Microelectronics Journal 51 (2016) 15–29

Contents lists available at ScienceDirect

Microelectronics Journal

journal homepage: www.elsevier.com/locate/mejo

Design of a compact reversible fault tolerant division circuit

Hafiz Md. Hasan Babu [a,*], Md. Solaiman Mia [b,*]

[a] Department of Mechatronics Engineering, University of Dhaka, Dhaka-1000, Bangladesh
[b] Department of Computer Science and Engineering, University of Dhaka, Dhaka 1000, Bangladesh

ARTICLE INFO

Article history:
Received 24 May 2014
Received in revised form
27 September 2015
Accepted 24 January 2016
Available online 1 March 2016

Keywords:
Reversible logic
Fault tolerant
Reversible divider
Quantum cost
Delay

ABSTRACT

In this paper, we propose an n-bit reversible fault tolerant binary division circuit, where n is the number of bits of dividend and divisor. We present a new algorithm for division operation with the optimum time complexity in the design of dividers. The proposed division method consists of four steps: Firstly, it considers floating-point data and rounding. Secondly, it performs correctly rounded division. Thirdly, it performs correct rounding from one sided approximations. Finally, it calculates the result of the division operation. The proposed design of the divider circuit shows that it is composed of reversible fault tolerant multiplexers, parallel-in–parallel out (PIPO) left shift registers, D-Latch, rounding and normalization registers and parallel adder. The proposed divisor register and the parallel adder have the minimum quantum cost with respect to the existing ones. Fredkin gates and Feynman double gates are also used to form the divider circuit. Finally, we present an algorithm to construct a compact n-bit reversible fault tolerant binary division circuit. In this paper, a new algorithm has also been proposed to reduce the number of steps required for performing division operation. Our circuit performs better than the existing approaches considering all the efficiency parameters of reversible logic design which includes number of gates, constant inputs, garbage outputs, quantum cost and delay of the circuit, e.g., for a 256-bit binary division circuit, the proposed reversible fault tolerant binary division circuit improves 27.75% on the number of gates, 0.03% on garbage outputs, 11.04% on quantum cost, 8.94% on constant inputs and 23.50% on delay with respect to the best known existing divider circuit. We also simulate the proposed n-bit reversible fault tolerant binary division circuit using Microwind DSCH 3 which shows the correctness of the circuit.

1. Introduction

Energy consumption is a vital issue in modern VLSI designs. Though the improvement in the higher-level integration and the advancement of new fabrication processes have significantly reduced the heat loss over the last decades, physical limit exists in the reduction of heat. It has been proved that for irreversible logic computations, each bit of information lost generates $kTln2$ joules of heat energy, where k is Boltzmann's constant of 1.38×10^{-22} J/K and T is the absolute temperature at which computation is performed [1]. Bennett showed that $kTln2$ energy dissipation would not occur if the computation was carried out in a reversible manner [2]. Reversible logic is a potential area of study regarding further technological innovations. The major application of reversible logic is in quantum computation as quantum circuits must be reversible [3]. It has been widely applied in various research areas such as optical computing [4], ultra low power CMOS design [5], DNA computing, quantum computing, thermodynamic technology, bioinformatics and nanotechnology [6].

Parity checking is one of the widely used error detection mechanisms in digital logic and data communication systems. This is because most of the arithmetic functions is not parity preserving. If the parity of the input data is maintained throughout the computation, no intermediate checking would be required [23]. A sufficient requirement for parity preservation of a reversible circuit is that each gate be parity preserving [23]. Thus, we need parity preserving reversible logic gate to construct parity preserving reversible circuits.

Division is the most difficult operation in the computer arithmetic [9,12]. Nowadays, people use a hardware module divider to implement the division algorithm. Conventionally sequential circuits are used to implement the divider. The division circuit can be used in the arithmetic unit of a processor. In this paper, we propose a fault tolerant reversible division circuit. We demonstrate that the proposed divider is better than the existing ones in terms of number of gates, constant inputs, garbage outputs, quantum cost and delay of the circuit.

* Corresponding authors. Tel.: +880 2 963 4854; fax: +880 2 861 5583.
E-mail address: hafizbabu@du.ac.bd (H.Md.H. Babu).

http://dx.doi.org/10.1016/j.mejo.2016.01.003
0026-2692/© 2016 Elsevier Ltd. All rights reserved.

2. Background study of existing division circuits

In arithmetic and algebraic computation, the basic operations are addition, subtraction, multiplication and division. It is a fundamental problem to find efficient algorithms for division, as it seems to be the most difficult of these basic operations. The division operation has been regarded as a significant verification challenge. This operation is usually implemented as a sequence of basic operations rather than a dedicated logic circuit [36]. As a result, it exhibits both an exceptionally wide array of intricate corner cases and an immense state space, challenging both simulation and formal methods.

