計算機アーキテクチャ 144−11
（２００１．７．２６）

シングルチップマルチプロセッサ上でのマルチメディアアプリケーションの近細粒度並列処理

小高 剛　宮下直久
木村啓二　笠原博徳

早稲田大学理工学部電気電子情報工学科

〒169-8555 東京都新宿区大久保 3-4-1 TEL:03-5286-3371
E-mail: {kodaka,naohisa,kimura,kasahara}@oscar.elec.waseda.ac.jp

近年のマルチメディアコンテンツの増加に伴い，JPEG，MPEG などのメディア系アプリケーションを効率良く処理できる，低コストかつ低消費電力のプロセッサの開発が望まれている．これらの要求を満たすプロセッサとして，簡素なプロセッサコアを複数搭載したシングルチップマルチプロセッサが注目を集めている．本稿では，シングルチップマルチプロセッサのメディア系アプリケーションでの有用性を確かめるため，まず，第一段階として画像圧縮処理の JPEG エンコーディングプログラムを用い，その処理単位が最終的に 8×8 画素のブロックになることに注目し，その 8×8 画素ブロックの処理に近細粒度並列処理を施し OSCAR 型シングルチップマルチプロセッサ上で性能評価を行った．その結果，シンプルなシングルイシュープロセッサコアを 4 基搭載した OSCAR 型シングルチップマルチプロセッサシステムは 4 イシュースーパースカラプロセッサの UltraSPARC-II 相当のプロセッサコアを 1 基搭載するシステムに対し約 2.32 倍の速度向上が得られた．

Near Fine Grain Parallel Processing on Multimedia Application for Single Chip Multiprocessor

Takeshi Kodaka, Naohisa Miyashita, Keiji Kimura and Hironori Kasahara

Dept. of Electrical, Electronics and Computer Engineering, Waseda University

3-4-1 Ohkubo Shinjuku-ku, Tokyo 169-8555, Japan Tel: +81-3-5286-3371
E-mail: {kodaka,naohisa,kimura,kasahara}@oscar.elec.waseda.ac.jp

With the recent increase of multimedia contents, such as JPEG and MPEG data, low cost and low power consumption processors that can process these multimedia contents efficiently are expected. In such microprocessors, single chip multiprocessor architecture having simple processor cores is attracting much attention. Considering the above facts, this paper evaluate a JPEG encoding program on OSCAR type single chip multiprocessor architecture using near fine grain parallel processing for 8×8 pixel block that is a fundamental part of JPEG algorithm. The evaluation shows an OSCAR type single chip multiprocessor having four single-issue simple processor cores gives 2.32 times speedup than four-issue UltraSPARC-II type super-scaler processor.

1　はじめに

近年のマルチメディアコンテンツの増加に従い，JPEG，MPEG などのメディア系アプリケーションを効率良く処理できる，低コストかつ低消費電力のプロセッサの開発が望まれている．このようなニーズに対応するために CPU ベンダーからは命令セットアーキテクチャにメディア用命令セットを追加しメディア用処理能力を向上させる試みが行われている [1]．

また，上記の要求を満たす別のアプローチとして，簡素なプロセッサコアを複数搭載したシングルチップマルチプロセッサが注目を集めており，シングルチップマルチプロセッサを用いてメディア系アプリケーションの処理の高速化を行う試みもなされている．[2)~5)]

本稿ではマルチメディアアプリケーションのシングルチップマルチプロセッサ上での性能を評価するために，まず，第一段階として画像圧縮処理の JPEG エンコーディングを用い，その処理単位が最終的に 8×8 画素のブロックになることに注目し，その 8×8 画素ブロックの処理に近細粒度並列処理を施し OSCAR 型シングルチップマルチプロセッサ [7)] 上で性能を評価した．

以降，2節にて近細粒度並列処理，3節にて評価プログラムの JPEG エンコーディングアルゴリズム，4節にて評価対象アーキテクチャついて説明した後，5節にて OSCAR型シングルチップマルチプロセッサ上で行った性能評価について述べる．

2　近細粒度並列処理 [6)]

近細粒度並列処理とは基本ブロック内のステートメント間の並列性を利用する並列処理である．ステートメントをプロセッサエレメント (PE) に割り当てる際には，スケジューリング手法としてデータ転送オーバヘッドを考慮し実行時間を最小化するヒューリスティックアルゴリズムである CP/DT/MISF 法，CP/ETF/MISF 法，ETF/CP 法，あるいは DT/CP 法 [8)] の 4 手法を適用し最良のスケジュールを選ぶ．

