National Aeronautics and
Space Administration
Lyndon B. Johnson Space Center
2101 NASA Parkway
Houston, Texas 77058-3696

Reply to Attn of: EV7-08/06-067

August 29, 2006

TO: Memorandum of Record

FROM: EV7/Chief, Systems Analysis and Verification Branch

SUBJECT: General Solution for Theoretical Packet Data Loss Rate

Enclosed is the subject document for your information and retention. "Packet Data Loss" (PDL) is a performance measure for the "Frame Synchronization Strategy" (FSS) of a communication system in the presence of errors. The paper serves as a general reference for FSS performance prediction. Some of the methods represent original work that can also be used to predict the performance of error-correction block codes. This is expansion of a paper distributed in 1996.

For additional information on this analysis and test result, please contact Mr. Chatwin Lansdowne at (281) 483-1265

Catherine C. Sham

Enclosure

cc:
EV/L. K. Bromley
EV2/S. Schadelbauer
EV7/H. Chen
EV7/R. S. Davis
EV7/L. E. Hood
EV7/C. T. Jih
EV7/J. R. Keiser
EV7/Q. Kroll
EV7/D. D. Lee
EV7/A. R. Olstad
EV7/R. L. Robinson
EV7/C. C. Sham
EV7/J. A. Soloff
EV7/A. C. Van Baalen
EV7/L. Vazquez
EV7/S. V. Vitalpur
EV7/C. A. Lansdowne
EV7/M. S. Marston
EV7/S. S. Marston
EV7/N. J. Robinson
EV7/C. C. Sham
EV7/B. G. Smith
EV7/HON/P. R. Flores
EV7/HON/J. A. Ross
EV813/ESCG/D. N. Cravey
EV813/ESCG/F. T. Shetz
EV813/ESCG/B. S. Collier
EV813/ESCG/C. M. Delgado
EV813/ESCG/T. B. Jenkins
EV813/ESCG/J. L. Fox
EV813/ESCG/R. E. Canup
EV813/ESCG/D. Stewart-Knight

EV7/CALansdowne:ksg:06/06/06:31265

NASA
National Aeronautics and
Space Administration

EV7-06-600

Lyndon B. Johnson Space Center
Houston, TX 77058

ELECTRONIC SYSTEMS TEST LABORATORY

GENERAL SOLUTION FOR THEORETICAL PACKET DATA LOSS RATE

EV/AVIONIC SYSTEMS DIVISION
EV7/SYSTEMS ANALYSIS AND TEST BRANCH
JULY 2006

GENERAL SOLUTION FOR THEORETICAL PACKET DATA LOSS RATE

Prepared by:

Chatwin Lansdowne

Reviewed by:

Ned J. Robinson, ESTL Facility Manager

Billy G. Smith, Test Manager

Approved by:

Catherine C. Sham, Chief
Avionics Test and Analysis Branch

Original, EV4-96-601
DECEMBER 1996

ACRONYMS AND ABBREVIATIONS

BER	Bit Error Rate
Bi-Phase-L	Manchester encoded, Level
Bi-Phase-M	Manchester encoded, Mark
Bi-Phase-S	Manchester encoded, Space
CADU	Channel Access Data Unit
CCSDS	Consultative Committee on Space Data Systems
ESTL	Electronic Systems Test Laboratory
FDP	Functionally Distributed Processor
FEPS	Front End Processor System
F/S	Frame Synchronizer
FSP	Frame Synchronization Pattern
FSS	Frame Synchronizer Strategy
FSW	Frame Synchronization Word (same as FSP)
kpbs	kilo-bits per second
NASA	National Aeronautics and Space Administration
NRZ-L	Non-Return to Zero - Level
NRZ-M	Non-Return to Zero - Mark
NRZ-S	Non-Return to Zero – Space
OCA	Orbiter Communications Adapter
PDL	Packet Data Loss rate
RF	Radio Frequency
RLL	Run-Length Limiting
R-S	Reed Solomon Coding
SGS	Space to Ground System

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GENERAL SOLUTION FOR THEORETICAL
PACKET DATA LOSS RATE

1. Introduction

Communications systems which transfer blocks ("frames") of data must use a marker ("frame synchronization pattern") for identifying where a block begins. A technique ("frame synchronization strategy") is used to locate the start of each frame and maintain synchronization as additional blocks are processed. A device which strips out the frame synchronization pattern [FSP] and provides an "end of frame" pulse is called a frame synchronizer. As clock and data errors are introduced into the system, the start-of-block marker becomes displaced and/or corrupted. The capability of the frame synchronizer to stay locked to the pattern under these conditions is a figure of merit for the frame synchronization strategy. It is important to select a strategy which will stay locked nearly all the time at bit error rates where the data is usable. ("Bit error rate" [BER] is the fraction of binary bits which are inverted by passage through a communication system.) The fraction of frames that are discarded because the frame synchronizer is not locked is called "Percent Data Loss" or "Packet Data Loss rate" [PDL].

A general approach for accurately predicting PDL given BER was developed in *Theoretical Percent Data Loss Calculation and Measurement Accuracy*, T. P. Kelly, LESC-30554, December 1992. Kelly gave a solution in terms of matrix equations, and only addressed "level" channel encoding. This paper goes on to give a closed-form polynomial solution for the most common class of frame synchronizer strategies, and will also address "mark" and "space" (differential) channel encoding, and burst error environments. The paper is divided into four sections and follows a logically ordered presentation, with results developed before they are evaluated. However, most readers will derive the greatest benefit from this paper by treating the results as reference material. The result developed for differential encoding can be extended to other applications (like block codes) where the probability is needed that a block contains only a certain number of errors.

2. Important Parameters of a Frame Sync Strategy

The variables on which packet data loss depends are as follows:

F_i is the number of consecutive good frame sync patterns required to go into the "lock" state; this counts "search" and "check" or "verify" states in which no data is transferred.

F_o is the number of consecutive bad frame sync patterns required to go into the "search" state; this includes "lock" and "verify" or "flywheel" states in which data is transferred.

P_{FSS} is the (uniform) probability of a frame synchronizer pattern being identified as "GOOD" when in a "search" or "check" state and data is not transferred.

P_{eFSL} is the (uniform) probability of a frame synchronizer pattern being declared "BAD" when in a "lock" or "verify" or "flywheel" state in which data was being transferred.

Together, these variables are sufficient to define a state diagram, and the probability of transitions between states, as shown in Figure 2-1.

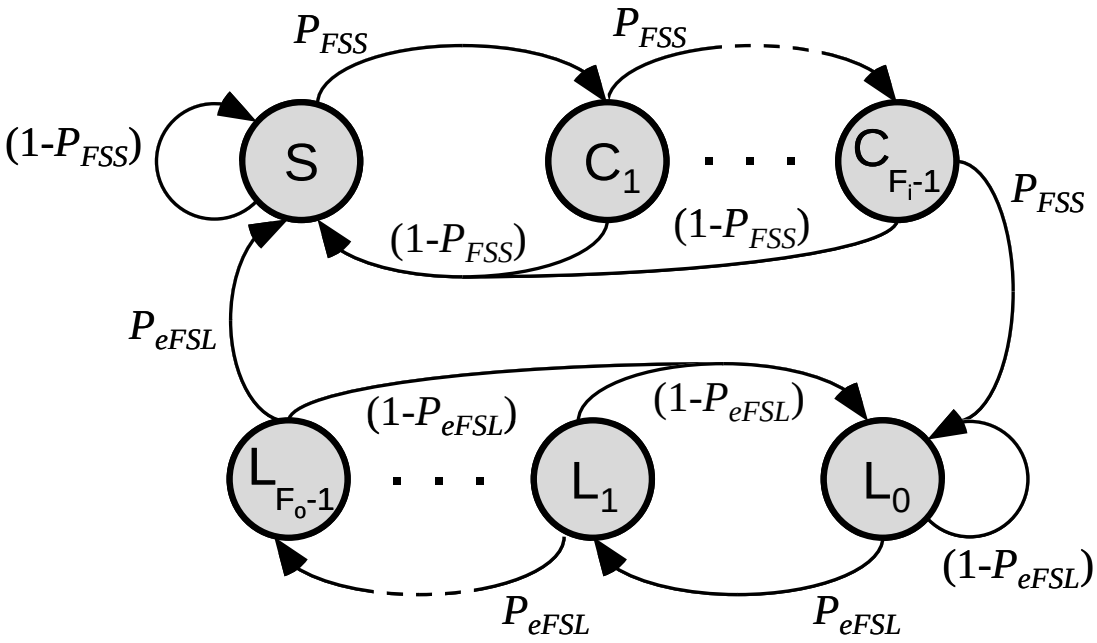

Figure 2-1. State Diagram for Generic Frame Sync Strategy

The variables P_{FSS} and P_{eFSL} are intermediate variables that depend on several additional parameters:

N is the number of bits in a frame synchronization pattern.

P_e is the (uniform) probability of a bit being in error. A non-uniform case will also be discussed.

P_{bs} is the (uniform) probability of a bit slip.

BS is the number of bit slips allowed in either direction for the frame sync pattern; the validation window size is twice this, plus one (for no slips).

FL is the number of bits in a frame (the "frame length").

E_i is the number of bit errors allowed in a frame sync pattern when in lock.

E_o is the number of bit errors allowed in a frame sync pattern when out of lock.

3. Analysis

Percent Data Loss performance must be broken into two calculations: First, what is the PDL in terms of the intermediate variables P_{FSS} and P_{eFSL}, and second, how do P_{FSS} and P_{eFSL} depend on the bit error rate, which is related to RF power in a system-dependent way.

3.1. General Solution Given Probability of State Transitions

The percentage of frames that will be discarded under the generic frame synchronization strategy is the same as the percentage of time spent in the Search and

Check states. The state diagram in Figure 2-1 can be used to construct a system of equations as follows:

1) $$P_S = P_S(1-P_{FSS}) + P_{C1}(1-P_{FSS}) + \cdots + P_{C(Fi-1)}(1-P_{FSS}) + P_{L(Fo-1)}P_{eFSL}$$

2) $$P_{C1} = P_S P_{FSS}$$

$$\vdots$$

$$P_{C(Fi-1)} = P_{C(Fi-2)}P_{FSS}$$

3) $$P_{L0} = P_{C(Fi-1)}P_{FSS} + P_{L0}(1-P_{eFSL}) + P_{L1}(1-P_{eFSL}) + \cdots + P_{L(Fo-1)}(1-P_{eFSL})$$

4) $$P_{L1} = P_{L0}P_{eFSL}$$

$$\vdots$$

$$P_{L(Fo-1)} = P_{L(Fo-2)}P_{eFSL}$$

5) $$1 = P_S + P_{C1} + \cdots + P_{C(Fi-1)} + P_{L0} + P_{L1} + \cdots + P_{L(Fo-1)}$$

6) $$PDL = P_S + P_{C1} + \cdots + P_{C(Fi-1)}$$

where

P_S is the probability of the frame synchronizer being in the search state

P_{C1} is the probability of the frame synchronizer being in the first check state

$P_{C(Fi-1)}$ is the probability of the frame synchronizer being in the last check state

P_{L0} is the probability of the frame synchronizer being in the primary lock state

P_{L1} is the probability of the frame synchronizer being in the second lock state

$P_{L(Fo-1)}$ is the probability of the frame synchronizer being in the last lock state

PDL is the fraction of data lost

Essentially, these equations say that the probability of being in a state, is the probability of being in each state, times the probability of making the transition from that state to the new state. These equations may be combined and simplified as follows:

1)&6) $$P_S = PDL \cdot (1-P_{FSS}) + P_{LFo-1}P_{eFSL}$$

2) $$P_{CFi-1} = P_S P_{FSS}^{Fi-1}$$

5)&6) $$(P_{L0} + P_{L1} + \cdots + P_{LFo-1}) = 1 - PDL$$

3)&5)&6) $$P_{L0} = P_S P_{FSS}^{Fi} + (1-PDL)(1-P_{eFSL})$$

6)&2) $$PDL = P_S(1 + P_{FSS} + P_{FSS}^2 + \cdots + P_{FSS}^{Fi-1})$$

4) $$P_{LFo-1} = P_{L0}P_{eFSL}^{Fo-1}$$

These equations can be algebraically combined with the formula for the sum of a finite geometric series,

$$\sum_{i=1}^{N} ax^{i-1} = a\frac{1-x^N}{1-x}$$

to reach a solution for the Probability of Data Loss which can be expressed as,

$$\boxed{PDL = \frac{1}{1 + \dfrac{P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL}^{Fo})}{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS}^{Fi})}}}$$

This equation presumes that data is discarded in the Search and Check states, and transferred in the Lock and Verify states. Some modern frame synchronizers allow data transfer states to be designated by the user; for these it is necessary to solve for the probability of being in any specific state. First, define the probabilities of being in the Search, Check, Lock, and Verify states respectively:

7) $P_S = P_S$

8) $P_C = P_{C1} + \cdots + P_{C(Fi-1)}$

9) $P_L = P_{L0}$

10) $P_V = P_{L1} + \cdots + P_{L(Fo-1)}$

These definitions can be combined with the equations 1) through 5) above to reach a solution for the probability of being in any given state. Then PDL is the sum of the probabilities of being in the states in which data is not transferred.

$$P_S = \frac{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS})}{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS}^{Fi}) + P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL}^{Fo})} = \frac{1}{\frac{P_{FSS}^{Fi}(1-P_{eFSL}^{Fo})}{P_{eFSL}^{Fo}(1-P_{eFSL})} + \frac{(1-P_{FSS}^{Fi})}{(1-P_{FSS})}}$$

$$P_C = \frac{P_{eFSL}^{Fo}(1-P_{eFSL})(P_{FSS}-P_{FSS}^{Fi})}{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS}^{Fi}) + P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL}^{Fo})} = \frac{1}{\frac{P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL}^{Fo})}{P_{eFSL}^{Fo}(1-P_{eFSL})(P_{FSS}-P_{FSS}^{Fi})} + \frac{(1-P_{FSS}^{Fi})}{(P_{FSS}-P_{FSS}^{Fi})}}$$

$$P_L = \frac{P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL})}{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS}^{Fi}) + P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL}^{Fo})} = \frac{1}{\frac{(1-P_{eFSL}^{Fo})}{(1-P_{eFSL})} + \frac{P_{eFSL}^{Fo}(1-P_{FSS}^{Fi})}{P_{FSS}^{Fi}(1-P_{FSS})}}$$

$$P_V = \frac{P_{FSS}^{Fi}(1-P_{FSS})(P_{eFSL}-P_{eFSL}^{Fo})}{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS}^{Fi}) + P_{FSS}^{Fi}(1-P_{FSS})(1-P_{eFSL}^{Fo})} = \frac{1}{\frac{(1-P_{eFSL}^{Fo})}{(P_{eFSL}-P_{eFSL}^{Fo})} + \frac{P_{eFSL}^{Fo}(1-P_{eFSL})(1-P_{FSS}^{Fi})}{P_{FSS}^{Fi}(1-P_{FSS})(P_{eFSL}-P_{eFSL}^{Fo})}}$$

3.2. Probability of State Transitions

What must now be addressed is, what is the probability of a frame sync pattern being declared good or bad? P_{FSS} and P_{eFSL} depend on how errors are distributed. Solutions for P_{eFSL} generically give rates at which blocks of N bits contain more than E_i errors. Caution: frame sync patterns as *separated* blocks are independent of each other, but for differential coding *consecutive* blocks would have overlap. Thus, P_{eFSL} gives an exact proportion of blocks over the error threshold, but for consecutive differentially coded blocks it is a simplification to treat the proportion is as uniform "rate."

