CONSTRUCTION OF SIMPLE AND PROVABLY ROBUST 4LZS CONTROL CODES

by

Richard Cam, Ph.D.

PMC-Sierra, Inc.
105-8555 Baxter Place
Burnaby, B.C.
Canada V5A 4V7

Telephone: (604) 415-6000
Direct: (604) 415-6022
FAX: (604) 415-6205
E-mail: richard_cam@pmc-sierra.bc.ca

31 Oct. 1996

Abstract

A simple control code suitable for PMC's Gigabit Ethernet UTP-5 PHY proposal is described in this report. It provides 16 8-quat code words, usable for embedded control sequences and run substitution, that meet IEEE 802 Hamming distance requirements.

I. Introduction

A 4-level code (2B1Q) was initially considered for PMC's UTP-5 PHY proposal. With this line coding scheme, it was not possible to generate control codes (for link management) uniquely distinguishable from the transmitted data. The limitations inherent with 2B1Q coding subsequently led to what is now referred as 4LZS (4-Level with Zero State) coding, in which a 5th level was added, corresponding to the zero amplitude level. In its initial conception, this zero state was reached by simply shutting off the transmitter, as in the case of 10Base-T Ethernet during intervals between packet transmission. The zero state would be used as an escape sequence that preceded control codes.

It was initially anticipated that this escape sequence would have to be long enough (to allow the line to settle down) for reliable detection. Subsequently, it became clear that the transmitter could drive the zero state (just as any other amplitude level) with minimal additional complexity, eliminating the need to budget for settling time. The characteristics of 5-level baseband signaling may not be fully exploited with 4LZS, since the zero state is used only for control code escape sequences. Further discussion on 5-level signaling is however beyond the scope of this report.

Since 4LZS coding is intended for (Gigabit) Ethernet, some due care must be taken to ensure that IEEE Project 802 functional requirements [1] are still met after introduction of the requisite control codes. One requirement which these 4LZS control codes have an impact on is that concerning Hamming Distance (Section 5.6.3 of [1]). This requirement is restated verbatim as follows: "A minimum of four bit cells in error shall be necessary for an undetected error to occur (Hamming distance 4)." There are other related requirements, such as 5.6.1 (MAC Frame Error Rate) and 5.6.2 (MAC Undetected Error Rate). These have more to do however with system design considerations beyond selection of control codes.

Data sequences that transform onto other data sequences do not pose any concern here as these errors will be detected by the CRC-32 FCS. Data sequences that transform to control codes however, can "restructure" an Ethernet frame in the case of end-of-packet or carrier extension. Control codes are not protected by the FCS; some error protection will be needed to detect errors that transform one control code to another. In some cases, erroneous control codes can be detected from the context in which they appear. For example, it does not make sense to have carrier extension prior to start-of-packet. In other cases however, control code errors may pass through undetected or not be detectable in a timely manner (e.g., carrier extension -> end-of-packet). Errors that transform a control code to a data sequence are also of concern in the case of carrier extension and end-of-packet. These are at least some scenarios in which the Hamming distance objective may be compromised.

II. Code Construction.

To address the Hamming Distance requirement, the following approach is recommended.

1. Ensure that any control code differs from any data sequence of equivalent length in at least three symbol positions. This guarantees that any combination of two symbol errors (spanning a total of four bit cells) will not cause a control codes or data sequences to map onto the other. Since data sequences use the entire 2B1Q symbol set and no other, any control code must therefore contain at least three zero-state symbols.

2. To provide for an additional margin of robustness, it is recommended that control codes contain at least four zero-state symbols so that they differ from data sequences in at least three symbol positions.

3. Suppose each control code is 16 bit cells (8 quats) long, with 4 quat positions set at the zero state. The remaining 8 bit cells can be encoded in such a way that an undetected error (mapping from control code to another) can only occur if at least 4 of those bit cells are in error. In this report, an (8, 4) extended Hamming code is used. This linear code is formed by adding an overall parity check bit to the (7, 4) Hamming code. Up to 16 control codes can be generated with this construction.

One "generator" matrix **G** for the (7, 4) Hamming code is given by

$$\mathbf{G} = \left[\mathbf{Q}^T \mathbf{I}_k\right] = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1}$$

Codewords $\mathbf{v} = (v_1, v_2, \ldots, v_n)$ are generated from source messages $\mathbf{u} = (u_1, u_2, \ldots, u_k)$ by applying $\mathbf{v} = \mathbf{u}\mathbf{G}$. To obtain an $(n+1, k)$ extended Hamming code, a parity-check bit v_0 is prepended to $\mathbf{v}$ to form the codeword $\mathbf{v}' = (v_0, v_1, v_2, \ldots, v_n)$. This check bit takes on the value of *1* if **v** has odd weight, and zero otherwise. The resulting codewords are as follows:

Table 1. Enumerated codewords for the (8, 4) extended Hamming code.

