書替え可能なFPGAを用いた汎用エンジンとその応用

沼昌宏，井上真一，澄川文徳，平野浩太郎

神戸大学大学院　自然科学研究科

CADにおける特定の処理を高速化するため，ワイヤード論理を用いた専用エンジン，マイクロプログラム制御を用いたエンジン，汎用の並列計算機の利用が行われてきたが，いずれも柔軟性・汎用性と処理性能間のトレード・オフの問題から逃れることは不可能であった。この問題を解決するために，我々は汎用エンジン（Reconfigurable Machine）の概念を提案し，そのプロトタイプとしてこれまでにRM-I, -II, -IIIを開発した。汎用エンジンでは，高速化の対象とする処理をFPGA上にワイヤード論理で実現するとともに，必要なデータを格納するメモリを用意することで，多様な処理を単体で実行可能としている。論理シミュレーション，論理診断等のCAD処理にとどまらず，画像処理，シストリック・アレイ構造に基づく行列積演算処理にも適用した。その結果，従来の専用エンジンと同等の処理性能と，高い柔軟性の両立が可能であることを確認した。

Conventional speed-up techniques for CAD algorithms using special-purpose engines based on wired-logic, microprogrammed engines, or general purpose parallel machines can not achieve high performance and high flexibility at the same time. This paper presents Reconfigurable Machines: RM-I, -II, and -III, capable of efficiently implementing a wide range of computationally complex algorithms on its flexible architecture combining reconfigurable FPGA's and memories. Their "gate-level programmability" allows us to implement various kinds of algorithms in wired-logic. Reconfigurable Machines have been applied to logic simulation, logic diagnosis, image processing, and systolic algorithm for matrix multiplication. The results show their high performance comparable to special-purpose engines as well as their high flexibility.

1. はじめに

大規模化する論理回路の設計期間を短縮するために，CADにおける特定の処理を高速に実行する専用エンジンが提案・開発されてきた [1]。これらのエンジンは，ソフトウェアによる処理と比較して2桁から3桁の処理速度向上を達成している。しかし反面では，特定の処理向けに徹底した専用化が図られているために処理の柔軟性に欠け，また大きな筐体を必要とするという問題がある [1]。一方，汎用の並列計算機を用いた並列処理によって，高速処理を達成する試み [2] も行われている。この方式では，プログラム言語レベルでの粒度の粗い並列性に頼らざるを得ないため，専用エンジンに比べて処理速度が低下する。

図1に各高速化手法の位置づけを示す。マイクロプログラム制御を導入したエンジン [3] は，ワイヤード論理を用いた専用エンジンと汎用並列計算機の中間に位置付けられるが，その構造上の制約から確保できる柔軟性には限界がある。以上のことから，CADの処理を高速化する場合，柔軟性・汎用性と処理性能間のトレード・オフから逃れることは不可能であった。

この問題を解決するために，我々は電気的に書替え可能なFPGAをメモリと組み合せることによって，複数の用途への適用を可能とする汎用エンジン（Reconfigurable Machine）の概念を提案し，その最初のプロトタイプとしてRM-Iを開発した [4]-[6]。図2にその構成を示す。対象とするアルゴリズムをハードウェア上に実現する実行モジュールと，ホスト・コンピュータとの通信を行うインタフェース・

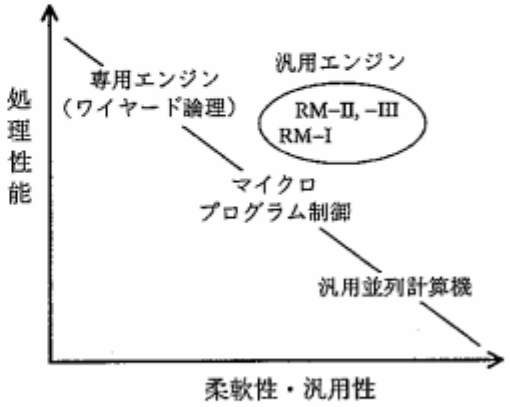


図1　高速化手法の位置づけ

Reconfigurable Machines based on SRAM-type FPGA's and their applications

Masahiro NUMA, Shin'ichi INOUE, Fuminori SUMIKAWA, and Kotaro HIRANO

The Gradurate School of Science and Technology, Kobe University

表 1　RM-I,-II,-III の仕様

比較項目	RM-I	RM-II	RM-III
実現可能な回路規模	20 K ゲート	45 K ゲート	30 K ゲート
総メモリ容量	384 K バイト	768 K バイト	512 K バイト
実行モジュールに搭載する FPGA	XC3090×4	XC4005×9	XC4005×6
メモリバンク数	4	8	8
メモリバンクの構成	32 K×24 ビット	32 K×24 ビット	32 K×16 ビット
FPGA 間の配線	固定配線	配線変更用 FPGA により柔軟性を確保	配線用 FPIC により変更可能
FPGA-メモリ間の配線	完全結合の固定配線，共通バス	変更可能な配線，クロス・バス	配線用 FPIC により変更可能

