計測自動制御学会東北支部 第195回研究集会 (2001.6.22)
資料番号 195-8

ブロックLMS適応フィルタの超高速VLSIアーキテクチャ

Very High-Speed VLSI Architecture of Block LMS Adaptive Filter

○高橋 強*，恒川佳隆**，田山典男**，関 享士郎**

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

*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： ブロックLMSアルゴリズム(Block LMS Algorithm), 超高速(Very High-Speed), VLSI アーキテクチャ(VLSI Architecture), 適応フィルタ (Adaptive Filter)

連絡先： 〒028-3615 岩手県紫波郡矢巾町大学南矢幅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

近年，移動体通信システムの急速な進展により適応信号処理の必要性がますます高まっている．また，動画像通信などへの要求から高速なデータ伝送レートでの通信が望まれており，適応フィルタを実現する際には高サンプリングレートでの処理を可能とする高性能アーキテクチャが望まれている．

適応フィルタに用いる適応アルゴリズムは，LMSアルゴリズム（Least Mean Square Algorithm），NLMSアルゴリズム（Normalized LMS），RLSアルゴリズム（Recursive Least Square Algorithm）などがよく知られているが，演算量が少ない割には比較的良好な収束特性を示すLMSが広く用いられている．これまで，LMSアルゴリズムやその改良アルゴリズムに基づく適応フィルタの構成法として，分散演算を用いた構成法[7]，DLMS（Delayed LMS）アルゴリズムに基づく構成法[8]，Pipelined LMSに基づく構成法[9]などが提案されている．分散演算に基づく構成法は出力滞在時間が非常に短くハードウエア規模が非常に小さい特徴をもつ．DLMSとPipelined LMSは高速なサンプリングレートを実現するためにパイプライン処理をおこない，タップ数に依存せずに約10MHz程度のサンプリングレートを達成可能である[7]．

LMSの高速アルゴリズムとして，ブロック処理をLMSに適用したブロックLMS（Block LMS, BLMS）アルゴリズムが提案されている[3]．BLMSアルゴリズムは，L個の入力信号をまとめて並列に処理することにより，高速なサンプリングレートを達成可能である．Clarkらは，プロセッサを用いたシリアル処理によりBLMS適応フィルタを実現することを検討し，演算量を低減するために周波数領域のBLMSアルゴリズムを用いた．以来，時間領域のBLMSアルゴリズムを周波数領域に変換した周波数領域のBLMSアルゴリズムが非常に効果的であるとこれまで考えられてきた．しかし，この方法はブロックLMSアルゴリズムが本来有す

る並列性を有効に活用しておらず，またパラメータを周波数領域に変換するために高速フーリエ変換を用いることから，タップ数に対して滞在時間が急激に増加するという問題がある．また，これまでBLMS適応フィルタの効果的なアーキテクチャに関する検討は行われていない．

本報告では，時間領域におけるBLMSアルゴリズムの完全並列処理に基づく超高速適応フィルタの効果的VLSIアーキテクチャを提案する．ブロック長LのBLMSアルゴリズムは，基本的にLMSアルゴリズムの並列実現として得られるため，ハードウエア規模はLMS適応フィルタのL倍になる．したがって，このままでは膨大なハードウエアが必要であり消費電力やハードウエア規模の点で不利である．そこで，我々はBLMS適応フィルタの構成要素であるLMS適応フィルタを極力小規模化して適用する．これにより，提案するBLMS適応フィルタはハードウエアの増加を最小限にとどめることが可能であり，ブロック長Lにほぼ比例したサンプリングレートを実現可能である．また，これまで適応フィルタの収束速度に対する評価は繰り返し回数で議論されてきた．しかし，この評価方法ではアルゴリズムの収束速度を評価するのみで，サンプリング周期を決定するアーキテクチャをも含めた適応フィルタの収束速度を評価しているとは言えない．そこで，我々はアルゴリズムとアーキテクチャを考慮した適応フィルタの収束速度を総合的に評価するために繰り返し回数とサンプリング周期の積である収束時間を用いることにする．

