Encoding Serial Graphical Data for EDP/Energy Minimization

Sasidharan Ekambavanan[1], Rajesh Garg[2], Sunil P. Khatri[3*] and Krishna R. Narayanan[3]

[1]Link-A-Media Devices, Santa Clara, CA 95051
e_sasidharan@yahoo.co.in

[2]Intel Corporation, Hillsboro, OR 97124.
rajesh.garg@intel.com

[3]Department of ECE, Texas A&M University, College Station, TX 77843.
sunilkhatri@tamu.edu, krn@ee.tamu.edu

*Corresponding author: Sunil P. Khatri

Address:
Department of Electrical and Computer Engineering
333F WERC, MS 3259
Texas A&M University
College Station, TX 77843-3259

Phone : 979-845-8371
Fax : 979-845-2630
E-mail : sunilkhatri@tamu.edu

Date of Receiving:

Date of Acceptance:

Encoding Serial Graphical Data for EDP/Energy Minimization

Sasidharan Ekambavanan, Rajesh Garg, Sunil P. Khatri and Krishna R. Narayanan

Abstract

In this paper we report a novel approach to reduce the Energy-Delay Product (EDP) and energy for graphical data transmission over serial buses. For any symbol that is to be transmitted on the bus, we select the best code (from an EDP or energy minimization standpoint) from among a set of codebooks. In particular, our implementation utilizes 3 codebooks. Our approach is compared with other reported techniques in the literature, and the results are quite promising. We demonstrate that our technique achieves a 10% improvement in EDP over the best approach known for serial data transmission, with a significant improvement in encoding/decoding complexity and energy. We also show that the average number of transitions of our scheme is 10.9% above the theoretical lower bound (in contrast to the best approach known for serial transmission of LCD data, which is 37.6% above the theoretical lower bound).

Keywords: Encoding, serial buses, energy, EDP, TMDS

I. Introduction

Recently, energy has become a major issue in computing [1]. This has led to a great deal of interest in investigating design styles that are energy aware. Low energy solutions are desired for many applications such as multimedia devices, microprocessors, wireless communications, etc. Bus encoding has been proposed for on/off-chip memory/processor buses to reduce power/energy consumption by minimizing switching activities [2], [3], [4], [5], [6]. Although this is a very effective way of reducing power/energy, it was not used for peripherals such as liquid crystal displays (LCDs) until a couple of years ago.

LCDs are one of the most power/energy consuming components of modern multimedia devices. Until a few years ago, an LCD panel consumed most of the power of the LCD sub-system. But with technological advances, the LCD panel power consumption has been reduced considerably [7]. Due to this, the power consumed by the *serial bus* connecting the LCD to the graphics controller has become a significant fraction of the total power consumption (usually more than 10-15% [8]) of the entire LCD system. This bus is usually implemented through a serial cable such as a DVI cable. These cables have a mutual capacitance of the order of tens of pF/m, with cable lengths as high as few meters [8] (much higher than the lengths of on/off-chip buses). Therefore, power/energy reducing bus encoding techniques for serial buses can be effectively used for LCD buses. Also, the large capacitance of the LCD serial bus allows the designer to expend more power/energy in the encoding/decoding electronics, and still obtain a significant net power/energy reduction. It should be noted that compared to conventional parallel on/off-chip buses, LCDs employ serial data links. *The focus of this work is energy-delay-product (EDP) minimization in serial buses*, and hence our work is directly applicable in a LCD setting. EDP reduction is a crucial metric in a serial data transmission setting, since we not only need to minimize energy, but also the total duration of the serial data transmission. To the best of the authors' knowledge, this is the first work to address EDP minimization in serial data transmission. Also this work is the first to utilize multiple codebooks (> 2) in such a setting. Herein lies the contribution of our approach.

There exist various LCD interface standards such as DVI (Digital Visual Interface) [9], OpenLDI (Open LVDS Digital Interface) [10], GVIF (Gigabyte Video Interface) [11], etc. Even though some of these standards have been designed to reduce the number of transitions by encoding the pixels, power, energy or EDP minimization was not the driving reason behind these standards. The standards were devised mainly to avoid excessive electromagnetic interference (EMI) levels from the cable. Recently some approaches have been proposed for power/energy efficient [7], [12], [13], [8] data transmission for some of these standards; however, none of them have explored the multiple codebook approach proposed in this paper, and hence these methods are not able to achieve large energy or EDP savings. In this work, we propose a multiple codebook scheme which uses a widely exploited property of digital images, that is, their inter-pixel correlations. Pixel *differences* are encoded rather than the pixel value itself, and large energy and EDP savings are achieved. The approach consists for 2 phases. In the first phase, codebooks are constructed, using data from a set of representative images. In phase II, The same codebooks are used to transmit any arbitrary image. For the DVI-compliant interface, our method is able to reduce EDP by approximately 78% over *Transition-Minimized Differential Signaling* (TMDS) encoding scheme [9]. The best known approach [8] achieves approximately a 68% improvement in EDP, over TMDS. Though this paper demonstrates the usefulness of our approach by applying it to the DVI interface, our approach *can be applied to any serial data transmission protocol*.

The rest of the paper is organized as follows. Section II presents the previous work done for reducing the transitions on a LCD serial data link. Section III describes our encoding approach, along with a discussion on its proximity to the theoretical lower bound of the number of transitions per symbol. Section IV reports the results obtained by applying our approach to the set of images taken from the SIPI database [14] and from [15]. Finally, conclusions are presented in Section V.

II. Previous Work

A significant amount of work has been done for reducing power/energy consumption in on/off-chip memory/processor (parallel) buses by bus encoding [2], [3], [4], [5], [6], but there has not been much attention devoted to serial buses until a few years ago. As mentioned in the last section, bus encoding is crucial for the serial data link of the LCD because of the large capacitance of these links compared to the relatively short memory/processor buses.

Information theoretic bounds on the average signal transition activity in a bus, given a source entropy rate $\mathcal{H}$ and an average rate R (bits per symbol) of transmission, have been provided in [16]. A lower bound on the *Energy-Delay-Product (EDP)*[1], given $\mathcal{H}$ and R, has also been provided. These bounds were used to determine the activity-reducing efficiency of various algorithms like entropy coding, transition signaling and bus-invert coding. Although [16] provides a strong theoretical analysis of the bus transition activity, practical schemes which achieve this lower bound were not presented.

In [17], an XOR-based encoding scheme was presented for serial communication. Here, each bit in a data word is XOR-ed (or *differentially encoded*) with the previous bit in the same data word. After this differential encoding, the bits are serialized and transmitted. The method in [17] can work only in a setting were the adjacent bits in a data-word are identical with a high probability. Also, the method aims only to reduce the number of 1's, and does not attempt to directly reduce the number of *transitions*. Reducing the number of 1's does not always reduce the number of transitions.

In the literature, methods for energy-efficient LCD sub-systems have mostly been restricted to specific control circuits inside the LCD controller [18], [19], or to the design of individual components of the display systems [20], [21]. The first low power encoding technique for data transmission on the digital LCD interface was proposed in [7], for the DVI standard. This technique is called *Chromatic Encoding*. One of the observations made in [7] is that the difference between adjacent pixel-values in images follows a non-uniform distribution with zero mean, and the majority of the pixel difference values concentrated near zero. A plot of this distribution for some images in the SIPI database [14] is also shown in [7]. Similar plots for the blue, green and red components of 10 images in the SIPI database [14] are shown in Figures 1, 2 and 3 respectively. Based on this distribution, an optimal code assignment is performed to minimize the transition counts, using a *single codebook*. Also the three color channels of the DVI interface are reciprocally encoded (i.e. the difference between the R, G and B color symbol values are encoded) to obtain further power savings. Although significant savings have been reported, there are several cases in which pixels cannot be encoded (such as overflow or underflow conditions, or situations in which the encoding must switch to plain TMDS [9], which is explained in the sequel).

Another low power encoding technique, called *Differential Bar-Encoding* was proposed in [12], also for the DVI standard. The approach of [12] was improved upon in [13] and [8], using a technique called Limited Intra-Word Transition (LIWT)

[1]Note that [16] assumes that frequency is constant, and hence energy is proportional to power. Therefore the Power-Delay Product reported in [16] is directly proportional to the EDP metric used in this paper.

codes. In [13], the technique was applied for DVI and OpenLDI, whereas in [8] it was applied for the DVI, OpenLDI and GVIF interface standards. In [13] and [8], pixel difference values within a range Δ (which is chosen so as to cover the majority of the distribution of difference values) are encoded using a *single codebook* (with all codewords of the same length), and "plain" pixel values are transmitted for the other cases. The codewords are chosen such that the number of transitions within a codeword is $\leq k$ and hence this approach was called $k-$LIWT. This approach is explained in further detail in Section III-B. Although the LIWT encoding improves the power/energy consumption compared to the work done in [7], it does not directly attempt to minimize EDP. It tries to minimize energy without considering the rate of transmission. The results presented in [7] do not report the EDP of the resulting encoded data.

All the above mentioned methods deal with the problem of power/energy savings, but do not directly consider the problem of minimizing the energy-delay-product (EDP). Energy can be minimized by sending a long codeword with fewer transitions. However, this would reduce the rate of transmission, increasing the total time for serial data transmission. Hence there is a trade-off between the transmission rate and energy. In order to take into account both these factors *simultaneously*, our scheme minimizes the *product* of the number of transitions and the length of the final encoded data. This product is called the Energy-Delay-Product (EDP) for serial data transmission. Another significant contribution of this work is the use of multiple codebooks (of varying lengths) to minimize the EDP. In particular, the experiments were performed with 3 codebooks. The results show that our approach yields a 10% improvement in EDP over the best energy reducing approach (i.e. the 2-LIWT scheme [8]) known for serial data transmission. The encoding/decoding complexity[2] of one of our best codes is 59.42% that of the 2-LIWT scheme [8]. Also, the average number of transitions of our scheme is 10.9% above the theoretical lower bound [16] (in contrast to the 2-LIWT scheme [8], which is 37.6% above the theoretical lower bound).

III. Our Approach

In this work, we compare our results with that of the LIWT encoding algorithm of [8]. Our results, as well as those of [8] are presented with respect to the TMDS [9] encoding scheme. As a consequence, we first briefly present the TMDS and the LIWT encoding approaches before providing details of our encoding technique.

