A Cost-Efficient Look-Up Table Based Binary Coded Decimal Adder Design

Zarrin Tasnim Sworna*, Mubin Ul Haque *, Hafiz Md. Hasan Babu* and Lafifa Jamal*
*Department of Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh.
Email:mubin10haque@gmail.com; sworna07@yahoo.com; hafizbabu@cse.univdhaka.edu; lafifa@yahoo.com
Corresponding author’s email: hafizbabu@cse.univdhaka.edu, hafizbabu@du.ac.bd

Abstract—The BCD (Binary Coded Decimal) being the more accurate and human-readable representation with ease of conversion, is prevailing in the computing and electronic communication. In this paper, a tree-structured parallel BCD addition algorithm is proposed with the reduced time complexity $O(N\log_2 b + (N - 1))$, where $N$ = number of digits and $b$ = number of bits in a digit. BCD adder is more effective with a LUT (Look-Up Table)-based design, due to FPGA (Field Programmable Gate Array) technology’s enumerable benefits and applications. A size-minimal and depth-minimal LUT-based BCD adder circuit construction is the main contribution of this paper. The proposed parallel BCD adder gains a radical achievement compared to the existing best-known LUT-based BCD adders. The proposed BCD Adder provides prominent better performance with 20.0% reduction in area and 41.32% reduction in delay for the post-layout simulation. Since the proposed circuit is improved both in area and delay parameter, it is 53.06% efficient in terms of area-delay product compared to the best known existing BCD adder, which is surely a significant achievement.

Keywords—Adder ; BCD ; FPGA ; LUT ; Correction

I. INTRODUCTION

BCD (Binary Coded Decimal) representation is advantageous due to its finite place value representation, rounding, easy scaling by a factor of 10, simple alignment and conversion to character form [1]. It is highly used in embedded applications, digital communication and financial calculations [3] [4]. Hence, faster and efficient BCD addition method is desired. In this paper, a $N$-digit addition method is proposed which omits the complex manipulation steps, reducing area and delay of the circuit. The application of FPGA in cryptography, NP (Non Polynomial)-Hard optimization problems, pattern matching, bioinformatics, floating point arithmetic, molecular dynamics is increasing radically [5] [6] [7]. Due to re-configurable capabilities, FPGA implementation of BCD addition is of concern. LUT being one of the main components of FPGA, a LUT-based adder circuit is proposed.

Two main contributions are addressed in this paper. Firstly, a new tree-based parallel BCD addition algorithm is presented. Secondly, a compact and high-speed BCD adder circuit with an improvement in time complexity of $O(N\log_2 b + (N - 1))$ is proposed, where $N$ represents the number of digits and $b$ represents the number of bits in a digit.

The organization of this paper is as follows: In the next section, the earlier approaches and their limitations are described. In Section [II] a novel BCD addition method is proposed. Then, the construction of BCD adder circuit is given. In Section [IV] the simulation results and performance analysis of the proposed circuit are elucidated. Last of all, the paper is concluded in Section [V].

II. LITERATURE OVERVIEW

In this section, various types of the latest existing LUT-based BCD adders are presented.

A. Existing LUT-Based BCD Adders

BCD adder uses BCD numbers as input and output [8]. Since a 4-bit binary code has 16 different binary combinations, the addition of two BCD digits may produce incorrect result that exceeds the largest BCD digit $(9)_{10} = (1001)_{BCD}$ [9] [10]. In such cases, the result must be corrected by adding $(6)_{10} = (0110)_{BCD}$ to guarantee that the result is a BCD digit. The resultant decimal carry output generated by the correction process is added to the next higher digit of the BCD addends.

Authors in paper [9] proposed a direct implementation of BCD adder circuit. They had proposed two different architectures for the construction of LUT-based BCD adder circuit. A truth table had been formed for each input/output combination and the corresponding circuit was proposed in first architecture. It consumed eleven 6-input LUTs. Two level of abstraction was performed for the second architecture. First least two significant input bits were fed into the first level of circuits whereas the rest input bits along with the output of the first level were provided to the second level of the circuits. The second approach required seven number of 6-input LUTs with a much delay. The direct implementation suffers a significant LUT-delay product.