モジュールから成る。各 FPGA 間は，おもにデータ転送に利用する共通バスと，データ転送と制御の両方に利用可能な完全結合の信号線で結ばれている。

汎用エンジンは，RM-I の例で示されるように，電気的に内部の論理を書替えることができる複数の FPGA（Field Programmable Gate Array）[7] とメモリによって構成される。ゲートレベルでプログラム可能とすることによって，ワイヤード論理を用いたエンジンを柔軟に構築可能とし，高い柔軟性と高い処理性能の両立を実現する。

汎用エンジンで処理を行うには，まずホスト・コンピュータから構成情報（Configuration Data）を送り，FPGA の内部機能を設定する。次に，必要に応じて汎用エンジンのメモリに初期データを設定する。ホスト・コンピュータからの指示によって，汎用エンジンは処理を開始する。結果がメモリに格納される場合は，インタフェース・モジュールを介してホスト・コンピュータに転送する。異なる処理を実行する際には，対応する別の構成情報によって FPGA を再設定する。

論理設計に含まれる誤りを自動的に特定する論理診断手法と論理シミュレーションの 2 種類について，高速に実行するエンジンを RM-I 上に構築した。その結果，従来の専用エンジンに近い処理速度の向上効果が確認された。その一方で，次の問題点も明らかとなった。

- FPGA 間の配線が固定され，完全結合であるため，柔軟性とモジュールの拡張性が低下。
- メモリバンク数の不足によるパイプライン処理における待機時間の発生 [6]。

本稿では，これら問題点の解決を図るべく性能と柔軟性を高めた汎用エンジンのプロトタイプである RM-II と RM-III を中心に，汎用エンジンの構成と応用について述べる。

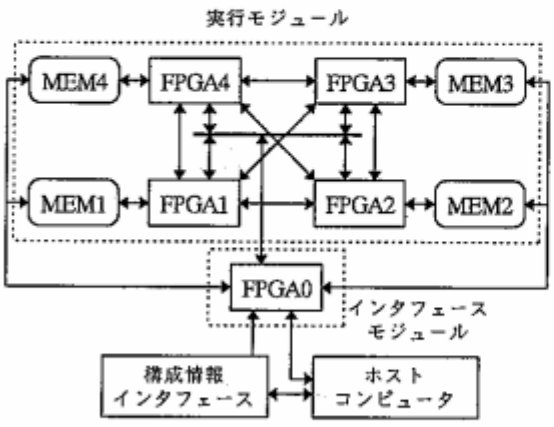


図 2　RM-I の構成

2.　汎用エンジン RM-II

2.1　RM-II の仕様

RM-II の仕様を RM-I との比較を含めて表 1 に示す。RM-II では，実現可能な回路規模，ならびに総メモリ容量を RM-I の約 2 倍とした。バンク数を 8 とすることにより，パイプライン段数に関する制約を緩和する。さらに，FPGA 間の配線を変更するための FPGA を用意することにより，柔軟性を向上させる。XC3090 の上位モデルである XC4005 [8] を FPGA として用いることにより，配線効率および内部資源利用率の向上と，システム・クロックの高速化を図る。

汎用エンジン上では，通常は単一のプロセッサによって処理される内容が，複数の FPGA に割り当てられる。よって，データパスが複数の FPGA に分岐する場合があり，一対多の通信を一対一の通信と同程度の時間内に行うと同時に，通信の遅れ時間が 1 クロック以内であることが望ましい。さらに，XC4005 では外部端子数が 112 と比較的少ないため，多くの外部端子を占有することなく通信を実現する必要がある。

以上の点を考慮して，FPGA 間を複数の縦横のバスで結合するとともに，FPGA とメモリとの接続を流用することによって外部端子の占有を最小限に抑える，クロスバスによる結合を採用した。クロスバ

結合や，ハイパー・クロスバ結合 [9] と類似しているが，クロスバス結合では複数の一対多の通信を，多くの場合1クロックで完了する。

2.2 RM-II の構成

図 3 に RM-II の構成を示す。以下，データバス，アドレスバスについて述べる。

2.2.1 クロスバス結合を用いたデータバス

図 3 における二重線と太線は，それぞれXおよびY方向について各 3 本のデータバス（24 ビット）を示す。FPGA1～8 はそれぞれ対応する MEM1～8 のメモリバンクに接続する。各 FPGA とメモリバンクを接続する信号線に SW1～8 のスイッチを付加することにより，X/Y 方向のデータバスと接続する。FPGA とメモリバンクとの接続信号線を流用することで，FPGA の外部端子を新たに占有することなくデータ転送路を確保できる。さらに，FPGA-FPGA 間の転送に留まらず，FPGA-メモリ，メモリ-メモリ間で転送を行うことも可能となる。なお，FPGA9 は FPGA1～8 の間の配線変更や制御部の実現のために用意され，対応するメモリバンクをもたない。

