International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Performance Comparison of 8-Digit BCD Adders using CLA and Brent–Kung Architectures

B.Leela Kumar, B.Yogesh, M.Yamuna Devi, V.Ganesh, M.Durga Prasad

B-Tech in Electronics and Communication Engneering , Department of Electronics and Communication Engneering, Sanketika Institute of Technology and Management

Ms. R.Geethika

Assistant Professor**,** Department of Electronics and Communication Engneering, Sanketika Institute of Technology and Management

Abstract - Binary Coded Decimal (BCD) adders play a crucial role in digital systems requiring precise decimal arithmetic, particularly in financial computing, commercial applications, and modern processors. The efficiency of a BCD adder is strongly influenced by the underlying binary adder architecture used for carry generation and sum computation. In this work, an 8-digit BCD adder is designed and implemented using two different architectures: the Carry Look-Ahead Adder (CLA) and the Brent–Kung Adder (BKA). Both designs are modeled in Verilog Hardware Description Language (HDL) and verified through simulation. The performance of the two architectures is evaluated in terms of speed and power consumption. Synthesis results reveal that the Brent–Kung based BCD adder achieves a propagation delay of 2.43 ns, outperforming the CLA-based design which exhibits a delay of
3.58 ns. Furthermore, the Brent–Kung architecture demonstrates superior power efficiency, consuming 25.083 mW compared to
27.020 mW for the CLA implementation. These findings highlight that the Brent–Kung Adder provides significant improvements in both speed and energy efficiency for multi-digit BCD addition. Consequently, the Brent–Kung based BCD adder is more suitable for high-performance decimal arithmetic applications in modern digital and VLSI systems.

Key Words : BCD addition,CLA,BKA and Verilog HDL

1. INTRODUCTION

In modern digital systems, arithmetic operations form the fundamental building blocks of computing and data processing tasks. Operations such as addition, subtraction, multiplication, and division are widely used in processors, digital signal processing systems, embedded systems, and application-specific integrated circuits. Among these operations, **addition** is the most frequently executed arithmetic operation in digital circuits. Therefore, the design of efficient adders with improved speed, reduced power consumption, and optimized hardware area is an important objective in digital system design and Very Large Scale Integration (VLSI) technology.

Most digital systems perform arithmetic operations using the **binary number system** because it is efficient for hardware implementation. Binary arithmetic simplifies circuit design since digital logic circuits

naturally operate with two logic states, namely logic „0" and logic „1". However, many real- world applications such as financial calculations, banking systems, accounting software, and commercial data processing require **accurate decimal arithmetic**. When decimal numbers are processed directly using binary arithmetic, errors may occur due to repeated conversions between binary and decimal formats. To overcome this issue and maintain accurate decimal representation, **Binary Coded Decimal (BCD)** representation is commonly used.

In this work, both **Brent–Kung based and Carry Look-Ahead based 8-digit BCD adders** are designed and implemented using **Verilog Hardware Description Language (HDL)**. The designs are simulated and synthesized to evaluate their performance in terms of **speed and power consumption**. A comparative analysis is then performed to determine the most efficient architecture for multi-digit BCD addition.The objective of this research is to analyze how different adder architectures influence the performance of decimal arithmetic circuits in digital systems. By comparing the Brent–Kung and Carry Look- Ahead architectures, this study provides insights into the design of efficient BCD adders for **high- performance VLSI applications**.

2. LITERATURE SURVEY

The design of efficient arithmetic circuits has always been a significant research area in digital electronics and Very Large Scale Integration (VLSI) systems. Arithmetic units form the fundamental components of digital processors, embedded systems, and signal processing architectures. Among various arithmetic operations, addition is the most frequently used operation, and therefore the design of efficient adders plays a crucial role in improving the overall performance of digital systems. Researchers have proposed several adder architectures over the years to enhance parameters such as speed, power consumption, and hardware area.Binary Coded Decimal (BCD) arithmetic is widely used in applications that require accurate decimal computations. In many real-world systems such as financial processing units, banking applications, digital calculators, and accounting systems, decimal numbers are processed frequently. When decimal numbers are represented in binary form, conversion between decimal and binary formats may introduce rounding errors and inaccuracies. To avoid such issues, BCD representation is used in digital systems. In BCD representation, each decimal digit is encoded using four binary bits, allowing digital circuits to perform decimal arithmetic operations with improved accuracy.

