計算機アーキテクチャ 130－21
（1998. 8. 7）

FPGAを用いた手書き漢字認識ニューラルネットのハードウェア支援

竹谷史生* 吉永努* 荒井正之†
大津金光* 馬場敬信*

*宇都宮大学工学部、†帝京大学理工学部

本稿では、書き換え可能なFPGAを用い、様々なニューラルネットアルゴリズムに対応可能なシステムを提案する。また、一例として、本システムにバックプロパゲーションネットワーク構造の手書き漢字認識アルゴリズムを適用したアーキテクチャについて述べる。4ニューロンを集積するPUのFPGAをXILINX社のXC40125XV-1で実現する場合の予備評価を行った。その結果、演算サイクル数はワークステーション上のCプログラムと比べて約1/200となり、学習演算処理能力は129.2MCUPS、フォワード演算処理能力は600.6MCPSという結果が得られた。

FPGA Hardware Support Tool for Recognition of Handwritten Kanji Using Neural Network

Fumio TAKEYA*,Tsutomu YOSHINAGA*,Masayuki ARAI†,
Kanemitsu OOTSU* and Takanobu BABA*

*Ustunomiya University,†Teikyou University

This paper presents a system which is able to implement various neural network algorithms. It utilizes the reconfigurability of FPGAs. We apply the system to an algorithm for hand-written kanji recognition using a back propagation network. Preliminary evaluation is made by supposing to use XILINX XC40125XV-1 for a processing unit. As a result, we show its performance of 129.2MCUPS for learning calculations and 600.6MCPS for forward calculations.

1 はじめに

ニューラルネットワークの学習計算は、膨大な計算量を必要とする。そのため、汎用のワークステーション等では、学習計算に非常に長い時間を必要とする。これに対して、ASICを用いたニューロチップ[1][2][3]やニューロシステム[4]などが開発されてきている。しかし、それらのニューロチップを使って高速に学習計算をしようとすると、そのチップ(ハードウェア)構成に合わせたアルゴリズムを採用しなくてはならない。さらに、新たなアルゴリズムに対しては、ハードウェアを再開発しなければならないことも多い。

そのような中、近年プログラム可能なゲートアレイFPGAの技術的な発展はめざましく、高集積、高速なFPGAが開発されている。1990年頃から現在までに、書き換え可能なFPGAを用いた様々なリコンフィギュラブル・コンピューティング・システム[5](RCシステム)の開発が報告されている[6][7][8][9]。これらのシステムの中で最も有名なのは、SPLASH[6]である。汎用計算機が苦手とするパターンマッチング問題を高速化し、一躍注目を浴びた。

本稿では、書き換え可能なFPGAを用い、様々なニューラルネットアルゴリズムに対応可能なシステムを提案する。また、一例として、本システムにバックプロパゲーション(以下BP)ネットワーク構造の手書き漢字認識アルゴリズム[10]を適用し、その予備評価を行った。

2 BPネットワーク構造の手書き漢字認識アルゴリズム

2.1 ネットワーク構造

ネットワーク構造は、3層BPネットワークとする。3層BPアルゴリズムは、外部からの入力データを入力層を通じてネットワークに入力し、中間層を経て出力層から出力する。そして、出力データとその入力データに対しての正しい出力(教師信号)に近づくように各層間の重みを修正し、その誤差が許容値以下になるまで繰り返し学習計算する方法である。

本研究では、手書き漢字の認識率を向上するため入力層の特徴ベクトルには外郭方向寄与度特徴(Peripheral Direction Contributivity 以下PDC)[11]を用いる。また、クラスタリング手法(k-means法[12])により、17文字の類似文字群(閘, 閏, 閖, 開, 閣, 間, 閑, 関, 閤, 閃, 陶, 闘, 閥, 聞, 閉, 問, 悶)を選択する。そして、以下の理由より、ネットワーク構造は入力層768、中間層34、出力層17とする。

