



National Library
of Canada

Acquisitions and
Bibliographic Services Branch

305 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Public Information Services

Public Information Services

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

UNIVERSITY OF ALBERTA

**A NOVEL ASIC FOR COMBINED FRAMING AND ERROR CORRECTION IN
DS1 AND DS3 SIGNAL FORMATS**

BY

DAVID ANTHONY DUNN

**A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment
of the requirements for the degree MASTER OF SCIENCE.**

DEPARTMENT OF ELECTRICAL ENGINEERING

Edmonton, Alberta

SPRING 1994



National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Author form - Auteur (04/08/88) (en)

A. N. 709 - Auteur (04/08/88) (fr)

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-612-11196-2

Canada

UNIVERSITY OF ALBERTA
RELEASE FORM

NAME OF AUTHOR: **David Anthony Dunn**


TITLE OF THESIS: **A Novel ASIC for Combined Framing and Error
Correction in DS1 and DS3 Signal Formats**

DEGREE: **Master of Science**

YEAR THIS DEGREE GRANTED: **1994**

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided neither the thesis or any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.


David Anthony (Tony) Dunn
4904 144 Street
Edmonton, Alberta, Canada
T6H 4G8

DATE: Dec 20, 1993

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **A NOVEL ASIC FOR COMBINED FRAMING AND ERROR CORRECTION IN DS1 AND DS3 SIGNAL FORMATS** submitted by **DAVID ANTHONY DUNN** in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE**.



Dr. W. Grover, Supervisor



Dr. W. Krzymien, Internal Examiner



Dr. K. Stromsmoe, Internal Examiner



Dr. A. Basu, External Examiner

DATE: Dec 20/93

Discipline is never an end in itself, only a means to an end.

- King Crimson

Come Watson, come! The game is afoot.

- The Return of Sherlock Holmes; The Adventure of the Abbey Grange

When you have eliminated the impossible, whatever remains, however improbable, must be the truth.

- Sherlock Holmes, The Sign of Four

I know what I like and I like what I know.

- Genesis; I Know what I Like

They don't know terra cotta from spaghetti ricotta.

- Lovejoy

Take my hand I lead you to salvation,

Take my love for love is everlasting,

And remember a truth that once was spoken:

To love another person is to see the face of God.

- Les Miserables; Finale

Come forth from love's spire, born in life's fire, born in life's fire,

Come forth from love's spire, in the burning of our yearning for life to be,

And in pain there must be gain - New life!

They were sent to the gates, ride the tides of faith, ride the tides of faith,

They were sent to the gates, in the burning of our yearning for life to be,

There's no end to my life, no beginning to my death; death is life!

- Emerson, Lake, and Palmer; Pictures at an Exhibition

Love all God's creation, the whole and every grain of sand in it. Love every leaf, every ray of God's light. Love the animals, love the plants, love everything. If you love everything, you will perceive the divine mystery in things. Once you perceive it, you will begin to comprehend it better every day. And you will come at last to love the whole world with an all embracing love.

- Dostoyevsky; The Brothers Karamazov

*Dedicated to the memories of
Grandma Alice and Grandpa Michael*

I know that you are with me always.

Abstract

DS1 (1.544 Mb/s) and DS3 (44.736 Mb/s) multiplex signals are the backbone of the modern North American digital telecommunications transport network. DS1 and DS3 lines are also leased as private digital carrier facilities for use in customer networks. The raw bit error rates (BER) of such facilities are often in the 10^{-7} to 10^{-9} range [1,2]. This is low in the usual sense of BER's at which error-correction coding is applied, but there are a number of reasons to consider coding to guarantee ultra-low BER's of 10^{-12} to 10^{-16} and to protect against anomalous rises in BER.

To do so we have developed a strategy for coding existing DS1 and DS3 signals by reusing their framing sequences to bear the checkbits of a very high rate (0.9912) shortened Hamming code. We then solve the frame alignment problem in conjunction with FEC decoding. This strategy has been implemented in a 38k gate HCMOS VLSI gate array prototype to demonstrate feasibility and to validate theoretical predictions of reframing speed and BER reduction.

This thesis describes noteworthy aspects of the ASIC design, test methods, and results including: continuously compensated syndrome computation (CCSC) for fast frame alignment, circuits to deduce error pattern information from conditions in the decoder, design for testability, and system level test arrangements.

High fault coverage was achieved using a full serial scan design for testability strategy and a highly compact set of functional test vectors.

A laboratory test environment for system level testing and design validation was constructed and used to verify complete functionality of the ASIC. The BER improvement measured in the laboratory agrees very closely with the theoretical BER improvement of $1360 \cdot (\text{BER}_{\text{in}})^2$ for DS3 and $2963 \cdot (\text{BER}_{\text{in}})^2$ for DS1. Measured out-of-frame detection and re-frame times are also observed to agree with the predicted values, resulting in a speedup factor over conventional framing speeds of 4.9 for DS3 and 3.6 for DS1 by using FEC framing.

Applications of the ASIC include dribble-error reduction in fibre system design, throughput enhancement in fast packet (e.g. ATM) networks, advanced performance monitoring, and ultra-reliable leased lines for financial networking and automatic teller machine networks.

Acknowledgment

I would like to express my sincere thanks to my supervisor, Dr. Wayne Grover, Director of Networks and Systems at TRLabs for his guidance and assistance throughout this project. I would also like to thank the members of my examining committee, Dr. Witold Krzymien, Dr. Keith Stromsmoe, and Dr. Anup Basu for taking time out of their busy schedules to read this thesis.

I would also like to express my thanks and appreciation to the following people and institutions:

- Man Yau Cheung for his development of the theory and predicted performance of Continuously Compensated Syndrome Computation in his M. Sc. work.
- Kanu Emeruwa and Alex Ngi of LSI Logic for their assistance during the ASIC design phase of this project.
- Jim Slevinsky of AGT Limited and Dr. Mike MacGregor of TRLabs for many stimulating and thought provoking discussions. Special thanks to Mike for the long jogs in the river valley at lunchtime.
- Mr. Tim Friesen of LSI Logic and Jeff Weslowski for the development and construction of the laboratory test environment.
- Mr. David Clegg of TRLabs for his guidance in the laboratory and for many enjoyable discussions about music, CD's, and stereos.
- Dean Michaels, Bob Gregorish, and Dr. Mike MacGregor for excellent computing support at TRLabs.
- The students of TRLabs for their friendship and for making my time at TRLabs very enjoyable.
- The Natural Sciences and Engineering Research Council (NSERC), TRLabs, and the University of Alberta for financial support.
- All of the members of my band: Chris Shaw, Joanne Wotypka, Gary Bergmann, and Dave Olsen for many entertaining hours of music making and giving me another way to use my creative energies. Special thanks to Chris for all those Wednesday nights of cheap beer and trivia at the Renford Inn.

I would finally like to thank all of my family and friends, especially Mom, Joanna, Dave, and Meaghan for their continuous love and encouragement throughout the past two years. And I would most especially like to thank Sharon for her love and support, and for always being there when I needed her.

Table of Contents

1. INTRODUCTION	1
1.1 Motivation for DS1 and DS3 FEC.....	1
1.2 Issues Addressed in this Thesis.....	2
1.3 Other Related Work	3
1.4 Applications of FEC for DS1 and DS3.....	3
1.5 Outline of the Thesis.....	4
2. BACKGROUND	6
2.1 Conventional DS1 and DS3 Signal Formats.....	6
2.1.1 DS1 Superframe Format	6
2.1.2 DS3 Masterframe Format	6
2.2 Frame Alignment in a Digital Multiplex System.....	7
2.2.1 Conventional Reframing.....	8
2.2.2 Maximum Average Reframe Time	9
2.2.3 False-in-Frame Time.....	9
2.2.4 Conventional Out-of-Frame Detection	9
2.2.5 Out-of-Frame Detection Time	9
2.2.6 Misframe Time.....	10
2.3 Hamming Codes for Forward Error Correction.....	10
2.3.1 DS1-FEC Hamming Code and Codeword Structure	11
2.3.2 DS3-FEC Hamming Code and Codeword Structure	12
2.4 The Principle for FEC-Derived Framing	12
2.4.1 FEC-based Re-frame Procedure	13
2.4.2 FEC-based Out-of-Frame Detection Procedure.....	13
2.4.3 Initial Comparison of Conventional and FEC Reframing Speeds	14
2.5 Re-configurable Circuit Architecture	16
3. THE DS1/DS3 FEC ENCODER	17
3.1 FEC Encoder - Top Level	17
3.2 I/O Synchronization	19
3.3 Encoder (State Machine) Synchronization Module	20
3.4 Codeword Delay Register	22
3.5 Encoding Register	24
3.6 Parity Bit Register.....	28
3.7 Holding Register	31
3.8 Output Multiplexor	32
3.9 Encoder Event Sequencer	34
3.9.1 Control Signals.....	35
3.9.2 Sequencer Operation in DS1 Mode	35
3.9.3 Sequencer Operation in DS3 Mode	36
3.9.4 State Machine Decoding Logic Minimization.....	37

3.9.5	Synchronization of Event Sequencer to Incoming Data Stream.....	37
4.	THE DS1/DS3 FEC DECODER	39
4.1	Top Level Decoder Module	39
4.2	Chip Level I/O Interface	42
4.2.1	Input Re-Timing Flip Flops	42
4.2.2	Output Re-Timing Flip Flops.....	42
4.3	Syndrome Computation	42
4.3.1	The Process of CCSC Decoding for the DS3-FEC Code	43
4.3.2	The Process of CCSC Decoding for the DS1-FEC Code	47
4.3.3	Codeword Delay Buffer	50
4.3.4	DS1 and DS3 CCSC Compensation Terms Generation	52
4.3.5	CCSC (+G(x)) Register.....	53
4.4	Parity Computation Register.....	56
4.5	Frame Alignment Unit (FAU)	57
4.6	Error Correction Unit (ECU)	62
4.7	Output Conditioning	70
4.7.1	Error Correction Function.....	71
4.7.2	OH-Bit Sequence Reconstruction.....	71
4.8	Main FEC Decoder System Controller	72
5.	THEORETICAL FEC-DERIVED FRAMING AND ERROR CORRECTION PERFORMANCE OF THE DS1/DS3/FEC DECODER.....	74
5.1	Theoretical Framing Performance	74
5.1.1	Probability Generating Functions	75
5.1.2	Theoretical Maximum Average Reframe Time.....	75
5.1.3	Theoretical Out-of-Frame Detection Time	78
5.2	Theoretical Error Correction Performance	79
5.2.1	Single Error Correction in DS1-FEC and DS3-FEC Codes	79
5.2.2	Double Error Detection in DS3-FEC Code	79
5.2.3	Higher Order Error Detection in DS1-FEC Code.....	79
5.2.4	Higher Order Error Detection in DS3-FEC Code.....	80
5.2.5	BER of DS3-FEC Error Corrected Data Signal.....	81
5.2.6	BER of DS1-FEC Error Corrected Data Signal.....	81
6.	LABORATORY SETUP	83
6.1	Laboratory Test Environment - General	83
6.2	DS1 and DS3 Bit Error Rate Test Sets (BERTs).....	83
6.3	FEC Board: BERT Interface	86
6.4	FEC Board: FEC Encoder.....	86
6.5	Error Injector Board: Controlled Error Patterns	87
6.5.1	Walking Single Error Injection.....	87
6.5.2	One-shot Double Error Injection; One-shot Single Error Injection.....	89
6.5.3	Walking Triple Error Injection; Walking Double Error Injection	89
6.6	Random Error Injection	89

6.7	FEC Board: Out-of-Frame Generator	90
6.8	FEC Board: FEC Decoder	92
6.9	Line Interface Board	93
7.	SYSTEM LEVEL LABORATORY TEST - EXPERIMENTS AND RESULTS (VALIDATION OF DESIGN AND THEORY).....	94
7.1	Controlled Error Injection.....	94
7.1.1	DS1 Mode Correction of Single Bit Errors.....	94
7.1.2	DS3 Mode Correction of Single Bit Errors.....	95
7.1.3	DS3 Mode Detection of Double Errors	95
7.1.4	DS1 Mode Detection of Higher Order Errors.....	97
7.1.5	DS3 Mode Higher Order Error Detection.....	101
7.2	Random Error Detection / Correction Performance	107
7.2.1	Random Error Performance and Results for DS3.....	107
7.2.2	Random Error Performance for DS1 and Results.....	108
7.3	Experimental Framing Performance	109
7.3.1	Out-of-Frame Detection DS1 and DS3.....	109
7.3.2	DS1 Maximum Length Reframe Time Measurement	111
7.3.3	DS3 Maximal Length Reframe Time Measurement.....	116
7.3.4	Comparison of FEC Framing and Conventional Framing (DS1).....	122
7.3.5	Comparison of DS3-FEC framing to DS3 Conventional Framing....	123
8.	ENGINEERING THE ASIC.....	124
8.1	Power Consumption / Heat Dissipation.....	124
8.2	The Die.....	125
8.3	The Package	126
8.4	Bonding Diagram.....	126
8.5	Clock Routing Strategy..	127
8.6	Overall Savings From Logic Minimization	129
8.7	Design For Testability.....	129
8.8	Automatic Test Equipment (ATE) Vectors	131
9.	SUMMARY AND CONCLUSIONS	134
9.1	Summary of Results.....	134
9.2	Further Work.....	136
9.2.1	Towards a Production Device.....	136
9.2.2	Towards SONET FEC	137
	REFERENCES	138
	APPENDIX A: FEC ENCODER / DECODER SUPPORT MODULES.....	140
A.1	D - Flip Flop with Synchronous Load and Shift Enable.....	141
A.2	Catch C-Bit Module	142
A.3	5-Bit Magnitude Comparator.....	143
A.4	D-Flip Flop with Synchronous Load and Synchronous Clear	143

A.5	Error Pattern Detection Circuit	144
A.6	Frame Boundary Divided By 2	145
A.7	Sample FEC ENABLE	146
A.8	Correction Pulse Timing Latch.....	147
APPENDIX B: MUSTANG AND ESPRESSO INPUT AND OUTPUT FILES.....		149
B.1	FEC Encoder System Controller.....	150
B.2	Main FEC Decoder System Controller	153
B.3	Frame Alignment Unit System Controller.....	156
B.4	Error Correction Unit System Controller.....	158

List of Tables

Table 3.1	Load Data Input Functions for encoding register flip flops.....	28
Table 3.2	Decoding delay for each look-ahead checkbit to become valid	30
Table 3.3	Input to output delay characteristics of 12 input XOR gate.....	30
Table 3.4	Checkbit assignments to XOR gate inputs.....	31
Table 3.5	Encoder Event Sequencer Logic Functions for DS1	36
Table 3.6	Encoder Event Sequencer Logic Functions for DS3	36
Table 3.7	Control Signal Values for System Counter = TC and Zero	38
Table 4.1	Coefficients and exponents in $D(x)$ for the DS3-FEC code	45
Table 4.2	Remainders of DS3 Difference Polynomial Term Division by $G(x)$	46
Table 4.3	DS3-FEC CCSC Compensation Functions.....	47
Table 4.4	Coefficients and exponents in $D(x)$ for the DS1-FEC code	48
Table 4.5	Remainder of DS1 Difference Polynomial Terms Division by $G(x)_{DS1}$	49
Table 4.6	DS1-FEC CCSC Compensation Functions.....	50
Table 4.7	4:1 MUX Control Signal Truth Table.....	72
Table 4.8	State Counter Values for which each Control Signal is Asserted.....	73
Table 7.1	Expected Number of Errors on BERT for Various WBI, FBI Combinations for Double Error Injection in DS3 mode.....	96
Table 7.2	Observed Number of errors and error correction/detection activity for double errors where $FBI \in OH$ -bits in DS3 mode.....	97
Table 7.3	Expected Number of Errors on BERT for Various WBI, FBI Combinations for Double Error Injection in DS1 mode.....	98
Table 7.4	Expected Number of Errors Recorded by BERT (DS1 Double Errors)	98
Table 7.5	Observed Number of errors and error correction/detection activity for DS1 double errors where $FBI \in OH$ -bits.....	99
Table 7.6	Observed Number of errors and error correction/detection activity for DS1 double errors where $FBI \notin OH$ -bits.....	100
Table 7.7	Expected Number of Errors on BERT for Various WBI, FBI Combinations for Triple Error Injection in DS3 mode	102
Table 7.8	Expected Number of Errors Recorded by BERT.....	102
Table 7.9	Observed Number of errors on BERT and error detection correction activity for DS3 Triple errors when $FBI_1, FBI_2 \in OH_{DS3}$	103

Table 7.10	Observed Number of errors on BERT and error detection correction activity for DS3 Triple errors when $FBI_1 \in OH_{DS3}$, $FBI_2 \notin OH_{DS3}$	104
Table 7.11	Observed Number of errors on BERT and error detection correction activity for DS3 Triple errors when $FBI_1, FBI_2 \notin OH_{DS3}$	105
Table 7.12	Expected OFD Time Probabilities for DS1	110
Table 7.13	Observed OFD Time Probabilities for DS1 (400 Trials).....	110
Table 7.14	K-S Test results for DS1 Reframe Distributions	116
Table 7.15	K-S Test Results for DS3 Reframe Time Distributions.....	122
Table 7.16	Conventional and FEC Framing Performance for DS1	122
Table 7.17	Conventional and FEC Framing Performance for DS3	123
Table A.1	Next state table for D-flip flop with synchronous load and shift enable	142
Table A.2	Next state table for D-flip flop with synchronous load and clear	144

List of Figures

Figure 2.1:	DS1 Superframe Format	6
Figure 2.2:	DS3 Masterframe Format	7
Figure 2.3:	State machine outline the frame alignment processes	8
Figure 2.4:	DS1-FEC Codeword Format.....	11
Figure 2.5:	DS3-FEC Codeword Format.....	12
Figure 2.6:	Alternative strategies for recovering framing speed (a) Parallel pickethunting, (b) Continuous parallel syndrome computation by vector matrix multiplication.....	15
Figure 3.1:	FEC encoder block diagram.....	18
Figure 3.2:	I/O interface for FEC encoder.....	19
Figure 3.3:	Mapping of DS3 frames and masterframes into DS3-FEC codewords ..	21
Figure 3.4:	Schematic diagrams for encoder synchronization module	21
Figure 3.5:	Output from fb+2 module	22
Figure 3.6:	FEC Encoder Operation Pipeline.....	23
Figure 3.7:	Variable length encoder delay register	23
Figure 3.8:	Linear feedback shift register implementing division by $G(x)_{DS3}$	24
Figure 3.9:	Re-configurable FEC Encoding Register ($+G(x)$)	25
Figure 3.10:	Diagram showing codewords j and j+1 are contiguous.....	27
Figure 3.11:	Schematic diagram of parity register	29
Figure 3.12:	Schematic Diagram of 12-bit holding register.....	31
Figure 3.13:	Demonstration of effect of sampling FEC on at codeword boundary times.....	33
Figure 3.14:	Schematic for FEC encoder output multiplexor	33
Figure 4.1:	DS1/DS3 FEC Decoder block diagram	40
Figure 4.2:	Codeword Delay Buffer Block Diagram	51
Figure 4.3:	CCSC Compensation Term Generation Circuitry	52
Figure 4.4:	CCSC Compensation Function Selector MUX.....	53
Figure 4.5:	CCSC Syndrome Register (Divide by $G(x)$)	54
Figure 4.6:	Received Word Parity Computation Circuit.....	57
Figure 4.7:	Block Diagram of FAU module.....	58
Figure 4.8:	Pseudocode for the FAU System Controller.....	59

Figure 4.9:	Error Correction Unit Block Diagram	62
Figure 4.10:	Meggitt Decoding Register	65
Figure 4.11 (a):	Pseudocode for ECU System Controller Operation in DS1 mode	66
Figure 4.11 (b):	Pseudocode for ECU Operation in DS3 mode	67
Figure 4.12:	Decoder Output Conditioning Module	70
Figure 5.1:	Falsehood rejection process for one bit in FEC-derived framing	76
Figure 5.2:	Combined search and confirmation process for FEC-derived framing ..	77
Figure 5.3:	Signal flow graph for the FEC-derived out-of-frame detection process.....	78
Figure 6.1:	Block Diagram of Laboratory Test Environment	84
Figure 6.2:	Schematic diagram of error injector board	88
Figure 6.3:	Walking Single Bit Error Injection in DS3 mode with $N=2$	89
Figure 6.4:	Theoretical and Measured relationship between SNR and BER	90
Figure 6.5:	Schematic diagram for random error injector module	91
Figure 6.6:	Theoretical and measured BER reduction by FEC for DS3	107
Figure 7.1:	Theoretical and Measured BER reduction by FEC for DS1	108
Figure 7.2:	Measurement intervals for OFD time ($C_0=1$)	109
Figure 7.4 (a):	Max. average reframe time distribution for $C_R=1$ and $BER=0$	111
Figure 7.4 (b):	Max. average reframe time distribution for $C_R=1$ and $BER=10^{-6}$	112
Figure 7.4 (c):	Max. average reframe time distribution for $C_R=1$ and $BER=10^{-4}$	112
Figure 7.4 (d):	Max. average reframe time distribution for $C_R=2$ and $BER=0$	113
Figure 7.4 (e):	Max. average reframe time distribution for $C_R=2$ and $BER=10^{-6}$	113
Figure 7.4 (f):	Max. average reframe time distribution for $C_R=2$ and $BER=10^{-4}$	114
Figure 7.4 (g):	Max. average reframe time distribution for $C_R=4$ and $BER=0$	114
Figure 7.4 (h):	Max. average reframe time distribution for $C_R=4$ and $BER=10^{-6}$	115
Figure 7.5 (a):	Max. average reframe time distribution for $C_R=1$ and $BER=0$	117
Figure 7.5 (b):	Max. average reframe time distribution for $C_R=1$ and $BER=10^{-6}$	117
Figure 7.5 (c):	Max. average reframe time distribution for $C_R=1$ and $BER=10^{-4}$	118
Figure 7.5 (d):	Max. average reframe time distribution for $C_R=2$ and $BER=0$	118
Figure 7.5 (e):	Max. average reframe time distribution for $C_R=2$ and $BER=10^{-6}$	119
Figure 7.5 (f):	Max. average reframe time distribution for $C_R=2$ and $BER=10^{-4}$	119
Figure 7.5 (g):	Max. average reframe time distribution for $C_R=4$ and $BER=0$	120
Figure 7.5 (h):	Max. average reframe time distribution for $C_R=4$ and $BER=10^{-6}$	120

Figure 7.5 (i):	Max. average reframe time distribution for $C_R=4$ and $BER=10^{-4}$	121
Figure 8.1:	Power Dissipation Measurement Setup	124
Figure 8.2:	Measured and Theoretical Power Dissipation in DS3 mode	125
Figure 8.3:	DS1/DS3 FEC ASIC Bonding Diagram	127
Figure 8.4:	High Fan out Clock Routing Strategy	128
Figure 8.5:	Clock Trunk Positioning on the Die	128
Figure 8.6:	Scan Flip Flop Structure	129
Figure 8.7:	Example Three element Scan Chain	130
Figure 8.8:	Functional Test Vector Suite	131
Figure 9.1:	Cyclical indexed RAM implementation of delay registers	136

List of Plates

Plate 6.1:	Laboratory Test Environment Unit.....	85
-------------------	--	-----------

List of Acronyms and Symbols

α -	Level of significance for Kolomorgov-Smirnov test
ASIC -	Application Specific Integrated Circuit
ATE -	Automatic Test Equipment
ATM -	Asynchronous Transfer Mode
BER -	Bit Error Rate
BER_{Channel} -	Bit error rate of a transmission channel
BER_{in} -	Input (uncorrected) bit error rate of a transmission channel
BER_{out} -	Output (error corrected) bit error rate of a transmission channel
BERT -	Bit Error Rate Test set
b_i -	FEC checkbit i in $B(x)$
B-ISDN -	Broadband Integrated Services Digital Network
b_p -	Overall parity bit in the DS3-FEC codeword format
$B(x)$ -	Checkbit polynomial
C-bit -	DS3 conventional stuff indication bits
CCSC -	Continuously Compensated Syndrome Computation
CFD -	Cumulative frequency distribution
CF_i -	Compensation function i
CMOS -	Complementary Metal Oxide Semiconductor
C_O -	Out-of-frame detection confirmation threshold
CODEC -	Coder/Decoder
$C_O\text{-out-of-}k$ -	The x -out-of- k out-of-frame detection scheme
C_R -	Reframe confirmation threshold
CWB -	Codeword boundary (i.e. the frame)
CW_i -	Codeword number i
D -	Kolomogorov-Smirnov Test Statistic
δ -	Normalized duration of a single bit interval (DS1: 1/2316, DS3: 1/1360)
D_{crit} -	Critical value of Kolomogorov-Smirnov Test Statistic

DED -	Double Error Detection
DIP -	Dual-in-line Package
d_j -	Difference term j in $D(x)$
DS0 -	Digital Signal level 0
DS1 -	Digital Signal level 1
DS1-FEC -	FEC encoded DS1 signal format
DS2 -	Digital Signal level 2
DS3 -	Digital Signal Level 3
DS3-FEC -	FEC encoded DS3 signal format
$D(x)$ -	Difference Polynomial
ECU -	Error Correction Unit
FAU -	Frame Alignment Unit
FAW -	Frame Alignment Word
f_b -	Bit Rate
FBI -	Fixed Bit Index
F-bit -	DS1 or DS3 conventional framing bit
FEC -	Forward Error Correction
F_i -	Theoretical cumulative frequency distribution in KS-Test
FIF -	False-in-frame
FOTS -	Fibre Optic Transmission System
FSM -	Finite State Machine
$G(x)$ -	FEC code generator polynomial
$G(x)_{DS1}$ -	Generator polynomial for the DS1-FEC code
$G(x)_{DS3}$ -	Generator polynomial for the DS3-FEC code
H_0 -	Null hypothesis
HCMOS -	High speed CMOS
HOE -	Higher Order Error
INF -	Inframe
$I_{WITH CHIPS}$ -	Current drawn by FEC board with FEC ASICs operating
$I_{WITHOUT CHIPS}$ -	Current drawn by FEC board with FEC ASICs removed
k -	Number of message bits in an (n, k) code (DS1: 2304, DS3: 1348)
KS-TEST -	Kolomogorov-Smirnov Test

l -	Number of bits a code has been shortened by
$L_{\text{DS1-FEC}}$ -	2316; length of the DS1-FEC codeword
$L_{\text{DS1-FEC-DATA}}$ -	2304; number of payload data bits covered by the DS1-FEC codeword
$L_{\text{DS3-FEC}}$ -	1360; length of the DS3-FEC codeword
$L_{\text{DS3-FEC-DATA}}$ -	1344; number of payload data bits covered by the DS3-FEC codeword
L_{Parent} -	Length of the parent code from which the DS1-FEC and DS3-FEC codes were shortened from (DS1: 4095, DS3: 2047)
m -	Number of checkbits in an (n, k) Hamming code ($m = n - k$)
MF -	Mis-frame
MSB -	Most Significant Bit
MUX -	Multiplexer
N -	F-bit spacing
n -	Number of bits in a codeword in an (n, k) code (DS1: 2316, DS3: 1360)
NACK -	Negative Acknowledgment
N_{ch} -	Expected number of errors in a codeword block
$N_{\text{e}}(k_0)$ -	Expected number of errors in a codeword after FEC decoding given that k_0 errors were present prior to decoding
NRZ -	Non Return to Zero
N_S -	Spacing of errored codewords in walking error injection modes in laboratory test environment
OFD -	Out-of-frame Detection
OH-bits -	Overhead (non-payload) bits in a DS1 or DS3 frame
OH_{DS1} -	Set of conventional OH-bit positions in the DS1-FEC codeword
OH_{DS3} -	Set of conventional OH-bit positions in the DS3-FEC codeword
OOF -	Out-of frame
OPBErr -	Overall parity bit in error
[P] -	Parity check matrix
P_d -	Probability of successful (error free) detection of a codeword
P_{DS3} -	
P_{e} -	Probability of a single bit being in error (=BER)
P_{EXT} -	Probability of error extension after decoding a codeword

PGA -	Pin Grid Array Package
PGF -	Probability generating function
P_{HOE} -	Probability of a Higher Order Error being detected
PAR_i -	Parity of the received codeword at time i
p_j -	Exponent for which difference term d_j is associated
P_m -	Probability of random data mimicking a valid DS1-FEC or DS3-FEC codeword ($\approx 2^{-12}$)
P_{OFD}(z) -	Out-of-frame detection time probability generating function
P_{RF}(z) -	Reframe time probability generating function
p_s -	Probability of slipping
P_X(z) -	An arbitrary probability generating function
q_e -	1 - p_e
Q_{EXT} -	1 - P_{EXT}
q_m -	1 - P_m
R⁽¹⁾(x) -	Cyclic shift of the received word R(x)
Rdist_i(x) -	Received word at time i with the checkbits in their distributed positions
Rsys_i(x) -	Received word at time i re-arranged into systematic form
R(x) -	Received word
SEC -	Single Error Correction
SED -	Single Error Detection
S_i -	Measured cumulative frequency distribution
SONET -	Synchronous Optical Network
S(x) -	Syndrome polynomial
t -	Number of trials to obtain the measured cumulative frequency distribution for the KS-Test
TC -	Terminal count values for system counters (DS1: 2315, DS3: 1359)
T_{Search} -	Search time component to the reframe time
τ(z) -	Probability generating function of the reframe search process
u_j -	Message bit j in U(x)
U(x) -	Message polynomial
V_{DD} -	Power supply voltage
VLSI -	Very Large Scale Integration

WBI -

Walking Bit Index

z -

Delay variable

Chapter 1

Introduction

1.1 Motivation for DS1 and DS3 FEC

The DS1 (1.544 Mb/s) and DS3 (44.736 Mb/s) digital multiplex signals are the backbone of the modern digital telecommunications transport network. DS1 and DS3 lines are also leased as private digital carrier facilities for use in customer networks. The raw bit error rates (BER) of such facilities are often in the 10^{-7} to 10^{-9} range [1,2]. While this is low for voice, it may not satisfy the financial data needs of some customers (e.g. banks). One method to guarantee ultra-low BER's of 10^{-12} to 10^{-16} and to protect against anomalous rises in BER on these systems is to implement a forward error correction (FEC) code in the transmitted data. To incorporate an FEC code into a signal format, some additional bits called checkbits are computed and transmitted along with the existing data. These checkbits add redundancy to the data which can be exploited at the receiver to locate and correct bit errors. However, the existing DS1 and DS3 signal formats have no spare overhead in which to transport the additional checkbits of an FEC code. One cannot increase the DS1 or DS3 bit rates to accommodate the addition FEC overhead. Likewise, it is highly impractical to sacrifice payload information for checkbit transportation. This is due to the plesiochronous multiplexing scheme used to construct the DS3 signal format from its DS2 tributaries. In this scheme, no specific DS3 payload position maps directly to a specific position in a DS2 tributary frame, i.e. sometimes a DS3 payload position is a DS2 data bit, sometimes it is a DS2 stuff bit, and sometimes it is a DS2 overhead bit. Thus, selecting a specific payload position at the DS3 level to transport FEC checkbits does not guarantee that only payload will be robbed from the tributary data bit positions. Thus to use payload positions to transport FEC checkbits for DS3 would require complete de-multiplexing down to the DS1 level to identify specific payload positions, re-multiplexing for transport, de-multiplexing for checkbit identification, and re-multiplexing before the errored bit could be located. Obviously, this is impractical. However, it has been shown in [3] that it is possible to incorporate an FEC code into the DS1 and DS3 signal formats where the FEC checkbits are transported in the primary framing (F) bit locations of the DS1 and DS3 frame.

The F-bits contain a fixed pattern at specific locations in each frame. By scanning for this pattern, the receiver is able to align itself to the frames of the incoming data stream and

properly de-multiplex the high rate digital signal into its lower rate components. However, if the bits used for frame alignment are reused to carry the FEC checkbits, then an alternative scheme must be used to achieve frame alignment. The alternative scheme considered in [3] is to solve the frame alignment problem in conjunction with the FEC decoding. To meet existing DS1 and DS3 re-frame times, a new method for continuous syndrome computation in the decoder is necessary, because the FEC codewords are so much longer than the existing framing bit interval.

1.2 Issues Addressed in this Thesis

Previously, the work in [3] has shown that FEC codes can be incorporated into the DS1 and DS3 signal formats by reusing the F-bits to transport the checkbits of the code. In [3] a conventional framing system was analyzed for use as a benchmark to compare to FEC-derived framing. The performance of the FEC derived framing strategy was also analyzed. As the reframe speeds were determined to be too long the CCSC syndrome method was developed to recover the framing speed. Specific calculation of the CCSC compensation functions for DS1-FEC and DS3-FEC were presented. Framing performance using this decoding strategy was analyzed. However, [3] focuses on the theory and simulation of a DS1-FEC or DS3-FEC system. In the present work we have achieved an actual VLSI system implementation and obtained measured laboratory performance results for both error correction and framing performance.

The focus of this thesis is the design, development, testing, and validation of a 38k gate HCMOS VLSI ASIC¹ which has correctly implemented the novel strategy of [3] for combined framing and forward error correction (FEC) coding of the DS1 and DS3 signal formats. The main issues addressed are the following:

- **FEC Encoder Circuit:** FEC encoder design and circuit implementation to implement FEC checkbit computation and F-bit replacement.
- **FEC Decoder Circuit:** FEC decoder design and circuit implementation to implement: continuously compensated syndrome computation, frame alignment functions, error detection, location, and correction, and conventional frame reconstruction.
- **Re-configurable Architecture:** Development of a re-configurable architecture which efficiently re-uses internal encoder and decoder structures so that the ASIC implementation is compatible for both DS1 or DS3 modes of operation.
- **Logic Minimization:** Presentation of the gate count saving by using logic minimization software for the minimization of boolean logic functions which would be intractable using traditional Karnaugh map techniques.
- **ASIC Documentation:** Specific details for ASIC implementation in the LSI Logic

1. LSI Logic Corporation, Inc., LCA100K 1.0-Micron Gate Array Technology

design environment such as power consumption, package and die selection, clock routing.

- **Testability:** Design for testability and the ASIC testing strategy both using Automatic Test Equipment (ATE) and laboratory level testing.
- **Performance:** System level design validation and performance characterization for comparison with theoretically predicted performance in terms of (i) framing performance, (ii) error detection/correction capabilities, and (iii) BER improvement in completely random error environments.

1.3 Other Related Work

In other related work [1], a (224, 216) FEC code was applied to the DS3 tributary of a proprietary 565 Mb/s fibre optic transmission system (FOTS) and shown to be quite effective in combatting dispersion related BER floors. The checkbits for that FEC code were transported in available space in the proprietary format of the high speed optical signal. One of the most significant aspects of the ASIC implementation now reported is that the checkbits are completely contained *within the DS3 (DS1) signal itself* and hence error protection of the DS3 (DS1) can be provided end-to-end across a network regardless of the medium or system by which it is transmitted.

1.4 Applications of FEC for DS1 and DS3

The FEC ASIC has a number of applications. FEC significantly reduces the power independent error rate floors which exist in many FOTS [4]. In [2] it was determined by simulation of the present DS3 code that all bit errors in a 22 day-sample of errored-second data from a DS3 tributary of a 565 Mb/s FOTS would be correctable. This included multiply-errored seconds where the raw BER rose to about 10^{-7} . In addition, when applied to each tributary of an $n \times$ DS3 lightwave system, an n -bit burst error correcting capability arises at the optical level.

High speed packet protocols such as ATM require that the BER of the networks on which they operate be very low so as to avoid the implications of error recovery by retransmission. At 10^{-7} BER, DS3-borne ATM applications will require a packet retransmission every 223 ms on average, perhaps seriously affecting throughput, especially when propagation delays are high. Assuming a 50 ms NACK delay, another 2.2 Mbits will be outstanding before the transmitter realizes that a packet must be resent. The detailed effects of such frequent retransmission on the variety of applications planned for ATM are unknown but could be quite detrimental in some applications. This is particularly true for ATM as opposed to X.25 type protocols because error recognition and recovery are left to the end-applications, unlike prior data protocols where transport integrity was managed on a link-

by-link basis. In sum, very low physical layer error rates may be important or at least advantageous to for the success of high speed lightweight protocols, such as are the basis of B-ISDN.

FEC also gives an advanced performance monitoring mechanism which allows detection of facility degradation while the customer is still receiving a low (corrected) error rate [1]. Maintenance of the degraded equipment may conceivably take place before the customer is even aware of the degradation because, as will be shown, at a raw BER of 10^{-6} , the customer will still observe $BER \approx 1.4 \times 10^{-9}$.

1.5 Outline of the Thesis

The structure of this thesis is as follows:

Chapter 2 presents the necessary background. This includes (i) a description of the DS1 and DS3 signal formats, (ii) the conventional process for frame alignment using the F-bits, (iii) Hamming codes for FEC, (iv) the development of FEC codes specifically for the DS1 and DS3 signals, and (v) the mechanism by which frame alignment is achieved on the FEC encoded DS1 and DS3's.

Chapter 3 presents the design of the FEC encoder. This circuit computes the checkbits for each FEC codeword and inserts them into the appropriate locations in the frames. The functions of each module which make up the encoder are presented in detail. The manner in which internal circuits of the encoder are re-configured for DS1 or DS3 compatibility is presented for each module.

Chapter 4 discusses the FEC decoder design. The decoder is responsible for frame alignment to the incoming data stream and for detecting and correcting errors in the FEC encoder data blocks. The modules which make up the decoder to achieve these functions are presented in detail. The elements of the design to implement re-configurable architecture are highlighted in the discussions.

Chapter 5 presents the theoretical framing and error correction performance of the FEC ASIC for both DS1 and DS3 modes of operation. The probability generating function and state transition diagram methods of [5] are used to estimate the re-frame time and out-of-frame detection time timing parameters for FEC framing. These are used later for comparison and validation of actual re-frame and out-of-frame detection times measured in the laboratory.

Chapter 6 outlines the main components of the laboratory test environment and describes the functions of each.

Chapter 7 presents the results of the laboratory testing and design validation of the FEC ASIC. The experiments consisted of three classes: (i) error detection/correction performance when subjected to controlled errors, (ii) error detection/correction performance when subjected to completely random errors, and (iii) re-frame and out-of-frame detection time measurements. These results are compared to the theoretical predictions derived in Chapter 5 for both DS1 and DS3.

Chapter 8 covers other issues of the ASIC design which are not specific to either the encoder or decoder.

Chapter 9 provides a summary of the work to develop the FEC ASIC. It also summarizes the main performance results of the FEC ASIC. The chapter concludes with recommendations for future work. Included in this section are recommended modifications to the design which may be desirable before going to production.

Chapter 2

Background

The chapter presents the relevant background for the concepts involved in developing the DS1/DS3 FEC CODEC. These include: the basic DS1/DS3 signals and formats, conventional frame alignment procedures, Hamming codes for FEC, the structures of the DS1-FEC/DS3-FEC codewords, the principle for FEC derived framing, strategies considered to reduce reframe time in FEC framing, and the goal of re-configurable design for DS1 and DS3 compatible architecture.

2.1 Conventional DS1 and DS3 Signal Formats

2.1.1 DS1 Superframe Format

The DS1 is the primary multiplex signal in the North American digital signal hierarchy. It is a 1.544 Mb/s signal which is a byte interleaved multiplex of 24 DS0 (64 kb/s) channels. Each DS1 frame is 193 bits in length, consisting of 1 framing (F) bit, and 192 information bits [6]. Twelve DS1 frames are grouped together to form one DS1 superframe. The structure of the DS1 superframe is shown in Figure 2.1:

$$F_1 [192] F_2 [192] F_3 [192] \dots F_{12} [192]$$

Figure 2.1: DS1 Superframe Format

where F_1, F_2, \dots, F_{12} , are the 12 framing bits used for frame alignment at the receiver, and [192] is the payload data for one DS1 frame. The F-bits carry a 100011011100 pattern, used for frame and multi-frame alignment (Section 2.2) at the receiver to allow proper demultiplexing of the 24 DS0's.

2.1.2 DS3 Masterframe Format

The DS3 signal is a 44.736 Mb/s signal which is a pulse stuffed multiplex of 7 plesiochronous DS2 signals, each of which carries 4 DS1's (equivalent to 96 DS0 channels). The DS3 transports 672 DS0 channels or its equivalent in total, and is increasing in use as a leased line for ATM and frame relay wide area networks. The structure of the DS3 master frame is shown in Figure 2.2 [7]:

X	[84]	F ₁	[84]	C ₁₁	[84]	F ₀	[84]	C ₁₂	[84]	F ₀	[84]	C ₁₃	[84]	F ₁	[84]
X	[84]	F ₁	[84]	C ₂₁	[84]	F ₀	[84]	C ₂₂	[84]	F ₀	[84]	C ₂₃	[84]	F ₁	[84]
P	[84]	F ₁	[84]	C ₃₁	[84]	F ₀	[84]	C ₃₂	[84]	F ₀	[84]	C ₃₃	[84]	F ₁	[84]
P	[84]	F ₁	[84]	C ₄₁	[84]	F ₀	[84]	C ₄₂	[84]	F ₀	[84]	C ₄₃	[84]	F ₁	[84]
M ₀	[84]	F ₁	[84]	C ₅₁	[84]	F ₀	[84]	C ₅₂	[84]	F ₀	[84]	C ₅₃	[84]	F ₁	[84]
M ₁	[84]	F ₁	[84]	C ₆₁	[84]	F ₀	[84]	C ₆₂	[84]	F ₀	[84]	C ₆₃	[84]	F ₁	[84]
M ₀	[84]	F ₁	[84]	C ₇₁	[84]	F ₀	[84]	C ₇₂	[84]	F ₀	[84]	C ₇₃	[84]	F ₁	[84]

Figure 2.2: DS3 Masterframe Format

where [84] is 84 bits of payload information from the DS2 tributaries. The remaining bits are the overhead (OH) bits of the DS3 masterframe. Each line in Figure 2.2 is a basic DS3 frame. Frame alignment (Section 2.2) to the basic DS3 frame is achieved by searching for the periodic 1-0-0-1 pattern carried in the primary framing (F) bits. The DS3 masterframe is composed of seven frames as shown in Figure 2.2. The i^{th} frame in the masterframe carries the stuff control indications (C-bits) for the i^{th} DS2 tributary. Masterframe alignment is achieved using the X-X-P-P-M₀-M₁-M₀ pattern at the first bit position in each DS3 frame of the masterframe. The M₀ and M₁ carry fixed 0 and 1 values. The P bit is the parity over the information portion of the preceding master frame. The X-bit is a signalling bit which may be used as an alarm service channel. Both of these bits are duplicated for integrity.

The DS2 tributary rate is nominally 6.312 Mbit/s but the actual frequency may vary by +/- 30 ppm. To accommodate for these deviations, a pulse stuffing multiplex technique is used. In this technique, the multiplexer is run at a rate slightly higher than the sum of the maximum expected tributary rates. Provision is made for the occasional addition of a stuff bit (which carries no information) to the multiplexed data stream to adapt each individual tributary rate [8]. The addition of a stuff bit can take place only once per masterframe for each DS2 tributary. The presence of a stuff bit for tributary i is indicated via the stuffing control bits C₁₁-C₁₂-C₁₃. If a stuff bit is present, then each of these bits is set to 1, otherwise they are set to 0. The C-bits are triplicated for single error protection and "2 of 3 majority" decoding is used to determine the presence of a stuff bit.

2.2 Frame Alignment in a Digital Multiplex System

To properly de-multiplex the high rate digital multiplex signal into its low rate components, the receiver must be aligned to the frame boundary of the incoming digital signal. The process by which this is achieved is known as frame alignment. Frame alignment is an essential process in any digital carrier system. The information presented in this section is largely derived from [3] and [5]. The finite state machine shown in Figure 2.3 outlines the

frame alignment process.

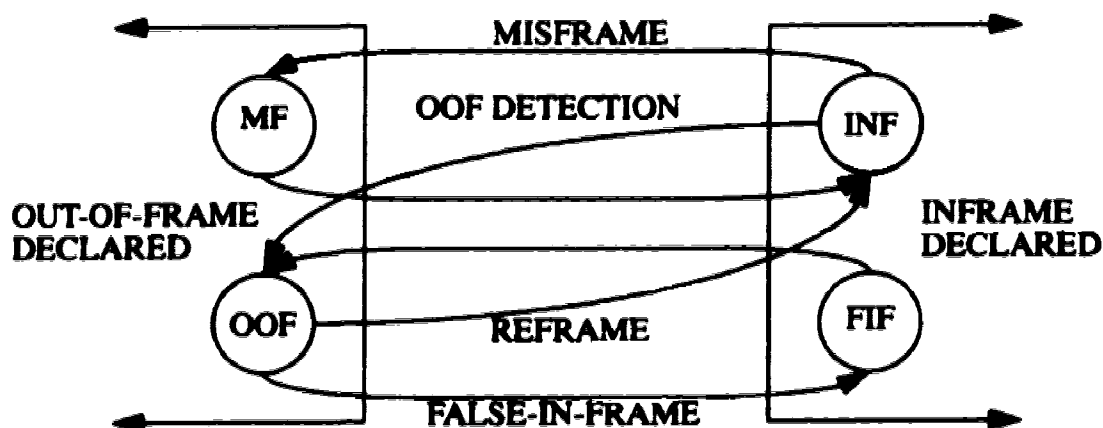


Figure 2.3: State machine outline the frame alignment processes

The following are the four states of the diagram:

- INF (In-frame):** The framing circuit is aligned to the correct frame boundary and believes this to be the case
- MF (Mis-frame):** The framing circuit is aligned to the correct frame boundary but does not believe that this is so
- OOF (Out-of-frame):** The framing circuit is not aligned to the frame boundary and believes this is the case
- FIF (False-in-frame):** The framing circuit is not aligned to the frame boundary but falsely that it is aligned

There are four relevant transitions in the state diagram, reframe (RF), out-of-frame detection (OFD), misframe (MF), and false-in-frame (FIF).

2.2.1 Conventional Reframing

Reframing is required to effect the transition from the OOF state to the INF state in Figure 2.3. The reframing procedure consists of two phases, search and confirmation. In DS1 and DS3 signals, frame alignment is conventionally achieved via serial search for the distributed frame alignment pattern carried at the frame alignment bit (F-bit) positions in a frame. The receiver slips through the data stream evaluating candidate positions until it finds the expected framing pattern in one of these positions. This location is provisionally held as a frame candidate. A confirmation process then verifies that the framing pattern persists in C_R subsequent frames, where C_R is the confirmation threshold. If so then frame alignment is declared, otherwise the search process resumes. For DS1 and DS3 the value of C_R in the c-successive-count is typically 12 to 20. The relatively high confirmation

level serves to prevent the framer from locking onto a position where random data mimics the expected framing pattern, i.e FIF.

2.2.2 Maximum Average Reframe Time

The reframe time is the amount of time it takes for the framer to re-establish the INF state from the instant that OOF is declared. In the worst case, the framing circuitry must begin searching forward for the correct frame boundary one bit after the true frame alignment position. This is the maximal length search case. In this case, $n-1$ positions must be rejected as the frame boundary before the framer arrives at the true frame aligned position. If bit errors corrupt the F-bits during the confirmation phase they will implicitly force a subsequent search by failure to verify. The average time for reframe given that the framer is initially facing a maximal length search including the effects of errors is the called maximum average reframe time.

2.2.3 False-in-Frame Time

It is possible for random data to mimic the frame alignment pattern carried by the F-bits. If the framer encounters a mimic of the frame alignment pattern during the search phase, it will dwell on that position as a frame candidate. If the mimic persists at subsequent frame intervals more times than the reframe confirmation threshold, C_R , then the framing circuit will believe it has located the frame boundary and falsely declare that it is in-frame. The average time to such a declaration is known as the false-in-frame (FIF) time. However, C_R is typically chosen large enough so that FIF's are made extremely rare while still keeping the maximum average reframe time as short as possible.

2.2.4 Conventional Out-of-Frame Detection

Out-of-frame detection is the process for determining if frame alignment has been lost once the system is in-frame (INF). When the system is inframe, the framer continually checks for the expected frame alignment pattern in the F-bit positions. There are two common strategies for OOF detection. In the first, called the C_O -out-of-k scheme, frame loss is declared when a fraction of consecutive F-bits are in error, such as 3-out-of-5 or 4-out-of-7. In the second strategy, called the C_O successive count scheme, frame loss is declared when C_O successive F-bits are in all error. The former scheme is used for OOF detection in conventional DS1 and DS3 framing systems.

2.2.5 Out-of-Frame Detection Time

The OOF detection time is the average amount of time it takes the framer to detect an OOF condition after it has actually occurred. The OOF detection time depends on the

value of C_O . It also depends on the probability that random data at the misaligned position which the framer believes to be the frame boundary mimics the valid framing pattern. This makes it appear to the framer that it is still inframe thus delaying the OOF declaration.

2.2.6 Misframe Time

Multiple corruptions of the F-bits due to line errors in transmission can make it appear to the framer that it has lost frame and declare OOF even when it still aligned to the correct frame boundary. The average time between such declarations for a given BER is the mis-frame time. The C_O value is chosen to give reasonably fast OOF detection times while making mis-frames extremely infrequent.

2.3 Hamming Codes for Forward Error Correction

To add error correction and detection capabilities to the DS1 and DS3 signals, Hamming codes are used. Hamming codes were selected because (i) very high rate codes are required for the DS1 and DS3 formats (0.9948 DS1; 0.9918 DS3) and (ii) the encoding and decoding procedures for cyclic Hamming codes are easily realized using linear feedback shift registers which implement polynomial division [9].

A perfect (n, k) Hamming code satisfies the following relation:

$$(n, k) = (2^m - 1, 2^m - m - 1) \quad (2.1)$$

where k is the number of message bits (i.e. the number of information bits) and n is the total number of bits in either FEC encoded codeword block. Thus there are $n-k=m$ check-bits in each codeword. The checkbits add redundancy to the message information which can be exploited at the receiver to detect and correct errors. Hamming codes have single error correcting capabilities.

The checkbits for an $(n-k)$ Hamming code are computed as follows:

- (i) Construct the message polynomial $U(x)$ where the message bits are the coefficients of the terms in the polynomial and the most significant message bit is the coefficient of the highest order term in $U(x)$.
- (ii) Multiply $U(x)$ by x^{n-k} (Pad out zeros to create positions for the checkbits).
- (iii) Compute the remainder, $B(x)$, of the polynomial division of $x^{n-k} U(x)$ by the irreducible generator polynomial for the code, $G(x)$. The coefficients of $B(x)$ are the checkbits for the FEC codeword.

The systematic form [9] of a Hamming code codeword is:

$$x^{n-k} U(x) + B(x) \quad (2.2)$$

i.e. all message bits followed by all checkbits in order. During transmission, the Hamming code codeword in (2.2) may be subjected to errors. At the receiver, the received word (potentially corrupted by errors) is denoted as $R(x)$. The process of decoding a Hamming code is to compute the remainder of the polynomial division of $R(x)$ by $G(x)$. This result is called the syndrome for the received word. If the syndrome is zero (all bits of the syndrome are zeros) then the received word is error free. If the syndrome is non-zero (the syndrome contains some non-zero bits), the codeword contains an error and the value of the syndrome can be used to determine the location of the error (if a single bit error has occurred).

As noted previously, Hamming codes have single error correcting capabilities. Double error detection (DED) capabilities can be added to the code with the addition of a single overall parity bit. Such an $(n+1, k)$ code is called an extended Hamming code.

