Implementation and Physical Design of 8/4-Bit Signed Divider

Project Seminar Report

Seminar/Project Report submitted in partial fulfilment of the requirements for the award of the degree of BE. in Electronics and Communication Engineering under Muffakham Jah College of Engineering and Technology.

By

Mohammed Abdur Rahman

Roll No. 1604-18-735-119

Shaik Musharraf Ali

Roll No. 1604-18-735-311

Syed Moizuddin

Roll No. 1604-18-735-112

Under the Guidance of

Md. Zakir Hussain

(Associate Professor, Dept. of Electronic and Communication Engineering)

Abstract

This paper focuses on implementing a signed binary divider using Verilog and performing a physical design process i.e., register transfer level (RTL) to graphic design systemII (GDSII) on 180nm and 45nm technology nodes to analyze different parameters such as delay, area, power, and bandwidth. It also extends the unsigned divider to the signed divider and hence increases the range of division. On 180nm, this divider consumes 14263.2 µm2 area, the power consumption of 20 µW and bandwidth of 666.66 MHz, whereas for 45nm node area consumption of this divider is 600.130 µm2 with a power consumption of 17.88 µW and bandwidth of 892Mhz.

INDEX

Introduction

1.1 Current trends in the chosen title

Divider is an inevitable and fundamental equipment module utilized in advanced and fast computerized flag handling (DSP) units of high accuracy. Division is one of the four essential operations of arithmetic mathematics. The division of two common numbers is the way toward ascertaining the quantity of times one number is contained inside each other. Divider is fundamental equipment utilizes in rapid and progressed advanced signal processing (DSP) units. It's most vital part in high exactness radar innovation, cryptography and Linear predictive coding (LPC) for signal processing. As the innovation shrivel, dividers play out an essential part in each field like signal processing, instalment through NFC. Computerized flag preparing is where quick handling of bits required, so speedier calculations are presented. In contrast with other scientific operations, division is the sequential kind of operation that outcomes in complex equipment usage. The exceptionally accurate division calculations are the fundamental necessity of signal and image handling applications.

1.2 Potential for the work in the area

Demands for more computational speeds of digital systems are continuously increasing. These demands for mobile devices (as well as other non-mobile ones) are accompanied by the requirement of low power/energy consumptions. In computational systems including those performing digital signal processing, the speed and energy consumption of the arithmetic units have determining roles in the speed and energy consumption of the whole system. Among the four basic operations of Add, Sub, Mul, and Div., the last one has been used less frequently compared to the others.

1.3 Synopsis of previous work

Fairly recent paper which we encountered presented about a high speed yet energy-efficient approximate divider for error resilient applications was proposed. For the division operation, the divisor is rounded to a value with a specific form resulting in the transformation of the division operation to the multiplication one. The proposed approximate divider enjoys the flexibility of increasing the accuracy at the price of higher delay and hardware usage. The results show that the delay and energy consumption of the proposed approximate divider are, on average, 14 and 300 times smaller than those of the Radix-2 SRT with the carry-save reminder computation

1.4Problem Statement

Internet of Things (IoT) makes limitation on power consumption for sensors and portable devices. Thus, exact computation for division processes is not allowed due to the huge amount of power consumption. To address these limitations, allow power division computing units is required in a typical program, an arithmetic unit produces thousands of division computations per second. To compute and produce correct results, the ALU ‘s division algorithms must be as effective as possible. Existing dividers are good enough but they are used to perform ordinary unsigned division which result in the following cons: -

More power consumption, large area, Less performance (Frequency) and More Latency. Usage of commercial electronic design automation (EDA) tools is not possible by an individual and also for start-up firms because of the Purchase cost, Maintenance and License renewal issues. Commercial process design kits (PDKs) are also expensive which individual or startup companies find it difficult to procure.

.

1.5 Motivation

Digital Signal Processing (DSP) has applications in many areas including efficient computational blocks. Low power dissipation and reduced area are some of the major concerns in implantable and portable electronics used in DSP systems. Divider is one of the important arithmetic blocks in DSP application. Floating point division algorithms for high-speed VLSI employ signed digit number systems because of its carry-free property number, an alternative design of VLSI-based array dividers with concurrent error detection by recomputing using partitioned architecture (REPA) is proposed. The basic concept is that the divider array can be divided into two identical parts and its operation can be completed by using one part through two iterative calculations. With two such parts, a concurrent error detection scheme can be designed by using a space redundancy approach, and the detecting action is achieved at each iteration.

