International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Design and Implementation of Parallel CRC for High Speed Application

Shreya Gawande[1], Dr. S. A. Ladhake[2]

[1]Electronics and Telecommunication, SIPNA C.O.E.T./SGBAU University, Maharashtra, India

[2]Principal, SIPNA C.O.E.T/SGBAU University, Maharashtra, India

Abstract: ***High speed data transmission is the current scenario in networking environment. Cyclic Redundancy Check (CRC) is essential method for detecting error when the data is transmitted. With the challenging speed of transmitted data, to synchronize with speed, it is necessary to increase the speed of CRC generation. With the help of serial architecture a recursive formula is used from which a parallel design is derived which reconfigure rapidly to new polynomials. Usually the hardware implementation of linear feedback shift register used in serial crc generation does not achieve a high throughput, but is possible in parallel generation. The hardware scheme for computing the transition matrix of parallel cyclic redundancy checksum is used. This improves the polynomial adaptability. The factor of an area, required for a realization for storing the pre-computed matrix is also considered. The design lowers the area requirement. The equations allow the width of data to be processed in parallel and is selected independent of the degree of polynomial. The new design performs significantly in speed, area and energy efficiency. The derivation of the matrix necessary to set up all latches can be performed in software. Architecture uses the minimum number of clock cycles which increases the efficiency of the transmission process and easily adaptable to variation in transmission data bytes. The design architecture for CRC generation will be functionally verified using ModelSim and synthesized on Altera FPGA using Quartus 2 software.***

Keywords: crc, matrix, generator, polynomial

1. Introduction

Cyclic redundancy check is a popular error detecting code computed through binary polynomial division. CRC is used in data communication and various fields such as data storage, data compression dealing with data errors. Many times speed requirement makes software schemes impractical and demands a dedicated hardware. CRC is a very powerful and easily implemented technique to obtain data reliability.
CRC can generate in 2 ways:
1) Serial CRC generation
2) Parallel CRC generation

Usually hardware implementation is based on the linear feedback shift register (LFSR), which process data in serial way. But we cannot achieve a high throughput by serial calculation of CRC code. Instead of it parallel CRC generation overcomes this problem. The parallel method process whole data words on cascading the LFSR

2. Background Work

The "**A Novel Approach for Parallel CRC generation for high speed application"** is presented by Hitesh H. Mathukiya; Naresh M. Patel [1]. This article addresses 64 bits parallel CRC architecture based on F matrix with order of generator polynomial is 32. Proposed design is hardware efficient and required 50% less cycles to generate CRC with same order of generator polynomial. The whole design is functionally verified using Xilinx ISE Simulator**.** In this paper, the proposed architecture deal with 64bit parallel processing based on built in F matrix generation; this gives CRC with half number of cycles. This paper starts with the introduction of serial CRC generation based on LFSR. F matrix based parallel architecture for 32 bits and 64 bits are described.

The paper **"VLSI Implementation of Parallel CRC Using Pipelining, Unfolding and Retiming"** is presented by Sangeeta Singh, S. Sujana, I. Babu, K. Latha [3]. This article demonstrate the implementation of parallel Cyclic Redundancy Check (CRC) based upon DSP algorithms of pipelining, retiming and unfolding. The architectures are first pipelined to reduce the iteration bound by using novel look-ahead techniques and then unfolded and retimed to design high speed parallel circuits. This article gives the comparison between the parallel implementation of CRC-9 and its serial implementation. It also shows that parallel implementation uses less number of clock cycles than the serial implementation of CRC-9 thereby increasing the speed of the architecture. The work is carried out and implemented using Verilog hardware description language simulated using Xilinx ISE tools and synthesized using Cadence tools. The proposed design achieves shorter critical path for parallel CRC circuits leading to high processing speed than commonly used generator polynomial. The proposed design starts from LFSR, which is generally used for serial CRC. An unfolding algorithm is used to realize parallel processing.

However, direct application of unfolding may lead to parallel CRC circuit with long iteration bound, which is the lowest achievable CP. Two novel look-ahead pipelining methods are developed to reduce the iteration bound of the original serial LFSR CRC structures; then, the unfolding algorithm is applied to obtain a parallel CRC structure with low iteration bound. The retiming algorithm is then applied to obtain the achievable lowest CP.

A new hardware scheme for computing the transition and control matrix of a parallel cyclic redundancy checksum is proposed by Martin Grymel and Steve B. Furber [5] in the paper **"A Novel Programmable Parallel CRC Circuit".** This design opens the possibilities for parallel high-speed cyclic redundancy checksum circuits that reconfigure very rapidly to new polynomials. The area requirements are lower than those for a realization storing a pre computed matrix.

An additional simplification arises as only the polynomial needs to be supplied. The derived equations allow the width of the data to be processed in parallel to be selected independently of the degree of the polynomial. The new design has been simulated and outperforms a recently proposed architecture significantly in speed, area, and energy efficiency**.** The derivation of the matrix necessary to set up all the latches can be performed in software. As the data bus width imposes a limitation on the transferable data, the matrix may need to be communicated line by line, which requires clock cycles. Additionally, the software function itself relies on a processor. Many scenarios may even imply a dedicated core for this task, if the polynomial needs to be changed frequently and faster than the matrix can be communicated over the data bus.

