Explicit Constructions of Memoryless Crosstalk Avoidance Codes via C-transform

Cheng-Shang Chang, Jay Cheng, Tien-Ke Huang and Duan-Shin Lee
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 30013, Taiwan, R.O.C.
Email: cschang@ee.nthu.edu.tw; jcheng@ee.nthu.edu.tw;
d915601@oz.nthu.edu.tw; lds@cs.nthu.edu.tw

Abstract— **One of the main problems in deep sub-micron designs of high speed buses is the propagation delay due to the crosstalk effect. To alleviate the crosstalk effect, there are several types of crosstalk avoidance codes proposed in the literature. In this technical brief, we develop explicit constructions of two types of memoryless crosstalk avoidance codes: forbidden overlap codes (FOCs) and forbidden transition codes (FTCs). Our constructions for both FOCs and FTCs have the largest set of codewords. To the best of our knowledge, this is the first explicit construction of a FOC that has the largest set of codewords. Our approach is based on the $\mathcal{C}$-transform developed for routing optical packets in optical queues. We show such an approach can also be used for constructing limited-weight No Adjacent Transition (NAT) codes.**

keywords: crosstalk, bus encoding, Zeckendorf's theorem

I. Introduction

High speed buses that provide information exchange among various electronic devices are crucial to the success of technology development in many electronic switches and routers. As pointed out in [1], one of the main problems in deep sub-micron designs of high speed buses is the propagation delay due to the crosstalk effect from the coupling capacitance between adjacent wires in the buses. Specially, for a bus of M parallel wires, it was shown in [1] that the delay of the i^{th} wire, denoted by T_i, can be modelled by the following equation:

$$T_i = \begin{cases} \tau_0[(1+\lambda)\Delta_1^2 - \lambda\Delta_1\Delta_2], & \text{if } i = 1, \\ \tau_0[(1+2\lambda)\Delta_i^2 - \lambda\Delta_i(\Delta_{i-1}+\Delta_{i+1})], & \text{if } i \neq 1, M, \\ \tau_0[(1+\lambda)\Delta_n^2 - \lambda\Delta_n\Delta_{n-1}], & \text{if } i = M, \end{cases} \tag{1}$$

where λ is the ratio of the coupling capacitance between adjacent wires and the loading capacitance between the i^{th} wire and the ground, τ_0 is the delay of a transition on a single wire, and

$$\Delta_i = \begin{cases} 1, & \text{a transition from 0 to 1 on the } i^{th} \text{ wire}, \\ -1, & \text{a transition from 1 to 0 on the } i^{th} \text{ wire}, \\ 0, & \text{no transition on the } i^{th} \text{ wire}. \end{cases} \tag{2}$$

In view of (2), it is clear that we can eliminate the normalized delay $(1+4\lambda)$ if we can construct a set of codewords that avoid the following two types of transitions for any three adjacent bits: $101 \to 010$ and $010 \to 101$. Codewords that have this property are known as the forbidden overlap codes (FOCs) [2] and the maximum delay of the FOCs in each wire is bounded by $\tau_0(1+3\lambda)$. If, furthermore, for any two adjacent wires there are no transitions of the following two types: $10 \to 01$ and $01 \to 10$, then one can eliminate all the normalized delay larger than $1+2\lambda$. Codewords that have this property are known as the forbidden transition codes (FTCs) [3], [4], [5], [2]. The maximum delay of the FTCs in each wire is then bounded by $\tau_0(1+2\lambda)$. Another way to achieve the same delay bound is to avoid the following two patterns in the codewords: 010 and 101. Such codewords are called the forbidden pattern codes (FPCs) [6], [2]. By further eliminating all the transitions that have delay larger than $\tau_0(1+2\lambda)$, one can then deduce the so-called one-lambda codes (OLCs) in the literature [2]. All these codes, including FOCs, FTCs, FPCs, and OLCs, are known as crosstalk avoidance codes.

