

Brief Contributions

Theory and Design of t -Unidirectional Error-Correcting and d -Unidirectional Error-Detecting Code

D. K. Ray-Chaudhuri, N. M. Singhi, S. Sanyal,
and P. S. Subramanian

Abstract—The basic theory of t -UEC d -UED codes is developed. Methods for construction of such codes from symmetric error-correcting and asymmetric error-correcting codes are developed. Some bounds for t -EC d -UED codes are improved. Encoding/decoding procedures for these codes are discussed.

Index Terms—Asymmetric error, code construction multiple error-correcting and error-detecting codes symmetric error unidirectional error.

I. INTRODUCTION AND STATEMENT OF RESULTS

The importance and benefits of error control for computer and communication systems are well recognized. Error-correcting/detecting codes that are effective against both symmetric and unidirectional errors are useful in protection against transient, intermittent and permanent faults [1]–[7]. In the past, codes which can correct all patterns of t or fewer symmetric errors and detect all unidirectional errors (t -EC AUED Codes) have been designed [4]–[6], [9].

Unidirectional errors have been found to be very commonly occurring in memory systems. Faults in address decoders, word lines, power supply, read/write circuits, etc. tend to generate unidirectional errors [7], [11]. It has been observed that unidirectional error-correcting and error-detecting codes can be designed that need fewer check bits than are required for the corresponding Symmetric codes [12]. Codes that correct single symmetric error and detect all unidirectional errors have been presented by Bose and Rao [4] and by Bose and Pradhan [2]. More efficient t -symmetric error-correcting all unidirectional error-detecting codes have been described in [6], [9].

This idea has been further extended to t - \underline{E} rror (Symmetric) \underline{C} orrecting $d(d \geq t)$ - \underline{U} nidirectional \underline{E} rror \underline{D} etecting Code (t - \underline{E} C d - \underline{U} ED code) and t - \underline{U} nidirectional \underline{E} rror \underline{C} orrecting $d(d \geq t)$ - \underline{U} nidirectional \underline{E} rror \underline{D} etecting Code (t - \underline{U} EC d - \underline{U} ED code) [1]. Putting an upper bound d on the correction/detection capability does not really restrict the utility of the codes. In practical cases, it is reasonable to assume that the number of unidirectional errors, even though large, is limited by the organization of the chips: e.g., in a byte/chip memory organization, a chip failure can cause at most

Manuscript received April 30, 1990; revised May 23, 1993. D. K. Ray-Chaudhuri was supported by NSA Grant No. 904-88-11-2034. This paper was written when N. M. Singhi was visiting Ohio State University, Columbus, OH 43210.

D. K. Ray-Chaudhuri is with the Department of Mathematics, Ohio State University, Columbus, OH 43210 USA; e-mail: dijan@math.ohio-state.edu.

N. M. Singhi is with the School of Mathematics, Tata Institute of Fundamental Research, Homi Bhabha Road, Bombay-400005, India; e-mail: singhi@tifrvax.tifr.res.in.

S. Sanyal is with the Computer Systems and Communications Group, Tata Institute of Fundamental Research, Homi Bhabha Road, Bombay-400005, India; e-mail: sanyal@tifrvax.tifr.res.in.

P. S. Subramanian is with the Department of Theoretical Computer Science, Tata Institute of Fundamental Research, Homi Bhabha Road, Bombay-400005, India.

IEEE Log Number 9402552.

eight bit (byte) errors. Thus, by restricting d (and consequently t , as $d \geq t$), one provides for fewer check bits than those required for t -EC AUED code, thus obtaining a more efficient code.

Lin and Bose, in their interesting paper [1], describe methods of constructing t -EC d -UED codes. However, their treatment of codes capable of correcting t or fewer unidirectional errors and detecting $d(d \geq t)$ or less unidirectional errors (t -UEC d -UED), was in error; this led them to the belief that such codes are t -EC d -UED codes. The aim of this paper is to correct this error and to develop an exact theory of such codes. We provide methods of construction of t -UEC d -UED codes. In this process we also improve upon some bounds for t -EC d -UED [1] codes.

We first define all the terms mentioned above.

\underline{x} and \underline{y} : two vectors of length n .

$w(\underline{x})$: weight of \underline{x} i.e., the number of 1's in \underline{x} .

$N(\underline{x}, \underline{y})$: number of $1 \rightarrow 0$ crossovers from \underline{x} to \underline{y}

$d(\underline{x}, \underline{y})$: Hamming Distance between \underline{x} and \underline{y} , i.e., $d(\underline{x}, \underline{y}) = N(\underline{x}, \underline{y}) + N(\underline{y}, \underline{x})$.

$d_a(\underline{x}, \underline{y})$: asymmetric distance between \underline{x} and \underline{y} , i.e., $d_a(\underline{x}, \underline{y}) = \max\{N(\underline{x}, \underline{y}), N(\underline{y}, \underline{x})\}$.

For vectors $\underline{x} = (x_1, x_2, \dots, x_i, \dots, x_n)$ and $\underline{y} = (y_1, y_2, \dots, y_i, \dots, y_n)$ we define $\underline{x} \subseteq \underline{y}$ iff $\forall i, x_i = 1 \implies y_i = 1$. $\underline{x} \subseteq \underline{y} \implies N(\underline{x}, \underline{y}) = 0$.

It is easy to prove that $d_a(\underline{x}, \underline{y})$ satisfies properties of a distance function.

Example: Let $\underline{x}=1100$, $\underline{y}=1010$, $\underline{z}=1111$

then $N(\underline{x}, \underline{y})=1$,

$N(\underline{y}, \underline{x})=1$, $N(\underline{x}, \underline{z})=0$, $N(\underline{z}, \underline{x})=2$

Thus, $\underline{x} \subseteq \underline{z}$.

Also, $d_a(\underline{x}, \underline{z}) = \max\{N(\underline{x}, \underline{z}), N(\underline{z}, \underline{x})\} = \max\{0, 2\} = 2$.

Now, suppose \underline{x} has been transformed into $\hat{\underline{x}}$ due to errors.

We say the following.

- \underline{x} has suffered *symmetric errors* if the errors contain both type of transitions $1 \rightarrow 0$ or $0 \rightarrow 1$.
- \underline{x} has suffered *unidirectional errors* if either $\hat{\underline{x}} \subseteq \underline{x}$ or $\underline{x} \subseteq \hat{\underline{x}}$, i.e., errors consist in transitions of either type, $1 \rightarrow 0$ or $0 \rightarrow 1$, but not both.
- \underline{x} has suffered *asymmetric errors* if errors are always of one type of transitions say $1 \rightarrow 0$, i.e., always $\hat{\underline{x}} \subseteq \underline{x}$.