The BCD adder proposed in [10] used Virtex-6 platform to implement their circuit architecture which had been proposed earlier in [11]. Gao et al. proposed a BCD adder, where the first bit of the addends are added using a full adder and the most significant three bits are added using 6-input LUTs [11]. A correction is ensured in 6-input LUTs by adding $(3)_{10}$ to the sum if the sum of the most significant three bits is greater than or equals to $(5)_{10}$. Moreover, extra circuits are required, when the sum of the most significant three bits is $(4)_{10}$ and the carry generated from the full adder is one. The circuit being serial in architecture except the LUTs portions, requires much time.
The proposed BCD addition method has mainly two steps: highly parallel BCD addition method is proposed with a tree-structure which needs to be removed for faster BCD addition. In this paper, a new LUT-based BCD adder is constructed.

Authors in [13] proposed a BCD addition method, where six is added as a correction factor, when the sum of \( A^i_U + B^i_U \) equals or greater than 8, where \( A^i_U \) and \( B^i_U \) represent the most significant three bits of the input operands \( A \) and \( B \), respectively. If the final output is (111), then a replacement of (111) with (100) is required, as a final step for the exact BCD output. Vazquez et al. presented various carry chain BCD addition methods and their implementations on the LUT architecture [14]. As the carry-chain mechanisms being serial in architecture, the proposed methods in [14] require much delay which are surely a huge drawback.

A power and area-efficient BCD adder circuit was proposed by the authors in paper [15]. They actually used the circuit architecture exhibited in [1] and estimate the power consumption of the circuit. The delay has been calculated on a Virtex-5 platform by using 6-input LUT and the value obtained was 6.22 ns. The average power consumption of the circuit described in [15] was 25 mW which achieved a significant improvement over conventional LUT-based BCD adder. However, the method proposed in [15] required a total of 48 logic elements which can be optimized further.

### III. PROPOSED DESIGN OF LUT-BASED BCD ADDER

In this section, firstly a BCD addition algorithm is proposed. Then a new LUT-based BCD adder is constructed. Essential figures and lemmas are presented to clarify the proposed ideas.

#### A. Proposed Parallel BCD Addition Method

The carry propagation is the main cause of delay of BCD adder circuit, which gives BCD adder a serial architecture. As the reduction of delay is one of the most important factors for the efficiency of the circuit, carry propagation mechanism needs to be removed for faster BCD addition. In this paper, a highly parallel BCD addition method is proposed with a tree-structured representation with significant reduction of delay. The proposed BCD addition method has mainly two steps which are as follows:

- **Bit-wise addition of the BCD addends produce the corresponding sum and carry in parallel. For the addition of first bit, the carry from the previous digit will be added too and the produced sum will be the direct first bit of the output.**

- **If the most significant carry bit is zero then, except the first sum and last carry bit, add the other sum and carry bits in pair in parallel; and if the sum is greater than or equals to five, add three to the result to obtain the correct BCD output.**

#### If the most significant carry bit is one then, update the final output values according to Equation 1 and 2.

Suppose, \( A \) and \( B \) be the two addends of a 1-digit BCD adder, where BCD representations of \( A \) and \( B \) are \( A_3A_2A_1 \) and \( B_3B_2B_1 \), respectively. The output of the adder will be a 5-bit binary number \( \{C_{out} S_3 S_2 S_1 S_0\} \), where \( C_{out} \) represents the position of tens digit and \( \{S_3 S_2 S_1 S_0\} \) symbolizes unit digit of BCD sum. \( A_0 \) and \( B_0 \) are added along with \( C_{in} \) which is the carry from the previous digit addition. If it is the first digit addition, the carry will be considered as zero. The produced sum bit will be the direct first bit of the output. Other pairwise bits \( (B_1, A_1), (B_2, A_2), (B_3, A_3) \) will be added simultaneously. The resultant sum and carry bits \( (S^0_3, C_2, S^0_2, C_1, S^0_1 \) and \( C_0 \)) are added pairwise providing output \( \{C_{out} S^3_3 S^2_2 S^1_1 S^0_0\} \) and corrected by addition of three according to the following Equation 1 and Equation 2.

