IRE Transactions on CIRCUIT THEORY

Volume CT-6 MARCH, 1959 Number 1

SEQUENTIAL TRANSDUCER ISSUE

IRE Transactions on Circuit Theory

Published Quarterly by the Professional Group on Circuit Theory

Volume CT-6 MARCH, 1959 Number 1

TABLE OF CONTENTS

Abstracts

S. SESHU, R. E. MILLER, AND G. METZE, *Transition Matrices of Sequential Machines*—Page 5

In this paper a matrix technique is introduced for the analysis of state diagrams of synchronous sequential machines. The matrices introduced are closely related to the relation matrices of the calculus of relations and provide a formal tool for discussing state diagrams. It is shown that several of the well-known theorems on state diagrams are consequences of properties of transition matrices, which remain invariant under matrix multiplication. A reduction procedure for state diagrams, based on transition matrices, which is similar to Moore's technique, is given. A method of extending the results to asynchronous machines is also included.

S. H. UNGER, *Hazards and Delays in Asynchronous Sequential Switching Circuits*—Page 12

This paper is concerned with asynchronous, sequential switching circuits in which the variables are represented by voltage levels, not by pulses. The effects of arbitrarily located stray delays in such circuits are analyzed, and it is shown that, for a certain class of functions, proper operation can be assured regardless of the presence of stray delays and without the introduction of delay elements by the designer. All other functions require at least one delay element in their circuit realizations to insure against hazards. In the latter case it is shown that a single delay element is always sufficient. The price that must be paid for minimizing the number of delay elements is that of greater circuit complexity.

J. M. SIMON, *A Note on Memory Aspects of Sequence Transducers*—Page 26

This paper defines several classes of sequence transducers whose operations exhibit simple forms of memory. Some of the special properties and interrelationships for these classes of transducers are established.

W. J. CADDEN, *Equivalent Sequential Circuits*—Page 30

Three types of sequential circuits are defined, two of which are synchronous and one of which is asynchronous. The concept of equivalent sequential circuits as discussed by Huffman, Mealy, and Moore is extended to circuits of different types. Transformation procedures are given for transforming a state table of one type into state tables of the other types. One of these transformations can also be used to introduce unit delay between corresponding inputs and outputs for a synchronous circuit. The transformation methods allow a comparison of circuits, or state tables of different types to be made for a given sequential circuit problem. A few general conclusions are drawn about the different types of sequential circuits.

FREDERICK C. HENNIE, *Analysis of Bilateral Iterative Networks*—Page 35

In the usual iterative switching circuit, which may be considered the space analog of a synchronous sequential transducer, the output of any cell is dependent only upon the inputs of the cells to its left. This paper describes a more general type of one-dimensional iterative network in which the output of each cell may be a function of the inputs of all the cells in the network, both to the left and to the right of the given cell. Starting from a table of combinations which specifies the behavior of an individual cell, a means of describing the steady-state behavior of the entire network is developed. This description is readily reduced to a fairly simple canonic form, so that equivalent networks can be recognized. Certain types of redundancy which do not occur in an ordinary iterative or sequential network are discussed, and a means of detecting these redundancies is described. Examples are presented which indicate that the process of removing redundancies is more complex than the corresponding process in the sequential case. Finally, one method of synthesizing a stable bilateral iterative network is described, and some of the problems of transient behavior are indicated.

BERNARD ELSPAS, *The Theory of Autonomous Linear Sequential Networks*—Page 45

Analysis and synthesis techniques for a class of sequential discrete-state networks are discussed. These networks, made up of arbitrary interconnections of unit-delay elements (or of trigger flip-flops), modulo-p adders, and scalar multipliers (modulo a, prime p), are of importance in unconventional radar and communication systems, in automatic error-correction circuits, and in the control circuits of digital computers. In addition, these networks are of theoretical significance to the study of more general sequential networks.

The basic problem with which this paper is concerned is that of finding economical realizations of such networks for prescribed autonomous (excitation-free) behavior. To this end, an analytical-algebraic model is described which permits the investigation of the relation between network logical structure and state-sequential behavior. This relation is studied in detail for nonsingular networks (those with purely cyclic behavior). Among the results of this investigation is the establishment of relations between the state diagram of the network and a characteristic polynomial derived from its logical structure. An operation of multiplication of state diagrams is shown to correspond to multiplication of the corresponding polynomials.

A criterion is established for the realizability of prescribed cyclic behavior by means of linear autonomous sequential networks. An effective procedure for the economical realization of such networks is described, and it is shown that linear feedback shift registers constitute a canonical class of realizations. Examples are given of the realization procedure. The problem of synthesis with only one-cycle length specified is also discussed. A partial solution is obtained to this "don't care" problem.

Some special families of feedback shift registers are investigated in detail, and the state-diagram structures are obtained for an arbitrary number of stages and an arbitrary (prime) modulus.

Mathematical appendixes are included which summarize the pertinent results in Galois field theory and in the factorization of cyclotomic polynomials into irreducible factors over a modular field.

The relation of the theory developed in this paper to Huffman's description of linear sequence transducers in terms of the D operator is discussed, as well as unsolved problems and directions for further generalization.

BERNARD FRIEDLAND, *Linear Modular Sequential Circuits*—Page 61

Sequential circuits comprising 1) modulo-p (p = prime) summers, 2) amplifiers whose gains are integers $<p$, and 3) unit delays are considered in this paper which constitutes an extension of earlier work by Huffman. Such circuits are characterized in terms of the modular field $GF(p)$ and vectors and matrices defined thereover. A summary of the properties of $GF(p)$ is given.

A linear sequential circuit is defined in terms of

$$\vec{y}(n) = C\vec{s}(n) + D\vec{x}(n)$$

$$\vec{s}(n+1) = A\vec{s}(n) + B\vec{x}(n)$$

where A, B, C, and D are $k \times k$ matrices defined over $GF(p)$. The latter equations constitute a canonical representation of any circuit comprising the above listed components. It is shown that circuits of this type meet the usual additivity criterion of linear systems.

The behavior of the circuit is described in a finite state space of k dimensions and p^k states. The autonomous circuit (A, B, C, D = constant and $\vec{x}(n) = \vec{0}$, all n) is characterized by the matrix A. If A is nonsingular all initial states are either finite equilibrium points or lie in periodic sequences of length $T_r \leq T_{max} = p^k - 1$. If the minimum polynomial of A^r has distinct roots, T_r divides $r(p-1)$. If A is singular, there are some singular initial states to which the circuit cannot return in the absence of excitation.

The use of Z transforms for linear modular sequential circuits is demonstrated. Inputs and outputs are represented by their "transforms" and the circuit by its "transfer function." The transform of the output is the product of the transfer function and the transform of the input. Several illustrative examples are included.

JURIS HARTMANIS, *Linear Multivalued Sequential Coding Networks*—Page 69

Linear multivalued sequential coding networks are circuits whose input and output are synchronized sequences of non-negative integers less than some fixed number m. The output depends linearly on the present input and a finite number of previous inputs and outputs. The transfer characteristics of such a network are described by a ratio of polynomials in the delay operator, where the multiplication and addition are performed with respect to the fixed modulus m. An algebraic theory of the delay polynomials is obtained. It is shown that a polynomial has a complete set of null sequences if, and only if, its first and last coefficients are prime to the modulus m. The polynomials with no null sequences are characterized. It is shown when common null sequences imply that the polynomials have common factors and that a complete set of null sequences defines the polynomial.

It is also shown that a transfer function can be realized if the denominator contains a constant term prime to m and explicit constructions are given. A network is stable if the polynomial in the denominator of the transfer junction has no null sequence. Thus any nontrivial polynomial or its inverse is unstable if we are working modulo a prime. If the modulus is not prime, stable networks with stable inverses are constructed. Finally it is indicated how polynomials with no null sequences can be used to simplify the construction of coding networks.

CARL F. SIMONE, *Equivalent Ladder Networks by the Use of Signal Flow Graphs*—Page 75

Signal flow graphs of ladder networks have properties that make them convenient for determining impedances and transfer ratios. Because of the symmetry of these flow graphs it is possible to recognize equivalent flow graphs, and hence equivalent networks, with respect to some desired characteristic. In particular, the output-input voltage ratio is the characteristic that is used as the basis for equivalence. Evaluation of elements in the equivalent circuits results from relating coefficients in the transfer voltage ratio to the element values.

Using 4-branch ladder networks, examples are given of distributing resistance, determining when networks must use active elements for certain transfer functions, and finding the number of equivalences that exist.

A particular equivalence is derived between a bridged-T and ladder, and between a lattice and ladder. In each case, three of the branches of the ladder are the same as in the bridged-T or lattice, while the fourth branch of the ladder is a function of all the impedances. These equivalences are derived by recognizing the flow graph configuration for a ladder within the flow graph of each of the other circuits and then reducing these flow graphs to the one for the ladder.

DANIEL C. FIELDER, *Identification of Certain Networks with Reflection Coefficient Zero Locations*—Page 81

In this paper, the coefficients of return loss expansions are found for certain low-pass, LC ladder networks which have n lossless elements and which exhibit Tchebycheff (or equal ripple) pass band and monotonic stop band transmission behaviors. The return loss expansion is the Taylor expansion of $\ln (1/\rho_1(s))$ about s equal to infinity, the variable s being the familiar complex frequency variable $s = \sigma + j\omega$, and ρ_1 being the reflection coefficient between a resistive termination and the remainder of the network. The return loss coefficients are tabulated according to reflection zero locations for odd and even n.