Let e denote one of the above type of errors: symmetric, unidirectional or asymmetric. For the vector \underline{x} , we define a *ball* $B_e(\underline{x}, t)$ to be the set of all vectors \underline{y} which can be obtained from \underline{x} by at most t errors of type e .

A *code* C is a set of binary vectors of a given length, say n and let e be a given type of error and let ϕ indicate an *empty set*.

A code C is said to be the following.

- t -error detecting for errors of type e if $B_e(\underline{x}, t) \cap C = \{\underline{x}\} \forall \underline{x} \in C$.
- t -error correcting for errors of type e if $B_e(\underline{x}, t) \cap B_e(\underline{y}, t) = \phi \forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$.
- t -error correcting and d -error detecting, both of type e , $d \geq t$, if $B_e(\underline{x}, t) \cap B_e(\underline{y}, d) = \phi \forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$.
- t -error correcting for errors of type e and d -error detecting for errors of type e' if $B_e(\underline{x}, t) \cap B_{e'}(\underline{y}, d) = \phi$ and if $B_e(\underline{x}, t) \cap B_{e'}(\underline{y}, d) = \phi \forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$, where e' is another given type of error.

- 5) *t*-EC Code: *t*-symmetric error-correcting code *C* is a *t*-EC Code.
- 6) *t*-UEC Code: *t*-unidirectional error-correcting code *C* is a *t*-UEC code.
- 7) *d*-UED Code: *d*-unidirectional error-detecting code *C* is a *d*-UED code.

The aim of this brief contribution is to study the theory and design of codes capable of correcting *t* or fewer unidirectional errors and detecting up to a fixed number of *d* ($d \geq t$) unidirectional errors.

In this section, we describe all the results derived in this paper. In the next section, we give proof of all those results not proved in this section and discuss the encoding/decoding procedures.

The following theorem, proved in Section II, gives the necessary and sufficient conditions for a code to be *t*-UEC *d*-UED.

Theorem 1.1: A code *C* is capable of correcting *t*-unidirectional errors and detecting *d*-unidirectional errors iff the following conditions are satisfied $\forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$

- 1) if $N(\underline{x}, \underline{y}) = 0$, then $N(\underline{y}, \underline{x}) \geq d + t + 1$
- 2) if $0 < N(\underline{x}, \underline{y}) \leq t$, then $N(\underline{y}, \underline{x}) \geq d + 1$.

Remark 1.2: Lin and Bose [1] have claimed the following result [1, Theorem 2.1]. A code can correct *t*-unidirectional errors and detect *d*-unidirectional ($d \geq t$) errors iff it satisfies Property 1.3.

Property 1.3: $\forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$ either $t + d + 1 \leq d(\underline{x}, \underline{y})$ or $t + 1 \leq N(\underline{x}, \underline{y})$ and $t + 1 \leq N(\underline{y}, \underline{x})$.

The following example shows that these conditions are not necessary for a code to be *t*-UEC *d* ($d \geq t$)-UED.

Example 1.4: Let *C* be a code consisting of three vectors $\{\underline{x} = 110000000, \underline{y} = 001111100, \underline{z} = 010001111\}$.

Note that $N(\underline{x}, \underline{z}) = 1$ and $N(\underline{z}, \underline{x}) = 4$. Thus, *C* does not satisfy property (1.3) for $t = 2$ and $d = 3$. Theorem 1.5 confirms that *C* is not a 2-EC 3-UEC code. However, it can be easily checked that *C* is a 2-UEC 3-UEC code (to be proved later). The vector 000001100 can be obtained from both \underline{y} and \underline{z} by 3 unidirectional errors. Hence, *C* is not a 3-UEC code.

However, the following theorem proved in [1] is quite correct and shows in conjunction with Theorem 1.1 that conditions for *t*-UEC *d*-UED are quite different from those of *t*-EC *d*-UED in general.

Theorem 1.5: A code *C* is *t*-EC *d*-UED ($d \geq t$) iff it satisfies property 1.3 (Property 1.3 and Theorem 1.5 combined forms Theorem 2.2 of [1]).

Theorem 2.2, proved in [1], is quite correct.

The following two corollaries follow immediately from Theorem 1.1 by taking $d = t$ and $d = n$ respectively. The corollaries were first proved by Bose and Rao [4].

Corollary 1.6: A code *C* is *t*-UEC iff it satisfies the following property:

- $$\forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$$
- 1) $d_a(\underline{x}, \underline{y}) = \text{maximum} \{N(\underline{x}, \underline{y}), N(\underline{y}, \underline{x})\} \geq t + 1$
 - 2) if $N(\underline{x}, \underline{y}) = 0$ then $N(\underline{y}, \underline{x}) \geq 2t + 1$

A *t*-EC *n*-UED (or *t*-UEC *n*-UED) code of length *n* is said to be a *t*-EC AUED (or *t*-UEC AUED) code.

Corollary 1.7: The following are equivalent:

- 1) A code *C* is *t*-UEC AUED;
- 2) A code *C* is *t*-EC AUED;
- 3) $\forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}, N(\underline{x}, \underline{y}) \geq t + 1$.

Remark 1.8: From the above result it can be seen that a code *C* is 1 EC iff it is 1 UEC.

The following is a well known result, see ([4], [9]).

Theorem 1.9: A code *C* is a *t*-Asymmetric EC code iff $\forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y} d_a(\underline{x}, \underline{y}) \geq t + 1$

Thus *t*-UEC codes are precisely *t*-Asymmetric EC codes (satisfying condition (1) of the Corollary 1.6). The *t*-UEC codes also satisfy condition (2) of the same Corollary.

Some methods of construction of Unidirectional Error-Correcting and -Detecting Codes are described. These constructions are based on the *adding a tail* method which has been used for construction of *t*-EC AUED codes and other similar codes ([1], [6], [9]).

II. CONSTRUCTION

Let *C* and *T* be codes of length *n* and length *s* respectively and let $g : C \rightarrow T$ be a map. Let $C_g = \{\underline{x}g(\underline{x}) \mid \underline{x} \in C\}$ be a code of length $n + s$, whose words are obtained by juxtaposing \underline{x} with $g(\underline{x}), \underline{x} \in C$. The code C_g will be called as having been obtained from *C* by pasting the tail map *g* and $g(\underline{x})$ will be called the tail of \underline{x} . We will be using only particular types of maps *g*, where the tail $g(\underline{x})$ depends only on the weight of \underline{x} .

Let $T = \{t_1, t_2, \dots, t_m\}$ and *q* be any positive integer. Define a map $g_q : C \rightarrow T$ as follows: For each word $\underline{x} \in C$, we define,

$$g_q(\underline{x}) = t_j \text{ where } j = \left[\frac{w(\underline{x})}{q} \right] \pmod{m}, 1 \leq j \leq m. \quad (1)$$

The code C_{g_q} obtained from *C* by pasting tail map g_q will be called a code obtained from *C* by adding a *q*-multiple of tail *T* and is denoted by CT^q . In particular $CT^1 = C.T$ is said to be the code obtained from *C* by adding tail *T*.

