

スケジューリング探索によるデータ通信消費電力削減

伊藤 和人[†]

[†] 埼玉大学 情報メディア基盤センター
〒338-8570 さいたま市桜区下大久保 255
E-mail: †kazuhito@ees.saitama-u.ac.jp

あらまし VLSI チップ上のモジュール間データ通信に伴う電力消費を P2P 接続やバス分割によって削減するとき、消費電力はモジュールの配置だけでなくモジュール間のデータ通信状況に依存する。モジュール間データ通信状況は、モジュールへの演算やデータの割り当てによって決まり、さらに割り当ての可否は演算スケジュールによって決まる。従って、データ通信消費電力を最小化する割り当てとフロアプランが達成できるスケジュールを得ることが重要となる。本研究では、データ通信消費電力削減を目的として最適な演算スケジュールを探索する手法を提案する。

キーワード 消費電力, バス分割, スケジューリング

Reduction of Power Consumption by Data Communications with Scheduling Exploration

Kazuhito ITO[†]

[†] Information Technology Center, Saitama University
255 Shimookubo, Sakura-ku, Saitama 338-8570, Japan
E-mail: †kazuhito@ees.saitama-u.ac.jp

Abstract Energy consumption by data communications of a VLSI with point-to-point or split bus architecture depends not on the assignment and floorplan of functional units and registers but how data communications are done. How data are communicated depends on the assignment, and the assignment depends on the schedule. Therefore, it is important to obtain the best schedule which leads to the best assignment and floorplan to minimize the power consumption by data communication. In this paper a schedule exploration method is presented to search the best one which achieves the minimized energy consumption.

Key words Low power, Bus split, Scheduling

1. はじめに

VLSI において、演算器やレジスタなどのモジュール間のデータ転送が相対的に大きなエネルギーを消費する [1]。データ転送は、転送元と転送先を接続する配線を使用して実現される。16 ビット整数加算器が加算を 1 回行うときの平均消費エネルギーを E とする。加算器の幅と同じ長さの配線を加算器出力に接続したところ、CMOS 0.5 μ m プロセスを仮定した SPICE シミュレーションでは、平均消費エネルギーは 1.556 E となる。この簡単な例から、わずかな配線長であってもデータ転送によるエネルギー消費は無視できないことが分かる。

データ転送では、配線を充電または放電することでビット値を伝達する。したがって、データ転送の消費エネルギーは配線の静電容量に比例し、さらに配線の静電容量は配線の長さに依存する。配線に起因する消費エネルギーは、通常の配線長では、

配線長にほぼ比例することが報告されている [2]。特定のデータ通信に必要な配線を最短化する接続方式として、ポイント・ツー・ポイント (P2P) 接続方式やバス分割を伴うバス接続方式が挙げられる [3-5]。これらの接続方式を用いる場合、データの転送元と転送先を物理的に近接して配置することによって配線を最短化し、データ転送の消費エネルギーを最小化することが可能となる。

VLSI におけるデータ転送のエネルギー消費は、物理的配置以外の要因にも依存する。一般に VLSI 設計では、フロアプランに先立ってスケジューリングと割り当てが行われる。スケジューリングでは、演算の実行開始時刻を決定する。割り当てでは、演算とその実行を担当する演算器、およびデータとその記憶を担当するレジスタの対応付けを決定する。フロアプランでは、2 次元 VLSI チップ上に演算器とレジスタを並べるために位置と向きを決定する。スケジューリングの結果は、割り当て

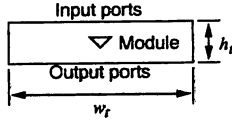


図1 モジュールのモデル.

