計測自動制御学会東北支部 第208回研究集会 (2003.5.22)
資料番号 208-8

転置形構造を有する分散演算形LMS適応フィルタのVLSIアーキテクチャ

VLSI Architecture of LMS Adaptive Filter using Distributed Arithmetic employing Non-canonical Structure

○高橋 強*, 恒川佳隆**, 田山典男**

○Kyo Takahashi*, Yoshitaka TSUNEKAWA**, Norio TAYAMA**

*岩手県立産業技術短期大学校, **岩手大学

*Iwate Industrial Technology Junior College, **Iwate University

キーワード： 分散演算(distributed arithmetic), 出力滞在時間(output latency), 転置形構造(non-canonical form), VLSIアーキテクチャ(VLSI architecture)

連絡先： 〒028-3615 岩手県紫波郡矢巾町大字南矢幅10-3-1 岩手県立産業技術短期大学校電子技術科
高橋 強, Tel.: (019)-697-9082, Fax.: (019)-697-9089, E-mail:kyo@iwate-it.ac.jp

1. はじめに

現在，適応フィルタはエコーキャンセラ，ノイズキャンセラ，自動等化器などに用いられており，さまざまな分野においてその実現の必要性が高まっている．適応フィルタを実現する際には，高速性，良好な収束特性，短い出力滞在時間（Latency），低消費電力，小規模ハードウエア等のように，定係数フィルタの場合よりも多くの性能が要求される．しかし，これらの相反する要求をすべて満足する適応フィルタを実現することは非常に困難である．また，TV会議などで必要な音響エコーキャンセラにおいては，室内音場のインパルス応答を高速に推定する能力とインパルス応答変動に対する追従性が要求される[1]．そのため，非常に高次の適応フィルタが必要とされている．

適応フィルタの実現に関する研究は乗算器や除算器を多用したパイプライン処理により高速性を追求するものが数多くなされてきた[2, 3, 4]．しかし，乗算器や除算器はハードウエア規模や消費電力が大きいため，高次においては膨大なハードウエア量と消費電力が必要となる．

これまで，分散演算をLMS適応アルゴリズムに適用した分散演算形LMS適応フィルタ(DA-ADF)と，高次における低消費電力，小規模ハードウエア，そして収束速度の改善を可能にするマルチメモリブロック構造を適用した分散演算型LMS適応フィルタ(MDA-ADF)を提案した[8]．分散演算では，乗算器を用いないいわゆるマルチプライヤレスの構成が可能であるため，小規模なハードウエアと低消費電力を達成することが可能である．MDA-ADFは，高次においても良好な収束速度と推定精度，高速なサンプリングレート，短い出力滞在時間，小規模ハードウエア，低消費電力を達成可能な高性能適応フィルタである．

本報告では，タップ数に全く依存しない短い出力滞在時間と，さらに高速なサンプリングレートを達成可能な転置形構造を有するMDA適応フィルタを提案する．なお，これまでLMSに対する転置形アルゴリズムは報告されているが[9, 10]，分散演算形LMSアルゴリズムの転置形構造に関する報告例はない．提案法は，以下の手順で得られる．まず，MDA適応フィルタにカットセットリタイミングを適用して転置形構造を有するマルチメモリブロック構成の分散演算形LMSアルゴリズム(NCMDAアルゴリズム）を導出する．このアルゴリズムの出力滞在時間は入力信号語長と加算時間にのみ依存し，フィルタのタップ数には依存しない．次いで，1時刻前の誤差信号を用いて適応関数空間を更新するディレード・アップデートを適用する．これにより，出力計算と更新動作を並列に実行することを可能とし，サンプル周期を短縮することも可能にする．これを，DNCMDAアルゴリズムと呼ぶ．DNCMDAアルゴリズムの収束特性を計算機シミュレーションによって評価し，通常のLMSアルゴリズムと同等の良好な収束特性を有することを示す．最後に，DNCMDAアルゴリズムを用いた適応フィルタ(DNCMDA-ADF)の高性能VLSIアーキテクチャを提案して評価する．これより，提案するDNCMDA-ADFは良好な収束速度，高速なサンプリングレート，短い出力滞在時間，小規模ハードウエア，低消費電力などの要求される性能を同時に満たすことが可能であることを示す．

2. 分散演算形LMSアルゴリズムとマルチメモリブロック構造

分散演算はベクトルの内積演算を部分積のシフト加算により実行する演算手法である．従来，分散演算は定係数のベクトル演算に用いられてきたが，係数が時変となる適応フィルタにおいても有効な演算手法となる[5, 6, 7, 8]．

