計算機アーキテクチャ　131－4
（１９９８．９．２１）

VLSIアーキテクチャと設計自動化技術の将来

小栗 清
NTT光ネットワークシステム研究所
〒239-0847 神奈川県横須賀市光の丘 1-1
0468-59-4232
oguri@acm.org

あらまし　集積度のさらなる向上を生かすにはこれまでの枠組みを大きく変更しなければならない。考えられる解は２つある。第１はタイミングを抽象化して、すなわち非同期回路として、実時間に依存しない順序のみのインタフェースを持つ階層化や部品化を行うことである。こうすると設計の困難さをソフトウェアによる階層化や部品化に近づけることができる。第２は大規模なモノの均質な構成要素となり得るゲートよりも大きい何かを定義し、設計と製造をその上下に分けることである。本稿では第２の考え方に基づく新しいアーキテクチャ、プラスティック・セル・アーキテクチャ(Plastic Cell Architecture: PCA)の提案を行う。

キーワード　FPGA、可変構造計算機、PCA、布線論理、セルラオートマトン、非同期回路

Breakthrough in VLSI Architecture and Design Automation Technology

Kiyoshi Oguri
NTT Optical Network Systems Laboratories
1-1 Hikarinooka Yokosuka Kanagawa 239-0847 Japan
+81-468-59-4232
oguri@acm.org

Abstract

VLSI Architecture and Design Automation Technology will have to change against for ever-increasing complexity, high frequencies, connection delay dominance, low voltages, and surging currents. We propose a new general-purpose homogeneous architecture based on the cellular automata model and programmable logic devices which divides design and manufacturing into two parts and decreases complexity extremely. It consists of a dual-structured array of cells accommodating a fixed "built-in part" and a freely programmable "plastic part". We call this composition the "Plastic Cell Architecture" (PCA).

key words　FPGA, Configurable computer, PCA, Wired logic, Cellular automaton, Asynchronous circuits

1 はじめに

競争に勝つためには、良くて安いものをできるだけ短期間で開発する必要がある。“良くて安い”とは、高機能、高性能、高拡張性、低消費電力、小さい、軽い、低コストなどを意味する。これらを達成するために、VLSIの高集積化、設計の自動化、方式改良などがさらに押し進められる。新素材や新素子の発明は、大きな変化をもたらすが予測が難しいため議論から除くと、変化の中心は微細加工技術の進歩に伴うVLSIの高集積化である。

高集積化によって設計規模が増大するため、その効率化のためには、設計の抽象度の向上、階層化、設計の再利用などが必要となる。抽象度の向上はソフトウェアの設計と同様の機能に関する抽象度向上だけでなく、物理的タイミングに関する抽象度の向上も不可欠となる。現状のクロックによる順序からクロックによらない順序のレベルが要求される。一方、階層化と設計の再利用は、まず少なくともタイミング設計の方式が決定されていることが必須であり、さらにこの前提となるタイミング設計の方式によって効果が異なる。設計規模が増大すると、実時間による規範なしのタイミング設計は論外としても、すべてをクロックで制御することは次第に困難となってきている。

高集積化によって面積と動作周波数当たりの消費電力は変化しないが、トランジスタの動作速度が向上し、これを有効に使うための設計を行う、すなわち動作周波数を上昇させるのが普通なので、面積当たりの消費電力は増大してしまう。消費電力の増大は製品の価値を下げるだけでなく、放熱の限界からさらなる高集積化の妨げになる。無駄な変化による電力消費は極力抑えるべきで、クロックがその第一候補となっている。またメモリ構造は最も離れた場所へのアクセスを最も近い場所へのアクセスと同等の時間で行うために大きな電力を消費しているということが広く認識されつつある。

さらに、高集積化によって配線の単位長当たりのRCによる遅延が増大する。回路の接続関係が同じならば、配線の長さも減少し、結果として配線遅延は変化しないが、トランジスタは高速動作するので、配線遅延の割合が大きくなる。配線は論理設計とレイアウト設計に依存するため、論理設計とレイアウト設計を同時に最適化することが重要になるが、良いアルゴリズムは知られていない。

