Vol.2015-SE-188 No.9
Vol.2015-EMB-37 No.9
2015/6/5

同一命令セットヘテロジニアスマルチコアのコアペア間におけるタスクスケジューリング

高瀬 英希[1,a)]　岩田 淳[1]　高木 一義[1]　高木 直史[1]

概要：組込みリアルタイムシステムの性能向上と消費エネルギー削減を両立するプロセッサアーキテクチャとして，同一命令セットヘテロジニアスマルチコアが注目されている．本アーキテクチャでは，命令セットは同一の高性能コアと高電力効率コアを一対一に対応させたコアペアとし，これらの動作を排他的に切り替えることで，性能を落とすこと無く消費エネルギー削減が期待できる．本稿では，複数のコアペアをもつ同一命令セットヘテロジニアスマルチコアを対象とした消費エネルギー削減手法を提案する．対象とするシステムにおける各コアペアは，それぞれ独立に周波数の設定および動作コアの切替えができるものとする．提案するコアペア間のタスクスケジューリングは，設計時におけるタスクのコアペア割付け，および．実行時における一時的なタスクマイグレーションからなる．設計時におけるタスクの割付けは，整数計画法を繰り返し適用することで，全タスクが平均の負荷で実行されるときの消費エネルギーがリアルタイム性を保証しつつ最小となるものを求める．さらに，実行時には，割り付けられたタスク群の負荷がコアペア間で平準化されるよう一時的なタスクマイグレーションを行い，消費エネルギーを抑制する．評価の結果，提案手法は，これまでに我々が提案してきたコアペア内の DVFS 手法と組み合わせることにより，平均で 42.5%，最大で 59.1%の消費エネルギーを削減できた．

1. はじめに

近年の組込みシステムでは，より高い性能が求められるとともに，消費エネルギーを最小化することが重要な課題となっている．この解決策のひとつとなるプロセッサアーキテクチャとして，同一命令セットヘテロジニアスマルチコアが注目されている．本アーキテクチャは，異なる性能と電力効率に基づいて設計された複数のコアを持つ．各コアの命令セットは同一であり，その差異を意識せずに動作コアを切り替えることができる．一般に，高性能コアと高電力効率コアを一対一に対応させたコアペアとして，各コアペアでは動作コアを排他的に切り替えてタスクが実行される．負荷の大きい時は高性能コアを，そうでない時は高電力効率コアを動作させることで，性能を落とすこと無く消費エネルギーの最小化を実現する [1]．今後，同一命令セットヘテロジニアスマルチコアを採用した組込みシステムの普及が見込まれ，高性能化と低消費エネルギー化の両立に貢献することが期待される．

同一命令セットヘテロジニアスマルチコアにおいて消費エネルギーを最小化するためには，動作コアの選択や動作周波数の設定，さらにはシステムの処理単位であるタスクの各コアペアへの割付けといった，適切なタスクスケジューリングが重要となる．ただし，リアルタイム性が要求される組込みシステムでは，デッドライン制約を保証することも考慮しなければならない．ここで，デッドライン制約とは，タスクの実行をそれぞれの所定の時刻であるデッドラインまでに完了しなければならないことを指す．つまり，同一命令セットヘテロジニアスマルチコアを採用した組込みリアルタイムシステムでは，デッドライン制約を保証した上での適切なタスクスケジューリングによって，高い消費エネルギー削減効果を実現することが望まれる．

我々は，1 つずつの高性能コアと高電力コアで構成された同一命令セットヘテロジニアスマルチコアのコアペア内における動的電圧・周波数制御（Dynamic Voltage and Frequency Scaling，DVFS）手法を提案してきた [2]．DVFS とは，コアの供給電圧および周波数を適切に制御する技術である [3]．文献 [2] の手法では，まず，コアペアを構成する各コアの性能比および周波数設定値から定義される正規化性能をもつコアペアテーブルを生成する．これを用いることで，シングルプロセッサ向けの DVFS によってコアペア内の動作コアの切替えが実現できる．さらに，実行時のタスク切替えの際に，タスクの平均実行時間およびデッドライン制約を考慮して 2 種類の正規化性能を算出し，より高いものから周波数設定値を選択して DVFS を適用する．

[1] 京都大学 大学院情報学研究科
[a)] takase@i.kyoto-u.ac.jp

Vol.2015-SE-188 No.9
Vol.2015-EMB-37 No.9
2015/6/5

本研究では，複数のコアペアからなる同一命令セットヘテロジニアスマルチコアを採用した組込みリアルタイムシステムにおいて，要求される性能を保証した上での消費エネルギーの最小化を目指す．研究対象のシステムでは，高性能コアと高電力効率コアを一対一に対応させたコアペアとして扱う．各コアペアは，それぞれ独立に周波数の設定および動作コアの切替えができるものとする．このため，DVFS および動作コアの選択に加え，各コアペアへの適切なタスク割付けも重要となる．目的達成のため，本研究では，コアペア内の DVFS 手法に加えたコアペア間のタスクスケジューリングによって，更なる消費エネルギーの削減を実現する手法を提案する．

提案するコアペア間のタスクスケジューリングでは，タスクの平均の負荷を実行できるコアペアの性能に注目する．文献 [4] によれば，一般のマルチプロセッサでは，負荷が均等となるようにタスクを割り付けた場合に消費エネルギーが最小となる．これに基づき，提案手法では，システム設計時に，各コアペアにある正規化性能が与えられた時に消費エネルギー最小となるタスク割付けを求める問題を定式化した整数計画法を繰り返し適用する．これにより，各コアペアが全タスクの平均負荷においてリアルタイム性を保証しつつ消費エネルギーが最小となるタスク割付けを探索する．このとき，各コアペアに割り付けられたタスク群の平均の負荷が実行できる最も低い正規化性能も求めておく．実行時には，あるコアペア内で DVFS 手法を適用して算出された正規化性能が，設計時に求めておいた値を超える場合，一時的なタスクマイグレーションを試みる．タスクを選定して他のコアペアに一時的に割り付けることで，コアペア間の負荷を平準化して消費エネルギーを抑制する．提案手法は，コアペア間の負荷を平準化することで，消費エネルギー削減効果の更なる向上を実現する．

