社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2005－ＳＬＤＭ－121（25）
2005／10／21

ビットレベル並列性を利用した演算器の小規模化

多田十兵衛† 江川 隆輔†† 後藤 源助† 中村 維男††

† 山形大学工学部 〒992-8510 山形県米沢市城南 4-3-16
†† 東北大学大学院情報科学研究科 〒980-8579 仙台市青葉区荒巻字青葉 6-6-01
E-mail: †{jubee,gengoto}@yz.yamagata-u.ac.jp, ††{egawa,nakamura}@archi.is.tohoku.ac.jp

あらまし 本研究では，回路の大規模化に伴うリーク電流による消費電力の増大を抑えることを目的として，小規模な演算器で高速に演算を行なう手法を提案する．演算内のビットレベル並列性に着目し，高ビット幅の演算を低ビット幅の演算器で行い，さらに回路をウェーブパイプライン化することで高速かつ小規模化な回路を実現する．また，シミュレーションにより提案手法の有効性を示す．

キーワード 低電力，演算器，ウェーブパイプライン

Compaction of Arithmetic Unit with Bit-Level-Parallelism

Jubei TADA†, Ryusuke EGAWA††, Gensuke GOTO†, and Tadao NAKAMURA††

† Faculty of Engineering, Yamagata University Jo-nan 4-3-16, Yonezawa, Yamagata, 992-8510 Japan
†† Graduate School of Information Sciences, Tohoku University Aramaki Aza Aoba 6-6-01, Aoba-ku, Sendai-shi, 980-8579 Japan
E-mail: †{jubee,gengoto}@yz.yamagata-u.ac.jp, ††{egawa,nakamura}@archi.is.tohoku.ac.jp

Abstract Aiming at reducing power consumption of VLSIs, we propose a fast and compact arithmetic unit. The arithmetic unit reduces static power consumption by the compaction or the circuit scale. To realize that compaction of the arithmetic unit, we exploit bit level parallelism of arithmetic operation, and also, our approach keeps the throughput and saves a dynamic power consumption employing advanced pipelining technique. The simulation results show a high validity of our proposal on VLSI design in deep submicron era.

Key words Low-power, Arithmetic Unit, Wave-Pipelining

1. まえがき

近年，半導体プロセスの微細化に伴うリーク電流の増加が問題となっている．70nm 以下のプロセスでは，リーク電流などによる静的な消費電力がマイクロプロセッサの消費電力の半分以上を占めると言われている [1]．これにより，低消費電力なマイクロプロセッサを設計するためにリーク電流を削減する回路設計技術が求められている．

本研究では，演算内のビットレベル並列性に着目し，並列性のあるビット列を低ビットの演算器に連続して投入する手法を提案する．遅延時間が短く，回路規模の小さい演算器を利用することにより，静的消費電力の削減を図る．

さらに，演算器をウェーブパイプライン化することにより，回路規模を押さえつつ性能を大幅に向上させる手法を提案する．ウェーブパイプラインは，回路の全てのパスの遅延を均一にすることにより，回路に複数の信号を連続して投入できるようにする技術である．ウェーブパイプラインは従来のパイプラインよりも動的な消費電力を削減できることが報告されている [3]．低ビット幅の演算器をウェーブパイプライン化することにより，動的および静的消費電力を削減しつつ演算を高速に実行することが可能となる．

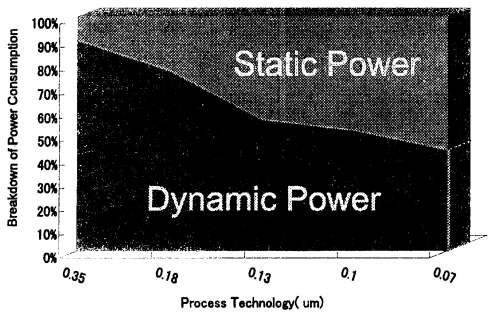
2. 演算回路の小規模化

2.1 静的消費電力の削減

マイクロプロセッサの消費電力 P は，(1) 式に示すように，右辺第一項の動的消費電力と第二項の静的消費電力の和で表される．

$$P = N_a \times f \times C \times V^2 + N_t \times V \times I_l \qquad (1)$$

ここで N_a はアクティブなノード数，f はクロック周波数，C は 1 ノードあたりの平均容量，V は電圧，N_t は全ノード数，そして I_l はリーク電流である．これまでの半導体加工技術下におけるマイクロプロセッサでは，動的な消費電力が支配的であっ



K.S. Khouri, et. Al, "Leakage Power Analysis and Reduction During Behavioral Synthesis," IEEE tran. on VLSI systems, Vol.10 No.6,2002.