2. ブロックLMS適応フィルタ

2.1 LMS適応フィルタ

LMSアルゴリズムは，最急降下法に基づいて2乗平均誤差を最小にする一方法で，演算量が少ない割には良好な収束を示すことから現在でも最もよく用いられている適応アルゴリズムである[1)2)]．LMSアルゴリズムを用いた適応フィルタ(LMS Adaptive Filter,LMS-ADF)は以下のように表される．

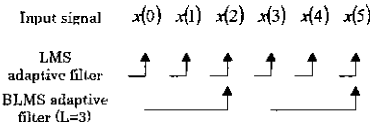


Fig. 1 Difference of update timming of LMS and BLMS adaptive filter.

p次入力信号ベクトルとp次タップ係数ベクトルをそれぞれ

$$\boldsymbol{X}(k) = [x(k), x(k-1), \cdots, x(k-p+1)]^T \quad (1)$$

$$\boldsymbol{W}(k) = [w_1(k), w_2(k), \cdots, w_p(k)]^T \quad (2)$$

と表すと，出力信号$y(k)$は次式で表される．

$$y(k) = \boldsymbol{W}(k)^T \boldsymbol{X}(k) \quad (3)$$

次に，係数ベクトルの更新式は次のように表される．

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + 2\mu e(k)\boldsymbol{X}(k) \quad (4)$$

ここで，誤差信号$e(k)$は

$$e(k) = d(k) - y(k) \quad (5)$$

と表され，$d(k)$は所望信号である．

2.2 ブロックLMS適応フィルタ

LMS適応フィルタは出力計算と係数更新を毎時刻実行するが，これに対してブロック長LのブロックLMS適応フィルタ(Block LMS Adaptive Filter, BLMS-ADF)はL時間ごとにのみフィルタ係数を更新する． Fig. 1 に入力信号に対する係数更新のタイミングを示す．LMS-ADFは入力信号が得られる毎に更新動作を行うのに対し，BLMS-ADFにおける更新動作はL時間毎にL時間分の更新計算を一括して実行する．また，出力計算も同様である．

ブロックLMS適応フィルタは以下のように表される[3)]．時刻$(j-1)L+1$から時刻jLまでのL

時間分の式 (3) と式 (5) をまとめて表現すると，

$$\boldsymbol{y}(j) = \boldsymbol{\Gamma}(j)\,\boldsymbol{w}(j) \tag{6}$$

$$\boldsymbol{e}(j) = \boldsymbol{d}(j) - \boldsymbol{y}(j) \tag{7}$$

となる．ただし，jはブロック番号であり，

$$\boldsymbol{y}(j) = [y((j-1)L+1), y((j-1)L+2), \cdots, y(jL)]^T \tag{8}$$

$$\boldsymbol{d}(j) = [d((j-1)L+1), d((j-1)L+2), \cdots, d(jL)]^T \tag{9}$$

$$\boldsymbol{e}(j) = [e((j-1)L+1), e((j-1)L+2), \cdots, e(jL)]^T \tag{10}$$

$$\boldsymbol{\varphi}(jL) = [x(jL), x(jL-1), \cdots, x(jL-p+1)]^T \tag{11}$$

$$\boldsymbol{\Gamma}(j) = [\boldsymbol{\varphi}((j-1)L+1),\ \boldsymbol{\varphi}((j-1)L+2), \cdots,\ \boldsymbol{\varphi}(jL)]^T$$

$$\begin{matrix} x((j-1)L+1) & \cdots & x((j-1)L-p+2) \\ x((j-1)L+2) & \cdots & x((j-1)L-p+3) \\ \vdots & \ddots & \vdots \\ x(jL) & \cdots & x(jL-p+1) \end{matrix} \tag{12}$$

