パワーキャップによる待機ノードの電力制御と削減効果

疋田 淳一 [1)], 當山 達也 [1)], 島袋 友里 [1)], 戸田 庸介 [1)]

1) 京都大学 情報部

hikita.junichi.5n@kyoto-u.ac.jp

Power Control and Reduction Effect of Idle Nodes Using PowerCap

Junichi Hikita [1)], Tatsuya Tohyama [1)], Yuri Shimabukuro [1)] , Yosuke Toda [1)]

1) Information Management Department, Kyoto University

概要

スーパーコンピュータシステムのように多数の演算ノードで構成される大規模システムにおいて、省電力化は環境負荷の軽減や電力コスト削減の観点から重要な課題である。Intel プロセッサが備える RAPL（Running Average Power Limit）のパワーキャップ機能を活用することで、ハードウェアレベルでの電力制御が可能となる。本稿では、待機状態の演算ノードに対して積極的な電力制限を適用し、その効果を実測により評価した。

1 はじめに

京都大学学術情報メディアセンター（以下、「本センター」という。）が運用するスーパーコンピュータシステム[1]の運用においては、バッチジョブスケジューラによる研究者への計算リソース割り当てを行っている。システムを構成する演算ノードの稼働率は、実行待ちのジョブ数やサイズ、演算ノードへのアサインの都合等により、複数の要因で 100%に満たないタイミングが生じるものである。特に、本センターの運用方針として、資源保障型のジョブスケジューリングを採用しており、研究者が投入したジョブのターンアラウンドタイムを考慮して、選択したコースに応じて一部または全部の演算ノードを契約者に対して常に確保する運用を取り入れている。そのため、現在の受け入れ方針の上限設定においては、システム全体の稼働率は 70～80%で推移する傾向にある。

システムの効率的な運転のために、過去には、実行待ち状態のジョブ状況に応じて待機状態のノードを自動で起動/停止する運用に取り組んでいたこともあるが、停止状態からの復帰には時間を要する欠点や、1000 ノードを超える大規模環境において、ジョブスケジューラによる演算ノードの状態管理が複雑化する欠点がある。そのため、現行システムの 1 世代前のシステムでは、CPU の省電力機能が進歩している状況を踏まえ、自動的な電源制御はせず、手動による意図的な縮退運転に留める運用をしていた。しかし、昨今の電気代の高騰により、従前の価格帯への回復が難しい状況においては、消費電力削減の重要性が高まっており、運用上の大きな課題である。

CPU の消費電力の削減の手法としては、従前から DVFS（動的電圧周波数制御）や C-State に代表される CPU アイドル状態の省電力機能の実装がされてきた。しかし、スーパーコンピュータは高性能であることが目的のシステムであることから、本センターの設定としてはパフォーマンスを重視した CPU のモードを採用しており、DVFS や C-State の効果は制限された状態となっている。一方で、パフォーマンスを必要としない待機状態の演算ノードに対しては電力削減の余地がある状態と言えるため、パワーキャップ機能に着目し更なる消費電力の削減方法の調査を行った。

RAPL[2]は、Intel プロセッサが提供するハードウェアベースの電力制御機構であり、OS やアプリケーションをカスタマイズすることなく、CPU 単位で即時に平均電力制限を適用することができる。演算ノードのハードウェア全体を対象としたプラットフォームレベルの電力制限も存在するが、今回は運用中の柔軟な適用性と即応性を重視し、RAPL による CPU パッケージに対するパワーキャップ制御を用いることとした。

本稿では、RAPL を用いた待機ノードへの電力制限の導入とその効果を評価し、本センターのスーパーコンピュータシステムに適用した削減効果について報告する。

2　システム環境

本センターが運用するスーパーコンピュータシステムの内、RAPL に対応したシステムの構成図を図 1 に示す。Camphor 3、Laurel 3、Cinnamon 3 の 3 種のシステムは Intel Xeon (Sapphire Rapids) をノードあたり 2 基搭載しており、RAPL の制御が可能なノード数の合計が 1506、CPU 数の合計が 3012 である。

Camphor 3 の CPU である Intel Xeon CPU Max 9480 の TDP は 350 W、Laurel 3 及び Cinnamon 3 の CPU である Intel Xeon Platinum 8480+の TDP は 350 W である。CPU の合計電力を TDP から単純計算すると、 1,054,200 W（1506 ノード×CPU 2 基×350W）である。

