Chromatic Encoding: a Low Power Encoding Technique for Digital Visual Interface

Wei-Chung Cheng, *Member, IEEE,* and Massoud Pedram, *Fellow, IEEE*

Abstract — *This paper presents a low-power encoding technique, called chromatic encoding, for the Digital Visual Interface standard (DVI), a digital serial video interface. Chromatic encoding reduces power consumption by minimizing the transition counts on the DVI. This technique relies on the notion of tonal locality, i.e., the observation - first made in this paper - that the signal differences between adjacent pixels in images follow a Gaussian distribution. Based on this observation, an optimal code assignment is performed to minimize the transition counts. Furthermore, the three color channels of the DVI may be reciprocally encoded to achieve even more power saving. The idea is that given the signal values from the three color channels, one or two of these channels are encoded by reciprocal differences with a number of redundant bits used to indicate the selection. The channel selection problem is formulated as a minimum spanning tree problem and solved accordingly. The proposed technique requires only three redundant bits for each 24-bit pixel. Experimental results show up to a 75% power reduction in the DVI.*

Index Terms — Chromatic encoding, Digital Visual Interface, encoding for low power.

I. Introduction

In mobile computing, power is undoubtedly one of the most important design challenges. To address this issue, fruitful low power techniques have been developed. Memory bus encoding for low power is not only widely-studied in the academia [1-5] but also well-utilized by performance-driven microprocessors [6]. The power reduction is accomplished by minimizing the switching activities on the off-chip memory buses. Cost-effective as it is, bus encoding has not been adopted by the peripherals such as liquid crystal displays (LCD's). The reason is that the power consumed by the peripheral devices eclipses that consumed by the peripheral interfaces, as the latter cannot be reduced by memory bus encoding. Nevertheless, the evolving of display technology changes the paradigm. For example, the organic electroluminescence display (OELD) provides wider viewing angle, higher brightness, and lower supply voltage, while consumes much less power than the conventional backlighted LCD [7]. OELD-equipped electronics, such as cell phones, car audios, and digital cameras, are currently available. More advanced display devices are emerging and keep shrinking their percentage of the entire power budget.

On the other hand, new video interface standards for the next generation display devices are called for replacing the legacy analog VGA standards [8]. The Digital Visual Interface (DVI) 1.0 is advocated by the Digital Display Working Group to provide a display-technology independent, high speed, digital and analog interface [9]. It has been successfully commercialized and is supported by most of today's desktop LCD's if equipped with a digital interface.

A DVI connection consists of one or two Transition Minimized Differential Signaling (TMDS) serial links. The voltage swings of the TMDS differential pairs are 0-780mV and –780-0mV, respectively. The maximum frequency is 1.65Gbps to support 165MHz pixel rate. The length of the DVI cable is limited to 4.6m. The typical mutual capacitance of high quality DVI cables is around 60pF/m [10]. Plugging these numbers in the approximate power equation $P=CV^2f$, in the worst case, the transceiver of each channel consumes $2*(0.780V)^2*(60pF/m)*(4.6m)*(1.65Gbps)=554mW$ to switch the cable capacitance alone if voltage signaling is used. The power consumption is significantly higher than that in the context of the conventional memory bus encoding. For example, the typical parameters of off-the-shelf standard memory chips, e.g. Synchronous DRAM, are $3.3V_{dd}$, 400MHz, and around 15pF capacitance on each wire. In other words, low power encoding techniques are more eligible for the DVI than the memory buses if the power consumption of the display device is comparable to the memory chips.

II. Digital Visual Interface

Consider a typical portable embedded system [11] as shown in Fig. 1. The system consists of a mainboard (including the CPU, memory, video controller, etc.), a display, and a DVI connection in between. The pixel data is prepared by the application and stored in the frame-buffer by the CPU via memory-write instructions. The video controller fetches the pixel data from the frame-buffer and then generates the proper video signals through the DVI. The pixels on the display are scanned in a left-to-right, top-down fashion. Therefore, the data of two adjacent pixels on the same row will be sent through the channel consecutively. The DVI consists of one or two TMDS links (Fig. 2).

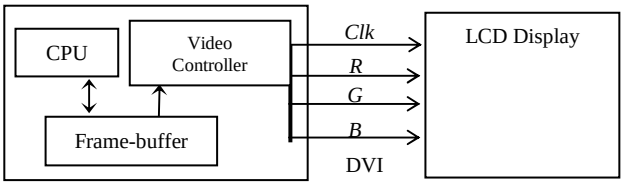


Fig. 1. The target system using a DVI display.

A. TMDS

A TMDS link consists of one clock channel and three data channels (0-1-2 or 3-4-5) for the R-G-B colors. For each pixel, the three color components are separately transmitted via the three data channels at the same time. A TMDS transmitter performs operations on each 8-bit source-word and obtains a 10-bit code-word. The first operation (transition minimization -- TM) minimizes bit-wise transitions in order to reduce power consumption, avoid excessive electromagnetic interference (EMI) levels and increase skew-tolerance. Given a source-word $D[7..0]$, the TM encoding selects either one of the following functions:

$$TM_{XOR}(D[7..0]) = E[8..0], E[i] = \begin{cases} D[0], i = 0 \\ D[i] <> E[i-1], 0 < i < 8 \\ 1, i = 8 \end{cases}$$

$$TM_{XNOR}(D[7..0]) = E[8..0], E[i] = \begin{cases} D[0], i = 0 \\ D[i] == E[i-1], 0 < i < 8, \\ 0, i = 8 \end{cases}$$

and inserts one redundant bit $E[8]$ to indicate the decision. The two functions can be converted by the following transition-invert (TI) function

$$TI(D[7..0]) = E[7..0], E[i] = \begin{cases} D[0], i = 0 \\ \overline{D[i-1] \oplus D[i] \oplus E[i-1]}, 0 < i < 8 \end{cases}$$

and have the following properties:

$$TI(TM_{XOR}(x)) = TM_{XNOR}(x),$$
$$TI(TM_{XNOR}(x)) = TM_{XOR}(x).$$

The TM encoding can be considered the serial version of the transition signaling encoding algorithm [2] with one redundant bit.

