社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2005－SLDM－118（25）
2005／1／26

動的再構成デバイスPCA上での自己複製型アプリケーション設計容易化手法の提案と実装

神山 知己　倉田 圭吾　池畑 陽介　北道 淳司　黒田 研一

会津大学大学院コンピュータ理工学研究科 〒965-8580 福島県会津若松市一箕町鶴賀
E-mail: {m5081211,d8061203,m5081208,kitamiti,kuroken} @u-aizu.ac.jp

あらまし 自律再構成可能デバイス Plastic Cell Architecture(PCA)における自己複製型アプリケーションのための設計容易化手法を提案する．自己複製型アプリケーションは，再帰的に定義された手続きにより自己を複製することで所望の処理を実現するものである．本論文で提案する設計容易化手法は，複製および削除などの自己複製型アプリケーションを設計する際に必要となる典型的な機能をフレームワーク化することで，回路設計者のアプリケーション設計時の負担を軽減するものである．本論文ではこのような自己複製型アプリケーションの設計を容易化する手法を提案し，エラトステネスのふるい，セレクションソート，多倍長乗算器，高速アダマール変換に適用することで本手法の評価を行った結果，設計の容易化が行われたことを確認した．

キーワード 動的再構成可能デバイス　PCA　アプリケーションフレームワーク

Proposal and Implementation of Framework for Self-Reproductive Applications on Dynamically Reconfigurable Device PCA

Tomoki KAMIYAMA, Keigo KURATA, Yousuke IKEHATA,

Junji KITAMICHI and Kenichi KURODA

The University of Aizu, Graduate school of Computer Science and Engineering

Turuga, Ikki-machi, Aizuwakamatu-City, Fukusima, 965-8580 Japan
E-mail: {m5081211,d8061203,m5081208,kitamiti,kuroken} @u-aizu.ac.jp

Abstract This paper proposes a design method of self-reproductive applications on an autonomous reconfigurable device, Plastic Cell Architecture (PCA). Self-reproductive applications realize distributed processing by duplication of a basic processing unit using our proposed method. We clarify typical functions in duplicate/delete processes on self-reproductive applications, and establish an application framework that makes design easy by separating core application processes from reproductive functions. Designers don't need to implement self-reproduction control circuits by themselves. We applied this method to the Eratosthenes' sieve, the Selection sort, the multiple precision multiplier and the fast Hadamard transformation circuits, and confirmed effectiveness of our method.

Keyword Autonomous reconfigurable device, PCA, application framework

1. はじめに

近年，ハードウェア資源の有効利用や多様なアプリケーションへの応用を目指して動的再構成可能デバイスが提案されている[1]-[5]．動的再構成可能デバイスの一つとして，Plastic Cell Architecture (PCA) が NTT より提案されている[2][3]．PCA は，他の動的再構成可能デバイスに比べて再構成粒度が細かく，部分的に再構成を行うことができる．さらに PCA 内部からの命令による再構成も可能であり，これを自律再構成と呼ぶ．この機能を用いることで，自身や周辺回路の負荷の状況に応じた適応型負荷分散処理が可能となる[6][7]．

本論文では，条件に応じて自己を複製し，処理を行う分散処理モデルを考える．これを自己複製型アプリケーションと呼ぶ．自己複製型アプリケーションでは，複製および削除を考慮した処理の設計を行わなければならないため，回路設計が困難である．そのため本論文では，自己複製型アプリケーションの設計を容易にする設計容易化手法を提案する．

以下，第 2 章では，PCA について説明する．第 3 章

では自己複製型アプリケーションとその問題点について述べる．第4章では，提案する設計容易化手法について述べ，第5章で，その実装について述べる．第6章では提案する設計容易化手法を具体的なアプリケーションに適用することで評価を行い，第7章でまとめとする．

2. PCA

2.1. PCAの構成

PCAは図1に示すようにPCA Cellと呼ばれる均一なセルを2次元メッシュ状に配置した構造を持つ．PCA Cellは，機能回路およびメモリを構成するPlastic Part (PP) と機能回路間の通信やPPへの回路構成を行うBuilt-in Part (BP) からなる．PPおよびBPは，隣接するPCA Cell同士で相互に通信することができる．PP上に構成した機能回路およびメモリをオブジェクトと呼ぶ．

