計算機アーキテクチャ 135－5
（1999．11．26）

パネル討論：　やわらかいハードウェア

－ デバイス，アーキテクチャ，設計技術，応用研究の展開 －

モデレータ：　末吉敏則[1]
パネリスト：　天野英晴[2]，今井正治[3]，小栗清[4]，樋口哲也[5]，本村真人[6]

[1]熊本大学　sueyoshi@cs.kumamoto-u.ac.jp
[2]慶応大学　hunga@am.ics.keio.ac.jp
[3]大阪大学　imai@ics.es.osaka-u.ac.jp
[4]ＮＴＴ未来ネット研究所　oguri@exa.onlab.ntt.co.jp
[5]電子技術総合研究所　higuchi@etl.go.jp
[6]ＮＥＣシリコンシステム研究所　motomura@mel.cl.nec.co.jp

あらまし

本パネル討論の「やわらかいハードウェア」という題は，多くの人々を困惑させるかも知れない．ハードウェアは硬い，ソフトウェアは柔らかい，というのが今日の常識であろう．ところが，PLD や FPGA という何時でも何処でもカスタム化できるプログラマブルデバイスの登場によって，応用毎にハードウェア構成を適応的に変更するアプローチが現実味を帯び，注目を集めている．

本パネル討論では，この分野の第一線でご活躍の５人の専門家を迎えて，「やわらかいハードウェア」を取り巻くデバイス，アーキテクチャ，設計技術，応用の研究が今後どのように展開するか議論する．

キーワード　やわらかいハードウェア，プログラマブルデバイス，FPGA，リコンフィギャラブル・コンピューティング，進化型ハードウェア，HW/SW コデザイン

Panel Discussion:　Flexible Hardware

– What Will Be the Future Development of the Researches on Devices, Architectures, Design Technologies and Applications ? –

Moderator : Toshinori Sueyoshi[1]
Panelists : Hideharu Amano[2], Masaharu Imai[3], Kiyoshi Oguri[4], Tetsuya Higuchi[5], Masato Motomura[6]

[1]Kumamoto University　sueyoshi@cs.kumamoto-u.ac.jp
[2]Keio University　hunga@am.ics.keio.ac.jp
[3]Osaka University　imai@ics.es.osaka-u.ac.jp
[4]NTT Network Innovation Lab.　oguri@exa.onlab.ntt.co.jp
[5]Electrotechnical Lab.　higuchi@etl.go.jp
[6]Silicon Systems Research Lab., NEC　motomura@mel.cl.nec.co.jp

Abstract

The term of "flexible hardware" which is the subject of this panel discussion may let many people confuse. It is probably today's common sense that hardware is hard and software is soft. However, the reconfiguration of hardware becomes possible after the appearance of PLD/FPGA, and attracts a great deal of attention as an approach to pursue the adaptability for the application with the reconfigurability.

In this panel discussion, we have five distinguished specialists on this field as the panelists, and will discuss several significant issues which will be facing at the "flexible hardware" of the future.

key words　flexible hardware, programmable device, FPGA, reconfigurable computing, evolvable hardware, HW/SW codesign

1. 問題提起

PLD(Programmable Logic Device) や FPGA(Field Programmable Gate Array)という何時でも何処でもカスタム化できるプログラマブル・デバイスの登場によって，ハードウェアによって動的な適応性を求めるアプローチが現実味を帯びてきた．とりわけ，FPGA が出回り始めた 1990 年代初頭から，アプリケーション毎にハードウェア構成を適応的に変更（最適化）できるリコンフィギャラブル・コンピューティング(Reconfigurable Computing)の研究開発が世界各地で活発化し，コンピュータ分野のみならず半導体分野，家電分野，計測制御分野などからも注目を集めている[1]．さらには，最適化を自律的に行ってしまう進化型ハードウェア(Evolvable Hardware)も提案され，着実に成果を上げて産業応用への展開が狙える段階まで研究が進み，世界的にも関心が高まっている[2]．

本パネル討論では，「やわらかいハードウェア」を取り巻く技術分野に関し，デバイス，アーキテクチャ，設計技術，応用の研究がどのように展開するか議論する．パネリストにはこれら研究分野において第一線でご活躍の 5 名の方にお願いしている．それぞれのお立場から，これまでの「やわらかいハードウェア」に関して得られた研究実績・知見に裏打ちされた忌憚の無いご意見や新たな方向性をご示唆頂けると信じている．モデレータからの質問としては，次のような現状認識と将来像についての問いをお送りしている．