2. 準備

2.1 同一命令セットヘテロジニアスマルチコア

同一命令セットヘテロジニアスマルチコアとは，同じ命令セットを持ち，性能が異なる複数のプロセッサコアを持つアーキテクチャである．本アーキテクチャでは，命令セットの差異を意識することなくタスクの割付けを変更できる．代表例として，ARM 社の big.LITTLE アーキテクチャ [5] や NVIDIA 社の Variable SMP[6] が挙げられる．前者については，商用プロセッサとして Samsung 社の Exynos 5 Octa やルネサスエレクトロニクス社の MP6530 がある．これらは，性能が高いが消費電力も大きい big コアとして Cortex-A15，性能は低いが性能対電力効率が高い LITTLE コアとして Cortex-A7 の 2 種類のコアを持つ．文献 [7] は，オンチップメモリを共有した高性能コアと高電力効率コアで構成される同一命令セットヘテロジニアスマルチコアの回路設計および実測による評価を示している．

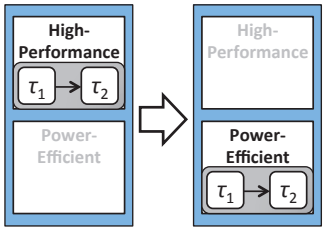

図 1　動作コアの切替え

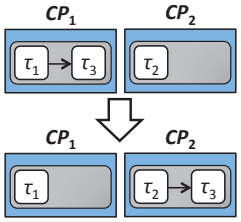

図 2　タスクマイグレーション

同一命令セットヘテロジニアスマルチコアでは，高性能コアおよび高電力効率コアを対応付けてコアペアとして，動作コアの切替えが適用できる．コアペアに割り付けられたタスク群は，その状況に応じて，コアペア内の高性能コアと高電力効率コアのいずれかで実行される．**図 1** は，高性能コアから高電力効率コアに動作コアを切り替える例を示している．さらに，あるタスクの割付けを実行時に変更させることをタスクマイグレーションと呼ぶ．**図 2** は，同一命令セットヘテロジニアスマルチコアにおけるコアペアの CP_1 に割り付けられていたタスク τ_3 を CP_2 にマイグレーションしている例である．このように同一命令セットヘテロジニアスマルチコアでは，適切なタスクスケジューリングによって高い性能対電力効率を実現できる．ただし，組込みリアルタイムシステムにおいては，動作コアの切替えおよびタスクマイグレーションは，デッドライン制約が保証されるように行う必要がある．

2.2 コアペア内の DVFS 手法

DVFS 手法を適用できる組込みシステムでは，コアの周波数を適切に設定することが消費エネルギー最小化を達成するための重要な課題となる．CMOS 回路の消費エネルギーは，周波数の 2 乗に比例すると近似できる [8]．ゆえに，DVFS によってコアの供給電圧と周波数を下げてタスクを実行することで，消費エネルギーが削減できる．

我々は，これまでに単一のコアペアを持つ同一命令セットヘテロジニアスマルチコアのコアペア内における DVFS 技術を研究してきた．文献 [2] では，コアペア内のタスクスケジューリングは Earliest Deadline First（EDF）[9] を採用するものを対象として，コアペアテーブルおよびコアペア内に適用可能な DVFS 手法を提案した．

コアペアテーブルとは，同一命令セットヘテロジニアスマルチコアを構成する高性能コアおよび高電力効率コアについて，それぞれの正規化性能 NF_n，性能対消費電力 PW_n，コアペア内の動作コアおよびその動作コアにおける周波数の識別子を表す情報をまとめたものである．NF_n は，高性能コアの最高周波数における性能を 1 としたときの，コアペア内の各コアの各周波数設定値における性能を正規化した値として定義される．PW_n は，それぞれの正規化性能に対する消費電力として定義される．なお，コアペアテーブルの生成時には，各行は正規化性能で降順に並

Vol.2015-SE-188 No.9
Vol.2015-EMB-37 No.9
2015/6/5

表 1　コアペアテーブルの例

NF_n	PW_n	コアの種類	周波数
1.00	2300	0	0
0.80	1500	0	1
0.55	1200	0	2
0.30	1000	1	0
0.15	750	1	1

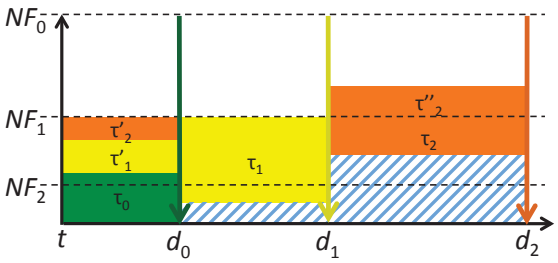

図 3　LBAR の適用例

べられ，隣り合う行で NF が低くかつ PW が大きいものはテーブルから除外する．

表 1 に，コアペアテーブルの例を示す．正規化性能が定義されたコアペアテーブルを利用することで，シングルプロセッサ向けの DVFS によってコアペア内の動作コアの切替えが実現できる．DVFS 手法で算出される周波数は，高性能コアの最高周波数の性能で正規化すれば，正規化性能と同等に扱える．そして，正規化された値以上で，かつ，コアペアテーブルにある最低の NF_n となる行を選択する．その行の値から，動作コアの種類および周波数を設定する．

