計算機アーキテクチャ 74－2
（１９８９．１．２６）

アルゴリズム駆動ニューロコンピュータＡＮ１のハードウェア

久長穣　新田健一　相原玲二　阿江忠
広島大学　工学部

本稿では、ニューラルネットワークと従来のノイマン型コンピュータとの融合をめざしたアルゴリズム駆動ニューロコンピュータＡＮ１のアーキテクチャと、そのハードウェアについて報告する。

ＡＮ１はホップフィールド型ニューラルネットワークをニューラルメモリと見なし、従来のコンピュータをメモリの側から拡張した形になっている。また、ホップフィールド型ニューラルネットワークを直接構成することは困難であるという立場から、我々はニューラルメモリを分割して実現する。この結果、結線数が減少し、ニューラルメモリの実現が容易になる。なお、分割された基本単位もホップフィールド型ニューラルネットワークである。

この基本単位のネットワークを構成するニューロンとして、素子感度が小さいことおよび、関数の変更が容易であるという点から、ＲＡＭニューロンを採用している。

Hardware of Algorithm-driven Neurocomputer AN1

Yutaka HISANAGA, Ken-ichi NITTA, Reiji AIBARA, Tadashi AE

Faculty of Engineering, Hiroshima University
Saijo, Higashi-Hiroshima, 724 Japan

This paper describes a hardware implementation of Algorithm-driven Neurocomputer AN1, which incorporates the neural network into the conventional Neumann-type computer.

In AN1, Hopfield neural network is regarded as a neural memory, and AN1 includes a more extended memory than the conventional computer does. In the implementation, we realize the Hopfield neural network, by dividing into some blocks, where each of them is also a Hopfield neural network. Consequently, the number of interconnections among neurons decreases, and the implementation of neural memory becomes easy.

In the hardware of AN1, we adopt a RAM as a neuron, where we call it RAM Neuron, and, therefore, the circuit sensitivity becomes very low and it is also easy to change the threshold function.

1. まえがき

ニューラルネットワークはいろいろな構造が提案されているが、以下のように大別できる。

(1) 多層ニューラルネットワーク

(2) ホップフィールド型ニューラルネットワーク

(1)は層状構造を持ち、情報伝達が一方向で帰還のないものであり、パーセプトロンに代表される。学習アルゴリズムのもと、例を繰り返し与えて学習させることで、ネットワークに機能あるいは知識を与えることができる。主として、パターン認識などに応用される。

これに対して、(2)のタイプ、すなわち、図1に示すホップフィールド型ニューラルネットワーク（以下、Hニューラルネットワークと略す）は、物理系のエネルギに相当する評価関数を最小化することで最適解を求める手法を与えるけれども[1][2]、Hニューラルネットワークを構成するためには、多層ニューラルネットワークの場合と異なり、学習のみならず、求解の対象となる問題の性質を反映させる必要がある。また、真の解ではない局所安定状態(local minima)に陥った場合は、一種の確率的なアルゴリズムも採用することが多い。

一般に、Hニューラルネットワークとアルゴリズムとの関係は図2のように描けるから、これをそのまま実現するアーキテクチャは最適化問題のための超高速ニューロコンピュータのアーキテクチャと考えることができる。筆者らは、このアーキテクチャを実現したニューロコンピュータをアルゴリズム駆動ニューロコンピュータと呼び[3]、そのプロトタイプAN1を製作している。本稿では、AN1のハードウェアについて述べる。

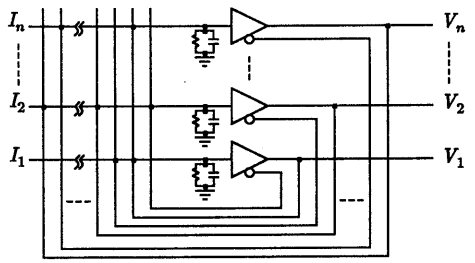


図1　ホップフィールド型ニューラルネットワーク（原形）

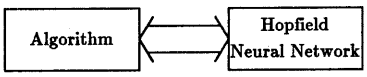


図2　ホップフィールド型ニューラルネットワークとアルゴリズム

2. AN1アーキテクチャの概要

AN1はニューラルネットワークと従来のノイマン型コンピュータとの融合をはかるものであり、次の部分より構成される。

(1) ニューラルメモリ

(2) ニューラル制御プロセッサ

(3) 合成アレイ

Hニューラルネットワークをメモリとみなすと図3のように書き込むデータ X が（一般には）読み出し時には X' に変化するメモリとなる。このメモリを以後、**ニューラルメモリNM**と呼ぶ。**NM**とアルゴリズムは図2のように相互作用をするが、アルゴリズムを実現するプロセッサが**NM**を制御しているとみなし、これを**ニューラル制御プロセッサNP**と呼ぶ。**NP**と**NM**は図4のようなデータフローサイクルを形成する。AN1では、アルゴリズム駆動アーキテクチャの具体的な形として直交アーキテクチャをとるが、**NM**から**NP**へのインターフェースを**合成アレイSA**と呼ぶ。