(1) 「やわらかいハードウェア」のイメージ，長所（優位性），短所は？
(2) 「やわらかいハードウェア」のテクノロジ・ドライバとなるシーズ技術は？
(3) どのようなニーズがあるか？　キラー・アプリは何か？
(4) それを具現化するための課題，およびその解決策（アプローチ）は？

ポジショントークにおいて，上記質問に対するご回答も併せて期待したい．なお，「やわらかいハードウェア」には馴染みのない方のために，討論に入る前の「やわらかいハードウェア概説」以外にも，下記のトピックについて簡潔な解説をお願いしている．「やわらかいハードウェア」のイメージや研究動向を捉えて頂く一助にならば幸いである．

- リコンフィギャラブルロジック LSI　　本村真人 氏
- HW/SW コデザインとやわらかいハードウェア　　今井正治 氏
- 進化型ハードウェア　　樋口哲也 氏

以下では，これら解説の前置きとしての「やわらかいハードウェア概説」のための資料として，PLD デバイスならびにリコンフィギャラブル・コンピューティング，そして課題と研究動向について述べる．

2. PLD デバイスの構造と特徴

ここでは，FPGA（あるいは CPLD）の構造と特徴を明確にする[3]，[4]．

2.1 全体構成

代表的 FPGA/CPLD の全体構成を図 1 に示す．FPGA は AND-OR アレイ構造をもつ PLD ブロックよりも小規模な基本論理ブロックを組み合わせて所望の論理回路を実現する．設計の自由度が高く，ゲートアレイに近い特徴をもつことから FPGA(Field Programmable GA)と呼ばれる．一方，基本論理ブロックとして PLD ブロックを組み合わせた構造の大規模 PLD は CPLD(Complex PLD)と呼ばれる．但し，デバイスによってはこの中間的な特徴に位置するも

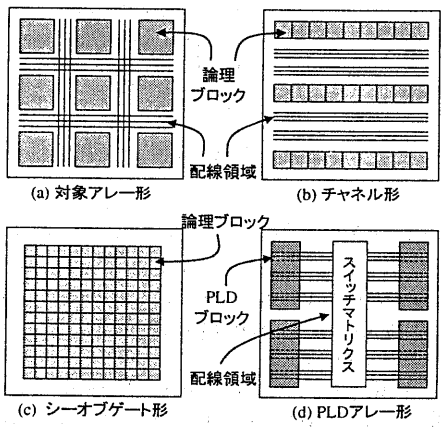

図 1　FPGA/CPLD の全体構成

のもあり，FPGA あるいは CPLD と明確に区別できないこともある．

2.2 プログラム素子

FPGA/CPLD のカスタム化を実現するプログラム素子は，現状でチップ内に数十万～数百万個は必要となる．また，プログラム素子を配線の接続に用いる場合，高速動作を期待するためには配線の時定数が小さくなければならない．さらに，製造ならびにユーザの使い勝手も重要である．現在の代表的プログラム素子としては， SRAM セル，アンチヒューズ， EPROM トランジスタ，EEPROM トラジスタの各方式がある．

SRAM 方式は MOS トランジスタに SRAM セルを付けて，その導通／不導通をセルの記憶内容によって切り替えるパストランジスタを使用する．再書込み回数は無制限だが，不揮発ではなく，電源をオフにすると構成データが失われる．このため，電源投入時に外部から構成データを毎回ロードしなければならない．これは欠点でもあるが，現在では ISP(In System Programmability，基板上に装着したままで書換え可能)機能がむしろ利点と見なされることも多い．製造技術的には，標準プロセスが直ちに FPGA/CPLD の製造技術として採用できるので，比較的容易に性能向上が見込まれる．

FPGA 固有のテクノロジとして利用されているアンチヒューズは，プログラム前は不導通状態で，プログラム後は導通状態になるスイッチで，合金ヒューズの逆特性になることからアンチヒューズと呼ばれる．これには，酸化膜系を用いるものとアモルファスシリコンを用いるものがある．アンチヒューズが面積や動作速度に関しては有利であるが，消去／再書込みができない．また，製造技術的には標準プロセスの変更が必要という欠点がある．