The largest number of codewords of a set of *memoryless* FTCs is known to be related to the Fibonacci number [3], [4]. By using the Fibonacci numeral system, Duan, Zhu, and Khatri [5] developed an *explicit* construction of a set of *memoryless* FTCs that has the largest number of codewords. Their approach was further extended by Wu and Yan [2] to general binary numeral systems for the constructions of *memoryless* FPCs and OLCs. Despite the success of using numeral systems for FTCs, FPCs, and OLCs, Wu and Yan [2] showed that memoryless FOCs cannot be efficiently constructed by their numeral systems. Instead, they are only able to construct a subset of the largest set of FOCs.

Our main construction of this technical brief is to show there exists an explicit construction of a set of memoryless FOCs that has the largest set of codewords. To the best of our knowledge, this is the first explicit construction of a *memoryless* FOC that has the largest set of codewords. Our approach is also applicable to the construction of FPCs and limited-weight No Adjacent Transition (NAT) codes [7]. The main idea is to use a "greedy" numeral system, called the $\mathcal{C}$-transform developed in our early papers [8], [9], to generate binary representations for integers that do not have consecutive 1's. Then we invert every even-numbered bit to construct FOCs and FPCs.

II. $\mathcal{C}$-TRANSFORM

In this section, we first briefly review the $\mathcal{C}$-transform and its associated properties that is developed by routing optical packets in optical queues in [8], [9].

Definition 1: Consider an M-vector $\mathcal{U}_M = (u_1, u_2, \ldots, u_{M-1}, u_M)$ with $u_i \in \mathbf{N}$, $i = 1, 2, \ldots, M$. Define a mapping $\mathcal{C} : x \in \{0\} \cup \mathbf{N} \mapsto 2^M$ as follows:

$$\mathcal{C}(x) = \Big(d_1(x), d_2(x), \ldots, d_{M-1}(x), d_M(x)\Big), \tag{3}$$

where

$$d_M(x) = \begin{cases} 1 & \text{if } x \geq u_M \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

and for $i = M-1, \ldots, 2, 1$, $d_i(x)$ is given recursively by

$$d_i(x) = \begin{cases} 1 & \text{if } x - \sum_{k=i+1}^{M} d_k(x) \cdot u_k \geq u_i \\ 0 & \text{otherwise} \end{cases}. \tag{5}$$

We call $\mathcal{C}(x)$ the $\mathcal{C}$-transform of x with respect to the *basis vector* $\mathcal{U}_M$. Intuitively, one can view the $\mathcal{C}$-transform as a "greedy" binary numeral system as the $\mathcal{C}$-transform of x is obtained by recursively subtracting x from u_M. In particular, if we choose $u_i = 2^{i-1}$ for all i, then the $\mathcal{C}$-transform of x is simply the usual binary representation of x. Moreover, if we choose $u_i = 2^{i-1}$, $1 \leq i \leq s$, and $u_i = \sum_{\ell=i-s}^{i-1} u_\ell$, $i \geq s+1$, then it is the normal-form Fibonacci number system of order s in [10].

One of the most important properties of the $\mathcal{C}$-transform is the complete decomposition property, i.e., every integer (within the representation range) can be written as a sum of distinct u_i's. For example, consider the 5-vector $\mathcal{U}_5 = (1, 2, 3, 6, 10)$ as the basis vector. Then $\mathcal{C}(14) = (1, 0, 1, 0, 1)$. Note that $14 = 1\times1 + 0\times2 + 1\times3 + 0\times6 + 1\times10$ and it can be written as a sum of distinct u_i's. The complete decomposition property was previously proved in Lemma 5 of [8] and it is stated formally in the following proposition. A similar result was also reported in Lemma 3.1 of [2].

Proposition 2: **(Complete decomposition)** Assume that

(A1) $u_1 = 1$, and $1 \leq u_{i+1} \leq \sum_{k=1}^{i} u_k + 1, i = 1, 2, \ldots, M-1$

Then $x = \sum_{k=1}^{M} d_k(x) \cdot u_k$ for $0 \leq x \leq \sum_{k=1}^{M} u_k$.

