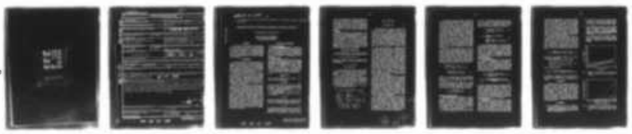
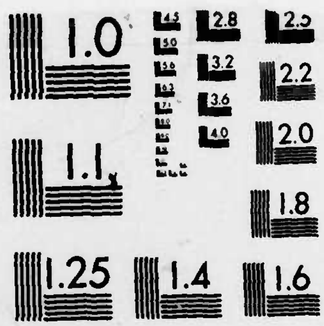
AD-A142 219 INCREASING THE PARALLELISM OF FILTERS THROUGH TRANSPORTATION TO BLOCK STA..(U) GEORGIA INST OF TECH ATLANTA SCHOOL OF ELECTRICAL ENGINEERING.. 1/1

UNCLASSIFIED D A SCHWARTZ ET AL. 1984 AFOSR-TR-84-0350 F/G 9/1 NL



END
DATE
FILMED
7-84
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A142 219

DTIC FILE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S): AFOSR-TR- 84-0550
6a. NAME OF PERFORMING ORGANIZATION: Georgia Institute of Technology; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Air Force Office of Scientific Research
6c. ADDRESS (City, State and ZIP Code): School of Electrical Engineering, Atlanta GA 30332	7b. ADDRESS (City, State and ZIP Code): Directorate of Mathematical & Information Sciences, Bolling AFB DC 20332
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: AFOSR; 8b. OFFICE SYMBOL (If applicable): NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: DAAG29-81-K-0024
8c. ADDRESS (City, State and ZIP Code): Bolling AFB DC 20332	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO.: 61102F; PROJECT NO.: 2304; TASK NO.: A9; WORK UNIT NO.

11. TITLE (Include Security Classification): INCREASING THE PARALLELISM OF FILTERS THROUGH TRANSPORTATION TO BLOCK STATE VARIABLE FORM

12. PERSONAL AUTHOR(S): D.A. Schwartz and T.P. Barnwell III

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Technical	FROM ______ TO ______	1984	4

16. SUPPLEMENTARY NOTATION

17. COSATI CODES FIELD	GROUP	SUB. GR.	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
			pp. 1-4. 1984

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The block state variable form is investigated as a technique to increase the parallelism of a filter. This increase in parallelism allows more parallel processors to be usefully applied to the problem, resulting in a faster processing rate than is possible in the unblocked form. Upper and lower bounds on the sample period bound and the number of processors required to support it are determined.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Dr. Joseph Bram	22b. TELEPHONE NUMBER (Include Area Code): (202) 767- 4939; 22c. OFFICE SYMBOL: NM

DD FORM 1473, 83 APR

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

84 06 18 153

AFOSR-TR- 84-0350 1/4

A142 219

To be presented at the International Conference on Acoustics, Speech, and Signal Processing.

INCREASING THE PARALLELISM OF FILTERS THROUGH TRANSFORMATION TO BLOCK STATE VARIABLE FORM

D. A. Schwartz and T. P Barnwell, III

School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

ABSTRACT

The block state variable form is investigated as a technique to increase the parallelism of a filter. This increase in parallelism allows more parallel processors to be usefully applied to the problem, resulting in a faster processing rate than is possible in the unblocked form. Upper and lower bounds on the sample period bound and the number of processors required to support it are determined.

INTRODUCTION

In digital filtering applications where the maximum processing rate is of fundamental importance, in particular real-time processing, higher rates can be achieved by faster processors or more parallel processors. For many problems faster hardware is not practical or cost effective compared to simple multiprocessor solutions, particularly for VLSI implementations.

Recurrence relations, such as recursive filters, specified by "fully specified signal flow graphs," have been shown to have a maximum parallelism that is constrained by one or more "critical loops." Adding additional processors, beyond the maximum parallelism, performs no directly useful work. However it is possible to increase the parallelism of the problem by transformation to a block form.

This paper concentrates on the block state variable form. Any particular fully specified member of this class of filters has a well defined sample period bound and any particular filter has a specific fully specified form which results in the minimum sample period bound. Determination of the exact bound requires a lengthy search operation. However, the determination of the sample period bound can itself be bounded by the gross properties of the system matrix. This paper explores the block state variable form and determines an upper and lower bound on the "sample period bound," and the associated number of processors required. It is also shown that for many problems the blocked form has lower computational requirements, and decreased finite word effects, even if evaluated on a typical sequential uniprocessor.

