

Algorithms for High-speed Generating CRC Error Detection Coding in Separated Ultra-precision Measurement

This article has been downloaded from IOPscience. Please scroll down to see the full text article.

2006 J. Phys.: Conf. Ser. 48 228

(<http://iopscience.iop.org/1742-6596/48/1/042>)

View [the table of contents for this issue](#), or go to the [journal homepage](#) for more

Download details:

IP Address: 38.107.179.213

The article was downloaded on 16/02/2012 at 01:27

Please note that [terms and conditions apply](#).

Algorithms for High-speed Generating CRC Error Detection Coding in Separated Ultra-precision Measurement

Z Zhi, J B Tan, X D Huang and F F Chen

Harbin Institute of Technology, Harbin, 150001, China

Abstract. In order to solve the contradiction between error detection, transmission rate and system resources in data transmission of ultra-precision measurement, a kind of algorithm for high-speed generating CRC code has been put forward in this paper. Theoretical formulae for calculating CRC code of 16-bit segmented data are obtained by derivation. On the basis of 16-bit segmented data formulae, Optimized algorithm for 32-bit segmented data CRC coding is obtained, which solve the contradiction between memory occupancy and coding speed. Data coding experiments are conducted triumphantly by using high-speed ARM embedded system. The results show that this method has features of high error detecting ability, high speed and saving system resources, which improve Real-time Performance and Reliability of the measurement data communication.

1. Introduction

In order to adjust ultra-precision measurement to the requirement of nanometer measurement, we must rigidly control the support environment, including air environment, thermal environment, vibration environment, acoustic environment, light environment and electromagnetic environment etc. The requirement of the Measuring environment is higher and higher, along with the requirement of the accuracy in measurement getting higher and higher. Nevertheless, data processing system in measurement can have much impact on the support environment. For instance, the super-computer can emit heat, electromagnetic field when working. And its heat radiating equipment can cause Vibration and air blasting. Operators have to enter the measuring field having much impact on the ultra-precision measuring environment.

According to the features of the ultra-precision measurement support environment, research is done to reduce the impact to the support environment from the data processing system. Presently, The general method is separating the measuring field and the data processing system. Signals from sensors are digitized and transmitted to the super-computer far away through a high-speed, low power consumption and small volume ARM embedded system. Data is processed by the super-computer. The system block diagram is shown as Figure 1.



Figure 1. Block diagram of measuring system.

In order to make sure the accuracy and speed of data transmission between measuring field and data processing system, solve the contradiction between error detection, transmission rate and system resources in data transmission of ultra-precision measurement, a kind of algorithm for high-speed generating CRC code has been put forward in this paper. Theoretical formulae for calculating CRC code of 16-bit segmented data are obtained by derivation. On the basis of 16-bit segmented data formulae, Optimized algorithm for 32-bit segmented data CRC coding is obtained, which solve the contradiction between memory occupancy and coding speed. Data coding experiments are conducted triumphantly by using high-speed ARM embedded system. The results show that this method has features of high error detecting ability, high speed and saving system resources, which improve Real-time Performance and Reliability of the measurement data communication.

2. CRC generating algorithms

CRC is a kind code with high error detecting ability. CRC-32 is used in the paper. Its generating polynomial is:

$$g(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1.$$

It can detect all odd errors, all random errors no longer than 14 and all single burst errors no longer than 32 in a frame of information. The detectable rate for single burst errors with length of 33 is $1-2^{-31}$. The detectable rate for single burst errors longer than 33 is $1-2^{-32}$. The maximum code length detectable is $2^{31}-1$.

CRC is generated based on modulo 2 operations. If the code is generated by bits, there are some disadvantages: long calculating time and complex to realize. So in practice, generating via table look-up method is popular [2,4]. Literature at present show that most of the tables used in this method are 8-bit segmented data, based on the early-stage 8-bit single chip processors, which have less data width and lower system storage. This paper put forward a table look-up method of 16-bit segmented data, giving full play to the 32-bit data width RISC microprocessor. 32-bit segmented data method is obtained for optimization.

2.1. 16-bit segmented data generating via table look-up algorithm for CRC

In 16-bit segmented data table look-up algorithm, table for all residue 16-bit information code is calculated in advance. $2^{16} \times 4$ (256K) bytes of storage is need to store this table. The whole information code is divided into segments of 16 bits. Residue for each segment is obtained via table look-up. Then we can get the CRC code for the whole information code according to the element of table look-up in segments. 16-bit segmented data table look-up method process data of 16 bits in each time, which is 2 times of data bits in 8-bit segmented data table look-up method, so the speed of this method in theory should be 2 times of the speed of 8-bit segmented data table look-up method. Therefore, this method is important to improve the real-time performance of the measuring system.

