HBRP PUBLICATION

Journals of Advancement in Electronics Design
Volume 5 Issue 1

Enhancement in Speed of BCD Adder

Akarsh Shrivastava[1]*, K.B. Ramesh[2]
[1]Student, Electronics & Instrumentation Engineering RVCE Bangalore, India
[2]Associate Professor, Department of Electronics & Instrumentation RVCE Bangalore, India
Corresponding Author
E-Mail Id: akarshs.ei20@rvce.edu.in

ABSTRACT

Almost all applications work with decimal data and spend the majority of their time doing so. Software implementation of decimal arithmetic is typically 100 times slower than hardware implementation of binary arithmetic. As a result, hardware with decimal arithmetic functionality is necessary. A high-speed binary coded decimal (BCD) adder is proposed in this study. By enhancing parallelism, the suggested adder improves the delay of BCD addition. Two 4-bit binary adders, a carry network, one AND gate, and one OR gate make up the proposed BCD adder's critical path. The programs for the proposed reduced delay BCD adder and the Conventional BCD adder are created in Verilog to compare delays.

Keywords: *BCD, adders, AND gate*

INTRODUCTION

Since man first learned to count on his ten fingers, humans have selected decimals as their number base for all hand calculations. Although binary has been chosen as the default base for practically all computers due to the storage and speed efficiency of binary hardware, this fact has never changed.

Because of the speed and simplicity of binary arithmetic, designers have favored binary computers, but there is currently a growing demand for hardware support for decimal arithmetic in financial and commercial applications.

The following are the reasons for this: Because most fractional decimal numbers, such as 0.1, cannot be expressed exactly in binary format, binary arithmetic operations employ their approximation representations. Most financial and commercial applications, which demand the accurate representation of decimal numbers, will not tolerate this. There are more decimal data in commercial databases than binary data. As a result, when binary hardware is utilized, decimal data is translated from decimal to binary, and then binary data is converted back to decimal after processing to store the output in decimal format. The conversion between decimal and binary formats, on the other hand, takes far t

oo long. An adder is employed in all arithmetic units, whether binary or decimal. As a result, adders are critical to the Design of a High-Speed BCD Adder 423 system's performance. In this paper, we suggest a reduced delay BCD adder, which has a substantially shorter delay than a typical BCD adder.[1-3]

PREVIOUS WORK

This section provides an overview of a decimal adder that has previously been created. To compare with the decreased delay BCD adder, several previously proposed popular BCD adders are constructed in VHDL and synthesized. The adders' synthesis results and design information are also discussed. The traditional decimal adder is the first adder explored in this research, as indicated in

 All Rights Reserved

the figure. Each decimal digit has two 4-bit binary adders that are connected via detection logic. The binary addition results are produced by the first-level adders.

A carry output is generated if the result is more than 9, and the result of the first level 4-bit adder is adjusted by adding 6. The carry output is also utilized as a carry input for the following digit. The typical decimal adder's main shortcoming is its slow speed, which is due to the fact that all first-level 4-bit adders must wait for a number of 4-bit additions to get the correct carry input.[4-6]

CONVENTION BCD ADDER

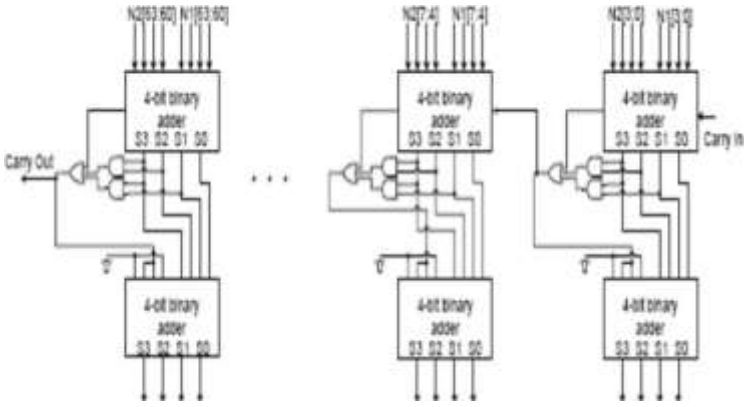
A BCD adder is a circuit that adds two BCD digits and outputs a BCD sum digit. BCD numbers have ten digits, ranging from 0 to 9, which are represented in binary as 0 0 0 0 to 1 0 0 1, i.e. each BCD digit is a four-bit binary number. A BCD number, for example, 5 2 6, can be written as 0101 0010 0110. It's worth noting that BCD can't be higher than 9.

The three situations that occur when two BCD digits are added are the best way to understand the addition of two BCD integers. The BCD addition technique can be summarized as follows by looking at the three situations of BCD addition: 1. using standard binary addition, add two BCD numbers. 2. No correction is required if a four-bit sum is equal to or less than 9. The total is written in BCD format. 3. The sum is invalid if the four-bit sum is more than 9 or if the four-bit sum generates a carry. 4. Add 0110 to the four-bit sum to rectify the erroneous sum. If a carry results from this addition, add it to the next higher-order BCD digit. Thus to implement the BCD adder we require:

- 4-bit binary adder for initial addition
- Logic circuit to detect sum greater than 9 and
- One more 4-bit adder to add 0110 in the sum if the sum is greater than 9 or carry is 1.

