Correspondence

On Bit Steering in the Minimization of the Control Memory of Microprogrammed Processors

NRIPENDRA N. BISWAS

Abstract — **The microcommands constituting the microprogram of the control memory of a microprogrammed processor can be partitioned into a number of disjoint sets. Some of these sets are then encoded to minimize the word width of the ROM storing the microprogram. A further reduction in the width of the ROM words can be achieved by a technique known as bit steering where one or more bits are shared by two or more sets of microcommands. These sets are called the steerable sets. This correspondence presents a simple method for the detection and encoding of steerable sets. It has been shown that the concurrency matrix of two steerable sets exhibits definite patterns of clusters which can be easily recognized. A relation "connection" has been defined which helps in the detection of three-set steerability. Once steerable sets are identified, their encoding becomes a straightforward procedure following the location of the identifying clusters on the concurrency matrix or matrices.**

Index Terms — **Bit minimization, bit steering, control memory, microprogramming, minimization, ROM compaction.**

I. Introduction

In a microprogrammed processor, each microinstruction is executed by performing one or more simpler operations known as microoperations or microcommands. A program consisting of such operations is stored in the ROM of the control memory and is known as the microprogram. A microinstruction thus becomes a word of the ROM and the width of a word is equal to the number of microcommands in that word. This type of arrangement where each microcommand is activated by a distinct bit of the ROM is known as the horizontal organization. This arrangement, although rendering maximum flexibility, is very wasteful of the ROM. It was Schwartz [1] who first recognized that the word width can be minimized by partitioning the set of microcommands into many disjoint sets, such that no two microcommands that occur at the same time are included in a set. These mutually exclusive sets are then encoded. Each encoded set is assigned a field of an appropriate number of bits in the ROM. Later, many authors [2]–[6] propounded improved methods to achieve this bit optimization of the control memory. In many cases the number of bits in a word can be further reduced by a technique known as bit steering [7], [8].

Table I gives the sample microprogram of Schwartz [1] as rewritten by Grasselli and Montanari [2], and Table II gives the five different but valid solutions obtained by various authors [2]–[6]. Each of these solutions requires 9 bits. Fig. 1 shows the implementation of Solution 5 with 9 bits, whereas Fig. 2 shows the implementation with 8 bits due to bit steering. In an earlier paper [8] Mathialagan and Biswas have discussed the technique of bit steering in detail, and have given an algorithm for the detection and encoding of steerable sets. In this correspondence we present a simpler method of doing the same.

II. The Method

In order to understand the principle of the method, let us study Fig. 3, where the two steerable sets $(x_0x_1x_2x_3)$ and $(y_0y_1y_2y_3)$ have been implemented by bits A (the common bit), B, and C. A combined truth table for the X set (implemented by AB) and the Y set (implemented by AC) is shown in Fig. 4. It may be observed from the circuit and the truth table that x_0 may occur concurrently with y_0 and y_1 as A is 0 in these cases. But x_0 cannot occur concurrently with y_2 or y_3 as A cannot take the values of 0 and 1 simultaneously. This gives rise to the concurrency matrix (CM) between members of the two disjoint sets X and Y as shown in Fig. 4. In the CM, a 1 has been placed in the cell of x_i column and y_j row, if x_i and y_j can occur concurrently. An interesting feature of the CM of Fig. 4 is that it has two disjoint clusters of four, each having entries in two rows and two columns. This pattern is very significant and can be explained by the following theorem.

Manuscript received November 8, 1983; revised March 12, 1985.
The author is with the Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore-560012, India.

