VLSI IMPLEMENTATION OF CRC-32 FOR 10 GIGABIT ETHERNET

Tomas Henriksson, Henrik Eriksson, Ulf Nordqvist, Per Larsson-Edefors, and Dake Liu

Department of Physics and Measurement Technology
Linköping University, SE-581 83 Linköping, Sweden
Phone: +46-13-28{8956, 2483, 8965, 1224, 1256}, Fax: +46-13-132285
E-mail: {tomhe, hener, ulfnor, perla, dake}@ifm.liu.se

ABSTRACT

For 10 Gigabit Ethernet a CRC-32 generation is essential and timing critical. Many efficient software algorithms have been proposed for CRC generation. In this work we use an algorithm based on the properties of Galois fields, which gives very efficient hardware. The CRC generator has been implemented and simulated in both standard cells and a full-custom design technique. In standard cells from the UMC 0.18 micron library a throughput of 8.7 Gb/s has been achieved. In the full-custom design for AMS 0.35 micron process we have achieved a throughput of 5.0 Gb/s. The conclusion, based on extrapolation of device characteristics, is that CRC-32 generation for 10 Gb/s can be designed with standard cells in a 0.15 micron process technology, or using full-custom design techniques in a 0.18 micron process technology.

1. INTRODUCTION

Digital communication is becoming more and more important and higher bit rates are constantly required. To manage this, not only the optical fibers have to work at higher speeds, but also the electronic equipment at the ends of these fibers. One thing that has been proven hard to speed up is the checksum calculation and especially the type cyclic redundancy check (CRC), which is used for example in the Ethernet and ATM protocols.

In the proposal for the new 10 Gb/s Ethernet standard it is specified that CRC will still be used and also that the minimum data unit is a symbol of 8 bits [1]. It is highly beneficial to perform the CRC calculation at wire speed, since buffering of all received data would imply high power consumption and introduce an unnecessary delay [2].

The aim of this paper is to show that it is possible to perform the 32-bit CRC used in Ethernet (CRC-32) at the speed of 10 Gb/s with process technologies available today. We look at two different designs and discuss their performance based on our simulations. Since we do not have access to cutting edge process technologies, we base our conclusions on extrapolation of device characteristics [9].

2. MODE OF OPERATION

All modern high-speed implementations of CRC make use of parallelism. This gives an obvious advantage over the original bit-serial implementation, since the required clock frequency can be reduced with a factor corresponding to the level of parallelism. In previous work, we have investigated some different architectures for parallel CRC generation and concluded that the fixed logic implementation is somewhat faster than a look-up-table based architecture [3].

The higher the parallelism, the lower the clock frequency can be used. Hobson and Cheung [4] have proposed a 32-bit parallel CRC-32 engine which could reach 5 Gb/s. For Ethernet however, it is desirable to have only 8-bit parallelism, since the minimum symbol in an Ethernet packet is 8 bits. This is to avoid complicated handling of initial and ending 8-bit symbols.

Glaise and Jacquart [5] have shown that the CRC-32 can be calculated efficiently by using the properties of Galois fields. Several other optimizations to the CRC algorithm have been made [6], but many of them are only suitable for software and actually make a pure hardware solution more complicated.

The general architecture of the CRC generator is shown in figure 1. The middle register is the CRC register. Since we have chosen the CRC-32 algorithm of Glaise & Jacquart, the CRC register must be preset not to all 1’s, as in most software implementations, but to the vector, in hex, “46AF6449” [5]. To achieve the correct CRC value, 4 bytes with all 0’s must also be supplied after the actual data. The output must be inverted, which is taken care of before the output register. The input and output registers do not have set or reset, but the register bits in the CRC register have set or reset according to the vector mentioned above. It is assumed that the set, reset, and clock signals can be generated to suit this architecture. That implies for example, that the clock signal is constantly “0” when set and reset are activated.