When a perfect Hamming code of length given in (2.1) does not apply for a particular system design, it is possible to shorten a code to meet the design requirements. A Hamming code can be shortened by a number of bits, l , to create an $(n-l, k-l)$ shortened Hamming code by using only those codewords in the full length parent code where the l most significant bits are equal to zero. In practice, the l leading zeros are not transmitted as they carry no information. The encoding and decoding of an $(n-l, k-l)$ shortened Hamming code can be performed using the same encoding and decoding circuit structures as for the parent code because these l leading bits do not affect the checkbit and syndrome computations. The circuits for error location require some slight but simple modifications to work with shortened codes

2.3.1 DS1-FEC Hamming Code and Codeword Structure

The DS1-FEC code developed here is a (2316, 2304) shortened Hamming code obtained from the (4095, 4083) full length parent code. The 12 checkbits are transported in the 12 F-bit positions of the DS1 superframe. It can easily be checked that this is the smallest number of DS1 frames over which a code is feasible given only one checkbit every 192 message bits. The structure of each DS1-FEC codeword is shown in Figure 2.4:

$$b_{11} [192] b_{10} [192] b_9 [192] \dots b_0 [192]$$

Figure 2.4: DS1-FEC Codeword Format

where b_0, b_1, \dots, b_{11} are the 12 DS1-FEC Hamming code checkbits (b_{11} is the most significant checkbit) and [192] is the payload of one DS1 frame. Note that the dimensions of

this code exactly match the DS1 SF format superframe of 12 frames. The code will be the basis for both frame and superframe alignment for the DS1 design. Note that this codeword is not in systematic form. The following irreducible, primitive generator polynomial was selected from [9] for the DS1-FEC code:

$$G(x)_{DS1} = x^{12} + x^6 + x^4 + x + 1 \quad (2.3)$$

2.3.2 DS3-FEC Hamming Code and Codeword Structure

The DS3-FEC code developed here is a (1360, 1348) shortened and extended Hamming code, shortened from the (2047, 2036) full length parent code and extended for double error detection (DED). A codeword covers exactly two DS3 frames and uses all F-bits and two of the three existing C-bits in each frame. In the conventional DS3 format, the C-bits are triplicated to protect against single bit errors, in the stuff indication mechanism (Section 2.1.2). With the FEC code, triplication is unnecessary so 2 of the 3 C-bits in each frame are re-used in the code design. The remaining C-bit in each frame continues to convey stuff control indications. The structure of the DS3-FEC codeword is shown in Figure 2.5

$$\begin{array}{cccccccccccccccc} V_i & [84]b_{10} & [84]C_i & [84]b_9 & [84]b_8 & [84]b_7 & [84]b_6 & [84]b_5 & [84] \\ V_{i+1} & [84]b_4 & [84]C_{i+1} & [84]b_3 & [84]b_2 & [84]b_1 & [84]b_0 & [84]b_p & [84] \end{array}$$

Figure 2.5: DS3-FEC Codeword Format

where b_0, b_1, \dots, b_{10} (MSB= b_{10}) are the 11 checkbits of the Hamming code, b_p is the overall parity bit to provide DED, [84] is 84 bits of payload., and $V_i \in \{X, P, M_0, M_1\}$ is a bit from the existing 7-frame DS3 masterframe overhead sequence. The DS3-FEC codewords continually walk through the DS3 masterframe because the 2 frame codeword does not mesh with the 7-frame masterframe. The V-bits are therefore samples of the DS3 masterframe alignment sequence. Accordingly, the DS3-FEC codeword will be the basis for frame alignment only. Masterframe alignment will continue to be based on the existing masterframe sequence (X-X-P-P-0-1-0). Note that this codeword is also not in systematic form. The following irreducible, primitive generator polynomial was selected from [9] for the DS3-FEC code:

$$G(x)_{DS3} = x^{11} + x^2 + 1 \quad (2.4)$$

2.4 The Principle for FEC-Derived Framing

As discussed in Section 2.3, when a codeword is decoded at the FEC decoder, a syndrome is produced which indicates the presence or absence of errors in the codeword. In the

absence of channel errors, a zero syndrome will be computed only if the decoder is aligned to a valid codeword or if it is aligned to a position where the data mimics a valid codeword. However, the probability of codeword mimic in random data has been shown (theoretically and verified by simulation) to be 2^{-12} for both the DS1-FEC and DS3-FEC codes [10]. This is negligibly low and means that a 99.9756% density of non-zero syndromes is expected whenever the decoder is truly out-of-frame. This density of non-zero syndromes is therefore used as a feedback signal to drive the frame alignment process. The FEC-derived reframe and out-of-frame detection processes are presented below.

2.4.1 FEC-based Re-frame Procedure

The FEC-based reframe procedure, like the conventional process, consists of a search phase and a confirmation phase. During the search phase, however, the FEC decoder *slides* through the data stream (unlike conventional search which advances at most one bit per frame), hunting for a bit position at which a zero syndrome is seen, i.e. a valid codeword or a mimic pattern. When such a position is found, it is held provisionally as a candidate for the codeword boundary (i.e. the frame). The confirmation process verifies that the zero syndrome persists at this position in C_R subsequent codewords. If this is the case then the decoder declares INF. If at any time during the confirmation phase a non-zero syndrome is decoded at the candidate position, the candidate is rejected and the search process resumes. Again, C_R is selected so as to make the probability of a false-in-frame very low.

2.4.2 FEC-based Out-of-Frame Detection Procedure

When the decoder is inframe, the syndrome is examined at the assumed codeword boundary time. As long as the syndrome is zero, then the decoder continues to believe that it is inframe. However, if a non-zero syndrome is decoded at the assumed codeword boundary, the decoder then checks that the non-zero syndrome persists in C_O subsequent codeword end times. If so, OOF is declared and the reframe process is initiated. If a zero syndrome is seen at the codeword boundary before C_O additional non-zero syndromes are seen, then the OOF detection process is reset and the system remains inframe. A C_O -successive OOF detection scheme is used for FEC framing instead of the C_O -out-of-k scheme used in conventional DS1 and DS3 framing for two reasons: (i) it is a stricter OOF detection criterion which gives longer misframe times for the large FEC codeword blocks, and (ii) because the probability of codeword mimic is so low (2^{-12}) the chance of seeing C_O -successive non-zero syndromes is very high when actually out-of-frame. [3]

2.4.3 Initial Comparison of Conventional and FEC Reframing Speeds

The reason why it was not possible to simply adapt the serial search strategy of conventional DS3 framing to the FEC case is presented in this section. In conventional reframing, the serial search time dominates over the time to detect loss of frame and re-confirm the inframe state. It is easily shown that for a serial search on a distributed framing sequence, the average time for the maximum length search is:

$$T_{\text{Search}} = \frac{N(N-1)}{p_s \cdot f_b} \approx \frac{N^2}{p_s \cdot f_b} \quad (2.5)$$

where N is the spacing of the F-bits, f_b is the bit rate, and $p_s=0.5$ is the slipping probability per dwell on a candidate position. For DS3, $N=170$, $f_b=44.736$ Mb/s, and hence $T_{\text{Search}} = 1.29$ ms.

The FEC framing strategy must meet or exceed the framing performance of the conventional system. However, a conventional block syndrome computation process requires a complete codeword time (i.e. n bit times) to compute the syndrome for one received word¹. This makes the search portion of the reframe process very long. For a maximal length search, $n-1$ positions must be rejected as the codeword boundary before the true frame boundary is reached. At each misaligned position, it requires one full codeword time to determine if the position decodes a non-zero syndrome to determine if a slip should be generated. Thus the search process using conventional syndrome computation techniques is also $O(n^2)$ as in (2.5). Using the DS3-FEC case as an example, the maximal length search time for the FEC framing strategy using a conventional syndrome decoding process is given by (2.5) with $N=1360$, $p_s=1 \cdot 2^{-12}$, and $f_b=44.726$ Mb/s. This yields $T_{\text{Search}} = 41.3$ ms. Obviously this strategy fails to meet the conventional DS3 framing speed requirements by a wide margin (32x). It is also far above the upper limit allowed for a complete DS3 reframe of 2 ms [6].

Some interesting alternatives were initially considered to regain the framing speed. These included: (i) using 32 separate search (syndrome computation) hunting circuits, each allocated a sub-range as shown in Figure 2.6 (a), and (ii) continual bit-by-bit parallel computation of the received vector syndrome by vector matrix multiplication decoding of the FEC code, shown in Figure 2.6 (b)

1. When out-of-frame, the syndrome is computed for a received word at an assumed codeword boundary. The received word is not technically an FEC codeword unless the decoder is aligned to the true codeword boundaries. The received word will be thus only be referred to as a "codeword" when the decoder is frame aligned.

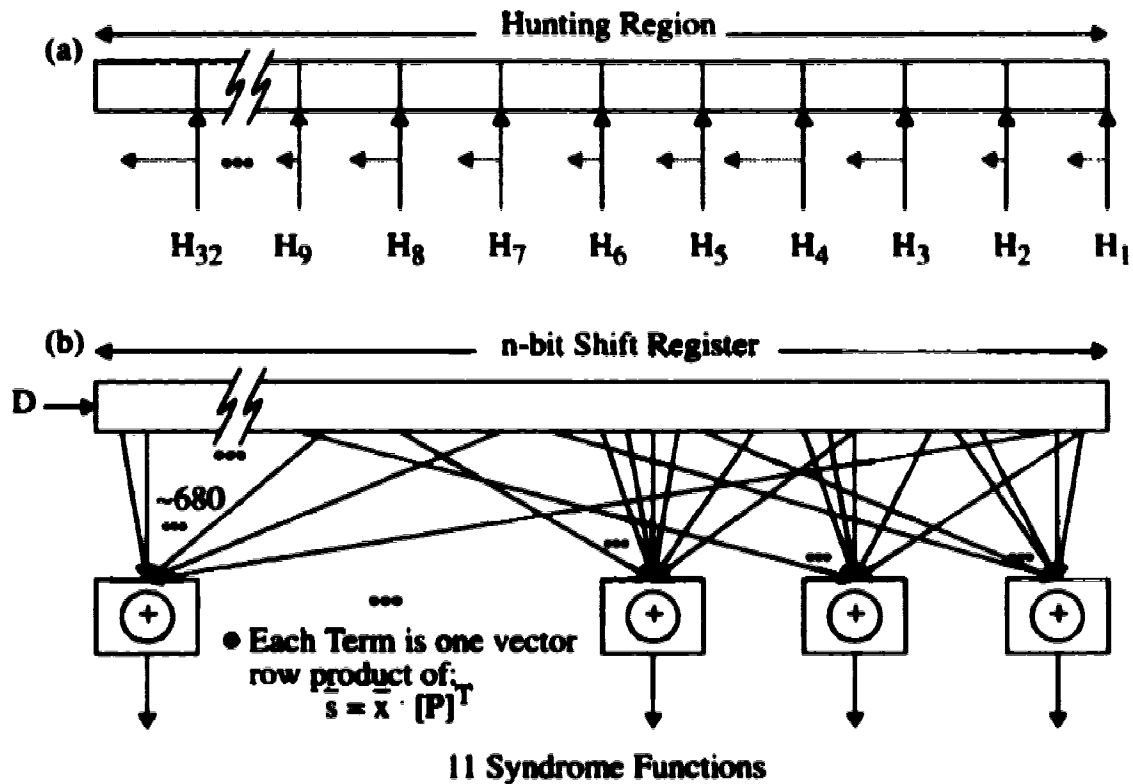


Figure 2.6: Alternative strategies for recovering framing speed (a) Parallel picket hunting, (b) Continuous parallel syndrome computation by vector matrix multiplication

The former strategy led to extremely complex system control problems. The latter was feasible to design (and was verified by computer simulation to frame as the scheme eventually developed) but would have required (using DS3 as the example) eleven 680-input (average term) exclusive-or gates to implement

$$\mathbf{s} = \mathbf{x} \cdot [\mathbf{P}]^T \quad (2.6)$$

where \mathbf{s} is the syndrome, \mathbf{x} is the received word, $[\mathbf{P}]$ is the 11×1360 parity check matrix representation of the FEC code, and "T" denotes matrix transposition. The estimated gate count in the LSI Logic design environment for the eleven ~680-input XOR gates alone is around 24k gates. The benefit of the second approach, however, is that a zero syndrome is seen (neglecting line errors) immediately as the boundaries of \mathbf{x} slide onto the true codeword because the value of \mathbf{s} is computed at all bit times for the preceding set of 1360 bits, not just at codeword end times. Thus bit-by-bit syndrome calculation allows bit-by-bit sliding detection of frame boundaries.

The solution which was finally found and implemented in the DS1/DS3 FEC ASIC gives

the advantage of the parity check matrix approach (single bit time syndrome computation) but without the huge gate count. It is based on continual compensation of the intermediate syndrome result in continual polynomial shift register division for decoding of cyclic codes. The method, which is the key to achieving fast framing with moderate circuit complexity, is called Continuously Compensated Syndrome Computation (CCSC). It is presented in Chapter 4.

2.5 Re-configurable Circuit Architecture

One of the goals of the FEC ASIC design is the development of a circuit architecture which is compatible for DS1 or DS3 operation. Rather than creating DS1 and DS3 versions of the same modules, each module in the design can be re-configured. With the addition of a small amount of logic, this strategy allows for extensive re-use of module components thereby reducing the total size of the design. The mode of operation will be pin-selectable. Each module in the FEC encoder and FEC decoder is presented in detail in Chapters 3 and 4. Along with the functional description, the control logic to re-configure each module for DS1 or DS3 mode will be described. This will show how each circuit is logically altered for the different modes of operation.

Chapter 3

The DS1/DS3 FEC Encoder

In this chapter, the design of the DS1/DS3 FEC encoder is presented. The FEC encoder is responsible for two main functions: (i) computation of the checkbits (and overall parity bit in DS3 mode) for a block of unencoded DS1 or DS3 data, and (ii) insertion of the checkbits into the appropriate distributed positions for construction the actual DS1-FEC or DS3-FEC codewords.

3.1 FEC Encoder - Top Level

The top level functional diagram for the DS1/DS3 FEC encoder is shown in Figure 3.1. The inputs to the FEC encoder are the following:

- **RESET:** asynchronous reset; When asserted (active low), all counters, registers, and finite state machines (FSM's) are brought into a known state
- **DATA IN:** serial NRZ data input line
- **CLK IN:** Bit rate clock; frequency is 1.544 Mb/s for DS1, 44.735 Mb/s for DS3
- **FRAME IN:** conventional framing information for synchronizing the encoder to the incoming data stream. The signal is a single pulse, one bit time in width, coincident with the first bit (bit 0) of each DS1 superframe (in DS1 mode), or the first bit of either each DS3 frame or masterframe in DS3 mode.
- **DS3:** configures the internal encoder structures to handle either the DS1 or DS3 signal and implement the corresponding FEC code. DS3 mode is obtained by setting this pin high; DS1 mode is obtained by setting this pin low.
- **FEC ENABLE:** enables or disables replacement of F-bits (and selected C-bits in DS3 mode) of an unencoded data block with the computed checkbits. When de-asserted, the process of computing the checkbits for an unencoded data block continues but the checkbits are not inserted into the outgoing data stream. The *FEC_ENABLE* line is sampled at the time that the first bit of a new codeword emerges from the encoder. The sampled value is held for the duration of the codeword. Thus only complete codewords or complete unencoded data blocks appear at the encoder output in the event that the user changes FEC operating status during use.

and the following are the output signals from the FEC encoder:

- **DATA OUT:** FEC encoder serial NRZ data out. If FEC encoding is enabled, this line carries DS1-FEC or DS3-FEC encoded data. If FEC encoding is turned off, the input data stream applied to the encoder data input emerges unchanged with the OH-bit sequence intact.

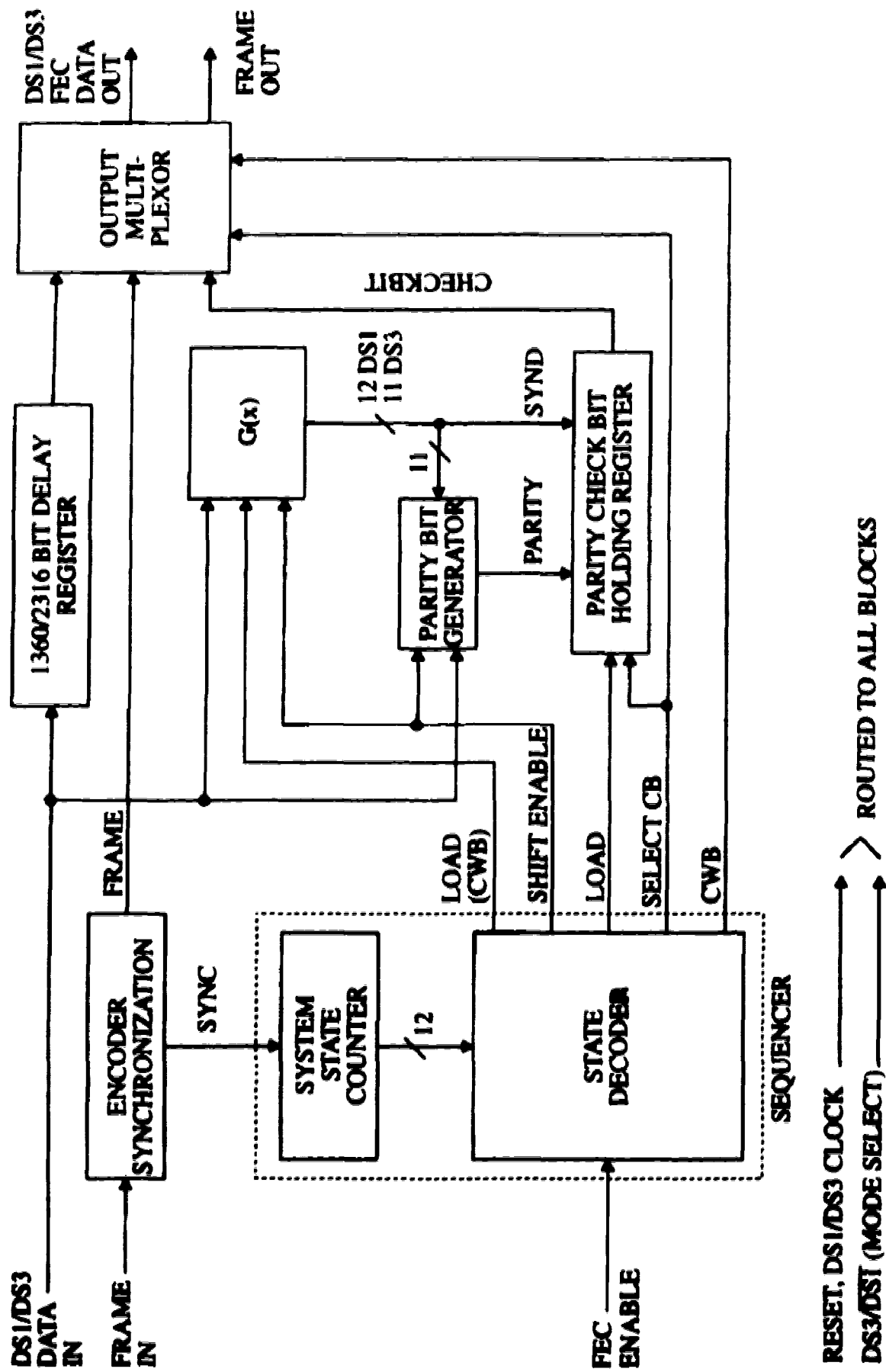


Figure 3.1: FEC encoder block diagram

- **FRAME OUT:** with FEC encoding turned on, this line carries a signal which marks the first bit of each codeword emerging from the encoder by going low for one bit time coincident with this bit. This signal is not for transport in the network but is used only for synchronizing the external test equipment for functional verification of the ASIC (Chapter 6).
- **CLK OUT:** (not shown) inverted version of the input system clock so that the rising edge is in the middle of each data bit on the FEC encoder **DATA OUT** line.

3.2 I/O Synchronization

The standard, recommended way to synchronize external input signals to the edges of the internal system clock is to use D-flip flops to re-time them onto the chip. This method is used for the inputs to the FEC encoder shown in Figure 3.2 .

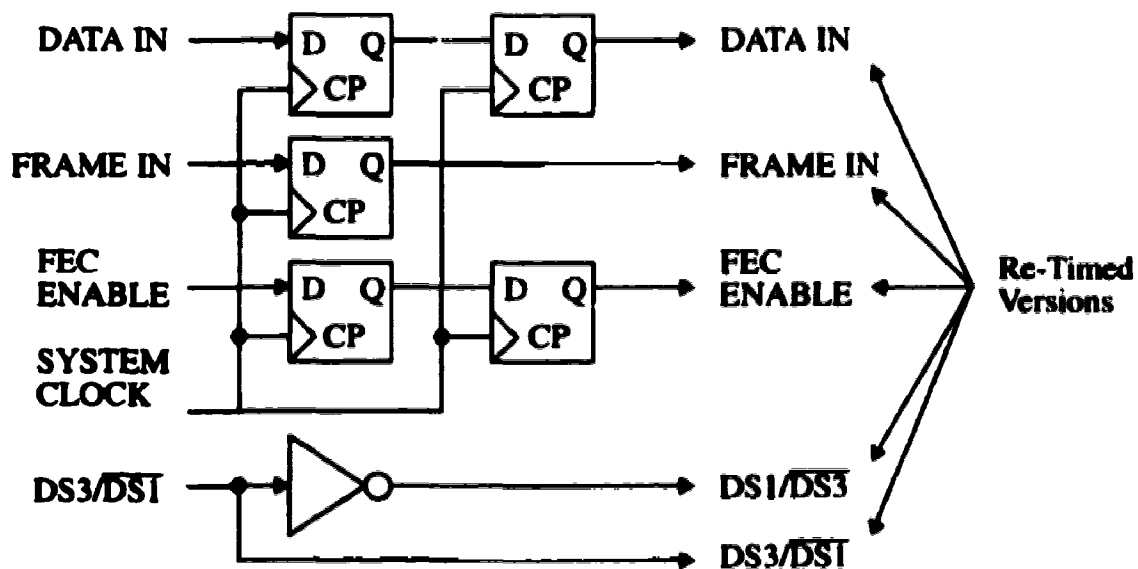


Figure 3.2: I/O interface for FEC encoder

The **DATA IN** and **FRAME IN** lines are re-timed once so that a full clock cycle is available for processing the data on each line. The **DATA IN** line is re-timed a second time so that it coincides with the synchronization signal from the encoder synchronization module. **FEC ENABLE** is an asynchronous input, i.e. it can change at any time. To eliminate possible metastable conditions, the signal is re-timed once to synchronize its changes to the internal system clock. LSI Logic design guidelines require a second re-time on asynchronous to eliminate possible metastable conditions which may arise from the changing of **FEC ENABLE** too close to the rising clock edge at the first re-time flip flop.

The **DATA OUT** and **FRAME OUT** output signals are re-timed once with D-flip flops to remove decoding glitches before they are sent off chip.

The $DS3/\overline{DS1}$ mode select line is not re-timed as its value is not expected to change during operation. Both the $DS3/\overline{DS1}$ signal and its complement are required at various points in the FEC encoder for re-configuring various structures. As shown in Figure 3.2, the complement of the signal ($DS1/\overline{DS3}$) is derived by inverting the $DS3/\overline{DS1}$ signal immediately after it is brought onto the chip. Both signals are then routed to the various places that they are needed. This uses more routing tracks in the die, but is preferable to inverting the $DS3/\overline{DS1}$ signal each time the complement is needed.

3.3 Encoder (State Machine) Synchronization Module

The function of this module is to generate a signal to synchronize the encoder to the incoming data stream, i.e so that the encoder knows the locations of the data and F, C-bits. The frame information supplied on the *FRAME IN* line is used to generate the synchronization signal. The resulting signal is a pulse, one clock cycle in width, coincident with the first bit of a DS1-FEC or DS3-FEC codeword and is generated by extracting only those pulses on *FRAME IN* which establish a consistent mapping of input DS1 superframes or DS3 frames into DS1-FEC or DS3-FEC codeword blocks respectively.

In DS1 mode, *FRAME IN* carries a signal which is normally high but goes low for one bit time coincident with the first bit of each incoming DS1 superframe. Because each DS1 superframe maps directly to a DS1-FEC codeword as described in Section 2.3.1, the pulse on the *FRAME IN* line in DS1 mode thus marks the beginning of each DS1-FEC codeword consistently and does not need to be changed in any way.

In DS3 mode, the signal on *FRAME IN* is a one clock cycle duration active low pulse which is coincident with the first bit of either (i) each DS3 frame or (ii) each DS3 master-frame. In either case, only every second pulse on the *FRAME IN* line marks the start of a DS3-FEC codeword. In case (i), this is obvious because each DS3-FEC codeword covers exactly two DS3 frames. It is also true for case (ii) because there are seven frames in each

masterframe and two frames per codeword. Figure 3.3 illustrates:

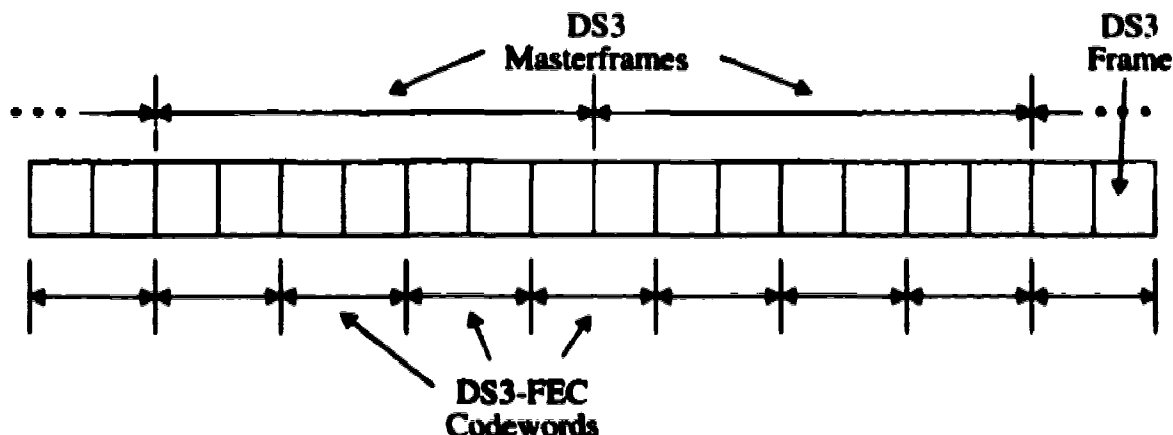


Figure 3.3: Mapping of DS3 frames and masterframes into DS3-FEC codewords

Note that if the masterframe signal is supplied then the synchronization pulses will not mark the beginning of every DS3-FEC codeword but only the beginning of every seventh codeword. This is not a problem as the synchronization is consistent with the codeword mapping established by previous synchronization pulses.

The schematic diagram of the synchronization module is shown in Figure 3.4:.

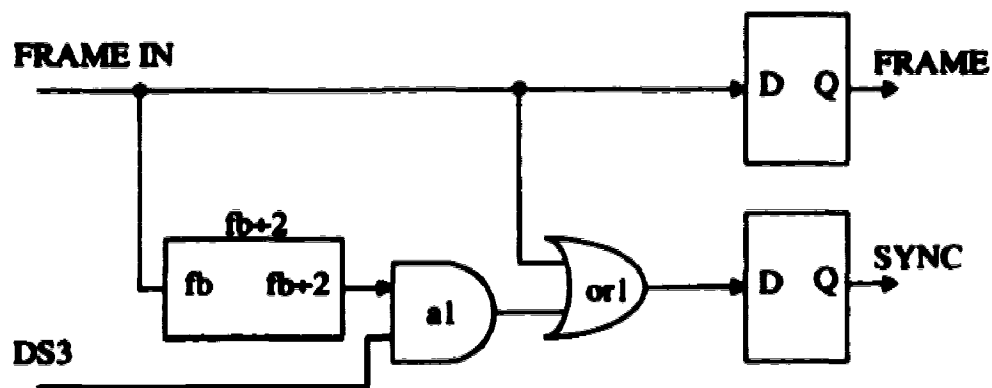


Figure 3.4: Schematic diagrams for encoder synchronization module

The inputs to the encoder synchronization module are as follows:

- **DS3/DS1**: mode select signal
- **FRAME IN**: the framing information supplied on the *FRAME IN* chip level input as outlined above

Although not shown, the system clock and reset signals also control the two D-flip flops in Figure 3.4. The outputs from this module are the following:

- **FRAME OUT:** synchronized input frame timing (re-timed to coincide with the twice re-timed encoder **DATA IN** signal)
- **SYNC:** active high version of the pulse to align the encoder system state controller to the incoming frames

The generation of the synchronization pulse in DS1 mode is straightforward. The **DS3/DS1** mode select input is low which forces the output of *a1* low subsequently making *or1* transparent. The start of superframe indication signal on the **FRAME IN** input is routed to *f1*. The flip flops *f0* and *f1* introduce a single bit time delay, but as noted in Section 3.2, the data is re-timed so that the synchronization pulses coincide with bit 0 of the DS1-FEC codeword.

In DS3 mode, **DS3/DS1** is high so *a1* is transparent. The “fb+2” (frame boundary divided by 2) module (Appendix A) processes the incoming DS3 frame or masterframe data to produce a 50% duty cycle square wave with period of twice the frame or masterframe period respectively, as shown in Figure 3.5.

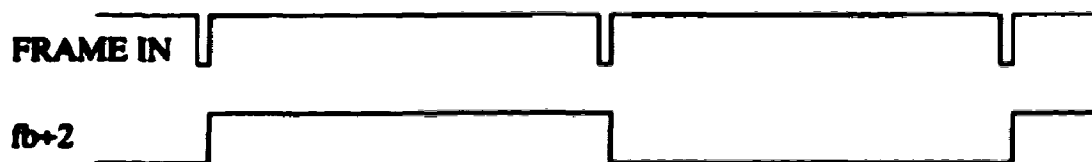


Figure 3.5: Output from fb+2 module

When the “fb+2” output is high, it forces the output of *or1* to a logical 1, preventing the signal on the **FRAME IN** line from reaching *f1*. When the output of “fb+2” is low, the signal on the **FRAME IN** input reaches *f1*. Thus only every second pulse on **FRAME IN** is transferred to the **SYNC** output of *f1*.

Once the system is synchronized to the incoming DS1 or DS3 frame structure, it can begin performing its functions of FEC checkbit computation and insertion of checkbits into the outgoing data stream. The incoming data is routed to three modules: the codeword length delay buffer, the encoding (+G(x)) register, and the parity bit computation register.

3.4 Codeword Delay Register

The delay register shown in Figure 3.1 delays the input serial data bit stream by one full codeword time (2316 bit times for DS1, 1360 bit times for DS3). Thus when bit *i* of a codeword is about to be clocked in at the input of the register, bit *i* of the previous codeword is emerging at its output. This delay is necessary because the computation of the checkbits requires one codeword time and the checkbits for a codeword are carried in the

F and C-bit (DS3) positions of the same codeword as the message bits with which they are associated. The checkbits for a codeword will be valid after all of the message bits for that codeword are shifted in to the encoder. At the same time that the checkbits are valid, the very first bit of the codeword with which they are associated is at the output of the delay register. The delayed codeword is then shifted out and its checkbits are inserted at the appropriate F and C-bit (DS3) positions. Simultaneously, the next codeword block is shifted in and its checkbits are computed. Thus there is a pipeline operation. This is shown in Figure 3.6.

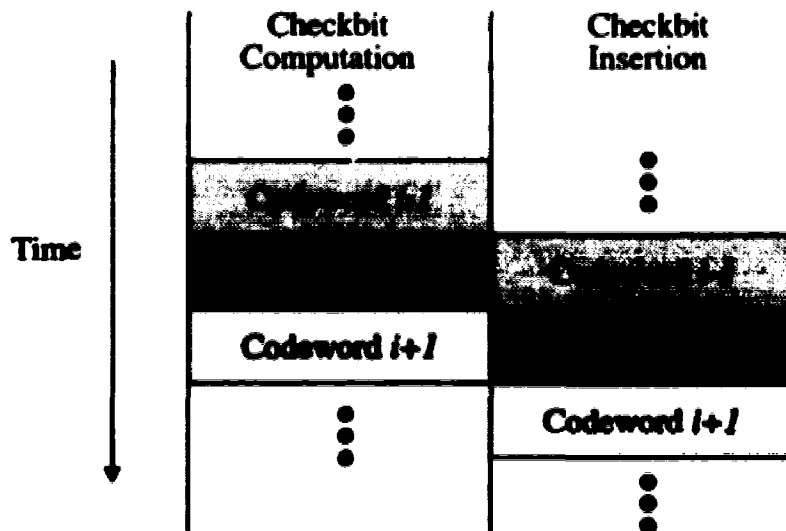


Figure 3.6: FEC Encoder Operation Pipeline

The schematic diagram of the re-configurable encoder delay register is shown in Figure 3.7.

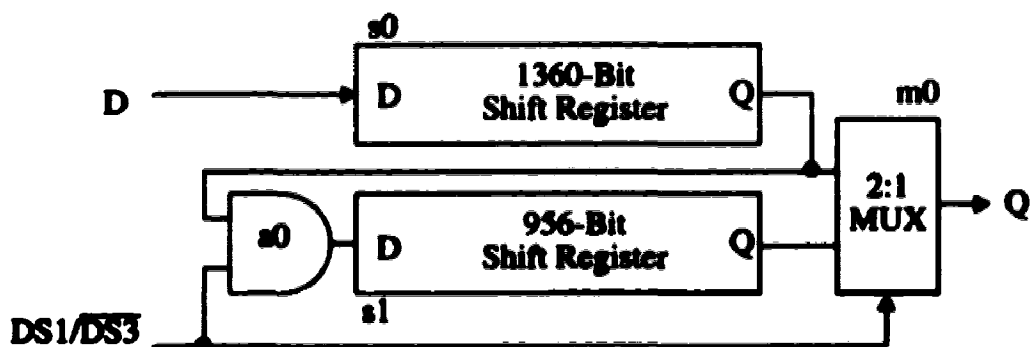


Figure 3.7: Variable length encoder delay register

The circuit has two inputs in addition to the system clock:

- **D:** Serial data input to the delay buffer.

- **DS1**: Mode select signal.

The output, Q , is the delayed data bit stream. Note that the delay register does not require asynchronous reset at initialization time. Its initial random contents are purged as the first (unencoded) codeword is shifted in to the system.

The lengths of the DS1 and the DS3 codewords are 2316 and 1360 bits respectively, and so the length of the delay register is selectable depending on the mode of operation. The sub-circuits $s0$ and $s1$ are shift registers which are 1360 and 956 bits long respectively.

In DS3 mode, $DS1$ is a logical zero. This selects the a input of multiplexer $m0$, making the output of $s0$ the Q -output of the delay buffer. It also forces the output of the AND gate $a0$ to a logical zero, which isolates $s1$ from $s0$ as the 956-bit shift register is unused in DS3 mode.

In DS1 mode, $DS1/\overline{DS3}$ is high. This makes AND gate $a0$ transparent so that $s0$ and $s1$ are connected in series to form a 2316 bit shift register. It also selects the b input of multiplexer $m0$ which makes the output of this concatenated register the Q -output pin of the module.

3.5 Encoding Register

The encoding ($+G(x)$) register is the second module that the incoming DS1/DS3 data is routed to in the FEC encoder. The function of the encoding register is to compute the Hamming code checkbits for a codeword in either the DS1-FEC or DS3-FEC code. As described in Section 2.3, the checkbits for a codeword are computed by (i) multiplying the message polynomial $U(x)$ by x^{n-k} , where $n-k$ is the number of checkbits in the code, and (ii) taking the remainder of the division of this polynomial by the code generator polynomial $G(x)$. These operations are realized using an $n-k$ stage linear feedback shift register with feedback connections based on $G(x)$ [9]. The generator polynomials for the DS1-FEC and DS3-FEC codes are given in Sections 2.3.1 and 2.3.2 respectively.

As an example, consider the linear feedback shift register which implements $G(x)_{DS3} = x^{11} + x^2 + 1$ for the DS3-FEC code shown in Figure 3.8.

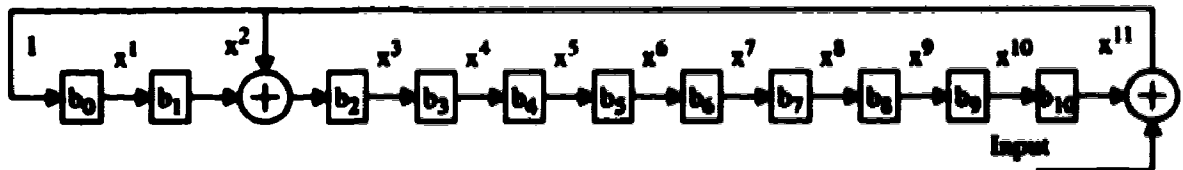


Figure 3.8: Linear feedback shift register implementing division by $G(x)_{DS3}$

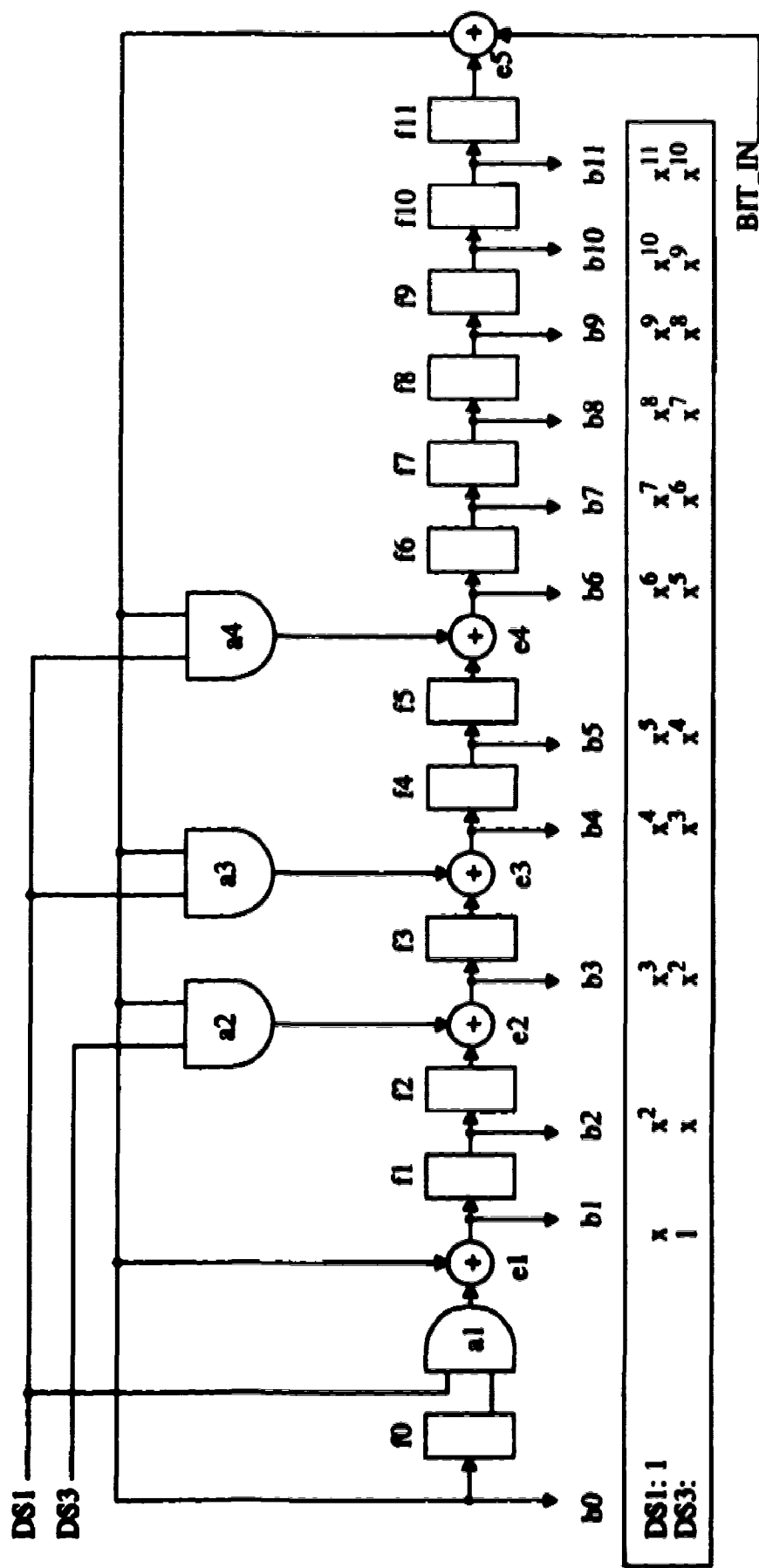


Figure 3.9: Re-configurable FEC Encoding Register (+ $G(x)$)

The checkbits are computed by first clearing this shift register and then shifting the k message bits into the input of the register (this is equivalent to pre-multiplying $U(x)$ by x^{n-k}). After all message bits have been clocked in, the encoding register contains the checkbits b_0, b_1, \dots, b_{10} , where b_{10} is the most significant checkbit. The checkbits are then ready for distributed insertion into the DS3-FEC codeword after parallel transfer to a holding register.

The schematic diagram for the re-configurable DS1/DS3 encoding register is shown in Figure 3.9

The encoding in register is composed of 12 D-flip flops, each having synchronous load and shift enable (Appendix A). In addition to the asynchronous reset and the system clock the following are the inputs to the encoding register:

- **BIT IN**: the new bit about to be shifted into the encoding register
- **LOOKAHEAD CLEAR**: synchronous “look ahead” clear of the encoding register on the next rising clock edge (detailed discussion to follow)
- **SHIFT ENABLE**: when asserted, enable shifting in encoding register. If de-asserted, prevent **BIT IN** from being shifted into the encoding register and hold the contents of the register unchanged.
- **DS3/DS1, DS1/DS3**: complementary mode select control signals

The outputs of the encoding register are the 11 DS3-FEC checkbits or the 12 DS1-FEC checkbits which are valid at the end of the a codeword division cycle.

The encoding register shown in Figure 3.9 is re-configurable to implement either $G(x)_{DS1}$ or $G(x)_{DS3}$ depending on the mode of operation. In DS1 mode, all 12 flip flops are used. In DS3 mode, only 11 of the 12 stages of the shift register are used. The least significant (left-most) stage is unused in DS3 mode. The AND gate, $a1$, is used to disconnect the left-most flip-flop from the encoding register in DS3 mode by setting the output of this stage to zero. In DS1 mode, $a1$ is transparent.

The modulo-2 addition at the feedback points is implemented using exclusive-or gates $e1$, $e2$, $e3$, and $e4$. The positions of the shift register feedback taps are different for DS1 and DS3 modes. In DS1 mode, flip-flop #0 is the least significant stage in the register and feedback taps are thus required at positions 0, 1, 4, and 6 to realize $G(x)_{DS1}$. In DS3 mode, flip flop #1 is the least significant element, so feedback taps are required at the inputs to flip flops 1 and 3. In both cases, the feedback signal is the modulo-2 sum of the bit about to be shifted in and the output of FF #11. The AND gates $a2$, $a3$, and $a4$ are used to enable to feedback taps. The tap at position 1 is common to both modes of operation so the feedback

line is simply routed to the input of $e1$. In DS1 mode, the output of $a2$ is a logical zero which disconnects the feedback tap for XOR gate $e2$. However, $a3$, and $a4$ are transparent and pass the value on the feedback line to the inputs of $e3$ and $e4$. In DS3 mode, $a3$ and $a4$ are turned off, breaking the feedback connection while $a2$ is transparent completing the feedback connection to $e2$.

The algorithm outlined at the beginning of this section for computation of the checkbits assumes that the message bits were contiguous. In the DS1-FEC and DS3-FEC codeword formats, this is not the case. The F-bits (and selected C-bits) in the positions where the checkbits are to be transported will be replaced and must be omitted from the encoding process. At those times when an F-bit is on the *BIT IN* line, about to be shifted in to the encoding register, the shift enable input is disabled, forcing the shift register to hold its present contents on the next rising clock edge. Thus the F-bit is effectively gapped out of the encoding process.

The checkbit computation process previously outlined also assumes that there is ample time between each unencoded data block to clear the encoding register. However, this is not the case for the codeword formats dealt with here because the first bit of the next codeword to be encoded follows immediately after the last message bit of the codeword currently being processed. This is shown in Figure 3.10 for DS3.

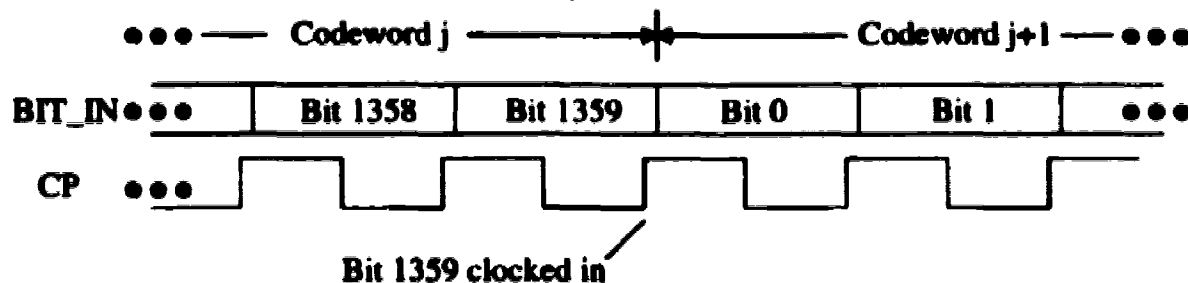


Figure 3.10: Diagram showing codewords j and $j+1$ are contiguous

Thus the first bit of codeword $j+1$ is at the *BIT IN* input in the same clock cycle at which the checkbit computation is completed for codeword j . To process the next bit, the following three operations are required in the same clock cycle:

- (i) Parallel transfer valid checkbits for codeword j into a holding register (Section 3.7) for insertion into the F (and C for DS3) positions of codeword j .
- (ii) Clear the encoding ($+G(x)$) register.
- (iii) Shift in the first bit of codeword $j+1$.

An elegant solution to combine these operations (ii) and (iii) is a "look-ahead" clear (or selective pre-load). In this case, the encoding register is loaded with the value it would

contain if it was first cleared and then the new bit clocked in. For DS3 mode, if the new bit is a zero, this will be 00000000000. But, if the new bit is a 1, then the register value becomes 10100000001. In DS1 mode this problem does not exist because the first bit of an unencoded DS1-FEC block is an F-bit which must not be shifted in. Therefore, once the last data bit of a codeword has been clocked in to the register, there is time to assert the clear line so that the register is set to all zeros. The F-bit is thus ignored and coding properly begins with the first data bit. Table 3.1 summarizes the input functions for the load data input pins on flip flops f0 to f11 to implement this look-ahead clear function.

Table 3.1: Load Data Input Functions for encoding register flip flops

Flip Flop #	Load Data Input Function
f0, f2, f4, f5, f6, f7, f8, f9, f10, f11	0
f1, f3	(BIT IN) AND (DS3)

The final point to note about the encoding register is that the checkbit lines are actually taken from the D-inputs of the flip flops in the register rather than from the Q-outputs. This allows access to the checkbits late in the clock cycle prior to when they actually become valid at the Q-outputs of the flip flops. The reason this is beneficial is outlined in the Section 3.7, but basically eliminates the need for additional bit time pipelining in the encoder. As the checkbit information is available one clock cycle early, they are called the “look-ahead” checkbits.

3.6 Parity Bit Register

The third and final module to which the incoming DS1 or DS3 data is routed is the parity register. The function of the parity register is to compute the parity bit for a DS3-FEC codeword, such that the overall parity of the codeword is even. The value of parity bit is thus defined as follows:

$$b_p = \sum_{j=0}^{1348} u_j + \sum_{j=0}^{10} b_j \quad (3.1)$$

where u_j is the j^{th} message bit, b_j is the j^{th} checkbit for the codeword, and the summation denotes modulo-2 addition. A bit serial approach is used to compute the first term in (3.1). The parity register initially sets b_p to zero and then computes the running sum of the parity of the message bits of the codeword as they are shifted in. Once the parity of the message bits is found, it is modulo-2 added in parallel with the 11 checkbits. The reason for this is that b_p must be ready at the next rising clock edge. The schematic diagram for the parity

register is shown in Figure 3.11

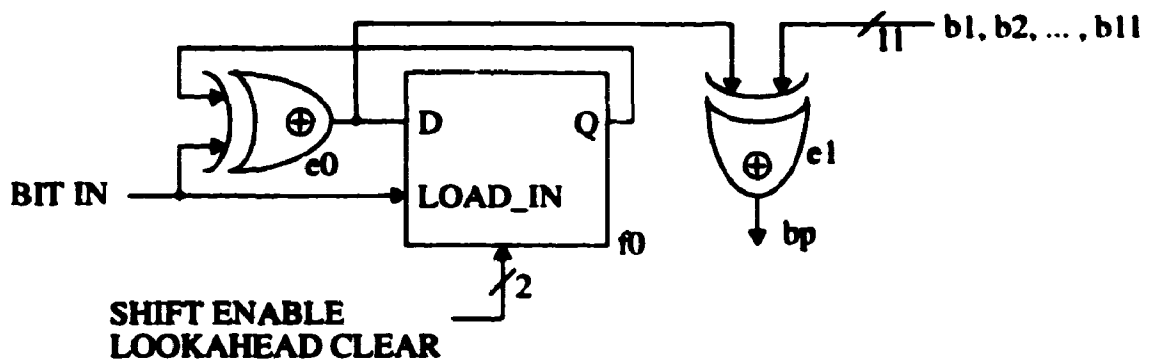


Figure 3.11: Schematic diagram of parity register

In addition to the system clock and the asynchronous reset, the following are inputs to the parity register

- **BIT IN**: the new data bit about to be clocked in to the parity register
- **SHIFT ENABLE**: enable shift of **BIT IN** into parity register. When disabled, hold current value in the register.
- **LOOKAHEAD CLEAR**: synchronous look-ahead clear of parity register on next rising clock edge. Loads the register with the value it would contain if it was first cleared and then the bit on the **BIT IN** line was shifted in
- **b1...b11**: checkbits from the encoding register (only the 11 most significant lines)

The output, **bp**, is the modulo-2 sum of the 1348 message bits and the 11 checkbits for the codeword as given in (3.1).

The circuit operates as follows. In the clock cycle, when bit 0 of a new codeword is on **BIT IN**, **LOOKAHEAD CLEAR** is asserted to “look-ahead” clear the parity register on the next rising clock edge. As each new bit arrives on **BIT IN**, its value is modulo-2 added to the cumulative parity stored on the Q-output of **f0**, via **e0**, and the result is stored on the Q-output of **f0** on the next clock edge. When the bit on the **BIT_IN** line is an F or selected C bit, the **SHIFT ENABLE** control signal is de-asserted, forcing the register to hold its current value, gapping out the bit. Once all bits in the unencoded data block have been shifted in, the parity of the message bits is at the Q-output of **f0**. However, the message bit parity can be obtained one clock cycle earlier. When the last message bit (bit 1359 in the DS3-FEC codeword) is on **BIT IN**, its value is added to the parity of the previous $k-1$ message bits and the result is at the D-input of **f0** (after the gate delay of **e0**), ready for transfer to the Q-output on the next rising clock edge. The message bit parity is taken from the D-input of **f0** and summed in parallel with the 11 “look-ahead” checkbits to form the **bp** out-

put signal via gate *e1*. Thus the parity of the entire codeword is available late in the clock cycle before the last bit of the unencoded data block is actually shifted in. As will be outlined in the next section, computing the parity this way eliminates the need for bit time pipelining in the FEC encoder.

Because the "look-ahead" checkbits are taken from the D-inputs of the flip-flops in the encoding register, they are not all available at the same time in the clock cycle, i.e. some must pass through more levels of logic than others. As well, the 12-bit XOR gate is composed of the 2 and 3-input XOR gates available in the LSI Logic design library, and each input has a different input-to-output delay characteristic. Table 3.2 summarizes the nominal computation delay times for each checkbit. In addition, the delay characteristic for the message bit parity signal at the D-input of *f0* is (rise / fall) = (3.02, 3.28). Table 3.3 presents the nominal input to output delays for each input of XOR gate *e1*. Careful mapping of the checkbits with the longest decoding delay to the inputs of *e1* with the shortest delay allowed for minimization of the total delay to realize *bp*. Table 3.4 summarizes the assignments of checkbits to XOR gate inputs.

Table 3.2: Decoding delay for each look-ahead checkbit to become valid

Checkbit	Decoding delay (rise/fall) (ns)	Checkbit	Decoding delay (rise/fall) (ns)	Checkbit	Decoding delay (rise/fall) (ns)
1	4.24/4.48	5	2.12/2.03	9	2.12/2.03
2	2.12/2.03	6	4.62/4.86	10	2.12/2.03
3	4.62/4.86	7	2.12/2.03	11	2.12/2.03
4	4.62/4.86	8	2.12/2.03		

Table 3.3: Input to output delay characteristics of 12 input XOR gate

Input	Delay (rise/fall) (ns)	Input	Delay (rise/fall) (ns)	Input	Delay (rise/fall) (ns)
1	3.16/3.12	5	2.55/2.51	9	1.43/1.39
2	2.87/2.83	6	2.26/2.22	10	1.87/1.83
3	2.83/2.79	7	2.11/2.07	11	1.49/1.45
4	2.54/2.50	8	1.73/1.69	12	1.19/1.15

Table 3.4: Checkbit assignments to XOR gate inputs

XOR Input Number	Checkbit Number	XOR Input Number	Checkbit Number	XOR Input Number	Checkbit Number
1	11	5	7	9	3
2	10	6	5	10	U(x) Parity
3	9	7	2	11	4
4	8	8	1	12	6

Once the checkbits and the parity bit (DS3) have been computed, they are ready for insertion into the F and C bit positions. To make the encoding structures available to compute the checkbits for the next codeword, the checkbits and overall parity bit (DS3) are transferred to the 12-bit parallel-in/serial-out (PISO) holding register.

3.7 Holding Register

The holding register stores the checkbits for distributed insertion into the codeword data emerging from the delay buffer, overwriting the F and selected C-bits (DS3). The schematic for the holding register is shown in Figure 3.12.