EPROM 方式と EEPROM 方式はいずれもフローティングゲートをもつ MOS トランジスタ構造をスイッチに使用し，フローティングゲートに電荷を注入してプログラムする．消去方法が異なり，前者は紫外線照射により，後者は制御ゲートに電圧をかけて消去する．つまり，両者とも消去／再書込み可能だが，EEPROM は ISP 対応であるのに対し，EPROM は ISP 対応ではない．最近では，プログラム素子として FLASH EEPROM も用いられている．通常の EEPROM はアドレス指定の消去タイプであるが，FLASH は構造を簡略化して高速・高集積化し，その代わりに一括消去型とした EEPROM である．

2.3 デバイスアーキテクチャ

FPGA/CPLD では，ゲートアレイと違って，メーカ毎にかなり異なったデバイスアーキテクチャを採用している．これらはプログラム素子と密接な関係があり，配線遅延のペナルティをできるだけ小さくする工夫がアーキテクチャに取り入れられ，基本論理ブロックには粒度(granularity，回路規模）が小さい（細かい）ものから大きい（粗い）ものまである．

配線遅延のペナルティを低減する手段としては，基本論理ブロックの粒度を大きくする方法が一般的である．基本論理ブロック内部では遅延時間の小さい回路を実現できるため，粒度が大きいほど高速な回路を実現できる．但し，粒度を大きくすると高速化はできるが，一般に内部のゲート使用効率が悪くなるため，明らかなトレードオフが存在する．一方，アンチヒューズでは配線遅延が比較的小さいので，基本論理ブロックには細粒度のマルチプレクサやトランジスタアレイが用いられる．

また，プログラム素子の使い方には，配線同士を接続するスイッチ素子として用いる場合と，ルックアップテーブルの論理決定やマルチプレクサの切替えなどを行う電位の設定として用いる場合がある．前者の場合，動作速度は配線の時定数に左右される．一方，後者の電位設定では抵抗値や容量は動作速度に直接影響しない．そこで，SRAM のようにプログラム素子の抵抗値や容量が大きい場合には配線の直接的な接続を減らし，プログラム素子を基本論理ブロックの定義する電位設定に利用するアーキテクチャをとる必要がある．つまり，基本論理ブロックとしてルックアップテーブルやマルチプレクサを用い，その論理仕様定義をプログラム素子で行うことにより，配線の時定数が直接遅延に関与しないようにする．

3. リコンフィギャラブル・コンピューティング

リコンフィギャラブル・コンピューティングとは，対象とするアプリケーションに合わせて自分自身のハードウェア構成を変更することによって自然な形で処理を行う計算パラダイムの総称である．これは専用ハードウェアを開発する方法と比べて柔軟であり，それぞれのアプリケーションに応じて最適な構成をとることができる．このため，専用ハードウェアのもつ高い性能と汎用計算機のもつ高い柔軟性とを兼ね備えた計算パラダイムとして大きな期待が寄せられている[5].

3.1 概念と特徴

現在では何らかの機械的な情報処理を必要とする場合，ハードウェア化するよりはコンピュータ上でソフトウェアを書いて実現する方が一般的となっている．これは手軽で短時間にでき，かかるコストも低いというのが主な理由である．しかし，実現性の検討をしてみると，それでは速度などの要求性能が満たせないケースも少なくない．そのような場合には，時間とコストをかけてもアクセラレータを追加したり，専用コンピュータを導入することになる．

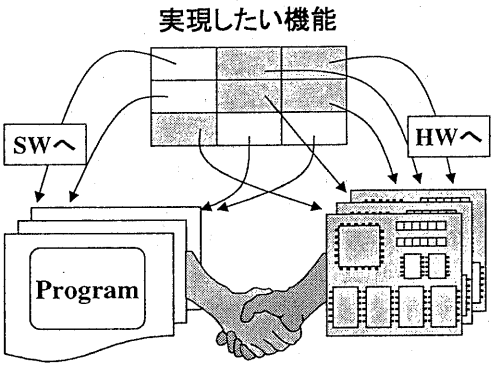

図２　リコンフィギャラブル・
コンピューティングの概念

見方を変えると，前者の方法はアーキテクチャ（ハードウェア）を固定してアルゴリズム（ソフトウェア）を変更する方法と言える．一方，後者の方法はアルゴリズム（ソフトウェア）を固定してアーキテクチャ（ハードウェア）を変えることで要求性能を満たそうとする方法と言い換えることができる．こうした試行錯誤は日常的に行われるが最良の方法とは言い難い．できることなら，最初から最適なアーキテクチャとアルゴリズムを一度に実装できることが望ましいが，これはアプリケーション毎に専用コンピュータを開発し続けることと等価で，現在のコンピュータの枠組みでは時間とコストの面から難しい．