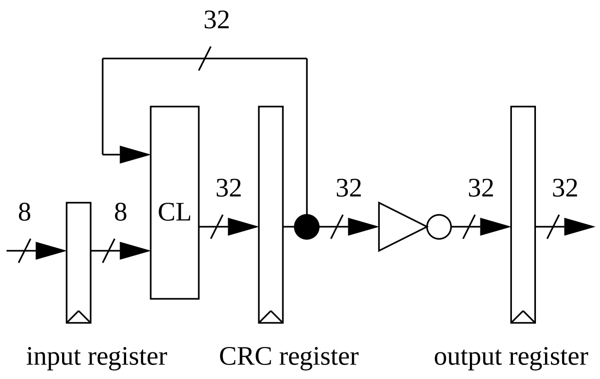

Figure 1. General architecture of the CRC generator

3. IMPLEMENTATION CONSIDERATIONS

The CRC generator has been implemented in two different ways. The target technology for a full-custom implementation was the 3 metal layer AMS 0.35 micron process and the target technology for a synthesized standard cell implementation was the 6 metal layer UMC 0.18 micron process.

3.1. Standard Cell Implementation

A key to achieve high speed is to have a flip-flop with low delay from clock to output and a high driving capability. The setup time should also be short. When doing a standard cell implementation, some flip-flops should be parallelized. This pertains to the flip-flops, that drive critical paths. The fan-out is shared on two flip-flops, so one is driving only the gate in the critical path and the other one is driving all other gates, that need the same input. By doing so the delay from the flip-flop to the first gate in the critical path is minimized. An example of this can be seen in figure 2. The input to the flip-flops is not in the critical path, so loading the last gate with two parallel flip-flops does not impact the maximum clock frequency for the design. Unfortunately the synthesis tool could not handle the parallelization of flip-flops, therefore it had to be done manually in the RTL code.

The CRC generator has been described in Mentor Graphics Renoir and VHDL. Synthesis was performed by Cadence Buildgates and timing-driven place&route was made by Cadence Silicon Ensemble (SE).

The standard cell implementation consists of totally 432 cells, which require a chip area of 11111 μm^2.

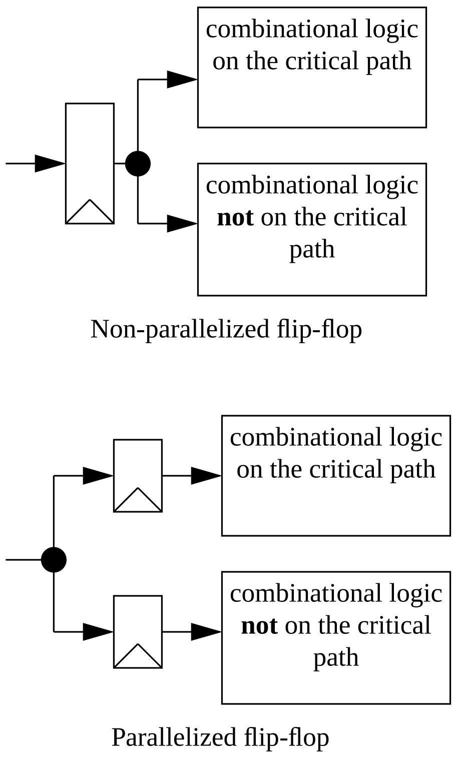
combinational logic on the critical path
combinational logic **not** on the critical path
Non-parallelized flip-flop
combinational logic on the critical path
combinational logic **not** on the critical path
Parallelized flip-flop

Figure 2. Parallelization of flip-flops

3.2. Full-Custom Implementation

By using the equations of Glaise and Jacquart, we can limit the maximum number of inputs to any of the bits in the CRC register to 8. This leaves us with the logic depth of 3 if we exclusively use 2 input XOR gates for the combinational logic. In the full-custom implementation we use the XOR gate proposed by Wang et. al. [7], which also was used by Hobson and Cheung.

To make the design work at the required high frequency, sizing of the logic gates is needed due to the high fan-out of some of the gates. Load balancing and logic sharing are also two subjects for optimization. We have generally tried to share as much logic as possible to reduce the fan-out of the CRC register.

In the full-custom design we have not used flip-flops for the CRC register. Instead we have used two stages of latches, with XOR gates and inverters merged into the latches, an example is shown in figure 3. The inverter in the latch is used as the last stage of the XOR gate. By doing this we reduce the logic depth and hence the total delay. The latches are the simplest possible, an inverter followed by a transmission gate, which have proven to be fast in [8].