配線の遅延が論理設計において支配的になったのはこれが初めてではない。CMOSによる高集積化が進む前のECLによる高速システムの設計では、配線の距離による電磁波としての伝播遅延が支配的であった。それがCMOSによる高集積化により距離が問題とならなくなり、しばらくは配線の長さよりも論理素子の接続関係が遅延を決定したので、多段論理式を最適化することが回路を最適化することとなり、論理設計の自動化が進んだ。また論理ゲートを構成単位としてもLSIの全体が扱える大きさであったこと、LSI内を単一のタイミング抽象度で扱えたことも論理設計の自動化が成功した理由である[1]。

以上説明したように、集積度のさらなる向上を生かすにはこれまでの枠組みを大きく変更する必要のあることが分かる。規模の増大を単なる階層設計でまかなおうとすると、構成要素の大きさも大きくなり、次第に均質な構造ではなくなってくる。たとえばHW/SW協調設計による解決では様々な構成のメモリ、プロセッサ、バス、DSP、専用回路などが構成要素となり、設計の自動化は困難になると思われる。特にハードウェアでは機能的動作と物理的動作が共に正しく設計できる必要があるために、階層設計がソフトウェアの場合よりも難しい。さらに、仮に設計は階層的に行えたとしても、VLSIを階層的に作っていくことはできないため、階層設計はソフトウェアの場合ほど徹底できない。

それではどうするか、考えられる解は2つある。第1はタイミングを抽象化して、すなわち非同期として、実時間に依存しない順序のみをインタフェースとする階層化や部品化を行うことである。こうすると設計の困難さをソフトウェアによる階層化や部品化に近づけることができる。第2は大規模なモノの均質な構成要素となり得るゲートよりも大きい何かを定義し、設計と製造をその上下に分けることである。広い意味でプロセッサとメモリはこの何かに相当するが、この何かによって布線論理が構成できることが望ましい。以下にこの第2の考え方に基づく新しいアーキテクチャ(プラスティック・セル・アーキテクチャ(Plastic Cell Architecture: PCA))について述べる。

2 プログラム論理と布線論理

ある論理的機能を実現しようとする時、さまざまな方法を採ることができる。もっとも手軽なのはCPU (Central Processing Unit)とメモリによるプログラム論理である。CPUとメモリを用意しておけば物理的構成を変えないで様々な機能を実現することができる。論理回路により直接、機能を実現する布線論理を使えばより高性能なものを得ることができる。しかし、物理的にものを作る行程が必要となり、手軽というわけにはいかない。さらに布線論理による専用ハードウェアの性能は、単一のCPUの性能向上に数年で追い越されてしまう場合が多い。

専用ハードウェアがCPUに対して大幅な性能向上をなし得ないのはメモリを使用しているからである。アドレスによる読み出し／書き込みサイクルで特徴づけられるメモ

リは非常に単純な機能ながら柔軟性に富み、これこそがノイマンアーキテクチャの本質である。CPU はそのメモリを最大限有効に使えるように作られた専用マシンである。従ってメモリを使っている限り、問題毎に専用マシンを設計しても CPU を大幅に越えることはできない。

それでは性能や柔軟性において CPU とメモリを越えるためにはどのような構造を考えれば良いのであろうか。

アプリケーションをプログラム論理に展開する場合と布線論理に展開する場合の共通点と相違点を図 1にまとめた。まず、コンパイラや論理合成系によって生成されたメモリイメージ、回路イメージを実際のメモリや ASIC、FPGA に構成するところが、当然ではあるが、各々、記憶機能、工場、記憶機能と異なる。もう一つ重要な相違点がある。それはプログラム論理ではメモリ上のデータ構造を手続きの実行に伴って動的に構成できる点である。複雑な情報処理はデータの表現を生成、操作することによって行われるため、この相違点は本質的である。

