この記事は、インテル® デベロッパー・ゾーンに掲載されている「Optimization and Performance Tuning for Intel® Xeon Phi™ Coprocessors, Part 2: Understanding and Using Hardware Events」の日本語参考訳です。

インテル® Xeon Phi™ コプロセッサー向けの最適化とパフォーマンス・チューニング - パート 2: ハードウェア・イベントの理解と使用

概要

この記事は、インテル® Xeon Phi™ コプロセッサー向けにアプリケーションを最適化する開発者向けに記述されています。ここでは、インテル® VTune™ Amplifier XE パフォーマンス・プロファイラーを利用して最適化を行います。アーキテクチャーの概要、パフォーマンス解析に用いるイベントと評価基準の詳細を説明し、推奨するチューニングを紹介します。

1 はじめに

インテル® Xeon Phi™ コプロセッサーは、インテル® メニー・インテグレーテッド・コア (インテル® MIC) アーキテクチャー・ベースの最初の製品です。インテル® Xeon Phi™ コプロセッサー向けアプリケーションのパフォーマンス・チューニングには、インテル® VTune™ Amplifier XE パフォーマンス・プロファイラーを利用することができます。

ソフトウェア・パフォーマンスの最適化は、アプリケーションのチューニング、システムのチューニング、オペレーティング・システムのチューニング、その他のさまざまなレベルがあります。一般に、トップダウン方式のアプローチが最も効果的です。最初にシステムのチューニングを行った後、アプリケーションのアルゴリズムを最適化して、マイクロアーキテクチャー・レベルのチューニングを行います。システムのチューニング (オペレーティング・システムのチューニングを含む) では通常、ハードウェアのボトルネックを排除します。アルゴリズムのチューニングには、並列化、I/O のチューニング、効率良いデータ構造やライブラリー・ルーチンの選択などが挙げられます。一般的に、アルゴリズムのチューニングは、アプリケーションの hotspot やソースコードに関する知識に関連しており、アプリケーションのパフォーマンスを向上することを目指します。

この記事のパート 1 では、インテル® Xeon Phi™ コプロセッサー上で実行するアプリケーションで考慮すべきアルゴリズムの最適化について説明しました。パート 2 では、マイクロアーキテクチャーの最適化と、それらの最適化が必要な場所を特定する方法について説明します。マイクロアーキテクチャーのチューニングには、アプリケーションがハードウェアでどのように実行されているか (パイプライン、キャッシュ、その他がどのように利用されているか) についての知識が必要です。このレベルのチューニングは、アーキテクチャーとハードウェアに特化しています。このマイクロアーキテクチャーのチューニングを行うには、アプリケーション実行中にコンピューティング・ハードウェアで収集されたリアルタイム・パフォーマンス情報を取得する必要があります。

この情報は、特定のイベントの回数をカウントするようにプログラム可能な、プロセシング・コアのパフォーマンス・モニタリング・ユニット (PMU) に格納されます。インテル® VTune™ Amplifier XE 2013 を利用すると、インテル® Xeon Phi™ コプロセッサーからサンプリングされたデータを収集することができます。この記事は、コプロセッサー上で実行されたアプリケーションから収集されたデータを解析するフレームワークを提供します。

2 インテル® Xeon Phi™ コプロセッサーの概要

インテル® Xeon Phi™ コプロセッサーは、データアクセスに対して計算比率が高い、高度な並列アプリケーションに適しています。インテル® Xeon Phi™ コプロセッサーは、双方向リングバスに接続された最大 61 個のコアで構成されます。各コアはラウンドロビン方式で最大 4 つのハードウェア・スレッドを切り替えて実行することができるため、合計 244 のハードウェア・スレッドを利用できます。各コアは、インオーダーのデュアル発行 x86 パイプライン、ローカル L1 および L2 キャッシュ、個別のベクトル・プロセシング・ユニット (VPU) から構成されます。以下に、キャッシュ階層の詳細を示します。

種類	サイズ	情報
L1 命令	32KB	8 ウェイ、64B キャッシュライン
L1 データ	32KB	8 ウェイ、64B キャッシュライン
L2 命令 + データ	512KB	8 ウェイ、64B キャッシュライン

表 1: キャッシュの情報

インテル® Xeon Phi™ コプロセッサーは、特定のデータ・アクセス・パターンを対象とする、5 種類のハードウェア・プリフェッチ機能を実装しています。この広範なハードウェア・プリフェッチと、コンパイラーにより生成されるソフトウェア・プリフェッチにより、ソフトウェア開発者が明示的にプリフェッチ命令を利用する負担を軽減します。この記事では、ソフトウェア・プリフェッチの最適化については説明しません。

インテル® Xeon Phi™ コプロセッサーのもう 1 つの特長は、セカンドレベルのデータ・トランスレーション・ルックアサイド・バッファー (DTLB) です。TLB は仮想メモリーアドレスから物理メモリーアドレスへの変換を高速化するキャッシュです。TLB が実装されることで、メモリーアクセスごとにアドレス変換を行う必要はありません。各コアの L2 DTLB には、64 のエントリーがあり、2M ページの変換または 4K ページのページ・ディレクトリー・エントリー (PDE) をキャッシュするのに使用されます。L2 DTLB により、L1 DTLB ミスで発生するレイテンシーを大幅に軽減できます。

種類	エントリー	ページサイズ	マップ
L1 命令	32	4KB	128KB
L1 データ	64	4KB	256KB
	8	2MB	16MB
L2 データ	64	4KB または 2MB	128M

表 1: TLB の情報

インテル® Xeon Phi™ コプロセッサーには、8 つのデュアルチャネル GDDR5 メモリー・コントローラーも備わっています。各チャネルのデータ転送レートは 5.5 GT/s で、理論的な合計メモリー帯域幅は 352 GB/s になります。

3 General Exploration (全般解析) 手法

インテル® VTune™ Amplifier XE を利用してインテル® Xeon Phi™ コプロセッサーからデータを収集して確認する手順は、「関連情報」にリストされたドキュメントで詳細に説明されています。ここでは、インテル® VTune™ Amplifier XE ですでに収集されているデータを解析する手順を説明します。データは複数回実行して収集する必要があります。また、評価基準はインテル® VTune™ Amplifier XE が提供するデータを基に手動で計算します。インテル® Xeon Phi™ 製品ファミリーに対するインテル® VTune™ Amplifier XE のサポートは今後も拡張される予定です。

