

Efficient EEG Data Compression Using Cellular Automata: A Rule 90 and Rule 150-Based Approach

B. Sreelatha

Geethanjali College of Engineering Technology

M. Kiran Kumar

kirankumarece@anurag.edu.in

Geethanjali College of Engineering Technology

Case Report

Keywords: Cellular Automata, Electroencephalogram, epileptic seizure, FPGA, low power, biomedical

Posted Date: February 24th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-8938940/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Efficient EEG Data Compression Using Cellular Automata: A Rule 90 and Rule 150-Based Approach

B. Sreelatha^{1,2} and M. Kiran Kumar^{1*}[†]

^{1,2*}Department of ECE, School of Engineering, Anurag University, Hyderabad, Telangana, India.

¹Department of ECE, Geethanjali College of Engineering Technology, Hyderabad, Telangana, India.

*Corresponding author(s). E-mail(s): kirankumarece@anurag.edu.in;

[†]These authors contributed equally to this work.

Abstract

This paper presents an innovative Cellular Automata (CA)-based framework for efficient compression of electroencephalogram (EEG) data, leveraging Rule 90 and Rule 150. These specific CA rules are selected due to their computational efficiency and their ability to preserve critical seizure-related patterns within EEG signals. Rule 90, a linear rule, maintains structural integrity and preserves essential bit transition patterns. In contrast, Rule 150, a chaotic rule, enhances edge detection and local variations, ensuring that diagnostically relevant features remain intact. The proposed approach efficiently compresses 28 input samples into just 4 output samples, achieving an 85% reduction in storage requirements and significantly minimizing memory utilization. This optimization enhances its suitability for large-scale, real-time EEG signal processing applications. The proposed CA-based data compression framework has been efficiently implemented on an Xilinx Zynq 7000 FPGA, demonstrating a total power consumption of 219mW, with 111mW allocated to dynamic power and 108mW to static power. The results indicate that the proposed CA-based compression methodology offers a highly efficient and hardware-optimized solution for EEG data processing, making it well-suited for low-power, embedded, and real-time biomedical applications.

Keywords: Cellular Automata, Electroencephalogram, epileptic seizure, FPGA, low power, biomedical.

1 Introduction

Cellular Automata (CA) are computational models that consist of a grid of cells, each of which can exist in a finite number of states. The state of each cell evolves over discrete time steps based on a predefined set of rules that consider the states of neighboring cells. Initially introduced by John von Neumann in the 1950s for self-replicating systems. [1, 2], CA have since been extensively studied for their ability to model complex systems with simple, local interactions.

One-dimensional CA, particularly elementary cellular automata (ECA), have gained attention for their applications in cryptography, image processing, and data compression [3]. Among these, Rule 90 and Rule 150 are widely used due to their unique structural properties. Rule 90, defined by an XOR operation between neighboring cells, produces self-replicating patterns, making it useful for error detection and data transformation. Rule 150, a more chaotic rule based on XOR of all three neighboring cells, is effective in capturing edge transitions and enhancing feature extraction [4]. CA, particularly Rule 90 and Rule 150, have shown potential in data encoding and pattern generation [5]. However, their application in EEG compression remains unexplored. Due to their parallel processing capability, modular structure, and low hardware complexity, CA is highly suitable for VLSI implementations and real-time processing applications. Their deterministic nature makes them an ideal choice for signal processing tasks such as EEG compression and seizure detection, where maintaining critical structural information is essential.

Memory reduction refers to techniques and strategies designed to minimize the storage and computational resources required for processing data efficiently. As modern applications generate vast amounts of data, optimizing memory usage has become critical for enhancing system performance, reducing hardware costs, and enabling real-time processing. Memory reduction methods often introduce trade-offs in terms of accuracy, computational complexity, or data integrity, as shown in Figure 1. With the exponential growth of data-driven applications such as big data analytics, artificial intelligence (AI), embedded systems, and biomedical signal processing, the need for efficient memory management has intensified. Memory reduction is needed for optimizing storage, improving processing speed, energy efficiency, cost reduction.