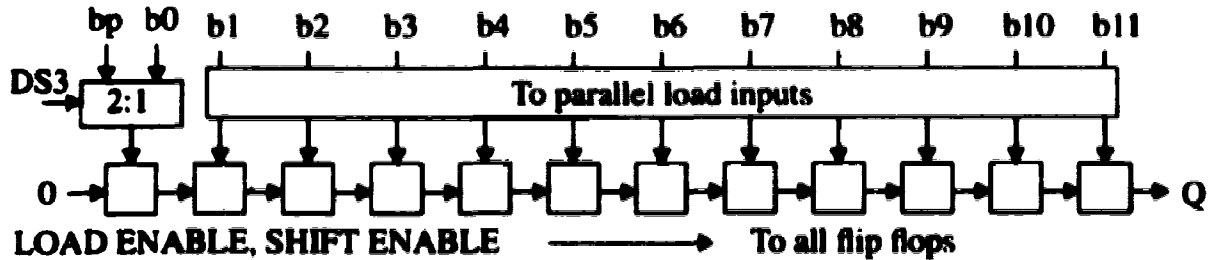


Figure 3.12: Schematic Diagram of 12-bit holding register

The holding register is composed of 12 D-flip flops, each having a shift enable and a load enable (Appendix A). In addition to the asynchronous reset and the system clock the following are the inputs to the holding register:

- $b_0, b_1, b_2, \dots, b_{11}$: 12 lines of checkbit information. In DS1 mode, all 12 lines are used. In DS3 mode, only the 11 most significant lines carry checkbit information.
- **PARITY**: overall parity bit for the DS3-FEC codeword.
- **LOAD ENABLE**: enable synchronous parallel load of the checkbits (and parity bit for DS3) at the "load in" inputs of the flip-flops into the register.
- **SHIFT ENABLE**: enable serial shift of holding register. Insert a logic 0 into the least

significant bit position in the holding register on each shift.

- **DS3/DS1**: mode select.

The holding register has only one output, *Q*, which is the next checkbit scheduled for insertion into the codeword leaving the delay register at the next F-bit (or C-bit DS3) time.

The 2:1 MUX, controlled by the *DS3* mode select line, selects between the 12th checkbit in DS1 mode and the overall parity bit in DS3 mode for load into the least significant stage of the register.

The holding register is used in the codeword construction process in the following way: In the clock cycle prior to the last data bit shifted into the encoder, the checkbits for the codeword and the overall parity bit (DS3 only) are at the D-inputs of the flip-flops in the encoding register and parity bit register, as described in Sections 3.5 and 3.6. These points are routed to the holding register parallel load inputs. In this same cycle, the *LOAD ENABLE* signal is asserted, to load these bits into the holding register on the next rising clock edge. As the contents of the delay buffer are shifted out, the contents of the holding register are held until an F-bit (or selected C-bit) time. At this time, the Output Multiplexor (Section 3.8) takes the checkbit at the serial output of the holding register and overwrites the bit in the outgoing data stream with it. The holding register *SHIFT ENABLE* input is simultaneously asserted to shift this checkbit out of the holding register on the next rising clock edge and move the next most significant checkbit to the serial output. The holding register contents are then held until the next F-bit time. This process continues until all checkbits have been serially shifted into the outgoing data stream. Meanwhile, the next set of checkbits are being computed in the +G(x) register as the next set of codeword data arrives.

3.8 Output Multiplexor

As alluded to above, the main function of the output multiplexor is to overwrite the F and C-bits in the data stream emerging from the delay register with the appropriate checkbits from the holding register. However, this is only done after the second task of the module is performed, that is, the sampling of the *FEC ENABLE* chip level input signal at the codeword boundary times, i.e. when bit 0 of a codeword is emerging from the decoder delay buffer. This allows a "clean" switch between coding on / coding off modes at the output.

This is illustrated in Figure 3.13:

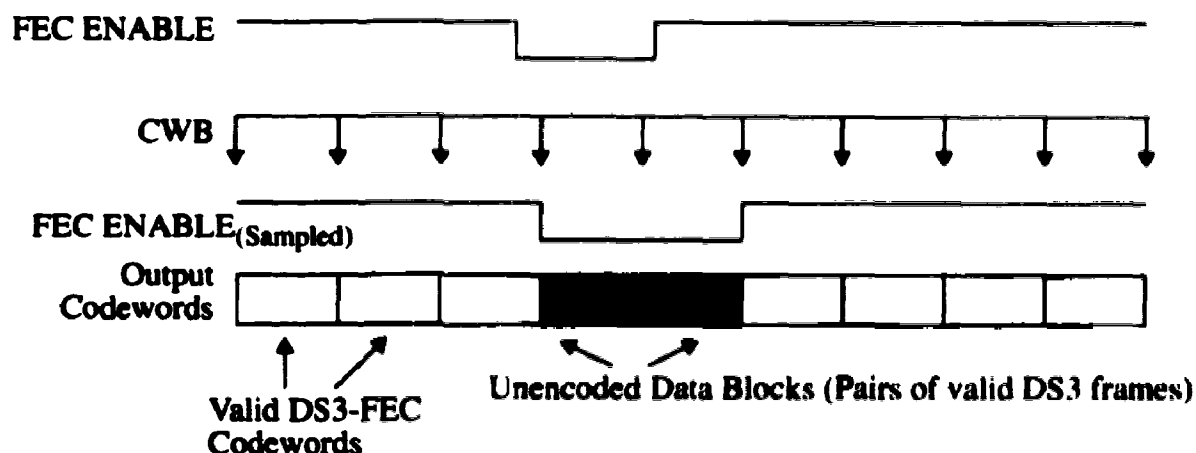


Figure 3.13: Demonstration of effect of sampling FEC on at codeword boundary times

Thus there is no case where a partially encoded data block emerges from the FEC encoder, it is either completely FEC encoded or left as two valid DS3 frames in DS3 mode or one valid DS1 superframe in DS1 mode. The schematic diagram of the output conditioning module is shown in Figure 3.14

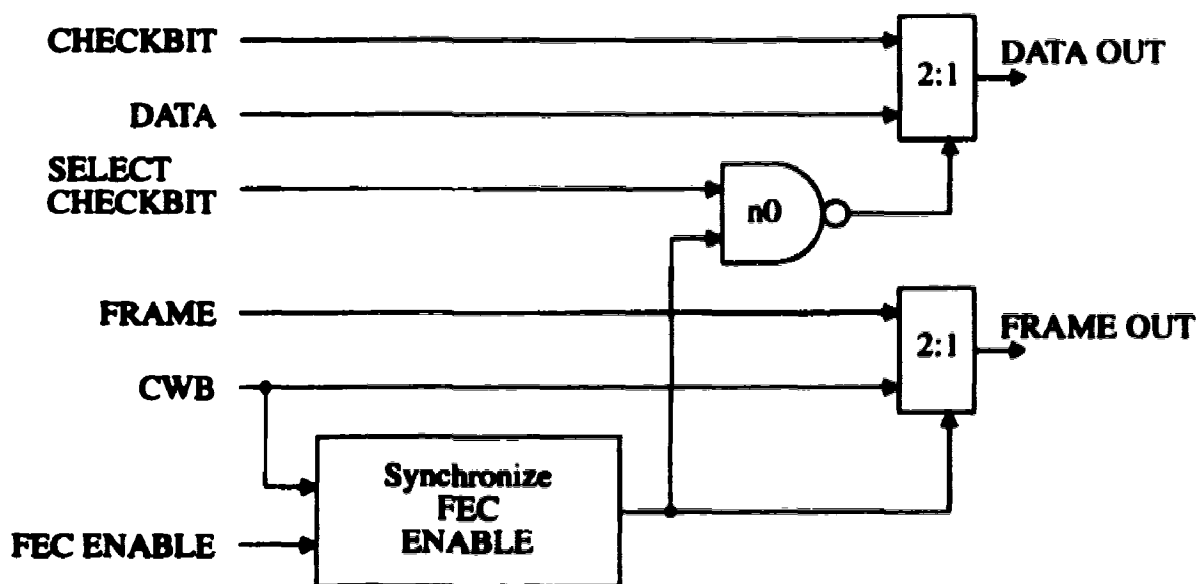


Figure 3.14: Schematic for FEC encoder output multiplexor

The following are the inputs to the output multiplexor:

- **DATA:** data bits emerging from the delay buffer
- **FRAME:** conventional framing information from *FRAME IN* chip level input

- **CWB**: active low pulse marking the first bit of the codeword emerging from the delay buffer
- **FEC ENABLE**: FEC coding on /off enable select
- **SELECT CHECKBIT**: control signal from system controller (timing state machine) asserted at those times that an F or selected C-bit (DS3) should be replaced by a check-bit
- **CHECKBIT**: checkbit current on the serial output of the holding register

and the following are the outputs from the module:

- **DATA OUT**: FEC encoded data if sampled value of **FEC ENABLE** is true; unchanged delayed data if sampled value of **FEC ENABLE** is false
- **FRAME OUT**: codeword boundary signal if sampled value of **FEC ENABLE** is true; conventional frame timing information if sampled value of **FEC ENABLE** is false.

The "Synchronize FEC ENABLE" module (Appendix A) is used to sample the value on the **FEC ENABLE** input at the codeword boundary time so that its output changes only synchronously with the codeword boundary. When the synchronized value of **FEC ENABLE** is low, the replacement of F and C-bits by checkbits is disabled. The output of the NAND gate, *n0*, is forced high, blocking the effect of the **SELECT CHECKBIT** signal. The bits emerging from the delay buffer are selected as the FEC encoder output data stream. The conventional framing information supplied to the FEC encoder on the **FRAME IN** chip level input is selected as the output frame information.

When the synchronized value of **FEC ENABLE** is high, the replacement of F and C bits with checkbits is enabled. Each time the **SELECT CHECKBIT** signal is asserted, the checkbit on the serial output of the holding register (**CHECKBIT**) is selected as output instead of the data bit emerging from the delay buffer. When **SELECT CHECKBIT** is de-asserted, the data bits emerging from the delay buffer are selected as the FEC encoder data output. Thus an encoded DS1-FEC or DS3-FEC codeword is the output. The **CWB** signal is passed as frame information.

3.9 Encoder Event Sequencer

The signals to control the various modules of the FEC encoder, (for example, to transfer the checkbits from the encoding register to the holding register, or to gap out the F-bits from the encoding process) come from the Encoder Sequencer. As shown in Figure 3.1, the sequencer consists of two modules, the 12-bit system state counter, and the state decoder. The 12-bit counter provides encoder state information by counting the bits of a DS1-FEC or DS3-FEC codeword as they are shifted into the encoder. The numbering con-

vention is the following: the MSB of a codeword is bit 0 and the value of the 12-bit number in the counter is the index of the bit which has just been clocked into to the delay register (and, if it is not an F or C-bit, into the encoding register and parity register). For example, when the counter reads 18, (i) bit 18 has just been shifted into the delay register, (ii) bit 19 from the same codeword is at the D-input of the delay register, but has not yet been clocked in, and (iii) bit 19 from the previous codeword is on the Q-output of the delay buffer. The $DS3/\overline{DS1}$ input selects the counting range. The counter counts from 0 to TC and then wraps to 0 where TC is the terminal count of 2315 for DS1 or 1359 for DS3. The state decoder monitors the value of the 12-bit counter and functions like a microcode program to execute all operations to process either 2 DS3 frames or 12 DS1 frames into the respective FEC codeword. The $DS3/\overline{DS1}$ and $DS1/\overline{DS3}$ signals are used to enable decoding terms for the appropriate mode of operation and disable those associated with the complementary mode.

3.9.1 Control Signals

The FEC encoder system controller generates four control signals.

- **CWB**: an active low signal, one clock cycle in width which marks the start of the DS1-FEC or DS3-FEC codeword. It is also used to control the “look-ahead clear” function of the encoding register and the parity bit computation register.
- **ENC_REG_SHIFT** (Encoding Register Shift Enable): Hold contents of encoding and parity bit registers when an F-bit (or C-bit DS3) is about to be clocked in (Gap out the F and C-bits from $+G(x)$).
- **HOLD_REG_LOAD** (Holding Register Load Enable): Enable synchronous parallel transfer of checkbits from the encoding register and the overall parity bit from the parity bit register into the 12-bit PISO holding register
- **SELECT_CHECKBIT**: Overwrite the current bit emerging from the delay buffer with a checkbit from the serial output of the holding register. Also enable shift of holding register to discard this bit on the next rising clock edge

3.9.2 Sequencer Operation in DS1 Mode

In the DS1-FEC code, the checkbits for the FEC code are at bit positions 0, 193, 386, 579, 772, 965, 1158, 1351, 1544, 1737, 1930, and 2123. Table 3.5 summarizes the values of the

system counter for which each control signal is asserted.

Table 3.5: Encoder Event Sequencer Logic Functions for DS1

Counter Value	CWB	SELECT CHECKBIT	ENC_REG SHIFT ^a	HOLD_REG LOAD
192, 385, 578, 771, 964, 1157, 1350, 1543, 1736, 1929, 2122	0	1	1	0
2314	0	0	0	1
2315	1	1	0	0

a. This is actually the complement of the ENC_REG_SHIFT signal. The proper polarity is established before the signal leaves the system controller.

At all other times, the values of the four signals are zero (i.e. de-asserted).

3.9.3 Sequencer Operation in DS3 Mode

For the DS3-FEC code, the checkbits are bit positions 85, 255, 340, 425, 510, 595, 765, 935, 1020, 1105, 1190, and 1275. Table 3.6 summarizes the system counter values for which the control signals are asserted in DS3 mode.

Table 3.6: Encoder Event Sequencer Logic Functions for DS3

Counter Value	CWB	SELECT CHECKBIT	ENC_REG SHIFT ^a	HOLD_REG LOAD
84, 254, 339, 424, 509, 594, 764, 934, 1019, 1104, 1189, 1274	0	1	1	0
254	0	0	0	1
339	1	0	0	0

a. This is actually the complement of the ENC_REG_SHIFT signal. The proper polarity is established before the signal leaves the system controller.

It is desirable to have these four important control signals as free as possible from decoding glitches. To do this, the control signals are actually decoded when the system counter is one less than the counter values listed in Tables 3.5 and 3.6. The resulting signals are then re-timed in D-flip flops which provide one clock cycle time delay and remove the asynchronous decoding glitches.

3.9.4 State Machine Decoding Logic Minimization

Note that in both cases above, the state machine decoding has a large number of “don’t cares”. Moreover, in DS3 mode, the system counter never counts beyond 1359. Therefore values of the control signals may be unspecified for counts between 1360 to 4095. In DSI mode, the counter wraps around after it reaches 2315, so the encoder control signals may be unspecified for system counter values in the range 2316 to 4095 in the 12-bit address space. Minimized implementations of the control functions can therefore be advantageously achieved by exploiting the “don’t care” set. However, traditional Karnaugh map minimization techniques are impractical here because each control signal is a function of 13 input variables (12 bits from the counter and the *DS3*, *DSI* mode select). Instead, a modern logic design tool called “espresso” [11] was used to minimize these boolean functions.

“Espresso” takes two data sets, the ON-set (conditions for which the function is a logical 1) and the “don’t care” set (conditions for which the value of the function is unspecified) and produces a Boolean minimized 2-level AND-OR representation of the function. The “espresso” input file and output file for the FEC encoder system control state decoder are in Appendix B. Simulation verified that the un-minimized and minimized realizations of the control signals were functionally identical.

Direct implementation of the un-minimized control functions as listed in Tables 3.5 and 3.6 would have required 371 gates in the LSI Logic design system. The minimized implementation used only 232 gates.

3.9.5 Synchronization of Event Sequencer to Incoming Data Stream

The system state counter in the system controller has a synchronous clear input which causes the state counter to go to the all zero state on the next rising clock edge. The *SYNC* signal from the FEC encoder synchronization module controls this input. When this signal is asserted, bit 0 of a codeword is at the D-input of the delay register. If the encoder is not synchronized to the incoming data, then on the next rising clock edge, bit zero is shifted into the delay register and the system counter is forced to zero. However, if the encoder is synchronized with the incoming data stream, the *SYNC* signal will be asserted in the same cycle at which the counter reaches its terminal count, and the counter will naturally wrap around to zero on the next rising clock edge. Thus synchronization of the system counter is established.

Some modifications are also made to the system controller decoding logic to allow for immediate encoding of a new FEC codeword to begin immediately after a re-synchroniza-

tion occurs. As noted in Section 3.9.3, the control signals are derived by decoding the system state one clock cycle earlier than the cycle times required in Tables 3.5 and 3.6, combining the terms into the respective control functions and routing these functions to D-flip flops to remove decoding glitches. When a re-synchronization occurs, the SYNC signal sets the outputs of these flip flops to the values that the control signals should take on when the counter is at its terminal count and the inputs of the flip flops to the values of the control signals when the counter is zero. These are listed in Table 3.7

Table 3.7: Control Signal Values for System Counter = TC and Zero

Signal	Value for Counter=TC		Value for Counter=0	
	DS1	DS3	DS1	DS3
CWB	1	1	0	0
SELECT CHECKBIT	1	0	0	0
ENC_REG SHIFT	0	0	0	0
HOLD_REG LOAD	0	0	0	0

The most important effect that this has is that the *CWB* signal is asserted when the synchronization occurs. The signal forces a “look-ahead” clear on the encoding and parity bit registers. If this was not done, the encoding and parity registers would contain some arbitrary value at the time of the synchronization and the checkbits for the first codeword shifted in would be incorrectly computed (although it would be correct for all subsequent codewords). By using the *SYNC* signal to synchronize the state decoding logic the encoding of the incoming codeword can begin immediately after a re-synchronization.

Chapter 4

The FEC Decoder

In this chapter, the implementation of the DS1/DS3 FEC Decoder is presented. The chapter presents detailed descriptions of each of the modules of the Decoder. The functions implemented by these modules include: continuous syndrome calculation and single bit time parity bit computation (DS3), frame alignment based on FEC syndrome decoding, error location, detection, and correction and conventional F-bit (and C-bit for DS3) sequence reconstruction.

4.1 Top Level Decoder Module

The top level module for the DS1/DS3 FEC decoder is shown in Figure 4.1. The following are the chip level inputs to this system:

- **DATA_IN**: DS1-FEC or DS3-FEC NRZ serial data input line.
- **OFD_ENABLE**: Enables the OFD process once the decoder is in-frame. When de-asserted, the OFD process is inhibited and the decoder will continue to maintain the current position as frame even if true frame is lost. (This is a test mode consideration only).
- **C_R** (RE-FRAME Confirmation Threshold): 5-bit number which indicates how many consecutive frames must decode a zero syndrome after a frame candidate position is found to declare the in-frame state.
- **C_O** (OFD Confirmation Threshold): 5-bit number of consecutive non-zero syndromes for out-of-frame declaration, after an initial non-zero syndrome is observed.
- **DS3/DS1** (MODE SELECT): Re-configures the internal structures in the FEC decoder for operation on either DS1-FEC or DS3-FEC signals. Set to HIGH for DS3 mode; set LOW for DS1 mode
- **CP**: DS1 or DS3 bit rate clock recovered from receiving line interface.
- **RESET**: Active-low asynchronous reset line for bringing internal finite state machines (FSM's) and registers into a known state.

The following are the chip level DS1/DS3 FEC decoder outputs:

- **DATA_OUT**: Error corrected DS1 or DS3 data with the F and C-bits (DS3 only) restored to the conventional format.
- **CP_OUT**: DS1 or DS3 output bit time clock. The rising edge of this clock signal is in the center of the data bits emerging on the decoder **DATA_OUT** line.

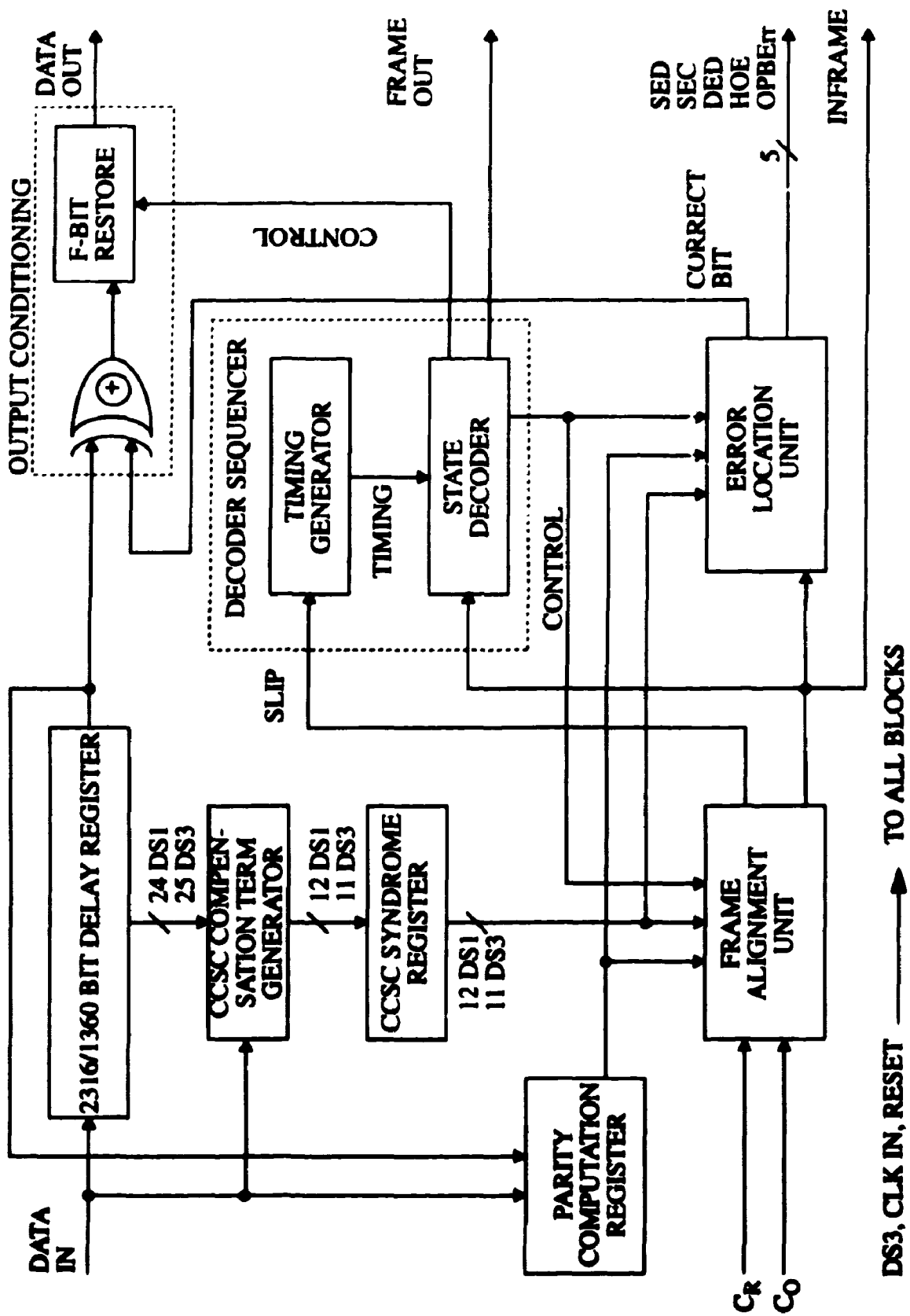


Figure 4.1: DS1/DS3 FEC Decoder block diagram

- **FRAME OUT** (Frame Boundary): This line carries a signal which marks the start of the DS1 superframe in DS1 mode or the start of each DS3 frame in DS3 mode. The signal is an active low pulse, one bit time in width, coincident with the first bit of the DS1 superframe or DS3 frame, as located by the frame finding logic.
- **INFRAME**: FEC Decoder Out-of-frame/ $\overline{\text{In-frame}}$ status indication.
- **SLIP**: Active when the system is in the search phase of the re-frame process. Each time a slip is made this line pulses high for one clock cycle. It is low when the system is in the confirmation phase of the re-frame process or when the decoder is in-frame.
- **SEC**: Single error correction event indication. The signal is a pulse, one clock cycle width, when the bit which was determined to be in error, emerges from the receiving delay register.
- **SED**: Single error detection event indication. Signal is a pulse which is one clock cycle in width. If the bit error is in a data bit then error correction activity will take place and a pulse will also occur on SEC. If an error was in a checkbit in a codeword, then no error correction is necessary because the checkbits are later replaced by the conventional OH-bit structure. This line does indicate that a particular error occurred even if no explicit error correction action was taken.
- **DED**: Double error event indication (DS3 mode only). Indicates received word contains a multiple weight even error, the most common being two bit errors.
- **HOE**: Higher order error event indication. Indicates when a multiple weight error event (>1 for DS1; 3, 5, 7, ... for DS3) has occurred. Only a percentage of such events can be detected.
- **OPBErr**: Overall Parity Bit error event indication (DS3 only)

The DS1/DS3 FEC Decoder is composed of seven main functional blocks. When the incoming data is received on the *DATA_IN* line, it is first routed to the *Codeword Delay Buffer*. This module serves two purposes. Firstly it delays the data by one full codeword time. This is necessary because all bits of the received word must be shifted in to the decoder before it can be determined that it contains an error. If it does contain an error then the error is potentially in the first bit of the codeword so a full codeword delay is needed to hold back the data for possible error correction. The delayed data is also used to generate the compensation functions required to implement CCSC for fast re-framing. The CCSC process is implemented via the *Codeword Delay Buffer*, the *CCSC Compensation Function Generator*, and the *CCSC syndrome (+G(x)) Register* (Section 4.3). In DS3 mode, the parity of the received word is also continuously computed. This is done by the *Parity Computation Register* using delayed data in the *Codeword Delay Buffer*. The syndrome and parity information are then passed to the *Frame Alignment Unit* (FAU) and to the *Error Correction Unit* (ECU). The FAU processes the syndrome and parity information and implements the frame alignment algorithms outlined in Sections 2.4.1 and 2.4.2. The

ECU utilizes the syndrome and parity information to determine if any errors exist in the data and if a single bit error has occurred, to locate the error. The *Output Conditioning* block is responsible for actually correcting a single bit error using error location information from the ECU module and for reconstructing the conventional overhead bit sequence after the checkbits of the FEC code are no longer needed. The *System Controller* provides control signals for synchronization and communication between modules and for processing of data. Each of these modules will be discussed in the following sections.

4.2 Chip Level I/O Interface

4.2.1 Input Re-Timing Flip Flops

To allow one complete clock cycle time for processing each incoming data bit, the *DATA IN* is re-timed with a D-Flip Flop (not shown in Figure 4.1). The *OFD_ENABLE* line is an asynchronous input, i.e. it can change at any time in a system clock cycle, so it is re-timed with two stages of D-flip flops to reduce metastability related problems. The C_R and C_O confirmation threshold lines and the DS3 mode select line are not re-timed at the chip inputs as they are expected to maintain hard-wired external values throughout the duration of the operation period. If it is necessary to change C_R or C_O during operation then the proper procedure is outlined in Section 4.4.

As both the signal on the *DS3/DS1* line and its complement are needed at various points in the FEC Decoder for controlling the re-configurable structures, the *DS1* signal is generated immediately after going on the chip by inverting the *DS3* input signal. Both signals are routed to the numerous places they are needed. This requires more routing space but is still preferred to inverting the *DS3* signal at each point that it is needed.

4.2.2 Output Re-Timing Flip Flops

All output signals are re-timed with D-flip flops so that they are completely glitch free before going off chip. The *CP_OUT* signal is the inverse of the input system clock signal and has its rising edge in the center of the outgoing data eye.

4.3 Syndrome Computation

Syndrome computation is an essential function within the FEC decoder. Not only is the syndrome used to locate single bit errors in the received word (its traditional function), but it is also used as the feedback control signal for the frame alignment process. As presented in Section 2.3, the syndrome for the received word at the decoder is the remainder after division of the received word polynomial, $R(x)$, by the code generator polynomial $G(x)$.

The division by $G(x)$ is implemented using a linear feedback shift register [9].

Using shift register division techniques, the traditional syndrome computation process for a Hamming code involves the following steps performed in synchronization with the received word boundaries:

- (i) Clear the syndrome register, i.e. the $G(x)$ division register.
- (ii) Shift all received word bits into the $+G(x)$ register in systematic form (i.e. all data bits followed by all checkbits).
- (iii) Evaluate syndrome.

Thus it takes n bit times to generate one syndrome for an n -bit codeword. If directly applied to serial search frame finding, up to $(n-1)$ complete codewords of an (n, k) code would be required in the maximum length hunt for codeword alignment. With the DS3 code, this is $O(1360^2)$ bit times and is far greater than conventional framing speeds (as seen in Section 2.4.3). Fast framing is therefore derived from the FEC codeword by *sliding* across the data and continually computing the syndrome of the preceding 1360 bits by compensating the previous syndrome result for every new bit received. By continual compensation of a never-reset syndrome calculation, a zero syndrome is observed as soon as the codeword boundary appears. With this approach, no more than 1360 bits will pass before the codeword boundary is seen (neglecting mimic effects). As a result, DS1-FEC and DS3-FEC signals now frame *faster* than in the conventional formats. The search time will be followed by the C_R confirmation period which will actually dominate over the search time.

The details of the CCSC method for the specific cases of the DS3-FEC and DS1-FEC codes are presented in Section 4.3.1 and 4.3.2 respectively. The three circuits in Figure 4.1 which implement the CCSC algorithm are the codeword length delay buffer, the CCSC compensation function generator, and the syndrome ($+G(x)$) register. Each of these modules is discussed in Section 4.3.3, 4.3.4, and 4.3.5 respectively.

4.3.1 The Process of CCSC Decoding for DS3

The delay register can be viewed as an n -bit sliding window which slides across the data stream. The principle of CCSC is that it modifies the current syndrome, which is representative of the data presently in the window, to construct the syndrome of the data in the window after a shift of a new bit in and the oldest bit out. This allows for single bit time updates of the syndrome for the received word in the window, as the window slides across the data stream [3].

The decoder delay register will at some arbitrary time, i , contain an assumed codeword with the checkbits at the distributed F and (C) bit locations. This received word can be re-arranged into systematic form. As well, the received word in delay buffer one bit time later ($i+1$) can also be logically re-arranged into systematic form. CCSC uses the differences between the shift of the systematic form polynomial at time i and the systematic form polynomial at time $i+1$ to methodically compensate an existing syndrome at time i to derive the syndrome at time $i+1$. The details of this method for the DS3-FEC are now presented.

$R_{dist}(x)$ is a DS3-FEC codeword polynomial with checkbits at distributed locations in the receiver delay register. Thus for time i and for time $i+1$, one can write:

$$R_{dist_i}(x) = b_{1359} x^{1359} + b_{1358} x^{1358} + b_{1357} x^{1357} + \dots + b_1 x + b_0 \quad (4.1)$$

$$R_{dist_{i+1}}(x) = b_{1358} x^{1359} + b_{1357} x^{1358} + b_{1356} x^{1357} + \dots + b_0 x + b_{NEW} \quad (4.2)$$

The coefficients of $R_{dist_i}(x)$ are the 1360 bits in the decoder delay register at an arbitrary time i , where b_{1359} is the oldest bit and b_0 is the bit most recently added. The coefficients of $R_{dist_{i+1}}(x)$ are the bits in the delay register at time $i+1$, i.e. after a single shift. Note that the oldest bit from time i , b_{1359} , has been discarded, and that a new bit, b_{NEW} , has been shifted in to the least significant position of the register.

As stated in Section 2.3, the syndrome, $S(x)$, for a cyclic code in systematic form is computed as:

$$S(x) = R(x) \bmod G(x) \quad (4.3)$$

where $R(x)$ is the polynomial representation of the received word in systematic form, and $G(x)$ is the irreducible generator polynomial. $R_{dist_i}(x)$ can be put into systematic form, $R_{sys_i}(x)$ by rearranging the distributed checkbits into systematic positions, so that all data bits are contiguous followed by all the checkbits, and the most significant data bit is the coefficient of the highest order term in $R_{sys_i}(x)$. Because the overall parity bit used for DED is not part of the syndrome decoding process, it is not included in $R_{sys_i}(x)$. Hence there are 1359 terms in $R_{sys_i}(x)$, the highest order term being x^{1358} . The polynomial $R_{sys_{i+1}}(x)$ is formed in a similar fashion from the re-arrangement of $R_{dist_{i+1}}(x)$ into systematic form.

$R_{sys_{i+1}}(x)$ can be written in terms of $R_{sys_i}(x)$ as follows:

$$R_{sys_{i+1}}(x) = xR_{sys_i}(x) + D(x) \quad (4.4)$$

where $D(x)$ is a *difference polynomial* which accounts for the differences between

$R_{sys_{i+1}}(x)$ and the shift of $R_{sys_i}(x)$. For the DS3-FEC code, $D(x)$ is:

$$D(x) = \sum_{j=1}^{25} d_j \cdot x^{p_j} \quad (4.5)$$

where p_j is the order of the term at which the difference occurs and d_j is the modulo-2 sum of the appropriate pairs of bits in $R_{dist_i}(x)$. The values for d_j and p_j for the 25 differences between $R_{sys_{i+1}}(x)$ and $xR_{sys_i}(x)$ in the DS3-FEC code are given in Table 4.1. There are 25 differences because for each shift of the word in the receiver delay register, 12 bits which were considered as FEC checkbits at time i move to data bit positions at time $i+1$, 12 bits which were in data bit positions at time i move to checkbit positions at time $i+1$, and the oldest bit is shifted out of the register.

Table 4.1: Coefficients and exponents in $D(x)$ for the DS3-FEC code

j	d_j	p_j	j	d_j	p_j	j	d_j	p_j
1	r_{1359}	1359	10	$r_{338} \oplus r_{339}$	347	19	$r_{764} \oplus r_{848}$	6
2	$r_{1273} \oplus r_{1274}$	1274	11	$r_{253} \oplus r_{254}$	263	20	$r_{594} \oplus r_{763}$	5
3	$r_{1103} \oplus r_{1104}$	1105	12	$r_{168} \oplus r_{169}$	179	21	$r_{424} \oplus r_{593}$	4
4	$r_{1018} \oplus r_{1019}$	1021	13	$r_{83} \oplus r_{84}$	95	22	$r_{339} \oplus r_{423}$	3
5	$r_{933} \oplus r_{934}$	937	14	$r_{1274} \oplus r_{NEW}$	11	23	$r_{254} \oplus r_{338}$	2
6	$r_{848} \oplus r_{849}$	853	15	$r_{1104} \oplus r_{1273}$	10	24	$r_{169} \oplus r_{253}$	1
7	$r_{763} \oplus r_{764}$	769	16	$r_{1019} \oplus r_{1103}$	9	25	r_{168}	0
8	$r_{593} \oplus r_{594}$	600	17	$r_{934} \oplus r_{1018}$	8			
9	$r_{423} \oplus r_{424}$	431	18	$r_{849} \oplus r_{933}$	7			

The syndrome for the data in the receive delay buffer at time $i+1$ can therefore be found by dividing each side of (4.4) by $G(x)$ and taking the remainder:

$$S_{i+1}(x) = xR_{sys_i}(x) \bmod G(x) + D(x) \bmod G(x) \quad (4.6)$$

The right hand side of (4.6) can be simplified using the following relation for full length cyclic codes:

$$xS(x) \bmod G(x) = R^{(1)}(x) \bmod G(x) \quad (4.7)$$

where $R^{(1)}(x)$ is a cyclic shift of $R(x)$, which may be written as:

$$R^{(1)}(x) = r_{n-1}(x^{n+1}) + xR(x) \quad (4.8)$$

where r_{n-1} is the MSB of the full length codeword. The DS3-FEC code is a shortened code. However, all codewords in a $(n-l, k-l)$ shortened code are derived from the subset of codewords in the parent code where the l leading bits are equal to zero (Section 2.3). Therefore, the MSB of the corresponding full length codeword in the parent code for any codeword belonging to the shortened code is zero. Hence $R^{(1)}(x) = xR(x)$. Thus for shortened codes, (4.7) becomes:

$$xS(x) \bmod G(x) = xR(x) \bmod G(x) \quad (4.9)$$

The left side of (4.9) is simply the current value of the syndrome clocked once in the $+G(x)$ register. Combining (4.9) and (4.6):

$$S_{i+1}(x) = xS_i(x) \bmod G(x) + D(x) \bmod G(x) \quad (4.10)$$

Finally, $D(x) \bmod G(x)$, i.e. the remainders of the division of each of the x^{p_j} terms in $D(x)$ by $G(x)_{DS3}$, is a fixed polynomial which can be precomputed. The results are given in Table 4.2:

Table 4.2: Remainders of DS3 Difference Polynomial Term Division by $G(x)$

dj	pj	$x^{p_j} \bmod G(x)$ (MSB First)	dj	pj	$x^{p_j} \bmod G(x)$ (MSB First)
d1	1359	01000001101	d14	11	00000000101
d2	1274	00011010100	d15	10	10000000000
d3	1105	01111001110	d16	9	01000000000
d4	1021	11111110111	d17	8	00100000000
d5	937	01010001001	d18	7	00010000000
d6	853	11111100100	d19	6	00001000000
d7	769	10111111011	d20	5	00000100000
d8	600	11011010010	d21	4	00000010000
d9	431	01110100111	d22	3	00000001000
d10	347	10100110000	d23	2	00000000100
d11	263	11001001011	d24	1	00000000010
d12	179	00101011100	d25	0	00000000001
d13	95	00010100010			

Each of the $x^{p_j} \bmod G(x)$ in Table 4.2 is a polynomial of less than degree 11 with coeffi-

cient d_j which is added to $xS_i(x) \bmod G(x)$ to obtain the syndrome for the data in the delay buffer at time $i+1$, i.e:

$$S_{i+1}(x) = xS_i(x) \bmod G(x) + d1(x^9+x^3+x^2+1) + d2(x^7+x^6+x^4+x^2) + \dots + d24(x) + d25$$

By grouping terms of equal polynomial order, a single compensation function can therefore be created for each term in $xS_i(x) \bmod G(x)$. These compensation functions are listed in Table 4.3 for the term of $xS(i,x) \bmod G(x)$ which each modifies:

Table 4.3: DS3-FEC CCSC Compensation Functions

Term	Compensation Function
x^0	$d1 \oplus d4 \oplus d5 \oplus d7 \oplus d9 \oplus d11 \oplus d14 \oplus d25$
x^1	$d3 \oplus d4 \oplus d7 \oplus d8 \oplus d9 \oplus d11 \oplus d13 \oplus d24$
x^2	$d1 \oplus d2 \oplus d3 \oplus d6 \oplus d9 \oplus d12 \oplus d14 \oplus d23$
x^3	$d1 \oplus d3 \oplus d4 \oplus d5 \oplus d7 \oplus d11 \oplus d12 \oplus d22$
x^4	$d2 \oplus d4 \oplus d7 \oplus d8 \oplus d10 \oplus d12 \oplus d21$
x^5	$d4 \oplus d6 \oplus d7 \oplus d9 \oplus d10 \oplus d13 \oplus d20$
x^6	$d2 \oplus d3 \oplus d4 \oplus d6 \oplus d7 \oplus d8 \oplus d11 \oplus d12 \oplus d19$
x^7	$d2 \oplus d3 \oplus d4 \oplus d5 \oplus d6 \oplus d7 \oplus d8 \oplus d9 \oplus d13 \oplus d18$
x^8	$d3 \oplus d4 \oplus d6 \oplus d7 \oplus d9 \oplus d10 \oplus d12 \oplus d17$
x^9	$d1 \oplus d3 \oplus d4 \oplus d5 \oplus d6 \oplus d8 \oplus d9 \oplus d11 \oplus d16$
x^{10}	$d4 \oplus d6 \oplus d7 \oplus d8 \oplus d10 \oplus d11 \oplus d15$

Therefore, in summary, the syndrome for the data in the delay register at time $i+1$ is computed by clocking the syndrome at time i in the standard $+G(x)$ register and then adding each compensation function listed in Table 4.3, recalling that each d_j maps into a delay register position, p_j , as given in Table 4.2, to the appropriate term in the syndrome register.

It is interesting to note that the newest incoming data bit is not clocked serially into the $+G(x)$ register but rather is combined with the delayed data bits ($d14$ in Table 4.1) directly to form compensation functions. Thus the CCSC syndrome register does not appear with an explicit serial input.

4.3.2 The Process of CCSC Decoding for the DS1-FEC Code

A similar procedure as that undertaken in Section 4.3.2 can be used to develop the compensation logic required to modify the syndrome of the data in the decoder delay register

at time i , to compute the syndrome for the data word in the delay buffer at time $i+1$, after the oldest bit is discarded and a new bit is included. The polynomials representing the contents of the decoder delay buffer at times i and $i+1$ respectively are:

$$R_{dist_i}(x) = b_{2315} x^{2315} + b_{2314} x^{2314} + b_{2313} x^{2313} + \dots + b_1 x + b_0 \quad (4.11)$$

$$R_{dist_{i+1}}(x) = b_{2314} x^{2315} + b_{2313} x^{2314} + b_{2312} x^{2313} + \dots + b_0 x + b_{NEW} \quad (4.12)$$

These polynomials represent DS1-FEC codewords with checkbits at distributed positions. The polynomials $R_{sys_i}(x)$ and $R_{sys_{i+1}}(x)$ can be constructed by re-arranging (4.11) and (4.12) into systematic form. The DS1-FEC does not have an overall parity bit for DED, so the number of terms in the systematic form polynomials is exactly the same as for the distributed form polynomials. Equation (4.10) relates the syndrome of the data in the window at time $i+1$ in terms to the syndrome at time i , and $D(x)$ is given by (4.5). The values of d_j and p_j for each term in $D(x)$ for the DS1-FEC code are given in Table 4.4.

Table 4.4: Coefficients and exponents in $D(x)$ for the DS1-FEC code

j	d_j	p_j	j	d_j	p_j	j	d_j	p_j
1	r_{2314}	2316	10	$r_{577} \oplus r_{578}$	588	19	$r_{1157} \oplus r_{1349}$	6
2	$r_{2121} \oplus r_{2122}$	2124	11	$r_{384} \oplus r_{385}$	396	20	$r_{964} \oplus r_{1156}$	5
3	$r_{1928} \oplus r_{1929}$	1932	12	$r_{191} \oplus r_{192}$	204	21	$r_{771} \oplus r_{963}$	4
4	$r_{1735} \oplus r_{1736}$	1740	13	$r_{2315} \oplus r_{NEW}$	12	22	$r_{578} \oplus r_{770}$	3
5	$r_{1542} \oplus r_{1543}$	1548	14	$r_{2122} \oplus r_{2314}$	11	23	$r_{385} \oplus r_{577}$	2
6	$r_{1349} \oplus r_{1350}$	1356	15	$r_{1929} \oplus r_{2121}$	10	24	$r_{192} \oplus r_{384}$	1
7	$r_{1156} \oplus r_{1157}$	1164	16	$r_{1736} \oplus r_{1928}$	9	25	r_{191}	0
8	$r_{963} \oplus r_{964}$	972	17	$r_{1543} \oplus r_{1735}$	8			
9	$r_{770} \oplus r_{771}$	780	18	$r_{1350} \oplus r_{1542}$	7			

Note that bit 2315 is the MSB in the receiver buffer at time i , and that bit "NEW" is the newly received bit at time $i+1$ (used to compute d_{13}). The remainder of the division of each x^{p_j} term in $D(x)$ by $G(x)_{DS1}$ is a fixed polynomial, of less than degree 12, and can be

precomputed. These are given in Table 4.5.

Table 4.5: Remainder of DS1 Difference Polynomial Terms Division by $G(x)_{DS1}$

dj	pj	$x^{pj} \bmod G(x)$ (MSB First)	dj	pj	$x^{pj} \bmod G(x)$ (MSB First)
d1	2316	100011011101	d14	11	100000000000
d2	2124	011000000111	d15	10	010000000000
d3	1932	101011011111	d16	9	001000000000
d4	1740	011110011010	d17	8	000100000000
d5	1548	110010001100	d18	7	000010000000
d6	1356	110100101000	d19	6	000001000000
d7	1164	111010100111	d20	5	000000100000
d8	972	110001110001	d21	4	000000010000
d9	780	010111100111	d22	3	000000001000
d10	588	110100011011	d23	2	000000000100
d11	396	100101010100	d24	1	000000000010
d12	204	010000001010	d25	0	000000000001
d13	12	000001010011			

Finally, by grouping terms of equal polynomial order in Table 4.5, the compensation function for each term in $xS_i(x)$ can be developed. These are given in Table 4.6.

Table 4.6: DS1-FEC CCSC Compensation Functions

Term	Compensation Function
x^0	$d1 \oplus d2 \oplus d3 \oplus d7 \oplus d8 \oplus d9 \oplus d10 \oplus d13 \oplus d25$
x^1	$d2 \oplus d3 \oplus d4 \oplus d7 \oplus d9 \oplus d10 \oplus d12 \oplus d13 \oplus d24$
x^2	$d1 \oplus d2 \oplus d3 \oplus d5 \oplus d7 \oplus d9 \oplus d1 \oplus d23$
x^3	$d1 \oplus d3 \oplus d4 \oplus d5 \oplus d6 \oplus d10 \oplus d12 \oplus d22$
x^4	$d1 \oplus d3 \oplus d4 \oplus d8 \oplus d10 \oplus d11 \oplus d13 \oplus d21$
x^5	$d6 \oplus d7 \oplus d8 \oplus d9 \oplus d20$
x^6	$d1 \oplus d3 \oplus d8 \oplus d9 \oplus d11 \oplus d13 \oplus d19$
x^7	$d1 \oplus d3 \oplus d4 \oplus d5 \oplus d7 \oplus d9 \oplus d18$
x^8	$d4 \oplus d6 \oplus d9 \oplus d10 \oplus d11 \oplus d17$
x^9	$d2 \oplus d3 \oplus d4 \oplus d7 \oplus d16$
x^{10}	$d2 \oplus d4 \oplus d5 \oplus d6 \oplus d7 \oplus d8 \oplus d9 \oplus d10 \oplus d12 \oplus d15$
x^{11}	$d1 \oplus d3 \oplus d5 \oplus d6 \oplus d7 \oplus d8 \oplus d10 \oplus d11 \oplus d14$

Thus the syndrome at time $i+1$ can be obtained by clocking the syndrome at time i in the $+G(x)$ register and adding the corresponding compensation functions from Table 4.6 to each term of the result.

4.3.3 Codeword Delay Buffer

We now consider the circuits to implement CCSC (as above) for the DS1 and DS3 signal formats.

The schematic diagram of the Codeword Delay Buffer is shown in Figure 4.2:

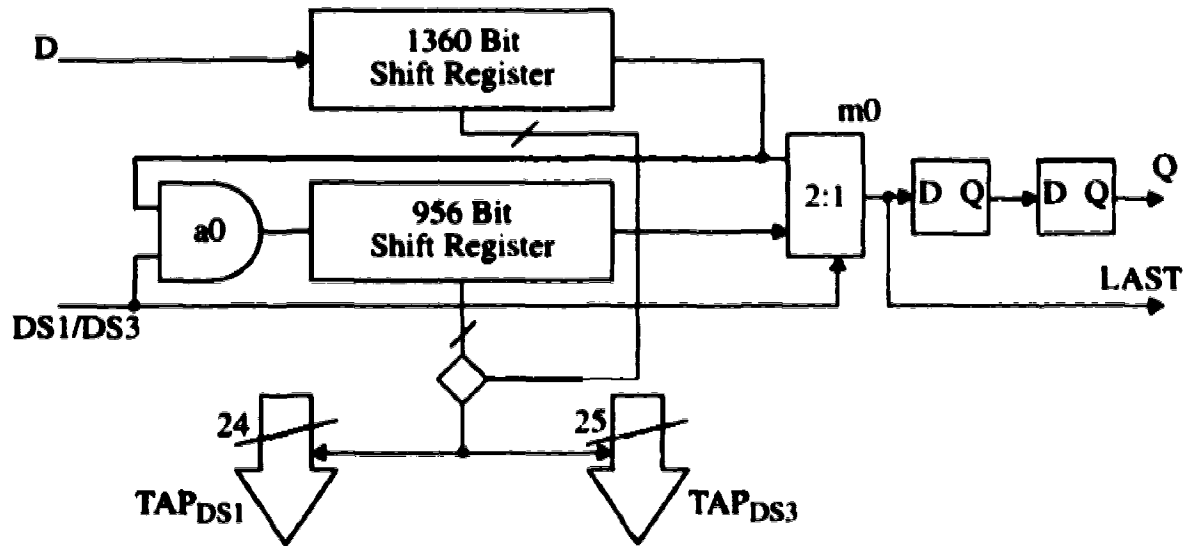


Figure 4.2: Codeword Delay Buffer Block Diagram

The following are the inputs to the Codeword Length Delay Buffer:

- *D*: Serial data input
- *DS1/DS3*: Mode select

and the outputs from the module are the following:

- *LAST*: Input data delayed by exactly one codeword time
- *Q*: Input data delayed by one codeword time plus two bit time delays. The additional delays compensate for two pipeline delays introduced by the FAU and ECU to de-glitch a number of control signals.
- *TAP_{DS1}*: Delayed data signals at various points along the 1360 bit and 956 bit shift registers used to generate the DS1-FEC CCSC compensation functions
- *TAP_{DS3}*: Delayed data at various points along the 1360 bit shift register used to generate the DS3-FEC CCSC compensation functions

The codeword delay buffer is composed of 2316 D-flip flops. This is the length of a DS1-FEC codeword. The size of the delay register is re-configurable for DS1 or DS3 mode. In DS3 mode only the first 1360 flip flops are used. A 2:1 MUX selects the length of the delay buffer. The MUX is controlled by the *DS1/DS3* mode signal. When it is low, the output of the 1360th flip flop in the delay chain is selected as the end of the delay register. If DS1 is high then the output of the 2316th flip flop in the shift register chain is selected as the delay register output. Note also in DS3 mode (*DS1*=0) the AND gate, *a0*, is turned off. This prevents data from being shifted into this unused portion of the delay register,

thereby reducing power consumption.

As presented in Sections 4.3.1 and 4.3.2, the delayed data at various positions along the shift register is used to generate the CCSC compensation terms for single bit time syndrome compensation. In DS3 mode, the set tap positions is $TAP_{DS3} = \{83, 84, 168, 169, 253, 254, 338, 339, 423, 424, 593, 594, 763, 764, 848, 849, 933, 934, 1018, 1019, 1103, 1104, 1273, 1274, 1359, \text{and NEW}\}$, where the number indicates the Q-output of that particular flip flop (where flip flop #0 is the first flip flop in the delay chain), and "NEW" is the new bit at the input to the delay register. In DS1 mode, the set of tap positions is $TAP_{DS1} = \{191, 192, 384, 385, 577, 578, 770, 771, 963, 964, 1156, 1157, 1349, 1350, 1542, 1735, 1736, 1928, 1929, 2121, 2122, 2314, 2315, \text{and NEW}\}$. These taps are grouped to form the two buses.

4.3.4 DS1 and DS3 CCSC Compensation Terms Generation

The circuitry for generating the CCSC compensation functions for the DS1-FEC and DS3-FEC codes from the delayed data on TAP_{DS1} or TAP_{DS3} consists of five main blocks shown in Figure 4.3:

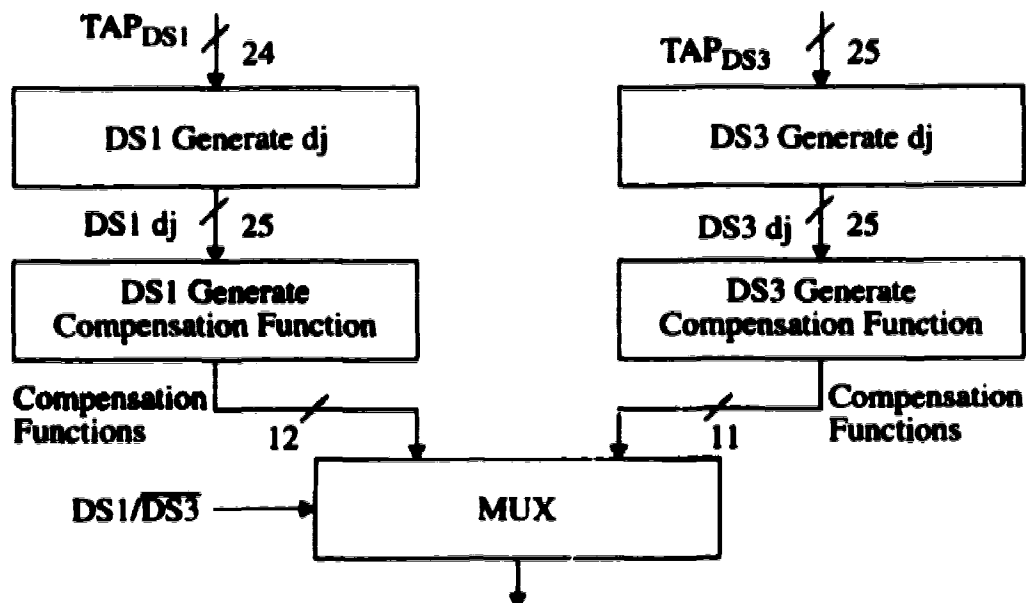


Figure 4.3: CCSC Compensation Term Generation Circuitry

The "DS1 Generate dj" and "DS3 Generate dj" blocks consist of 2-input XOR gates which combine the delayed data to create the difference terms (d_j 's) presented in Tables 4.4 and 4.1 respectively. The difference terms are then routed to the "DS1 Generate Compensation Function" and "DS3 Generate Compensation Function" modules. These blocks consist of multi-input XOR gates (from 5 to 10 inputs) which combine the d_j 's to create the DS1-FEC and DS3-FEC compensation terms presented in Tables 4.6 and 4.3

respectively.