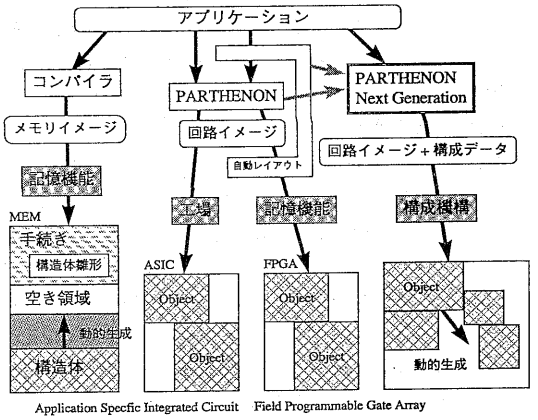

図 1: プログラム論理と布線論理の共通点と相違点

3 汎用情報処理システムの二重構造

動的構成、すなわちある初期状態から出発して、後から新しい機能や構造を追加できるということは、構成の変更を行える部分 (可変部) が存在し、かつその構成操作のための基本機能があらかじめ備え付けられている (本能) ということである。これを汎用情報処理システムの二重構造と呼ぼう。CPU とメモリの例では、メモリの内容が前者に相当し、回路として作られた CPU とメモリの機能が後者に相当する。

布線論理のみによって、この汎用情報処理の二重構造を形成するために、図 2の構造を導入する。図 2は布線論理を変更可能な部分 (可変部) とあらかじめ特定の機能が組み込まれている部分 (本能) にわけて形成することを意味する。本能としては、少なくとも可変部を任意の布線論理として構成できるだけの機能がなければならない。

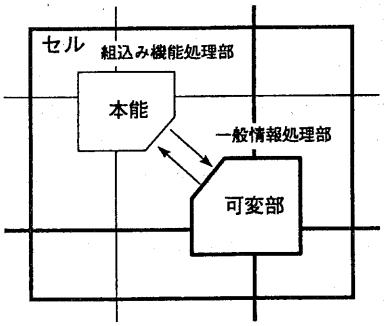

図 2: プラスティックセル

図 2の構造が単独で存在するものとすると、構成処理は逐次的なものとなってしまう。そこで、構成処理そのものにも並列性を導入するために、図 2の構造が複数個連結されたものを考える。可変部については、可変部そのものを任意の結合網とできるので均一な結合でよい。また本能についても可変部が一様である以上、特別の構造を導入することはできない。よってここでは VLSI の作りと相性がよいと思われる 2 次元のメッシュ結合を選ぶ (図 3)。

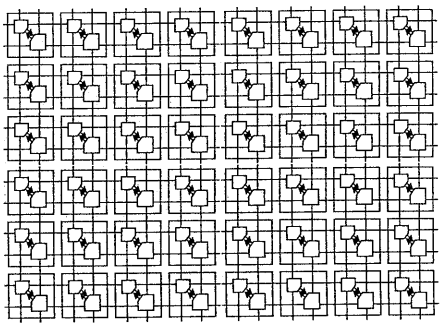

図 3: プラスティックアレイ

本能の構成機能により、可変部は機能、サイズ共に任意の布線論理回路に構成できる。一旦構成された布線論理回路は他の部分をさらに構成したり、すでに構成された布線論理回路と通信し、連携して処理を進めることができる。すなわち、図 2、図 3の構造により、布線論理と並列処理の枠組みの中で汎用情報処理システムの二重構造を作ることができる。

図 2、図 3による汎用情報処理システムをプラスティックセルアーキテクチャ (Plastic Cell Architecture: PCA) と名付ける。Plastic は「形成力のある」という意味で用い

た。以後、図2をプラスティックセル(または単にセル)、図3をプラスティックセルアレイ(または単にセルアレイ)と呼ぶ。

プラスティックセルアーキテクチャは規則正しい構造を持つため、VLSI向きであり、メモリに匹敵する集積化の可能性もある。さらに機能の構成を製造後に行うため、製造欠陥が与える影響をより少なくでき、一層の高集積化に対応できる。また2次元のメッシュは容易に折り畳むことができ、LSIの3次元方向への拡張も可能である。

4 プラスティックセルアーキテクチャの設計

4.1 オブジェクトとメッセージ通信

動的に生成される布線論理回路をオブジェクトと呼ぶ。オブジェクトを構成するセルの可変部は互いに直接結合されて回路を形作る。オブジェクトの境界が定義できるためにはオブジェクトは境界を越えて可変部の結合を持ってはならない。オブジェクト間の通信は本能を介してのみ可能である。本能は可変ではないので通信はあらかじめ決められたメッセージ通信の形態をとる(図4)。