DVFS 手法としては，実行すべきタスクが切り替わる際に，2 種類の算出方法から得られた要求される正規化性能のうち，より高いもののコアペアテーブルに対応する動作コアおよび周波数設定値を選択する手法を提案した．提案する 2 種類の算出方法では，タスクの平均実行時間およびデッドライン制約を考慮する．コアペア内において要求される正規化性能を平均実行時間によって平準化することにより，デッドライン制約を保証した上での高い消費エネルギー削減効果を実現する．

1 つ目は，各タスクの実行時間の平均値と最悪値の比を用いて要求される正規化性能を算出する Load Balancing with Average Ratio（LBAR）である．LBAR では，隣接するデッドライン間の実行容量（ある区間における時間と最高の正規化性能の積）に対して，各タスクの平均の仕事量（ある区間におけるタスクの平均実行時間と設定される正規化性能の積）を予約していく．これを優先度の高いタスクから繰り返すことで，全タスクの平均の負荷を平準化できる正規化性能が算出される．**図 3** に，3 タスクでの LBAR の適用例を示す．d_1，d_2 および d_3 は，タスク τ_1，τ_2 および τ_3 それぞれの絶対デッドライン時刻，NF_1，NF_2 および NF_3 は，コアペアテーブルの正規化性能をあらわす．

2 つ目は，デッドライン制約を保証できる最低の正規化性能を厳密に算出する full look ahead EDF（flaEDF）である．flaEDF は，仕事量が予約されない部分が生じないように隣接デッドライン間に最悪の仕事量を順に予約し，直近のデッドラインまでに要求される最低の正規化性能を算出する．算出された正規化性能は最悪時の負荷を用いるため，デッドライン制約を保証することができる．**図 4** に，3 タスクでの flaEDF の適用例を示す．

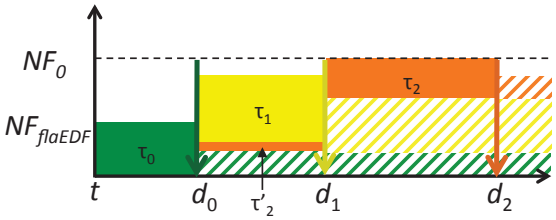

図 4　flaEDF の適用例

以上のコアペア内における DVFS 手法は，シングルプロセッサ向けの既存手法である look ahead EDF（laEDF）[10] にある問題点を解消しており，より高い消費エネルギー削減効果が得られる．文献 [2] における評価実験では，このコアペア内における DVFS 手法によって，単一のコアペアをもつ同一命令セットヘテロジニアスマルチコアにおいて最大で 59.2 %の消費エネルギーの削減が達成できた．

3. コアペア間のタスクスケジューリング

本章では，複数のコアペアからなる同一命令セットヘテロジニアスマルチコアシステムにおける消費エネルギー最小化のためのタスクスケジューリング手法を提案する．

提案手法は，まず，システム設計時に，全タスクの平均の負荷を用いて，デッドライン制約を保証した上で消費エネルギー最小となる各コアペアへのタスク割付けを決定する．タスク割付けは，各コアペアにある正規化性能が与えられたときにシステム全体で消費エネルギーが最小となるタスク割付けを求める問題を整数計画法に定式化し，これを繰り返し適用することによって探索する．このとき，決定されたタスク割付けにおいて要求される最低の正規化性能を，実行時に用いる閾値として記憶しておく．

実行時には，コアペアの負荷が変動したときに一時的なタスクマイグレーションを試みる．具体的には，あるコアペア内で DVFS 手法を適用して算出された正規化性能が，設計時に求められた閾値を超える場合にタスクマイグレーションの実行を判定する．負荷のより小さいタスクを選定して他のコアペアに一時的に割り付け，コアペア間の負荷を平準化してコアペア内の負荷の変動を抑えることで，システム全体の消費エネルギーの抑制を狙う．

本研究で提案するコアペア間のタスクスケジューリング手法は，これまで我々が提案してきたコアペア内の DVFS 手法に加えて，同一命令セットヘテロジニアスマルチコアを採用した組込みリアルタイムシステムの消費エネルギー

削減効果の更なる向上を実現する．

3.1 対象とするシステムモデル

本研究の対象とするシステムモデルについて説明する．対象システムは，各々 M 個の高性能コア $Core_{HP,m}$ および高電力効率コア $Core_{PE,m}$ で構成されるコアペア CP_m（$m = 0, 1, ..., M-1$）で構成される．各コアの周波数は有限個の離散値が設定でき，それぞれ独立して周波数およびそれに対応した供給電圧が設定できるものとする．各コアペアは同等の性能および消費電力の構成をとり，各コアペアで同等の正規化性能 NF_n および性能対消費電力 PW_n（ともに $n = 0, 1, ..., N-1$）を持つ．タスクの実行時間は，割り付けられた動作コアの周波数に対応した正規化性能に反比例するとする．また，各コアのアイドル時における静的電力，動作コアの切替え，および，DVFS の実行にかかる消費エネルギーは無視できるものととする．

タスクセット T は，I 個のタスク τ_i（$i = 0, 1, ..., I-1$）から構成される．各コアペアでは，それぞれ独立して，動的優先度ベースのリアルタイムスケジューリングである EDF [9] に従って割り付けられたタスク群が実行される．すなわち，全タスクは周期タスクであり，τ_i のリリース周期 P_i と相対デッドライン D_i は等しい．絶対デッドライン時刻 d_i は，τ_i のリリース毎に D_i を加算した値に更新される．各タスクは独立して動作し，あるタスクのリリース時に最も d_i が小さい τ_i が最高優先度となる．添字 i は，$d_0 \leq d_1 \leq \cdots \leq d_{i+1}$ を満たす．最悪実行時間 C_i は，高性能コアの最高周波数で実行する場合の最長の実行時間であり，既知とする．各タスクが高性能コアの最高周波数で実行される場合の平均実行時間も既知とし，平均実行時間と最悪実行時間の比を AR_i（$0 < AR_i \leq 1$）とする．τ_i の負荷は $U_i = \frac{C_i}{P_i}$，タスクセット T の全体の負荷 U は $U = \sum_i U_i$ と定義される．

