(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2021-108129
(P2021-108129A)
(43)公開日　令和3年7月29日(2021.7.29)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 11/34	(2006.01)	G06F	11/34	176	5B042

審査請求 有　請求項の数 21　ＯＬ　外国語出願　（全 34 頁）

(21)出願番号　特願2021-28781 (P2021-28781)
(22)出願日　令和3年2月25日 (2021.2.25)
(62)分割の表示　特願2019-539979 (P2019-539979)の分割
原出願日　平成29年10月20日 (2017.10.20)
(31)優先権主張番号　15/473,101
(32)優先日　平成29年3月29日 (2017.3.29)
(33)優先権主張国・地域又は機関
米国 (US)

(71)出願人　502208397
グーグル　エルエルシー
Ｇｏｏｇｌｅ　ＬＬＣ
アメリカ合衆国　カリフォルニア州　９４０４３　マウンテン　ビュー　アンフィシアター　パークウェイ　１６００
１６００　Ａｍｐｈｉｔｈｅａｔｒｅ　Ｐａｒｋｗａｙ　９４０４３　Ｍｏｕｎｔａｉｎ　Ｖｉｅｗ，　ＣＡ　Ｕ．Ｓ．Ａ．
(74)代理人　110001195
特許業務法人深見特許事務所
(72)発明者　ノリー，トーマス
アメリカ合衆国、９４０４３　カルフォルニア州、マウンテン・ビュー、アンフィシアター・パークウェイ、１６００
最終頁に続く

(54)【発明の名称】分散ハードウェアトレーシング

(57)【要約】　（修正有）
【課題】プログラムコードの実行を分析する方法を提供する。
【解決手段】方法は、第１のプロセッサコンポーネントによって実行されるプログラムコードの実行を監視するステップと、第２のプロセッサコンポーネントによって実行されるプログラムコードの実行を監視するステップと、を含む。コンピューティングシステム１００は、ハードウェアイベントを識別するデータをＳＰＣ１０６のメモリバッファに格納する。ハードウェアイベントは各々、イベントを特徴付けるメタデータ及びイベントタイムスタンプを含む。システムは、ハードウェアイベントを識別するデータ構造を生成し、ホストシステム１２６に関連付けられたホストデバイスのメモリバンクに格納し、第１のプロセッサコンポーネント又は第２のプロセッサコンポーネントによって実行されるプログラムコードの性能を分析するために、データ構造をホストシステムによって使用する。
【選択図】図１

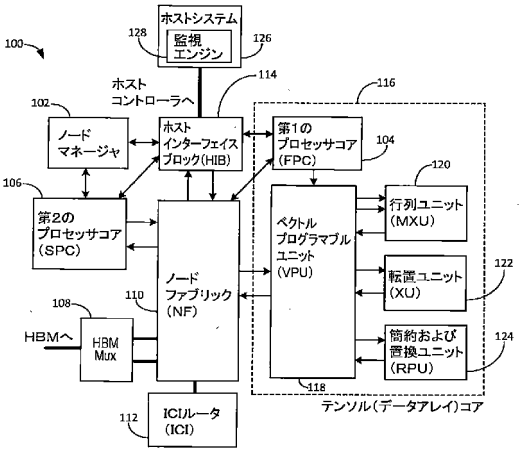


FIG. 1

【特許請求の範囲】
【請求項１】
１つ以上のプロセッサを有するコンピューティングシステムによって実行される、コンピュータにより実現される方法であって、前記方法は、
第１のプロセッサコンポーネントによるプログラムコードの実行を監視するステップを含み、前記第１のプロセッサコンポーネントは、前記プログラムコードの少なくとも第１の部分を実行するように構成され、前記方法はさらに、
第２のプロセッサコンポーネントによる前記プログラムコードの実行を監視するステップを含み、前記第２のプロセッサコンポーネントは、前記プログラムコードの少なくとも第２の部分を実行するように構成され、前記方法はさらに、
前記コンピューティングシステムが、前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントを含むプロセッサユニット間にわたって生じる１つ以上のハードウェアイベントを識別するデータを格納するステップを含み、各ハードウェアイベントは、前記プログラムコードのメモリアクセス動作、前記プログラムコードの発行済命令、または前記プログラムコードの実行済命令に関連付けられたデータ通信のうちの少なくとも１つを表わしており、前記１つ以上のハードウェアイベントの各々を識別する前記データは、前記ハードウェアイベントを特徴付けるメタデータおよびハードウェアイベントタイムスタンプを含み、前記方法はさらに、
前記コンピューティングシステムが、前記１つ以上のハードウェアイベントを識別するデータ構造を生成するステップを含み、前記データ構造は、前記１つ以上のハードウェアイベントを、少なくとも前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントに関連付けられたイベントの時系列の順序で配置するように構成され、前記方法はさらに、
前記コンピューティングシステムが、生成された前記データ構造を、ホストデバイスのメモリバンクに格納するステップを含む、方法。
【請求項２】
前記コンピューティングシステムが、前記第１のプロセッサコンポーネントまたは前記第２のプロセッサコンポーネントのうちの少なくとも１つによって実行されるプログラムコードの一部に関連付けられたトリガー機能を検出するステップと、
前記トリガー機能を検出するステップに応答して、前記コンピューティングシステムが、前記１つ以上のハードウェアイベントに関連付けられたデータを少なくとも１つのメモリバッファに格納させる少なくとも１つのトレースイベントを始動するステップとをさらに含む、請求項１に記載の方法。
【請求項３】
前記トリガー機能は、前記プログラムコードにおける特定のシーケンスステップ、または、前記プロセッサユニットによって使用されるグローバルタイムクロックによって示される特定の時間パラメータ、のうちの少なくとも１つに対応し、
前記少なくとも１つのトレースイベントを始動するステップは、トレースビットが特定の値に設定されていると判定するステップを含み、前記少なくとも１つのトレースイベントは、前記プロセッサユニット間にわたって生じる複数の中間動作を含むメモリアクセス動作に関連付けられており、前記トレースビットが前記特定の値に設定されていると判定するステップに応答して、前記複数の中間動作に関連付けられたデータが１つ以上のメモリバッファに格納される、請求項２に記載の方法。
【請求項４】
前記１つ以上のハードウェアイベントを識別するデータを格納するステップはさらに、前記第１のプロセッサコンポーネントの第１のメモリバッファに、前記１つ以上のハードウェアイベントのハードウェアイベントを識別するデータの第１の部分集合を格納するステップを含み、格納するステップは、前記第１のプロセッサコンポーネントが、前記プログラムコードの少なくとも前記第１の部分に関連付けられたハードウェアトレース命令を実行することに応答して生じる、請求項１に記載の方法。

【請求項５】
前記１つ以上のハードウェアイベントを識別するデータを格納するステップはさらに、前記第２のプロセッサコンポーネントの第２のメモリバッファに、前記１つ以上のハードウェアイベントのハードウェアイベントを識別するデータの第２の部分集合を格納するステップを含み、格納するステップは、前記第２のプロセッサコンポーネントが、前記プログラムコードの少なくとも前記第２の部分に関連付けられたハードウェアトレース命令を実行することに応答して生じる、請求項４に記載の方法。
【請求項６】
前記データ構造を生成するステップはさらに、
前記コンピューティングシステムが、ハードウェアイベントを識別するデータの前記第１の部分集合におけるそれぞれのイベントの少なくともハードウェアイベントタイムスタンプを、ハードウェアイベントを識別するデータの前記第２の部分集合におけるそれぞれのイベントの少なくともハードウェアイベントタイムスタンプと比較するステップと、
前記コンピューティングシステムが、前記第１の部分集合における前記それぞれのイベントと前記第２の部分集合における前記それぞれのイベントとの比較に部分的に基づいて、相関された一組のハードウェアイベントを、前記データ構造における提示のために提供するステップとを含む、請求項５に記載の方法。
【請求項７】
生成された前記データ構造は、特定のハードウェアイベントの待ち時間属性を示す少なくとも１つのパラメータを識別し、前記待ち時間属性は少なくとも、前記特定のハードウェアイベントの持続時間を示す、請求項１に記載の方法。
【請求項８】
前記コンピューティングシステムの少なくとも１つのプロセッサは、１つ以上のプロセッサコンポーネントを有するマルチコアマルチノードプロセッサであり、前記１つ以上のハードウェアイベントは、少なくとも第１のノードの前記第１のプロセッサコンポーネントと第２のノードの前記第２のプロセッサコンポーネントとの間で生じるデータ転送に部分的に対応する、請求項１に記載の方法。
【請求項９】
前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントは、前記コンピューティングシステムのプロセッサ、プロセッサコア、メモリアクセスエンジン、またはハードウェア機能のうちの１つであり、前記１つ以上のハードウェアイベントは、ソースと宛先との間のデータパケットの移動に部分的に対応し、
前記ハードウェアイベントを特徴付けるメタデータは、ソースメモリアドレス、宛先メモリアドレス、一意的なトレース識別番号、または、直接メモリアクセス（ＤＭＡ）トレースに関連付けられたサイズパラメータ、のうちの少なくとも１つに対応する、請求項１に記載の方法。
【請求項１０】
分散ハードウェアトレーシングシステムであって、
１つ以上のプロセッサコアを含む１つ以上のプロセッサと、
命令を格納するための１つ以上の機械読取可能記憶ユニットとを含み、前記命令は、動作を行なうために前記１つ以上のプロセッサによって実行可能であり、前記動作は、
第１のプロセッサコンポーネントによるプログラムコードの実行を監視することを含み、前記第１のプロセッサコンポーネントは、前記プログラムコードの少なくとも第１の部分を実行するように構成され、前記動作はさらに、
第２のプロセッサコンポーネントによる前記プログラムコードの実行を監視することを含み、前記第２のプロセッサコンポーネントは、前記プログラムコードの少なくとも第２の部分を実行するように構成され、前記動作はさらに、
コンピューティングシステムが、前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントを含むプロセッサユニット間にわたって生じる１つ以上のハードウェアイベントを識別するデータを格納することを含み、各ハードウェアイベントは

、前記プログラムコードのメモリアクセス動作、前記プログラムコードの発行済命令、または前記プログラムコードの実行済命令に関連付けられたデータ通信のうちの少なくとも１つを表わしており、前記１つ以上のハードウェアイベントの各々を識別する前記データは、前記ハードウェアイベントを特徴付けるメタデータおよびハードウェアイベントタイムスタンプを含み、前記動作はさらに、

前記コンピューティングシステムが、前記１つ以上のハードウェアイベントを識別するデータ構造を生成することを含み、前記データ構造は、前記１つ以上のハードウェアイベントを、少なくとも前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントに関連付けられたイベントの時系列の順序で配置するように構成され、前記動作はさらに、

前記コンピューティングシステムが、生成された前記データ構造を、ホストデバイスのメモリバンクに格納することを含む、分散ハードウェアトレーシングシステム。

【請求項１１】

前記動作はさらに、

前記コンピューティングシステムが、前記第１のプロセッサコンポーネントまたは前記第２のプロセッサコンポーネントのうちの少なくとも１つによって実行されるプログラムコードの一部に関連付けられたトリガー機能を検出することと、

前記トリガー機能を検出することに応答して、前記コンピューティングシステムが、前記１つ以上のハードウェアイベントに関連付けられたデータを少なくとも１つのメモリバッファに格納させる少なくとも１つのトレースイベントを始動することとを含む、請求項１０に記載の分散ハードウェアトレーシングシステム。

【請求項１２】

前記トリガー機能は、前記プログラムコードにおける特定のシーケンスステップ、または、前記プロセッサユニットによって使用されるグローバルタイムクロックによって示される特定の時間パラメータ、のうちの少なくとも１つに対応し、

前記少なくとも１つのトレースイベントを始動することは、トレースビットが特定の値に設定されていると判定することを含み、前記少なくとも１つのトレースイベントは、前記プロセッサユニット間にわたって生じる複数の中間動作を含むメモリアクセス動作に関連付けられており、前記トレースビットが前記特定の値に設定されていると判定することに応答して、前記複数の中間動作に関連付けられたデータが１つ以上のメモリバッファに格納される、請求項１１に記載の分散ハードウェアトレーシングシステム。

【請求項１３】

前記１つ以上のハードウェアイベントを識別するデータを格納することはさらに、前記第１のプロセッサコンポーネントの第１のメモリバッファに、前記１つ以上のハードウェアイベントのハードウェアイベントを識別するデータの第１の部分集合を格納することを含み、格納することは、前記第１のプロセッサコンポーネントが、前記プログラムコードの少なくとも前記第１の部分に関連付けられたハードウェアトレース命令を実行することに応答して生じる、請求項１０に記載の分散ハードウェアトレーシングシステム。

【請求項１４】

前記１つ以上のハードウェアイベントを識別するデータを格納することはさらに、前記第２のプロセッサコンポーネントの第２のメモリバッファに、前記１つ以上のハードウェアイベントのハードウェアイベントを識別するデータの第２の部分集合を格納することを含み、格納することは、前記第２のプロセッサコンポーネントが、前記プログラムコードの少なくとも前記第２の部分に関連付けられたハードウェアトレース命令を実行することに応答して生じる、請求項１３に記載の分散ハードウェアトレーシングシステム。

【請求項１５】

前記データ構造を生成することはさらに、

前記コンピューティングシステムが、ハードウェアイベントを識別するデータの前記第１の部分集合におけるそれぞれのイベントの少なくともハードウェアイベントタイムスタンプを、ハードウェアイベントを識別するデータの前記第２の部分集合におけるそれぞれ