Example:

$$C = \{000, 100, 001, 110, 011, 111\}$$

$$T = \{t_1, t_2\}, t_1 = 00, t_2 = 11$$

$$C.T = \{00011, 10000, 00100, 11011, 01111, 11100\}$$

$$C.T^2 = \{00011, 10011, 00111, 11000, 01100, 11100\}$$

$$C.T^3 = \{00011, 10011, 00111, 11011, 01111, 11100\}$$

Remark 1.10: We observe the following points here.

- 1) $|C.T^q| = |C|$, the cardinality of the respective codes, length of $C.T^q = n + s$.
- 2) If *C* is systematic in *n'* bits then $C.T^q$ is also systematic in *n'* bits. In general, *s* will be quite small compared to *n*. Thus, if there is a good encoding procedure for *C*, there will also be one for $C.T^q$.

We first describe a construction which shows that by adding just 3 parity check bits more a *t*-Asymmetric EC code can be made into a *t*-UEC code.

Theorem 1.11: Let *C* be a *t*-Asymmetric EC code of length *n* and let $T = \{t_1, t_2, t_3\}$ be a code of length 3 where $t_1 = 100, t_2 = 010, t_3 = 001$. Then the code $C.T^{(t+1)}$ is a *t*-UEC code of length $n + 3$.

The above theorem is a particular case of a more general construction described in the next theorem which shows how any code *C* satisfying the following Property 1.12 (which is essentially condition (1) of Theorem 1.1) can be modified to satisfy the same property with a larger value of ℓ . Note that a *t*-Asymmetric EC code of length *n* has the Property 1.12 with $\ell = t + 1$. Thus the above result follows from Corollary 1.6, Theorem 1.9 and the theorem below (Theorem 1.13).

Property 1.12: Let ℓ be a given positive integer $\forall \underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y} N(\underline{x}, \underline{y}) = 0 \implies N(\underline{y}, \underline{x}) \geq \ell$.

Theorem 1.13: Let *C* be a code of length *n* satisfying the Property (1.12). Let $T = \{t_1, t_2, \dots, t_{m+1}\}$ be an AUED code of length *s*. Then the code $C.T^\ell$ satisfies the Property (1.12) with ℓ replaced by $\ell + m + 1$.

It is well known that for a given *m* there exists an AUED code *T* of length *s* with $|T| = m + 1$, where *s* is the smallest integer such that $\left[\frac{s}{2} \right] \geq m + 1$. In fact, *T* is the well known Sperner Code [10]. By using Sterling Approximation the above *s* is approximately

$2^s/\sqrt{\pi s/2}$ as shown in the following derivation. Thus we can take this code as T in the above theorem. We now give two results which are essentially modifications of methods B and C of Lin and Bose [1].

We now present two results which are modifications of the methods B and C of Lin and Bose [1]. We present a few Lemmas whose results will be useful for the proof of the theorem to follow.

Theorem 1.14: Let $T_s = \{t_1, t_2, \dots, t_s\}$ be a code of length s , where $t_1 = 000 \dots 0111 \dots 1$, (there are $\lceil s/2 \rceil$ 1's and $\lfloor s/2 \rfloor$ 0's in t_1) and t_{i+1} is a 1 bit circular left shifted transform of t_i , $0 \leq i \leq s-1$. Let C be a code of length n with minimum distance $2t+2$ in which every codeword is of even weight. Then the code $C.T_s^2$ is a t -UEC code of length $n+s$ with the following property: for $\underline{x}, \underline{y} \in C.T_s^2$,

- 1) $0 < N(\underline{y}, \underline{x}) \leq t \implies N(\underline{x}, \underline{y}) \geq \text{maximum}\{t + s + 1, 2s - t\}$
- 2) $\underline{x} \neq \underline{y} \implies N(\underline{x}, \underline{y}) = 0 \pmod{2s}$ and $N(\underline{x}, \underline{y}) \geq 2t + 2$.

The following Theorem is proved in Lin and Bose [1].

Theorem 1.15: Let $s \geq t+3$, the code $C.T_s^2$ of the Theorem 1.14 is a t -EC $(2s-t-1)$ -UED Code.

Corollary 1.16: Let $T_1 = \{0, 1\}$ and $T_2 = \{01, 10\}$ be codes of length 1 and 2, respectively. Let $C.T_s^2$ be the code as defined in the statement of Theorem 1.14.

- 1) Let $t = i \pmod{s}$, $0 \leq i < \frac{s}{2}$; then $C.T_s^2$ is a t -UEC $(s+t)$ -UED code of length $n+s$.
- 2) Let $t = i \pmod{s}$, $\frac{s}{2} \leq i < s$ and let m be a unique integer defined by $2t+2 \geq m \geq 2t+s$ and $m \equiv 0 \pmod{2s}$, then (a) $(C.T_s^2).T_2^m$ is a t -UEC $(s+t)$ -UED code of length $n+s+2$. (b) $(C.T_s^2).T_1^m$ is a t -UEC $(s+t)$ -UED code if $m = 2t+s$.

We remark here that Corollary 1.16 gives much better bounds on s for t -UEC d -UED codes than similar bounds for t -EC d -UED codes in [1].

For example, for $d = t+2$, Corollary 1.16 yields the following results. $d = t+2$ and let $s = 2$.

From statement 1 of Corollary 1.16, we get the code $C.T_s^2$ which is a t -UEC $(s+t)$ -UED code of length $n+s$.

Here, the code $C.T_s^2$ will generate t -UEC $(t+2)$ -UED code of length $n+2$; i.e., addition of only 2 bits.

From statement 2 of Corollary 1.16, we get the code $(C.T_s^2).T_1^m$ such that $2t+2 \geq m \geq 2t+s$ and $m \equiv 0 \pmod{2s}$.

Here, with $s = 2$, $2t+2 \leq m \leq 2t+2 \implies m = 2t+2 = 2t+s$. Again, $m \equiv 0 \pmod{2s} = 2s = 4$. So, we have the code $(C.T_s^2).T_1^m$ which becomes a $(C.T_s^2).T_1^4$ with length $n+s+1$ i.e., addition of only 3 bits.

In comparison, for a t -EC $(t+2)$ -UED code (see Corollary 3.4 in [1]) one needs $s = \lceil \log_2(t+2) \rceil$ bits. So we need larger number of additional check bits as t increases. For a t -UEC $(t+2)$ -UED code, the number of additional check bits does not vary with t .

For the next result we will consider the code C and the tail T with the following properties.

Property 1.17: There exist positive integers q and t , such that, $\forall a, b \in C, a \neq b, N(\underline{b}, a) \leq t \implies w(a) - w(b) \geq q$.

