General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

Produced by the NASA Center for Aerospace Information (CASI)

JPL PUBLICATION 81-87

(NASA-CR-168992) ARCHITECTURE FOR VISI DESIGN OF REED-SCLGMON ENCOLEES (Jet Propulsion Lab.) 51 F HC A04/MF A01 CSCL 09B

N82-25801

1

Unclas G3/60 27981

Architecture for VLSI Design of Reed-Solomon Encoders

K. Y. Liu



June 8, 1981

NNSN

National Aeronautics and Space Administration

Jet Propulsion Laboratory California Institute of Technology Pasadena, California JPL PUBLICATION 81-87

Architecture for VLSI Design of Reed-Solomon Encoders

K. Y. Liu

3

Jurie 8, 1981

NASA

National Aeronautics and Space Administration

Jet Propulsion Laboratory California Institute of Technology Pasadena, California The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aerchautics and Space Administration.

ACKNOWLEDGMENT

This document consolidates one phase of research under the NASA End-to-End Information System (NEEDS) program and the LSI Product Assurance Technology program spongored by the Chief Engineer's Office at NASA. The author expresses his gratitude to Dr. A. J. Ferrari, J. A. Hunter, Richard B. Miller, and R. H. Nixon for their support which made this publication possible. Also, the author thanks Dr. J. J. Lee, M. Perlman, and R. F. Rice for their comments and suggestions. Finally, the author thanks J. T. Sumida and J. C. Packard for providing the detailed logic design and fabrication, respectively.

ABSTRACT

In this document, the logic structure of a universal VLSI chip called the symbol-slice Reed-Solomon (RS) encoder chip is presented. An RS encoder can be constructed by cascading and properly interconnecting a group of such VLSI chips. As a design example, it is shown that a (255,223) RS encoder requiring around 40 discrete CMOS IC's may be replaced by an RS encoder consisting of four identical interconnected VLSI RS encoder chips. Besides the size advantage, the VLSI RS encoder also has the potential advantages of requiring less power and having a higher reliability.

TABLE OF CONTENTS

I.	INTRODUCTION	• • • • • • • • • • • • • • • • • • • •	1
ΙΪ.	BASIC CONCEPTS OF FINITE FIELDS		1
III.	REED-SOLOMON ENCODING PROCEDURES		1
Ι٧.	SYMBOL-SLICE VISI RS ENCODER ARCHITECTURE	4-	1
	4.1 VLSI RS ENCODER USING THE ROW PARTITIO	NING TECHNIQUE 4-	3
	4.1.1 Generator Polynomial Coefficien	ts Table 4-	3
	4.1.2 Finite Field Multiplier	4-0	6
	4.1.3 Input and Feedback Control Swit	ches 4-0	6
	4.1.4 Input/Output Data Connections	4-8	8
	4.2 VLSI RS ENCODER USING THE COLUMN PARTI	TIONING TECHNIQUE 4-4	9
V.	PERFORMANCE OF THE VLSI RS ENCODER SYSTEM	••••••••••••••••	1
VI.	CONCLUSIONS	6-3	1
REFERI	RENCES		1

APPENDIX

U

VLSI	REED-SOLO	110N	ENCODER	FOI	1	ГНE	EI	?RC)P()SI	ED	NÆ	۱SA	./E	SA	T	EL	Elf	IEI	RY	,			
CHANN	EL CODING	ST	ANDARD						•		•							•	•			•	Λ-1	L

LIST OF FIGURES

Ť

Figure		Р	age
1	A Block Diagram of a $(2^{J}-1, 2^{J}-1-2E)$ RS Encoder Using a Conventional Generator Polynomial	•	3-2
2	A Block Diagram of a (2^{J} -1, 2^{J} -1-2E) RS Encoder Using a Generator Polynomial With Symmetrical Coefficients	•	3-5
}	Code Array Structure and Order of Symbol Transmission For Type B Interleaving, Where Interleaving Level = I	•	3-6
4	A Block Diagram of a (255,223) RS Encoder With Interleaving Level I and	•	3-8
	$g(x) = \prod_{i=1}^{143} (x - a^{i})$		
5	VLSI RS Encoder Chip Logic Structure (Row Partitioning)	•	4-4
6	VLSI RS Encoder System Diagram (Row Partitioning)	•	4-5
7	Logic Structure of a Serial-Parallel Finite Field Multiplier Which Performs $\alpha^{1} \star \alpha^{j}$ Modulo $\alpha^{8} + \alpha^{4}$ + $\alpha^{3} + \alpha^{2} + 1$	•	4-7
8	VLSI RS Encoder Chip Logic Structure (Column Partitioning)	•	4-10
9	Inter-Chip Connection Diagram (Column Partitioning)	•	4-11
10	Throughput Improvement by Multiplexing VLSI RS Encoder Chips (Row Partitioning Version)	•	5 - 2
A1	Logic structure of a serial-parallel finite field multiplier which performs $\alpha^{i} + \alpha^{j}$ modulo $\alpha^{8} + \alpha^{7} + \alpha^{2} + \alpha + 1$,	•	۸6

v1

£1. -

LIST OF TABLES

Table

Page

3 Generator Polynomial Coefficients for g (x) = $\prod_{i=112}^{1}$ (x - α^{i}). . . . 3-4

- A2 Generator Polynomial Coefficients for

$$g(x) = \prod_{i=112}^{143} (x - \alpha^{ij})$$
, where j=1,7,11,13,19,23,37 and 43 A-5

SECTION I

INTRODUCTION

Reed-Solomon (RS) codes (Ref. 1) are nonbinary BCH codes. These codes can correct both random and burst errors over a communication channel. Recently concatenated coding systems using RS codes as the outer codes have been proposed for space communication to achieve very low error probabilities (Refs, 2 to 7). Several deep space flight projects such as the Voyager at Uranus encounter, the Galileo, and the International Solar Polar Mission (ISPM) have also considered using the concatenated RS/Viterbi channel coding scheme. Eence RS codes are quite important for space communications.

The complexity of an RS encoder is proportional to the errorcorrecting capability of the code, the speed of the encoding, and the interleaving level used (Ref. 4). For reliable space communication there is a need to use RS codes with large error-co.recting capability and large interleaving level (Refs. 4, 5, 8, and 9. Hence one is especially interested in minimizing the complexity of RS encoders for space communication applications. In a spacecraft the power, size, and reliability requirements are usually quite severe. Thus there is considerable interest in a VLSI (Very Large Scale Integration) RS encoder which has the potential for significant savings in size, weight, and power while at the same time providing higher reliability over an RS encoder implemented in discrete logic circuits.

This document introduces a symbol-sliced logic structure suitable for a VLSI implementation of RS encoders. By cascading and properly interconnecting a group of such VLSI chips, each consisting of a fixed portion of the encoder, it is possible to obtain an RS encoder with any desired error-correcting capability

and interleaving level. As a design example, it is shown that a (255,223) RS encoder requiring 40 discrete CMOS IC's may be replaced by an RS encoder consistin, of four identical interconnected VLSI encoder chips. It is also shown that these VLSI RS encoder chips can be paralleled to improve the encoding speed.

SECTION II

BASIC CONCEPTS OF FINITE FIELDS

A field is a set of elements, including 0 and 1, any pair of which may be added or multiplied (denoted by + and *) to give a unique result in the field. The addition and multiplication are associative and commutative, and multiplication distributes over addition in the usual way, i.e.,

Every field element u has a unique negative element -u in the same field such that

u + (−u) □ 0

Every nonzero field element u has a unique reciprocal field element 1/u, such that

For every field element u

o + u = u = 1 * u

and

o * u = 0

If the number of elements in a field is infinite, then it is called an infinite field. Examples of infinite fields are:

- (a) The rational number field.
- (b) The real number field.
- (c) The complex number field.

If the number of elements in a field is finite, then it is called a finite field or a Galios field $GF(q_2)$, where q is the number of elements in the field. Two examples of finite fields are given as follows. The first example is the finite field GF(p) which is formed by integers modulo p, where p is a prime. If p = 2, then the field is called GF(2). GF(2) contains only two elements, i.e., 0 and 1. The addition and multiplication tables of GF(2) are given as below:



Addition

Multiplication

The second example is the finite field formed by polynomials modulo and irreducible polynomial of degree m with coefficients in GF(p), where p is a prime. The definition of an irreducible polynomial is given as follows. A polynomial

$$p(x) = \sum_{i=0}^{m} a_{i} x^{i}$$

with $a_i \in GF(p)$ is called irreducible over GF(p) if there exists no polynomials A(x) and B(x) with coefficients from GF(p) such that

$$p(x) = A(x) B(x)$$

where

$$1 \leq \text{Degree of } A(x) \leq m-1$$

As an example, if p = 2, then the polynomials

$$x^8 + x^4 + x^3 + x^2 + 1$$

and

$$x^{4} + x + 1$$

are irreducible over GF(2).

