進化するハードウェアをめざして

樋口 哲也

一般にハードウェアは，設計時点で想定した条件下でしか機能せず，もしそれら以外の状況に遭遇すると，もう役に立たない．しかし，本稿で提案する〝進化するハードウェア（Evolvable Hardware; EHW と略す)〟は，環境が変化したり，あるいは未知の環境におかれても，目的とする行動を達成するために，ハードウェア自身が適応的に自らのハードウェア構成を変更させていくことができるものである．どのような論理機能を実現してよいかハードウェア設計者に予知できない環境への適用を目的とするため，従来のハードウェア設計手法は成り立たない．しかし，プログラム可能型大規模論理素子にみられる可変構造型アーキテクチャと，遺伝的学習を組み合わせることによって，〝進化するハードウェア〟を構築することが可能である．本稿では，ブーリアン関数の学習の例として6マルチプレクサ問題をとりあげ，GAL と呼ぶプログラム可能型論理素子においてその機能が実現できることを示すとともに，VLSI EHW の基本構想を述べる．

1．はじめに

一般にハードウェアといえば，設計者が予め意図した機能だけを実現すればよい．つまり設計者が想定しない状況や意図した機能以上のものが要求される状況が出現して，それに対してハードウェアが有効に機能しなくても当然である．

しかし，もし環境の変化に応じてハードウェア自身に自らのハードウェア構成を変更できる能力があれば，従来の応用では考えられなかった新しい応用形態が出現することは確実であり，さまざまなメリットが享受されるものと考えられる．そのような応用形態の必然性を示唆するものとして，昨今の，人工生命（Artificial Life）に対しての関心の高まりがある[7]．

人工生命の分野は多岐にわたっており，まだ鳥瞰図を描くまでのまとまりさえ得られないほどであるが，確実にその中の一分野を占めているものとして，ANIMAT[注1] がある[10]．これは，未知の環境，あるいは外界において，人工的な生物や物体が，環境からのフィードバック情報だけを頼りに適応的に学習を繰り返して自分自身の能力を高め，目的とする行動なり性能を達成しようというものである．

そこでの基本は，強化（reinforcement）学習であり，正解は与えられない．すなわち未知の外界から間つけ的に与えられる報酬（reward）を唯一のフィードバック情報として，ときには不正確な入力データなどをうまく処理しつつ，学習を繰り返す．現在はソフトウェアシミュレーションによる研究がほと

Tetsuya HIGUCHI 電子技術総合研究所

注1 シミュレートされた animal と，自律的な robot とを合わせた造語と考えられる．

んどであるが，これらが究極的に押し進められた時の形態の一つが，本稿で提案する「進化するハードウェア」，すなわち環境に適応して自らの構造を変化させることのできるハードウェアであると筆者らは考えている．

このようなハードを作るのに従来のような設計手法は全く役に立たない．なぜならハードで実現すべき論理機能が設計者に設計時点でわかっていないからである．しかしながら，最近はプログラム可能型理素子（Programmable Logic Device；以下 PLD）に代表される，ユーザが自由に書き変え可能な大規模な論理素子が利用可能になってきており，そのアーキテクチャ技術と，遺伝的学習を組み合わせることによって，進化するハードウェアが構築できるものと考えられる．具体的には，ハードウェア構成の決定情報を，遺伝的アルゴリズム（Genetic Algorithm；以下 GA）の染色体とみなし，環境に最も適応するハードウェア決定情報を選んで，それに従いハードウェア構造自体を再構成する．

本稿では，このような進化するハードウェアを構築するための基礎実験として PLD を使った機械学習の成果も示しつつ，進化するハードウェアのイメージ，VLSI 化を想定したマシンアーキテクチャ，および今後の研究課題について論ずる．

2．進化するハードウェアとは

2．1 従来のハードウェアとの相違

進化するハードウェア（EHW）のイメージを，従来型のハードウェア（Conventional Hardware; CHW）と対比して，図1に示す．いま，ある CHW が機能レベル1を満たすように設計されているとする．しかし，仮にその CHW の適用される環境が変わり，より高度な機能，すなわち機能レベル2まで要求されることになったとする．このような場合，CHW はもはや何もできない．ハードウェア

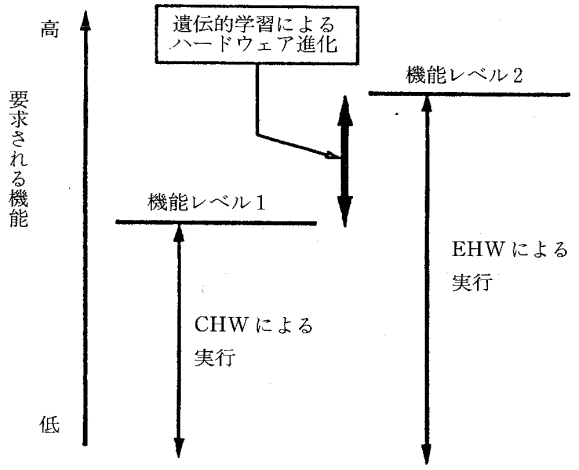

図1 従来のハードウェアとの相違

機能が固定されていて変更の手段がないからである．

しかし，EHW の場合は，次節に述べるように，EHW の内部に常に複数の再構成可能ハードウェア構造とそれらの決定情報（染色体）とを保持している．このため，もし環境が変わって機能レベル2が要求されるようになった場合には，それまでのハードウェア構造を捨て，遺伝的学習によって新たなハードウェア構造が選ばれて，その出力が EHW の出力となる．

