THE BELL SYSTEM
TECHNICAL JOURNAL

DEVOTED TO THE SCIENTIFIC AND ENGINEERING
ASPECTS OF ELECTRICAL COMMUNICATION

Volume 51 September 1972 Number 7

On the Addressing Problem of Loop Switching

By L. H. BRANDENBURG, B. GOPINATH, and R. P. KURSHAN

(Manuscript received April 5, 1972)

R. L. Graham and H. O. Pollak recently proposed an addressing scheme to J. R. Pierce's Loop Switching Network–a network system which would exploit the one-way nature of much data transmission. Our paper provides a generalization of the scheme and proposes a new approach to certain unanswered questions.

I. INTRODUCTION

Recently, J. R. Pierce[1] proposed a data communication system concept whereby subscribers are placed on interconnected "loops". A message originating in one loop and destined for a recipient in another loop must find its way through the network of interconnections. Such a system is potentially efficient for data communication because it exploits the one-way nature of many such transmissions by eliminating the need to set up an entire line between two parties in the conventional manner.

The realization of this potential efficiency depends in part on the existence of an efficient routing scheme. Such a scheme is discussed by Graham and Pollak in Ref. 2. Briefly, they assigned to each loop a ternary address; a message destined for a certain loop is tagged with the address of its destination, and upon entering a loop, a computation involving that loop address and the message destination address yields the distance of the message from its destination. The message then moves in such a way as to decrease this distance. In particular, it is assumed that there are n loops and each address is a sequence of N (fixed) 0's, 1's and d's; let H be the $n \times N$ matrix the ith row H_i of which is the address of the ith loop. The addresses are chosen in such a way that if D_{ij} is the distance between the ith and jth loop (by D_{ij} is meant the fewest number of loops which it is necessary to cross to to get from the ith to the jth), then

$$D_{ij} = \sum_{k=1}^{N} (H_{ik} \oplus H_{jk}), \quad \text{where } a \oplus b = \begin{cases} 1 \text{ if the set } \{a, b\} = \{0, 1\} \\ 0 \text{ otherwise} \end{cases}$$

(this is an "extended" Hamming distance between H_i and H_j).

Alternatively, the problem may be considered in graph-theoretic terms: associate with the loop network a graph the vertices of which correspond to the loops and the edges of which correspond to direct connections between two loops. Then D_{ij} is the minimal number of edges in a path between the ith and jth vertices.

Central to the efficiency of the above scheme was the possibility of keeping N small; if G is a connected graph on n vertices, then Graham and Pollak showed that an address matrix H with corresponding N–call it N_H–could always be found to satisfy $N_H \leqq (n-1) \times \operatorname{diam} G$ ($\operatorname{diam} G = \max_{i,j} D_{ij}$). Unfortunately, $(n-1) \times \operatorname{diam} G$ can be unpleasantly large. However, they conjectured that, in fact, for any connected graph G, one could always find an H matrix such that $N_H \leqq n - 1$. This was proved in the case of complete graphs, trees and cycles. Furthermore, defining $N(G) = \min_H N_H$, they showed that, $N(G) = n - 1$ for G a cycle on an odd number of vertices, a complete graph or a tree; and $N(G) = n/2$ for G a cycle on an even number of vertices. In any case, it was shown that $N(G) \geqq \max \{n^+, n^-\}$ where $n^+(n^-)$ is the number of positive (negative) eigenvalues of the distance matrix D with elements D_{ij} defined above.

For a graph G (which we will always assume to be connected) referred to implicitly or explicitly, let us reserve the letter D to denote the distance matrix of G, and n to denote the number of vertices of G. Then the conjecture of Ref. 2 becomes

$$N(G) \leqq n - 1. \tag{1}$$

Incidentally, this conjecture is equivalent to the following conjecture: that any graph G on n nodes can be embedded in a "factor" F of an $(n-1)$-cube in such a way that the distance matrix for G is a submatrix of that for F; by a "factor" of a cube C is meant a graph F defined as follows: Let $\mathfrak{F} = \{f_1, \cdots, f_k\}$ be a set of pairwise disjoint faces (of various dimensions) of C; let V be the set of vertices of C appearing in no f_i. Then the set of vertices of F is $V \cup \mathfrak{F}$ (i.e., the vertices V of C along with the faces of $\mathfrak{F}$). There is an edge between two vertices x, $y \in V \cup \mathfrak{F}$ if:

(i) $x, y \in V$ and x is connected to y by an edge in C;
(ii) $x, y \in \mathfrak{F}$ and some vertex of (the face) x is connected by an edge to some vertex of (the face) y;
(iii) $x \in V$, $y \in \mathfrak{F}$ and some vertex of (the face) y is connected by an edge to x.

More descriptively, F is formed from C by shrinking each f_i to a vertex.

In this paper, we discuss the conjecture (1) by formulating the addressing problem in matrix-algebra terms and then present an algebraic generalization of this that immediately yields an addressing scheme more efficient than that of Ref. 2.

Specifically, we show that a graph G has an $n \times N$ address matrix H if and only if there exist $n \times N$ binary (0, 1)-valued matrices A and B such that

$$D = AB^{t} + BA^{t} \tag{2}$$

(where t means "transpose"). Using this, we expand the class of graphs known to satisfy (1). Then we rewrite (2) as

$$D = [A \vdots B]\begin{bmatrix} B^{t} \\ \cdots \\ A^{t} \end{bmatrix} \tag{3}$$

(the product of the two pairs of concatenated matrices). Note that the problem of finding $n \times N$ matrices A and B satisfying (2) [and (3)] can be generalized to the problem of finding $n \times 2N$ binary (0, 1)-valued matrices P and Q satisfying

$$D = PQ^{t}. \tag{4}$$

Clearly, any decomposition (2) yields the corresponding decomposition (4). On the other hand, any decomposition of the form (4) with P and Q (binary-valued) $n \times L$ matrices gives rise to a binary addressing of length L which, as in Ref. 2, will enable a message to compute its distance from its destination. In particular, the ith vertex is tagged with Q_i, the ith row of Q, while a message destined for the ith vertex is tagged with P_i, the ith row of P. Then the message, at vertex j, makes the computation $P_i Q_j^t = D_{ij}$ and determines its distance from its destination. This computation involves summing L products, each of the form $a \cdot b$ where $a, b \varepsilon \{0, 1\}$, as compared to summing N sums of the form $a \oplus b$ defined above where $a, b, \varepsilon \{0, 1, d\}$. Note that two sets of addresses are required: the set $\{Q_i\}$ is stored in the network, and members of the set $\{P_i\}$ are tagged onto messages. But users of the network only need a directory of the P_i's.

Hence, assuming that the addresses of Ref. 2 are to be transmitted in binary, the decomposition (4) gives rise to no larger, and in certain cases, smaller addresses than those derived in Ref. 2. If $L(G)$ is the minimum value of L over all such decompositions (4), then corresponding to the conjecture (1) [and a necessary consequence, if (1) is true] we conjecture that

$$L(G) \leqq 2(n - 1). \tag{5}$$

In any case, we show that

$$L(G) \leqq \sum_i \max_j D_{ij} \leqq n(\text{diam } G.) \tag{6}$$

This improves on the upper bound derived in Ref. 2, which after translation of addresses into the form of (3), gives $L \leqq 2(n - 1)$ diam G.

Using the decomposition (4) which yields (6), we discuss a secondary coding of the addresses which reduces the number of bits actually transmitted; by using this decomposition rather than one with possibly smaller L, we achieve an addressing scheme the addresses of which are very easy to compute and which is hence amenable to continual updating of the loop network.

II. EQUIVALENT FORMULATIONS OF THE GRAHAM–POLLAK SCHEME

Here we present three theorems: the first characterizes in various terms the property that a graph G admits an $n \times N$ address matrix H, the second and third present some sufficient (but not necessary) conditions for (1) to hold.

We will call a matrix *binary-valued* if its entires are either 0 or 1. Clearly, G admits an $n \times N$ H if and only if $N \geqq N(G)$.

Theorem 1: Let D be the distance matrix of a graph G on n vertices. Then the following conditions are equivalent:

(i) $N \geqq N(G)$;

(ii) There exist binary-valued $n \times N$ matrices A and B such that $D = AB^t + BA^t$;

(iii) There exist $n \times N$ matrices C and E, C binary-valued and E with entries 0, ± 1 *such that* $C_{ij} = | E_{ij} |$ *and* $2D = CC^{t} - EE^{t}$.

(iv) There exist $n \times n$ matrices F_i, $i = 1, \cdots, N$ with each F_i binary valued, symmetric, rank $\leqq 2$, *and* $D = \sum_{i=1}^{N} F_i$.

Proof: (*i*) ⇒ (*ii*): Since $N \geqq N(G)$ there exists for G an $n \times N$ address matrix H. By construction we know that D_{ij} is the extended Hamming distance between H_i and H_j, which is in fact the number of positions where H_i and H_j differ without either being a d. Define for each H_i a binary-valued N-vector (i.e., $1 \times N$ matrix) A_i by $A_{ik} = 1$ if $H_{ik} = 1$ and $A_{ik} = 0$ otherwise, and define a binary-valued N-vector B_i by $B_{ik} = 1$ if $H_{ik} = 0$ and $B_{ik} = 0$ otherwise. It is then clear that D_{ij} is the sum of inner products $\langle A_i , B_j \rangle + \langle B_i , A_j \rangle$, i.e., $D_{ij} = A_i(B_j)^{t} + B_i(A_j)^{t}$. We thus obtain $D = AB^{t} + BA^{t}$, where $A[B]$ is the $n \times N$ matrix whose ith row is $A_i[B_i]$.

(*ii*) ⇒ (*i*): Suppose we have $n \times N$ binary-valued matrices A and B such that $D = AB^{t} + BA^{t}$. For all i, since $D_{ii} = 0$, $A_i(B_i)^{t} = 0 = B_i(A_i)^{t}$ and A_i and B_i can hence have no 1's in common positions. It should be clear from the converse case above that by defining H as

$$H_{ij} = \left\{ \begin{matrix} 1 & \text{if} & A_{ij} = 1 \\ 0 & \text{if} & B_{ij} = 1 \\ d & \text{if} & A_{ij} = B_{ij} = 0 \end{matrix} \right\},$$

we obtain the desired address matrix.

(*ii*) ⇒ (*iii*): Given (*ii*), define matrices C and E as $C = A + B$ and $E = A - B$. Since A and B have no common ones, C is binary-valued with zeros only in those positions where A and B have common zeros and one elsewhere. Moreover, E has values 0, ±1 and has zeros only in those positions where A and B have common zeros and is ±1 elsewhere. The definition of C and E immediately leads to verification of the relation

$$2D = CC^{t} - EE^{t}.$$

(*iii*) ⇒ (*ii*): Given (*iii*), define $A = \frac{1}{2}(C + E)$ and $B = \frac{1}{2}(C - E)$. Clearly, A and B are binary-valued (with no ones in common positions) and (*ii*) follows directly from (*iii*).

(*i*) ⇒ (*iv*): Let H^{i}, $i = 1, \cdots N$ be the ith column of matrix H and let F_i be the $n \times n$ matrix of Hamming distances between the rows (actually sequences of length one) of H^{i}. Then clearly

$$D = \sum_{i=1}^{N} F_i$$

and we must show that each F_i has the required structure. Suppose that H^i has n_1 0's, n_2 1's, and n_3 d's. Reorder the elements of H^i so that all of the 0's appear first, followed by all of the 1's. Then, the rows and columns of F_i can be reordered (symmetrically) to yield

$$\begin{array}{c|c|c|c|}
 & n_1 & n_2 & n_3 \\
 & 0\cdots0 & 1\cdots1 & 0\cdots0 \\
n_1 & \vdots\quad\vdots & \vdots\quad\vdots & \vdots\quad\vdots \\
 & 0\cdots0 & 1\cdots1 & \cdots\cdots \\
\hline
 & 1\cdots1 & 0\cdots0 & \cdots\cdots \\
n_2 & \vdots\quad\vdots & \vdots\quad\vdots & \vdots\quad\vdots \\
 & 1\cdots1 & 0\cdots0 & \cdots\cdots \\
\hline
 & 0\ \cdots & \cdots\cdots & 0\cdots0 \\
n_3 & \vdots\quad\vdots & \vdots\quad\vdots & \vdots\quad\vdots \\
 & 0\ \cdots & \cdots\cdots & 0\cdots0
\end{array} = F_1 \tag{7}$$

Clearly, F_i is binary-valued, symmetric, and rank two, (or rank zero if n_1 or $n_2 = 0$) and these properties are invariant with respect to symmetric permutation of rows and columns.

(iv) $\Rightarrow$ *(i)*: Let $D = \sum_{i=1}^{N} F_i$ with each F_i satisfying the conditions of the theorem. We show that by appropriate permutation of rows and columns, F_i can be partitioned as in (7). Note first that since diagonal elements of D are zero, the same must be true of each F_i. Hence rank $(F) \leqq 2$ implies either rank $(F_i) = 2$ or $F_i \equiv 0$ since F_i is symmetric.

Now, let F be any binary-valued, symmetric, rank two, $n \times n$ matrix with zeros on the main diagonal. Let $e_1, \cdots e_n$ represent the diagonal elements of F. (The value of each e_i is 0). Two elements e_i and e_j will be defined as equivalent, $e_i \sim e_i$ if $F_{ij} = 0$. We now show that under the assumptions concerning F, $\sim$ is a true equivalence relation with respect to the e_i's that do not index a row or column of zeros in F[†]. Clearly, $e_i \sim e_i$, $i = 1, \cdots n$, and $e_i \sim e_j$ implies $e_j \sim e_i$. In order to

[†] The equivalence class method of proof was suggested to the authors by H. S. Witsenhausen.

prove transitivity, let us (temporarily) remove from F all rows and columns consisting entirely of zeros. Suppose now that $e_i \sim e_j$, $e_j \sim e_k$, but that $F_{ik} = 1$. Then F has the form

$$\begin{array}{c|c|c|c|c|c|c|c|}
 & & i & j & k & & m & \\
\hline
 & & & & & & & \\
\hline
i & & 0 & 0 & 1 & & y & \\
\hline
j & & 0 & 0 & 0 & & 1 & \\
\hline
k & & 1 & 0 & 0 & & & \\
\hline
 & & & & & & & \\
\hline
m & & & 1 & x & & 0 & \\
\hline
 & & & & & & & \\
\end{array}$$

Since rows and columns of zeros have been removed, $F_{jm} = F_{mj} = 1$ for some m. The unspecified values of F_{mk} and F_{im} are denoted by x and y respectively. Clearly the 3×3 submatrix of F

$$\begin{bmatrix} 0 & 1 & y \\ 0 & 0 & 1 \\ 1 & x & 0 \end{bmatrix}$$

consisting of columns j, k, m and rows i, j, m, is nonsingular regardless of how x and y are specified, thus contradicting the rank two condition. Hence, $\sim$ is transitive, and F can be permuted in such a way that all diagonal blocks consist entirely of zeros, and all off-diagonal blocks consist entirely of ones. This partitioned matrix can then be bordered on the right and bottom with the previously removed rows and columns of zeros. The resulting matrix then resembles (7) except that we must discount the possibility of having more than two equivalence classes. However, it is obvious from the partitioned structure of F that the number of equivalence classes equals rank (F) which is two by assumption. To each F_i we can thus associate a column vector H^i consisting of 0's, 1's and d's. We place d's in positions of H^i corresponding to the elements e_k that index rows and columns of 0's in F_i. We place 0's in positions of H^i corresponding to the elements e_k that determine one of the equivalence classes, and we place 1's in the remaining positions of H^i. The matrix of Hamming distances associated with H^i is thus equal to F_i. If rank $(F_i) = 0$ for some i, then $F_i = 0$ and the corresponding column H^i can be chosen to consist entirely of d's. This completes the proof.

We now present a method for joining address matrices of graphs to form an address matrix of their union. Suppose a graph G contains (connected) subgraphs G_1 and G_2 such that $G = G_1 \cup G_2$ and $G_1 \cap G_2$ are exactly a vertex of G (i.e., G is found by connecting the two graphs G_1 and G_2 at some vertex). Then we will say that G is *separable into G_1 and G_2*.

Theorem 2: Suppose the graph G is separable into the subgraphs G_1 and G_2. Then $N(G) \leqq N(G_1) + N(G_2)$.

Proof: Let n_i be the number of vertices of G_i and let H_i be an $n_i \times N(G_i)$ address matrix for G_i for $i = 1, 2$. We may assume that $G_1 \cap G_2$ is the n_1th vertex of G_1 and the 1st vertex of G_2. Define H as

$$H = \begin{array}{r} n_1\left\{\vphantom{\begin{array}{c}H_1\\ \\ \end{array}}\right. \\ \\ \end{array}\left[\begin{array}{c|c} \overbrace{\qquad}^{N(G_1)} & \overbrace{\qquad}^{N(G_2)} \\ H_1 & J_2 \\ \cline{2-2} & \\ \hline J_1 & H_2 \end{array}\right]\begin{array}{l} \\ \\ \left.\vphantom{\begin{array}{c}H_2\\ \\ \end{array}}\right\}n_2 \end{array}$$

where $J_1(J_2)$ is the $(n_2 - 1) \times N(G_1)((n_1 - 1) \times N(G_2))$ matrix each row of which is the n_1th (1st) row of $H_1(H_2)$. Clearly H is an address matrix for G with addresses of length $N(G_1) + N(G_2)$.

Corollary: If the graph G is separable into subgraphs each of which satisfies the conjecture (1), then G also satisfies (1): i.e., $N(G) \leqq n - 1$.

Proof: With notation as above, if $N(G_1) \leqq n_1 - 1$ and $N(G_2) \leqq n_2 - 1$ then $N(G) \leqq n_1 + (n_2 - 1) - 1 = n - 1$.

Clearly, this corollary can be extended to any number of subgraphs.

We will now present a class of graphs for which $N(G) \leqq n - 1$. Call a linear tree (i.e., $D_{ij} = | i - j |$) contained in a graph G a *spine* if each vertex of G is either a vertex of the linear tree or else distance one away from the linear tree. Clearly, any diameter two graph (i.e., the maximum distance between each pair of vertices is two) admits a spine; any maximal linear tree will do. It can also be shown that any diameter three graph admits a spine, while, on the other hand, there exists a 14-point nonseparable diameter four graph admitting no spine.

If L is a spine for a graph G, label the vertices of L consecutively: v_{10}, v_{20}, v_{30} $\cdots$ and label each vertex v of G not on L v_{ij}, $j = 1, 2, \cdots$ where $i = \min \{k \mid d(v, v_{k0}) = 1\}$.

Theorem 3: A sufficient condition for $N(G) \leqq n - 1$ *is that G admit a spine L satisfying*

EITHER (1) $d(v_{je}, v_{ik}) \geqq d(v_{j0}, v_{ik}) - \gamma_k$ for $i = 1, 2, \cdots ; j > i$; $k, e = 0, 1, \cdots$ *(when defined) where* $\gamma_k = 0$ *if* $k = 0$ *and* 1 *if* $k \neq 0$.

OR (2) *There are no six vertices* x_1, y_1, y_2, x_2, x_3, x_4, *distinct except possibly* $y_1 = y_2$, *and increasing from left to right in the lexicographic order of the vertices' double-indexing, with each* x_i *a vertex of L and such that:*

$$\begin{aligned} i)\ & d(x_2, x_1) = d(y_2, x_1) + 1 + \gamma(y_2), \\ ii)\ & d(x_4, x_2) = d(x_3, x_2) + 1, \\ iii)\ & d(x_4, x_1) = d(x_3, x_1) + 1, \\ iv)\ & d(x_3, y_1) = d(x_4, y_1) + 1 + \gamma(y_1), \end{aligned}$$

where $\gamma(y_i) = 0$ *if* y_i *is a vertex of L and* 1 *otherwise.*

[*Note:* (1) $\Rightarrow$ (2), *and there exist even weaker (but more complicated) sufficient conditions than* (2); *however, no graph is known to the authors which admits a spine and fails to satisfy* (1) *for some spine*].

Corollary 1: For any diameter 3 *graph G admitting a spine of* 3 *or fewer vertices,* $N(G) \leqq n - 1$.

Corollary 2: For any diameter 2 *graph G admitting a spine L such that, either no four vertices of L form a* 4*-cycle in the original graph, or else L has at most* 4 *points, we have* $N(G) \leqq n - 1$; *this includes all complete bipartite graphs.*

Proof: Both statements follow directly from condition (2) of the theorem.

Proof of Theorem: Suppose we have a graph which admits a spine; double-indexing the vertices as above, the associated distance matrix D will have the form

$$D = \begin{array}{c|c|c|c|c|c|c|c|c}
 & v_{10} & v_{11}\; v_{12}\; \cdots & v_{20} & v_{21}v_{22}\cdots & v_{30} & v_{31}v_{32}\cdots & v_{40} & v_{41}v_{42} \\ \hline
v_{10} & 0 & 1\;\;1\cdots 1 & 1 & \leqq 2 & & \leqq 3 & & \leqq 4 \\ \hline
\begin{matrix} v_{11} \\ v_{12} \\ \vdots \\ \\ \end{matrix} & \begin{matrix} 1 \\ 1 \\ \vdots \\ 1 \end{matrix} & \begin{matrix} 0 & & \leqq 2 \\ & \ddots & \\ \leqq 2 & & 0 \end{matrix} & & \leqq 3 & & \leqq 4 & & \leqq 5 \\ \hline
v_{20} & 1 & & 0 & 1\;\;1\cdots 1 & 1 & \leqq 2 & & \leqq 3 \\ \hline
\begin{matrix} v_{21} \\ v_{22} \\ \vdots \end{matrix} & \leqq 2 & \leqq 3 & \begin{matrix} 1 \\ 1 \\ 1 \end{matrix} & \begin{matrix} 0 & & \\ & \ddots & \leqq 2 \\ \leqq 2 & & 0 \end{matrix} & & \leqq 3 & & \leqq 4 \\ \hline
v_{30} & & & 1 & & 0 & 1\;\;1\cdots 1 & 1 & \leqq 2 \\ \hline
\begin{matrix} v_{31} \\ v_{32} \\ \vdots \\ \\ \end{matrix} & \leqq 3 & \leqq 4 & \leqq 2 & \leqq 3 & \begin{matrix} 1 \\ 1 \\ \vdots \\ 1 \end{matrix} & \begin{matrix} 0 & & \leqq 2 \\ & \ddots & \\ \leqq 2 & & 0 \end{matrix} & & \leqq 3 \\ \hline
v_{40} & & & & & 1 & & 0 & 1\;\;1\cdots \\ \hline
\begin{matrix} v_{41} \\ v_{42} \\ \\ \end{matrix} & \leqq 4 & \leqq 5 & \leqq 3 & \leqq 4 & \leqq 2 & \leqq 3 & \begin{matrix} 1 \\ 1 \\ \vdots \end{matrix} & \begin{matrix} 0 & & \leqq 2 \\ & \ddots & \\ \leqq 2 & & \ddots \end{matrix} \\
 & & & & & \vdots & & &
\end{array} \cdots$$

Conditions (1) and (2) both place certain restrictions on the extent of non-monotonicity of any column of D below the main diagonal; (1) says that the columns under each v_{i_0} shall be monotonically increasing and that the other columns shall contain decreases of at most one unit per square (roughly speaking). Condition (2) allows a greater degree of non-monotonicity but excludes an unmanageable combination of such. It is not difficult to see that (1) $\Rightarrow$ (2).

Now assume that (2) is satisfied. We will construct $n \times (n - 1)$ matrices A and B satisfying $D = AB^t + BA^t$. Roughly speaking, one can think of A as being superimposed on the $n \times (n-1)$ matrix D' remaining after removing from D its first column; we will use the notation $[k_1, k_2, \cdots]$ to indicate that the designated spot in A shall be a 1 if the associated spot in D' is either k_1 or k_2 or $\cdots$; all non-1 entries in A are 0. The matrix B^t will basically be an $n-1 \times n-1$ identity matrix with a concatenated first column of zeros; 1's will then be added above the existing diagonal of 1's for the purpose of "repeating" the associated column of A.

First, consider the case in which (1) holds. In this case (it is easily checked) AB^t can have the form

$$\underbrace{\begin{array}{c:c|c:c|c:c|c}
\hline
1\ \ 1\cdots 1 & 1 & {[2]} & & {[3]} & & {[4]} \\ \hline
\begin{matrix}0 & & 1\\ & \ddots & \\ {[2]} & & 0\end{matrix} & \begin{matrix}1\\1\\ \vdots\end{matrix} & {[3]} & & {[4]} & & {[5]} \\ \hdashline
{[2,3]} & \begin{matrix}0\\0\\0\\ \vdots\\0\end{matrix} & \begin{matrix}1 & 1\cdots & 1\\ 0 & & 1\\ & \ddots & \\ {[2]} & & 0\end{matrix} & \begin{matrix}1\\1\\1\\ \vdots\\1\end{matrix} & \begin{matrix}{[2]}\\ \\ {[3]}\end{matrix} & & \begin{matrix}{[3]}\\ \\ {[4]}\end{matrix} \\ \hdashline
 & 0 & & 0 & 1\ \ 1\cdots 1 & 1 & {[2]} \\ \hline
{[2,3,4]} & \begin{matrix}0\\0\\ \vdots\\0\end{matrix} & {[2,3]} & \begin{matrix}0\\0\\ \vdots\\0\end{matrix} & \begin{matrix}0 & & 1\\ & \ddots & \\ {[2]} & & 0\end{matrix} & \begin{matrix}1\\1\\ \vdots\\1\end{matrix} & {[3]} \\ \hdashline
 & 0 & & 0 & & 0 & 1\ \ 1\cdots \\ \hline
{[2,\cdots,5]} & \begin{matrix}0\\0\\ \vdots\end{matrix} & {[2,3,4]} & \begin{matrix}0\\0\\ \vdots\end{matrix} & {[2,3]} & & \begin{matrix}0 & & 1\\ & \ddots & \\ {[2]} & & \end{matrix} \\ \hline
\end{array}}_{A}$$

$$
\underbrace{
\begin{array}{c|c|c|c|c|c|c|c}
\begin{matrix}0\\0\\\vdots\\0\end{matrix} & \begin{matrix}1 & & & 0\\ & 1 & & \\ 0 & & \ddots & \\ & & & 1\end{matrix} & & 0 & & 0 & & 0 \\
\hline
0 & & 1 & 1\ \ 1 \cdots 1 & 1 & 1\ \ 1 \cdots 1 & 1 & 1\ \ 1 \cdots \\
\hline
\begin{matrix}0\\0\\\vdots\\0\end{matrix} & 0 & & \begin{matrix}1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1\end{matrix} & & & & \\
\hline
0 & & & & 1 & 1\ \ 1 \cdots 1 & 1 & 1\ \ 1 \cdots \\
\hline
\begin{matrix}0\\0\\\vdots\\0\end{matrix} & 0 & & 0 & & \begin{matrix}1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1\end{matrix} & & \\
\hline
0 & & & & & & 1 & 1\ \ 1 \cdots \\
\hline
\begin{matrix}0\\0\\ \\ \end{matrix} & 0 & & 0 & & & & \begin{matrix}1 & & \\ & 1 & \\ & & \ddots\end{matrix}
\end{array}
}_{B^{t}}
$$

$$
AB^{t} = \left.\begin{array}{c|c:c|c:c|c:c|c}
0 & 1 \quad 1 \cdots 1 & 1 & 1\to 1,\ 2\to 2 & & 1\to 1,\ 2\to 2,\ 3\to 3 & & \\
\hline
\begin{matrix}0\\0\\\vdots\\0\end{matrix} & \begin{matrix}0 & & & 1\\ & \cdot & & \\ & & \cdot & \\ 2\to 1 & & & 0\end{matrix} & \begin{matrix}1\\1\\\vdots\\1\end{matrix} & \begin{matrix}1,\ 2\to 1\\ 3\to 2\end{matrix} & & \begin{matrix}1,\ 2\to 1\\ 3\to 2\\ 4\to 3\end{matrix} & & \\
\hdashline
0 & & 0 & 1 \quad 1 \cdots 1 & 1 & & & \\
\hline
\begin{matrix}0\\0\\\vdots\\0\end{matrix} & 2,\ 3\to 1 & \begin{matrix}1\\1\\\vdots\\1\end{matrix} & \begin{matrix}0 & & & 1\\ & \cdot & & \\ & & \cdot & \\ 2\to 1 & & & 0\end{matrix} & \begin{matrix}1\\1\\\vdots\\1\end{matrix} & \begin{matrix}1,\ 2\to 1\\ 3\to 2\end{matrix} & & \\
\hdashline
0 & & 1 & & 0 & 1 \quad 1 \cdots 1 & 1 & \\
\hline
\begin{matrix}0\\0\\\vdots\\0\end{matrix} & 2,\ 3,\ 4\to 1 & & 2,\ 3\to 1 & \begin{matrix}1\\1\\\vdots\\1\end{matrix} & \begin{matrix}0 & & & 1\\ & \cdot & & \\ & & \cdot & \\ 2\to 1 & & & 0\end{matrix} & \begin{matrix}1\\1\\\vdots\\1\end{matrix} & \\
\hdashline
0 & & & & 1 & & 0 & 1 \quad 1 \cdots \\
\hline
\begin{matrix}0\\0\\ \\ \end{matrix} & & & & & & \begin{matrix}1\\1\\\vdots\end{matrix} & \begin{matrix}0 & & \\ & \cdot & \\ & & \cdot\end{matrix}
\end{array}\right.
$$

where $k_1k_2 \cdots \to l$ means that whenever k_1 or k_2 or $\cdots$ appears in D, then l appears in AB^{t}. It is then clear that $AB^{t} + (AB^{t})^{t} = D$.

The general case of (2) now follows in an analogous fashion; the matrix B^{t} will be as above, except that certain of the off-diagonal 1's will be replaced by 0's. The matrix A will be as above, except that the bracket notation will have to be written for each column above the diagonal, each row below the diagonal and the entries in the brackets will have to be altered to meet the needs of the situation. It can be shown that given the conditions of (2), this can actually be done.

III. EXTENSION TO THE PQ^t SCHEME

As explained in the introduction, an alternative way of looking at the addressing problem is to determine the smallest L for which there are $n \times L$ binary-valued matrices P and Q such that $D = PQ^t$. This formulation also allows for a possibility where D is nonsymmetric. A nonsymmetric D can arise if the given graph is directed, or can arise if the elements of D are chosen so as to obtain a desirable routing scheme not strictly dependent on the actual distances between vertices. In these terms, the analog of the Graham and Pollak conjecture (1) is that $L(G) \leqq 2(n - 1)$ (where $L(G)$ is the smallest L). We know this is true when $N(G) \leqq n - 1$ since in any case (from (3) and (4))

$$L(G) \leqq 2N(G). \tag{8}$$

In Theorem 4 we will show that $L(G) \leqq \sum_i \max_j D_{ij} \leqq n$ diam G.

On the other hand, we have the lower bounds

$$L(G) \geqq \text{rank } D \tag{9}$$

and

$$L(G) \geqq 2 \text{ diam } G. \tag{10}$$

The bound (9) follows since $L(G) \geqq$ rank $P \geqq$ rank D. To see (10), find i and j such that $D_{ij} =$ diam G. Then $P_i(Q_j)^t =$ diam $G = P_j(Q_i)^t$. Hence P_i and Q_j have diam G 1's in common and the same is true for P_j and Q_i. But $D_{ii} = 0$, so that in each spot that P_i has a 1, Q_i has a zero. Thus, Q_i must contain at least diam G 1's and diam G 0's.

Theorem 4: $L(G) \leqq \sum_i \max_j D_{ij} \leqq n$ diam G

Proof: Let $s_i = \max_j D_{ij}$ and $r_k = \sum_{i=1}^{k} s_i$ with $r_o = 0$. It is easy to see that the following construction for P and Q gives an address of length r_n. For every k, the kth row of P, P_k is zero everywhere except in positions $r_{k-1} + 1$ to r_k, where it is one. For every j, the jth row of Q, Q_j is divided into cells, the kth cell defined as positions $r_{k-1} + 1$ to r_k. In the kth cell is placed exactly D_{kj} 1's with the rest being equal zero. Obviously $P_kQ_j^t$ will then be the number of 1's in the kth cell of Q_j ., i.e., D_{kj}. Notice that this proof is not dependent on D being the distance matrix of a graph; the theorem is true for any nonnegative integer-valued matrix D.

A comparison of (3) and (4) suggests that the inequality (8) can be strict. This is illustrated in several of the following special cases where we show, in particular, that if G is a cycle with n odd or a complete graph, then $L(G) = N(G) + 1$, and for certain trees, $L(G) \leqq \frac{3}{2}N(G)$.

(i) Complete Graphs

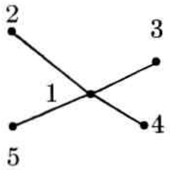
For a complete graph, all off-diagonal elements of D are 1's. Let $P = I$, the n-dimensional identity matrix, and let $Q = D$. Then obviously $D = ID = PQ^t$, and in light of (9), since rank $D = n$, this factorization is minimal.

For a binary-valued matrix X, define $\bar{X}$ to be X with all the 1's changed to 0's and all the 0's changed to 1's. Let $X^* = \bar{X}^t$. Then the decomposition for the complete graph above has the special form $D = PP^*$. This property has important implications which will be discussed later.

(ii) Trees

In Ref. 2 (Theorem 3), it was shown that for any tree G, $N(G) = n - 1$, and in fact, the address matrix H could be chosen to be binary-valued–without d's. Hence, by (3), for any tree G, $L(G) \leqq 2(n - 1)$. For G a linear tree, i.e., $D_{ij} = | i - j |$, in fact $L(G) = 2(n - 1)$ by (10), and hence for an arbitrary graph G there is no general upper bound for $L(G)$ lower than $2(n - 1)$. However, we shall show that for some trees $L(G) < 2(n - 1)$. Also, since the address matrix for a tree can be chosen to have no d's, it is clear from (3) and Theorem 1 that the distance matrix for a tree can be written as $D = PP^*$.

Now let G be a star on five vertices.



Then

$$D = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 2 & 2 & 2 \\ 1 & 2 & 0 & 2 & 2 \\ 1 & 2 & 2 & 0 & 2 \\ 1 & 2 & 2 & 2 & 0 \end{bmatrix}.$$

Let

$$P = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix},$$

and

$$Q^t = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Then $D = PQ^t$ (note that we no longer have $Q^t = P^*$) and $L(G) \leqq 6 < 8 = 2N(G)$. Conversely, we show $L(G) \geqq 6$. To see this, suppose there exists P and Q of order 5×5 such that $D = PQ^t$. Then for D_1, the submatrix of D defined below we have

$$D_1 = \begin{bmatrix} 0 & 2 & 2 & 2 \\ 2 & 0 & 2 & 2 \\ 2 & 2 & 0 & 2 \\ 2 & 2 & 2 & 0 \end{bmatrix} = RS^t = SR^t,$$

where R and S are the obvious submatrices of P and Q, respectively. Observe first that each row of R and S must contain at least two 1's. Also, at least one of the rows of R or S must contain at most two 1's, since if this were not the case, the diagonal elements of D_1 would not vanish. Since the roles of R and S are interchangeable, we assume without loss of generality that the first two elements of the first row R_1 of R are 1's and the rest are zeros. Now since

$$R_1 S_j^t = 2, \qquad j = 2, 3, 4,$$

we have

$$S_{j1} = S_{j2} = 1, \qquad j = 2, 3, 4;$$

thus

$$R_{j1} = R_{j2} = 0 \qquad j = 2, 3, 4$$

because

$$R_jS_j^t = 0 \quad \text{for all } j.$$

Now, at least two of the numbers R_{2j} , $j = 3, 4, 5$ must be 1, and hence, at most one of the numbers S_{2j} , $j = 3, 4, 5$ can be 1, which implies $R_jS_2^t = \leqq 1$, which contradicts the condition that $R_3S_2^t = 2$. Thus $L(G) > 5$.

If G is a star with $n = 4k + 1$, then $L(G) \leqq 6k < 8k = 2N(G)$. This follows from the above example and a result analogous to Theorem 2: viz., if G is separable into G_1 and G_2 then $L(G) \leqq L(G_1) + L(G_2)$.

(iii) Cycles–n odd

Suppose G is a cycle with distance matrix D and n odd. Let $m = (n - 1)/2$ and let L_m be the distance matrix for the linear tree on m vertices. Let A be the $m \times m$ triangular matrix with 1's on and below the diagonal, and let U be the $m \times m$ matrix all the entries of which are 1:

$$A = \begin{bmatrix} 1 & & & & \\ & 1 & & 0 & \\ & & 1 & & \\ & 1 & & \ddots & \\ & & & & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & & & \vdots \\ 1 & \cdot & \cdots & 1 \end{bmatrix}.$$

Then

$$D = \left[\begin{array}{c|c|c} L_m & \begin{matrix} m \\ m-1 \\ \vdots \\ 2 \\ 1 \end{matrix} & mU - L_m + A^* \\ \hline m \quad m-1 \cdots 2 \quad 1 & 0 & 1 \quad 2 \cdots m-1 \quad m \\ \hline mU - L_m + \bar{A} & \begin{matrix} 1 \\ 2 \\ \vdots \\ m-1 \\ m \end{matrix} & L_m \end{array}\right].$$

Define

$$P = \left[\begin{array}{c|c|c} A & \begin{matrix} 0 \\ \vdots \\ 0 \end{matrix} & A^{t} \\ \hline 1 \cdots 1 & 1 & 0 \cdots 0 \\ \hline \bar{A} & \begin{matrix} 1 \\ \vdots \\ 1 \end{matrix} & A \end{array}\right] .$$

Then, by direct computation, we get

$$PP^{*} = \left[\begin{array}{c|c|c} AA^{*} + A^{t}\bar{A} & \sum^{r} A^{t} & AA^{t} + A^{t}A^{*} \\ \hline \sum^{c} (A^{*} + I) & 0 & \sum^{c} A^{t} \\ \hline \bar{A}A^{*} + A\bar{A} + U & \sum^{r} A & \bar{A}A^{t} + AA^{*} \end{array}\right]$$

where $\sum^{r}(\sum^{c})$ means: sum the rows (columns) of, and I is the $m \times m$ identity matrix. It can be shown that $PP^{*} = D$. Note that P is $n \times n$ and hence $L(G) \leqq n$; but D is nonsingular (see Ref. 2, Theorem 4) and so by (9), $L(G) = n$.

(iv) Cycles–n even

In Ref. 2 (Theorem 4), it was shown that for G, a cycle with n even, $N(G) = n/2$, and hence by (8), $L(G) \leqq n$; but diam $G = n/2$ and by (10) $L(G) \geqq n$, so $L(G) = n$. Furthermore, the minimal address matrix H for G can be chosen without d's and hence we can write $D = PP^{*}$ with P $n \times n$.

(v) Cube

Let G be the vertex and edge structure of the m-dimensional cube; then $n = 2^{m}$. It can be shown that $N(G) = m$, and since diam $G = m$,

$L(G) = 2m$. Also, it is apparent that we can write $D = PP^*$ with $P\ n \times 2m$.

(vi) Diameter 2 Graphs

We assume that the graph has at least four vertices, since a graph on three or fewer vertices must be a complete graph or a tree, and we know how to address such graphs. There are now two cases to be considered.
Case 1. Here, we assume that the graph is a star. We know $N(G) = n - 1$, so $L(G) \leqq 2(n - 1)$.
Case 2. Here we assume that the graph is not a star. Then, since the graph is connected, there must be four vertices, say v_1, v_2, v_{n-1}, and v_n, such that v_1 is adjacent to v_2 and v_{n-1} is adjacent to v_n. The distance matrix has the structure

$$D = \left[\begin{array}{cc|c|cc} 0 & 1 & & & \\ 1 & 0 & & & \\ \hline & & & & \\ \hline & & & 0 & 1 \\ & & & 1 & 0 \end{array}\right],$$

where the unspecified elements are 0's, 1's, and 2's. Let D_1 represent the distance matrix of a complete graph on n vertices and let $D = D_1 + D_2$. The remainder matrix D_2 has the form

$$D_2 = \left[\begin{array}{cc|c|cc} 0 & 0 & & & \\ 0 & 0 & & & \\ \hline & & & & \\ \hline & & & 0 & 0 \\ & & & 0 & 0 \end{array}\right],$$

and the unspecified elements are 0's and 1's. The idea now is to take 1's from D_1 and put them into D_2 in such a way as to reduce the rank of D_2. This can be done by taking ones from the first two and last two rows of D_1 and placing them into the first two and last two rows of D_2. The result is $D = D_1' + D_2'$ where

$$D'_2 = \left.\begin{array}{c} \overbrace{\hphantom{1 \cdots 1 \quad 1 \;\; 1 \quad 1 \; 1}}^{n} \\ \left[\begin{array}{cc|ccc|cc} 0 & 0 & 1 & \cdots & 1 & 1 & 1 \\ 0 & 0 & 1 & \cdots & 1 & 1 & 1 \\ \hline \multicolumn{2}{c|}{X} & \multicolumn{3}{c|}{Y} & \multicolumn{2}{c}{Z} \\ \hline 1 & 1 & 1 & \cdots & 1 & 0 & 0 \\ 1 & 1 & 1 & \cdots & 1 & 0 & 0 \end{array}\right] \end{array}\right\} n,$$

and D'_1 remains a 0, 1 valued matrix. Note that rank $(D'_2) \leqq n - 2$. Let I_{n-2} and I_n be the identity matrices of order $n - 2$ and n respectively. Then $D = PQ^t$, where

$$P = \left.\begin{array}{c} \begin{array}{cc} \overbrace{\hphantom{I_n}}^{n} & \overbrace{\hphantom{1 \;\; 0 \cdots 0 \;\; 0}}^{n-2} \end{array} \\ \left[\begin{array}{c|ccc} & 1 & 0 \cdots & 0 \\ I_n & & I_{n-2} & \\ & 0 & \cdots 0 & 1 \end{array}\right] \end{array}\right\} n,$$

and

$$Q^t = \begin{array}{c} \overbrace{\hphantom{0 \;\; 0 \;\; 1 \cdots 1 \;\; 1 \;\; 1}}^{n} \\ \left[\begin{array}{cc|ccc|cc} \multicolumn{7}{c}{D'_1} \\ \hline 0 & 0 & 1 & \cdots & 1 & 1 & 1 \\ \hline \multicolumn{2}{c|}{X} & \multicolumn{3}{c|}{Y} & \multicolumn{2}{c}{Z} \\ \hline 1 & 1 & 1 & \cdots & 1 & 0 & 0 \end{array}\right] \end{array} \begin{array}{l} \} n \\ \left.\begin{array}{c} \\ \\ \\ \end{array}\right\} n - 2. \end{array}$$

Hence, $L(G) \leqq 2(n - 1)$. It is not yet known whether $N(G) \leqq n - 1$ for all diameter 2 graphs. Note that in the above construction $P^* \neq Q^t$ in general.

IV. SOME LINEAR ALGEBRA

In the previous section we realized the representation (4) for several classes of graphs. It is possible to evaluate lower bounds for $N(G)$ directly from these representations, (without having to calculate $n^{\pm}$). This and other results will follow directly from some simple linear algebra. In Lemmas 1 and 2 below, we use the following observation:

if $\mathcal{H}^+$ and $\mathcal{H}^-$ are two subspaces of an n-dimensional real linear space $\mathcal{H}$, such that $\mathcal{H}^+$ and $\mathcal{H}^-$ have no non-zero vectors in common, then $\dim (\mathcal{H}^+) + \dim (\mathcal{H}^-) \leqq n$.

Lemma 1: Let D be any real $n \times n$ matrix (not necessarily symmetric), and let $\mathcal{H}^-$ be any subspace of $\mathcal{H}$ such that for all $x \neq 0$, $x \in \mathcal{H}^-$, we have $\langle x, Dx \rangle < 0$. If W and Z are $n \times n$ matrices such that $\langle x, Wx \rangle \geqq 0$ for all $x \in \mathcal{H}$ and $D = W - Z$, then rank $Z \geqq$ dim $(\mathcal{H}^-)$.

Proof: Let $\mathcal{H}^+$ denote the null space of Z. Clearly $\langle x, Dx \rangle \geqq 0$ for $x \in \mathcal{H}^+$, and $\dim (\mathcal{H}^+) = n$-rank (Z). Also, $\mathcal{H}^+$ and $\mathcal{H}^-$ have no non-zero vectors in common so that $\dim (\mathcal{H}^-) + \dim (\mathcal{H}^+) = \dim (\mathcal{H}^-) +$ n-rank $(Z) \leqq n$ or rank $(Z) \geqq \dim (\mathcal{H}^-)$.

The following result is a dual version of the above:

Lemma 2: Let D be any real $n \times n$ matrix and let $\mathcal{H}^+$ be any subspace of $\mathcal{H}$ such that for all $x \neq 0$, $x \in \mathcal{H}^+$, we have $\langle x, Dx \rangle > 0$. If $D = W - Z$ with $\langle x, Zx \rangle \geqq 0$ for all x, then rank $W \geqq$ dim $(\mathcal{H}^+)$.

The following corollaries are immediate from the two lemmas.

Corollary: Let D be a real $n \times n$ symmetric matrix with n^+ and n^- positive and negative eigenvalues respectively, and let $D = W - Z$. If $\langle x, Zx \rangle \geqq 0$ [$\langle x, Wx \rangle \geqq 0$] for all x, then rank $(W) \geqq n^+$ [rank $(Z) \geqq n^-$].

Corollary: Let D be the distance matrix of a graph G. Then $N(G) \geqq$ max $\{n^+, n^-\}$.

Proof: From condition 3 of Theorem 1, we have $2D = CC^t - EE^t$ where C and E are $n \times N$ matrices. Then application of the previous corollary yields

$$N \geqq \text{rank } C \geqq \text{rank } CC^t \geqq n^+,$$

and

$$N \geqq \text{rank } E \geqq \text{rank } EE^t \geqq n^-.$$

Note that the proof of this result does not depend on the integer nature of the matrices involved.

We mentioned in the previous section that factorizations of the form $D = PP^*$ have special significance. We now demonstrate this.

Theorem 5: Let D be any symmetric non-zero matrix satisfying $D = PP^*$ for some binary-valued $n \times L$ matrix P. Then $n^+ = 1$ and $n^- =$ rank $(D) - 1$.

Proof: Let e_L and e_n be vectors of length L and n respectively, all of whose elements are 1. Then $e_L e_n^t$ is a rank one, $L \times n$ matrix of 1's and $P^* = e_L e_n^t - P^t$. Thus $D = Pe_L e_n^t - PP^t$. But PP^t is non-negative definite and rank $(Pe_L e_n^t) = 1$. Hence, Lemma 2 implies $n^+ \leqq 1$. However, $D_{ij} \geqq 0$ for all i and j and $D \neq 0$ so that $e_n^t D e_n > 0$. Thus $n^+ = 1$, and since $n^+ + n^- = \text{rank } (D)$, we have $n^- = \text{rank } (D) - 1$.

Incidentally, the above shows that every row of P has the same number of 1's.

This Theorem is useful because it enables one to determine n^+ and n^- from rank D. Thus, if a distance matrix of a graph G satisfies $D = PP^*$, then $N(G) \geqq \max\ (1, \text{rank } D - 1) = \text{rank } D - 1$. This result has immediate application to complete graphs, trees, cycles, and cubes considered in the previous section, since in each of those cases, we exhibited a factorization of the form PP^*. We remarked previously that for any graph that admits an address matrix H having no d's as entries, $D = PP^*$ for some P. We now derive an upper bound on the number of rows of H that do not contain d's.

Theorem 6: Let $D = PQ^t$ *be symmetric, and suppose that* r *rows of* P *are the complements of the corresponding* r *rows of* Q. *Then* $r \leqq n - n^+ + 1$.
Proof: $D = PQ^t = P(Q^t + P^t - P^t) = P(Q^t + P^t) - PP^t$
From Lemma 2,

$$\text{rank } (Q^t + P^t) \geqq \text{rank } (P(Q^t + P^t)) \geqq n^+.$$

But $Q^t + P^t$ has n columns, r of which have ones in all positions. Hence rank $(Q^t + P^t) \leqq n - r + 1$, and the desired result follows.

V. AN ALTERNATE ADDRESSING SCHEME

In this section we discuss an addressing scheme that achieves minimum distance routing, and is very simple to construct for all graphs (in fact, D could be any non-negative integer-valued matrix; this could often arise whenever preference for routes is not dictated by just the path length). Let s be the diameter of G, a graph on n vertices. We consider a PQ^t addressing of G with P and Q matrices each of order $n \times ns$. The addressing that follows is a simplified version of the construction used in Theorem 4. The ith row of P, P_i, is zero everywhere except in positions $(s(i - 1) + 1)$ to si where it is one. The ith row of Q, Q_i, is constructed as follows: for every j, there are exactly D_{ji} 1's in positions $(s(j - 1) + 1)$ to sj with the rest of the entries of Q_i equal to zero.

$$P = \begin{bmatrix} 1 \cdots \overset{\overset{s}{\downarrow}}{1} & 0 \cdots \overset{\overset{2s}{\downarrow}}{0} & \overset{3s}{0 \cdots 0} & \cdots & \overset{ns}{0 \cdots 0} \\ 0 \cdots 0 & 1 \cdots 1 & 0 \cdots 0 & & \\ & & & & \\ 0 \cdots 0 & 0 \cdots 0 & 0 \cdots 0 & \cdots & 1 \cdots 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} 0 \cdots & \overset{\overset{s}{\downarrow}}{0} & \overbrace{1 \cdots 1}^{D_{12}} \cdots \overset{\overset{2s}{\downarrow}}{0} & \overbrace{1 \cdots 1}^{D_{13}} \cdots \overset{\overset{3s}{\downarrow}}{0} & \cdots & \overbrace{1 \cdots 1}^{D_{1n}} \cdots \overset{\overset{ns}{\downarrow}}{0} \\ 1 \cdots 1 & 0 & 0 \quad \cdots \quad 0 & 1 \cdots 1 \cdots 0 & & 1 \cdots 1 \cdots 0 \\ & & & & & \\ \underbrace{1 \cdots 1}_{D_{n1}} & 0 & \underbrace{1 \cdots 1}_{D_{n2}} \cdots 0 & \underbrace{1 \cdots 1}_{D_{n3}} \cdots 0 & & 0 \quad \cdots \quad 0 \end{bmatrix}$$

Obviously, the length of addresses in this scheme in ns. The P matrix defined above, rows of which are addresses to be prefixed onto messages, is the same for all graphs with diameter s. The Q matrix, rows of which are stored in the loops in the network, contains the information that identifies a particular graph. But the simplicity of the rows of P can be exploited to minimize the length of the message addresses in the following way: since the integer i ($\leqq n$) uniquely specifies row P_i , the message address need only consist of the binary representation of i, which, of course, requires at most $\log_2 n$ bits. The set of binary representations of the integers $1, \cdots n$ is the set of addresses to be prefixed to messages. In each loop, a device is placed which generates row P_i from the binary representation of i. The distance calculation, say in loop j is accomplished by forming the scalar product of the generated P_i sequence with the stored Q_j sequence. This calculation can be mechanized by an "and gate" followed by a counter (as can be any of the PQ^t decompositions discussed in Section III).

This scheme has the following advantages:

(*i*) The simplicity of constructing addresses by inspection of the D matrix is important in the case of large graphs with no special structure. Even if one were to find good minimum $N(G)$ and $L(G)$, we suspect that the construction of a minimal length addressing will be very complicated.

(*ii*) Since present plans for length of message blocks envisage lengths of perhaps a few thousand bits, the length of the message address is an important parameter in any large loop system. The method of this section guarantees minimum length addresses ($\log_2 n$) provided that the graph is not constrained to have a particular structure.

One of the disadvantages of the present scheme is that it requires a large amount of storage in each loop. However, this can be remedied to a certain extent by coding the number of 1's in each cell of length s of the rows of Q in binary form with $\log_2 s$ bits. Thus the storage requirement can be reduced fron ns bits to $n \log_2 s$ bits. The engineering aspects of this scheme and some of its modifications will be the subject of a forthcoming memorandum.