1. 入力層
   PDC特徴は文字を大局的にみて、文字線の複雑さ、方向、接続関係、相対位置関係を反映する特徴である。文字を45度おきに8方向から走査し、横切る各文字線の最初に横切る輪郭点での水平・垂直の各方向寄与度を投影する。この場合、横切る文字線数(外郭深度)、3本目までの方向寄与度が成分別に投影される。投影軸を8区間に等分割し、各区間内の各方向寄与度成分の値を平均することによって区間の寄与度成分とする。さらに、±45度の方向寄与度を合計し、入力次元数は8×2×3×8×2=768とする。

2. 中間層
   出力層の2倍の34とする。

3. 出力層
   識別文字数と同数の17とする。

2.2 3層BPアルゴリズム

3層BPアルゴリズムを8つのステージに分ける。

ステージ0:入力層-中間層の積和演算u_jを計算する。

$$u_j = \sum_{i=0}^{767} Wim_{ij} x_i \tag{1}$$

ここで、Wim_{ij}は入力層-中間層間荷重係数、x_iは入力層データである。

ステージ1:ステージ0で計算したu_jのシグモイド関数m_jを計算する。

$$m_j = f(u_j) = \frac{1}{1+exp(-u_j)} \tag{2}$$

ステージ2:中間層-出力層の積和演算v_kを計算する。

$$v_k = \sum_{j=0}^{33} Wmo_{jk} m_j \tag{3}$$

ここで、Wmo_{jk}は中間層-出力層間荷重係数である。

ステージ3:ステージ2で計算したv_kのシグモイド関数o_kを計算する。

$$o_k = f(v_k) = \frac{1}{1+exp(-v_k)} \tag{4}$$

ステージ4:出力層から中間層への荷重係数を修正する学習データδo_kを計算する。

$$\delta o_k = (t_k - o_k) o_k (1 - o_k) \tag{5}$$

ステージ5:中間層から入力層への荷重係数を修正する学習データδm_jを計算する。

$$\delta m_j = m_j (1 - m_j) \sum_{k=0}^{16} \delta o_k Wmo_{jk} \tag{6}$$

ステージ6:中間層-出力層間の荷重係数、結合係数$\Delta Wmojk$を修正する。

$$\begin{aligned} \Delta Wmo_{jk} &= \eta \delta o_k m_j + \alpha \Delta Wmo_{jk} \\ Wmo_{jk} &= Wmo_{jk} \Delta Wmo_{jk} \end{aligned} \tag{7}$$

ここで、ηは学習定数、αは安定化定数である。

ステージ7:入力層-出力層間の荷重係数、結合係数$\Delta Wimij$を修正する。

$$\begin{aligned} \Delta Wim_{ij} &= \eta \delta m_j x_i + \alpha \Delta Wim_{ij} \\ Wim_{ij} &= Wim_{ij} \Delta Wim_{ij} \end{aligned} \tag{8}$$

学習計算はステージ0〜7の計算(フィールドフォワード処理と学習処理)を繰り返す。また、識別時にはステージ0〜3のフィールドフォワード処理のみを行い、出力値が最大となる出力層ニューロンに学習させた文字を識別結果とする。