2.1 分散演算形LMS適応アルゴリズム

タップ数Nの入力信号ベクトルを

$$\boldsymbol{S}(k) = [s(k), s(k-1), \ldots, s(k-N+1)]^T \qquad (1)$$

タップ数Nの係数ベクトルを

$$\boldsymbol{W}(k) = [w_0(k), w_1(k), \ldots, w_{N-1}(k)]^T \qquad (2)$$

とする．入力信号ベクトル$\boldsymbol{S}(k)$を

$$\boldsymbol{S}(k) = \boldsymbol{A}(k)\,\boldsymbol{F} \qquad (3)$$

と定義すると，フィルタ出力は次式で表される．

$$y(k) = \boldsymbol{S}^T(k)\,\boldsymbol{W}(k) = \boldsymbol{F}^T\,\boldsymbol{A}^T(k)\,\boldsymbol{W}(k) \qquad (4)$$

(3)式，(4)式において，アドレスマトリクス$\boldsymbol{A}(k)$は

$$\boldsymbol{A}(k) = \begin{bmatrix} b_0(k) & b_0(k-1) & \cdots & b_0(k-N+1) \\ b_1(k) & b_1(k-1) & \cdots & b_1(k-N+1) \\ \vdots & \vdots & \ddots & \vdots \\ b_{B-1}(k) & b_{B-1}(k-1) & \cdots & b_{B-1}(k-N+1) \end{bmatrix}^T \qquad (5)$$

スケーリングベクトル$\boldsymbol{F}$は

$$\boldsymbol{F} = [-2^0, 2^{-1}, \ldots, 2^{-(B-1)}]^T \qquad (6)$$

である．なお，Bは入力信号の語長を表し，$b_i(k)$は時刻kにおける入力信号$s(k)$のiビット目の値である．(4)式において

$$\boldsymbol{P}(k) = \boldsymbol{A}^T(k)\,\boldsymbol{W}(k) \qquad (7)$$

と定義する．ここで，$\boldsymbol{P}(k)$は$\boldsymbol{A}(k)$の列ベクトルに対するタップ係数$\boldsymbol{W}(k)$の部分積を要素に持つB次ベクトル

$$\boldsymbol{P}(k) = [p_0(k), p_1(k), \ldots, p_{B-1}(k)]^T \qquad (8)$$

である．これにより，(4)式は

$$y(k) = \boldsymbol{F}^T \boldsymbol{P}(k) \tag{9}$$

となる．

次に，LMSアルゴリズムに分散演算を適用する．LMSアルゴリズムは次式で表される[11]．

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + 2\mu e(k)\,\boldsymbol{S}(k) \tag{10}$$

ここで，誤差信号$e(k)$は

$$e(k) = d(k) - y(k) \tag{11}$$

であり，$d(k)$は所望信号を表す．(10)式の両辺に左から$\boldsymbol{A}^T(k)$を掛けて部分積を定義すると，

$$\begin{aligned} &\boldsymbol{A}^T(k)\,\boldsymbol{W}(k+1) \\ &\quad = \boldsymbol{A}^T(k)\{\boldsymbol{W}(k) + 2\mu e(k)\,\boldsymbol{A}(k)\,\boldsymbol{F}\} \end{aligned} \tag{12}$$

となり，さらに，(12)式において

$$\boldsymbol{P}(k+1) = \boldsymbol{A}^T(k)\,\boldsymbol{W}(k+1) \tag{13}$$

と置くと次の更新式が得られる．

$$\boldsymbol{P}(k+1) = \boldsymbol{P}(k) + 2\mu e(k)\,\boldsymbol{A}^T(k)\,\boldsymbol{A}(k)\,\boldsymbol{F} \tag{14}$$

この更新式では，マトリクスの乗算$\boldsymbol{A}^T(k)\,\boldsymbol{A}(k)$を実行する必要がある．しかし，乗算は多くの演算時間が必要であり，また乗算器はハードウエア規模や消費電力が大きい．そこで，入力信号に統計的な性質を仮定して$\boldsymbol{A}^T(k)\,\boldsymbol{A}(k)\,\boldsymbol{F}$を次のように簡略化する[8]．

$$\boldsymbol{A}^T(k)\,\boldsymbol{A}(k)\,\boldsymbol{F} = 0.25N\,\boldsymbol{F} \tag{15}$$

これを(14)式に適用して，分散演算形LMS適応アルゴリズム(DAアルゴリズム)は

$$\boldsymbol{P}(k+1) = \boldsymbol{P}(k) + 0.5\mu N e(k)\,\boldsymbol{F} \tag{16}$$

となる．

タップ数Nに対して部分積は2^N個存在するが，全ての部分積から構成される集合を全適応関数空間(Whole Adaptive Function Space, WAFS)と呼ぶ．前述の$\boldsymbol{P}(k)$は，時刻kにおいて出力計算に用いられるWAFSのB個の要素である．これを適応関数空間(Adaptive Function Space, AFS)と呼ぶ．

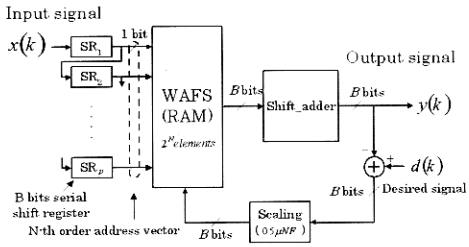

Fig. 1　Basic structure of DA adaptive filter.

