計算機アーキテクチャ　152－20
（２００３．３．１１）

再構成可能プロセッサ486RCPによる再構成可能ハイパープロセッサの予備実験

下尾浩正† 岩根雅彦†

プロセッサが得意としない計算集中型の処理部分だけでなく，他の一般的な処理までも含めてハードウエア化し，それらを動的に再構成する動的再構成可能プロセッサDRHPを提案する．DRHPは，特徴を持った制御部とその制御部が解釈する命令列（コマンドプロシージャ）によって，再構成時間を隠蔽し，中粗粒度での並列実行やパイプライン実行などの様々な実行方式を実現する．また，より小さい文レベル以下の粒度では，並列プログラミング技法やハードウエアの並列性により並列化した処理をハードウエア化して実行し，更なる性能向上を図る．最大公約数を求めるプログラムに並列化を施した場合，1.70倍の性能向上が見られた．また，予備実験としてコマンドプロシージャを用いたDHRPの動作をシミュレーションによって評価したたところ，AES暗号化処理では15.5倍の性能向上が得られた．

Preliminary experiments for Reconfigurable Hyper Processor using 486RCP

Kosei Shimo † and Masahiko Iwane†

The Dynamic Reconfigurable Hyper Processor (DRHP) executes all portions in an application program that consist of not only calclating parts unsuitable to CPU but also other general processing parts. The control logic and Command Procedure can hide a reconfiguration time and provides several execution methods such as parallel and pipeline execution on the DRHP at the midium grain parallelism. Moreover, at the finer grain parallelism, a parallel processing method and hardware parallelism are utilized to improve performance more. For utilizing the finer grain parallelism, the result shows that DRHP archives the speedup of 1.70 times. A preliminary experiment of execution method using command procedure shows the speedup of 15.5 times on AES encryption processing.

1. はじめに

アプリケーションのハードウエア化では，PCIインターフェイスに接続されたFPGA（Field Programmable Gate Array）を用いたハードウエアアルゴリズムの研究が多い．また，再構成可能コプロセッサに従来のプロセッサが得意としない多倍長や可変長のビット幅を持つ演算などを実行させるコンピュータ構成の研究[1)2)]，さらに，このような構成上におけるハードウエアとソフトウエアのコデザインに関した研究が行われている．しかし，そのような計算集中部分だけでなく，その他の一般的な処理を含めたプログラム全体をハードウエア化して実行することによって，より一層の速度向上が見込まれる．

そこで，プログラム全体のハードウエア化を考えて，サブルーチン（関数）レベルの中粒度での並列処理やパイプライン処理などの多様な処理を可能とし，さらにそれらの処理を動的に再構成することによってローディング時間を隠蔽する動的再構成可能ハイパープロセッサ（Dynamic Reconfigurable Hyper Processor:DRHP）を提案する．

また，現代のプロセッサはプログラムをより高速に実行するため高度なパイプライン処理を行ったり，十分な数のレジスタを実装するといった様々な工夫が行われているが，パイプラインのフラッシュによってプログラムに内在している時間的並列性や演算器およびレジスタなどの制限によって空間的並列性を十分に引き出していない．そこで，ハードウエアの並列性および並列プログラミング技法による文レベルの並列化によって，プログラムをハードウエア化して実行し，更なる速度向上を図る．

本論文は，動的再構成可能プロセッサDRHPの概要について述べ，中粒度での並列処理などを可能とする制御機構とプログラミング手順について説明する．次に，文レベル以下の並列化に関して，文レベルの並列化手法およびハードウエアの並列性におけるハードウエアの設計手法を示す．その後，評価プログラムを

† 九州工業大学 工学部 電気工学科
Department of Electrical Engineering, Faculty of Engineering, Kyushu Institue of Technology