With an additional constraint on the choice of u_i's in (A2) below, we show that the $\mathcal{C}$-transform further has the following property. This property will be used for the constructions of FOCs and FTCs.

Lemma 3: Assume that (A1) in Proposition 2 holds and

(A2) for some $\ell \geq 2$, $u_{i+1} \leq \sum_{k=i-\ell+1}^{i} u_k$ for $i = \ell, \ldots, M-1$.

Then for all $0 \leq x < \sum_{k=M-\ell+1}^{M} u_k$, there are no ℓ consecutive 1's in $\mathcal{C}(x)$, i.e., there does not exist any i such that $d_i(x) = d_{i-1}(x) = \ldots = d_{i-\ell+1}(x) = 1$.

Proof.

We prove this by contradiction. Since we assume that $x < u_M + u_{M-1} + \cdots + u_{M-\ell+1}$, from the definition of the $\mathcal{C}$-transform it is impossible to have $d_M(x) = d_{M-1}(x) = \ldots = d_{M-\ell+1}(x) = 1$.

Now suppose there are ℓ consecutive 1's in $\mathcal{C}(x)$. This implies that there exists an index i such that $d_{i+1}(x) = 0$ and $d_i(x) = d_{i-1}(x) = \ldots = d_{i-\ell+1}(x) = 1$ in $\mathcal{C}(x)$. Since $d_{i+1}(x) = 0$, it then follows from the complete decomposition property in Proposition 2 that

$$x - \sum_{k=i+2}^{M} d_k(x) \cdot u_k = x - \sum_{k=i+1}^{M} d_k(x) \cdot u_k$$
$$= \sum_{k=1}^{i} d_k(x) \cdot u_k \geq \sum_{k=i-\ell+1}^{i} u_k.$$

Since we assume that $u_{i+1} \leq \sum_{k=i-\ell+1}^{i} u_k$ for $i = \ell, \ldots, M-1$, we then have $x - \sum_{k=i+2}^{M} d_k(x) \cdot u_k \geq u_{i+1}$. This implies that $d_{i+1}(x) = 1$ from the definition of the $\mathcal{C}(x)$ and we reach a contradiction.

■

One particular sequence that satisfies the assumptions in (A1) and (A2) with $\ell = 2$ is to choose the Fibonacci sequence, i.e., $u_1 = 1$ $u_2 = 2$ and $u_{i+1} = u_i + u_{i-1}$ for $i = 2, \ldots, M-1$. Then Lemma 3 recovers the well known Zeckendorf theorem [11], i.e., every positive integer can be written as the sum of one or more distinct Fibonacci numbers in such a way that the sum does not include any two consecutive Fibonacci numbers (see also Lemma 1 in [10]).

III. MEMORYLESS FOCs

In this section, we show how one can construct memoryless FOCs by using the $\mathcal{C}$-transform.

Consider a symbol set S. An M-dimensional memoryless binary code C for S is a mapping that maps every element in S to a codeword with an M-dimensional binary representation. An M-dimensional memoryless binary code for S is a *forbidden overlap code* (FOC) if a transition from one codeword to another codeword does not have the following two types of transitions for any three adjacent bits: $101 \rightarrow 010$ or $010 \rightarrow 101$.

Algorithm for the construction of a FOC:

Symbol set: Let $u_1 = 1$, $u_2 = 2$, $u_3 = 4$ and $u_{i+1} = u_i + u_{i-1} + u_{i-2}$ for $i = 3, \ldots, M-1$. Consider the symbol set

$$S = \{0, 1, 2, \ldots, u_M + u_{M-1} + u_{M-2} - 1\}.$$

Encoding: For $x \in S$, compute the $\mathcal{C}$-transform of x,

$$\mathcal{C}(x) = \Big(d_1(x), d_2(x), \ldots, d_{M-1}(x), d_M(x)\Big).$$

Generate the M-dimensional binary codeword for x, denoted by $\mathbf{c}(x) = (c_1(x), c_2(x), \ldots, c_M(x))$, by

$$c_i(x) = \begin{cases} d_i(x), & \text{if } i \text{ is odd}, \\ 1 - d_i(x), & \text{if } i \text{ is even}. \end{cases} \tag{6}$$