更新式においてNおよびμを2のべき乗で考えた場合，更新値は誤差$e(k)$に対するシフト操作のみで求めることが可能となり，高速なリアルタイム処理が実現できる．DAアルゴリズムを用いた適応フィルタ(DA-ADF)の構成をFig. 1に示す．WAFSはRAMを用いて実現されるが，高次での実現を考慮した場合，RAMの容量は膨大となるため消費電力とハードウエア規模の点で不利である．また，WAFSの要素を更新する確率が減少するため収束速度も劣化する[7, 8]．この問題を解決するために，次節で説明するマルチメモリブロック構造が提案されている．

2.2　MDA適応アルゴリズム

マルチメモリブロック構造では，適応関数空間を分割して個々の空間を小容量化する．これにより，消費電力とハードウエア規模の増加を抑制し，収束速度を向上させることが可能になる[7, 8]．マルチメモリブロック構造を適用した分散演算形LMS適応アルゴリズムをMDA適応アルゴリズムと呼ぶ．

NタップのDA-ADFの適応関数空間をM個に分割した場合のMDA適応アルゴリズムについて

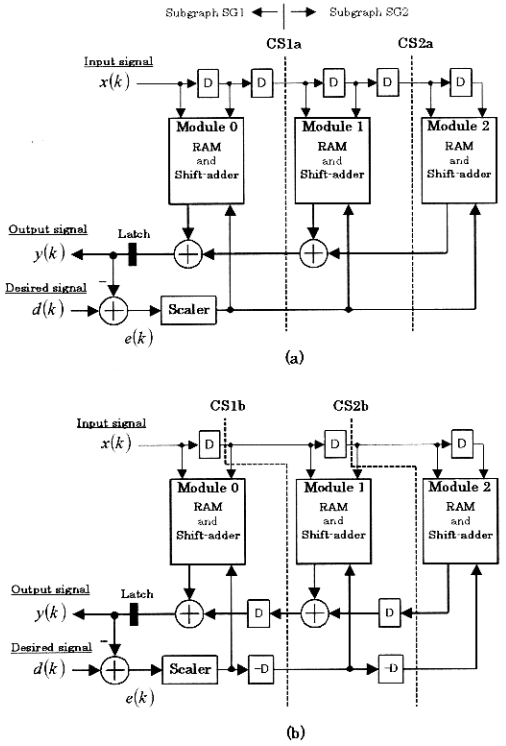

Fig. 3 Transformation of MDA-ADF from canonical form to non-canonical form applying a cut-set retiming (N=6, M=3, R=2). (a) Applying cut-set retiming to MDA-ADF. (b) Applying cut-set retiming to the structure resulting from (a).

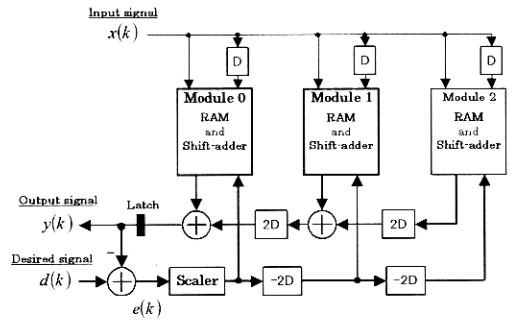

Fig. 4 Structure of non-causal NCMDA-ADF.

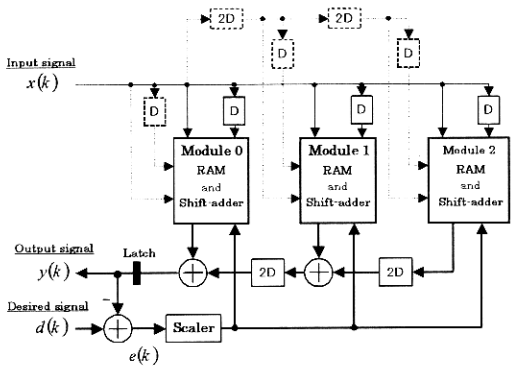

Fig. 5 Structure of causal NCMDA-ADF.

ダ-ラインが構成されている．なお，この例では2回のリタイミングを行ったが，一般にはR回のリタイミングを実行する必要がある．しかし，誤差のフィードバックラインに挿入された”逆時間の遅延器”は因果律を満たさないため，この構成は実現不可能である．そこで，M個に分割された各モジュールの更新値における誤差信号と入力信号の時間差を保ちながら，誤差信号$e(k)$を用いるための時間のシフトを以下のように行う．

$$
\begin{aligned}
Module1 \quad &: \quad x(k)e(k+2) \rightarrow x(k-2)e(k) \\
&: \quad x(k-1)e(k+2) \rightarrow x(k-3)e(k) \\
Module2 \quad &: \quad x(k)e(k+4) \rightarrow x(k-4)e(k) \\
&: \quad x(k-1)e(k+4) \rightarrow x(k-5)e(k)
\end{aligned}
$$

