Nibble Mode Realization of Parallel CRC

Sulipta Das
Department of ECE
NIST, Berhampur, Odisha,
India-761008

Devashree Mahato
Department of ECE
NIST, Berhampur, Odisha,
India-761008

Durga Prasad Dash
Assistant Professor
Department of ECE
NIST, Berhampur, Odisha,
India-761008

ABSTRACT

This paper presents a parallel Cyclic Redundancy Check (CRC) architecture for IEEE 802.3u Medium Access Control (MAC) transmitter using data flow modelling. The purpose of the design is to improve the processing speed of data frames over fast ethernet Local Area Network (LAN). The input to the designed circuit is in nibble format. Synthesis options are explored along with Verilog Hardware Description Language (HDL) styles to have a balance among area and speed optimization in Xilinx ISE 12.1 tool. It is verified that the processing of MAC frames is faster in case of parallel CRC than the serial one. We investigate the inverse relationship between resource utilization and frequency of operation. A Critical Path Delay (CPD) of 3.801ns is achieved for parallel CRC implementation on Spartan3E XC3S100E-4-VQ100 Field Programmable Gate Array (FPGA). The behaviour of the design is analysed on target devices of Virtex6, Spartan6 and Virtex5 families of FPGA. It is found that Virtex6 XC6VLX75T-3-FF484 (40 nm technology) device offers the highest operating frequency of 1203.804 MHz allocating 32 Look Up Tables (LUTs) and 32 slices.

Keywords

Ethernet, MAC frame, Nibble, Parallel CRC-32, Modulo-2 arithmetic, Data flow modeling, LFSR, LUTs.

1. INTRODUCTION

The physical channel for communication system is unreliable due to channel impairments. Noise and interference are the two primitive causes that immensely degrade the fidelity of the transmission media. As a result, the information is subjected to the risk of unpredictable and unwanted changes giving rise to either single bit error or burst error or combination of both. So, for a reliable system there must be a provision of error detection and correction [1],[2].

Cyclic Redundancy Code (CRC) is one of the most widely used error detection technique in serial data communication systems and data storage devices to ensure the correctness of the received and stored data respectively [3]. Especially, this method has an attractive feature of identifying the burst errors along with single bit error detection. CRC is the remainder obtained from the binary division of the message polynomial by the CRC generator polynomial [1]-[5]. Here, concatenation of evaluated CRC to the end of the data unit results in a dividend which is exactly divisible by the predetermined divisor. At destination, the process of division by the same divisor over the resultant data containing redundant bits is done to check the remainder. Presence of remainder notifies that data is corrupted and hence, must be rejected [1], [2]. Some of the commonly utilized CRC standards are CRC-8, CRC-12, CRC-16, CRC-CCITT and CRC-32 suitable for specific application areas [5]. Amongst these standards, CRC-32 is commonly applicable to ethernet local area networks. Fast ethernet MAC transmitter supports nibble mode data transfer and therefore requires a parallel CRC architecture capable of processing four bits simultaneously. Nibble-wise CRC calculation cannot be accomplished by the conventional serial implementation.

This paper presents a simplified and compact parallel CRC-32 implementation to have compatibility with IEEE 802.3u MAC transmitter. The performance of the design is compared with serial LFSR architecture on the basis of device utilization and processing speed. In the current scenario, the demand for the high speed circuit is growing at a faster rate as per the digital communication is concerned. Research regarding the implementation of computer network protocols concentrates on FPGA area and speed optimization for enhancement of network performance. The new scheme proposed by us follows effective Verilog HDL coding style, selection of appropriate target device and application of specific synthesis constraints to the target technology for optimization of CPD in FPGA as well as resource allocation.

The rest of this paper is structured as follows. Section 2 gives a brief overview of CRC-32 error detection technique used for IEEE 802.3 protocol and the LFSR implementation. In section 3, we derive the logic required for nibble wise parallel CRC computation in data flow modelling from the serial LFSR. Section 4 includes the synthesis observations in addition to simulation results. In Section 5, an analytical study has been carried out to verify the behaviour of the designed parallel structure when implemented on different target devices in terms of speed and area utilization. Finally, we draw the conclusions from the findings in section 6.

