A Knuth-Based RDS-Minimizing Multi-Mode Code

C.H. Heymann, H.C. Ferreira
University of Johannesburg, Dept. E&E Eng. Science, RSA
Email: ch.heymann@googlemail.com, hferreira@uj.ac.za

J.H. Weber
TU Delft, The Netherlands
Email: J.H.Weber@ewi.tudelft.nl

Abstract—The Knuth codeword balancing approach is adapted to a DC-free, RDS-minimizing multi-mode coding scheme. Power spectra and sum variance metrics obtained with simulations are compared with those of the existing Knuth constructions, and error performance is evaluated for a binary symmetric channel.

I. INTRODUCTION

A DC-free code is a modulation code which maps a given input bit-sequence onto an output sequence which has zero power spectral density at zero frequency, before this new sequence is written or transmitted. This property is desirable in various transmission and recording system applications. These mappings are reversible, so that the receiver-reader can reconstruct the original bit-sequence.

Following work by Fair et al. on a technique called guided scrambling (GS) [1] [2], Immink and Pátrovics [3] gave a general classification of DC-free codes (as mono-, bi- and multi-mode codes), and analysed the performance of multi-mode codes (of which GS is an example) which include any scheme where a large selection set of candidate codewords are generated for each source word, and the candidate which best reduces the DC frequency component is used.

In [4], a method to construct balanced codewords from unbalanced source words was described by Knuth, where a number of initial bits of a source word is inverted to obtain a balanced codeword. Knuth also proved that each binary word has at least one such “inversion point”. Knuth’s approach selects the first codeword (with smallest number of bits inverted) that is balanced, so there is only one codeword per source word. This means it is not a multi-mode code.

In [5], an improvement on the Knuth approach was described, which takes advantage of the fact that a source word could have multiple possible balanced codewords, obtained by inverting different numbers of bits. If one considers all such possibilities, and selects the best one by some criterion, the result is a multi-mode code, with improved performance.

This paper describes a further adaptation of Knuth’s balanced codes, adapting it to a multi-mode code, where the running digital sum (RDS) is minimized throughout each codeword, rather than to be simply constrained to zero at the codeword boundaries. The scheme encodes the inversion point directly into the redundant prefix bits, so that there is no look-up table required for either encoding or decoding. The new scheme is more flexible in terms of the number of index bits than the original Knuth schemes, allowing more fine-grained trade-off between overhead and performance. At normalized frequency of $10^{-3}$, a (20,16) construction outperforms the original Knuth codes by at least 3.7dB, with larger gains for larger source words.

The remainder of this paper is organized as follows: Preliminaries in Section II are followed by a description of the new Knuth-based multi-mode code in Section III, which is also the new contribution in this paper. Section IV discusses results obtained with simulation. In Section V, the bit error performance of the new construction on a binary symmetric channel is analysed and compared with simulated results. Conclusions are drawn in Section VI.

II. PRELIMINARIES

A. Running Digital Sum

Most DC-suppression approaches minimize the running digital sum (RDS). The RDS is defined as follows in [6]:

Given an input sequence $\{x_i\} = \{x_{-1}, x_0, \ldots, x_i, \ldots\}$ with $x_i \in \{1, -1\}$, with “1” mapping to a digital “1” and a “−1” mapping to a digital “0”. The running digital sum $z_i$ can then be defined as the recursive relationship

$$z_i = z_{i-1} + x_i$$

The difference between the number of ones and zeroes in a single word is also referred to as the disparity of the word.

B. Mono-, Bi- and Multi-Mode Codes

From [3], there are three general classes of DC-free codes. In mono-mode codes, there is a one-to-one mapping from each source word to each codeword, with each codeword being balanced.

In bi-mode codes, each source word may be encoded using two possible codewords. The codeword that is chosen is the one that minimizes the RDS according to some selection criterion. The polarity switch method [7] is an example, in which an additional “1” is added to each source word, and the entire word is then either inverted or not, depending on which one minimizes the RDS after the codeword.

Multi-mode codes are an extension of bi-mode codes. More than two candidate codewords are evaluated against a selection criterion. The selection criterion chooses a codeword for each source word, such that specific properties of the coded bit sequence may be achieved. Guided scrambling [1] [2] is an example of a multi-mode code.

In [3], it is demonstrated that a sufficient condition for a multi-mode code to guarantee that the RDS can be limited is
that, for each codeword in the selection set, its complement must also be in the selection set.

The following selection criteria for multi-mode codes are discussed:

1) Minimum word-end RDS (MRDS): selects the codeword that minimizes the absolute RDS at the end of each codeword.
2) Minimum Squared Weight (MSW): selects the codeword with the minimum sum of squares of RDS at each codeword position. It has been shown [3] that MSW outperforms MRDS in multi-mode codes.
3) Modified MSW (MMSW): In [8], it is shown that some source word sequences may exist for which MSW allows the RDS to grow without bounds, even if the selection set contains complemented codewords. A simple solution is provided: A word-end RDS limit is applied first, and MSW is applied to select the optimal codeword from all those that result in a bounded word-end RDS. In [8], it is also shown that in simulations, the power spectrum is not significantly different between MSW and MMSW.