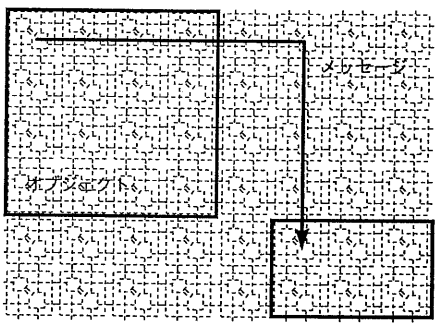

図 4: オブジェクトとメッセージ

4.2 オブジェクトの利用に必要な処理

複雑な並列動作は既に構成されたオブジェクトが、テンプレートから新しいオブジェクトを生成し、利用し、削除することによって行われる。その手順は次の様になる。

1. まず必要な空き領域を探索し確保する。
2. 構成情報を注入して損傷が発生しないように回路を構成する。
3. 回路が完成したら、誤動作しないように回路を初期化する。
4. 本能とオブジェクトとの間の通信ゲートを開く。
5. 処理をメッセージとして送付する。
6. 解放メッセージを注入し領域を開放する。

4.3 経路指定によるメッセージの伝播

経路情報を共有する複数の情報をまとめて一つのメッセージとし、メッセージは処理の単位である命令から構成されるものとする。メッセージは

- 経路設定情報
- メッセージ本体
- 経路設定解除命令

から構成される。経路設定情報はメッセージの後続部分をどのように伝えるかをセルの本能部分に記憶させるもので、経路設定命令(west命令、north命令、east命令、south命令)からなる。セルの本能部分は一旦経路を設定されるとすべての情報をそのまま経路設定された次のセルに送り続ける。すなわち、シフトレジスタとして動作するようになる。経路設定命令は経路を設定するセルで消費され、次々とシフトレジスタを構成していく。経路設定解除命令(clear命令)は経路設定されたセルでは消費されずに伝播されて次々とシフトレジスタを構成していたセルの本能部分を初期化する。

複数メッセージのデッドロックを避け易くするため、各セルの本能は図5に示すように独立に動作可能なwest、north、east、southの4つの入力ポートを持ち、それぞれは4つの出力ポートへメッセージを転送できるものとする。出力ポートが競合したときは待ちが発生する。この構成によって直交する2つのメッセージなどを扱うことが可能となる。メッセージの経路をeast-southであるものとwest-northであるものに制限すると、循環待機条件が成立しなくなるのでとりあえず、デッドロックを回避することができる。もっと効率の良い方法の発見は今後の課題である。

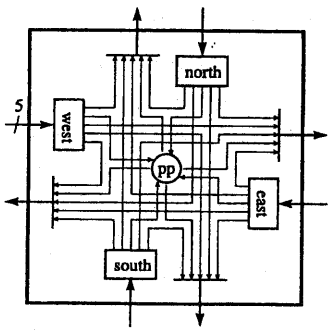

図 5: 本能部分のポート構造

4.4 すべてのメッセージと命令

可変部によって実現できないものだけを本能の機能とするという方針をとると、

- 構成、書込みメッセージ
- 解放メッセージ
- 読出しメッセージ
- オブジェクトへのメッセージ

の4つのメッセージが必要十分である。空き領域の管理などは可変部を構成することによりその役を行わせるものとする。いずれにしてもプラスティック・セル・アーキテクチャの上に様々な詳細アーキテクチャが設計可能であり、どのようなものが最善なのかは今の時点では不明である。上記のメッセージについて以下に説明する。

構成、書込みメッセージ 構成メッセージの本体は回路構成部、リセット部、本能との通信開始部からなる (図 6)。回路構成部は構成命令 (config_in 命令)、構成データ、経路設定命令の繰返しであり、構成命令を受取ったセルの本能は後続の構成データを可変部の LUT (後述) に書込む。構成命令と構成データはすべてこのセルで消費される。このセルは後続の経路設定命令を受取り消費しシフトレジスタの一部となる。これを繰返して必要な回路を形成する。回路の形成過程では無意味な回路や誤った動作をする回路が形成される可能性があるが、回路の形成が完了した時点では正しい回路となっている。リセット部はこの回路に対して必要なリセットを特定の LUT へ値を書込むことで行う。回路はリセットが行えるように合成するものとする。本能との通信開始部は通信を行う位置の本能に通信許可命令 (open 命令) を送る。