Number	$\mathbf{v}'$
0	0 0 0 0 0 0 0 0
1	1 1 0 1 0 0 0 1
2	0 1 1 1 0 0 1 0
3	1 0 1 0 0 0 1 1
4	1 0 1 1 0 1 0 0
5	0 1 1 0 0 1 0 1
6	1 1 0 0 0 1 1 0
7	0 0 0 1 0 1 1 1
8	1 1 1 0 1 0 0 0
9	0 0 1 1 1 0 0 1
10	1 0 0 1 1 0 1 0
11	1 0 0 0 1 0 1 1
12	0 1 0 1 1 1 0 0
13	1 0 0 0 1 1 0 1
14	1 0 0 0 1 1 1 0
15	1 1 1 1 1 1 1 1

The minimum distance of this code is 4 since it is a linear block code whose non-zero codewords have minimum weight 4.

As an aside, the corresponding parity-check matrix $\mathbf{H}$ for the (7, 4) Hamming code is given by

$$\mathbf{H} = \left[\mathbf{I}_{n-k}\mathbf{Q}\right] = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}, \qquad (2)$$

and the corresponding matrix $\mathbf{H}'$ for the (8, 4) extended Hamming code has the following form

$$\mathbf{H}' = \begin{pmatrix} 0 & & & \\ \vdots & & \mathbf{H} & \\ 0 & & & \\ 1 & 1 & \cdots & 1 \end{pmatrix}. \qquad (3)$$

A received sequence $\mathbf{w} = (w_0, w_1, \ldots, w_n)$ is in the extended code generated as described above if and only if the syndrome $\mathbf{w} \bullet \mathbf{H}'^T = \mathbf{0}$. For the application at hand however, computation of the syndrome is extraneous since the specific control codes will have to be decoded anyway and, moreover, the control codes may not occupy the entire code space.

4. By using Gray code -type mappings for the 2B1Q symbol space (as in BR-ISDN [2]) [3], any symbol error will tend to result in only one rather than two bit errors. For convenience, the ISDN 2B1Q bit mapping is shown in Table 2.

Table 2. Bit mapping for 2B1Q.

Bit Combination	Symbol
1 0	+3
1 1	+1
0 1	-1
0 0	-3

Table 3 lists the control codes currently deemed necessary and sufficient to obtain *functional* equivalence to the 8B10B-coded PCS control codes under consideration in IEEE 802.3z.

Table 3. PCS Link Control Codes.

Equivalent PCS Code	Function
F	Link Not Available
C	Link Configuration
I	Idle
P	Start of Packet
R	Carrier Extension
H	Invalid Code

In addition to these codes, additional control codes for run substitution [4] are defined below in Table 4.

Table 4. Run Substitution Codes.

Control Code	Function
S10	Substitute Run of +3's.
S11	Substitute Run of +1's.
S01	Substitute Run of -1's.
S00	Substitute Run of -3's.

Further discussion on the control codes in Tables 3 and 4 can be found in [5].
By mapping the control codes in Tables 3 and 4 to the first ten codewords in Table 1, and using the bit mapping given in Table 2, the following control codes are defined below in Table 5, in terms of 2B1Q symbols.

Table 5. PMC Gigabit Ethernet Control Codes.

Control Code	2B1Q Encoding
F	-3 -3 -3 -3
C	+1 -1 -3 -1
I	-1 +1 -3 +3
P	+3 +3 -3 +1
R	+3 +1 -1 -3
H	-1 +3 -1 -1
S10	+1 -3 -1 +3
S11	-3 -1 -1 +1
S01	+1 +3 +3 -3
S00	-3 +1 +3 -1

The codes in Table 5 are prefixed by a sequence of four consecutive zero-state quats. This prefix serves as an escape code to indicate that the next four quats to follow are to be interpreted as control codes. The zero-quat prefix is not repeated if a control code is repeated (as in the case of F, I, R and H) contiguously. For example, a sequence of idle codes is encoded as 0 0 0 0 -1 +1 -3 +3 -1 +1 -3 +3 ... Repeated codewords are scrambled to reduce EMC problems arising from introduced periodicities. Long symbol runs are precluded by scrambling and run substitution.

III. Concluding Remarks.

This report has discussed the construction of control codes that meet IEEE Hamming distance requirements. The overall approach has been to make all control codes sufficiently distinct from data sequences, and then to make the control codes sufficiently distinct from each other. This is accomplished in the first case by zero-state escape sequences, and in the second case by using an (8, 4) extended Hamming code.

IV. References.

[1] IEEE Project 802, "Functional Requirements, Version 6.10", 12 Nov. 1991, IEEE, Inc.

[2] Stallings, W., "ISDN and Broadband ISDN with Frame Relay and ATM, 3rd Ed.", Prentice-Hall, 1995.

[3] Cam, R., "RE> GEnet Status Review - 7/26", internal e-mail correspondence on 2B1Q code mapping rationale, 30 Jul. 1996.

[4] Cam, R., "Scrambling and 4LZS with Run Substitution", PMC-Sierra report, 21 Oct. 1996.

[5] Dabecki, S., et al, "PCS Mapping Using 4LZS Codes", PMC-Sierra report, Sept. 1996.