By simplifying the Boolean expression of a particular truth table, the logic circuit to detect sums bigger than 9 can be determined. The two BCD values, along with the input carry, are first added in the top 4-bit binary adder to produce a binary sum, as shown in the diagram.

Nothing (zero) is added to the binary total when the output carry is equal to zero (i.e. when Sum = 9 and Cout=0). Binary 0110 is added to the binary total through the bottom 4-bit binary adder when it equals one (i.e. when Sum > 9 or Cout = 1). The bottom binary adder's output carries can be ignored because they provide information that is already available at the output-carry terminal.



HIGH-SPEED ADDER

The conventional BCD adder is fairly simple, but because of the carry ripple effect, it is also very sluggish. When we look at the BCD addition closely, we can see that there are three possibilities:

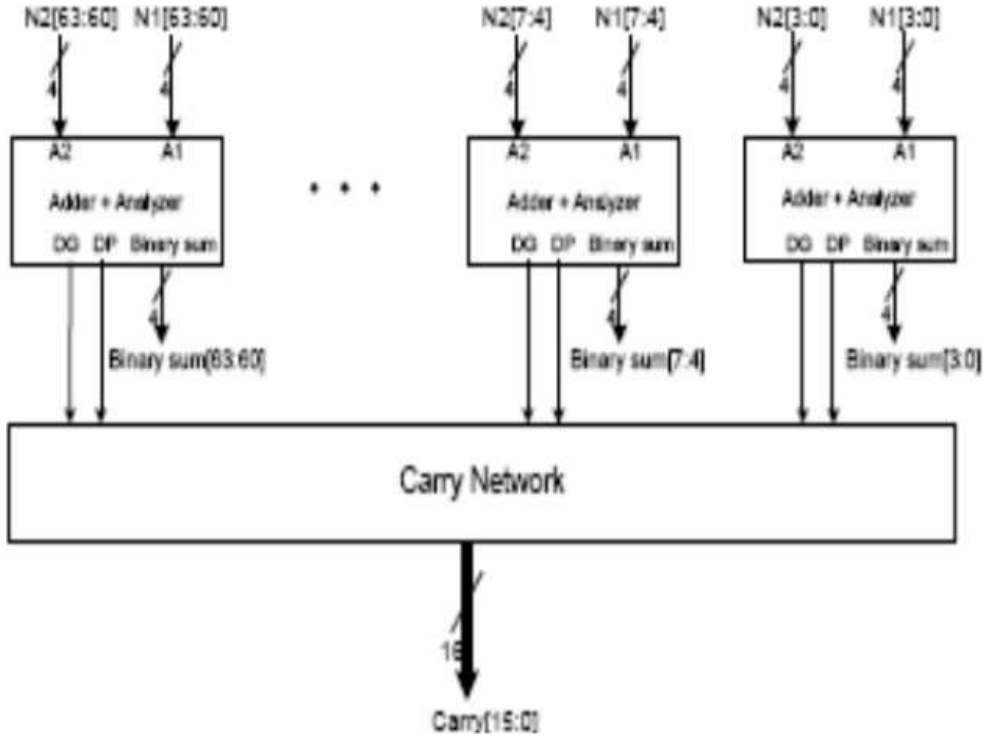
Case 1: The total of two BCD digits is less than nine in case one. Even if there is a carry input, it is assumed that there is no carry output in this scenario. Furthermore, this digit's result does not need to be corrected.

Case 2: The total of two BCD digits exceeds 9. A correction is required in this circumstance. Furthermore, regardless of the carry input, a carry output is generated.

Case 3: The sum of two BCD digits equals nine. The input carry decides whether a correction is needed and whether a carry output is generated in this situation.

The entering carry has no effect on finding the carry output in the first two examples, hence the carry output can be determined without knowing whether or not they carry input exists. If the addition result is 9, on the other hand (Case 3), the input carry defines the existence of the carry output, which may ripple even up to the most significant digit.

