低演算精度の計算における電力削減手法

山口, 誠一朗
九州大学大学院システム情報科学府

室山, 真徳
九州大学システムLSI 研究センター

樽見, 幸祐
九州大学大学院システム情報科学府

安浦, 寛人
九州大学大学院システム情報科学研究院

出版情報：DAシンポジウム2005, pp.267-272, 2005-08
バージョン：
権利関係：

低演算精度の計算における電力削減手法

山口 誠一朗 † 室山 真徳 ‡ 樽見 幸祐 † 安浦 寛人 ‡†

† 九州大学 大学院システム情報科学府 〒 816-8580 福岡県春日市春日公園 6-1
‡ 九州大学 システム LSI 研究センター 〒 814-0001 福岡県福岡市早良区百道浜 3-8-33
‡† 九州大学 大学院システム情報科学研究院 〒 816-8580 福岡県春日市春日公園 6-1

E-mail: †{seiichirou, tarumi}@c.csce.kyushu-u.ac.jp, ‡muroyama@slrc.kyushu-u.ac.jp,
‡†yasuura@c.csce.kyushu-u.ac.jp

概要 計算量が多いシステムにおいて，算術演算回路の電力を削減することは重要である．一般にプロセッサはデータパス幅が固定されているため，求められる演算精度にかかわらず常に一定の演算精度のもとで計算が実行される．低い演算精度の計算を実行する場合には上位ビットに符号ビットが挿入されている．この符号部分の計算においても電力を消費するため，低い演算精度の計算を頻繁に実行する場合には符号部分の電力消費が大きくなることがある．本稿では，演算精度を保ちつつ計算実行前にデータを上位にシフトし，計算実行後にデータを下位にシフトすることで，計算実行時の符号部分における無駄な電力を削減する手法を提案する．また，シフトすることにより発生するオーバーヘッドについても検討する．

A Power Reduction Method for Low-Precision Computation

Seiichiro YAMAGUCHI†, Masanori MUROYAMA‡, Kosuke TARUMI† and Hiroto YASUURA†

† Graduate School of Information Science and Electrical Engineering, Kyushu University
6-1 Kasuga-koen, Kasuga, Fukuoka 816-8580 JAPAN
‡ System LSI Research Center, Kyushu University
3-8-33 Momochihama, Sawara-ku, Fukuoka 814-0001 JAPAN

E-mail: †{seiichirou, tarumi, yasuura}@c.csce.kyushu-u.ac.jp, ‡muroyama@slrc.kyushu-u.ac.jp

Abstract To reduce power consumption of arithmetic circuits is an important issue. In general, datapath width of a processor is fixed. Therefore, each computation is always done regardless of the required computational precision. A sign bit is extended for low-precison computation. Since the sign extension part consumes power, we cannot ignore the power consumption of the sign extension part if low-precision computation is dominant. In this paper, we propose a power reduction method which reduces redundant power consumption due to computations of the sign extension part by using shift operations in a case of low-precision computations. We also discuss overheads of the additional shift operations.

1 はじめに

携帯電話や PDA などのバッテリー駆動の携帯機器では，バッテリー持続時間の延長という要求から LSI の低消費電力化が求められている．一方で，発熱の問題もある．発熱が高ければ，高価なセラミックパッケージを使用しなければならなくなりコストがかかる．また，携帯機器をはじめとする多くの組み込みシステムにおいて，ファンなどの冷却装置を設置することは難しい．このような背景から，LSI の低消費電力化は重要である．

LSI の消費電力は，ダイナミック消費電力およびスタティック消費電力に分類できる．最新のプロセッサでは，全体の消費電力の 70%がダイナミック消費電力で占められている [1]．ダイナミック消費電力は，信号遷移確率，出力負荷容量，電源電圧の二乗，および動作周波数に比例する．従って，それぞれのパラメータ値を削減することで電力を削減できる．それぞれのパラメータ値を削減する様々な研究が行われているが [2][3]，本稿では信号遷移確率に着目する．また，電力削減の対象はプロセッサとする．

近年，携帯機器で写真や動画像といったマルチメディアアプリケーションを利用する機会が増えてきている．このような計算量が多いシステムにおいて，算術演算回路の電力を削減することは重要な課題である．一般にプロセッサはデータパス幅が固定されているため，求められる演算精度にかかわらず常に一定の演算精度のもとで計算が実行される．低い演算精度の計算を実行する場合，データは符号拡張されているが，この符号部分の計算においても信号が