\[
C^i_{out} S^i_3 S^i_2 S^i_1 S^i_0 = \begin{cases} 
(C^i_{out} S^i_3 S^i_2 S^i_1), & \text{if } C_3 = 0 \text{ and } C^i_{out} S^i_3 S^i_2 S^i_1 < 5 \\
(C^i_{out} S^i_3 S^i_2 S^i_1) + 3, & \text{if } C_3 = 0 \text{ and } C^i_{out} S^i_3 S^i_2 S^i_1 \geq 5 \\
1C^i_0 S^i_3 S^i_2 S^i_1, & \text{otherwise} 
\end{cases}
\]

(1)

where \( S^i_1 = S^i_2 = 0, \text{ if } C_0 = 1 \), \( S^i_2 = 1, \text{ otherwise} \).

In Table II, the truth table is designed with \( (A_3, A_2, A_1) \) and \( (B_3, B_2, B_1) \) as input and \( (C_{out} S_3 S_2 S_1) \) as the final BCD output by following required correction. \( (S^0_3, C_2, S^0_2, C_1, S^0_1 \) and \( C_0 \)) are added pairwise as intermediate step, producing \( (F_3, F_2, F_1) \) by considering carry \( C_0 \) always 1. A numeric 3\((0111)\) is added to the intermediary output \( F \), if \( F \) is greater than or equals to five. A similar table considering \( C_0 \) as 0 can be calculated which is shown in Table II. The truth tables verify the functions of each output of the LUTs of the BCD adder. The algorithm of \( N \)-digit BCD addition method is presented in Algorithm I.

Two example of BCD addition method using the proposed algorithm is demonstrated in Fig. 1 and 2 where \( C^i_3 = 0 \) and \( C^i_4 = 1 \), respectively. Each step of the example is mapped to the corresponding algorithm step for more clarification.

The proposed BCD addition method can be represented as a tree-structure as it is parallel which is shown in Fig. 2. There are basically two operational levels of the tree. Starting from the inputs, in level 1, the bit-wise addition is performed and the intermediary resultants are obtained. Then, in level 2, the addition and correction are performed providing the correct BCD output. Hence, the time complexity of the proposed algorithm is logarithmic according to the operational depth of the tree. Lemma 3.1 is given to prove the time complexity of our proposed method. The time complexity of existing and proposed BCD adders are elucidated in Table II.

**Lemma 3.1** The proposed BCD addition algorithm requires at least \( O(N(\log_2 b) + (N - 1)) \) of time complexity, where \( N \) is number of BCD digits and \( b \) is the number of bits in a digit.
TABLE I: The Truth Table of 1-Digit BCD Addition with $C_0 = 1$

| $B(3 : 1)$ | $A(3 : 1)$ | $S^v(3 : 1)$ | $C_i(3 : 1)$ | $C_0$ | $F(4 : 1)$ | Add 3 | $C_{out}$ | $S_3$ | $S_2$ | $S_1$ |
|------------|------------|--------------|--------------|-------|-------------|-------|-----------|-------|-------|-------|
| 000        | 001        | 000          | 000          | 0     | 0010        | 1     | 0011      | 0     | 0     | 0     |
| 000        | 010        | 010          | 000          | 1     | 0100        | 1     | 0011      | 0     | 0     | 0     |
| 000        | 011        | 010          | 000          | 1     | 0100        | 1     | 0011      | 0     | 0     | 0     |
| 001        | 001        | 000          | 001          | 1     | 0011        | 0     | 0010      | 0     | 0     | 0     |
| 001        | 010        | 010          | 000          | 1     | 0100        | 1     | 0011      | 0     | 0     | 0     |
| 001        | 011        | 010          | 000          | 1     | 0100        | 1     | 0011      | 0     | 0     | 0     |
| 000        | 001        | 100          | 100          | 1     | 1001        | 1     | 0010      | 0     | 0     | 0     |
| 000        | 010        | 110          | 100          | 1     | 1100        | 1     | 0010      | 0     | 0     | 0     |

*'-' Represents “No correction by adding 3 is required.”