Clearly the compensation functions for DS1-FEC and DS3-FEC have nothing in common, i.e. the difference terms (d_j) are computed from different delayed data points and each compensation function is derived from different d_j 's. Therefore only one set of them is routed to the $+G(x)$ register depending on the mode of operation. The DS1 and DS3 compensation functions are selected by a 12-bit 2:1 MUX bank as shown in Figure 4.4. The MUX is controlled by the $DS1/\overline{DS3}$ mode line to select the appropriate compensation functions to connect to the $+G(x)$ syndrome register.

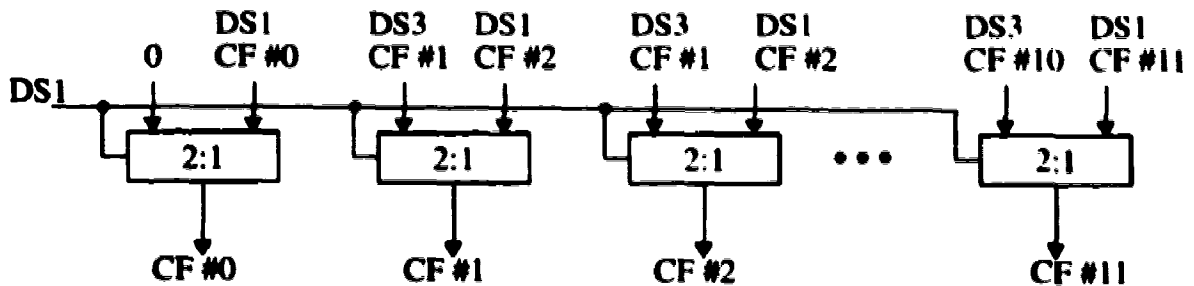


Figure 4.4: CCSC Compensation Function Selector MUX

Note that there are 12 DS1-FEC compensation terms and only 11 DS3-FEC compensation functions. The unused compensation term line in DS3 mode is hard-wired to zero.

4.3.5 CCSC ($+G(x)$) Register

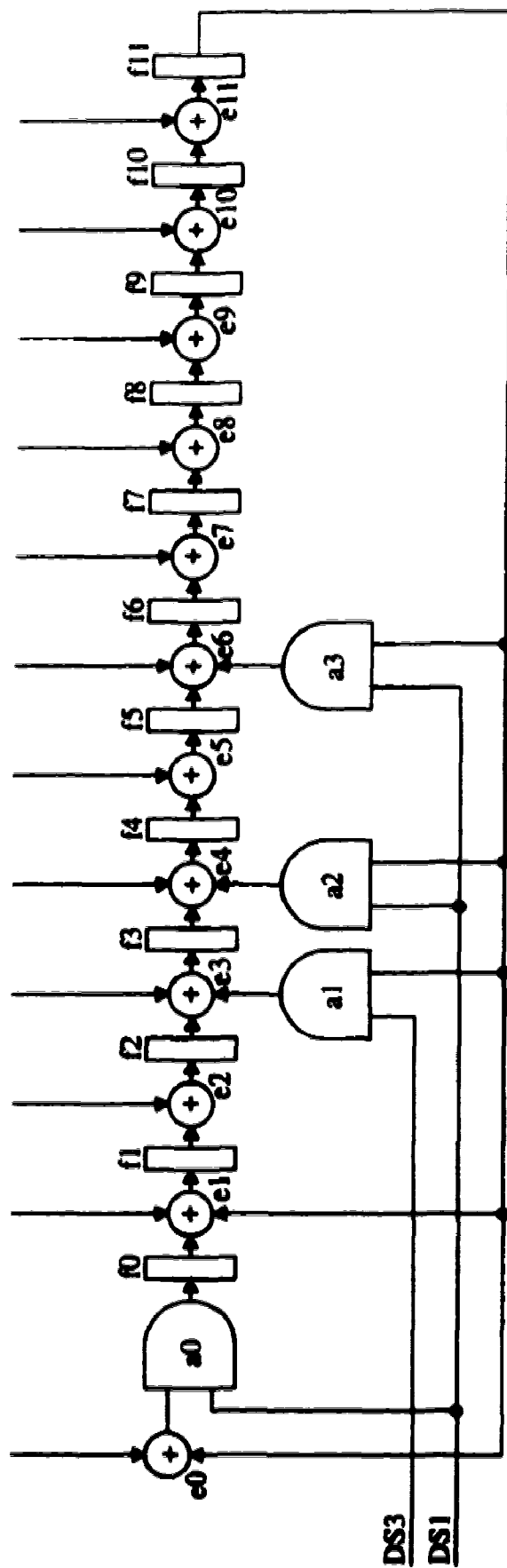
The CCSC ($+G(x)$) register implements the division of the current syndrome at time i by the code generator polynomial, $G(x)$, and the addition of the compensation functions to the result. The CCSC register is shown in Figure 4.5. The following are the inputs to the module (in addition to the system clock and the asynchronous reset not shown in Figure 4.5):

- **DS1, DS3:** Mode select lines to configure the register architecture for DS1 or DS3 mode of operation
- **CCSC Compensation Functions:** The CCSC compensation functions for either DS1 or DS3 mode of operation as derived in the CCSC Compensation Function Generation block.

The following are the module outputs:

- **SYNDROME:** Syndrome for the received word currently in the codeword length delay buffer. (12 bits DS1; 11 bits DS3)
- **ZERO:** Asserted HIGH if the all bits of the syndrome register are zero. Asserted LOW in the syndrome register contents contain any non-zero bits.

CCSC Compensation Functions



4

DS1: 1	x	x ²	x ³	x ⁴	x ⁵	x ⁶	x ⁷	x ⁸	x ⁹	x ¹⁰	x ¹¹	x ¹²
DS3:	1	x	x ²	x ³	x ⁴	x ⁵	x ⁶	x ⁷	x ⁸	x ⁹	x ¹⁰	x ¹¹

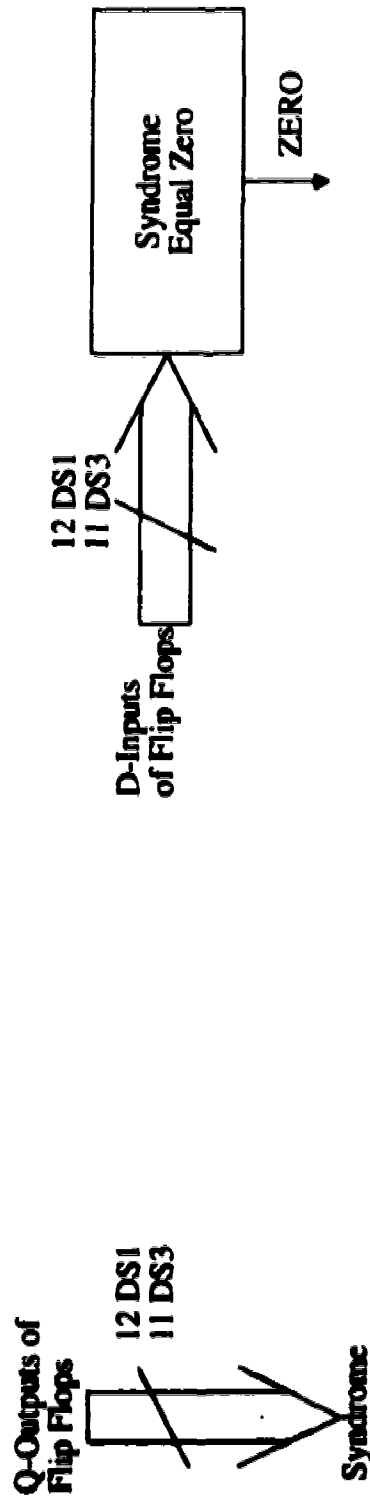


Figure 4.5: CCSC Syndrome Register (Divide by $G(x)$)

Ignoring the compensation functions for a moment, the CCSC is a linear feedback shift register which implements division by the code generator polynomial, $G(x)$. The register is re-configurable to implement $G(x)_{DS1}$ or $G(x)_{DS3}$ depending on the mode of operation. Only the 11 left-most flip flops are used in DS3 mode. In this case, the AND gate, $a0$, is turned off, forcing the output of $f0$ to zero. This removes the effect of $f0$ on the feedback connection at $e1$. In DS1 mode, $a1$ is transparent, connecting the feedback signal from $e0$ to the D-input of $f0$.

The feedback points in the register are controlled by AND gates $a1$, $a2$, and $a3$. In DS1 mode, $f0$ is the least significant flip flop in the register, so feedback is necessary at the D-inputs of $f0$, $f1$, $f4$, and $f6$ to implement $G(x)_{DS1}$. In DS3 mode, $f1$ is the least significant stage in the register so feedback taps are required at the D-inputs to $f1$ and $f3$ to implement $G(x)_{DS3}$. In both cases, the feedback signal comes from the output of $f11$. In DS1 mode, $a2$, and $a3$ enable the feedback taps at the inputs to flip flops $f4$ and $f6$ while $a1$ is turned off, breaking the feedback tap at the input to $f3$. In DS3 mode, $a2$ and $a3$ are turned off, breaking their feedback connections, while $a1$ is transparent, connecting the feedback signal to the XOR gate at the D-input of $f3$.

Note that the compensation functions are derived from the delay register contents at time i , but must be added to the current syndrome once it has been clocked in the $+G(x)$ register (i.e. at time $i+1$). One solution to this problem would be to route the 12 compensation function lines to 1-bit delay elements (D-Flip Flops) before adding them to the terms in CCSC syndrome register. However, a more efficient scheme is to add the compensation functions to the signals at the D-inputs of the flip flops in the syndrome register. This is valid as the D-inputs carry the values which will be transferred to the Q-outputs of the flip-flops on the next rising clock edge. Thus the need for 12 delay elements is removed. This the method is implemented in Figure 4.5.

The “*Syndrome=0*” module examines the values at the D-inputs to the flip flops in the CCSC register (after the compensation functions have been added in), to determine if the syndrome is equal to zero. In DS1 mode all 12 bits of syndrome information are considered. In DS3 mode, only the 11 left-most bits in the syndrome register need to be examined. The “*Syndrome=0*” module thus implements the following boolean function:

$$Z = \overline{s11 + s10 + s9 + s8 + s7 + s6 + s5 + s4 + s3 + s2 + s1 + (DS1)(s0)} \quad (4.13)$$

where s_j is the D-input of flip flop j in Figure 4.5. This function, Z , is routed to a D-flip flop. Thus the signal on the ZERO output line changes simultaneously with the syndrome information it represents.

Note that for the CCSC process to work correctly, the system must be initialized so the syndrome in the CCSC syndrome register is initially representative of the data in the delay buffer. This can be achieved by purging the delay register of its contents and filling it with zeros, and clearing all of the flip flops on the CCSC syndrome register to zero. With this initial condition satisfied on power-up, the syndrome will thus always be representative of the received word, regardless of the error weight or frame misalignment. Only an error on-chip between the delay buffer input and the CCSC compensation function generator would cause a failure of the CCSC approach.

4.4 Parity Computation Register

This module computes the parity of the data bits in the 1360 bit delay buffer in DS3 mode for double error detection purposes and for use in the frame alignment process. Recall that in DS3 mode, a parity bit was computed to make the overall parity of each DS3-FEC codeword even, and this bit is carried in bit position 1275 of the DS3-FEC codeword format. The parity of the data in the delay buffer at an arbitrary time i :

$$P_i = r_0 \oplus r_1 \oplus r_2 \oplus \dots \oplus r_{1359} \quad (4.14)$$

where $r_{0..1359}$ are the values of the bits at the Q-outputs of the 1360 flip flops which make up the delay register. (Flip flop 0 is at the input to the delay register and flip flop 1359 is the output).

The mechanism for computing the parity of the data in the delay buffer is similar in principle to CCSC. The parity computation circuit computes the parity of the data in the window at time $i+1$, i.e. after including the next bit and discarding the oldest bit from the register, by modifying the parity from the data in the window at time i . At time i , if P_i is the parity of the data in the delay register, r_{NEW} is the bit which is charging up the input to the delay buffer (but has not been clocked in yet), and r_{1359} is the bit at the delay register output, then the parity of the data in the delay buffer after r_{NEW} is shifted in and r_{1359} is discarded is:

$$P_{i+1} = P_i \oplus r_{NEW} \oplus r_{1359} \quad (4.15)$$

Figure 4.6 shows the circuit which implements (4.15).

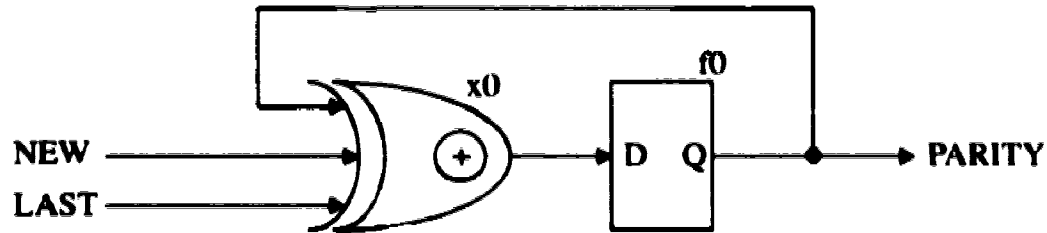


Figure 4.6: Received Word Parity Computation Circuit

The inputs to this circuit in addition to the system clock, cp , and asynchronous reset, cd (not shown) are the following:

- **NEW**: the value of the bit which will be clocked into the delay register on the next rising clock edge (r_{NEW}).
- **LAST**: the value of the bit at the output of the delay buffer (without pipeline delay compensation), i.e. the bit at the same index into the received word as r_{NEW} except delayed by one full codeword time (r_{1359})

The only output is the parity of the data in the delay buffer, **PARITY**.

The D-flip flop, $f0$, maintains the value of the parity for the received word at time i on its Q-output. This value is fed back to one input of the 3-input XOR gate, $x0$. The values of the newest and oldest bits are routed to the other inputs of $x0$ to implement (4.15). The output of $x0$ is connected to the D-input of $f0$ and will be transferred to its Q-output pin on the next rising clock edge. Thus the parity on the Q-output of $f0$ will always be representative of the data contained in the delay buffer.

This method for continuously computing the parity of the data in the delay register requires that the parity be initially representative of the data. This is achieved at reset time when the delay buffer is purged and set to contain all zeros. The D-flip flop, $f0$ is also cleared to zero, and thus the parity is initially representative of the data in the sliding window.

Once the syndrome and the parity for the received word have been computed, they are routed to the FAU and the ECU.

4.5 Frame Alignment Unit (FAU)

The FAU implements the algorithms for reframing and out-of-frame detection using FEC syndromes which were outlined in Sections 2.4.1 and 2.4.2.

The operation of the FAU is identical in both DS1 and DS3 modes except for the definition of a zero syndrome. In DS1 mode, the syndrome is zero only if all 12 bits in the

CCSC syndrome register are identically zero. In DS3 mode, both the syndrome and the parity of the received word are used to drive the frame alignment process. In this case, a "zero syndrome" means that the 11 DS3 syndrome bits are equal to zero and the parity of the received word is simultaneously zero. Thus the definition of a zero syndrome for the framing logic is given by:

$$ZS = (12 \text{ bit Syndrome}=0)(DS1) + (11 \text{ bit Syndrome}=0)(\text{Parity}=0)(DS3) \quad (4.16)$$

Note that the FAU does not require the actual 12/11-bit value of the syndrome. All the framing logic is concerned with is whether the syndrome is zero or non-zero. The value on the ZERO line of the CCSC syndrome register and the value of the received word parity are combined as in (4.15) and routed to the FAU.

The block diagram for the FAU is shown in Figure 4.7:

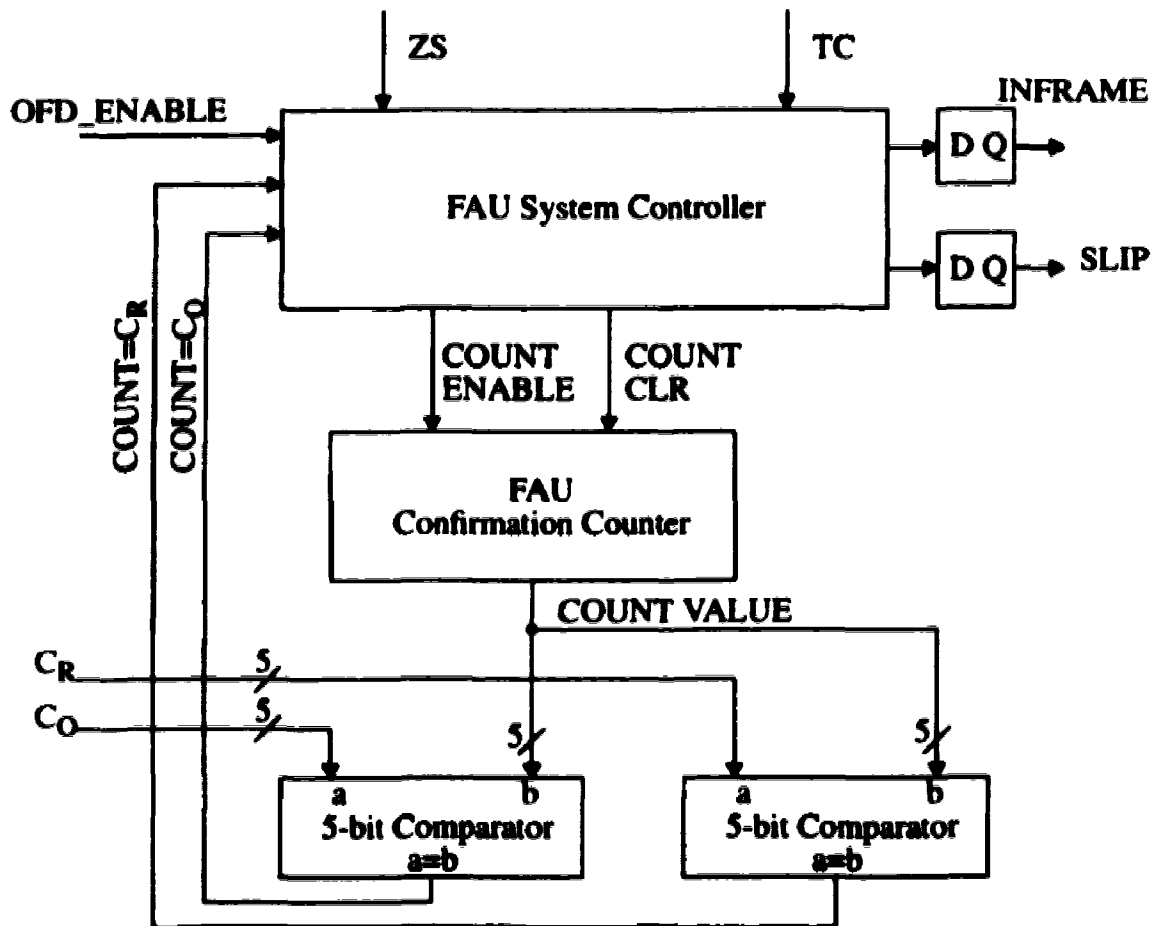


Figure 4.7: Block Diagram of FAU module

The inputs to this module, in addition to the system clock and the asynchronous reset (not shown) are the following:

- **ZS**: Asserted LOW if the syndrome is equal to zero.

STATE A: OOF and SEARCHING FOR A FRAME CANDIDATE

```
IF (ZS is TRUE) {  
  IF (FAU Confirmation Counter = CR) {  
    declare INFRAME ;  
    STATE = C ;  
  } ELSE {  
    increment FAU Confirmation Counter  
    STATE = B ;  
  }  
} ELSE {  
  SLIP one bit position ;  
}
```

STATE B: OOF and CONFIRMING

```
Wait for Codeword end time .  
IF (ZS is TRUE) {  
  IF (FAU Confirmation Counter = CR) {  
    declare INFRAME ;  
    STATE = C ;  
  } ELSE {  
    increment FAU Confirmation Counter ;  
    STATE = B ;  
  }  
} ELSE {  
  Clear FAU Confirmation Counter ;  
  SLIP one bit position ;  
  STATE = A ;  
}
```

STATE C: INFRAME and PERFORMING OOF DETECTION

```
IF (OFD_ENABLE is de-asserted) {  
  STATE = C ;  
} ELSE {  
  Wait for Codeword end Time ,  
  IF (ZS is TRUE) {  
    clear FAU Confirmation Counter ;  
    STATE = C ;  
  } ELSE {  
    IF (FAU Confirmation Counter = CO) {  
      declare OOF ;  
      clear FAU Confirmation Counter ;  
      SLIP one bit position ;  
      STATE = A ;  
    } ELSE {  
      increment FAU Confirmation Counter ,  
      STATE = C ;  
    }  
  }  
}
```

Figure 4.8: Pseudocode for the FAU System Controller

- C_R : Chip level input; 5-bit Reframe confirmation threshold value
- C_O : Chip level input; 5-bit OOF Detection confirmation threshold value
- TC (Terminal Count): Indicates the last bit of a codeword has clocked in to the FEC decoder, i.e. it is the codeword end time.

and the outputs from the FAU are the following:

- ***INFRAME***: Asserted HIGH when the framing logic has established frame alignment. Asserted LOW for when the frame alignment is lost
- ***SLIP***: The SLIP signal is asserted for one bit time for each bit position which is rejected as a candidate for the frame boundary during the search phase of the reframe process. It is de-asserted during the reframe confirmation interval and when the system is in-frame.

The FAU consists of four main components, shown in Figure 4.7., The confirmation counter, two five bit equivalence comparators, and the system controller. The confirmation counter is a 5-bit up counter with count enable and synchronous clear. Each 5-bit comparator compares two 5-bit numbers. Its output is asserted high only if the two numbers are the same (Appendix A). The system controller is a logic minimized state machine which implements the re-frame and out-of-frame detection algorithms outlined in Sections 2.4.1 and 2.4.2. The pseudocode for this state machine is given in Figure 4.8.

When in State A, the FAU is out-of-frame and searching for a frame candidate. The syndrome at the current bit position is examined and tested for zero. If it is non-zero, the position is rejected as a frame candidate and the FAU circuitry slips to the next bit position. The timing of control signals in the FEC decoder is controlled by an event sequencer (Section 4.8) similar to that found in the encoder (Section 3.9). An OOF declaration will always occur exactly one bit time after the system counter in the sequencer reaches its terminal count, i.e. count equal to zero. The *SLIP* output asserts the synchronous clear input on this state counter, keeping it at zero until a position which decodes a zero syndrome is found. When a zero syndrome position is located, *SLIP* is de-asserted so that the state counter is free count. This establishes the sequencing so that the terminal count of the state counter occurs at the assumed codeword boundary. Then, the value of the FAU confirmation counter is checked for a match with C_R via one of the 5-bit comparators. At this point this can only happen if C_R is equal to zero (i.e. no confirmations). If this is the case then in-frame is declared immediately (State C). Otherwise the confirmation counter is incremented, by asserting the *COUNT ENABLE* signal and the confirmation process begins (State B).

In State B, the FAU waits for one full codeword time from the assumed codeword bound-

ary position defined by the frame candidate before examining the syndrome again. The codeword end time is defined by the assertion of the TC input. If the syndrome is zero, the value of the FAU confirmation counter is checked via the 5-bit comparator to see if it is equal to C_R . If it is not then further confirmations are required before the in-frame state can be declared. The FAU confirmation counter is incremented by asserting the *COUNT ENABLE* signal and the framing circuitry waits for another full codeword time before examining the syndrome again. When the counter value reaches C_R , in-frame is declared (State C), the confirmation counter is cleared via the *COUNT CLR* line, and the OOF detection process begins. If a non-zero syndrome is decoded at the assumed codeword end time before the counter reaches C_R , the candidate position is rejected, the counter is cleared, and the search process resumes (State A).

The system is in-frame in State C. In this state, the framing logic is constantly testing for OOF detection by examining the syndrome at codeword end times. If a non-zero syndrome is seen, the value of the FAU confirmation counter is compared to C_O via the second 5-bit comparator. If the value is less than C_O , the in-frame state is maintained but the counter is incremented. If non-zero syndromes persist at the codeword end times then the counter will eventually reach C_O , OOF will be declared, and the search State A entered. However, if a zero syndrome is decoded at the codeword end time before the counter reaches C_O , the counter is cleared and the in-frame state is maintained. Note that the OOF detection process can be disabled by de-asserting the *OFD_ENABLE* signal. In this case, syndrome examination at the codeword end times is inhibited and the current position which the FAU believes is the frame boundary is maintained, even if frame loss occurs.

Note that only one counter is used for the reframe confirmation process and the OOF detection process as these states are mutually exclusive. Also note that the *SLIP* and *INFRAME* signals are routed to D-flip flops to remove decoding glitches as they drive modules of the FEC decoder outside of the FAU. This introduces one pipeline delay in the system as the *SLIP* and *INFRAME* lines must change coincident with the first bit of a codeword emerging from the decoder. The pipelining is compensated for by one of the additional D-flip flops in the codeword delay register. The *COUNT ENABLE* and *COUNT CLR* signals are not transported outside the FAU and hence are not de-glitched.

The state machine logic for the FSM pseudocode in Figure 4.8 was minimized using “espresso” [11]. The “espresso” input and output files for the FAU System Controller are in Appendix B. Direct implementation of the pseudocode would have required 31 gates in the LSI Logic design system. The minimized implementation required 28. Thus the logic minimization still resulted in a 10% saving in system controller gate count.

After asynchronous reset or power-up, the FAU system controller is brought up in State B. This allows it to clock in one full 1360/2316 bit received word before it begins using the syndrome data to scan for the frame boundaries.

4.6 Error Correction Unit (ECU)

As noted previously, the syndrome information from the CCSC syndrome register is also routed to the ECU. The ECU is responsible for processing the syndrome information at codeword end times to determine if the received word contains any errors. If a single error is present in the received word, the ECU locates the errored bit. The block diagram of the ECU is shown in Figure 4.9:

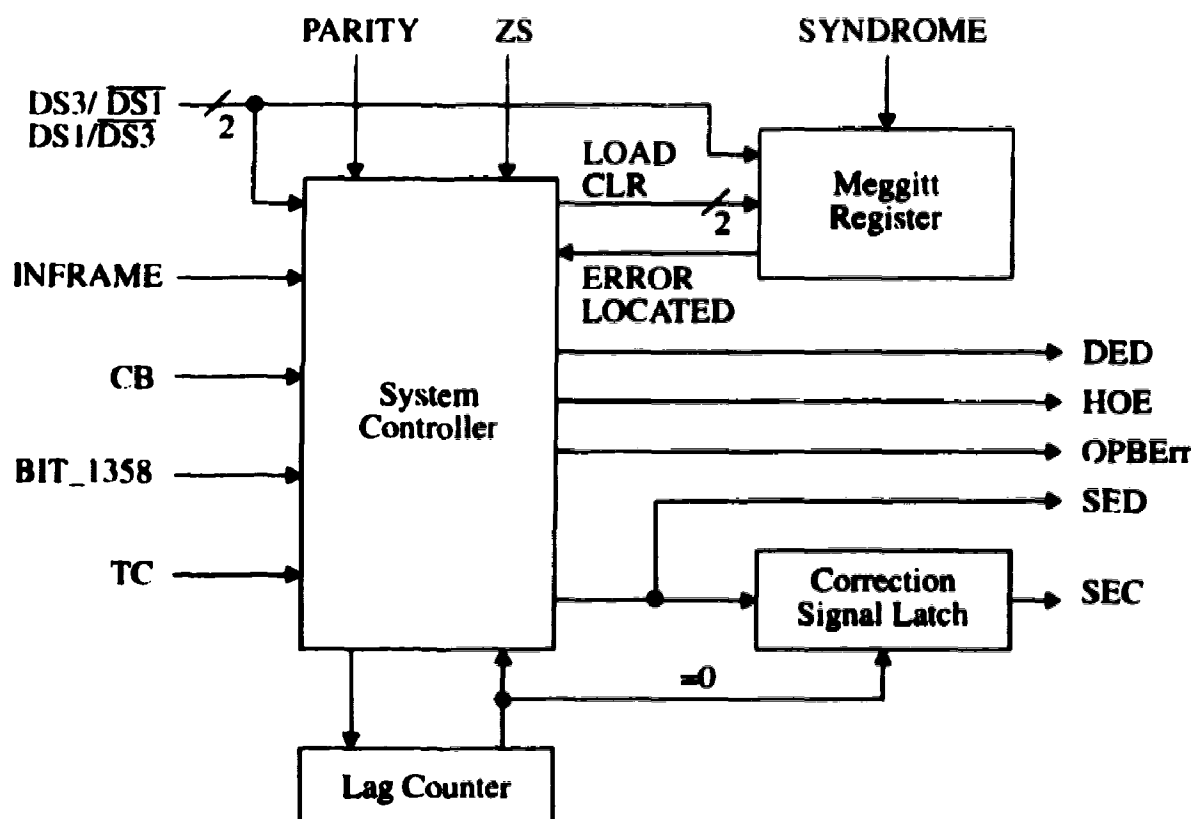


Figure 4.9: Error Correction Unit Block Diagram

The inputs to this module are the following (excluding system clock and asynchronous reset, not shown):

- **DS1, DS3:** Complementary mode select signals
- **INFRAME:** Frame alignment status of the FEC decoder from FAU.
- **CB:** Asserted HIGH if the bit at the output of the delay buffer in the next clock cycle (note not the current cycle) is an FEC checkbit.

- ***BIT_1358***: Used in DS3 mode only; indicates when all possible bit positions have been examined for an error in the DS3-FEC codeword.
- ***TC***: Terminal Count; indicates codeword end time.
- ***PARITY***: The value of the parity of the received word.
- ***SYNDROME***: The syndrome for the received word.
- ***ZS***: Asserted if all bits of 12/11-bit SYNDROME (for DS1 or DS3 respectively) are identically zero.

and the outputs of the ECU are the following:

- ***SED*** (Single Error Detect): Asserted HIGH when a single error is detected. The pulse occurs at the bit time in which the errored bit is located, but not yet corrected.
- ***SEC*** (Single Error Correct): Asserted HIGH when a single error is corrected. The pulse occurs in the same clock cycle that the errored bit emerges from the codeword delay register.
- ***DED*** (Double Error Detect): Asserted HIGH for one clock cycle, coincident with the first bit of a codeword containing a double error when it leaves the codeword delay register. (DS3 only).
- ***HOE*** (Higher Order Error): Asserted for one clock cycle in the last bit position (DS1) or next to last bit position (DS3) of a codeword containing a higher order error (to be defined).
- ***OPBErr*** (Overall Parity Bit in Error): Asserted HIGH for one clock cycle coincident with the first bit of a codeword, when it exits the delay register, in which the overall parity bit itself was observed to be in error. (DS3 only)

The ECU determines the location of a single bit error and generates a signal to correct that error. Error location is disabled under the following conditions:

- (i) while the decoder is OOF.
- (ii) when in-frame, and a zero syndrome is decoded, i.e. an error free received word.
- (iii) when in-frame, a zero syndrome occurs, and the received word parity is odd, i.e. the overall parity bit itself is in error.
- (iv) when in-frame, a non-zero syndrome occurs, and the received word parity is even (DS3 only).

Case (iv) represents a double weight error in one codeword. In this case, error correction is inhibited to prevent error extension. Only when the system is in-frame, a non-zero syndrome is decoded, and the received word parity is odd is error location enabled. These are the conditions which indicate that a single error has occurred.

The strategy for locating the bit in error is the Meggitt decoding technique [9]. The non-zero syndrome on the *SYNDROME* input lines is parallel loaded into the Meggitt register. The Meggitt register is a conventional $\div G(x)$ linear feedback shift register, i.e. without CCSC compensation. The block diagram for this register is shown in Figure 4.10.

The flip-flops which make up the Meggitt register D-flip flops with synchronous load and synchronous clear (Appendix A). The register implements division by $G(x)_{DS1}$ or $G(x)_{DS3}$ depending on the mode of operation. The feedback taps are controlled by AND gates $a0$, $a1$, $a2$, and $a3$ in a manner identical to the CCSC syndrome register (Section 4.3.5).

As each bit of the received word is shifted out of the delay register, the syndrome in the Meggitt register is examined for a specific remainder result which indicates the error position. For a shortened Hamming code, the error location pattern is given by [9]:

$$x^{n-l-1} \bmod G(x) \quad (4.17)$$

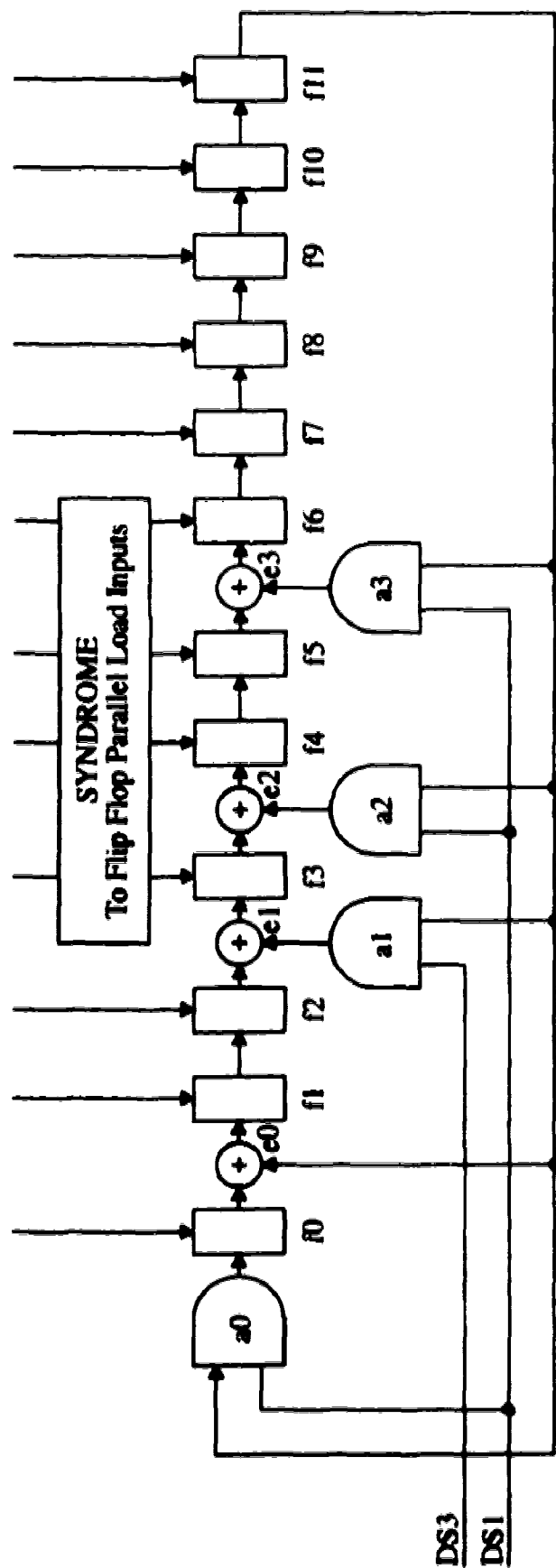
where n is the number of codeword bits in the full length parent code (not including the overall parity bit if one exists), and l is the number of bits by which the parent code has been shortened. For the DS1-FEC and DS3-FEC codes, the error location patterns are the following:

$$ELP(x)_{DS1} = x^{2314} \bmod G(x)_{DS1} = x^{11} + x^{10} + x^6 + x^2 + x + 1 = C47_{16} \quad (4.18)$$

$$ELP(x)_{DS3} = x^{1358} \bmod G(x)_{DS3} = x^{10} + x^8 + x^2 = 504_{16} \quad (4.19)$$

when the syndrome in the Meggitt register matches this pattern, the bit emerging from the delay buffer is the bit in error. The respective pattern is decoded by the pattern detection block in Figure 4.10 (Appendix A for circuit description). When the error is found, the Meggitt register is cleared to prevent any further error correction activity in the current codeword.

The Meggitt decoding strategy assumes that the received word is in systematic form. The fact that the checkbits are actually distributed throughout the DS1-FEC and DS3-FEC codewords causes an offset between the time at which the error position is located and the time at which the corresponding data actually emerges from the delay buffer. For example, consider the DS1-FEC codeword. The very first position in the codeword is an FEC checkbit. Suppose then that the CCSC decodes a syndrome of $C47_{16}$ for a received codeword. This syndrome value means that the very first bit of the codeword in systematic form, i.e. the first message bit, is in error. However, the first message bit actually resides in the second bit position of the DS1-FEC codeword structure. Thus the signal to correct the



DS1: 1	x	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}	x^{11}	x^{12}
DS3:	1	x	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}	x^{11}

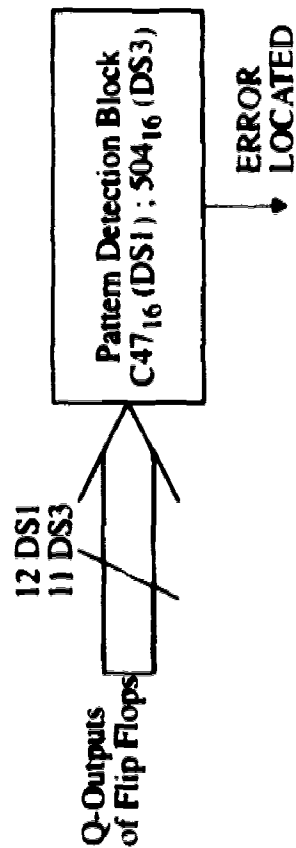


Figure 4 10: Meggitt Decoding Register

STATE A: DISABLED

```
Wait Until INFRAME ;
Wait for Terminal Count;
if (ZS is TRUE) {
    STATE = A ;
} else {
    Load Syndrome into Meggitt Register ;
    Clear Correction Pulse Timing Latch ;
    Clear Lag Counter ;
    STATE = B ;
}
```

STATE B: LOCATING BIT ERROR

```
if (Terminal Count) {
    if (Meggitt SYNDROME = C4716) {
        Assert SED (Arms Correction Pulse Latch)1;
    } else {
        Assert HOE;
    }
}
if (Next Codeword's ZS is TRUE) {
    Clear Meggitt Register ;
    STATE = A ;
} else {
    Load Syndrome into Meggitt Register ;
    Clear Correction Pulse Timing Latch ;
    Clear Lag Counter ;
    STATE = B ;
}
} else {
    if (Meggitt SYNDROME = C4716) {
        Assert SED (Arms Correction Pulse Latch)1;
        Clear Meggitt Register ;
        if (Next bit is NOT an FEC Checkbit) {
            Decrement Lag Counter ;
            STATE = C ;
        } else {
            if (Next bit is an FEC Checkbit) {
                Increment Lag Counter ;
            }
            STATE = B ;
        }
    }
}
```

STATE C: ERROR LOCATED, RESOLVING LAG

```
if (Lag Counter=0) {
    if (Terminal Count) {
        if (Next Codeword's ZS is TRUE) {
            STATE = A ;
        } else {
            Load Syndrome into Meggitt Register ;
            Clear Correction Pulse Timing Latch ;
            Clear Lag Counter ;
            STATE = B ;
        }
    } else {
        STATE = A ;
    }
} else {
    if (Terminal Count) {
        if (Next Codeword's ZS is TRUE) {
            STATE = A ;
        } else {
            Load Syndrome into Meggitt Register ;
            Clear Correction Pulse Timing Latch ;
            Clear Lag Counter ;
            STATE = B ;
        }
    } else {
        if (Next bit is NOT an FEC Checkbit) {
            Decrement Lag Counter ;
        }
        STATE = C ;
    }
}
```

Figure 4.11 (a): Pseudo-code for ECU System Controller Operation in DSI mode

1. At this point the error is located and the correction pulse timing latch is armed. The ECU FSM then controls the Lag Counter to resolve any lag in correction pulse timing. The actual error correction pulse is asserted by the latch at the instant that the lag is resolved.

<pre> STATE A: DISABLED Wait Until INFRAME ; Wait for Terminate Count; if (PARITY=0) { if (SYNDROME=0) { Assert DED ; STATE = A ; } } else { if (SYNDROME=0) { Assert OPBErr ; STATE = A ; } else { Load SYNDROME into Meggitt Register , Clear Correction Pulse Timing Latch ; Clear Lag Counter ; STATE = B ; } } </pre>	<pre> STATE B: LOCATING BIT ERROR if (EQ_1358 is FALSE) { if (Meggitt SYNDROME=504₁₆) { if (Next bit is an FEC Checkbit) { Increment Lag Counter ; } STATE = B ; } else { Clear Meggitt Register ; Assert SED (Arms Correction PulseLatch) ; if (Lag Counter=0) { STATE = A ; } else { if (Next bit is NOT an FEC checkbit) { Decrement Lag Counter ; } STATE = C ; } } } else { if (Meggitt SYNDROME=504₁₆) { Assert SED (Arms Correction PulseLatch) ; } else { Assert HOE ; } Clear Meggitt Register ; STATE = A ; } </pre>	<pre> STATE C: ERROR LOCATED, RESOLVING LAG if (Lag Counter=0) { STATE = A ; } else { if (EQ_1358 is TRUE) { Decrement Lag Counter ; STATE = A ; } else { if (Next bit is NOT an FEC checkbit) { Decrement Lag Counter ; } STATE = C ; } } </pre>
---	---	--

Figure 4.11 (b) Pseudocode for ECU System Controller Operation in DS3 mode

error in this bit must be delayed by one bit time. In general, for each checkbit passed over, the number of bit times which the error correction signal must be delayed increases by one. This is the difference between the signals on the *SED* and *SEC* output lines. *SED* is asserted during the bit time that the error is located; *SEC* is asserted when the errored bit actually emerges from the delay register, taking the interleaved checkbit positions into account.

The required delay is managed with the Lag Counter and the Correction Pulse Timing Latch. The Lag Counter is a 4-bit up/down counter with synchronous clear and count enable. The Correction Timing Latch is a state machine which is used to delay the application of the correction pulse (Appendix A). While the Meggitt decoder is sniffing to locate the bit error, the Lag Counter is incremented each time a checkbit is passed over. When the error position has been determined, the Correction Timing Latch is armed. The Lag Counter is then decremented once for each message bit passed over. When the counter equals zero, the Correction Pulse Timing Latch asserts the SEC signal. The SEC signal is used by the Output Conditioning Block to complement the errored bit before it goes off chip. The Correction Pulse Timing Latch uses a D-Flip flop to de-glitch the SEC signal which introduces a one bit-time delay. This delay is compensated for by the second additional D-flip flop in the codeword delay buffer. Note also that in DS1 mode, because the very first bit of the DS1-FEC codeword is a checkbit, there is an immediate a lag of one bit time between the error location and correction pulse timing. In DS3 mode, the first timing lag occurs after bit 85 (the first FEC checkbit). Thus the operation of clearing the Lag counter in DS1 mode actually involves loading the counter with 0001 rather than setting its content to zero as for DS3 mode.

Because the DS1-FEC and DS3-FEC codes are shortened from their respective parent codes, a percentage of higher weight errors (>1 DS1 or >2 DS3) can also be detected. In the parent codes, each non-zero syndrome value indicates a specific bit error position. In shortened codes, only those syndromes corresponding to single bit errors in the shortened code space have physical meaning. When a higher order error (HOE) occurs, the decoder will compute a non-zero syndrome with odd parity over the codeword (DS3 only), and thus the error location block will be enabled. If the syndrome is associated with a single bit error in the shortened code space, an error correction attempt will be made, but it will typically be at a bit position which is not truly in error causing error extension rather than error correction if a HOE truly has occurred. However, if the end of the shortened codeword space is reached (e.g. terminal count for DS1, BIT_1358 for DS3) and the error has not yet been located by the ECU, then the syndrome corresponds to an error in a bit posi-

tion outside the shortened codeword space and a HOE is thereby detected in the decoder and indicated on the HOE line.

The ECU system controller manages all of the control signals to implement the error location algorithm described above. Pseudocode for this controller in DS1 and DS3 modes of operation is shown in Figure 4.11. Notice that the error location algorithms are slightly different for DS1 and DS3. The key differences are as follows:

- (i) DS3 mode must process the parity bit also for DED.
- (ii) The search length is less than the codeword length for DS3 by one bit due to the overall parity bit.

Each DS3-FEC codeword is 1360 bits long, but only 1359 of these are message bits and checkbits. The Meggitt decoding scheme does not include the overall parity bit. Therefore, as far as the Meggitt decoding scheme is concerned, the DS3-FEC codeword is a 1359 bit codeword in systematic form. Thus an error in any position of such a codeword will be determined within 1358 shifts of the Meggitt register. This is why the DS3 error location algorithm exits after the 1358th bit has been clocked out of the delay register (indicated by the *BIT_1358* signal) rather than when the last bit is clocked out. An error in the last bit is still correctable because the Correction Pulse Timing Latch delays the correction pulse timing as needed. The fact that the DS3-FEC decoder is always able to return to the disabled state (State A) before the codeword end time means that its actions whether the subsequent codeword is error free or in error are identical. This luxury does not exist in the DS1 state machine. If the error is in the very last data bit then the system controller will still be in state C when the syndrome must be examined for the subsequent codeword. If the codeword is error free then the machine goes to state A, otherwise it goes directly to state B and begins the process of locating the error.

The system controller was implemented by combining the two state machine algorithms, and reacting appropriately depending on the mode of operation, i.e. DS1, DS3 are parameters of the input control logic. An optimal state assignment for the controller was obtained using a program called "mustang" [11]. Mustang makes state variable assignment so that the silicon area occupied by the finite state machine (FSM) is a minimum, implying that the amount of next state and output function logic must be minimized. A minimized logic implementation of the controller was achieved using "espresso" [11]. The "mustang" and "espresso" input and output files are in Appendix B. The minimized decoding logic for the FSM required only 127 gates in the LSI Logic design system. The corresponding un-minimized version would have required 254 gates. Thus a savings of 50% was achieved.

4.7 Output Conditioning

This module has the following two functions. The first is the actual correction action to correct a single bit error. The second function is reconstruction of the conventional DS1 or DS3 OH-bit sequence. For DS1 this means replacing the checkbits of the FEC code with the conventional F-bit sequence. For DS3, in addition to restoration of the F-bits, the two C-bits which were used to transport FEC checkbits must be restored to match the value of the one remaining C-bit in the frame.

The Decoder Output Conditioning Module is shown in Figure 4.12.

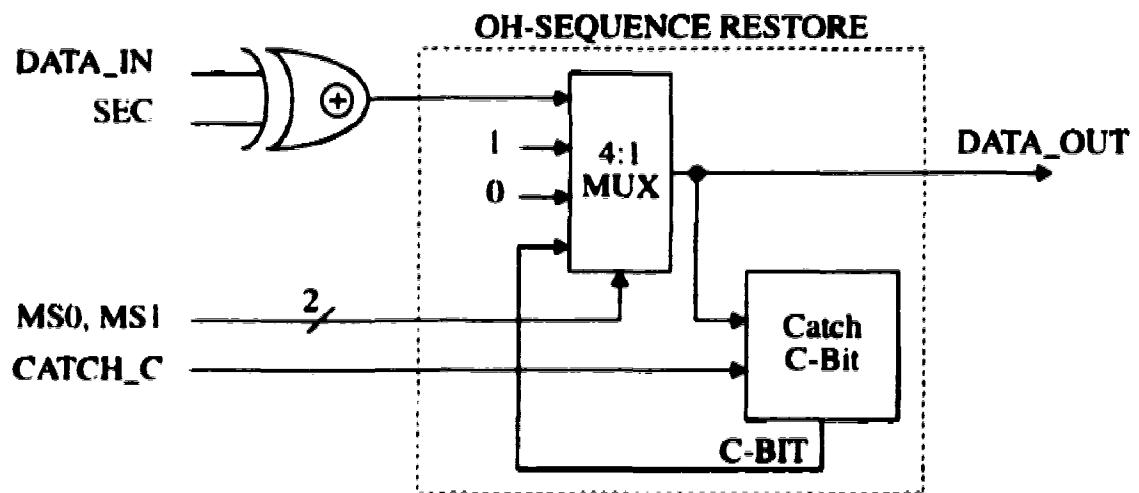


Figure 4.12: Decoder Output Conditioning Module

The inputs to this module, in addition to the system clock and asynchronous reset lines (not shown), are the following:

- **DATA_IN**: the value of the bit currently emerging from the delay buffer (including pipeline delay re-times)
- **CORRECT_BIT**: single bit error correction pulse from ECU
- **MS0, MS1**: multiplexor control signals for OH-bit sequence reconstruction. (Come from main decoder control event sequencer)
- **CATCH_C**: control signal (from main event sequencer) to indicate that the bit on the **DATA_IN** line is one of the two valid C-bits covered by the DS3-FEC codeword and should be sampled for use in replacing the checkbits at the other C-bit positions in the corresponding frame.

The module has a single output, **DATA_OUT**, which is the error corrected DS1 or DS3 data signal with the conventional framing sequence restored.

4.7.1 Error Correction Function

The XOR gate, *x0*, implements the correction action to correct a single bit error. When the bit emerging from the delay buffer, i.e. the bit on the *DATA_IN* line has been identified as being in error, the *SEC* line is asserted high. The logic level at the output of *x0* is the complement of the bit value on *DATA_IN*. Hence the erroneous bit is corrected. When no error correction action is to take place, *SEC* will be low and the data will pass through *x0* unchanged.

4.7.2 OH-Bit Sequence Reconstruction

For the DS1 mode of operation, the 4:1 MUX is used to restore the conventional F-bit sequence to the framing bit locations of the DS1 superframe. The *MS0* and *MS1* control signals are used to select the value of the bit to be output from the FEC decoder. If the bit emerging from the delay buffer, i.e. the bit on the *DATA_IN* line, is a message bit, then it is allowed to pass through the 4:1 MUX and is put out on the *DATA_OUT* line. If the bit on the *DATA_IN* line is an FEC checkbit which occupies an F-bit position which is normally a 1 in the DS1 superframe then the MUX controls are used to select the MUX input which is hard-wired to a 1 rather than this bit. Likewise for a logical zero.

The process of frame sequence restoration for the DS3 signal format is similar. Each time a bit is presented to the Output Conditioning from the delay register which occupies an F-bit position, the bit is ignored and the appropriate value of the conventional F-bit is selected on the MUX for insertion instead. However, in the DS3 case, the C-bits which were used to transport FEC checkbits must also be restored to the value of the one valid C-bit for the associated frame, i.e. the C-bits must emerge triplicated from the FEC decoder. Each time that a valid C-bit emerges from the codeword delay buffer (it happens twice per DS3-FEC codeword), the value of this bit is sampled by the *CATCH C-BIT* module (see Appendix A for a circuit description). This value is held on the *C_BIT* output of this module until the next valid C-bit emerges from the delay buffer. The value of the sampled C-bit is then inserted into the data stream at the two remaining C-bit times in the frame via control of the 4:1 MUX. Note that the error correction takes place prior to the C-bit restoration because if the valid C-bit is in error then it is necessary to correct this error prior to sampling the bit and reconstructing the frame sequence.

The *MS0*, *MS1*, and *CATCH_C* control signals also come from the main decoder system controller module (Section 4.8). These are used to select the appropriate input of the 4:1 MUX for correct reconstruction of the OH-bit sequence. Table 4.7 presents the functions

implemented for each possible value of MS0 and MS1.

Table 4.7: 4:1 MUX Control Signal Truth Table

MS0	MS1	Value on DATA_OUT line
0	0	DATA_IN
0	1	0
1	0	1
1	1	Sampled C-Bit

Note that there is no explicit re-configurability to the architecture of this module for DS1 or DS3 mode of operation. The *CATCH C-BIT* module is simply not used in DS1 mode as there are no C-bits to restore.

4.8 Main FEC Decoder System Controller

The main decoder system controller is responsible for asserting the required control signals at the appropriate times to allow correct operation of the frame alignment, error correction, and frame reconstruction process. These control signals include the following: *MS0*, *MS1* MUX control signals for F-bit replacement, *CATCH_C* for C-bit sampling, *FRAME_OUT*, marking the start of each outgoing DS1 Superframe or DS3 Frame; *MEMBER_CB*, indicating that the bit after the current bit emerging from the delay register is an FEC checkbit; *EQ_1358*, to indicate to the ECU in DS3 mode that the end of the error location space has been reached, and *TC*, to indicate the codeword end time. The main decoder system controller consists of two units, the system state counter and the state decoding logic. The system state counter is a 12-bit binary cyclic counter. It counts from 0 to 2316 in DS1 mode and from 0 to 1359 in DS3 mode. The state decoding logic decodes the value of the counter and asserts the control signals depending on Table 4.8 summarizes the counter values for which each signal is asserted.

When the system loses frame alignment, the OOF declaration will be made when the system state counter is equal to zero. The SLIP output from the FAU is applied to the synchronous clear of the system counter, thus holding it at zero until a frame candidate is found.