VI. ANOTHER ALTERNATE ADDRESSING SCHEME

We present here a coding scheme of the type PQ^t, which can address all graphs, using ternary logic symbols $+1$, -1, 0, and requiring addresses of length $2(n - 1)$ for a graph on n vertices. The scheme uses the fact that for any graph there exists a numbering of vertices such that every vertex i is adjacent to some vertex j where $j < i$. This, of course, implies for some pairs i and j that $|D_{im} - D_{jm}| \leqq 1$ for all m. We exhibit the construction inductively. Let $D(r)$ be the $r \times r$ submatrix of D consisting of the first r rows and columns of D.

Suppose $D(r) = P(r)Q^t(r)$ where $P(r)$, $Q(r)$ are $(\pm 1, 0)$-valued matrices of order $r \times 2(r - 1)$. Then it is easily verified that

$$D(r+1) = \begin{bmatrix} P(r) & \underline{0} & A \\ P_s(r) & 1 & 0 \end{bmatrix} \begin{bmatrix} Q^t(r) & Q_s^t(r) \\ A^t & 0 \\ \underline{0}^t & 1 \end{bmatrix}$$

Here $P_s(r)$ is the sth row of $P(r)$ where the $(r + 1)$st vertex is adjacent to the sth vertex, or $|D_{(r+1)m} - D_{sm}| \leqq 1$ for all m. The symbol $\underline{0}$ denotes the $r \times 1$ matrix of zeros, A_j, the jth component of the $r \times 1$ matrix A is $+1$, -1 or 0 according as

$$D_{(r+i)j} - D_{sj} = +1, -1, \text{ or } 0 \text{ respectively.}$$

Calling the obvious matrices $P(r + 1)$ and $Q(r + 1)$, we have $D(r + 1) = P(r + 1)Q(r + 1)$ where $P(r + 1)$, $Q(r + 1)$ are $(r + 1) \times 2r$ matrices. Since $D(2) = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right] \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$, the construction is complete. Because of the structure induced on P and Q by this construction, one can code the rows P and Q making the length of the code smaller, at the expense of increasing the complexity of mechanization. The way this address is constructed simplifies changing the address when a new node is added to an existing coded graph. Notice also that the positivity of the elements of D is unnecessary for this construction.

REFERENCES

1. Pierce, J. R., "Network for Block Switching of Data," B.S.T.J., *51*, No. 6 (July–August 1972), pp. 1133–1145.
2. Graham, R. L., and Pollak, H. O., "On the Addressing Problem for Loop Switching," B.S.T.J., *50*, No. 8 (October 1971), pp. 2405–2519.

Copyright © 1972 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

On the Accuracy of the Depletion Layer Approximation for Charge Coupled Devices

By J. McKENNA and N. L. SCHRYER

(Manuscript received February 28, 1972)

In this paper we examine the accuracy of a variant of the depletion layer approximation as used in the analysis of the electrostatic potential, in the absence of mobile charge, in various charge coupled devices. The approximation is a linearization of the nonlinear potential equations and is important in the numerical solution of two-dimensional problems. A one-dimensional model is solved both exactly and by means of the depletion layer approximation and the two results are compared. We conclude that this variant of the depletion layer approximation is excellent for buried channel devices, and adequate though not as good for surface devices. Some criteria are given which indicate the excellence of the approximation and can be used to estimate errors in two-dimensional calculations.

The purpose of this paper is to discuss the accuracy of a variant of the well known depletion layer approximation[1,2] used in the analysis of the electrostatic potential, in the absence of mobile charge, in various charge coupled devices (CCD's).[3,4] The approximation is a linearization of the nonlinear potential equations and is important in the numerical solution of two-dimensional problems. An application of these results is given in Refs. 5 and 6. We consider a one-dimensional model, which can be solved exactly, and examine in some detail the errors involved in the approximation. This analysis is a generalization of similar results in a previous paper.[1]

Consider a buried channel CCD[7] formed by a first layer, $0 \leqq y \leqq h_1$, of SiO_2; a second layer, $h_1 \leqq y \leqq h_2$, of completely ionized p-type Si; and a third layer, $h_2 \leqq y < \infty$, of uniformly doped n-type Si. (See Fig. 1) Then the dimensionless equations governing the potential $\varphi(x, y)$ are

$$\nabla^2\varphi_1(x, y) = 0, \qquad 0 < y < h_1, \tag{1a}$$

$$\nabla^2\varphi_2(x, y) = \sigma(y), \qquad h_1 < y < h_2, \tag{1b}$$

$$\nabla^2\varphi_3(x, y) = \exp(\varphi_2(x, y)) - 1, \qquad h_2 < y < \infty, \tag{1c}$$

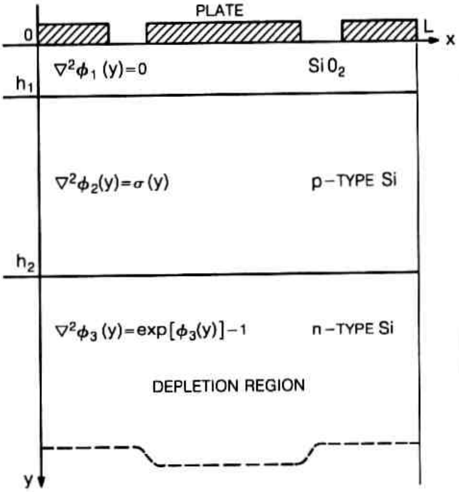


Fig. 1—A schematic diagram of a buried channel CCD.

where $\varphi_i(x, y)$ is the value of $\varphi(x, y)$ in the j-th layer and $\nabla^2 = \partial^2/\partial x^2 + \partial^2/\partial y^2$. In eq. (1), the dimensionless potential $\varphi(x, y)$, and distances, x, y, h_1, h_2, are related to the dimensional quantities $\varphi^*(x^*, y^*)$, x^*, y^*, h_1^*, h_2^*, respectively, (measured in MKS units) by

$$\varphi(x, y) = \frac{e}{kT}\varphi^*(x^*, y^*), \tag{2a}$$

$$x = x^*/\lambda_D , \qquad y = y^*/\lambda_D , \qquad h_1 = h_1^*/\lambda_D , \qquad h_2 = h_2^*/\lambda_D , \tag{2b}$$

where the Debye length λ_D is defined by

$$\lambda_D = (\epsilon_2 kT/e^2 N_{\mathrm{D}}^*)^{\frac{1}{2}}, \tag{2c}$$

and $e > 0$ is the electronic charge, k is Boltzmann's constant, T is the absolute temperature, ϵ_2 the permittivity of the Si, and N_{D}^* is the number density of the donors. Also, the dimensionless density of acceptor ions in the p-layer, $\sigma(y)$, is related to the physical density $N_{\mathrm{A}}^*(y^*)$ by

$$\sigma(y) = N_{\mathrm{A}}^*(y^*)/N_{\mathrm{D}}^* . \tag{2d}$$

We assume that at the boundary $y = 0$, $\varphi_1(x, 0)$ is completely specified.

$$\varphi_1(x, 0) = \mathrm{V}_0(x) \leqq 0, \tag{3a}$$

while at the base $y = \infty$,

$$\varphi_3(x,\,\infty) = 0. \tag{3b}$$

At the internal boundaries, $y = h_1$, $y = h_2$, φ satisfies standard electromagnetic boundary conditions:

$$\varphi_1(x,\,h_1) = \varphi_2(x,\,h_1), \qquad \eta\,\frac{\partial\varphi_1}{\partial y}\,(x,\,h_1) = \frac{\partial\varphi_2}{\partial y}\,(x,\,h_1) + Q_{ss}\,, \tag{4a}$$

$$\varphi_2(x,\,h_2) = \varphi_3(x,\,h_2), \qquad \frac{\partial\varphi_2}{\partial y}\,(x,\,h_2) = \frac{\partial\varphi_2}{\partial y}\,(x,\,h_2), \tag{4b}$$

where $\eta = \epsilon_1/\epsilon_2$ is the ratio of the permittivities of SiO_2 to Si, and Q_{ss} is a possible trapped charge at the oxide-semiconductor interface. The remaining boundary condition is taken care of by assuming periodicity in the x-direction

$$\varphi(0,\,y) = \varphi(L,\,y), \quad 0 \leqq y < \infty \tag{4c}$$

(see Ref. 5 for a fuller discussion of these boundary conditions).

In the buried channel CCD, the parameters are adjusted so that the potential minimum occurs in layer 2, $h_1 \leqq y \leqq h_2$, where the mobile charges are to be trapped. For most purposes, it is only necessary to know the potential accurately in layer 2. The doping in the p-layer is typically introduced by ion implantation followed by drive in diffusion.[8] Then the acceptor density can be given quite accurately by

$$\sigma(y) = c_s \exp\left\{-\left(\frac{x - h_1}{h_2 - h_1}\right)^2 \ln c_s\right\} - 1 \tag{5}$$

where c_s is the surface density of charge due to the original ion implantation. In some applications it is convenient to replace $\sigma(y)$ by its average

$$\bar{\sigma} = \frac{1}{h_2 - h_1}\int_{h_1}^{h_2} \sigma(y)\,dy = \frac{\sqrt{\pi}}{2}\,\frac{c_s}{\sqrt{\ln c_s}}\,\mathrm{erf}\,(\sqrt{\ln c_s}) - 1, \tag{6}$$

where erf is the error function.[9] In a typical buried channel CCD, region 1, the oxide layer, is 0.1μ thick and $\eta = 1/3$. Region 2, the p-layer is 5μ thick with an initially implanted charge density at the surface of $4.6 \times 10^{15}/\mathrm{cm}^3$, which corresponds to an average doping of $\sim 2 \times 10^{15}/\mathrm{cm}^3$. Region 3, the n-layer, has a doping of $10^{14}/\mathrm{cm}^3$. In our dimensionless units, this corresponds to $h_1 = 0.24$, $h_2 = 12.24$, $c_s = 46$ and $\bar{\sigma} = 19.71$ at room temperature. (In all the examples where $\bar{\sigma}$ is used, we take $\bar{\sigma} = 20$ for simplicity). The potential has been scaled so that 1 volt corresponds to 40 dimensionless units. Typical plate voltages are in the range $-400 \leqq V_0 \leqq 0$. Although we shall typically assume $Q_{ss} = 0$ in this paper, we include it because it is necessary in some applications.

If layer 2 is removed, we have a surface CCD.[3] In a surface CCD, the mobile charge is trapped at the oxide-semiconductor interface, and only near this interface does the potential need to be known accurately. The remaining dimensions are typically the same as for a buried channel CCD, and the plate voltage is in the range $-400 \leqq V_0 \leqq -160$.

If $\varphi_3(x, h_2)$ is large and negative for $0 \leqq x \leqq L$, then for $y > h_2$ and near h_2, $\exp(\varphi_3(x, y)) \ll 1$ and the equation for φ_3 is approximately $\nabla^2\varphi_3 = -1$. The region where this holds is called the depletion region, and at any point x, the depth of the boundary of this region below $y = h_2$, $R(x)$, is called the depletion depth (see Fig. 1). For (x, y) such that $y > R(x) + h_2$, $|\varphi_3(x, y)| \ll 1$ and we have approximately $\nabla^2\varphi_2 = \varphi_2$. In most problems $R(x)$ is not a constant, but it is nevertheless often a good approximation to replace $R(x)$ by some suitable constant $\hat{R}$. This suggests replacing the system of eqs. (1) by the system of linear equations

$$\nabla^2\psi_1(x, y) = 0, \qquad 0 < y < h_1, \tag{7a}$$

$$\nabla^2\psi_2(x, y) = \sigma(y), \qquad h_1 < y < h_2, \tag{7b}$$

$$\nabla^2\psi_3(x, y) = -1, \qquad h_2 < y < h_3 = h_2 + \hat{R}, \tag{7c}$$

$$\nabla^2\psi_4(x, y) = \psi_4(x, y), \qquad h_3 = h_2 + \hat{R} < y < \infty, \tag{7d}$$

(See Fig. 2). In addition to ψ_1, ψ_2 and ψ_3 satisfying boundary conditions (3a), (4a), (4b) and (4c), we have the boundary conditions

$$\psi_3(x, h_3) = \psi_4(x, h_3), \quad \frac{\partial\psi_3}{\partial y}(x, h_3) = \frac{\partial\psi_4}{\partial y}(x, h_3), \quad \psi_4(x, \infty) = 0 \tag{8}$$

which must hold for $0 \leqq x \leqq L$.

This paper is devoted to studying how the choice of the pseudo-depletion depth $\hat{R}$ affects the accuracy with which $\psi(x, y)$ approximates $\varphi(x, y)$. We do this by considering the special case $V_0(x) \equiv V_0 < 0$, where V_0 is constant, in boundary condition (3a). For this choice of $V_0(x)$, eqs. (1) and (7) reduce to systems of ordinary differential equations the solutions of which are functions of y only. In all that follows, we drop all reference to x, replace ∇^2 by d^2/dy^2 in eqs. (1) and (7), and consider only the one-dimensional problem. The remainder of the paper is devoted to calculating and comparing the values of $\varphi_2(h_2)$ and $\varphi_2'(h_2)$ with $\psi_2(h_2)$ and $\psi_2'(h_2)$, where $' = d/dy$. It is easy to show that for $h_1 \leqq y \leqq h_2$,

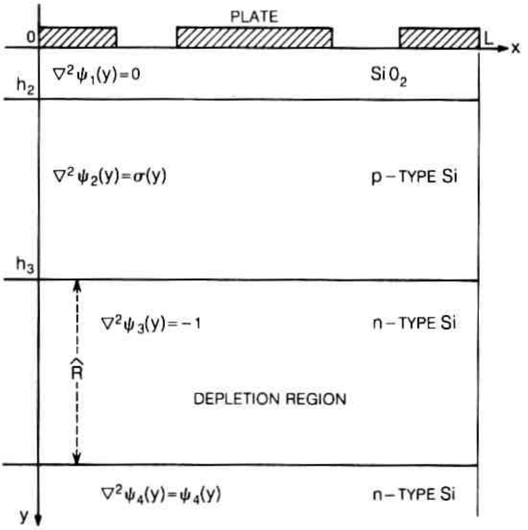


Fig. 2—A schematic diagram of the depletion layer approximation to a one-dimensional, buried channel CCD.

$$| \varphi(y) - \psi(y) | \leqq | \varphi_2(h_2) - \psi_2(h_2) |,$$
$$| \varphi'(y) - \psi'(y) | \leqq | \varphi_2'(h_2) - \psi_2'(h_2) |. \tag{9}$$

We only give the analysis for the buried channel CCD, that is very similar for the surface CCD. We give numerical results for both.

We begin by solving for $\varphi_2(h_2)$ and $\varphi_2'(h_2)$. We can now write

$$\varphi_1(y) = ay + V_0 , \tag{10}$$

$$\varphi_2(y) = \int_y^{h_2} (\xi - y)\sigma(\xi)\, d\xi + b(y - h_2) + c. \tag{11}$$

In addition, a first integral of (1c) subject to (3b) is[1]

$$\tfrac{1}{2}\{\varphi_3'(y)\}^2 = \exp\{\varphi_3(y)\} - \varphi_3(y) - 1. \tag{12}$$

Making use of the boundary conditions (4), it is now easy to show that

$$a = \frac{1}{\eta}\left(b - \int_{h_1}^{h_2} \sigma(\xi)\, d\xi + Q_{ss}\right), \tag{13}$$

$$c = \left(h_2 - h_1 + \frac{h_1}{\eta}\right)b + V_0 + \frac{h_1}{\eta} Q_{ss} - \int_{h_1}^{h_2} \left(\xi - h_1 + \frac{h_1}{\eta}\right)\sigma(\xi)\, d\xi, \tag{14}$$

$$\tfrac{1}{2}b^2 = e^c - c - 1. \tag{15}$$

Equations (14) and (15), for $c = \varphi_2(h_2)$ and $b = \varphi_2'(h_2)$, can be solved numerically. In fact, for most cases of interest $c \ll -1$, so the term e^c in (15) can be neglected, yielding

$$b = \varphi_2'(h_2) = -\left(h_2 - h_1 + \frac{h_1}{\eta}\right)$$

$$+\sqrt{\left(h_2-h_1+\frac{h_1}{\eta}\right)^2-2\left[\mathrm{V}_0+1-\int_{h_1}^{h_2}\left(\xi-h_1+\frac{h_1}{\eta}\right)\sigma(\xi)\,d\xi+\frac{h_1}{\eta}Q_{ss}\right]}, \quad (16)$$

$$c = \varphi_2(h_2) = -\tfrac{1}{2}b^2 - 1. \quad (17)$$

It can be shown that for a surface CCD, $\varphi_2'(h_1)$ and $\varphi_2(h_1)$ can be obtained from (16) and (17) by setting $\sigma(y) \equiv 0$ and $h_1 = h_2$.

To solve eqs. (7), we can write

$$\psi_1(y) = \mathrm{A}y + \mathrm{V}_0, \qquad 0 \leqq y \leqq h_1, \quad (18a)$$

$$\psi_2(y) = \int_y^{h_2} (\xi - y)\sigma(\xi)\,d\xi + \mathrm{B}(y - h_2) + \mathrm{C}, \qquad h_1 \leqq y \leqq h_2, \quad (18b)$$

$$\psi_3(y) = -\tfrac{1}{2}(y - h_2)^2 + \mathrm{D}(y - h_2) + \mathrm{E}, \qquad h_2 \leqq y \leqq h_3, \quad (18c)$$

$$\psi_4(y) = \delta e^{h_2+\hat{\mathrm{R}}-y}, \qquad h_3 = h_2 + \hat{\mathrm{R}} \leqq y < \infty, \quad (18d)$$

where the boundary conditions at $y = 0$ and $y = \infty$ have already been used. Upon employing the remaining boundary conditions at $y = h_1$, h_2 and $h_2 + \hat{\mathrm{R}} = h_3$, we obtain six equations relating the seven constants A, B, C, D, E, δ and $\hat{\mathrm{R}}$.

Our main interest is in cases where $\hat{\mathrm{R}}$ is given, so that A, B, C, D, E and δ can be determined in terms of $\hat{\mathrm{R}}$. However, we first consider the case where δ is specified. This will provide useful information about picking an optimal value of $\hat{\mathrm{R}}$ later on, and will also give a useful criterion for estimating how good the approximate solution is when $\hat{\mathrm{R}}$ is specified.

In Ref. (1) the effects of requiring that $\delta = 0$ (the usual depletion layer approximation) or $\delta = -1$ were studied, and it was shown that the choice $\delta = -1$ yields a better approximate solution. It should be noted that requiring $\delta = -1$ is equivalent to requiring that $\psi_3''(h_3) = \psi_4''(h_3)$. We show here that if the value of δ is specified, the choice $\delta = -1$ does yield the best approximation, but the values of $\psi_2(h_2)$ and $\psi_2'(h_2)$ are relatively insensitive to the choice of δ. It is easy to show that in terms of δ,

$$\hat{\mathrm{R}} = \delta - \left(h_2 - h_1 + \frac{h_1}{\eta}\right) + \left[\left(h_2 - h_1 + \frac{h_1}{\eta}\right)^2 + \delta^2 + 2\delta - 2\mathrm{V}_0 + 2\int_{h_1}^{h_2}\left(\xi - h_1 + \frac{h_1}{\eta}\right)\sigma(\xi)\,d\xi - 2\,\frac{h_1}{\eta}\,Q_{ss}\right]^{\frac{1}{2}}, \tag{19a}$$

$$A = \frac{1}{\eta}\,\hat{\mathrm{R}} + \frac{1}{\eta}\left(Q_{ss} - \int_{h_1}^{h_2}\sigma(\xi)\,d\xi - \delta\right), \tag{19b}$$

$$\psi_2'(h_2) = \mathrm{B} = \mathrm{D} = \hat{\mathrm{R}} - \delta, \tag{19c}$$

$$\psi_2(h_2) = \mathrm{C} = \mathrm{E} = \delta + \hat{\mathrm{R}}\,\delta - \tfrac{1}{2}\hat{\mathrm{R}}^2. \tag{19d}$$

For a surface CCD, the expressions for $\psi_2'(h_1)$ and $\psi_2(h_1)$ are given by (19) when $\sigma(y) \equiv 0$ and $h_1 = h_2$.

As a first example, pick the typical parameters mentioned earlier, $h_1 = 0.24$, $h_2 = 12.24$, $Q_{ss} = 0$, $\eta = 1/3$, $\mathrm{V}_0 = 0$, and let $\sigma(y) \equiv \bar{\sigma} = 20$. In Fig. 3, using eqs. (16), (17) and (19), we show a plot of

$$e(\psi) = |\ \psi_2(h_2) - \varphi_2(h_2)\ |/|\ \varphi_2(h_2)\ | \tag{20a}$$

and

$$e(\psi') = |\ \psi_2'(h_2) - \varphi_2'(h_2)\ |/|\ \varphi_2'(h_2)\ | \tag{20b}$$

as functions of δ for $-10 \leqq \delta \leqq 10$. It can be shown from (17) that these relative errors have a minimum at $\delta = -1$ (and are even functions

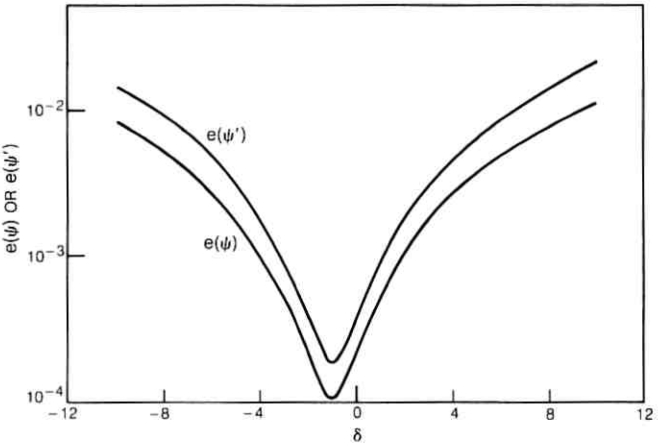


Fig. 3—Plots of $e(\psi)$ and $e(\psi')$ for a buried channel device with $h_1 = 0.24$, $h_2 = 12.24$, $\bar{\sigma} = 20$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$, and $\mathrm{V}_0 = 0$ as functions of δ for $-10 \leqq \delta \leqq 10$.

of $\delta + 1$), but over the whole range of δ shown, the maximum relative error in $\psi_2(h_2)$ is only *1.27* percent and in $\psi_2'(h_2)$ is only *2.28* percent.

As a second example, consider a buried channel CCD with the same parameters as in example one, except that in this case we assume that $\sigma(y)$ is given by eq. (5) with $c_s = 46$. In Fig. 4, we show a plot of $e(\psi)$ for this case as a function of δ for $-10 \leqq \delta \leqq 10$. The results are qualitatively the same as in the first example, except that the relative error is about twice as large. It should be remarked that the potential distributions in the first and second examples are quite different. The potential minimum in example one occurs much nearer the bottom of the p-layer than in example two. In example one, $\varphi(h_2) = -1034.5$ while in example two, $\varphi(h_2) = -530.6$, so we would expect the depletion layer approximation to be better in example one than in example two, as it is.

As a third example, consider a surface CCD with $h_1 = h_2 = 0.24$, $\sigma = 0$, $Q_{ss} = 0$, $\eta = 1/3$, and $V_0 = -160$. In Figs. 5a and 5b, using eqs. (16), (17) and (19), we again show plots of $e(\psi)$ and $e(\psi')$ as functions of δ over $-10 \leqq \delta \leqq 10$. We see a maximum relative error of *1.53* percent in $\psi_2(h_1)$ and *18.3* percent in $\psi_2'(h_1)$.

Since the approximation is based on the assumption that $\varphi(h_2)$ is large and negative, the magnitude of $\varphi(h_2)$ must affect the accuracy of the approximation. For the buried channel CCD, $\varphi(h_2)$ is $\lessapprox -500$ for all $V_0 \leqq 0$, so in this case, the approximation is very accurate for

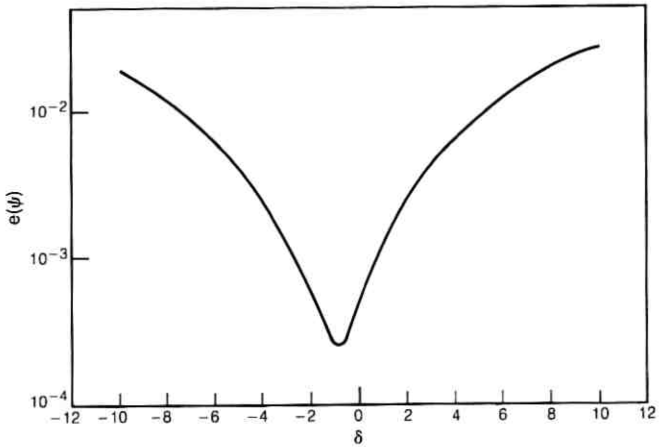


Fig. 4—Plot of $e(\psi)$ for a buried channel device with $h_1 = 0.24$, $h_2 = 12.24$, $c_s = 46$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$, and $V_0 = 0$ as a function of δ for $-10 \leqq \delta \leqq 10$.

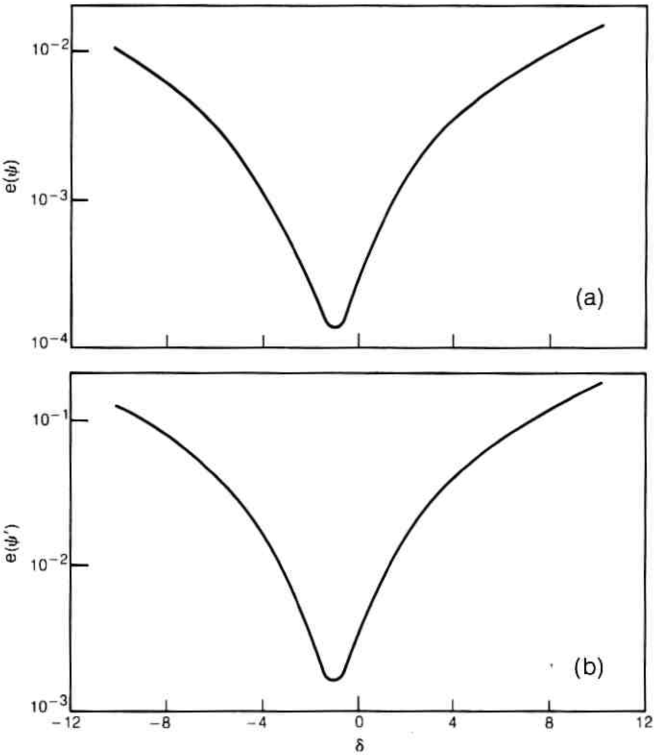


Fig. 5(a) and 5(b)—Plots of $e(\psi)$ and $e(\psi')$, respectively, for a surface device with $h_1 = h_2 = 0.24$, $\sigma = 0$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $V_0 = -160$ as functions of σ for $-10 \leqq \delta \leqq 10$.

any negative V_0. However, in the surface CCD, as $V_0 \to 0$ we have $\varphi(h_1) \to 0$. Thus for "small" V_0, we expect the approximation to be quite bad. Figure 6 gives a plot of the relative error $e(\psi)$ for a surface CCD as a function of V_0 using the same parameters as given for Fig. 5 and $\delta = -10$. To maintain an error of the order of 1 percent for all $\delta \epsilon [-10, 10]$, it is clearly necessary to keep $V_0 \leqq -160$.

We turn now to the main problem where $\hat{R}$ is arbitrarily specified. This occurs typically in the numerical solution of two-dimensional problems where it can be reasonably assumed that the depletion depth $R(x)$ does not vary too much with the various plate potentials. Then an average depletion depth is estimated, say from eq. (19a) by setting $\delta = -1$ and choosing for V_0 some average plate potential.

We now investigate the errors involved in this approximation. If

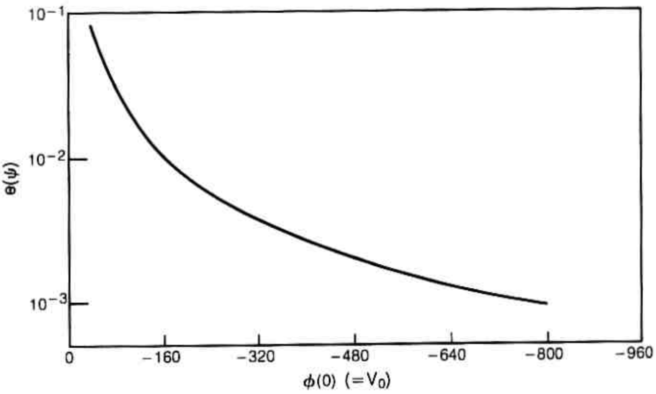


Fig. 6—A plot of $e(\psi)$ for a surface device with $h_1 = h_2 = 0.24$, $\sigma = 0$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\delta = -10$ as a function of V_0 for $-800 \leqq V_0 \leqq -40$.

$\hat{R}$ is given, then it is easy to show that

$$\psi_2'(h_2) = B = D = \left[\int_{h_1}^{h_2}\left(\xi - h_1 + \frac{h_1}{\eta}\right)\sigma(\xi)\,d\xi - \frac{h_1}{\eta}Q_{ss} - V_0 + \tfrac{1}{2}\hat{R}^2 + \hat{R}\right]\Big/\left[\hat{R} + 1 + h_2 - h_1 + \frac{h_1}{\eta}\right], \tag{21a}$$

$$A = \frac{1}{\eta}\left(B - \int_{h_1}^{h_2}\sigma(\xi)\,d\xi + Q_{ss}\right), \tag{21b}$$

$$\psi_2(h_2) = C = E = \tfrac{1}{2}\hat{R}^2 + \hat{R} - (\hat{R} + 1)B, \tag{21c}$$

$$\delta = \hat{R} - B. \tag{21d}$$

As before, the above expressions give $\psi_2(h_1)$ and $\psi_2'(h_1)$ for a surface device when we set $\sigma = 0$ and $h_1 = h_2$.

As a fourth example, we consider a buried channel CCD with $h_1 = 0.24$, $h_2 = 12.24$, $\sigma(y) \equiv \bar{\sigma} = 20$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$, and we choose $\hat{R} = 50.98$. From (19a), this corresponds to the "true" depletion depth R (with $\delta = -1$) for a plate voltage $V_0 = -400$. In Fig. 7, using eqs. (16), (17) and (21), we give plots of $e(\psi)$ and $e(\psi')$ as functions of V_0 for $-800 \leqq V_0 \leqq 0$. In Fig. 8, we give a plot of δ as a function of V_0 from (21d) for the same data, and a plot of the "true" value of R from (19a) for $\delta = -1$ as a function of V_0. Several important conclusions can be drawn from these graphs. First, over $-800 \leqq V_0 \leqq 0$, $\psi(y)$ is an excellent approximation to $\varphi(y)$ for $h_1 \leqq y \leqq h_2$, the maximum relative errors in $\psi_2(h_2)$ and $\psi_2'(h_2)$, which occur at $V_0 = 0$, being

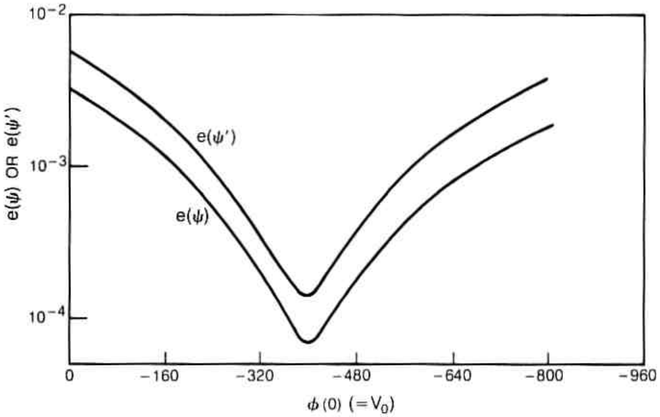


Fig. 7—Plots of $e(\psi)$ and $e(\psi')$ for a buried channel device with $h_1 = 0.24$, $h_2 = 12.24$, $\bar{\sigma} = 20$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{R} = 50.98$ as functions of V_0 for $-800 \leqq V_0 \leqq 0$.

0.41 percent and 0.74 percent respectively. Second, $\psi(y)$ approximates $\varphi(y)$ excellently over $h_1 \leqq y \leqq h_2$, even though a relatively large error has been made in estimating the true value of R. In fact, $\hat{R} = 50.98$ differs from the "true" value by *11.6* percent at $V_0 = -800$ and by

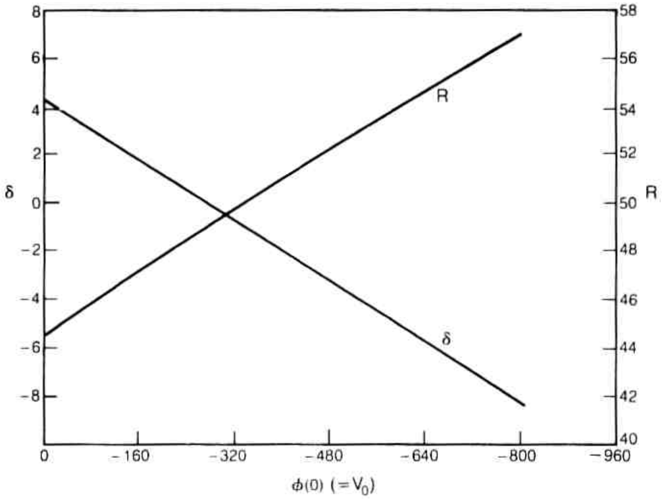


Fig. 8—Plots of δ and "true" R for a buried channel device with $h_1 = 0.24$, $h_2 = 12.24$, $\bar{\sigma} = 20$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{R} = 50.98$ as functions of V_0 for $-800 \leqq V_0 \leqq 0$.

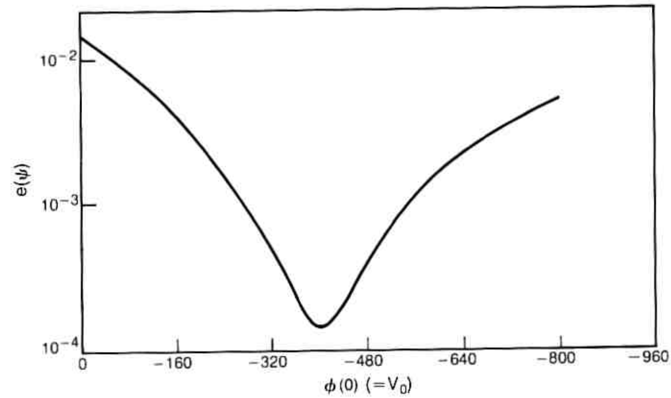


Fig. 9—Plot of $e(\psi)$ for a buried channel device with $h_1 = 0.24$, $h_2 = 12.24$, $c_s = 46$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{R} = 39.66$ as a function of V_0 for $-800 \leqq V_0 \leqq 0$.

12.7 percent at $V_0 = 0$. Finally, an estimate of the error in $\psi(h_2)$ can be obtained from monitoring δ. For $V_0 \leqq 0$, as long as $-7 \leqq \delta \leqq 5$, the error in $\psi_2(h_2)$ and $\psi_2'(h_2)$ is less than 1 percent.

Next, consider a buried channel CCD with $h_1 = 0.24$, $h_2 = 12.24$,

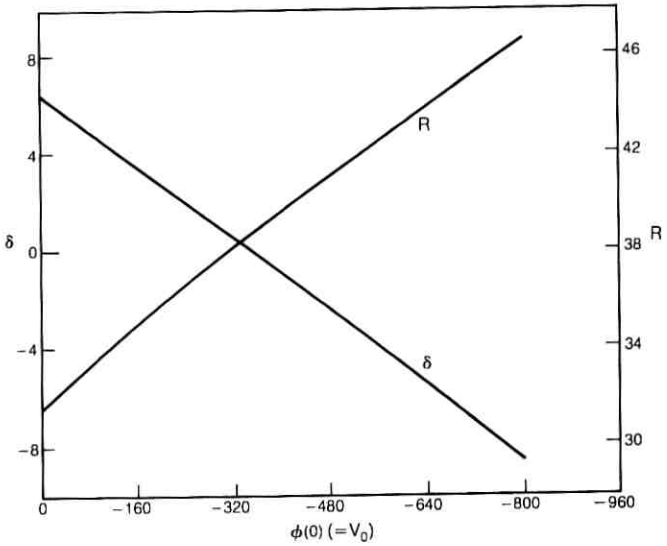


Fig. 10—Plots of δ and "true" R for a buried channel device with $h_1 = 0.24$, $h_2 = 12.24$, $c_s = 46$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{R} = 39.66$ as functions of V_0 for $-800 \leqq V_0 \leqq 0$.

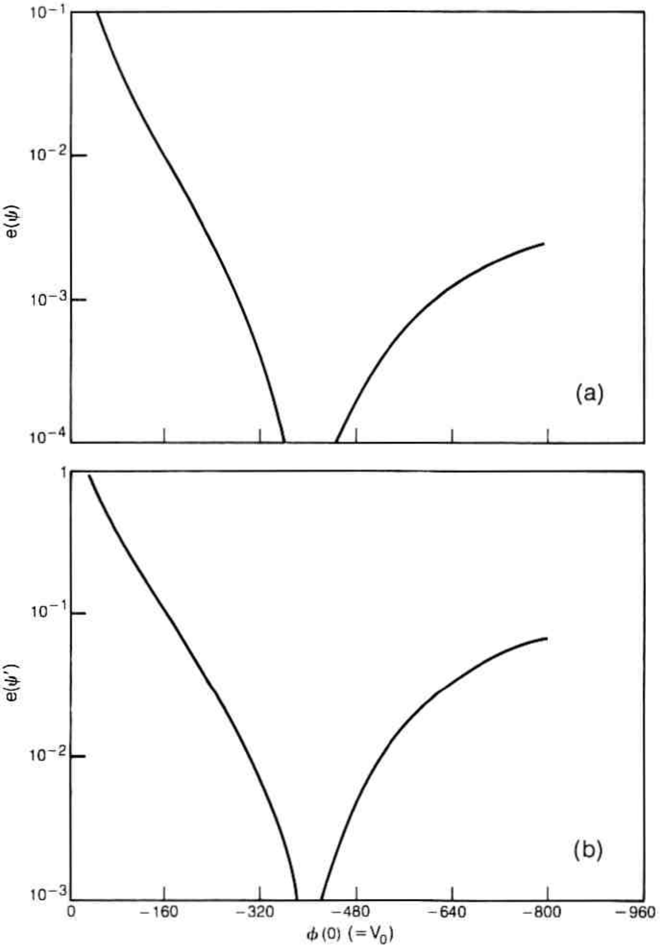


Fig. 11(a) and 11(b)—Plots of $e(\psi)$ and $e(\psi')$, respectively, for a surface device with $h_1 = h_2 = 0.24$, $\sigma = 0$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{\mathrm{R}} = 26.55$ as functions of V_0 for $-800 \leqq \mathrm{V}_0 \leqq -40$.

$Q_{ss} = 0$, $\eta = \frac{1}{3}$, with $\sigma(y)$ given by (5) with $c_s = 46$, and with $\hat{\mathrm{R}} = 39.66$. From (19a), this corresponds to a "true" value of R (with $\delta = -1$) for a plate voltage of $\mathrm{V}_0 = -400$. In Fig. 9, we plot $e(\psi)$ as a function of V_0 for $-800 \leqq \mathrm{V}_0 \leqq 0$. In Fig. 10, we give a plot of δ as a function of V_0 from (21d) for the same data, and a plot of the "true" value of R from (19a) for $\delta = -1$ as a function of V_0. The results are qualitatively the same as for example four, but as we should expect, the errors in using the depletion layer approximation are greater in example five than in example four.

As a final example, we consider a surface CCD with $h_1 = h_2 = 0.24$, $\sigma = 0$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{\text{R}} = 26.55$. From (19a), this corresponds to a "true" value of R (with $\delta = -1$) for a plate voltage of $\text{V}_0 = -400$. In Figs. 11a and 11b, using eqs. (16), (17) and (21), we give plots of $e(\psi)$ and $e(\psi')$ as functions of V_0 for $-800 \leqq \text{V}_0 \leqq -40$. In Fig. 12, we give a plot of δ as a function of V_0 from (21d) for the same data, and a plot of the "true" value of R from (19a) (for $\delta = -1$) as a function of V_0. Hence, we note that over the range $-800 \leqq \text{V}_0 \leqq -160$, the error in $\psi_2(h_1)$ remains remarkably small, less than 1 percent. However, $\psi_2'(h_1)$ does not approximate $\varphi_2'(h_1)$ nearly as well, especially as $\text{V}_0 \rightarrow 0$. Note also that in this case much larger errors have been made in estimating R, the error being 39 percent at $\text{V}_0 = -160$ and 44.2 percent at $\text{V}_0 = -800$. Again, δ provides a check on the accuracy of the approximation, with $-10 \lessapprox \delta \lessapprox 8$ assuring an error of less than 1 percent in approximating $\varphi(y)$ in $h_1 \leqq y \leqq h_2$.

We can conclude that this variant of depletion layer approximation works excellently for the buried channel devices in providing approximations to $\varphi(y)$ and $\varphi'(y)$ in the region $h_1 \leqq y \leqq h_2$. In addition, the parameter δ is a good index of the excellence of the approximation. If in a two-dimensional calculation this variant of the depletion layer

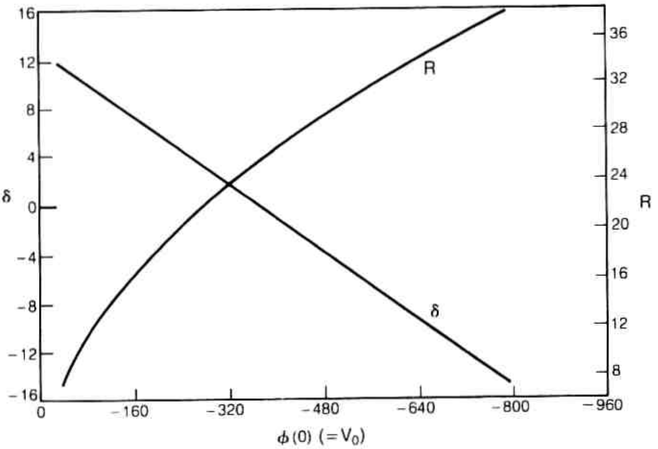


Fig. 12—Plots of δ and "true" R for a surface device with $h_1 = h_2 = 0.24$, $\sigma = 0$, $Q_{ss} = 0$, $\eta = \frac{1}{3}$ and $\hat{\text{R}} = 26.55$ as functions of V_0 for $-800 \leqq \text{V}_0 \leqq -40$.

approximation is used, then a value of the pseudo-depletion depth $\hat{R}$ is estimated, say from using an average plate potential in eq. (19a). Then, so long as the solution of this approximate problem remains in the interval $[-10, 10]$ along the line $y = \hat{R} + h_2$, one can feel reasonably confident that the approximate solution differs from the true solution, in the region of interest, by at most a few percent. It should be noted that the depletion layer approximation is poor near $y = h_2 + \hat{R}$, but in many important cases this is unimportant. Finally, for surface CCDs, the approximation is good to $\varphi(y)$ but not as good to $\varphi'(y)$.

REFERENCES

1. Lewis, J. A., McKenna, J., and Wasserstrom, E., "Field of Negative Point, Line or Plane Charges in an n-Type Semiconductor," J. Appl. Phys., *41*, No. 10 (September 1970), pp. 4182–4189.
2. For a list of many different treatments of the one-dimensional, nonlinear problem treated here, see: Macdonald, J. R., "Distribution of Space Charge in Homogeneous Metal Oxide Films and Semiconductors," J. Chem. Phys., *40*, No. 12 (June 1964), pp. 3735–3737.
3. Boyle, W. S., and Smith, G. E., "Charged Coupled Semi-conductor Devices," B.S.T.J., *49*, No. 4 (April 1970), pp. 587–593.
4. Amelio, G. F., Tompsett, M. R., and Smith, G. E., "Experimental Verification of Charge Coupled Devices," B.S.T.J., *49*, No. 4 (April 1970), pp. 593–600.
5. McKenna, J., and Schryer, N. L., unpublished work.
6. McKenna, J. and Schryer, N. L., unpublished work.
7. Walden, R. H., Krambeck, R. H., Strain, R. J., McKenna, J., Schryer, N. L., and Smith, G. E., "A Buried Channel Charge Coupled Device," B.S.T.J., this issue, pp. 1635–1640.
8. Grove, A. S., *Physics and Technology of Semiconductor Devices*, John Wiley, New York: John Wiley, 1967, p. 49.
9. Abramowitz, M., and Stegun, J. A., *Handbook of Mathematical Functions*, Washington, D. C.: National Bureau of Standards, 1964, p. 297.

Copyright © 1972 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Perceptual Evaluation of the Effects of Dither on Low Bit Rate PCM Systems

By L. R. RABINER and J. A. JOHNSON

(Manuscript received April 10, 1972)

It has previously been shown that by adding a pseudo-random "dither" noise to a signal to be quantized, and by subtracting an identical noise sequence from the quantizer output, it is possible to break up undesirable signal-dependent patterns in the quantization error sequence without increasing the variance of the error. The effect of the dither noise becomes significant when the number of bits per sample is less than about six. An experimental evaluation of the perceptual effects of dither on speech has shown:

(*i*) *strong preferences for dithered speech over straight PCM encoding at identical bit rates,*
(*ii*) *for low bit rates (2–4 bits/sample), a preference for dithered speech over PCM encoded speech even when the PCM speech had one more bit per sample than the dithered speech,*
(*iii*) *an increase in word intelligibility for dithered speech over PCM speech when 4 to 6 bits/sample were used,*
(*iv*) *a decrease in word intelligibility for dithered speech over PCM speech when 2 to 3 bits/sample were used.*

I. INTRODUCTION

When a signal, such as a speech waveform, is quantized, the quantization error waveform is usually correlated with the original signal. This correlation is virtually inperceptible when the quantization is quite fine–i.e., a large number of bits/sample. For crude quantizations, however, the correlation becomes quite large and the quantization error is easily perceived. As a result, it can become quite disturbing to listen to speech quantized to a low number of bits/sample for an extended period of time. In such cases, techniques that decorrelate the quantization error from the signal are attractive, even if they do not increase the signal-to-noise ratio of the system. Dithering is such a technique in which a pseudo-random "dither" noise is added to the speech before quantizing, and then the identical noise is subtracted producing a quantization error which is uncorrelated with the original speech waveform.[1] Figure 1 shows a comparison between straight PCM and a system in which dithering is used. In an earlier work, Jayant and Rabiner[2] discussed several theoretical issues involved with dithering and demonstrated its utility for the quantization of speech signals. In this paper, we present experimental results on the perceptual effects of dither on both the preference and intelligibility of PCM encoded speech.

II. PREFERENCE EVALUATION TEST

The purpose of this experiment was to determine the perceptibility of the decrease in correlation between the quantization error and the original speech, as a function of the number of signal bits.

The stimuli used in the preference test were a set of ten sentences chosen from a list of "everyday speech" sentences[3] compiled at the Central Institute for the Deaf. The sentences used are shown in Table I. These ten sentences were spoken by a General American speaker, digitized at a 10 kHz rate with 16 bits/sample, and stored on the disc of the DDP-516 computer.

In order to limit the number of stimuli to be used in the paired-comparisons preference test, the number of bits/sample was restricted to the range of 2 to 6 bits. Therefore, there were ten distinct stimuli in the test, i.e., (five possible values for the number of bits) × (two types of quantization–dither or straight PCM). For notational convenience, the stimuli were coded using a two-digit code. The first digit refers to the number of bits/sample (i.e., 2–6) and the second digit specifies the type of quantization. A 0 in the second digit means straight PCM encoding, whereas a 1 in the second digit means dithered speech. Thus stimulus 31 has 3 bits/sample and uses the dither noise, whereas stimulus 50 has 5 bits/sample and does not use dither noise.

Since there were ten distinct conditions to be evaluated, a complete

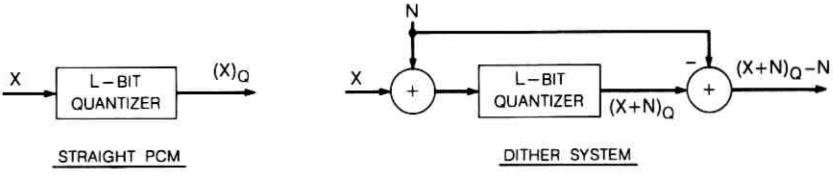


Fig. 1—Block diagrams of a straight PCM system and a dither system.

TABLE I—SENTENCES USED IN PREFERENCE TEST

1. Walking's my favorite exercise.
2. Here's a nice quiet place to rest.
3. Our janitor sweeps the floor every night.
4. It would be much easier if everyone would help.
5. Good morning.
6. Open your windows before you go to bed.
7. Do you think she should stay out so late.
8. How do you feel about changing the time when we begin work.
9. Here we go!
10. Move out of the way.

paired-comparison preference test involved 100 pairs. These 100 pairs were randomly generated by a DDP-516 program which randomly accessed each of the ten stimulus sentences ten times in the course of the experiment. Each of the 100 stimulus pairs was recorded on magnetic tape for offline running of the experiment.

Ten subjects participated in the experiment. Each subject was given the following instructions:

> "In this test you will be listening to pairs of sentences. Each of the two sentences (first is called A, second B) was processed by some type of speech transmission system. After you hear both sentences, there is a five-second interval in which you are to write down the sentence, A or B, you prefer, i.e., the type of transmission system you would prefer listening to for an extended period of time. You *must* choose either A or B–even if you have no preference."

The preference test required two 15-minute listening sessions per subject and was run on two separate days.

III. RESULTS OF PREFERENCE TEST

For each of the ten subjects, a matrix of preferences was determined in which a 1 in a particular cell of the matrix denoted that stimulus B is preferred to stimulus A, and a 0 indicated the reverse condition. Table II shows the matrix obtained by summing the matrices for the ten subjects. Careful inspection of this matrix shows a strong preference for dithered speech over straight PCM encoding at a fixed number of bits/sample, and, in many cases, a preference for dithered speech at L bits/sample (L = 2–4) over straight PCM encoded speech at (L + 1) bits/sample.

To verify these preference results, the data was analyzed using a multidimensional preference program of Carroll.[4] The program indicated

Table II—Matrix of Sum of Preferences for Paired Compression Preference Test

Stimulus B

Stimulus A	20	21	30	31	40	41	50	51	60	61
20	3	8	8	10	10	10	10	10	10	10
21	0	5	5	9	7	10	10	10	10	9
30	0	5	0	9	9	10	10	10	10	10
31	0	1	0	4	2	10	10	10	10	10
40	0	1	0	8	4	9	9	10	10	10
41	0	0	0	0	2	7	4	8	9	10
50	0	0	2	4	1	6	4	10	10	10
51	0	0	0	1	1	4	0	7	8	7
60	0	0	0	0	0	1	0	7	2	3
61	0	0	0	0	1	0	1	3	1	4

Number of preferences of B over A in 10 trials

that the preferences were essentially one-dimensional (over 95 percent of the variance was accounted for by one dimension), and produced a graphical interpretation of the overall preferences which is shown in Fig. 2. Since the preference judgments were one-dimensional, all the conditions lie on a line. The direction of preference goes from left to right in terms of decreasing preference. Figure 2 clearly shows:

(*i*) For a fixed number of bits/sample the dithered speech samples are always preferred to straight PCM encoding,

(*ii*) For 2–4 bits/sample, dithered speech is preferred to straight PCM encodings even with one extra bit/sample, i.e., condition 41 is preferred to condition 50, condition 31 is preferred to condition 40, and condition 21 is preferred to condition 30.

Thus in some perceptual sense, dithered PCM speech has a one-bit advantage over straight PCM encoding under certain conditions. This,

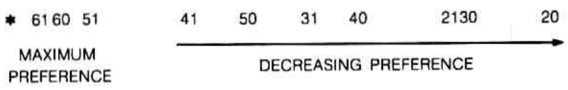


Fig. 2—Ordering of the stimuli in terms of preference.

of course, is not correct in terms of physical measures such as signal-to-noise ratio, or, as we will see, word intelligibility.