The second operation (inversion -- INV) eclectically inverts the TM-coded 9-bit word to balance the DC signal, i.e., the numbers of ones and zeros. The INV function is defined as

$$INV(D[8..0]) = E[9..0], E[i] = \begin{cases} \overline{D[i]}, 0 \le i < 9 \\ 1, i = 9 \end{cases}.$$

The TMDS encoder has an internal counter to keep track of the difference between the numbers of ones and zeros so that the INV encoder can balance the occurrence of ones and zeros. Another redundant bit $E[9]$ is used to indicate the INV decision. The INV encoding is performed after the TM encoding, and does not change the transition count within the word.

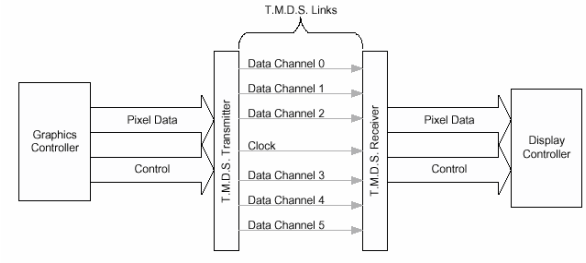


Fig. 2. Digital Visual Interface [9].

The sum of the transition counts on the channel is $\sum_i \left(\sum_{j=0}^{8} e(x_i)[j] \oplus e(x_i)[j+1] + e(x_i)[9] \oplus e(x_{i+1})[0] \right)$, where x_i is the source-word at time i, and e is the TMDS encoding. The first term represents the intra-word transitions, and the second term represents the inter-word transitions. Define w as the intra-word transition counts $w(x) = \sum_{j=0}^{8} x[j] \oplus x[j+1]$. The total transition counts can be re-written as $\sum_i \left(w(e(x_i)) + e(x_i)[9] \oplus e(x_{i+1})[0] \right)$. Since the INV encoding effects only the 1-bit inter-word transitions and is controlled by the number of previously sent ones/zeros, it is not considered in this paper. Thus, the goal is to find the optimal substitute of the TM encoding, an encoding function e from 8-bit source-words to 9-bit code-words that minimizes the intra-word transitions.

III. TONAL LOCALITY

In this section, the tonal locality claim is made -- a pixel is likely to be similar to its neighboring pixels in terms of the signal values.

A. Motivation

The tonal locality can be intuitively explained from the aspect of the edge detection techniques of digital image processing [12]. Given the set of the pixels on an image as a two-dimensional array of signals and consider its frequency space. Edge detection algorithms act as high-pass filters. The signals with high frequencies, i.e., the pixels located in the regions that present fast transition in tonality, pass through the filter and are reported as edges. The remaining pixels, such as those from surfaces of objects, have low frequencies and

therefore have similar tone as their neighboring pixels.

Another source of tonal locality is image compression. The kernel of the image compression algorithms is the inverse discrete cosine transform (IDCT). The IDCT is applied to the luminance and chromaticity of each macroblock. After the transformation, the high-frequency coefficients, which represent the very fine details on the image that are hardly perceivable by human eyes, are discarded to reduce the information entropy. Therefore, the variation between adjacent pixels in the same macroblock is limited, though fast transitions may still exist between adjacent macroblocks.

B. Benchmark image suite

To justify the claim of tonal locality, the statistics were examined on a set of benchmark images from the USC SIPI Image Database (USID) [13]. The USID is considered the *de facto* benchmark suite in the signal and image processing research field [12]. It consists of three volumes (texture patterns, aerial photos, and miscellaneous) of color and black-and-white images in different sizes. The results reported here are from 8 color images from volume 3. All of them have 256 by 256 pixels. The color depth is 24 bits, i.e., 8 bits per color-channel in the range of 0 to 255. These images are uncompressed and stored in the TIFF (Tag-based Image File Format) format.

C. Spatial locality

Correlation analysis was performed on these images. First, each image was decomposed into 3 channels. The coefficients of correlation between every pair of pixels within a 5*5 region were calculated. The coefficients of determination (the square of the coefficient of correlation) were then calculated. Table 1 shows the results of the benchmark image 4.1.1. The value in the table at row v and column u represents the coefficient between the pixel at location (i,j) and the pixel at location $(i+u,j+v)$.