さまざまなイベントのカウントを個々に確認することも重要ですが、この記事では、ほとんどのイベントは評価基準を求めるために使用されます。セクション 4 では一般的な効率性の評価基準をリストします。これは、コードの特定の領域の最適化をいつ開始して、いつ終了するかを評価する上で役立つでしょう。セクション 5 では、アプリケーション解析に重要な評価基準を詳しく説明します。各評価基準とその説明に加えて、利用可能なイベントから評価基準を計算する式、パフォーマンス問題を調査すべき評価基準のしきい値、推奨するチューニングについても説明します。

インテル® VTune™ Amplifier XE によるパフォーマンス解析は、一般に以下の手順で行います。

1. hotspot (アプリケーションの合計 CPU サイクルの大半を占める関数) を特定します。
2. セクション 4 の評価基準を用いてその hotspot の効率性を評価します。
3. 効率が悪い場合は、セクション 5 で説明する適用可能な評価基準をチェックします。評価基準の値がしきい値未満の場合や、ほかの基準との相互作用により適用できない場合、この記事の追加情報を参照して問題を特定および修正します。
4. 上位の hotspot をすべて評価するまで上記のステップを繰り返します。

この手順に従う場合、典型的なワークロードを慎重に選択することが重要です。評価基準の多くは、複数のイベントを収集します。このため、ワークロードを複数回実行してデータを収集する必要があります。理想的なワークロードは、データ収集間隔よりも長い間動作が一定であり、定常処理状態を含むものです。ワークロードは、一貫性があり、繰り返し同じ結果が得られるもので、データの収集中に多くの CPU 時間を費やすアプリケーションであるべきです。データを収集するためワークロードを複数回実行する場合は、パフォーマンスに影響するウォームキャッシュ効果やその他の要因がないことを確認してください。最後に、解析を開始する前に、基本的な CPU クロックサイクルおよび命令イベントに問題がないこと (繰り返し実行してもイベントカウントが一定で期待値内に収まっていること) を確認します。この記事のイベントの評価基準と説明は、B ステッピングのコプロセッサーに基づいています。

4 効率性の評価基準

さまざまな評価基準を用いて、インテル® Xeon Phi™ コプロセッサーの効率を測定できます。アプリケーションがリソースを適切に利用しているかどうかを確認するため、最初にこれらの評価基準を調べることを推奨します。これらの評価基準は、チューニング・サイクルの一部としてさまざまな最適化の影響を評価するのに役立ちます (注記されている場合を除く)。

各評価基準の式は、(実行しているすべてのハードウェア・スレッドからのサンプルの合計により) 関数レベルで計算します。インテル® VTune™ Amplifier XE の「カスタム解析」ハードウェア・イベントベース・サンプリング解析タイプ、および [関数/コールスタック] グループの [PMU イベント] タブを選択した場合、この集計を自動的に行います。これにより (関数ごと) から集計された値は、この記事の評価基準の計算に利用できます。

4.1 CPI

使用されるイベント

イベント	意味
CPU_CLK_UNHALTED	コアで実行されたサイクル数
INSTRUCTIONS_EXECUTED	スレッドで実行された命令数

式

評価基準	式
スレッドあたりの平均 CPI	CPU_CLK_UNHALTED / INSTRUCTIONS_EXECUTED
コアあたりの平均 CPI	(スレッドあたりの CPI) / 使用されたハードウェア・スレッドの数

しきい値

評価基準	以下の場合は調査:
スレッドあたりの平均 CPI	> 4.0、または全般的に増加
コアあたりの平均 CPI	> 1.0、または全般的に増加

説明と使用方法

命令あたりのサイクル数 (CPI) は、長年インテル® VTune™ Amplifier XE で使用されてきた評価基準です。CPI は命令をリタイアするのに必要な CPU サイクルの平均数であり、実行中のアプリケーションに影響したシステムのレイテンシーの指標となります。CPI は比率であるため、アプリケーションで実行された CPU サイクル数 (分子) の変化、または実行された命令数 (分母) の変化によって変化します。そのため、CPI は比率の一部分が変わる場合の比較に最適です。例えば、(異なる) hotspot で CPI が低いコード領域のデータ構造を変更する場合、その領域内のコードが変更されていなければ、その hotspot の「新しい」CPI と「以前の」CPI を比較できます。目標は、両方の hotspot およびアプリケーション全体の CPI を低くすることです。

さまざまな評価基準を利用するには、複数のハードウェア・スレッドを実行した場合の CPI の解釈方法を理解することが重要です。インテル® MIC アーキテクチャーでは、「コアあたり」または「ス

レッドあたり」の 2 つの方法で CPI を算出できます。どちらの CPI 算出方法にもメリットがあります。スレッドごとにハードウェア・スレッドの CPI を計算する解析は最も簡単です。この CPI は、CPU_CLK_UNHALTED (クロックまたはサイクルとも呼ばれます) と INSTRUCTIONS_EXECUTED の 2 つのイベントから計算されます。CPU_CLK_UNHALTED は、CPU コアのクロックをカウントします。クロックはコアに実装されているため、1 つのコアのハードウェア・スレッドはすべて同じクロックになります。このイベントはコアレベルでカウントされます。特定のサンプルにおいて、同じコアで実行されるスレッドはすべて同じ値になります。

INSTRUCTIONS_EXECUTED イベントはスレッドレベルでカウントされます。各スレッドでリタイアした命令数に依存し、コアで実行する各スレッドでこのイベントの値は異なります。スレッドあたりの CPI を計算するのは簡単です。CPU_CLK_UNHALTED を、INSTRUCTIONS_EXECUTED で割るだけです。特定のサンプルにおいて、この計算は、クロックについてはコアの値を使用し、実行された命令については個々のハードウェア・スレッドの値を使います。この計算は、すべてのサンプルを各関数に適用し、関数レベルで行われるため、関数を実行するすべてのハードウェア・スレッドあたりの平均 CPI を計算することになります。

