4

An LMS Adaptive Filter Using Distributed Arithmetic - Algorithms and Architectures

Kyo Takahashi[1], Naoki Honma[2] and Yoshitaka Tsunekawa[2]
[1]*Iwate Industrial Research Institute*
[2]*Iwate University*
Japan

1. Introduction

In recent years, adaptive filters are used in many applications, for example an echo canceller, a noise canceller, an adaptive equalizer and so on, and the necessity of their implementations is growing up in many fields. Adaptive filters require various performances of a high speed, lower power dissipation, good convergence properties, small output latency, and so on. The echo-canceller used in the videoconferencing requires a fast convergence property and a capability to track the time varying impulse response (Makino & Koizumi, 1988). Therefore, implementations of very high order adaptive filters are required. In order to satisfy these requirements, highly-efficient algorithms and architectures are desired. The adaptive filter is generally constructed by using the multipliers, adders and memories, and so on, whereas, the structure without multipliers has been proposed.

The LMS adaptive filter using distributed arithmetic can be realized by using adders and memories without multipliers, that is, it can be achieved with a small hardware. A Distributed Arithmetic (DA) is an efficient calculation method of an inner product of constant vectors, and it has been used in the DCT realization. Furthermore, it is suitable for time varying coefficient vector in the adaptive filter. Cowan and others proposed a Least Mean Square (LMS) adaptive filter using the DA on an offset binary coding (Cowan & Mavor, 1981; Cowan et al, 1983). However, it is found that the convergence speed of this method is extremely degraded (Tsunekawa et al, 1999). This degradation results from an offset bias added to an input signal coded on the offset binary coding. To overcome this problem, an update algorithm generalized with 2's complement representation has been proposed (Tsunekawa et al, 1999), and the convergence condition has been analyzed (Takahashi et al, 2002). The effective architectures for the LMS adaptive filter using the DA have been proposed (Tsunekawa et al, 1999; Takahashi et al, 2001). The LMS adaptive filter using distributed arithmetic is expressed by DA-ADF. The DA is applied to the output calculation, i.e., inner product of the input signal vector and coefficient vector. The output signal is obtained by the shift and addition of the partial-products specified with the bit patterns of the N-th order input signal vector. This process is performed from LSB to MSB direction at the every sampling instance, where the B indicates the word length. The B partial-products

used to obtain the output signal are updated from LMB to MSB direction. There exist 2^N partial-products, and the set including all the partial-products is called Whole Adaptive Function Space (WAFS). Furthermore, the DA-ADF using multi-memory block structure that uses the divided WAFS (MDA-ADF) (Wei & Lou, 1986; Tsunakawa et al, 1999) and the MDA-ADF using half-memory algorithm based on the pseudo-odd symmetry property of the WAFS (HMDA-ADF) have been proposed (Takahashi et al, 2001). The divided WAFS is expressed by DWAFS.

In this chapter, the new algorithm and effective architecture of the MDA-ADF are discussed. The objectives are improvements of the MDA-ADF permitting the increase of an amount of hardware and power dissipation. The convergence properties of the new algorithm are evaluated by the computer simulations, and the efficiency of the proposed VLSI architecture is evaluated.

2. LMS adaptive filter

An N-tap input signal vector S(k) is represented as

$$\mathbf{S}(k) = \left[s(k), s(k-1), \cdots, s(k-N+1) \right]^T , \tag{1}$$

where, s(k) is an input signal at k time instance, and the T indicates a transpose of the vector. The output signal of an adaptive filter is represented as

$$y(k) = \mathbf{S}^T(k)\mathbf{W}(k) , \tag{2}$$

where, W(k) is the N-th coefficient vector represented as

$$\mathbf{W}(k) = \left[w_0(k), w_1(k), \cdots, w_{N-1}(k) \right]^T , \tag{3}$$

and the $w_i(k)$ is an i-th tap coefficient of the adaptive filter.

The Widrow's LMS algorithm (Widrow et al, 1975) is represented as

$$\mathbf{W}(k+1) = \mathbf{W}(k) + 2\mu e(k)\mathbf{S}(k) , \tag{4}$$

where, the e(k), μ and d(k) are an error signal, a step-size parameter and the desired signal, respectively. The step-size parameter deterimines the convergence speed and the accuracy of the estimation. The error signal is obtained by

$$\mathrm{e(k)} = \mathrm{d(k)} - \mathrm{y(k)} . \tag{5}$$

The fundamental structure of the LMS adaptive filter is shown in Fig. 1. The filter input signal s(k) is fed into the delay-line, and shifted to the right direction every sampling instance. The taps of the delay-line provide the delayed input signal corresponding to the depth of delay elements. The tap outputs are multiplied with the corresponding coefficients, the sum of these products is an output of the LMS adaptive filter. The error signal is defined as the difference between the desired signal and the filter output signal. The tap coefficients are updated using the products of the input signals and the scaled error signal.

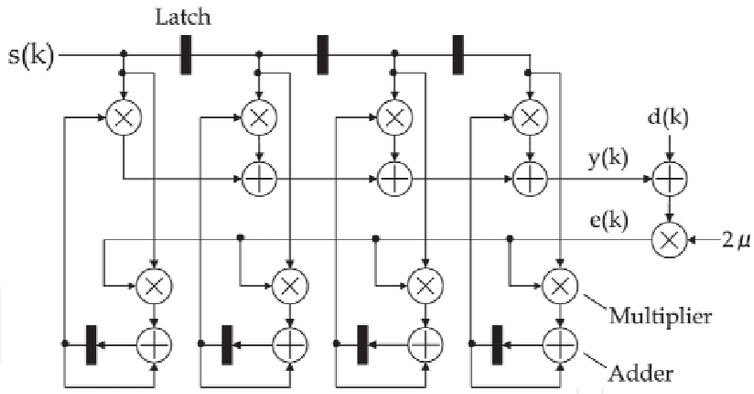


Fig. 1. Fundamental Structure of the 4-tap LMS adaptive filter.

3. LMS adaptive filter using distributed arithmetic

In the following discussions, the fundamentals of the DA on the 2's complement representation and the derivation of the DA-ADF are explained. The degradation of the convergence property and the drastic increase of the amount of hardware in the DA-ADF are the serious problems for its higher order implementation. As the solutions to overcome the problems, the multi-memory block structure and the half-memory algorithm based on the pseudo-odd symmetry property of WAFS are explained.