3. METHODOLOGY

Single Digit BCD Adder using CLA

In the **binary addition stage of the BCD adder is implemented using a 4-bit Carry Look-Ahead Adder**. The CLA performs the addition of two BCD digits along with the carry input. The output of this stage produces a **5-bit intermediate result**, which includes the sum and carry output.After the binary addition stage, the intermediate result is checked to determine whether it falls within the valid BCD range. If the sum exceeds the value **1001 (decimal 9)** or if a carry is generated, the result becomes invalid in BCD format. To correct this condition, a **BCD correction mechanism** is used.

The correction condition is defined as: Correction =

$$\text{Cout} + (S3 \cdot S2) + (S3 \cdot S1)$$

If this condition is satisfied, the correction value **0110 (decimal 6)** is added to the intermediate sum using another CLA adder. This ensures that the final output is a valid BCD digit.

The single-digit BCD adder therefore consists of two major components:

1. **4-bit CLA adder for binary addition**
2. **Correction adder to add 6 when required**

This architecture ensures correct decimal arithmetic while maintaining high-speed operation.

8-Digit BCD Adder Architecture

To support multi-digit decimal arithmetic, the proposed design extends the single-digit BCD adder to an **8-digit BCD adder architecture**. In this structure, eight BCD adder blocks are connected sequentially.Each block processes one decimal digit of the input numbers. The carry output from each stage is passed as the carry input to the next stage. This cascading structure enables the system to perform addition of large decimal numbers represented in BCD format.The inputs to the system are two **32-bit BCD numbers**, where every four bits correspond to one decimal digit. The architecture processes these digits starting from the least significant digit to the most significant digit. The final output is a **32-bit BCD sum along with a carry output**.

Carry Lookahead Adder(CLA)

To reduce the computation time, there are faster ways to add two binary numbers by using carry lookahead adders. They work by creating two signals P and G known to be Carry Propagator and Carry Generator. The carry propagator is propagated to the next level whereas the carry generator is used to generate the output carry ,regardless of input carry. The block diagram of a 4-bit Carry Lookahead Adder is shown here below .

1. Pre processing

This step involves computation of generate and propagate signals corresponding too each pair of bits in A and B.

pi = Ai *xor* Bi gi = Ai *and* Bi

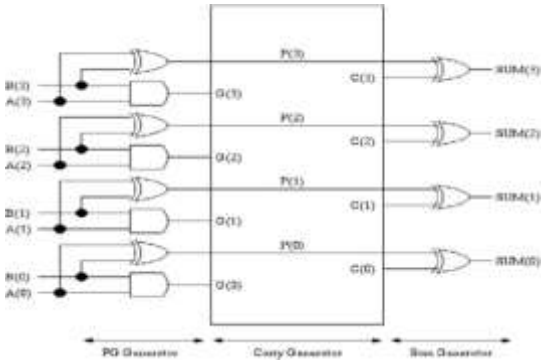

Fig 1: 4bit Carry Lookahead Adder 2 . Carry look

ahead network

This block differentiates KSA from other adders and is the main force behind its high performance. This step involves computation of carries corresponding to each bit . It uses group propagate and generate as intermediate signals .

Gi:j = Gi:k+1 *or* (Pi:k+1 *andGk:j*)

3. Post processing

This is the final step and is common to all adders of this family (carry look ahead). It involves computation of sum bits.

Si = pi xor Ci-1

Despite these limitations, the CLA- based BCD adder provides a reasonable balance between speed and implementation complexity. For this reason, it has been widely used in decimal arithmetic circuits and serves as the **existing reference design** in this work. The performance of this design is later compared with the **proposed Brent–Kung based BCD adder** to evaluate improvements in speed and power consumption.

8-Digit BCD Adder using Brent–Kung Adder (BKA)

To improve the performance of decimal arithmetic circuits, an efficient adder

architecture is required. In this work, the **proposed design** implements an **8-digit BCD adder using the Brent–Kung Adder (BKA)** architecture. The Brent–Kung adder is a type of **parallel prefix adder** that is designed to compute carry signals efficiently using a tree- based structure. This architecture reduces carry propagation delay and minimizes wiring complexity compared to other parallel prefix adders.