図 1 プロセッサの消費電力の割合

Fig. 1 A Ratio of Processor Power Consumption.

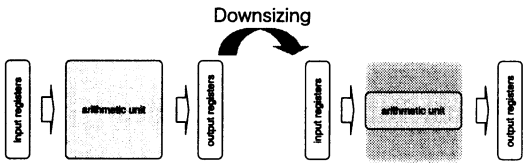


図 2 演算器の小規模化

Fig. 2 Compaction of an Arithmetic Unit.

た．しかし，近未来のディープサブサブミクロン下において，高速，かつ低消費電力なマイクロプロセッサを実現するには双方の電力消費を削減することが強く求められている．Kamal らは，70nm 以下の CMOS 加工技術において，静的な消費電力が半分以上を占めると報告している (図 1) [1]．これは半導体加工技術の著しい微細化が進むにつれ，閾値電圧の低下や，ゲート酸化膜の微薄化に伴いリーク電流が大きくなることに起因する．リーク電流を削減するべく，SOI (Silicon on Insulator) に代表されるようにデバイスレベルにおける研究が精力的に行われている [2]．しかし，デバイスレベルの研究に比べマイクロアーキテクチャレベル，論理回路設計に於けるリーク電力抑制に関する研究において著しい効果を示したものは多くない．

しかし，式 (1) からも明らかなように，リーク電流は回路のノード数に比例することから，回路を小規模化することによってリーク電流にまつわる諸費電力消費を抑制することが可能であると考えられる．そこで本報告では，回路の小規模化を行なうことで静的な消費電力の削減を試みる．

図 2 は本研究の基本的な戦略を示したものである．前述の議論のとおり，回路のリーク電流は回路規模に大きく依存する．そこで本研究では，回路の消費電力を削減することを目的として，回路の規模の小規模化を試みる．具体的にはビットレベル並列性を利用し，n-bit の演算を n/m ビットの演算回路で行なうことで回路の小規模化を実現する．

本研究の詳細なアプローチを図 3 に示す．はじめに n-bit の演算を n/m ビットの演算に分割する．図 3 は m=4 の場合である．しかし，複数 (m 個) の演算回路を用いた場合回路規模の

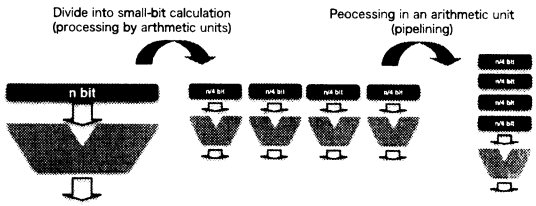


図 3 演算の分割

Fig. 3 Division of an Arithmetic

削減にはならない．そこで，一つの n/m ビットの演算回路を用いて n ビットの演算が可能となるように，演算のビットレベル並列性を抽出する．これによって単一の小ビット幅な演算回路を用いて，ビット幅の大きな演算を可能となる．ビットレベル並列性を用いることによって被るスループットの減少は，演算回路の時間的並列性，つまりパイプライン手法を適用することで補う．これらにより，高速かつ低消費電力の実現をめざす．

3. 乗算器の小規模化

演算のうち，加算には多くの並列性があり，桁上げ選択加算器 (CSLA) のようにハードウェアを冗長に用いることにより高速な演算の実行が可能である．また，冗長 2 進数を用いて桁上げの伝播の必要性をなくし，並列性を高めた加算器も提案されている．これらで用いられている複数のハードウェアを，一つのハードウェアに置き換えることで並列性を利用することが可能であるため，演算器の小型化は比較的容易である．

乗算では，部分積の圧縮過程に多くの並列性があり，これを Wallace-Tree [4] を用いて利用することで高速な計算を実現することが可能である．乗算内のビットレベル並列性を生かした研究としては，高ビット幅の乗算器で複数の低ビットの乗算を行なう手法が提案されている [6]．しかし，Wallace-Tree の回路規模は演算のビット幅が大きくなるにつれ爆発的に増加するため，高ビットかつ高速な乗算器は設計しにくいという問題がある．

そこで本研究では，高ビットの乗算器の小規模化を目的として，Wallace-Tree の回路規模を大幅に削減しつつ，高速な計算を実現する手法を提案する．

並列乗算器における演算の実行は以下の 3 つのステップで行われる．

Step1. Booth エンコーダによる部分積の生成

Step2. Wallace-Tree による部分積の圧縮

Step3. 加算器による加算 (桁上げ吸収)