に影響を及ぼす。なぜならば、スケジューリングにより同時に実行する演算は同一の演算器に割り当てられないためである。さらに割り当てはデータ転送によるエネルギー消費に影響を及ぼす。なぜならば、演算器とレジスタ（以降これらをモジュールと呼ぶ）間のデータ通信要求は割り当てに依存して決まるのであり、フロアプランはデータ通信要求のあるモジュール同士が近接して配置することで配線長が最短になるように作用するためである。以上のように、データ転送の消費エネルギーを最少化するには、スケジューリングと割り当てでも考慮する必要がある。

スケジューリングには多くの手法が知られており、演算器数やレジスタ数の最少化 [6]、実行時間最短化、モジュール間データ通信回数の最少化などを最適化目標としている。しかし、これらの最適化目標は必ずしもデータ転送の消費エネルギー最少化につながらない。さらに、スケジューリング結果は割り当てとフロアプランを通して間接的にデータ転送の消費エネルギーに影響するので、データ転送の消費エネルギーを最少化するスケジューリングを割り当ておよびフロアプランと無関係に求めることは困難である。そこで、実行可能なスケジューリングを探索し、エネルギー消費を最少にする最適解を得る手法が必要となる。

文献 [2] では、並行するデータ配線の間に非データ配線を挟むことで結合容量を低減してエネルギー消費を最少化している。バス分割に関しては、モジュールをクラスタに分割し、1つのクラスタが1つの分割バス区間を形成する場合に、バスのエネルギー消費を最少化するクラスタ分割手法 [4] が提案されている。文献 [5] もバス分割を考慮しているが、エネルギー消費よりはむしろデータ転送遅延の最小化を目指している。これらはいずれもスケジューリングと割り当てを工夫することでデータ転送のエネルギー消費を最少化することを考慮していない。

本論文では、データ転送によるエネルギー消費最少化を目的として実行可能なスケジューリングを探索する手法を提案する。

本稿は以下のように構成される。2 節ではハードウェアモデルを定義する。3 節では本研究の動機付けとなる例を示す。4 節ではスケジューリング探索手法を提案し、データ通信のエネルギー消費最少化を実装する方法を述べる。実験結果を 5 節に示す。

2. ハードウェアモデル

2.1 演算器およびレジスタ

加算器、乗算器といった演算器 (FU) は出力レジスタは持たず、演算結果は FU 外部のレジスタに記録する。FU がパイプライン化されている場合は、中間結果を保持するためのパイプラインレジスタを FU 内部に有する。また、FU には入力部にはラッチやレジスタは持たない。

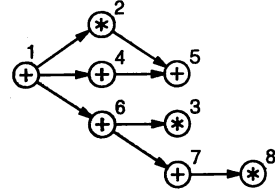


図2 例題 DFG.

以降では、演算器とレジスタを合わせてモジュールと呼ぶ。モジュールの形状は長方形とする。種類 t のモジュールは幅 w_t 、高さ h_t とする。図 1 に示すように、モジュールのデータ入力端子は幅方向の辺の中心にあり、出力端子は入力端子と対面の辺の中心にあると仮定する。

2.2 モジュール相互接続とエネルギー消費

モジュールはデータ転送のために相互に接続されている。セグメント化バス (分割バス [5]) または P2P 接続を用いることで、データ転送により充電または放電される配線長が最短になるようにモジュール間を接続する。

配線長が最短化されているので、配線の充放電によるエネルギー消費はデータの転送元と転送先の配線長に比例する [2]。配線長は、配線がモジュール上を通らない制約下で転送元と転送先間の最短経路長として計算できる。転送元はモジュールの出力端子、転送先は別のモジュールの入力端子となる。モジュール m_k の入力端子と出力端子をそれぞれ m_k^i, m_k^o とする。端子 m_k^i から m_k^o へのデータ通信 d が消費するエネルギー P_d は以下の式で与えられる。

$$P_d = WL(m_k^o, m_k^i) \times K_d \quad (1)$$

ここで $WL(m_k^o, m_k^i)$ は配線長、 K_d は比例係数とする。

転送元と転送先のペアに対して一般に複数回データ通信が行われることを考慮すると、データ通信の総消費エネルギー P_C^d は以下の式で与えられる。

$$P_C^d = \sum_{d \in D} \{P_d \times M_d\} \quad (2)$$

ここで D はすべてのデータ通信の転送元と転送先のペアの集合、 M_d はデータ通信 d の多重度とする。例えばデータ通信 d が 3 回行われる場合は、 $M_d = 3$ となる。