C. Efficiency of Multi-Mode Codes

In [3], the efficiency of an RDS-limited code is defined as the ratio between the redundancy-sum variance product of maxentropic sequences and the redundancy-sum variance product of a sequence obtained via the code. The redundancy-sum variance product of maxentropic sequences is constrained as follows:

$$0.25 \geq (1 - C(N))\sigma_s^2(N) > \frac{\pi^2/6 - 1}{4\ln 2} = 0.2326$$ (2)

where $(1 - C(N))$ is the redundancy of a maxentropic sequence which can take on N different RDS values, and $\sigma_s^2(N)$ is the sum variance (RDS variance) of the same sequence. Since the redundancy-sum variance product for maxentropic sequences is limited to such a small range of values, subsequent calculations can be simplified by approximating it as the lower bound 0.2326. The efficiency of RDS-limited codes is then approximated as:

$$E = \frac{(1 - C(N))\sigma_s^2(N)}{(1 - R)s^2} \approx \frac{0.2326}{(1 - R)s^2}$$ (3)

where $s^2$ is the sum variance of the code, and $R$ is the code rate.

D. Knuth’s Original Balanced Code and Derived Work

In [4] Knuth proposed a mono-mode scheme, where each source word is balanced by finding an index in the word, such that when all bits up to this index are inverted, the resulting codeword has zero disparity. Knuth proved that such an index always exists for all binary words. The inversion index is encoded and added to the transmitted word (adding redundancy, called the prefix bits in this text). The decoder extracts the inversion index from these additional bits, and uses it to rectify the inverted portion of the word. In all constructions the RDS of the entire sequence is exactly zero at the end of each word. Knuth proposes the following prefix bit schemes:

1) Simple Parallel Scheme: The inversion index is encoded in a balanced prefix. The number of prefix bits $(n - k)$ must satisfy (4) to guarantee that all possible inversion indices can be encoded in the balanced prefix.

$$\left(\frac{n - k}{(n - k)/2}\right) \geq k$$ (4)

2) Serial Scheme: Any disparity in the prefix bits is compensated by corresponding disparity in the remainder of the codeword, so that the overall disparity for the codeword is zero. Knuth shows that this requires only $\log(k)$ prefix bits for a $k$ bit source word.

For a given source word, there are often multiple inversion points which result in a balanced codeword. The Knuth code always selects the first valid index, resulting in a non-uniform distribution of the inversion index, which [9] attempted to exploit (unsuccessfully, but a reduced redundancy could be obtained through an auxiliary data technique). [10] presented a method to encode a balanced prefix in a more efficient manner, without the need for look-up tables. In [5], the fact that multiple inversion points exist were exploited by creating a multi-mode code, which applies the MSW criterion to each possible zero-disparity codeword, to determine the optimal codeword. About 3dB better DC suppression could be obtained between selecting the worst MSW candidate and the best MSW candidate. This scheme is referred to as modified Knuth.

III. PROPOSED CODE CONSTRUCTION: “MULTI-MODE KNUTH”

The new proposed construction is based on Knuth’s balanced codes, but with a number of changes, which are in line with the generation of a multi-mode selection set:

The constraint that the RDS must be zero after each codeword is removed. Also, the prefix bits are not required to be balanced, and represent the inversion index directly. Instead of considering each bit position of the source word for the inversion point, a more course-grained approach is allowed (see description of granularity below).

For each word added to the selection set, the polarity switch method is applied to the candidate word: a “one” is added to the front of the codeword, and both this new codeword as well as its complement are then used as candidates for the selection. The resulting scheme satisfies the sufficient requirements of a multi-mode code in which the RDS can be controlled, provided that a suitable selection criterion is used. In this paper, the MSW criterion is applied.

Going forward, this scheme is referred to as Multi-Mode Knuth, or MMK($n, k$), parameterized with codeword size $n$ and source word size $k$, as conventional for block codes.