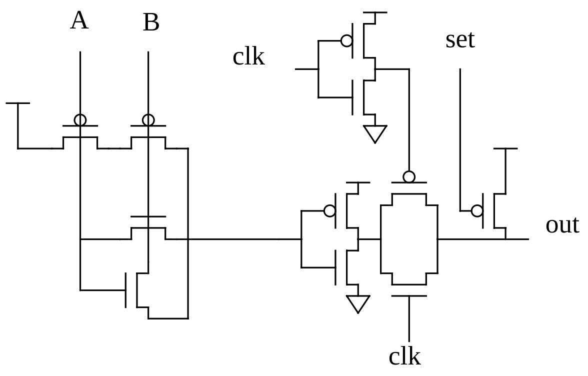

Figure 3. XOR merged into positive latch, with active low set signal

The XOR gates have one input, that is slightly faster than the other. In the critical path we have throughout the design used the faster input.

The full-custom implementation was designed in Mentor Graphics Layout Editor. The layout is shown in figure 4. Totally 908 transistors have been used and the largest transistors have W_p=17 µm and W_n=9 µm respectively.

4. SIMULATION RESULTS AND STATIC TIMING ANALYSIS

4.1. Standard Cell Implementation

For the standard cell implementation, the maximum clock frequency was identified by using the static timing analysis (STA) in Buildgates. When doing this, parasitic wire capacitances extracted from the layout by Silicon Ensemble were used.

A maximum clock frequency of 1.09 GHz was achieved. That corresponds to 8.7 Gb/s of processed data.

4.2. Full-Custom Implementation

The complete design has been simulated in switch-level mode in Mentor Graphics LSim simulator to verify the functional correctness.

Parasitic wire capacitances from the complete layout were extracted and the critical path was manually identified to be the structure, which can be seen in figure 5. When the simulation was performed, the critical path was separated from the layout and the adequate parasitic capacitances were used. This was done in order to be able to easily apply the worst-case input vector. The critical path was simulated with HSpice™ using level 49 with mean parameters.

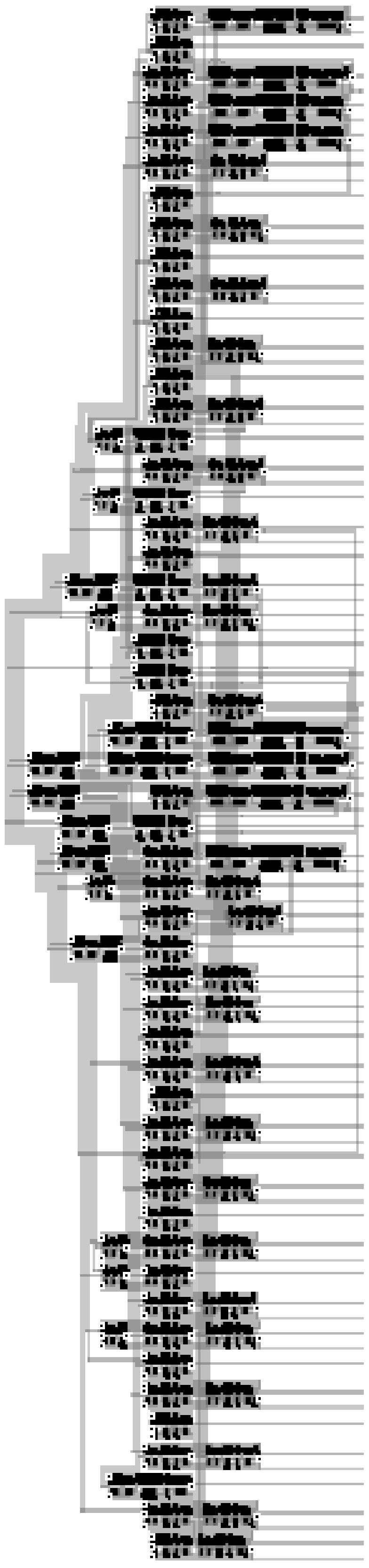

Figure 4. Layout of the CRC Generator in the full-custom implementation (width 150 µm, height 720 µm). Input and output registers are not included in the full-custom design.

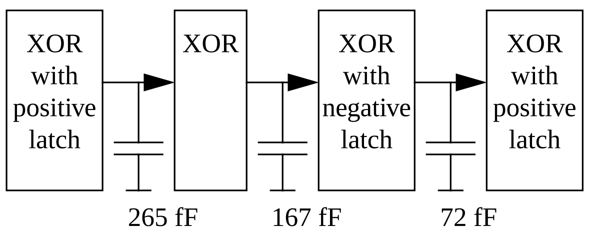

Figure 5. Critical path in the full-custom implementation. The loads specify the total output load, including output capacitance of the driving stage, wire capacitance, and input capacitance of the driven gates.