A. The TMDS scheme

The DVI standard is based on TMDS [9]. This interface encodes and serializes the digital RGB data and then serially transmits the data on three twisted pair cables, one each for the R, G and B colors. An additional twisted pair is used for the clock signal. Each pixel data, which is 8 bits long, is encoded and serialized into 10 bits. TMDS uses transition coding, where *transitions* are encoded instead of actual values, to obtain low transition counts. Of the 2 additional bits, the first additional bit determines the type of encoding. When the first additional bit is 1 (0), transitions are encoded by a Boolean XOR (XNOR) operation. The second additional bit determines whether the transmitted word is complemented or not. This is done for DC balancing, i.e., to balance the number of 0's and 1's sent on the bus.

[2]The encoder and decoder for our scheme can be realized using a Look-Up Table (LUT), as was done in the 2-LIWT scheme [8]. The encoding/decoding complexity is measured in terms of the *Code Size Ratio (CSR)*, which is defined as the ratio of the size of our encoder/decoder Look-Up Table (LUT) compared to that of the 2-LIWT scheme [8].

Figure 4 describes the timing diagram of a 24-bit, XOR-based, Single-Link TMDS scheme. R_i, G_i and B_i denote the *i*-th bit of R, G, and B channel. In this figure, $C0$, $C1$ and $C2$ are the sequence of codewords transmitted on the R, G and B color channels respectively.

B. LIWT Encoding

The LIWT encoding approach exploits the spatial correlation of digital images. A distribution of the difference between adjacent pixel values is given in [8]. This distribution turns out to be non-uniform with zero mean and a majority of the pixel difference values concentrated near zero.

The LIWT encoding encodes inter-pixel difference values which are within a range Δ, into a 9-bit word with fewer transitions. The other pixel values are transmitted in an uncoded manner (8 bits long). Whether the transmitted data is encoded or not is indicated by a *conditional* bit. Hence the serially transmitted data is either 10 bits long (when encoded) or 9 bits long (when not encoded), for an 8 bit pixel value.

Algorithm 1 LIWT Algorithm

1: $Encode(w_t, w_{t-1})\{$
2: *Diff* $= w_t - w_{t-1}$
3: **if** *Diff* $\in \Delta$ **then**
4: $\quad Code_t = map($*Diff*$)$
5: **else**
6: $\quad Code_t = w_t$
7: **end if**
8: return $Code_t$
9: $\}$

Algorithm 1 describes the LIWT algorithm. The two consecutive pixel values are denoted as w_t and w_{t-1}. The pixel-differences (called symbols) within the range Δ are mapped in decreasing order of occurrence probability, to codewords sorted in increasing order of their number of intra-word transitions. If the absolute value of the pixel-difference is not within the range Δ, then w_t is transmitted without encoding. The range Δ depends on the number of bits per pixel. The codewords are chosen such that for a *k*-LIWT approach, the intra-word transitions of the codeword (including the conditional bit) is less than or equal to *k*.

C. Our Approach

The major thrust of our approach (referred to as the *Multiple Codebook Approach (MCA)*) is EDP minimization, as opposed to power/energy minimization as in [7], [12], [13], [8]. However, the MCA *also* achieves better energy savings compared to the best known approach [8] in the literature.

The MCA exploits the inter-pixel correlation of digital images. The difference between adjacent pixels in a digital image was found to follow a non-uniform distribution with zero mean and a majority of the pixel difference values concentrated near zero. Although the variance of the probability distribution for each of the 10 images is different, the shape of the probability distribution remains almost the same for all the images. This property is essential for the working of the MCA.

The MCA scheme encodes pixel differences which are within a range Δ, and transmits the remaining pixel data values as such (uncoded). The key difference between the MCA and the previous techniques [7], [12], [13], [8] is that the MCA employs a *variable length encoding* using N codebooks, each of different length. Also, while assigning symbols to codewords, the MCA scheme directly tries to minimize either EDP or energy (rather than power/energy as in the previous techniques [7], [12], [13], [8]). As we have seen in Section II, EDP is a more relevant metric for the serial data transmission. Note that the MCA encodes symbols using codes from N codebooks. The result is a single codebook, where each codeword has preamble bits indicating which of the N codebooks it came from.

The MCA method consists of several steps, which are briefly outlined below.

- First the MCA considers the serial data stream to be transmitted, and splits it up into contiguous segments of size m bits (for serial LCD data transmission, m is 8, the number of bits used to represent a pixel value). The difference between adjacent pixels can be represented using $m+1$ bits and hence there are $2^{m+1}-1$ difference values. Each of these $2^{m+1}-1$ difference value is henceforth referred to as a *symbol*.
- The $2^{m+1}-1$ symbols are arranged in decreasing order of probability, and only those difference values which fall within a range Δ are encoded. The remaining pixel values are transmitted without encoding. Δ is chosen such that difference values whose magnitude is $\leq \Delta$ cover a majority of the pixel-difference distribution. The difference values whose magnitude is $> \Delta$ would have a cumulative probability less than the difference values whose magnitude is $\leq \Delta$.
- The symbols to be encoded are assigned to codewords, in a manner that minimizes EDP or energy. The algorithm selects a codeword for each symbol from *one of N codebooks*. $N = 2$ and $N = 3$ are the simplest forms of MCA. Although both $N = 2$ and $N = 3$ require 2 preamble bits, $N = 3$ has a larger set of codewords as compared to $N = 2$. Hence, in this work, $N = 3$ was chosen to demonstrate the results.

Since the MCA scheme uses $N = 3$ codebooks, and also does not perform encoding when the difference values fall outside the range Δ, 2 bits are required for the preamble of the codeword. These two bits indicate which of the $N = 3$ codebooks are utilized, in case encoding is performed. They also indicate if no encoding is performed. The lengths of the 3 codebooks are denoted as k, n and p with $k \leq n \leq p$. Table I indicates the meaning of each combination of preamble values. The two preamble bits are referred to as $pre1$ and $pre2$ in Figure 5. Note that the MCA approach combines all these codewords into a single codebook, with each codeword's preamble bits indicating which of the N codebooks it belongs to. To illustrate this, assume we want to transmit symbols s_1, s_2, and s_3. Assume that s_1 is encoded with a k-bit codeword c_1^k, s_2 with a n-bit codeword c_2^n and s_3 with a p-bit codeword c_3^p. The combined codebook is shown graphically in Figure 6. Note that the preamble bits for smaller codewords (length k and n) are assigned such that there are no transitions between $pre1$ and $pre2$, thereby minimizing EDP or energy for the symbols that occur most frequently.

The application of the serial EDP reducing code is primarily targeted to the serial transmission of pixel color information for LCD displays. Therefore, the probability p_i of each symbol (here each symbol is a specific value of adjacent pixel-differences) s_i among the $2^{m+1}-1$ alternatives, is computed first. These probabilities were computed for 10 different (randomly chosen) reference images in the SIPI database [14]. Since each pixel value was represented using 8 bits for these images, $m = 8$, as mentioned previously. The pixel-differences were observed to follow the non-uniform distribution as claimed in [7] and [8].

Next all the symbols s_i are sorted in decreasing order of their probability of occurrence p_i. After this, symbols are assigned to codewords, starting with the symbol with the highest probability. For each symbol, a codeword is assigned from one of N codebooks. The codeword is prefixed by the appropriate preamble bits (to indicate the codebook it belongs to), and transmitted through the serial channel. The codeword prefixed by the preamble bits is hereafter referred to as a *vector*.

The MCA scheme reduces the EDP or energy by reducing the number of intra-vector *and* inter-vector transitions. In the remainder of this section, our method of reducing the intra-vector and inter-vector transitions is described in detail.

Now, using the symbol probability statistics obtained from the 10 reference images, we encode any new image. A simple-minded encoding scheme can be devised to minimize the intra-vector transitions. In such a method, in order to minimize EDP (energy), codewords are sorted in increasing order of EDP (energy). Here, EDP of a codeword c_i is defined as $l_i \cdot t_i$, where t_i is the number of transitions in c_i, and l_i is its length. Energy of a codeword c_i is proportional to the number of transitions t_i. Note that both metrics (EDP and energy) *include transitions due to the preamble bits* (which are used to indicate the codebook from which any codeword is chosen) as well. The next highest probability symbol is assigned to a free codeword c_i (from among *all* codebooks) with the lowest EDP (energy). This codeword is now removed from consideration. In case of a tie in the EDP (energy) value of two codewords, the codeword of smaller length is selected. This process is performed for each symbol, in decreasing order of probability, until all the symbols within a range Δ are covered.

Our experiments show that once we have assigned codewords to minimize the intra-vector transitions in this manner, the inter-vector transitions become a significant portion of the total number of transitions. Hence, reducing the inter-vector transitions *along with* the intra-vector transitions is desired. Our method tries to minimize both the intra-vector and inter-vector transitions simultaneously, as described next.

In order to reduce the inter-vector transitions, the probability of transitions occurring at the interface of two consecutive vectors (since data is transmitted serially) should be known at the time of mapping the symbols to the codewords. In other words, the probability of the occurrence of a "1" value at the front (P_f) and at the back (P_b) of a vector should be known apriori. However, without mapping the symbols to the codewords, these two probabilities cannot be found. Hence, this is a chicken and egg problem. To solve this problem, the probabilities P_f and P_b are calculated in a successive manner as explained below.

After each symbol is assigned a codeword, the probabilities P_f and P_b are updated. These probabilities are used as *apriori information* in assigning a codeword to the next symbol so as to avoid inter-vector transitions as much as possible. In this manner, the codebook generator, after each assignment, *successively* gets a better *estimate* of P_f and P_b, and thereby reduces the inter-vector transitions.

1) Successive Probability Calculation: Assume that we have assigned a codeword to the $(q-1)^{th}$ symbol, and now we are assigning the codeword to the q^{th} symbol. Let p_q denote the probability of the q^{th} symbol sorted in descending order of probability. Let P_f^{old} denote the old probability P_f (i.e. it is the value of P_f after assigning a codeword to the $(q-2)^{th}$ symbol) and P_f^{new} denote the new probability P_f (i.e. it is the value of P_f after assigning a codeword to the $(q-1)^{th}$ symbol). Similarly, P_b^{old} and P_b^{new} are defined accordingly for the back bit of a vector. We now discuss how P_f^{new} and P_b^{new} are computed after assigning codeword to the $(q-1)^{th}$ symbol, and used in assigning the codeword for the q^{th} symbol.