3.1 Distributed arithmetic

The DA is an efficient calculation method of an inner product by a table lookup method (Peled &Liu, 1974). Now, let's consider the inner product

$$y = \mathbf{a}^T \mathbf{v} = \sum_{i=1}^{N} a_i v_i \tag{6}$$

of the N-th order constant vector

$$\mathbf{a} = \left[\mathrm{a}_0, \mathrm{a}_1, \cdots \mathrm{a}_{(N-1)} \right]^{\mathrm{T}} \tag{7}$$

and the variable vector

$$\mathbf{v} = \left[v_0, v_1, \cdots v_{(N-1)} \right]^T . \tag{8}$$

In Eq.(8), the v_i is represented on B-bit fixed point and 2's complement representation, that is,

$$-1 \le \mathrm{v}_{\mathrm{i}} < 1$$

and

$$v_i = -v_i^0 + \sum_{k=1}^{B-1} v_i^k 2^{-k}, \quad i = 0,1,\cdots,N-1 . \tag{9}$$

In the Eq.(9), v_i^k indicates the k-th bit of v_i, i.e., 0 or 1. By substituting Eq.(9) for Eq.(6),

$$y = -\Phi\left(v_0^0, v_1^0, \cdots, v_{N-1}^0\right) + \sum_{k=1}^{B-1} \Phi\left(v_0^k, v_1^k, \cdots, v_{N-1}^k\right) 2^{-k} \tag{10}$$

is obtained. The function Φ which returns the partial-product corresponding argument is defined by

$$\Phi\left(v_0^k, v_1^k, \cdots, v_{N-1}^k\right) \equiv \sum_{i=0}^{N-1} a_i v_i^k \ . \tag{11}$$

Eq.(10) indicates that the inner product of y is obtained as the weighted sum of the partial-products. The first term of the right side is weighted by -1, i.e., sign bit, and the following terms are weighted by the 2^{-k}. Fig.2 shows the fundamental structure of the FIR filter using the DA (DA-FIR). The function table is realized using the Read Only Memory (ROM), and the right-shift and addition operation is realized using an adder and register. The ROM previously includes the partial-products determined by the tap coefficient vector and the bit-pattern of the input signal vector. From above discussions, the operation time is only depended on the word length B, not on the number of the term N, fundamentally. This means that the output latency is only depended on the word length B. The FIR filter using the DA can be implemented without multipliers, that is, it is possible to reduce the amount of hardware.

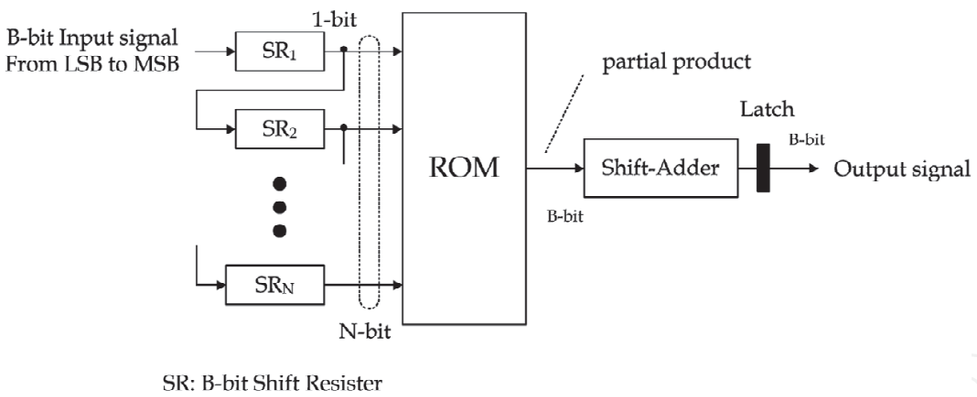


Fig. 2. Fundamental structure of the FIR filter using distributed arithmetic.

3.2 Derivation of LMS adaptive algorithm using distributed arithmetic

The derivation of the LMS algorithm using the DA on 2's complement representation is as follows. The N-th order input signal vector in Eq.(1) is defined as

$$\mathbf{S}(k) \equiv \mathbf{A}(k)\mathbf{F} \, . \tag{12}$$

Using this definition, the filter output signal is represented as

$$y(k) = \mathbf{S}^T(k)\mathbf{W}(k) = \mathbf{F}^T\mathbf{A}^T(k)\mathbf{W}(k) \, . \tag{13}$$

In Eq.(12) and Eq(13), an address matrix which is determined by the bit pattern of the input signal vector is represented as

$$\mathbf{A}(k)=\begin{bmatrix} b_0(k) & b_0(k-1) & \cdots & b_0(k-N+1) \\ b_1(k) & b_1(k-1) & \cdots & b_1(k-N+1) \\ \vdots & \vdots & \ddots & \vdots \\ b_{B-1}(k) & b_{B-1}(k-1) & \cdots & b_{B-1}(k-N+1) \end{bmatrix}^T, \tag{14}$$

and a scaling vector based on the 2's complement representation is represented as

$$\mathbf{F}=\left[-2^0, 2^{-1}, \cdots, 2^{B-1}\right]^T, \tag{15}$$

where, $b_i(k)$ is an i-th bit of the input signal s(k). In Eq.(13), $A^T(k)W(k)$ is defined as

$$\mathbf{P}(k) \equiv \mathbf{A}^T(k)\mathbf{W}(k), \tag{16}$$

and the filter output is obtained as

$$\mathrm{y(k)} = \mathbf{F}^{\mathrm{T}}\mathbf{P}(\mathrm{k}). \tag{17}$$

The P(k) is called adaptive function space (AFS), and is a B-th order vector of

$$\mathbf{P}(k)=\left[p_0(k), p_1(k), \cdots, p_{B-1}(k)\right]^T. \tag{18}$$

The P(k) is a subset of the WAFS including the elements specified by the row vectors (access vectors) of the address matrix. Now, multiplying both sides by $A^T(k)$, Eq.(4) becomes

$$\begin{aligned} \mathbf{A}^T(k)\mathbf{W}(k+1) &= \mathbf{A}^T(k)\left\{\mathbf{W}(k)+2\mu e(k)\mathbf{A}(k)\mathbf{F}\right\} \\ &= \mathbf{A}^T(k)\mathbf{W}(k)+2\mu e(k)\mathbf{A}^T(k)\mathbf{A}(k)\mathbf{F}. \end{aligned} \tag{19}$$

Furthermore, by using the definitions described as

$$\mathbf{P}(k) \equiv \mathbf{A}^T(k)\mathbf{W}(k)$$

and

$$\mathbf{P}(\mathrm{k}+1) \equiv \mathbf{A}^{\mathrm{T}}(\mathrm{k})\mathbf{W}(\mathrm{k}+1), \tag{20}$$

the relation of them can be explained as

$$\mathbf{P}(k+1)=\mathbf{P}(k)+2\mu e(k)\mathbf{A}^T(k)\mathbf{A}(k)\mathbf{F}. \tag{21}$$

It is impossible to perform the real-time processing because of the matrix multiplication of $A^T(k)A(k)$ in Eq.(21).

To overcome this problem, the simplification of the term of $A^T(k)A(k)F$ in Eq.(21) has been also achieved on the 2's complement representation (Tsunekawa et al, 1999). By using the relation

$$E\left[\mathbf{A}^T(k)\mathbf{A}(k)\mathbf{F}\right] = 0.25N\mathbf{F}\,, \tag{22}$$

the simplified algorithm becomes

$$\mathbf{P}(k+1) = \mathbf{P}(k) + 0.5\mu Ne(k)\mathbf{F}\,, \tag{23}$$

where, the operator E[ ] denotes an expectation. Eq.(23) can be performed by only shift and addition operations without multiplications using approximated μN with power of two, that is, the fast real-time processing can be realized. The fundamental structure is shown in Fig.3, and the timing chart is also shown in Fig.4. The calculation block can be used the fundamental structure of the DA-FIR, and the WAFS is realized by using a Random Access Memory (RAM).

