Freescale Semiconductor
Application Note

Document Number: AN2918
Rev. 0, 06/2005

Correcting Single-bit Errors with CRC8 in ATM Cell Headers

by *Bo Lin*
Computing Platform Division
Freescale Semiconductor, Inc.
East Kilbride, Scotland

1 Introduction

It is well known that CRC is a class of codes providing strong error detection. However, a CRC code can also be used to correct up to two error bits in a data block (Blahut, 1983). In many cases, such as ITU-T I.432.1 (1999), a CRC code is required to correct a single-bit error in a data block.

This application note gives a generic introduction on forward error correction code and describes a very efficient table lookup method to correct a single-bit error in an ATM cell header defined by ITU-T I.432.1.

Contents

freescale™ semiconductor

Let $d_{N-1}\ d_{N-2} \ldots d_1\ d_0$ denote an N-bit data block D where D's polynomial representation is

$$D(x) = d_{N-1}x^{N-1} + d_{N-2}x^{N-2} + \ldots + d_1x + d_0.$$

A single-bit error occurs in data block D means that there exists a j with $0 \le j < N$ such that d_j is inverted to $\overline{d}_j$ by the single-bit error.

This is equivalent to represent a corrupted data block associated with D as $\tilde{D}(x) = D(x) + x^j$ where + denotes bit-wise modulo 2 addition or simply ⊕.

On the other hand, given a binary string $B = b_{N-1}\ b_{N-2} \ldots b_1\ b_0$, B's $CRC8$ can be calculated as

$$CRC8(B) = B(x)x^8 \bmod G(x)$$

where $G(x)$ is the generator polynomial. It is defined as

$$G(x) = x^8 + x^2 + x + 1$$

in ITU-T I.432.1. As a result, D's CRC8 can be calculated as

$$CRC8(D) = D(x)x^8 \bmod G(x).$$

The data block $D||CRC8$ is transmitted. In other words, $D(x)x^8 + CRC8(x)$ is transmitted.

On the receiving side, $D'||CRC8'$ is received where there might be a single-bit error in the received data string $D'||CRC8'$. This means that the

$$\text{data block } D'(x)x^8 + CRC8' = D(x)x^8 + CRC8(D) + e\cdot x^j$$

is received, where $0 \le j \le (N - 8)$ and e is either 0 or 1.

By verifying CRC8 over the received data block (data part only), we have

$$V(D') = ((D(x) + f\cdot x^i)\cdot x^8 \bmod G(x) \text{ (there might be single-bit error at location } i)$$
$$= CRC8(D) + (f\cdot x^i)\cdot x^8 \bmod G(x),$$

where $0 \le i \le N$ and f is either 0 (no error) or 1 (single-bit error in data).

On the other hand,

$$\text{the received CRC checksum } CRC8' = CRC8(D) + h\cdot x^j \bmod G(x),$$
$$\text{where } (N - 1) < j < (N + 8) \text{ and } h \text{ is either 0 (no error) or 1}$$

which indicates the possible single-bit error in CRC checksum field. It is noticed that only one of f or h can be 1 since the code is designed to correct at most one error over $D'||CRC8'$.

By adding the re-calculated $V(D')$ and the received $CRC8'$ together, we have

$$S = V(D') + CRC8' = (CRC8(D) + (f\cdot x^j)\cdot x^8 + CRC8(D) + (h\cdot x^j)\cdot x^8) \bmod G(x)$$
$$= ((f\cdot x^i)\cdot x^8 + h\cdot x^j) \bmod G(x) \text{ (note: + = ⊕, } N\ 0 \le i \le N \text{ and } N \le j < (N + 8))$$
$$= e\cdot x^j \bmod G(x) \text{ with } 0 \le j < (N + 8)$$

since only one of f or h can be 1 and the e is used to replace the two with a wider range.

If there is no error, then the S is 0, otherwise, $S = x^j \bmod G(x) = S(j)$ indicating an error at logical location j.

To correct the single-bit error, the index j needs to be determined from the calculated $S(j)$ on the receive side. For a CRC code, it is proven that

$x^i \bmod G(x) \neq x^j \bmod G(x)$ if $i \neq j$ (Blahut, 1983). In coding theory, $S(j)$ is called error syndrome since it indicates the single-bit error and gives clue of its index or location j.

This application note gives a simple table lookup method to determine j. The method generates a very small code footprint and can be efficiently implemented by many of Freescale's platforms such as the e600, e500, and CPM (communications processor module).

2 Correcting Single-Bit Errors in ATM Cell Headers

An ATM cell header can be described in Figure 1 where the 0, 1, …, 39 is the logical index of the bits in an ATM cell header where data occupies bits 8 to 39 while HEC (header error check, which is the CRC8 value of data) has logical bits 0 to 7.