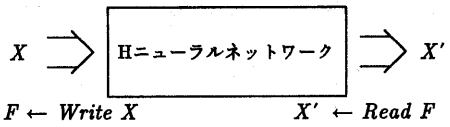


図3　メモリとしてみたホップフィールド型ニューラルネットワーク

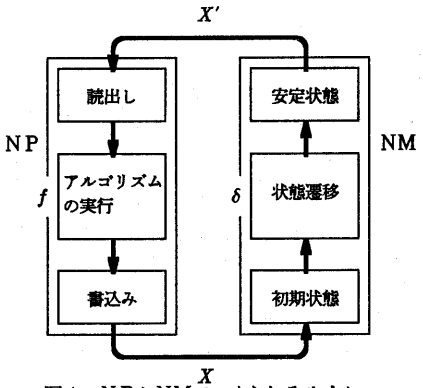


図4　NPとNMでつくられるサイクル

NMをディジタル・ホップフィールド・ニューラルネットワークにより実現すると1サイクルタイムで *Write* と *Read* が完了するから、**NP**も X のビット長分を並列に処理するメカニズムが要求される。一方、与えられた問題を解くためのHニューラルネットワークを直接構成するにはいろいろ困難な点があるという立場から、我々は**NM**を分割して実現する[3]-[5]。この結果、結線数が減少し、**NM**の実現が容易になる。**NM**を分割したときの単位を**基本ブロック**と呼び、基本ブロックも一つのHニューラルネットワークである。このとき**NP**も同じように分割すれば、**NM**の分割数と同じ数のマイクロプロセッサを用い、マルチプロセッサとして実現するのが現実的な方法である。しかし、AN1（暫定版）ではとりあえずシングルプロセッサで実現する。

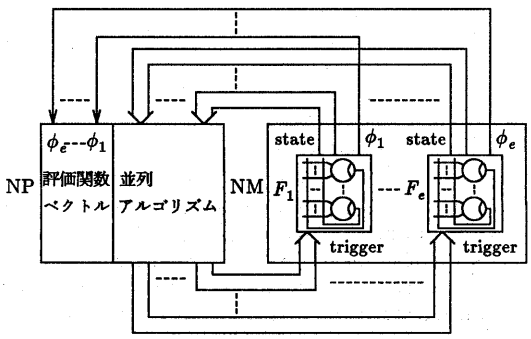


図5　アルゴリズム駆動ニューロコンピュータ

2.1. アルゴリズム駆動アーキテクチャ

以上のようにして構成されるアルゴリズム駆動ニューロコンピュータは図5のように描かれる。図5はノイマ

ン型アーキテクチャのようにプロセッサとメモリに二分されているが、ノイマン型の場合とは異なり、メモリの側から展開した構成になっている。その細部は次のとおりである。

（1）ニューラルメモリNM

基本ブロックであるHニューラルネットワークの集合、すなわち、

$$NM = \{ F_1, F_2, \cdots, F_e \}$$

からなる拡張Hニューラルネットワークを**ニューラルメモリNM**と呼ぶ。ある時刻において、$F_i(i=1,2,\cdots,e)$ の状態ベクトルを $s_i(i=1,2,\cdots,e)$ とすると、**NM**の状態ベクトル X は

$$X = (s_1, s_2, \cdots, s_e)$$

と表される。この状態ベクトル X が安定になるのを待って、次のニューラル制御プロセッサPが *Read, Write* する。なお、各 F_i は付随した評価関数 ϕ_i をもつから、評価ベクトル

$$(\phi_1, \phi_2, \cdots, \phi_e)$$

が**NM**のもつ評価関数である。

（2）ニューラル制御プロセッサNP

NPの動きは、

① **Read** input **from** NM
② { アルゴリズムの実行 }
③ **Write** output **to** NM

を1サイクルとし、**NM**の状態遷移が安定するまでの時間を待って、このサイクルを繰り返す。

②のアルゴリズムは、①で読んだ**NM**の状態ベクトル X' を入力とし、次の**NM**の状態遷移を起こさせるための初期条件であるベクトル X を出力する。状態ベクトル X' と X のビット長 n は

$$n = \sum_{i=1}^{e} |s_i|$$

である。$|s_i|$は各 F_i のビット長であり、$\max(|s_i|)=m$ とすると、

$$n \leq em$$

となる。このとき、**NM**は m 個のニューロン素子からなる F(基本ブロック)を e 個用意すればよいことになる。ユーザの記述するアルゴリズムは系全体の評価関数

$$g(\phi_1, \phi_2, \cdots, \phi_e)$$

を最小とする関数列 τ として実行される（図4参照）。

$$\tau = f^{1}\delta \; f^{2}\delta \cdots f^{t}\delta$$

ここに、各サイクル $i=1,2,\cdots,t$ に対して

$$X = f(X')$$
$$X' = \delta(X)$$

(ただし、*Read X' from NM, Write X to NM* である)。

$\delta(X)$は X を初期条件としたときの**NM**の自律的な遷移であり、プログラマブルではあるが、f と比べた場合、固定的なものと考える。すなわち、f がユーザのつくるプログラムであるのに対し、δ はシステムの供給するユーティリティのようなものである。f はユーザの考えるアルゴリズムにより与えられるが、δ は①論理的な解析結果から得られたもの、のほか②学習によって得られたものであっても構わない。τ における f はふつうの関数であるが、δ は非決定的な関数にもなりうる。また、構造からいえば、δ は**NM**の遷移を表現するものであるから、F_i の集合から成っている。一方、f は**NP**において実行される並列プログラムであれば何でもよいことになる。