これにより，非因果的である Fig. 4 の構成は，因果律を満たす Fig. 5 に変換され，NCMDA-ADF の構成が得られる．新たに追加された遅延器と各モジュールへの入力信号（点線で表示）は，適応関数空間を更新するために用いられる．なお，この例以外についても同様の操作を実行することにより変換が可能である．

これより，M個に分割されたm番目のモジュール出力$y_m(k)$は

$$
\begin{aligned}
y_m(k) &= \boldsymbol{F}^T \boldsymbol{P}_{m,0}(k) \\
& \qquad m = 0, 1, \cdots, M-1 \qquad (22)
\end{aligned}
$$

となる．ここで，

$$
\boldsymbol{P}_{m,0}(k) = \boldsymbol{A}_0^T(k)\, \boldsymbol{W}_m(k) \qquad (23)
$$

である．そして，フィルタ出力$y(k)$は

$$
y(k) = \sum_{i=0}^{M-1} y_i(k - Ri) \qquad (24)
$$

となる．ここで，Rはアドレス線数である．

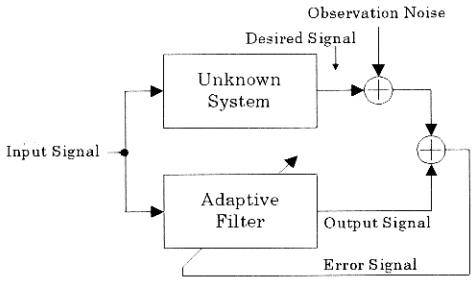

Fig. 6 Simulation model.

分割されたm番目の適応関数空間に対する更新式は，Fig. 5 より，

$$\boldsymbol{P}_m(k+1) = \boldsymbol{P}_m(k) + 0.5\mu Re(k)\,\boldsymbol{F} \qquad (25)$$

となる．なお，

$$\boldsymbol{P}_m(k) = \boldsymbol{A}_m^T(k)\,\boldsymbol{W}_m(k) \qquad (26)$$

である．

3.2 DNCMDAアルゴリズムの導出

NCMDAアルゴリズムにおいて，出力計算動作と更新動作を並列に実行することを考える．NCMDAアルゴリズムにおける更新動作は，誤差信号が求められないと開始することができない．そこで，更新値を求めるために1時刻前の誤差信号を用いることにする．この更新方法をディレードアップデートと呼ぶ．更新式は以下のように表される．

$$\boldsymbol{P}'_m(k+1) = \boldsymbol{P}'_m(k) + 0.5\mu Re(k-1)\,\boldsymbol{F} \qquad (27)$$

なお，適応関数空間を以下のように定義した．

$$\begin{aligned} \boldsymbol{P}'_m(k) &\triangleq \boldsymbol{A}_m^T(k-1)\,\boldsymbol{W}_m(k) \\ \boldsymbol{P}'_m(k+1) &\triangleq \boldsymbol{A}_m^T(k-1)\,\boldsymbol{W}_m(k+1) \\ & \qquad m = 0, 1, \cdots, M-1 \end{aligned} \qquad (28)$$

この更新式により，更新動作は必ずしも出力計算が終了した後に実行されなくても良い．

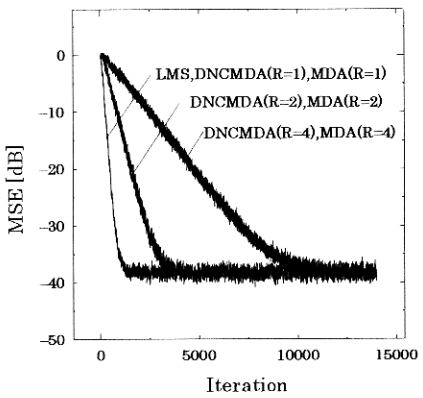

Fig. 7 Convergence properties when the input signal was white gaussian noise.

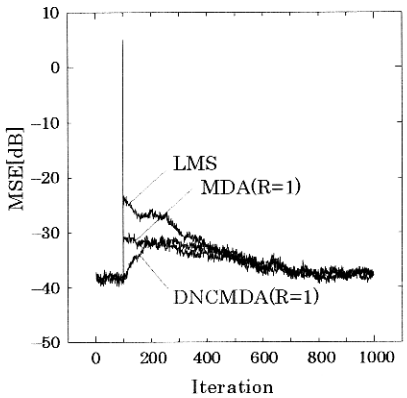

Fig. 8 Convergence properties when an impulsive noise was included at the iteration of 100.

4. 収束特性

提案するDNCMDA適応アルゴリズムの収束特性を評価する．シミュレーションモデルは，Fig. 6 に示すシステム同定問題である．未知システムはタップ数120の低域通過FIRフィルタ，入力信号は平均0，分散0.05の白色信号である．そして，未知システムの出力には観測雑音として-40.13[dB]の入力信号と無相関な白色信号を加えた．なお，ステップサイズパラメータは同じMSEを達成し，かつ最も高速に収束する値を設定した．また，比較の対象は通常のLMSアルゴリズム，MDAアルゴリズムである． Fig. 7 より，提案するDNCMDAアルゴリズムはMDAアルゴリズムと同等の収束特性を示しており，また，$R = 1$においてはMDAアル

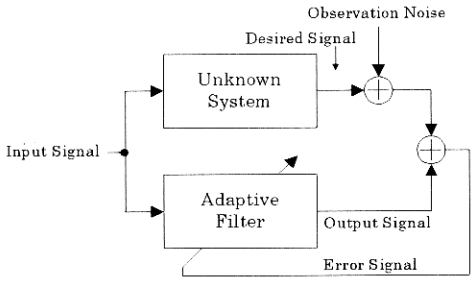

Fig. 6 Simulation model.