$$\boldsymbol{w}(j) = [w_1(j), w_2(j), \cdots, w_p(j)]^T \tag{13}$$

とおいた．また，タップ係数$\boldsymbol{w}(j)$の更新式は

$$\boldsymbol{w}(j+1) = \boldsymbol{w}(j) + \frac{2\mu_B}{L}\,\boldsymbol{\Gamma}^T(j)\,\boldsymbol{e}(j) \tag{14}$$

$$= \boldsymbol{w}(j) + \frac{2\mu_B}{L}\,\boldsymbol{q}(j) \tag{15}$$

である．なお，

$$\boldsymbol{q}(j) = \boldsymbol{\Gamma}^T(j)\,\boldsymbol{e}(j) \tag{16}$$

$$= [q_1(j), q_2(j), \cdots, q_p(j)]^T \tag{17}$$

とおいた．

2.3 従来のBLMS適応フィルタ

BLMSアルゴリズムは，L個の入力信号に対して処理を行うためLMS適応フィルタに比較して演算量が大きくなる．そこで，プロセッサを用いたシリアル処理を行う従来のBLMS適応フィルタでは，演算量を低減するためにアルゴリズムを周波数領域に変換して用いていた[3]．この方法は，特に演算量の多い式(6)と式(16)において効果的である．Fig. 2 に一例を示すが，時間領域信号である$\Gamma(j)$と$\boldsymbol{w}(j)$をそれぞれフーリエ変換し，周波数領域で演算を行う．時間領域でのフィルタリングは係数ベクトルと入力信号ベクトルの畳み込み演算で実行されるが，周波数領域では乗算により実行されるため，大幅な演算量の削減が可能になる．そして，出力信号$\boldsymbol{y}(j)$は逆フーリエ変換により求められる．ここで，フーリエ変換は高速フーリエ変換(Fast Fourier Transform,FFT)，逆フーリエ変換は高速逆フーリエ変換(Inverse Fast Fourier Transform,IFFT)を用いることにより離散フーリエ変換(Discrete Fourier Transform,DFT)を用いた場合よりも処理時間を短縮可能である．また，従来のBLMS-ADFではFFTを用いることから，入力信号を効率よく利用するためにブロック長Lをタップ数pと等しくしている．

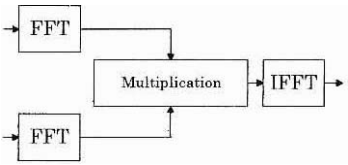


Fig. 2 Output calculation of conventional BLMS-ADF.

しかし，シリアル処理を行う従来法はブロックLMSアルゴリズム本来の性質である並列性を有効に活用していない．また，FFT，IFFTの使用は滞在時間を大幅に増加させる．そして，サンプリング周波数を決定する大きな要因であるブロック長に自由度がないという問題がある．

3. 収束特性

計算機シミュレーションにより，LMSアルゴリズムとBLMSアルゴリズムの収束特性を比較する．シミュレーションにはFig. 3 に示されるシステム同定モデルを用いる．ここで，未知システムはタップ数128のFIR形低域通過フィルタであり，適応フィルタも128タップの低域通過フィルタを用いている．入力信号は平均0の白色ガウス信号を用い，観測

雑音として入力信号とは無相関で平均0，S/N比が55.2dBの白色ガウス信号を加えた．LMS(BLMSの$L-1$と等価)とBLMS($L=100$)の収束特性をそれぞれFig. 4，Fig. 5に示す．なお，ステップサイズパラメータは，同じMSEを達成するという条件のもとで最も高速な収束速度を示す値を設定した．BLMSは，LMSに対してブロック長($L=100$)に相当する繰り返し回数だけMSEの減少は遅延するが，同等の収束特性を示している．そして，BLMSは100回毎に係数を更新するためMSEは階段状に減少していることがわかる．

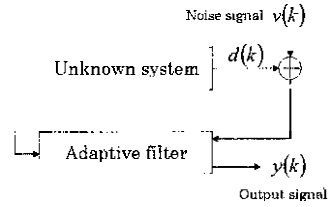


Fig. 3 Simulation model.