TABLE II: The Truth Table of 1-Digit BCD Addition with $C_0 = 0$

| $B(3 : 1)$ | $A(3 : 1)$ | $S^v(3 : 1)$ | $C_i(3 : 1)$ | $C_0$ | $F(4 : 1)$ | Add 3 | $C_{out}$ | $S_3$ | $S_2$ | $S_1$ |
|------------|------------|--------------|--------------|-------|-------------|-------|-----------|-------|-------|-------|
| 000        | 001        | 000          | 000          | 0     | 0001        |       | 0010      | 0     | 0     | 1     |
| 000        | 010        | 100          | 100          | 0     | 0100        | 1     | 0010      | 0     | 1     | 0     |
| 000        | 011        | 100          | 100          | 0     | 0100        | 1     | 0010      | 0     | 1     | 0     |
| 000        | 011        | 000          | 001          | 0     | 0011        | 0     | 0010      | 0     | 1     | 0     |
| 000        | 011        | 000          | 001          | 0     | 0011        | 0     | 0010      | 0     | 1     | 0     |
| 000        | 100        | 000          | 100          | 1     | 1001        | 1     | 0010      | 0     | 1     | 0     |
| 000        | 100        | 000          | 100          | 1     | 1001        | 1     | 0010      | 0     | 1     | 0     |
| 000        | 100        | 000          | 100          | 1     | 1001        | 1     | 0010      | 0     | 1     | 0     |

*'-' Represents “No correction by adding 3 is required.”

---

### Variable | BCD Value | Algorithm Steps
--- | --- | ---
$A (A_3 A_2 A_1 A_0)$ | 1001 | Add $(A_0 A_0, C_i), (A_1 B_1), (A_2 B_2), (A_3 B_3)$ in parallel
$B (B_3 B_2 B_1 B_0)$ | 0101 |  
$C_i$ | 0 |  

### Algorithm Steps
- If $C_i^v = 0$, then $C_i^v = C_i$.
- If $S_{i+1}^v S_i^v > 0$, then $C_i^v = C_i^v + 1$.
- If $S_i^v > 0$, then $C_i^v = C_i^v + 1$.

Fig. 1: Example Demonstration of the Proposed BCD Addition Algorithm for $C_i^v = 0$.

---

### Variable | BCD Value | Algorithm Steps
--- | --- | ---
$A (A_3 A_2 A_1 A_0)$ | 1000 | Add $(A_0 A_0, C_i), (A_1 B_1), (A_2 B_2), (A_3 B_3)$ in parallel
$B (B_3 B_2 B_1 B_0)$ | 1001 |  
$C_i$ | 0 |  

### Algorithm Steps
- If $C_i^v = 0$, then $C_i^v = C_i$.
- If $S_{i+1}^v S_i^v > 0$, then $C_i^v = C_i^v + 1$.
- If $S_i^v > 0$, then $C_i^v = C_i^v + 1$.

Fig. 2: Example Demonstration of the Proposed BCD Addition Algorithm for $C_i^v = 1$. 
Parallel BCD Addition

Input: Two $N$-digit BCD numbers

$$A = A^N \ldots A^3 A^2 A^1$$
and $B = B^N \ldots B^3 B^2 B^1$
where $A^i = A_i A_{i-1} A_{i-2}$ and

$$B^i = B_i B_{i-1} B_{i-2}$$
with $i = 1, 2, 3, \ldots, N$.

Output: Sum, $S = S^N \ldots S^3 S^2 S^1$ where

$$S^i = S_i^3 S_i^2 S_i^1 S_i^0$$
with $i = 1, 2, 3, \ldots, N$ and

Carry, $C = C^N$.

Algorithm 1: Proposed Algorithm for an $N$-digit Parallel BCD Addition

| Input          | Time Complexity |
|----------------|-----------------|
| $A = A^N \ldots A^3 A^2 A^1$ and $B = B^N \ldots B^3 B^2 B^1$ | $O(N \log b + (N - 1))$ |

Proof

The proposed BCD addition algorithm being parallel, can be represented as a tree structure where addends are the root node of the tree $u$ and child nodes $v$ are direct logic implementation circuits, addition with 3-correction logic circuits as well as the output selection circuits.

