MultiMedia Compact Disc: Channel Coding and System Requirements 4059 (D-8)

Kees A. Schouhamer Immink
Philips Research Laboratories
5656 AA Eindhoven, The Netherlands

Presented at the 99th Convention 1995 October 6–9 New York

AES

AN AUDIO ENGINEERING SOCIETY PREPRINT

MultiMedia Compact Disc: Channel Coding and System Requirements

Kees A. Schouhamer Immink,
Philips Research Laboratories, 5656 AA Eindhoven, The Netherlands.
e-mail: immink@prl.philips.nl

July 3, 1995

The MultiMedia Compact Disc is a proposal for a new optical recording medium with a storage capacity five times higher than the conventional Compact Disc. The major part of the capacity increase is achieved by the use of optics, shorter laser wavelength and larger numerical aperture, that reduces the spot diameter by a factor 1.5. The track formed by the recorded pits and lands as well as the track pitch can be reduced by the same factor. The storage capacity is further increased by a complete redesign of the logical format of the disc including a more powerful error correction (CIRCPlus) and recording code (EFMPlus). In our paper, we will outline the channel coding and the related system requirements.

1 Introduction

The Compact Disc (CD), introduced about fifteen years ago [1], has become a successful medium for the distribution and storage of digital information. It is anticipated that its storage capacity, 680 MByte, will be insufficient for future graphics-intensive computer applications and high-quality digital video programs. An extension of the Compact Disc family, the *MultiMedia Compact Disc (MM-CD)*, is a proposal for a new optical recording medium with a storage capacity five times higher than the conventional Compact Disc. The major part of the capacity increase is achieved by the use of optics, shorter laser wavelength and larger numerical aperture, that reduces the spot diameter by a factor 1.5. The track formed by the recorded pits and lands as well as the track pitch can be reduced by the same factor. Provision is made for two disc types: single and dual layer. The single-layer disc can hold 3.7 GByte which amounts to five times more data than an audio CD. Typically, this is room for 135 minutes of wide-screen (16:9 aspect ratio) video at MPEG-2 quality accompanied by multiple audio and subtitle channels.

In the Compact Disc system, a concatenation of two codes, namely EFM (Eight-to-Fourteen Modulation) and CIRC (Cross Interleaved Reed Solomon Code) is used. CIRC is used for correction and detection of erroneously retrieved information, while EFM is used for transforming the digital audio bit stream into a sequence of binary symbols, called *channel bits,* which are suitable for storage on the disc [2]. Not only are the EFM and CIRC coding schemes useful for the Compact Disc for which they have been designed, but they have been extensively employed in a large variety of digital audio players and home-storage products such as CD-ROM, CD-I, and MiniDisc. The storage capacity of the MM-CD is also increased by a complete redesign of the logical format of the disc including a more powerful error correction (CIRCPlus) and recording code (EFMPlus). We start with a brief description of the Multimedia CD and the newly developed recording format.

2 System description

The main parameters of the MM-CD are listed in Table 1. Note that mechanical specifications such as disc thickness, outer diameter, and center hole diameter of the MM-CD are equal to those of CD, allowing full backward compatibility.

Table 1: Main parameters of MultiMedia CD vs CD-ROM.

	MultiMedia CD	CD-ROM
read-out wavelength (nm)	635	780
reference NA	0.52	0.45
disc diameter (cm)	12	12
disc thickness (mm)	1.2	1.2
layers	single or dual layer	single layer
data capacity (GByte)	1 layer: 3.7	mode I: 0.68
	2 layer: 7.4	mode II: 0.78
reference scanning speed (m/s)	4	1.2
reference channel bit rate (Mbit/s)	26.6	4.32
min. pit (or land) length (μm)	0.451	0.85
track pitch (μm)	0.84	1.6
recording code	EFMPlus	EFM
sector size (bytes)	2048	2048
error correction	CIRCPlus	CIRC
max. user bit rate @ ref speed (Mbit/s)	11.2	1.41

By employing a red laser at 635 nm wavelength and a numerical aperture (NA) of 0.52, the read-out resolution, and thus the physical scaled density of the disc is doubled. The subsequent halving of the track pitch and near-halving of the (tangential) pit length per bit, increase the actual physical data density by a factor of 3.5. The nominal read-out reference speed goes up from 1.2 to 4 meters per second. The user capacity of the single-disc is 3.7 GByte, and the dual-layer disc increases the data capacity to 7.4 GBytes. The principle of operation of the dual-layer disc is shown in Figure 1.

