組み合わせてネイティブな命令フォーマットにするための、プログラム指定のミックスマスクを使用するように構成される。

レベル1キャッシュは、各々の圧縮命令のためのネイティブな命令フォーマットを記憶するように構成される。

【０００８】

本発明の別の実施形態は、圧縮命令を効率的に変換するためのミックスマスクを決定する方法を取り上げる。ミックスマスクの種子集団からの遺伝子と表現されるミックスマスクのペアが交配され、子ミックスマスクのペアを生成する。子ミックスマスクは、種子集団を更新する変異した子ミックスマスクを生成するように、変異させられる。ミックスマスクは、高レベルの圧縮を提供する更新された種子集団から決定され、ビットのパターンは、決定されたミックスマスクに従って組み合わされて、プログラムの圧縮命令を実行可能な形式へと変換する。

【０００９】

本発明のさらに完全な理解と、本発明のさらなる特徴および利点とは、以下の発明を実施するための形態と添付図面から明らかとなろう。

【図面の簡単な説明】

【００１０】

【図１】本発明の実施形態が有利に利用され得る例示的なワイヤレス通信システムのブロック図である。

【図２】本発明によるランタイム解凍のために設計されたコード圧縮のためのシステムである。

【図３】本発明による、ミックスマスクに基づいてバイト重複パッドビットを伴うXパターンおよびYパターンへと命令を分割する、命令分割プロセスの例示的な要素を示す図である。

【図４】本発明による、圧縮形式でメインメモリに記憶されるプログラムと、レベル1キャッシュに記憶されるように解凍されるレベル2キャッシュとを有する、解凍器システムである。

【図５Ａ】本発明による、第2のミックスマスクに基づいてバイト重複パッドビットを伴う第2のXパターンおよび第2のYパターンへと第2の命令を分割する、第2の命令分割プロセスの例示的な要素を示す図である。

【図５Ｂ】本発明による、3つの異なるマスクAmm≠Bmm≠Cmmを有するXメモリおよびYメモリにおける、図3のAx、Ayパターン、および図5AのBx、Byパターンの例示的な記憶構成を示す図である。

【図５Ｃ】本発明による、圧縮形式でメインメモリに記憶されるプログラムと、非圧縮形式でレベル1キャッシュに記憶されるように複数のミックスマスクおよびインデックス圧縮を使用して解凍されるレベル2キャッシュとを有する、解凍器システムである。

【図６】本発明による、VLIWパケット圧縮フォーマットを示す図である。

【図７】本発明による、ページ化された命令変換索引バッファ(ITLB)と、圧縮ページおよび非圧縮ページを有するメモリ編成とを示す。

【図８Ａ】本発明による、圧縮命令ページを解凍し、非圧縮命令ページにアクセスするための、ページ化された解凍器システムを示す。

【図８Ｂ】プロセッサパイプライン上で実行される、L2キャッシュの圧縮ページ命令およびL1命令キャッシュの解凍命令の状態を示す、例示的な解凍状態図を示す。

【図９Ａ】本発明による、遺伝的アルゴリズムの変動に基づく、遺伝的なミックスマスク決定プロセスを示す図である。

【図９Ｂ】本発明による、遺伝的アルゴリズムの変動に基づく、遺伝的なミックスマスク決定プロセスを示す図である。

【図９Ｃ】本発明による、遺伝的アルゴリズムの変動に基づく、遺伝的なミックスマスク決定プロセスを示す図である。

【図９Ｄ】本発明による、交差アルゴリズムの実装形態を示す図である。

【図９Ｅ】本発明による、変異アルゴリズムの実装形態を示す図である。

【図１０】本発明による、例示的な現在のコストのプロセスを示す図である。

【図１１】本発明による、重み付けられたハミングヒューリスティックを示す図である。
【図１２】本発明による、パッド並べ替えを伴う網羅的な探索によるX/Yテーブル圧縮プロセスを示す図である。
【図１３】本発明による、圧縮された記憶領域へのシンボル挿入のプロセスを示す図である。
【図１４】本発明による、パッド並べ替えを伴う三者間のX/Yテーブル圧縮プロセスを示す図である。
【発明を実施するための形態】
【００１１】
次に、本発明のいくつかの実施例が示されている添付図面を参照して、本発明がさらに十分に説明される。しかしながら、本発明は、様々な形態で具現化されてもよく、本明細書に明記された実施形態に限定されるように解釈されるべきではない。むしろ、これらの実施態様は、本開示が徹底的で完全なものとなり、本発明の範囲を当業者に十分に伝えることになるように提供されている。
【００１２】
本発明の教示に基づいて動作するか、もしくは本発明の教示に従って動作を実行するための、コンピュータプログラムコードまたは「プログラムコード」は、C、C++、JAVA(登録商標)、SmallTalk、JavaScript(登録商標)、Visual Basic(登録商標)、TSQL、Perlなどの高レベルのプログラミング言語、または、様々な他のプログラミング言語で最初は書かれてもよい。これらの言語の1つで書かれたソースプログラムまたはソースコードは、ネイティブな命令フォーマットへとエンコーディングされる命令を使用して、高レベルのプログラムコードをネイティブなアセンブラプログラムに変換することによって、ターゲットプロセッサのアーキテクチャにコンパイルされる。また、ターゲットプロセッサのアーキテクチャのためのプログラムは、ネイティブなアセンブラ言語で直接書かれてもよい。ネイティブなアセンブラプログラムは、マシンレベルのバイナリ命令において、命令のニーモニック表現を使用する。本明細書で使用される場合、コンパイラまたは人間のプログラマによって生成されるプログラムコードまたはコンピュータ可読媒体は、プロセッサがフォーマットを理解できる、オブジェクトコードのようなマシン言語コードを指す。
【００１３】
図1は、本発明の実施形態が有利に利用され得る例示的なワイヤレス通信システム100を示す。説明のために、図1は、3つの遠隔ユニット120、130、および150、ならびに2つの基地局140を示す。一般的なワイヤレス通信システムは、より多くの遠隔ユニットおよび基地局を有し得ることが認識されよう。コンポーネント125A、125C、125B、および125Dでそれぞれ表されるハードウェアコンポーネント、ソフトウェアコンポーネント、またはその両方を含む遠隔ユニット120、130、150、および基地局140は、さらに後で論じるように、本発明を具現化するように適合されている。図1は、基地局140から遠隔ユニット120、130、および150への順方向リンク信号180、ならびに、遠隔ユニット120、130、および150から基地局140への逆方向リンク信号190を示している。
【００１４】
図1では、遠隔ユニット120は携帯電話として示され、遠隔ユニット130はポータブルコンピュータとして示され、遠隔ユニット150は、ワイヤレスローカルループシステムにおける固定位置遠隔ユニットとして示されている。例として、遠隔ユニットは、代替的には、セル式電話、ページャ、ウォーキートーキー、ハンドヘルド個人通信システム(PCS)ユニット、携帯情報端末などのポータブルデータユニット、またはメータ読取り機器などの固定位置データユニットでもよい。図1は、本開示の教示による遠隔ユニットを示すが、本開示は、これらの例示的な示されたユニットには限定されない。本発明の実施形態は、任意のプロセッサシステムにおいて好適に利用され得る。
【００１５】
図2は、本発明による、ランタイム解凍のために設計されたコード圧縮のための圧縮システム200である。圧縮システム200は、上で説明されたようなソースコードと、ソースコ

ード中の非圧縮形式のバイナリライブラリファイルと、コンパイルされている現在のプログラムアプリケーションを含むライブラリファイル204とを含む。圧縮システム200はまた、コンパイラおよびリンカ206と、任意選択のプロファイルフィードバック情報208とを含み、プロファイルフィードバック情報208は、ネイティブな命令セットアーキテクチャ(ISA)フォーマットおよびサポートするデータセクション212に基づいて、リンクされる実行可能コード210を生成するために使用される。ネイティブなISAは、固定の非圧縮フォーマットによって表され、たとえば、固定の64ビット、32ビット、または16ビットのエンコーディング、およびそのようなエンコーディングの混合を含む、種々の手法を表し得る。ネイティブなISAは、一般の用途のために開発され、手元の現在のアプリケーションのために特別に調整されていない。32ビット命令の語境界のような、固定の語境界を維持することによって、分岐、呼び出し、戻りなどのための固定の語アドレスのみをサポートするアドレッシングモデルなどが、16ビット命令および32ビット命令がコードにおいて一緒に混合し得る場合でも、使用され得る。

【００１６】

コードのアドレス可能性を維持し、高速でレイテンシが固定の解凍時間を保証しつつ、ISAなどから選択された命令は、圧縮され、現在のアプリケーションに対して調整され得る。そのような圧縮は、線形時間において圧縮コードを生成するように自動化され得る。元のISAは、その非常に一般化された性質によって、情報エントロピーが低く、情報エントロピーは、所与のソースコードおよびライブラリファイル204のための、エントロピー限界に達した専用のエンコーディングを生成することによって、本発明に従って増大する。リンクされた実行可能コード210は、圧縮コード218および復号テーブル220を生成する、変換ツール216への入力として提供される。圧縮コード218およびサポートするデータセクション212は、ハードディスク、光ディスク、組み込まれたデバイスのフラッシュメモリ、または、選択されたコードが実行のためにそこからプロセッサ複合体203へとダウンロードされ得る他のそのような記憶媒体のような、スタティック型の記憶デバイス214に記憶される。プロセッサ複合体203は、メインメモリ222、レベル2キャッシュ(L2キャッシュ)、およびプロセッサコア226を含む。プロセッサコア226は、本発明による変換メモリ(TM)230を有するデコーダ228、レベル1命令キャッシュ(L1命令キャッシュ)232、および実行パイプライン234を含む。圧縮コードは、スタティック型の記憶デバイス214、メインメモリ222、およびL2キャッシュ224に記憶される。解凍コードは、L1キャッシュに記憶され、実行パイプライン234によって実行される。圧縮コード218を生成し、圧縮命令をデコーダ228において復号するための、変換ツール216の様々な実施形態が、以下でより詳しく説明される。

【００１７】

プロセッサ複合体203は、L1命令キャッシュ232に非圧縮形式で記憶され、L2キャッシュ224およびメインメモリ222に圧縮形式で記憶される、プログラムコードを実行するために、図1のハードウェアコンポーネント125A～125Dにおいて好適に利用され得る。プロセッサ複合体に接続され得る周辺デバイスは、議論を簡潔にするために図示されていない。プロセッサコア226は、汎用プロセッサ、デジタル信号プロセッサ(DSP)、特定用途向けプロセッサ(ASP)などであってよい。プロセッサ複合体203の様々なコンポーネントは、特定用途向け集積回路(ASIC)技術、フィールドプログラマブルゲートアレイ(FPGA)技術、または他のプログラマブルロジック、個別ゲートもしくはトランジスタロジック、または意図される用途に適した任意の他の利用可能な技術を使用して実装され得る。シングルプロセッサコア226が示されるが、本発明の圧縮命令の処理は、スーパースカラー設計、ならびに、マルチスレッド設計、マルチコア設計、および超長命令語(VLIW)設計のような並列パイプラインを実装する他のアーキテクチャに、適用可能である。

【００１８】

図3は、本発明による、バイナリミックスマスク(MM)304に基づいて重複パッドビット310および312を伴うAxパターン306およびAyパターン308へと元のISA不変サイズ命令A 302を分割する、命令分割プロセスの例示的な要素300を示す。パッドビットは、命令およびデ

ータを少なくともバイト粒度のセグメントにおいて表すために現在のメモリシステムに課される要件が原因で、生成される。バイト粒度のセグメントを有するフォーマットの使用は、ビット粒度における記憶の重複を許容しつつ、記憶システムのバイト粒度の要件を満たす、新規な圧縮された表現を提供するために利用される。

【００１９】

元のISAコードセグメントを圧縮するために、コードセグメントは、命令のグループへと分割され、各グループは、単一の共有されるXパターンと固有のYパターンのセットとに寄与する。Axパターン306は、命令Aが属する命令のグループに共通のビットパターンを表す。Ayパターン308は、命令の同じグループにおける、固有のビットパターンのセットの1つを表す。コードセグメントは、1とNの間の任意の数のグループへと分割されてよく、Nはコードセグメントにおける元の命令の数であることに留意されたい。グループの最適な数または最適に近い数と、最適なまたは最適に近いミックスマスクとを決定するプロセスは、以下でさらに説明される重要なプロセスである。コードセグメントのXパターンは、Xメモリからなる Xディクショナリに記憶され、コードセグメントのAyパターンは、Yメモリからなる Yディクショナリに記憶される。Xインデックスは、Xメモリの中の位置のアドレスであり、Yインデックスは、Yメモリの中の位置のアドレスである。これらの2つのインデックスの組合せ、XディクショナリおよびYディクショナリからのパターン、ならびに、バイナリミックスマスクが、元の命令を確定的に表す。バイトアドレス可能なX/Yディクショナリパターンは、圧縮形式でX/Yメモリに記憶されるが、可変長の解凍なしでアクセス可能である。可変長の解凍は、可変長の圧縮シンボルをネイティブな不変サイズのアルファベットへ逆マッピングするプロセスに基づく。不変サイズのインデックス圧縮が、この表現のために使用され、後で論じられる。圧縮動作314は、コードセグメントに対して少なくとも1つのミックスマスクを使用し、命令302からAxパターン306およびAyパターン308を選択する。次の例では、16進数の数すなわちHex数が、「0x」というプレフィックスとともに表される。たとえば、命令302 [0x9F6D0121]は、ミックスマスク304 [0xFF80FF80]と組み合わされて、Axパターン306 [0x9F00(8,9,A,B)]およびAyパターン308 [0xDA8(4,5,6,7)]を生成する。デコーダ228の解凍動作316は、コードセグメントのための少なくとも1つのミックスマスク、Xインデックスによりフェッチされる Xメモリパターン、およびYインデックスによりフェッチされる Yメモリパターンを使用して、圧縮命令を解凍する。たとえば、ミックスマスク304 [0xFF80FF80]は、Xメモリからフェッチされた Axパターン306 [0x9F00(8,9,A,B)]およびYメモリからフェッチされた Ayパターン308 [0xDA8(4,5,6,7)]と組み合わされて、元の命令302[0x9F6D0121]を生成する。