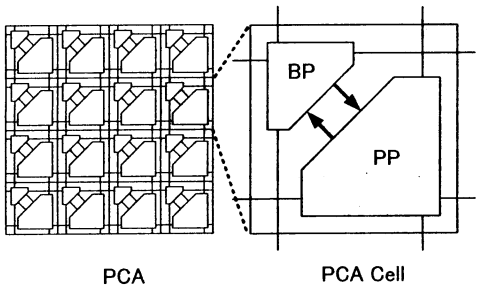

図 1　PCAの基本構成

PCAの実現の一つとしてPCA-1が試作されている[2]．PCA-1は，4サイクル・ハンドシェイクによる非同期通信を採用しており，5bitを1ワードとするデータ通信を行う．PCA-1には，表1に示す12の命令が実装されており，PCAチップの外部からだけではなく，PPに構成されているオブジェクトによって生成することも可能である．

表1　BPにおける命令リスト

命令名	ビットコード	作用
CLEAR	0xxxx	設定経路の削除
PP_OPEN	1000x	PPとの接続を開く
PP_CLOSE	1001x	PPとの接続を閉じる
CO	10100	構成情報の読出し
COCI	10101	PPの複写
CI/M	10110	メモリ情報の書込み
CI/F	10111	回路情報の書込み
WEST	11000	経路を西へ設定
NORTH	11001	経路を北へ設定
EAST	11010	経路を東へ設定
SOUTH	11011	経路を南へ設定
PP_OUT	111xx	PPへ経路設定

3. 自己複製型アプリケーション

本論文で対象とする自己複製型アプリケーションは，状況に応じて自己を動的に複製し処理を行う．この動作をループ処理の例を挙げて説明する．このとき，必要とされる処理に加えて複製および削除の機能も含めた単位をステージと呼ぶことにする．

ループ処理は，図2(a)のように一つの処理を繰り返し実行することで逐次的に処理を行う．これに対して，自己複製型アプリケーションでは，図2(b)のように必要なループ回数だけステージを複製し，並列に処理を行うことで処理を高速化する．処理を終了し必要がなくなったステージは，順次削除することでリソースを解放することができる．このように自己複製を行うことで，再帰的な処理を実現することが可能となり，入力データに応じて再帰の深さが変化する場合などにも柔軟に対応できる．

自己複製型アプリケーションは，自身の回路を複製するため，最初に構成するステージ以外の回路情報を必要としない．また，複製・削除機構を各ステージが内包するため，複製または削除の制御をアプリケーションの外部から行わなくてもよい．さらに，複製回数が見積もれない場合でも，各ステージにおいて条件を判定する機能を実現する機能を定義することで柔軟に対応することが可能になる．

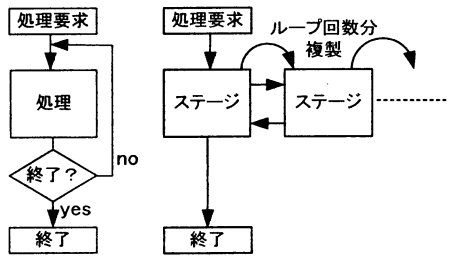

図 2　ループ処理

しかし，自己複製型アプリケーションを設計するには，所望の処理を行う本体部分のほかに，複製および削除の動作や，条件判定を考慮する必要がある．これらをアプリケーション毎に設計するのは非効率であり設計困難となる．本論文では，これらの動作を処理本体から分離し，複製削除の手続きを規則化することで，PCA-1上への自己複製型アプリケーションの実装を容易にする設計手法を提案する．

4. 設計容易化手法の提案

自己複製型アプリケーションの回路構成をフレームワーク化し，それを用いた設計手順により設計を容

易化する.

回路構成の決定において，アプリケーション固有の処理の設計を容易化するため，複製および削除とアプリケーション固有の処理を分離することとした.また，フレームワークには回路間のインターフェイスの決定や再利用率の高いモジュールの作成を含む.

4.1. 回路構成のフレームワーク化

各ステージが行う処理はアプリケーション固有の処理以外に複製および削除の判定や制御，隣接ステージ間のデータのやり取りが必要となる．また，複製および削除の動作は基本的に同じであるが，複製および削除を行う条件および処理内容はアプリケーションによって異なる．これらの機能を実現するため，ステージを図 3 のように複製部，削除部，処理部，制御部の 4 つの機能に分割する.

