Vol.2020-HPC-175 No.25
2020/7/31

スーパーコンピュータ「富岳」における電力制御の評価

児玉 祐悦[1,a)]　小田嶋 哲哉[1]　有間 英志[2]　佐藤 三久[1]

概要：
スーパーコンピュータ「富岳」では性能を維持しつつ消費電力を削減するパワーノブ機能や、周波数を上げて性能を向上させるブーストモード、さらには使用しないコアを低電力に設定するリテンション機能を有している。本稿ではこれらの電力制御の効果を、富岳共用前評価環境で評価した結果について報告する。エコモードを用いることにより、メモリバンド幅律速となるアプリケーションでは、性能を維持したまま、ノーマルモードに対して電力を 15 %程度削減できることを確認した。また、演算律速となるアプリケーションではブーストモードを用いることにより、ノーマルモードに対して 10 %程度の性能向上を達成できるが、電力は 17 %程度増大する事を確認した。

1. はじめに

理化学研究所では、日本における次世代のフラグシップスーパーコンピュータとして「富岳」を開発してきた。現在、富岳は全ノードの納入が完了し、2021 年春の一般共用を目指して調整を進めている。また、納入されたノードの一部が富岳共用前評価環境として限定されたユーザに提供されている。本稿では、この共用前評価環境を利用して、富岳に搭載されている様々な電力制御機構を用いた評価を行い、その効果について検討を行う。

本稿では、まず、富岳の概要について述べた後、富岳における電力制御機構について詳しく説明する。次に、共用前評価環境を用いて、それらの電力制御機構の効果について評価を行う。最後にまとめを述べる。

2. 富岳の概要

富岳は総ノード数 158,976 の大規模なスーパーコンピュータである。432 ラックから構成され、1 ラックに 384 ノードを搭載している。ただし、一部には 192 ノードを搭載しているラックもある。2.0GHz で動作し、倍精度理論最大性能は 488PFLOPS である。

各ノードは 1 チップのプロセッサ Fujitsu A64FX[1], [2] から構成される。A64FX は Armv8-A の 64 ビットアーキテクチャであり、SVE(Scalable Vector Extension) を実装した最初のプロセッサである。A64FX では SVE として 512 ビットの SIMD 演算パイプラインを 2 本持っている。A64FX はパッケージ内に搭載された HBM2 を 4 基搭載しており、容量 32GB、スループット 1024GB/s のメモリ性能を実現している。A64FX は計算コアを 48 個持つメニコアプロセッサであり、内部では 4 つの CMG（Core Memory Group）に分かれている。各 CMG は 12 コア、8MB の L2 キャッシュ、メモリコントローラから構成され、4 つの CMG はリングバスで接続されている。ノード間ネットワークは Tofu Interconnect D (TofuD) という京と同様の 6 次元メッシュ／トーラス接続であり、ネットワークコントローラや PCI Express コントローラもチップ内に実装している。

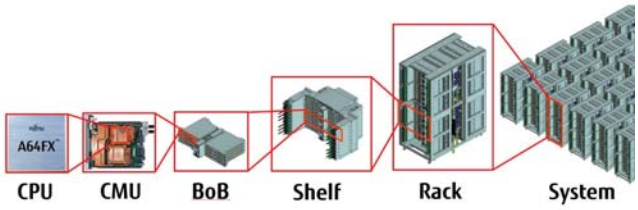

図 1　富岳のシステム構成

図 1 に富岳のシステム構成を示す。1 ノードは 1CPU であり、2 ノードが CMU（CPU Memory Unit）と呼ぶボードに搭載されている。8 CMU（16 ノード）を BoB（Bunch of Blades）と呼ぶ制御単位とし、3 BoB を組み合わせたも

[1] 理化学研究所 計算科学研究センター
[2] 東京大学 情報基盤センター
[a)] yuetsu.kodama@riken.jp

のを Shelf と呼び、最小構成単位となっている。8 Shelf がラックに組み込まれており、1 ラックに 384 ノードを搭載している。

富岳には 2 種類のノードがある。一つは計算処理を行う計算ノードであり、もう一つは計算処理とともに I/O 処理を行う計算ノード兼 I/O ノード（以下では I/O ノードと略記する）である。各ノードには計算コアの他に、OS 処理を行うアシスタントコアが搭載されている。アシスタントコアは計算コアと同じコアである。計算ノードにはアシスタントコアが 2 コア、I/O ノードにはアシスタントコアが 4 コア搭載されている。I/O ノードには BIO, SIO, GIO の三種類がある。BIO はブート用の不揮発メモリを搭載したノードで、BoB（16 ノード）に 1 ノード存在し、他のノードがネットワークブートする際のサーバとなる。SIO は BoB（16 ノード）で共有する第一階層のストレージである SSD を搭載し、そのサーバとなる。GIO は、第二階層のクラスタストレージである FEFS に対して、InfiniBand を通じてアクセスするノードで、1 ラック（384 ノード）に 2 ノード存在する。

3. 富岳の電力制御機構

富岳の開発当初から、電力削減は重要な課題であった。京コンピュータは、10.5PFLOPS の性能を 12.7MW で実現していた。富岳の開発目標では、アプリケーション性能 100 倍を電力 30-40MW で実現することとされており、電力効率として 20 倍以上の向上を求められていた。