コアあたりの CPI はやや複雑です。コアで実行されるハードウェア・スレッドはすべて、クロック数については共通の値を共有し、実行された命令数については個々の値を持ちます。「合計」CPI またはコアあたりの平均 CPI を計算するには、コアの CPU_CLK_UNHALTED 値を、すべてのスレッドの INSTRUCTIONS_EXECUTED 値の合計で割ります。例えば、インテル® Xeon Phi™ コプロセッサーで 1 つのコアあたり 2 つのハードウェア・スレッドを実行しているアプリケーションを仮定します。アプリケーションのあるホットな関数が 1,200 クロックで完了し、その 1,200 サイクルの間に各スレッドが 600 の命令を実行したとします。この関数のスレッドあたりの CPI は、(1200 / 600) で 2.0 になります。この関数のコアあたりの CPI は、(1200 / (600 + 600)) で 1.0 になります。次に、同じアプリケーションが 1 つのコアあたり 3 つのハードウェア・スレッドを実行し、同じ時間に各スレッドが 400 の命令、つまり合計で 1,200 の命令をリタイアしたと仮定します。この場合、この関数のスレッドあたりの CPI は、(1200 / 400) で 3.0 になります。ただし、コアあたりの CPI は、(1200 / (400 + 400 + 400)) で 1.0 のままです。コアあたりの平均 CPI の計算は、実行しているすべてのコアとスレッドのサンプルを使用して行われます。このため、コアあたりの平均 CPI を求める最も簡単な方法は、スレッドごとの平均 CPI を、使用されたコアあたりのハードウェア・スレッドの数で割ることです。

上記の例は、スレッドあたりの CPI とコアあたりの CPI が、アプリケーションの変更により異なる影響を受けることを示しています。このケースでは、ハードウェア・スレッドを追加した後、スレッドあたりの CPI は低下し、コアあたりの CPI は同じままでした。実際は、3 つ目のハードウェア・スレッドが追加されたことにより、アプリケーションは同じ時間に同じワーク量を完了することができました。データを調べると、状況をより深く理解することができます。コアあたりの CPI が同じということは、コアが以前と同じ割合で命令を実行したことを示しています。スレッドあたりの CPI が 2.0 から 3.0 に低下したということは、ハードウェア・スレッドが以前よりも効率が高くなかったことを示しています。この 2 つの解析はどちらも正しいものです。コアのパフォーマンスは同じで、2 つのハードウェア・スレッドを実行した場合は高効率となり、3 つのハードウェア・スレッドを実行した場合は低効率になります。しかし、開発者がスレッドあたりの CPI のみを調べた場合、パフォーマンスが低下していると判断してしまいます。インテル® Xeon Phi™ コプロセッサーの一般的な

使用状況では、スレッドあたりの CPI とコアあたりの CPI に異なる影響を及ぼす変更が加えられる可能性があるため、両方を測定して評価することが重要です。

インテル® Xeon Phi™ コプロセッサーは、各物理コアで 4 つのハードウェア・スレッドをサポートします。しかし、パイプラインのフロントエンドは、サイクルあたり 2 命令しか発行できません (逆に、インテル® Xeon® プロセッサーのパイプラインは、現在 2 つのハードウェア・スレッドをサポートし、サイクルあたり 4 つの命令を発行できるフロントエンドが特長です)。インテル® Xeon Phi™ コプロセッサーの 4 つのハードウェア・スレッドは、ワークロードのデータアクセスのレイテンシーを隠蔽するのに役立ちます。インテル® Xeon Phi™ コプロセッサーのパイプラインはインオーダーで命令を処理する (ある命令を実行する前に、前の命令が完了するのを待つ) ため 、アプリケーションの種類によっては、ハードウェア・スレッドの追加が重要になります。1 つのハードウェア・スレッドがデータを待つ間、別のハードウェア・スレッドを実行できるためです。

インテル® Xeon Phi™ コプロセッサーのパイプラインのフロントエンドについて理解しておくべきもう 1 つの重要な点は、ハードウェア・コンテキストを 1 つしか実行していない場合でも、連続するクロックサイクルで同じハードウェア・コンテキスト (ハードウェア・スレッド) に命令を発行しないことです。このため、命令の発行効率を最大にするには、少なくとも 2 つのハードウェア・コンテキストを実行している必要があります。複数のコンテキストを実行している場合、フロントエンドはラウンドロビン方式でコンテキストを切り替えます。これらの要件とクロックあたり 2 命令を発行できることから、インテル® Xeon Phi™ コプロセッサーで実行するアプリケーションの理論上の最小 (最高) CPI を計算することができます。計算結果を表 3 に示します。

ハードウェア・スレッドの数 / コア	**理論上の最小 (最高) CPI / コア**	**理論上の最小 (最高) CPI / スレッド**
1	1.0	1.0
2	.5	1.0
3	.5	1.5
4	.5	2.0

表 3: 理論上の最小 CPI

データアクセスのレイテンシーが高いアプリケーションでは、4 つのハードウェア・スレッドをすべて利用することで、それぞれパフォーマンスが向上します。この場合、スレッドを追加するとコアあたりの CPI は低下します。これらの変更は実行される命令数に影響するため、処理するワークの量が増減したときに CPI を確認する際は注意が必要です。一般的に完了したワークの量が増加して各ハードウェア・スレッドが効率良く利用された場合、コアあたりの CPI は処理されたワークの増加よりも低い割合で増えます。コアあたりの CPI は、ハードウェア・スレッドを追加する利点を判断する上で役立ちます。コアあたりの CPI が減少した (良くなった) 場合でも、スレッドあたりの CPI が増加することがあります。開発者が行うコード最適化の多くは、スレッドレベルの CPI に注目するため、これを知っておくと役立つでしょう。

表 4 は、インテルの実験室で実際のワークロードを実行し、ハードウェア・スレッド/コアの数を 1 から 4 に変えて測定した、コアあたりの CPI とスレッドあたりの CPI を示しています。このアプリケーションでは、4 つ目のスレッドを追加したときは 2 つ目や 3 つ目のスレッドを追加したときほ

どパフォーマンスは向上していませんが、スレッドの増加とともにアプリケーションのパフォーマンスが向上しています。このデータは、スレッドの増加とともにコアあたりの CPI が増えている (効率が低下している) ことを示していますが、パフォーマンスが向上しているため、コアあたりの CPI は予想通り全体的に減少しています。このワークロードでは、実行される命令数はすべてのハードウェア・スレッド構成でほぼ一定であるため、CPI は直接実行時間に影響します。コアあたりの CPI が減少した場合、アプリケーションの合計実行時間も減少すると解釈できます。

評価基準	1 ハードウェア・スレッド / コア	2 ハードウェア・スレッド / コア	3 ハードウェア・スレッド / コア	4 ハードウェア・スレッド / コア
スレッドあたりの CPI	5.24	8.80	11.18	13.74
コアあたりの CPI	5.24	4.40	3.73	3.43

表 4: CPI の例

コアあたりの CPI とスレッドあたりの CPI のしきい値は保守的であることに注意してください。多くのアプリケーションは、CPI 値がしきい値より高い場合でも適切に動作します。一般に、コア内で処理を行っている (つまり、キャッシュ可能なワーキングセットを計算している) アプリケーションの CPI は、しきい値以下になります。少なくとも一部を複数のコアで処理する、またはメモリーを利用するアプリケーションの CPI は、しきい値よりも高くなります。