2．2 EHW の基本構成

VLSI 化を前提とした EHW の基本構成を，図2に示す．以下に示すように大別して5つの部分より成るが，最も特徴的なのは複数の回路構成を再構成可能素子として同時に保持し，その選択，淘汰のために機械学習部，および GA 処理部をハードウェアとして1チップ内に備えていることである．つまりハードウェア進化のために必要な機能を1チップの中にすべて包含する．

1．再構成可能素子群
2．ハードウェア構成情報（染色体）群
3．GA 処理部
4．機械学習部
5．入力，および出力インタフェイス

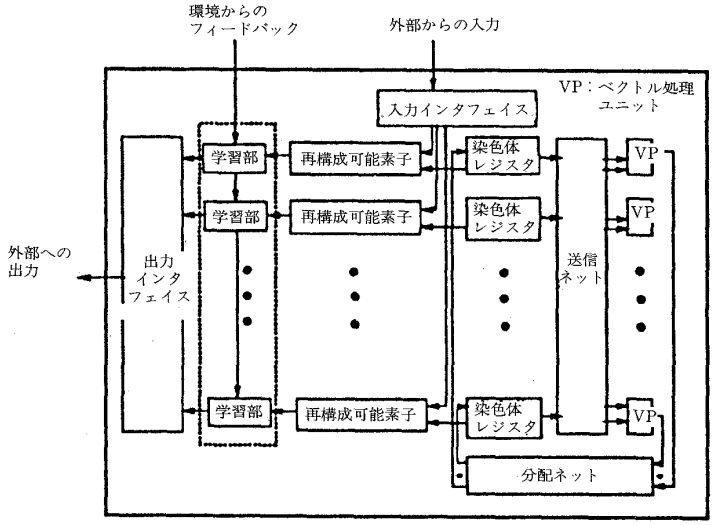

図2 LSI化されたEHWの構成イメージ

以下，各部分について説明する．

環境からの入力は入力インタフェイス（センサも含む）を経て，各々の再構成可能素子に入る．再構成可能素子は，それに付随するハードウェア構成情報によりそのハードウェア構成が一意に決まり，与えられた入力に対応して出力値を生成する．機械学習部は，環境からのフィードバック情報と，各出力値とを比較し，各再構成可能素子がどの程度環境に適合しているか，適応度を計算する．再構成可能素子の具体的イメージは現在のPLDを発展させたものだが，それについては5.2で述べる．

機械学習部の内部はどのような機械学習アルゴリズムをとるかによって大きく変わる．複雑なアルゴリズムの場合にはハードウェア化ができず，マイクロプロセッサが来ることも考えられる．しかし単純な学習では，比較器とカウンタがあれば十分なこともあり，この場合には機械学習部のハードウェア化が可能である． 出力インタフェイスでは，そのときどきでもっとも適応度の高い再構成可能素子の出力が選ばれる．必要があればランダムな選択をここで行なうこともできる．

GA処理部は，機械学習部によって得られた適応度をもとに，ハードウェア構成情報を染色体とみなしてGAを行なう．このGA処理部の特徴は，GAオペレータ自体を並列ハードウェアによって高速化することにあるが，詳細については7章で論じる．

以上のようにEHWでは，複数の回路構成（個体）を同時に維持し，それらの間で選択，淘汰させることにより，環境に柔軟に適応することのできるハードウェアを実現する．このようなハードウェアをLSI化し，一つの汎用チップ部品として種々のシステムの中に組み込むことが，EHWの一般的な使用形態となる．

3．進化するハードウェアの応用

もしEHWのようにハードウェアが環境に応じてその構造（論理回路の構造）を適応的に変化させていくことができるようになれば，人間が予め予測できない未知の環境や極限的な状況において，実時間的な処理能力を発揮可能なハードウェアができあがる．従ってEHWの応用としてまず考えられるのは，深い海の底や宇宙空間といった極限的な状況で機能するハードウェアシステムである．

たとえば深海に対して作業ロボットを送り込むとする．未知の環境であるから，そのロ

ボットの制御回路にしても設計者はデータ不足で設計を行えない．そのような状況における対応として，現在のANIMAT研究が行なっているようなソフトウェアによる低速な制御で許されるならよいが，どうしても実時間的な対応が必要であるとすれば，その制御回路を未定のまま現場に送り込み，あとはハードウェア自身に構造を変えさせるしかないということが起こりうる．

ただし，このような応用形態は，今後EHW研究が実用段階に達した時のいわばりそうであって，現在すぐに実現できるものではない．そこで以下では，近い将来に実現可能なレベルにある応用形態について述べる．

3．1 サブサンプション・アーキテクチャ

ロボット制御で注目されているサブサンプションアーキテクチャ（subsumption architecture；以下SSA）[2]は，自律型ロボットを作るための新しいアプローチであり，単純なモジュールが相互作用をすることによって目的とする動作を達成する．さらに高い機能をロボットに持たすには，新たなモジュールを追加すればよく，その場合には既存のモジュールとの相互作用の調整を行えるアーキテクチャを持つ．たとえば二つのモジュール間で相反する行動をとろうとする場合にはどちらかのモジュールの動きを抑制(subsume)する．

