**RC24173 (W0703-082) March 16, 2007**
**Electrical Engineering**

# IBM Research Report

## Circuit Implementation of a dc-Balanced 7B8B Transmission Line Code

**Albert X. Widmer**
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

# Circuit Implementation of a dc-Balanced 7B8B Transmission Line Code

**Albert X. Widmer**

*IBM T.J. Watson Research Center*
*1101 Kitchawan Rd. Route 134, Yorktown Heights, NY 10598-0218*
*c/o: ritter@us.ibm.com*

## Abstract

A dc-balanced 7B8B baseband transmission line code is presented with details in the form of coding tables, coding equations, and a representative gate level circuit implementation. Less than 350 inverting type primitive logic gates are required for the encoder/decoder circuit arranged in logic paths at most seven deep. The circuits have been structured so pipe-lining can be used with modest overhead to reduce the logic depth to 6, 5, 4, or even 3 per stage. The encoded alphabet includes 7 control vectors recognizable as other than data, one of which can be used to generate a comma to mark the vector boundaries. The maximum run length is 6. The two vectors with all alternating ones and zeros are defined as control vectors, so for strings of data vectors, the run of alternating ones and zeros is limited to less than two vectors which limits the error recovery time of systems using differential encoding with Decision Feedback Equalization (DFE). Vectors which require selective bit changes for encoding and decoding are confined to dc-balanced disparity independent vectors which have no alternate representation. The code can be used alone or as a building block for a larger partitioned code such as 16B18B or 12B14B in combination with a 9B10B or 5B6B code.

## Table of Contents

# I. INTRODUCTION

Ref. 1 and 2 describe the principles for the construction of a 16B16B transmission code which is partitioned into a 9B10B and a 7B8B part. This report provides details in the form of coding tables, coding equations, and gate level circuit diagrams for the implementation of a similar 7B8B code which can be used either alone or in combination with other codes such as the 3B4B, 5B6B or 9B10B code. For applications such as high speed busses as described in Ref. 3, the compatibility with an 8-bit byte format is often not an advantage or irrelevant for very wide busses with dozens to hundreds of parallel lines. The higher coding efficiency and other features may outweigh the lower complexity of the traditional 8B10B code (Ref. 4, Ref. 5). For a better fit for these other applications, the codes of Ref. 1 and 2 have been slightly modified with minimal added complexity to help in the definition of suitable control and comma sequences for these other applications. No encoded data vector consists of a string of five 10 or five 01 bit patterns which limits the error recovery time of systems using differential encoding with Decision Feedback Equalization (DFE). These changes are also applicable for the 16B18B code, so a single set of 7B8B and 9B10B macros can be built for all applications.

The modifications allow also a much more efficient circuit implementation with less latency. The new 7B8B encoder and decoder circuit can be built with a total of fewer than 350 inverting type primitive logic gates arranged in logic paths at most seven deep. The circuits have been structured so pipe-lining can be used with modest overhead to reduce the logic depth to 6, 5, 4, or even 3 per stage.

For very high speed transceivers, clock rate ratios which are a power of two are sometimes preferred and the 7B8B is naturally compatible with such clock systems.

Various versions of 7B8B codes have been used by British Telecom (Ref. 6, Ref. 7) and Standard Telephones and Communications plc (Ref. 8). A coding table for one such version is listed in Ref. 9. It is not suitable for implementation with combinational logic elements.

Cattermole (Ref. 10) and Brooks/Jessop (Ref. 11) give a good introduction to this kind of line coding.

### Outline

The report starts with a description of the several sets of encoded vectors, represented by trellis diagrams, which constitute the 8B alphabet including suitable comma characters for various applications. A very brief description of the transmission characteristics of the code follows. Some useful notation is described and a full 7B8B encoding table is presented with the trellis diagrams and tables used for its construction. Then the logic equations for the encoding of each individual bit and the disparity control are developed based on tables. Similarly, logic equations are developed for decoding and error checks. The equations are the basis for the gate level logic circuit diagrams shown. The report ends with some suggestions for applications.

## II. DESCRIPTION OF THE 7B8B CODE

At all 8B boundaries, the running disparity can assume one of four values D=±1, or D=±3. Encoded vectors in this code are either balanced and disparity independent, balanced and disparity dependent, or they are disparity dependent with a disparity of ±2 or ±4. If the current running disparity is positive (+1 or +3), only disparity independent vectors or vectors with a required positive entry disparity may be entered and complementary rules apply for a negative running disparity. Almost half the source vectors are translated into a single balanced disparity independent encoded vector. All other 7B vectors are translated into one of a pair of complementary 8B vectors, respectively, according to the disparity rules above. All vectors requiring individual bit changes are mapped into balanced, disparity independent vectors.

### Notation

The bits of the uncoded 7B vectors are labelled with the upper case letters ‘*STUVWXY*’, and the bits of the coded 8B vectors are labelled with the lower case letters ‘*stuvwxyz*’.

Serial transmission is in alphabetical order starting with ‘s’.

## A. 7B8B Code Definition

The 7B8B code comprises a total of *135* code points with 202 coded 8B vectors as illustrated by the trellis diagrams of FIG. 1.

### 1) 68 Balanced 8B Vectors (FIG. 1A.1)

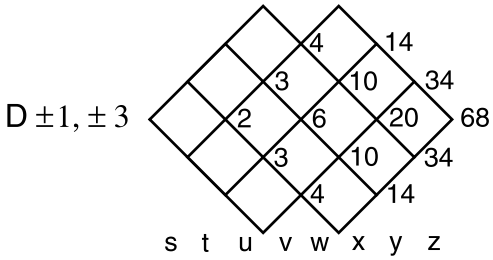


FIG. 1A.1

A set of *68 disparity independent, balanced* vectors is illustrated in FIG. 1A.1. The subset (68) of all possible 8B vectors (256) chosen is the set of balanced vectors with a run length of no more than three at the leading and trailing boundaries.

### 2) One disparity dependent, balanced, complementary vector pair

The code includes *one disparity dependent, balanced,* complementary vector pair as illustrated in FIG. 1A.2 with a leading and trailing run of four. It is assigned to the source vector D15 = 1111000.

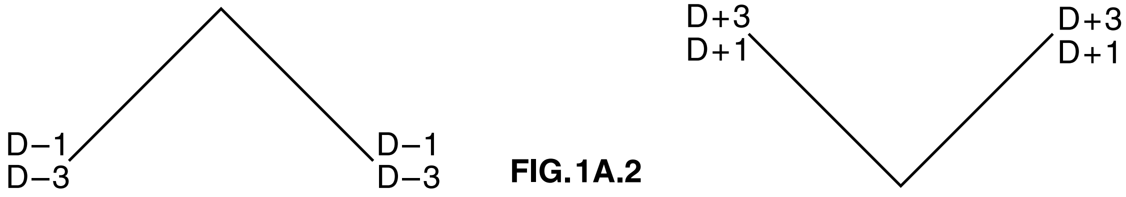


FIG. 1A.2

### 3) 2x48 8B Vectors with Disparity ±2 (FIG.1B)

A set of 48 8B vectors comprises all bit patterns with a disparity of +2, no more than three ones or two zeros at the front end and no more than two zeros or three ones at the trailing end. An exact complementary set of another 48 vectors has a disparity of −2.

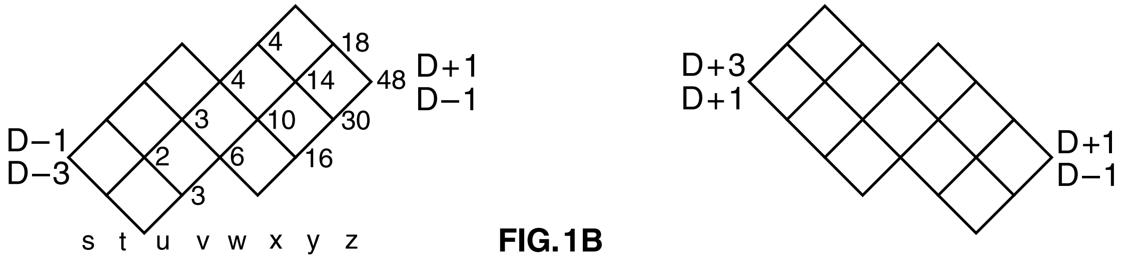


FIG.1B

### 4) 2x18 8B Vectors with Disparity ±4 (FIG.1C)

The set of twelve 8B vectors of FIG.1C.1 comprises all bit patterns with a disparity of +4, no more than three ones or one zero at the front end and one to three ones at the trailing end. An exact complementary set of another 12 vectors has a disparity of −4.

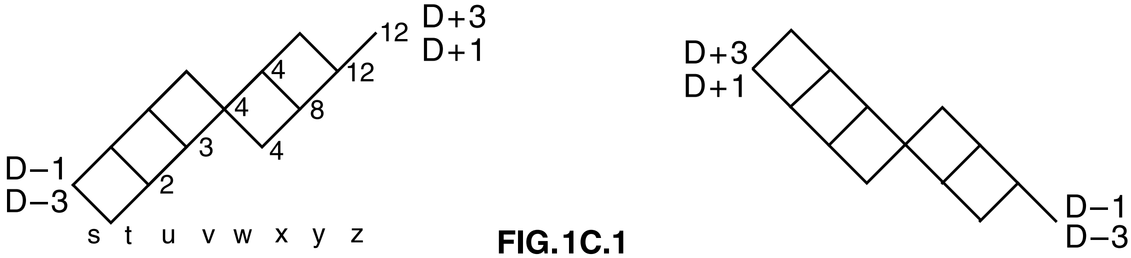


FIG.1C.1

FIG.1C.2 below illustrates a set of six vectors with a disparity of +4 and no more than two ones or one zero at the front and exactly one zero or 4 ones at the end. An exact complementary set of another 6 vectors has a disparity of −4. The leading part of the comma character for concatenated 8B vectors belongs to FIG. 1C.2.

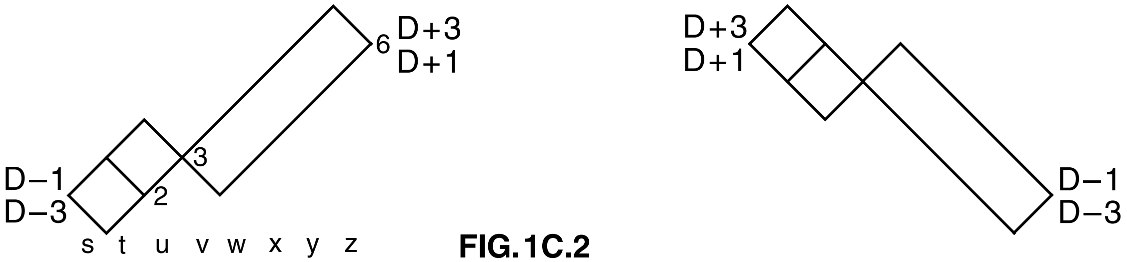


FIG.1C.2

### 5) Comma Characters for concatenated 7B8B Coding Blocks and for 16B18B Code

To generate a comma, two 8B blocks are required. For this purpose, the control character C126 with a run of six has been added. It is listed at the bottom of Table 1D. The control character C126 can be used to generate a singular comma consisting of a run length of six followed contiguously by a run of one and ending with a run of three of the same polarity as the leading run of six (**000000**1’**000** or **111111**0’**111**). Only the nine bold bits must be checked for synchronization. The comma is embedded in two blocks of eight coded bits and is illustrated for one polarity in FIG. 2. The second byte is taken from the group of balanced vectors of FIG. 1A.1. These vectors must be made disparity dependent if they follow C126 of Table 1D to obtain a comma sequence regardless of the running disparity.

| | | | | | |
|---|---|---|---|---|---|
| D7 | + | 11100001 | – | 00011110 | D120 |
| D23 | + | 11101000 | – | 00010111 | D112 |
| D39 | + | 11100100 | – | 00011011 | D95 |
| D71 | + | 11100010 | – | 00011101 | D63 |

**Comma Sequence embedded in two 8B Blocks**

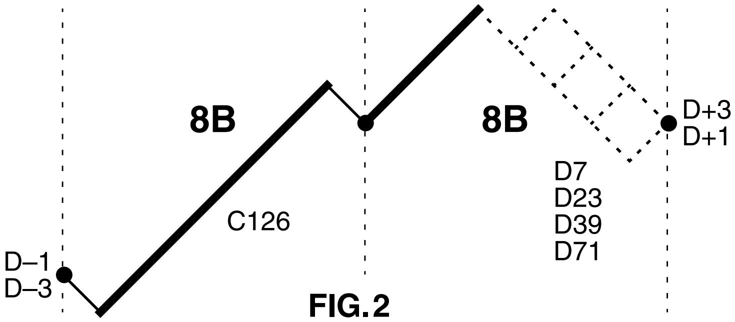


**FIG. 2**

The trailing 8B patterns are identical to the trailing vector of the 16B18B comma of Ref. 1 where C126 is replaced by C508 (0011111110/1100000001) from the 10B alphabet.

## B. Other Applications, 17B20B, 12B14B Code (FIG. 3)

Machine upgrades sometimes require serialization of parallel buses to deal with entry and exit congestion at the board level or other modular building blocks. These serial links are usually not based on neatly designed new serial architectures but must be based on existing bus structures which may not be modulo eight in width. To serve these requirements, it is useful to have a variety of code widths in the design arsenal and techniques to combine them into a wider structure. As an example, one application requires the efficient conversion of a 17-bit bus into serial form. This could be solved by two parallel 9B10B coders which would provide one bit of spare capacity in a 20-bit coded block. Another, perhaps simpler and adequate solution combines one 7B8B coder with two 5B6B coders taken from Ref. 4 or 5 to translate the 17 source bits into 20 coded bits suitable for serial transmission.

The resulting 17B20B code has a maximum run length of 6 and a digital sum variation of 10. The synchronizing sequence or comma can be defined as a run of 6, contiguously followed by a run of one and ending with run of 2 (**111111**0**11** or **000000**1**00**) as shown in FIG. 3. This sequence can be generated by C126 from the 8B alphabet followed by the balanced vectors D3, D11, or D19 from the 6B alphabet of Ref. 4 or 5. Again, the three balanced 6B vectors must be made disparity dependent if they follow C126. If the running disparity at the front of the 6B section is negative, they must be complemented as shown below.

|  |  |  |  |  |
|---|---|---|---|---|
| D3 | + 110001 | – 001110 | D28 |
| D11 | + 110100 | – 001011 | D20 |
| D19 | + 110010 | – 001101 | D12 |

It is obvious, that the same rules apply to a *12B14B code* which would be partitioned into a 7B8B code followed by a single 5B6B code.

**Comma Sequence embedded in a 8B Block and a 6B Block**

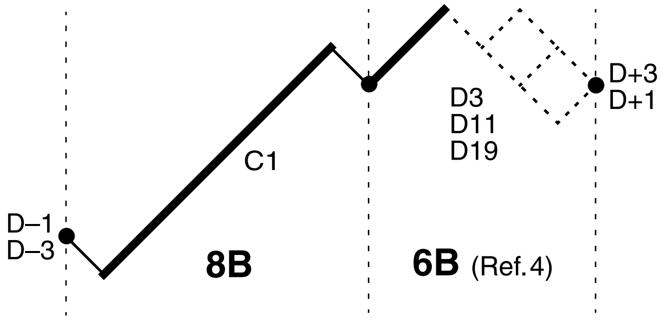


FIG.3

## C. Properties of the 7B8B Code (FIG.4)

The important characteristics of the code can be directly extracted from the trellis diagram of FIG.4 which also shows four possible configurations for the comma sequence. Using FIG.4 together with the trellis diagrams of FIG.1, one can verify that the comma sequence is singular, i.e. it cannot be reproduced in any other position relative to the vector boundaries neither within two 8B blocks nor across the 8B block boundaries.

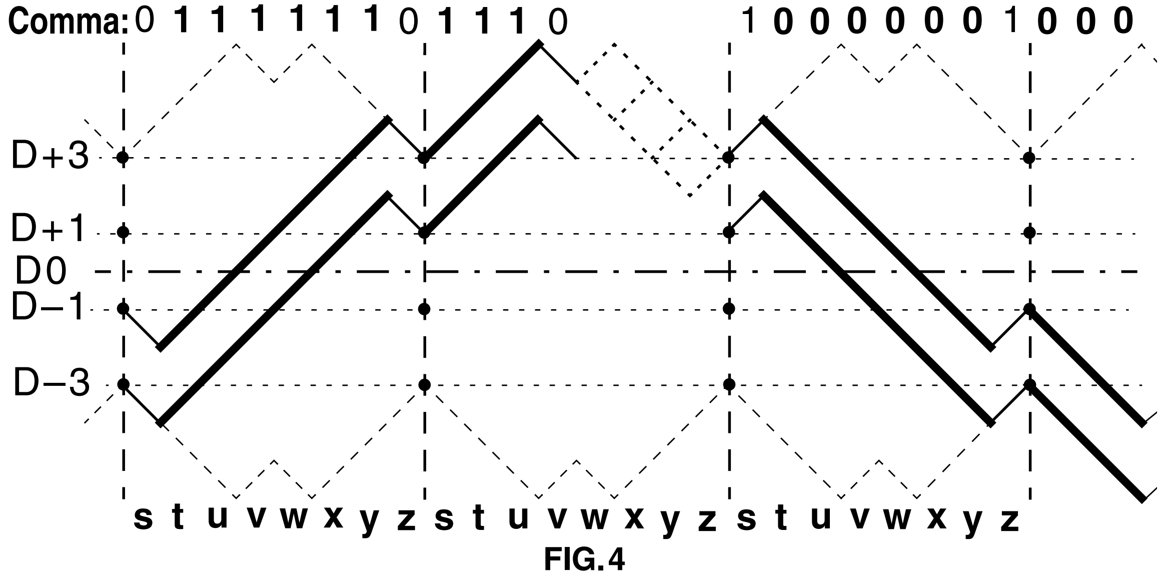

Comma: 0 1 1 1 1 1 1 0 1 1 1 0
1 0 0 0 0 0 0 1 0 0 0
D+3
D+1
D0
D–1
D–3
s t u v w x y z s t u v w x y z s t u v w x y z

FIG.4

### 1) Low Frequency Characteristics

The code is DC balanced. The maximum digital sum variation is 12. The normalized DC offset as defined in reference 1 is 4.75. As a point of reference, the offset value of 8B10B code is 1.9. The low frequency cut-off point for high pass filters must be located about 2.5 times lower than for Fibre Channel 8B10B code for equal eye closure. The low frequency wander can be reduced on a statistical basis by *scrambling the data before encoding*. 8B10B coded, scrambled data can operate with a 50% higher low frequency cut-off point than a coded worst case pattern. For 7B8B code, the gain from scrambling before encoding is expected to be more because there are more and larger low frequency components to randomize.

