High Precision Binary Coded Decimal (BCD) unit for 128-bit addition

Bayader Abdulrazaaq and Israa Subri
High Precision Binary Coded Decimal (BCD) unit for 128-bit addition

Bayadir A. Issa
Information Technology Department
Southeren Technical University
Basrah Iraq
bayader.phd@gmail.com

Israa S. AL-Furati
Electrical Engineering Department
University of Basrah
Basrah Iraq
israa.subri.1@email.com

Abstract—Financial and business applications utilize decimal information and invest the majority of their energy in decimal number-crunching. Programming usage of decimal number-crunching is common, at any rate, multiple times slower than paired math actualized in equipment. This paper proposes a reduced delay binary coded decimal (BCD) adder that improves BCD addition delay by expanding parallel processing. The ordinary BCD adders are delayed because of the utilization of two binary adders, we structured and executed a new double mode BCD adder which utilizes just a single adder that produced the sum and sum+6. Using a pipeline procedure, an additional 128-bit BCD adder was implemented. The proposed BCD adder was planned and implemented using VHDL with XILINX 9.2 modification. The sequences of the regular BCD adder contrast with the proposed BCD adder. Experimental results show that the proposed BCD adder is 16.7% quicker than a traditional BCD adder. The proposed BCD 128-bit adder is 61.2% faster than the regular BCD 128 adder.

Keywords—Binary coded decimal, Binary adders, VHDL

I. INTRODUCTION

Decimal arithmetic has its wide use in money related and business applications. It assumes a more noteworthy job in these applications due to its abnormal state of accuracy [1]. The decimal number changing is preferred for cash computations over binary arithmetic. Since then, many product frameworks have demonstrated the use of a decimal account in calculations. BCD representation offers one big advantage over binary representation. Switching between decimal and BCD representations is easy. This component deals with fractal quality because fixed and floating-point binary representations cannot transmit a large number of commonly used attributes anywhere in the 0 and 1 range (for example 1/10) sometimes especially useful. Along these lines, BCD tasks can be productive when perusing from a decimal number, and afterward composing the BCD incentive to some other gadget. While performing expansion activity utilizing BCD adder, are delayed because of the utilization of two binary adders so the propagation delay is more [2-4] and the processor speed impacts by the speed of addition activity. So, the performance of the adder must be high to build the exhibition of the processor and the whole system. The delay will influence the speed of the adder which thus influences the speed of the whole system wherein it is utilized. So there is a need to structure the BCD adder with less delay to expand the speed of the activity [5]. In every arithmetic units, regardless of whether binary or decimal, an adder is utilized. In this manner, it isn't astonishing that different expansion methods have been designed up to now, notwithstanding the decimal addition, which is substantially less well known than the binary addition. In this paper, a novel design and simulation of 128-bit binary coded decimal (BCD) adder is proposed and optimized using dual modes adder and a modified (4:2) carry-save adder tree, the BCD adder is designed with less delay and also the proposed BCD adder will be compared with the existing BCD adder in terms of delay (ns). The association of this paper is as follows: In the following section, the prior approaches and their impediments are depicted. In Sect. 3, a novel BCD addition method is proposed. At that point, the development of the BCD adder circuit is given. In Sect. 4, the simulation results and performance analysis of the proposed circuit are clarified. Last, of all, the paper is concluded in Sect. 5.

II. REVIEW OF BINARY CODED DECIMAL ADDER

BCD numbers are a number system that utilized by BCD adder as response and yield which has the binary numbers or digits to characterize the decimal number. A decimal number comprises 10 digits (0-9) [6]. For a 4-bit binary code, there are 16 distinctive binary code mixes together. If the option of two BCD digits surpasses the biggest BCD digit (9)10 = (1001) BCD, it might deliver mistaken BCD yield. For such cases, the outcome must be readdressed by including (6)10 = (0110) BCD to assurance that the outcome is right BCD yield. At that point, the resultant decimal convey yield produced by the alternation procedure is added to the following upper digit of the BCD addends [7]. In a BCD addition, two numbers M and N are provided to the BCD adder structure as shown in Fig.1. With these first ripple additions, it includes adding M and N values with four consecutive complete additions. The modified adder created by the four complete additions is used. When the number exceeds 9, use the setting to include (0110)2 in each addition. The correction rate 0110 is computed by the yield of c + (S [3] • S [2]) + (S [2] • S [1]), but the BCD adder is very simple, with the load ripple effect. Two binary additions were used. The first is used to add an entry and the next is for
the addition of alteration value to the yield of the main binary addition [8-10].

