# New Realization and Implementation of IIR Digital Filters Using Residue Feedback

A. Tawfik, P. Agathoklis and F. El-Guibaly

Dept. of Electrical & Computer Engineering
University of Victoria, Victoria, CANADA

## Abstract

*A new low-roundoff realization for narrow-band IIR digital filter, which uses residue feedback technique, is presented. The structure has a saving of $N(N-2)/2$ multiplies over the optimal structure proposed by Mullis, Roberts and Hwang [4-5], as well as, the optimal residue feedback structure proposed by Williamson [1]. The output roundoff noise for the proposed structure is slightly higher than that of the structure proposed by Williamson. Further, the proposed structure has a block-triangular state-update matrix which is suitable for high-speed hardware implementation. A VLSI array-processor implementation of the proposed IIR structure in which the computation rate is maximized is also presented.*

## 1 Introduction

A general method that has been used to reduce the error inherent in any quantization operation is the residue feedback (RF) [1] (also called error spectral shaping [2-3]). The RF technique is implemented by extracting the quantization error (residue) due to product quantization. This residue is sometimes weighted before it is fed back for use in the next iteration. In general, weighing the residue requires extra multiplication operations [4-5] which leads to increase in the required computational complexity. One powerful solution which eliminates the need for RF multiplications or shift operations is reported in [1] in which the residue coefficients are restricted to $\pm 1$ values. This RF scheme leads to structures with low coefficient sensitivity which may provide better output roundoff noise than that of the optimal structures of [4-5] which do not use RF techniques at the price of $N$ extra additions. Unfortunately, these RF optimal structures have full matrices which adversely impacts both the hardware area and the computation speed of any implementation.

In this paper, a new realization of IIR digital filters with reduced number of coefficients is presented. The proposed low-noise realization is based on the RF technique reported in [1]. It provides slightly higher output roundoff noise than that of [1] while saving a large number of multiplications and additions. Efficient array-processor implementation for the proposed realization is also presented.

This work was supported by grants from NSERC, Micronet and U.Vic. Faculty Research.

## 2 Preliminaries For Residue Feedback

A technique known as integer residue feedback [1] will be used here. The diagram of the IIR structure with integer residue feedback is shown in Fig. 1. The state-space equations that describe Fig. 1 can be written as [1]

$$\tilde{x}(n+1) = AQ[\tilde{x}(n)] + Je(n) + Bu(n) \quad (1)$$

$$\tilde{y}(n) = CQ[\tilde{x}(n)] + Du(n) \quad (2)$$

The quantization operation $Q[.]$ in (1) and (2) rounds the states after the multiplications and additions are completed in double-precision, i.e., $\hat{x}(n) = Q[\tilde{x}(n)]$ and $e(n) = \tilde{x}(n) - \hat{x}(n)$ where $\hat{x}(n)$ is the quantized version of $\tilde{x}(n)$. All coefficients of the state-space matrices $A,B,C,D$ are assumed to have an exact representation. Fixed-point arithmetic is implemented using a two's complement representation. Matrix $J$ in (1) will be restricted to be equal *to* $I$ (for LPF) or $-I$ (for HPF) where $I$ is the identity matrix. By restricting $J$ to take these specific values, the RF scheme in Fig. 1 does not require any extra multiplications and it requires only $N$ extra additions. The roundoff residue error $e(n)$ can be modelled as zero-mean noise process with covariance $q^2 I$ where $q^2 = 2^{-2n_1}/12$ and $n_1$ is the wordlength of the states after quantization. $l_2$-scaling which guarantees equal probability of overflow for all states is equivalent to imposing the constraint

$$k_{ii} = 1 \text{ for all } i \quad (3)$$

where $k_{ii}$ is the $i$th diagonal element of the covariance matrix $K$ defined in [6]. The output noise variance $NP$ of the structure in Fig. 1 can be expressed as

$$NP = q^2(1+g^2) \quad (4)$$

where the noise gain $g^2$ can be calculated as

$$g^2 = tr(P) \quad (5)$$

$tr$ is the trace of a matrix and $P$ is the residue matrix defined as [1]

$$P = (I \pm A)^t W + W(I \pm A) \quad (6)$$

The notation "$t$" refers to the transpose of a matrix. The "+" sign in (6) corresponds to $J = -I$, "- "sign corresponds to $J = I$, and $W$ is the noise-matrix defined in [6].

