Software implementation research of CRC computation algorithms compatible with PKZIP, WINRAR, ETHERNET

E. Mytsko,
A. Malchukov,
V. Kim,
A. Osokin,
I. Zoev,
S. Ryzova
Department of Computer Engineering
National Research Tomsk Polytechnic University
Tomsk, Russia
evgenvt@tpu.ru, 1man@tpu.ru

***Abstract*—The paper describes software implementation research of CRC computation algorithms. Table-driven and matrix-driven algorithms were presented schematically. Also different implementations of the matrix-driven algorithm such as single-byte; two-byte and four-byte were researched. Graphical results of a computer experiment on supercomputer cluster to determine the speed of CRC32 software implementation were described. It is shown that a high-speed four-byte matrix-driven algorithm should be used in embedded systems and industrial data transmission systems. Research of the matrix-driven algorithms acceleration of relative table-driven shows that even two-bytes matrix-driven algorithm ahead of ~29%, while the four-bytes – by ~54%, which is a significant increasing in speed with respect to the table-driven algorithm.**

Keywords—Check sum, cyclic redundancy code, table-driven algorithm, matrix-driven algorithm, software implementation, polynomial.

I. INTRODUCTION

There are various algorithms for computing the CRC checksum and methods for their implementation. Classic algorithm involves bit-checksum computation by dividing the polynomial which represents the data on the generator polynomial which is used to compute the CRC in a variety of data transfer protocols (Ethernet, ZigBee) and archivers (Pkzip, WinRAR). There is also a table-driven algorithm [1], which accelerates the checksum computation using byte offset. Next we will focus on software implementations of algorithms for CRC32 computation which is used in Pkzip, WinRAR, Ethernet [2–5].

II. CLASSIC IMPLEMENTATION OF CRC

The basis for this algorithm is polynomial arithmetic [6]. Classic implementation of the checksum computation is a bit-wise division of the polynomial which represents the data (file) by the generator polynomial. The generator polynomial is computed depending on the scope of the algorithm. In this case it will be polynomial which is used in Pkzip, WinRAR, Ethernet.

Classic algorithm is implemented by successive iteration in the data shift register with feedback on one bit [1]. The result of this operation in the register is a CRC checksum (Fig. 1).

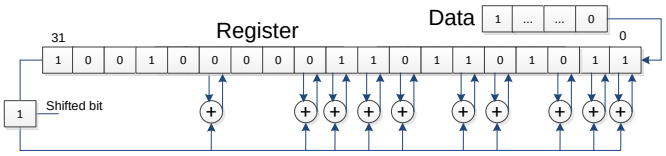

Fig. 1. Schematic representation of CRC32 classic algorithm.

The main drawback of this algorithm is that at any one time it is evaluated for one data bit which greatly slows down its operation. Therefore, byte computation is used to speed up the algorithms.

III. TABLE-DRIVEN ALGORITHM

A feature of the table-driven algorithm is that data is read byte by byte, which speeds up the CRC computation. The table with precomputed values by generator polynomial is used when checksum is computed. The generator polynomial 104C11DB7h [7, 8] which is submitted in hexadecimal is used in algorithms Pkzip, WinRAR, Ethernet. Fig. 2 schematically shows table-driven algorithm. Byte from data is summed by modulo 2 with byte from register with initial value 0xFFFFFFFF. The result is used for table index to get polynomial by address. The polynomial from table is summed by modulo 2 with bytes from register and result stores in the register. The process repeats iteratively until all the data bytes are not processed. Checksum will be stored in the register at the end of transactions.