### 2) Control Characters

The 7B8B code provides seven control characters which are recognizable as other than data. One of the control characters (C126) is used to generate the singular comma sequence for instantaneous vector boundary synchronization and other signalling purposes. The comma sequence extends over 10 baud intervals and 9 of the coded bits must be monitored. The sequence requires two contiguous 8B vectors and as shown in FIG. 4. The comma sequence is followed by one of four different 4-bit trailing sequences.

### 3) Clocking and Synchronization Parameters

The maximum run length of the code is seven and no more than two contiguous runs of seven are possible (...0111–11110000–0001... or complement). The minimum transition density is two per 8B block for an indefinite length (–11110000–11110000– or complement).

## D. Encoding Table

### 1) Design Principles

101 of the 135 encoded primary vectors are obtained by appending a bit with a default value of zero. An alternate, equivalent code can be constructed by choosing complementary values for the appended bit and the vector sets. All 34 vectors with individual bit changes other than full vector complementation are disparity independent with an appended bit value of one. Only 25 vectors require any changes in one to four individual source bits. This arrangement has the advantage that full vector complementation and bit encoding and decoding can be executed independently of each other in parallel.

### 2) Coding classifications for bit mapping (FIG.5)

- The first capital letter B, D, or F indicates the disparity of the coded vector:

  B indicates a **B**alanced coded vector

  D (**D**ual) indicates a complementary pair of coded vectors with a disparity of two

  F indicates a complementary pair of coded vectors with a disparity of **F**our

- The second capital letter indicates the disparity of the uncoded vector or the vertical ending coordinate in the left-side trellis of FIG.5 using the notation just below:

  U (**U**p, **U**ni) indicates a disparity of +1

  M (**M**inus) indicates a disparity of –1

C (**C**ube) indicates a disparity of + 3

T (**T**hree) indicates a disparity of –3

V (Roman numeral **V**) indicates a disparity of + 5

Q (**Q**uint) indicates a disparity of –5

H (**H**epta) indicates a disparity of +7

S (**S**even) indicates a disparity of –7

- A third capital letter, if present, indicates the value of the control input bit K.
- Up to three leading capital letters may be followed by one or more sets of a number paired with a lower case letter to indicate trellis nodes through which the members of the class must go, or not go if negated. The number marks the horizontal x coordinate and the lower case letter marks the vertical y coordinate. For an odd x-coordinate, the letter is the lower case version of the corresponding letter classification (HVCUMTQS) which is tied to the disparity of the uncoded vector as described above. For even x-coordinates, the lower case letter b (balanced) indicates a zero y-coordinate and for non-zero y-coordinates, the letter corresponds to the next letter classification closer to balance. Vectors going through negated nodes, e.g. 4t', must not be part of the specified class of vectors. This notation is illustrated in the left-side trellis of FIG. 5.
- The third and following capital letters other than K mark the uncoded bits, if any, which must be complemented to obtain the respective coded primary vector. The last coded bit *z* is appended with a default value of zero and complemented to a value of one, if indicated by a classification name ending in *Z*.

The left-side trellis of FIG. 5 below is used to define the vector classifications for the 7-bit source vectors. The trellis in the center shows the number of different vectors leading from the starting point to each node. The trellis on the right side covers the 8 coded bits.

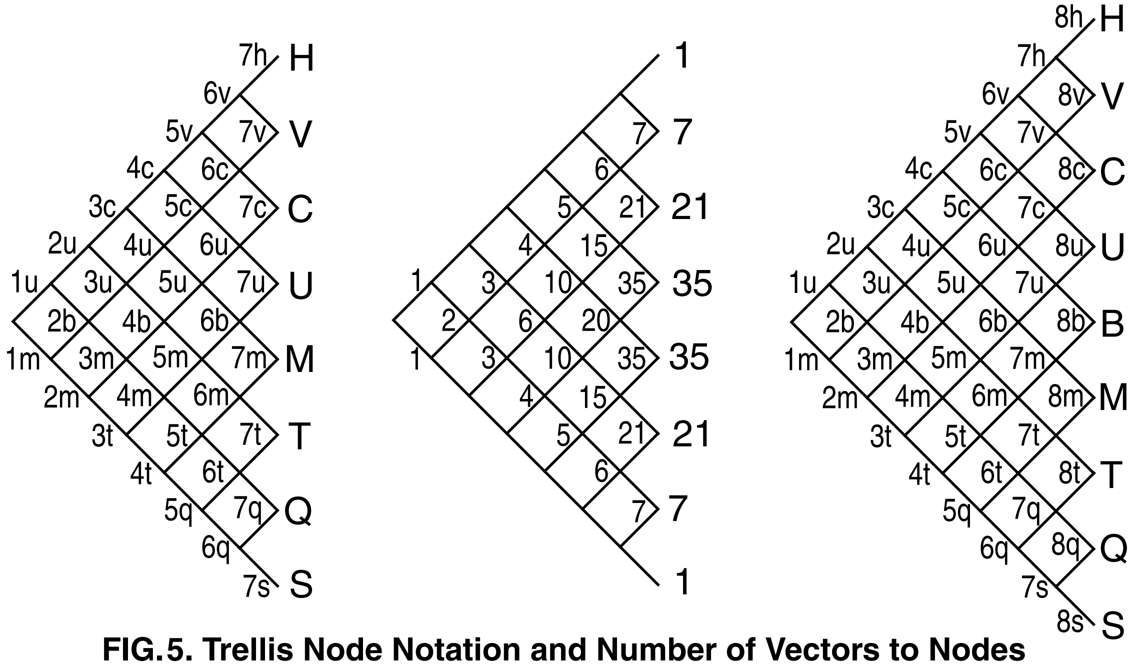


**FIG. 5. Trellis Node Notation and Number of Vectors to Nodes**

### 3) 7B8B Coding Table Construction

Table 1 represents a specific coding assignment between uncoded and coded vectors in the 7B8B domain. In the column ‘Bit Encoding Class’, K’ within parentheses for the vectors D7, D23, D39, and D71 means that the K-bit value need not be considered for bit encoding since the encoded Dx vector and the primary KxP vector are identical; the K-bit value for these vectors is only significant for the required entry disparity DR.

*a) 101 7B Primary Vectors congruent with the first 7 Bits of the coded 8B Vectors*

For 101 source vectors, represented by the trellis diagrams of FIG. 6, 7, 8, 9, 10, and 11, the first 7 bits of the primary encoded vectors are identical to the corresponding source vectors and the appended bit z assumes the default value (0).

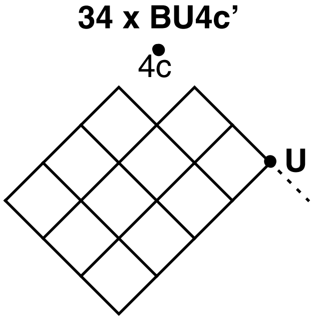


FIG. 6

The set of vectors BU4c’ of FIG. 8 uses up half the disparity independent balanced vectors of FIG. 1A.1.

*Enumeration of 34 Vectors BU4c’ of FIG. 6:*

D23 D27 D29 D30 D39 D43 D45
D46 D51 D53 D54 D57 D58 D60
D71 D75 D77 D78 D83 K85 D86
D89 D90 D92 D99 D101 D102 D105
D106 D108 D113 D114 D116 D120

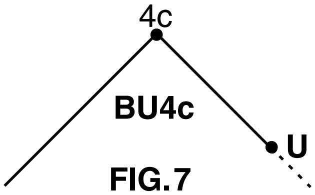


FIG. 7

The primary vector BU4c of FIG. 7 and FIG. 1A.2 is balanced and disparity dependent with a negative required entry disparity. It is assigned to the data vector D15.

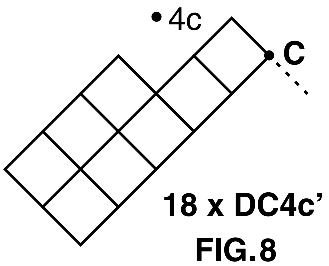


FIG. 8

The primary vector set DC4c’ of FIG. 8 has a disparity of +2 and uses 18 of the 48 vectors shown in FIG. 1B(L). The complementary alternate set is part of FIG. 1B(R).

*Enumeration of 18 Vectors DC4c’ of FIG. 8:*

D55 D59 D61 D62 D87 D91 D93
D94 D103 D107 D109 D110 D115 D117
D118 D121 D122 D124

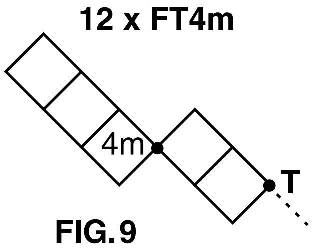


FIG. 9

The primary vector set FT4m of FIG. 9 has a disparity of −4 and matches all 12 vectors shown in FIG. 1C.1(R). The complementary alternate set is shown in FIG. 1C.1(L).

*Enumeration of 12 Vectors FT4m of FIG. 9:*

D17 D18 D20 D24 D33 D34 D36
D40 D65 D66 D68 D72

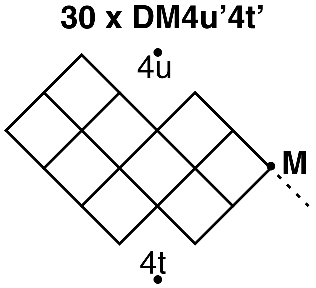


FIG. 10

The set of 30 primary vectors DM4u'4t' of FIG. 10 has a disparity −2 and uses the remaining 30 of the 48 vectors shown in FIG. 1B(R). The complementary alternate set is part of FIG. 1B(L).

*Enumeration of 30 Vectors DM4u'4t' of FIG. 10:*

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | D19 | D21 | D22 | D25 | D26 | D28 | D35 |
| | | D37 | D38 | D41 | D42 | D44 | D49 | D50 |
| | | D52 | D56 | D67 | D69 | D70 | D73 | D74 |
| D76 | D81 | D82 | D84 | D88 | D97 | D98 | D100 | D104 |

The 6 primary vectors shown in FIG. 11 are the vectors also illustrated in true and complement form in FIG. 1C.2. The vector C126 in dash-dot lines on the right side is used to generate a comma character for concatenated 7B8B sequences and for 17B20B code.

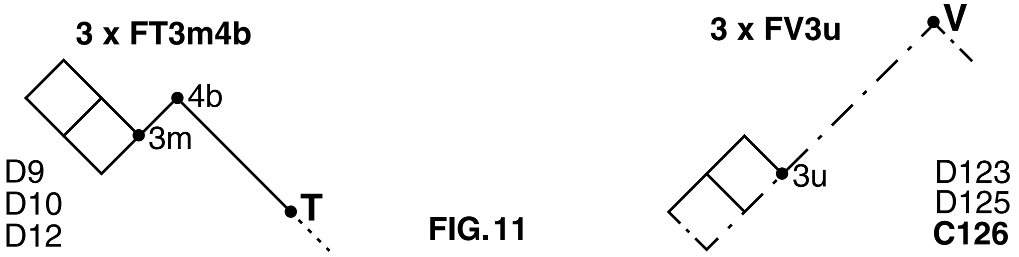


FIG. 11

*b) 34 Primary Vectors with modified Source Bits for Encoding*

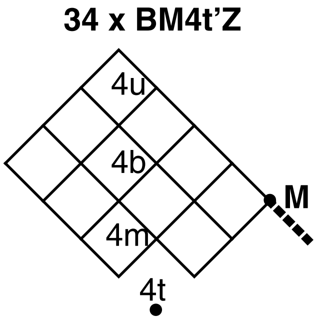


FIG. 12

All the encoded vectors with individual bit changes belong to the set of balanced disparity independent vectors BM4t'Z of FIG. 12 and are identified in Table 2 below. The expression BM4t' refers to the leading 7 encoded bits only. This set of vectors uses up the remaining half of FIG. 1A.1. For this subset of disparity independent vectors, one or more bits STUVWXYZ of the augmented source vector has to be complemented to fit the respective coded vector.

The 34 encoded vectors together with their assigned uncoded vectors are listed in Table 2. The bit in the S column of Table 2 is one if there is some symmetry in the bit patterns between the left and right side. The encoded bits which are obtained by complementation of the respective uncoded bit are shown in bold type.

## 7B8B Encoding

| Name | STUVWXY K | Bit Encoding Class | Primary stuvwxyz | Alternate stuvwxyz | DR | DB |
|---|---|---|---|---|---|---|
| D0 | 0000000 x | BSTVWZ | 0**1**0**11**00**1** | | ± | 0 |
| D1 | 1000000 x | BQVWZ | 100**11**00**1** | | ± | 0 |
| D2 | 0100000 x | BQWYZ | 0100**1**0**11** | | ± | 0 |
| D3 | 1100000 x | BT3uYZ | 110000**11** | | ± | 0 |
| D4 | 0010000 x | BQXYZ | 00100**111** | | ± | 0 |
| D5 | 1010000 x | BT3uYZ | 101000**11** | | ± | 0 |
| D6 | 0110000 x | BT3uYZ | 011000**11** | | ± | 0 |
| D7 | 1110000 0 | BM(K')4uZ | 1110000**1** | | ± | 0 |
| D8 | 0001000 x | BQSXZ | **1**0010**1**0**1** | | ± | 0 |
| D9 | 1001000 x | FT3m4b | 10010000 | 01101111 | + | −4 |
| D10 | 0101000 x | FT3m4b | 01010000 | 10101111 | + | −4 |
| D11 | 1101000 x | BM4uZ | 1101000**1** | | ± | 0 |
| D12 | 0011000 x | FT3m4b | 00110000 | 11001111 | + | −4 |
| D13 | 1011000 x | BM4uZ | 1011000**1** | | ± | 0 |
| D14 | 0111000 x | BM4uZ | 0111000**1** | | ± | 0 |
| D15 | 1111000 x | BU4c | 11110000 | 00001111 | − | 0 |
| D16 | 0000100 x | BQSUZ | **1**0**1**0100**1** | | ± | 0 |
| D17 | 1000100 x | FT4m | 10001000 | 01110111 | + | −4 |
| D18 | 0100100 x | FT4m | 01001000 | 10110111 | + | −4 |
| D19 | 1100100 0 | DMK'4u'4t' | 11001000 | 00110111 | + | −2 |
| D20 | 0010100 x | FT4m | 00101000 | 11010111 | + | −4 |
| D21 | 1010100 x | DM4u'4t' | 10101000 | 01010111 | + | −2 |
| D22 | 0110100 0 | DMK'4u'4t' | 01101000 | 10010111 | + | −2 |
| D23 | 1110100 0 | BU(K')4c' | 11101000 | | ± | 0 |
| D24 | 0001100 x | FT4m | 00011000 | 11100111 | + | −4 |
| D25 | 1001100 x | DM4u'4t' | 10011000 | 01100111 | + | −2 |
| D26 | 0101100 x | DM4u'4t' | 01011000 | 10100111 | + | −2 |
| D27 | 1101100 x | BU4c' | 11011000 | | ± | 0 |
| D28 | 0011100 x | DM4u'4t' | 00111000 | 11000111 | + | −2 |
| D29 | 1011100 x | BU4c' | 10111000 | | ± | 0 |
| D30 | 0111100 x | BU4c' | 01111000 | | ± | 0 |
| D31 | 1111100 x | BC4cSTZ | **00**11100**1** | | ± | 0 |
| D32 | 0000010 x | BQSTZ | **11**00010**1** | | ± | 0 |
| D33 | 1000010 x | FT4m | 10000100 | 01111011 | + | −4 |
| D34 | 0100010 x | FT4m | 01000100 | 10111011 | + | −4 |

**Table 1A**

## 7B8B Encoding

| Name | STUVWXY K | Bit Encoding Class | Primary stuvwxyz | Alternate stuvwxyz | DR | DB |
|---|---|---|---|---|---|---|
| D35 | 1100010 x | DM4u’4t’ | 11000100 | 00111011 | + | −2 |
| D36 | 0010010 x | FT4m | 00100100 | 11011011 | + | −4 |
| D37 | 1010010 x | DM4u’4t’ | 10100100 | 01011011 | + | −2 |
| D38 | 0110010 x | DM4u’4t’ | 01100100 | 10011011 | + | −2 |
| D39 | 1110010 0 | BU(K’)4c’ | 11100100 |  | ± | 0 |
| D40 | 0001010 x | FT4m | 00010100 | 11101011 | + | −4 |
| D41 | 1001010 x | DM4u’4t’ | 10010100 | 01101011 | + | −2 |
| D42 | 0101010 0 | DMK’4u’4t’ | 01010100 | 10101011 | + | −2 |
| D43 | 1101010 x | BU4c’ | 11010100 |  | ± | 0 |
| D44 | 0011010 x | DM4u’4t’ | 00110100 | 11001011 | + | −2 |
| D45 | 1011010 x | BU4c’ | 10110100 |  | ± | 0 |
| D46 | 0111010 x | BU4c’ | 01110100 |  | ± | 0 |
| D47 | 1111010 x | BC4cSTZ | **00**11010**1** |  | ± | 0 |
| D48 | 0000110 x | BT4tSZ | **1**000110**1** |  | ± | 0 |
| D49 | 1000110 x | DM4u’4t’ | 10001100 | 01110011 | + | −2 |
| D50 | 0100110 0 | DMK’4u’4t’ | 01001100 | 10110011 | + | −2 |
| D51 | 1100110 x | BU4c’ | 11001100 |  | ± | 0 |
| D52 | 0010110 x | DM4u’4t’ | 00101100 | 11010011 | + | −2 |
| D53 | 1010110 x | BU4c’ | 10101100 |  | ± | 0 |
| D54 | 0110110 x | BU4c’ | 01101100 |  | ± | 0 |
| D55 | 1110110 x | DC4c’ | 11101100 | 00010011 | − | +2 |
| D56 | 0001110 x | DM4u’4t’ | 00011100 | 11100011 | + | −2 |
| D57 | 1001110 x | BU4c’ | 10011100 |  | ± | 0 |
| D58 | 0101110 x | BU4c’ | 01011100 |  | ± | 0 |
| D59 | 1101110 x | DC4c’ | 11011100 | 00100011 | − | +2 |
| D60 | 0011110 x | BU4c’ | 00111100 |  | ± | 0 |
| D61 | 1011110 x | DC4c’ | 10111100 | 01000011 | − | +2 |
| D62 | 0111110 x | DC4c’ | 01111100 | 10000011 | − | +2 |
| D63 | 1111110 x | BV3cSTUZ | **000**1110**1** |  | ± | 0 |
| D64 | 0000001 x | BQSVZ | **1**00**1**001**1** |  | ± | 0 |
| D65 | 1000001 x | FT4m | 10000010 | 01111101 | + | −4 |
| D66 | 0100001 x | FT4m | 01000010 | 10111101 | + | −4 |
| D67 | 1100001 x | DM4u’4t’ | 11000010 | 00111101 | + | −2 |
| D68 | 0010001 x | FT4m | 00100010 | 11011101 | + | −4 |
| D69 | 1010001 x | DM4u’4t’ | 10100010 | 01011101 | + | −2 |