電力効率の向上のため、富岳のプロセッサである A64FX では、最先端の半導体プロセスを採用するとともに、電力効率を高める回路設計を行い、昨年の SC19 での Green500[3] リストでは 1 位を達成した。

このようなプロセッサでの低電力化を意識した設計の他に、A64FX ではいくつかの電力制御の機構を備えている。以下ではそれらについて述べる。

3.1 パワーノブ

富岳の開発コンセプトであるコデザインにおいて、電力効率を向上させるためにパワーノブと呼ぶ機構について検討を重ねてきた。これは、アプリケーション性能を出来るだけ維持しつつ、そのアプリケーションでは使用しない回路の電力を削減することにより、電力効率を上げようという基本コンセプトに基づく機構である。本機構では、いくつかのパワーノブを用意し、アプリケーションごとにそのノブを on/off することにより、アプリケーションに応じたパワーノブの設定を選択し、性能を維持しつつ電力を削減することを目指していた。

A64FX で採用されたパワーノブは以下の通りである。

周波数　通常は 2.0GHz であるが、これを 1.6GHz に下げることが可能。また、2.2GHz に上げるブーストモードもあるが、これについては別途 3.2 節で説明する。

メモリ　通常は連続的にメモリアクセスが可能であるが、メモリアクセス頻度を最大 100 %から最小 10 %まで 10 %刻みで制限することができる。

命令発行　通常は 1 サイクルに最大 4 命令発行可能であるが、これを 2 命令発行に制限することができる。

整数演算器　通常は 2 本の整数演算パイプラインが利用可能であるが、これを 1 本に制限することができる。

浮動小数点演算器　通常は 2 本の浮動小数点演算パイプラインが利用可能であるが、これを 1 本に制限することができる。

これらのパワーノブの設定は、ジョブスクリプトによる指定によってジョブ単位で指定することが可能である。また、Sandia Power API[4], [5] を利用して、各アプリケーション内でより細かく制御することも可能である。

3.2 ブーストモード

富岳向けの A64FX では、電力効率を重視して、ノーマルモードとして周波数は 2.0GHz としている。一方で、少しでも性能を向上させたい、という要求に応えるために、2.2GHz のブーストモードを用意している。しかし、富岳ではノーマルモード時の電源電圧は可能な限り低く抑えているため、2.2GHz で動作させるためには電源電圧を上げる必要があり、ブーストモードでは電力効率は低下してしまう。また、A64FX では周波数は全コアで共通であるため、あるコアのみを 2.2GHz にすることは出来ない。

ブーストモードで全システムが動作すると、アプリケーションによっては施設の供給可能電力を超えてしまう可能性がある。これまでのセンター運用では、最大電力を考慮して、施設の供給電力を準備するか、ノード数を決定することが通常である。一方、富岳では、施設の供給電力ならびに冷却性能を超えないように、アプリケーション側で動作させるモードを選択する方針としている。具体的な運用については、現在検討中であるが、ブーストモードでは最大電力を超える可能性があるアプリケーションではノーマルモードで実行するか、3.3 節で述べるエコモードを利用するか、アプリケーション内で細かく Power API を用いて最大電力を超えないような制御を行う事が求められる。

しかし、意図せずにアプリケーションが高い電力となってなってしまった時には、上記のソフトウェアによる制約では電力を抑えることができない。そのため、富岳では BoB ごとに電力を監視して、設定した電力を超えた場合には、各ノードの周波数を下げることにより電力削減を図るとともに、この周波数低減後も電力超過が一定時間以上続いた場合には強制的に電源を落とすパワーキャッピング機構がついている。

3.3 エコモード

3.1 節で述べたパワーノブについて、基本設計時から評価ツールやシミュレータなどを用いて評価を行っていたが、期待したほどは電力が削減できなかった。これは、電力変動に対する安定動作機構により、演算器が動かなくてもある程度の電力を消費しているためであった。

浮動小数点演算パイプラインを 1 本に制限するパワーノブは、メモリバンド幅律速であるアプリケーションに対して電力削減効果が高いと期待される。また、浮動小数点パイプラインはプロセッサの中でも電力が大きいユニットであるので、これを 1 本にすると最大電力が削減されることは明らかである。そこで、このパワーノブを適用した際の最大電力に対応して安定動作を行うモードを新たに設定することにした。これをエコモードと呼ぶ。

このエコモードは、ブーストモードと直交した設定であるため、次の 4 つの組み合わせが可能である。ただし、ブーストモードは全コアで共通に設定されるのに対して、エコモードはコアごとに設定が可能であるため、ノーマルとノーマルエコ（以下ではエコと略記する）、あるいはブーストとブーストエコはコアごとの混在が可能である。