3.2 整数計画法の繰り返しによるタスク割付けの決定

まず，あるコアペア CP_m に割り付けられた I_m 個のタスク $\tau_{i_m} \in T_m$ の平均の負荷を実行するために必要となる最低の正規化性能 ANF_{n_m} を，以下のように定義する．

$$ANF_{n_m} = \min \left\{ NF_n \mid NF_n \geq \sum_{i_m=0}^{I_m-1} AR_{i_m} \cdot U_{i_m} \right\} \tag{1}$$

つまり，全タスクの平均の負荷が平準化されて実行された場合，ANF_{n_m} が CP_m の正規化性能として要求される．

ここで，文献 [2] における LBAR と flaEDF を組み合わせたコアペア内の DVFS 手法は，タスクの仕事量の変動に対応して動的に平準化された正規化性能を算出する．タスクの負荷に変動がなく平均の実行時間がかかる場合，コアペア CP_m では DVFS 手法により ANF_{n_m} に近い正規化性能が常に算出されることになる．

以上の議論，つまり，コアペア内の DVFS 手法と ANF_{n_m} の関係を活かし，システム設計時には，全てのコアペアで ANF_{n_m} が最低となるタスク割付けとすることが，システム全体の消費エネルギー最小化に繋がるといえる．これに基づき，各コアペアへのタスク割付けは，全タスクの平均の負荷を考慮して消費エネルギーを最小化し，かつ，デッドライン制約を保証できるものとする．ただし，各コアペアでの最適なタスク割付けと ANF_{n_m} を同時に決定する問題は，整数計画法によって定式化することはできない．そこで，NF_{n_m} は入力として与えて最適なタスク割付けを求める問題を，整数計画法に定式化する．さらに，この最適解が得られるかの結果に応じて NF_{n_m} を増減させて整数計画法を繰り返し適用する．この繰り返しにより，デッドライン制約を保証した上で消費エネルギーが最小となるタスク割付けおよび割り付けられたタスク群を実行できる最適な正規化性能 ANF_{n_m} を求める．

3.2.1 整数計画法への定式化

整数計画法によって定式化する問題は，各コアペア CP_m がそれぞれ与えられた正規化性能 NF_{n_m} で常に実行されるとき，時刻 0 からハイパーピリオド $HyperPeriod$[*1] における全コアペアの合計の消費エネルギー E_{total} が最小となるタスク群 $\tau_{i_m} \in T_m$ の割付けを求めるものとなる．

まず，0-1 変数 $x_{i,m}$ を以下のように定義する．

$$x_{i,m} = \begin{cases} 1 & (\tau_i \text{が} CP_m \text{に割り付けられる}) \\ 0 & (otherwise) \end{cases} \tag{2}$$

すなわち，$x_{i_m} = 1$ のとき $\tau_{i_m} \in T_m$ となる．

制約条件として，まず 1 つめに，全てのタスクは，1 個のコアペアのみに割り付けられる．つまり，以下を満たす．

$$\forall i, \sum_{m=0}^{M-1} x_{i,m} = 1 \tag{3}$$

2 つめに，各コアペアに割り付けられたタスク群は，EDF によってスケジュール可能でなければならない．つまり，以下を満たす．

$$\forall m, \sum_{i=0}^{I-1} x_{i,m} \cdot U_i \leq 1 \tag{4}$$

3 つめに，各コアペアに与えられた正規化性能によって割り付けられたタスク群の平均負荷が EDF によってスケジュール可能でなければならない．つまり，以下を満たす．

$$\forall m, \sum_{i=0}^{I-1} x_{i,m} \cdot AR_i \cdot U_i \leq NF_{n_m} \tag{5}$$

目的関数は，以下のように定義される．

[*1] 全タスクの起動周期の最小公倍数となる時間

$$E_{total} = \sum_{m=0}^{M-1} E_m \cdot HyperPeriod \quad (6)$$

$$E_m = \sum_{i=0}^{I-1} x_{i,m} \cdot AR_i \cdot U_i \cdot PW_{n_m} \quad (7)$$

ここで，PW_{n_m} は，CP_m に与えられる正規化性能 NF_{m_n} に対応した性能対消費電力である．

整数計画法によって E_{total} を最小化する $x_{i,m}$ が，各コアペアに NF_{n_m} を与えたときにシステム全体の消費エネルギーが最小となるタスク割付けとして求められる．

3.2.2 整数計画法の繰り返しによるタスク割付けの探索

3.2.1 節で定式化した整数計画法を繰り返し適用することにより，消費エネルギー最小となる各コアペアへのタスク割付けを探索する．繰り返しの際には，前回の整数計画法で求められた解から正規化性能を増減させる．これによって，消費エネルギー最小となる各コアペアの正規化性能およびタスク割付けが求められる．