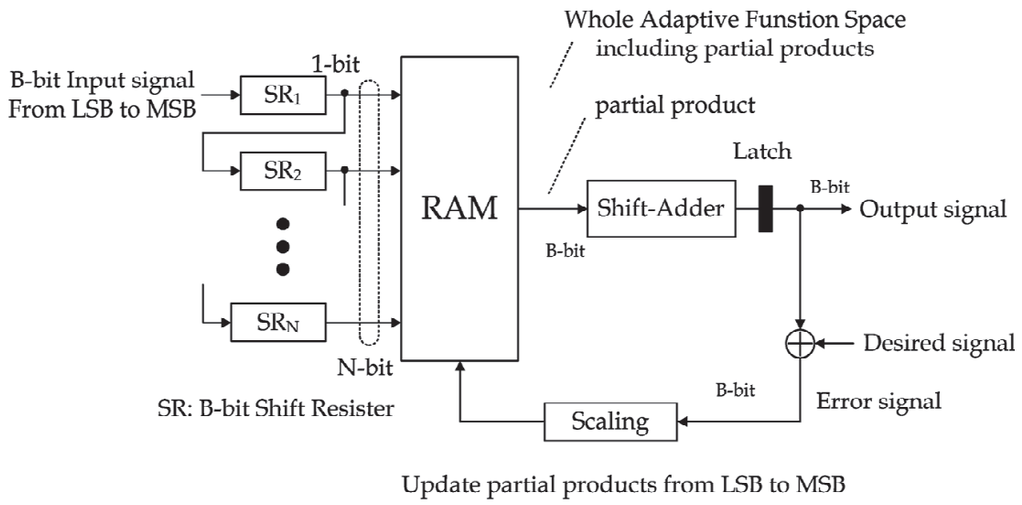


Fig. 3. Fundamental structure of the DA-ADF.

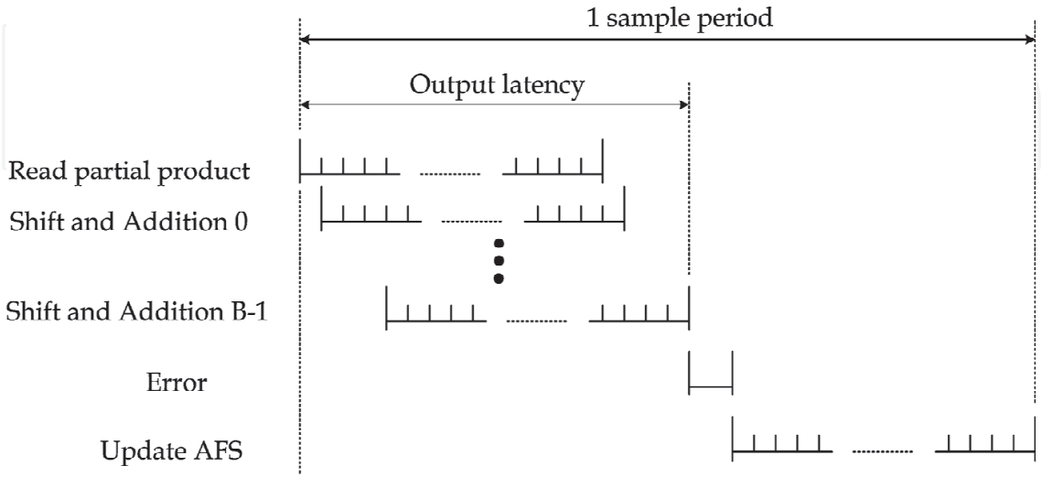


Fig. 4. Timing chart of the DA-ADF.

3.3 Multi-memory block structure

The structure employing the DWAFS to guarantee the convergence speed and the small

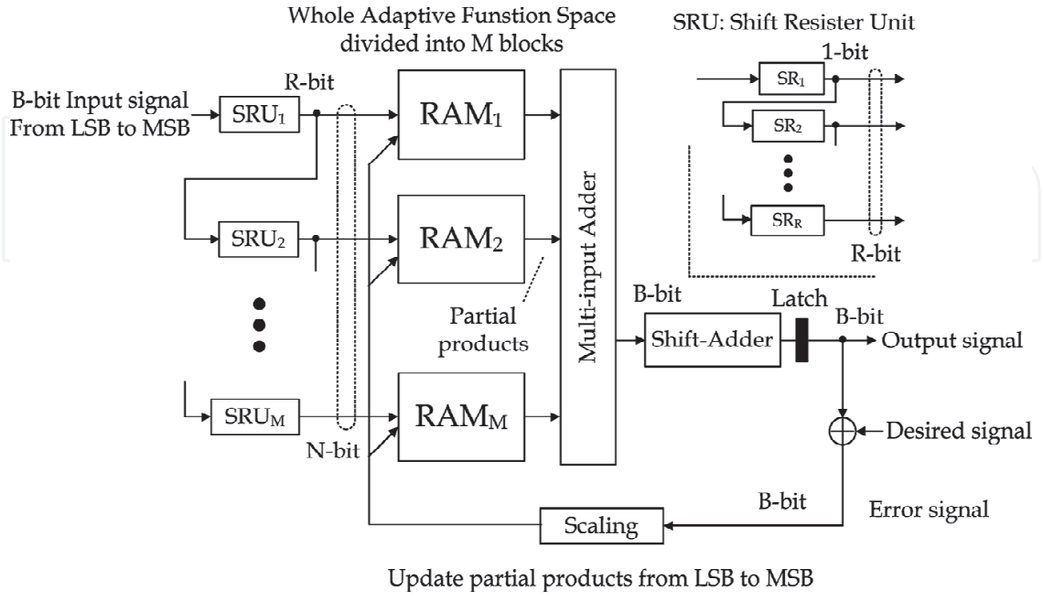


Fig. 5. Fundamental structure of the MDA-ADF.