のイベントの少なくともハードウェアイベントタイムスタンプと比較することと、
前記コンピューティングシステムが、前記第１の部分集合における前記それぞれのイベントと前記第２の部分集合における前記それぞれのイベントとの比較に部分的に基づいて、相関された一組のハードウェアイベントを、前記データ構造における提示のために提供することとを含む、請求項１４に記載の分散ハードウェアトレーシングシステム。
【請求項１６】
生成された前記データ構造は、特定のハードウェアイベントの待ち時間属性を示す少なくとも１つのパラメータを識別し、前記待ち時間属性は少なくとも、前記特定のハードウェアイベントの持続時間を示す、請求項１０に記載の分散ハードウェアトレーシングシステム。
【請求項１７】
少なくとも１つのプロセッサは、１つ以上の処理コンポーネントを有するマルチコアマルチノードプロセッサであり、前記１つ以上のハードウェアイベントは、少なくとも第１のノードの前記第１のプロセッサコンポーネントと第２のノードの前記第２のプロセッサコンポーネントとの間で生じるデータ通信に部分的に対応する、請求項１０に記載の分散ハードウェアトレーシングシステム。
【請求項１８】
前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントは、前記コンピューティングシステムのプロセッサ、プロセッサコア、メモリアクセスエンジン、またはハードウェア機能のうちの１つであり、
前記１つ以上のハードウェアイベントは、ソースと宛先との間のデータパケットの移動に部分的に対応し、
前記ハードウェアイベントを特徴付けるメタデータは、ソースメモリアドレス、宛先メモリアドレス、特有のトレースイベント識別（ＩＤ）番号、または、直接メモリアクセストレース要求に関連付けられたサイズパラメータ、のうちの少なくとも１つに対応する、請求項１０に記載の分散ハードウェアトレーシングシステム。
【請求項１９】
特定のトレースＩＤ番号が、前記プロセッサユニット間にわたって生じる複数のハードウェアイベントに関連付けられ、前記複数のハードウェアイベントは、特定のメモリアクセス動作に対応しており、前記特定のトレースＩＤ番号は、前記複数のハードウェアイベントのうちの１つ以上のハードウェアイベントを相関させるために使用され、相関に基づいて前記メモリアクセス動作の待ち時間属性を判定するために使用される、請求項１８に記載の分散ハードウェアトレーシングシステム。
【請求項２０】
データ処理デバイスに配置され、コンピュータプログラムで符号化された、非一時的コンピュータ記憶ユニットであって、前記プログラムは、１つ以上のプロセッサによって実行されると、前記１つ以上のプロセッサに動作を行なわせる命令を含み、前記動作は、
第１のプロセッサコンポーネントによるプログラムコードの実行を監視することを含み、前記第１のプロセッサコンポーネントは、前記プログラムコードの少なくとも第１の部分を実行するように構成され、前記動作はさらに、
第２のプロセッサコンポーネントによる前記プログラムコードの実行を監視することを含み、前記第２のプロセッサコンポーネントは、前記プログラムコードの少なくとも第２の部分を実行するように構成され、前記動作はさらに、
コンピューティングシステムが、前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントを含むプロセッサユニット間にわたって生じる１つ以上のハードウェアイベントを識別するデータを格納することを含み、各ハードウェアイベントは、前記プログラムコードのメモリアクセス動作、前記プログラムコードの発行済命令、または前記プログラムコードの実行済命令に関連付けられたデータ通信のうちの少なくとも１つを表わしており、前記１つ以上のハードウェアイベントの各々を識別する前記データは、前記ハードウェアイベントを特徴付けるメタデータおよびハードウェアイベントタイ

ムスタンプを含み、前記動作はさらに、
前記コンピューティングシステムが、前記１つ以上のハードウェアイベントを識別するデータ構造を生成することを含み、前記データ構造は、前記１つ以上のハードウェアイベントを、少なくとも前記第１のプロセッサコンポーネントおよび前記第２のプロセッサコンポーネントに関連付けられたイベントの時系列の順序で配置するように構成され、前記動作はさらに、
前記コンピューティングシステムが、生成された前記データ構造を、ホストデバイスのメモリバンクに格納することを含む、非一時的コンピュータ記憶ユニット。
【発明の詳細な説明】
【技術分野】
【０００１】
関連出願との相互参照
本願は、２０１７年３月２９日に出願された、「同期するハードウェアイベント収集」（Synchronous Hardware Event Collection）と題された米国特許出願第１５／４７２，９３２号、および代理人ドケット番号１６１１３－８１２９００１に関する。米国特許出願第１５／４７２，９３２号の全開示は、その全体がここに引用により明白に援用される。
【背景技術】
【０００２】
背景
この明細書は、プログラムコードの実行を分析することに関する。
【０００３】
分散ハードウェアコンポーネント内で実行される分散ソフトウェアの効果的な性能分析は、複雑な作業である場合がある。分散ハードウェアコンポーネントは、より大きいソフトウェアプログラムまたはプログラムコードの一部を実行するために協働し対話する２つ以上の中央処理装置（Central Processing Unit：ＣＰＵ）（またはグラフィックス処理装置（Graphics Processing Unit：ＧＰＵ））のそれぞれのプロセッサコアである場合がある。
【０００４】
（たとえばＣＰＵまたはＧＰＵ内の）ハードウェアの観点から見ると、性能分析に利用可能である情報または機能としては概して、１）ハードウェア性能カウンタと、２）ハードウェアイベントトレースという２種類がある。
【発明の概要】
【課題を解決するための手段】
【０００５】
概要
概して、この明細書で説明される主題の一局面は、１つ以上のプロセッサによって実行される、コンピュータにより実現される方法において具現化され得る。方法は、第１のプロセッサコンポーネントによるプログラムコードの実行を監視するステップを含み、第１のプロセッサコンポーネントは、プログラムコードの少なくとも第１の部分を実行するように構成され、方法はさらに、第２のプロセッサコンポーネントによるプログラムコードの実行を監視するステップを含み、第２のプロセッサコンポーネントは、プログラムコードの少なくとも第２の部分を実行するように構成される。
【０００６】
方法はさらに、コンピューティングシステムが、第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントを含むプロセッサユニット間にわたって生じる１つ以上のハードウェアイベントを識別するデータを、少なくとも１つのメモリバッファに格納するステップを含む。各ハードウェアイベントは、プログラムコードのメモリアクセス動作、プログラムコードの発行済命令、またはプログラムコードの実行済命令に関連付けられたデータ通信のうちの少なくとも１つを表わす。１つ以上のハードウェアイベントの各

々を識別するデータは、ハードウェアイベントを特徴付けるメタデータおよびハードウェアイベントタイムスタンプを含む。方法は、コンピューティングシステムが、１つ以上のハードウェアイベントを識別するデータ構造を生成するステップを含み、データ構造は、１つ以上のハードウェアイベントを、少なくとも第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントに関連付けられたイベントの時系列の順序で配置するように構成される。

【０００７】

方法はさらに、コンピューティングシステムが、生成されたデータ構造を、少なくとも第１のプロセッサコンポーネントまたは第２のプロセッサコンポーネントによって実行されるプログラムコードの性能を分析する際に使用するために、ホストデバイスのメモリバンクに格納するステップを含む。

【０００８】

これらのおよび他の実現化例は各々オプションで、以下の特徴のうちの１つ以上を含み得る。たとえば、いくつかの実現化例では、方法はさらに、コンピューティングシステムが、第１のプロセッサコンポーネントまたは第２のプロセッサコンポーネントのうちの少なくとも１つによって実行されるプログラムコードの一部に関連付けられたトリガー機能を検出するステップと、トリガー機能を検出するステップに応答して、コンピューティングシステムが、１つ以上のハードウェアイベントに関連付けられたデータを少なくとも１つのメモリバッファに格納させる少なくとも１つのトレースイベントを始動するステップとを含む。

【０００９】

いくつかの実現化例では、トリガー機能は、プログラムコードにおける特定のシーケンスステップ、または、プロセッサユニットによって使用されるグローバルタイムクロックによって示される特定の時間パラメータ、のうちの少なくとも１つに対応し、少なくとも１つのトレースイベントを始動するステップは、トレースビットが特定の値に設定されていると判定するステップを含み、少なくとも１つのトレースイベントは、プロセッサユニット間にわたって生じる複数の中間動作を含むメモリアクセス動作に関連付けられており、トレースビットが特定の値に設定されていると判定するステップに応答して、複数の中間動作に関連付けられたデータが１つ以上のメモリバッファに格納される。

【００１０】

いくつかの実現化例では、１つ以上のハードウェアイベントを識別するデータを格納するステップは、第１のプロセッサコンポーネントの第１のメモリバッファに、１つ以上のハードウェアイベントのハードウェアイベントを識別するデータの第１の部分集合を格納するステップを含む。格納するステップは、第１のプロセッサコンポーネントが、プログラムコードの少なくとも第１の部分に関連付けられたハードウェアトレース命令を実行することに応答して生じる。

【００１１】

いくつかの実現化例では、１つ以上のハードウェアイベントを識別するデータを格納するステップはさらに、第２のプロセッサコンポーネントの第２のメモリバッファに、１つ以上のハードウェアイベントのハードウェアイベントを識別するデータの第２の部分集合を格納するステップを含む。格納するステップは、第２のプロセッサコンポーネントが、プログラムコードの少なくとも第２の部分に関連付けられたハードウェアトレース命令を実行することに応答して生じる。

【００１２】

いくつかの実現化例では、データ構造を生成するステップはさらに、コンピューティングシステムが、ハードウェアイベントを識別するデータの第１の部分集合におけるそれぞれのイベントの少なくともハードウェアイベントタイムスタンプを、ハードウェアイベントを識別するデータの第２の部分集合におけるそれぞれのイベントの少なくともハードウェアイベントタイムスタンプと比較するステップと、コンピューティングシステムが、第１の部分集合におけるそれぞれのイベントと第２の部分集合におけるそれぞれのイベント

との比較に部分的に基づいて、相関された一組のハードウェアイベントを、データ構造における提示のために提供するステップとを含む。

【００１３】

いくつかの実現化例では、生成されたデータ構造は、特定のハードウェアイベントの待ち時間属性を示す少なくとも１つのパラメータを識別し、待ち時間属性は少なくとも、特定のハードウェアイベントの持続時間を示す。いくつかの実現化例では、コンピューティングシステムの少なくとも１つのプロセッサは、１つ以上のプロセッサコンポーネントを有するマルチコアマルチノードプロセッサであり、１つ以上のハードウェアイベントは、少なくとも第１のノードの第１のプロセッサコンポーネントと第２のノードの第２のプロセッサコンポーネントとの間で生じるデータ転送に部分的に対応する。

【００１４】

いくつかの実現化例では、第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントは、コンピューティングシステムのプロセッサ、プロセッサコア、メモリアクセスエンジン、またはハードウェア機能のうちの１つであり、１つ以上のハードウェアイベントは、ソースと宛先との間のデータパケットの移動に部分的に対応しており、ハードウェアイベントを特徴付けるメタデータは、ソースメモリアドレス、宛先メモリアドレス、一意的なトレース識別番号、または、直接メモリアクセス（direct memory access：ＤＭＡ）トレースに関連付けられたサイズパラメータ、のうちの少なくとも１つに対応する。

【００１５】

いくつかの実現化例では、特定のトレースＩＤ番号が、プロセッサユニット間にわたって生じる複数のハードウェアイベントに関連付けられ、複数のハードウェアイベントは、特定のメモリアクセス動作に対応しており、特定のトレースＩＤ番号は、複数のハードウェアイベントのうちの１つ以上のハードウェアイベントを相関させるために使用され、相関に基づいてメモリアクセス動作の待ち時間属性を判定するために使用される。

【００１６】

この明細書で説明される主題の他の局面は、１つ以上のプロセッサコアを含む１つ以上のプロセッサと、命令を格納するための１つ以上の機械読取可能記憶ユニットとを含む、分散ハードウェアトレーシングシステムにおいて具現化され得る。命令は、動作を行なうために１つ以上のプロセッサによって実行可能であり、動作は、第１のプロセッサコンポーネントによるプログラムコードの実行を監視することを含み、第１のプロセッサコンポーネントは、プログラムコードの少なくとも第１の部分を実行するように構成され、動作はさらに、第２のプロセッサコンポーネントによるプログラムコードの実行を監視することを含み、第２のプロセッサコンポーネントは、プログラムコードの少なくとも第２の部分を実行するように構成される。

【００１７】

方法はさらに、コンピューティングシステムが、第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントを含むプロセッサユニット間にわたって生じる１つ以上のハードウェアイベントを識別するデータを、少なくとも１つのメモリバッファに格納することを含む。各ハードウェアイベントは、プログラムコードのメモリアクセス動作、プログラムコードの発行済命令、またはプログラムコードの実行済命令に関連付けられたデータ通信のうちの少なくとも１つを表わす。１つ以上のハードウェアイベントの各々を識別するデータは、ハードウェアイベントを特徴付けるメタデータおよびハードウェアイベントタイムスタンプを含む。方法は、コンピューティングシステムが、１つ以上のハードウェアイベントを識別するデータ構造を生成することを含み、データ構造は、１つ以上のハードウェアイベントを、少なくとも第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントに関連付けられたイベントの時系列の順序で配置するように構成される。

【００１８】

方法はさらに、コンピューティングシステムが、生成されたデータ構造を、少なくとも

第１のプロセッサコンポーネントまたは第２のプロセッサコンポーネントによって実行されるプログラムコードの性能を分析する際に使用するために、ホストデバイスのメモリバンクに格納することを含む。

【００１９】

このおよび他の局面の他の実現化例は、コンピュータ記憶デバイス上で符号化された方法のアクションを行なうように構成された、対応するシステム、装置、およびコンピュータプログラムを含む。１つ以上のコンピュータのシステムは、動作時にシステムにアクションを行なわせる、システムにインストールされたソフトウェア、ファームウェア、ハードウェア、またはそれらの組合せによって、そのように構成され得る。１つ以上のコンピュータプログラムは、データ処理装置によって実行されると当該装置にアクションを行なわせる命令を有することによって、そのように構成され得る。

【００２０】

この明細書で説明される主題は、以下の長所のうちの１つ以上を実現するように、特定の実施形態で実現され得る。説明されるハードウェアトレーシングシステムは、マルチノードマルチコアプロセッサを含む分散処理ユニットによる分散ソフトウェアプログラムの実行中に生じるハードウェアイベントの効率的相関を可能にする。説明されるハードウェアトレーシングシステムはさらに、複数のクロスノード構成でのハードウェアイベント／トレースデータの収集および相関を可能にするメカニズムを含む。

【００２１】

ハードウェアトレーシングシステムは、ハードウェアノブ／機能を通して実行される動的トリガーを使用することによって計算効率を高める。さらに、ハードウェアイベントは、一意的なトレース識別子、イベントタイムスタンプ、イベントソースアドレス、およびイベント宛先アドレスといったイベント記述子を用いて、順序立てて時系列化され得る。そのような記述子は、ソフトウェアプログラマーおよびプロセッサ設計エンジニアが、ソースコード実行中に生じ得るソフトウェアおよびハードウェア性能問題を効果的にデバッグし分析するのを助ける。