Table 4.8: State Counter Values for which each Control Signal is Asserted

Signal Name	Counter Values (DS1) Decoded	Counter Values (DS3) Decoded
MS0	193, 386, 579, 1158, 1930, 2123	255, 340, 425, 510, 935, 1020, 1105, 1190
MS1	0, 772, 965, 1351, 1544, 1737,	85, 340, 510, 595, 765, 1020, 1190, 1275
CATCH_C	N/A	170, 850
FRAME_OUT	0	0, 680
MEMBER_CB	192, 385, 578, 771, 964, 1157, 1350, 1543, 1736, 1929, 2122	84, 254, 339, 424, 509, 594, 764, 934, 1019, 1104, 1189, 1274
EQ_1358	N/A	1358
TC	2315	1359

Note that the *MS0* and *MS1* signals are asserted one bit time after the *MEMBER_CB* signal. The values decoded for these three control signals are decoded when the system counter reads 83, 191, 253, 338, 384, 423, 508, 577, 593, 763, 770, 933, 963, 1018, 1103, 1156, 1188, 1273, 1349, 1542, 1735, 1928, and 2121. They are combined appropriately to create the *MEMBER_CB* signal and routed to a 1-bit delay element to remove decoding glitches and generate the correct signal timing. The terms are also appropriately combined to form *MS0* and *MS1*. These signals are routed to 2-bit delay elements to achieve their correct timing.

The FSM logic was minimized using “espresso” [11]. The espresso input and output files can be found in Appendix B. The ratio of minimized implementation gate count to unminimized gate count is 312/393.

Chapter 5

Theoretical FEC-Derived Framing and Error Correction Performance of the DS1/DS3 FEC Decoder

In this chapter the theoretical framing and error correction performance of the FEC ASIC is presented. The framing performance analysis is examined first in Section 5.1, because if frame alignment is not achieved then no error correction can take place. The theoretical error correction performance and BER improvement is presented in Section 5.2

5.1 Theoretical Framing Performance

The process of frame alignment and its associated timing parameters, reframe time, OOF detection time, MF time and FIF time, were defined in Chapter 2, along with the strategy by which FEC framing was possible. In [3], a method was presented to estimate the aforementioned timing parameters for FEC-framing for the purpose of (i) selecting C_R and C_O such that FIF and MF times were at least as long as in the conventional framing system to which FEC-derived framing was compared, and (ii) for analyzing the theoretical reframe time and OOF detection time performance using the selected threshold values. In this chapter, the methods from [3] for obtaining the theoretical distributions of reframe time and OOF detection times for FEC framing are presented. We have made some notational changes so that the timing parameter equations are in terms of the ASIC inputs C_R and C_O . Given that MF and FIF times are made extremely long, T_{RF} and T_{OFD} are the frame alignment timing parameters of most practical interest as they characterize the two most common transitions in the frame alignment process.

The technique in [3] for estimating the timing parameters is identical to the method for lumped frame alignment word (FAW) framing presented in [5]. This technique is valid to apply to FEC framing as FEC framing is effectively equivalent to FAW framing. After transformation of the codewords into their respective syndrome representations, searching for the 12 bit zero syndrome (or 11 bit zero syndrome + even overall parity for DS3) is equivalent to searching for an all zero FAW (with some adjustments in the probabilities associated with FAW detection). The maximum average reframe and OOF detection times for FEC-derived framing are estimated using probability generating functions (PGF) derived from signal flow diagrams as presented in [5]. Section 5.1.1 defines the PGF, presents some of its useful properties, and outlines the method for deriving the PGF for the transition time between two states in a generalized signal flow diagram. In Sections 5.1.2

and 5.1.3 respectively the PGF's for the maximum average reframe and OOF detection times are derived.

5.1.1 Probability Generating Functions

The PGF of a random variable, X , is given by:

$$P_X(z) = E\{z^X\} = \sum_i p_i z^{x_i} = p_0 z^{x_0} + p_1 z^{x_1} + p_2 z^{x_2} \dots \quad (5.1)$$

where $E[\cdot]$ is the expectation operator, z is a dummy variable, and p_0, p_1, p_2, \dots are the discrete probabilities for $X = x_0, x_1, x_2, \dots$ etc. The mean and the variance of X can be obtained from the PGF in the following way:

$$E[X] = P'_X(1) \quad (5.2)$$

$$\text{VAR}[X] = P''_X(1) + P'_X(1) - \{P'_X(1)\}^2 \quad (5.3)$$

The PGF's for the maximum average reframe time and OOF detection time are derived as signal transfer functions from a generalized state transition diagram describing each process. In this type of state transition diagram, each edge has both a transition probability and transition time associated with it. The transition time is expressed as the exponent of the dummy variable z , which indicates time delay. The transition probability and transition time indicate that the state transition on the associated edge happens with the indicated probability and that when it does happen, it takes the indicated transition time. The PGF of the transition time between any two states is simply the signal transfer function of the state transitions. The signal transfer function can be found using signal flow graph reduction techniques.

5.1.2 Theoretical Maximum Average Reframe Time

The reframe process for FEC-derived framing consists of two distinct phases: search for the codeword boundary, and confirmation to verify correct frame alignment. The PGF for the search process, where a maximal length search must be performed is first derived and then used as a timing parameter for developing the PGF for the complete reframe process (combined search and confirmation).

5.1.2.1 The Search Process

The maximum average reframe time assumes that the search process begins at one bit past the true frame boundary, i.e. the maximal length search. The search process must therefore reject all misaligned bit positions of the codeword ($n-1$ bits). Each time that a bit position is rejected, a slip is generated causing the framing circuit timing to shift to the next bit

position. This process is called the falsehood detection process and is depicted in Figure 5.1 as the transition from bit position i in the codeword to the next position, $i+1$, where position i is an arbitrary misaligned bit position.

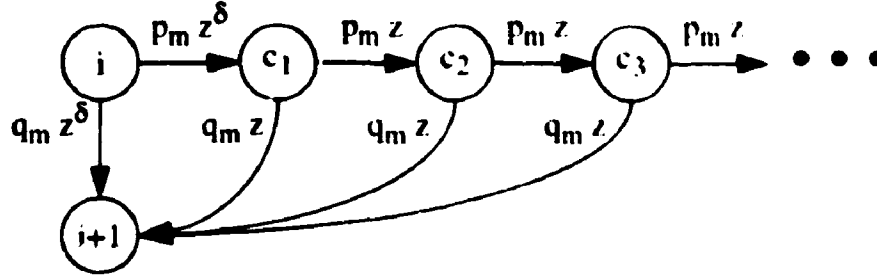


Figure 5.1: Falsehood rejection process for one bit in FEC-derived framing

The criterion in FEC framing for the rejection of bit position i as the codeword boundary is the syndrome decoded at position, i . If the syndrome is non-zero, it is immediately apparent that position i is not the codeword boundary and a slip is generated, i.e. the transition is made to state $i+1$. This transition takes a one bit time because CCSC allows single bit time computation of the altered syndrome for the succeeding bit position. However, if the syndrome at position i is zero, due to data mimicking a valid codeword, position i is held as a candidate frame position (transition from state i to state c_1). Each instance that the mimic persists at the candidate position, incurs an additional codeword time until it is rejected by the confirmation (falsehood detection) process. The model shown in Figure 5.1 assumes that c is actually infinite when performing the falsehood detection process, i.e. a false codeword boundary candidate is always rejected so FIF can never occur. The assumption is reasonable because c is chosen large enough so that the probability of FIF is negligibly small and hence incurs negligible discrepancy in the estimated values of search time.

The signal transfer function for the transition between state i and state $i+1$ in Figure 5.1 is

$$\frac{q_m z^\delta}{1 - p_m z} \quad (5.4)$$

For the maximal length search, $(n-1)$ bit positions in the codeword must be rejected and it takes a one bit time to slip to the true codeword boundary. Therefore, the signal transfer function for the complete maximal length search process is:

$$\tau(z) = \left(\frac{q_m z^\delta}{1 - p_m z} \right) z^\delta \quad (5.5)$$

where $p_m = 2^{-12}$ is the probability that data at a misaligned position in the codeword mim-

ics a valid codeword, $q_m = 1 - p_m$, and δ is the normalized duration of a single bit time relative to the duration of a complete DS1-FEC or DS3-FEC codeword. For the DS1-FEC code, $\delta = 1/2316$ and for the DS3-FEC code, $\delta = 1/1360$.

5.1.2.2 The Combined Search and Confirmation Process

The signal flow graph for the combined search and confirmation process is shown in Figure 5.2.

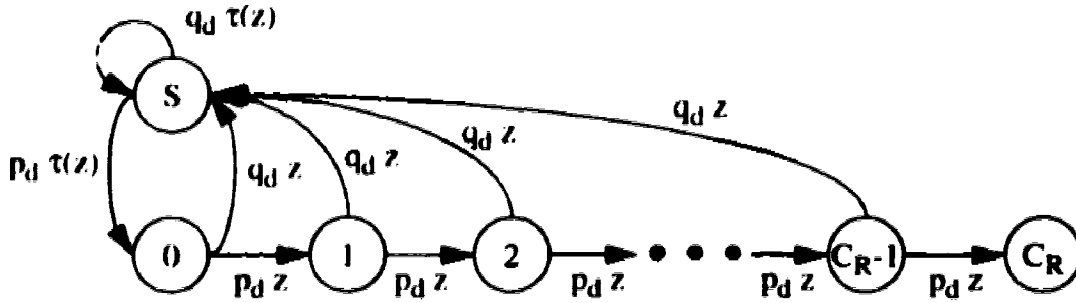


Figure 5.2: Combined search and confirmation process for FEC-derived framing

After the search process (with timing characterized by $\tau(z)$) is complete, the framing circuit arrives at the true frame candidate. However, due to bit errors in the data stream, the syndrome at the true codeword boundary may be non-zero. If this is the case, the true frame boundary is rejected as a frame candidate and the search process is called in again. This is indicated at state S in Figure 5.2 by the feedback loop onto itself. However, if the codeword at the true frame position is error-free, then the confirmation process is called in with the confirmation counter initially equal to zero (transition from state S to state 0). The confirmation process then requires that C_R additional zero syndromes to be detected in subsequent codewords before in-frame is declared. However, if a non-zero syndrome is decoded during the confirmation phase, the confirmation counter is reset, the candidate is rejected, and another maximal length search is required (transition back to state S).

The signal transfer function of the combined search and confirmation process, i.e. between state S and state C_R in Figure 5.2 is the following:

$$P_{RF}(z) = \frac{p_d^{C_R+1} \cdot \tau(z) \cdot z^{C_R}}{1 - q_d \cdot \tau(z) \cdot \left(\frac{1 - p_d^{C_R+1} \cdot z^{C_R+1}}{1 - p_d \cdot z} \right)} \quad (5.6)$$

where p_d is the probability of the successful detection of a codeword, i.e. the probability that each bit of the codeword is uncorrupted. For the DS1-FEC and DS3-FEC codes respectively, p_d is defined as follows:

$$p_d = \begin{cases} (1 - \text{BER}_{\text{channel}})^{2316} & \text{DS1-FEC} \\ (1 - \text{BER}_{\text{channel}})^{1360} & \text{DS3-FEC} \end{cases} \quad (5.7)$$

and $q_d = 1 - p_d$. C_R is the number of times that a zero syndrome must be seen at the code word boundary after the initial zero syndrome is seen at the end of the search process.

The signal transfer function for the reframe process given in (5.6) is the PGF of the maximum average reframe time and can be used in equations (5.2) and (5.3) to find its mean and the variance. The mean maximum average reframe is [5]:

$$T_{\text{RF}} = \frac{1}{p_d^{C_R+1}} \left\{ (n-1) \frac{p_m}{q_m} + \frac{1 - p_d^{C_R+1}}{q_d} \right\} \quad (5.8)$$

The probability distribution of the maximum average reframe time can be obtained by expanding (5.6) as a power series as in (5.1).

5.1.3 Theoretical Out-of-Frame Detection Time

In FEC-derived framing, a c-successive count scheme is used for OOF detection as well as for reframing [3]. The signal flow diagram for the out-of-frame detection process for FEC derived framing is shown in Figure 5.3

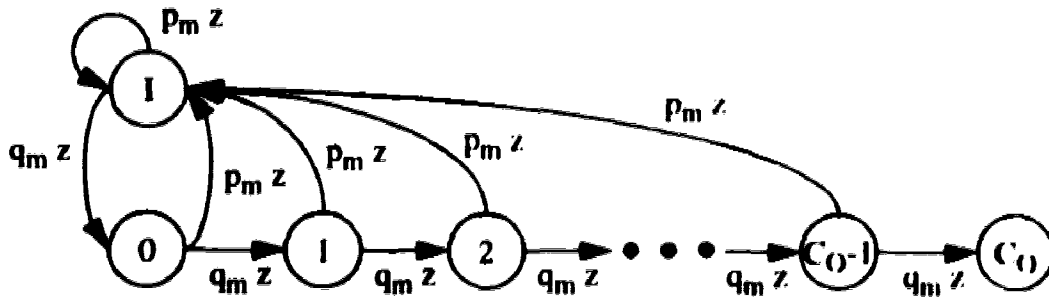


Figure 5.3: Signal flow graph for the FEC-derived out-of-frame detection process

The OFD process begins in state I where the system believes itself to be in-frame but is actually out-of-frame. At each codeword end time, the syndrome value is examined. If a non-zero syndrome is seen, the system enters state 0 with the OFD is initially equal to zero. At this point, C_O additional consecutive non-zero syndromes in subsequent codewords must be seen before out-of-frame is declared. If a zero syndrome is decoded at any time during the confirmation process, due to a codeword mimic at the misaligned position, the OFD counter is reset and the system returns to state I. The signal transfer function from state I to state C_O in Figure 5.3 is

$$P_{OFD}(z) = \frac{q_m^{C_0+1} \cdot z^{C_0+1}}{1 - p_m \cdot z \left(\frac{1 - q_m^{C_0+1} \cdot z^{C_0+1}}{1 - q_m \cdot z} \right)} \quad (5.9)$$

where p_m and q_m are as defined in Section 5.3, and C_0 is the number of confirmations in which a non-zero syndrome must be seen at the assumed codeword boundary before OOF is declared, once the initial non-zero syndrome is seen. Equation 5.9 is the PGF of the out-of frame detection time and can be used in equations (5.2) and (5.3) to obtain the mean and variance of the OFD time for DS1-FEC or DS3-FEC framing, and can be expanded into a power series to obtain the probability distribution of OOF detection times. The mean OOF detection time is given by [5]:

$$T_{OF} = \left(\frac{1}{q_m^{C_0+1}} \right) \left(\frac{1 - q_m^{C_0+1}}{p_m} \right) \quad (5.10)$$

5.2 Theoretical Error Correction Performance

In this section the error correction/detection capabilities of the DS1-FEC and DS3-FEC code are summarized. This information is used to determine the BER of the error corrected data stream for a given input error weight where bit errors are assumed to be random and independent.

5.2.1 Single Error Correction in DS1-FEC and DS3-FEC Codes

Both the DS1-FEC and DS3-FEC codes have single error correction capabilities. Thus a single bit error at any bit position in the DS1-FEC or DS3-FEC codeword structure will be corrected.

5.2.2 Double Error Detection in DS3-FEC Code

The DS3-FEC code has the capability to detect any combination of two errors in a codeword, i.e. all possible double errors. In this case, error correction is inhibited and no error extension is possible.

5.2.3 Higher Order Error Detection in DS1-FEC Code

Both the DS1-FEC and DS3-FEC codes have the ability to detect a certain percentage of higher weight errors via the HOE detection mechanism described in Section 4.6. In the case of the DS1-FEC, a higher order error is any error pattern of weight greater than one. When a HOE is detected, error correction is suspended so that no error extension takes place. The probability that a HOE will be detected is the probability that the syndrome decoded for the codeword containing the HOE is associated with an error position outside

the shortened code space. For the DS1-FEC code, this is given by:

$$P_{\text{HOE}} = \frac{L_{\text{Parent}} - L_{\text{DS1-FEC}}}{L_{\text{Parent}}} = \frac{4095 - 2316}{4095} = 0.434 \quad (5.11)$$

where L_{Parent} is the length of the parent code from which the DS1-FEC code is derived, and $L_{\text{DS1-FEC}}$ is the number of bits in the DS1-FEC codeword format. Error extension only occurs if the syndrome decoded for the errored codeword is associated with an error in a DS1-FEC data bit position as errors in the FEC checkbits are corrected by the decoder when the DS1 superframe sequence is reconstructed. The probability that error extension takes place is:

$$P_{\text{EXT}} = \frac{L_{\text{DS1-FEC-DATA}}}{L_{\text{Parent}}} = \frac{2304}{4095} = 0.563 \quad (5.12)$$

where $L_{\text{DS1-FEC-DATA}}$ is the number of bits in the data portion of the DS1-FEC codeword format. Thus error extension will only occur in 56.3% of the codewords which contain two or more bit errors.

5.2.4 Higher Order Error Detection in DS3-FEC Code

As presented in Section 5.2.2, the DS3-FEC code has DED capabilities so all double errors are detected without error extension. In fact, the mechanism for detecting a double error, i.e. nonzero syndrome with even parity over the codeword, will prevent error extension in a codeword containing any even weight multiple error. This is because the parity of a codeword containing an even weight multiple error is always even. Regardless of the value of the syndrome (zero or non-zero) error correction is always inhibited. This means that for DS3 mode, the types of errors which are potentially detectable via the HOE detection mechanism are odd weight multiple errors. The probability of detecting a HOE for the DS3-FEC code is

$$P_{\text{HOE}} = \frac{L_{\text{Parent}} - L_{\text{DS3-FEC}}}{L_{\text{Parent}}} = \frac{2047 - 1360}{2047} = 0.336 \quad (5.13)$$

where L_{Parent} is the length of the parent code from which the DS3-FEC code is derived and $L_{\text{DS3-FEC}}$ is the number of bits in the DS3-FEC codeword. The probability that error extension will take place if the decoder decodes an odd weight multiply errored frame is

$$P_{\text{EXT}} = \frac{L_{\text{DS3-FEC-DATA}}}{L_{\text{Parent}}} = \frac{1348}{2047} = 0.663 \quad (5.14)$$

where $L_{\text{DS3-FEC-DATA}}$ is the number of data bits in the DS3-FEC codeword format.

5.2.5 BER of DS3-FEC Error Corrected Data Signal

In this section, the results of the preceding sections are used to determine the output BER of the error corrected data stream for a given input BER where the errors in the coded data stream are random and assumed to be independent. The expected number of errors, after decoding, in a DS3-FEC codeword which contains k_0 errors prior to FEC decoding is given by the following:

$$N_e(k_0) = \begin{cases} 0 & k_0 = 0, 1 \\ k_0 & k_0 = 2, 4, 6, \dots \\ (k_0 + 1) \cdot P_{EXT} + (k_0) \cdot Q_{EXT} & k_0 = 3, 5, 7, \dots \end{cases} \quad (5.15)$$

where P_{EXT} is given in (5.12) and $Q_{EXT} = 1 - P_{EXT}$. The probability of exactly k_0 errors in a DS3 codeword is:

$$P(k=k_0) = \binom{L}{k_0} p_e^{k_0} q_e^{L-k_0} \quad (5.16)$$

where $L = L_{DS3-FEC} = 1360$, p_e is the raw BER of the channel, and $q_e = 1 - p_e$. Using (5.13) and (5.14) the BER of the error corrected data is given by:

$$BER_{OUT} = \left(\frac{1}{L_{DS3-FEC}} \right) \sum_{i=2}^{L_{DS3-FEC}} N_e(i) \cdot P(k=i) \quad (5.17)$$

For most BER levels of interest, only the first two terms of (5.17) contribute significantly:

$$BER_{OUT} = 1359 \cdot p_e^2 \cdot q_e^{1358} + 1125768 \cdot p_e^3 \cdot q_e^{1357} \quad (5.18)$$

For input BER levels $< 10^{-5}$, the dominant term in (5.15) is the second order term and the BER of the error corrected signal is approximately given by $1359p_e^2$.

5.2.6 BER of DS1-FEC Error Corrected Data Signal

The expected number of errors after FEC decoding in a DS1-FEC codeword which contains k_0 errors prior to decoding is given by:

$$N_e(k_0) = \begin{cases} 0 & k_0 = 0, 1 \\ (k_0 + 1) \cdot P_{EXT} + (k_0) \cdot Q_{EXT} & k_0 = 2, 3, 4, \dots \end{cases} \quad (5.19)$$

where P_{EXT} is now given by (5.10). The probability of exactly k_0 errors in a DS1 codeword is given by (5.14) with $L = L_{DS1-FEC} = 2316$. Thus the BER of the error corrected data stream emerging from the FEC decoder is given by the following:

$$\text{BER}_{\text{OUT}} = \left(\frac{1}{L_{\text{DS1 FEC}}} \right) \sum_{k=2}^{L_{\text{DS1 FEC}}} N_c(k) \cdot P(k=1) \quad (5.20)$$

Again, at BER levels of interest, only the first two terms of (5.20) make a significant contribution to the output BER:

$$\text{BER}_{\text{OUT}} = 2963 \cdot p_e^2 \cdot q_e^{2314} + 3178433 \cdot p_e^3 \cdot q_e^{2314} \quad (5.21)$$

For BER levels $< 10^{-5}$, the dominant term in (5.20) is the p_e^2 term, so the BER of the error corrected DS1 data stream is approximately $2963 \cdot p_e^2$.

Chapter 6

Laboratory Setup

The system level testing of the FEC ASIC had three objectives: (i) verification of the predicted error correction and detection behavior for controlled, deterministic error patterns, (ii) verification of performance in a totally random error environment, (iii) measurement of re-frame and out-of-frame detection time statistics for various C_R , C_O and levels of BER. In this chapter, the laboratory test environment to achieve these objectives is presented. The description of the test environment presented here is primarily at the functional level. Complete documentation and circuit designs are in [12]. Experimental results are presented in Chapter 7.

6.1 Laboratory Test Environment - General

A block diagram of the laboratory test environment is shown in Figure 6.1 (see also Plate 6.1). The setup consists of a DS1 or DS3 bit error rate test set (BERT), and the following three printed circuit boards: the FEC board, the error injector board, and the random error generator board. The FEC board houses four functional blocks: two FEC ASIC's, the BERT interface, and the Out-of-frame generator. Each ASIC contains both encoder and decoder circuits but in the experimental tests, one ASIC is used for each separate function. Two function generators and a number of totalizing counters are used for measuring statistical variables in the experiments. In the following subsections the test equipment configurations and the functions of each of sub-circuits on the circuit boards are described in order of signal flow.

6.2 DS1 and DS3 Bit Error Rate Test Sets (BERTs)

The Tautron S5104 (DS1) and S5200E (DS3) BERTs each consist of a transmitter and a receiver. The transmitter supplies the bit rate clock and pseudorandom data signals. In DS1 mode the clock frequency is 1.544 MHz and the data is in superframe (SF) format. For DS3 mode, the clock frequency is 44.736 MHz and the data is framed according to the standard DS3 format. The payload data bits are a pseudo-random bit sequence (PRBS) of period $2^{20}-1$ for DS1 and $2^{15}-1$ for DS3. The DS3 transmitter also supplies a signal which locates the start of each DS3 masterframe. This is used for synchronization of the FEC encoder. A similar signal in DS1 mode identifies the beginning of the DS1 superframe, but is not available from the DS1 BERT. It is derived instead on the FEC board in the equip-

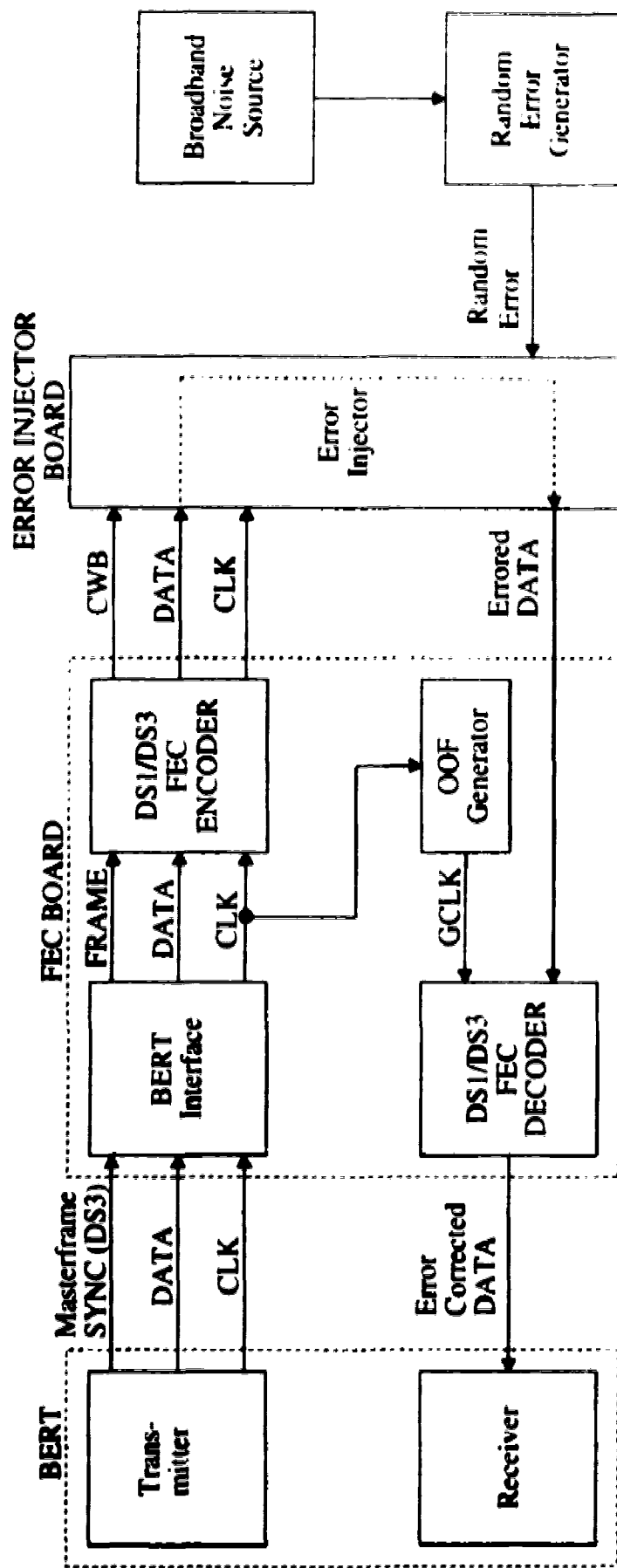


Figure 6.1: Block Diagram of Laboratory Test Environment



Plate 6.1: Laboratory Test Environment Unit

ment interface block from the DS1 data signal (Section 6.3). The BERT receiver frames on the restored conventional DS1 or DS3 frame structure emerging from the decoder, counts the number of errors in the payload bits, and computes the average BER over the duration of each experiment. Note that errors in the OH-bits of the DS1 superframe or DS3 masterframe are not counted by the BERT.

6.3 FEC Board: BERT Interface

The signals from the transmitter BERT are routed to the BERT interface. The BERT interface serves two main functions: (i) condition (or derive for DS1) the frame synchronization signal in the form expected at the FB input of the FEC encoder, and (ii) ensure the clock and data have the required phase relationship at the input to the FEC encoder. Most of the processing surrounds task (i).

In DS1 test, the inputs to the BERT interface are the 1.544 Mb/s clock signal and DS1 data in SF format. Before being routed to the appropriate pins on the FEC encoder the clock and data signals pass through a T1 primary rate framer. In DS1 mode, the FEC encoder requires a pulse, one clock cycle in width, which is coincident with the first bit of the DS1 superframe for encoder synchronization on its *FRAME IN* input as described in Section 3.1. The T1 framer module in the test environment produces a 50% duty-cycle square wave, with a period of the DS1 superframe and with its rising edge at the start of the DS1 superframe. This edge is captured, and converted to an active low pulse one bit time wide. The clock, DS1 data, and start of superframe signals are then routed to the FEC encoder.

In DS3 testing, the inputs to the BERT interface are the 44.736 MHz clock, framed DS3 data, and masterframe synchronization (MSYNC) signals. Although the period of the raw MSYNC signal out of the BERT is that of the DS3 masterframe, neither the rising nor the falling edge of the waveform occurs at the beginning of the masterframe. The closest edge to the start of masterframe is the falling edge which occurs 312 clock cycles prior. The BERT interface therefore captures this edge and delays it as a pulse 312 clock cycles later to mark the start of masterframe. The clock, DS3 data and start of masterframe signals are routed to the appropriate input pins on the FEC encoder.

6.4 FEC Board: FEC Encoder

A functional description of the FEC encoder has already been presented in detail in Chapter 3. This section will briefly discuss the external supporting circuitry for operation and testing of the FEC encoder and the signal connections. The bit time clock, data and frame synchronization signals from the BERT interface are routed to the *CLK IN*, *DATA IN*, and

FRAME IN inputs of the FEC encoder. The *FEC ENABLE* control signal and the *DS3/DS1* mode select control signal are each set via toggle switches. There are also switches for hand clocking the FEC encoder and for placing it into “scan” mode (scan chain test) for debugging purposes (Chapter 8.7). The encoder reset comes from the FEC encoder reset circuitry. When activated by a momentary contact switch, this circuit applies an active low pulse to the reset pin of the FEC encoder for one clock time. It also inhibits the external system clock signal applied to the ASIC for this clock cycle and the next one to allow complete application and removal of the reset signal before the system is clocked. The outputs of the FEC encoder are the FEC encoded data stream, the output clock with its rising edge in the center of each outgoing data bit, and the codeword boundary (CWB) signal which marks the first bit of each DS1-FEC or DS3-FEC codeword. These signals are routed to the Error Injector Board.

6.5 Error Injector Board: Controlled Error Patterns

The purpose of the error injector is to introduce errors into the data stream so that the error correction and detection capabilities of the FEC decoder can be tested. A toggle switch configures the error injector for operation in either DS1 or DS3 mode, on either FEC encoded or unencoded formats. Errors can be injected in two overall fashions: controlled pattern or random. In controlled pattern error injection mode, single, double, or triple errors are injected into pre-determined bit positions in a codeword. In random error injection mode, the errors which are injected into the data stream are at completely random times determined by sampling a noise source. Thus random error injection is characterized only by an average BER level.

The top level schematic diagram for the error injector board, reproduced from [12] is shown in Figure 6.2

6.5.1 Walking Single Error Injection

The single bit error injection mode of the error injector is automated. In this mode when the error injector is armed, a single bit error is first introduced in the last bit of the next codeword. Then, $N_s - 1$ codewords are allowed to pass error free, where $2 \leq N_s \leq 15$ is selected via a bank of DIP switches. A single bit error is then introduced in the second last bit of the following codeword and again $N_s - 1$ error free blocks are allowed to pass unchanged and so on. This process continues until a single bit error has been injected in all possible bit positions of the codeword block. This is known as “walking single error injection” mode and is illustrated for DS3 mode with $N=2$ in Figure 6.3. Once triggered, all

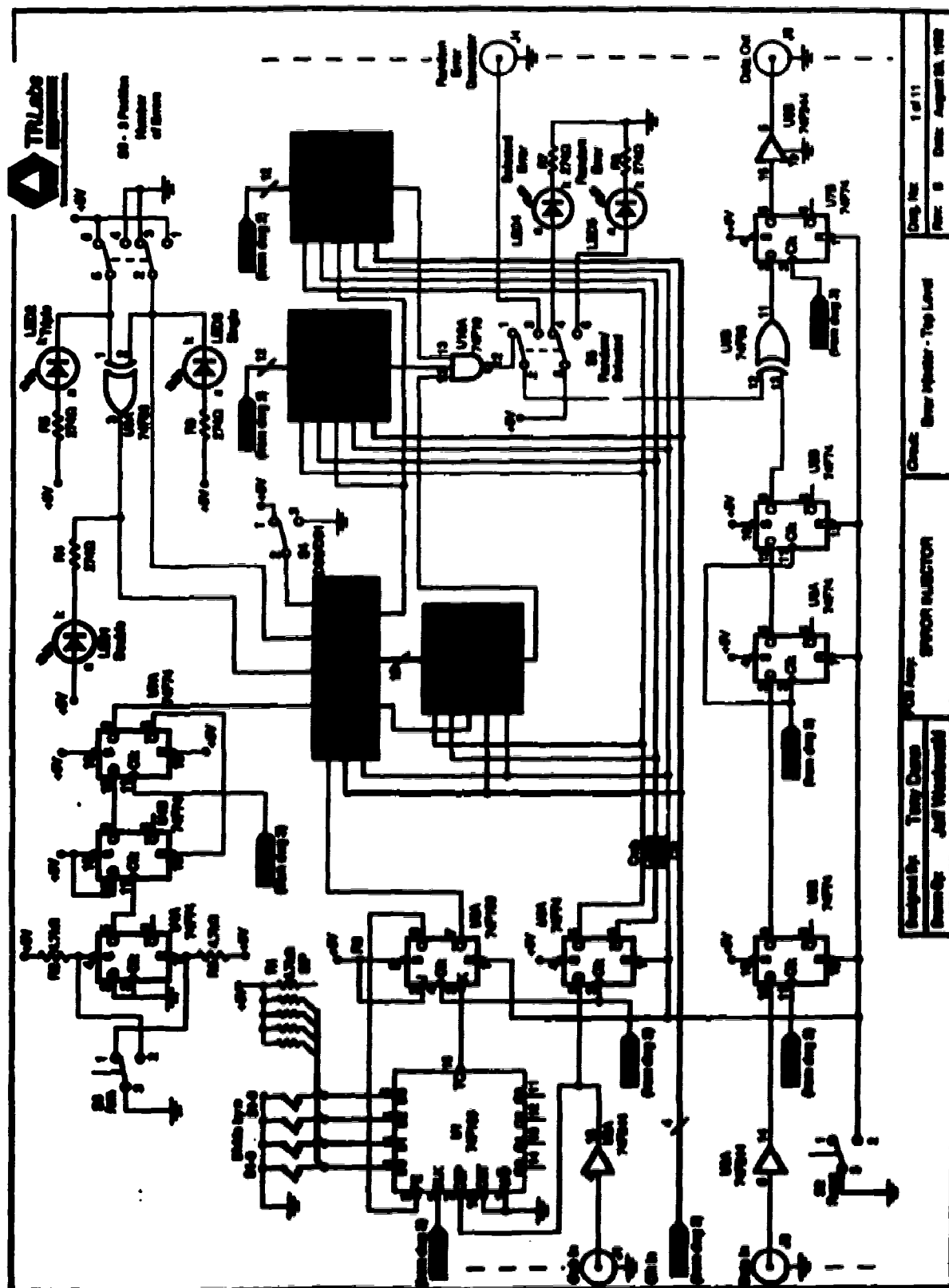


Figure 6.2: Schematic diagram of error injector board [12]

positions of the codeword are errored once and error injection stops.

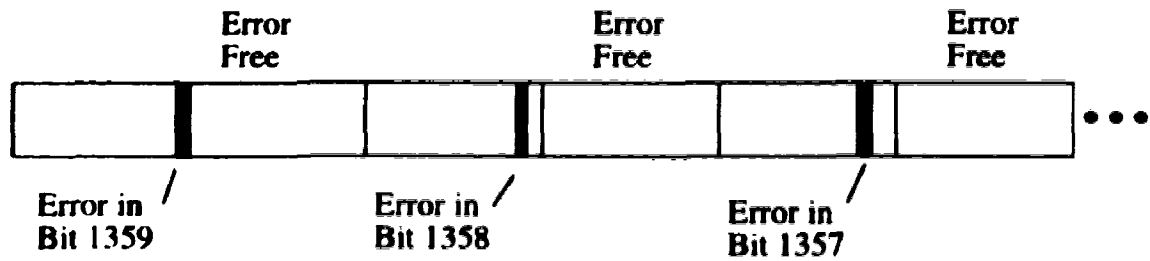


Figure 6.3: Walking Single Bit Error Injection in DS3 mode with $N=2$

The errored codewords are spaced to prevent the FEC decoder from falling out of frame by detecting C_0 successive errored codewords. This concern may also be handled by disabling out-of-frame detection at the FEC decoder via the *OFD ENABLE* input.

6.5.2 One-shot Double Error Injection; One-shot Single Error Injection

To inject double error patterns, the two indices of the bits to be errored are dialed up on two banks of DIP switches. When the injector board is placed in double error mode and is armed, the double error positions are counted down and injected once into the immediately following codeword. If the two switch banks are programmed with the same bit index, this mode can be used to inject one single bit error.

6.5.3 Walking Triple Error Injection; Walking Double Error Injection

To create triple error patterns, two fixed bit indices are dialed up on the DIP switch banks described in Section 6.5.2 and remain constant throughout the experiment. The third error is walked across all possible bit positions in the codeword by the walking single error injector. This mode of operation is called “walking triple error injection” mode. Double error injection with one fixed bit index and the other bit walking across the length of the codeword can be performed by placing the error injector board in triple mode and specifying the same fixed bit index on each of the two bit index switch banks. This mode is known as “walking double error injection”. In the controlled error injection experiments presented in Chapter 7, only “walking single”, “walking double”, and “walking triple” error injection modes are used.

6.6 Random Error Injection

In “random error injection mode”, randomly timed errors at a controlled average rate are generated by sampling the broadband white noise source shown in Figure 6.1. The schematic diagram for the random error injector module, reproduced from [12] is shown in

Figure 6.5. The noise signal is applied to one input of a high speed comparator. The other input is set to an adjustable voltage threshold. The output of the comparator is then sampled on each rising clock edge. When the noise voltage exceeds the threshold at the sample time a single bit error is injected into the data stream. By adjusting the threshold voltage, different BER levels may be set. The ideality of the random error injector was verified by testing which showed that it well reproduces the “erfc”-function classical BER vs SNR characteristic in white noise. (See Figure 6.4 reproduced from [12]).

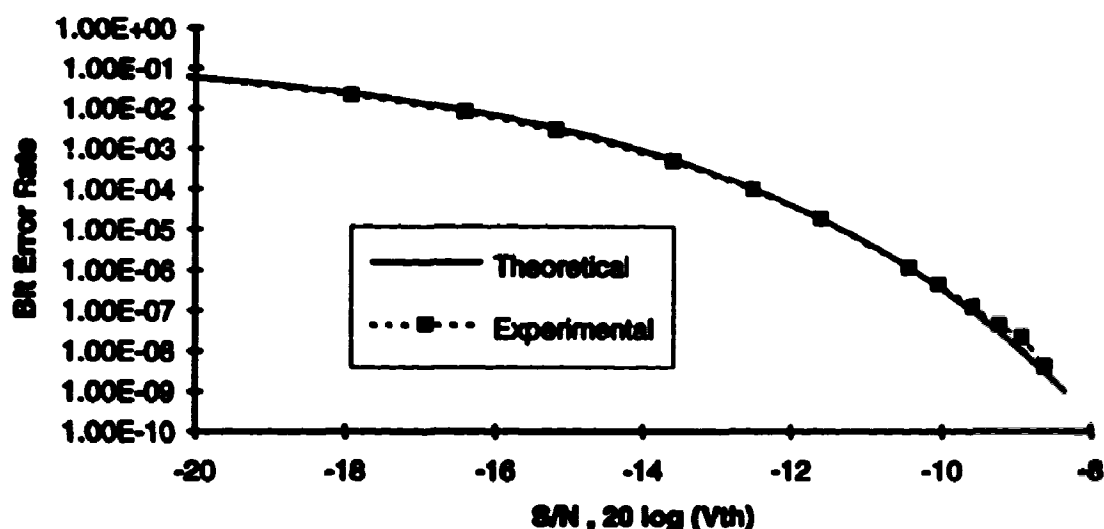


Figure 6.4: Theroetical and Measured relationship between SNR and BER

This validates the truly random nature of this injector. PRBS based error sequence injection would not suffice for testing the FEC ASIC because the error patterns it would produce would be completely deterministic and would repeat themselves. A PRBS could provide an error source signal which is characterized by a certain BER_{IN} , but if all errors correspond to single bit errors in codewords, then they would all be corrected and the BER_{OUT} would be zero. Likewise, if a double weight error occurred in a codeword then the error pattern would re-occur with period of the PRBS. This could make BER_{OUT} artificially high. Neither case is appropriate for characterizing the performance of the ASIC. Truly random errors, characterized only by average BER are needed.

Once the data stream has passed through the error injector it is routed to the FEC framer-decoder at the second ASIC position on the FEC board.

6.7 FEC Board: Out-of-Frame Generator

Although the errored data stream loops back to the FEC board from the error injector board, the system clock for the FEC framer-decoder comes from the BERT interface.

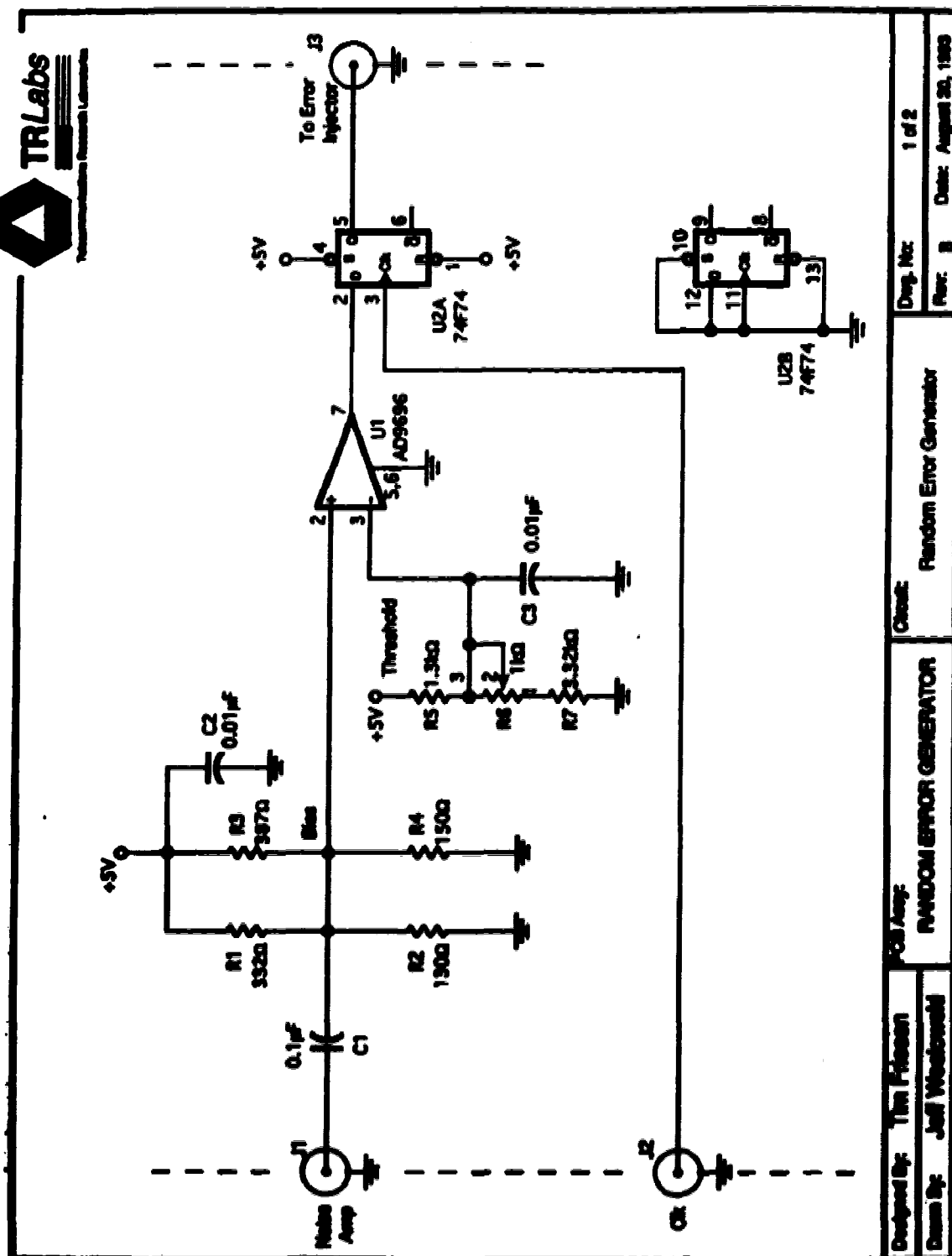


Figure 6.5: Schematic diagram for random error injector module [12]

However, before clock from the BERT interface is applied to the FEC decoder it passes through the out-of-frame generator as shown in Figure 6.1. When activated by a momentary contact switch and conditioning circuit, the OOF generator gaps out a single rising clock edge from the clock signal applied to the decoder (GCLK in Figure 6.1) preventing the data bit associated with that clock time from being clocked in. The position that the framer-decoder believes to be the frame boundary is then one bit past the actual frame boundary when clock continues. This forces the framer-decoder to go through an OOF detection and a maximal length reframe (i.e. maximum length search for the frame boundary). The OFD and maximum average reframe times can then be measured. The OFD time is measured as the interval between the instant that the rising clock edge is gapped out (true frame loss time) and the instant that the framer-decoder declares OOF. The reframe time is measured as the time from when the decoder declares OOF to the time at which the inframe state is re-established (including confirmation).

The following technique is used to measure the OFD and re-frame times: a signal is created on the FEC board which is normally low but goes high for the interval between the OOF gap event and the FEC decoder declaring OOF. This signal is connected to the gate input of a function generator. The function generator generates a square wave at the DS1 or DS3 bit clock rate at its output when its gate input goes high. The output of the function generator is connected to a totalizing counter. Thus when the OOF generator is activated, the totalizing counter counts the number of bit times between the gap event and the OOF declaration. Measurement of the reframe time is similar except that the OOF output from the FEC decoder is used. This signal is low when the decoder is inframe and high when it is OOF. Thus it will be high for the interval that the decoder is trying to re-establish the inframe state and can be similarly used to gate a counter counting bit times. This arrangement of function generator and counter is equivalent to a simpler fast start-stop real time interval timer but the latter was not available.

6.8 FEC Board: FEC Decoder

The operation of the FEC decoder was presented in detail in Chapter 4 and will be revisited here only as necessary to describe test environment interfaces to the ASIC. The FEC decoder is reset by the external decoder reset circuitry. The reset signal is applied to the reset input of the decoder for 1.85 ms. The decoder is still clocked during this interval so as to clock logic zeros into the codeword delay buffer. 1.85 ms is enough time to ensure serial clearing of the delay register at the DS1 clock rate. The system clock is inhibited for one clock cycle after the reset is de-asserted to ensure adequate recovery time before clocking at full speed in all internal gates.

The first task which the FEC framer-decoder must perform is frame alignment on the incoming data from the error injector. The 5-bit C_R and C_O confirmation thresholds are set by banks of DIP switches. The $DS1/\overline{DS3}$ and $OFD\ ENABLE$ mode select lines are set with toggle switches. The frame state of the FEC decoder is indicated by status LED's which indicate whether the decoder is inframe or out-of-frame, and when out-of-frame, when it is slipping through the data in search of the frame boundary. The out-of-frame indication signal from the decoder is also connected to a function generator as in Section 6.7. When the decoder is inframe it also provides an output signal which marks the start of each frame on its *FRAME OUT* output line.

The second task of the FEC decoder is error correction of the incoming data blocks. Error correction/detection activity is indicated on the single error detect (SED), single error correct (SEC), overall parity bit error detect (OPBErr), double error detect (DED), and higher order error detect (HOE) ASIC output lines. Each of these outputs is connected to a totalizing counter to record totals of corrections or detections. The error corrected data with restored DS1 or DS3 overhead bit format is routed to the receiver side of the BERT via TTL level line drivers.

6.9 Line Interface Board

The line interface board (not shown in Figure 6.1) is also present in the prototype test assembly and will be used in future field testing of the FEC the system in the network at DS3 rates. This unit is based on a commercial DSX-3 cross-connect compatible equipment interface. In field trials over real networks, this unit displaces the error injector, random error generator and noise source. The clock and FEC encoded data signals emerging from the encoder are routed to the transmitter side of the line interface. The transmitter contains the circuitry for converting the NRZ data to bipolar line B3ZS encoded data and for line build out pulse shaping for various electrical lengths of transmission to the DSX-3 panel. The line encoded data can then be transmitted any distance through the real network and subjected to real errors in transmission. After being transmitted through the network, the data is routed to the receiver side of the line interface. The receiver side recovers the clock signal from the bipolar line encoded data and converts the data back to NRZ format. The clock and data signals are then routed to the FEC decoder. Note that in this mode, the clock signal comes from the line interface and not from the BERT interface circuitry. The performance of the FEC system when subjected to real errors in transmission can then be characterized.

Chapter 7

System Level Laboratory Testing - Experiments and Results (Validation of Design and Theory)

In this chapter the results for three classes of experiments are presented. The first verifies the functionality of the FEC ASIC design in DS1 and DS3 modes by subjecting the ASIC to controlled error patterns (i.e. single, double, or triple errors at known positions in the codewords). The results are compared to the theoretically expected values. The error correction performance of the FEC ASIC in DS1 and DS3 modes when subjected to totally random errors is the second class of experiments. This verifies the basic theory of Chapter 5 for the error correction effectiveness of the exact codes implemented. In the third class of experiments, the system is knocked out of frame so that statistics for the out-of-frame detection time and maximum average reframe time can be obtained. This verifies the function design and theory of FEC-derived framing. These results are compared with theoretical values and with the performance of conventional DS1 and DS3 framers

7.1 Controlled Error Injection

7.1.1 DS1 Mode Correction of Single Bit Errors

For this experiment, the error injector is configured for "walking single error" injection in DS1 mode to inject a single bit error into each of the 2316 bit positions of the DS1-FEC codeword. With FEC coding turned off, the BERT is expected to count only 2304 errors because the 12 errors in the F-bits are not counted by the BERT. This was verified in ten experimental runs with FEC off. With the FEC coding turned on, all single bit errors are corrected, and hence the number of errors counted by the BERT is expected to be zero. This was formally observed to be the case in ten runs of the experiment. For each run, the following error correction totals resulted: SED=2316, SEC=2304. This means that errors were detected by the ASIC in each of the 2316 bits of the DS1-FEC codeword, but only errors in the 2304 payload bits were explicitly corrected. The errors in the 12 F-bit positions (CB's) were implicitly corrected as these bits are overwritten as the FEC decoder reconstructs the DS1 superframe F-bit sequence. Thus the single error correcting capabilities of the FEC decoder design have been verified in DS1 mode.

7.1.2 DS3 Mode Correction of Single Bit Errors

For this experiment, the error injector was configured for “walking single error” injection in DS3 mode to inject a single bit error into each of the 1360 bit positions of the DS3-FEC codeword (Section 6.5.1). With the FEC coding off, the BERT is expected to record 1344 errors per trial because the errors in the 16 overhead (OH) bits of the two DS3 frames covered by one DS3-FEC codeword will not be counted by the BERT. This was verified in ten experiment runs. With the FEC coding turned on, the number of errors on the BERT is expected to be zero because all single errors should be corrected. This was observed to be the case in ten formal runs of the experiment. For each run, the following error correction activity was observed: SED=1359, SEC=1348, OPBErr=1. This means that each of the 1360 errors injected was detected (SED+OPBErr), and all errors in the 1348 message bits of the DS3-FEC code (1344 payload bits, two masterframe sequence bits, and two C-bits) were corrected. The errors in the 12 F,C-bit positions which carry the CB's and parity bit are corrected when the FEC decoder reconstructs the DS3 frame sequence. Thus the SEC capabilities of the FEC decoder design in DS3 mode have been verified.

7.1.3 DS3 Mode Detection of Double Errors

For this experiment, the error injector is configured for “walking double error” injection in DS3 mode (Section 6.5.2). Under this arrangement, one bit position where an error is introduced in each double errored frame is fixed (FBI) and the other (WBI) walks across the 1360 bit positions of the DS3-FEC code. This means that when WBI=FBI, only a single bit error will be injected into that codeword. When the FEC coding is on, this error will be corrected. Therefore, double errors will actually be injected into only 1359 codewords for each run of the experiment. The number of errors which will be counted by the BERT (FEC coding on or off) will depend on whether or not the FBI is an OH-bit position as the BERT does not count errors in the OH bits. Note also, the BERT will not count the errors injected at WBI when $WBI \in OH_{DS3}$, where OH_{DS3} is the set of DS3 overhead bit positions and is given by: $OH_{DS3} = \{0, 85, 170, 255, 340, 425, 510, 595, 680, 765, 850, 935, 1020, 1105, 1190, 1275\}$.

Table 7.1 lists the number of errors which the BERT should record if FEC coding is off and $FBI \notin OH_{DS3}$, if FEC coding is on and $FBI \notin OH_{DS3}$, and if $FBI \in OH_{DS3}$ (FEC

coding off or on) respectively.