ノーマル　2.0GHz で浮動小数点演算パイプラインを 2 本使えるモード。

ノーマルエコ　2.0GHz で浮動小数点演算パイプラインを 1 本に制限したモード。

ブースト　2.2GHz で浮動小数点演算パイプラインを 2 本使えるモード。

ブーストエコ　2.2GHz で浮動小数点演算パイプラインを 1 本に制限したモード。

3.4 リテンション

大規模なシステムでは、ジョブが割り当てられないノードが生じる。例えば、100 ノードのシステムに対し 64 ノードを使う 2 つのジョブがサブミットされたとすると、一度には 1 つのジョブしか実行できないため、残りの 36 ノードはジョブが割り当てられない。より細かなジョブがサブミットされれば、残りのノードにジョブを割り当てることも可能であるが、京のようにあるノード数以上のジョブのみがサブミット可能である場合にはある程度のノードが割り当てられずに残ることになる。

このようなジョブが割り当てられないノードの電力を削減する目的で、富岳ではノードリテンションという状態を設定している。これは、1 つのアシスタントコア以外のコアをすべてコアリテンションという状態にする。コアリテンションとは、3.3 節のエコモードの説明で述べた電力安定動作機構をオフにした状態である。ノードリテンションの状態でもメモリ内容などは全て保持されているため、ノードリテンションから通常のアイドル状態へは数ミリ秒で遷移可能である。

ノードリテンションは、ジョブ待ちノードの電力削減を目的として採用された機能であるが、コアリテンションはコアごとの設定が可能であるため、アプリケーション実行でも有効であると考えられる。ただし、直接、コアリテンションへの移行を指定できるわけではない。ユーザが Power API を用いて、あるコアをコアリテンション可能と設定すると、OS がコアにプロセスが割り当てられていないことを判断して、そのコアをコアリテンションへと遷移させる。したがって、コアリテンション可能と設定されていてもプロセスが実行されている間は通常と同じくコアは動作する。

3.5 電力測定機能

富岳では電力測定方法が複数用意されており、要求に応じて使い分ける必要がある。大きく分けて富岳としての測定と、施設としての測定がある。

富岳としては、ノード毎に電力測定が可能であり、3.1 節のパワーノブで述べた Power API を用いてユーザがノード電力を取得可能である。ノード電力については、実測電力と推定電力の 2 種類がある。

実測電力の測定方法は以下の通りである。BoB 内にある FPGA が各 CMU に搭載されている DC/DC 変換の電流を元に電力量を測定しており、各ノードに定期的に分配している。その電力量を Power API を通じてユーザが取得し、その差分を測定時間間隔で割ることにより、平均実測電力を求めることができる。電力量は、ノード内では約 5 ミリ秒間隔で更新される。実測電力には、ノード毎にばらつきがある。これはノード構成に起因するばらつきと半導体プロセスのばらつきに起因するばらつきとがある。ノード構成の違いとは、計算ノードか I/O ノードかという違いである。さらに、計算ノード間でもノード構成による電力の違いがある。TofuD の光ケーブル (AOC) の電力は CMU 内の 2 ノードのうち片方に割り当てられている。そのため、AOC の電力が割り当てられているかいないかで、計算ノードにも電力に違いがある。

実測電力には種々のノード間のばらつきが存在するため、アプリケーションの電力チューニングに用いるには不適切である。そのため、A64FX ではノードによらずに同じ動作をしたときには同じ電力量となる推定電力量を、Power API を通じてユーザに提供している。この推定電力量を測定時間間隔で割ることにより平均推定電力を求めることができる。この推定電力量は約 1 ミリ秒間隔で変化し、ノード全体の電力量のほかに、各 CMG のコア群の電力量、各 CMG の L2 の電力量、各 HBM の電力量、アシスタントコアの電力量などノード内の推定電力量の内訳を取得することも可能になっている。詳細は Power API ユーザーズガイド [6] を参照のこと。

ジョブ全体のノード毎の実測電力および推定電力は、ジョ

ブスクリプトで指定することにより、ジョブ実行の統計情報として取得することが可能である。-S を指定することによりノード単位のデータを、-s を指定することによりジョブトータルのデータのみを取得できる。ノード数が大きくなるとノード単位のデータはかなり大きくなるので注意が必要である。この統計情報には、平均電力の他に、最大電力も出力される。ただし、この最大電力は一分単位の平均電力の最大値であり、ノードデータとしては意味を持つ。一方、各ノード電量が最大となるタイミングが異なるジョブでは、ジョブトータルの最大電力は、実際の最大値より大きな値となる場合があることに、注意が必要である。

一方、施設としても以下のような電力測定が複数のレベルで可能となっている。

- ラック単位での電力を 1 分単位で全ラックで測定可能。ただし、ラック単位での電力のため、ラックを占有して測定する必要がある。
- 54 ラック単位での電力を 6,600V の高圧幹線系統で 1 秒単位で測定可能。ただし、4 系統のみ実装されており、全体の 1/2 のラックのみ測定可能。Green500 の測定は本測定結果を用いており、レベル 2 の測定が可能。各系統の 54 ラックを占有して測定する必要があり、大規模実行で全ノードを占有するような場合でないと利用するのは難しい。
- ラック単位での 1 秒単位の電力を測定可能な測定器が 1 台ある。任意のラックを指定して測定は可能であるが、ラックを占有して実行する必要がある。

4. 評価

富岳の電力評価ならびに電力制御の効果について、富岳共用前評価環境を用いて評価を行った。なお、富岳共用前評価環における評価結果は、スーパーコンピュータ「富岳」の供用開始時の性能・電力などの結果を保証するものではない。