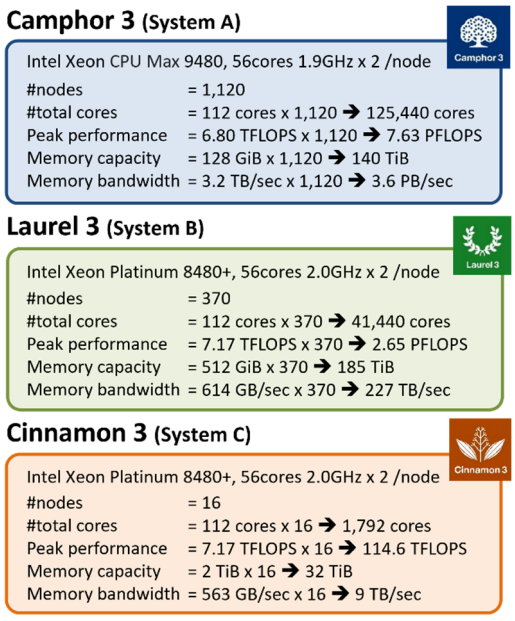

図 1　システム仕様

3　RAPL による節電効果

RAPL によるパワーキャップの効果を検証するため、無効にした状態と有効にした状態（以下、それぞれ「キャップ無効」、「キャップ有効」という。）における電力消費の変化を測定した。キャップ有効時の利用可能な電力の制限値は、制限値を徐々に下げていき省電力の効果を最大限確認できた値を用いている。測定環境は、OS およびシステム運用に必要なサービスは起動しているが、ユーザプロセスは存在していない状態を待機状態として実施した。なお、パワーキャップの有効化により処理性能が低下した状態であっても、OS 動作や管理上必要な処理においては、致命的な処理の遅延やエラー等の問題が生じないことを確認している。

turbostat コマンドによりノードに搭載している 2 基の CPU 待機電力の合計値を取得し、比較した結果を図 2 に、IPMI によりノード待機電力の値を取得し、比較した結果を図 3 に示す。いずれも、キャップの有効/無効による変化を、対象システムごとに並べて表記している。turbostat のデータについては、複数回実行した内の最も低い値、IPMI のデータについては、10 分間隔のサンプリングデータを 24 時間分取得した結果の平均値を使用している。

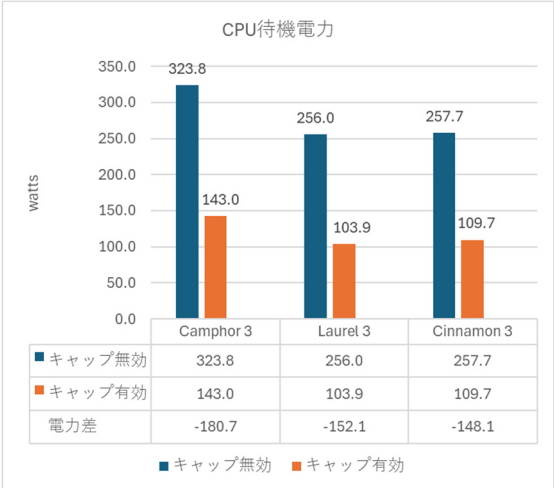

図 2　キャップ有効時の CPU 待機電力

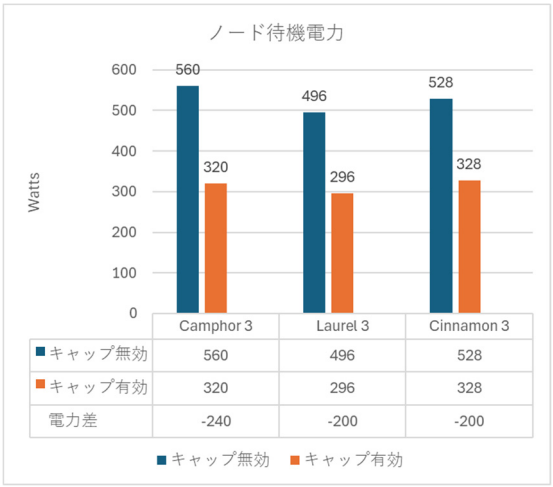

図 3　キャップ有効時のノード待機電力

図 2 の Laurel 3、Cinnamon 3 の結果については、キャップ無効時の 256.0 W、257.7 W に対し、有効時は 103.9 W、109.7 W に抑えられており、59%と 57%の削減率である。Camphor 3 については、他と TDP は同じであるが、HBM2e メモリをチップ上に搭載していることから、CPU の待機電力としては、他と比較して高い状況であるものの、削減率は 56%と同等となっている。今回評価した同一コア数且つ同一 TDP の CPU においては、同程度の削減効果を確認した。