3.2.1. "Level"-Encoding with Stable Clock

The simplest systems to compute performance for, are those in which the probability that a bit is in error is uniformly random, represented by a constant "bit error

rate," P_e. Said another way, bit errors are independent of one another. For this case, the probability that a frame sync pattern is accepted when in the search states is,

$$P_{FSS} = \sum_{i=0}^{E_o} \frac{N!}{(N-i)!i!} P_e^i (1-P_e)^{N-i}$$

and the probability that a frame sync pattern is rejected when in the lock states is,

$$P_{eFSL} = 1 - \sum_{i=0}^{E_i} \frac{N!}{(N-i)!i!} P_e^i (1-P_e)^{N-i}$$

where

P_{FSS} is the probability of a "good" frame sync pattern when in the search states

P_{eFSL} is the probability of a "bad" frame sync pattern when in the lock states

P_e is the probability that a received bit is in error

N is the size (in bits) of a frame synchronization pattern

E_i is the number of bit errors allowed in a frame sync pattern when in lock

E_o is the number of bit errors allowed in a frame sync pattern when out of lock

These equations employ the binomial probability formula under a sum, so that the likelihood of a frame sync pattern being accepted when in the search state, is the likelihood of having zero, or one, or… errors (up to the allowable number) out of the number of bits in a frame sync pattern.

3.2.2. "Mark"- or "Space"- Encoding with Stable Clock

Under any of the "mark" or "space" ("differential") channel encoding schemes, errors are known to be connected. The following variables will be used here:

P_{eL} is the probability of a level error

P_e is the probability of a bit error

Errors in logic level occur because of noise and are relatively statistically independent of one another. Data, however, is coded, for example, in an NRZ-M format. The nature of NRZ-M is such that a level error causes not one but two bit errors. In fact, it is apparent by extension of Figure 3-1 that any number of consecutive level errors will cause exactly two bit errors. It is also important that the interpretation of some number of bits depends on one level more than the number of bits to be interpreted. The strategy, then, will be to determine both the bit error rate which would be observed at the NRZ-M decoder, and the percentage of data loss, in terms of the rate of statistically independent level errors.

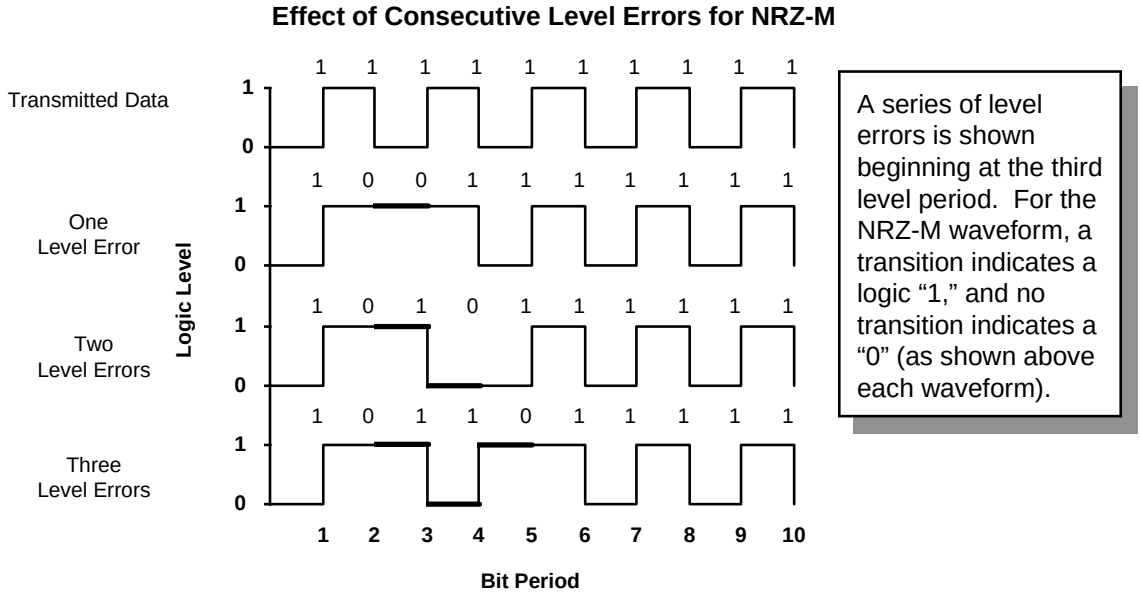

Figure 3-1. Propagation of Errors Using NRZ-M Coding

For randomly distributed level errors, the probability of a bit error is:

$$P_e = 2(1-P_{eL})P_{eL}(1-P_{eL}) + 2(1-P_{eL})P_{eL}^2(1-P_{eL}) + 2(1-P_{eL})P_{eL}^3(1-P_{eL}) + \ldots$$

$$= 2(1-P_{eL})^2 \frac{P_{eL}}{1-P_{eL}}$$

So,

$$\boxed{P_e = 2P_{eL}(1-P_{eL})}$$

This equation gives the bit error rate that would be observed at the NRZ-M decoder, although the errors were not uniformly distributed. Here, the probabilities of a correct bit then one error and then one correct bit, one correct and two errors and one correct, one correct and three errors and one correct, *etc*. have been summed to the improbable infinite limit using the geometric series formula,

$$\sum_{i=1}^{\infty} ar^i = \frac{ar}{1-r}, \; 0 \le r < 1$$

In each of these events, two (2) bit errors occur. Observe that the bit error rate is 50% when the level error rate is 50%, is twice the level error rate for small level error rates, and is 0% when the level error rate is 0% or 100%. An important property of NRZ-M is that the data can be recovered correctly whether the levels are all correct or all inverted. Conversely, using the quadratic equation,

$$\boxed{P_{eL} = \frac{1 \mp \sqrt{1-2P_e}}{2}}$$

These relationships are plotted in Figure 3-2 and Figure 3-3.

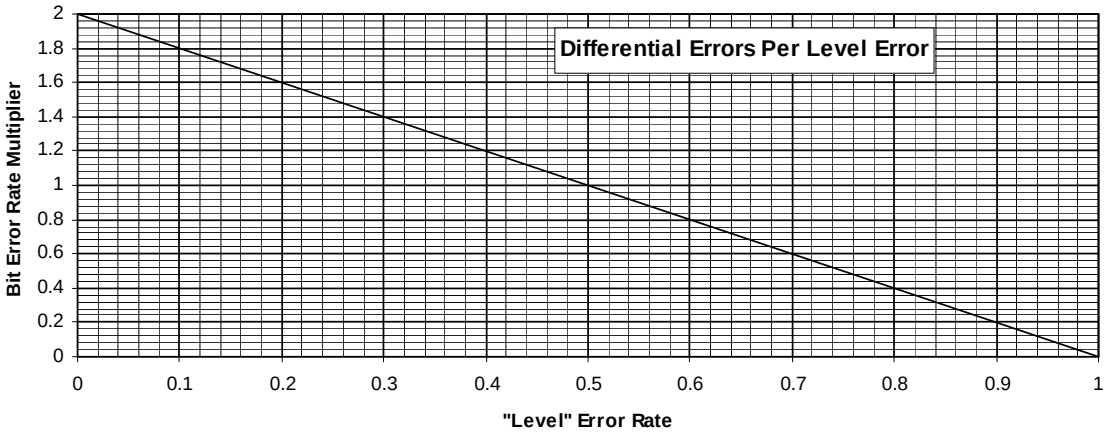

Figure 3-2. Average Number of Differential Errors with "Level" Error Probability

Figure 3-2 shows that for parts-per-million error rates, a level error causes essentially two bit errors. As the level error rate approaches 50% (no information recoverable), so does the bit error rate. And for a 100% level error rate (inverted levels), the bit error rate after differential decoding is zero. Figure 3-3 illustrates the 0.3dB loss of performance (at 10^{-5} BER) that differential encoding sacrifices in order to achieve polarity insensitivity. Figure 3-3 is based on a theoretical curve that applies to several modulation techniques.

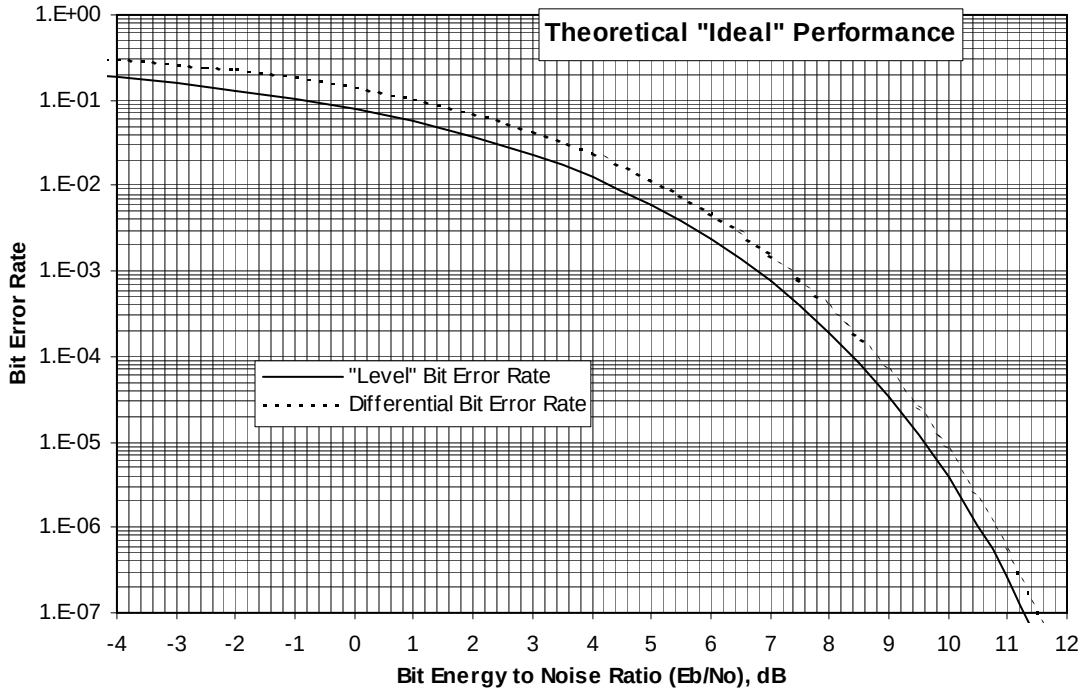

Figure 3-3. Theoretical Shape of Differential Bit Error Rate Curve

Calculating the probabilities for state transitions (the probabilities that the number of errors in the frame synchronization word will exceed the thresholds) is complex because errors are not independent, and so the relative locations of level errors can be more important than their quantity. Recall that the interpretation of the bits in a frame

sync pattern depends on one level more than the number of bits in the pattern. This is all an accounting problem, and it can be broken down as follows:

Table 1. Accounting for NRZ-M, Error-Free Case

	Levels											Bits									
Level or bit position:	1	2	3	4	5	6	7	. . .	N-1	N	N+1	1	2	3	4	5	6	7	. . .	N-1	N
All the ways to	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
have no bit errors:	e	e	e	e	e	e	e	e	e	e	e	x	x	x	x	x	x	x	x	x	x

In Table 1, an “x” indicates a level or bit was received correctly, and an “e” indicates a level or bit was received in error. There are only two ways that N bits can be error-free: either all the levels are error-free, or all the levels are in error. Now, what if one bit error is allowed?

Table 2. Accounting for NRZ-M, Single-Error Case

	Levels											Bits									
Level or bit position:	1	2	3	4	5	6	7	. . .	N-1	N	N+1	1	2	3	4	5	6	7	. . .	N-1	N
The ways to have	e	x	x	x	x	x	x	x	x	x	x	e	x	x	x	x	x	x	x	x	x
one bit error:	e	e	x	x	x	x	x	x	x	x	x	x	e	x	x	x	x	x	x	x	x
							- - -										- - -				
	e	e	e	e	e	e	e	e	e	e	x	x	x	x	x	x	x	x	x	x	e
	x	x	x	x	x	x	x	x	x	x	e	x	x	x	x	x	x	x	x	x	e
	x	x	x	x	x	x	x	x	x	e	e	x	x	x	x	x	x	x	x	e	x
							- - -										- - -				
	x	e	e	e	e	e	e	e	e	e	e	e	x	x	x	x	x	x	x	x	x

The accounting is more complex. At least one level at one end must be in error, and there must be at least one correct level at the other end. For any number of level errors, there are two ways they could cause one bit error (be clustered at one end, or the other). The equation which represents this is given in Table 4.

Table 3. Accounting for NRZ-M, Double-Error Case

	Levels											Bits									
Level or bit position:	1	2	3	4	5	6	7	. . .	N-1	N	N+1	1	2	3	4	5	6	7	. . .	N-1	N
The ways to have	e	x	x	x	x	x	x	x	x	x	e	e	x	x	x	x	x	x	x	x	e
two bit errors	e	e	x	x	x	x	x	x	x	x	e	x	e	x	x	x	x	x	x	x	e
with fringe	e	x	x	x	x	x	x	x	x	e	e	e	x	x	x	x	x	x	x	e	x
level errors:							- - -										- - -				
	e	e	e	e	e	e	e	e	e	x	e	x	x	x	x	x	x	x	x	e	e
	e	e	e	e	e	e	e	e	x	e	e	x	x	x	x	x	x	x	e	e	x
							- - -										- - -				
	e	x	e	e	e	e	e	e	e	e	e	e	e	x	x	x	x	x	x	x	x
The ways to have	x	e	e	e	e	e	e	e	e	e	x	e	x	x	x	x	x	x	x	x	e
two bit errors	x	x	e	e	e	e	e	e	e	e	x	x	e	x	x	x	x	x	x	x	e
without fringe	x	e	e	e	e	e	e	e	e	x	x	e	x	x	x	x	x	x	x	e	x
level errors:							- - -										- - -				
	x	x	x	x	x	x	x	x	x	e	x	x	x	x	x	x	x	x	x	e	e
	x	x	x	x	x	x	x	x	e	x	x	x	x	x	x	x	x	x	e	e	x
							- - -										- - -				
	x	e	x	x	x	x	x	x	x	x	x	e	e	x	x	x	x	x	x	x	x

Now suppose two bit errors are allowed in the frame sync pattern. To have an even number of errors, the first and last level must both be correct or in error, alike. For

simplicity, make two equations: there is one way for two level errors to cause two bit errors, two ways for three level errors to cause two bit errors, *etc*., and conversely. These equations are given in Table 4.

Table 4. Components of Frame Sync Pattern Error Equation

1	All frame sync words (FSW) are erroneous ...except those that are not:
$-P_{eL}^{N+1}-(1-P_{eL})^{N+1}$	except those which are entirely composed of level errors, or are error free (no bit errors either way);
$-2\sum_{i=1}^{N}P_{eL}^{i}(1-P_{eL})^{N+1-i}$	(if one error is permitted) except those which contain exactly one error: contiguous level errors must start from either the first or last level with the rest correct, or conversely;
$-\sum_{i=2}^{N}(i-1)\left(P_{eL}^{i}(1-P_{eL})^{N+1-i}+P_{eL}^{N+1-i}(1-P_{eL})^{i}\right)$	(if two errors are permitted) except those which contain exactly two bit errors: the accounting here is more complex, refer to the text;
$-2\sum_{i=\frac{j+1}{2}}^{N-\frac{j-1}{2}}C\left(i-1,\frac{j-1}{2}\right)\cdot C\left(N-i,\frac{j-1}{2}\right)\cdot\left(P_{eL}^{i}(1-P_{eL})^{N+1-i}\right)$	(if j errors are permitted, odd j) except those which contain exactly j bit errors (refer to the text);
$-\sum_{i=\frac{j}{2}+1}^{N+1-\frac{j}{2}}C\left(i-1,\frac{j}{2}\right)\cdot C\left(N-i,\frac{j}{2}-1\right)\cdot\left(P_{eL}^{i}(1-P_{eL})^{N+1-i}+P_{eL}^{N+1-i}(1-P_{eL})^{i}\right)$ or $-\sum_{i=\frac{j}{2}+1}^{N+1-\frac{j}{2}}C\left(i-1,\frac{j}{2}\right)\cdot C\left(N-i,\frac{j}{2}-1\right)\cdot\left(P_{eL}^{i}(1-P_{eL})^{N+1-i}\right)$ $-\sum_{i=\frac{j}{2}}^{N-\frac{j}{2}}C\left(i-1,\frac{j}{2}-1\right)\cdot C\left(N-i,\frac{j}{2}\right)\cdot\left(P_{eL}^{i}(1-P_{eL})^{N+1-i}\right)$	(if j errors are permitted, even j) except those which contain exactly j bit errors (refer to the text);

The entire process can be generalized for the case where j errors occur. There are two categories of problems: j is even, or j is odd. In either case, the key to the whole problem is this: there are required to be for any case a certain number of "error" clusters, distinguished by some other number of "correct" clusters. Each cluster must contain at least one member level, and any "error" or "correct" levels have to fall into one

of the appropriate clusters. In general, h interchangeable objects can be grouped into k groups in $C(h\text{-}1,k\text{-}1)$ ways. $C(N,n)$ is the combinations function, which gives the number of combinations of n objects that can be chosen from a population N.

$$C(N,n) = \frac{N!}{n!(N-n)!}$$

For the case that j is odd, there are i levels in error, to be divided among $(j+1)/2$ "error" clusters— and the remaining $(N+1\text{-}i)$ levels can be independently divided among the $(j+1)/2$ "correct" clusters. For the case that j is even, there are i levels in error, to be divided among $j/2+1$ "error" clusters; and the remaining $(N+1\text{-}i)$ levels can be independently divided among the $j/2$ "correct" clusters, and conversely.