Lowering the data footprint allows systems to store more information within limited hardware constraints. Reducing memory usage enhances access times and minimizes computational overhead. Memory optimization is crucial in power-constrained devices such as Internet of Things (IoT) sensors and mobile devices to extend battery life. Efficient memory management reduces the need for high-capacity and expensive storage components.

Memory reduction techniques are widely applied in various domains, such as biomedical signal processing, for efficient compression and processing of EEG and ECG signals for real-time health monitoring. Image and video processing is used in medical imaging, streaming services, and computer vision to reduce storage requirements and embedded systems and IoT for memory optimization in microcontrollers, smart sensors, and edge computing devices. Artificial intelligence and machine learning techniques like model pruning, quantization, and knowledge distillation reduce model size while maintaining accuracy.

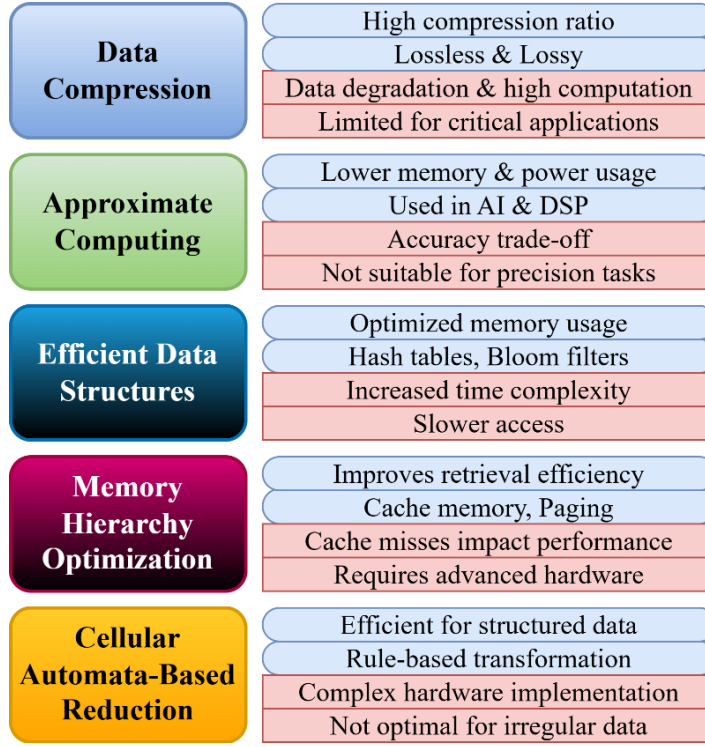


Fig. 1 Memory reduction methods and their tradeoffs

2 Cellular Automata-Based Compression Methodology

CA is a discrete computational model consisting of an array of cells, where each cell updates its state based on a predefined rule and the states of its neighboring cells. It is widely used in data compression, pattern generation, and cryptography due to its simplicity and parallelism.

2.1 Fundamentals of Cellular Automata

2.1.1 Mathematical Representation

Each cell C_i^t at time t is updated at the next time step $t + 1$ based on its state and the states of its left C_{i-1}^t and right C_{i+1}^t neighbors:

$$C_i^{t+1} = f(C_{i-1}^t, C_i^t, C_{i+1}^t) \quad (1)$$

where f is the transition function defined by the Cellular Automaton (CA) rule.

2.1.2 Rule 90 (XOR-Based Linear Rule)

Rule 90 follows an exclusive OR (XOR) operation, making it a linear CA rule. It is defined as:

$$C_i^{t+1} = (C_{i-1}^t \oplus C_{i+1}^t) \quad (2)$$

where $\oplus$ represents the XOR operation. This rule generates self-replicating fractal patterns and is useful in structured data reduction.