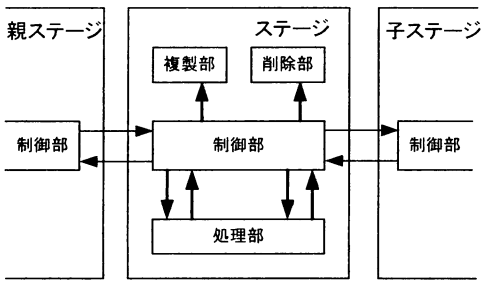

図 3　フレームワークの構成

このような構成にすることで，回路設計者は，アプリケーション固有の処理である処理部を設計するだけで，自己複製型アプリケーションを実現できる．また，必要に応じて制御部で行われる複製および削除の条件判定を変更できる.

以下，それぞれについて説明する.

(a) 制御部

制御部は，親ステージおよび子ステージとのデータ通信，処理部とのデータ通信，また複製・削除の判定を行い，必要に応じて複製部・削除部へ命令を発行する.

制御部は，図 3 の様に隣接するステージとの通信ポートを持つ.

複製・削除の条件については，あらかじめ決められたパターンを複数用意しておき，設計者が任意に選択できるようにする.もし，必要な複製あるいは削除条件が用意されていない場合は，設計者が新たに設計を行う.

また制御部は，処理部へのデータの流れを制御し，親ステージ，子ステージとの通信はすべて制御部を通じて行われる．親・子ステージの制御部で扱うデータは，処理部間で扱うデータと同じとする.

(b) 複製部

複製部は，制御部からの複製命令を受け取るとステージ全体を複製する．複製終了後，子ステージとの通信経路を設定し，複製完了信号を制御部へ発行する.

複製部は，制御部との通信のため，複製命令受信ポートと複製完了信号発行ポートをもつ.

ステージの構成による複製範囲の差異を複製部が含むことで，回路設計者はアプリケーションごとに複製動作を考慮する必要がない.

(c) 削除部

削除部は，制御部からの複製命令を受け取ると自身のステージ全体を削除する．削除部は制御部との通信のため，削除命令受信ポートをもつ.

ステージの構成による削除されるべき領域の違いを削除部が含むことで，回路設計者はアプリケーションごとに削除動作を考慮する必要はない.

(d) 処理部

フレームワークの設計方針により，処理部はそれのみで動作可能なように設計するものとし，複製および削除動作について考慮する必要はない.

ただし，処理完了時には PP 上に未送信のデータが残らないように設計する必要がある.

4.2. 複製および削除の条件

提案手法の前提として，ステージの複製および削除の条件について述べる．以降，複製元のステージを親ステージ，複製されたステージを子ステージ，一番元になるステージを初期ステージ，最後に複製されたステージを末端ステージと呼ぶことにする.

まず，複製を行う条件については，

・定められた信号が入力された時

・定められた回数だけ入力を受け取った時

などが考えられ，以上のようなときに複製を行う.

次に，削除については，以下の制約を与える.

・子ステージが存在しない

・定められた削除命令を受け取る

これらの制約により，末端ステージ側から順に削除されるため，未処理のデータが残らないことが保障される.

4.3. 入出力データフォーマット

フレームワークを適用するとき処理部を変更せずに用いることができるようにステージ間のデータフォーマットを以下のように決める.

初期ステージへの入出力データは，フレームワーク非適用時の入出力データと同一である．そのため，入力データは，特別な命令を含まず設定した条件に応じて制御部内で複製および削除命令が生成される．こうすることで，複製および削除動作がフレームワーク内

に隠蔽され，設計者は処理部に対する入出力のみを考慮するだけでよい．

4.4. フレームワークを用いた設計手法

提案したフレームワークを用いた自己複製型アプリケーションの設計手法を示す．

自己複製型アプリケーションは，以下の3手順によって設計を行う．

(1) 処理部の設計
(2) 複製および削除条件の設定
(3) フレームワークとの接続

まず，処理部の設計に関して述べる．図4のように，均一な処理部を静的に配置することで動作可能であるならば，処理部に変更を加える必要はない．上記のような処理部でない場合や，新たに設計を行う場合は以下の2点に従い処理部を変更する．