All the elements of an expression for P_{eFSL} are given in Table 4. This equation is symmetrical about the level error rate of 50%. A formula for P_{FSS} can be developed in the same way (for the same number of errors allowed, one is just the complement of the other).

The final result is clearly a sum of these sums. There is some opportunity for simplification, because the sums contain a common part (representing the likelihood of a particular number of level errors and correct levels) and a coefficient representing a count of how many ways this combination of levels could cause a certain number of bit errors to occur. Clearly, the counts for all the valid numbers of bit errors can be added together before multiplying with the probabilities of level error counts. Because of the odd-even differences there is no clean way to write a double-sum formula. Instead, an efficient algorithm is suggested, with pseudo-code below:

```
PSUM=0                    // PSUM accumulates the main sum
x=PeL*(1-PeL)^N           // x starts with 1 error, N correct
y=PeL/(1-PeL)             // factor to advance x
for i=1 to N              // all the combinations
   P=0                    // P is sum of coefficients
   for j=1 to Ei          // for all the coefficients
      if even(j) then     // if j is even . . .
         if i>=j/2+1 and i<=N+1-j/2 then
            P=P+C(i-1,j/2)*C(N-i,j/2-1)
         end if
         if i>=j/2 and i<=N-j/2 then
            P=P+C(i-1,j/2-1)*C(N-i,j/2)
         end if
      else                // if j is odd . . .
         if I>=(j+1)/2 and I<=N-(j-1)/2 then
            P=P+2*C(i-1,(j-1)/2)*C(N-i,(j-1)/2)
         end if
      end if
   next j
   PSUM=PSUM+P*x          // add to main summation
   x=x*y                  // one more error, one less correct
next i
PeFSL=1-PeL^(N+1)-(1-PeL)^(N+1)-PSUM
```

Efficiency can be increased at the expense of readability by expanding the calls to the combinations function into lines that reuse the product (factorials, like integer exponents, generate the next result by a product of a new term with the last result). This final result is not so simple or elegant as one might wish for, but it is practical nowadays to embed this solution in a spreadsheet software package.

3.2.3. Unstable Clock Recovery

In general, a communication system should be designed so that the bit synchronizer can remain locked at usable error rates. If bit slips are occurring by design at usable error rates where the frame synchronizer should be passing a significant percentage of data, then losses caused by bit slips must be accounted for in the theoretical performance calculation.

One needs to know the frequency and uniformity of bit slips. For uniformly random, statistically independent bit slips,

$$P_{FSS} = \left(\sum_{i=0}^{BS} \frac{FL!}{(FL-i)!i!} P_{bs}^{i} (1-P_{bs})^{FL-i} \right) P_{FS}$$

and similarly,

$$P_{eFSL} = 1 - \left(\sum_{i=0}^{BS} \frac{FL!}{(FL-i)!i!} P_{bs}^{i} (1-P_{bs})^{FL-i} \right) P_{FS}$$

where

P_{bs} is the (uniform) probability of a bit slip.

FL is the number of bits in a frame (the "frame length").

BS is the number of bit slips allowed for the frame sync pattern (the validation window size, less one for no bit slips, divided by two for symmetry).

P_{FS} is the probability that the frame sync pattern would be declared "good" based on any other parameters.

Simply put, these equations say that the probability of a frame sync pattern being good, is the probability that the number of bit slips over the span of a frame is within the allowable tolerance, and any additional conditions are met.

To maintain a two-dimensional relationship, P_{bs} needs to be correlated to P_e (which is a system-dependent relationship). Both depend on E_b/N_o: probability of bit slips is a function of signal to noise ratio and parameters of the phase locked loop in the bit synchronizer.

3.2.4. Coded Systems

Error corrected systems cause errors to cluster. There are two kinds of error correction, or "coding:" block codes and convolutional codes. Block codes are applied after frame synchronization and have no effect on packet data loss as defined in this paper. However, the relationships derived in this section for finding the probability that a "block" of N bits (which we have called a frame sync pattern) contains E or fewer errors can be extended to determine when a block contains a correctable number of errors.

Errors can occur in bursts for other reasons also. Decryption causes errors to happen in bursts. Pulsed noise can cause errors to happen in groups. The analysis in this section can be applied to these systems also, so long as burst lengths are shorter than frame length, are about the same size, and are asynchronous with the frames.

Convolutional codes correct errors before frame synchronization. To determine the effect of coding on packet data loss, one needs to know the typical size of a cluster error (m bits). This could be found by analysis of the coding scheme being used, but might best be found by experiment. Bear in mind that the number of errors observed in a burst is actually spread out over about twice that many bits (m) because the bit error rate during the burst is about 50% (100% would just mean that all the bits were inverted—not the case).

The problem will be broken into two parts. First, what is the probability that any given bit position will be the start of a burst error (P_{burst}, assumed to be a constant), in terms of the decoded bit error rate; and second, how many bit positions must not be the start of a burst error in order for a frame sync pattern to be recognized. It is assumed here that the burst size is less than a frame length; otherwise, any burst would certainly damage a sync pattern.

For the first part, think of a decoder error burst, plus some number of entirely corrected bits following, such that the average number of errors over the average number of bits represents the observed bit error rate.

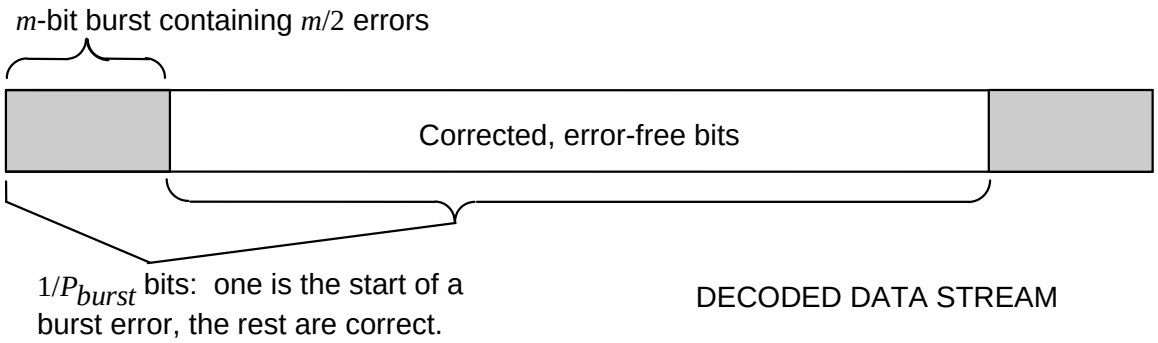

Figure 3-4. Average Burst Error Scenario

The bit error rate (P_e) that is observed here is $m/2$ bit errors out of $(1/P_{burst}+m-1)$ bits. This is saying that a burst cannot begin during a burst in progress, and represents the decoder recognizing that it has lost lock and emptying its buffer. This equation is easily solved to find

$$P_{burst} = \frac{1}{\frac{m}{2P_e} - (m-1)}$$

Next, find the probability-weighted number of bit positions for which, if a block of errors began there, a frame sync would be lost; call this number k. As can be seen from Figure 3-5, if there is not at least $E+1$ bits of overlap (E is the number of bit errors permitted in the frame sync pattern), then there is no chance (probability weighting of zero) of the frame sync pattern being in error. There is half a chance at $2E+1$ bits overlap (because of the 50% bit error rate for the duration of the burst), so this location only contributes half a bit position. As the burst is (pardon the expression) "convolved" with each of the positions that could potentially cause a frame sync pattern error, it becomes essentially certain that such an error will occur (and the bit positions are added at full weight).

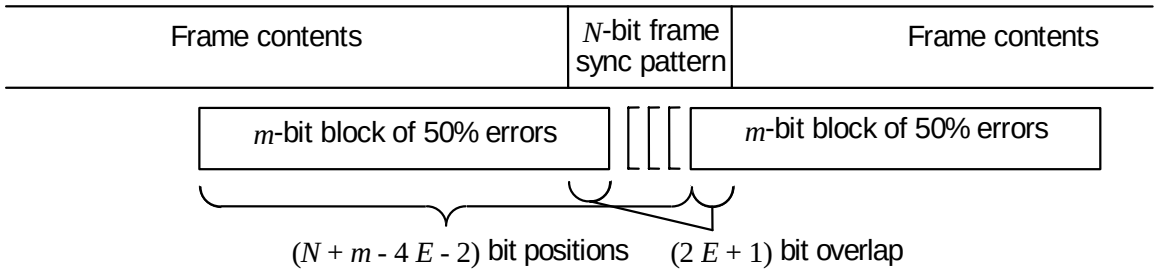

Figure 3-5. Bit Positions for Burst Error Damage to Frame Sync Pattern

The “true” result of this convolution would be,

$$k = 2\sum_{i=E+1}^{\min(N,m)-1}\left(1-\sum_{j=0}^{E}C(i,j)\left(\frac{1}{2}\right)^{i}\right)+\left(1+\left|N-m\right|\right)\left(1-\sum_{j=0}^{E}C(\min(N,m),j)\left(\frac{1}{2}\right)^{\min(N,m)}\right)$$

effective bit positions from which the initiation of a block error would cause a frame sync pattern error. This equation says that, for each position from which a pattern error could occur at all, add up the probability that such an error happens (the complement of the probability that E or fewer errors occur, by the binomial equation). A very good approximation is,

$$k \approx N + m - 4E - 3 \qquad m >> E, N >> E$$

Then finally, the probability that a frame sync pattern is not hit, is the probability that a block error does *not* occur in any of *k* positions. The probability that a frame sync pattern is accepted when in the search states is,

$$P_{FSS} = \left(1-\frac{1}{\frac{m}{2P_e}-(m-1)}\right)^{N+m-4E_o-3}$$

and the probability that a frame sync pattern is rejected when in the lock states is,

$$P_{eFSL} = 1-\left(1-\frac{1}{\frac{m}{2P_e}-(m-1)}\right)^{N+m-4E_i-3}$$

where

P_{FSS} is the probability of a “good” frame sync pattern when in the search states

P_{eFSL} is the probability of a “bad” frame sync pattern when in the lock states

P_e is the probability that a received bit is in error

N is the number of bits in a frame synchronization pattern

m is the number of bits in a cluster error (twice the number of errors)

E_i is the number of bit errors allowed in a frame sync pattern when in lock

E_o is the number of bit errors allowed in a frame sync pattern when out of lock

Other effects happen beyond a 10% to 20% bit error rate, and this model is pessimistic.

4. Applications

This section will bridge the gap between the variables called out in the theoretical solutions, and the controls which are available on some frame synchronizers used in the manned space program. Finally, possible failings of the theoretical predictions will be discussed.

4.1. Channels

Channel characteristics for the Space Shuttle and Space Station programs are given in Table 5.

Table 5. Some Channel Characteristics in the Manned Space Program

Channel Description	Source Encoding	N	FL	Strategy
Shuttle S-band Return Link	Bi-Phase-L 3x Convolutional	24 bits FA F3 20	1920 960	403(1,2,3,3) FEPS(1,2,4,3)
Shuttle S-band Forward Link	Bi-Phase-L 3x Convolutional	24 bits FA F3 20	1440 640	NSP same as 403(2,2,2,0)
Station ACS Link (Forward and Return)	NRZ-M Block (R-S)	32 bits 1A CF FC 1D	2048	403 N/A FEPS(2,3,3,0)
Station SGS Link (Return)	NRZ-M Block (R-S)	32 bits 1A CF FC 1D	10112	403 N/A FEPS(2,3,3,0) FDP (0,2,5,3,0,3)

4.2. Frame Synchronizers

Two types of frame synchronizers have been used in the Space Shuttle program, and two for Space Station. The parameters which make up their strategies are defined below, along with the range of allowable values (all given in states, or bits).

Table 6. Frame Synchronizer Characteristics

	FEPS		FDP		Aydin 403	
Variable	Value	Range	Value	Range	Value	Range
F_i	N2	1-16	N2-1	0-15	2	*Fixed*
F_o	N3	1-16	N3-1	0-15	L PATT (Fl)+1	0-7
BS	N4	0-3	$N4_C$, $N4_{LF}$	0-4	BS	0-3?
E_o	N1	0-15	$N1_{SC}$	0-31	S ERR (Es)	0-7
E_i	N1	0-15	$N1_{LF}$	0-31	L ERR (El)	0-7
N		4-32		0-64		7-32
FL		1048576		524288		15888

The Frame Sync Strategy for the Aydin 403 is often described with a four-parameter shorthand, "(Es, El, Fl, BS)," where the variables signify

- Es — Search Allowable Errors: Maximum number of bit errors allowed in a valid frame sync pattern in the Search (data loss) modes.
- El — Lock Allowable Errors: Maximum number of bit errors allowed in a valid frame sync pattern in the Lock (data transfer) modes.
- Fl — Lock Allowable Patterns: Number of consecutive invalid frame sync patterns *allowed before* transition to Search mode.
- BS — Window: Number of bit slips allowed in either direction, except in the Search mode.

Similarly, the FEPS FSS may be described with the four-parameter shorthand, "(N1, N2, N3, N4)," where the variables are defined as

- N1 Maximum Errors: Maximum number of bit errors allowed in a valid frame sync pattern in *any* state.
- N2 Search -> Lock: Number of consecutive valid frame sync patterns to transition from Search to Lock mode.
- N3 Lock -> Search: Number of consecutive invalid frame sync patterns to transition from Lock to Search mode.
- N4 Sync Aperture: Number of bit slips allowed in either direction, except in the Search mode.

The FDP has more flexibility. The FSS may be described with the six-parameter shorthand, "($N1_{SC}$, $N1_{LF}$, N2, N3, $N4_C$, $N4_{LF}$)," where the variables are defined as

- $N1_{SC}$ "Check Sync Bit Error Tolerance" Maximum number of bit errors allowed in a valid frame sync pattern in the Search and Check states.
- $N1_{LF}$ "Lock/Flywheel Sync Error Tolerance" Maximum number of bit errors allowed in a valid frame sync pattern in the Lock and Flywheel states.
- N2 "Check Tolerance (# of frames)" Number of consecutive valid frame sync patterns to transition from Check to Lock states.
- N3 "Flywheel Tolerance" Number of consecutive invalid frame sync patterns to transition from Flywheel to Search states.
- $N4_C$ "Check Slip Tolerance" Sync Aperture: Number of bit slips allowed in either direction in the Check state.
- $N4_{LF}$ "Lock Slip Tolerance" Sync Aperture: Number of bit slips allowed in either direction in the Lock and Flywheel states.

Also, the FDP allows the user to select whether data is transferred in the Check and Flywheel states. Data is always transferred in Lock and always discarded in Search.