ここで，乗算器の遅延時間 T_{mul} は，Booth エンコーダ，Wallace-Tree，そして加算器の遅延時間をそれぞれ T_{booth}，T_{tree}，T_{add} とすると

$$T_{mul} = T_{booth} + T_{tree} + T_{add} \quad (2)$$

となる．

ここで，T_{booth}，T_{tree} は，並列性を有効に利用した高速化が

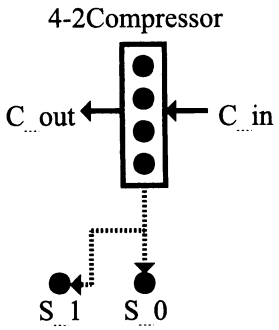


図 4　4-2Compressor の構造
Fig. 4　Structure of 4-2Compressor

可能であるが，多くのハードウェア資源を必要とする．また，Wallace-Tree の回路規模は，ビット数の増加につれて大きく増加する．回路を小規模しつつ遅延時間を削減するためには，T_{add} を削減する必要がある．そこで，T_{add} の時間を削減しつつ，回路規模を大幅に削減する手法として，加算器を低ビットで設計し，Wallace-Tree をそのビット数に合わせて設計する手法を提案する．

図 4 に Wallace-Tree に用いられる 4-2Compressor [5] の動作を示す．4-2Compressor は，4 つの部分積の同一ビットと桁上がりの入力を加算し，加算結果をその桁と上位の桁に，そして桁上がりを上位ビットに出力する．8 ビットの乗算における 4-2Compressor を用いた部分積の圧縮処理を図に示す．ビット数が増大すると 4-2Compressor の数が大幅に増加する．4-2Compressor では，桁上がりの入力による出力が次の桁に出力される．従って，4-2Compressor の出力を確定させるためには，必要な桁のビット数＋ 2-bit の入力を行えばよい．

乗算のビット数を N_{mul} とすると，Booth エンコーダが出力する部分積の数は $N_{mul}/2+1$ となる．ここで，後述するウェーブパイプラインにおいて遅延調整を容易にするため，Wallace-Tree には $log_2(N_{mul}/2)$ 個の部分積のみを入力することとする．4-2Compressor を用いた Wallace-Tree の段数は $log_2(N_{mul}/2)-1$ となる．加算器のビット数を N_{add} とすると，Wallace-Tree の入力ビット数 N_{tree} は

$$N_{tree} = N_{add} + 2 \times (log_2(N_{mul}/2) - 1) \qquad (3)$$

となる．従って，32-bit の乗算を 8-bit の加算器で行なう場合には，Wallace-Tree の入力ビット数は 14-bit となる．

図 5 に 32-bit の乗算を 8-bit 加算器を用いて実行する場合の乗算器の構成を示す．乗算器は BoothEncoder，Selector，Wallace-Tree，Adder，Distributor から構成される．ここで Selector は，BoothEncoder が出力した部分積を Wallace-Tree に入力する機構である．Distributor は，Adder の出力を正しい位置に出力するための機構である．

図 6 に BoothEncoder が出力した部分積のうちの 1 つを Wallace-Tree に入力する Selector の構造を示す．Selector は，

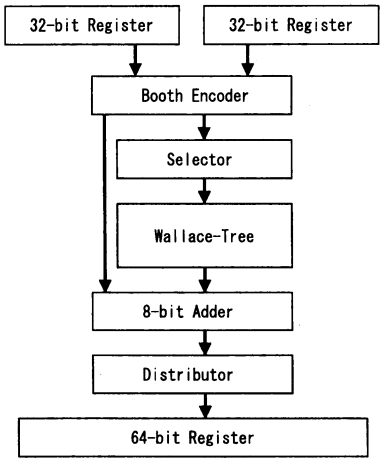


図 5　乗算器の構造
Fig. 5　Structure of a Multiplier

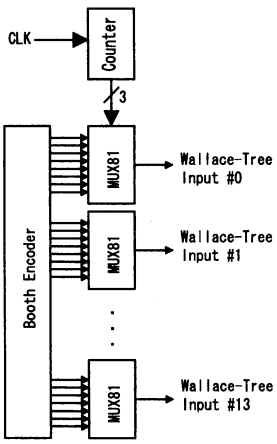


図 6　Selector の構造
Fig. 6　Structure of a Selector

BoothEncoder の出力から適当なビット列を選択し，Wallace-Tree に入力する．ここで選択されるのは，Adder の最上位ビットに対応する位置から，(3) 式で表されるビット数のビット列である．

図 7 に Wallace-Tree の構造を示す．Wallace-Tree の入力ビット数は (3) 式から求められる．一段ごとに 4-2Compressor の数が 2 つずつ削減して行き，最終段では Adder のビット数＋ 2 個となる．