2.1.3 Rule 150 (Chaotic Rule for Feature Extraction)

Rule 150 is a non-linear CA rule that introduces chaotic behavior and aids in feature extraction, often used in edge detection applications. It is defined as:

$$C_i^{t+1} = f(C_{i-1}^t \oplus C_i^t \oplus C_{i+1}^t) \quad (3)$$

This rule produces complex and seemingly random patterns, helping in information preservation and enhancement.

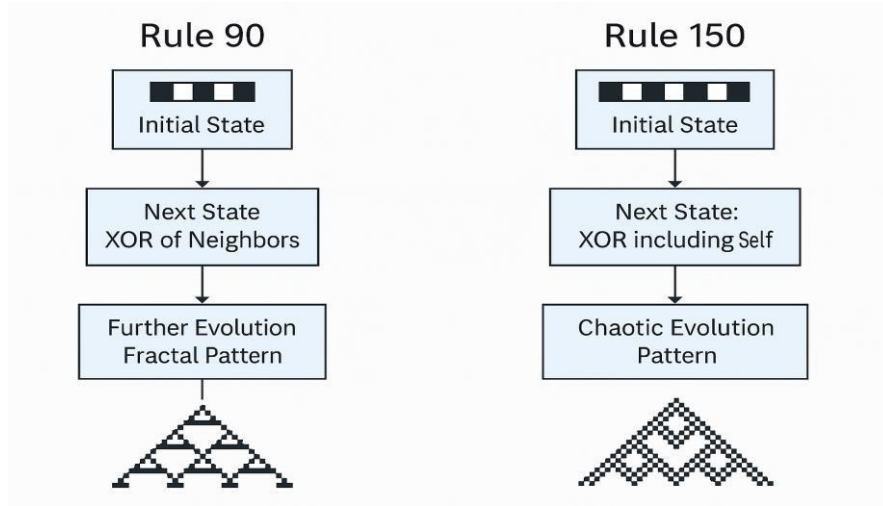


Fig. 2 Rule 90 and Rule 150 Cellular Automata Transformations

2.2 CA-Based Compression Framework

The EEG data compression framework leverages Cellular Automata (CA) to efficiently reduce the size of EEG signals while retaining crucial information. The process consists of the following steps:

2.2.1 Segmentation

The continuous EEG signal is divided into fixed-size blocks, each containing 28 samples of each 10-bits. This segmentation helps in localized pattern recognition and structured data processing. The Fig. 3 shows the organization of a sample-based memory segmentation, where data is stored and accessed in fixed-size of 10-bits. It shows the structure of memory addressing and data alignment for efficient retrieval and processing.