しかし，モジュール数が増えてくると，モジュール間での相互作用の調停（behaviour arbitration）の必要性が指数関数的に増大してくることが指摘されている．人間が調整機構を設計することはもはや困難である．このため，相互作用の調整を自動的に行なう研究が，アルバータ大学やサセックス大学で行なわれている．

以下ではアルバータ大学のCRIP（Collective Robotic Intelligence Project）[6]を例にとり，EHWが相互作用の調整機構をハードウェアで直接的に実現できる可能性があることを示す．

CRIPでは，複数のロボットを協調的に動作させ，一つの仕事をやりとげることを目標とする．たとえば大きな箱をロボット全部で目的地まで押すといったことをさせる．この例の場合，複数のロボットは，次の4つの行動，すなわち，(1)箱を見つける，(2)箱のところまで行く，(3)箱を押す，(4)目的地まで行く，をとる．

これら4つの行動は，それぞれがサブサンプションによる制御をとっている．たとえば(2)の「箱のところまで行く」行動は，図3に示す制御構造を持つ．すなわち，この行動はさらに，他のロボットとの衝突を避けるAVOID行動と，箱のところまで行くGOAL行動からなる．これら2つのモジュールからモーターの制御コマンド（S1からS4）が出る．ここで，e1は箱の発見，e2は箱との接触，o1，o2は他のロボットの存在を関知するセンサーである．

ただし状況によっては矛盾するモーター制御コマンドが出る．そこでその場合にはSと示したノードのところでどちらかのコマンドを抑制している．たとえば他のロボットと衝突しそうになったときは，本来の行動（つまりGOAL行動）よりも，AVOID行動を優先させる．

このような調停を，より大きな規模において自動的に行なわせる枠組としてCRIPではAdaptive Logic Network（ALN）を用い

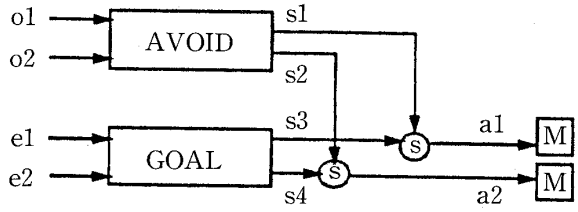

図3 サブサンプション型ロボットの行動の制御構造例

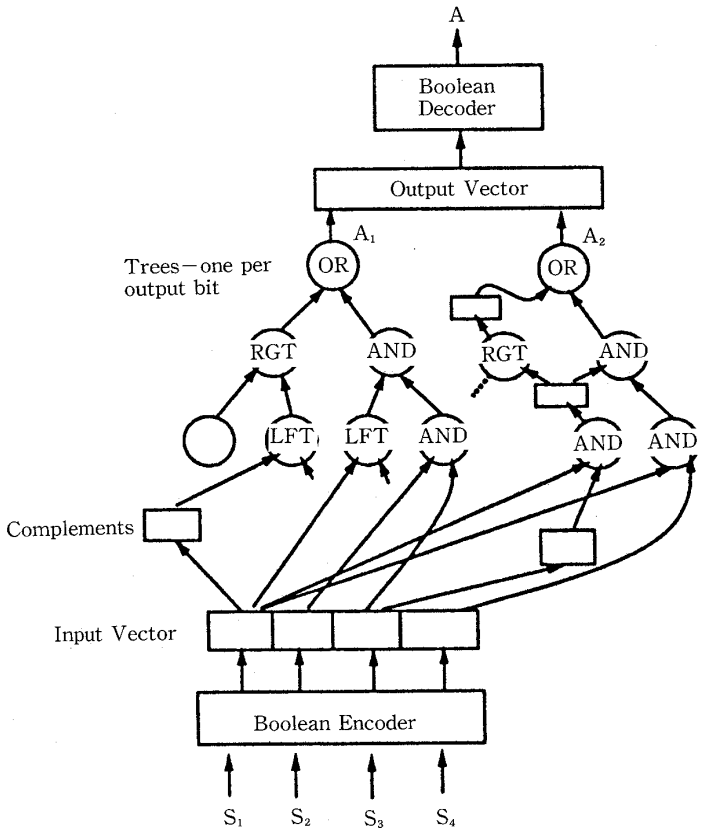

図4 ALNの例

ている．ALNはアームストロングによって提案された一種のニューラルネット(1)であり，CRIPの場合，センサーのパターンを入力，状況に最適なモーター制御コマンドを出力とする．つまり，センサーとモーター制御コマンドの入出力パターンをトレーニングデータとして与え，最適な制御機構をALNによって実現する．上記(2)の行動についてのALNは図4のようになる．

ALNの特徴は，ネットワーク自体がブーリアン2進木を構成しており，ハードウェアに落しやすいことである．つまり，一種のニューラルネットワークがブーリアン関数の回路で実現できることになり，ロボットの実時間制御を，通常のニューラルネットよりも簡単に実現できる可能性が高い．