Methods for synthesizing low-pass, LC ladder networks from return loss coefficients are available. A presentation of the modifications necessary to adapt these methods for use with the particular coefficients discussed above is given. Thus, it is possible to synthesize certain Tchebycheff networks through use of return loss coefficients which are, in turn, directly identified with reflection zero locations.

The paper concludes with a brief discussion of the extension of existing tables of Tchebycheff network element values for finding the element values for several reflection zero distributions and LC ladder arrangements.

ABRAHAM BERS, *The Degrees of Freedom in RLC Networks*—Page 91

It is shown here that the number of degrees of freedom, or what is equivalent—the number of natural frequencies—of any RLC network can readily be determined from the number of energy-storing elements and the topology of the network. The effect of loss (resistance) in altering the number of degrees of freedom is explained.

RICHARD LA ROSA, *Pole Migration in Coupled-Resonator Filters*—Page 95

The narrow-band, coupled-resonator filter is analyzed by giving a vector interpretation to the transfer function. The significant parameters are the natural frequencies of the complete filter and the natural frequencies of the individual resonators. It is shown that insertion loss is related to the migration of the natural frequencies (poles) as the coupling coefficients between resonators are increased from zero. A vector construction is described for the pole migration of three coupled resonators.

D. C. YOULA, L. J. CASTRIOTA, AND H. J. CARLIN, *Bounded Real Scattering Matrices and the Foundations of Linear Passive Network Theory*—Page 102

In this paper the most general linear, passive, time-invariant n-port (*e.g.*, networks which may be both distributed and nonreciprocal) is studied from an axiomatic point of view, and a completely rigorous theory is constructed by the systematic use of theorems of Bochner and Wiener. An n-port Φ is defined to be an operator in H_n, the space of all n-vectors whose components are measurable functions of a real variable t, $(-\infty < t < \infty)$ (and as such need not be single-valued). Under very weak conditions on the domain of Φ, it is shown that linearity and passivity imply causality. In every case, Φ_a, the n-port corresponding to ϕ augmented by n series resistors is always causal (Φ_a is the "augmented network," Fig. 2). Under the further assumptions that the domain of Φ_a is dense in Hilbert space and ϕ is time-invariant, it is proved that Φ possesses a frequency response and defines an $n \times n$ matrix $S(z)$ (the scattering matrix) of a complex variable $z = \omega + i\beta$ with the following properties: 1) $S(z)$ is analytic in $\operatorname{Im} z > 0$; 2) $Q(z) = 1_n - S^*(z)S(z)$ is the matrix of a non-negative quadratic form for all z in the strict upper half-plane and almost all ω. Conversely, it is also established that any such matrix represents the scattering description of a linear, passive, time-invariant n-port Φ such that the domain of Φ_a contains all of Hilbert space. Such matrices are termed "bounded real scattering matrices" and are a generalization of the familiar positive-real immittance matrices.

When Φ and Φ^{-1} are single-valued, it is possible to define two auxiliary positive-real matrices $Y(z)$ and $Z(z)$, the admittance and impedance matrices of Φ, respectively, which either exist for all z in $\operatorname{Im} z > 0$ and almost all ω or nowhere. The necessary and sufficient conditions for an $n \times n$ matrix $A_n(z)$ to represent either the scattering or immittance description of a linear, passive, time-invariant n-port Φ are derived in terms of the real frequency behavior of $A_n(\omega)$.

Necessary and sufficient conditions for Φ_a to admit the representation

$$\mathbf{i}(t) = \int_{-\infty}^{\infty} dW_n(\tau)\mathbf{e}(t - \tau)$$

for all integrable $\mathbf{e}(t)$ in its domain are given in terms of $S(z)$. The last section concludes with a discussion concerning the nature of the singularities of $S(z)$ and the possible extension of the theory to active networks.

KENT E. ERICKSON, *A New Operation for Analyzing Series-Parallel Networks*—Page 124

The operation * is defined as $A * B = AB/A + B$. The symbol * has algebraic properties which simplify the formal solution of many series-parallel network problems. If the operation * were included as a subroutine in a digital computer, it could simplify the programming of certain network calculations.

Buyer's Guide

DAVID A. HUFFMAN†, *GUEST EDITOR*

IN THE SEARCH for understanding of the theoretical capabilities of computers and other decision-making logical machines, two abstract models have been found useful: finite-state machines and Turing machines. Roughly speaking, a finite-state machine emulates a collection of elemental bistable devices interconnected so that they exhibit a memory. A Turing machine consists of a finite-state circuit operating in conjunction with an infinite tape which it scans and upon which symbols are read, erased, and rewritten. Since the finite-state model is a component of even the Turing machine, a significant amount of effort has gone into the study of its organization, internal communication problems, and the complexity requisite to accomplishing various general transformations on the sequences of digits which constitute its input data. In the role of a sequence transducer between its input and output data streams the finite-state circuit (also called sequential circuit, or finite-state automaton, etc.) acts somewhat analogously to an electrical filter operating on voltage or current waveforms. That is the primary reason that the papers of this issue are found here rather than in, say, a mathematics journal.

One need go back less than twenty-five years to find the earliest papers on the theory of logical machines. Turing's paper, "On Computable Numbers," published in 1936[1] is perhaps, even today, the single most important one, creating as it does the relationship between logical propositions and associated idealized machines. Later McCulloch and Pitts, in "A Logical Calculus of the Ideas Immanent in Nervous Activity," (1943) established the possibility of simulating the logical aspects of nervous activity by nets of neuron-like hypothetical digital elements.

The important idea of "state" as it applies to finite-state circuits was developed independently by Huffman ("The Synthesis of Sequential Switching Circuits," 1954), Kleene ("Representation of Events in Nerve Nets and Finite Automata," 1956), and Moore ("Gedanken Experiments on Sequential Machines," 1956). The necessity for a precise definition of state was implicit in the problems to which they devoted themselves. Huffman gave an explicit procedure for synthesizing finite-state circuits from networks of switching elements whose response times were not uniformly the same, and this required an exact specification of the desired terminal action. Kleene investigated in detail the capabilities and limitations of arbitrary networks of artificial neurons, putting the earlier work in the area in a neat mathematical form. Moore required a precise characterization of the terminal behavior of finite-state circuits for his investigations of the equivalence of various forms of an automaton as determined by external experiments.

The "state" of a finite-state machine refers to the set of signal responses existing at any time at the output ends of the feedback loops within the circuit. The set of signals at the inputs of these loops will then determine the next state of the circuit. The signals which the circuit receives from and those which it delivers to its outside environment also enter into its operation. In fact the key statement in the theory of finite-state machines is merely that the next state and the output of the machine are functions determined by its present state and its input.

In this issue the paper by Seshu, Miller, and Metze investigates a matrix formulation of this statement and of some of its consequences. Unger examines in some detail the effects of stray delays in the elemental switching devices making up asynchronous sequential circuits. Simon shows some special properties of some sequential transducers which have simple forms of memory. In Cadden's paper the question of equivalence of circuits whose data are presented in various forms, but which have the same ultimate design objective, is treated.

A formal analysis of a quite general class of one-dimensional iterative circuits is set forth by Hennie. Since one subclass of these circuits is analogous to the conventional finite-state machine this paper may eventually lead to a better understanding both of finite-state circuits and also of uniform arrays of logical elements in more than one dimension. A comprehensive review of the present state of the art in linear sequential networks is the end-product of the paper by Elspas. The subject matter of the papers by Friedland and Hartmanis is the extension of previous work on linear binary circuits to other number bases.

The guest editor wishes to acknowledge gratefully the copious assistance of the assistant editors of this issue: Prof. Dean Arden, M.I.T.; Dr. William H. Kautz, Stanford Research Institute; Dr. E. F. Moore, Bell Telephone Laboratories; Prof. D. Muller, University of Illinois; and Prof. G. W. Patterson, University of Pennsylvania.

† Mass. Inst. Tech., Cambridge 39, Mass.

[1] Nearly all of the papers referred to here and other important ones as well may be found as references in or as papers comprising "Automata Studies," Annals of Mathematics Studies No. 34, Princeton University Press, Princeton, N. J.; 1956.

Transition Matrices of Sequential Machines*

S. SESHU†, R. E. MILLER‡, AND G. METZE§

I. Definitions

SINCE the terminology in switching theory is far from standardized, it is well for us to begin with a few basic definitions. Following Moore[1] we define a *sequential machine M* as consisting of a finite collection of 1) states, 2) possible inputs, and 3) possible outputs, and satifying the conditions: 1) the present output is uniquely determined by the present state, and 2) the present state is uniquely determined by the previous state and the previous input.

In this definition, one considers the words *state*, *input*, and *output* to be undefined concepts. The terms *input* and *output* have the familar interpretations in terms of practical machines. The state may be interpreted as the set of states of the internal components of the machine, in which case the description above applies only to a small subset of practical machines; or it may be interpreted as the combined state of the internal components and the input, thus encompassing a larger class of machines. In terms of the flow table of Huffman[2] the first interpretation corresponds to calling each row a state, and the second interpretation corresponds to calling each circled entry a state. We shall, however, restrict our discussion to well-behaved or *deterministic* machines.

With a sequential machine as defined above, we can associate a *weighted directed graph* or a *net*,[3] called the *transition diagram* by Moore.[1] Each vertex q_i of the net corresponds to a state of the sequential machine and each edge corresponds to a transition between states. The output associated with a state is assigned to the corresponding vertex of the net as the *weight* of the vertex. Similarly the input that causes a given transition is assigned as the weight of the edge corresponding to the transition. The net associated with a sequential machine is known as the *state diagram* of the machine. The assumption that the machine is deterministic (rather than probabilistic) is reflected in the state diagram by the fact that no two edges with the same initial vertex have the same weight. An example of a state diagram is shown in Fig. 1.