A complete analysis of variance was performed on the preference data and the results of this analysis are shown in Table III. The three factors and the number of levels of each are:

(*i*) number of bits/sample	(5)
(*ii*) type of quantization	(2)
(*iii*) subjects	(10)

The analysis reconfirms the conclusions already discussed in that the most significant effects (significance ≫ 0.999 level) were number of bits/sample, and type of quantization. Subjects were significant at the 0.95 level, and the interaction between bits and dither was also significant at this level.

IV. WORD INTELLIGIBILITY TEST

The purpose of the intelligibility test was to determine the effects of dithering on the intelligibility of isolated monosyllables. As discussed earlier, the effect of dither is to make the quantization noise act like an additive wideband uncorrelated noise. Earlier studies[5] have indicated that such a noise tends to mask consonants, thereby lowering intelligibility. The effect of the correlated quantization noise on straight PCM encoding on word intelligibility was also measured.

In this experiment, 200 PB words[6] (Lists 2, 4, 5 and 6 in Ref. 5) were recorded, digitized, and stored on the disc of the DDP-516. The words were accessed at random, in groups of 50 (i.e., an entire list was processed before a new list was used), by one of the ten systems used in the

TABLE III—ANALYSIS OF VARIANCE OF PREFERENCE DATA

Factor	Degrees of Freedom	Mean Square	F-ratio	Significance Level
Subjects (s)	9	1.8	2.3	0.95
Type of quantization (TQ)	1	53.3	67.7	≫0.999
Number bits per sample (NB)	4	142.9	181.5	≫0.999
S × TQ	9	0.8	1.1	N.S.*
NB × S	36	1.0	1.2	N.S.
NB × TQ	4	2.3	2.9	0.95
Residual	36	0.8		

* N.S. ⇒ not significant above 0.90 level.

preference test. The 200 words were divided into two tests of 100 words, each test containing 10 versions of each stimulus condition. The same ten subjects were used in the intelligibility test as in the preference test. The two tests were given on separate days to all ten subjects.

V. RESULTS OF INTELLIGIBILITY TEST

Table IV shows the average error scores as a function of the number of bits/sample, and the type of quantization. (The notation of the previous section is used again here.) These data are averaged over subjects and tests. This table shows that at 2 bits/sample, the PCM system has an error rate of 59.5 percent as opposed to 76 percent for the dither system, i.e., a decrease of 16.5 percent in word intelligibility due to consonant masking. At 3 bits/sample, the PCM system has an error rate of 34.5 percent whereas the dither system has an error rate of 46.5 percent. Thus even at 3 bits/sample, the masking of the dither noise reduces word intelligibility by about 12 percent. At 4-6 bits/sample, the dither system has lower error rates than the PCM system–the differences being 10 percent at 6 bits/sample, 1.5 percent at 5 bits/sample and 0.5 percent at 4 bits/sample. Thus only at 6 bits/sample is the error rate difference significant. The data of Table IV are plotted in Fig. 3 to show how the error rate varies with the number of bits/sample for the two systems.

A complete analysis of variance was performed on the raw data of the intelligibility test. The four factors used in the analysis (and the number of levels of each factor) were

(*i*)	number of bits/sample	(5)
(*ii*)	type of quantization	(2)
(*iii*)	subjects	(10)
(*iv*)	repetitions	(2)

TABLE IV—WORD ERROR SCORES AVERAGED OVER SUBJECTS AND REPETITIONS

Number of Bits per Sample	Error Rate		
	PCM	Dither	Difference
2	59.5%	76%	−16.5%
3	34.5%	46.5%	−12%
4	29.5%	29%	0.5%
5	25%	23.5%	1.5%
6	16.5%	6.5%	10%

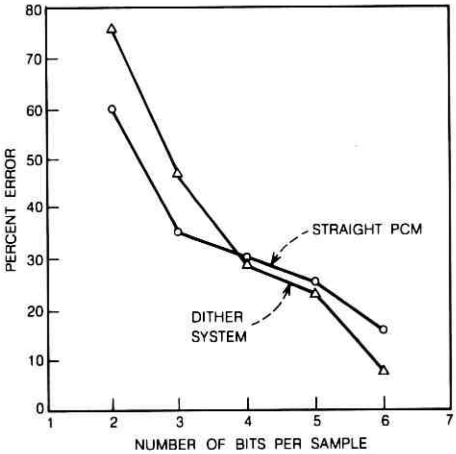


Fig. 3—The percentage error for word intelligibility as a function of the number of bits/sample for straight PCM and dither systems.

The results of the analysis are shown in Table V. The most significant factor was, of course, the number of bits/sample. The next most significant factors were subjects, repetitions, bits/sample × type of quantization, and bits/sample × repetitions. These results indicate a fairly large amount of learning between repetitions 1 and 2, as well as a lack of consistency between the intelligibility scores of the different subjects.

VI. CONCLUSIONS

The results of the preference test were quite encouraging in that subjects uniformly showed strong preferences for dithered speech over straight PCM encoding at all bit rates employed in the experiment. At the lower bit rates, the preference for dithered speech over higher bit rate PCM encoded speech presents strong evidence for the perceptibility and annoyance of highly correlated quantization noise.

The word intelligibility tests showed that the wideband uncorrelated dither noise tended to mask the consonants more than the correlated PCM noise thereby reducing word intelligibility by about 14 percent at low bit rates. At the higher bit rates used in the experiment, there was no decrease in word intelligibility for the dither system, and, in fact, at 6 bits/sample, the dithered words were 10 percent more intelligible than the straight PCM encoded words. Since the average percentage correct for the PCM system was 83.5 percent, an increase of 10 percent is a significant increase in intelligibility.

Overall, these experiments indicate that the use of dither noise in the range of 4–6 bits per sample has many beneficial effects.

Table V—Analysis of Variance of Intelligibility Data

Factor	Degrees of Freedom	Mean Square	F-ratio	Significance Level
Repetitions (R)	1	29.6	30.5	>0.999
Subjects (S)	9	5.7	5.8	>0.999
Type of quantization (TQ)	1	5.4	5.6	0.975
Number bits per sample (NB)	4	180.3	185.6	≫0.999
R × S	9	2.1	2.2	0.95
TQ × R	1	0.05	0.05	N.S.*
TQ × S	9	1.9	1.9	0.90
NB × R	4	7.1	7.3	>0.999
NB × S	36	1.7	1.7	0.90
NB × TQ	4	11.6	11.9	>0.999
TQ × R × S	9	0.5	0.5	N.S.
NB × R × S	36	1.1	1.1	N.S.
NB × TQ × R	4	0.3	0.3	N.S.
NB × TQ × S	36	0.8	0.8	N.S.
Residual	36	1.0		

* N.S. ⇒ not significant above 0.90 level.

VII. ACKNOWLEDGMENTS

The authors would like to thank J. L. Flanagan for suggesting the idea of applying dither noise to PCM encoding, and Professor H. Levitt and S. Pruzansky for assistance in analyzing the data and suggestions on the experimental procedures.

REFERENCES

1. Roberts, L. G., "Picture Coding Using Pseudo Random Noise," IRE Trans. Inform. Theory, *IT-8*, No. 2 (February 1962), pp. 145–154.
2. Jayant, N. S., and Rabiner, L. R., "The Application of Dither to the Quantization of Speech Signals," B.S.T.J., *51*, No. 6 (July–August 1972), pp. 1293–1304.
3. Davis, H., and Silverman, S. R., Appendix in *Hearing and Deafness*, Holt, Rinehart and Winston, 1970.
4. Carroll, J. D., "Non-parametric Multidimensional Analysis of Paired Comparisons Data," unpublished work.
5. Miller, G. A., and Nicely, P. E., "Analysis of Perceptual Confusions Among Some English Consonants," Jour. Acoust. Soc. Amer., *27*, No. 2 (March 1955), pp. 338–352.
6. Egan, J. E., "Articulation Testing Methods," Laryngoscope, *58*, No. 9 (September 1948), pp. 955–991.

Copyright © 1972 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Methods for Designing Differential Quantizers Based on Subjective Evaluations of Edge Busyness

By J. C. CANDY and R. H. BOSWORTH

(Manuscript received February 18, 1972)

This is a study of the visibility of television noise and its dependence on the instantaneous rate of change of the video signal. Noise added to a picture tends to be least noticeable in regions where the brightness changes rapidly, but the relationship between visibility-of-noise and slope-of-the signal is dependent on the scene being displayed. Measurements are presented for four different scenes, and these data are used to design companding laws that minimize the visibility of noise from Differential Quantizers. The designs agree with those that have been satisfactory in practical applications.

I. INTRODUCTION

Experience with Differential Quantizers[1,2,3] teaches us that noise superimposed on brightness boundaries of a picture is less objectionable than similar noise on areas of uniform brightness. This work is an attempt to evaluate the phenomenon and make efficient use of it when designing quantizing scales for Differential Coders. We know that there is advantage in companding the quantization levels so that the largest steps are used for reconstructing rapidly changing signals and the smallest steps for slowly changing signals. In the past, the companding laws for differential quantizers have usually been obtained experimentally by trial and error, but these techniques are inadequate when there are a large number of levels. A more objective method is described here; it provides a basis for theoretical synthesis and evaluation of companding laws independent of the number of levels that are required.

The essence of the method is a subjective evaluation of the dependence of the visibility of noise on the rate of change of the signal. The measurement is used to determine how the visibility of the net quantization noise in a picture depends on the companding law. Laws that minimize this visibility are then obtained and they agree very well with those that have been obtained experimentally. Reference 4 describes a design of differential quantizers that is based on the probability of slopes occuring in pictures and on a weighting function for the visibility of noise.

II. EVALUATING EDGE BUSYNESS

2.1 *Simulating Edge Busyness in Pictures*

Noise added to a video signal at times when it changes rapidly has the appearance of "busyness" near brightness boundaries of the scene.[1,3] In order to measure its visibility, the noise must be introduced into the signal in a controlled fashion. A convenient method compares the rate of change of the signal amplitude with a threshold value, and adds calibrated noise at times when the threshold is exceeded.

Figure 1 illustrates the circuit used for the experiment. The input, a signal that simulates that of a *Picturephone*® system, passes through two parallel paths. In the upper path the signal is first differentiated, and then full-wave rectified to obtain a voltage that is proportional to the magnitude of the rate of change of the input. Whenever this voltage exceeds a threshold value, a trigger circuit fires and gate G conducts. A random signal is then fed through a high-pass filter and added, under control of switch S, to the video signal. In the picture, this noise resembles the edge busyness which is characteristic of differential coding. Its amplitude is controlled by calibrated attenuator A1.

The observer has the choice of two pictures by means of switch S: in position A, edge busyness is added to the scene; in position B, white noise is added to the entire scene. The observer varies the setting of attenuator B1 until, in his judgment, the quality of the picture is the same for the two positions of the switch. The experiment is repeated for various settings of the threshold and of the attenuator A1.

Attenuators A1 and B1 were calibrated to read the ratio of the rms amplitude of the noise to the peak signal. The rms value of the "edge-noise" was measured when the threshold was set at zero, in order that the noise be present at all times. This gives a measure of the noise in areas where it is displayed as opposed to a measure of the noise in the entire picture.

The video signal used in the experiment was derived from a picture scanned with 271 interlaced lines at 30 frames a second. The visible portion of the picture was about 13 cm high and 14 cm wide. The peak

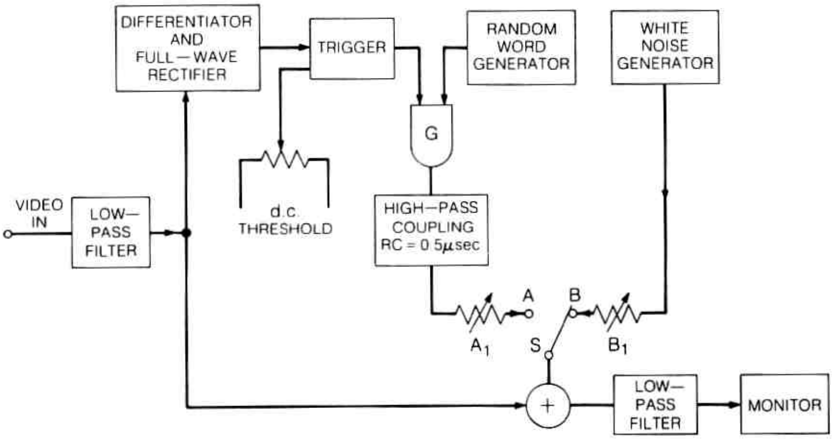


Fig. 1—The circuit used for simulating edge busyness and comparing it with white noise.

luminance was about 70 foot lamberts and the room illuminance about 60 foot candles. The white noise used in the experiments was approximately Gaussian with a flat spectrum from 100 Hz to 1 MHz. The edge noise was obtained from a generator of random binary-words, driven at 2 MHz. The low-pass filters placed at the input and output of the system had frequency characteristics approximating a fourth-order Butterworth filter. At 1 MHz, the gains of the input and output filters had fallen by 6 dB and 15 dB, respectively.

White noise with Gaussian amplitude distribution is used as a standard of comparison because techniques already exist[5] for describing the impairments introduced by uniform quantizing as an equivalent white noise. Moreover, it is a convenient and readily available reference for testing quantizers. A quasi-random binary signal is used as the source of edge noise because it is easily gated and may be synchronized to the scanning waveforms.

2.2 *Comparing Edge Busyness with White Noise*

The success of the experiments depends on the ability of observers to compare different impairments in a picture. These, and previous experiments[5,6] have demonstrated that experienced observers have little difficulty in making the measurements. Inexperienced observers either tire or go through a learning period during which their measurements are not always reproducible. The results reported here were obtained by experienced observers. Their measurements tend to be consistent and reproducible provided the picture quality is reasonably good;[7] that is, having signal-to-noise ratios in excess of 40 dB. Eight observers took part in this study, and the differences between their opinions were small. The difference could usually be expressed as a variation of less than ±1 dB in the sensitivity to noise. Most of the results presented are those obtained by one of the observers. His measurements have been compared with the other observers and are representative of this population. In future studies it may be useful to examine the differences between observers more carefully, but here we shall investigate the much larger differences that occur when scenes, and the type of noise is changed.

Three main experiments are described, their purpose is to find the following:

(*i*) The relationship between the amplitude of the equivalent white noise and the amplitude of the edge busyness for fixed threshold values. (It will be shown that their amplitudes are proportional to one another.)

(*ii*) The relationship between the amplitude of the equivalent white noise and the threshold setting. (It will be demonstrated that the relationship is strongly dependent on the scene.)

(*iii*) An expression for the net equivalent white noise when several unrelated distortions are introduced into the picture. (It will be shown that the power of the white noise that is equivalent to the combined distortions equals the sum of the powers of the white noises that are equivalent to the individual components of the distortions.)

III. EFFECT OF VARYING THE AMPLITUDE OF THE EDGE NOISE

The amplitude of the white noise that was considered to be equivalent to edge busyness was measured for a number of amplitudes of the busyness. Typical results obtained by eight observers are plotted in Fig. 2. The ordinate of this graph is divided into four regions which correspond to subjective classifications of the noise: Objectionable, Annoying, Tolerable, and Undetectable. We are mostly interested in the region where the noise is tolerable; here the results are represented quite well by a straight line having unit slope. This indicates that the amplitude of the equivalent white noise is proportional to the amplitude of the edge noise. The constant of proportionality depends on the observer, but the variation is only ±5 percent.

The data in Fig. 2 were obtained for a threshold value of 0.07; that is,

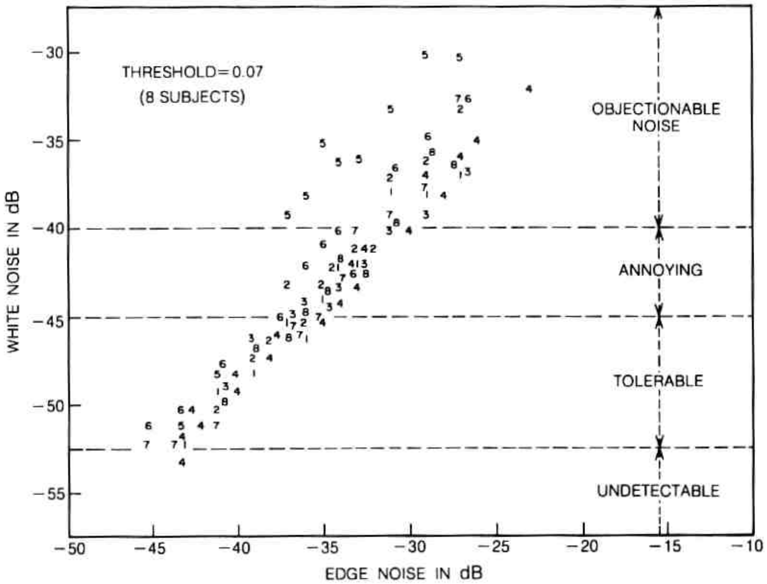


Fig. 2—The white noise that is equivalent to various amplitudes of edge noise for one threshold. There were eight observers.

the noise was added to the signal whenever it changed by more than 7 percent of peak amplitude in a Nyquist Interval (0.5 μs). Figure 3 shows similar graphs for different threshold values. Notice that for zero threshold, the white noise equals the edge noise; this is a consequence of the method used for calibrating the system. As the threshold increases, the amplitude of the equivalent noise decreases but for each threshold, the amplitudes of the two noises, as indicated by the attenuator settings, are proportional to one another.

We now define $F(x)$ as the relative power of the white noise for a threshold x. It is the power of the white noise, $V_w(x)$, that is equivalent to unit noise added to the picture at times when the video signal changes at a rate greater than x, ie.,

$$F(x) = \frac{V_w^2(x)}{V_E^2}. \tag{1}$$

IV. THE RELATIVE WHITE NOISE AS A FUNCTION OF THRESHOLD

The relative white noise is plotted against threshold values in Fig. 4b for the scene shown in Fig. 4a. Also shown in Fig. 4b is the probability

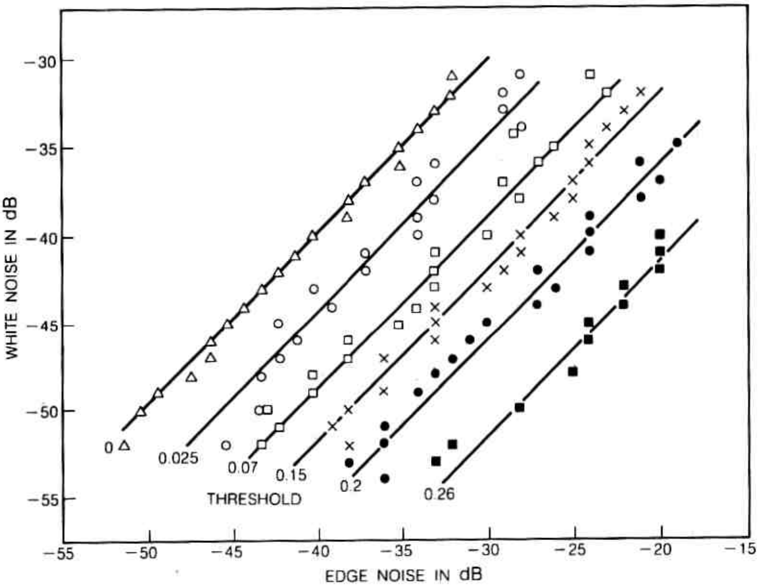


Fig. 3—The white noise that is equivalent to edge noise for six threshold values. There was one observer.

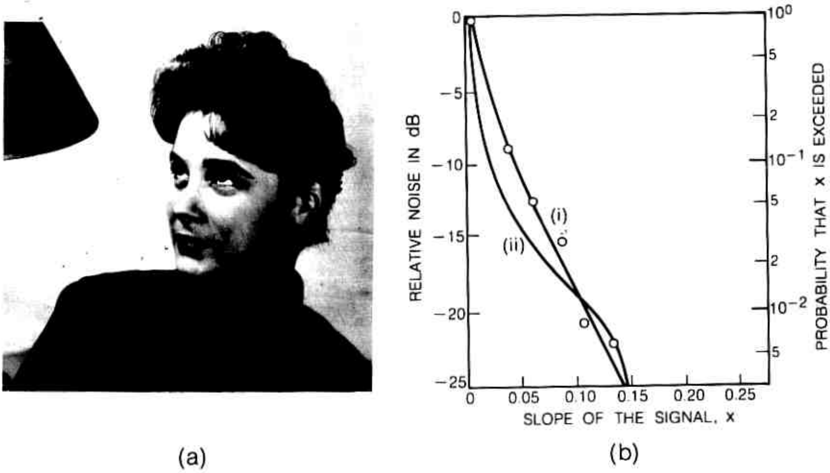


Fig. 4a, b—Picture of a girl with a lamp: (*i*) The relative white noise plotted against threshold. (*ii*) The probability that the threshold is exceeded.

that the slope of the signal exceeds the threshold value. The relative noise for this scene may be represented approximately by the expression

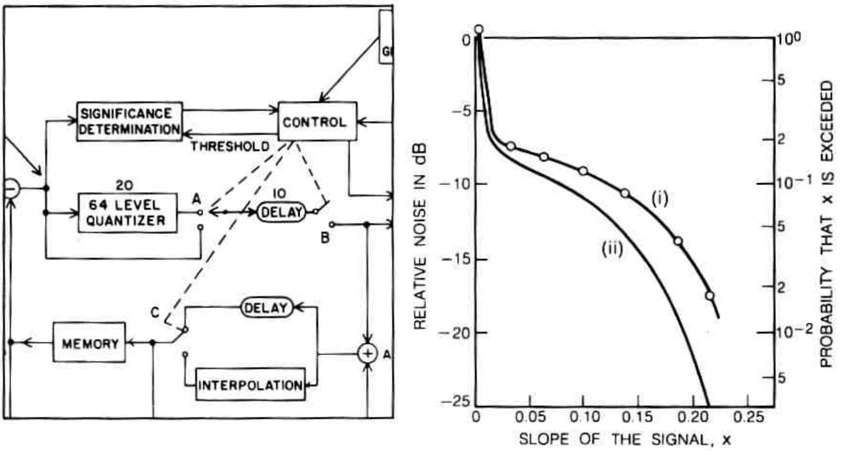
$$F(x) = \exp(-x/\lambda) \tag{2}$$

where λ is a parameter. A similar exponential relationship is obtained for other scenes that contain a variety of detail. The parameter λ depends on the amount of detail in the scene: λ is small for flat scenes and larger for more highly detailed scenes.

The relationship (2) does not apply to scenes made up of abrupt changes separated by regions of uniform brightness such as a page of graphics, as illustrated in Fig. 5. At very small threshold values $F(x)$ falls rapidly with increasing threshold because it is determined largely by noise that is visible on the flat background, but when x exceeds 0.02, the noise is concentrated on the characters. Thereafter, further increases of threshold have less effect on $F(x)$. Other interesting scenes with their graphs are shown in Figs. 6 and 7. Notice that signals with slopes exceeding 0.25 of the maximum possible rarely occur.

V. COMBINING NOISES FROM SEVERAL SOURCES

In order to design and analyze systems, we need an expression for the visibility of noise that is added to the signal at times when its slope has a particular value. Our measurements describe the visibility



(a) (b)

Fig. 5a, b—Picture of graphics: (*i*) The relative white noise plotted against threshold. (*ii*) The probability that the threshold is exceeded.

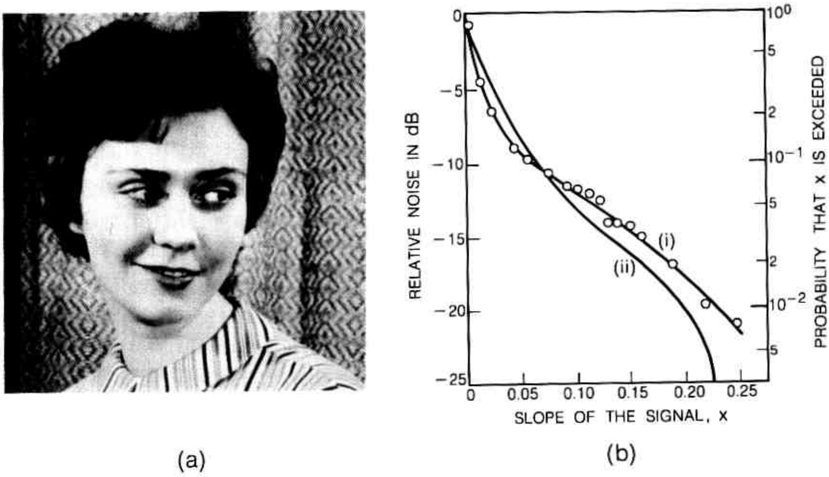


Fig. 6a, b—Picture of Karen: (*i*) The relative white noise plotted against threshold. (*ii*) The probability that the threshold is exceeded.

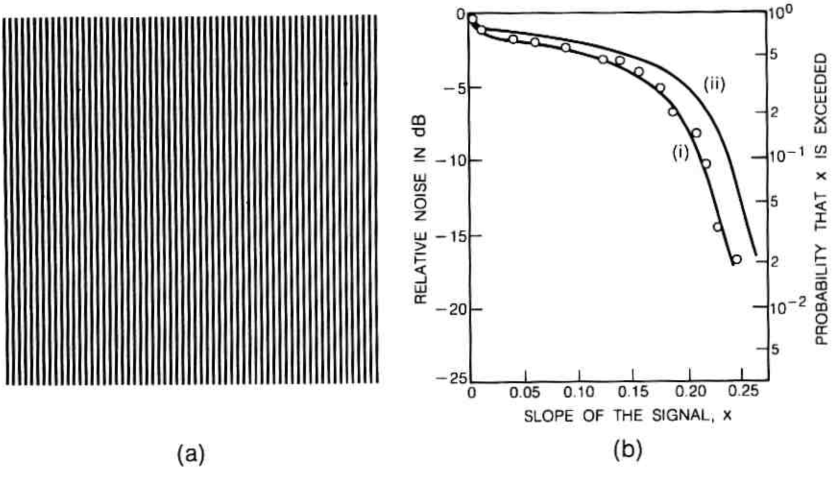


Fig. 7a, b—Picture of stripes: (*i*) The relative white noise plotted against threshold. (*ii*) The probability that the threshold is exceeded.

of noise added to the signal at times when the magnitude of its slope exceeds specified values: it is an accumulation of the required function. In order to obtain the required function we need to know how the visibility of a combination of noises depends on the visibility of its components.

There is strong evidence that, for a combination of distortions, the net equivalent white noise has power equal to the sum of the powers of white noise that are separately equivalent to the component distortions. This hypothesis has been checked by measuring the equivalent white noise for two uncorrelated distortions, separately and in combination. The results are shown in Fig. 8. The abscissa of this graph is the amplitude of the white noise that was equivalent to the least visible component; and the ordinate is the amplitude of the white noise that was equivalent to the combined distortions. Both values are expressed as a ratio of the white noise that was equivalent to the more visible component. The continuous curve is the result expected assuming addition of power. Evidently the hypothesis is a reasonable one, although there is a tendency for the more visible distortion to mask the less visible one to a greater degree than power addition predicts.

The distortions used for these measurements were:

(*i*) White noise added to the entire picture

(*ii*) White noise added to sections of the picture

(*iii*) Noise added when the slope of the signal exceeds a threshold value

(*iv*) Noise added when the slope of the signal lay in a certain interval.

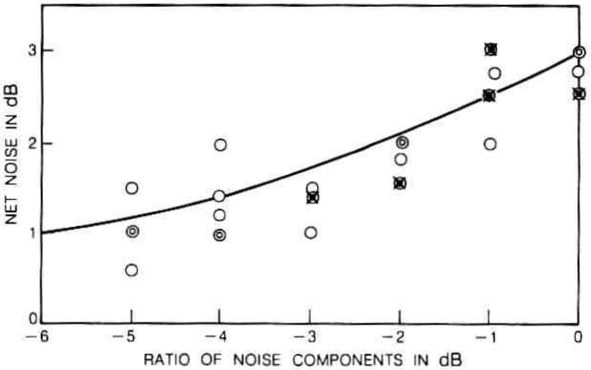


Fig. 8—The white noise that is equivalent to a combination of two distortions plotted against the least visible distortion. Both noise amplitudes are expressed as a ratio of the more visible distortion.

When two distortions of the same kind were introduced into the picture, the result always corresponded with points on the continuous curve in Fig. 8. Such results are not plotted on the graph.

VI. THE VISIBILITY OF NOISE AS A FUNCTION OF SIGNAL SLOPE

The expression $f(x)$ will be used to describe the visibility of unit noise power added to the signal at times when the magnitude of its slope is x. $f(x)$ is defined such that, when rms noise $\alpha(x)$ is added to the signal, the net equivalent white noise has power given by

$$N^2 = \int_0^1 \alpha^2(x) f(x)\, dx. \tag{3}$$

No attempt has been made to separate the visibility into two components dependent on the sign of x. If it is necessary to accommodate one scene, it will surely be necessary to also accommodate its mirror image; it therefore is unwise to make the noise dependent on the sign of x. The expression $f(x)$ can be determined from the subjective measurements of $F(x)$. $F(x)$ is the power of the white noise that is equivalent to unit noise added to the signal at times when the magnitude of the slope exceeds x. Therefore,

$$F(x) = \int_x^1 f(x)\, dx \tag{4}$$

or

$$f(x) = -\frac{dF(x)}{dx} = -F'(x). \tag{5}$$

Values of $f(x)$ are plotted in Fig. 9a for the scenes shown in Figs. 4 through 7. Notice that the visibility of noise tends to decrease as the slope of the signal increases. This decrease is caused by a combination of three effects: It is well known that the ability of the eye to resolve detail decreases near abrupt changes of brightness,[4,8,9] thus we expect the noise to be less visible for large values of x. Moreover, the fraction of the signal that changes rapidly is usually small, as Figs. 4 through 7 demonstrate; thus, for large values of x, noise is added only to a small fraction of the scene. These effects are counteracted to some extent by the fact that the eye tends to concentrate on regions of the scene that carry essential information[10], thus the brightness boundaries tend to be more significant than the overall probability of occurrence indicates. In general, there is not a simple relationship between the function, $f(x)$ and the probability density of slopes, $p(x)$.

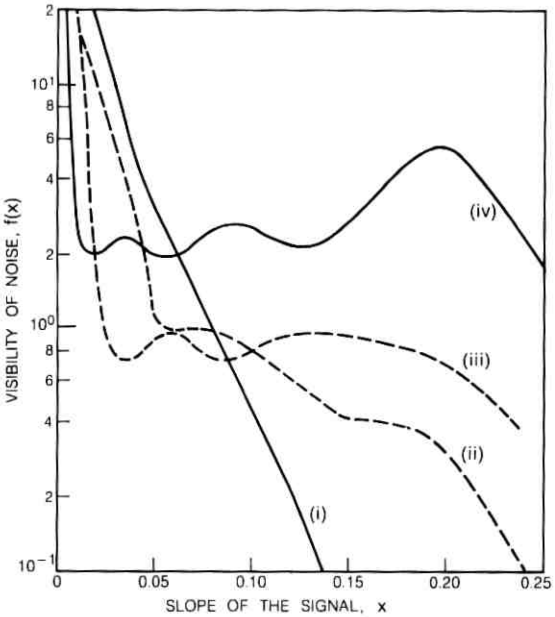


Fig. 9a—The relative visibility of noise $f(x)$ plotted against slope: (i) Girl with a lamp. (ii) Karen. (iii) Graphics. (iv) Stripes.

These functions are compared in Fig. 9b for two scenes; a page full of printed characters and a page containing only one line of characters.

Attempts to measure $f(x)$ directly instead of deriving its value from $F(x)$ have been unsatisfactory for practical reasons. It is much easier to detect the times when the slope of the signal exceeds a threshold value than it is to detect the times when it lies in a narrow range of values. Viewers find that it is easier to measure $F(x)$ than $f(x)$.

VII. OPTIMUM COMPANDING

Knowing the visibility function $f(x)$, it is possible to find distributions of the noise, $\alpha(x)$, that minimize the net equivalent noise given by eq. (3). Before discussing applications of eq. (3), we must emphasize that its derivation assumes that the noise in the picture can be expressed as a function, $\alpha(x)$, of the local slope of the input. This will be true only in special cases, because of storage properties of the integrator that are needed at the receiver. In general, integration will retain the average value of the noise, thus causing streaks to emanate from edges of the scene and persist into flat areas. The above discussion is valid only

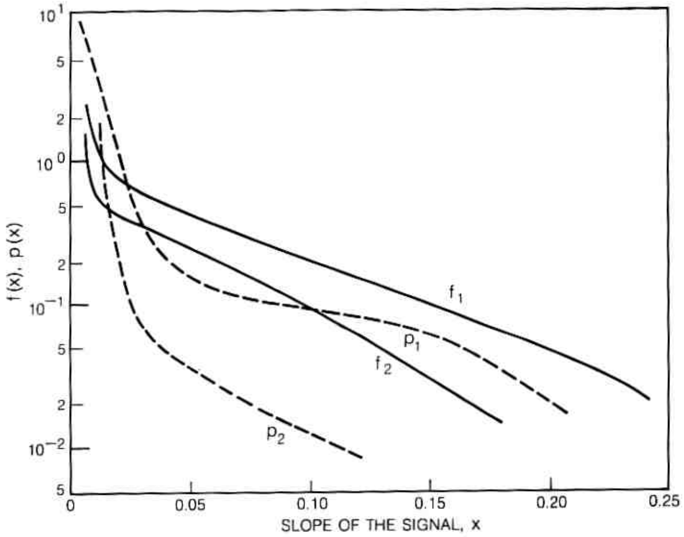


Fig. 9b—A comparison of the function $f(x)$ with the probability density of slopes $p(x)$, for two scenes.

when the average value of the noise added to each edge is small. This is true for Differential Quantization and it can be true for systems of the type illustrated in Fig. 10a, provided the channel noise has only high-frequency components.

The advantages of transmitting video signals as their first derivative are easily realized in digital systems in which the dominant source of noise is the quantization process. The average value of this noise can be kept small by placing a feedback loop around the quantizer, as in a conventional Differential Quantizer.[11,12]

In order to set up equations, let us use the model of a Differential Quantizer that is shown in Fig. 10b. In this circuit, the impairments introduced by quantization are represented as an added noise,[13] $n(t)$. To represent companding of the quantizer levels, the differential signal, x, is compressed[14] according to the function $g(\cdot)$ before noise is added, and expanded afterwards according to the function $h(\cdot)$, which is the inverse of $g(\cdot)$.

The signal emerging from the expandor is given by

$$y = h[g(x) + n], \tag{6}$$

and when the distortion is small, $n \ll g(x)$, it may be written as

$$y \cong h[g(x)] + nh'[g(x)], \tag{7}$$

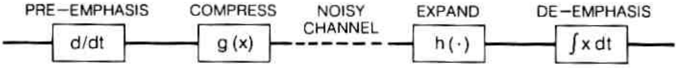


Fig. 10a—Companding the derivative of a signal.

$h'(\cdot)$ and $g'(\cdot)$ are the first derivatives of $h(\cdot)$ and $g(\cdot)$. Because $g(\cdot)$ and $h(\cdot)$ are monotonic and inverses of one another

$$h[g(x)] = x \quad \text{and} \quad h'[g(x)]g'(x) = 1 \tag{8}$$

therefore,

$$y(t) = x(t) + n(t)/g'(x). \tag{9}$$

Now, let $u(t)$ and $v(t)$ be the input and output signals of the system, then

$$v(t + \tau) = y(t) + v(t) \tag{10}$$

and

$$x(t) = u(t) - v(t) \tag{11}$$

therefore

$$v(t + \tau) = u(t) + n(t)/g'(x) \tag{12}$$

and

$$x(t) = u(t) - u(t - \tau) + n(t)/g'(x). \tag{13}$$

Of particular interest are conditions where

$$x(t) \cong u(t) - u(t - \tau), \tag{14}$$

that is, where x approximates the slope of the input signal. Then the noise present in the output signal is a function of the slope of the input and the result in eq. (3) applies. For convenience, normalize the expansions by assuming $n(t)$ has unit rms value, then the output noise is

$$\alpha(x) = 1/g'(x). \tag{15}$$

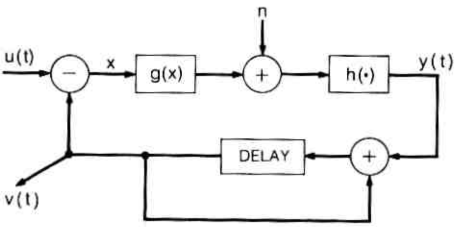


Fig. 10b—Model of a Differential Quantizer.

In practical systems, it is convenient to place constraints on the value of $g(\cdot)$. For example, to have fixed signal levels, let

$$g'(x) > 0, \quad g(0) = 0 \quad \text{and} \quad g(1) = 1. \tag{16}$$

We therefore require that

$$\alpha(x) > 0 \quad \text{and} \quad \int_0^1 \frac{1}{\alpha(x)}\, dx = 1. \tag{17}$$

With these properties, the noise given by eq. (3) can be minimized using variational calculus. (Similar functions are minimized in Refs. 14 and 15.) The visibility of the net noise is a minimum when

$$\alpha_0(x) = f^{-\frac{1}{3}}(x) \int_0^1 f^{\frac{1}{3}}(x)\, dx \tag{18}$$

ie,

$$\alpha_0(x) \propto [f(x)]^{-\frac{1}{3}}.$$

The optimum noise amplification factor $\alpha_0(x)$ is plotted in Fig. 11 for four scenes. Curve (i) shows that, for simple scenes, a compandor can be used to increase the noise continuously as the slope of the signal increases. For very busy scenes, curve (iv) shows that the edge noise can be made about four times larger than the flat-area noise.

VIII. DIFFERENTIAL QUANTIZATION

We have represented the quantization process as additive noise; this is an oversimplification, as is the approximation in eq. (14). However, the conclusions do apply to practical Differential Quantizers. Their action is illustrated in Fig. 12, which shows a typical response to an input rising at a constant rate. The output voltage is seen to step up at each sample time by an amount equal to a quantization level. If the signal change is not matched exactly by an available level, then an appropriate sequence of levels is used to approximate it. In general, the noise introduced by this process is determined by the spacing of the levels that are used; and a well-designed quantizer will use the levels closest to the local slope of the signal. We will now assume that the quantization noise, $\alpha_q(x)$, is proportional to the spacing of the quantization levels that are adjacent to the slope, x, of the signal. This function is plotted in Fig. 13 for an eight-level and a sixteen-level quantizer. These scales were selected by experimental trial and error; the eight-level quantizer[1,2,3] was optimized for pictures of faces and

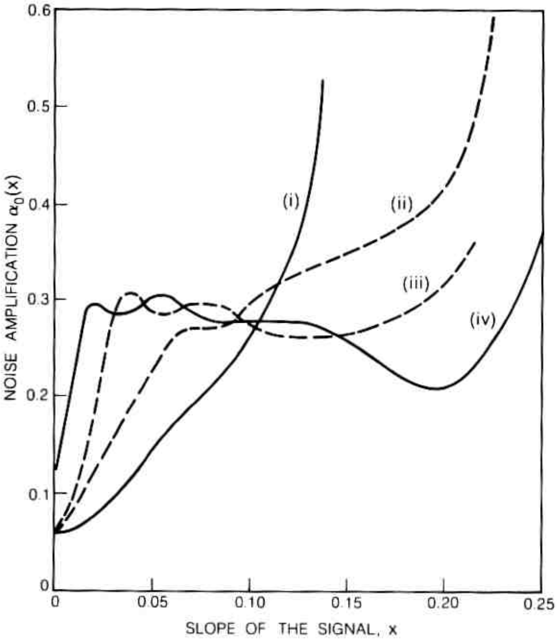


Fig. 11—The optimum noise amplification factor α (x): (i) Girl with a lamp. (ii) Karen. (iii) Graphics. (iv) Stripes.

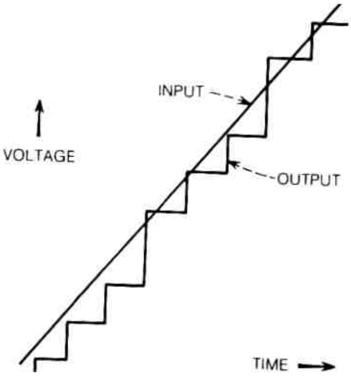


Fig. 12—The action of a Differential Quantizer.

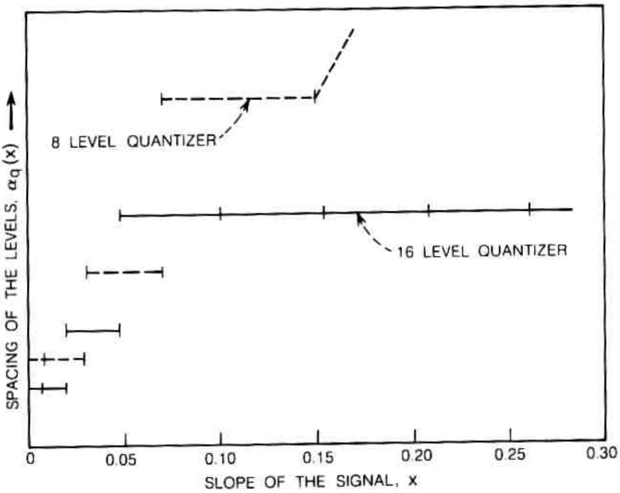


Fig. 13—The quantization intervals for an 8-level and a 16-level quantizer.

is inadequate for graphics. More emphasis was given to the reproduction of high-detail graphics in the design of the sixteen-level scale.[16] The shape of the graphs in Fig. 13 corresponds quite well with the graphs in Fig. 11; the eight-level quantizer approximates curve (i) and the sixteen-level quantizer approximates curve (iv).

The assumption that quantizing noise is proportional to the spacing of the quantization levels is true if $\alpha_q(x)$ is taken to be the average noise for values of x in each quantization interval. In reality, the variation of quantization noise with slope is very complex and has much fine structure. Similarly, any attempt to match the curves in Fig. 11 by line segments, such as in Fig. 13, must necessarily ignore the finer variations of $\alpha(x)$. This is not a disadvantage because the finer variations are very dependent on picture content and gain settings. Indeed, it is advisable to match the quantizer to an average of $\alpha(x)$ in order that a range of scenes can be accommodated. If the intent is to make the spacing of the quantization levels $\alpha_q(x)$ equal to the average value of $\alpha(x)$, there is an easy routine for designing quantizers. It is described in Appendix B.

IX. VISIBILITY OF THE NET QUANTIZATION NOISE

It is useful to have the ability to calculate an expression that describes the net impairment of the received picture; for this purpose we will use the amplitude of an equivalent white noise. The equivalent noise for uniform quantization is easily calculated if overloading is neglected. The error at the decision time ranges with uniform probability between plus and minus half a step interval. Therefore,

$$N_1 = \frac{d}{2\sqrt{3}}, \tag{19}$$

where d is the level spacing. The advantage of optimally companding the level of the Differential Quantizer is shown in Appendix A to be*

$$\left[\int_0^1 f^{\frac{1}{3}}(x)\, dx\right]^3. \tag{20}$$

The predicted advantage over uniform quantization is 17 dB for the picture shown in Fig. 6. It is customary to separate the advantage into two parts. One part is a consequence of the fact that quantization levels need not be provided for slopes greater than certain maximum values. For example, no slopes exceed 0.25 in this scene. The second advantage is a consequence of tapering the levels; this is predicted to be 5 dB for the picture in Fig. 6, about 5 dB improvement has been realized by companding an eight-level quantizer.[17] A similar improvement has also been obtained by weighting the steps of a delta modulator.[6]

A method for calculating the equivalent signal-to-noise ratio for particular quantizers is described in Appendix C. It predicts 50 dB signal-to-noise ratio for the sixteen-level quantizer and 42 dB for the eight-level quantizer when processing the picture in Fig. 6. Both results agree reasonably well with practical measurements.

X. THE VISIBILITY OF NOISE IN DIFFERENT PICTURES

White noise has been used as a standard for measuring impairments of a video signal and has proven to be very useful for evaluating distortions of a given scene. Care is needed, however, in using it to compare the visibility of distortions of different scenes because the visibility of the white noise itself depends on the picture content.[18] Indeed[9], it is very difficult to obtain numerical comparisons of the visibility of a given distortion in different pictures. Table I lists the main properties of the pictures used in this paper, together with an estimate of the visibility of white noise. For example, a given amplitude of white noise was judged to be 2.5 dB less annoying in the picture of Fig. 4 than it was in the graphics of Fig. 5.

* This assumes that the signal-to-noise ratio lies within the range for which our results apply, ie, 40–53 dB.

TABLE I—MAIN PROPERTIES OF PICTURES SHOWN IN FIGS. 4–7.

Scene	Peak/RMS	Visibility of Noise
Girl with lamp (Fig. 4)	11 dB	0
Graphics (Fig. 5)	11 dB	−2.5 dB
Karen (Fig. 6)	13 dB	−1.5 dB
Stripes (Fig. 7)	12 dB	−1.5 dB

The visibility of noise is also dependent on the motion in the scene. We have estimated that white noise is 3 dB less visible on the face of a person talking normally than it is when the face is stationary.

It has also been observed that, for the picture in Fig. 6, the edge busyness is about 2 dB less visible when the noise is synchronized to the scanning waveforms than when it is random. The advantage of synchronizing the noise is fully realized only in stationary scenes.

XI. DISCUSSION AND CONCLUSIONS

We have described a method for measuring and evaluating how the visibility of noise depends on the instantaneous rate of change of the signal. The visibility of quantization noise can be significantly reduced by making its amplitude depend on the slope of the signal. The optimum distribution is very dependent on picture content, therefore measurements have been presented for several different scenes. These scenes were chosen as being typical of important classes and their properties should be useful for characterizing system requirements.

The measurements have been used to design quantization scales for Differential Coders and to predict their performances. The results agree very well with scales that were obtained experimentally and with measured signal-to-noise ratios.

It would be useful to relate the visibility of noise, as measured for this work, to basic properties of scenes and to psychophysical properties of viewers. This has proven to be very difficult. For example, attempts to relate the visibility of noise to the probability density of slopes of the signal have been successful only for scenes of uniform texture, such as Fig. 7a.

The work reported here relates to the rate of change of brightness in the horizontal direction. Comparable results could be expected for the vertical direction.[19] It will be interesting to see similar measurements obtained for other types of pre-emphasis besides simple differentiation. The experiments should also be repeated with other inputs such as color and audio signals, and frame-to-frame changes.

XII. ACKNOWLEDGMENTS

We thank our colleagues at Bell Laboratories for inspiration and assistance, particularly R. C. Brainard, E. F. Brown, D. J. Connor, A. J. Goldstein, B. G. Haskell, J. O. Limb, and F. W. Mounts. We are especially grateful to M. R. Schroeder who suggested the measuring technique, and T. V. Crater for advice on the presentation of our results.

APPENDIX A

Net Noise in an Optimally Companded System

An optimally companded system is understood to be one in which the noise is distributed according to eq. (18), ie,

$$\alpha_0(x) = f^{-\frac{1}{3}}(x) \int_0^1 f^{\frac{1}{3}}(x)\,dx. \tag{21}$$

The net relative-noise power is obtained by substituting in eq. (3) to get

$$N_0^2 = \int_0^1 \alpha_0^2(x) f(x)\,dx = \left[\int_0^1 f^{\frac{1}{3}}(x)\,dx\right]^3. \tag{22}$$

When there is no companding, $\alpha(x) = 1$ the noise is given by

$$N_1^2 = \int_0^1 f(x)\,dx. \tag{23}$$

Therefore, the advantage of companding can be expressed as

$$A = \left[\int_0^1 f^{\frac{1}{3}}(x)\,dx\right]^{-3} \tag{24}$$

because the measurements have been calibrated so that

$$\int_0^1 f(x)\,dx = F(1) = 1. \tag{25}$$

Most viewers, when judging pictures, are conscious of two components in the noise: flat area noise which corresponds approximately to areas of the picture where $x < 0.02$, and "edge-noise" which corresponds to $x > 0.02$. With optimum companding, the "edge-noise" usually predominates because

$$\int_0^{0.02} f^{\frac{1}{3}}(x)\,dx < \int_{0.02}^1 f^{\frac{1}{3}}(x)\,dx. \tag{26}$$

APPENDIX B

Determining Quantization Levels

The objective is to design a quantizer for which the relative spacing of the levels $\alpha_q(x)$ represents a given value of the function $\alpha_0(x)$ in eq. (10). An example is shown in Fig. 14. The function $\alpha_0(x)$ is plotted as a continuous curve and is extended to negative values of x by assuming it is an even function. The rectangles superimposed on this curve have the following properties: First, their height is proportional to the average value of $\alpha_0(x)$ during the interval defined by the width of the rectangle. Second, the inclination of the diagonal is the same for all the rectangles; this makes their width proportional to their height.

The quantization levels are set equal to the value of x that corresponds with the sides of the rectangles. Then the spacing of the levels is proportional, in turn, to the width of the rectangle, the height of the rectangle, and to the average value of $\alpha_0(x)$; thus satisfying our requirements. The number of levels in the quantizer is one more than the number of rectangles and this is determined by the inclination selected for the diagonals. Quantizers with given numbers of levels can be obtained by an iteration of the design method. The decision levels would be placed midway between the quantization levels.[20]

A "worst case" design is required for systems that must accommodate a variety of scenes. For example, for small values of x, the levels would be chosen to accommodate scenes with properties given by curve (i) of Fig. 11, and for large values of x, the levels would be chosen to accommodate scenes with properties (iv).

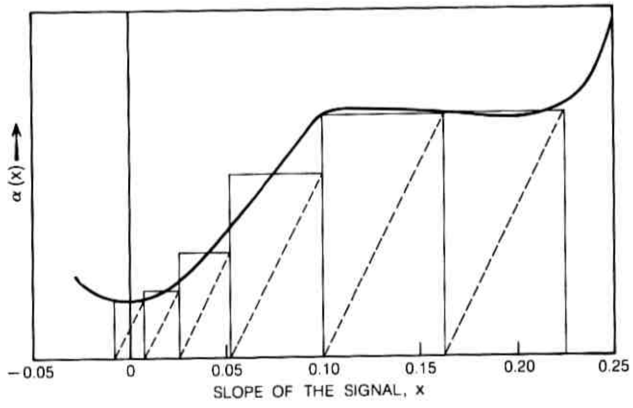


Fig. 14—A method for selecting quantization levels.

APPENDIX C

Determining the Visibility of Quantization Noise

Having determined the values of the quantization levels it is useful to be able to calculate the visibility of net quantization noise. The net quantization noise power will be given by

$$N_q^2 = \int_0^1 \alpha_q^2(x) f(x)\, dx \tag{27}$$

where $\alpha_q(x)$ is the rms quantization noise which is assumed to be proportional to the spacing of the levels. The constant of proportionality is given by eq. (11). If the nth quantization level is ln then

$$\alpha_q(x) = \frac{(l_n - l_{(n-1)})}{2\sqrt{3}}, \qquad l_{n-1} < x \leqq l_n . \tag{28}$$

The constraints (16) can be interpreted as requiring a fixed number of quantization levels, which are numbered in ascending order according to their magnitude. Therefore

$$N_q^2 = \frac{1}{12} \sum_n \left[(l_n - l_{n-1})^2 \int_{l_{n-1}}^{l_n} f(x)\, dx \right]. \tag{29}$$

Assuming that the companding is symmetrical, $l_n = -l_n$, then using eq. (4) and neglecting overloading we obtain

$$N_q^2 = \frac{1}{12} \left[4l_1^2(1 - F_{l_1}) + \sum_{n=2}^{N} (l_{n-1} - l_n)^2 [F(l_{n-1}) - F(l_n)] \right] \tag{30}$$

for an even number of levels $2N$, and

$$N_q^2 = \frac{1}{12} \left[\sum_{n=1}^{N} (l_{n-1} - l_n)^2 [F(l_{n-1}) - F(l_n)] \right] \tag{31}$$

for an odd number of levels $(2N + 1)$.

These expressions are easily evaluated when l_n and F are known. Values of the quantization levels l_n are found by using the approximate method described in Appendix B because we do not have an easy method for selecting values of l that minimize N_q^2. It has been found that levels, determined by the method described, give signal-to-noise ratios that are only 2 dB less than that predicted by eq. 14, that is, for optimum shaping of the noise using the continuous function $\alpha_0(x)$ instead of the discontinuous $\alpha_q(x)$.