EHWは後述するように，ブーリアン関数を環境に適応して自動的に合成し，それをハードウェア自身で実現することができる．したがってEHWがALNに対応するブーリアン関数ネットを適応的に作ることができれば，サブサンプション・アーキテクチャにおける調停機構を，EHWで自動的にハードウェアで実現できると考えられる．

3．2 柔らかな連想機構

ここで言う柔らかな連想機構は，ニューラルネットが実現しているような，雑音のある入力に対しても，正しい出力を生成する機能を指す．前節で述べたALNはまさにこのようなinsensitivity，あるいはassociativityをサブサンプション・アーキテクチャに応用していたわけである．したがって，この機能はニューラルネットが適用される応用領域にもそのまま適用できる可能性があり，たとえばALNはこれまでにOCRに応用されている．

ロボットの場合と同様にALNに対応したブーリアン関数をEHWで実現できれば，EHWはニューロチップにかわる柔らかな連想機構，つまり汎用的なパターン認識機構と

して広範囲の用途に適用できると考えられる．

4．進化するハードウェアの意義

EHWの持つ三つの意義について述べる．

4．1 実行速度

適応型学習システムの大半は，適応の結果を表現するのにルール(例：クラシファイア)などの記号表現か，ニューラルネット表現のどちらかを用いている．これらのシステムの実行は基本的にソフトウェアである．したがって，これらが実世界のデータを対象に運用されるとき（つまり学習フェーズが終った後)，その実行はソフトによるため，とても実時間対応の処理には適用できない．この解決の一つとして考えられるのは超並列計算機の導入であるが，記号処理においては超並列計算機を生かすだけの十分な並列性が得られないことが知られている．

しかしながら，EHWは適応の結果が最適なハードウェア構成そのもので表現されるため，実行速度が速い．たとえばGALL16V8チップと呼ぶプログラム可能型論理素子を用いた場合，組み合わせ回路の出力を得るのに10ナノ秒しかかからないし，順序回路の場合は63 MHzのクロック入力まで対応できる．

4．2 有限状態機械を用いた実時間制御

ANIMATにおいては，その制御が基本的に状態遷移に基づくため，内部状態を表現する手段が必要である．これはニューラルネット(Neural Network；以下NN)で実現しようとする場合には難しく，内部状態を含む問題のための学習アルゴリズムの開発が課題の一つとなっている．このような難しさがNNに伴う理由は，NNの構造自体が状態の保持に向いていないためである．NNで状態保持を実現しようとする場合にはフィードバックループを実現するしかなく，これは簡単ではない（例：Hintzの3ビットカウンター[9]．NNに比べると，EHWはハードウェア素子自体が状態であるため（たとえばフリップフロップ)，状態の表現がずっと容易に行なえる．

EHW上に実現される有限状態機械はソフトウェアに比べて著しく速く，したがってリアルタイムの反応を要求される制御にも使える可能性がある．マイクロプロセッサが広く使われる前は，システムの逐次制御を，有限状態機械を実現する順序回路によって行なっていた．EHWはこのような制御系を適応的に学習できる可能性があり，簡単なものであればマイクロプロセッサの制御プログラムをEHWで置き換えうるかもしれない．

4．3 フォールトトレラントなシステム

EHWは未知の環境に対して適用されることが前提となる．つまりハードウェアが論理的にどういった振舞いをすべきなのかは事前にはわかっておらず，環境に適応しつつハード構成を変えていくしかない．このような状況においては従来のハードウェア設計は役に立たない．なぜなら設計者は設計開始時点でシステム論理的振舞いを知っておかねば設計できないからである．この意味でEHWは新しいハードウェア設計方法論のあり方を示しているともいえる．

加えて，EHWが効果的なのは，ハードウェア故障がおきたり，環境からの入力が事故等によって利用できなくなった場合に，ハードウェア構成を再構成できることである．通常のハードウェアでも，故障を意識した設計を行なえるが，その場合，故障の箇所を予測しておくことが必要であるし，またそういった冗長な回路設計が負担になることが多い．EHWではこのような状況に対して，信頼性の低い回路部分や入力部分を適応型学習により回避できる可能性がある．

5．進化するハードウェアの実現法

2章ではEHWの基本構成について述べたが，実現にあたってのポイントはいかにしてハードウェアを再構成していくかである．そのための基本的アイデアについて述べる．

5．1 染色体によるハードウェア構成の決定

現在のANIMAT研究は，環境に対する適応の結果を，ルールなどの記号的形式で獲得したり（たとえばクラシファイアシステム(classifier system)[10]）する形が一般的である．強化学習という点では，ニューラルネットも同様なことが達成可能であり，その場合は学習結果がリンクの重みやネットワークのトポロジーという形で表現される．

これらに対し，EHWの違いは，環境に対する適応の学習結果を，ルールや重みなどではなく，ハードウェア自身が自分自身の構造を変化させることによって表現するということである．

具体的には，EHWではハードウェア設計を，

1．論理素子の「機能」の決定
2．論理素子間の「接続パターン」の決定

であると考える．つまりこの二つが決まりさえすればハードウェアの機能が規定されるわけで，EHWでは，この二つを染色体として表現し，望ましい染色体表現を遺伝的アルゴリズムに基づく学習を通じて適応的に求めていく．この染色体は次節で述べるプログラム可能型論理素子のハードウェア構成を決定するビット列に対応している．もし環境に最適な染色体が選ばれれば，そのビット列によって最適なハードウェア構成が決定される．