3　JPEG エンコーディングアルゴリズム

本稿では，JPEG エンコーディングアルゴリズムとして整数演算ベースラインシステムについて考える．

ベースラインシステムの JPEG エンコードアルゴリズムは，入力信号を輝度と色差成分信号に変換する YCbCr 変換処理，画像信号を空間周波数成分に変換する離散コサイン変換，ベクトル量子化を行う量子化処理，エントロピー符号化を行う可変長符号化処理の 4 つの処理から構成される．また，これらの処理は，8×8 画素からなるブロックを単位として行われる．

本節では，評価に用いたベースラインシステムの JPEG エンコードアルゴリズムについて説明する．

3.1　YCbCr 変換

YCbCr 変換は，入力信号を JPEG 処理系の輝度と色差の成分に変換する処理である．入力信号により変換式は変化するが本稿では一般的な R(赤)，G(緑)，B(青) の三原色からの変換について述べる．

RGB から YCbCr 変換は

$$Y_{ij} = R_{ij} * 0.2990 + G_{ij} * 0.5870 + B_{ij} * 0.1140$$
$$Cb_{ij} = -R_{ij} * 0.1684 - G_{ij} * 0.3316 + B_{ij} * 0.5000$$
$$Cr_{ij} = R_{ij} * 0.5000 - G_{ij} * 0.4187 - B_{ij} * 0.0813$$

で表される．

この式より，変換式には各演算間にデータの依存が存在せずこれらの演算は並列に処理できる．

3.2　離散コサイン変換

離散コサイン変換 (DCT) は，データを周波数成分に変換する直交変換の 1 つである．現在 DCT は，画像符号化の基本技術となっており多くの符号化標準規格で採用されている．

なお，ここでは 1 次元 DCT のアルゴリズムについて説明するが，2 次元への拡張は，1 次元 DCT を行方向に対して 1 回処理を行った後，列方向へ 1 回処理を行うことにより容易に実現できる．

3.2.1　基本式

DCT の基本式は，入力データ列 $f(u)$:$(u = 0..N)$ に対する DCT 出力を $F(u)$ とするとき

$$F(u) = \sqrt{\frac{2}{N}} C(u) \sum_{x=0}^{N-1} f(x) \cos \frac{(2x+1)u\pi}{2N}$$
$$C(u) = \begin{cases} 1/\sqrt{2} & (u = 0) \\ 1 & (u \neq 0) \end{cases}$$

と表される．この場合，計算量は，加算・乗算ともに N^2 となり，計算量が大きく実用的ではないため計算量をおさえるために次節に述べるようなバタフライ演算法が広く利用されている．

3.2.2　バタフライ演算法

計算量をおさえるため，FFT(高速フーリエ変換) のようなバタフライ演算による高速演算法が広く利用されている．代表的なアルゴリズムに Chen のアルゴリズム [10)] があり次式で表される．

$$A_N = P_N \begin{bmatrix} A_{N/2} & 0 \\ 0 & R_{N/2} \end{bmatrix} \begin{bmatrix} I_{N/2} & \tilde{I}_{N/2} \\ \tilde{I}_{N/2} & -I_{N/2} \end{bmatrix}$$

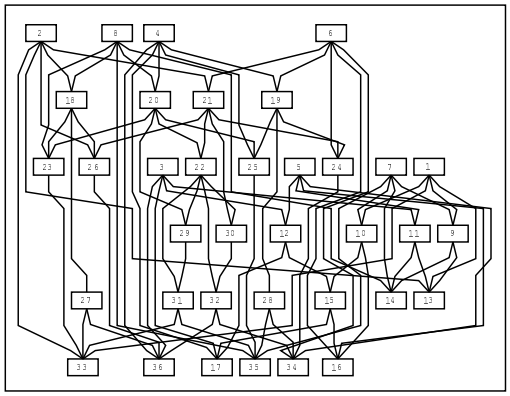
P_N　　巡回行列

R_N　　$\tilde{R}_N$ の要素 $\tilde{r}_N(i,k)$ を
行と列について逆に並べた行列

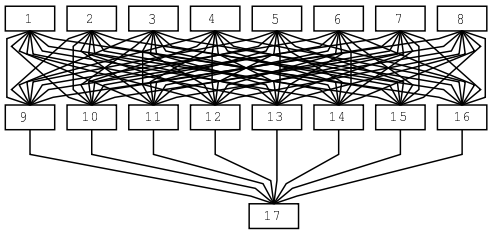
この方法により，計算量が，加算:$(3N/2)(\log_2 N - 1) + 2$, 乗算: $N \log_2 N - 3N/2 + 4$ に削減される．