In high-speed digital systems, the delay associated with carry propagation is one of the major factors that affects the performance of adders. Traditional adders such as ripple carry adders suffer from large propagation delays because the carry signal must pass through each stage sequentially. Even though Carry Look- Ahead Adders improve the speed by generating carry signals in advance, the complexity of the carry generation logic increases with the number of bits. To address these limitations, parallel prefix adders such as the Brent–Kung adder are used.The prefix network combines these signals to compute carry values efficiently across multiple stages. Compared to other parallel prefix adders, the Brent–Kung architecture provides a balanced trade-off between **speed, hardware complexity, and wiring overhead**.

In the proposed design, the **8-digit BCD adder is constructed by cascading eight single-digit BCD adder modules**. Each module performs BCD addition using a **4-bit Brent–Kung adder** followed by correction logic. The correction logic ensures that the result remains within the valid BCD range.

The operation of each BCD adder stage in the proposed design consists of the following steps:

Binary Addition using Brent–Kung Adder

Two 4-bit BCD digits are added using a **4-bit Brent–Kung adder**. The prefix tree structure computes carry signals in parallel, producing a 4-bit binary sum and a carry output with reduced delay.

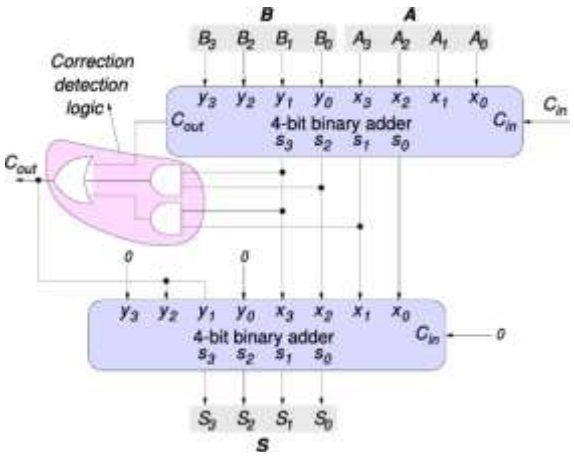

Fig2: 1-digit BCD adder

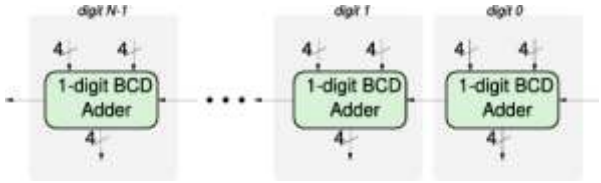

Fig3 : N-digit BCD adder

Detection of Invalid BCD Result

The intermediate binary sum obtained from the Brent–Kung adder is checked to determine whether it represents a valid BCD value. The result becomes invalid if:

- The binary sum is greater than **1001 (decimal 9)**
- A carry is generated during the binary addition

Brent–Kung Adder Architecture

The Brent–Kung adder is a **parallel prefix adder** designed to reduce the logic depth required for carry computation. It uses **generate (G) and propagate (P) signals** to determine carry values efficiently.

The generate and propagate signals are defined as:

- **Generate (G) = A · B**
- **Propagate (P) = A ⊕ B**

Using these signals, the carry output for each stage can be computed using prefix operations. The Brent–Kung structure organizes these operations in a **tree-like prefix network** consisting of multiple levels. This network reduces the number of logic levels required to compute carries compared to ripple carry adders.

1. Pre-possessing stage:- Generate and propagate signals to each pair of the inputs A and B are computed in this stage.

Pi = Ai xor Bi Gi = Ai and Bi

2. Carry generation network:- In this stage, carries equivalent to each bit is calculated. All these operations are implemented and carried out in parallel. Carries in parallel are segmented into smaller pieces after the implementation of the stage. Carry propagate and generate are used as intermediate signals which are given by the logic equations : CPO = Pi and Pj
CGO = (Pi and Gj) or Gi