Notice that the parameters are not one-to-one between the FEPS, the FDP, and the Aydin 403. Further, the FEPS has special parameters, "Sync Qualifier" and "Frame Lock Status," which must be set to "LOCK/CHECK" so that data is transferred in all the lock states. These parameters default to "LOCK." The theoretical equation developed in this paper does not apply to the FEPS or FDP default states, and significantly more data loss will result if data is only transferred in the first lock state.

4.3. Interpretation of Real-World Results

Test results will not always match the theoretical performance, and some of the reasons are summarized in this section. However, one must first make certain that the parameters of the frame sync strategy are all clearly understood.

1. **Test results are scattered around theoretical.** Obviously not a problem, this is just a result of using finite sample sizes to draw conclusions about the actual bit error rate and packet data loss. T. P. Kelley developed a relationship between sample size, tolerance, confidence level, and the total probability of a bad frame sync pattern:

$$\pm \varepsilon_{PDL} = \sqrt{\frac{(1-P_b)}{(1-C)M_f P_b}},\ P_b = (1-P_{FSS})PDL + P_{eFSL}(1-PDL)$$

where

ε_{PDL} is the tolerance of the test run

C is the confidence level that a test result will fall within tolerance

M_f is the number of frames tested

P_b is the composite probability of a bad frame sync pattern

An approximate relationship for the accuracy of the bit error rate test was developed (*Theoretical Accuracy for ESTL Bit Error Rate Tests*, C. A. Lansdowne, EV4-95-601, September 1995), summarized as:

C	$\pm\varepsilon_{BER} \approx$
0.99	$2.575835\sqrt{\frac{P_e(1-P_e)}{M_b}}$
0.95	$1.959961\sqrt{\frac{P_e(1-P_e)}{M_b}}$
0.90	$1.644853\sqrt{\frac{P_e(1-P_e)}{M_b}}$

where

ε_{BER} is the tolerance of the test run

C is the confidence level of the tolerance

M_b is the number of bits tested

P_e is the probability of a bit error

Independence of errors is assumed, so for "Mark" or "Space" encoding, replace P_e with P_{eL} in this equation. The two equations may be combined to form an error ellipse on the PDL *vs*. BER curve to find the likelihood that the theoretical is a "true" representation of reality. The ellipse is given by,

$$PDL = PDL_{CALC} \pm \varepsilon_{PDL}\sqrt{1-\frac{(BER - BER_{MEAS})^2}{\varepsilon_{BER}^2}}$$

where

BER_{MEAS} is the measured bit error rate

PDL_{CALC} is the PDL calculated for the measured bit error rate

BER is the *x*-axis variable

PDL is the *y*-axis variable

Notice that if the BER was measured over the same period as the PDL test run, then even if the measurement does not represent the true steady-state system BER it still accurately measures how many bit errors were observed during the PDL test.

2. **All test runs show too little data loss at a given bit error rate.** This is an effect of a non-uniform distribution of errors. When errors occur in bunches, only the frames near the bunch are affected, and between bunches the data is relatively clean. Possible causes: convolutional coding, differential encoding, inter-symbol interference, or instability in the system. Data inversions and bit slips can cause large numbers of errors to be measured, while the frame synchronizer may be programmed to detect and correct these problems.

3. **All test runs show too much data loss at a given bit error rate.** This should never happen. Logically, this test result is saying that errors are more likely to occur in a frame sync pattern than in the rest of the frame. This could actually happen if for example the frame sync pattern contains longer runs between transitions than the data or the sequence used to measure the bit error rate. Otherwise, there is an equipment failure, the frame sync strategy is misunderstood, or the test technique is giving false measurements. Check that data is not being discarded between the bit synchronizer and frame synchronizer for causes other than the frame-sync strategy. Perhaps the radio receiver is dropping in and out of lock. Remember also that bit slips, if not "allowed," cause extra data to be lost.

A. Appendix. Validating Data

This section compares the theoretical equations developed in the previous sections to actual data taken in the ESTL.

A.1. Bi-Phase-L Links

The Space Shuttle uses Bi-Phase-L for command and telemetry links, and can be operated with convolutional encoding disabled. Payloads also use this approach. Performance of a bit synchronizer evaluated for Hubble Space Telescope is included.

EV4-95-713 Hubble Space Telescope/Orbiter/Aydin Bit Sync Evaluation Tests

SET-081495-01	403 (0,0,0,0)	HST Return Link	32k, Norm, TN/F
SET-081495-02	403 (0,0,0,0)	HST Return Link	32k, Invert, TN/F
SET-081595-01	403 (0,0,0,0)	HST Return Link	4k, Norm, AN/F Random
SET-081595-08	403 (0,0,0,0)	HST Return Link	4k, Invert, AN/F Random
SET-081595-11	403 (0,0,0,0)	HST Return Link	4k, Norm, AN/F 2AAAAB
SET-081595-12	403 (0,0,0,0)	HST Return Link	4k, Invert, AN/F 2AAAAB
SET-081595-09	403 (0,0,0,0)	HST Return Link	4k, Norm, AN/F FFFFFE
SET-081595-10	403 (0,0,0,0)	HST Return Link	4k, Invert, AN/F FFFFFE
SET-081695-01	403 (0,0,0,0)	HST Return Link	4k, Norm, D
SET-081695-02	403 (0,0,0,0)	HST Return Link	4k, Invert, D

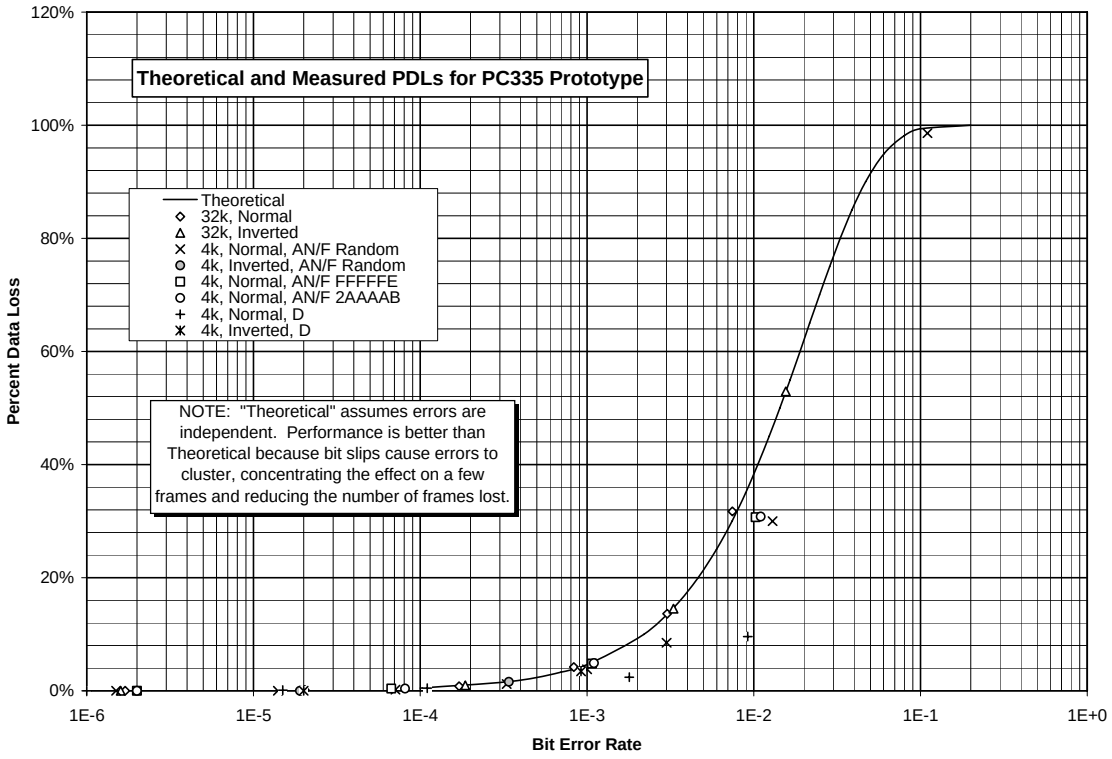

Figure A-1. Validation of Theoretical for "Level" Encoding

EV4-96-717 FEPS Frame Sync Strategy System Engineering Test

SET-051396-02B	403 (1,2,3,3)	Shuttle Return Link	S-band High Data Rate
SET-051396-04	FEPS (1,2,4,3)	Shuttle Return Link	S-band High Data Rate

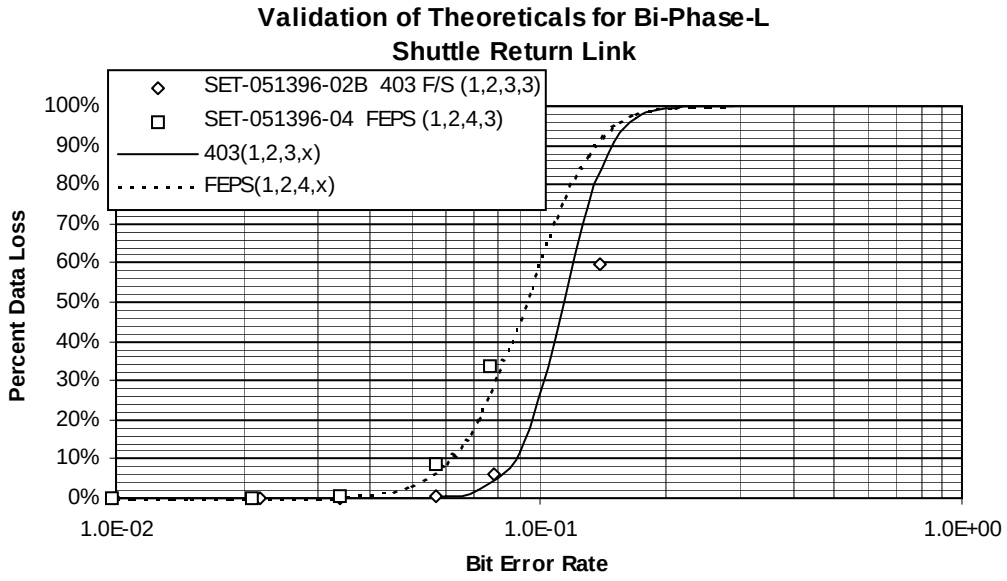

Figure A-2. Validation of Theoretical for Shuttle Return Link

For additional examples refer to T. Kelley.

A.2. NRZ-M Links

The Space Station uses block-coded NRZ-M (A.4). However, special tests were conducted to demonstrate the quality of the equations developed in this paper.

Unofficial Engineering Tests Conducted in Support of This Paper

As listed below 403 (x,x,1,3) ESTL Payload to PI S-band 1Mbps Data Rate

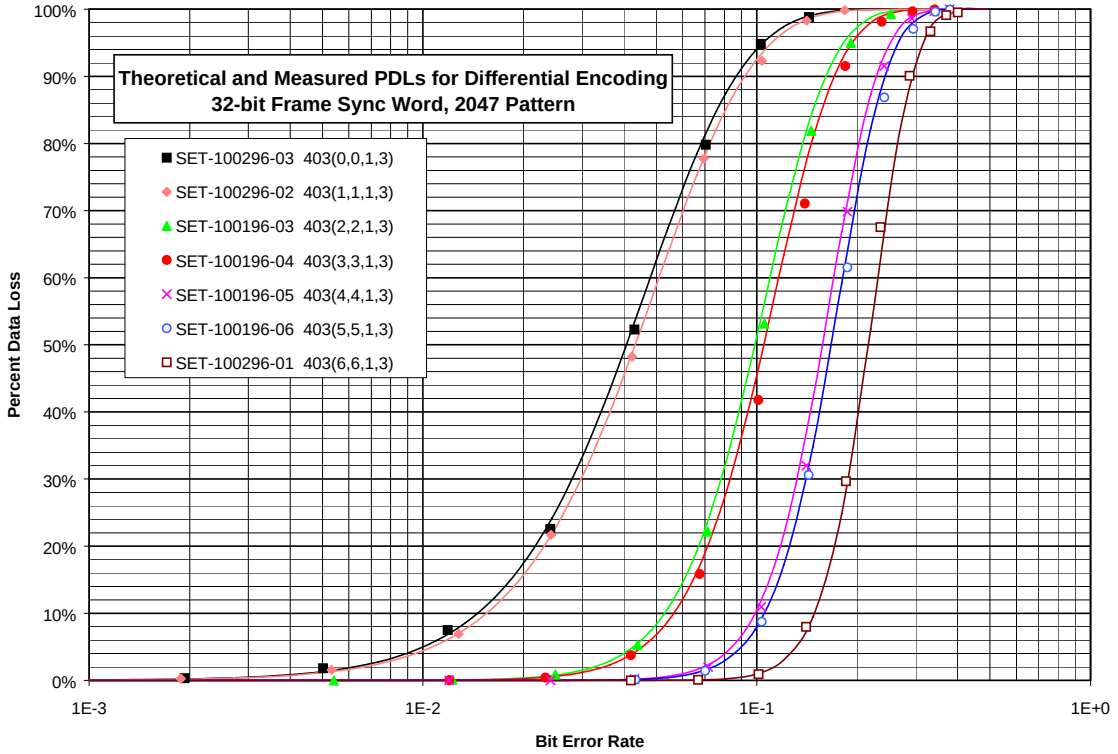

Figure A-3. Validation of Theoretical for NRZ-M

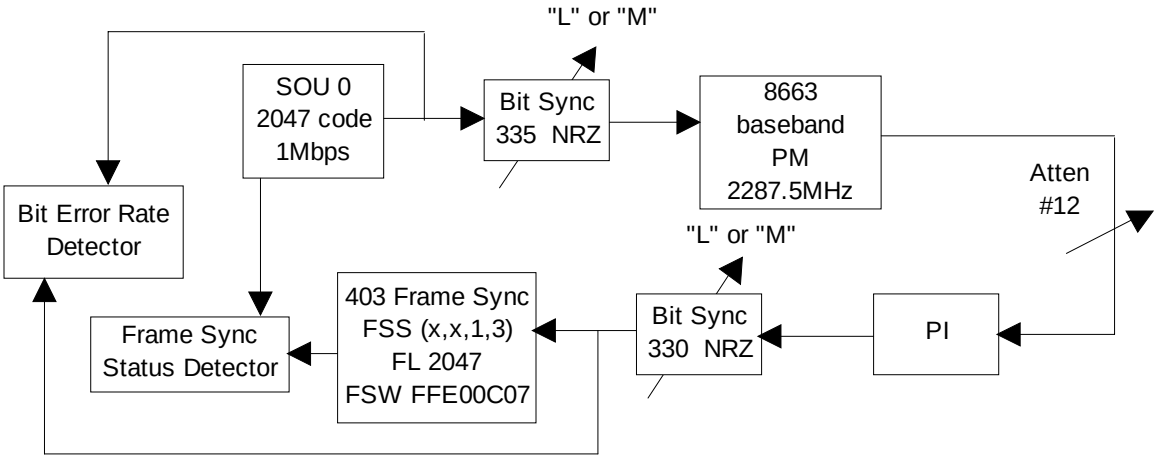

Figure A-4. Block Diagram for "Mark/Level" Engineering Tests

A.3. Coded Links

The Space Shuttle uses rate 1/3 convolutionally coded links for commands and telemetry. Examples of performance were extracted from tests and plotted.