Property 1.18: $T = \{t_1, t_2, \dots, t_m\}$ is such that for all $1 \leq i, j \leq m, i \neq j$

- 1) $N(t_i, t_j) \leq t \implies j = i + 1 \pmod{m}$
- 2) $N(t_i, t_j) > 0$ for $j = i + 1 \pmod{m}$.

Remark 1.19: We observe the following points here.

- 1) Property 1.17 implies that C is a t -Asymmetric EC code and any t -Asymmetric EC code satisfies Property 1.17 for at least $q = 1$.

TABLE I

s	d	t	:	s	d	t	:	s	d	t
2	4	2	:	3	6	3	:	5	10	5
2	6	4	:	3	9	6	:	5	11	6
2	8	6	:				:	6	12	6
			:	4	8	4	:	14	82	2*
			:	4	9	5	:	18	113	3*

2) If T is a t -EC AUED code then $N(t_i, t_j) \geq t+1$ for all $t_i, t_j \in T$. Thus Property 1.18 is vacuously satisfied for such a code T .

3) A code T satisfying Property 1.18 is clearly an AUED code.

Theorem 1.20: Let C and T be codes of length n and s respectively satisfying Properties (1.17) and (1.18).

Then,

- 1) the code $C.T^q$ is a t -UEC $((m-1)q-t)$ -UED code of length $n+s$, if $q \leq t$.
- 2) if T is also a t -EC AUED code of length s , then $C.T^q$ is a t -EC $((m-1)q-t)$ -UED code of length $n+s$.

We note that $q = 1, 2$ case of the above theorem is essentially the method C of Lin and Bose [1]. In fact, it also shows that in Method C of Lin and Bose [1], we could start with a t -Asymmetric EC code in place of t -EC $(t+1)$ -ED code.

Corollary 1.21: Let C be a code of length n with the maximum distance $2t+2$, in which every code word is of even weight. Let T be t -EC AUED code of length $s, |T| = m$ and let $T_2 = \{01, 10\}$ be code of length 2. Then the code $(C.T_s^2).T^4$ is a t -EC $(4(m-1)t)$ -UED code of length $n+s+2$.

Finally we give in Table I some values of triplets $(s, d, t), t \leq 6$, for which there exists a code T of length s such that for any code C with minimum distance $2t+2$, in which every code word is of even weight, the code $C.T$ is a t -UEC d -UED code. We only give those triples where the values of d is better than that given in Table I of [1]. All the values are obtained by using Corollary 1.16 (1) or Corollary 1.21. The rows marked with an asterisk (*) are those for which Corollary 1.21 is used. The codes T for these are taken to be Hadamard Codes [8]. Note that the rows marked with an asterisk (*) give actually t -EC d -UED codes. Thus, they improve the bounds in [1] for such codes where corresponding triples are (14, 81, 2) and (18, 92, 3), respectively.

Remark 1.22: We have not used t -Asymmetric EC codes in above construction. The bounds for s could be improved further by using t -Asymmetric EC codes. In general, it should also give better bounds for 1-EC d -UED codes. We also note that other cases of Theorem 1.14 and Theorem 1.20 should be more useful for large t and large d . It has been shown by Lin and Bose [1] that SEC d -UED codes, constructed by method of Theorem 1.20 with $q = 1$ are close to optimal. We also note here that Theorem 1.11 shows the close relationship between t -Asymmetric EC and t -UEC codes. A similar relationship between these codes and t -EC codes can also be developed. A more detailed paper on relationships between these codes is under preparation.

III. PROOFS AND DECODING ALGORITHM

Proof of Theorem 1.1: Let C be a code of length n satisfying conditions 1 and 2 of the theorem. Suppose that a codeword \underline{x} has suffered t or fewer unidirectional errors and the received word is \underline{x}' . Let $\underline{y} \in C, \underline{y} \neq \underline{x}$. Suppose $N(\underline{x}, \underline{y}) = 0$ and $N(\underline{y}, \underline{x}') \leq d$.

Case 1: Suppose \underline{x} has suffered $t(1 \rightarrow 0)$ unidirectional errors and has become \underline{x} , so $\underline{x} \subseteq \underline{x}'$. Now, $N(\underline{x}, \underline{x}') \leq t$ and $N(\underline{x}', \underline{x}) = 0$. It can easily be checked that $N(\underline{y}, \underline{x}') \leq N(\underline{y}, \underline{x}) \leq d$ and $N(\underline{x}, \underline{y}) \leq N(\underline{x}, \underline{x}') \leq t$. Thus, condition 2 of Theorem 1.1 is not satisfied.

Case 2: Suppose \underline{x} has suffered $t (0 \rightarrow 1)$ unidirectional errors and has become \underline{x}' , so $\underline{x} \subseteq \underline{x}'$. Now, $N(\underline{x}, \underline{x}') = 0$ and $N(\underline{x}', \underline{x}) \leq t$ and since $N(\underline{x}, \underline{y}) = 0$ (\underline{x} has suffered $t (0 \rightarrow 1)$ errors to become \underline{x}' , therefore $N(\underline{x}, \underline{y})$ atmost will be 0), $N(\underline{x}, \underline{y}) = 0$. Also, since $N(\underline{y}, \underline{x}') \leq d$, then $N(\underline{y}, \underline{x}) \leq d + t$. Thus, condition 1 of Theorem 1.1 is not satisfied.

Similarly, for $N(\underline{y}, \underline{x}') = 0$ and $N(\underline{x}', \underline{y}) \leq d$, one can show that \underline{x} and \underline{y} will not satisfy condition 1 or 2 of Theorem 1.1.

Thus, we have shown that \underline{x}' cannot be obtained from any $\underline{y} \in C, \underline{y} \neq \underline{x}$ by d or less unidirectional errors. This proves that C is a t -UEC d -UED code and also that the given conditions are sufficient.

To prove that conditions 1 and 2 are necessary, suppose that $\underline{x}, \underline{y} \in C$. Now if $N(\underline{x}, \underline{y}) = 0$ and $N(\underline{y}, \underline{x}) \leq d + t$, we can easily find a vector \underline{z} so that $\underline{x} \subseteq \underline{z} \subseteq \underline{y}$ and $N(\underline{z}, \underline{x}) \leq t$ and $N(\underline{y}, \underline{z}) \leq d$. Thus \underline{z} can be obtained by making atmost t -unidirectional errors in \underline{x} and atmost d -unidirectional errors in \underline{y} . Thus, C is not a t -UEC d -UED code if condition 1 is not necessarily satisfied.