推奨するチューニング

一般に、アプリケーションに変更を加えると、実行される命令数または完了までにかかった時間のいずれかが変わるため、CPI に影響します。特に変更前のアプリケーションと比較する場合、コアあたりの CPI を減らす (つまり実行時間を短縮する) ことが一般的な目標です。セクション 5 で推奨されている手法を利用して、CPI を減らすことができます。「ベクトル化強度」 (セクション 5.3) を高めるような一部の最適化では、1 つの命令で実行されるワークの量が増加し、全体的に命令数が減少するため、CPI が増加する傾向があることに注意してください。CPI は、唯一のパフォーマンスの決定要因としてではなく、一般的な比較や効率の評価基準として役立ちます。

4.2 データアクセスに対する計算の比率

使用されるイベント

イベント	意味
VPU_ELEMENTS_ACTIVE	スレッドで実行された VPU 命令数
DATA_READ_OR_WRITE	スレッドの L1 データキャッシュへのロードとストアの数
DATA_READ_MISS_OR_WRITE_MISS	スレッドの L1 キャッシュをミスした要求されたロードまたはストアの数
L1_DATA_HIT_INFLIGHT_PF1	L2 から L1 にすでにプリフェッチされたキャッシュラインへの要求されたロードまたはストアの数

式

評価基準	式
データアクセスに対する L1 計算の比率	VPU_ELEMENTS_ACTIVE / DATA_READ_OR_WRITE
データアクセスに対する L2 計算の比率	VPU_ELEMENTS_ACTIVE / DATA_READ_MISS_OR_WRITE_MISS

しきい値

評価基準	以下の場合は調査:
データアクセスに対する L1 計算の比率	< ベクトル強度 (セクション 5.3 を参照)
データアクセスに対する L2 計算の比率	< 100x データアクセスに対する L1 計算の比率

説明と使用方法

これらの評価基準は、アプリケーションの計算密度、またはロードされたデータが処理される計算の平均数を測定する方法です。1 つ目の「データアクセスに対する L1 計算の比率」は、インテル® MIC アーキテクチャーで実行されるアプリケーションの適応性を判断するのに用います。インテル® MIC アーキテクチャーで適切に実行されるアプリケーションは、ベクトル化され、理想的にはデータの同じ部分 (または同じキャッシュライン) を複数の演算で実行するべきです。L1 比率は、各 L1 キャッシュアクセスで発生するベクトル演算の平均数を計算します。データ演算を含む、ベクトル演算はすべて、VPU_ELEMENTS_ACTIVE イベントの定義による分子に含まれます。VPU_ELEMENTS_ACTIVE は行われた演算数をより正確に示すため、VPU_INSTRUCTIONS_EXECUTED の代わりに使用されていました。例えば、16 個の float 型データをパックしたレジスターを処理する命令は、16 演算としてカウントされます。要求されるロードとストアはすべて分母に含まれます。プリフェッチは含まれません。

L1 評価基準のしきい値はガイドラインであり、インテル® MIC アーキテクチャーで適切に動作するほとんどのコードは、L1 アクセスに対する計算の比率が「ベクトル化強度」(セクション 5.3 を参照) 以上になるべきです。この比率は、各計算をベクトル化して一度に複数の要素を演算しないスカラー操作では、1:1 (1 つの計算で 1 つのデータアクセス) に近くなります。このしきい値を超える比率を達成できないアプリケーションは、インテル® MIC アーキテクチャーを十分に活用する計算集約型アプリケーションとは言えません。

L1 レベルの計算密度は重要です。L2 レベルの計算密度は、コードが効率良く動作しているかどうかの指標であり、しきい値はガイドラインです。最適なパフォーマンスを得るには、L1 に格納されているデータにアクセスします。これは、データをメモリーからアクセスできないことを意味するわけではありません。インテル® Xeon Phi™ コプロセッサーの高い帯域幅は、メモリーアクセスに極めて有利です。しかし、理想的には、データはプリフェッチによりメモリーからキャッシュに渡され、ロードが要求されたときに L1 に格納されたデータを利用可能であるべきです。これは、インテル® Xeon Phi™ コプロセッサーでは従来のプロセッサーよりもさらに重要です。長いデータ・レイテンシーは、インテル® MIC アーキテクチャーの特長の 1 つである、ベクトル化によるパフォーマンスの利点を損ないます。「データアクセスに対する L2 計算の比率」は、各 L2 アクセスで発

生するベクトル演算の平均数を示します。一般に、L1 キャッシュにデータをブロック化できるアプリケーション、またはデータアクセスを減らせるアプリケーションは、この比率が高くなります。ベースラインとして使われている 100x L1 比率のしきい値は、100 回の L1 データアクセスごとにほぼ 1 回の L2 データアクセスが行われることを意味します。L1 評価基準と同様に、(データの移動を含む) ベクトル演算はすべて分子に含まれます。

L1 評価基準の分母は、要求されたロードとストアをすべて (つまり、L1 データキャッシュへのアクセスをすべて) 含みます。L2 評価基準の分母はやや複雑で、L1 をミスした (L2 から要求される) データアクセスをすべて適用します。セクション 5.2 で説明する「L1 ヒットレート」と密接に関連しています。

推奨するチューニング

L1 計算密度の評価基準については、値が「ベクトル化強度」未満の場合、データアクセスを減らす一般的なチューニングを行います。一般に、クリティカル・パスの命令数を減らすチューニングがよく行われます。条件、初期化、その他の内部ループで必要ないものをすべて削除し、データ構造を合理化します。データをアライメントして、コンパイラーがロードとストア命令を生成する時にアライメントを仮定していることを確認します。また、コンパイラーが適切にベクトル化されたコードを生成している (例えば、レジスターをスピルしていない) ことを確認します。さらに、タスクやスレッド管理のオーバーヘッドを最小限に抑えます。

L2 計算密度の評価基準については、セクション 5.3 で説明する手法を適用して、L1 キャッシュのデータの局所性を向上させます。また、インテル® Cilk™ Plus やプラグマを利用してコードを変更し、コンパイラーがより効率良くベクトル化されたコードを生成すると、L1 と L2 の評価基準の向上につながります。