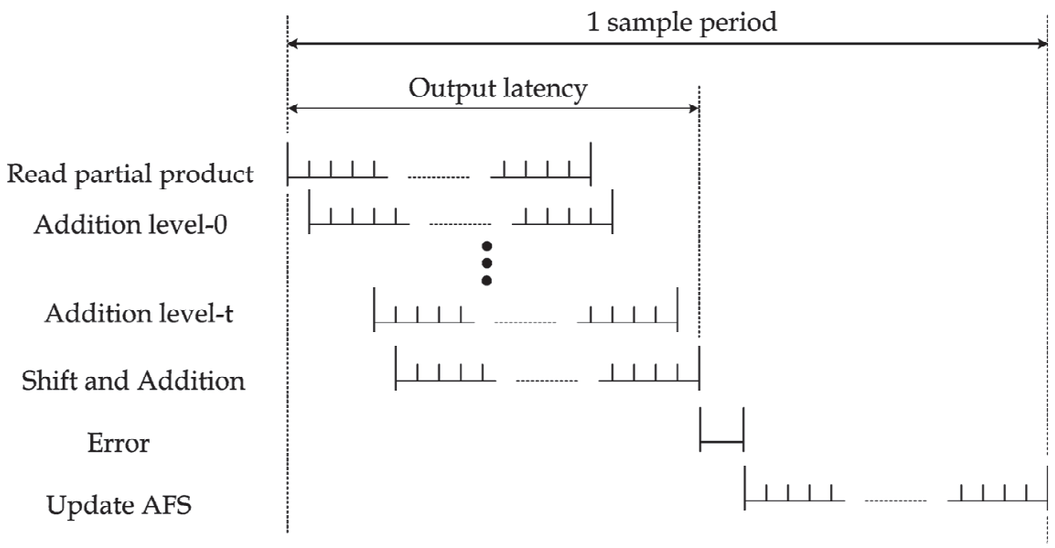


Fig. 6. Timing chart of the MDA-ADF.

hardware for higher order filtering has been proposed (Wei & Lou, 1986; Tsunakawa et al, 1999). The DWAFS is defined for the divided address-line of R bit. For a division number M, the relation of N and R is represented as

$$R = N / M \,.$$

The capacity of the individual DWAFS is 2^R words, and the total capacity becomes smaller $2^R M$ words than the DA's 2^N words. For the smaller WAFS, the convergence of the algorithm can be achieved by smaller iterations. The R-th order coefficient vector and the related AFS is represented as

$$\mathbf{W}_m(k) = \left[w_{m0}(k), w_{m1}(k), \cdots, w_{m(R-1)}(k) \right]^T, \tag{24}$$

$$\mathbf{P}_m(k) = \left[p_{m0}(k), p_{m1}(k), \cdots, p_{m(B-1)}(k) \right]^T, \tag{25}$$

$$(m = 0,1,\cdots,M-1\,;\, R = N/M),$$

where, the AFS is defined as

$$\mathbf{P}_m(k) \equiv \mathbf{A}_m^T(k)\mathbf{W}_m(k)\,. \tag{26}$$

Therefore, the filter output signal is obtained by

$$y(k) = \sum_{m=1}^{M} F^T \mathbf{P}_m(k)\,, \tag{27}$$

where, the address matrix is represented as

$$\mathbf{A}_m(k) = \begin{bmatrix} b_{m0}(k) & b_{m0}(k-1) & \cdots & b_{m0}(k-R+1) \\ b_{m1}(k) & b_{m1}(k-1) & \cdots & b_{m1}(k-R+1) \\ \vdots & \vdots & \ddots & \vdots \\ b_{m(B-1)}(k) & b_{m(B-1)}(k-1) & \cdots & b_{m(B-1)}(k-R+1) \end{bmatrix}^T. \tag{28}$$

The update formula of the MDA-ADF is represented as

$$\mathbf{P}_m(k+1) = \mathbf{P}_m(k) + 0.5\mu\,\mathrm{Re}(k)\mathbf{F}\,. \tag{29}$$

The fundamental structure and the timing chart are shown in Fig.5 and 6, respectively.

3.4 Half-memory algorithm using pseudo-odd symmetry property

It is known that there exists the odd symmetry property of the WAFS in the conventional DA-ADF on the offset binary coding (Cowan et al, 1983). Table 1 shows the example of the odd symmetry property in case of R=3. The stored partial-product for the inverted address has an equal absolute values and a different sign. Using this property, the MDA-ADF is can be realized with half amount of capacity of the DWAFS. This property concerned with a property of the offset binary coding. However, the pseudo-odd symmetry property of WAFS on the 2's complement representation has been found (Takahashi et al, 2001). The stored partial-product for the inverted address is a nearly equal absolute value and a different sign. The MDA algorithm using this property is called half-memory algorithm, and the previous discussed MDA algorithm is called full-memory algorithm. The access method of the DWAFS is represented as follows (Takahashi et al, 2001).

Address	Stored partial-product
000	$-0.5w_0 -0.5w_1-0.5w_2$
001	$-0.5w_0 -0.5w_1+0.5w_2$
010	$-0.5w_0 +0.5w_1-0.5w_2$
011	$-0.5w_0 +0.5w_1+0.5w_2$
100	$+0.5w_0 -0.5w_1-0.5w_2$
101	$+0.5w_0 -0.5w_1+0.5w_2$
110	$+0.5w_0 +0.5w_1-0.5w_2$
111	$+0.5w_0 +0.5w_1+0.5w_2$

Table 1. Example of the odd-symmetry property of the WAFS on the offset binary coding. This property is approximately achieved on the 2's complement representation.

Read the partial-products

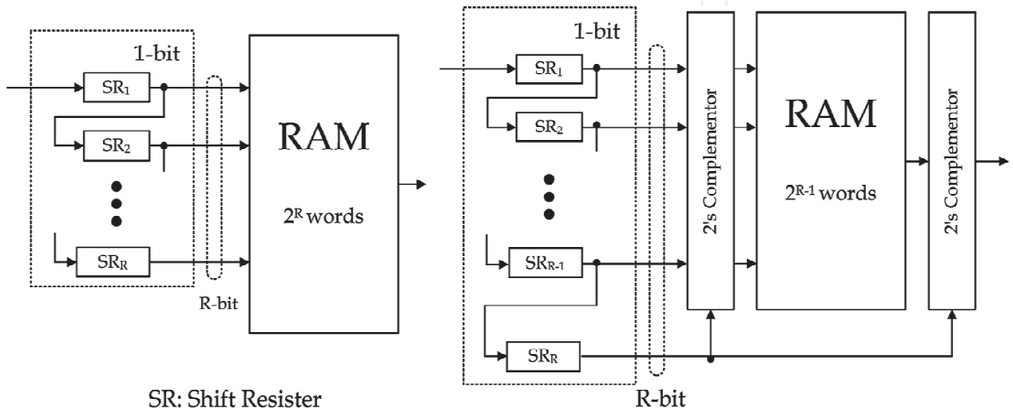
```
begin
  for i:=1 to B do
  begin
     if address_MSB = 0 then
        Read the partial-product using R-1 bits address;
     if address_MSB = 1 then
        Invert the R-1 bits of address;
        Read the partial products using inverted R-1 bits address;
        Obtain the negative read value;
  end
end
```

Update the WAFS

```
begin
  for i:=1 to B do
  begin
     if address_MSB = 0 then
        Add the partial-product and update value;
     if address_MSB = 1 then
        Invert the R-1 bits of address;
        Obtain the negative update value;
        Add the partial-product and the negative update value;
  end
end
```

The expression of "addressMSB" indicates the MSB of the address. Fig.7 shows the difference of the access method between MDA and HMDA. The HMDA accesses the WAFS with the R-1 bits address-line without MSB, and the MSB is used to activate the 2's complementors located both sides of the WAFS. Fig. 8 shows the comparison of the convergence properties of the LMS, MDA, and HMDA. Results are obtained by the computer simulations. The simulation conditions are shown in Table 2. Here, IRER

represents an impulse response error ratio. The step-size parameters of the algorithms were adjusted so as to achieve a final IRER of -49.5 [dB]. It is found that both the MDA(R=1) and the HMDA(R=2) achieve a good convergence properties that is equivalent to the LMS's one. Since both the MDA(R=1) and the HMDA(R=2) access the DWAFS with 1 bit address, the DWAFS is the smallest size and defined for every tap of the LMS. The convergence speed of the MDA is degraded by increasing R (Tsunekawa et al, 1999). This means that larger capacity of the DWAFS needs larger iteration for the convergence. Because of smaller capacity, the convergence speed of the HMDA(R=4) is faster than the MDA(R=4)'s one (Takahashi et al, 2001). The HMDA can improve the convergence speed and reduce the capacity of the WAFS, i.e., amount of hardware, simultaneously.