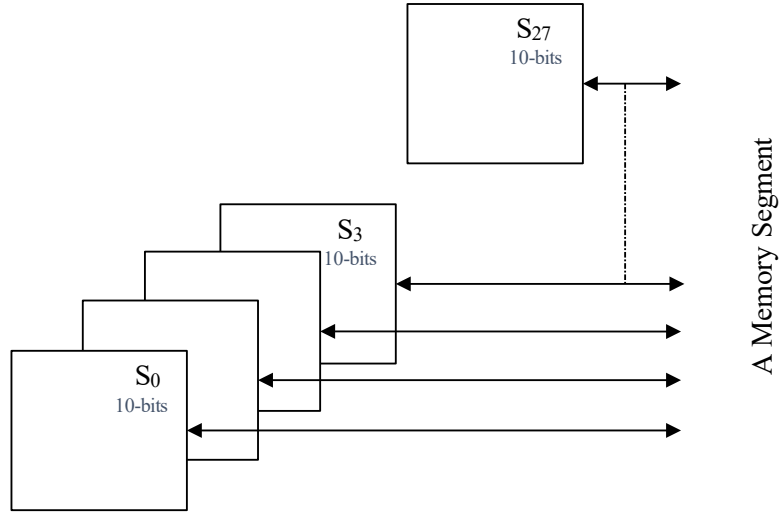


Fig. 3 Structure of a Sample-Based Memory Segmentation

2.2.2 Transformation

Each segmented block undergoes CA-based processing using Rule 90 and Rule 150. Two matrices are considered: a characteristic matrix (T) of size 10×10 , corresponding to a 10-cell maximum length CA, and an identity matrix (I) of the exact dimensions. The rule vector $\langle 150, 150, 90, 150, 90, 150, 90, 150, 90, 150 \rangle$ is applied to individual rows of the identity matrix (I) to compute the characteristic matrix (T). The calculated T matrix is as follows.

$$T = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

2.2.3 Down Sampling

After transformation, the data is down-sampled by compressing 28 samples to 4 samples. The compressed output is further decoded to facilitate transmission or storage, as shown in Fig. 4. The final output retains essential EEG signal features, ensuring that medically relevant patterns are not lost.

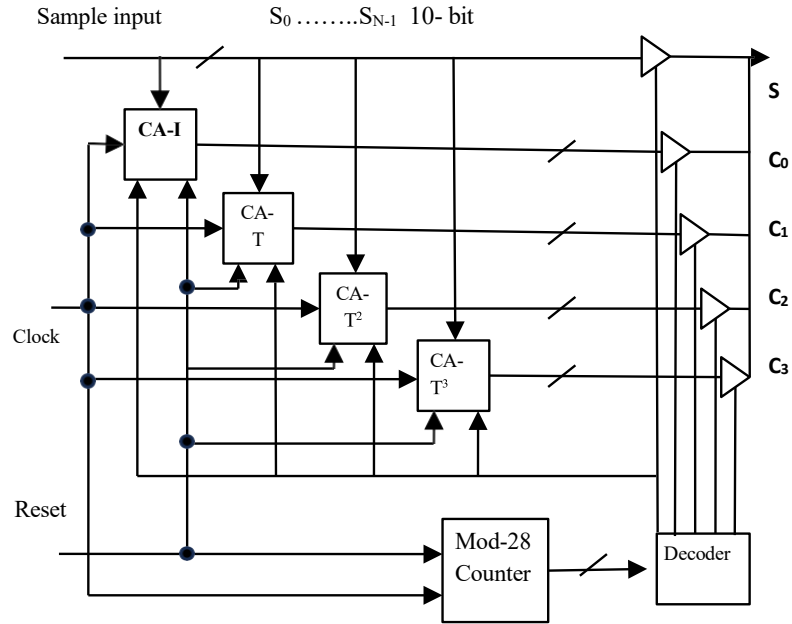


Fig. 4 Proposed CA-Based Compressor

3 FPGA Implementation of CA Based Compressor

The proposed CA-based compression framework is implemented on a Xilinx Zynq 7000 FPGA, leveraging hardware parallelism to achieve real-time processing [6].

3.1 Compression Ratio (CR)

Defined as the ratio of input to output data. A higher CR indicates better compression efficiency, reducing storage requirements and transmission bandwidth without significantly compromising critical EEG signal features.

$$CR = \frac{\text{Original Data Size}}{\text{Compressed Data Size}}$$

The schematic of proposed CA based compressor using digital logic components is shown in Figure 5. It consists of Clock and Control Signals, Registers, Counters, Comparators, Multiplexers and ROM Blocks. The clk signal synchronizes the operations of the registers and logic components. The load signal is used to control when new data is loaded into the registers. Multiple synchronous registers that store intermediate and final values. These registers update based on the clock signal conditions. The count0.i performs an addition operation to update a counter. count0.i 0 (RTL GT) checks if the count has exceeded 28 value ($V=B'11100''$) to start the data compression process at this time the output c i.e compute will be high. count1.i checks if the count is below 28, loading the data samples making the output Load high. Several multiplexers select between different inputs based on control conditions i.e load data, compress data and start new data. For example, count i 1 and count i choose between different counter values i.e greater than 28, less than 28 and zero. c i and n_i_0 are used for selecting different values based on compression logic. The ROM blocks, such as count i_0, n_i, and i_0, store sample values. The CA-based compressor follows rule 90 and 150 based transformation, where the c_reg and n_reg store compressed output. The logic elements determine how the input data is mapped to a compressed form, based on a cellular automata model. The load_reg ensures that compression starts at specific times when new data is available. The system processes input data in cycles, updating counters and selecting values based on CA logic. It uses multiplexers and ROM-based lookup tables to transform the input dynamically. The compression happens through structured state transitions controlled by the registers.