![Typical Binary Coded Decimal Adder (BCD)](image)

In the vast majority of the papers, the prime center has been given to increment the speed of the calculation of the BCD Adder. Designers have proposed a few upgrades to the fundamental BCD addition calculation. In [11] a BCD Adder which is planned dependent on parallel preparing and pipelining at the architectural level to decrease the power utilization. While in [12] a calculation is proposed to accomplish proficient BCD addition/subtraction. The BCD digits are 4 bits long, every one of the tasks, be it BCD addition/subtraction or binary addition/subtraction, are done on 4-bit numbers. The calculation separates the suggested structure into three noteworthy parts, the P-G Block, the Prefix Block, and the Correction Block. [13] proposed structure calculate the input carry by given to the lower part of 4-bit adder rather than to the upper 4-bit adder, which conquers the input carry overhead starting with one digit then onto the next digit. All the n-number of BCD digits are included simultaneously without hanging tight for input carry propagation in the upper part of the design. In [14] a profoundly parallel BCD addition method is proposed with a tree-organized portrayal with a huge decrease in delay. In [15], there are other justifications for incorporating modified bits into a binary sum which is faster than regular additives. Used to increase BCD adder speed. Pipeline strategies also reduce propagation delay by extending the whole. In [16] an enhanced design of BCD adder utilizing quantum-spot cellular automata technology in which they plan a BCD adder using a QCA device. Such contraptions are work with high working rate and ultra-low control utilization. The “9 info and 5 yields” is realized by a rationale circuit. A comparable rationale has been imparted using particular enunciation which would acknowledge the least number of cells similarly as predominant part entryways. It would make most of its yield with the least inertness.

III. PROPOSED 128-BIT BINARY CODED DECIMAL ADDER (BCD)

The essential area of this algorithm is to perform a productive BCD addition with decreased delay and least area. A new design has been proposed for BCD adder better in terms of speed and space compared to traditional BCD additives. The designed BCD adder is shown in Fig.2. The proposed BCD adder utilizes a changed double mode adder which gives the sum and sum_{i+6} to acquire the correct result of the addition in the BCD number system. The strategy for creating the double mode option in [17] is changed in such an approach to permit the two BCD numbers to be included effectively. The double mode adder that generates sum and sum_{i+4} are changed to produce the generates sum and sum_{i+6}. The accompanying advances are disclosed how to decide the two entities:

A. Determine The Carry Propagate and The Carry Generate to Obtain The Sum

\[ pro_i = x_i \oplus y_i \]  \hspace{1cm} (1)

\[ gen_i = x_i \land y_i \]  \hspace{1cm} (2)

\[ N_i = x_i \lor y_i \]  \hspace{1cm} (3)

\[ sum_i = GEN_{i-1} \oplus PRO_i \]  \hspace{1cm} (4)

Where:

\[ GEN_i = \prod_{j=0}^{i} gen_i \]  \hspace{1cm} (5)

\[ N_i = \prod_{j=0}^{i} N_j \]  \hspace{1cm} (6)

\[ CA_i = GEN_{i-1} \lor (N_{i-1} \land C_{in}) \]  \hspace{1cm} (7)

B. To compute the sum_{i+1} Make \ C_{in} = 1

\[ c_{in+1} = GEN_{i-1} \lor N_{i-1} \]  \hspace{1cm} (8)

\[ sum_{i+1} = (GEN_{i-1} \lor N_{i-1}) \oplus PRO_i \]  \hspace{1cm} (9)

C. This Adder Generates The sum and sum_{i+1}

they can be modified to generate the sum and sum_{i+m} where m is a constant. For example: let X is a BCD number to be added. A bit in the position i of X denoted as \( X_i \). Represent the bits string of X in positions I to j. \( X + m \) represent the BCD number generated by the addition of X and the constant m. For example, \( X = 10100011 \) and q=6
then \((X + m) = 10101111\). The modification of the double mode adder is shown in Fig. 3.