5．2 プログラム可能型論理素子

EHWの中で最も重要な部分は，2章で述べた再構成可能素子であって，これはプログラム可能型論理素子（以下PLD）を発展させたものである．

PLDは一般に図5のように，グリッド状の配線機構と，それによって接続される論理マクロセル（QLMC）よりなる．図の左側から電気信号（P2からP5）が入力されるが，グリッド格子点のオン，オフにより，その信号がPLDに印加されるかどうかが決まる（正確にはこのとき，AND入力項が決定される）．続いてこれらの入力は論理マクロセルに入るが，その内部にはAND, ORゲートやフリップが備わっており，特定のビット列を指定することで組み合わせ回路や順序回路を実現することができる．

このように，ハードウェア構成を規定する，配線情報と，素子の機能情報（論理マクロセル）が，PLDではともにビット列で指定できるが，EHWではこれらの情報を染色体として表現する．

この種のPLDは，最近，素子数や種類とも増加傾向にあり，Xlinks, Lattice, Alteaなど商品開発も盛んである．現在では2万ゲートレベルの素子も現れており，速度もピン間15ナノ秒と，十分に高速システムを構成し得るだけの機能を備えてきている．

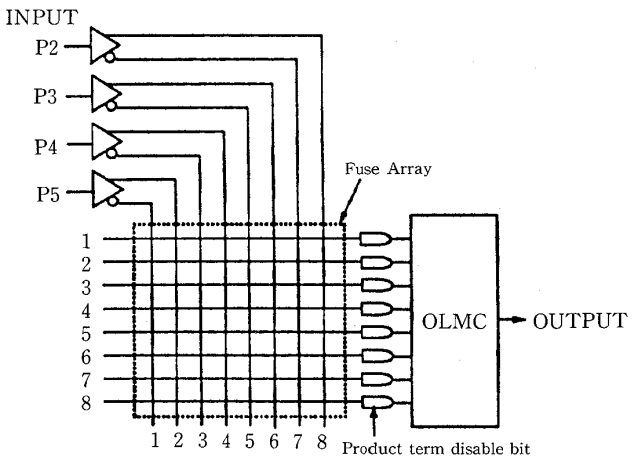

図5 PLDの構成

しかしながら，市販のPLDは必ずしもEHWにとって適切な構造にはなっていない．これは，PLDがプロトタイプ作成時の設計変更の容易性を主目的としているである．まず，ハードウェア構成情報となるビット列が巨大になりすぎ，染色体として扱うには不適当な大きさになることである．20000ゲートのPLDの場合で30Kビットにもなり，GAをもちいるにしても収束は容易ではない．

また，PLDは汎用性を主眼としているため，機能的にはかなり冗長性を含んでおり，これもハードウェア構成情報のビット列を大きくする原因となっている．

このため，EHWを構築するにあたっては，再構成可能素子の設計に重点をおき，進化に適したアーキテクチャをとるようにする．端的には，ハードウェア構成情報のビット列を小さく保ったまま，いかに柔軟な機能を盛り込むかということである．また，現在のPLDはビット列情報をEPROMをベースとした手法によって記録保存しているが，EHWではビット列を頻繁に，かつすばやく書き換える必要があるため，この面での改良も必要である．

6．進化するハードウェアの基礎実験

EHWが有用であるかを決めるのは，ハードウェア構成情報のビット列を染色体とみなしたGAを行うことで，実際に回路が進化するかどうかである．これを確認するために，筆者らはこれまでに組み合わせ回路，順路回路の進化実験を行ない，成功している[3][4]．組み合わせ回路では6マルチプレクサ，順序回路では3ビットカウンタと4状態有限オートマンを扱った．ここでは，6マルチプレクサを例にその基礎実験の概要と結果を示す．

今回，基礎実験に用いたPLDはLattice社のGAL（Generic Array Logic）16V 8と呼ぶ比較的小規模な素子である[8]．

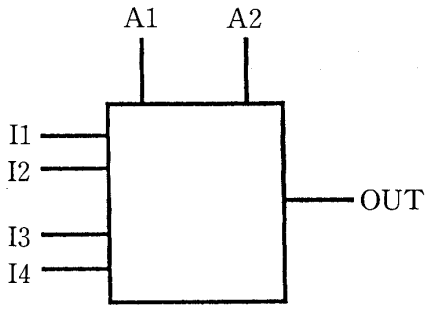

図6 マルチプレクサ

6マルチプレクサはいわゆるBoolean Satisfactory Problemsの一つである．図6に示すように，4つの入力データポートと2つのアドレスビットがあり，アドレスビットの指定により選択された入力データポートのデータが選ばれて出力される[注2]．以下ではGALでの実現にあたっての，染色体表現，および学習方法について述べる．