Important cases are ones for which the quantization levels are sufficiently close that $f(x)$ is effectively constant during each interval. Then the method shown in Fig. 14 for obtaining levels gives optimum companding $\alpha_q = \alpha_0$. Equation (30) shows that every quantization interval then contributes the same amount to the visible noise.

REFERENCES

1. Millard, J. B., and Maunsell, H. I., "Digital Encoding of the Video Signal," B.S.T.J., *50*, No. 2 (February 1971), pp. 459–480.
2. Limb, J. O., and Mounts, F. W., "Digital Differential Quantizer for Television," B.S.T.J., *48*, No. 7 (September 1969), pp. 2583–2599.
3. Brown, Earl F., "A Sliding-Scale Direct—Feedback PCM Coder for Television," B.S.T.J., *48*, No. 5 (May–June 1969), pp. 1537–1553.
4. Limb, J. O., "Source-Receiver Encoding of Television Signals," Proc. IEEE, *55* (April 1967), pp. 364–379.
5. Brainard, R. C., "Subjective Evaluation of PCM Noise-Feedback Coder for Television," Proc. IEEE, No. 3 (March 1967), pp. 346–353.
6. Bosworth, R. H., and Candy, J. C., "A Companded One-Bit Coder for Television Transmission," B.S.T.J., *48*, No. 5 (May–June 1969), pp. 1459–1479.
7. Crater, T. V., "The PICTUREPHONE® System: Service Standards," B.S.T.J., *50*, No. 2 (February 1971), pp. 235–269.
8. Brown, E. F., and Kaminski, W., "Television: Visual Threshold to Errors Near Brightness Boundaries," Private Communication, July 1970.
9. Novak, S., and Sperling, G., "Visual Thresholds Near a Continuously Visible or Briefly Presented Light-Dark Boundary," Optica Acta, *10* (April 1963), pp. 87–91.
10. Yarbus, Alfred L., *Eye Movements and Vision*, New York: Plenum Press, 1967, Chapter VII, pp. 171–211.
11. Cutler, C. C., "Differential Quantization of Communication Signals," Patent No. 2, 605, 361 (June 1950).
12. Graham, R. E., "Predictive Quantizing of Television Signals," IRE Wescon Conv. Rec. 2, Part 4, 1958, pp. 147–157.
13. Brainard, R. C., and Candy, J. C., "Direct-Feedback Coders: Design and Performance with Television Signals," Proc. IEEE, *57*, No. 5 (May 1969), pp. 776–786.
14. Smith, Bernard, "Instantaneous Companding of Quantized Signals," B.S.T.J., *36*, No. 3 (May 1957), pp. 653–709.
15. Panter, P. F., and Dite, W., "Quantization Distortion in Pulse-Count Modulation with Nonuniform Spacing of Levels," Proc. IRE. *39* (January 1951), pp. 44–48.
16. Abbot, R. P., "DPCM Codec for Video Telephony Using 4-Bits Per Sample," IEEE, Trans. Commun. Tech. (December 1971).
17. Kaminski, W., Private Communication.
18. Hacking, K., "The Relative Visibility of Random Noise Over the Grey-Scale," Journal Brit., I.R.E. (April 1962).
19. Connor, D. J., Pease, R. F. W., and Scholes, W. G., "Television Coding Using Two-Dimensional Spacial Prediction," B.S.T.J., *50*, No. 3 (March 1971), pp. 1049–1061.
20. Max, Joel, "Quantizing for Minimum Distortion," IRE Trans., *IT-6* (March 1960), pp. 7–12.

Copyright © 1972 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Modeling the Growth of Jumpers on the Main Distributing Frame

By S. HALFIN, C. J. McCALLUM, JR., and M. SEGAL

(Manuscript received April 18, 1972)

The buildup of dead jumpers in the Main Distributing Frame (MDF) plays a central role in MDF problems; for example, a recent survey ranked dead jumpers as the number two problem on a list of the most frequently reported MDF problems. In this paper, two models are proposed to quantify the buildup of both live and dead jumpers and to investigate the factors influencing the buildup. These models provide tools for the analysis and comparison of possible solutions to the buildup problem.

I. INTRODUCTION

The Main Distributing Frame (MDF) in a central office building serves as the connecting point between the cable outside and the equipment inside. Conventional MDF's are iron or wooden structures with terminal strips mounted on each side. The two sides are termed *vertical* and *horizontal* due to the manner in which the terminal strips are mounted. Cable pairs from subscribers' stations are terminated on the vertical side of the MDF, while line and trunk equipments are wired to terminals on the horizontal side of the frame. In order to provide service to a subscriber, it is necessary to connect his cable pair to the proper line equipment. A frameman makes this cross-connection by manually stringing a wire, called a *jumper*, between the corresponding vertical and horizontal terminals. These jumpers, which can easily be 100 feet long, are laid along horizontal shelves in the frame. As more and more jumpers are added to the frame, these horizontal shelves tend to become crowded.

When the service to a particular station is discontinued, it is necessary to manually disconnect and remove the corresponding jumper. Unfortunately, disconnected jumpers are not always removed, which gives rise to *dead* jumpers in addition to the *live* jumpers.

In the last few years, main frames which were originally conceived over seventy years ago are becoming highly congested with live, and dead, jumpers.[1] The problem of the buildup of jumpers on the shelves of the MDF has been discussed extensively and many solutions have been suggested and examined.[2,3] These proposals include preferential assignment of line equipments to cable pairs, multipling line equipments to several appearances on the horizontal side of the frame, spreading line equipments along the horizontal side, using several separated frames connected by tie cables, mechanization of the assignment records, and others. A natural way to evaluate such proposals is by means of a growth model for the buildup of jumpers in the MDF. Desirable features of such models are:

(*i*) The models should enable the comparison of the amount of live and dead jumpers during a given period of time, for various conditions and methods of administration of the MDF.

(*ii*) The models should use as inputs either data which are easily available or which can be theoretically deduced from available data. Such data are: the physical description of the frame, the rate of connect and disconnect orders, distribution of lengths of the jumpers, which can be deduced from administrative practices (for instance, the method of assignment), etc.

One would expect such models to be able to clarify why some MDF's suffer from dead jumper accumulation, while others, though apparently similar, do not. Also, the models should facilitate evaluation of the proposed solutions.

In this paper, we investigate the jumper buildup in a frame over time, with emphasis on the dead jumper buildup. Two models are proposed to quantify this buildup.

The first model is based on the assumption that the fraction of dead jumpers not removed from the frame has a certain functional form which depends on two presumably measurable parameters which are described later. Given values of these two parameters and the values of two other frame parameters which are easily determined, the model can be used to iteratively calculate the jumper buildup over time starting from some initial state.

In the second model, it is assumed that a dead jumper remains on the shelf either because its assignment record is erroneous, or because the force needed to remove it is too large. Based on these assumptions, recursive equations are deduced for the quantity of jumpers on a shelf at any given time.

In Section II, the first model is developed and some numerical results are given. Section III is concerned with the second model. Some concluding comments are given in Section IV.

II. DEVELOPMENT OF MODEL 1

2.1 *The Model*

Let $t = 0, 1, \cdots$ denote discrete points in time. Time period t will refer to the time period between time t and time $t + 1$. Let $J(t)$ be the total number of jumpers (both live and dead) in the frame at time t; equivalently, $J(t)$ is the number of jumpers in the frame at the beginning of time period t before any service orders for that time period are processed. Let $c(t)$ denote the number of connects (i.e., service orders which require the installation of a jumper) that occur during time period t. Similarly, let $d(t)$ denote the number of disconnects (i.e., service orders which require the disconnection and removal of a jumper) that occur during time period t. For ease in exposition, we assume that each service order calls for the connection or the disconnection of only one jumper.

Using the terms introduced above, we can write $J(t + 1)$ as

$$J(t+1) = J(t) + c(t) - \beta(t)\, d(t), \qquad t = 0, 1, \cdots \tag{1}$$

where $\beta(t)$, defined as the fraction of the disconnects $d(t)$ that are removed, accounts for the fact that disconnected jumpers are sometimes left in the frame. Clearly, $0 \leqq \beta(t) \leqq 1$.

It is convenient to introduce the substitution $\beta(t) = 1 - \alpha(t)$ into eq. (1) where $\alpha(t)$ represents the fraction of disconnects that are *not* removed. Note that $0 \leqq \alpha(t) \leqq 1$. This gives

$$J(t+1) = J(t) + c(t) - d(t) + \alpha(t)\, d(t) \tag{2}$$

where $c(t) - d(t)$ represents the net gain in live jumpers during time period t. Similarly, the expression $\alpha(t)\, d(t)$ represents the increase in dead jumpers during time period t. It is clear that we favor the situation in which $\alpha(t)$ is near 0 (equivalently, $\beta(t)$ is near 1). In such a case, the jumper buildup in time will correspond only to live jumpers added to the frame and the additional frame congestion brought on by the presence of dead jumpers will not be felt.

We now simplify the model and remove the dependence of $c(t)$ and $d(t)$ on time. Let S be the number of service orders per time period and let G be the fraction of the service orders that result in net gain, i.e.,

$$S = c(t) + d(t) \tag{3}$$

$$G = \frac{c(t) - d(t)}{c(t) + d(t)} \tag{4}$$

for all $t = 0, 1, \cdots$. We assume $S > 0$. Note that in the usual case in which $c(t) \geqq d(t)$ we have $0 \leqq G \leqq 1$. Using eq. (3) and eq. (4) to eliminate $c(t)$ and $d(t)$ from eq. (2), we have

$$J(t+1) = J(t) + GS + \tfrac{1}{2}(1 - G)S\alpha(t). \tag{5}$$

2.2 *The Parameter $\alpha(t)$*

We now investigate the parameter $\alpha(t)$, the fraction disconnects not removed. Though S and G were taken to be time independent, this seems to be an unreasonable assumption to make for $\alpha(t)$. One might expect that $\alpha(t)$ will increase with time as the number of jumpers in the frame increases. That is, when there are few jumpers in the frame (i.e., $J(t)$ is small) we expect that most disconnects will actually be removed (i.e., $\alpha(t)$ is near 0) since the frame is certainly not congested and only the disinclination of a frameman to remove an occasional disconnect or an error in the records will lead to its being left in the frame. However, as $J(t)$ increases, the frame becomes congested due to jumper buildup, and we expect that, due to physical limitations, fewer and fewer of the disconnects will actually be removed (i.e., $\alpha(t)$ is near 1). The removal of a disconnect will become physically difficult if not impossible and could jeopardize the continued operation of nearby live jumpers. Thus, we define $\alpha(t)$ to be an increasing function of $J(t)$ with a range of 0 to 1. Furthermore, we require that $\alpha(t)$ actually reaches the value 1, i.e., at some time t^* when $J(t^*) = K$, we have $\alpha(t^*) = 1$. K, the jumper removal congestion number, indicates the number of jumpers at which disconnects can no longer be removed. In practice, frames reaching this stage require complete replacement or extensive cleanup campaigns. Note that when disconnects are not being removed, the growth rate of $J(t)$ is the full rate of the connects. The assumed form of $\alpha(t)$ is

$$\alpha(t) = \begin{cases} \left[\dfrac{J(t)}{K}\right]^{\gamma}, & J(t) \leqq K \\ 1, & J(t) > K \end{cases} \tag{6}$$

where γ, a non-negative real number, indicates the rate at which $\alpha(t)$ approaches 1. To see the effect of the number γ on $\alpha(t)$, refer to Fig. 1. For $\gamma = 1$, $\alpha(t)$ increases linearly with $J(t)$. For $\gamma = 2$, $\alpha(t)$ starts more slowly with respect to $J(t)$ but then accelerates. The extreme case in

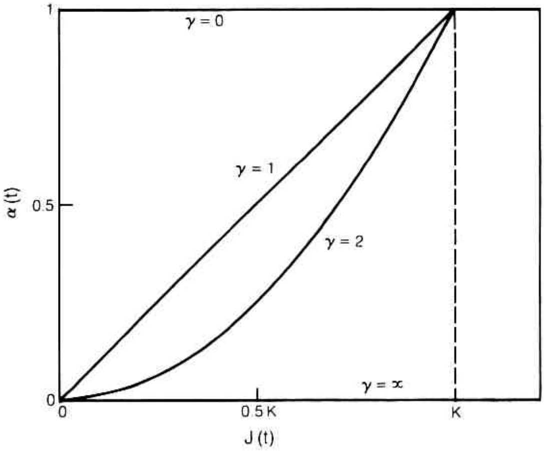


Fig. 1—Graph of $\alpha(t)$ vs $J(t)$.

which $\gamma = \infty$ corresponds to the case in which $\alpha(t) = 0$ for $0 \leqq t \leqq t^*$; this is the situation in which all disconnects are removed until the live jumper buildup exceeds K jumpers. For $\gamma = 0$, we have the extreme case in which $\alpha(t) = 1$ for all t, i.e., no disconnects are ever removed.

A number of factors influence the parameters K and γ. Those factors determining K are primarily frame parameters determined by the hardware of the frame and its use. One such factor is the frame configuration which includes such items as the number of verticals on the frame, the number of shelves, the positioning of various terminal densities on the vertical and horizontal sides of the frame, and the presence of bends in the frame (if any). Another factor influencing K is the jumper assignment procedure which, in part, determines neatness of the frame. For example, one might intuitively expect that preferential assignment would smooth out the jumper congestion inherent in random assignment. Thus, a frame would be able to accommodate a great number of jumpers before dead jumpers could no longer be removed. Hence, K would be larger under preferential assignment.

The factors determining γ are generally related to the human environment. One such factor is record quality. A frameman who attempts to remove a disconnect based on instructions derived from erroneous records may not be able to complete the task. In such a situation, the jumper in question will usually become a dead jumper. Other factors influencing γ are the numbers of both framemen and frame foremen and their skills and motivation.

Note that the model assumes that all hardware features and administrative procedures of a frame are tied together in the parameters K and γ. This feature of the model allows us to treat radically different configurations, assignment procedures, etc. This abstraction is in contrast to the work of Swanson[4] in which physical properties of the frame are considered explicitly.

If we substitute the form of $\alpha(t)$ given in eq. (6) into eq. (5), we get

$$J(t+1) = \begin{cases} J(t) + GS + \frac{1}{2}(1-G)S\left[\dfrac{J(t)}{K}\right]^{\gamma}, & t \leqq t^* \\ J(t) + \frac{1}{2}(1+G)S, & t > t^*. \end{cases} \tag{7}$$

Note that as t passes t^*, $\alpha(t)$ becomes 1, and the future growth in $J(t)$ is independent of K and γ. Given the parameters G, S, K, and γ, eq. (7) allows us to iteratively calculate the jumper buildup over time starting from some initial value $J(0)$. In numerical calculations, $J(0)$ is usually taken to be 0 though it would also be reasonable to interpret $J(0)$ as the number of jumpers in the frame at its cutover time, which might be a substantial positive quantity. Note that if $J(0)$ is not equal to 0, then $\alpha(0)$ is a number greater than 0. If one desires to grow $\alpha(t)$ from 0, he may replace $\alpha(t)$ in eq. (6) (and then eq. (7)) by

$$\alpha^*(t) = \begin{cases} \left[\dfrac{J(t) - J(0)}{K - J(0)}\right]^{\gamma}, & J(t) \leqq K \\ 1, & J(t) > K. \end{cases} \tag{8}$$

2.3 *Computer Program and Numerical Results*

A computer program was written to perform the iterative calculations of eq. (7). It is worth noting that for cases in which γ is a non-negative integer, it is unnecessary to perform the iterative scheme of eq. (7) since $J(t)$ can be obtained directly by solving a nonlinear first order differential equation.

The jumper buildup $J(t)$ is now examined for typical values of the parameters S, G, γ, and K. Let $S = 300$ service orders per day and let the gain fraction $G = 0.05$. Assume $\gamma = 2$ which seems to be reasonable. Typical estimates for reasonable values of K tend to be in the range of fifty to one hundred percent greater than the terminal capacity of the frame. By terminal capacity, denoted C, we mean that design parameter of the frame which indicates the maximum number of *live* jumpers that the frame can accommodate. For example, C might equal the number of horizontal terminals of the frame. We shall take $C = 80{,}000$ jumpers and $K = 120{,}000$ jumpers. Finally, let $J(0) = 0$.

The jumper buildup based on the above parameter values is graphed versus time (in days) in Fig. 2. The straight line in the graph indicates the buildup of live jumpers alone due to the gain GS. At the rate of $GS = 15$ jumpers per day, the terminal capacity C is exhausted after 5334 days or approximately 20 years considering 270 working days per year. At this point of 5334 days, $J(t)$ equals 446,033 jumpers with only 18 percent of these jumpers live. The point in the graph surrounded by a small box indicates the time $t^* = 3264$. At this time, $\alpha(t)$ has become equal to 1, and $J(t)$ is growing linearly at the rate of the connects $\frac{1}{2}(1 + G)S = 157.5$ jumpers per day or 10.5 times the rate of the gain.

Figure 3 considers various values of K for the fixed value of $\gamma = 2$. In this graph, $J(t)$ is plotted on a logarithmic scale. Notice that $J(t)$ decreases, for fixed values of t, as we increase K. The line corresponding to $K = \infty$ represents the case in which the jumper buildup is due to live jumpers alone through the gain GS.

In Fig. 4, we consider the case of $K = 120{,}000$ with various values of γ. Note that $J(t)$ decreases, for fixed values of t, as γ is increased. Note also that t^* is an increasing function of γ. We see that by controlling the factors that influence γ so as to increase γ, we postpone the time t^* at which point we no longer remove dead jumpers. The case of $\gamma = \infty$

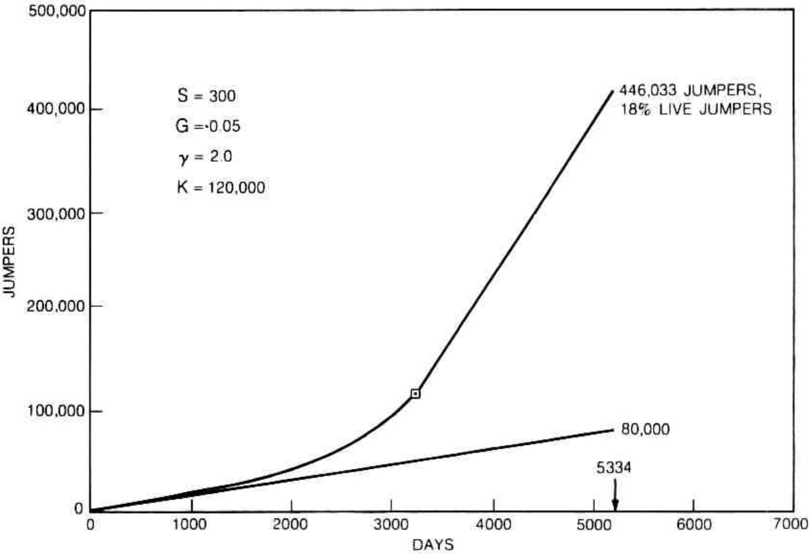


Fig. 2—Jumpers vs time for $\alpha = 2$ and $K = 120{,}000$.

corresponds to the ideal situation in which the jumper buildup is due only to live jumpers.

It should be stressed that the functions $J(t)$ given in Figs. 2, 3, and 4 follow from our assumed choice of parameter values. To actually use the model and predict jumper growth in a frame, one must estimate parameter values for the particular frame in question.

2.4 *Use of the Model*

One suggested use of the model is to predict the "breathing time" that a frame has before total congestion sets in. Presumably there is some total congestion number (in jumpers) such that when $J(t)$ reaches this number, the frame can no longer operate but must shut down. This total congestion number should not be confused with K, the jumper removal congestion number. Given the values of the parameters and the current jumper buildup, the model can be used to predict the time at which $J(t)$ equals the total congestion number. The time until this total congestion time is the "breathing time."

Any use of the model is predicated on our ability to make good estimates of the values of the model parameters S, G, K, and γ. Data is readily available for the parameters S and G. However, K and γ are not so easy to estimate. Though more work needs to be done on ways

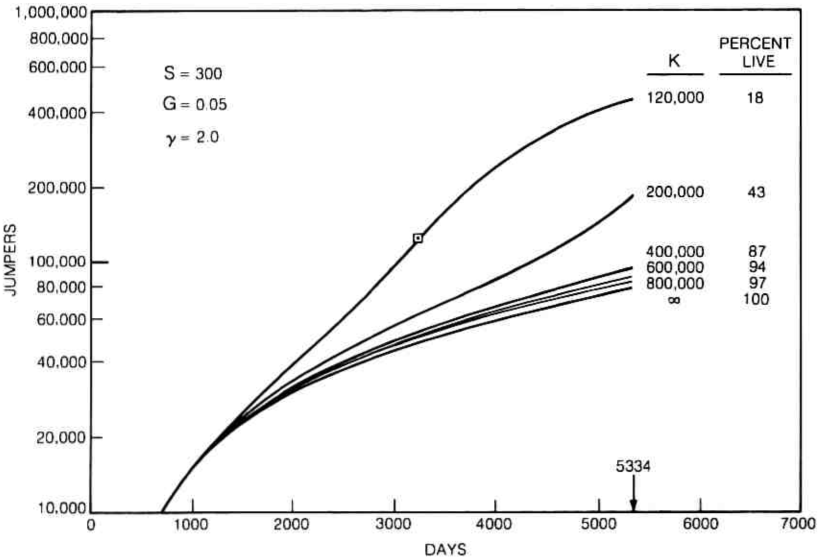


Fig. 3—Jumpers vs time for $\gamma = 2$.

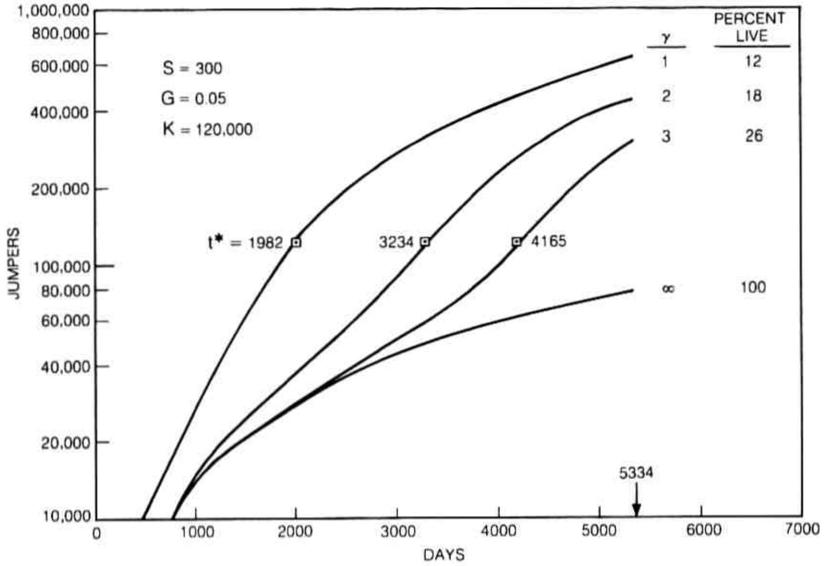


Fig. 4—Jumpers vs time for $K = 120{,}000$.

to properly estimate K and γ, it is sufficient to obtain an estimate of $J(t)$ for two or more (nonzero) values of t to determine these parameters.

Numerous refinements of the model can be considered. One could take S and G as functions of time as they most certainly are. In addition, although the form of $\alpha(t)$ given in eq. (6) is a reasonable starting point, more realistic functions need to be considered. One such effort in this direction is the model developed in the next section. That model which is built on a slightly different approach introduces parameters that are easier to estimate. Both friction and the weight of jumpers in the frame are explicitly considered as well as the percentage of errors in the assignment records.

III. DEVELOPMENT OF MODEL 2

3.1 *The Model*

We make the following assumptions:

(*i*) The frameman removes a disconnected jumper if the following two conditions are satisfied: there is no error in the assignment record of the jumper, and the force needed to pull it out is not greater than R_0 pounds.

(*ii*) The force required to pull out a jumper is proportional to its length, and to the total weight of all the jumpers which were put in the shelf after it.

(*iii*) The length of a jumper, the correctness of its record, and whether or not it will be disconnected at a given time are statistically independent events.

Let us denote by X_t the length of a live jumper which was installed at time t, let $F(x, t) = Pr(X_t \leqq x)$ be the distribution function of X_t, and let $\bar{X}_t$ denote the expected value of X_t. Let $B(t, \tau)$ be the expected total *length* of *all* jumpers at time t, which were installed in the interval $[t - \tau, t]$. Let $A(t, \tau)$ be the expected *number* of *live* jumpers at time t which were installed in the interval $[t - \tau, t]$.

Next, let $R(X, t, \tau)$ be the force required to pull out a jumper of length X at time t, given that it is of age τ. According to assumption (*ii*),

$$R(X, t, \tau) = R^*(t, \tau)X$$

where $R^*(t, \tau)$ is the force required per unit of length. $R^*(t, \tau)$ is a random variable proportional to the total weight, and hence to the total length, of all the jumpers which are of age $\leqq \tau$.

To simplify matters, we make $R^*(t, \tau)$ deterministic by replacing it by its expected value $aB(t, \tau)$ where a is a constant which depends on the weight per unit of length of a jumper, the dimensions of the shelf and the average friction of the jumpers (see Section 3.2). Note that the friction may be higher than that attributable to the theoretical friction coefficient of the jumper's coating, because the jumpers consist of twisted pairs and do not generally lie in straight lines. Thus we assume:

$$R(X, t, \tau) = aB(t, \tau)X. \tag{9}$$

Let

$$\hat{X}(t, \tau) = \frac{R_0}{aB(t, \tau)}. \tag{10}$$

Thus $\hat{X}(t, \tau)$ is the maximal length of a "pullable" jumper of age τ.

Let ϕ denote the probability of an incorrect record, and let $E(t, \tau)$ denote the average removed length of a disconnected jumper of age τ at time t. $E(t, \tau)$ is, according to assumption (*iii*), the product of the expected proportion of such live jumpers having correct records and the expected length of such live jumpers, given that their length is bounded by $\hat{X}(t, \tau)$. Hence,

$$E(t, \tau) = (1 - \phi) \int_0^{\hat{X}(t,\tau)} x d_x F(x, t - \tau). \tag{11}$$

We now introduce assumptions about the rates of connect and disconnect orders. Let us denote the connect order rate at time t by $\lambda(t)$. $\lambda(t)$ may depend on $A(t, t)$–the total expected number of live jumpers on the shelf, which represents the size of the main frame, or on $A(t, t_0)$ (where say $t_0 = 1$ year)–the recent growth of the shelf, or on external parameters such as changes in the community which is served by the central office.

Let $\mu(t, \tau)$ be the percent of disconnect orders for jumpers of age τ at time t. This rate will be, in many cases, independent of t. If a jumper is of age τ at time t, then the probability of a disconnect order for it in a small interval $[t, t + \Delta t]$ is approximately $\mu(t, \tau)\, \Delta t$.

Now let Δt be a time increment small enough so that quadratic and higher powers of Δt can be ignored. We then obtain the following system of equations:

$$\begin{aligned} B(t + \Delta t, \tau + \Delta t) &= B(t, \tau) - \Delta t \sum_{i=1}^{[\tau/\Delta t]} E(t, i\Delta t)\mu(t, i\Delta t) \\ &\qquad \cdot [A(t, i\Delta t) - A(t, (i-1)\,\Delta t)] + \lambda(t)\, \Delta t\, \bar{X}_t \\ A(t + \Delta t, \tau + \Delta t) &= A(t, \tau) - \Delta t \sum_{i=1}^{[\tau/\Delta t]} \mu(t, i\Delta t) \\ &\qquad \cdot [A(t, i\Delta t) - A(t, (i-1)\,\Delta t)] + \lambda(t)\, \Delta t \end{aligned} \tag{12}$$

for all t, τ such that $t \geqq \tau \geqq 0$, with the initial conditions:

$$\left.\begin{aligned} B(t, 0) &= 0 \\ A(t, 0) &= 0 \end{aligned}\right\}, \ t > 0$$

$$B(0, 0) = A_0 \bar{X}_0$$

$$A(0, 0) = A_0$$

where A_0 is the initial number of jumpers placed on the shelf.

By letting $\Delta t \to 0$, one obtains a system of partial differential-integral equations:

$$\begin{aligned} B_t(t, \tau) + B_\tau(t, \tau) &= -\int_0^\tau E(t, u)\mu(t, u)\, d_u A(t, u) + \lambda(t)\bar{X}_t \\ A_t(t, \tau) + A_\tau(t, \tau) &= -\int_0^\tau \mu(t, u)\, d_u A(t, u) + \lambda(t). \end{aligned} \tag{13}$$

Since $E(t, \tau)$ is nonlinear in the unknown function $B(t, \tau)$, it seems that an analytic treatment of these equations will not lead us far. On the other hand, a numerical solution is readily obtained by using the recursive equation given in eq. (12).

3.2 *An Explicit Formula for a*

The constant a can be evaluated in the following way. Denote

l—the length of the shelf.

w—the width of the shelf.

σ—the weight of a pair of jumpers per unit of length.

ρ—the number of pairs of jumpers in a unit of area in the cross-section of the shelf.

δ—the friction coefficient of the pairs of jumpers. (δ is the force needed to pull a pair per unit of vertical force which is applied on the pair.)

We assume that the buildup of jumpers is homogeneous throughout the length and the width of the shelf, and that one jumper lies higher than another jumper if and only if it was installed later. Likewise we assume that there are $w\sqrt{\rho}$ pairs of jumpers in any horizontal layer at any cross section of the shelf.

Let us consider a pair of jumpers of length X which was installed at time $t - \tau$, and let the present time be t. Then the total vertical weight on the pair is

$$\sigma B(t, \tau) \frac{X}{l} \frac{1}{w\sqrt{\rho}}.$$

Thus the force which is needed to pull it out is

$$R(X, t, \tau) = \delta\sigma B(t, \tau) \frac{X}{l} \frac{1}{w\sqrt{\rho}}.$$

Comparing this formula with eq. (9) we obtain

$$a = \frac{\delta\sigma}{lw\sqrt{\rho}}.$$

3.3 *Distributions of Jumper Lengths*

One of the major factors determining the behavior of the main frame is the distribution of jumper lengths. One might expect that the shorter the jumpers, the slower is the growth of the mass of jumpers on a frame. There are two reasons for this phenomenon: (*i*) short jumpers have small mass, and (*ii*) short jumpers are easier to pull out when disconnected. The distribution of jumper lengths is determined, and hence can be controlled, by the method used for assigning vertical terminals to horizontal terminals. Various administrative and planning methods have been proposed, such as preferential assignment, duplication of equipment terminals, etc. We believe that with the present model one can analyze the effect of any of these methods on the buildup of jumpers. To perform such an analysis one needs to determine the distributions $F(x, t)$ for the various methods. We present here the distributions for the case of random assignment, and for the case of preferential assignment with leakage.

Random assignment yields the following distribution:

$$F(x, t) = \begin{cases} \frac{1}{l^2}\, x(2l - x) & \text{for} \quad 0 \leqq x \leqq l \\ 1 & \text{for} \quad l < x \end{cases}$$

where l is the length of the shelf. Note that, in general, l will be a function of t due to growth of the frame.

By preferential assignment with leakage we mean that the MDF is divided into N zones with equal length l/N. We assume that there are two types of connect orders: type 1, which can be connected within the home zone, and type 2, which, for various reasons, are randomly assigned over *all* zones. Let ζ be the fraction of type 2 orders among all the orders. $0 \leqq \zeta \leqq 1$. ζ will be called the *leakage coefficient*. Our model is somewhat different from the model of Swanson.[4] There, the leakage is defined as the fraction of those orders which cannot be executed within one zone. Note that in our case, type 2 orders may still be executed in the home zone. If Swanson's leakage coefficient is denoted by ζ_s, then we have

$$\zeta_s = \frac{N - 1}{N}\, \zeta.$$

The distribution of jumper length for preferential assignment with N zones and leakage coefficient ζ is:

$$F(x, t) = \begin{cases} (1 - \zeta)\, \frac{xN^2}{l^2}\left(2\,\frac{l}{N} - x\right) + \zeta\, \frac{x}{l^2}\,(2l - x) & \text{for} \quad 0 \leqq x \leqq \frac{l}{N} \\ 1 - \zeta + \zeta\, \frac{x}{l^2}\,(2l - x) & \text{for} \quad \frac{l}{N} < x \leqq l \\ 1 & \text{for} \quad l < x. \end{cases}$$

As before, the length l of the frame, the number of zones and the leakage coefficient may be functions of t, that is, $l = l(t)$, $N = N(t)$ and $\zeta = \zeta(t)$. The case $\zeta = 1$ corresponds to the random assignment method.

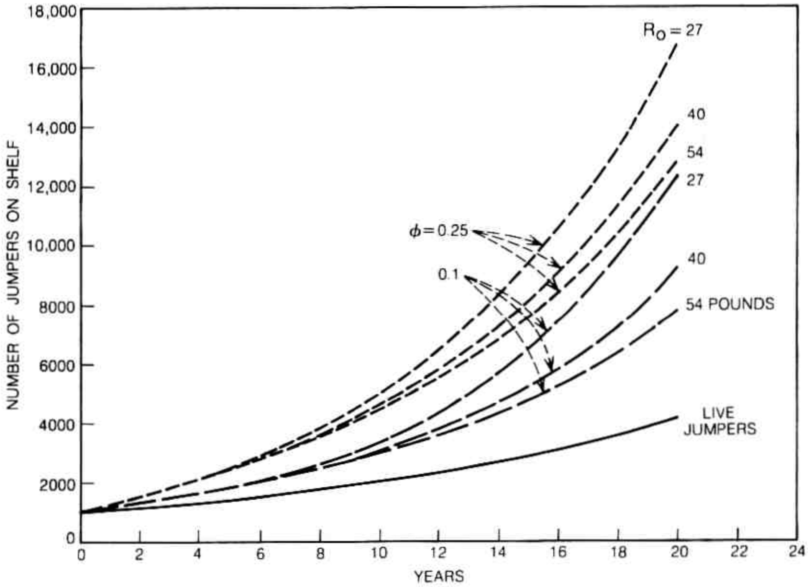


Fig. 5—Growth curves for various values of the maximal pulling force R_0, and the fraction of erroneous records ϕ, for random assignments.

3.4 *Computer Program and Numerical Results*

The model was programmed for the case of preferential assignment with leakage, where the length of the frame, the number of zones and leakage factor remain constant over time. As mentioned before, this includes random assignment as a special case. We made the following simplifying assumptions about the connect and disconnect processes:

(*i*) $\lambda(t) = \lambda_0 A(t, t)$, namely, the rate of connect orders is linear with respect to the total number of live jumpers.

(*ii*) $\mu(t, \tau) = \mu_0$, namely, the percent of disconnect orders is identical for all age groups of jumpers, and is independent of time.

These assumptions may be changed by minor modifications of the program.

The program solves the system of equations given in eq. (12) for a given time step size Δt. The user must specify $\lambda_0\ \Delta t$ and $\mu_0\ \Delta t$. Note that Δt should not be too large, otherwise eq. (12) will not be valid. A good criterion for the size of Δt is that $A(t, t)$ and $B(t, t)$ should not be much different from $A(t + \Delta t, t + \Delta t)$ and $B(t + \Delta t, t + \Delta t)$ respectively (say by 1 percent at most).

Other input parameters are: R_0, a, l, ϕ, ζ, A_0, N and finally n, the desired number of iterations, so that the equations are solved up to the time $T_{\max} = n\ \Delta t$. At the starting point for the iteration, $t = 0$, the shelf contains $A_0 = A(0, 0)$ jumpers.

The output gives, at various predetermined points in time, the number of live jumpers on the shelf, the total length of the live jumpers, the total length of all the jumpers, and the percentage of live jumpers in the mass on the shelf.

Sample runs of the program were made with the following input parameters:

$$\begin{aligned} \Delta t &= 1 \text{ month} \\ \lambda_0\ \Delta t &= 0.063 \\ \mu_0\ \Delta t &\equiv 0.057 \\ n &= 240 \text{ (20 years)} \\ A_0 &= 1000 \text{ jumpers} \\ l &= 100 \text{ ft} \\ a &= \frac{1}{150{,}000} \text{ lb/ft}^2. \end{aligned}$$

One set of runs was made for the case $\zeta = 1$ (random assignments) for values of ϕ ranging between 0 and 0.25, and values of R_0 ranging between 0 to 67 lb. The results are illustrated in Figs. 5, 6, and 7.

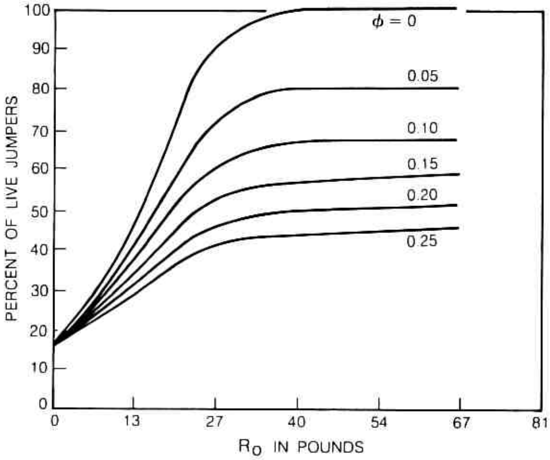


Fig. 6—Percent of live jumpers after 10 years, as a function of the maximal pulling force R_0 and the fraction of erroneous records ϕ, for random assignments.

A second set of runs was made for fixed $R_0 = 40$ lb and fixed $\phi = 0.1$ with values of ζ between 0 and 0.8 and $N = 4, 6, 8$. The results are illustrated in Fig. 8. The nominal data for this hypothetical shelf seems reasonably in line with some existing frames having approximately 7.2 percent annual growth of live jumpers; l corresponds to an MDF with 150 verticals.

The formula given previously in Section 3.2 can be used to evaluate the coefficient a. This formula requires knowledge of the friction coefficient. Since references for the magnitude of this coefficient do not appear to be readily available, we resorted to the following alternative approach.

By the same reasoning used to deduce eq. (9), we have

$$a = \frac{\tilde{R}}{\tilde{B}X}$$

where $\tilde{R}$ is the force needed to pull out the jumper, X is the length of the jumper, and $\tilde{B}$ is the total length of all the jumpers which are higher on the shelf relative to this particular jumper. Thus, a can be estimated by pulling jumpers from the shelf, noting the required force, the length, and the depth of the jumper in the pile.

Experiments of this type were conducted by Tengelsen:[5] "Readings taken thus far range from 39 pounds for a 29 foot jumper located deep in a pileup to a value of 9 pounds for a jumper approximately 100 feet long in the upper part of the pileup". We estimate that "deep in a pileup"

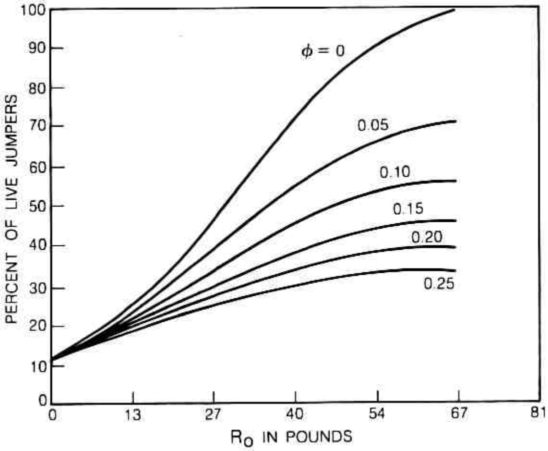
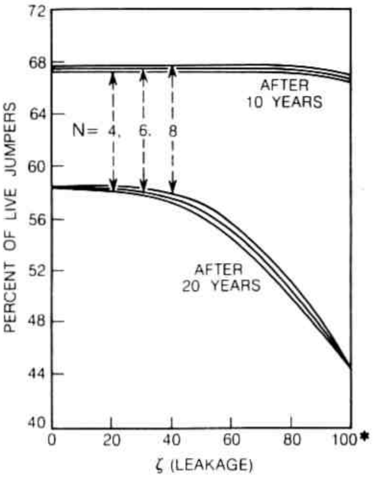


Fig. 7—Percent of live jumpers after 20 years, as a function of the maximal pulling force R_0 and the fraction of erroneous records ϕ, for random assignments.



* This corresponds to random assignment (with one zone).

Fig. 8—Percent of live jumpers as a function of the leakage coefficient and the number of zones N, for preferential assignments with $R_0 = 40$ lb and $\phi = 0.10$.

corresponds to $\tilde{B}$ of 200,000 feet. Thus

$$a = \frac{39}{29 \times 200{,}000} \sim \frac{1}{150{,}000} \text{ lb/ft}^2.$$

Finally, in the same paper (Ref. 5) it is stated that a value of 40 pounds was recommended as a safe working limit. This gave us an approximate upper limit for R_0.

From Fig. 5, one can deduce that for the range of parameters chosen, the accumulation of jumpers is more sensitive to ϕ than to R_0. This fact is also reflected in Figs. 6 and 7. In Fig. 6, we see that the percent of live jumpers on the shelf after 10 years is nearly independent of R_0 for $R_0 > 30$ lb. However, Fig. 7 indicates that after 20 years, the magnitude of R_0 is important. Figure 8 shows the influence of preferential assignment; we observe that dividing the frame into more than 4 zones seems to give no advantage when the leakage coefficient is fixed. However, one should be cautioned that, in practice, the leakage coefficient might be an increasing function of the number of zones.

IV. SUMMARY AND CONCLUSIONS

Two models for the growth of jumper buildup on the MDF have been developed. Given the values of various model parameters obtainable from data on frame characteristics and operations, these models permit the user to predict the jumper buildup on a frame.

Many suggestions have been made for rehabilitating MDF's which are either jammed or deteriorating because of dead jumpers. Should the effort be in the direction of automating the records to reduce ϕ, or in the direction of requiring less force to remove jumpers or enabling the framemen to pull harder? Does preferential assignment offer any advantages (for a particular MDF), and if yes, into how many zones should the MDF be divided? We believe that many of these questions can be answered with the aid of the models presented here.

REFERENCES

1. Dacey, G. C., "The Exchange Plant," Bell Laboratories Record, *50*, No. 2 (February 1972), pp. 34–35.
2. Osifchin, N., "Connective Systems: Evolving New Strategies," Bell Laboratories Record, *50*, No. 2 (February 1972), pp. 44–51.
3. Schumer, M. A., "A Combinatorial Analysis of the Main Distributing Frame: Spare Requirements for Conversion to Preferential Assignment from Random Assignment," B.S.T.J., *50*, No. 7 (September 1971), pp. 2465–2484.
4. Swanson, R. A., "Single Shelf MDF Jumper Analysis," unpublished work.
5. Tengelsen, R. A., "Common Systems—Main Distributing Frame—Analysis of Frame Capacity," unpublished work.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Multidimensional Polynomial Algebra for Bubble Circuits

By S. V. AHAMED

(Manuscript received November 17, 1971)

The paper expands the basic concepts of the coding theorists in the representation of data strings by algebraic polynomials, and develops the representation of both time and location of individual binary positions in the same polynomial. Further, it advances a set of algebraic operations on such polynomials to correspond to the various subfunctions that are accomplished in the actual domain circuits. The specific applications of the techniques proposed in this paper for the design and synthesis of such circuits is presented in a companion paper.

"Choose a set of symbols, endow them with certain properties and postulate certain relationships between them. Next, . . . deduce further relationships between them . . . . We can apply this theory *if* we know the "exact physical significance" of the symbols. . . . The applied mathematician always has the problem of deciding what is the exact physical significance of the symbols. *If* this is known, then at any stage in the theory we know the physical significance of our theorems. But the weakest link of physical significance is extremely fragile." The original source of this principium is J. E. Kerrick in *An Experimental Introduction to the Theory of Probability*, Belgisk Import Company, Copenhagen. It is also quoted in a slightly different form by F. M. Reza in *An Introduction to Information Theory*, McGraw-Hill Book Co., New York, 1961.

I. INTRODUCTION

Magnetic domains exist freely in thin platelets of orthoferrite crystals obtained by slicing them so that their crystalline axis is perpendicular to the surface of the platelets. Such domains are also present in very thin epitaxial garnet films (Fig. 1a) on suitable substrates. When the platelets or films are subjected to bias fields, these domains assume cylindrical shape and their diameter shrinks to microscopic sizes (Fig. 1b). Such domains (also called "bubbles") are stable under an appropriate bias field condition and they may be manipulated to perform[1,2] storage, gating, looping and also certain elementary logic functions.[3]

The domains are generally propagated from one location in the circuit to the next by subjecting them to the local bias field gradient. Basically there are two methods of providing such a field gradient to propagate the bubbles. In the "field access propagation,"[4] an alternating magnetization is imposed in a patterned soft magnetic overlay by an in-plane rotating magnetic field generated by a pair of coils carrying an alternating current. The coils completely surround the platelet with their axis in its plane. Two of the most commonly used overlay patterns are shown in Figs. 2a and b. During one cycle of the alternating current in the coils, the domains in the platelet move from one point in a pattern to the corresponding point in the adjoining pattern. This finite distance that the domain traverses during one cycle is defined as a "period." In

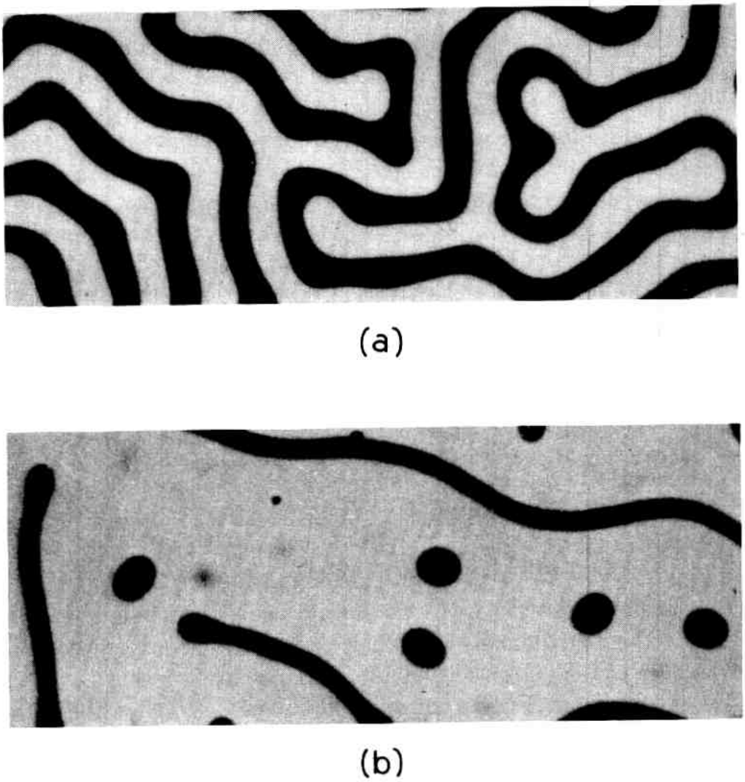


Fig. 1—(a) Magnetic domains as they are observed by Faraday effect in a typical epitaxial film 5 to 8 microns deep, deposited on Gadolinium-Gallium-Garnet (GGG) substrate 20 to 40 mils thick. Magnification 340. (b) Formation of "bubbles" from magnetic domains at a bias field of 30 Oe in same material used in Fig. 1a. Magnification 340.

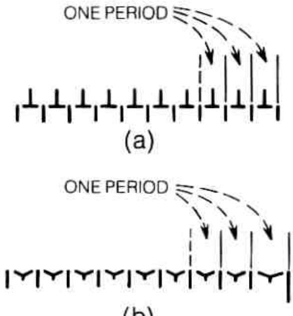


Fig. 2—(a) T-bar type of overlay used for field access propagation. (b) Y-bar type of overlay used for field access propagation.

the "conductor propagation," the local field gradient to move the bubbles is supplied by a current in a conductor (Fig. 3). A single phase current is generally used to periodically shift a bubble from one stable position to the next. Such stable positions are derived by a soft magnetic overlay also embedded on the platelet, and the periodicity of movement of the bubble from one position to the next depends on the frequency of the single phase excitation of the conductor. The distance which the bubble moves during one cycle of the single phase current is also defined as one "period."

Typical orthoferrites ($YbFeO_3$, $YFeO_3$, etc.) sustain 40 to 50 micron diameter bubbles, and the period is approximately 200 microns. Typical garnets ($Er_2Tb_1Al_{1.1}Fe_{3.9}O_{12}$ and $Gd_{2.3}Tb_{0.7}Fe_3O_{12}$) can support 4 to 8 micron diameter bubbles and the period is about 25 microns. The orthoferrites require about one micro-second to shift a bubble position by one period. The newer garnet materials also require about the same time, thus yielding a data rate of about one megacycle. It is customary to employ "bubble-no-bubble coding" with field access propagation and "lateral displacement coding" (LDC) with conductor propagation. In the former type of coding, the presence or absence of a bubble at an appropriate location denotes one or zero. In the latter type of coding, the bubble positions are coded as one or zero by laterally displacing them from one coding position to the other coding position (see Fig. 3).

All the bits of information are propagated by one period in one clock cycle in the field access propagation. In conductor drive circuits with lateral displacement coding, the average velocity during propagation is generally limited to one finite value, even though information bits are sometimes held stationary. When one finite velocity of propagation is

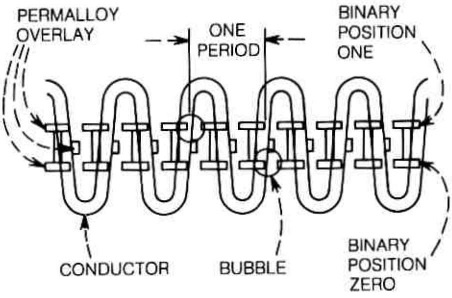


Fig. 3—Conductor propagation of bubbles.

assumed, it is possible to extend the capabilities of the conventional polynomial algebra used by coding theorists[5,6] to encompass the space-time relationships in bubble circuits. Bubble circuits perform in the time dimension and in the space dimension. The space dimension is further divided into subdimensions; the different sections (or elements) of the circuit which perform independently. Hence, any algebraic representation should encompass the representation of time and the representation of space dimensions which constitute the circuit. It is proposed that the time dimension be associated with X and the space dimension be associated with Y in the algebra.

II. DECOMPOSITION OF TIME, SPACE (CIRCUITS) AND FUNCTIONS

Consider the subclassification of time and space dimensions as follows:

The total time for a circuit to perform a function consists of a series of individual time intervals neccessary for the submodular functions. These individual time intervals can each be represented as a certain known number of clock cycles. Thus the unit of time is one clock cycle at the excitation frequency of the main field in field access propagation, or is one clock cycle at the drive circuit frequency in conductor propagation.

The physical layout of the circuit can be classified into various sections (elements such as paths, loops, functional modules, etc.). Each element further consists of a certain predefined number of periods. This leads to the unit of physical (or spatial) dimension as one period corresponding to one pole pitch in the T-bar, Y-bar or chevron pattern in field access propagation, or to one pole pitch of the driving conductor in conductor propagation.

Next consider the space-time relation. In field access propagation, the speed of the bubbles is one period per clock cycle, and is influenced by the angular velocity of the main rotating field; and stationary bubbles are rarely encountered.* In conductor propagation, the frequency of the exciting circuit determines the velocity;* and stationary bubbles are commonly encountered. However, in every circuit, over a limited duration and within a preselected element of the circuit, one can express the space-time relation with absolute certainty.

Finally, consider the overall algebraic representation. The entire circuit function is modeled by a series of algebraic operations, each one of which corresponds to a subfunction in the circuit. Each subfunction is carried out in the time dimension and in the space dimension. Hence, if we can resolve the function into its subfunctions, the circuit into its elements, and time into sets of clock cycles, and identify the individual subfunction with the circuit element and the appropriate set of clock cycles, then we can analyze and predict the functioning of the circuit with great accuracy. The representations of individual subfunctions by corresponding algebraic operations are developed in Section IV.