初期値として整数計画法に与える各コアペアの正規化性能について考える．文献 [4] より，各コアペアの負荷が均等になる割付けである場合，消費エネルギーが最小となる．つまり，各コアペアの ANF_{n_m} が全て等しくなる場合にシステム全体の消費エネルギーが最小化される．ただし，一般にタスクは有限個であり，各タスクの負荷は等しくないため，各コアペアに負荷を等しく割り付けることはできない．このため，各コアペアがそれぞれ ANF_{n_m} に近い値になるようなタスク割付けによって，消費エネルギーが最小になると考えられる．以上の議論から，整数計画法に与える正規化性能の初期値 NF_{n_m} は，全タスクの平均負荷の総和をコアペア数で除算した負荷 U_{ave} を実行できる最小のものとする．U_{ave} は，以下の式で求められる．

$$U_{ave} = \frac{\sum_{i=0}^{I-1} AR_i \cdot U_i}{M} \quad (8)$$

これを利用して，初期値 NF_{n_m} は以下によって求められる．

$$NF_{n_m} = \min\{NF_n \mid NF_n \geq U_{ave}\} \quad (9)$$

この初期値を全コアペアに与え，整数計画法の最適解を求める．解が得られない場合は，全てのコアペアの NF_{n_m} を NF_{n_m-1} に更新して再び整数計画法を解く．つまり，与える正規化性能を全コアペアに対して一段階上げる．これを，いったん整数計画法の解が求められるまで繰り返す．

初期値によって解が得られた場合，または，正規化性能を上げていき解が得られた場合は，まず，この解をタスク割付けおよび ANF_{n_m} として記録する．次に，いずれかのコアペアに与える正規化性能を NF_{n_m+1} に一段階引き下げて解が得られるかを調べる．解が得られる場合は，記録していた解を更新し，NF_{n_m+1} とするコアペアを 1 つ増やして再び整数計画法を解く．これを解が得られなくなるまで繰り返す．解が得られなくなったところで整数計画法の適用を終了し，最後に記録されたいた解を，探索されたタスク割付けおよび ANF_{n_m} として得る．

```
Input: τ_i ∈ T, AR_i, N, NF_n
Output: タスク割付け mapping，各コアペアの ANF_{n_m}
 1: U_ave ⇐ (Σ_{i=0}^{I-1} AR_i·U_i) / M
 2: n' ⇐ min{n | NF_n ≥ U_ave}
 3: for m = 0 to M − 1 do
 4:     NF_{n_m} ⇐ NF_{n'}
 5: end for
 6: possible ⇐ false
 7: while possible = false do
 8:     (possible, mapping) ⇐ ILP(T, CP, NF_{n_m})
 9:     if possible = false then
10:         n' ⇐ n' − 1
11:         for m = 0 to M − 1 do
12:             NF_{n_m} ⇐ NF_{n'}
13:         end for
14:     else
15:         tmp_mapping ⇐ mapping
16:         for m = 0 to M − 1 do
17:             (possible, mapping) ⇐ ILP(T, CP, NF_{n_m})
18:             if possible = false then
19:                 possible ⇐ true
20:                 break
21:             else
22:                 NF_{n_m} ⇐ NF_{n'+1}
23:                 tmp_mapping ⇐ mapping
24:             end if
25:         end for
26:     end if
27: end while
28: mapping ⇐ tmp_mapping
29: for m = 0 to M − 1 do
30:     ANF_{n_m} ⇐ NF_{n_m}
31: end for
```

図 5 整数計画法の繰り返しによるタスク割付けの探索

図 5 に，整数計画法の繰り返しによって消費エネルギー最小となる各コアペアへのタスク割付けおよび ANF_{n_m} を探索する手法の手順を示す．ここで，5 行目および 9 行目の関数 $ILP(T, CP, NF_{n_m})$ は，3.2.1 節における整数計画法を適用していることを意味している．この関数は，整数計画法により解が得られる場合，$possible$ には $true$ を，$mapping$ には最適なタスク割付けを返す．以上の探索により，各コアペアができる限り均等な ANF_{n_m} をとり，かつ，その ANF_{n_m} において消費エネルギー最小となるようなコアペア間のタスク割付けを得ることができる．

整数計画法の繰り返し回数について議論する．整数計画法に与える初期値 NF_{n_m} について，全コアペアが NF_{n_m} 未満で動作する場合は，全タスクの平均負荷を実行することはできない．これは，以下の不等式を満たすためである．