In case if $0 < N(\underline{x}, \underline{y}) \leq t$ and $N(\underline{y}, \underline{x}) \leq d$, we can again find a word \underline{z} that can be obtained by making atmost t -unidirectional errors in \underline{x} and atmost d -unidirectional errors in \underline{y} . Thus, C is not a t -UEC d -UED code if condition 2 is not necessarily satisfied. This completes the proof. \square

Example: In this example, we elaborate above proof and show how we can identify codewords which show that the code is not a t -UEC d -UED code, if the conditions of Theorem 1.1 are not satisfied.

- 1) If $N(\underline{x}, \underline{y}) = 0$ and $N(\underline{y}, \underline{x}) \leq d + t$, we can easily find a word \underline{z} so that $\underline{x} \subseteq \underline{z} \subseteq \underline{y}$. $N(\underline{z}, \underline{x}) \leq t$ and $N(\underline{y}, \underline{z}) \leq d$.

Without loss of generality, let $\underline{x}, \underline{y}$ be as follows. We can obtain a word \underline{z} by at most $t (0 \rightarrow 1)$ errors from \underline{x} and at most $d (1 \rightarrow 0)$ from \underline{y} :

$$\begin{aligned} \underline{x} &= 111100000000000000 \\ \underline{z} &= 111100000001110000 \\ \underline{y} &= 111100001111110000. \end{aligned}$$

Hence, it is seen that $N(\underline{x}, \underline{y}) = 0, N(\underline{y}, \underline{x}) \leq d + t, N(\underline{y}, \underline{z}) \leq d, N(\underline{z}, \underline{x}) \leq t$ and \underline{z} is obtainable from both \underline{x} and \underline{y} as stated above.

- 2) If $0 < N(\underline{x}, \underline{y}) \leq t$ and $N(\underline{y}, \underline{x}) \leq d$, we can obtain a word \underline{z} by $t (1 \rightarrow 0)$ from \underline{x} and $d (1 \rightarrow 0)$ from \underline{y} .

Without loss of generality, let $\underline{x}, \underline{y}$ be as follows. The word \underline{z} is obtainable both from \underline{x} and \underline{y} as stated above:

$$\begin{aligned} \underline{x} &= 11111110000010000 \\ \underline{z} &= 111100000000000000 \\ \underline{y} &= 111100001111110000. \end{aligned}$$

Proof of Theorem 1.13: Let C and T be as in the statement of the theorem. Let $\underline{x}, \underline{y}' \in C.T^\ell, \underline{x}' \neq \underline{y}'$ and $N(\underline{x}, \underline{y}') = 0$. Suppose that $\underline{x}' = \underline{x}t_i$ and $\underline{y}' = \underline{y}t_j, \underline{x}, \underline{y} \in C; t_i, t_j \in T$. Then clearly $N(\underline{x}, \underline{y}) = 0$ and $N(t_i, t_j) = 0$. Since T is an AUED code (from Theorem 5 of Bose and Rao [4]) we deduce that in the above code T , every pair of codewords is unordered; i.e., $\forall i, j, i \neq j, N(t_i, t_j) \geq 1$ implying that $t_i \not\subseteq t_j$ and $t_j \not\subseteq t_i$ and C satisfies Property 1.12, the above implies that $N(\underline{y}, \underline{x}) \geq \ell$. As we have already seen, $t_i \not\subseteq t_j$ and $t_j \not\subseteq t_i$, condition $N(t_i, t_j) = 0$ is only possible if $t_i = t_j$. Thus, $w(\underline{y}) - w(\underline{x}) \geq \ell > 0$ and their tails are equal. From (1), for $C.T^\ell$, it now follows that $w(\underline{y}) - w(\underline{x}) \geq m\ell + 1$. This implies that $w(\underline{x}') - w(\underline{y}') \geq m\ell + 1$. This completes the proof. \square

Before proceeding with proof of Theorem 1.14, we state the following two simple results whose proofs are easy [1].

Lemma 2.1: Let C be a code of length n with minimum distance $2t + 2$, such that all codewords of C are of even weight. Let $\underline{x}, \underline{y} \in C, \underline{x} \neq \underline{y}$. and let $q = w(\underline{x}) - w(\underline{y}) \geq 0$. Then,

- 1) q is even and $N(\underline{x}, \underline{y}) = q + N(\underline{y}, \underline{x})$;
- 2) $N(\underline{x}, \underline{y}) \geq \max\{t + 1 + \frac{1}{2}q, q\}$ $N(\underline{y}, \underline{x}) \geq t + 1 - \frac{1}{2}q$.

Lemma 2.2: Let T_s be the code of length s as defined in the statement of Theorem 1.14. Then for $1 \leq j, i \leq s, N(t_i, t_j) = \text{Minimum}\{|j - i|, s - |j - i|\}$

We will need one more Lemma for the proof of Theorem 1.14.

Lemma 2.3: Let C and T_s be as in the statement of Theorem 1.14.

Let $\underline{x}, \underline{y} \in C.T_s^2, \underline{x} \neq \underline{y}$.

$\underline{x} = \underline{a}t_i, \underline{y} = \underline{b}t_j, \underline{a}, \underline{b} \in C; t_i, t_j \in T_s$. Let $w(\underline{a}) - w(\underline{b}) = q \geq 0$. Then,

- 1) if $q \leq s, N(\underline{y}, \underline{x}) \geq t + 1$;
- 2) if $q > s, N(\underline{x}, \underline{y}) \geq t + 1 + s$;
- 3) $0 < N(\underline{y}, \underline{x}) \leq t \implies N(\underline{x}, \underline{y}) \geq 2s - t$.

Proof of Lemma 2.3: We first note that $\underline{a} \neq \underline{b}$ since otherwise $\underline{x} = \underline{y}$. Also using (1) for $C.T_s^2$, we have, $|j - i| = \frac{1}{2}q \pmod{s}$. Hence, using Lemma 2.2 we get, for $q \leq 2s$,

$$\begin{aligned} N(t_i, t_j) &= N(t_j, t_i) = \text{Minimum}\{\frac{q}{2}, s - \frac{q}{2}\} \\ &= \begin{cases} \frac{1}{2}q & \text{if } q \leq s \\ s - \frac{q}{2} & \text{if } s < q \leq 2s. \end{cases} \end{aligned}$$

Using this and Lemma 2.1, we have the following.

- a) For $q \leq s, N(\underline{y}, \underline{x}) = N(\underline{b}, \underline{a}) + N(t_j, t_i) \geq (t + 1 - \frac{q}{2}) + \frac{q}{2} = t + 1$.
- b) For $2s \geq q > s, N(\underline{x}, \underline{y}) = N(\underline{a}, \underline{b}) + N(t_i, t_j) \geq (t + 1 + \frac{q}{2}) + (s - \frac{q}{2}) = t + 1 + s$.
- c) For $q \geq 2s, N(\underline{x}, \underline{y}) \geq N(\underline{a}, \underline{b}) \geq t + 1 + \frac{q}{2} \geq t + 1 + s$.
- d) Since $N(\underline{a}, \underline{b}) - N(\underline{b}, \underline{a}) = w(\underline{a}) - w(\underline{b}) = q$, we have $N(\underline{a}, \underline{b}) \geq q$.