III. REPRESENTATION OF TIME, LOCATIONS AND BINARY VALUES OF A BIT POSITION

The origin of time may be chosen to be at any desired instant. However, a certain amount of flexibility and ease of representation results if the origin of time is chosen to coincide with a definite function in the circuit. Generally, bubble circuits perform repetitive functions and it may be convenient to choose the origin of time at the start of a repetitive cycle. When circuits perform a wide variety of nonrepetitive functions, then the analysis should be attempted for each function independently to ascertain the correct operation of each one of the functions. In the algebraic analysis of bubble circuits, it is proposed that the exponent of X (the carrier of time dimension as introduced earlier) be used to represent the number of clock cycles that have elapsed between a prechosen origin of time and the instant under consideration.

Further, it is proposed that the location of any given binary bubble position be represented by two components: (*i*) the element in which the binary position is presently located and (*ii*) the exact period in that

* The momentary variations of bubble speed at crossovers and compressors are ignored, and the entire distance is considered as one period. The effect of corners is dealt with separately.

element at which the binary position is currently present. This leads to designating a subscript to Y (the carrier of space dimension as introduced earlier) to indicate the element number and an exponent of Y to indicate the exact location within that element.

Finally, it is proposed that the binary value of a bubble position at a given instant of time be denoted by a. The binary value changes as a bubble position passes through the known transition points in the circuit. However, at a given time, which is a certain number of clock cycles past a preselected origin of time (a known exponent of X), and at a given location (known subscript and known exponent of Y), the binary value of a bubble position is either known, or it can be determined with absolute certainty from other circuit considerations.

3.1 *Representation of an Isolated Bubble*

Examine a single binary bubble position (Fig. 4a) the binary value of which is 'a' at an instant of time j clock cycles from a prechosen origin of time within the kth element of the circuit located at the lth location. Then it may be represented as

$$u = aX^jY_k^l.$$

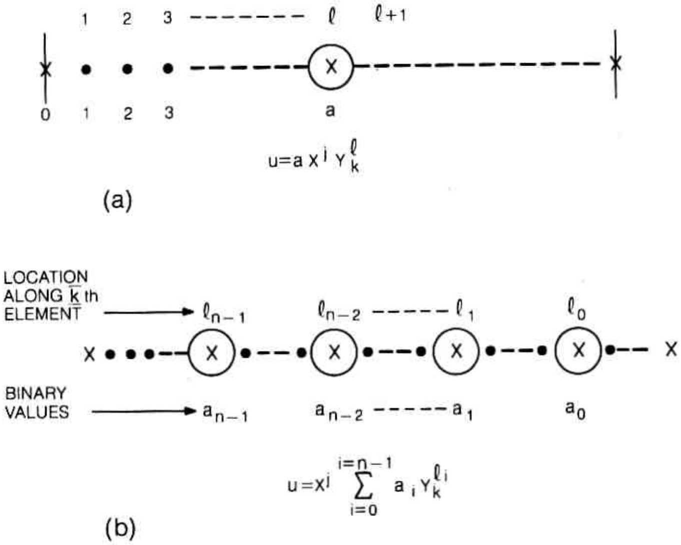


Fig. 4—(a) Representation of a single bubble. See text for explanation of j, k, and l. (b) Representation of a bubble stream j clock cycles after a prechosen origin of time.

3.2 *Representation of a Bubble Stream*

Consider a string of n binary bubble positions (Fig. 4b). Let the binary values of these positions be $a_0, a_1, \cdots, a_{n-1}$ during a clock cycle which is j clock cycles from a prechosen origin or time. If these binary positions are located in the kth element at location $l_0, l_1, \cdots, l_{n-1}$, respectively, then the string of data can be represented as

$$u = a_0 X^j Y_k^{l_0} + \cdots a_{n-1} X^j Y_k^{l_{n-1}} = X^j \sum_{i=0}^{i=n-1} a_i Y_k^{l_i}. \tag{1}$$

The sign of individual terms does not carry any significance. When the binary positions are adjacent to one another, then $l_0, l_1, \cdots, l_{n-1}$ are consecutive numbers.

Example 1: Consider four bubble positions the binary value of which is a_0, a_1, a_2 and a_3. If they occupy the 3, 2, 1, and 0 location of a sixth element after 28 cycles from a prechosen origin of time, then they may be represented as

$$u = X^{28}(a_0 Y_6^3 + \cdots a_3 Y_6^0) = X^{28} \sum_{i=0}^{i=3} a_i Y_6^{(3-i)}. \tag{2}$$

3.3 *Explanation of the Algebraic Representation*

From the point of view of circuit analysis, the algebraic representation leads to the following propositions.

(*i*) A series of bubble streams in a circuit are represented by a series of polynomials.

(*ii*) Any one bubble stream is represented by a particular polynomial (the sum of individual terms).

(*iii*) Each bit within a bubble stream is represented by a term (the product of components).

(*iv*) The binary value of the bubble position is represented by a.

(*v*) The number of clock cycles between a prechosen origin of time and the end of the cycle under consideration is the exponent of X.

(*vi*) The location of the bit of information is represented in two sections: the element within the circuit (the subscript of Y) and the location within the element (the exponent of Y).

3.4 *Implications of Representation*

The implications of the prechosen representation are:

(*i*) That a "snapshot" (i.e., a complete description of binary values of bubble positions and their respective locations) may be extracted from the algebra after a predetermined interval of time from a prechosen origin of time (a known exponent of X).

(*ii*) That there is a unique polynomial for every bubble stream in a circuit during any one cycle.

(*iii*) That all the pertinent information regarding all binary bits of information in a stream is available within the algebraic polynomial representing it.

IV. REPRESENTATION OF FUNCTIONS

4.1 *Generation of Bubble Streams*

Choose an origin of time at the end of the generation cycle of the first bubble position (i.e., as it is leaving the generator). If the generator is going to generate n binary positions the values of which are a_0, a_1, $\cdots$, and a_{n-1}, then at the end of $(n - 1)$ clock cycles, the binary string just generated may be written as

$$u = X^{n-1}(a_0 Y^{n-1} + \cdots a_{n-1} Y^0) = X^{n-1} \sum_{i=0}^{i=n-1} a_i Y^{(n-1-i)} \qquad (3a)$$

at an instant when the a_{n-1} position is just leaving the generator. The alternate representation of the string after n cycles is

$$u = X^n \sum_{i=0}^{i=n-1} a_i Y^{(n-i)}. \qquad (3b)$$

In (3a) and (3b), the exponents of Y indicate the locations along the bubble path which lead out of the generator. When the coefficients a_0, a_1, $\cdots$, a_{n-1} are consistently one, the action of an unconditional generator is represented (see Fig. 4 of Ref. 4, and Fig. 3 of Ref. 3 for T-bar and Y-bar configurations).

4.2 *Annihilation of Bubble Streams in Bubble-No-Bubble Coding*

The function of annihilation of any bubble stream u may be simply represented as

$$u = 0.$$

It is important to note that it is not identical to a polynomial in which all the binary bit positions are zero. When a conditional generator generates a string of binary bits which are all zero, it is still necessary to represent the binary string (3a) or (3b), since this string of data may interact with other strings at a later point in the circuit.

The function of a conditional annihilator may be represented as the action of a conditional gate (to be discussed later) and the action of an unconditional annihilator.

4.2.1 *Resetting of Streams in Lateral Displacement Coding*

In a functional sense, the function of resetting in Lateral Displacement Coding (LDC) is analogous to the function of annihilating in bubble-no-bubble coding. However, there is an important representational difference. The location of a bubble position which is reset in LDC is still identifiable, whereas the bubble position in field access circuits loses its identity and location upon entering the annihilator. Upon resetting in LDC, the bubble positions enter a "dead interval", or a sort of coma from the activity of the circuit, yet the algebra has to account for the elapsed time during which the bubble positions are in the reset state. Hence, a stream of n reset bubble positions in locations 0, 1, 2, $\cdots$, $n - 1$ may be represented in the usual way as

$$u = X^{j} \sum_{i=0}^{i=n-1} a_i Y^{n-1-i}, \tag{4}$$

where j represents the number of clock cycles from a prechosen origin of time, and where a_0, a_1, $\cdots$, a_{n-1} invariably represent the reset status. *Example 2:* The representation of 3 reset bubble positions in location 9, 5 and 1 after 20 clock cycles past a given time origin is

$$u = X^{20}(a_0 Y^9 + a_1 Y^5 + a_2 Y^1) = X^{20} \sum_{i=0}^{i=2} a_i Y^{(9-4i)}. \tag{4a}$$

4.3 *Functions of One Stream Resulting in One Stream*

Let u_p be the initial polynomial about to undergo the function, and let u_q be the resulting polynomial after m clock cycles. In a categorical sense, the operation may be represented as

$$u_q = F_m(u_p); \quad \text{or,} \quad F_m(u_p) \rightarrow u_q\,,$$

where F may represent: temporary freezing, translation in the forward direction, translation in the reverse direction, looping, special looping used in memory operations for dynamic data allocation, etc.

4.3.1 *Temporary Freezing of Bubble Streams in their Locations*

This function is generally encountered in conductor pattern and rail propagation with lateral displacement coding. Consider an n bit data stream j clock cycles from a prechosen origin of time, and represented as

$$u_p = X^{j} \sum_{i=0}^{i=n-1} a Y_k^l\,, \tag{5}$$

where a and l are a_i and l_i respectively. After freezing the movement of the stream for m clock cycles, we have the stream represented as

$$u_q = X^m \cdot u_p = X^{j+m} \sum_{i=0}^{i=n-1} aY_k^l \, . \tag{6}$$

If the bubble positions are all located at adjoining locations then

$$u_q = X^{j+m} \sum_{i=0}^{i=n-1} aY_k^{\langle l \rangle - i}, \tag{7}$$

where $\langle l \rangle$ is the location l_0 of the bubble position a_0 .

4.3.2 *Movement of Bubble Streams*

This is the most common bubble function and it is necessary to establish a sense of directionality in the movement. If the bubble stream is moving so that a_{n-1} bubble position of eq. (1) occupies position l_0 after a certain number of clock cycles, then a_0 would be the leading bubble. It is easier to work with a_0 as the leading bubble,* and this implies that $l_0 > l_1 > l_2 \cdots > l_{n-1}$. If the positions are in adjoining locations, then l_{i-1} is $l_i + 1$. Consider a stream of bubbles represented by u_p in eq. (5), which has moved in the forward direction for m clock cycles, then the resulting polynomial u_q is

$$u_q = X^m Y^m \cdot u_p = X^{j+m} \sum_{i=0}^{i=n-1} aY_k^{l+m}, \tag{8}$$

where a and l as defined earlier are a_i and l_i . If the data positions are in adjoining locations with a_0 in $\langle l \rangle$, then

$$u_q = X^{j+m} \sum_{i=0}^{i=n-1} aY_k^{\langle l \rangle - i + m}. \tag{9}$$

Generally, the bubble stream crosses elemental boundaries when it moves. If this stream shifts from element k to element t during the movement, and if their intersection is located at Y_k^z and Y_t^o (Fig. 5), then

$$u_q = X^{j+m} \sum_{i=0}^{i=n-1} aY_t^{l+m-z} \tag{10}$$

* This notation helps the circuit designer to comprehend the location and movement of the leading bubble position first, rather than comprehending the location and movement of the last bubble position first.

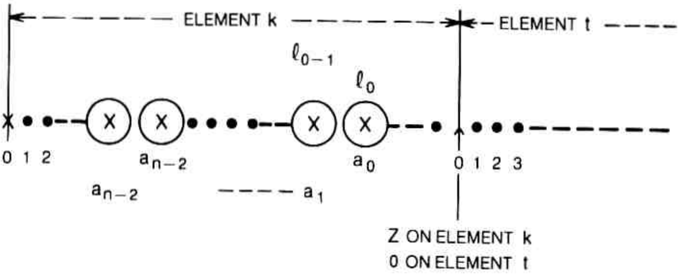


Fig. 5—Forward movement with a shift from element k to element t.

in general.* If the bubble positions are located in adjoining positions $l = l_i = l_0 - i = \langle l \rangle - i$ and

$$u_q = X^{j+m} \sum_{i=0}^{i=n-1} aY_t^{\langle l \rangle - i + m - z}. \tag{11}$$

Example 3: The representation of 7 bubble positions, the binary positions of which are a_0 through a_6 located at the 15th, 14th, $\cdots$, 9th locations of the 3rd element of a circuit at an instant of time, 36 clock cycles from a prechosen origin of time, is

$$u_p = X^{36} \sum_{i=0}^{i=6} a_i Y_3^{15-i}. \tag{10a}$$

Further, if this stream travels from element 3 to element 4, after traversing for 20 cycles with the boundary between elements 3 and 4 being located at Y_3^{20} and Y_4^0, then the final polynomial is represented as

$$u_q' = X^{56} \sum_{i=0}^{i=6} a_i Y_3^{35-i},$$

but

$$Y_3^{35-i} = Y_4^{35-i-20} = Y_4^{15-i},$$

[*] An alternate way to visualize the crossing of boundaries is to write

$$u_q' = X^{j+m} \sum_{i-0}^{i=n-1} aY_k^{l+m},$$

and then replace $Y_k^{z+\alpha}$ by Y_t^{α}, yielding $\alpha = l + m - z$, which leads to eq. (10). The prime indicates that the polynomial as such does not represent an observable stream but will do so after the next operation (s).

and thus

$$u_q = X^{56} \sum_{i=0}^{i=6} a_i Y_4^{15-i}. \tag{11a}$$

The exponent of Y_4 becomes negative if the stream does not traverse at least 11 cycles when it is 7 bits long, or if the stream traverses for 20 cycles but is 16 bits long. It is thus possible to have the stream segmented as u_p and u_q by an improper choice of the traversing time. Sometimes negative exponents may be carried for a few steps in the analysis without interpreting polynomials as streams during these steps.

Examine a bubble stream represented by u_p in eq. (5) having moved in the backward direction for m clock cycles, then

$$u_q = X^m Y^{-m} \cdot u_p = X^{j+m} \sum_{i=0}^{i=n-1} a_i Y_k^{l-m},$$

where a and l are a_i, and l_i respectively, and a_0 is the last bubble in this case. If the bubble stream moved from element k and entirely shifted in tth element, then

$$u_q = X^{j+m} \sum_{i=0}^{i=n-1} a_i Y_t^{z-(l-m)}, \tag{12}$$

where* the intersection of elements k and t is located at Y_k^0 and Y_t^z. If the bubble positions are in the adjoining locations, then $l_i = l_0 - i$ where l_0 is the location of the bubble a_0 in u_p before its transformation to u_q.

Example 4: The representation of a 4 bubble stream, the binary values of which are a_0, a_1, a_2 and a_3 located in the 7, 5, 3, 1 periods of element 5 at an instant 35 clock cycles after a prechosen origin of time, is

$$u_p = X^{35} \sum_{i=0}^{i=3} a_i Y_5^{7-2i}. \tag{12a}$$

After 9 cycles of backward movement, if the stream is in element 3 with the boundary of elements 5 and 3 located at Y_5^0 and Y_3^{20}, then

$$u_q = X^{44} \sum_{i=0}^{i=3} a_i Y_3^{18-2i}. \tag{12b}$$

* An alternate way to visualize the crossing of the boundary is to write

$$u_q' = X^{j+m} \sum_{i-0}^{i=n-1} a_i Y_t^{l-m},$$

and then replace $Y_k{}^\alpha = Y_t{}^{z+\alpha}$, when α is ≤ 0, thus yielding $z + \alpha = z + (l - m)$ which leads to eq. (12).

4.3.3 *Looping of Bubble Streams*

Looping is generally encountered in T-bar or Y-bar propagation when a fixed delay or storage is necessary for bubble streams without actually freezing their movement.

Consider a bubble stream represented by u_p in (5), which is circulating in a loop with g periods[*] for m clock cycles, then

$$u_q = X^m Y^m \cdot u_p = X^{i+m} \sum_{i=0}^{i=n-1} a Y_k^{l+m},$$

but in a loop $(l + m = (l + m) \bmod g)$ thus leading to

$$u_q = X^{i+m} \sum_{i=0}^{i=n-1} a Y_k^{(l+m) \bmod g}, \tag{13}$$

where a and l are read as a_i and l_i respectively, and $l_i = (l_0 - i)$ if the bubble positions are in the adjoining location with a_0 as the leading bubble.

Example 5: The representation of 26 bubble positions[†] $a_0, a_1, \cdots, a_{25}$, occupying locations 28, 27, $\cdots$, 3 in a loop (element 5) at an instant 1214 clock cycles from a prechosen origin of time is

$$u_p = X^{1214} \sum_{i=0}^{i=25} a_i Y_5^{28-i}. \tag{13a}$$

After looping for 980 clock cycles, the final representation is

$$u_q = X^{2194} \sum_{i=0}^{i=25} a_i Y_5^{(1008-i) \bmod 39} = X^{2194} \sum_{i=0}^{i=25} a_i Y_5^{33-i}. \tag{13b}$$

4.3.4 *Movement Around Corners*

Corners[‡] in T-bar and Y-bar circuits need special attention since bubble positions lose or gain a quarter period when a 90-degree turn is present. In most cases, their effect may be eliminated by considering a movement to span two or an even number of compensating 90-degree corners. However, when it is necessary to predict the movement of

[*] The number of T-bar periods should be considered as the number of clock cycles to bring back the leading bubble to its original location in the loop. The orientation of the T-bars and the direction of rotation both play an important part in the determination of the number. In any case, the value of g is the actual number of T-bars ±1 depending on the orientation of T-bars and the direction of rotation.

[†] One encounters this bubble stream in (39, 26) the shortened BCH encoder with magnetic domains constructed along the same principles as discussed in Section 3 of Ref. 7.

[‡] When the effect of the corners in a loop is being considered, the value of k used should be (k mod 4), and the number of periods in the loop should be measured as indicated in Sec. 4.3.3. Else the value of k should be its real value.

streams which include k number of 90-degree unidirectional corners, then a polynomial u_p in eq. (5) becomes

$$u_q = X^m Y^{m \pm k/4} \cdot u_p \,, \tag{14}$$

where the sign is determined by the direction of the turn with respect to the direction of rotation of the main driving field and the orientation of the T-bars.

No special algebraic consideration is necessary in conductor pattern and rail propagation.

Example 6: Consider the two bubble positions represented as

$$u_p = X^{20}(a_0 Y_4^7 + a_1 Y_4^6). \tag{14a}$$

If these positions traverse for 13 cycles, and encounter three unidirectional -90-degree turns in the path, then the final polynomial is

$$u_q = X^{33} Y^{(13-\frac{3}{4})} \cdot u_p = X^{33}(a_0 Y_4^{19\frac{1}{4}} + a_1 Y_4^{18\frac{1}{4}}). \tag{14b}$$

The fractional exponent of Y indicates that the binary positions a_0 and a_1 are lagging 270-degrees behind their corresponding positions had there been no turns. If it is necessary to bring them back in phase, then an additional $\frac{3}{4}$ clock cycle is required, and the polynomial u_q would be

$$u_q = X^{33\frac{3}{4}}\,(a_0 Y_4^{20} + a_1 Y_4^{19}). \tag{14c}$$

4.3.5 *Routing of Bubble Streams*

This function plays a critical role when it is necessary to transfer streams of binary information into a certain branch of a circuit at a node. The algebraic representation after this function is identical to the polynomial u_q in eq. (10), when the polynomial crosses the boundary of one element k at Y_k^z and enters another element t at Y_t^0. In the general case, when the bubble stream is channeled from an element p at Y_p^z, and enters another element t at $Y_t^{(z)}$, u_q may be represented as

$$u_q = X^{i+m} \sum_{i=0}^{i=n-1} a_i Y_t^{l+m-z+(z)}, \tag{15}$$

where a and l as defined earlier are a_i and l_i respectively.

4.3.6 *Inverting the Binary Content of Bubble Streams*

Consider a polynomial

$$u_p = X^i \sum_{i=0}^{i=n-1} a_{ip} Y_k^l \,. \tag{16}$$

If the bubble stream so represented has gone through an inverting gate during the following m cycles, then the resulting polynomial may be written as

$$u_q = X^{i+m} \sum_{i=0}^{i=n-1} a_{iq} Y_k^{l+m}, \tag{16a}$$

where l denotes l_i and

$$a_{iq} = \bar{a}_{ip} \, .$$

Example 7: Consider 3 data positions in an LDC circuit. If a_0, a_1, a_2 denote their status at an instant 49 clock cycles from a time origin, and are located in the seventh element at locations 17, 13 and 9 then,

$$u_p = X^{49} \sum_{i=0}^{i=2} a_{ip} Y_7^{17-4i}. \tag{16b}$$

If these binary positions pass through an inverting gate located at Y_7^{20} and Y_9^3, then the bubble stream after 15 clock cycles is

$$u_q = X^{64} \sum_{i=0}^{i=2} a_{iq} Y_9^{15-4i}. \tag{16c}$$

4.3.7 *Opening and Closing Gaps in Bubble Streams*

This function is quite effectively used in dynamic data reallocation with T-bars. The bubble stream in the loop has two preferred paths, one for each direction of rotation. In one direction, the stream traverses $(n + 1)$ periods in the loop, and in the other, it traverses (n) periods in the loop. (See Fig. 6). If after z clock cycles of clockwise movement of a bubble stream in the loop the direction is reversed for one clock cycle, then the data bit at the nth (Fig. 6) location is at the 0th location, and the data position which was at $((z - 1) \bmod n)$ location is at the nth location. Now z clock cycles of anticlockwise rotation would have effectively included the data position at the nth bit at the $(z \bmod (n + 1))$ location, and all the remaining positions one location behind their original locations. A converse process takes place for excluding a data bit position in the stream.

The algebraic equivalent of this function may be represented as follows:

Let the numbering of the locations in the loop be in the direction (anticlockwise in Fig. 6) which permits the stream to traverse n periods in the loop. Let the contents of the loop (without the nth position) be represented as

$$u_p = X^i \sum_{i=0}^{i=n-1} a_i Y^i. \tag{17}$$

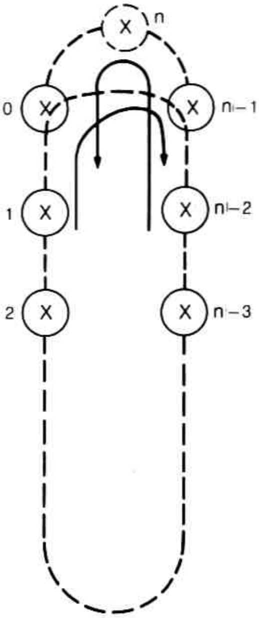


Fig. 6—Schematic representation of the dynamic data reallocation function.

The polynomial u_p^* undergoes a shift in clockwise direction for m (with $m < n - 1$) clock cycles. The resulting polynomial u_q becomes

$$u_q = X^{i+m} \sum_{i=0}^{i=n-1} a_i Y^{(i-m) \bmod n}. \tag{18}$$

As a distinct step from the above representation, let

$$u_p = X^i \sum_{i=0}^{i=n} a_i Y^i \tag{19}$$

represent the contents of the loop including the nth location before a series of counter clockwise shifts for m clock cycles. The resulting polynomial is

$$u_q = X^{i+m} \sum_{0}^{n} a_i Y^{(i+m) \bmod (n+1)}. \tag{20}$$

Example 8: Consider a string of bubbles the binary values of which are a_0, a_1, $\cdots$, a_7 at an instant 92 clock cycles from a time origin located at 0, 1, 2, $\cdots$, 7 in a memory loop (Fig. 6) with 255 (i.e., $n - 1$)

[*] There is no need for a subscript for Y since only one element (the loop) is being considered.

periods in the clockwise direction and 256 periods in the anticlockwise direction. This can be represented as

$$u_p = X^{92} \sum_{i=0}^{i=7} a_i Y^i. \tag{17a}$$

Let the 256th (i.e., nth) position contain the bubble position a_n at the same instant of time. Four cycles of rotation of the field causing a clockwise circulation of binary positions yields

$$u_q = X^{96}\left\{\sum_{i=0}^{i=3} a_i Y^{252+i} + \sum_{i=4}^{i=7} a_i Y^{i-4}\right\}. \tag{18a}$$

In the first four terms in the polynomial (38), $(i - 4) \bmod 256$ would correspond to 252, 253, 254 and 255 for i ranging from 0 to 3.

Now if the field is rotated for 5 clock cycles in the opposite direction, resulting in an anticlockwise shift of bubble positions, then the resulting polynomial is

$$u = X^{101}\left\{\sum_{i=0}^{i=3} a_i Y^{(257+i) \bmod 257} + a_n Y^4 + \sum_{i=4}^{i=7} a_i Y^{i+1}\right\},$$

or

$$u = X^{101}\left\{\sum_{i=0}^{i=3} a_i Y^i + a_n Y^4 + \sum_{i=4}^{i=7} a_i Y^{i+1}\right\}. \tag{20a}$$

In effect it is seen that the bubble position after the nth location has been inserted at the 4th location in the bubble stream.

4.4 *Functions of One Stream Resulting in Two or More Streams*

Let u_p be the initial polynomial about to undergo the function resulting in two polynomials u_q and u_r. In general, functions of this type may be represented as

$$u_q + u_r = F_m(u_p); \quad \text{or,} \quad F_m(u_p) \rightarrow u_q + u_r,$$

where F* represents duplication or addressing sections of the initial polynomial u_p into one or the other branch of a circuit. A finite number of clock cycles (m) are allowed during which the operation takes place.

4.4.1 *Duplication and Replication of Bubble Streams*

Duplicators may require a certain finite number of clock cycles to

* These functions are used extensively in the companion paper[8] dealing with the applications of the algebra.

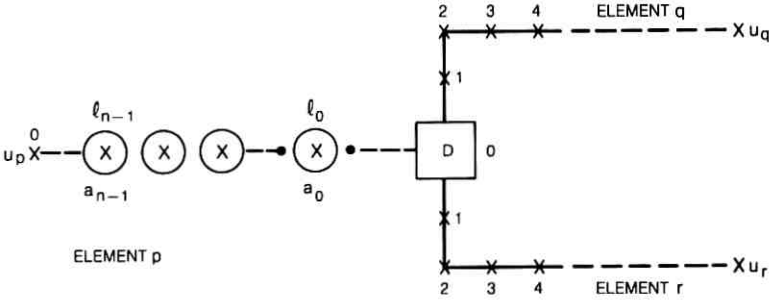


Fig. 7—The bubble stream u_p entering the duplicator D.

operate. In the T-bar propagation, one clock cycle is necessary to duplicate. The effect may easily be included in the algebra by considering the duplicator as an extra period of element p with u_p (see eq. 5), and the zero location of elements q and r in which u_q and u_r will be positioned after m clock cycles. (See Fig. 7).

Consider a string of binary data represented as u_p in eq. (5) to be duplicated during the m clock cycles yielding u_q and u_r represented as

$$u_q = X^m Y^{m-d} \cdot u_p = X^{i+m} \sum_{i=0}^{i=n-1} a Y_q^{l+m-z-d} \tag{21}$$

$$u_r = X^m Y^{m-d} \cdot u_p = X^{i+m} \sum_{i=0}^{i=n-1} a Y_r^{l+m-z-d}, \tag{22}$$

where a and l are a_i and l_i respectively, the duplicator is located after z periods in k, and d is the number of clock cycles to accomplish the duplication. It is to be noted that when the bubbles occupy adjoining locations, d cannot exceed 1 for satisfactory duplication. Under such conditions*

$$u_q = X^{i+m} \sum_{i=0}^{i=n-1} a Y_q^{\langle l \rangle - i+m-z-1} \tag{23}$$

and a similar expression for u_r , where $\langle l \rangle$ denotes l_0 .

* An alternate way to visualize this transformation is to consider that the duplicator is located at the intersection of Y_p^{z+1}, Y_q^0 and Y_r^0 which leads to $Y_p^{\alpha} = Y_q^{\alpha-(z+1)} = Y_r^{\alpha-(z+1)}$, when $\alpha > z + 1$, thus yielding

$$u_q' = X^{j+m} \sum_{i-0}^{i=n-1} a Y_{p^{\langle l \rangle - i+m}},$$

where $\langle l \rangle$ is the location l_0 of the first bubble position a_0 prior to duplication. This leads to u_q in eq. (23).

Replication leads to a series of resulting polynomials u_q , u_r , $\cdots$, etc., but the algebraic treatment is exactly the same.

4.4.2 *Addressing Sections of a Stream into Different Branches of a Circuit*

Consider a polynomial u_p in element p which will approach a gate. The gate operating for g clock cycles will address the first n' bubble positions in q and the rest into r. This function can be visualized as the effect of two independent translatory functions: (see Section 4.3.1) (*i*) the first n' positions are translated in the forward direction and change elements from p to q, and (*ii*) the remaining positions are translated in the forward direction and change element from p to r. Algebraically this can be expressed as follows:

$$u_p = X^i \sum_{i=0}^{i=n-1} aY_p^l = X^i \sum_{i=0}^{i=n'-1} aY_p^l + X^i \sum_{i=n'}^{i=n-1} aY_p^l \tag{24}$$

with

$$u_q = X^{i+m} \sum_{i=0}^{i=n'-1} aY_q^{l+m-z} \tag{25}$$

and

$$u_r = X^{i+m} \sum_{i=n'}^{i=n-1} aY_r^{l+m-z}, \tag{26}$$

where a and l are a_i and l_i respectively (see Section 5.3.1). The gate should be located at Y_p^z , Y_q^0 and Y_r^0 . It can be seen that the first bubble position does not reach the gate till $(z - (l_0 + 1))$ clock cycle, and to divert the first n' positions the gate should be operating to divert into element q for exactly $(l_0 - l_{n'} + 1)$ clock cycle, leading to the design detail that the gate should divert into q for g clock cycles where $g = (z - l_{n'} + 1)$. If the bubble positions are in the adjoining location, then the gate has to operate diverting into q for n' clock cycles starting after $(z - (l_0 + 1))$ clock cycles. Further, it has to act for the next $(n - n')$ clock cycles to divert the bubble position into r.

When a gate addresses various sections of a data stream into more than two elements, the algebraic representation is similar. Such a condition exists if m is not chosen large enough in the previous case, and then there will be two bubble strings u_q and u_r together with a section of u_p , which has not been processed by the gate.

Example 9: Consider a data stream* in element 0 of a circuit. Eight data

* Such a data stream is encountered in general rate change circuits represented in Fig. 1 and 1a of Ref. 9.

bits a_0 through a_7, 12 clock cycles after a prechosen origin of time, occupy location Y_0^{12} through Y_0^5. The polynomial representing the stream is

$$u_p = X^{12} \sum_{i=0}^{i=7} a Y_0^{12-i}. \tag{24a}$$

If the data stream passes through a gate at Y'^{13}_0 which diverts the first 4 bits into element Y_1 and the last four into Y_2, then after 14 clock cycles the resulting polynomial u_q in 1 and u_r in 2 may be represented as

$$u_q = X^{26} \sum_{i=0}^{i=3} a_i Y_1^{13-i}, \tag{25a}$$

and

$$u_r = X^{26} \sum_{i=0}^{i=3} a_{i+4} Y_2^{9-i}. \tag{26a}$$

4.5 *Functions of Two or More Streams Resulting in One Stream*

Consider two streams u_p and u_q (Fig. 8) interacting to yield one stream u_r at a gate ⓖ, where ⓖ may denote the function of logical gating, combining, etc. This function is different from the previous functions, since the individual binary values a_i in the polynomials are likely to be changed by this function. Let

$$u_p = X^i \sum_{i=0}^{i=n-1} a_{ip} Y_p^{l(ip)}, \tag{27}$$

and

$$u_q = X^i \sum_{i=0}^{i=n'-1} a_{iq} Y_q^{l(iq)}, \tag{28}$$

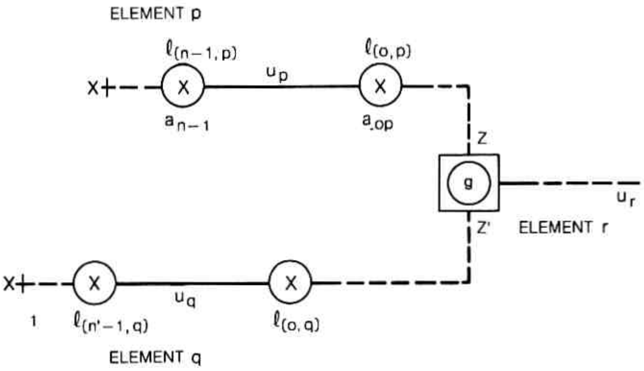


Fig. 8—Gating or merging of u_p and u_q to yield u_r.

where $l\langle ip\rangle$ and $l\langle iq\rangle$ represent l_{ip}, and l_{iq} respectively. The number of locations including the first and the last binary position of u_p is $(l_{0p} - l_{(n-1,p)} + 1)$; (i.e., n if adjoining locations). It can be seen (Fig. 8) that the length of the resulting bubble stream, u_r, is the largest number n'' of the following four numbers*

$$(i) \qquad l_{0p} - l_{(n-1,p)} + 1,$$

$$(ii) \qquad l_{0q} - l_{(n'-1,q)} + 1,$$

$$(iii) \qquad (l_{0p} - l_{(n-1,p)}) + 1 + z - l_{0p} - (z' - l_{0q})$$
$$= (z - l_{(n-1,p)}) - (z' - l_{0q}) + 1,$$

or

$$(iv) \qquad (l_{0q} - l_{(n'-1,q)}) + 1 + z' - l_{0q} - (z - l_{0p})$$
$$= (z' - l_{(n'-1,q)}) - (z - l_{0p}) + 1,$$

where z and z' denote the number or periods along elements p, and q at which the gate $\textcircled{g}$ is located (see Fig. 8). The polynomial u_r can be written as

$$u_r = X^{i+m} \sum_{i=0}^{i=n''-1} a_{ir} Y_r^{l\langle ir\rangle}, \tag{29}$$

where $l\langle ir\rangle$ represents l_{ir} respectively, and

$$a_{ir} = a_{ip} \textcircled{g} a_{iq} \tag{30}$$

$$l_{ir} = l_{ip} + m - z, \tag{31a}$$

or

$$l_{ir} = l_{iq} + m - z'. \tag{31b}$$

The subscripts for a and l should be chosen with adequate care to consider only the interacting binary positions in streams u_p and u_q that pass through the gate $\textcircled{g}$ simultaneously. It is important to note that each term in u_r results from a term in u_p, or in u_q, or from terms in both. When there is no term in one polynomial (u_q or u_p) corresponding to a particular term in the other (u_p or u_q), then the appropriate equation (31a or b) for the exponent of Y should be chosen. When there is a term in one polynomial (u_q or u_p), corresponding to a given term in (u_p or u_q),

* (i) or (ii) indicates the length u_p or u_q with the longer stream u_p or u_q completely overlapping u_q or u_p respectively. (iii) or (iv) indicate partial overlap, with u_q or u_p being nearer the gate.

then the location calculated from l_{ip} or l_{iq} will yield the same result for l_{ir}.

Example 10: Consider a bubble stream u_p with four binary bits a_{0p}, a_{1p}, a_{2p} and a_{3p}, after 41 clock cycles from an origin of time located at 3, 2, 1 and 0 of element 5 and represented as

$$u_p = X^{41} \sum_{i=0}^{i=3} a_{ip} Y_5^{3-i}. \tag{27a}$$

Consider a next stream u_q with seven bubble positions u_{0q} through u_{6q}, also 41 clock cycles from the prechosen origin of time occupying 10, 9, 8, $\cdots$, 4th locations of element 6, and represented as

$$u_q = X^{41} \sum_{i=0}^{i=6} a_{iq} Y_6^{10-i}. \tag{28a}$$

If u_p and u_q approach an exclusive-or gate ⓖ located at Y_5^6, Y_6^{11}, and Y_7^0, then the polynomial u_r after 12 cycles can be derived as follows: a_{0q}, a_{1q} pass through the gate during the first three cycles while u_p is still approaching the gate. The binary positions a_{2q} through a_{5q} interact with a_{0p} through a_{3p} in the gate ⓖ for the next four cycles. a_{6q} passes through the gate and the gating is now complete. During the last five cycles, the bubble stream in element Y_7 moves away from the gate thus leading to the resulting polynomial

$$u_r = X^{53} \sum_{i=0}^{i=6} a_{ir} Y_7^{11-i}, \tag{29a}$$

where a_{0r}, a_{1r} and a_{6r} are a_{0q}, a_{1q} and a_{6q} respectively, and

$$a_{ir} = a_{ip} \oplus a_{iq} \; (i = 2 \text{ through } 5). \tag{30a}$$

This example corresponds to the case (i) in Section 4.5.

4.6 *Functions of Two or More Streams Resulting in Two or More Streams*

This function, though rarely encountered in normal bubble circuits, can still be conveniently represented as an integral procedure of many subfunctions in which two or more streams result in one stream. If g_1, g_2, g_3 $\cdots$ are individual functions yielding streams 1, 2, 3 $\cdots$, etc., then the algebraic representation of Section 4.5 may be extended to represent streams 1, 2, 3 $\cdots$, etc. One such example is presented in Section 2 of Ref. 8.

V. OVERALL BUBBLE CIRCUIT FUNCTIONS

We have a set of mathematical tools to predict the binary values and locations of individual bubble positions as the binary streams undergo different submodular functions within the circuit. The interval of time chosen for these submodular functions in Section IV is 'm' clock cycles. To study the overall circuit function, the function is divided into a series of submodular functions (F_1, F_2, $F_3 \cdots$); the circuit which performs the functions is divided into a series of elements

$$(1, 2, 3 \cdots, k, k + 1, \cdots),$$

and the time necessary to accomplish the function is divided into a series of (m_1, m_2, $m_3 \cdots$, etc.) clock cycles.

5.1 *Subdivision of Circuit into Elements*

After isolating the subfunctions within the overall circuit function, the elements that accomplish these subfunctions may be identified. A series of fine functional subdivisions may be necessary to identify the particular circuit elements.

Some of the examples of the elements are storage paths, transmission paths, loops, etc. Sometimes a particular element (k) of a circuit designed is very short, and it cannot accommodate the entire string of data. When it is still desired to study the contents of the n bit binary string of the polynomial u_p in that element, then it is possible to fictitiously extend the element to just accommodate the n data bits. With the observer located at a preselected period (say Y_k^b) in the element, the binary values of data which flow past this location would still be the values of a_0, a_1, $a_2 \cdots a_n$ in the calculated polynomial u_k. The instant of incidence of the leading bubble a_0 would be $(n - b)$ clock cycles prior to its value as predicted by the exponent of X(i.e., j_0) in the polynomial u_k.

5.2 *Subdivision of Time*

Subfunctions are accomplished by elements within specified intervals of time. The interval of time for a specific subfunction is almost entirely determined by the circuit parameters and the clock frequency. Any one particular interval of time may, however, be conveniently expressed as a certain number of clock cycles.

These different values of clock cycles m_1, m_2, m_3, $\cdots$, etc., are necessary to calculate the polynomials u_1, u_2, u_3, $\cdots$, etc., and they in turn uniquely define the values, intervals and positions of binary bits in the circuit. Generally, the subfunctions may proceed in series or in parallel, and different bubble streams may simultaneously undergo different functions in different elements of the circuit. However, the complete function of a circuit starts and finishes at an instant of time. Hence, whenever subfunctions are accomplished in parallel, it should be realized that the summation of m_i does not equal the total number of clock cycles necessary for the complete circuit function. Each subfunction proceeding simultaneously should be modelled independently by its corresponding algebraic operation.

VI. CONCLUSIONS

The operation of bubble circuits may be effectively analyzed by multidimensional polynomial algebra without actually constructing the circuits. The location of all the data positions in the circuit can be accurately predicted at any preselected instant of time during the operation of a circuit by this technique. When all the circuit parameters are not accurately known, the analysis helps in the calculation of some of the circuit parameters. Further, it helps to algebraically check the validity and effectiveness of a conceived circuit in the performance of specified functions.

The algebra may also be used for circuits that do not perform instantly but need a certain predetermined duration for movement, duplication, gating, etc. Several technologies (magnetic domain, charge coupled and charge transfer technologies) presently being developed in the Bell System fall into this category.

VII. ACKNOWLEDGMENT

The author thanks K. M. Poole for the numerous discussions while the ideas presented in this paper were being developed, and in particular for his suggestion to select one common origin of time, while the author was proposing a multi-origin system. Both systems yield the same results but in a slightly different format.

REFERENCES

1. Bobeck, A. H., "Properties and Device Applications of Magnetic Domains in Orthoferrites," B.S.T.J., *46*, No. 8 (October 1967), pp. 1901–1925.
2. Bobeck, A. H., Fischer, R. F., and Perneski, A. J., "A New Approach to Memory and Logic Cylindrical Domain Wall Devices," Proc. of FJCC, 1959, pp. 489–498.
3. Bonyhard, P. I., Danylchuck, I., Kish, D. E., and Smith, J. L., "Application of Bubble Devices," IEEE Trans. Magnetics, *Mag. 6*, No. 3 (September 1970), pp. 447–451.
4. Perneski, A. J., "Propagation of Cylindrical Magnetic Domains in Orthoferrites," IEEE Trans. Magnetics, *5*, No. 3 (September 1969), pp. 554–557.
5. Peterson, W. W., *Error Correcting Codes*, Cambridge: MIT Press, 1961.
6. Berlekamp, E. R., *Algebraic Coding Theory*, New York: McGraw Hill Book Company, 1968.
7. Ahamed, S. V., "The Design and Embodiment of Magnetic Domain Encoders and Single Error Correcting Decoders for Cyclic Block Codes," B.S.T.J., *51*, No. 2 (February 1972), pp. 461–485.
8. Ahamed, S. V., "Applications of Multidimensional Polynomial Algebra to Bubble Circuits," B.S.T.J., this issue, pp. 1559–1580.
9. Ahamed, S. V., "A General Class of Rate-Change Circuits," B.S.T.J., *50*, No. 10 (December 1971), pp. 3177–3194.

Copyright © 1972 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Applications of Multidimensional Polynomial Algebra to Bubble Circuits

By S. V. AHAMED

(Manuscript received November 17, 1971)

The principles of Multidimensional Polynomial Algebra developed in a companion paper[1] *are applied to two T-bar circuits with bubble-no-bubble coding and one double rail circuit with lateral displacement coding. The object of this paper is to indicate the flexibility of the algebra in its use with real circuits and to emphasize the potential of the algebra as a design tool for bubble circuits.*

I. INTRODUCTION

The operation of bubble circuits depends on the accurate functioning of individual elements such as channeling gates, logic gates, generators, etc., at the critical instants of time. When the circuit becomes complicated, it is not easy to comprehend a multiplicity of functions and determine accurately the instant and duration of operations of these critical elements. Further, a bubble circuit cannot be easily altered like a prototype experimental electrical circuit; and it becomes necessary to ascertain the proper functioning of the designed bubble circuit prior to its actual construction.

In this paper the technique developed in Ref. 1 is applied to three circuits, and the operation of each circuit is predicted. In the second example, the design parameters such as the operating time of the gates, their duration, and the overall timing for the circuit, are derived step by step as the algebra progresses.

II. APPLICATION OF ALGEBRA TO A T-BAR STATION SCAN MEMORY

2.1 *The Principle of Operation*

Consider a hundred stations with 25 lines each. The status of each line is stored in a loop* with 2500 periods as shown in Fig. 1. A controlled

* This configuration of the station scan memory was supplied by A. J. Perneski and R. M. Smith.

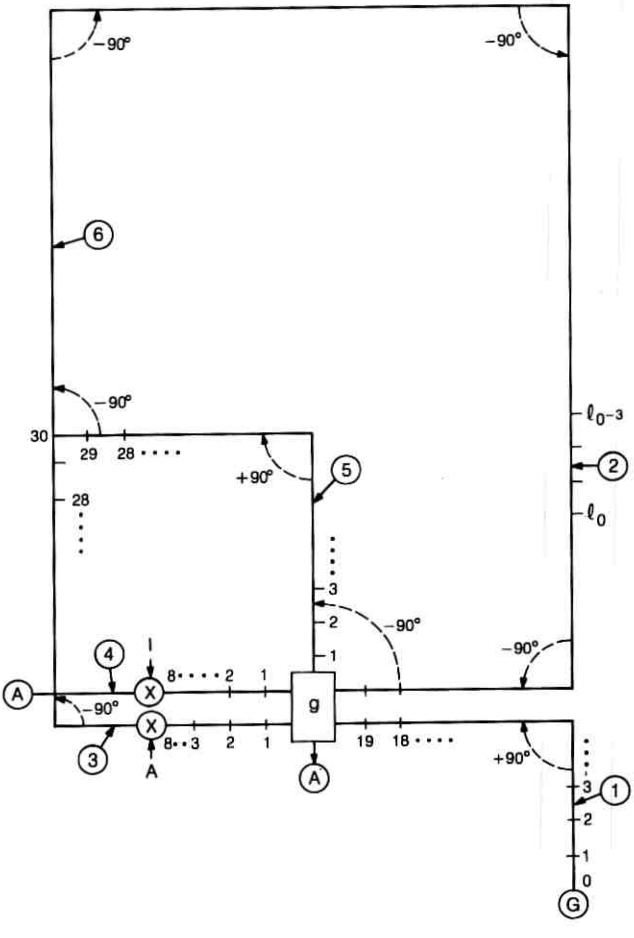


Fig. 1—Block diagram of a station scan memory.

generator G codes the status of each line (active or inactive) by generating a bubble or no bubble during the clock cycles through which the line is being scanned. Thus each line is scanned every 2500 clock cycles. The operation of the gate g is based on the repulsion between bubbles (if any) arriving through elements 1 and 2. One bubble in 1 or 2 passes through the gate to elements 3 or 4 respectively. Two bubbles in 1 and 2 are diverted into the annihilator A and into the element 5. The gate g thus stores the most recent status of each line in the loop through either element 5 or element 3. Two sensors A and I in paths

3 and 4 indicate whether the line has just become active or just become inactive.

The six elements in the circuit are each marked serially so that the bubble streams can be observed in these elements. The gate g does not have an independent status as an element since the streams can only pass through the gate and they cannot be observed within it.

2.2 *The Algebra of the Circuit*

The gate g permits the interaction of the two bubble streams in elements 1 and 2, yielding streams in elements 3, 4 and 5. The truth table for the operation is:

Inputs		Outputs		
1	2	3	4	5
0	0	0	0	0
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

Let the generator G be located 20 periods behind the gate g (including its own location) on element 1. A binary position just generated by G would interact with location (l_0), 20 periods behind the gate on the element 2. Further, let the elements 3 and 5 contain 30 periods each and let the sensors I and A be located 8 periods from g on elements 4 and 3, respectively. These circuit conditions may be algebraically represented as:

g located at Y_1^{20}, $Y_2^{l_0+20}$, Y_3^0 , Y_4^0 and Y_5^0

P located at Y_3^{30} and Y_5^{30}

I and A located at Y_4^8 and Y_5^8 , respectively.

The circuit operates on a repetitive basis, and it is possible to choose an origin of time at the end of the generation cycle of any one bubble position at G. The origin also corresponds to the end of a coding cycle for a particular line L. The algebra may be carried out with any number of bubble positions in the polynomials. If four positions are chosen to illustrate the use of the algebra, then the four bit string after three cycles from this prechosen origin of time may be written as:

$$u_1 = X^3(a_{01}Y_1^3 + a_{11}Y_1^2 + a_{21}Y_1^1 + a_{31}Y_1^0). \qquad (1)$$

At this instant of consideration, the corresponding string of four bubble positions in element 2 may be written as:

$$u_2 = X^3(a_{02}Y_2^{l_0} + a_{12}Y_2^{l_0-1} + a_{22}Y_2^{l_0-2} + a_{32}Y_2^{l_0-3}). \tag{2}$$

It will take 20 clock cycles for the generated binary position a_{31} to be completely processed by the gate g. Thus with $m_1 = 20$, u_1' and u_2' (see footnote in Sec. 4.3.2 of Ref. 1) can be written as:

$$u_1' = X^{23}(a_{01}Y_1^{23} + a_{11}Y_1^{22} + a_{21}Y_1^{21} + a_{31}Y_1^{0}), \tag{3}$$

and

$$u_2' = X^{23}(a_{02}Y_2^{l_0+20} + a_{12}Y_2^{l_0+19} + a_{22}Y_2^{l_0+18} + a_{32}Y_2^{l_0+17}). \tag{4}$$

The last positions a_{31} and a_{32} would be processed by the gate at the end of the 23rd cycle. Now the conversion of u_1' and u_2' to u_3, u_4 and u_5 is feasible by the truth tables for inputs and outputs, and by the spatial conversions:

$$Y_1^{20+z} \rightarrow Y_3^z\,,\; Y_4^z \quad \text{or} \quad Y_5^z$$
$$Y_2^{20+z} \rightarrow Y_3^z\,,\; Y_4^z \quad \text{or} \quad Y_5^z\,.$$

Hence, if a_{01}, a_{11}, a_{21} and a_{31} are 0, 1, 0 and 1; and a_{02}, a_{21}, a_{22} and a_{23} are 1, 1, 1 and 0, then:

$$u_3 = X^{23}(a_{03}Y_3^3 + a_{13}Y_3^2 + a_{23}Y_3^1 + a_{33}Y_3^0) \tag{5a}$$

$$u_4 = X^{23}(a_{04}Y_4^3 + a_{14}Y_4^2 + a_{24}Y_4^1 + a_{34}Y_4^0) \tag{5b}$$

$$u_5 = X^{23}(a_{05}Y_5^3 + a_{15}Y_5^2 + a_{25}Y_5^1 + a_{35}Y_5^0) \tag{5c}$$

where a_{03}, a_{13}, a_{23} and a_{33} are 1, 0, 1 and 0; a_{04}, a_{14}, a_{24} and a_{34} are 0, 0, 0 and 1; and a_{05}, a_{15}, a_{25} and a_{35} are 0, 1, 0 and 0, respectively.

When u_3 and u_4 are multiplied by X^5Y^5, the exponents of Y_3 and Y_5 are both 8. The sensors A and I can read the status during the 28th (i.e., 23 + 5) clock cycle from the prechosen origin of time. In this case, the origin of time corresponds to the start of the coding cycle for the particular line the status of which has just been read, and the 28 cycles indicate the delay between coding the status at G, and reading the change in status at A or I of any particular line L.

Now consider the merger of the streams 3 and 5 at P, i.e., at Y_3^{30} and Y_5^{30}. The merger is complete when the exponent of Y reaches 30 which leads to

$$u_6 = X^{30}Y^{30}\cdot(u_3 + u_5),$$

or

$$u_6 = X^{53}(a_{06}Y_6^3 + a_{16}Y_6^2 + a_{26}Y_6^1 + a_{26}Y_6^0), \tag{6}$$

with a_{06}, a_{16}, a_{26} and a_{36} being 0, 1, 0 and 1. The position a_{06} reaches the merger point Y_6^0 during the 50th cycle, a_{16} during the 51st cycle and so on.

If the Y_6^3 is 2447 (i.e., 2500–53) periods* from $Y_2^{l_0}$, then

$$u_2 = X^{2447}Y^{2447}\cdot u_6 ,$$

thus leading to the polynomial u_2

$$u_2 = X^{2500}(a_{02}Y_2^{l_0} + a_{12}Y_2^{l_0-1} + a_{22}Y_2^{l_0-2} + a_{32}Y_2^{l_0-3}). \tag{7}$$

The origin of time was chosen at the end of a coding cycle of a particular line. At the end of 2500 clock cycles, the cyclic process is repeated and the next set of calculations may be started at this instant.

2.1.1 *Effect of Corners in the Circuit*

Corners were ignored in the polynomial calculation in the previous section. Considering their effects, we have

$$u_1' = X^{20}Y^{20+\frac{1}{4}}u_1 \; ; \quad \text{and,} \quad u_2' = X^{20}Y^{20-\frac{1}{4}}u_2 \qquad \text{(8a; 8b)}$$

from Sec. 4.3.4 in Ref. 1. However, at the gate g, the two inputs u_1 and u_2 should be in phase. This condition implies that the generator G should generate the position a_{14} half a clock cycle after the instant as assumed in the previous calculation, leading to

$$u_3 = X^{-\frac{1}{2}}Y^{-\frac{1}{2}}\cdot u_1' = X^{22\frac{1}{2}} \sum_{i=0}^{i=3} a_{i3}Y_3^{(2\frac{3}{4}-i)} \tag{9}$$

$$u_4 = X^{22\frac{1}{2}} \sum_{i=0}^{i=3} a_{i4}Y_4^{(2\frac{3}{4}-i)}. \tag{10}$$

The element u_5 has a -90 degree corner at the gate, and hence,

$$u_5 = X^{22\frac{1}{2}} \sum_{i=0}^{i=3} a_{i5}Y_5^{(2\frac{1}{2}-i)}. \tag{11}$$

The constants a_{ij} ($i = 0$ through 3, $j = 3$ through 5) are the same as their previous values. The sensors A and I should be read during the cycle ending at $22\frac{1}{2} + (8 - 2\frac{3}{4}) = 27\frac{3}{4}$ clock cycles, after the generation of the first data position a_{01}.