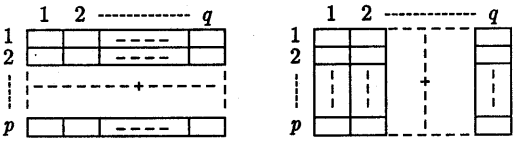
以上述べた**NM**と**NP**から図5のように構成されるアーキテクチャを、**アルゴリズム駆動アーキテクチャ** (Algorithm-driven Architecture) と呼ぶ。その特徴は、ニューラルコンピュテーションがアルゴリズムにもとづいてつくられた並列プログラム f と相互に作用しながら形成されるところにある。なお、アルゴリズムは基本ブロックの評価関数から求めた系全体の評価関数を考慮してつくられるから、系としては、Hニューラルネットワークの場合を拡張した意味でのニューラルコンピュテーションを実現している。

2.2. 直交アーキテクチャ

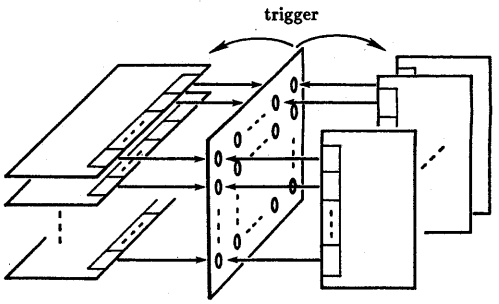
AN1では、アルゴリズム駆動アーキテクチャの具体的な形として、図6に示すような**直交アーキテクチャ** (Orthogonal Architecture)[5] をとる。直交アーキテクチャでは**NM**の要素である基本ブロックを

行集合 $NM_r = \{ F_{1r}, F_{2r}, \cdots, F_{pr} \}$
列集合 $NM_c = \{ F_{c1}, F_{c2}, \cdots, F_{cq} \}$

に二分する（図6）。基本ブロックの大きさ（ニューロン数）は行ブロック q、列ブロック p と表されるが、**AN1**の場合 p,q とも最大15である。



(a) row blocks　(b) column blocks



p row blocks　synthesis array　q column blocks

(c) Orthogonal Architecture

図6　直交アーキテクチャ

基本ブロックごとに評価関数 $\phi_{1r}, \ldots, \phi_{pr}, \phi_{c1}, \ldots, \phi_{cq}$ が付随しているので、**NM**はその値をそのまま**NP**へ渡す。**NM**から**NP**へのインターフェースである**合成アレイSA**は、評価関数の引渡のほか、行と列のつくる格子点に対応する2つのニューロン間の演算（出力）とニューロンへのトリガ（入力）の役割をする。すなわち、**NP**の側から見ると、まず、

$$read \;\; x_{ij} \;\; s.t. \;\; x_{ij} = x_{ij}^{(r)} \circ x_{ij}^{(c)}$$

{行と列の交点の演算}

(ただし、$x_{ij}^{(r)}$は行ブロックに属する i 行目の基本ブロックの左から j 番目のニューロンの出力であり、$x_{ij}^{(c)}$は列ブロックに属する j 列目の基本ブロックの上から i 番目のニューロンの出力であり、2つのニューロンは交点にある）により**NM**からの値 x_{ij} ($i=1, \ldots, p$, $j=1, \ldots, q$)を読む。$\circ$は任意の2変数ブール演算 (and, or, ...) を表す。むろん、単なる値の読出し

$$read \;\; x_{ij}^{(r)} \quad \{行読出し\}$$
$$read \;\; x_{ij}^{(c)} \quad \{列読出し\}$$

も独立に行える。次に、書込みはすべて独立に

$write\ \ 0\ or\ 1\ to\ x_{ij}^{(r)}$ {行書込み}

$write\ \ 0\ or\ 1\ to\ x_{ji}^{(c)}$ {列書込み}

となる。なお、これらはビットごとにすべて並列に行うことができる。

SAが出力のビット処理を並列に行うことで、SIMD処理を支援することもできる。

2.3. ニューラルメモリの設定法

アルゴリズム駆動ニューロコンピュータAN1は、NMをベースにしたアーキテクチャである点がノイマン型アーキテクチャともっとも異なる点である。NMにはユーザが開発する基本ブロック F を付加することは可能とするが、基本的なものについては初めから用意しておくことも必要である。