SW1～8 については，3 分岐の任意の一つが信号源となり，他の 2 分岐の両方または一方に出力できるように，3 組の双方向バッファによって構成される。FPGA とメモリについても，それぞれ双方向の出力を利用する。1 クロック内での転送を実現するため，各スイッチにはレジスタ要素を含まない。

SW9 については，2 組の双方向バッファを用いて FPGA0 を X/Y 方向バスのそれぞれと接続している。X 方向バスと Y 方向バス間でデータを転送することも可能である。

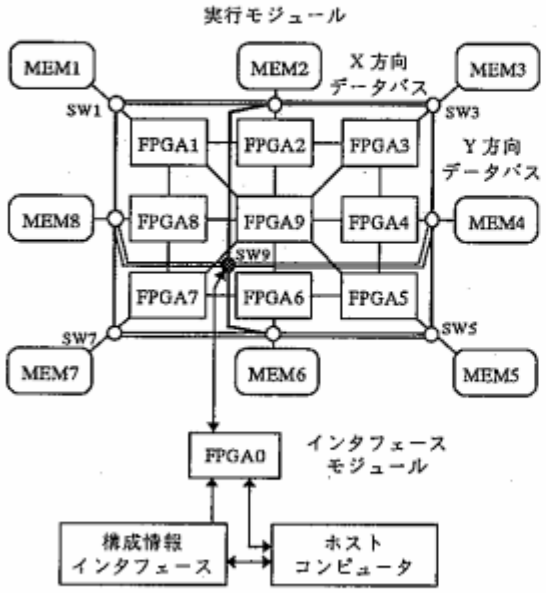


図 3 RM-II の構成

2.2.2 アドレスバス

図 3 では省略しているが，RM-II ではメモリ・アドレスを転送するバスを用意することによって，アドレスを指定して他のメモリバンクを読む操作を 1 クロックで完了する。データバスに比べると利用頻度が低いことを考慮して，X/Y 方向に 1 行／列おきに間引いて配置している。アドレスを送る必要が無い場合には，アドレスバスを第 2 のデータバスとして利用することもできる。

2.3 RM-II の応用

論理シミュレーション，論理設計誤りの診断処理，画像処理の一種である Wavelet 変換を RM-II 上に実現し，性能を評価した。

2.3.1 RM-II の実装と単体性能

RM-II のハードウェアを 1 枚のボード（450×350 mm^2）上にワイヤ・ラッピングで実現した。クロック周波数は，2, 4, 5, 8, 10 MHz から選択可能である。現状では，115 K バイト の構成情報の書込みに 9.5 s を要しているが，I/O タイミングの調整と FPGA への並列書込みによって，0.14 s 程度に短縮可能である。

クロスバスの最大伝搬遅延時間については，10 MHz の信号をアドレスおよびデータバスに与え，5 段のスイッチを通過する時間を測定した。その結果 30 ns 以下となり，比較的高速なデータ転送を行えることが確認された。

2.3.2 論理シミュレータ LSIM-II への応用

RM-I 上に構築した LSIM（Logic SIMulator）[6] と同様のアルゴリズムによる，割当遅延方式のゲートレベル論理シミュレータ LSIM-II を RM-II 上に構築した。LSIM-II の構成を図 4 に示す。LSIM と同様，同一素子を同一時刻に重複して評価することを避けるため，イベント伝搬と素子評価の二つのフェーズに分け，それぞれについてパイプライン処理を行う。

各フェーズの処理について概要を述べる。

(1) イベント伝搬

スケジューラが，各時刻ごとのイベントを管理する TQ (Time mapping Queue) より，現シミュレーション時刻に出力端子イベントを生じる素子の番号と信号値を取り出す。信号値を観測すべき素子にイベントが発生した場合，モニタ・イベントとしてホストに通知する。プロパゲータが，素子番号をもとに接続索引表を参照し，接続本表の先頭アドレスを得る。そのアドレスから接続本表を順次読み出すことで，素子出力から接続する入力端子へのファンアウト処理を行う。イベント・マネージャによってイベントが伝搬する入力端子の信号値を更新するとともに，イベント・フラグによって未登録と判断された場合

表 2　論理シミュレータの処理速度比較

回路名	ゲート数	LSIM	LSIM-II	LSIM-III
DDA8	584	1.11	1.50	1.56
INV1K	1,000	0.97	1.80	1.81
OKI	420	0.97	1.86	1.85
演算器	156	1.30	1.79	1.60

単位：Mイベント／s

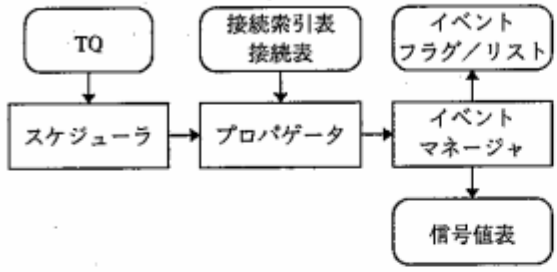
は，評価すべき素子としてイベント・リストに登録する。

(2)　素子評価

イベント・マネージャが，評価すべき素子をイベント・リストから取り出すと同時に，イベント・フラグをクリアする。演算器が，入力信号値と素子機能を示すコードをもとに，この素子の出力信号値を評価する。比較器によって，現在の出力値との比較が行われる。変化が生じた場合は，TQ の現時刻に遅延時間を加えた時刻にイベントとして格納する。