図 3 のノードの待機電力については、ノード全体の消費電力であり、図 2 の CPU に限定した測定値よりも大きな削減効果を確認した。CPU の消費電力を抑えることで、冷却用の FAN、メモリ、I/O や周辺装置、電源ユニット等の電力効率も含めて波及した効果ではないかと考えている。いずれのシステムもノードの消費電力として 200W 以上の削減効果を示しており、ノードの待機状態においては、ノードあたり 38%～43%の削減効果があることを確認した。

4 C-State の節電効果

C-State の効果を RAPL と比較するために、BIOS の設定変更により、C-State の省電力機能としてより深い節電状態である C1E 及び C6 状態を有効化した場合の測定を行った。Laurel 3 のノード待機電力を図 4 に示す。図 2 の Laurel 3 のデータに、C1E を有効にした状態、C1E 及び C6 を有効化した状態、C1E 及び C6 とパワーキャップを併用した状態の消費電力を追記したグラフである。消費電力は、10 分間隔のサンプリングデータとして 24 時間分取得した結果の平均値を使用している。

C1E 有効及び C1E/C6 有効の場合における電力削減の効果は確認したが、C1E/C6 有効と比較しキャップ有効の方が 20 W 低い結果となり、RAPL の優位性が確認された。併用した場合は、キャップ有効と比較しさらに 21 W 低い結果となった。消費電力の観点を重視すると併用が望ましい結果ではあるが、設定変更には BIOS の変更及び再起動が必要となることから、管理性やパフォーマンス影響を考慮し、C1E/C6 による深い省電力機能については実運用には採用しない方針とした。

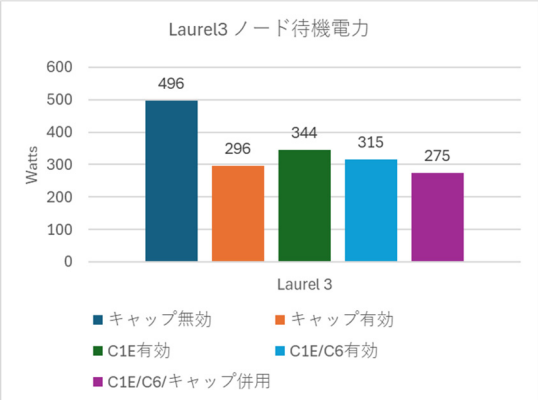

図 4 C-State 有効時のノード待機電力

5 運用システムへの適用

5.1 RAPL 適用の概要

RAPL によるパワーキャップを、本センターのシステム全体に動的に適用するためには、システム管理の効率性を維持したうえで、ジョブスケジューラとの連携について考慮する必要がある。システム管理の観点では、管理者による保守作業の際の応答速度や、管理上必要なプログラムの動作速度に対する考慮が必要となり、ジョブスケジューラとの連携においては、ユーザプログラムの実行速度に影響を与えないための考慮が必要となる。

RAPL によるパワーキャップは即時反映されるため、図 5 に示す状態遷移を実現することで、有効／無効の変化が運用に与える影響を最小化する方針として、キャップの有効/無効を自動制御するプログラムを実装した。制御には後述する RAPL 設定ツールと状態管理常駐プログラムを用いている。

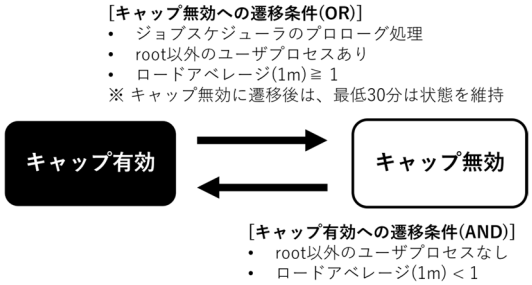

図 5 状態遷移

5.2 RAPL 設定ツール

RAPL の設定を制御するためのツールとして、github で公開されている powercap ソフトウェア[3]をシステムにインストールし、加えてより簡便に操作可能なラッパープログラムを用意することで、コマンドラインインターフェイスによる操作を可能とした。別途用意する自動制御用の状態管理常駐プログラム、ジョブスケジューラのプロローグ処理、管理者による手動操作のいずれの方法を用いたとしても同一コマンドで制御することが可能となるように設計している。

5.3 状態管理常駐プログラム

システムの状態から RAPL の状態を自動管理するために、RAPL 設定ツールを用いて自動制御する常駐プログラムを用意している。プログラムは、Linux の /proc の情報からユーザプロセスの存在とロードアベレージの情報を取得して制御する方式を採用し、メモリ上のデータのみで動作可能な