EV4-96-717 FEPS Frame Sync Strategy System Engineering Test

SET-050996-02A	403 (1,2,3,3)	Shuttle Return Link	S-band	High Data Rate
SET-050996-02B	FEPS (1,2,4,3)	Shuttle Return Link	S-band	High Data Rate

EE7-91-708 Data Package for the STS-48 Countdown Anomaly Investigation

SET-101791-01	403 (1,2,3,3)	Shuttle Return Link	S-band	High Data Rate

EE7-80-702 Data Package for TDRS/Orbiter SSA Forward Link Vol. II

SVT-012280-02	403 (2,2,3,0)	Shuttle Forward Link	S-band	High Data Rate
SVT-012980-06	403 (2,2,3,0)	Shuttle Forward Link	S-band	Low Data Rate
SVT-012980-04	403 (2,2,3,0)	… Encrypted	S-band	Low Data Rate
SVT-012280-01	403 (2,2,3,0)	… Encrypted	S-band	High Data Rate

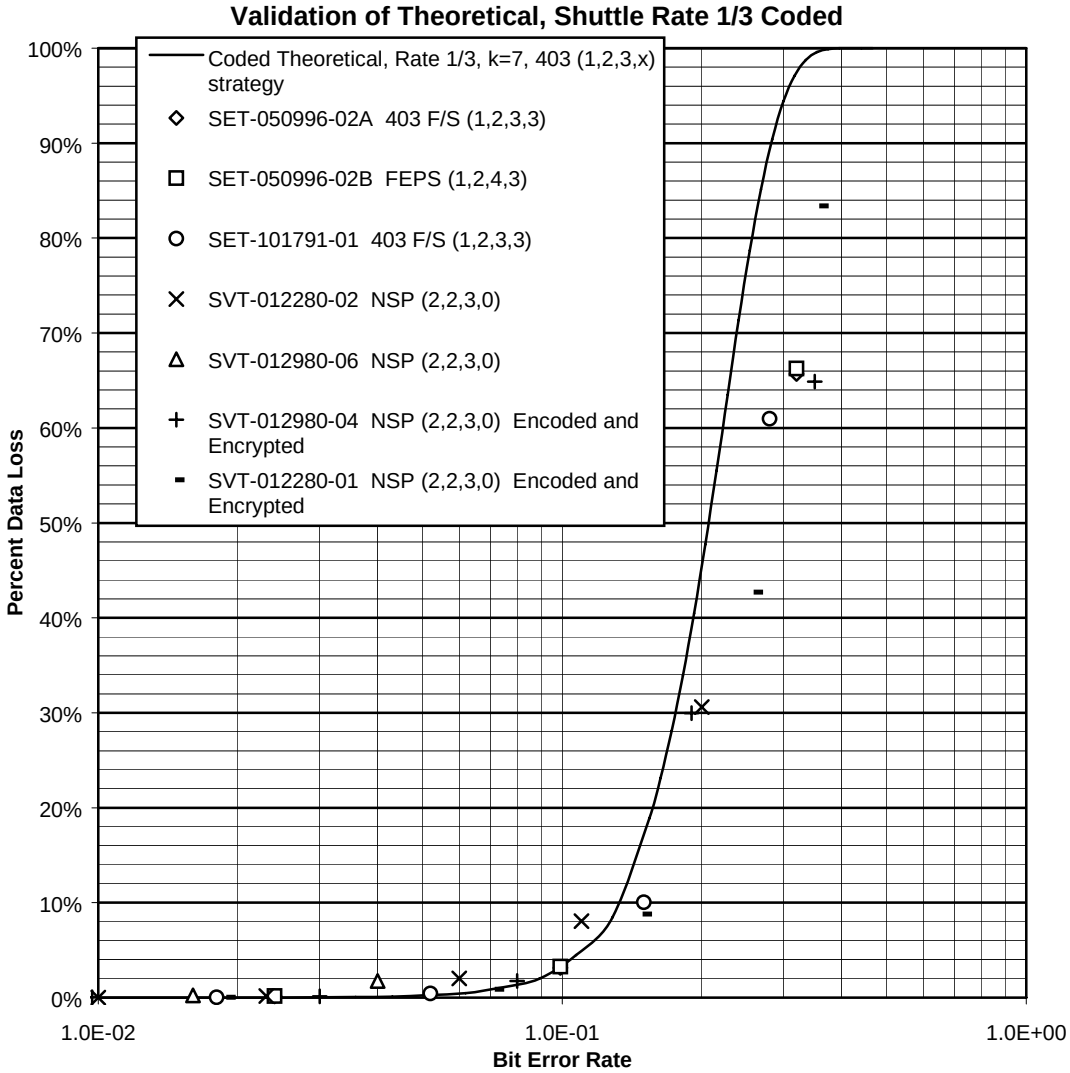

Figure A-5. Validation of Theoretical for Burst Errors

A.4. Various Data Transfer States

The SGS return link uses the FDP/FEP, which can optionally transfer data in Check and optionally transfer data in Flywheel. In fact, the default configuration transfers data in Check and not in Flywheel. This link is NRZ-M encoded. Examples of performance were extracted from tests and plotted.

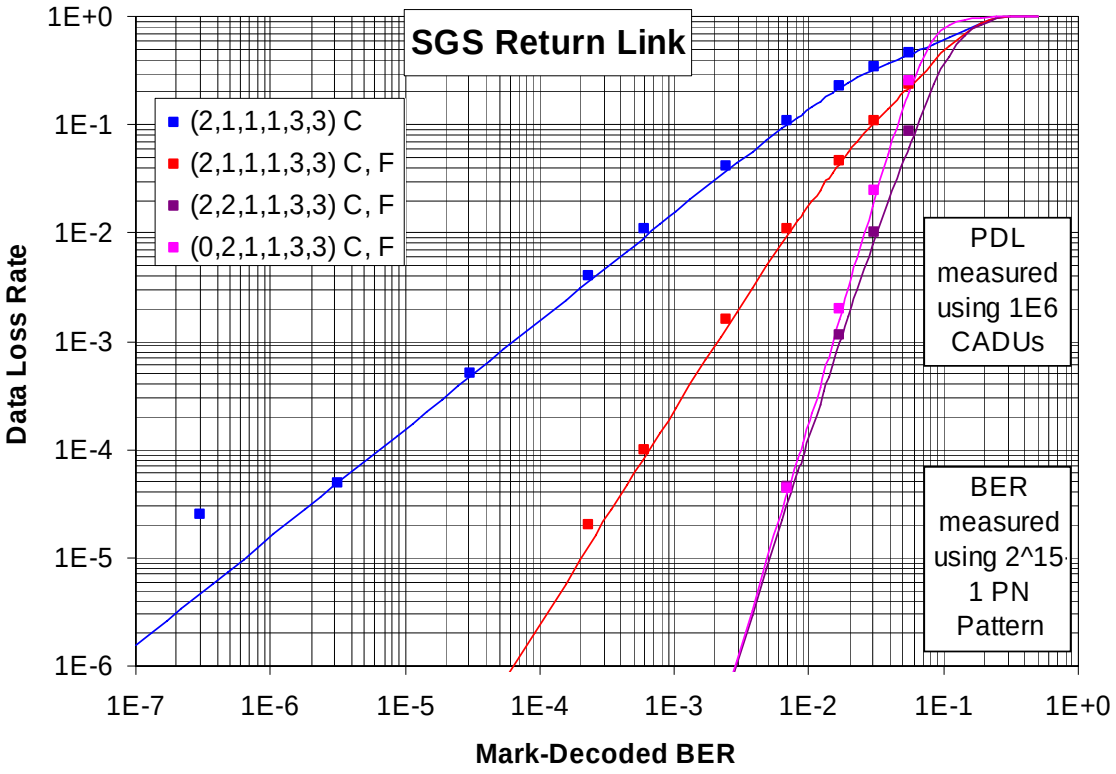

Figure A-6. Validation of Theoretical for Alternative Transfer States

B. Appendix. Analysis of Viterbi Decoder Bit Error Distribution

Models for error distribution in convolutionally decoded data are developed here. However, because of the complexity of Viterbi decoder behavior, data was taken and parameters of the models are set to match test results. This paper has already assumed that errors occur in independent bursts, with a 50% probability of bit error during the burst. Data bears out the hypothesis of independence of bursts, but also shows that the probability of bit error within a burst increases as the decoder approaches break-down. Data also shows how to determine burst rate from an observed bit error rate, in as much as burst length increases as the decoder breaks down. Test data in this appendix is also compared to data from a JPL simulation, *On the Error Statistics of Viterbi Decoding and the Performance of Concatenated Codes*, Miller, Deutsch, Butman; JPL Pub. 81-9, September 1981.

B.1. The JPL Model

JPL's approach used three parameters: frequency of bursts, average length of the bursts, and error density of the bursts. Distribution of burst spacing and burst size were modeled as geometric. One property of the geometric distribution is a specific relationship between the mean and the standard deviation, $\sigma = \bar{x}\sqrt{1-1/\bar{x}}$, which was not borne out in ESTL test results. However, the statistics that JPL computed from their simulation results are close to ESTL test results and are useful for comparison. Also, JPL simulated additional rates and constraint lengths that were not tested at ESTL.

In the tables that follow, JPL simulated a given E_b/N_o (dB), and found their bursts to be a certain average duration $\overline{B}$ (first bit error to last) with a certain average spacing $\overline{W}$ and average ratio of bits in error to bits in the burst θ . From these results I have additionally computed some other statistics. I backed out the average bit errors per burst by multiplying bits per burst $\overline{B}$ and error density θ . Then I found the bit error rate out of the decoder as bit errors per burst divided into bits per burst plus bits between bursts. Finally, I approximated an "inner bit error rate" as a ratio of bit errors per burst, less the two outer "known bad" bits, and bits per burst less the same two bits.

Table 7. Viterbi Decoder Burst Statistics (7, 1/2)

E_b/N_o	$\overline{B}$	$\overline{W}$	θ	BER	$\overline{B} \cdot \theta$	Inner BER
0.5	25.84	131.2	0.564	9.28E-02	14.57	52.7%
0.75	23.46	158.5	0.566	7.30E-02	13.28	52.6%
1	21.07	220.5	0.571	4.98E-02	12.03	52.6%
1.1	19.78	275.8	0.574	3.84E-02	11.35	52.6%
1.2	19.27	293.2	0.574	3.54E-02	11.06	52.5%
1.3	18.02	371.2	0.573	2.65E-02	10.33	52.0%
1.4	17.46	430.6	0.573	2.23E-02	10.00	51.8%
1.5	17.01	474.1	0.578	2.00E-02	9.83	52.2%
1.6	15.76	600.8	0.578	1.48E-02	9.11	51.7%
1.7	15.21	702.2	0.579	1.23E-02	8.81	51.5%
1.8	14.32	847.0	0.586	9.74E-03	8.39	51.9%
1.9	13.50	931.7	0.584	8.34E-03	7.88	51.2%
2	12.89	1122	0.590	6.70E-03	7.61	51.5%
2.1	12.86	1530	0.589	4.91E-03	7.57	51.3%
2.5	10.17	3258	0.599	1.86E-03	6.09	50.1%
3	8.67	9596	0.584	5.27E-04	5.06	45.9%
3.5	6.70	3.7E+04	0.630	1.14E-04	4.22	47.3%
4	4.40	2.0E+05	0.591	1.30E-05	2.60	25.0%

Table 8 Viterbi Decoder Burst Statistics (7, 1/3)

E_b/N_o	$\overline{B}$	$\overline{W}$	θ	BER	$\overline{B} \cdot \theta$	Inner BER
0.5	16.80	228.3	0.596	4.09E-02	10.01	54.1%
0.6	15.79	258.6	0.598	3.44E-02	9.44	54.0%
0.7	15.31	290.1	0.601	3.01E-02	9.20	54.1%
0.8	14.70	308.2	0.602	2.74E-02	8.85	53.9%
0.9	13.94	355.5	0.605	2.28E-02	8.43	53.9%
1	13.24	440.1	0.612	1.79E-02	8.10	54.3%
1.1	13.13	473.5	0.611	1.65E-02	8.02	54.1%
1.2	12.13	567.1	0.613	1.28E-02	7.44	53.7%
1.3	12.01	663.4	0.615	1.09E-02	7.39	53.8%
1.4	11.40	787.2	0.620	8.85E-03	7.07	53.9%
1.5	11.30	980.8	0.624	7.11E-03	7.05	54.3%
1.6	10.79	1146	0.622	5.80E-03	6.71	53.6%
2	9.46	2556	0.635	2.34E-03	6.01	53.7%
2.5	7.53	8613	0.653	5.70E-04	4.92	52.8%
3	6.35	2.9E+04	0.685	1.50E-04	4.35	54.0%
3.5	5.25	1.2E+05	0.672	2.94E-05	3.53	47.0%

Table 9. Viterbi Decoder Burst Statistics (10, 1/2)

E_b/N_o	$\overline{B}$	$\overline{W}$	θ	BER	$\overline{B} \cdot \theta$	Inner BER
0.5	37.98	162.4	0.511	9.69E-02	19.41	48.4%
0.6	35.99	184.8	0.512	8.35E-02	18.43	48.3%
0.7	32.72	221.7	0.517	6.65E-02	16.92	48.6%
0.8	30.11	248.5	0.515	5.57E-02	15.51	48.0%
0.9	28.07	292.9	0.518	4.53E-02	14.54	48.1%
1	26.98	353.0	0.523	3.71E-02	14.11	48.5%
1.1	25.76	440.9	0.522	2.88E-02	13.45	48.2%
1.2	25.16	526.7	0.530	2.42E-02	13.33	48.9%
1.3	22.86	601.0	0.530	1.94E-02	12.12	48.5%
1.4	21.15	857.6	0.537	1.29E-02	11.36	48.9%
1.5	21.13	983.6	0.531	1.12E-02	11.22	48.2%
1.6	20.86	1217	0.545	9.18E-03	11.37	49.7%
1.7	18.80	1566	0.541	6.42E-03	10.17	48.6%
2	16.95	4048	0.551	2.30E-03	9.34	49.1%
2.5	14.14	2.5E+04	0.585	3.31E-04	8.27	51.7%
3	11.25	2.5E+05	0.622	2.80E-05	7.00	54.0%

Table 10. Viterbi Decoder Burst Statistics (10,1/3)

E_b/N_o	$\overline{B}$	$\overline{W}$	θ	BER	$\overline{B} \cdot \theta$	Inner BER
0.5	25.29	398.1	0.533	3.18E-02	13.48	49.3%
0.6	24.84	455.3	0.532	2.75E-02	13.21	49.1%
0.7	22.06	549.4	0.539	2.08E-02	11.89	49.3%
0.8	21.37	642.4	0.541	1.74E-02	11.56	49.4%
0.9	20.76	813.0	0.540	1.34E-02	11.21	49.1%
1	19.34	990.1	0.540	1.03E-02	10.44	48.7%
1.1	19.09	1317	0.547	7.82E-03	10.44	49.4%
1.2	17.68	1606	0.546	5.95E-03	9.65	48.8%
1.3	16.33	2094	0.555	4.29E-03	9.06	49.3%
1.5	14.08	3245	0.566	2.45E-03	7.97	49.4%
2	11.21	1.6E+05	0.566	3.97E-05	6.34	47.2%
2.5	8.20	6.8E+05	0.646	7.79E-06	5.30	53.2%

B.1 ESTL Test Data

The ESTL test used a BPSK link with a Viterbi decoder, rate 1/2 and constraint length 7. A diagram of the test setup is given in Figure B-1. The test setup allowed control over received power level, and measurement of coded link error rate and packet data loss. Prior to testing, a baseline uncoded bit error rate was performed at 200kbps, corresponding to the symbol rate when coded; this provides an estimate of the symbol error rate observed at the decoder. At each power level tested, a sample of the decoded data was logged and a custom computer program was used to return the index number of every bit error in the file. Additional processing used as a rule that any error within 7 bits of another was part of a burst of errors; this rule was used to find the size (in number of bits and in number of errors) and separation of every burst. For each power level,

hundreds of bursts were recorded for analysis. The results can only be summarized here.