4.1 Stream によるエコモードの評価

メモリバンド幅律速なベンチマークとして Stream ベンチマークを用いて、ノーマルモードとエコモードの比較を行った。同一の 1 ノードで、スレッド数を変化させて実行した。スレッドは各 CMG に均等に割り当てている。データサイズは 800MB と L2 キャッシュより十分大きなサイズとし、ファーストタッチで各スレッドのアクセスするデータが CMG 内のローカルなメモリに割り付けるようにしている。

本評価で用いた Stream ベンチマークでは、ソフトウェアプリフェッチと ZFILL 最適化を施している。ZFILL 最適化とは、キャッシュへの書き込み前のメモリからのプレロードを抑止するために、L2 キャッシュ上に 0 クリアしたラインを割り当てる最適化である。Stream ではプログラム上は 1 ループ当たり 2 ロード 1 ストアである。ZFILL 最適化を行わない場合は、ストアの前にプレロードが必要となるため、メモリアクセスとしては 3 ロード 1 ストアが必要となる。そのため、ベンチマークスループットとしてはメモリ性能の 3/4 しか出すことができない。一方、ZFILL 最適化を行う事によりメモリ性能をフルに出すことができる。

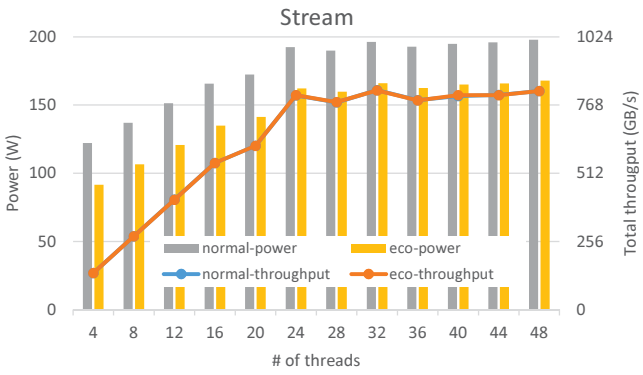

図 2 Stream の性能と電力

図 2 に評価結果を示す。横軸はスレッド数である。折れ線グラフがトータルのメモリスループットを示し、右の縦軸に対応する。ノーマルモード (*normal-throughput*) とエコモード (*eco-throughput*) のそれぞれのスループットを示しているが、ほぼ重なっており、モードによりスループットに差が無いことが分かる。24 スレッドまではスループットはスレッド数に比例して増大しているが、24 スレッドで飽和している。CMG あたり 6 スレッドでメモリスループットを使い切ることができている。スループットはメモリ理論性能の 8 割に当たる 800GB/s を実現している。なお、ZFILL 最適化を行わないハードウェアプリフェッチを用いた場合には 3-4 スレッドで飽和し、スループットは 600GB/s 程度である。

棒グラフがノードの実測電力で、左縦軸に対応し、ノーマルモード (*normal-power*) とエコモード (*eco-power*) の電力を示している。電力はスループットと同様に 24 スレッドまではスレッド数に比例して増大しており、24 スレッドで飽和している。エコモードの電力はノーマルモードの 75 %から 85 %であり、スレッド数が少ないほど差が大きい。エコモードでは Stream ベンチマークのようにメモリバンド幅律速で浮動小数点演算パイプラインのビジー率が低い場合には、性能を維持したまま電力を 15 %程度削減できることが確認できた。

なお、このグラフでは実測電力を示しており、あくまで利用したノードでの電力である。4.3 節のノード間のばらつきで示すようにノード間の電力ばらつきはそれなりに大きい。同一ノードで比較しないと電力の評価はできない。

また、電力評価を行う場合は、アクセスするデータの値にも注意が必要である。本 Stream はデータを 64 ビット

表 1　DGEMM の性能と電力

モード	性能 (GFLOPS)	実測電力 (W)	電力効率 (GFLOPS/W)
ノーマル	2915	121.6	24.0
ブースト	3205	141.9	22.6

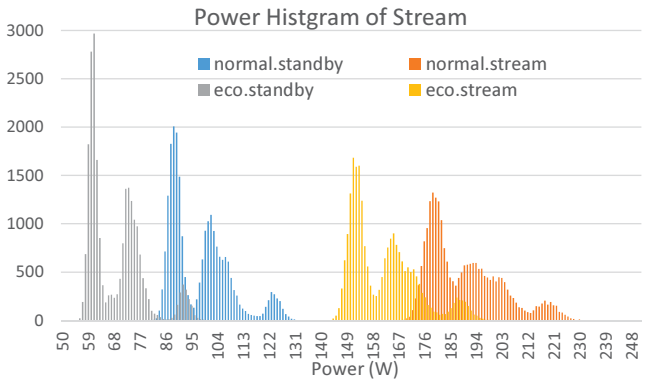

図 3　Stream 実行時電力のノード間ばらつき

整数としている。当初は定数で初期設定した配列をアクセスしていたが、電力は図 2 の値に比べて、13 %程度低かった。また、128 バイト単位でデータを 0 と-1 で入れ替わるようにした場合では 8 %程度低い電力となっていた。図 2 の結果はデータを乱数で設定した場合の電力である。