分割されたm番目の適応関数空間に対する更新式は，Fig. 5 より，

$$\boldsymbol{P}_m(k+1) = \boldsymbol{P}_m(k) + 0.5\mu Re(k)\,\boldsymbol{F} \qquad (25)$$

となる．なお，

$$\boldsymbol{P}_m(k) = \boldsymbol{A}_m^T(k)\,\boldsymbol{W}_m(k) \qquad (26)$$

である．

3.2 DNCMDAアルゴリズムの導出

NCMDAアルゴリズムにおいて，出力計算動作と更新動作を並列に実行することを考える．NCMDAアルゴリズムにおける更新動作は，誤差信号が求められないと開始することができない．そこで，更新値を求めるために1時刻前の誤差信号を用いることにする．この更新方法をディレードアップデートと呼ぶ．更新式は以下のように表される．

$$\boldsymbol{P}'_m(k+1) = \boldsymbol{P}'_m(k) + 0.5\mu Re(k-1)\,\boldsymbol{F} \qquad (27)$$

なお，適応関数空間を以下のように定義した．

$$\begin{aligned} \boldsymbol{P}'_m(k) &\triangleq \boldsymbol{A}_m^T(k-1)\,\boldsymbol{W}_m(k) \\ \boldsymbol{P}'_m(k+1) &\triangleq \boldsymbol{A}_m^T(k-1)\,\boldsymbol{W}_m(k+1) \\ & \qquad m = 0, 1, \cdots, M-1 \end{aligned} \qquad (28)$$

この更新式により，更新動作は必ずしも出力計算が終了した後に実行されなくても良い．

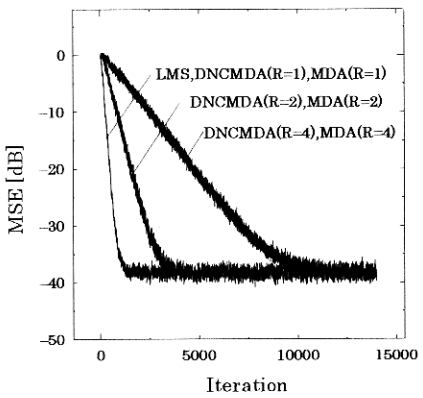

Fig. 7 Convergence properties when the input signal was white gaussian noise.

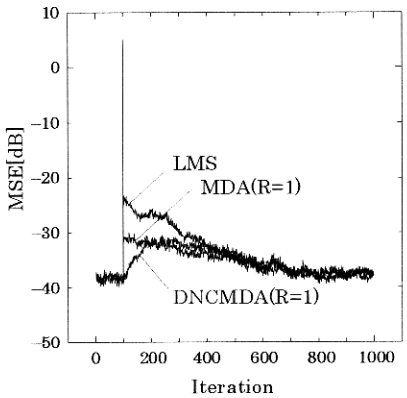

Fig. 8 Convergence properties when an impulsive noise was included at the iteration of 100.

4. 収束特性

提案するDNCMDA適応アルゴリズムの収束特性を評価する．シミュレーションモデルは，Fig. 6 に示すシステム同定問題である．未知システムはタップ数120の低域通過FIRフィルタ，入力信号は平均0，分散0.05の白色信号である．そして，未知システムの出力には観測雑音として-40.13[dB]の入力信号と無相関な白色信号を加えた．なお，ステップサイズパラメータは同じMSEを達成し，かつ最も高速に収束する値を設定した．また，比較の対象は通常のLMSアルゴリズム，MDAアルゴリズムである． Fig. 7 より，提案するDNCMDAアルゴリズムはMDAアルゴリズムと同等の収束特性を示しており，また，$R = 1$においてはMDAアル