Table 1
Coefficients of determination of 4.1.1

R	*i*	*i+1*	*i+2*	*i+3*	*i+4*
j	1.00	0.97	0.93	0.88	0.84
j+1	0.96	0.95	0.91	0.87	0.83
j+2	0.90	0.90	0.87	0.84	0.81
j+3	0.85	0.84	0.83	0.81	0.78
j+4	0.79	0.79	0.79	0.77	0.75

G	*i*	*i+1*	*i+2*	*I+3*	*i+4*
j	1.00	0.97	0.92	0.89	0.85
j+1	0.96	0.95	0.91	0.88	0.85
j+2	0.92	0.91	0.89	0.86	0.83
j+3	0.88	0.87	0.86	0.84	0.82
j+4	0.84	0.84	0.83	0.81	0.79

B	*i*	*i+1*	*i+2*	*i+3*	*i+4*
j	1.00	0.95	0.91	0.87	0.83
j+1	0.95	0.93	0.90	0.86	0.83
j+2	0.91	0.90	0.87	0.85	0.82
j+3	0.87	0.86	0.85	0.82	0.80
j+4	0.84	0.83	0.82	0.80	0.78

According to the above data, a pixel is highly correlated (≥0.95) to its adjacent pixels in each channel. The coefficients decrease gradually as the distance increases.

To exploit the above correlation, the approach presented here is to use the neighboring pixels to predict the present pixel. The remaining errors will be coded and sent over the channels. The more accurate the predictor is, the fewer transition counts the code-words will have. The predictor gains more accuracy if more reference pixels are used at the cost of memorizing the reference pixels. However, while scanning an image, only half of the 8 neighboring pixels have been processed by the encoder. How many pixels are sufficient to build an accurate predictor? To answer this question, 1-, 2-, 3-, and 4-way multiple linear regression analysis was performed on the 8 benchmark images. The independent variables were chosen from the four neighboring pixels (west, north, northwest, and northeast) incrementally. Table 2 shows the average of the standard deviation of the residues of the three color channels. The confidence level is 0.95.

Table 2
Multiple regression analysis

	1-way	2-way	3-way	4-way
4.1.01	11.0130	8.7331	8.2357	8.0786
4.1.02	12.0754	8.2141	7.0998	7.0313
4.1.03	6.8144	6.2441	5.3144	5.2100
4.1.04	15.8660	8.4618	5.8766	5.8178
4.1.05	11.0146	8.6054	6.3802	6.2936
4.1.06	16.6531	13.5351	11.9507	11.7988
4.1.07	8.0694	5.8347	4.4617	4.3720
4.1.08	10.3856	7.7344	5.6498	5.4513

The above data show that while the two-way predictor outperforms the one-way predictor, three and four-way

predictors do not gain much accuracy. However, to implement the two-way predictor, all of the pixels on the previous (north) row, usually hundreds, have to be stored. Due to the extra buffers and power overhead required, a one-way predictor, i.e., encoding against the previous (west) pixel, will be used here.

Fig. 3 shows the distributions of the signal value differences, $x_i'=x_i-x_{i-1}$, of the benchmark images. In fact, three curves from each color channel of each image are plotted. However, they are completely overlapped and only 8 curves are shown. The curves follow Gaussian distributions with the same means equal to zero and different standard deviations. Most of the differences are within the range of [-15,15].

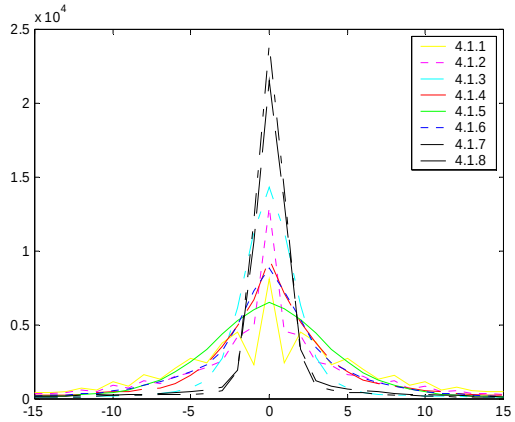


Fig. 3. The distributions of the signal value differences.

D. Chromatic locality

Other than the spatial correlation, the correlation among the color-channels is also of our interest. The results of correlation are shown in Table 3. Unlike the spatial locality, no special pattern is found.

Table 3
Coefficients of correlation among the R, G, and B color channels

CorrCoef	(R,G)	(G,B)	(B,R)
4.1.01	0.7712	0.9126	0.6819
4.1.02	0.8992	0.9478	0.8040
4.1.03	0.8579	0.9837	0.9098
4.1.04	0.6207	0.9274	0.6880
4.1.05	0.6378	0.9418	0.4823
4.1.06	0.0583	0.9736	0.0689
4.1.07	0.7016	0.8519	0.6478
4.1.08	0.6766	0.8481	0.6297

IV. ENCODING FRAMEWORK

A. Bus encoding framework

A generic framework of memory bus encoding is proposed in [1]. Fig. 4 shows the organization of a memory bus encoder. For each source word x_i, its predecessor x_{i-1} is stored in a register. The predict function F of x_{i-1} is used by the encoder and decoder simultaneously to predict the current source word. The discrepancy between x_i and $F(x_{i-1})$ is calculated by the function D and coded by the entropy coder E. Every memory bus encoding algorithm can be described by the three parameters (F, D, E). Take memory address bus as an example, based on the heuristic of spatial locality, the next source word is likely to be the address of the following instruction, which is 4 bytes away. If the prediction is incorrect (e.g. branch), then the offset is calculated as the difference between the target address and predicted address. The entropy coder reduces the switching activities by using a proper data representation such as binary or the limited-weight code [3]. This encoding scheme can be described as (`add4`, `subtract`, `binary`). Likewise, the decoder operates in the reverse way.