So, a directed graph $G = (V, E)$ can be constructed where,

$$V \in \{u, v_1, v_2, \ldots, v_n\}$$
and

$$E \in \{(u, v_1), (u, v_2), (v_1, v_2), \ldots, (v_{2n})\}.$$
d(y, z) ≥ d(y, v)

But according to Dijkstra algorithm,

\[ d(u, z) = d(u, y) + d(y, z) ≥ d(u, y) + d(y, v) > d(u, z) + d(z, v) = d(u, v) \]

Hence, the assumption that \( d(u, v) \) is the diameter is contradicted. In each case, we have seen that there is a contradiction if \( z \) is not one of \( u \) or \( v \). Hence it follows that \( z \), the furthest vertex from \( w \), is either \( u \) or \( v \). So, it is proved that the furthest vertex from \( u \) is \( v \). Hence, while calculating the distance using DFS algorithm, we actually find the diameter of the tree in the second run of DFS. Since the diameter is unique, the cost of traversing from \( v_1 \) to \( v_2 \) is \( \log_2 b \). For a \( N \)-digit BCD adder, the time complexity becomes \( O(N(\log_2 b) + (N - 1)) \).

### B. Proposed Parallel BCD Adder Circuit Using LUT

A LUT-based BCD adder is designed by using the proposed BCD addition algorithm and LUT architecture. An algorithm for the construction of proposed BCD adder circuit is presented in Algorithm 2. According to the algorithm, the circuit is depicted in Fig. 4. For the addition of the least significant bit with carry from the previous digit addition, a full adder is used. Three half-adders are used for individual bit-wise addition operation of the most significant three bits. Depending on the value of \( C_3 \), Equation 1 and Equation 2 are followed in the proposed circuit architecture by using the transistors and LUTs, where four number of 6-input LUTs are used to add the output from the half-adders and full adder \( \{S_3, ..., S_1, C_0\} \) with the correction by adding 3, if the sum is greater than or equals to five. Depending on the value of \( C_3 \), a switching circuit is used to follow Equation 3. The proposed circuit gains huge delay reduction due to its parallel working mechanism compared to existing BCD adder circuits.

By using the proposed 1-digit BCD adder circuit, we can easily create an \( N \)-digit BCD adder circuit, where the \( C_{out} \) of one digit adder circuit is sent to the next digit of the BCD adder circuit as a \( C_{in} \). Therefore, the generalized \( N \)-digit BCD adder computes sequentially by using the previous carry, the block diagram of which is shown in Fig. 5.

\[
C_{out}S_3S_2S_1 = \begin{cases} 
C_{out}^\beta S_3^\beta S_2^\beta S_1^\beta, & \text{if } C_3 = 1 \\
C_{out}^\alpha S_3^\alpha S_2^\alpha S_1^\alpha, & \text{otherwise} 
\end{cases} \tag{3}
\]

### IV. Simulation Results and Performance Analysis

As the BCD adder circuits being compared contain different types of logic gates and logic modules, it is better to preserve the basic modules as described in the architectures as long as they correspond to the commonly available cells in a typical standard cell library. The area and delay of the proposed BCD adder circuits are derived and expressed in terms of the area and critical path delay of the basic logic modules that can be found in a typical standard cell library for different operator sizes. These theoretical estimates are then calibrated by the basic logic modules from CMOS 45 nm open cell library [16]. Table IV shows the area and critical path delay of basic logic gates. In this table, we have taken the core logic gates such as inverter, 2-input AND, OR and EX-OR gates. Table VI calculates the area and critical path delay of some logic modules such as full adder, half adder and multiplexer by using the Table IV. It is required to mention that, the area has been calculated in terms of number of transistors.