TABLE I
Sample Microprogram

Words	Microcommands					
1	*a*	*b*	*c*	*d*	*e*	*f*
2	*c*	*g*	*h*	*i*		
3	*a*	*b*	*h*	*i*	*j*	
4	*d*	*h*	*k*			
5	*f*	*h*				

TABLE II
Minimum Bit Solutions for the Sample Microprogram

1	(*a*)	(*b*)	(*c*)	(*d*)	(*h*)	(*eik*)	(*fgj*)
2	(*a*)	(*b*)	(*c*)	(*f*)	(*h*)	(*dgj*)	(*eik*)
3	(*a*)	(*b*)	(*c*)	(*eh*)		(*dgj*)	(*fik*)
4	(*a*)	(*b*)	(*c*)	(*e*)	(*h*)	(*dgj*)	(*fik*)
5	(*a*)	(*b*)	(*c*)	(*d*)	(*h*)	(*egj*)	(*fik*)

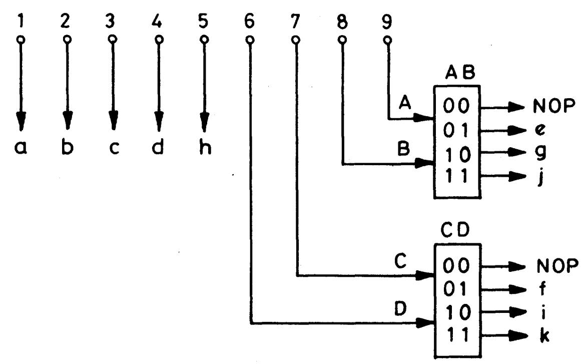


Fig. 1. Implementation of Solution 5 with 9 bits.

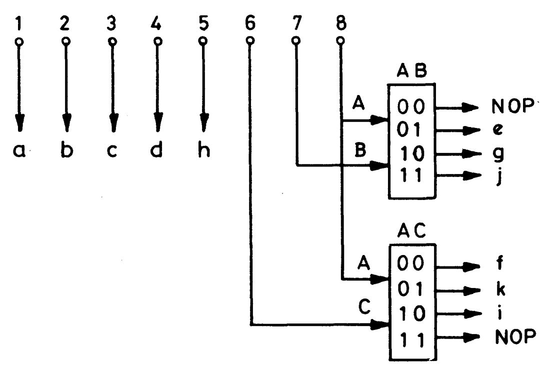


Fig. 2. Implementation of Solution 5 with 8 bits.

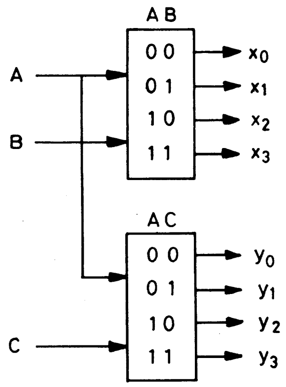
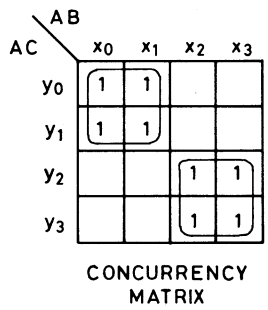


Fig. 3. Implementation of two steerable sets X and Y.

A	B(C)	AB	AC
0	0	x_0	y_0
0	1	x_1	y_1
1	0	x_2	y_2
1	1	x_3	y_3

TRUTH TABLE



CONCURRENCY MATRIX

Fig. 4. Truth table and concurrency matrix of the two steerable sets X and Y.

In this theorem we shall call X the column set (and Y the row set) as its members represent the columns (rows) of the concurrency matrix. Let the set $X(Y)$ be encodable by $n(m)$ bits so that the concurrency matrix has 2^n columns and 2^m rows. We shall also require two disjoint clusters to satisfy the following definition.

Definition 1: Two clusters are *disjoint* when not only the clusters themselves but also their projections in either direction are disjoint.

This means that a particular row or a column occupied by one or more elements of a cluster will belong to this cluster only, and cannot be shared by any element of another cluster, if the two clusters are disjoint.

Theorem 1: The column set X (encodable by n bits) and the row set Y (encodable by m bits) are steerable with s bits in the common field if their concurrency matrix having 2^m rows and 2^n columns can produce 2^s disjoint clusters of 2^p rows and 2^q columns such that

$$0 < s < \text{Minimum } (m, n)$$

$$p = m - s$$

$$q = n - s.$$

The concurrency matrix must not produce any other clusters, joint or disjoint.