・複製および削除動作について考慮しない
・各処理部はまったく同じものである

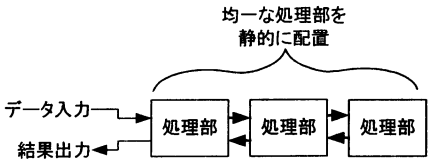

図 4　処理部

次に，複製および削除の条件の設定に関しては，あらかじめ用意された判定回路の中から適したものを選択する．もし，必要とされる複製・削除条件に対して適したものが用意されていない場合は，**4.2** で定義された複製・削除判定回路の仕様を満たす回路を新たに設計し，制御部へと組み込むことで必要とされる複製および削除条件を設定することができる．

最後に，設計を行った処理部とフレームワークとの接続を行う．図5で示すように，データの入出力はすべて制御部を通過させる．入力されるデータは，処理部で処理されるデータのみとする．

このように設計を行った初期ステージを一つ配置しデータを入力することで，必要に応じて自律的に自己を複製し処理が行われる．また，処理が終了した不要なステージを削除することで，ハードウェアリソースを解放する．

なお本稿では取り上げなかったが，複製・削除判定回路の制御部への組み込みとフレームワーク，処理部間の接続を自動化するツールを開発している．これらの自動化ツールを用いることによって回路設計者はこの点を考慮する必要もなくなる．

5. PCA デバイスへの実装

フレームワークにおける複製部，削除部および制御部を PCA-1 へ実装した．詳細を以下に述べる．処理部については6章において述べる．

5.1. 複製部，削除部

複製部および削除部は，メモリ読み出し部，メモリオブジェクトおよびメモリオブジェクト内のデータをデコードする回路からなる．各回路は制御部からそれぞれ複製および削除命令を受け取ると，メモリオブジェクトに格納したコマンド群を読み出し出力することで複製および削除を行う．

メモリオブジェクトの内容は，オブジェクト間の経路情報，ステージを複製および削除するための命令群で構成される．この内容はアプリケーションごとに異なるため，回路の構成情報を入力することで，メモリオブジェクトの内容を自動的に出力するプログラムを作成した．

5.2. 制御部

制御部は，複製および削除を行うための条件判定部とデータフロー制御からなり，必要に応じて複製命令および削除命令を複製部または削除部へ発行する．回路構成は図5のようになる．複製判定は，親ステージからの入力もしくは子ステージへの出力を，削除判定は子ステージからの入力もしくは親ステージへの出力を用いる．図5では条件判定部が2つずつあるが，どちらを用いるかは条件によって決定される．削除の判定を子ステージからの入力で行う場合，処理が終了したことを確認してから削除を行わなければならない．

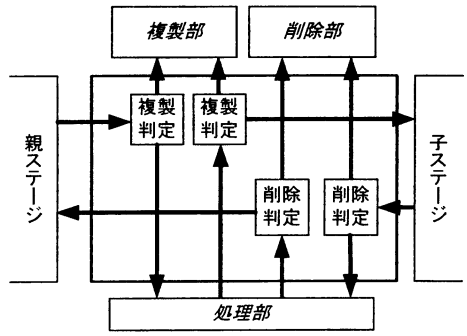

図 5　制御部の回路構成

6. 設計容易化手法の評価

提案した設計容易化手法をエラトステネスの篩い，セレクションソート，高速アダマール変換，多倍長乗算器に適用することで評価を行った．

6.1. 処理部の設計と複製・削除条件の決定

各アプリケーションにおける処理部の設計と，複

製・削除条件について述べる．

(a) エラトステネスの篩い

この処理への入力は，データ幅4ビットの2から始まる昇順に並べられた自然数の列とその最後に入力の終了信号を付加したものとする．出力として入力されたデータの素数の列とその最後に終了信号を付加したものとする．

各ステージの動作は，最初に入力された値を保持し，次から入力される値が保持した値の倍数であれば破棄し，そうでなければ次のステージに渡す．このとき，各ステージが保持した数が素数となるので，各ステージは自身が保持した値を親ステージへ出力し次に子ステージから受け取った値を親ステージへ出力する．その結果，初期ステージからの出力として素数列が得られる．