4.2　DGEMM によるブーストモードの評価

浮動小数点演算律速なベンチマークとして DGEMM のライブラリ呼び出しを行うベンチマークを用いて、ノーマルモードとブーストモードの比較を行った。同一の 1 ノードで 48 スレッドで実行した。DGEMM は富士通製の LAPACK ライブラリを用いており、データサイズは N=12000 の正方行列としている。

表 1 に評価結果を示す。性能は、ノーマルモード、ブーストモード共にピークの 95 %ほどを達成しており、ブーストモードはノーマルモードと比べて 9.5 %性能が向上している。一方、ブーストモードの実測電力はノーマルモードと比べて 16.5 %増大しており、電力効率では 5.8 %低下という結果になっている。

Stream と同様に、データの値についても注意が必要である。本結果は、配列を 1.0E-4 ずつ増加するような値で初期化した場合である。定数で初期化した場合は、これより 10 %ほど低い電力となっていた。また、乱数で初期化すると 4.3 節のノード間ばらつきの評価で示すように、20 %以上高い電力となっている。

4.3　電力のノード間ばらつきの評価

評価環境では 25,344 ノード（66 ラック相当）が利用可能であったので、それらのノード間の電力ばらつきについて評価を行った。

まずは、4.1 節で用いた Stream ベンチマークを用いて、ノーマルモードとエコモードの比較を行った。図 3 に各ノードの実測電力のヒストグラムを示した。図では 4 つのヒストグラムが重ねて表示されている。青がノーマルモードのスタンバイ実測電力 (*normal.standby*)、赤がノーマルモードの Stream 実行時の実測電力 (*normal.stream*)、グレイがエコモードのスタンバイ実測電力 (*eco.standby*)、黄色がエコモードの Stream 実行時の実測電力 (*eco.stream*) をそれぞれ表している。ただし、利用した評価環境では一部の I/O ノードがブーストモード固定となっていた。また、Stream の実行は 66 ラック規模であったが、実際には 72 ラックにまたがって実行されていた。そのため、図 3 はブーストモードに固定されていたノードを除いた総計 23,472 ノードの結果となっている。

それぞれのヒストグラムには 3 つの山が見える。これはノード構成の違いによる要因が大きいと思われる。ノード構成の違いとは、3.5 節で述べたように、I/O ノードかどうか、光ケーブルの電力が割り当てられている計算ノードかどうかといった違いである。*normal.standby* で考察すると、以下の通りとなる。ブーストモード固定となっていない I/O ノードが 1,728 ノード含まれており、それらの平均電力は 123W であり、電力が高い山に対応している。光ケーブル（AOC）の電力を含む計算ノードの平均は 104W であり、2 つ目の山に対応している。AOC の電力を含まない計算ノードの平均は 90W であり、一つ目の山に対応している。それ以外の電力ばらつきの原因は、半導体プロセスによる電力ばらつき、および、電源電圧によるばらつきが考えられる。ヒストグラムの各山が広がりを持っているのは半導体プロセスによるばらつきであると考えられる。後者は、A64FX にはいくつかの設定電源電圧があり、製造時のチェックにパスした一番低い電源電圧で動作させるようにして歩留まりの向上と電力削減を図っている。しかし、図 3 からはノード構成のばらつきの方が大きいため、電源電圧によるばらつきは読み取れない。AOC を含まない計算ノードのみを取り出してヒストグラムを生成してみると、2 つの山が存在し、山に含まれるノード数の比は 10:1 程度とほとんどが低い山に含まれていた。

スタンバイ時のヒストグラムに比べて、実行時のヒストグラムの広がりが大きくなっているのは、半導体プロセスのばらつきの影響が実行時の方が大きいためと思われる。一方、図には表示していないが、各ノードの Stream ベンチマークのメモリバンド幅はどのノードもほぼ同じであり、電力のばらつきは性能には影響していない。

ノーマルモードとエコモードを比較すると、スタンバイおよび実行時ともに 30W 程度のエコモードの電力削減効果が確認できる。

ノード単位では電力のばらつきは大きい。そこで、BoB やラック単位の平均ノード電力のばらつきを確認したのが図 4 である。BoB は、電力キャッピングの単位であり、16