The polynomial u_3 makes a -90 degree corner, and u_5 makes compensating $+90$ and -90 degree turns, before reaching P, yielding

$$u_3' = X^{30}Y^{29\frac{3}{4}}u_3 \; ; \quad \text{and,} \quad u_5' = X^{30}Y^{30}\cdot u_5 \qquad \text{(9a; 11a)}$$

* It can be seen that the top section of the storage loop need not have two independent element numbers 2 and 6; but such a numbering facilitates the representation and its boundary may be considered to lie at any period z located at $Y_6{}^z$ and $Y_2{}^{l_0}{}^{2450+z}$.

leading to

$$u_6 = X^{52\frac{1}{2}} \sum_{i=0}^{i=3} a_{i6} Y_6^{(2\frac{1}{2}-i)}. \tag{12}$$

If incremental space positions are considered, u_6 after half a clock cycle is

$$u_6 = X^{53} \sum_{i=0}^{i=3} a_{i6} Y_6^{3-i}. \tag{13}$$

The polynomial u_6 has moved around two -90 degree corners before it reaches l_0 . Hence

$$\begin{aligned} u_6' &= X^{2447} Y^{2446\frac{1}{2}} \cdot u_6 = X^{2500} \sum_{i=0}^{i=3} a_{i6} Y^{2449\frac{1}{2}-i} \\ &= X^{2500} \sum_{i=0}^{i=3} a_{i2} Y_2^{l_0-\frac{1}{2}-i}. \end{aligned} \tag{14}$$

An additional half cycle is now necessary to obtain incremental space positions, which results in

$$u_2 = X^{\frac{1}{2}} Y^{\frac{1}{2}} u_6' = X^{2500\frac{1}{2}} \sum_{i=0}^{i=3} a_{i2} Y_1^{l_0-i} \tag{15}$$

indicating that the bubble position a_{01} generated by G is at l_0 , $2500\frac{1}{2}$ clock cycles from the prechosen origin of time. In this case it is at l_0 $2500\frac{1}{2}$ cycles after its generation (having lost $\frac{1}{2}$ cycle at the two -90 degree corners of the loop), and it is going in a direction opposite to the direction at the prechosen origin of time, i.e., at the instant of its generation.

III. APPLICATION OF THE ALGEBRA TO (7, 4) HAMMING CODE, MAGNETIC DOMAIN ENCODER

3.1 *Principle of Operation*

A configuration of the encoder[2] is shown in Fig. 2. The four incoming data bits of every code word are generated by G uniformly during 28 t seconds, where t denotes the interval for one rotation of the main magnetic field (see Ref. 2). The data bits are accumulated in adjoining T-bar periods in loop 1 and gated out by g_1 to a duplicator D. One of the two resulting streams is allowed to circulate in loop 3 and the other is divided[3,4] by the generator function

$$g(X) = X^3 + X^2 + 1.$$

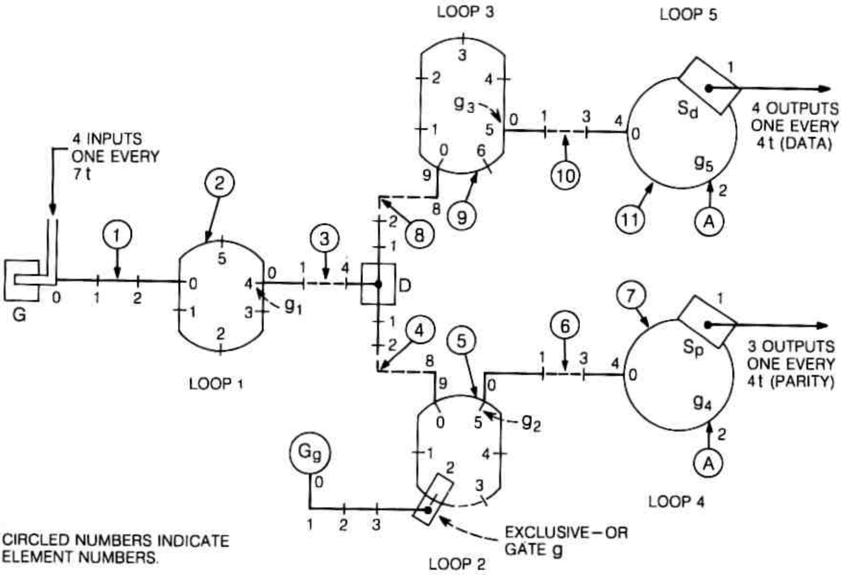

LOOP 3
LOOP 5
4 INPUTS ONE EVERY 7t
4 OUTPUTS ONE EVERY 4t (DATA)
3 OUTPUTS ONE EVERY 4t (PARITY)
G
D
Sd
Sp
g1
g2
g3
g4
g5
Gg
A
LOOP 1
LOOP 2
LOOP 4
EXCLUSIVE-OR GATE g
CIRCLED NUMBERS INDICATE ELEMENT NUMBERS.

Fig. 2—(7, 4) Hamming encoder with magnetic domains and T-bars.

There are four steps in the division cycle, each step being accomplished as the data stream passes through the gate g. After 4 steps and 3 circulations of the stream in loop 2 (6 periods), the remainder (the three parity bits) are gated by g_2 from loop 2 to loop 4. Meanwhile, the data in the loop 3 (7 periods), also having completed three circulations, is gated by g_3 to loop 5. Loops 4 and 5 have 3 periods each, and the data or parity bit is read by S_d or S_p as the leading bit of the circulating stream in loops 5 or 4 respectively. The gates g_5 and g_4 operate identically by diverting the bit just read by S_d or S_p into the annihilator A. Therefore, the outgoing information (7 bits) is read every $4t$ while the received information (4 bits) is generated every $7t$.

3.2 *The Algebra of the Circuit*

It is necessary to choose an origin of time and proceed in the time domain step by step as the algebra progresses. The exponent of X indicates the number of clock cycles between a prechosen origin of time and the instant of consideration. The binary values of the bit positions are indicated by the values of a, and the location in the circuit is indicated by the subscript of Y (the element number) and the exponent of Y (the particular period in that element).

This circuit operates on a repetitive basis. The information is being continuously received at G, and the coded words are being continuously read at S_d and S_p. The origin of time can thus be chosen at the end of a cycle during which the first binary position of any particular data block is being generated by G.

To facilitate the representation, it is advantageous to divide the algebra of the circuit into a series of operations* corresponding to the subfunctions in the circuit.

If the four positions of the data string generated are represented† and considered 21 clock cycles from the prechosen origin of time,

$$\begin{aligned} u_1' &= X^{21}(a_{01}Y_1^{21} + a_{11}Y_1^{14} + a_{21}Y_1^{7} + a_{31}Y_1^{0}) \\ &= X^{21}\sum_{i=0}^{i=3} a_{i1}Y_1^{7(3-i)}. \end{aligned} \tag{16}$$

Operation 1: Transportation of u_1' from generator (element 1) to loop 1 (element 2): With their boundary located at Y_1^3 and Y_2^0, we have (from Sec. 4.3.2 of Ref. 1)

$$u_1'' = X^3Y^3 \cdot u_1' = X^{24}\sum_{i=0}^{i=3} a_{i2}Y_1^{7(3-i)+3} \tag{16a}$$

$$u_2' = X_{24}\sum_{i=0}^{i=3} a_{i2}Y_2^{7(3-i)} \tag{17}$$

since $Y_1^{3+\alpha} = Y_2^{\alpha}$.

Operation 2: Looping of u_2' in element 2 with six periods in the loop:

$$u_2 = X^{24}\sum_{i=0}^{i=3} a_{i2}Y_2^{7(3-i) \bmod 6} = X^{24}\sum_{i=0}^{i=3} a_{i2}Y_2^{(3-i)}. \tag{18}$$

Operation 3: Gating the stream u_2 out of loop (element 2) to the path (element 3) between loop and duplicator D: If the gate g_1 is at Y_2^4, then

$$u_3 = X^4Y^4 \cdot u_2 = X^{28}\sum_{i=0}^{i=3} a_{i3}Y_3^{(3-i)}, \tag{19}$$

since $Y_2^{4+\alpha} = Y_3^{\alpha}$ when $\alpha \geqq 0$ while the gate g_1 is operational.

DESIGN PARAMETER 1: *Operating the gate* g_1 : This gate should be operational for the 25th, 26th, 27th and 28th cycles from the origin of time.

[*] This type of distinct numbering of operations is suggested when the circuit accomplishes complex functions.

[†] The prime (s) indicates that the polynomial as such does not represent a bubble stream. But after certain ensuing algebraic operations they will represent observable streams.

Operation 4: Transportation of u_3 from gate to duplicator and its duplication: Let the duplicator D be located at Y_3^4. Then

$$u_4 = X^{32} \sum_{i=0}^{i=3} a_{i4} Y_4^{(3-i)}; \qquad u_8 = X^{32} \sum_{i=0}^{i=3} a_{i8} Y_8^{(3-i)}. \tag{20; 21}$$

Operation 5: Transportation of u_4 and u_8 to loops 2 and 3 respectively: If the encoder design has 9 periods in the paths between D and loops 2 and 3 then

$$u_5 = X^9 Y^9 \cdot u_4 = X^{41} \sum_{i=0}^{i=3} a_{i5} Y_5^{(3-i)}, \tag{22}$$

and

$$u_9 = X^9 Y^9 \cdot u_8 = X^{41} \sum_{i=0}^{i=3} a_{i9} Y_9^{(3-i)}. \tag{23}$$

The binary positions a_{ij} ($i = 0$ through 3 and $j = 1, 2, 3, 4, 5, 8$ and 9) have not undergone any transition points in the circuit. However, in the polynomial u_5, the binary values of bubble positions a_{i5} will undergo definite changes as u_5 is divided by the generator function. The effect of these changes may be represented in the algebra by discriminate use of superscripts for a_{i5} which leads to

$$u_5 = X^{41}(a_{05}^0 Y_5^3 + a_{15}^0 Y_5^2 + a_{25}^0 Y_5^1 + a_{35}^0 Y_5^0). \tag{22}$$

Operation 6: Generation of the divisor polynomial u_{g0} (g0 *for the first time) by the generator* G_g : It is seen that the instant of start of the generating cycle for the first bit position is not known from the prechosen origin of time. For this reason we may assume that this interval of time is g0 and determine its value as the algebra progresses.

Four binary positions* are generated by G_g . Three cycles after the generation of the first position the bubble stream may be written as:

$$u_{g0} = X^{g0+3}(a_{0_g} Y_g^3 + a_{1_g} Y_g^2 + a_{2_g} Y_g^1 + a_{3_g} Y_g^0). \tag{24}$$

Operation 7: Transportation of u_{g0} to gate g: If the designer has allocated 4 periods between the generator G_g and the gate g, then† (Sec. 4.3.2 of Ref. 1)

$$u'_{g0} = X^{g0+7} \sum_{i=0}^{i=3} a_{ig} Y_g^{7-i}. \tag{24a}$$

* See Ref. 2 for the details of magnetic domain encoding and decoding.

† The prime indicates that the polynomial u_g in this equation has already undergone the first step of the division cycle, and the binary values are no longer the same as in the previous polynomial.

Operation 8: Transportation of u_5 to gate g: If the designer has permitted 2 periods between the entry of the loop 2 and the gate g then

$$u_5' = X^2Y^2 \cdot u_5 = X^{43} \sum_{i=0}^{i=3} a_{i5}^0 Y_5^{(5-i)}. \tag{25}$$

DESIGN PARAMETER 2: *The instant of generation of* u_{g0} : If the gate g is located at Y_5^2 and Y_g^4 , and if a_{05} in (25) interacts with a_{0g} in (24a), then they should pass through g during the same clock cycle, and the exponents of X associated with a_{05} in (25) and a_{0g} in (24a) when the corresponding exponents of Y_5 and Y_g are 2 and 4, may be equated yielding $43 - 3 = \text{g}0 + 7 - 3$ or

$$\text{g}0 = 36 \text{ clock cycles}. \tag{26}$$

The implication of this equation is that the generator G_g must generate u_{g0} (if a_{05} is one, Ref. 2) with its bubble position a_{0g} , exactly during the 36th clock cycle from the prechosen origin of time. This is depicted in Fig. 3a.

Operation 9: Exclusive-or operation of u_5 in (22) and u_{g0} in *(24):* At this stage of the calculation it is necessary to assign binary values for the a_{05}^0 , a_{15}^0 , a_{25}^0 and a_{35}^0 (which are the same as data bits a_{01} , a_{11} , a_{12} and a_{13} respectively) and let these be chosen as 1111 respectively. The values of a_{0g} , a_{1g} , a_{2g} and a_{3g} are 1101 respectively, if the generator function (see Ref. 3 or 4) of the code has a form as defined in Section 3.1.

Now u_5 may be written as*

$$u_{51} = X^{43} \sum_{i=0}^{i=3} a_{i5}^1 Y_5^{(5-i)}, \tag{27}$$

where $a_{i5}^1 = a_{i5}^0 \oplus a_{ig}$ thus yielding $a_{05}^1 = 0$, $a_{15}^1 = 0$, $a_{25}^1 = 1$ and $a_{35}^1 = 0$, since a_{0g} , a_{1g} , a_{2g} and a_{3g} correspond to 1101.

It can be seen that a_{05}^1 is always zero and it can be dropped from the notation, since it is never again activated in the circuit. This leads to

$$u_{51} = X^{43} \sum_{i=1}^{i=3} a_{i5}^1 Y_5^{(5-i)}. \tag{27}$$

Operation 10: Generation of the generator polynomial u_{g1} (g1 *for the second time*) *by the generator* G_g : Four binary positions are generated. Three cycles after the generation of the first position the binary stream

* The second subscript 1, of u, indicates that it has gone through the exclusive-OR operation once.

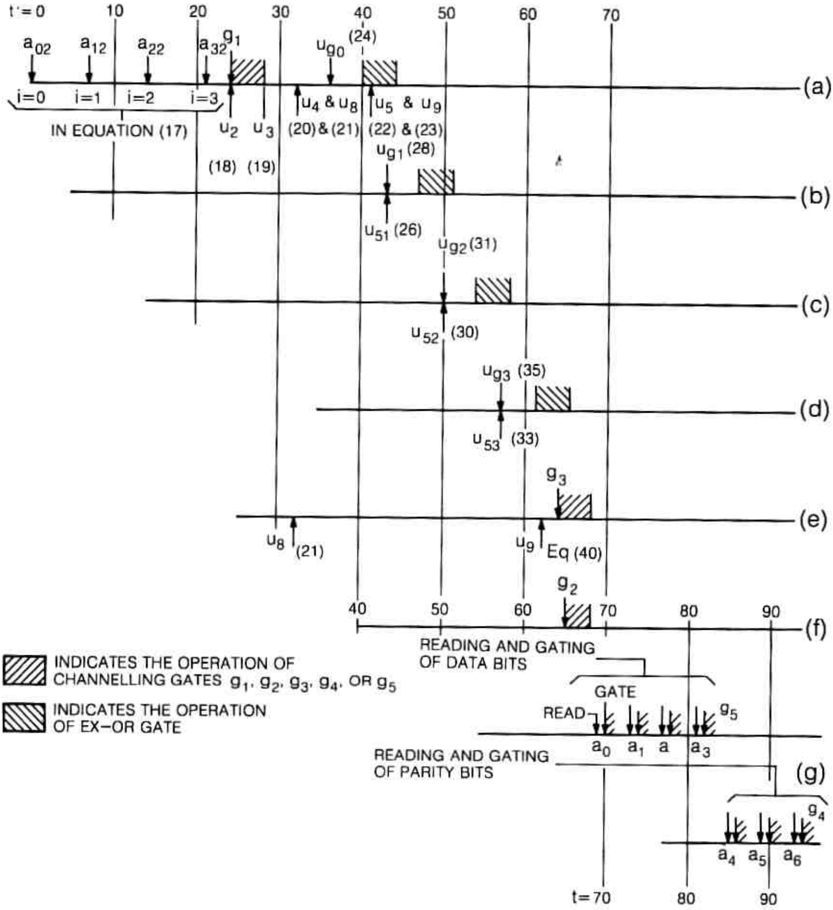


Fig. 3—The timing diagram for the various functions in the (7, 4) Hamming encoder. (a) The generation of data bits a_{01}, a_{11}, a_{21}, a_{31} and subsequent looping, gating, duplicating functions. u_{g0} indicates the generation of the general polynomial by G_g. (b) The generation of u_{g1} by G_g. (c) The generation of u_{g2} by G_g. (d) The generation of u_{g3} by G_g. (e) The polynomial u_9 and its gating to element 10. (f) The parity bits [polynomial u_{54}, eq. (37)] and its gating. (g) The encoded data a_0 through a_6 and its reading by S_d and S_p.

may be represented as

$$u_{g1} = X^{g1+3} \sum_{i=0}^{i=3} a_{ig} Y_g^{(3-i)}, \tag{28}$$

where g1 is another design parameter to be determined as the algebra progresses.

Operation 11: Transportation of u_{g1} *to gate* g:

$$u'_{g1} = X^{g1+7} \sum_{i=0}^{i=4} a_{ig} Y_g^{(7-i)}. \tag{28a}$$

Operation 12: Looping of u_{51} *(26) once in loop 2:* Loop 2 has six periods and after 7 clock cycles we have from Sec. 4.3.3 of Ref. 1

$$u'_{51} = X^7 Y^{7 \bmod 6} \cdot u_5 = X^7 Y \cdot u_{51} = X^{50} \sum_{i=1}^{i=3} a_{15}^1 Y_5^{(6-i)}. \tag{29}$$

DESIGN PARAMETER 3: *The instant of generation of* u_{g1} : If the encoder is to function properly (see Ref. 2) then a_{0g} in (28a) should interact with a_{15}^1 . A calculation similar to that in operation 8 yields g1 = 43. It is to be noted that G_g generates its first binary position a_{0g} in (28a), during the 43rd clock cycle from the prechosen origin of time, which corresponds to the generation of a_{01} by G. (See Fig. 3b.)

Operation 13: Exclusive-or function between u'_{51} *and* u_g : For the correct functioning of the encoder, the generator G_g generates the sequence of four bubble positions 1101, only if the leading bubble position in polynomial u'_{51} is one. In this case it can be seen that a_{15}^1 is zero, so the generator G_g generates a sequence of 4 bubble positions whose binary values are zero thus leading to

$$u_{52} = X^{50} \sum_{i=1}^{i=3} a_{i5}^2 Y^{(6-i)} + a_{45} X^{50} Y_5^2 , \tag{30}$$

where $a_{i5}^2 = a_{i5}^1 \oplus a_{(i-1)g}$ thereby yielding $a_{15}^2 = 0$, $a_{25}^2 = 1$, $a_{35}^2 = 0$, $a_{45}^2 = 0$ since a_{0g} , a_{1g} , a_{2g} and a_{3g} are 1101. Once again, a_{15}^2 being always zero, can be dropped from the equation leading to

$$u_{52} = X^{50} \sum_{i=2}^{i=4} a_{i5}^2 Y_5^{(6-i)}. \tag{30a}$$

The last term in (30) and (30a) is the binary position a_{3g} of (28a) with the exponent of X being 50 since g1 was calculated as 43 clock cycles.

Operation 14: Generation of generator polynomial u_{g2} (g2 *for the third time*) *by the generator* G_g : Four binary positions are generated. Three *cycles:* after the generation of the first binary position, the binary position is written as

$$u_{g2} = X^{g2+3} \sum_{i=0}^{i=3} a_{ig} Y_g^{(3-i)}, \tag{31}$$

where g2 will be evaluated as a design parameter.

Operation 15: Transportation of u_{g2} to gate:

$$u'_{g2} = X^{g2+7} \sum_{i=0}^{i=3} a_{ig} Y_g^{(7-i)}. \tag{31a}$$

Operation 16: Looping u_{52} in (30) in loop 2 once for 7 clock cycles:

$$u'_{52} = X^7 Y \cdot u_{52} = X^{57} \sum_{i=2}^{i=4} a^2_{i5} Y_5^{(7-i)}. \tag{32}$$

DESIGN PARAMETER 4: *The instant of generation of u_{g2}* : Equating the exponents of X associated with the interacting terms a_{0g} in (31a) and a_{25} in (32) when they pass through the gate g we have g2 = 50 clock cycles.

Operation 17: Exclusive-or function:

$$u_{53} = X^{57} \sum_{i=3}^{i=5} a^3_{i5} Y_5^{7-i}, \tag{33}$$

where a^3_{35} , a^3_{45} , a^3_{55} may be evaluated as 101 respectively, since a_{1g} , a_{2g} and a_{3g} are 101.

Operation 18: Generation of u_{g3} : Three cycles after the generation of a_{0g} we have

$$u_{g3} = X^{g3+3} \sum_{i=0}^{i=3} a_{ig} Y_g^{3-i}. \tag{34}$$

Operation 19: Transportation of u_{g3} :

$$u'_{g3} = X^{g3+7} \sum_{i=0}^{i=3} a_{ig} Y_g^{7-i}. \tag{35}$$

Operation 20: Looping of u_{53} in loop 2

$$u'_{53} = X^{64} \sum_{i=3}^{i=5} a^3_{i5} Y_5^{(8-i)}. \tag{36}$$

DESIGN PARAMETER 5: *The instant of generation of u_{g3}* : The value of g3 can be calculated as 57 clock cycles (by equating the exponents of X in (35) and (36); see also Fig. 3d).

Operation 21: Exclusive-or function between u'_{53} and u'_{g3} :

$$u_{54} = X^{64} \sum_{i=4}^{i=6} a^4_{i5} Y_5^{(8-i)}, \tag{37}$$

where a^4_{45} , a^4_{55} , a^4_{65} correspond to 111 respectively (since a_{1g} , a_{2g} and a_{3g} are 101, thus leading to the parity bits for the (7, 4) Hamming code with the four data bits as 111 and 1.

Operation 22: Gating of parity bits from loop 2: Now the parity bits at Y_5^4, Y_5^3 and Y_5^2 can be channeled out of the loop 2 into the element 6 by the action of the gate g_2 located at Y_5^5. This function may be represented as

$$u_5' = X^3Y^3\cdot u_{54} = X^{67}\sum_{i=4}^{i=6} a_{i5}Y_5^{11-i}$$

$$= X^{67}(a_{45}Y_5^7 + a_{55}Y_5^6 + a_{65}Y_5^5), \tag{38}$$

since $Y_5^{5+\alpha} = Y_6^\alpha$ when $\alpha > 0$ thereby leading to

$$u_6 = X^{67}(a_{46}Y_6^2 + a_{56}Y_6^1 + a_{66}Y_6^0). \tag{39}$$

DESIGN PARAMETER 6: *The operation of the gate* g_2 : The gate is located at Y_5^5 and Y_6^0 and it can be seen that a_{46} , a_{56} and a_{66} reach Y_6^0 when the exponent of X is 65, 66 and 67, indicating that the gate must operate during these three cycles to channel the bits for loop 2.

*Operation 23: Looping of u_9 three times:** The data stream u_9 has been circulating in loop 3 for (3 × 7) clock cycles leading to

$$u_9 = X^{41+21}\sum_{i=0}^{i=3} a_{i9}Y_9^{(24-i)\bmod 7} = X^{62}\sum_{i=0}^{i=3} a_{i9}Y_9^{3-i}. \tag{40}$$

Operation 24: Transportation of u_9 to gate g_3 : This gate is located at Y_9^5 to match the location of the gate g_2 at Y_5^5 (see operation 22). If the gate operates at the appropriate time for four clock cycles, then the polynomial representing the stream in the path between loop 3 and loop 5 is

$$u_{10}' = X^5Y^5\cdot u_9 = X^{67}\sum_{i=0}^{i=3} a_iY_9^{(8-i)}. \tag{41}$$

But $Y_9^{5+\alpha} = Y_{10}^\alpha$ during gating of g_3 and therefore we have

$$u_{10} = X^{67}\sum_{i=0}^{i=3} a_iY_{10}^{(3-i)}.$$

DESIGN PARAMETER 7: *The operation of gate* g_3 : The polynomial u_{10} in (41) indicates that the gate g_3 should operate when the bubble positions a_0 , a_1 , a_2 , and a_3 are at Y_{10}^0 . Further, when the exponents of X are exactly 64, 65, 66 and 67, the four binary positions a_0 , a_1 , a_2 and a_3 are in the gate g_3 . Hence, the operation of this gate must coincide with the 64th, 65th, 66th and 67th clock cycles from the prechosen origin of time.

* This operation takes place during the 26 clock cycles allocated for operations 6 through 22 for the polynomials u_5, u_{51}, u_{52}, u_{53} and u_6.

The single clock cycle between the operation of gates g_3 and g_2 (Fig. 3e and f) is due to the incomplete fourth rotation of the parity bits u_6. These are gated out by g_3 after 3 rotations and after 4 steps of the division. (See Ref. 2.)

Operation 25: Transportation of data and parity bits to loops 5 and 4: If there are 4 periods in elements 10 and 6, then

$$u'_{11} = X^4Y^4\cdot u_{10} = X^{71}\sum_{i=0}^{i=3} a_i Y_{11}^{3-i}, \tag{42}$$

and

$$u_7^* = X^4Y^4\cdot u_6 = X^{71}\sum_{i=4}^{i=6} a_i Y_7^{(3-i)}. \tag{43}$$

DESIGN PARAMETER 8: *Reading of data bit a_0 and its gating by g_5 into the annihilator* A: If the sensor S_d is located at Y_{11}^1, then a_0 will be at Y_{11}^1 at X^{69}, indicating that the sensor should be read during the 69th clock cycle from the prechosen origin of time. If the gate g_5 is located at Y_{11}^2, then a_0 is at Y_{11}^2 at X^{70} and it should operate during the 70th cycle. After the gating of a_0 by g_5, we have

$$u_{11} = X^{71}\sum_{i=1}^{i=3} a_i Y_{11}^{(3-i)}. \tag{44}$$

Operation 26: Looping for 2 clock cycles: After 2 clock cycles, a_1 will be at Y_{11}^1, and it can be read again, since

$$u_{11} = X^{73}\sum_{i=1}^{i=3} a_i Y_{11}^{(5-i)\bmod 3}. \tag{45}$$

DESIGN PARAMETERS 9, 10, 11: *Reading and gating of a_1, a_2 and a_3 by S_d and g_5 respectively:* The sensor S_d is read during the 73rd clock cycle and gate g_5 operates during the 74th clock cycle from the origin of time. More operations of the type 27 indicate that S_d should read a_2 during the 77th clock cycle, and g_5 should gate a_2 during the 78th clock cycle, S_d should read a_3 during the 81st clock cycle, and g_4 should gate a_3 during the 82nd clock cycle. These details are plotted in Fig. 3g.

Operation 27: Looping[†] *of parity bits in loop 4:* After 10 clock cycles the parity bits in loop 4 may be represented as

[*] It is no longer necessary to carry a second subscript for a, since none of the binary values in any of the polynomials change in the remainder of the circuit.

[†] This operation takes place during the 10 (i.e., -2 for a_0, and 4 for a_1, a_2 and a_3 each) clock cycles allocated for reading, gating and looping of the four data bits a_0, a_1, a_2 and a_3.

$$\begin{aligned} u_7 &= X^{71+10}(a_4 Y_7^{12 \bmod 3} + a_5 Y_7^{11 \bmod 3} + a_6 Y_7^{10 \bmod 3}) \\ &= X^{81}(a_4 Y_7^0 + a_5 Y_7^2 + a_6 Y_7^1). \end{aligned} \tag{46}$$

Operation 28: Looping of u_7 : After 4 clock cycles*

$$u_7 = X^{85}(a_4 Y_7^1 + a_5 Y_7^0 + a_6 Y_7^2). \tag{47}$$

DESIGN PARAMETERS 12, 13 AND 14: *Reading and gating of a_4 , a_5 and a_6 :* It is seen that a_4 is at Y_7^1 (location of S_p) during the 85th clock cycle. The gate g_4 at Y_7^2 should divert a_4 during the 86th clock cycle. More operations of type 28 indicate that S_p should read a_5 during the 89th clock cycle, and g_4 should gate a_5 during the 90th clock cycle, S_p should read a_6 during 93rd clock cycle, and g_4 should gate a_6 during the 94th clock cycle. These details are also plotted in Fig. 3g.

A block of data is thus completely processed by the circuit after the 94th clock cycle, and the position of every bit of information may be accurately predicted during any prechosen clock cycle. The design parameters are also accurately determined by the analysis.

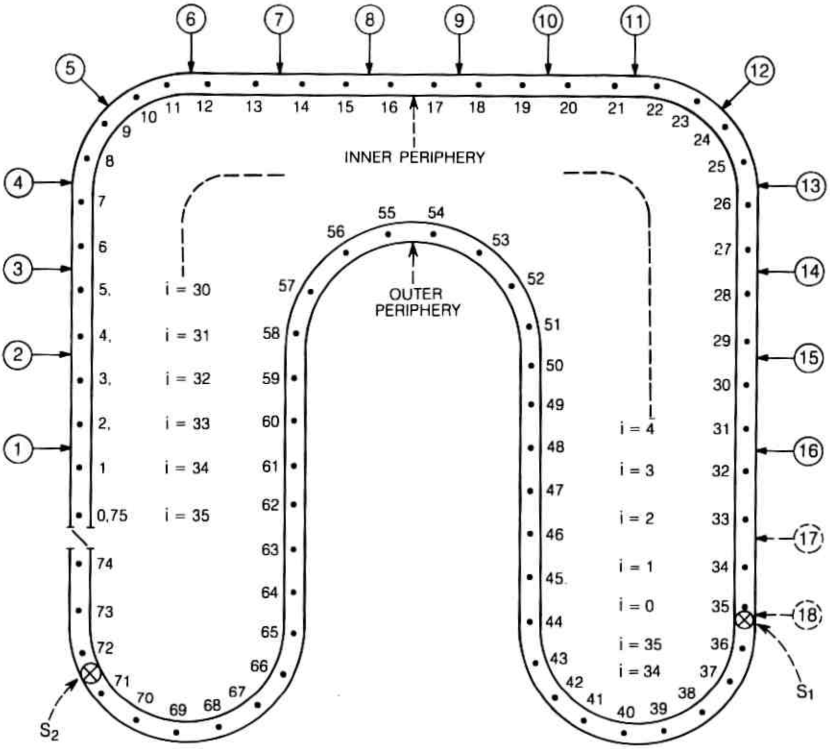
VI. APPLICATION OF THE ALGEBRA TO 16 LINE LDC MAGNETIC DOMAIN LINE SCANNER

4.1 *Principle of Operation*

The circuit for the line scanner is shown in Fig. 4. Sixteen inputs, 1 through 16, carry telephone line currents in a loop for each line. When the line current is not sensed (i.e., on-hook status) during a scan interval, a bubble in the position (1–2), (3–4), ··· (31–32) of a 75-position loop is moved from the outer periphery to the inner periphery of the loop, thus effecting a *L*ateral *D*isplacement *C*oding.

The laterally displaced bubble positions are moved under two sensors S_1 and S_2 . The spacing between these is arranged to sense the status of a particular line at two instants of time. When the sensor S_2 is sensing the contents of a certain cell coded by the circuit in line j, ($j = 1$ through 16) at an instant t, then the sensor S_1 is sensing the contents of the cell Y coded by the current in the same line j at $t + \delta$. (The value of δ is the duration required to move the bubbles from sensor S_1 to S_2 and is also the scanning interval of the lines.) When there is no discrepancy between the readings of S_1 and S_2 , then there is no change in status of the line (on hook or off hook) during the scanning interval and vice versa. The status of the jth line is coded during the movement of the

* It is interesting to note that the extra clock cycle between the gating of g_3 and g_2 (design parameter 7) is really necessary for correct functioning at this stage.



i DENOTES THE BUBBLE POSITION IN STREAM.
0-74 DENOTES THE LOCATION NUMBERS.

S_1 AND S_2 ARE SENSORS TO DETECT ON-HOOK STATUS OF CODED BUBBLE POSITIONS.

Fig. 4—The schematic diagram of 16 Telephone-Line-Scanner.

bubble across the cell from a position $(2j - 1)$ to $2j$. A conductor carrying a single phase current around the loop propagates the bubble positions around its periphery at a uniform speed. Further, when the bubble positions pass through the cell 74–0, they are all positioned towards the outer periphery of the loop, and are reset ready to be coded again.

4.2 *Algebra of the Circuit*

The origin of time may be chosen at the start of a coding cycle which repeats every 36 clock cycles. The lateral displacement coding occurs during the single clock cycle immediately after all the cells 0 through 35 contain bubbles at the outer periphery of the loop. The start of this cycle would then constitute the origin of time, and the polynomial representing the stream in this section of the loop is*

$$u_{00} = X^0 \sum_{i=0}^{i=35} a_i Y^{35-i}, \tag{48}$$

where $i = 0$ denotes the leading bubble position in the stream; and a_i is the uncoded status of the bubble position. Next consider the bubble stream in the section 36 to 71 of the loop. This stream carries the status of each line from the last cycle, and the polynomial representing this stream at the prechosen origin of time is

$$u_{10} = X^0 \sum_{i=0}^{i=35} a_i Y^{71-i}, \tag{49}$$

where i indicates the bubble position in the stream. Similarly, the polynomial u_2 between 72 and 74 may be written as:

$$u_{20} = X^0 \sum_{i=33}^{i=35} a_i Y^{107-i}. \tag{50}$$

The value of i ranges from 33 to 35 since there are only three locations 72, 73, and 74 to accommodate the last three bubble positions of the stream coded during their own coding cycle. The contents of the entire loop at the origin of time may be written as:

$$\begin{aligned} u_0 &= u_{00} + u_{10} + u_{20} \\ &= X^0\left(\sum_{i=0}^{i=35} a_i Y^{35-i} + \sum_{i=0}^{i=35} a_i Y^{71-i} + \sum_{i=33}^{i=35} a_i Y^{107-i}\right). \end{aligned} \tag{51}$$

Now consider the movement of the bubble stream represented by (51); for m ($m \leqq 36$) clock cycles, the resulting polynomial according to Sec. 4.3.2 of Ref. 1 is:

$$\begin{aligned} u_1 = X^m\Bigg(&\sum_{i=0}^{i=m-1} a_i Y^{m-1-i} + \sum_{i=0}^{i=35} a_i Y^{35+m-i} \\ &+ \sum_{\substack{i=0 \\ \text{or} \\ i=m-3}}^{i=35} a_i Y^{71+m-i} + \sum_{i=33+m}^{i=35} a_i Y^{107-i}\Bigg) \\ = u_{01} &+ u_{11} + u_{21} + u_{31}\,, \end{aligned} \tag{52}$$

where u_{01} represents the first m binary positions of u_{00} being generated in the section 0 through 35 of the loop; u_{11} results from the translatory

* There is no need to subscript Y since there is only one element (the loop) in the circuit.

movement (Sec. 4.3.2 of Ref. 1) of u_{00} in (51), and u_{21} results from the movement of u_{10} in (51). The lower limit of i in u_{21} should be chosen according to the value of m. When m is less than 3, the value of i equals 0 is appropriate, since the leading bubble position a_0 is within the maximum number of binary bubble positions (i.e., 74) in the loop. When m exceeds three, the leading bubble a_0 of u_{10} in (51) is transformed as the fourth term of u_{01} in (52). The polynomial u_{31} is the transformation of u_{20} when m is less than 3. When m exceeds two, u_{31} drops out of (52), since the lower limit of i exceeds the upper limit of 35.

Examine the polynomial u_1 in (52), when $m = 0$, u_{01} drops out of the equation, and u_{11} , u_{21} and u_{31} will assume the roles of the polynomials u_{00} , u_{10} and u_{20} in (51). Next observe the polynomial u_1 in (52); when m reaches a value of 36, u_{01} , u_{11} and u_{21} will assume the roles of u_{00} , u_{10} and u_{20} in (51), and u_{31} drops out of the equation. In essence, we have two cyclic processes taking place simultaneously, the first one being in the time dimension and repeating every 36 clock cycles, the second one being in the spatial dimension and repeating every 75 periods. The effect of the first cyclic process may be eliminated by always considering m as (m mod 36). The effect of the second one may be eliminated by always considering the exponent (e) for Y as (e mod 75).

Table I relates the values of the exponents of Y for various values of i and m in the polynomials u_{01} , u_{11} , u_{21} , and u_{31} . It also indicates the locations of the first and last bubble positions of streams represented by these polynomials.

4.4 *Implication and Use of the Representation for the Line Scanner*

4.4.1 *Prediction of Bubble Positions*

Consider the tenth ($i = 10$) bubble position in a data stream coded for twenty ($m_1 = 20$) cycles and propagated for ninety ($m = 90$) cycles.

The initial position under consideration is $a_{10}X^{20}Y^{l}$. Table I indicates that this term exists in u_{11} with a value of $l = 45$ yielding the location of this position. When this location is propagated for 90 cycles, the new position is $Y^{(45+90) \bmod 75}$, i.e., Y^{60}; and the corresponding value of m* is $(20 + 90) \bmod 36$, i.e., 2 cycles. The bubble position is then $a_iX^2Y^{60}$. The only positive value of i which satisfies the constraints on the exponents of both X and Y is 13; and the individual term denoting this bubble position lies in the polynomial u_{21} in (52). This implies that if the original position in u_{11} is $a_{10}X^{20}Y^{45}$, then after 90

* When the exponent of Y is less than $(m - 1)$, it should be concluded that the term is in u_{01}, (see Table I) and is in the dead interval of the circuit.

TABLE I—LOCATION OF BUBBLE STREAMS

Limits of the exponents of Y in the polynomials u_{01}, u_{11}, u_{21}, and u_{31} in (52)

m or m mod 36	$u_{01} = X^m \sum_0^{m-1} a_i Y^{m-1-i}$	$u_{11} = X^m \sum_0^{35} a_i Y^{35+m-i}$	$u_{21} = X^m \sum_{0 \text{ or } m-3}^{35} a_i Y^{71+m-i}$	$u_{31} = X^m \sum_{33+m}^{35} a_i Y^{107-i}$
0	—	35 to 0	71 to 36	
		($2 = 0$ to $i = 35$)	($i = 0$ to $i = 35$)	74, 73, 72
1*	$i = 0$	36 to 1	72 to 37	74, 73
		($i = 0$ to $i = 35$)	($i = 0$ to $i = 35$)	
2	1,0	37 to 2	73 to 38	74
3	2 to 0	38 to 3	74 to 39	—
4	3 to 0	39 to 4	74 to 40	—
			($i = \mathrm{m} - 3$)	
20	19 to 0	55 to 20	74 to 56	—
m	$(m - 1)$ to 0	$35 + m$ to m	74 to $36 + m$	—
			($i = m - 3$)	
34	33 to 0	69 to 34	74 to 70	—
35	34 to 0	70 to 35	74 to 71	—
36†	—	35 to 0	71 to 36	74, 73, 72

* 0–1 constitutes the coding cycle

† See $m = 0$ values

clock cycles the final position in u_{21} is $a_{13}X^2Y^{60}$ serving as the thirteenth bubble position in the data stream. Further, during the coding cycle i.e., as the exponent of X is changing from 0 to 1, the location of the initial position is at Y^{45-20}, i.e., at Y^{25} serving as an active bubble carrying the status of the 13th line in Fig. 4. During its next coding cycle, the same bubble position is at Y^{60-2}, or at Y^{58} corresponding Y^{58-36}, or at Y^{22} during coding, serving as an inactive bubble position between lines 11 and 12 in Fig. 4.

4.4.2 *Detection of the Nonexistence of Bubbles*

For the correct functioning of the line scanner, all the bubble positions should carry bubbles. Sixteen lines are actively used; and the status of these lines is carried by bubbles in positions 1, 3, $\cdots$ 31. The bubble positions 33 and 35 always carry the status of two fictitious lines (on on-hook and next off-hook) to check the correct operation of the overall magnetic and electronic circuitry. Generally, it is also desirable to check if all bubble positions do carry bubbles by sensors S_1 and S_2 which are capable of detecting only the off-hook status of lines. Such an inspection can be effected when each of the bubble positions is arranged to periodically occupy the position 35, which should always carry the off-hook status. All the bubbles are moved to this status as they traverse the position 74.

Examine the bubble position at Y^{35} just prior to coding. After 36 cycles (i.e., next coding) the bubble position now at $Y^{(35-36) \bmod 75}$, i.e., Y^{74} will occupy Y^{35}. In general, after n coding cycles, the present position $Y^{(35-36n) \bmod 75}$ will occupy Y^{35}. The exponent of Y generates a series 35, 74; 38, 2; $\cdots$, $(35 + 3(n)/2)$, $74 + (3(n - 1)/2) \bmod 75$; $\cdots$, 32, 71; repeating every time n reaches 25. This indicates that the bubble positions now occupying Y^{36}, Y^{0} $\cdots$, etc., Y^{37}, Y^{1} $\cdots$, etc., never occupy Y^{35} at any finite value of n. To eliminate this condition, the electronic circuitry may be programmed* to delay the coding by one clock cycle every 25 coding cycles. This leads to a new location series: 35, 74, 82, 2, $\cdots$, 32, 71; 36, 0, 39, 3, $\cdots$, 33, 72; 37, 1, 40, 4, $\cdots$, 34, 73; 38, 2, $\cdots$, etc. Alternatively, if the loop is designed with 73, 109, 145, etc., periods, then the need for building additional delay circuits will not be necessary. With 73 periods, every bubble position will be located at Y^{35} every 73 coding cycles, or every 2628 (i.e., 73 × 36) cycles and so on.

* The general concept of shuffling periodically was suggested by D. Denburg.

V. CONCLUSIONS

The polynomial algebra is a flexible mathematical tool available for the step by step design of conceived circuits, and for the sequential verification of their operation. Various design parameters may be calculated accurately.

When the operations of numerous circuits are to be synchronized, the algebra provides an excellent insight into their combined functioning. The effect of errors or defects of certain sections of the overall circuitry may also be accurately analyzed by the algebraic modeling of the circuit operation.

REFERENCES

1. Ahamed, S. V., "Multidimensional Polynomial Algebra for Bubble Circuits," B.S.T.J., this issue, pp. 1535–1558.
2. Ahamed, S. V., "The Design and Embodiments of Magnetic Domain Encoders and Single Error Correcting Decoders for Cyclic Block Codes," B.S.T.J., *51*, No. 2 (February 1972), pp. 461–485.
3. Peterson, W. W., *Error Correcting Codes*, MIT Press, Cambridge, Mass., 1968.
4. Lucky, R. W., Salz, J., and Weldon, Jr., E. J., *Principles of Data Communication*, McGraw-Hill Book Co., 1967.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

XYTOLR–A Computer Program for Integrated Circuit Mask Design Checkout

By MICHAEL YAMIN

(Manuscript received March 7, 1972)

XYTOLR is a computer program which checks integrated circuit mask designs for compliance with design rules such as those which require minimum clearances between diffused and metallized areas. The input is the same machine readable mask description that is used for automatic mask art work generation. Clearance violations, if present, are reported in graphic and printed form with pointers to the erroneous statements in the user's input.

I. INTRODUCTION

The increasing size and complexity of large-scale integrated circuit mask designs has created a demand for computer aids to the circuit designer. Without such aids, the design of many circuits would be not merely difficult, but practically impossible. Perhaps the best established design aids in this field are computer art work generation systems, which accept as input numerical descriptions of mask layouts, and drive high-precision graphical output devices which create the mask masters. XYMASK,[1,2] used at Bell Laboratories, is such a system.

The use of a computer art work generation system implies that there exists a complete and unambiguous machine-readable description of the masks required to produce an integrated circuit. In principle, this information, together with the physical properties of the substrate and diffused layers, is sufficient to predict any desired property of the completed circuit. The availability of computer programs to predict and analyze important properties of circuit designs in advance of their fabrication would be a major aid to the circuit designer.

One early phase of the design process for which a computer aid is highly desirable is that of checking a mask set, before it enters production, for compliance with design rules such as required clearances between areas delineated by the same or different masks. Deviations from these rules may result in loss of manufacturing yield or early device failure. Visual checking of mask art work for such tolerances is not only tedious and time-consuming, but has frequently been found to miss one or two significant errors in the mask design. The exhaustive patience of the computer is better suited for such important but mechanical verification procedures. This paper describes a computer program called XYTOLR, which tests mask designs automatically for tolerance errors. The data structures and other components of XYTOLR can be used to construct other mask analysis programs.

II. USER INTERFACES—INPUT AND OUTPUT

Computer programs, in general, should require the minimum necessary input from the user, and should accept it in a form which is simple and natural for the user to write. The output should include all the information which the user might desire, but indexed and organized so that specific items can be located without extensive searching. An attempt has been made to follow these principles in the design of XYTOLR.

2.1 *Mask Design Input*

The primary input to the tolerance-checking program is the geometrical description of the mask set in XYMASK[1,2] input language. This is the same input deck (except for a few command statements) that is used for mask art work generation via XYMASK. Since such decks can run to several thousand cards, it is unreasonable to ask the user to make any general alterations for the benefit of XYTOLR. To preserve language compatibility, a version of the XYMASK language processor is incorporated in XYTOLR to read this input.

In the XYMASK language, the mask designer defines closed plane figures composed of line and circular arc segments. Each such defined figure is called an "atomic," and its definition includes a symbolic name (mask level name) which assigns the atomic to a specific mask of the set being designed. An atomic may also be given a label (atomic name). By the use of statements referring to this label, an atomic may be translated, rotated, reflected, and repeated many times at different locations in the mask. Groups of atomics may be combined into compound structures called "clumps," which may also be labelled and manipulated like atomics. A "clump" may include atomics assigned to different mask levels. In this way, devices such as transistors which require shapes on several masks can be designed and manipulated as units.

While XYMASK permits atomics to be defined without labels (atomic names), XYTOLR requires a name for each atomic. A name is therefore generated for any atomic not named by the user. Such names are of the form ATnnnnn, where nnnnn is a number pointing to the location (line number) of the atomic definition statement in the XYMASK deck listing. The user thus can find the statement to which the generated name refers.

The structure described by a XYMASK deck consists essentially of many geometrical figures distributed among a number of mask levels. Each figure is called an "occurrence." XYTOLR assigns a serial number to each occurrence. A single atomic definition may result in many occurrences, as a result of repetitions.

2.2 *Tolerance Test Input*

XYTOLR, like XYMASK, is technology-independent, dealing only with the geometry of the mask design, and knowing nothing about design rules or the device functions of the various masks. The user must provide input which describes the "tolerance tests" he wishes the program to conduct. A "tolerance test" asks for a report of all instances in which the clearance between elements on specified mask levels is less than a specified quantity. The operation involved in a tolerance test is as follows: enlarge each occurrence on a designated mask level by some quantity and report all new connections which develop, and old connections which are broken, between these occurrences and the occurrences on another designated mask level. The enlargement quantity is generally a little less than the specified minimum clearance, so that clearances just at the minimum are not reported as errors. If checking of clearances within a mask level is desired, the operation is: enlarge each occurrence by a little less than half the desired clearance and report all new connections within the mask level.

A simple, free-form language is used to specify tolerance tests. For example, suppose one wishes to assure that every emitter in a bipolar integrated circuit lies no closer than 0.5 units to the boundary of the base diffusion which surrounds it. (The units are the same as those used in the XYMASK design deck.) The statement

EMITDIF +.5 BASEDIF

results in every emitter being enlarged by a little less than 0.5 units, and any new contact with the base diffusion boundaries being reported.

Suppose one wishes to check that every occurrence on a metallization mask clears every other occurrence by 0.3 units. The statement

/ THINMET +.15

results in every occurrence on the THINMET level being expanded by a little less than 0.15 units, and any new connections within that level being reported.

A negative number results in contraction of occurrences, and may be used to test for minimum overlaps.

The tolerance test language has the capability of describing quite complicated test patterns within a statement. It includes control words which enable the user to modify the normal handling of a tolerance test by XYTOLR: for instance, expand by exactly the quantity requested rather than a little less. Any number of such statements may be submitted, and each will induce a tolerance test and generate a report.

In practice, a mask designer working on a particular type of circuit will have developed a standard test protocol to be applied to all of his designs. Thus, specifying a tolerance test sequence will consist simply of selecting the appropriate tolerance test input deck for the circuit type being tested.

2.3 *Masklevel Connectivity Input*

The user may know that tolerance tests between certain mask levels will never be made, either because these mask levels can never be in physical contact (for example, a metallization and a buried collector in bipolar technology) or because there is no current interest in such a test. XYTOLR can accept input that tells it which mask levels are to be considered capable of connection, and which are not. It is never necessary, in principle, to supply such input, but, for large mask designs, considerable savings in running time and memory occupancy can result if it is supplied in advance of all tolerance test input.

2.4 *Output*

Both printed and graphical reports are generated by XYTOLR. For each tolerance test, a plot is produced of all occurrences involved in violations of the tolerance rule. Each plotted occurrence is identified with its serial number. Comparison of this plot with drawings of the individual masks (provided on the same scale) facilitates locating the errors in the mask design.

The drawing of tolerance errors is accompanied by printed output identifying each pair of occurrences involved in a violation. The occurrences are identified by atomic name, mask level name, occurrence serial number, and coordinates. This information makes it possible to find the offending statements in the XYMASK input deck.

Self-intersecting shapes, which are not permitted by some precision plotters although they are accepted by XYMASK, are also reported.

The program has been designed to err on the side of strictness in reporting tolerance violations. Marginal cases are reported as violations, leaving it up to the user to decide whether he considers them acceptable. If no violations are found, that fact is reported explicitly.

Figures 1 and 2 are examples of XYTOLR output. Figure 1 shows a simple metallization pattern. The clearance desired between metallized areas was 0.3 units. A number of occurrences in violation of this rule were deliberately inserted into the XYMASK description of the mask.

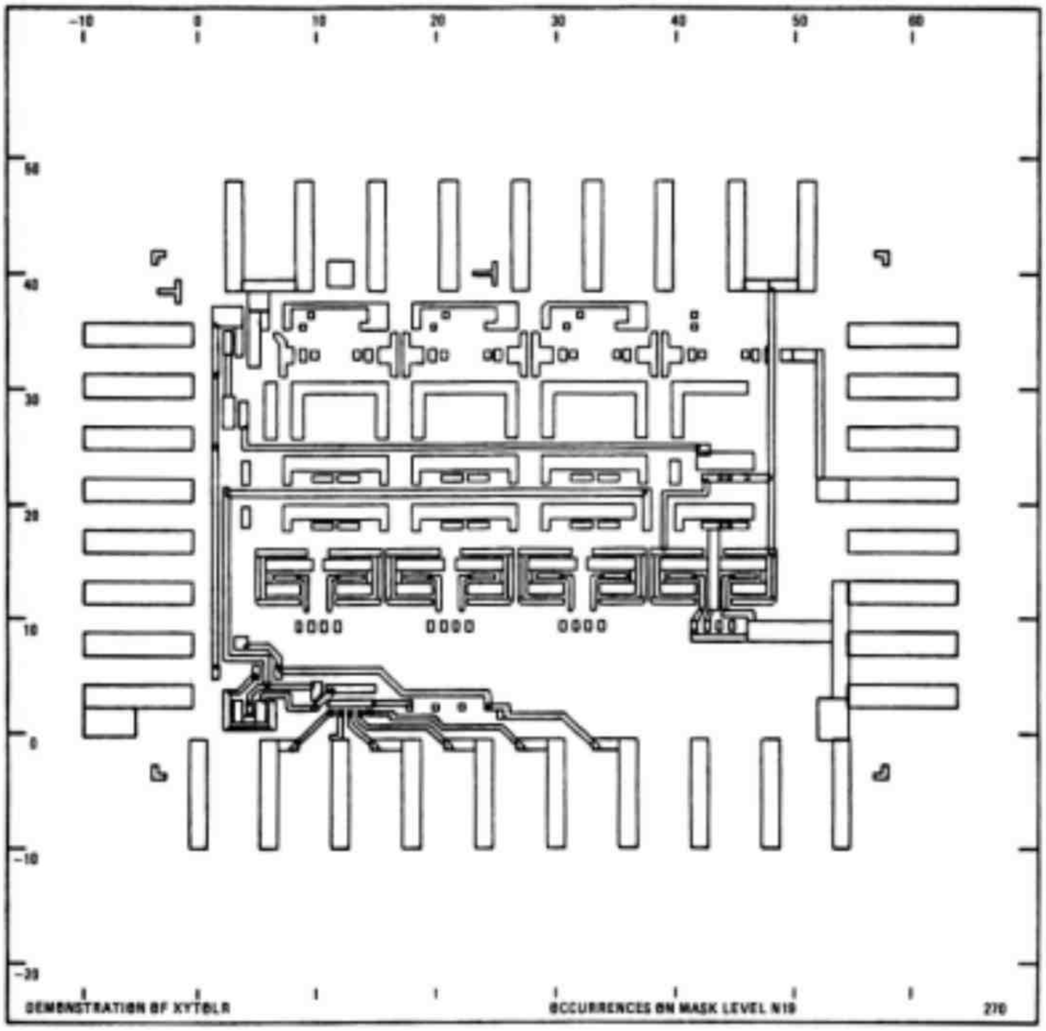


Fig. 1—A simple metallization pattern. Two shapes have been inserted in violation of the required minimum clearance of 0.3 units.