【００２０】

上で説明されたように、XパターンおよびYパターンは、バイトアドレス可能なXメモリおよびバイトアドレス可能なYメモリに記憶される。Xパターン、Yパターン、またはこれらの両方のインデックス圧縮は、重複を除去し、物理的な記憶装置の二重利用を減らし、重複パッドビット310および312のようなパッドビットとバイトアドレス可能な位置とを重複させるプロセスを、利用する。第1のXパターンは、第1のXパターンの終了ビットを第2のXパターンの開始ビットと共有することによって、第2のXパターンと重複し得る。この比較において、重複したパッドビットを有するバイトは、図3に示されるように、さらなる範囲の選択肢をもたらす。たとえば、Axパターン306は、第1のバイト0x9Fと、第2のバイト0x00と、重複パッドビット310が原因でセット{8,9,A,B}のうちの1つの値を有する第1のニブルおよび0x00以上0xFF以下の任意の数であり得る第2のニブルからなる第3のバイトとを含む、3バイトからなる。第1のAxパターン306と重複することになる第2のXパターンは、開始ビットの第1のニブルにセット{8,9,A,B}のうちの1つを有し、開始ビットの第2のニブルに0x00以上0xFF以下の任意の数を有する。第1のAxパターン306と重複することになる第3のXパターンの別の可能な重複パターンは、開始ビットの第1のバイトに0x00を有し、開始ビットの次のニブルにセット{8，9，A，B}のうちの1つを有し、開始ビットの次の連続するニブルに0x00以上0xFF以下の任意の数を有する。Yパターンでは、Ayパターン308は、第1のバイト0xDAと第2のバイトとを含む2バイトからなり、第2のバイトは、第1のニ

ブル0x8と、重複パッドビット312が原因でセット{4，5，6，7}のうちの1つであり得る第2のニブルとからなる。そのような能力によって、128ビットラインに4個の非圧縮命令を記憶することが可能であり、パディングおよび重複を伴う32ビットから16ビットへの圧縮では、同じ空間に9個以上の圧縮命令を記憶することが可能である。たとえば、圧縮命令のうちの4個がバイト重複を伴って記憶され得る場合、10個の16ビットの圧縮命令が、128ビットラインに記憶され得る。したがって、アドレッシング空間が圧縮され、インデックスアドレッシングも、パディングを伴わない手法と比較して圧縮される。第1のAyパターン308と重複することになる第2のYパターンは、開始ビットの第1のニブルに「0x8」を有し、セット{4，5，6，7}のうちの1つである第2のニブルを有する。単一のミックスマスクでは、すべてのXパターンは同じ数のビットであり、すべてのYパターンは同じ数のビットである。XパターンおよびYパターンの数が多いと、わずかなマッピングが最小限の記憶領域を占めるだけで、種々のマッピングを有することが可能であり、これは、最適に近い、または許容可能なマッピングであると考えられる。最適なマッピング、または最適に近いマッピングの選択は、NP完全問題であり、任意の多数のXパターンおよびYパターンに対して現実的に解くことができない。それでも、本発明は、線形時間において許容可能なマッピングを生成する、ヒューリスティックを使用する。

【００２１】

図4は、本発明による、圧縮形式でメインメモリ222に記憶されるプログラムと、L1命令キャッシュ232に記憶されるように解凍されるL2キャッシュ224とを有する、解凍器システム400である。L2キャッシュ224は、0x54という7ビットのXインデックス値と0x734という9ビットのYインデックス値とを有するXYエントリー404のような、XインデックスとYインデックスのペアをアドレス可能な位置に記憶する、XYインデックスメモリ402を含む。マルチプレクサ405は、L2キャッシュヒット406があるとXYエントリーを選択し、または、L2キャッシュにおいてミスがあるとメインメモリ222からXY値407を選択するために使用される。解凍動作は、インデックスXレジスタ408、インデックスYレジスタ410、Xメモリ412、Yメモリ414、単一のMMレジスタ416、および結合器418を有する、デコーダ228において遂行される。L1命令キャッシュ232は、非圧縮命令を保持するキャッシュライン420のような、複数のキャッシュラインを含む。

【００２２】

プログラムをロードするプロセス、または組込みシステムのブートプロセスにおいて、メインメモリ222は圧縮コードをロードされ、Xメモリ412およびYメモリ414は関連するXディクショナリコンテキストおよびYディクショナリコンテキストをロードされ、単一のバイナリミックスマスクがMMレジスタ416において設定される。Xメモリコンテキストおよび Yメモリコンテキスト、さらにはミックスマスクは、必要であれば実行中にリロードされてよいことに留意されたい。このリロードは、各々が専用のエンコーディングを伴うさらに小さなセクションにおける、元のコードセグメントのさらなる粒度をもたらし得る。たとえば、スマートフォンのような一部の複雑な組込みシステムは、メインアプリケーションから複数の独立した子アプリケーションを呼び出すことがあり、これらの子アプリケーションはコード空間を共有せず、自己完結型である。各々のそのようなアプリケーションは、X/YディクショナリおよびMMからなる固有の専用のエンコーディングを有してよく、そのエンコーディングは子プロセスの始動においてロードされる。

【００２３】

そのような状況の良い例は、その実行の期間においてシステムリソースの大半を占め、かなりの長さの時間、電子メール処理アプリケーションのみに属するコードを実行する、電子メール処理アプリケーションをスマートフォンのオペレーティングシステム(OS)が起動することであろう。電子メールアプリケーションのための専用のエンコーディングが、起動時にロードされ、電子メールアプリケーションが動作を終了した時だけ、異なるOS専用エンコーディングによって置き換えられる。さらに、あり得る状況は、単一のディクショナリがOSおよび電子メール処理アプリケーションのために使用されるが、インデックスX(408)および/またはインデックスY(410)のレジスタの内容を調整してX/Yディクショナリ

の適切なセクションへとオフセットすることを介して、異なるセクションが利用されるというものである。インデックスX(408)および/またはインデックスY(410)のレジスタの内容のロードは、たとえば、OS割り込みによるシステムレベル要求を介して、または、アプリケーションコードとともにエンコーディングされる特別な命令を介して達成されてよく、特別な命令は、インデックスX(408)および/またはインデックスY(410)のレジスタの内容のリロードに対する要求として、プロセッサによって理解される。この場合、復号テーブル220の数は、L2において同時に存在する可能性のある異なるエンコーディングを伴うプログラムセグメントの数と等しい。上記のスマートフォンの例を使用すると、OSが、OS固有のエンコーディングを常に保持する変換メモリ(TM)230を有する固有のデコーダ228に専用である場合、システムは、利用されるシステムアプリケーションに対してカスタマイズされた、異なるエンコーディングを保持する別のアプリケーション固有のTM 230を組み込み得る。どの変換メモリ230が適切かを判定することは、コードページごとにTLBエントリーを介して行われ得る。対照的に、1つのTM 230しかシステムにおいて設計されない場合、次の手順が、設置されたTM 230の容量に応じて適切であり得る。OS TM 230の容量を超える複数の独立のエンコーディングが依然として望まれる場合、上の例のように、OSが電子メール処理アプリケーションを起動すると、新たなX/Yデ
ィクショナリがX/Yメモリ(412、414)に配置される。MMレジスタ416が更新されてよく、L2/TLBフラッシュ要求が発せられる。L2/TLBフラッシュ要求は、L2/TLBの中のOSに対応する圧縮コードを無効にする。説明の残りの部分では、解凍器システム400は、任意のアプリケーションコードを含むシステム全体に対して使用されるべき単一のTM 230および単一のエンコーディングを有する、デコーダ228を使用して説明される。

【００２４】

次に、実行パイプライン234は、L1命令キャッシュ232から命令をフェッチすることを開始する。最初に、L1命令キャッシュへの各アクセスが、L2キャッシュ224へのアクセスを引き起こすミス指示422を生成する。最初に、L2キャッシュ224へのアクセスも、メインメモリ222へのアクセスを引き起こすミスを生成し、メインメモリ222は、L2キャッシュ224にロードされマルチプレクサ405を通じてデコーダ228に転送される圧縮命令に応答する。図3の解凍動作316に関して説明されたように、デコーダ228は、L1命令キャッシュ232への記憶および実行パイプライン234における実行のために、XYインデックス圧縮命令を非圧縮フォーマットへと解凍する。短い動作期間の後、L1命令キャッシュ232およびL2キャッシュ224は、安定状態に達するであろう。

【００２５】

プロセッサから見ると、実行パイプライン234は、L1命令キャッシュ232の中で探索されるべき命令の、フェッチアドレスおよび制御信号421によってフェッチ動作を試みる。最初に、L1命令キャッシュ232は、命令が存在しないと判定し、ミス指示422をL2キャッシュ224に発する。たとえば、L2キャッシュフェッチ動作は、XYエントリー404のためのものであり、XYエントリー404は、XYエントリー404がマルチプレクサ405を通じてデコーダ228に渡されるようにする、L2キャッシュ224におけるヒットである。XYエントリー404は、インデックスXレジスタ408において受け取られるXインデックス値0x54と、インデックスYレジスタ410において受け取られるYインデックス値0x734とに分割される。アドレス0x54においてXメモリ412からフェッチされたXパターン306が、結合器418に与えられる。アドレス0x734においてXメモリ414からフェッチされたYパターン308も、結合器418に与えられる。MMレジスタ416に記憶される単一のミックスマスク(MM)304 [0xFF80FF80]がさらに、結合器418に与えられる。結合器418は、MM 304に従って、Xパターン306からの適切なビットをYパターン308からの適切なビットと組み合わせて、キャッシュライン420に記憶され実行パイプライン234に渡される元の命令302を生成する。

【００２６】

そのようなシステムの別の追加の特徴は、暗示された暗号化フォーマットで記憶されるプログラムコンテンツである。特定の暗号化タイプのデータスクランブリングが命令ストリーム上で実行されない場合でも、プログラムコードは、アプリケーション固有の圧縮形

式で、スタティック型の記憶デバイス214、メインメモリ222、およびL2キャッシュ224に記憶される。プログラムコードのエンコーディングされた状態の一部は、TM 230の中のプロセッサコア226の内部に常駐し、これは最終製品において容易にアクセス可能ではないので、スタティック型の記憶装置214および上層のメモリ階層222および224の内容は、元のプログラムを復元するのに十分ではなく、分析および複製を難しくする。

【００２７】

プロセッサコア226は、複数のアドレス空間を処理する。実行パイプライン234は、メモリ階層全体で使用される物理アドレス空間とは異なる、仮想アドレス空間において動作している。あるアドレス空間から別のアドレス空間への変換というジョブは一般に、変換索引バッファ(TLB)によって実行され、物理アドレス空間が同じサイズの命令を含む場合、複雑さが低い。したがって、本発明によれば、実行コードセグメントは、固定のバイト境界調整された空間を合計で占める、XインデックスとYインデックスの不変サイズのペアとして表される。以下でより詳しく説明されるように、この手法は、物理アドレス空間ページ化プロセスに基づく、圧縮命令と非圧縮命令の混合を可能にし、コードページは、TLBの単一エントリーによって処理される、コードのアトミックユニットを表す。

【００２８】

図5Aは、XインデックスおよびYインデックスのXメモリおよびYメモリへの配置における、パッドビットの利用を示す。本発明による、第2のミックスマスク504に基づいてバイト重複パッドビットを伴う第2のXパターンおよび第2のYパターンへと命令B 502を分割する、別の命令Bの分割プロセスの例示的な要素500。第2のコードセグメントを圧縮するために、第2のコードセグメントは、命令のグループへと分割され、各グループは、Bxパターンと固有のByパターンのセットとに寄与する。Bxパターン506は、第2のコードセグメントからの命令のグループに共通のビットパターンを表す。Byパターン508は、命令の同じグループにおける、固有のビットパターンのセットの1つを表す。コードセグメントのBxパターンは、Xメモリからなる Xディクショナリに記憶され、コードセグメントのByパターンは、Yメモリからなる Yディクショナリに記憶される。Xインデックスは、Xメモリの中の位置のアドレスであり、Yインデックスは、Yメモリの中の位置のアドレスである。圧縮プロセスにおいて、たとえば、命令502 [0xBAFF0FEB]は、ミックスマスク504 [0xFFC0FF80]と組み合わされて、Bxパターン506 [0xBAC3(E,F)]およびByパターン508 [0xFF5(8,9,A,B,C,D,E,F)]を生成する。解凍プロセスにおいて、たとえば、ミックスマスク504 [0xFFC0FF80]は、Xメモリからフェッチされた Bxパターン506 [0xBAC3(E,F)]およびYメモリからフェッチされたByパターン508 [0xFF5(8,9,A,B,C,D,E,F)]と組み合わされて、元の命令502[0xBAFF0FEB]を生成する。

【００２９】

図5Bは、図3のAx、Ayパターン、図5AのBx、Byパターン、ならびに、Xメモリ522およびYメモリ528の中の、CxおよびCyによってここでは表されるさらに別の命令Cの、例示的な記憶構成520を示す。本発明によれば、3つの命令A、BおよびCは、3つの異なるマスクAmm≠Bmm≠Cmmを有する。たとえば、Xメモリ522およびYメモリ528は、バイトアドレス可能であり、64ビット長のアクセスラインを利用する。Axパターン306 [0x9F00(8,9,A,B)]は、Ax 523 [1001 1111 0000 0000 10]としてバイナリでエンコーディングされ、Byパターン506 [0xBAC3(E,F)]は、Bx 524 [1011 1010 1100 0011 111]としてバイナリでエンコーディングされ、例示的なCx 525は、Cx 525 [1110 0000 0000 0000 1]としてバイナリでエンコーディングされる。例示的な記憶構成520では、ミックスマスクAmm 304 [0xFF80FF80]、Bmm 504 [0xFFC0FF80]、および例示的なCmm [FF80FF00]は等しくない。したがって、各ミックスマスクと関連付けられるXパターンは、異なる数のビットであり、同様に各ミックスマスクと関連付けられるYパターンは、異なる数のビットである。

【００３０】

Xパターン、Yパターン、またはこれらの両方のインデックス圧縮は、重複パッドビット310、312、510、および512のような重複パッドビットと、バイトアドレス可能位置とを利用する。第1のXパターンは、第1のXパターンの終了ビットを第2のXパターンの開始ビット

