C-018

２次元ＦＦＴの専用計算機

Special-Purpose Computer of Two-Dimensional FFT

佐竹 信一† 廣井 義明† 鈴木 裕也† 増田 信之‡ 伊藤 智義‡
Shin-ichi Satake Yoshiaki Hiroi Yuya Suzuki Nobuyuki Masuda Tomoyoshi Ito

1. 緒言

近年の数値シミュレーション技術及び計算機の発展により，様々な数値実験が研究室内で行われるようになった．それに伴い，シミュレーションに要する計算時間も増加する傾向にある．計算時間を短縮する方法として，高速な計算機の使用やプログラムソフトのアルゴリズムを高速化することなどが挙げられるが，スーパーコンピュータなどを使える環境になく，かつ計算対象が限定されるのであれば，PC と LSI 等を組み合わせた専用計算機を作製することが考えられる[1]． しかし，LSI は小規模の生産には向かず，内部回路の再構成が不可能であるために，計算内容の変更が不可能である．

我々は FPGA(Field Programmable Gate Array)を使用して専用計算機を試作した．FPGA は通常の LSI と異なり，プログラミングされた記述に従って電気的に論理配線を行う素子である．PC やワークステーションで回路を設計し，ターゲット用に配線配置したデータを転送してコンフィグレーションすることにより，内部論理を再構成することが可能である[2]．よって各シミュレーションに必要な専用計算機を目的に応じて再構成して使用することが出来る．

ところで現在，FFT はさまざまな分野において利用されている．特に数値シミュレーションにおける分野では，その計算速度が重要となる．本論文では，基数 2 の 2 次元 FFT(Two-dimensional Fast Fourier Transform)を行う専用計算機を設計し，FPGA にコンフィグレーションして計算を行い，その性能について評価した．

2．FPGA 回路設計における問題点

本論文では正方平面を 8×8 のメッシュ領域に分割して 2 次元 FFT を行うことを考える． $N \times N$ 個の値 $f(a,b)$ が与えられたとき，二次元 FFT の公式は以下の式で示される．

$$F(p,q)=\sum_{a=0}^{N-1}\sum_{b=0}^{N-1} f(a,b) W_N^{ap} W_N^{bq} \tag{1}$$

$$f(a,b)=\frac{1}{N^2}\sum_{p=0}^{N-1}\sum_{q=0}^{N-1} F(p,q) W_N^{-ap} W_N^{-bq} \tag{2}$$

ただし

$$W_N = \exp(-2\pi i / N) \tag{3}$$

で定義される [3].

2 次元 FFT をソフトウェアベースで高速化する手法は既にいくつかの方法が考案されているが，これらの手法を FPGA に応用することで高速化を達成することは難しい．原因として大部分のアルゴリズムでは計算の効率化のために複数のデータを保持する配列を使用している事が挙げられる．FPGA による計算では，メモリバンド幅があまり多く設定されていないため，一時データの保持のためにレジスタを多数作成，参照するようなアルゴリズムは好ましくない．よってこの事をふまえ，FPGA で FFT を行うのに有利なアルゴリズムを選ぶ必要がある．今回は，Cooley-Tukey のアルゴリズム[4]による 1 次元 FFT を走査する軸を変えながら複数回行う方法，Temperton のアルゴリズムによる 2 次元 FFT[5]，Ooura による 2 次元 FFT[6]，Sande-Tukey のアルゴリズム[7]による 2 次元 FFT の 4 つの方法を比較した．結果，Sande-Tukey による FFT アルゴリズムがデータの参照回数の点で有利であったため，これを採用することにした．

†東京理科大学　基礎工学部電子応用工学科
‡千葉大学大学院　工学研究科　人工システム科学専攻

3. 2 次元 FFT 回路

本論文では基数 2 の Sande-Tukey のアルゴリズムを参考にレジスタの参照効率を踏まえた変換手法を用いている．このアルゴリズムでは計算過程を

1. FFT，IFFT の選択及び回転因子の設定
2. バタフライ演算
3. データの整列

の 3 つに分解できる[3]．過程 1 のフローチャートを図 1 に示す．本回路ではこの回転因子の計算を行わず，あらかじめテーブルを FPGA 内に作製しておき，計算の効率化を図っている．