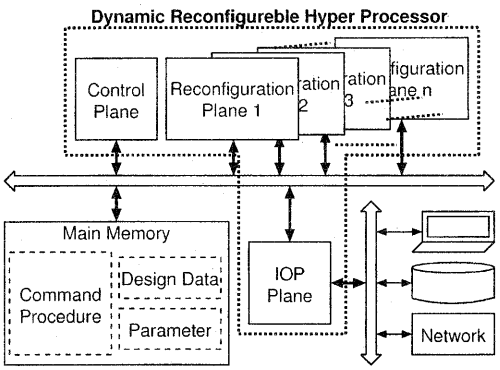

図 1　DRHP の構成
Fig. 1　Organization of DRHP

用いて，再構成可能プロセッサ 486RCP[3] で予備実験を行った結果を示し，それについて考察を述べる．

2. 動的再構成可能ハイパープロセッサ DRHP

2.1 DRHP の概要

図 1 に DRHP の構成を示す．DRHP は，プロセッサ全体を制御・管理する制御プレーン（Control Plane），設計データをローディング・演算処理を実行する n 個の再構成プレーン（Reconfiguration Plane），外部との入出力を制御・処理する IOP プレーン，および，メインメモリから成る．ユーザは，対象とするアプリケーションに対し，DRHP が実行するコマンドを並べたコマンドプロシージャを作成し，メインメモリに格納する．また，対象とするアプリケーションを意味のあるまとまり（ファンクション）に分割，ハードウエア化して，設計データを生成する．この設計データとファンクションの実行に必要なパラメータもメインメモリに格納する．制御プレーンは，メインメモリからコマンドプロシージャを読み出し，各再構成プレーンの状態を考慮して，コマンドを各再構成プレーンに発行する．コマンドを受け取った再構成プレーンは，設計データのローディング，あるいはファンクションの実行を行う．

2.2 制御プレーン

制御プレーンは，**図 2** の様におもにコマンドプロシージャを読み出し，再構成プレーンにコマンドを発行するコマンド発行部，再構成プレーン状態を把握・管理するためのプレーン管理 CAM，再構成プレーン間の情報の受け渡しに使用するバッファであるコミュニケーションボックスなどから成る．

コマンド発行部は，メインメモリに格納されているコマンドプロシージャを読み出し，各再構成プレーンの状態をプレーン管理 CAM によって確認し，適切にコマンドを発行する．プレーン管理 CAM は，再構成プレーンの情報を持ち，再構成プレーンの数（n）のエントリを持つ．プレーン管理 CAM の各ビットについて示す．プレーンには，物理プレーン番号とコマンドプロシージャ作成の際に使用する論理プレーン番号があり，これを示すフィールドが PP#と LP#である．再構成プレーンの状態は，プレーン内容が空であることを表す V，ファンクションの実行中を表す X，設計データのローディング中を表す L から成る．プレーン間の情報は，複数のプレーンに渡ってファンクションが実装されているかを表す構成ビット M，プレーン間の実行順序を表す P，および，プレーン間の情報転送の有無を表す F/E ビットから成る．制御プレーンは，ローディングや実行の状態を再構成プレーンのステータスレジスタから得る．また，ファンクションが複数のプレーンに渡って実装されていることおよびその実行順序を表す情報はあらかじめ設計データのヘッダに埋め込む．

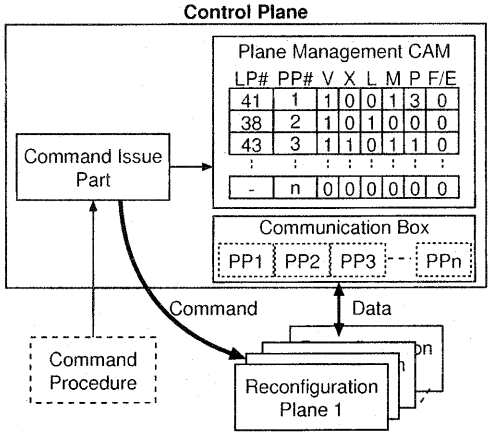

図 2　制御プレーンの概要
Fig. 2　Concept of Control Plane

2.3 コマンドプロシージャ

2.3.1 コマンドの仕様

コマンドプロシージャを利用して，適切に設計データのローディングおよびファンクションの実行を制御することによって様々な実行方式を実現することができる．DRHP で使用するコマンドのフォーマットを**図 3** に示す．コマンドのフォーマットは，W ビット，論理プレーン番号，ファンクション番号，および，アドレス部から成る．Ldplane は，再構成プレーンに設計データのローディングを指示するコマンドである．W ビットは，W=0 のときコマンドの実行終了まで

```
Ldplane W, Logical Plane#,                    Address
Explane W, Logical Plane#, Function#,  Address
Jmp     Logical Plane#, Function#,     Address
```