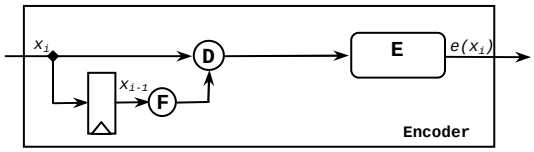


Fig. 4. A generic framework of bus encoding.

B. Tonal encoding

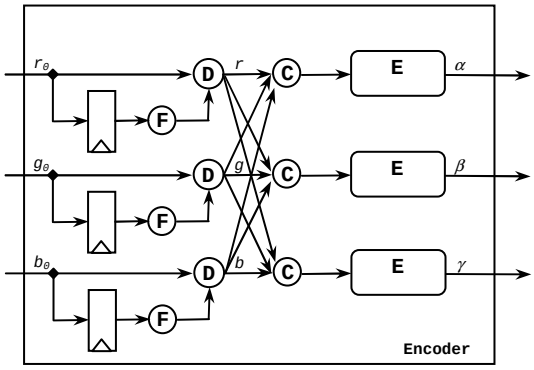


Fig. 5. Tonal encoding framework.

The proposed encoding framework consists of two stages: spatial encoding and chromatic encoding. Based on the correlation analysis done in previous section, the spatial encoding is performed before the chromatic encoding. The spatial encoding (F and D) subtracts the previous (west) pixel value from the current one, the chromatic encoding (C and E) takes the differences

between the three color channels, and encodes the signed binary numbers by the codebook lookup. In this paper, an encoding algorithm is described by (*F*, *D*, *C*, *E*).

C. Spatial encoding

As the tonal locality suggests, the spatial encoding is performed first. A one-way predictor is implemented by the register and the subtract function *D* in Fig. 5. The register stores the previous source word x_{i-1}. According to the pixel scanning order, the previous source word is the neighboring pixel in the west (except for the boundary conditions). The prediction function *F* is the identity function, so it does not exist in the hardware implementation. Using one-way predictor reduces the cost of implementing the *F* function and the storage for the previous pixel. The function *D* calculates the prediction error $x_i - x_{i-1}$, and extends the range to [-255,255].

D. Chromatic encoding

After the spatial encoding, the chromatic encoding applies two additional functions. The chromatic function *C* encodes the source word by taking the values of the other two channels into account. The entropy coder *E* instructs the final data representation. Two combinations are proposed: chromatic-diff and chromatic-xor.

E. Chromatic-Diff

Chromatic-diff is a numerical approach to reduce the transition counts. The three values from each channel can be encoded by their reciprocal differences to reduce the transition counts. Recall the monotonously increasing property of the OTC: $f(x) \leq f(y)$ if $|x|<|y|$. To minimize the transition counts of the code-words, our encoding strategy is to minimize the magnitudes of the source-words. The reciprocal differences are calculated and compared against the original values. The three values with smaller magnitudes out of six will be chosen as the code-words and sent along with the redundant bits.

The selection problem can be formulated as an equivalent graph problem. Let *r*, *g*, and *b* be the three values from the three color channels. Consider a weighted clique of four vertices {*a*, *b*, *c*, *o*}. The weights of *oa*, *ob*, and *oc* (called inputs) are *r*, *g*, and *b*, respectively. The weights of *ab*, *bc*, and *ca* (called differences) are the differences *g*–*r*, *b*–*g*, and *r*–*b*, respectively. Among these six edges, in order to represent the original values *r*, *g*, and *b*, exactly three edges are needed, and they must form a tree. The optimal solution is the tree with the minimum weight, i.e., the minimum spanning tree [13].

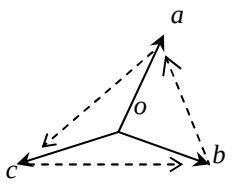


Fig. 6. The minimum spanning tree problem.

Since {*a*, *b*, *c*, *o*} are actually scalars, they can be sorted into 24 cases as shown in the first column of Table 4. The optimal selection is simple to determine. The 24 cases can be reduced to 12 cases because each case has a dual, which is in reverse order and has the same optimal selection. These 12 cases can again be categorized into two types. In the first type ($0<x<y<z$ or $z>y>x>0$), including cases 1~6, one input (*x*) and two differences (*y*-*x* and *y*-*z*) are chosen. The *x* is indicated by the redundant bits. The two differences are assigned to opposite signs so that (*y*-*x*) has the same sign as *x* to indicate that *y* is encoded as the difference from *x*. In the second type ($x<0<y<z$ or $z<y<0<x$), including cases 7-12, two inputs (*x* and *y*) and one difference (*z*-*y*) are chosen. The *x* and *y* have opposite signs and are indicated by the redundant bits. The difference *z*-*y* has the same sign as *y* to indicate that *z* is encoded as the difference from *y*. In this way, 12 cases are reduced to 6, and only 3 redundant bits are required.

Table 4 shows the encoding and decoding functions and the assignment of the redundant bits. For the encoder, the inputs are *r*, *g*, and *b*, and the outputs are α, β, and γ. The redundant bits are z_0, z_1, and z_2. The original signal values before spatial encoding are r_0, g_0, and b_0. The '+' and '-' operators are arithmetic addition and subtraction, respectively. For example, in the first case $0<r<g<b$, *r* is the closest value to zero, so *r* is sent as is. The redundant bit z_0 is set to 1, indicating that no encoding/decoding is needed for channel 0. The *g* is encoded as *g*-*r*, which has the same sign as *g*, while *b* is encoded as *g*-*b*, which has the opposite sign. The z_1z_2 are set as zeros, meaning that decoding is needed, which in turn requires the comparison of their signs. Consider another example. In the 7th case $b<0<g<r$, the *b* and *g* are sent without encoding, while the *r* is encoded as *r*-*g*, which has the same sign as *g*. The $z_0z_1z_2$ are set as 011. The *r* can be recovered correctly based on the fact that *r*-*g* has the same sign as *g*.