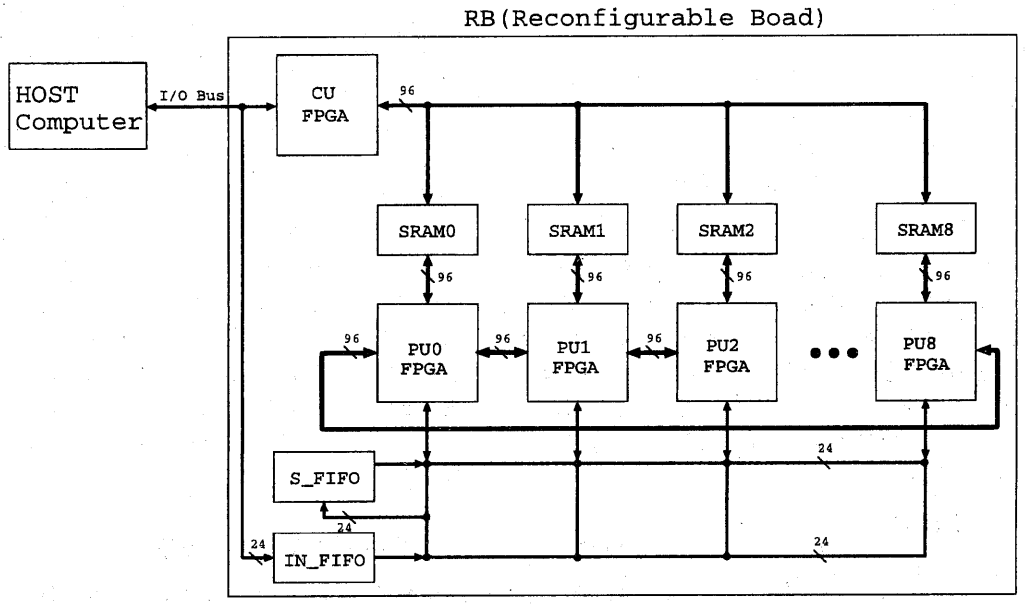

図 1: ニューロシステムの全体構成図

3 システムの全体構成

図1にシステム全体の構成図を示す。本システムは、ホストコンピュータとリコンフィギャラブルボード(以下 RB) で構成する。RB は、RB 上のバスや I/O バスなどを制御する Control Unit(以下 CU)、ニューラルネットの学習計算などの計算集約的な処理を行う 9 個の Processing Unit(以下 PU)、データ格納用の 9 個の SRAM と 2 個の FIFO で構成する。CU と PU は FPGA である。また、PU-PU 間のデータ通信を柔軟にするため、すべての PU にブロードキャストやマルチキャスト転送を可能にするデータバスと、隣接する PU 間にリング状のデータバスを設けた。

各ブロックは、先に説明した BP ネットワーク構造の手書き漢字認識アルゴリズムを実現するために、次のように用いる。

1. CU
   RB 全体の動作を規定するステートマシンを構成。
2. PU
   ニューロンを構成。
3. SRAM
   結合係数、荷重係数、教師信号、出力層データなどを格納。
4. FIFO
   入力層データを格納。

次に、学習計算時のシステム全体の流れを簡単に説明する。

1. 各 SRAM を初期設定する。
2. ホストコンピュータは、IN_FIFO に向けて、そのハーフ・フラグがアクティブになるまで一文字分の入力層データ (768 ワード) を転送する。
3. CU は、IN_FIFO のエンプティー・フラグがネガティブになったらフィールドフォワード処理を制御する。
4. 各 PU は、フィールドフォワード処理を行い、出力層データを SRAM に格納する。CU はホストコンピュータに出力層データを転送する。また、study_flag が立っていれば 5 へ、立ってなければ 3 へ移行する。ホストコンピュータは次の入力層データ用の教師信号を転送する。

   ※ study_flag はホストコンピュータが制御し、現在の計算が学習計算であるかフィールドフォワード処理のみかを示す。
5. CU が学習処理を制御する。また、ホストコンピュータが誤差 ($err = err + \sum_{k=0}^{16}(t_k - o_k)^2$) を求める。

これを 17(文字) × 100(セット) 行う。そして、誤差の平均値を求め、前の誤差平均と比較し、許容値より大きければ引続き学習計算を行う。また、小さければ学習計算を終了する。

このように、本システムではハードウェアとホストコンピュータの両方で処理を行う。

4 ハードウェアアーキテクチャ

PU は、24 ビット浮動小数点乗算器 4 個、加算器 4 個、除算器 1 個と 24 ビット幅のレジスタ 26 個を搭載する。また、各ステージの状態遷移レジスタを持ち、その状態遷移により各演算が切り替わり高速な計算ができる。また、入力層-中間層間の計算は 1PU4 ニューロン、中間層-出力層間の計算は 1PU2 ニューロンを構成する。