The area complexity of the proposed BCD adder is derived from its basic logic modules. The proposed BCD adder requires three half adders, one full adder, four 6-input LUTs, six inverters and twenty six transistors. Thus, the total area of the proposed BCD adder (\( A_{\text{proposed}} \)) can be determined as

### Algorithm 2: Proposed Algorithm for the Construction of an 1-Digit BCD Adder Circuit

| Input: | Two 1-digit BCD numbers \( A = \{A_3A_2A_1A_0\} \) and \( B = \{B_3B_2B_1B_0\} \); |
| --- | --- |
| Output: | Sum \( S = \{S_3S_2S_1S_0\} \) and Carry \( C_{out} \); |
| Apply a full adder circuit where Input:= \( \{A_0, B_0, C_{in}\} \) and Output:= \( \{C_0, S_0\} \); |
| \( i \leftarrow 1 \); |
| repeat |
| Apply a half adder circuit where Input:= \( \{A_i, B_i\} \) and Output:= \( \{C_i, S_i\} \); |
| until \( i = 3 \); |
| if \( (C_3 = 1) \) then |
| \( S_3^d \leftarrow C_0; C_{out}^d \leftarrow 1; \) |
| if \( C_0 = 0 \) then |
| \( S_3^d \leftarrow 1; S_2^d \leftarrow 1; \) |
| else |
| \( S_3^d \leftarrow 0; S_2^d \leftarrow 0; \) |
| end |
| else |
| Apply four 6-input LUTs where each LUT’s Input:= \( \{S_3^d, S_2^d, S_1^d, C_3, C_2, C_1\} \) and combined Output:= \( \{C_{out}^d, S_2^d, S_1^d\} \); |
| end |
| \( j \leftarrow 1 \); |
| repeat |
| Apply a switching circuit where Input:= \( \{S_j^d, S_1^d\} \) and Output:= \( \{S_j^d\} \); |
| until \( j = 3 \); |
| Apply fourth switching circuit where Input:= \( \{C_{out}^d, C_{out}^d\} \) and Output:= \( \{C_{out}^d\} \); |

### Table IV: Area and Critical Path Delay of Basic Logic Gates

| Basic Logic Gates | Area (in transistors) | Critical Path Delay (ns) |
| --- | --- | --- |
| Inverter (INV) | 2 | 1 |
| 2-input AND | 6 | 4.68 |
| 2-input OR | 6 | 4.5 |
| 2-input EX-OR | 8 | 4.72 |

### Table V: Area and Critical Path Delay of Basic Logic Modules

| Elements | Area (in transistors) | Critical Path Delay (ns) |
| --- | --- | --- |
| 2-to-1 Multiplexer (MUX) | 20 | 10.18 |
| Half Adder (HA) | 14 | 4.72 |
| Full Adder (FA) | 34 | 11.9 |
Fig. 4: Proposed 1-Digit BCD Adder Circuit.

Fig. 5: Block Diagram of the Proposed N-Digit BCD Adder Circuit.
Specific Integrated Circuits (ASICs) and fixed microprocessors can be a wild, exploratory idea to a viable alternative to Application-Specific Integrated Circuits (ASICs) and fixed microprocessors. The proposed BCD adder is highly parallel, which mitigates the significant carry propagation delay of addition operation. The proposed BCD adder circuit is not only faster but also area-efficient compared to the existing best known circuit. The pre-layout simulation provides 18.18% and 39.9% efficiency in terms of area and critical path delay reduction, respectively compared to the existing best known BCD adder circuit. The proposed BCD adder circuit is simulated using Xilinx Virtex-6. The correctness and efficiency of the circuit is proved in the proposed section and simulation section using corresponding tables, figures and lemma. It is shown by the comparative analysis that the proposed BCD adder is 20% and 41.3% improved in terms of area and delay, respectively compared to the existing best known adder circuit along with 53.06% improvement in area-delay product. These improvements in FPGA-based BCD addition will consequently influence the advancement in computation and manipulation of decimal digits, as it is more convenient to convert from decimal to BCD than binary. Besides, FPGA implementation will be beneficial to be applied in bit-wise manipulation, private key encryption and decryption acceleration, heavily pipelined and parallel computation of NP-hard problems, automatic target generation and many more applications.

ACKNOWLEDGMENT

Zarrin Tasnim Sworna and Munib Ul Haque have been granted fellowship from the Ministry of Information and Technology, People’s Republic of Bangladesh under the program of higher studies and research with the reference no. 56.00.0000.028.33.058.15-629.