Proof: Number of clusters = number of binary combinations of the bits in the steering (shared) field

$$= 2^s.$$

Therefore, number of rows in a cluster

$$= (\text{number of rows in the CM})/(\text{number of clusters})$$

$$= 2^m/2^s = 2^{m-s} = 2^p$$

Number of columns in a cluster

$$= (\text{number of columns in the CM})/(\text{number of clusters})$$

$$= 2^n/2^s = 2^{n-s} = 2^q$$

It is obvious that

$$0 < s < \text{Minimum } (m, n)$$

and the concurrency matrix must not produce any other clusters, joint or disjoint.

Q.E.D.

The following corollary is now obvious.

Corollary 1.1: The above two sets X and Y are not steerable if there are more than $2^p(2^q)$ entries in a column (row).

It now follows from this theorem and its corollary that two disjoint sets X and Y, which can be encoded by 2 bits each, are steerable if their concurrency matrix (CM) can produce two, and only two, disjoint clusters of four, each having entries in two rows and two columns. The CM must not exhibit any other cluster, joint or disjoint. Moreover, the two sets are not steerable if there are more than two entries in any row/column.

Let us now apply these criteria to detect the steerability of the two sets (egj) and (fik) of Solution 5 of the sample microprogram. The concurrency matrix of the two sets is to be derived from the microprogram and is shown in Fig. 5.

It can be seen that we do not get eight entries in the CM. This simply means that, although eight concurrencies exist in the circuit and the truth table, they do not exist (or are not required) in the microprogram. We may, therefore, consider these as *don't care* concurrencies (ϕ). We are also free to place these don't care entries (ϕ's) in any of the blank cells. But once a ϕ participates in forming a cluster it must be considered as a 1. Now, if we interchange rows and columns in the CM of Fig. 5 and use three don't care concurrencies, the CM of Fig. 5 gets transformed into the CM as shown in Fig. 6(a). We see that the sets (egj) and (fik) satisfy Theorem 1 and are, therefore, steerable.

In actual practice it is not necessary to interchange rows and columns, but the clusters can be formed and the steerability detected on the original CM itself as shown in Fig. 6(b). Once the clusters have been formed it is easy to encode the rows and columns. In the CM of Fig. 4 the columns x_0 and x_1 are interchangeable. The rows y_0 and y_1 are also interchangeable. The other interchangeable pairs are x_2x_3 and y_2y_3. So, we have many choices to encode the rows and columns. Assigning the $00(x_0)$ code to the *NOP* of the (egj) group, one of the possible encodings is shown in Fig. 7. It should be observed that once we assign x_0 to the *NOP* of the (egj) group, the concurrency matrix dictates that e must be assigned x_1 and g and j must be assigned $x_2(x_3)$ and $x_3(x_2)$, respectively. Again the assignment of x_0 to *NOP* of the egj group requires that the f and k rows must be assigned $y_0(y_1)$ and $y_1(y_0)$, respectively. The i and *NOP* (of the *fik* group) rows then must be assigned $y_2(y_3)$ and $y_3(y_2)$, respectively.

The concurrency matrices of Solutions 1, 3, and 4 are given in Fig. 8. The results with reasons are as follows.

a) Solution 1: Any attempt to make a cluster of four puts three concurrencies in a row (column). Hence not steerable.

b) Solution 3, the sets (eh), and (fik): There are three 1's in column h. Hence not steerable.

c) Solutions 3 and 4: The clusters of 4's are as shown. Hence steerable. The encodings are also shown. Starting point of the encoding was the assignment of 00 to the *NOP* of the *dgj* group.