2.1. Existing division algorithms

Designers of processors with enhanced arithmetic and logic unit (ALU) are always on a look out for algorithms for performing basic operations one of them being division. The algorithm should be such that it involves processes requiring compact hardware while satisfying efficiency constraints. In this section, we briefly discuss about the existing division algorithms with their properties.

2.1.1. Binary floating-point division algorithm [7]

This algorithm deals with the implementation of low latency software for binary floating-point division with correct rounding to nearest. The approach presented in [7] targets a VLIW integer processor of the ST200 family and is based on fast and accurate programs for evaluating some particular bivariate polynomials. With the ST200 compiler and compared to previous implementations, the speed-up observed with this approach is by a factor of almost 1.8.

But the method in [7] has some critical problems. This method is good for binary32 format and for a particular 32-bit architecture. But when optimizing division codes presented in [7] for other processors and/or other floating-point formats (like binary64, binary128, or smaller formats used in computer graphics), the performance of this method will reduce because of gradual underflow [8]. Moreover the division code in [7] should definitely be extended in order to support subnormal numbers while keeping the latency as low as possible. Finally, all the rounding attributes that are prescribed in [13] should be implemented.

2.1.2. Floating-point division algorithm [10]

This method presents the AMD-K7 IEEE 754 and $\times 87$ compliant floating-point division. Highly accurate initial approximations and a high performance shared floating point multiplier assist in achieving low division latencies at high operating frequencies. This algorithm also describes a novel time-sharing technique which allows independent floating point multiplication operations to proceed while division is in progress.

But with this method, the extra multiplier hardware required to support division that actually impacted total area included flip-flops to store intermediate results, an incremental and the state machine. The division algorithm presented in [10] accounts for about 10% of the total area of the multiplier unit.

2.1.3. Floating-point division using a Taylor-series expansion algorithm [11]

This method presents the implementation of a fused floating point divide unit based on a Taylor-series expansion algorithm [11]. By this algorithm, the resulting arithmetic unit also exhibits high throughput and moderate latency as compared with other floating-point unit (FPU) implementations of leading architectures. Moreover, this algorithm achieves fast computation by using parallel powering units such as squaring and cubing units, which compute the higher-order terms significantly faster than traditional multipliers with a relatively small hardware overhead.

Even though the Taylor-series expansion algorithm with powering units exhibits the highest performance among multiplicative algorithms, it consumes a large area because the architecture shown in [11] consists of four multipliers, which is not suitable for area-critical applications.

2.2. Existing designs of reversible division circuits

Division is the most difficult operation in the computer arithmetic. Now-a-days people use a hardware module-divider to implement the division algorithm. Conventionally sequential circuits are used to implement the divider. In this section, we describe some existing divider circuits with their properties.

2.2.1. Fault tolerant reversible divider [24]

Dastan et al. proposed fault tolerant design of a reversible divider [24] in 2011. In this design, some reversible fault tolerant components like reversible fault tolerant parallel adder, reversible fault tolerant shift register and reversible fault tolerant n-bit register have been proposed.

Even though the design of [24] illustrated with the first design of fault tolerant reversible divider, the design required more circuitry compared to later designs in the literature. Also the design can handle only positive integers.

2.2.2. Reversible signed divider with overflow checking capability [30]

In 2012, Dastan et al. proposed another fault tolerant design of the reversible divider [30]. This reversible division circuit is a signed divider and has an overflow checking capability. This division circuit is the first reversible signed divider with overflow checking capability.

The design of [30] can handle signed numbers, but this design is not fault tolerant.

2.2.3. Reversible floating point divider [31]

Jamal et al. proposed two approaches for constructing a reversible divider [31] in 2013. A conventional division array was used in the first approach and a high speed division array was used in the second approach. Both of the approaches can handle floating point numbers.

But both of the approaches designed using non-fault tolerant reversible logic.

None of the existing reversible dividers [24,30,31] can have both the floating point numbers and the fault tolerant features at the same time.

3. Proposed division method

In this section, we show a new method for binary floating-point division. In Sections 3.1 and 3.2, we propose a new algorithm for binary floating-point division and present a comparison of the proposed algorithm with others in terms of time complexity, respectively.

Although floating-point (FP) divisions are less frequent in applications than other basic arithmetic operations, reducing their latency is often an issue [37]. Since low latency implementations may typically be obtained by expressing and exploiting instruction parallelism, intrinsically parallel algorithms tend to be favored. In this paper, we propose an algorithm to get the desired result which is described below.

In the first step, working with floating-point numbers requires some understanding of the internal representation of data. Researchers must be aware of the finite precision issues. For