Under any similarity transformation $z = T^{-1}x$, any initial realization can be transformed to another realization which performs differently under finite wordlength effects. The new realization matrices $(A_i, B_i, C_i, D)$, Covariance matrix $K_T$, noise matrix $W_T$ and the residue matrix $P_T$ satisfy

$$A_T = T^{-1}A_iT \qquad B_T = T^{-1}B_i \qquad C_T = C_iT$$

$$K_T = (T^{-1})K_i(T^{-1})^t \qquad W_T = T^tW_iT \qquad P_T = T^tP_iT.$$

The synthesizing problem can now be formulated as follows: starting from any initial realization with block-triangular update-state matrix, it is required to find a similarity transformation $T$ which minimizes the noise gain in (5) under the constraint in (3) provided that the new update-state matrix $A_T$ remains block-triangular. This similarity transformation matrix $T$ can be obtained by applying a minimization algorithm as discussed in the following section.

## 3 The New Realization

In this section, a technique to obtain the similarity transformation $T$ is given and is based on a modified version of the minimization algorithm reported in [7]. The minimization algorithm performs the minimization process iteratively by forming a transformation matrix $T$ as a product of simple upper (or lower) triangular matrices. i.e., [7]

$$T = T(0)T(1)T(2)\ldots\ldots \qquad (7)$$

where each individual transformation matrix $T(l)$ is chosen to retain the $l_2$-norm scaling condition in (3) and in the same time to reduce the roundoff noise gain in (5). By insisting that each individual transformation matrix be upper (or lower) triangular, the overall transformation matrix will be upper (or lower) triangular matrix.

The realizations obtained by applying the aforementioned minimization algorithm may differ depending on the initial structure. These different realizations perform, in general, differently under finite wordlength implementation. Realizations with block-triangular update matrices (which are suitable for high-speed VLSI implementations) can be obtained by adopting the cascade (or parallel) combinations of first- and second-order sections structure as the initial structure for the minimization algorithm. The new structures proposed in this paper are obtained by applying the minimization algorithm to an initial structure consisting of a cascade combination of second-order optimal (in the sense of [4-5]) sections. This initial structure can easily be obtained as shown in [6]. The initial structure is neither required to be optimized in the sense of zero-pole pairing or section ordering nor to be properly $l_2$-scaled.

If the realization of section direct form has been chosen as an initial structure, another realization with reduced number of coefficients and with excellent performance under finite wordlength effects can be obtained [8]. However, the update state matrix of this realization has the form of Hessenberg matrix which is not suitable for high-speed VLSI implementations.

## 4 Roundoff Noise, Coefficient Sensitivity and Computational Complexity Comparison

For the sake of comparison, three sixth-order filters which included an elliptic low-pass filter (LPF), an elliptic high-pass filter (HPF) and a chebyshev high-pass filter (HPF) are realized by four different low-roundoff realizations including the one proposed in this paper. The specifications of the three filters are listed in Table 1. The computational complexity in terms of number of multiplications and additions of the four low-roundoff realizations are given in Table 2 and the resulting output noise gains are listed in Table 3. It can be seen from Table 2 and 3 that the proposed structure provides noise gains close to the RF optimal structure in [1] and requires less number of multiplications and additions (see Table 2.) It can be also seen that the proposed structure provides significant reduction in the noise gains compared to the optimal structures [4-5] with a significant reduction in multiplication and addition operations. The proposed realization also provides significant reduction in the noise gains compared to the structure reported in [7] with a small increase in multiplication and addition operations.

The results for the coefficient sensitivity analysis for three realizations (the realizations in [4-5] and [1] and the proposed one) are depicted in Figs. 2(a)-(c). The coefficient wordlengths for the elliptic LPF, elliptic HPF and chebyshev HPF were assumed to be 9, 9 and 8 bits, respectively. It can be seen from Fig. 2 that, the performance of the proposed structure is close to the ideal one and to the optimal structures [4-5] which are known to have the lowest coefficient sensitivity. Although the proposed structure is mainly concerned with LPF and HPF cases, the same algorithm can be extended to cover the cases of bandpass and bandstop filters [8].