The input data is loaded sample by sample with each clock pulse, where each sample is 10 bits in size. A counter is used to track blocks of 28 samples. Fig. 6, 7 illustrate the output waveforms, showing data samples being loaded at the positive edge of each clock pulse.

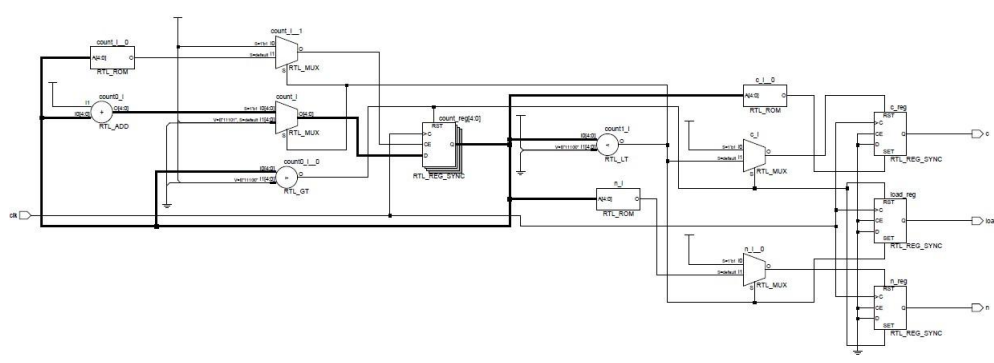


Fig. 5 RTL Schematic of the proposed CA based Compressor

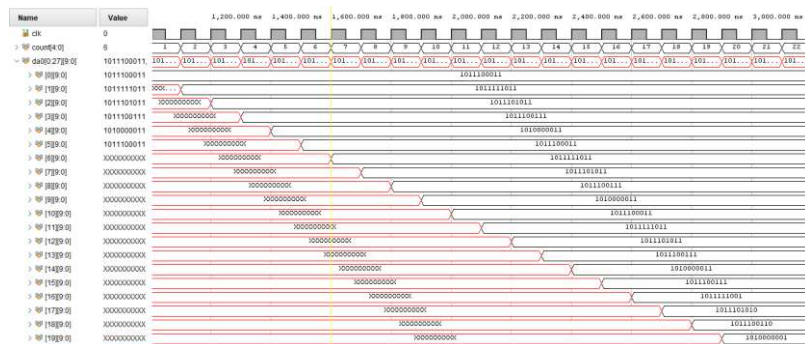


Fig. 6 Input samples loading from 1 to 20

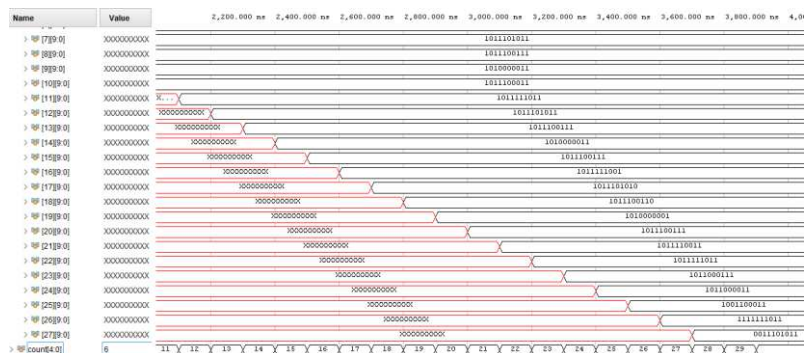


Fig. 7 Input samples loading from 21 to 28

Figure 8 shows the T matrix that is computed by applying CA rules to identity matrix.