4.1 FIFO

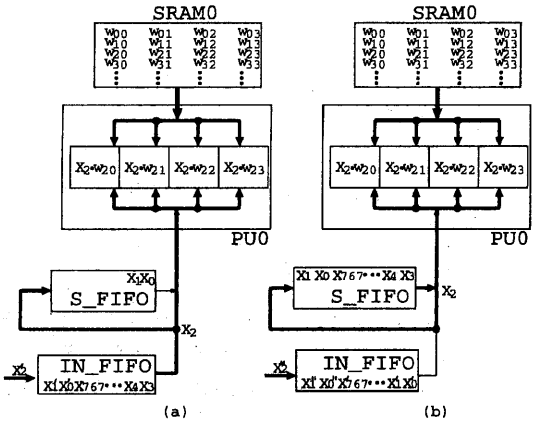

図 2: 2 つの FIFO

FIFO は入力層データを格納するのに用いる。入力層データを格納するのに FIFO を用いた理由は、以下の通りである。

1. データを格納するとエンプティー・フラグがネガティブになる。
   これにより、ホストコンピュータから転送された入力層データが FIFO に格納してあるかどうかが分る。
2. FIFO にデータを半分格納するとハーフ・フラグがアクティブになる。
   これにより、ホストコンピュータは 1 文字分の入力層データ (768 ワード) を転送できるかどうかが分る。
3. アドレスバスが不必要。
   RB 側の読み出しアドレスが不要となる。

また、FIFO を 2 つ用いる。1 つ (IN_FIFO) はホストコンピュータから転送してきた入力層データを格納するために、もう 1 つ (S_FIFO) は現在学習している入力層データを格納するために用いる。このようにすることにより、図 2に示すようにステージ 0、ステージ 7 で計算中でも、IN_FIFO に次の入力層データ (X_i')、その次の入力層データ (X_i'') と I/O Bus が空いるときに格納できる。また、現在学習している入力層データ (X_i) は、図 2の (a) のように最初に入力層データを使うときに S_FIFO に格納する。そして、次に使うときから S_FIFO にある入力層データを読み出し、S_FIFO に書き込む。このようにすることにより、ニューロン数が増え 36 ニューロンを越えた場合でも対応ができる。

4.2 ブロードキャスト転送

中間層データはステージ 1 で各 PU の内部レジスタ mreg に格納する。そして、図 3に示すようにステージ 2、ステージ 6 において、CU に選択された PU がその中間層データを他の PU にブロードキャストする。

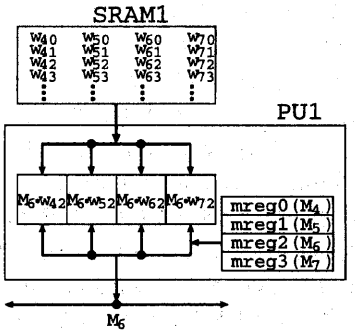

図 3: ブロードキャスト転送

4.3 出力層-中間層間学習信号

ステージ 5 で行う計算 (6) 式で、各 PU のローカルメモリ SRAM はすべての結合係数データ Wmo_{jk} を持っていない。そこで、図 4に示した後向き積和演算の計算方法 [1] (中間層-出力層の計算を 1PU4 ニューロンとしたときの例) のように、内部レジスタ breg に格納してある中間層-出力層間学習データ δo_k と SRAM から読み込んだ結合係数データ Wmo_{jk} から $\delta o_k Wmo_{jk}$ を求め、隣接する PU に転送しながら計算していく。