2.1.1. Element of table look-up method between 16-bit data segments. Information codes $M(x)$ are divided into segments of 16 bits. Each segment is marked by $D_j(x)$, $j+1$ segments in total:

$$M(x)=D_0(x)+D_1(x)+\dots+D_j(x)$$

Let $M_i(x)$ to be the current information code:

$$M_0(x)=D_0(x);$$

$$M_1(x)=x^{16}M_0(x)+D_1(x);$$

.....

$$M_{n+1}(x)=x^{16}M_n(x)+D_{n+1}(x);$$

$R_n(x)$ is the residue of the calculation for the n time, we have:

$$R_n(x) = x^{32}M_n(x) \bmod g(x).$$

Then $R_{n+1}(x)$ is the residue for the $n+1$ time:

$$R_{n+1}(x) = x^{32}M_{n+1}(x) \bmod g(x) = x^{32}x^{16}M_n(x) \bmod g(x) + x^{32}D_{n+1}(x) \bmod g(x) = x^{16}R_n(x) \bmod g(x) + x^{32}D_{n+1}(x) \bmod g(x) \quad (1)$$

Let $R_n(x)=x^{16}R_{nH}(x)+R_{nL}(x)$, where $R_{nH}(x)$, $R_{nL}(x)$ are polynomials of order less than 15. Insert into equation (1), we got:

$$R_{n+1}(x) = x^{16}x^{16}R_{nH}(x) \bmod g(x) + x^{16}R_{nL}(x) \bmod g(x) + x^{32}D_{n+1}(x) \bmod g(x) \\ = x^{32}[R_{nH}(x) + D_{n+1}(x)] \bmod g(x) + x^{16}R_{nL}(x) \tag{2}$$

Formula (2) shows the recursion relation via table look-up between segments.

2.1.2. *Algorithm description.* We summarize the algorithm via table look-up according to the derivation above (“+” operation is modulo 2 addition, the same for below):

- (1) Initialize CRC as $R_H=0, R_L=0$;
- (2) Read current information D , calculate: $I = R_H + D$;
- (3) Use I as the index to look up the table, result: R_{HH} and R_{LL} ;
- (4) Calculate the new CRC: $R_H = R_{HH} + R_L, R_L = R_{LL}$;
- (5) If there’s information to be process, go to 2)
- (6) CRC calculation completed, output R_H and R_L .

2.2. CRC generating algorithm for optimization

The speed of 16-bit segmented data table look-up method in theory is 2 times of the speed of 8-bit segmented data table look-up method. If 32-bit segmented data table look-up method is used, the speed will be higher. Nevertheless, the table for 32-bit segmented data table look-up method needs $2^{32} \times 4(16G)$ bytes of storage which is not practical. We put forward a kind of optimized 32-bit segmented data table look-up method, dividing 32-bit data segments into sub-segments and look up table in sub-segments [1,4]. This method needs $2 \times 2^{16} \times 4(512K)$ bytes of storage.

2.2.1. *Element of table look-up method in 32-bit data segments.* Assume $I(x)$ is the index to be look up. $I(x)$ can be obtained via xor operation between two group of polynomials:

$$I(x) = I_a(x) + I_b(x)$$

where high 16 bits of $I_a(x)$ are derived from high 16 bits of $I(x)$, low 16 bits are all 0; low 16 bits of $I_b(x)$ are derived from low 16 bits of $I(x)$, high 16 bits all 0. They are shown as Figure 2 as below:

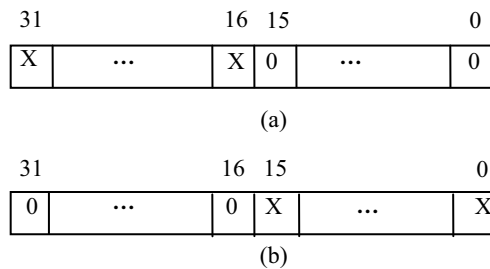


Figure 2. Diagrammatic chart for group a and b.