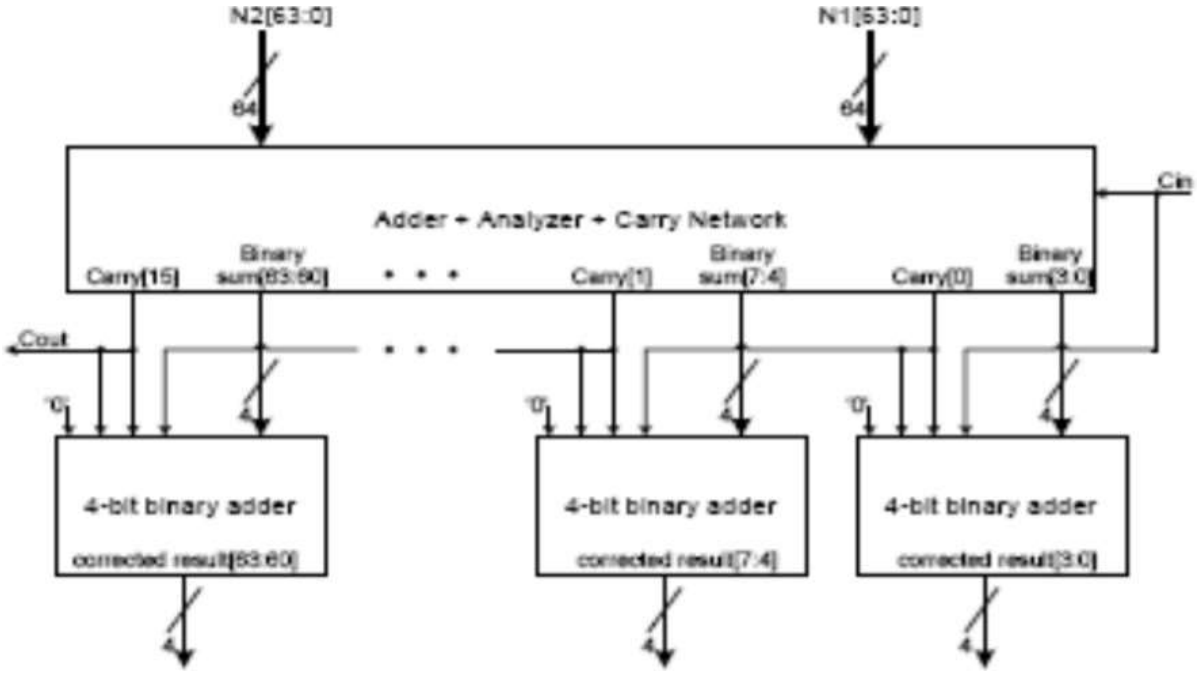
As a result, digit generates (DG) and digit propagate (DP) signals can be used to represent Cases 2 and 3.



In our design, the DG and DP signals of a digit are computed as shown in the diagram. Equation 1 may simply find the output carries for each digit after having all the DG and DP signals. Input Carry=DG+DP Output Carry=DG+DP (1) We can generate DP by ANDing only Sum[0] and Sum[3] instead of using all bits of Sum[3:0] due to the structure of this equation. The DG and DP signals can be used in a binary CLA circuit in the same way that generate and propagate signals are. As a result, any strategy devised for a binary CLA may be utilized to Design a

High-Speed BCD Adder 425 to speed up carry computation. The combination of the first level 4-bit adders and the Carry Network is shown in the figure.

Equation 1 is used to calculate the carrying value for each digit inside the Carry Network. Any type of parallel prefix network or two-level carry look-ahead logic can be employed as the Carry Network. In the rectification step, the carries estimated by carrying networks are employed. The complete BCD adder, including the 4-bit adder, is shown in the diagram.



Correction is done by adding 0, 1, 6, or 7 to the binary sum coming from the first level adder. For each digit, the existence of the output carry and the input carry determine the value to be added for correction.

CONCLUSION

This paper shows how to execute decimal addition with a high-speed BCD adder. When compared to the standard BCD adder, the new suggested adder has the shortest delay.

It also takes up less space on the computer than a traditional BCD adder. By enhancing parallelism, the new decimal adder improves the delay of BCD addition. The suggested BCD adder's critical path comprises the delay of two 4-bit binary adders and a carrier network if we omit one AND gate and one OR gate delay. As a result, the decreased delay BCD adder takes less time to complete 128-bit BCD addition, but the traditional BCD adder takes longer.

REFERENCES

1. Bayrakci, A. A., & Akkas, A. (2007, July). Reduced delay BCD adder. In *2007 IEEE International Conf. on Application-specific Systems, Architectures and Processors (ASAP)* (pp. 266-271). IEEE.
2. Burks, A. W., Goldstine, H. H., & Neumann, J. V. (1982). Preliminary discussion of the logical design of an electronic computing instrument. In

The Origins of Digital Computers (pp. 399-413). Springer, Berlin, Heidelberg.
3. Wakerly, J. F. (2008). *Digital Design: Principles and Practices*, 4/E. Pearson Education India.
4. Schmookler, M. S., & Weinberger, A. (1971). High speed decimal addition. *IEEE Transactions on Computers*, *100*(8), 862-866.
5. Thompson, J., Karra, N., & Schulte, M. J. (2004, February). A 64-bit decimal floating-point adder. In *IEEE Computer Society Annual Symposium on VLSI* (pp. 297-298). IEEE.
6. Shirazi, B., Yun, D. Y., & Zhang, C. N. (1989). RBCD: Redundant binary coded decimal adder. *IEE Proceedings E-Computers and Digital Techniques*, *136*(2), 156-160.