Now the multiplicative structure of finite fields will be discussed. If a field contains an element α , then the least positive integer N for which $\alpha^{N} = 1$ is called the order of α . In a finite field of q elements, GF(q), there is a primitive element α , i.e., an element of order q - 1. Every nonzero element of GF(q) can be expressed as a power of α .

Next the vector space structure of finite fields will be presented. To do it, one needs the following definition of a primitive polynomial. An irreducible polynomial of degree m over GF(q) is called primitive if it has a primitive element of $GF(q^m)$ as a root. A finite field of q^m elements, $GF(q^m)$, can be considered as an m-dimensional vector space over GF(q). A choice for a basis of $GF(q^m)$ over GF(q) is the set

$$\{1, \alpha, \alpha^2, \dots, \alpha^{m-1}\}$$

which is called the canonical basis, where α is a root of a primitive polynomial of degree m over GF(q). In vector form

$$\{1, \alpha, \alpha^2, \dots, \alpha^{m-1}\}$$

is represented as

 $1 \iff (0, 0, ---, 0, 0, 1)$

$$\alpha \iff (0, 0, ---, 0, 1, 0)$$

$$\alpha^{2} \iff (0, 0, ---, 1, 0, 0)$$

$$\|$$

$$\alpha^{m-1} \iff (1, 0, ---, 0, 0, 0)$$

ΟĮ

whereas 0 is mapped to (0, 0, ---, 0). Thus all elements in $GF(q^m)$ except 0 can be formed by linear combinations of the canonical basis

As an example, the Galois field of 2^4 elements, $GF(2^4)$, may be formed as the field of polynomials over GF(2) modules $(x^4 + x + 1)$, which is a primitive polynomial of degree 4. Let α be a root of $x^4 + x + 1$, i.e.,

$$\alpha^4 + \alpha + 1 = 0$$

Then by the definition of the primitive polynomial, all elements of $GF(2^4)$ except 0 are powers of α . The representation of α^i for $4 \le i \le 15$ can be determined by the primitive polynomial. In this example α is a root of $x^4 + x + 1$ over GF(2). Thus

$$\alpha^4 + \alpha + 1 = 0$$

It follows that

$$\alpha^4 = -\alpha - 1$$

Since -1 = 1 in GF(2), one has

$$\alpha^4 = \alpha + 1$$

The rest of the element α^{i} for $5 \le i \le 15$ can be obtained likewise. The 15 nonzero field elements of $GF(2^{4})$ are shown below in both multiplicative and vector space forms.

¥7.

a ⁰]		(0001)
α ¹	8		α	8	(0010)
α ²	8	α ²		1	(0100)
α ³	□ α ³			۵	(1000)
α4	=		α +]	. =	(0011)
_α ⁵	-	α ² +	·a	=3	(0110)
α ⁶	= α ³	+ a ²		•	(1100)
α ⁷	= α ³	+	-α+1	. =	(1011)
a ⁸	=	α^2	+ 1	. =	(0101)
α ⁹	= α ³	+	α	8	(1010)
α ¹⁰	-	α ² +	• a + 1	. =	(0111)
α^{11}	= α ³	+ α ² +	-α		(1110)
α^{12}	= α ³	+ α ² +	- α + 1	. 8	(1111)
α ¹³	= α ³	+ α ²			(1101)
α ¹⁴	= α ³		+]	. =	(1001)
α ¹⁵	=		1	. =	$(0001) = \alpha^{\circ}$

Another example is the finite field $GF(2^8)$ generated by the primitive polynomial

 $x^{8} + x^{4} + x^{3} + x^{2} + 1$

The exponents versus elements and elements versus exponents tables for this $GF(2^8)$ are shown in Table 1 and Table 2, respectively.

	LI DEN	14																																										
ı	REPORT	255																																										
	neven	133	55	218	158	132	94	X1	16	73	114	(*) 01 r1	203	198	126	215	246	227	75	98	55	165	65	ŝ	141	28	224	83	162	121	239	64	69	18	144	544	243	203	22	176	233	17	Z16 147	747
	EXPONENT	128	131	134	12 î	071	143	146	149	152	155	158	161	164	167	170	173	176	179	182	185	188	161	194	197	200	203	206	209	212	215	218	221	224	227	230	233	236	239	242	245	248	251 254	5
	REVER	204	46	109	52	z	42	11	82	170	57	213	230	66	63	229	123	255	171	67	671	220	174	25	200	14	112	167	81	178	249	155	172	ማ	72	122	247	235	7	88	250	151	103 1	11
,	EXPONENT	127	130	133	136	139	142	145	148	151	154	157	160	163	166	169	172	175	178	181	184	187	190	193	156	199	202	205	108	111	214	217	220	223	226	229	232	235	238	241	244	247	250	53
	INEVERIE	102	23	184	169	33	21	168	41	85	146	228	115	191	145	252	179	241	219	150	196	110	87	130	100	7	56	221	166	68	242	195	86	138	36	61	245	251	139	77	125	207	2 2	C11
	EXPONENT	126	129	132	135	138	141	144	147	150	153	156	159	162	165	168	171	174	177	180	183	186	189	192	195	198	201	204	207	210	213	216	219	222	225	228	231	234	237	240	243	246	249	707
	LENEN	4	32	52	232	19	152	180	201	¢	48	157	156	148	212	238	35	s	40	63	210	222	190	153	188	137	99	253	187	177	225	16	226	67	34	13	104	103	31	248	147	236	27	
1.	EXPORENT	2	Ś	30	11	14	17	20	23	26	29	32	35	38	41	44	47	50	53	56	59	62	65	88	71	74	11	80	83	86	68	92	95	98	101	104	107	110	113	911	119	122	125	
+ x ² +	REVENT	7	16	128	116	135	76	96	121	(*1	1	192	90 1	74	106	119	159	140	20	160	105	111	95	194	54	202	30	240	211	214	254	163	113	175	17	136	52	189	129	124	199	118	151	
x ⁴ + x ³	TKENOAXE	Ч	4	7	16	13	16	19	22	25	28	31	34	37	40	43	46	67	52	55	58	61	z	67	70	73	76	79	82	85	88	16	76	97	100	103	106	109	112	115	118	121	124	
× ⁸ +	TIENENT	l	60	79	58	205	38	45	117	143	ព	96	39	37	53	181	193	70	10	80	186	185	161	97	47	101	15	120	231	107	127	223	182	217	134	6 8	26	208	206	62	237	59	197	
	EXPONENT	0	m	ę	6	12	15	18	21	24	27	30	33	36	39	42	45	48	51	2	57	60	63	66	69	72	75	78	81	84	87	06	93	96	66	102	105	108	111	114	117	120	123	

Exponents versus elements in a finite field GF(2⁸) generated by the primitive polynomial Table 1.

nial	EXPONENT	112	140	13	222	197	227	119	180	68	35	46	209	188	744	178	190	86	20	158	57	109	31	216	164	23	127	246	59	157	251	1//	62	88	156	81	22	117	79	233	173	214	168	175
e polyno	ELEMENT	129	132	135	138	141	144	147	150	153	156	159	162	165	168	171	174	177	180	183	186	189	192	195	198	201	204	207	210	213	216	219	222	225	228	231	234	237	240	243	246	249	252	255
primitiv	EXPONENT	7	247	66	74	67	74	153	184	17	217	137	63	149	205	151	252	242	171	93	60	11	162	67	123	196	236	111	161	41	170	134	204	203	176	160	245	122	215	213	231	116	234	88
ed by the	ELEMENT	128	131	134	137	140	143	146	149	152	155	158	101	164	167	170	173	176	179	182	185	188	191	194	197	200	203	206	209	212	215	218	221	224	227	230	233	9c7	239	242	245	248	251	254
generate	EXPONENT	87	192	128	103	237	254	165	38	124	146	32	55	16	207	135	220	97	211	42	132	83	65	45	183	118	73	12	108	82	85	96	187	90	95	169	H	235	44	174	230	232	244	80
1d GF(2 ⁸)	ELEMENT	127	130	133	136	139	142	145	148	151	154	157	160	163	166	169	172	175	178	181	184	187	190	193	196	199	202	205	208	211	214	217	220	223	226	229	232	235	238	241	244	247	250	253
inite fie	EXPONENT	25	26	223	27	75	224	141	28	248	76	138	225	33	142	18	29	125	249	154	77	166	139	221	226	179	34	208	143	189	19	56	30	163	126	58	250	61	155	21	78	172	167	
s in a fi	ELEMENT	m	9	6	12	15	18	21	24	27	30	33	36	39	42	45	48	51	54	57	60	63	66	69	72	75	78	81	84	87	06	93	96	66	102	105	108	111	114	117	120	123	126	
exponent x ² + l.	EXPONENT	-1	50	e	238	199	100	52	129	105	80	Ś	47	15	147	240	69	194	39	201	120	114	191	102	253	37	145	54	206	219	210	131	64	182	72	107	84	186	94	10	43	229	243	
<pre>cs versus t + x³ + 3</pre>	ELEMENT	2	Ś	8	11	14	17	04	23	26	29	32	35	38	41	77	47	50	53	56	59	62	65	68	71	74	17	80	83	86	89	92	95	98	101	104	107	110	113	116	119	122	125	
Element x ⁸ + x ⁴	EXPONENT	0	2	198	51	104	4	14	239	193	200	113	101	36	53	218	130	181	106	185	6	228	9	86	48	152	16	136	148	150	241	92	70	66	195	110	40	133	202	159	121	212	115	
Table 2.	ELEMENT	1	4	7	10	13	16	19	22	25	28	31	34	37	40	43	46	49	52	55	58	61	64	57	70	73	76	79	82	85	88	16	94	97	100	103	106	109	112	115	118	121	124	