6．1 染色体表現

6マルチプレクサ問題は1出力の関数であり，GALの一つの論理マクロセルで実現可能である．実際のGALには8つの論理マクロセルがあるが，ここでは論理マクロセルが一つであるとして扱った．これに伴い，ヒューズアレイの大きさも8行×12列とし，フィードバックの線は除外した．この結果，ヒューズアレイを表すための染色体表現として96ビットを割り当てた．また論理マクロセルの機能を決定するアーキテクチャビットの情報として，積項禁止ビット，出力極性ビットなど，12ビットを割り当てた．したがって染色体表現のビット長は108ビットとした．

一つの染色体は，一つのGALチップの構成に対応する．つまり染色体が違えば，別の機能を持つGALとなる．このような染色体

[注2] 6マルチプレクサの「6」の意味は，入力データポート数とアドレスビット数の合計であり，6マルチプレクサの次は11マルチプレクサとなる．

を64個与えておき，各々に対して繰り返し6ビットの入力パターンを提示する．そして各染色体に対し，次節で述べる方法で適応度を決める．

6. 2 学習方法

まず，適応度の計算は次のように行なう．6マルチプレクサ問題では全部で64個の入出力パターンをトレーニングデータセットとして用いる．それぞれのパターンを与えるたびごとに，GALの回路の出力が，トレーニングデータの出力と合致するかを調べ，もしあっているならその染色体の適応度を高め，あっていないなら減ずる．最大値は64であるが，実際に適応度として用いる時は百分率としている．

GALチップが目標とする性能を達するまで，以下のサイクルを繰り返す．

- ステップ1　各染色体の適応度の計算
- ステップ2　適応度に比例する確率で二つの染色体を選ぶ．
- ステップ3　遺伝的アルゴリズムにより新しい染色体を二つ得る．
- ステップ4　古い染色体集団中の染色体を，ステップ3の染色体と置き換える．

上記のサイクルが繰り返されるのは，集団のうちどれか一つの染色体が100%の正解率に達するまでであり．つまり，各染色体に64のパターンを与える時に正解率を計算している．

6. 3 実験結果

染色体が100個の集団を用意し，上述の実験方法を適用した．正解率100%の固体が現れるまでGAを繰り返した．一様交差（uniform crossover）を20%の確率で用いた．また突然変異はビットあたり0.1%の確率とした．

図7は進化の一例であり，集団の平均適応度（正解率）とベストの適応度が学習につれてどう変化するかが示されている．約2000サイクルで収束している．GALに対してこの値だけ書き換えが繰り返されることになるが，GALは実用上最大で10万回程度書き換えが可能なので，この収束の仕方であれば十分に実用になると考えられる．

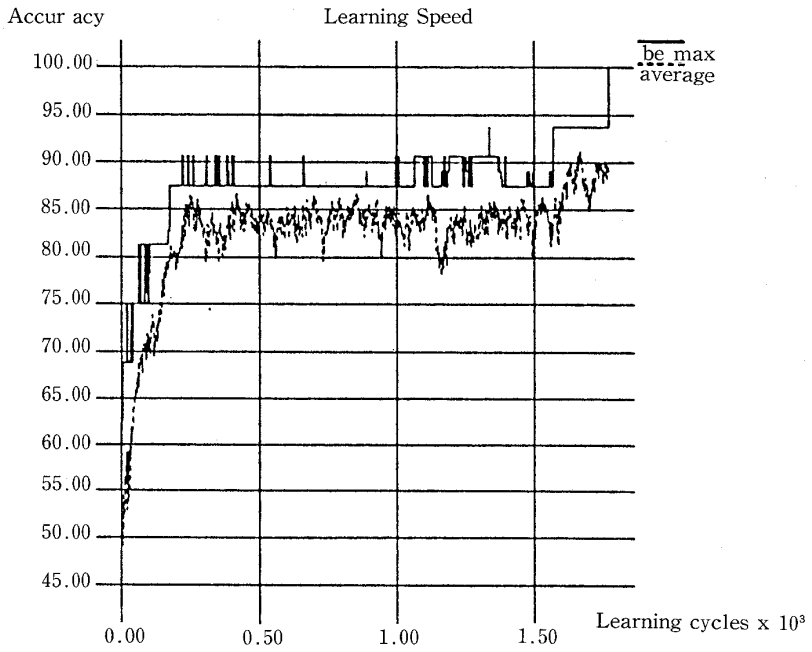

図7　マルチプレクサ回路の学習過程

7．GA 並列マシン

2章ではEHWとして次の5つの構成要素をあげた：再構成可能素子群，ハードウェア構成情報(染色体)群，GA処理部，機械学習部，入力および出力インタフェイス．現在，これらのうちのGA処理部をGA並列マシンとして実装することを計画している．ここではGA並列マシンの開発目的，基本構想について述べる．

7．1 EHW用のGAの問題点

このEHWを遺伝的アルゴリズムを用いて実現するとき，次の問題をよく検討する必要がある．

1．問題点1　GA操作が適応度評価よりもはるかに時間がかかる．

2．問題点2　長い染色体の処理

まず問題点1について述べる．GAの処理時間を，適応度の評価に要する時間Tfと，GA操作(選択，交叉，突然変異等)に要する時間Tgの合計と考えた場合，従来，GAの応用では一般にはTfが支配的であるとされている．このためGA操作自体の高速化についてはこれまでほとんど顧みられなっか．