## 5 VLSI Array-Processor Implementation of the Proposed Structure

VLSI array processors are special-purpose architectures which maximize the processing concurrence by pipeline and parallel processing. In this section, a high-speed array-processor implementation for the proposed structure is presented in which the pipeline technique is used in order to maximize the computation rate. The block-triangular update matrix of the proposed realization allows us to obtain fully-pipelined implementation. The proposed array-processor can be seen as a modified version of the implementation presented in [9] for block-state filters. An array processor implementation will be obtained for the proposed structure without using any pipeline technique and then the slice pipelining technique reported in [10] will be used to increase the computation rate.

The required computations for the proposed structure in (1) and (2) can be divided into four subcomputations. Each subcomputation will be carried out by a different array-processor network. The first array-processor network performs the computations required for the terms $A\hat{x}(n) + e(n)$ in (1). This network is called state-update network (SUN). The other three array-processor networks are assigned to the required computations for $Bu(n)$, $C\hat{x}(n)$ and $Du(n)$, respectively. The SUN network contains the feedback operation of the IIR filter and this will determine the maximum computation rate.

### *5.1 State Update Network (SUN)*

Fig. 3 shows the array-processor implementation for the lower block-triangular state-update matrix of the proposed realization. The internal details of the diagonal processor elements are shown in Fig. 4. It can be seen from Fig. 4 that the computation delay inside the feedback loop is equal the delay of one multiplication and three additions (the quantizer delay is neglected). As this delay determines the maximum throughput rate, it is essential to reduce it. One way to reduce the SUN computation delay is to use the Parallel Array Multiplier (PAM) reported in [11-12] to execute an addition operation and a multiplication operation concurrently. The PAM can be used to execute one multiplication and one double-precision addition concurrently without any need for separate adder [11]. The possibility of incorporating the PAM in the implementation is due to the parallelism inherent in both the array-processing implementation and the RF technique. The schematic I/O diagram of the PAM is shown in Fig. 5(a) and the diagonal processor element of the SUN using the PAM is shown in Fig. 5(b). It can be seen from Fig. 5(b) that the use of PAM inside each diagonal processing element eliminates the need for two adders (one adder inside each feedback loop) and reduces the computation delay inside the feedback loop to the delay of one PAM operation and two additions. Furthermore, the use of the PAM eliminates the need for the separate adders at the left side (outside the feedback loop) of Fig. 4. The delay of the PAM is slightly longer than that of the corresponding Array multiplier (AM) and its area is also slightly larger than that of the corresponding array-multiplier. The internal details of the off-diagonal processor element is shown in Fig. 5(c).

### *5.2 The Pipelined Array-Processor*

The complete array-processor implementation of the proposed realization is shown in Fig 6(a) in which the slice pipeline technique [10] has been applied to maximize the computation rate. The internal details of the processor elements required to compute $Bu(n)$, $C\hat{x}(n)$ and $Du(n)$ are shown in Fig. 6(b). All the latches for pipeline are triggered at the clock rate $F_c$ which is equal to $1/T_c$ where $T_c$ is the delay of the computation inside the feedback loop of the SUN diagonal processor element in Fig. 5(b). From Fig. 5(c) and Fig. 6(b), it is obvious that the delay of involved processor element is not compatible with that of SUN diagonal processor element. All these delays are less than $T_c$ and therefore they will not put any burden on the computation speed. It is easy to slightly alter the structure of the processor elements in Fig. 5(b) and Fig. 6(b) to make their computation delays completely match $T_c$.

## 6 Performance Analysis

In this section, the performance analysis of the proposed array-processor in terms of computation speed, latency, and area complexity is presented.

### *6.1 Computation Delay*

The proposed array-processor implementation for the IIR digital filter can be used to filter data with a sample period $T_s$ provided that

$$T_s \geq T_c \tag{8}$$

where $T_c$ is the clock period which is used to trigger all the latches in Fig. 6(a). $T_c$ can be calculated from

$$T_c = T_{PAM} + 2T_{add} \tag{9}$$

where $T_{PAM}$ is the delay required for a PAM of Fig. 5(a) and $T_{add}$ is the delay required for a double-precision addition.

### *6.2 Latency*

From Fig. 6(a), it can see that the latency $l$ for the proposed array-processing implementation is

$$l = NT_c \tag{10}$$

Therefore the latency is proportional to the filter order. However, in most digital filter applications, the latency is not of prime importance.