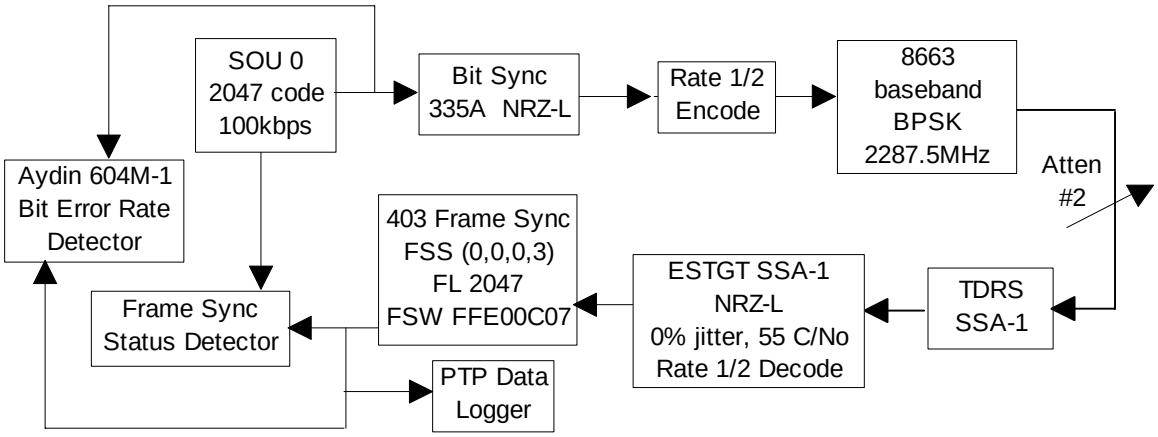

Figure B-1. Block Diagram for Coded Engineering Tests

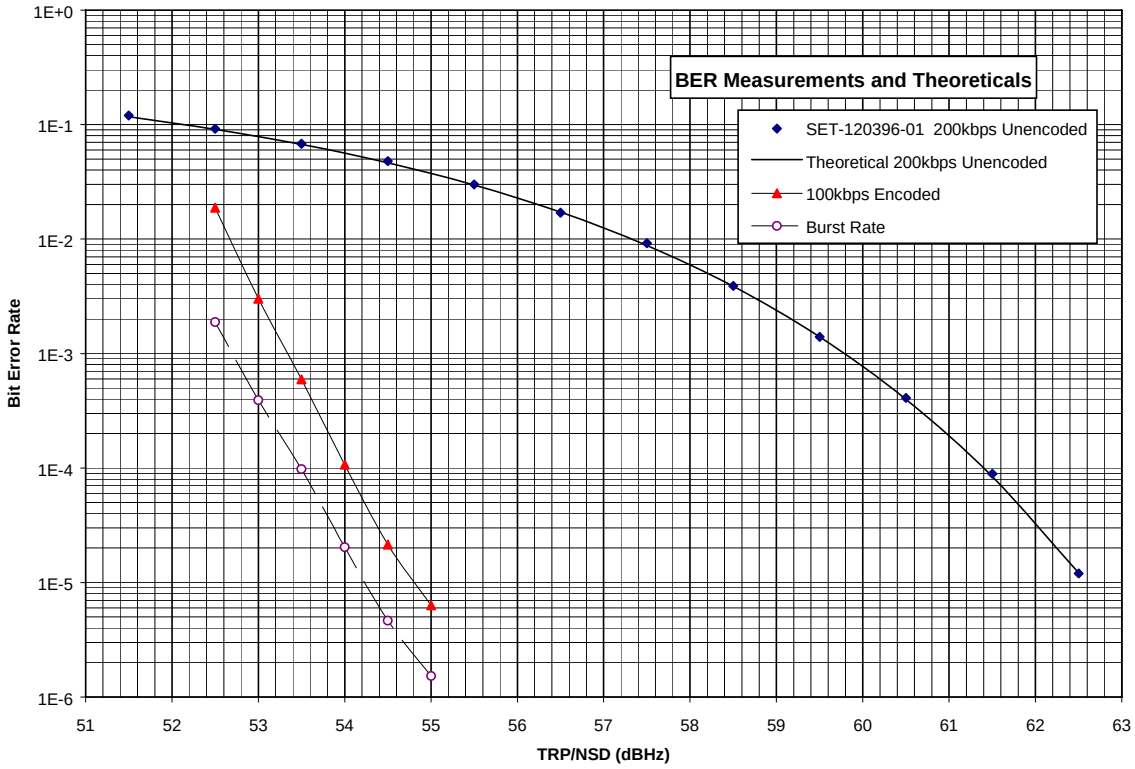

Figure B-2. Bit and Burst Error Rate Measurements

The curves shown in Figure B-3 are the actual percentage of bursts that contained a given number of bit errors (PDF), or the actual percentage of bursts that contained a given number of bit errors *or less* (CDF). They appear to be somewhat erratic, and do not follow any of the common probability distributions. This trait is real. It cannot be attributed to sampling error, as the bottom-most pair of curves ($2 \cdot 10^{-2}$ BER) were refined from more than 8000 points (Table 11).

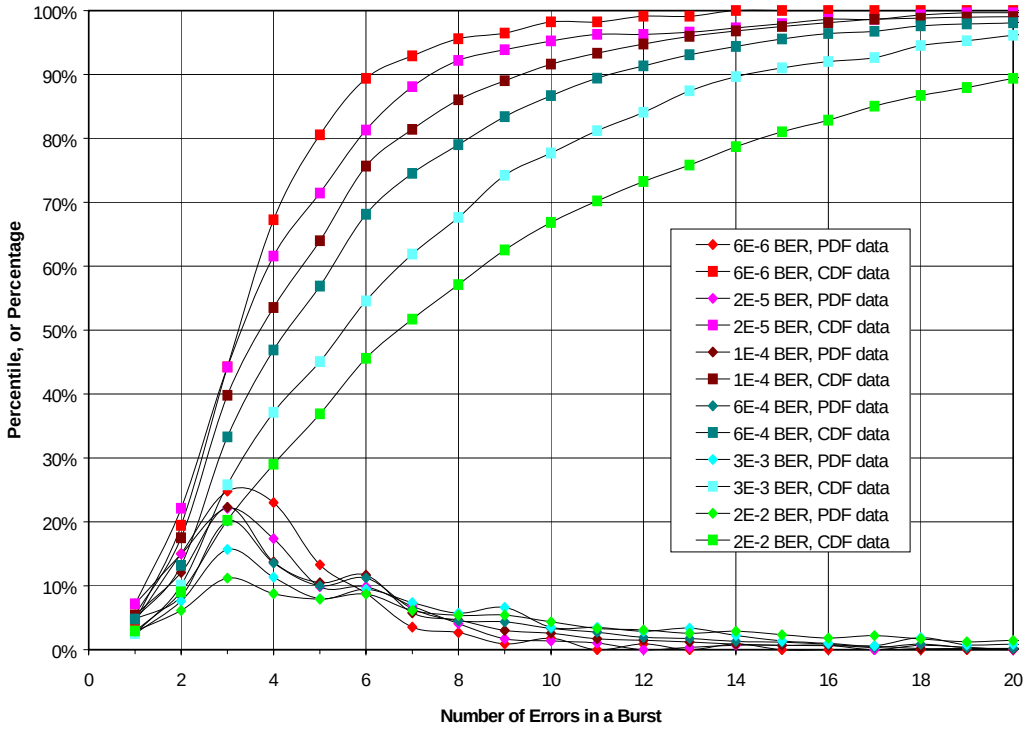

Figure B-3. Bit Errors per Burst Characteristic, Rate ½ Decoder

Table 11. Summary for Rate ½ Decoded Error Distribution Tests

TRP/NSD Setting	**55**	**54.5**	**54**	**53.5**	**53**	**52.5**
Measured Uncoded BER (hard decision)	**3.8E-2**	**4.6E-2**	**5.6E-2**	**6.7E-2**	**7.9E-2**	**9.1E-2**
Coded BER						
Measured PDL (0,0,0,3), 32-bit FSW	**0.02%**	**0.02%**	**0.22%**	**0.92%**	**4.02%**	**15.73%**
Simple Model PDL (BER & errors/burst data)	*0.01%*	*0.04%*	*0.16%*	*0.80%*	*3.38%*	*15.86%*
Bits Tested	**73901567**	**63252735**	**57272275**	**20449641**	**2047000**	**4767111**
Bit Errors	**469**	**1360**	**6139**	**12135**	**6173**	**84294**
Bursts of Errors	**113**	**294**	**1166**	**1997**	**803**	**8447**
Bits in Bursts	**740**	**2226**	**10156**	**20036**	**10701**	**145445**
BER (Bit Errors/Bits Tested)	6.35E-6	2.15E-5	1.07E-4	5.93E-4	3.02E-3	1.77E-2
Burst Rate (Bursts of Errors/Bits)	1.53E-6	4.65E-6	2.04E-5	9.77E-5	3.92E-4	1.77E-3
Bit Errors/Bursts of Errors (Errors per Burst)	4.2	4.6	5.3	6.1	7.7	10.0
Standard Deviation of Errors per Burst	**2.2**	**3.2**	**3.7**	**4.5**	**6.2**	**8.7**
Skewness of Errors per Burst	**1.7**	**2.1**	**2.1**	**2.1**	**2.4**	**2.3**
Kurtosis of Errors per Burst	**4.3**	**6.3**	**6.5**	**8.3**	**9.3**	**8.2**
Bits in Bursts/Bursts of Errors (burst length)	6.5	7.6	8.7	10.0	13.3	17.2
Standard Deviation of Bits per Burst	**3.7**	**5.6**	**6.7**	**7.9**	**11.4**	**15.6**
Skewness of Bits per Burst	**1.2**	**1.9**	**2.0**	**2.1**	**2.3**	**2.2**
Kurtosis of Bits per Burst	**3.5**	**5.3**	**6.2**	**8.0**	**8.1**	**8.1**
Bit Errors/Bits in Bursts (error density, theta)	63%	61%	60%	61%	58%	58%
Burst Inner Bit Error Rate (burst length > 2)	**47.8%**	**47.8%**	**49.1%**	**51.0%**	**50.3%**	**52.5%**
1/Burst Rate less Bits/Burst (waiting time)	653989.6	215137.8	49109.9	10230.1	2535.9	547.1
Average Bits between Bursts	**653977.4**	**215123.6**	**49093.5**	**10211.1**	**2532.6**	**513.7**
Std. Dev. of Bits between Bursts	**692211.5**	**219827.0**	**51334.9**	**10648.3**	**2508.2**	**531.5**
Fitted Burst Rate for Given BER	*1.81E-06*	*5.34E-06*	*2.22E-05*	*1.02E-04*	*4.31E-04*	*2.07E-03*
Errors per Burst from BER / Fitted Burst	*3.5*	*4.0*	*4.8*	*5.8*	*7.0*	*8.5*
Simple Model PDL (BER & fitted errors/burst)	*0.01%*	*0.04%*	*0.17%*	*0.80%*	*3.49%*	*17.91%*

In Table 11, actual data and statistics extracted directly from the data are boldfaced. Models are shown in italics for comparison with the data. All other entries are indirect statistics calculated from the data.

The number of correct bits between the bursts of errors (spacing of the bursts) is distributed so that the average is very nearly the standard deviation (Table 11). This is a characteristic of the exponential distribution, a probability distribution that is used to study time lapse between independent events that occur randomly at a uniform rate.

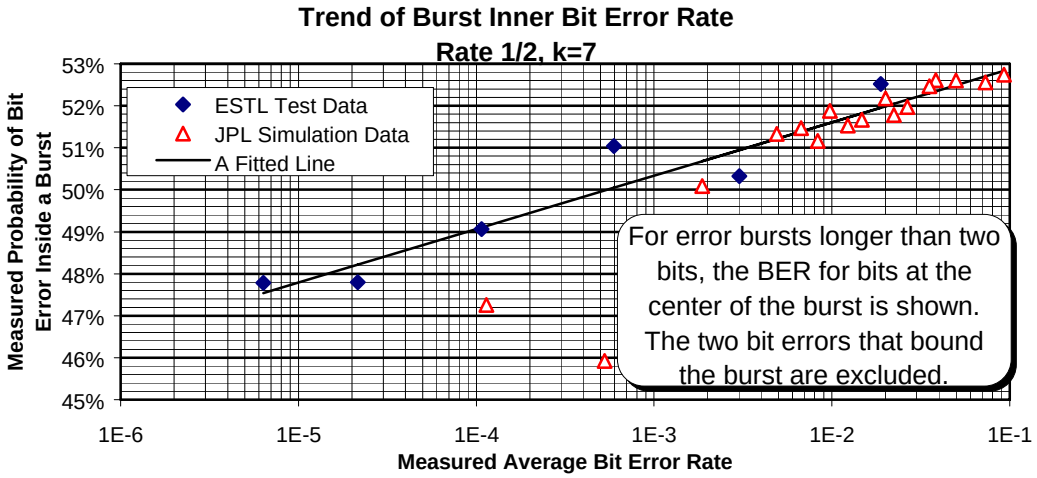

Figure B-4. Bit Error Rate within a Burst, as Decoder Breaks Down

Also of interest is the "Inner Bit Error Rate:" the ratio of error bits to tested bits, for bits between the two "known bad" bits that bound a burst error, considering bursts of more than two bits. This number is approximately 50%, but a trend can be observed in the ESTL test data, and is confirmed by the JPL simulation results. The data (Figure B-4) do not clearly define the shape of the trend, but it is clear that the decoder guesses more than half of the bits correctly early in the breakdown. As the breakdown progresses, the quality of the decoder's guesses declines until it is usually wrong.

B.2 Model for Burst Characteristics

Spacing of error bursts for Viterbi decoders can be accurately modeled by the exponential distribution. Error density within a burst can be approximated by 50%, but actually starts below 50% and increases as the decoder breaks down.

What can be modeled (for decoded BER below 10%) is the relationship between Burst Rate and Bit Error Rate. On the log-log scale (Figure B-5), the relationship is almost linear. Dependency on rate is so weak as to be negligible, and dependency on constraint length is slight. The relationship can then be expressed as, $\overline{W} \approx A(\text{BER})^{-r}$ where A and r are given in Table 12. These are parameters that were fitted to the data.

Table 12. Coefficients for Burst Rate Model

k	A	r
7	15.289	0.87572
10	17.956	0.89652

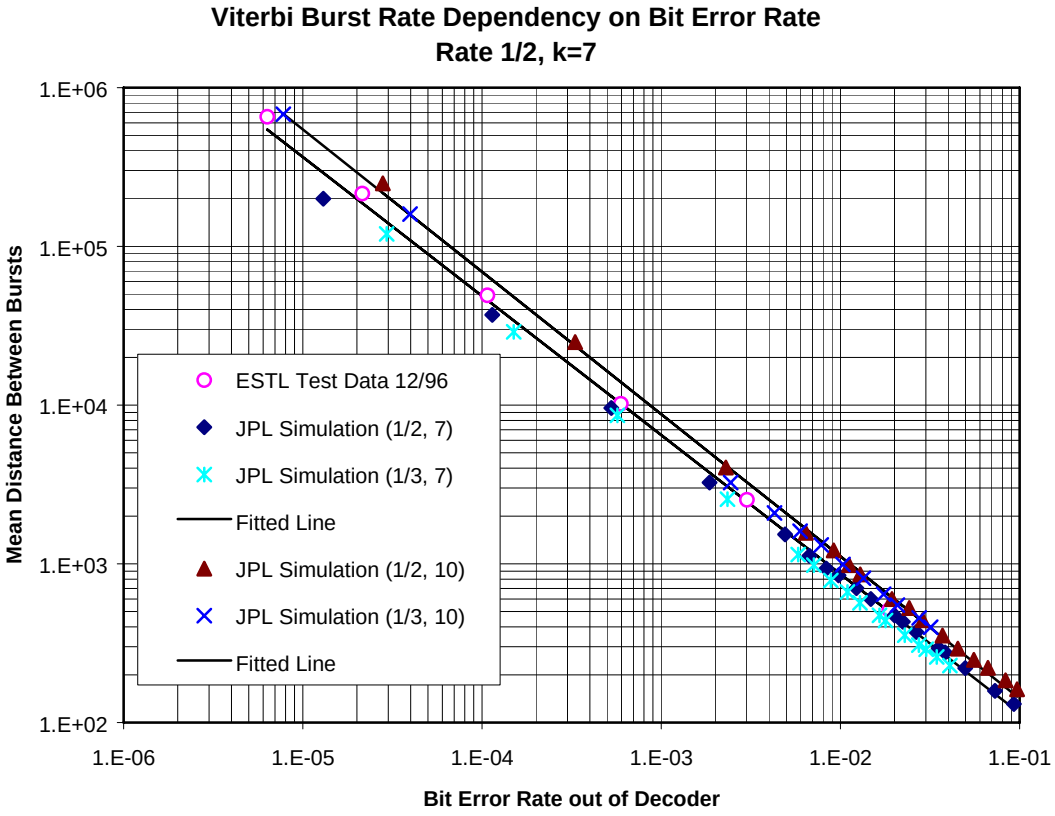

Figure B-5. Burst Rate as a Function of Decoded Bit Error Rate

For a given BER observed at the output of a decoder, the size and spacing of bursts can now be inferred. The average number of bit errors per burst can be found from

$$\frac{m}{2} = \overline{B}\theta = \text{BER} \cdot (\overline{W} + m)$$

which says that the bit error rate (ratio of bit errors to bits) multiplied by the sum of the average burst spacing and the average burst length, is the number of bit errors in a burst. This simplifies to,

$$\frac{m}{2} = \overline{B}\theta = \frac{\text{BER} \cdot \overline{W}}{1 - 2 \cdot \text{BER}} \approx \text{BER} \cdot \overline{W}$$

This model was developed for decoder output BER observed to be less than 10%. Results can be extrapolated somewhat higher, but beyond a BER of 20% or so other factors are at play. Data is not as poor as implied under these circumstances because most errors are concentrated in extremely large bursts of a size that is strikingly inconsistent with the distribution of burst sizes for the vast majority of bursts. These bursts are attributed to loss of clock quality. They are not explored further here because systems are never designed to operate in this region, and performance will be system-dependent.