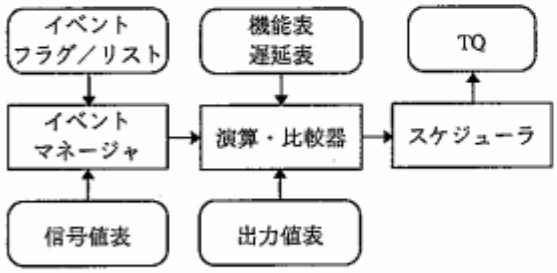
LSIM では，メモリバンク数の不足によってイベント当たり 2 クロックの待機時間が必要であった。RM-II ではメモリバンク数を倍増させたため，LSIM-II ではこの待機時間をなくすことが可能となった。

表 2 に処理速度を示す。LSIM と同様に 4 MHz のクロックを用いた。特に平均ファンアウト数が少ない INV1K，OKI [10] の回路例で処理速度向上の効果が大きい。これらの例では，イベント当たり約 2 クロックで処理を完了する。この点では，専用エンジンによる SP（Simulation Processor）[11] と同等と考えられる。



TQ: Time mapping queue

(a)　イベント伝搬時の構成



(b)　素子評価時の構成

図 4　論理シミュレータ LSIM-II の構成

表 3　論理診断エンジンの性能比較

項目	LDE	LDE'	LDE-II
メモリサイクル／クロック	2	1	1
状態数	16	24	16
処理クロック数の比	1	1.85	0.95
クロック周波数	4 MHz	8 MHz	8 MHz
処理速度比	1	1.08	2.1

2.3.3　論理診断エンジン LDE-II への応用

ゲートレベルの回路に含まれる論理設計誤りを追跡・特定する論理診断の一手法である，拡張X－伝搬法における多値シミュレーション処理を，RM-I 上で高速に行う論理診断エンジン：LDE [4] がすでに開発されている。ここでは，RM-II 上で同様の処理を行う LDE-II について述べる。

表 3 に文献 [4] と同一の 7 回路例に対する実験の結果に基づく性能比較を示す。LDE では 4 MHz のクロックを採用して各クロック当たり 2 回のメモリサイクルを用意することで，read-modify-write を行っていた。LDE-II ではクロック周波数を倍の 8 MHz としたが，遅延の問題で現状ではクロック当たり 1 メモリサイクルとしている。表の LDE' は，RM-I を用いて LDE-II と同様のクロックおよびメモリサイクルを利用した場合の仮想的な性能を示す。LDE' では状態数が 16 から 24 に増加し，処理に要するクロック数が 1.85 倍となるため，処理速度の向上は 1.08 倍にとどまると見積もられる。一方 LDE-II では，バンク数の増加とクロスバスの利用によって状態数を減らすことが可能となり，メモリサイクル数の減少にも関らず 2.1 倍の処理速度を得た。

2.3.4　Wavelet 変換エンジンへの応用

RM-II が論理設計の CAD 以外の分野にも適用可能であることを示すため，画像に対する Wavelet 変換 [12] に応用した。実現した Haar Wavelet 変換は，隣接 4 画素間で加減算を行い，新たな 4 画素の値を得る操作を反復する。RM-II 上の 8 バンクのメモリのうち原画像格納に 4 バンク，変換後の画像格納に 4 バンクを割り当てた。また，1 ピクセルの画像データが 8 ビット で表せることから，各バンクの 1 ワードで 2 画素分のデータを表現した。その結果，256×256 画素の画像に対して 1.64 ms で処理が完了し，11 MIPS の計算機と比べて約 60 倍の速度を得た。

3.　汎用エンジン RM-III

RM-II では，中央部に配線変更用の FPGA によって柔軟性を向上させるとともに，クロスバスによっ

て多くの場合に必要な通信容量を確保することが可能となった。しかし，FPGA 内部の論理変更とは別に，FPGA 間や FPGA–メモリ間の配線そのものを変更することは困難であった。そのため，アプリケーション開発，すなわち汎用エンジン上で実現する論理回路の設計に当たり，対象とする汎用エンジンのアーキテクチャを十分に考慮する必要があった。

そこで，より高い柔軟性を確保するために，電気的に配線変更可能な LSI [13] とそれを搭載したボードを用いて，FPGA 間，および FPGA–メモリ間の配線そのものを変更可能とする，汎用エンジン RM–III を考案した。