4. 超高速VLSIアーキテクチャ

4.1 BLMSの基本構成

BLMS適応フィルタは基本的にはLMS適応フィルタの並列実現として得られる．ブロック長Lに対する構成をFig. 6に示し，これを基本構成と呼ぶことにする．また，基本構成において用いるモジュールをFig. 7に示す．

LMS-ADFは，FIRフィルタモジュールと更新値計算モジュールから構成される．FIRフィルタモジュールでは，遅延器出力とタップ係数の積であるタップ出力を求め，そしてこれらの総和を求めることによりフィルタ出力を得る．ここで，タップ出力はp個の乗算器を並列に配置した乗算器アレーを，タップ出力の総和は加算器をツリー状に配置したツリーアダーを用いている．更新値計算モジュールでは，まず所望信号とフィルタ出力の差である誤差信号を求める．そして，誤差信号に$2\mu_B/L$を乗じてスケーリングを行い，これと入力信号ベクトルとの積である更新値ベクトルを求める．BLMS-ADFは，L個のLMS-ADFがタップ係数を共有するため，各LMS-ADFから得られたタップ係数の更新値の総和を求める必要がある．更新モジュールでは，L個のLMS-ADFから得られた更新値の総和をツリーアダーにより求めタップ係数に加えることにより係数を更新する．このように，BLMS-ADFはLMS-ADFによる並列構成であるため基本的にモジュール化が容易である．また，LMS-ADFはFIRフィルタモジュールと係数更新モ

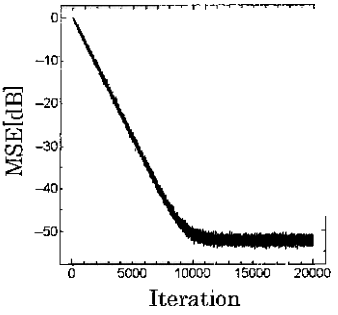


Fig. 4 Convergence property of LMS algorithm.

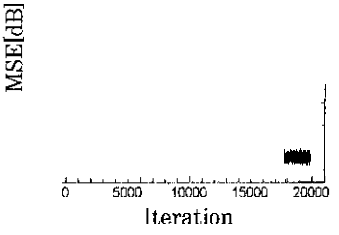
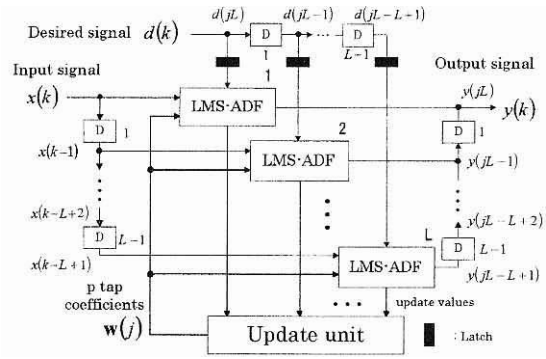
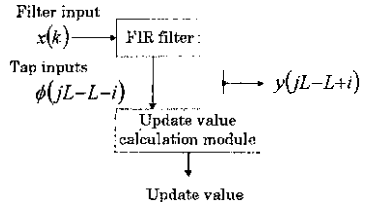


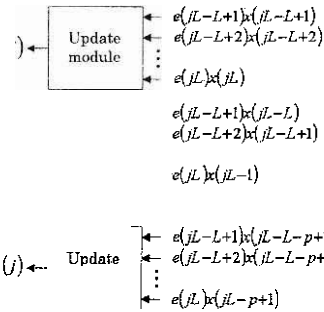
Fig. 5 Convergence property of BLMS algorithm (Block length 100).



