A Division-Free(255,k) RS code Syndrome Computation Scheme for PC-based DVB-T Software Radio Implementation

Shu-Ming Tseng
Department of Electronic Engineering
National Taipei University of Technology, Taipei 106,Taiwan
Email: shuming@ntut.edu.tw

Yao-Teng Hsu
Liteon Corp., Taipei, Taiwan
Email: Matthew.Hsu@liteon.com

Jheng-Zong Shih
Department of Electronic Engineering
National Taipei University of Technology, Taipei 106,Taiwan
Email: hallway330@hotmail.com

***Abstract*—In this paper, we propose a novel division-free algorithm of syndrome evaluation for (255,k) Reed Solomon code on PC-based software radio platform. The proposed algorithm has reduced execution time of syndrome evaluation. The syndrome evaluation with the proposed division-free algorithm is almost three times faster then typical one.**

***Index Terms*—modulo, division-free algorithm, PC platform**

I. INTRODUCTION

PC-based software radio implementation has been become popular because the ease of implementing and modifying the baseband processing algorithms. In [1], a PC-based software GPS receiver is implemented. In [2], a real-time notebook PC-based Digital Audio Broadcasting (DAB) receiver is implemented.

Based on our previous work in DAB, we now want to implement PC-based Digital Video Broadcasting-Terrestrial (DVB-T) software receiver. However, the bottleneck for real-time processing is the Reed-Solomon (RS) code syndrome computation, which accounts for more than half of the total computation time in our PC-based DVB-T software receiver using C programming, as shown in Table. 1.

In this paper, we propose a simplified algorithm of syndrome evaluation for GF(2^8) for x86 PC platform to reduce the computation time. We replace the modulo (division) operation by addition operation to reduce the massive execution time. The new algorithm also can be executed parallel with SSE parallel instruction to get more gain.

This paper is organized as follows: In Section II, we briefly describe the syndrome evaluation. In Section III, we describe the proposed algorithm. The simulation result is given in Section IV. Section V is the conclusion.

II. THE SYNDROME EVALUATION

We focus on (255,k) RS codes because it is popular in industry standards. Let 255-k=2t, where t is the number of correctable symbol errors.

The received code word is

$$\bar{r}(x) = \bar{v}(x) + \bar{e}(x) \tag{1}$$

where $\bar{v}$ is the transmitted codeword, and $\bar{e}$ is error polynomial as

$$\bar{e} = e_{j_1} x^{j_1} + e_{j_2} x^{j_2} + \cdots + e_{j_v} x^{j_v} \tag{2}$$

The $\{e_{j_1}, e_{j_2}, \cdots, e_{j_v}\}$ are error values and $\{x^{j_1}, x^{j_2}, \cdots, x^{j_v}\}$ are error positions.

$$e_j \in GF(2^m) \quad \text{and} \quad x \in GF(2^m) \tag{3}$$

Syndromes are evaluated as

$$\begin{aligned} S_1 &= r(\alpha^1) = r_0\alpha^{1\cdot 0} + r_1\alpha^{1\cdot 1} + \cdots + r_{2^m-1}\alpha^{1\cdot(2^m-1)} \\ S_2 &= r(\alpha^2) = r_0\alpha^{2\cdot 0} + r_1\alpha^{2\cdot 1} + \cdots + r_{2^m-1}\alpha^{2\cdot(2^m-1)} \\ &\vdots \\ S_{2t} &= r(\alpha^{2t}) = r_0\alpha^{(2t)\cdot 0} + r_1\alpha^{(2t)\cdot 1} + \cdots + r_{2^m-1}\alpha^{(2t)\cdot(2^m-1)} \end{aligned} \tag{4}$$

III. THE PROPOSED SIMPLIFIED ALGORITHM

The mod operation is achieved with DIV (division) instruction in x86 platform as there is no mod instruction in that platform. According to the Intel report [3], we can know the latency typical DIV instruction is 23 CPU cycle. It is far larger then an add instruction which latency is 0.5 CPU cycle, so it is a great hotspot of performance. If we can replace the DIV instruction with ADD, SUB or other low latency instructions, we are closer to real-time software-radio implementation.

The conventional x mod 255 scheme is shown in Fig. 1, where anti_log is the inverse function of the logarithm function. For further analysis, we can divide the x mod 255 operation into 3 cases

- Case I: if $x < 255, ah = 0$

$$\begin{aligned} & x \bmod 255 \\ &= (0 \times 256 + al) \bmod 255 \\ &= (al) \bmod 255 \\ &= al \\ &= ah + al \end{aligned} \tag{5}$$

- Case II: if $x = 255, al = 255, ah = 0$

$$\begin{aligned} & ah + al = 255 \\ & x \bmod 255 = 0 \neq ah + al \end{aligned} \tag{6}$$

- Case III: if $256 \le x < 510, ah = 1$

$$0 \le al < 255$$

$$\begin{aligned} & x \bmod 255 \\ &= (ah \times 256 + al) \bmod 255 \\ &= (1 \times 256 + al) \bmod 255 \\ &= (1 + al) \bmod 255 \\ &= 1 + al \\ &= ah + al \end{aligned} \tag{7}$$

The result is list in Table 2. From this table we can notice that most case of (x mod 255) can be replaced with ah+al. With x=255, this relation can not be keep because of x mod 255 = 0 ≠ ah + al = 255 . In order to avoid this exception, we consider another approach as "(x+1) mod 255". The result of this approach is showed as Table 3. With this table, we note that the column "(al+ah-1) of (x+1)" is the same as "x mod 255" for x<510.