The addition of two field elements in $GF(2^n)$ is performed by componentwise addition in the vector representations of the two elements. For example in $GF(2^8)$, $\alpha^{26} + \alpha^{238}$ is given by 1

L.

$$6 = (0,0,0,0,0,1,1,0) \leftrightarrow \alpha^{26}$$

$$+ 11 = (0,0,0,0,1,0,1,1) \leftrightarrow \alpha^{238}$$

$$13 = (0,0,0,0,1,1,0,1) \leftrightarrow \alpha^{104}$$

The multiplication of two field elements in $GF(2^n)$ is performed by a modulo (2^n-1) addition on the exponents of the two elements. For example in $GF(2^8)$, $a^{26} * a^{238}$ is given by

$$6 = (0,0,0,0,0,1,1,0) \leftrightarrow \alpha^{26}$$
*) 11 = (0,0,0,0,1,0,1,1) $\leftrightarrow \alpha^{238}$

$$58 = (0,0,1,1,1,0,1,0) \leftrightarrow \alpha^{(26} + 238) \text{ MOD } 255$$

$$= \alpha^{9}$$

SECTION III

REED-SOLOMON ENCODING PROCEDURES

An RS code word has $(2^{J}-1)$ symbols, where each symbol has J bits. Of the $(2^{J}-1)$ symbols there are $(2^{J}-1-2E)$ information symbols and 2E parity-check symbols, where E is the number of symbols an RS code is able to correct. If one treats the $(2^{J}-1-2E)$ information symbols as the coefficients of the polynomial

$$f(x) = x^{2E} \left(s_{2^{J}-1-2E} + s_{2^{J}-2-2E} + s_{2^{J}} x^{2^{J}-2-2E} + s_{1^{J}} x^{2^{J}-1-2E} \right)$$

where s_i is the ith transmitted symbol, then the 2E parit-check symbols can be obtained as the coefficients of the remainder of

منابعة والمراجع ورجاع والتراجع والمعالية والمعالية

where g(x) is the generator polynomial (Ref. 9) of the code. Usually g(x) is defined as

$$g(x) = \prod_{i=1}^{2E} (x-\alpha^{i}) = \sum_{j=0}^{2E} g_{j} x^{j}$$

where a is a primitive element of the Galois field $GF(2^J)$, and g_j 's are the coefficients of g(x) with $g_{2E} = 1$. The generator polynomial defined above does not have symmetrical coefficients, i.e.,

$$g_{i} \neq g_{2E-i}$$
 for $j = 0, 1, 2, ---, 2E$.

A block diagram of an RS encoder which generates the remainder of f(x)/g(x) is given in Fig. 1. The switches in Fig. 1 are normally in the "ON" position until the last information symbol gets into the encoder. At this



ORIGINAL PAGE IS OF POOR QUALITY moment all switches are switched to the "OFF" position and the encoder is behaving like a long shift register. The output of the encoder is then taken from the output of the last shift register. Note that in Fig. 1, 2E multipliers are needed in the encoder.

and the second second

To reduce the number of multipliers needed, a special class of the generator polynomial which has symmetrical coefficients was proposed by Berlekamp (Ref. 10). This generator polynomial is defined as

$$g(x) = \prod_{i=2}^{2^{J-1}+E-1} (x-\alpha^{i}) = \sum_{j=0}^{2E} g_{j}x^{j}$$

where

$$g_j = g_{2E-j}$$
 and $g_o = g_{2E} = 1$.

Note that since $g_0 = 1$, only E multipliers are needed (see Fig. 2). Thus using this new generator polynominal will reduce the number of multipliers required by one-half. As an example, the coefficients of all generator polynomials of the form

$$g(x) = \prod_{i=112}^{143} (x - \alpha^{ij})$$

for a 16-error-correcting RS code with 8-bit per symbol are shown in Table 3 for $\alpha = 2$, 128, 232, 135, 201, 90, 74, 119, respectively.

There are several schemes for interleaving the RS codes (Ref. 5). One scheme illustrated in Fig. 3 as "Interleave B" requires memory only for the paritycheck symbols in the encoder is described as follows. In this scheme the input bits are grouped into J-bit symbols and transmitted in their natural order. However every Ith symbol belongs to the same code word, where I is the interleaving

3=119	1	73	192	158	246	160	94	159	171	Ś	240	129	124	188	242	173	80	173	242	188	124	129	240	ŝ	171	159	76	160	246	158	192	73	• -1
3=74	Г	ø	135	138	187	65	130	123	198	77	102	117	77	138	214	106	125	106	214	138	17	117	102	44	198	123	130	65	187	138	135	80	
α=90	1	169	247	73	177	80	53	59	129	106	103	18	47	184	64	83	168	83	64	184	47	18	103	106	129	59	53	80	177	73	247	169	-4
a=201	I	139	11	140	63	43	223	97	236	41	88	156	102	145	98	31	220	31	98	145	102	156	88	41	236	97	223	43	63	140	11	139	1
a=135	-1	174	158	204	153	89	77	16	154	158	188	234	191	72	226	59	78	59	226	72	191	2 54	188	158	154	16	44	89	153	204	158	174	1
7=232	1	213	184	106	132	36	144	194	6	59	67	81	56	158	204	74	135	74	204	158	56	81	67	59	6	194	144	36	132	106	184	213	1
u=128		227	151	109	47	104	26	184	213	٠	1.13	1:'4	238	15	157	232	6	232	157	67	238	174	133	4	213	184	26	104	47	109	151	227	-1
α=2		236	244	220	133	238	137	201	7	141	11	226	34	252	209	22	78	22	209	252	34	226	11	141	7	201	137	238	133	220	244	236	1
DEG(x)	0		7	m	4	S	ę	7	80	6	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	2ò	27	28	29	30	31	32

ORIGINAL PACE 13 OF POOR QUALITY

143 ∏ i = 112

 $(x-a^{i})$

Table 3. Generator Polynomial Coefficients for g(x) =



OFIGINAL PACE IS OF POOR QUALITY 针

بلا فيد . ب



ORIGINAL FALL ! . OF POOR CULLITY

> Code Array Structure and Order of Symbol Transmission For Type B Interleaving, Where Interleaving Level = I

> > Figure 3.

depth used. Thus I code words make up such an interleaved code block. After the information symbols are transmitted, the parity-check symbols of each interleaved code word are then transmitted.

If interleaving is used, then the encoder logic structure is the same as shown in Fig. 1, except now each J-bit shift register is replaced by an I x J-bit shift register. As an example, a block diagram of a (255,223) RS encoder with interleaving level I and generator polynomial

$$g(x) = \prod_{i=112}^{143} (x-\alpha^{i})$$

where $\alpha = 2$ in GF(2⁸), which is generated by the primitive polynomial

$$x^8 + x^4 + x^3 + x^2 + 1$$
,

is shown in Figure 4. Note that a generator polynomial with symmetrical coefficients is used here to save multipliers.

ORIGINAL PAGE IS OF POOR QUALITY



Figure 4. A Block Diagram of a (255,223) RS Encoder With Interleaving Level I and

$$g(x) = \prod_{i=112}^{143} (x - x^{i})$$

SECTION IV

SYMBOL-SLICE VLSI RS ENCODER ARCHITECTURE

A finite field multiplication is a quite complicated operation. There are basically three techniques for implementing a finite field multiplication. The first technique is to use log and antilog tables stored in read-only memories (ROM's) (Ref. 4). The second technique is to use a linear feedback shift register cype of approach (Ref. 10). The third technique is to use the property of the trace in a finite field to form a smaller ROM 1/ ok-up table (Ref. 11). Due to the advent of LSI ROM technology, techniques 1 and 3 are usually used in an RS encoder design optimized for discrete IC's. As an example a 400 KHZ (255,233) RS encoder using the Berlekamp's approach (Ref. 11) requires only around 40 CMOS IC's.