Let us now find the steerability criteria for two sets, each requiring a field of 3 bits for encoding. It can be easily seen that if m and n are the lengths (number of members) of the sets then

$$4 < m \leq 8$$

and

$$4 < n \leq 8.$$

Here two cases may arise. In one, there may be only 1 bit in the common field, and in the other there are 2 bits in the common field. The conditions to be satisfied for the two sets to be steerable can be

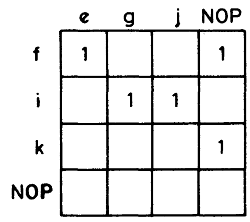
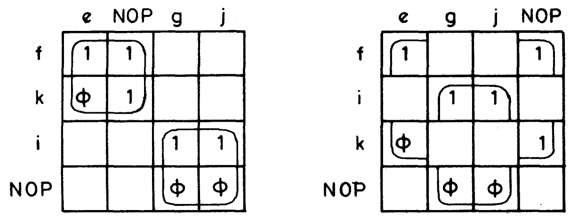


Fig. 5. Concurrency matrix of the sets (egj) and (fik).



(a) (b)

Fig. 6. Two disjoint clusters of four on the concurrency matrix.

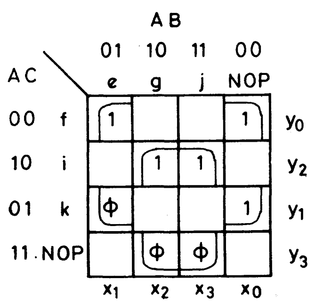


Fig. 7. Encoding the columns and rows of the steerable sets.

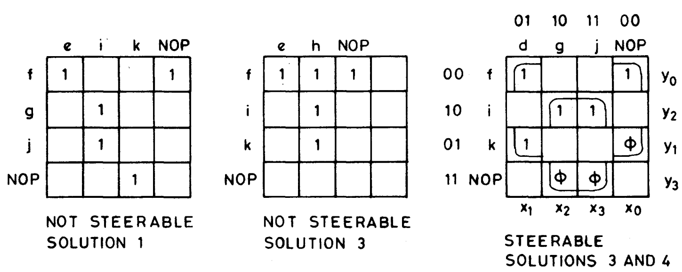


Fig. 8. Concurrency matrices of Solutions 1, 3, and 4. Also shows the encoding of the steerable pair of sets of Solutions 3 and 4.

A	B(D)	C(E)	ABC	ADE
0	0	0	x_0	y_0
0	0	1	x_1	y_1
0	1	0	x_2	y_2
0	1	1	x_3	y_3
1	0	0	x_4	y_4
1	0	1	x_5	y_5
1	1	0	x_6	y_6
1	1	1	x_7	y_7

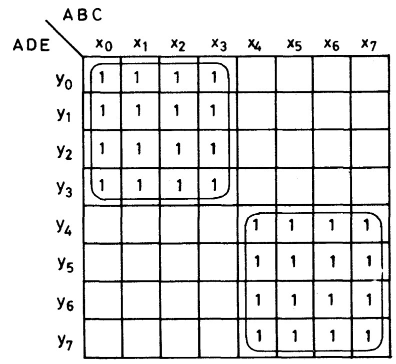


Fig. 9. Truth table and concurrency matrix of two steerable sets X (3 bits) and Y (3 bits) with 1 bit in the common field.

A	B	C(D)	ABC	ABD
0	0	0	x_0	y_0
0	0	1	x_1	y_1
0	1	0	x_2	y_2
0	1	1	x_3	y_3
1	0	0	x_4	y_4
1	0	1	x_5	y_5
1	1	0	x_6	y_6
1	1	1	x_7	y_7

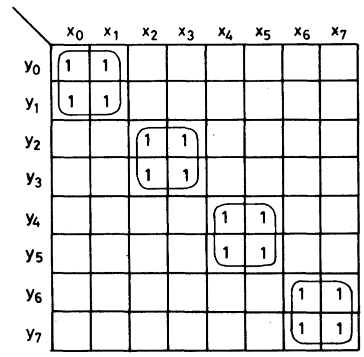


Fig. 10. Truth table and concurrency matrix of two steerable sets X (3 bits) and Y (3 bits) with 2 bits in the common field.