A tolerance test was induced by the input statement

/ N19 +0.15 NODEXX.

NODEXX is a control word requesting that new connections (violations) involving occurrences already indirectly connected through other occurrences (i.e., members of the same node) be plotted separately from new connections between occurrences not so connected (i.e., members of different nodes). Figure 2 is the graphic output illustrating the latter set of violations, which are the ones of interest in this case.

III. PROGRAM STRUCTURE

3.1 *Data Structures*

The output of the XYMASK language processor is a sequential file or tape containing an explicit description of each occurrence in the mask set. Normally, this file is used to drive a precision plotting device. XYTOLR, however, needs parallel access to many occurrence descriptions and therefore must have a randomly accessible data base, in core memory, or on random access disc or drum. Since XYTOLR has been implemented on a computer (HIS 6070) with a large core memory, its data structures are held entirely in core. The principles of operation would not be significantly different for virtual memories or paged memories, or for random access external storage.

The information contained in the XYMASK output file (as generated by a somewhat modified version of the XYMASK language processor) is converted to a data structure, the unit of which is the "occurrence block." Each occurrence block describes one occurrence. The length of each occurrence block is $11 + 2L + 6A$ core locations (36 bit words), where L is the number of line segments, and A is the number of arc segments, composing the occurrence. All geometric information is stored as single precision floating point numbers.

Occurrence blocks are stacked sequentially in memory. Each occurrence block contains, along with line and arc descriptions, a header containing the extreme coordinates of the figure (circumscribed rectangle), its serial number, an index defining its atomic name, and pointers which serve to organize the data structure. By means of these address pointers, each of which is the address of an occurrence, the occurrence block is made a member of three linked lists. By following one list ("main list") from the beginning, every occurrence is encountered in order of increasing x (min), which is the minimum x-coordinate in the figure (including projecting arcs). By following another list, every occurrence of the same atomic name is found. The third list pointer is the "intersection ring pointer," by which all occurrences which

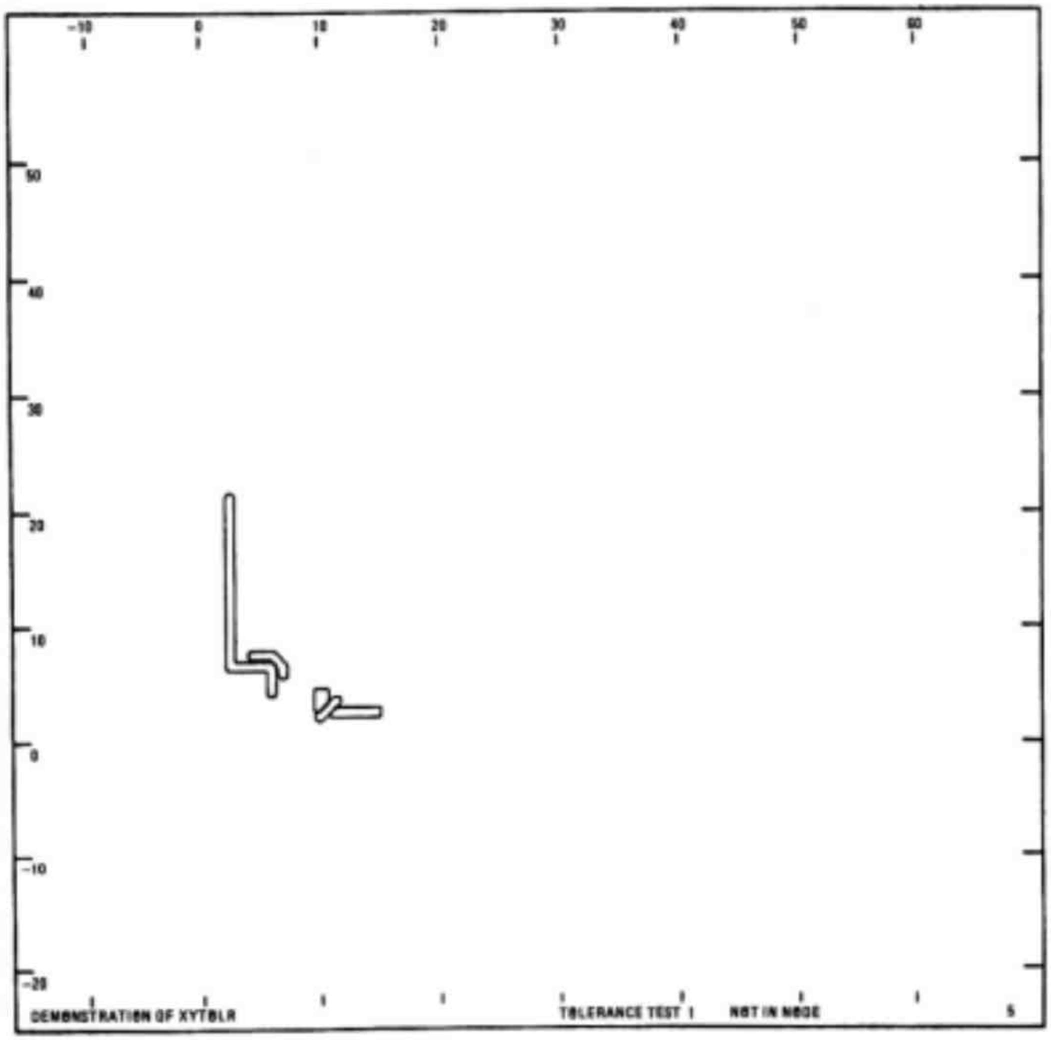


Fig. 2—Output of an XYTOLR test on the pattern of Fig. 1. The input statement was / N19 +0. 15 NODEXX.

contact one other directly or indirectly are ultimately chained into a ring, or circular list, which represents a node.

The data structure also contains, among other things, a table of atomic names and of mask level names. Cross-referencing among these tables and the occurrence blocks facilitates finding the address of a desired occurrence block or the properties of an occurrence the block address of which is known.

One location within the data structure contains an important number known as the "critical distance." Two points separated by less than this distance are considered coincident. The critical distance, while easily variable, is generally set as 0.0001 times the overall mask size. The quantity by which tolerance test enlargement parameters are modified, as described in Section 2.2, is 1.1 times the critical distance.

3.2 *Subroutines*

The working tools required to perform analyses of the data structure are subroutines which generate, manipulate, and display occurrences, given their block addresses. Using these subroutines, along with the crossreferencing provided by the various pointers and tables, a programmer may construct systems which operate on mask levels or on the entire mask set without concerning himself about the detailed structure of occurrence blocks. Many such subroutines are contained in XYTOLR, performing both simple and complex functions. Two of particular importance are CONTAC and NLARGR.

CONTAC tests a pair of occurrence blocks (call them "a" and "b") for connectivity. It returns one of four numerical values, each corresponding to one of the four possible classes of connectivity of a and b: disjoint, a inside b, b inside a, and overlapping (intersecting boundaries). CONTAC first checks the extreme coordinates of the two occurrences to see whether they are clearly disjoint. If not, the test for overlapping boundaries is made: each side of a is solved analytically with a side of b to see if there is an intersection within the bounds of both sides. (This analytical solution is returned by a subroutine called INSECT, which handles lines and arcs interchangeably.) If no such intersection is found, the "inside" test is performed: a line segment is generated from a point on the smaller occurrence (in area) to a point outside the larger one, and the number of intersections of this line segment with the larger occurrence boundary is found. If this number is odd, the smaller occurrence must be within the larger one; if even, they are disjoint.

NLARGR expands or contracts occurrences. Given the address of an occurrence block and a positive or negative expansion distance, it generates a new occurrence block representing the expanded occurrence. In its normal mode of operation ("square mode"), it moves each occurrence side perpendicular to itself by the specified distance and extends it to intersect its relocated neighbors. NLARGR has a more precise, but more expensive, mode of operation ("round mode") which inserts arcs in outside corners so that the new outline truly represents all points equidistant from the original.

IV. OPERATION OF XYTOLR

XYTOLR checks a mask design for tolerance and clearance errors by the systematic application of subroutines NLARGR and CONTAC to the data structure representing the mask design. If the connectivity value returned by CONTAC for a pair of occurrences is changed when one or both of the occurrences is expanded by the prescribed clearance distance, a violation of the tolerance rule is indicated.

XYTOLR constructs an essentially complete connectivity matrix for all the occurrences in the mask set. The data structure is then modified in accordance with the user's instructions so that all occurrences on the appropriate mask levels are enlarged or contracted; each matrix element is recomputed and those elements which differ from the original are flagged as errors.

In principle, a complete connectivity matrix for a mask set would have as many elements as the square of the number of occurrences in the mask set—a million matrix elements for a thousand-occurrence mask set. The problem of storing such a large matrix is simplified by the following considerations:

(*i*) The matrix is diagonally symmetric and the principal diagonal is of no interest.

(*ii*) Each matrix element may have only four values (the four types of connectivity) and therefore may be represented by a two-bit binary field.

(*iii*) The matrix is generally sparse; that is, most of its elements are zero, and it may usually be reduced to a set of submatrices with a much smaller total number of elements.

(*iv*) The manner in which the tolerance test is conducted permits the matrix to be scanned in a predetermined order rather than randomly; therefore, it may be kept in peripheral storage with only relevant elements in core at any time.

XYTOLR first constructs the occurrence-block data structure from the modified XYMASK output file. At this stage, the main list pointer of each occurrence block points to the next sequential block, and the intersection ring pointer of each occurrence block points at the block itself. A sorting program rearranges the main list pointers so that occurrences can be followed in order of increasing x (min).

Next, the "nodes" of the data structure are identified. A node is a set of occurrences each of which is connected to every other, either directly or indirectly through other members of the set. (Two occurrences are considered "connected" if they are not disjoint.) If the data structure represented a metallization pattern, these nodes would be the electrical nodes; the term is here generalized to mean any set of interconnected occurrences. To find the nodes, each occurrence is tested for contact with other occurrences using CONTAC. The first occurrence in the main list is tested against each subsequent occurrence, similarly the second, etc. The list need be followed in each case only until an occurrence is found the x (min) of which is greater than the x (max) of the test occurrence. Geometrically, each occurrence is tested against all those which impinge on a vertical strip defined by its minimum and maximum x-coordinate.

When two occurrences are found to be in contact (i.e., not disjoint), their intersection ring pointers are interchanged. This operation is sufficient to combine two isolated occurrences into a ring (circular list), to add an isolated occurrence to an existing ring, or to combine two existing rings. (Occurrences already members of the same ring are not further tested against one another.) When the operation is complete, each node is represented in the data structure as a separate circular list. Isolated occurrences are "one-membered" rings.

Next, the intersection matrix is generated. Since an occurrence must be disjoint with any occurrence not a member of its node, only matrix elements relating members of the same node need be computed. Clearly, the more nodes, and the fewer occurrences per node, the fewer matrix elements need be computed. At this point, the nodes and their submatrices are stored on two sequential files, the "node file" and the "matrix file." The data structure is scanned for nodes; each time one is found, its occurrences are written out sequentially; as each occurrence is written, its connectivity values with subsequent members of its node are determined and written out as a sequence of two-bit fields. The files thus generated are used for all subsequent tolerance tests.

Each tolerance test statement in the user's input is decoded and the appropriate enlargement for each mask level determined. For each tolerance test, the data structure is reconstructed using occurrence blocks read back from the node file. As each occurrence block is read, its mask level is determined and the required enlargement applied, if necessary, using NLARGR. When the new data structure is complete, the connectivity values relating occurrences in each node are recomputed in the same order as that in which they appear on the matrix file and compared with corresponding values read back from that file. Any failure of these values to match indicates a tolerance test violation, and output routines are called to add descriptions of the pair of offending occurrences to the output report and write the occurrence blocks on a graphics file for later plotting. Finally, each occurrence is tested for connectivity with occurrences not members of its node; any contact indicates an error.

The generation of the original noded data structure and of the node and matrix files, which are used in all tolerance tests, incurs a considerable overhead expense for large mask sets. The user is therefore given the option to have all this information saved on a magnetic tape as soon as it is generated. The data structures and files can be reconstructed efficiently from this tape if further tolerance tests are necessary, or in case of abnormal termination of the computer run before all tolerance tests have been completed.

V. EXTENSIONS OF THE XYTOLR SYSTEM

The program XYTOLR is built about a data structure and a set of subroutines which generate, manipulate, and display elements of this data structure. These subroutines can be used to construct mask design and analysis programs other than XYTOLR. The programmer need never concern himself with the detailed structure of the occurrence blocks or with computations involving the fundamental geometric data, but may interact with the data structure almost entirely through subroutine calls which provide the information or perform the actions logically required by his program.

The program ENMASK, assembled mainly from XYTOLR components in this way, creates new mask level descriptions from XYMASK originals. The new descriptions, which are punched in XYMASK input language, represent mask levels on which each original shape has been expanded or contracted by a given quantity, so that it surrounds, or is surrounded by, the original shape. ENMASK can also, if desired, merge connected shapes, replacing each connected set of occurrences by the outer boundary of their union. This program provides a simple solution to certain mask design problems. It utilizes an occurrence-manipulating subroutine called OVRLAP,[3] which, given two occurrence blocks, can generate new occurrence blocks representing each simple closed figure resulting from their superposition.

D. G. Schweikert has used XYTOLR components to construct a system called ICSORT, which can "decode" an XYMASK mask set description representing a bipolar integrated circuit, automatically identify the transistors, diodes, and resistors, and generate a node list topologically equivalent to the circuit diagram. The designer can thus assure himself that the masks he has designed in fact represent the desired circuit.

VI. IMPLEMENTATION

XYTOLR has been implemented on a Honeywell Information Systems (formerly General Electric) 6070 computer, under the GECOS operating system. The component programs of XYTOLR have been written principally in FORTRAN IV. Assembly language subroutines have been used sparingly to perform functions unique to the HIS 6070. Some FORTRAN code involving character manipulation (for titles, etc.) has been designed on the assumption of a six-character machine word and would have to be modified to execute on a computer with a different internal character representation. Graphic output may be obtained on such devices as the General Dynamics SC-4060 microfilm plotter. Calls to the Bell Laboratories plotting subroutines have been centralized for easy replacement.

XYTOLR occupies a core region the size of which it varies dynamically according to the current size of the occurrence-block data structure. The size of the region is usually limited by system considerations to about 96000 36-bit words. Of this, the program (which is heavily overlaid) occupies about 20000 words, and the rest is available for data. Under these conditions, 2800 to 4000 four-sided figures, or 2200 to 2800 eight-sided figures, can be accommodated, the exact number depending on the manner in which the mask set design is organized.

Running time of XYTOLR depends on the size and complexity of the mask design and on the tolerance tests requested. Running XYTOLR on the simple metallization pattern of Fig. 1 took 62 seconds of processor time, including generation of tape to drive the microfilm plotter. The tolerance test itself took 18 seconds, and additional tests would add about this cost apiece. Extensive checking of large and complex mask sets has on occasion required as much as an hour or more of processor time. The computer charges necessary for such checkout are considered minor compared to the overall cost of the mask set.

VII. CONCLUSIONS

A computer program called XYTOLR tests integrated circuit mask designs, described in the XYMASK input language, for conformity with tolerance rules which require minimum clearances between different regions and minimum overlaps between regions intended to be connected. Tests are specified in a simple input language. Graphical and printed output is produced which identifies all deviations from the specified rules and facilitates their correction. In the HIS-6070 computer environment at Bell Laboratories, mask sets with as many as 3000 to 4000 individual shapes may be processed.

Mask designers have found XYTOLR to be a valuable tool for final mask checkout before production approval is granted. It is especially impressive when the program finds one or two demonstrable errors in a mask set which had been considered completely checked and ready for production. The program has proved reasonably easy to use and the output reasonably easy to interpret.

The techniques, data structures, and subroutines which constitute the XYTOLR program may be used as the fundamental building blocks from which other mask analysis and design programs may be constructed.

REFERENCES

1. Fowler, B. R., "XYMASK," Bell Laboratories Record, *47*, No. 6 (July 1969), pp. 204–209.
2. Gross, A. G., Raamot, J., and Watkins, Mrs. S. B., "Computer Systems for Pattern Generator Control," B.S.T.J., *49*, No. 9 (November 1970), pp. 2011–2029.
3. Yamin, M., "Derivation of All Figures Formed by the Intersection of Generalized Polygons," B.S.T.J., this issue, pp. 1595–1610.

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Derivation of All Figures Formed by the Intersection of Generalized Polygons

By MICHAEL YAMIN

(Manuscript received March 7, 1972)

*A computer program is described which generates every intersection figure resulting from the superposition of two closed polygon-like plane figures, each consisting of an arbitrary number of line segments or circular arc segments. Each intersection figure is assigned to one of four regions of the plane, representing the union, the intersection, and the two "exclusive-*OR*'s" formed by the pair of input figures. The two input figures may intersect or be tangent at any number of points and may have sections of coincident boundaries. No grid approximation is used. The program operates in two stages: the first stage analytically finds and classifies every point of intersection or tangency of the figures; the second stage regards these points as the nodes of a graph and applies an algorithm which causes each intersection figure to be traced just once.*

I. INTRODUCTION

In the course of a project related to computer-aided integrated circuit mask design, it became necessary to describe the configuration which is formed when two polygon-like plane figures are superimposed on one another. Two closed figures in a plane divide the plane into four regions: inside both figures, inside the first but not the second, inside the second but not the first, and inside neither. Each region may consist of one or more figures. Assuming that the original figures consist of an arbitrary number of sides, each of which may be a line or circular arc segment, it was desired to describe every figure resulting from their intersection and to assign each to one of the four regions.

The simple approach of establishing a grid of points and determining which sets of points are included in each region was not considered applicable, because the number of grid points required for sufficient resolution would have led to excessive computation. Instead, the following approach was used. Pairs of sides, one from each input figure, were investigated analytically for points of intersection or tangency, or colinear sections. From the resulting information, a table of intersection points was developed. Included in this table was information concerning the type of the intersection: crossing, tangency, beginning or end of a colinear section resulting in a crossing, or beginning or end of a colinear tangency. With the intersection table available, the collection of line and arc segments constituting the superimposed figures could be considered as a graph, the intersections being the nodes. An algorithm for tracing the graph so as to generate every intersection figure once and only once was developed. The tracing paths thus determined were used to select appropriate analytical data from the original figure descriptions so as to generate descriptions of the new figures in the same numerical formats as the original figure descriptions. Classification of the figures was an automatic result of the tracing algorithm.

II. NUMERICAL REPRESENTATION OF FIGURES

Both input and output figures are described as sequences of sides in a numerical format designed for efficiency of computation rather than compactness of storage. The rotational tracing sense (clockwise or counterclockwise) of each figure is specified, and thus each side has a direction. A side, which may be a line or circular arc segment, is represented by eight sequential words in memory: the starting coordinate pair, parameters of the analytical equation, and the terminal coordinate pair. Since the terminal point of one side is the starting point of the next, each additional side requires six additional words of storage. Each figure is explicitly closed with a side, which terminates at the starting point of the first side.

The side sequence of each figure is preceded by a header containing information about the figure as a whole: its extreme x and y coordinates, allowing for projecting arcs; its area; the number of sides; and the sense (cw or ccw) of the entire figure. The header also contains pointers which can be used to associate the figure with externally tabulated information or can be used to associate it with other figures in one or more linked lists. Figure descriptions are stacked sequentially in memory; specific figures are located by pointers into this data structure. The exact arrangement of pointers and external tables depends on the application.

Two limitations are imposed on the input figures; they must nowhere intersect themselves; and no side should be colinear with an adjacent side. It is possible to override the second limitation with a preprocessing program which combines such sides and condenses the figure description appropriately.

III. DEVELOPMENT OF INTERSECTION TABLE

Given two figures described in the above format, it is desired to list all their points of intersection. This list of intersections will include the coordinates of each intersection point, indices assigning it to a specific side of each of the two figures, and a code defining it as a crossing point, a tangency, one end of a colinear section topologically equivalent to a crossing, or one end of a colinear section topologically equivalent to a tangency.

It is typical of this entire analysis that much of the work is involved with the treatment of special cases. When the middle of one side is intersected by the middle of another, it is no trouble to make the appropriate entry in the intersection list. The problems arise when intersections involve more than one side of each of the input figures, as when they occur at corners, or when colinear sections "wrap around" several sides of each figure. To handle these cases, the intersection table must be developed in stages, entries ambiguous at a given stage being flagged as such and clarified in the next stage.

One of the input figures is arbitrarily declared the operating, or A, figure; the other the passive, or B figure. The elementary process of the intersection table development is the solution of the side equation of one side of A with one of B to find any intersections, tangencies, or colinearities within the bounds of both segments. A solution which is detected just at the end of one of the sides can not generally be classified as a crossing or tangency at this stage. The second stage is to repeat the process, with the same side of A, for every side of B, so that a table of points of intersection for this one line or arc segment with B is created. At this stage, while ambiguities resulting from intersection at a corner of B have generally been cleared up, solutions at the ends of the A test side are still not finally classified. The third and last stage is to repeat the above for every side of A, pairing up unclassified solutions which correspond to intersections at corners of the A figure.

Each of these stages of analysis is performed by a subroutine which loops on the previous one. Thus, a subroutine called INSECT returns the point or points of intersection of two line or arc segments (that is, those solutions of their equations which lie within the bounds of both segments) or reports their colinear coincidence. SLASH returns the points of intersection of a line or arc segment with a closed figure consisting of such segments; classifying as crossings, tangencies, etc., those intersections which do not involve the ends of the segment and returning special codes to provide what information it can about the others. INTLST returns the points of intersection of two closed figures consisting of line and arc segments, each intersection being classified as a crossing, a tangency, or the start or end of a colinear tangency or crossing. This is the "intersection list" previously described; the intersections appear in the order in which they would be encountered when tracing the A figure. Two special situations are recognized: if the figures nowhere intersect, an intersection list of zero length is returned; if the figures are coincident (everywhere colinear), a special code is returned.

In principle, this procedure requires a number of calls to INSECT equal to the product of the number of sides of the two figures. However, each of the subroutines applies a quick preliminary test for overlap of the circumscribed rectangles of the input line segments or figures. In most cases, this avoids a great deal of detailed computation.

INTLST and SLASH use a complicated sieve of analytical tests to determine the nature of intersections which occur at figure corners and involve three, sometimes four, arc or line segments. Typically, these tests must decide whether the intersections represent a crossing or tangency of the two figures, and the geometrical question to be resolved is usually whether the direction of incidence of one line or arc segment on another segment, or on the point of intersection of two other segments, is from the left or from the right. In the case of lines, a vector cross product is used to make this determination; in the case of arcs, which may be tangent at a point of intersection, logic involving the direction of the arc center is required.

All three levels of geometric analysis make use of a quantity called the "error margin," which is necessary to allow for computational inaccuracies. The assumption is made that two points closer together than this small distance are in reality the same point. The error margin differs from a grid approximation in that its magnitude has no influence on computation time.

One more item of information must be added to the intersection list for the purposes of figure tracing. This is the incidence direction of the A figure on the B figure, and of B on A, at every intersection. The incidence is a two-valued parameter; one value corresponds to *entry* of one figure into the other at a crossing intersection or *inside incidence* at a tangency; the other value corresponds to *exit* at a crossing or *outside incidence* at a tangency. There are obviously four types of intersection, corresponding to the combinations of the two possible incidence values of A-on-B with those of B-on-A. It is convenient to determine these values as part of the program next to be described, rather than as part of INTLST.

IV. GENERATION OF INTERSECTION FIGURES

The development of the intersection table described above has been a geometrical problem in that finding the coordinates and the nature of each intersection has required knowledge of analytical parameters describing the exact position of each geometrical entity in the plane. With the intersection table in hand, the problem of tracing each intersection figure becomes topological in nature. The overlapping polygons may be considered as a graph, that is, a set of intersection points, or nodes, between some of which exist connecting paths, or edges. For the purpose of figure-tracing, it is immaterial whether an edge between two intersections consists of one or many sides, or whether they are lines or arcs. It is enough to know to which figure the edge belongs.

An algorithm has been devised by which each circuit of the graph which corresponds to an intersection figure may be traced once and only once. The sequence of edges constituting such a circuit may then be used to select analytical information from the input figure descriptions, from which a description in similar format of the intersection figure represented by the circuit may be assembled. Since each edge contains no intersections, it must belong to a single figure, which is known. Its start and end points, and the figure sides upon which they lie, are associated with its terminating nodes as listed in the intersection table. The tracing direction of a given edge may be the same as or opposite to the direction of the original figure. The generation of the intersection figures then consists essentially of copying sequences of sides or partial sides, forward or backward, as guided by the edge data and the edge sequence in the specified circuit. The region of the plane to which the intersection figure belongs is determined from the incidence information.

The most direct way to implement such a procedure is based on the fact that each edge is part of two intersection figures. The method is to list, in cyclic order, all the edges radiating from each node. Then, starting from any node along an edge, proceed to the next node and transfer to the adjacent edge in cyclic order, clockwise or counterclockwise as appropriate, so that the trace remains within a "cell" of the graph. Continue until the starting node is reached, then repeat with a new starting node. The complete set of intersection figures will be traced when each edge has been traversed just twice (a colinear section being considered one edge). Unfortunately, no explicit numerical description of the edges connecting the nodes is available at this stage in the analysis; the necessary information must be computed from the input figure descriptions, the side blocks of which do not, in general, explicitly represent edges of the graph.

The algorithm actually used, therefore, centers its attention on the nodes (intersection table entries), rather than the edges. As the trace encounters each node, the algorithm specifies the next node which must be encountered to keep the trace within a "cell" of the graph; the numerical description of the connecting edge is only then generated and posted directly to the output figure description. In principle, each node must be encountered four times for completeness, once for each of the figures of which it is a corner. The implementation of this test for completeness is complicated by special considerations which arise at nodes which are tangencies or colinearities; while the verbal description of these special cases in Section 4.1 appears complex, their implementation in computer code requires only minor modifications in the program flow.

The circuit tracing and intersection figure generation operations are embodied in a single subroutine called OVRLAP. This subroutine calls INTLST to establish the intersection table, in which the intersections are listed in A-figure order. For efficiency, it is desirable to have a table of intersections in B-figure order. This could be created by another call to INTLST, but it is much more efficient to produce it by reordering a copy of the A-figure intersection table. The two tables are crossreferenced so that any entry in one can be located in the other. The OVRLAP subroutine also finds, and tabulates, the A-on-B and B-on-A incidence values at each intersection as described previously.

OVRLAP is a modular program, with modules (procedures) which perform elementary functions such as:

(*i*) Extract a side from the A or B figure.
(*ii*) Insert a side in an output figure description.
(*iii*) Replace a side description by a description of the same side, but backwards.
(*iv*) Given a side description and a point on the side, truncate the side to return the first or second segment.

Other modules use these to perform higher-level functions, such as:

(*i*) Insert in an output figure description a sequence of sides from one intersection to another, derived from the A or B figure as required, forward or backward as required.

(*ii*) Generate an output figure by tracing a complete circuit, according to the circuit-tracing algorithm, from a given intersection back to itself.

(*iii*) Copy an input figure bodily as an output figure, making appropriate changes in the figure header.

4.1 *Circuit Tracing Algorithm*

First, for simplicity, consider a pair of intersecting figures in which all the intersections are point crossings. Tangencies and colinearities introduce complications which require modified treatment.

(*i*) Starting at any intersection, trace two circuits. One is initiated by tracing forward on the B figure, the other by tracing backward on the B figure. Before starting these traces, compute a circuit incidence code for each. This code has four possible values, one for each of the possible combinations of A-on-B and B-on-A incidence values. For the B-forward trace, it combines the two incidence values (B-on-A and A-on-B) stored for the starting intersection; for the B-backward trace, it combines the A-on-B value with the opposite of the B-on-A value.

The procedure of "tracing" consists of going to the next intersection on the appropriate figure in the chosen direction. The "next intersection" is an adjacent item in the A-figure or B-figure intersection table, as appropriate. The last item on each list is considered "adjacent" to the first.

(*ii*) At the next intersection:

If tracing on B, switch over to A backwards.

If tracing on A, switch back to B in the starting direction.

Label every intersection encountered in this way with the circuit incidence code for the current trace. Each item block in the B-figure intersection table has space to store four such code labels.

Repeat step (*ii*) until the starting intersection is encountered, completing the circuit.

(*iii*) On completing both circuits from the current starting intersection, go to the next intersection on the A figure and proceed as under (*i*) above. However, do not perform any trace for which the starting intersection is found already to be labelled with its circuit incidence code.

(*iv*) Continue this procedure, making each intersection along the A figure the starting intersection in turn, until every intersection has been the starting intersection.

On termination of this procedure, every circuit on the graph which corresponds to an intersection figure will have been traced just once. Moreover, the circuit incidence code associated with each figure directly assigns the figure to one of the four regions into which the two intersecting figures divide the plane, thus:

Circuit Incidence Code Means	*Region of Plane*
A enters B, B enters A	Inside A, inside B (A and B)
A enters B, B leaves A	Outside A, inside B (B and not-A)
A leaves B, B enters A	Inside A, outside B (A and not-B)
A leaves B, B leaves A	Outside A, outside B (not-A and not-B) (A or B)

(The last category represents the logical union of the two figures. Such a figure may be a "hole" in the union, or the outer boundary of both figures. In the latter case, the inside of the figure is logically the complement of not-A and not-B, namely, A or B).

This algorithm follows from the fact that each crossing intersection forms a corner of four figures, each in a different region of the plane as described above. Step (*i*) "stakes a claim" on two of these figures. (The other two will be or have been claimed by the previous A intersection.) Step (*ii*) assures that the trace stays within the "claimed" figure. Step (*iii*) moves the procedure forward, assuring that no figure previously "claimed" is retraced. Since every intersection figure is "claimed" by at least one intersection, step (*iv*) assures that every figure is eventually described.

A point of tangency between two figures differs from a crossing intersection in that two of the four figures which impinge on it belong to the same region of the plane. The following special considerations apply to ta ıgencies:

(*i*) On encountering a tangency when tracing on the A figure, switch over to the B figure only if the B-on-A incidence value at the tangency is the same as at the starting intersection for a B-forward trace or opposite for a B-reverse trace. When tracing on the B figure, switch to the A figure only if the A-on-B incidence is the same as at the starting intersection. If this condition is not met, go on to the next intersection.

ction along the intersection has

he graph which :aced just once. h figure directly ı the two inter-

Plane

(A and B)
3 (B and not-A)
3 (A and not-B)
: B (not-A and

he two figures. er boundary of is logically the

ing intersection n of the plane f these figures. evious A inter- the "claimed" : that no figure ction figure is res that every

crossing inter- n it belong to considerations

the A figure, e value at the r a B-forward ı the B figure, he same as at go on to the

(*ii*) When switchover from B to A is made at a tangency, do not label the tangency with the circuit incidence code.

(*iii*) When it is the turn of a tangency to be the starting intersection, only one tracing circuit, at most, rather than two, is originated. The permissible tracing direction of B is determined by the following "inside-outside" rule: If the rotation senses of the two input figures are the same (both cw or both ccw), trace backward on B if the B on A and A on B incidence values at the tangency are the same, and forward on B if they are opposite. If the rotation senses are opposite, the rule is reversed. Of course, just as with crossing intersections, no trace is performed if the tangency is already labelled with the appropriate starting incidence code.

Either a tangency or a crossing of the two input figures may occur as a colinear section. Such a colinear section appears in the intersection tables as two adjacent intersections, one labelled as the "start" and the other as the "end" of the colinearity. "Start" and "end" refer to the direction sense of the A figure. The following special considerations apply to colinearities.

(*i*) On encountering either end of a colinearity while tracing a circuit, take the appropriate switchover action immediately, following the rules described above for point crossings or point tangencies as appropriate.

(*ii*) Never begin tracing a circuit at the "start" of a colinearity; go to the "end" of the colinear section and make that the starting intersection as described above.

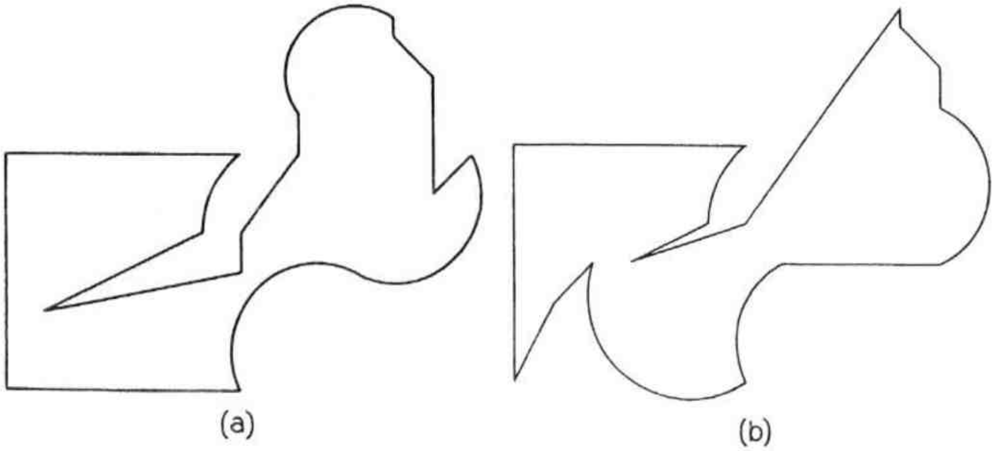


Fig. 1—Two input figures: (a) composed of 16 sides, 12 of them lines and 4 arc segments; (b) composed of 15 sides, 11 of them lines and 4 arcs.

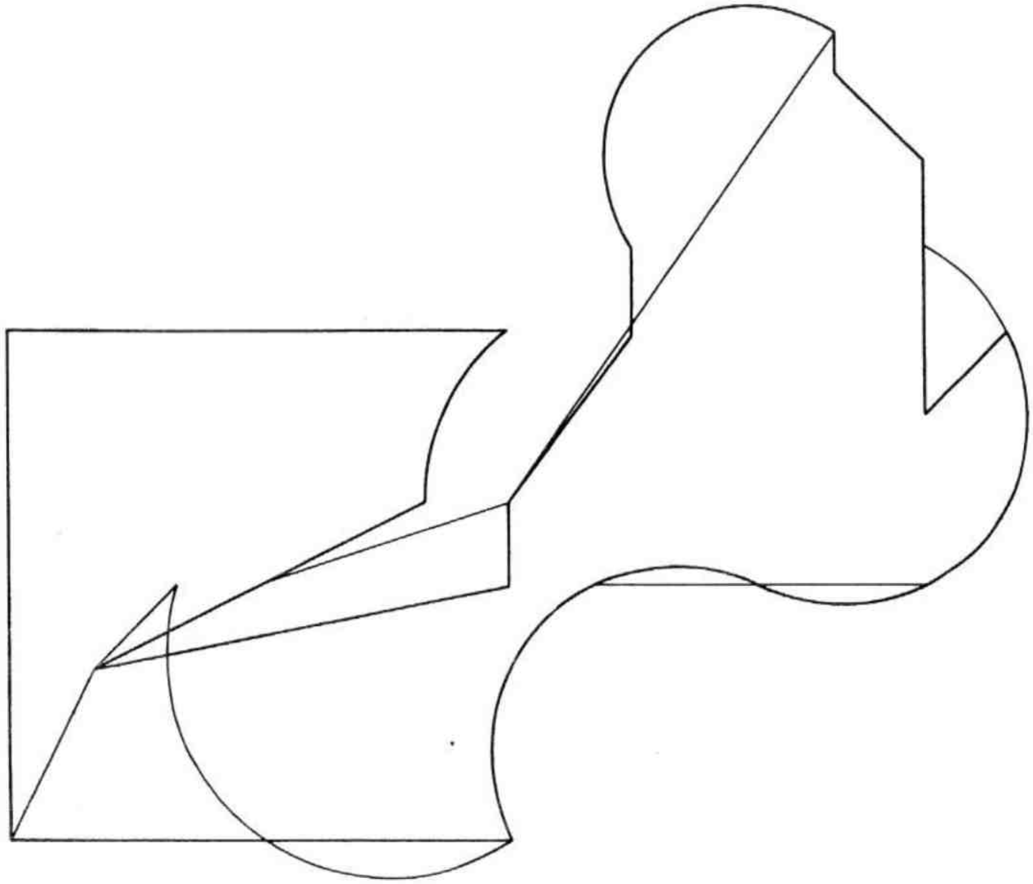
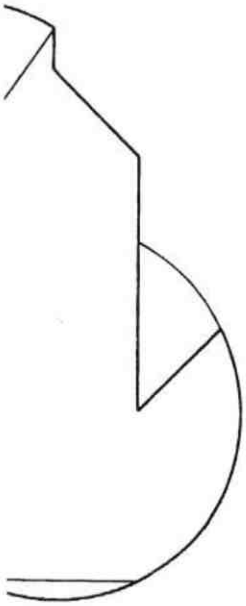


Fig. 2—Graph resulting from superposition of Figs. 1a and 1b.

(*iii*) If a colinear region is being traced, having been entered at the "end" point (whether by switchover or by initiation of circuit tracing), the next intersection encountered will be the "start" point. Do not switch figures at this point; just go on to the next intersection. Any switchover required will already have been done at the "end" intersection.

4.2 *Generation of Output Figure Description*

The algorithm just described sets out a procedure for stepping from node to node in the intersection lists in such a way as to describe certain unique circuits. In essence, it tells one to choose an appropriate item in the A-figure list; find the corresponding entry in the B-figure list; follow the B list item by item, in a specified direction, until certain conditions are met; switch to the corresponding entry in the A-figure list; follow this list backward, item by item, until certain conditions are met; switch to the corresponding entry in the B-figure list; and



. and 1b.

een entered at
ation of circuit
e "start" point.
ext intersection.
e at the "end"

r stepping from
describe certain
ppropriate item
ne B-figure list;
n, until certain
in the A-figure
tain conditions
figure list; and

continue switching back and forth in this way until the starting point is attained.

The numerical description of the output figure which is the objective of this program is generated simultaneously with the execution of this algorithm. As was mentioned earlier, the program contains a module which, given two specified intersection nodes, will generate a sequence of side descriptions representing a path from the first intersection to the second, on the A or B figure as specified, and forward or backward as specified. This module is invoked every time a switch from one intersection list to the other is made. The two intersections specified are that at which the list was entered, and that from which the switch is being made (if there are no tangencies or colinearities, these are adjacent intersections). The figure is that represented by the intersection list, and the direction is that in which the list was followed. The resulting side sequences are stacked one after another in memory,

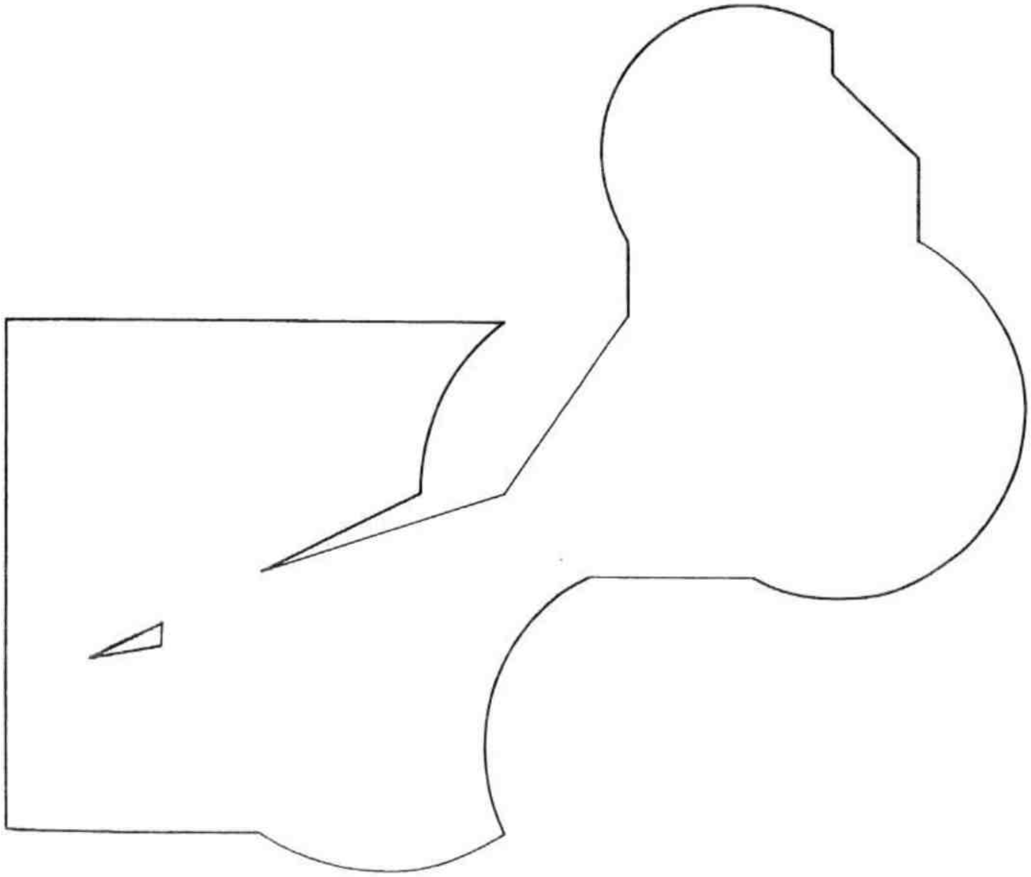


Fig. 3—Figures classified as "A or B"-the union of two input figures.

and when the circuit trace is complete, so is the intersection figure description.

An anomaly can occur as a result of the circuit-tracing algorithm: two successive sides of an intersection figure may be returned which are colinear extensions of one another. It is easy to recognize this situation, and the program module which transfers individual sides to the output figure simply combines any two such sides.

Before a circuit trace is initiated, the intersection figure description must be initialized by allocation of space for its header. After the circuit trace is complete, certain header items must be filled in, particularly extreme coordinate values and the area. The area of a figure is found by a sum-of-trapezoids technique, with corrections for arcs. The area returns as a signed number, the sign denoting the rotational sense (cw or ccw) of the figure. The header includes a code which specifies to which of the four regions of the plane the figure belongs.

It can occur that two input figures have no crossings at all, but only tangencies. One may be inside the other, or they may be external to each other. In this case, the input figures are themselves intersection figures, but they will not be traced by the above algorithm. Therefore, they must be copied bodily as output figures and assigned by simple logic to an appropriate region of the plane. The same is true for figures which do not touch each other at all.

Another problem arises when two figures have one point tangency as their sole intersection. A figure will be generated which is wasp-waisted or self-tangent, either of which violates the rule against self-intersection. The program returns these descriptions regardless, and the calling program which uses OVRLAP must watch out for this situation. Finally, when one figure lies within another without any contact, one region of the plane (inside one figure but outside the other) cannot be described as a simply connected figure, though its area is easily calculated.

4.3 *Organization of Output Data Structure*

Intersection figures generated as described in the previous section are stored sequentially in a region of memory allocated for output. To be useful, this data structure must be organized and indexed in some way. The following system has been used:

(*i*) All the figures of a given type (that is, belonging to the same one of the four regions of the plane) are chained together in a linked list by a pointer in the figure headers.

ersection figure

cing algorithm:
returned which
recognize this
ndividual sides
s.
gure description
After the circuit
in, particularly
figure is found
arcs. The area
otational sense
which specifies
ngs.
ngs at all, but
nay be external
ves intersection
hm. Therefore,
gned by simple
true for figures

point tangency
which is wasp-
le against self-
regardless, and
h out for this
r without any
ut outside the
ure, though its

revious section
ed for output.
nd indexed in

ng to the same
d together in a

(*ii*) A summary table is provided. This table has four blocks, each corresponding to one of the four regions of the plane. Each block contains: the number of figures of this type, the total area of these figures, and a pointer to the first figure in the corresponding linked list.

(*iii*) Each figure header has a pointer to the appropriate block of the summary table, and one to the next sequential figure description in the data structure.

V. DETAILS OF IMPLEMENTATION

The program described in this paper has been coded in FORTRAN IV and compiled and executed on the General Electric 635 computer. It has been coded as a subroutine, called OVRLAP, with three arguments: The locations of the two input figure descriptions and the starting location of a region of storage available for output. In addition,

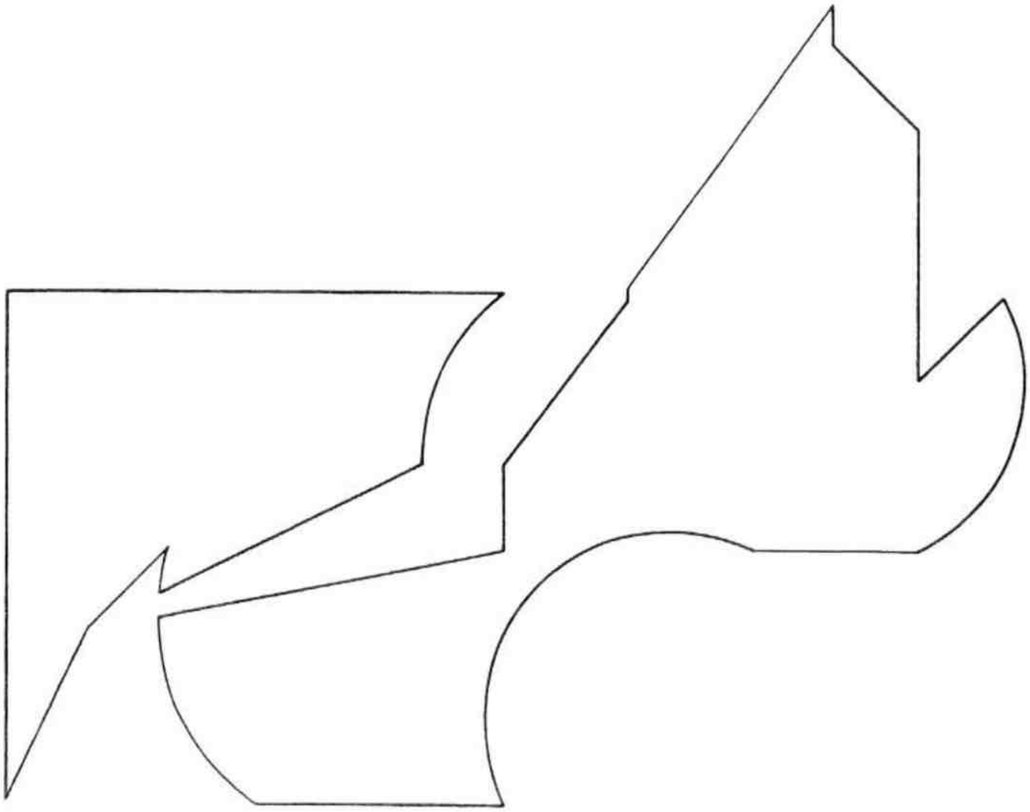


Fig. 4—Two regions of overlap, "A and B."

common blocks must be provided for the summary table described above, for a summary table (of different form) which controls the data structure in which the input figures are located, and for the error margin quantity discussed in Section III. The common block containing the summary table has three additional locations. One reports back the total number of figures generated; another carries a code for the mode of intersection (no contact, tangencies only, coincident figures, or normal intersection). The third is set on input as the total number of locations available for output figure descriptions and returns as a pointer to the first unfilled location. An error return occurs if the output data structure tries to overflow the space allocated for it.

Figures 1 through 6 are examples of the operation of this program. Figure 1 shows the two input figures. The first has 16 sides, 12 of them lines and the other 4 arc segments; the second has 15 sides, 11 of them lines and the other 4 arcs. Figure 2 shows the graph resulting from their superposition. There are 15 points of intersection, and every

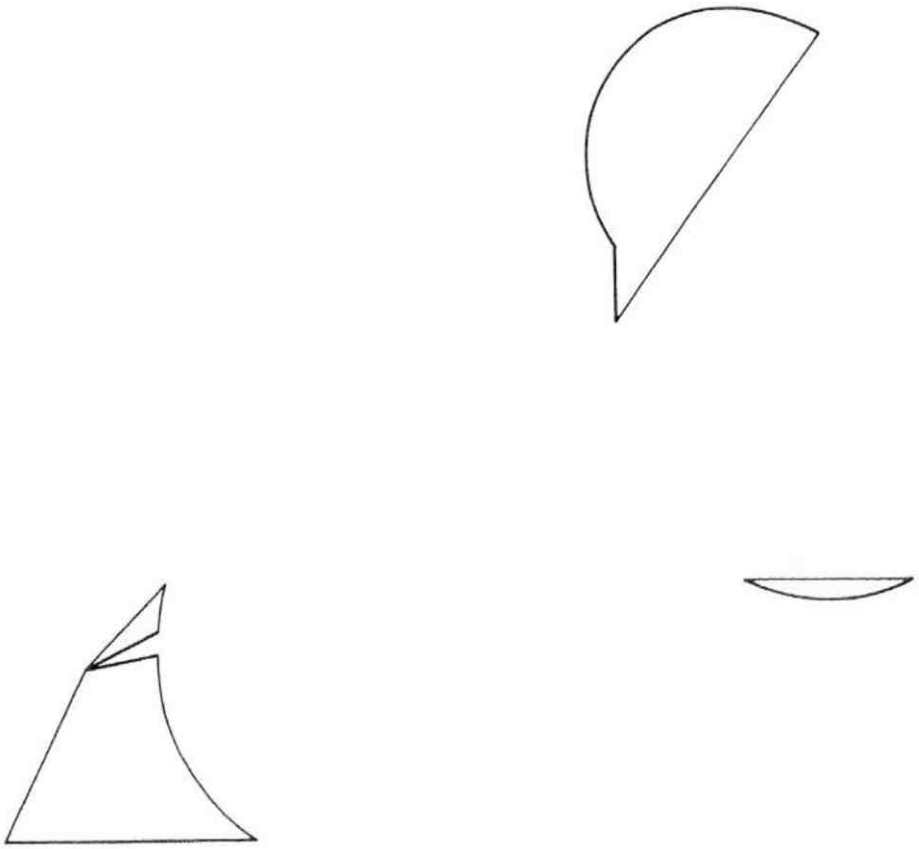


Fig. 5—Four regions of A not covered by B—A and not-B.