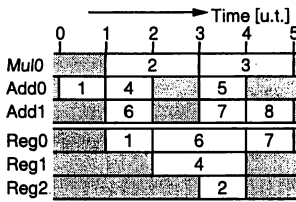
単純化のため、 K_d はすべてのデータ通信に共通とする。したがって、データ通信エネルギー消費度 P_C を以下のように定義する。

$$P_C = \sum_{d \in D} \{WL(m_k^o, m_k^i) \times M_d\} \quad (3)$$

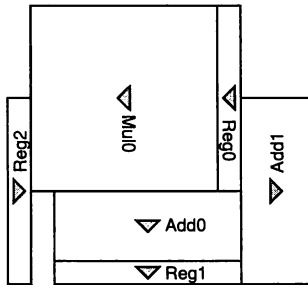
P_C を最小化することによってデータ通信の消費エネルギーを最少化できる。

2.3 フロアプラン

レイアウト設計におけるフロアプランでは、配線、バス分割器 (分割バス方式の場合)、マルチプレクサとデマルチプレクサ (P2P 方式の場合) の面積は無視する。



(a)



(b)

図3 設計結果。(a) 実行時刻および割り当て。(b) フロアプラン。

表1 図3の設計におけるデータ通信状況

Communication d	WL	M_d
Add0 → Reg0	11	1
Add0 → Reg1	0	1
Add1 → Reg0	7	2
Mul0 → Reg2	5	1
Reg0 → Add0	7	1
Reg0 → Add1	5	3
Reg0 → Mul0	0	2
Reg1 → Add1	8	1
Reg2 → Add0	5	1
$P_C = \sum \{WL_d \times M_d\}$	65	

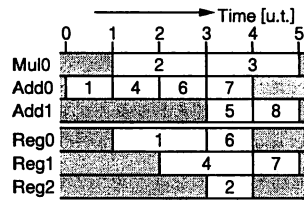
3. スケジューリングによるデータ通信エネルギー消費の影響

簡単な例を用いて、データ通信の消費エネルギーが割り当てとフロアプランだけでなくスケジューリングにも依存することを示す。

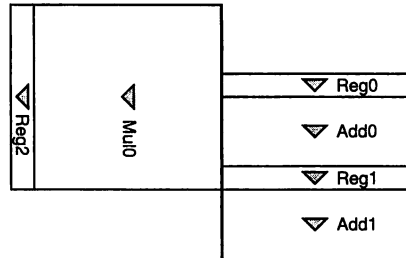
例に用いるデータフローグラフ (DFG) を図2に示す。DFGは2個の乗算 (内部に '*' 印のあるノード2と3) と6個の加算 (内部に '+' 印のあるその他のノード) からなる。加算時間は1単位時間 (unit of time, u.t.)、乗算時間は2u.t. とする。すべての計算を5u.t.以内に完了するように要求されていると仮定する。実際には、加算1, 4, 6, 7, 8には2個の入力が必要であるがこの例では省略し、図2に示されているデータ通信のみを考慮する。

乗算器の大きさは $w_M = 8$, $h_M = 8$ 、加算器は $w_A = 8$, $h_A = 3$ 、レジスタは $w_R = 8$, $h_R = 1$ とする。

FUとレジスタの数の最少化を目的とするスケジューリング手法を適用すると、ノードの実行時刻が図3(a)の通り決定する。図3(a)には、さらにノードをFUに、データをレジスタに割り当てた結果も示されている。ここで割り当ては、データ転送元と転送先のペアの数が最少になるように行う。



(a)



(b)