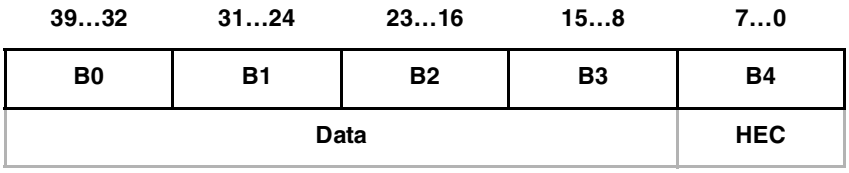

Figure 1. ATM Cell Header Format

By denoting

txHeader = txData|txHEC for the ATM cell header being transmitted and

rxHeader = rxData|rxHEC for the received.

The prefix tx indicates 'transmitted' and rx indicates 'received'. The following procedure calculates the syndrome for single-bit error correction:

Procedure syndromeCalculate()

1. Calculate verHEC over rxData as verHEC = CRC8(rxData)
2. return (S = verHEC ⊕ rxHEC)

After obtaining the syndrome value S, the next step is to locate the error bit. We need to find the logical index j associated with S.

Since $x^i \bmod G(x) \neq x^j \bmod G(x)$ if $i \neq j$, this means each of $x^0 \bmod G(x)$, $x^1 \bmod G(x)$, $x^2 \bmod G(x)$, …, and $x^{39} \bmod G(x)$ has a different value. By building a table with entries as (j, $x^j \bmod G(x)$) and searching the table with the calculated syndrome value, the logical index of the single-bit error can be determined. If the syndrome value is in the table, the error index is found. Otherwise, this means that a multiple-bit error has occurred and the error is uncorrectable.

Table 1. The Error-Location and Syndrome-Search Table

errorLoc	syndrome	errorLoc	syndrome	errorLoc	syndrome	errorLoc	syndrome
0	1	10	28	20	87	30	134
1	2	11	56	21	174	31	11
2	4	12	112	22	91	32	22

Table 1. The Error-Location and Syndrome-Search Table (continued)

3	8	13	224	23	182	33	44
4	16	14	199	24	107	34	88
5	32	15	137	25	214	35	176
6	64	16	21	26	171	36	103
7	128	17	42	27	81	37	206
8	7	18	84	28	162	38	155
9	14	19	168	29	67	39	49

For example, if syndrome = 91 is calculated, by searching Table 1, errorLoc = 22 is obtained. By comparing with Figure 1, it is found that B2's second bit is corrupted and can be corrected by B2 = B2' ⊕ 0100000. On the other hand, if syndrome = 123 is calculated, by searching Table 1, no errorLoc will be found. This means an uncorrected error or multiple-bit error has occurred in the ATM cell header.

The above approach can be described in the following procedure:

Procedure errorLocationSearch()

1. *S* = syndromeCalculate()
2. return errorLoc = Search Table 1 with *S*

One drawback of the above single-bit error approach is that it requires a search operation on an unsorted (unordered) table which is not very efficient. However, by taking advantage of the small size of the table, sorting the table with errLoc, and padding the illegal entries with a special value, such as –1, Table 2 can be obtained for efficiently locating a single-bit error.

Table 2. Syndrome-Indexed Error-Location Index Table

syndrome	errorLoc	syndrome	errorLoc	syndrome	errorLoc	syndrome	errorLoc
0	–1	64	6	128	7	192	–1
1	0	65	–1	129	–1	193	–1
2	1	66	–1	130	–1	194	–1
3	–1	67	29	131	–1	195	–1
4	2	68	–1	132	–1	196	–1
5	–1	69	–1	133	–1	197	–1
6	–1	70	–1	134	30	198	–1
7	8	71	–1	135	–1	199	14
8	3	72	–1	136	–1	200	–1

Table 2. Syndrome-Indexed Error-Location Index Table (continued)

9	–1	73	–1	137	15	201	–1
10	–1	74	–1	138	–1	202	–1
11	31	75	–1	139	–1	203	–1
12	–1	76	–1	140	–1	204	–1
13	–1	77	–1	141	–1	205	–1
14	9	78	–1	142	–1	206	37
15	–1	79	–1	143	–1	207	–1
16	4	80	–1	144	–1	208	–1
17	–1	81	27	145	–1	209	–1
18	–1	82	–1	146	–1	210	–1
19	–1	83	–1	147	–1	211	–1
20	–1	84	18	148	–1	212	–1
21	16	85	–1	149	–1	213	–1
22	32	86	–1	150	–1	214	25
23	–1	87	20	151	–1	215	–1
24	–1	88	34	152	–1	216	–1
25	–1	89	–1	153	–1	217	–1
26	–1	90	–1	154	–1	218	–1
27	–1	91	22	155	38	219	–1
28	10	92	–1	156	–1	220	–1
29	–1	93	–1	157	–1	221	–1
30	–1	94	–1	158	–1	222	–1
31	–1	95	–1	159	–1	223	–1
32	5	96	–1	160	–1	224	13
33	–1	97	–1	161	–1	225	–1
34	–1	98	–1	162	28	226	–1
35	–1	99	–1	163	–1	227	–1