Table 7.1: Expected Number of Errors on BERT for Various WBI, FBI Combinations for Double Error Injection in DS3 mode

Case Description	FBI \in OH _{DS3} and FEC Off		FBI \in OH _{DS3} and FEC On		FBI \in OH _{DS3}	
	# Cases	# Errors on BERT	# Cases	# Errors on BERT	# Cases	# Errors on BERT
WBI = FBI	1	1	1	0	1	0
WBI \in OH	16	16	16	16	15	0
WBI \notin OH	1343	2686	1343	2686	1344	1344
TOTALS	1360	2703	1360	2702	1360	1344

The experiment was run with FEC coding turned off and with the FBI set to bit #634 (\notin OH_{DS3}). The number of errors counted by the BERT was 2703. The experiment was then run with the FBI set to bit #595 (\in OH_{DS3}) and the number of errors counted by the BERT was 1344. These results match the expected results in Table 7.1.

The FEC coding was then turned on and 28 runs of the experiment were performed with FBI = {1, 29, 40, 129, 153, 310, 339, 372, 394, 467, 486, 519, 535, 626, 762, 862, 927, 947, 1004, 1023, 1038, 1108, 1137, 1237, 1269, 1317, 1328, 1349}, none of which are OH-bit positions. In all cases, there were 2702 errors counted by the BERT and the following error detection activity was observed: SED=1, SEC=1, DED=1359. This means that the one codeword which contains the single error has that error corrected as indicated by the SED and SEC activity. In the 1359 other cases where the codeword contained a double error, this error condition was correctly identified and no error extension occurred. This indicates that the design operates as expected and indicated.

The experiment was then run 16 times with FBI \in OH_{DS3}. In all cases, the BERT recorded 1344 errors which matches the expected value from Table 7.1. Table 7.2 shows the error detection/correction activity for each experiment. As shown in Table 7.2, each of the 1359 double errors was detected and no error extension occurred. For the one codeword containing a single bit error injected when FBI=WBI, the error detection/correction action varies as expected. OH-bit positions 0, 170, 680, and 850 are covered by the DS3-FEC code and hence the single bit error is explicitly corrected. However, OH-bit 1275 is the overall parity bit for the codeword. The detection of the single bit error in this bit is indicated on the OPBErr line. The error is implicitly corrected when the FEC decoder

reconstructions the conventional DS3 OH-bit sequence. The 11 remaining OH-bits carry the checkbits of the Hamming code. Errors in the checkbits are detected on the SED line and are implicitly corrected when the FEC decoder reconstructs the DS3 frame sequence.

Table 7.2: Observed Number of errors and error correction/detection activity for double errors where FBI \in OH-bits in DS3 mode

Fixed Bit Indices	SED	SEC	OPBE	DED
0, 170, 680, 850	1	1	0	1359
85, 255, 340, 425, 510, 595, 765, 935, 1020, 1105, 1190	1	0	0	1359
1275	0	0	1	1359

From these double error experiments for DS3, two main conclusions can be drawn. In all observed cases, the FEC decoder correctly detected the presence of a double error in a codeword. Thus it is verified that any double error in a codeword will be detected. The theoretical value for the number of errors recorded on the BERT when the FEC coding is on was derived assuming that no error extension occurs. Because the observed number of errors recorded by the BERT with the FEC coding on matches this theoretical number exactly, it is also clear that no error extension is taking place. When a double error is detected, the error correction circuitry is being correctly disabled.

7.1.4 DS1 Mode Detection of Higher Order Errors

Because the DS1-FEC code has only single error correction capabilities, any error pattern of weight greater than one constitutes a higher order error which may or may not be detectable. The most common of these higher weight error patterns is the double error. The error detection performance of the FEC ASIC in DS1 mode when subjected to double errors was thus investigated. For this experiment, the error injector was configured for “walking double error injection” in DS1 mode (Section 6.5.2). Note that the BERT again will not count errors in the DS1 OH-bit positions. The set of DS1 overhead bit positions, OH_{DS1} , is given as: $OH_{DS1} = \{0, 193, 386, 579, 772, 965, 1158, 1351, 1544, 1737, 1930, 2123\}$. As in Section 7.1.3, the numbers of errors which the BERT should record when the FEC coding is off if $FBI \notin OH_{DS1}$ or $FBI \in OH_{DS1}$ can be calculated exactly. These are shown in Table 7.3.

Table 7.3: Expected Number of Errors on BERT for Various WBI, FBI Combinations for Double Error Injection in DS1 mode

Case Description	FBI \notin OH _{DS3}		FBI \in OH _{DS3}	
	# Cases	# Errors on BERT	# Cases	# Errors on BERT
WBI = FBI	1	1	1	0
WBI \in OH	12	12	11	0
WBI \notin OH	2303	4606	2304	2304
TOTALS	2316	4619	2316	2304

With the FEC coding turned off, the experiment was first run with FBI=323 (\notin OH_{DS1}) and then with FBI=193 (\in OH_{DS1}). The number of errors counted by the BERT were 4619 and 2304 respectively, which match the expected values in Table 7.3.

When the FEC coding is turned on, the number of errors which will be counted on the BERT depends on how many double errors cause error extension at the (DS1-FEC) decoder. The number codewords with double errors that cause error extension will be different for each run of the experiment with a different FBI. The expected value for the number of errors counted on the BERT, N_{eh} , for a double error injection experiment with FEC coding on, can be determined by treating N_{eh} as follows: Consider the following development: for a single codeword containing a double error, Table 7.4 lists the number of errors which will be recorded on the BERT depending on whether the locations of the two bits in error (BI_1 and BI_2) are OH-bit positions and whether or not error extension takes place.

Table 7.4: Expected Number of Errors Recorded by BERT (DS1 Double Errors)

	With Error Extension	Without Error Extension
$BI_1, BI_2 \in OH_{DS1}$	1	0
$BI_1 \in OH_{DS1} ; BI_2 \notin OH_{DS1}$	2	1
$BI_1, BI_2 \notin OH_{DS1}$	3	2

The "walking double error injection" experiment will introduce double errors into 2315 codewords in total. From Table 7.33, if FBI (i.e. BI_1) \in OH_{DS1}, there will be 11 code-

words where WBI (i.e. BI_2) $\in OH_{DS1}$ and 2304 where $WBI \notin OH_{DS1}$. Using this, Table 7.4, and (5.10), the expected value for number of errors on the BERT for each DS1 double error experiment can be calculated:

$$E[N_{ch}] = 11[1 \cdot P_{EXT} + 0 \cdot Q_{EXT}] + 2304[2 \cdot P_{EXT} + 1 \cdot Q_{EXT}] = 3607 \quad (7.1)$$

where $Q_{EXT} = 1 - P_{EXT}$. However if $FBI \in OH_{DS1}$, then there will be 12 codewords where $WBI \in OH_{DS1}$ and 2303 where $WBI \notin OH_{DS1}$. The expected number of errors on the BERT for this experiment is then given by:

$$E[N_{ch}] = 12[2 \cdot P_{EXT} + 1 \cdot Q_{EXT}] + 2303[3 \cdot P_{EXT} + 2 \cdot Q_{EXT}] = 5920 \quad (7.2)$$

The results of the double error experiments run for the 12 possible cases where $FBI \in OH_{DS1}$ are now shown in Table 7.5. (These data are exactly repeatable for any run of these experiments)

Table 7.5: Observed Number of errors and error correction/detection activity for DS1 double errors where $FBI \in OH$ -bits

FBI	Number of Errors on BERT	SED ^a	SEC	HOE
0	3604	1306	1300	1009
193	3605	1308	1301	1007
386	3604	1308	1300	1007
579	3603	1306	1299	1009
772	3603	1306	1299	1009
965	3602	1306	1298	1009
1158	3602	1306	1298	1009
1351	3601	1304	1297	1011
1544	3599	1302	1295	1013
1737	3598	1302	1294	1013
1930	3599	1302	1295	1013
2123	3598	1300	1294	1015

a. SED in the table is one less than the actual number of pulses on SED observed in the lab to compensate for the detection of the single error in a CB injected when $WBI=FBI$

The results of 24 "walking double error injection" experiments where $FBI \in OH_{DS1}$ are listed in Table 7.6.

Table 7.6: Observed Number of errors and error correction/detection activity for DS1 double errors where FBI \neq OH-bits

FBI	Number of Errors on BERT	SED^a	SEC^b	HOE
18	5911	1302	1293	1013
176	5903	1290	1285	1025
286	5914	1304	1296	1011
350	5901	1292	1283	1023
403	5896	1284	1278	1031
484	5899	1286	1281	1029
687	5919	1308	1301	1007
717	5922	1310	1304	1005
853	5921	1310	1303	1005
936	5905	1294	1287	1021
975	5910	1300	1292	1015
983	5909	1298	1291	1017
1194	5924	1316	1306	999
1342	5917	1306	1299	1009
1401	5911	1298	1293	1017
1514	5914	1302	1296	1013
1555	5920	1310	1302	1005
1737	5943	1334	1325	981
1894	5923	1312	1305	1003
1925	5920	1310	1302	1005
1968	5917	1306	1299	1009
2080	5921	1318	1309	997
2220	5910	1300	1292	1015
2263	5915	1306	1297	1009

a. SED in the table is one less than the actual number of pulses on SED observed in the lab to compensate for the detection of the single error in a data bit injected when WBI=FBI

b. SEC in the table is one less than the actual number of pulses on SEC observed in the lab to compensate for the correction of the single error in a data bit injected when WBI=FBI

Over the 12 runs listed in Table 7.5, the average number of errors recorded on the BERT is 3601 errors. From the results in Table 7.6, the average number of bit errors recorded on the BERT is 5914. In both of these cases, the number of errors recorded on the BERT with the FEC coding turned on is larger than the number obtained with the coding turned off. Thus error extension is taking place as expected for the DS1-FEC code because there is no overall parity bit to permit DED. (7.1) and (7.2) were derived assuming that no error extension took place when a HOE was detected. The fact that these average numbers of errors recorded by the BERT closely match their theoretical counterparts in (7.1) and (7.2) indicates that no error extension is taking place when a HOE is detected. This can also be confirmed by observing the activity on the SEC and HOE lines in Tables 7.5 and 7.6. The pulses on the SEC line indicate error extension events. Over all runs tabulated in Tables 7.5 and 7.6, the average values of SEC and HOE are 1297 and 1011 respectively. The measured value for the overall probability of error extension is therefore:

$$P_{EXT} = \frac{\overline{SED}}{2315} = \frac{1297}{2315} = 0.560 \quad (7.3)$$

and the measured value for the probability of higher order error detection is:

$$P_{HOE} = \frac{\overline{HOE}}{2315} = \frac{1011}{2315} = 0.437 \quad (7.4)$$

These observations both agree very well with the theoretical values given by (5.10) and (5.9) respectively, and confirm that HOE detection and error extension are taking place as often as predicted. Thus the DS1-FEC code will detect 43.7% of all frames containing double errors and will not attempt any error correction activity on these frames., thereby minimizing error extension for uncorrectable error patterns

7.1.5 DS3 Mode Higher Order Error Detection

The lowest weight error event which is detectable as a HOE in DS3 mode is a triple weight error. Thus the experiments for HOE detection performance for DS3 will be characterized with codewords containing three errors. For this experiment, the error injector was configured to operate in "walking triple error inject" mode for DS3 (Section 6.5.3). As in sections 7.1.3 and 7.1.4, the number of errors which the BERT should record with the FEC coding off for one, both, or neither of the FBI's being members of the set of OH-

bits for DS3 can be calculated. The calculations are presented in Table 7.7.

Table 7.7: Expected Number of Errors on BERT for Various WBI, FBI Combinations for Triple Error Injection in DS3 mode

Case Description	FBI ₁ , FBI ₂ ∈ OH _{DS3}		FBI ₁ ∈ OH _{DS3} , FBI ₂ ∉ OH _{DS3}		FBI ₁ , FBI ₂ ∉ OH _{DS3}	
	# Cases	# Errors on BERT	# Cases	# Errors on BERT	# Cases	# Errors on BERT
WBI = FBI	2	0	2	2	2	4
WBI ∈ OH	14	0	15	15	16	32
WBI ∉ OH	1344	1344	1343	2686	1342	4026
TOTALS	1360	1344	1360	2703	1360	4062

With FEC coding turned off, the experiment was run with FBI₁=1012 and FBI₂=48 (FBI₁, FBI₂ ∉ OH_{DS3}). The number of errors recorded by the BERT was 4062. The experiment was then run with FBI₁=340 and FBI₂=48 (FBI₁ ∈ OH_{DS3}). The number of errors recorded by the BERT was 2703. Finally, the experiment was run with FBI₁=340 and FBI₂=1275 (FBI₁, FBI₂ ∈ OH_{DS3}). The BERT counted 1344 errors. Each of these values agrees with the corresponding expected results from Table 7.7.

With the FEC coding turned on, Table 7.8 lists the number of errors which would be recorded on the BERT if a single triple errored codeword is decoded, depending on whether the locations of the three bits in error (B₁, B₂, B₃) are OH-bit positions and whether or not error extension is avoidable.

Table 7.8: Expected Number of Errors Recorded by BERT

	With Error Ext.	Without Error Ext.
BI ₁ , BI ₂ , BI ₃ ∈ OH _{DS3}	1	0
BI ₁ ∈ OH _{DS3} ; BI ₂ , BI ₃ ∉ OH _{DS3}	2	1
BI ₁ , BI ₂ ∈ OH _{DS3} ; BI ₃ ∉ OH _{DS3}	3	2
BI ₁ , BI ₂ , BI ₃ ∉ OH _{DS3}	4	3

Using Tables 7.7 and 7.8, the expected number of errors to be recorded on the BERT for both, one, or neither of the FBI's being DS3 OH-bit positions can be calculated. These are given in equations (7.5), (7.6), and (7.7):

$$E[N_{eh}] = 14[1 \cdot P_{EXT} + 0 \cdot Q_{EXT}] + 1344[2 \cdot P_{EXT} + 1 \cdot Q_{EXT}] = 2238 \quad (7.5)$$

$$E[N_{eh}] = 2 + 15[2 \cdot P_{EXT} + 1 \cdot Q_{EXT}] + 1343[3 \cdot P_{EXT} + 2 \cdot Q_{EXT}] = 3597 \quad (7.6)$$

$$E[N_{eh}] = 4 + 16[3 \cdot P_{EXT} + 2 \cdot Q_{EXT}] + 1342[4 \cdot P_{EXT} + 3 \cdot Q_{EXT}] = 4956 \quad (7.7)$$

where P_{EXT} is given by (5.12) and $Q_{EXT}=1-P_{EXT}$. The constants 2 in (7.6) and 4 in (7.7) come from the double errors which get injected into codewords when $WBI = FBI_1$ or FBI_2 .

The results of 10 triple error experiments where $FBI_1, FBI_2 \in OH_{DS3}$ are presented in Table 7.9.

Table 7.9: Observed Number of errors on BERT and error detection correction activity for DS3 Triple errors when $FBI_1, FBI_2 \in OH_{DS3}$

FBI_1	FBI_2	# Errors on BERT	SED	SEC	OPBE	DED	HOE
0	510	2264	928	921	0	2	430
170	1190	2250	916	908	0	2	442
340	1275	2237	906	897	0	2	452
510	1105	2258	922	917	0	2	436
680	935	2235	898	893	0	2	464
850	765	2251	916	910	0	2	442
1020	595	2232	893	889	0	2	464
1190	425	2228	891	886	1	2	466
0	255	2240	903	898	1	2	454
340	85	2235	899	894	1	2	458

The results of 10 triple error experiments where $FBI_1 \in OH_{DS3}$ and $FBI_2 \notin OH_{DS3}$ are presented in Table 7.10.

Table 7.10: Observed Number of errors on BERT and error detection correction activity for DS3 Triple errors when $FBI_1 \in OH_{DS3}$, $FBI_2 \in OH_{DS3}$

FBI_1	FBI_2	# Errors on BERT	SED	SEC	OPBE	DED	HOE
0	1352	3588	895	886	1	2	462
255	1173	3591	895	892	1	2	462
340	6	3599	907	900	1	2	450
425	1112	3600	906	901	0	2	452
595	102	3592	898	892	0	2	460
765	547	3598	907	899	1	2	450
850	206	3597	905	896	1	2	452
935	642	3600	906	899	0	2	452
1020	422	3590	897	889	1	2	460
1190	904	3604	907	904	1	2	450

The results of 20 triple error experiments where $FBI_1, FBI_2 \in OH_{DS3}$ are presented in Table 7.11.

Table 7.11: Observed Number of errors on BERT and error detection correction activity for DS3 Triple errors when $FBI_1, FBI_2 \in OH_{DS3}$

FBI_1	FBI_2	# Errors on BERT	SED	SEC	OPBE	DED	HOE
929	1324	4950	897	892	1	2	460
1049	904	4958	903	898	1	2	454
1200	861	4947	895	889	1	2	462
629	1257	4950	899	891	1	2	458
953	393	4956	902	896	0	2	456
8	1083	4960	908	900	0	2	450
661	72	4945	893	886	1	2	464
1127	330	4954	900	894	0	2	458
1340	1021	4961	908	901	0	2	450
856	570	4943	891	883	1	2	466
1227	71	4949	898	891	0	2	460
813	1263	4949	895	889	1	2	462
682	209	4978	926	917	0	2	432
285	642	4945	893	886	1	2	464
1104	1149	4936	885	877	1	2	472
352	673	4966	915	905	1	2	442
1114	41	4966	911	906	1	2	446
218	954	4934	879	875	1	2	478
902	848	4939	889	881	1	2	468
852	496	4948	895	887	1	2	462

The average values for the number of errors on the BERT from the experiment results in Tables 7.9, 7.10, and 7.12 are 2243, 3596, and 4952 respectively. As in the DS1 HOE detection performance, the number of errors recorded on the BERT with the FEC coding turned on is larger than with the coding turned off. This indicates that error extension is

taking place. However, because the mean observed values for the number of errors on the BERT are close to their respective theoretical counterparts and the theoretical values were derived assuming that no error extension takes place when a HOE is detected, it is clear that the only time error extension occurs is when the syndrome decoded for the error pattern is associated with a DS3-FEC data bit position in the shortened Hamming code. This conclusion is further supported by the agreement between the measured values for the probability of HOE detection (7.8) and the probability of error extension (7.9):

$$P_{\text{HOE}} = \frac{458}{1360} = 0.337 \quad (7.8)$$

$$P_{\text{EXT}} = \frac{892}{1360} = 0.656 \quad (7.9)$$

and their respective theoretical values (0.336), (0.663) in (5.13) and (5.14). Thus error extension will only occur in 65.6% of the cases where a codeword contains an odd error weight of 3 or greater.

It is interesting to note that some triply errored codewords are decoded as codewords which have the overall parity bit in error. This is because the DS3-FEC code is based on a distance-3 Hamming code before it is extended for DED when it becomes a distance-4 code. Ignoring the parity bit initially, a small probability exists that a triple error will convert one valid codeword into another. When this happens and the parity bit is included, the parity of the frame is incorrect because three errors have taken place. Thus this appears to be a valid codeword which has had a single bit error in the overall parity bit (the more likely situation). No error extension takes place because a single bit error in the overall parity bit position is corrected when the framing sequence is reconstructed by the decoder.

Also note that when error extension does take place, only one additional error is ever added to the frame. These deterministic error weight pattern tests serve to well validate design correctness and the theoretically expected error correction/detection/extension behaviors. To summarize these are:

• **DS1:**

- SEC
- HOE (error weight >1) detection in 43.7% of the cases (at most one additional error added)

: DS3:

- SEC
- DED (No error extension)
- Detect multiple even weight errors (No error extension)
- Multiple odd weight HOE detection in 33.7% of the cases (at most one additional error added)

7.2 Random Error Detection / Correction Performance

7.2.1 Random Error Performance and Results for DS3

For this experiment, the error injector system was configured for “random error injection”. In this mode, bit errors are injected into the bit stream at truly random times at a selected BER. In Section 5.2.5, the theoretical improvement in BER due FEC coding in the DS3 mode was derived. Equation (5.17) which relates the input BER (i.e. the raw BER of the channel) to the output BER of the error corrected data, is plotted in Figure 7.1, along with the BER 1:1 line relating input to output BER without FEC coding. Error corrected BER levels were measured for a number of input BER points between 10^{-2} and 10^{-8} . These are also shown in Figure 7.1. The confidence bars on the points represent the 95% confidence interval based on the total number of bits observed and the number of errors observed. The method for deriving the confidence intervals is given in [13].

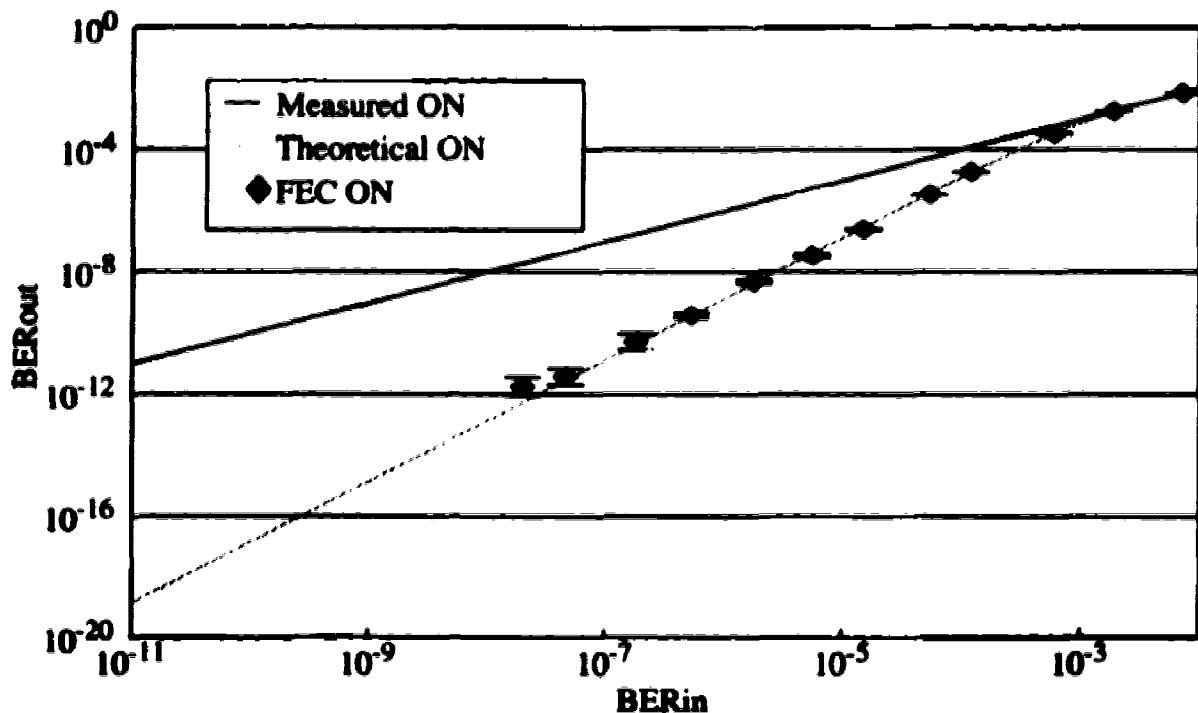


Figure 7.1: Theoretical and measured BER reduction by FEC for DS3

As can be seen in Figure 7.1, the measured improvement in BER closely matches the theoretical prediction. The last point is somewhat more uncertain than others because the circuit for generating the random bit error rate drifts over time due to changes in ambient temperature, hence the BER was not constant throughout the experiment, which must last for over 62 hours to get 10 errors at $\text{BER}_{\text{OUT}}=10^{-12}$. Measurements of 10 errors for $\text{BER}_{\text{OUT}}=10^{-15}$ at DS3 and DS1 rates would take longer than 7 and 205 years respectively. The fact that the BER is not constant throughout the experiment leads to error in the x-coordinate of the point on the graph. While the confidence bars based on the number of errors observed may be computed as for the BER_{OUT} levels, the magnitude of the error in the x-coordinate due to BER_{IN} drift is unknown because the amount of drift either above or below the original threshold setting is unknown.

7.2.2 Random Error Performance for DS1 and Results

For the DS1-FEC code, the theoretical relationship between the input (raw) BER and the output (error corrected) BER is given in Section 5.2.6 by equation (5.20). Equation (5.20) is plotted in Figure 7.2 along with the curve relating output to input BER without FEC coding.

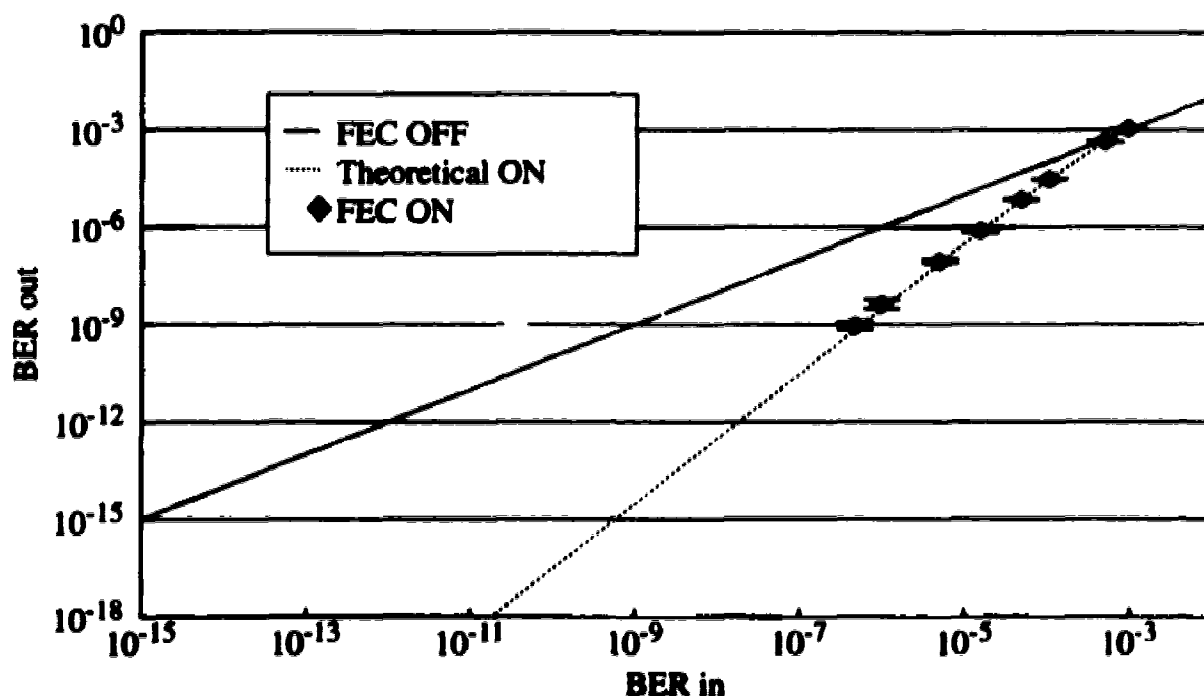


Figure 7.2: Theoretical and Measured BER reduction by FEC for DS1

Error corrected BER levels were measured for input BER levels between 10^{-3} and 10^{-7} . These points are also shown in Figure 7.2. Again, the measured values for the improvement in BER due to FEC coding closely match the predicted values. The error bars on the

points represent the 95% confidence interval based on the total number of bits observed and the number of errors observed. Again the slight deviations in the measured values from the theoretical curve are likely due to drift in the actual value of the input error rate in the random error generator due to noise source and comparator threshold voltage drifts and initial settability accuracy of the BER_{IN} values.

7.3 Experimental Framing Performance

To obtain out-of-frame detection (OFD) time and reframe time statistics, the OOF generation circuitry described in Section 6.7 was used to force maximal length reframes. The experimentally observed framing performance of the FEC ASIC for various C_R , C_O and levels of BER is presented in this section.

7.3.1 Out-of-Frame Detection DS1 and DS3

By expanding (5.9) into a power series, one can obtain the probability distribution for the out-of frame detection time. As shown in Figure 7.3, the theoretical OFD time is measured from the instant that the last zero syndrome was seen while the system was still inframe.

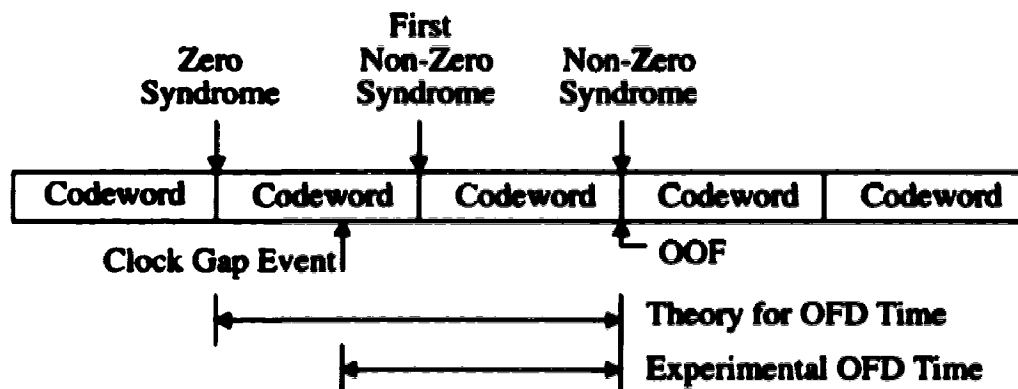


Figure 7.3: Measurement intervals for OFD time ($C_O=1$)

Because the probability of a codeword mimic at a misaligned position is so small (2^{-12}), it is most likely that the framer will acknowledge that the system is out of frame exactly C_O+1 codewords after this point in time. Table 7.12 lists the probability of OFD in exactly C_O+1 codeword times for various values of C_O .

Table 7.12: Expected OFD Time Probabilities for DS1

C_0	Prob. OFD Time = C_0+1 Codewords	Prob. OFD Time > C_0+1 Codeword
1	0.999512	0.000488
2	0.999268	0.000732
4	0.998780	0.001220
5	0.998536	0.001464

The probabilities in Table 7.12 apply to both DS1 and DS3 modes of operation because the probability of codeword mimic is the same for both modes. As shown in Figure 7.3, the measured values for the OFD are measured from the point in time that the single clock cycle is stolen to force the system out-of-frame to the point at which the decoder actually declares out-of-frame. Thus to compare the measured values of OFD time to theoretical values, the probabilities listed in Table 7.12, are interpreted as the probabilities that the measured OFD time falls between C_0 and C_0+1 codeword times.

In DS1 mode, 400 out-of-frame detection events were performed for each value of C_0 . Table 7.13 summarizes the results:

Table 7.13: Observed OFD Time Probabilities for DS1 (400 Trials)

C_0	Number of OFD Times $C_0 < T_{OFD} < C_0+1$	Measured Probability that $C_0 < T_{OFD} < C_0+1$
1	400	1.0
2	400	1.0
4	400	1.0
5	400	1.0

The measured results of Table 7.13 agree with the theoretical results insofar as they state that the probability that out-of-frame detection takes longer than C_0+1 codeword times is too small to be observed in 400 trials and that $\text{Prob}(T_{OFD} = C_0+1) \geq (1 - 1/400) = 0.9975$.

In DS3 mode, 400 out-of-frame experiments were also performed for each value of C_0 . The results of these experiments are identical to those presented in Table 7.13. Again, these results corroborate the theoretical prediction that the probability that out-of-frame

detection takes longer than C_0+1 codeword times is less than $1/400$.

7.3.2 DS1 Maximum Length Reframe Time Measurement

Figure 7.4 shows the probability distributions for maximal length DS1 reframe times for C_R values of 1, 2, and 4, and for BER ranging from 0 to 10^{-4} . Shown along with the measured probability distribution is the theoretical distribution obtained by expanding (5.6) into a power series.

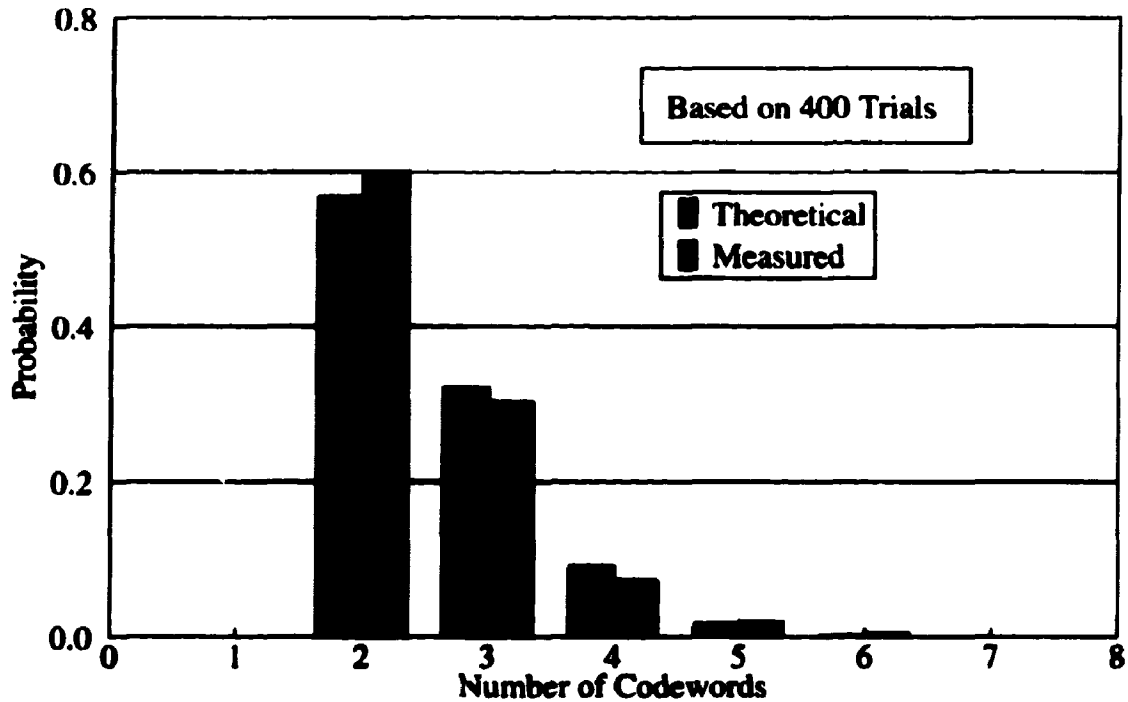


Figure 7.4 (a): Maximum average reframe time distribution for $C_R=1$ and BER = 0

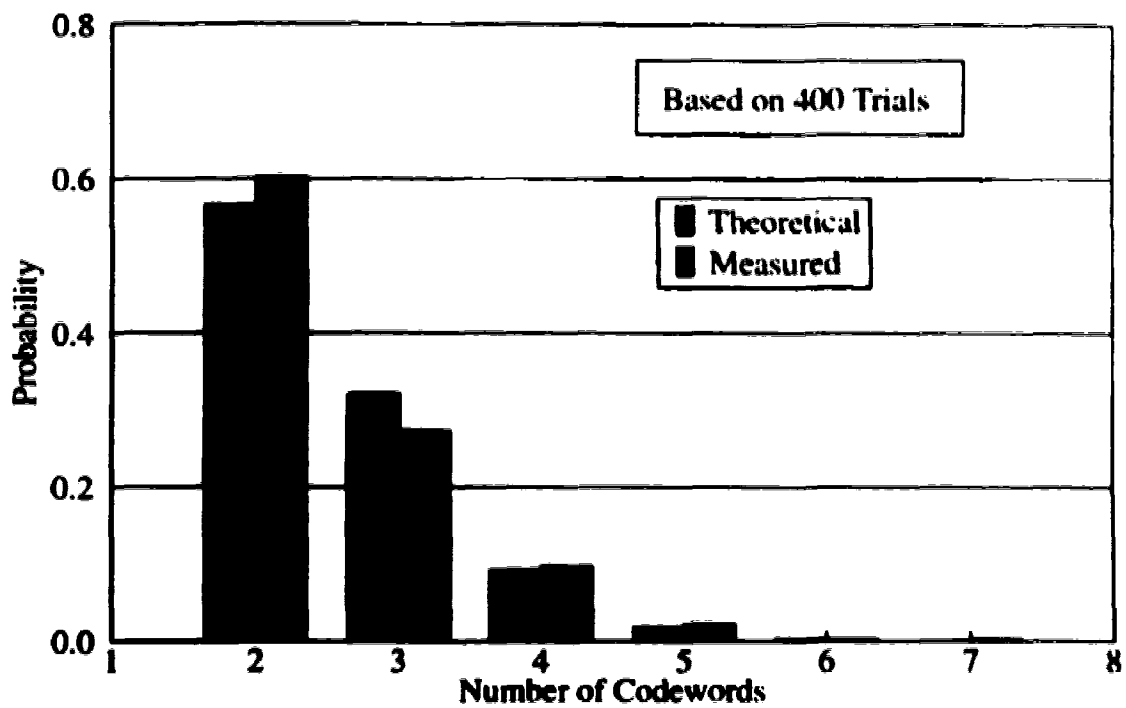


Figure 7.4 (b): Maximum average reframe time distribution for $C_R=1$ and $BER=10^{-6}$

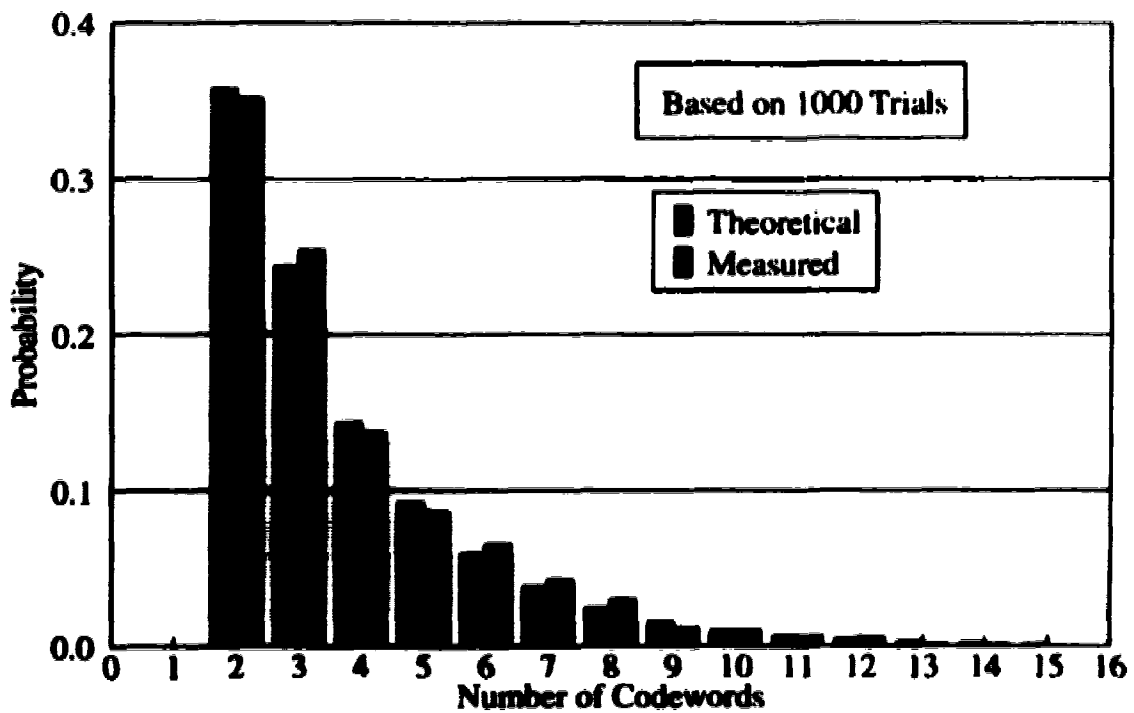


Figure 7.4 (c): Maximum average reframe time distribution for $C_R=1$ and $BER=10^{-4}$

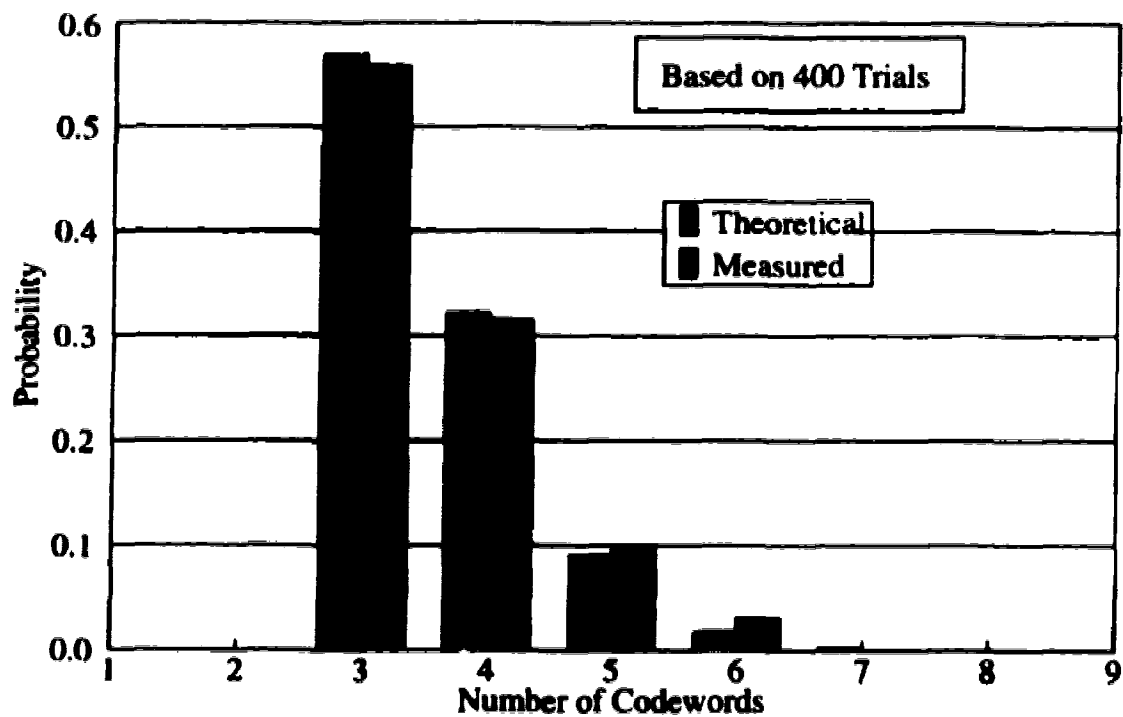


Figure 7.4 (d): Maximum average reframe time distribution for $C_R=2$ and $BER=0$

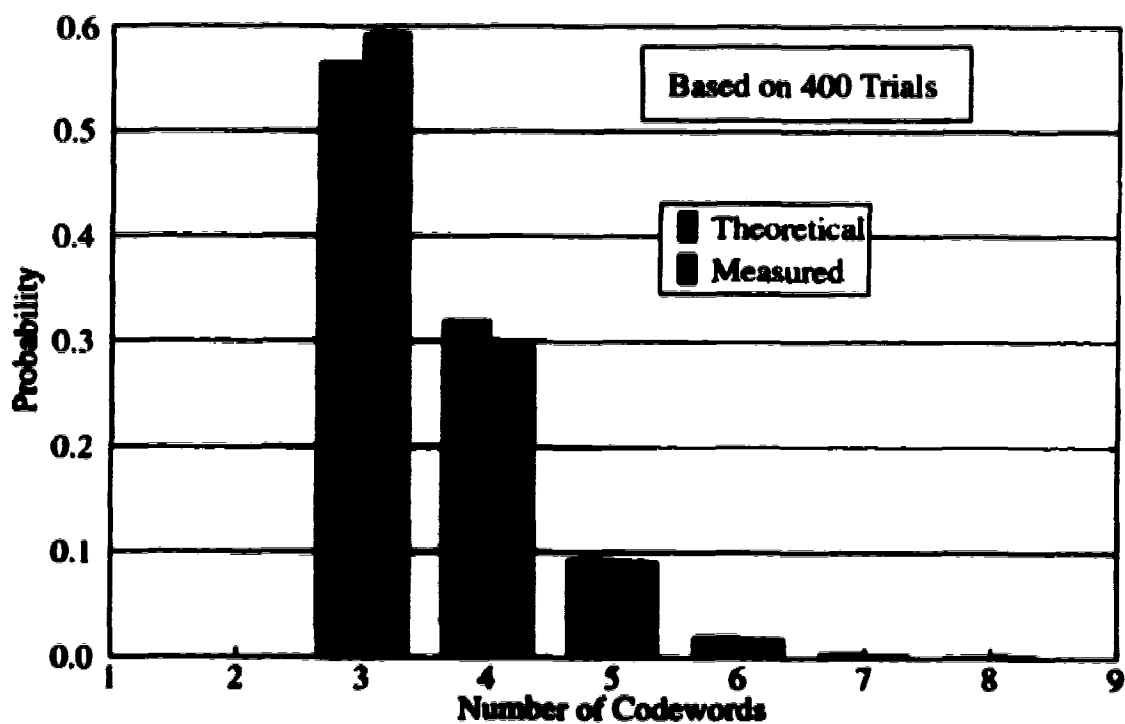


Figure 7.4 (e): Maximum average reframe time distribution for $C_R=2$ and $BER=10^{-6}$

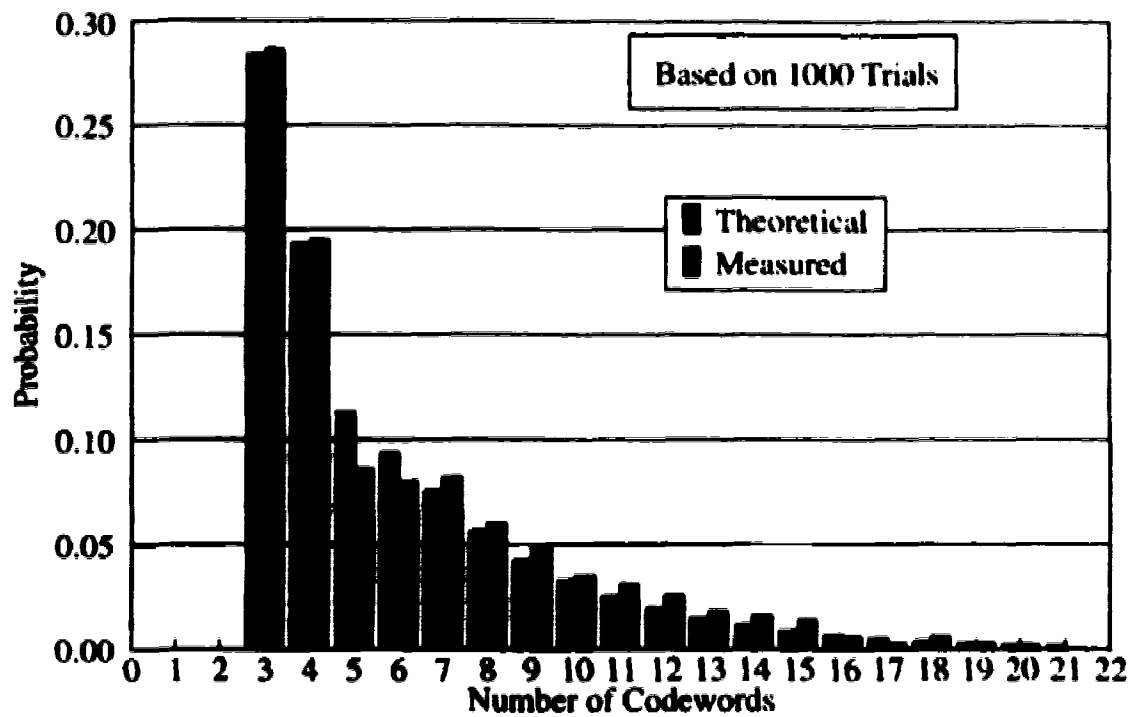


Figure 7.4 (f): Maximum average reframe time distribution for $C_R=2$ and $BER=10^{-4}$

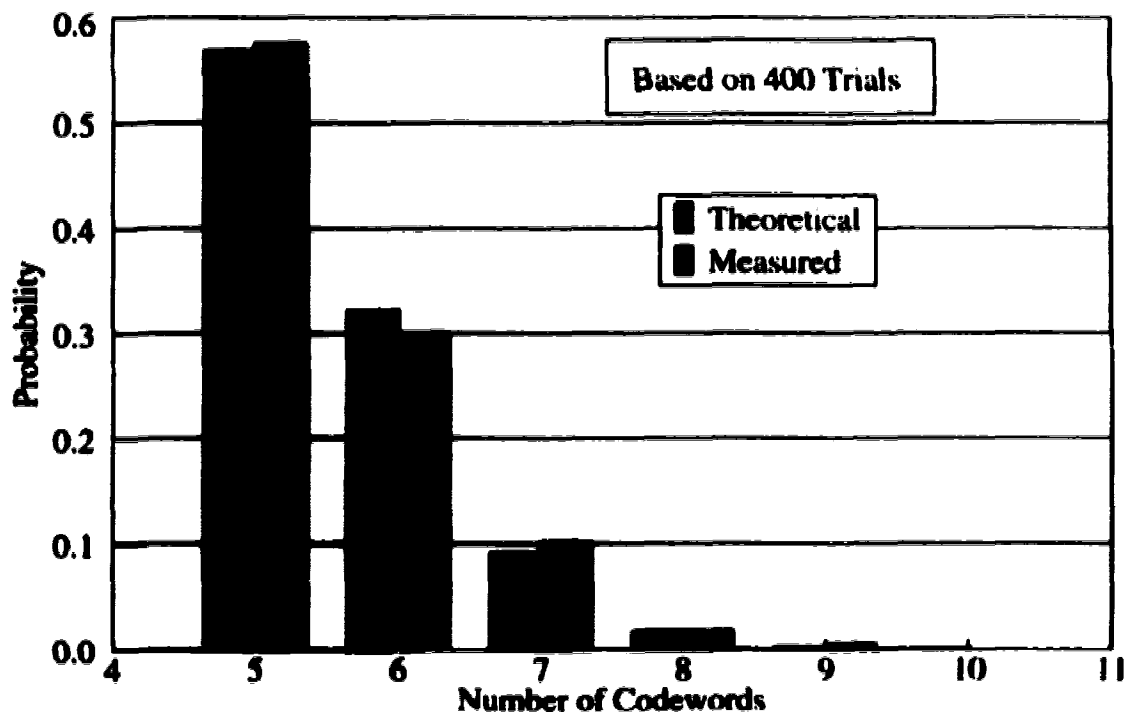


Figure 7.4 (g): Maximum average reframe time distribution for $C_R=4$ and $BER=0$

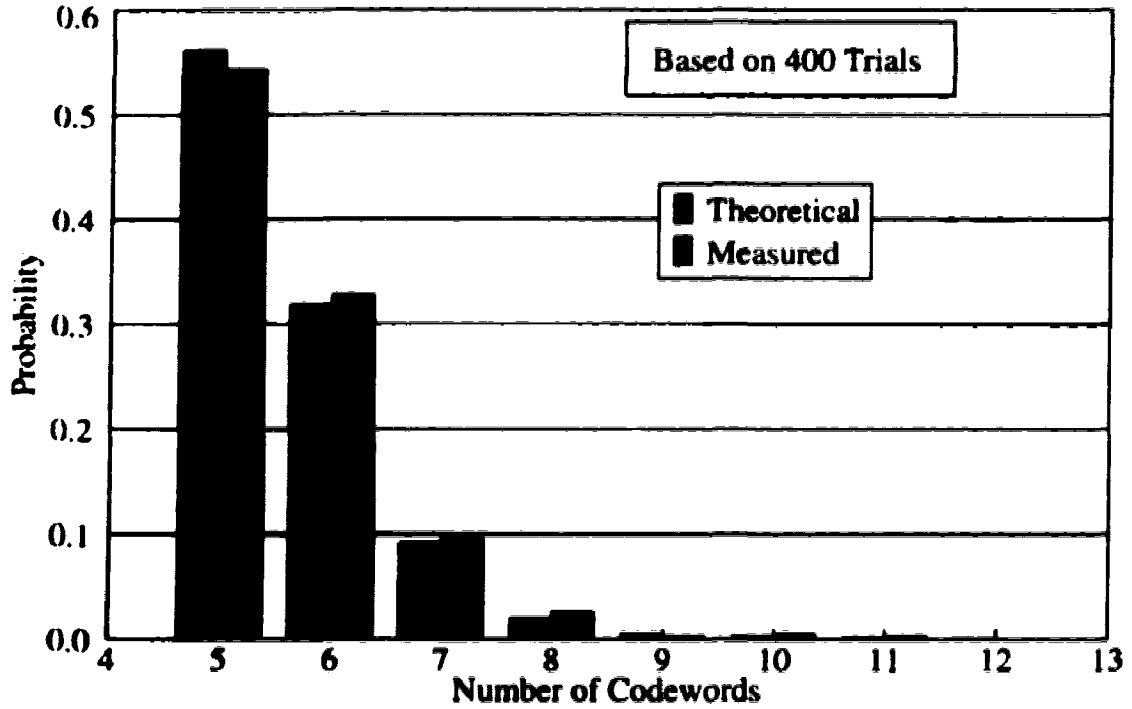


Figure 7.4 (h): Maximum average reframe time distribution for $C_R=4$ and $BER=10^{-6}$

To determine how closely the observed distributions of reframe time match the theoretical expectations in Figure 7.4, the Kolmogorov-Smirnov (K-S) goodness of fit test was used [14]. The K-S test involves a comparison between the theoretical and measured cumulative frequency distributions (CFD) to test the null hypothesis, H_0 , that the two distributions are the same. The statistic for the K-S test is the maximum absolute difference between the two CFD's, i.e:

$$D = \text{Max}_i | F_i - S_i | \quad (7.10)$$

where F_i is the cumulative relative frequency for the theoretical distribution and S_i is the cumulative relative frequency for the observed distribution. The decision to reject H_0 is based on D ; the larger D is, the greater the confidence that H_0 is false. The critical value (point at which the acceptance region is separated from the rejection region) for D at the $\alpha=0.05$ significance level is given by:

$$D_{\text{crit}} = \frac{1.36}{\sqrt{t}} \quad (7.11)$$

where t is the number of trials used to obtain the observed distribution. It is common to choose the $\alpha=0.05$ significance level for the K-S test as this means that there is no more

than a 5% chance of rejecting H_0 when, in fact, H_0 is true, i.e. 95% likelihood that the distributions are the same. Table 7.14 lists the values of D for each of the distribution pairs in Figure 7.4 along with the critical values of D . The confirmation threshold and BER level at which the data was gathered are also listed in Table 7.14.