図 3　コマンドフォーマット
Fig. 3　Command format

次のコマンドの実行を待ち，W=1 のとき終了を待たずに次のコマンドを実行する．ローディングする設計データの論理プレーン番号を指定し，アドレス部は設計データの先頭アドレスを指定する．Explane は，再構成プレーンにファンクションの実行を指示するコマンドである．W ビットは Ldplane と同様であり，実行する論理プレーン番号，実行するファンクション番号，および，ファンクションに使用するパラメータの先頭アドレスを指定する．Jmp は，条件が成立したら指定したアドレスに分岐するコマンドである．分岐条件に使用するステータスレジスタを格納している論理プレーン番号およびファンクション番号を指定し，アドレスは分岐条件や分岐先アドレスを指定する．ユーザは，以上の 3 種類のコマンドを用いて，実行するアプリケーションに対し，設計データのローディングおよびファンクションの実行をプログラミングしたコマンドプロシージャを作成する．

2.3.2　コマンドプロシージャの動作

図 4 は，4 個の再構成プレーンで処理を並列に実行し，それぞれの結果をまとめる処理の実行を一定の条件で繰り返す場合のコマンドプロシージャとタイミングチャートである．まず，S_1〜S_4 の Ldplane コマンドで 4 個の再構成プレーンに設計データをローディングする．W=1 にすることによって，制御部はコマンドの終了を待たずに次のコマンドを発行するため，4 個の再構成プレーンに対し，並列に設計データをローディングすることができる．つぎに，S_5 で再構成プレーン 1 は処理を実行する．制御部はプレーン管理 CAM の Ld ビットを調べ，ローディングが終了したら Explane コマンドを発行する．S_6 に再構成プレーン 1 への Ldplane コマンドを挿入することによって，処理の実行終了後に直ちに次の設計データのローディングを行う．ここで，S_6 で W=1 にすることによって，S_7, S_8, S_9 の Explane コマンドを発行し，4 個の再構成プレーンで処理を並列に実行する．S_9 の Explane コマンドで W=0 にすることによってコマンドの終了を待って，再構成プレーン 1 が再構成プレーン 1〜4 の結果をまとめる処理を実行する．S_{11} で再構成プレーン 1 の実行結果によって条件が成立しないならば，S_1 に分岐し条件が成立するまで処理を繰り返す．

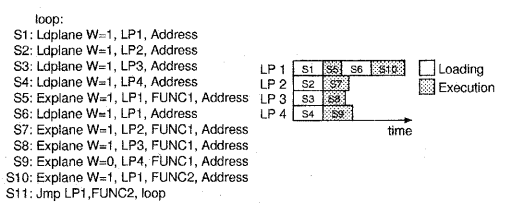

図 4　コマンドプロシージャ
Fig. 4　Command Procedure

3.　ハードウエアの設計手法

3.1　基本ブロック間の制御依存

図 5 に最大公約数を求めるプログラム（gcd）を示し，ハードウエアにおけるシリアルプログラム性能向上のためのハードウエア設計手法について述べる．

まず，プログラムを基本ブロックに分割する．図 5 の例では，4 つの基本ブロックに分割される．基本ブロック間の制御依存は，FSM（Finite State Machine）を用いて，イベント発生時に次状態（次の基本ブロック）に遷移することよって解消する．gcd では，S_{12} の $g2 = 0$ のイベントが発生するまで基本ブロック 2 を実行し，イベントが発生したら，状態遷移することによって次の基本ブロック 3 を実行する．

3.2　文レベルの並列化

基本ブロック内の並列化は，依存グラフを基に行い，基本的に最大並列度での実行を考える．図 5 の gcd において，基本ブロック 1 は初期値を代入する文であり初期値を持つレジスタを 4 つ実装することによっ