It is convenient to define the terms synchronous and asynchronous in terms of the state diagram. The machine M is a *synchronous machine* if for each permissible input i and each state q_i of M, there is an edge with weight i leaving vertex q_i of the state diagram. The machine is *asynchronous* if no edge leaving vertex q_i has the same weight as any edge entering vertex q_i. In other words any input i_1 may follow any input i_2 ($i_1 = i_2$ or $i_1 \neq i_2$) in a synchronous machine whereas $i_1 = i_2$ is not permissible in an asynchronous machine. The words "entering" and "leaving" are used in the geometrical sense implied by the arrowheads in the state diagram. (This definition differs from that of Aufenkamp and Hohn.[4])

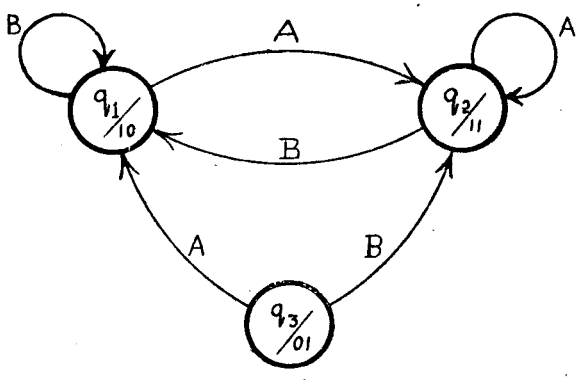


Fig. 1—Example of a state diagram.

Aufenkamp and Hohn have devised a matrix scheme for discussing state diagrams based on the *connection matrix*. (They have applied the connection matrix technique to Mealy's[5] model of a sequential machine, but the application to Moore's model is certainly possible.) The connection matrix simply describes the structure of the net and is defined as follows. The connection matrix has one row and one column for each state and is given by:

$$C = [c_{ij}]_{n,n} \tag{1}$$

$$c_{ij} = \sum_k w_{ij}^k \tag{2}$$

where w_{ij}^k takes state i into state j when input w^k occurs and the summation is over all such inputs (with the interpretation OR for the sum).

The *transition matrix* on the other hand describes the *distribution* of edges of a *given weight* (input). More precisely, we associate with each input i (that is permissible) a transition matrix T^i defined by

* Manuscript received by the PGCT, December 23, 1957. The work of Seshu was sponsored by the graduate college of the University of Illinois in 1955.

† University of Toronto, Toronto, Ont., Can.
‡ IBM Corp., Yorktown Heights, N. Y.
§ Digital Computer Lab., University of Illinois, Urbana, Ill.

[1] E. F. Moore, "Gedanken-Experiments on Sequential Machines," in "Automata Studies," Princeton University Press, Princeton, N. J., Study 34, pp. 129–153; 1956.

[2] D. A. Huffman, "The synthesis of sequential switching circuits," *J. Franklin Inst.*, vol. 257, pp. 161–190, 275–303; March and April, 1954.

[3] F. E. Hohn, S. Seshu, and D. D. Aufenkamp, "The theory of nets," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-6, pp. 154–161; September, 1957.

[4] D. D. Aufenkamp and F. E. Hohn, "Analysis of sequential machines," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-6, pp. 276–285; December, 1957.

[5] G. H. Mealy, "Method for synthesizing sequential circuits," *Bell Sys. Tech. J.*, vol. 34, pp. 1054–1079; September, 1955.

$$T^i = [t^i_{kj}]_{n,n} \tag{3}$$

where $t^i_{kj} = 1$ if input i takes q_k into q_j, $t^i_{kj} = 0$ otherwise. Thus T^i is square and has the same order as C, namely the number of states of M. The transition matrices are evidently related to the connection matrix by[3]:

$$C = \sum_i w_i T^i \tag{4}$$

where w_i stands for the ith input.

With suitable correspondences it can be seen that these transition matrices are the same as the relation matrices defined by Copi[6] for the calculus of binary relations. Very similar matrices have also been used by Shimbel[7] who called them structure matrices.

II. Elementary Properties of Transition Matrices

Without further statement we shall assume that, except for the concluding remarks at the end of the paper, all machines being considered are *synchronous machines.*

Theorem 1

Every row of the transition matrix T^i contains exactly one nonzero entry 1.

This theorem is an immediate consequence of the definitions of "synchronous" and "deterministic."

Theorem 2

The property given in Theorem 1 is an invariant under multiplication of transition matrices (the multiplication being performed in the usual way with $0 + 1 = 1 + 0 = 1$, $1 \cdot 0 = 0 \cdot 1 = 0 \cdot 0 = 0$, $1 \cdot 1 = 1$, $0 + 0 = 0$).

Proof: Let T^i and T^j be transition matrices. Then we have to prove that each row of $T^i \cdot T^j$ contains exactly one 1, all other entries being zero. This follows immediately, since each row of T^i contains exactly one 1 and so the rows of $T^i \cdot T^j$ are chosen from the rows of T^j, which has the required property. The result extends by induction to the product of any finite number of matrices.

If $q_1, q_2, \cdots, q_n$ are the states of the machine and $\omega_1, \omega_2, \cdots, \omega_n$ are the corresponding outputs, then we define the state vector Q_0 and the output vector Ω_0 by:

$$Q_0 = \begin{bmatrix} q_1 \\ q_2 \\ \cdot \\ \cdot \\ q_n \end{bmatrix} \qquad \Omega_0 = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \cdot \\ \cdot \\ \omega_n \end{bmatrix}. \tag{5}$$

Theorem 3

If the machine is in state q_k and the input i is applied, the next state and the next output are given by the kth rows of T^iQ_0 and $T^i\Omega_0$, respectively, where T^i is the transition matrix for input i.

This result is obvious.

Theorem 4

If the input sequence i_1i_2 (*i.e.*, i_1 followed by i_2) is applied to the machine in state q_k, the state and output after i_1i_2 are given by the kth rows of $T^{i_1}T^{i_2}Q_0$ and $T^{i_1}T^{i_2}\Omega_0$, respectively.

Proof: Let i_1 take q_k into q_r and let i_2 take q_r into q_p. Then by the definition of a transition matrix, the kth row of T^{i_1} contains a 1 in column r and the rth row of T^{i_2} contains a 1 in column p. Therefore, the kth row of $T^{i_1}T^{i_2}$ is merely the rth row of T^{i_2}. Therefore the kth row of $(T^{i_1}T^{i_2})Q_0$ is q_p and the kth row of $(T^{i_1}T^{i_2})\Omega_0$ is ω_p. (The multiplication is associative, the parentheses being included merely to illustrate the argument.)

This result may be extended by induction to yield the following.

Theorem 5

If the input sequence $i_1i_2 \cdots i_p$ is applied to the machine in state q_k, the final state and output are given by the kth rows of $T^{i_1}T^{i_2} \cdots T^{i_p}Q_0$ and $T^{i_1}T^{i_2} \cdots T^{i_p}\Omega_0$, respectively.

We may remark here that the order in which the transition matrices are multiplied in Theorems 4 and 5 is the reverse of what one would expect intuitively. If, for instance, we were to develop an operator algebra, one would expect the operator for the first input i_1 to operate on Q_0, the operator for the next input i_2 to operate on the result, etc. Results that are very similar to Theorems 3–5 have also been obtained by Shimbel[7] who uses row vectors rather than column vectors.

We can also observe that the information obtained by multiplying transition matrices is somewhat analogous to the multiple experiment of Moore,[1] who considers taking n copies of a machine, all of them in the same initial state, and performing different experiments on them. Considering transition matrices on the other hand is equivalent to taking n copies of the machine, each in a different initial state, and performing the same experiment on all of them.

We shall conclude this section with the statement of a very simple invariant property of transition matrices that characterizes combinational machines thus hinting at an invariant theory of sequential machines.

Theorem 6

For every i, let the nonzero entries of transition matrix T^i be in the same column r_i (r_i may vary with i). Then the product of any number of these matrices also has the same property.

This result is obvious. The rows of T^i are identical and hence so are the rows of T^iT^j.

This particular invariant characteristic really characterizes *combinational* machines. At the outset this remark

[6] I. M. Copi, "Matrix development of the calculus of relations," *J. Symbolic Logic*, vol. 13, pp. 193–203; December, 1948.
[7] A. Shimbel, "Application of matrix algebra to communication nets," *Bull. Math. Biophys.*, vol. 13, no. 3, pp. 165–173; 1951.

may seem misleading as "state" is usually associated with a memory and a combinational machine has no memory. However, if we take "state" in the larger sense (see remarks at the beginning of Section I), each of the states represented in the transition matrix may be just an input state, the machine having no memory. In any case, any machine with the property assumed in Theorem 6 can be replaced by an equivalent combinational machine (with at most a different delay in the output).

III. Invariant Theory of Equivalence of States

To avoid certain logical circles (such as in the example of Huffman[8]) we define two types of equivalences of states. Neither of them agrees with Moore's[1] definition of indistinguishability except for the classes of strongly connected machines and machines with a natural *initial* or *cleared* state (which encompass most practical, useful or well-designed machines). Moore chooses to define indistinguishability with respect to a given experiment; so that two states may be indistinguishable by one experiment but distinguishable by another. Although this definition is more general and more in keeping with the "gedanken-experiment" point of view, the more restricted definition that we shall give is more amenable to theoretical treatment. In general, for machines for which the definitions do not agree, Moore's definition would lead to a simpler reduced machine than the other more restricted definitions.[9] Our first definition, that of simple equivalence, is similar to those of Mealy[5] and Aufenkamp and Hohn.[4]

States $q_{i_1}, q_{i_2}, \cdots, q_{i_m}$ of machine M are *simply equivalent* if and only if 1) the outputs associated with those states are identical, and 2) every input sequence applied to M with M in any one of $q_{i_1}, q_{i_2}, \cdots, q_{i_m}$ leads to the same output sequence; with 2) being independent of the outputs associated with states other than $q_{i_1}, q_{i_2}, \cdots, q_{i_m}$.