- The first highest probable symbol is always assigned the all-zero codeword from the shortest codebook (i.e. the codebook with length k, with preamble "00"), since this codeword has the least EDP or energy. Hence, initially P_f = 0 and P_b = 0. The second highest probable symbol is assigned a codeword based on these values of P_f and P_b in a manner that minimizes the inter-vector transitions *along with* the intra-vector transitions.
- Once the $(q-1)^{th}$ codeword has been assigned, the value of P_f and P_b need to be updated. Hence, the new values P_f^{new} and P_b^{new} are calculated by the following two equations.

$$P_f^{new} = \frac{P_f^{old}\Big(\sum_{i=1}^{q-2} p_i\Big) + p_{q-1}F}{\sum_{i=1}^{q-1} p_i} \tag{1}$$

$$P_b^{new} = \frac{P_b^{old}\Big(\sum_{i=1}^{q-2} p_i\Big) + p_{q-1}B}{\sum_{i=1}^{q-1} p_i} \tag{2}$$

  where F and B are the value of bits at the front and back of the $(q-1)^{th}$ vector (codeword prefixed by preamble) assigned to the current (i.e. the $(q-1)^{th}$) symbol respectively.
- Now based on these two new probabilities, the next highest probable symbol (i.e. the q^{th}) is assigned a codeword, as described in Section III-C.2 below.
- The probabilities P_f and P_b are updated in a successive manner after each symbol is assigned a codeword. In other words, the P_f^{new} and P_b^{new} values computed while assigning the q^{th} symbol are treated as P_f^{old} and P_b^{old} values respectively while assigning the $(q-1)^{th}$ symbol.

From Equations 1 and 2, it is clear that the current value of P_f^{new} and P_b^{new} (i.e. after the codeword assignment of the $(q-1)^{th}$ symbol) depends on the codeword assignments made to all $q-2$ previous symbols. Hence, as the mapping of the symbols to codewords proceeds, the probabilities P_f^{new} (P_b^{new}) are expected to approach the true probabilities of a bit being "1" at the front and back of a vector in an encoded data stream.

2) Codebook Generation: Symbols are sorted in decreasing order of probability, and the codewords from the 3 codebooks are arranged in increasing order of EDP or energy (depending on whether EDP or energy is minimized respectively). In the sorted list of codewords, we may find groups of codewords with the same EDP or energy. Let us denote each of these groups as g_i. Also, for brevity, we drop the "new" superscript in the following discussion.

During the codeword assignment of the q^{th} symbol, the probabilities P_f and P_b available at this time are used to sort the codewords within each of the groups g_i, based on the following rules.

- If $P_f > 0.5$, then the probability of bit "1" occurring at the front of a vector is higher. Hence, in order to decrease the probability of inter-vector transitions, codewords which end with a bit "1" must be given more preference. On the other hand, if $P_f \leq 0.5$, then codewords which end with a bit "0" must be given more preference.
- If $P_b > 0.5$, then the probability of bit "1" occurring at the back of a vector is more. Hence, in order to decrease the probability of inter-vector transitions, codewords which begin with a bit "1" must be given more preference. On the other hand, if $P_b \leq 0.5$, then codewords which begin with a bit "0" must be given more preference.

Combining the above rules, the algorithm for assigning codewords is as follows.

- Based on the above rules, let F_b and B_b denote front and back bits which would be preferred in the selected codeword vector (codeword prefixed by the preamble). A codeword starting with F_b and ending in B_b would probabilistically minimize inter-vector transitions. Let F and B denote the front and back bit of any codeword vector in the group g_l, such that g_l is the group with the lowest EDP or energy.
- To each codeword within a group g_l, assign a *weight* W as follows.
  - $W = 0$, if $F = F_b$ and $B = B_b$
  - $W = 1$, if ($F \neq F_b$ and $B = B_b$) or ($F = F_b$ and $B \neq B_b$)
  - $W = 2$, if $F \neq F_b$ and $B \neq B_b$
- Now the codewords within group g_l are sorted in ascending order of W.

After the above sorting, the next highest probable (q^{th}) symbol is assigned the first codeword in the sorted list g_i. Then this codeword is removed from consideration. Now the probabilities P_f and P_b are updated based on Equations 1 and 2. The updated probabilities are then used to assign a codeword to the next symbol. This process continues until all symbols (pixel-differences) within a range Δ are assigned a codeword.

Algorithm 2 summarizes the codebook generation process. The parameter "Cost" in Algorithm 2 can be either EDP or energy. Once the codebook (the *map* function) is formed using Algorithm 2, the encoding is done using Algorithm 1 (similar to that of LIWT). It should be noted that the codebook is formed *only once*, using data from a set of representative images. This codebook is used to encode other images (in general, these could be any arbitrary image being transmitted). Note that our codebook has codewords of three different lengths k, n and p. Hence, we treat the codewords to be part of three different codebooks.

Note that codeword vectors of the same codebook (with lengths k, n or p) have the same preamble and hence have the same front bit. Since codeword vectors (in the group g_i) belonging to the same codebook have identical preambles, they have the same number of transitions, and hence end up having the same back bit as well. Hence they do not compete with each other for reducing inter-vector transitions. This is true regardless of whether EDP or energy is being minimized.

The MCA results in an improvement of 10% in EDP over the best known approach [8] to date. The excellent performance of the MCA is due to the fact that the symbols are non-uniformly distributed, which suggests that the data can be compressed before transmission. The use of N codebooks, each of different length, allows us to have codeword lengths which are less than m. This inherently means that we are compressing the data to be transmitted (thereby minimizing delay), while simultaneously minimizing the energy by selecting minimum transition codewords. Also, the MCA uses a novel *successive probability* based approach to minimize the inter-vector transitions along with the intra-vector transitions. Hence the MCA is able to obtain a better EDP reduction compared to previously proposed methods. The MCA also provides an improvement in energy over the previously proposed methods.

The MCA can also incorporate DC balancing, by using DC balancing as an additional "Cost" term while ranking the unassigned codewords in Algorithm 2. In general, DC balancing is not an issue for most LCD displays due to the limited length of connections [8]. Hence, DC balancing was not included in our experiments.

Algorithm 2 Codebook Generation Algorithm

```
1: Generate_Codebook(k, n, p, Δ, Cost){
2: s_i = adjacent pixels' difference
3: p_i = probability of s_i
4: S = List of s_i's such that abs(s_i)≤ Δ
5: S = sort s_i ∈ S in decreasing order of p_i
6: P_f = 0 and P_b = 0
7: C = list of codewords c_i with length k, n and p put together
8: C = sort c_i ∈ C in increasing order of Cost
9: g_i = group of codewords in C with the same cost
10: for q = 0, q < size(S), q = q + 1 do
11:    s_q = S(q)
12:    g_l = group with least cost among g_i's with unassigned codewords
13:    F_b = 0 and B_b = 0
14:    if P_f > 0.5 then
15:       B_b = 1
16:    end if
17:    if P_b > 0.5 then
18:       F_b = 1
19:    end if
20:    for each c_j ∈ g_l do
21:       F = Front bit of c_j
22:       B = Back bit of c_j
23:       if F = F_b and B = B_b then
24:          W_j = 0
25:       else
26:          if (F = F_b and B ≠ B_b) or (F ≠ F_b and B = B_b) then
27:             W_j = 1
28:          else
29:             W_j = 2
30:          end if
31:       end if
32:    end for
33:    g_l = sort c_j ∈ g_l in increasing order of W_j
34:    codeword(s_q) = map(s_q) = first codeword of g_l
35:    Update P_f and P_b using on Equations 1 and 2
36: end for
37: }
```

D. Example of Codebook Generation

The following example would help to better understand the Codebook Generation process. In order to keep the example simple, let us consider a codebook configuration with small values such as $k = 2$, $n = 2$, $p = 2$ and $\Delta = 2$. Tables II, III and IV show the preamble, codewords, Energy and EDP of the k, n and p codebooks respectively. Note that EDP of a codeword c_i is measured by $l_i \cdot t_i$, where t_i is the number of transitions in c_i, and l_i is its length. Energy of a codeword c_i is measured by the number of transitions t_i. Table V shows the codewords from all the 3 codebooks (k, n and p), arranged in ascending order of Energy or EDP (In this particular example, Energy is proportional to EDP since all the 3 codebooks have the same length. Hence, minimizing Energy and EDP would yield the same result. But in general, this may not be the case.). Since $\Delta = 2$, we have 5 symbols (i.e. pixel differences) to be encoded. These 5 symbols arranged in the descending order of their probability are shown in Table VI. Now, the Codebook Generation Algorithm works as follows.

- To start with, the first codeword (which belongs to group g_0) in Table V is assigned to the first symbol (0) in Table VI. Hence, $c_0 = 0000$. Now, $F = 0$, $B = 0$, $P_f^{old} = 0$ and $P_b^{old} = 0$. Using Equations 1 and 2, we get $P_f^{new} = 0$ and $P_b^{new} = 0$.
- Since group g_0 has only one more unassigned codeword, this codeword is assigned to the next symbol (1) in Table VI. Hence, $c_1 = 1111$. Now, $F = 1$, $B = 1$, $P_f^{old} = 0$ and $P_b^{old} = 0$. Using Equations 1 and 2, we get $P_f^{new} = 0.4$ and $P_b^{new} = 0.4$.
- Since group g_0 has no more unassigned codewords, we move to the next group g_1 (having unassigned codewords with the least cost). Now, since $P_f < 0.5$ and $P_b < 0.5$, we get $F_b = 0$ and $B_b = 0$ (see lines 13 through 19 of Algorithm 2). Hence, $W = 1$ for all codewords in group g_1 (see lines 20 through 32 of Algorithm 2). Therefore, the first codeword in group g_1 is assigned to the next symbol (-1) in Table VI. Hence, $c_2 = 0001$. Now, $F = 0$, $B = 1$, $P_f^{old} = 0.4$ and $P_b^{old} = 0.4$. Using Equations 1 and 2, we get $P_f^{new} = 0.29$ and $P_b^{new} = 0.565$.
- Now, since $P_f < 0.5$ and $P_b > 0.5$, we get $F_b = 1$ and $B_b = 0$ (see lines 13 through 19 of Algorithm 2). Hence, $W = 2$ for codeword '0011' and $W = 0$ for all other unassigned codewords in group g_1 (see lines 20 through 32 of Algorithm 2). The unassigned codewords in group g_1 are arranged in ascending order of W (see line 33 of Algorithm 2). The first codeword in group g_1 (after arranging in ascending order of W), is assigned to the next symbol (2) in Table VI (see line 34 of Algorithm 2). Hence, $c_3 = 1100$. Now, $F = 1$, $B = 0$, $P_f^{old} = 0.29$ and $P_b^{old} = 0.565$. Using Equations 1 and 2, we get $P_f^{new} = 0.388$ and $P_b^{new} = 0.487$.
- Now, since $P_f < 0.5$ and $P_b < 0.5$, we get $F_b = 0$ and $B_b = 0$ (see lines 13 through 19 of Algorithm 2). Hence, $W = 1$ for all unassigned codewords in group g_1 (see lines 20 through 32 of Algorithm 2). Therefore, the first unassigned codeword in group g_1 is assigned to the next symbol (-1) in Table VI. Hence, $c_4 = 1110$. The algorithm ends here as all the symbols to be encoded are assigned codewords.