図4 データ通信によるエネルギー消費を最小化する設計結果。(a) 実行時刻および割り当て。(b) フロアプラン。

表2 図4の設計におけるデータ通信状況

Communication d	WL	M_d
Add0 → Reg0	12	2
Add0 → Reg1	0	2
Mul0 → Reg2	0	1
Reg0 → Add0	0	3
Reg0 → Mul0	4	2
Reg1 → Add1	0	2
Reg2 → Add0	17	1
$P_C = \sum \{WL_d \times M_d\}$	49	

割り当て結果に基づいて、データ通信エネルギー消費度 P_C を最小化するようにフロアプランを行った結果を図3(b)に示す。必要な通信 d 、配線長 WL 、多重度 M_d を表1に示す。例えば加算器 Add1 の出力からレジスタ Reg0 の入力へのデータ通信 (3行目) は、長さ7の配線を利用して2回実行される。最小化した P_C の値は65となる。

図2のDFGについて求めた別のスケジュールと割り当てを図4(a)に示す。このスケジュールは実行可能なスケジュールを探索し、フロアプラン後に P_C が最小となるものとして得ている。演算実行開始時刻は図3(a)と異なっているが、演算器数とレジスタ数はどちらも最少を達成している。

P_C 最小化を目的として同様にフロアプランを行った結果を図4(b)に示す。必要な通信 d 、配線長 WL 、多重度 M_d を表2に示す。この設計では P_C は49となり、図3の設計と比べてデータ通信消費エネルギーを約25%削減できる。

この例により、エネルギー消費を指向したスケジューリング、割り当て、フロアプランによってデータ通信によるエネルギー消費を最小化できることが分かる。

さらにこの例から、 P_C がデータ通信要求がモジュールのペアに分散する様子を依存することも分かる。図3の設計ではデータ通信が9個のモジュールペア間で行われており、データ通信

が7個のモジュールペアに限定されている図4の設計よりも P_C 値が大きいの。 P_C 値を減少するには、 M_d を減少し、さらにデータ通信配線長 WL を短くすればよい。式(3)中の M_d の値は、演算とFUおよびデータ期間とレジスタの割り当てによって決まる。したがって与えられた割り当てについて P_C を最小にするには、データ通信を行うモジュールペアを近接して配置することで WL を最小化する必要がある。ところが、モジュールペアが多くなるほどモジュールペアをそれぞれ近接して配置することは困難となる。データ通信が少数のモジュールペア間に限定されていれば、データ通信配線長が最短となるモジュール配置を得ることが容易であり、 P_C を最小化することができる。以上より、データ通信によるエネルギー消費を最小化するには、データ通信が多くのモジュールペアに分散することを防ぐことが有効であると考えられる。

4. データ通信エネルギー消費最小化

4.1 スケジュール探索

スケジューリングは、DFG中の全ての演算間先行制約を満足するように演算実行開始時刻を決定することであり、NP困難な問題である。そのため、発見的スケジューリング手法が数多く提案されており、スケジューリング結果の質および計算複雑度もさまざまである。最も簡単なものの1つが最早(ASAP)スケジュールであり、最長経路問題を解くことで先行制約関係を満足する演算実行時刻を得る。ASAPスケジュールは、FU数やエネルギー消費を考慮しない。

レンジチャートガイドスケジューリング手法[6]は、与えられたDFGの実行に必要なFU数を最小化するスケジュールを得る。演算のスケジューリングレンジは、先行制約に違反せずに演算を実行開始できる時刻の集合と定義される。スケジューリングレンジの下限(LB)と上限(UB)は、それぞれASAPスケジュールと最遅(ALAP)スケジュールとして得られる。各演算について、同時に実行する演算数、すなわち必要なFU数が最少となる時刻をスケジューリングレンジの中から選び、演算実行開始時刻として決定する。

レンジチャートガイドスケジューリング手法では、多くの同順が発生する。例えば、実行開始時刻を決定する演算を選ぶ際に複数の演算が同じ選択順になる場合、任意に1つの演算を選択する。さらに、選択した演算について、スケジューリングレンジの中で複数の時刻について必要なFU数が最少になる場合、LBまたはUBに近い時刻を選ぶ。これら同順解決方法を用いても、FU数最小化の面では最適性を損なわない解を得ることができる。しかし、このような単純な同順解決方法では、データ通信エネルギー消費を真に最小化するスケジュールを得ることは困難である。そこで、可能なスケジュールを探索し最適解を得る手法を提案する。