【００２２】

この明細書で説明される主題の１つ以上の実現化例の詳細を、添付図面および以下の説明で述べる。主題の他の潜在的な特徴、局面、および長所は、説明、図面、および請求項から明らかになるであろう。

【図面の簡単な説明】

【００２３】

【図１】分散ハードウェアトレーシングのための例示的なコンピューティングシステムのブロック図である。

【図２】分散ハードウェアトレーシングのための例示的なコンピューティングシステムのトレースチェーンおよびそれぞれのノードのブロック図である。

【図３】例示的なトレース多重化設計アーキテクチャおよび例示的なデータ構造のブロック図である。

【図４】分散ハードウェアトレーシングのための例示的なコンピューティングシステムによって実行される直接メモリアクセストレースイベントのためのトレース活動を示すブロック図である。

【図５】分散ハードウェアトレーシングのための例示的なプロセスのプロセスフロー図である。

【００２４】

さまざまな図面における同様の参照番号および名称は、同様の要素を示す。

【発明を実施するための形態】

【００２５】

詳細な説明

この明細書で説明される主題は概して、分散ハードウェアトレーシングに関する。特に、コンピューティングシステムが、１つ以上のプロセッサコアによって実行されるプログ

ラムコードの実行を監視する。たとえば、コンピューティングシステムは、第１のプロセッサコアによって実行されるプログラムコードの実行と、少なくとも第２のプロセッサコアによって実行されるプログラムコードの実行とを監視することができる。コンピューティングシステムは、１つ以上のハードウェアイベントを識別するデータをメモリバッファに格納する。イベントを識別する格納されたデータは、少なくとも第１および第２のプロセッサコアを含む分散プロセッサユニット間にわたって生じるイベントに対応する。

【００２６】

各ハードウェアイベントについて、格納されたデータは、そのハードウェアイベントを特徴付けるメタデータおよびイベントタイムスタンプを含む。システムは、ハードウェアイベントを識別するデータ構造を生成する。データ構造は、イベントを時系列の順序で配置し、イベントを少なくとも第１または第２のプロセッサコアに関連付ける。システムは、データ構造をホストデバイスのメモリバンクに格納し、第１または第２のプロセッサコアによって実行されるプログラムコードの性能を分析するためにデータ構造を使用する。

【００２７】

図１は、分散ハードウェアトレーシングのための例示的なコンピューティングシステム１００のブロック図を示す。この明細書で使用されるように、分散ハードウェアシステムトレーシングは、例示的なプロセッサマイクロチップのコンポーネントおよびサブコンポーネント内に生じるイベントを識別するデータの格納に対応する。さらに、ここで使用されるように、分散ハードウェアシステム（またはトレーシングシステム）は、プロセッサマイクロチップまたは分散処理ユニットの集合のうち、分散実行のために構成されたソフトウェア／プログラムコードのそれぞれの一部を実行するよう協働するプロセッサマイクロチップまたは処理ユニットの集合に対応する。

【００２８】

システム１００は、ソフトウェアプログラムを分散的に実行する、すなわち、プログラムコードの異なる部分をシステム１００の異なる処理ユニットに対して実行することによって実行する１つ以上のプロセッサまたは処理ユニットを有する、分散処理システムであってもよい。処理ユニットは、２つ以上のプロセッサ、プロセッサマイクロチップ、または処理ユニット、たとえば、少なくとも第１の処理ユニットおよび第２の処理ユニットを含み得る。

【００２９】

いくつかの実現化例では、第１の処理ユニットが分散ソフトウェアプログラムのプログラムコードの第１の部分を受信して実行する場合、および、第２の処理ユニットが同じ分散ソフトウェアプログラムのプログラムコードの第２の部分を受信して実行する場合、２つ以上の処理ユニットが分散処理ユニットであり得る。

【００３０】

いくつかの実現化例では、システム１００の異なるプロセッサチップが、分散ハードウェアシステムのそれぞれのノードを形成することができる。代替的な実現化例では、単一のプロセッサチップが、そのプロセッサチップのそれぞれのノードを各々形成することができる１つ以上のプロセッサコアおよびハードウェア機能を含み得る。

【００３１】

たとえば、中央処理装置（ＣＰＵ）の状況では、プロセッサチップは少なくとも２つのノードを含んでいてもよく、各ノードはＣＰＵのそれぞれのコアであってもよい。それに代えて、グラフィカル処理装置（ＧＰＵ）の状況では、プロセッサチップは少なくとも２つのノードを含んでいてもよく、各ノードはＧＰＵのそれぞれのストリーミングマルチプロセッサであってもよい。コンピューティングシステム１００は、複数のプロセッサコンポーネントを含み得る。いくつかの実現化例では、プロセッサコンポーネントは、コンピューティングシステム１００全体のプロセッサチップ、プロセッサコア、メモリアクセスエンジン、または少なくとも１つのハードウェアコンポーネントのうちの少なくとも１つであってもよい。

【００３２】

いくつかの例では、プロセッサコアなどのプロセッサコンポーネントは、実行中のプログラムコードの少なくとも１つの発行済命令に基づいて少なくとも１つの特定の動作を実行するように構成された固定機能コンポーネントであってもよい。他の例では、メモリアクセスエンジン（memory access engine：ＭＡＥ）などのプロセッサコンポーネントは、システム１００の他のプロセッサコンポーネントによって実行されるプログラムコードよりも低い詳細度または粒度でプログラムコードを実行するように構成され得る。

【００３３】

たとえば、プロセッサコアによって実行されるプログラムコードは、ＭＡＥ記述子が生成されてＭＡＥに送信されるようにすることができる。記述子の受信後、ＭＡＥは、ＭＡＥ記述子に基づいてデータ転送動作を実行することができる。いくつかの実現化例では、ＭＡＥによって実行されるデータ転送は、たとえば、システムのあるデータ経路またはインターフェイスコンポーネントを介してシステム１００のあるコンポーネント間でデータを動かすこと、または、システム１００の例示的な構成バスにデータ要求を発行することを含み得る。

【００３４】

いくつかの実現化例では、システム１００の例示的なプロセッサチップの各テンソルノードは、プログラム命令を処理するハードウェアブロック／機能であり得る少なくとも２つの「フロントエンド」を有し得る。以下により詳細に説明されるように、第１のフロントエンドは第１のプロセッサコア１０４に対応することができ、一方、第２のフロントエンドは第２のプロセッサコア１０６に対応することができる。よって、第１および第２のプロセッサコアはまた、ここに第１のフロントエンド１０４および第２のフロントエンド１０６と記載されてもよい。

【００３５】

この明細書で使用されるように、トレースチェーンとは、トレースエントリがシステム１００内の例示的なチップマネージャへの送信のために置かれ得る、特定の物理データ通信バスであってもよい。受信されたトレースエントリは、複数のバイトと複数の２進値または２進数とを含むデータワード／構造であってもよい。このため、「ワード」という記述子は、例示的なプロセッサコアのハードウェアデバイスによって１単位として扱われ得る、固定サイズの２進データ片を示す。

【００３６】

いくつかの実現化例では、分散ハードウェアトレーシングシステムのプロセッサチップは、チップのそれぞれのコアでプログラムコードの一部を各々実行する、マルチコア（すなわち複数のコアを有する）プロセッサである。いくつかの実現化例では、プログラムコードの一部は、例示的な多層ニューラルネットワークの推論作業負荷のためのベクトル化計算に対応することができる。一方、代替的な実現化例では、プログラムコードの一部は概して、従来のプログラミング言語に関連付けられたソフトウェアモジュールに対応することができる。

【００３７】

コンピューティングシステム１００は概して、ノードマネージャ１０２と、第１のプロセッサコア（first processor core：ＦＰＣ）１０４と、第２のプロセッサコア（second processor core：ＳＰＣ）１０６と、ノードファブリック（node fabric：ＮＦ）１１０と、データルータ１１２と、ホストインターフェイスブロック（host interface block：ＨＩＢ）１１４とを含む。いくつかの実現化例では、システム１００は、信号切替、多重化、および逆多重化機能を行なうように構成されたメモリｍｕｘ１０８を含み得る。システム１００はさらに、ＦＰＣ１０４が内部に配置されたテンソルコア１１６を含む。テンソルコア１１６は、ベクトル化計算を多次元データアレイに行なうように構成された例示的な計算デバイスであってもよい。テンソルコア１１６はベクトル処理ユニット（vector processing unit：ＶＰＵ）１１８を含んでいてもよく、それは、行列ユニット（matrix unit：ＭＸＵ）１２０、転置ユニット（transpose unit：ＸＵ）１２２、簡約および置換ユニット（reduction and permutation unit：ＲＰＵ）１２４と対話する。いくつかの

実現化例では、コンピューティングシステム１００は、ロード／格納ユニット、算術論理演算ユニット（arithmetic logic unit：ＡＬＵ）、およびベクトルユニットといった、従来のＣＰＵまたはＧＰＵの１つ以上の実行ユニットを含み得る。

【００３８】

システム１００のコンポーネントは、大きい一組のハードウェア性能カウンタと、コンポーネント内のトレース活動の完了を促進するサポートハードウェアとを一括して含む。以下により詳細に説明されるように、システム１００のそれぞれのプロセッサコアによって実行されるプログラムコードは、コード実行中に複数の性能カウンタを同時にイネーブルにするために使用される埋込みトリガーを含み得る。一般に、検出されたトリガーは、トレースデータが１つ以上のトレースイベントのために生成されるようにする。トレースデータは、カウンタに格納され、プログラムコードの性能特性を識別するために分析され得るインクリメンタルパラメータカウントに対応することができる。それぞれのトレースイベントについてのデータは、例示的な記憶媒体（たとえばハードウェアバッファ）に格納可能であり、トリガーの検出に応答して生成されるタイムスタンプを含み得る。

【００３９】

さらに、トレースデータは、システム１００のハードウェアコンポーネント内に生じるさまざまなイベントについて生成され得る。例示的なイベントは、直接メモリアクセス（ＤＭＡ）動作および同期フラグ更新（各々、以下により詳細に説明される）といった、ノード間およびクロスノード通信動作を含み得る。いくつかの実現化例では、システム１００は、一般にグローバル時間カウンタ（Global Time Counter：ＧＴＣ）と呼ばれる、グローバルに同期するタイムスタンプカウンタを含み得る。他の実現化例では、システム１００は、ランポート（Lamport）クロックといった他のタイプのグローバルクロックを含み得る。

【００４０】

ＧＴＣは、プログラムコード実行と、分散処理環境で実行されるソフトウェア／プログラムコードの性能との正確な相関のために使用され得る。加えて、およびＧＴＣに一部関連して、いくつかの実現化例では、システム１００は、分散システムでデータトレーシングを非常に協調的な態様で起動および停止するために分散ソフトウェアプログラムによって使用される１つ以上のトリガーメカニズムを含み得る。

【００４１】

いくつかの実現化例では、ホストシステム１２６が、埋込まれたオペランドを含み得るプログラムコードをコンパイルする。オペランドは、検出されると、ハードウェアイベントに関連付けられたトレースデータの取込みおよび格納を引き起こすことをトリガーする。いくつかの実現化例では、ホストシステム１２６は、コンパイルされたプログラムコードを、システム１００の１つ以上のプロセッサチップに提供する。代替的な実現化例では、プログラムコードは例示的な外部コンパイラによって（埋込まれたトリガーを用いて）コンパイルされ、システム１００の１つ以上のプロセッサチップにロードされ得る。いくつかの例では、コンパイラは、ソフトウェア命令の一部に埋込まれたあるトリガーに関連付けられた（以下に説明される）１つ以上のトレースビットを設定することができる。コンパイルされたプログラムコードは、システム１００の１つ以上のコンポーネントによって実行される分散ソフトウェアプログラムであってもよい。

【００４２】

ホストシステム１２６は、システム１００の１つ以上のコンポーネントによるプログラムコードの実行を監視するように構成された監視エンジン１２８を含み得る。いくつかの実現化例では、監視エンジン１２８は、ホストシステム１２６が、少なくともＦＰＣ１０４およびＳＰＣ１０６によって実行されるプログラムコードの実行を監視することを可能にする。たとえば、コード実行中、ホストシステム１２６は、少なくとも、生成されたトレースデータに基づいてハードウェアイベントの周期的なタイムラインを受信することによって、監視エンジン１２８を介して実行コードの性能を監視することができる。ホストシステム１２６のために単一のブロックが示されているが、いくつかの実現化例では、シ

ステム１２６は、システム１００の複数のプロセッサチップまたはチップコアに関連付けられる複数のホスト（またはホストサブシステム）を含み得る。

【００４３】

他の実現化例では、データトラフィックがＦＰＣ１０４と例示的な第３のプロセッサコア／ノードとの間の通信経路を横断する際、少なくとも３つのプロセッサコアを伴うクロスノード通信が、ホストシステム１２６に、１つ以上の中間「ホップ」でデータトラフィックを監視させてもよい。たとえば、ＦＰＣ１０４と第３のプロセッサコアとは、所与の期間にプログラムコードを実行する唯一のコアであってもよい。よって、ＦＰＣ１０４から第３のプロセッサコアへのデータ転送は、データがＦＰＣ１０４から第３のプロセッサコアへ転送される際、ＳＰＣ１０６で中間ホップについてのトレースデータを生成することができる。別の言い方をすると、システム１００でのデータルーティング中、第１のプロセッサチップから第３のプロセッサチップに向かうデータは、第２のプロセッサチップを横断する必要があるかもしれず、そのため、データルーティング動作の実行は、トレースエントリが第２のチップでのルーティング活動のために生成されるようにするかもしれない。

【００４４】

コンパイルされたプログラムコードが実行されると、システム１００のコンポーネントは、分散コンピュータシステムで生じるハードウェアイベントのタイムラインを生成するために対話することができる。ハードウェアイベントは、ノード内およびクロスノード通信イベントを含み得る。分散ハードウェアシステムの例示的なノードおよびそれらの関連付けられた通信は、図２を参照して以下により詳細に説明される。いくつかの実現化例では、少なくとも１つのハードウェアイベントタイムラインのためのハードウェアイベントの集合を識別するデータ構造が生成される。タイムラインは、分散システムで生じるイベントの再構築を可能にする。いくつかの実現化例では、イベント再構築は、特定のイベントの発生中に生成されたタイムスタンプの分析に基づく正しいイベント順序付けを含み得る。