Decoding: For a binary codeword $\mathbf{c} = (c_1, c_2, \ldots, c_M)$, generate the M-vector $(d_1, d_2, \ldots, d_M)$ by

$$d_i = \begin{cases} c_i, & \text{if } i \text{ is odd}, \\ 1 - c_i, & \text{if } i \text{ is even}. \end{cases} \tag{7}$$

Decode the codeword $\mathbf{c}$ as

$$x = \sum_{i=1}^{M} d_i \cdot u_i. \tag{8}$$

Note that the steps in (6) and (7) are simply to invert every even-numbered bit. Thus, they are inverse functions of each other. Since we choose $u_1 = 1$, $u_2 = 2$, $u_3 = 4$ and $u_{i+1} = u_i + u_{i-1} + u_{i-2}$ for $i = 3, \ldots, M-1$, the assumption (A1) in Proposition 2 holds and thus we have from the complete decomposition property that every codeword can be decoded correctly by using (8).

In the following theorem, we show that the set of codewords generated by the above algorithm is indeed a FOC.

Theorem 4: The set of codewords $\{\mathbf{c}(x) = (c_1(x), c_2(x), \ldots, c_M(x)),\ x \in S\}$ generated by the above algorithm is indeed a FOC. Moreover, it is optimal in the sense that it has the largest number of codewords in a memoryless M-dimensional FOC.

Proof.

For this, we need to show a transition from one codeword $\mathbf{c}(x_1)$ to another codeword $\mathbf{c}(x_2)$ does not have the following two types of transitions for any three adjacent bits: $101 \to 010$ or $010 \to 101$.

We prove this by contradiction. First, since we choose $u_1 = 1$, $u_2 = 2$, $u_3 = 4$ and $u_{i+1} = u_i + u_{i-1} + u_{i-2}$ for $i = 3, \ldots, M-1$, the assumptions (A1) and (A2) in Lemma 3 are satisfied with $\ell = 3$. Thus,we know from Lemma 3 that for all $x \in S$, there are no *three* consecutive 1's in $\mathcal{C}(x)$.

Case 1: There is a transition of $101 \to 010$:

Suppose that for some $x_1, x_2 \in S$ and some i such that $(c_i(x_1), c_{i-1}(x_1), c_{i-2}(x_1)) = (1, 0, 1)$ and $(c_i(x_2), c_{i-1}(x_2), c_{i-2}(x_2)) = (0, 1, 0)$. If i is an even number, then $d_i(x_2) = 1 - c_i(x_2)$, $d_{i-1}(x_2) = c_{i-1}(x_2)$, and $d_{i-2}(x_2) = 1 - c_{i-2}(x_2)$. Thus, we have

$$(d_i(x_2), d_{i-1}(x_2), d_{i-2}(x_2)) = (1, 1, 1).$$

This contradicts to the result that there are no three consecutive 1's in the $\mathcal{C}$-transform. On the other hand, if i is an odd number, $d_i(x_1) = c_i(x_1)$, $d_{i-1}(x_1) = 1 - c_{i-1}(x_1)$, and $d_{i-2}(x_1) = c_{i-2}(x_1)$. Thus, we have

$$(d_i(x_1), d_{i-1}(x_1), d_{i-2}(x_1)) = (1, 1, 1).$$

This also contradicts to the result that there are no three consecutive 1's in the $\mathcal{C}$-transform.

Case 2: There is a transition of $010 \to 101$:

The argument for this case is exactly the same as Case 1 with x_1 and x_2 being interchanged.