When one is interested in further drastic reduction of the power and size of an KS encoder for high speed applications, one has to consider VLSI implementations. An RS encoder design optimized for discrete IC's usually does not have a modular structure. Hence when one uses such an architecture for VLSI layout, one has the following problems:

- (1) The design is too big to be put on one chip.
- (2) If a multichip approach is used, then one needs several chip designs, where each chip has an impractical number of input/ output pins.
- (3) The design is not modular. Therefore the design is not easy to adapt to other RS code parameters.

Hence there is a need to find a VLSI logic structure which can alleviate the above problems.

The repetitive architecture of the RS encoders shown in Figs. 1, 2, and 4, suggested that a symbol-slice type of VLSI chips, each one consisting of a fixed portion of the encoder, may be cascaded to form a complete RS encoder. Also to reduce the VLSI chip size, RS encoders using generator polynomials with symmetrical coefficients are preferred. Hence we will put emphasis on this type of VLSI RS encoder.

As an example, we will design a VLSI encoder chip for a 255-symbol, 8-bit per symbol, 16-error-correcting, RS code with an interleaving level of 5. The primitive polynomial used is

$$x^{8} + x^{4} + x^{3} + x^{2} + 1$$

The generator polynomial for this RS code is

$$g(\mathbf{x}) = \prod_{i=112}^{143} (\mathbf{x} - \alpha^{i})$$

where $\alpha = 2$. The coefficients of this g(x) are given in the first column of Table 3. The encoder logic structure for the above RS code parameter is identical to the one shown in Fig. 4, except now I = 5. There are several ways of partitioning the RS encoder into four sections. One way which requires a minimum of input/ output pins is to include four rows of logic shown in Fig. 4 into one section. Each section is then realized by a universal VLSI RS encoder chip. Another way to partition the RS encoder shown in Fig. 4 into four sections is to include 8 rows of logic in each column into one section. Each section is then realized by

a universal VLSI RS encoder chip. These VLSI RS encoder configurations are described as follows.

4.1 VLS1 RS ENCODER USING THE ROW PARTITIONING TECHNIQUE

The logic structure of the universal VLS1 RS encoder chip using the row partitioning technique is shown in Fig. 5. The entire VLSI encoder system, which consists of four identical VLS1 RS encoder chips cascaded and properly interconnected together is shown in Fig. 6. Each VLS1 RS encoder chip has 24 pins. A detailed description of the VLS1 RS encoder chip and the entire VLS1 RS encoder system is describe as follows:

4.1.1 Generator Polynomial Coefficients Table

Since a generator polynomial g(x) with symmetrical coefficients is used, the coefficients of x^0 is always 1. Hence there is no need for a multiplier to operate on this coefficient (see Fig. 4). Consequently if the new generator polynomial is used, then one only needs E/N multipliers on each VLS1 chip, where E is the error correcting capability of the code and N is the total number of chips required in a VLS1 encoder system. In the design example, E = 16 and N = 4. Hence 4 multipliers are used on each VLS1 RS encoder chip. To make the VLS1 chip universal, all distinct coefficients except 1 of the generator polynomial are stored in a read-only memory on the chip. In general, an EXJ-bit table is needed. In the design example E = 16, J = 8. Hence a 16 x 8-bit table is selected. The outputs of an EXJ-bit table is fed into N, E/N-to-one multiplexors. The outputs of the multiplexors are selected by $\log_2 N$ input pins called the "chip select" or "G select" pins. These outputs are then fed into the inputs of the E/N multipliers. In the design example two "G select" pins (pins 22 and 23) and four, four-to-one multiplexors are used. The 16 x 8 table and the multiplexors can easily be

ORIGINAL PAGE IS OF POOR QUALITY



ORIGINAL PAGE IS OF POOR QUALITY

2

an 194 au 194 au 194 au

1. an

١,



implemented by four, $4 \ge 8$ ROM, with G select signals as the address control lines of each ROM. The coefficients of g(x) selected on each chip are shown in Fig. 6.

4.1.2 Finite Field Multiplier

Next we will discuss the architecture of the finite field multiplier. To connect the multiplier properly between chips and at the same time minimize the number of input/output pins used, a linear feedback shift register type of multiplier (Ref. 9) rather than a ROM table look up type of multiplier (Ref. 4) is adopted. The multiplier used is of a serial-parallel type. The logic structure of the multiplier is shown in Fig. 7. The J-bit generator polynomial coefficient is read out from the ExJ-bit ROM table and fed into the multiplier J-bit in parallel whereas the other input, generated by the feedback input (pin 2) "ANDed" with the feedback enable (pin 1), is fed into the multiplier bit-by-bit in serial.

The output of the multiplier is loaded into an 8-bit shift register in parallel at the end of every 8th bit clock (1 symbol clock time). The parallel data is serialized by this shift register. The most significant bit (MSB) output of this shift register is added with the MSB of the 40-bit shift register, which is either on the same chip or on a different chip, and the resulting data is shifted into the least significant bit (LSB) of the next 40-bit shift register. The adder is implemented by a two-input EXCLUSIVE-OR gate (there are eight EXCLUSIVE-OR gates on each chip). Each adder takes an input from a multiplier output which is either on the same chip or on a different chip, depending on the coefficients of the generator polynomial.

4.1.3 Input and Feedback Control Switches

The switch #1 shown in Fig. 4 is implemented by an AND gate on the chip with the bit-serial data input (pin 3) and the feedback enable signal as the

Logic Structure of a Serial-Parallel Finite Field Multiplier Which Performs α^1 * α^j modulo α^8 + α^4 + α^3 + α^2 + 1 Figure 7.



ORIGINAL PAGE IS OF POOR QUALITY two inputs. The feedback enable signal is provided by an external modulo 255 counter which is driven by a clock equal to the bit clock divided by 40 (see Fig. 6). This signal is true when the counter is counting from 1 to 223; otherwise it is false. The output of switch #1 is added with the MSB of the 40-bit shift register S5 output on the same chip to generate the feedback output signal (pin 5). This signal is redundant in all but the first chip (see Fig. 6).

The switch #2 shown in Fig. 4 is implemented by an AND gate on the chip with the feedback enable and feedback input signals as the two inputs. The teedback input signals on all chips (pin 2) are connected to the feedback output signal (pin 5) on the first VLSI chip. The output of switch #2 is fed into all multipliers on the chip bit-by-bit in serial.

4.1.4 Input/Output Data Connections

There are eight input/output lines on each chip. Of these eight lines, four lines (pins 8, 9, 11, 17) are input lines and the remaining are output lines (pins 6, 7, 10, 13). Pin 8 is normally connected to pin 6 on the same chip except for the last chip, where pin 8 is grounded. Pins 7 and 9 are normally connected to pins 17 and 13, respectively on the next chip except for the last chip, where pin 7 is connected to pin 11 on the same chip and pin 9 is connected to pin 4 on the first chip. Thus one has a railroad type of data connections between chips. The reason for connecting pin 4 on the first chip to pin 9 on the last chip is a consequence of an inherent 8-bit multiplier delay. To replace the multiplier on the x° position by the switch #1 output, one needs to delay this output also by 8 bits to line up the bits. For other chips besides the first chip, the 8-bit register outputs are not used.

The encoder output is taken 8 bits earlier from the MSB of the last 40-bit shift register on the first chip. This is because when the last bit of the last symbol in the information part of the code is shifted into the encoder, the contents of each multiplier are loaded into the 8-bit output shift registers waiting to be added with the MSB of the 40-bit shift registers. These 8-bit symbols in the multiplier output registers actually belong to the fifth code word in the interleaved code array. However the parity-check symbol of the first code word is already being computed and now sitting 8 bits from the MSBs of each 40-bit shift registers. Hence the 32-bit output (pin 14) of the Past 40-bit shift register on the last chip is the output of the VLSI RS encoder system. This output is taken when the external modulo 255 counter is counting from 224 to 255. Since at these times switches #1 and #2 are turned off, the entire VLSI encoder system is behaving like an lx2ExJ-bit (e.g., 5 x 32 x 8-bit in the design example) shift register. Thus the 5 x 32, 8-bit parity-check symbols are read out from the VLSI RS encoder bit-by-bit in serial and appended to the 5 x 223, 8-bit information symbols.

4.2

VLSI RS ENCODER USING THE COLUMN PARTITIONING TECHNIQUE

The logic structure of the universal VLSI RS encoder chip using the column partitioning technique is shown in Fig. 8. Note that this chip is very similar to the one shown in Fig. 5 except now one has to provide output pins to all multipliers and input pins to all adders on the chip. Hence this chip uses 6 more input/output pins than the one shown in Fig. 5. The entire VLSI RS encoder system using this logic structure is similar to the one shown in Fig. 6 except now the interchip connections between adders and multipliers are of the pyramid type shown in Fig. 9.



