Simple high-rate constrained codes

K.A. Schouhamer-Immink and A. Wijngaarden

Indexing term: Constrained codes

A new coding technique is proposed that translates binary user information into a constrained sequence having the virtue that at most k 'zeros' between logical 'ones' will occur. The new construction offers a high rate while both the complexity for encoding and decoding are still very low. Single channel bit errors will result in at most one decoded byte error. A worked example is described with rate 16/17, $k = 6$ code.

Introduction: In the transmission of binary data, it is generally desirable that the received signal is self-synchronising or self-clocking. Timing is commonly recovered with a phase-locked loop which adjusts the phase of the detection instant according to observed transitions of the received waveform. The maximum runlength parameter, denoted by k, ensures adequate frequency of transitions for synchronisation of the read clock. A k-constrained code generates sequences characterised by the fact that at most k 'zeros' will occur between consecutive 'ones'.

High-rate codes have been extensively used in recording systems, specifically in magnetic tape drives. Examples of such codes are the rate 4/5, Group-Coded Recording [1] and the rate 8/9, (0,3) code [2]. Other k-constrained codes with a rate > 8/9 have not, to the authors' knowledge, been used in recording systems. It is not unrealistic to say that the hardware and power requirements of high-rate codes have hampered their introduction.

Description of new block codes: We now focus on simple k-constrained block codes that translate n–1 user bits into codewords of n, $n \geq 9$, channel bits. The bit stream formed by concatenating the codewords has the virtue that at most k = 1+ entier (n/3), $n \geq 9$, between logical 'ones' will occur. The construction is characterised by the fact that to build the codeword at most eight bits of the (n–1)-tuple have to be altered, which has a bearing on the worst-case error propagation.

Definitions: Let $k(x)$ be the maximum runlength of 'zeros' in the word x. Let $l(x)$ be the number of consecutive leading 'zeros' of the word x, that is, the number of 'zeros' preceding the first 'one'. And let $r(x)$ be the number of consecutive trailing 'zeros' of the word x, that is, the number of 'zeros' succeeding the last 'one'.

Preliminaries: Let the (n–1)-tuple $z = (z_1, ..., z_{n-1})$ be the binary input word and let $p = 1+(n-2)/2$. Define the intermediate n-tuple $\mathbf{x} = (x_1, ..., x_n)$ by $x_i = z_i$, $1 \leq i \leq p-1$, $x_{i+1} = z_i$, $p \leq i \leq n$, and set the pivot bit $x_p := 1$. The left and right parts of the vector $\mathbf{x}$, $\mathbf{xl}$ and $\mathbf{xr}$, are defined by $xl_i = z_i$, $xr_i = z_{i+p}$, $1 \leq i \leq n/2$. A more judicious allocation of the source bits is possible in a byte-oriented system (see later). Set the pivot bit $x_p := 1$.

Main algorithm, k = 1+entier(n/3): Let r = entier(k/2); $l = k-r$, and define the intermediate n-tuple $\mathbf{y} = (y_1 ..., y_n)$, where $y_i = x_i$, $1 \leq i \leq n$

(i) If $l(\mathbf{y}) \leq 1$ and $r(\mathbf{y}) \leq r$ and $k(\mathbf{y}) \leq k$ then transmit $\mathbf{y}$ as is.

(ii) If ($l(\mathbf{y}) \leq l$ and $k(\mathbf{xl}) \leq k$) and ($r(\mathbf{y}) > r$ or $k(\mathbf{xr}) > k$) then begin $y_{p-1} := 1$; $y_p := 0$; $y_{p+1} := 0$; $y_{n-r} := 1$; $y_{n-r+1} := x_{p-1}$; $y_{n-r+2} := x_{p+1}$; transmit $\mathbf{y}$; end;

(iii) If ($r(\mathbf{y}) \leq l$ and $k(\mathbf{xr}) \leq k$) and ($l(\mathbf{y}) > l$ or $k(\mathbf{xl}) > k$) then begin $y_{p-1} := 0$; $y_p := 0$; $y_{p+1} := 1$; $y_{l+1} := 1$; $y_{l-1} := x_{p+1}$; $y_l := x_{p-1}$; transmit $\mathbf{y}$; end;

(iv) If ($l(\mathbf{y}) > l$ and $k(\mathbf{xl}) > k$) and ($r(\mathbf{y}) > r$ or $k(\mathbf{xr}) > k$) then begin $y_{p-1} := 1$; $y_p := 0$; $y_{p+1} := 1$; $y_{n-r} := 1$; $y_l := x_{p-1}$; $y_{n-r+1} := x_{p+1}$; transmit $\mathbf{y}$; end;