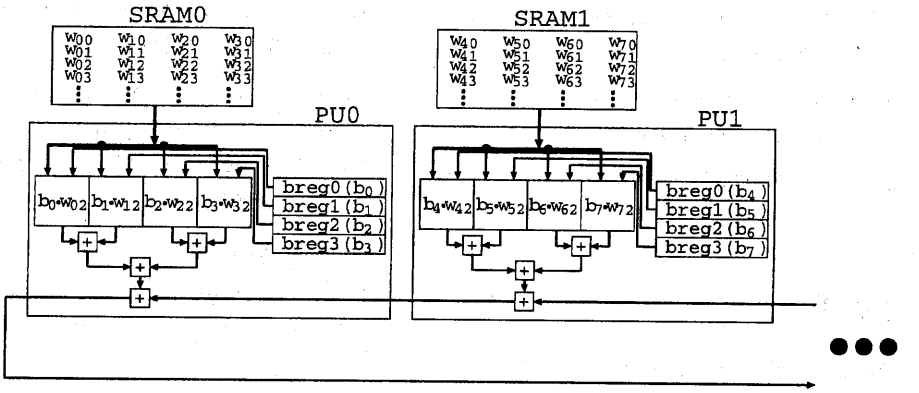

図 4: 後向き積和演算の計算方法

4.4 シグモイド計算

シグモイド関数を計算するために (9) 式のようにテイラー展開を用いる。

$$f(x) = \frac{1}{1+e^{-x}} = \frac{1}{2 + \frac{-x}{1!} + \frac{(-x)^2}{2!} + \ldots + \frac{(-x)^n}{n!}} \quad (9)$$

ステージ 1 とステージ 3 では、内部レジスタに格納してある重み付き和 u_j ,v_kを入力とし、4 個の乗算器と加算器、そして 1 個の除算器で 2 段のパイプラインを構成し計算する。また、シグモイド関数は点 (0,0.5) に対称であることと、(9) 式の場合入力データが負の数の方が収束しやすいことより、入力が負の数の場合そのまま出力し、正の数の場合は $1-f(-x)$ を出力する。

5 予備評価

表 1: ワークステーションとの比較

ステージ	WS(O2)	RB	比率
S0	261834	769	340.5
S1	—	47	—
S2	5563	37	150.3
S3	—	27	—
S4	179	3	59.7
S5	5202	46	113.1
S6	11757	88	133.6
S7	522589	3075	226.5
Sum-(S1+S3)	807124	4018	200.9

5.1 演算サイクル数

本システムの性能を評価するために、各演算ステージの演算サイクル数を計算し、ワークステーション (以下 WS) の演算サイクル数と比較した結果を表 1に示す。表 1からわかるようにほとんどの計算時間は、ステージ 0 とステージ 7 である。本システムの演算サイクル数は、WS の約 1/200 であることが分る。なお、WS の演算サイクルは、C 言語プログラムコンパイル結果から、一命令を一サイクルと仮定して求めた。コンパイル時の最適化には O2 を用いた。また、WS でのシグモイド計算は、指数関数が x の値によって演算サイクル数が違うので S1 と S3 は比較をしていない。

5.2 論理合成

RB を Verilog-HDL を用いて記述した。そして、Verilog-XL シミュレータで動作確認を行い、PU を論理合成した。論理合成の結果を表 2に示す。論理合成には、Synopsys の FPGA EXPRESS を用い、ターゲット FPGA は XILINX の XC40125XV-1(4,624clb) とした。

表 2: PU の仕様

ターゲット FPGA	XC40125XV-1
パッケージ	499 ピン PG
CLB 数	4,241
クロック	18MHz(50.5ns)

5.3 パフォーマンス評価

本稿でシステムに実装したネットワークは、入力層 768、中間層 34、出力層 17 である。これの総結合数は 26,690 である。また、表 1より学習演算処理での演算サイクル数は 4092、フォワード演算処理 (S0 から S3) は 880、表 2より一演算サイクルは 50.5ns である。よって、学習運算処理能力 CUPS(Connections Update Per Second) とフォワー

ド演算処理能力 CPS(Connections Per Second) は、以下のように計算できる。

$$\begin{aligned} CUPS &= 26,690/(4092 * 50.5 * 10^{-9}) \\ &= 129.2MCUPS \\ CPS &= 26,690/(880 * 50.5 * 10^{-9}) \\ &= 600.6MCPS \end{aligned}$$