と比較することによって、第2のXパターンと重複し得る。この比較において、重複したパッドビットを有するバイトは、図5Bに示されるように、さらなる範囲の選択肢をもたらす。たとえば、Axパターン523は、第1のバイト0x9Fと、第2のバイト0x00と、重複パッドビット310が原因でセット{8,9,A,B}のうちの1つの値を有する第1のニブルおよび0x00以上0xFF以下の任意の数であり得る第2のニブルからなる第3のバイトとを含む、3バイトからなる。Bxパターン524は、第1のAxパターン523と重複し、Bxパターン524の第1のニブルにおいて、セット{8,9,A,B}のうちの1つである0xBという値を有する。Bxパターン524の第2のニブルは、0x00から0xFFまでの数である、0XAという値を有する。Cxパターン525の開始ビットは、Bxパターン524の終了ビットと重複する。Yパターンに対しては、Yメモリ528において重複は示されていない。ビット534および535のような、複数のパターンの間のビットは、圧縮または解凍のプロセスにおいて使用されないので、任意のバイナリ値をとってよい。コードの変更または追加によって、0x58という第1のバイトを有する新たなDyパターンが生成されると、新たなDyパターンは、図5Bに示されるように、Byパターン532と重複し得る。また、0x5という第1のバイト、およびセット{8,9,A,B,C,D,E,F}のうちの1つとして第2のバイトを有する他のパターンは、Byパターン532の終了ビットと重複し得る。

【００３１】

図5Cは、本発明による、圧縮形式でメインメモリ542に記憶されるプログラムと、非圧縮形式でレベル1命令キャッシュ(L1命令キャッシュ)548に記憶されるように複数のミックスマスクおよびインデックス圧縮を使用してデコーダ546において解凍されるレベル2キャッシュ544とを有する、解凍器システム540を示す。L2キャッシュ544は、0x102という9ビットのXインデックス値と、0xA9という8ビットのYインデックス値と、0x2という2ビットのMインデックス値とを有するXYMエントリー554のような、Xインデックス、Yインデックス、およびミックスマスクインデックスを、アドレス可能な位置に記憶するXYMインデックスメモリ552を含む。マルチプレクサ555は、L2キャッシュヒット556があるとXYMエントリーを選択し、または、メインメモリ542からXYM値557を選択するために使用される。解凍動作は、インデックスXレジスタ558、インデックスYレジスタ560、MMインデックスレジスタ561、Xメモリ562、Yメモリ564、複数のMMメモリ566、ダブルラインXレジスタ568、ダブルラインYレジスタ570、および結合器572を有する、デコーダ546において遂行される。L1命令キャッシュ548は、キャッシュライン580のような、複数のキャッシュラインを含む。初期動作の期間の後、L2キャッシュ544およびメインメモリ542は圧縮コードをロードされ、L1命令キャッシュ548は非圧縮コードをロードされる。

【００３２】

実行パイプライン550が、L1命令キャッシュ548からの命令のフェッチ581を試み、命令が存在しないと判定すると、ミス指示582が発せられ、フェッチはL2キャッシュ544にリダイレクトされる。たとえば、このフェッチは、XYMエントリー554のためのものであり、XYMエントリー554は、XYMエントリー554がマルチプレクサ555を通じてデコーダ546に渡されるようにする、L2キャッシュ544におけるヒットである。XYMエントリー554は、インデックスXレジスタ558において受け取られるXインデックス値0x102と、インデックスYレジスタ560において受け取られるYインデックス値0xA9と、インデックスMMレジスタ561において受け取られるMインデックス値0x2とに分割される。Xパターン506 [0xBAC3E]を含むラインが、アドレス0x0x100においてXメモリ562からフェッチされ、ダブルラインXレジスタ568にロードされる。Yパターン508 [0xFF58]を含むラインが、アドレス0xA8においてYメモリ564からフェッチされ、ダブルラインYレジスタ570にロードされる。ミックスマスク504 [0xFFC0FF80]は、アドレス0x2において複数のMMメモリ566からフェッチされ、MMレジスタ571にロードされる。Xパターン506 [0xBAC3E]は、0x102というXインデックス値に基づいてダブルラインXレジスタ568から選択され、結合器572に与えられる。Yパターン508 [0xFF58]は、0xA9というYインデックス値に基づいてダブルラインYレジスタ570から選択され、結合器572に与えられる。MMレジスタ571に記憶される単一のミックスマスク(MM)504 [0xFFC0FF80]がさらに、結合器572に与えられる。結合器572は、MM 504に従って、Xパターン506からの適切なビットをYパターン508からの適切なビットと組み合わせて、キャッ

シュライン580に記憶され実行パイプライン550に渡される元の命令502を生成する。ダブルラインXレジスタおよびダブルラインYレジスタにおいてXYMエントリー554とともに記憶される他の圧縮命令は、特に同じミックスマスク504を使用する場合、命令シーケンスに応じて、XYMエントリー554の解凍と並列に、または順番に、解凍され得る。解凍可変長命令のフォーマット長は、圧縮命令と関連付けられるミックスマスクによって決定される。

【００３３】

図5Bおよび図5Cに示される、XメモリおよびYメモリ内の個々のパターンの配置の特徴は、その配置が制約されず、最も左側のバイトの境界調整によって支配されるということである。パッドビットの利用率が高い、許容可能な、最適な、または最適に近い配置は、バイト境界調整の断片化を最小化または除去できる可能性がある。左側が合わせられる境界調整の要件のような、シンボル配置に対する任意の制約が、断片化を発生させる可能性がある。たとえば、ランダムなサイズのオブジェクトのバイト境界調整された記憶領域において、最悪の状況は、記憶されたシンボル当たり7ビットが無駄になり、特定のシンボル配置が保証するもの以外の有用な目的を果たさないことである。一般にインデックス圧縮の助けを借りて、特にパッドビットの使用によって、これらの7ビットは有用なデータの記憶のために使用され得る。無駄なビット利用に対する、記憶領域の節減の上限は、7ビットに記憶領域中のシンボル数を乗じたものである。一般に、インデックス圧縮による全体の節減はより大きく、当然、コンテンツの情報エントロピーにしか制約されない。同じ物理ビットを占め、シンボルの長さを暗黙的に決定する適切なMMの存在下でのみ正しく抽出される、異なるサイズのシンボルも存在し得る。単一のMMが使用される場合、X/Yディクショナリのシンボルエントリーは同じサイズを有する。複数のミックスマスクが利用される場合、シンボルサイズには、MMの変動と同程度の変動があり得る。たとえば、可変長命令フォーマットは、32ビットフォーマットの命令と64ビットフォーマットの命令とを含み得る。それでも、シンボルの最適な配置というタスクはNP完全問題であり、適度な線形時間で実際に完全に解かれると考えることはできない。この制約を克服するために、図12～図14に関して以下でさらに詳しく説明されるように、線形時間において最適に近い配置を生成するためにヒューリスティックアルゴリズムが使用される。

【００３４】

圧縮命令および非圧縮命令は、動作中のモード切り替えの必要なく、同じ実行可能なコードセグメントにおいて混合され得る。ページ化された仮想メモリシステムでは、メモリのページは、圧縮命令と非圧縮命令のいずれかを格納し得る。図6は、本発明による、圧縮ページおよび非圧縮ページを有する、ページ化された仮想的なキャッシュ編成600を示す。仮想アドレスは一般に、2つの部分にエンコーディングされる。アドレスビットの上側のフィールドは通常、4Kバイトのページのような、選択されたページサイズに基づいてエンコーディングされる、仮想ページ番号を表す。アドレスビットの下側のフィールドは、アドレス指定されたページ内のアドレスを特定する、ページオフセットである。仮想アドレスから物理アドレスへの変換において、仮想ページ番号は物理ページ番号に変換される。ページオフセットは、仮想アドレスと物理アドレスの両方に対して同じであり、変換されない。

【００３５】

仮想アドレスから物理アドレスへの変換システムは、変換プロセスの性能を向上させるために、1つまたは複数の変換索引バッファ(TLB)、たとえば命令およびデータTLBを含み得る。命令TLB(ITLB)は、エントリー検証およびアクセス許可のような記憶されたページの属性とともに、最近の仮想アドレスから物理アドレスへの変換を記憶する、小さなキャッシュである。ITLBは一般に、ランダムアクセスメモリ(RAM)回路と結合されるコンテンツアドレス可能メモリ(CAM)回路を含み、比較的小さく、たとえば32個または64個のエントリーを有する。各ITLBエントリーは、RAM回路中の変換された物理ページ番号と関連付けられる、最近使用された仮想ページ番号を有する、CAM回路中のタグを含む。たとえば、ページ化された仮想的なキャッシュ編成600は、ITLB 602と、圧縮ページ608と混合された非圧縮ページ606を有する物理メモリ604とを使用する。ITLB 602の各エントリーは、仮

想アドレスタグ610、有効(V)フラグ、読取り(R)フラグ、書込み(W)フラグのようなエントリーフラグ612、物理ページ(P-page)アドレス614、および圧縮(C)フィールド616を有する。Cフィールド616は、たとえば、すべての圧縮ページのために単一のミックスマスクを有するシステムに対して、圧縮されているまたは圧縮されていないものとしてページを識別するのに適切な、単一のビットであり得る。あるいは、Cフィールド616は2ビット以上のビットであってよく、2ビットのフィールドの場合、「00」は圧縮されていないことを示し、「01」は第1のミックスマスクによって圧縮されていることを示し、「10」は第2のミックスマスクによって圧縮されていることを示し、「11」は第3のミックスマスクによって圧縮されていることを示し得る。命令またはコードのブロックを圧縮するか圧縮しないかの判断は、コンパイルおよびコード圧縮の時に統計的に行われ、種々の要因に依存し得る。たとえば、システムの実際の実装形態が、復号のレイテンシに何らかの影響を受けやすい場合、アプリケーションの性能に重要な部分は元の非圧縮形式に保たれてよいが、あまり頻繁に実行されないコードは圧縮され得る。頻繁に実行されるコードの部分か頻繁に実行されないコードの部分かの判定は、コード
の制御構造に基づいて、任意選択のプロファイル対象フィードバック情報208またはコンパイル時間のヒューリスティックのいずれかによって、コンパイラおよびリンカ206によって行われる。しかしながら、L2キャッシュ224とL1キャッシュ232の間へのデコーダ228の配置によって、システムの性能に重要な経路からデコーダが実質的に除去される。デコーダのそのような配置による追加の利益には、実行パイプライン234への変更を必要とせず、L2容量の向上による電力節減の可能性があることによって、メインメモリへのアクセスが最小限になることがある。これらの圧縮の判断は、変換ツール216の、コンパイラおよびリンカ206との密接な相互動作に基づき、プロファイルフィードバック情報208を利用することができ、プロファイルフィードバック情報208は、たとえば、レガシーコードを圧縮されるべきではないものとして識別し、新たな機能コードを圧縮されるべきものとして識別し得る。

【００３６】

図7は、図6のページ化された仮想的なキャッシュ編成600と同様のシステムにおいて、物理メモリページごとに圧縮コードと非圧縮コードを混合する、システム700を示す。圧縮ページ701と非圧縮ページ702の合成が、4バイト(4B)のエントリーを伴う4キロバイト(4KB)のページを有する、例示的なメインメモリ物理アドレス空間に対して示されている。圧縮命令の各ページおよび非圧縮命令の各ページは、同じ固定の数のバイトを有する、同じ容量である。しかしながら、たとえば、非圧縮命令フォーマットで使用されるビット数の半分のビット数である圧縮命令フォーマットを使用すると、圧縮ページ701に記憶される命令の数は、非圧縮ページ702に記憶される命令の数の2倍である。たとえば、圧縮ページ701は、16ビットまたは2バイトのX/Yインデックスのペアとして、圧縮命令d、e、f、gなどを記憶する。したがって、4Kページである圧縮ページ701は2048個の命令を格納し、非圧縮ページ702は1024個の命令しか格納しない。ページ境界は、ページが圧縮命令を保持するか非圧縮命令を保持するかによって影響を受けないので、ページの物理的な位置に対するアドレス変換は変化しないが、圧縮ページはより個別の命令を保持する。

【００３７】

図8Aは、本発明による、圧縮命令ページを解凍し、非圧縮命令ページにアクセスするための、ページ化された解凍器システム800を示す。ページ化された解凍器システム800は、プロセッサパイプライン802、ITLB 804、物理アドレスバッファ806、L1命令キャッシュ808、L2キャッシュ回路810、および圧縮命令デコーダ回路812からなる。変換プロセスは、仮想アドレス803から選択された仮想ページ番号805をCAM回路に与えることによって開始し、CAM回路は、与えられた仮想ページ番号全般と、CAMタグ819中のエントリータグとともに記憶された、記憶されている最近使用された仮想ページ番号のすべてとの同時比較を行う。一致がある場合、CAM回路は、RAM回路中の対応するエントリー820にアクセスし、対応するエントリー820は、物理アドレスバッファ806に記憶される変換された物理ページアドレス815として出力される。変換された物理アドレス809は、仮想アドレス803からの

ページオフセット817と連結された、変換された物理ページアドレス815を含む。

【００３８】

たとえば、4ギガバイト(4GB)および4Kバイトのページの仮想アドレス空間を有する組込みシステムでは、仮想アドレス803は、ビット[31:12]を有する仮想ページ番号805およびビット[11:0]を有するページオフセット807からなる。同じ組込みシステムにおいて、キャッシュおよびメインメモリのメモリ階層は、512Kバイトの物理メモリ空間および4Kバイトのページを包含し得る。ITLB 804においてヒットがあると、仮想アドレス803は物理アドレス809に変換される。物理アドレス809は、ビット[28:12]を有する物理ページ番号815およびビット[11:0]を有するページオフセット817からなる。そのようなシステムでは、仮想から物理への変換システムは、ビット[31:12]においてエンコーディングされた仮想ページ番号805を、ビット[28:12]においてエンコーディングされた物理ページ番号815に変換する。また、ヒットがあると、圧縮ビットフィールド821も、Cビットフィールド822として物理アドレスバッファ806に記憶されるために出力される。圧縮ビットフィールド821とCビットフィールド822の配置は例示的である。