本稿の評価では，離散コサイン変換処理にバタフライ演算法を用いた．バタフライ演算のタスクグラフを図 1 に示す．図中 (a) は，1 次元 DCT 演算の近細粒度タスクグラフを表している．また (b) は，2 次元 DCT 処理を行うときの各行・列処理のデータ依存を示すグラフであり各ノードは (a) のタスクグラフを表している．

図 1(a) に示したようにバタフライ演算法では，1 次元演算において並列処理できる部分が存在する．



(a) 1-Dimension DCT



(b) 8x8 block DCT

図 1: DCT タスクグラフ

また，2 次元処理では図 1(b) のように 1 次元処理を各行で行方向に一回，各列で列方向に一回処理を行うことで実現させるため，行方向処理中では各行処理の間でのデータ依存が無く並列に実行できる．同様に列方向についても同様にデータ依存がないため並列に実行できる．

3.3 量子化

量子化処理は，DCT 処理後の値を予め定められた量子化ステップサイズに対する比率を整数値で求めることであり次式で表される．

$$Sq_{ij} = round\left(\frac{S_{ij}}{q_{ij}}\right)$$

$$Q = \begin{pmatrix} q_{11} & q_{12} & \cdots & q_{1j} \\ q_{21} & q_{22} & \cdots & q_{2j} \\ \cdots & \cdots & \cdots & \cdots \\ q_{i1} & q_{i2} & \cdots & q_{ij} \end{pmatrix}$$

ここで Sq_{ij} は量子化値，S_{ij} は入力となる DCT 係数，q_{ij} は量子化ステップサイズを示し，round() 演算は，四捨五入を行う演算子である．

本稿の評価で用いた量子化テーブルは MediaBench[9] に含まれている JPEG エンコーディングプログラムにおけるデフォルト値を用いる．

量子化演算では，式中の各要素が一意に決まるため各演算間にデータ依存が存在せず，これらの演算は並列に処理できる．

3.4 可変長符号化

量子化後，エントロピー符号化のため可変長符号化処理が行われる．ベースラインシステムでの可変長符号化は，8×8 の量子化値に対して DC 成分と AC 成分に別けて行われる．

3.4.1 DC 成分の可変長符号化

DC 成分とは，入力となる 8×8 の量子化値中の最左上の値，すなわち座標 (0,0) の値を示す．DC 成分の可変長符号化には予測が用いられ，現在の 8×8 の量子化値の DC 成分に対し隣接する直前の 8×8 の量子化値の DC 成分との差分をとりその差分値が符号化される．

DC 成分の可変長符号化を行うには，上記 DC 差分値の絶対値において MSB 側から見て始めにビット"1" が立つ場所が LSB 側から数えて何ビット目であるかという数値 $SSSS$ を求め，$SSSS$ に対応する可変長符号を求める．次に，DC 差分値が正のときは DC 差分値の LSB から $SSSS$ ビット目までを，負のときは (DC 差分値 -1) の LSB から $SSSS$ ビットまでを付加ビットとして，(可変長符号 + 付加ビット) の形で可変長符号化を行う．

3.4.2 AC 成分の可変長符号化

AC 成分とは，8×8 の量子化値の DC 成分以外の 63 個の値を示す．AC 成分は，ジグザグスキャン順にスキャニングされ符号化される．

AC 成分の可変長符号化は，ジグザグスキャン順に 0 係数の連続出現回数 (RUN) と非 0 係数の値により行われる．まず，スキャン順に非 0 係数が出るまで RUN 値をカウントする．ただし RUN 値が 16 になった場合は，ZRL 符号 (0 が 16 個続いたことを示す符号) を出力し RUN 値を 0 にリセットし RUN 値のカウントを再開する．非 0 係数が現れたらその非 0 係数に対して DC 成分の符号化と同様に $SSSS$ 値を求め，非 0 係数が現れるまでの RUN 値と $SSSS$ 値を用いて 2 次元可変長符号テーブルから対応する可変長符号を求める．そして，非 0 係数が正のときは非 0 係数の LSB から $SSSS$ ビットを，負のときは (非 0 係数 -1) の LSB から $SSSS$ ビットを付加ビットとし，(可変長符号 + 付加ビット) の形で可変長符号化を行う．