BACKGROUND AND DEFINITIONS

Flow Graph Specification

A fully specified flowgraph is a generalized flow graph in which the node operations are all fundamental operations of the constituent processor on which the algorithm will be implemented [1]. The definition of the node operations in the fully specified flow graph sets the granularity with which the parallelism can be exploited.

Flow Graphs Bounds

Given a fully specified flow graph it is possible to compute the lower bound on the sample period bound (or rate bound which is the reciprocal of the sample period bound), which is always achievable. The sample period bound is best understood in the context of a recursive single-time-index flow graph (e.g. an IIR digital filter), although the concept is meaningful in systems which have no explicit sample period.

For such systems the sample period bound is given by

$$T_0 = \mathrm{MAX}_{\ell} [D_\ell / n_\ell] \qquad (1)$$

Where ℓ varies over the set of all loops. D_ℓ is the total delay around loop ℓ.

$$D_\ell = \sum_{i \in \ell} [d_i] \qquad (2)$$

The computational time to perform the operation of node i is d_i, and n_ℓ is the number of delays in loop ℓ. This is a generalization of a result published by Renfors and Nuevo [2].

Any loop for which $T_\ell = D_\ell / n_\ell = T_0$ is considered a criticial loop.

Let D be the total computational delay of all the nodes.

$$D = \sum_i [d_i] \qquad (3)$$

Then the maximum parallelism, or number of processors in a "processor optimal" solution, is the total delay divided by the sample period bound.

$$P = D/T_0 \qquad (4)$$

The maximum parallelism thus defined is the maximum parallelism such that at all instances P operations can be performed in parallel. It is important to note tha this is not the same con-

Approved for public release;
distribution unlimited.

84 06 18 153

cept as the maximum number of parallel operations that can be achieved by a "greedy scheduler," in which each operation is performed as soon as it is possible. Rather, it is a constant level of parallelism which allows for exactly P operations to be performed on every cycle. Another important point to restate is that using more than P processors will not decrease the sample period bound.

Optimality

This work assumes an implementation that meets the following optimality criteria. An implementation is processor optimal if it exhibits perfect processor efficiency, if every cycle of every processor is used directly on the fundamental operations of the algorithm (flow graph) and no cycles are used for synchronization or system control. If an implementation achieves the sample period bound it is considered rate optimal. From the previous section it is seen that an implementation that is processor and rate optimal requires exactly P processors.

BLOCK STATE VARIABLE FORM

Any system, $H(s)$, with a rational transfer function can be expressed in state variable form:

$$H(s) = C(sI - A)^{-1}B + D$$

Let W_k be the state vector, U_k the input and y_k the output at time k. The state equation is then the familiar:

$$\Sigma: \quad W_{k+1} = AW_k + BU_k$$

$$y_k = CW_k + DU_k \qquad (5)$$

For simplicity and clarity this paper will only consider single input, single output (SISO) systems. The generalization is straightforward. For the case of a system of order N, A is N×N, B is N×1, C is 1×N and D a scalar.

The original scalar system, Σ, can be converted to a block form system, Σ_L, that operates on a block of L (sequential) inputs and produces L (sequential) outputs in parallel. Our goal is to show that as the block size increases the sample period bound decreases.

The new system, $\hat{\Sigma}_L$, is defined in terms of the original system as follows:

$$\hat{\Sigma}: \quad \hat{A} = A^L, \quad \hat{B} = \left[A^{L-1}B \,|\, A^{L-2}B \,|\, \cdots \,|\, AB \,|\, B\right]$$

$$\hat{C} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{L-1} \end{bmatrix} \qquad \hat{D} = \begin{bmatrix} D & & & & \underline{O} \\ CB & D & & & \\ CAB & CB & D & & \\ \vdots & \ddots & \ddots & \ddots & \\ CA^{L-2}B & \cdots & CAB & CB & D \end{bmatrix}$$