2. Ethernet CRC-32

All The most dominant technology used in the LAN market is Ethernet, standardized by IEEE 802.3 committee. Stations connected to the LAN differ slightly in the construction of MAC sub-layer depending on the ethernet versions. Ethernet MAC frame consists of seven fields namely preamble, start of frame delimiter (SFD), destination address (DA), source address (SA), length of the protocol data unit, data field (46-1500 bytes) and CRC (4 bytes). The CRC value is computed on destination address, source address, length and data fields of MAC frame.

As defined by IEEE 802.3, Ethernet uses CRC-32 generator polynomial which is represented as [1]-[5];

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

Modulo-2 arithmetic operations performed in the Ethernet CRC generator follows the steps given below [5]:

Step 1: Message polynomial M(x) is multiplied by x^{32} to provide shifting operation.

Step 2: The shifted message M(x) is then divided by G(x) and the remainder polynomial R(x) obtained represents the desired CRC. Here the degree of $R(X) \leq 31$

Step 3: The codeword C(x) is formed by appending the evaluated CRC value in step 2 to the end of the original message M(x). Mathematically,

$$C(x) = [\{M(x) \times x^{32}\} + R(x)] \qquad (2)$$

As this codeword c(x) is absolutely divisible by generator polynomial G(x), it can be represented as;

$$C(x) = Q(x) \times G(x) \qquad (3)$$

where G(x) is the quotient polynomial in the binary division.

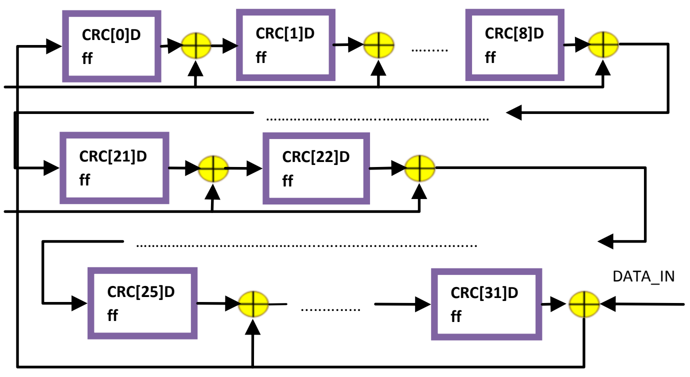

Figure 1: LFSR hardware implementation of serial CRC

The modulo-2 arithmetic process can be accomplished by a LFSR hardware implementation consisting of 32 flip-flops and 14 xor gates, as depicted in Figure 1. Here, the equivalent logic for multiplication and addition is implemented by shift registers and xor gates respectively.

The DATA_IN input port of the LFSR configuration as indicated in Figure 1, accepts the binary data from the MAC transmitter in the serial mode starting from the most significant bit (MSB) of the first octet of the DA field up to the least significant bit (LSB) of the last octet of the data field. Presetting 32 flip-flops to binary ones performs the complementing operation on the first 8 nibbles of the information sequence [5].

Each flip-flop output is updated at the end of every clock pulse after one step of CRC computation. When the last bit of the input message stream is processed, the sequential circuit holds the final checksum value which is to be complemented before concatenating to the end of data field.

The prime limitation of the given serial architecture is that, it is not compatible with higher version of the ethernet standard, supporting transmission of more than one bit per one clock cycle simultaneously [3]-[5]. As 100Mbps ethernet operates in nibble mode, means sending four bits at a time per clock pulse, a parallel implementation is desirable.