本稿では，可変長符号テーブルは MediaBench に含まれている JPEG エンコーディングプログラムにおけるデフォルト値を用いる．

上記のように，DC 成分の符号化には 8×8 ブロック間の依存が，AC 成分の符号化には非 0 係数の判定および直前までの 0 係数の個数が分からなければ符号化できないので，8×8 ブロック内で制御依存およびデータ依存が存在し，そのため並列処理が難しい．

本稿の評価では，並列性抽出のため AC 成分の可変長符号化処理を PE 台数分に分割し処理分割に伴い分割境界部分の RUN 値が分離されてしまうため符号化の最後に値を修正するための処理を追加している．

ここでは簡単のために分割数を 2 とし，スキャニング順で始めの 2〜32 までを分割ブロック 0，後の 33〜64 を分割ブロック 1 とし説明する (AC 成分は DC 成分を抜いた 2〜64 までとなる)．

分割ブロック 0 では，分割しない場合と同様の処理が行われる．分割ブロック 1 では，非 0 係数を可変長符号化するときに，その非 0 係数の可変長符号化が分割ブロック内で初めてかどうか判定する．もし，その分割ブロック内で初めての可変長符号化を行う非 0 係数であった場合，そのときの RUN 値と非 0 係数の位置および非 0 係数の $SSSS$ 値を保存しておく．

分割ブロック 0，1 での可変長符号化が終了したら，符号値を修正するために，分割ブロック 0 での最後の RUN 値と保存した分割ブロック 1 で初めに出現した非 0 係数までの RUN 値との和を取る．この和をとった RUN 値と保存しておいた分割ブロック 1 で初めに出現した非 0 係数の $SSSS$ 値をもとに可変長符号を求め可変長符号化を行い分割ブロック 1 における初めの非 0 係数の位置に求めた可変長符号を上書する．

以上が，分割数 2 での可変長符号化である．分割数を増やすには同様に分割ブロック内での最初の非 0 係数出現までの RUN 値と非 0 係数の位置および非 0 係数の $SSSS$ 値を保存し，各分割ブロックの可変長符号化処理が終了したら前の分割ブロックの最後の RUN 値の和を取りその和を取った RUN 値で可変長符号を求め直して符号を作り直す部分を増やすことにより対応できる．

4 対象アーキテクチャ

本節では評価対象アーキテクチャである OSCAR 型シングルチップマルチプロセッサのアーキテクチャについて述べる．また，比較対象とする UltraSPARC-II 相当プロセッサを CPU コアとするアーキテクチャについても述べる．

4.1 OSCAR 型シングルチップマルチプロセッサアーキテクチャ

OSCAR 型アーキテクチャ [7] によるシングルチップマルチプロセッサの構成を図 2 に示す．OSCAR 型アーキテクチャでは，CPU，データ転送ユニット (DTU)，ローカルプログラムメモリ (LPM)，ローカルデータメモリ (LDM)，そしてデュアルポートメモリで構成された分散共有メモリ (DSM) をもつプロセッシングエレメント (PE) と，集中共有メモリ (CSM) が相互接続網を介して接続されている．

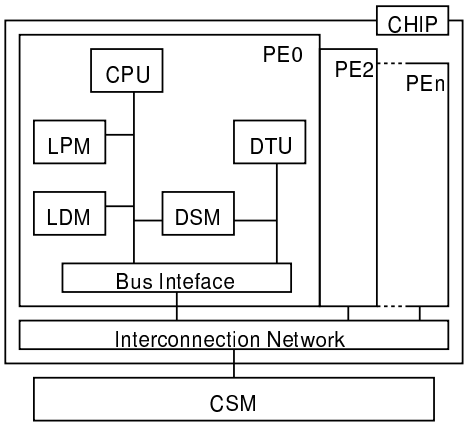


図 2: OSCAR 型 SCM アーキテクチャ

本評価では各メモリへのアクセスレイテンシを，LPM に対して 1 クロック，LDM に対して 1 クロック，DSM は，自 PE 上の DSM へのアクセスに対して 1 クロック，他 PE 上の DSM へのアクセスに対して 4 クロック，CSM に対して 20 クロックかかるとする．また、相互接続網は 3 本のバスを使用する．