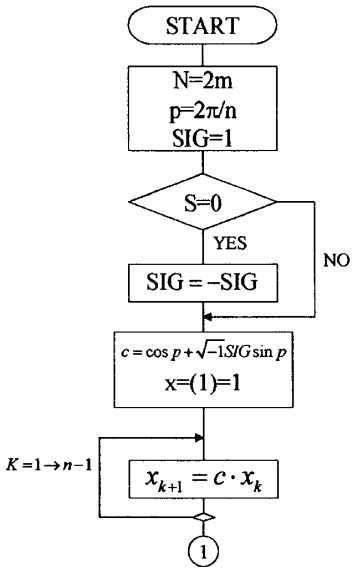

図1　FFT，IFFTの選択及び回転因子の設定

次にバタフライ演算を行う．添え字 a,b,p,q を 2 進数で表す時，その各桁の値は

$$\left.\begin{aligned} p &= (p_{m-1}p_{m-2}\cdots p_1p_0) \\ q &= (q_{m-1}q_{m-2}\cdots q_1q_0) \\ a &= (a_{m-1}a_{m-2}\cdots a_1a_0) \\ b &= (b_{m-1}b_{m-2}\cdots b_1b_0) \end{aligned}\right\} \tag{4}$$

と表される．ここで $m=\log_2 n$ である．これを(1)式に代入すると，

$$\begin{aligned} F(p,q) &= F(p_{m-1}\cdots p_0, q_{m-1}\cdots q_0) \\ &= \sum_{b_0}\cdots\sum_{b_{m-1}}\sum_{a_0}\cdots\sum_{a_{m-1}} f(a_{m-1}\cdots a_0, b_{m-1}\cdots b_0) \\ &\quad \times W_n^{p(a_{m-1}2^{m-1}+\cdots+a_0)+q(b_{m-1}2^{m-1}+\cdots+b_0)} \end{aligned} \tag{5}$$

を得る．加算の順序を変換して次のように書き換える．

$$\left.\begin{aligned} & f_1(p_0a_{m-2}\cdots a_0, q_0b_{m-2}\cdots b_0) \\ &= \sum_{b_{m-1}}\sum_{a_{m-1}} f(a_{m-1}\cdots a_0, b_{m-1}\cdots b_0)\times W_N^{p_0a+q_0b} \\ & f_2(p_0p_1a_{m-3}\cdots a_0, q_0q_1b_{m-3}\cdots b_0) \\ &= \sum_{b_{m-1}}\sum_{a_{m-1}} f_1(p_0a_{m-2}\cdots a_0, q_0b_{m-1}\cdots b_0)\times W_N^{p_1 2a+q_1 2b} \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ & f_l(p_0\cdots p_{l-1}a_{m-l-1}\cdots a_0, q_0\cdots q_{l-1}b_{m-l-1}\cdots b_0) \\ &= \sum_{b_{m-1}}\sum_{a_{m-1}} f_{l-1}(p_0\cdots p_{l-2}a_{m-l}\cdots a_0, q_0\cdots q_{l-2}b_{m-l}\cdots b_0) \\ &\quad \times W_N^{p_{l-1}2^{l-1}a+q_{l-1}2^{l-1}b} \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ & f_m(p_0\cdots p_{m-1}, q_0\cdots q_{m-1}) \\ &= \sum_{b_0}\sum_{a_0} f_{m-1}(p_0\cdots p_{m-2}a_0, q_0\cdots q_{m-2}b_0)\times W_N^{p_{m-1}2^{m-1}a+q_{m-1}2^{m-1}b} \\ &\equiv F(p_{m-1}\cdots p_0, q_{m-1}\cdots q_0) \end{aligned}\right\} \tag{6}$$