遷移する場合もあり，電力を消費してしまう．本稿では，演算精度を保ちつつ計算実行前にデータを上位にシフトし，計算実行後にデータを下位にシフトすることで，計算実行時の符号部分における無駄な電力を削減する手法を提案する．また，シフトすることにより発生するオーバーヘッドについても検討する．

本稿の構成は以下の通りである．第 2 章で準備として，本稿で使用する消費電力のモデルを示し，有効ビット幅 [5][6] という概念について述べる．第 3 章では，電力削減手法の基本アイデアと実現するための手順について述べ，第 4 章で，予備実験の結果を示し効果見積もりを行う．最後に第 5 章で本稿をまとめる．

2 準備

2.1 消費電力モデル

CMOS 回路の消費電力は式 (1) に表されるように 3 つの成分に分類できる．

$$P_{total} = P_{sw} + P_{sc} + P_l \quad (1)$$

ここで，

- P_{total}：CMOS 回路全体の消費電力
- P_{sw}：出力負荷容量の充放電による消費電力
- P_{sc}：貫通電流による消費電力
- P_l：リーク電流による消費電力

である．

最新のプロセッサでは，スタティック消費電力である P_l は P_{total} の 30%程度である [1]．また，スイッチングによる消費電力は P_{sw}，および P_{sc} の和であるが，P_{sc} は最大でも P_{total} の 20%以下である [4]．本稿では P_{sw} に着目する．P_{sw} は式 (2) で表される．

$$P_{sw} = \sum_{k=1}^{N_g} Swit_k \cdot C_{L_k} \cdot V_{dd}^2 \cdot f \quad (2)$$

ここで，

- $Swit_k$：論理ゲート k の 1 クロックサイクル当たりの平均信号遷移確率
- C_{L_k}：論理ゲート k の出力負荷容量
- V_{dd}：電源電圧
- f：クロック周波数
- N_g：CMOS 回路に存在する全論理ゲート数

である．

各パラメータに着目した低消費電力化の研究が数多く行われているが [2][3]，本稿では $Swit_k$ に着目する．

2.2 有効ビット幅

一般に，プロセッサのデータパス幅は 8 ビット，16 ビット，32 ビット，および 64 ビットなどが多い．設計者は用途に応じて最適なデータパス幅を選択する．一方で，アプリケーションプログラム中の各変数が実行時に最低限必要なビット幅は異なる．本稿では，このビット幅を有効ビット幅と呼ぶ [5][6]．unsigned 整数 x に代入され得る値の最大値が n_{max} の時，x の有効ビット幅 $e(x)$ は，式 (3) となる．

$$e(x) = \lceil \log_2(n_{max} + 1) \rceil \quad (3)$$

また，signed 整数 $x^{'}$ に代入され得る値の最小値が $n^{'}_{min}$，最大値が $n^{'}_{max}$ の時，$x^{'}$ の有効ビット幅 $e(x^{'})$ は式 (4) となる．

$$e(x^{'}) = \lceil \log_2 \mathcal{N} \rceil + 1 \quad (4)$$

ここで，MAX$(a, b, \cdots)$ は引数の最大値を返す関数であるとすると，

$$\mathcal{N} = \mathrm{MAX}(|n^{'}_{max}| + 1, |n^{'}_{min}|) \quad (5)$$

である．図 1 のように，あるプログラムにおいて signed int 型 (16 ビットとする) で宣言された変数 x の値域が [-1000,1000] であると仮定する．式 (4)，および式 (5) より，$e(x)$ は 11 ビットである．

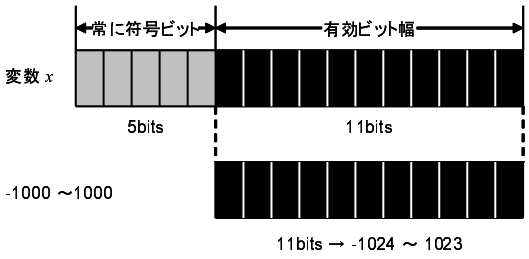

図 1: 変数の有効ビット幅

アプリケーションプログラム中の各変数の有効ビット幅を解析する技術はいくつか提案されている

[5][6][7][8] ．表 1 は，MPEG-2 video decoder のプログラムにおける変数と有効ビット幅の関係を示している [9]．MPEG-2 video decoder のプログラムでは，384 種の変数が signed int 型 (32 ビット) で宣言されている．表 1 より，多くの変数において有効ビット幅は小さく，それらの計算において要求される演算精度は低いということになる．