The work was performed under the Federal Program «Research and development in priority areas of science and technology Complex of Russia in 2014 - 2020 years» state contract № 14.578.21.0032

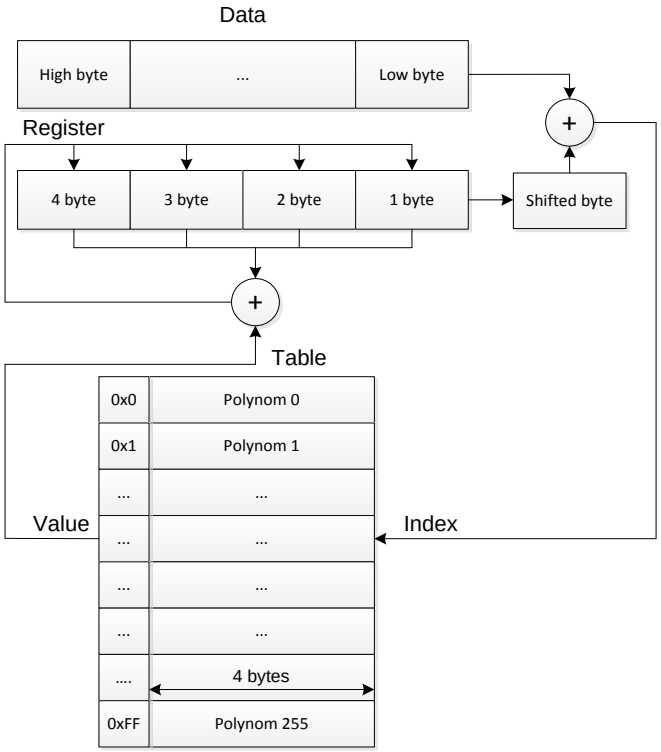

Fig. 2. Schematically representation of table-driven algorithm.

IV. MATRIX-DRIVEN ALGORITHM

Process of CRC32 checksum computation in matrix-driven algorithm is produced in the same way as in the table, except that instead of using a table, multiplication operation (extended byte) on matrix by modulo 2 is used [9] (Fig. 3, 4).

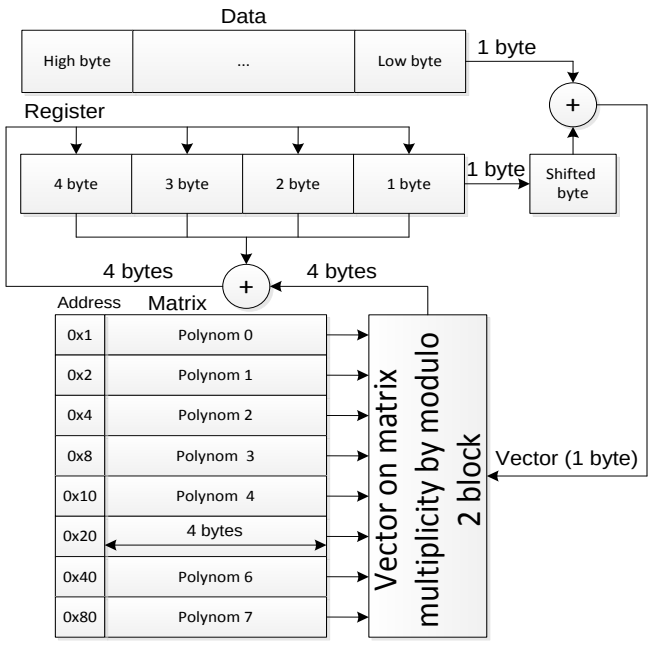

Fig. 3. Matrix-driven algorithm with 1 byte shift.