F をHニューラルネットワークで構成した場合、ニューロン数が n ならば、安定ベクトルの個数の最大値は ${}_nC_k$ であることが知られている。このときのベクトルが重み一定となることに着目し、しきい値のみを変化させて、任意の一定の重みの安定ベクトルの集合をもつようにできる[6]。（表1）表1において、${}_nC_i$ 個の安定ベクトルをもつとき、n ビットベクトル中 i 個の 0（すなわち、$n-i$ 個の 1）をもつ重み一定ベクトルが安定ベクトル集合になることを意味している。安定ベクトル以外は不安定であるが、状態遷移は束グラフで与えられ、連結している。重み一定符号は、これまでの計算機構成においても採用されてきた符号の一つであり、融合性の点では都合がよい。したがって、AN1では、**システム側からは、F として、k-out-of-n 符号を安定状態としてもつ ${}_nC_k$ 安定回路を供給するという方針をとっている。**

表1　しきい値と安定ベクトルの個数の関係

Threshold θ	Number of Stable Vectors
$n-1 < \theta$	${}_nC_0 = 1$
$n-1 < \theta < n-1$	${}_nC_1 = n$
⋮	⋮
$n-i-1 < \theta < n-i$	${}_nC_i$ $(max.\ {}_nC_{\lfloor n/2 \rfloor})$
⋮	⋮
$0 < \theta < 1$	${}_nC_{n-1} = n$

3. ディジタル・ニューロン－RAMニューロン－

ニューラルネットワークはニューロンを多数相互結合させたものである。このニューロンの工学的モデルは図7(a) のようなしきい値素子であり、その入出力関係は次式で表される。

$$y = \begin{cases} 1 & \text{if } X \geq 0 \\ 0 & \text{if } X < 0 \end{cases} \qquad (1)$$

$$X = \sum a_i x_i - \theta$$

ここで、x_i は入力、y は出力、a_i は入力 x_i に対応する重みを表す。このモデルをVLSI技術により実現することにより、ハードウェア・ニューロンが実現できることになる。このためには次の条件を満足しなければならない。

(1) 入出力関数がしきい値関数であること

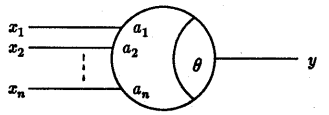
(2) 結合と重み、つまり入出力関数が可変であること

(3) 多くのニューロン間の相互結合を可能とする多入力素子であること

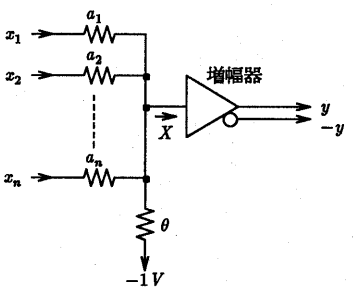
この実現法の多くは、アナログ回路を用いて行なわれる[7]。しかし、式(1)のモデルの入出力は0と1の離散値である。一方、Hニューラルネットワークでは状態遷移に連続性のあることが望ましい[8]。そのため、式(1)をシグモイド関数で近似するモデルが用いられる。シグモイド関数は式(2)のように表され、$\lambda \to 0$ のときしきい値関数と一致する。

$$y = \frac{1}{1+exp(-\frac{X}{\lambda})} \qquad (2)$$

AN1では、以下の理由によりデジタル回路を採用している[9]。



(a) しきい値素子



(b) アナログしきい値素子

図7　しきい値素子とアナログしきい値素子

アナログ素子で実現する方法では、図7(b)のように重みとシグモイド特性はそれぞれ抵抗と飽和特性を持つ増幅器で構成される。入力を電圧に、重みをコンダクタンスに対応させることにより、式(1)の線形和Xは電流加算によりアナログ的に計算される。増幅器により電流－電圧変換を行い、出力を得る。この方法は入力数を大きくできるけれども、抵抗値は増幅器の入力、出力特性に影響され、各値の調整が必要となる。また、入力関数を可変にするには抵抗値を変更しなければならず、一般的な実現は必ずしも容易ではない。

アナログ演算回路の欠点を補うべく、部分的にディジタル素子を用いたニューロン実現法もあるが、AN1では完全なデジタル回路でニューロンを実現している。式(1)のしきい値関数はブール素子により実現できる。結合と重みの変更には可変なブール素子が必要となるが、PLA，ROM／RAMがそれに該当する[5][10]。

PLAはAND層とOR層からなり、図8のようにAND層の接続を自由に変更できる素子である。ブール関数を積和標準形で表したとき、AND層のANDゲート数は少なくとも積和標準形の積項数だけ用意し、しかも、可変にできる必要がある。ところが、市販のPLAのAND数は高々8～64程度と少なく、オンラインで可変にできるものはない。将来はPLAでもニューロンが実現できる可能性があるが、AN1では現在の技術で実現することにした。

ROM／RAMはアドレス入力に対して、それに対応する記憶セルの内容を出力する素子であるから、ブール関数の真理値表を記憶させれば、ブール関数素子を直接実現できる。RAMの場合はオンラインで真理値表の書き込みが可能である。RAMを用いるニューロンの基本回路を図9に示す。

ＡＮ１では、以上の理由によりＲＡＭニューロンを採用しており、具体的には市販の　256kbit　のＳＲＡＭ(62256)を用いて、１７入力（うち、２入力はトリガ用）のニューロンを構成している。このとき実際のＲＡＭニューロン回路は図１０のようになる。実際には、ＲＡＭに真理値表の書き込み回路および読み出し回路が付加されているが、この図では省略している。ここで、真理値表は１７入力１出力の表として与えられる。