表 1: MPEG-2 video decoder における変数と有効ビット幅の関係

有効ビット幅	変数の数	有効ビット幅	変数の数
1bit	50	17bits	39
2bits	17	18bits	3
3bits	11	19bits	6
4bits	11	20bits	0
5bits	10	21bits	6
6bits	14	22bits	0
7bits	16	23bits	0
8bits	9	24bits	14
9bits	7	25bits	0
10bits	3	26bits	2
11bits	6	27bits	4
12bits	17	28bits	3
13bits	0	29bits	3
14bits	46	30bits	7
15bits	2	31bits	0
16bits	39	32bits	82

各変数の有効ビット幅の情報を利用した研究の一つとしてソフトコアプロセッサの研究がある [10][11]．ソフトコアプロセッサはアプリケーションプログラムに対して，性能，コスト，および消費電力などの面で最適なデータパス幅を選択することができる．

3 低演算精度の計算における電力削減手法

本章では，電力削減手法の基本アイデアと実現するための手順について述べる．第 2 章で，アプリケーションプログラムの各変数の有効ビット幅の情報をもとにプロセッサのデータパス幅を最適に選択することで，プロセッサの性能向上や低消費電力化を実現するソフトコアプロセッサの研究があることを述べた．しかし，さまざまなアプリケーションプログラムを実行する必要がある状況下では，プロセッサのデータパス幅をある一つのアプリケーションプログラムに最適な幅にしてしまうと，他のアプリケーションプログラムを実行する場合に性能低下などを引き起こす可能性がある．そこで本稿では，プロセッサのデータパス幅を変更するのではなく，有効ビット幅の情報をもとにデータを上位にシフトすることで計算実行時の信号遷移を削減し，電力を削減する手法を提案する．

3.1 基本アイデア

従来の計算方法では，低い演算精度の計算を実行する場合，signed データは符号拡張され上位ビットに符号ビットが挿入されている．しかし，符号部分の計算においても信号が遷移する場合もあり，電力を消費する．そこで，低い演算精度の計算を実行する際は，計算実行前に入力データを上位にシフトし，計算実行後に出力データを下位にシフトすることにより，計算実行時の符号部分の信号遷移を削減する．図 2 に従来の計算方法と提案手法の計算方法を示す．図 2 ではデータパス幅は 8 ビット，入力データの有効ビット幅は全て 4 ビットとしている．図 2 のような計算においては，Add 8(8 ビット加算器) の信号遷移を削減できる．

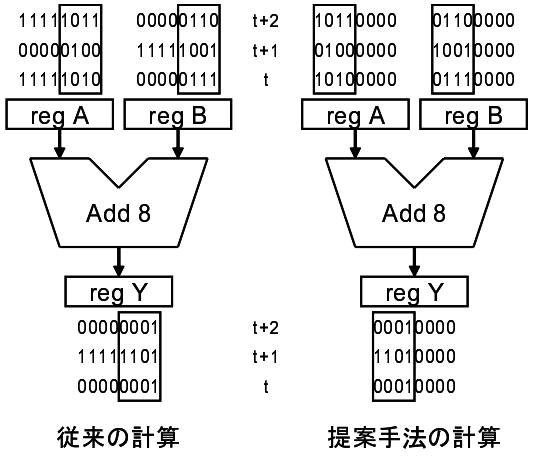

図 2: 従来の計算と提案手法の計算

提案手法の効果が大きいのは，入力データの有効ビット幅が小さい場合，かつ入出力データの正負が前データの正負から反転する場合である．一方で，提案手法の効果が小さいのは，有効ビット幅がデータパス幅に近い場合，または入出力データの正負が前データの正負と同じ場合，または入出力データが unsigned データの場合である．

提案手法の特徴は，ハードウェアには手を加えずソフトウェアの工夫のみで電力を削減することである．これにより，既にチップ化されているプロセッサに対しても提案手法を適用することが可能である．

3.2 手順

提案手法を実現するための手順は以下の三つである．また，有効ビット幅を考慮してデータをシフトする様子を図 3 に示す．

手順 1． 有効ビット幅解析

アプリケーションプログラムの各変数 $\{x_1, x_2, \cdots, x_i, \cdots, x_n\}$ の有効ビット幅 $\{e(x_1), e(x_2), \cdots, e(x_i), \cdots, e(x_n)\}$ を解析する．