ところが，PLD や FPGA というプログラマブル・デバイスの登場によってハードウェア（回路）自身も容易に変更できるようになった．最初から最適なアーキテクチャとアルゴリズムを一度に実装できる可能性がでてきた訳である．リコンフィギャラブル・コンピューティングとは，まさにこの「やわらかいハードウェア」の再構成性を最大限活用し，図２に示すようにアプリケーションの処理内容に合わせてハードウェア／ソフトウェア分割を適応的に最適化する計算パラダイムである．

リコンフィギャラブル・コンピューティングの特徴は，ハードウェアの持つ高い性能とソフトウェアの持つ高い柔軟性を合わせ持つところにある．したがって，図３に示すように固定的なハードウェアを必要最小限にし，ソフトウェアと「やわらかいハードウェア」の境界を変更することで最も効率的な処理を行うことができる．もちろん，すべての処理を「やわらかいハードウェア」で行うことも考えられる．その結果，以下に示す効果が期待できる．

a)高性能化

従来の汎用プロセッサは逐次的な命令実行を基本とし，データ幅（語長）ならびに演算器構成が固定であるため，メディア処理や通信処理が不得手である．一方，リコンフィギャラブル・コンピューティングはアプリケーション毎にハードウェア構成を最適化できるので，並列的なデー

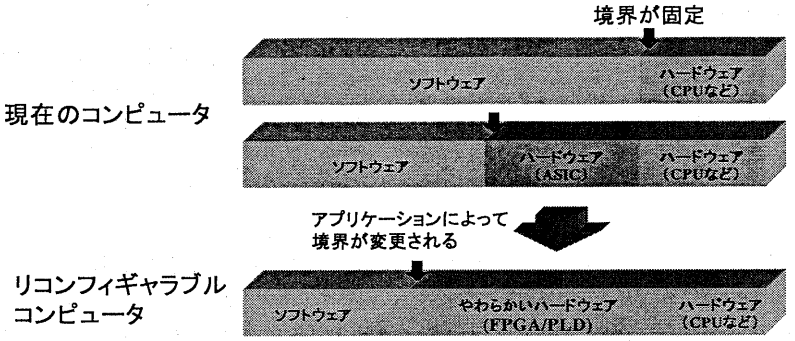

図 3　SW と HW の境界の変化

タ処理，可変データ幅，多様な演算器構成，ならびにデータの流れの最適化が可能となり，高性能化が図れる．

b)低コスト化，低消費電力化

複数の処理を高速化したい場合にそれらすべてを ASIC に作り込むと，コストや消費電力などの点で要求仕様を満たせないことがある．しかし，それらすべてが同時に動作しないのなら，リコンフィギャラブル・コンピューティングでは再構成性という特徴を活かし必要時に実装することでハードウェア量を低減でき，低コスト化や低消費電力化が見込める．

c)耐故障性の向上

ASIC の場合，一箇所でも故障すれば全体が機能不全に陥る可能性が高い．しかし，リコンフィギャラブル・コンピューティングでは故障箇所を使用しないように再構成することによって，耐故障性の向上が図れる．

3.2 利用形態による分類

では，実際の利用形態は如何なるものであろうか．利用形態はプログラマブル・デバイスを計算処理のどの部分に適用し，どのような役割を持たせているかで以下のように分類できる（図 4）．

