Engineering Science and Technology, an International Journal 21 (2018) 1–6

HOSTED BY

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

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

Full Length Article

Decimal multiplication using compressor based-BCD to binary converter

Sasidhar Mukkamala [a,*], Pradeep Rathore [b], Rangababu Peesapati [b,*]

[a] Department of Electronics & Communication Engineering, National Institute of Technology Calicut, Kozhikode, Kerala 673601, India
[b] Department of Electronics & Communication Engineering, National Institute of Technology Meghalaya, Shillong 793003, India

ARTICLE INFO

Article history:
Received 25 August 2017
Revised 9 January 2018
Accepted 9 January 2018
Available online 1 February 2018

Keywords:
BCD to binary (BCD-Bin) converter
Compressor
Field Programmable Gate Array (FPGA)
Application Specific Integrated Circuit (ASIC)

ABSTRACT

The objective of this work is to implement a scalable decimal to binary converter from 8 to 64 bits (i.e 2-digit to 16-digit) using parallel architecture. The proposed converters, along with binary coded decimal (BCD) adder and binary to BCD converters, are used in parallel implementation of Urdhva Triyakbhyam (UT)-based 32-bit BCD multiplier. To increase the performance, compressor circuits were used in converters and multiplier. The designed hardware circuits were verified by behavioural and post layout simulations. The implementation was carried out using Virtex-6 Field Programmable Gate Array (FPGA) and Application Specific Integrated Circuit (ASIC) with 90-nm technology library platforms. The results on FPGA shows that compressor based converters and multipliers produced less amount of propagation delay with a slight increase of hardware resources. In case of ASIC implementation, a compressor based converter delay is equivalent to conventional converter with a slight increase of gate count. However, the reduction of delay is evident in case of compressor based multiplier.

1. Introduction

The need for decimal arithmetic is rapidly increasing in financial, commercial and internet based computations [1]. Binary approximations produce less accurate results while the decimal fractions give more accurate results. The traditional method used for the conversion of binary numbers to decimal format is shift and add-3 algorithm [2], but conversion of decimal to binary is computationally intensive due to multiplication operations. Implementation of these algorithms in hardware produce much better execution time results than produced in software. Typically, hardware implementation accelerates the execution process and reduces the execution time 3–4 times compared with software [1]. Recent microprocessors such as the IBM PowerPC [3] has started to include a separate hardware for decimal-floating point operations to achieve lower execution time, however hardware realization of decimal arithmetic is not mature yet to be implemented in commercial processors. When implemented in hardware, a larger area is consumed due to complex operations, which result in slower functioning and high power consumption [4]. A need has araised to develop efficient hardware architectures for BCD to binary converters, BCD adders and BCD multipliers. Field Programmable Gate Array (FPGA) is a modern computing platform for the hardware implementation of algorithms by developing the architectures using pipeline or parallel processing techniques.

In the recent past, several works have been reported in literature on improving the performance of BCD to binary converters and multiplier on the basis of speed, area and power. Rekha et al. [5] developed fixed point multiplier using single digit BCD multiplier. The results show that design has a latency of 34.97∼ns and area of 30575 μm. Shuli et al. [6] developed BCD multipliers (4×4, 8×8, and 16×16) using compressor structures. Novelty of this work concerns generation of partial products which uses 4-bit binary multipliers and parallel binary operations based on 2-digit columns. The resultant product is obtained by combining 2-digit column-based partial products. Further, a decimal compressor structure was developed and used for partial product reduction which was implemented on Xilinx Virtex-5 and Virtex-6 FPGAs. The results show critical path delay reduction. Fazlali et al. [7] developed a BCD digit multiplication scheme based on binary multiplier and binary to BCD converter. This work used UMC 65 nm CMOS standard library for hardware implementation and results shown 19% hardware acceleration. Al-Khaleel et al. [8] proposed a 1 and 2 digit multiplier in Xilinx Virtex-5 XC5VLX30-3 FPGA. The results of 1-digit multiplier showed minimal delay and area as 4.4∼ns, 25 look-up-tables (LUTS) which are smaller than theoretical expectations. The same group [9] developed BCD to binary conversion circuits where bits of the BCD inputs are grouped. The binary result of each group was

* Corresponding authors.
E-mail addresses: sasidhar2904@gmail.com (S. Mukkamala), pradeeprathore@nitm.ac.in (P. Rathore), p.rangababu@nitm.ac.in (R. Peesapati).
Peer review under responsibility of Karabuk University.