During decoding, the pivot symbol y_p is observed. If $y_p = 1$ then decoding is straightforward. If, on the other hand, $y_p = 0$ the bits y_{p-1} and y_{p+1} are used to uniquely re-constitute the original (n–1)-tuple. Note that, in total, at most eight bits of the original (n–1)-tuple $\mathbf{z}$ are involved in the scheme, namely y_1, y_2, y_3, y_{p-1}, y_{p+1}, y_{n-2}, y_{n-1} and y_n. To avoid error propagation in a byte-oriented system, these bits should be taken from one input byte. Then any error propagation resulting from an error in the pivot bit y_p occurring during transmission is confined to one decoded byte.

Description of rate 16/17 code: The code translates two bytes of user data into 17 channel bits. The 17 bit codewords are characterised by the fact that they have at most six consecutive 'zeros' and have at most three leading and at most three trailing 'zeros'. Error propagation is limited because any single channel bit error made during retrieval will result in at most one decoded byte error. Let $\mathbf{z} = (z_1, ..., z_{16})$ be the 16 bit input word. The 17 bit word $\mathbf{y} = (y_1, ..., y_{17})$ is obtained by shuffling:

z_1	z_9	z_{10}	z_{11}	z_2	z_3	z_4	z_{12}	1	z_{13}	z_5	z_6	z_7	z_{14}	z_{15}	z_{16}	z_8

Encoding algorithm: Define the Boolean variables (the '+' denotes the logical 'or'-function) $L1 = y_1+y_2+y_3+y_4$, $L2 = y_1+ ... +y_8$ and let $R1 = y_{14}+y_{15}+y_{16}+y_{17}$, $R2 = y_{10}+ ... +y_{16}$.

Transmission of a word is based on the following four-step algorithm:

(i) If $L1L2R1R2$, then transmit $\mathbf{y}$ as it is.

(ii) If $L1L2\overline{R1R2}$, then reshuffle and transmit:

z_1	z_9	z_{10}	z_{11}	z_2	z_3	z_4	1	0	0	z_5	z_6	z_7	1	z_{12}	z_{13}	z_8

(iii) If $\overline{L1L2}R1R2$, then reshuffle and transmit:

z_1	z_{12}	z_{13}	1	z_2	z_3	z_4	0	0	1	z_5	z_6	z_7	z_{14}	z_{15}	z_{16}	z_8

(iv) If $\overline{L1L2R1R2}$, then reshuffle and transmit:

z_1	z_{12}	0	1	z_2	z_3	z_4	1	0	1	z_5	z_6	z_7	1	0	z_{13}	z_8

Each modification made during the encoding process is uniquely identifiable and decoding can therefore be carried out in a straightforward fashion. Note that the source bits (z_1, ..., z_8), which constitute the first source byte, are transmitted unaltered. The retrieved source bits (z_9, ..., z_{16}), which constitute the second source byte, are a function of the pivot bit y_5. It can therefore easily be seen that error propagation owing to any single channel bit error in the received codeword is restricted to, at most, one decoded byte error.

Conclusions: We have presented a new coding technique that translates binary user information into a constrained sequence, having the virtue that, at most, k 'zeros' between logical 'ones' will occur. Both the complexity of the encoding and decoding circuitry are very low and the coding scheme involves only a minor drawback in terms of overhead needed.

12 August 1996
Electronics Letters Online No: 19961268

K.A. Schouhamer Immink and A. van Wijngaarden (*Philips Research Laboratories, 5656 AA Eindhoven, The Netherlands*)

References

1 MILLIGAN, G.E.: 'Self-clocking five bit record playback system'. US Patent 3,641,525, February 1972

2 IMMINK, K.L.S.: 'Coding techniques for digital recorders' (Prentice-Hall International (UK) Ltd., Englewood Cliffs, New Jersey, 1991)

Algorithm based on modified threaded binary tree for estimating delay affected by internal charges in CMOS gates

M. Chang, Shuih-Jong Yih and Wu-Shiung Feng

Indexing terms: Delays, CMOS integrated circuits

Internal charges affect the delay behaviour of CMOS gates. A modified threaded binary (MTB) tree and a recursive algorithm are proposed to solve this problem. The charge sharing effect, which takes place among the internal nodes, is also considered in this algorithm.

Introduction: The internal charges in a CMOS gate will evidently increase the delay. For example, there is a 25% increase in delay for a five-input NAND gate with four fully charged internal nodes [1]. Therefore, we cannot neglect the delay affected by the charges