書込みメッセージは構成メッセージからリセット部と通信開始部を除いたものであり、可変部を記憶機能として、すなわちメモリオブジェクトとして利用するときに使用する。この場合、LUT には回路としては意味をなさない値が設定されるが、通信許可命令を実行しなければ外部に影響は及ばない。

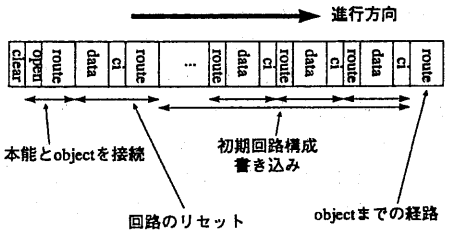

図 6: 構成、書き込みメッセージ

解放メッセージ 解放メッセージの本体は通信禁止命令 (close 命令) である (図 7)。回路が形成されたまま外部との連絡を停止する。

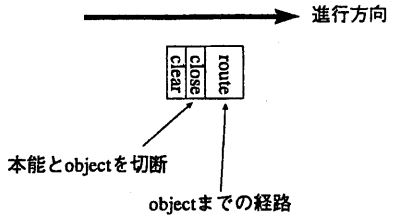

図 7: 解放メッセージ

読出しメッセージ 読出しメッセージでは構成出力命令 (config_out 命令) が中心的役割を果す (図 8)。config_out 命令を受取った本能はこれを消費すると共に受取ったことを記憶する。この記憶は clear 命令に対する動作にだけ影響を与える。読出しメッセージではまず最初の経路設定情報が読出しを行うオブジェクトの位置までをシフトレジスタとする。次に config_out 命令と経路設定命令の組の必要個数が読出しを行うセルに印を付けシフトレジスタとする。そのあと任意の命令が続く。この部分には読出しデータを渡すべきオブジェクトまでの経路などが入る。最後が clear 命令である。config_out 命令を受取っていたセルは LUT の内容を clear 命令の前に挿入する。

オブジェクトへのメッセージ オブジェクトへのメッセージは経路設定情報の最後が pp_out 命令となっており、シフトレジスタを可変部へつなぐ (図 9)。先に各セルは 4 つの出力ポートを持つと述べたが実際は、可変部への出力ポートを持つので出力ポートは 5 つである。pp_out 命令は経路設定命令に属する。また同じく、先に各セルは 4 つの入力ポートを持つと述べたが可変部からのメッセージが存在するのでそのための入力ポートがあり、全体で 5 つの入力ポートを持つ。

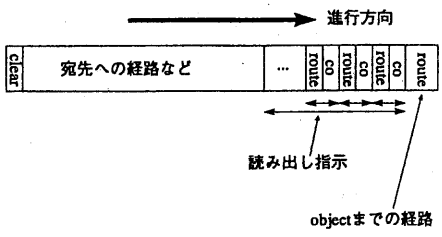

図 8: 読出しメッセージ

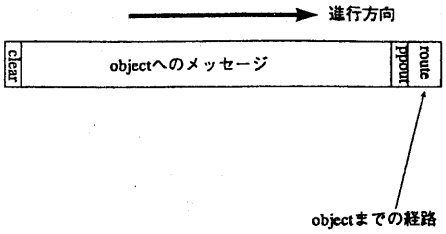

図 9: オブジェクトへのメッセージ

命令　命令は各メッセージの説明に現れた、clear 命令、pp_open 命　令、pp_close 命　令、config_in 命　令、config_out 命令、west 命令、north 命令、east 命令、south 命令、pp_out 命令の 10 命令である。これらのために 5 ビットを使用する。clear 命令のために 1 ビットを使用し、残りの 9 命令を 4 ビットにエンコードする。こうすれば構成データに clear 命令を除く任意のパターンが使えるようになり、都合がよい。