It is known (see e.g., [12], [2]) that the largest number of codewords in a memoryless M-dimensional FOC is N_M, where N_M is characterized by the following recursive equation:

$$N_{i+1} = N_i + N_{i-1} + N_{i-2}, \ \ 3 \le i \le M-1. \tag{9}$$

with $N_1 = 2$, $N_2 = 4$, and $N_3 = 7$. It is easy to see from (9) that $N_i = u_{i+1}$, $1 \le i \le M-1$ and $N_M = u_M + u_{M-1} + u_{M-2}$. Thus, our construction of the M-dimensional FOC indeed has the largest number of codewords. ∎

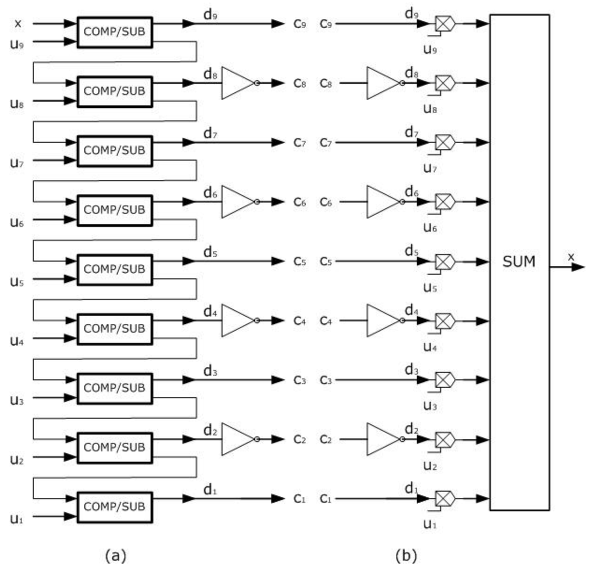

Fig. 1. (a) The 8B/9B FOC encoder, and (b) The 8B/9B FOC decoder.

As shown in [2], efficient FOC codes cannot be constructed by using their numeral systems. Instead, they only construct a suboptimal FOC using their numeral systems. In comparison with the suboptimal FOC in [2], we note that the (asymptotic) code rate in [2] is 0.7925, while the code rate of our optimal memoryless FOC is 0.8791. The key difference between theirs and ours is that we add the inverters on the even-numbered buses so that an optimal FOC can be represented by using numeral systems and these inverters.

We also note that the hardware implementation complexity of our algorithm is $O(M^2)$. This is the same as those in [5], [2] because they all require implementing numeral systems. As an illustrating example, we show the block diagram of the 8B/9B FOC encoder and decoder in Figure 1. The encoder takes an 8-bit input and encodes it into a 9-bit codeword. The basis vector $(u_1, \ldots, u_9)$ is shown in the 8B/9B FOC column of Table I. The COMP/SUB processing unit in the encoder in Figure 1 is further illustrated in Figure 2. These are similar to the implementations of the numeral systems in [5], [2] (except the additional inverters).

The decoder in Figure 1(b) takes a 9-bit codeword and decodes it into an 8-bit output. It consists of three stages. In the first stage, all the even-numbered bits are *inverted*. In the second stage, the i^{th} output after the first stage is then multiplied by u_i $i = 1, 2, \ldots, 9$. In the third stage, the 8-bit output is generated by summing up the outputs from the second stage.

Recently, Mutyam [10] used the transition signaling technique [13] to construct FTCs. The transition signaling technique takes a set of input data indexed in time and computes the *transition signals* (by using the exclusive OR operation)

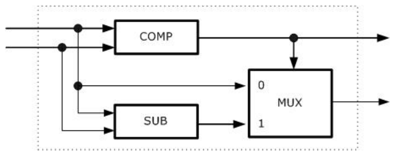

Fig. 2. The COMP/SUB processing unit in the encoder in Figure 1(a).