手順 2． シフト命令の追加

アプリケーションプログラムのコンパイル時に，計算 j における入出力変数 x_a, x_b, x_y の有効ビット幅の最大値 E_j=MAX$(e(x_a), e(x_b), e(x_y))$ を求め，計算実行前に入力データを $(D_p\text{-}E_j)$ ビット論理左シフトし，計算実行後に出力データを $(D_p\text{-}E_j)$ ビット算術 (または論理) 右シフトする命令を追加する．ここで，D_p はデータパス幅である．また，計算実行後の右シフトは出力データが signed であれば算術右シフトを行い，unsigned であれば論理右シフトを行う．

手順 3． シフト命令の削減

ある計算 j の出力データが別の計算 j' の入力データとして利用される場合，計算実行後の算術 (または論理) 右シフトと計算実行前の論理左シフトを相殺し，シフト命令の数を削減する．具体的には以下の 3 通りで削減する．

- $E_j > E_{j'}$ の時 $(E_j\text{-}E_{j'})$ ビット論理左シフトする命令に変更する．
- $E_j < E_{j'}$ の時 $(E_{j'}\text{-}E_j)$ ビット算術 (または論理) 右シフトする命令に変更する．
- $E_j = E_{j'}$ の時シフト命令を削除する．

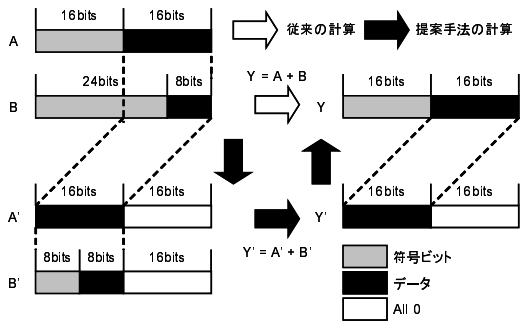

図 3: 有効ビット幅に合わせてデータをシフト

4 評価

本章では，予備実験の結果を示し電力削減の効果見積もりを行う．また，シフトすることによるオーバーヘッドについても検討する．

4.1 予備実験

各変数のデータを上位にシフトすることが，算術演算回路の消費電力にどの程度影響があるのか調査を行った．実験に使用した算術演算回路は 16 ビット Carry Look-ahead Adder[12](以下，CLA)，および 16 ビット Wallace-Tree Multiplier[13](以下，WALLACE) である．WALLACE は Booth デコーダ付きで，最終加算は CLA である．また，シフトによる消費電力のオーバーヘッドを調査するため，16 ビットのバレルシフタにおける消費電力も解析した．このバレルシフタは LSL(Logical Shift Left)，LSR(Logical Shift Right)，および ASR(Arithmetic Shift Right) の 3 機能を実現する．なお，今回の実験では HITACHI0.18μm のセルライブラリを用いた．

Cadence 社の Verilog-XL を用いて，従来の方法で計算を実行した場合と，提案手法で計算を実行した場合の信号遷移回数を求め，Synopsys 社の Design-Power を用いて消費電力を解析した．算術演算回路に入力するデータはランダムな値とし，5000 パタン与えた．また，データの有効ビット幅は 5000 パタン全て等しいと仮定した．ただし，提案手法の計算においてデータは既に上位にシフトされている状態であると仮定し，消費電力を解析した．つまり，有効ビット幅が 8 ビットの時，5000 パタン全てのデータにおいて，従来の計算では上位 8 ビットは常に符号拡張されており，提案手法の計算では下位 8 ビットは常に 0 となっている．またバレルシフタ回路でも同様の条件で実験を行った．

図 4，および図 5 に CLA，および WALLACE における有効ビット幅と消費電力の関係を示す．また，図 6 にバレルシフタにおけるシフト幅と消費電力の関係を示す． データの有効ビット幅が 8 ビットの時，図 4 においては約 50%電力削減できている．また，図 5 においては約 25%電力削減できている．ただし，実際には提案手法の計算においてシフトによるオーバーヘッドが存在するため，全体としての削減率は低下する．図 6 においてはシフト幅が大きくなるにつれ消費電力も低くなっている．以上の予備実験から，有効ビット幅が小さく，かつ有効ビット