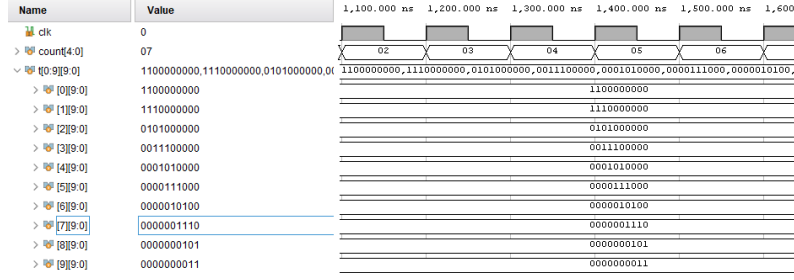


Fig. 8 T matrix output waveform

The compressed data is calculated by

$$C_a = T_a D_{N-1} \oplus T_a^{(2)}[D_{N-2}] \oplus \dots \oplus T_a^{(N)}[D_0] \quad (4)$$

where $0 \leq a \leq 3$, and given by Equations (2)–(5) as follows:

$$C_0 = S_{N-1} \oplus S_{N-2} \oplus \dots \oplus S_0 \quad (5)$$

$$C_1 = T S_{N-1} \oplus T^{(2)}[S_{N-2}] \oplus \dots \oplus T^{(N)}[S_0] \quad (6)$$

$$C_2 = T^2 S_{N-1} \oplus T^{2(2)}[S_{N-2}] \oplus \dots \oplus T^{2(N)}[S_0] \quad (7)$$

$$C_3 = T^3 S_{N-1} \oplus T^{3(2)}[S_{N-2}] \oplus \dots \oplus T^{3(N)}[S_0] \quad (8)$$

The above equations T -matrices for 28 samples computation flowchart is shown in Fig. 9.