Table 7.14: K-S Test results for DS1 Reframe Distributions

C_R	BER	t	D	D_{crit}
1	0	400	0.0318	0.0680
	10^{-6}	400	0.0372	0.0680
	10^{-4}	1000	0.0085	0.0430
2	0	400	0.0169	0.0680
	10^{-6}	400	0.0287	0.0680
	10^{-4}	1000	0.0376	0.0430
4	0	400	0.0144	0.0680
	10^{-6}	400	0.0185	0.0680

Note that in all cases $D < D_{crit}$. Thus in all cases, the null hypothesis that the two distributions are the same cannot be rejected. In other words, the observed data is consistent (at the 95% confidence level) with the hypothesis that it is the same as the given theory. From these results it is clear that the reframe circuitry in the ASIC has been correctly implemented and that its performance is as predicted.

7.3.3 DS3 Maximal Length Reframe Time Measurement

Figure 7.5 shows the theoretical reframe time distributions for C_R values of 1, 2, and 4, and for BER levels ranging from 0 to 10^{-4} for DS3.

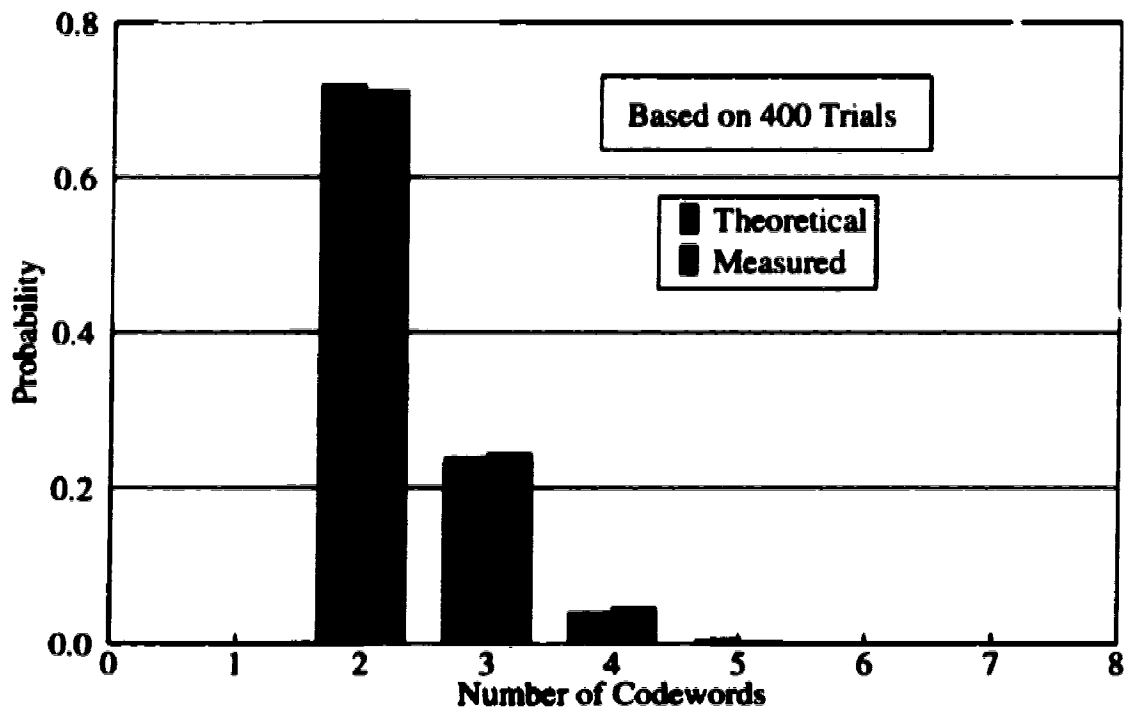


Figure 7.5 (a): Maximum average reframe time distribution for $C_R=1$ and $BER=0$

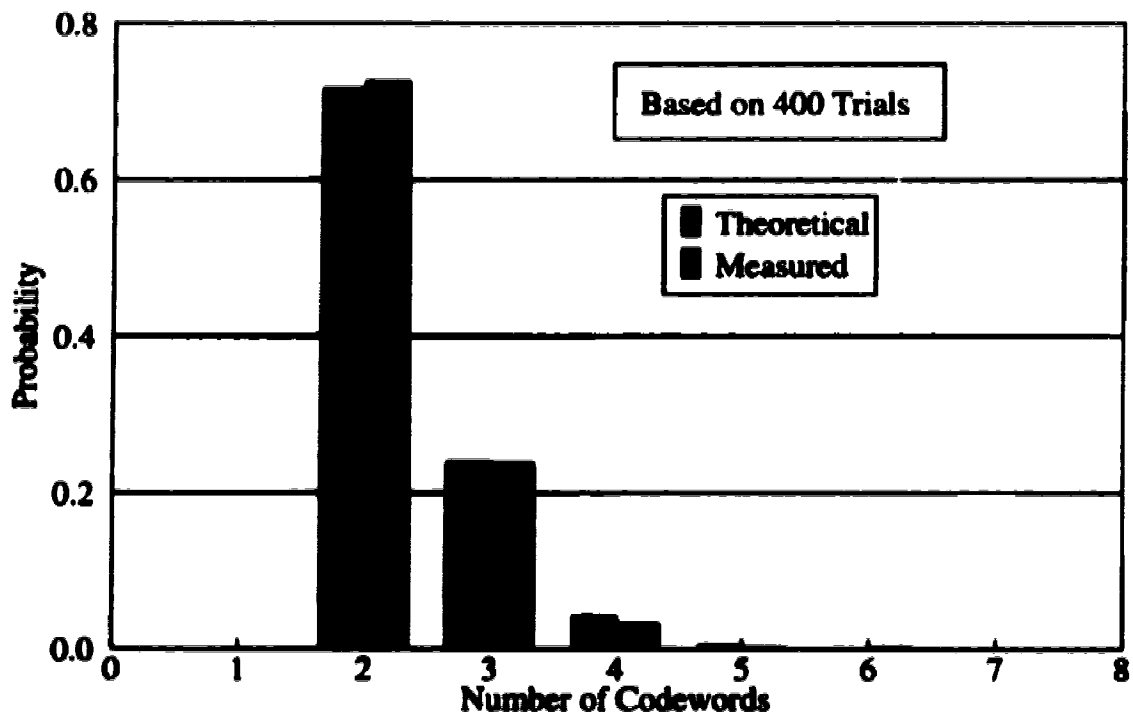


Figure 7.5 (b): Maximum average reframe time distribution for $C_R=1$ and $BER=10^{-6}$

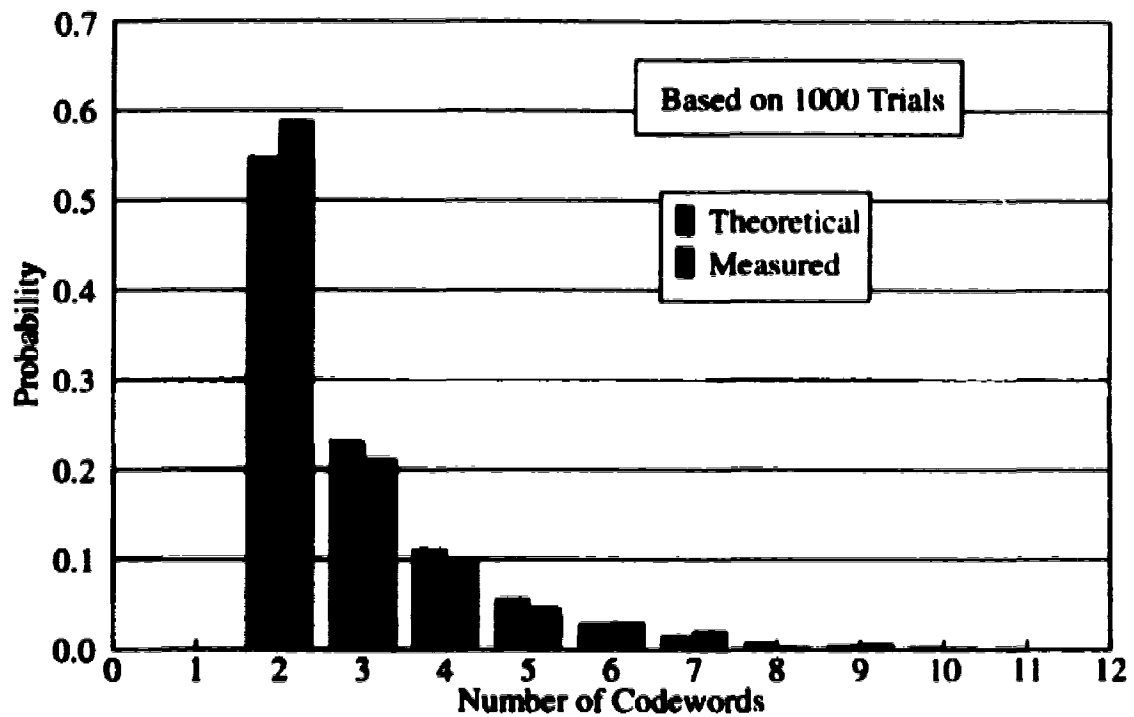


Figure 7.5 (c): Maximum average reframe time distribution for $C_R=1$ and $BER=10^{-4}$

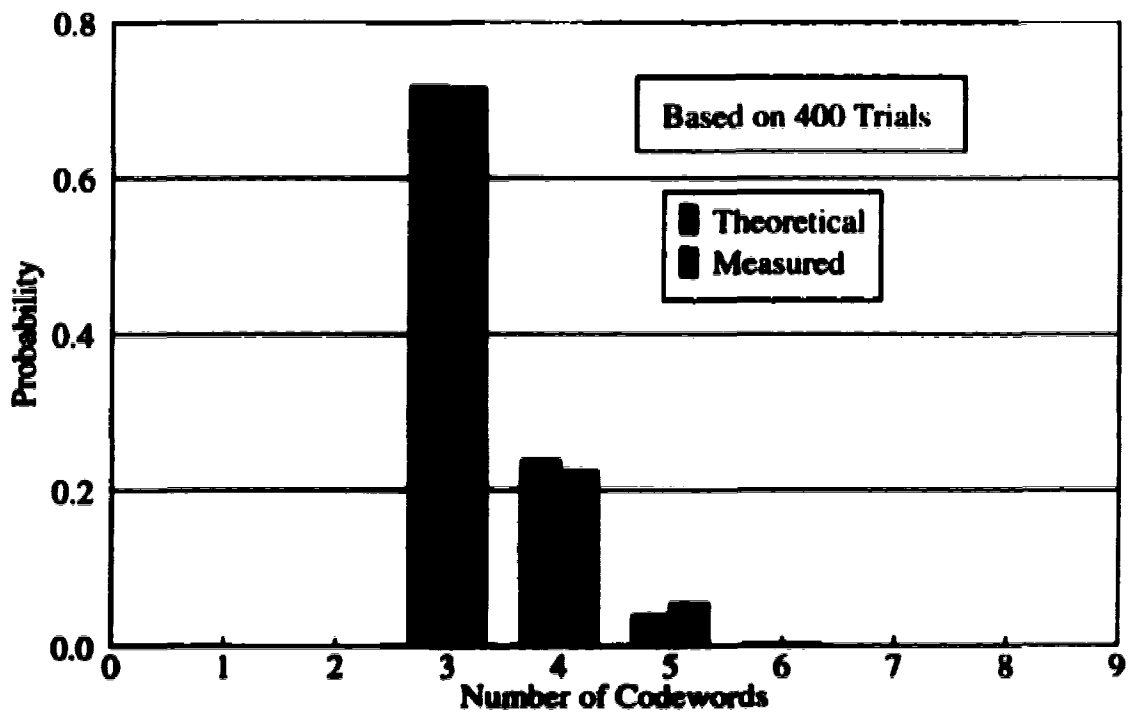


Figure 7.5 (d): Maximum average reframe time distribution for $C_R=2$ and $BER=0$

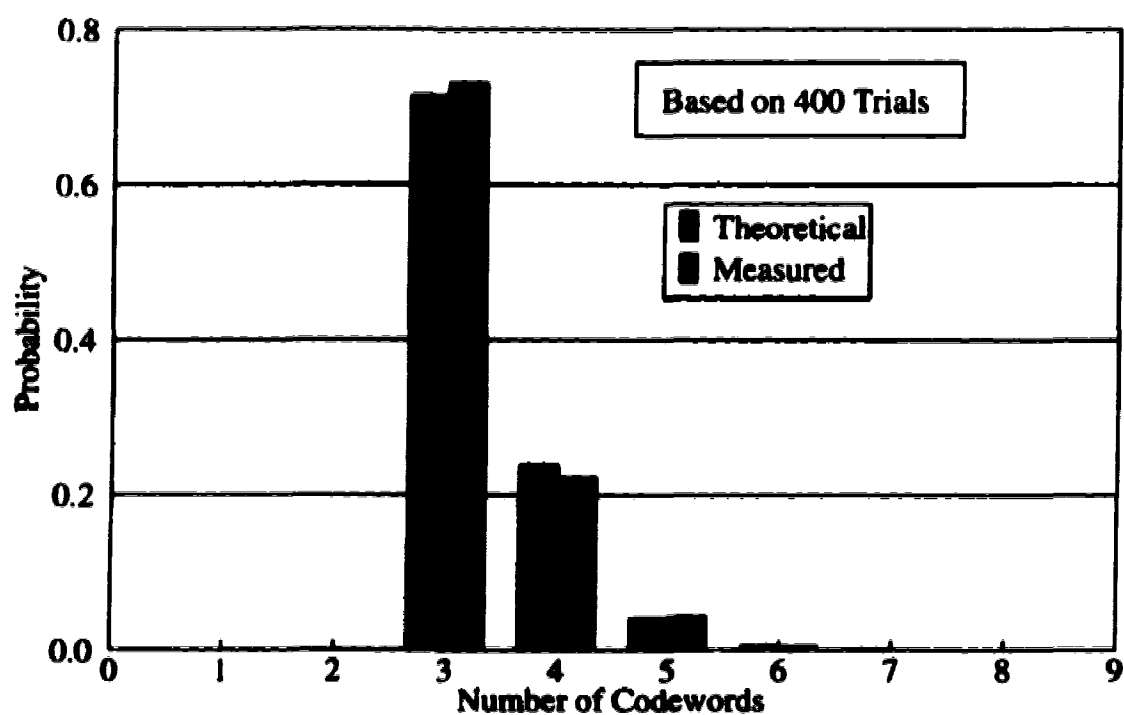


Figure 7.5 (e): Maximum average reframe time distribution for $C_R=2$ and $BER=10^{-6}$

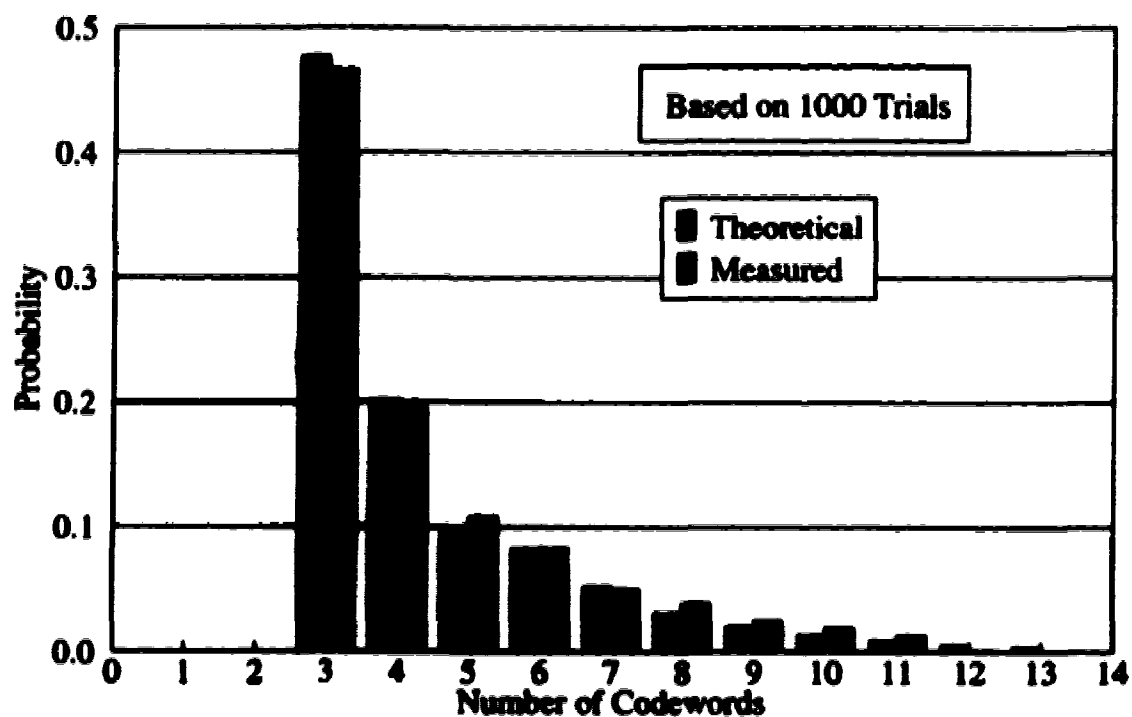


Figure 7.5 (f): Maximum average reframe time distribution for $C_R=2$ and $BER=10^{-4}$

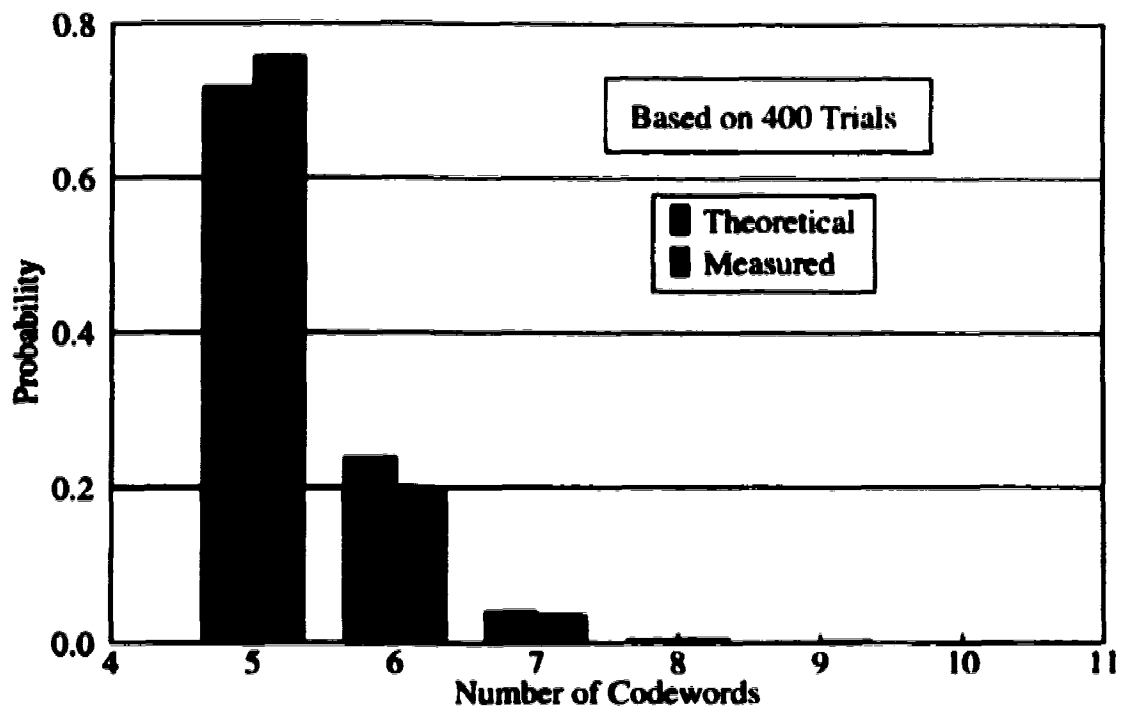


Figure 7.5 (g): Maximum average reframe time distribution for $C_R=4$ and $BER=0$

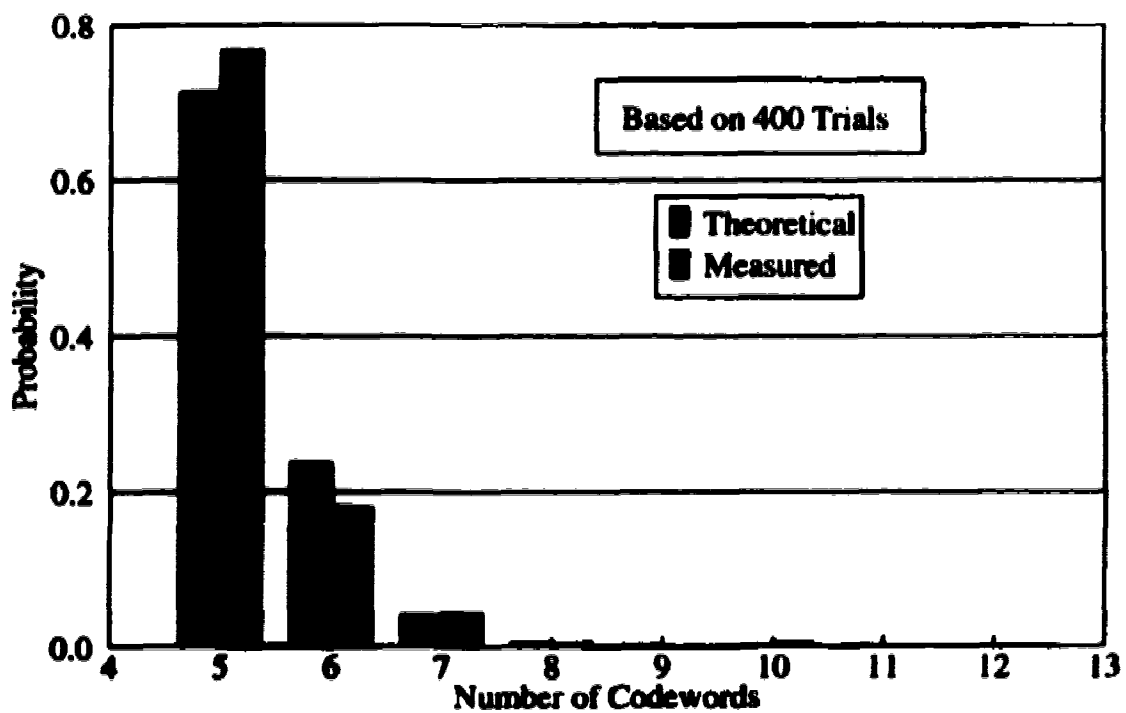


Figure 7.5 (h): Maximum average reframe time distribution for $C_R=4$ and $BER=10^{-6}$

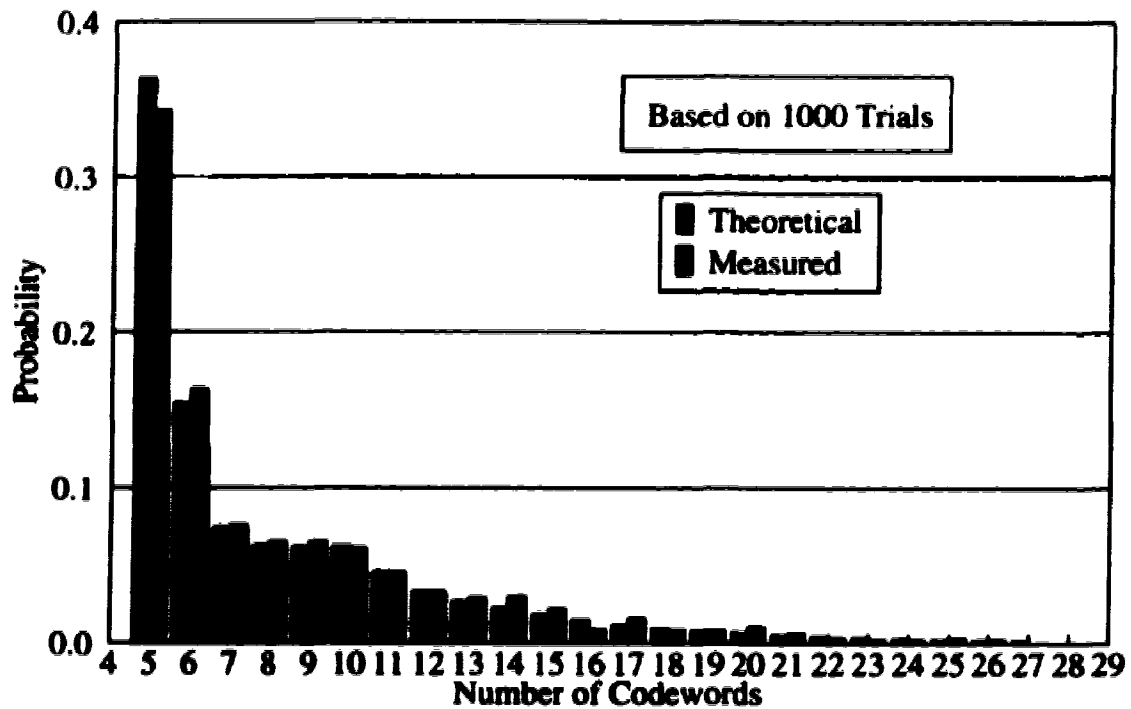


Figure 7.5 (i): Maximum average reframe time distribution for $C_R=4$ and $BER=10^{-4}$

As for the DS1 reframe time distributions, the K-S test is used to test the null hypothesis that the observed and the theoretical distributions are the same for DS3. The results of the K-S test at the $\alpha=0.05$ significance level for DS3 reframe time distributions are given in Table 7.15.

For each observed/theoretical distribution pair, the value of $D < D_{crit}$. Thus in all cases, the null hypothesis that the observed and theoretical distributions are the same cannot be rejected. These results verify correct implementation of the reframe algorithm in DS3 mode and prove that the reframe times are as expected.

Table 7.15: K-S Test Results for DS3 Reframe Time Distributions

C_R	BER	t	D	D_{crit}
1	0	400	0.00761	0.0680
	10^{-6}	400	0.00934	0.0680
	10^{-4}	1000	0.0412	0.0430
2	0	400	0.0157	0.0680
	10^{-6}	400	0.0153	0.0680
	10^{-4}	1000	0.0171	0.0430
4	0	400	0.0399	0.0680
	10^{-6}	400	0.0548	0.0680
	10^{-4}	1000	0.0205	0.0430

7.3.4 Comparison of FEC Framing and Conventional Framing (DS1)

In [3], the recommended values of C_O and C_R (to match the misframe and false-in-frame probabilities for the conventional DS1 framer to which DS1 FEC framing was benchmarked) are 5 and 2 respectively. The observed DS1-FEC framing performance for these threshold values, and BER 10^{-6} , is summarized in Table 7.16 along with the framing performance of the conventional DS1 framer analyzed in[3].

Table 7.16: Conventional and FEC Framing Performance for DS1

	Out-of-Frame Detection		Reframe		OFD + Reframe	
	Mean	99.5%	Mean	99.5%	Mean	99.5%
F-bit	0.91 ms	3.5 ms	51 ms	57 ms	51.9 ms	60.5 ms
FEC	9.0 ms	9.0 ms	5.3 ms	9.0 ms	14.3 ms	18 ms

From Table 7.16, it can be seen that the DS1-FEC framer with $C_O=5$ has longer OFD time than the conventional framer. However, the FEC framer with $C_R=2$ has significantly faster reframe time than its conventional counterpart. Even at a degraded BER level of 10^{-4} , DS1-FEC framing has a 99.5 percentile value for maximum average reframe time of 28.5 ms which is still faster than conventional framing. Thus the fact that the OFD time for DS1-FEC framing is longer than for conventional DS1 framing is not an issue because of

the large improvement in reframe search time, which results in an overall speedup of the complete OFD + Reframe process. Considering the sums of the mean values of the OFD and maximum average reframe times for conventional and FEC framing, an overall speedup of 3.6 is achieved as a result of FEC framing for DS1.

7.3.5 Comparison of DS3-FEC framing to DS3 Conventional Framing

The same values of $C_O=5$ and $C_R=2$ recommended for DS1-FEC framing in [3] are also recommended for DS3-FEC framing. Table 7.17 summarizes the performance of the DS3-FEC framer with these threshold settings at $BER=10^{-6}$. The performance of the conventional DS3 framer analyzed in [3] is also presented.

Table 7.17: Conventional and FEC Framing Performance for DS3

	Out-of-Frame Detection		Reframe		OFD + Reframe	
	Mean	99.5%	Mean	99.5%	Mean	99.5%
F-bit	27.6 μ s	106 μ s	1.36 ms	1.55 ms	1.39 ms	1.66 ms
FEC	182 μ s	182 μ s	101 μ s	151 μ s	283 μ s	333 μ s

As for DS1, the OFD time for FEC-derived framing for DS3 is longer than for conventional framing. However, this is offset by a considerable reduction in maximum average reframe time. This reduction in reframe time (due to speedup of search phase) yields for an overall average speedup of 4.9 as a beneficial side effect of DS3-FEC framing.

Chapter 8

Engineering the ASIC

The DS1/DS3 FEC ASIC is a large VLSI system. In this section, some of the design, test, and other engineering considerations which were not specific to either the FEC encoder and FEC decoder are presented. These are some of the additional design decisions or design results which were used to create this design as an ASIC in the LSI Logic C-MDE Design Environment.

8.1 Power Consumption / Heat Dissipation

For CMOS designs, power dissipation is based on three factors: the operating voltage (V_{DD}), the internal gate capacitance (C), and the frequency (f) of operation as shown in (8.1).

$$P = \frac{f \cdot C \cdot V_{DD}}{2} \quad (8.1)$$

For the DS1/DS3 FEC ASIC, the worst case power consumption comes when the system is in DS3 mode, operating at a frequency of 44.736 Mbit/s. The gate count of the FEC ASIC is heavily dominated by the two codeword delay buffers (one in the encoder, one in the decoder). The system clock fans out to each of the flip flops which make up these registers. Thus on every clock edge, each flip flop clocks in a new bit. In CMOS, energy is only dissipated when a logic gate changes state, i.e. there is virtually no static power dissipation. Thus the worst possible case for power consumption is when the input signal is a 10101010... toggle pattern. In this case, every flip flop in the delay registers will change its value on every clock edge.

The setup shown in Figure 8.1 was used to measure the actual power dissipation of the FEC ASIC in the laboratory under these conditions:

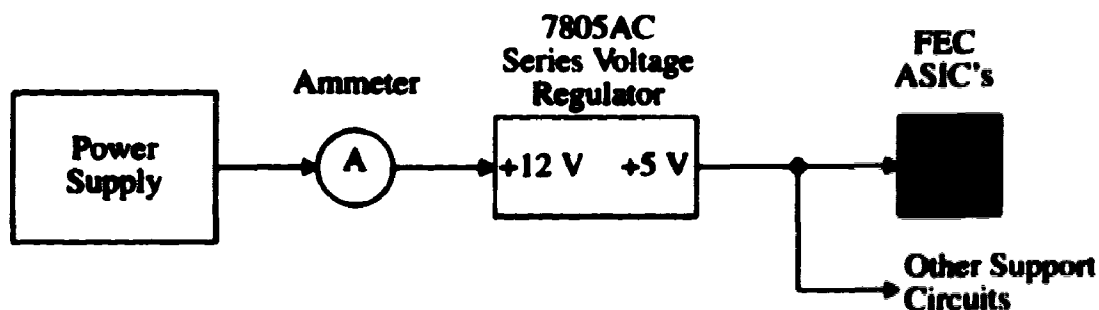


Figure 8.1: Power Dissipation Measurement Setup

The ammeter was used to measure the current draw from the power supply when the chips were (i) in the system and operating and (ii) when the chips were removed from the system. The difference between these two current measurements is the current drawn from the power supply by the FEC ASIC. As it is known the system is operating at +5 V, the power can be calculated. For DS3 the measured power dissipation by the FEC ASIC is calculated as follows:

$$P_{DS3} = (V_{DD})(I_{WITH\ CHIPS} - I_{WITHOUT\ CHIPS}) = (5\ V)(2.03\ A - 0.97\ A) = 5.3\ W \quad (8.2)$$

(8.2) neglects the small amount of power, due to leakage currents, consumed by the complementary parts (encoder, decoder) on the two ASICs which are not being clocked. Figure 8.2 shows how the value in (8.2) compares to simulated results:

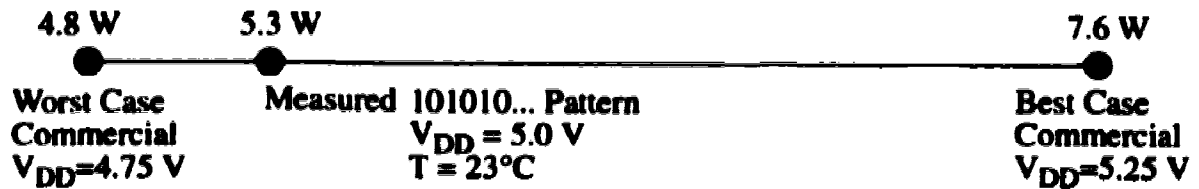


Figure 8.2: Measured and Theoretical Power Dissipation in DS3 mode

Note there is a small amount of current which is not transferred to the load by the regulator. This current is on the order of 4.3 to 6.0 mA. Not only is this small enough compared to the currents measured above to be neglected, but because the difference in current was measured the effect of this small term is reduced even more.

In DS1 mode the power dissipation was also measured as above to be 300 mW. This agrees very well with the simulated value of 380 mW computed for best case conditions (5.25 V operation).

Assuming power consumption in the FEC ASIC is dominated by the power consumed by the two codeword delay registers (i.e. the structures which occupy most of the device) we can verify that the CMOS power (8.1) is directly proportional to clock frequency. The ratio of DS3:DS1 clock frequencies is 44.736 MHz / 1.544 MHz = 28.97. The ratio of the length of each delay register chain in DS3 and DS1 modes is 1360/2316 = 0.587. Thus observed power consumption at DS1 scales to DS3 as follows:

$$P_{DS3} = (P_{DS1})(28.97)(0.587) = (0.3\ W)(17.01) = 5.1\ W \quad (8.3)$$

which agrees with (8.2) and verifies (8.1).

8.2 The Die

The total gate count of the FEC ASIC is 38,179 gates. This gate count requires the use of

die class L100094 in the LSI Logic process. This die can house up to 40,000 gates.

The FEC encoder and FEC decoder reside on the same die, even though they are independent devices. As clock trunks (Section 8.5) can only be routed vertically, it was decided to split the silicon real estate vertically, down the middle of the die. Arbitrarily, the encoder was selected to occupy the left half, and the decoder the right half.

8.3 The Package

Due to the high power dissipation, a package for the chip was selected which allowed efficient heat conduction from the chip through the package to the environment. This package is a 95-pin ceramic pin grid array (PGA) with cavity-down mounting. In most packages, the die sits on the bottom of the package and a thin cover is put on the top. In a cavity down package, the die is mounted against the top of the package and the cover is on the bottom (the same side as the pins). This allows for better heat conduction through the top of the PGA. To further assist in the removal of heat from the die, heat sinks were attached to the top of the package in the laboratory and fans were positioned to blow air across the package at approximately 700 linear feet per minute. This was sufficient to keep the junction temperature below critical value (155°C).

The package has many more I/O pins than are actually used in the design, but the 95-pin package was the smallest cavity-down ceramic PGA available from LSI Logic.

8.4 Bonding Diagram

The bonding diagram for the FEC ASIC is shown in Figure 8.3.

As noted in Section 8.2, the die has been divided into two portions, one part for the FEC encoder and one for the FEC decoder. Consistent with this arrangement, the encoder I/O's have been bonded out to the left side of the die and the decoder I/O's on the right side. Note the signal names shown on the diagram are the actual signal names used in the LSI Logic design environment and differ somewhat from the signal names used in this thesis.

The large bonding points along the inside of the package bonding region are the connections to the power and ground planes in the package. The L100094 die has a number of pre-assigned bonding pads which are reserved for connections to power and ground. In addition, additional ground pads must be assigned based on the power consumption of the device. Because of the high frequency of operation and hence the high power consumption of this design, 24 additional ground connections were required. These connections were placed at approximately even intervals around the perimeter of the die. As well, to reduce noise on the input clock signals, a ground pad is placed on each side of the high fan out

clock trunk drivers (Section 8.5) as required by LSI Logic design guidelines.

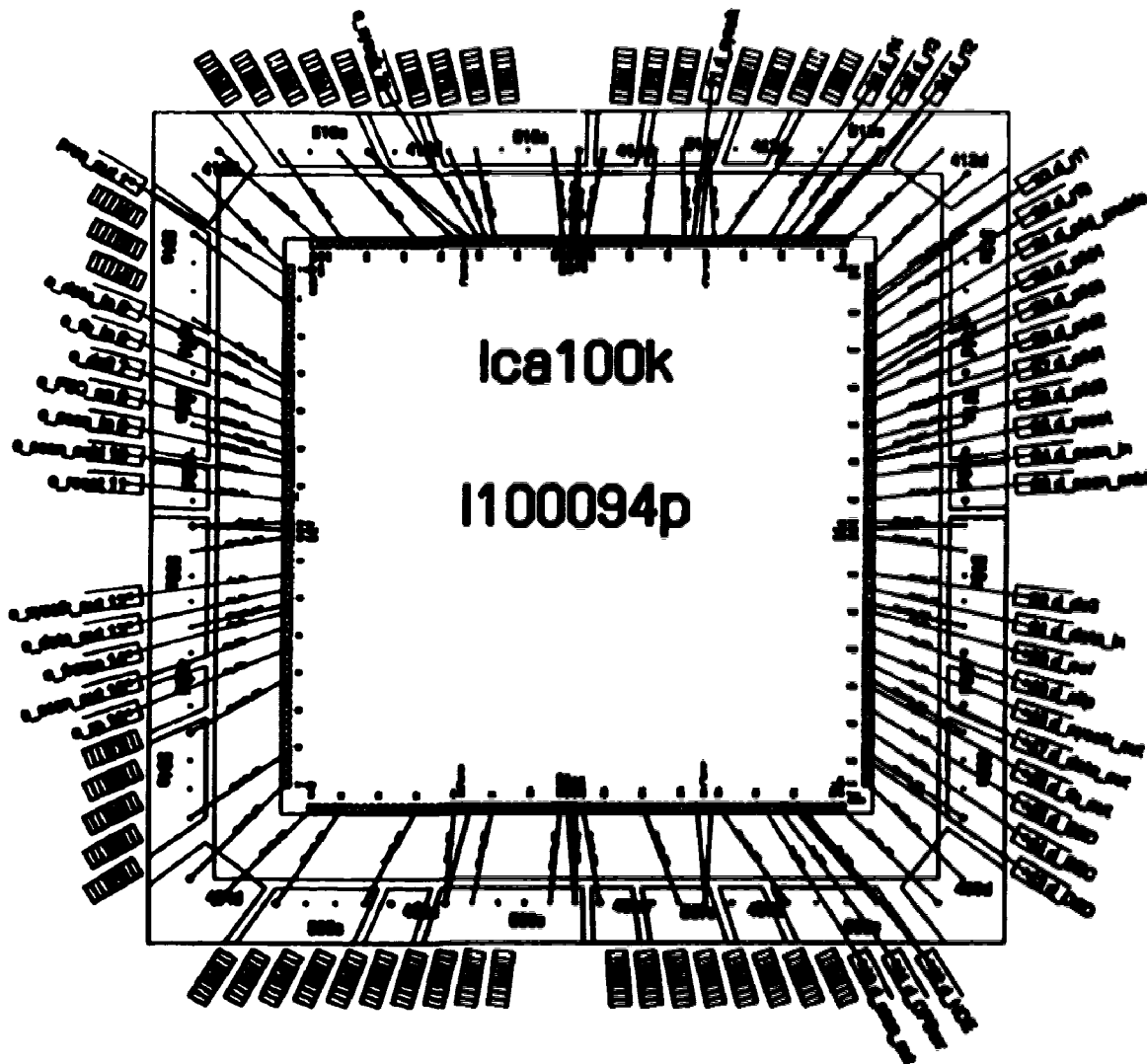


Figure 8.3: DS1/DS3 FEC ASIC Bonding Diagram

8.5 Clock Routing Strategy

In both the encoder and the decoder, the clock signal fans out to a large number of flip flops. For the encoder, this number is 2369. for the decoder, it is 2408. The large number of fan outs is due to the two codeword delay buffers in which each element must be clocked at the full signal rate. For designs where the clock signal fans out to more than

2000 locations, the clock routing scheme shown in Figure 8.4 is used:

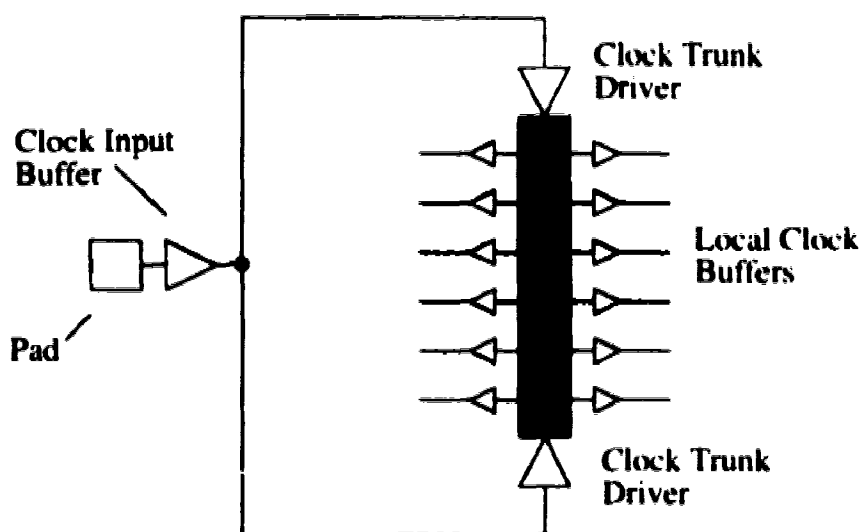


Figure 8.4: High Fan out Clock Routing Strategy

A clock trunking arrangement similar to that shown in Figure 8.4 exists for both the FEC encoder and the FEC decoder. Each respective clock signal is routed to two high drive buffers which drive a large clock trunk. The clock trunk uses six routing tracks filled in with metal. Each trunk drives local clock buffers which in turn drive the clock inputs of 10 flip flops. The encoder clock trunk drives 237 local clock buffers; the decoder trunk drives 241.

Because the FEC encoder and FEC decoder reside on the same die, the two clock trunks are placed in the center of the region occupied by each respective module on the die as shown in Figure 8.5

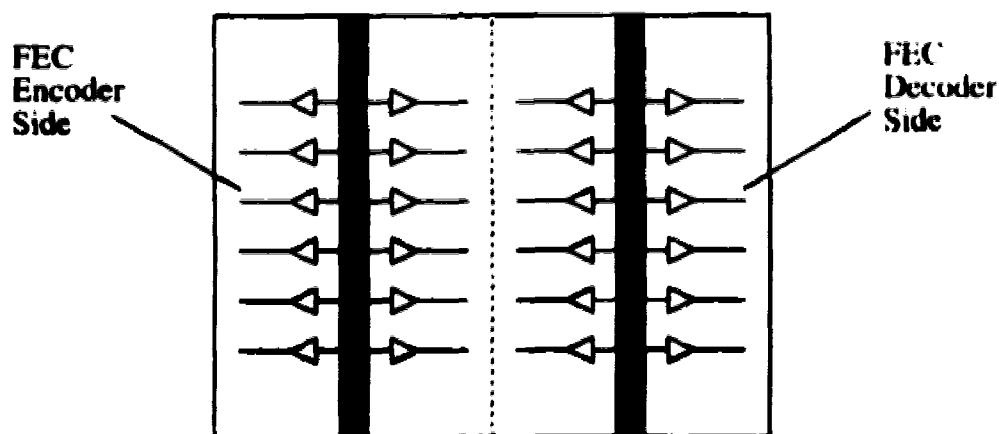


Figure 8.5: Clock Trunk Positioning on the Die

The clocking scheme, according to LSI Logic guidelines, guarantees less than 1.5 ns of skew on the clock line. The total latency (delay) from rising clock edge on the chip input to rising clock edge at flip flop inputs is around 13 ns. This latency doesn't affect the speed of the design. It just shifts the phase requirements for the clock and the data by 13 ns at the chip level inputs.

8.6 Overall Savings From Logic Minimization

As has been presented in Chapters 3 and 4, the finite state machines (FSM's) is the FEC encoder and FEC decoder are often functions of a large number of input variables, e.g. 12 bits of a counter. Thus traditional logic minimization techniques such as Karnaugh maps are not feasible. The minimized FSM realizations were obtained using logic minimization software. The optimal state assignments were determined using "mustang" [11] and the boolean function minimizations was obtained with "espresso" [11]. The total number of gates to implement all minimized logic functions in the design was 699. The un-minimized versions of these functions required 1049 gates. Thus, using the logic minimization software allowed a savings of 350 gates in the entire design.

8.7 Design For Testability

The design uses a full serial scan technique for high observability of internal nodes [15]. In the scan testing methodology, all flip flops in the design are converted to their scan equivalents. The scan equivalent flip flop of a conventional D-Flip flop is shown in Figure 8.6.

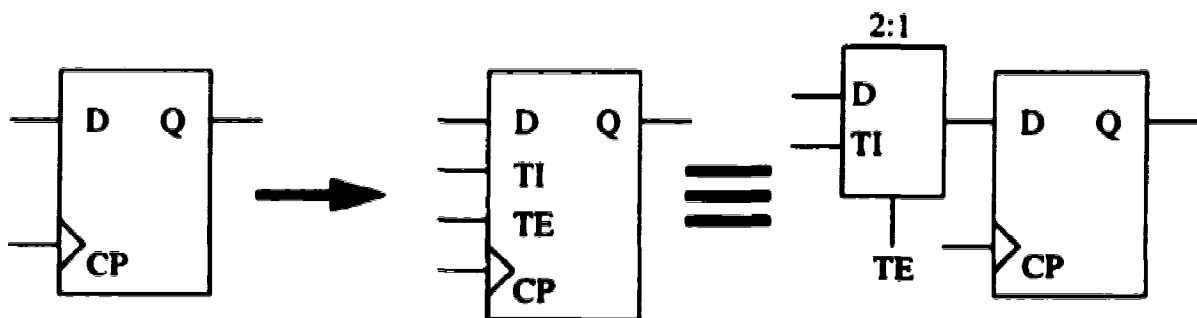


Figure 8.6: Scan Flip Flop Structure

Effectively, a scan flip flop is simply the conventional flip flop prefixed by a 2:1 multiplexer. The scan flip flop thus has two data inputs, the normal D-input and the TI-input (scan in). The inputs selection is controlled by the TE (scan enable) input. An example of how scan flip flops are connected in a design is shown in Figure 8.7 for a three flip-flop

system.

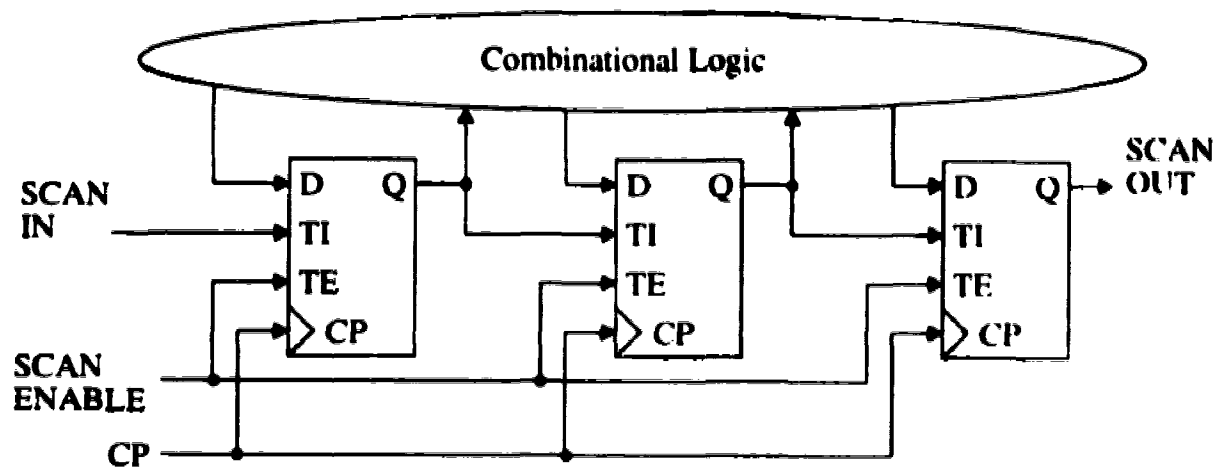


Figure 8.7: Example Three element Scan Chain

When scan enable is de-asserted, the scan flip flops function as normal flip flops, i.e the D-input is selected, and the flip flops interface with the combinational logic to implement desired functions. When the scan enable line is asserted, the TI input is selected as flip flop input. The TI input to the very first scan flip flop comes from a chip level input buffer (*SCAN IN*). As shown in Figure 8.7, the output of this flip flop is connected to the TI input of the next stage. The output of that stage is then connected to the TI input of the subsequent stage. In a larger design, all flip flops are connected in this manner. Thus in scan mode, the scan flip flops are connected into a shift register chain.

To use the scan chain for device testing, the ASIC is first placed in scan mode and a test vector pattern is then shifted into this chain to load up each flip flop with a preassigned value. The states on the flip flops are then applied to the surrounding combinational logic. The outputs from the combinational logic are applied to the standard D-inputs of the flip flops. The device is then placed in normal mode for one clock cycle to clock the values on the D-inputs into the flip flops. The ASIC is placed in scan mode once again and the resulting pattern which was clocked into the flip flops in normal mode is then shifted out of the scan chain. The resulting pattern is then compared with the expected output response.

The benefits of this testing strategy include the following: (i) high observability of internal nodes, (ii) high fault coverage, and (iii) automatic generation of test patterns to achieve high fault coverage. These benefits do not, however, come without costs. The costs of using the scan path technique to test the ASIC are as follows: (i) higher gate count from the additional 2:1 MUX components in each scan flip flop, (ii) higher fan outs on each Q-output of each flip flop, (iii) performance delay through 2:1 MUX, (iv) requires three addi-

tional chip pins, "scan enable", "scan in", and "scan out".

In the FEC ASIC design, separate scan chains were developed for both the FEC encoder and decoder modules as they are clocked from different input pins. This requires 6 additional chip pins. The codeword delay buffers are excluded from the scan chain, however for obvious reasons: these modules are shift registers with their inputs and outputs available as chip level inputs and outputs. Thus they can be tested as scan chains in themselves. This greatly reduces scan chain length and testing time. However, any flip flops in these chain which feeds combinational logic was included in the scan chain so that this logic can be tested.

The scan test vector set to test the FEC ASIC was generated automatically and consisted of 643 vectors which achieved a fault coverage of 100% of all testable faults

8.8 Automatic Test Equipment (ATE) Vectors

The test vectors automatically generated by LSI Logic design software for serial scan give high fault coverage but do not verify design functionality. To verify the operations of reframing, out-of-frame detection, and error correction requires processing of a number of DS1-FEC or DS3-FEC frames at the decoder. Each of these frames is either 2316 (DS1) or 1360 (DS3) bits long so the test vector set easily becomes very large. Thus careful planning was put into the test vector design to make the set manageable. The final functional vector set was 15968 vectors long and tests the system in DS3 mode. Figure 8.8 shows the structure of this functional test vector suite:

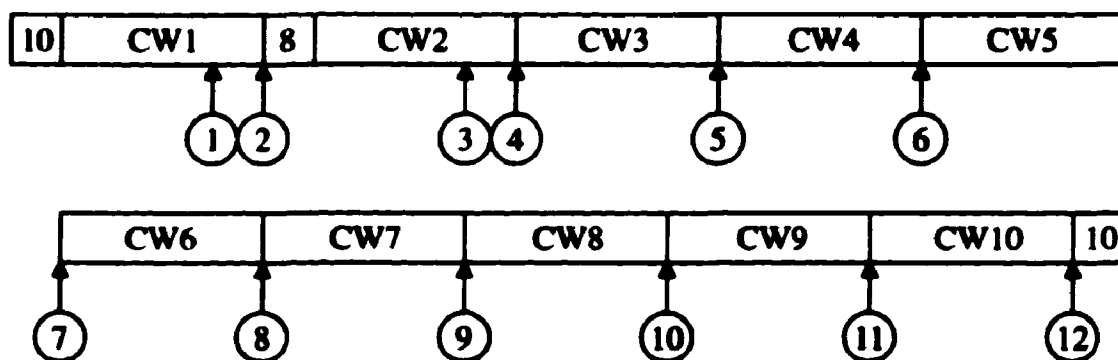


Figure 8.8: Functional Test Vector Suite

In Figure 8.8, CW_i is a DS3 codeword and the "10" and "8" blocks are blocks of ten and eight random bits respectively. There is a single bit error in CW_4 . CW_6 contains only 1359 bits, i.e. it is missing a bit. This is akin to gapping out a clock cycle at the decoder in

the laboratory test environment (Section 6.7), and will force the decoder OOF. All other codewords are error free. In the following discussion, the operation of the FEC decoder on this data set is presented. Note that for this vector set $C_R=1$ and $C_O=2$.