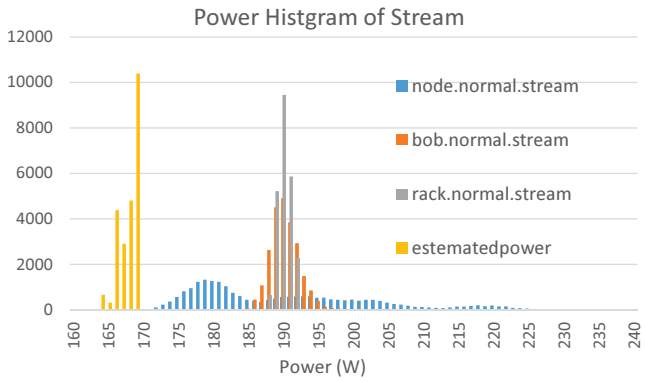

図 4　Stream 実行時電力のラック間ばらつき

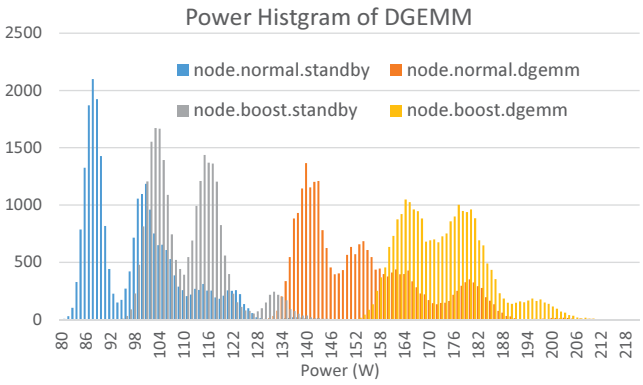

図 5　DGEMM 実行時電力のノード間ばらつき

ノードから構成される。ラックは 384 ノードから構成される。この図はノーマルモードの Stream 実行時のノード実測電力の 3 種類のヒストグラムを示している。1 つ目はノード単位 (*node.normal.stream*)、2 つ目は BoB 単位の平均ノード電力 (*bob.normal.stream*)、3 つ目はラック単位の平均ノード電力 (*rack.normal.stream*) である。BoB 単位、およびラック単位の平均ノード電力の場合は、それに含まれるノード数倍した値を度数としている。すなわち、BoB 単位の平均が 190W である場合は、その BoB に含まれる 16 ノードが全て 190W だとしてヒストグラムを生成している。ノード単体では平均電力 190W に対し 167W(-12 %) から 235W(+24 %) まで電力が分布している。それに対して、BoB 単位では 184W(-3 %) から 201W(+6 %) と電力の広がりが狭まり、ラック単位では 188W(-1 %) から 192W(+1 %) とさらに電力の広がりが狭まっている。ただし、この結果は今回利用したラックが電力的に均一であったという事を示しているだけで、他のラックについてはその限りではない。全システムでの評価を再度行いたいと考えている。

また、図 4 には、Power API で取得できる推定電力 (*estematedpower*) のヒストグラムも一緒に示している。推定電力は、3.5 節で述べたように、電力チューニングを行うために、ノード構成などに依存しない電力を取得する機能である。そのため、推定電力からは、ノード構成によって差となる電力が除かれている。図 4 によると、推定電力のノード間のばらつきは 5W 程度と電力チューニングを行う上で十分である。また、その平均は 167W と実測ノード電力の第一の山よりも 10W ほど低い。これは、実測電力では含まれている 2 コアのアシスタントコアの電力や CMU ボード上のその他の電力、POL 自体のロスなどが除かれているためであり、妥当であると考えられる。

図 5 は、DGEMM 実行におけるノーマルモードとブーストモードの 25,344 ノードの電力分布である。本実行は 66 ラックでの実行となっている。図では 4 つのヒストグラムを重ねて表示している。青がノーマルモードのスタンバイ実測電力 (*node.normal.sdanby*)、赤がノーマルモードの DGEMM 実行時の実測電力 (*node.normal.dgemm*)、グレイがブーストモードのスタンバイ実測電力 (*node.boost.standby*)、黄色がブーストモードの DGEMM 実行時の実測電力 (*node.boost.dgemm*) をそれぞれ表している。ただし、利用した評価環境では一部の I/O ノードがブーストモード固定となっていたため、1,716 ノードではノーマルモードを指定してもブートモードで実行されている。

Stream 実行と同様に各ヒストグラムは 3 つの山があることが分かる。3 つの山の原因は Stream と同じであると考えられる。ノーマルモードの DGEMM 実行時の平均電力は 153W に対して、ブーストモードの平均電力は 173W と 13.3 %増大している。一方、性能は 9.2 %向上しており、電力効率は 3.7 %低下している。なお、本評価では、データは乱数で初期化されており、表 1 より電力は大きくなっている。また、BoB 平均、ラック平均のノード電力を確認したところ、Stream と同様にばらつきは減っており、ノードでは-16 %から 37 %のばらつきが、BoB 平均で ±4 %、ラック平均では ±1 %となっていることを確認した。

4.4　実測ノード電力の精度の評価

Power API で取得できる実測電力が正しいことを確認するため、ノードの外部の電力測定との比較を行った。ただし、電力ばらつきで用いた Stream や DGEMM は数分程度の実行時間であるのに対し、ラック単位の電力測定は 1 分単位の測定であるため、不適切であった。そこで、ここでは Graph500 での結果を紹介する。富岳における Graph500 の測定については別途報告 [7] があるため、詳細はそちらを参照してほしい。

Graph500 の事前準備として、ノーマルモードで 3 ラックを占有して実行を行った。その時の Power API を用いた実測電力の合計は 117kW であった。一方、この時のラックの電力計の合計は 126kW であった。この電力の差は、測定している電力の違いと考えられる。Power API で測定している電力は PSU(Power Supply Unit) から供給される直流電流であるのに対し、ラック側で測定しているのは PSU に供給している 200V の交流電流である。したがって、2 つ