The final Codebook (LUT) for this example is shown in Table VII.

E. Information Theoretic Lower Bound on Energy and EDP

In [16], information theoretic bounds on the average signal transition activity in a bus, given a source entropy rate $\mathcal{H}$ and an average rate R (bits per symbol) of transmission, have been provided. A lower bound on the *Energy-Delay-Product (EDP)*[3], given $\mathcal{H}$ and R, has also been provided. We have used these bounds to measure the proximity of the proposed scheme to the theoretical lower bound, and compare the result with the corresponding result for [8].

1) Lower Bound on Energy: The lower bound on the average number of transitions (which is proportional to the energy consumed) per symbol is given by [16]

$$T \geq H^{-1}\left(\frac{\mathcal{H}}{R}\right) \cdot R \tag{3}$$

[3]Note that [16] assumes that frequency is constant, and hence energy is proportional to power. Therefore the Power-Delay Product reported in [16] is directly proportional to the EDP metric used in this paper.

where $\mathcal{H}$ is the entropy rate of the source in bits/symbol, R is the rate of transmission in bits/symbol and H^{-1} is the inverse of the binary entropy function $H(\ )$ defined as follows :

$$H^{-1}(y) = x, \text{ if } y = H(x) \text{ and } x \in \left[0, \frac{1}{2}\right] \tag{4}$$

A plot of the binary entropy function is shown in Figure 7. The function $H^{-1}(\ )$ maps the entropy of a binary-valued discrete random variable to a probability value that lies between 0 and 1/2.

2) Lower Bound on EDP: The lower bound on EDP is given by [16]

$$EDP_{min} = KH^{-1}\left(\frac{\mathcal{H}}{R}\right)R^2 \tag{5}$$

where K is a constant of proportionality. A plot of EDP_{min} versus R is shown in Figure 8. In Figure 8, EDP_{min} is represented as a fraction of $K\mathcal{H}^2$ and R is represented in multiples of $\mathcal{H}$. From Figure 8, the value of R that minimizes EDP is found to be [16],

$$R_{min,EDP} = 1.25392\mathcal{H} \tag{6}$$

Hence Equation 6 indicates that a source with entropy rate $\mathcal{H}$ requires approximately an average of $1.25\mathcal{H}$ bits per symbol to encode for minimum EDP. If $R > 1.25\mathcal{H}$, the delay component will increase, reducing the overall rate of transmission. On the other hand, if $R < 1.25\mathcal{H}$, the energy component will increase, since less redundancy is being used.

In the next section, we discuss the results obtained by our approach (MCA) and the 2-LIWT scheme [8], with a comparison of the proximity to the theoretical lower bound of both approaches.

IV. Experimental Results

We implemented the TMDS scheme [9], the 2-LIWT scheme [8] and the MCA in the C programming language. Our code to model these schemes was run on a 1.66 GHz Intel Duo Core machine with 1 GB of RAM. Our implementation of TMDS and 2-LIWT obtained the same results as reported in [9], [8] respectively. In all the experiments, EDP is measured by $l_i \cdot t_i$, where t_i is the number of transitions and l_i is the number of bits (or length) of the transmitted data. Energy is measured by the number of transitions t_i of the transmitted data.

The average probability distribution of pixel differences or symbols (which is required for the MCA) is calculated using 10 different images (shown in the top rows of Tables VIII, IX, X and XI) in the SIPI database [14]. The MCA was applied to these 10 images in the SIPI database [14] as well as 8 other images (shown in the bottom rows of Tables VIII, IX, X and XI) from [15] (which are not included in calculating the average probability distribution). In other words, the codebooks are constructed once using data from the 10 images. The same codebooks are used for other images (in general, these could be any arbitrary image being transmitted). The MCA (with EDP minimization and energy minimization) was tested with various values of Δ, k, n and p. We used $m = 8$ since the data in the SIPI database [14], as well as the data in [15] use 8 bits per pixel value. The parameter k was varied from 3 to 6, n from 3 to 10 and p from 3 to 14. This was done for six values of range Δ = 10, 20, 30, 40, 50 and 255. The range Δ = 255 implies that *all* the pixel-differences are encoded.

Results were obtained for all the 3 channels (RGB) using the TMDS standard. The performance of the MCA was analyzed in terms of the percentage savings of EDP and energy over TMDS. The results of the MCA are also compared with our implementation of the 2-LIWT encoding scheme [8].

In a serial data transmission scenario, the transfer delay of any image is proportional to the total length of the serial data stream. Also, the power required for transferring any image is proportional to the total number of transitions in the serial data stream.

Table VIII shows the total length of the serial data stream, the total number of transitions and the number of intra-vector transitions of the TMDS scheme, for the 10 different images from the SIPI database and 8 different images from [15], for the Blue, Green and Red channels. Tables IX, X and XI show the total length of the serial data stream, the total number of transitions, the percentage total EDP savings and the percentage total energy savings (compared to the TMDS scheme) of the 2-LIWT scheme [8], the MCA with EDP minimization and the MCA with energy minimization respectively. Note that the 2-LIWT scheme does not include DC balancing (i.e. Unbalanced 2-LIWT [8]). The results shown in Table X correspond to the MCA scheme optimized for EDP minimization (with parameters $k = 4$, $n = 4$, $p = 6$ and $\Delta = 40$ corresponding to our best EDP minimization code[4]). From Tables IX and X, we see that on an average, the MCA optimized for EDP minimization yields a 9.6%, 9.79% and 8.47% improvement in the total EDP over the 2-LIWT encoding scheme [8] for the Blue, Green and Red channels respectively. Since the MCA is optimized for reducing EDP in X, the energy savings are less than that of the 2-LIWT encoding scheme [8] by 3.74%, 3.48% and 3.7% for the Blue, Green and Red channels respectively. However, as discussed in Section II, the EDP is a more relevant metric for comparing the serial bus encoding schemes. We also computed average EDP (energy) savings achieved by the MCA optimized for EDP minimization and (from Table X), averaged over 8 images from [15] (shown in the bottom rows of Table X). For these 8 images, the MCA optimized for EDP minimization yields a 83.78% (73.45%), 83.32% (72.72%) and 87.4% (79.48%) reduction in the total EDP (energy) over TMDS for the Blue, Green and Red channels respectively. The corresponding average (averaged over the 8 images from [15], shown in the bottom rows of Table IX) total EDP (energy) savings of the 2-LIWT encoding scheme [8] are 75.91% (75.86%), 74.92% (74.87%) and 82.21% (82.16%) over TMDS for the Blue, Green and Red channel respectively.

The results shown in Table XI correspond to the MCA scheme optimized for energy minimization (with parameters $k = 6$, $n = 10$, $p = 14$ and $\Delta = 255$ corresponding to our best energy minimization code). From Tables IX and XI, we see that on an average the MCA optimized for energy minimization yields a 2.31%, 3% and 2.33% improvement in the total EDP and 1.57%, 2.08% and 1.68% improvement in the total energy over the 2-LIWT encoding scheme [8] for the Blue, Green and Red channels respectively. This shows that the MCA also provides improvement in energy (along with improvement in EDP) compared to the 2-LIWT encoding scheme [8]. Note that, on an average over the 8 images from [15] (shown in the bottom rows of Table XI), the MCA scheme optimized for energy minimization achieves a 78.04% (75.82%), 77.51% (75.21%) and 84.14% (82.33%) reduction in the total EDP (energy) over TMDS for the Blue, Green and Red channels respectively.

The sets of k, n and p values which yield the four highest EDP savings (on an average for the 10 images in the SIPI

[4]From Table XII, it should be noted that the code with parameters $k = 4$, $n = 4$, $p = 6$ and $\Delta = 40$ yields almost the same savings in EDP as the code with parameters $k = 3$, $n = 4$, $p = 6$ and $\Delta = 40$. However, the former code was chosen as the best code since it yields better energy savings than the latter.

database [14] for the MCA scheme optimized for EDP minimization) were selected from each range Δ, and are displayed in Table XII, along with their averaged total and intra-vector percentage EDP savings. This is shown for all 3 color channels. The corresponding averaged total and intra-vector percentage energy savings for each value of Δ, k, n and p have also been provided. Table XIII shows similar results for the four highest energy savings of the MCA scheme optimized for energy minimization. Column 5 of Tables XII and XIII describes the *Code Size Ratio (CSR)*, defined as the ratio of the size of our encoder/decoder Look-Up Table (LUT) compared to that of the 2-LIWT scheme [8].

The encoder and decoder for the MCA can be realized using a Look-Up Table (LUT), as was done in the 2-LIWT scheme [8]. The size of the LUT for our scheme depends on the parameters p and Δ. For example, one of our best codes with parameters $k = 4$, $n = 4$, $p = 4$ and $\Delta = 20$, has a LUT size of 246 bits (the LUT size for our scheme is $(2\Delta + 1)(p + 2)$). The size of the LUT for the 2-LIWT scheme [8] is 414 bits (46 encoded symbols x 9 bits). In this case our LUT size is 59.42% that of the 2-LIWT scheme [8] (i.e. the CSR is 59.42%). The averaged total EDP savings (averaged over the 10 images in the SIPI database [14]) of this code (optimized for EDP minimization) are 70.87%, 71.17% and 71.54% over TMDS for the Blue, Green and Red channels respectively. The corresponding average total EDP savings (averaged over the 10 images in the SIPI database [14]) of the 2-LIWT scheme [8] are 61.04%, 61.5% and 61.55% over TMDS for the Blue, Green and Red channels respectively. Hence, this code has 59.42% the LUT size of the 2-LIWT scheme [8] and still provides 9.83%, 9.67% and 9.99% more EDP savings than the 2-LIWT scheme [8] for the Blue, Green and Red channels respectively. Several other codes in the MCA scheme have similar performance.

Our scheme also has codes (with parameters $k = 3$, $n = 3$, $p = 3$ and $\Delta = 10$) with an LUT size of 105 bits. In this case our LUT size is 25.36% that of the 2-LIWT scheme [8].

Note that as Δ increases, the CSR increases as well. Based on the results in Table XII, we observe that several of our codes improve on the 2-LIWT scheme [8] in terms of EDP as well as CSR. Note that a CSR improvement would be accompanied by an energy reduction as well, due to the reduced power consumption of the smaller LUT of the MCA. The choice of $\Delta = 255$ is to be avoided since it results in large LUTs. Note that the best code for energy minimization uses $\Delta = 255$. However, there are other energy minimizing configurations (as shown in Table XIII) such as $k = 6$, $n = 10$, $p = 14$ and $\Delta = 50$, which are close in performance to the best energy minimizing code with $\Delta = 255$, but have a much smaller CSR.