A	B(D)	C	ABC	AD
0	0	0	x_0	y_0
0	0	1	x_1	y_0
0	1	0	x_2	y_1
0	1	1	x_3	y_1
1	0	0	x_4	y_2
1	0	1	x_5	y_2
1	1	0	x_6	y_3
1	1	1	x_7	y_3

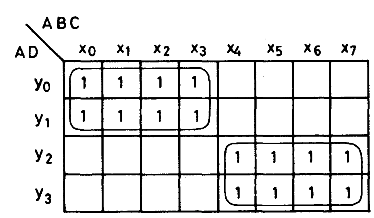


Fig. 11. Truth table and concurrency matrix of two steerable sets X (3 bits) and Y (2 bits) with 1 bit in the common field.

derived from Theorem 1. But as this is an important case the criteria are stated as Theorem 2 and Corollary 2.1. The truth tables and concurrency matrices of these two cases are shown in Figs. 9 and 10.

Theorem 2: Two disjoint sets X and Y, which can be encoded by 3 bits each, are steerable with 1(2) bit (bits) in the common field, if their concurrency matrix can produce 2(4) and only 2(4) disjoint clusters of 16(4) entries each having 4(2) rows and 4(2) columns, and it does not exhibit any other cluster, joint or disjoint.

Corollary 2.1: Two disjoint sets X and Y, which can be encoded by 3 bits each, are not steerable with 1(2) bit (bits) in the common field, if the concurrency matrix has more than 4(2) entries in a row or a column.

The truth table and the concurrency matrix for two steerable sets of uneven length, say, one encodable with 2 bits and the other with 3 bits, are shown in Fig. 11. The steerability criteria for this case can also be derived from Theorem 1.

Although the number of elements including *NOP* in the row (column) set $Y(X)$ may not always be exactly equal to $2^m(2^n)$, the concurrency matrix must always have exactly 2^m rows and 2^n columns. The clusters can be formed by including don't care concurrencies which may be located in, shall we say, don't care rows or columns. (See Fig. 12.)

A. Three-Set Steerability

It has been shown in [8] that the steerability relation is not transitive, and that a necessary condition for three sets to be steerable together is that they must be pairwise steerable. Let the pairwise concurrency matrices C_1, C_2, and C_3 of the three sets $X(abc)$, $Y(def)$, and $Z(ghi)$ be drawn in a three-dimensional coordinate system as shown in Fig. 13. The concurrency matrices show that the three sets are pairwise steerable. The encoding, which is according to the concurrency matrices C_1 and C_3, is shown in Fig. 13. Now, if g and h of the Z set are assigned the codes z_0 and z_1, then the CM C_2 requires that f of set Y must be assigned either y_0 or y_1. This is in conflict with the requirement of the CM C_1, which must assign either y_2 or y_3 to f. Thus, the three sets of Fig. 13 are not steerable together. On the other hand, if the CM C_2 is as shown in Fig. 14,

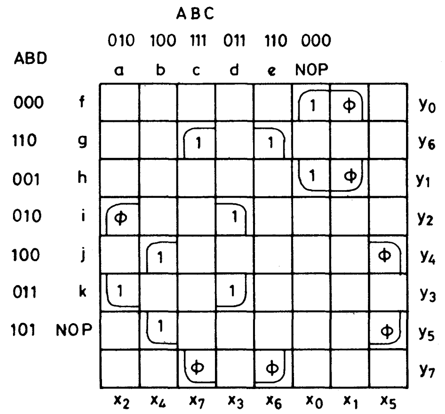


Fig. 12. Detection and encoding of steerable sets on a concurrency matrix exhibiting don't care rows and columns.