【００３９】

物理アドレス809は、一致するエントリーをL1命令キャッシュ808において探索するために使用される。一致するエントリーが発見される場合、それは、一致するエントリーと関連付けられ、L1/L2マルチプレクサ824を通ってプロセッサパイプライン802に達するように選択される、非圧縮命令である。L1命令キャッシュ808でミスがあると、物理アドレスは、一致するエントリーを探索するためにL2キャッシュ810に向けられる。L2キャッシュ810においてヒットがあると、圧縮ページからの圧縮命令を示すCビットフィールド822によって、関連付けられる一致するエントリーを有する圧縮命令のラインがフェッチされ、L2読取りバッファ826に記憶される。L2キャッシュ810においてヒットがあると、非圧縮ページからの非圧縮命令を示すCビットフィールド822によって、関連付けられる一致するエントリーを有する非圧縮命令のラインがフェッチされ、L2読取りバッファ826に記憶される。非圧縮命令は、圧縮命令デコーダ812を迂回し、L1命令キャッシュ808への記憶のために、L2読取りマルチプレクサ828において利用可能にされ、L1/L2マルチプレクサ824を通ってプロセッサパイプライン802達するように選択される。

【００４０】

L2キャッシュ810においてヒットがあると、L2読取りバッファ826中のフェッチされた圧縮命令は、圧縮命令を制御回路830に示すCビットフィールド822に基づいて、圧縮命令デコーダ812で解凍される。解凍プロセスは、図8Bに関して以下でより詳しく説明される。解凍命令は、L1命令キャッシュ808への記憶のために、L2読取りマルチプレクサ828によって選択され得る解凍バッファ832に記憶され、L1/L2マルチプレクサ824を通ってプロセッサパイプライン802に達するように選択され得る。

【００４１】

図5Cで前に示されたように、ネイティブアプリケーションの圧縮は、複数のミックスマスクの使用を規定し得る。所望の粒度が単一の命令にまで至る場合、図5Cで示唆されるように、各々のX/Yインデックスペアは、適切なマスクを選択するための方法を有さなければならない。ミックスマスクが命令ごとに選択可能である場合、ミックスマスクの識別には、たとえば、XYNエントリー554で示される2ビットのミックスマスクインデックスのような、追加の記憶ビットを必要とする。別の手法は、ミックスマスクがコードページごとに選択されることを許容し、これによって、X/Yインデックスペア554からミックスマスクマーカーの必要性がなくなり、たとえばCフィールドの一部として、ミックスマスクマーカーがTLBページ記述子に配置される。

【００４２】

図8Bは、プロセッサパイプライン上で実行される、L2キャッシュの圧縮ページ命令およびL1命令キャッシュの解凍命令の状態を示す、例示的な解凍状態図850を示す。図8Bは、L2キャッシュ852、L2読取りバッファ854、圧縮命令デコーダ(CID)856、解凍バッファ858、L1命令キャッシュ860、物理アドレスバッファ862、およびプロセッサパイプライン864を

示す。L2キャッシュ852は、非圧縮ライン866および圧縮ライン868を、初期状態として有する。非圧縮命令は32ビットのネイティブな命令であり、圧縮命令の各々は、たとえば図4に関連して上で説明されたような、X-インデックスとY-インデックスのペアからなる16ビットである。圧縮ライン868は、非圧縮ライン866の2倍の命令を保持する。命令Mのフェッチは、プロセッサパイプライン864によって行われ、プロセッサパイプライン864は、図8AのITLB 804のようなITLBによって仮想アドレスから物理アドレスに変換され、物理アドレスバッファ862に記憶される。最初は、L1命令キャッシュ860は命令Mを格納せず、メモリ階層のより上位のレベルへのフェッチを引き起こす、ミスが生成される。この例示的な状況では、解凍バッファ858が最初に確認され、命令Mは発見されない。L2キャッシュ852が次に確認され、圧縮ライン868が、圧縮形式「m」870で命令Mを格納していることが発見される。圧縮ライン868がフェッチされ、L2読取りバッファ854に記憶される。CID 856はCビットフィールド872およびフェッチアドレスを受け取り、L2読取りバッファ854からの圧縮命令が、圧縮命令「m」870から始まって順番にフェッチされることが可能になる。フェッチシーケンスの中の最初の命令が最初にアクセスされ解凍されて、タイミング要件を最小にするためにプロセッサ実行パイプラインへと迅速に転送され得ることに留意されたい。本明細書で説明されるような圧縮命令は、固定長と、解凍の前に知られている記憶アドレス位置とを有するので、フェッチされたシーケンスにおける最初の圧縮命令が、圧縮命令のラインにおいて識別され得る。L1命令キャッシュは、8個の非圧縮命令というライン長を有するので、圧縮命令は、「m、n、o、p、次いでi、j、k、およびl」から開始するL1命令キャッシュライン長で、L2読取りバッファ854から入手される。次に、「q、r、s、t、u、v、w、およびx」から開始するL2読取りバッファ854の後半が入手される。したがって、圧縮命令は、L2読取りバッファ854中の任意の開始点から入手される。

【００４３】

命令Mが最初にフェッチされるので、命令MがCID 856において最初に解凍され、次いでプロセッサパイプライン864に転送され得る。他の命令「n、o、p、次いでi、j、k、およびl」が解凍されるにつれて、それらは、解凍命令Mと組み合わされて、解凍バッファ858にロードされる。命令のすべてのラインが解凍されると、それらはL1命令キャッシュ860にロードされる。あるいは、個々の解凍命令または解凍命令のペアが、たとえば、L1命令キャッシュ860において個々に更新されて、必要に応じてプロセッサパイプライン864に転送され得る。解凍バッファ858に記憶された解凍命令ラインがL1命令キャッシュ860において重複する場合、解凍命令ラインを保持する解凍バッファは、L1から除去された最後の命令ブロックを一時的に記憶するビクティムキャッシュとして使用され得ることに留意されたい。

【００４４】

X-インデックス、Y-インデックス、Xパターン、Yパターン、および1つまたは複数のミックスマスクの決定は、特定のコードセグメントにおける、ネイティブの命令アーキテクチャおよび命令の使用法に依存する。スマートフォンにおけるコードセグメントのようなコードセグメントを分析する時、単一のミックスマスクは、複数のミックスマスクの使用を通じて得られる圧縮と同等の圧縮を提供することができる。単一のミックスマスクの使用は、単一の命令グループを使用し、コードセグメントで使用されるネイティブな命令を2つの部分、すなわちXパターンおよびYパターンへと効果的に分離するものとして理解され得る。グループ中での、固定の部分(X)と可変の部分(Y)の元の区別は、あまり重要ではなくなり、互いに置き換え可能であると見なされ得る。また、単一のミックスマスクの使用は、図4の結合器418のような結合器回路の設計を単純にする。たとえば、16542個のネイティブな32ビット命令を有する実験的なコードセグメントでは、6464個の固有の命令しかなく、全体のエントロピーは0.76であった。単一のミックスマスクを使用して、これらの6464個の命令は、1351個のXパターンと1345個のYパターンに分割された。本発明の技法を使用すると、16542個のネイティブな32ビット命令の各々は、23ビットのX/Y-インデックスのペアによって置き換えられ、図2に示されるようなシステム中のL2キャッシュおよびメインメモリの記憶容量を、28.1%圧縮する。その結果、エンコーディングされたデー

タのエントロピーは0.76から0.89に上がる。しかしながら、理論的に可能な1.0の情報エントロピーという基準に対して0.89というエントロピーの測定結果は、異なるミックスマスクがかなりより良い結果をもたらし得ることを示している。複数のコードセグメントにわたる最良の可能な圧縮をもたらすための、最適なまたは最適に近い単一のミックスマスクの判定は、複雑な問題である。

【００４５】

図9A～図9Cは、本発明による、遺伝的アルゴリズムの変動に基づく、自動的なミックスマスク決定プロセス900を示す。図2Aの変換ツール216のような変換ツールは、プロセス900を実施する。ブロック902において、プロセス900が開始される。ブロック904において、圧縮されるべきコードセグメントPが取得される。たとえば、コードセグメントPは、ブートコード、オペレーティングシステムコード、および特定の製品において利用されるような複数の機能プログラムを含み得る。ブロック906において、コードセグメントP中の固有の命令が、Pハッシュテーブル(P_unique)において収集される。ブロック908において、P_unique命令が、命令フォーマットに基づいて、意味的な順序で、たとえばビットパターンの昇順で並べ替えられる。たとえば、Pハッシュテーブルにおける各命令は、32ビットを使用した整数として解釈され、リストにおいて昇順で並べ替えられる。これによって、隣り合う命令の間のハミング距離が最小になることが保証され、後のステップにおける命令のグルーピングが可能になる。ブロック910において、固有の命令の並べ替えられた順序において、ビット切り替え頻度が収集される。たとえば、位置pのビットが「1」から「0」に、または「0」から「1」に変わると、bit_toggle[p]の値は1だけインクリメントされる。bit_toggleの列サイズは、32ビット命令に対しては32のように、ビット単位で表される元のISAにおける最大の命令の幅に等しい。ブロック912において、種子ミックスマスクと種子MM集団のセットが、いくつかの既知の良好なMM、いくつかのランダムに生成されたMM、および収集されたビットトグルのカウントから生成されたいくつかのMMを使用して、選択される。既知の良好なマスクは、同様のアプリケーションから以前に取得されたMMから事前に用意され、任意の所与のアーキテクチャに特有である。収集されたビットトグルのカウントは、実際のコード中の所与の位置においてビットが値を変化させる確率を推定する際の、良好なヒューリスティックとして機能する。経験的に、所与の位置の閾値に応じてミックスマスクを選択することで、良好な種子マスクが生成される。ブロック912において、種子MM集団は、10個から30個の間のサイズの
個別のMMの列であり、偶数個の個別のMMとして選択され得る。より大きな集団がより良好な結果を生成し得るが、選ばれるMMの数の線形増加は、対応する計算時間の増加を必要とする。特定の種子MM集団は、特定の計算プラットフォームに適合するように選択され得る。実験的な構成では、20個のMMの種子集団の値が良好なトレードオフであるものとして示されている。ブロック914において、種子MM集団における各々の選択されたMMに対する適応度の値が、たとえば、小さいが0ではない値に設定される。適応度の値は、たとえば、MMの提供する利益のレベルを表す、倍精度の浮動小数点数である。プロセス900は、図9Bの接続部A 916へのリンクである、図9Aの接続部A 916へと進む。

【００４６】

図9Bには、プロセス900の繰返し部分が含まれる。20個のMMの種子集団が、プロセス900への入力として与えられる。判定ブロック920で、別の繰返しが行われるかどうかの判定が行われる。閾値は、数値的な終了条件と比較され、閾値が終了条件以下である場合のように終了条件に達しない場合、別の繰返しが開始される。

【００４７】

ブロック918において、最良の適応度の値を有する個別のMMが、最良の個別のMMであると決定され、その適応度の値(コスト関数)が記録される。また、以下でさらに詳しく説明されるように、集団中の個別のMMのコスト関数の合計として求められる全体の適応度の値が記録され、図10のプロセス1000から取得される。より高い変化率が望ましいという仮定に基づいて、全体の圧縮の効果が低い場合、より高い変異率をもたらす変異アルゴリズムに対するパラメータとして、全体の適応度の値が後で使用される。判定ブロック924にお

いて、プロセス900の完了の進行状況について判定が行われる。最後の繰返しでの最良の適応度の値が、前の繰返しから変化していなかった場合、プロセス900はブロック926に進む。ブロック926において、閾値が上げられる。最後の繰返しでの最良の適応度の値が、前の繰返しの最良の適応度の値と異なる場合、プロセス900はブロック928に進む。ブロック928において、集団は適応度の値によって並べ替えられる。判定ブロック930において、判定ブロック920で開始するプロセス900が最初の繰返しの途中であるかどうかの判定が行われる。最初の繰返しであると判定されると、プロセス900はブロック932に進む。そうでなければ、MMの現在の集団を伴うプロセス900は、新たな集団を作成するためにブロック934～952に進む。

【００４８】

判定ブロック934で、エリート主義インジケータが設定されるかどうかの判定が行われる。エリート主義インジケータは、複数のクラス中最良のMMが保存されるべきかどうかを示す。エリート主義インジケータが設定される場合、プロセス900はブロック936に進む。ブロック936において、前の繰返しからの、2つの最良性能のMMが、新たな集団に直接コピーされる。ブロック938において、図9Cに関して以下でさらに詳しく説明されるように、2つの子MMが、2つの最良性能のMMから生成される。ブロック940において、図9Dに関して以下でさらに詳しく説明されるように、2つの子MMが、2つの変異した子MMを作成するように、ランダムに変異する。ブロック942において、現在の集団の最初の4個のメンバーが、2つの最良性能のMMおよび2つの変異した子MMによって置き換えられる。次いで、プロセス900はブロック946に進む。

【００４９】

判定ブロック934に戻ると、エリート主義インジケータが設定されず、エリート主義が選択されないことが示された場合、プロセス900はブロック946に進む。ブロック946および948において、ブロック938において説明されたものと同様の交配手順およびブロック940において説明されたような変異が、前の繰返しの集団における残りの個別のMMの処理されていないペアに対して繰り返される。ブロック946において、パートナー選択アルゴリズムがランダムに個別のMMを選択するために使用され、選択の確率は、個別のMMの適応度に比例する。より性能が良い個別のMMは、保証はされないが、子を生成する可能性がより高い。ブロック948の終わりにおいて、新たな繰返しの集団が決定される。ブロック950において、新たな繰返しの集団において重複がないことを確実にするように、新たな繰返しの集団におけるあらゆる重複が変異する。ブロック952において、新たな繰返しの集団における個別のMMの適応度の値がリセットされる。この点において、新たな繰返しの集団は、コスト計算の準備ができている。計算時間を節減するために、以前に処理されたMM(個別のMM)の以前に計算されたコストのリストが保持され、すでに求められたコストはブロック932において単に事前に用意され、計算されない。次いで、プロセス900は、図9Cの接続部B 922に進む。