4.5　可変部の構成

従来の FPGA は論理関数を LUT(Look Up Table) すなわち小規模のメモリで実現する場合であってもフリップフロップと配線を専用に持ち、これらを接続するセレクタやトランスファゲート、そしてそれらを構成するメモリから構成される。また、LUT には構成情報の格納のためのデコーダと LUT を構成するためのデコーダを個別に設けることが多く、また誤った構成情報から回路を守るための付加回路なども必要である。これらは必要なものではあるが必要なものを機能ごとに集めてきたという構造であり、その機能が使われないときは全く無駄になってしまい、メモリの様な均質な構造でもない。このためこれまでの FPGA はメモリやゲートアレイに比べると集積量や速度に於いて大きく劣っている。このため現状ではプロトタイピングやエミュレーションなどのような傍流で利用されることが多く、その将来性に疑問も持たれている。ここに、そもそも FPGA をもっと均質な構造にすることはできないのかという疑問がある。図 10にトライステート出力を持つ LUT だけからなる可変部のセルの構造を示す。LUT の内容を設定することによりセルは結線や論理ゲートそしてラッチなどに構成することができる。これを図 5の構造に近づけた図 11はまさに LUT だけからなる構造であるとともに、LUT の利用度が高い。これを Sea of LUTs と呼ぶ。LUT である小規模メモリへの構成情報の書込み読出しはシフトレジスタ構成で行う。Sea of LUTs の構成ではメモリと同様の集積度が期待できる。この結果、可変部をメモリオブジェクトとして使うときは完全にメモリと同様の集積度を使うことができる。一方可変部を布線論理オブジェクトとして使うときも従来 FPGA よりも高い LUT 集積度を使うことができる。性能については今後の検討を必要とするが、規則正しい構造はすべてに於いて有利であると考えている。

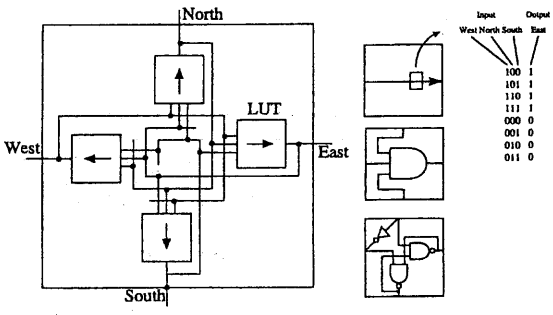

図 10: トライステート出力を持つ LUT だけからなる可変部のセルの構造

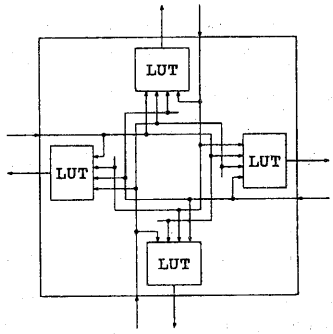

図 11: LUT だけからなる可変部のセルの構造

4.6　非同期アーキテクチャ

可変部ではレジスタやラッチを含むすべての布線論理を LUT のみで形成する。このためレジスタやラッチの配置位置や方向の自由度は高いが、グローバルクロック線を用意することができない。LUT を通過する信号をクロックに用いることは全く現実的でないので、可変部は非同期回路とせざるを得ない。しかしながら PCA は 2 次元のアーキテクチャであり、その拡張性を自然に生かす、地理的に広がりを持つ通信と処理をカバーするという将来像を考えると、グローバル通信は非同期とせざるを得ず、本能、可変部ともに非同期とすることは必然かもしれない。

PCA では回路を自動合成することが前提であるため、出来るだけトップダウンに構成できる非同期回路アーキテクチャを採用する。同期回路からグローバルクロック信号を

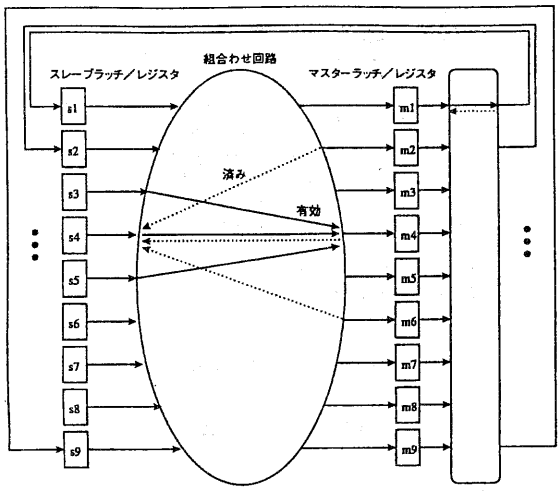

図 12: 非同期回路の構成

削除し、それを等価なタイミング信号に置換えることにより非同期回路を構成する方法を以下に示す。

ここではSutherlandのマイクロパイプラインとは違ってレジスタや組み合わせ回路の性質は動作時に変化しないものとする(レジスタが組み合わせ回路の一部になったりしない)。そうするとレジスタには有効データが存在しているか存在していないかのいずれかになる。