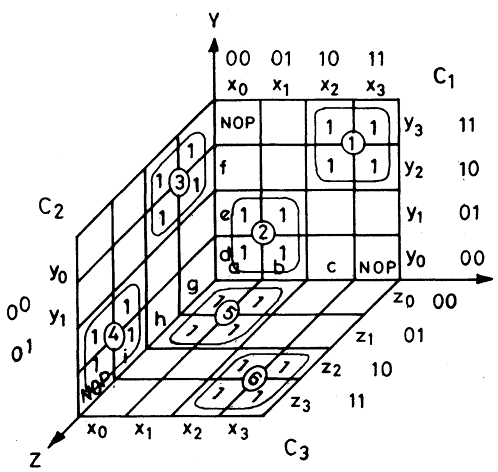


Fig. 13. Concurrency matrices on a three-dimensional coordinate system.

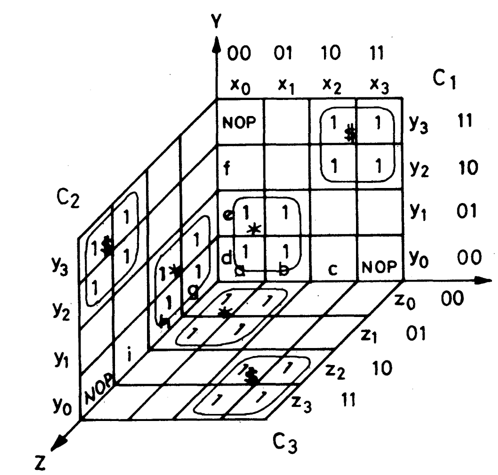


Fig. 14. Concurrency matrices showing two groups of strongly connected clusters.

then there is no conflict and an encoding exists to steer all the three sets together. A close look at the orientations of the clusters of Figs. 13 and 14 reveals that this existence of an encoding, which is also a necessary condition for the three sets to be steerable, depends on the special orientation which is present among the clusters of Fig. 14 and absent among those of Fig. 13. We shall now develop a formal definition of this special orientation.

At this point let us define a relation *connection* which may exist between two clusters on the two concurrency matrices drawn in a three-dimensional coordinate system.

Definition 2: A *connection* is said to exist between two clusters situated on two concurrency matrices drawn in a three-dimensional cartesian coordinate system if any row/column of one cluster is connected at one of the axes (of the coordinate system) to any row/column of the other cluster.

For example, in Fig. 13 cluster 1 of CM C_1 has a connection with cluster 6 of CM C_3 because two columns of 1 are connected to two columns of 6 at X axis. Similarly, 1 and 3 are connected via the Y axis, 3 and 5 via the Z axis, 2 and 4 via the Y axis, and so on.

Definition 3: Two clusters situated on two concurrency matrices are said to be *strongly connected* when each row/column of a cluster is connected to a corresponding row/column of the other cluster.

In Fig. 13, the pairs of strongly connected clusters are 1 and 3, 1 and 6, 2 and 4, 2 and 5, 3 and 5, and 4 and 6.

Definition 4: A group of clusters composed of one each from the three concurrency matrices on a three-dimensional coordinate system is said to form a *strongly connected group,* if they are pairwise strongly connected.

It may be easily verified that it is not possible to form such a group in Fig. 13. For example, if we try to form such a group with say 1, 3, and 5, then, whereas the pairs (1, 3) and (3, 5) are strongly connected, the pair (1, 5) is not. On the other hand, in Fig. 14 the group of * clusters is pairwise strongly connected, and so is the group of $ clusters. This property of the clusters forming two pairwise strongly connected groups removes the conflict of encoding the three sets, and thereby renders them steerable.

The following theorem is then obvious.

Theorem 3: Three sets are steerable together with s bits if they are pairwise steerable and if their concurrency matrices drawn in a three-dimensional cartesian coordinate system exhibit 2^s disjoint groups of clusters that are pairwise strongly connected.

Fig. 15 shows how don't care concurrencies can be utilized to make the clusters pairwise strongly connected and thereby render the three sets steerable. The figure also shows how the method can handle three sets that may not be of equal length.