As Aufenkamp has pointed out to the authors, this definition is slightly peculiar, in that a proper subset of a simply equivalent set may not be simply equivalent. For example, in the state diagram of Fig. 2, the states 1, 2, and 3 form a simply equivalent set. But under the given definition states 1 and 2 are not simply equivalent, *i.e.*, independently of the output associated with state 3. However, there are no theoretical difficulties introduced by this peculiarity since simple equivalence has not been defined as a binary relative.

Theorem 7

Let the rows and columns of the transition matrices T^i be ordered such that $i_1, i_2, \cdots, i_m$, are the first m rows and columns; and let the matrices be partitioned after the first m rows and columns, as:

$$T^i = \begin{bmatrix} T^i_{11} & T^i_{12} \\ T^i_{21} & T^i_{22} \end{bmatrix}. \tag{6}$$

[8] Huffman, *op. cit.*, p. 182.
[9] The authors are indebted to Moore for this clarification.

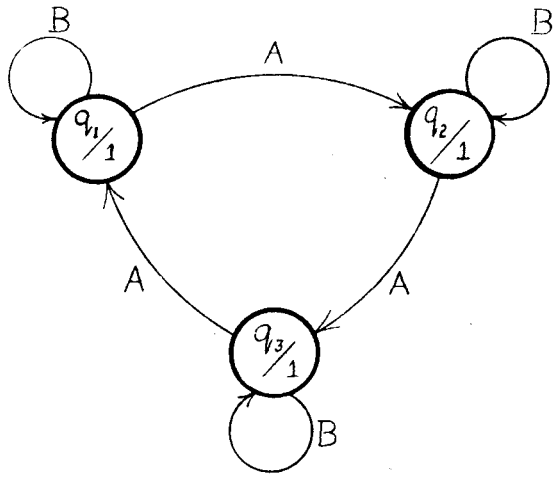


Fig. 2—Illustration of Aufenkamp's remark.

Then the property that the rows of T^j_{12} are identical for $1 \leq j \leq k$ (where k is the number of permissible inputs) is an invariant under multiplication of transition matrices.

Proof: Let $1 \leq j \leq k$, $1 \leq p \leq k$. Let

$$T^i = \begin{bmatrix} T^i_{11} & T^i_{12} \\ T^i_{21} & T^i_{22} \end{bmatrix} \quad \text{and} \quad T^p = \begin{bmatrix} T^p_{11} & T^p_{12} \\ T^p_{21} & T^p_{22} \end{bmatrix} \tag{7}$$

satisfy the condition that the rows of the (1, 2) submatrix are identical. Let

$$T^pT^i = \begin{bmatrix} \tau_{11} & \tau_{12} \\ \tau_{21} & \tau_{22} \end{bmatrix}. \tag{8}$$

(All the matrices are partitioned after m rows and columns.) Then

$$\tau_{12} = T^p_{11}T^i_{12} + T^p_{12}T^i_{22}. \tag{9}$$

Since T^p contains exactly one 1 per row and T^p_{12} has identical rows, either $T^p_{11} = 0$ or $T^p_{12} = 0$ but not both. Now if $T^p_{11} = 0$, then the rows of τ_{12} are identical since the rows of T^p_{12} are identical and $\tau_{12} = T^p_{12}T^i_{22}$. And if $T^p_{12} = 0$, $\tau_{12} = T^p_{11}T^i_{12}$; so that the rows of τ_{12} are selected from identical rows of T^i_{12}. Hence, the rows of τ_{12} are identical. An obvious induction step completes the proof.

Theorem 8

Let the transition matrices be arranged and partitioned as in Theorem 7. Then the states $q_{i_1}, q_{i_2}, \cdots, q_{i_m}$ are simply equivalent if and only if they have the same output and the rows of T^i_{12} are identical for $1 \leq j \leq k$, where k is the number of permissible inputs.

Proof: The sufficiency follows from the invariance of the property proved in Theorem 7. To prove that the given condition is necessary for equivalence, we make use of the fact that equivalence must be independent of the outputs associated with the other states and so the first m rows of

$$T^i\Omega_0 \ (j = 1, 2, \cdots, k)$$

will be equal if and only if the rows of T^i_{12} are identical for $1 \leq j \leq k$.

The second type of equivalence that we shall consider is that of *multiple equivalence*. This is the case where the equivalence of one set of states depends on the equivalence of another set of states and conversely. (Of course, the cycle may include more than two sets of states.)

Let $S_1, S_2, \cdots, S_m, S_{m+1}$ be a partition of the states of a machine M. Within a given S_p, $1 \leq p \leq m$, let the outputs associated with all the states be the same. Suppose every input sequence applied with the machine in state $q_j \in S_p$ leads to the same output sequence for all $q_j \in S_p$, $1 \leq p \leq m$; independently of the outputs associated with the partitions $S_1, S_2, \cdots, S_m$ and the states in S_{m+1}. Then the states in $S_1, S_2, \cdots, S_m$ are *multiply equivalent*.

We might remark here that the states in S_1 and S_2 might be multiply equivalent without the states in S_1 (or S_2) being simply equivalent.

Theorem 9

Let the states in $S_1, S_2, \cdots, S_m$ be multiply equivalent. Then the simultaneous identification of the states in S_p, $1 \leq p \leq m$, *i.e.*, replacement of each set by a state, leads to an equivalent machine in Moore's definition.

This result is obvious.

Theorem 10

Let $S_1, S_2, \cdots, S_m, S_{m+1}$ be a partition of the states of M. Let the rows and the columns of the transition matrices be ordered correspondingly, and partitioned as:

$$T^i = \begin{bmatrix} T^i_{1,1} & T^i_{1,2} & \cdots & T^i_{1,m} & T^i_{1,m+1} \\ T^i_{2,1} & T^i_{2,2} & \cdots & T^i_{2,m} & T^i_{2,m+1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ T^i_{m,1} & T^i_{m,2} & \cdots & T^i_{m,m} & T^i_{m,m+1} \\ T^i_{m+1,1} & T^i_{m+1,2} & \cdots & T^i_{m+1,m} & T^i_{m+1,m+1} \end{bmatrix} \tag{10}$$

where T^i_{ij} corresponds to S_j, $1 \leq j \leq m+1$. Further let the transition matrices satisfy property P:

The nonzero entries in any one of the first m rows of the partitioned matrix T^i are all in the same submatrix, $1 \leq i \leq k$; and the rows of the submatrix $T^i_{j,m+1}$ are identical for $1 \leq j \leq m$, for each i, $1 \leq i \leq k$.

Then property P is an invariant under matrix multiplication.

The invariance of property P follows from a computation similar to that of Theorem 7.

Theorem 11

Let the states of the machine M be partitioned into sets $S_1, S_2, \cdots, S_m, S_{m+1}$. Let all the states in a given partition S_j have the same output for $1 \leq j \leq m$. Then the invariant property P of Theorem 10 is a necessary and sufficient condition for the multiple equivalence of the states in $S_1, S_2, \cdots, S_m$.

The necessity follows from the condition that multiple equivalence be independent of the outputs as stated. The sufficiency follows from the invariance of property P.

IV. Reduction Algorithms and Theorems Concerning the Reduced Machine

For the sake of completeness we shall state the formal procedure for the reduction of state diagrams containing equivalent states. The formal procedure may be readily automated.

Let the states $q_{i_1}, q_{i_2}, \cdots, q_{i_m}$ be simply equivalent. Let the rows and columns of the transition matrices T^i be arranged so that these states correspond to the first m rows, and are partitioned after the first m rows and columns as:

$$T^i = \begin{bmatrix} T^i_{11} & T^i_{12} \\ T^i_{21} & T^i_{22} \end{bmatrix}. \tag{11}$$

We now obtain the reduced machine M' by replacing the states $q_{i_1}, q_{i_2}, \cdots, q_{i_m}$ by a single state q' with an output equal to the common output of the deleted states. The transition matrix S^i of M' (for input i) is defined by:

$$S^i = \begin{bmatrix} S^i_{11} & S^i_{12} \\ S^i_{21} & S^i_{22} \end{bmatrix} \tag{12}$$

where

$$\begin{aligned} S^i_{22} &= T^i_{22} \text{ of (11)}, \\ S^i_{12} &= \text{one row of } T^i_{12}, \\ S^i_{11} &= \begin{cases} 0 \text{ if } T^i_{11} \text{ is a zero matrix} \\ 1 \text{ if } T^i_{11} \text{ is not a zero matrix,} \end{cases} \\ S^i_{21} &= \Sigma \text{ columns of } T^i_{21} \text{ (Boolean sum),} \end{aligned}$$

i.e., S^i_{21} is a single column matrix, each entry of which is a Boolean sum of the entries in the corresponding row of T^i_{21}.

Theorem 12

The reduced machine is synchronous and is equivalent to the original machine under Moore's definition.

This theorem is obvious on observing that each row of the reduced transition matrix contains exactly one 1 (remembering that $T^i_{12} \neq 0$ implies that $T^i_{11} = 0$ and $T^i_{11} \neq 0$ implies $T^i_{12} = 0$); the equivalence follows directly from the definition.