**Table 1B**

**7B8B Encoding**

| Name | STUVWXY K | Bit Encoding Class | Primary stuvwxyz | Alternate stuvwxyz | DR | DB |
|---|---|---|---|---|---|---|
| D70 | 0110001 x | DM4u’4t’ | 01100010 | 10011101 | + | −2 |
| D71 | 1110001 0 | BU(K’)4c’ | 11100010 |  | ± | 0 |
| D72 | 0001001 x | FT4m | 00010010 | 11101101 | + | −4 |
| D73 | 1001001 x | DM4u’4t’ | 10010010 | 01101101 | + | −2 |
| D74 | 0101001 0 | DMK’4u’4t’ | 01010010 | 10101101 | + | −2 |
| D75 | 1101001 x | BU4c’ | 11010010 |  | ± | 0 |
| D76 | 0011001 x | DM4u’4t’ | 00110010 | 11001101 | + | −2 |
| D77 | 1011001 x | BU4c’ | 10110010 |  | ± | 0 |
| D78 | 0111001 x | BU4c’ | 01110010 |  | ± | 0 |
| D79 | 1111001 x | BC4cSTZ | **00**11001**1** |  | ± | 0 |
| D80 | 0000101 x | BT4tSZ | **1**000101**1** |  | ± | 0 |
| D81 | 1000101 x | DM4u’4t’ | 10001010 | 01110101 | + | −2 |
| D82 | 0100101 x | DM4u’4t’ | 01001010 | 10110101 | + | −2 |
| D83 | 1100101 x | BU4c’ | 11001010 |  | ± | 0 |
| D84 | 0010101 x | DM4u’4t’ | 00101010 | 11010101 | + | −2 |
| D85 | 1010101 0 | BUK’SZ | **0**010101**1** |  | ± | 0 |
| D86 | 0110101 x | BU4c’ | 01101010 |  | ± | 0 |
| D87 | 1110101 x | DC4c’ | 11101010 | 00010101 | − | +2 |
| D88 | 0001101 x | DM4u’4t’ | 00011010 | 11100101 | + | −2 |
| D89 | 1001101 x | BU4c’ | 10011010 |  | ± | 0 |
| D90 | 0101101 x | BU4c’ | 01011010 |  | ± | 0 |
| D91 | 1101101 x | DC4c’ | 11011010 | 00100101 | − | +2 |
| D92 | 0011101 x | BU4c’ | 00111010 |  | ± | 0 |
| D93 | 1011101 x | DC4c’ | 10111010 | 01000101 | − | +2 |
| D94 | 0111101 x | DC4c’ | 01111010 | 10000101 | − | +2 |
| D95 | 1111101 x | BV3cSTUZ | **000**1101**1** |  | ± | 0 |
| D96 | 0000011 x | BT4tSZ | **1**000011**1** |  | ± | 0 |
| D97 | 1000011 x | DM4u’4t’ | 10000110 | 01111001 | + | −2 |
| D98 | 0100011 x | DM4u’4t’ | 01000110 | 10111001 | + | −2 |
| D99 | 1100011 x | BU4c’ | 11000110 |  | ± | 0 |
| D100 | 0010011 x | DM4u’4t’ | 00100110 | 11011001 | + | −2 |
| D101 | 1010011 x | BU4c’ | 10100110 |  | ± | 0 |
| D102 | 0110011 x | BU4c’ | 01100110 |  | ± | 0 |
| D103 | 1110011 x | DC4c’ | 11100110 | 00011001 | − | +2 |
| D104 | 0001011 x | DM4u’4t’ | 00010110 | 11101001 | + | −2 |

**Table 1C**

## 7B8B Encoding

| Name | STUVWXY K | Bit Encoding Class | Primary stuvwxyz | Alternate stuvwxyz | DR | DB |
|---|---|---|---|---|---|---|
| D105 | 1001011 x | BU4c’ | 10010110 |  | ± | 0 |
| D106 | 0101011 x | BU4c’ | 01010110 |  | ± | 0 |
| D107 | 1101011 x | DC4c’ | 11010110 | 00101001 | − | +2 |
| D108 | 0011011 x | BU4c’ | 00110110 |  | ± | 0 |
| D109 | 1011011 x | DC4c’ | 10110110 | 01001001 | − | +2 |
| D110 | 0111011 x | DC4c’ | 01110110 | 10001001 | − | +2 |
| D111 | 1111011 x | BV3cSUVZ | **0**1**00**011**1** |  | ± | 0 |
| D112 | 0000111 x | BM4tVWZ | 000**10**11**1** |  | ± | 0 |
| D113 | 1000111 x | BU4c’ | 10001110 |  | ± | 0 |
| D114 | 0100111 x | BU4c’ | 01001110 |  | ± | 0 |
| D115 | 1100111 x | DC4c’ | 11001110 | 00110001 | − | +2 |
| D116 | 0010111 x | BU4c’ | 00101110 |  | ± | 0 |
| D117 | 1010111 x | DC4c’ | 10101110 | 01010001 | − | +2 |
| D118 | 0110111 x | DC4c’ | 01101110 | 10010001 | − | +2 |
| D119 | 1110111 x | BV3cSTYZ | **00**1011**01** |  | ± | 0 |
| D120 | 0001111 x | BU4c’ | 00011110 |  | ± | 0 |
| D121 | 1001111 x | DC4c’ | 10011110 | 01100001 | − | +2 |
| D122 | 0101111 x | DC4c’ | 01011110 | 10100001 | − | +2 |
| D123 | 1101111 x | FV3u | 11011110 | 00100001 | − | +4 |
| D124 | 0011111 x | DC4c’ | 00111110 | 11000001 | − | +2 |
| D125 | 1011111 x | FV3u | 10111110 | 01000001 | − | +4 |
| D126 | 0111111 0 | BVK’1mVWYZ | 011**00**1**01** |  | ± | 0 |
| D127 | 1111111 x | BHTVWYZ | 1**0**1**00**1**01** |  | ± | 0 |
| K7[1] | 1110000 1 | BMK4u | 1110000**1** | 00011110 | + | 0 |
| K23[1] | 1110100 1 | BUK4c’ | 11101000 | 00010111 | + | 0 |
| K39[1] | 1110010 1 | BUK4c’ | 11100100 | 00011011 | + | 0 |
| K71[1] | 1110001 1 | BUK4c’ | 11100010 | 00011101 | + | 0 |
| K19 | 1100100 1 | BMKZ | 1100100**1** |  | ± | 0 |
| K22 | 0110100 1 | BMKZ | 0110100**1** |  | ± | 0 |
| K42 | 0101010 1 | BMKZ | 0101010**1** |  | ± | 0 |
| K50 | 0100110 1 | BMKZ | 0100110**1** |  | ± | 0 |
| K74 | 0101001 1 | BMKZ | 0101001**1** |  | ± | 0 |
| K85 | 1010101 1 | BUK | 10101010 |  | ± | 0 |
| C126 | 0111111 1 | FVK3u | 01111110 | 10000001 | − | +4 |

**Table 1D**

[1] Use restricted to comma trailer

**34 Disparity Independent Vectors BM4t'Z from FIG14**

| NAME | STUVWXY K | stuvwxyz | S | NAME | STUVWXY K | stuvwxyz |
|---|---|---|---|---|---|---|
| D3 | 1100000 x | 11000011 | 0 | D7 | 1110000 x | 11100001 |
| D5 | 1010000 x | 10100011 | 0 | D11 | 1101000 x | 11010001 |
| D6 | 0110000 x | 01100011 | 0 | D13 | 1011000 x | 10110001 |
| D2 | 0100000 x | 01001011 | 0 | D14 | 0111000 x | 01110001 |
| D4 | 0010000 x | 00100111 | | | | |
| D0 | 0000000 x | 01011001 | 1 | D127 | 1111111 x | 10100101 |
| D1 | 1000000 x | 10011001 | 1 | D126 | 0111111 0 | 01100101 |
| D8 | 0001000 x | 10010101 | 1 | D119 | 1110111 x | 00101101 |
| D16 | 0000100 x | 10101001 | 1 | D111 | 1111011 x | 01000111 |
| D32 | 0000010 x | 11000101 | 1 | D95 | 1111101 x | 00011011 |
| D64 | 0000001 x | 10010011 | 1 | D63 | 1111110 x | 00011101 |
| D48 | 0000110 x | 10001101 | 1 | D79 | 1111001 x | 00110011 |
| D80 | 0000101 x | 10001011 | 1 | D47 | 1111010 x | 00110101 |
| D96 | 0000011 x | 10000111 | 1 | D31 | 1111100 x | 00111001 |
| D112 | 0000111 x | 00010111 | 0 | D85 | 1010101 0 | 00101011 |
| K19 | 1100100 1 | 11001001 | 0 | K50 | 0100110 1 | 01001101 |
| K22 | 0110100 1 | 01101001 | 0 | K74 | 0101001 1 | 01010011 |
| K42 | 0101010 1 | 01010101 | | | | |

**Table 2**

*c) Value of Control Bit K*

The vectors K7, K23, K39, and K71 of Table 1 are not true control characters because by themselves, they can not be distinguished from data. They have control functions only in combination with the preceding control character C126, or C508 in the context of the 16B18B code. The K-bit value for these pseudo control characters can be either supplied externally or be supplanted by the leading comma part which is the preferred implementation because it simplifies the recovery of the K-bit for the true control characters at the cost of limiting the use of C126 and C508 to comma sequences and no other independent stand alone control functions. For the preferred implementation, the K-bit value of these 4 vectors can assume a zero value and the respective entries in the K-column of Table 1 could be changed to x. A value of 1 is shown as a reminder that these vectors are disparity dependent when following C126 or C508.

For a majority of data vectors, the value of the K-bit can be ignored. It must be considered for all true control characters and for all data classes for which the bit encoding for some source vectors is different for data and control. For the class of DM4u'4t' of FIG. 10, the vectors D19, D22, D42, D50, and D74 are encoded differently from K19, K22, K42, K50, and K74, respectively. The same is true for the vector K85 of FIG. 6 and D85 of Table 2 and the vector D126 of Table 2 and C126 of FIG. 11(R). In contrast, the coded primary vectors and the disparity DB for D/K7, D/K23, D/K39, and D/K71 are identical; the only difference is the required entry disparity DR.

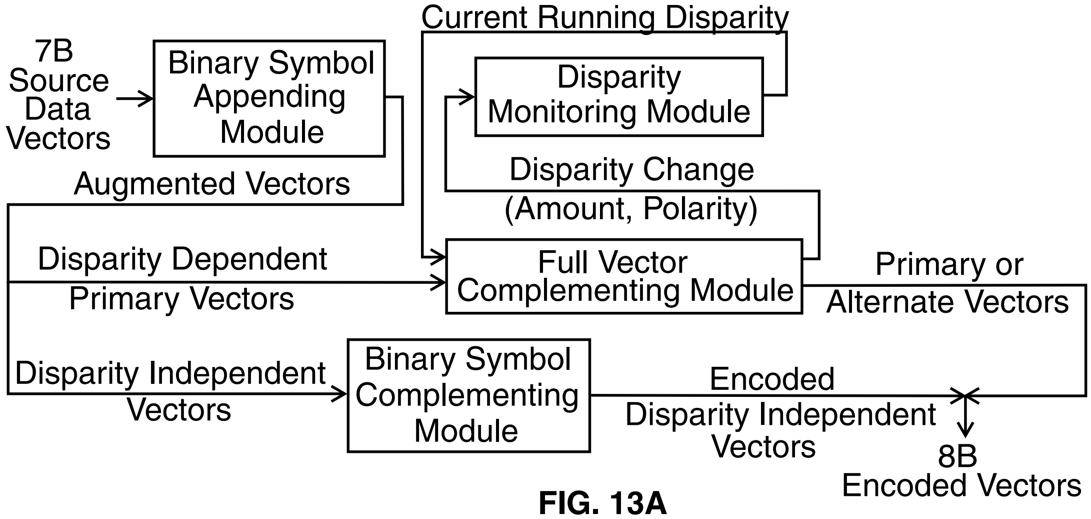
## III. LOGIC EQUATIONS FOR IMPLEMENTATION

FIGS.13A and 14A show a conceptual view of encoding and decoding, respectively. They illustrate the parallelism in the processing of various vector classes which is the key to the relatively simple implementation with low latency. Note that full vector complementation and changes in individual bits are completely separate and independent of each other.

FIGS. 13B and 14B present another view of encoding and decoding which is more circuit oriented. The decoder circuit includes code violation and disparity error checks. In the presence of errors, the received blocks may have a disparity of ±6 or ±8, which are outside the normal range but are assigned a disparity value of ±4 for purposes of the running disparity.

The implementation problems to be solved for the blocks labelled ‘Encoder’ and ‘Decoder’ are circuit area and delay reduction.

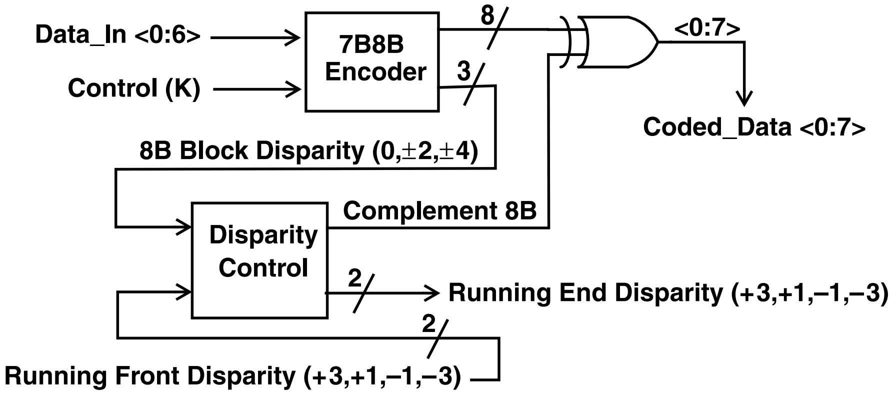
**Conceptual View of 7B8B Encoding**



Current Running Disparity
7B Source Data Vectors
Binary Symbol Appending Module
Disparity Monitoring Module
Augmented Vectors
Disparity Change (Amount, Polarity)
Disparity Dependent Primary Vectors
Full Vector Complementing Module
Primary or Alternate Vectors
Disparity Independent Vectors
Binary Symbol Complementing Module
Encoded Disparity Independent Vectors
8B Encoded Vectors

**FIG. 13A**

**Circuit View of 7B8B Encoder**



Data_In <0:6>
Control (K)
7B8B Encoder
8
3
<0:7>
Coded_Data <0:7>
8B Block Disparity (0,±2,±4)
Complement 8B
Disparity Control
2
Running End Disparity (+3,+1,–1,–3)
2
Running Front Disparity (+3,+1,–1,–3)

**FIG. 13B**

**Conceptual View of 8B7B Decoding**

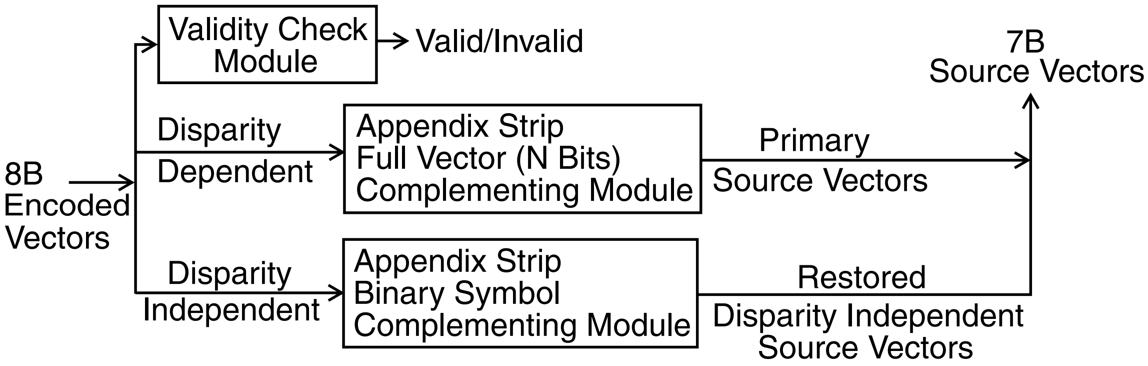

Validity Check Module
Valid/Invalid
7B Source Vectors
Disparity Dependent
Appendix Strip Full Vector (N Bits) Complementing Module
Primary Source Vectors
8B Encoded Vectors
Disparity Independent
Appendix Strip Binary Symbol Complementing Module
Restored Disparity Independent Source Vectors

FIG. 14A

**Circuit View of 8B7B Decoder**

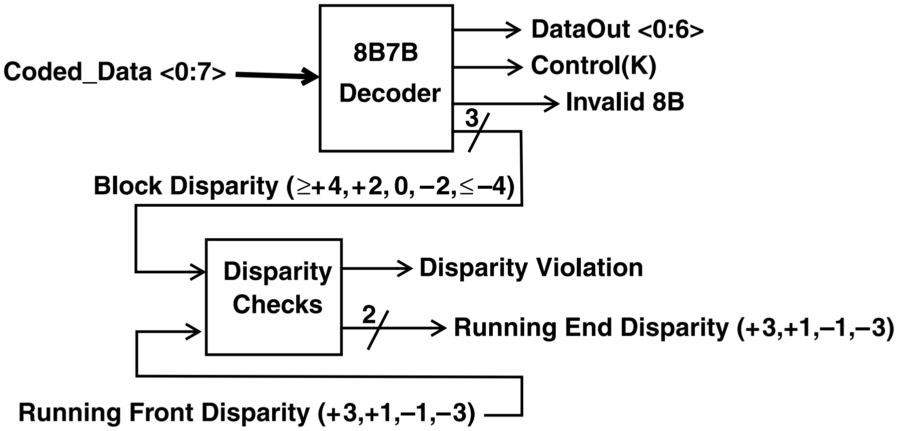

8B7B Decoder
Coded_Data <0:7>
DataOut <0:6>
Control(K)
Invalid 8B
3
Block Disparity (≥+4,+2,0,−2,≤−4)
Disparity Checks
Disparity Violation
2
Running End Disparity (+3,+1,–1,–3)
Running Front Disparity (+3,+1,–1,–3)

FIG. 14B

*Notation:* In the following equations, some capital letters have overlapping use for group classifications and for the designation of a specific uncoded input bit. If the dual use can be ambiguous such as for a single letter designating a classification, the classification is referred to with bold, underlined type. The bit designations are always referred to with plain type. As an example, the bold letter **S** refers to all input patterns which lead to the node 7s in the trellis of FIG. 7, and the plain letter S refers to the uncoded bit S of the 7B8B code. In some tables, a free standing letter **S** in the column header alerts for some symmetry between the left and right side of the table if there is a 1 in a specific row. The symmetric relationship might be complementary values for the bit positions marked by bold type or equal values marked by italic type. In the equations below, the Boolean function ⊕ is executed before the AND or OR functions. This function is defined here with a single parameter on each side, i.e. x⊕y is equivalent to (x⊕y) to allow the elimination of one level of parentheses.