The optimal configuration (i.e. the best k, n, p and Δ) for a certain application can be obtained by selecting a configuration from Table XII and Table XIII (based on whether EDP or Energy minimization is the primary design goal). The CSR of the code can also be considered if design cost is an important factor.

As mentioned in Section III-E, the information theoretic lower bound on the achievable transitions per symbol (given the average rate R in bits/symbol and the source entropy rate $\mathcal{H}$) was reported in [16]. Hence, no encoding scheme (operating with a rate of R bits/symbol and source entropy rate $\mathcal{H}$), can achieve lesser number of transitions per symbol (T) than the lower bound given by Equation 3. An encoding scheme would be able to achieve a number of transitions per symbol as close to the lower bound as possible. Hence, the proximity of the MCA and the 2-LIWT schemes to their respective lower bounds would be a good measure to compare the efficiency of both these schemes. In order to compute the lower bound, the entropy rate $\mathcal{H}(\mathcal{X})$ (given by Equation 9 in Appendix) is required. Since it is practically difficult to determine the entropy rate $\mathcal{H}(\mathcal{X})$,

it can be approximated using the entropy $H(X)$ as given by Equation 12 (see Appendix). Note that the pixel differences can be assumed to be independent and identically distributed (which is required for this approximation). For the images from the SIPI database (which were used for the experiments), the entropy $H(X) = 5$ bits per symbol. The average rate of transmission (R) of the best code of the MCA optimized for EDP minimization (i.e. with parameters $k = 4$, $n = 4$, $p = 6$ and $\Delta = 40$) is 6.2 bits per symbol (which is $1.24H$). At this rate R and entropy $H(X)$, the lower bound on the number of transitions (from Equation 3) is 1.56 transitions per symbol, whereas the best code of the MCA achieves 1.73 transitions per symbol on an average (calculated from experimental results). This indicates that the average number of transitions of the MCA is 10.9% above the lower bound. The average rate of transmission (R) of the 2-LIWT scheme is 9.92 bits per symbol (which is $1.984H$). At this rate R and entropy $H(X)$, the lower bound on the number of transitions (from Equation 3) is 1.09 transitions per symbol, whereas the 2-LIWT scheme achieves 1.5 transitions per symbol on an average (calculated from experimental results). This implies that the average number of transitions of the 2-LIWT scheme is 37.6% above the lower bound (in contrast to the MCA which is 10.9% above the lower bound). This explains the energy savings provided by the MCA compared to the 2-LIWT scheme.

In Section III-E.2, it was explained that for achieving the minimum lower bound on EDP, the rate of transmission should be approximately $1.25H$ bits per symbol on average. The rate of transmission (R) of the MCA (which is $1.24H$) is more closer to $1.25H$ compared to the rate of transmission of the 2-LIWT scheme (which is $1.984H$). Hence the MCA scheme is able to yield better EDP savings compared to the 2-LIWT scheme.

Based on these results, we note that the MCA technique provides impressive improvements in EDP over that of [8], which is the best energy reducing serial data encoding technique available.

V. Conclusions

In serial data transmission, it is crucial to reduce the energy associated with the data transfer, as well as the total time required for transmission. As a result, the Energy-Delay-Product (EDP) is the important metric to be minimized. Previous encoding approaches in this area have targeted energy minimization alone. In this paper, we report an encoding based EDP minimization technique for serial data transmission. Our approach utilizes 3 codebooks of varying lengths to achieve this. In comparison to existing low energy serial data encoding approaches, our method yields a reduction in EDP by 10% (averaged over several serial data transmission benchmarks), with a reduced encoding/decoding complexity as well as energy, compared to the best known prior approach [8] Also, the average number of transitions of our scheme is 10.9% above the theoretical lower bound (in contrast to the best approach known for serial data transmission [8], which is 37.6% above the theoretical lower bound).

Appendix

Entropy Rate : In order to get an insight on the lower bounds discussed in Section III-E, it is essential to understand the concept of *entropy rate* [22]. To understand the *entropy rate* of a source, we need to know the definition of *entropy* and *joint entropy*.

The *entropy* of a discrete random variable X with probabilities p_i is defined as

$$H(X) = -\sum_{i} \left(p_i \log_2 p_i\right) \tag{7}$$

The *joint entropy* of discrete random variables X_1, X_2,...,X_N is given by

$$H(X_1, X_2, ..., X_N) = -\sum_{x_1,\ldots,x_N} \left(p(x_1, ..., x_N) \log_2 p(x_1, ..., x_N)\right) \tag{8}$$

where $p(x_1, ..., x_N)$ is the joint probability distribution of random variables X_1, X_2,...,X_N.

Now, the *entropy rate* of a stochastic process $\{X_i\}$ is defined by

$$\mathcal{H}(\mathcal{X}) = \lim_{N\to\infty} \frac{1}{N} H(X_1, X_2, ..., X_N) \tag{9}$$

The joint entropy $H(X_1, X_2, ..., X_N)$ can be upper-bounded by the entropy of each random variable X_1, X_2,...,X_N as

$$H(X_1, X_2, ..., X_N) \leq \sum_{i=1}^{N} H(X_i) \tag{10}$$

with equality if and only if the random variables X_1, X_2,...,X_N are independent. If the random variables X_i are identically distributed, then they all have the same entropy $H(X_i) = H(X)$. Then Equation 10 becomes

$$H(X_1, X_2, ..., X_N) \leq NH(X) \tag{11}$$

Hence, if a stochastic process $\{X_i\}$ is identically distributed at each time instant i, then its entropy rate $\mathcal{H}(\mathcal{X})$ can be upper-bounded as

$$\mathcal{H}(\mathcal{X}) \leq H(X) \tag{12}$$

where $H(X)$ is the entropy of the outcome of the source at each time instant.

REFERENCES

[1] "The International Technology Roadmap for Semiconductors (ITRS)," **2008**. [Online]. Available: http://public.itrs.net.

[2] M. Stan and W. Burleson, "Bus-invert coding for low-power I/O," *IEEE Transactions on VLSI Systems*, vol. 3, pp. 49–58, March **1995**.

[3] L. Benini, G. D. Micheli, E. Macii, D. Sciuto, and C. Silvano, "Address bus encoding techniques for system-level power optimization," in *Proc. of the Conf. on Design Automation and Test in Europe*, February **1998**, pp. 861–867.

[4] L. Benini, G. D. Micheli, E. Macii, M. Poncino, and S. Quer, "Power Optimization of Core-Based Systems by Address Bus Encoding," *IEEE Transactions on VLSI Systems*, vol. 6, pp. 554–562, December **1998**.

[5] L. Benini, A. Macii, E. Macii, M. Poncino, and R. Scarsi, "Architectures and Synthesis Algorithms for Power-Efficient Bus Interfaces," *IEEE Transactions on CAD*, vol. 19, pp. 969–980, September **2000**.

[6] Y. Aghaghiri, F. Fallah, and M. Pedram, "Irredundant address bus encoding for low power," in *Proc. of the International Symposium on Low Power Electronics and Design*, August **2001**, pp. 182–187.

[7] W. C. Cheng and M. Pedram, "Chromatic Encoding: a Low Power Encoding Technique for Digital Visual Interface," *IEEE Transactions on Consumer Electronics*, vol. 50, pp. 320–328, February **2004**.

[8] S. Salerno, E. Macii, and M. Poncino, "Energy-Efficient Bus Encoding for LCD Digital Display Interfaces," *IEEE Transactions on Consumer Electronics*, vol. 51, pp. 624–634, May **2005**.

[9] "DDWG, Digital Visual Interface, V1.0." [Online]. Available: www.ddwg.org.

[10] "Open LVDS Display Interface (OpenLDI) Specification, Version 0.9, National Semiconductor," February **1999**.

[11] H. Kikuchi, T. Fukuzaki, R. Tamaki, and T. Takeshita, "Gigabit Video Interface: A Fully Serialized Data Transmission System for Digital Moving Pictures," in *ICCE'98*, **1998**, pp. 30–32.

[12] A. Bocca, S. Salerno, E. Macii, and M. Poncino, "Energy-Efficient Bus Encoding for LCD Displays," in *Proc. of the Great Lakes symposium on VLSI*, April **2004**, pp. 240–243.

[13] S. Salerno, A. Bocca, E. Macii, and M. Poncino, "Limited Intra-Word Transition Codes: An Energy-Efficient Bus Encoding for LCD Digital Display Interfaces," in *Proc. of the International Symposium on Low Power Electronics and Design*, August **2004**, pp. 206–211.

[14] A. G. Weber, "USC-SIPI Images Database Version 5." USC-SIPI Report No. 315, October **1997**. [Online]. Available: http://sipi.usc.edu/services/database/Database.html.

[15] K. Barnard, L. Martin, B. V. Funt, and A. Coath, "A Data Set for Color Research," *Color Research and Application*, vol. 27, no. 3, pp. 147–151, **2002**. [Online]. Available: http://www.cs.sfu.ca/∼colour/data/objects_under_different_lights/index.html.

[16] S. Ramprasad, N. R. Shanbhag, and I. N. Hajj, "Information-Theoretic Bounds on Average Signal Transition Activity," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, pp. 359–368, September **1999**.

[17] K. Lee, S.-J. Lee, and H.-J. Yoo, "SILENT: Serialized Low Energy Transmission Coding for On-Chip Interconnection Networks," in *Proc. of the Intl. Conf. on Computer-Aided Design*, November **2004**, pp. 448–451.

[18] I. Choi, H. Shim, and N. Chang, "Low-Power Color TFT LCD Display for Handheld Embedded Systems," in *Proc. of the International Symposium on Low Power Electronics and Design*, **2002** pp. 112–117.

[19] F. Gatti, A. Acquaviva, L. Benini, and B. Ricco, "Low Power Control Techniques For TFT LCD Displays," in *CASES* **2002**, pp. 218–224.

[20] K. M. et al., "A Low-Power High-Voltage Column/Dot Inversion Drive System," in *Society for Information Display International Symposium Digest of Technology*, vol. 28, May **1997**.

[21] B. D. Choi and O. K. Kwon, "Stepwise Data Driving Method and Circuits for Low-Power TFT-LCDs," *IEEE Transactions on Consumer Electronics*, vol. 46, no. 11, pp. 1155–1160, **2000**.

[22] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley Series, **2006**.