5 潜在的なパフォーマンス問題

このセクションでは、イベントから検出できる潜在的なパフォーマンス問題について説明します。各問題について、使用するイベントと意味を説明します。各問題は、評価基準としきい値で判別します。セクション 4 で説明した評価基準と同様に、評価基準の式は、(実行されるすべてのハードウェア・スレッドからのサンプルの合計を使って) 関数レベルで計算します。インテル® VTune™ Amplifier XE の「カスタム解析」ハードウェア・イベントベース・サンプリング解析タイプ、および [関数/コールスタック] グループの [PMU イベント] タブを選択した場合、この集計を自動的に行います。これにより (関数ごと) から集計された値は、この記事の評価基準の計算に利用できます。

各評価基準について計算した値をしきい値と比較します。この記事で示されているしきい値は、控えめに設定されています。つまり、しきい値を超えずに問題が発生するケースよりも、しきい値を超えても問題が発生しないケースのほうが多いことを意味します。しきい値は、開発者がさらに調査すべきかどうかを提示しているにすぎません。セクション 5 の評価基準もすべて、実行環境が固定された後に利用するように設定されています。使用されるハードウェア・スレッドまたはコアの数を変更すると、評価基準の予測に影響することがあります。

5.1 全般的なキャッシュ使用率

使用されるイベント

イベント	意味
CPU_CLK_UNHALTED	コアが実行していたサイクル数
DATA_READ_MISS_OR_WRITE_MISS	L1 データキャッシュをミスした要求されたロードまたはストアの数
L1_DATA_HIT_INFLIGHT_PF1	L2 から L1 にすでにプリフェッチされたキャッシュラインへの要求されたロードまたはストアの数
DATA_READ_OR_WRITE	スレッドの L1 データキャッシュへのロードとストアの数
EXEC_STAGE_CYCLES	スレッドが演算を実行していたときのサイクル数
L2_DATA_READ_MISS_CACHE_FILL / L2_DATA_WRITE_MISS_CACHE_FILL	L2 読み込みまたは (同じカードの) 別のコアの L2 キャッシュで処理された所有権ミスの読み込みをカウントします。ローカル L2 キャッシュをミスした L2 プリフェッチを含むため、要求されたキャッシュフィルの決定には役立ちません。
L2_DATA_READ_MISS_MEM_FILL / L2_DATA_WRITE_MISS_MEM_FILL	L2 読み込みまたは (同じカードの) メモリーで処理された所有権ミスの読み込みをカウントします。ローカル L2 キャッシュをミスした L2 プリフェッチを含むため、要求されたメモリーフィルの決定には役立ちません。

式

評価基準	式
L1 ミス	DATA_READ_MISS_OR_WRITE_MISS + L1_DATA_HIT_INFLIGHT_PF1
L1 ヒットレート	(DATA_READ_OR_WRITE – L1 ミス) / DATA_READ_OR_WRITE
レイテンシーの影響の推定値	(CPU_CLK_UNHALTED – EXEC_STAGE_CYCLES – DATA_READ_OR_WRITE) / DATA_READ_OR_WRITE_MISS

しきい値

評価基準	以下の場合は調査:
L1 ヒットレート	< 95%
レイテンシーの影響の推定値	> 145

説明と使用方法

インテル® Xeon Phi™ コプロセッサーで実行するアプリケーションのパフォーマンスを引き出すには、データの局所性が重要になります。アプリケーションのベクトル化のメリットを最大限に活用するには、できるだけ低いレイテンシーで VPU レジスターにパックしてアクセスできるようにしなければいけません。そうしないと、レジスターをパックする時間が、計算時間のほとんどを占めてしまいます。4 つのハードウェア・スレッド間で実行を切り替えることでデータアクセスのレイテンシーは多少隠蔽されますが、パフォーマンスに大きな影響を与える要因であることには変わりません。このため、データの局所性を向上することは、インテル® Xeon Phi™ コプロセッサーにとって最も価値のある最適化の 1 つです。L1 と L2 の局所性はどちらも重要です。リモートキャッシュやメモリーではなく、ローカル L2 キャッシュのデータにアクセスするようにプログラムを変更すると、少なくとも 250 サイクルのアクセス時間が節約できます。ロードでは、さらに多くの時間が節約されます。L2 ではなく L1 からデータにアクセスすると、約 20 サイクル節約できます。

「ヒットレート」評価基準は、各レベルのキャッシュの使用状況を示します。この評価基準は、各レベルのキャッシュのヒット数をアクセスの総数で割って計算します。ヒットレートは「要求された」アクセス (つまり、ソフトウェア・プリフェッチやハードウェア・プリフェッチではなくアプリケーションからの実際のロード) にのみ適用されます。データ (または L1) キャッシュの要求されたヒットレートを求められますが、式には多少の説明が必要でしょう。データキャッシュのアクセスは、標準ヒット、ミス、プリフェッチへのヒットのいずれかです (別々にカウントされます)。プリフェッチへのヒットは、データがキャッシュで見つからず、プリフェッチにより同じキャッシュレベルですでに検索されたキャッシュラインで見つかった場合に発生します。この種のヒットのレイテンシーは標準ヒットより長くなりますが、ミスより短くなります。ヒットレートが控えめになるように、この記事ではミスのように扱い、分子で引いています。

インテル® Xeon Phi™ コプロセッサーの L2 イベントと FILL イベントは、要求されたロードとストアの両方、そしてさまざまな種類のプリフェッチをカウントしています。すべてのプリフェッチがほかのイベントで正確にカウントされるとは限らないため、実際の要求された L2 ヒットまたはミスを計算するように式を調整することはできません。この記事では、(プリフェッチを含んでもかまわない) メモリー帯域幅を除いて、L2 イベントや FILL イベントに依存する評価基準を使用することは推奨しません。「レイテンシーの影響の推定値」評価基準は、L2 評価基準がない場合の代替手段として利用します。この評価基準は、各 L1 キャッシュミスのクロックサイクルの大まかな値を示します。分子は合計 CPU サイクルを使い、各 L1 キャッシュヒットで 1 (各 L1 アクセスは 1 サイクルかかるため)、EXEC_STAGE_CYCLES イベントがアクティブな各サイクルで 1 を引いて計算します。EXEC_STAGE_CYCLES は多くの計算で利用でき、計算サイクルを部分的にフィルターするのに用いられます。残りは、L1 キャッシュ外へのデータアクセスのサイクルであると考えられます。分母は L1 キャッシュミス (各 L1 キャッシュミスに費やされた CPU サイクル数の推定値) です。この値は近似値であり、パイプライン効果、不明なサイクル、メモリーアクセスのオーバーラップなどの多くの要因により、完全ではないことに注意してください。