(a) Basic BLMS adaptive filter
i=1,2,...,L

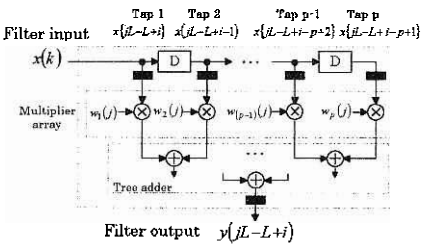


(b)LMS·ADF unit

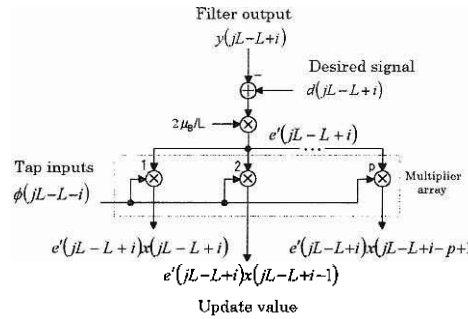


(c) Update unit

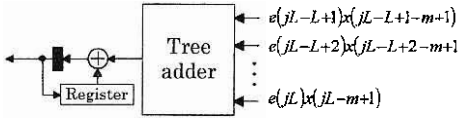
Fig. 6 Basic configuration of BLMS adaptive filter.



(a) FIR filter module
i=1,2,...,L



(b) Update value calculation module
i=1,2,...,L



(c) Update module
m=1,2,...,p

Fig. 7 Modules used in Basic BLMS-ADF

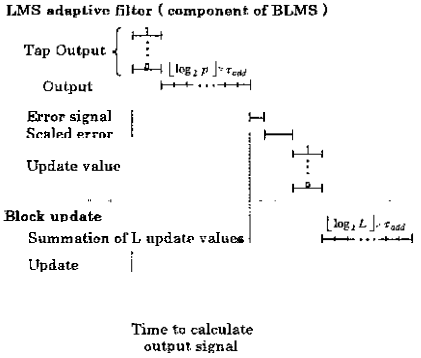


Fig. 8 Timming chart of basic BLMS adaptive filter.

ジュールにより構成されるが，それらは規則的な構成を有しているためにさらに細分化したモジュール構成が可能であることがわかる．また，信号の流れもスムーズであるためVLSI実現に非常に適した構成が可能である．

次に，タイミングチャートを Fig. 8 に示す．出力計算に必要な処理時間τ_{out}は，乗算時間をτ_{mul}，加算時間をτ_{add}とすると

$$\tau_{out} = \tau_{mul} + \lfloor log_2 p \rfloor \tau_{add} \tag{18}$$

である．ここで，$\lfloor a \rfloor$はaより大きい最小の整数を表す．誤差計算と係数更新をあわせて更新時間τ_{up}とすると，

$$\tau_{up} = 2\tau_{mul} + (2 + \lfloor log_2 L \rfloor)\tau_{add} \tag{19}$$

である．BLMS適応フィルタの動作周期はこれらを加えて

$$\tau_{BLMS} = \tau_{out} + \tau_{up} \tag{20}$$

$$= 3\tau_{mul} + (2 + \lfloor log_2 L \rfloor + \lfloor log_2 p \rfloor)\tau_{add} \tag{21}$$