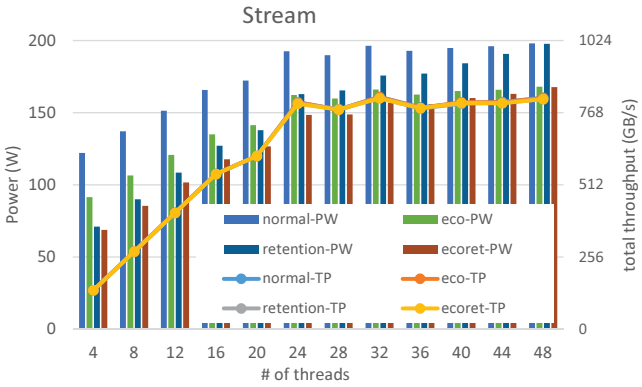

図 6　Stream におけるコアリテンションの効果

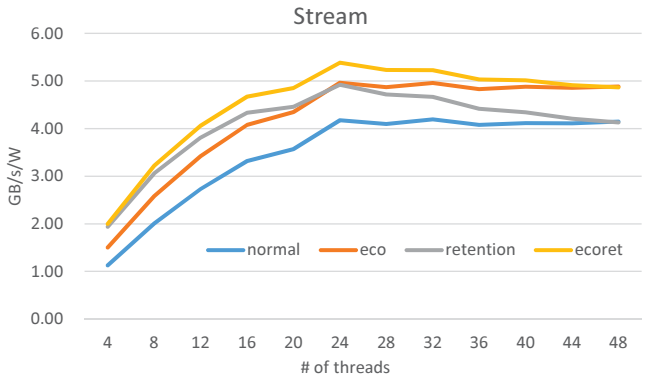

図 7　Stream における電力当たりスループット

の電力の差の 7 %は AC/DC の変換ロスおよびノード電力に含まれないラック内の制御装置などの電力と考えられる。また、PSU としては最大効率は 96 %であるが、Graph500 の電力がラック供給可能電力の半分程度と低いため、効率がやや低下していると思われる。

Green Graph500 では電力測定に Green500 相当の精度が求められている。そのため、Green Graph500 の電力測定には、高圧幹線系統での電力測定を用いた。Graph500 をブーストエコモードを用いて 240 ラックで実行した。この 240 ラックには、高圧幹線の 1 系統に属する 54 ラックを含んでいる。Power API で取得した実測電力の合計は 7.5MW に対し、高圧幹線系統の外部電力計で測定した 54 ラックの測定結果から 240 ラックの電力を推定した値は 8.3MW であった。この 10.7 %の差は、ラック間の電力ばらつき、AC/AC の変換ロス、AC/DC の変換ロスおよびラック内制御装置などの電力によるものであると考えられ、設計範囲内の値である。

4.5　リテンションの評価

コア単位のリテンションについて、Stream と DGEMM を用いて評価を行った。ただし、現在の評価環境では、スタンバイ状態から最大電力状態への電力変動幅には冷却設備の対応が可能であることが確認できているのに対し、ノードリテンション状態から最大電力状態への電力変動幅については冷却設備として対応を検討中である。そのため、一般ユーザにはリテンション制御は許可されていない。少数ノードでのみ実験するという条件の下で特別に許可をもらって評価を行ったため、1 ノードでの評価となっている。

図 6 に、全コアにリテンション可能と設定した場合に動作スレッド数を変化させて Stream を実行した結果を示す。比較として、コアリテンション不可とした場合のノーマルモード、エコモードをそれぞれ *normal*, *eco* として示し、コアリテンション可能としたノーマルモード、エコモードを *retention*, *ecoret* として示している。図の軸などは図 2 と同じであり、*normal*, *eco* はデータも一緒である。図には各実行のスループット（**-TP*）と実測電力（**-PW*）を示しているが、スループットはどのモードでも同じであり重なっている。

スレッド数 4 の時には、*retention* では 44 コアがコアリテンションとなっており 4 コアのみがアクティブとなっている。そのため、電力は *normal* の 58.2 %と低く抑えられている。*ecoret* も同様であるが、エコモードの効果が 4 コアのみとなるため *retention* との差は小さく *normal* の 56.3 %となっている。スレッド数が増えるに従い、コアリテンションのコアが減るため、リテンションの効果は減っていく。24 スレッドで性能は飽和するが、この時 *retention* の電力は *eco* とほぼ同じで *normal* の 84.6 %となるのに対し、*reteco* では動作しているコアのエコモードの効果が大きくなり *normal* の 77.1 %と *retention* に比べて 7.5 %の電力削減を示している。

同じ結果を電力当たりのスループット（GB/s/W）で表したのが図 7 である。スレッド数が増えるにしたがって電力当たりスループットは向上していく。24 スレッドで性能が飽和した後は、コアリテンション不可の場合には一定であるのに対し、コアリテンション可能の場合は、徐々に効率が低下し 48 スレッドではコアリテンション不可と同じになる。24 スレッドでは *eco* と *retention* の効率がほぼ同じになるが、これらを組み合わせた *ecoret* はさらに効率が高いことも確認できる。