$$\hat{U}_k = \left[U_{KL} \;\; U_{KL+1} \;\cdots\; U_{KL+L-1}\right]^T$$

$$\hat{y}_k = \left[y_{KL} \;\; y_{KL+1} \;\cdots\; y_{KL+L-1}\right]^T$$

$$\hat{W}_{k+1} = \hat{A}\hat{W}_k + \hat{B}\hat{U}_k$$

$$\hat{y}_k = \hat{C}\hat{W}_k + \hat{D}\hat{U}_k \qquad (6)$$

The poles of Σ and $\hat{\Sigma}_L$ are the eigenvalues of A and $\hat{A}$ respectively, denoted $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ and $\{\hat{\lambda}_1, \hat{\lambda}_2, \cdots, \hat{\lambda}_n\}$. Since $\hat{A} = A^L$, then $\hat{\lambda}_i = \lambda_i^n$. As the block size L increases, the poles of $\hat{\Sigma}_L$ spiral into the origin. This leads to increased stability and decreased coefficient quantization error since the coded coefficients are those of A^L. Given fixed point implementation with finite precision and a sufficiently large block size, L>N, $\hat{\Sigma}_L$ reduces to an FIR (no recursion) system, which has unbounded parallelism. The sampling rate of an FIR system is bounded only by available resources and tolerable throughput delay [3].

What of the parallelism of a blocked system with block size less than N? A difficulty with the block state variable (state variable) form is that while the $\hat{A}$ matrix defines the form of the recursive part of the network, it does not specify the order of the additions. To rephrase, the state equations only define the generic signal flow graph. Recall that only a fully specified signal flow graph has a sample period bound and that a generic graph may have a large number of different fully specified graphs with different associated bounds. A good example is a non-blocked, Nth order direct form canonic filter. The optimal fully specified flow graph has a sample period bound of $t_a + t_m$ (add time + multiply time), while the worst case fully specified graph has a bound of $(N-1)t_a + t_m$. For a specific generic graph, it is straightforward to find the fully specified form with the lowest possible sample period bound using an iterative tree height balancing algorithm.

Despite these difficulties, it is still possible to specify an upper and lower bound on the sample period bound of the state equations. Consider the block state system. Computation of $\hat{y}_k$ given $\hat{W}_k$ is non recursive and can be overlapped with following blocks, if necessary. The matrix product $\hat{B}_k\hat{U}_k = \hat{V}_k$ can be precomputed, over preceeding blocks, if necessary, resulting in a new simple input vector. Thus, the sample period is determined by the recursive portion plus a simple input. Examining the form of the update equations it can be seen that the update of each state variable can proceed in parallel. This viewpoint leads to the determination of the upper bound on the sample period bound.

In general for the block state form, occurrence of zeroes in the $\hat{V}_k$ vector are rare for non-zero input sequences. All of the multiplications of $(\hat{A})_{ij} * (\hat{W}_k)_j$ can proceed in parallel contributing a delay of t_m. Summing the products of row i of $\hat{A}$ with $\hat{W}_k$ plus the input term $(\hat{V}_k)_i$, with a balanced tree summer, introduces a delay of $\lceil \log_2 n_i \rceil t_a$, where n_i is the number of non zero coefficients in row i of $\hat{A}$. The upper bound on the sample period is therefore determined by the row of $\hat{A}$ with the most non zero coefficients.

To determine the lower bound, recall that what determines the bounds are loops. The update of the state variable $(W_k)_i$ is a weighted sum of all state variables. If in the computation of $(W_k)_i$, $(W_k)_j$ does not form a loop with $(W_k)_i$ then the weighted sum of all $(W_k)_j$, $j \in J$ (J is the set of indexes j such that $(W_k)_j$ does not form a loop with $(W_k)_i$) can be precomputed as a single input $(u_i = (A)_{ij} \cdot (W_k)_j)$. This leads to at least one of the state variables not containing a precomputable partial weighted sum. Therefore, the lower bound must be greater than or equal to that associated with the row of A with the least coefficients. However since it is possible that the critical loop contains a p unit delay instead of a unit delay if is necessary to divide the computational delay by p to yield the sample period bound. A necessary condition for a p unit delay to exits in a critical loop is that $\hat{A}$ contains p rows with precisely one non-zero coefficient.

For block form systems, what is of main interest in not the sample period bound, but the sample period bound per output sample. This is just the sample period bound divided by the block size, which yields the average time between successive output samples. The per output qualification hereafter is implied when referring to the sample period bound, unless stated otherwise. The bounds on the sample period bound is therefore given as follows (for the original unblocked system substitute L=1 and A for $\hat{A}$):

$$\frac{\lceil \log_2(\min_i \{n_i\} + 1)\rceil t_a + t_m}{pL} < \hat{T}_o < \frac{\lceil \log_2(\max\{n_i\} + 1)\rceil t_a + t_m}{L} \quad (7)$$

Where n_i is the number of non zero coefficients in row i, p is the number of rows of $\hat{A}$ with exactly one non-zero coefficient and L is the block size.