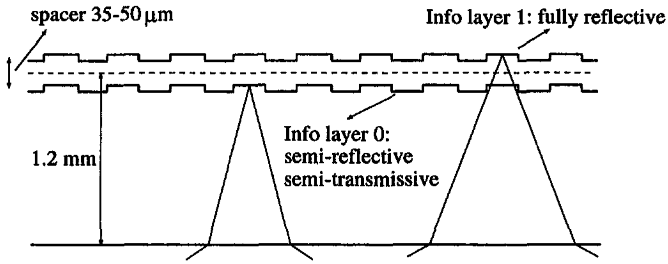

Figure 1: Dual-layer reading principle diagram.

The dual-layer CD is similar to the standard CD. It adopts the same molded substrate for the first data layer. The second layer is made by overlaying a fully-reflective aluminium layer on a partially reflective aluminium layer. The second layer is applied using the same principles as the 2P (Photo Polymerization) process. Focal plane servos are used to switch from one layer to the other. Continuous play of video signals is obtained by reading outwards one layer, then inwards on the other.

As in CD-ROM, the user data is organized into sectors. Under MM-CD format rules, sectors of 2048 user bytes are translated into 2436 bytes (2048 user + 56 sync + 24 header + 308 erco) which are in turn translated into 16 × 2436 channel bits. Thus, one user bit is translated into 4872/2048 = 2.38 channel bits. In conventional audio CD, an audio bit is translated into 588/192 = 3.05 channel bits [3], and we conclude that the 'format efficiency' of the MM-CD is improved by 29 %. The new format is even 47 % more efficient with respect to CD-ROM, which uses a 'third' error correction layer. Even though consuming a significantly lower data overhead, the error correction code, CIRCPlus, can cope with longer bursts and burst sequences, and more random errors.

Table 2: Comparison of format efficiency.

Parameter	*Audio CD*	*MultiMedia CD*	*Gain (%)*
synchronization (% per sector)	4	1	3
error corr. code (% per sector)	33	15	18
channel code (1/rate)	17/8	16/8	6
subcode (% per sector)	3	-	3
		Total	30

A comparison of the format efficiency of the MultiMedia CD compared with conventional CD audio is provided in Table 2.

3 Error correction code: CIRCPlus

The CD system uses the Cross Interleaved Reed Solomon (CIRC) error correction code. For the MM-CD a more powerful error correction, called *CIRCPlus*, has been developed [4]. Enhanced error correction capability is required for several reasons. Firstly, the increased physical density implies that physical imperfections affect more bits. It is further anticipated, since the system margins are much tighter, that the random error rate of MM-CD is larger than that of conventional CD. Secondly, as we cannot rely anymore on the concealment techniques used in conventional audio CD, the reliability of the decoded data must be much higher. As MM-CD is a true MultiMedia disc its data integrity is comparable to that of computer data.
CIRCPlus, like CIRC, uses a combination of two Reed-Solomon (RS) codes denoted by C_1 and C_2 [5]. In CIRC, C_1 is [32,28] and C_2 is a [28,24] code, where $[n, k]$ denotes a code with k input and n output bytes. The redundant $n - k$ bytes are generated under the rules of the RS code. The rate of CIRC, is 24/32=3/4. In other words, under CIRC rules one redundant byte is added to three user bytes. In CIRCPlus, the C_1 and C_2 codes are significantly longer than in CIRC, namely [170,162] and [170,156]. As a result, the rate of CIRCPlus is much higher than that of CIRC, namely 156/170.