低負荷な仕組みとして実装している。状態管理をジョブスケジューラとは独立した設計とすることで、既存のジョブスケジューラへの影響を最小化し、運用の複雑化を回避している。

キャップの有効／無効の制御のルールは図 5 に示すとおりであり、root 以外のログインユーザが存在しない且つロードアベレージが 1 未満の場合は、キャップを有効にすることで節電状態に遷移する。そうでない場合は、キャップを無効状態にすることで、通常状態に戻すというシンプルな条件としている。管理者が一般のローカルユーザ経由で SSH ログインした場合や、root ユーザのプロセスに負荷がかかった場合には、自動でキャップが無効になるため、判定までの若干のタイムラグを除いて、運用への影響を抑えることができる。なお、状態確認の頻度は任意の値を設定することが可能であるが、現在は 1 分間隔で稼働している。

5.4 ジョブスケジューラ連携

ジョブスケジューラによるユーザプログラム実行時のキャップ無効判定のタイムラグを解消するために、ジョブ実行開始前のプロローグ処理において、キャップを無効にする操作を取り入れている。ジョブ実行中はユーザプロセスが存在するため、キャップの無効状態が継続する仕組みである。ジョブ実行が完了しノードが待機状態に戻った後は、常駐プログラムによる制御に任せる方式としている。

6 節電効果

RAPL による制御を未適用な状態の長期的な電力データは取得できていないため、キャップ有効の消費電力のデータ、システムの稼働率、ノードあたりの節電効果を元に、キャップの有無による節電効果を計算してグラフ化したものを図 6 に示す。データの範囲は、執筆時の直近のデータとして 2024 年 8 月～2025 年 7 月の 1 年間の消費電力について、月ごと推移を示している。待機電力の削減の取り組みのため、ノードの稼働率（ノードでジョブが実行されている割合）により効果が変動するため、月によってキャップの効果に差が生じていることが分かる。12 ヶ月分の平均削減率は 7.67%となっており、その時期の電気代単価の平均が約 22 円/kWh であったことから、電気代への影響としては約 1,800 万円の削減効果があったことが推定される。この結果は、大規模システムに対して RAPL による待機ノードの省電力化の有効性を示している。

7 まとめ

本稿では、Intel RAPL を用いた待機ノードへの電力制限による電力削減効果を確認した。スーパーコンピュータシステムに適用することで、電力及び電気代の削減として効果的な手法であることを示した。待機中の演算ノードが存在しない際には効果のない手法ではあるが、特に本センターの資源保障を行うジョブスケジューリングとの相性が良いため、今後の運用においても引き続き効果を期待している。

参考文献

[1] 深沢圭一郎, 新スーパーコンピュータシステムのご紹介, 京都大学情報環境機構広報誌「Info!」, Vol. 28, p10-12, 2023.

[2] Intel Corporation, Intel® 64 and IA-32 Architectures Software Developer's Manual: Combined Volumes 3A, 3B, 3C, and 3D: System Programming Guide, Intel Corporation, 2023.

[3] powercap. https://github.com/powercap/powercap

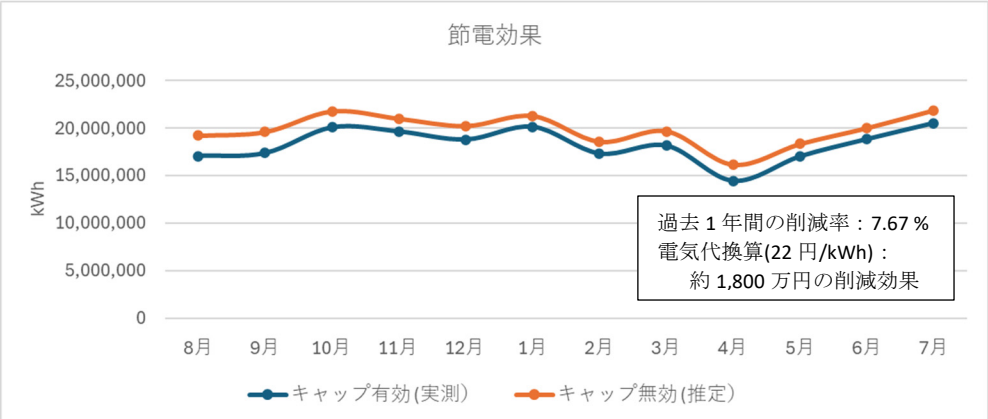

図 6 節電効果（2024 年 8 月～2025 年 7 月）