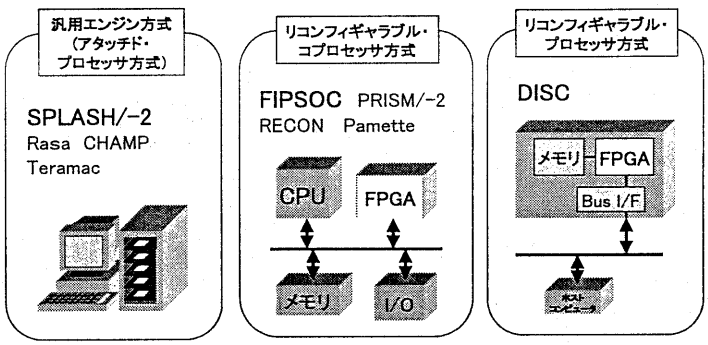

図 4　利用形態による分類

a)汎用エンジン方式（アタッチド・プロセッサ方式）

汎用計算機のベクトルファシリティのように，ワークステーション等のホスト計算機に付加して計算集約的な処理やプロセッサの不得手な処理を任せる方式である．この方式はホスト計算機に対して独立性が高く，いわゆるサーバ・クライアントモデルで運用する場合が多い．代表的なシステムには米 SRC の Splash，Splash2 などがある．

b)リコンフィギャラブル・コプロセッサ方式

汎用マイクロプロセッサと組み合せて利用する点に特徴があり，アプリケーションに適した命令の実装や入出力処理などをハードウェア化することで高性能化を図る方式である．この方式は，あくまで処理主体はマイクロプロセッサにあり，プログラマブル・ロジックはそれを補完することが役割となる．最近はマイクロプロセッサとプログラマブル・ロジックを 1 チップに収めた製品も発表されている．

c)リコンフィギャラブル・プロセッサ方式

プログラマブル・デバイスが主体に処理を行う方式である．処理方式としては，アプリケーションに特化した命令を適応的に実装して処理を行う方式，アプリケーションを完全に回路として実装する方式，さらには内部に RISC プロセッサや DSP などの回路を必要に応じて読み込む方式などがある．

4. 課題と研究動向

研究の進展に伴って技術的課題も明らかになってきた．アイデアはその単純さにおいて魅力的である．つまり，従来のように計算機の固定的な構造・機能に合わせてプログラムを書くのではなく，「やわらかいハードウェア」という特徴を利用してその問題向きにハードウェアを再構成するのである．しかし，いざ設計を始めるとリコンフィギャラブル・コンピューティングの単純さは，それ自身の複雑さを呈し始めている．結局，高水準言語からの翻訳工程は機械語よりもさらに低いゲートレベルまで進まねばならず，その開発工程の複雑さは研究者を悩ますことになった．その結果，従来の FPGA コンピューティングともいうべきアプローチだけでなく，最近では脱 FPGA アプローチという新たな潮流も現れてきている．リコンフィギャラブル・コンピューティングは従来のいわゆるノイマン型コンピュータとは動作原理から異なるため，システム，デバイス，設計手法，アプリケーションの全分野に亘って課題が山積し，それらに対する研究が同時進行している状況である．以下では，多岐にわたるリコンフィギャラブル・コンピューティングの研究分野のうち，デバイス，開発環境，アプリケーションについて言及する．

4.1 デバイスの課題

リコンフィギャラブル・コンピューティングの最もチャレンジングな面は，実行時に回路を再構成する動的な性質である．しかし，従来の FPGA は本来，リコンフィギャラブル・コンピューティング向けに設計されておらず，要求性能を満たしているとは言い難い．特に，動的な再構成を行う場合には，回路の書換えに時間がかかることが性能抑制原因として明らかになっている．また，容量的にも速度的にも決して満足いくものではない．それゆえ，この分野の研究は盛んで，リコンフィギャラブル・コンピューティング向けデバイスのアイディアが多数出されている．デバイスに関する代表的な課題を整理すると，