またすべての因果関係を正しく評価するために、有効データは使われて次の有効データを生成するものとする。すなわち、有効データが使われもせず消失したり、使われたのに存在し続けたり、無から突然出現したりすることはないものとする。

さらに、有効データが生成された(到着した)ことは信号の変化でしか伝えられないので、簡単のため、有効データが生成される前には必ず、有効データが存在していない時間があることとし、この有効データの「なし・あり」をそのまま有効データ到着の信号とする。

有効データなしの時間を設けることにより、有効データが存在しない時に限って有効データを生成するという簡単なメカニズムで(有効データを消失させる)上書きを防げる。

このような原則に基づく非同期回路構成方式を2相式という。

さて、同期回路ではレジスタに設定された有効データを数クロックに渡って利用することがある。有効データであることを到着でしか表現できない非同期回路では、このような場合、同期回路の1つのレジスタに対し少なくとも2つのレジスタを用意して、その間で有効データをループさせなければならない。

同期回路のレジスタをタイミングレベルで詳細に見るとマスタラッチとスレーブラッチからなっているので、簡単のため、非同期へ変換した回路もすべてマスターラッチとスレーブラッチの構造を持ち、処理を行う組合わせ回路はスレーブラッチの出力からマスターラッチの入力までの間にあるものとする。この構造を図12に示す。

以下ラッチとレジスタを区別する必要のない時はレジスタという。2相式の非同期回路ではレジスタ上の有効データのありなしが交互に配置されたときに最大性能となるので、初期化信号のハイレベルにより、すべてのマスターレジスタを有効データ存在、すべてのスレーブレジスタを有効データ不在の状態とし、初期化信号の立下がりにより、すべてのスレーブレジスタに有効データを生成し、すべてのマスタレジスタの有効データを消去することとする。初期化信号の変化は同時ではないのでこのような工夫を行う。

グローバルクロックに代るタイミング規則を例を使って説明する。s3,s4,s5のデータが加工されてm4へ書込まれる場合は、s3,s4,s5へ有効データが到着したこと、加工が終了したこと、m4が消去されたことのすべてを待って書込む。この書込み信号は済み情報として左へ送られる。s4の有効データをm2,m4,m6が利用していたとすると、s4の有効データはm2,m4,m6からの済み信号を待って消去する。

2線式を使う場合は組合わせ回路内の信号がすべて無効状態へクリアされたことを待つ必要があるのでもう少し複雑となる。

レジスタをグラフの節、有効データ利用の関係をグラフの枝とするときグラフが連結であるとすると、順序に関し元の同期回路と同じ動作をすることを証明できる。またこの構成はデッドロックフリーであることを証明できる。さらに2線式の論理回路はユネイトであるので論理簡単化にこの性質を有効に使うことができる。

5 関連する研究

プラスティックセルアーキテクチャは汎用情報処理システムの二重構造を、まず可変部を布線論理とし、次にこれを2次元のメッシュに敷き詰めて得られたものであり、FPGAの技術とセルラオートマトンの技術を結合させたものになっている。その生い立ちから当然のことではあるが、セル1個のものは可変部をメモリとすればノイマン型コンピュータとなり、セルアレイから可変部をのぞくとセルラオートマトンとなる。セルアレイの可変部をメモリとするとメッシュ結合網のマルチプロセッサとなる。可変部

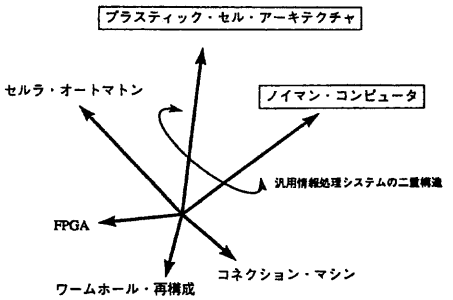

図 13: 関連する技術

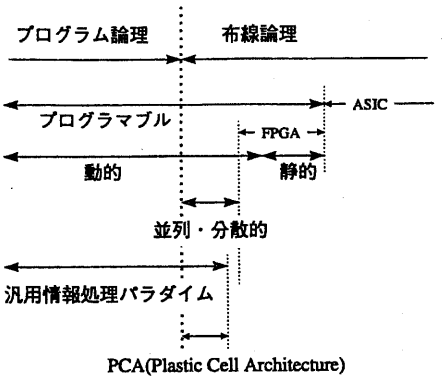

図 14: PCA の位置付け

から本能への制御パスを除き、本能を単なる可変部の構成パスとするとFPGAとなる。可変部を布線論理としてこれを集中制御により一斉に同一のものに構成する場合に限定するとコネクションマシンとなる。