# A. Logic Equations for 7B8B Encoder

## 1) Equations for Individual Bit Encoding

Generally, the encoded bits retain the value of the unencoded bit (s=S, t=T, etc), but the source bit is complemented (s=S’, t=T’, etc) if the respective equation below is true.

*Encoded Bit s*

The ‘s’ column has bold entries in the Tables 1 and 2 for the 15 vectors listed in Table 3s. The s-bit encoding equation is derived from the coding labels of Table 3s.

**s-bit Encoding**

| Name | STUVWXY K | a | S | Name | STUVWXY K | a | Coding Label |
|---|---|---|---|---|---|---|---|
| D8 | **000**1**000** x | 1 | 1 | D119 | **111**0**111** x | 0 | (U⊕X’•X⊕Y’•V⊕W + U⊕V’•V⊕W’•X⊕Y) • S⊕T’•T⊕U’ |
| D16 | **000**01**00** x | 1 | 1 | D111 | **111**10**11** x | 0 | |
| D32 | **00000**10 x | 1 | 1 | D95 | **11111**01 x | 0 | |
| D64 | **00000**01 x | 1 | 1 | D63 | **11111**10 x | 0 | |
| D48 | **00001**10 x | 1 | 1 | D79 | **11110**01 x | 0 | (V⊕W’•W⊕X•X⊕Y’ + V⊕W•X⊕Y) • S⊕T’•T⊕U’•U⊕V’ |
| D80 | **00001**01 x | 1 | 1 | D47 | **11110**10 x | 0 | |
| D96 | **0000011** x | 1 | 1 | D31 | **1111100** x | 0 | |
| | | | | D85 | 1010101 0 | 0 | S•T’•U•V’•W•X’•Y•K’ |

**Table 3s**

$$s = S\oplus\{(U\oplus X'\bullet X\oplus Y'\bullet V\oplus W + U\oplus V'\bullet V\oplus W'\bullet X\oplus Y) \bullet S\oplus T'\bullet T\oplus U' + \qquad \text{(Pn1)}$$
$$(V\oplus W'\bullet W\oplus X\bullet X\oplus Y' + V\oplus W\bullet X\oplus Y) \bullet S\oplus T'\bullet T\oplus U'\bullet U\oplus V' + \qquad \text{(Pn2*)}$$
$$S\bullet T'\bullet U\bullet V' \bullet W\bullet X'\bullet Y\bullet K'\} \qquad \text{(Pn2*)}$$

*Encoded Bit t*

The ‘t’ column has bold entries in the Tables 1 and 2 for the 9 vectors listed in Table 3t.

**t-bit Encoding**

| Name | STUVWXY K | t | S | Name | STUVWXY K | t | Coding Label |
|---|---|---|---|---|---|---|---|
| D0 | **00000**00 x | 1 | 1 | D127 | **11111**11 x | 0 | S⊕T’•T⊕U’•U⊕V’•V⊕W’•W⊕Y’ |
| D32 | **00000**1**0** x | 1 | 1 | D95 | **11111**0**1** x | 0 | |
| D63 | *111***1***11***0** x | 0 | 1 | D119 | *111***0***11***1** x | 0 | V⊕Y • S•T•U•W•X |
| | | | | D31 | **1111**100 x | 0 | (W•X’•Y’ + W’•X•Y’ + W’•X’•Y) • S•T•U•V |
| | | | | D47 | **1111**010 x | 0 | |
| | | | | D79 | **1111**001 x | 0 | |

**Table 3t**

$$t = T\oplus\{(W\bullet X'\bullet Y' + W'\bullet X\bullet Y' + W'\bullet X'\bullet Y) \bullet S\bullet T\bullet U\bullet V + \qquad \text{(Pn3)}$$
$$S\oplus T'\bullet T\oplus U'\bullet U\oplus V'\bullet V\oplus W'\bullet W\oplus Y' + V\oplus Y \bullet S\bullet T\bullet U\bullet W\bullet X\} \qquad \text{(NCt1*)}$$

*Encoded Bit u*

The ‘u’ column has bold entries in the Tables 1 and 2 for the 4 vectors listed in Table 3u.

**u-bit Encoding**

| Name | STUVWXY K | u | S | Name | STUVWXY K | u | Coding Label |
|---|---|---|---|---|---|---|---|
| D16 | **0000100** x | 1 | 1 | D111 | **1111011** x | 0 | S⊕T’•T⊕U’•U⊕V’•V⊕W•W⊕X•X⊕Y’ |
| D63 | *11111***10** x | 0 | 1 | D95 | *11111***01** x | 0 | X⊕Y • S•T•U•V•W |

**Table 3u**

$$u = U\oplus (S\oplus T'\bullet T\oplus U'\bullet U\oplus V' \bullet V\oplus W\bullet W\oplus X\bullet X\oplus Y' + X\oplus Y\bullet S\bullet T\bullet U\bullet V\bullet W) \qquad (NCu1)$$

*Encoded Bit v*

The ‘v’ column has bold entries in the Tables 1 and 2 for the 7 vectors listed in Table 3v.

**v-bit Encoding**

| Name | STUVWXY K | v | S | Name | STUVWXY K | v | Coding Label |
|---|---|---|---|---|---|---|---|
| D0 | 0**000000** x | 1 | 1 | D127 | 1**111111** x | 0 | T⊕U’•U⊕V’•V⊕W’•W⊕X’•X⊕Y’•K’ |
| D1 | 1**000000** x | 1 | 1 | D126 | 0**111111** 0 | 0 | |
| D112 | **00001***11* x | 1 | 1 | D111 | **11110***11* x | 0 | S⊕T’•T⊕U’•U⊕V’•V⊕W • X•Y |
| D0* | **000000**0 x | 1 | | | | | S’•T’•U’•V’•W’•X’ |
| D64 | **0000001** x | 1 | | | | | |

**Table 3v**

$$\begin{aligned} v = V\oplus(&T\oplus U'\bullet U\oplus V'\bullet V\oplus W'\bullet W\oplus X'\bullet X\oplus Y' \bullet K' + && (Pn5^*) \\ &S\oplus T'\bullet T\oplus U'\bullet U\oplus V' \bullet V\oplus W \bullet X\bullet Y + S'\bullet T'\bullet U'\bullet V'\bullet W'\bullet X') && (Pn5^*;\ NCv1^*) \end{aligned}$$

*Encoded Bit w*

The ‘w’ column has bold entries in the Tables 1 and 2 for the 6 vectors listed in Table 3w.

**w-bit Encoding**

| Name | STUVWXY K | w | Name | STUVWXY K | w | Coding Label |
|---|---|---|---|---|---|---|
| D0 | 00**00000** x | 1 | | | | (S’+T’) • U’•V’•W’•X’•Y’ |
| D1 | 10**00000** x | 1 | | | | |
| D2 | 01**00000** x | 1 | | | | |
| | | | D112 | 0000111 x | 0 | S’•T’•U’•V’ • W•X•Y |
| | | | D126 | 0**111111** 0 | 0 | T•U•V•W•X•Y•K’ |
| | | | D127 | **1111111** x | 0 | |

**Table 3w**

$$w = W\oplus\{(S'+T')\bullet U'\bullet V'\bullet W'\bullet X'\bullet Y'+ S'\bullet T'\bullet U'\bullet V'\bullet W\bullet X\bullet Y + T\bullet U\bullet V\bullet W\bullet X\bullet Y\bullet K'\} \qquad (NCw1)$$

*Encoded Bit x*

The ‘x’ column has bold entries in the Tables 1 and 2 for the 2 vectors listed in Table 3x.

**x-bit Encoding**

| Name | STUVWXY K | x | S | Name | STUVWXY K | x | Coding Label |
|---|---|---|---|---|---|---|---|
| D4 | 0010000 x | 1 | 1 | D8 | 0001000 x | 1 | U⊕V • S’•T’•W’•X’•Y’ |

**Table 3x**

$$x = X \oplus (U \oplus V \bullet S' \bullet T' \bullet W' \bullet X' \bullet Y') \qquad (NCx1)$$

*Encoded Bit y*

The ‘y’ column has bold entries in the Tables 1 and 2 for the 8 vectors listed in Table 3y.

**y-bit Encoding**

| Name | STUVWXY K | y | Name | STUVWXY K | y | Coding Label |
|---|---|---|---|---|---|---|
| D2 | 0100000 x | 1 | | | | (T⊕U + S’•T•U) • V’•W’•X’•Y’ |
| D4 | 0010000 x | 1 | | | | |
| D3 | 1100000 x | 1 | | | | |
| D5 | 1010000 x | 1 | | | | |
| D6 | 0110000 x | 1 | | | | |
| | | | D119 | 1110111 x | 0 | (S+V) • T•U•W•X•Y•K’ |
| | | | D126 | 0111111 0 | 0 | |
| | | | D127 | 1111111 x | 0 | |

**Table 3y**

$$y = Y \oplus \{(T \oplus U + S' \bullet T \bullet U) \bullet V' \bullet W' \bullet X' \bullet Y' + (S+V) \bullet T \bullet U \bullet W \bullet X \bullet Y \bullet K'\} \qquad (NCy1)$$

*Encoded Bit z*

The default value for the z-bit is zero. The z-bit is changed to one for the vectors with bold entries in the ‘z’ column of Tables 1 and 2. The respective 34 vectors are listed in Table 3z.

**z-bit Encoding**

| Name | STUVWXY K | z | S | Name | STUVWXY K | z | Coding Label |
|---|---|---|---|---|---|---|---|
| D2 | 010**0000** x | 1 | | | | | (T+U) • V’•W’•X’•Y’ |
| D3 | 110**0000** x | 1 | | | | | |
| D4 | 001**0000** x | 1 | | | | | |
| D5 | 101**0000** x | 1 | | | | | |
| D6 | 011**0000** x | 1 | | | | | |
| D7 | 111**0000** x | 1 | | | | | |
| D3* | *110*0*000* x | 1 | 1 | D11 | *110*1*000* x | 1 | (S•T•U’ + S•T’•U + S’•T•U) • W’•X’•Y’ |
| D5* | *101*0*000* x | 1 | 1 | D13 | *101*1*000* x | 1 | |
| D6* | *011*0*000* x | 1 | 1 | D14 | *011*1*000* x | 1 | |
| D0 | 0**000000** x | 1 | 1 | D127 | 1**111111** x | 1 | T⊕U’•U⊕V’•V⊕W’•W⊕X’•X⊕Y’•K’ |
| D1 | 1**000000** x | 1 | 1 | D126 | 0**111111** 0 | 1 | |
| D8 | **000**1000 x | 1 | 1 | D119 | **111**0111 x | 1 | (V⊕W•U⊕X’•X⊕Y’ + U⊕V’•V⊕W’•X⊕Y) • S⊕T’•T⊕U’ |
| D16 | **000**0100 x | 1 | 1 | D111 | **111**1011 x | 1 | |
| D32 | **000**0010 x | 1 | 1 | D95 | **111**1101 x | 1 | |
| D64 | **000**0001 x | 1 | 1 | D63 | **111**1110 x | 1 | |
| D48 | **0000**110 x | 1 | 1 | D79 | **1111**001 x | 1 | (V⊕W’•W⊕X•X⊕Y’+V⊕W•X⊕Y) • S⊕T’•T⊕U’•U⊕V’ |
| D80 | **0000**101 x | 1 | 1 | D47 | **1111**010 x | 1 | |
| D96 | **0000**011 x | 1 | 1 | D31 | **1111**100 x | 1 | |
| D112 | 0000111 x | 1 | | | | | S’•T’•U’•V’ • W•X•Y |
| K42 | **0101010 1** | 1 | 1 | D85 | **1010101 0** | 1 | S⊕T•T⊕U•U⊕V•V⊕W • W⊕X• X⊕Y•Y⊕K |
| | | | | | | | |
| K19 | **1***1***0***0100 1* | 1 | 1 | K22 | **0***11***0***100 1* | 1 | S⊕U • T•V’•W•X’•Y’•K |
| K50 | *010***0110** *1* | 1 | 1 | K74 | *010***1001** *1* | 1 | V⊕W•W⊕X’•X⊕Y • S’•U•V’•K |

**Table 3z**

$$z = Z\oplus\{(V\oplus W'\bullet W\oplus X\bullet X\oplus Y'+V\oplus W\bullet X\oplus Y) \bullet S\oplus T'\bullet T\oplus U'\bullet U\oplus V' + \qquad (Pn6)$$

$$(V\oplus W\bullet U\oplus X'\bullet X\oplus Y' + U\oplus V'\bullet V\oplus W'\bullet X\oplus Y) \bullet S\oplus T'\bullet T\oplus U' + \qquad (Pn7)$$

$$(S\bullet T\bullet U' + S\bullet T'\bullet U + S'\bullet T\bullet U) \bullet W'\bullet X'\bullet Y' + \qquad (n8)$$

$$S\oplus T\bullet T\oplus U\bullet U\oplus V\bullet V\oplus W \bullet W\oplus X\bullet X\oplus Y\bullet Y\oplus K + \qquad (Pn9^*)$$

$$T\oplus U'\bullet U\oplus V'\bullet V\oplus W'\bullet W\oplus X'\bullet X\oplus Y'\bullet K' + \qquad (Pn9^*)$$

$$V\oplus W\bullet W\oplus X'\bullet X\oplus Y \bullet S'\bullet U\bullet V'\bullet K + \qquad (n10^*)$$

$$S\oplus U \bullet T\bullet V'\bullet W\bullet X'\bullet Y'\bullet K + (T+U) \bullet V'\bullet W'\bullet X'\bullet Y' + S'\bullet T'\bullet U'\bullet V' \bullet W\bullet X\bullet Y\} \qquad (n10^*)$$

## 2) Equations for Required Disparity for Encoding DR

*a) Positive Required Disparity PDR, Table 4*

A total of 49 vectors require a positive entry disparity. 30 belong to the class DM4u'4t' of FIG. 10, 15 to the class FT4m of FIG. 9 and FT3m4b of FIG. 11(L). In addition, 4 primary pseudo-control vectors for the generation of commas require also a positive entry disparity (K7, K23, K39, K71). All 49 vectors are listed and sorted for easy implementation.

**Source Vectors with Positive Required Disparity DR=+**

| Name | DR Class | STUVWXY K | Coding Label |
|---|---|---|---|
| D9 | FT3m4b | 10010**00** x | {(V⊕W•X⊕Y•K' + (V+W)•X'•Y' + (X+Y)•V'•W'} • (S•T'•U' + S'•T•U' + S'•T'•U) |
| D10 | FT3m4b | 01010**00** x | |
| D12 | FT3m4b | 00110**00** x | |
| D17 | FT4m | 10001**00** x | |
| D18 | FT4m | 01001**00** x | |
| D20 | FT4m | 00101**00** x | |
| D25 | DM4u'4t' | 10011**00** x | |
| D26 | DM4u'4t' | 01011**00** x | |
| D28 | DM4u'4t' | 00111**00** x | |
| D41 | DM4u'4t' | 1001010 x | |
| D42 | DM4u'4t' | 0101010 0 | |
| D44 | DM4u'4t' | 0011010 x | |
| D49 | DM4u'4t' | 1000110 x | |
| D50 | DM4u'4t' | 0100110 0 | |
| D52 | DM4u'4t' | 0010110 x | |
| D73 | DM4u'4t' | 1001001 x | |
| D74 | DM4u'4t' | 0101001 0 | |
| D76 | DM4u'4t' | 0011001 x | |
| D81 | DM4u'4t' | 1000101 x | |
| D82 | DM4u'4t' | 0100101 x | |
| D84 | DM4u'4t' | 0010101 x | |
| D33 | FT4m | 100**00**10 x | |
| D34 | FT4m | 010**00**10 x | |
| D36 | FT4m | 001**00**10 x | |
| D65 | FT4m | 100**00**01 x | |
| D66 | FT4m | 010**00**01 x | |
| D68 | FT4m | 001**00**01 x | |
| D97 | DM4u'4t' | 100**00**11 x | |
| D98 | DM4u'4t' | 010**00**11 x | |
| D100 | DM4u'4t' | 001**00**11 x | |

**Table 4A**

**Source Vectors with Positive Required Disparity DR=+**

| Name | DR Class | STUVWXY K | Coding Label |
|---|---|---|---|
| D19 | DM4u'4t' | 110**0**100 0 | (S•T•U' + S•T'•U + S'•T•U) • (W•X'•Y' + W'•X•Y' + W'•X'•Y) •V'•K' |
| D21 | DM4u'4t' | 101**0**100 x | |
| D22 | DM4u'4t' | 011**0**100 0 | |
| D35 | DM4u'4t' | 110**0**010 x | |
| D37 | DM4u'4t' | 101**0**010 x | |
| D38 | DM4u'4t' | 011**0**010 x | |
| D67 | DM4u'4t' | 110**0**001 x | |
| D69 | DM4u'4t' | 101**0**001 x | |
| D70 | DM4u'4t' | 011**0**001 x | |
| D24 | FT4m | **0001***100* x | (W'+X'+Y') • (W+X+Y) • S'•T'•U'•V |
| D40 | FT4m | **0001***010* x | |
| D72 | FT4m | **0001***001* x | |
| D56 | DM4u'4t' | **0001***110* x | |
| D88 | DM4u'4t' | **0001***101* x | |
| D104 | DM4u'4t' | **0001***011* x | |
| K7 | | **111000**0 **1** | (W⊕X•Y' + W'•X') • S•T•U•V'•K |
| K71 | | **111000**1 **1** | |
| K23 | | **1110**10**0 1** | |
| K39 | | **1110**01**0 1** | |

**Table 4B**

The equation for positive required disparity PDR can thus be written as follows:

$$
\begin{aligned}
&PDR = \{(V \oplus W \bullet X \oplus Y \bullet K' + (V+W) \bullet X' \bullet Y' + (X+Y) \bullet V' \bullet W'\} \bullet && (n20)\\
&\qquad (S \bullet T' \bullet U' + S' \bullet T \bullet U' + S' \bullet T' \bullet U) + && (Pn21^*)\\
&(S \bullet T \bullet U' + S \bullet T' \bullet U + S' \bullet T \bullet U) \bullet (W \bullet X' \bullet Y' + W' \bullet X \bullet Y' + W' \bullet X' \bullet Y) \bullet V' \bullet K' + && (n22;Pn23^*)\\
&(W \oplus X \bullet Y' + W' \bullet X') \bullet S \bullet T \bullet U \bullet V' \bullet K + && (Pn24)\\
&(W'+X'+Y') \bullet (W+X+Y) \bullet S' \bullet T' \bullet U' \bullet V && (Pn25)
\end{aligned}
$$

The 4 pseudo-control characters may be governed by the higher level protocol which may set the respective K-value to 1, or it may be governed by an encoding circuit which automatically sets the K-value to 1 for vectors which follow the leading part of a comma. In the second case, 'K' in the last coding label of Table 4B is replaced by 'C126', which assumes a value of one if it is preceded by the C126 vector for concatenated 8B vectors, or K is replaced by 'C508' which likewise assumes a value of one if preceded by the C508 vector of the 9B10B code in the 16B18B application.