プロセッサコアには，SPARC V9 規格に準拠した表 1 に示す単一命令発行のシンプルな RISC プロセッサを使用する．命令レイテンシは，UltraSPARC-II と同様とする．プロセッサ数は 1〜4 基とし，LDM の総容量は，1Mbyte，DSM は 16Kbyte とする．

4.2 比較対象プロセッサコア

比較対象プロセッサコアとして表 1 に示す in-order 4 命令発行アーキテクチャの UltraSPARC-II 相当の演算器構成を持つプロセッサを用いる．本評価では，データサイズが小さく使用するデータが全てキャッシュにのるため，このプロセッサコアを持つシステムでは，メモリアクセスが必ず 1 クロックで可能な理想的なメモリシステムを仮定する．また，命令のレイテンシは 4.1 節で述べた OSCAR 型アーキテクチャと同様とする．

表 1: プロセッサコア仕様

	SCM	US-II Type
パイプライン段数	9	
同時命令発行数	1	4
IEU	1	2
FPU	1	2
LSU	1	1
命令発行タイプ	in-order	

5 性能評価

本節では，JPEG エンコーディングアルゴリズムに近細粒度並列処理を適用して性能評価を行った結果を述べる．

対象アーキテクチャは，4 節で述べたシングルイシュープロセッサコアを持つ OSCAR 型シングルチップマルチプロセッサアーキテクチャ(プロセッサ数:1〜4 基)，および，比較対象とする 4 イシュースーパースカラプロセッサコアの UltraSPARC-II 相当のプロセッサコアを持つアーキテクチャ(プロセッサ数:1 基) である．

評価は，YCbCr 変換，離散コサイン変換，量子化，可変長符号化の個別評価と，YCbCr 変換から可変長符号化までを通して行う全体評価により行い，入力データは，8×8RGB データとし，輝度成分のエンコーディングのみを対象とした．なお，可変長符号化では，3.4 節で述べたように並列性抽出のため PE 台数分分割し後処理を加えたプログラムを使用する．

この結果を図 3 に示す．図 3 では，YCBCr は YCbCr 変換，DCT は離散コサイン変換,Quant は量子化，VLC は可変長符号化をそれぞれ個別評価した結果であり Total は YCbCr 変換から可変長符号化までの全体評価を行った結果である．評価結果は，シングルチップマルチプロセッサアーキテクチャでのシングルイシュープロセッサ 1 基での実行結果に対する速度向上率として表す．

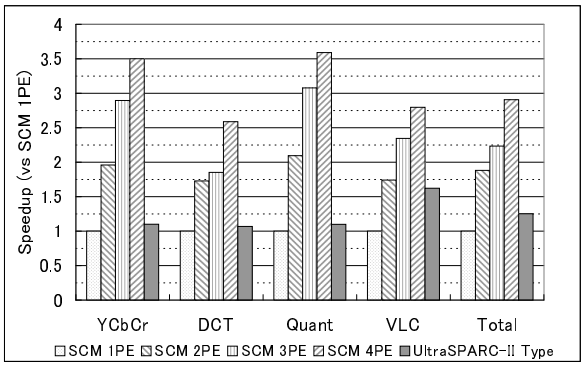


図 3: 評価結果

YCbCr 変換では，ステートメント間のデータ依存が存在しない．そのため，PE 数 4 台にて，PE 数 1 台に対し約 3.50 倍というスケーラブルな速度向上を示した．また，UltraSPARC-II 相当のプロセッサコアに対しても約 3.18 倍の速度向上が得らた．

離散コサイン変換では，PE 数 4 台にて，PE 数 1 台に対し約 2.58 倍，UltraSPARC-II 相当のプロセッサコアに対して約 2.42 倍の速度向上が得られた．アルゴリズム上，行方向処理から列方向処理へ移行するとき約半分のデータが自 PE から他の PE へ転送されることの影響により YCbCr 変換と比較して速度向上率が低くなってしまったと考えられる．

量子化では YCbCr 変換と同様に，ステートメント間のデータ依存が存在しない．そのため，PE 数 4 台にて，PE 数 1 台に対し約 3.59 倍というスケーラブルな速度向上を示した．また，UltraSPARC-II 相当のプロセッサコアに対しても約 3.26 倍の速度向上が得らた．

可変長符号化では，分割処理により並列実行可能となり速度向上を示した．PE 数 4 台にて，PE 数 1 台に対し約 2.79 倍，UltraSPARC-II 相当のプロセッサコアに対して約 1.72 倍の速度向上が得られた．