【００４５】

一般に、例示的な分散ハードウェアトレーシングシステムは、システム１００の上述のコンポーネントと、ホストシステム１２６に関連付けられた少なくとも１つのホストコントローラとを含み得る。分散トレーシングシステムから得られたデータの性能またはデバッグは、イベントデータがたとえば時系列でまたは順序立てて相関される場合に有用であり得る。いくつかの実現化例では、接続されたソフトウェアモジュールに対応する複数の格納されるハードウェアイベントが格納され、次に、ホストシステム１２６による構造化分析のために順序立てられた場合に、データ相関が生じ得る。複数のホストシステムを含む実現化例については、異なるホストを介して得られたデータの相関は、たとえばホストコントローラによって行なわれてもよい。

【００４６】

いくつかの実現化例では、ＦＰＣ１０４およびＳＰ１０６は各々、１つのマルチコアプロセッサチップの別個のコアである。一方、他の実現化例では、ＦＰＣ１０４およびＳＰ１０６は、別個のマルチコアプロセッサチップのそれぞれのコアである。上述のように、システム１００は、少なくともＦＰＣ１０４およびＳＰＣ１０６を有する分散プロセッサユニットを含み得る。いくつかの実現化例では、システム１００の分散プロセッサユニットは、より大きい分散ソフトウェアプログラムまたはプログラムコードの少なくとも一部を実行するように構成された１つ以上のハードウェアまたはソフトウェアコンポーネントを含み得る。

【００４７】

データルータ１１２は、システム１００のコンポーネント間にデータ通信経路を提供するチップ間相互接続（inter-chip interconnect：ＩＣＩ）である。特に、ルータ１１２は、ＦＰＣ１０４とＳＰＣ１０６との間に、およびコア１０４、１０６に関連付けられたそれぞれのコンポーネント間に、通信結合または接続を提供することができる。ノードフ

ァブリック１１０は、システム１００の分散ハードウェアコンポーネントおよびサブコンポーネント内でデータパケットを動かすために、データルータ１１２と対話する。

【００４８】

ノードマネージャ１０２は、マルチノードプロセッサチップにおける低レベルのノード機能を管理する高レベルのデバイスである。以下により詳細に説明されるように、プロセッサチップの１つ以上のノードは、ハードウェアイベントデータを管理してローカルエントリログに格納するためにノードマネージャ１０２によって制御されるチップマネージャを含み得る。メモリｍｕｘ１０８は、例示的な外部高帯域メモリ（high bandwidth memory：ＨＢＭ）に提供されたデータ信号、または外部ＨＢＭから受信されたデータ信号に対してスイッチング、多重化、および逆多重化動作を行なうことができる多重化デバイスである。

【００４９】

いくつかの実現化例では、ｍｕｘ１０８がＦＰＣ１０４とＳＰＣ１０６とを切り替える際、（以下に説明される）例示的なトレースエントリがｍｕｘ１０８によって生成され得る。メモリｍｕｘ１０８は、ｍｕｘ１０８にアクセスできない特定のプロセッサコア１０４、１０６の性能に影響を与える可能性がある。このため、ｍｕｘ１０８によって生成されたトレースエントリデータは、それぞれのコア１０４、１０６に関連付けられたあるシステム活動の待ち時間において結果として生じるスパイクを理解することの助けとなり得る。いくつかの実現化例では、ｍｕｘ１０８内で発生するハードウェアイベントデータ（たとえば、以下に説明されるトレースポイント）は、例示的なハードウェアイベントタイムラインで、ノードファブリック１１０についてのイベントデータとともにグループ化され得る。あるトレース活動が、複数のハードウェアコンポーネントについてのイベントデータを、例示的なハードウェアバッファ（たとえば、以下に説明されるトレースエントリログ２１８）に格納させる場合、イベントグループ化が生じ得る。

【００５０】

システム１００では、性能分析ハードウェアは、ＦＰＣ１０４、ＳＰＣ１０６、ｍｕｘ１０８、ノードファブリック１１０、データルータ１１２、およびＨＩＢ１１４を包含する。これらのハードウェアコンポーネントまたはユニットの各々は、ハードウェア性能カウンタと、ハードウェアイベントトレース機構および機能とを含む。いくつかの実現化例では、ＶＰＵ１１８、ＭＸＵ１２０、ＸＵ１２２、およびＲＰＵ１２４は、それら自体の専用性能ハードウェアを含んでいない。むしろ、そのような実現化例では、ＦＰＣ１０４は、ＶＰＵ１１８、ＭＸＵ１２０、ＸＵ１２２、およびＲＰＵ１２４のための必要なカウンタを提供するように構成され得る。

【００５１】

ＶＰＵ１１８は、例示的な行列－ベクトルプロセッサのベクトル要素に関連付けられたローカル化高帯域データ処理および算術演算をサポートする内部設計アーキテクチャを含み得る。ＭＸＵ１２０は、たとえば最大１２８×１２８の行列乗算を被乗数のベクトルデータセットに対して行なうように構成された行列乗算ユニットである。

【００５２】

ＸＵ１２２は、たとえば最大１２８×１２８の行列転置演算を、行列乗算演算に関連付けられたベクトルデータに対して行なうように構成された転置ユニットである。ＲＰＵ１２４は、シグマユニットと置換ユニットとを含み得る。シグマユニットは、行列乗算演算に関連付けられたベクトルデータに対して順次簡約を実行する。簡約は、和およびさまざまなタイプの比較演算を含み得る。置換ユニットは、行列乗算演算に関連付けられたベクトルデータのすべての要素を完全に置換し、または複製することができる。

【００５３】

いくつかの実現化例では、システム１００のコンポーネントによって実行されるプログラムコードは、機械学習、ニューラルネットワーク推論計算、および／または１つ以上の直接メモリアクセス機能を代表し得る。システム１００のコンポーネントは、システムの処理ユニットまたはデバイスに１つ以上の機能を実行させる命令を含む１つ以上のソフト

ウェアプログラムを実行するように構成され得る。「コンポーネント」という用語は、あらゆるデータ処理デバイス、または制御ステータスレジスタなどの記憶デバイス、またはデータを処理し格納することができる任意の他のデバイスを含むよう意図される。

【００５４】

システム１００は概して、１つ以上のプロセッサ（たとえばマイクロプロセッサまたは中央処理装置（ＣＰＵ））、グラフィックス処理装置（ＧＰＵ）、特定用途向け集積回路（application specific integrated circuit：ＡＳＩＣ）、または異なるプロセッサの組合せを含み得る、複数の処理ユニットまたはデバイスを含み得る。代替的な実施形態では、システム１００は各々、この明細書で説明されるハードウェアトレース機能に関連する計算を行なうための追加の処理オプションを提供する、他のコンピューティングリソース／デバイス（たとえばクラウドベースのサーバ）を含み得る。

【００５５】

処理ユニットまたはデバイスはさらに、１つ以上のメモリユニットまたはメモリバンク（たとえばレジスタ／カウンタ）を含み得る。いくつかの実現化例では、処理ユニットは、この明細書で説明される１つ以上の機能を行なうために、メモリに格納された、システム１００のデバイスへのプログラミングされた命令を実行する。メモリユニット／バンクは、１つ以上の非一時的機械読取可能記憶媒体を含み得る。非一時的機械読取可能記憶媒体は、ソリッドステートメモリ、磁気ディスク、および光学ディスク、ランダムアクセスメモリ（random access memory：ＲＡＭ）、読取専用メモリ（read-only memory：ＲＯＭ）、消去可能プログラマブル読取専用メモリ（たとえばＥＰＲＯＭ、ＥＥＰＲＯＭ、またはフラッシュメモリ）、もしくは、情報を格納可能な任意の他の有形媒体を含み得る。

【００５６】

図２は、システム１００によって実行される分散ハードウェアトレーシングに使用されるトレースチェーンおよびそれぞれの例示的なノード２００、２０１のブロック図を示す。いくつかの実現化例では、システム１００のノード２００、２０１は、単一のマルチコアプロセッサ内の異なるノードであってもよい。他の実現化例では、ノード２００は第１のマルチコアプロセッサチップにおける第１のノードであってもよく、ノード２０１は第２のマルチコアプロセッサチップにおける第２のノードであってもよい。

【００５７】

図２の実現化例では２つのノードが図示されているが、代替的な実現化例では、システム１００は複数のノードを含み得る。複数のノードを伴う実現化例については、クロスノードデータ転送が、複数のノードを横断する例示的なデータ経路に沿った中間ホップでトレースデータを生成することができる。たとえば、中間ホップは、特定のデータ転送経路における別個のノードを通過するデータ転送に対応することができる。いくつかの例では、１つ以上のノードを通過するクロスノードデータ転送中に生じる１つ以上の中間ホップのために、ＩＣＩトレース／ハードウェアイベントに関連付けられたトレースデータを生成することができる。

【００５８】

いくつかの実現化例では、ノード０およびノード１は、推論作業負荷のためのプログラムコードの一部に関連付けられたベクトル化計算に使用されるテンソルノードである。この明細書で使用されるように、テンソルは多次元の幾何学的オブジェクトであり、例示的な多次元の幾何学的オブジェクトは行列とデータアレイとを含む。

【００５９】

図２の実現化例に示すように、ノード２００は、システム１００のコンポーネントの少なくとも部分集合と対話するトレースチェーン２０３を含む。同様に、ノード２０１は、システム１００のコンポーネントの少なくとも部分集合と対話するトレースチェーン２０５を含む。いくつかの実現化例では、ノード２００、２０１はコンポーネントの同じ部分集合の例示的なノードであり、一方、他の実現化例では、ノード２００、２０１は別個のコンポーネント部分集合のそれぞれのノードである。データルータ／ＩＣＩ１１２は、トレースデータをチップマネージャ２１６に提供するためにトレースチェーン２０３および

２０５と概して収束するトレースチェーン２０７を含む。
【００６０】
図２の実現化例では、ノード２００、２０１は各々、少なくともＦＰＣ１０４、ＳＰＣ１０６、ノードファブリック１１０、およびＨＩＢ１１４を有するそれぞれのコンポーネント部分集合を含み得る。ノード２００、２０１の各コンポーネントは、ノードの特定のコンポーネントによって生成される（以下に説明される）トレースポイントをグループ化するように構成された１つ以上のトレースｍｕｘを含む。ＦＰＣ１０４はトレースｍｕｘ２０４を含み、ノードファブリック１１０はトレースｍｕｘ２１０ａ／ｂを含み、ＳＰＣ１０６はトレースｍｕｘ２０６ａ／ｂ／ｃ／ｄを含み、ＨＩＢ２１４はトレースｍｕｘ２１４を含み、ＩＣＩ２１２はトレースｍｕｘ２１２を含む。いくつかの実現化例では、各トレースｍｕｘのためのトレース制御レジスタは、個々のトレースポイントがイネーブルおよびディスエーブルにされることを可能にする。いくつかの例では、１つ以上のトレースｍｕｘのために、それらの対応するトレース制御レジスタは、個々のイネーブルビットと、より広範なトレースｍｕｘ制御とを含み得る。
【００６１】
一般に、トレース制御レジスタは、トレース命令データを受信して格納する従来の制御ステータスレジスタ（control status register：ＣＳＲ）であってもよい。より広範なトレースｍｕｘ制御に関し、いくつかの実現化例では、システム１００によって実行されるＣＳＲ書込に基づいて、トレーシングがイネーブルおよびディスエーブルにされ得る。いくつかの実現化例では、トレーシングは、グローバル時間カウンタ（ＧＴＣ）の値、ＦＰＣ１０４（またはコア１１６）における例示的なトレースマークレジスタの値に基づいて、もしくは、ＳＰＣ１０６におけるステップマークの値に基づいて、システム１００によって動的に起動および停止され得る。
【００６２】
トレース活動を動的に起動および停止するための、ならびに、同期されたハードウェアイベント収集のための、コンピューティングシステムおよびコンピュータにより実現される方法に関連する詳細および説明は、２０１７年３月２９日に出願された、「同期するハードウェアイベント収集」と題された関連する米国特許出願第１５／４７２，９３２号、および代理人ドケット番号１６１１３－８１２９００１に記載されている。米国特許出願第１５／４７２，９３２号の全開示は、その全体がここに引用により明白に援用される。
【００６３】
いくつかの実現化例では、コア１１６のために、ＦＰＣ１０４は、コア１１６内に生じるイベント活動に関連付けられたトレースウィンドウを定義するためにトレース制御パラメータを使用することができる。トレース制御パラメータは、トレースウィンドウが、ＧＴＣについての下限および上限、ならびに、トレースマークレジスタについての下限および上限によって定義されることを可能にする。
【００６４】
いくつかの実現化例では、システム１００は、トレースイベントフィルタリング機能といった、生成されるトレースエントリの数の減少を可能にする機能を含み得る。たとえば、ＦＰＣ１０４およびＳＰＣ１０６は各々、各コアが（以下に説明される）例示的な生成されたトレース記述子においてトレースビットを設定するレートを制限するフィルタリング機能を含み得る。ＨＩＢ１１４は、あるＤＭＡトレースイベントの取込みに関連付けられたトレースビットを制限する例示的なＤＭＡレートリミッタといった、同様のフィルタリング機能を含み得る。加えて、ＨＩＢ１１４は、ソースＤＭＡトレースエントリをキューに入れる制限のための（たとえばイネーブルビットを介した）制御を含み得る。
【００６５】
いくつかの実現化例では、ＤＭＡ動作用の記述子は、ホストシステム１２６の例示的なコンパイラによって設定されるトレースビットを有し得る。トレースビットが設定されると、トレースデータを判定して生成するハードウェア機能／ノブが、例示的なトレースイベントを完了するために使用される。いくつかの例では、ＤＭＡにおける最後のトレース

ビットは、コンパイラによって静的に挿入されるトレースビットと、特定のハードウェアコンポーネントによって動的に判定されるトレースビットとの論理和演算であり得る。よって、いくつかの例では、コンパイラが生成したトレースビットは、フィルタリングとは別に、生成されるトレースデータの全体量を減少させるためのメカニズムを提供することができる。