(a) Access method of MDA (b) Access method of HMDA

Fig. 7. Comparison of the access method for the WAFS. (a) Full-memory algorithm (b) Half-memory algorithm.

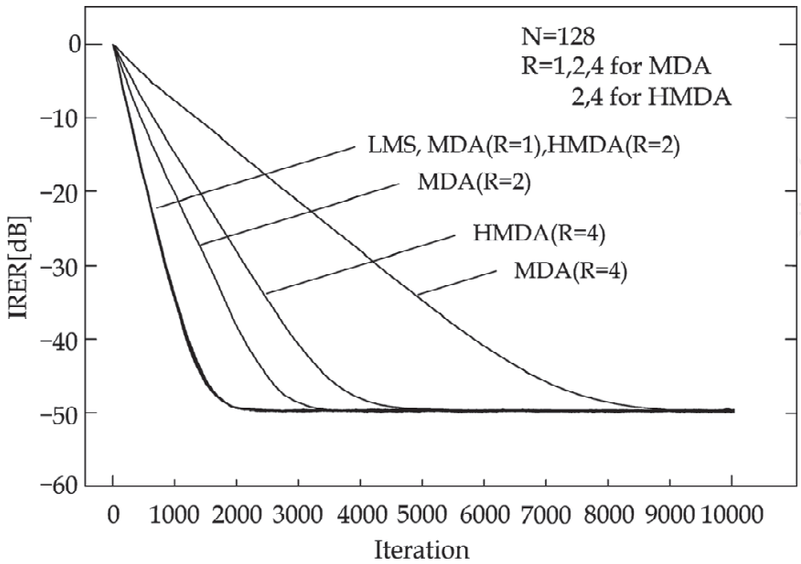


Fig. 8. Comparison of the Convergence properties.

Simulation Model	System identification problem
Unknown system	128 taps low-pass FIR filter
Method	LMS, MDA, and HMDA
Measurement	Impulse Response Error Ratio(IRER)
Number of taps	128
Number of address-line	1, 2, 4 for MDA, 2 and 4 for HMDA
Input signal	White Gaussian noise, variance=1.0, average=0.0
Observation noise	White Gaussian noise independent to the input signal, 45dB

Table 2. Computer simulation conditions.

4. New algorithm and architecture

The new algorithm and effective architecture can be obtained by applying the following techniques and ideas. 1) In the DA algorithm based on 2's complement representation, the pseudo-odd symmetry property of WAFS is applied to the new algorithm and architecture from different point of view on previously proposed half-memory algorithm. 2) A pipelined structure with separated output calculation and update procedure is applied. 3) The delayed update method (Long, G. Et al, 1989, 1992; Meyer & Agrawal, 1993; Wang, 1994) is applied. 4) To reduce the pitch of pipeline, two partial-products are pre-loaded before addition in update procedure. 5) The multi-memory block structure is applied to reduce an amount of hardware for higher order. 6) The output calculation procedure is performed from LSB to MSB, whereas, the update procedure is performed with reverse direction.

4.1 New algorithm with delayed coefficient adaptation

To achieve a high-speed processing, the parallel computing of the output calculation and the update in the MDA-ADF is considered. It is well known that the delayed update method enables the parallel computation in the LMS-ADF permitting the degradation of the convergence speed. This method updates the coefficients using previous error signal and input signal vector in the LMS-ADF.

Now, let's apply this method to the MDA-ADF. In the MDA and the HMDA, both of the output calculation and the update are performed from LSB to MSB. However, in this new algorithm, the output calculation procedure is performed from LSB to MSB, whereas, the update procedure is performed with reverse direction. Here, four combinations of the direction for the two procedures exist. However, it is confirmed by the computer simulations that the combination mentioned above is the best for the convergence property when the large step-size parameter close to the upper bound is selected. Examples of the convergence properties for the four combinations are shown in Fig.9. In these simulations, the step-size of 0.017 is used for the (a) and (c), and 0.051 is used for the (b) and (d) to achieve a final IRER of -38.9 [dB]. Both the (b) and (d) have a good convergence properties that is equivalent to the LMS's one, whereas, the convergence speed of the (a) and (c) is degraded. This implies that the upper bound of the (a) and (c) becomes lower than the (b) , (d) and LMS's one.

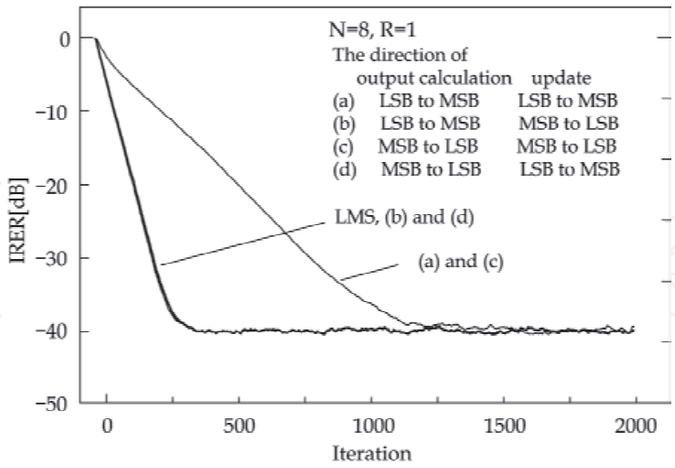


Fig. 9. Comparison of the convergence characteristics for the different combinations of the direction on the output calculation and update. The step-size of 0.017 is used for the (a) and (c), and 0.051 is used for the (b) and (d).