Hence, if $2s \geq q > s$, we have:

for $\frac{q}{2} \geq s - t; N(\underline{x}, \underline{y}) = N(\underline{a}, \underline{b}) + N(t_i, t_j) \geq q + (s - \frac{q}{2}) = \frac{q}{2} + s \geq (s - t) + s \geq 2s - t$ and for $\frac{q}{2} \leq \frac{q}{2} \leq s - t - 1; N(\underline{y}, \underline{x}) = N(\underline{b}, \underline{a}) + N(t_j, t_i) \geq N(t_j, t_i) = s - \frac{q}{2} \geq t + 1$.

a), b), c), d) together imply the statements 1), 2), and 3) of the Lemma. \square

Proof of Theorem 1.14: The first statement follows from Lemma 2.3. We now prove statement 2) of the theorem.

Let $\underline{x}, \underline{y} \in C.T_s^2$; also let $\underline{a}, \underline{b} \in C$ and $t_i, t_j \in T_s$ be as in the statement of Lemma 2.3. Now, $\underline{x} \neq \underline{y}, N(\underline{y}, \underline{x}) = 0 \implies N(\underline{b}, \underline{a}) = 0, N(t_j, t_i) = 0$ and $\underline{a} \neq \underline{b}$.

Using above lemmas, if $N(t_j, t_i) = 0$, it is obvious that $t_i = t_j$ and $w(\underline{a}) \neq w(\underline{b})$ (since otherwise $N(\underline{b}, \underline{a}) \geq t + 1$). Thus, $w(\underline{a}) \neq w(\underline{b})$ while $t_i = \text{tail of } \underline{a} = t_j = \text{tail of } \underline{b}$. From construction of the code $C.T_s^2$ it now follows that $q = w(\underline{a}) - w(\underline{b}) \equiv 0 \pmod{2s}$. Since $N(\underline{b}, \underline{a}) = 0$ and $t_i = t_j$, it implies that $N(\underline{x}, \underline{y}) = N(\underline{a}, \underline{b}) \equiv 0 \pmod{2s}$. Also using Lemma 2.1, if $N(\underline{b}, \underline{a}) = 0, q \geq 2(t + 1)$ and hence if $N(\underline{y}, \underline{x}) = 0, N(\underline{x}, \underline{y}) \geq N(\underline{a}, \underline{b}) \geq 2(t + 1)$. This completes the proof. \square

Proof of Corollary 1.16: Let $C.T_s^2$ be the code as defined in the statement of the Theorem 1.14. Now the Theorem clearly implies that condition 2) of Theorem 1.1 is satisfied for $C.T_s^2$ with $d = t + s$ and also for $\underline{x}, \underline{y} \in C.T_s^2$ if $N(\underline{y}, \underline{x}) = 0$ then $N(\underline{x}, \underline{y}) \geq 2t + 2$ and $N(\underline{x}, \underline{y}) \equiv 0 \pmod{2s}$. Thus if $N(\underline{y}, \underline{x}) = 0$ and $N(\underline{x}, \underline{y}) \leq d + t + 1, d = t + s$ then $N(\underline{x}, \underline{y}) = m$, where m is as defined in the statement of the Corollary. Note that such an m exists only if $t \equiv i \pmod{s}$, for $\frac{s}{2} \leq i < s$. Thus, if there is no such m then clearly $C.T_s^2$ also satisfies condition 1) of Theorem 1.1. The statement 1) of the Corollary now follows from Theorem 1.1. Also clearly codes $(C.T_s^2), T_2^m$ or $(C.T_s^2), T_1^m$ satisfy condition (2) of Theorem 1.1. Now suppose $t \equiv i \pmod{s}, \frac{s}{2} \leq i < s$ and $\underline{x}, \underline{y} \in C.T_s^2$ with $N(\underline{y}, \underline{x}) = 0$ and $N(\underline{x}, \underline{y}) \leq 2t + s$ then $N(\underline{x}, \underline{y}) = m$ as above and hence if $\underline{x}' = \underline{x}t_i$ and $\underline{y}' = \underline{y}t_j$

are corresponding elements of code (C, T_s^2) , T_2^m (or (C, T_s^2) , T_1^m when $m = 2t + s$) then clearly using (1) for these codes it can be easily seen that $t_i \neq t_j$. It now follows easily that $N(\underline{y}, \underline{x}) \neq 0$, unless $m = 2t + s$ and $\underline{x}', \underline{y}' \in (C, T_s^2)$, T_1^m and in this case $N(\underline{x}', \underline{y}') = m + 1 = 2t + s + 1$. Thus (C, T_s^2) , T_2^m and (C, T_s^2) , T_1^m both satisfy condition 1) of Theorem 1.1, also. This completes the proof of statement 2) of the Corollary, by using Theorem 1.1. \square

Proof of Theorem 1.20: Let C and T be codes as in the statement of the Theorem. Let $\underline{x}, \underline{y} \in C, T^q$, $\underline{x} = a t_i, \underline{y} = b t_j, a, b \in C$ and $t_i, t_j \in T$ and let $\ell = w(a) - w(b) \geq 0$. Thus using Property 1.17 for the code C , Property 1.18 for the code T and (1) for the code C, T^q , the following facts can be easily seen:

- 1) $N(\underline{x}, \underline{y}) \geq N(a, b) \geq t + 1$ and if $\ell < q$ then $N(\underline{y}, \underline{x}) \geq N(b, a) \geq t + 1$
- 2) if $(m - 2)q \geq \ell \geq q$ then $i \neq j \pmod{m}$ and $j \neq i + 1 \pmod{m}$ and hence $N(\underline{y}, \underline{x}) \geq N(t_j, t_i) \geq t + 1$.
- 3) if $(m - 1)q \geq \ell \geq (m - 2)q + 1$ then $j \neq i \pmod{m}$ and hence $N(\underline{y}, \underline{x}) \geq N(t_{i+1}, t_i) > 0 \pmod{m}$
- 4) If T is a t -EC AUED code then for $(m - 1)q \geq \ell \geq (m - 2)q + 1$, $N(\underline{y}, \underline{x}) \geq N(t_j, t_i) \geq t + 1$, since $j \neq i \pmod{m}$.

From the above facts it follows that $\underline{x}, \underline{y} \in C, T^q$ have the following properties.

- g) if $N(\underline{y}, \underline{x}) \leq t$ then $N(\underline{x}, \underline{y}) \geq w(a) - w(b) \geq (m - 2)q + 1$.
- h) if $N(\underline{y}, \underline{x}) = 0$ then $N(\underline{x}, \underline{y}) \geq w(a) - w(b) \geq (m - 1)q + 1$.
- i) if T is t -EC AUED code then $N(\underline{y}, \underline{x}) \leq t \implies N(\underline{x}, \underline{y}) \geq w(a) - w(b) \geq qm$.