### *6.3 Complexity*

It can be seen from Fig. 6(a) that the proposed implementation requires $N^2/2 + 3N + 1$ multipliers ($2N$ PAMs and the other are AMs) and $N^2/2 + 3N$ double-precision adders. The area required for the buffers can not be neglected since the implementation is fully pipelined. The proposed implementation requires $N(N+1)/2$ double-precision buffers and $N^2/2 + 3N/2$ single-precision buffers. From the aforementioned calculations, it appears that the proposed implementation needs relatively high area especially for filters of high order. However, It should be noted that the sizes of the involved multipliers and the lengths of the involved adders are of a low order since the corresponding realization has low output roundoff noise and low coefficient sensitivity as shown in Section 3. For example, any direct realization of the LPF elliptic filter in Table 1 requires 24-bit coefficient wordlength to approximately fit the required frequency response while the proposed realization in section 3 requires only 12-bit coefficient wordlength. This simple comparison shows that although the direct structure requires only $2N + 1$ multipliers, the size of these multipliers are huge that it may render the hardware complexity of implementation based on the direct realization expensive compared to the proposed implementations (for narrow-band medium-order filters).

## 7 Conclusion

Efficient realization (based on the residue feedback technique) for narrow-band IIR digital filter has been proposed. The proposed realization offers excellent performance in terms of output roundoff noise and coefficient sensitivity with a reduced number of required coefficients. An efficient (in terms of speed) VLSI array-processing implementation of the proposed IIR realization is also presented which takes advantage of both the parallelism feature of the residue feedback technique and the block-triangular update state matrix of the proposed realization.

## References


[1] D. Williamson, "Roundoff noise minimization and pole-zero sensitivity in fixed-point digital filters using residue feedback", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, no. 5, pp. 1210-1220, Oct. 1986.

[2] T. L. Chang, "A low roundoff noise digital filter structure", *in Proc. IEEE Int. Symp. Circuit Syst.*, New York, NY, pp. 1004-1008, May 1978.

[3] A. I. Abu-El-Haija and A. M. Peterson, "An approach to eliminate roundoff errors in digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 195-198, Apr. 1979.

[4] C. T. Mullis and R. A. Roberts, "Synthesis of minimum roundoff noise fixed-point digital filters", *IEEE Trans. Circuit Syst.* vol. CAS-23, pp. 551-561, Sept. 1976.

[5] S. Y. Hwang, "Minimum uncorrelated unit noise in state-space digital filtering", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp.273-281, Aug. 1977.

[6] R. A. Roberts and C. T. Mullis, *Digital Signal Processing*, Addison Wesely, 1987.

[7] L. M. Smith and B. W. Bomar, "An algorithm for constrained roundoff noise minimization in digital filters with application to two-dimensional filters", *IEEE Trans. Circuit Syst.* vol. CAS-35, no. 11, pp. 1359-1368, Nov. 1988.

[8] A. Tawfik, P. Agathoklis and F. El-Guibaly, "New IIR digital filter realizations using residue feedback", *submitted to IEEE trans. on Signal Processing.*

[9] H. H. Lu, E. A. Lee, and D. G. Messerschmitt, " Fast recursive filtering with multiple slow processing elements", *IEEE Trans. Circuits Syst.*, vol. CAS-32, pp. 1119-1129, Nov. 1985.

[10] J. R. Jump and S. R. Ahuja, "Effective pipelining of digital systems", *IEEE Trans. Computers*, vol. c-27, no. 9, pp. 855-865, Sept. 1978.

[11] M. O. Ahmad and D. V. Poornalah "Design of an efficient VLSI inner-product processor for real-time DSP applications", *IEEE Trans. Circuits Syst.*, vol. CAS-36, pp. 324-329, Feb. 1989.

[12] S. Sunder, F. El-Guibaly and A. Antoniou, " VLSI implementation of a second-order digital filter", *Canadain Journal of Elect. & Comp. Eng.*, vol. 19, No. 3, pp. 143-147, 1994.