VLSI RS Encoder Chip Logic Structure (Column Partitioning). Figure 8.

ORIGINAL PAGE IS OF POOR QUALITY





The 40-bit shift registers in the above two versions of the VLSI RS encoder chips can be replaced by random access memories (RAM_b). In this case, a write-after-read operation should be performed during each bit time from the same location just read out from the RAM to simulate the shift register operation. Consequently, this version will not operate as fast as the shift register version. The first advantage of using the RAM approach is that interleaving level control can easily be incorporated into the RAM address control logic. Of course, input pins must be provided to select the interleaving level. Hence a more flexible VLSI RS encoder chip can be obtained using the RAM approach.

The second advantage of using the RAM approach is that a RAM cell usually occupies a smaller chip area than that of a static shift register. Hence a smaller VLSI RS encoder chip size can be obtained using the RAM approach. It is estimated that it is impossible to put the entire shift register version of the VLSI RS encoder chip design on a 235x235 mils CMOS/bulk VLSI chip using a 7 μ m standard cell approach on all logic. However, if one uses custom RAM and ROM cells design to implement the 40-bit static shift registers and the 16x8 table and 7 μ m standard cell design for the rest of the logic, then it is possible to have a one-chip design. Of course, the entire RS encoder chip design can easily be put on a smaller chip if a VLSI technology (say 3 μ m standard cell approach) is used.

SECTION V

It

PERFORMANCE OF THE VLSI RS

ENCODER SYSTEM

For design verification, both the shift register version and the RAM version of the VLSI RS encoder chip and VLSI RS encoder system are implemented using discrete CMOS IC's and are now operational. The throughputs of these two versions are 800K bits/sec for the shift register version and 200K bits/sec for the RAM version. These throughputs are expected to go much higher if the actual VLSI encoder chips are used.

One technique to improve the VLSI RS encoding speed is to multiplex the RS encoder chips. One type of multiplexing is to set the RS encoder chip interleaving level selection in the RAM approach to 1 (no interleaving) such that for a N-level of interleaving, a N-fold parallelism can be achieved. This scheme is illustrated in Fig. 10. Note that each row of the RS encoder chips in Fig. 10 are used to encode a RS code word corresponding to the one shown in each row of the code array structure in Fig. 3.

Another type of multiplexing is to use the relationship that if

$$f(x) = f_1(x) + f_2(x) + --- + f_N(x)$$

then

$$\frac{f(x)}{g(x)} = \frac{f_1(x)}{g(x)} + \frac{f_2(x)}{g(x)} + --- + \frac{f_N(x)}{g(x)}$$

can be realized by implementing $f_i(x)/g(x)$ for i = 1, 2, ---, N in parallel and then summing the results from these parallel operations. Thus if one treats each

ORIGINAL PAGE IS OF POOR QUALITY



 $f_i(x)$ as a polynomial, whose coefficients are selected from every Nth incoming symbols starting from the ith symbol, where i = 1, 2, ---, N, then one can select the encoder chip interleaving level to 1 and use the logic structure similar to the one shown in Fig. 10 to realize the $f_i(x)/g(x)$ operation. The output of each row of encoder chips in this case needs to be properly delayed and summed to generate the encoder output f(x)/g(x).

Another technique to improve the VLSI RS encoding speed is to process the J-bit incoming symbols in parallel. Parallel adders and multipliers are needed in this configuration. A throughput improvement of J times can be achieved using this approach. The disadvantages of this technique are:

- A lot of input/output pine are needed for the parallel data paths.
- (2) A larger chip size is required to implement the VLSI RS encoder.

SECTION VI

CONCLUSIONS

We have just shown the logic structure of a symbolic-slice VLSI RS encoder chip and the VLSI RS encoder system built by these chips. A design example has been given for a (255,223) VLSI RS encoder chip and VLSI RS encoder system. It has been shown that an RS encoder consisting of four identical CMOS VLSI RS encoder chips connected together may replace around 40 CMOS IC's required by an encoder design optimized for discrete IC's. Besides the size advantage, the VLSI kS encoder also has the potential advantages of requiring less power and having a higher reliability. Finally, it is shown that such chips can easily be multiplexed to improve the encoding speed. The symbol-sliced logic structure presented in the report could also be applied to design VLSI RS decoders [12]. A separate report on this subject will be provided.

REFERENCES

14

- Reed, I.S., and Solomon, G., "Polynomial Codes Over Certain Finite Fields,"
 J. Soc. Indust. Appl. Math., 8, pp. 300-304.
- [2] Forney, G.D., Concatenated Codes, The MIT Press, Cambridge, Mass., 1966.
- [3] Odenwalder, J.P., "Optimum Decoding of Convolutional Codes," Ph.D. dissertation, Syst. Sci. Dept., Univ. of Calif., Los Angeles, 1970.
- [4] Odenwalder, J.P., et al. "Hybrid Coding System Study," submitted to NASA Ames Research Center by Linkabit Co., San Diego, Calif., Final Report, Contract No. NAS-2-6722, Sept. 1972.
- [5] Rice, R.F., "Channel Coding and Data Compression System Considerations for Efficient Communication of Planetary Imaging Data," <u>Technical Memorandum</u> <u>33-695</u>, Jet Propulsion Laboratory, Pasadena, CA., June 1974.
- [6] Rice, R.F., "Potential End-to-End Imaging Information Rate Advantages of Various Alternative Communication Systems," JPL Publication 78-52, June 15, 1978.
- [7] Hauptschein, A., "Practical, High Performance Concatenated Coded Spread Spectrum Channel for JTIDS," NTC '77, pp. 35:4-1 to 4-8.
- [8] Liu, K.Y., and Lee, J., "An Experimental Study of the Concatenated Reed-Solomon/Viterbi Channel Coding System Performance and its Impact on Space Communications," JPL Pub. 81-58, Pasadena, CA, Aug. 15, 1981; also in Proc. Nat. Telecommun. Conf., 1981.
- [9] Liu, K.Y., and Woo, K.T., "The Effects of Receiver Tracking Phase Error on the Performance of the Concatenated Reed-Solomon, Viterbi Channel Coding System," in Proc. Nat. Telecommun. Conf., 1980, pp. 51.5.1-51.5.5; also
 JPL Pub. 81-62, Pasadena, CA, September 1, 1981.
- [10] Peterson, W.W., and Weldon, E.J., Jr., Error Correcting Codes, The MIT Press, 1972.

R-1

- [11] Berlekamp, E.R., "Better Reed-Solomon Encoders," presented at Calif. Inst. of Tech. EE Seminar, Pasadena, CA., Dec. 12, 1979.
- [12] Liu, K.Y., "Architectures for VLSI Design of Reed-Solomon Decoders,"
 presented at IEEE 1st Int. Workshop on VLSI in Commun., Santa Barbara,
 CA, Sept. 1-4, 1981.

APPENDIX

VLS1 REED-SOLOMON ENCODER FOR THE PROPOSED NASA/ESA TELEMETRY CHANNEL CODING STANDARD

.

.

After the completion of the VLSI RS encoder project, the following set of RS code parameters have been proposed in the NASA/ESA telemetry channel coding standard (Ref. Al):

(1) primitive polynomial

$$x^8 + x^7 + x^2 + x + 1$$
.

(2) generator polynomial

$$g(x) = \prod_{i=1}^{143} \left[x - (x^{11})^{i} \right]$$

where α satisfies the equation

$$x^8 + x^7 + x^2 + x + 1 = 0.$$

- (3) Number of bits per symbol (J) = 8.
- (4) Number of symbols per code word (N) = 255.
- (5) Number of correctable symbol errors (E) = 16.
- (6) Interleaving depth (I) = 5.

Note that the foregoing R3 code parameters are those used in the design example in Section IV with two exceptions. These are the primitive polynomial (i.e., the field generator polynomial over GF(2) and the code generator polynomial. These polynomials were selected by Berlekaup in an architecture which minimizes the discrete ICs. Reference A2 provides a detailed description of such an architecture. To adapt to this new set of PS code parameters, one

A-2

needs to use a new field element table as well as a new table of generator polynomial coefficients. These are given in Tables Al and A2 respectively.

Thus, to design a VLSI RS encoder for the new code parameters, one only needs to replace the generator polynomial coefficients ROM table shown in Figure 5 by the coefficients given in Table A2 under the heading α^{11} . Also one needs to replace the finite field multiplier shown in Figure 7 by the multiplier shown in Figure A1.

polynomial	
le primitive	
r th	
ې ۲	
generated	
2 ⁸)	
GF (
field	
finite	
t)	
in	
exponents	
versus	ç
ents	7
Elem	α
Table Al.	

.