1.6 Organization of the Report

The proposition is organized as the following sections which present basis for the said approach.

a.) Background work

b.) Literature Survey

c.) Proposed problem

d.) Design Methodology

e.) Algorithm

f.) Objective

g.) References

Basic Background

In recent years, the arithmetic logical unit (ALU) has been changed many times. A significant change is observed in the implementation of an individual multiplication and division unit of the ALU. Algorithms for division units also get updated from time to time for different applications. In a typical program, an arithmetic unit produces thousands of division computations per second. To compute and produce correct results, the ALU ‘s division algorithms must be as effective as possible. Since the age of computer arithmetic, researchers have designed many algorithms to divide numbers. Some algorithms work well for hand analysis, and some are good for hardware computation. Digital dividers generally can be categorized as employing arithmetic operations to execute a division operation. Arithmetic dividers receive an input that combines the numerator and denominator. Look-up table implementations often require large look-up tables to be accurate for high-speed division, which is generally requiring significant processing time and chip space. Many look-up table implementations also require multiple iterations to improve accuracy, which increases latency associated with the division operation.

Literature Survey

[1] H. Kaur,” Performance Analysis of Various Multiplication and Division Algorithms for Large Numbers”

This paper provides a detailed study on the algorithms used by an ALU to perform multiplication and division for large numbers, and recommends one algorithm that will give best performance for division and multiplication. The multiplication algorithms that are analyzed are Pen and Paper algorithm, Booth’s algorithm, and Divide and Conquer algorithm. The division algorithms that are analyzed are Radix 2 restoring algorithm, Radix 2 non-restoring algorithm, and Radix 4 restoring algorithm. The algorithms are implemented using Verilog and the timing and area reports generated after synthesis is used to compare the algorithms. This paper concludes that out of the examined algorithms divide and conquer algorithm gives the best performance for multiplication, while Radix 4 restoring algorithm gives the best performance for division.

[2] Gaalswyk, F. Matthew, and J. Stine,” A Low-Power Recurrence-Based Radix 4 Divider Using Signed-Digit Addition,”

This paper presents a novel radix-4 division by recurrence architecture that utilizes a hierarchical Signed-Digit (SD) adder. The implementations are easily generated based on the methodology as it is suited towards digital implementations. Results are generated for several designs using Global

Foundries 45nm SOI technology and ARM standard cells. Results indicate that power dissipation can be reduced using these architectures for division by recurrence as the area is significantly decreased.

[3] R. Vemula, et. al.," Design and Implementation of 64-Bit Divider Using 45nm CMOS Technology,"

Divider is an inevitable and fundamental equipment module utilized in advanced and fast computerized flag handling (DSP) units of high accuracy. This paper proposes the usage of a low power and fast Vedic Divider in view of old Indian Vedic science. In this paper, a calculation in view of the "ParavartyaYojayet" is applied, all through this sutra the proliferation deferral(delay) and power utilization are diminished to a degree. As considered, division operation is more mind boggling in the calculation of the advanced applications. The most noteworthy part of this paper is to diminish the power utilization and give high speed. In this work decimal and binary division calculations are performed. The synthesized results are implemented on RTL compiler and SoC Encounter i.e GDSII file Cadence tools utilizing 45nm innovation of CMOS. The recreated comes about for proposed Vedic divider demonstrates a lessening in deferral(area) and power utilization against other division techniques.