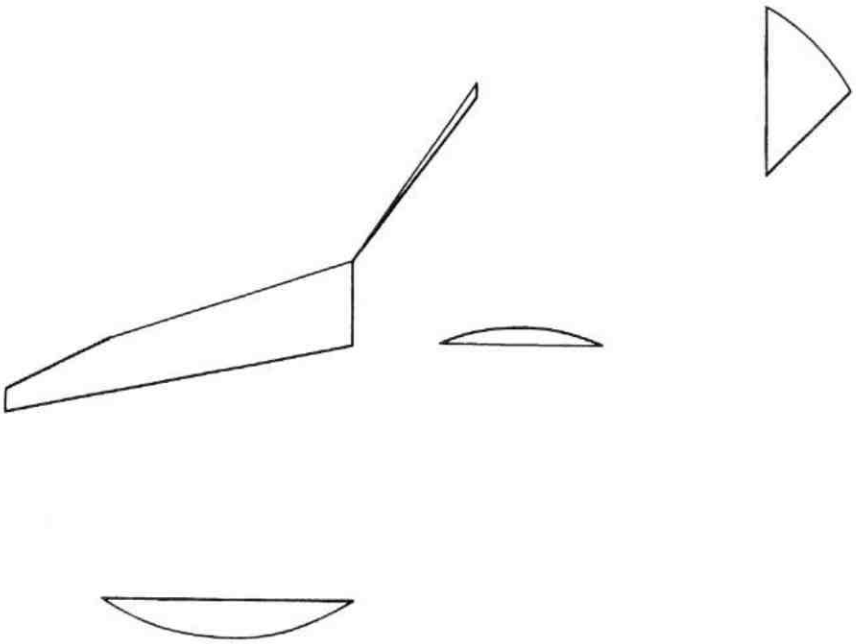


Fig. 6—Five regions of B not covered by A—B and not-A.

type is represented: point crossings, point tangencies, colinear crossings, and a colinear tangency. Thirteen intersection figures, including the outer boundary of the pair, are generated. Figure 3 shows the figures classified as "A or B" (the union of the two input figures). Note that there is an inner figure, or hole, which can also be described as bounding a region "not-A and not-B". Figure 4 shows two regions of overlap, "A and B" (the intersection, in the logical sense, of the two figures). Figure 5 shows four regions of A not covered by B (A and not-B) and Fig. 6 shows five regions of B not covered by A (B and not-A).

Execution of this analysis by the OVRLAP subroutine required 0.75 second of processor time on the GE 635, the memory cycle of which is one microsecond. This does not include time spent on physical input, generation of the input data structure, or physical output.

OVRLAP and all its required subroutines and common blocks, exclusive of system library subroutines, occupies about 11000 (decimal) words of storage.

VI. SUMMARY

A computer program has been developed which, given two polygon-like figures each consisting of an arbitrary number of line segments or circular arc segments, generates every intersection figure resulting from their superposition and identifies its logical relationship with the input figures. The two input figures may intersect or be tangent at any number of points and may have sections of coincident boundaries. Each input figure is described as a sequence of bounded analytical expressions and the output figures are generated in the same numerical format. No grid approximation is used; the program operates by analytically finding every point of intersection or tangency of the figures; these points are the nodes of a graph through which the program traces according to an algorithm which causes every intersection figure be traced just once. A FORTRAN IV implementation of this program took 0.75 second on a GE 635 computer to resolve the superposition of two figures, of 15 and 16 sides respectively, into 13 resultant figures.

Copyright © 1972 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 7, September, 1972
Printed in U.S.A.

Dynamic Channel Assignment in Two-Dimensional Large-Scale Mobile Radio Systems

By D. C. COX and D. O. REUDINK

(Manuscript received April 4, 1972)

A computer simulation of two-dimensional mobile radio systems arranged with square coverage areas on a square grid and using dynamic channel assignment techniques is described. Parameters for the simulation are: (i) 729 distinct coverage areas (27 on a side), (ii) 160 radio channels and (iii) a radio channel reuse interval of every fourth coverage area. Three different channel assignment strategies are considered and the results are compared to previous one-dimensional simulated systems and to a fixed channel assignment system. At call blocking rates below about 10 percent, the two-dimensional dynamic channel assignment systems carry more traffic and produce fewer forced call terminations at coverage cell boundaries than do fixed channel assignment systems. For example, at a blocking rate of 1 percent, the traffic carried, TC, expressed in Erlangs per channel per coverage area by the various systems are as follows: fixed channel assignment systems, TC = 0.44, one-dimensional dynamic channel assignment system, TC = 0.62, two-dimensional dynamic channel assignment system, TC = 0.63.

I. INTRODUCTION

When a large pool of radio channels is available for use in a large-scale multiple base station mobile radio system, its assignment to individual base stations on the basis of instantaneous demand can improve systems performance.[1-3] Previous studies of coordinated dynamic channel assignment in one-dimensional systems (that may be used along an expressway or major air route for example) have shown that at blocking rates below 10 percent these assignment procedures produce increased channel occupancy over systems using the same channels but with channel allocation to each base station fixed.

The most general form of dynamic channel assignment assumes that any channel can be used at any base station. Efficient use of channels requires the simultaneous assignment (reuse) of the radio channel in radio coverage areas which are spaced as close together as possible without incurring excessive cochannel interference. Mobile radio systems which reuse channels within a metropolitan area often are referred to as small-cell or small coverage area systems.[3-6]

The analysis of dynamic channel assignment systems which employ complex channel assignment algorithms appears to be intractable. For this reason, the performance of these systems is most readily determined by large-scale computer simulation. The overall system performance of two-dimensional dynamic channel assignment mobile radio systems is expected to be similar to that of the one-dimensional systems previously studied.[1,2] However, the performance of the one-dimensional systems was shown to depend upon the total number of channels available to the radio system as well as the average number of channels available in each coverage area. Thus, since the channel reuse situation is more complex for two-dimensional systems, some difference in the performance of the two-dimensional systems was expected.

This paper describes a computer simulation of a large-scale two-dimensional mobile radio system using different dynamic channel assignment strategies and compares the performance characteristics of these systems with both a one-dimensional dynamic channel assignment system and with a fixed channel assignment system.

II. THE SIMULATION MODEL

2.1 *The Model of the System Configuration*

The system simulated consists of a set of square radio coverage areas arranged to completely cover a square grid with no overlap as illustrated in Fig. 1. For this model, a channel used in a specified coverage area, such as the shaded area at coordinates m, n may be reused anywhere on or outside of a specified square ring which surrounds that particular coverage area. This reuse ring is defined by those coverage areas which have either an x or y coordinate which is a specified integral number of coverage areas separated from the center coverage area. This specified interval of coverage areas is referred to as the channel reuse interval. In the example in Fig. 1, the cross-hatched coverage areas on the reuse ring surrounding the coverage area at $x = m$, $y = n$ are at a reuse interval of 4. Several coverage areas on the reuse ring may also use the same channel as long as all coverage areas are separated by at least

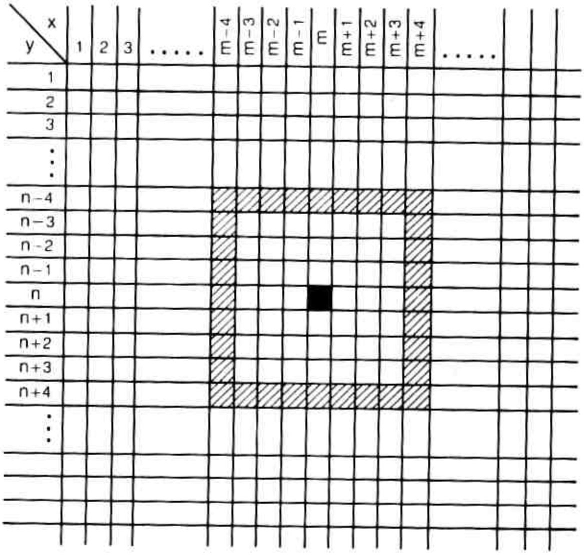


Fig. 1—Coverage area configuration for the simulation.

the reuse interval. Thus, in the example, the maximum number which can use the same channel simultaneously on the reuse ring is eight. For the simulation the coverage squares were specified to be 2 miles on a side.

The "local" mean (average over several wavelengths) signal strength in an urban area measured on a circle of constant radius from an omnidirectional base station has a log normal distribution.[7,8] The variance of the log normal distribution is sufficiently large ($\sigma = 5$ to 9 dB) so that circular coverage areas are seldom realized. Since traffic handling capabilities of dynamic channel assignment systems are the primary concern of this study, it is desirable to define geometric coverage areas which completely cover a region with no overlap. The only candidates are triangles, rectangles, and hexagons. Of these, the geometry associated with squares is simplest to manipulate. Since the propagation is highly irregular, radio channels obviously may not be used simultaneously in immediately adjacent coverage areas regardless of the geometry defining the coverage area. Thus, square coverage areas on a square grid with the reuse intervals described previously provide a reasonable model for these traffic studies. Such a system could be approximated by placing base stations at the corners of the coverage areas and radiating into them with directional antennas.

2.2 *The Stimulus to the Simulated System*

The initiation of call attempts and the movement of vehicles in this simulation is almost identical with that described previously[1] for a one-dimensional simulation. Call attempts are generated in the simulation as a Poisson process in time which has a controllable average attempt rate in each coverage area. For this study, initial call attempts from vehicles are uniformly distributed in both the x and y coordinates and the x and y locations are independent. Call attempts are tapered at the edges of the simulated universe, and the performance data accumulated only from the central portion of the system, so that the results are representative of an infinite system.

In this method of simulation, vehicles making calls are identifiable entities the locations and movements of which are stored in the computer. Vehicle velocities (speeds and direction) are chosen randomly from a population having statistical characteristics which can be prescribed. Vehicle motions are primarily parallel to one of the coordinate axes with 45 percent of the vehicles having an x velocity component only, and 45 percent a y component only. Ten percent of the vehicles have both x and y velocity components which are mutually independent. The velocity components (x and y) have a truncated Gaussian distribution with a zero mean, a standard deviation of 30 miles per hour, and a maximum velocity magnitude of 60 miles/hour (see Figure 8 of Ref. 1).

Call attempts that are assigned a channel remain on in the system for call durations which are taken from a random population with a specified distribution. Part of the data to be presented later is for a call duration distribution which is exponential with a mean of about 98 seconds. Other data are for a call duration distribution which is a truncated Gaussian with a minimum call duration of 30 seconds, a maximum call duration of 10 minutes, and a true mean of 103.5 seconds. The mode of this distribution is 90 seconds (see Figure 7 of Ref. 1). Some of the vehicles which cross coverage area boundaries have their calls prematurely terminated by the system because of the unavailability of channels in the new coverage area. This effect shortens some calls and thus perturbs the specified distribution slightly. This is discussed in more detail later.

2.3 *The Simulated Operating Systems*

The method of handling call attempts, calls in progress and call terminations is similar to that described in Ref. 1 for a one-dimensional system. The first step in processing a new call attempt is to assign a coverage area. Since coverage areas are geometrically defined and the vehicle coordinates known, the appropriate coverage area is readily determined. In the simulation, vehicles within a particular coverage area are required to be served by channels available in that coverage area. The next step is a channel search which tries to find a radio channel to serve the awaiting call attempt. Some channel search procedures are discussed in detail later. If no channel can be found, the call is immediately blocked and cleared from the system. This blocking strategy is the same as that used in deriving the Erlang B (block call cleared) telephone traffic formula.[9]

The simulated system detects vehicles crossing coverage area boundaries and checks to see whether the call can continue on the original channel or on a new channel. If no channel is available, the call is immediately forced to terminate and is cleared from the system. In an actual system, the crossing of coverage area boundaries will not be critical and, hence, some delay could be tolerated in finding a new channel for boundary crossing vehicles. When a significant number of the calls handled by the system experience a boundary crossing, the calls forced to terminate have an effect on other system performance parameters. With the coverage area size and the velocities used in this simulation the effect of boundary crossings on traffic carried and call blocking is small.

2.4 *Channel Assignment Strategies*

Information which identifies the coverage area and the channel being used by each active subscriber in the network is stored in the memory of the simulated system control computer. In addition, a list is kept of every channel being used in each coverage area. The second list, although redundant, is in a form which permits rapid access to the information needed for the dynamic channel assignment procedures. A channel search is initiated after the coverage area of a call attempt has been determined.

The procedures used in the channel assignment strategies are best described in terms of an example which can be referred to Fig. 1. Assume that a call attempt located at coverage area $x = m$; $y = n$ is awaiting channel assignment. The first step is to determine which channels (if any) are available to serve the call attempt. This is accomplished by compiling a list of the channels which are not being used currently within any coverage area surrounding the designated coverage area out to but not including the reuse ring defined previously. If more than one radio channel is available within this reuse interval, then some channel assignment strategy must be applied to determine the channel that should be assigned. The main object of a channel assignment strategy is to increase the channel occupancy or to optimize some other system parameter. Briefly, the two-dimensional channel assignment strategies considered here are as follows:

First Available

This strategy assigns the first available radio channel encountered during a channel search. This strategy obviously minimizes the system computation time and as shown later its performance may be adequate for some applications.

Selection of a Channel With Maximum Usage on the Reuse Ring (RING)

A search is made through the list of available channels to determine which of these channels is currently in use in the most coverage areas lying on the reuse ring. If more than one channel has this maximum usage, an arbitrary selection of one of the channels is made to serve the call attempt. If none of the available channels is in use on the reuse ring (an infrequent event as illustrated in Fig. 8), then selection is made on a first available basis. This strategy is described in detail in Ref. 10.

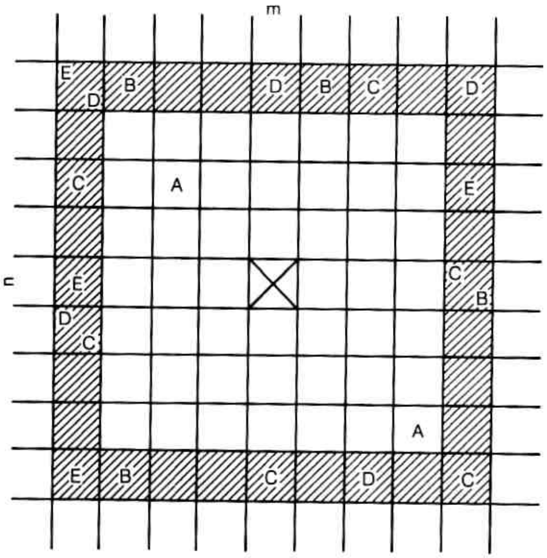
Selection of a Channel With Maximum Usage on a Side(s) of the Reuse Ring (Orthogonal NN)

A search is made through the list of available channels to determine which channel(s) meets the following criteria:

(*i*) On some side of the square reuse ring channel usage is a maximum.
(*ii*) Of those channels having a maximum usage on some side, channel usage is again a maximum on a side adjacent (orthogonal) to that first maximum usage side.

If none of the available channels are in use on the reuse ring, channel assignment is made on a first available basis. If only one channel has a maximum channel usage on some side, that channel is assigned without regard to the usage on other sides. If more than one channel has maximum usage on some side and equal secondary maximum usage on an adjacent side, one of these channels is selected at random. For this simulation optimization was not carried to the third and fourth sides. This strategy is one possible extension of the best performing one-dimensional channel assignment strategy[2] previously described.

Figure 2 illustrates some of the differences in these channel assignment strategies. In this figure, assume that channels are designated by the



STRATEGY	CHANNEL ASSIGNED
1ST AVAIL	B
RING	C
⊥ NN	D

Fig. 2—A channel assignment example.

letters A through E. Channel A is not available for assignment because it is being used in coverage areas within the reuse interval. For channel selection by the RING strategy, channel B is used four times on the reuse ring, C six times, D five times, and E four times. Thus the RING strategy selects channel C for assignment.

For the orthogonal NN strategy the maximum side usage is three, and occurs for channel D on the upper side and channel E on the left side. Channel usage on a corner is included in the usage count for each side which includes that corner. For example, the channel D usage on the upper left-hand corner of Fig. 2 results in a channel usage count of three for the upper side and a count of two for the left side. Thus, the adjacent side usages for channel D are two on the left and one on the right. The corresponding adjacent side usages for channel E are only one each on the top and bottom. Therefore, channel D has the maximum adjacent side channel usage and is thus selected for assignment by the orthogonal NN strategy.

If the channel assignment search is assumed to proceed in alphabetical order, the first available strategy would select channel B for assignment.

2.5 *System Parameters*

The simulated operating systems were run for a 27 by 27 grid of coverage areas which were 2 miles on a side. The systems had available

160 duplex radio channels. With the assumed reuse interval of 4, this results in 10 channels (160/16) available on the average at each base station. Of course, depending upon the system activity and the instantaneous demand, some coverage areas will have more than 10 channels in use simultaneously and some will have fewer. Because of the uniform spatial and temporal statistics of the call-attempt process, averages of system parameters taken over a large number of coverage areas at a particular time must yield the same results as averages taken over an extended time interval for one coverage area. Thus, the stability of the statistics for performance parameters depends upon the product of the number of coverage areas included and the run time. Statistics were taken from only the central 225 coverage areas (15 by 15) to avoid any effects from the edges of the finite system. Call loading and system response at the edges of the finite system were tapered in a way which insured that these central areas were operating as members of an infinite set. The simulation was started initially with no calls on in the system and statistics of parameters were monitored until the initial transients died out. The time required for stability was about 2 minutes. Data were collected after stabilization for about 7 minutes (in simulated time). This run time and spatial size was sufficient produced data points which lie on smooth curves and have very little scatter.

The data presented in the following sections were obtained by counting and storing the number of actual events which occurred in the stabilized system. Some of the events counted were new call-attempts, new call-attempts blocked, coverage boundary crossings and calls forced to terminate after crossing coverage boundaries. In addition, the actual number of calls in progress in the system was counted and stored periodically during each simulation run.

III. COMPARISONS OF SYSTEMS PERFORMANCE

The computer simulation was run at several call attempt rates for the channel assignment strategies and system parameters described in Section II. The data from the simulation, which show directly the relationship between channel occupancy (traffic carried per channel per coverage area) and the blocking of new call attempts, are plotted in Fig. 3. The solid data points are for the exponential call duration distribution and the open data points are for the truncated Gaussian call duration distribution described in Section 2.2. The triangles and circles are for the two-dimensional systems with a reuse interval of four. The solid curve is drawn through the data points for both the RING and the orthogonal NN two-dimensional channel assignment

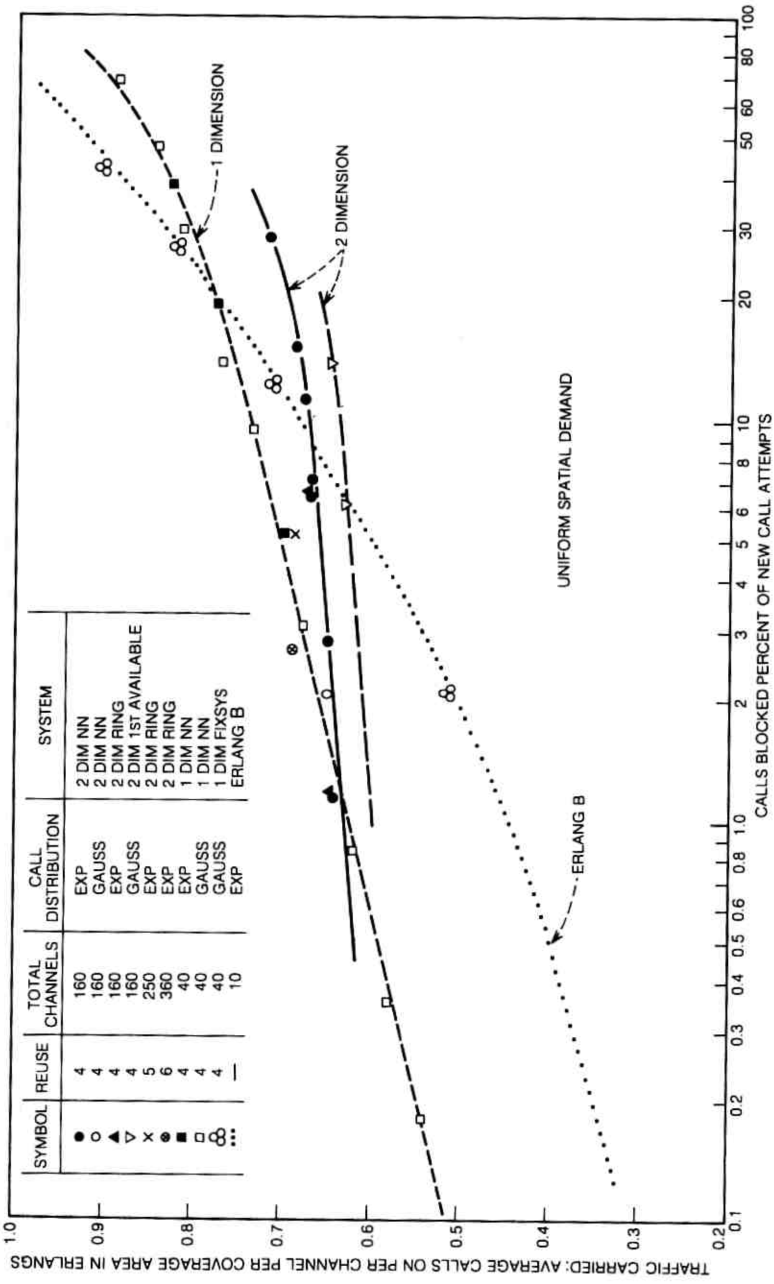


Fig. 3—Normalized channel occupancy.

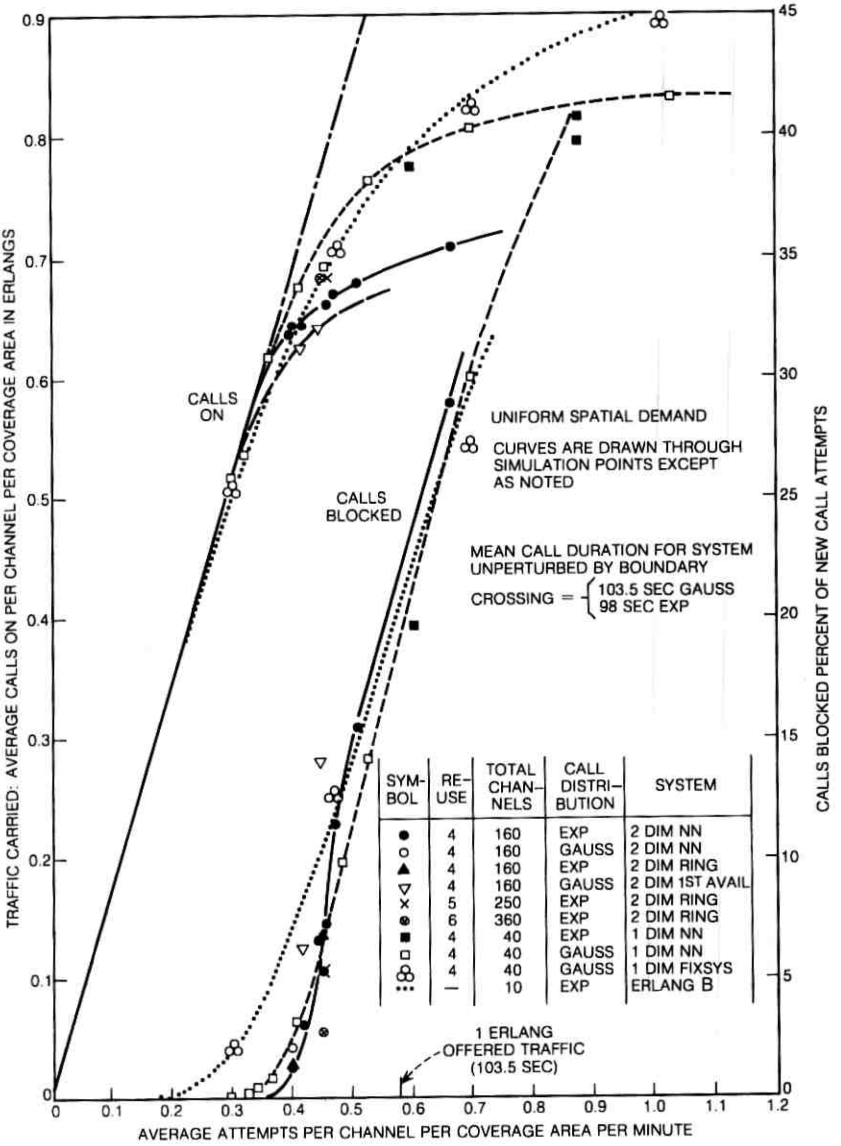
strategies. The performance of these two strategies in terms of these parameters is nearly identical, as indicated by the fact that the solid triangles at 1.1 percent and 6.6 percent blocking lie nearly on top of the solid circles. All of the points for these two strategies fall on the smooth curve over the blocking range indicated (1 through 30 percent). In fixed channel assignment systems for which the Erlang B telephone traffic formula is applicable, it has been shown[11] that the shape of the call duration (holding time) distribution has no effect on the relationship between traffic carried and the blocking of new call attempts. The open circle at 2 percent blocking that lies on the solid curve is for a different call duration distribution than the solid points as noted in the key. This indicates that for these dynamic channel assignment strategies the performance parameters are also not affected by the shape of the call duration distribution. The inverted triangles connected by the long dashed curve are for the two-dimensional first available channel assignment strategy. This strategy results in slightly lower channel occupancy for a given blocking. It is much simpler to implement, however, and may be acceptable in some applications.

The short dashed curve through the open square data points is from a one-dimensional channel assignment system with a reuse interval of 4, as described in detail in Ref. 2 (NN strategy). The solid squares are for the same one-dimensional channel assignment strategy, but for an exponential call duration distribution. The fact that the points lie on the curve for the truncated Gaussian call duration distribution further illustrates the independence of the performance of dynamic channel assignment to the type of call duration distribution. Over a wide blocking range, the traffic carried by the two-dimensional system is more nearly constant than for the one-dimensional system. We would expect that at very low blocking rates the two-dimensional system would perform better (carry more traffic at a given blocking rate) because it has more channels available to the system (160 channels) than the one-dimensional system (40 channels) even though the average number of channels available per coverage area is the same. However, as seen from Fig. 3, the two-dimensional system does not perform as well as the one-dimensional system above a blocking rate of about 1 percent. This is probably due to the fact that the simultaneous use of the same channel in coverage areas separated by exactly a reuse interval is more difficult to achieve in the two-dimensional area situation than on a one-dimensional line. To further check the performance of the simulations, the two-dimensional orthogonal NN strategy system was run as a one-dimensional system by restricting the location of call attempts and vehicle motion to one axis. The resulting data points for this simulation were located along the performance curves from the previous one-dimensional simulation.

The four data points denoted as ∴ are from a simulation of a one-dimensional fixed channel assignment system.[1,2] In this system, specific channels are allocated to each coverage area and the same channels are reallocated at coverage areas separated by exactly a reuse interval. For the simulated fixed channel assignment system, 10 channels were allocated to each coverage area and the reuse interval was 4. The performance of the fixed channel assignment system can be determined directly from the Erlang B telephone traffic equations.[3,9] The relationship between the traffic carried and call blocking from the Erlang B formula is the dotted line in Fig. 3. This curve represents the performance of fixed channel assignment in both one and two dimensions since the curve depends only on the number of channels available per coverage area. The fixed channel assignment system performs worse at low blocking than any of the dynamic channel assignment systems because it is less able to meet the peaking of call attempts in the randomly offered traffic. At high blocking rates, fixed channel assignment systems handle more traffic because channel reuse is fixed in an optimum configuration.

The two-dimensional simulation using the RING strategy was also run for a reuse interval of five with 250 channels, and for a reuse interval of six with 360 channels. The resulting data points are indicated by the X and ⊗ in Fig. 3. As was found in the one-dimensional case,[2] there was a small increase in the traffic carried at a specified blocking as the reuse interval increased even though the average number of channels available per coverage area was held constant. The factors affecting this increase in traffic carried are discussed in Ref. 2.

Another measure of system performance is the behavior of the blocking and traffic-carried parameters plotted as functions of the new call-attempt rate as shown in Fig. 4. The symbols and the types of lines denote the same system parameters and channel assignment strategies as they did in Fig. 3. At low attempt rates, no new call-attempts are blocked in any of the systems and the traffic carried (average calls on per channel per coverage area) is a linear function of the new call-attempt rate. As the attempt rate increases, the fixed channel assignment system begins to block calls before any of the dynamic channel assignment systems. The two-dimensional systems are the last to experience significant (greater than 1 percent) blocking.



UNIFORM SPATIAL DEMAND

CURVES ARE DRAWN THROUGH SIMULATION POINTS EXCEPT ꝏ AS NOTED

MEAN CALL DURATION FOR SYSTEM UNPERTURBED BY BOUNDARY CROSSING = { 103.5 SEC GAUSS
98 SEC EXP

SYMBOL	REUSE	TOTAL CHANNELS	CALL DISTRIBUTION	SYSTEM
●	4	160	EXP	2 DIM NN
○	4	160	GAUSS	2 DIM NN
▲	4	160	EXP	2 DIM RING
▽	4	160	GAUSS	2 DIM 1ST AVAIL
×	5	250	EXP	2 DIM RING
⊗	6	360	EXP	2 DIM RING
■	4	40	EXP	1 DIM NN
□	4	40	GAUSS	1 DIM NN
ꝏ	4	40	GAUSS	1 DIM FIXSYS
•••	—	10	EXP	ERLANG B

Fig. 4—Comparative performance of mobile radio systems.

The curves labeled "calls on" illustrate that at high attempt rates the fixed channel assignment systems carry more traffic than the dynamic channel assignment systems.

With no vehicle motion, the loading or traffic offered to these systems is the product of the new call-attempt rate and the mean call duration determined by the stimulus part of the simulation. One Erlang of offered traffic (one channel occupied 100 percent of the time) for the system unperturbed by motion is indicated on the call attempt axis. When vehicles cross coverage area boundaries and do not find channels available for continuing their calls, their premature termination reduces the actual average call duration. The fact that these systems must attempt to find new channels for some of these boundary crossing calls produces additional loading on the system. The actual average call duration of the exponential call duration distribution (98 seconds) is

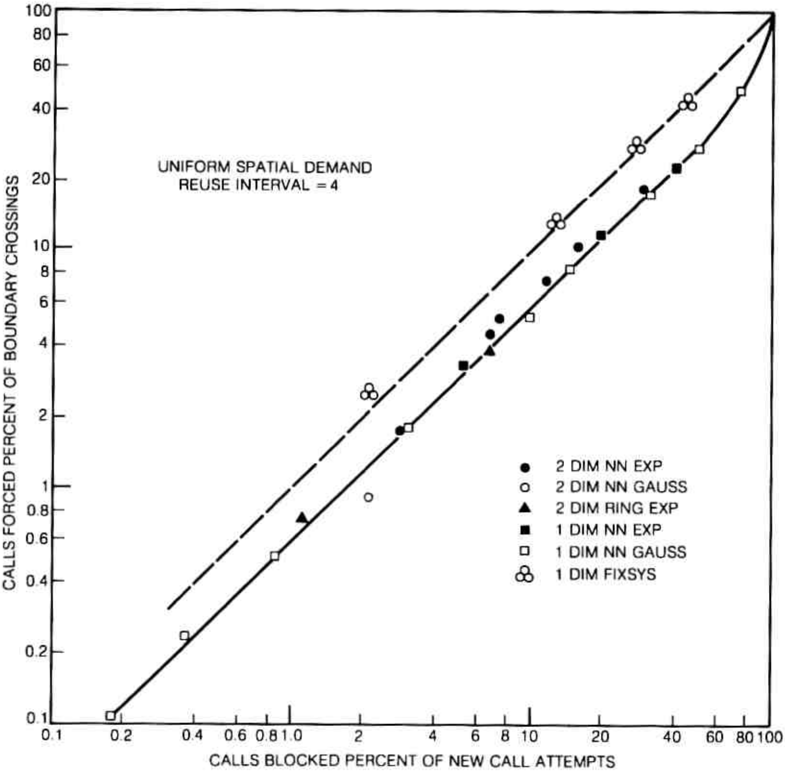


Fig. 5—Forced call terminations relative to boundary crossings.

less than that for the truncated Gaussian distribution (103.5 seconds). This difference in call duration causes the actual loading on the systems to be different at a given new call-attempt rate and accounts for the displacement of the data points for the exponential call duration distribution from those of the truncated Gaussian in Fig. 4. This effect is not seen in Fig. 3 because the traffic carried contains the actual call durations. The small difference in call durations resulted from different quantization effects in the methods used to generate samples from the two distributions.

Since some of the boundary crossing calls in the dynamic channel assignment systems keep their originally assigned channels, the forced call termination rate is less than the blocking rate in these systems. In the fixed channel assignment system, all boundary crossing calls require a new channel, and thus the forced termination rate is the same as the blocking rate. In Fig. 5, the forced call termination rate expressed as the ratio of the number of calls forced to terminate in a time period to the total number of active call boundary crossings in that time period is plotted vs the blocking rate of new call-attempts. This function appears to be the same for all of the dynamic channel assignment systems included in Fig. 5.

In Fig. 6, the ratio of the number of calls keeping the same channel

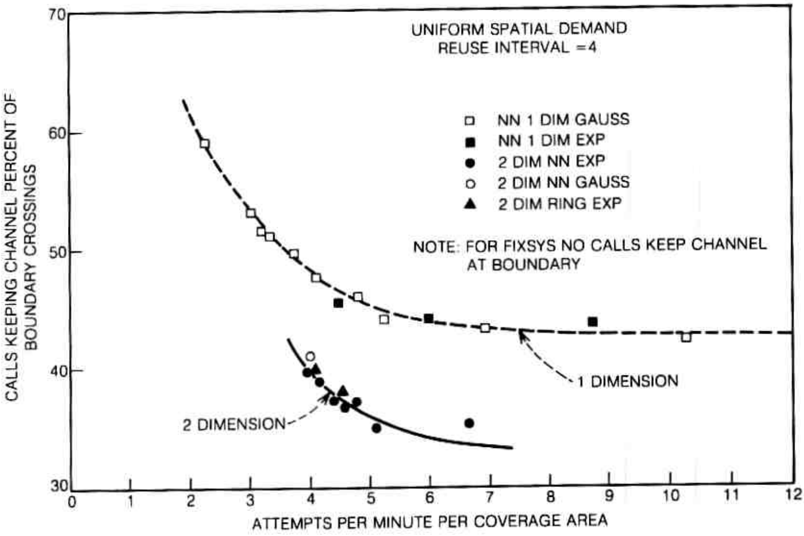


Fig. 6—Calls keeping channel at boundary.

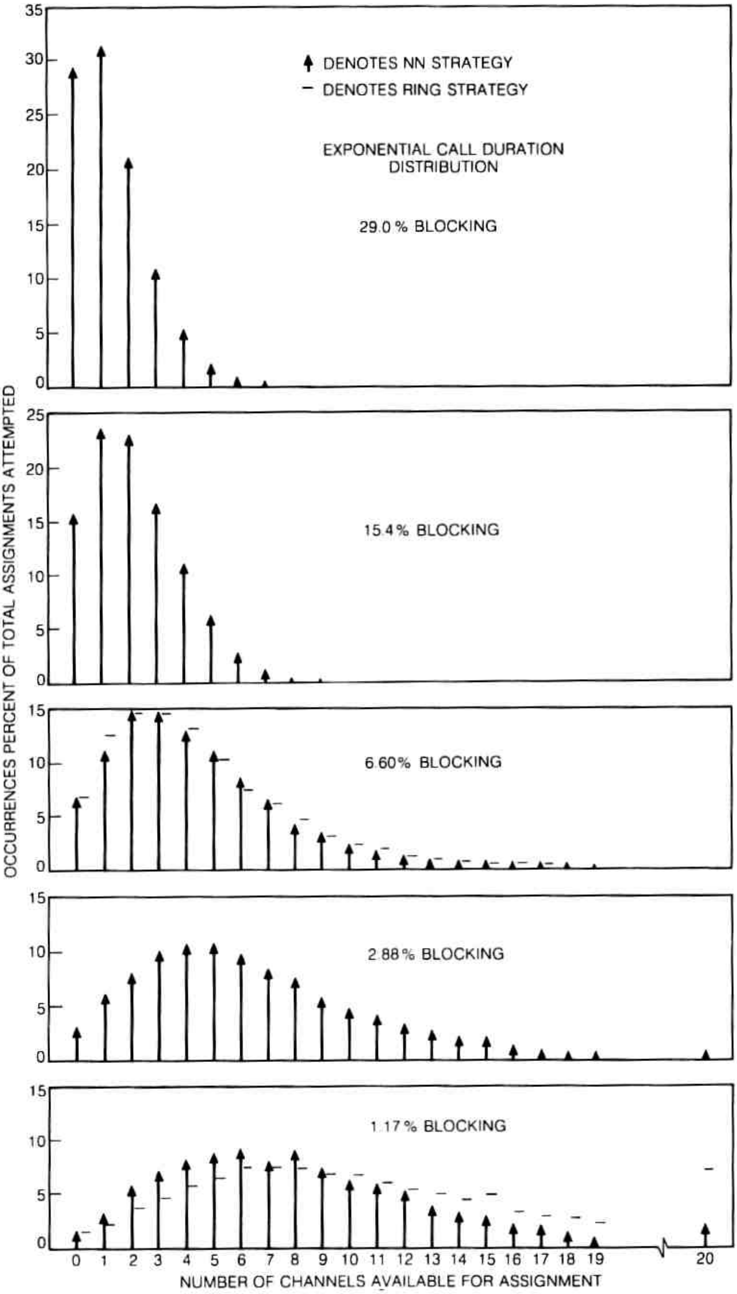


Fig. 7—Channel availability two-dimensional systems.

when crossing a coverage boundary to the total number of boundary crossings is plotted as a function of the new call-attempt rate. This relationship indicates the extent to which the originally assigned radio channel "floats" with the vehicle. The behavior of the two-dimensional RING and orthogonal NN strategy are similar for these parameters also. Fewer callers keep their originally assigned channels when crossing boundaries in the two-dimensional systems than do in the one-dimensional systems.

IV. OPERATING CHARACTERISTICS OF THE TWO-DIMENSIONAL DYNAMIC CHANNEL ASSIGNMENT SYSTEMS

More than one channel must be available for assignment in a particular coverage area if a channel assignment strategy is to improve system performance. Figure 7 shows the number of channels available for assignment expressed as the percentage of total channel assignments (new call attempts plus boundary crossing calls having channels reassigned) attempted for several blocking rates. At low blocking rates, several channels are available to select from for most of the channel assignments attempted. As the loading on the system increases, fewer channels are available from which to choose. Thus, the effectiveness of the channel assignment strategies decreases as the system loading increases. Although slight differences exist in the histograms for the different channel assignment strategies, the overall availability of channels for each assignment strategy is about the same.

4.1 *RING Strategy*

The two-dimensional RING strategy attempts to maximize channel usage on the reuse ring. Figure 8 illustrates the degree to which this is achieved for two blocking rates. The percentage of total channel assignments made (new call attempts plus boundary crossings having channels reassigned) is given for each possible number of simultaneous channel usages on the reuse ring. The histograms of Fig. 8 show that this dynamic channel assignment strategy is not able to maximize channel reuse as it serves the randomly-offered call attempts. It should be possible to reassign channels to calls in progress so that the channel reuse is more concentrated and thus the average channel occupancy is increased for a given blocking rate.

The number of occurrences of channel assignments with each possible number of simultaneous channel usages on the reuse ring is expressed

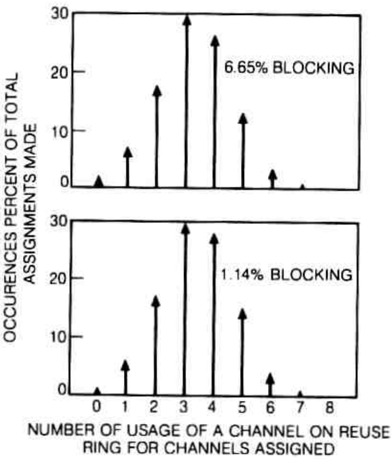


Fig. 8—Channel usage on reuse ring for two-dimensional ring strategy.

as the percentage of the total number of channel assignments made. For example, at a 6.65 percent blocking rate, the channel assigned was in use in three coverage areas on the reuse ring for about 30 percent of the total channel assignments made. It is interesting to note that the assigned channel was being used in the maximum possible number (8) of coverage areas on the reuse ring in fewer than 0.1 percent of the assignments made.

4.2 *Orthogonal NN Strategy*

Some characteristics of the behavior of the two-dimensional orthogonal NN channel assignment strategy are shown in Fig. 9 as a function of the blocking rate of new call-attempts. The curve through the open squares shows that for very few of the channel assignments attempted there was no usage of the channel assigned in any coverage area on the reuse ring. The curve through the open triangles indicates that channel assignment was made on the basis of the maximum usage on only one side of the reuse ring for 40 to 50 percent of the assignments attempted. Below a 10 percent blocking rate, about 50 percent of the channel assignments attempted benefitted from orthogonal side optimization as indicated by the curves through the open circles and the X's. The curve through the X's indicates that up to about 25 percent of the assignments attempted could have benefitted from usage maximization on the third side.

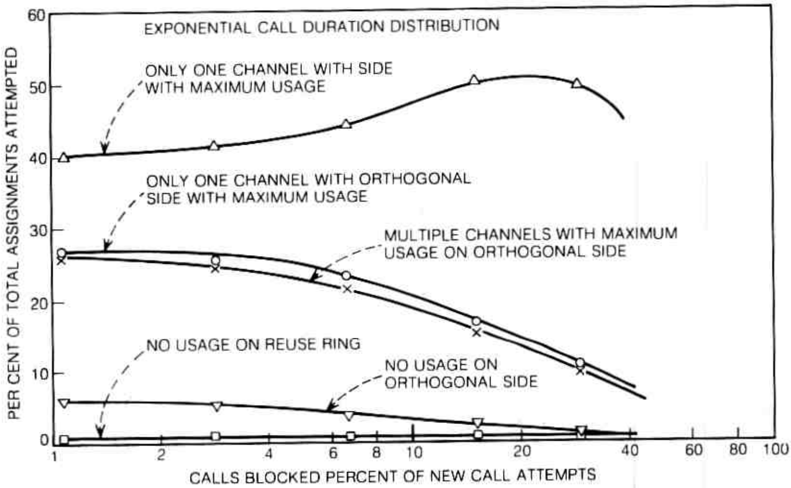


Fig. 9—Performance characteristics of two-dimensional NN channel assignment.

V. CONCLUSIONS

The results from the simulation of the two-dimensional dynamic channel assignment mobile radio systems show that these systems operate at very low blocking until the traffic offered reaches some critical value. Small increases in loading above this value produce a considerable increase in the blocking of new call attempts and result in very little increase in the traffic carried by the system. The loading at which blocking begins to occur in the two-dimensional systems is somewhat greater than the loading at which blocking begins to occur in one-dimensional dynamic channel assignment systems, and is considerably greater than the loading at which a fixed channel assignment system begins to incur significant blocking. For example, at a blocking rate, B of 1 percent, the traffic carried, TC, in Erlangs per channel per coverage area and the traffic offered, TO, in Erlangs per channel per coverage area (TO = TC/(1 − B)) for the systems studied which had an average of 10 channels available per coverage area and a reuse interval of 4 are: two-dimensional dynamic channel assignment TO = 0.636, TC = 0.630; one-dimensional dynamic channel assignment systems, TO = 0.631, TC = 0.625; fixed channel assignment systems, TO = 0.444, TC = 0.440.

It was found that the two strategies aimed at channel reuse optimization performed similarly and that both were somewhat better (carrying approximately 5 percent more traffic at a given blocking rate) than a channel assignment strategy which chose the first available channel encountered in a channel search without any regard to reuse optimization. The reason that these dynamic channel assignment systems do not carry as much traffic at high blocking rates as fixed channel assignment systems is that they are not able to maximize channel reuse as they serve the randomly-offered call attempts. It should be possible to reassign channels to calls in progress so that the channel reuse is more concentrated and thus more traffic could be carried per channel at a given blocking rate.

VI. ACKNOWLEDGMENTS

The authors wish to thank Mrs. Wanda Mammel and Mrs. Diane Vitello for their help in debugging and running the computer programs.

REFERENCES

1. Cox, D. C., and Reudink, D. O., "Dynamic Channel Assignment in High Capacity Mobile Communications Systems," B.S.T.J., *50*, No. 7 (July–August 1971), pp. 1833–1857.
2. Cox, D. C., and Reudink, D. O., "A Comparison of Some Channel Assignment Strategies in High Capacity Mobile Radio Systems," IEEE Trans. Commun. Tech. (April 1, 1972).
3. Schiff, L., "Traffic Capacity of Three Types of Common User Mobile Radio Communication Systems," IEEE Trans. Commun. Tech., *COM-18*, No. 1 (February 1970), pp. 12–21.
4. Frenkiel, R. H., "A High Capacity Mobile Radiotelephone System Model Using a Coordinated Small-Zone Approach," IEEE Trans. Veh. Tech., *VT-19*, No. 2 (May 1970), pp. 173–177.
5. Schulte, H. J., Jr., and Cornell, W. A., "Multiarea Mobile Telephone System," IRE Trans. Veh. Commun., *9*, 1960, pp. 49–53.
6. Araki, K., "Fundamental Problems of Nationwide Mobile Radio Telephone System," Rev. Elec. Comm. Lab., Japan, *16*, No. 5–6 (May–June 1968), pp. 357–373.
7. Okurmura, Y., Ohmori, E., Kawano, T., and Fukuda, K., "Field Strength and Its Variability in VHF and UHF Land-Mobile Radio Service," Rev. Elec. Commun. Lab., Japan, *16*, No. 9–10 (September–October 1968), pp. 825–873.
8. Black, D. M., and Reudink, D. O., "Some Characteristics of Mobile Radio Propagation at 836 MHz in the Philadelphia Area," to be published in IEEE Trans. Veh. Tech.
9. ITT Handbook, *Reference Data for Radio Engineer*, Fifth Edition, First Printing, October 1968.
10. Cox, D. C., and Reudink, D. O., "Dynamic Channel Assignment in Multidimensional Mobile Communication Systems," U. S. Patent Office Application BTL ID Cox-Reudink 2–4.
11. Kosten, L., "On the Validity of the Erlang and Engset Loss-Formulae," P-T Bedrijs, *2*, No. 1, 1948–49.

Contributors to This Issue

Syed V. Ahamed, B.E., 1957, University of Mysore, India; M.E., 1958, Indian Institute of Science; Ph.D., 1962, University of Manchester, U. K.; Post Doctoral Research Fellow, 1963, University of Delaware; Assistant Professor, 1964, University of Colorado; Bell Laboratories, 1966—. Mr. Ahamed worked in Computer Aided Engineering Analysis and Software Design at Whippany. He has applied algebraic analysis to the design of domain circuits. Presently, he is investigating computer aids to the design of bubble circuits.

Raymond H. Bosworth, Air Force Radio Technical Institute, 1950; Union Junior College, 1952–1956; R.C.A. Institutes, 1962–1964; Bell Laboratories, 1952—. Mr. Bosworth has worked on negative impedance repeaters, repertory dial telephones, and PCM coding techniques. He holds patents in printed circuit technology.

L. H. Brandenburg, B.S., 1962, M.S., 1963, Ph.D., 1968, Columbia University; Bell Laboratories, 1968—. Mr. Brandenburg has worked on various analytical problems associated with communication theory. Member, IEEE, Sigma Xi, Tau Beta Pi.

J. C. Candy, B.Sc., 1951, Ph.D., 1954, University of Wales, Bangor; British Atomic Energy Authority, 1956–59; Research Associate, University of Minnesota, 1959–60; Bell Laboratories, 1960—. Mr. Candy has worked on digital circuits and pulse transmission systems. He is studying methods for processing video signals, and designing digital coders. Member, IEEE.

Donald C. Cox, B.S. (E.E.), 1959, M.S. (E.E.), 1960, University of Nebraska; Ph.D. (E.E.), 1968, Stanford University; U. S. Air Force Research and Development Officer, Wright-Patterson AFB, Ohio, 1960–1963; Bell Laboratories, 1968—. Since coming to Bell Laboratories from Stanford, where he was engaged in microwave transhorizon propagation research, Mr. Cox has been engaged in microwave propagation research in mobile radio environments and in high-capacity mobile radio systems studies. Senior Member, IEEE; Member, Commission II of USNC/URSI, Sigma Xi, Sigma Tau, Eta Kappa Nu, Pi Mu Epsilon.

B. Gopinath, M.S. (Mathematical Physics), 1964, University of Bombay, India; M.S.E.E. and Ph.D. (E.E.), 1968, Stanford University; Postdoctoral Research Associate, Stanford University, 1967–1968; Bell Laboratories, 1968—. Mr. Gopinath's primary interest, as a member of the Mathematics of Physics and Networks Department, is in the applications of mathematical methods to physical problems.

Shlomo Halfin, M.Sc., 1958, and Ph.D., 1962, The Hebrew University of Jerusalem (Israel); Bell Laboratories, 1968—. Mr. Halfin's work is in the field of Operations Research theory and its applications. Member, American Mathematical Society, Operations Research Society of America, Society for Industrial and Applied Mathematics.

Judith A. Johnson, A.A., 1969, Colby Jr. College; Bell Laboratories, 1970—. Ms. Johnson is a laboratory assistant in the Acoustics Research Department. She assists staff members in research on digital processing of speech, auditory information processing, and automatic and subjective speaker verification.

Robert P. Kurshan, Ph.D. (Mathematics), 1968, University of Washington; Bell Laboratories, 1968—. Mr. Kurshan's recent interests have been in the field of algebra, with work in invariant theory and applications in coding theory and information theory.

Charles J. McCallum, Jr., S.B. (Mathematics), 1965, Massachusetts Institute of Technology; M.S. (Statistics), 1967, and Ph.D. (Operations Research), 1970, Stanford University; Bell Laboratories, 1970—. Mr. McCallum is a member of the Operations Research Studies Department. His current work involves the application of mathematical programming techniques to network optimization problems, such as the multicommodity network flow problem. Member, Operations Research Society of America, The Institute of Management Sciences, Mathematical Programming Society, Sigma Xi.

James McKenna, B.Sc. (Mathematics), 1951, Massachusetts Institute of Technology; Ph.D. (Mathematics), 1961, Princeton University; Bell Laboratories, 1960—. Mr. McKenna has done research in quantum mechanics, electromagnetic theory, and statistical mechanics. He has recently been engaged in the study of nonlinear partial differential equations that arise in solid state device work, and in the theory of stochastic differential equations.

Lawrence R. Rabiner, S.B., S.M., 1964, Ph.D., 1967, Massachusetts Institute of Technology; Bell Laboratories, 1962—. Mr. Rabiner has worked on digital circuitry, military communications problems, and problems in binaural hearing. Presently he is engaged in research on speech communications and digital signal processing techniques. Member, Eta Kappa Nu, Sigma Xi, Tau Beta Pi; Fellow, Acoustical Society of America; Chairman of the IEEE G-AU Technical Committee on Digital Signal Processing; associate editor of IEEE Transactions on Audio and Electroacoustics; member of the technical committees on speech communication of both the IEEE and Acoustical Society.

Douglas O. Reudink, B.A., 1961, Linfield College; Ph.D. (Mathematics), 1965, Oregon State University; Bell Laboratories, 1964—. Since joining Bell Laboratories, Mr. Reudink has been engaged in electronic systems research with particular emphasis on the field of mobile communications. His recent work has been concerned with fundamentals of mobile radio propagation, diversity techniques, and the configuration and control of mobile systems. Member, Sigma Pi Sigma, American Mathematical Society, Pi Mu Epsilon, IEEE.

N. L. Schryer, B.S., 1965, M.S., 1966, and Ph.D., 1969, University of Michigan; Bell Laboratories, 1969—. Mr. Schryer has worked on the numerical solution of parabolic and elliptic partial differential equations. He is currently studying problems of this type which arise in semiconductor device theory.

Moshe Segal, B.Sc. (Mechanical Eng.), 1955, and Ingenieur (Mechanical Eng.), 1956, Technion Israel Institute of Technology; Dr. of Engineering (Operations Research), 1961, The Johns Hopkins University; Bell Laboratories, 1961—. Mr. Segal has worked in the fields of queueing theory and mathematical programming and their application to operational and engineering problems. He is supervisor of the Operations Research Methodology Group. Member, Operations Research Society of America, The Institute of Management Sciences, Mathematical Programming Society.

Michael Yamin, B.S., 1949, Polytechnic Institute of Brooklyn; Ph.D., 1952, Yale University; Mellon Institute, 1953–1956; Bell Laboratories, 1957—. Mr. Yamin has worked on problems of semiconductor materials and process development, especially those concerned with passivating thin films such as silicon dioxide, and on the development of computer aids to integrated circuit design.