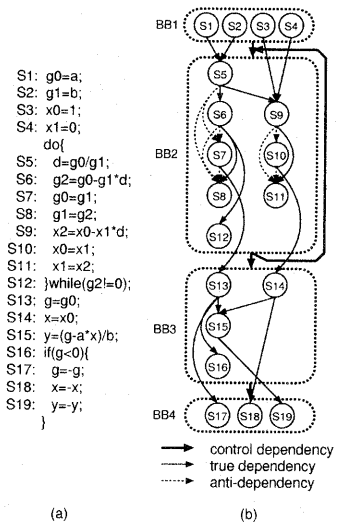

図 5　2 つの整数の最大公約数を求めるプログラム
Fig. 5　Program for gcd

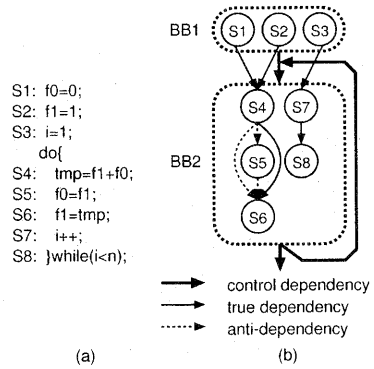

図 6 フィボナッチ数列の第 n 項を求めるプログラム
Fig. 6 Program for Fibonacci Numbers

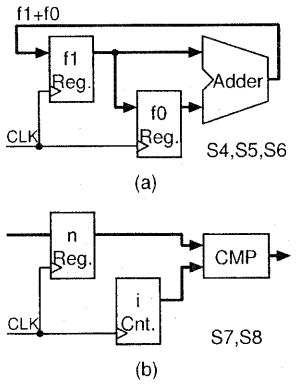

図 7 ハードウエアの並列性
Fig. 7 Parallel of hardware

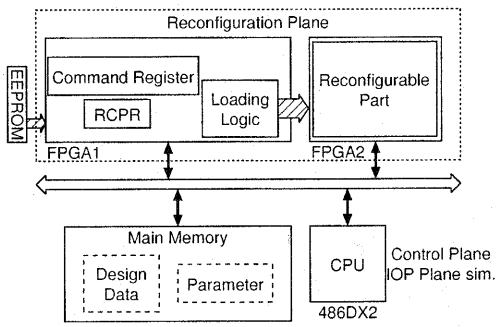

図 8 再構成可能プロセッサ 486RCP
Fig. 8 Reconfigurable Processor 486RCP

て，並列に実行する．基本ブロック 2 は，依存関係が無い S_6, S_7, S_8 と S_9, S_{10}, S_{11} を並列に実行する．S_6, S_7, S_8 と S_9, S_{10}, S_{11} は，変数が異なるだけで演算内容は同じであるため，同じモジュールを 2 個実装する．基本ブロック 3 は，S_{13} と S_{14} の変数代入を並列に実行する．基本ブロック 4 は，依存関係が無い S_{17}, S_{18}, S_{19} の符号変換を行うモジュールを 3 個実装し，並列に実行する．

3.3 ハードウエアの並列性

ハードウエアの特長を生かした並列性について，**図 6** (a) のフィボナッチ数列の第 n 項を求めるプログラムを用いて示す．基本ブロック 2 において，S_4, S_5, S_6 は f1+f0 の演算と 3 つの変数を更新する式である．通常，逆依存があるため順番に実行する必要があるが，クロックの立ち上がりで f1 と f0 の和を f1 Reg. に，f1 を f0 Reg. に同時に書き込み，次のクロックの立ち上がりまでに f1 と f0 の和を計算することによって，S_4, S_5, S_6 を並列に実行する（**図 7** (a)）．また，S_7, S_8 は変数 i をインクリメント後，変数 n との比較を行う．インクリメントにはカウンタを用い，比較はコンパレータを用い，これらは組み合わせ回路で実現できるため，S_7, S_8 を並列に実行する（図 7 (b)）．さらに，依存グラフより S_4, S_5, S_6 と S_7, S_8 を並列に実行できる．したがって，基本ブロック 2 はハードウエアの並列性を用いることによって，S_4〜S_8 を並列に実行する．

4. 予備実験

4.1 予備実験機の概要

DRHP における 1 個の再構成プレーンの予備実験機として，**図 8** に示す再構成可能プロセッサ 486RCP を開発した．486RCP は，DRHP における制御プレーンおよび IOP プレーンをシミュレートするための CPU，1 個の再構成プレーン，および，メインメモリから成っており，再構成プレーンは，Command Register，Loading Logic，ステータスレジスタ（RCPR），および，再構成部から構成する．再構成プレーンは，1 個目の FPGA に Command Register，Loading Logic，RCPR，および，バスアービタを実装しており，電源投入時に EEPROM からローディングされる．2 個目の FPGA は再構成部とし，Loading Logic から設計データをローディングする．システムクロックは，16MHz である．

4.2 並列化の効果

ハードウエア開発手法に基づいて，先に示した図 5 の 2 つの数の最大公約数を求めるプログラムをハードウエア化して実行し，シリアルプログラムを 486DX2 で実行した場合との性能評価を行った．また，シリアルプログラムを直接ハードウエア化して実行する場合およびマルチプロセッサ MTA/TCSMII[4] を用