3. NIBBLE WISE CRC COMPUTATION

A set of equations governing the parallel CRC computation can be deduced from the basic LFSR hardware used for serial CRC. The first step towards extraction of nibble mode CRC logic involves a systematic and detail reckoning of entire set of flip-flop outputs for each serial input binary digit. The aforementioned procedure is repeated four times and each modified results obtained at the end of clock cycle are monitored. Each successive output is represented in terms of the previous checksums. A careful investigation of the pre-valuated outcomes is carried out to represent the modified LFSR output in terms of the previous one and four bits of information sequence. Thus the combinational logic for dataflow modelling is derived from the intermediate checksum values from the four numbers of iterative processes followed by a backward computation process.

The newly developed xor equations are coded in verilog HDL to realize the behaviour of the nibble mode parallel CRC circuitry to be utilized in the fast ethernet MAC transmitter. The uses of "for loop" statements are avoided in the module description rather "assign" statements are preferred to infer the logic with ease [6], [7]. As a consequence, the execution time is reduced significantly. To fully synchronize the sequential circuit, importance is given on performing the logic through D inputs instead of providing control along the clock path by insertion of variable gates [6]. Furthermore, a common clock pulse with of 25 MHz frequency is applied to all the flip-flops to trigger them simultaneously.

4. SYNTHESIS & SIMULATION RESULTS

We synthesized CRC-32 on Spartan3E XC3S100E-4-VQ100 for both cases; at first for serial architecture and then for the proposed parallel design. The synthesis result of former design is compared with that of the latter one.

Table 1. Comparison of serial and parallel architectures based on f_{max} and CPD

Optimization Goal / Processing	Speed		Area	
	f_{max}(MHz)	CPD(ns)	f_{max}(MHz)	CPD(ns)
Serial	94.768	10.552	94.162	10.66
Parallel	263.089	3.801	197.1224	5.073

Table 2. Comparison between serial and parallel CRC circuits considering area utilization of FPGA

Optimization Goal / Processing	Speed		Area	
	LUTs	Slices	LUTs	Slices
Serial	32	18	32	18
Parallel	49	26	40	23

Table 3. Comparison of f_{max} and device utilization for implementation on various target devices

Target Device	f_{max} (MHz)	CPD (ns)	LUTs	Slices
Spartan 3E XC3&100E-5-VQ100	305.064	3.278	49	26
Spartan 6 XC6SLX4-3-TQG144	489.261	2.044	32	32
Virtex 4 XC4VLX15-12-SF363	653.445	1.530	52	28
Virtex 5 XC5VLX30-3-FF324	911.826	1.097	32	32
Virtex 6 XC6VLX75T-3-FF484	1203.804	0.831	32	32

A brief analysis is made on the basis of critical path delay (CPD) considering both of the synthesis constraints: speed and area. The observations for maximum frequency and CPD are reported in Table 1. Further, we inspected the resource utilization in terms of required number of look up tables (LUTs) and slices for hardware realization as presented in Table 2.

We have compiled the Verilog codes for different target devices from Spartan 3E, Spartan 6, Virtex 4, Virtex 5 and Virtex 6 families of FPGA. The synthesis reports are investigated for performance evaluation of CRC-32 parallel module. For ethernet implementation we specifically concentrate on maximum frequency achieved. The observations are tabulated as illustrated in Table 3.

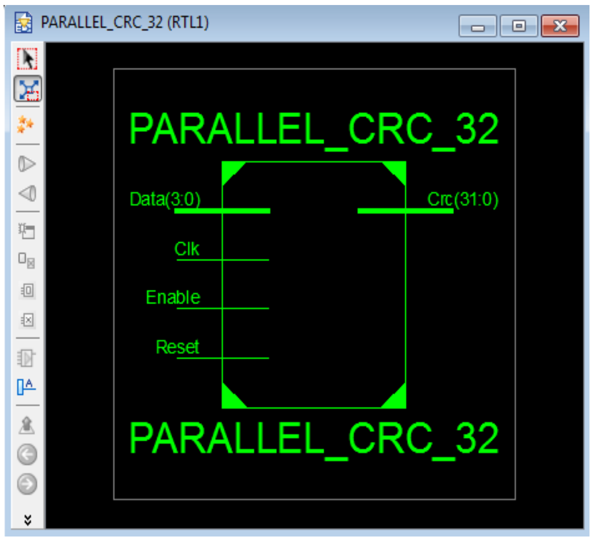

Figure 2: RTL schematic view of parallel CRC design

Figure 2 presents the RTL schematic view of the nibble mode parallel CRC-32 showing the input and output ports of the module.