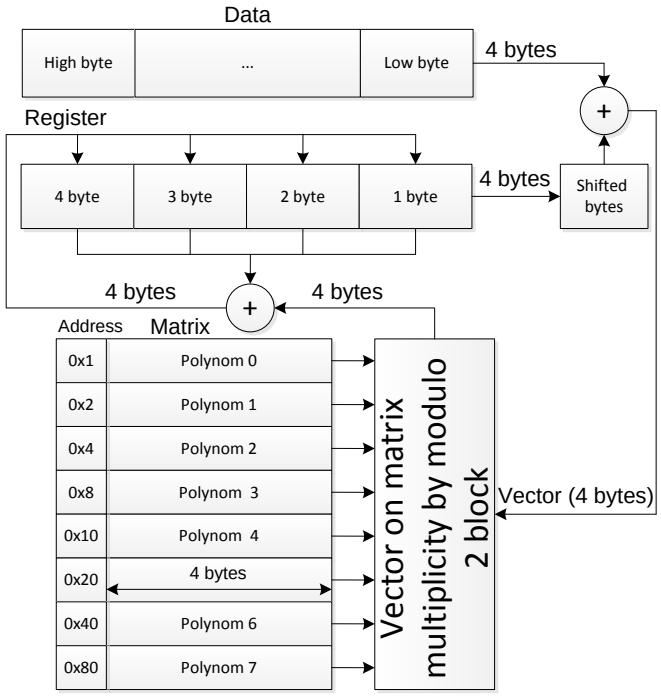

Fig. 4. Matrix-driven algorithm with 4 byte shift.

Software implementation of matrix-driven algorithm (one-byte offset) requires memory size is 32 bytes (for array storage).

Matrix-driven algorithm can be accelerated up if 2 or 4 bytes offset is used instead of one-byte offset. Shift by 3 bytes of this case is not considered, since for a software implementation data type size of 3 bytes is not available, which causes difficulties and loss of speed in the implementation of the algorithm.

As the number of bytes processed per iteration, the size of the matrix which is used in the computations will increase. For double-byte shift matrix will expand to 64 bytes; for a four-byte shift – up to 128 bytes.

V. COMPUTER EXPERIMENT

A. *Supercomputer description*

The supercomputer cluster «SKIF-polytech» is used to put a computer simulation to determine the performance of different software implementations of CRC32 algorithms. Table 1 shows the configuration of the supercomputer cluster «SKIF-polytech» [10].

TABLE I. «SKIF-POLYTECH» SUPERCOMPUTER CLUSTER CONFIGURATION

Hardware configuration	**Software**
Nodes: 24 Processors: 48 (Intel XEON 5150) Cores: 96 (2.66Ghz) Full RAM: 192ГБ Full HDD: 2880ГБ Data storage: 5ТБ	Novell SLES 10 Microsoft Windows HPC Server 2008 Including complete OS: Software translation source parallel programs for the

Hardware configuration	Software
System network: Infiniband 4x, 24 ports Additional network: Gigabit Ethernet, 48 ports Service network: ServNet, 25 ports Performance: 1.02TFlops Nodes: 24 Processors: 48 (Intel XEON 5150) Cores: 96 (2.66Ghz) Full RAM: 192ГБ Full HDD: 2880ГБ	languages C, C and FORTRAN ++ Unified system for user authentication Support for simultaneous execution of commands on all or selected cluster nodes

For getting checksum computation time on a particular file for specific software implementation 50 program executes were made for the data file with size of 10 up to 1010 megabytes with 200 megabytes step. Based on data for 50 runs of each program, average values with confidence intervals were illustrated in Fig. 5 – 14.

B. Computer experiment with 1 MB buffer size

The first stage was conducted research of software implementation for computing CRC32 with buffer 1MB which has been described in source code [11].

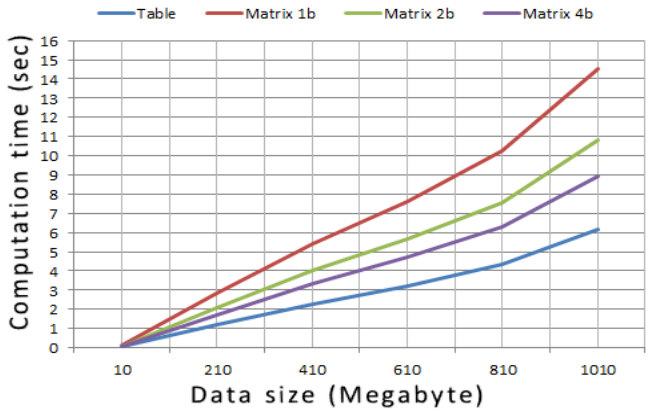

Fig. 5. Average time values for CRC32 computation with 1 MB buffer.