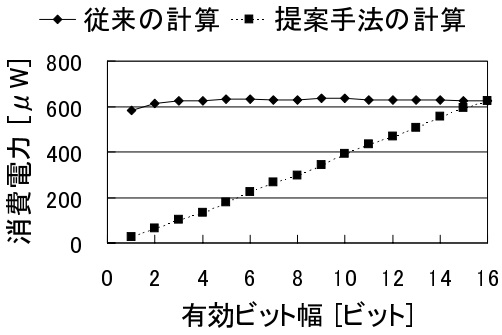

図 4: CLA における有効ビット幅と消費電力の関係

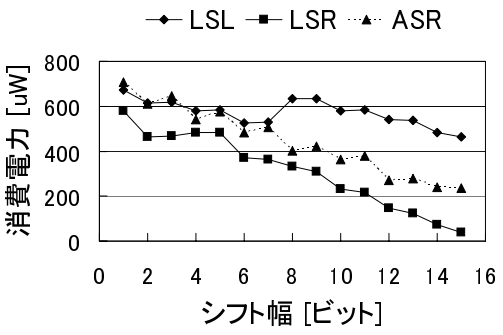

図 6: バレルシフタにおけるシフト幅と消費電力の関係

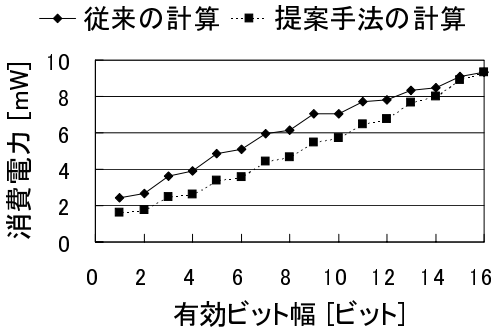

図 5: WALLACE における有効ビット幅と消費電力の関係

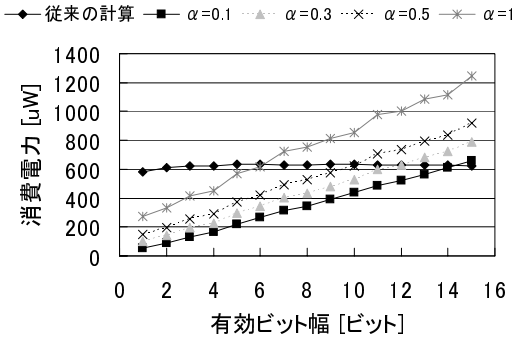

図 7: 効果見積もり

幅が同じである変数を連続して計算するような場合には，シフトによるオーバーヘッドも少なく，電力を削減できると推測される．

4.2 効果見積もり

ここでは提案手法を適用した場合の電力削減効果の見積もりを行う．ただし，以下を仮定する．

- 各変数の有効ビット幅は一定である
- データはランダムな signed データである

この仮定により，予備実験で解析した消費電力をモデルとして使用して効果見積もりを行うことができる．

提案手法により追加されたシフト命令が，一回の加算命令に対して出現する確率を $\alpha(0\leq \alpha \leq 1)$ とおく．提案手法の効果が大きいのは，実現手順 3 によりシフト命令が削除され，α が小さい時であり，データを上位に格納したまま何度も計算を実行する場合である．α が 0.1，0.3，0.5，1 の場合における加算器の消費電力のグラフを図 7 に示す．図 7 より，α が小さい時や，有効ビット幅が小さい時に効果があることがわかる．ただし，提案手法適用後の消費電力は式 (6) により求めた．

$$P'_{add} \quad = \quad P_{cla_e} + \alpha \cdot P_{shift_e} \qquad (6)$$

ここで，

- P'_{add}：提案手法適用後の加算による消費電力
- P_{cla_e}：有効ビット幅が e ビットの場合の図 4 における提案手法の消費電力
- P_{shift_e}：シフト幅が $(16-e)$ ビットの場合の図 6 における各シフトの消費電力の平均