It may be mentioned that the method as described above cannot be extended to detect complete steerability among four or more sets. However, the principle of the method can be extended to any number of sets, if the data are represented in a generalized n-dimensional space rather than in a restricted three-dimensional space. But then the resultant algorithm, although computationally feasible, will be highly complex, as it is known that both the problems of microcode bit optimization and large-variable logic minimization are NP-complete [9], [10]. Due to another limitation imposed by the current semiconductor technology [11], it is also not desirable to apply bit steering to four or more sets. If a bit is to be shared by four or more sets, then the problems associated with large fan-out have to be handled, and extra hardware is required to provide additional amplification. Extra hardware means extra space on the chip and also an additional location where faults may occur. This defeats the very purpose for which we started bit steering. Hence, a simple algorithm which is adequate for three sets is preferable to a perfectly general but highly complex algorithm.

III. Conclusion

Although bit steering reduces the word width by only 1, 2, or 3 bits, this may prove to be quite critical in certain situations since most of the available ROM's have 8 bit words. If the microprogram is embedded in a VLSI chip, then the reduction of the word width assumes special importance as it results in a more compact implementation of the microprogram. This means a better utilization of the chip area. It may be observed that the technique of bit steering as described above does not require any extra hardware. Thus, we save one or more bits virtually at no cost. However, it is fair to ask here if the computation cost to detect and encode the steerable sets is too high. In our opinion it is not so. The concurrency matrix can be computed by a simple computer program. The detection and encoding of the steerable sets by the method as given in this correspondence is as simple as the map method of minimizing Boolean

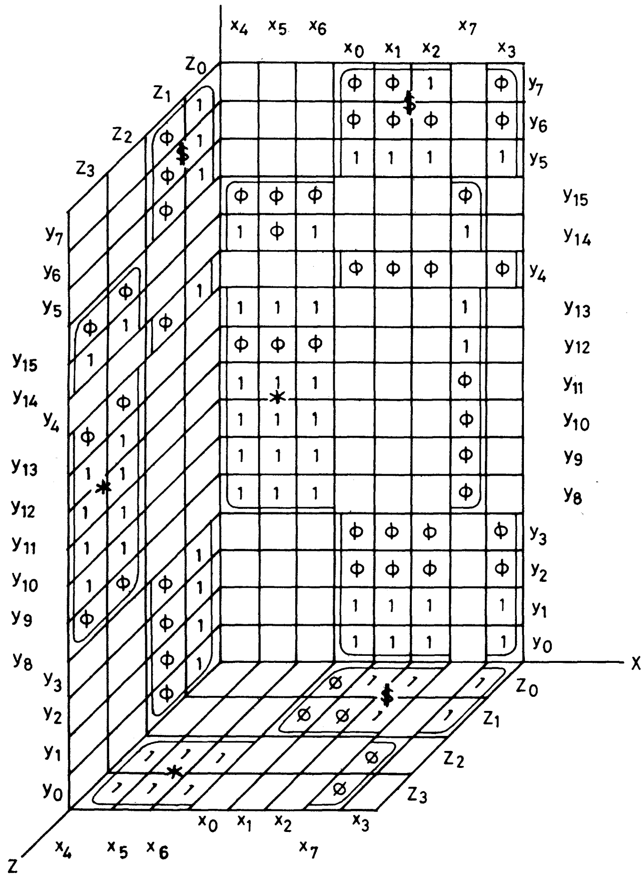


Fig. 15. Detection and encoding of three steerable sets of unequal lengths.

functions where the remarkable ability of the human mind to perceive patterns in pictorial representation of data is utilized [12]. Moreover, the method as expounded here is not only useful in ROM compaction, but can also be fruitfully applied in other areas where some resources are to be shared by a group of disjoint users working in a concurrency environment.

References

[1] S. J. Schwartz, "An algorithm for minimizing read-only memories for machine control," in *Proc. IEEE 9th Annu. Symp. Switching Automata Theory*, 1968, pp. 28–33.

[2] A. Grasselli and U. Montanari, "On the minimization of read only memories in microprogrammed digital computers," *IEEE Trans. Comput.*, vol. C-19, pp. 1111–1114, Nov. 1970.