FIGURES AND TABLES

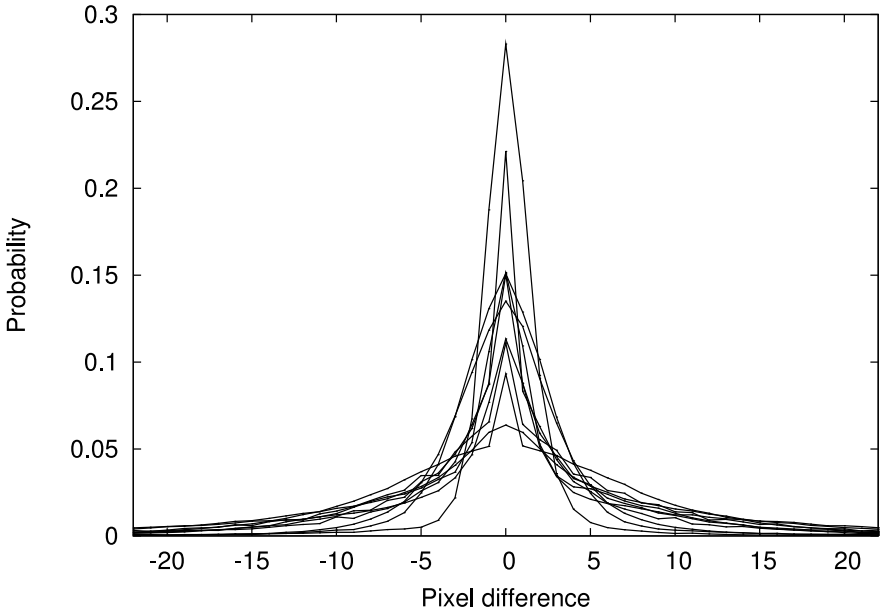

Fig. 1. Distribution of pixel difference for Blue component of 10 images in the SIPI database

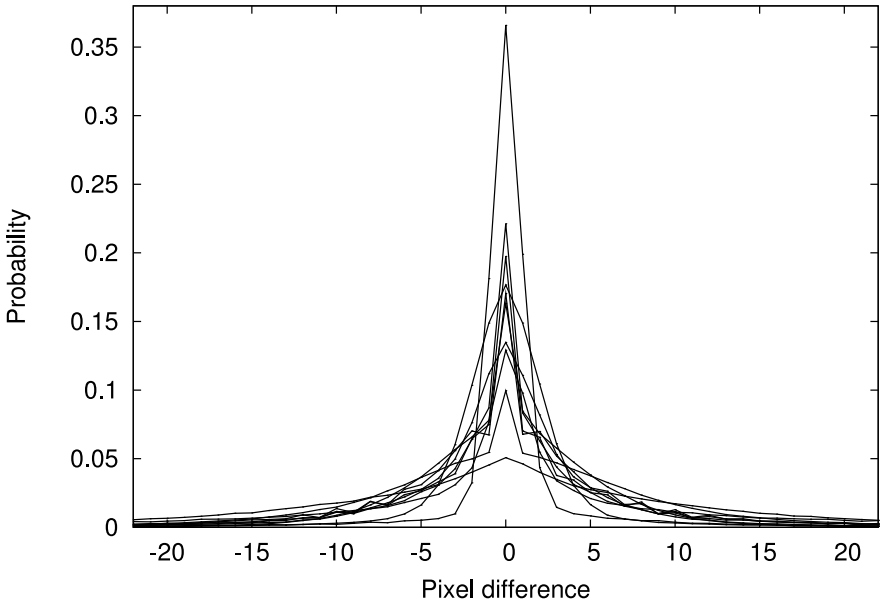

Fig. 2. Distribution of pixel difference for Green component of 10 images in the SIPI database

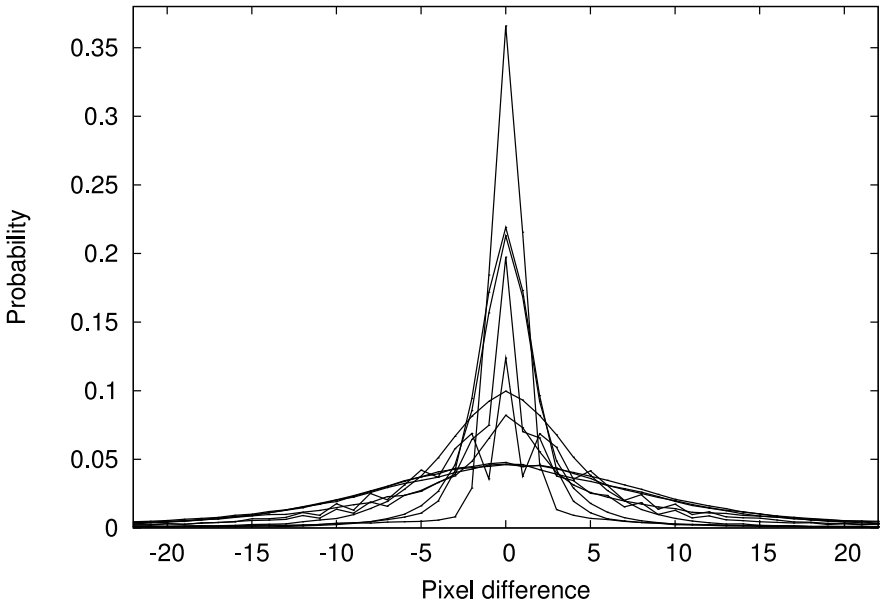

Fig. 3. Distribution of pixel difference for Red component of 10 images in the SIPI database

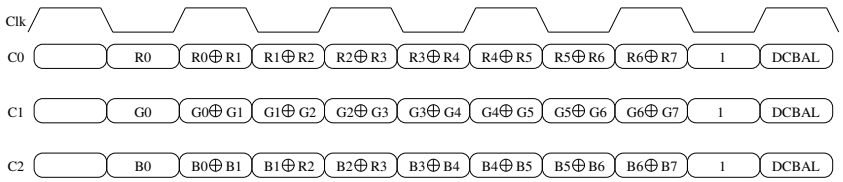

Fig. 4. 24-bit, XOR-based, Single-Link TMDS

Fig. 5. Codeword with preamble bits

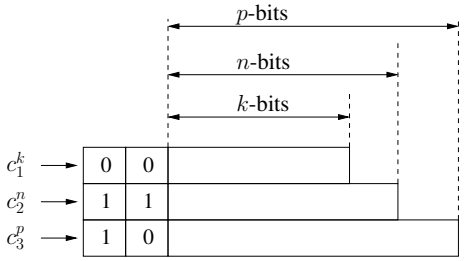

Fig. 6. An example of the combined codebook, shown for illustration

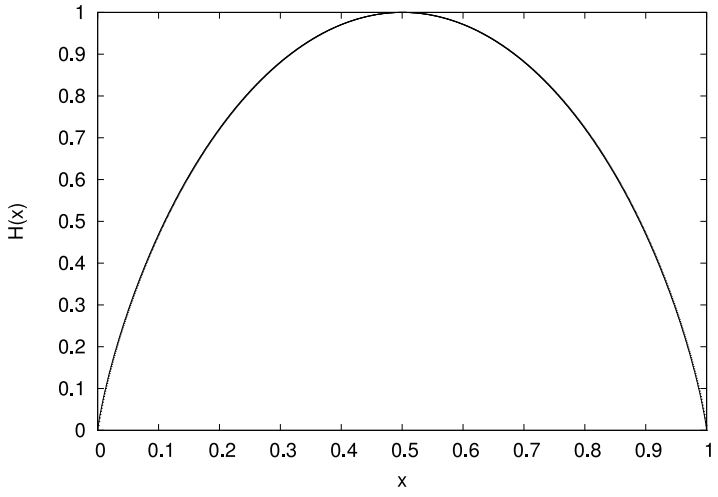

Fig. 7. Binary entropy function

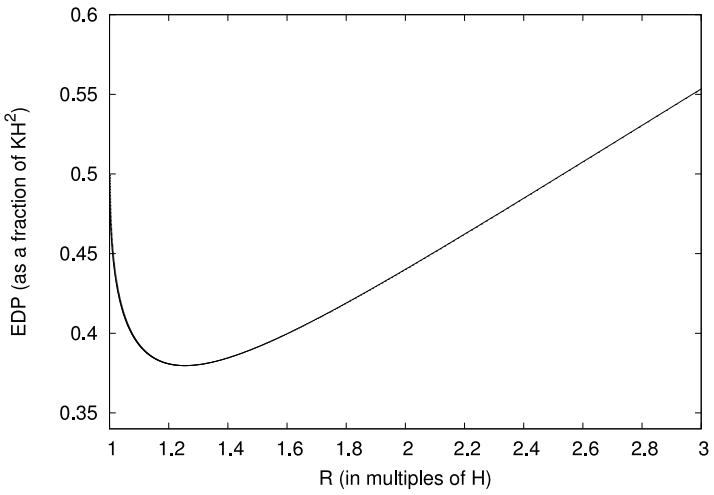

Fig. 8. Lower bound on EDP versus R for a given $\mathcal{H}$

$pre1$	$pre2$	$Description$
0	0	k bit codeword
0	1	not encoded
1	0	p bit codeword
1	1	n bit codeword

TABLE I
PREAMBLE BITS DESCRIPTION

$Preamble$	$Codeword$	$Energy$	EDP
00	00	0	0
00	01	1	4
00	10	2	8
00	11	1	4

TABLE II
k CODEBOOK

$Preamble$	$Codeword$	$Energy$	EDP
11	00	1	4
11	01	2	8
11	10	1	4
11	11	0	0

TABLE III
n CODEBOOK

$Preamble$	$Codeword$	$Energy$	EDP
10	00	1	4
10	01	2	8
10	10	3	12
10	11	2	8

TABLE IV
p CODEBOOK

$Preamble$	$Codeword$	$Energy$	EDP	$Group$
00	00	0	0	g_0
11	11	0	0	g_0
00	01	1	4	g_1
00	11	1	4	g_1
11	00	1	4	g_1
11	10	1	4	g_1
10	00	1	4	g_1
00	10	2	8	g_2
11	01	2	8	g_2
10	01	2	8	g_2
10	11	2	8	g_2
10	10	3	12	g_3