図 ??に Adder の構造を示す．この加算器では，Wallace-Tree から出力された 2 つの部分積と，BoothEncoder が出力する 1 つの部分積が加算され出力される．最上位桁上がりは保存され，次の計算で最下位ビットの桁上がり入力となる．

次節では実際に設計を行い，提案手法を評価する．

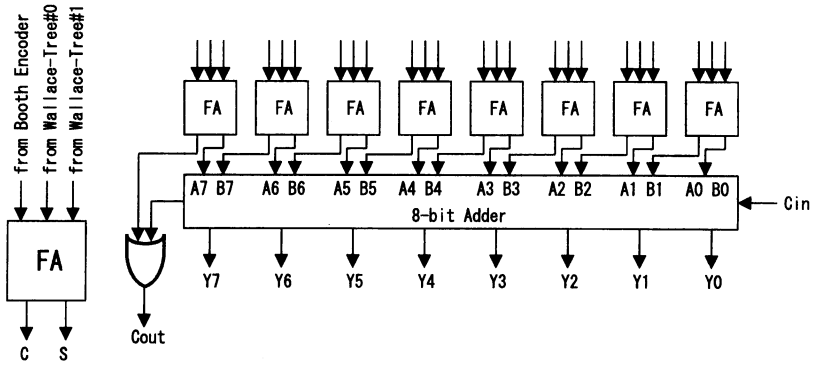


図 8 Adder の構造
Fig. 8 Structure of an Adder

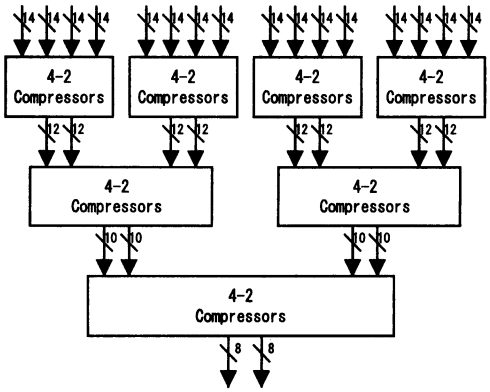


図 7 Walalce-Tree の構造
Fig. 7 Structure of a Wallace-Tree

表 1 4-2 Compressor の総数
Table 1 Total Number of 4-2 Compressors

Input of the Multipiler	Input of the Adder 4-bit	8-bit	16-bit	Normal
16-bit	14	22	38	80
32-bit	24	36	60	320
64-bit	36	52	84	1216

4. 実　　験

本節では，前節で述べた乗算器を設計し，その回路規模および遅延時間について評価を行なう．設計環境として Xilinx ISE7.1i, FPGA デバイスは Virtex4(LX25-10FF668) を用いた．

表 1 に，各ビットの乗算における Wallace-Tree に必要な 4-2Compressor の数を示す．通常の乗算器では，ビット数が増加するにつれ必要な 4-2Compressor の数が大幅に増加する．これに対して，提案手法では演算回数はビット数に比例して増加するが，回路規模の増加は緩やかである．

表 2 に，32-bit 乗算器における各構成でのロジック部の遅延時間および計算の実行に必要なサイクル数を示す．加算器のビット数が少ない場合，演算回路へ信号を入力するための機構が複雑となり，遅延が増加している．また計算回数も増加するため，演算の分割数としては 4 程度が適当と考えられる．

表 2 乗算器の遅延時間
Table 2 Delay of Multipliers

	Input of the Adder 4-bit	8-bit	16-bit	normal
Delay(ns)	6.578	6.559	6.124	11.06
# of Cycles	16	8	4	1

次節では，スループットの低下を防ぎ，高速かつ低消費電力な演算の実行を実現するため，演算回路をウェーブパイプライン化する手法について考察する．

4.1 ウェーブパイプライン

従来のパイプライン手法は組み合わせ論理回路を複数のセグメントに分割し，各セグメント間の同期を取るために複数個のパイプラインレジスタを必要とする．このパイプラインレジスタにと伴う遅延時間，回路面積，消費電力に関するオーバーヘッドは半導体プロセスの微細化が進むにつれ顕著になる．