以上の式を $l=m$ まで繰り返すと，フーリエ変換が終了する．この過程のフローチャートを図 2 に示す．

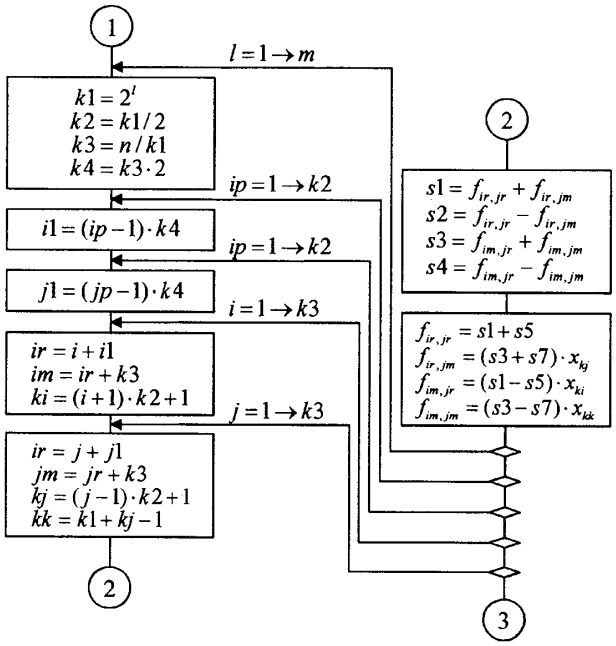

図2　バタフライ演算

バタフライ演算回路はステートマシンで構成されており，ステートの遷移には前述の同期クロックを用いている．一回の計算では 4 つの離散データを同時に計算し，この計算の終了に 6 クロック（同期クロックが High になってから再び High になるまでを 1 クロックとする）を要する．領域分割数が $n\times n$ とすると，$mn^2/4$ 回この計算を繰り返すことになる．図 3 に図 2 で示したバタフライ演算部のブロックダイアグラムを示す．