である．提案手法適用前の加算器による消費電力 P_{add} の割合が全体の $\beta(0\leq \beta \leq 1)$ である時，プロセッサ全体での電力削減効果 Δ は式 (7) を利用して求めることができる．

$$\Delta \quad = \quad \frac{P_{add} - P'_{add}}{P_{add}} \cdot \beta \qquad (7)$$

e=8，α=0.1，および β=0.4 の時，Δ はおよそ 0.23 である．従って，プロセッサ全体の 23%の電力を削減できることになる．

5 おわりに

本稿では，低い演算精度の計算における電力削減手法に関して基本アイデアを紹介した．また，予備実験の結果をもとに電力削減の効果見積もりを行った．アプリケーションプログラムの加算に用いられる変数の有効ビット幅が 8 ビット，提案手法により追加されたシフト命令が一回の加算命令に対して出現する確率が 10%，加算器による電力の割合がプロセッサ全体の電力の 40%の場合においては，プロセッサ全体で 23%の電力削減効果があることを示した．提案手法の特徴は，ハードウェアには手を加えずソフトウェアの工夫のみで電力を削減することである．これにより，既にチップ化されているプロセッサに対しても提案手法を適用することが可能である．今後の課題は，実アプリケーションプログラムを用いて提案手法の効果を確認すること，および提案手法を実現するためのシフト命令の挿入や，シフト命令の削減を実現するコンパイラを開発することである．

謝辞

本研究は，一部科学研究費補助金 (学術創成研究費 (2))(課題番号:14GS0218) による．また，本研究は東京大学大規模集積システム設計教育研究センターを通し，株式会社日立製作所，ケイデンス株式会社，およびシノプシス株式会社の協力で行われたものである．

参考文献

[1] J. Schutz and C. Webb, “A Scalable X86 CPU Design for 90nm Process,” In Proceedings of the 2004 IEEE International Solid-State Circuits Conference, pp.62-63, 2004.

[2] Luca Benini and Giovanni de Micheli, “System-Level Power Optimization : Techniques and Tools,” ACM Transactions on Design Automation of Electronic Systems, Vol. 5, No. 2, pp.115-192, Apr. 2000.

[3] Massoud Pedram and Jan M. Rabaey, “Power Aware Design Methodologies,” Kluwer Academic Publishers, 2002.

[4] Koichi Nose and Takayasu Sakurai, “Analysis and Future Trend of Short-Circuit Power,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 19, No. 9, Sep. 2000.

[5] Hajime Yamashita, Hiroyuki Tomiyama, Akihiko Inoue, Eko Fajar Nurprasetyo, Takanori Okuma and Hiroto Yasuura, “Variable Size Analysis for Datapath Width Optimization,” In Proceedings of the 5th Asian Pacific Conference on Hardware Description Languages, pp.69-74, Jul. 1998.

[6] Hajime Yamashita, Hiroto Yasuura, Fajar N. Eko and Cao Yun, “Variable Size Analysis and Validation of Computation Quality,” In Proceedings of the 2000 IEEE International High Level Design Validation and Test Workshop, pp.95-100, Nov. 2000.

[7] Scott Mahlke, Rajiv Ravindran, Michael Schlansker, Robert Schreiber and Timothy Sherwood, “Bitwidth Cognizant Architecture Synthesis of Custom Hardware Accelerators,” IEEE Transactions on Computer-Aided Design of Integrated Ciucuits and Systems, Vol. 20, No. 11, Nov. 2001.

[8] Mark Stephenson, Jonathan Babb and Saman Amarasinghe, “Bitwidth Analysis with Application to Silicon Compilation,” In Proceedings of the ACM SIGPLAN 2000 Conference on Programming Language design and Implementation, pp.108-120, Jun. 2000.

[9] Hiroyuki Tomiyama, Yun Cao, Uddin Mesbah, Akihiko Inoue, Eko Fajar, Hajime Yamashita and Hiroto Yasuura, “A Framework for Bitwidth Optimization in System-on-Chip Design,” In Proceedings of Workshop on Application Specific Processors, Nov. 2002.

[10] H. Yasuura, H. Tomiyama, A. Inoue and F. N. Eko, “Embedded System Design Using Soft-Core Processor and Valen-C,” Journal of Information Science and Engineering, Vol. 14, No. 3, pp.587-603, Sep. 1998.

[11] F. N. Eko, A. Inoue, H. Tomiyama and H. Yasuura, “Soft-Core Processor Architecture for Embedded System Design,” IEICE Transactions on Electronics, Vol. E81-C, No. 9, PP.1416-1423, Sep. 1998.

[12] A. Weinberger and J. L. Smith, “A logic for high-speed addition,” National Bureau of Standards Circular, 591, pp.3-12, 1958.

[13] C. S. Wallace, “A suggestion for a fast multiplier,” IEEE Transactions on Electronic Computers, Vol. EC-13, No. 1, pp.14-17, Feb. 1964.