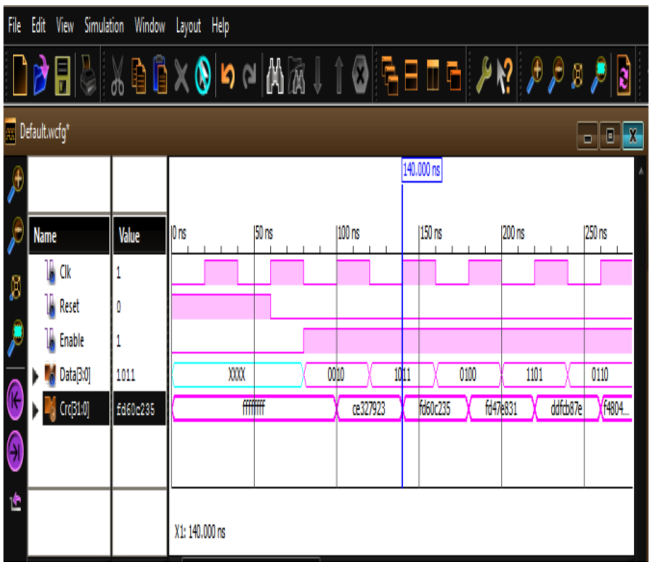

Figure 3: Simulation waveform PARALLEL_CRC_32 module

5. ANALYSIS

The graphical analysis is being carried out from the observations provided in the previous section. Here, we investigate the relationship between the components of FPGA utilized for implementation and the maximum frequency achieved.

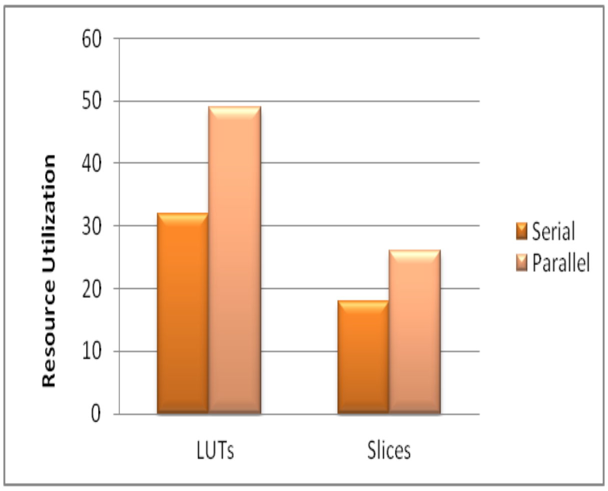

Figure 4: Comparison of resource allocation

Figure 4 depicts the pictorial representation about device utilization ratio for speed constraint in terms of LUTs and slices for both the designs. Here, we examined that in comparison with the serial LFSR, the parallel modeling requires approximately 35% and 31% more LUTs and slices respectively. In overall, we can state that the hardware savings is possible in case of serial design whereas the performance in terms of speed is better in the latter design with respect to the former one.