*b) Negative Required Disparity for Encoding NDR, Table 5*

A total of 22 vectors require a negative entry disparity. 18 belong to the class DC4c’ of FIG. 8, 3 to the class FV3u of FIG. 11(R), and the vector BU4c is shown in FIG. 7. All 22 vectors are listed and sorted for easy implementation in Table 5.

**Negative Required Disparity DR = −**

| Name | DR Class | STUVWXY K | Coding Label |
|---|---|---|---|
| D55 | DC4c’ | 1110110 x | U⊕V•X⊕Y • S•T•W |
| D59 | DC4c’ | 1101110 x | |
| D87 | DC4c’ | 1110101 x | |
| D91 | DC4c’ | 1101101 x | |
| D61 | DC4c’ | 1011110 x | S⊕T•X⊕Y • U•V•W |
| D62 | DC4c’ | 0111110 x | |
| D93 | DC4c’ | 1011101 x | |
| D94 | DC4c’ | 0111101 x | |
| D109 | DC4c’ | 1011011 x | S⊕T•V⊕W • U•X•Y |
| D110 | DC4c’ | 0111011 x | |
| D117 | DC4c’ | 1010111 x | |
| D118 | DC4c’ | 0110111 x | |
| D121 | DC4c’ | 1001111 x | (V•W•X•Y) • (S+T+U) • (T’+U’) |
| D122 | DC4c’ | 0101111 x | |
| D123 | FV3u | 1101111 x | |
| D124 | DC4c’ | 0011111 x | |
| D125 | FV3u | 1011111 x | |
| D15 | BU4c | 1111000 x | V⊕X • X⊕Y’ • S•T•U•W’ |
| D103 | DC4c’ | 1110011 x | |
| D107 | DC4c’ | 1101011 x | V⊕W • S•T•U’•X•Y |
| D115 | DC4c’ | 1100111 x | |
| C126 | FV3uK | 0111111 1 | K•X•Y |

**Table 5**

The coding label for C126 can be verified by an examination of the bottom 11 rows of Table 1D. The equation for negative required disparity NDR can be written as follows:

$NDR = V\oplus X \bullet X\oplus Y' \bullet S\bullet T\bullet U\bullet W' + U\oplus V\bullet X\oplus Y \bullet S\bullet T\bullet W +$ (n26*)

$S\oplus T\bullet X\oplus Y \bullet U\bullet V\bullet W + S\oplus T\bullet V\oplus W \bullet U\bullet X\bullet Y +$ (n26*)

$V\oplus W \bullet S\bullet T\bullet U'\bullet X\bullet Y + (V\bullet W\bullet X\bullet Y) \bullet (S+T+U) \bullet (T'+U') + K\bullet X\bullet Y$ (n27)

### 3) Equation for Complementation of the Primary Vector (CMPLP8)

The running disparity at the vector boundaries is constrained to the four values plus or minus one or three. If the required entry disparity PDR or NDR does not match the polarity of running disparity RD, the alternate vector must be used. The alternate vector is generated by complementation of the primary vector. The positive or negative running disparity in front of a byte is referred to as PRDF or NPRDF=NRDF, respectively.

*CMPLP8 = PDR•NRDF + NDR•PRDF*

The signals PRDF and NRDF are applied preferably upstream to each logic path, instead of to the complete PDR and NDR functions, to eliminate one level of gating.

### 4) Equations for Running Disparity RD (FIG. 15)

FIG. 15 is a state transition diagram for the running disparity RD based an the block disparities DB of the encoded vectors. The vector complementation circuit ensures that the block polarities of vectors conform with the constraints of FIG15. The running disparity can be represented by two flip-flops which pass the value along from byte to byte. The trailing values become the front values of the next encoding cycle. The output of a first flip-flop FFP indicates a positive (PRDF) or negative (NRDF) polarity and the output of a second flip-flop FFA indicates an arithmetic value of one (RD1) or three (RD3).

The two flip-flops can assume arbitrary initial values and disparity violations may be generated initially. At least three unbalanced vectors must be transmitted before payload data transmission is allowed to start. Additional requirements may have to be met before the receiver disparity monitor is in the ready state.

The conditions for complementing these two flip-flop values can be derived from FIG15.

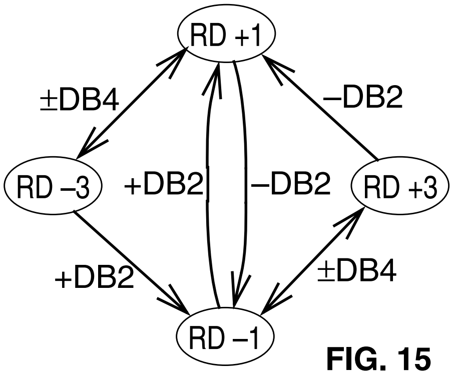


**FIG. 15**

*CMPLFFP = DB2•RD1 + DB4*
*CMPLFFA = DB2•RD3 + DB4*

The block disparity DB2 in the above equation can have a value of ±2 and DB4 can have a value of ±4. The running disparity RD1 may be RD+1 or RD−1 and RD3 may be RD+3 or RD−3. The polarities of the above parameters can be ignored for purposes of the above two disparity equations because the complementation function CMPLP8 enforces compliance.

### *a) DB4, Block Disparity of Four for Encoding*

The set of three primary vectors FV3u with a positive block disparity of four is illustrated in FIG. 11(R) and the set of 15 primary vectors with a negative block disparity of four is illustrated in FIG. 9 and FIG. 11(L) and belongs to the coding class FT4m and FT3m4b, respectively. The 18 vectors are listed in Table 6 and grouped for easy implementation.

**Disparity of Four, DB4**

| Name | STUVWXY K | S | Name | STUVWXY K | Coding Label |
|---|---|---|---|---|---|
| D17 | 1000100 x | 1 | D24 | 0001100 x | (W•X'•Y' + W'•X•Y'+ W'•X'•Y) •<br>(S⊕V • T'•U' + T⊕U • S'•V') |
| D33 | 1000010 x | 1 | D40 | 0001010 x | |
| D65 | 1000001 x | 1 | D72 | 0001001 x | |
| D18 | 0100100 x | 1 | D20 | 0010100 x | |
| D34 | 0100010 x | 1 | D36 | 0010010 x | |
| D66 | 0100001 x | 1 | D68 | 0010001 x | |
| D9 | 1001000 x | 1 | C126 | 0111111 1 | {S⊕T•T⊕U'•U⊕W' • (K+Y') +<br>S⊕W'•T⊕U} • W⊕X'•X⊕Y' • V |
| D10 | 0101000 x | 1 | D125 | 1011111 x | |
| D12 | 0011000 x | 1 | D123 | 1101111 x | |

**Table 6**

From the Table 6, the following equation can be extracted:

$$DB4 = \{S\oplus T\bullet T\oplus U'\bullet U\oplus W'\bullet (K+Y') + S\oplus W'\bullet T\oplus U\} \bullet W\oplus X'\bullet X\oplus Y'\bullet V + \qquad (Pn30;Pn31)$$

$$(W\bullet X'\bullet Y' + W'\bullet X\bullet Y'+ W'\bullet X'\bullet Y) \bullet (S\oplus V \bullet T'\bullet U' + T\oplus U \bullet S'\bullet V') \qquad (Pn32;Pn33)$$

*b) DB2, Block Disparity of Two for Encoding*

A set of 18 primary vectors DC4c' illustrated in FIG. 8 has a positive block disparity of two. A set of 30 primary vectors DM4u'4t' illustrated in FIG. 10 has a negative block disparity of two. The 48 vectors are listed and sorted for easy implementation in Table 7.

**Block Disparity of Two, DB2**

| Name | STUVWXY K | S | Name | STUVWXY K | Coding Label |
|---|---|---|---|---|---|
| D19 | 11001000 0 | 1 | D44 | 0011010 x | S⊕T'•T⊕U•U⊕V'•W⊕X • Y'•K' |
| D35 | 1100010 x | 1 | D28 | 0011100 x | |
| D88 | 0001101 x | 1 | D103 | 1110011 x | S⊕T'•T⊕U'•U⊕V•V⊕W'•W⊕X •Y |
| D55 | 1110110 x | 1 | D91 | 1101101 x | U⊕V•X⊕Y • S•T•W |
| D59 | 1101110 x | 1 | D87 | 1110101 x | |
| D21 | 1010100 x | 1 | D42 | 0101010 0 | S⊕T•U⊕V•W⊕X • Y'•K' |
| D22 | 0110100 0 | 1 | D41 | 1001010 x | |
| D37 | 1010010 x | 1 | D26 | 0101100 x | |
| D38 | 0110010 x | 1 | D25 | 1001100 x | |
| D61 | 1011110 x | 1 | D94 | 0111101 x | S⊕T•X⊕Y • U•V•W |
| D62 | 0111110 x | 1 | D93 | 1011101 x | |
| D49 | 1000110 x | 1 | D98 | 0100011 x | (S⊕T•U'•V'•K' + U⊕V•S'•T') • W⊕Y • X |
| D50 | 0100110 0 | 1 | D97 | 1000011 x | |
| D52 | 0010110 x | 1 | D104 | 0001011 x | |
| D56 | 0001110 x | 1 | D100 | 0010011 x | |
| D73 | 1001001 x | 1 | D110 | 0111011 x | (S⊕T'•T⊕U•U⊕X + S⊕T•U⊕X' • K') • V⊕W •Y |
| D74 | 0101001 0 | 1 | D109 | 1011011 x | |
| D81 | 1000101 x | 1 | D118 | 0110111 x | |
| D82 | 0100101 x | 1 | D117 | 1010111 x | |
| D76 | 0011001 x | 1 | D107 | 1101011 x | |
| D84 | 0010101 x | 1 | D115 | 1100111 x | |
| D67 | 1100001 x | 1 | D124 | 0011111 x | (S⊕T'•T⊕U •U⊕V' + S⊕T•U⊕V) • V⊕W'•W⊕X'•Y |
| D69 | 1010001 x | 1 | D122 | 0101111 x | |
| D70 | 0110001 x | 1 | D121 | 1001111 x | |

**Table 7**

From the Table 7, the following equation can be extracted:

*DB2 = (S⊕T'•T⊕U•U⊕V' + S⊕T•U⊕V) • V⊕W'•W⊕X' • Y +* *(P1DB2*)*
*(S⊕T'•T⊕U•U⊕X + S⊕T•U⊕X' • K') • V⊕W • Y +* *(P1DB2*;P2DB2*)*
*(S⊕T • U'•V'•K' + U⊕V • S'•T') • W⊕Y • X +* *(P2DB2*)*
*S⊕T'•T⊕U'•U⊕V•V⊕W'•W⊕X•Y + S⊕T'•T⊕U•U⊕V'•W⊕X•Y'•K' +* *(P3DB2*)*
*S⊕T•U⊕V•W⊕X • Y'•K' + S⊕T•X⊕Y • U•V•W + U⊕V•X⊕Y • S•T•W* *(n34)*

## B. Logic Equations for 8B7B Decoder

Significant circuit simplifications are enabled if the outcome of the decoding process for invalid vectors is allowed to be arbitrary. This primarily refers to vectors with disparities other than ±4, ± 2, or 0, and to vectors with violations of the leading or trailing run length limitations.

The decoding process is also simplified because of the following features:

- Full vector complementation to obtain the primary vector from an alternate vector can proceed in parallel with individual bit complementation because all 25 vectors which require individual bit changes are disparity independent and have no alternate version.
- The code has been constructed so all 71 alternate vectors with the exception of K7 have a z-value of one. The only other vectors with a z-value of one are the 34 balanced, disparity independent vectors listed in Table 2 which have no alternate version.
- All 25 vectors which require individual bit changes are balanced and have a z-value of one.
- Decoding and validity checks are independent of each other and can proceed in parallel as illustrated in FIG. 14A.

Because of the simplicity of the decoding process, no decoding table is given. Please refer to Tables 1, 2, or 8 which list all 25 vectors which require individual bit changes. Bits among the first seven encoded positions which must be complemented for decoding are marked in bold type.

### 1) Decoding Procedures

1. All vectors ending with z = 0 except K7A (00011110) are decoded by simply stripping bit z.
2. For all unbalanced vectors ending with z = 1 and the vector 00001111 (D15A), the z-bit is dropped and the leading 7 bits are complemented to obtain the decoded vector.
3. For the 34 balanced vectors of Table 2 with z = 1, the z-bit is dropped and 25 of these vectors, also listed in Table 8, require one to four individual bit changes in the leading 7 positions. For 7, 11, and 6 vectors 1, 2, and 3 bits are complemented, respectively. A single vector requires 4 bit changes.
4. The control bit K is recovered by special considerations and 4 alternate vectors associated with comma generation (K7A, K23A, K39A, K71A) follow a special rule for vector complementation.

### 2) Full Vector Complementation

A *single image, the primary vector,* of each complementary pair of vectors is created by complementing the leading 7 bits of all alternate vectors. There are two categories of alternate vectors:

- A first category of 67 alternate vectors is identified by a z-bit value of one and a bit pattern ‘stuvwxy’ other than associated with the 34 vectors illustrated in FIG. 12 and listed in Table 2. A set of equations for these vectors can be derived directly from FIG. 12. The first line in the equation CMPLA (Complement **A**lternate Vector) below represents the 18 vectors through node 4b, the second line represents the 12 vectors through node 4m, and the third line represents the 4 vectors through node 4u.
- A second category of four alternate vectors listed above under “Comma Characters for concatenated 7B8B Coding Blocks and for 16B18B Code” is identified by its position contiguously following the character C126, or C508 in the context of the 16B18B code, i.e. having one of the C126 or C508 vectors as a prefix, and having three leading zeros followed by a one and a single zero in the last four bit positions. This condition is identified in the equation for complementation below by the expression $C126PREF\bullet (w\oplus x\bullet y\bullet z + y\oplus z\bullet w\bullet x) \bullet s'\bullet t'\bullet u'\bullet v$.

The equation CMPLA for the complementation of alternate vectors can now be expressed as follows:

$$CMPLA = z\bullet\{(s\oplus t'\bullet t\oplus u\bullet u\oplus v' + s\oplus t\bullet u\oplus v)\bullet(w\bullet x'\bullet y' + w'\bullet x\bullet y' + w'\bullet x'\bullet y)\}'\bullet \tag{n7}$$

$$\{(s\oplus t\bullet u'\bullet v' + u\oplus v\bullet s'\bullet t')\bullet(w\bullet x\bullet y' + w\bullet x'\bullet y + w'\bullet x\bullet y)\}'\bullet \tag{n8}$$

$$\{(s\oplus t\bullet u\bullet v + u\oplus v\bullet s\bullet t)\bullet w`\bullet x'\bullet y'\}' + \tag{n9}$$

$$C126PREF\bullet (w\oplus x\bullet y\bullet z + y\oplus z\bullet w\bullet x)\bullet s'\bullet t'\bullet u'\bullet v \tag{NCMPLA2}$$

### 3) Individual Bit Complementation

Bit mapping from the primary coded vectors back to the source vectors is accomplished by dropping the *z*-bit and complementation of selected bits for a minority of 25 disparity independent vectors extracted from Table 2 and listed in Table 8.

In Table 8, bit values in bold type must be complemented for decoding. Bit values in italic are either identical on several rows or complementary between the left and right side. Non-italic bit values are equal on the left and right side if there is a 1 in the SY column. SY in this case stands for ‘Symmetry’, not the decoded bit values S and Y.

For the decoding of each bit, the vectors with a bold bit value for the bit column in question in Table 8 are extracted and arranged in groups with commonalities in new Tables 9S through 9Y. Explicit decoding equations are then derived from the set of coding labels. In the Tables 9S through 9Y, and 9K, bit values in bold type are complementary between the left and right side, and bit values in italic type are equal on the left and right side if there is a 1 in the SY column.

The value of a bit position before decoding of that bit can be ignored because the same bit position of a vector which is complementary in that position and equal in all other positions is an alternate or invalid vector. Alternate vectors are complemented for decoding, as an example, D8=**1**0010101 has the first bit complemented to 0, but the entire vector **0**0010101 (D87A) is complemented for decoding. However, for decoding classes which are applicable to several bits, the redundant bit is usually included to enable circuit sharing but underlined in the logic equations to indicate that it could be left out perhaps to reduce delay in a critical path.