Using a supercomputer cluster helps to save time for the experiment and get time estimates with sufficiently high reliability (Fig. 5). The fig. 6 shows confident intervals for average time computation.

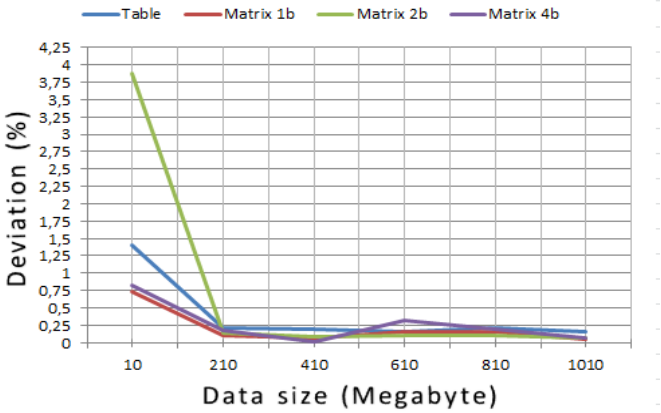

Fig. 6. Confident intervals for average time computation.

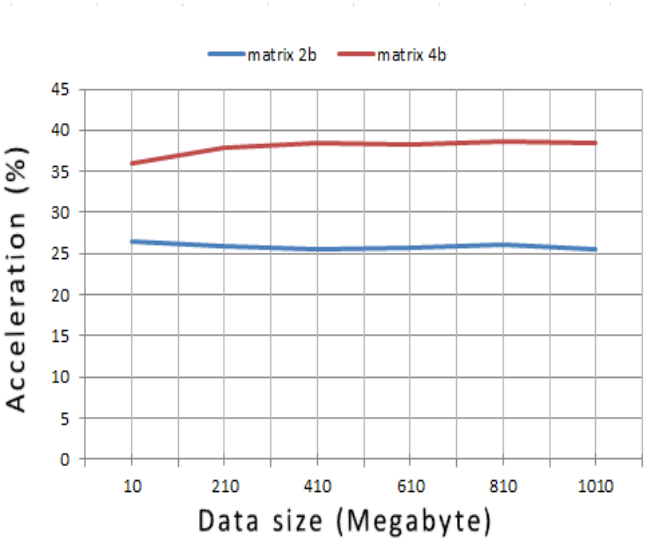

Fig. 7. Acceleration relative one-byte shift with 1MB buffer.

The difference for the data in Fig. 5 matrix-driven algorithms for 2- and 4- bytes offset from the one-byte (Fig. 7) was calculated to determine the effect of increasing bytes number which are processed per iteration.

Matrix-driven algorithm with 4-bytes shift computes a checksum for ~38 % faster than matrix-driven algorithm with single-byte shift, as can be seen from Fig. 7.

However, even the 4-bytes matrix-driven algorithm lags by speed from table-driven algorithm on ~44 % (Fig. 8).

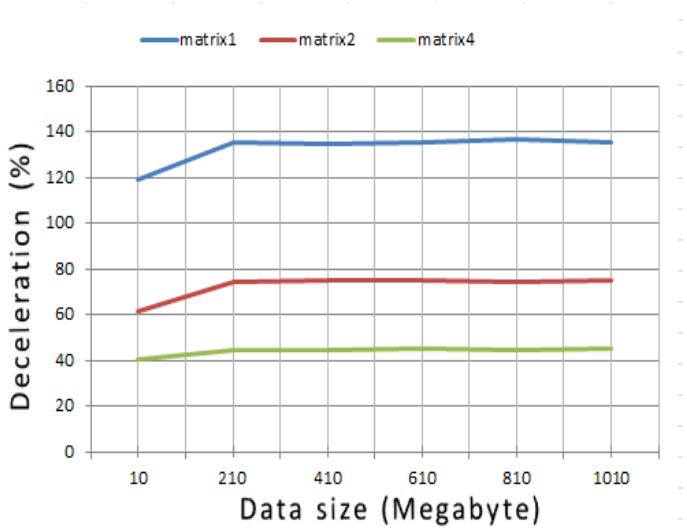

Fig. 8. Deceleration of matrix-driven algorithm relative table.