[3] S. R. Das, D. K. Banerji, and A. Chattopadhay, "On control memory minimization in microprogrammed digital computers," *IEEE Trans. Comput.*, vol. C-22, pp. 845–848, Sept. 1973.

[4] T. Jayasri and D. Basu, "An approach to organizing microinstruction which minimizes the width of control store words," *IEEE Trans. Comput.*, vol. C-25, pp. 514–521, May 1976.

[5] J. L. Baer and B. Koyama, "On the minimization of the width of the control memory of microprogrammed processors," *IEEE Trans. Comput.*, vol. C-28, pp. 310–316, Apr. 1979.

[6] C. D. V. P. Rao and N. N. Biswas, "On the minimization of word width in the control memory of microprogrammed digital computers," *IEEE Trans. Comput.*, vol. C-32, pp. 863–868, Sept. 1983.

[7] T. Agerwala, "Microprogramming optimization: A survey," *IEEE Trans. Comput.*, vol. C-25, pp. 962–973, Oct. 1976.

[8] A. Mathialagan and N. N. Biswas, "Bit steering in the minimization of control memory in microprogrammed digital computers," *IEEE Trans. Comput.*, vol. C-30, pp. 144–147, Feb. 1981.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[10] R. K. Brayton, G. D. Hachtel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*. Boston, MA: Kluwer Academic, 1984.

[11] C. Mead and L. Conway, *Introduction to VLSI Systems*. Reading, MA: Addison-Wesley, 1980.

[12] F. J. Hill and G. R. Peterson, *Introduction to Switching Theory and Logical Design*. New York: Wiley, 1981.

Parallel Algorithms and Architectures for Optimal State Estimation

JAMES H. GRAHAM AND THADDEUS F. KADELA

***Abstract*— Optimal state estimation procedures, such as the Kalman–Bucy filter, require a high-speed parallel implementation to fully realize their potential. This correspondence first analyzes and restructures these equations based on an expression tree representation. The resulting equations are then simulated on three proposed SIMD architectures (rectangular systolic array, linear array, and quad-clustered tree array) and evaluated for speedup, efficiency, and utilization as compared to a known serial implementation. Results are presented which graphically show the design tradeoffs between these architectures.**

***Index Terms*— Kalman–Bucy filter, parallel processing, performance improvement, SIMD architectures, state estimation.**

I. Introduction

As alleged fundamental limits to technology are approached, emphasis is shifting more to improved system design [1], [2] to not only get higher performance, but to fully exploit the power of VLSI [3], [4]. Indeed, computer architects have been given an extra degree of freedom to form new structures with highly repeated subunits.

One immediate and obvious usage of parallel systems is for the computation of matrix-based equations for the analysis, simulation, and control of systems represented by state space formulations. While measuring the state variables for use in system control is a conceptually simple idea, it may be difficult if some of the states are not directly observable or if noise is present in the system. It is then necessary to estimate states, given information on the system's dynamics and disturbances, using a state estimator. One of the most popular state estimators is the recursive filter specified by Kalman and Bucy, which yields the optimal *a posteriori* state estimates for linear systems with Gaussian noise. This filter may be extended to handle nonlinear systems, and also performs quite well in many cases with non-Gaussian noise sources [5], [6].

Kalman–Bucy filters are used in a wide range of high performance control applications such as aircraft flight control, fire control, and air and sea navigation. A low-cost compact processor would extend the applications to include process control, robotics, and computer vision.

This correspondence presents a set of parallelized state estimation equations and three parallel processing architectures for high-speed implementation of these algorithms. The performance of these architectures are compared to each other and to a serial Kalman–Bucy filter implementation. Conclusions are presented about the relative strengths and weaknesses of each approach.

Manuscript received March 16, 1984; revised February 22, 1985.

J. H. Graham is with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180-3590.

T. H. Kadela was with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180-3590. He is now with IBM Corporation, Poughkeepsie, NY 12603.

0018-9340/85/1100-1061$01.00