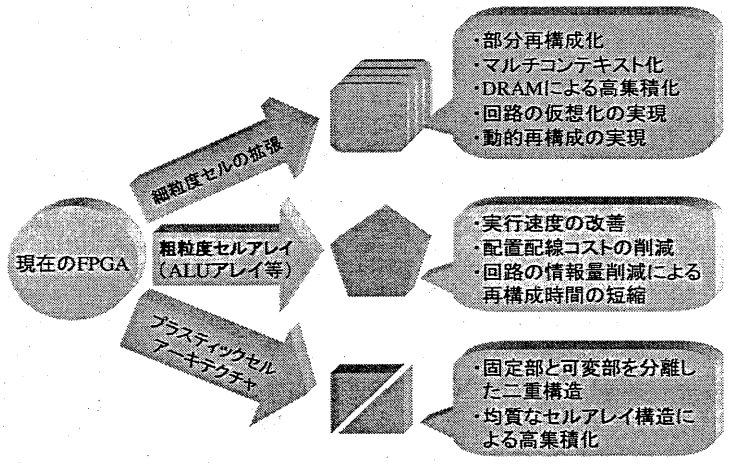

図 5 リコンフィギャラブル・デバイスの研究動向

1)回路の仮想化を実現すること
2)再構成時間を短くすること
3)回路の動作速度を上げること

と言える．これらの課題を解決すべく，図 5 に示すような研究アプローチがある．

第一のアプローチは，現在主流の細粒度セルのデバイスを拡張する方式である．これには，部分再構成デバイスとマルチコンテキスト（回路情報を複数内蔵する構成）デバイスという二つの

流れがある．部分再構成デバイスは実行時に回路の一部を書き換えることができ，再構成時間の隠蔽や回路の仮想化を実現できる．部分再構成デバイスは既に一部の FPGA ベンダで実用化され始めているが，まだ不十分で改良の余地がある．一方，マルチコンテキストデバイスは，内部に持つ複数の回路情報を実行時に切り替えることによって，再構成時間の大幅な短縮や回路の仮想化を実現する．さらに，回路情報を貯えるメモリに DRAM を用いることにより，占有面積を再構成論理部より格段に少なくできるため，実装密度の向上も期待できる．マルチコンテキストデバイスの研究には，MIT による DPGA(Dynamic Programmable Gate Array)の提案，NEC による DRL(Dynamically Reconfigurable Logic)チップの提案･試作，慶応大の DRAM 混載 FPGA の提案等がある．

第二のアプローチは粗粒度セルを用いる方式で，脱 FPGA というべきアプローチである．つまり，FPGA における数ゲートから十数ゲート程度のセルに代わって，プログラマブルな ALU などの高機能演算器をセルとして敷き詰める方式である．この方式ではセルが大きいため，配置配線の領域が相対的に少なく，構成情報自体がコンパクトになる．よって，生成時間と実行時の再構成時間が短い．算術演算を中心としたアプリケーションには実行速度，実装密度とも有利である．その代わりに演算自由度が低いことから，アプリケーションを選ぶといった欠点もある．この方式の研究は 90 年代半ば頃から行われ，独 Kaiserslautern 大，英 Imperial College，広島市立大，東北大等から成果報告がなされている．

さらに，新たなアプローチを採る研究も行われている．例えば，NTT では固定論理と可変論理を分離したセル構造を持つデバイスに関する研究などがある．このように様々なデバイス・アーキテクチャが提案されているが、先のデバイスに関する課題をすべてクリアしているものは未だない．今後はアーキテクチャのみならずデバイス・テクノロジも含めた進展を期待したい．

4.2 開発環境の課題

リコンフィギャラブル・コンピューティングでは，アプリケーションの処理内容に合わせてハードウェア構成を最適化するため，ハードウェア開発とソフトウェア開発を同時に行わねばならない．理想的にはシステムレベルの仕様記述による高位合成を前提とするハードウェア／ソフトウェア・コデザイン環境[7]が望ましいが，現状では ASIC 開発（LSI 設計）とソフトウェア開発の両方がユーザに課せられることになる．多くの場合，ASIC の設計環境を流用しているのが実情であり，ハードウェア部分はハードウェア記述言語で直接記述するものも珍しくなく，多大な時間と労力が必要である．

一方，最近ではリコンフィギャラブル・コンピューティングを念頭において Java 言語をシステム仕様記述言語とする研究も盛んになってきた．Java などのオブジェクト指向言語が持つオブジェクトの生成と削除のメカニズムは，固定的な ASIC 等では対応が難しいが，リコンフィギャラブル・デバイスの動的な書換えの概念と親和性が良く，リコンフィギャラブル・コンピューティング向けのシステム仕様記述言語として期待されている．