Since the reduced machine is synchronous, one can again apply the reduction process if there are any more equivalent states. Of course if one wishes to use this procedure in practice, one would group together the states that have the property of Theorem 7. Needless to say, this process is not the fastest method of finding equivalent states. Nonetheless it is routine.

Example 1:

As a simple example of reduction of state diagrams with simply equivalent states, consider Fig. 3. With rows and columns arranged in the order q_1, q_2, q_3, the transition matrices are:

$$T^{00} = \left[\begin{array}{cc|c} 0 & 1 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right] \quad T^{01} = \left[\begin{array}{cc|c} 0 & 0 & 1 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \end{array}\right]$$

$$T^{10} = \left[\begin{array}{cc|c} 0 & 0 & 1 \\ 0 & 0 & 1 \\ \hline 0 & 0 & 1 \end{array}\right] \quad T^{11} = \left[\begin{array}{cc|c} 0 & 1 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 1 & 0 \end{array}\right].$$

It is seen that these transition matrices possess the invariant property of Theorem 7, with the partitioning shown. Since q_1 and q_2 also have the same output, they are simply equivalent. The reduction process now gives:

$$S^{00} = \begin{array}{c} q' \\ q_3 \end{array}\left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right] \quad S^{01} = \begin{array}{c} q' \\ q_3 \end{array}\left[\begin{array}{cc} 0 & 1 \\ 1 & 0 \end{array}\right]$$

$$S^{10} = \begin{array}{c} q' \\ q_3 \end{array}\left[\begin{array}{cc} 0 & 1 \\ 0 & 1 \end{array}\right] \quad S^{11} = \begin{array}{c} q' \\ q_3 \end{array}\left[\begin{array}{cc} 1 & 0 \\ 1 & 0 \end{array}\right].$$

The reduced machine is shown in Fig. 4.

The reduction process for multiple equivalence is very similar. Here let the original matrices be given by:

$$T^i = \begin{bmatrix} T^i_{1,1} & T^i_{1,2} & \cdots & T^i_{1,m} & T^i_{1,m+1} \\ T^i_{2,1} & T^i_{2,2} & \cdots & T^i_{2,m} & T^i_{2,m+1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ T^i_{m,1} & T^i_{m,2} & \cdots & T^i_{m,m} & T^i_{m,m+1} \\ T^i_{m+1,1} & T^i_{m+1,2} & \cdots & T^i_{m+1,m} & T^i_{m+1,m+1} \end{bmatrix} \tag{13}$$

where the first m rows and columns correspond to sets of multiply equivalent states. Then, of course, the matrices T^i have the invariant property P of Theorem 10. Now we obtain the reduced machine as follows. Each of the sets $S_1, S_2, \cdots, S_m$ is replaced by a state q'_i with the output corresponding to the set S_i. To get the transition matrix τ^i of the reduced machine, each of the submatrices $T^i_{k,j}$, $1 \leq k \leq m$, $1 \leq j \leq m$, is replaced by a single element, 0 or 1, determined from $T^i_{k,j}$ being zero or nonzero, respectively. Each submatrix $T^i_{k,m+1}$, $1 \leq k \leq m$, is replaced by one of its rows. Each submatrix $T^i_{m+1,j}$ is replaced by the Boolean sum of its columns [see remark below (12)]. Finally $T^i_{m+1,m+1}$ is left unaltered. Then we have, as in simple equivalence the following theorem.

Theorem 13

The machine obtained by the reduction process above (for multiply equivalent states) is synchronous and is equivalent to the original machine under Moore's definition.

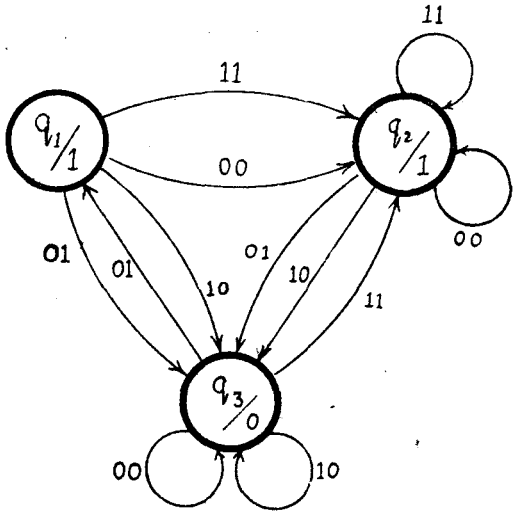


Fig. 3—Example for simple equivalence.

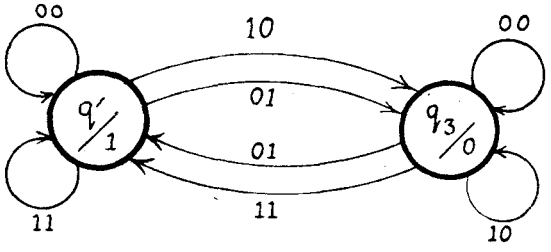


Fig. 4—Reduced machine of Example 1.

Since the reduced machine is synchronous, one can again apply the reduction process. We may also remark here that the simple and multiple equivalence reductions may be intermixed arbitrarily. Also, if one were to identify only a proper subset of the states in an equivalence class, the remaining states, together with the state obtained by reduction, still form an equivalence class. Further, an identification of sets in equivalence classes cannot generate new sets of equivalence classes[2] in a "completely specified machine," to use Huffman's terminology. Therefore, the machine obtained by an arbitrary sequence of reductions, provided only that it contains no equivalent states, will be unique. Thus:

Theorem 14

The reduced machine M, obtained by the reduction process, if it contains no equivalent states, is synchronous and unique (up to an isomorphism).

This theorem, which is similar to results obtained by Mealy[5] and Aufenkamp and Hohn,[4] was provable only because of our more restricted definition of equivalence. Moore's[1] general definition permits him to prove such a theorem only for strongly connected machines.

Example 2

As an example of the reduction process for multiple equivalence we consider the state diagram of Fig. 5.

$$T^A = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \end{array} \left[\begin{array}{ccc|cc} q_1 & q_2 & q_3 & q_4 & q_5 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

$$T^B = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \end{array} \left[\begin{array}{ccc|cc} q_1 & q_2 & q_3 & q_4 & q_5 \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{array}\right].$$

It is evident that although q_1 has the same output as q_2 and q_3, the set of states (q_1, q_2, q_3) is not a simply equivalent set. If, however, we rearrange the rows and columns as q_2, q_3, q_4, q_5, q_1 and partition after q_3 and q_5, we get

$$T^A = \begin{array}{c} \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_1 \end{array} \left[\begin{array}{ccc|cc} q_2 & q_3 & q_4 & q_5 & q_1 \\ \hline 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$T^B = \begin{array}{c} \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_1 \end{array} \left[\begin{array}{ccc|cc} q_2 & q_3 & q_4 & q_5 & q_1 \\ \hline 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \end{array}\right].$$

With this partitioning we see that the sets of states $\{q_2, q_3\}$ and $\{q_4, q_5\}$ are multiply equivalent by Theorem 11 and neither pair is simply equivalent. The reduction process now yields:

$$T^A = \begin{array}{c} \\ q_2' \\ q_4' \\ q_1 \end{array} \left[\begin{array}{ccc} q_2' & q_4' & q_1 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{array}\right] \qquad T^B = \begin{array}{c} \\ q_2' \\ q_4' \\ q_1 \end{array} \left[\begin{array}{ccc} q_2' & q_4' & q_1 \\ \hline 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right].$$

The state diagram of the reduced machine is given in Fig. 6.

VI. Strongly Connected Machines

The concept of a strongly connected machine was introduced by Moore[1] and plays a fundamental role in Moore's theory. In terms of the state diagram, we may define strong connectedness as follows:

The state diagram (or in general a directed graph) is *strongly connected* if for every ordered pair of states a and b (vertices for the general case) there exists a path from a to b in which the orientations of the edges agree with the orientation of the path (in other words an oriented path or a "bahn" in Koenig's[10] terminology).

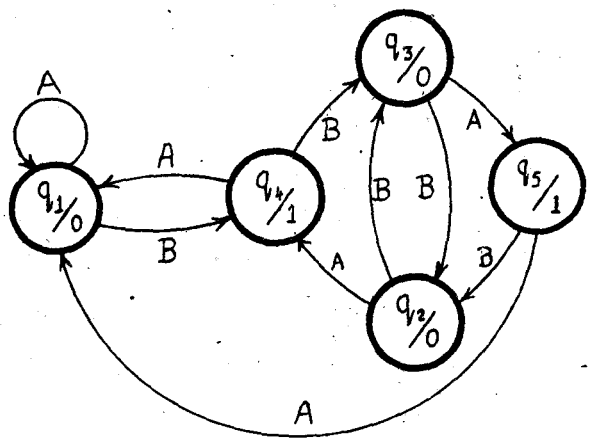


Fig. 5—Example for multiple equivalence.

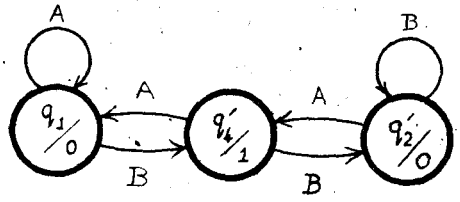


Fig. 6—Reduced machine of Example 2.

In order to prove theorems about strongly connected machines similar to Moore's theorems, we first need to characterize strong connectedness in terms of transition matrices. Since strong connectedness is equivalent to the existence of input sequences which take any given state a into any given state b; since the length of the longest proper path equals the number of states, we can state this characterization as follows.