TABLE V
ALL CODEWORDS FROM THE 3 CODEBOOKS ARRANGED IN ASCENDING ORDER OF ENERGY OR EDP

$Pixel\ Difference$	$Probability$
0	0.30
1	0.20
-1	0.19
2	0.11
-2	0.10

TABLE VI

SYMBOLS (OR PIXEL DIFFERENCES TO BE ENCODED) ARRANGED IN DESCENDING ORDER OF THEIR PROBABILITY

$Pixel\ Difference$	$Codeword$
0	0000
1	1111
-1	0001
2	1100
-2	1110

TABLE VII

FINAL CODEBOOK (OR LUT) FOR EXAMPLE IN SECTION III-D

	BLUE			GREEN			RED		
		Transitions			Transitions			Transitions	
Image	Length	Total	Intra	Length	Total	Intra	Length	Total	Intra
Couple	655360	258656	220690	655360	257117	218023	655360	260334	224226
Girl1	655360	264318	228752	655360	263925	228523	655360	270067	236365
Girl2	655360	269843	236654	655360	257981	224497	655360	262778	227992
House	655360	261742	228672	655360	275750	244128	655360	278504	244430
Jellyb	655360	278280	245268	655360	279969	250543	655360	243155	220228
Peppers	2621440	1062423	925284	2621440	1067678	931205	2621440	1078916	947837
Sail	2621440	1118204	990100	2621440	1100844	971888	2621440	1092560	957862
Splash	2621440	1066812	924644	2621440	1049917	905722	2621440	1121516	989212
Tree	655360	275485	241244	655360	273029	240927	655360	277041	243885
Moon	655360	253386	218358	655360	253386	218358	655360	253386	218358
Ball1	2981160	646997	386798	2981160	748148	492518	2981160	636371	377334
Blocks	2981160	834241	596816	2981160	862981	624816	2981160	758699	518028
Book3	2981160	851806	621944	2981160	922390	702752	2981160	855933	635346
Book4	2981160	879465	659944	2981160	947557	733974	2981160	870307	653732
Hairgel	2981160	793860	545564	2981160	768246	518502	2981160	661108	406531
Rollups	2981160	902990	676428	2981160	887988	656489	2981160	787224	553092
Shirt2	2981160	1103261	919332	2981160	1134788	955624	2981160	1099010	917191
Shorts	2981160	833163	590456	2981160	819982	576102	2981160	808228	571922

TABLE VIII

EXPERIMENTAL RESULTS OF TMDS

	Blue				Green				Red			
		Transitions	% Total Savings			Transitions	% Total Savings			Transitions	% Total Savings	
Image	Length	Total	Energy	EDP	Length	Total	Energy	EDP	Length	Total	Energy	EDP
Couple	652183	100389	61.19	61.38	652288	91639	64.36	64.53	651656	99107	61.93	62.15
Girl1	651943	110425	58.22	58.44	652581	97609	63.02	63.17	652629	107755	60.10	60.27
Girl2	654280	79861	70.40	70.45	654291	76659	70.29	70.33	654123	74429	71.68	71.73
House	652720	102756	60.74	60.90	652547	96969	64.83	64.99	653491	96594	65.32	65.42
Jellyb	653955	69851	74.90	74.95	652698	71707	74.39	74.49	653368	72687	70.11	70.20
Peppers	2609635	452760	57.38	57.58	2606393	454210	57.46	57.70	2610455	485460	55.00	55.19
Sail	2590952	520961	53.41	53.95	2585174	562823	48.87	49.58	2602903	516450	52.73	53.06
Splash	2616861	339549	68.17	68.23	2614996	365208	65.22	65.30	2617033	302743	73.01	73.05
Tree	647364	128183	53.47	54.04	646262	126202	53.78	54.42	648424	128703	53.54	54.04
Moon	651300	126339	50.14	50.45	651300	126339	50.14	50.45	651300	126339	50.14	50.45
Ball1	2981118	60954	90.58	90.58	2981044	132224	82.33	82.33	2980885	21650	96.60	96.60
Blocks	2981158	183472	78.01	78.01	2980105	172902	79.96	79.97	2979552	100330	86.78	86.78
Book3	2977828	223158	73.80	73.83	2977180	231208	74.93	74.97	2977288	170054	80.13	80.16
Book4	2977186	208888	76.25	76.28	2975696	251698	73.44	73.49	2975191	182900	78.98	79.03
Hairgel	2978956	175702	77.87	77.88	2978195	172422	77.56	77.58	2978240	54768	91.72	91.72
Rollups	2972706	234388	74.04	74.12	2970804	228038	74.32	74.41	2971102	146518	81.39	81.45
Shirt2	2966695	447710	59.42	59.62	2965987	445840	60.71	60.91	2966864	423078	61.50	61.69
Shorts	2981096	192202	76.93	76.93	2980637	199322	75.69	75.70	2980173	160030	80.20	80.21
Avg			67.50	67.65			67.29	67.46			70.60	70.73

TABLE IX

EXPERIMENTAL RESULTS OF 2-LIWT (RANGE : PIXEL DIFFERCE $\in$ [-23, 22])

	Blue				Green				Red			
		Transitions	% Total Savings			Transitions	% Total Savings			Transitions	% Total Savings	
Image	Length	Total	Energy	EDP	Length	Total	Energy	EDP	Length	Total	Energy	EDP
Couple	413656	115049	55.52	71.93	411812	105461	58.98	74.23	414508	113354	56.46	72.46
Girl1	416484	126774	52.04	69.52	411414	110964	57.96	73.61	415120	123370	54.32	71.06
Girl2	401450	90162	66.59	79.53	400286	83196	67.75	80.30	400386	82272	68.69	80.87
House	412618	112735	56.93	72.88	409872	106122	61.52	75.93	406788	106740	61.67	76.21
Jellyb	399358	73456	73.60	83.91	404254	76313	72.74	83.19	401620	74908	69.19	81.12
Peppers	1668322	514479	51.57	69.18	1673664	512900	51.96	69.33	1678204	556783	48.39	66.96
Sail	1728534	595928	46.71	64.86	1749998	638870	41.97	61.26	1696462	582549	46.68	65.49
Splash	1611254	380556	64.33	78.07	1627616	405848	61.34	76.00	1599482	332668	70.34	81.90
Tree	430238	141150	48.76	66.36	432696	140346	48.60	66.06	428318	142370	48.61	66.41
Moon	421338	142048	43.94	63.96	421338	142048	43.94	63.96	421338	142048	43.94	63.96
Ball1	1789552	68724	89.38	93.62	1789774	133961	82.09	89.25	1790100	22506	96.46	97.88
Blocks	1790420	189400	77.30	86.36	1793782	177966	79.38	87.59	1796634	114451	84.91	90.91
Book3	1807274	245426	71.19	82.53	1810748	248277	73.08	83.65	1813318	192398	77.52	86.33
Book4	1814348	240016	72.71	83.39	1819076	278791	70.58	82.05	1820804	211078	75.75	85.19
Hairgel	1799326	184198	76.80	86.00	1800928	180124	76.55	85.84	1800670	65244	90.13	94.04
Rollups	1823894	256378	71.61	82.63	1829442	252025	71.62	82.58	1829436	171248	78.25	86.65
Shirt2	1881620	518805	52.98	70.32	1882680	518495	54.31	71.15	1879282	493375	55.11	71.70
Shorts	1793356	202832	75.66	85.36	1797032	211844	74.16	84.43	1800736	180322	77.69	86.52
Avg			63.76	77.25			63.81	77.25			66.90	79.20

TABLE X

EXPERIMENTAL RESULTS OF MCA WITH EDP MINIMIZATION FOR $k = 4$, $n = 4$, $p = 6$ AND $\Delta = 40$

	Blue				Green				Red			
		Transitions	% Total Savings			Transitions	% Total Savings			Transitions	% Total Savings	
Image	Length	Total	Energy	EDP	Length	Total	Energy	EDP	Length	Total	Energy	EDP
Couple	653856	95667	63.01	63.10	637240	87999	65.77	66.72	644956	92590	64.43	65.00
Girl1	661404	102962	61.05	60.69	643656	92345	65.01	65.64	655408	101385	62.46	62.46
Girl2	650608	82663	69.37	69.59	636784	76001	70.54	71.38	639824	75427	71.30	71.98
House	650588	93859	64.14	64.40	651892	90026	67.35	67.53	662424	92143	66.92	66.56
Jellyb	633176	68095	75.53	76.36	619996	66860	76.12	77.41	620944	66693	72.57	74.01
Peppers	2642688	416884	60.76	60.44	2640696	406819	61.90	61.62	2659692	447854	58.49	57.88
Sail	2684528	459200	58.93	57.95	2694580	478630	56.52	55.31	2675844	458963	57.99	57.12
Splash	2615744	342527	67.89	67.96	2595224	346182	67.03	67.36	2572356	306442	72.68	73.19
Tree	666780	108563	60.59	59.91	667956	105772	61.26	60.52	671004	110501	60.11	59.16
Moon	668952	112689	55.53	54.60	668952	112689	55.53	54.60	668952	112689	55.53	54.60
Ball1	2466048	67710	89.53	91.34	2552916	133139	82.20	84.76	2423444	21188	96.67	97.29
Blocks	2622940	186906	77.60	80.29	2616428	172844	79.97	82.42	2538864	107295	85.86	87.96
Book3	2696208	226276	73.44	75.97	2692724	226885	75.40	77.78	2624132	168690	80.29	82.65
Book4	2679196	216518	75.38	77.87	2717464	251107	73.50	75.84	2639140	182058	79.08	81.48
Hairgel	2617660	174882	77.97	80.66	2608448	169972	77.88	80.64	2466684	54528	91.75	93.18
Rollups	2683328	226128	74.96	77.46	2669524	218367	75.41	77.98	2575532	137738	82.50	84.88
Shirt2	2939828	426902	61.31	61.84	2931492	429472	62.15	62.78	2918676	402976	63.33	64.10
Shorts	2658120	197202	76.33	78.90	2653044	203552	75.18	77.91	2631260	168420	79.16	81.61
Avg			69.07	69.96			69.37	70.46			72.28	73.06