From (g), (h), (i) it follows that the code C, T^q satisfies the conditions 1) and 2) of the Theorem 1.1 with $d = (m - 1)q - t$ if $t \geq q$, also if T is a t -EC AUED code then C, T^q satisfies Property 1.3 with $d = (m - 1)q - t$. Theorem 1.1 and Theorem 1.5 now imply that for $t \geq q$, C, T^q is t -UEC d -UED code and if T is t -EC AUED code, then C, T^q is t -EC d -UED code.

Proof of Corollary 1.21: Let C, T, T_2 be as in the statement of the Corollary. Now it can be easily seen (for example by using Lemma 2.1 for C), C, T_2^2 satisfies Property 1.17 with $q = 4$. Hence, the corollary follows from Theorem 1.20, condition 2).

We now discuss decoding algorithms for the codes constructed by Theorem 1.11, 1.14, and 1.20 and their Corollaries. Since the algorithms are essentially the same for all these codes and essentially similar to the decoding algorithms given in Lin and Bose [1, Section IV] for t -EC d -UED codes, we just give a few details and sketch the proof of validity, for the case of code C, T^q , constructed by Theorem 1.20 considered as t -UEC d -UED code. Other details are similar to those in [1]. We assume that a good algorithm to decode C as t -Asymmetric EC code (see Remark 1.19) is known, which decodes given any received word \underline{x}' to give a word $\underline{x}'' \in C$ such that either $N(\underline{x}', \underline{x}'') \leq t$ and $N(\underline{x}'', \underline{x}') = 0$ or $N(\underline{x}', \underline{x}') \leq t$ and $N(\underline{x}', \underline{x}') = 0$. \square

IV. DECODING ALGORITHM

Let $\underline{x}^* = \underline{x}t$ be the transmitted codeword in C, T^q , $\underline{x} \in C$, $t \in T$. Let $(\underline{x}^*)' = \underline{x}'t'$ be the received word, length of $\underline{x}' = n$, length of $t' = s$.

- 1) Decode \underline{x}' using algorithm for C to get \underline{x}'' . Thus $\underline{x}'' \in C$.
- 2) Compute tail of $\underline{x}'' = t''$, so that $\underline{x}''t'' \in C, T^q$.
- 3) If $N(\underline{x}''t'', \underline{x}'t') \leq t$ and $N(\underline{x}'t', \underline{x}''t'') = 0$ or $N(\underline{x}'t', \underline{x}''t'') \leq t$ and $N(\underline{x}''t'', \underline{x}'t') = 0$ then output $\underline{x}''t''$. Otherwise declare "errors detected" and stop.

V. SKETCH OF PROOF OF VALIDITY OF ALGORITHM

Case a: t or fewer unidirectional errors have occurred; then $\underline{x}'' = \underline{x}'$ and hence $t'' = t'$. Thus, algorithm decodes the codeword $\underline{x}t$ correctly.

Case b: More than t but no more than d unidirectional errors have occurred.

- 1) If t or less unidirectional errors have occurred in \underline{x} , then clearly $\underline{x}'' = \underline{x}'$ and $t'' = t'$. Thus either $d \geq N(\underline{x}''t'', \underline{x}'t') \geq t + 1$ or $d \geq N(\underline{x}'t', \underline{x}''t'') \geq t + 1$ and the algorithm will, in step 3), correctly signal "errors detected."
- 2) If more than t -unidirectional errors have occurred in \underline{x} , i.e. either $N(\underline{x}, \underline{x}') = 0$ and $d \geq N(\underline{x}', \underline{x}) \geq t + 1$ or $N(\underline{x}', \underline{x}) = 0$ and $d \geq N(\underline{x}, \underline{x}') \geq t + 1$. Also, clearly either $N(\underline{x}'', \underline{x}') = 0$ and $N(\underline{x}', \underline{x}'') \leq t$ or $N(\underline{x}', \underline{x}'') = 0$ and $N(\underline{x}'', \underline{x}') \leq t$. From these conditions, it can be easily seen that $1 \leq |w(\underline{x}) - w(\underline{x}'')| \leq d + t$ and hence, from construction of C, T^q , it follows that $t \neq t''$. Also, since $\underline{x}'t'$ is obtained from $\underline{x}t$ by unidirectional errors $t' = t$, thus $t' \neq t''$ and hence $\underline{x}''t''$ cannot be obtained from $\underline{x}'t'$ by unidirectional changes and the algorithm will, in step 3) correctly signal "errors detected."

VI. CONCLUDING REMARKS

We have presented the basic theory and methods of construction of t -UEC d -UED codes in this chapter. Encoding and decoding algorithms are also discussed. It has been shown that by assuming t -Asymmetric EC code in place of t -UEC code, we can save at most 3 check bits. We have presented t -UEC d -UED and t -EC d -UED codes in tabular form. Some improvement in the bounds of unidirectional error detection capability over Lin and Bose [1] bounds is shown. We have not used t -Asymmetric EC Codes in the construction of the codes which could have improved the bounds further. We also observe that in [1] one needs $s = \lceil \log_2(t + 2) \rceil$ bits for t -EC $(t + 2)$ -UED codes. Hence, the additional check bits increase with the increase of t . In t -UEC $(t + 2)$ -UED code, we have observed that the size of additional check bits is constant i.e., it is not a function of t .

REFERENCES

- [1] D. J. Lin and B. Bose, "Theory and design of t -error correcting and $d(d \geq t)$ -detecting (t -EC d -UED) codes," *IEEE Trans. Comput.*, vol. 37, no. 4, pp. 433-439, Apr. 1988.
- [2] B. Bose and D. K. Pradhan, "Optimal unidirectional error detecting/correcting codes," *IEEE Trans. Comput.*, vol. C-31, no. 6, pp. 564-568, June 1982.
- [3] B. Bose, T. R. N. Rao, "Unidirectional error correcting codes for shift register memories," *IEEE Trans. Comput.*, vol. C-33, no. 6, pp. 575-578, June 1984.
- [4] —, "Theory of unidirectional error correcting/detecting codes," *IEEE Trans. Comput.*, vol. C-31, no. 6, pp. 521-530, June 1982.
- [5] D. Nikolos, N. Gaitanis, and G. Philokyprou, " t -error correcting and all unidirectional error detecting codes, starting from cyclic AN codes," presented at the 14th Int. Symp. Fault-Tolerant Computing, June 1984.
- [6] —, "Systematic t -Error correcting/all unidirectional error detecting codes," *IEEE Trans. Comput.*, vol. C-35, no. 5, pp. 394-402, May 1986.
- [7] B. Bose, "Unidirectional error correction/detection for VLSI memory," in *Dig. Papers, 11th Symp. Comput. Architecture*, June 1984, pp. 242-244.
- [8] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam: North Holland, 1978.
- [9] M. Blaum and H. van Tilborg, "On t -error correcting/all unidirectional error-detecting codes," Res. Rep., IBM RJ 5566(56685), Mar. 1987.
- [10] T. Verhoeff, "Delay-insensitive codes—An overview," *Distributed Computing*, vol. 3, no. 1, pp. 1-8 (Springer-Verlag), 1988.
- [11] D. K. Pradhan, "A new class of error correcting/detecting codes for fault-tolerant computer applications," *IEEE Trans. Comput.*, vol. C-29, no. 6, pp. 471-481, June 1980.