	8B/9B FOC	6B/9B FTC	$(9,6,3)$-NAT
u_1	1	1	1
u_2	2	2	2
u_3	4	3	3
u_4	7	5	5
u_5	13	8	8
u_6	24	13	12
u_7	44	21	20
u_8	81	34	32
u_9	149	55	48

TABLE I
THE BASIS VECTORS FOR THE 8B/9B FOC, THE 6B/9B FTC AND THE $(9,6,3)$-NAT CODE.

between any two successive input data. The transition signals are then sent through the bus. We note that the transition signaling technique can also be used here to construct FOCs. It is easy to show if there are no 3 consecutive 1's in each input data, then by sending the transition signals of these input data there are no transitions for any three adjacent bits: $101 \rightarrow 010$ or $010 \rightarrow 101$. However, the FOCs constructed this way are not *memoryless* as the encoder has to store the previous input data for computing the transition signals. Moreover, a simple bit error in the bus might have a cascading effect in the decoder that might cause a serious decoding failure. Since both the transition signaling technique and our construction for FOCs have the same encoding efficiency, our *memoryless* construction is clearly a better choice than the transition signaling technique as it not only requires lower hardware implementation complexity (in both encoding and decoding) but also it is more reliable in decoding (as it does not have the cascading decoding failure problem).

We note that our approach can also be used for the constructions of optimal memoryless FTCs. This is done by choosing the basis vector with $u_1 = 1$, $u_2 = 2$ and $u_{i+1} = u_i + u_{i-1}$ for $i = 2, \ldots, M-1$. For the symbol set $S = \{0, 1, 2, \ldots, u_M + u_{M-1} - 1\}$, we can then construct an optimal FTC by using the encoding scheme and the decoding scheme described in the algorithm for the construction of a FOC. Even though the construction in [5] and ours are both optimal, these two sets of codewords are different as each codeword in [5] has a Fibonacci representation. As an illustrating example, we can use the block diagram in Figure 1 for a 6B/9B FTC encoder/decoder. The basis vector $(u_1, \ldots, u_9)$ is shown in the 6B/9B FTC column of Table I.

IV. NO ADJACENT TRANSITION CODES

The $\mathcal{C}$-transform in [9] for constructing routing paths with a limited number of recirculations in optical queues in fact generates limited-weight codes. As such, in conjunction with Lemma 3, one can use the $\mathcal{C}$-transform to construct limited-weight No Adjacent Transition (NAT) codes in [7], where there are no two consecutive 1's in each codeword. An (M, n, k)-NAT code consists of three parameters, where M is the length of the codeword, n is the length of the dataword, and k is the maximum weight (the maximum number of 1's) in each codeword. As shown in Lemma 2 of [10], when an NAT code is transmitted through a bus via the transition signalling technique, there are no transitions for any *two* adjacent bits of these two types: $10 \rightarrow 01$ or $01 \rightarrow 10$. Thus, they can be used for FTCs. Moreover, the total number of transitions (either from 0 to 1 or from 1 to 0) over a period of time in the bus is the same as the total number of 1's in the transmitted codewords (Property 1 of [7]). As such, NAT codes with the transition signaling technique not only tackle the crosstalk problem but also the power consumption problem.

It is known [7] that the largest number of M-bit NAT codes with maximum weight k is $g(M,k) = \sum_{i=0}^{k} f(M,i)$, where

$$f(M,i) = \binom{M+1-i}{i} \tag{10}$$

is the largest number of M-bit NAT codes with weight i. By ranking all the $g(M,k)$ NAT codes first by their weights and then by their values, it was shown in [7] there is an encoding/decoding algorithm that generates the largest number of M-bit NAT codes with maximum weight k. However, such an approach requires storing all the $f(M,i)$'s in (10), $i = 1, 2, \ldots, k$, $m = 1, 2, \ldots, M$ for comparison. Thus, the hardware implementation complexity could be very high if k is proportional to M.

In the following algorithm, we describe how one generates an NAT code via the $\mathcal{C}$-transform. Our algorithm does not generate the largest number of M-bit NAT codes with maximum weight k. However, it only requires storing the basis vector $(u_1, \ldots, u_M)$ for comparison and thus greatly reduces the implementation complexity. Let $B(u_1, \ldots, u_M; k)$ be the maximum representable integer via the $\mathcal{C}$-transform with respect to the basis vector $(u_1, \ldots, u_M)$ under the constraint of maximum weight k, i.e.,

$$\begin{aligned} & B(u_1, \ldots, u_M; k) \\ & = \max\left\{ y : \sum_{i=1}^{M} d_i(x) \le k,\ x = 0, 1, \ldots, y \right\}. \end{aligned}$$

For obvious reasons, we also define $B(u_1, \ldots, u_M; k) = 0$ if $M = 0$ or $k = 0$.