以下，RM–III で採用した配線変更用の LSI とボードについて述べた後，RM–III の構成と応用例を示す。

3.1 配線変更用の LSI とボード

電気的に配線変更が可能な LSI として，'92 年 6 月に Aptix 社から FPIC（Field Programmable Interconnect Component）と呼ばれるデバイスが発表された。この FPIC は，32×32 の格子状に並んだ 1,024 本の外部端子をもち，そのうち 936 本についてユーザが相互接続をプログラムできる。図 5 にその内部構成を示す。パス・トランジスタのゲート入力を制御する SRAM の内容により ON/OFF がプログラムされる。入出力バッファが削除されているため，外部端子間は単なる抵抗と同様に，双方向の信号を接続することができる。FPGA を配線変更に用いた場合と比べて，方向制御が不要となり，結果的に遅延時間が短くなる効果が期待できる。出力ドライバなしで駆動するため，ファンアウト数の増加によって遅延が大きくなるが，ファンアウト数が少ない場合は，遅延時間は 10 ns 程度である。

この FPIC を 4 個搭載した，Aptix 社製のボード FPCB（Field Programmable Circuit Board）（233.4×340 mm^2）[13] を利用した。このボードでは，従来のブレッド・ボードにおいて手作業で行っていた配線が，FPIC を利用することで計算機からのデータ書込みによって実現される。3,018 本の着脱可能な端子が用意され，300 mil DIP 型の素子に適合する。一部の 600 mil 対応のための重複端子 78 本を除いた 2,940 本が，異なる信号のために利用できる。

図 6 に，FPCB 上の端子と 4 個の FPIC の接続関係を示す。各 FPIC から，735 本の端子に個別に接続する。FPIC 間はそれぞれ 51 本の信号線で接続されている。このほか，グローバル端子接続用の 40 本によって，すべての FPIC が共通に接続する。このうち 8 本はクロック信号線のために用意されている。

FPIC と同様に配線変更を可能とする LSI として，I–Cube 社の IQ160 [14] もあるが，FPIC に比べて平均的な外部端子間の遅延が大きく（17 ns），またプログラム可能な外部端子数が 160 となり，FPIC の 1/6 程度であるため，FPGA 間の配線変更に用いるためには，FPIC 以上に多段接続する必要がある。よって，RM–III では FPIC と，これを搭載した FPCB を採用した。

3.2 RM–III の特徴と仕様

FPGA によってプロセッサ間の結合網を可変とする例として，可変構造型相互結合網 [15] がすでに提案されている。しかし，汎用エンジンにおける FPGA 間，および FPGA とメモリ間のすべての配線に FPGA を介在させることは，遅延時間の問題と信号の方向制御が必要な点で不適当である。汎用エンジン RM–III では，前節で紹介した配線変更専用の LSI を用いることによって，遅延時間の増加を最小限にくい止めつつ，柔軟性の向上を目指した。RM–I, –II では，高速化の対象とするアルゴリズムについて，それぞれのアーキテクチャに適した実現法を注意深く考える必要があった。一方 RM–III では，以下の 4 点を特徴としている。

(1) 適切な結合網を選択・実現可能

実現するアルゴリズムに適した FPGA 間の相互結合網を選択・実現により，性能向上が期待できる。

(2) メモリ・バンクの容量と接続形態を変更可能

FPIC を介した FPGA–メモリ間接続を採用することで，メモリ・バンクの容量や，FPGA との接続形態

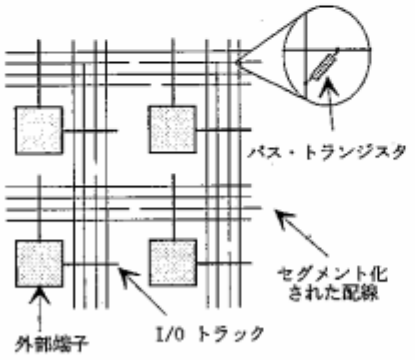


図 5 FPIC の内部構成

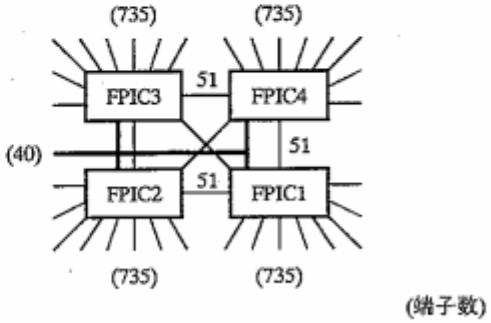


図 6 FPCB 上の端子と FPIC 間の接続

についても柔軟性を確保できる。

(3) 遅延削減

従来は，FPGA の外部端子をすべて固定した上で FPGA 内部の配置・配線を行うことが必要であり，配線の無理な引き回しによる遅延時間の増加が問題となる場合があった。RM-III では，FPGA 間の配線を FPIC で実現するため，外部端子を固定せずに配置・配線を行うことが可能となり，FPGA 内部の配線遅延を削減する効果が期待できる。