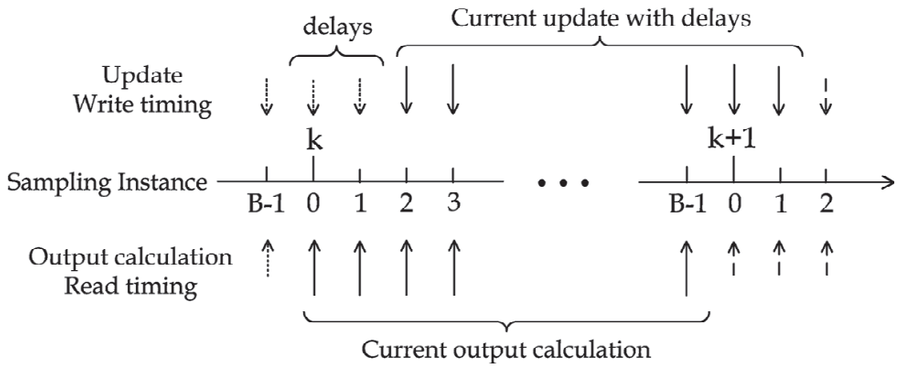


Fig. 10. Relation of the timing between read and write of the DWAFS.

In the HMDA-ADF, the activation of the 2's complementor is an exceptional processing for the algorithm, that is, the processing time increases. The new algorithm is performed without the activation of the 2's complementor by use of the pseudo-odd symmetry property. This is realized by using the address having inverted MSB instead of the 2's complementor. This new algorithm is called a simultaneous update algorithm, and the MDA-ADF using this algorithm is called SMDA-ADF. Fig. 10 shows the timing of the read and write of the DWAFS. The partial-product is read after writing the updated partial-products.

The SMDA algorithm is represented as follows. The filter output is obtained as the same manner in the MDA-ADF. The m-th output of the m-th DWAFS is

$$y_m(k) = \sum_{i=0}^{B-1} \mathbf{F}'^{T}_{B-1-i} \mathbf{P}_{m,B-1-i}(k) . \tag{30}$$

The output signal is the sum of these M outputs, and this can be expressed as

$$y(k)=\sum_{m=1}^{M} y_m(k)\,. \tag{31}$$

The scaling vectors are

$$\mathbf{F}_0^{'}(k)=\left[2^0,0,\cdots,0\right]^T,\ \mathbf{F}_1^{'}(k)=\left[0,2^{-1},\cdots,0\right]^T,\cdots,\ \mathbf{F}_{B-1}^{'}(k)=\left[0,0,\cdots,2^{-B+1}\right]^T. \tag{32}$$

The address matrix including the inverted MSB for the output calculation is represented as

$$\mathbf{A}_m^{out}(k)=\begin{bmatrix} \overline{b}_{m0}(k) & \overline{b}_{m0}(k-1) & \cdots & \overline{b}_{m0}(k-R+1) \\ b_{m1}(k) & b_{m1}(k-1) & \cdots & b_{m1}(k-R+1) \\ \vdots & \vdots & \ddots & \vdots \\ b_{m(B-1)}(k) & b_{m(B-1)}(k-1) & \cdots & b_{m(B-1)}(k-R+1) \end{bmatrix}^T. \tag{33}$$

This algorithm updates the two partial-products according to the address and its inverted address, simultaneously. When the delays in Fig.10 is expressed by d, the address matrix for the update procedure is represented as

$$\mathbf{A}_m^{up}(k)=\begin{bmatrix} \overline{b}_{m0}(k) & \overline{b}_{m0}(k-1) & & \overline{b}_{m0}(k-R+1) \\ \vdots & \vdots & \cdots & \vdots \\ b_{m(B-1-d)}(k) & b_{m(B-1-d)}(k-1) & & b_{m(B-1-d)}(k-R+1) \\ b_{m(B-d)}(k-1) & b_{m(B-d)}(k-2) & \ddots & b_{m(B-d)}(k-R) \\ \vdots & \vdots & & \vdots \\ b_{m(B-1)}(k-1) & b_{m(B-1)}(k-2) & \cdots & b_{m(B-1)}(k-R) \end{bmatrix}^T. \tag{34}$$

The update formulas are

$$\mathbf{P}_{m,i}(k+1)=\mathbf{P}_{m,i}(k)+0.5\mu\,\mathrm{Re}(k-1)\mathbf{F}_i^{'}, \tag{35}$$

$$\overline{\mathbf{P}}_{m,i}(k+1)=\overline{\mathbf{P}}_{m,i}(k)-0.5\mu\,\mathrm{Re}(k-1)\mathbf{F}_i^{'}, \tag{36}$$

$$(\mathrm{m}=1,2,\cdots,\mathrm{M};\ \mathrm{i}=0,1,\cdots,\mathrm{B}-\mathrm{d}),$$

and

$$\mathbf{P}_{\mathrm{m,i}}(\mathrm{k}+1)=\mathbf{P}_{\mathrm{m,i}}(\mathrm{k})+0.5\mu\,\mathrm{Re}(\mathrm{k}-2)\mathbf{F}_{\mathrm{i}}^{'}, \tag{37}$$

$$\overline{\mathbf{P}}_{\mathrm{m,i}}(\mathrm{k}+1)=\overline{\mathbf{P}}_{\mathrm{m,i}}(\mathrm{k})-0.5\mu\,\mathrm{Re}(\mathrm{k}-2)\mathbf{F}_{\mathrm{i}}^{'}, \tag{38}$$

$$(\mathrm{m}=1,2,\cdots,\mathrm{M};\ \mathrm{i}=\mathrm{B}-\mathrm{d}-1,\cdots,\mathrm{B}-1).$$

The error signal is obtained by

$$e(k) = d(k) - y(k)$$

In Eq.(36) and Eq.(38), $\bar{\mathbf{P}}_{m,i}(k)$ is the AFS specified by the inverted addresses.

4.2 Evaluation of convergence properties

The convergence properties are evaluated by the computer simulations. Table 3 shows the simulation conditions, and Fig.11 shows the simulation results. The step-size parameters of the algorithms were adjusted so as to achieve a final IRER of -49.8 [dB]. The SMDA and the HMDA (Takahashi et al, 2001) with R=2 achieve a good convergence properties that is equivalent to the LMS's one. The convergence speed of the DLMS (LMS with 1 sample delayed update) degrades against the LMS's one because of the delayed update with 1 sample delay, whereas, in spite of the delayed update with 1 and 2 sample delays, the SMDA with R=2 can achieve a fast convergence speed.

4.3 Architecture

Fig.12 shows the block diagram of the SMDA-ADF. Examples of the sub-blocks are shown in Fig.13, Fig.14, and Fig.15. In Fig.12, the input signal register includes (2N+1)B shift-registers. The address matrix is provided to the DWAFS Module (DWAFSM) from the input register. The sum of the M-outputs obtained from M-DWAFSM is fed to the Shift-Adder.

After the shift and addition in B times, the filter output signal is obtained. The obtained two error signals, the e(k-1) and the - e(k-1), are scaled during reading the partial-products to be updated. In Fig.13, the DWAFSM includes the 2^R+2 B-bit register, 1 R-bit register, 2 decoders, 5 selectors, and 2 adders. The decoder provides the select signal to the selectors. The two elements of DWAFS are updated, simultaneously. Fig.16 shows the timing chart of the SMDA-ADF. The parallel computation of the output calculation and update procedure are realized by the delayed update method.