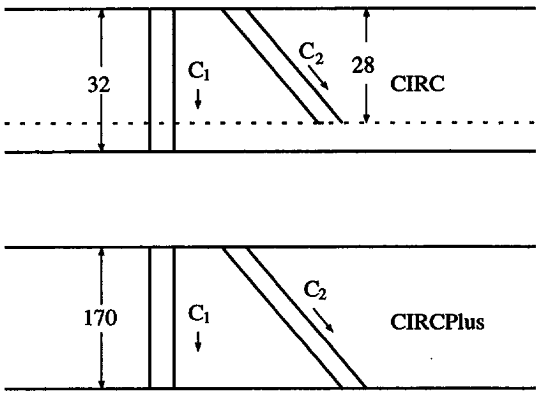

Figure 2: Block diagrams of CIRC and CIRCPlus encoder.

Figure 2 depicts schematically the principle of operation of the CIRC and CIRCPlus codes. The combination of C_1 and C_2 is designed in such a way that in CIRCPlus the parities generated by C_1 are also taken into account in code C_2. This has the advantage that the error correcting capability of the codes in tandem is improved relative to the CIRC code where this 'double' check is absent. Figure 3 shows the CIRCPlus decoder configuration. In CIRCPlus, a second C_1 decoding step, following the C_1 and C_2 decoding steps, can be made to improve the reliability. Note that CIRCPlus has the virtue of fast data output, i.e. header information, after the first C_1 decoding step, since data are in correct (user) order at that stage.

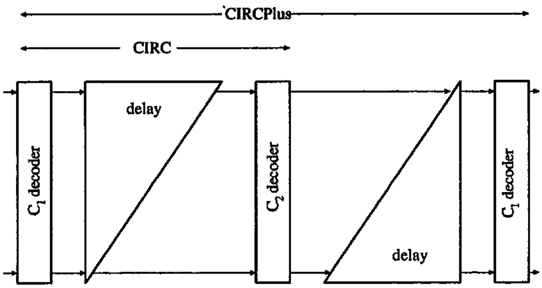

Figure 3: Block diagrams of CIRC and CIRCPlus decoder.

The maximum correctable burst length is approximately 500 bytes for CIRC while it is 2200 bytes for CIRCPlus. CIRCPlus is capable of reducing a random input error rate of 2×10^{-2} to a data error rate of 10^{-15}, which is a factor of ten better than in CD.

4 EFM recording Code

The EFM code is a member of the family of *dc-free runlength-limited codes.* The number of sequential like symbols in a (binary) sequence is known as *runlength.* A runlength-limited sequence is a sequence of binary symbols characterized by two parameters, $T_{\min} = (d+1)$ and $T_{\max} = (k+1)$, which stipulate the minimum and maximum runlength, respectively, that may occur in the sequence. The parameter d controls intersymbol interference when the sequence is transmitted over a bandwidth-limited channel. The maximum runlength parameter k ensures adequate frequency of transitions for synchronization of the read clock. There are two reasons why EFM suppresses the low-frequency components. In the first place, the servo systems for track following and focusing are controlled by low-frequency signals, so that low-frequency components of the information signal could interfere with the servo-systems. The second reason is that low-frequency disturbances resulting from fingerprints on the disc can be filtered out without distorting the data signal itself.

Under EFM rules, the data bits are translated eight at a time into fourteen channel bits, with minimum runlength parameter $d = 2$ and a maximum runlength parameter $k = 10$ channel bits (this means at least 2 and at most 10 successive 'zeros' between successive 'ones'.

Table 3: Part of the EFM coding table.