【００５０】

ブロック938における交配動作に戻り図9Bを参照すると、2つの最良性能のMMが、図9Dの親A 960および親B 962として選ばれ、ミックスマスク(MM)は32ビットの数として表され、このステップにおいてMMは遺伝子と表される。ブロック946において実施される図9Dの交配プロセス959では、2つの個別のMMは、入力されたMM集団から選択され、親A 960および親B 962とも表され得る。ブロック938および946において、2つの親は、ランダムな頻度で、遺伝子中のランダムな点で、遺伝子交換プロセス964を介して図9Dに示されるように交配され、2つの子A 966およびB 968を生成する。ランダムな選択は、擬似乱数生成器によって誘導され、半ランダムな選択と称され得る。一方の親である親A 960は、一方の子である子A 966の新たな遺伝子の開始部分970を提供し、他方の親である親B 962は、その子の遺伝子の末尾部分971を提供する。他方の子B 968は、上記の一方の子に継承されなかった遺伝子を含む。子A 966の末尾部分971を生成する親B 962は、子B 968の開始部分を生成する。子A 966の開始部分970を生成する親A 960は、子B 968の末尾部分を生成する。交差点972は、0～32の範囲でランダムに選択される。高速なアルゴリズムの収束の背後にある

理由は、親の一方から子の一方への、良好な適応形質の保存によるものであると理解することができ、このことは後で、適応率の計算によって確認される。図9Dに示される交配プロセス959は、それ自体が擬似ランダム事象であり、CROSSOVER_RATE閾値として識別される閾値の影響を受ける。たとえば、2つの親の遺伝子を交換しようとする100回の試みのうち、100にCROSSOVER_RATEを乗じた回数の交換の試みしか行われない。

【００５１】

図9Bの部分940および948に戻ると、新たに生成される子MM集団は、各々の子遺伝子において、半ランダムに個別のビットを反転させ、「0」から「1」に、または「1」から「0」に変化させることによって、変異する。この変異プロセス974は図9Eに示され、図9Eにおいて、個別のMM A 975が、ビット976、977、および978と関連付けられるビット反転によって変異する。変異した個別のMM A 979は、ビット反転の結果とともに示される。変異プロセス974において、擬似ランダム変異は、以前は利用できなかった形質をMM集団に導入するように動作する。たとえば、以前には試されなかった新たなMMが、望まれない変異が迅速に除去され得るように、制御可能な頻度で作成される。ブロック942において、MM集団の4個のメンバーは、ブロック936からの保存された2つの最良性能のMMと、ブロック940からの2つの新たに変異した子のMMとに設定される。ブロック946において、図9Cのプロセス955に関して説明されるように、残りのMM集団は、新たな子のペアを得るためにペアで交配される。ブロック948において、図9Dの変異プロセス974が続き、2つの新たな子を生成する。ブロック950において、あらゆる重複するMMが変異する。ブロック952において、各々の個別のMMの適応度の値がリセットされる。ブロック932において、繰り返すMMの以前に計算されたコストが再使用される。プロセス900は次いで、図9Cの接続部B 922へのリンクである、接続部B 922へと進む。

【００５２】

図9Cのブロック955において、現在の個別のMMのコスト関数が、選択されたMMと並べ替えられたP_uniqueのリストとの動的圧縮関数として計算される。コスト関数は、外部記憶装置のサイズと、変換ディクショナリのサイズという、2つの主要構成要素からなる。これらの組み合わされた2つの構成要素は、得られた圧縮の程度または達成されたエントロピーのレベルを反映する。外部記憶装置のサイズの、内部X/Yメモリ解凍テーブルサイズに対する正確な関係は、2つの構成要素の相対的な重要性を示すパラメータによって決定される。たとえば、コスト関数は、xインデックスサイズとyインデックスサイズの合計と、X/Yディクショナリのサイズとの、Kバイト単位の積に等しい。よって、コストは、X/Yディクショナリサイズ×(xインデックスサイズ+yインデックスサイズ)に等しい。この式は、外部記憶装置のサイズに対する、内部ディクショナリサイズの相対的な重要性を評価する際の、バランスのとれた手法を反映する。相対的な重要度の加重定数を取り入れることで、コスト関数に選好を取り入れることができる。たとえば、コストは、(加重定数×X/Yディクショナリサイズ)×(xインデックスサイズ+yインデックスサイズ)に等しい。現在の実装形態では、加重定数は1.00に設定されるが、0.01から1.00の間で変化してよい。

【００５３】

判定ブロック956において、プロセス900のランタイムが割り当てられた時間を超えたかどうかの判定が行われる。ランタイムが割り当てられた時間を超えている場合、プロセス900はブロック958に進み、停止する。ランタイムが割り当てられた時間を超えていない場合、プロセス900は判定ブロック957に進む。判定ブロック957において、現在のコストが最小であるMMが発見されたかどうかの判定が行われる。たとえば、閾値よりも低い現在のコストを生成する単一のMMを発見しただけでは、プロセス900を終了する理由としては不十分であり得る。一方、理論的に最小のエントロピー限界を表すものにすでに到達し、さらなる探索が何ら必要ではないことを検出する可能性がある。別の例では、ブロック958においてプロセス900を終了させるのに十分な理由となり得る、閾値未満のほぼ同一の現在のコストを各々が生成する、多数のMMが発見されることがあり、現在のコストが最小であるMMを選択するか、または、同一の許容可能な最小の現在のコストを各々が有する、複数のMMのうちの1つを選択する。現在のコストが最小であるMMが発見されていない場合、

処理900は、図9Bの接続部A 916に進む。

【００５４】

図10は、本発明による、例示的な動的な圧縮および現在のコストの計算980を、プロセス1000として示す。ブロック1002において、判定ブロック932において以前に評価されていないと判定された後で選択される、図9Bのブロック952から利用可能なMMのような、現在のコストの値が入力され、並べ替えられたP_unique命令が入力される。ブロック1004において、並べ替えられたP_unique命令は、コードセグメントのための単一のMMを有するか複数のMMを有するかに基づいて、選択されたMMの現在のグループ化のヒューリスティックに従って、グループ化される。たとえば、単一のMMのグループ化は、図5Aの例示的な要素500のようなX部分とY部分とに固有の命令を分離し、最初の圧縮されていないXおよびYテーブルを構築するプロセスである。一例では、全体のネイティブな命令プログラムは、単一のグループとして見なされ得る。一方、複数のMMのグループ化は、固有の命令のいくつかのグループへの分割を伴い、いくつかのグループは各々が固有のMMを有する。この場合、Xパターンは、グループの固定のパターンを表し、グループ全体が単一の固定のXパターンを共有し、グループごとの複数のYパターンは、グループの変異可能な部分を表す。いくつかのヒューリスティックがグループへの最適に近い分割のために可能であるが、1つの特定のヒューリスティックの選択は、別のヒューリスティックによって引き起こされてよく、これは、固有の命令の入力リストのサイズとその初期エントロピーとに基づき得る。ハミング、重み付けられたハミング、1の数、および重み付けられた1の数を含む、次のヒューリスティックが、実験を通じて実用的であることが判明している。

【００５５】

図11は、並べ替えられた固有の命令リストにおいて隣接するエントリーの間のハミング距離が規定された閾値を超える時に、新たなグループを作成するために使用される、ハミングヒューリスティック1100を示す。重み付けられたハミングヒューリスティック1100は、図11に示されるように、閾値と比較される変化スコアを計算する際に、あるビットフィールドの位置に対して、他の位置よりも重みを割り当てる。プロセス1100のブロック1102において、並べ替えられた入力の生の命令パターンの完全なリストが取得される(並べ替えられたP_unique)。次いで、2つの隣接するエントリーの間のハミング距離を分析することによって、2つの隣接するエントリーを一度に処理する。ブロック1104において、可変のハミング距離の値(hamm_dist)が、2つの隣接するエントリーのXORとして計算される。たとえば、生の命令のリストが、3個のエントリー0、1、および2しか含まない場合、hamm_distは、最初に第0のエントリーと第1のエントリーとのXORとして、次いで第1のエントリーと第2のエントリーとのXORとして、2回計算される。ブロック1106において、hamm_distは、一度に1ビット解析され、sum_weight値が各々の0ではないビットに対してインクリメントされる。ブロック1106は、各hamm_distにおいて0ではないビットをカウントするものと見なされ得る。インクリメントの量は、ビットの位置によって変わり得る。正確なインクリメントの量はweight_arrayから読み取られ、weight_arrayは図9Aのブロック910において計算されるBit_Toggle列から計算され、または統計的に事前に計算される。決定ブロック1108において、sum_weight値がweight_thresholdを超えているかどうかの判定が行われる。肯定的な結果は、新たな命令グループが形成されるべきであることを示す。ブロック1110において、新たなグループが形成され、プロセス1100は判定ブロック1112に進む。判定ブロック1108の否定的な結果によって、プロセス1100は、判定ブロック1112へと進む。判定ブロック1112において、並べ替えられたP_uniqueリストの最後に到達したかどうかの判定が行われる。リストの最後に到達していない場合、プロセス1100はブロック1114に進み、ループ変数「i」をインクリメントしてブロック1104に進む。リストの最後に到達している場合、プロセス1100はブロック1116に進み、停止する。

【００５６】

両方が命令中のビットフィールドの変異性の測定に基づく、1の数のヒューリスティックおよび重み付けられた1の数のヒューリスティックのような、他のヒューリスティックが使用され得る。たとえば、MMにおいて1の数がより少ないことは、グループごとのXパタ

ーンがより多くYパターンがより少ないことを示す。MMにおいて1の数がより多いことは、グループごとのXパターンがより少なくYパターンがより多いことを示す。最大の圧縮をもたらす1の数の選択において、単一の点に対する非線形な依存性が存在する。現在の手法では、ブロック1004において、単一のMM分割戦略が選択される。続いてブロック1006において、図4のXメモリ412に記憶されたXパターンおよびYメモリ414に記憶されたYパターンのような、XメモリおよびYメモリテーブルがそれぞれ、図4に関して上で説明されたように決定される。判定ブロック1008において、プロセス900において存在する可能性がある、適応していない個別のMMを表す低水準のMMを除去するためにフィルタリングヒューリスティックが使用される。このフィルタリングは、計算時間を節減するために行われ、プロセス1000の効果に影響を与えない。あるマスクから生成された圧縮されていないX/Yテーブルが、入力された生の命令リストのサイズをバイトサイズで上回る場合、そのマスクは明らかに適応していないと判定される。判定ブロック1008において、Xメモリサイズ+Yメモリサイズが、P_unique命令の数にネイティブな命令の幅を乗じたものよりも大きいかどうかの判定が行われ、ネイティブな命令の幅はこの例示的な場合では32ビットである。判定ブロック1008において、判定が肯定的であれば、プロセス1000はブロック1010に進み、ミックスマスクを拒絶する。ブロック1010において、1008において拒絶されたMMの適応度の値は0に設定され、プロセス1000は図9Cの接続部B 922に進む。判定が否定的であれば、プロセス1000はブロック1012に進む。

【００５７】

ブロック1012において、XメモリおよびYメモリテーブルが圧縮される。ブロック1012は、XおよびYテーブルのサイズに直接影響を与え、これは転じて、これらのテーブルをアドレス指定するために必要なインデックスのサイズを決定し、外部の圧縮コードセグメントのサイズを最終的に決定する。XおよびYテーブルは、図5Bおよび図12～図14に関して上で説明されたようなインデックス圧縮を使用して圧縮される。ブロック1014において、現在のコストは、定数C1と、XメモリサイズとYメモリサイズの合計とを乗じ、次いでこれにXインデックスサイズとYインデックスサイズの合計を乗じたものに等しいものとして生成される。このプロセス1000は、現在はX/YインデックスおよびX/Y圧縮テーブルの外部集合体へと分離されている圧縮コードの実際のエントロピーに近い、現在のコストによって表されるエントロピー値を計算するための、有利なヒューリスティックを提供する。任意選択でブロック1014において、理論的に最小のエントロピー測定値を求めることができ、さらなる探索が必要かどうかが判定され得る。ブロック1016において、生成された現在のコストが図9Cの判定ブロック956に出力される。

【００５８】

上で述べられたように、インデックス圧縮と組み合わされたX/Yテーブルのエントリーにおける最適に近い選択が、X/Yテーブルのサイズを大きく減らすことができる。2つの例示的な配置アルゴリズムが、圧縮の程度と実行時間とのトレードオフに基づいて使用され得る。第1のヒューリスティックは、本発明による、図12に提示されたパッド並べ替えを伴う、ほぼ網羅的な探索によるX/Yテーブル圧縮アルゴリズムである。処理は、圧縮すべきシンボルのリストを取得することによって、ブロック1202において開始する。これは、圧縮すべきX/Yテーブルシンボルの現在のセットのリストである。テーブルには、「table_size」シンボルがある。ブロック1204において、パッド重複の確率が最大になる順序で、このリストが並べ替えられる。また、ブロック1204において、繰返しインデックス「i」が0にリセットされる。

【００５９】

ブロック1204におけるパッド並べ替え動作は、実際の圧縮プロセスの開始の前にパッド重複の確率を上げるために、X/Yテーブル中のシンボルを並べ替えるための特別な方法である。パッド並べ替え動作は、高速並べ替えアルゴリズムによって実施され、比較機能が、あるシンボルの1つまたは複数の最も左側の有効バイトと、他のシンボルの1つまたは複数の最も右側の有効バイトとの間に一致がある場合、2つのシンボルの交換を要求する。ここで、シンボルの長さは全バイトにおいて測定され、左側に合わされる。最も右側の有

効バイトは、いくつかの有効ビットしか有さないことがあるが、並べ替えの目的では、これは重要ではない。

【００６０】

判定ブロック1206において、繰返しインデックス「i」が「table_size」より小さいかどうかの判定が行われる。繰返しインデックス「i」が「table_size」以上である場合、プロセス1200はブロック1232に進み、停止する。それ以外の場合、X/Yテーブルのi番目のシンボルがまだ処理されていないので、プロセス1200は判定ブロック1208に進む。判定ブロック1208において、パラメータseen[i]が「1」に等しく、i番目のシンボルがすでに処理されていることを示すかどうかの判定が行われる。seen[i]=1である場合、プロセス1200はブロック1210に進み、そこで繰返しインデックス「i」が1だけインクリメントされる。シンボルがまだ処理されていない、すなわちseen[i]≠1である場合、プロセス1200はブロック1212に進む。ブロック1212において、i番目のシンボルが圧縮されたテーブルに挿入される。挿入プロセスの詳細が、図13に関して以下で説明される。やはりブロック1212において、パラメータseen[i]を1に設定し、best_score変数を0に設定し、新たな繰返しインデックス「k」をi+1という値へと初期化することによって、挿入の事実が示される。