である．入力サンプルあたりのサンプリング周期T_Sとサンプリング周波数F_Sは

$$T_S = \tau_{BLMS}/L \tag{22}$$

$$F_S = 1/T_S \tag{23}$$

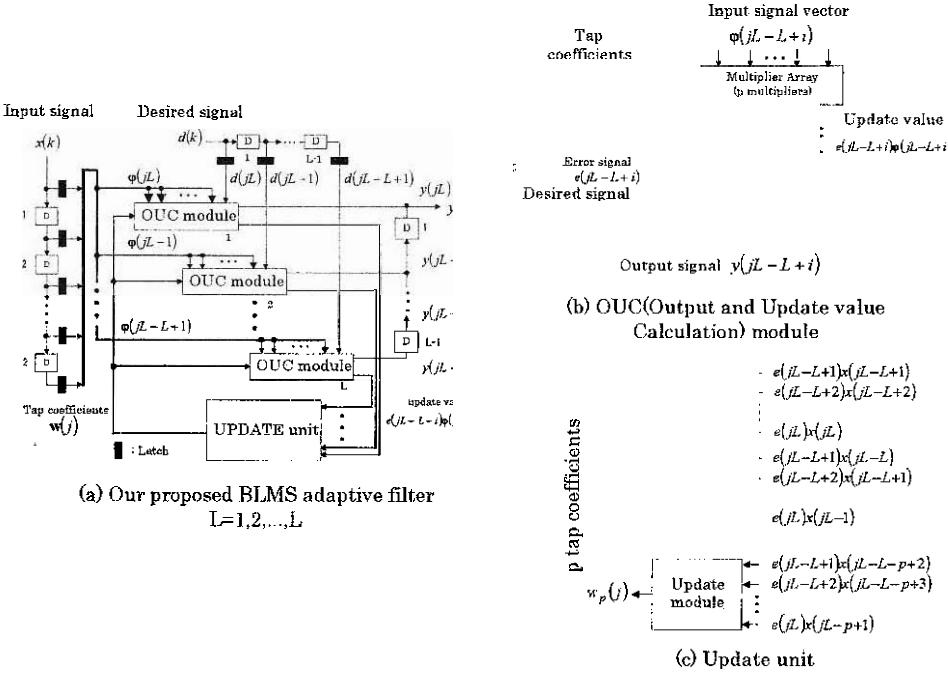
である．

4.2 高性能BLMSアーキテクチャ

BLMS適応フィルタはL個のLMS適応フィルタを並列に実行するため，ブロック長Lにほぼ比例してハードウエアが増加する．そこで，ハードウエア規模を小さく抑えるためにBLMS-ADFを構成するLMS-ADFを極力小規模化して適用する．

まず，BLMS-ADFの基本構成において，出力計算と更新値計算はどちらも乗算器アレーを用いるが，Fig. 8 のタイミングチャートより，これらは同時に動作することはないため共用することが可能であることがわかる．また，式(14)において更新値をスケーリングするために基本構成では乗算器を用いていた．しかし，パラメータ$2\mu_B/L$を2のべき乗で近似することにより乗算器は不要になり，誤差信号のシフト操作によりスケーリングを実行することが可能になる．さらに，L個のLMS適応フィルタに配置されている遅延器にも冗長性があるため，遅延器をFIRフィルタモジュール内に配置せずに外部に$p+L-2$個配置する．提案するBLMS-ADFの構成を Fig. 9 に示す．入力信号$x(k)$は$p+L-2$個の遅延器によって蓄積される．OUC(Output and Update value Calculation)モジュールは出力計算，誤差計算そして更新値計算を実行するが，基本構成において出力計算と更新値計算に用いている2つの乗算器アレーを共有するために，乗算器アレーの入力と出力にマルチプレクサ(Multiplexer,MUX)とデマルチプレクサ(Demultiplexer,DEMUX)を配置している．さらに，更新値計算において用いるステップサイズパラメータを2のべき乗で近似することにより，基本構成で用いていた乗算器を用いずにシフト操作によるスケーリングを可能としている．更新モジュール(基本構成のモジュールに同じ)は，求められたL個の更新値をツリーアダーを用いて加算しフィルタ係数を更新する．

この提案法においてもモジュール単位での規則的な構成が可能であり，ハードウエアの共有のために追加したマルチプレクサとデマルチプレクサにおける選択信号数も最小であるためデータの流れもスムーズである．これより，提案法はVLSI実現に適した構成である．



(a) Our proposed BLMS adaptive filter L=1,2,...,L

(b) OUC(Output and Update value Calculation) module

(c) Update unit

Fig. 9 Configuration of our proposed BLMS adaptive filter

Table 1 Comparison of operators between Basic and proposed BLMS-ADF. N_L and N_P indicates a number of adders used in the L inuts and P inputs Tree-adder,respectively.