Data	*Code*	*Data*	*Code*
100	01000100100010	110	10010010000010
101	00000000100010	111	00100000100010
102	01000000100010	112	01000010000010
103	00100100100010	113	00000010000010
104	01001001000010	114	00010001000010
105	10000001000010	115	00100001000010
106	10010001000010	116	01001000000010
107	10001001000010	117	00001001001000
108	01000001000010	118	10010000000010
109	00000001000010	119	10001000000010

Part of the EFM coding table is presented in Table 3, which shows the decimal representation of the 8-bit source word (left column) and its 14-bit channel representation. It should be appreciated that the codewords are described in NRZI (nonreturn-to-zero inverted) notation, which means that a 'one' represents a transition of either positive or negative polarity, and a 'zero' represents the absence of a transition. As a result, all information written on the Compact Disc is contained in the positions of the transition pit/land or land/pit. Three bits, called *merging bits*, are used to ensure that the runlength conditions continue to be satisfied when the codewords are cascaded. If the runlength is in danger of becoming too short, we choose 'zeros' for the merging bits; if it is too long we choose a 'one' for one of them. If we do this, we still retain a large measure of freedom in the choice of the merging bits. This freedom is used for minimizing the low-frequency content of the signal. The measure of the low-frequency content is the *running digital sum* (RDS), which is the difference between the totals of pit and land lengths accumulated from the beginning of the disc. The system now opts for the merging combination that makes the RDS at the end of the codeword as close to zero as possible. The Power Spectral Density (PSD) function of conventional EFM has been obtained by computer simulation. Results are plotted in Figure 4.

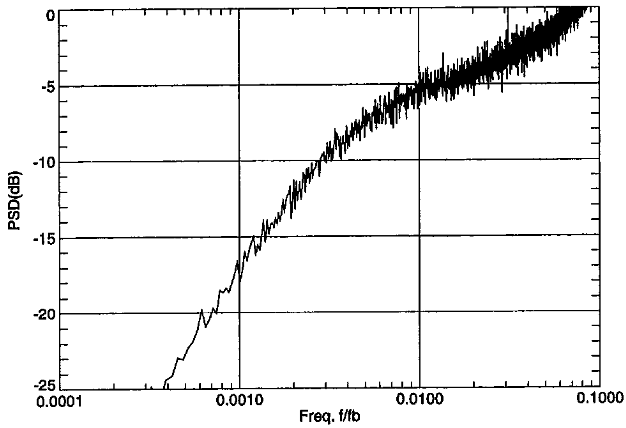

Figure 4: Spectrum of conventional EFM. Both axes are normalized for fixed user bit rate f_b.

5 Description of the EFMPlus encoder

Van Uijen and Spruit [6] have studied the performance of modulation codes and detection techniques in various recording media. They concluded that significant improvements in information density cannot easily be made. The aim of our investigations was therefore the design of a new 'EFM-like' code having a better rate, i.e, quotient of number of bits entering the encoder and number of bits leaving the encoder, than the conventional one. The most important design issue is that critical parameters such as lf-content and timing should definitely not be worsened. We consider these parameters as extremely critical as they affect the servos and the timing recovery which are the Achilles' heels of the system.

The EFMPlus encoder is much more involved than the relatively simple EFM encoder. The principle of operation of the EFMPlus encoder can be represented by a finite-state sequential machine with an 8-bit input, a 16-bit output, and four states which are functions of the (discrete) time. We say that the states are connected by edges, and the edges, in turn, are labeled with tags called *words*. A word in our context is a 16-bit sequence that obeys the prescribed ($d = 2, k = 10$) constraints. Each of the four states is characterized by the type of words that enter, or leave, the given state. The states and word sets are characterized as follows.

- Words entering State 1 end with $\{0, 1\}$ trailing 'zero';
- Words entering State 2 and 3 end with $\{2, \ldots, 5\}$ trailing 'zeros';
- Words entering State 4 end with $\{6, \ldots, 9\}$ trailing 'zeros'.