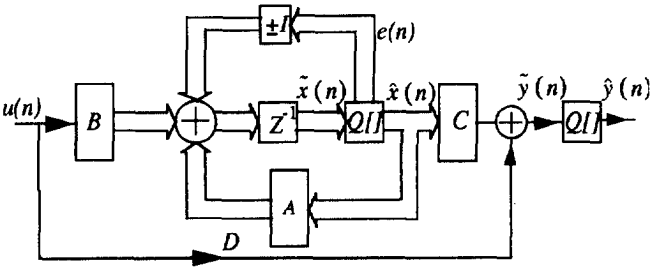


Fig. 1. The block diagram of the RF structure. $\hat{y}(n)$ is the quantized $\tilde{y}(n)$.

**Table 1: Filters Specifications**

| | Elliptic LPF | Elliptic HPF | Cheb. HPF |
|---|---|---|---|
| $A_p$ dB | 1.00 | 0.80 | 0.60 |
| $A_a$ dB | 72.89 | 66.50 | 48.55 |
| $w_{p1}$ rad/s | 250.00 | 4650.00 | 4000.00 |
| $w_{a1}$ rad/s | 400.00 | 4450.00 | 3300.00 |
| $w_s$ rad/s | 10 000.00 | | |

$A_p$ maximum passband ripple, dB

$A_a$ minimum stopband attenuation, dB

$w_{p1}$, $w_{a1}$ passband and stopband edges, rad/s

$w_s$ sampling frequency, rad/s

**Table 2: Complexity of different IIR structures ($N$ is even)**

| | Number of Multiplications | Number of Additions |
|---|---|---|
| Optimal st. [4-5] | $(N+1)^2$ | $N(N+1)$ |
| RF Optimal st. [1] | $(N+1)^2$ | $N(N+2)$ |
| Low-noise st. [7] | $(N^2+5N+2)/2$ | $(N^2+3N)/2$ |
| The proposed st. | $N^2/2+3N+1$ | $N^2/2+3N$ |

**Table 3: Output Roundoff Noise Gains**

| | Elliptic LPF | Elliptic HPF | Cheb. HPF |
|---|---|---|---|
| Optimal st. [4-5] | 1.387 | 1.363 | 1.347 |
| RF optimal st. [1] | 0.040 | 0.069 | 0.370 |
| Low-noise st. [7] | 1.390 | 1.500 | 1.479 |
| The proposed st. | 0.054 | 0.088 | 0.412 |

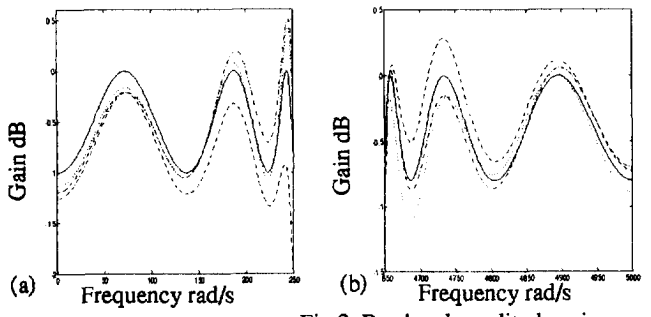


Fig.2. Passband amplitude gain.
(a) Elliptic low-pass filter (wordlength = 9 bits).
(b) Elliptic high-pass filter (word-length = 9 bits).
(c) Chebyshev high-pass filter (wordlength = 8 bits).
"-" Ideal, "--" optimal structure [4-5], ".." RF optimal structure [1]
"-." The proposed structure.

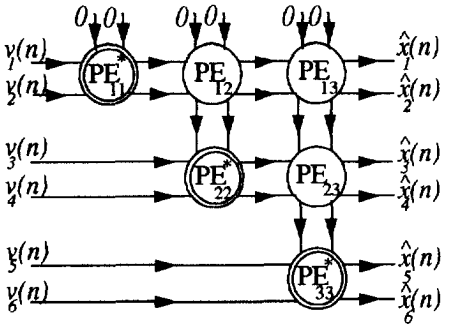


Fig. 3. State-update network (SUN) array-processor ($N$=6).