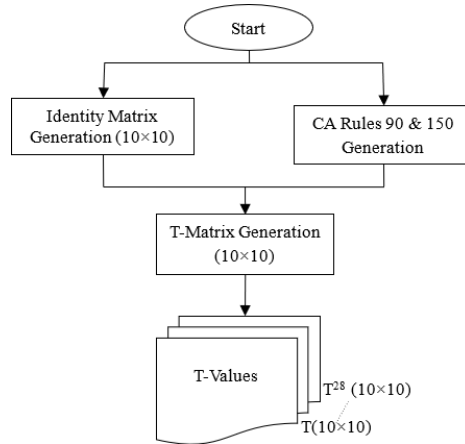


Fig. 9 T matrixes calculation for all 28 samples

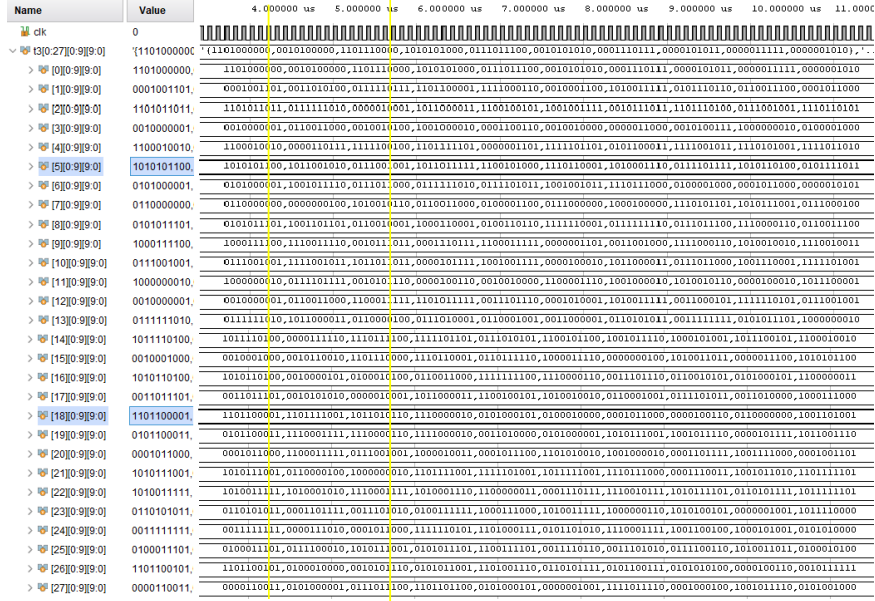


Fig. 12 Matrices T^3 , $T^{3(2)}$, ..., $T^{3(M)}$ for the computation on compressed data C_3

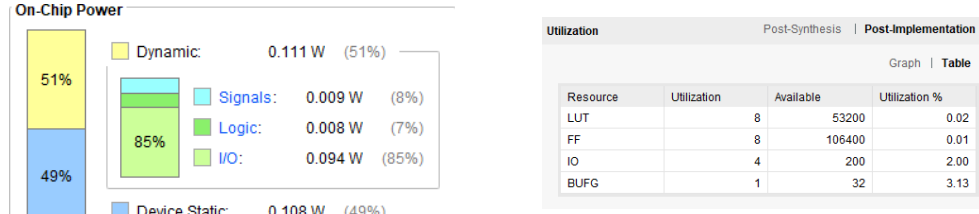


Fig. 13 Power and Utilization report of the Proposed CA Based Compressor

The results from Fig. 10, 11, 12 indicate that the CA-based compression method significantly reduces storage and power requirements while maintaining signal integrity.

4 Conclusion

This paper presents a Cellular Automata-based EEG data compression framework utilizing Rule 90 and Rule 150 for efficient real-time signal processing. The proposed approach achieves an 85% reduction in storage with minimal signal loss, demonstrating its potential for low-power, embedded biomedical applications. Future work will explore adaptive CA-based compression techniques to further enhance performance across diverse EEG datasets.

References

- [1] S. Wolfram, "Statistical mechanics of cellular automata," *Rev. Mod. Phys.*, vol. 55, no. 3, pp. 601–644, Jul. 1983, doi:10.1103/RevModPhys.55.601.
- [2] J. von Neumann, *Theory of Self-Reproducing Automata*, University of Illinois Press, 1966.
- [3] P. P. Chaudhuri, D. R. Chowdhury, S. Nandi, and S. Chattopadhyay, *Additive Cellular Automata: Theory and Applications*, John Wiley & Sons, 1997.
- [4] S. Wolfram and M. Gad-el-Hak, "A New Kind of Science," *Applied Mechanics Reviews*, vol. 56, no. 2, pp. B18–B19, Mar. 2003, doi:10.1115/1.1553433.
- [5] K. K. Mandrumaka and F. Noorbasha, "A low power 10 bit SAR ADC with variable threshold technique for biomedical applications," *SN Appl. Sci.*, vol. 1, p. 918, 2019, doi:10.1007/s42452-019-0940-3.
- [6] K. K. Mandrumaka, R. V. R. Gankidi, M. Bhargavi, and A. Sajja, "Design of high-speed Vedic multiplier using three-operand parallel prefix adder," *AIP Conf. Proc.*, vol. 2942, p. 020026, Feb. 2024, doi:10.1063/5.0196495.
- [7] A. Sajja and S. Rooban, "FPGA implementation of automatic seizure detection in EEG signals using machine learning algorithm," *Discov Appl Sci*, vol. 6, p. 383, 2024, doi:10.1007/s42452-024-06060-4.