If the system is not a parallel or serial cascade then blocking the system with a block size of L>N-2 typically results in a system with no (very few) non-zero coefficients. This results in the worst case sample period bound of:

$$\hat{T}_o = \frac{\lceil \log_2(N+1)\rceil t_a + t_m}{L} \quad (8)$$

Computational Requirements

The computational requirements in terms of the number of operations and number of required processors is derived by assuming a straight forward implementation of the state variable equations. It is further assumed that the system is of state space form, and has no zero coefficients (worst case). The constituent processors are assumed to have kernal operations of "two input addition" and "multiplication."

The number of multiplications are the number of non zero coefficients in $\hat{A}$, $\hat{B}$, $\hat{C}$, and $\hat{D}$ (A, B, C and D). The number of additions are n-1 for each n element row a column inner product and n for each addition of n element vectors. Therefore the number of multiplies per output and the number of additions per output are given by:

$$\Sigma: \quad \begin{aligned} N_{mult}/\text{output} &= N^2 + 2N + 1 \\ N_{add}/\text{output} &= N^2 + N + 1 \end{aligned} \quad (9)$$

$$\hat{\Sigma}_L: \quad \begin{aligned} \frac{\hat{N}_{mult}}{\text{output}} &= \frac{N}{L} + 2N + \frac{L+1}{2} = \frac{L}{2} \\ \frac{\hat{N}_{add}}{\text{output}} &= \frac{N(N-1)}{L} + 2N + \frac{L-1}{2} = \frac{L}{2} \end{aligned} \quad (10)$$

As can be seen from Fig. 1, for block sizes less than approximatley $2N^2$, the total number of multiplications is less than for the nonblocked or L=1 form. The minimum for the number of multiplies per output occurs for a block size of $L=\sqrt{2N}$. The graphs for additions are nearly identical to those for the multiplications, with the minima occuring at $L = \sqrt{2N(N-1)}$. More sparse realisations may have less significant savings in total operations.

Number of Processors

Making the assumption that $t_m = \alpha t_a$, allows for a simpler determination of the number of processors or parallelism from the number of operations. As in the previous portions, this result is for the fully populated state space form, which is known to have processor and rate optimal solutions. The number of processors is equal to the total aritmetic delay divided by the sample period bound.

$$p = \frac{D}{T_o} = \frac{(\alpha+1)N^2 + (2\alpha+1)N + \alpha+1}{\lceil \log_2 N+1\rceil + \alpha} t_a \quad (11)$$

$$\hat{p} = \frac{\hat{D}}{\hat{T}_o} = \frac{(\alpha+1)(2L+N)N - N + [(\alpha+1)L^2 + (\alpha-1)L]/2}{\lceil \log_2 N+1\rceil + \alpha} t_a \quad (12)$$

For block systems the order of the number of the processors required is roughly proportional to the block size squared (since N is fixed). Combining equations (8) and (12) for, α=1, the number of normalized processors as a function of the normalized rate bound is shown in Fig. 2. Normalization in this case implying that for L=1, one normalized processor processes at a normalized rate of one. Thus the graph indicates the relative cost of a given rate increase.

Block Normal Form

Increasing the block size tends to decease the sparseness of the system matrix $\hat{A}$, and then leads to larger increases in the number of operations. If the unblocked system is of block diagonal form, the blocked system matrix is of the same block diagonal form, with no attendant decrease in sparseness. While the first form that may occur to the reader is the Jordan normal form, this implies complex arithmetic which leads to greater complexity and an increased sample period bound (t_m-complex = t_m-real + t_s-real). The parallel cascade of second order normal form sections leads to an attractive block diagonal form. The block normal form is particularly attractive in that Barnes [4] has shown that 1) average roundoff noise is decreased by a factor of L, 2) for L sufficiently large all autonomous limit cycle can be eliminated, 3) minimum noise unblocked forms lead to minimum noise blocked forms and 4) scaling for fixed point implementations of the unblocked system results in a blocked system with proper scaling.