The words leaving the states are chosen in such a way that the concatenation of words entering a state and those leaving that state obey the ($d = 2, k = 10$) channel constraints. For example, words leaving State 1 start with a runlength of at least two and at most nine 'zeros'. In an analogous manner, we conclude that words leaving State 4 start with at most one 'zero'. Obviously, the sets of words leaving State 1 or 4 have no words in common. Words emerging from State 2 and 3 comply with the above runlength constraints, but they also comply with other conditions. Words leaving State 2 have been selected such that the first (msb) bit, x_1, and the thirteenth bit, x_{13}, are both equal 'zero'. In a similar fashion, words leaving State 3 are characterized by the fact that the 2-tuple $x_1 x_{13}$ does not equal '00'. The attributes of the four states defined above guarantee that any walk through the graph, stepping from state to state, produces a ($d = 2, k = 10$)-constrained sequence by reading the words tagged to the edges that connect the states. Given the above definitions, it is elementary to compute the number of edges (or words) that leave each of the four states. With a simple computer program we find that from each of the states at least 351 words are leaving. An encoder can now be constructed by assigning a source word to each of the 351 edges that leave each state. Excess edges are removed. As a result, each edge in the graph has now two labels, namely a 16-bit word and a source word numbered from 0 to 350. Given the source word and the encoder state, the encoder will transmit the word tagged to the same edge as the source word at hand. It is immediate which codeword will be sent, and also which state will be next. A sequential input will define a walk in the graph, and as said, this walk generates a ($d = 2, k = 10$)-constrained sequence. As a result, the above finite-state encoder graph is a ($d = 2, k = 10$) RLL encoder that freely accommodates 351 source words. We ignore for a while a very important item, namely whether or not, and how, an inverse operation can be found, i.e., given the received string of 16-bit words whether or not, and how, the string of source words can be uniquely decoded at the receiver's site. The encoder requires accommodation for only 256 source words. The excess, 95, words will be used for controlling the low-frequency power (see later). We will first take a closer look at the details of the encoder graph.

Table 4: Part of the EFMPlus coding table.

i	$h(i,1)$, $g(i,1)$	$h(i,2)$, $g(i,2)$	$h(i,3)$, $g(i,3)$	$h(i,4)$, $g(i,4)$
0	0010000000001001, 1	0100000100100000, 2	0010000000001001, 1	0100000100100000, 2
1	0010000000010010, 1	0010000000010010, 1	1000000100100000, 3	1000000100100000, 3
2	0010000100100000, 2	0010000100100000, 2	1000000000010010, 1	1000000000010010, 1
3	0010000001001000, 2	0100010010000000, 4	0010000001001000, 2	0100010010000000, 4
4	0010000010010000, 2	0010000010010000, 2	1000000100100000, 2	1000000100100000, 2
5	0010000000100100, 2	0010000000100100, 2	1001001000000000, 4	1001001000000000, 4
6	0010000000100100, 3	0010000000100100, 3	1000100100000000, 4	1000100100000000, 4
7	0010000001001000, 3	0100000000010010, 1	0010000001001000, 3	0100000000010010, 1
8	0010000010010000, 3	0010000010010000, 3	1000010010000000, 4	1000010010000000, 4

5.1 Encoder graph

The encoder graph is defined in terms of three sets: the inputs, the outputs and the states, and two logical functions: the *output function* and the *next-state function*. The specific codeword, denoted by $\boldsymbol{x}_t$, transmitted by the encoder at instant t is, of course, a function of the source word $\boldsymbol{b}_t$ that enters the encoder, but depends further on the particular state, s_t, of the encoder. Similarly, the "next" state at instant $t+1$ is a function of $\boldsymbol{x}_t$ and s_t. The output function $h(.)$ and the next-state function $g(.)$ can be succinctly written as