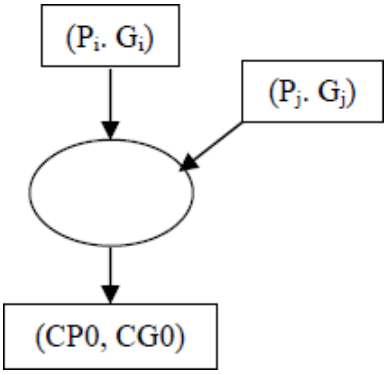

Fig4:Carry network

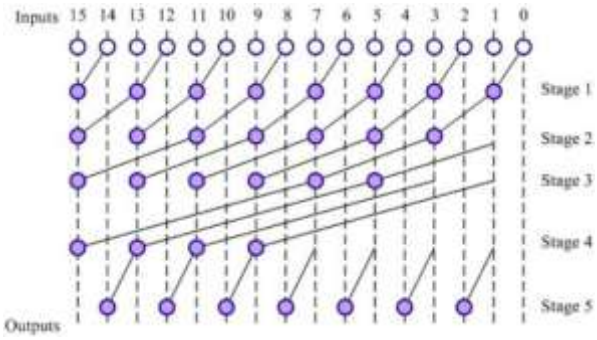

Fig5: carry tree network of brent kung adder

3. Post processing Stage:- This is the concluding step to compute the summation of input bits.

Ci-1= (Pi and Cin) Si =Pi xor Ci-1

4. RESULTS

RTL SCHEMATIC: The RTL schematic is abbreviated as the register transfer level it denotes the blue print of the architecture and is used to verify the designed architecture to the ideal architecture that we are in need of development .The hdl language is used to convert the description or summery of the architecture to the working summery by use of the coding language i.e verilog ,vhdl. The RTL schematic even specifies the internal connection blocks for better analyzing .The figure represented below shows the RTL schematic diagram of the designed architecture.

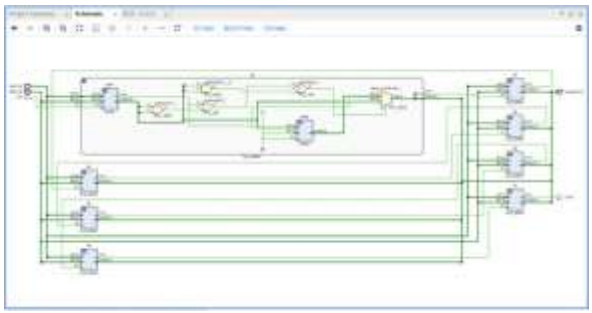

Fig 5: RTL Schematic of BCD adder using CLA

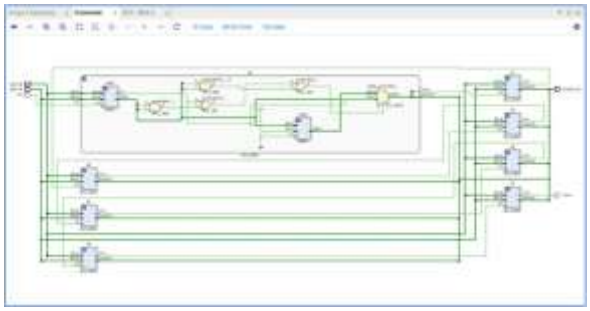

Fig 6: RTL Schematic of BCD adder using BKA

SIMULATION: The simulation is the process which is termed as the final verification in respect to its working where as the schematic is the verification of the connections and blocks. The simulation window is launched as shifting from implantation to the simulation on the home screen of the tool ,and the simulation window confines the output in the form of the wave forms. Here it has the flexibility of providing the different radix number systems.

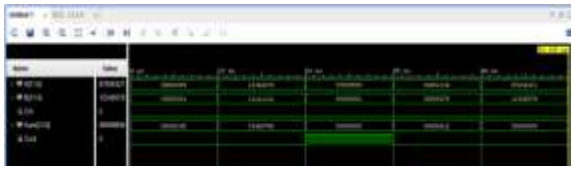

Fig 7: Simulated wave form of BCD adder using CLA

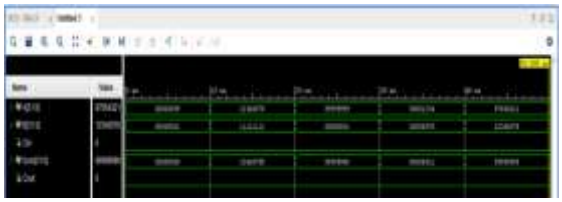

Fig 8: Simulated wave form of BCD adder using BKA

TECHNOLOGY SCHEMATIC: The technology schematic makes the representation of the architecture in the LUT format ,where the LUT is consider as the parameter of area that is used in VLSI to estimate the architecture design
.the LUT is consider as an square unit the memory allocation of the code is represented in there LUT s in FPGA.