Design Algorithm of Divider

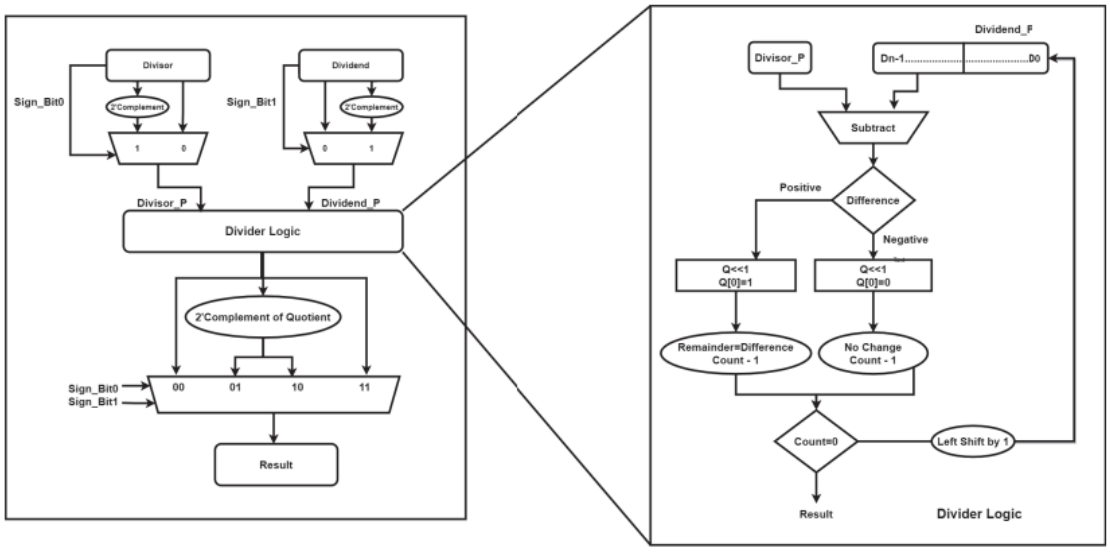

Fig. 1. Proposed design Algorithm of divider.

Project Overview

4.1 Proposed Problem

To analyze different parameters such as delay, area, power, and bandwidth. It also extends the unsigned divider to the signed divider and hence increases the range of division. On 180nm, this divider consumes 14263.2 μm2 area, the power consumption of 20 μW and bandwidth of 666.66 MHz, whereas for 45nm node area consumption of this divider is 600.130 μm2 with a power consumption of 17.88 μW and bandwidth of 892Mhz. the vendor to provide an optimizer and to keep it up-to date, while they are granted only limited insight into the reliability of its results.

Internet of Things (IoT) makes limitation on power consumption for sensors and portable devices. Thus, exact computation for division processes is not allowed due to the huge amount of power consumption. To address these limitations, a low power division computing units is required

4.2 Design Specifications

Physical design is converting an RTL gate-level netlist into planar geometric shapes GDSII file, which can be used by foundries for fabrication. It contains different steps, like floor planning, placement, clock tree synthesis, and routing. All steps of physical design are shown in Fig.2.

Physical design starts with a netlist, the netlist is synthesized from RTL, and it describes the circuit components in a manner how they are connected.