As an example, consider an ($n, k$) = (20, 16) configuration. Disregarding the polarity switch bit, there are 3 index bits, $l$, as per (5), to encode the number of inverted bits.

$$l = (n - k) - 1$$ (5)

If the first 6 source bits are inverted, then the index bits will be binary $v = 011$, or decimal $v = 3$. The index bits in this
was implemented for simulations. and implementation simplicity, only this idealized Knuth code that any improvement gained on this "idealized" Knuth scheme than that of a correctly implemented prefix scheme, meaning bits of the word to obtain a zero-disparity codeword, is better

$$g = \frac{k}{2^l}$$  \hspace{1cm} (6)

In this paper, we only consider cases where \( k \) is divisible by \( 2^l \). The number of bits to invert, \( v' \), is then defined as:

$$v' = vg$$  \hspace{1cm} (7)

Consider the example source word 0111011101001100. A candidate codeword, with \( v' = 6 \), would then be:

$\begin{array}{c|c}
\text{bit} & \text{code} \\
\hline
1 & 011 \\
& 100010 \\
& 1101001100 \\
\end{array}$

Note that all possible values of \( v \) are considered in the selection set, both inverted and non-inverted.

To decode, the first (polarity switch) bit is inspected. If 0, the entire codeword is inverted. The next \( l \) bits are inspected to determine \( v' \), then \( v' \) is obtained with (7). After first removing the prefix bits, then inverting the first \( v' \) bits, the original source word results.

By increasing the number of prefix bits (lowering the code rate), the granularity is reduced, giving more fine-grained control over the number of inverted bits in a codeword than the original Knuth construction, resulting in a larger multi-mode selection set. This enables trading off performance for code rate. Another benefit is that no look-up table is needed to map prefix bits to inversion indices, because the inversion point is directly encoded into the prefix bits, scaled by the granularity.

IV. SIMULATIONS

For each code, the maximum absolute RDS found during the simulation at any point, \( |RDS|_{max} \), is recorded, although this makes no guarantee about the absolute maximum that may occur for worst-case source sequences, as discussed in [8]. Code redundancy \( 1 - R \) and the sum variance calculated from simulation are used as parameters to compute the efficiency \( E \) with (3).

Power spectrum measurements are obtained by simulation of \( > 400e6 \) source bits (generated with equal probability for 1 and 0, using Mersenne Twister PNG), calculating a \( 2^{20} \) point DFT every \( 2^{20} \) bits, then folding the squared magnitude components of all these DFT results together using Bartlett’s method. The large size of the DFT is necessary to obtain reasonable resolution down to normalized \( 10^{-4} \) bits/sec.

It was shown in [5] that the DC-suppression obtained by simply simulating words of length \( n \), and inverting the initial bits of the word to obtain a zero-disparity codeword, is better than that of a correctly implemented prefix scheme, meaning that any improvement gained on this “idealized” Knuth scheme also implies an improvement on the original. For this reason, and implementation simplicity, only this idealized Knuth code was implemented for simulations.

![Fig. 1. Spectral Performance for \( k = 16 \)](image1)

![Fig. 2. Spectral Performance for \( k = 32 \)](image2)

Power spectra for source word sizes \( k = 16, 32 \) and 64, are given in Figures 1, 2 and 3. Each includes a comparison of MMK with the idealized serial Knuth scheme and the “improved Knuth” scheme from [5], with different numbers of prefix bits for MMK. The parallel Knuth scheme is not shown, as its performance is only slightly worse than the serial scheme, with a lower code rate.

If only 2 prefix bits are used, MMK’s performance is clearly worse than the existing Knuth schemes. However, with 4 or more prefix bits, MMK outperforms the other schemes. For \( k = 32 \) and 64, better DC suppression is obtained with less redundancy, for 4 or more index bits. Numerical results are given in Table I. MMK(20,16) outperforms the serial Knuth scheme by 3.71dB, and the improved scheme by 3.04dB. MMK(70,64) outperforms the same two prior schemes by 8.8dB and 7.13dB, respectively. The general pattern for MMK,
is that a lower code rate (more prefix bits) improves DC suppression.

The efficiencies of MMK, for different $k$, are shown as a function of prefix bits in Figure 4. For a low number of prefix bits, the efficiencies are comparable over the different source word lengths. As the prefix bits increase, shorter source word lengths become more efficient. The peak represents the case where the maximum possible prefix bits are present for $k = 16$, where granularity is 1, which holds when $k = 2^i$. For longer source word lengths, e.g. 64, the same condition does not result in a peak in efficiency.

V. MMK EFFECT ON BIT-ERRORS FOR BSC

The effect of MMK on a binary symmetric channel (BSC) with bit-error probability $p$ is analysed. Since MMK encodes a potentially large change to the source word when creating a codeword, any errors that affect the prefix bits will have a significant effect during decoding, so the average bit-error rate (BER) is expected to increase.

A. BER Estimation

The approach followed is to estimate the expected number of error bits $q$ that will occur, on average, in a $k$ bit word after the decoding step. Then, $\frac{q}{k}$ then gives the expected BER for a larger bit-sequence consisting of many words.