Architecture	Multiplier	Adder	Register	MUX	DEMUX
LMS-ADF	2p+1	N_p+p+1	2p-1	0	0
Basic BLMS-ADF	L(2p+1)	L(1+N_p)+p(1+N_L)	2L(p+2)+2p-3	0	0
Proposed BLMS-ADF	Lp	L(1+N_p)+p(1+N_L)	6L+4p-4	L	L

Table 2 Comparison of number of gates between Basic and proposed BLMS-ADF for p=L=128

Architecture	Multiplier	Adder	Register	Mux	DeMux	Total
Basic BLMS-ADF	71,252,736	804,864	2,137,920	0	0	74,195,520
Proposed BLMS-ADF	35,487,744	804,864	73,536	1,343,488	1,101,824	38,811,456

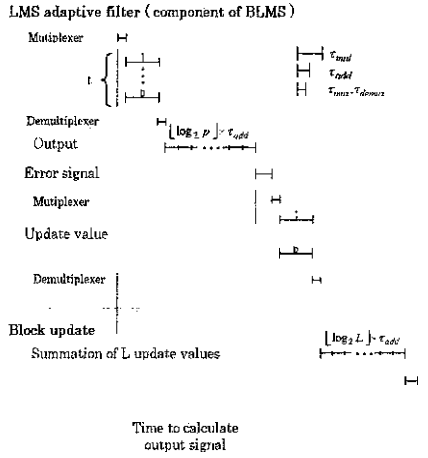


Fig. 10 Timming chart of proposed BLMS adaptive filter.

基本構成と提案法の演算器数を Table. 1 に，$p=L=128$ に対するゲート数の見積もりを Table. 2 に示す．ここで，乗算器は Booth のアルゴリズムに Wallace tree 方式と CLA 加算器を用いた構成，加算器は CLA 方式を用いた．また，語長は 16bit である．これより，提案法は基本構成に対して約 48% のゲート数が削減可能である．

次に，タイミングチャートを Fig. 10 に示す．出力計算に必要な処理時間τ'_{out}は，マルチプレクサとデマルチプレクサの滞在時間をそれぞれτ_{mux}，τ_{demux}とすると

$$\tau'_{out} = \tau_{mul} + \lfloor log_2 p \rfloor \tau_{add} + \tau_{mux} + \tau_{demux} \quad (24)$$

である．誤差計算と係数更新をあわせて更新時間τ'_{up}とすると，

$$\tau'_{up} = \tau_{mul} + (2 + \lfloor log_2 L \rfloor)\tau_{add} + \tau_{mux} + \tau_{demux} \quad (25)$$

である．BLMS適応フィルタの動作周期はこれらを加えて

$$\begin{aligned} \tau'_{BLMS} &= \tau'_{out} + \tau'_{up} \quad (26) \\ &= 2\tau_{mul} + (2 + \lfloor log_2 L \rfloor + \lfloor log_2 p \rfloor)\tau_{add} \\ &\quad + 2\tau_{mux} + 2\tau_{demux}, \quad (27) \end{aligned}$$

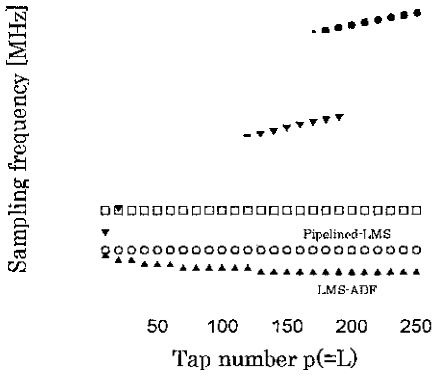


Fig. 11 Comparison of sampling Frequency.