$$\boldsymbol{x}_t = h(\boldsymbol{b}_t, s_t)$$

$$s_{t+1} = g(\boldsymbol{b}_t, s_t).$$

Both the output function $h(.)$ and the next-state function $g(.)$ are described by four lists with 351 entries. A part of the output function and the next-state function is listed in Table 4. Table 4 has an entry column that describes the source (input) word i by an integer between '0' and '255'. The table also shows $h(i,s)$ the 16-bit output to a particular input i when the encoder is in one of the four states s. The words are written in NRZI notation. It is assumed that the msb bit is transmitted first. The 3rd, 5th, 7th, and 9th columns show the next state function $g(i,s)$. An example may clarify the principles of operation of the encoder. Let the encoder graph be initialized at State 1 (it will be shown in a moment, that the initial state is irrelevant for the decoding operation), and let further the source sequence be '8', '3', '4'. The response to input '8', while being in State 1, equals $h(8,1) =$ (see Table 4) '0010000010010000'. The new state becomes $g(8,1) = 3$. As a result, the response to input '3', while now being in

State 3, is '0010000001001000'. In the next clock cycle, the encoder state becomes $g(3,3) = 2$. From State 2 with the input equal to '4' we find from the table that the corresponding output is $h(4,2)$ ='0010000010010000'. Note that the responses to the distinct inputs '3' and '7' while in State 1 are the same, namely '0010000001001000'. In the next section, we will describe how this ambiguity can be resolved during decoding.

5.2 Decoding operation

Basically, knowledge of the encoder state at the receiver site is necessary to re-constitute the source word. This operation can be succinctly written as

$$\boldsymbol{b}_t = h^{-1}(\boldsymbol{x}_t, s_t).$$

Such state-dependent decoding is a technique that should be avoided as it usually entails a strong error propagation. In our case, the source words can be judiciously assigned to the various codewords in such a way that decoding of the channel representations can be uniquely accomplished without knowledge of the encoder state. On a few occasions, as we can observe in the table, two source words have the same channel representation(s). For example, source words '3' and '7' generate the same channel representation, '0010000001001000', when the encoder is in State 1. Evidently these words cannot be decoded by a sole observation of the 16-bit codeword. This ambiguity can be remedied by observing that the code was constructed in such a way that if the same two words do leave a given state, one of them goes to State 2 and the other to State 3. As the sets of words leaving State 2 or 3 have been chosen such that they they differ in the symbols at positions 1 and 13, words can be uniquely decoded by observing a 16-bit codeword plus the 1st and 13th bit of the upcoming codeword. In other words,

$$\boldsymbol{b}_t = h^{-1}(\boldsymbol{x}_t, x_{t+1,1}x_{t+1,13}).$$

Under EFM rules, it suffices to observe 14 of the 17 channel bits that constitute an EFM codeword. In contrast, decoding of the new code is done by a logic array that translates (16+2) channel bits into 8 bits.

5.3 Suppression of low-frequency components

The encoder defined above can freely accommodate 351 source words. In order to make it possible to use a unique 26-bit sync word, seven candidate words were barred, leaving a code size of 344. As we only need accommodation for 256 source words, the surplus 344-256=88 words can be exploited for minimizing the power at low frequencies. The suppression of low-frequency components, or dc-control, is done in the same vein as in the EFM code, namely by controlling the running digital sum (RDS). The 88 surplus words are used as an alternative channel representation of the source words 0,...,87. The full encoder is described by two tables called *main* and *substitute table*, respectively (see Figure 5).

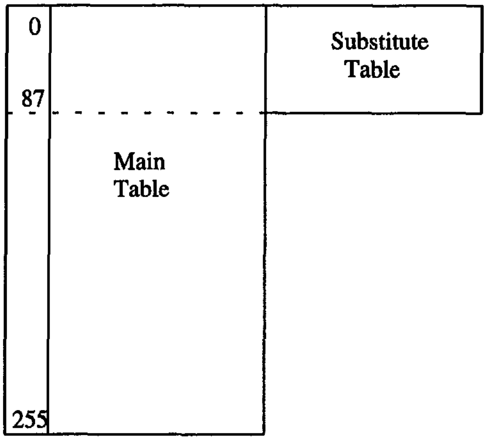

Figure 5: Block diagram of EFMPlus encoder.