4.1.1 Strutを用いるスケジューリング

レンジチャートガイドスケジューリングでは、演算の実行開始時刻はスケジューリングレンジの中から選ぶ。すなわち、演算 j のスケジューリングレンジの下限が LB_j 、上限が UB_j のとき、 j の実行開始時刻 t_j は $LB_j \leq t_j \leq UB_j$ を満たす。ただ

し、 $W_j = UB_j - LB_j$ とする。演算 j へ入力するDFG上の枝を (i, j) とすると、先行制約によって一般に

$$t_j \geq t_i + q_i - d_{ij} * Tr \quad (4)$$

が成り立つ。ただし q_i は演算 i の演算時間、 d_{ij} は枝 (i, j) の遅延数、 Tr は繰り返し周期とする。ここで、式(5)により与えられるパラメータ S_{ij} を導入する。 S_{ij} は、 $0 \leq S_{ij} \leq W_k$ を満たす。

$$S_{ij} = t_j - (t_i + q_i - d_{ij} * Tr) \quad (5)$$

演算 j へ入力する枝の集合を E_j とすれば

$$t_j \geq \max_{(i,j) \in E_j} t_i + q_i - d_{ij} * Tr \quad (6)$$

であり、 S_{ij} を用いて

$$t_j = \max_{(i,j) \in E_j} t_i + q_i - d_{ij} * Tr + S_{ij} \quad (7)$$

となる。これは、枝 (i, j) の重みを $t_i + q_i - d_{ij} * Tr + S_{ij}$ として最長経路を求めていることになる。すなわち、枝 (i, j) に重み S_{ij} を加えてASAPスケジューリングを行うことで、レンジチャートガイドスケジューリング手法と同じスケジュールを導くことができることを意味する。 S_{ij} は演算 j の実行時刻を下限 LB_j よりも遅らせる効果があり、支柱(Strut)と定義する。

Strutを伴うスケジューリングアルゴリズム

入力:

DFG = (N, E)

N : 演算集合

E : 演算間データ依存関係の枝集合

q_j : 演算 j ($j \in N$)の演算時間

d_{ij} : 枝 (i, j) ($i \in E$)上の遅延数

S_{ij} : 枝 (i, j) ($i \in E$)のStrut

s : 参照演算 ($s \in N$)

Tr : DFGの繰り返し周期

出力:

全演算 $k \in N$ の演算実行開始時刻 t_k

1. 全ての演算 j について $t_j = -\infty$ とする。
参照演算 s について $t_s = 0$ とする。
2. 各演算 $j \in N$ について、以下のように t'_j を計算する。

$$t'_j = \max_{(i,j) \in E} \{t_i + q_i - d_{ij} * Tr + S_{ij}\} \quad (8)$$

3. もし $t_j < t'_j$ ならば、 t_j に t'_j を代入する(更新)。いずれかの演算について更新があれば2.へ戻る。
更新がなければ終了。

このアルゴリズムはBellman-Ford法[7]による最長経路問題の解であり、Strutの値が適切で正閉路がなければ必ず終了し、得られた t_j は全ての先行制約を満たす。 S_{ij} の組み合わせを変えることで、可能なスケジュールを探索することができる。

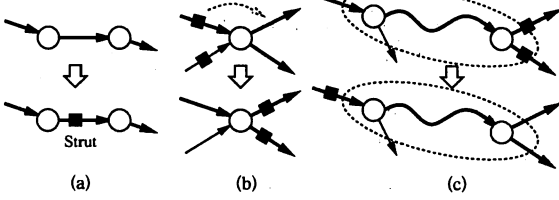


図5 Strutの変更。(a)Strutの単体増加。(b)演算の入力から出力へStrut移動。(c)部分木の出力から入力へStrut移動。

4.1.2 焼きなまし法によるスケジューリング探索

焼きなまし法 (Simulated annealing, SA) は、組み合わせ最適化問題を解く汎用の手法である。本研究では、SA を用いて Strut の最適な組み合わせを見つける。最適化手法を以下に示す。