### 25 Vectors with Changes in the first 7 bits for Decoding

| NAME | stuvwxyz | SY | NAME | stuvwxyz |
|---|---|---|---|---|
| D3 | 11000011 | | | |
| D5 | 10100011 | | | |
| D6 | 01100011 | | | |
| D2 | 01001011 | | | |
| D4 | 00100111 | | | |
| D0 | 01011001 | 1 | D127 | 10100101 |
| D1 | 10011001 | 1 | D126 | 01100101 |
| D8 | 10010101 | 1 | D119 | 00101101 |
| D16 | 10101001 | 1 | D111 | 01000111 |
| D32 | 11000101 | 1 | D95 | 00011011 |
| D64 | 10010011 | 1 | D63 | 00011101 |
| D48 | 10001101 | 1 | D79 | 00110011 |
| D80 | 10001011 | 1 | D47 | 00110101 |
| D96 | 10000111 | 1 | D31 | 00111001 |
| D112 | 00010111 | 0 | D85 | 00101011 |

**Table 8**

### 4) Logic Equations for 8B7B Bit Mapping

#### S-bit Decoding

| Name | stuvwxyz | S | SY | Name | stuvwxyz | S | Decoding Label |
|---|---|---|---|---|---|---|---|
| D8 | 10010101 | 0 | 1 | D119 | 00101101 | 1 | s⊕u•u⊕v•v⊕w • t'•x•y'z |
| D16 | 10101001 | 0 | 1 | D111 | 01000111 | 1 | s⊕t•t⊕u•t⊕x'•w⊕x•x⊕y' • v'•z |
| D32 | 11000101 | 0 | 1 | D95 | 00011011 | 1 | s⊕t'•t⊕x'•v⊕w'•w⊕x•x⊕y • u'•z |
| D64 | 10010011 | 0 | 1 | D63 | 00011101 | 1 | s⊕w•w⊕x'•x⊕y • t'•u'•v•z |
| D48 | 10001101 | 0 | 1 | D79 | 00110011 | 1 | (v⊕w'•w⊕x•x⊕y' + v⊕w•x⊕y) • s⊕u•u⊕v' • t'•z |
| D80 | 10001011 | 0 | 1 | D47 | 00110101 | 1 | |
| D96 | 10000111 | 0 | 1 | D31 | 00111001 | 1 | |
| | | | | D85 | 00101011 | 1 | s'•t'•u•v'•w•x'•y•z |

**Table 9S**

The 15 vectors which require complementation of the s-bit for decoding as indicated by a bold bit-value in the s-column of Table 8 are listed in Table 9S. The expression $\underline{s\oplus t}$ in the second row of Table 9S is retained to allow circuit sharing with U-bit decoding. The expression $\underline{s\oplus t'}$ in the third row is retained to allow circuit sharing with T-bit decoding.

$$S = s\oplus\{(v\oplus w'\bullet w\oplus x\bullet x\oplus y' + v\oplus w\bullet x\oplus y) \bullet u\oplus v' \bullet t'\bullet z + \qquad (Pn0)$$
$$\underline{s\oplus t}\bullet t\oplus u\bullet t\oplus x'\bullet w\oplus x\bullet x\oplus y' \bullet v'\bullet z + \underline{s\oplus t'}\bullet t\oplus x'\bullet v\oplus w'\bullet w\oplus x\bullet x\oplus y \bullet u'\bullet z + \qquad (Pn1)$$
$$u\oplus v\bullet v\oplus w \bullet t'\bullet x\bullet y'z + w\oplus x'\bullet x\oplus y \bullet t'\bullet u'\bullet v\bullet z + t'\bullet u\bullet v'\bullet w\bullet x'\bullet y\bullet z\} \qquad (Pn2)$$

### T-bit Decoding

| Name | stuvwxyz | T | SY | Name | stuvwxyz | T | Decoding Label |
|---|---|---|---|---|---|---|---|
| D0 | $\mathbf{010110}\mathit{01}$ | 0 | 1 | D127 | $\mathbf{101001}\mathit{01}$ | 1 | $s\oplus\underline{\underline{t\bullet}t}\oplus u\bullet u\oplus v\bullet v\oplus w'\bullet w\oplus x \bullet y'\bullet z$ |
| D32 | $\mathbf{11}\mathit{0}\mathbf{0010}\mathit{1}$ | 0 | 1 | D95 | $\mathbf{00}\mathit{0}\mathbf{1101}\mathit{1}$ | 1 | $s\oplus\underline{t'\bullet t}\oplus x'\bullet v\oplus w'\bullet w\oplus x\bullet x\oplus y \bullet u'\bullet z$ |
| D63 | $\mathit{00}\mathbf{01}\mathit{1101}$ | 1 | 1 | D119 | $\mathit{00}\mathbf{10}\mathit{1101}$ | 1 | $u\oplus v\bullet s'\bullet\underline{t'}\bullet w\bullet x\bullet y'\bullet z$ |
| | | | | D47 | $\mathbf{00110}101$ | 1 | $(x\oplus y\bullet w' + w\bullet x'\bullet y') \bullet s'\bullet\underline{t'}\bullet u\bullet v\bullet z$ |
| | | | | D79 | $\mathbf{00110}011$ | 1 | |
| | | | | D31 | $\mathbf{0011}1001$ | 1 | |

Table 9T

The 9 vectors which require complementation of the t-bit for decoding as indicated by a bold bit-value in the t-column of Table 8 are listed in Table 9T. Because the value of bit t can be ignored, the expression $s\oplus t\bullet t\oplus u$ in the first row could be replaced by $s\oplus u'$, but the full expression is retained to allow circuit sharing with V-bit and W-bit decoding. The expression $s\oplus t'\bullet t\oplus x'$ in the second row could be replaced by $s\oplus x'$ and is also retained to allow circuit sharing with S-bit decoding.

$$T = t \oplus \{s\oplus\underline{t\bullet t}\oplus u\bullet u\oplus v\bullet v\oplus w'\bullet w\oplus x \bullet y'\bullet z + s\oplus\underline{t'\bullet t}\oplus x'\bullet v\oplus w'\bullet w\oplus x\bullet x\oplus y \bullet u'\bullet z + \tag{Pn3}$$

$$(x\oplus y \bullet w' + w\bullet x'\bullet y') \bullet s'\bullet u\bullet v\bullet z + u\oplus v \bullet s'\bullet w\bullet x\bullet y'\bullet z\} \tag{Pn4}$$

### U-bit Decoding

| Name | stuvwxyz | U | SY | Name | stuvwxyz | U | Decoding Label |
|---|---|---|---|---|---|---|---|
| D16 | $\mathbf{101}\mathit{0}\mathbf{100}\mathit{1}$ | 0 | 1 | D111 | $\mathbf{010}\mathit{0}\mathbf{011}\mathit{1}$ | 1 | $s\oplus t\bullet\underline{t\oplus u}\bullet t\oplus x'\bullet w\oplus x\bullet x\oplus y' \bullet v'\bullet z$ |
| D63 | $\mathit{00011}\mathbf{10}\mathit{1}$ | 1 | 1 | D95 | $\mathit{00011}\mathbf{01}\mathit{1}$ | 1 | $x\oplus y \bullet s'\bullet t'\bullet\underline{u'}\bullet v\bullet w\bullet z$ |

Table 9U

The 4 vectors which require complementation of the u-bit for decoding as indicated by a bold bit-value in the u-column of Table 8 are listed in Table 9U. The expression $\underline{t\oplus u}$ in the first row of Table 9U is retained to allow circuit sharing with S-bit decoding.

$$U = u \oplus (s\oplus t\bullet t\oplus u\bullet t\oplus x'\bullet w\oplus x\bullet x\oplus y' \bullet v'\bullet z + x\oplus y \bullet s'\bullet t'\bullet v\bullet w\bullet z) \tag{PU1}$$

### V-bit Decoding

| Name | stuvwxyz | V | SY | Name | stuvwxyz | V | Decoding Label |
|---|---|---|---|---|---|---|---|
| D0 | $01\mathbf{0110}\mathit{01}$ | 0 | 1 | D127 | $10\mathbf{1001}\mathit{01}$ | 1 | $s\oplus t\bullet u\oplus\underline{v\bullet v}\oplus w'\bullet w\oplus x \bullet y'\bullet z$ |
| D1 | $10\mathbf{0110}\mathit{01}$ | 0 | 1 | D126 | $01\mathbf{1001}\mathit{01}$ | 1 | |
| D64 | $\mathbf{1}\mathit{0010}\mathbf{0}\mathit{11}$ | 0 | 1 | D112 | $\mathbf{0}\mathit{0010}\mathbf{1}\mathit{11}$ | 0 | $s\oplus x \bullet t'\bullet u'\bullet\underline{v}\bullet w'\bullet y\bullet z$ |
| | | | | D111 | 01000111 | 1 | $s'\bullet t\bullet u'\bullet\underline{v'}\bullet w'\bullet x\bullet y\bullet z$ |

Table 9V

The 7 vectors which require complementation of the v-bit for decoding as indicated by a bold bit-value in the v-column of Table 8 are listed in Table 9V. Because the value of bit v can be ignored, the expression $u\oplus\underline{v\bullet v}\oplus w'$ in the first row can be replaced by $u\oplus w$ but is retained to enable circuit sharing with W-bit decoding.

$$V = v \oplus (s\oplus t\bullet u\oplus \underline{v\bullet v}\oplus w'\bullet w\oplus x \bullet y'\bullet z + s\oplus x \bullet t'\bullet u'\bullet w'\bullet y\bullet z + s'\bullet t\bullet u'\bullet w'\bullet x\bullet y\bullet z) \qquad (PV1)$$

**W-bit Decoding**

| Name | stuvwxyz | W | SY | Name | stuvwxyz | W | Decoding Label |
|---|---|---|---|---|---|---|---|
| D0 | 01011001 | 0 | 1 | D127 | 10100101 | 1 | $s\oplus t\bullet u\oplus v\bullet v\oplus \underline{w'\bullet w}\oplus x \bullet y'\bullet z$ |
| D1 | 10011001 | 0 | 1 | D126 | 01100101 | 1 | |
| D2 | 01001011 | 0 | 1 | D112 | 00010111 | 1 | $t\oplus x\bullet v\oplus \underline{w\bullet w}\oplus x \bullet s'\bullet u'\bullet y\bullet z$ |

**Table 9W**

The 6 vectors which require complementation of the w-bit for decoding as indicated by a bold bit-value in the w-column of Table 8 are listed in Table 9W. Because the value of bit w can be ignored, the expression $v\oplus\underline{w'\bullet w}\oplus x$ in the first row could be replaced by $v\oplus x$ and the expression $v\oplus\underline{w\bullet w}\oplus x$ in the third row by $v\oplus x'$. The expression $v\oplus\underline{w'\bullet w}\oplus x$ is retained to allow circuit sharing with V-bit decoding and $v\oplus\underline{w\bullet w}\oplus x$ is retained because $v\oplus x'$ would require an additional XOR gate.

$$W = w \oplus (s\oplus t\bullet u\oplus v\bullet v\oplus w'\bullet w\oplus x \bullet y'\bullet z + t\oplus x\bullet v\oplus w\bullet w\oplus x \bullet s'\bullet u'\bullet y\bullet z) \qquad (PW1)$$

**X-bit Decoding**

| Name | stuvwxyz | X | SY | Name | stuvwxyz | X | Decoding Label |
|---|---|---|---|---|---|---|---|
| D4 | 00100111 | 0 | 1 | D8 | 10010101 | 0 | $s\oplus u\bullet u\oplus v\bullet u\oplus y' \bullet t'\bullet w'\bullet \underline{x}\bullet z$ |

**Table 9X**

The 2 vectors which require complementation of the x-bit for decoding as indicated by a bold bit-value in the x-column of Table 8 are listed in Table 9X.

$$X = x \oplus (s\oplus u\bullet u\oplus v\bullet u\oplus y' \bullet t'\bullet w'\bullet z) \qquad (NCMPLx1)$$

**Y-bit Decoding**

| Name | stuvwxyz | Y | SY | Name | stuvwxyz | Y | Decoding Label |
|---|---|---|---|---|---|---|---|
| D2 | 01001011 | 0 | 1 | D4 | 00100111 | | $t\oplus u\bullet t\oplus x\bullet w\oplus x \bullet s'\bullet v'\bullet \underline{y}\bullet z$ |
| D3 | 11000011 | 0 | | | | | $(s\bullet t\bullet u' + s\bullet t'\bullet u + s'\bullet t\bullet u) \bullet v'\bullet w'\bullet x'\bullet \underline{y}\bullet z$ |
| D5 | 10100011 | 0 | | | | | |
| D6 | 01100011 | 0 | | | | | |
| | | | | D127 | 10100101 | 1 | $(s\oplus t\bullet w' + s'\bullet t'\bullet w) \bullet u\bullet v'\bullet x\bullet \underline{y'}\bullet z$ |
| | | | | D126 | 01100101 | 1 | |
| | | | | D119 | 00101101 | 1 | |

**Table 9Y**

The 8 vectors which require complementation of the y-bit for decoding as indicated by a bold bit-value in the y-column of Table 8 are listed in Table 9Y.

$$
\begin{aligned}
&Y = y \oplus \{(s{\bullet}t{\bullet}u' + s{\bullet}t'{\bullet}u + s'{\bullet}t{\bullet}u) \bullet v'{\bullet}w'{\bullet}x'{\bullet}z + && \text{(Pn5)}\\
&\quad (s{\oplus}t{\bullet}w' + s'{\bullet}t'{\bullet}w) \bullet u{\bullet}v'{\bullet}x{\bullet}z + && \text{(Pn6)}\\
&\quad t{\oplus}u{\bullet}t{\oplus}x{\bullet}w{\oplus}x \bullet s'{\bullet}v'{\bullet}z\} && \text{(NCMPLy1*)}
\end{aligned}
$$

**K-bit Decoding**

| Name | stuvwxyz | SY | Name | stuvwxyz | Decoding Label |
|---|---|---|---|---|---|
| K19 | **1**_1_**0**_01001_ | 1 | K22 | **0**_1101001_ | s⊕u • t•v’•w•x’•y’•z |
| K42 | **01010101** | 1 | K85 | **10101010** | s⊕t•t⊕u•u⊕v•v⊕w•w⊕x•x⊕y•y⊕z |
| K50 | _010_**0110**_1_ | 1 | K74 | _010_**1001**_1_ | v⊕w•w⊕x’•x⊕y • s’•t•u’•z |
| C126P | **01111110** |  | C126A | **10000001** | s⊕t•t⊕u’•u⊕v’•v⊕w’•w⊕x’•x⊕y’•y⊕z |

**Table 9Y**

The 8 true control vectors with a decoded K-bit value of one are listed in Table 9K. The first 7 bits of the vector C126A are complemented along with the unbalanced, alternate data vectors with a z-bit value of one. The equation for the decoding of the K-bit below is derived from the coding labels of Table 9K.

$$
\begin{aligned}
&K = s{\oplus}t{\bullet}t{\oplus}u{\bullet}u{\oplus}v{\bullet}v{\oplus}w{\bullet}w{\oplus}x{\bullet}x{\oplus}y{\bullet}y{\oplus}z + && \text{(PK*)}\\
&\quad s{\oplus}t{\bullet}t{\oplus}u'{\bullet}u{\oplus}v'{\bullet}v{\oplus}w'{\bullet}w{\oplus}x'{\bullet}x{\oplus}y'{\bullet}y{\oplus}z + && \text{(NC126)}\\
&\quad v{\oplus}w{\bullet}w{\oplus}x'{\bullet}x{\oplus}y \bullet s'{\bullet}t{\bullet}u'{\bullet}z + s{\oplus}u \bullet t{\bullet}v'{\bullet}w{\bullet}x'{\bullet}y'{\bullet}z && \text{(PK*)}
\end{aligned}
$$

## C. Error Checking

### 1) Invalid 8B Vectors

The 8B alphabet of FIGS. 1A.1, 1A.2, 1B, 1C.1, and 1C.2 comprises 202 valid vectors, so there are a total of 54 invalid 8B vectors. One invalid vector I255P ends with node 8h in FIG. 5(R), eight invalid vectors end with node 8v, ten with node 8c, and eight with node 8u. All complements of these 27 vectors are also invalid. All 54 invalid vectors are listed in Table 10 and the equation for invalid 8B characters INVAL8 is derived from the coding labels listed there.

## 54 Invalid Vectors

| Name | DB Class | DB | stuvwxyz | stuvwxyz | Decoding Label |
|---|---|---|---|---|---|
| I31/I224 | U/M | 2 | **1111**1000 | **0000**0111 | s⊕t’•t⊕u’•u⊕v’ •<br>(v⊕w•w⊕x’•x⊕y’•y⊕z’)’ |
| I47/I208 | U/M | 2 | **1111**0100 | **0000**1011 | |
| I63/I192 | C/T | 4 | **1111**1100 | **0000**0011 | |
| I79/I176 | U/M | 2 | **1111**0010 | **0000**1101 | |
| I95/I160 | C/T | 4 | **1111**1010 | **0000**0101 | |
| I111/I144 | C/T | 4 | **1111**0110 | **0000**1001 | |
| I127/I128 | V/Q | 6 | **1111**1110 | **0000**0001 | |
| I143/I112 | U/M | 2 | **1111**0001 | **0000**1110 | |
| I159/I96 | C/T | 4 | **1111**1001 | **0000**0110 | |
| I175/I80 | C/T | 4 | **1111**0101 | **0000**1010 | |
| I191/I64 | V/Q | 6 | **1111**1101 | **0000**0010 | |
| I207/I48 | C/T | 4 | **1111**0011 | **0000**1100 | |
| I223/I32 | V/Q | 6 | **1111**1011 | **0000**0100 | |
| I239/I16 | V/Q | 6 | **1111**0111 | **0000**1000 | |
| I255/I0 | H/S | 8 | **1111**1111 | **0000**0000 | |
| I241/I14 | U/M | 2 | 10**001111** | 01**110000** | (s⊕t•u⊕v’ + s⊕t’•u⊕v) •<br>v⊕w•w⊕x’•x⊕y’•y⊕z’ |
| I242/I13 | U/M | 2 | 01**001111** | 10**110000** | |
| I244/I11 | U/M | 2 | **00**10**1111** | **11**01**0000** | |
| I247/I8 | V/Q | 6 | **11**10**1111** | **00**01**0000** | |
| I248/I7 | U/M | 2 | 000**11111** | 111**00000** | v⊕w’•w⊕x’•x⊕y’•y⊕z’ |
| I249/I6 | C/T | 4 | 100**11111** | 011**00000** | |
| I250/I5 | C/T | 4 | 010**11111** | 101**00000** | |
| I251/4 | V/Q | 6 | 110**11111** | 001**00000** | |
| I252/I3 | C/T | 4 | 001**11111** | 110**00000** | |
| I253/I2 | V/Q | 6 | 101**11111** | 010**00000** | |
| I254/I1 | V/Q | 6 | 011**11111** | 100**00000** | |
| *I255/I0 | H/S | 8 | 111**11111** | 000**00000** | |
| I119/I136 | C/T | 4 | **11101110** | **00010001** | s⊕t’•t⊕u’•u⊕w’•w⊕x’•x⊕y’•y⊕z |
| *I127/I128 | V/Q | 6 | **11111110** | **00000001** | |

**Table 10**

*INVAL8* = s⊕t’•t⊕u’•u⊕v’ • (v⊕w•w⊕x’•x⊕y’•y⊕z’)’ + *(n10)*
(s⊕t•u⊕v’ + s⊕t’•u⊕v) • v⊕w•w⊕x’•x⊕y’•y⊕z’ + *(n11)*
s⊕t’•t⊕u’•u⊕w’•w⊕x’•x⊕y’•y⊕z + v⊕w’•w⊕x’•x⊕y’•y⊕z’ *(PINVALB*)*

### 2) Disparity Checks on Decoding

Some important applications of this code will not be helped much by disparity monitoring and not implement it. As an example, a computer bus as described in Ref. 3 requires separate extensive error checking and correction facilities with low latency. Disparity errors often show up with some delay after one or more disparity independent coding blocks have passed.

Other applications may implement simplified monitoring circuits which miss a small fraction of disparity violations, or they may tolerate some double counts, or they may want to deactivate monitoring until a reliable running disparity value is reestablished after an error indication. Below there are some expressions defined which can be used as building blocks for any such monitoring process.

At a receiver, the vector sequences are monitored to see whether they still conform to the rules imposed by the encoder. A single error in transmission will always cause a violation of the disparity rules without necessarily generating an invalid vector as described above. In a mixture of balanced vectors, and vectors with a block disparity of ±2 or ±4, the running disparity in the absence of errors is constrained to values of ±1 and ±3 at the vector boundaries and a transmission error is not always immediately detectable by just adding and subtracting the cumulative block disparities to see whether the actual running disparity of the received vector sequence meets the above constraints. The following rules assume that the error, if any, occurred before the two-vector blocks under consideration. If an error is present in the block itself, a duplicate error indication may occur later because the value of the original running disparity following an error is uncertain. The rules apply to any mixture of vectors in the sequence such as 6B, 8B, 10B, or other vectors with compatible disparity characteristics.