「レイテンシーの影響の推定値」評価基準は、L1 データミスが L2 でヒットしているかどうかの指標にできます。L2 データアクセスのレイテンシーが 21 サイクルである場合、この数に「レイテンシーの影響の推定値」が近いほど L2 ヒットの割合が高くなります。しきい値は、アンロードされた L2 およびメモリーアクセス時間の平均である 145 に設定されています。「レイテンシーの影響の推定値」に関して注意すべきもう 1 つの重要な点は、すべての比率と同様に、分子または分母の変化による影響を受けることです。ほとんどの場合、データアクセスに影響する最適化を行うと、この評価基準の値は減少します。ただし、L1 ミスの減少のような変化では、分子と分母の両方が減

るため、この評価基準の値は変わりません。この種の変化は「L1 ヒットレート」評価基準に影響します。

評価基準では使用されていませんが、L2_DATA_READ_MISS_CACHE_FILL、L2_DATA_WRITE_MISS_CACHE_FILL、L2_DATA_READ_MISS_MEM_FILL、および L2_DATA_WRITE_MISS_MEM_FILL などのイベントもデータ局所性のチューニングに役立ちます。しかし、これらのイベントはプリフェッチを含むため、L2 関連の評価基準の計算には使用できません。これらのイベントの量は正確であると考えるべきではありませんが、MEM_FILL に対する CACHE_FILL の一般的な比率は、多くのデータがほかのコアのキャッシュへアクセスされたことを示します。リモート・キャッシュ・アクセスのレイテンシーはメモリーアクセスと同様に高いため、可能であれば回避すべきです。

推奨するチューニング

データの局所性を向上させる従来の手法の多くは、インテル® Xeon Phi™ コプロセッサーにも適用できます。キャッシュ・ブロッキング、ソフトウェア・プリフェッチ、データ・アライメント、ストリーミング・ストアはすべて、データをキャッシュに保持するのに有効です。隣接するキャッシュに存在するデータに関する問題には、キャッシュを意識したデータ圧縮やプライベート変数を利用します。キャッシュのセット連想性に関連する問題は、検出が難しい別の種類のデータ局所性の問題であり、上記のいくつかの手法によりデータの局所性を向上させようとしたにもかかわらずヒットレートが低い場合、多くのキャッシュラインが同じセットにマップされたことによる競合ミスが発生していると考えられます。インテル® Xeon Phi™ コプロセッサー上で、アプリケーションが L1 のデータを 4KB ストライドで (または L2 のデータを 64KB ストライドで) アクセスすると、セット連想性の問題 (競合ミス) が発生します。この問題により引き起こされるミスは、イベントにより検出される一般的なミスと区別できません。セット連想性問題が疑われる場合は、(アライメントするように) データ構造をパディングするか、アクセスストライド幅を変更してください。

5.2 TLB ミス

使用されるイベント

イベント	意味
DATA_PAGE_WALK	L1 TLB ミスの数
LONG_DATA_PAGE_WALK	L2 TLB ミスの数
DATA_READ_OR_WRITE	読み込みまたは書き込み操作の数

式

評価基準	式
L1 TLB ミスの比率	DATA_PAGE_WALK / DATA_READ_OR_WRITE
L2 TLB ミスの比率	LONG_DATA_PAGE_WALK / DATA_READ_OR_WRITE
L2 TLB ミスあたりの L1 TLB ミスの数	DATA_PAGE_WALK / LONG_DATA_PAGE_WALK

しきい値

評価基準	以下の場合は調査:
L1 TLB ミスの比率	> 1%
L2 TLB ミスの比率	> .1%
L2 TLB ミスあたりの L1 TLB ミスの数	約 1

説明と使用方法

インテル® Xeon Phi™ コプロセッサーには 2 レベルの TLB と 2 つのサイズのページ (4K および 2MB) があります。現在のバージョンの OS はデフォルトで 4K ページを使用します。この場合、L2 TLB はページ・テーブル・キャッシュとして動作し、(L2 TLB ヒットの) L1 TLB ミスのペナルティーは約 25 クロックサイクルに減ります。ラージ (2MB) ページでは、L2 TLB は標準 TLB として動作し、(L2 TLB ヒットの) L1 TLB ミスのペナルティーは約 8 クロックサイクルに減ります。

L2 TLB ミスのペナルティーは少なくとも 100 クロックであり、プリフェッチのレイテンシーを隠蔽できないため、L2 TLB ミスを回避することは重要です。L2 TLB でヒットする L1 TLB ミスはそれほど重要ではありません。

4KB ページには 64 のキャッシュラインがあるため、ページのすべてのキャッシュラインに対するシーケンシャル・アクセスの L1 TLB ミスの比率は 1/64 です。このため、L1 TLB ミスの比率が大きいことは空間の局所性が不足している (プログラムがページのすべてのデータを使っていない) ことを示します。また、スラッシングが発生している (同じループで複数のページがアクセスされるときに、TLB の連想性または容量がすべての TLB エントリーを保持するのに十分ではない) ことも示しています。同様のことはラージページと L2 TLB にも当てはまります。

L2 TLB ミスの比率に対して L1 TLB ミスの比率が高い場合、多くの L1 TLB ミス、そして L2 TLB ミスが発生しています。これは、L2 TLB の容量がプログラムのワーキングセットを保持可能で、プログラムはラージページによるメリットがあることを意味します。

推奨するチューニング

複数のデータ・ストリームを含むループでは、TLB への負荷が減るようにループを複数のループに分割します (キャッシュの局所性の向上にもつながります)。ループ内でアクセスされたアドレスが 2 のべき乗ではない場合、連想性競合により TLB の有効なサイズは減少します。1 つの 4KB ページで配列間をパディングすることを検討してください。

L2 比率に対して L1 比率が高い場合、ラージページを使用することを検討してください。

一般に、空間の局所性を向上するプログラムの変更は、キャッシュの効率と TLB の効率の両方を高めます。TLB は単に別の種類のキャッシュと言えます。

5.3 VPU の使用率

使用されるイベント

イベント	意味
VPU_INSTRUCTIONS_EXECUTED	スレッドで実行された VPU 命令数
VPU_ELEMENTS_ACTIVE	VPU 命令でアクティブなベクトル要素の数、またはベクトル演算の数 (各命令は複数のベクトル演算を実行するため)。

式

評価基準	式
ベクトル化強度	VPU_ELEMENTS_ACTIVE / VPU_INSTRUCTIONS_EXECUTED

しきい値

評価基準	以下の場合は調査:
ベクトル化強度	< 8 (倍精度)、< 16 (単精度)