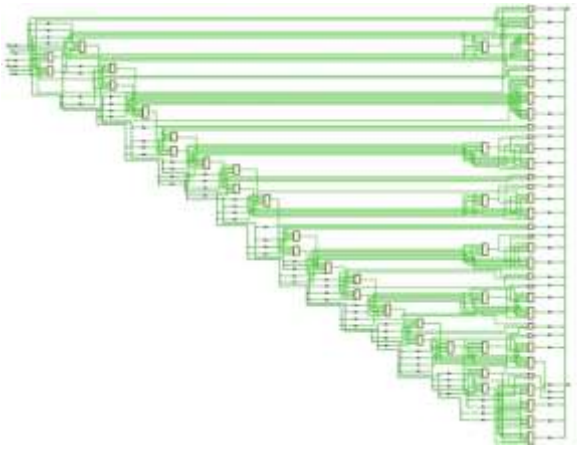

Fig 9: Technology schematic of BCD adder using CLA

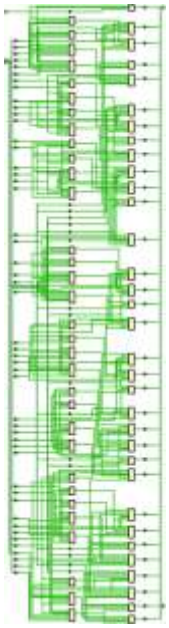

Fig10: Technology schematic of BCD adder using BKA

PARAMETERS: Consider in VLSI the parameters treated are area and power ,based on these parameters one can judge the one architecture to other.

Table 1: Parameter comparison

Parameter	BCD adder using CLA	BCD adder using BKA
Speed (ns)	3.58	2.43
Power(mW)	27.020	25.083

Fig 11: Speed and Power comparison bargraph

3.CONCLUSION AND FUTURE SCOPE

In this work, an **8-digit Binary Coded Decimal (BCD) adder** was designed and implemented to perform accurate decimal arithmetic operations in digital systems. The existing design utilizes the **Brent–Kung Adder (BKA)** architecture to perform binary addition within each BCD digit stage. The Brent–Kung adder, being a parallel prefix adder, reduces carry propagation delay and improves the speed of arithmetic operations compared to conventional ripple carry adders.To evaluate alternative adder architectures, a **Carry Look- Ahead Adder (CLA) based BCD adder** was also designed and implemented. The CLA architecture computes carry signals in advance using propagate and generate logic, which helps reduce sequential carry propagation and improves computational efficiency. Both designs were modeled using **Verilog HDL** and verified through simulation.

The performance of the two designs was analyzed in terms of **speed and power consumption**. The synthesis results indicate that the **BCD adder using Brent–Kung architecture achieves better performance**, with a delay of **2.43 ns** compared to **3.58 ns** for the CLA-based design. Additionally, the Brent– Kung based BCD adder consumes **25.083 mW**, which is lower than the **27.020 mW** consumed by the CLA-based implementation.From the obtained results, it can be concluded that the **Brent–Kung adder provides improved speed and lower power consumption for multi-digit BCD addition** due to its efficient parallel prefix carry computation. Therefore, the Brent–Kung based BCD adder is more suitable for high-performance decimal arithmetic applications in modern digital systems

.

REFERENCES

[1] M. M. Mano and M. D. Ciletti, *Digital Design*, 5th ed. Boston, MA, USA: Pearson, 2013.

[2] N. H. E. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed. Boston, MA, USA: Pearson, 2011.

[3] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Upper Saddle River, NJ, USA: Pearson, 2003.

[4] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, 2nd ed. New York, NY, USA: Oxford University Press, 2010.

[5] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," *IEEE Transactions on Computers*, vol. C-31, no. 3, pp. 260–264, Mar. 1982.

[6] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Transactions on Computers*, vol. C-22, no. 8, pp. 786–793, Aug. 1973.

[7] R. Zimmermann, *Binary Adder Architectures for Cell-Based VLSI and their Synthesis*, Zurich, Switzerland: Swiss Federal Institute of Technology, 1997.

[8] K. Hwang, *Computer Arithmetic: Principles, Architecture, and Design*. New York, NY, USA: Wiley, 1979.

[9] S. Knowles, "A family of adders," in *Proc. 15th IEEE Symposium on Computer Arithmetic*, 2001, pp. 277–281.

[10] T. Lynch and E. Swartzlander, "A spanning tree carry lookahead adder," *IEEE Transactions on Computers*, vol. 41, no. 8, pp. 931–939, Aug. 1992.