https://doi.org/10.1016/j.jestch.2018.01.003
2215-0986/

calculated separately. The final result was obtained by adding each groups. Arvind et al. [10] proposed a binary to BCD conversion and multiplication on Virtex-4XC4VFX-12. The synthesized results of 2-digit BCD multiplier showed 24.42 ∼ns and 139 slices. Lakshmi et al. [11] developed a BCD multiplier based on vinculum method. The synthesized results were obtained using TSMC 180 nm library and critical path delay for 2-digit multiplier was found to be 8.273 ∼ns and the area consumed in terms LUTs and slices were 430 and 242 respectively. Jaberipuri et al. [12] developed parallel decimal multiplier. The hardware consisted of three circuits for taking care of Partial Product Generation (PPG), Partial Product Reduction (PPR), and final carry-propagating addition. Results were obtained using 130 nm CMOS technology. The delay and area for 16×16 multiplier were found to be 3.71∼ns and 445×725 μm. Castillo et al. [13] developed a new area-efficient 2-stage BCD multiplier based on BCD to binary converter, binary multipliers and binary-BCD converter. These circuits were implemented on Xilinx devices. Similarly, Yogitha et al. [14] developed a binary multiplier based on Urdhva Triyakbhyam (UT) algorithm with compressors. [15]. The design was implemented in spartan-3E FPGA. The critical path delay obtained was ∼32 ns. According to the work done in [16], a set of multiples of the multiplicand are generated in the pre-processing stage and partial products were obtained by appropriate combination of multiples depending upon the value of multiplier digit. Sequential decimal multipliers have also been reported but the multipliers uses parallel operations and shows less delay at the cost of area [17,18].

The proposed work implements BCD to binary converters of different sizes from 2 to 16 digits, which are later used in a case study of 32-bit decimal multiplier based on Urdhva Triyakbhyam (UT) algorithm. A comparative study on BCD converters and multipliers with and without compressor has also been carried out in this work. The organization of this paper is as follows. Section 1 presents introduction and literature to decimal multiplier. Section 2 explains the BCD to binary conversion algorithm followed by its architecture on Section 3. Section 4 explains the multiplier architecture. Section 5 describes the results followed by the conclusion in Section 6.

2. BCD to binary converter algorithm

The conversion of a binary number to BCD and vice versa plays a key role while interfacing the binary operated microprocessor to pass BCD values. The objective of this work is to develop a hardware by minimizing the delay generated by conversion algorithms to an extent where time taken for conversion becomes negligible compared to the time consumed by decimal operation. Algorithms for 8-bit and 16-bit BCD to binary conversion are elaborated below and their software version can be accessed from [19]. In this paper, a method is proposed to design converters using bottom up approach.The converter is converting 16-bit BCD input to 16-bit binary output using a 16-bit BCD to binary converter algorithm which is further scaled down to two 8-bit BCD to binary converters. This algorithm can be extended to any number of digits. Each decimal digit is represented using four bits in BCD.

2.1. Procedure for 8-bit BCD to binary conversion

Step-1: Consider a number BCD 2-digit decimal number (equivalent to binary 8-bit) i.e., from 0 to 99.

Step-2: Separate the individual digits by doing the following operations.,

$$BCD_{Unit} = (BCD\&0X0F)$$

$$BCD_{Ten} = (BCD\&0X0F0)$$

Step-3: For the units place(LSB bits) binary and BCD are the same.

$$BIN_{Unit} = BCD_{Unit}$$

Step-4: When comes to tens place, split the number in ten's place as a sum of number into powers of two

$$10 = 8 + 2$$

Step-5: This can be achieved by appropriate right shifting operation as below

$$BIN_{Ten} = BCD_{Ten} \gg 1 + BCD_{Ten} \gg 3$$

Step-6: Now add the binary formats of the individual digits to get the final result

$$BIN = BIN_{Unit} + BIN_{Ten}$$

2.2. Procedure for 16-bit BCD to binary conversion

The 8-bit BCD to binary algorithm given above works as a base for all the higher order BCD to binary conversions.

Step-1: Consider a 4-digit decimal number (equivalent to 16-bits) i.e., from 0 to 9999.

Step-2: Split the number into two halves where each half represents 2-digit decimal number i.e., from 0 to 99.

$$BIN_{Upper} = (BCD_{Upper}\&0X0FF00 \gg 8)$$

$$BIN_{Lower} = (BCD_{Lower}\&0X0FF)$$

Step-3: Use 8-Bit BCD to binary converters on four digits

$$BIN_{Upper} \overset{8-\text{bit BCD}-\text{binary}}{\longleftarrow} \text{BCD}_{Upper}.$$

$$BIN_{Lower} \overset{8-\text{bit BCD}-\text{binary}}{\longleftarrow} \text{BCD}_{Lower}.$$

Step-3: For the Lower byte, BIN_{Lower} is the same as BCD_{Lower}.

Step-4: Concatenate BCD_{Upper} with 8-bit binary 0.

$$BIN_{Upper} = Concat\,(BCD_{Upper}, 0).$$

Step-5: For the Upper byte, it represents as multiplied by 100 in BCD.

$$BIN_{Upper} = BCD_{Upper} \ll 8.$$

Step-6: Split the number hundred in powers of two 100 = 64 + 32 + 4

Step-7: Right shift the BIN_{Upper} appropriately as below and add all of them together.

$$BIN_{Upper} \times 100 = (BCD_{Upper} \gg 6) + (BCD_{Upper} \gg 3) + (BCD_{Upper} \gg 2).$$

Step-8: Now add the binary formats of the individual digits to get the final result.

$$BIN = BIN_{Upper} \times 100 + BIN_{Lower};$$

3. Proposed BCD to binary converter

A fully parallelized decimal to binary conversion could be implemented by the proposed architecture with the objective to achieve a minimal time delay and resource utilization. The proposed BCD to binary converter architecture by using algorithm is discussed in previous section. The 8-bit BCD circuit is developed with the help of shifting and multi-operand addition, as shown in Fig. 1. The proposed work designs the converters using bottom up approach. To perform a decimal arithmetic using proposed