全体を通して行った性能評価では，PE 数 4 台にて，PE 数 1 台に対し約 2.90 倍，UltraSPARC-II 相当のプロセッサコアに対し約 2.32 倍の速度向上が得らた．

UltraSPARC-II 相当のプロセッサコアを用いた

アーキテクチャの場合速度向上が少ないのは，積和演算が多用されるため命令に乗算，除算のマルチサイクル命令が瀕出するのでパイプラインがストールしIPCが向上しなかったためである．

6 まとめ

本稿では，マルチメディアアプリケーションの画像圧縮処理のひとつであるJPEGエンコーディングについて近細粒度並列処理を適用しOSCAR型シングルチップマルチプロセッサ上での性能評価を行った．その結果，シングルイシュープロセッサを4基搭載したOSCAR型シングルチップマルチプロセッサシステムでは同システムにシングルイシュープロセッサを1基搭載したシステムに対し約2.90倍，4イシューのUltraSPARC-II相当プロセッサコアを1基搭載したシステムに対し約2.32倍の性能が得られた．

これにより，JPEGエンコーディングアルゴリズムにおいて近細粒度並列処理を適用することの有用性が確認できた．

今後の課題として，8×8画素ブロック間の並列性と8×8画素ブロック内の近細粒度並列性を階層的に用いるシングルチップマルチプロセッサ上でのマルチグレイン並列処理やMPEG-2やJPEG2000などに用いられるマルチメディアアルゴリズムでの並列処理を行うことがあげられる．

7 謝辞

本研究の一部は，STARC「自動並列化コンパイラ協調型シングルチップマルチプロセッサの研究」及び経済産業省ミレニアムプロジェクト「アドバンスト並列化コンパイラ」により行われた．本論文作成にあたり，有益なコメントをいただいたSTARC研究員である，STARC小澤時典氏，平田雅規氏，東芝 浅野滋徳氏，富士通 高橋宏政氏，ソニー 倉田隆弘氏，松下 高山秀一氏に感謝致します．

参考文献

[1] S.K.Raman, V.Pentkovski and J.Keshava,Intel Corporation, “Implementing Streaming SIMD Extensions on the Pentium III Processor” ,IEEE MICRO,July-August.2000

[2] M.Edahiro, S.Matsushita and M.Yamashina, N.Nishi, “A Single-Chip Multiprocessor for Smart Terminals”, IEEE MICRO,July-August.2000

[3] L.A.Barroso, K.Gharachorloo, R.McNamara, A.Nowatzyk, S.Qadeer, B.Sano,S.Smith, R.Stets and B.Verghese, “Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing” In Proc. ISCA 00, Jun.2000

[4] P.Sriram, S.Sudharsanan and Amit Gulati, “MPEG-2 Video Decompression on a Multiprocessing VLIW Microprocessor”, Sun Microsystems

[5] E.Iwata and K.Olukotun, “Exploiting Coarse-Grain Parallelism in the MPEG-2 Algorithm”, Stanford University Computer System Lab. Technical Report CSL-TR-98-771, Sep.1998

[6] H.Kasahara, M.Obata and K.Ishizaka, “Automatic Coarse Grain Task Parallel Processing on SMP using OpenMP”, Proc. of 13th International Workshop on Languages and Compilers for Parallel Computing (LCPC'00), Aug.2000.

[7] 木村, 加藤, 笠原, “近細粒度並列処理用シングルチップマルチプロセッサにおける プロセッサコアの評価”, 情報処理学会論文誌, Vol. 42, No. 4, Apr.2001

[8] 笠原, “並列処理技術”, コロナ社, Jun.1991

[9] C.Lee, M.Potkonjak and W.H.Mangione-Smith “MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems”, 30th International Symposium on Microarchitecture (MICRO-30), 1997

[10] W.H.Chen, C.H.Smith and S.C.Fralick, “A Fast Computational Algorithm for the Discrete Cosine Transform”, IEEE Trans. Comm., Vol. COM-25, No.9, pp.1004-1009, Sep.1997

[11] Sun Microelectronics, “UltraSPARC^{TM} User's Manual”, Jul.1997

[12] 吉田, 渡辺, “ディジタル画像圧縮の基礎”, 日経BP出版センター, Jun.1999

[13] 小野, 鈴木, “JPEG/MPEG2の技術”, オーム社, Jan.2001

[14] 笠原, 木村, “シングルチップマルチプロセッサ”, 特許出願番号第363702号