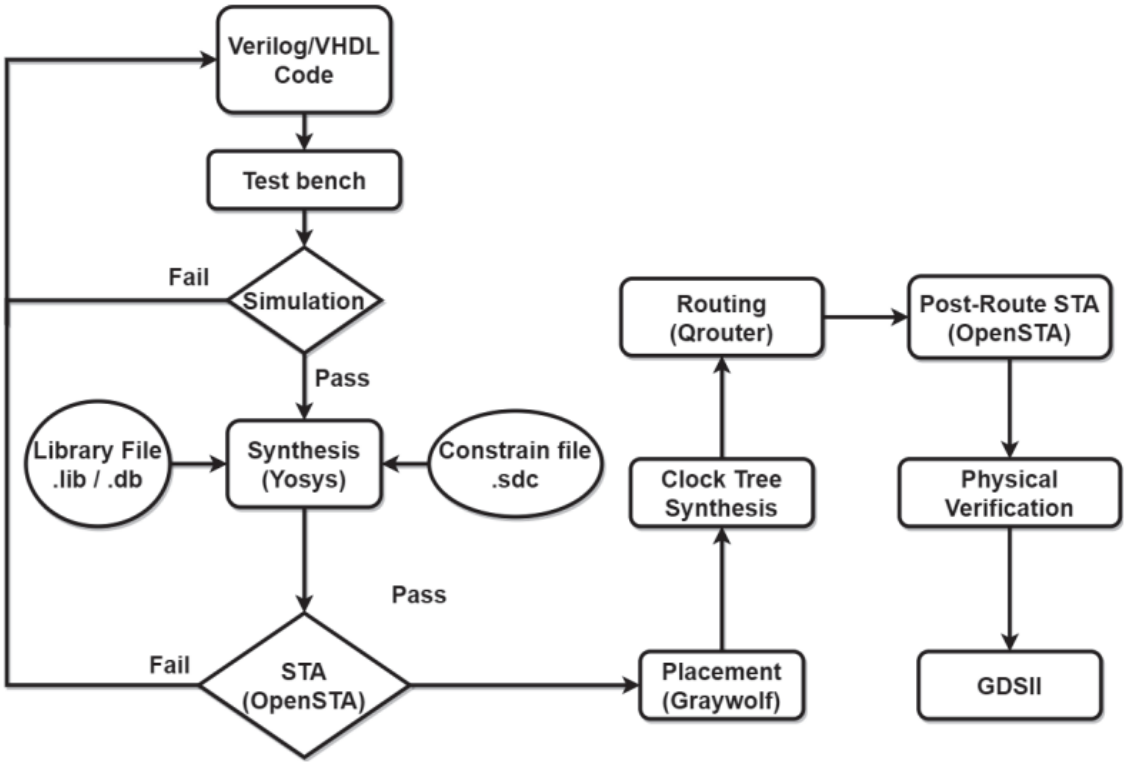

Fig. 2. Physical design flow.

RTL of our proposed design is converted to GDSII by passing through various steps using the free EDA tool i.e., Qflow. This tool takes RTL code as an input, which is synthesized in yosys, and generates the synthesized netlist. This netlist is then given as an input to the next tool in the process and at last, the GDSII layout is attained.

4.3 Description

Here we are doing project in two main ways, they are

1) Physical Design Flow

2) Front end design and Back-end design, which will be done by various open tools.

Physical Design Flow

This is a 5-step process they are

1. **Synthesis:** Synthesis is a process of converting the Verilog code into a logic gate-level netlist. The synthesis tool requires two input files. First is the technology library file(.lib), which contains standard cells. The second file is a constraint file(.sdc), including timing, loading, and optimization algorithm for logic optimization. After synthesis, the next step is floor Planning.
2. **Placement**: The location of every component on the die can be determined by Placement by considering the timing data and length of interconnects. To optimize the placement step, it is divided into 4 phases: 1. Pre-placement optimization, 2. In placement optimization 3. Post-placement optimization (PPO) before clock tree synthesis (CTS) 4. Post-placement optimization after CTS.
3. **Clock tree synthesis (CTS):** CTS's main objective is to minimize the skew and latency by inserting inverters or buffers so that the clock shall be distributed equally.
4. **Routing**: Routing aims to decide the interconnects paths, which includes macro pins & standard cells. Routing makes all the connections described in the netlist too effectively without violating setup and holding time constraints.
5. **Physical Verification**: Physical verification checks whether the generated design is correct or not.

A. Front end design using Open-Source EDA tools

a) **RTL simulation**: To check the functionality of the circuit proposed in this paper Xilinx Ise tool was used which gives the output waveform for certain inputs as shown in below

b) **Synthesis**: Open-source tool yosys is used for generating synthesizable Verilog code which is necessary for timing analysis in the back-end flow.

c) **Static timing analysis**: Static timing analysis (STA) is required to check the setup and hold time constraints of the design. STA is performed by OpenSTA, a free, open-source tool.

B. Back End Design using Open-Source EDA tools

1)Placement: Location of every component on the die can be determined by Placement by considering the timing data and length of interconnects. To optimize the placement step, it is divided into 4 phases: Pre-placement optimization, in placement optimization, post-placement optimization before CTS, post placement optimization after CTS. We carried out the placement step using the Qflow tool with an aspect ratio setting of 0.5, 1, 0.75 and an initial density setting of 1. Initial density and aspect ratio do not make much difference in the physical design layout area's size.

2) Routing: The goal of routing is to decide the interconnect paths, including macro pins and standard cells. Routing makes all the connections described in the netlist too effectively without violating setup and holding time constraints.