Then the procedure of table look-up becomes:

$$x^{32} I(x) \bmod g(x) = x^{32} I_a(x) \bmod g(x) + x^{32} I_b(x) \bmod g(x) \tag{3}$$

2.2.2. *Element of table look-up method between 32-bit data segments.* Information codes $M(x)$ are divided into segments of 32 bits. Each segment is marked by $D_j(x)$, $j+1$ segments in total:

$$M(x) = D_0(x) + D_1(x) + \dots + D_j(x)$$

Let $M_i(x)$ to be the current information code:

$$M_0(x) = D_0(x); \\ M_1(x) = x^{32}M_0(x) + D_1(x); \\ \dots \\ M_{n+1}(x) = x^{32}M_n(x) + D_{n+1}(x);$$

$R_n(x)$ is the residue of the calculation for the n time, we have:

$$R_n(x) = x^{32} M_n(x) \bmod g(x)$$

Then $R_{n+1}(x)$ is the residue for the $n+1$ time:

$$\begin{aligned} R_{n+1}(x) &= x^{32} M_{n+1}(x) \bmod g(x) = x^{32} x^{32} M_n(x) \bmod g(x) + x^{32} D_{n+1}(x) \bmod g(x) \\ &= x^{32} R_n(x) \bmod g(x) + x^{32} D_{n+1}(x) \bmod g(x) = x^{32} [R_n(x) + D_{n+1}(x)] \bmod g(x) \end{aligned} \quad (4)$$

2.2.3. algorithm description.

- (1) Initialize CRC as $R=0$;
- (2) Read current information D , calculate: $I= R+D$;
- (3) Respectively use high 16 bits of I and low 16 bits of I as the index to look up the table, result: R_a and R_b ;
- (4) Calculate the new CRC: $R = R_a + R_b$;
- (5) If there's information to be process, go to 2)
- (6) CRC calculation completed, output R .

3. experimental result

Algorithms in this paper are realized under S3C4510B, 50MHz clock. Code calculating time is 100 times in average, showing in Table 1.

Results show that the speed of 8-bit data segment algorithm 24 times of the speed of by-bit algorithm; 16-bit data segment 42 times, 32-bit data segment 68 times. 8-bit data segment algorithm needs 28×4 BYTE(1K BYTE) storage, 16-bit data segment algorithm 216×4 BYTE(256K BYTE), 32-bit data segment algorithm $2 \times 216 \times 4$ BYTE(512K BYTE). Based on the contrast above, we can conclude the 32-bit data segment algorithm is the optimized algorithm on the balance of speed and storage.

Table 1. results of 8-bit data segment, 16-bit data segment, 32-bit data segment algorithm.

		By bit		8-bit data segment	
bytes	cycles	time (μ s)	cycles	time (μ s)	
8	3466	69.32	134	2.68	
24	9642	192.84	392	7.84	
40	15810	316.20	646	12.92	
80	31282	625.64	1286	25.72	
152	59089	1181.78	2439	48.78	
200	77626	1552.52	3205	64.10	
224	86892	1737.84	3589	71.78	
240	93055	1861.10	3845	76.90	
248	96140	1922.80	3973	79.46	
		16-bit data segment		32-bit data segment	
bytes	cycles	time (μ s)	cycles	time (μ s)	
8	74	1.48	47	0.94	
24	210	4.20	127	2.54	
40	346	6.92	207	4.14	
80	706	14.12	427	8.54	
152	1394	27.88	843	16.86	
200	1962	39.24	1227	24.54	
224	2273	45.46	1472	29.44	
240	2385	47.70	1516	30.32	
248	2454	49.08	1557	31.14	

4. Conclusion

In order to ensure the veracity and speed of the transmission between measuring field and data processing system, to solve the contradiction between error detection, transmission rate and system resources in data transmission of ultra-precision measurement, this paper put forward a kind of high speed CRC-32 generating algorithm and its optimized algorithm. The algorithms are triumphantly realized under high speed ARM embedded system through coding experiments. Results show that this method has features of high error detecting ability, high speed and saving system resources, which improve real-time performance and reliability of the measurement data communication.

References

- [1] Zhang Hailin, Ge Sibao and Shi Ren 2003 New Cyclic Redundancy Code Check Algorithm *Journal of Xi'an Jiaotong University* **10** 1056-58
- [2] Shi Zhongzhuo and Wu Wenqi 2005 Software Method of CRC in Safety Data Transmission in Signal System of Urban Mass Transit *China Railway Science* **2** 113-117
- [3] Chen Jinping, Wang Shengze and Wu Wenying 2004 LabVIEW Based CRC Error Code Detection and Correction *Process Automation Instrumentation* **5** 74-76
- [4] Lin Hua and Wang Wei 2004 The Theory of CRC Algorithm and its Application in MPEG-2 System Layer *Computer Engineering and Applications* **8** 110-113
- [5] Baicheva T, Dodunekov S and Kazakov P 1998 On the Cyclic Redundancy-Check Codes with 8-bit Redundancy *Computer Commun* **21** 1030-33
- [6] Castagno li G, Ganz J and Graber P 1990 Optimum Cyclic Redundancy-Check Codes with 16-bit Redundancy *IEEE Trans Commun* **38** 111-114
- [7] Ramabadran, T. V. and Gaitonde, S. S. 1988 A tutorial on CRC computations *IEEE Micro* **8** 62-75