Theorem 15

Let $T^1, T^2, \cdots, T^k$ be the transition matrices of the machine M containing n states. Then M is strongly connected if and only if

$$\sum_{i=1}^{k} T^i + \sum_{i,j=1}^{k} T^i T^j + \cdots + \sum_{i,j=1}^{k} T^{i_1} T^{i_2} \cdots T^{i_n} = E \quad (14)$$

where the sums are Boolean and E is the *universal matrix*, $E = [e_{ij}]$, $e_{ij} = 1$, all i, j.

This result is true since the nonzero entries in $T^i T^j \cdots T^p$ correspond to the paths (proper and intersecting) with edge sequence $i, j, \cdots, p$ and so all the proper paths are accounted for in (14). (When an intersecting path connects a to b, there is also a shorter proper path from a to b.[10])

Theorem 16

If a machine M is strongly connected and contains m simply equivalent states, then the reduced machine

[10] D. Koenig, "Theorie der Graphen," Chelsea Publishing Co., New York, N. Y.; 1950.

obtained by the procedure of Section IV is also strongly connected.

Proof: This fundamental theorem is known and was stated and proved by Moore.[1] The proof based on transition matrices is much too long to be given here. We shall therefore be satisfied with an outline of the matrix proof.

Let T^i and S^i stand for the transition matrices of the original and reduced machines, which are assumed to be partitioned as in (11) and (12), respectively. The proof consists of proving, by induction on the number of factors, that the properties,

1) $S^i_{22} = T^i_{22}$,
2) S^i_{12} = one of the identical rows of T^i_{12},
3) S^i_{21} = Σ columns of T^i_{21} [see comment below (12)],
4) $S^i_{11} = 0$ if T^i_{11} is a zero matrix, $S^i_{11} = 1$ otherwise,

are all invariant under matrix multiplication. Several cases have to be considered depending on the location of the nonzero entries in T^i which makes the proof long. Once this invariance is established it follows that the S^i's satisfy (14) whenever the T^i's do, thus making the reduced machine strongly connected.

A similar method of proof establishes the next theorem.

Theorem 17

If machine M is strongly connected and contains multiply equivalent states, then the reduced machine obtained by the reduction process of Section IV is also strongly connected.

Finally, Theorems 16 and 17 can be combined to yield Moore's result.[1]

Theorem 18

If a machine M is strongly connected, then the corresponding reduced machine containing no equivalent states is strongly connected, synchronous, and unique (to within an isomorphism).

VI. Concluding Remarks

Although no startlingly new results have been obtained using transition matrices, they seem to bring in a new point of view to the theory of sequential machines, inasmuch as several of the interesting properties of sequential machines are invariants of the transition matrix (under matrix multiplication). Thus is posed the problem of finding whether the transition matrices have any other invariant properties.

In order to bring about this closer correspondence between the transition matrix and the behavior of the sequential machine we have had to introduce an unconventional concept of equivalence. Simple and multiple equivalences, as defined here, are not binary relatives as is conventional, but are properties associated with sets.

However, the concepts of simple and multiple equivalence as defined here can be compared with the reduction procedures defined by Huffman,[2] Mealy,[5] and Aufenkamp and Hohn.[4] This comparison is possible only under the assumption that the machine is completely specified and under suitable interpretations of states. Thus we observe, quite obviously, that simple equivalence is the same concept as mergability of two rows of the flow table[2] and Rule III of Mealy.[5] Multiple equivalence on the other hand is analogous to rule II of Mealy[5] and the equivalence argument of Huffman.[3] To see the relationship of multiple equivalence to the Aufenkamp-Hohn theory[4] we have to construct the connection matrix. Aufenkamp and Hohn partition the connection matrix further than we have done in Theorem 10. Namely, each of the states in S_{m+1} is assigned a separate row and column in the partitioned matrix. Once this is done, we see that the property P of Theorem 10 is the same as the Aufenkamp-Hohn condition[11] that each submatrix be a 1-matrix. The invariance of property P under matrix multiplication is the same as Theorem 1 of Aufenkamp and Hohn.[12]

Thus, with completely specified machines, the reduced machine obtained by using the procedures given here would agree with those obtained by Huffman, Mealy, and Aufenkamp and Hohn. All of these would agree with Moore's scheme for strongly connected machines and for machines with a natural initial or cleared state (see remarks at the beginning of Section III).

The paper has been concerned exclusively with synchronous machines and so one might ask whether the theory as formulated can be extended to include asynchronous machines as well. There are two possible methods in which this extension can be accomplished. In the first method we merely write down the transition matrix of the asynchronous machine from the definition. If we did, we would find several rows of zeros in the transition matrix. In fact, whenever column j contains a 1, row j is a zero row. Such zero rows would be reflected in all the products, output vectors and state vectors, showing that an unpermissible sequence of inputs has been applied. It may be seen that many of the theorems as stated here would apply with minor changes in wording to take care of the zero rows. However it is more elegant (from the theoretician's point of view) and more natural, to reduce the case of the asynchronous machine to that of the synchronous one by suitably modifying the state diagram. Since an asynchronous machine recognizes only changes in the input, one might consider an input following itself as an input that has not changed. Thus we can add loops at each state corresponding to the inputs that bring the machine to that particular state.[13] In terms of the tran-

[11] Aufenkamp and Hohn, *op. cit.*, see theorem 2.

[12] This fact is not particularly surprising since the work of Aufenkamp and Hohn was inspired by a preliminary draft of the present paper.

[13] If we draw a Moore state diagram for an asychronous machine, considering the stable states (circled entries of Huffman) as the states of the machine, we would obtain precisely this diagram; in this sense the procedure is natural rather than a force fit.

sition matrices, whenever we have a zero row, we insert a 1 on the main diagonal. It is seen that the new machine, obtained by such a process, is synchronous and is restrictedly equivalent to the original machine. That is, as long as we apply only those sequences of inputs as are permissible sequences for the original machine, the two machines produce the same output sequence. Now, of course, all the results of this paper apply to this modified machine. It is easily seen that the loops remain loops through all the reduction processes. There will be no additional loops introduced; for, as we can easily observe, these additional loops appear when some input permutes the equivalent states, which is impossible if the original machine is asynchronous. Thus we can remove the loops after the reduction, making the machine once again asynchronous, and thus extend Theorems 12–14 to completely specified asynchronous machines (which have completely filled flow tables).

We might make, in conclusion, one small claim in favor of the transition matrix. Transition matrices provide a more formal language for proofs and are thus free of the semantic problems involved in sequential machine theory.

Hazards and Delays in Asynchronous Sequential Switching Circuits*

S. H. UNGER†

I. Introduction

THIS paper is concerned with the class of sequential switching circuits which has been treated by Huffman, Caldwell, and others.[1-4] We shall not discuss clocked systems. The reader will be expected to be familiar with the essential ideas contained in these references, and so no effort will be made to review the material presented there.

Switching circuits can be physically realized with a wide variety of devices, ranging from electromechanical relays to transistors and cryotrons. The problems that we shall attack, and our results will not depend on the nature of the devices used. However, it will sometimes be convenient to think in terms of specific circuits, and in such cases we will refer to gate-type logic, which can be realized with diodes or transistors.

An inherent property of all physical systems is delay. Signals are never transmitted instantaneously, and devices never react in zero time, but nevertheless it is frequently possible to ignore relatively small (or sometimes even large) delays when analyzing or designing systems. In the case of sequential switching circuits, early studies took into account only certain first-order effects of delays. Other effects were generally swamped out by the deliberate insertion of delay elements at key points in the circuits. The large delays inherent in certain devices, such as relays, served, in a fortuitous manner, to prevent malfunctioning that might otherwise have been caused by delays due to second-order effects such as stray reactances. However, modern electronic components operate at such high speeds that the blanketing effect due to inherent delays may be lost, and the deliberate introduction of delay elements becomes necessary.

This may entail an added expense and, when great efforts are being made to develop faster components, it seems somewhat incongruous to have to put back some of the delay that was so painstakingly removed. It is therefore desirable to understand the nature of the troubles caused by unwanted delays. In particular it might often be important to know how to combat these troubles with a minimum amount of additional delay elements.

We shall demonstrate here that a certain class of sequential functions (which we shall specify precisely) can always be realized physically in a trouble-free manner without introducing delay elements. It will also be proved that circuits corresponding to all other functions will be subject to malfunctioning due to unwanted delays if no delay elements are specified in the design. Finally, a proof will be presented that any function can be realized in a trouble-free manner by a circuit containing just one delay element. We shall also indicate some simple tech-

* Manuscript received by the PGCT, August 19, 1958. This paper is based on an Sc. D. dissertation, Dept. Elec. Eng., M.I.T., Cambridge, Mass.; April, 1957. The work was performed at the Res. Lab. of Electronics, M.I.T., and was supported in part by the U. S. Army (Signal Corps), the U. S. Air Force (Office of Sci. Res., Air Res. and Dev. Command), and the U. S. Navy (Office of Naval Res.).

† Bell Telephone Labs., Inc., Whippany, N. J.

[1] D. A. Huffman, "The synthesis of sequential switching circuits," *J. Franklin Inst.*, vol. 257, pp. 161–190, 275–303; March and April, 1954.

[2] S. H. Caldwell, "Switching Circuits and Logical Design," McGraw-Hill Book Co., Inc., New York, N. Y.; 1958.

[3] D. A. Huffman, "Design of hazard-free switching circuits," *J. Assoc. Computing Machinery*, vol. 4, pp. 47–62; January, 1957.

[4] D. A. Huffman, "A Study of the Memory Requirements of Sequential Switching Circuits," Res. Lab. of Electronics, M.I.T., Cambridge, Mass., Tech. Rep. No. 293; March, 1955.

niques that might prove useful in designing circuits with few or no delay elements for functions not so pathological as to call for the all-powerful methods used to prove the theorems for the most general cases.