6 むすび

本稿では、書き換え可能な FPGA を用い、様々なニューラルネットアルゴリズムに対応可能なシステムを提案した。また、一例として、本システムに BP ネットワーク構造の手書き漢字認識アルゴリズムを適用した時のアーキテクチャについて述べた。そして、処理を行う PU の FPGA を XILINX 社の XC40125XV-1 として論理合成し、予備評価を行った。その結果、演算サイクル数はワークステーション上の C プログラムと比べて約 1/200 となり、学習演算処理能力が 129.2MCUPS、フォワード演算処理能力が 600.6MCPS という結果が得られた。

今後は、ニューラルネットワークに認識させる漢字数を増やし、本システムで大規模 BP ネットワークを構成する場合について評価する。また、他のニューラルネットアルゴリズムを適用し、本システムが様々なアルゴリズムに対応できることを示していく予定である。

謝辞日頃より御指導、御助力を頂いた宇都宮大学情報工学科馬場、吉永、大津研究室の諸氏に感謝する。

本研究の一部は東京大学 大規模集積システム設計教育研究センターより提供して頂いた CAD ツールを使用した。深く感謝する。

本研究は一部文部省科学研究費 基盤研究 (C) 課題番号 09680324、基盤研究 (B) 課題番号 10558039、奨励研究 (A) 課題番号 09780237 の援助による。

参考文献

[1] 広瀬佳生, 安仏英明, 山下公一, 後藤源助, "バックプロパゲーション専用プロセッサのアーキテクチャ", 信学技報 ICD92-17,pp.40-46,1992.

[2] Y.Kondo, et.al., "A 1.2FLOPS Neural Network Chip Exhibiting Fast Convergence," ISSCC '94 IEEE International Solid-State Circuits Conference, TP 13.1,pp.218-219,Feb.1994.

[3] 齋藤修, 相原公久, 藤田修, 内村国治, "大規模ニューラルネットワークに対応可能な学習機能内蔵ディジタルニューロチップ" 信学技報 CPSY98-4,pp.25-32,1998.

[4] 柴田克成, 安永守利, 大山光男, 益田昇, 柳生正義, 浅井光男, 山田稔, 坂口隆宏, 橋本雅, "高速学習型ニューロ WSI のシステム設計", 信学技報 ICD90-127,pp.49-56,1990.

[5] 末吉敏則, "Reconfigurable Computing System の現状と課題-Computer Evolution へ向けて-", 信学技報 CPSY96-91,pp.111-118,1996-12.

[6] Maya Gokhale,Bill Holmes,Andrew Kopser, Dick Kunze,Dan Lopresti,Sara Lucas,Ron Minnich, "SPLASH:A Reconfigurable Linear Logic Array", January 26,1994.

[7] M.Wazlowski,L.Agarwal,T.Lee,A.Smith, E . Lam,P.Athanas,H.Silverman,S.Ghosh,"PRISM-II Compiler and Architecture", Proc. IEEE Workshop on FPGAs for Custom Computing Machines,pp.147-157,1994.

[8] Jean-Luc Beuchat,Jacques-Oliver Haenni and Eduardo Sanchez, "Hardware Reconfigurable Neural Networks", LNCS1388. Rolim Parallel and Distributed Processing, 1998.

[9] Michael J . Wirthlin , Brad L . Hutchings, "DISC:The dynamic instruction set computer ", Proc.SPIE-Int.Soc.Opt.Eng.(USA),vol.2607, pp.92-103.1995.

[10] 荒井正之, 王晋申, 奥田健三, 宮道壽一, "Honeycomb ネットによる多字種の手書き漢字認識", 信学論 (D-II),J77-D-II,9,pp.1708-1715.1994-9.

[11] 萩田泰一, 内藤誠一郎, 増田功, "外郭方向寄与度特徴による手書き漢字の識別", 信学論,J66-D,10,pp.1185-1192,1983-10.

[12] ハイ, 森下, 蕪山, 伊崎, 山本, "手書き漢字認識におけるテンプレート複数化の検討", 信学技報,PRL81-42,pp.49-56,1981-9.