TABLE XI

EXPERIMENTAL RESULTS OF MCA WITH ENERGY MINIMIZATION FOR $k = 6$, $n = 10$, $p = 14$ AND $\Delta = 255$

					Blue				Green				Red			
					% Energy Savings		% EDP Savings		% Energy Savings		% EDP Savings		% Energy Savings		% EDP Savings	
Δ	k	n	p	CSR	Total	Intra	Total	Intra	Total	Intra	Total	Intra	Total	Intra	Total	Intra
10	3	3	3	0.2536	43.45	66.32	65.99	79.64	45.24	67.82	67.06	80.52	44.34	66.93	66.60	80.03
10	2	3	4	0.3043	41.62	67.37	65.61	80.58	43.47	68.53	66.82	81.33	42.66	67.95	66.23	80.89
10	3	3	4	0.3043	43.72	66.70	65.60	79.53	45.49	68.15	66.69	80.42	44.61	67.29	66.19	79.90
10	3	3	5	0.3551	43.94	66.96	65.03	79.28	45.74	68.43	66.20	80.20	44.83	67.62	65.60	79.67
20	3	4	5	0.6932	51.88	72.92	71.02	83.64	52.53	73.28	71.35	83.81	52.76	73.74	71.70	84.20
20	3	3	5	0.6932	50.05	72.55	70.93	83.97	50.77	73.24	71.25	84.31	50.75	73.46	71.43	84.55
20	4	4	4	0.5942	53.72	72.49	70.87	82.66	54.44	72.80	71.17	82.77	54.56	73.22	71.54	83.21
20	2	4	5	0.6932	50.35	73.14	70.72	84.06	51.08	73.49	71.18	84.28	51.27	73.98	71.39	84.59
30	4	4	5	1.0314	55.23	73.87	71.95	83.61	55.86	74.14	72.24	83.72	56.06	74.66	72.60	84.18
30	3	4	6	1.1787	53.50	74.16	71.78	84.27	54.07	74.46	72.10	84.43	54.36	75.00	72.43	84.84
30	3	5	5	1.0314	55.23	74.42	71.73	83.80	55.70	74.69	72.05	83.98	56.09	75.26	72.43	84.41
30	4	4	6	1.1787	55.65	74.08	71.67	83.42	56.28	74.34	71.98	83.53	56.49	74.87	72.31	83.98
40	3	4	6	1.5652	53.71	74.45	72.04	84.52	54.30	74.78	72.40	84.72	54.56	75.30	72.68	85.09
40	4	4	6	1.5652	56.00	74.47	72.02	83.75	56.68	74.78	72.39	83.91	56.83	75.26	72.65	84.30
40	3	5	6	1.5652	55.94	75.07	71.91	84.06	56.49	75.41	72.31	84.30	56.81	75.93	72.58	84.66
40	4	4	7	1.7609	55.87	73.85	71.62	83.16	56.65	74.25	72.04	83.37	56.67	74.62	72.24	83.71
50	3	5	6	1.9517	55.95	75.10	71.98	84.12	56.50	75.45	72.41	84.37	56.82	75.98	72.65	84.72
50	4	4	7	2.1957	56.07	74.11	71.79	83.35	56.90	74.56	72.24	83.59	56.86	74.86	72.39	83.88
50	3	5	7	2.1957	56.46	75.49	71.74	84.03	57.05	75.86	72.20	84.31	57.29	76.32	72.39	84.61
50	3	4	7	2.1957	54.13	74.50	71.66	84.19	54.81	74.91	72.10	84.45	54.99	75.39	72.29	84.77
255	3	5	9	13.5773	56.12	74.51	71.09	83.14	56.90	75.06	71.60	83.49	56.85	75.33	71.72	83.73
255	4	5	9	13.5773	57.60	74.46	71.07	82.54	58.26	74.97	71.43	82.83	58.25	75.19	71.64	83.10
255	3	6	9	13.5773	58.24	75.70	70.72	82.89	58.83	76.04	71.27	83.20	58.89	76.42	71.33	83.45
255	4	6	9	13.5773	59.20	75.43	70.57	82.24	59.78	75.81	71.02	82.53	59.89	76.17	71.14	82.80

TABLE XII

EXPERIMENTAL RESULTS OF MCA WITH EDP MINIMIZATION FOR VARIOUS VALUES OF Δ, k, n AND p AVERAGED OVER THE 10 IMAGES IN THE SIPI DATABASE

					Blue				Green				Red			
					% Energy Savings		% EDP Savings		% Energy Savings		% EDP Savings		% Energy Savings		% EDP Savings	
Δ	k	n	p	CSR	Total	Intra	Total	Intra	Total	Intra	Total	Intra	Total	Intra	Total	Intra
10	6	10	10	0.6087	49.25	66.71	49.26	66.69	51.17	67.88	51.66	68.17	50.33	67.39	50.40	67.37
10	6	10	11	0.6594	49.25	66.71	49.19	66.64	51.17	67.88	51.60	68.12	50.33	67.39	50.32	67.32
10	6	10	12	0.7101	49.25	66.71	49.11	66.59	51.17	67.88	51.53	68.08	50.33	67.39	50.25	67.27
10	6	10	13	0.7609	49.25	66.71	49.04	66.54	51.17	67.88	51.47	68.03	50.33	67.39	50.17	67.22
20	6	10	10	1.1884	58.85	73.52	59.47	73.90	59.53	73.85	60.50	74.46	59.78	74.45	60.48	74.85
20	6	10	11	1.2874	58.85	73.52	59.41	73.86	59.53	73.85	60.44	74.42	59.78	74.45	60.42	74.81
20	6	10	12	1.3865	58.85	73.52	59.35	73.82	59.53	73.85	60.39	74.39	59.78	74.45	60.36	74.78
20	6	10	13	1.4855	58.85	73.52	59.30	73.79	59.53	73.85	60.34	74.35	59.78	74.45	60.30	74.74
30	6	10	10	1.7681	61.39	75.26	61.70	75.44	62.03	75.57	62.66	75.95	62.30	76.26	62.68	76.46
30	6	10	11	1.9155	61.39	75.26	61.64	75.40	62.03	75.57	62.61	75.92	62.30	76.26	62.63	76.42
30	6	10	12	2.0628	61.39	75.26	61.59	75.36	62.03	75.57	62.56	75.89	62.30	76.26	62.57	76.39
30	6	10	13	2.2101	61.39	75.26	61.53	75.33	62.03	75.57	62.51	75.85	62.30	76.26	62.52	76.35
40	6	10	10	2.3478	62.48	75.92	62.67	76.03	63.27	76.33	63.76	76.62	63.33	76.91	63.61	77.04
40	6	10	11	2.5435	62.48	75.92	62.61	75.99	63.27	76.33	63.71	76.59	63.33	76.91	63.55	77.01
40	6	10	12	2.7391	62.48	75.92	62.55	75.95	63.27	76.33	63.66	76.55	63.33	76.91	63.49	76.97
40	6	10	13	2.9348	62.48	75.92	62.50	75.91	63.27	76.33	63.61	76.52	63.33	76.91	63.44	76.93
50	6	10	14	3.9034	63.00	76.34	62.87	76.23	63.90	76.83	64.08	76.90	63.81	77.29	63.78	77.22
50	6	10	13	3.6594	62.99	76.32	62.95	76.27	63.89	76.80	64.15	76.93	63.80	77.28	63.85	77.26
50	6	10	12	3.4155	62.99	76.30	63.02	76.30	63.88	76.78	64.22	76.96	63.79	77.26	63.92	77.29
50	6	10	11	3.1715	62.98	76.29	63.09	76.34	63.87	76.77	64.29	77.00	63.78	77.25	63.98	77.33
255	6	10	14	19.7488	63.68	76.79	63.50	76.65	64.70	77.33	64.81	77.35	64.25	77.61	64.20	77.53
255	6	10	13	18.5145	63.67	76.78	63.57	76.69	64.69	77.31	64.87	77.39	64.24	77.60	64.26	77.56
255	6	10	12	17.2802	63.66	76.76	63.64	76.73	64.68	77.28	64.94	77.42	64.23	77.59	64.33	77.60
255	6	10	11	16.0459	63.65	76.75	63.70	76.76	64.66	77.27	65.00	77.45	64.22	77.58	64.39	77.64

TABLE XIII

EXPERIMENTAL RESULTS OF MCA WITH ENERGY MINIMIZATION FOR VARIOUS VALUES OF Δ, k, n AND p AVERAGED OVER THE 10 IMAGES IN THE SIPI DATABASE

BIOGRAPHIES

Sasidharan Ekambavanan obtained his M.S. degree in Electrical and Computer Engineering from Texas A&M University, College Station in 2007. Sasidharan received his B.E. degree in Electronics and Communication from College of Engineering Guindy (Anna University), Chennai, India in 2005. Sasidharan's research work, during his MS, was on techniques to reduce the Energy and Energy-Delay Product in serial data communication, as well as custom VLSI implementaiton of MIMO sphere decoding algorithms. His undergraduate research was focused on Analysis and Design of various MIMO-OFDM schemes using Turbo and LDPC codes as Outer Error-Control Codes. Sasidharan is currently working as a Digital Design Engineer at Link-A-Media Devices, Santa Clara, CA.

Rajesh Garg received his B.Tech degree in Electrical Engineering (Power) from the Indian Institute of Technology-Delhi (IIT-Delhi), India in 2004 and his M.S. and Ph.D. degrees in Computer Engineering from the Texas A&M University, College Station, in 2006 and 2009 respectively. Currently he is working at Intel Corporation, Hillsboro, OR. During his graduate and doctoral studies he has done research and published papers in many aspects of VLSI including radiation tolerant circuit design, process variation tolerant circuit design, circuit modeling, SRAM design, structured ASIC design, logic synthesis, low power design, Viterbi decoder design and statistical timing analysis. During May-August 2006, he worked as a research intern on low power receiver implementation for UWB communication systems at Mitsubishi Electric Research Lab (MERL), Cambridge, MA. He also worked at Intel Corporation, Austin, TX during May-August 2007 as an intern on clock distribution, power gating blocks and IO drivers. He received the President's Silver medal and Best Project in Electrical Engineering award at IIT-Delhi in 2004. Rajesh is an IEEE member.

Sunil P. Khatri received his B.Tech (EE) degree from IIT Kanpur, his M.S.(ECE) degree from the University of Texas, Austin, and the Ph.D. in EECS from the University of California, Berkeley. He worked at Motorola, Inc for four years., where he was a member of the design teams of the MC88110 and PowerPC 603 RISC microprocessors. Sunil is currently an Assistant Professor in ECE at Texas A&M University. His research interests include logic synthesis, novel VLSI design approaches to address issues such as power, cross-talk, hardware acceleration of CAD algorithms and cross-disciplinary applications of these topics. He has coauthored about 100 technical publicatons, 5 United States Patent awards, one book and a book chapter. His work has received three best paper awards and two best paper nominations. Sunil's research is supported by Intel Corporation, Lawrence Livermore National Laboratories, the National Science Foundation, Accelicon, Inc. and Nascentric, Inc.

Krishna R. Narayanan was born in Madurai, India. He received the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, in 1998. He is currently an Associate Professor at the Department of Electrical and Computer Engineering, Texas A&M University, College Station. His research interests are broadly in communication theory with emphasis on coding and signal processing for wireless communications and digital magnetic recording. Dr. Narayanan is currently the area editor for the coding theory and applications area of the IEEE TRANSACTIONS ON COMMUNICATIONS. He has also served on the editorial board of the IEEE transactions on wireless communications and the IEEE communications letters. He was the recipient of the CAREER Award from the National Science Foundation in 2001.