Simulation Model	System identification problem
Unknown system	128 taps low-pass FIR filter
Method	LMS, DLMS, SMDA , and HMDA
Measurement	Impulse Response Error Ratio(IRER)
Number of taps	128
Number of address-line	2 and 4 for DA method
Input signal	White Gaussian noise, variance=1.0, average=0.0
Observation noise	White Gaussian noise independent to the input signal, 45dB

Table 3. Simulation conditions.

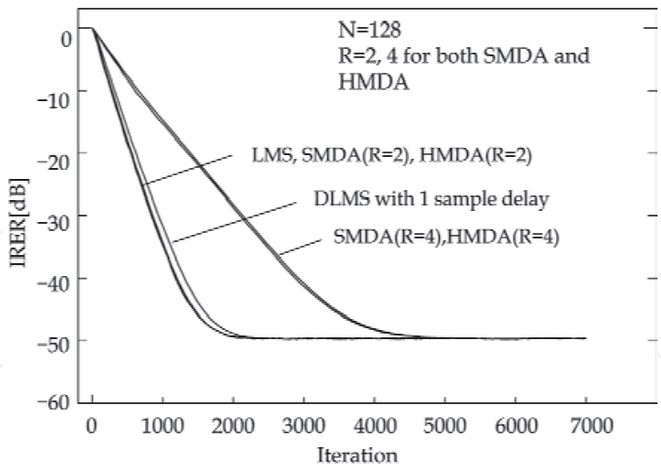


Fig. 11. Comparison of the convergence properties.

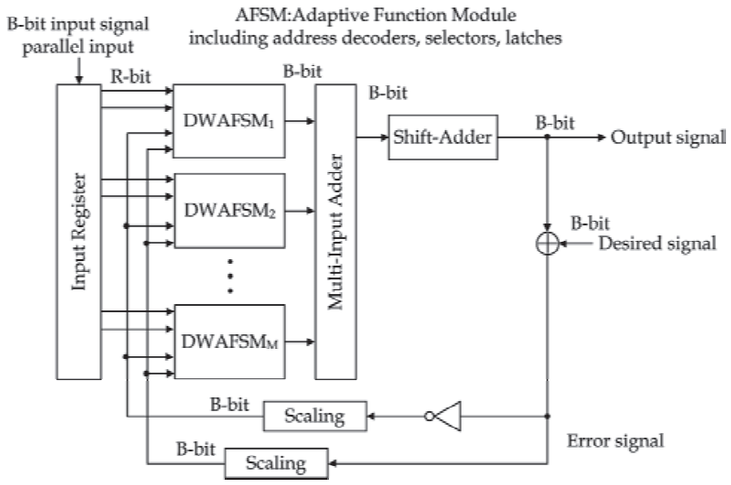


Fig. 12. Block Diagram of the SMDA-ADF.

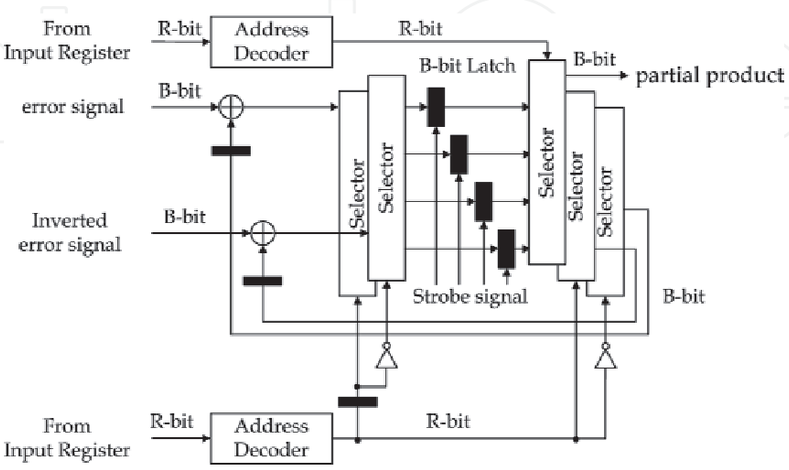


Fig. 13. Example of the DWAFS module for R=2.

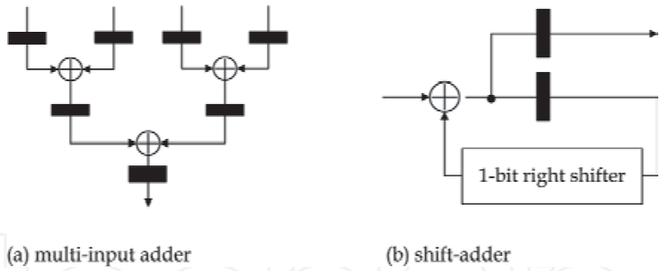


Fig. 14. Example of the multi-input register and shift-adder.

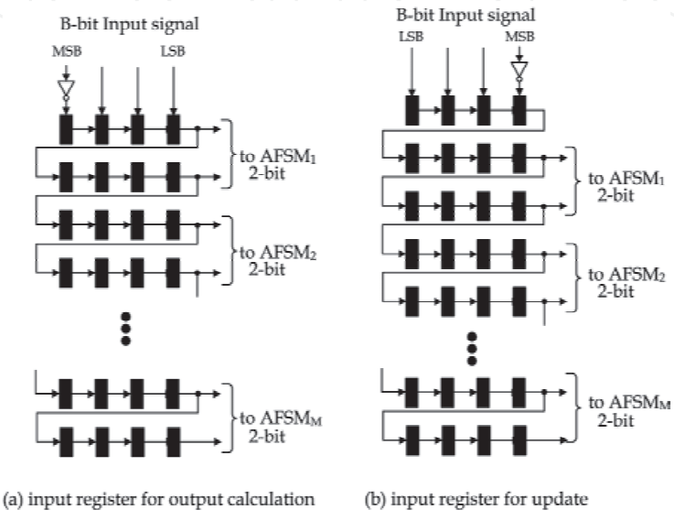


Fig. 15. Example of the input registers for B=4 and R=2. The (a) is for output calculation, and the (b) is for update.

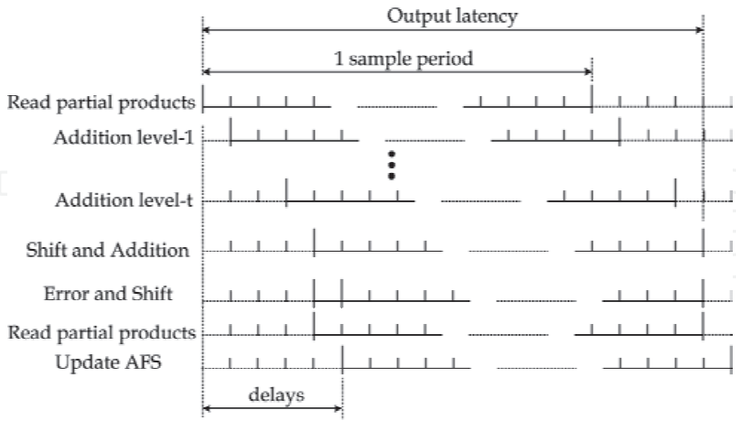


Fig. 16. Timing chart of the SMDA-ADF. The sampling period is equal to the word length of the input signal. The current update procedure begins after delays.