We consider two cases:

1) Polarity Bit Correct: If the polarity bit is not affected by the channel (with probability $(1 - p)$), then the remaining bits are left intact. In this case, we consider what happens if 1 or more of the $l$ index bits are affected. This results in a change in $v$ after decoding, amplified by a factor $g$, thus changing the number of bits that the decoder inverts. The most dominant effect we can expect with small $p$ is that a single index bit is affected. The number of expected errors, due to a single index bit error, is:

$$q_1 = \sum_{i=0}^{l-1} p(1-p)^i (l-1)^2 g$$

(8)

The probabilities for specific numbers of inversions out of $l$ index bits are:

$$P(0) = (1-p)^l$$

(9)

$$P(1) = lp(1-p)^{l-1}$$

(10)

$$P(>2) = 1 - P(0) - P(1)$$

(11)

Further, if 2 or more index bit errors occur (very low probability), we approximate that $k/2$ bits are decoded incorrectly, so the number of expected errors contributed due to 2 or more inverted index bits is:

$$q_{>2} = P(>2) \times \frac{k}{2}$$

(12)

Lastly, all the source bits are affected by the channel with probability $p$. Since $p$ is small, and affects the encoded source

| Code          | $(n, k)$ | PSD at $10^{-3}$ | $s^2$ | $E$ | $|RDS|_{max}$ |
|---------------|----------|-----------------|-------|-----|--------------|
| Ser.Knuth     | (20,16)  | -28.12          | 6.88  | 0.17| 16           |
| Impr.Knuth    | (20,16)  | -29.79          | 2.90  | 0.40| 10           |
| MMK           | (18,16)  | -23.81          | 5.19  | 0.40| 18           |
| MMK           | (20,16)  | -31.83          | 2.27  | 0.51| 12           |
| MMK           | (21,16)  | -35.55          | 1.64  | 0.60| 10           |
| Ser.Knuth     | (38,32)  | -22.55          | 7.25  | 0.20| 16           |
| Impr.Knuth    | (38,32)  | -24.37          | 5.34  | 0.28| 16           |
| MMK           | (34,32)  | -18.36          | 10.04 | 0.39| 24           |
| MMK           | (36,32)  | -26.79          | 4.20  | 0.50| 18           |
| MMK           | (38,32)  | -31.76          | 2.76  | 0.53| 14           |
| Ser.Knuth     | (70,64)  | -17.40          | 13.25 | 0.19| 24           |
| Impr.Knuth    | (70,64)  | -19.09          | 9.67  | 0.26| 22           |
| MMK           | (66,64)  | -12.93          | 19.68 | 0.39| 36           |
| MMK           | (68,64)  | -21.29          | 8.09  | 0.49| 24           |
| MMK           | (70,64)  | -26.22          | 5.41  | 0.50| 18           |
| MMK           | (71,64)  | -28.38          | 4.72  | 0.50| 17           |
bits in all cases, we assume simplistically that the channel independently contributes $pk$ errors to each decoded word.

2) Polarity Bit Inverted: If the polarity bit is inverted by the channel (prob. $p$), then the index bits and source bits are incorrectly inverted during decoding, resulting in a different error pattern. Because the index bits are inverted, $v$ moves some distance. If all possible values of $v$ are equiprobable, the distance is $\frac{k}{2}$, and the average number of errors is $k - \frac{k}{2} = \frac{k}{2}$. The effects of all further errors tend to cancel out, on average. However, all values of $v$ are not equiprobable, with very low and very high $v$ being selected less frequently, resulting in a slight increase in errors. From distributions of $v$ obtained via simulations for up to 10 prefix bits, the following empirical formula was found to be a good approximation:

$$q_{pol} = (0.5 + \frac{l + 2}{100})k$$ (13)

The complete BER estimation formula is then:

$$p_{MMK_{approx}} = \frac{1}{k} \left[ (1 - p)(q_1 + q_{22} + pk) + \frac{p_{pol}}{q_{pol}} \right]$$ (14)

B. BER Simulations

The simulated bit-error results of MMK with $k = 16$ on a BSC are shown in Figure 5, for different numbers of prefix bits. Other source word lengths are not displayed, because they give very similar results. Numerical results, comparing simulated results with (14), are given in Table II. The BER of the channel is multiplied by a factor between approximately 2 and 2.6. The estimation formula (14) is accurate to within 20%, mostly within 2%. On a logarithmic scale, the BER performance of all codes is almost identical. A simple, safe and accurate empirical estimate derived from the simulation results would be to multiply the channel BER by a factor of 2.6.

VI. CONCLUSION

It was shown that the principles in the original Knuth balancing approach can be adapted to a multi-mode code, which is more flexible in terms of code rate than the original scheme. Simulations show that the new code has better DC suppression characteristics than the original Knuth scheme, sometimes even at a higher code rate. The scheme increases BER for a binary symmetric channel.

ACKNOWLEDGMENT

We thank Prof. Ivan Fair of the University of Alberta, Canada, for many helpful observations.

REFERENCES