さらに，柔軟性の向上によって FPGA 内部資源の利用効率が高まり，FPGA の入出力ブロックを通過する際に生じる遅延も削減する効果が期待される。

(4) アプリケーション開発時間の短縮

(1) から (4) までに述べた項目と関連して，汎用エンジン上での実現に関する制約条件が緩和されるため，アプリケーション開発に要する時間の短縮が期待される。

RM-III の仕様については，RM-I, -II とともに表 1 に示した。FPCB 上で接続変更が可能な端子数の制限により，実行モジュールに含まれる FPGA を 6 個としている。一方で，並列処理，パイプライン処理を容易に実現するために，メモリバンク数については 8 とした。メモリの利用効率向上のために，ワード長を 16 ビットに縮小しているが，FPIC を介した配線変更により，その整数倍に拡張することもできる。

3.3 RM-III の構成

図 7 に RM-III の構成を示す。実行モジュールを構成する 6 個の FPGA について，それぞれ独立した ZIF ソケット付きサブボード上に配置している。このサブボード上にメモリを搭載して互いに配線しておくことも考えられたが，FPIC に基づく配線変更による柔軟性を高めるため，メモリと FPGA は FPIC を介してのみ接続することとした。

各メモリバンクは，32 K×8 ビット構成の 256 K ビット SRAM 2 個と，アルゴリズム実行時にデータバスとアドレスバスをインタフェースから切り離すためのバッファから構成される。メモリと対応するバッファを同一サブボード上に搭載して予め配線しておくことで，実装密度の向上と配線情報の簡略化を実現した。

FPIC の構成情報を設定した後で，FPGA の構成情報を設定する。ユーザはアプリケーションに応じて，これら双方の構成情報を書き換えることができる。

3.4 支援ソフトウェアの環境

RM-III の支援ソフトウェアには，RM-III の配線接続データ作成，FPGA の構成情報設定，RM-III の制御やメモリへの読み書きを行うためのプログラムが含まれる。以下，支援ソフトウェアと開発環境について述べる。

RM-III 上にアプリケーションを構築するための開発環境を図 8 に示す。

(1) 動作レベル・シミュレーション

対象となるアルゴリズムを，C 言語，C++ 言語を用いて記述し，シミュレーションを行う。レジスタ転送レベルのシミュレーションを行えるライブラリを C++ 言語用に開発しており，現在はこれによって動作検証を行っている。一般的なハードウェア記述言語を用いることも考えられる。

(2) 各メモリ，FPGA への割付け

アルゴリズムで必要とする記憶要素，論理ブロッ

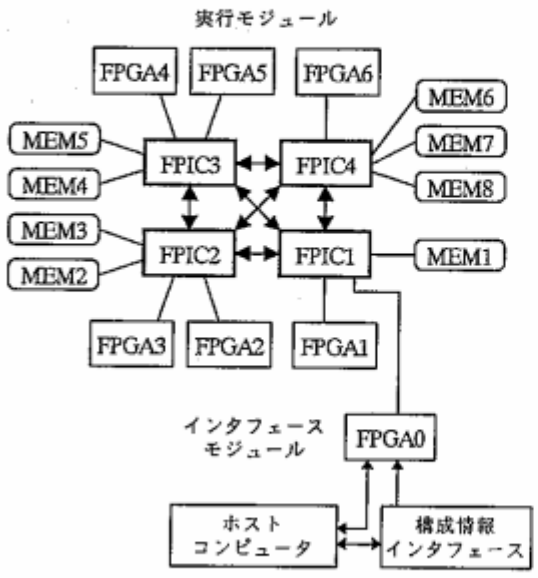


図 7 RM-III の構成

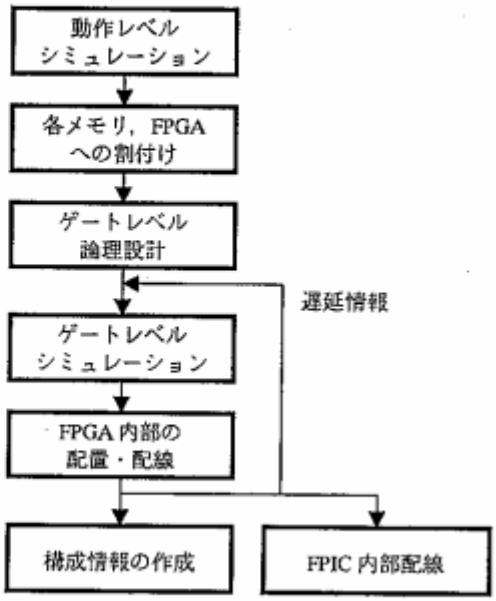


図 8 RM-III のアプリケーション開発

クのそれぞれを割り付けるべきメモリバンクとFPGAを人手で決定する。同時に，FPGA間結線上の転送経路の割付けについても人手で決定する。実際には，(1)と並行して割付けを行う場合が多い。割付けが具体化した段階で，レジスタ転送レベルのシミュレーションによって性能評価を行う。

この工程は，(1) の結果とともに，アプリケーションの性能に大きな影響を及ぼす。一方で，人手による割付けの必要性は設計上の負担となっており，その自動化が課題となっている。