Algorithm for constructing an (M, n, k)-NAT code via the $\mathcal{C}$-transform:

Finding the basis vector $(u_1, u_2, \ldots, u_M)$: Choose positive integers $n_1, n_2, \ldots, n_k$ such that $\sum_{i=1}^{k} n_i = M$ and $n_1 \ge 2$. Let $s_0 = 0$ and $s_i = \sum_{\ell=1}^{i} n_\ell$ for $i = 1, 2, \ldots, k$. Let

$u_j = j, \quad j = 1, 2, \ldots, s_1$. Find the rest of the basis elements recursively as follows:

$$u_{s_i+j} = \min\Big[u_{s_i+j-1} + u_{s_i+j-2}, \\ B(u_1, \ldots, u_{s_i+j-1}; i+1) + 1\Big], \qquad (11)$$

where $1 \le i \le k-1,\ 1 \le j \le n_{i+1}$.

Encoding: Consider the symbol set $S = \{0, 1, 2, \ldots, 2^n - 1\}$, where

$$n = \lfloor \log_2(\min[u_M + u_{M-1}, B(u_1, \ldots, u_M; k) + 1]) \rfloor. \qquad (12)$$

For $x \in S$, compute the $\mathcal{C}$-transform of x with respect to the basis vector $(u_1, u_2, \ldots, u_M)$,

$$\mathcal{C}(x) = \Big(d_1(x), d_2(x), \ldots, d_{M-1}(x), d_M(x)\Big).$$

Decoding: For a binary codeword $\mathbf{c} = (d_1, d_2, \ldots, d_M)$, decode the codeword $\mathbf{c}$ as

$$x = \sum_{i=1}^{M} d_i \cdot u_i. \qquad (13)$$

Theorem 5: The set of codewords $\{\mathbf{c}(x) = (c_1(x), c_2(x), \ldots, c_M(x)), \quad x \in S\}$ generated by the above algorithm is indeed an (M, n, k)-NAT code.

Proof. Note that the choice of the basis elements in (11) ensures $u_{i+1} \le u_i + u_{i-1}$ for $i = 2, \ldots, M-1$ and that the choice of n in (12) ensures $x < u_M + u_{M-1}$ for all $x \in S$. From Lemma 3, it follows that there are no two consecutive 1's in each codeword. Also, we have from (12) that there are at most k 1's in each codeword. Thus, the encoding scheme generates an (M, n, k)-NAT code. The complete decomposition property in Proposition 2 then ensures the correctness of the decoder in (13). ■

There are many choices of $n_1, n_2, \ldots, n_k$ such that $\sum_{i=1}^{k} n_i = M$. For the setting in [9], the optimal choices can be found via a specific algorithm. However, it is much more involved to find the optimal choice in this setting. As an illustrating example, we construct a $(9, 6, 3)$-NAT code. For this example, we have $M = 9$ and $k = 3$. We choose $n_1 = n_2 = n_3 = 3$. As a result of the above algorithm, the basis vector $(u_1, \ldots, u_9)$ is shown in the $(9, 6, 3)$-NAT column of Table I. This is better than using the Fibonacci number system of order 2 in the FTC column of Table I as it only generates a $(9, 6, 4)$-NAT code. Also, since

$$g(9, 3) = \sum_{i=0}^{3} \left(\begin{array}{c} 9+1-i \\ i \end{array} \right) = 73,$$

the length of the dataword is at most 6 for a $9-$bit NAT code with maximum weight 3. Thus, such a $(9, 6, 3)$-NAT code is optimal among all the $9-$bit NAT codes with maximum weight 3. Also, the hardware implementation complexity of our $(9, 6, 3)$-NAT code is much simpler than that in [7]. We also test our algorithm for various values of M with $M = 3k$ and $k = 3, 4, \ldots, 12$. Our numerical results show that our algorithm generates an optimal (M, n, k)-NAT code for $3 \le k \le 11$. However, for $k = 12$, there exists an optimal $(36, 25, 12)$-NAT code and our algorithm only generates a $(36, 24, 12)$-NAT code.