スケジューリング探索

1. 全ての枝 $(k, j) \in E$ について $S_{kj} = 0$ とする。
2. Strut を伴うスケジューリング手法を用いて全ての演算 k ($\in N$) の実行開始時刻を決定する。これを最初のスケジューリング候補 SC とし、割り当てを行ってコスト $Cost$ を評価する。
3. 温度 T を T_{start} に初期化する。
4. 図5に示す (a)Strut 増加、(b)単一演算上の Strut の前方移動、(c)部分木上の Strut 後方移動、のいずれかの操作を選び、先行制約に違反しない範囲内で枝 (i, j) の新しい Strut S'_{ij} を得る。Strut 値が変化する枝の集合を Ex とする。
5. 新しい Strut 値を用いて演算の実行時刻を決定する。これを新しいスケジューリング候補 SC' とする。
6. SC' について割り当てを行ってコスト $Cost'$ を評価する。
7. $Cost' < Cost$ のとき、または $Cost' \geq Cost$ の場合は以下の確率 p でスケジューリング候補 SC' を受理する。すなわち、 Ex に属する枝 (i, j) について $S_{ij} = S'_{ij}$ とする。

$$p = \exp\left(\frac{-Cost' - Cost}{T}\right) \quad (9)$$

8. 所定回数 M だけステップ4~7を繰り返す。
9. 温度 T に係数 $\alpha (< 1)$ を乗じることで T を低下する。もし $T < T_{end}$ ならば終了する。さもなければ4.へ戻る。

操作 (a) は枝に任意の Strut を加える。操作 (b) は演算 k を選び、 k の入力枝の Strut を $u (> 0)$ だけ減らし、出力枝の Strut を u だけ増やす。操作 (c) は最長経路木の部分木を選び、木から外へ出る枝の Strut を $u (> 0)$ だけ減らし、木根の演算への入力枝の Strut を u だけ増やす。これらの操作は、正閉路を生成せず、かつ Strut が負にならない条件下で行う。

4.2 エネルギー消費指向割り当て

SA によるスケジューリング探索では、各スケジューリング候補についてデータ通信の消費エネルギーをコストとして評価する。式 (3) に示すように、エネルギー消費は割り当てとフロアプランに依存する。各スケジューリング候補について、消費エネルギーを最少

とする割り当てとフロアプランが存在する。しかし、最適な割り当てとフロアプランを得るには時間を要する。そこで、まず割り当てを求め、その後にフロアプランを求めることとする。

割り当ての目的はファンアウトの最少化とする。モジュール k のファンアウト $fanout_k$ を、モジュール k が出力するデータを受け取るモジュールの数と定義する。ファンアウトが小さければデータ通信が少数の配線に集中するので、式 (3) 中の M_d が大きくなるが、データ通信を行うモジュール同士を近接して配置することが可能となり、配線長 WL を短くできるので、結果的に P_C を低減できる。

以下に割り当て手法を示す。

ファンアウト最少化割り当て手法

入力: 演算 k ($\in N$) の実行開始時刻

出力: 演算と FU およびデータ区間とレジスタの割り当て。

1. 未割り当ての演算を選び k とする。
2. 演算実行時間の重複により k を既存の FU に割り当てられない場合は新たに FU を発生して k を割り当てる。さもなければ、既存の FU のうち k を割り当てたときのファンアウトの増加が最少の FU を選び k を割り当てる。この FU を F_k とする。
3. 演算 k が出力するデータの生存期間を I_k とする。データ期間の重複により I_k を既存のレジスタに割り当てられない場合は新たにレジスタを発生して I_k を割り当てる。さもなければ、既存のレジスタのうち I_k を割り当てたときの F_k のファンアウトとレジスタのファンアウトの増加が最少のレジスタを選び I_k を割り当てる。
4. すべての演算が割り当てられたら終了する。さもなければ1.へ戻る。