3) Migration: Migration is used to convert DEF files produced by routing into a database for the magic layout editor. In this step, a layout can be corrected manually to avoid errors in design rule check (DRC) and layout vs. schematic (LVS). Fig. 7 shows the layout of developed signed divider on 180nm technology node with an aspect ratio of 1 and initial density is also 1. The layout contains standard cells i.e., NAND, NOR, AOI, OAI.

4.4 Design Methodology

Register Transfer Level (RTL) is an abstraction for defining the digital portions of a design. It is the principal abstraction used for defining electronic systems today and often serves as the golden model in the design and verification flow. The RTL design is usually captured using a hardware description language (HDL) such as Verilog or VHDL. While these languages are capable of defining systems at other levels of abstraction, it is generally the RTL semantics of these languages, and indeed a subset of these languages defined as the synthesizable subset. This means the language constructs that can be reliably fed into a logic synthesis tool which in turn creates the gate-level abstraction of the design that is used for all downstream implementation operations.

RTL is based on synchronous logic and contains three primary pieces namely, registers which hold state information, combinatorial logic which defines the nest state inputs and clocks that control when the state changes.

GDS II is a database file format which is the de facto industry standard for data exchange of integrated circuit or IC layout artwork. It is a binary file format representing planar geometric shapes, text labels, and other information about the layout in hierarchical form. The data can be used to reconstruct all or part of the artwork to be used in sharing layouts, transferring artwork between different tools, or creating photo masks. Initially, GDS II was designed as a format used to control integrated circuit photo mask plotting. Despite its limited set of features and low data density, it became the industry's default format for transfer of IC layout data between design tools of different vendors, all of which (at that time) operated with often incompatible and proprietary data formats. The file format was originally developed by Calma for its layout design software, "Graphic Data System" ("GDS") and "GDS II". Currently, the format is owned by Cadence Design Systems. GDS II files are usually the final output product of the IC design cycle and are given to silicon foundries for IC fabrication. GDS II files were originally placed on magnetic tapes. This moment was fittingly called "tape out" though it is not the original root of the term. Objects contained in a GDS II file are grouped by assigning numerical attributes to them including "layer number", "datatype" or "text type". While these attributes were designed to correspond to the

"layers of material" used in manufacturing an integrated circuit, their meaning rapidly became more abstract to reflect the way that the physical layout is designed

4.5 Algorithms

The proposed architecture for the signed binary divider uses the non-restoring divider logic as a base block. In this work the size of the dividend is 8-bit and the size of the divisor is 4-bit. The algorithm is demonstrated in Fig. 1. The steps of a block diagram are explained below. In Fig. 1 Dn−1 to D0 are bits of Dividend p in this paper it is given as D7 to D0 as we designed the divider for an 8-bit dividend.

1) First based on sign bit of dividend and divisor, positive values of dividend and divisor is processed.

2) These positive values are then going to non-restoring divider logic.

3) Quotient and reminder generated from divider logic are still incorrect.

4) Quotient and the remainder is calculated with the help of sign bits of dividend and divisor. The Table I for the same is mentioned below.

TABLE I
SIGN OF QUOTIENT AND REMINDER FOR DIFFERENT COMBINATION OF DIVIDEND AND DIVISORS

Dividend	Divisor	Quotient	Remainder
Positive	Positive	Positive	Positive
Positive	Negative	Negative	Positive
Negative	Positive	Negative	Negative
Negative	Negative	Positive	Negative

NON-RESTORING DIVISION ALGORITHM

The non-restoring algorithm comes from restoring division and it calculates the remainder by successively subtracting the shifted divisor from the dividend until the remainder is in the appropriate range. The method is [9]: Assume that we have dividend, D and divisor, X as an input

data, quotient, Q as division result and R as remainder. The steps of the non-restoring algorithm are calculated as visible in Figure-1.