,

-

NENT ELEMENT EXPONENT	2 5 198	99 11 188	4 17 141	89 23 225	08 29 32	42 35 232	01 41 217	26 47 180	28 53 177	33 59 18	6 65 155	33 71 112	80 77 117	16 83 138	91 89 152		81 95 21	81 95 21 23 101 76	81 95 21 23 101 76 78 107 63	81 95 21 23 101 76 78 107 63 10 113 24	81 95 21 23 101 76 78 107 63 10 113 24 19 119 247	81 95 21 23 101 76 78 107 63 10 113 24 19 119 247 45 125 164	81 95 21 23 101 76 78 107 63 10 113 24 19 119 247 45 125 164 56 131 157	81 95 21 23 101 76 78 107 63 10 113 24 19 119 247 45 125 164 46 131 157 44 137 9	81 95 21 23 101 76 78 107 63 10 113 24 19 119 247 45 125 164 46 131 157 14 137 9 13 143 90	81 95 21 23 101 76 78 107 63 10 113 24 19 119 247 45 113 24 14 137 90 13 143 90 13 149 10	81 95 21 23 101 76 78 107 63 10 113 24 19 119 24 56 131 157 56 131 157 44 137 9 13 143 90 14 157 14 15 157 114	81 95 21 23 101 76 78 107 63 19 113 24 19 119 24 56 131 157 56 131 157 51 137 9 74 143 90 78 155 114 13 143 90 14 155 114 13 161 124	81 95 21 23 101 76 78 107 63 19 113 24 19 113 24 56 131 157 56 131 157 44 137 9 45 137 9 13 143 9 13 143 9 14 157 16 18 155 114 18 155 114 18 155 114 161 124 10 167 210 210	81 95 21 23 101 76 78 107 63 19 113 24 19 113 24 56 131 157 56 131 157 56 131 157 56 131 157 56 137 9 51 143 90 74 149 10 73 161 124 39 167 210 39 167 210 173 173 11	81 95 21 23 101 76 78 107 63 19 113 24 19 113 24 56 131 157 56 131 157 44 133 24 13 143 9 43 143 9 13 143 9 14 157 16 13 143 16 13 143 9 14 155 114 16 124 10 16 173 114 179 179 119 53 179 119 53 179 119	81 95 21 23 101 76 78 107 63 19 113 24 19 113 24 56 131 157 56 131 157 56 131 157 56 131 157 56 133 90 74 143 9 13 143 90 14 155 164 13 159 90 14 155 114 16 173 11 53 167 210 53 179 119 53 185 236 236 236 236	81 95 21 23 101 76 78 107 63 19 113 24 19 113 24 45 125 164 56 131 157 56 133 24 13 143 9 74 143 9 13 143 9 143 164 17 13 161 124 161 173 10 23 161 124 173 167 210 23 167 210 21 173 119 23 191 134	81 95 21 23 101 95 21 78 107 63 76 19 113 24 76 56 113 24 76 56 131 157 63 56 131 157 9 13 143 9 9 13 149 164 164 13 155 164 164 13 161 124 9 9 161 173 9 9 10 161 155 114 10 114 161 125 114 11 10 238 185 236 119 11 21 197 125 119 125 197 127 134 125 126	81 95 21 23 101 95 21 78 107 63 76 19 113 24 76 45 125 125 164 45 131 157 9 44 131 157 9 13 149 164 9 13 155 164 9 13 149 10 11 13 155 114 9 149 155 114 10 153 161 124 11 161 173 11 11 236 173 111 124 21 191 123 119 236 191 125 140 236 1236 140 140	81 95 21 23 101 95 21 78 107 63 76 19 113 247 76 45 125 125 164 45 131 157 9 46 131 157 9 13 143 9 9 13 155 164 157 39 161 124 10 131 155 114 9 149 155 114 12 39 161 124 11 21 173 111 124 21 173 111 124 21 197 113 111 203 117 125 114 203 123 124 125 203 127 126 114 203 170 126 171	81 95 21 23 101 95 21 78 107 63 76 55 113 24 76 55 131 157 63 44 131 157 63 56 131 157 63 74 131 157 9 74 143 9 164 13 143 90 164 33 155 114 9 53 179 124 124 33 166 124 11 53 179 119 11 53 179 210 210 54 197 123 216 503 170 125 119 515 216 126 216 203 170 125 216 215 216 171 216	81 95 21 23 101 95 78 107 63 19 113 24 44 131 157 56 131 157 56 131 157 74 137 9 74 137 9 74 143 9 13 143 9 13 155 164 13 155 114 13 155 114 13 161 124 13 161 124 13 161 124 13 161 124 13 161 124 13 161 124 13 161 124 13 161 124 13 119 119 13 119 119 13 203 140 13 215 216 21 215 216 21 12 12 12 119 12 13 12 12 140 12 11 15 119	81 95 21 78 101 76 19 113 24 19 113 24 19 113 24 113 24 15 23 131 24 45 133 24 24 133 24 25 133 24 26 133 24 27 143 9 39 161 124 17 173 114 23 161 124 23 161 124 23 161 124 23 161 124 23 161 124 23 161 124 23 161 124 23 161 124 23 210 119 23 210 125 23 216 125 23 216 126 23 140 125 23 140 125 23 140 125 23 140 125 23 140 126 23 140 <	81 95 21 23 101 95 10 113 24 10 113 24 10 113 24 25 131 24 26 131 24 27 125 164 39 167 164 39 167 124 39 161 124 30 173 114 23 161 124 17 191 124 21 191 124 22 191 124 23 123 116 23 123 119 25 210 119 25 191 124 26 173 210 27 203 140 28 236 171 25 215 126 25 221 125 25 160 171 25 236 160 26 171 125 27 154 154 283 160 161 210 154 154	81 95 21 23 101 95 10 113 76 119 113 24 125 131 157 24 131 24 25 131 24 26 131 24 27 125 164 39 167 164 39 167 124 30 167 124 27 203 114 28 185 210 17 191 124 28 185 210 27 203 161 28 133 124 29 210 124 210 123 111 223 123 125 236 216 125 237 125 126 233 126 125 233 140 125 233 151 151	81 95 21 23 101 95 10 113 24 10 113 24 10 113 24 25 131 157 26 131 24 27 125 164 39 167 164 39 167 164 39 167 124 30 179 119 179 179 119 27 203 119 28 185 210 39 167 124 210 123 119 222 191 124 236 216 119 253 215 216 254 236 171 255 233 140 256 233 140 257 151 125 253 233 140 254 233 151 253 151 151	81 95 21 23 101 95 19 113 24 19 113 24 19 113 24 23 101 76 24 131 24 25 131 24 26 133 24 27 143 9 39 161 124 53 179 11 53 179 114 53 179 114 53 167 124 53 167 124 53 159 10 54 191 124 55 221 125 56 216 119 55 236 171 55 233 140 55 233 140 55 233 151 53 236 171 53 236 161 54 233 140 53 233 140 54 233 140 53 236 171 53 233 140 53 233
EMENT EXPONEN	4 2	10 199	16 4	22 189	28 108	34 142	40 201	46 226	52 128	58 33	64 6	70 233	76 80	82 215	88 191	94 181		100 223	100 223 106 178	100 223 106 178 112 110	100 223 106 178 112 110 118 19	100 223 106 178 112 110 118 19 124 245	100 223 106 178 112 110 118 19 124 245 130 156	100 223 106 178 112 110 118 19 124 245 136 144	100 223 106 178 112 110 118 19 124 245 136 144 144 142 113	100 223 106 178 112 118 124 245 136 156 136 144 148 113	100 223 106 178 112 118 124 245 136 156 136 144 144 113 148 114 148 113	100 223 116 178 112 178 118 19 124 245 136 144 142 113 142 113 142 113 148 174 154 245 156 136 160 203	100 223 112 178 112 178 118 19 124 245 130 156 144 113 148 113 148 113 148 113 146 113 154 136 148 113 148 113 148 133 154 133 160 203 166 203 166 203	100 223 1106 178 1112 110 1124 19 124 245 130 156 144 113 148 113 148 113 148 113 148 113 148 113 148 113 154 133 148 133 154 133 154 133 154 133 154 133 154 133 155 133 156 139 157 67	100 223 1106 178 1112 178 1118 19 1118 19 1118 19 1130 156 1316 144 148 113 148 113 148 113 148 113 148 113 148 133 146 133 154 133 154 133 154 133 154 133 154 133 154 133 156 133 178 133 178 133	100 223 112 178 118 19 118 19 124 245 130 144 148 113 148 113 154 245 130 144 148 113 154 113 148 113 154 133 154 133 154 133 154 133 154 133 154 133 154 133 154 133 160 203 178 153 178 153	100 223 112 178 118 19 124 245 130 144 148 174 148 174 154 133 148 174 154 133 148 178 178 155 178 156 178 178 178 153 160 203 178 203 178 203 178 23 178 23 178 23 178 23 178 23 190 22	100 223 1112 1178 1112 1178 1113 144 1136 144 1136 144 1136 144 1136 144 1136 144 1136 144 1136 144 1136 145 144 173 154 174 154 178 154 178 155 174 160 203 178 178 178 178 178 203 160 203 195 67 196 228 197 228 196 228 196 218	100 223 1112 116 1112 1178 1124 245 1136 144 1136 144 1148 113 154 144 154 136 154 136 154 136 154 136 154 139 154 139 155 139 166 203 178 153 178 228 196 228 197 228 197 228 197 228 196 228 197 228 197 278	100 223 1112 118 1118 19 1118 19 1118 14 1130 144 1148 113 154 144 154 144 154 136 154 144 154 136 154 136 154 136 155 139 166 139 178 153 178 139 178 203 184 228 195 41 202 139 203 130 204 130 205 41 206 41 207 130	100 223 112 118 118 196 118 144 1130 144 1142 144 1148 144 1148 144 1154 245 1160 203 154 113 154 113 154 113 154 130 156 139 177 67 178 153 178 153 178 203 178 153 178 203 178 203 178 203 178 203 178 203 178 203 177 64 64 64	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	100 223 112 116 118 178 124 245 130 156 142 144 148 173 148 173 154 265 172 67 178 153 178 153 178 173 178 203 178 203 178 203 178 233 220 139 232 232 233 236 238 236 238 238 238 238 238 238 238 238 238 238 238 255 238 255 238 255 238 255 248 248	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	100 223 1112 116 1118 119 1124 245 1130 124 1130 144 1130 145 1148 113 148 113 154 155 154 156 154 156 166 203 178 153 178 153 178 203 178 203 178 203 178 203 178 203 178 233 220 233 233 234 233 235 255 35 250 15 250 15 15 15 250 25 250 165 250 165
EXPONENT EL	66	205	42	212	49	87	131	148	240	69	56	121	168	252	36	94	186	,,,,	230	230 16	230 16 136	230 16 136 61	230 16 61 158	230 16 136 158 158 8	230 160 158 158 160 160	230 16 136 158 158 160 123	230 16 61 158 158 160 123 91	230 16 136 158 158 160 91 235 235	230 136 136 158 158 160 81 123 91 235 30 30	230 16 136 158 158 158 123 235 170 170	230 16 61 158 158 158 123 123 235 1123 170 115	230 16 136 158 158 158 123 235 123 235 1170 82 1170 82	230 136 61 136 158 158 158 123 235 123 235 123 170 82 115	230 136 136 158 158 160 1123 235 1123 235 115 235 235 235 254	230 136 61 136 158 158 1123 115 235 11 235 235 235 235 235 235 235 235 235 235	230 136 136 158 158 158 158 1170 235 1170 254 254 254 254 255 254 255 254 255 254 255 255	230 136 136 158 158 158 123 158 1170 235 1170 235 1170 254 1170 254 1170 254 1170 254 1170 254 1170 254 1170 254 1170 254 1170 255 1170 1170 1170 1170 1170 1170 1170 11	230 136 136 158 158 158 123 170 1170 235 1170 235 1170 235 1170 330 330 331 332 333 333 333 337 330 330 330 330 330 330	230 136 136 158 158 123 1170 1235 1170 1170 1170 1170 1170 1170 1170 117	230 136 136 158 158 123 123 1170 1170 123 123 123 123 123 123 123 123 123 123	230 136 136 158 158 158 1170 1170 1170 1170 1170 1170 1170 117	230 136 136 158 158 158 1170 1170 1170 1170 1170 1170 1170 117	230 136 136 136 158 158 123 158 1170 123 123 123 123 123 123 123 123 123 123