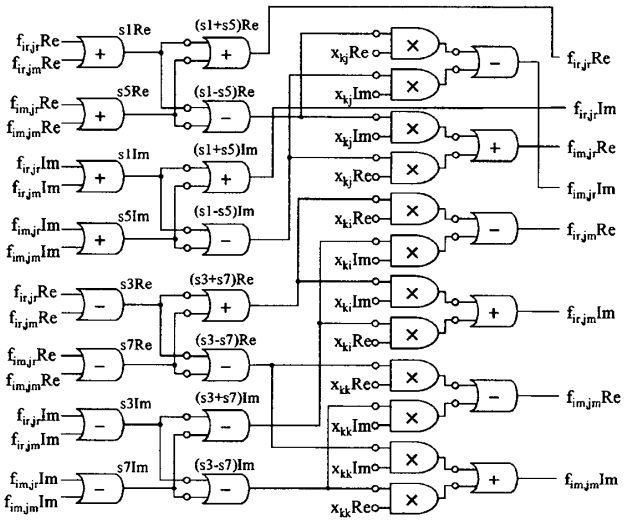

図3　バタフライ演算のブロックダイアグラム

この図 3 中の全ての演算器を 1 クロックごとに動かし，パイプラインで計算することにより計算速度の向上を図った．

最後に，バタフライ演算によって逆転した離散データの置換を行う．図 4 に示される置換を実行し 2 次元 FFT の結果が得られる．

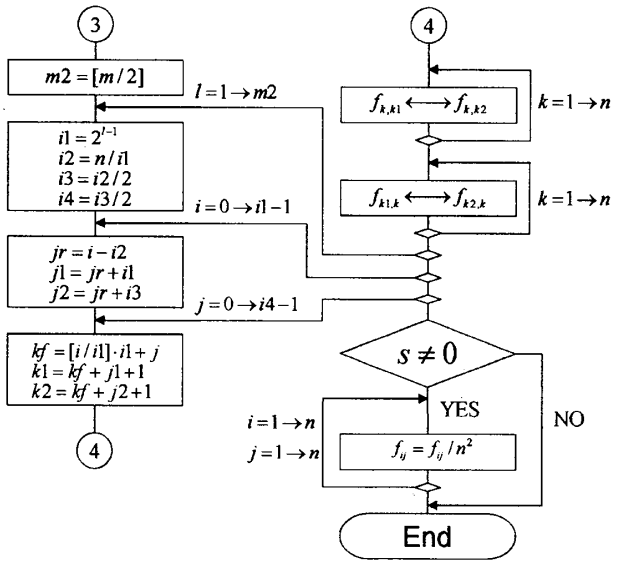

図4　データの整列

4. FPGA 設計

通常 FPGA や LSI を使用した専用計算機では PC 側と LSI 側で計算を分担し，計算速度の向上を図っているが，今回の回路では計算式が単純であるということもあり，計算を分担するような形にはしておらず，全計算を FPGA で行うように設計した．本計算機の設計には Xilinx 社製 FPGA virtex II Pro(700 万ゲート相当)を 4 つ搭載している千葉大学と理化学研究所が共同開発した HORN-5 ボードを使用した[1]．本ボードは PCI を介して PC と接続されており，33MHz で動作する PCI バスと同期を取るために 66MHz と 133MHz の 2 種類のクロックが用意されている．4 つの FPGA を利用した分割計算も可能な本ボードであるが，今回は 1 個のみを使用して回路の有効性を検証した．

そしてこの HORN-5 を使用し，ハードウェア記述言語である VHDL(VHSIC Hardware Description Language)を用いて回路の設計を行った．VHDL プログラムの記述及び配線配置データの作成には Xilinx 社のプログラマブルロジック作成ソフトウェアである，ISE6.3i を使用した．FPGA への回路の実装には HORN-5 付属のソフトウェアを用いていた．

回路自体の構成は，32bit の配列，加算器及び乗算器で構成されている．バタフライ演算は 16bit 固定小数点演算を行っている．本計算では 16bit のうち上位 1bit を符号部として固定し，残りの 15bit を必要に応じて整数部と小数部に割り当てることにした．また，乗算を行うと積は元の bit 数の倍になるため，小数点の位置を考えて適切な範囲を選び切り出した．

5. 専用計算機によるデータ転送

PC と FPGA の間のデータの交換には，8bit 幅の CBUS と，64bit 幅の DBUS の 2 種類の信号線によって行われる．このうち CBUS は PC から FPGA へのデータの書き込み，及び FPGA から PC へのデータの読み込みの開始，終了を表す信号の送受信に使用されるため，メモリのアドレス指定，及び計算結果の取出しなどの具体的なデータのやり取りには DBUS を用いることになる．これらの信号線によるデータの受け渡しには，HORN-5 付属のライブラリを使用した C++によるプログラムを作成し，これにより制御を行った．回路の動作開始，終了，結果受け渡し等は全てこの C++によって作成したプログラムによって PC 側から行われることになる．

まず OpenPciDev 関数によってデバイスをオープンする．複数枚のボードを使用している際はここでどのボードを使用するかを決定する．次に計算の開始を意味する信号を FPGA に送信する．前述のとおり，データの送受信には DBUS を用いる必要があるため，データを FPGA に書き込むための WriteBase1Ulong 関数を使用して，規定のアドレスにデータを書き込む．FPGA 内のメモリにはあらかじめ計算開始と計算終了を確認するための領域をあらかじめ用意しておく．計算開始判定用のアドレス領域にデータが書き込まれると，それを計算開始の合図として内部で計算が開始される．