$$NF_{n_m+1} \cdot M < U_{ave} \cdot M \leq NF_{n_m} \cdot M \quad (10)$$

この不等式から，1 つずつ NF_{n_m} を引き下げていく処理

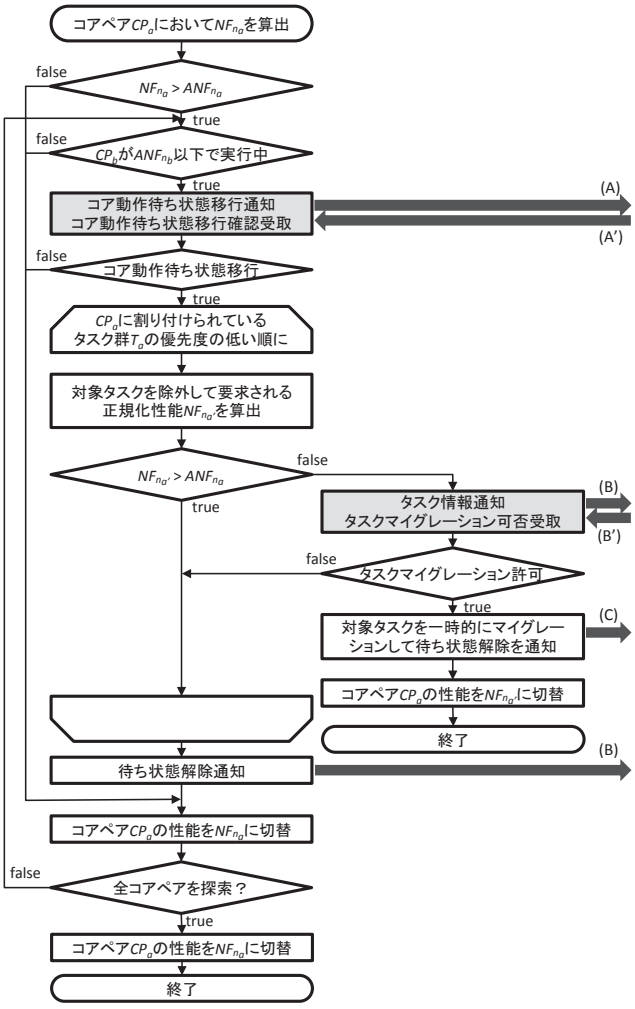

図 6 タスクマイグレーション元コアペア CP_a のフローチャート

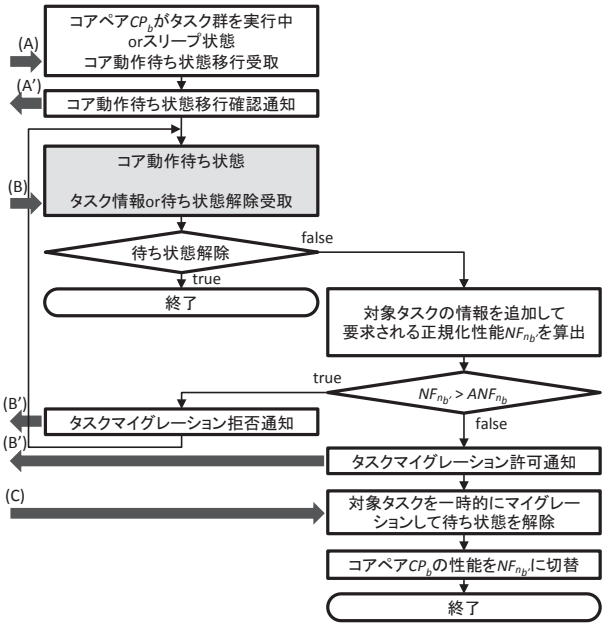

図 7 タスクマイグレーション先コアペア CP_b のフローチャート

（図 5 の 16–24 行目）は，最大でも $M-1$ 回の繰り返しとなることがわかる．一般に，厳しい資源制約を満たす必要のある組込みシステムの設計には，開発時間を掛けて最適化することが許容される．つまり，整数計画法を繰り返してコアペア間の最適なタスク割付けを決定する本手法は，組込みシステムの設計時には妥当なものであると考える．

3.3 一時的なタスクマイグレーション

コアペア間のタスクスケジューリングにおいて，実行時には，設計時に求めた ANF_{n_m} を閾値として利用する．以降，タスクのマイグレーション元のコアペアを CP_a $(a \in 0, 1, ..., M-1)$，マイグレーション先のコアペア CP_b $(b \in 0, 1, ..., M-1$，ただし $a \neq b)$ とする．

あるコアペア CP_a においてタスクの負荷が変動して要求される正規化性能が ANF_{n_a} を超える場合，あるタスク τ_i を他のコアペア CP_b に一時的にマイグレーションすることを試みる．ここで，一時的なマイグレーションとは，マイグレーションされたタスク τ_i が実行完了するまでの期間のみ割付けを変更することを意味する．つまり，CP_b において τ_i の実行が完了したら，τ_i の割付けは CP_a に戻される．ただし，タスクを一時的に引き受けることによって CP_b の正規化性能が ANF_{n_b} を超えてしまう場合は，マイグレーションは行わない．以上のタスクスケジューリング手法により，要求される正規化性能をできる限り ANF_{n_m} 以下に抑え，タスクの負荷をコアペア間で平準化してシステム全体の消費エネルギーを抑制する．

図 6 および**図 7** に，提案するコアペア間のタスクスケジューリング手法による，マイグレーション元のコアペア CP_a およびマイグレーション先のコアペア CP_b のフローチャートを示す．図中の太線矢印はコアペア間での情報のやり取りを，フローチャートの灰色部分は相手コアペアの処理待ち状態であることをあらわす．マイグレーション先のコアペアは，正規化性能の低いものから順に選択する．

一時的なタスクマイグレーションの実行の判定は，コアペア内の DVFS 手法によってコアペア CP_a に要求される正規化性能 NF_{n_a} が算出される時である．その際，NF_{n_a} が ANF_{n_a} を超え，かつ，要求されている正規化性能が最小のコアペア CP_b について，NF_{n_b} が ANF_{n_b} 以下であるか，CP_b がスリープ状態である時に限り，タスクマイグレーションの実行を試みる．これを満たす時，まず，CP_a から CP_b を動作待ち状態に移行させる通知（図中の (A)）を行って CP_a の処理を中断する．CP_b が移行したことが (A') の通知によって確認されたら，CP_a は処理を再開する．

その後，CP_a に割り付けられているタスクの優先度の低いものから順に，そのタスク τ_{i_a} を T_a から除外した場合に要求される $NF_{n'_a}$ を算出する．これが ANF_{n_a} を下回る，すなわち，τ_{i_a} を CP_b にマイグレーションすれば CP_a が ANF_{n_a} 以下で実行し続けられる場合に，τ_{i_a} の情報を CP_b に通知する（図中の (B)）．CP_b では，マイグレーション候補である τ_{i_a} の情報を追加した場合に要求される正規化性

能 $NF_{n'_b}$ を算出する．$NF_{n'_b}$ が ANF_{n_b} を超えない場合は，τ_{i_a} のマイグレーションを CP_b に一時的に行えることを意味する．CP_b は，この判定結果を CP_a に通知する（図中の (B')）．τ_{i_a} のマイグレーションが行えない場合は，CP_a はより優先度の高いタスクを候補として判定を繰り返す．