Burst lengths for a given BER are distributed in a way that cannot easily be perfectly modeled. Certainly, the distribution is skewed. But some unusual tendencies are apparent, for example, beyond a 10^{-4} BER the decoder is more likely to produce a burst containing 3 or 6 errors than 5 or 7. Figure B-6 shows statistics generated from

ESTL data; for bursts up to a length of 5, the distribution of errors does not approach random. One can infer that at least for short bursts, errors tend to occur in specific patterns, presumably because the polynomials fail from specific weaknesses that could only be modeled accurately by simulation.

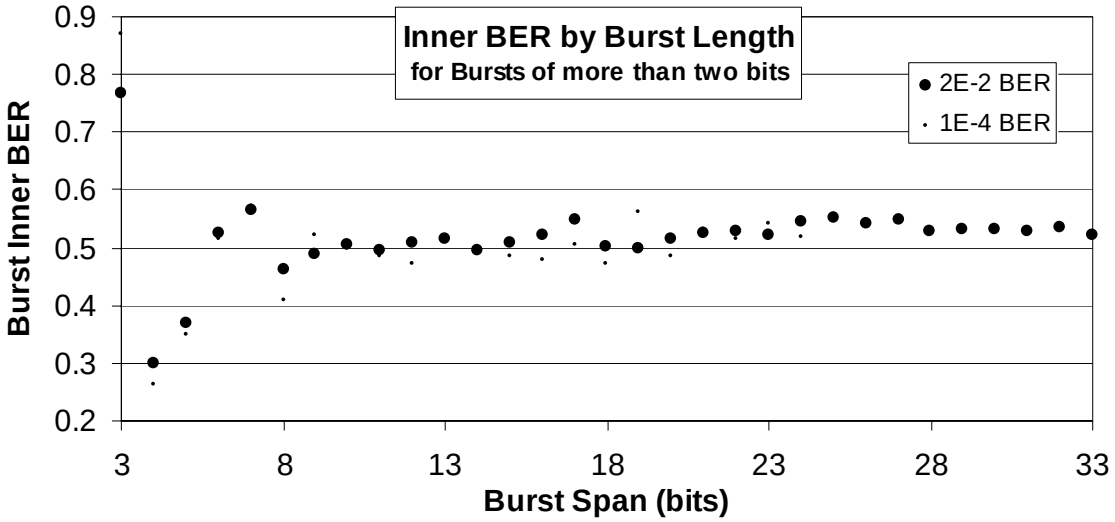

Figure B-6. Rate ½ Inner BER Distribution by Burst Length

The gamma distribution provides a fit for burst length that may be good enough for some applications. The gamma distribution is a function of two parameters $\alpha=k$ and $\beta=1/\theta$ and has properties,

$$\mu = \alpha\theta \qquad \text{Average}$$

$$\sigma^2 = \alpha\theta^2 \qquad \text{Variance}$$

$$\gamma_1 = 2/\sqrt{\alpha} \qquad \text{Skewness}$$

$$\gamma_2 = 6/\alpha \qquad \text{Kurtosis}$$

These characteristics can be compared with the statistics in Table 11.

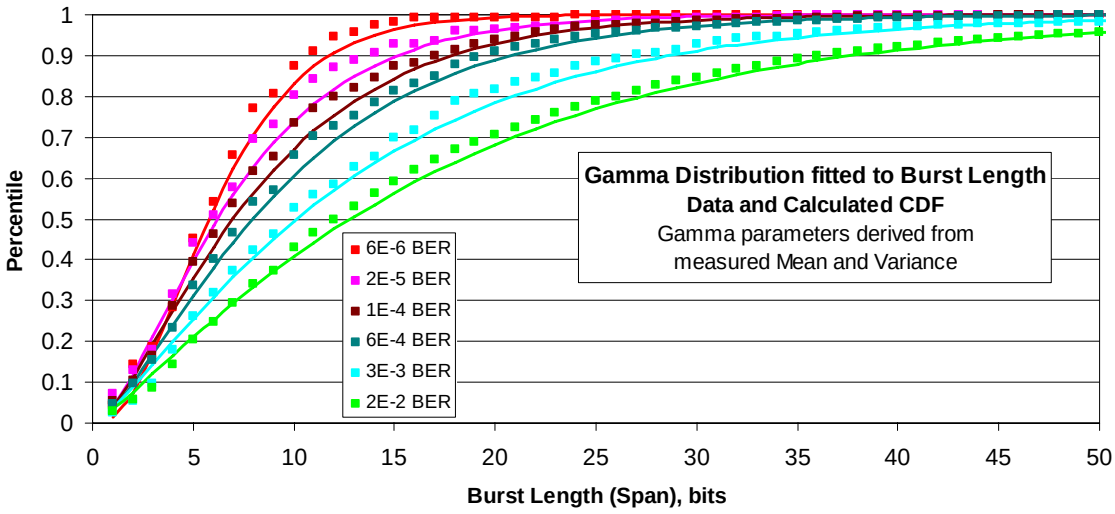

Figure B-7. Gamma Distribution Fit for Burst Length

C. Appendix. Analysis of False Lock Properties

Discussion in this paper of theoretical performance can, of course, be applied when making design choices about frame synchronization strategy. When selecting a specific frame synchronization pattern [FSP], one must consider what length to use and how many errors to allow. But there is one additional consideration: it is desired that false detection of the FSP in the data stream be unlikely and infrequent. This event is made unlikely by choosing a FSP that is different from itself (when shifted) and unusual in the data stream. False locks are made infrequent by restricting the number of bits per second that are searched for the FSP, by applying knowledge about the data structure.

The frequency of data or noise looking like a FSP depends on the size of the FSP, the fidelity of the identification (bit errors allowed), the composition of the bit stream, and how much of the data stream is being checked for the FSP.

Looking continuously through a random data stream for a perfectly matched (no errors) FSP n bits long, a match will be found every 2^n bits on average (n bits that can each be 1 or 0 can form 2^n combinations). For example, if n=24, a match for the FSP will occur every 16,777,216 bits on average; at 500kbps, this is 33.6 seconds.

If the FSP is allowed to be inverted, we now accept two of the 2^n possible combinations as matches, and a match will be found (in the example) every 17 seconds on average. If errors are permitted, the effect is the same in that more of the possible combinations represent matches. For one error allowed, n new combinations match, plus the error-free combination. If inversions are allowed, this doubles to $2n$+2. Generalizing, for a random data stream continuously parsed,

$$P_{FL} = \frac{i \cdot \sum_{j=0}^{e} C(n, j)}{2^n}$$

where,

- P_{FL} is the probability of a false lock (a random FSP match)
- i is 2 if inversion is allowed, 1 if not
- e is the number of bit errors permitted
- n is the length in bits of the FSP

and $C(n,j)$ is the combinations function defined on page 10, which gives the number of combinations of j objects that can be chosen from a population n.

The frequency of false FSP recognition may be further reduced by reducing exposure. This is what happens when noise is squelched if no signal is present, or if, once a FSP is found, searching is inhibited until another FSP is expected. Every bit-position searched from is an opportunity to false lock.

Finally, control of the content of the data stream can influence the probability of false lock. Select a FSP that is distant from idle or other patterns that are common in the data. Distance the FSP from the data by selecting a sequence of bits that is invalid as data. As error concentrations in the data increase to a level where these distances are consistently violated, the asymptote for performance is P_{FL} derived above. That is, a data stream with many errors degenerates to a stream of random bits.

Using run-length limiting [RLL] to distance the FSP from data can cause system-specific effects that cannot be easily or accurately modeled. For example, the FSP may be more susceptible to bit errors than other patterns. Or, as ESTL discovered, the UW may appear falsely in system noise more often than other patterns. Informal ESTL simulations showed that low frequency content was dominant. E5CB96h, for example,

was distributed very much as the analysis predicts, but E07E07h occurred about two to three times more often; in random data, any pattern would occur equally often. Properties of the noise are shaped by the specific receiver and bit synchronizer, so any theories about performance should be tested before design decisions are completed.

Probabilities developed in this section can be weighted by the parts of the time that different conditions prevail, and summed to give a net estimate of effect.

Some popular choices of FSP are given in Table 13. These patterns have been selected to be very distinguishable from offset versions of themselves.

Table 13. Optimized Frame Sync Patterns

Bits	FSP, Binary	FSP, Hex
7	1011 000	B0
8	1011 1000	B8
9	1011 1000 0	B80
10	1101 1100 00	DC0
11	1011 0111 000	B70
12	1011 0110 0000	B60
13	1110 1101 0000 0	EB00
14	1110 0110 1000 00	E680
15	1110 1100 1010 000	ECA0
16	1110 1011 1001 0000	EB90
17	1111 0011 0101 0000 0	F3500
18	1111 0011 0101 0000 00	F3500
19	1111 1001 1001 0100 000	F9940
20	1110 1101 1110 0010 0000	EDE20
21	1110 1110 1001 0110 0000 0	EE9600
22	1111 0011 0110 1010 0000 00	F36A00
23	1111 0101 1100 1101 0000 000	F5CD00
24	1111 1010 1111 0011 0010 0000	FAF320
25	1111 1001 0110 1110 0010 0000 0	F96E200
26	1111 1010 0110 1011 0001 0000 00	FA6B100
27	1111 1010 1101 0011 0011 0000 000	FAD3300
28	1111 0101 1110 0101 1001 1000 0000	F5E5980
29	1111 0101 1110 0110 0110 1000 0000 0	F5E66800
30	1111 1010 1111 0011 0011 0100 0000 00	FAF33400
31	1111 1110 0110 1111 1010 1000 0100 000	FE6FA840
32	1111 1110 0110 1011 0010 1000 0100 0000	FE6B2840

Source: Decom Systems Inc. (DSI), San Marcos CA

For example, consider the 24-bit FSP, FAF320h, with 3 bit-slips allowed in either direction. When offset three bits either way, nine bit errors would have to occur (and the surrounding data bits must match) for a perfect correlation. If, say, two errors are allowed, then only seven more bit errors have to happen to reach correlation if the data bits match. A false positive correlation is more likely than a false inverse correlation because fewer errors must occur; however, the bit error rate required for any false correlation renders the entire data stream unusable unless the errors are in bursts.

Table 14. Auto-correlation Properties of FAF320h

XXX11111010111100110010000XXX	offset	distance	inverse
11111010111100110010000	-3	9	12
11111010111100110010000	-2	10	12
11111010111100110010000	-1	9	14
11111010111100110010000	0	0	24
11111010111100110010000	1	9	14
11111010111100110010000	2	10	12
11111010111100110010000	3	9	12

These choices of FSP appear to be sub-optimal for the special case that a data stream could not only be inverted but also played backwards, especially if the state is not known prior to examination of the data stream.

D. Appendix. Theoretical Performance of Reed-Solomon Block Codes

This appendix will forego an explanation of how Reed-Solomon works and instead only summarize what parameters drive performance and how to calculate performance. Such a summary is appropriate here for two reasons. The math used in Section 3.2 to determine how often a frame sync pattern is bad can be used to determine how often a Reed-Solomon symbol is in error. Secondly, since Reed-Solomon is a block code it must be processed after frame synchronization, so it is important to select frame synchronizer performance to maximize situational Reed-Solomon performance.

D.1. Definitions and Performance Summary

Reed-Solomon codes are used to both detect and correct errors within a block of data by way of a parity block (overhead) appended to the data block. Reed-Solomon codes work at a "symbol" (typically, byte) level, with one or several bit errors in a symbol causing the symbol to be incorrect; this gives Reed-Solomon codes an advantage in channels with error extension (*e.g.*, differential decoding, decryption, convolutional decoding). When Reed-Solomon is combined (called "concatenated") with convolutional coding the resulting performance is referred to as a "brick wall" because performance switches very quickly from perfect to failed.

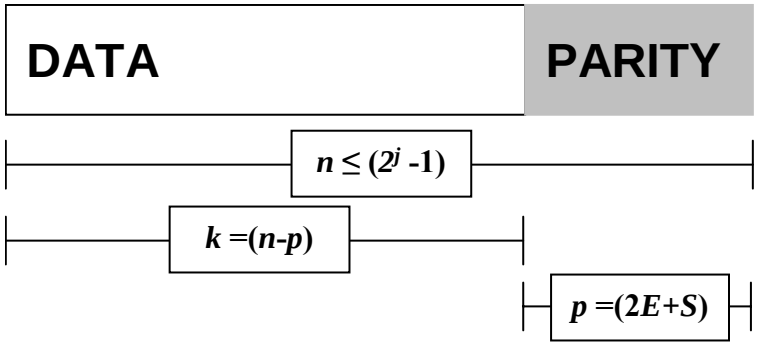

Figure D-1. Anatomy of a Reed-Solomon (*n*,*k*) Codeword

Parameters which describe a Reed-Solomon code are,

j	=	number of bits per symbol (usually 8, a "symbol" is a byte)
n	=	number of symbols per Reed-Solomon codeword
p	=	number of parity-check symbols per codeword, n-k
k	=	number of data symbols per codeword
S	=	number of "erasure" symbols per codeword
E	=	number of correctable symbol errors per codeword, (p-S)/2 (truncate if p is odd)
D	=	number of detectable symbol errors per codeword, (p-S)/2 if correction is used, (p-S) if correction is not used.
d	=	codeblock interleave depth
p^*	=	minimum distance (in symbols), n-k+1
η	=	efficiency or "code rate," k/n

The maximum or "natural" block size is related to the number of bits per symbol. The code can be strengthened by "shortening" the block; then the encoder and decoder both pad the block to the natural size using a known value (usually zero) in a known location (usually the beginning). The code can be weakened by using the natural size but not transmitting "erasure" symbols; these have a known location and can be reconstructed using only one parity symbol. Channel symbol errors must be reconstructed using two parity symbols, one to locate the error and one to correct it. Reed-Solomon codeblocks are often "interleaved" together within a larger data frame to improve performance on channels where large error extensions cause an error burst to span several symbols, this way adjacent symbols are corrected by different parity blocks. If the interleave depth is well suited to the channel then symbol errors will be independent, which simplifies the probability model.