計算は FPGA 内で自動的に規定回数（ステップ）まで行われる．ユーザはその間計算が終了したかどうか判断するためにデータを取り出すための ReadBase1Ulong 関数により計算終了判定用のアドレスのデータを取り出す．計算が終了するとこのアドレスのデータが書き換えられるように作製してあるので，このデータを確認することで計算が終了しているかどうかを判断することが出来る．このような手法をとるのは今回使用したボードでは FPGA 側から PC 側に自動的に計算が終了した旨を知らせる方法がないためである．

計算が終了したら，前述の WriteBase1Ulong 関数を使用して全計算結果を取り出し，C++のプログラムにて 10 進数のデータに変換する．その後に ClosePciDev 関数によってオープンしたデバイスをクローズし，全計算過程を終了する．

6. 計算速度評価

作製した回路を前述の FPGA に実装し，動作確認を行った．領域分割数は 8×8 とし，FFTW のソフトウェア計算ライブラリ[8]による計算速度と比較を行った．8×8 の 2 次元 FFT は一瞬で終了するため，この FFT を連続して 100000 回実行し，実行回数で割ることで得られた時間を計算時間とした．計算時間の測定には OS が Microsoft Windows 2000 を，CPU に Intel Pentium4 1.8GHz を搭載した PC を使用した．また，プログラムのコンパイラには Microsoft Visual C++の C コンパイラを使用した．

今回，FPGA の動作周波数は 66MHz とした．FPGA に計算開始信号を送り，計算が終了して FPGA 内のメモリに全てのデータが格納されるまでの時間を FPGA による計算時間とした．表 1 に PC と FPGA のそれぞれが FFT にかかった時間を示す．

表 1　計算時間

	計算時間[ns]	速度比
FFTW	8672	1
FPGA	850	10.2

表 1 から分かる通り，FPGA による計算では PC による計算と比較しておよそ 10.2 倍の高速化を達成している．

今回作成した計算回路による FFT と，PC による FFTW の実行結果との誤差は，最大で 0.04%程度となった．

7. 結言

FPGA を用いた 2 次元 FFT 専用計算機を作成し，FFT の高速化を図った．パイプラインで複数の計算を同時に行うことにより，1 つの FPGA で 10.2 倍の高速化が達成された．より高性能な CPU を使用した場合の，PC の計算速度の向上を考慮しても，今回設計した回路は十分な高速化が成されていることが分かる．

今回は回路の有効性を検証するために，FPGA の動作周波数は 66MHz 動作としたが，本回路は 133MHz でも動作することが予想されている．したがって，PC との通信があるため単純に計算速度が 2 倍にはならないが，さらなる高速化の実現は可能である．

また，HORN-5 ボードは 4 つの FPGA を搭載している．そのため，計算回路作成時に FPGA 同士の通信が少なくなるような工夫ができれば，より分割数が多く，高速な 2 次元 FFT ができることが予想される．

以上

参考文献

[1] Tomoyoshi Ito, Nobuyuki Masuda, Kotaro Yoshimura, Atsushi Shiraki, Tomoyoshi Shimobaba and Takashige Sugie, Special-purpose computer HORN-5 for a real-time electroholography, Optics Express 13(6) 2005, 1923-1932

[2] 仲野巧，VHDL によるマイクロプロセッサ設計入門，CQ 出版社，2002，24

[3] 町田東一，小島紀男，FORTRAN 応用数値計算，東海大学出版会，1989，120,121,135-141

[4] J.W.Cooley, J.W.Tukey, An Algorithm for the Machine Calculation of Complex Fourier Series, Mathematics of Computation, Vol.19 1965, 297-301

[5] C. Temperton, Fast Mixed-Radix Real Fourier Transforms Journal of Computational Physics 52 1983, 340-350

[6] http://www.kurims.kyoto-u.ac.jp/~ooura/index-j.html

[7] Gentleman, W. M., and G. Sande, Fast Fourier Transforms for fun and profit, AFIPS Proc., 1966 Fall Joint Computer Conf., Vol. 29, 563-578

[8] Matteo Frigo and Steven G. Johnson, The Design and Implementation of FFTW3, Proceedings of the IEEE 93 (2), 2005, 216–231