Table 4
The chromatic encoding function

#	Ranking	Encoding					Decoding		
		α	*β*	γ	$z_0z_1z_2$	if sign()	*r*	*g*	*b*
1	*0<r<g<b; b<g<r<0*	*r*	*g-r*	*g-b*	100	α=*β*	α	α+*β*	α+*β*–γ
2	*0<r<b<g; g<b<r<0*	*r*	*b-g*	*b-r*	100	α=γ	α	α+γ–*β*	α+γ
3	*0<g<b<r; r<b<g<0*	*b-r*	*g*	*b-g*	010	*β*=γ	*β*+γ–α	*β*	*β*+γ
4	*0<g<r<b; b<r<g<0*	*r-g*	*g*	*r-b*	010	*β*=α	*β*+α	*β*	*β*+α–γ
5	*0<b<r<g; g<r<b<0*	*r-b*	*r-g*	*b*	001	γ=α	γ+α	γ+α–*β*	γ
6	*0<b<g<r; r<g<b<0*	*g-r*	*g-b*	*b*	001	γ=*β*	γ+*β*–α	γ+*β*	γ
7	*b<0<g<r; r<g<0<b*	*r-g*	*g*	*b*	011	α=*β*	*β*+α	*β*	γ
8	*g<0<b<r; r<b<0<g*	*r-b*	*g*	*b*	011	α=γ	γ+α	*β*	γ
9	*r<0<b<g; g<b<0<r*	*r*	*g-b*	*b*	101	*β*=γ	α	*β*+γ	γ
10	*b<0<r<g; g<r<0<b*	*r*	*g-r*	*b*	101	*β*=α	α	*β*+α	γ
11	*g<0<r<b; b<r<0<g*	*r*	*g*	*b-r*	110	γ=α	α	*β*	γ+α
12	*r<0<g<b; b<g<0<r*	*r*	*g*	*b-g*	110	γ=*β*	α	*β*	γ+*β*
13	*overflow*	r_0	g_0	b_0	000	-	α	*β*	γ
14	*overflow*	*TI*(r_0)	*TI*(g_0)	*TI*(b_0)	111	-	*TI*(α)	*TI*(*β*)	*TI*(γ)

Notice that the subtraction computation is performed in the extended range [-255,255]. After the spatial and chromatic encoding, if the resulted encoded data from any channel falls out of the range [-128,127], then all of the three channels will be encoded by the original TMDS algorithm. This condition is indicated by the use of redundant bits equal to 000 or 111, and interpreted in the same way as the *E*[8] bit of the TMDS encoding.

A. Code assignment

If the statistics of the pixel signals are known, then the total transition count can be written as $T = \sum_{x=0}^{255} p(x)w(e(x))$, where *p* is the occurrence of signal *x*. The objective is to find an optimal assignment *e* such that *T* is minimized. Based on the tonal locality, the differences between consecutive pixels follow a Gaussian distruibution. The transition count can be written in terms of the signal differences *x'*: $T = \sum_{x=-255}^{255} p(x')w(x')$. The *p(x')*, which follows a Gaussian distribution, has the property $p(x) \le p(y)$ if $|x|>|y|$. The optimal code assignment will be any function *f* that satisfies $f(x) \le f(y)$ if $|x|<|y|$. Functions in this class are called ordered transition codes (OTC). If the length of the code-words is *N*, then there are $2\binom{L-1}{i} = \frac{2(L-1)!}{i!(L-1-i)!}$ code-words having exactly *i* intra-word transitions.

The code assignment is generated as a two-column lookup table. In the first column, all of the source-words are enumerated and sorted by their magnitudes. In the second column, all of the code-words (only *E*[7..0]) are enumerated and sorted by their intra-transition counts. The mapping from the first column to the second is the encoding function. The reverse mapping is the decoding function.

B. Chromatic-XOR

Chromatic-xor is a symbolic approach to reduce the transition counts. The decoding functions of chromatic-xor are shown in Table 5. Given the code-words α, β, and γ, the source-words *r*, *g*, and *b* can be obtained by examining the redundant bits z_0, z_1, and z_2. If a redundant bit is 1, the correspondent code-word is the original source-word and does not need to be decoded. Otherwise, it can be decoded by performing exclusive-or with the other un-encoded code-words. When all the redundant bits are zeros, it is a special case indicating that the spatial encoding overflows/underflows and the code-words are the original pixel values.

Table 5
Decoding of Chromatic-XOR

$z_0z_1z_2$	*r*	*g*	*b*
100	α	α⊕*β*	α⊕γ
010	α⊕*β*	*β*	*β*⊕γ
001	α⊕γ	*β*⊕γ	γ
101	α	α⊕*β*⊕γ	γ
110	α	*β*	α⊕*β*⊕γ
011	α⊕*β*⊕γ	*β*	γ
111	α	*β*	γ
000	r_0=α	g_0=*β*	b_0=γ

The chromatic-xor encoder works in the reverse way. It evaluates the transition counts for all cases and chooses the best one to encode accordingly.