The main table describes an encoder table of 256 inputs. The substitute table shows a similar table of 88 words which act as alternative representations of the source words 0,...,87 of the main table. The source words 0,...,87 can thus be represented by the designated entries of the main table or alternatively by the entries of the substitute table. The power spectral density of the new code has been computed by a computer program which simulated the encoder algorithm. Results are plotted in Figure 6. A 3 dB improvement can be gained by using a look-ahead strategy, that is decisions are made on the basis of (in this case) three consecutive source bytes. The look-ahead algorithm provides a better trade-off between long and short term success.

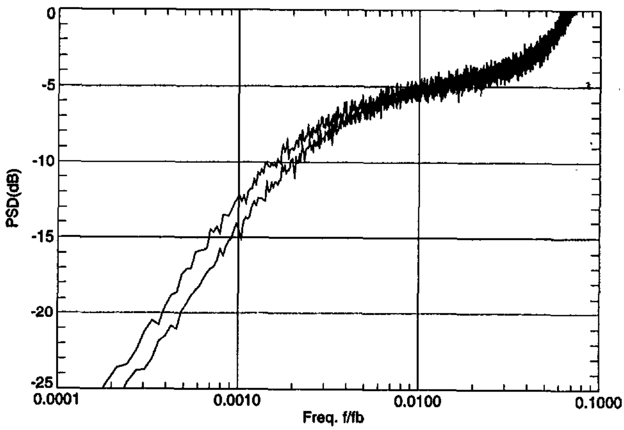

Figure 6: Spectrum of EFMPlus. The upper curve shows the spectral density without look-ahead while the lower curve shows the spectral density using three bytes look-ahead.

6 Conclusions

The MultiMedia Compact Disc is a proposal for a new optical recording medium with a storage capacity five times higher than the conventional Compact Disc. The user capacity of the single-disc is 3.7 GByte, and the dual-layer disc increases the data capacity to 7.4 GBytes. The major part of the capacity increase has been achieved by the use of optics, shorter laser wavelength and larger numerical aperture, that reduces the spot diameter by a factor 1.5. The track formed by the recorded pits and lands as well as the track pitch can be reduced by the same factor. We have outlined the system requirements of the MM-CD and the related channel coding. The conventional CIRC and EFM codes have been replaced by a more powerful error correction, CIRCPlus, and recording code, EFMPlus. As a result, the 'format efficiency' of the MM-CD relative to audio CD is improved by 29 % (47 % with respect to CD-ROM). Even though consuming a significantly lower data overhead, CIRCPlus can cope with longer bursts and more random errors. The strategy of EFMPlus is much more refined than the original

EFM. The saving in rate of EFMPlus offers a serviceable 6 % increase in capacity as compared with its predecessor, without in the least compromising the reliability of the servo systems or data recovery.

References

[1] L.B. Vries, K.A.S. Immink, J.B. Nyboer, H. Hoeve, T.T. Doi, K. Odaka, and H. Ogawa, 'The Compact Disc Digital Audio System: Modulation and Error correction', 67th AES Convention, no 1674 (h-8) New York, 1981.

[2] K.A.S. Immink, *Coding Techniques for Digital Recorders,* Prentice-Hall International (UK) Ltd., Englewood Cliffs, New Jersey, 1991.

[3] J. Watkinson, *The Art of Digital Audio,* Focal Press, London, 1988.

[4] C. Baggen and L. Tolhuizen, 'On the Diamand Code Construction', *Proceedings 16th Symposium on Information Theory in the Benelux*, pp. 123-126, 1995.

[5] S.B. Wicker and V.K. Bhargava, Edrs, *Reed-Solomon Codes and Their Applications*, IEEE Press, 1994.

[6] C.M.J van Uijen and J.H.M. Spruit, 'Performance of Modulation Codes in Various Optical Recording Media', *Japan Journal Appl. Phys.,* vol. 31, pp. 670-679, Feb. 1992.