An error is flagged if the required polarity of the entry disparity of a received coded block does not match the polarity of the running disparity at the start of that block.

### 3) Equations for Required Disparity on Decoding (DR)

*a) Positive Required Disparity PDR*

Any received vector with five or more zeros or a leading run of four zeros requires a positive entry disparity, regardless whether the vector is valid or not. The primary pseudo control characters K7P, K23P, K39P, and K71P with a C126 prefix (C126PREF) require also a positive entry disparity but this rule can be ignored for the general case because this vector position might at the users choice be assigned to a data vector with the same bit pattern and no disparity dependence. The remaining vectors belong to one of the following three groups:

- 3 or 4 zeros in the leading 4 bit positions combined with 2 or more zeros in the last 4 positions.
- 2 or more zeros in the leading 4 bit positions combined with 3 or 4 zeros in the last 4 positions.
- 4 leading zeros

$$PDR = (s'\bullet t'\bullet u'+s'\bullet t'\bullet v'+s'\bullet u'\bullet v'+t'\bullet u'\bullet v')\bullet(w'\bullet x'+w'\bullet y'+w'\bullet z'+x'\bullet y'+x'\bullet z'+y'\bullet z') + (s'\bullet t'+s'\bullet u'+s'\bullet v'+t'\bullet u'+t'\bullet v'+u'\bullet v')\bullet(w'\bullet x'\bullet y'+w'\bullet x'\bullet z'+w'\bullet y'\bullet z'+x'\bullet y'\bullet z') +s'\bullet t'\bullet u'\bullet v'$$

*b) Negative Required Disparity NDR*

Any received vector with five or more ones or a leading run of four ones requires a negative entry disparity, regardless whether the vector is valid or not. The alternate pseudo control characters K7A, K23A, K39A, and K71A with a C126 prefix require also a negative entry disparity and can be ignored for the same reason given for PDR above. The remaining vectors belong to one of the following three groups:

- 3 or 4 ones in the leading 4 bit positions combined with 2 or more ones in the last 4 positions.
- 2 or more ones in the leading 4 bit positions combined with 3 or 4 ones in the last 4 positions.
- 4 leading ones

$$NDR = (s\bullet t\bullet u+s\bullet t\bullet v+s\bullet u\bullet v+t\bullet u\bullet v)\bullet(w\bullet x+w\bullet y+w\bullet z+x\bullet y+x\bullet z+y\bullet z) + (s\bullet t+s\bullet u+s\bullet v+t\bullet u+t\bullet v+u\bullet v)\bullet(w\bullet x\bullet y+w\bullet x\bullet z+w\bullet y\bullet z+x\bullet y\bullet z) + s\bullet t\bullet u\bullet v$$

**4) Equations for Running Disparity on Decoding (RD)**

The running disparity is determined by the characteristics of the most recent one or two disparity dependent blocks. Quicker recovery of the running disparity is possible by looking at the three most recent disparity dependent vectors, but the added complexity is probably not worthwhile for most applications. Disparity independent blocks are ignored. From the state diagram of FIG. 15 it is evident that after a block disparity of 4 (DB4), the polarity (PRD/NRD) is known, but not the arithmetic value (RD1/RD3). It also shows that the arithmetic value is RD1 after any block with a disparity of 2 (DB2). The running disparity is at +1 after DB2 of either polarity followed by DB2 with a positive disparity or the vector D15A, and the running disparity is at –1 after DB2 of either polarity followed by DB2 with a negative disparity or the vector D15P.

The Table 11 illustrates how the running disparity can be initially established or reestablished after an error. The following acronyms are used:

PRD = Positive Running Disparity    NRD = Negative Running Disparity
PDB4 = Positive Block Disparity of 4    NDB4 = Negative Block Disparity of 4
PDB2 = Positive Block Disparity of 2    NDB2 = Negative Block Disparity of 2
RD1/RD3 = Arithmetic value of running disparity is equal 1 or 3, respectively
D15P = 11110000    D15A = 00001111

The appended letter L refers to the next preceding disparity dependent block.

**Running Disparities PRD, NRD, RD1**

| PRD | NRD | RD1 | RD3 | PDB4 | NDB4 | PDB2 | NDB2 | D15A | D15P | PDB2L | NDB2L | RD1L | RD3L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | 1 | | | | | | | | | |
| 1 | | | | | | 1 | | | | 1 | 1 | | |
| 1 | | | | | | | | 1 | | 1 | | | |
| | 1 | | | | 1 | | | | | | | | |
| | 1 | | | | | | 1 | | | 1 | 1 | | |
| | 1 | | | | | | | | 1 | | 1 | | |
| | | 1 | | | | 1 | 1 | | | | | | |
| | | 1 | | 1 | 1 | | | | | | | | 1 |
| | | | 1 | 1 | 1 | | | | | | | 1 | |

**Table 11**

The equations below for the polarity and the arithmetic value of the running disparity are extracted from Table 11.

$$PRD = PDB4 + PDB2 \bullet (PDB2L + NDB2L) + D15A \bullet PDB2L$$
$$NRD = NDB4 + NDB2 \bullet (PDB2L + NDB2L) + D15P \bullet NDB2L$$
$$RD1 = PDB2 + NDB2 + (PDB4 + NDB4) \bullet RD3L$$
$$RD3 = (PDB4 + NDB4) \bullet RD1L$$

### 5) Equations for Block Disparity (BD)

For the block disparity, invalid vectors are considered as well. Vectors with more than six ones or zeros are lumped together with vectors of a disparity of four. Any vector other than D15P or D15A with four leading ones or zeros is invalid. If such a vector is received, it is assumed for classification purposes that originally there were only three ones or three zeros, respectively. Similarly, any vector with five trailing ones or zeros is invalid. Therefore, for vectors with four trailing ones or zeros, it is assumed that the preceding bit ‘v’ has a complementary value.

*a) Positive Block Disparity of Four PBD4*

All vectors with six ore more bits with a value of one are part of this set. These vectors end with nodes 8h, 8v, or 8c in the trellis of FIG. 5(R). The vectors belong to one of the following two groups:

- 3 or 4 ones in the leading 4 bit positions combined with 3 or 4 ones in the trailing 4 positions.
- 2 or more ones in the leading 4 bit positions combined with 4 ones in the trailing 4 positions.

Note that a vector with four leading ones followed by anything other than four trailing zeros is invalid.

$$\begin{aligned}PBD4 = {} & (s\bullet t\bullet u+s\bullet t\bullet v+s\bullet u\bullet v+t\bullet u\bullet v)\bullet(w\bullet x\bullet y+w\bullet x\bullet z+w\bullet y\bullet z+x\bullet y\bullet z) + \\ & (s\bullet t+s\bullet u+s\bullet v+t\bullet u+t\bullet v+u\bullet v)\bullet w\bullet x\bullet y\bullet z\end{aligned}$$

*b) Positive Block Disparity of Two PBD2*

This set includes all vectors with exactly 5 ones ending with node 8u in FIG. 5(R). The vectors belong to one of the following three groups:

- 3 or 4 ones in the leading 4 bit positions combined with 2 ones and 2 zeros in the trailing 4 positions. The 4 leading ones are assumed to be generated by
- 2 ones and 2 zeros in the leading 4 bit positions combined with 3 ones and 1 zero in the trailing 4 positions.

$$\begin{aligned}PBD2 = {} & (s\bullet t\bullet u+s\bullet t\bullet v+s\bullet u\bullet v+t\bullet u\bullet v) \bullet (w\oplus z\bullet x\oplus y + w\oplus x\bullet y\oplus z) + \\ & (s\oplus v\bullet t\oplus u + s\oplus t\bullet u\oplus v) \bullet (w\bullet x\bullet y+w\bullet x\bullet z+w\bullet y\bullet z+x\bullet y\bullet z)\end{aligned}$$

*c) Negative Block Disparity of Two NBD2*

This includes all vectors with exactly 5 zeros ending with node 8m in FIG. 5(R). The vectors belong to one of the following three groups:

- 3 or 4 zeros in the leading 4 bit positions combined with 2 ones and 2 zeros in the trailing 4 positions.
- 2 ones and 2 zeros in the leading 4 bit positions combined with 1 one and 3 zeros in the trailing 4 positions.

$$\begin{aligned}NBD2 = {} & (s'\bullet t'\bullet u'+s'\bullet t'\bullet v'+s'\bullet u'\bullet v'+t'\bullet u'\bullet v') \bullet (w\oplus z\bullet x\oplus y + w\oplus x\bullet y\oplus z) + \\ & (s\oplus v\bullet t\oplus u + s\oplus t\bullet u\oplus v) \bullet (w'\bullet x'\bullet y'+w'\bullet x'\bullet z'+w'\bullet y'\bullet z'+x'\bullet y'\bullet z')\end{aligned}$$

*d) Negative Block Disparity of Four NBD4*

All vectors with six ore more bits with a value of zero are part of this set. These vectors end with nodes 8s, 8q, or 8t in the trellis of FIG. 5(R). The vectors belong to one of the following two groups:

- 3 or 4 zeros in the leading 4 bit positions combined with 3 or 4 zeros in the trailing 4 positions.
- 2 or more zeros in the leading 4 bit positions combined with 4 zeros in the trailing 4 positions.

Note that a vector with four leading zeros followed by anything other than four trailing ones is invalid.

$$\begin{aligned}NBD4 = {} & (s'\bullet t'\bullet u'+s'\bullet t'\bullet v'+s'\bullet u'\bullet v'+t'\bullet u'\bullet v')\bullet(w'\bullet x'\bullet y'+w'\bullet x'\bullet z'+w'\bullet y'\bullet z'+x'\bullet y'\bullet z') + \\ & (s'\bullet t'+s'\bullet u'+s'\bullet v'+t'\bullet u'+t'\bullet v'+u'\bullet v')\bullet w'\bullet x'\bullet y'\bullet z'\end{aligned}$$

For some of the above equations, the number of logic levels can be reduced at the cost of extra gates by merging the vector sets used for the definition of the expressions, e.g. for (PBD2+NBD2), for (PBD2+D15A), and for (NBD2+D15P).

## III. CIRCUIT IMPLEMENTATION

For the circuit implementation, it is assumed that all inputs are available in complementary form, i.e. both the +L2 and –L2 outputs of the input register latches are made available. Nevertheless, the assumption is that the –L2 outputs are slightly delayed relative to the +L2 outputs. If this is not true, minor restructuring of the circuit is needed for minimum delay.

The circuit diagrams show only NAND, NOR, INV, XOR, and XNOR gates. The use of AND and OR gates has been avoided because of their increased delays and to better keep track of the number of logic levels. For the NAND and NOR gates, the upper inputs of the logic symbols usually have less delay than the lower ones. The presumed critical paths are therefore routed through the top inputs. The wire routing also assumes that XNOR delays are shorter than XOR delays. The circuit diagrams show no complex gates to allow maximum circuit sharing. The logic processing programs will introduce complex gates automatically where appropriate and merge some of the inverting gates followed by an inverter.

There is some leeway in the definition of the basic logic equations and in the partitioning of the longer expressions to match the fan-in limitations of the gates. Variations in these choices leads to different ranges in circuit sharing and circuit counts and no assurance can be given that the circuit presented is minimum area. Another reason for equivalent different circuit implementation and opportunity to slightly improve the design is the selection of the specific redundant factors in the decoding circuits. In circuit areas which are suspected to be at the upper end of circuit delay, the circuit count has occasionally been increased to reduce delay primarily by reducing the fan-in of gates in the critical path by adding extra gates in non-critical paths. For delay considerations, both XOR and XNOR gates have been used at the input to generate both polarities and some of those gates can be replaced by inverters driven from the gate of opposite polarity if they are not part of any critical timing path.

An attempt was also made to minimize the number of outputs after 4 levels of logic or the number of inputs 3 logic levels from the end to facilitate pipe-lining.

Note that some parts of the equations are not present explicitly in the circuit diagrams. If so, they have been merged with other functions in a single gate to reduce overall circuit delay. As an example, the signal CMPLA for the complementation of the leading 7 bits of alternate vectors in the decoder circuit of FIG. 19B is represented by the 2 signals NCMPLA1 and NCMPLA2 which are merged with the signals for individual bit complementation.

## A. Encoding

### 1)Block Diagram for Encoding

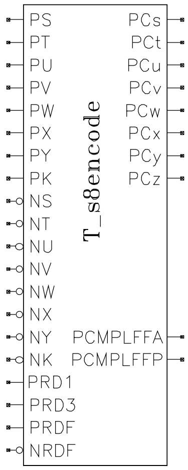


FIG. 16

The block diagram for the 7B8B encoding circuit with all inputs and outputs is shown in FIG. 16. The output PCMPLFFA complements the arithmetic flip-flop described above under Disparity Control. The outputs of flip-flop “A” are the PRD1 and PRD3 inputs for the next cycle. The output PCMPLFFP complements the polarity flip-flop, the output of which are the inputs PRDF and NRDF for the next cycle.

### 2) Gate Level Circuit Diagram for Encoding

A gate-level circuit diagram of the encoder is shown in FIGS. 17A and 17B which represent a single circuit with net sharing. FIG. 17A shows the circuit required for bit encoding and FIG. 17B shows the disparity control circuit without the two flip-flops which keep track of the running disparity. The upper right side of FIG. 17B shows the last two gate levels for bit encoding performing selective bit (NCx1, where x = s, t, u, v, w, x, or y) or full vector (NPRDFaNDR, NRDFaPDR) complementation. Selective bit and full vector complementation are orthogonal functions, i.e. no individual bits are changed when a full vector is complemented and vice-versa. This feature of the code allows the merger of both types of signals in a single OR function.

As pointed out above, a shorter delay was generally preferred over minor additions to area. As an example, in the logic paths for the signals PCMPLFFP and PCMPLFFA near the lower right corner of FIG. 17B, three parallel NAND2 gates replace a single gate to eliminate one OR gating level in each path to reduce the logic depth to 6 levels. Shorter paths for these two signals are desirable because they each control a complementing flip-flop with a MUX input which adds to the setup time. As a result, the total delay conforms to the limit of 7 levels in all other parts of the codec.

### 3) Notation for Net Names in the Encoding Circuit Diagrams

Some abbreviated signal and wire names are used in the circuits for convenience and brevity and to avoid special symbols which are not compatible with the logic design system.

In the encoding circuit, the letters ‘a’ and ‘o’ within net-names refer to the Boolean AND and OR functions, respectively, but in most cases, the AND operator is omitted. The letter ‘n’ within a name negates the preceding parameter. The letters ‘e’ and ‘u’ represent the symbols ‘=’ and ‘≠', respectively. The capital letters “STUVWXY” represent the uncoded input bits and the lower case letters “stuvwxyz” represent the coded format, usually prefixed with C(oded) because some chip design and simulation programs do not distinguish between upper and lower case letters. The lower case letter “n” followed by a number refers to a net number. Leading capital letters “P” or “N” refer to logic functions which are true at the upper or lower logic level, respectively. Numbered net names such as n45, are true at the lower level and take a P prefix if true at the upper level, e.g. Pn1.