The implementation overhead of the chromatic-xor encoder/decoder is less than that of chromatic-diff

because of the lack of table-lookup and the simpler XOR circuits. However, if any of the inputs (the signed 8-bit differences generated by the spatial encoder) overflows or underflows, then the encoder must select the 000 case and send the original pixel values bypassed by the spatial encoder.

V. EXPERIMENTAL RESULTS

A. *Still images*

Experiments are performed to evaluate the power savings of the chromatic encoding over the original DVI encoding. For fairness, some minor modifications are needed.

In the DVI implementation, since the *M*[9] does not affect the intra-word transition counts, it is set to zero, and therefore no inversion is applied. Because the decision of XOR or XNOR, which is indicated by *M*[8], does not affect the transition counts, the *M*[8] is always set to zero. In the chromatic encoding implementation, the *M*[9] is set to zero as is the DVI. The *M*[8]'s from each channel are used to carry the three redundant bits $z_0z_1z_2$.

The (*Id,Diff,-,Bin*) uses 9 bits to represent the extended range for the differences. The (*Id,Diff,-,TM*) uses 8-bit difference and one bit for TM and therefore may transmit the pixels incorrectly. The (*Id,Diff,-,OTC*) uses 9-bit extended range, too. The (*Id,Diff,XOR,Bin*) and (*Id,Diff,Diff,OTC*) use two special cases to send the original or transition-inverted pixel values when overflow occurs and therefore are lossless.

Table 6
Comparison of encoding algorithms.

F D C E	- - - TM	Id Diff - Bin	Id Diff - TM	Id Diff - OTC	Id Diff XOR Bin	Id Diff Diff OTC
4.1.01	27.12	46.03	48.19	60.13	56.53	64.64
4.1.02	19.43	44.60	47.04	57.21	55.68	63.39
4.1.03	32.11	64.57	65.18	67.92	72.32	73.48
4.1.04	28.53	50.72	53.04	60.97	61.32	66.43
4.1.05	32.13	55.73	57.18	65.14	64.34	69.18
4.1.06	34.65	48.35	52.19	59.85	59.12	67.05
4.1.07	24.82	67.73	68.73	71.01	72.49	76.08
4.1.08	26.69	63.74	65.29	68.70	69.60	73.96
Average	28.19	55.18	57.11	63.87	63.93	69.28
Ratio	1.00	1.96	2.03	2.27	2.27	2.46
Area	399	2644	-	-	1656	21881
Power	1.58	4.03	-	-	4.63	33.63

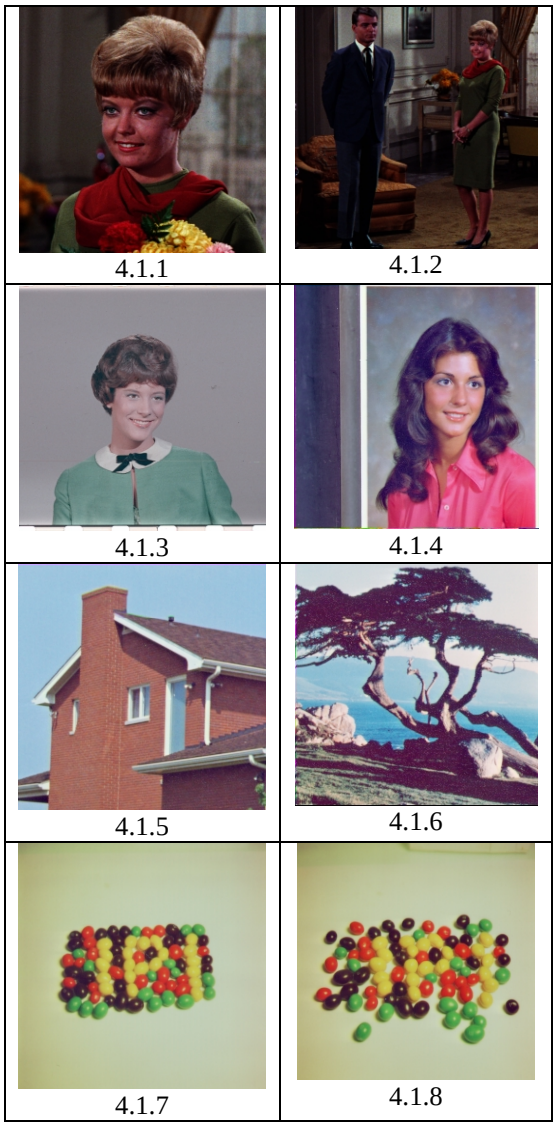


Fig. 7. The USID Images.

The results of the 6 encoding algorithms can be sorted into 4 groups by their efficiency in transition reduction. The original DVI algorithm (-,-,-,*TM*) reduces 28% of transitions. In the second group, the (*Id,Diff,-,Bin*) and (*Id,Diff,-,TM*) algorithms improve the reduction by 96 and 103% by applying spatial encoding. The TM following the spatial encoding gains only 7% improvement. In the third group, the (*Id,Diff,-,OT*C) and (*Id,Diff,XOR,Bin*) deliver a 127% improvement over the DVI by using the OTC coding and the XOR operation, respectively. In the last group, the (*Id,Diff,Diff,OTC*) is the best performer providing a 146% improvement over the DVI.

Notice that in the (*Id,Diff,-,Bin*) and (*Id,Diff,-,OTC*) encoding, the output values have been sign-extended to 9 bits. However, in the (*Id,Diff,-,TM*) encoding, the

pixels will be transmitted incorrectly if the spatial encoding overflows or underflows.