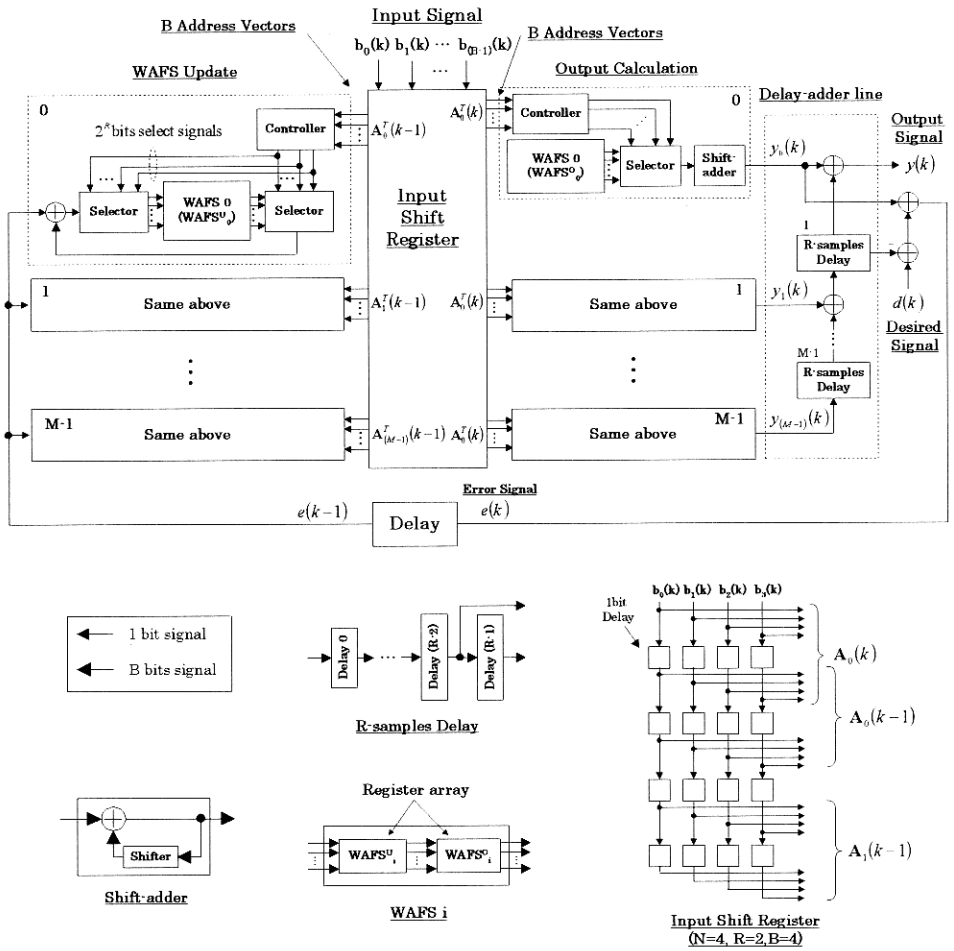

Fig. 9 Proposed VLSI architecture of DNCMDA-ADF.

ゴリズムと同様に通常のLMSアルゴリズムと同等の収束特性を示している．

次に，収束状態において観測雑音にインパルス性ノイズが加えられた場合の収束特性を Fig. 8 に示す．この例では，繰り返し数100回において0.03のインパルスノイズを加えており，DNCMDAとMDAはともに$R=1$である．繰り返し数100回におけるMSEの値は，LMSが5.02[dB]，DNCMDAとMDAはともに1.66[dB]であり，それ以降の劣化も小さいことがわかる．これらより，分散演算形アルゴリズムはインパルス性ノイズに対して低感度であることがわかる．また，101回以降のDNC-MDAの劣化は200回付近で最大を示している．これは，インパルス性ノイズの影響を受けたm番のモジュール出力がフィルタ出力に現われるまで$mR-1$回の繰り返しが必要になるためである．

5. VLSIアーキテクチャ

提案するDNCMDA-ADFのVLSIアーキテクチャを Fig. 9 に示す．このアーキテクチャは，入力信号レジスタ(N+1words)，M個の出力計算モジュールとディレーアダーライン，ディレードアップデートのための誤差の遅延器，M個の適応関数空間更新モジュールから構成される．WAFSは出力計算用のWAFS_i^Oと更新動作用のWAFS_i^Uから構成され，容量はどちらも2^Rwordsである．

[出力計算]

各モジュールの出力信号は(22)式を用いて計算されるが，これらのモジュールは並列に動作する

ため，M個のモジュール出力は同時に得られる．それらはディレーアダーラインに入力され，最終段のRサンプルディレーの出力と出力計算モジュール0の出力の和がフィルタ出力$y(k)$になる．また，誤差信号は次式にしたがって計算を実行する．

$$\begin{aligned} e(k) &= d(k) - y(k) \\ &= d(k) - \sum_{i=0}^{M-1} y_i(k - Ri) \\ &= \{d(k) - \sum_{i=1}^{M-1} y_i(k - Ri)\} - y_0(k) \end{aligned}$$

最終段のRサンプルディレー中の信号

$$\sum_{i=1}^{M-1} y_i(k - Ri)$$

と所望信号$d(k)$との差をあらかじめ求めておき，出力計算モジュール0の出力$y_0(k)$が得られた時点でこれを引く．これにより，誤差信号はフィルタ出力信号と同時に得られる．

[更新動作]

(27)式を用いて，分割されたM個の適応関数空間を並列に更新する．なお，更新動作に用いられる誤差信号は1時刻前の$e(k-1)$であり，適応関数空間を指定するアドレスマトリクスも1時刻前の$\boldsymbol{A}_m(k-1)$を用いる．