しかし，EHWは適応度評価よりもGA操作が支配的な応用(TgがTfに比べてかなり大きくなる)であると，EHWでは実時間処理が要求されることが多くなると考えられるため，GA操作の高速化が必要である．この高速化の視点に立った場合，GA操作は並列処理的にみてSIMD(Single Instruction Multiple Data stream)に分類でき，並列ハードウェアを構成しやすいメリットがある．

なお，適応度評価よりもGA操作が支配的な応用というのはEHWに限った特殊なものではない．たとえば有名なDe Jongのｆ1関数でも，GA操作が適応度評価時間の約40倍にもなっており，GA応用が増えるにつれてこのようなケースも増加すると考えられる．

問題点2として挙げた長い染色体の処理は，EHWに特有のものである．PLDの構造を決定するビット列は，小規模なPLDでも数千ビット以上であり，将来何らかの圧縮方法をおそらく用いるにしても，従来のGA応用では考えられなかった長さの染色体を取り扱う必要がある．長い染色体を扱うということは，それだけ大きな探索空間を扱うことであり，当然収束に要する時間もかかる．これまでの実験でも二百数十ビットの染色体の場合で，収束に一昼夜を要することもあった．

7．2 GA並列マシンの特徴

前節で述べた二つの問題点の解決策として，ここではGA操作自体の並列化をとる．従来の並列GAでは適応度計算の並列処理が主体で，GA操作自体の並列化は行なっていなかった．この並列化の基本的な考え方は次の通りである．通常の逐次GAで交叉や突然変異などのGA操作を行なう場合には，染色体集団の中から2つの染色体を選び，そのペアにGA操作を行なって新たな染色体を作り，これを繰り返す．しかし，これらの処理は逐次的に行なう必要はなく，染色体ペアがすべて決まったら(mating)，並列にGA操作を実行できる．

この並列処理のための機構は図2に含まれているが，この部分のみを図8に示す．すなわち，複数の染色体レジスタが接続ネットワークを介して，ベクトル処理ユニットに結合されている．接続ネットワークは，任意の内部接続パターンを定めることができる．

その接続パターンは，別の計算機によるmatingで決められ，この後，GA操作の並列処理が次のように実行されていく．

まず染色体レジスタは，1ビットづつ，データを接続ネットワークへ送りだす．このとき染色体レジスタは一斉に動作している．各

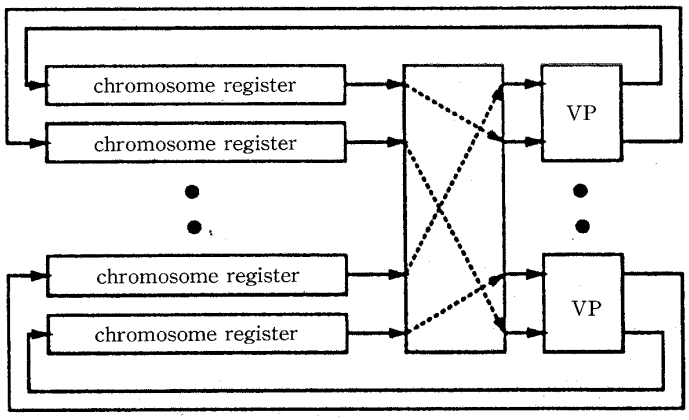

図8 GA 並列処理機構

ビットは接続ネットワークの内部接続パターンによって，いずれかのベクトル処理ユニットに到達する．ベクトル処理ユニットは2つの染色体レジスタからの2ビットを受けとり，交叉や突然変異を行なう．この処理もビット単位で実行される．このようにクロックレートでGA操作が並列に実行されるのがこのマシンの特徴であり，GAの並列処理方式としてはじめてのものである．

接続ネットワークの接続パターンの設定を変えることによって，エリート保存戦略やランクベーストなどGAにおける選択の圧力を制御することができる．また交叉の種類もベクトル処理ユニットを制御することにより，一点，二点，一様など各種扱うことができる．

7. 3 実装についての検討

このGA並列マシンを実装する上での一番の問題は，接続ネットワークをどう構成するかである．つまり，図8からわかるようにこのネットワークは単なるクロスバースイッチネットであるものの，その実装は簡単ではない．つまりネットへの入出力の数が問題で，この数は染色体レジスタの本数の2倍に等しいが，最低でも128を考えている．つまり128個の染色体集団に対して並列を行なう．この場合128×128のクロスバーを構成すれば良いことになる．しかしこの接続ネットワークの回路基板に，GA並列マシンの他の基板か