【００６１】

判定ブロック1214において、新たな繰返しインデックス「k」が圧縮されるべきシンボルの数より小さいかどうかの判定が行われる。新たな繰返しインデックス「k」が「table_size」より小さい場合、k番目のシンボルがまだ処理されていないので、プロセス1200はブロック判定ブロック1216に進む。判定ブロック1216において、パラメータseen[k]が「1」に等しく、k番目のシンボルがすでに処理されていることを示すかどうかの判定が行われる。seen[k]=1である場合、プロセス1200はブロック1218に進み、そこで新たな繰返しインデックス「k」が1だけインクリメントされる。シンボルがまだ処理されていない、すなわちseen[k]≠1である場合、プロセス1200はブロック1220に進む。ブロック1220において、一致スコアがk番目のシンボルに対して計算される。

【００６２】

ブロック1220において、score_matchの計算は、シンボル挿入プロセスを模倣するが、何らシンボルを挿入しない。むしろ、score_matchの計算は、現在のシンボルを、テーブルの現在の形式の圧縮テーブルへと配置することから予測される、節減の量に比例するスコアを返す。この計算プロセスは、最良の一致候補である可能性がある、圧縮されていないX/Yリスト中の残りの処理されていないシンボルを見つける。Score_machは、シンボルが存在するコンテンツ内で完全に一致する場合、一致表示すなわち、たとえば無限大のような非常に高いスコアを返し、または、シンボルが圧縮されたメモリの終わりにおいて一致する場合、いくつかの重複するバイトを与える。重複が発見されない場合、0のスコアが返される。

【００６３】

計算されたスコアは、k番目のシンボルを、現在の状態の圧縮テーブルへと挿入することから予測される、節減の程度を反映する。判定ブロック1224において、スコアが無限大に等しいかどうかの判定が行われる。上で述べられたように、無限大というスコアのような一致表示は、k番目のシンボルが圧縮されたテーブルの現在の状態に完全に一致し、k番目のシンボルの挿入が圧縮テーブルのサイズを増やさないということを意味し、プロセス1200はブロック1228の挿入ステップに直接進む。この場合、挿入のプロセスは、後に外部メモリにおいてシンボルを表すものになる、圧縮X/Yテーブルの開始点からの現在のシンボルのオフセットを記録することであることに留意されたい。一般に、複数のミックスマスクが使用される場合、共通のプレフィックスを共有する複数のシンボルは、オフセットは同じであるが長さが異なることがあることにも、留意されたい。スコアが無限大に等しくない場合、ブロック1226において、計算されたスコアとそれまでの最良のスコアとの比較が行われる。新たに計算されたスコアがそれまでの最良のスコアを超える場合、プロセス1200はブロック1230に進む。ブロック1230において、現在のbest_scoreは、新たに計算されたスコアと等しく設定され、best-indexは「k」に設定される。ブロック1218におい

て、新たな繰返しインデックス「k」が1だけインクリメントされ、プロセス1200はブロック1214に進む。ブロック1214において判定されるように、「k」がリスト(table_size)の終わりに達すると、プロセス1200はブロック1222に進む。ブロック1222において、best_scoreが決定されたかどうの判定が行われる。最良のスコアが決定されていない場合、繰返しインデックス「i」はブロック1210においてインクリメントされ、プロセス1200はブロック1206から継続する。ブロック1222において、0ではないbest_scoreは、最良の適応するシンボルが発見されたことを示し、プロセス1200はブロック1228に進む。ブロック1228において、最良の適応するシンボルが、圧縮テーブルに挿入され、挿入の事実が、パラメータseen[best_index]列を「1」に等しく設定することによって示される。繰返しインデックス「i」がブロック1206において判定されるようにtable_sizeに達すると、プロセスはブロック1232において終了する。この時点で、評価されたシンボルは、圧縮テーブルに挿入されている。

【００６４】

図13は、本発明による、シンボル挿入プロセス1300を示す。ブロック1302において、圧縮メモリの現在の状態と、挿入されるべき新たなシンボルが取得される。判定ブロック1304において、新たなシンボルが0であるかどうかの判定が行われる。新たなシンボルが0である場合、ブロック1320において、シンボルは圧縮メモリ中の特別な位置にマッピングされ、シンボルサイズに関係なく、1バイトだけが記憶される。判定ブロック1304において、新たなシンボルが0ではないと判定されると、プロセス1300はブロック1306に進む。ブロック1306において、現在のpad_maskが計算される。pad_maskは、シンボル中の最後の有効バイトの無効な部分をマスクするための使用される、バイナリ変数である。たとえば、挿入されているシンボルが15ビットのサイズである場合、最後のバイトの最後のビットは無効であり、配置のための比較において無視される必要がある。pad_maskは、この場合0xfeである(pad_maskは常に、最後の有効バイトのみに適用される)。判定ブロック1308において、繰返しインデックス「i」が、現在の圧縮テーブルのサイズから新たなシンボルサイズを引き1を足したものより小さいかどうかの、判定が行われる。判定が肯定的であれば、プロセス1300はブロック1310に進む。ブロック1310において、新たなシンボルの圧縮メモリへの配置が繰り返し試みられ、有効ビットのみを現在の内容に対してバイトごとに照合するために、各シンボル位置を検査する。そのような動作は、新たなシンボルの最初の有効バイトを、すでにメモリの中にある各バイトと繰り返し比較することによって達成され、最初の有効バイトに対して一致が発見されると、新たなシンボルの中の他の残りのビットが照合される。この照合は、上で説明されたようなpad_maskの適用を介して、シンボル中の有効ビットのみを考慮する。判定ブロック1312においてそのような位置が発見される場合、プロセス1300はブロック1318に進む。ブロック1318において、シンボルは決定された位置において配置され、繰返しインデックス「i」は、圧縮テーブルの開始点からのシンボルのオフセットになる。このインデックスは、外部メモリに最終的に記憶され、X/Y圧縮テーブルにおける元のX/Yパターン
を見つけるために利用される。ブロック1316に達すると、それは、現在のシンボルが、圧縮メモリの既存のコンテンツと完全には一致し得なかったということを意味し、現在のシンボルはリストの終わりに配置される必要がある。その場合、最大限の重複の検査が、シンボルの各バイトに対して実行される。重複が発見されない場合、新たなシンボルは単に、圧縮テーブルの終わりに配置される。

【００６５】

図14は、本発明による、パッド並べ替えを伴う三者間のX/Yテーブルの圧縮プロセス1400を示す。プロセス1400は、網羅的な探査によるX/Yテーブル圧縮プロセス1200を代替するものである。ヒューリスティックプロセス1400は、プロセス1200よりも0(n)速いが、それほど良好な圧縮をもたらさないことがある。しかしながら、プロセス1400の圧縮の結果は、多数の現実的な用途に対して許容できるものであり得る。その上、2つのプロセスの組み合わせが、実際の実装形態では可能であり得る。図14に示されるより高速なプロセスが最良のマスク候補を決定するために使用され、図12に示されるより低速なプロセスが最終

的なエンコーディングを生成するために使用される。プロセス1400は、図12のステップ1202、1204、1206、1208、および1212と同じである、ステップ1402、1404、1406、1408、および1412で開始する。ブロック1412において、i番目の要素の同様の挿入が、ブロック1212において行われるのと同様に実行されるが、ブロック1414以降では、図12のブロック1214および1222で使用されるようなbest_scoreは使用されない。最良スコアの評価を使用する代わりに、図14の残りのブロックでは、i+1とtable_sizeの間のシンボルが1回処理され、判定ブロック1424において判定されるような0ではないスコアとともに発見されるすべての要素が、ブロック1426において圧縮テーブルへと挿入される。

【００６６】

本明細書で開示される実施形態に関して説明される方法は、ハードウェアの組合せで、およびプロセッサによって実行される非一時的信号を記憶するソフトウェアモジュールで具現化され得る。ソフトウェアモジュールは、ランダムアクセスメモリ(RAM)、フラッシュメモリ、読取り専用メモリ(ROM)、電気的プログラマブル読取り専用メモリ(EPROM)、ハードディスク、リムーバブルディスク、テープ、コンパクトディスク読取り専用メモリ(CD-ROM)、または、当技術分野で既知の、もしくは今後勧められる任意の他の形態の記憶媒体中に常駐し得る。記憶媒体は、プロセッサがその記憶媒体から情報を読み取り、場合によってはその記憶媒体に情報を書き込むことができるようにプロセッサに結合され得る。プロセッサに結合している記憶媒体は、回路実装と一体の直接結合であってもよく、直接アクセス、またはダウンローディング技法を使用したデータストリーミングをサポートする、1つまたは複数のインターフェースを利用してもよい。

【００６７】

本発明は、プロセッサにおいて使用するための例示的な実施形態の文脈で開示されたが、上記の説明および添付の特許請求の範囲と矛盾することのない多種多様な実装形態が当業者によって採用され得ることが認識されよう。

【符号の説明】

【００６８】

200　圧縮システム
203　プロセッサ複合体
204　ライブラリファイル
206　コンパイラおよびリンカ
208　プロファイルフィードバック情報
210　リンクされる実行可能コード
212　データセクション
214　スタティック型の記憶デバイス
216　変換ツール
218　圧縮されたコード
220　復号テーブル
222　メインメモリ
224　L2キャッシュ
226　プロセッサコア
228　デコーダ
230　変換メモリ
232　レベル1命令キャッシュ
234　実行パイプライン

【図１】

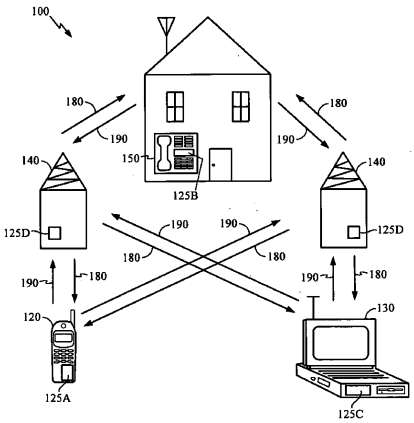
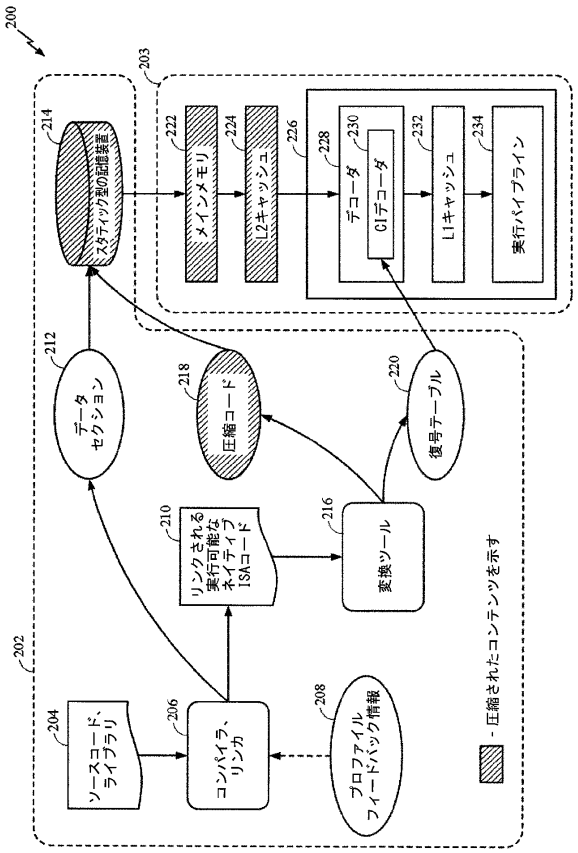
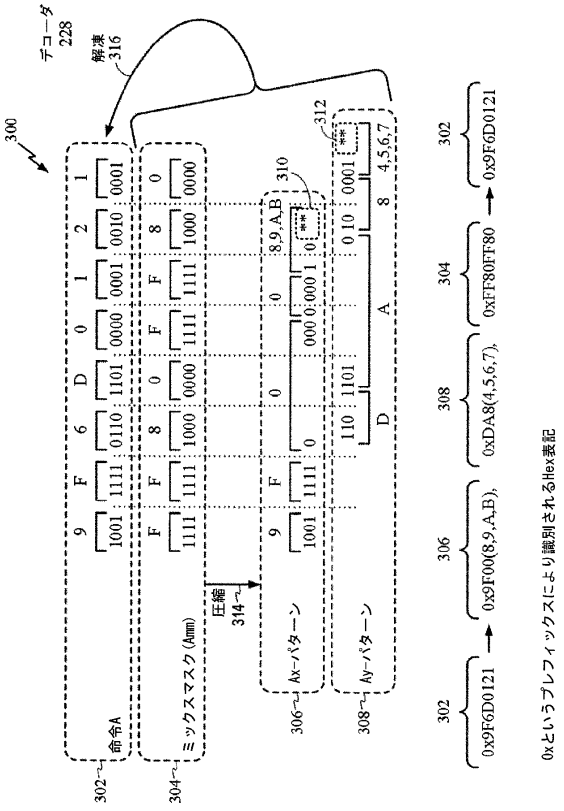