D. The modification of double mode adder

The sum + 6 can be generated using the following equations:

\[
\begin{align*}
C_{a+2} &= 1, \quad C_{in+3} = 1 & (10) \\
C_{a+6} &= GEN_{(l-1)+2} \lor N_{(l-1)+2} & (11) \\
C_{a+3} &= GEN_{(l-1)+3} \lor N_{(l-1)+3} & (12) \\
\text{sum} + 6(1) &= C_{a+2} \oplus pro_{l-2} & (13) \\
\text{sum} + 6(2) &= C_{a+3} \oplus pro_{l-3} & (14)
\end{align*}
\]

The sum of the first bit of \(\text{sum} + 6\) of double mode adder is the same in sum.

\[
\text{sum} + 6(0) = \text{sum}(0) \tag{15}
\]

E. The double mode adder generates \(\text{sum}\) and \(\text{sum}+6\) are spilled into blocks and reverse carry is used to generate input carry for each block. The operands are split into a group of bits and the addition is performed in each block.

F. Each block produces \(\text{sum}\) and \(\text{sum}+6\)

One of these outputs (namely; \(\text{sum}\) and \(\text{sum}+6\)) of each block will be selected by the carry using MUX2:1 to obtain the final sum at the output of each block namely \(S_B\).

\[
S_B = \begin{cases} 
\text{sum} & \text{if } C_{out} = 0 \\
\text{sum}+6 & \text{if } C_{out} = 1 
\end{cases}
\]

G. For 128-bit BCD addition

the double mode adder will be spilled into 32 BCD adder of 4-bit as shown in Fig. 4

IV. RESULT AND COMPARTION

Fig. 5 shows the evaluation of the performance of a predictable BCD and adjusted BCD adder and the results of simulation of the proposed 4-bit high-speed BCD adder. It is exposed that this module has the nine input signs as \(X_0, Y_0, X_1, Y_1, X_2, Y_2, X_3, Y_3\) and \(C_{in}\) separately. As a first stage, the yield sign of experiences four clock regions that denote the delay is precisely a full clock cycle. The other five output estimations of BCD addition \(S_0, S_1, S_2, S_3\) \(C_{out}\). The simulation result of 128-bit BCD addition is made as shown in Fig. 6. Also, only the two inputs \(X_i, Y_i\) and the output addition \(\text{sum}_i\) where \(i\) is 128 bits. The suggested double mode BCD adder is compared with the conventional BCD adder and other in [2], [3] in terms of delay (ns) and with the number of logic gates to build the circuits. As exposed in Table 1, the proposed BCD adder for all bit size has a minimum delay and take less area than the others. It’s faster than the others.
Fig. 3. Modified 4-bits double mode BCD adder

Fig. 4. 128-Bit double mode BCD adder
Fig.5. simulation result of the proposed 4-bit double mode BCD adder

Fig.6 simulation result of the proposed 128-bit double mode BCD adder.

TABLE I. THE DELAY, THE NUMBER OF BITS AND THE GATES FOR DIFFERENT TYPES OF ADDERS

<table>
<thead>
<tr>
<th>Adder</th>
<th>No. of bits</th>
<th>Delay (ns)</th>
<th>Area (gates)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conventional BCD Adder</td>
<td>4-bit</td>
<td>9.8</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>128-bit</td>
<td>198.7</td>
<td>1408</td>
</tr>
<tr>
<td>64-bit BCD adder and multiplier in [2]</td>
<td>4-bit</td>
<td>12.35</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>128-bit</td>
<td>201.30</td>
<td>1312</td>
</tr>
<tr>
<td>128-bit BCD adder using CLA adder in [3]</td>
<td>4-bit</td>
<td>12.81</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>128-bit</td>
<td>203.78</td>
<td>1216</td>
</tr>
<tr>
<td>Proposed double mode BCD adder</td>
<td>4-bit</td>
<td>7.67</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>128-bit</td>
<td>165.20</td>
<td>1024</td>
</tr>
</tbody>
</table>

V. CONCLUSION

High-speed low area double mode BCD Adder for 4 bit and 128 bits are designed in this paper. The suggested BCD adder is produced sum and sum+6 and a multiplexer select between them according to the output carry. The additional tools are designed based on low-power tuning technology, i.e. parallel processing and pipeline processing at the architectural level. A comparison using the synthesis results as described and outperformed the proposed BCD Adder other previous designs in terms of region use and delay, the designed adder is synthesized using Xilinx 14.2 ISE EDA tool using VHDL. We compare the performance of a designer with a traditional designer and compare it with simulation and job synthesis [2], [3] using the same tool mentioned above. The designer is then compared in terms of area (ns) and several gates. As shown in table I our proposed BCD adder using a double mode adder for 4 bit and 128 bit has a minimum delay and area compared to conventional BCD adder and in [2], [3].

REFERENCES


Jain, R., Singh, K., and Jangid, “A 64 Bit Pipeline Based Decimal Adder Using a New High Speed BCD Adder,”