REFERENCES

[1] Al-Khaleel, Osama, Mohammad Al-Khaleel, Zakaria Al-QudahJ, Chris-
tos A. Papachristou, Khaldoon Mhaidat, and Francis G. Wolff. “Fast binary/decimal adder/subtractor with a novel correction-free BCD ad-
dition.” In Electronics, Circuits and Systems (ICECS), 2011 18th IEEE International Conference on, pp. 455-459. IEEE, 2011.
[2] Sundaresan, C., C. V. S. Chaitanya, P. R. Venkateswaran, Somashekara Bhat, and J. Mohan Kumar. “High speed BCD adder.” In Proceedings of the 2011 2nd International Congress on Computer Applications and Computational Science, pp. 113-118. Springer, Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-28308-6_15.
[3] Z. T. Sworna, M. U. Haque and H. M. H. Babu. “A LUT-based matrix multiplication using neural network.” 2016 IEEE International Symposium on Circuits and Systems (ISCAS), Montreal, Canada, 2016, pp. 1982-1985. DOI: 10.1109/ISCAS.2016.7538964.
[4] Z. T. Sworna, M. U. Haque, N. Tara, H. M. H. Babu and A. K. Biswas. “Low-power and area efficient binary coded decimal adder design using a look up table-based field programmable gate array.” IET (The Institution of Engineering and Technology) Circuits, Devices & Systems, 2015, volume: 10, issue: 3, pp. 1-10. DOI: 10.1049/iet-cds.2015.0213.
[5] Pocek, Kenneth, Russell Tessier, and André DeHon. “Birth and adoles-
cence of reconfigurable computing: A survey of the first 20 years of field-
programmable custom computing machines.” In Field-Programmable Custom Computing Machines (FCCM), 2013 IEEE 21st Annual Inter-
national Symposium on, pp. 1-17. IEEE, Seattle, WA, USA, 2013. DOI: 10.1109/FPGA.2013.6882273.
[6] Han, Liu, and Seok-Bum Ko. “High-speed parallel decimal multi-
plication with redundant internal encodings.” IEEE Transactions on Computers 62, no. 5 (2013): 956-968. DOI: 10.1109/TC.2012.35.
TABLE VI: Comparison of Area among the Existing and the Proposed \(N\)-Digit BCD Adders for Pre-Layout Simulation

| Method            | Area Expression                                                                 | Area\* | LUT Count |
|-------------------|---------------------------------------------------------------------------------|--------|-----------|
| Gao et al [10]    | \(N \times (1 \times A_{FA} + 3 \times A_{MUX} + 3 \times A_{E-OR} + 2 \times A_{INV} + 2 \times A_{AND} + 4 \times A_{6-LUT})\) | 132N   | 4N        |
| Bioul et al [12]  | \(N \times (8 \times A_{6-LUT} + 6 \times A_{MUX})\)                            | 120N   | 8N        |
| Vazquez et al [13]| \(N \times (5 \times A_{E-OR} + 4 \times A_{MUX} + 2 \times A_{AND})\)           | 134N   | 5N        |
| Vazquez et al [14]| \(N \times (8 \times A_{E-OR} + 7 \times A_{MUX} + 8 \times A_{6-LUT})\)         | 204N   | 8N        |
| Proposed          | \(N \times (3 \times A_{HA} + 1 \times A_{FA} + 4 \times A_{6-LUT} + 6 \times A_{INV} + 26 \times A_{\text{transistor}})\) | 108N   | 4N        |

*\* Represents “Area has been calculated in terms of transistors.”