FIG. 1

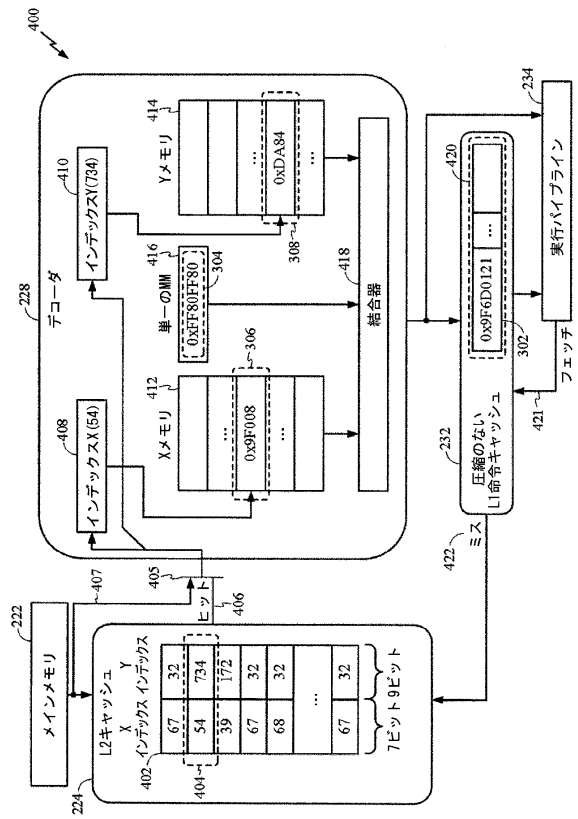
【図２】



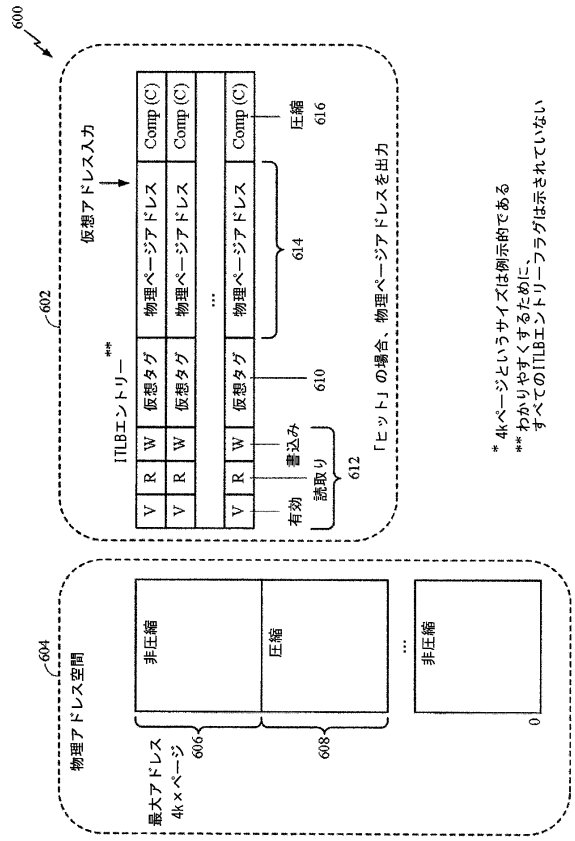
【図３】



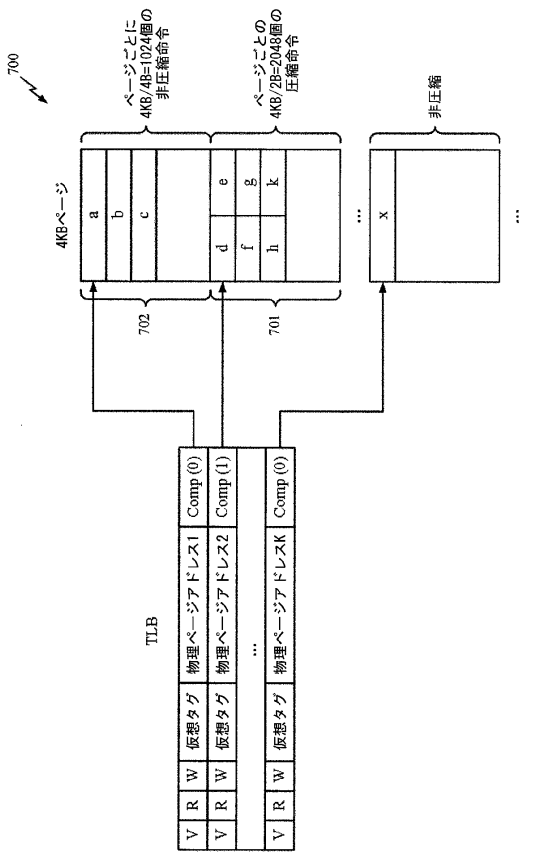
【図４】



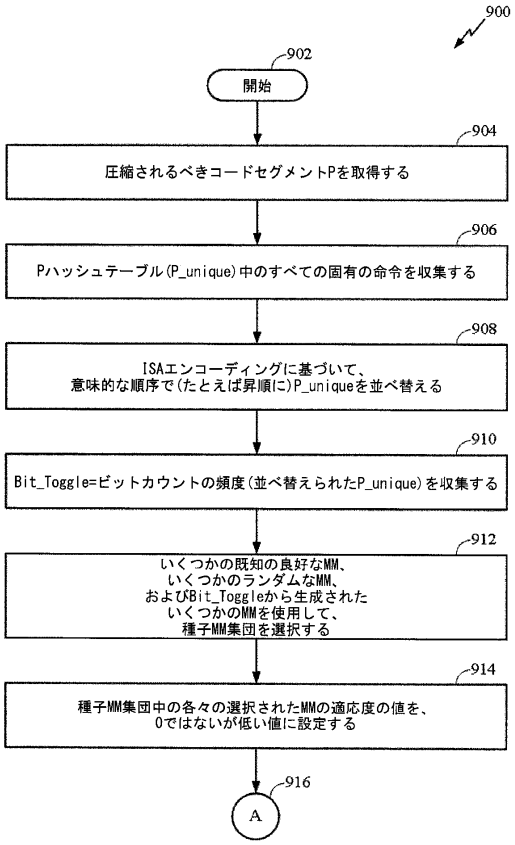
【図６】


600
602
仮想アドレス入力
ITLBエントリー***
V R W 仮想タグ 物理ページアドレス Comp (C)
圧縮
616
614
610
612
有効
読取り
書込み
「ヒット」の場合、物理ページアドレスを出力
* 4kページというサイズは例示的である
** わかりやすくするために、すべてのITLBエントリーフラグは示されていない
604
物理アドレス空間
非圧縮
圧縮
非圧縮
最大アドレス 4k×ページ
606
608
0

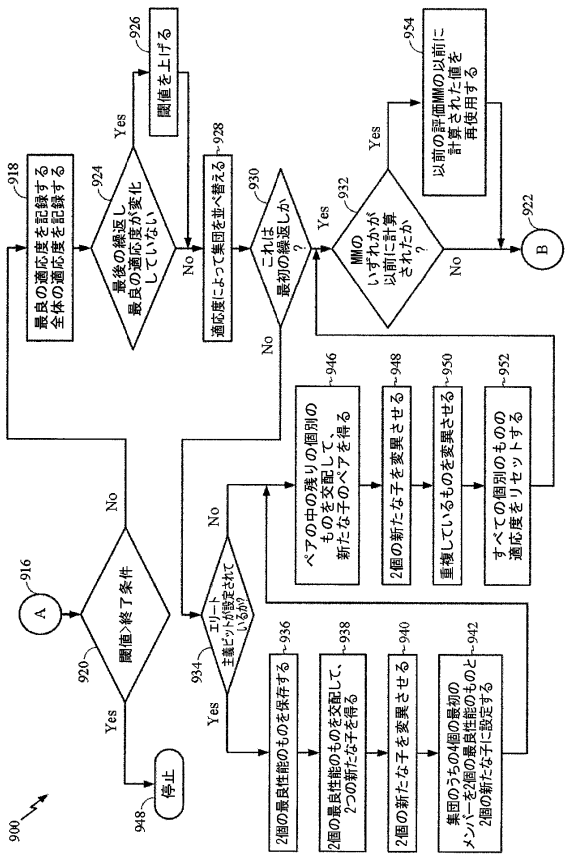
【図７】


700
ページごとに 4KB/4B=1024個の 非圧縮命令
ページごとの 4KB/2B=2048個の 圧縮命令
非圧縮
4KBページ
a
b
c
d
e
f
g
h
k
x
702
701
TLB
V R W 仮想タグ 物理ページアドレス1 Comp (0)
V R W 仮想タグ 物理ページアドレス2 Comp (1)
V R W 仮想タグ 物理ページアドレスK Comp (0)

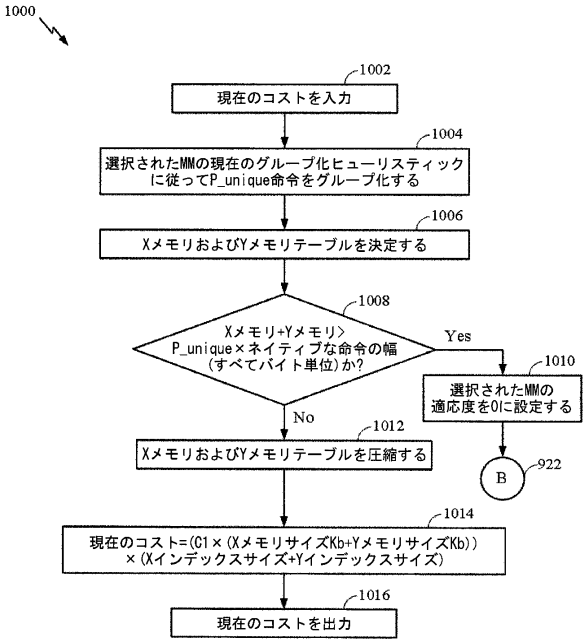
【図９Ａ】


900
902
開始
904
圧縮されるべきコードセグメントPを取得する
906
Pハッシュテーブル(P_unique)中のすべての固有の命令を収集する
908
ISAエンコーディングに基づいて、
意味的な順序で(たとえば昇順に)P_uniqueを並べ替える
910
Bit_Toggle=ビットカウントの頻度(並べ替えられたP_unique)を収集する
912
いくつかの既知の良好なMM、
いくつかのランダムなMM、
およびBit_Toggleから生成された
いくつかのMMを使用して、
種子MM集団を選択する
914
種子MM集団中の各々の選択されたMMの適応度の値を、
0ではないが低い値に設定する
916
A

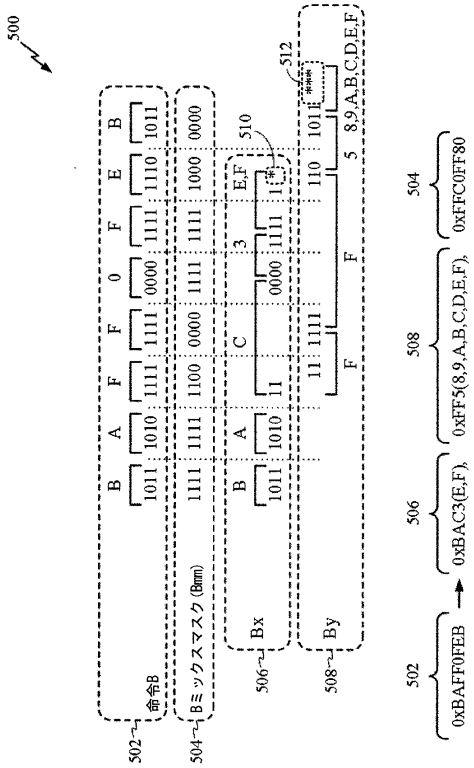
【図９Ｂ】


900
A
916
閾値>終了条件
No
Yes
920
停止
948
最良の適応度を記録する
全体の適応度を記録する
918
最後の繰返し
最良の適応度が変化
していない
924
Yes
No
閾値を上げる
926
適応度によって集団を並べ替える
928
これは
最初の繰返しか
?
930
Yes
No
MMの
いずれかが
以前に計算
されたか
?
932
Yes
No
以前の評価MMの以前に
計算された値を
再使用する
954
B
922
エリート
主義ビットが設定されて
いるか?
934
No
Yes
2個の最良性能のものを保存する
936
2個の最良性能のものを交配して、
2つの新たな子を得る
938
2個の新たな子を変異させる
940
集団のうちの4個の最初の
メンバーをの2個の最良性能のものと
2個の新たな子に設定する
942
ペアの中の残りの個別の
ものを交配して、
新たな子のペアを得る
946
2個の新たな子を変異させる
948
重複しているものを変異させる
950
すべての個別のものの
適応度をリセットする
952