【００６６】

たとえば、ホストシステム１２６のコンパイラは、１つ以上のリモートＤＭＡ動作（たとえば、少なくとも２つのノード間にわたるＤＭＡ）のためのトレースビットだけを設定し、１つ以上のローカルＤＭＡ動作（たとえば、ノード２００といった特定のテンソルノード内のＤＭＡ）のためのトレースビットをクリアするよう、決定してもよい。このように、生成されるトレースデータの量は、クロスノード（すなわちリモート）およびローカルＤＭＡ動作の双方を含むトレース活動ではなく、クロスノードＤＭＡ動作に制限されたトレース活動に基づいて減少され得る。

【００６７】

いくつかの実現化例では、システム１００によって始動された少なくとも１つのトレースイベントが、システム１００全体にわたって生じる複数の中間動作を含むメモリアクセス動作に関連付けられ得る。メモリアクセス動作用の記述子（たとえばＭＡＥ記述子）は、複数の中間動作に関連付けられたデータを１つ以上のメモリバッファに格納させるトレースビットを含み得る。このため、トレースビットは、データパケットがシステム１００を横断する際、ＤＭＡ動作の中間ホップで中間メモリ動作に「タグ付け」し、複数のトレースイベントを生成するために使用され得る。

【００６８】

いくつかの実現化例では、ＩＣＩ１１２は、ノード２００、２０１の特定のコンポーネントの各入口および出口ポートのための制御機能性を提供する、１組のイネーブルビットおよび１組のパケットフィルタを含み得る。これらのイネーブルビットおよびパケットフィルタは、ＩＣＩ１１２が、ノード２００、２０１の特定のコンポーネントに関連付けられたトレースポイントをイネーブルおよびディスエーブルにすることを可能にする。トレースポイントをイネーブルおよびディスエーブルにすることに加えて、ＩＣＩ１１２は、イベントソース、イベント宛先、およびトレースイベントパケットタイプに基づいてトレースデータをフィルタリングするように構成され得る。

【００６９】

いくつかの実現化例では、ステップマーカー、ＧＴＣ、またはトレースマーカーを使用することに加えて、プロセッサコア１０４、１０６、およびＨＩＢ１１４のための各トレース制御レジスタはまた、「各自」（everyone）トレースモードを含み得る。この「各自」トレースモードは、プロセッサチップ全体にわたるトレーシングがトレースｍｕｘ２０４またはトレースｍｕｘ２０６ａのいずれかによって制御されることを可能にし得る。各自トレースモード時、トレースｍｕｘ２０４および２０６ａは、その特定のトレースｍｕｘ、すなわち、ｍｕｘ２０４またはｍｕｘ２０６ａのいずれかがトレースウィンドウ内にあるか否かを特定する、「ウィンドウ内」トレース制御信号を送信することができる。

【００７０】

ウィンドウ内トレース制御信号は、たとえば、１つのプロセッサチップ内の、または複数のプロセッサチップ間にわたる他のすべてのトレースｍｕｘに一斉送信され、または全般的に送信され得る。ｍｕｘ２０４またはｍｕｘ２０６ａのいずれかがトレース活動を実行している場合、他のトレースｍｕｘへの一斉送信によって、すべてのトレーシングがイネーブルにされ得る。いくつかの実現化例では、プロセッサコア１０４、１０６、およびＨＩＢ１１４に関連付けられたトレースｍｕｘは各々、「各自トレース」制御信号がいつ、および／またはどのように生成されるかを特定するトレースウィンドウ制御レジスタを含む。

【００７１】

いくつかの実現化例では、トレースｍｕｘ２１０ａ／ｂおよびトレースｍｕｘ２１２に

おけるトレース活動は一般に、トレースビットが、ＩＣＩ／データルータ１１２を横断するＤＭＡ動作または制御メッセージのためにデータワードにおいて設定されるかどうかに基づいて、イネーブルにされる。ＤＭＡ動作または制御メッセージは、ある状況またはソフトウェア状態に基づいて設定された２進データパケット内にトレースビットを有し得る、固定サイズの２進データ構造であってもよい。

【００７２】

たとえば、ＤＭＡ動作がトレースタイプＤＭＡ命令によってＦＰＣ１０４（またはＳＰＣ１０６）で始動され、イニシエータ（プロセッサコア１０４または１０６）がトレースウィンドウ内にある場合、トレースビットはその特定のＤＭＡにおいて設定されるであろう。別の例では、ＦＰＣ１０４について、ＦＰＣ１０４がトレースウィンドウ内にあり、トレースデータを格納させるトレースポイントがイネーブルにされた場合、システム１００内の別のコンポーネントへのデータ書込みのための制御メッセージがトレースビットを設定させるであろう。

【００７３】

いくつかの実現化例では、ゼロレングスＤＭＡ動作が、システム１００内のより広範なＤＭＡ実現化例の一例を提供する。たとえば、いくつかのＤＭＡ動作は、システム１００内に非ＤＭＡ活動を生成することができる。非ＤＭＡ活動の実行も、非ＤＭＡ活動がまるでＤＭＡ動作（たとえば、非ゼロレングス動作を含むＤＭＡ活動）であるかのようにトレースされ得る（たとえば、トレースデータを生成する）。たとえば、ソース位置で始動されたものの、送信または転送されるべきデータがない（たとえばゼロレングスである）ＤＭＡ動作は、代わりに、制御メッセージを宛先位置へ送信してもよい。制御メッセージは、宛先に受信または作業されるべきデータがないことを示すであろう。そして、制御メッセージ自体は、非ゼロレングスＤＭＡ動作がトレースされるように、システム１００によってトレースされるであろう。

【００７４】

いくつかの例では、ＳＰＣ１０６について、ゼロレングスＤＭＡ動作は制御メッセージを生成することができ、ＤＭＡがトレースビットを設定させた場合のみ、すなわち、制御メッセージがゼロレングスを有していなかった場合のみ、そのメッセージに関連付けられたトレースビットが設定される。一般に、ＨＩＢ１１４がトレースウィンドウ内にある場合、ホストシステム１２６から始動されたＤＭＡ動作がトレースビットを設定させるであろう。

【００７５】

図２の実現化例では、トレースチェーン２０３は、ノード０と整列するコンポーネント部分集合についてのトレースエントリデータを受信し、一方、トレースチェーン２０５は、ノード１と整列するコンポーネント部分集合についてのトレースエントリデータを受信する。各トレースチェーン２０３、２０５、２０７は、チップマネージャ２１６の例示的なトレースエントリデータログ２１８にトレースエントリデータを提供するためにそれぞれのノード２００、２０１、およびＩＣＩ１１２によって使用される別個のデータ通信経路である。このため、トレースチェーン２０３、２０５、２０７のエンドポイントは、トレースイベントが例示的なメモリユニットに格納され得るチップマネージャ２１６である。

【００７６】

いくつかの実現化例では、チップマネージャ２１６の少なくとも１つのメモリユニットは１２８ビット幅であり、少なくとも２０，０００個のトレースエントリというメモリ深度を有し得る。代替的な実現化例では、少なくとも１つのメモリユニットは、より大きい、またはより小さいビット幅を有していてもよく、より多い、またはより少ないエントリを格納できるメモリ深度を有していてもよい。

【００７７】

いくつかの実現化例では、チップマネージャ２１６は、受信されたトレースエントリデータを管理するための命令を実行する少なくとも１つの処理デバイスを含み得る。たとえ

ば、チップマネージャ２１６は、トレースチェーン２０３、２０５、２０７を介して受信されたトレースデータのそれぞれのハードウェアイベントについてのタイムスタンプデータを走査／分析するための命令を実行することができる。分析に基づいて、チップマネージャ２１６は、ハードウェアトレースイベントの時系列の順序を識別（または生成）するために使用され得るデータを含むように、トレースエントリログ２１８をポピュレートすることができる。ハードウェアトレースイベントは、システム１００の処理ユニットが例示的な分散ソフトウェアプログラムを実行する際にコンポーネントおよびサブコンポーネントレベルで生じるデータパケットの移動に対応することができる。

【００７８】

いくつかの実現化例では、システム１００のハードウェアユニットは、例示的なハードウェアトレースバッファを非時系列的に（すなわち順不同に）ポピュレートするトレースエントリ（および対応するタイムスタンプ）を生成してもよい。たとえば、チップマネージャ２１６は、生成されたタイムスタンプを有する複数のトレースエントリをエントリログ２１８に挿入させることができる。挿入された複数のトレースエントリのうち、それぞれのトレースエントリは、互いに対して時系列化されていなくてもよい。この実現化例では、非時系列のトレースエントリは、ホストシステム１２６の例示的なホストバッファによって受信され得る。ホストバッファによって受信されると、ホストシステム１２６は、それぞれのトレースエントリについてのタイムスタンプデータを走査／分析するために、性能分析／監視用ソフトウェアに関連する命令を実行することができる。実行された命令は、トレースエントリをソートするために、および、ハードウェアトレースイベントのタイムラインを構築／生成するために使用され得る。

【００７９】

いくつかの実現化例では、トレーシングセッション中、ホストＤＭＡ動作を介して、トレースエントリをエントリログ２１８から除去することができる。いくつかの例では、ホストシステム１２６は、ＤＭＡエントリを、それらがログに追加されるのと同じくらい速く、トレースエントリログ２１８から除去しないかもしれない。他の実現化例では、エントリログ２１８は、予め定義されたメモリ深度を含み得る。エントリログ２１８のメモリ深度制限に達すると、追加のトレースエントリが失われるかもしれない。どのトレースエントリが失われるかを制御するために、エントリログ２１８は、先入れ先出し（first-in-first-out：ＦＩＦＯ）モードで、またはそれに代えて、上書き記録モードで動作することができる。

【００８０】

いくつかの実現化例では、上書き記録モードは、事後デバッグに関連付けられた性能分析をサポートするために、システム１００によって使用され得る。たとえば、プログラムコードは、トレース活動がイネーブルにされ、上書き記録モードがイネーブルにされた状態で、ある期間実行され得る。システム１００内の事後ソフトウェアイベント（たとえばプログラム破損）に応答して、ホストシステム１２６によって実行される監視用ソフトウェアは、プログラム破損前に生じたハードウェアイベントを把握するために、例示的なハードウェアトレースバッファのデータ内容を分析することができる。この明細書で使用されるように、事後デバッグは、コードが破損した後の、または、意図されたように実行される／動作することがおおむねできなくなった後のプログラムコードの分析またはデバッグに関する。

【００８１】

ＦＩＦＯモードでは、エントリログ２１８がいっぱいである場合、および、ホストシステム１２６がある時間枠内の保存されたログエントリを除去する場合、メモリリソースを節約するために、新しいトレースエントリはチップマネージャ２１６のメモリユニットに保存されないかもしれない。一方、上書き記録モードでは、ホストシステム１２６がある時間枠内の保存されたログエントリを除去するためにエントリログ２１８がいっぱいである場合、メモリリソースを節約するために、新しいトレースエントリを、エントリログ２１８内に格納された最も古いトレースエントリに上書きすることができる。いくつかの実

現化例では、トレースエントリは、ＤＭＡ動作がＨＩＢ１１４の処理機能を使用することに応答して、ホストシステム１２６のメモリへ動かされる。

【００８２】

この明細書で使用されるように、トレースポイントは、トレースエントリと、チップマネージャ２１６によって受信され、トレースエントリログ２１８に格納された当該トレースエントリに関連付けられたデータとの生成元である。いくつかの実現化例では、マルチコアマルチノードプロセッサマイクロチップは、チップ内に３つのトレースチェーンを含んでいてもよく、第１のトレースチェーンはチップノード０からトレースエントリを受信し、第２のトレースチェーンはチップノード１からトレースエントリを受信し、第３のトレースチェーンはチップのＩＣＩルータからトレースエントリを受信するようになっている。

【００８３】

各トレースポイントは、そのトレースチェーン内に、それがトレースエントリのヘッダに挿入する一意的なトレース識別番号を有する。いくつかの実現化例では、各トレースエントリは、それが発生したトレースチェーンを、１つ以上のバイト／ビットのデータワードによって示されたヘッダにおいて識別する。たとえば、各トレースエントリは、特定のトレースイベントに関する情報を伝える定義されたフィールドフォーマット（たとえばヘッダ、ペイロードなど）を有するデータ構造を含み得る。トレースエントリにおける各フィールドは、トレースエントリを生成したトレースポイントに適用可能な有用なデータに対応する。

【００８４】

上述のように、各トレースエントリは、トレースエントリログ２１８に関連付けられたチップマネージャ２１６のメモリユニットに書込まれ、または格納され得る。いくつかの実現化例では、トレースポイントが個々にイネーブルまたはディスエーブルにされてもよく、複数のトレースポイントが、同じタイプであるものの異なるトレースポイント識別子を有するトレースエントリを生成してもよい。

【００８５】

いくつかの実現化例では、各トレースエントリタイプは、トレース名と、トレース記述と、トレースエントリ内の特定のフィールドおよび／またはフィールドの集合のための符号化を識別するヘッダとを含み得る。これらの名前、記述、およびヘッダは、トレースエントリが表わすことの記述を一括して提供する。チップマネージャ２１６の観点から見ると、この記述は、特定のトレースエントリが特定のプロセッサチップ内に加わった特定のトレースチェーン２０３、２０５、２０７を識別することもできる。このため、トレースエントリ内のフィールドは、記述に関する（たとえばバイト／ビット単位の）データ片を表わしており、どのトレースポイントが特定のトレースエントリを生成したかを判定するために使用されるトレースエントリ識別子であってもよい。

【００８６】

いくつかの実現化例では、格納されたハードウェアイベントのうちの１つ以上に関連付けられたトレースエントリデータは、ａ）少なくともノード０とノード１との間、ｂ）少なくともノード０内のコンポーネント間、および、ｃ）少なくともノード１内のコンポーネント間に生じるデータ通信に部分的に対応することができる。たとえば、格納されたハードウェアイベントは、１）ノード０のＦＰＣ１０４とノード１のＦＰＣ１０４との間、ノード０のＦＰＣ１０４とノード０のＳＰＣ１０６との間、２）ノード０のＳＰＣ１０６とノード１のＳＰＣ１０６との間、のうちの少なくとも１つで生じるデータ通信に部分的に対応することができる。