4.3 アプリケーションの課題

現状では前述したデバイスの問題もあり，すべてのアプリケーションにおいて劇的な効果があるわけではない．現在は，キラー・アプリケーションを模索している段階である．有望なアプリケーションとしては，メディアデータ処理，通信のビットストリーム処理，パターンマッチング処理，シミュレーション，暗号処理などの汎用マイクロプロセッサが苦手とする分野が挙げられる．汎用エンジン方式の Splash は，DNA 塩基配列の距離計算でスーパーミニコンの約 2,700 倍，スーパーコンピュータの約 320 倍の性能を達成している．また，暗号解析処理でも三菱電機の RASH[8]などは，マシンレベルで比較すると 400MHz Pentium マシンの 2,000 倍以上の性能を出している．今後の応用研究では効果的なアプリケーションのさらなる開拓と集積が重要である．また，それらのアプリケーションの多角的な分析をシステム構成やデバイス・アーキテクチャへ反映させることが待望される．

5. 将来展望

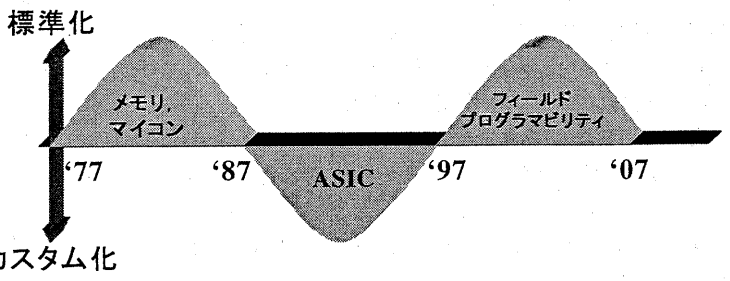

図 6　Makimoto's Wave

現在の LSI 技術は，「ムーアの法則」によれば 3 年で 4 倍という速度で集積度を上げている．いよいよ 1 チップに数千万個，さらには数億個のトランジスタが集積されるようになり，システム自体が 1 チップに載る，いわゆるシステム LSI の時代の到来である．しかし，システム LSI というからには用途を限定せざるを得ず，集積度が増大するほどその傾向は強くなる．すなわち，システム LSI 時代の到来とは多品種小量生産の時代を迎えることを意味しており，少品種大量生産向きの集積回路製造技術は本質的矛盾を抱えることになる．それゆえに，半導体産業界においても「多品種小量となるシステム LSI 開発に少品種大量生産システムでどう乗り切るか」という根本命題に対する解決策の一つとして，この動的な柔軟性をもたらすリコンフィギャラブル・コンピューティングの特質に大きな期待が寄せられている．

このような状況は，ASIC に代表されるカスタム化や多品種小量生産の時代の一つの終焉を示しているのかもしれない．Makimoto's Wave[9]（図 6）に示されているように，標準化とカスタム化は 10 年を周期に選手交代が行われている．現在は，まさにその狭間であろう．これはソフトウェアによるカスタム化を図ったマイクロプロセッサとメモリだけの時代と同様に，処理内容と LSI 構成が再び分離される時期が来ていると言える．まさにその枠組みが「やわらかいハードウェア」を活用したリコンフィギャラブル・コンピューティングではないだろうか．

参考文献

[1] 末吉敏則: Reconfigurable Computing System の現状と課題-Computer Evolution へ向けて-, 信学技報, Vol.96, No.426, pp.111-118 (1996).
[2] 樋口哲也: 進化型ハードウェア, 情報処理学会誌, Vol.40, No.8, pp.795-800(1999).
[3] 末吉敏則: リコンフィギャラブルロジック, 電子情報通信学会誌, Vol.81, No.11, pp.1100-1106 (1998).
[4] 田丸啓吉: やわらかい LSI デバイス：FPGA, 情報処理学会誌, Vol.40, No.8, pp.783-788 (1999).
[5] 末吉, 飯田: リコンフィギャラブル・コンピューティング, 情報処理学会誌, Vol.40, No.8, pp.777-782 (1999).
[6] Thomas C. Waugh: Field programmable gate array key to reconfigurable array outperforming supercomputers, Proc.IEEE CICC, pp.6.6.1-6.6. (1991).
[7] 今井正治: HW/SW コデザインとやわらかいハードウェア, 情報処理学会誌, Vol.40, No.8, pp.789-794 (1999).
[8] 中島, 森, 佐藤, 高橋, 浅見, 水上, 飯田, 新留: FPGA ベース並列マシン RASH, 並列処理シンポジウム JSPP'99, p.222 (1999).
[9] D.Manners and T.Makimoto:Living With The Chip, Chapman & Hall, (1995).