With this relationship, we design a new flow as Fig. 2. The proposed contains no division operations.

IV. SIMULATION RESULTS

The throughput of syndrome evaluation of typical algorithm and proposed algorithm is measured with a x86 computer which has Intel Pentium M 1.4Ghz CPU and brief specific is show in Table 4. The development tool is Microsoft Visual Studio 2005 Team Edition for software developers. The performance measuring tool is Performance explorer within Microsoft Visual Studio 2005. The result of evaluation is listed in Table 5. The throughput of syndrome evolution with our proposed algorithm in the testing platform is 85470 packets per second, and it is 3.06 times faster then typical algorithm (27917 packets per second).

V. CONCLUSION

The Reed Solomon code is very common in many communication systems, and software decoding of that in x86 PC is a problem in the past. Our proposed algorithm is division-free and thus much faster. It is very helpful for software radio related research in the future.

REFERENCE

[1] N. Kubo, S. Kondo and A. Yasuda, "Evaluation of code multipath mitigation using a software GPS receiver," IEICE Trans. Commun., Vol. E88-B, No.11, pp. 4204-4211, November 2005.

[2] S. M. Tseng, Y. T. Hsu, M. C. Chang and H. L. Chan, "A Notebook PC Based Real-Time Software Radio DAB Receiver," IEICE Transactions on Communications, vol. E89B, no. 12, pp. 3208-3214, Dec. 2006.

[3] R. Gerber, "The Software Optimization Cookbook," Intel, pp.63, 2002

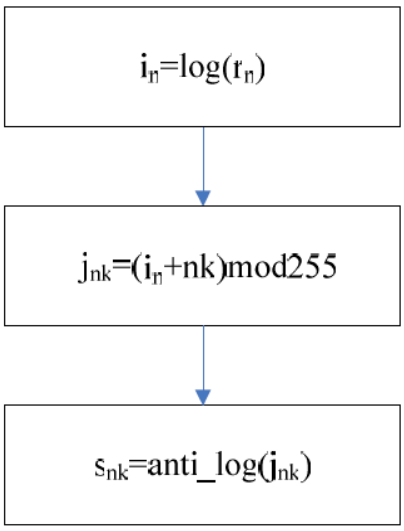

Fig. 1. The conventional scheme to compute $r_n\alpha^{nk}$ in GF(256)

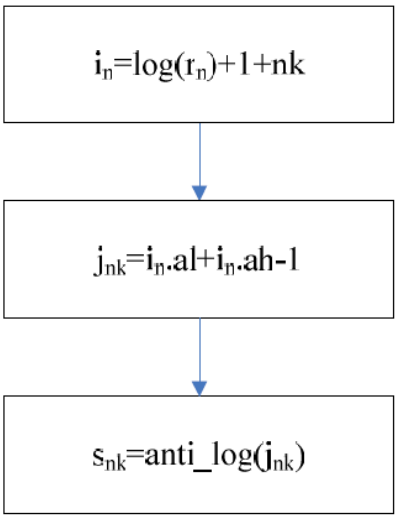

Fig. 2. The proposed scheme to compute $r_n\alpha^{nk}$ in GF(256)

Table 1. Computation time of RS decoder using C programming

Get Syndrome	59.7%
Berlekamp-Massey algorithm	1.8%
Chien search	37.7%
Forney algorithm	0.8%

Table 2. x mod 255 result

	x	x mod 255	ah+al
CASE I	$0 \le x < 255$	x	X
CASE II	255	0	255
CASE III	$256 \le x < 510$	x mod 255	x mod 255

Table 3. (x+1) mod 255 result

	x	x mod 255	ah+al for (x+1)	ah+al-1 for (x+1)
CASE I	$0 \le x < 254$	x	x+1	x
CASE II	254	254	255	254
CASE III	255	0	1	0
CASE IV	256	1	2	1
CASE V	$256 < x < 510$	x-255	x -254	x-255

Table 4. Platform Specification

Component	**Spec**
OS	Windows XP
CPU	Intel Pentium M 1.4G
Ram	256MB
Type	ASUS M2400N

Table 5. Performance comparison

Name	**Throughput**
Typical algorithm	27917 packets/s
Our proposed algorithm	85470 packets/s