【００８７】

図３は、例示的なトレース多重化設計アーキテクチャ３００および例示的なデータ構造３２０のブロック図を示す。トレース多重化設計３００は概して、トレースバス入力３０２、バスアービタ３０４、およびローカルトレースポイントアービタ３０６、バスＦＩＦＯ３０８、少なくとも１つのローカルトレースイベントキュー３１０、共有トレースイベ

ントＦＩＦＯ３１２、およびトレースバスアウト３１４を含む。
【００８８】
多重化設計３００は、システム１００のコンポーネント内に配置された例示的なトレースｍｕｘに対応する。多重化設計３００は、以下の機能性を含み得る。バスイン３０２は、時間アービトレーション論理（たとえばアービタ３０４）によってトレースデータが例示的なトレースチェーンに置かれるようになるまでバスＦＩＦＯ３０８内に一時的に格納されるローカルトレースポイントデータに関連し得る。コンポーネントのための１つ以上のトレースポイントが、トレースイベントデータを、少なくとも１つのローカルトレースイベントキュー３１０に挿入することができる。アービタ３０６は第１のレベルのアービトレーションを提供し、キュー３１０内に格納されたローカルトレースイベントからのイベントの選択を可能にする。選択されたイベントは、格納キューとしても機能する共有トレースイベントＦＩＦＯ３１２に置かれる。
【００８９】
アービタ３０４は、ＦＩＦＯキュー３１２からローカルトレースイベントを受信し、ローカルトレースイベントをトレースバスアウト３１４を介して特定のトレースチェーン２０３、２０５、２０７上に併合する第２のレベルのアービトレーションを提供する。いくつかの実現化例では、トレースエントリは、それらが共有ＦＩＦＯ３１２に併合され得るよりも速く、ローカルキュー３１０に押込まれてもよい。または、それに代えて、トレースエントリは、それらがトレースバス３１４上に併合され得るよりも速く、共有ＦＩＦＯ３１２に押込まれてもよい。これらのシナリオが生じる場合、それぞれのキュー３１０および３１２はトレースデータでいっぱいになるであろう。
【００９０】
いくつかの実現化例では、いずれかのキュー３１０または３１２がトレースデータでいっぱいになると、システム１００は、最新のトレースエントリがドロップされて特定のキューに格納または併合されないように構成され得る。他の実現化例では、あるキュー（たとえばキュー３１０、３１２）がいっぱいになった場合にトレースエントリをドロップするのではなく、システム１００は、もう一度いっぱいになったキューがエントリを受信するための利用可能キュー空間を有するまで、例示的な処理パイプラインをストールさせるように構成され得る。
【００９１】
たとえば、キュー３１０、３１２を使用する処理パイプラインは、十分なまたはしきい値の数のトレースエントリがトレースバス３１４上に併合されるまでストールされ得る。十分なまたはしきい値の数は、１つ以上のトレースエントリがキュー３１０、３１２によって受信されるための利用可能なキュー空間をもたらす、特定の数の併合されたトレースエントリに対応することができる。下流のキュー空間が利用可能になるまで処理パイプラインがストールされる実現化例は、トレースエントリがドロップされるのではなく保たれることに基づいた、より高い忠実度のトレースデータを提供することができる。
【００９２】
いくつかの実現化例では、ローカルトレースキューは、各トレースエントリがローカルキュー３１０において１ヶ所のみを占めるように、トレースエントリによって要求されるのと同じくらい幅が広い。しかしながら、共有トレースＦＩＦＯキュー３１２は、いくつかのトレースエントリが共有キュー３１２において２つの位置を占め得るように、一意的なトレースエントリライン符号化を使用することができる。いくつかの実現化例では、トレースパケットのいずれかのデータがドロップされた場合、部分パケットがトレースエントリログ２１８に現われないように、パケット全体がドロップされる。
【００９３】
一般に、トレースは、システム１００の特定のコンポーネントに関連付けられた活動またはハードウェアイベントのタイムラインである。集合体データである（以下に説明される）性能カウンタとは異なり、トレースは、特定されたトレースウィンドウ中に生じるハードウェア活動についての洞察力を提供する詳細なイベントデータを含む。説明されるハ

ードウェアシステムは、トレースエントリの生成、ハードウェア管理バッファでのトレースエントリの一時的格納、１つ以上のトレースタイプの静的および動的イネーブリング、ならびに、ホストシステム１２６へのトレースエントリデータのストリーミングを含む、分散ハードウェアトレーシングのための大規模なサポートを可能にする。

【００９４】

いくつかの実現化例では、トレースは、ＤＭＡ動作の生成、ＤＭＡ動作の実行、ある命令の発行／実行、または同期フラグの更新といった、システム１００のコンポーネントによって実行されるハードウェアイベントのために生成され得る。いくつかの例では、トレース活動は、システムを通してＤＭＡを追跡するために、または、特定のプロセッサコア上で実行される命令を追跡するために使用され得る。

【００９５】

システム１００は、ハードウェアイベントのタイムラインから１つ以上のハードウェアイベント３２２、３２４を識別する少なくとも１つのデータ構造３２０を生成するように構成され得る。いくつかの実現化例では、データ構造３２０は、１つ以上のハードウェアイベント３２２、３２４を、少なくともＦＰＣ１０４およびＳＰＣ１０６に関連付けられたイベントの時系列の順序で配置する。いくつかの例では、システム１００は、データ構造３２０を、ホストシステム１２６のホスト制御デバイスのメモリバンクに格納することができる。データ構造３２０は、少なくともプロセッサコア１０４および１０６によって実行されるプログラムコードの性能を評価するために使用され得る。

【００９６】

ハードウェアイベント３２４によって示されるように、いくつかの実現化例では、特定のトレース識別（ＩＤ）番号（たとえばトレースＩＤ‘００３）が、分散プロセッサユニット間にわたって生じる複数のハードウェアイベントに関連付けられ得る。複数のハードウェアイベントは、特定のメモリアクセス動作（たとえばＤＭＡ）に対応することができ、特定のトレースＩＤ番号は、１つ以上のハードウェアイベントを相関させるために使用される。

【００９７】

たとえば、イベント３２４によって示されるように、ＤＭＡ動作用の単一のトレースＩＤは、ＤＭＡにおける複数の異なるポイントに対応する複数のタイムスタンプを含み得る。いくつかの例では、トレースＩＤ‘００３は、互いに対して何らかの時間離れているとして識別される、「発行済」イベント、「実行済」イベント、および「完了済」イベントを有することができる。よって、この点に関し、トレースＩＤはさらに、相関に基づいて、およびタイムスタンプを参照して、メモリアクセス動作の待ち時間属性を判定するために使用され得る。

【００９８】

いくつかの実現化例では、データ構造３２０を生成することは、たとえば、システム１００が、ハードウェアイベントの第１の部分集合におけるそれぞれのイベントのイベントタイムスタンプを、ハードウェアイベントの第２の部分集合におけるそれぞれのイベントのイベントタイムスタンプと比較することを含み得る。データ構造３２０を生成することはさらに、システム１００が、イベントの第１の部分集合とイベントの第２の部分集合との比較に部分的に基づいて、相関された一組のハードウェアイベントを、データ構造における提示のために提供することを含み得る。

【００９９】

図３に示すように、データ構造３２０は、特定のハードウェアイベント３２２、３２４の待ち時間属性を示す少なくとも１つのパラメータを識別することができる。待ち時間属性は少なくとも、特定のハードウェアイベントの持続時間を示し得る。いくつかの実現化例では、データ構造３２０は、ホストシステム１２６の制御デバイスによって実行されるソフトウェア命令によって生成される。いくつかの例では、構造３２０は、制御デバイスがトレースエントリデータをホストシステム１２６のメモリディスク／ユニットに格納することに応答して生成され得る。

【０１００】
図４は、システム１００によって実行される直接メモリアクセス（ＤＭＡ）トレースイベントのための例示的なトレース活動を示すブロック図４００である。ＤＭＡトレーシングのために、第１のプロセッサノードから第２のプロセッサノードへと発生する例示的なＤＭＡ動作についてのデータが、ＩＣＩ１１２を介して進むことができ、データ経路に沿って中間ＩＣＩ／ルータホップを生成することができる。ＤＭＡ動作がＩＣＩ１１２を横断する際、ＤＭＡ動作は、プロセッサチップ内の各ノードで、および各ホップに沿ってトレースエントリを生成するであろう。ノードおよびホップに沿ったＤＭＡ動作の時間的推移を再構築するために、これらの生成されたトレースエントリの各々によって情報が取込まれる。

【０１０１】
例示的なＤＭＡ動作は、図４の実現化例に示すプロセスステップに関連付けられ得る。この動作のために、ローカルＤＭＡが、プロセッサコア１０４、１０６の少なくとも１つに関連付けられた仮想メモリ４０２（ｖｍｅｍ４０２）からＨＢＭ１０８へデータを転送する。ブロック図４００に示された番号付けは、表４０４のステップに対応しており、概して、ノードファブリック１１０における活動、またはノードファブリック１１０によって始動される活動を表わす。

【０１０２】
表４０４のステップは概して、関連付けられたトレースポイントを説明する。例示的な動作は、このＤＭＡのために６つのトレースエントリを生成するであろう。ステップ１は、プロセッサコアからノードファブリック１１０への最初のＤＭＡ要求を含み、それはノードファブリックにおいてトレースポイントを生成する。ステップ２は、ノードファブリック１１０がプロセッサコアにデータを転送するよう求める読出コマンドを含み、それはノードファブリック１１０において別のトレースポイントを生成する。ｖｍｅｍ４０２がノードファブリック１１０の読出を完了すると、例示的な動作は、ステップ３のためのトレースエントリを有していない。

【０１０３】
ステップ４は、ノードファブリック１１０が、プロセッサコアにおいて同期フラグ更新を引き起こすために読出リソース更新を行なうことを含み、それはプロセッサコアにおいてトレースポイントを生成する。ステップ５は、次のデータがＨＢＭに書込まれることをノードファブリック１１０がメモリｍｕｘ１０８に通知する書込コマンドを含む。書込コマンドを介した通知は、ノードファブリック１１０においてトレースポイントを生成し、一方、ステップ６で、ＨＢＭへの書込の完了も、ノードファブリック１１０においてトレースポイントを生成する。ステップ７で、ノードファブリック１１０は、プロセッサコアにおいて同期フラグ更新を引き起こすために書込リソース更新を行ない、それはプロセッサコアにおいて（たとえばＦＰＣ１０４において）トレースポイントを生成する。書込リソース更新に加えて、ノードファブリック１１０は、ＤＭＡ動作のためのデータ完了がプロセッサコアに信号で送り返される受信確認更新（ａｃｋ更新）を行なうことができる。ａｃｋ更新は、書込リソース更新によって生成されたトレースエントリと同様のトレースエントリを生成することができる。

【０１０４】
別の例示的なＤＭＡ動作では、ＤＭＡ命令が発生元ノードのノードファブリック１１０において発行されると、第１のトレースエントリが生成される。ＤＭＡについてのデータを読出し、そのデータを送信用キューに書込むために使用される時間を取込むために、追加のトレースエントリがノードファブリック１１０において生成され得る。いくつかの実現化例では、ノードファブリック１１０は、ＤＭＡデータを、より小さいチャンクのデータへとパケット化することができる。より小さいチャンクへとパケット化されたデータのために、読出および書込トレースエントリが、最初のデータチャンクおよび最後のデータチャンクについて生成され得る。オプションで、最初と最後のデータチャンクに加えて、すべてのデータチャンクが、トレースエントリを生成するために設定され得る。

【０１０５】
ＩＣＩホップを要求し得るリモート／非ローカルＤＭＡ動作のために、最初のデータおよび最後のデータチャンクは、ＩＣＩ／ルータ１１２に沿った各中間ホップにおける入口および出口ポイントで、追加のトレースエントリを生成することができる。ＤＭＡデータが宛先ノードに到着すると、以前のノードファブリック１１０エントリと同様のトレースエントリが、宛先ノードで生成される（たとえば、最初と最後のデータチャンクの読出／書込）。いくつかの実現化例では、ＤＭＡ動作の最後のステップは、ＤＭＡに関連付けられた実行された命令が宛先ノードで同期フラグの更新を引き起こすことを含み得る。同期フラグが更新されると、ＤＭＡ動作の完了を示すトレースエントリが生成され得る。

【０１０６】
いくつかの実現化例では、トレースポイントが実行され得るように、ＤＭＡトレーシングが、ＦＰＣ１０４、ＳＰＣ１０６、またはＨＩＢ１１４によって、各コンポーネントがトレースモードである場合に始動される。システム１００のコンポーネントは、トリガーメカニズムを介したＦＰＣ１０４またはＳＰＣ１０６におけるグローバル制御に基づいて、トレースモードに入ることができる。システム１００のコンポーネントによるプログラムコードの実行に関連付けられた特定のアクションまたは状態の発生に応答して、トレースポイントはトリガーする。たとえば、プログラムコードの一部は、システム１００の少なくとも１つのハードウェアコンポーネントによって検出可能である、埋込まれたトリガー機能を含み得る。

【０１０７】
システム１００のコンポーネントは、ＦＰＣ１０４またはＳＰＣ１０６の少なくとも１つによって実行されるプログラムコードの一部に関連付けられたトリガー機能を検出するように構成され得る。いくつかの例では、トリガー機能は、１）実行されたプログラムコードの一部またはモジュールにおける特定のシーケンスステップ、または、２）システム１００の分散プロセッサユニットによって使用されるＧＴＣによって示される特定の時間パラメータ、のうちの少なくとも１つに対応することができる。

【０１０８】
トリガー機能の検出に応答して、システム１００の特定のコンポーネントは、１つ以上のハードウェアイベントに関連付けられたトレースエントリデータをハードウェアコンポーネントの少なくとも１つのメモリバッファに格納させる少なくとも１つのトレースポイント（たとえばトレースイベント）を、始動、トリガー、または実行することができる。上述のように、格納されたトレースデータは次に、少なくとも１つのトレースチェーン２０３、２０５、２０７を経由して、チップマネージャ２１６に提供され得る。