ELEMENT	e.	6	15	21	27	33	39	45	51	57	63	69	75	81	87	93	66		105	105 111	105 111 117	105 111 117 123	105 111 117 123	105 111 117 123 129	105 111 117 123 129 135	105 111 117 123 129 141 141	105 111 117 117 129 141 141 153	105 111 117 117 129 129 141 153 153	105 111 117 117 129 141 147 153 159	105 111 117 117 129 129 147 147 153 153 159 171	105 111 117 117 129 129 147 147 153 159 171 177	105 111 117 117 129 147 147 147 153 165 171 171 183	105 111 117 117 129 129 129 129 129 171 183 183	105 111 117 123 129 141 147 153 153 171 171 183 183	105 111 117 123 141 141 153 141 153 165 171 171 183 183 195 195	105 111 117 117 129 141 141 153 147 153 147 153 165 165 171 171 183 195 201 207	105 111 117 117 129 141 141 141 153 147 141 153 147 153 165 165 177 177 165 165 177 177 165 177 177 177 177 177 177 177 177 177 17	105 111 117 117 129 141 141 141 141 153 147 147 153 165 165 171 165 177 165 177 165 177 177 165 177 177 177 177 177 177 177 177 177 17	105 111 117 117 129 141 141 147 141 153 147 147 141 153 147 141 153 165 113 165 113 177 165 113 177 165 177 177 177 177 177 177 177 177 177 17	105 111 117 117 129 141 147 141 147 147 147 147 147 147 147	105 111 117 117 129 141 141 141 141 141 142 141 142 141 147 141 147 141 147 141 147 141 147 141 147 141 147 147	105 111 117 117 129 141 141 141 141 142 142 142 142 142 142	105 117 117 117 129 141 141 141 147 141 147 141 147 141 147 141 147 141 147 141 147 141 147 141 147 141 147 147
EXPONENT	Ч	e	107	200	127	Ś	79	190	222	109	244	143	173	202	66	227	40		129	129 54	129 54 34	129 54 14	129 54 34 14	129 54 34 14 27	129 54 14 27 234	129 54 14 27 234 195	129 54 14 27 234 195 81	129 54 14 27 234 195 81	129 54 14 27 234 81 81 219	129 54 14 14 23 23 81 81 81 85	129 54 14 14 23 81 81 81 85 192	129 54 14 14 195 81 81 85 219 219 219 250	129 54 34 14 195 195 219 250 182 182	129 54 14 14 195 195 192 192 182 182 98	129 54 14 14 195 195 192 192 182 224 182	129 54 14 14 195 195 195 195 182 182 182 182 182 182 182 182	129 54 14 14 195 81 195 195 192 198 179 179	129 54 14 14 195 195 81 195 192 239 239 239	129 54 14 14 195 81 195 195 195 1192 111 239 224 239 111	129 54 14 14 195 195 195 195 192 192 192 192 193 111 111	129 54 14 14 195 195 195 195 1192 1192 1192 1192 224 111 239 20 20	129 54 14 14 195 195 195 195 1192 185 111 111 111 111 111 111 111 111 111	129 54 54 14 195 195 195 195 192 185 111 111 20 246 111 20 246
ELEMENT	2	Ø	14	20	26	32	38	44	50	56	62	68	74	80	86	92	98		104	104 110	104 110 116	104 110 116 122	104 110 122 128	104 110 122 128 134	104 110 116 122 134 140	104 110 116 122 134 140	104 110 116 122 134 140 152	104 110 116 122 134 146 152 158	104 110 116 122 134 140 152 158 164	104 110 116 122 134 146 152 158 158 170	104 110 116 122 134 146 158 158 158 170	104 110 116 122 128 146 158 158 158 170 170	104 110 122 128 146 146 146 146 146 176 176 188 188	104 110 1122 1228 1288 1460 1460 1460 1460 176 176 188 194	104 110 1122 122 128 128 152 156 156 156 152 156 156 152 156 152 156 152 156 152 152 152 156 152 152 152 152 152 152 152 152 152 152	104 110 1122 1228 1288 1568 1568 1568 1568 1568 1568 1568 15	104 110 1152 1158 1158 1158 1170 1170 1170 1170 1170 1170 1170 117	104 110 1152 1228 1546 1558 1558 1568 1568 1588 1882 1882 1882 1822 2126 2126 2126 21	104 110 1152 1228 1528 1528 1528 1268 1268 1268 1268 1268 1268 1268 12	104 110 1152 1238 1460 1588 1552 1364 170 170 1882 1882 1882 1988 1882 1948 1882 1948 1882 1948 1882 1948 1948 1948 1948 1948 1948 1948 1948	104 110 1152 1158 1158 1158 1158 1158 1158 1158	104 110 1152 1153 1154 1155 1155 1155 1155 1155 1155	104 110 1152 1153 1154 1553 1155 1155 1155 1155 1155
EXPONENT	0	106	126	78	221	243	172	65	39	53	13	26	194	72	84	249	97		47	47 238	47 238 166	47 238 166 184	47 238 166 184 59	47 238 166 184 59	47 238 166 184 159 28	47 238 166 159 28 29	47 238 166 159 28 28 145	47 238 166 159 28 28 29 161	47 238 166 159 28 28 145 161	47 238 166 59 159 28 28 161 161 146	47 238 166 159 159 145 161 146 175	47 238 166 159 159 161 175 175 175	47 238 166 159 159 145 145 175 175 175	47 238 166 159 159 161 175 175 175 175 175 175 175 175 175 17	47 238 166 159 159 161 174 174 174 197 197 197	47 238 166 159 159 196 197 197 204 31	47 238 166 159 159 159 161 1756 11455 204 204 204 311 86	47 238 166 159 159 1756 1756 1756 1756 1756 1756 1756 1756	47 238 166 159 159 159 161 175 175 175 175 175 175 175 175 175 17	47 238 166 159 159 159 161 175 161 204 204 204 204 204 204 204 204 204 204	47 238 166 159 159 159 161 175 116 204 204 204 251 58 251 58	47 238 166 159 159 159 161 175 1145 176 176 176 176 176 176 176 261 271 251 251	47 238 166 159 159 159 161 175 161 204 204 204 204 205 204 205 204 205 204 205 204 205 204 205 205 205 205 205 205 205 205 205 205
ELEMENT	1	7	13	19	25	31	37	43	49	55	61	67	73	79	85	16	97		103	103 109	103 109 115	103 109 115 121	103 109 115 121	103 109 115 121 133	103 109 121 127 133 139	103 109 121 127 133 139	103 109 121 127 127 133 145 151	103 109 121 127 133 133 151 151	103 109 121 121 127 133 151 151 151	103 109 121 121 127 127 133 151 151 151 163	103 109 121 121 127 127 127 127 125 125 125 125 125 125 125 125	103 109 121 121 121 127 127 127 125 125 125 125 125 125 125 125 125 125	103 109 121 121 121 121 122 122 122 122 122 12	103 109 121 121 122 122 122 122 122 123 123 123	103 109 121 121 122 122 122 122 123 123 123 123	103 109 121 121 121 122 123 123 123 123 123 123	103 109 121 121 121 122 133 152 153 153 155 155 155 155 155 155 155 155	103 109 121 121 122 123 123 125 125 125 125 125 125 125 125 125 125	103 109 121 121 122 123 123 125 125 125 125 125 125 125 125 125 125	103 109 121 121 122 123 123 125 125 125 125 125 125 125 125 125 125	103 109 1121 121 122 122 122 122 122 122 122 1	103 109 1121 121 121 122 122 122 223 223 223 22	103 103 1121 121 121 122 122 122 122 122 122 1
EXPONENT	*	100	101	206	102	43	207	213	103	50	77	88	208	132	214	149	104	176	T # 7	51 51	51 51 70	41 51 45	51 51 45 57	241 51 89 89	241 51 70 89 122	241 51 70 45 89 122 209	241 51 57 57 89 122 169	241 51 57 57 89 122 169 133	241 51 51 51 51 51 253 253 253	241 51 51 51 51 253 122 122 153 253 253	241 51 52 89 89 122 169 1169 215 215 37	241 51 51 52 53 209 215 253 215 253 215 253 215	241 51 57 57 57 57 209 215 215 215 215 215 215	241 51 57 57 57 57 209 215 215 215 215 215 215 215 215 215 215	241 51 57 57 57 57 209 215 215 215 215 215 215 215 215 215 215	241 51 57 57 57 57 50 122 253 215 215 215 215 253 215 253 215 253 215 253 253 253 253 253 253 253 253 253 25	241 51 57 57 57 57 57 57 57 57 57 57 57 57 57	241 51 57 57 57 52 53 52 53 52 52 52 52 52 52 52 52 52 52 52 52 52	241 51 51 52 53 53 53 53 53 53 53 53 53 53 53 53 53	241 51 51 52 53 53 53 53 53 53 53 53 53 53 53 53 53	241 51 51 52 53 53 53 53 53 53 53 53 53 53 53 53 53	241 51 51 533 533 531 553 531 553 531 553 531 553 531 553 531 553 531 553 531 553 553	241 253 253 253 253 253 253 253 253 253 253
LEMENT	0	9	12	18	24	30	36	42	48	54	60	66	72	78	84	90	96	,	102	102 108	102 108 114	102 108 114 120	102 108 114 120 126	102 108 114 120 126 132	102 108 114 120 132 132	102 108 114 120 126 132 138	102 108 114 126 132 138 138 150	102 108 114 126 132 138 138 150 150	102 108 114 126 132 138 138 156 150	102 108 114 126 132 138 150 156 156	102 108 114 126 132 138 138 156 156 156	102 108 114 126 132 138 156 156 156 156 156	102 114 126 126 132 156 156 168 168 174 180	102 108 114 126 132 132 156 156 156 168 186 192	102 108 114 126 126 133 144 156 168 168 192 192 192	102 114 114 126 132 144 156 156 168 186 192 198 192 192	102 108 114 126 132 156 156 162 168 168 168 168 198 198 198 198 198 198	102 108 114 126 132 156 156 168 168 168 168 198 198 198 198 198 198 198 198 198 19	102 108 114 126 126 138 138 168 168 168 168 168 168 192 192 192 216 216 222	102 108 114 126 126 138 138 168 168 168 168 168 168 198 198 198 198 198 198 198 198 198 19	102 108 114 126 132 132 156 156 156 162 168 188 198 198 198 198 198 198 198 198 19	102 108 114 126 132 132 156 156 156 156 156 156 156 156 156 156	102 108 114 126 132 156 156 156 156 156 156 156 156 156 156