SRAMベースのFPGAは外部の制御部により任意の布線論理に構成可能であり、動作中の構成を許すものも存在する。一方、外部制御ではなく内蔵の分散制御により必要に応じて構成を行うワームホール動的再構成などの研究が存在する [3][4][5][6]。また、セルラアルゴリズムを高速化するためにセルラオートマトンのセルに可変な構造を追加する研究が存在する [2]。プラスティックセルアーキテクチャとFPGAやワームホール動的再構成、可変構造セルラオートマトンとの違いは、汎用の情報処理機構であるための条件を議論しているかどうかにある。PCAの汎用情報処理モデルとしての位置付けを図13、図14に示した.

6 おわりに

PCAにより最初に議論した問題がどのように解決されるのかをまとめる。最初の議論では非同期化と均質な階層の導入は別の解決法としたが、結果としてPCAは非同期システムであるのでPCAはその両方の特徴を持つ。まずLSIの設計と製造を、繰り返し構造を持つPCAのレベルに閉じこめたので、詳細物理レベルの設計は格段に容易になった。メモリと同様の徹底的な最適化が可能である。PCA上の設計は論理設計もレイアウト設計もオブジェクト単位であり、LSIの規模とは無関係になった。クロックを廃したので無駄な動作による電力消費は回避される。またブロック間の非同期のインタフェースは物理的時間によらずブロック間を連携させることができ、ソフトウェアと同様の階層化部品化が可能になった。次に規則正しい構造により、従来よりもレイアウトの自由度が減少し、かつレイアウト結果の評価項目が単純化された。これは論理最適化と連携したレイアウト最適化アルゴリズムの開発がそれだけ容易になる可能性があることを示している。

謝辞

非同期システムについては東京大学の南谷 崇教授、中村 宏助教授の指導を受けた。また本論文の内容はNTT光ネットワークシステム研究所の永見 康一、伊藤秀之、小西隆介、塩澤 恒道、Norbert Imlig、中田 広、稲森 稔の各氏、そしてNTTコミュニケーション科学研究所の名古屋彰氏、京都大学の中村 行宏教授、メリーランド大学の中島和生教授との議論に基づくものである。

参考文献

[1] Y. Nakamura, K. Oguri and A. Nagoya: "Synthesis From Pure Behavioral Descriptions," High-Level VLSI Synthesis, Edited by R. Camposano and W. Wolf, Kluwer Academic Publishers, pp.205-229, June, 1991 http://www.kecl.ntt.co.jp/car/parthe/

[2] 中野博嗣,"動的可変パスを持つ並列計算機上の定数時間アルゴリズム," 情報処理学会誌 Vol. 38, No. 11, pp. 1019-1025 (Nov. 1997)

[3] H. Schmit: "Incremental Reconfiguration for Pipelined Applications," in Proc. IEEE Workshop on FPGAs for Custom Computing Machines, pp. 47-55, April, 1997

[4] W. Luk, N. Shirazi and P. Y.K. Cheung: "Compilation Tools for Run-Time Reconfigurable Designs," in Proc. IEEE Workshop on FPGAs for Custom Computing Machines, pp. 56-65, April, 1997

[5] J. Burns, A. Donlin, J. Hogg, S. Singh and M. Wit: "A Dynamic Reconfiguration Run-Time System," in Proc. IEEE Workshop on FPGAs for Custom Computing Machines, pp. 66-75, April, 1997

[6] R. A. Bittner, Jr and P. M. Athanas: "Computing Kernels Implemented with a Wormhole RTR CCM," in Proc. IEEE Workshop on FPGAs for Custom Computing Machines, pp. 98-105, April, 1997