次に，タイミングチャートをFig. 10に示す．これより，出力計算の過程（WAFS_i^Oからの部分積の読み出しとシフト加算）とWAFSの更新動作

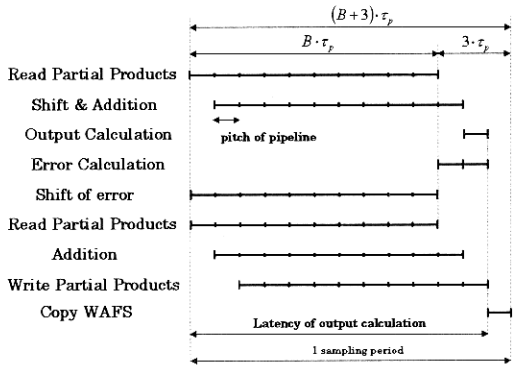

Fig. 10 Timing chart of proposed DNCMDA-ADF.

(WAFS_i^Uからの部分積の読み出し，更新値の加算，そしてWAFS_i^Uへの書き込み）が並列に実行されている．この並列動作はディレードアップデートにより可能になり，サンプリング周期を大幅に短縮することができる．WAFS_i^Uを更新した後に，WAFS_i^Uに蓄えられている部分積を出力計算用のWAFS_i^Oに書き込む．以上より，出力滞在時間τ_o，サンプリング周期T，そしてサンプリング周波数F_sは

$$\tau_o = (B+2)\cdot\tau_p \tag{29}$$

$$T = (B+3)\cdot\tau_p \tag{30}$$

$$F_s = 1/T \tag{31}$$

となる．なお，τ_pはパイプラインのピッチを表す．

6. VLSI評価

提案するDNCMDA-ADFのアーキテクチャをVLSI設計システムPARTHENONを用いて評価する[12]．実部品として用いたセルライブラリの設計ルールは0.6μmCMOSスタンダードセル(VLSIテクノロジ社)，電源電圧は5.0[V]である．演算に用いるデータ形式は2の補数表現による語長16ビットの固定小数点とする．なお，比較対象としてMDA-ADFの高速形構成，乗算器を用いたパイプライン処理の代表例としてDLMS-ADF [3]，そして出力滞在時間の短い構成としてPipelined-LMS-ADFを用いた[4]．Table. 1にタップ数60,120，アドレス線数R=2,3,4,5に対するDNCMDA-ADFのVLSI評価結果を示す．これより，サンプリングレートと出力滞在時間はタップ数とアドレス線数には全く依存せず，ゲート数，面積，そして消費電力はアドレス線数やタップ数に依存していることがわかる．R=2では収束速度が最も高速で，かつゲート数，面積，そして消費電力が最小になる．次に，MDA-ADFとの比較をTable. 2に示す．サンプリングレートと出力滞在時間はタップ数とアドレス線数にかかわらずDNCMDA-ADFが優れてい

Table 1　VLSI evaluations of proposed DNCMDA-ADF.

Number of taps	60				120			
Number of address line	1	2	3	4	1	2	3	4
Number of division	60	30	20	15	120	60	40	30
Machine cycle[ns]	13	13	13	13	13	13	13	13
Sampling rate[MHz]	4.05	4.05	4.05	4.05	4.05	4.05	4.05	4.05
Latency[ns]	234	234	234	234	234	234	234	234
Power dissipation[W]	0.54	0.39	0.41	0.48	1.08	0.79	0.81	0.97
Area[mm^2]	19.15	14.42	14.74	18.06	38.24	28.81	29.45	36.12
Number of gates	172,446	129,068	131,770	160,302	344,346	257,768	263,350	320,592

Table 2　Ratio of proposed DNCMDA-ADF to MDA-ADF.

Number of taps	60				120			
Number of address line	1	2	3	4	1	2	3	4
Number of division	60	30	20	15	120	60	40	30
	Ratio between DNCMDA and MDA							
Machine cycle[ns]	1	1	1	1	1	1	1	1
Sampling rate[MHz]	2.11	2.05	2.05	2.00	2.16	2.11	2.11	2.05
Latency[ns]	0.78	0.82	0.82	0.86	0.75	0.78	0.78	0.82
Power dissipation[W]	1.41	1.73	2.09	2.41	1.45	1.78	2.15	2.48
Area[mm^2]	0.62	0.81	0.98	1.18	0.62	0.81	0.99	1.19
Number of gates	0.63	0.82	0.99	1.19	0.63	0.82	0.99	1.19

る．またR=2では，タップ数60，120のどちらにおいてもサンプリングレート，出力滞在時間，面積，そしてゲート数の全てにおいてDNCMDA-ADFが有利であり，R=3ではほぼ同等であることがわかる．なお，消費電力はサンプリング周波数に比例するが，N=60，R=2では2.05倍のサンプリングレートに対して消費電力が1.73倍，N=120，R=2では2.11倍のサンプリングレートに対して消費電力が1.78倍であるため，サンプリングレートを同じ条件にした場合ではDNCMDA-ADFの消費電力が小さい．