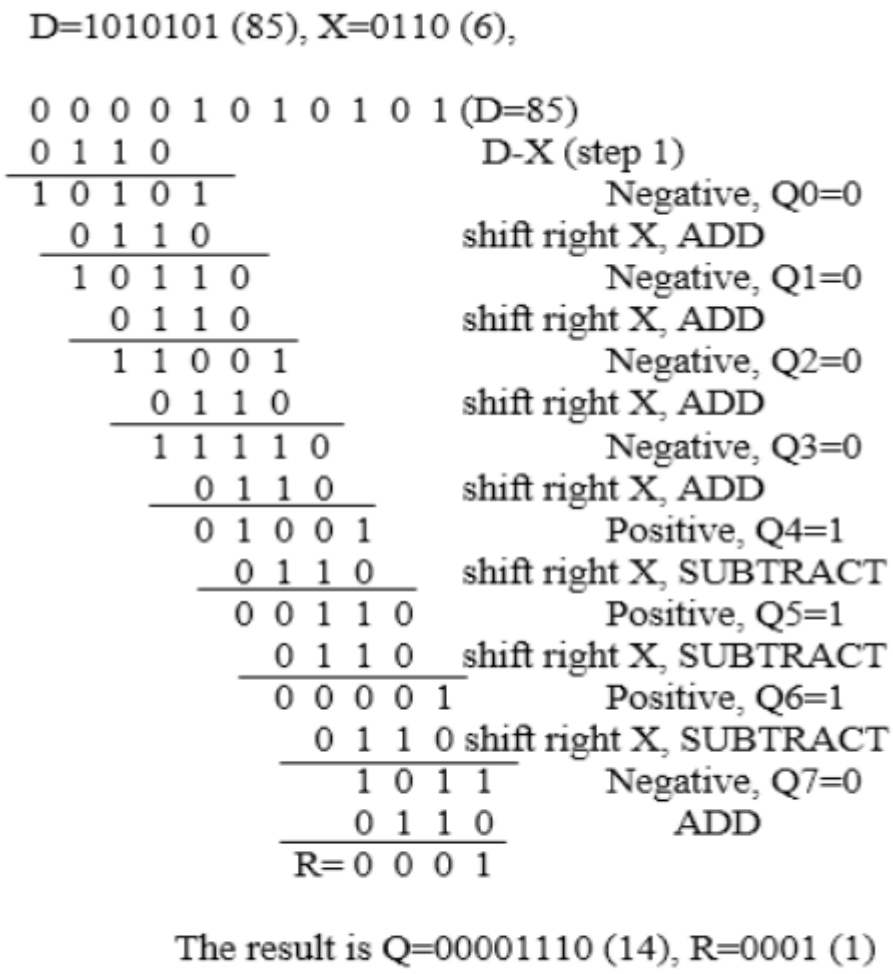

Figure-1. Steps to calculate binary non-restoring division algorithm.

References

- H. Kaur," Performance Analysis of Various Multiplication and Division Algorithms for Large Numbers".
- Y. Yusmardiah, et al.," Translation of Division Algorithm Into Verilog HDL," ARPN Journal of Engineering and Applied Sciences, vol. 12, pp. 3214–3217, 2006.
- T. Aoki, K. Nakazawa, and T. Higuchi," High-radix parallel VLSI dividers without using quotient digit selection tables," in Proc. 30th IEEE International Symposium on Multiple-Valued Logic (ISMVL 2000), pp.345- 352, 2000.
- K. Reddy, et al.," Design of approximate dividers for error-tolerant applications," in Proc. 61st International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 496-499, 2018.
- Gaalswyk, F. Matthew, and J. Stine," A Low-Power Recurrence-Based Radix 4 Divider Using Signed-Digit Addition, "in proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 391-396, 2019.
- R. Zendegani, et al.," SEERAD: A high speed yet energy-efficient rounding-based approximate divider," in proc. Design, Automation & Test in Europe Conference & Exhibition, pp. 1481-1484, 2016.
- R. Vemula, et. al.," Design and Implementation of 64-Bit Divider Using 45nm CMOS Technology," International Journal of Pure and Applied Mathematics, 2018.
- S. Panda, and A. Sahu," A novel Vedic divider architecture with reduced delay for VLSI applications," International Journal of Computer Applications, 2015
- M. Basha, et al.," Novel Low Power and High-speed array divider in 65 nm Technology," International Journal of Advances in Science and Technology, pp.44-56.
- C. Senthilpari, et al.," Lower delay and area efficient non-restoring array divider by using Shannon based adder technique," in proc. IEEE International Conference on Semiconductor Electronics, pp. 140-144, 2010.
- S. Venkatachalam, et al.," Design of approximate restoring dividers," IEEE International Symposium on Circuits and Systems, pp. 1-5, 2019.