V. Conclusion

In this technical brief, we developed an explicit construction for a set of memoryless forbidden overlap codes (FOCs). Such a set of memoryless FOCs also contains the largest number of codewords. The same approach can also be applied for the construction of a set of memoryless forbidden transition codes (FTCs) and a set of No Adjacent Transition (NAT) codes. Both FOCs and FTCs considered in this technical brief are memoryless codes, and their code rates could be significantly improved by considering codes with memory. In [14], it was shown that there exists a simple bit stuffing algorithm that yields a much higher code rate than the Fibonacci representation in [5]. Beside, it hardware implementation complexity is only $O(M)$ for an M-bit bus, which is also much lower than the $O(M^2)$ complexity in [5]. Extension along this line for FOCs will be reported separately.

References

[1] P. P. Sotiriadis, "Interconnect modeling and optimization in deep submicron technologies," Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 2002.

[2] X. Wu and Z. Yan, Efficient CODEC designs for crosstalk avoidance codes based on numeral systems, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, pp. 548–558, 2011.

[3] B. Victor and K. Keutzer, "Bus encoding to prevent crosstalk delay," in *Proceedings IEEE/ACM International Conference on Computer-Aided Design (ICCAD'01)*, San Jose, CA, USA, 2001. pp. 57–63.

[4] M. Mutyam, "Preventing crosstalk delay using Fibonacci representation," in *Proceedings 17th International Conference on VLSI Design (VLSID'04)*, Mumbai, India, 2004, pp. 685–688.

[5] C. Duan, C. Zhu, and S. P. Khatri, "Forbidden transition free crosstalk avoidance CODEC design," in *Proceedings 45th Annual Design Automation Conference (DAC'08)*, Anaheim, CA, USA, 2008, pp. 986–991.

[6] C. Duan, V. C. Calle, and S. Khatri, "Efficient on-chip crosstalk avoidance codec design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, pp. 551–560, 2009.

[7] P. Subramanya, R. Manimeghalai, V. Kamakoti, and M. Mutyam, "A bus encoding technique for power and cross-talk minimization," in *Proc. IEEE Int. Conf. VLSI Design*, 2004, pp. 443–448.

[8] C.-C. Chou, C.-S. Chang, D.-S. Lee and J. Cheng, "A necessary and sufficient condition for the construction of 2-to-1 optical FIFO multiplexers by a single crossbar switch and fiber delay lines," *IEEE Transactions on Information Theory*, vol. 52, pp. 4519–4531, 2006.

[9] J. Cheng, C.-S. Chang, T.-H. Chao, D.-S. Lee, and C.-M. Lien, "On constructions of optical queues with a limited number of recirculations," in *Proceedings of IEEE INFOCOM 2008.*

[10] M. Mutyam, "Fibonacci codes for crosstalk avoidance," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, pp. 1899-1903, 2012.

[11] http://en.wikipedia.org/wiki/Zeckendorf's_theorem.

[12] S. R. Sridhara and N. R. Shanbhag, "Coding for reliable on-chip buses: A class of fundamental bounds and practical codes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 26, pp. 977–982, May 2007.

[13] M. Stan and W. Burleson, "Limited-weight codes for low power I/O," in *Proc. IEEE/ACM Int. Workshop Low Power Design*, 1994. pp. 209–214.

[14] C.-S. Chang, J. Cheng, T.-K. Huang, X.-C. Huang, and D.-S. Lee, "A bit-stuffing algorithm for crosstalk avoidance in high speed switching," in *Proceedings of IEEE INFOCOM 2010.*