(3) ゲートレベルの論理設計とシミュレーション

各 FPGA に割り付けられた論理をゲートレベルに詳細化する。ゲートレベルのシミュレータを用いて，メモリバンクを含めた RM-III のシミュレーションを行う。

(4) 配置配線

Xilinx 社の配置配線プログラムにより，FPGA の構成情報を生成する。i486DX2-66MHz の PC 上で，各 FPGA に対して平均約 20 分で処理を終える。また，これまでの数十例に対する配置配線について未結線は生じていない。配置配線後，必要であれば，遅延を考慮したシミュレーションを行う。

(5) 構成情報の生成

7 個の FPGA の配線情報をもとに，FPGA 構成情報を生成する。Motorola EXORMACS と呼ばれる ROM データ・フォーマットを利用している。また，FPGA の配線結果から外部端子割り当て情報を抽出し，FPIC の配線接続データを生成する。FPIC の配線についても，これまでの数十例に対する配線で未結線は生じていない。

3.5 RM-III の応用

RM-III では，配線変更のために FPIC を導入することで，柔軟性の向上を図っている。ここでは，RM-I, -II と処理性能，柔軟性を比較するために，論理シミュレーションと行列積演算に RM-III を応用した結果を示す。

3.5.1 論理シミュレータ LSIM-III への応用

RM-II との処理性能比較のために，LSIM-II と同一のアルゴリズムを採用した論理シミュレータ LSIM-III を，RM-III 上に構築した。ワード長を確保するために，8 個のメモリバンクのうち 6 個について，2 個一組で 32 K×32 ビットの 3 バンクを構成した。残りの 2 バンクについては 16 ビット幅のまま利用した。合計 5 個の独立したバンクを用いて待機時間をなくした。その結果，表 2 に示したように，LSIM-II とほぼ同等の処理速度が得られた。わずかな差は，回路分割結果によってラッチを挿入する位置が異なるためである。

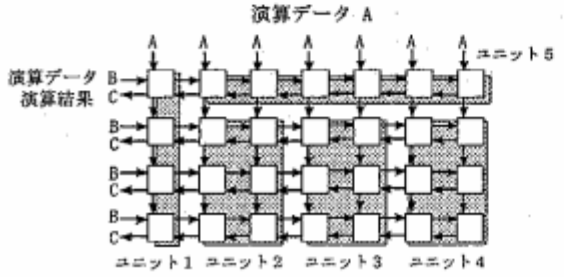


図 9 行列積演算エンジンの構成

3.5.2 行列積演算への応用

配線変更を可能とする FPIC の導入によって，多様な結合網に対応可能となることを確認するために，2 次元シストリック・アレイ構造に基づく行列積演算エンジン MTX（MaTriX multiplier）を RM-III 上に実現した。

Hung によるシストリック・アルゴリズム [16] を利用した。このアルゴリズムの特徴は，わずかな種類の簡単なセルを規則的に配列し，データの送受を隣接したセル間のみで行い，データと制御の流れを単純に規則的にして全体を同期させて演算を実現することである。メモリ読み書き回数を減らすために同一のデータを多数回利用するとともに，演算の中間結果を直接次のセルに渡す。

MTX では，このセルを 2 次元状に配置して，シストリック・アルゴリズムに基づき行列積演算を行う。最大 2,730 組の 4×4 正方行列の乗算を一つの処理単位として実行する。各要素は 7 ビット幅で表され，演算結果は 16 ビット幅で表現される。各セルは，

$$AB = C = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix}$$

を計算するとともに，a_{ij}, b_{ij} をそれぞれ縦方向，横方向に隣のセルに渡す。28 個のセルが 4×7 の 2 次元配列で並ぶ。5 個の FPGA 上にセルを実現し，残り 1 個に制御回路を実現した。

実験の結果，5 MHz のシステム・クロックを用いて，2,730 組の行列積演算を 6.6 ms で完了した。13 MIPS の計算機上のソフトウェアを用いて同じ処理を実現すると，290 ms を要する。すなわち，MTX によって約 44 倍の処理速度が得られた。

この結果から，RM-I, -II では不可能であった 4×7 の 2 次元アレイに基づく結合網を，RM-III では実現できることが確認された。

3.6 内部資源利用率

今回実験した回路に関する，RM-III の内部資源利用効率を表 4 に示す。LSIM-II と-III を比較すると

表 4　内部資源利用率

回路名	FPGA 数	CLB 総数（利用率／%）	IOB 総数（利用率／%）
LSIM-II	9	671（19）	732（73）
LSIM-III	4	412（26）	393（88）
MTX	6	1669（71）	313（47）

CLB の利用率が向上しているが，FPGA 数が 4 個に減っているため CLB 総数は減少している。IOB についても -III のほうが高い利用率を示している。RM-II では端子接続が固定され，利用できない端子が生じたが，RM-III ではそのような制約がないためである。このことから，RM-III では，少ない FPGA 数で RM-II と同規模の回路を実現できると期待される。