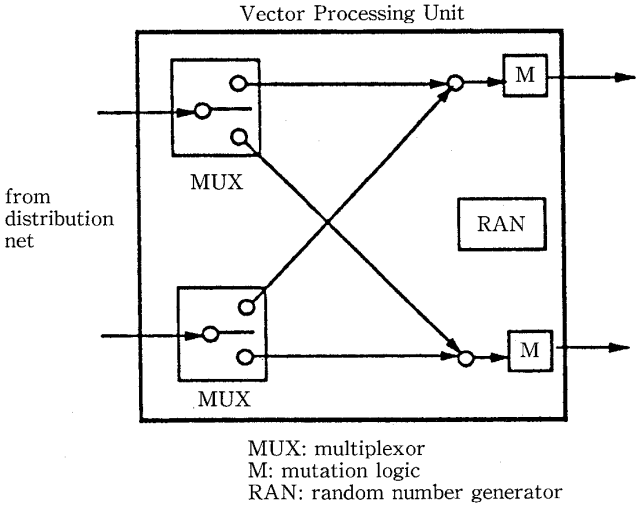

図9 ベクトル処理ユニット

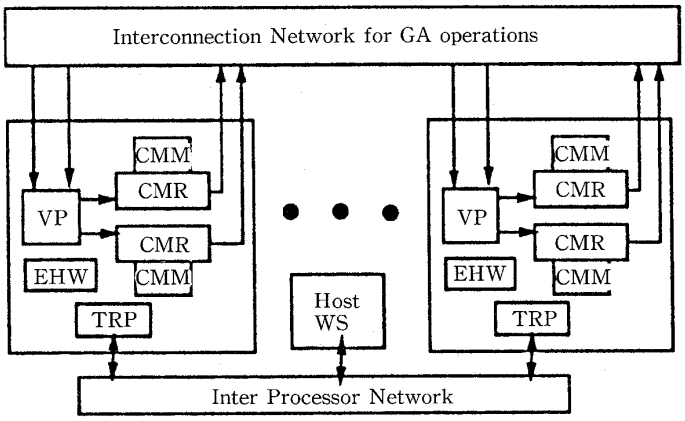

図10 GA並列マシンの概要

ら信号線が10メガヘルツ程度の速度で集中することになるので実装上問題が出てくる．このため，64×64のクロスバLSIがTIやLSIロジックから市販されてはいるものの，本GAマシンの規模からすると選択しにくい．

現在は図10のような構成を想定して，種々の接続ネットワークアーキテクチャの検討を行なっている．そこでの要件は，最低10メガでも動作し，かつネットワークがスケーラブル，かつ転送が非同期であることである．この接続ネットワークにつながるのがベクトル処理ユニットや染色体レジスタであり，トランスピュータ（T 9000を予定）等とともに一枚の回路基板に置く．

染色体レジスタには染色体メモリが付随し，ここから一つ染色体が取り出されて染色体レジスタに格納される．これらの制御には，1万ゲート規模の大規模プログラマブル論理素子を用いて専用プロセッサ的なハードウェアを作り込むことを計画している．

トランスピュータは適応度の計算のほか，EHW関連の制御や，他のトランスピュータと高速度で接続してGAにおけるグローバルな処理にあたらせる．

8．展望と課題

進化するハードウェアという概念は，計算機の新しい設計方法の提案をも含む広範な概念であり，さまざまな研究アプローチがこの範疇に含まれる．たとえば，ニューラルネットの自動成長に関する種々の研究[5]も，その大半が実装にまで至っていない（あるいは実装を意図していない）にしても，進化するハードウェアと捉えることができよう．しかし，本稿で示したアプローチはニューラルネットに限らない，より一般的なハードウェアシステムを目指している点，さらに具体的なハードウェアアーキテクチャに踏み込み汎用チップを目標とする点で従来の研究とは異なる．

本稿のアプローチは，実際に複数のハードウェア個体をLSIチップの中に同時に維持し，実時間的にそれらの個体間の淘汰，選択を繰り返すものである．そして，学習とGA操作をも1チップ内に取り込み，1つの汎用チップとして種々の応用に対応することを目指す．この方法での課題は，個体のハードウェア機能を決定する染色体情報を短く保った上で，多様な応用にも対応可能な汎用論理素子のアーキテクチャを考案することである．また，環境からのフィードバックに基づく強化

学習が進化を図る上で極めて重要であるが，これに関してはそのハードウェア化も含め，検討すべきことが多く残されている．

参考文献

(1) Armstrong, W. and Gecs'ei, J.: Adaptation algorithms for binary tree networks, *IEEE Trans.,* Vol. SMC−9, No.5, (1979).

(2) Brooks, R.: A robot that walks: Emergent behavior from a carefully evolved network, *Neural Computation,* (1989).

(3) Higuchi, T., Niwa, T., Iba, H., Hugo, D., Tanaka, T., and Furuya, T.: Evolvable hardware with genetic learning, SAB92, (1992).

(4) Higuchi, T., Niwa, T., Iba, H., Hugo, D., Tanaka, T., and Furuya, T.: Evolvable hardware: genetic based generation of electronic circuitry at gate and hardware description language levels, *ETL Tech. Report,* (1993).

(5) Harvey, I.: Evolutionary robotics and SAGA: the case for Hill Crawling and Tournament Selection, *CSRP 222,* Univ. of Sussex, (1992).

(6) Kube, C.R. and Zhang, H.: Controlling collective tasks with environmental cues, *IROS 93,* (1993).

(7) Langton, C.G. et al. (ed.): *Artificial Life II* Addison-Wesley, (1992).

(8) Lattice Semiconductor Corporation: *GAL Data Book,* (1990).

(9) Spofford, J.J. and Hintz, K.J.: Evolving sequential machines in amorphous neural network, in *Artificial Neural Network* (ed. Kohonen, et al.), Elsevier Science Publishers, (1991).

(10) Wilson, S.: Classifier systems and the animat problem, *Machine Learning,* Vol.2, pp.199−228, (1987).