説明と使用方法

FLOPS (1 秒あたりの浮動小数点演算の回数) で効率を測定すると、マシンのピーク浮動小数点パフォーマンスと容易に比較できるため便利です。しかし、インテル® Xeon Phi™ コプロセッサーには浮動小数点演算をカウントするイベントがありません。代案は、実行されたベクトル命令の数を測定することです。

ベクトル命令には、浮動小数点演算を行う命令、ベクトルレジスターにメモリーからデータをロードしてメモリーにストアする命令、ベクトル・マスク・レジスターを操作する命令、ベクトルシャッフルのような特殊命令が含まれます。

フルベクトルで行われるベクトル演算は、ハードウェアの「すべて 1 の」マスクレジスター %k0 を使用します。このため、2 つのフルベクトルでベクトル演算が行われると、VPU_ELEMENTS_ACTIVE イベントは 16 (単精度) または 8 (倍精度) インクリメントされます。スカラー FP 演算はベクトルレジスターを利用してコンパイラーにより一般に生成されますが、スカラー演算を示すマスクは 1 つのベクトル要素にのみ適用されます。

このため、ループがどの程度適切にベクトル化されているかを確認する方法は、ループ中のすべてのアセンブリー命令に VPU_ELEMENTS_ACTIVE と VPU_INSTRUCTIONS_EXECUTED の値を加えて比率を調べることです。この数が 8 または 16 に近い場合、ループは適切にベクトル化されています。ベクトル化強度は倍精度では 8、単精度では 16 を超えることはありません。この数が小さい場合、ループは適切にベクトル化されていません。

この手法は、コンパイラーのベクトル化レポートと併せて使用してください。

この手法を広範囲のコードに適用する場合は注意が必要です。コード生成における多様な振る舞いやマスク操作命令がベクトル命令としてカウントされることで、比率がおかしくなり、誤った結論を導く恐れがあります。

推奨するチューニング

低いベクトル化強度は、コンパイラーが特定のループのベクトル化に失敗したか、ベクトル化が非効率だったことを示します。ベクトル化レポートを検証することで、問題に対する考察を得られます。問題の原因は通常、以下の 1 つまたは複数です。

1. 不明なデータ依存関係。#pragma simd および #pragma ivdep を追加して、不明な依存関係を無視するように、あるいは依存関係がリダクションのような特定の種類の依存関係であることをコンパイラーに知らせることができます。
2. 非ユニットストライドなアクセス。これらのアクセスは、多次元配列のインデックス、または構造体配列のフィールドへのアクセスによるものです。ループ変換およびデータ構造の変換により、これらのアクセスを減らせます。
3. 純粋な間接指定 (配列要素である添字を含む配列のインデックス)。これらはアルゴリズムに依存するため、排除するには大幅なデータ構造の再構成が必要です。

5.4 メモリー帯域幅

使用されるイベント

イベント	意味
L2_DATA_READ_MISS_MEM_FILL	メモリー読み込みに起因するリード操作の数。プリフェッチを含みます。
L2_DATA_WRITE_MISS_MEM_FILL	メモリー読み込みに起因するライト操作の数。書き込みはコヒーレンシーを維持するためメモリー所有権の読み込み (RFO) トランザクションにより実装されます。プリフェッチを含みます。
L2_VICTIM_REQ_WITH_DATA	メモリー書き込み操作に起因するエビクション (追い出し) の数
HWP_L2MISS	L2 をミスしたハードウェア・プリフェッチの数
SNP_HITM_L2	L2 で変更されたデータをヒットしたスヌープの受信数 (L2 エビクションに起因)
CPU_CLK_UNHALTED	サイクル数

式

評価基準	式
読み込み帯域幅 (バイト/クロック)	(L2_DATA_READ_MISS_MEM_FILL + L2_DATA_WRITE_MISS_MEM_FILL + HWP_L2MISS) * 64 / CPU_CLK_UNHALTED
書き込み帯域幅	(L2_VICTIM_REQ_WITH_DATA + SNP_HITM_L2) * 64 /

(バイト/クロック)	CPU_CLK_UNHALTED
帯域幅 (GB/秒)	(読み込み帯域幅 + 書き込み帯域幅) * 周波数 (GHZ)

しきい値

評価基準	以下の場合は調査:
帯域幅	< 80GB/s

説明と使用方法

この式は、メモリー読み込みまたは書き込みが発生するすべての種類のイベントによって生じるデータ転送を合計して帯域幅を計算します。ストリーミング・ストアは考慮されません。ストリーミング・ストアを使用するアプリケーションでは、帯域幅は少なく評価されます。

コアがメモリーを読み込む命令を実行すると、L2 キャッシュと L1 キャッシュの両方にデータが格納されます。両方のキャッシュにデータがなければ、コアは別のコアのキャッシュまたはメモリーからデータを読み込みます。後者のケースでは L2_DATA_READ_MISS_MEM_FILL イベントが発生します。コアがメモリーに書き込む命令を実行する場合、最初に所有権読み込み (RFO) を実行してキャッシュ階層にデータを格納します。そのデータがメモリーから格納される場合、書き込み操作により L2_DATA_WRITE_MISS_MEM_FILL イベントが発生します。セクション 5.1 で説明したように、FILL イベントはさまざまな種類のプリフェッチを含みます。このため、FILL イベントを (要求されたデータのみを仮定する) ヒットレートの計算に用いることはできませんが、プリフェッチは実際の帯域幅を消費するため、帯域幅の計算には利用できます。

L2 エントリーにデータを保持が要求された時に、利用可能なラインがない場合、コアは何れかのラインを追い出します。そのラインが変更されていると、メモリーに書き戻す必要があります。その場合、L2_VICTIM_REQ_WITH_DATA イベントが発生します。あるコアのキャッシュでデータが変更され、別のコアでそのデータが必要な場合、最初のコアは、そのデータを追い出しするスヌープ・ヒット・モディファイド (HITM) イベントを受け取ります。その場合、SNP_HITM_L2 イベントが発生します。通常、スヌープにより 2 つ目のコアに対してキャッシュからキャッシュへの転送が行われます (セクション 5.1 を参照)。しかし、コアが clevict 命令を使用している場合、同じコアで生成された場合でもスヌープ受信として見なされます。このイベントを無視しても通常は安全ですが、コンパイラーやランタイムが clevict 命令を (ストリーミング・ストアと組み合わせて) 利用する場合、 2 つのコアで共有される変更データが多いと、(キャッシュからキャッシュへの転送が含まれるため) このイベントを含めるとメモリー帯域幅が大きく評価されます。