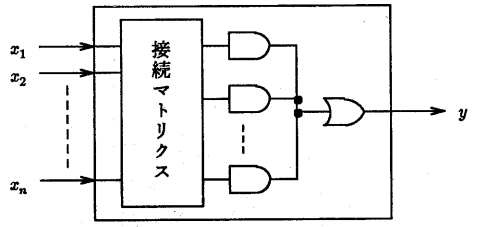


図８　ＰＬＡニューロン

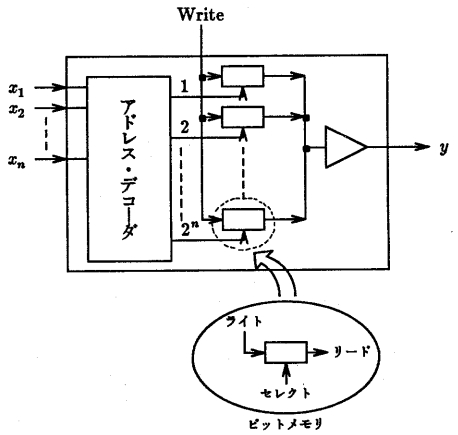


図９　ＲＡＭニューロン

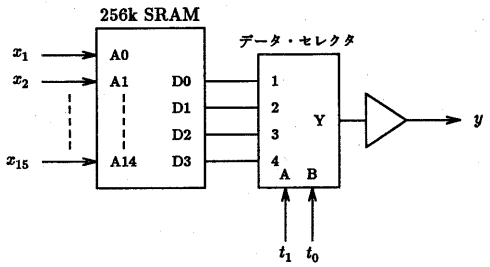


図１０　実際のＲＡＭニューロン回路

表２　データセレクタの動作

A B	Y
0 0	D0
0 1	D1
1 0	D2
1 1	D3

あいにく、256kbit(32k×8bit)のＳＲＡＭは１５本のアドレス入力しかない。２本の入力を増やすために、データ出力を 4bit 利用する。その 4 bit の中から 1bit を選択する働きをするのが、データセレクタである。この選択には、トリガ入力を用いて行ない、このときのデータセレクタの選択を表２に表す。したがって、ＳＲＡＭに書き込む真理値表は１５入力４出力の表となる。データセレクタは上記のような働きの他に、ＲＡＭニューロンの出力ドライバの働きもする。またＳＲＡＭに真理値表を書き込む際の、ＳＲＡＭのアドレス入力とデータ出力の分離も行う。

4. 基本ブロック

4.1. ＲＡＭホップフィールド・ニューラルネットワーク

基本ブロックは、すべてのしきい値素子相互間に帰還のある結合を持つＲＡＭホップフィールド・ニューラルネットワークである（図１１）。Ｈニューラルネットワークはエネルギを評価関数とし、これを極小化するよう動作するが、ＡＮ１の基本ブロックは、いわば、メモリのサイクルタイム内でそれを実現する。

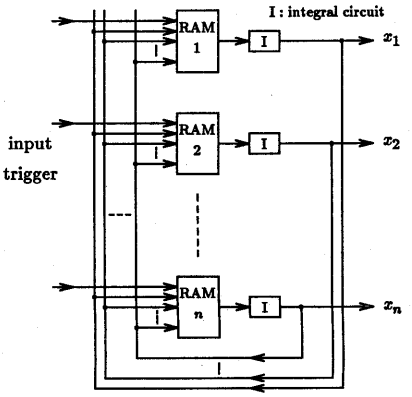


図１１　ＲＡＭホップフィールド
ニューラルネットワークの原理

ＡＮ１では、基本ブロックはＲＡＭニューロンにより実現されている。ＲＡＭには一定の遅延が存在するため、図１１は積分回路がないと発振する。RAM (62256) 単独で測定した遅延が約80nsであるので、少なくとも、二素子間の最短帰還路で発振する可能性があった。事実、実験回路では周期150ns （単純な予測では 160ns）で発振した。この発振を抑えるために、積分回路を挿入し、遅れ補償を行っている。積分回路の時定数は発振周期の約2.5倍程度にとり 350～400ns に選んでいる。

基本ブロックには、関数の設定すなわちＳＲＡＭへの真理値表の書込み／読出しモードとニューラルメモリとしての動的な動作モード、の２つのモードがある。さらに、ニューラルメモリとしての動的な動作モードは、状態の書込み，状態遷移，状態の読出し，の３フェーズに分かれる。実際の回路は図１２のようになるが、この図では真理値表の書込み／読出し回路は省略している。

状態書込みは、各ニューロンの出力を強制的に０，１または不定にすることにより行われる。これを実現するために、２ビットのトリガ入力を用意している。ニューロン本来の真理値表はＳＲＡＭのデータビット０(D_0) に書き込まれている。ＳＲＡＭの出力を強制的に０(１)にするためには、すべての入力に対して０(１)を出力するように真理値表を構成し、D_1(D_2)に書き込んでおく。状態の書込みは、３つのデータビットをデータセレクタ(図１２)で切り替えることにより実現される。