マイグレーションを一時的に行えるタスクがある場合は，対象とするタスクの情報を通知して CP_b の待ち状態を解除する（図中の (C)）．その後，CP_a は性能を $NF_{n'_a}$ に，CP_b は性能を $NF_{n'_b}$ にそれぞれ切り替え，τ_{i_a} の実行が完了するまでスケジューリングを続ける．CP_a に割り付けられている全てのタスクが CP_b にはマイグレーションできないと判定された場合は，より正規化性能の高いコアペアをタスクマイグレーション先の候補として処理を繰り返す．全てのコアペアに対してどのタスクもマイグレーションできない場合は，タスクマイグレーションは行わず，CP_a の性能をはじめに算出された NF_{n_a} へと切り替える．

4. 評価

4.1 実験環境

提案手法の有効性を評価するため，自作のタスクスケジューリングシミュレータ上で実験を行った．本シミュレータは，与えられたパラメータにより生成されたランダムタスクセットに対して我々の提案手法を適用し，ハイパーピリオドまでの消費エネルギーを算出する．

ランダムタスクセットは，負荷 U および平均と最悪時の実行時間の比 AR をパラメータとして自動生成した．各タスクの起動周期は，2ms 間隔の 2–100ms の一様分布で決定し，ハイパーピリオドが 10s 以下となるタスクを選んだ．タスク数 I はコアペア数 M に比例して $5M$ 個とし，同じ U と AR で生成された 100 個のランダムタスクセットを実行した平均値を評価した．

コアペアテーブルの生成に必要な各コアの IPC，設定できる周波数および消費電力のパラメータは，Cortex-A15 および A7 コアをそれぞれ 4 個もつ Exynos 5422 を搭載する ODROID-XU3 [11] から取得した．パラメータには，ODROID-XU3 上で動作させた Ubuntu 14.04 にて姫野ベンチマーク [12] を実行して計測した値を用いた．

評価対象は，(1) コアペア間のタスク割付けは WFD [4]，コアペア内の DVFS は laEDF [10]，(2) タスク割付けは WFD [4]，DVFS は文献 [2] の提案手法（WFD + Proposed DVFS），(3) タスク割付けは設計時の整数計画法の繰り返し（3.2 節）のみ，DVFS は提案手法（Proposed ILP + DVFS），(4) タスク割付けは WFD に実行時の一時的なタスクマイグレーション（3.3 節）も適用するもの，DVFS は提案手法（WFD + Proposed DVFS + Migration），および，(5) 提案手法（Proposed ILP + DVFS + Migration）の各手法である．なお，WFD（Worst-Fit Decreasing）[4] とは，設計時に全タスクの最悪の負荷を各コアペアへ均等に分配

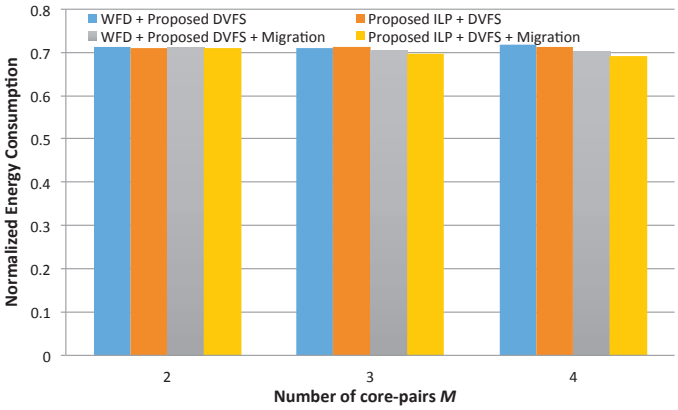

図 8 コアペア数 M を変動させた場合の評価結果
（$U = 0.7M$，$AR = 0.3$）

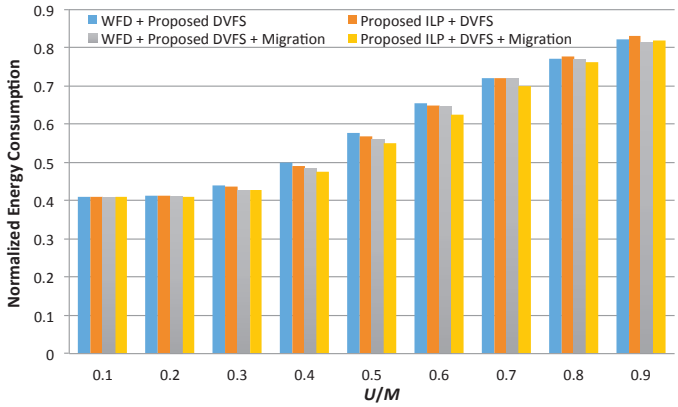

図 9 コアペア数 4 で U/M を変動させた場合の評価結果
（AR は 0.1 から 0.5 までの一様分布）

してタスク割付けを決定する手法である．評価では，提案手法の有効性を比較するため，(1) の消費エネルギーで結果を正規化した．なお，正規化性能の算出処理や動作コアの切替え等，タスクスケジューリングに掛かる時間および消費エネルギーのオーバヘッドは無視した．

4.2 評価結果

まず，各タスクの AR_i は AR と正規分布から，最悪実行時間および実際の実行時間は U および AR_i と正規分布から生成されるランダムタスクセットによって評価を行った．U には $0.1M$ から $0.9M$ まで，AR には 0.1 から 1.0 までのそれぞれ 0.1 刻みを入力パラメータとして，計 90 通りの組合せを評価した．紙面の都合上，提案手法の効果の特徴が表れている入力パラメータの結果のみを示す．**図 8** は，$U = 0.7M$，$AR = 0.3$，コアペア数 $M = 2, 3, 4$ とした場合の実験結果である．$M = 4$ のとき，提案手法は，WFD によるタスク割付けのみの (2) と比較して最も消費エネルギーが削減され，その効果は 3.6%であった．また，コアペア数が多いほど消費エネルギー削減効果は高くなった．