割り当てのコストを以下の式で計算する。

$$Cost = \sum_{\text{module } m} \left\{ (fanout_m)^2 + \beta \right\} \quad (10)$$

右辺第2項は重み $\beta > 0$ をつけてモジュール数を考慮するためである。十分大きな β を用いると、ファンアウトの最少化とともにモジュール数も最少化できる。式 (10) の $Cost$ は、SA によるスケジューリング探索の際のコストとして用いる。

4.3 エネルギー消費指向フロアプラン

割り当て結果に基づいて Sequence-Pair[8]を用いた SA によって最適なフロアプランを求める。SA の目的関数はデータ通信の消費エネルギー最少化である。各フロアプラン候補についてデータ通信の転送元と転送先の最短経路を求めて配線長とする。最小化するコストとして式 (3) の P_C を用いる。

5. 実験結果

提案手法を C++言語を用いて実装し、PC(Pentium 2.4GHz, 512MB メモリ)上で実行した。表3に提案手法および従来手法の実験結果を示す。従来手法では、レンジチャートガイドスケジューリング手法を用いて演算器数最少化を目的にスケジュー

表3 結果比較

DFG	Tr	Proposed					Conventional				
		Assign (Cost)	fanout	TimeA	FPlan (P_C)	TimeFP	Assign (Cost)	fanout	TimeA	FPlan (P_C)	TimeFP
WEF	16	1551	151	31.9s	720	295s	1571	171	0s	740	167s
	17	1431	131	50.4s	726	238s	1570	170	0s	759	142s
	18	1322	122	48.8s	690	201s	1493	193	0s	1093	142s
	19	1298	98	50.9s	648	211s	1479	179	0s	1070	141s
DCT	8	1782	182	113s	876	281s	2263	263	0s	1033	520s
	9	1685	185	139s	863	279s	2392	292	0s	1050	636s
	10	1631	231	162s	908	215s	2056	256	0s	988	429s
	11	1601	201	134s	899	235s	1902	202	0s	974	384s

ルを生成し、提案する割り当てとフロアプランを施している。

実験に用いた DFG は 5 次ウェーブ楕円フィルタ (WEF)[6] と 8 点 1 次元 DCT[9] である。FU の実行時間と形状は 3 節と同じとする。各 DFG についてそれぞれ 4 通りの繰り返し周期 Tr を用いて設計を行った。WEF では 16 u.t.、DCT では 8 u.t. は実現可能な最短の繰り返し周期である。本実験では、式 (10) で用いる β は 100 とした。SA パラメータは、スケジューリング探索では $T_{start} = 10$, $T_{end} = 0.1$, $\alpha = 0.95$, $M = 500$ 、フロアプランでは $T_{start} = 50$, $T_{end} = 0.5$, $\alpha = 0.99$, $M = 1000$ を用いた。

表 3 の列 'Assign (Cost)' は、式 (10) によって算出するコストを示す。提案手法では、SA によって最小化されている。列 'fanout' は式 (10) の右辺第 1 項の値、列 'TimeA' は CPU 時間を示す。例えば WEF で $Tr = 16$ u.t. のとき、最小割り当てコスト 1551 が 31.9 秒で得られている。コスト 1551 の中で、ファンアウト項は 151 である。 $\beta = 100$ であることから、 $(1555 - 151)/\beta = 14$ としてモジュール数が 14 個であることが分かる。一方、従来手法では発見的に求めたスケジューリングに対して割り当てを行う。レンジチャートガイドスケジューリング手法および割り当て手法は単純であり、CPU 時間は 1 秒未満である。列 'FPlan (P_C)' は SA によって最小化したフロアプランのコスト P_C 、列 'TimeFP' はフロアプランに要した CPU 時間を示す。