トリガ設定後は、データセレクタを D_0 へ切り替えることによって、書き込まれた状態から次の状態へ遷移す

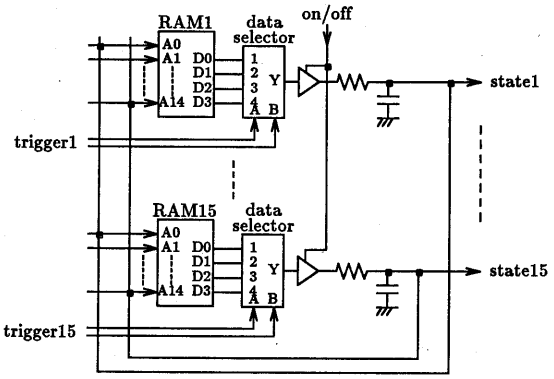


図１２　ＲＡＭニューロンによる実際の
ホップフィールド型ニューラルネットワーク

る。また、基本ブロックの一部のビットを１または０に固定することによってそこをマスクした状態で遷移を行わせることもできる。

遷移後の基本ブロックの状態は、ニューラル制御プロセッサ**NP**によって読み出されるとともに、次章で述べる合成アレイに渡される。

本来、トリガ入力はＳＲＡＭから見ると、真理値表の入力という意味で他のＳＲＡＭからの入力と同じである。ＳＲＡＭとして 32k×8bit のものを用い１枚の基板上に１５個が相互結合されている。そのとき、ＳＲＡＭのアドレス線はすべて相互結合に使用されている。

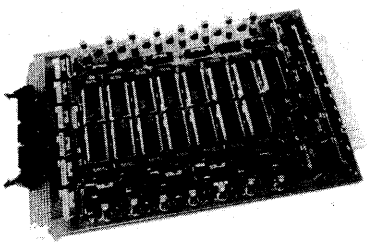
ＳＲＡＭへの真理値表の書込みは、バッファによってＨニューラルネットワークにおけるフィードバックループを切り、通常のメモリとして**NP**がアクセスする。読出しは関数を検証するために用いる。真理値表の書込み／読出しはオフライン的に用いられるモードであり、しきい値関数を与えると自動的に真理値表が作成される。

基本ブロックは、以上の機能の他に、テストなどのために**NP**からトリガが入力できるように、トリガレジスタを用意している（本来、トリガは合成アレイを用いて入力される）。これによって基本ブロック単体でも動作可能となっている。

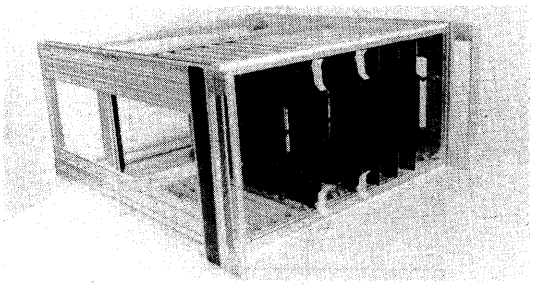
１つの基本ブロックは、189×279*mm* のプリント基板上に実装されている(図１３(a))。ＡＮ１では、この基板を最大３０枚(行,列基本ブロック、各々１５枚)実装することができる(図１３(b))。したがって、ＡＮ１の最大ニューロン数は最大ＳＲＡＭ数と等しく、４５０となっている。

4.2. 状態遷移に関する実験

このネットワークは状態をもつという点から、メモリであると同時に、ニューラルコンピュテーションを行う状態機械とみることが出来る。状態遷移はトリガベクトル（初期状態）により、(1)安定状態から安定状態への遷移と(2)不安定状態から安定状態への遷移に分けられる。そこで、それぞれの場合について、ネットワークの状態遷移時間と遷移を起こすに必要なトリガベクトルのパルス幅（トリガ幅）の測定を行った。これらの時間は図１４のように定義している。実験の対象となる基本ブロックは、２．で述べた ${}_nC_k$ 安定回路に設定している。表３に測定結果を示すが、各時間は最大値を意味し、トリガベクトルを与えて、遷移した状態を読み出す動作を行う限り、その動作を補償するものである。この値はアナログニューラルチップに比較して数倍程度大きい。これはＲＡＭの遅延が大きいために発振を補償する時定数を大きくしなければならないためである。しかし、ＲＡＭニューロンを専用ＬＳＩ化することにより改善可能であると思われる。

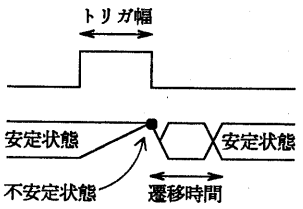


(a)基本ブロック

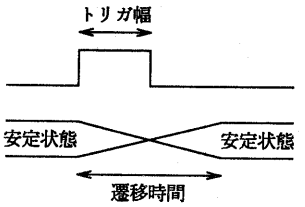


(b)ニューラルメモリ

図１３　基本ブロックとニューラルメモリ



(a)安定状態 → 不安定状態 → 安定状態



(b) 安定状態 → 安定状態

図１４　トリガ幅と遷移時間の定義

表３　遷移時間とトリガ幅

	トリガ幅	遷移時間
安定→安定	1.06	2.50
安定→不安定	2.40	-
不安定→安定	-	3.80

(ニューロン数 n=8 , 時定数 350ns , 単位 μs)