At point 1, the decoder has clocked in a complete 1360-bit data block and is ready to begin search for the codeword boundary. Point 1 is not at the end of a valid codeword, so a non-zero syndrome is decoded there. Thus, the decoder slips through the data stream until it reaches point 2, the end of a valid codeword, where a zero syndrome is decoded. Point 2 is dwelled on as a frame candidate and the syndrome at this position one codeword time later (Point 3) is considered. However, point 3 is not the end of a valid codeword either so a non-zero syndrome is decoded and the frame candidate is rejected. The decoder then slips through the data until a location is found which decodes a zero syndrome (Point 4). This time when the syndrome at one codeword time later (Point 5) is examined it is the end of a valid codeword and hence is zero. Because $C_R=1$, this single confirmation is sufficient for the decoder to declare INF at point 5.

As note previously, CW4 contains a single bit error. Thus the syndrome at point 6 is non-zero. Because $C_O \neq 0$, the system remains INF. The syndrome information is processed to correct the single bit error. CW5 is error free so a zero syndrome is decoded at point 7, resetting the OOF detection process.

Because a bit has been stolen from CW6, the system will clock in the first bit from CW7 as part of the CW6 codeword. This will result in a non-zero syndrome at point 8. These non-zero syndromes persist at points 9 and 10. Because $C_O=2$, the second confirmation at point 10 causes the system to declare OOF. The decoder then slips until it finds a location which decodes a non-zero syndrome. This occurs at point 11. Because this is the true frame boundary, the system will declare INF after the confirmation at point 12.

Thus, this function vector suite tests the following functions of the FEC decoder: (i) scan for frame candidate, (ii) rejection of a false frame candidate, (iii) resume scan for new frame candidate, (iv) confirm candidate is the correct frame boundary and declare in-frame, (v) decode an error free codeword and initiate no error correction activity, (vi) decode a codeword containing a single bit error, correct the error, and increment OFD counter, (vii) decode a second error free codeword and clear OFD counter, (viii) decode a codeword which is missing one bit to cause an OOF condition, and finally (ix) confirm and declare OOF and then re-frame again.

During this period, the FEC encoder encodes 10 complete DS3 codewords. It is verified that the checkbits for these codewords are correctly computed and inserted into the proper

F and C-bit positions.

The toggle coverage of the device by this vector set alone (not including scan vectors) was 99.36% of all nodes. This high coverage is noteworthy and is attributed to the almost totally serial-processing nature of this design. After a single codeword is shifted in, every flip-flop in the encoder and decoder delay registers is toggled and the main system counters and decoding logic have experienced their complete address range. While high toggle coverage does not necessarily imply high fault coverage, it is clear that this functional test vector set exercises the modules well. Combined with the high fault coverage of the scan vectors, this ATE testing strategy will be quite effective in identification of defective production units. All 10 prototypes which were fabricated passed all scan and functional test vectors.

The FEC ASIC in DS3 mode operates at a clock frequency of 44.736 MHz (22.35 ns clock period). However, the ATE used can operate at a maximum frequency of 20 MHz. To verify the ASIC could operate at speed, a "representative delay path" was brought out to a chip pin. This path is one of the longest in the design, passing (un-clocked) through eight levels of logic before reaching the chip output pin. It is noted as "e_rp" on the bonding diagram. The simulated delay on this signal line from the time the clock signal is applied to the chip clock pin and the time "e_rp" is stable is around 21 ns under ATE test environment conditions. This includes the approximately 13 ns latency introduced by the clock trunking scheme (so the actual delay on this path is around 8 ns under ATE environment conditions). It is expected that any change in the actual delay observed on this pin will be consistent with the other internal signals on the ASIC. Thus as long as this signal meets its simulated timing requirements, the all internal signals are expected to also meet theirs. The timing requirements for the representative path are tested by strobing the signal 21 ns after the clock signal is applied to the ASIC for each vector in the functional vector suite.

Chapter 9

Summary and Conclusions

9.1 Summary of Results

The incorporation of an FEC code into the DS1 and DS3 signal formats using the primary framing (F) bits to transport to FEC checkbits has been successfully implemented in a 38k gate ASIC using LSI Logic's 1.0-micron gate array process. The main accomplishments are summarized as follows:

A successful design of the FEC encoder was achieved. This device correctly maps the incoming conventional DS1 Superframes or DS3 frames into DS1-FEC or DS3-FEC codeword blocks respectively, computes the FEC checkbits for each data block, and inserts the checkbits into the distributed F-bit positions.

The FEC decoder design has also been validated and fully meets the design intent. The FEC decoder computes the syndrome for the received word on a bit-by-bit basis using the CCSC technique to modify the existing syndrome information at each bit time thereby deriving the syndrome for the data after each new bit is shifted in. The syndrome information is used by the frame alignment unit to achieve rapid frame alignment. Once frame alignment is established, the syndrome information is used to locate and correct single bit errors in the received codewords. The conventional framing sequences of the DS1 Superframe and the DS3 frame are also correctly restored to their conventional format by the FEC decoder.

The FEC encoder and FEC decoder are configurable for operation on DS1 or DS3 signal formats. The internal structures are re-configurable, controlled by a mode select line, for efficient re-use of internal logic structures.

Modern logic design software for FSM state assignment and for boolean equation minimization, yielded minimized implementation of the sequencers and FSM's in the encoder and decoder, saving 350 out of a possible 1049 gates subject to minimization.

A full scan chain methodology was used in design for testability. This permitted high fault coverage (100% of all testable faults) in 643 scan test vectors. The codeword delay buffers were excluded from the scan chains and tested independently, with the exception of flip flops at parallel tap points in these registers which drive combinational logic. The latter were also included in the scan chains. A compact set of functional test vectors was also

presented to test the design functionality using automatic test equipment. All 10 prototypes which were received passed all of the ATE tests.

The laboratory test environment was developed for design validation and to characterize the error correction and framing performance of the FEC ASIC. The following is a summary of the properties which were confirmed by test and measurement:

- Single bit error in any position of the DS1-FEC or DS3-FEC codeword is correctable.
- All even weight multiple errors are detectable in DS3 mode. (No error extension).
- 43.7% of higher order errors (>1) are detected in DS1 mode. (At most one additional error is introduced in codewords containing a higher order error which was not detected).
- 33.7% of higher order errors (3, 5, 7, ...) are detectable in DS3 mode
- For channel BER $< 10^{-5}$, the BER improvement from FEC coding in DS1 mode is approximately : $2963 \times \text{BER}^2$.
- For channel BER $< 10^{-5}$, the BER improvement from FEC coding in DS3 mode is approximately by: $1359 \times \text{BER}^2$.
- DS1 maximum average reframe time is 5.3 ms for $C_R=2$ (9.6 times faster than conventional F-bit framing)
- DS1 OOF detection time is 9.0 ms for $C_O=5$ (10 times slower than conventional F-bit framing)
- DS1 Overall OOF detection + Reframe time is 14.3 ms average (3.6 times faster than for conventional F-bit framing)
- DS3 maximum average reframe time is 101 μs for $C_R=2$ (13.5 times faster than conventional F-bit framing)
- DS3 OOF detection time is 182 μs for $C_O=5$ (6.6 times slower than conventional F-bit framing)
- DS3 Overall OOF detection + Reframe time is 283 μs average (4.9 times faster than for conventional F-bit framing)

Some additional details about the ASIC are as follows:

- Measured power consumption is 5.3 Watts in DS3 mode, 300 mW in DS1 mode
- LSI Logic Die Class L100094 (Max 40k gates)
- 95-pin ceramic PGA package

9.2 Further Work

9.2.1 Towards a Production Device

The ASIC design in this work was a prototype. Numerous sub-circuits and pinouts were employed to support the research-oriented aims of the project. If it is desired now to take this design into production, the following changes could be made to the design to reduce the gate count, allowing a smaller die, reduced power consumption, and possibly a smaller plastic package. The first is to implement the codeword delay buffers with a circular RAM buffer design approach rather than a true flip flop shift register. The RAM would be indexed by two pointers, NEW and LAST as shown in Figure 9.1

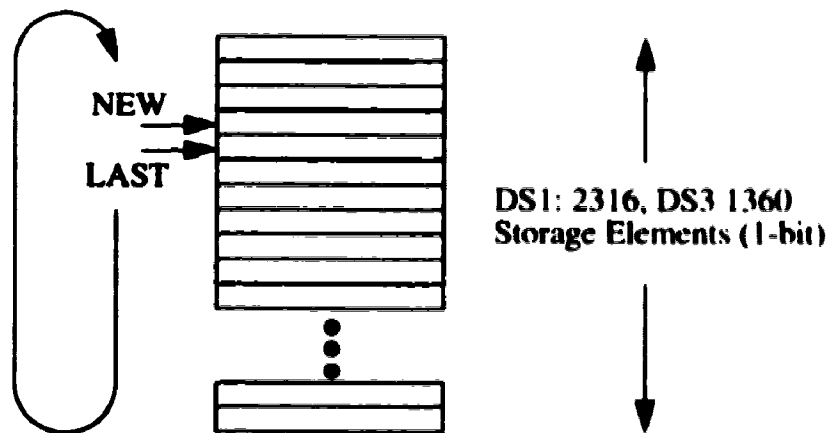


Figure 9.1: Cyclical indexed RAM implementation of delay registers

Pointer NEW would always point to the address to which the next bit arriving in the system would be written. Pointer LAST would point to the address containing the oldest data bit. Pointer NEW would chase Pointer LAST around the RAM cyclically. This strategy is more electrically efficient than the serial shift register implementation using flip flops because (i) each RAM cell has fewer transistors than a D-flip flop, thus reducing the size of the delay buffer structure, and (ii) only one read and one write of the RAM is required in each clock cycle, so the number of gates changing is very small, hence the power consumption will be lower, even though the RAM will dissipate some static power. This method works well for the encoder delay buffer which has only a serial input and serial output.

In the FEC decoder, the large number of parallel taps required to implement CCSC does not lend itself to using a single RAM structure. However, smaller RAM's could be used to implement the shorter delay segments between parallel taps in the existing decoder delay buffer. In this case, a combination of RAM and flip flop registers would be used. A standard shift register would be used in those cases where the number of gates to realize the

RAM and the address decoding logic for the two pointers exceeds the gate count of a simple shift register to implement the same delay.

Another recommendation to modify the design for production would be to combine all of the research-oriented error detection output signals (e.g. SED, SEC, DED, HOE, OPBErr) into one output signal which simply indicates that an error has occurred or been detected. In addition, the C_R , C_O values could now be hard-wired into the design, eliminating 10 input pins. Both of these measures would reduce pin count, potentially allowing a smaller package to be used.

9.2.2 Towards SONET FEC

Finally, the concept of using the framing bits to transport the checkbits of and FEC code can be applied to other signals which are prevalent in the telecommunications transport industry, for example the SONET STS-1 or STS-3 signals. The development would require the design of related codes to cover an STS-1 or STS-3 frames. The CCSC compensation functions for these formats could then be derived in a manner similar to that in Chapter 4. The only difficulties foreseen at this point for CMOS implementation of FEC framing for these format surround STS-3. As this signal format has a bit rate of 155.52 Mb/s [7], only 6.43 ns are available for processing each bit. This means that the design would need to be generously pipelined to effect the required processing. Also, because the power consumption in CMOS is directly related to the system clock frequency, the power consumption of an ASIC implementing FEC framing for STS-3 would be high and likely require special care in package selection and operating environment.

References

- [1] Grover, W. D.: 'Forward error correction in dispersion-limited lightwave systems', *J. Lightwave Technol.*, vol. 6, May 1988, pp. 643-654
- [2] Grover, W. D.: 'Effect of error correcting code using DS3 framing bits on measured dribble error pattern of 565 Mbit/s fibre optic transmission system', *Electronics Letters*, Vol. 28, No. 20, September 24, 1992, pp. 1869-1870
- [3] Cheung, M. Y.: 'Combined framing and FEC coding for digital multiplex signals', M. Sc. Thesis, University of Alberta, 1991
- [4] Fishman, D. A.: 'Elusive bit-error-rate floors resulting from transient partitioning in 1.5- μ m DFB Lasers', *J. lightwave Technol.*, vol. 8, no. 5, May 1990, pp. 634-641
- [5] Choi, D.: 'Frame alignment in a digital carrier system - a tutorial', *IEEE Comunciations Magazine*, February 1990, pp. 47-54
- [6] Telecom Canada: 'Digital Network Notes', 1983, pp. 5-9 to 5-17
- [7] Molloy, K.: "ABC's of digital formats: do they spell i-n-c-o-m-p-a-t-i-b-l-e for the variety of DS3 terminals now out in the field?", *Telephone Engineer and Management*, vol. 93, no. 12, pp. 70-75, 141
- [8] Stremler, F. G.: *Introduction to Communication Systems*, Third Edition, Addison-Wesley, 1990
- [9] Lin, S., Costello Jr., D. J.: *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, 1983
- [10] Cheung, M. Y., Grover, W. D., Krzymien, W. A.: 'Combined framing and FEC coding for multiplex signals,' Proceedings 1991 *IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, Victoria, B. C., May 1991, pp. 205-210
- [11] Octtools 5.1, Electronics Research Laboratory, University of California, Berkeley, October 1991]
- [12] Weslowski, J. A.: 'Test Environment for the DS1/DS3 FEC/Framing ASIC Project Report', TRLabs TR-93-07, September 1993
- [13] Jeruchim, M. C.: 'Techniques for estimating the bit error rate in the simulation of digital communication systems', *IEEE Journal on Selected Areas in Communications*, vol. SAC-2, no. 1, January 1984
- [14] Harnett, D. L.: *Statistical Methods*, Third Edition, Addison-Wesley, 1982

- [15] **McCluskey, E. J.: *Logic Design Principles*, Prentice-Hall, 1986**
- [16] **Cheung, M.Y., Grover, W.D., Krzymien, W.A.: "Combined FEC coding and framing for DS-1 and DS-3 signal formats", (Submitted to IEEE Trans. Commun.)**
- [17] **Dunn, D. A., Grover, W. D.: 'A novel ASIC for combined framing and Error Correction Coding in DS1 and DS3 signal formats', Proceedings: *Canadian Conference on Very Large Scale Integration 1993*, November 1993**

Appendix A

FEC Encoder / Decoder Support Modules

Appendix A

FEC Encoder/Decoder Support Modules

A.1 D - Flip Flop with Synchronous Load and Shift Enable

In this section, the construction of the D-flip flop with synchronous load and shift enable is presented. This flip flop is used in both the encoding register, the parity bit computation register and the holding register in the FEC encoder. The schematic diagram for the flip flop is shown in Figure A.1

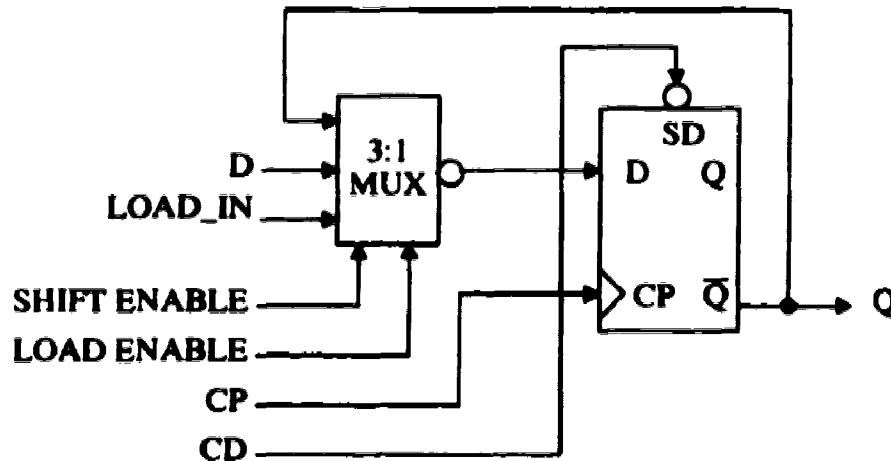


Figure A.1: Schematic diagram of D-flip flop with synchronous load and shift enable

The following are the inputs to the flip flop:

- *D*: standard data input of a D-flip flop
- *LOAD_IN*: synchronous load input data
- *SHIFT_ENABLE*: enable shift of data from *D*-input to *Q*-output
- *LOAD_ENABLE*: enable synchronous load of data from *LOAD IN* input to *Q* output
- *CP*: clock input
- *CD*: asynchronous reset

The flip flop has only the *Q* data output. The operation of the flip flop for all possible control signal inputs is given in Table A.1.

Table A.1: Next state table for D-flip flop with synchronous load and shift enable

LOAD ENABLE	SHIFT ENABLE	Q
0	0	Q
0	1	D
1	X	LOAD_IN

Note that the \bar{Q} output of the basic D-flip flop is used because only an inverting 3:1 MUX is available in the LSI Logic macrocell library.

A.2 Catch C-Bit Module

Figure A.2 shows the circuit used to trap the valid C-bits.

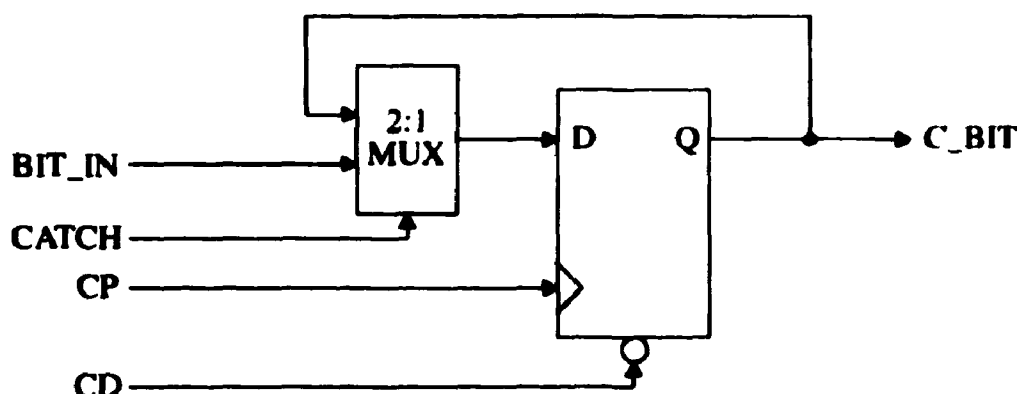


Figure A.2: Module for Sampling C-bit

The inputs to this module are as follows:

- **BIT_IN**: the value of the bit which is currently emerging from the delay buffer (including pipeline delay re-times) after error correction
- **CATCH**: a control signal indicating that **BIT_IN** is a valid C-bit and that its value should be sampled.
- **CP**: system clock
- **CD**: asynchronous reset

The circuit has a single output, **C_BIT**, which contains the value of the last valid C-bit which was seen.

When **BIT_IN** is not a valid C-bit, **CATCH** is de-asserted (low) and the a-input of the

MUX is selected. Hence the flip flop will hold its current value on the next rising clock edge. However, at bit positions 170 and 850 in the DS3-FEC codeword, *BIT_IN* is a valid C-bit. The *CATCH* signal is asserted (high). This routes *BIT_IN* to the d-input of the flip-flop so that its value is transferred to the q-output on the next rising clock edge. Hence the value of the valid C-bit for the frame is stored on the *C_BIT* output of the block and can be used by the parent block to replace those bits occupying the two C-bit positions of the DS3 frame used to carry FEC checkbits.

A.3 5-Bit Magnitude Comparator

Additional information on the 5-bit comparator used in the FAU for comparison of the FAU confirmation counter to the C_R or C_O confirmation thresholds is presented in this section. Figure A.3 shows the schematic for the 5-bit comparator:

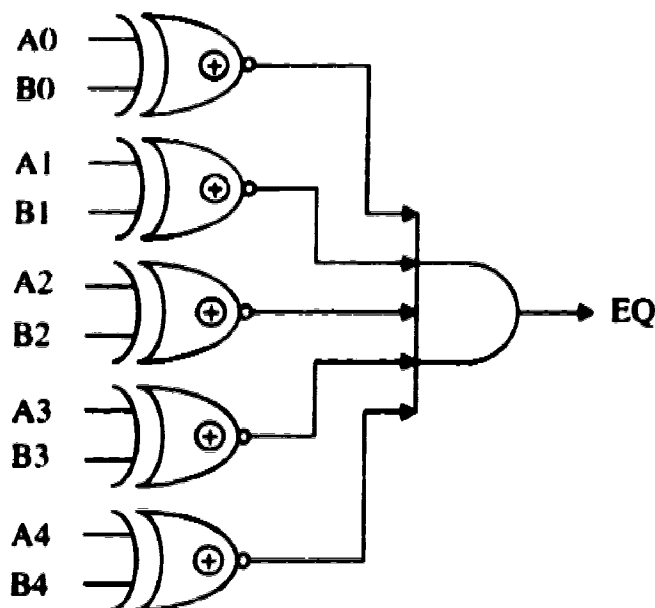


Figure A.3: Schematic Diagram for 5-Bit Comparator

The module has the following two inputs:

- *A0, A1, A2, A3, A4*: 5-bit binary number, *A4* is the MSB
- *B0, B1, B2, B3, B4*: 5-bit binary number, *B4* is the MSB

The module has one output, *EQ*, which is asserted only if the two binary numbers are exactly equal, i.e. ($A0=B0$), ($A1=B1$), ($A2=B2$), ($A3=B3$), and ($A4=B4$).

A.4 D-Flip Flop with Synchronous Load and Synchronous Clear

In this section, the design of the D-flip flop with synchronous load and synchronous clear

used in the Meggitt register in the ECU is presented. The schematic diagram is shown in Figure A.4:

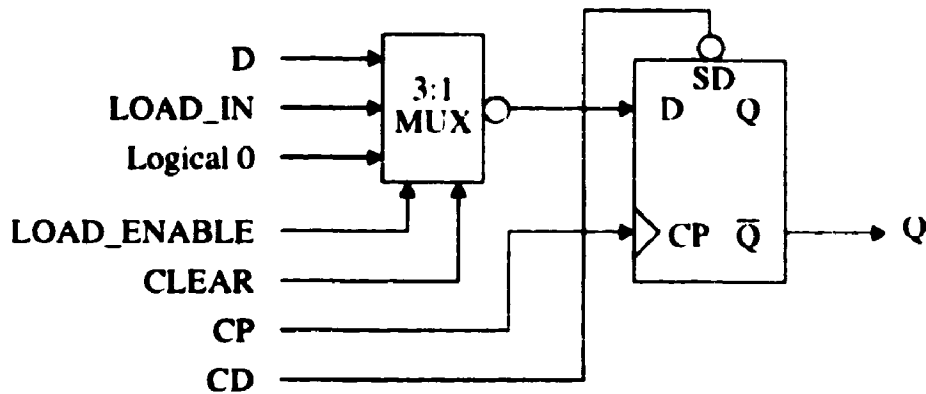


Figure A.4: Schematic diagram of D-Flip Flop with synchronous load and clear

The module has the following inputs:

- **D**: Conventional D-flip flop data input
- **LOAD_IN**: Synchronous load input data
- **LOAD_ENABLE**: enable synchronous load of data on the **LOAD_IN** input into the flip flop
- **CLEAR**: Enable synchronous clear of the flip flop
- **CP**: System clock
- **CD**: Asynchronous reset

The flip flop has one output, **Q**. The values which the **Q**-output of the flip flop will take on after the next rising clock edge for all possible control signal input conditions is given in Table A.2:

Table A.2: Next state table for D-Flip Flop with synchronous load and clear

LOAD ENABLE	CLEAR	Q^+
0	0	D
1	0	LOAD_IN
X	1	0

A.5 Error Pattern Detection Circuit

The error pattern detection module is used by the Meggitt decoder to locate a single bit

error. The error is found when the contents of the Meggitt register are \$C47 in DS1 mode or \$504 in DS3 mode.

In DS1 mode, all 12 bits of the register are examined, and in DS1 mode, only the right-most 11 bits carry valid syndrome information. Thus in DS1 mode, the LSB of the Meggitt register is on the Q-output of f0 in Figure 4.10 and in DS3 mode, the LSB in the register is on the Q-output of f1.

The error detection circuit hence implements the following equation:

ERROR LOCATED =

$$\begin{aligned} & (DS1)(Q_0)(Q_1)(Q_2)(\bar{Q}_3)(\bar{Q}_4)(\bar{Q}_5)(Q_6)(\bar{Q}_7)(\bar{Q}_8)(\bar{Q}_9)(Q_{10})(Q_{11}) \\ & + (DS3)(\bar{Q}_1)(\bar{Q}_2)(Q_3)(\bar{Q}_4)(\bar{Q}_5)(\bar{Q}_6)(\bar{Q}_7)(\bar{Q}_8)(Q_9)(\bar{Q}_{10})(Q_{11}) \end{aligned} \quad (A.1)$$

A.6 Frame Boundary Divided By 2

This module processes the active low pulse on the *FRAME IN* line of the FEC encoder to produce a 50% duty cycle square wave with a period of two DS3 frames or masterframes depending on whether the frame or masterframe information is supplied respectively. The output of the module is a logic zero for all time if the system is in DS1 mode. The schematic for the circuit to achieve this is shown in Figure A.5:

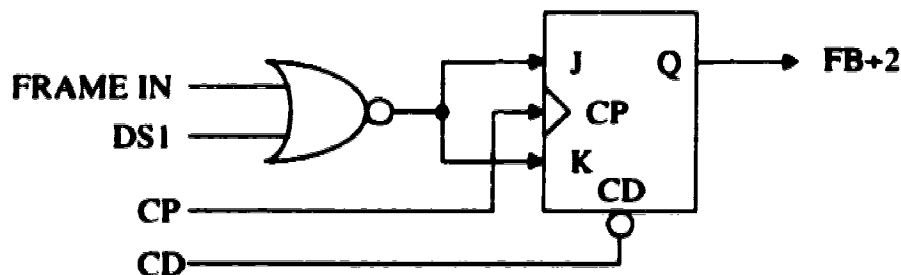


Figure A.5: Schematic diagram for the frame boundary divided by 2 module

The four inputs to this module are the following:

- **FRAME IN:** The signal on the chip level *FRAME IN* line, either the DS3 frame or masterframe signal
- **DS1:** DS1, DS3 mode select line
- **CP:** System clock
- **CD:** Asynchronous clear

The module has only one output, *FB+2*. In DS1 mode, the output of the NOR gate is held

to a zero. This causes the JK-flip flop to hold its present value for all time. This value will be the logical zero placed on the Q-output of the flip flop at reset time. In DS3 mode, the NOR gate acts like an inverter. For those clock cycles while the *FRAME IN* line is high, the JK inputs on the flip flop are each a 0 so the flip flop holds its value. Each time the *FRAME IN* line goes low, the JK inputs are brought to a logical 1 which causes the flip flop to toggle its value on the next rising clock edge. The resulting output waveform is shown in Figure 3.4.

A.7 Sample FEC ENABLE

The schematic for the "Sample FEC ENABLE" module is shown in Figure A.6:

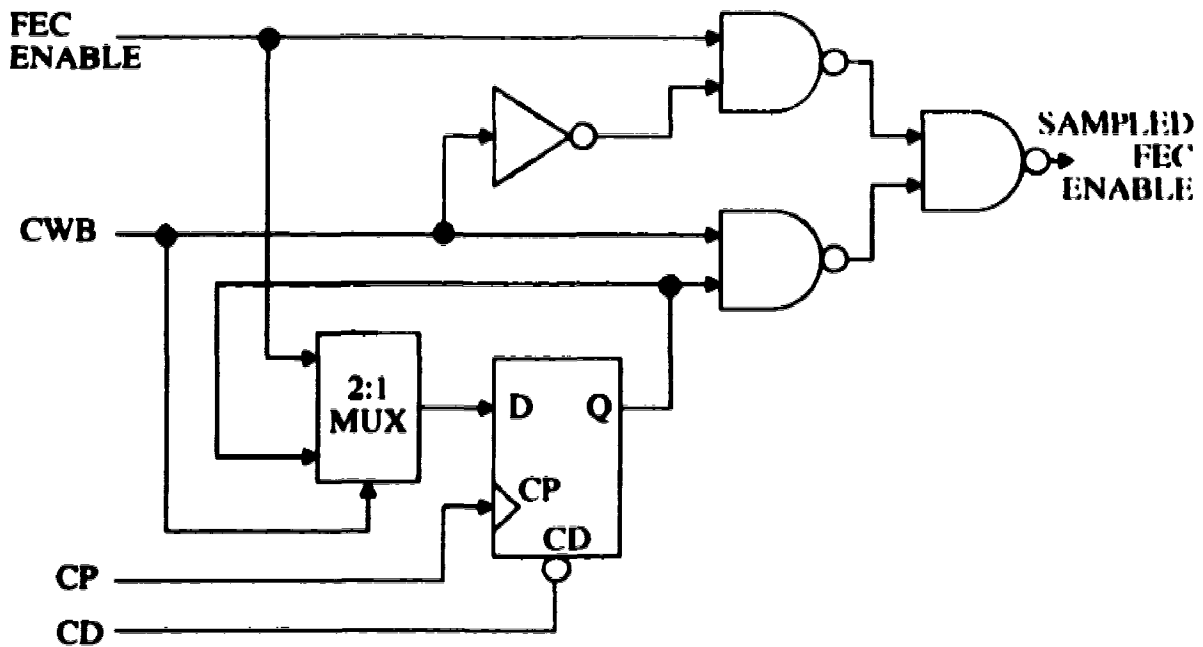


Figure A.6: Schematic for "Sample FEC ENABLE" module

The inputs to this module are the following:

- **CWB:** Codeword boundary, active low pulse, one bit time width, coincident with the first bit of a DS1 Superframe, DS3 Frame, or DS3 Masterframe.
- **FEC ENABLE:** Enables replacement of F and C-bits with FEC checkbits when asserted.
- **CP:** System Clock
- **CD:** Asynchronous reset

The output, *SAMPLED FEC ENABLE* is a signal which changes only coincident with the

codeword boundaries and holds the value of the *FEC ENABLE* line at the last codeword boundary time.

When *CWB* goes low (the codeword boundary time), the value currently on the *FEC ENABLE* line is sampled and stored on the Q-output of the flip flop in the next rising clock edge. However, when the *CWB* signal is high (not the codeword boundary time) the flip flop holds its current value. Thus the Q-output of the D-flip flop changes one bit time after the codeword boundary. The three NAND gates, *nd1*, *nd2*, *nd3* generate the *SAMPLED FEC ENABLE* signal in the following way:

$$\text{SAMPLED FEC ENABLE} = (\overline{\text{CWB}}) (\text{FEC ENABLE}) + (\text{CWB}) (Q) \quad (\text{A.2})$$

Thus at the codeword boundary time (*CWB* = 0), *SAMPLE FEC ENABLE* takes on the the current value on the *FEC ENABLE* line while *FEC ENABLE* is simultaneously sampled flip flop. At subsequent bit times, *SAMPLED FEC ENABLE* takes on the stored on the Q-output of the flip flop, i.e. the last sampled value of *FEC ENABLE*. Thus *SAMPLED FEC ENABLE* only changes at the codeword boundary time, perfectly synchronous to the emerging codewords from the delay buffer.

A.8 Correction Pulse Timing Latch

The Correction Pulse Timing Latch is used to generate the SEC signal after the lag between the time that the Meggitt decoding scheme locates the bit in error to the time the bit in error actually emerges from the codeword length delay register. The lag is managed with the ECU lag counter. The role of the latch is to generate the SEC pulse after the lag

has been resolved. Pseudocode for this FSM is given in Figure A.7

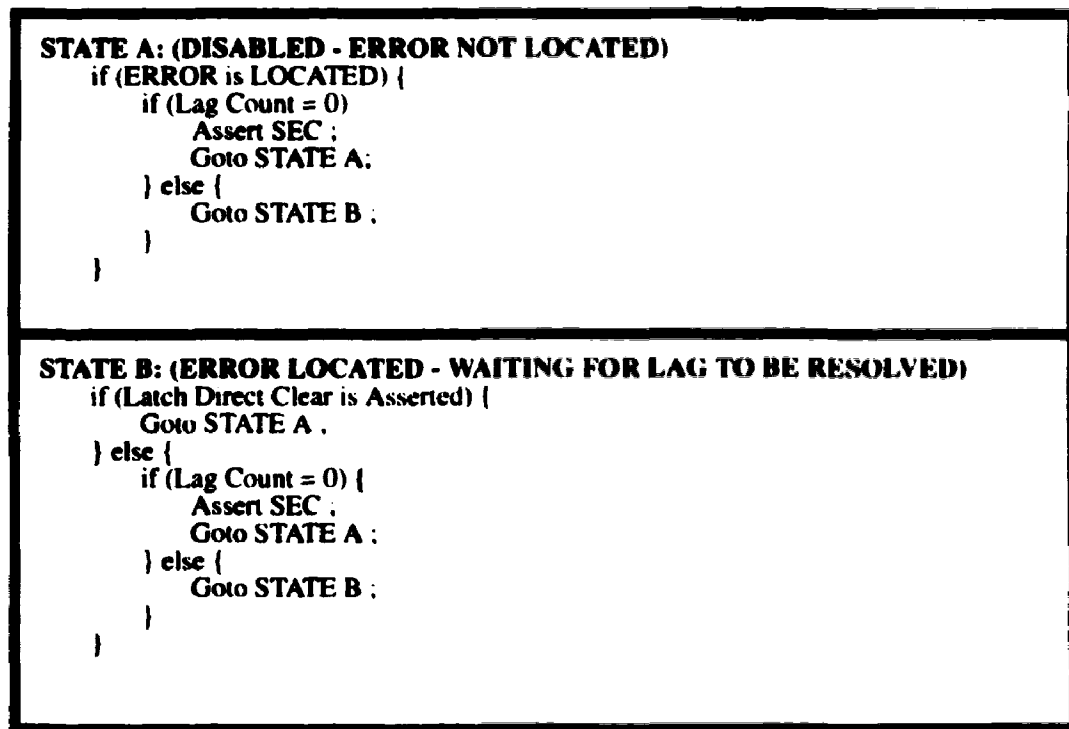


Figure A.7: Pseudocode for the Correction Pulse Timing Latch FSM

Once the error has been located, the latch is armed and the lag counter is tested for zero. If it is zero, then the correction pulse is sent immediately. Otherwise, the latch waits for the lag to be resolved before firing the correction pulse. Note that the latch has a direct synchronous clear of its own. If the lag is not resolved before the end of a codeword is reached, i.e. a higher order error, or an error located in the parity check bit positions, the latch is cleared to prevent any error correction activity.

Appendix B

**Mustang and Espresso
Logic Minimization Software
Input and Output Files**

B.1 FEC Encoder System Controller

This section contains the espresso input and output files for minimization of the state decoding logic for the FEC encoder system controller. The input and output variables are documented in the file headers along with any additional documentation.

```

#-----
# DS1/DS3 FEC ENCODER SYSTEM CONTROLLER - ESPRESSO INPUT FILE (ON-SET)
#
# INPUTS:
#   ds3                : Asserted HIGH for DS3 mode of operation
#   s0, s1, s2, ... , s11 : System Controller State Counter Value.
#                           s0 is LSB, s11 is MSB
#
# OUTPUTS:
#   cwb*               : Asserted Low for one clock cycle coincident with
#                           the first bit of a DS1-FEC or DS3-FEC codeword
#   selParity           : Asserted HIGH at each time a parity check bit is to
#                           removed from the holding register and inserted into
#                           the outgoing bit stream
#   encRegShift*        : Asserted HIGH to enable shift of new, incoming
#                           bit into encoding register; asserted LOW to gap
#                           out F-bits (Holds the register contents)
#   holdRegLoad         : Asserted to synchronously transfer the parity
#                           checkbits and the overall parity bit from the
#                           encoding and parity bit computation registers
#                           respectively into the holding register.
#
# * The minimization will derive the complement of this signal.
#   Be sure to complement it in the actual implementation.
#
# Note that the following listing is only the ON-SET for the espresso
# minimization (i.e. those conditions for which the signals assume
# a logical 1. The don't care set (those conditions for which the
# functions are unspecified) is very large. However, it logically
# contains all of the following entries:
#   For (1360 <= s <= 2315) and DS3=1 all outputs are ----
#   For (2316 <= s <= 4095) all outputs ----
#
# All other conditions not specified are the OFF-set.
#-----
.i 13
.o 4
.ilb ds3 s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11
.ob cwb selParity encRegShift holdRegLoad
# DS3 Mode Control Signals
1110010100000 0110
1101111110000 0110
1010010101000 0110
1111001011000 0110
1001111111000 0110
1100010100100 0110
1110111110100 0110
1101001011100 0110
1010111111100 0110
1111100100010 0110
1001001010010 0110
1100111110010 0110
1101100101010 0001
1011100101010 1000
# DS1 Mode Control Signals
0111111010000 0110
0000000011000 0110
0100000100100 0110
0010000001100 0110

```

```

0110000111100 0110
0001000010010 0110
0101000101010 0110
0011000000110 0110
0111000110110 0110
0000100011110 0110
0100100100001 0110
0100100001001 0001
0010100001001 1100

```

```

#=====
#  ENCODER SYSTEM CONTROLLER - ESPRESSO OUTPUT FILE
#=====
.i 13
.o 4
.ilb ds3 s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11
.ob cwb selParity encRegShift holdRegLoad
.p 27
-01-1----1--1 1100
-10-1----1--1 0001
10111--1-1-1- 1000
1010111111-- 0110
-01100000011- 0110
-10010010--1 0110
100100101--1- 0110
-00010001111- 0110
110011111--1- 0110
11011--1-1-1- 0001
0111111010000 0110
10011111110-- 0110
11101111101-- 0110
11010010111-- 0110
110111111000- 0110
001000000110- 0110
000100001001- 0110
10100101010-- 0110
11000101001-- 0110
-11100011011- 0110
11110010110-- 0110
1111100100-1- 0110
010100010101- 0110
111001010000- 0110
000000001100- 0110
010000010010- 0110
011000011110- 0110
.

```

B.2 Main FEC Decoder System Controller

This section contains the espresso input and output files for minimization of the state decoding logic for the main FEC decoder system controller. The input and output variables are documented in the file headers along with any additional documentation.

```

=====
# DS1/DS3 FEC DECODER - SYSTEM CONTROLLER STATE DECODER
#
# ESPRESSO INPUT FILE
#
# Controller Inputs:
#   ds3           : Asserted HIGH for DS3 mode operation
#   s0, s1, s2, ... , s10 : System Controller State Counter Value.
#                       s10 is MSB; s0 is LSB.
#
# Controller Outputs:
#   ms0, ms1 : Mux Control signals for Frame reconstruction
#   catch_c  : Asserted HIGH to sample current bit as the valid
#               C-bit for a DS3 frame
#   frame    : Asserted LOW for one bit time coincident with the
#               first bit of the DS1 superframe or the first bit of
#               each DS3 frame.
#   memberCB : Asserted HIGH when the bit following the current
#               one is an FEC checkbit
#   eq1358   : Asserted HIGH when the system controller state
#               counter reaches 1358
#
# Note that the listing which follows is only the ON-set for the
# espresso minimizations. The don't care set is too large to
# present here. However, the don't care set contains the following
# entries:
#   For (1360 <= s <= 2315) and (DS3-1) all outputs are -----
#   For (2316 <= s <= 4095) all outputs are -----
#=====
.i 13
.o 6
.ilb ds3 s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11
.ob ms0 ms1 catch_c frame memberCB eq1358
.type fd
0000000000000 010100
0111111010000 100010
0000000011000 100010
0100000100100 100010
0010000001100 010010
0110000111100 010010
0001000010010 100010
0101000101010 010010
0011000000110 010010
0111000110110 010010
0000100011110 100010
0100100100001 100010
1000000000000 000100
1110010100000 010010
1100101010000 001000
1101111110000 100010
1010010101000 110010
1111001011000 100010
1001111111000 110010
1100010100100 010010
1000101010100 000100
1110111110100 010010
1100010101100 001000
1101001011100 100010
1010111111100 110010
1111100100010 100010

```

1001001010010 110010
1100111110010 010010
-101100101010 000001

```

#-----
# DS1/DS3 FEC DECODER - SYSTEM CONTROLLER STATE DECODER
#
# ESPRESSO OUTPUT FILE
#
# Controller Inputs:
#   ds3           : Asserted HIGH for DS3 mode operation
#   s0, s1, s2, ... , s10 : System Controller State Counter Value.
#                       s10 is MSB; s0 is LSB.
#
# Controller Outputs:
#   ms0, ms1 : Mux Control signals for Frame reconstruction
#   catch_c  : Asserted HIGH to sample current bit as the valid
#               C-bit for a DS3 frame
#   frame     : Asserted LOW for one bit time coincident with the
#               first bit of the DS1 superframe or the first bit of
#               each DS3 frame.
#   memberCB  : Asserted HIGH when the bit following the current
#               one is an FEC checkbit
#   eq1358    : Asserted HIGH when the system controller state
#               counter reaches 1358
#
# Note that the listing which follows is only the ON-set for the
# espresso minimizations. The don't care set is too large to
# present here. However, the don't care set contains the following
# entries:
#   For (1360 <= s <= 2315) and (DS3-1) all outputs are -----
#   For (2316 <= s <= 4095) all outputs are -----
#-----
.i 13
.o 6
.ilb ds3 s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11
.ob ms0 ms1 catch_c frame memberCB eq1358
.p 29
0111111010000 100010
0000000000000 010000
-0000000000000 000100
110010101000- 001000
-10110010101- 000001
10001010101-- 000100
11000101011-- 001000
001000000110- 010010
110111111000- 100010
000100001001- 100010
111001010000- 010010
010100010101- 010010
000000001100- 100010
011000011110- 010010
010000010010- 100010
-01100000011- 010010
11101111101-- 010010
1111100100-1- 100010
11110010110-- 100010
-00010001111- 100010
10101111111-- 110010
-11100011011- 010010
10011111110-- 110010
11010010111-- 100010
11000101001-- 010010
10100101010-- 110010

```



```
110011111--1- 010010
100100101--1- 110010
-10010010---1 100010
.e
```

B.3 Frame Alignment Unit System Controller

This section contains the espresso input and output files for minimization of the state decoding logic for the FEC decoder Frame Alignment Unit (FAU) system controller. The input and output variables are documented in the file headers along with any additional documentation.

```

#=====
# REFRAME & OUT-OF-FRAME DETECTION CIRCUIT SYSTEM CONTROLLER
# ESPRESSO Input File
#
# INPUTS:
# q0, q1      : State Variables
# ZSbar       : Asserted LOW when SYNDROME=0
# TCbar       : Asserted LOW when system counter reaches
#               its Count (DS1: 2315 ; DS3: 1359)
# countEQCR   : Asserted HIGH when RF/OFD Counter = CR
# countEQCO   : Asserted HIGH when RF/OFD Counter = CO
# enable      : Asserted HIGH to enable OOF Detection
#
# OUTPUTS:
# inframe     : Assert HIGH when frame aligned
# slip*      : Pulse LOW once for each bit rejected as a frame
#               Candidate
# count_enable : Assert HIGH to increment counter on next rising
#               clock edge
# count_clear* : Assert LOW to clear count on next clock edge
# j0, k0, j1, k1 : Next State Control Logic for JK FF's
#
# * The minimization will derive the complement of this
#   signal. Be sure to complement it in the actual implementation.
#=====
.i 7
.o 8
.ilb q0 q1 ZSbar TCbar syscount_eq_0 countEQCR countEQCO enable
.ob inframe slip count_enable count_clear j0 k0 j1 k1
.type fd
000---- 01000-0-
001-1-- 10001-0-
001-0-- 00101-1-
11-0--- 0000-0-0
1101--- 01-1-1-1
11110-- 0010-0-0
11111-- 10-1-0-1
10----0 10-1-00-
10-0--1 1000-00-
1011--1 10-1-00-
1001-01 1010-00-
1001-11 01-1-10-
01----- -----

#=====
# ESPRESSO Output File
#=====
.ilb q0 q1 ZSbar TCbar syscount_eq_0 countEQCR countEQCO enable
.ob inframe slip count_enable count_clear j0 k0 j1 k1
1-01-11 01010100
10-1-0- 10100000
0-1-0-- 00101010
10-0--- 10000000
0-1-1-- 10001000
-1111-- 10010001
1011--- 10010000
0-0---- 01000000
10----0 10010000
-1-1--- 00100000
-101--- 01010101

```

B.4 Error Correction Unit System Controller

This section contains the mustang and the espresso input and output files for minimization of the state decoding logic for the FEC decoder Error Correction Unit (ECU) system controller. The input and output variables are documented in the file headers along with any additional documentation. Note that the mustang output file is the espresso input file.

```

#-----
# ERROR CORRECTION UNIT - SYSTEM CONTROLLER STATE DECODER
# MUSTANG INPUT FILE
#
# Allows mustang to make an optimal state assignment for this
# controller based on a minimization of physical area to
# implement the controller.
#
# File Format: Input Signals
#               Current State
#               Next State
#               Output Signals
#
# State Definitions:
#   State A: Error Correction Unit Disabled
#   State B: Search For Error; increment lag counter for
#             each checkbit passed over
#   State C: Error Located; decrement lag counter for
#             each bit passed over until it is zero
#
# Controller Inputs (From left to right):
#   ds3       : Signal Format mode select
#   tc        : Asserted HIGH when system counter reaches
#               its terminal count; Asserted LOW otherwise.
#   parityEQ0 : Asserted HIGH when the value of the overall
#               parity bit is a zero.
#   syndEQ0   : Asserted HIGH when the contents of the
#               syndrome register is all zeros.
#   eq1358    : Asserted HIGH when the value of the system
#               counter reaches 1358. (Used for DS3 only).
#   err_located : Asserted HIGH during the bit time when the
#               single bit error is located.
#   memberCB   : Asserted HIGH when the next bit (after the
#               current one) is a checkbit position.
#   lagEQ0     : Asserted HIGH when the value of the lag
#               counter is equal to zero.
#   inframe    : Asserted HIGH when the decoder is in-frame
#
# Controller Outputs (from left to right):
#   lagCLR     : Asserted LOW to synchronously clear the lag counter.
#   lagENAB    : Asserted HIGH to enable increment or decrement
#               of lag counter.
#   lagUP      : Asserted HIGH to enable up count of lag counter.
#               Asserted LOW to enable down count of lag counter.
#   meggittLOAD : Asserted HIGH to load the meggitt register.
#   meggittCLR  : Asserted HIGH to clear the meggitt register.
#   sed        : Pulse high during the bit time when the single error
#               is located
#   ded        : Pulse high for one bit time if a double error is
#               detected
#   hoe        : Pulse HIGH for one bit time if a higher order error
#               is detected
#   OPBerr     : Pulse HIGH for one bit time of the overall parity
#               bit is found to be in error
#   latchCLR   : Assert HIGH to clear the error correction signal
#               latch
#-----
.i 9
.o 10

```

```

.s 2
#####
# DS3: State A Data
#####
1-----0 A A 00-0000000
10-----1 A A 00-0000000
1111----1 A A 00-0000000
1110----1 A A 00-0001000
1101----1 A A 00-0000010
1100----1 A B 1--1000001
#
#####
# DS3: State B Data
#####
1-----0 B A 00--100000
1---10--1 B A 00--100100
1---11--1 B A 00--110000
1---000-1 B B 00-0000000
1---001-1 B B 0110000000
1---01-11 B A 00--110000
1---01001 B C 010-110000
1---01101 B C 00--110000
#
#####
# DS3: State C Data
#####
1-----1- C A 00-0000000
1---1--0- C A 0100000000
1---0-00- C C 0100000000
1---0-10- C C 00-0000000
#
#####
# DS1: State A Data
#####
0-----0 A A 00-0000000
00-----1 A A 00-0000000
01-1----1 A A 00-0000000
01-0----1 A B 1--1000001
#
#####
# DS1: State B Data
#####
0-----0 B A 00--100000
01-1-0--1 B A 00--100100
01-1-1--1 B A 00--110000
01-0-0--1 B B 1--1000101
01-0-1--1 B B 1--1010001
00---00-1 B B 00-0000000
00---01-1 B B 0110000000
00---10-1 B C 010-110000
00---11-1 B C 00--110000
#
#####
# DS1: State C Data
#####
00----00- C C 0100000000
00----10- C C 00-0000000
01-1---0- C A 00-0000000
01-0---0- C B 1--1000001

```

00-----1- C A 00-0000000
01-1---1- C A 00-0000000
01-0---1- C B 1--1000001

```

=====
# ERROR CORRECTION UNIT - SYSTEM CONTROLLER STATE DECODER
#   MUSTANG OUTPUT FILE / ESPRESSO INPUT FILE
#
# MUSTANG STATE ASSIGNED FINITE AUTOMATON
#   State A: 01
#   State B: 00
#   State C: 10
#
# Controller Inputs:
#   Refer to the documentation of the MUSTANG INPUT FILE
#   q0, q1 : Finite state machine state variables
#
# Controller Outputs:
#   Refer to the documentation of the MUSTANG INPUT FILE
#   q0p, q1p : Finite state machine state variables (Next State)
=====
.i 11
.o 12
#
.ilb ds3 tc parityEQ0 syndEQ0 eq1358 err_located memberCHB lagEQ0
    inframe q0 q1
#
.ob q0p q1p lagCLR lagENAB lagUP meggittLOAD meggittCLR sed ded hoe
    OPBerr latchCLR
#
.type fd
1-----0 01 01 00-0000000
10-----1 01 01 00-0000000
1111----1 01 01 00-0000000
1110----1 01 01 00-0001000
1101----1 01 01 00-0000010
1100----1 01 00 1--1000001
1-----0 00 01 00--100000
1---10--1 00 01 00--100100
1---11--1 00 01 00--110000
1---000-1 00 00 00-0000000
1---001-1 00 00 0110000000
1---01-11 00 01 00--110000
1---01001 00 10 010-110000
1---01101 00 10 00--110000
1-----1- 10 01 00-0000000
1---1--0- 10 01 0100000000
1---0-00- 10 10 0100000000
1---0-10- 10 10 00-0000000
0-----0 01 01 00-0000000
00-----1 01 01 00-0000000
01-1----1 01 01 00-0000000
01-0----1 01 00 1--1000001
0-----0 00 01 00--100000
01-1-0--1 00 01 00--100100
01-1-1--1 00 01 00--110000
01-0-0--1 00 00 1--1000101
01-0-1--1 00 00 1--1010001
00---00-1 00 00 00-0000000
00---01-1 00 00 0110000000
00---10-1 00 10 010-110000
00---11-1 00 10 00--110000
00----00- 10 10 0100000000

```



```
00----10- 10 10 00-0000000
01-1---0- 10 01 00-0000000
01-0---0- 10 00 1--1000001
00----1- 10 01 00-0000000
01-1---1- 10 01 00 0000000
01-0---1- 10 00 1--1000001
----- 11 -- -----
```

```

#=====
# ERROR CORRECTION UNIT - SYSTEM CONTROLLER STATE DECODER
# ESPRESSO OUTPUT FILE
#
# Logic Minimized State Machine
#
#=====
.i 11
.o 12
#
.ilb ds3 tc parityEQ0 syndEQ0 eq1358 err_located memberCHB lagEQ0
    inframe q0 ql
#
.ob q0p qlp lagCLR lagENAB lagUP meggittLOAD meggittCLR sed ded hoe
    OPBerr latchCLR
#
.p 29
1---01001-0 000100000000
00---01-100 000110000000
1---001-100 000110000000
00---10-100 000100000000
1101----1-1 000000000010
1---01-0100 100000100000
1110----1-1 010000001000
1---10--100 000000000100
01---0--100 000000000100
-100----1-1 001001000001
00---1--100 100000100000
-0----00-1- 000100000000
00-----0-1- 100000000000
01-1-----00 000000100000
1---0--0-1- 100000000000
1----1-1-00 010000100000
1-----00-1- 000100000000
01-0----1-- 001001000001
-0-----1-1- 010000000000
01-1----- 010000000000
1---1--0-1- 010100000000
-----1--100 000000010000
1---1----00 010000100000
1-----1-1- 010000000000
-----000 010000100000
-----0-1 010000000000
---1-----1 010000000000
-0-----1 010000000000
01-0-----1- 001001000001
.

```