TABLE VII: Comparison of Delay among the Existing and the Proposed \(N\)-Digit BCD Adders for Pre-Layout Simulation

| Method            | Delay Expression                                                                 | Critical Path Delay (ns) |
|-------------------|---------------------------------------------------------------------------------|--------------------------|
| Gao et al [10]    | \(N \times (1 \times D_{FA} + 2 \times D_{MUX} + 1 \times D_{E-OR} + 1 \times D_{INV} + 1 \times D_{6-LUT})\) | 69.56N                   |
| Bioul et al [12]  | \(N \times (4 \times D_{6-LUT} + 4 \times D_{MUX})\)                          | 140.72N                  |
| Vazquez et al [13]| \(N \times (1 \times D_{6-LUT} + 4 \times D_{MUX} + 2 \times D_{E-OR} + 1 \times D_{6-LUT})\) | 80.74N                   |
| Vazquez et al [14]| \(N \times (4 \times D_{6-LUT} + 4 \times D_{MUX} + 6 \times D_{E-OR})\)       | 168.64N                  |
| Proposed          | \(N \times (1 \times D_{FA} + 1 \times D_{6-LUT} + 2 \times D_{\text{transistor}})\) | 41.8N                    |

Fig. 6: Simulation Result of BCD Adder with Intermediate Carry \(C_1= 1\).

Fig. 7: Simulation Result of BCD Adder with Intermediate Carry \(C_1= 0\).
Future Technologies Conference (FTC) 2017

Fig. 8: Graphical Analysis of Area of Existing and Proposed BCD Adder Circuits for Post-Layout Simulation.

Fig. 9: Graphical Analysis of Delay of Existing and Proposed BCD Adder Circuits for Post-Layout Simulation.

Fig. 10: Graphical Analysis of Area-Delay Product of Existing and Proposed BCD Adder Circuits.

[7] Ning, Yonghai, Zongqiang Guo, Sen Shen, and Bo Peng. “Design of data acquisition and storage system based on the FPGA.” Procedia Engineering 29 (2012): 2927-2931. Elsevier. DOI: 10.1016/j.proeng.2012.01.416

[8] G. Sutter, E. Todorovich, G. Bioul, M. Vazquez, and J.-P. Deschamps. 2009. “FPGA Implementations of BCD Multipliers.” In International Conference on Reconfigurable Computing and FPGAs, Quintana Roo, Mexico, 2009. ReConFig ‘09, pp. 36–41. DOI: 10.1109/ReConFig.2009.28.

[9] O.D. Al-Khaleel, N.H. Tulic, and K.M. Mhaidat. “FPGA implementation of binary coded decimal digit adders and multipliers.” In 8th International Symposium on Mechatronics and its Applications (ISMA), U.A.E 2012. DOI: 10.1109/ISMA.2012.6215199.

[10] Gao, Shuli, Dhamin Al-Khalili, J. M. Langlois, and Noureddine Chabini. “Efficient Realization of BCD Multipliers Using FPGA.” International Journal of Reconfigurable Computing 2017 (2017). DOI: 10.1155/2017/2410408.

[11] ShuliGao, D. Al-Khalili, and N. Chabini. 2012. “An improved BCD adder using 6-LUT FPGA.” In 10th International Conference on New Circuits and Systems (NEWCAS), 2012 IEEE , pp: 13–16. DOI: 10.1109/NEWCAS.2012.6328944.

[12] G. Bioul, M. Vazquez, J. P. Deschamps, and G. Sutter. 2010. “High-speed FPGA 10’s Complement Adders-subtractors.” Int. J. Reconfig. Comput. 2010, Article 4 (Jan. 2010), DOI: 10.1155/2010/219764.

[13] Alvaro Vazquez and Florent De Dinechin. 2010. “Multi-operand Decimal Adder Trees for FPGAs.” Research Report RR-7420. 20 pages. DOI: hal.inria.fr/inria-00526327.

[14] M. Vazquez, G. Sutter, G. Bioul, and J.P. Deschamps. 2009. “Decimal Adders/Subtractors in FPGA: Efficient 6-input LUT Implementations.” In Reconfigurable Computing and FPGAs, 2009. ReConFig ‘09. International Conference on. 42–47. DOI: 10.1109/ReConFig.2009.29.

[15] Mishra, Shambhavi, and Gaurav Verma. Low power and area efficient implementation of BCD Adder on FPGA.” In Signal Processing and Communication (ICSC), 2013 International Conference on, pp. 461-465. IEEE, Noida, India, 2013. DOI: 10.1109/ICSPCom.2013.6719834.

[16] “CMOS 45 nm Open Cell Library”. Available at http://www.si2.org/openeda.si2.org/projects/nangatelib Last access date: 14 March, 2017.