WEF の場合、 Tr が 16 から 17 へ増加すると P_C の値も増加している。これは、モジュール数が 14 から 13 へ減少したことでデータ通信の多重度 (M_d) が増すためである。この場合でも、提案手法は従来手法よりも小さな P_C 、つまりより少ない消費エネルギーとなっている。レンジチャートガイドスケジューリング手法では演算器数は最小化するがレジスタ数は考慮しないため、 Tr が 17, 18, 19 の場合に従来手法ではモジュール数が増えている。モジュール数が多いと配線長が長くなる傾向があり、結果として P_C が大きくなる。 Tr が大きくなると、より多くのスケジューリング候補が可能であり、提案手法を用いることで最適なスケジューリング候補を探していることが分かる。

DCT の場合、並列に実行される演算が多く、レンジチャートガイドスケジューリング手法では多くの同順を発生する。一方、提案するスケジューリング探索ではデータ通信エネルギー消費を最小化する割り当てとフロアプランに到達する最適なスケジューリングを得ている。 $Tr = 11$ のとき、Cost のファンアウト項は提案手法と従来手法でほぼ同じ値となっている。しかし、提案手法では従来手法よりも少ないモジュール数のスケジューリングを得て

おり、フロアプランの結果では P_C が小さい。

すべての場合において提案手法を用いることで従来手法よりも小さな P_C が得られている。WEF では最大 40%、DCT では最大 18% の消費エネルギーの削減ができる。以上の結果より、提案手法はデータ通信によるエネルギー消費最小化に効果があることが確認できる。

6. ま と め

本研究では、スケジューリングを探索する手法を提案し、P2P 方式または分割バス方式のアーキテクチャにおけるデータ通信のエネルギー消費の最小化に適用する方法を示した。スケジューリング探索により、多くのスケジューリング候補の中から、データ通信エネルギー消費を最小化する割り当てとフロアプランを達成する最適なスケジューリングを見つけることができる。実験により提案手法の有効性を示した。

今後の課題として SA パラメータのチューニング、バスアーキテクチャにおいて複数のデータ転送先に同時にデータ通信する場合の考慮、配線とバス分割用スイッチの面積の考慮などが挙げられる。

文 献

- [1] S. Komatsu, M. Ikeda, and K. Asada, "Low Power Microprocessors for Comparative Study on Bus Architecture and Multiplexer Architecture," Proc. ASP-DAC '98, pp. 323-324, 1998.
- [2] N. Taylor, Sujit Dey, and Y. Zhao, "Modeling and Minimization of Interconnect Energy Dissipation in Nanometer Technologies," Proc. DAC 2003, pp. 754-757, 2003.
- [3] V. Raghunathan, M. B. Srivastava, and R. K. Gupta, "A Survey of Techniques for Energy Efficient On-Chip Communication," Proc. DAC 2003, pp. 900-905, 2003.
- [4] C. T. Hsieh and M. Pedram, "Architectural Energy Optimization by Bus Splitting," IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst., vol. 21, pp. 408-414, 2002.
- [5] R. Lu and C.-K. Koh, "A High Performance Bus Communication Architecture through Bus Splitting," Proc. ASP-DAC 2004, pp. 751-755, 2004.
- [6] S. M. Heemstra de Groot, S. H. Gerez, and O. E. Herrmann, "Range-Chart-Guided Iterative Data-Flow Graph Scheduling," IEEE Trans. Circuits Syst.-I: Fund. Theory & Appl., vol. 39, pp. 351-364, 1992.
- [7] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, 1976.
- [8] M. Murata, S. Natakake, K. Fujiyoshi, and Y. Kajitani, "VLSI Module Placement based on Rectangle Packing by Sequence-Pair," IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst., vol. 15, pp. 1518-1524, 1996.
- [9] C. Loeffler, A. Ligtenberg, and G. S. Moschytz, "Practical Fast 1-D DCT Algorithms with 11 Multiplications," Proc. IEEE ICASSP '89, pp. 988-991, 1989.