To determine the sample period bound and parallelism of a parallel normal form consider an Nth order (N even) system with block size L. The system matrix for this case is block diagonal with each block being a 2×2 submatrix with non-zero coefficients. Since each row of $\hat{A}$ has exactly two non-zero coefficients, the upper and lower bounds on the sample period bound are the same. Therefore the sample period bound is given by:

$$\hat{T}_o = \frac{\lceil \log_2 3 \rceil t_s + t_m}{L} = \frac{2t_s + t_m}{L} = \frac{(\alpha+1)t_s}{L} \quad (13)$$

Note that this system exhibits direct linear speedup with block size.

Counting operations per output sample:

$$\begin{aligned} \hat{N}\text{mult/output} &= 2N(L+1)/L + (L+1)/2 \\ \hat{N}\text{add/output} &= N(2L+1)/L + (L-1)/2 \end{aligned} \quad (14)$$

The parallelism is then:

$$\hat{P} = \hat{T}_o/\hat{D} = \frac{(\alpha+1)L^2 + [(\alpha+1)4N+\alpha-1]L + (4\alpha-2)N}{2\alpha+4} \quad (15)$$

The number of processor is thus of order $L^2/2$. The number of multiplies is minimised for $L = 2\sqrt{N}$, and the number of adds is minimised for $L = \sqrt{2N}$.

CONCLUSION

Transforming a state variable system to a block state variable form increases the effective parallelism and decreases the sample period bound. The sample period bound asymptotically approaches direct linear speedup as the block size increases, with an attendant cost of order L^2 processors. The block form not only has better numerical properties than the unblocked form, it may require fewer operations. Even if the implementation is to be a sequential uniprocessor the numerical and complexity properties of the block form offer significant benefits over the unblocked form.

REFERENCES

[1] T. P. Barnwell, III and D. A. Schwartz, "Optimal Implementations of Flow Graphs on Synchronous Multiprocessors," Proc. 1983 Asilomar Conf. on Circuits and Sys., Pacific Grove, CA, November 1983.

[2] M. Renfors and Y. Neuvo, "The Maximum Sampling Rate of Filters Under Hardware Speed Constraints," IEEE Trans. Circuits Syst., Vol. CAS-28, No. 3, pp. 196-202, Mar. 1981.

[3] T. P. Barnwell, III and C. J. M. Hodges, "Optimum Implementation of Single Time Index Signal Flow Graphs on Synchronous Multiprocessor Machines," 1982 International Conference on Acoustics, Speech and Signal Processing, Paris, France, May 1982.

[4] C. W. Barnes, "Finite Word Effects in Block-State Realizations of Fixed-Point Digital Filters," IEEE Trans. on Circuits and Systems, Vol. CAS-27, No. 5, May 1980, pp. 345-349.

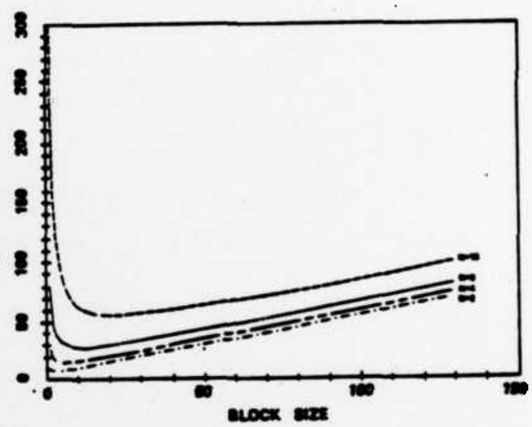


Fig. 1 Number of multiples as a function of block size for state space system of order N.

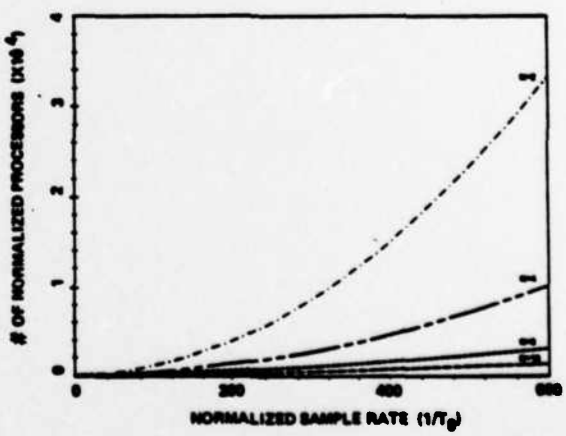


Fig. 2 Normalized number of processors required to achieve a normalized sample rate increase for state space system of order N.

DATE
ILME