The critical path simulation gives a worst-case delay of 1.60 ns, which corresponds to a maximum clock frequency of 625 MHz, and thus a throughput of 5 Gb/s is achieved.

5. DISCUSSION AND EXTRAPOLATION OF RESULTS

With 8-bit parallelism and 10 Gb/s requirement, obviously a clock frequency of 1.25 GHz is needed, i.e. a clock period of 800 ps. However, this high speed cannot be expected from the 0.35 micron process technology, nor could it be achieved with the synthesized standard cell implementation in a 0.18 micron process.

The synthesized standard cell implementation technique is normally preferred in industry and therefore it is most important to be able to use that technique in order to achieve 10 Gb/s throughput. According to the SIA roadmap [9] the gate delay reduces with a factor of 1.27 when going from 0.18 micron to 0.15 micron technologies. We achieved 8.7 Gb/s in a 0.18 micron process and thus, using the scaling factor, approximately 11 Gb/s could be achieved with a 0.15 micron process. However, the scaling factor 1.27 only considers gate delay reduction. It does not say anything of what will happen with delays due to interconnects. The CRC generator is a fairly small logic block and gate delays dominate over delays due to interconnects. That means that we have strong indications that at least 10 Gb/s will be achieved in a 0.15 micron process.

If a full-custom implementation is made, it is possible to achieve a throughput of more than 10 Gb/s with a 0.18 micron technology, since the scaling factor when going from 0.35 micron to 0.18 micron technologies is more than 2.

6. CONCLUSION

We conclude that it is possible to handle CRC-32 calculation for data streams up to 10 Gb/s by using a standard cell synthesized design. The requirement is to use a 0.15 micron process technology.

A full-custom design can reach a throughput of more than 10 Gb/s in a 0.18 micron process technology. This requires that a high-performance XOR gate is used in combination with fast latches and that a compact layout is made.

7. ACKNOWLEDGEMENT

The authors would like to acknowledge SwitchCore for interesting discussions. This study was supported by the Swedish Foundation for Strategic Research (SSF), the Swedish Research Council for Engineering Sciences (TFR) under contract 314-97-289, and the Center for Industrial Information Technology at Linköping Institute of Technology (CENIIT).

REFERENCES

[1] H. Frazier, “10Gig MII update”, on *the www*, http://grouper.ieee.org/groups/802/3/10G_study/public/nov99/index.html

[2] D. Liu, U. Nordqvist, and C. Svensson, “Configuration-Based Architecture for High Speed and General-Purpose Protocol Processing”, *IEEE Workshop on Signal Processing Systems*, Taipei, Taiwan, 1999, pp. 540-547.

[3] U. Nordqvist, T. Henriksson, and D. Liu, “CRC Generation for Protocol Processing”, *Norchip 2000*, Turku, Finland, pp. 288-293.

[4] R. F. Hobson and K. L. Cheung, “A High-Performance CMOS 32-Bit Parallel CRC Engine”, *IEEE Journal of Solid-State Circuits*, vol. 34, no. 2, February 1999, pp. 233-235.

[5] R. J. Glaise and X. Jacquart, “FAST CRC CALCULATION”, *IEEE International Conference on Computer Design: VLSI in Computers and Processors*, Cambridge, MA, USA, 1993, pp. 602-605.

[6] R. N. Williams, “A Painless Guide to CRC Error Detection Algorithms”, on *the www*, http://www.ross.net/crc/ version 3.00, 19 Aug. 1993.

[7] J.-M. Wang, S.-C. Fang, and W.-S. Feng, “New efficient designs for XOR and XNOR functions on the transistor level”, *IEEE Journal of Solid-State Circuits*, vol. 29, July 1994, pp. 780-786.

[8] C. Svensson and J. Yuan, “Latches and Flip-flops for Low Power Systems”, *Low-Power CMOS Design*, New York: IEEE Press, 1998, part IV, pp. 233-238.

[9] Semiconductor Industry Association, *International Technology Roadmap for Semiconductors: 1999 edition*, Austin, TX:International SEMATECH, 1999.