5. 合成アレイとニューラル制御プロセッサ

合成アレイＳＡには、行基本ブロックと列基本ブロックのつくる格子点ごとに演算ユニットがあり、**NM**より送られてくる２つのニューロンの状態を入力とする演算

を行い、その結果を２つのニューロンまたはNPに渡す（図１５）。さらに、特別な演算として、すべての格子点において行と列のニューロンの状態が一致しているかどうかを出力することができる。（ただし、ニューロンへは出力されない。）

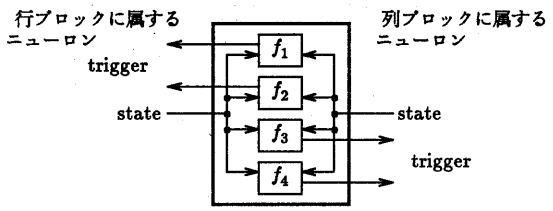


図１５　１つの格子点に対応する演算ユニット

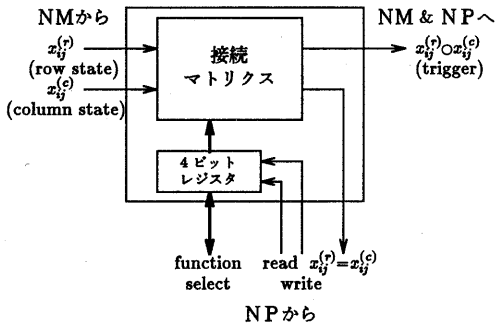


図１６　一つの$f_k(k=1\sim4)$の実現するＰＬＡ (PAL16R4)

表４　$f_k(k=1\sim4)$ の真理値表

$x_{ij}^{(r)}$ $x_{ij}^{(c)}$	$f_k(x_{ij}^{(r)},x_{ij}^{(c)})$
0 0	b_{k0}
0 1	b_{k1}
1 0	b_{k2}
1 1	b_{k3}

前節で述べたように、各ニューロンは２ビットのトリガ入力を持っているので合成アレイの出力も２ビット用意している。各演算 f_1〜f_4 は任意の２変数ブール関数（１６個ある）であり、各々独立に演算を選択できる。図１６のように、１つの f_k は１個のＰＬＡによって実現され、演算の選択はＰＬＡ内の４ビットのレジスタ$(b_{k3}b_{k2}b_{k1}b_{k0})$を用いて行われる（表４参照）。このレジスタはNPにより書き込まれるが、ＰＬＡごとに独立に設定できる。さらに、全ての演算ユニットを同一に設定したい場合は、全ユニットを同時に設定可能である。これにより、ＳＩＭＤ処理をさせることもできる。

演算結果は、基本ブロックのトリガに出力するかどうかが選択でき、またNPで読み出すこともできる。

８個の格子点に対応する演算ユニットを 203×216mm のプリント基板上に実装する設計になっている。１つの行または列に対して２枚の基板を対応させるため、ＳＡは１５×１５の行列を全部で３０枚の基板で実現している。ＳＡの各基板と各基本ブロックの接続はNPを介さず、直接接続される。したがって、ＳＡ各基板にはNPと接続するインターフェースと、NMと接続するインターフェースがある。

ニューラル制御プロセッサNPは本来、基本ブロックに対応するだけのマルチプロセッサであるべきであるが、現在のＡＮ１ではシングルプロセッサで代用しており、ＮＥＣのＰＣ－９８０１を一台用いている。NPは、主に次の処理を行なう。

(0) NMの基本ブロックを問題解法に必要なホップフィールド・ニューラルネットワークとして設定するために、各ＲＡＭニューロンの真理値表を導出したのち、その書込みを行う。（また、必要ならば読出しを行う。）

(1) アルゴリズム駆動アーキテクチャのサイクル（図４参照）をアルゴリズムの面から支援する。つまり、NMの書込み、読出しを行う。

(2) ＳＡの各演算ユニットの関数の設定、演算結果の読出しを行う。

NPとNM、および、NPとＳＡの接続は図１７のようにＰＣ－９８０１のＩＯ空間に設けた二つのポートを通して行われる。NM,ＳＡの各基板にはアドレスが割り当てられており、NPはこのアドレスを用いて各基板をアクセスする。ただし、ＳＡの基板はブロードキャスティングができる。