MTX では，平均で 71%の CLB 利用率を示しており，内部資源の有効利用が実現された。

4. 今後の課題

これまでに開発した汎用エンジン RM-I, -II, -III に関する結果をふまえ，今後の汎用エンジンに関する課題を述べる。

(1)　ハードウェア

最新の RM-III では FPGA 内部と外部の双方についてプログラム可能としている。汎用エンジンのハードウェアの基本的形態としては，今後もこの RM-III の形態を踏襲することが考えられる。

一方で，実現するアプリケーションよって異なるが，汎用エンジンは現状で 500～600 MIPS 程度の計算能力を示している。性能向上が続くワークステーションや並列計算機に対する優位性を将来に渡って維持するためには，当然ながらより高速に動作する FPGA や配線変更用 LSI の採用が不可欠である。また，ホスト・コンピュータとのインタフェースについて，PCI に代表されるローカルバスを採用することで，データ転送を飛躍的に高速化することも考えられる。さらに，DSP のような既存の LSI を一部の FPGA の代わりに利用することも不可能ではない。

(2)　アプリケーション設計支援

3.4 節で述べたように，アルゴリズム設計とメモリバンクや FPGA への割付けの部分が，性能に与える影響が大きく，今後自動化が望まれている。また，汎用エンジンで実現したときの性能を早い段階で評価することが可能となれば，アプリケーションの移植に関する優先度を決定する上で有益である。

5. まとめ

FPGA とメモリを組み合わせることで，多様な処理の高速化に利用できる汎用エンジンについて述べた。高速化の対象とする処理をワイヤード論理で FPGA 上に実現することにより，高い処理性能と柔軟性の両立を実現した。特に RM-III は，配線変更用の LSI である FPIC の導入によって，FPGA 間および FPGA-メモリ間の結合網に関して高い柔軟性をもつ。

今後は，ホスト・コンピュータとのデータ転送に要する時間の短縮と，設計支援環境の構築等を課題とする。

参考文献

[1] T. Blank, "A survey of hardware accelerators used in computer-aided design," *IEEE Design and Test of Computers*, vol. 1, no. 3, pp. 21–39, 1984.

[2] K. Y. Tham, "Parallel processing for CAD applications," *IEEE Design and Test of Computers*, vol. 4, no. 5, pp. 13–17, 1987.

[3] P. Agrawal et al., "MARS: a multiprocessor-based programmable accelerator," *IEEE Design and Test of Computers*, vol. 4, no. 5, pp. 29–36, 1987.

[4] 菅沼直昭, 村田之広, 富田昌宏, 平野浩太郎, "汎用エンジンの開発と論理診断への応用", DA シンポジウム '92, pp. 89–92, 1992.

[5] N. Suganuma, Y. Murata, S. Nakata, S. Nagata, M. Tomita, and K. Hirano, "Reconfigurable machine and its application to logic diagnosis," *International Conf. on Computer Aided Design*, pp. 373–376, 1992.

[6] 澄川文徳, 永田真一, 菅沼直昭, 富田昌宏, 平野浩太郎, "RM-I による論理シミュレーション", 情報処理学会第 45 回全国大会講演論文集, vol. 6, pp. 159–160, 1992.

[7] プログラマブル・ゲートアレイ　データブック, ザイリンクス社, 1990.

[8] *The XC4000 Data Book*, Xilinx, Inc., 1991.

[9] 中越順二, 田中輝雄, 濱中直樹, 面田耕一郎, "並列計算機 H2P の要素プロセッサ間非同期データ転送方式", 情報処理学会第 38 回全国大会講演論文集, pp. 1488–1489, 1989.

[10] 菊池原秀行, 今村真人, 青柳洋介, 浜崎良二, 白木昇, "HLS: 論理シミュレーション専用計算機 (6) 性能評価", 情報処理学会第 41 回全国大会講演論文集 (6), pp. 137–138, 1990.

[11] M. Saitoh, K. Iwata, A. Nakamura, M. Kakegawa, J. Masuda, H. Hamamura, F. Hirose, and N. Kawato, "Logic simulation system using Simulation Processor (SP)," *Proc. 25th Design Automation Conference*, pp. 225–230, 1988.

[12] O. Rioul and M. Vetterli, "Wavelet and signal processing," *IEEE SP.*, vol. 8, no. 4, pp. 14–38, 1991.

[13] *Programmable Interconnect Data Book*, Aptix corp., 1993.

[14] *IQ160 – Field Programmable Interconnect Device (FPID)*, I-Cube Design Systems, Inc., 1993.

[15] 末吉敏則, 杷野公平, 有田五次郎, "書換え可能な LSI による可変構造型相互結合網の実現法", 情報処理学会論文誌, vol. 33, no. 3, pp. 260–269, 1992.

[16] H. T. Kung, "Why systolic architecture ?," *IEEE Computer*, vol. 15, no. 1, pp. 37–46, 1982.