In the next section we shall describe more precisely the kind of functions and problems we are going to study, and define a number of concepts and terms which will be employed in the main body of the paper.

The material presented here is based on a report by the author[5] which deals somewhat more extensively with some of the topics.

II. Basic Concepts and Terminology

A. Systems Under Consideration

Asynchronous, level-type sequential switching circuits are characterized by the fact that the signal at each input terminal is at all times in one of two distinct states—zero or one—except for brief intervals during which a transition from one state to the other is occurring. Such input changes may be made to occur at any time, subject only to the restriction that a minimum time interval, determined by the reaction times of the components, must separate successive changes.

The discussion here will be restricted to sequential functions in which no single input change is required to produce more than one change at any one output terminal. In flow table terminology (which will be presented shortly) this is equivalent to saying that if an uncircled j appears in any column of the flow table, there must also be a circled j in that same column and the output states must be the same for both total states.

Unless otherwise stated, it will be assumed that only one input variable at a time is permitted to change.

B. Flow Tables

The terminal characteristics of sequential switching circuits (sequential functions) are described by flow tables such as Table I, where the columns correspond to *input states*, that is, states of the input variables x_1 and x_2. The rows represent *internal states* of the system and the cells of the table represent *total states*. Entries in the centers of the cells indicate the next internal state, which in the case of circled entries is the same as the present internal state. These are the stable states. *Output states* for each total state are specified by the entries in the upper right corners of the cells.

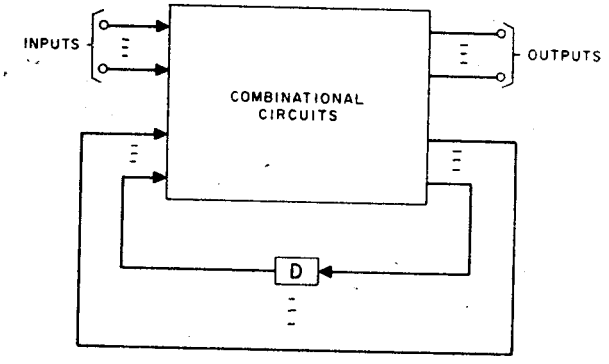
TABLE I

	$x_1 x_2$ 00	01	11	10
1	①^{00}	2^{10}	①^{01}	4^{01}
2	1^{00}	②^{10}	②^{01}	3^{10}
3	4^{11}	③^{00}	1^{01}	③^{10}
4	④^{11}	3^{00}	2^{01}	④^{01}

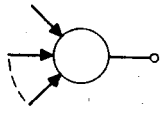
We shall often be focusing our attention on the effects of changes in a single input variable. In such cases it will be convenient to refer to the input state as being in column L (left) or R (right) and total states will be described as L3 or R5, meaning, for example, the state corresponding to the left-hand member of the pair of columns under discussion and the third row or the right column, fifth row, respectively.

C. Circuit Models and Flow Matrices

The circuits that will be considered here will be in the form shown in Fig. 1(a). All signals are binary valued,



(a)



(b)

Fig. 1.

and the box labeled "combinational circuits" contains no feedback paths. The elements in the box are all of the type shown in Fig. 1(b), that is, devices with one output terminal whose signal is a combinational function of the signals at the one or more input terminals. The only other components allowed in our model are delay elements.

Those delays deliberately introduced by the designer are defined as *delay elements* and appear only in the external feedback branches of Fig. 1(a). It is assumed that every circuit lead, including those inside the box may contain a *stray delay*, defined as a delay not under the control of the designer.

Any sequential switching circuit composed of the type of elements mentioned here can be represented in the form of Fig. 1(a) (but not always uniquely).

The external feedback branches in Fig. 1(a) will be

[5] S. H. Unger, "A Study of Asynchronous Sequential Switching Circuits," Res. Lab. of Electronics, M.I.T., Cambridge, Mass., Tech. Rep. No. 320; April, 1957.

called "state-branches" and are defined as follows. A set of branches of a sequential circuit is a *state-branch set* (and its members are *state-branches*) if the set contains all delay elements in the circuit and if at least one member is included in every feedback loop. Such a set is obviously not generally unique. The signal at the output end of each state-branch is defined as a *state-variable* and will be represented by y_i, for the ith state-branch. The *y-state* of a circuit at any given time is given by the set of values assumed by all of the state-variables of the circuit at that time. *State-variable excitations* are defined as the signals at the input ends of the state-branches and are represented by Y_i's. They may differ from the y_i's in cases where the branches contain delays.

The bridge between the flow table, describing the external characteristics of the circuit and the detailed circuit, is the *flow matrix*. Each row of the flow table (internal state) is associated with one or more y-states, defined as *row-sets*. The flow matrix of Table II illustrates such a row assignment for the flow table of Table I. Each row set is comprised of two y-states. Given a flow matrix, a terminal description of the corresponding combinational circuit box can be obtained in the form of a Y-matrix. This is a table of combinations giving the Y_i values as functions of the y states and can be obtained directly from the flow matrix.[1,2] Table III is the Y matrix for Table II.

TABLE II

	$x_1 x_2$ 00	01	11	10	y_1	y_2	y_3
1	①	2	①	4	0 1	0 1	0 1
2	1	②	②	3	0 1	0 1	1 0
3	4	③	1	③	0 1	1 0	1 0
4	④	3	2	④	0 1	1 0	0 1

TABLE III

$x_1 x_2$ 00	01	11	10	y_1	y_2	y_3
000	001	000	010	0	0	0
111	110	111	101	1	1	1
000	001	001	011	0	0	1
111	110	110	100	1	1	0
010	011	111	011	0	1	1
101	100	000	100	1	0	0
010	011	110	010	0	1	0
101	100	001	101	1	0	1

$Y_1 Y_2 Y_3$

D. Hazards

A circuit will be defined as *proper* if, for any $\epsilon > 0$ there exists a δ such that if all stray delay values are less than ϵ and if all delay element values are greater than δ, then the circuit will operate in accordance with the given flow table, provided that successive input changes are separated by some minimum finite time interval m, which is a function of ϵ and the *maximum* delay element value. Roughly speaking, a circuit is proper if it operates in a consistent manner despite variations in all delay values, provided that a suitable margin is maintained between the values of stray delay and delay element magnitudes and provided that the circuit is allowed to settle down after each input change before the next input change is permitted to occur.

(The nature of the relationships among the various bounds for a proper circuit may be surmised from the following statements: A conservative value of δ would be the maximum time for a signal to propagate from any input terminal to any output terminal of the portion of the circuit labeled "combinational circuit" in the model of Fig. 1(a), assuming that a stray delay of magnitude ϵ is placed in every branch of the circuit. If β is the maximum delay element value in the circuit, then let L_i be the sum of all delay values in the ith loop of the circuit when all delay element values are set at β and all stray delays are set at ϵ, and designate L as the maximum of the L_i's. Let T be the maximum number of y-states, including the final one, traversed during any transition in the Y-matrix of the circuit as a result of any one input change. Then LT will be a safe value for m. Note that the bounds given here are not the tightest ones possible.) If a circuit is not proper in the above sense then one or more hazard conditions will be said to exist.

Two types of hazards will be distinguished. A *transient hazard* is present if momentary false output signals occur at one or more output terminals following certain changes in the input state. If it is possible for a circuit to enter the wrong internal state after certain input changes, then a *steady-state hazard* will be said to exist.

Transient hazards can exist in combinational circuits, but Huffman has shown that they can always be avoided by appropriate design techniques.[3] Henceforth we shall assume that all of the combinational functions which we must realize will be designed in a *hazard-free* manner. (This is possible in general only when just one input at a time is permitted to change.)

Another form in which a hazard can appear in a sequential circuit is a *critical race condition*.[1,2,4] This is a condition in a flow matrix where two or more state-variables become unstable simultaneously as a result of an input change and where the next stable state of the system depends on the order in which the unstable variables change. This source of hazards can be eliminated by design procedures described elsewhere[1,2,4] and will not be considered here.

Two types of delays will be defined. A *pure delay* is one which converts on input signal $f(t)$ into an output $f(t - D)$, where D is the value of the delay. An *inertial delay* (defined only in terms of binary signals) behaves like a pure delay, except that input changes persisting for a time less than D, the delay magnitude, are ignored. Thus, rapid signal fluctuations are filtered out by such a device, which is somewhat analogous to a low-pass filter. While few if any physical devices behave exactly in accordance with either of the above descriptions, many real components closely approximate pure or inertial delays. Transmission lines and certain LC ladder networks resemble pure delays, whereas electromechanical relays behave like inertial delays. Stray delays, generally dominated by wiring reactances, possess important inertial characteristics and we shall assume that they are representable as inertial delays. This assumption will be used in the proof of the theorem of Section IV.

A typical hazard situation is depicted in the circuit of Fig. 2. If we assume that the stray delays in the B_1 and B_2 branches are large compared to the delays in the other branches, then the circuit behavior will be correctly characterized by the flow matrix shown in Table IV. (In fact, one way of assuring proper operation is to insert delay elements in B_1 and B_2.)

Suppose now that a relatively large delay appears in the branch labeled H (how large will become clear during the subsequent discussion). Then if, while the system is in the state labeled a in Table IV, x_2 is turned on (switched from zero to one) the next stable state will be in row 4 and not in row 2, as indicated in the flow matrix. This comes about as follows.

The switching of x_2 first changes to unity the output of the multiplier gate labeled M_4 in Fig. 2, and this switches on y_2, the output of A_2, which is connected to one input lead of M_2. The other input to this gate remains at unity since the delay in H prevents the x_2-change from having an immediate effect at this point. Therefore a one-signal appears at the output of M_2, penetrates through A_1, and turns on y_1. A feedback path through M_1 and A_1 now holds y_1 on independently of the output of M_2. A second effect of y_1 changing to unity is that a zero appears at the output of I_3, thus preventing anything but zeros from appearing at the outputs of M_3 and M_4. Now $y_1 = y_2 = 1$, and the system is in the state marked b in Table IV. But only the output of M_2 is holding y_2 on, and as soon as the effect of the original x_2-change penetrates I_2 and the delay in H, a zero appears at the output of M_2 causing the output of A_2 to go to zero, and the system is in row 4 as stated.

The possibility of such behavior could have been predicted from the structure of the flow matrix. When x_2 changes, y_2 becomes unstable and also switches. If the effects of the y_2-action penetrate to y_1 before the direct effect of the x_2-change (which is the case in our circuit), then as far as y_1 is concerned, the system state switches from a to c, where y_1 is unstable. This causes y_1 to switch to one and moves the internal state of the system to row 3 and, once this occurs, y_1 remains at unity value even after

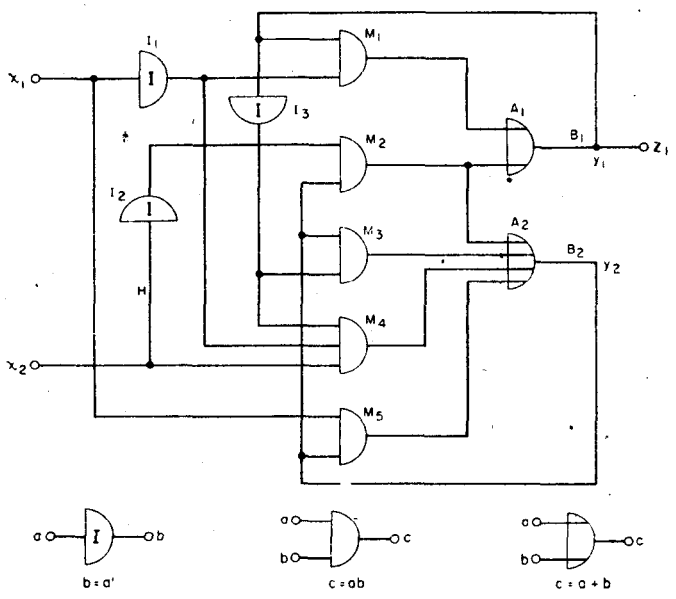


Fig. 2.

TABLE IV

	$x_1 x_2$ 00	01	11	10	y_1	y_2
1	a ①⁰	2	①⁰	①⁰	0	0
2	c 3	b ②⁰	②⁰	3	0	1
3	d ③¹	4	2	③¹	1	1
4	④¹	e ④¹	1	1	1	0

the effect of the x_2-change reaches it. Finally, since y_2 is unstable in state b, it changes again and the system settles down in state e.

We shall see later that such behavior will always be possible in circuits realizing functions corresponding to flow matrices such as Table IV, unless delay elements are used.

III. Functions Realizable Without Delay Elements

A. Definitions and Lemmas

Each definition and lemma in this section is numbered, and the first time the term or lemma is used in the main body of the text it will be italicized and the reference number will follow.

1. Definition—Essential Hazard: A sequential function contains an essential hazard if there exists a stable state S_0 and an input variable x such that, starting with the system in S_0 three consecutive changes in x bring the system to a state other than the one arrived at after the first change in x. The function corresponding to the matrix of Table IV in the previous section has three essential hazards. One starts in state a with x_2 as the relevant input variable (and was discussed in the example). A second has its

initial state labeled b, with x_2 as the input variable. Can you find the third?

2. Definition—Mod Sum: The mod sum of two positive integers a and b, written as $a(+)\ b$ is the arithmetic sum of those powers of two appearing in the binary representation of a or of b, but not both. For example, $3(+)7 = 4$ and $5(+)\ 3 = 6$. This operation is clearly associative and commutative, and $x(+)x = 0$ for any x.

3. Definition—Hamming Row-Set: A y-state will be said to belong to the Hamming row set S_i if, and only if, the q's determined by the parity relations (involving modulo-two addition).

$$q_0 = y_1 \oplus y_3 \oplus y_5 \oplus y_7$$
$$q_1 = y_2 \oplus y_3 \oplus y_6 \oplus y_7$$
$$q_2 = y_4 \oplus y_5 \oplus y_6 \oplus y_7$$

correspond to the digits of the number i written in binary form, that is if $i = q_2 2^2 + q_1 2^1 + q_0 2^0$. For example, 0110111 belongs to S_5. These sets, which can be generated from any set of $2^n - 1$ binary variables, are related to Hamming error correcting codes,[6] and were first applied to sequential switching circuits by Huffman.[4] They have the property that, given any state in S_i, changing $y_{i(+)j}$ converts the state to one belonging to S_j. For example, changing y_2 when the system is in state 0001101 (in set S_6) changes the state to 0101101 which is in S_4 [$6(+)4 = 2$]. Transitions from any set to any other set thus require a change in only one variable. It will sometimes be convenient to refer to $y_{i,j}$ as the variable separating the states assigned to rows i and j. Where S_i and S_j are Hamming sets assigned to rows i and j, respectively,

$$y_{i,j} = y_{i(+)j}.$$

4. Definition—d-trio: Three rows of a flow table r_1, r_2, and r_3 constitute a d-trio if, for some pair of input states L and R differing only in the value of one input variable, the next-state entries in L are r_1, r_3, and r_3 in rows r_1, r_2, and r_3, respectively, and the next-state entries in R are r_2 for all three rows. Part (d) of Table VI in the next section depicts a d-trio (and a glance at this illustration will no doubt be more edifying than the tongue twisting definition given above.)

5. Definition—d-Transition: A d-transition is the activity occurring as a result of an input change from L and R with the system originally in row r_1 of a d-trio.

6. Definition—Trap Row: An auxiliary row added to a flow matrix to guard against malfunctioning during a particular d-transition. The row-set assigned to the trap row for each d-trio corresponds to the y-states other than those assigned to the d-trio which might be seen during the d-transition due to the effects of stray delays. The next-state entries assigned to the trap rows assure proper termination of the d-transition.

[6] R. W. Hamming, "Error detecting and error correcting codes," *Bell Sys. Tech. J.*, vol. 29, pp. 147–160; April, 1950.

7. Lemma: If a, b, and c are three integers no two of which are equal, then there is no integer d such that $2^d = 2^a (+)2^b(+)2^c$.

Proof: a) From the definition of (+) (see Section III-A, 2) it is clear that $2^a(+)2^b(+)2^c = 2^a + 2^b + 2^c$ since each term in the sum is a unique power of two.

b) The sum $2^a + 2^b + 2^c$ can be written as a binary number with exactly *three* nonzero bits, one corresponding to each term.

c) If there existed an integer d such that $2^d = 2^a + 2^b + 2^c$, then the sum referred to in b) could also be written as a binary number with *one* nonzero bit corresponding to 2^d. But this is impossible since the binary form of any number is unique. Hence there cannot be any d such that $2^d = 2^a + 2^b + 2^c$. (Q.E.D.)

8. Lemma: If a, b, c, a^*, b^*, and c^* are positive integers with $a < b < c$ and $a^* < b^* < c^*$, then $2^a(+)2^b(+)2^c = 2^{a^*}(+)2^{b^*}(+)2^{c^*}$ implies that $a = a^*$, $b = b^*$, and $c = c^*$.

Proof: First note that, from the definition of the (+) operation (Section III-A, 2), $2^a + 2^b + 2^c = 2^{a^*} + 2^{b^*} + 2^{c^*}$. If we write the sums on both sides of the equation in binary form we can see that each term corresponds to one binary digit, so that each term on one side must be matched by a term on the other side, and hence the exponents must be equal. (Q.E.D.)

B. Theorem on Sequential Functions Without Essential Hazards

If a sequential function has no essential hazards (Section III-A, 1) *then it can always be realized by a circuit without delay elements, and this circuit will be free of steady-state hazards. Transient hazards may occur in some cases.*

There will be two parts to the proof. First we shall describe a method for assigning row-sets to the rows of an augmented version of the flow table describing the function, and then we shall demonstrate that this assignment will result in proper operation without reliance on delay elements. (It is assumed that hazard-free combinational circuits will be used to realize the flow matrix.)

Let us assign the *Hamming row-set* (Section III-A, 3) S_{2^i} to row $i (i = 1, 2, 3, \cdots)$. For every *d-trio* (Section III-A, 4) with rows a, b, and c form an auxiliary row of the flow matrix [called a *trap row* (Section III-A, 6)] and assign to it the row-set S_t, where $t = 2^a(+)2^b(+)2^c = 2^a + 2^b + 2^c$. [See definition in Section III-A, 2 for meaning of (+) operation.] From our first lemma (Section III-A, 7) we note that t cannot be expressed in the form 2^d (d an integer), so S_t could not have been assigned to any of the original rows of the table. Our second lemma (Section III-A, 8) implies that every d-trio will have a trap row with a unique row-set assigned to it since $2^a(+)2^b(+)2^c = 2^{a^*}(+)2^{b^*}(+)2^{c^*}$ implies $a = a^*$, $b = b^*$, and $c = c^*$.

The entries in the trap rows of the matrix are filled as follows. For the input states in which the *d-transitions* (Section III-A, 5) of the corresponding d-trios terminate we fill in numbers corresponding to the final states of the d-transitions. [That is the state labeled 2 in Table VI (d).]