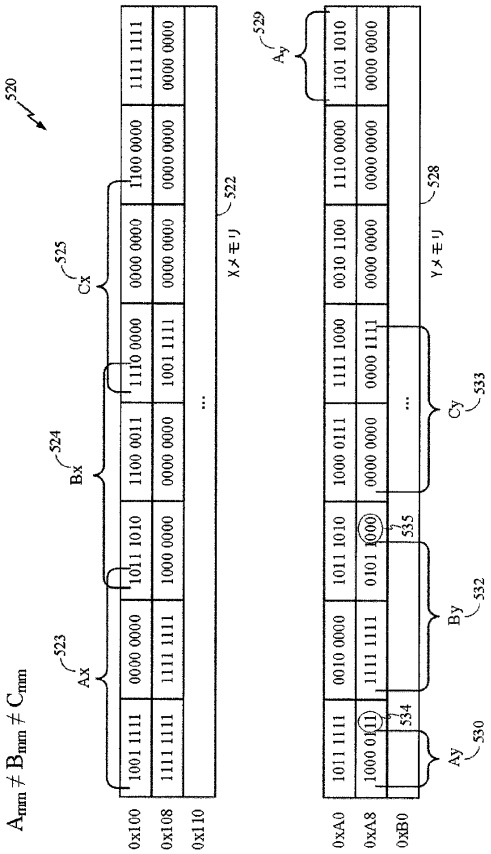
【図１０】



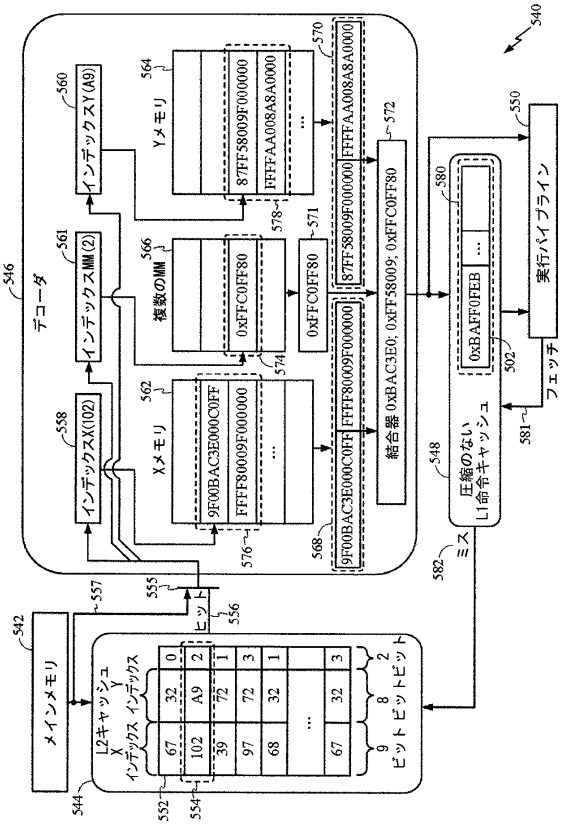
【図５Ａ】



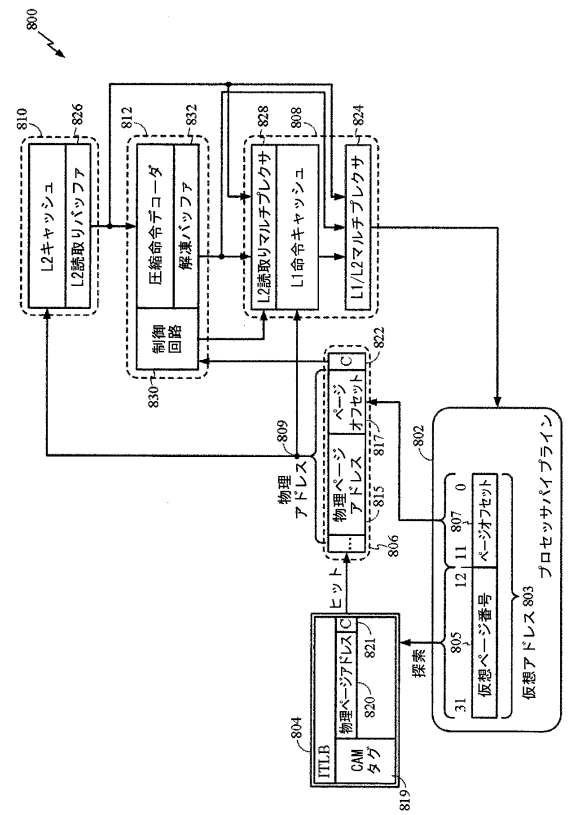
【図５Ｂ】



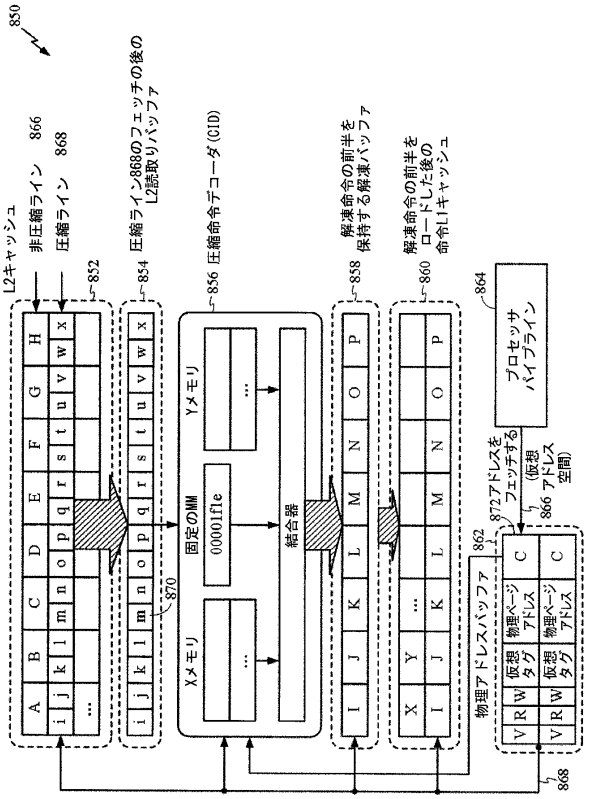
【図５Ｃ】



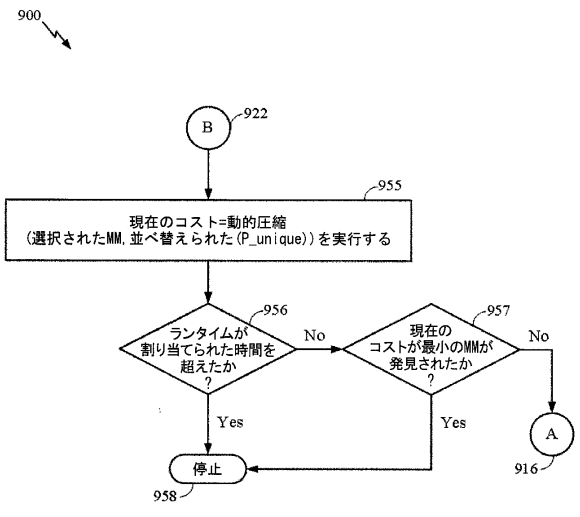
【図８Ａ】



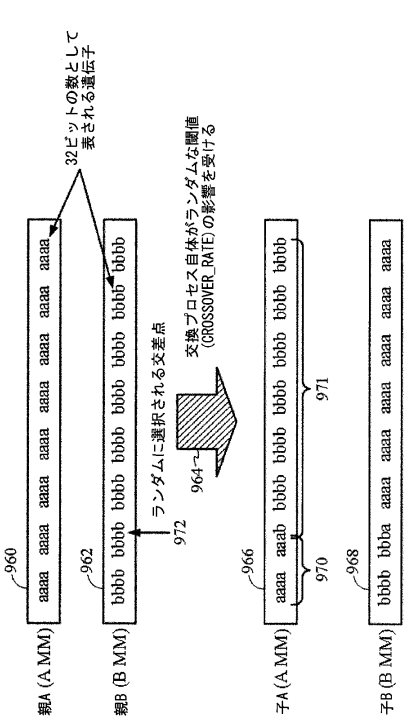
【図８Ｂ】



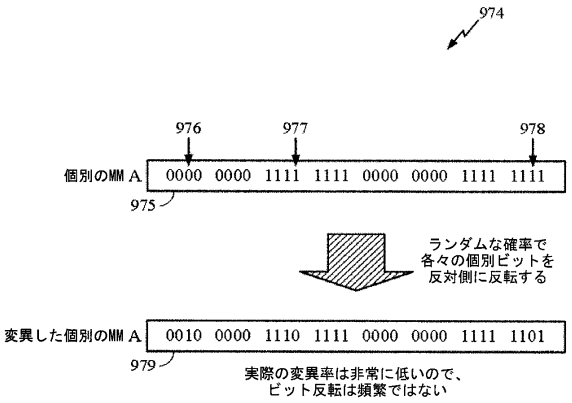
【図９Ｃ】



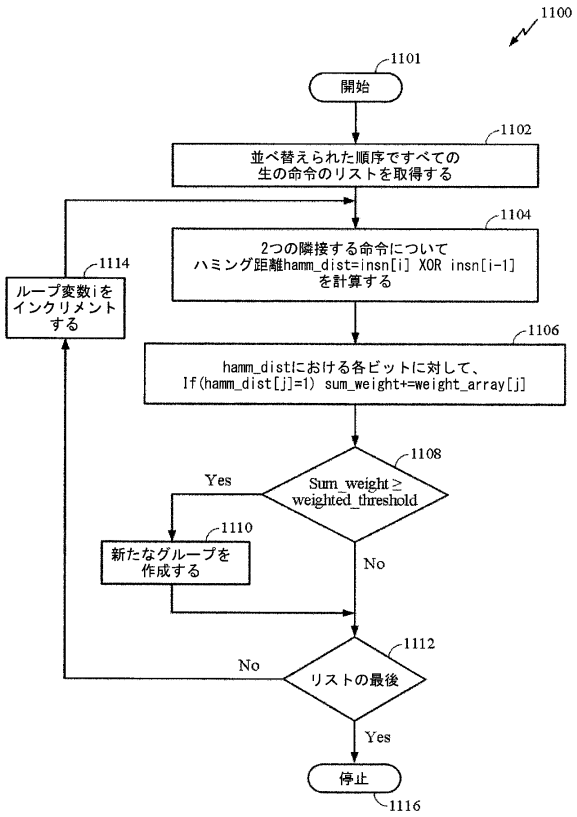
【図９Ｄ】



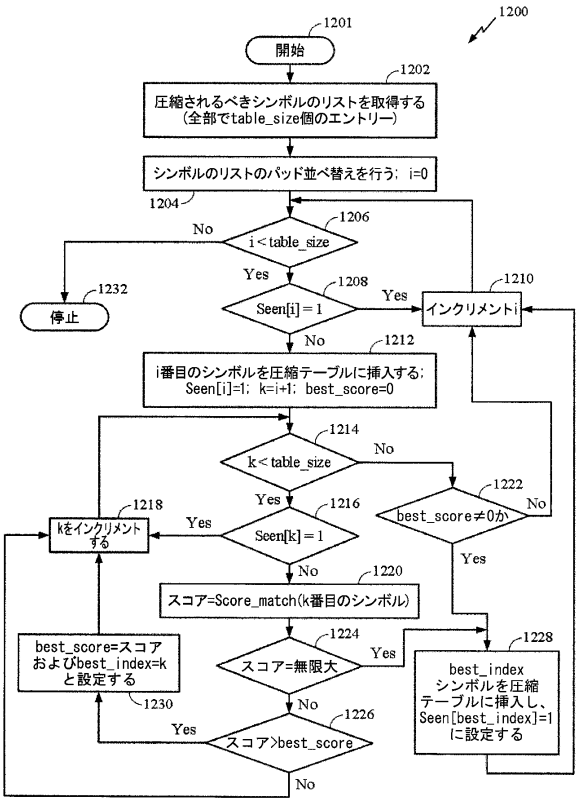
【図９Ｅ】



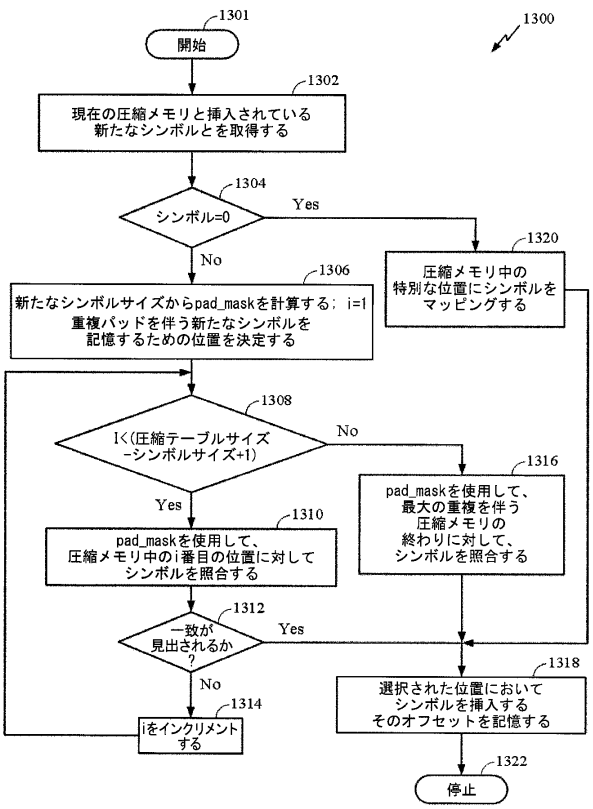
【図１１】



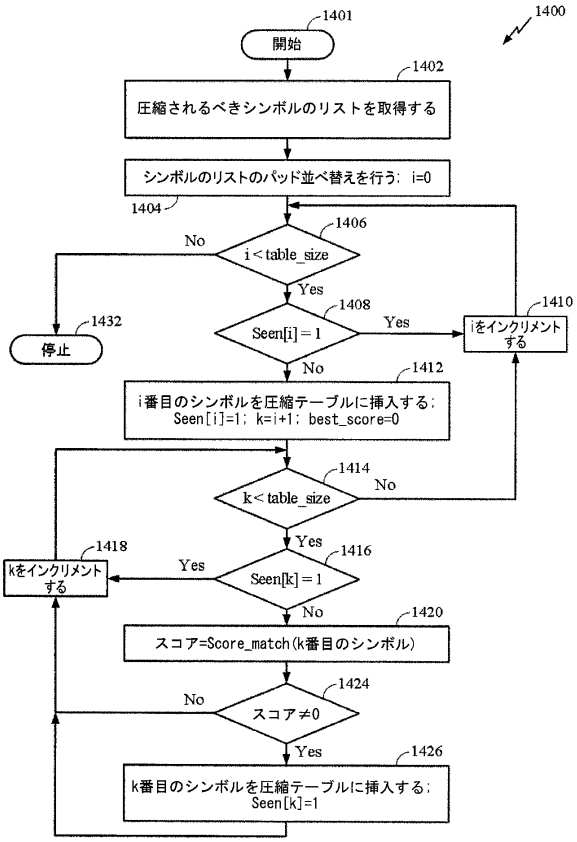
【図１２】



【図１３】



【図１４】


1400
1401
開始
1402
圧縮されるべきシンボルのリストを取得する
シンボルのリストのパッド並べ替えを行う; i=0
1404
1406
No
i＜table_size
Yes
1408
Seen[i]＝1
Yes
1410
iをインクリメントする
1432
停止
No
1412
i番目のシンボルを圧縮テーブルに挿入する;
Seen[i]=1; k=i+1; best_score=0
1414
k＜table_size
No
Yes
1418
kをインクリメントする
Yes
1416
Seen[k]＝1
No
1420
スコア=Score_match(k番目のシンボル)
1424
No
スコア≠0
Yes
1426
k番目のシンボルを圧縮テーブルに挿入する;
Seen[k]=1

フロントページの続き

(51)Int.Cl. FI
G06F 9/30 310B

(72)発明者 ルシアン・コドレスク
アメリカ合衆国・カリフォルニア・９２１２１・サン・ディエゴ・モアハウス・ドライヴ・５７７５
(72)発明者 アンシュマン・ダスグプタ
アメリカ合衆国・カリフォルニア・９２１２１・サン・ディエゴ・モアハウス・ドライヴ・５７７５

審査官 三坂 敏夫

(56)参考文献 米国特許出願公開第２００４／００１５９３１（ＵＳ，Ａ１）
米国特許出願公開第２００６／０１５０１７０（ＵＳ，Ａ１）
米国特許出願公開第２０１１／００８３００１（ＵＳ，Ａ１）
特表２００２－５０６２６４（ＪＰ，Ａ）
特表２００８－５３５５２８（ＪＰ，Ａ）
特開平０９－０１６４７１（ＪＰ，Ａ）
米国特許第０６１９９１２６（ＵＳ，Ｂ１）
特開平７－１６８６９４（ＪＰ，Ａ）
ウォルフ ウェイン，組込みシステム設計の基礎，日本，日経ＢＰ社，２００９年 3月１６日，第1版，330頁

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／３１８
Ｇ０６Ｆ ９／３０
Ｇ０６Ｆ １２／０８
Ｇ０６Ｆ １２／１０