- [12] B. Bose and J. Metzner, "Coding theory for fault tolerant systems," in *Fault-Tolerant Computing: Theory and Techniques*, D. K. Pradhan, Ed. Englewood Cliffs, NJ: 1986, vol. 1.

High-Speed Microprogrammable Asynchronous Controller Modules

R. F. Tinder, R. I. Klaus, and J. A. Snodderley

Abstract—A unique family of high-speed, microprogrammable asynchronous controller (MAC) modules is described in this correspondence. Each MAC module consists of two fundamental mode machines that communicate by means of a handshake interface that permits it to be driven by any programmable logic device including ROM's. Any state machine controller designed with a MAC module will operate free of critical races, essential hazards and output race glitches, and will have static hazard-free state variables.

A multiplicity of programmable logic devices can be used to drive one or more MAC modules to achieve complex, but reliable, asynchronous, time-shared and/or parallel processing of data. Individual MAC modules having state variables numbering l, m, n, \dots can be cascaded to produce an available system state capacity of $2^l \times 2^m \times 2^n \times \dots$ states with up to $(l + m + n + \dots)$ -way transition capability, all without compromising speed or reliability.

Index Terms—Asynchronous controllers, asynchronous modules, asynchronous sequencers, asynchronous state machines, high-speed controllers, modules, programmable controllers, sequencers.

I. INTRODUCTION

This correspondence describes the details of a family of high-speed, cascable, microprogrammable asynchronous controller (MAC) modules that are designed to operate in the fundamental mode. These modules can be used to design asynchronous state machines of virtually any size, complexity and input and output capability, and that cannot malfunction by any of the timing defects common to fundamental mode state machines. An important feature of the MAC module is that it can be driven by any programmable logic device (PLD) to produce a state machine (controller) that is guaranteed to operate correctly, free of critical races, essential hazards, output race glitches, and static hazards in the state variables. In effect, use of a MAC module for controller design eliminates the need to analyze and correct each design for timing defects that could cause controller failure. This feature permits a single MAC module to be rapidly operated on a time shared basis between any number of radically different asynchronous controller designs.

Methods for static hazard-free designs of asynchronous state machine are not new, and algorithms for selective hazard detection and elimination and for the synthesis of asynchronous state machines have even been developed [1], [2]. In fact, sources of information on the analysis and selective elimination of critical races, output race glitches, and static and essential hazards in asynchronous machines are numerous. Typical examples, past and present, that provide

Manuscript received August 8, 1990; revised March 8, 1993. This work is covered by United States Patent No. 5063 536, November 5, 1991.

R. F. Tinder is with the Department of Electrical and Computer Engineering, Washington State University, Pullman, WA 99164-2752 USA; e-mail: rtinder@eecs.wsu.edu.

R. I. Klaus is with Hewlett Packard, Research and Development Laboratory, Vancouver Div., Vancouver, WA 98668-C006 USA.

J. A. Snodderley is with Hewlett Packard Procurement Department, Spokane Div., Spokane, WA 99220-2500 USA.

IEEE Log Number 9402556.

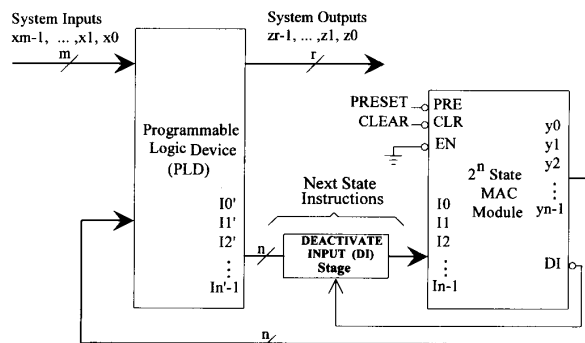


Fig. 1. Functional partition showing a generalized MAC module, the programmable logic device (PLD) used to drive it, and the DEACTIVATE INPUT stage.

extensive coverage in this area are the works of Unger [3] and Tinder [4]. What is new is the design of a family of cascable modules each of which can be used to switch rapidly from one fundamental mode state machine to another without the need to analyze each machine for timing defects such as those just mentioned. No previous work in this specific area is known to exist. Traditionally, the emphasis has been on the design of individual asynchronous FSM's, each of which must be carefully analyzed so as to avoid the types of timing defects mentioned previously. The general approach to design and analysis of fundamental mode state machines is detailed by Tinder [4]. Typical of other recent sources of information on asynchronous state machines are the texts by Dietmeyer [5] and McCluskey [6].

The PLD used to drive a MAC module can be a ROM, PLA, PAL, or programmable gate array device. Interchanging PLD's or reprogramming an existing PLD permits a single PLD/MAC module system to be easily converted from one state machine design to another radically different one. Alternatively, an array of PLD's could be connected to a single MAC module via multiplexers, thereby permitting immediate switching between controllers. Equally interesting is the prospect of replacing the array of PLD's with static RAM so as to produce user-actuated asynchronous controllers during operation. Furthermore, systems of cascaded MAC modules make possible multiple time-shared and/or parallel asynchronous processors, all guaranteed to operate free of critical races, output race glitches, essential hazards, and to have static hazard-free state variables. Static hazards that are generated in the output functions can be eliminated by well known means as discussed later.

II. MAC MODULE ARCHITECTURE AND OPERATION

Shown in Fig. 1 are the block symbols for a generalized MAC module of n state variables and 2^n states, and any PLD that is programmed to provide the next state instructions to the MAC module and form the output logic. As indicated, the MAC module has n next state instruction inputs from the PLD which, in turn, receives n present state variables from the MAC module. Also shown is a DEACTIVATE INPUT stage that may be considered as part of either the PLD or MAC module, or may be used as a stand-alone stage.

The MAC module consists of two fundamental mode machines, a state array machine (SAM) and a timing control machine (TCM), which interface by a unique handshake configuration. The block diagram symbols and their interconnections are presented in Fig. 2. Five outputs of the SAM (select parameters S_e and S_o , parity