一方，ウェーブパイプライン手法は，最初のクロックで出力された遅い信号が次のクロックで出力された速い信号に衝突するのを避けるために，組み合わせ論理回路の全ての入力から全ての出力までの経路遅延時間を可能な限り等しくすることによってパイプライン動作（時間的並列）を可能にする手法である．従来のパイプライン手法と異なり，ウェーブパイプラインラインのクロックサイクルは組み合わせ論理回路の最大遅延時間と最小遅延時間の差によって決まる．この特徴によりウェーブパイプライン手法は従来のパイプラインを凌ぐ高速動作が可能である．また，同期のパイプラインレジスタを用いないことから，ウェーブパイプライン化された組み合わせ論理回路は従来のパイプラインと比較して，低コスト，低消費電力であることが報告されている [3]．

しかし，向上したスループットを利用して性能を上げるためには，並列に実行可能な多くの演算が必要である．また，ウェーブパイプライン化した演算器にデータを入力する間隔は，演

表 3 ウェーブパイプライン化乗算器の遅延時間
Table 3 Delay of Multipliers

	Input of the Adder			
	4-bit	8-bit	16-bit	normal
Delay(ns)	12.97	9.541	7.402	11.06

算器を組み込むシステムのクロック周波数に依存するため，クロック周波数を高く出来ない場合，演算器のスループットが向上してもそれに見合うだけのデータを演算器に投入できないという問題が発生する．

そこで本研究では，提案した乗算器の演算回路をウェーブパイプライン化し，各ビット列をウェーブとして回路内に投入することで，演算を高速に実行する手法を提案する．ウェーブパイプラインの性能を引き出すための並列性をビットレベルに求め，高スループットを利用して高速かつ低消費電力な演算の実行を図る．

前節で提案した乗算器をウェーブパイプライン化する場合には，Wallace-Tree および Adder をウェーブパイプライン化し，信号を演算回路内に投入するための Selector，演算回路の出力を適切な位置に振り分けるための Distributor を，ウェーブの投入間隔で動作するよう調整する手法が適当と考えられる．

ここで，ウェーブパイプライン化を行なった場合の回路の性能について述べる．回路の遅延時間を D，ウェーブの投入間隔を W，計算回数を N とすると，提案するウェーブ化を行なった回路におけるクロック時間 T_{wave} は

$$T_{wave} = D + (N-1) \times W \tag{4}$$

となる．W は，回路の最大最小遅延差などから決定される．ここで，W をインバータ回路 1 つ分の遅延にすることができると仮定する．この場合の各構成におけるクロック時間を表 3 に示す．ウェーブパイプライン化した乗算器では，回路規模の抑制と計算時間の短縮が同時に達成される．特に，従来の乗算器ではビット数の増加につれ指数的に増加していた回路規模を大幅に削減可能であるため，高ビットの演算を行なう場合に効果が大きいと考えられる．

しかし，演算回路の全てのパスの遅延を均一にするためには，回路内に多くの遅延素子を挿入する必要がある．また提案手法では，信号を分割し，ウェーブとして入出力を行なう機構が必要となる．これらのオーバーヘッドによる回路規模および遅延の増加の評価は，今後の課題である．

5. ま と め

本研究では，回路の大規模化に伴うリーク電流による消費電力の増大を抑えるために，演算器を小規模化し，さらにウェーブパイプライン手法を用いて高速に演算を行なう手法を提案した．提案手法にもとづき回路設計を行い，乗算器の性能を評価した．ウェーブパイプライン化した場合の遅延素子の挿入，入出力装置の追加による回路規模の増加および性能の評価は今後の課題である．

文 献

[1] Kamal S. Khouri and Niraj K. Jha, "Leakage Power Analysis and Reduction During Behavioral Synthesis", IEEE trans. on VLSI Systems, Vol. 10, No.6, pp.876 – 885, Dec. 2005.

[2] B. Yu, Z.-J. Ma, G. Zhang, and C. Hu, "Hot-carrier effect in ultrathin film (UTF) fully-depleted SOI MOSFETs", Proc. 54th Annu. Device Research Conf., Jun. 24–26, 1996, pp. 22–23.

[3] G. Lakshminarayanan and B. Venkataramani, "Optimization Techniques for FPGA-Based Wave-Pipelined DSP Blocks", IEEE trans. on VLSI Systems, Vol. 13, No. 6, pp. 783 – 793, July 2005.

[4] C. S. Wallace, "A suggestion for a fast multiplier", IEEE trans. on Computers, Vol. EC-13, pp.14-17, Feb. 1964.

[5] A. Weinberger,"4:2 Carry-save adder module", IBM Tech. Discl. Bulletin, 23, 1981.

[6] Y. L. and D. B. Roberts,"A High-Performance and Low-Power 32-bit Multiply Accumulate Unit with Single-Instruction-Multiple-Data (SIMD) Feature", IEEE Journal of Solid-State Circuits, Vol.37, No.7, pp. 926–931, 2002.