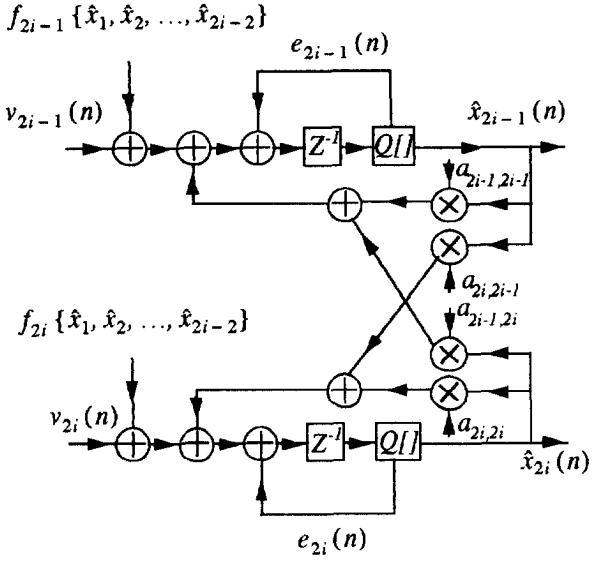


Fig. 4. Details of the $ii$th diagonal processor element PE*.

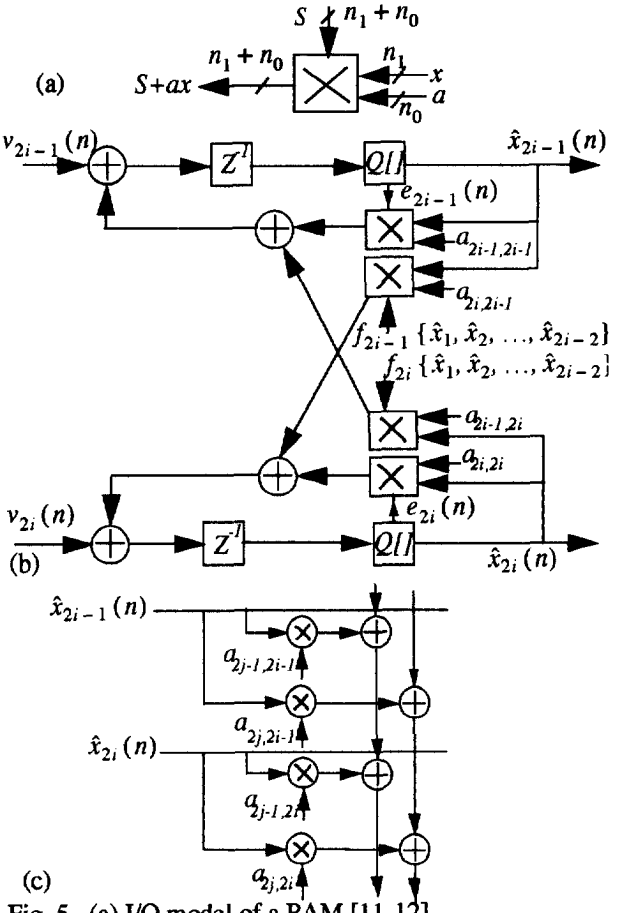


Fig. 5. (a) I/O model of a PAM [11-12]
(b) Details of the $ii$th SUN diagonal processing element using the PAM.
(c) Details of the $ij$th SUN off-diagonal processing element ($j>i$).

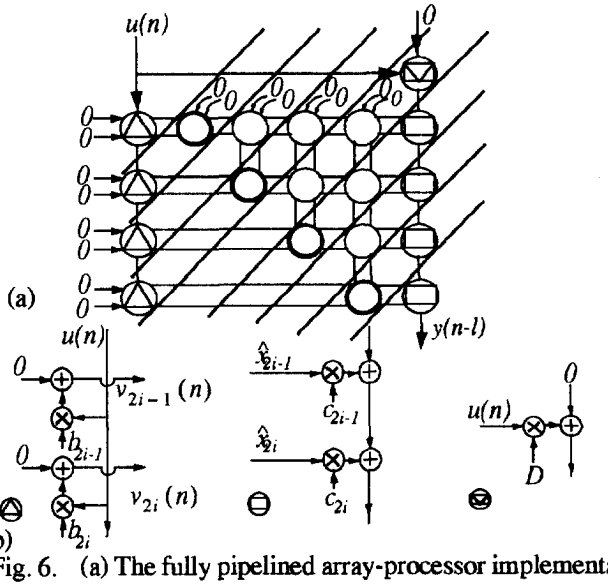


Fig. 6. (a) The fully pipelined array-processor implementation. ($N$ = 8 and $l$ is the latency).
(b) The internal details and the symbols of the involved processors.