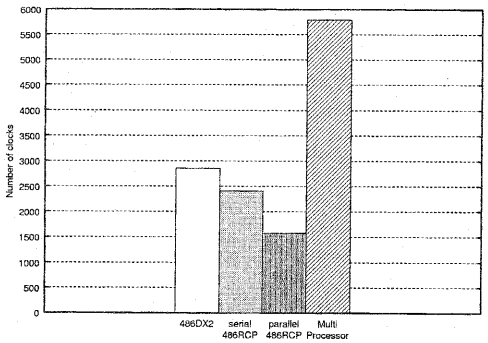

図 9 最大公約数の実験結果
Fig. 9 Result of gcd

表 1 フィボナッチ数列の実験結果
Table 1 Result of Fibonacci Numbers

Fucntion	486DX2	486RCP	Speedup
Fibonaccci	298	63	4.73

いて並列実行した場合についても性能評価を行った．MTA/TCSMIIは，8台の486DX2プロセッサを搭載しており，L1キャッシュは486DX2内に持っている．また，8台の486DX2，L2キャッシュおよびメモリは共有バス結合されている．但し，マルチプロセッサはシリアルプログラムをリストスケジューリングにより並列度2で並列化を行い，依存関係による先行制約はメモリを用いたフラグによる条件同期を用いた．**図9**に486DX2との速度向上比を示す．但し，486DX2は，データおよびコード共にキャッシュに入った状態，L1キャッシュはライトスルーで評価を行った．最大公約数は，a=92736，b=57314とし，ループ文の繰り返し回数は22回である．

また，ハードウエアの並列性について，先に示した図6のフィボナッチ数列の第n項を求めるプログラムをハードウエア化して実行し，486DX2との評価を行った．求めるフィボナッチ数は，32ビットで求めることのできる最大の数である第49項とした．実験結果を表1に示す．表1中の数字はクロック数を表す．

4.3 コマンドプロシージャのシミュレーション

DRHPにおける1個の再構成プレーンの予備実験として，簡単なコマンドプロシージャのシミュレーションを486RCPを用いて行った．**図10**に，表の行方向の和および列方向の和を求める表計算を実行するコマンドプロシージャのシミュレーションプログラムを示す．表における列方向の和は配列要素の和を求めるファンクション，行方向の和は和のベクトル演算を行うファンクションを使用し，列方向の演算，行方向の

```
LDRCP(Design Data);
while(RRCPR&0x03 != 0);
EXERCP(Parameter1);
while(RRCPR&0x05 != 0);
EXERCP(Parameter2);
```

図 10 表計算を行うシミュレーションプログラム
Fig. 10 Program for Spreadsheet

```
AES_Cipher(State, C_Key){

 KeyExpansion(C_Key, E_Key);

 AddRoundKey(State, E_Key);
 for(i=1; i<10; i++){

  ByteSub_ShiftRow(State);

  MixColumn(State);
  AddRoundKey(State, E_Key);
 }

 ByteSun_ShiftRow(State);

 AddRoundKey(State, E_Key);
}
```

(a) AES program

```
LDRCP(Design Data1);
EXERCP(Parameter1);
LDRCP(Design Data2);
EXERCP(Parameter2);
for(i=0;i<10;i++){
 LDRCP(Design Data3);
 EXERCP(Parameter3);
 LDRCP(Design Data2);
 EXERCP(Parameter4);
 EXERCP(Parameter2);
}
LDRCP(Design Data3);
EXERCP(Parameter3);
LDRCP(Design Data2);
EXERCP(Parameter2);
```

(b) simulation program

図 11 AES 暗号化処理
Fig. 11 AES encryption processing

表 2 表計算およびAES暗号化処理の実験結果
Table 2 Result of Spreadsheet and AES encryption processing

Program	486DX2	486RCP	Speedup
Spreadsheet	220	186	1.18
AES	12,117	782	15.5

演算の順に処理を行う．シミュレーションプログラムでは，Ldplaneコマンドおよび Explaneコマンドの代わりに486RCPの専用命令を使用した．すなわち，設計データのローディングを指示するLDRCP命令およびファンクションの実行を指示するEXERCP命令を使用し，コマンドの終了の確認は再構成プレーンのステータスレジスタを読み出すRRCPR命令を使用した．

つぎに，AES（Advanced Encryption Standard）暗号化処理を用いて，コマンドプロシージャのシミュレーションを行った．**図11**（a）にAES暗号化処理のプログラムを，(b) に対応するコマンドプロシージャのシミュレーションプログラムを示す．ただし，シミュレーションプログラムはコマンドの終了を確認する部分を除いている．図11（b）のシミュレーションプログラムは，設計データをローディングするLDRCP命令とファンクションを実行するEXERCP命令を交互に実行している．これは，実験で用いたFPGAのリソースの制限から設計した4個のファンクションに

対し，3 個の設計データを生成したためである．また，Jmp コマンドによる繰り返し制御には for 文を用いた．

このコマンドプロシージャのシミュレーションにおいて，486DX2 との性能評価を行った結果を**図 2** に示す．ただし，AES 暗号化処理に関して 486RCP は全体の実行時間からローディングにかかる時間を差し引いたクロック数，すなわちファンクション実行のクロック数である．

5. 考　　察

図 9 の最大公約数の実験結果よりシリアルプログラムを直接ハードウエア化したファンクションでは 1.18 倍の性能向上であった．これは，プロセッサはパイプライン処理を行うため常に命令の実行を行うので，プログラムを直接ハードウエア化するだけでは性能向上が得られないことを示している．これに対し，ハードウエア設計手法を用いてプログラムをハードウエア化して並列に実行した場合では，1.70 倍の性能向上が得られた．ここで，シリアルプログラムをリストスケジューリングによって並列度 2 で並列化したときのガントチャートより求めた理想的な速度向上比は 1.41 倍である．これに対し，マルチプロセッサを用いた実行では，共有変数の通信によって生じるバスネックによって理論的な速度向上が得られず性能が低下している．ハードウエア化して並列に実行した最大公約数では，文レベルの並列化によって性能向上が得られた．

また，ハードウエアの並列性に関して，表 1 のフィボナッチ数列の実験結果より，4.73 倍の性能向上が得られた．これは，図 7 に示したプログラムの S_4, S_5, S_6, S_7, S_8 を並列に実行したためであり，ハードウエアの並列性によって性能向上が得られた．

したがって，一般的な処理をハードウエア化する場合，プログラムを直接ハードウエア化するのではなく，並列プログラミング技法によるハードウエア設計手法を用いてハードウエア化して実行することによって，性能向上を図る．

また，コマンドプロシージャのシミュレーションにおいて，AES 暗号化処理では 15.5 倍の性能向上であり，ファンクションの連続実行による効果が確認できた．

6. む　す　び

処理の全てをハードウエアで実行する動的再構成可能プロセッサ DRHP を提案した．制御部とコマンドプロシージャによって，粗粒度および中粒度での並列実行やパイプライン実行などの様々な実行方式が実現できる．また，複数の再構成プレーンの内容を動的に変更することによってローディング時間を隠蔽する．文レベル以下の並列化に関して，ハードウエアの設計手法を示し，最大公約数を求めるプログラムをハードウエア化して実行し性能評価を行ったところ，1.70 倍の性能向上が得られた．

今後は，様々なプログラムについて評価を行い，ハードウエア設計手法について更に検討する．

参　考　文　献

1) Hauser, J. R. and Wawrzynek, J.: Garp: A MIPS Processor with a Reconfigurable Coprocessor, in Pocek,K.L. and Arnold,J. eds., *IEEE Symposium on FPGAs for Custom Computing Machines*, pp. 12–21, Los Alamitos, CA (1997), IEEE Computer Society Press.
2) Singh, H., Lee, M.-H., Lu, G., Kurdahi, F. J., Bagherzadeh, N. and Filho, E. M. C.: MorphoSys: An Integrated Reconfigurable System for Data-Parallel and Computation-Intensive Applications, *IEEE Trans. on Computers*, Vol. 49, No. 5, pp. 465–481 (2000).
3) 安河内真弓, 下尾浩正, 山脇彰, 岩根雅彦：再構成コンピューティングシステムの開発, 情報処理学会研究報告 2002-ARC-147, Vol. 2002, No. 22, pp. 91–96 (2002).
4) 山脇彰, 岩根雅彦: 同期通信用メモリにおけるカウンタとブロッキングの効果, 信学論, Vol. J84-D-I, No. 9, pp. 1457–1460 (2001).
5) 岩根雅彦：再構成可能ハイパーコンピューティングシステムの開発, 設計メモ（内部資料）(2002).