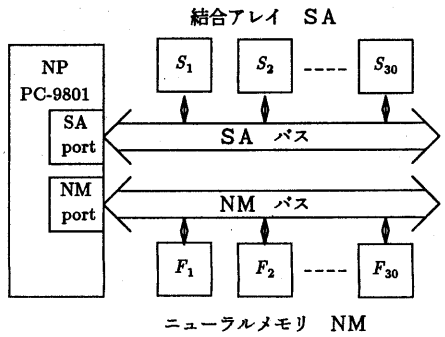


図１７　NPとNM，NPとＳＡの接続

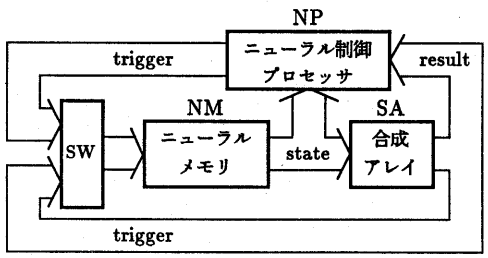


図１８　ＡＮ１の全体構成

これまでに述べた基本ブロック、合成アレイおよびNPを含めた全体構成を図１８に示す。ただし、ＳＲＡＭの書換えおよび合成アレイの演算の変更を行う部分は図には含まれていない。図１８のSWは、基本ブロック上にあるトリガレジスタからのトリガ入力と合成アレイからのトリガ入力を切り替えるためのものであり、基本ブロックのプリント基板上に実装されている。

6. おわりに

ホップフィールド型ニューラルネットワークを拡張させたニューラルメモリをベースとするニューロコンピュータであるアルゴリズム駆動ニューロコンピュータＡＮ１のハードウェアについて述べた。ディジタル・ニューロンとしてＲＡＭニューロンを用いると、次の長短がある。

(1) ＲＡＭの性質として、書き換えが可能である。そのため、ＡＮ１ではニューラル制御プロセッサNPの

支援のもと、オンラインにニューロンの関数を可変にできることを意味している。このため、**従来のニューラルネットワークより、空間複雑さ (space complexity) の点で優れている。**

(2) しかし、**RAMのデコーダはニューロン数 n に対し O(2^n) の空間複雑さをもつ。**今後、(2)の欠点を改良するためには、しきい値関数およびその部分クラスの特徴を解明する必要がある。一般のしきい値関数については悲観的であるが、いくつかの部分クラスについては複雑さを減少させることが可能であると思われる。

現在のＡＮ１（暫定版）ではニューラルメモリと合成アレイは並列接続されているが、これらと**NP**との結合が弱いため、この部分がボトルネックになる可能性がある。これはＮＰを並列化することで解決されるので、基本ブロック数と同じ数（３０）のプロセッサを用いた並列プロセッサによる実現に変更する予定である。

ホップフィールド型ニューラルネットワークの特徴を生かすために、系の評価関数を定めそれを用いて問題を解く必要がある。ＡＮ１では評価関数を計算するハードウェアがなくても、ホップフィールドのエネルギ関数に相当する評価関数は物理系として直接実現されている。もっとも、リアプノフ関数である評価関数をユーザが自由に設定したい場合は、基本ブロックの状態は**NP**で読出し可能であるから、任意の評価関数が計算できるが、計算時間が問題になる。今後、問題に応じた評価関数が必要となるならば [11]、それを計算するハードウェアを用意する必要がある。

最後に、いろいろ討論していただいている山下雅史助教授、藤田聡氏、ならびに製作に協力してくれている三井靖博、平田秀一両氏に感謝する。

文献

(1) J.J.Hopfield and D.W.Tank; " 'Neural' computation of decisions in optimization problems", Biological Cybernetics, **52**, pp.141-152 (1985).

(2) D.W.Tank and J.J.Hopfield; " Simple "Neural" optimization networks: an A/D converter, signal decision circuit, and a liner programming circuit", IEEE Trans. Circuits and Systems, Vol.CAS-33, no.5, pp.533-541 (May 1986).

(3) 阿江,山下,相原,新田; "アルゴリズム駆動ニューロコンピュータＡＮ１", 信学技報 88-129 ICD (Dec. 1988).

(4) 阿江忠; 「ＶＬＳＩコンピュータ」, 第４.４節,電子情報通信学会

(5) T.Ae and R.Aibara; "A neural network for 3-D VLSI accelerator", Proc. International Workshop on VLSI for Artificial Intelligence, Oxford (July 1988).

(6) T.Ae, H.Nagami and N.Yoshida; "On multistable transistor circuits using threshold logic operation", Int.J.Electronics, 36, 6, pp.849-856 (1974).

(7) R.E.Howard, L.D.Jackel, and H.P.Garf; "Electronic neural network chips", 5th Internal Workshop on Future Electron Devices - Three Dimensional Integration -, May 30-June 1, 1988 Miyagi-Zao pp.33-37.

(8) J.J.Hopfield and D.W.Tank; "Neurons with graded response have collective computational properties like those of two-state neurons", Proc. Acad. Sci. USA, 81, pp.3088-3092 (1984).

(9) 久長,新田,相原,山下,阿江; "ＳＲＡＭを用いたニューラル・ネットワークの試作" 昭和63年電気関係学会中国支部連大 122508.

(10) I.Aleksander "Are special chips necessary for neural computing ?" International Workshop on VLSI for Artificial Intelligence 20th - 22nd July, 1988

(11) T.Ae, M.Yamashita and K.Nitta; "Neural scheduler for real-time networks", Proc. Hawaii International Conference on System Sciences, Jan.1989.