Due to the increased number of bytes which is processed per iteration to 4, in matrix-driven algorithm relative speed of table-driven algorithm is reduced to ~ 44 %, but it requires 8 times less memory (128 bytes) for storage matrix than to implement the table-driven algorithm (1024 bytes).

C. Computer experiment with 8 bytes buffer size

The buffer size was decreased to 8 bytes for microcontroller implementation, which has a limit of memory. For 8 byte buffer size results are shown in Fig. 9.

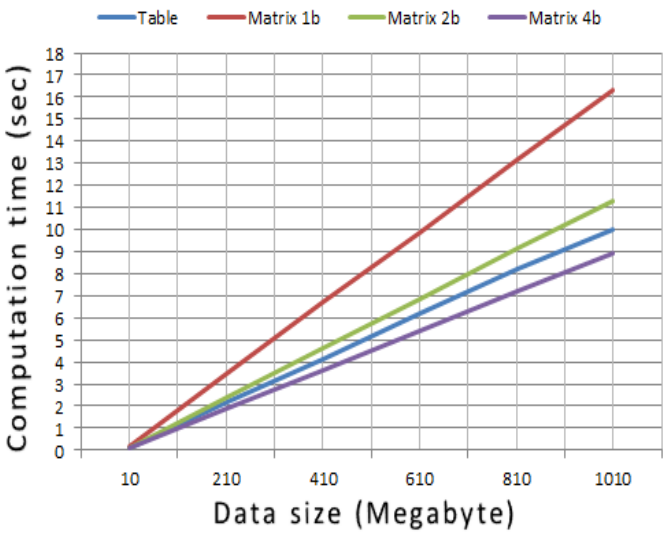

Fig. 9. Average time values for CRC32 computation with 8 bytes buffer.

As can be seen from Fig. 10 and 11 reducing the size of the buffer to 8 bytes changed pattern which was obtained previously.

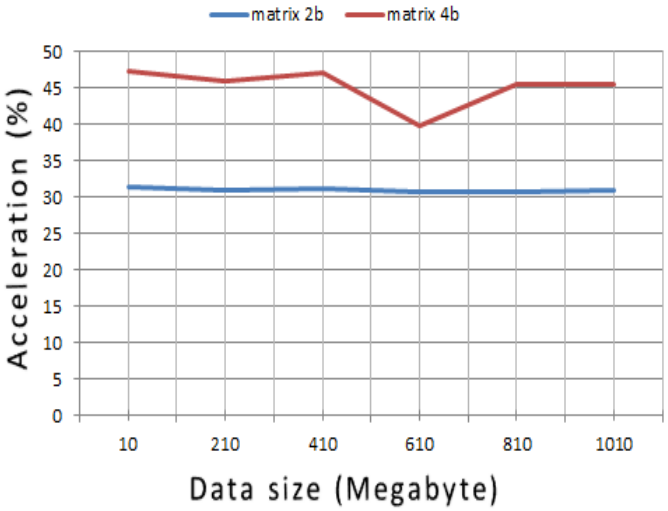

Fig. 10. Acceleration relative one-byte shift with 8 bytes buffer.

For a two-bytes matrix-driven algorithm acceleration increased relative single-byte to an average of ~30%, while for the four-bytes algorithm – to ~45%. Also fig. 11 shows that the single-byte and two-bytes algorithms reduced the lagging by the speed computation to ~61% and ~11% respectively; while four-bytes matrix-driven algorithm has been speed up in average by ~12 % relative to the table-driven.