この手法は、コアイベントから帯域幅を計算します。メモリー・コントローラーで見つかったアンコアイベントからサンプルを収集する方法もあります。インテル® VTune™ Amplifier XE の「Bandwidth (帯域幅)」プロファイルは、アンコアのサンプリングを行います。ほとんどの場合、どちらの方法でもメモリー帯域幅はほぼ同じ値になります。

推奨するチューニング

開発者は、アプリケーションが占有するメモリー帯域幅を知っている必要があります。データセットがコアの L2 キャッシュに完全に収まる場合、メモリー帯域幅の値は少なくなります。アプリケー

ションが多くのメモリー帯域幅を占有すると予想される場合 (例えば、長いベクトルをストリーミングする場合)、この手法により、理論的な帯域幅がどの程度達成されるか推定できます。

達成された帯域幅が > 140GB/s であれば、アプリケーションで占有可能なほぼ最大値です。達成された帯域幅がこの値未満の場合、セット連想性の競合、または不十分なプリフェッチにより、キャッシュの空間の局所性が低いと考えられます。極端な場合 (メモリーへのランダムアクセス)、多くの TLB ミスが発生します。

6 まとめ

パート 1 で説明したように、インテル® Xeon Phi™ コプロセッサーの潜在的なパフォーマンスを効率良く利用するには、アプリケーションを適切に並列化し、ベクトル化して、データの局所性を活用する必要があります。この記事の評価基準は、これら 3 つの要素の問題に対するマイクロアーキテクチャーの影響を判断する上で役立つでしょう。現在、インテル® Xeon Phi™ コプロセッサー向けにソフトウェアを最適化する開発者を支援する多くの情報が提供されています。セクション 6.1 に関連情報へのリンクをリストします。セクション 6.2 は、この記事の評価基準を計算するのに必要なすべてのイベントの表を示しています。

6.1 関連情報

インテル® VTune™ Amplifier XE 2013 Evaluation Center (英語)	http://software.intel.com/en-us/intel-vtune-amplifier-xe-2013-evaluation-options/
インテル® VTune™ Amplifier XE 2013 製品ページ (英語)	http://software.intel.com/en-us/intel-vtune-amplifier-xe/
インテル® Xeon Phi™ コプロセッサー・デベロッパー・ポータル	http://www.isus.jp/article/idz/mic-developer
インテル® メニー・インテグレーテッド・コア・アーキテクチャー向けのコンパイラー手法 (パフォーマンスの最適化を含む)	http://www.isus.jp/article/mic-article/xeon-phi/

6.2 この記事で紹介したイベントのリスト

これらのイベントは、インテル® VTune™ Amplifier XE 2013 で [New Knights Corner Hardware Event-based Sampling Analysis] を選択してカスタム解析を生成することにより収集できます。イベントは個々に追加できます。

CPU_CLK_UNHALTED	http://software.intel.com/en-us/intel-vtune-amplifier-xe-2013-evaluation-options/
インテル® VTune™ Amplifier XE 2013 製品ページ (英語)	コアで実行されたサイクル数
INSTRUCTIONS_EXECUTED	スレッドで実行された命令数
VPU_ELEMENTS_ACTIVE	スレッドで実行された VPU 命令数

DATA_READ_OR_WRITE	スレッドの L1 データキャッシュへのロードとストアの数
DATA_READ_MISS_OR_WRITE_MISS	スレッドの L1 キャッシュをミスした要求されたロードまたはストアの数
L1_DATA_HIT_INFLIGHT_PF1	L2 から L1 にすでにプリフェッチされたキャッシュラインへの要求されたロードまたはストアの数
DATA_READ_OR_WRITE	スレッドの L1 データキャッシュへのロードとストアの数
EXEC_STAGE_CYCLES	スレッドが演算を実行していたときのサイクル数
L2_DATA_READ_MISS_CACHE_FILL / L2_DATA_WRITE_MISS_CACHE_FILL	L2 読み込みまたは (同じカードの) 別のコアの L2 キャッシュで処理された所有権ミスの読み込みをカウントします。ローカル L2 キャッシュをミスした L2 プリフェッチを含むため、要求されたキャッシュフィルの決定には役立ちません。
L2_DATA_READ_MISS_MEM_FILL / L2_DATA_WRITE_MISS_MEM_FILL	L2 読み込みまたは (同じカードの) メモリーで処理された所有権ミスの読み込みをカウントします。ローカル L2 キャッシュをミスした L2 プリフェッチを含むため、要求されたメモリーフィルの決定には役立ちません。
DATA_PAGE_WALK	L1 TLB ミスの数
LONG_DATA_PAGE_WALK	L2 TLB ミスの数
VPU_INSTRUCTIONS_EXECUTED	スレッドで実行された VPU 命令数
L2_VICTIM_REQ_WITH_DATA	メモリー書き込み操作に起因するエビクション (追い出し) の数
HWP_L2MISS	L2 をミスしたハードウェア・プリフェッチの数
SNP_HITM_L2	L2 で変更されたデータをヒットしたスヌープの受信数 (L2 エビクションに起因)

表 5: この記事で使用したインテル® Xeon Phi™ コプロセッサーのイベント

著作権と商標について

インテルによる書面での合意がない限り、インテル製品は、その欠陥や故障によって人身事故が発生するようなアプリケーションでの使用を想定した設計は行われていません。

インテル製品は、予告なく仕様や説明が変更される場合があります。機能または命令の一覧で「留保」または「未定義」と記されているものがありますが、その「機能が存在しない」あるいは「性質が留保付である」という状態を設計の前提にしないでください。これらの項目は、インテルが将来のために留保しているものです。インテルが将来これらの項目を定義したことにより、衝突が生じたり互換性が失われたりしても、インテルは一切責任を負いません。この情報は予告なく変更されることがあります。この情報だけに基づいて設計を最終的なものとしないでください。

本資料で説明されている製品には、エラッタと呼ばれる設計上の不具合が含まれている可能性があり、公表されている仕様とは異なる動作をする場合があります。現在確認済みのエラッタについては、インテルまでお問い合わせください。

最新の仕様をご希望の場合や製品をご注文の場合は、お近くのインテルの営業所または販売代理店にお問い合わせください。

本資料で紹介されている資料番号付きのドキュメントや、インテルのその他の資料を入手するには、1-800-548-4725 (アメリカ合衆国) までご連絡いただくか、インテルの Web サイトを参照してください。

パフォーマンスに関する注意事項

* パフォーマンスおよびベンチマークの結果に関する詳細は、www.intel.com/benchmarks (英語) を参照してください。

コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。