Table 2. Syndrome-Indexed Error-Location Index Table (continued)

36	–1	100	–1	164	–1	228	–1
37	–1	101	–1	165	–1	229	–1
38	–1	102	–1	166	–1	230	–1
39	–1	103	36	167	–1	231	–1
40	–1	104	–1	168	19	232	–1
41	–1	105	–1	169	–1	233	–1
42	17	106	–1	170	–1	234	–1
43	–1	107	24	171	26	235	–1
44	33	108	–1	172	–1	236	–1
45	–1	109	–1	173	–1	237	–1
46	–1	110	–1	174	21	238	–1
47	–1	111	–1	175	–1	239	–1
48	–1	112	12	176	35	240	–1
49	39	113	–1	177	–1	241	–1
50	–1	114	–1	178	–1	242	–1
51	–1	115	–1	179	–1	243	–1
52	–1	116	–1	180	–1	244	–1
53	–1	117	–1	181	–1	245	–1
54	–1	118	–1	182	23	246	–1
55	–1	119	–1	183	–1	247	–1
56	11	120	–1	184	–1	248	–1
57	–1	121	–1	185	–1	249	–1
58	–1	122	–1	186	–1	250	–1
59	–1	123	–1	187	–1	251	–1
60	–1	124	–1	188	–1	252	–1
61	–1	125	–1	189	–1	253	–1

Table 2. Syndrome-Indexed Error-Location Index Table (continued)

62	–1	126	–1	190	–1	254	–1
63	–1	127	–1	191	–1	255	–1

For example, if syndrome = 91 is calculated, by looking up Table 2, errorLoc = 22 is obtained. On the other hand, if syndrome = 123 is calculated, by looking up Table 2, errorLoc = –1 is returned and this indicates that an uncorrected error or multiple-bit error has occurred in the ATM cell header.

Table 2 only occupies 256 bytes, and its lookup procedure is extremely efficient and can be easily implemented in various Freescale processors.

3 A Pseudo-Procedure and Numerical Examples

The following pseudo-procedure summarizes the table lookup method of correcting single-bit errors in ATM cell headers:

Procedure errorCorrectionIndex()

1. *S* = syndromeCalculate();
2. if (*S* == 0)
3. return (“no error”);
4. else
5. errLoc = lookup Table 2 with *S*;
6. if (errLoc < 0)
7. return (“uncorrectable error”);
8. else
9. if (0 ≤ errLoc ≤ 7) // i.e. a single-bit error in HEC
10. return (“no error in rxData”);
11. else
12. errorPattern = (`0x00000001`)<<(errorLoc – 8);
13. rxData = rxData ⊕ errorPattern
14. return (“error corrected”)
15. endif
16. endif
17. endif

Example 1: Single-bit error case

txCell = txData|txHEC = `0x30313233|0x69`

rxCell = rxData|rxHEC = `0x31313233|0x69`

verHEC = CRC8(`0x31313233`) = `0x7F`

S = verHEC ⊕ rxHEC = 0x7F ⊕ 0x69 = 0x16 = 22

errorLoc = 32 after looking up Table 2 with index 22.

errorPattern = 0x000000001<<(32 – 8) = 0x01000000

rxData = 0x31313233 ⊕ 0x01000000 = 0x30313133

return (“error corrected”);

Example 2: 2 bit error case

txCell = txData|txHEC = 0x30313233|0x69

rxCell = rxData|rxHEC = 0x31333233|0x69

verHEC = CRC8(0x31313233) = 0x7F

S = syndromeCalulate() = verHEC ⊕ rxHEC = 0xA9 ⊕ 0x69 = 0xC0 = 192

errorLoc = –1 after looking up Table 2 with index 192.

Return (“uncorrectable error”)

4 Summary

A single-bit error correction can be achieved by calculating a CRC value on the received data field, XORing the calculated CRC with the received CRC checksum to obtain an error syndrome. By looking up Table 2 with the error syndrome, the logical location of a single-bit error can be found. The single-bit error can be corrected by reversing the error bit.

5 References

Blahut, R.E, *Theory and Error Control Codes*. Reading: Addison-Wesley Publishing Company, Inc., 1983.

ITU-T I.432.1 (International Telecommunication Union), “B-ISDN user-network interface—Physical layer specification: General characteristics”, 02/1999.

6 Revision History

Table 3 provides a revision history for this application note.

Table 3. Document Revision History

Revision Number	Date	Substantive Change(s)
0	06/08/2005	Initial public release.

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

email:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN2918
Rev. 0
06/2005

freescale™ semiconductor