ORIGINAL PAGE IS OF POOR QUALITY

14

in the second process process and the second se

Table A2. Generator Polynomial Coefficients for

			143					
		= (::) d	i= 112	ı ^{ij}), where	e j=1,7,11	13,19,23,37	and 43.	
DES (%)	r=2	, ⁷ =173	11=173	13 <u>=</u> 61	19=222	²³ =251	37 = 174	, ⁴³ =30
C	1	-4	-1	4	-1	4	e -4	P-4
Г	236	227	213	174	169	139	a)	73
2	244	151	154	158	247	11	135	192
٣	220	109	191	204	73	140	135	155
4	133	17	132	153	177	63	137	245
ŝ	238	701	36	89	80	41	65	160
ę	137	26	144	77	53	223	130	94
7	201	134	194	16	59	97	123	159
80	2	213	6	154	129	236	198	171
6	171	t-	59	158	106	41	44	ŝ
10	11	133	67	188	103	83	102	240
11	226	174	81	234	18	156	117	129
12	34	238	56	191	17	192	77	124
13	252	97	158	7.2	184	145	138	188
14	209	157	204	226	64	98	214	242
15	22	232	74	59	83	31	106	173
16	78	\$	135	7.8	168	220	125	80
17	22	232	74	59	83	31	901	173
18	209	157	204	226	64	98	214	242
19	252	97	158	72	184	145	138	188
20	34	238	56	191	47	102	17	124
21	226	174	81	234	18	156	117	129
22	11	133	67	183	103	86	102	240
23	141	-1	59	155	106	41	77	2
24	7	213	6	154	129	236	198	171
25	291	184	194	16	59	67	123	159
ع ت	137	26	144	77	53	223	130	94
27	238	104	36	<u>69</u>	80	43	65	160
28	133	47	132	153	177	63	167	246
29	220	109	106	204	73	140	138	158
30	244	151	184	158	247	11	135	192
31	236	2.2.2	213	174	169	139	ε	73
32	-	1			1	-1	1	-4

ORIGINAL PAGE IS OF POOR QUALITY,

ORIGINAL PAGE IS OF POOR QUALITY AND ٠ 6⁺¹ S 1-BIT SHIFT REGISTER AND ٠ , 1-BIT Shift Register S AND + 9:+5 1-BIT Shift Register S AND + ₹ 6 1-BIT Shift Register ŝ AND SERIAL OUTPUT + 9:+3 S 1-BIT SHIFT REGISTER 8-BIT SHIFT REGISTER AND 9₁₊₂ S 1-BIT SHIFT REGISTER FINITE FIELD ADDER GATE + AND 6. 1-BIT Shift Register ŝ AND ő 1-BIT Shift Register SWITCH NO. 2 OUTPUT Ś

1.844.14

٠

Logic structure of a serial parallel finite field multiplier which performs $a^{1} + a^{1} + a^{2} + a^{2} + a + 1$. i modulo Figure Al.

Λ-6

REFERENCES

- [A1] "Guidelines for Data Communications Standards: Space Telemetry Channel Coding (Issue-3)," prepared by the NASA/ESA Working Group for space data system standardization (NEWG), Jan. 1982.
- [A2] Perlman, M. and Lee, J., "Reed-Solomon Encoders Conventional Versus Berlekamp's Architecture," JPL IOM 3610-81-119 (internal document), July 10, 1981 (also to appear as JPL publication).

11