Given a codeblock with symbol errors beyond the limit where detection is assured, there is a possibility that the errors occur in such a pattern that they form an apparently correct or (more likely) an apparently correctable codeword. This results in "mis-correction." An upper bound on the probability of a decoder error or mis-correction is given in *Primer: Reed-Solomon Error Correction Codes (ECC),* April 2004, an Application Note by Comtech AHA Corporation,

$$P_{DE} \le \frac{1}{\left(\frac{p^*-S-1}{2}\right)!} = \frac{1}{\left(\frac{p-S}{2}\right)!}, \quad p\text{-}S \text{ even}$$

$$P_{DE} \le \frac{1}{\left(\frac{p^*-S-2}{2}\right)!(2^j-1)} = \frac{1}{\left(\frac{p-S-1}{2}\right)!(2^j-1)}, \quad p\text{-}S \text{ odd}$$

There is an important misconception about the limit of assured detection for Reed-Solomon, reflected for example in the Altera application notes (*Reed-Solomon Compiler MegaCore Function User Guide*, January 2003, p. 12, Table 6). Many block codes can simultaneously detect twice as many errors as they correct. Reed-Solomon can also detect twice as many errors as it corrects, but not simultaneously; the designer must choose how the decoder will apply the parity symbols. Therefore, when Reed-Solomon is used for error correction, any uncorrectable codeword is a candidate for mis-correction.

D.2. Examples

Two examples are given with contrasting performance that will give the reader a feel for how weak or strong R-S coding can be and what design features lend strength.

EXAMPLE: ***OCA using Shuttle Ku-Band Forward Link***
The OCA uses a packet structure consisting of a 24-bit (3 byte) frame sync word followed by a single R-S (253,237) codeblock (shortened from R-S (255,239)) using 8-bit symbols. The Shuttle Ku-band forward link is uncoded (no error extension from convolutional decoding) and level-encoded (no error extension from differential decoding). The FSS uses one check state (F_i=2), two flywheel states (F_o=3), and two errors are allowed in the FSW while in lock (E_i=2) but no errors are allowed before lock (E_o =0). The FSW can be either polarity. This version of R-S appends 16 parity symbols and uses them for error correction, so it can detect and correct up to and including 8 symbol errors in the 253 transmitted symbols.

The probability of a symbol error is the probability that any given 8 bits are not error-free, P_{eS}=1-(1-P_e)^8. Using the technique from 3.2.1, the R-S discard rate is the probability of having more than 8 symbol errors in 253 symbols.

$$P_{eRS} = \sum_{i=0}^{E} \frac{n!}{(n-i)!i!} P_{eS}^{i} (1-P_{eS})^{n-i}, \quad n = 253, \quad E = 8$$

The probability that the detection threshold is exceeded is the same value,

$$P_{UD} = \sum_{i=0}^{D} \frac{n!}{(n-i)!i!} P_{eS}^{i} (1-P_{eS})^{n-i}, \quad n = 253, \quad D = 8$$

This is a precondition for decoder errors. From the formula in D.1, decoder errors occur at a rate as high as 1/(8!) =2.48E-5 for blocks which are beyond the threshold for guaranteed detection. This rate is easily verifiable, at 20Mbps and a 1E-2 BER the result is a mis-correct every four seconds. These relationships are shown in Figure D-2. Observe that essentially 100% of codeblocks are over the detection threshold above the 7E-3 BER, and so these are all exposed to the 2.5E-5 decode error rate; but, the frame synchronizer squelches error rates beyond 1E-1 to form the composite pedestal curve.

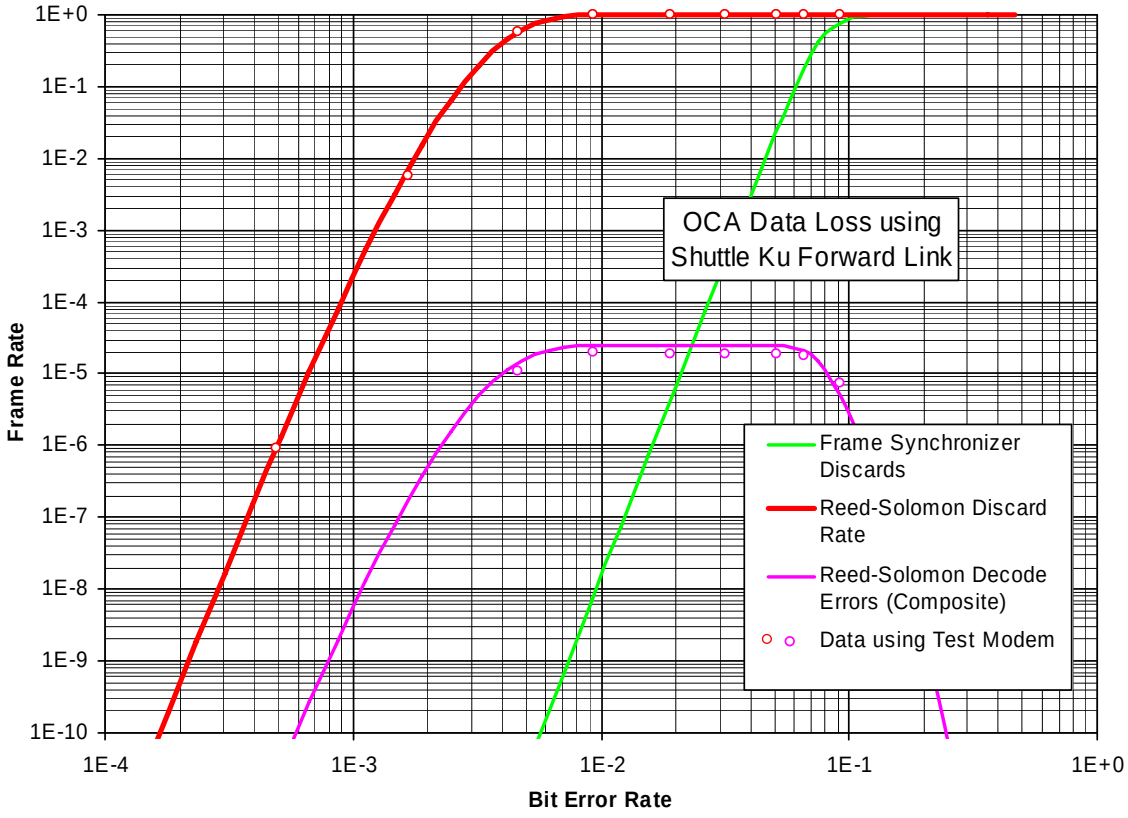

Figure D-2. Performance Plot for OCA using Shuttle Ku Forward Link

When the OCA is used in other situations the R-S and Frame Synchronizer performances will be different because the channel error distribution characteristics are different. Still, it is always the goal, since R-S can do a good job of detecting and correcting errors, to select a frame sync strategy that will first pass all correctable data through to the decoder, and secondarily squelch uncorrectable data.

EXAMPLE: ***CCSDS Packet Format using SGS Return Link***

The SGS uses a CCSDS packet structure (CADU) consisting of a 32-bit (4 byte) frame sync word followed by R-S (252,220) codeblocks using 8-bit symbols and interleaved to a depth of 5. The SGS return link is uncoded (no error extension from convolutional decoding) and Mark-decoded (one-bit error extension from differential decoding). The FSS uses five check states (F_i=6), three flywheel states (F_o=4); two errors are allowed in the FSW in Lock and Flywheel (E_i =2) but no errors are allowed in Search and Check (E_o=0). This version of R-S appends 32 parity symbols used for error correction, so it can detect and correct up to and including 16 symbol errors in the 252 transmitted symbols in the codewords. But all five interleaved codewords must appear correct, or the packet is discarded. Thus, a combination of correctable and mis-corrected codewords is necessary to form a mis-corrected packet.

The probability of a symbol error is again the probability that any given 8 bits are not error-free, P_{eS}=1-(1-P_e)^8. Using the technique from 3.2.1, the R-S codeword discard rate is the probability of having more than 16 symbol errors in 252 symbols. But packets are discarded if any one or more of the five interleaved codewords is incorrect.

$$P_{eRSCADU} = 1 - \left(1 - \sum_{i=0}^{E} \frac{n!}{(n-i)!i!} P_{eS}^{i} (1 - P_{eS})^{n-i}\right)^{d}, \quad n = 252, \; E = 16, \; d = 5$$

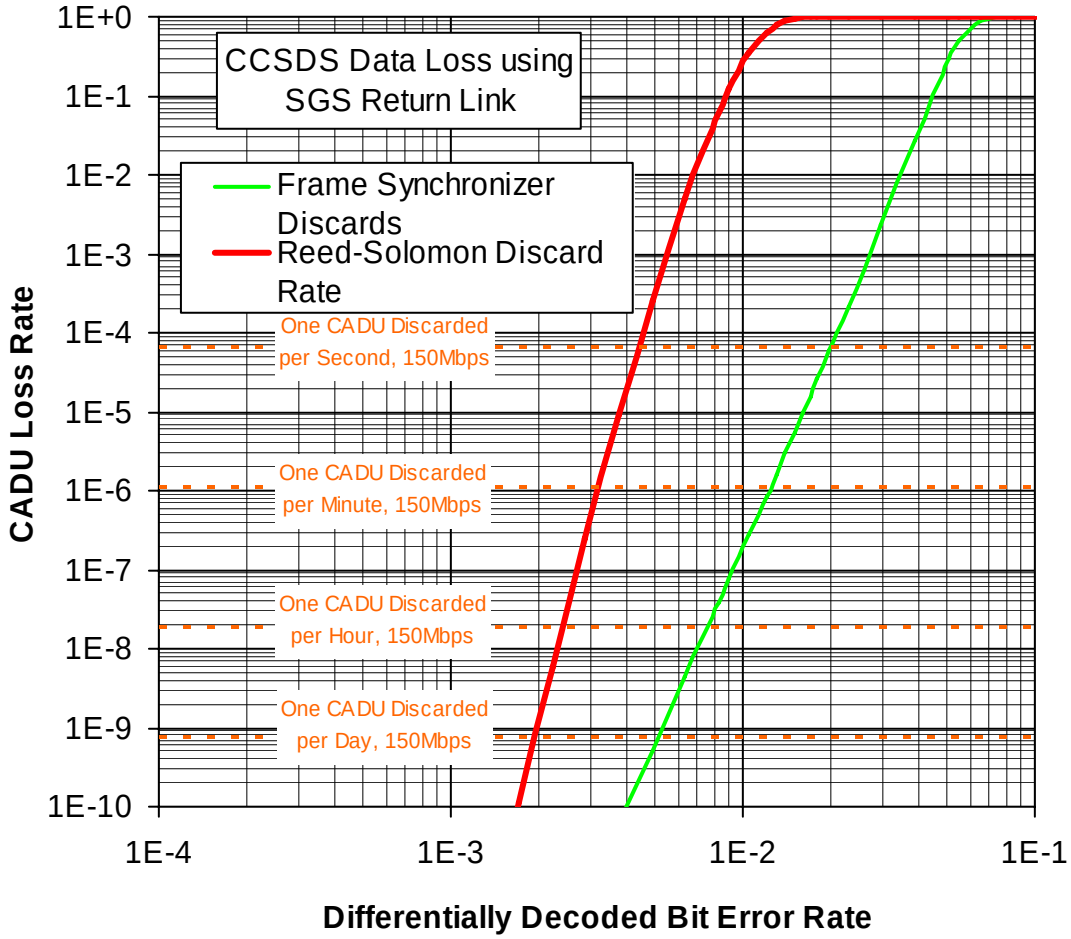

Figure D-3. Performance Plot for CCSDS using SGS Return Link

The probability that the guaranteed detection threshold is exceeded for a codeblock can be found similarly. But for a decode error in a packet to occur, it must have one codeword over the detection threshold and mis-corrected and four correctable (which can happen in five arrangements); or it must have all five over the detection threshold and mis-corrected (which can happen in one arrangement), or some combination weighted by the probability that the requisite number of mis-corrects could even happen.

$$P_{DECW} \le 1/16!$$

$$P_{eRSCW} = \sum_{i=0}^{E} \frac{n!}{(n-i)!i!} P_{eS}^{i} (1-P_{eS})^{n-i}, \; n = 252, \; E = 16$$

$$P_{UDCW} = \sum_{i=0}^{D} \frac{n!}{(n-i)!i!} P_{eS}^{i} (1-P_{eS})^{n-i}, \; n = 252, \; D = 16$$

$$P_{DECADU} = \sum_{i=1}^{d} P_{DECW}^{i} \left(\frac{d!}{(d-i)!i!} P_{UDCW}^{i} (1-P_{eRSCW})^{d-i} \right), \; d = 5$$

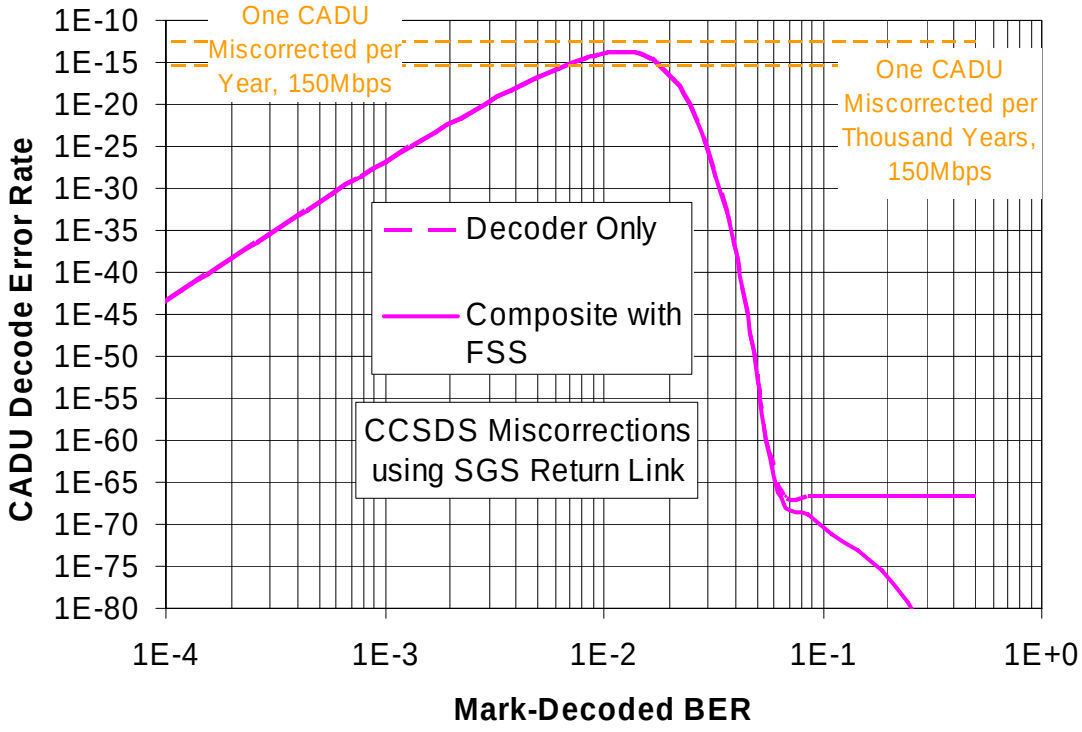

Figure D-4. Miscorrect Profile for CCSDS using SGS Return Link

These relationships describe performance beyond measurement that can only be determined by calculation. Below 1E-2, decode errors only occur as often as four codeblocks could have 16 or fewer errors while one codeblock has 17 or more symbol errors. This in itself is not a rare occurrence, peaking at 1.2E-2 BER at a rate of 41%, but it is squashed by the 1/16! (about 4.8E-14) rate at which that one codeblock could actually be mis-corrected, forming a composite peak at 2E-14, or once per 15 years—*of continuous operation at a 1.2E-2 BER*. The curve then declines and levels out at 2.5E-67 at 7E-2 BER as the dominant case becomes five mis-corrects in the same CADU. When the Frame Synchronizer performance is considered, error rates beyond

6E-2 are squelched and the performance ultimately declines to a terminal probability which must consider the probability of 6 perfect frame sync words (either polarity) appearing in random data; as often as this happens, 4 frames are flywheeled through to the decoder; then all 5 of the codewords in one of those frames must form nearly valid codewords. Mathematically,

$$P_{DECADU}\big|_{BER=50\%} = \left(\frac{2}{2^{32}}\right)^{6} \cdot 4 \cdot \left(\frac{1}{16!}\right)^{5} \approx 10^{-122}$$