B. Video clips

The eight benchmark images may not be representative enough to evaluate the tonal locality and chromatic encoding due to the limited sample size. Instead of still images, three mpeg2 video clips, which consist of hundreds of images, are used as a more general benchmark suite. The video clips were selected from different sources to present diversity. The `tiger` movie is a documentary on wild animals. The `wg` movie is a clay animation. The `final3` movie is a Japanese anime. The `mpegplay` program was used to decode and dump the individual frames. The results are presented in Table 7. Up to 81% of the transition counts can be eliminated by the spatial and chromatic encoding. The encoding schemes gain more transition reduction because of the mpeg2 compression [14].

Table 7
Transition reduction by chromatic encoding

Clip	Frame	Size	- TM	Id Diff OTC	Id Diff XOR Bin
wg	331	304*224	30.56%	75.58%	81.93%
			1.00	2.47	2.68
tiger	634	320*240	29.81%	64.63%	74.46%
			1.00	2.17	2.5
final3	1018	160*128	30.41%	72.91%	78.92%
			1.00	2.40	2.60

C. Encoder overhead

To evaluate the power overhead of the proposed chromatic encoders, a real video controller core was synthesized to compare their power consumptions. A product-grade IP core, `VGA/LCD-core`, from the public domain [16] was used. It supports both CRT and LCD displays with user-programmable resolutions and video timings. Different color depths including 24 bit-per-pixel were supported. Dual-ported memory was used as the FIFO for higher performance. The color-processor unit was modified to perform chromatic encoding and synthesize the core. The simulated power consumption of the chromatic-diff encoder is 55.07mW at 1.8V 1.65GHz. The spatial encoder, chromatic encoder, and codebook consume 12.52mW, 38.97mW, and 1.04mW, respectively.

VI. CONCLUSION

This paper has presented a low-power encoding technique, chromatic encoding, for the DVI, a digital serial video interface. We have proved that chromatic encoding reduces power consumption by minimizing the transition counts on the DVI. This technique relies on the notion of tonal locality, i.e., the observation - first made in this paper - that the signal differences between adjacent pixels in images follow a Gaussian distribution. Based on this observation, an optimal code assignment is performed to minimize the transition counts. Furthermore, it is found that the three color channels of the DVI may be reciprocally encoded to achieve even more power saving. The channel selection problem is formulated exactly as a minimum spanning tree problem and solved accordingly. The proposed technique requires only three redundant bits for each 24-bit pixel. Experimental results show up to a 75% power reduction in the DVI.

REFERENCES

[1] S. Ramprasad, N. R. Shanbhag, and I. N. Hajj, "A coding framework for low-power address and data busses," *IEEE Transaction on VLSI Systems*, Vol. 7, No. 2, pp. 212-221, June 1999.

[2] L. Benini, G. DeMicheli, E. Macii, M. Poncino, and S. Quer, "System-level power optimization of special purpose applications: The beach solution," *ISLPED-97: ACM/IEEE International Symposium on Low Power Electronics and Design*, 1997, pp. 24-29.

[3] E. Musoll, T. Lang, and J. Cortadella. "Exploiting the locality of memory references to reduce the address bus energy," *ISLPED-97: ACM/IEEE International Symposium on Low Power Electronics and Design*, 1997, pp. 202-207.

[4] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," *IEEE Transactions on VLSI Systems*, Vol. 3, No. 1, pp. 49-58, Month, 1995.

[5] W. C. Cheng and M. Pedram, "Power-optimal encoding for DRAM address bus," *ISLPED-00: ACM/IEEE International Symposium on Low Power Electronics and Design*, 2000, pp. 250-252.

[6] Intel Pentium 4 Processor with 512-KB L2 Cache on 0.13u Process at 2 GHz, 2.20 GHz, and 2.40 GHz Datasheet, Intel, March 2002.

[7] W. Kowalsky et al., "OLED matrix displays: technology and fundamentals," *First International Conference on Polymers and Adhesives in Microelectronics and Photonics*, 2001, pp. 20-28.

[8] http://www.vesa.org

[9] Digital Visual Interface, V 1.0. Digital Display Working Group, April, 1999. http://www.ddwg.org.

[10] Extron Electronics, *Cable Products Guide*, http://www.extron.com.

[11] Intel PXA250 and PXA210 Application Processors Developer's Manual, Intel, February 2002.

[12] W. K. Pratt, *Digital Image Processing*, New York: John Wiley & Sons, Inc., 1991.

[13] A. G. Weber, "The USC-SIPI Image Database. Version 5," USC-SIPI Report #315, October 1997. http://sipi.usc.edu/services/database/Database.html

[14] MPEG Standard; ISO/IEC 13818.

[15] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, Cambridge: The MIT Press, 1990.

[16] Opencores. http://www.opencores.org/projects/vga_lcd.

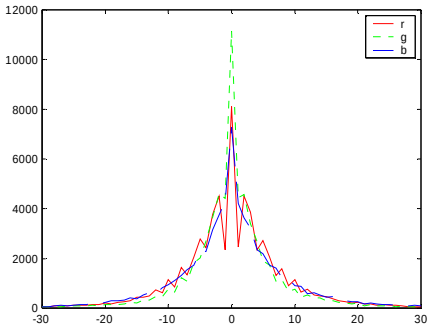
Wei-Chung Cheng (M'00) received his Ph.D. degree in Electrical Engineering from the University of Southern California in 2003. His current work focuses on system-level power optimization.