【０１０９】
図５は、システム１００のコンポーネント機能とシステム１００の１つ以上のノード２００、２０１とを使用する分散ハードウェアトレーシングのための例示的なプロセス５００のプロセスフロー図である。このため、プロセス５００は、ノード２００、２０１を含むシステム１００の上述のコンピューティングリソースのうちの１つ以上を使用して実現され得る。

【０１１０】
プロセス５００はブロック５０２で始まり、コンピューティングシステム１００が、（少なくともＦＰＣ１０４およびＳＰＣ１０６を含む）１つ以上のプロセッサコンポーネントによって実行されるプログラムコードの実行を監視するステップを含む。いくつかの実現化例では、トレース活動を生成するプログラムコードの実行は、複数のホストシステム、または単一のホストシステムのサブシステムによって、少なくとも部分的に監視され得る。よって、これらの実現化例では、システム１００は、分散処理ユニット間にわたって生じるハードウェアイベントのためのトレース活動の分析に関連する複数のプロセス５００を行なうことができる。

【０１１１】
いくつかの実現化例では、第１のプロセッサコンポーネントは、監視されるプログラム

コードの少なくとも第１の部分を実行するように構成される。ブロック５０４で、プロセス５００は、コンピューティングシステム１００が、第２のプロセッサコンポーネントによって実行されるプログラムコードの実行を監視するステップを含む。いくつかの実現化例では、第２のプロセッサコンポーネントは、監視されるプログラムコードの少なくとも第２の部分を実行するように構成される。

【０１１２】

コンピューティングシステム１００のコンポーネントは各々、少なくとも１つのメモリバッファを含み得る。プロセス５００のブロック５０６は、システム１００が、１つ以上のハードウェアイベントを識別するデータを、特定のコンポーネントの少なくとも１つのメモリバッファに格納するステップを含む。いくつかの実現化例では、ハードウェアイベントは、少なくとも第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントを含む分散プロセッサユニット間にわたって生じる。ハードウェアイベントを識別する格納されたデータは各々、ハードウェアイベントを特徴付けるメタデータおよびハードウェアイベントタイムスタンプを含み得る。いくつかの実現化例では、ハードウェアイベントの集合はタイムラインイベントに対応する。

【０１１３】

たとえば、システム１００は、システム１００内のソースハードウェアコンポーネントとシステム１００内の宛先ハードウェアコンポーネントとの間のデータパケットの移動に部分的に対応する１つ以上のハードウェアイベントを識別するデータを格納することができる。いくつかの実現化例では、ハードウェアイベントを特徴付ける格納されたメタデータは、１）ソースメモリアドレス、２）宛先メモリアドレス、３）ハードウェアイベントを格納させるトレースエントリに関連する一意的なトレース識別番号、または、４）直接メモリアクセス（ＤＭＡ）トレースエントリに関連付けられたサイズパラメータ、のうちの少なくとも１つに対応することができる。

【０１１４】

いくつかの実現化例では、ハードウェアイベントの集合を識別するデータを格納するステップは、たとえば、少なくとも１つのローカルトレースイベントキュー３１０に対応するＦＰＣ１０４および／またはＳＰＣ１０６のメモリバッファにイベントデータを格納するステップを含む。格納されたイベントデータは、ハードウェアイベントのより大きいタイムラインを生成するために使用され得るハードウェアイベントデータの部分集合を示し得る。いくつかの実現化例では、イベントデータの格納は、ＦＰＣ１０４またはＳＰＣ１０６の少なくとも１つが、システム１００のコンポーネントによって実行されるプログラムコードの一部に関連付けられたハードウェアトレース命令を実行することに応答して生じる。

【０１１５】

プロセス５００のブロック５０８で、システム１００は、ハードウェアイベントの集合から１つ以上のハードウェアイベントを識別する、構造３２０などのデータ構造を生成する。データ構造は、１つ以上のハードウェアイベントを、少なくとも第１のプロセッサコンポーネントおよび第２のプロセッサコンポーネントに関連付けられたイベントの時系列の順序で配置する。いくつかの実現化例では、データ構造は、特定のトレースイベントについてのハードウェアイベントタイムスタンプ、そのトレースイベントに関連付けられたソースアドレス、または、そのトレースイベントに関連付けられたメモリアドレスを識別する。

【０１１６】

プロセス５００のブロック５１０で、システム１００は、生成されたデータ構造を、ホストシステム１２６に関連付けられたホストデバイスのメモリバンクに格納する。いくつかの実現化例では、格納されたデータ構造は、少なくとも第１のプロセッサコンポーネントまたは第２のプロセッサコンポーネントによって実行されるプログラムコードの性能を分析するために、ホストシステム１２６によって使用され得る。同様に、格納されたデータ構造は、システム１００の少なくとも１つのコンポーネントの性能を分析するために、

ホストシステム１２６によって使用され得る。
【０１１７】
たとえば、ユーザまたはホストシステム１２６は、プログラムコード内の特定のソフトウェアモジュールの実行に関連付けられた性能問題があるかどうかを検出または判定するために、データ構造を分析することができる。例示的な問題は、ソフトウェアモジュールが割り当てられた実行時間ウィンドウ内で実行を完了しないことを含み得る。
【０１１８】
さらに、ユーザまたはホストデバイス１２６は、システム１００の特定のコンポーネントがしきい値性能レベルを上回って動作しているか、または下回って動作しているかを検出または判定することができる。コンポーネント性能に関連する例示的な問題は、特定のハードウェアコンポーネントがあるイベントを実行するものの、結果データについての許容可能パラメータ範囲外にある結果データを生成することを含み得る。いくつかの実現化例では、この結果データは、実質的に同様の動作を実行するシステム１００の他の関連するコンポーネントによって生成された結果データと一致していないかもしれない。
【０１１９】
たとえば、プログラムコードの実行中、システム１００の第１のコンポーネントが、動作を完了するために、および結果を生成するために必要とされ得る。同様に、システム１００の第２のコンポーネントが、実質的に同様の動作を完了するために、および実質的に同様の結果を生成するために必要とされ得る。生成されたデータ構造の分析は、第２のコンポーネントが、第１のコンポーネントによって生成された結果とは大幅に異なる結果を生成したことを示し得る。同様に、データ構造は、許容可能結果パラメータの範囲外にあることが顕著である第２のコンポーネントの結果パラメータ値を示すかもしれない。これらの結果はおそらく、システム１００の第２のコンポーネントの潜在的な性能問題を示し得る。
【０１２０】
この明細書で説明される主題および機能的動作の実施形態は、デジタル電子回路で、有形に具体化されたコンピュータソフトウェアまたはファームウェアで、この明細書に開示された構造およびそれらの構造的同等物を含むコンピュータハードウェアで、もしくは、それらのうちの１つ以上の組合せで実現され得る。この明細書で説明される主題の実施形態は、１つ以上のコンピュータプログラムとして、すなわち、データ処理装置による実行のために、またはデータ処理装置の動作を制御するために、有形の非一時的プログラム担体上で符号化されたコンピュータプログラム命令の１つ以上のモジュールとして実現され得る。それに代えて、またはそれに加えて、プログラム命令は、データ処理装置による実行のために好適なレシーバ装置へ送信される情報を符号化するために生成される、人工的に生成された伝搬信号、たとえば、機械によって生成された電気信号、光学信号、または電磁信号上で符号化され得る。コンピュータ記憶媒体は、機械読取可能記憶デバイス、機械読取可能記憶基板、ランダムまたはシリアルアクセスメモリデバイス、もしくは、それらのうちの１つ以上の組合せであってもよい。
【０１２１】
この明細書で説明されるプロセスおよび論理フローは、１つ以上のプログラマブルコンピュータが、１つ以上のコンピュータプログラムを、入力データ上で動作して出力を生成することによって機能を行なうように実行することによって、行なわれ得る。プロセスおよび論理フローはまた、特殊用途論理回路、たとえばＦＰＧＡ（field programmable gate array：フィールドプログラマブルゲートアレイ）、ＡＳＩＣ（特定用途向け集積回路）、またはＧＰＧＰＵ（General purpose graphics processing unit：汎用グラフィック処理ユニット）によって行なわれてもよく、装置はまた、当該特殊用途論理回路として実現されてもよい。
【０１２２】
コンピュータプログラムの実行に好適なコンピュータは、汎用または専用マイクロプロセッサまたはそれら双方、もしくは任意の他の種類の中央処理装置を例として含み、それ

に基づき得る。一般に、中央処理装置は、読取専用メモリまたはランダムアクセスメモリまたはそれら双方から、命令およびデータを受信するであろう。コンピュータの本質的要素は、命令を行なうか実行するための中央処理装置と、命令およびデータを格納するための１つ以上のメモリデバイスとである。一般に、コンピュータはまた、データを格納するための１つ以上の大容量記憶デバイス、たとえば磁気ディスク、光磁気ディスク、または光学ディスクを含み、もしくは、当該大容量記憶デバイスからデータを受信し、または当該大容量記憶デバイスへデータを転送し、またはそれら双方を行なうために動作可能に結合されるであろう。しかしながら、コンピュータはそのようなデバイスを有していなくてもよい。

【０１２３】

コンピュータプログラム命令およびデータを格納するのに好適なコンピュータ読取可能媒体は、半導体メモリデバイス、たとえばＥＰＲＯＭ、ＥＥＰＲＯＭ、およびフラッシュメモリデバイス；磁気ディスク、たとえば内部ハードディスクまたはリムーバブルディスクを例として含む、あらゆる形態の不揮発性メモリ、媒体、およびメモリデバイスを含む。プロセッサおよびメモリは、特殊用途論理回路によって補足され、またはそれに組み込まれ得る。

【０１２４】

この明細書は多くの特定の実現詳細を含むものの、これらは、発明の範囲または特許請求の範囲に対する限定として解釈されるべきでなく、むしろ、特定の発明の特定の実施形態に特有であり得る特徴の説明として解釈されるべきである。この明細書において別々の実施形態の状況で説明されるある特徴を、単一の実施形態において組合せて実現することもできる。逆に、単一の実施形態の状況で説明されるさまざまな特徴を、複数の実施形態で別々に、または任意の好適な部分的組合せで実現することもできる。さらに、特徴はある組合せで作用するとして上述され、そういうものとして当初特許請求され得るが、場合によっては、特許請求された組合せからの１つ以上の特徴がその組合せから削除されてもよく、特許請求された組合せは、部分的組合せまたは部分的組合せの変形に向けられてもよい。

【０１２５】

同様に、動作は特定の順序で図面に示されているが、これは、望ましい結果を達成するために、そのような動作が図示された特定の順序または連続する順序で行なわれること、もしくは、図示された動作がすべて行なわれることを要求するものとして理解されるべきではない。ある状況では、マルチタスクおよび並行処理が有利であるかもしれない。さらに、上述の実施形態におけるさまざまなシステムモジュールおよびコンポーネントの分離は、すべての実施形態においてそのような分離を要求するものとして理解されるべきではなく、説明されたプログラムコンポーネントおよびシステムは一般に、単一のソフトウェア製品にともに一体化されるか、または複数のソフトウェア製品にパッケージ化され得るということが理解されるべきである。

【０１２６】

主題の特定の実施形態が説明されてきた。他の実施形態は、特許請求の範囲内にある。たとえば、請求項に記載されたアクションは、異なる順序で行なわれ、依然として望ましい結果を達成することができる。一例として、添付図面に示されたプロセスは、望ましい結果を達成するために、図示された特定の順序または連続する順序を必ずしも必要とはしない。ある実現化例では、マルチタスクおよび並行処理が有利であるかもしれない。

【図１】

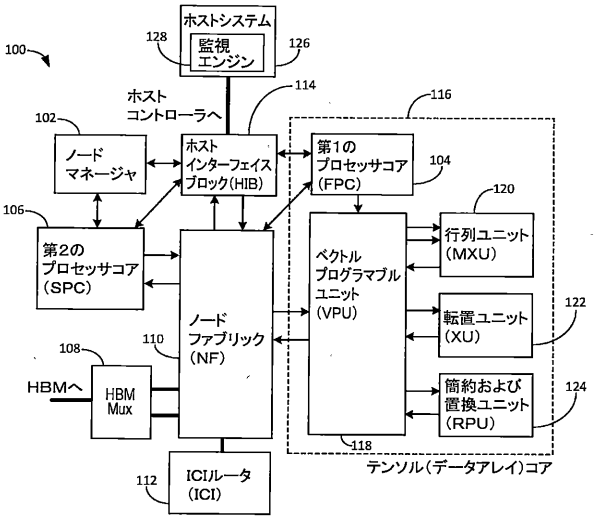


FIG. 1

【図２】

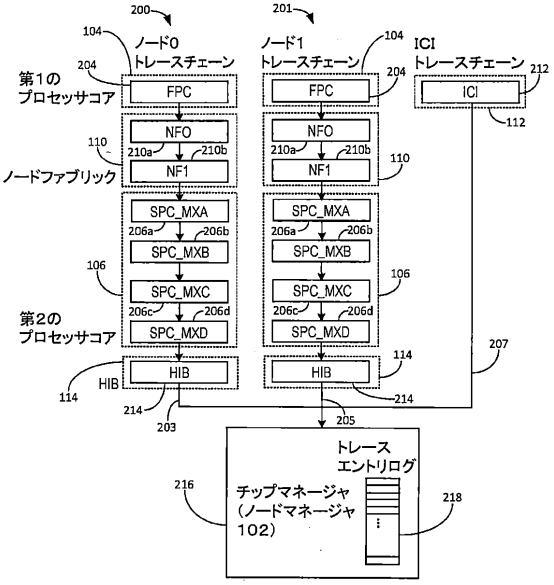
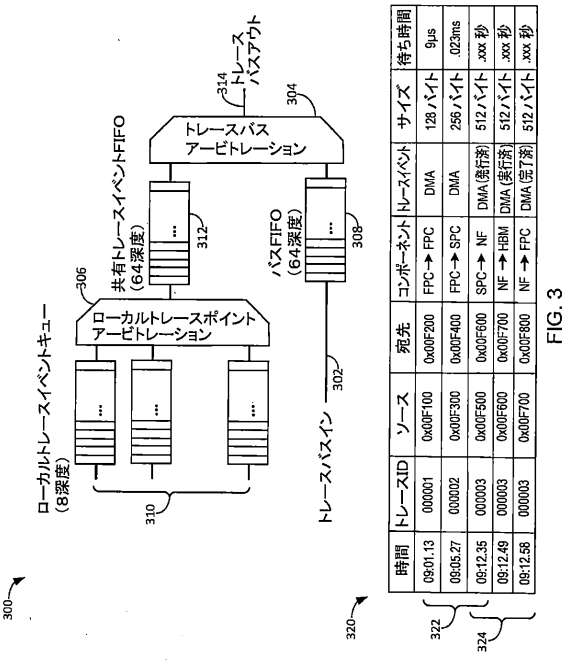