複製条件は親ステージからの終了信号以外の入力とし，削除条件は処理結果における終了信号の出力とした．

(b) セレクションソート

この処理への入力は，データ幅4ビットのデータとその最後に終了信号を付加したものとし，出力として整列された入力データとその最後に終了信号を付加したものとする．

セレクションソートを実行する各ステージの動作は以下のようになる．親ステージから最初に入力された値を保持し，次入力と保持している値を比較し，大きい値を子ステージに出力し，小さいほうの値を保持する．

複製条件は終了信号以外のデータを受け取ることとし，削除条件はデータの終端信号を親ステージへ出力することとする．

(c) 高速アダマール変換

アダマール変換は直交変換の一種であり，音声・画像などの圧縮や認識などの分野で用いられる[9]．高速アダマール変換は，計算回数を減らし処理を高速にするため，バタフライ演算と呼ばれる方法を用いる．

この処理への入力は，まずパラメータとして4ビット×4×必要となるステージ数，次に処理すべきデータとして4ビット×3を単位とした任意の長さのデータとする．出力としてアダマール変換を行ったデータとする．

本実装では，文献[9]において高速アダマール変換回路を再帰的な処理として表現する手法を元にし，これを自己複製型アプリケーションとして変更をおこない処理部とした．この処理部への入力は処理内容を決定するためのパラメータと終端信号を付加した処理すべきデータであり，並列に入力される．

複製条件として処理内容を決定するためのパラメータをもちいた．削除条件として処理すべきデータの終端信号とした．

(d) 多倍長乗算器

この処理への入力はデータの終了信号を付加した任意のビット数の乗数と被乗数である．出力として乗算結果と終了信号を出力する．

多倍長乗算を実行する各ステージの動作は以下のようになる．まず，乗数と被乗数が並列で入力される．入力された乗数の下位4ビットと被乗数を乗算することで部分積を計算する．また，乗数の残りビットを子ステージへ出力する．そして，小ステージから受け取った処理結果を4ビットシフトしたのち部分積に加算し親ステージへ出力する．この処理を行うには乗数のビット数／4+1ステージを必要とする．

複製条件は，乗数における親ステージからの終了信号以外の入力とし，削除条件はデータの終端信号を親ステージへ出力することとする．

6.2. 評価と考察

(a)，**(b)**，**(c)**および**(d)** で述べたアプリケーションをPCA-1上に実装を行った結果，正常に複製動作することを確認した．ただし，高速アダマール変換回路の削除部の実装は行っていない．

以下，実装したフレームワークについての考察を行う．まず，フレームワーク部(ステージにおいて処理部を除いた複製部，削除部，制御部)のPCA Cell数に対するアプリケーションに依存する回路のPCA Cell数の比率を表2に示す．

表2　フレームワーク部におけるアプリケーション依存となる回路の割合

	フレームワーク部 (PCA Cell)	アプリケーション依存(PCA Cell)	比率
エラトステネス	16	2	13%
ソート	16	2	13%
アダマール	20	6	30%
多倍長乗算器	20	6	30%

フレームワーク部において各アプリケーションに共通部分のCell数は14であり，アプリケーションに依存しない．エラトステネスの篩い，セレクションソートにおいてアプリケーション依存となる2 Cellは複製・削除判定回路であり，これらの2つのアプリケーションでは同じ回路を使用している．高速アダマール変換においてアプリケーション依存となる6 Cellは複製および削除判定回路 (2Cell)と処理部の入出力ポート増加による配線を制御する回路(4 Cell)である．

以上により，フレームワーク部の使用Cellの大部分

はアプリケーションに非依存の共通なものであり，異なるアプリケーションに適用する場合でも使用可能である．

表 3 処理部とフレームワーク部の面積比較

	処理ユニット全体 (PCA Cell)	フレームワーク部 (PCA Cell)	フレームワーク部の割合
エラトステネス	24	17	71%
ソート	24	17	71%
アダマール変換	88	22	25%
多倍長乗算器	47	27	57%

表3は各アプリケーションのステージ全体(メモリオブジェクトを含む)と処理部の面積比を示す．処理部の使用 Cell 数が大きくなるにつれて，フレームワーク部の割合が小さくなる．複製あるいは削除命令を格納するメモリオブジェクトのセル数が処理部の Cell 数に応じて増加する．しかし，表 1 で示す命令を 1024 個保持することが可能なメモリ 1 Cell で複製可能な回路面積は約 20 Cell であり，また，削除可能な回路面積は約 100 Cell であることから回路面積の増加に比べ非常に小さい．