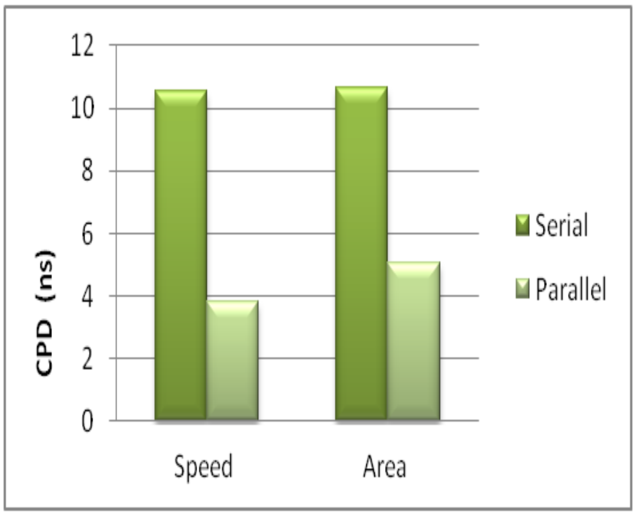

Figure 5: Comparison of CPD

From Table 1 and Figure 5 we estimated that the path delay is around 64% less for speed optimization and nearly 52% less for area criterion for parallel CRC. It indicates parallel model requires less synthesis time with respect to the serial structure.

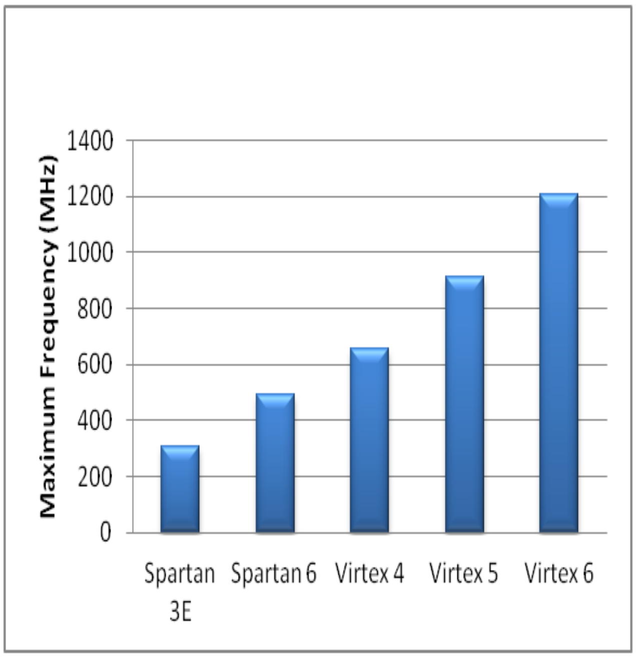

Figure 6: Comparison of maximum frequency

Again the same parallel CRC-32 proposed design for nibble mode operation is synthesized on different target technology of XILINX FPGAs. From Figure 6 we noted that the design achieved the optimized operating frequency for FPGA with process technology of 40 nm. Performance results for Spartan 6 and Virtex 5, based on process technology of 45 nm and 65 nm respectively are next to Virtex 6 family.

6. CONCLUSION

In this paper we propose an efficient way of Verilog HDL coding style and optimized method to realize nibble mode parallel CRC-32 encoder for the purpose of IEEE 802.3u MAC transmitter. We observed that frequency of operation is inversely proportional to the resource utilization. Functional simulation is verified for accuracy in the calculation of 4 bytes FCS to be appended to the end of the data field. The conclusions drawn are as follows;

1] Parallel CRC module exhibits relatively faster processing than the serial LFSR block.

2] The design construction is less complex and compact for dataflow modeling.

3] Virtex 6 target technology offers the maximum frequency of operation to achieve high speed circuits.

7. REFERENCES

[1] G. W. Stallings, *Data And Computer Communications,* Prentice Hall, 2009.

[2] Behrouz A. Forouzan, *Data Communications And Networking*, Tata McGraw-Hill, 2009.

[3] G. Campobello, G. Patane, M. Russo, "Parallel CRC Realization", *IEEE Transactions On Computers*, October 2003,Vol.52, No. 10, pp. 245-256.

[4] Tong-Bi Pei, Charles Zukowski, "High-speed Parallel CRC Circuits in VLSI", *IEEE Transactions On Communications*, April 1992,Vol.40, No. 4, pp. 456-467.

[5] Chris Borrelli, "IEEE 802.3 Cyclic Redundancy Check", *XAPP209 (v1.0)*, March 23, 2001.

[6] M. Morris Mano, *Digital design*, Prentice Hall, 2007.

[7] Samir Palnitkar, *Verilog HDL*, Pearson Educatin, 2008.