DLMS-ADFの評価結果を Table. 3 に示す．これは高速性を重視した構成であるため，21.27MHzという非常に高速なサンプリングレートを有するが，出力滞在時間が極端に大きく，乗算器を多用するため消費電力，ゲート数，面積も非常に大きい．

Pipelined-LMS-ADFの評価結果を Table. 4 に示す．これは，出力滞在時間の短く，かつ，比較的高速なサンプリングレートを有しているがゲート規模と面積はDLMSよりもさらに大きい．

以上より，提案するDNCMDA-ADFは良好な収束速度，高速なサンプリングレート，短い出力滞在時間，小規模ハードウエア，低消費電力などの要求される性能を同時に満たすことが可能である．

Table 3　VLSI evaluations of the DLMS-pipelined adaptive filter.

Number of taps	60	120
Machine cycle[ns]	47	47
Sampling rate[MHz]	21.27	21.27
Latency[ns]	2868	5688
Power dissipation[W]	10.69	21.28
Area[mm^2]	64.06	127.61
Number of gates	582,647	1,160,567

Table 4　VLSI evaluations of the LMS-pipelined adaptive filter.

Number of taps	60	120
Machine cycle[ns]	107	107
Sampling rate[MHz]	9.34	9.34
Latency[ns]	34	34
Power dissipation[W]	11.14	22.24
Area[mm^2]	150.26	300.05
Number of gates	1,368,893	2,733,593

7.　まとめ

本報告では，転置形構造を有する分散演算形LMS適応アルゴリズムにディレードアップデートを適用したDNCMDAアルゴリズムと，これを用

いた適応フィルタのVLSIアーキテクチャを提案した．導出したDNCMDAアルゴリズムは，通常のLMSアルゴリズムと同等の良好な収束特性を有し，また，インパルス性ノイズに対して低感度であることが計算機シミュレーションによって明らかになった．次いで，DNCMDA-ADFの高性能VLSIアーキテクチャを提案して評価した．その結果，タップ数120，アドレス線数1において，MDA-ADFよりも2.16倍の高速なサンプリングレート，0.75倍の短い出力滞在時間，0.62倍の面積で実現可能である．以上より，提案するDNCMDA-ADFは，タップ数に全く依存しない短い出力滞在時間と高速なサンプリングレートを有し，そして，小規模なハードウエアと低消費電力を同時に満たす構成が可能であることが明らかになった．

今後は，収束条件などの理論解析を進める予定である．

参考文献

1) 牧野，小泉，“エコーキャンセラの室内音場における適応特性の改善について，” 信学論(A)，vol.J71-A，no.12，pp.2212–2214，Dec. 1988.

2) K.J. Raghunath and K.K. Parhi, “A 100 MHz pipe-lined RLS adaptive filter,” Proc. IEEE ICASSP'95, Detroit, Michigan, pp.3187–3190, May 1995.

3) 松原，西川，貴家，“Delayed LMSアルゴリズムに基づくパイプライン適応フィルタ，” 信学論(A)，vol.J79-A，no.5，pp.1050–1057，May 1996.

4) A. Harada,K. Nishikawa and H. Kiya, “Pipelined Architecture of the LMS Adaptive Digital Filter with the Minimum Output Latency,”IEICE Trans. Fundamentals,pp.1578-1585,Vol.E81-A No.8 1998

5) C.F.N. Cowan and J. Mavor, “New digital adaptive-filter implementation using distributed-arithmetic techniques,” IEE Proc., vol.128, Pt.F, no.4, pp.225–230, Aug. 1981.

6) C.F.N. Cowan, S.G. Smith and J.H. Elliott, “A digital adaptive filter using a memory-accumulator architecture:theory and realization,” IEEE Trans. Acoust., Speech & Signal Process., vol.31, no.3, pp.541–549, Jun. 1983.

7) C.H. Wei, J.J. Lou, “Multimemory block structure for implementing a digital adaptive filter using distributed arithmetic,” IEE Proc., vol.133, Pt.G, no.1, pp.19–26, Feb. 1986.

8) 恒川，高橋，豊田，三浦，“分散演算によるマルチプライヤレスLMS適応フィルタの高性能アーキテクチャ，” 信学論(A)，vol.J-82-A, no.10, pp.1518-1528, Oct. 1999.

9) R.W. Stewart, J.J.Soraghan, T.S. Durrani,“Noncanonical FIR Filters and Adaptive Signal Processing,”Electronics Letters, 16th March., 1989, Vol.25, No.6, pp.414-415.

10) W.S.Gan, J.J.Soraghan, R.W. Stewart, T.S. Durrani,“The Non-canonical LMS algorithm (NCLMS):Characteristics and Analysis, ”IEEE International Conference Acoustics, Speech and Signal Processing,Vol.3,pp.2137-2140,1991.

11) B. Widrow and M. E. Hoff,‘Adaptive Switching Circuit,” IRE EWSCON Conv. Rec.,pp 96-104, 1960.

12) NTTデータ通信株式会社，“PARTHENON User's Manual,” 1990.