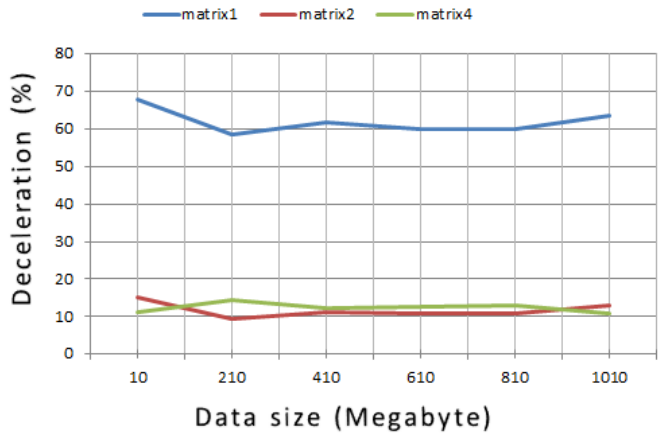

Fig. 11. Deceleraton of matrix-driven algorithm relative table with 8 bytes buffer.

D. Computer experiment with 1 byte buffer size

The buffer size was decreased to 1 byte for microcontroller. The results are presented on the fig. 12 – 14.

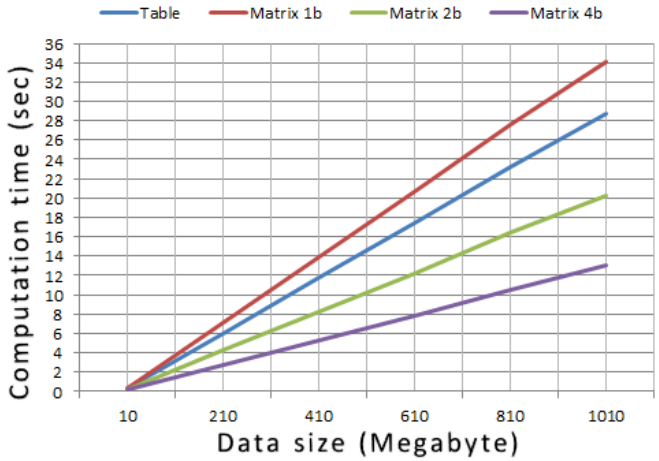

Fig. 12. Average time values for CRC32 computation with 1 byte buffer.

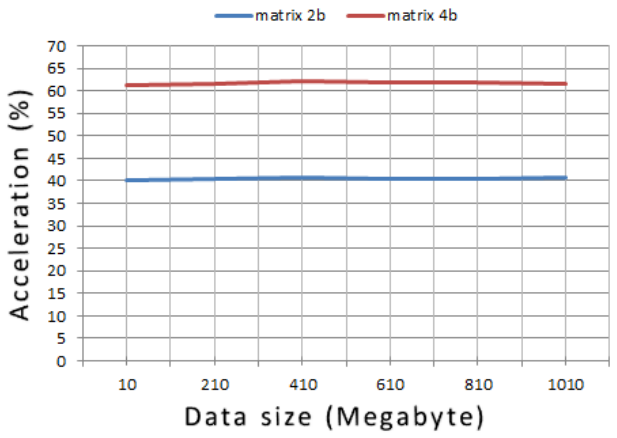

Fig. 13. Acceleration relative one-byte shift with 1 byte buffer.

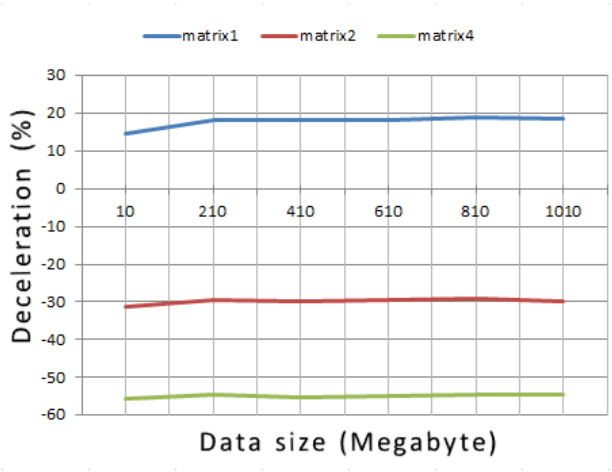

Fig. 14. Deceleration of matrix-driven algorithm relative table with 1 byte buffer.