# 7B8B Bit Encoding

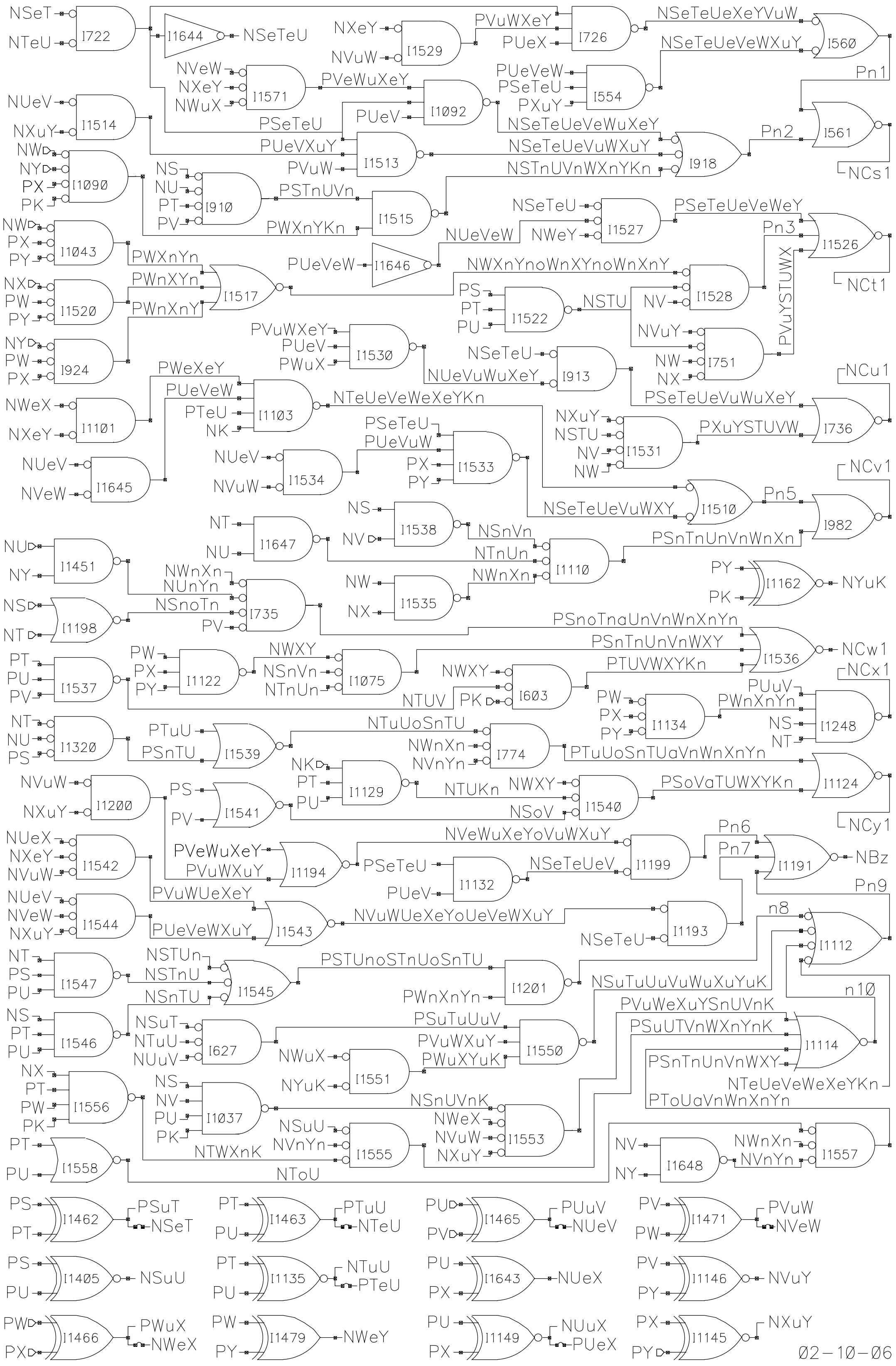


**FIG.17A**

# 7B8B Disparity Control

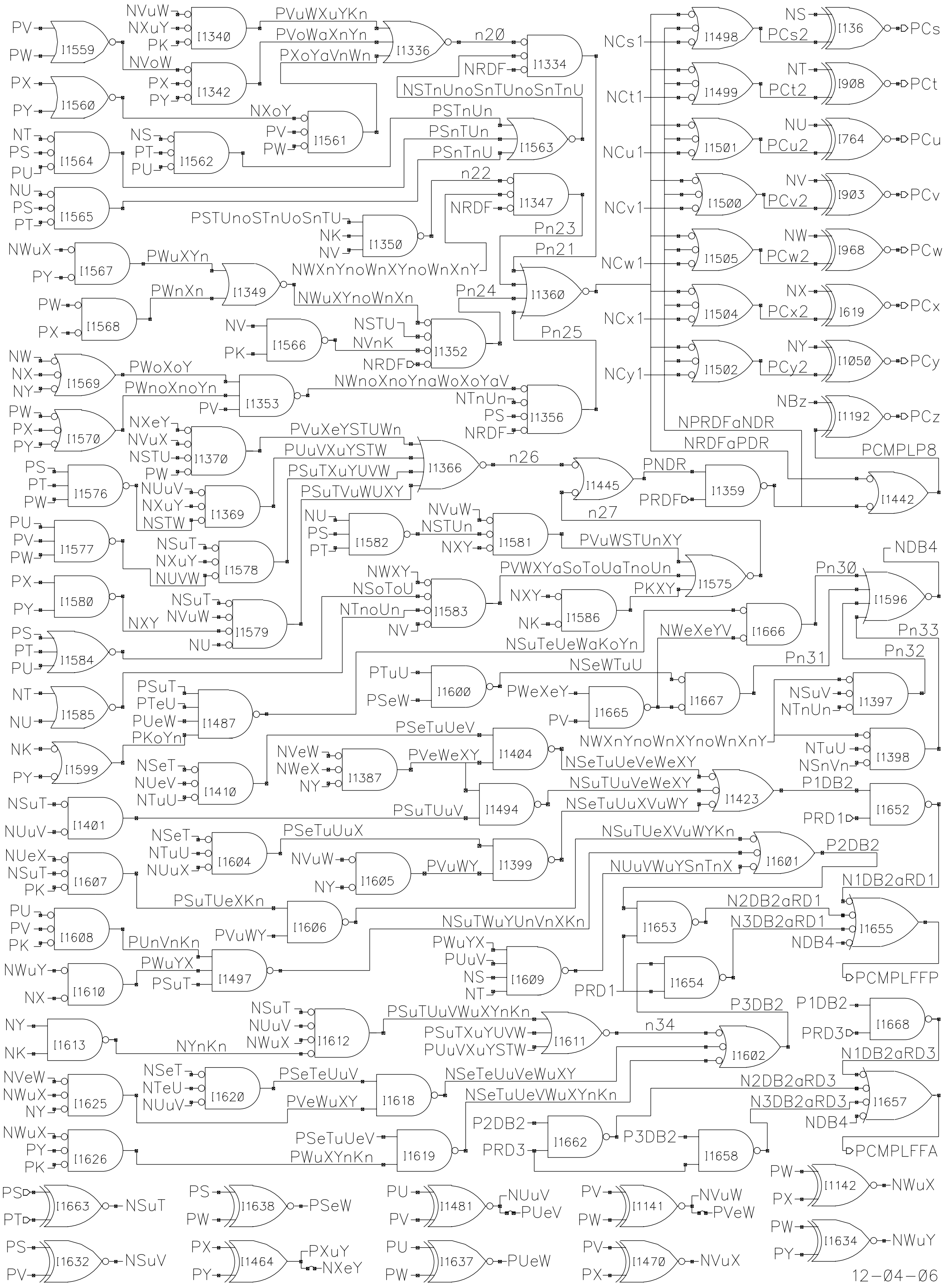


**FIG.17B**

### 4) Gate Count, Circuit Delays and Pipe-Lining for Encoding

The encoder comprises 203 gates and two flip-flops (not shown) to keep track of the disparity. No logic path exceeds 7 gates. All gates are of the inverting type with shorter delay except some XOR gates which for most power and loading levels have comparable or only slightly more delay than XNOR gates.

The circuit presented has been structured for easy forward pipe-lining for fast operation at the cost of a few extra gates. If a first encoding step is limited to six logic levels, the 8 trailing EXCLUSIVE OR functions for the coded bits can be moved into a second cycle. The first encoding step can be reduced to five gating levels, if the OR functions immediately before the XOR and the last gate in the PCMPLFFP and PCMPLFFA path are also moved to a second step. A reduction to four gating levels in the first step requires additionally:

- Minor modifications in the leading segments of the t, u, v, w, and PDB4 paths.
- Moving the NOR gates driving NCs1 and NCT1 at the top right side of FIG. 17A into the second cycle.
- Moving the trailing gates driving NPRDFaNDR and NRDFaPDR into the second step.
- Moving the trailing two gating levels for PCMPLFFA and PCMPLFFP into the second step.

A further delay reduction can be accomplished by itself or in combination with any of the above versions by minor circuit modifications and moving the leading EXCLUSIVE OR functions into the preceding clock cycle in the data source path.

## B. Decoding

### 1) Block Diagram for 8B7B Decoding

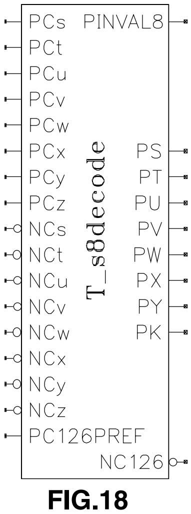


FIG.18

The block diagram for the 8B7B decoding circuit with all inputs and outputs is shown in FIG. 18. A gate-level circuit diagram of the decoder and the validity checks according to the equations derived above is shown in FIGS. 19A and 19B which represent a single circuit with net sharing. The comments given with respect to the encoding circuits generally are applicable for the decoding circuits as well.

### 2) Notation for Net Names in the Decoding Circuit Diagrams

The notation used in the decoding diagrams is analogous to that of the encoding circuit but lower case letters for logic functions are exchanged for upper case and vice versa. The letters ‘A’ and ‘O’ within net-names refer to the Boolean AND and OR functions, respectively. The letter ‘N’ within a name negates the preceding parameter. The letters ‘E’ and ‘U’ represent the symbols ‘=’ and ‘≠', respectively.

### 3) Gate Level Circuit Diagram for Decoding

FIG. 19A shows the leading sections of the circuits for individual bit decoding (STUVWXYK).

FIG. 19B shows the last two gating levels for bit decoding at the top right side. These circuits perform individual bit complementation (NCMPL*1, *=s, t, u, v, w, x, or y) or alternate vector complementation (NCMPLA1, NCMPLA2) which are all orthogonal as explained above under encoding. The bottom of FIG. 19B shows the vector validity check. The shared EXCLUSIVE OR functions of both decoding diagrams are shown on the left side. Again, inverters can be substituted for some of these gates depending on speed requirements.

No circuits are shown for disparity monitoring for the reason stated above in chapter C., 2).

### 4) Gate Count, Circuit Delays and Pipe-Lining for Decoding

The decoder comprises 145 gates. No logic path exceeds seven gates, all of the inverting type except some XOR gates. The INVAL8 path is five gating levels, and the PK path is four gating levels.

For fast operation, the circuit presented has been structured for easy forward pipe-lining at the cost of a few extra gates similar to the encoding circuit.

## 8B7B Bit Decoding

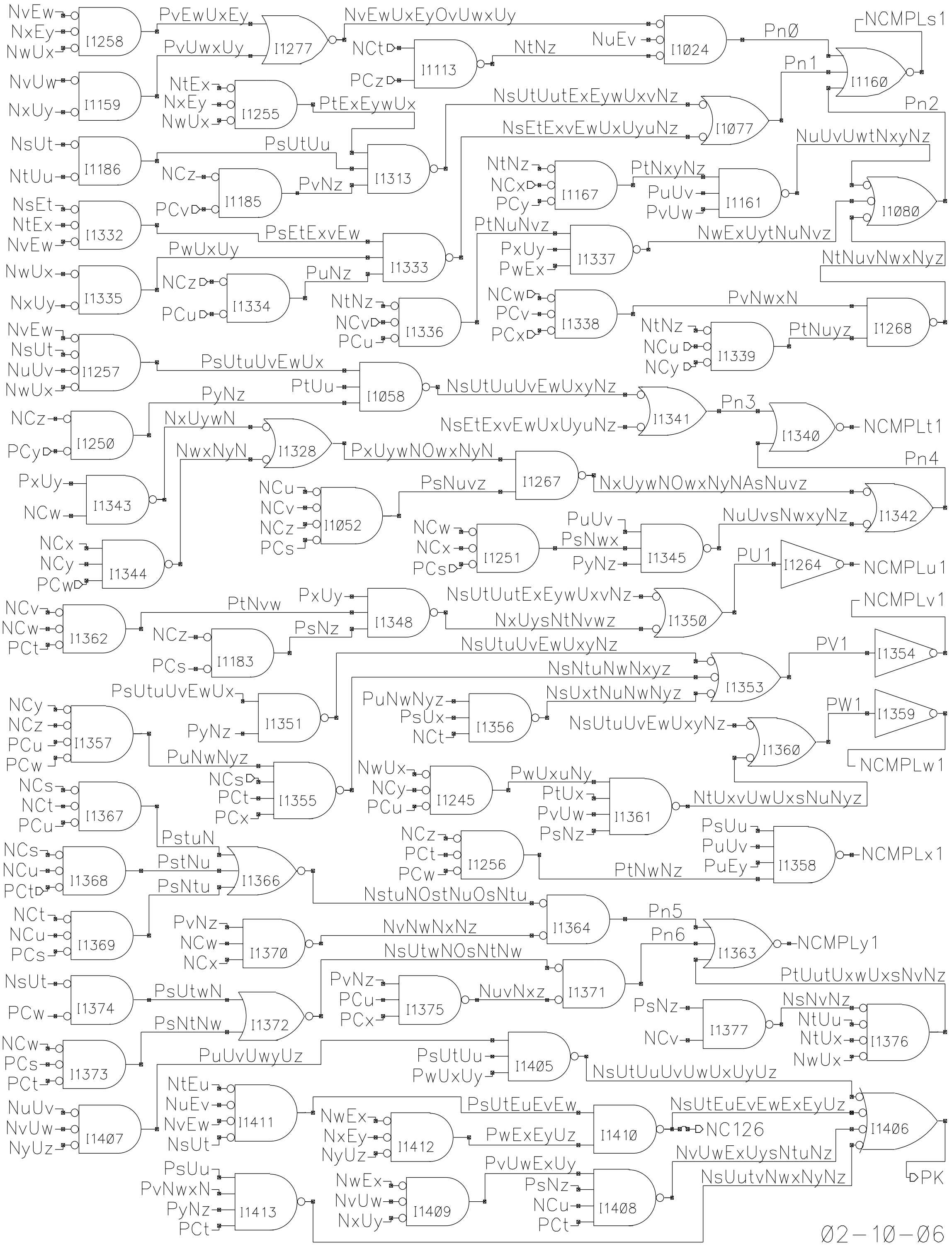


FIG.19A

# 8B7B Bit Complementation

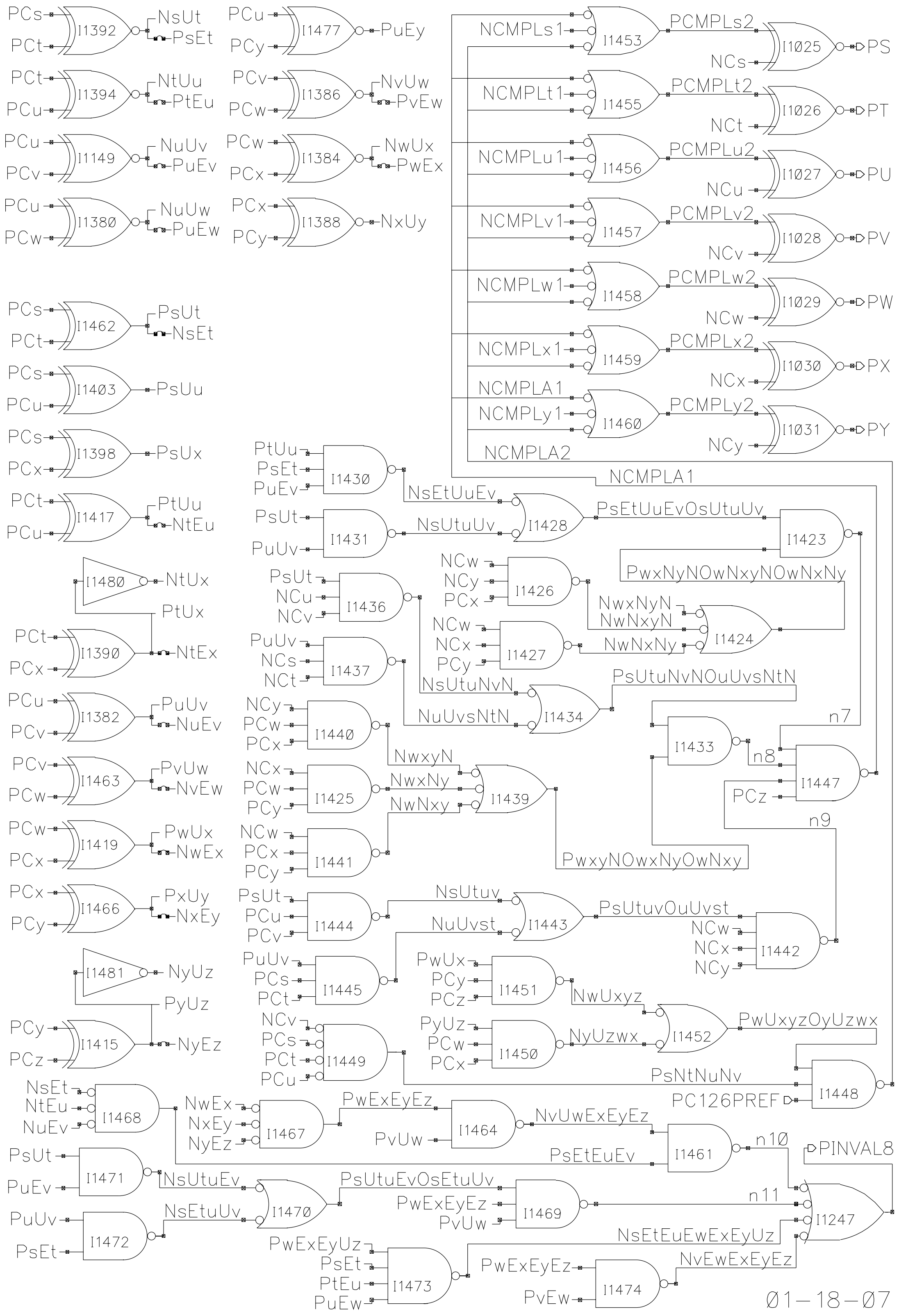


**FIG.19B**

# IV. Conclusion

The 7B8B encoder and decoder can be implemented with far fewer gates than one would expect, about twice as many as for the traditional 8B10B code. The reason for this is the availability of vectors with a disparity of four which allow more source vectors to be mapped directly into the coded format without individual bit changes. The maximum number of concatenated gates in the critical path is the same as for 8B10B code.

The 9B10B and 7B8B components of the 16B18B code can be used as stand alone codes or in combination with the 5B6B and 3B4B components of the 8B10B code and 1B2B Manchester codes.

An attractive application of the 7B8B code is for busses which may be many dozens to hundreds of lines wide if the bus width is an integer multiple of seven or one or two bits less and if it is desirable to have a serial transmission rate which is a modulo two multiple of the bus rate. Such applications usually are equipped with receivers incorporating Decision Feedback Equalization (DFE). This code limits the recovery time of DFE circuits from an error because strings of alternating ones and zeros are limited to less than two vectors.

The tables and equations have only been manually checked, and no programmed computer checks have been run so far. So the possibility of errors is there. However, the basic coding principles are sound and detail errors can be corrected by engineers with ordinary skills using the methods described in this report or other techniques. A user may also want to make minor modifications for a better match for a specific application.

## References

1. WIDMER, A. X., ‘Partitioned dc Balanced (0,6) 16B/18B Transmission Code with Error Correction’, US Patent 6,198,413 B1, March 6, 2001
2. WIDMER, A. X., ‘dc Balanced 7B/8B, 9B/10B, and Partitioned dc Balanced 12B/14B, 17B/20B, and 16B/18B Transmission Codes’, US Patent 6,614,369 B1, Sep. 2, 2003
3. WIDMER, A. X., ‘Error Correction with Low Latency for Bus Structures’, US Patent 6,978,416 B2, Dec. 20, 2005
4. WIDMER, A. X., ‘The ANSI Fibre Channel Transmission Code’, IBM Research Report RC 18855, 4/23/93
5. FRANASZEK, P. A. and WIDMER, A. X., ‘Byte Oriented dcBalanced (0,4) 8B/10B Transmission Code’, US Patent 4,486,739, Dec. 4, 1984
6. ALEXANDER, J. R. and NAGRA, A. S. T., ‘Transformation of binary coded signals into a form having lower disparity,’ British Patent 1540617, 14 Feb. 1979
7. P. COCHRANE, P., BROOKS, R., and DAWES, R., ‘A High Reliability 565 Mbit/s Trunk Transmission System’, ‘IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL.SAC-4, NO.9, December 1986, pp. 1396-1403
8. WILLIAMSON, R. L. and M. CHOWN, M., ‘The NL1 Submarine System’, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL.SAC-1, NO.3, APRIL 1983, pp. 454-458
9. SHARLAND, A. J. and STEVENSON, A., ‘A simple in-service error detection scheme based on the statistical properties of line codes for optical fibre Systems’, INT. J. ELECTRONICS, 1983, VOL. 55, NO. 1, 3-33
10. CATTERMOLE, K. W., ‘Principles of digital line coding’, INT. J. ELECTRONICS, 1983, VOL. 55, NO. 1, 3-33
11. BROOKS, R. M. and JESSOP, A., ‘Line coding for optical fibre systems’, INT. J. ELECTRONICS, 1983, VOL. 55, NO. 1, 81-120

Reference File Locations:

Full Report (Frame Maker): /homes/axw/widmer/doc/coding/Code7B8B-RC
Circuit Diagrams (Cadence cteCds): define ether/homes/axw/widmer/artist/serdesg
→ ether → s8encode, s8decode