FIG. 2

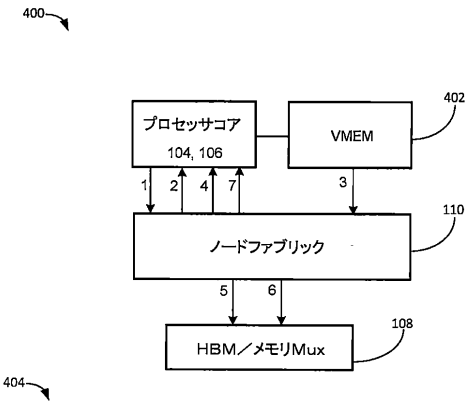
【図３】



時間	トレースID	ソース	宛先	コンポーネント	トレースイベント	サイズ	待ち時間
09:01.13	000001	0x00F100	0x00F200	FPC→FPC	DMA	128バイト	9μs
09:05.27	000002	0x00F300	0x00F400	FPC→SPC	DMA	256バイト	023ms
09:12.35	000003	0x00F500	0x00F600	SPC→NF	DMA(発行済)	512バイト	xxx秒
09:12.49	000003	0x00F600	0x00F700	NF→HBM	DMA(発行済)	512バイト	xxx秒
09:12.58	000003	0x00F700	0x00F800	NF→FPC	DMA(完了済)	512バイト	xxx秒

FIG. 3

【図４】



ステップ	動作
1	最初のDMA要求；ノードファブリックにおけるトレースポイント
2	読出コマンド：NFはコアにデータを転送するよう求める；NFにおけるトレースポイント
3	読出完了：ここではNFにトレースポイントはない！
4	読出リソース更新：コアにおける同期フラグ更新；FPCにおけるトレースポイント
5	書込コマンド：NFはHBMに通知する；NFにおけるトレースポイント
6	書込完了：NFにおけるトレースポイント
7	書込リソース更新：FPCにおける同期フラグ更新；FPCにおけるトレースポイント

FIG. 4

【図５】

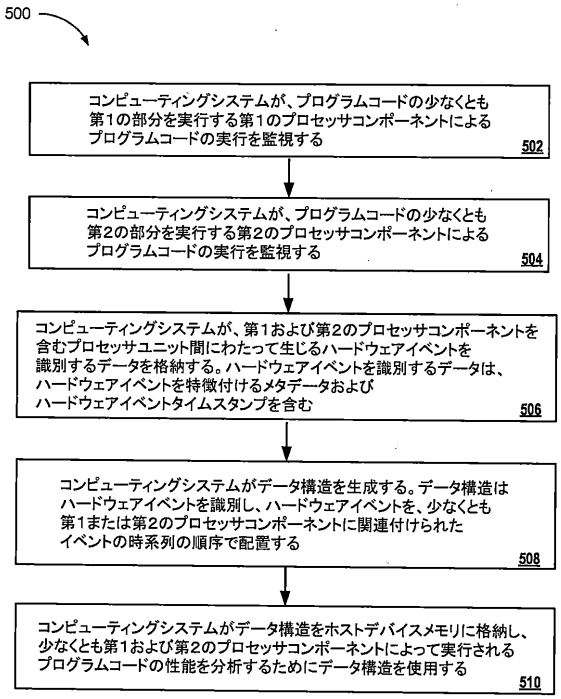


FIG. 5

【手続補正書】
【提出日】令和3年3月26日(2021.3.26)
【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】
トレーシングシステムを使用して行なわれる方法であって、前記方法は、
複数のノードを含むプロセッサにコードを提供するステップと、
前記プロセッサの前記複数のノードの各々を使用して前記コードを実行するステップと、
前記プロセッサに含まれる前記複数のノードのうちの特定のノードについて、トレースビットが第１の値に設定されていることを検出するステップとを含み、前記第１の値は、トレースデータが前記トレーシングシステムのカウンタを使用して前記プロセッサで生成されるようにし、前記方法はさらに、
前記トレースビットが前記第１の値に設定されていることを検出するステップに応答して、前記トレースデータを生成するステップを含み、前記トレースデータは前記カウンタに格納され、前記方法はさらに、
前記カウンタに格納された前記トレースデータを、ホストでの分析のために前記ホストに提供するステップを含む、方法。
【請求項２】
前記複数のノードのうちの少なくとも１つのノードは、前記プロセッサによって実行された前記コードによって発行された命令に基づいて特定の動作を実行するように動作可能

である固定機能コンポーネントノードであり、前記特定の動作は、前記トレースビットが前記第１の値に設定されていることを検出するステップに応答して、前記トレースデータを生成するために使用される、請求項１に記載の方法。
【請求項３】
前記固定機能コンポーネントノードはメモリアクセスエンジンであり、
前記トレースデータを生成するステップは、前記コードによって発行された前記命令に基づいて、前記メモリアクセスエンジンを使用して行なわれるべきメモリアクセス動作用の記述子を生成するステップを含み、前記記述子は、前記トレースデータが前記カウンタを使用して前記プロセッサで生成されるようにするための前記第１の値に設定される前記トレースビットを含む、請求項２に記載の方法。
【請求項４】
前記特定の動作は、前記メモリアクセスエンジンを使用して前記メモリアクセス動作を行なうことを含み、
前記メモリアクセス動作は、前記複数のノードのうちの前記特定のノードおよび少なくとも１つの他のノードを伴う複数の中間動作を含み、
前記メモリアクセス動作を行なうことは、中間動作に関連付けられたトレースデータが、前記特定のノードのメモリバッファに格納されるようにする、請求項３に記載の方法。
【請求項５】
前記トレースデータを生成するステップは、前記複数のノードのうちの２つ以上のノードを結合するデータ通信バスに対応するトレースチェーンに提供されるべきトレースエントリを表わすデータを生成するステップを含む、請求項１から４のいずれか１項に記載の方法。
【請求項６】
前記トレースチェーンを使用して、前記トレーシングシステムのチップマネージャに格納されるべき前記トレースエントリを表わす前記データを提供するステップと、
前記トレースエントリを表わす前記データを前記チップマネージャに格納するステップとをさらに含み、前記チップマネージャは、前記コードが実行されている場合に前記プロセッサで生じる複数のハードウェアイベントのために複数のトレースエントリを格納するように動作可能である、請求項５に記載の方法。
【請求項７】
前記プロセッサのノードに、ハードウェアイベントを記述するトレースデータを格納させるように、前記ホストが、前記プロセッサに提供された前記コードにオペランドを埋込むステップをさらに含み、前記ハードウェアイベントは、前記コードの実行中に前記プロセッサで生じる、請求項１から６のいずれか１項に記載の方法。
【請求項８】
前記コードを提供するステップは、前記ホストによって前記コードをコンパイルすることに応答して前記コードを提供するステップを含み、前記ホストは、前記トレーシングシステムのホストインターフェイスおよびノードファブリックによって前記プロセッサに結合され、前記ホストは、コンパイルされた前記コードを前記プロセッサに提供するために、前記ホストインターフェイスおよび前記ノードファブリックの各々を使用する、請求項７に記載の方法。
【請求項９】
前記コードをコンパイルすることは、前記トレーシングシステムの前記ホストインターフェイスおよび前記ノードファブリックと通信する前記ホストの監視エンジンを使用して、コンパイルされた前記コードの性能分析を可能にするために、前記プロセッサに提供された前記コードに前記オペランドを埋込むことを含む、請求項８に記載の方法。
【請求項１０】
トレーシングシステムであって、
１つ以上の処理デバイスと、
命令を格納する１つ以上の非一時的機械読取可能記憶デバイスとを含み、前記命令は、

動作の実行を引き起こすために前記１つ以上の処理デバイスによって実行可能であり、前記動作は、
複数のノードを含むプロセッサにコードを提供することと、
前記プロセッサの前記複数のノードの各々を使用して前記コードを実行することと、
前記プロセッサに含まれる前記複数のノードのうちの特定のノードについて、トレースビットが第１の値に設定されていることを検出することとを含み、前記第１の値は、トレースデータが前記トレーシングシステムのカウンタを使用して前記プロセッサで生成されるようにし、前記動作はさらに、
前記トレースビットが前記第１の値に設定されていることを検出することに応答して、前記トレースデータを生成することを含み、前記トレースデータは前記カウンタに格納され、前記動作はさらに、
前記カウンタに格納された前記トレースデータを、ホストでの分析のために前記ホストに提供することを含む、トレーシングシステム。
【請求項１１】
前記複数のノードのうちの少なくとも１つのノードは、前記プロセッサによって実行された前記コードによって発行された命令に基づいて特定の動作を実行するように動作可能である固定機能コンポーネントノードであり、前記特定の動作は、前記トレースビットが前記第１の値に設定されていることを検出することに応答して、前記トレースデータを生成するために使用される、請求項１０に記載のトレーシングシステム。
【請求項１２】
前記固定機能コンポーネントノードはメモリアクセスエンジンであり、
前記トレースデータを生成することは、前記コードによって発行された前記命令に基づいて、前記メモリアクセスエンジンを使用して行なわれるべきメモリアクセス動作用の記述子を生成することを含み、前記記述子は、前記トレースデータが前記カウンタを使用して前記プロセッサで生成されるようにするための前記第１の値に設定される前記トレースビットを含む、請求項１１に記載のトレーシングシステム。
【請求項１３】
前記特定の動作は、前記メモリアクセスエンジンを使用して前記メモリアクセス動作を行なうことを含み、
前記メモリアクセス動作は、前記複数のノードのうちの前記特定のノードおよび少なくとも１つの他のノードを伴う複数の中間動作を含み、
前記メモリアクセス動作を行なうことは、中間動作に関連付けられたトレースデータが、前記特定のノードのメモリバッファに格納されるようにする、請求項１２に記載のトレーシングシステム。
【請求項１４】
前記トレースデータを生成することは、前記複数のノードのうちの２つ以上のノードを結合するデータ通信バスに対応するトレースチェーンに提供されるべきトレースエントリを表わすデータを生成することを含む、請求項１０から１３のいずれか１項に記載のトレーシングシステム。
【請求項１５】
前記動作は、
前記トレースチェーンを使用して、前記トレーシングシステムのチップマネージャに格納されるべき前記トレースエントリを表わす前記データを提供することと、
前記トレースエントリを表わす前記データを前記チップマネージャに格納することとを含み、前記チップマネージャは、前記コードが実行されている場合に前記プロセッサで生じる複数のハードウェアイベントのために複数のトレースエントリを格納するように動作可能である、請求項１４に記載のトレーシングシステム。
【請求項１６】
前記動作は、前記プロセッサのノードに、ハードウェアイベントを記述するトレースデータを格納させるように、前記ホストが、前記プロセッサに提供された前記コードにオペ

ランドを埋込むことを含み、前記ハードウェアイベントは、前記コードの実行中に前記プロセッサで生じる、請求項１０から１５のいずれか１項に記載のトレーシングシステム。
【請求項１７】
前記コードを提供することは、前記ホストによって前記コードをコンパイルすることに応答して前記コードを提供することを含み、前記ホストは、前記トレーシングシステムのホストインターフェイスおよびノードファブリックによって前記プロセッサに結合され、前記ホストは、コンパイルされた前記コードを前記プロセッサに提供するために、前記ホストインターフェイスおよび前記ノードファブリックの各々を使用する、請求項１６に記載のトレーシングシステム。
【請求項１８】
前記コードをコンパイルすることは、前記トレーシングシステムの前記ホストインターフェイスおよび前記ノードファブリックと通信する前記ホストの監視エンジンを使用して、コンパイルされた前記コードの性能分析を可能にするために、前記プロセッサに提供された前記コードに前記オペランドを埋込むことを含む、請求項１７に記載のトレーシングシステム。
【請求項１９】
命令を有するコンピュータプログラムであって、前記命令は、動作の実行を引き起こすためにトレーシングシステムの１つ以上のプロセッサによって実行可能であり、前記動作は、
複数のノードを含むプロセッサにコードを提供することと、
前記プロセッサの前記複数のノードの各々を使用して前記コードを実行することと、
前記プロセッサに含まれる前記複数のノードのうちの特定のノードについて、トレースビットが第１の値に設定されていることを検出することとを含み、前記第１の値は、トレースデータが前記トレーシングシステムのカウンタを使用して前記プロセッサで生成されるようにし、前記動作はさらに、
前記トレースビットが前記第１の値に設定されていることを検出することに応答して、前記トレースデータを生成することを含み、前記トレースデータは前記カウンタに格納され、前記動作はさらに、
前記カウンタに格納された前記トレースデータを、ホストでの分析のために前記ホストに提供することを含む、コンピュータプログラム。
【請求項２０】
前記トレースデータを生成することは、
前記複数のノードのうちの２つ以上のノードを結合するデータ通信バスに対応するトレースチェーンに提供されるべきトレースエントリを表わすデータを生成することと、
前記トレーシングシステムのチップマネージャが、前記トレースエントリを表わす前記データを、前記チップマネージャでの格納のために受信することと、
受信することに応答して、前記トレースエントリを表わす前記データを前記トレーシングシステムの前記チップマネージャに格納することとを含み、前記チップマネージャは、前記コードが実行されている場合に前記プロセッサで生じる複数のハードウェアイベントのために複数のトレースエントリを格納するように動作可能である、請求項１９に記載のコンピュータプログラム。
【請求項２１】
請求項１から９のいずれか１項に記載の方法をコンピュータに実行させるためのプログラム。

フロントページの続き

(72)発明者　クマー，ナビーン
アメリカ合衆国、９４０４３　カルフォルニア州、マウンテン・ビュー、アンフィシアター・パークウェイ、１６００

Ｆターム(参考) 5B042 HH30　MA08　MC35　MC40

【外国語明細書】
2021108129000001.pdf