It was found that the acceleration of matrix-driven algorithms relative single-byte matrix-driven algorithm increased to 40.52 and ~61% respectively. Research of the matrix-driven algorithms deceleration of relative table-driven shows that even two-bytes matrix-driven algorithm ahead of ~29%, while the four-bytes – by ~54%, which is a significant increasing in speed with respect to the table-driven algorithm.

The results obtained are characterized by the fact that reducing the buffer to 1 byte has loss computations rate for the table-driven algorithm is much more significant than for the matrix-driven algorithms.

VI. Conclusion

The paper describes the results of the performance research for different CRC32 checksum implementations. Computer experiment which was delivered to determine the software implementation speed of the CRC32 algorithm showed that the matrix-driven algorithm suitable for using in microcontrollers, where the available memory is less than 8 bytes. The table-driven algorithm should be used in systems with more available memory, as implementations for microcontrollers performance of this algorithm decreases significantly.

As a result, it was found that the fastest of the matrix-driven algorithms – four-bytes should be used in embedded systems, industrial data transmission systems that use microcontrollers. Recommendations of matrix-driven algorithms using due to the ease of implementation and significant savings of memory required for the microcontroller's control program and also due to the good speed performance compared to the table-driven algorithm.

References

[1] A Painless Guide to CRC Error Detection Algorithms. Available: http://www.ross.net /crc/download/crc_v3.txt, accessed December. 2015.

[2] H. Payal, K. Mankar, "A Paper on Parallel CRC Generation for High Speed Application", International Journal of Innovative Research in Advanced Engineering (IJIRAE). Issue 1, Vol. 2, pp. 254 – 256, Janurary. 2015.

[3] S. Cherian, N. George, S. Enosh, S. Pillai, "An Efficient Way of Generating CRC bit for Serial data using any polynomial", International Journal of Scientific and Research Publications. Vol. 4, Issue 4, pp. 1 – 3, 2014.

[4] A. Tanenbaum, D. Wetherall, Computer Networks. Boston : Pearson, 2011.

[5] P. Koopman, T. Chakravarty, "Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks", The International Conference on Dependable Systems and Networks. Palazzo dei Congressi, Florence, June 28 - July 1, 2004, pp. 1-10.

[6] P. Koopman, "32-Bit Cyclic Redundancy Codes for Internet Applications", Intern. Conf. on Dependable Systems and Networks (DSN), Washington, July 2002, pp. 459–468.

[7] Mytsko Evgeniy, Malchukov Andrey, "Adaptation of technology MPI and OpenMP to search for the generators polynomials", Proceedings of the 9th International Forum on Strategic Technology (IFOST-2014). Chittagong, October 21-23, 2014, pp. 5-8.

[8] Mytsko Evgeniy, Malchukov Andrey, "Application of parallel computing technology openmp to search for the generator polynomials", Mechanical Engineering, Automation and Control Systems: Proceedings of International Conference. Tomsk, October 16-18, 2014, p. 1-5.

[9] A. Malchukov, A. Osokin, Y. Bourkatovskaya, "Algorithms of accelerated division on modulo 2", Proceedings of 7th Korea-Russia International Symposium on Science and Technology, Korus 2003, Vol. 2, 2003, pp. 189-192.

[10] CKP SKK "SKIF-Politeh", Available: http://cluster.tpu.ru/, accessed March 2016.

[11] 32 bit Cyclic Redundancy Check Source Code for C++, Create Window Website, Available: http://www.pudn.com/downloads132/ sourcecode/windows/other/detail561362.html, accessed March 2016.