"Parallel Computation of CRC using Special Generator Polynomials" [7] paper is presented by Hamed Sheidaeian and Behrouz Zolfaghari. This paper studies a case for parallel CRC computation using special generators which have special multiples called OZO (One-Zero-One) polynomials are divisible. We first provide a systematic approach to finding such polynomials and then design and evaluate the algorithm and the hardware required to perform the parallel division. This paper proposes a novel method for parallel computation of CRC using mathematical properties of a special category of generator polynomials called ODPs (OZO Dividing Polynomials). ODPs are polynomials having multiples of form 100…001. The latter form of polynomials is called OZO (One-Zero-One). We demonstrate that if the generator is selected from this category, the CRC can be calculated by parallel circuits with minor hardware requirements. Zolfaghari introduced OZO and ODPs. They developed a systematic method for constructing ODP polynomials. The main idea behind our proposed approach is performing a number of operations (each of which takes a cycle in the traditional modulo-2 division circuits) in a single cycle. We use the properties of OZO strings to achieve this goal

3. System Design

In the proposed design data is parallel processed. Here the concept of F matrix is used. F matrix is constructed using generator polynomial with the help of specific algorithm. Because of this the checksum which will be calculated can be used to variable number of data bits. The data is ANDed with the F-matrix generated from the generator polynomial. Result of that will XORed with present state CRC checksum. The new formed checksum will be used for the new data input. This process will be repeated as shown in the figure.

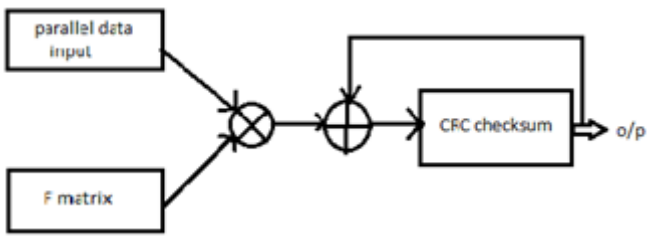

Figure : Algorithm based on F matrix

In order to accommodate requirements, a reconfigurable parallel CRC unit is desired that can easily adapt to new polynomials without slowing down the data communication. A polynomial directly affects the transition and control matrix of system. Scheme is presented allowing the inexpensive computation of the CRC transition and control matrix independent of the degree of polynomial. This leads to transformation of parallel CRC architecture into a programmable entity that is no longer bound to a specific CRC polynomial. Programmability can be achieved by introducing an AND gate with a controlling latch for each signal that may be a potential input to an XOR function. Flip flops can be utilized as well, but will have in general higher demands in terms of area, which may become crucial as bits need to be stored.

In proposed architecture if 64 bits are parallely processed and order of generator polynomial is 32 then CRC-32 will be generated after some cycles. If the new data is entered, the equations based on this algorithm can be regarded as a recursive calculation of the next state by F matrix , current state and parallel input data, make the 32-bit parallel input vector suitable for any length of messages besides the multiple of 32 bits. But the length of the message should be byte based. If the length of message is not the multiple of 32, after a sequence of 32-bit parallel calculation, the final remaining number of bits of the message could be 8, 16, or 24. For all these situations, an additional parallel calculation for bits 8, 16, 24 is needed by choosing the corresponding F matrix. Since F can be easily derived from 32 bits F matrix, the calculation can be performed using the same circuit as 32-bit parallel calculation.

4. Conclusion

The proposed design requires less cycles which reduces the time and increases the throughput. Also pre-calculation of F matrix is not required at each time when the data is entered.

5. Application

The CRC can be applied to,

1) Data storage devices, such as a disk drive to check bits in each block.
2) In the both Transmitter and Receiver block in order to detect error in digital data during high-speed transmission.

References

[1] Hitesh H. Mathukiya; Naresh M. Patel, "A Novel Approach for Parallel CRC generation for high speed application", 2012 International Conference on Communication Systems and Network Technologies.
[2] Yan Sun; Min Sik Kim; , "A Pipelined CRC Calculation Using Lookup Tables," Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE , vol., no., pp.1-2, 9-12 Jan.2010
[3] Sangeeta Singh, S. Sujana, I. Babu, K. Latha. "VLSI Implementation of Parallel CRC Using Pipelining, Unfolding and Retiming", IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) Volume 2, Issue 5 (May. – Jun. 2013)
[4] C.Toal, K. McLaughlin, S. Sezer, and X.Yang, "Design and implementation of a field programmable CRC circuit architecture," IEEE Trans. Very Large Scale

Integr. (VLSI) Syst., vol. 17, no. 8, pp. 1142–1147, Aug. 2009.

[5] Martin Grymel and Steve B. Furber, " A Novel Programmable Parallel CRC Circuit", IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS,2010 IEEE

[6] High speed parallel CRC based on unfolding, pipelining, retiming IEEE transactions on circuits and systems—ii: express briefs, vol. 53, no. 10, October 2006

[7] "Parallel Computation of CRC using Special Generator Polynomials" paper is presented by Hamed Sheidaeian and Behrouz Zolfaghari.

[8] Weidong Lu and Stephan Wong, "A Fast CRC Update Implementation", IEEE Workshop on High Performance Switching and Routing ,pp. 113-120, Oct. 2003.

[9] S.R. Ruckmani, P. Anbalagan, " High Speed cyclic Redundancy Check for USB" Reasearch Scholar, Department of Electrical Engineering, Coimbatore Institute of Technology, Coimbatore- 641014, DSP Journal, Volume 6, Issue 1, September, 2006.

[10] "A practical parallel CRC generation method" by Evgeni stavinov.