次に，平均実行時間と最悪実行時間の比 AR が一様分布によって生成されるタスクセットによって評価を行った．

タスクセット生成のためのパラメータは U のみとなる．各タスクの AR_i は，0.1 から 0.5 まで 0.1 刻みの一様分布を利用して決定した．最悪実行時間および実際の実行時間は，U および AR_i と正規分布を利用して決定した．**図 9** は，$M = 4$ として U を変動させた場合の実験結果である．提案手法は，(1) と比較して平均で 42.5%，最大で 59.1%の消費エネルギーを削減できた．コアペア間のタスク割付けに関しては，他のものとほぼ同等，または，最も優れた消費エネルギー削減効果があった．特に $U/M = 0.5$ のときに (2) と比較して最も消費エネルギーが削減され，その効果は 4.6%であった．ただし U/M が大きい場合には，提案手法は WFD によるタスク割付けよりも劣る結果となった．

4.3 考察

提案手法は，コアペア間のタスク割付けは WFD，コアペア内の DVFS は laEDF のものと比較して，消費エネルギーを大きく削減できることが確認された．特に，図 9 では，U/M が小さい時に高い消費エネルギー削減効果があった．このことから，提案するコアペア間のタスクスケジューリング手法は，これまでに我々が提案したコアペア内の DVFS 手法と組み合わせることで，複数のコアペアを持つ同一命令セットヘテロジニアスマルチコアにおけるより高い消費エネルギー削減効果を実現できた．

提案手法のうちの一時的なタスクマイグレーションは，U，AR および M がどのような値でも，タスクマイグレーションを適用しない (2) および (3) と比較して消費エネルギーを削減することができた．このことから，提案するタスクマイグレーション手法は，同一命令セットヘテロジニアスマルチコアのコアペア間での消費エネルギーを削減できる有効な手段といえる．

提案手法のうち，設計時のタスク割付けについては，WFD によるものと比較して優れている場合とそうでない場合に分かれた．図 9 のようにタスク毎の AR_i が一様分布となるタスクセットでは，提案するタスク割付け手法の消費エネルギー削減効果が高くなる場合が多かった．いっぽうで，紙面の都合上割愛したものの，AR_i が正規分布で決定されるタスクセットについては，WFD より劣る場合が多かった．これは，正規分布によるタスクセットでは，実行時間の変動および負荷の偏りが小さく，最悪実行時間を用いる WFD によるタスク割付けのほうが適していたためと考えられる．ただし，AR_i が一様分布で各タスクの実行時間に相関関係が無い場合，WFD によるタスク割付けでは実行時間の変動および負荷の偏りに対処できなくなる．このようなタスクセットでは，提案するコアペア間のタスクスケジューリング手法の有効性が高くなる．

5. おわりに

本研究では，複数のコアペアをもつ同一命令セットヘテロジニアスマルチコアにおけるタスクスケジューリング手法を提案した．まず，設計時において，デッドライン制約を保証しつつ消費エネルギー最小となる各コアペアへのタスク割付けを決定する手法を提案した．各コアペアへのタスク割付けは，あらかじめ各コアペアに要求される正規化性能を与えた下で消費エネルギー最小となるタスク割付けの探索を整数計画法として定式化し，これを繰り返し適用することによって決定される．実行時には，コアペア間の負荷が平準化されるよう，一時的なタスクマイグレーションを試みる．これにより，タスクの負荷の変動による消費エネルギーの増大を抑止する．

本研究におけるコアペア間のタスクスケジューリング手法は，我々がこれまでに提案してきたコアペア内の DVFS 手法と組み合わせることで，より高い消費エネルギー削減効果を実現することができる．評価の結果，提案手法は，最大で 59.1%の消費エネルギー削減が達成できた．今後の方針として，提案手法の実行にかかるオーバヘッドを考慮することや，実機上での実用アプリケーションを用いた有効性の評価が挙げられる．

謝辞 本研究は，JSPS 科研費 26870303 の助成による．

参考文献

[1] Kumar, R., et al.: Single-ISA Heterogeneous Multi-Core Architectures: The Potential for Processor Power Reduction, *Proc. of MICRO*, pp. 81–92 (2003).
[2] 岩田, 他: 同一命令セットヘテロジニアスマルチコアのための消費エネルギーを削減するタスクスケジューリング, 情報処理学会研究報告, Vol. 2015-EMB-36, No. 33 (2015).
[3] Hu, S. X., et al.: Fundamental of Power-aware Scheduling, *Designing Embedded Processors: A Low Power Perspective*, Springer (2007).
[4] Aydin, H. and Yang, Q.: Energy-Aware Partitioning for Multiprocessor Real-Time Systems, *Proc. of IPDPS*, pp. 9–17 (2003).
[5] Greenhalgh, P.: Big.LITTLE Processing with ARM Cortex-A15 & Cortex-A7, *White Paper* (2011).
[6] NVIDIA: Variable SMP - A Multi-Core CPU Architecture for Low Power and High Performance, *White Paper* (2011).
[7] 高瀬, 他: 排他動作する非均質マルチコアプロセッサとそのリアルタイム OS の実装, 情報処理学会研究報告, Vol. 2014-SLDM-165, No. 15 (2014).
[8] Weste, N. and Harris, D.: *CMOS VLSI Design: A Circuits and Systems Perspective*, Addison-Wesley Publishing Company (2010).
[9] Liu, C. L., et al.: Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, *Journal of the ACM*, Vol. 20, No. 1, pp. 46–61 (1973).
[10] Pillai, P. and Shin, K. G.: Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems, *Proc. of SOSP*, pp. 89–102 (2001).
[11] Hardkernel: Odroid XU3 - ODROID [online]. http://www.hardkernel.com/main/products/prdt_info.php?g_code=G140448267127 (2014).
[12] 理化学研究所情報基盤センター: 姫野ベンチマーク [online]. http://accc.riken.jp/2145.htm (2001).