図 6 は，高速アダマール変換回路におけるフレームワーク適用時と非適用時の，入力データ数と処理時間の関係を示すグラフである．フレームワーク適用したアプリケーションの処理時間は，初期ステージを配置し，データの入力から処理が終了するまでの時間とする．また非適用時の処理時間は，ステージを必要数だけ静的に配置し終わった状態に対して，データを入力したときから処理終了までの時間とする．

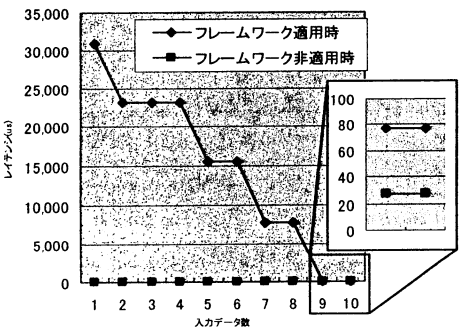

図 6 自己複製型アプリケーションと処理部のみのレイテンシ

入力されたデータ数が 1 のとき，レイテンシに 30.8ms の差がある．これは自己複製型アプリケーションが自己複製も行うためである．一つのステージを複製するのに必要な時間は 7.745ms である．回路を全て複製し終わった後は，フレームワーク非適用時と同じくレイテンシは一定であり，77μs となる．フレームワーク非適用時のレイテンシよりも 50μs 多いのは制御部を通ることによる．

7. まとめ

本論文では，自己複製型アプリケーションのための設計容易化手法を提案し，評価を行った．提案した手法は，複製・削除などの自己複製型アプリケーションでの典型的な機能やパラメータをフレームワーク化することで，回路設計者のアプリケーション設計の負担を軽減するものである．

評価としてエラトステネスの篩い，セレクションソート，高速アダマール変換回路および多倍長乗算器を実装し，利用容易性の評価を行った．その結果，設計の容易化が計られたことを確認した．

文 献

[1] 末吉敏則，飯田全広，“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.

[2] H. Ito, R. Konishi, H. Nakada, K. Oguri, M Inamori, and A Nagoya, “Dynamically Reconfigurable Logic LSI - PCA-1. The First Realization of the Plastic Cell Architecture,” IEICE TRANS. INF. & SYST., Vol.E86-D, pp. 859-867, No.5, May 2003.

[3] H. Ito, R. Konishi, H. Nakada, T. Tsuboi, Y. Okuyama, and A. Nagoya, “Dynamically Reconfigurable Logic LSI:PCA-2,” IEICE TRANS. INF. & SYST., Vol.E87-D, pp. 2011-2020, No.8, Aug. 2004.

[4] 天野英晴，“リコンフィギュラブル技術の最近の動向，” 第 17 回 回路とシステム軽井沢ワークショップ，pp. 253-258, Apr. 2004.

[5] T. Sugawa, K. Ide, T. Sato, “Dynamically Reconfigurable Processor Implemented with IPFlex’s DAPDNA Technology,” IEICE TRANS. INF & SYST., Vol. E87-D, No. 8, pp.1997-2003, Aug. 2004.

[6] 多田一仁，湯浅隆史，泉知論，尾上孝雄，中村行宏，“自己複製可能デバイスにおける適応的負荷分散モデル，” 第 16 回 回路とシステム(軽井沢)ワークショップ論文集，pp. 171-176, Apr. 2003.

[7] 伊藤俊之，高橋宏章，山本英邦，小幡悄，北道淳司，黒田研一，“PCA におけるマスタ・スレーブ型適応的負荷分散モデルの提案と実装，” 第 16 回回路とシステム(軽井沢)ワークショップ論文集，pp.177-182, Apr. 2003.

[8] 田中一好，山本英邦，倉田圭吾，北道淳司，“PCA における自律再構成機能を用いた分散処理モデルの提案と実装，” FIT2003, C-007, 2003.

[9] 高橋宏章，北道淳司，黒田研一，“N 次元アダマール変換アルゴリズムの提案と動的再構成可能デバイスへの実装，” 信学技報 Vol.10 VLD2003-123, pp. 352-358, 2004.