このようにあるスレッド数で性能が飽和するプログラムでは、コアリテンション可能としてスレッド数を制限することにより、高い性能と低い電力を両立させることができる。

図 8 が、DGEMM の実行に対してコアリテンションを適用した評価結果である。比較としてコアリテンション不可とした場合のノーマルモード、ブーストモードをそれぞれ *normal*, *boost* として示し、コアリテンション可能としたノーマルモード、ブーストモードを *retention*, *boostret* として示している。左縦軸は実測電力、右縦軸は GFLOPS を表している。図には各実行の GFLOPS 値と電力を示しているが、GFLOPS 値はコアリテンションの設定の有無では同じであり重なっている。本評価で用いたデータは 4.2

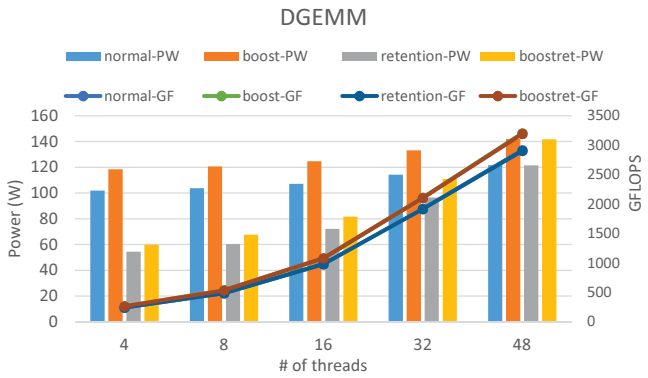

図 8　DGEMM におけるコアリテンションの効果

節と同じ初期化データを用いた結果であり、図 5 の乱数を用いた場合に比べて電力は低くなっている。

スレッド数 4 の時には、*retention* の電力は *normal* と比べておよそ半分になっている。*boostret* の電力も同様に *boost* のおよそ半分になっている。スレッド数が 32 の場合、*normal* と *boostret* を比較すると、電力がほぼ同じで、性能は 10 % *boostret* の方が高い。もし、2 冪のスレッド数などの制約がある場合には、CMG あたり 8 スレッドをブーストモードで使い、残りをコアリテンション状態とすることにより、電力は同じままで、10 %の性能向上を引き出すことが可能である。

5. まとめ

富岳共用前評価環境を用いて、富岳の電力について評価を行った。エコモードを用いることにより、メモリバンド幅律速となるようなアプリケーションでは、性能を維持したまま、電力を 15 %程度削減できることを確認した。また、ブーストモードを用いて 10 %程度の性能向上を達成できるが、電力は 17 %程度増大してしまう事を確認した。電力のノード間ばらつきについて 66 ラックのノード規模で評価を行い、I/O ノード等のノード構成の違いによって電力ばらつきが大きいことを明らかとした。しかし、BoB 単位、並びにラック単位で平均化することによりそのばらつきは小さく収まることを確認した。ただし、電力ばらつきについては、まだシステムの一部の評価にとどまっており、全システムでの評価を行う必要がある。また、コアリテンションをエコモードやブーストモードと組み合わせることにより、性能を維持しつつ電力を下げたり、同じ電力で性能を向上させることができることを確認した。

富岳は全体では 432 ラック、158,976 ノードから構成される非常に大規模なシステムである。当初から電力削減を考えて開発を進めてきており、40MW 以下という目標の電力規模に抑え、京と比べて電力効率を大幅に向上させることが出来た。しかし、消費電力が京よりも大きくなっているため、少しでも電力を減らす工夫が必要であり、今後の運用に向けてどのような電力制御を行っていくのが良いのかを検討している。本稿で述べたエコモード、ブーストモードなどの効果については、既に運用側にフィードバックしており、今後の省電力運用に活用されることを期待している。

謝辞　ノード電力の測定方法や評価結果について議論いただいた富士通株式会社の皆様に感謝いたします。本稿の一部は，文部科学省「特定先端大型研究施設運営費等補助金（次世代超高速電子計算機システムの開発・整備等）」で実施された内容に基づくものである。

参考文献

[1] Y. Yoshida, *Fujitsu High Performance CPU for the Post-K Computer*, 2018 IEEE Hot Chips 30 Symposium, 2.13, Aug. 2018.
[2] Y. Ajima, T. Kawashima, Takayuki. Okamoto, N. Shida, K. Hirai, T. Shimizu, S. Hiramoto, Yoshiro. Ikeda, T. Yoshikawa, K. Uchida and T. Inoue, *The Tofu Interconnect D*, IEEE International Conference on Cluster Computing, pp.646-654, Sep. 2018.
[3] http://top500.org/green500/lists/2019/11, *Green500 Nov. 2019*
[4] R. E. Grant, M. Levenhagen, S. L. Olivier, D. DeBonis, K. T. Pedretti and J. H. Laros III, "Standardizing Power Monitoring and Control at Exascale," Computer, vol. 49, no. 10, pp. 38-46, Oct. 2016.
[5] http://powerapi.sandia.gov/, *Sandia Power API*
[6] FUJITSU Software Technical Computing Suite V4.0L20, ジョブ運用ソフトウェア API ユーザーズガイド Power API 編, J2UL-2545-01Z0(00), Feb. 2020.
[7] 中尾, 藤澤, 児玉, 佐藤, スーパーコンピュータ「富岳」における *Graph500* ベンチマークの幅優先探索の性能評価, 情報処理学会 HPC 研究会, July. 2020.