Massoud Pedram (S'88–M'90–SM'98–F'01) received a B.S. degree in Electrical Engineering from the California Institute of Technology in 1986 and M.S. and Ph.D. degrees in Electrical Engineering and Computer Sciences from the University of California, Berkeley in 1989 and 1991, respectively. He then joined the department of Electrical Engineering - Systems at the University of Southern California where he is currently a professor. Dr. Pedram has served on the technical program committee of a number of conferences, including the Design Automation Conference (DAC), Design and Test in Europe Conference (DATE), Asia-Pacific Design automation Conference (ASP-DAC), and International Conference on Computer Aided Design (ICCAD). He served as the Technical Co-chair and General Co-chair of the International Symposium on Low Power Electronics and Design (SLPED) in 1996 and 1997, respectively. He is the Technical Program Chair of the 2002 International Symposium on Physical Design. Dr. Pedram has published three books, 50 journal papers, and more than 130 conference papers. His research has received a number of awards including two ICCD Best Paper Awards, a Distinguished Paper Citation from ICCAD, a DAC Best Paper Award, and an IEEE Transactions on VLSI Systems Best Paper Award. He is a recipient of the NSF's Young Investigator Award (1994) and the Presidential Faculty Fellows Award (a.k.a. PECASE Award) (1996).

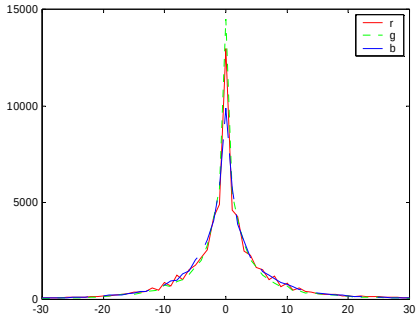
Dr. Pedram is a Fellow of the IEEE, a member of the Board of Governors for the IEEE Circuits and systems Society, an IEEE Solid State Circuits Society Distinguished Lecturer, a board member of the ACM Interest Group on Design Automation, and an associate editor of the IEEE Transactions on Computer Aided Design and the ACM Transactions on Design Automation of Electronic Systems.

His current work focuses on developing computer aided design methodologies and techniques for low power design, system-level dynamic power management, smart battery design and management, and integrated RT-level synthesis and physical design.

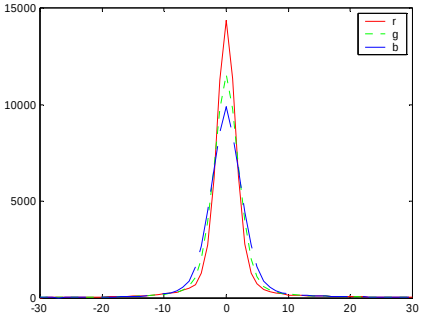
APPENDIX: THE DISTRIBUTIONS OF THE PIXEL VALUE DIFFERENCES



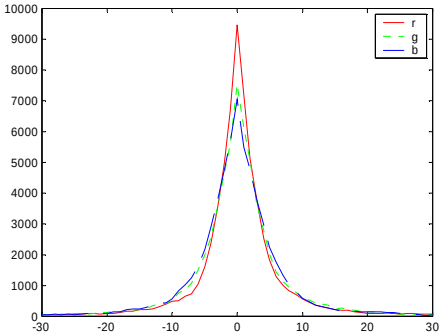
4.1.1



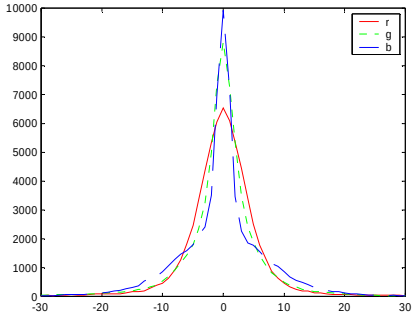
4.1.2



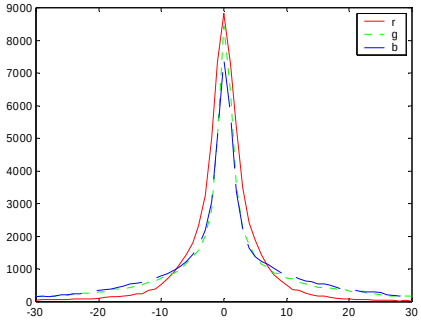
4.1.3



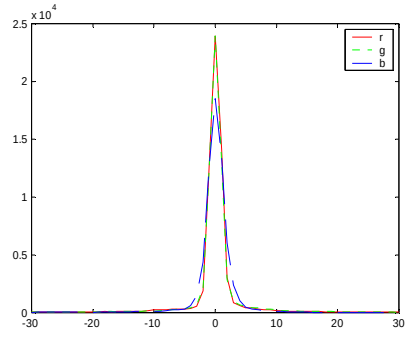
4.1.4



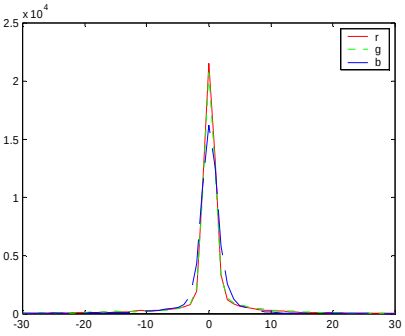
4.1.5



4.1.6



4.1.7



4.1.8