入力サンプルあたりのサンプリング周期T'_Sとサンプリング周波数F'_Sは，

$$T'_S = \tau'_{BLMS}/L \quad (28)$$

$$F'_S = 1/T'_S \quad (29)$$

である．

ブロック長Lに対するサンプリング周波数を Fig. 11 に示す．なお，タップ数は$p=L$とし，演算器の滞在時間を$\tau_{add}=15ns$，$\tau_{mul}=35ns$，マルチプレクサとデマルチプレクサの滞在時間を$\tau_{mux}=\tau_{demux}=8ns$とした．また，従来法の数値は並列構成の場合である．同図より，タップ数128($L=128$)におけるサンプリング周波数を比較すると，従来法は並列構成を用いたとしても65.5MHzであるのに対し提案法は344.1MHzであり約5.3倍高速であることがわかる．したがって，我々の提案するブロックLMS適応フィルタの構成は処理速度に関して最適な構成である．

これらの構成で用いているアルゴリズムの収束回数は通常のLMS-ADFと同等であるため，アーキテクチャを含めた適応フィルタの収束速度は，この場合はサンプリング周期の比として評価することができる．これより，提案するBLMS-ADFはこれらの中で最も高速な収束速度を有する適応フィルタである．

以上より，提案法は基本構成と同等の超高速なサンプリング周波数を達成可能であり，かつゲート数を約1/2に削減可能である．

5. あとがき

本報告では，ブロックLMS適応フィルタの超高速VLSIアーキテクチャを検討した．提案法は機能毎に分割したモジュールによる規則的な構成が可能であり，またデータの流れもスムーズであることからVLSI実現に適した構成である．この際，ハードウエアの小規模化のために構成要素であるLMS適応フィルタを極力小規模化して適用した．提案する構成法では，タップ数とブロック長を128とした場合，約344.1MHzのサンプリングレートを達成可能であり，従来の構成に対して約5.3倍の高速化が可能である．これらより，提案法は処理速度に関して最適な構成である．さらに，任意のブロック長を選択可能であり，これにより所望のサンプリングレートを実現可能である．また，適応フィルタとしての収束速度をアルゴリズムとアーキテクチャを考慮して総合的に評価した結果，提案法の収束速度は非常に高速であることが示された．

今後は，詳細なVLSI評価を行うことにより提案法の有効性をさらに明らかにしていきたい．

参考文献

1) B. Widrow et al.,"Adaptive noise cancelling : Principles and applications," Proc. IEEE, vol.63, pp.1692–1716, Dec.1975.

2) 辻井重男，久保田一，古川利博，趙晋輝:適応信号処理，昭晃堂(1995)

3) Gregory A. Clark, Sanjit K. Mitra, Sydney R. Parker, "Block Implementation of Adaptive Digital Filters," IEEE Trans. Circuits and Syst., vol. CAS-28, pp.584–592, June. 1981.

4) A. Feuer, "Performance Analysis of the Block Least Mean Square Algorithm," IEEE Trans. Circuits and Syst., vol. CAS-32, pp.960–963, Sept. 1985.

5) A. Feuer, E. Weinstein, "Convergence Analysis of LMS Filters with Uncorrelated Gaussian DATA," IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-33, pp.222 230, Feb. 1985.

6) 飯國洋二:適応信号処理アルゴリズム,培風館(2000)

7) Y.Tsunekawa, K.Takahashi, S.Toyoda, M.Miura, "High-Performance VLSI Architecture of Multiplierless LMS Adaptive Filters Using Distributed Arithmetic," IEICE Trans. Fundamentals, vol.J-82-A, no.10, pp.1518-1528, Oct.1999.

8) K. Matsubara,K. Nishikawa,H. Kiya, "Pipelined Adaptive Filters Based on Delayed LMS Algorithm, " IEICE Trans. Fundamentals, vol.J79-A, no.5, pp.1050-1057, May.1996.

9) A. Harada, K Nishikawa and H. Kiya, "Pipelined architecture of the LMS adaptive digital filter with the minimum output latency," IEICE Trans. Fundamentals, vol.E81-A, no.8, pp.1578-1585, Aug. 1998.