4.3 VLSI evaluations

The VLSI architecture of the SMDA is evaluated with comparison to the previous proposed methods using the multipliers, the MDA (Tsunakawa et al, 1999), conventional LMS, pipelined

DLMS (Meyer & Agrawal, 1993), pipelined LMS structure (Harada et al, 1998), and pipelined NCLMS (Takahashi et al, 2006). Table 4 shows the evaluation conditions. The result for the SMDA and MDA are shown in Table 5, and the others using the multipliers are shown in Table 6. These results were obtained by a VLSI design system PARTHENON (NTT DATA Corporation, 1990). It is found that the SMDA can achieve the high-sampling rate of 380% and small output latency of 67% against the MDA, whereas, the power dissipation and the area are increased. However, the improvement of the sampling rate and latency exceed the degradation of the power dissipation and the area. The methods in Table 6 need both of the very large amount of gates and the area against the SMDA. From these results, it is found that the SMDA has advantages of small amount of hardware, a sampling rate close to the LMS.

Methods	LMS, Pipelined-DLMS, Pipelined-LMS, Pipelined-NCLMS, SMDA
Number of taps	128
Word length	16 bit
Division number	64
VLSI library	0.8 micron CMOS standard cell, 5V
Adder	Carry look-ahead adder
Multiplier	Booth's encode algorithm, Wallace tree, Carry look-ahead adder

Table 4. The condition of the VLSI evaluation.

	MDA	SMDA
Machine cycle [ns]	31	21
Sampling rate [MHz]	0.79	3.00
Latency [ns]	713	479
Power dissipation [W]	6.40	16.47
Area [mm^2]	36	54
Number of gates	175,321	258,321

Table 5. Comparison of the VLSI evaluations for the MDA and the SMDA.

	LMS	Pipe-DLMS	Pipe-LMS	Pipe-NCLMS
Machine cycle [ns]	297	63	131	61
Sampling rate [MHz]	3.37	15.87	7.63	16.39
Latency [ns]	214	8,064	131	61
Power dissipation [W]	6.23	25.79	27.33	18.20
Area [mm^2]	297	205	429	187
Number of gates	1,570,084	997,760	2,082,304	916,893

Table 6. Comparison of the VLSI evaluations for ADFs employing multipliers.

5. Conclusions

In this chapter, the new LMS algorithm using distributed arithmetic and its VLSI architecture have been presented. According the discussions, we conclude as follows:

1. The SMDA-ADF can achieve the good convergence speed, higher sampling rate and small output latency than the conventional MDA-ADF.

2. The small amount of hardware is the feature of the SMDA-ADF against the ADFs employing the multipliers.
3. In spite of the delayed adaptation, the convergence speed is equivalent to the LMS's one.
4. The convergence property depends on the combination of the direction of the output and update procedure. The output calculation from LSB to MSB and the update procedure with reverse direction is the best, when the step-size parameter is close to the upper bound.

6. References

Widrow, B., Glover,J.R., McCool, J.M., Kaunitz, J., Williams, C.S., Hearn, R.H., Zeidler, J.R., Dong, E. & Goodlin, R.C. (1975). Adaptive noise cancelling: Principles and applications. Proc. IEEE, vol.63, pp.1692-1716

Makino, S. & Koizumi, N. (1988). Improvement on Adaptation of an Echo Canceller in a Room. IEICE Letter Fundamentals, Vol. J 71-A, No.12, pp.2212-2214

Tsunekawa,Y., Takahashi,K., Toyoda,S. & Miura, M. (1999). High-performance VLSI Architecture of Multiplier-less LMS Adaptive Filter Using Distributed Arithmetic. IEICE Trans. Fundamentals, Vol.J82-A, No.10, pp.1518-1528

Takahashi,K., Tsunekawa,Y., Toyoda,S. & Miura, M. (2001). High-performance Architecture of LMS Adaptive Filter Using Distributed Arithmetic Based on Half-Memory Algorithm. IEICE Trans. Fundamentals, Vol.J84-A, No.6, pp.777-787

Takahashi, K., Tsunekawa, Y., Tayama, N., Seki, K. (2002). Analysis of the Convergence Condition of LMS Adaptive Digital Filter Using Distributed Arithmetic. IEICE Trans. Fundamentals, Vol.E85-A, No.6, pp1249-1256

Takahashi, K., Kanno, D. & Tsunekawa, Y. (2006). High-Performance Pipelined Architecture of Adaptive Digital Filter Employing FIR Filter with Minimum Coefficients Delay and Output Latency. IEICE Trans. Fundamentals, Vol.J89-A, No.12, pp.1130-1141

Cowan, C.F.N. & Mavor., J. (1981). New digital adaptive-filter implementation using distributed-arithmetic techniques. IEE Proc., vol.128, Pt.F, no.4, pp.225--230

Cowan, C.F.N, Smith, S.G. & Elliott, J.H. (1983). 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

Peled, A. & Liu, B. (1974). A new hardware realization of digital filters. IEEE Trans. Acoust., Speech \& Signal Process., vol.22, no.12, pp.456--462

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

Long, G., Ling, F. & Proakis, J.G. (1989). The LMS algorithm with delayed coefficient adaptation. IEEE Trans. Acoust. Speech Signal Process., vol.37, no.9, pp.1397-1405

Long, G., Ling, F. & Proakis, J.G. (1992). Corrections to "The LMS algorithm with delayed coefficient adaptation". IEEE Trans. Acoust. Speech Signal Process., vol.40, no.1, pp.230-232

Meyer, M.D. & Agrawal, D.P. (1993). A high sampling rate delayed LMS filter architecture. IEEE Trans. Circuits \& Syst. II, vol.40, no.11, pp.727-729

Wang, C.L. (1994). Bit-serial VLSI implementation of delayed LMS transversal adaptive filters. IEEE Trans. Signal Processing, vol.42, no.8, pp.2169--2175

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

NTT DATA Corporation (1990). PARTHENON User's Manual. Japan.

Adaptive Filtering
Edited by Dr Lino Garcia

ISBN 978-953-307-158-9
Hard cover, 398 pages
Publisher InTech
Published online 06, September, 2011
Published in print edition September, 2011

Adaptive filtering is useful in any application where the signals or the modeled system vary over time. The configuration of the system and, in particular, the position where the adaptive processor is placed generate different areas or application fields such as prediction, system identification and modeling, equalization, cancellation of interference, etc., which are very important in many disciplines such as control systems, communications, signal processing, acoustics, voice, sound and image, etc. The book consists of noise and echo cancellation, medical applications, communications systems and others hardly joined by their heterogeneity. Each application is a case study with rigor that shows weakness/strength of the method used, assesses its suitability and suggests new forms and areas of use. The problems are becoming increasingly complex and applications must be adapted to solve them. The adaptive filters have proven to be useful in these environments of multiple input/output, variant-time behaviors, and long and complex transfer functions effectively, but fundamentally they still have to evolve. This book is a demonstration of this and a small illustration of everything that is to come.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen