PROGRESS REPORT

Structure Theory for the
Realization of Finite State Automata

NASA Grant NGR-44-012-049

Period 1 November 1966 - 30 April 1967

Principal Investigator: C. L. Coates

May 10, 1967

PROGRESS REPORT

Structure Theory for the Realization of Finite State Automata
NASA Grant NGR-44-012-049
Period 1 November 1966 - 30 April 1967

I. SUMMARY

The work initiated under this grant during the first six month period falls within two catagories. One is concerned with controlling the structure of synchronous realizations of finite state automata (sequential machines) when the storage elements of the machine are flip-flops. Basically this is the problem of assigning binary codes to the state, input and output alphabets in such a way as to control the extent to which the flip-flop inputs depend upon the contents of the other flip-flop elements. The results are directed toward determining when realizations can be fabricated by networks of smaller machines. This not only reduces the number of components but allows some control of how the components are interconnected.

The initial work (Ref. 1) on this problem was completed prior to the initiation of this grant. During the period of this report, investigations have continued toward the objective of determining how the feedback can be controlled when the storage elements are flip-flops. The results of this study have not been completed but will be reported at the end of the next period.

Related to the problem of controlling the structure by which machines are realized is the problem of controlling machine errors. In this regard Hartmanis and Stearns (Ref. 2, 3 and 4) defined the concept of inessential errors. Basically these are state errors within the machine that occur because of a temporary malfunction. Although they remain as state errors they produce only a finite number of output errors even for infinitely long input sequences. Although Hartmanis and Stearns characterized inessential errors in terms of a state partition Π_E, they did not provide a procedure for calculating this partition. In fact, they showed that it was not determined by state partitions with either the pair or the substitution property relations.

During the current reporting period we have defined a procedure for determining inessential errors. An initial draft of the results of this study is included in Part II.

The second category of investigations is concerned with asynchronous realizations of finite state automata that consist of a combinational logic network with feedback. Included in this study is the use of threshold logic gates.

The problems associated with asynchronous realizations are those of assigning state, input and output codes such that hazard and race conditions within the realization are minimized. During the current reporting period work was completed on the study of hazards in threshold logic and a means for obtaining realizations that are free of logic hazards. The results of this study is included in Part III.

References

1. C. Harlow and C. L. Coates, "On the Structure of Realizations Using Flip-Flop Memory Elements," Information and Control, February 1967, pp. 159-174.

2. J. Hartmanis and R. E. Stearns, (1966), Algebraic Structure Theory at Sequential Machines, Prentice-Hall.

3. J. Hartmanis and R. E. Stearns, (1963),"A Study of Feedback and Errors in Sequential Machines", IRE Trans. on Electronic Computers, EC-12.

4. J. Hartmanis, (1961), "On the State Assignment Problem for Sequential Machines, I," IRE Trans. on Electronic Computers, EC-10.

II. ERRORS IN SEQUENTIAL MACHINES

In papers by Hartmanis and Stearns (Ref. 1 and 2) the concept of an inessential error is defined and some of the properties of inessential errors are derived. An error partition Π_E is defined and investigated. However, it is now shown in these papers how to calculate Π_E. The purpose of this paper is to give an algorithm for determining Π_E. First we shall review some concepts that are given in Ref. 1.

Definition 1

A Moore type sequential machine is a quintuple $M = (\{s\}, \{x\}, 0, \delta, \lambda)$, $\{s\}$ is a finite set called the set of states, $\{x\}$ is the set of inputs, 0 is the set of outputs, $\delta: \{s\} \times \{x\} \to \{s\}$ and $\lambda: \{s\} \to 0$.

In this paper the only machines we will consider are Moore machines which are completely specified; that is, the domain of δ is all of $\{s\} \times \{x\}$. The next definition extends the function δ to all sequences of inputs.

Definition 2

Let M be a sequential machine. Let $\{x_i\}_1^j$ be a sequence of inputs of length $j \geq 0$ we define a function $\bar{\delta}$ as follows. Let $a \epsilon \{s\}$ $\bar{\delta}(a, \{x_i\}_1^j) = a$ if $j = 0$, $\bar{\delta}(a, \{x_i\}_1^1) = \delta(a, x_1)$ if $j = 1$. In general $\bar{\delta}(a, \{x_i\}_1^j) = \delta(\bar{\delta}(a, \{x_i\}_1^{j-1}), x_j)$ for every $j \geq 2$.

Definition 3

Let M be a sequential machine. Let $j \geq 0$, $\{x_i\}_1^j$ be a sequence of inputs and let $a \epsilon \{s\}$. Then $\bar{\lambda}(a, \{x_i\}_1^j) = \lambda(\bar{\delta}(a, \{x_i\}_1^j))$.

Definition 4

Let M be a sequential machine.

i) Let τ be a partition on $\{s\}$. Then if $a, b \epsilon \{s\}$ $\tau[a] = \tau[b]$ means a and b are in the same set, sometimes called a block, of τ.

Example 1. If $\{s\} = \{1,2,3,4,5\}$ and $\tau = (\overline{1,2};\overline{3,4,5})$ then $\tau[3] = \tau[4] = \tau[5]$ and $\tau[2] \neq \tau[3]$.

ii) An S.P. partition τ is a partition in $\{s\}$ such that for every $a, b \epsilon \{s\}$ with $\tau[a] = \tau[b]$ then $\tau[\delta(a,x)] = \tau[\delta(b,x)]$ for every input x.

Definition 5

Let M be a machine. An error is a partition τ_{ab} where $a, b \epsilon \{s\}$ and a and b are the only two states in the same block of τ_{ab}.

Example 2. If $\{s\} = \{1,2,3,4,5\}$ then $\tau_{13} = (\overline{1,3};\overline{2};\overline{4};\overline{5})$.

Definition 6

An error τ_{ab} is an inessential error iff for every input sequence $\{x_i\}_1^\infty$ there exists a finite set $\Delta \leq \{1,2,\ldots\}$ such that $\bar{\lambda}(a,\{x_i\}_1^k) \neq \bar{\lambda}(a,\{x_i\}_1^k)$ if and only if $k \epsilon \Delta$.

The next result which is proved in Ref. 1 verifies the existence of Π_E the partition which we want to determine.

Result 1 (Theorem)

Let M be a machine. There exists an S.P. partition Π_E such that if $\tau_{ab} \leq \Pi_E$ then τ_{ab} is an inessential error and conversely if τ_{ab} is an inessential error then $\tau_{ab} \leq \Pi_E$.

We conclude the introductory concepts with a brief discussion of set systems. It turns out that the set system is the principal concept to be used in determining Π_E.

Definition 7

A set system on $\{s\}$ is a collection $\rho = \{A_i | i \epsilon \Lambda\}$ where Λ is a finite index set and $A_i \leq \{s\}$ for every $i \epsilon \Lambda$. Also

i) $\underset{\Lambda}{\cup} A_i = \{s\}$

ii) $A_i \geq A_j$ implies that $A_i = A_j$ for every $i, j \epsilon \Lambda$.

Definition 8

Let M be a Moore machine

i) Let $E = \{[a,b] \mid a,b \epsilon \{s\}$ and $\lambda(a) \neq \lambda(b)\}$

ii) Let $K = \{[a,b] \mid a,b \epsilon \{s\}$ with $a \neq b\}$

Definition 9

If τ is a set system in $\{s\}$ then we define

i) $m_{ss}(\tau) = \{C \leq \{s\}$ } $C = \delta(A,x)$ where $A \epsilon \tau$ and $x \epsilon \{x\}$ and C is not less than $\delta(B,x')$ for any $B \epsilon \tau$ and $x' \epsilon \{x\}$ }.

Note that $\delta(A,x) = \{b \mid b = \delta(a,x)$ for some $a \epsilon A\}$ and that $m_{ss}(\tau)$ is a set system on $\{s\}$.

ii) $m_{ss}^{i+1}(\tau) = m_{ss}(m_{ss}^{i}(\tau))$

Result 2

If τ is an S.P. set system on $\{s\}$ then $m_{ss}(\tau) \leq \tau$ and $m_{ss}^{i+1}(\tau) \leq m_{ss}^{i}(\tau)$ for every $i \geq 1$.

Proof:

Since τ has S.P. for every $A \epsilon \tau$ and $x \leq \{x\}$ $C = \delta(A,x) \leq B$ for some $B \epsilon \tau$. Hence if $C \epsilon\ m_{ss}(\tau)$ then $C \leq B$ a set of τ. This implies $m_{ss}(\tau) \leq \tau$. The second part of the result is easily proved by induction.

Figure 1 contains an example of these concepts. In Figure 1 τ has S.P.and the m_{ss} operators on τ are computed. For this example $E = \{[1,2],[1,4],[2,3],[2,5],[3,4],[4,5]\}$.

	0	1	λ
1	2	4	0
2	2	5	1
3	3	1	0
4	4	2	1
5	1	4	0

$$\tau = (\overline{1,2,4,5};\overline{3})$$
$$m_{ss}^{1}(\tau) = (\overline{1,2,4};\overline{2,4,5})$$
$$m_{ss}^{2}(\tau) = (\overline{1,2,4};\overline{2,4,5})$$

Figure 1. Machine A

Result 3 which follows is one of the principal results of this paper in that it gives a necessary condition that Π_E must satisfy. When we write $[a,b] = [c,d]$ this means $a = c$ and $b = d$ or $a = d$ and $b = c$.

Result 3 (Theorem)

Let M be a q state machine where $q \geq 2$. Let $p = \binom{q}{2}$. If τ is a partition of $\{s\}$ such that $\tau \leq \Pi_E$ and τ has S.P. then for every $[a,b]\ \epsilon E$ such that $\tau[a] = \tau[b]$ we have that $m_{ss}^{i}(\tau_{ab})\,[a] \neq m_{ss}^{i}(\tau_{ab})\,[b]$ for every i such that $1 \leq i \leq p$.

Proof:

Suppose there exists $[a,b]\ \epsilon E$ such that $\tau[a] = \tau[b]$ and an integer i with $1 \leq i \leq p$ where $m_{ss}^{i}(\tau_{ab})\,[a] = m_{ss}^{i}(\tau_{ab})\,[a]$. This implies that there exists a_{i-1}, b_{i-1}, and an x_i such that $m_{ss}^{i-1}(\tau_{ab})\,[a_{i-1}] = m_{ss}^{i-1}(\tau_{ab})\,[b_{i-1}]$ and $[\delta(a_{i-1},x_i),\ \delta(b_{i-1},x_i)] = [a,b]$. Continue in this fashion until we have a_1, b_1 such that $m_{ss}^{1}(\tau_{ab})\,[a_1] = m_{ss}^{1}(\tau_{ab})[b_1]$. This implies that there exists x_1 such that $[a_1,b_1] = [\delta(a,x_1),\ \delta(b,x_1)]$. Thus we have a sequence $\{x_j\}_1^i$ such that $[\bar{\delta}(a,\{x_j\}_1^i),\ \bar{\delta}(b,\{x_j\}_1^i)] = [a,b]$. Form the sequence $\{y_i\}_1^\infty$ as follows.

$y_n = x_n$ if $1 \leq n \leq i$

$y_n = x_{n-i}$ if $n > i$

Let $\Lambda = \{\, r \mid r = ni \ \ n$ a positive integer$\}$. If $k \in \Lambda$ then $k = ni$ which implies $[\bar{\delta}(a,\{x_j\}_1^{ni}),\ \bar{\delta}(b,\{x_j\}^{ni})] = [a,b]$. This in turn implies that $\bar{\lambda}(a,\{x_i\}_1^{ni}) = \lambda(a)$ and $\bar{\lambda}(b,\{x_i\}_1^{ni}) = \lambda(b)$. Since $[a,b] \in E$ this implies $\bar{\lambda}(a,\{x_i\}_1^k) \neq \bar{\lambda}(b,\{x_i\}_1^k)$ for every $k \in \Lambda$. Since Λ is an infinite set this means τ_{ab} is not an inessential error. Thus from Result 1 τ_{ab} is not less than Π_E. But since $\tau[a] = \tau[b]$ and $\Pi_E \geq \tau$ this is a contradiction which proves the theorem. ||

The remainder of this paper will be concerned with proving the converse of Result 3. The proof of the converse is fairly involved. For this reason we will isolate certain parts of the proof with the following lemmas.

Result 4 (Lemma)

Let M be a machine. If $a,b \in \{s\}$ and $\Pi_E[a] \neq \Pi_E[b]$ then there exists a sequence of inputs $\{x_i\}_1^\infty$ and $[a_o,b_o] \in E$ such that $[a_o,b_o] = [\delta(a,\{x_i\}_1^k),\ \delta(b,\{x_i\}_1^k)]$ for every $k \in J_o$ where J_o is an infinite subset of $\{1,2,\ldots\}$.

Proof:

Since $\Pi_E[a] \neq \Pi_E[b]$ we know that τ_{ab} is not an inessential error from Result 1. This implies there exists $\{x_i\}_1^\infty$ such that $\lambda(a, \{x_i\}_1^k) \neq \lambda(b, \{x_i\}_1^k)$ for every $k \in J'$ where J' is an infinite subset of $\{1, 2, \ldots\}$. Let $a_k = \bar{\delta}(a, \{x_i\}_1^k)$ and $b_k = \bar{\delta}(b, \{x_i\}_1^k)$. Let $J_{[c,d]} = \{k \in J' \mid [c,d] = [a_k, b_k]\}$. $J' \leq \bigcup_{[c,d] \in E} J_{[c,d]}$ since if $k \in J'$ this implies $[a_k, b_k] \in E$ or $[a_k, b_k] = [c,d] \in E$. This implies $k \in J_{[c,d]} \Rightarrow k \in \bigcup_{[c,d] \in E} J_{[c,d]}$. Since J' is infinite this means $J_{[a_o, b_o]} = J_o$ is infinite for some $[a_o, b_o] \in E$. Thus $[a_o, b_o] = [a_k, b_k] = [\delta(a, \{x_i\}_1^k), \delta(b, \{x_i\}_1^k)]$ for every $k \in J_o$ which is an infinite set. ||

Result 5 (Lemma)

Let M be a q state machine with $q \geq 2$. Let $p = \binom{q}{2}$. Further suppose there exists $\{x_i\}_1^k$ a sequence of inputs with $k \geq 1$ such that $[a_o, b_o] = [\delta(a, \{x_i\}_1^k), \delta(b, \{x_i\}_1^k)]$ where a_o, b_o, a, and b are states of M. Then there exists a sequence of inputs $\{y_i\}_1^\ell$ where $1 \leq \ell \leq p$ and $1 \leq \ell \leq k$ and $[a_o, b_o] = [\delta(a, \{y_i\}_1^\ell), \delta(b, \{y_i\}_1^\ell)]$.

Proof:

i) If $k \leq p$ then the theorem is satisfied. Suppose $k > p$. Let $B_n = \{[c,d] \mid c \neq d,\ [c,d] = [\delta(a, \{x_i\}_1^j), \delta(b, \{x_i\}_1^j)]$ for some j such that $0 \leq j \leq n\}$ for n such that $0 \leq n \leq k$. Clearly $B_{n+1} \geq B_n$. Also $B_{n+1} = B_n$ iff $0 \leq n \leq k-1$ and $[\delta(a, \{x_i\}_1^{n+1}), \delta(b, \{x_i\}_1^{n+1})] = [\delta(a, \{x_i\}_1^j), \delta(b, \{x_i\}_1^j)]$ for some j such that $0 \leq j \leq n$. Note that for every n $B_n \leq K$ and the cardinality of K is p. This implies that there exists integer r such that

$B_{r+1} = B_r$ where $0 \le r \le p-1$. Let $r_{\min}$ be the minimum such r and let $\ell_1 = r_{\min} + 1$. Then $1 \le \ell_1 \le p$ and $B_{\ell_1} = B_{\ell_1 - 1}$. This implies that there exists j_1 such that $0 \le j_1 \le \ell_1 - 1$ and

$[\delta(a, \{x_i\}_1^{j_1}), \delta(b, \{x_i\}_1^{j_1})] = [\delta(b, \{x_i\}_1^{\ell_1}), \delta(b, \{x_i\}_1^{\ell_1}].$

Note that $p-1 \ge \ell_1 - j_1 \ge 1$ and $k - (\ell_1 - j_1) > p - (\ell_1 - j_1) \ge j_1 \ge 0$. Let $k_1 = k - (\ell_1 - j_1)$. Then $k > k_1 \ge 1$. Form the sequence $\{x_{i,1}\}_1^{k_1}$ as follows. Let

$x_{i,1} = x_i$ if $1 \le i \le j_1$

$x_{i,1} = x_i + (\ell_1 - j_1)$ if $j_1 < i \le k_1$.

Show $[\delta(a, \{x_{i,1}\}_1^{k_1}), \delta(b, \{x_{i,1}\}_1^{k_1})] = [a_o, b_o]$. From the definition of $\{x_{i,1}\}_1^{k_1}$ $[\delta(a, \{x_{i,1}\}_1^{j_1}), \delta(b, \{x_{i,1}\}_1^{j_1})] = [\delta(a, \{x_i\}_1^{j_1}), \delta(b, \{x_i\}_1^{j_1}] = [\delta(a, \{x_i\}_1^{\ell_1}), \delta(b, \{x_i\}_1^{\ell_1})]$. Thus if $j_1 < r < k_1$ since for i such that $j_1 < i \le k_1$ $x_{i,1} = x_i + (\ell_1 - j_1)$ $[\delta(a, \{x_{i,1}\}_1^r), \delta(b, \{x_{i,1}\}_1^r)] =$

$[\delta(\delta(a, \{x_{i,1}\}_1^{j_1}), \{x_{i,1}\}_{j_1+1}^r), \delta(\delta(b, \{x_{i,1}\}_1^{j_1}), \{x_{i,1}\}_{j_1+1}^r)] =$

$[\delta(\delta(a, \{x_i\}_1^{\ell_1}), \{x_i\}_{\ell_1+1}^{r+(\ell_1-j_1)}), \delta(\delta(b, \{x_i\}_1^{\ell_1}), \{x_i\}_{\ell_1+1}^{r+(\ell_1-j_1)})] =$

$[\delta(a, \{x_i\}_1^{r+(\ell_1-j_1)}), \delta(b, \{x_i\}_1^{r+\ell_1-j_1}]$. If we let $r = k_1$ we get that

$[\delta(a, \{x_{i,1}\}_1^{k_1}), \delta(b, \{x_{i,1}\}_1^{k_1})] = [\delta(a, \{x_i\}_1^{k}), \delta(b, \{x_i\}_1^{k})] = [a_o, b_o]$.

Thus $\{x_{i,1}\}_1^{k_1}$ is a sequence of inputs such that $1 \le k_1 < k$ and $[\delta(a, \{x_{i,1}\}_1^{k_1}), \delta(b, \{x_{i,1}\}_1^{k_1})] = [a_o, b_o]$.

ii) If $k_1 \le p$ then the lemma is satisfied. If not repeat step i) and get a sequence $\{x_{i,2}\}_1^{k_2}$ where $1 \le k_2 < k_1$ and $[\delta(a, \{x_{i,2}\}_1^{k_2}),$ $\delta(b, \{x_{i,2}\}_1^{k_2})] = [a_o, b_o]$. Continue in this manner until for some k_j

we have $k_j \leq p$. Let $\ell = k_j$. Then the sequence $\{x_{i,j}\}_{i=1}^{k_j} = \{y_i\}_{i=1}^{\ell}$ satisfies the lemma. ||

The next result is the converse of Result 3. Its proof follows easily from the two preceeding lemmas.

Result 6 (Theorem)

Let M be a q state Moore Machine where $q \geq 2$. If τ is a S.P. partition on $\{s\}$ and if for every $[a,b] \epsilon E$ such that $\tau[a] = \tau[b]$ we have that $m_{ss}{}^{i}(\tau_{ab})\ [a] \neq m_{ss}{}^{i}(\tau_{ab})\ [b]$ for every i such that $1 \leq i \leq p$ then $\tau \leq \Pi_E$.

Proof:

Suppose $\tau \leq \Pi_E$. Then there exists a, b such that $\tau[a] = \tau[b]$ and $\Pi_E[a] \neq \Pi_E[b]$. From esult 4 this implies that there exists a sequence of inputs $\{x_i\}_1^{\infty}$ and $[a_o, b_o] \epsilon E$ such that $[a_o, b_o] = [\delta(a, \{x_i\}_1^k), \delta(b, \{x_i\}_1^k)]$ for every $k \epsilon J_o$ an infinite subset of $\{1, 2, 3, \ldots\}$. Let k_1 be the minimum element of J_o. This means that $m_{ss}{}^{k}(\tau)\ [a_o] = m_{ss}{}^{k}(\tau)\ [b]$ which implies $\tau[a_o] = \tau[b_o]$ since $\tau \geq m_{ss}{}^{k}(\tau)$ from Result 2. Let k_2 be the minimum element of $J_o - \{k_1\}$. Then $[a_o, b_o] = [\delta(a, \{x_i\}_1^{k2}), \delta(b, \{x_i\}_1^{k2})] = [\delta(\delta(a, \{x_i\}_1^{k1}), \{x_i\}_{k_1+1}^{k2}),$

$\delta(\delta(b, \{x_i\}_1^{k1}), \{x_i\}_{k_1+1}^{k2})] = [\delta(a_o, \{x_i\}_{k_1+1}^{k2}), \delta(b_o, \{x_i\}_{k_1+1}^{k2})]$. Thus we have a sequence $\{x_i\}_{k_1+1}^{k2}$ of length $k_2 - k_1$ such that $[a_o, b_o] = [\delta(a_o, \{x_i\}_{k_1+1}^{k2}), \delta(b_o, \{x_i\}_{k_1+1}^{k2})]$. From Result 5 there exists $\{y_i\}_1^{\ell}$

such that $[a_o, b_o] = [\delta(a_o, \{y_i\}_1^\ell), \delta(b_o, \{y_i\}_1^\ell)]$ and $1 \leq \ell \leq p$. This implies $[a_o, b_o] \in m_{ss}{}^{\ell}(\tau_{a_o b_o})$. Since $[a_o, b_o] \in E$ and $\tau[a_o] = \tau[b_o]$ this is a contradiction. ||

Results 3 and 6 imply the following theorem which is an algorithm for finding Π_E.

Result 7 (Theorem)

Let M be a q state machine with $q \geq 2$. Let $F = \{\tau \mid \tau$ satisfies i and ii below$\}$.

i) τ is an S.P. partition of $\{s\}$.

ii) For every $[a,b] \in E$ such that $\tau[a] = \tau[b]$ $m_{ss}{}^{i}(\tau_{ab})[a] \neq m_{ss}{}^{i}(\tau_{ab})[b]$ when $1 \leq i \leq p$. Then there is a largest partition in F and this partition is Π_E.

Proof:

From result 3 $\Pi_E \in F$. If $\tau \in F$ then $\tau \leq \Pi_E$ from 6. ||

We conclude this paper with some examples of Result 7. Consider machine A in Figure 1. The only S.P. partitions are those in the Figure. We compute F. Clearly 0 the zero partition is in F. Consider $\tau = (\overline{1,2,4,5};\overline{3})$. Note that $[1,2] \in E$ and $\tau[1] = \tau[2]$. $m_{ss}{}^{1}(\tau_{12}) = (\overline{1};\overline{2};\overline{3};\overline{4,5})$, $m_{ss}{}^{2}(\tau_{12}) = (\overline{1,4};\overline{2,4};\overline{3};\overline{5})$, $m_{ss}{}^{3}(\tau_{12}) = (\overline{2,4};\overline{2,5};\overline{3};\overline{1})$ and $m_{ss}{}^{4}(\tau_{12}) = (\overline{2,4};\overline{2,5};\overline{1,2};\overline{4,5})$. Since $p = \binom{5}{2} = 10$ and $[1,2] \in m_{ss}{}^{3}(\tau_{12})$ this implies $\tau \not\in F$. Thus $F = \{0\}$ and $\Pi_E = 0$.

Consider machine B in Figure 2. All the S.P. partitions are given in the Figure. We again want to compute F. Since $[4,5] \in E$ and

	Inputs		Output
	0	1	
1	1	2	0
2	3	2	1
3	3	5	0
4	5	4	1
5	1	5	0

I

$\tau_2 = (\overline{1,2,3,5};\overline{4})$

$\tau_1 = (\overline{1,3};\overline{2,5};\overline{4})$

0

Machine B

Figure 2.

$m_{ss}^{1}(\tau_{45})[4] = m_{ss}^{1}(\tau_{45})[5]$ clearly $I \not\in F$. Consider τ_2. Note that the only a, b such that $[a,b] \in E$ and $\tau[a] = \tau[b]$ are $\{[1,2],[2,3],[2,5]\}$. Consider these pairs. $m_{ss}^{1}(\tau_{12}) = (\overline{1,3};\overline{2};\overline{4};\overline{5})$, $m_{ss}^{2}(\tau_{12}) = (\overline{1,3};\overline{2,5};\overline{4})$, $m_{ss}^{3}(\tau_{12}) = (\overline{1,3};\overline{2,5};\overline{4})$. Since $m_{ss}^{2}(\tau_{12}) = m_{ss}^{3}(\tau_{12})$ we can stop. $m_{ss}^{1}(\tau_{23}) = (\overline{1};\overline{2,5};\overline{3};\overline{4})$, $m_{ss}^{2}(\tau_{23}) = (\overline{1,3};\overline{2,5};\overline{4})$, $m_{ss}^{3}(\tau_{23}) = (\overline{1,3};\overline{2,5},\overline{4})$ and again we can stop. Also $m_{ss}^{1}(\tau_{25}) = (\overline{1,3};\overline{2,5},\overline{4})$ and $m_{ss}^{2}(\tau_{25}) = (\overline{1,3};\overline{2,5};\overline{4})$. By noting for example that $[1,2] \not\in m_{ss}^{i}(\tau_{12})$ for any i we know that $\tau_2 \in F$. Since there can be no other partition $\tau \geq \tau_2$ in F this implies $\tau_2 = \Pi_E$.

References

[1] J. Hartmanis and R. E. Stearns (1966), Algebraic Structure Theory of Sequential Machines, Prentice-Hall.

[2] J. Hartmanis and R. E. Stearns (1963), "A Study of Feedback and Errors in Sequential Machines, IRE Trans. on Electronic Computers, EC-12.

[3] J. Hartmanis (1961), "On the State Assignment Problem for Sequential Machines, I", IRE Trans. on Electronic Computers, EC-10.

III. LOGIC HAZARDS IN THRESHOLD NETWORKS

Threshold Definitions and Theorems

The notation of Reference [1] will be used and will be briefly reviewed here.

A threshold gate has binary inputs $x_1, \ldots, x_n$ and a binary output y. The threshold gate has an internal threshold T, and each binary input has an internal weight a_i. Let $\{0,1\}^n$ denote the collection of 2^n n-tuples of $x_1, \ldots, x_n$. Associated with each gate is a function f which is defined on $\{0,1\}^n$ as follows:

$f(p) = \sum_{i=1}^{n} a_i x_i(p)$; where $p \epsilon \{0,1\}^n$, $x_i(p)$ is the value of x_i at p, and where normal arithmetic operations are used.

The function f is called the separating function.

If F is a Boolean function defined on $\{0,1\}^n$, then F is linearly separable if, and only if, there exists numbers $a_1, \ldots, a_n$ and T such that $f(p) \geq T \Leftrightarrow F(p) = 1$ and $f(p) < T \Leftrightarrow F(p) = 0$.

The Boolean function F, which is realized by a threshold gate with threshold T and separating function f, can be represented as

$$F(x_1, \ldots, x_n) = \langle f(x_1, \ldots, x_n) \rangle_T = \langle a_1 x_1 + \cdots + a_n x_n \rangle_T.$$

The collection $\{F(p), f(p)\}$ is called the map of F. Let u denote the smallest f(p) such that F(p) = 1 and ℓ denote the largest f(p) such that F(p) = 0. A map of F is separated if $\ell < u$. The gap for a separated map is the set of real numbers z such that $\ell < z \leq u$ and is denoted by $u{:}\ell$. If F is linearly separable then for some F it follows that $\ell < T \leq u$. In terms of this f, the previous expression can be written as:

$$F(x_1,\ldots,x_n) = \langle a_1x_1+\cdots+a_nx_n\rangle_{u:\ell} = \langle f\rangle_{u:\ell} .$$

Obviously, all Boolean functions are not linearly separable; hence, they cannot be realized with one threshold gate. When such a case occurs the multigate realization can be represented as

$$\langle f(p)\rangle_{u:\ell} = \langle \sum_{i=1}^{m} \beta_i y_i(p)\rangle_{u:\ell}$$

where $u:\ell$ is the gap of the output gate, y_i is the Boolean function realized by the i^{th} input, and β_i is the associated weight.

The following threshold theorems, which are proved in Reference [1], will be needed in later sections.

Theorem A. If $F(x_1,\ldots,x_n)$ is realized by $\langle \beta_1y_1+\cdots+\beta_my_m\rangle_{u:\ell}$ then any y_i, $i = 1,2,\ldots,m$ can be replaced by $(1-\bar{y}_i)$, and conversely, and the result is an equivalent realization.

Theorem B. If $F(x_1,\ldots,x_n)$ is realized by $\langle \beta_1y_1+\cdots+\beta_my_m\rangle_{u:\ell}$ then an equivalent realization is

(i) $\langle a\beta_1y_1+\cdots+a\beta_my_m\rangle_{au:a\ell}$ for any real $a > 0$ and

(ii) $\langle a+\beta_1y_1+\cdots+\beta_my_m\rangle_{u+a:\ell+a}$ for any real a.

Theorem C. If $F(x_1,\ldots,x_n) = \langle f(y_1,y_2,\ldots,y_m)\rangle_{u:\ell} = \langle \sum_{i=1}^{m} \beta_iy_i\rangle_{u:\ell}$ then the complement Boolean function $\bar{F}$ can be realized by

$$\bar{F}(x_1,\ldots,x_n) = \langle \bar{f}\rangle_{\bar{u}:\bar{\ell}} = \langle f(\bar{y}_1,\bar{y}_2,\ldots,\bar{y}_m)\rangle_{\sigma-\ell:\sigma-u}$$

$$= \langle \sum_{i=1}^{m} \beta_i\bar{y}_i\rangle_{\sigma-\ell:\sigma-u} \text{ where } \sigma = \sum_{i=1}^{m} \beta_i .$$

Corollary C. If F is realized by a threshold gate network, then a realization of $\bar{F}$ is obtained by complementing each of the input variables to the network and replacing each gap $u_j:\ell_j$ by $(\sigma_j-\ell_j):(\sigma_j-u_j)$, where σ_j is the sum of the coefficients of the $j^{\underline{th}}$ threshold gate.

When using the reconstruction technique of Lewis and Coates [1], the addition of a gate is accomplished by using the following theorem.

Theorem D.

$$[\langle f\rangle_{u:\ell}] = [\langle f+a\cdot 0\rangle_{u:\ell}] = [\langle f+a\cdot 1\rangle_{u+a:\ell+a}]$$

where 0 and 1 represent constant Boolean functions.

The constant functions, in Theorem D, represent gates which have been added. It is shown in Reference [1] that the separating functions for these can be $0 = \langle 0\rangle_{o:-\infty}$ and $1 = \langle 0\rangle_{o:-\infty}$.

Reference [1] gives a step by step procedure for the tree realization technique and gives numerous examples. Also, Examples 3 and 4 will illustrate this technique.

Definitions Concerning the Boolean Function F and Its Realization

Let p represent a variable on $\{0,1\}^n$ where the i^{th} component of p is the variable x_i^*. In this space x_i^* can be represented by either the literal x_i or $\bar{x}_i$ since $x_i = 1$ if, and only if, $\bar{x}_i = 0$.

Definition 1. A subset K of $\{0,1\}^n$ such that $x_i^* = b_i$ for $i = 1, 2, \ldots, m \leq n$ is a subcube of $\{0,1\}^n$ and will be denoted as $\{p | x_1^* = b_1, x_2^* = b_2, \ldots, x_m^* = b_m\}$, where b_i represents a specific value of 1 or 0.

Obviously there exists 2^m different ways a specific subcube can be represented. For instance, the subcube $K = \{p|x_1=1, x_2=0\}$ can also be represented as $\{p|\bar{x}_1=0, \bar{x}_2=1\}$, $\{p|x_1=1, \bar{x}_2=1\}$, or $\{p|\bar{x}_1=0, x_2=0\}$.

Definition 2. Let F be a Boolean function defined on the space $\{0,1\}^n$. A 1(0) subcube $K_1(K_0)$ of F is a subcube of $\{0,1\}^n$ such that $F(p) = 1(0)$ for all $p \epsilon K_1(K_0)$.

Definition 3. A 1(0) prime implicant $P_1(P_0)$ of F is a 1(0) subcube of F such that for all subcubes $K > P_1(P_0)$ there exists a $p \epsilon K$ such that $F(p) = 0(1)$.

Example 1. For instance, consider the Boolean function

$$F(x_1, x_2, x_3) = x_1 x_2 \bar{x}_3 + x_1 x_3 x_4 + \bar{x}_1 \bar{x}_2 x_3 + \bar{x}_1 \bar{x}_3 x_4 \, . \qquad (1)$$

Referring to Figure 1, the subcube $\{p | x_1=1, x_2=1, x_3=0\}$ is a 1 prime implicant of F; whereas, the subcube $\{p | x_1=0, x_2=1, x_4=0\}$ is a 0 prime implicant of F. Table 1 lists all the 1 and 0 prime implicants of F.

Whereas the previous examples and definitions were concerned with properties of Boolean functions, the following examples and definitions will be concerned with properties of realizations.

McCluskey [5]* has shown that for detecting logic hazards it is necessary to draw a distinction between the literals of the Boolean function and the literals of the realization. The distinction being, that in the realization, the literal x_i and its complement must be treated as independent of each other; whereas, in the Boolean function F this is not true. The necessity of this distinction is based on the fact that during an input state change it is possible for the input lines x_i and $\bar{x}_i$ to be temporarily the same and, as shown in Reference [4] and [5], it is

* Most of the material of Reference [5] has been recently published in the following book. Introduction to the Theory of Switching Circuits, McGraw-Hill Book Co., 284-306, 1965.

exactly this property that causes a hazard. Henceforth, let F^t denote the transient or output function that is realized by a given realization when x_i and its complement are treated as independent variables. The following paragraphs, which are based primarily upon McCluskey work [5], will be concerned with obtaining F^t for a given realization.

When considering the Boolean function F, the "barred" literal $\bar{x}_i$ will be used to denote the complement of x_i. The literals x_i and $\bar{x}_i$ are not independent of each other. Whereas when considering the realization of F, the "primed" literal x_i' will be used to denote the input literal which is independent of x_i but which would be the complement of x_i if an input state change is not occuring (i.e. the input literals corresponding to $\bar{x}_i$ are represented as x_i').

The method for determining F^t will depend upon the following two sets of rules. The first set of rules, Property 1, contains Boolean relations which are allowable. Similar Boolean relations which do not involve cancellation of literals would be included in this set. Property 2 contains Boolean relations which are not allowable. Similar Boolean operations which involve the cancellation of a literal would be included in this set.

Property 1

$X \cdot X = X;\ (X')' = X$

$X + X = X;\ (X + Y + \cdots + Z)' = X'Y' \ldots Z'$

$X + XY = X;\ (XY \cdots Z)' = X' + Y' + \cdots + Z'$

$X(X + Y) = X$

Property 2

$X + X' \neq 1 \quad X \cdot X' \neq 0$

$XZ + YZ + X'Y \neq XZ + X'Y$

$(X + Z)(Y + Z)(X' + Y) \neq (X + Z)(X' + Y)$

The following procedure can now be given for determining the output function F^t of a given realization.

Procedure 1

(1) Change all independent input literals $\bar{x}_i$ to x_i' .

(2) Starting with the input gates, determine the Boolean function realized by these gates in the usual manner, except $\bar{x}_i$ is replaced by x_i'. This will give the transient function for the input gates.

(3) Next consider the set of gates such that their inputs are either independent inputs or outputs of input gates. Using the transient functions obtained in Step (2), determine the function realized by each of these gates in the usual manner, except that the restrictions of Property 2 must be observed. This will give the transient functions for the gates of this set.

(4) Continue this process for all gates of the realization. The function realized by the output gate is F^t.

For example, referring to Figure 1, the transient functions realized by G_1 and G_2 are, respectively

$$F_1^t = x_1x_2x_3' \text{ and } F_2^t = (x_1+x_2x_3)' = x_1'x_2' + x_1'x_3'.$$

Hence, the transient function realized by G_0 is

$$F_0^t = x_1x_2x_3' + (x_1'x_2' + x_1'x_3')(x_4+x_3) + x_1x_3x_4$$

$$= x_1x_2x_3' + x_1'x_2'x_4 + x_1'x_2'x_3 + x_1'x_3'x_4 + x_1'x_3'x_3 + x_1x_3x_4 . \qquad (2)$$

Since F^t may contain both x_i and x_i' the domain of the transient function is $\{0,1\}^{2n}$. Hence, when studying hazards the problem becomes that of distinguishing between the properties of F and the properties of F^t. Before continuing, notice the following fact. Since $x_i' = \bar{x}_i$ for the steady state condition, it follows that if x_i' is replaced by $\bar{x}_i$ in the function F^t and if the usual Boolean operations are used, then F can be obtained from F^t.

Consider now the relations between F^t and F. Let q represent a variable on $\{0,1\}^{2n}$ where the (2i-1)th component is x_i and the 2i-th component is x_1', and where x_i and x_i' are considered as independent of each other. Thus, q is a function of $x_1, x_1', x_2, x_2', \ldots, x_n, x_n'$.

Definition 4. For each point p of $\{0,1\}^n$ such that $p = (x_1^* = b_1, \ldots, x_n^* = b_n)$, there is a point q_p of $\{0,1\}^{2n}$, called the image point of p, such that $q_p = (x_1^* = b_1, x_1^{*\prime} = \bar{b}_1, \ldots, x_n^* = b_n, x_n^{*\prime} = \bar{b}_n)$.

For example, consider the point $p = (x_1 = 1, x_2 = 0, x_3 = 1)$ of the space $\{0,1\}^3$. The image point is $q_p = (x_1 = 1, x_1' = 0, x_2 = 0, x_2' = 1, x_3 = 1, x_3' = 0)$.

Definition 5. For each subcube K of $\{0,1\}^n$ such that $K = \{p \mid x_1^* = b_1, \ldots, x_m^* = b_m\}$ there is an image subcube S of $\{0,1\}^{2n}$ such that $S = \{q \mid x_1^* = b_1, x_1^{*\prime} = \bar{b}_1, \ldots, x_m^* = b_m, x_m^{*\prime} = \bar{b}_m\}$.

For example, consider the subcube $K = \{p \mid x_1 = 1, x_3 = 0\}$ of $\{0,1\}^3$. The corresponding image subcube is $S = \{q \mid x_1 = 1, x_1' = 0, x_3 = 0, x_3' = 1\}$.

Obviously, if a subcube K contains 2^{n-m} points, the image subcube S will contain $2^{2(n-m)}$ points and 2^{n-m} of these points will be the image points of the points of K.

Notice that since the inputs x_i and x_i' in a realization may not change simultaneously, it is possible for the two to be temporarily the same. Hence, the realization $\langle f \rangle_{u:\ell}$ is actually defined on the space $\{0,1\}^{2n}$. In fact, now that an image point and output function have been defined, a realization can be redefined as follows.

Definition 6. Let F be an arbitrary Boolean function and let F^t be the output function of an arbitrary realization $\langle f \rangle_{u:\ell}$ of F, where F and F^t are defined on $\{0,1\}^n$ and $\{0,1\}^{2n}$, respectively. Let p be an arbitrary point of $\{0,1\}^n$ and q_p the corresponding image point. Then $\langle f \rangle_{u:\ell}$ is said to realize F if, and only if, for every $p \epsilon \{0,1\}^n$

$$F(p) = 1 \Leftrightarrow f(q_p) \geq T \text{ (i.e. } F^t(q_p) = 1)$$

$$F(p) = 0 \Leftrightarrow f(q_p) < T \text{ (i.e. } F^t(q_p) = 0).$$

Notice that Definition 6 does not place any requirement upon the non-image points of $\{0,1\}^{2n}$. Hence, $F^t(q)$, for the non-image points can be either 1 or 0. These points, however, do determine the hazard conditions of the realization.

Definition 7. Let F^t be the transient function of an arbitrary threshold realization $\langle f \rangle_{u:\ell}$. A 1(0) subcube of F^t is a subcube S of $\{0,1\}^{2n}$ such that $F^t(q) = 1(0)$ for all $q \epsilon S$.

For example, the subcube $\{q \mid x_1=1, x_2=1, x_3=0\}$ is a 1 subcube of Equation 2.

Now that the properties relating the Boolean function F and the transient function F^t have been established, the next task is to define a logic hazard in terms of our newly established definitions.

A logic hazard, first defined by Eichelberger [2], can be defined in the following equivalent manner.

Definition 8. A realization $\langle f \rangle_{u:\ell}$ of F contains a logic 1(0) hazard within the 1(0) subcube $K_1(K_0)$ of F if, and only if, the corresponding image subcube $S_1(S_0)$ of $\{0,1\}^{2n}$ is not a 1(0) subcube of F^t, where F^t is the transient function of $\langle f \rangle_{u:\ell}$.

Consider the function $F^t(x_1, x_1', \ldots, x_n, x_n')$ and the subcube $S = \{q | x_1 = b_1, x_1' = \bar{b}_1, \ldots, x_m = b_m, x_m' = \bar{b}_m\}$. The function which results when x_i and x_i' are set equal to b_i and $\bar{b}_i$, respectively, in the function F^t, is referred to as a reduced function of F^t and is denoted by $F^t(S)$. $F^t(S) = 1(0)$ implies that the reduced function $F^t(S)$ is the constant function 1(0). By using the idea of reduced functions, Definition 8 can also be expressed in the following equivalent manner.

Definition 9. A realization $\langle f \rangle_{u:\ell}$ of F contains a logic 1(0) hazard within the 1(0) subcube $K_1(K_0)$ of F if, and only if, $F^t(S_1) \neq 1$ $(F^t(S_0) \neq 0)$, where $S_1(S_0)$ is the image subcube of $K_1(K_0)$ and F^t is the transient function of $\langle f \rangle_{u:\ell}$.

Detection of Logic Hazards - Method 1

The following theorem will now give a method for determining if a realization $\langle f \rangle_{u:\ell}$ of F contains any logic hazards. The proof follows directly from Definitions 3, 5, and 9.

Theorem 1. Let $\{P_1^i\}(\{P_0^i\})$ denote the set of 1(0) prime implicants of F and let $\{S_1^i\}(\{S_0^i\})$ denote the corresponding set of image subcubes in the space $\{0,1\}^{2n}$. A realization $\langle f\rangle_{u:\ell}$ of F will not contain any logic 1(0) hazards if, and only if, $F^t(S_1^i) = 1, (F^t(S_0^i) = 0)$ for all $S_1^i \epsilon \{S_1^i\}, (S_0^i \epsilon \{S_0^i\})$, where F^t is the transient function of $\langle f\rangle_{u:\ell}$.

Summarizing, Theorem 1 can be used to determine if a given realization contains a logic hazard. If it does, then, Definition 8 or Definition 9 can be used to determine the subcube within which the logic hazard occured. The following procedure can be used to determine if a realization contains any logic 1(0) hazards.

Procedure 2

(1) Determine the transient function F^t of $\langle f\rangle_{u:\ell}$.

(2) Determine the set of 1(0) prime implicants $\{P_1^i\}(\{P_0^i\})$ of F.

(3) Determine $F^t(S_1^i), (F^t(S_0^i))$ for all $S_1^i(S_0^i)$, where $S_1^i(S_0^i)$ is the image subcube of $P_1^i(P_0^i)$.

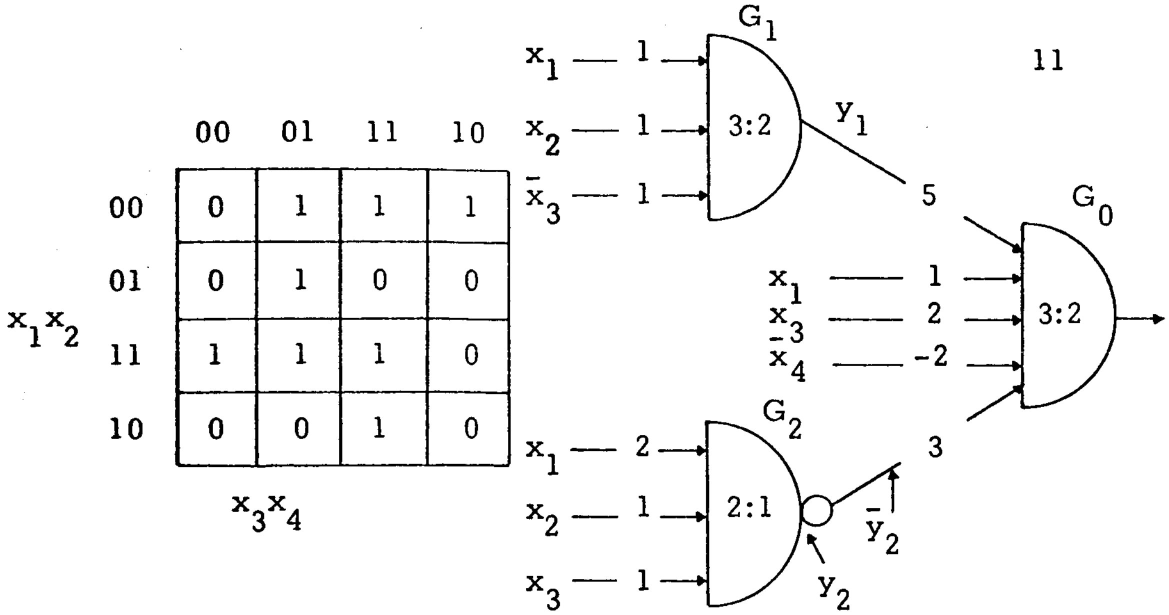
(4) The realization $\langle f\rangle_{u:\ell}$ does not contain a logic 1(0) hazard within $P_1^i(P_0^i)$ if, and only if, $F^t(S_1^i) = 1$ $(F^t(S_0^i) = 0)$.

The following example will illustrate Procedure 2.

Example 1. Consider the Boolean function of Equation 1,

$$F(x_1, x_2, x_3, x_4) = x_1x_2\bar{x}_3 + x_1x_3x_4 + \bar{x}_1\bar{x}_2x_3 + \bar{x}_1\bar{x}_3x_4 .$$

The Karnaugh map and a realization for Equation 1 are given in Figure 1.



x_1x_2 \ x_3x_4	00	01	11	10
00	0	1	1	1
01	0	1	0	0
11	1	1	1	0
10	0	0	1	0

(a) (b)

Figure 1: Karnaugh Map and Realization for Example 1

The problem is to determine if the given realization contains any logic hazards. From Procedure 2, the first step is to obtain F^t. In this case, F^t is given by Equation (2). The following table contains the set of 1(0) prime implicants $\{P_1^i\}(\{P_0^i\})$ of F and the corresponding set of reduced functions $\{F^t(S_1^i)\},(\{F^t(S_0^i)\})$.

For example, from Table 1, $P_1^4 = \{p|x_2=0,x_3=1,x_4=1\}$; then, $S_1^4 = \{q|x_2=0,x_2'=1,x_3=1,x_3'=0,x_4=1,x_4'=0\}$. Hence, the reduced function $F^t(S_1^4)$ is $F^t(x_2=0,x_2'=1,x_3=1,x_3'=0,x_4=1,x_4'=0) = x_1' + x_1$.

Referring to Table 1, $F^t(S_1^4) \neq 1$, $F^t(S_1^7) \neq 1$, $F^t(S_1^8) \neq 1$, and $F^t(S_0^2) \neq 0$. [For the present, disregard the columns labeled $\tilde{f}(q_0^i)$ and $\tilde{f}(q_1^i)$]. From Definition 9, the realization will contain a logic 1 hazard in P_1^4, P_1^7, and P_1^8 and a logic 0 hazard in P_0^2.

	P_1^i			P_0^i		
i	$x_1x_2x_3x_4$	$F^t(S_1^i)$	$\gamma(q_0^i)$	$x_1x_2x_3x_4$	$F^t(S_0^i)$	$\gamma(q_1^i)$
1	1 1 0 -	1	6	- 1 1 0	0	3
2	0 0 1 -	1	5	0 1 - 0	$x_1'x_3'x_3$	5
3	0 - 0 1	1	5	1 0 0 -	0	3
4	- 0 1 1	x_1+x_1'	4	- 0 0 0	0	4
5	1 - 1 1	1	5	0 1 1 -	0	4
6	0 0 - 1	1	5	1 - 1 0	0	3
7	1 1 - 1	x_3+x_3'	3	1 0 - 0	0	3
8	- 1 0 1	x_1+x_1'	2	0 - 0 0	0	3

Table 1: Table for Example 1

Procedure 2 requires the calculation of the transient function F^t and the calculation of all of the prime implicants of F. McCluskey [5] has presented several alternate methods for determining if a given realization contains any static hazards, all of which require the calculation of F^t. These methods can be readily extended to include logic hazards and for further detail the reader is referred to Reference [5].

Before continuing, notice the following fact. If the realization contains negative weights and/or inverting gates, one cannot determine F^t by successively applying the 2^{2n} possible combinations of $\{0,1\}^{2n}$ as inputs to the realization and determining the value of the output for each. In terms of the separating function this gives the surprising result that

$$F^t(q) = 1(0) \not\Rightarrow f(q) \geq u \; (f(q) \leq \ell) \quad \underline{\text{OR}}$$

$$f(q) \geq u \; (f(q) \leq \ell) \not\Rightarrow F^t(q) = 1(0), \text{ where } q \in \{0,1\}^{2n}.$$

For example, consider the point $q = (x_1=0, x_1'=0, x_2=0, x_2'=1, x_3=1, x_3'=0, x_4=1, x_4'=0)$. Referring to Figure 1(b), $f_1(q)=0$; hence, the output of G_1 is 0. Also, $f_2(q)=1$; herce, the output of G_2 is 1. Likewise, $f_0(q)=5$; thus, the output of the realization is 1. Now referring to Equation (2), $F^t(q)=0$. Hence, $F^t(q)=0$; whereas, $f(q) > u$.

The next section will be concerned with modifying the given realization $\langle f\rangle_{u:\ell}$ in such a manner that F^t can always be obtained by considering only the 2^{2n} possible input states. This modification will, in many cases, give an easier method for determining F^t. Also, it will yield a method for determining if a realization contains any logic hazards which does not require the calculation of F^t. But even more important, it will develop the fundamentals which will be needed to synthesize hazard free threshold gate networks.

Detection of Logic Hazards - Method 2

First we will prove a lemma which concerns modifying the given realization $\langle f\rangle_{u:\ell}$ and then we will prove the theorem which will give the desired results. However, the following term must be defined first.

Definition 10. A threshold realization which does not contain any negative weights or inverting gates will be called a positive threshold realization and will be denoted as $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$.

Lemma 1 For each realization $\langle f\rangle_{u:\ell}$ of F there exists a unique corresponding positive realization $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ of F; moreover, the transient functions of the two are the same.

Proof. First we will prove the following two assertions.

(1) If a realization $\langle \sum_{i=1}^{m} \beta_i y_i^* \rangle_{u:\ell}$ is followed by an inverter and the combined realization yields the transient function F^t, then F^t is also realized by $\langle \sum_{i=1}^{m} \beta_i \bar{y}_i^* \rangle_{\sigma-\ell:\sigma-u}$, where $\sigma = \sum_{i=1}^{m} \beta_i$.

(2) If a realization $\langle -\sum_{i=1}^{m'} |\beta_i| y_i^* + \sum_{i=m'+1}^{m} \beta_i y_i^* \rangle_{u:\ell}$ yields the transient function F^t, then F^t is also realized by

$$\langle \sum_{i=1}^{m'} |\beta_i| \bar{y}_i^* + \sum_{i=m'+1}^{m} \beta_i y_i^* \rangle_{u':\ell'}, \text{ where } u':\ell' = u + \sum_{i=1}^{m'} |\beta_i| : \ell + \sum_{i=1}^{m'} |\beta_i|.$$

Proof of Assertion 1: Referring to Figure 2(a), let F_i^t be the input transient function of the gate $G_0 = \langle \sum_{i=1}^{m} \beta_i y_i^* \rangle_{u:\ell}$ which corresponds to y_i^* and let $F^{t'}$ be the corresponding output function of the gate G_0.

Referring to Figure 2(a), assume that $F^{t'}$ is expressed in the following minimum sum-of-product form, $F^{t'} = z_1 + z_2 + \cdots + z_n$. From Property 1, the complement is $F^t = z',z'_2, \ldots, z'_n$. Hence the output function of the combined realization is $F^t = z',z'_2, \ldots, z'_n$. Now consider the gate G_0 of Figure 2(b), which has a transient output of F^* and inputs $F_1^{t'}, \ldots, F_m^{t'}$. From Theorem C, it is known that $F^* = z', z'_2, \ldots, z'_n$, hence, $F^t = F^*$.

Since $F^t = F^*$, it can be concluded that the hazard condition will not change if the inverter is moved from the output of the gate G_0 to the inputs of G_0 and $u:\ell$ is changed to $\sigma-\ell:\sigma-u$ (see Figure 2). This in effect says that the hazard condition is not changed if the inverter is removed from the realization and instead $F^{t'}_i$ is realized on all of the inputs of G_0 and $u{:}\ell$ is changed to $\sigma-\ell{:}\sigma-u$. If F^{t}_i is an independent input literal x^*_k, this involves realizing $x^{*'}_k$ instead of x^*_k. Whereas, if F^{t}_i is a dependent function it follows, from the above proof, that this involves realizing the complements of the transient functions which realize F^{t}_i. By continuing this process from G_0 to the input gates, the final results will be (1) all independent inputs are changed from x^*_i to $x^{*'}_i$ and (2) all gaps $u_j{:}\ell_j$ will be changed to $\sigma_j-\ell_j{:}\sigma_j-u_j$, where G_j is an arbitrary gate of the realization. However, in the steady state condition $x^{*'}_i = \bar{x}^*_i$ and from theorem C, $\bar{u}_j{:}\bar{\ell}_j = \sigma_j-\ell_j{:}\sigma_j-u_j$. Referring to Corollary C, steps (1) and (2) are the same conditions needed to obtain the realization $\langle \sum_{i=1}^{m} \beta_i \bar{y}^*_i \rangle_{\sigma-\ell:\sigma-u}$. Thus, part (1) of Lemma 1 is proved.

The proof of Assertion 2 is similar to that of Assertion 1 and will be omitted. Δ

It now follows from Assertion 1 that all inverting gates in the realization $\langle f \rangle_{u:\ell}$ of F can be replaced by non-inverting gates and the hazard condition will not change.

Also, it follows from Assertion 2 that all negative weights in the realization $\langle f \rangle_{u:\ell}$ can be replaced by positive weights and the hazard condition will not change.

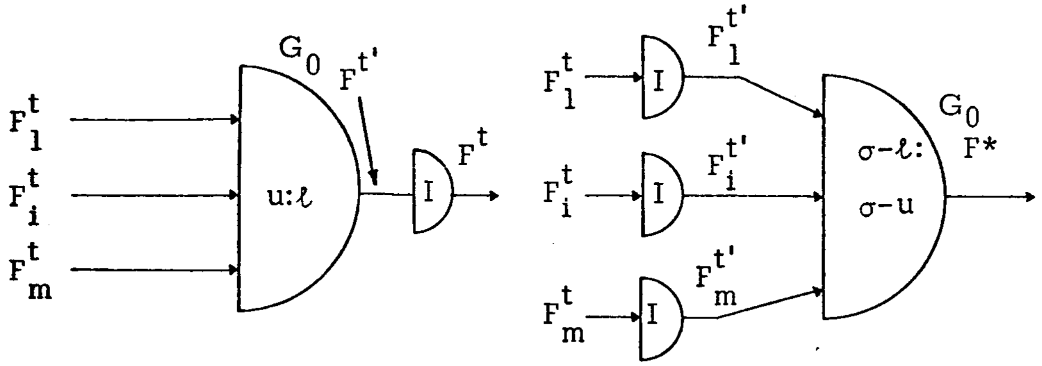


Figure 2. Removal of Invertes From A Given Realization

Before continuing, notice that the inverse of the statements used to prove Lemma 1 are also true. Hence, given an arbitrary realization, if Theorem C is used to change noninverting gates to inverting gates and if Theorems A and B are used to change positive weights to negative weights the realization obtained will have the same hazard condition. Hence, there is no loss of generality by considering only realizations which do not have inverting gates and/or negative weights.

By using Lemma 1, we can now state the following procedure for obtaining the corresponding positive realization $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ of a given realization $\langle f\rangle_{u:\ell}$. The following procedure will not differentiate between an inverting gate and a gate followed by an inverter.

Procedure 3

(1) Starting on the 0 logic of $\langle f\rangle_{u:\ell}$ (i.e., with the output gate), if a gate $\langle \sum_{i=1}^{m} \beta_i y_i^* \rangle_{u:\ell}$ is followed by an inverter, then replace it and the corresponding inverter with the gate $\langle \sum_{i=1}^{m} \beta_i \bar{y}_i^* \rangle_{\sigma-\ell:\sigma-u}$, where $\sigma = \sum_{i=1}^{m} \beta_i$.

(2) If the preceding gate that corresponds to y_i^* is followed by an inverter, then realize $\bar{y}_i^*$ by removing the inverter.

(3) If the preceding gate that corresponds to y_i^* is not followed by an inverter, then realize $\bar{y}_i^*$ by using Theorem C.

(4) Continue this process until all inverters have been removed from $\langle f \rangle_{u:\ell}$.

(5) Starting on the 0 logic level of $\langle f \rangle_{u:\ell}$, if a gate $\langle -\sum_{i=1}^{m'} |\beta_i| y_i^* + \sum_{i=m'+1}^{m} \beta_i y_i^* \rangle_{u:\ell}$ has negative weights for $1 \leq i \leq m'$ and positive weights for $m'+1 \leq i \leq m$ then, using Theorem C, replace it with the gate $\langle \sum_{i=1}^{m'} |\beta_i| \bar{y}_i^* + \sum_{i=m'+1}^{m} \beta_i y_i^* \rangle_{u':\ell'}$, where $u':\ell' = u + \sum_{i=1}^{m'} |\beta_i| : \ell + \sum_{i=1}^{m'} |\beta_i|$.

(6) Repeat step (5) until all negative weights have been removed from $\langle f \rangle_{u:\ell}$. The final realization will be the corresponding positive realization of $\langle f \rangle_{u:\ell}$.

For example, consider the realization of Figure 1(b). The corresponding positive realization is

$$\langle x_1+2x_3+2x_4+5 \ \langle x_1+x_2+\bar{x}_3 \rangle_{3:2} + 3\langle 2\bar{x}_1+\bar{x}_2+\bar{x}_3 \rangle_{3:2} \rangle_{5:4} . \qquad (3)$$

<u>Definition 11.</u> Consider the subcube $S = \{q \mid x_1=b_1, x_1'=\bar{b}_1, \ldots, x_m=b_m, x_m'=\bar{b}_m\}$ of the space $\{0,1\}^{2n}$. Define q_1 as the element $(x_1=b_1, x_1'=\bar{b}_1, \ldots, x_m=b_m, x_m'=\bar{b}_m, x_{m+1}=1, \ldots, x_n=1, x_n'=1)$ and q_0 as the element $(x_1=b_1, x_1'=\bar{b}_1, \ldots, x_m=b_m, x_m'=\bar{b}_m, x_{m+1}=0, \ldots, x_n=0, x_n'=0)$. The elements q_0 and q_1 will be called the <u>minimum</u> and <u>maximum elements of S</u>, respectively.

A theorem can now be given for determining if a given realization contains a logic hazard within a specific subcube. The theorem will not require the calculation of F^t.

Theorem 2. Let $\langle f\rangle_{u:\ell}$ be an arbitrary realization of F and let $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ be the corresponding positive realization. Let $\{P_1^i\}(\{P_0^i\})$ be the set of 1(0) prime implicants of F, let $\{S_1^i\}(\{S_0^i\})$ be the corresponding set of image subcubes, and let $\{q_0^i\}(\{q_1^i\})$ be the corresponding set of minimum (maximum) elements. The realization $\langle f\rangle_{u:\ell}$ will not contain any logic 1(0) hazards if, and only if, $\tilde{f}(q_0^i)\geq\tilde{u}(\tilde{f}(q_1^i)\leq\tilde{\ell})$ for all $q_0^i\epsilon$ $\{q_0^i\}(q_1^i\ \epsilon\{q_1^i\})$.

Proof. Let P_1 be an arbitrary 1 prime implicant of F. From Definition 9, the realization $\langle f\rangle_{u:\ell}$ will not contain a logic 1 hazard within P_1 if, and only if, $F^t(S_1) = 1$, where, by Lemma 1, F^t is the output function of both $\langle f\rangle_{u:\ell}$ and $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$. Thus, we will prove that $F^t(S_1)=1$ if, and only if, $\tilde{f}(q_0^1)\geq\tilde{u}$.

First we will prove that $F^t(S_1) = 1$ implies that $\tilde{f}(q_0)\geq\tilde{u}$. By Definition 7, $F^t(S_1) = 1$ implies that $F^t(q) = 1\ \forall q\epsilon S_1$. Let q be an arbitrary element of S_1. Since there are no negative weights or inverting gates in $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$, it follows that $F^t(q) = 1$ implies $\tilde{f}(q)\geq\tilde{u}$. Also, since q is an arbitrary element, it follows that $F^t(q_0) = 1$ and, hence $\tilde{f}(q_0)\geq\tilde{u}$.

Next assume that $\tilde{f}(q_0)\geq\tilde{u}$. Let q be an arbitrary element of S_1. Since the realization $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ does not include any negative weights or inverting gates, the value of the separating function $\tilde{f}$ can only be increased by inputs which are a 1; hence, $\tilde{f}(q) \geq f(q_0) \geq \tilde{u}$. Also, since $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\ell}}$ does not contain any negative weights or inverting gates, it follows that $\tilde{f}(q) \geq u$ implies $F^t(q) = 1$. Finally, since q is an arbitrary element of S_1, it can be concluded that $F^t(q) = 1 \; \forall q \epsilon S_1$; hence, $F^t(S_1) = 1$.

The proof concerning logic 0 hazards is similar and will be omitted. Δ

The following procedure will outline a method for determining if a given realization contains any logic 1(0) hazards.

Procedure 4.

(1) Determine the set of 1(0) prime implicants $\{P_1^i\}(\{P_0^i\})$ of F.

(2) Obtain the corresponding positive realization $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\ell}}$ of $\langle f \rangle_{u:\ell}$.

(3) Determine $\tilde{f}(q_0^i)$, $(\tilde{f}(q_1^i))$ for all $q_0^i(q_1^i)$, where $q_0^i(q_1^i)$ is the corresponding minimum (maximum) image point of $P_1^i(P_0^i)$.

(4) The realization $\langle f \rangle_{u:\ell}$ will not contain a logic 1(0) hazard within $P_1^i(P_0^i)$ if, and only if, $\tilde{f}(q_0^i) \geq \tilde{u}, (\tilde{f}(q_1^i) \leq \tilde{\ell})$.

Suppose that we apply Procedure 4 to Example 1. The set of prime implicants are given in Table 1 and the corresponding positive realization is given by Equation (3). Now consider step (3). For example, consider the 1 prime implicant P_1^4. The corresponding image subcube is $S_1^4 = \{q \mid x_2{=}0, x_2'{=}1, x_3{=}1, x_3'{=}0, x_4{=}1, x_4'{=}0\}$. Hence, the corresponding minimum image point is $q_0^4 = (x_1{=}0, x_1'{=}0, x_2{=}0, x_2'{=}1, x_3{=}1, x_3'{=}0, x_4{=}1, x_4'{=}0)$. From Equation (3), $\tilde{f}(q_0^4) = 4$. Since $\tilde{u} = 5$, it follows from Theorem 2 that the realization will contain a logic 1 hazard in P_1^4.

Table 1 contains $\tilde{f}(q_0^i)$ for each $P_1^i \epsilon F$ and $\tilde{f}(q_1^i)$ for each $P_0^i \epsilon F$. As can be seen, the results of Procedure 4 agree with those of Procedure 2.

Detection of Logic Hazards - Method 3

Methods 1 and 2 for detecting logic hazards were based on either F^t or the input-output relations of the realization. A method will now be given which will be based on the structure of the realization. As will be seen, this later method is inferior to the previous two for detecting logic hazards; however, the results obtained from this method are needed to obtain a theorem for synthesizing a hazard free threshold network.

Definition 12. Let $\langle f\rangle_{u:\ell}$ denote an arbitrary realization of the Boolean function F and let B denote a set of gates contained in $\langle f\rangle_{u:\ell}$. The set of gates B will be defined as an output connected subset of gates if (1) B contains the output gate of $\langle f\rangle_{u:\ell}$ and (2) excluding the output gate of $\langle f\rangle_{u:\ell}$, the output of each gate in B is an input to another gate in B.

For example, in Figure 1(b) the sets $\{G_1, G_0\}$ and $\{G_0\}$ are output connected subsets of $\langle f\rangle_{u:\ell}$; whereas, the set $\{G_1, G_2\}$ is not.

Definition 13. Let G_i denote an arbitrary gate in the positive realization $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ and let B* denote an arbitrary "output connected subset" of $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$. Also, let S be an arbitrary subcube in the space $\{0,1\}^{2n}$ with minimum element q_0 and maximum element q_1. The set B* is defined as a *1(0) branch of $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ which realizes $\underline{S}$* if, when $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ is modified such that all gates not in B* have 0(1) output, then the value of $\tilde{f}_i(q_0)$, $(\tilde{f}_i(q_1))$ for the modified realization satisfies the condition $\tilde{f}_i(q_0) \geq \tilde{u}_i (f_i(q_1) \leq \tilde{\ell}_i)$ for all $G_i \epsilon B^*$.

For example, consider the realization of Equation 3, which is

$$\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}} = \overset{G_0}{\langle} x_1+2x_3+2x_4+5 \overset{G_1}{\langle} x_1+x_2+\bar{x}_3\rangle_{3:2}+3\overset{G_2}{\langle} 2\bar{x}_1+\bar{x}_2+\bar{x}_3\rangle_{3:2}\rangle_{5:4}.$$

Suppose one wishes to determine if the set of gates $\{G_0, G_1\}$ is a 1 branch which realizes the subcube $S = \{q | x_1=1, x_1'=0, x_2=1, x_2'=0, x_3=0, x_3'=1\}$. According to Definition 13, in order for $\{G_0, G_1\}$ to be a 1 branch which realizes S the condition $\tilde{f}_1(q_0) \geq \tilde{u}_1$ and $\tilde{f}_0(q_0) \geq \tilde{u}_0$ must exist when the output of G_2 is 0. Under the condition $G_2=0$, the modified realization becomes

$$\overset{G_0}{\langle} x_1+2x_3+2x_4+5 \overset{G_1}{\langle} x_1+x_2+\bar{x}_3\rangle_{3:2}+0\rangle_{5:4} .$$

The point q_0 is $(x_1=1, x_1'=0, x_2=1, x_2'=0, x_3=0, x_3'=1, x_4=0, x_4'=0)$. From the above equation, $\tilde{f}_1(q_0)=3$ and $\tilde{f}_0(q_0)=6$. Thus, from Definition 13, the set $\{G_0, G_1\}$ is a 1 branch which realizes S.

The following lemma will give a relationship between the term $\tilde{f}(q_0)(\tilde{f}(q_1))$ and a 1(0) branch.

Lemma 2. Let $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ be an arbitrary positive realization of F. Also, let $K_1(K_0)$ be an arbitrary 1(0) subcube of F, let the set $\{q_p\}_1(\{q_p\}_0)$ be the corresponding set of image points of $K_1(K_0)$, and let $q_0(q_1)$ be the corresponding minimum (maximum) element of $K_1(K_0)$. Then $\tilde{f}(q_0) \geq \tilde{u}, (\tilde{f}(q_1) \leq \tilde{\ell})$ if, and only if, there exists a 1(0) branch $B_1(B_0)$

of $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\mathcal{L}}}$ which realizes all $q_p \in \{q_p\}_1 (\{q_p\}_0)$.

Proof. First we will prove that all $q_p \in \{q_p\}_1$ are realized by a particular 1 branch B_1 implies that $\tilde{f}(q_0) \geq \tilde{u}$. We will prove this result by using induction.

Let G_j be an arbitrary gate of B_1 which is on the r logic level of $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\mathcal{L}}}$. Let G_i be an arbitrary gate of B_1 such that its output is an input to G_j (i.e., G_i must be on the r+1 or greater logic level). Let $K_1 = \{p | x^*_1 = b_1, \ldots, x^*_m = b_m\}$ be an arbitrary 1 subcube of F.

Assume that Lemma 2 is true for all G_i (i.e., assume $\tilde{f}_i(q_0) \geq u_i$). Thus, the output of G_i is not a function of $x_{m+1}, x'_{m+1}, \ldots, x_n, x'_n$.

Now consider the independent inputs to G_j. Let $\hat{x}_i$ denote an independent input to G_j. Assume that G_j does not have both $\hat{x}_i$ and its complement $\hat{x}'_i$ as inputs, which is true for all gates. From the hypothesis of Lemma 2, it is known that the output of G_j is a 1 for all $q_p \in \{q_p\}_1$. Hence, there exists a q_p for which $\hat{x}_{m+1} = 1, \ldots, \hat{x}_n = 1$ and some other q_p for which $\hat{x}_{m+1} = 0, \ldots, \hat{x}_n = 0$ such that the output of G_j is 1 for both. Thus, the output of G_j is not a function of $\hat{x}_{m+1}, \ldots, \hat{x}_n$. Now since the corresponding complements $\hat{x}'_{m+1}, \ldots, \hat{x}'_n$ are not inputs to G_j and since the output of G_i is not a function of $x_{m+1}, x'_{m+1}, \ldots, x_n, x'_n$, it follows that the output of G_j is not a function of $x_{m+1}, x'_{m+1}, \ldots, x_n, x'_n$. Thus $\tilde{f}_j(q_0) \geq \tilde{u}_j$. Hence, we have proved that if Lemma 2 is true for all G_i it is true for G_j.

Next consider an arbitrary input gate of the branch. Since for the modified realization (i.e., all gates not contained in B_1 have 0 outputs) the gate contains only independent inputs and since it does not contain both x_i and x_i', it follows, by the above reasoning, that Lemma 2 is true for all input gates of B_1. Thus, it follows from induction that the $\tilde{f}(q_o) \geq \tilde{u}$.

Next we will prove that $\tilde{f}(q_0) \geq u_0$ implies that all $q_p \epsilon \{q_p\}_1$ are realized by the same 1 branch.

First assume $\tilde{f}(q_0) \geq \tilde{u}$; hence, the output of $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\ell}}$ is a 1 at q_0. Let B* be the largest output connected subset of gates that have unit output at q_0. Clearly this is nonempty. Since all coefficients are positive and no inverters exist in the realization, then the gates of B* will have unit output for all $q \epsilon S_1$, where S_1 is the image subcube of K_1, independent of the outputs of the gates not in B*. Hence, B* is a 1 branch which realizes all $q \epsilon S_1$, and hence, all $q_p \epsilon \{q_p\}_1$.

The proof concerning the inequality $\tilde{f}(q_1) \leq \tilde{\ell}$ is similar and will be omitted. Δ

Theorem 3 follows directly from Lemma 2 and the proof of Theorem 2.

<u>Theorem 3</u>. Let $\langle f \rangle_{u:\ell}$ be an arbitrary realization of F and let $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\ell}}$ be the corresponding positive realization. Also, let $K_1(K_0)$ be an arbitrary 1(0) subcube of F and let the set $\{q_p\}_1(\{q_p\}_0)$ be the corresponding set of image points for all $p \epsilon K_1(K_0)$. The realization $\langle f \rangle_{u:\ell}$ will not contain a logic 1(0) hazard in $K_1(K_0)$ if, and only if, there exists a 1(0) branch $B_1(B_0)$ of $\langle \tilde{f} \rangle_{\tilde{u}:\tilde{\ell}}$ which realizes all $q_p \epsilon \{q_p\}_1(\{q_p\}_0)$.

Summarizing, we have proved the following facts. Let K^α be an arbitrary 1(0) subcube of F and let S^α and $q_0^\alpha(q_1^\alpha)$ be the corresponding image subcube and minimum (maximum) element, respectively. An arbitrary realization $\langle f\rangle_{u:\ell}$ of F will not contain a logic 1(0) hazard within $K^{\alpha} \Leftrightarrow F^{t}(S^{\alpha}) = 1(0) \Leftrightarrow \tilde{f}(q_0^{\alpha}) \geq \tilde{u}(\tilde{f}(q_1^{\alpha}) \leq \tilde{\ell}) \Leftrightarrow$ there exists a 1(0) branch $B_1(B_0)$ of $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ which realizes all of the image points of K^{α}, where $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ is the corresponding positive realization.

As in the preceding cases, a theorem can now be given for determining if a given realization contains any logic hazards. The proof follows from Theorem 2 and Theorem 3.

Theorem 4. Let $\langle f\rangle_{u:\ell}$ be an arbitrary realization of F and let $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ be the corresponding positive realization. Also, let $\{P_1^i\}(\{P_0^i\})$ be the set of all 1(0) prime implicants of F and let $\{\{q_p\}_1^i\}(\{\{q_p\}_0^i\})$ be the corresponding collection of sets of image points. The realization $\langle f\rangle_{u:\ell}$ will not contain any logic 1(0) hazards if, and only if, there exists a 1(0) branch $B_1^{\alpha}(B_0^{\alpha})$ of $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ which realizes all $q_p \epsilon \{q_p\}_1^i, (\{q_p\}_0^i)$ for all $\{q_p\}_1^i \epsilon \{\{q_p\}_1^i\}, (\{q_p\}_0^i \epsilon \{\{q_p\}_0^i\})$.

Example 2 will illustrate the application of Theorem 4:

Example 2. Again consider the realization of Figure 1(b). From Equation 3 the positive realization is

$$\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}} = \langle x_1+2x_3+2x_4+5\langle x_1+x_2+\bar{x}_3\rangle_{3:2}+3\langle 2\bar{x}_1+\bar{x}_2+\bar{x}_3\rangle_{3:2}\rangle_{5:4}. \quad (4)$$

Only the logic 1 hazards will be considered. Therefore, the following table will contain only the points p_j of $\{0,1\}^n$ such that $F(p_j) = 1$, the corresponding image points q_p^j, and the set of 1 branches which realize each q_p^j. Actually, in this example each point q_p^j is realized by only one 1 branch.

In Table 2, the points $p_j \epsilon \{0,1\}^n$ such that $F(p_j)=1$ can be obtained from Figure 1(a); whereas, the corresponding 1 branch can be obtained from Equation 4.

Consider the 1 prime implicant $\{p|x_1=1, x_2=1, x_3=0\}$. The corresponding image points are $q_p^1 = (x_1=1, x_1'=0, x_2=1, x_2'=0, x_3=0, x_3'=1, x_4=0, x_4'=1)$ and $q_p^2 = (x_1=1, x_1'=0, x_2=1, x_2'=0, x_3=0, x_3'=1, x_4=1, x_4'=0)$. Referring to Table 2, both image points are realized by the 1 branch $\{G_1, G_0\}$. Therefore, by Theorem 3, the realization will not contain a logic 1 hazard within $\{p|x_1=1, x_2=1, x_3=0\}$.

Next consider the 1 prime implicant $\{p|x_1=1, x_2=1, x_4=1\}$. The corresponding image points are q_p^2 and q_p^3. Referring to Table 2, q_p^2 and q_p^3 are realized by the 1 branches $\{G_1, G_0\}$ and $\{G_0\}$, respectively. Hence, from Theorem 3, the realization will contain a logic 1 hazard within $\{p|x_1=1, x_2=1, x_4=1\}$.

Likewise, it can be determined that the realization will contain a logic 1 hazard within the 1 prime implicants $\{p|x_2=1, x_3=0, x_4=1\}$ and $\{p|x_2=0, x_3=1, x_4=1\}$. Hence, the results agree with those of Example 1.

j	p_j	q_m^j	B_1^α
	$x_1x_2x_3x_4$	$x_1x_1'x_2x_2'x_3x_3'x_4x_4'$	
1	1 1 0 0	1 0 1 0 0 1 0 1	$\{G_1, G_0\}$
2	1 1 0 1	1 0 1 0 0 1 1 0	$\{G_1, G_0\}$
3	1 1 1 1	1 0 1 0 1 0 1 0	$\{G_0\}$
4	1 0 1 1	1 0 0 1 1 0 1 0	$\{G_0\}$
5	0 1 0 1	0 1 1 0 0 1 1 0	$\{G_0, G_2\}$
6	0 0 0 1	0 1 0 1 0 1 1 0	$\{G_0, G_2\}$
7	0 0 1 1	0 1 0 1 1 0 1 0	$\{G_0, G_2\}$
8	0 0 1 0	0 1 0 1 1 0 0 1	$\{G_0, G_2\}$

Table 2: Table for Example 1

As previously mentioned, the application of Theorem 3 or 4 to determine if a given realization contains any logic hazards is considerably more involved than Procedure 4. Hence, it is doubtful if Theorem 3 or 4 would be used to detect logic hazards. However, Theorem 4 will be used, in conjunction with a later lemma, to derive a theorem for synthesizing a hazard free threshold network directly from the Boolean function F. How this is accomplished is the subject of the next section.

Synthesis of Hazard Free Threshold Networks

This section will be concerned with synthesizing positive threshold gate realizations which are hazard free. Then, as previously shown, Theorems A and B can be used to change positive weights to negative weights and Theorem C can be used to change noninverting gates to inverting gates and the resulting realizations will also be hazard free. Hence, one can obtain any type of desired hazard free threshold realization. The following synthesis technique will be primarily based on the multigate realization technique of Reference [1]. However, at the end of this section, an alternate method is given for obtaining a two level hazard free threshold realization.

Since the material will be largely concerned with the n-level of the function tree, the following terms are needed. Let $\{q_p\}$ be the subset of $\{0,1\}^{2n}$ such that q_p is the image point of p for each $p \varepsilon \{0,1\}^n$. Consider a function tree for a Boolean function F. A given position on the n-level of the tree corresponds to a specific p of $\{0,1\}^n$, in that each position corresponds to a unique reduced function F(p), where p is the subcube of $\{0,1\}^n$ consisting of the single point p. Moreover, any realization of F(p) must have an output function F^t such that $F^t(q_p) = F(p)$. Thus, the position corresponding to p also corresponds to q_p.

Now consider the reconstruction procedure of Reference [1]. A separating function f^n is selected which, with appropriate gaps $u^n{:}\ell^n$, will realize the n-level reduced function $F^t(q_p) = F(p)$. Unless specifically indicated, such n-level realizations, $\langle f^n \rangle_{u^n:\ell^n}$, will not contain negative weights or inverters. Appendix 1 gives several possible n-level realizations.

Consider some n-level realization $\langle f^n \rangle_{u^n:\ell^n}$. If G_i denotes an arbitrary gate of $\langle f^n \rangle_{u^n:\ell^n}$, then let y_i^n denote the Boolean function realized by G_i. Notice that y_i^n is a constant function of either 1 or 0. Referring to Appendix 1, an example of an n-level realization and the corresponding Boolean functions is

$$\langle f^n\rangle_{u^n:\ell^n} = \overset{G_0}{\langle} 0+\beta_1 \overset{G_1}{\langle} 0+\beta_2 \overset{G_2}{\langle} 0\rangle_{\infty:o}\rangle_{o:-\infty}\rangle_{\beta_1:-\infty} \tag{5}$$

where $y_2^n = 0,\ y_1^n = 1,$ and $y_0^n = 1.$

The following definition will define a set of constants which can be associated with a n-level realization.

Definition 14. Consider an arbitrary n-level realization $\langle f^n\rangle_{u^n:\ell^n}$. Let G_j and G_i denote arbitrary gates of $\langle f^n\rangle_{u^n:\ell^n}$ such that the output of G_i is an input to G_j. Define C_j as

$$C_j = \sum_{i=1}^{m} \beta_i y_i^n + k_j$$

where β_i is the weight of input y_i^n to G_j, and k_j is the weight of a constant input to G_j.

For instance, consider Equation 5. The set of constants are: $C_2=0, C_1=0, C_0=\beta_1$.

The following definition will define a branch which corresponds to the n-level of the tree. The definition is similar to the previous definition of a branch except that it is defined for an n-level realization $\langle f^n\rangle_{u^n:\ell^n}$.

Definition 15. Let G_i denote an arbitrary gate in $\langle f^n\rangle_{u^n:\ell^n}$ and let B* denote an arbitrary "output connected subset" of $\langle f^n\rangle_{u^n:\ell^n}$. The set B* is defined as a n-level 1(0) branch $B_1^n(B_0^n)$ and is said to realize the constant function 1(0) if, when $\langle f^n\rangle_{u^n:\ell^n}$ is modified such that all the gates not contained in B* have 0(1) output, then C_i for the modified n-level realization satisfies the condition $C_i \geq u_i^n$ $(C_i \leq \ell_i^n)$ $\forall G_i \in B^*$.

As an example of a n-level 0 branch, consider the following n-level realization, which can be obtained from Appendix 1.

$$\qquad\qquad G_0 \qquad G_1 \qquad G_2$$

$$\langle f^n \rangle_{u^n:\ell^n} = \langle 0 + \beta_1 \langle 0 + \beta_2 \langle 0 \rangle_{0:-\infty} \rangle_{\infty:\beta_2} \rangle_{\infty:0}$$

Consider the set $\{G_1, G_0\}$ as a possible n-level 0 branch. According to Definition 15, the condition $C_0 \leq \ell_0^n$ and $C_1 \leq \ell_1^n$ must exist when the output of G_2 is a 1. For this condition, the modified n-level realization becomes

$$\langle 0 + \beta_1 \langle 0 + \beta_2 \rangle_{\infty:\beta_2} \rangle_{\infty:0}.$$

Hence, $C_1 = \beta_2$ and $C_0 = 0$. Since $\ell_1^n = \beta_2$ and $\ell_0^n = 0$, it follows that the set $\{G_1, G_0\}$ is a n-level 0 branch.

The following lemma will now give a relationship between n-level branches and branches of a positive realization. The lemma assumes a configuration of gates for the n-level realization such that no void ranges occur during reconstruction (i.e., no additional gates are necessary during reconstruction).

<u>Lemma 3</u>. Consider an arbitrary point $q_p \in \{0,1\}^{2n}$ such that $F^t(q_p) = 1(0)$. Assume that no void ranges occur during reconstruction. Given that a set of gates B^* is a n-level 1(0) branch $B_1^n(B_0^n)$ which realizes $F^t(q_p)$ on the n-level of the tree, then B^* is a 1(0) branch $B_1(B_0)$ which realizes q_p in the final realization.

Proof. We will prove this result by using induction.

Let G_j be an arbitrary gate of B^* which is on the r logic level of $\langle f^n\rangle_{u^n:\ell^n}$; and, hence of $\langle \tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$. Let G_i be one of the m arbitrary gates whose output is an input to G_j (i.e., G_i must be on the r+1 or greater logic level). Then the n-level realization for G_j can be expressed as

$$\langle f_j^n\rangle_{u_j^n:\ell_j^n} = \langle 0 + \sum_{i=1}^{m'} \beta_i y_i^n + \sum_{i=m'+1}^{m} \beta_i y_i^n \rangle_{u_j^n:\ell_j^n}$$

where $1 \leq i \leq m' \Leftrightarrow G_i \epsilon B_1^n$

and $m' + 1 \leq i \leq m \Leftrightarrow G_i \not\epsilon B_1^n$.

Referring to the previous equation, when all $G_i \not\epsilon B_1^n$ have 0 output, then

$$C_j = \sum_{i=1}^{m'} \beta_1 .$$

Since B^* is an n-level 1 branch, it follows from Definition 15 that

$$C_j = \sum_{i=1}^{m'} \beta_i \geq u_j^n . \qquad (6)$$

From properties of reconstruction it is known that if $F^t(q_p) = 1$, then

$$u_j^n + \sum_{k=1}^{n} a_k x_k^* (q_p) \geq \tilde{u}_j \qquad (7)$$

where $a_k \geq 0$, x_k^* is a literal of x_k, and $\tilde{u}_j$ is the upper gap for G_j in the final realization.

Assume that Lemma 3 is true for all G_i, where $1 \leq i \leq m'$. Thus, all G_i, for $1 \leq i \leq m'$, are elements of B_1 for the point q_p in the final realization. Therefore,

$$\tilde{f}_j (q_p) = \sum_{k=1}^{n} a_k x_k^* (q_p) + \sum_{i=1}^{m'} \beta_i \text{, when all } G_i \not\epsilon B_1 \text{ have 0 output.} \qquad (8)$$

From Equations 6, 7, and 8, it follows that

$\tilde{f}_j(q_p) \geq \tilde{u}_j$, when all $G_i \not\in B_1$ have 0 output.

Hence, G_j is an element of B_1 for the point q_p in the final realization.

Now consider an arbitrary input gate G_i of B_1^n. (For this case $u_i^n = 0$). It follows from the above reasoning that the gate will be an element of B_1 for the point q_p in the final realization. Thus, it follows from induction that B_1 will be a 1 branch for the point q_p in the final realization.

The proof concerning 0 branches is similar and will be omitted.

Δ

Definition 16. Let $\langle f \rangle_{u:\ell}$ denote an arbitrary realization of the Boolean function F. Consider the real numbers u' and ℓ' such that $u > u' > \ell' > \ell$. The gaps $u:\ell'$, $u':\ell$, or $u':\ell'$ are defined as reduced gaps of $u:\ell$.

Consider a realization $\langle f \rangle_{u:\ell}$ of the Boolean function F. Obviously, $u:\ell$ can be replaced by a reduced gap and the Boolean function F is still realized, provided T is properly selected. For instance, consider the gate G_0 of the Equation 5 which has a n-level gap of $\beta_1:-\infty$. A possible reduced gap is $o:-\infty$. The n-level realization then becomes

$$\begin{array}{c} \overset{G_0}{\langle 0 + \beta_1} \; \overset{G_1}{\langle 0 + \beta_2} \; \overset{G_2}{\langle 0 \rangle}_{\infty:o} \rangle_{o:-\infty} \rangle_{o:-\infty} . \end{array} \qquad (9)$$

This still realizes the Boolean function 1 and $\{G_1, G_0\}$ is still a n-level 1 branch. However, now by Definition 15, the gate G_0 is also a n-level 1 branch, where before it was not. Hence, by Lemma 3, if Equation 9 is used to realize a specific $F^t(q_p)$ on the n-level of the tree, then the 1 branches $\{G_0\}$ and $\{G_0, G_1\}$ will both realize q_p in the final realization. Thus, reduced n-level gaps provide a means for obtaining a final realization such that the point q_p will be realized by more than one branch.

The value of a point q_p being realized by more than one branch follows from Theorem 3. For example, let K^α and K^β be two 1 subcubes of F which have the set of points $\{p_j\}$ in common and let $\{q_p^j\}$ be the corresponding set of image points. Let S^α and S^β be the corresponding image subcubes. Clearly, $\{q_p^j\}$ belongs to the set of points common to S^α and S^β. Assume that for some positive realization S^α and S^β are realized by the 1 branches B^α and B^β, respectively, where $B^\alpha \neq B^\beta$. It follows, from Theorem 3, that the realization will not contain a logic 1 hazard within K^α and K^β. However, notice that the points of $\{q_p^j\}$ are realized by both B^α and B^β.

Once the set of n-level gaps are chosen the reconstruction process of Reference [1] is a technique for obtaining the coefficients of the independent input variables for each gate of the network. In some realizations, the restricted n-level range, caused by using a reduced n-level gap for some arbitrary $F^t(q_p)$, will not have any effect upon the coefficients. When such a case occurs, the reduced n-level gap is referred to as an unnecessary reduced n-level gap. However, if the reduced n-level gap does effect the final coefficient, it is referred to as a <u>necessary</u> reduced n-level gap. Example 3 illustrates both types of reduced n-level gaps.

The following theorem can now be used for synthesizing a logic hazard free threshold network directly from the Boolean function F.

<u>Theorem 5</u>. Let $\{P_1^i\}$ $(\{P_0^i\})$ be the set of 1(0) prime implicants of F, let $\{\{q_p\}_1^i\}$, $(\{\{q_p\}_0^i\})$ be the corresponding collection of sets of image points, and let $\{\{F^t(q_p)\}_1^i\}$, $(\{\{F^t(q_p)\}_0^i\})$ be the corresponding collection of sets of n-level reduced functions. A final realization $\langle\tilde{f}\rangle_{\tilde{u}:\tilde{\ell}}$ of F will not contain any logic 1(0) hazards if (1) the n-level gaps are assigned such that there exists at least one n-level 1(0) branch which realizes all $F^t(q_p) \epsilon \{F^t(q_p)\}_1^i$, $(\{F^t(q_p)\}_0^i)$ for all $\{F^t(q_p)\}_1^i \epsilon \{\{F^t(q_p)\}_1^i\}$, $(\{F^t(q_p)\}_0^i \epsilon \{\{F^t(q_p)\}_0^i\})$ and (2) reconstruction is possible without adding any additional gates.

<u>Proof</u>. Let P_1' be an arbitrary 1 prime implicant of F, let $\{q_p\}_1'$ be the corresponding set of image points, and let $\{F^t(q_p)\}_1'$ be the corresponding set of n-level reduced functions. Assume that the n-level gaps are assigned such that all $F^t(q_p) \epsilon \{F^t(q_p)\}_1'$ are realized by the same n-level 1 branch and that reconstruction is possible without void ranges. By Lemma 3, all $q_p \epsilon \{q_p\}_1'$ will be realized by the same 1 branch in the final realization. Hence, by Theorem 3, P_1' will not contain any logic 1 hazards.

The proof concerning 0 prime implicants is similar and will be omitted. Δ

Examples 3 and 4 will illustrate the application of Theorem 5. However, several additional facts must be considered first.

In general, a set of n-level gaps, which will yield a hazard free solution, will not be known. Therefore, suppose that while trying to obtain a hazard free solution, condition (1) of Theorem 5 is satisfied but condition (2) is not. Two alternatives exist, (1) a procedure analogous to Reconstruction III of Reference [1] or (2) an additional threshold gate or gates can be added in such a way as to remedy the situation.

Only the second alternative will be considered here. Assume that an inconsistency occurs for a gate G_j on the k-level of the tree. Normally one determines the inconsistency, uses Theorem D to add the necessary gate(s) in such a manner that the inconsistency is removed, and then continues on up the tree. However, if the final realization is to be logic hazard free, Theorem 4 must also be satisfied; hence, the application of Theorem D is restricted in the following manner.

The set of n-level gaps consisting of the n-level gaps for the additional gate(s) plus the changed n-level gaps for the previously chosen gates must satisfy condition (1) of Theorem 5. In practice, the most straightforward procedure for satisfying this restriction of Theorem D is the following.

Determine the inconsistency on the k-level of the tree and then, instead of adding gates on the k-level of the tree, add the gates on the n-level of the tree in such a manner that (1) the inconsistency is removed from the k-level of the tree and (2) condition (1) of Theorem 5 is satisfied. The gates that need to be added on the n-level of the tree, to remove the inconsistency of the k-level, can easily be determined by tracing the gaps that caused the inconsistency to the bottom of the tree. Example 4 will illustrate this.

Assume that all additional gates are added in this manner. It is proved in Reference [7] that a final realization can always be obtained with this restriction on Theorem D. Clearly the final realization will be hazard free.

The next fact concerns the application of Theorem 5. In order to apply Theorem 5 it is necessary to associate each n-level reduced function $F(p) = F^t(q_p)$ with a specific set of prime implicants of F. Namely, the set for which the corresponding point p is an element. For example, assume p is an element of the 1 prime implicants P_1^3, P_1^6, and P_1^7. Hence, $F^t(q_p)$ must be associated with P_1^3, P_1^6, and P_1^7. In which case, $F^t(q_p) = 1$ can be labeled 1_3, 1_6, and 1_7 on the n-level of the tree. A similar statement exists for 0 prime implicants. Now, from Theorem 5 the final realization will be hazard free if (1) all n-level points bearing the same label are realized by the same n-level branch and (2) reconstruction is possible without additional gates. Examples 3 and 4 will illustrate this procedure. Also, Chapter 6 of Reference [7] gives an algorithm for identifying the points at the bottom of the tree.

The last fact to be considered is concerned with incomplete functions. A function is said to be incomplete if for some $p \epsilon \{0, 1\}^n$, F(p) is not specified as either 1 or 0. Such p' s are called "don' t cares". This type of Boolean function should also be considered when studying hazards in threshold gate networks. A method has been presented for synthesizing incomplete logical functions by threshold gate networks [1]. The method, in effect, assigns the n-level gaps of the "don' t cares" points to be $\infty : -\infty$. Therefore, when assigning the n-level gaps in accordance to Theorem 5 (i.e., such that a logic hazard will not occur) the n-level gaps of the don' t care points are assigned as $\infty : -\infty$. Unfortunately, however, by this method one has no control of the logic hazards associated with the don' t care points.

In summary, the following procedure is outlined for obtaining a hazard free threshold network.

Procedure 5.

(1) Using the function tree, decompose F in the usual manner.

(2) Using the method previously described, identify the reduced functions $F^t(q_p)$ at the bottom of the tree with their associated prime implicants.

(3) Referring to Theorem 5, assign the n-level gaps such that there exists at least one n-level branch which realizes all of the reduced functions $F^t(q_p)$ which bear the same label.

(4) Reconstruct in the normal way, if no void common ranges occur the final realization is hazard free; whereas, if a void common range occurs go to Step 5.

(5) Using Theorem D, add a sufficient number of gates to eliminate the void common range. However, the set of n-level gaps consisting of the n-level gaps for the additional gates plus the changed n-level gaps for the previously chosen gates must satisfy Step (3).

(6) Repeat Steps (3), (4), and (5) until a final realization is obtained.

The following two examples will illustrate Procedure 5.

Example 3. Consider the Boolean function

$$F = x_1x_2 + \bar{x}_2x_3 + x_3x_4. \tag{9}$$

The Karnaugh map is shown in Figure 3. The problem is to obtain a hazard free threshold realization directly from the Boolean function F by application of Theorem 5. The first step is to obtain the prime implicants of F. These are obtained from Figure 3(a) and are given in Figure 3(b).

Decompose F by removing x_1, x_2, x_3, and x_4, respectively. The resulting function tree is shown in Figure 4. The prime implicants

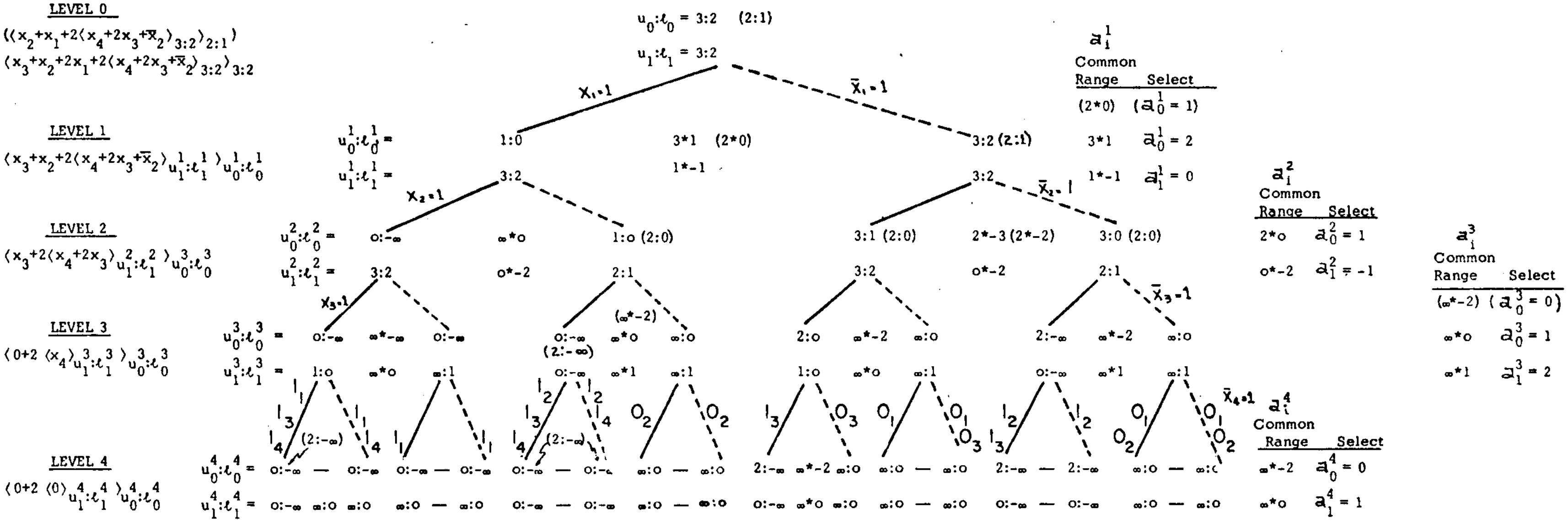
		x_3x_4			
		00	01	11	10
	00	0	0	1	1
	01	0	0	1	0
x_1x_2	11	1	1	1	1
	10	0	0	1	1

1 prime implicants	n-level branch
$\{p \mid x_1=1,\ x_2=1\}=P_1^1$	$\{G_0\}$
$\{p \mid x_2=0,\ x_3=1\}=P_1^2$	$\{G_1, G_0\}$
$\{p \mid x_3=1,\ x_4=1\}=P_1^3$	$\{G_1, G_0\}$
$\{p \mid x_1=1,\ x_3=1\}=P_1^4$	$\{G_0\}$
0 prime implicants	
$\{p \mid x_1=0,\ x_3=0\}=P_0^1$	$\{G_1, G_0\}$
$\{p \mid x_2=0,\ x_3=0\}=P_0^2$	$\{G_1, G_0\}$
$\{p \mid x_1=0,\ x_2=1,\ x_4=0\}=P_0^3$	$\{G_1, G_0\}$
(b)	(c)

Figure 3. Karnaugh Map, Prime Implicant List, and n-level Branch Assignment for Example 3.

that each n-level reduced function corresponds to are identified at the bottom of the tree. For example the reduced function $F^t(x_1=1,\ x_1'=0,\ x_2=1,\ x_2'=0,\ x_3=1,\ x_3'=0,\ x_4=1,\ x_4'=0)$, henceforth to be denoted as $F^t(10,\ 10,\ 10,\ 10)$, corresponds to P_1^1, P_1^3, and P_1^4.

The next step is to assign the n-level gaps according to Step (3) of Procedure 5. Referring to Figure 4, all of the points labeled $1_1(0_1)$ should be realized by the same n-level 1(0) branch, etc. Disregarding the terms in parentheses, one such assignment is given in Figure 4.



NOTE: $u_i^q:\ell_i^q$ = q level gap for gate G_i.

Figure 4. Function Tree and Realization for Example 3.

For example, consider the four reduced functions labeled 1_4. The n-level reduced function $F^t(10, 10, 10, 01)$ is realized by the n-level realization $\langle 0+2\langle 0\rangle_{\infty:0}\rangle_{0:-\infty}$. Whereas, the other three reduced functions labeled 1_4 are realized by the n-level realization $\langle 0+2\langle 0\rangle_{0:-\infty}\rangle_{0:-\infty}$. Hence, $F^t(10, 10, 10, 01)$ is realized by the n-level 1 branch consisting of the gate $\{G_0\}$; whereas, the other three are realized by the n-level 1 branches consisting of both $\{G_0\}$ and $\{G_1, G_0\}$. Thus, all four are realized by the n-level branch, $\{G_0\}$. Similarly the n-level branch which corresponds to each prime implicant of F is shown in Figure 3(c).

Figure 4 shows that reconstruction is possible, and that the final realization is

$$\langle f\rangle_{u:\ell} = \langle x_3+x_2+2x_1+2\ \langle x_4+2x_3+\overline{x}_2\rangle_{3:2}\rangle_{3:2}.$$

Thus, condition (2) of Theorem 5 is satisfied. Therefore, the final realization will not contain any logic hazards.

Consider the realization which results when $F^t(10,10,10,10)$, $F^t(10,01,10,10)$, and $F^t(10,01,10,01)$ are assigned normal n-level gaps (see Figure 4). This assignment and the reconstruction changes caused by this assignment are enclosed in parentheses in Figure 4, the final realization being

$$\langle f\rangle_{u:\ell} = \langle x_2+x_1+2\ \langle x_4+2x_3+\overline{x}_2\rangle_{3:2}\rangle_{2:1}.$$

Notice that the 1 prime implicant $\{p\,|\,x_1=1,\ x_3=1\}$ is not contained in the latter realization. It will therefore contain a logic 1 hazard. Also, notice that on comparing the two realizations we see that the prime implicant can be realized without requiring any additional gates. This is not possible with conventional elements such as AND, OR, NAND, NOR, or Relays (i.e., a conventional element can only realize one prime implicant).

By comparing the two previous reconstructions, it can be determined that the reduced n-level gap for $F^t(10,10,10,10)$ is an unnecessary reduced n-level gap; whereas, the reduced n-level gaps for $F^t(10,01,10,10)$ and $F^t(10,01,10,01)$ are necessary reduced n-level gaps. By definition, if a gap is an unnecessary reduced n-level gap, the normal gap could be used and the final realization would be the same. However, when the n-level gaps are being assigned it is not known if a reduced n-level gap is an unnecessary reduced n-level gap or not. Therefore, to obtain a logic hazard free realization, the best approach is to apply Theorem 5 and assume all reduced n-level gaps are necessary reduced n-level gaps.

Example 4. Consider the Boolean function

$$F = x_1\overline{x}_3 + \overline{x}_1x_3 + \overline{x}_2x_4 + x_2\overline{x}_4 \cdot$$

The Karnaugh map is shown in Figure 5. Again the problem is to obtain a hazard free threshold realization directly from the Boolean function F by application of Theorem 5. The first step is to obtain the prime implicants of F. Referring to Figure 5(a), it is obvious that a logic 0 hazard can not occur; hence, only the 1 prime implicants of F need be considered.

x_3x_4

x_1x_2	00	01	11	10
00	0	1	1	1
01	1	0	1	1
11	1	1	0	1
10	1	1	1	0

(a)

<u>1 prime implicants</u>

$P_1^1=\{p|x_2=1,x_4=0\}$

$P_1^2=\{p|x_1=1,x_3=0\}$

$P_1^3=\{p|x_1=0,x_3=1\}$

$P_1^4=\{p|x_2=0,x_4=1\}$

(b)

Figure 5. Karnaugh Map for Example 4.

Decompose F by removing x_1, x_2, x_3, and x_4, in that order. The resulting function tree is shown in the following figure. The prime implicants that each $F^t(q_p)$ correspond to are identified at the bottom of the tree.

The next step is to obtain n-level realizations and to assign n-level gaps according to Step (3) of Procedure 5. Since F is not unate, the initial n-level realization must contain at least two gates. Consider the n-level assignment labeled A in Figure 6, where the n-level realization is

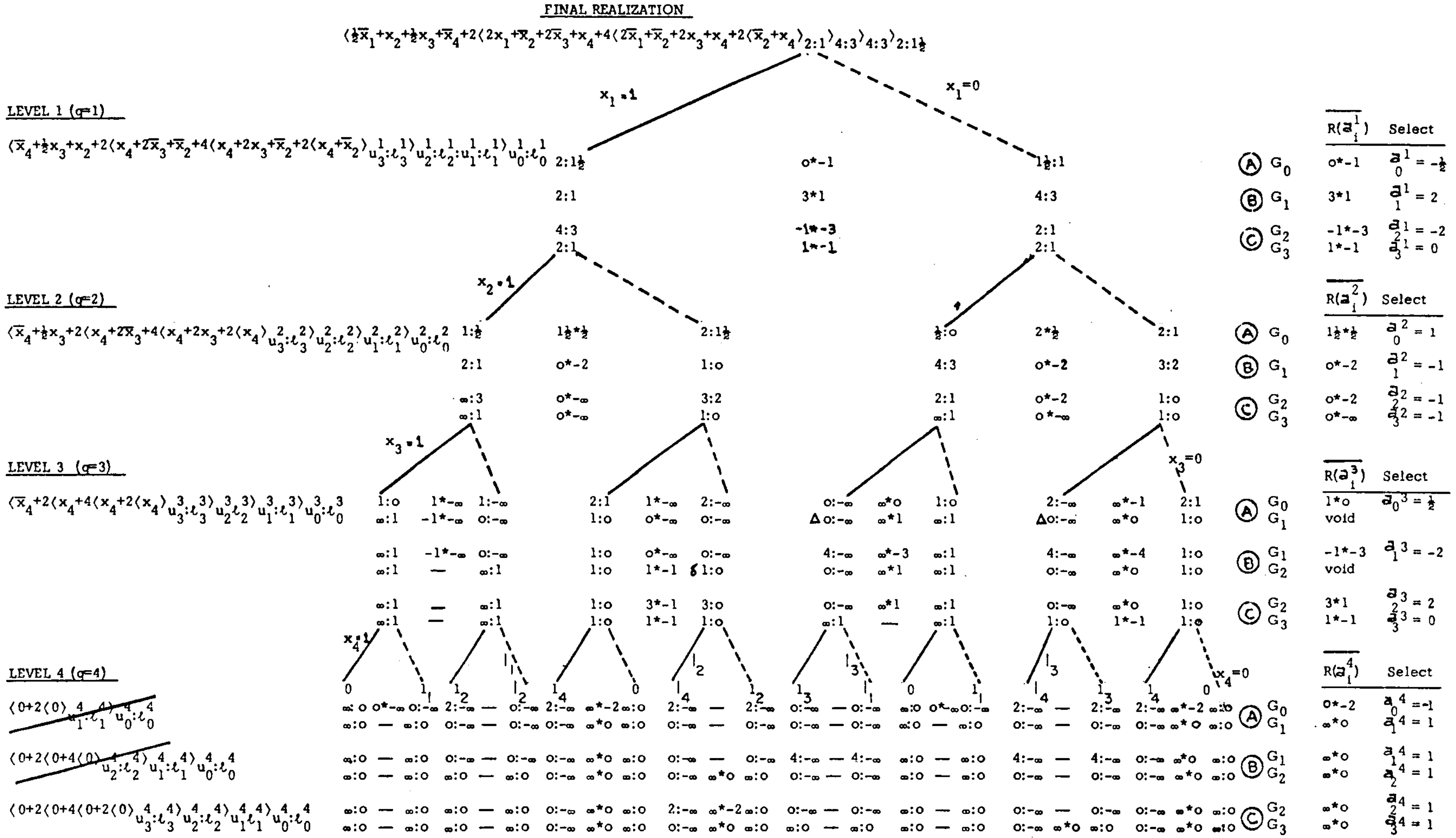
$$\langle f^n\rangle_{u^n:\ell^n} = \langle \overset{G_0}{0} + 2\ \overset{G_1}{\langle 0\rangle}_{u_1^n:\ell_1^n}\rangle_{u_0^n:\ell_0^n}\ .$$

Referring to Definition (15) and assignment A, the n-level reduced functions labeled 1_1 are realized by the n-level 1 branch $\{G_0\}$, whereas, the other n-level reduced functions which are 1 are realized by the n-level 1 branch $\{G_1, G_0\}$. Hence, a hazard free threshold realization will be obtained if reconstruction is possible. However, a void common range occurs for G_1 on the 3rd level of reconstruction. Therefore, an additional threshold gate or gates must be added to complete reconstruction. However, before considering the addition of a gate or gates to correct the void range for G_1, further reconstruction for G_0 will be considered. Referring to Figure 6, it is seen that reconstruction of G_0 can be completed without any void common range.

Now consider the inconsistency which caused the void common range for G_1 on the 3rd level of the tree. Referring to Figure 6, if the gaps identified by the symbol "Δ" are increased by more than 2 the void common range will not occur. Hence, the n-level gaps for $F^t(01,10,10,10)$, $F^t(01,10,10,01)$, $F^t(01,01,10,10)$, and $F^t(01,01,10,01)$ must be increased by more than 2. Therefore, change these n-level gaps from $o:-\infty$ to $4:-\infty$ by adding the gate G_2 as input to G_1, where $\beta_2=4$. The n-level assignment for G_1 and G_2, denoted by B, is shown in Figure 6. The n-level realization is now

$$\langle f^n \rangle_{u^n:\ell^n} = \langle 0 + 2\, \langle 0 + 4\, \langle 0 \rangle_{u_2^n:\ell_2^n} \rangle_{u_1^n:\ell_1^n} \rangle_{u_0^n:\ell_0^n} .$$

Referring to assignment B and the previous assignment for G_0, the n-level reduced functions labeled 1_1 and 1_2 are realized by the n-level 1 branch $\{G_0\}$ and $\{G_1, G_0\}$, respectively. Whereas, the other n-level reduced functions which are 1 are realized by the n-level 1



NOTE: 1. $\overline{R(a_i^q)}$ implies the "common range" of a^q for the gate G_i

2. The symbol "—" implies range ∞*-∞

Figure 6. Function Tree and Realization for Example 4.

branch $\{G_2, G_1, G_0\}$. By Theorem 5, if reconstruction is possible the final realization will not contain any logic hazards.

Before continuing, notice the following two things:

(1) Since reconstruction was complete for G_0, the image points of P_1^1 will be realized by the 1 branch consisting of G_0 regardless of which gates are added.

(2) In this example when the inconsistency is removed from the 3rd level of G_1, it so happens that condition (1) of Theorem 5 is also satisfied. This is not true in general, and in such cases some addition n-level gaps for G_1 must be changed.

The process of reconstruction will now be continued. Referring to Figure 6, reconstruction of G_1 can now be completed without any void common ranges. Next consider the reconstruction for G_2. A void common range occurs for G_2 on the 3rd level of the tree. If the gap, identified by the symbol "δ", is increased by any positive amount the inconsistency will not occur. Hence, the n-level gap of $F^t(10,01,01,10)$ is increased from $o{:}-\infty$ to $2{:}-\infty$ by adding G_3 as an input to G_2, where $\beta_3 = 2$. The n-level assignment for G_2 and G_3, denoted by C, is shown in Figure 6. The corresponding n-level realization is also shown in Figure 6. Reconstruction is now possible, the final realization being

$$\langle f\rangle_{u:\ell} = \langle \tfrac{1}{2}\overline{x}_1+x_2+\tfrac{1}{2}x_3+\overline{x}_4+2\,\langle 2x_1+\overline{x}_2+2\overline{x}_3+x_4$$
$$+4\langle 2\overline{x}_1+\overline{x}_2+2x_3+x_4+2\langle \overline{x}_2+x_4\rangle_{2:1}\rangle_{4:3}\rangle_{4:3}\rangle_{2:1\frac{1}{2}}\;.$$

Hence, from Theorem 5, the realization will not contain any logic hazards.

Theorem 5 gives a means for obtaining a general threshold realization which is hazard free. However, the following special realizations are also of interest.

(1) Obviously if the desired realization can contain logic 0 hazards, but not logic 1 hazards, then only the 1 prime implicants of F must be contained in the realization. Hence, in Theorem 5, the n-level reduced functions $F^t(q_p)$ which are 0 can be chosen arbitrarily. A similar statement exists if the realization can contain logic 1 hazards, but not 0 hazards.

(2) Eichelberger [2] has proved that a sum-of-product (product-of-sum) realization will not contain any logic hazards if each 1(0) prime implicant of F is realized by a unique AND (OR) gate. Consider the case where the number of 0 prime implicants of F is less than the number of 1 prime implicants of F (i.e., the number of 1 prime implicants of $\bar{F}$ is less than the number of 1 prime implicants of F). From the previous statement, the product-of-sum hazard free realization requires fewer gates than the sum-of-product hazard free realization. If the number of 1 prime implicants is less, the sum-of-product realization would require fewer gates.

The following theorem will show that a similar situation exists for the following type of two level threshold realization.

Theorem 6. Let $\{P_1^i\}$ be the set of 1 prime implicants of the Boolean function F and let s_1^i be the product Boolean function which realizes the 1 prime implicant P_1^i. Then the Boolean function F can be expressed as

$$F = s_1^1 + \cdots + s_1^i + \cdots + s_1^n, \text{ where } 1 \leq i \leq n. \qquad (10)$$

If the above equation can be expressed as a sum of m Boolean threshold function i.e.

$$F_{T_1} + \cdots + F_{T_{m-1}} + F_{T_m}, \qquad (11)$$

then it can be realized by the two level positive threshold realization

$$\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha} = \langle \beta_1 F_{T_1} + \cdots + \beta_{m-1} F_{T_{m-1}} + a_1 x_1^* + \cdots + a_n x_n^* \rangle_{u_\alpha : \ell_\alpha}$$

where $\beta_j = u_\alpha$ for $1 \leq j \leq m-1$ and $\langle a_1 x_1^* + \cdots + a_n x_n^* \rangle_{u_\alpha : \ell_\alpha}$ realizes F_{T_m}; moreover, $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$ will not contain any logic hazards.

Proof. Since $\beta_j = u_\alpha$, it follows that Equation 11 and hence, F, can be realized with m gates. Now consider the logic hazards.

All of the reduced functions such that $F^t(q_p) = 0$ are realized by the same 0 branch; namely, the 0 branch which consists of all of the gates contained in $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$. Thus, from Theorem 4, $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$ will not contain any logic 0 hazards. Obviously, $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$ will not contain any logic 1 hazards. Δ

The realization $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$ is equivalent to a sum-of-product realization.

Now consider the complement Boolean function $\bar{F}$ and express it in the form of Equations 10 and 11, respectively, and let m* denote the number of Boolean threshold functions obtained in the latter expression. From Reference [6], m^* will not necessarily equal m; it may be greater or less. From Theorem 6, there exists a realization $\langle f^*_\alpha \rangle_{u^*_\alpha : \ell^*_\alpha}$ which contains m^* gates, realizes $\overline{F}$, and does not contain any logic hazards. From Theorem C, there exists a corresponding complement realization $\langle \overline{f}^*_\alpha \rangle_{\overline{u}^*_\alpha : \overline{\ell}^*_\alpha}$ which realizes F with m^* gates. Obviously, $\langle \overline{f}^*_\alpha \rangle_{\overline{u}^*_\alpha : \overline{\ell}^*_\alpha}$ will not contain any logic hazards.

Therefore, if $m^* < m$ the realization $\langle \overline{f}^*_\alpha \rangle_{\overline{u}^*_\alpha : \overline{\ell}^*_\alpha}$ will require fewer gates than $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$; whereas, if $m^* > m$ the realization $\langle f_\alpha \rangle_{u_\alpha : \ell_\alpha}$ requires fewer gates.

(3) Theorem 5 and Theorem 6 give means for synthesizing threshold networks which do not contain any logic hazards. However, from another point of view, they also enable us to design threshold networks which contain certain specified logic hazards and only those specified. The utilization of such networks and their designed with conventional type gates has been considered in a paper by Eichelberger [3].

(4) One of the most common applications of hazard free combinational circuits is in the design of asynchronous systems. Moreover, in most asynchronous systems the assumption is made that only one input variable can change at a time. When such an assumption is made, it has been shown that it is not necessary to realize

all of the 1 and 0 prime implicants of F, [4] and [5]. In such a case, Theorem 5 can be changed accordingly, in that the set of necessary 1 and 0 prime implicants will be a subset of $\{P_1^i\}$ and $\{P_0^i\}$, respectively.

APPENDIX 1

Appendix 1 contains the possible n-level separating functions with n-level gaps for two gate and three gate - three level realizations. The parameters β can be selected equal to any number greater than unity. It should be selected equal to 2 or greater if it is desired not to decrease the gap length [1]. For the following table the n-level realizations are:

$$\langle 0+\beta_1 \langle 0 \rangle_{u_1:\ell_1} \rangle_{u_0:\ell_0} \qquad \text{2 gate}$$

$$\langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{u_2:\ell_2} \rangle_{u_1:\ell_1} \rangle_{u_0:\ell_0} \qquad \text{3 gate - 3 level}$$

where β_i, u_i, and ℓ_i correspond to the gate G_i.

NORMAL GAPS

n-level Separating Function	n-level Branch
$1 = \langle 0+\beta_1 \langle 0 \rangle_{0:-\infty} \rangle_{\beta_1:-\infty}$	$\{G_0, G_1\}$
$1 = \langle 0+\beta_1 \langle 0 \rangle_{\infty:0} \rangle_{0:-\infty}$	$\{G_0\}$
$1 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{0:-\infty} \rangle_{\beta_2:-\infty} \rangle_{\beta_1:-\infty}$	$\{G_0, G_1, G_2\}$
$1 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{0:-\infty} \rangle_{\infty:\beta_2} \rangle_{0:-\infty}$	$\{G_0\}$
$1 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{\infty:0} \rangle_{0:-\infty} \rangle_{\beta_1:-\infty}$	$\{G_0, G_1\}$
$1 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{\infty:0} \rangle_{\infty:0} \rangle_{0:-\infty}$	$\{G_0\}$
$0 = \langle 0+\beta_1 \langle 0 \rangle_{\infty:0} \rangle_{\infty:0}$	$\{G_0, G_1\}$
$0 = \langle 0+\beta_1 \langle 0 \rangle_{0:-\infty} \rangle_{\infty:\beta_1}$	$\{G_0\}$
$0 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{\infty:0} \rangle_{\infty:0} \rangle_{\infty:0}$	$\{G_0, G_1, G_2\}$
$0 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{\infty:0} \rangle_{0:\infty} \rangle_{\infty:\beta_1}$	$\{G_0\}$
$0 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{0:-\infty} \rangle_{\infty:\beta_2} \rangle_{\infty:0}$	$\{G_0, G_1\}$
$0 = \langle 0+\beta_1 \langle 0+\beta_2 \langle 0 \rangle_{0:\infty} \rangle_{\beta_2:-\infty} \rangle_{\infty:\beta_1}$	$\{G_0\}$

REDUCED GAPS

n-level Separating Function	n-level Branches
$1 = \langle 0+\beta_1\langle 0\rangle_{0:-\infty}\rangle_{0:-\infty}$	$\{G_0,G_1\}, \{G_0\}$
$1 = \langle 0+\beta_2\langle 0+\beta_1\langle 0\rangle_{0:-\infty}\rangle_{0:-\infty}\rangle_{\beta_1:-\infty}$	$\{G_0,G_1,G_2\},\{G_0,G_1\}$
$1 = \langle 0+\beta_2\langle 0+\beta_1\langle 0\rangle_{0:-\infty}\rangle_{0:-\infty}\rangle_{0:-\infty}$	$\{G_0,G_1,G_2\},\{G_0,G_1\},\{G_0\}$
$1 = \langle 0+\beta_2\langle 0+\beta_1\langle 0\rangle_{\infty:0}\rangle_{0:-\infty}\rangle_{0:-\infty}$	$\{G_0,G_1\}, \{G_0\}$
$0 = \langle 0+\beta_1\langle 0\rangle_{\infty:0}\rangle_{\infty:\beta_1}$	$\{G_0,G_1\}, \{G_0\}$
$0 = \langle 0+\beta_2\langle 0+\beta_1\langle 0\rangle_{\infty:0}\rangle_{\infty:\beta_2}\rangle_{\infty:0}$	$\{G_0,G_1,G_2\},\{G_0,G_1\}$
$0 = \langle 0+\beta_2\langle 0+\beta_1\langle 0\rangle_{\infty:0}\rangle_{\infty:\beta_2}\rangle_{\infty:\beta_1}$	$\{G_0,G_1,G_2\},\{G_0,G_1\},\{G_0\}$
$0 = \langle 0+\beta_2\langle 0+\beta_1\langle 0\rangle_{0:-\infty}\rangle_{\infty:\beta_2}\rangle_{\infty:\beta_1}$	$\{G_0,G_1\}, \{G_0\}$

BIBLIOGRAPHY

[1] P. M. Lewis, II, and C. L. Coates, "Threshold Logic," John Wiley and Sons, Inc., 1967.

[2] E. B. Eichelberger, "Hazard Detection in Combinational and Sequential Switching Circuits," IEEE Conference Record on Switching Circuits Theory and Logical Design, 111-121, 1964.

[3] E. B. Eichelberger, "The Synthesis of Combinational Circuits Containing Hazards," Princeton University, Digital Systems Laboratory, T.R. No. 17, March 1962.

[4] D. A. Huffman, "The Design and Use of Hazard Free Switching Networks," J. ACM, Vol. 4, No. 1, 47-62, January 1957.

[5] E. J. McCluskey, "Transients in Combinational Logic Circuits," Redundancy Techniques for Computing Systems, Spartan Book Co., 9-46, 1962.

[6] C. L. Sheng, "Compound Synthesis of Threshold Logic Networks for Realization of General Boolean Functions," IEEE Transactions on Electronic Computers, EC-14, No. 6, 798-814, Dec. 1965.

[7] A. B. Howe, and C. L. Coates, "A Study of Hazards in Threshold Networks, The University of Texas, Laboratories for Electronics and Related Science Research, T.R. No. 21, August 1966.