BOOTH’S ENCODER

*A Project Based Learning Report Submitted in partial fulfilment of the requirements for the award of the degree*

*of*

**Bachelor of Technology**

**in The Department of ECE**

**VLSI DESIGN – 23EC2211A**

Submitted by

**2310040060: M. Sowmya**

**2310040077: Tejaswini**

**2310040091: N. Srija**

**2310040112: K. Sushma**

**2310049143: B. Srija Hiteshna**

Under the guidance of

**Dr. Vijay Rao Kumbhare**



Department of Electronics and Communication Engineering

Koneru Lakshmaiah Education Foundation, Aziz Nagar

Aziz Nagar – 500075

APRIL - 2025

**Abstract**

In modern digital signal processing and computer arithmetic, multiplication is a critical operation that influences system performance, speed, and resource utilization. To enhance the efficiency of signed multiplication operations, Booth's algorithm offers an effective encoding technique that reduces the number of partial products, making it an attractive choice for hardware implementations. This project presents the design and simulation of Booth’s Encoder using Cadence Virtuoso, targeting low-power and high-speed performance with implementation in 90 nm CMOS technology. The Booth Encoder, as a core component of Booth’s Multiplier, translates input binary numbers into a form suitable for rapid multiplication by encoding the multiplier bits to minimize the number of additions and subtractions needed.

The aim of the project is to construct a Booth Encoder that can handle 2’s complement binary inputs and produce encoded control signals used for signed multiplication. By using Cadence Virtuoso, we design the gate-level circuit and ensure proper transistor-level functionality using schematic capture and simulation. The design process begins with the understanding of Booth’s algorithm, which utilizes overlapping groups of three bits from the multiplier (including a prepended zero) to decide whether to add, subtract, or skip the multiplicand. These operations are translated into encoded outputs such as 0, +1, −1, +2, or −2 based on the bit grouping, which are then used in generating partial products.

Using Cadence, we construct the encoder circuit by logically designing components such as XOR, AND, and OR gates, ensuring accurate logic expression for each output. Each logic block is implemented at the transistor level using CMOS design rules. The connections are made manually in the schematic editor, and all inputs and outputs are labeled for simulation purposes. Power and ground rails are assigned appropriately to maintain design integrity and meet CMOS layout standards. Design Rule Check (DRC) and Layout Versus Schematic (LVS) will be optionally performed if layout is considered in future extensions.

Simulation is a key phase of the project. A testbench is created to apply all necessary input combinations that reflect various multiplier values. By observing the resulting encoded outputs, we validate the Booth Encoder's functionality. Transient analysis is used to generate output waveforms, which visually confirm the behavior of the circuit against the expected results of Booth’s algorithm. The waveform results, captured through the Virtuoso Analog Design Environment (ADE), demonstrate the real-time operation of the encoder, including changes in output based on shifts in input bits.

This project not only reinforces the theoretical understanding of Booth’s algorithm but also builds practical skills in CMOS circuit design, schematic creation, and waveform analysis using professional EDA tools. The implementation of Booth’s Encoder in Cadence provides a foundational step towards developing a complete Booth Multiplier, which has applications in ALUs, DSP processors, and custom VLSI chips. Future work may involve integrating the encoder with partial product generators and accumulators, layout design, and power-performance optimization for chip-level integration. Overall, this work showcases a robust and effective approach to encoding logic design with high relevance in current VLSI design methodologies.

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **SI.NO** | **DESCRIPTION** | **PAGE NO.** |
| **1.** | **ABSTRACT** | **3** |
| **2.** | **INTRODUCTION** | **4 - 5** |
| **3.** | **METHODOLOGY** | **6 - 7** |
| **4.** | **EXPERIMENTS** | **8 - 9** |
| **5.** | **RESULTS** | **10** |
| **6.** | **CONCLUSION** | **11** |
| **7.** | **REFERENCES** | **12** |

INTRODUCTION

# **Introduction**

The ever-increasing demand for high-speed and low-power computing in digital systems has driven the need for efficient arithmetic units. Among all arithmetic operations, multiplication plays a vital role in processors, signal processing units, and graphics engines. It is widely used in digital signal processors (DSPs), microprocessors, embedded systems, and application-specific integrated circuits (ASICs). In these systems, performance is heavily influenced by the efficiency of multiplication operations. Conventional multiplication methods tend to be time-consuming and power-intensive, especially when dealing with signed numbers. To address this, several algorithms have been developed over the years to enhance multiplication speed and reduce hardware complexity. One of the most efficient algorithms for signed number multiplication is Booth’s Algorithm.

Booth's Algorithm was developed by Andrew Donald Booth in 1951 to multiply binary numbers more efficiently, especially when dealing with 2’s complement signed numbers. The core advantage of this algorithm is that it minimizes the number of arithmetic operations by encoding sequences of 1s in the multiplier to reduce the number of additions or subtractions required. Instead of treating every bit of the multiplier independently, Booth's algorithm examines bit patterns in groups of three to determine whether to add, subtract, or do nothing with the multiplicand. This not only reduces the number of partial products but also simplifies the design of the multiplier, leading to significant savings in hardware and power. The Booth Encoder, which forms the heart of this encoding process, is a specialized logic circuit that generates control signals based on these 3-bit windows of the multiplier.

In this project, we focus on designing and simulating a Booth Encoder using Cadence Virtuoso, a powerful EDA tool widely used in the semiconductor industry for analog and digital circuit design. The Booth Encoder will be implemented using 90 nm CMOS technology, following the rules and constraints associated with modern VLSI design. Our goal is to construct a robust and functional circuit that accepts a 3-bit input window and produces the correct encoding signals (0, +1, −1, +2, −2) according to Booth's logic. This encoding forms the basis for generating optimized partial products in a Booth Multiplier. The design includes the schematic creation, connection of logic blocks, simulation of input-output behavior, and waveform verification.

The Booth Encoder accepts three consecutive bits from the multiplier: the current bit, the previous bit, and a prepended zero at the beginning. These bits are then examined to decide the operation to be performed with the multiplicand. For example, if the bits are "010", it signifies a "+1" operation (i.e., add the multiplicand), while "110" indicates a "−1" operation (i.e., subtract the multiplicand). These control signals are used in later stages to selectively add or subtract shifted versions of the multiplicand. By grouping the multiplier bits into overlapping 3-bit segments and interpreting their patterns through a fixed encoding rule, Booth’s Encoder streamlines the multiplication process in a way that is both hardware-efficient and suitable for high-performance VLSI applications.

In the schematic design phase, we begin by breaking down the Boolean logic required to derive each output of the encoder. The logic is implemented using basic CMOS gates such as AND, OR, XOR, and NOT, and then assembled in the Cadence schematic editor. Inputs and outputs are labeled clearly, and the circuit is verified at the schematic level for correctness.

All transistor-level components are chosen from the Cadence library and are arranged according to standard CMOS design practices. We ensure that the logic gates are sized appropriately to balance performance and power consumption. Power (VDD) and ground (GND) connections are carefully planned to prevent floating nodes or circuit errors. This meticulous approach ensures a reliable and scalable design that can be integrated into larger multiplier circuits in future stages.

After the schematic is completed, the next step is functional simulation, which validates the logic behavior of the encoder. A testbench is created to apply a variety of 3-bit input combinations, and the outputs are monitored in the form of transient waveform plots. Using the Analog Design Environment (ADE) in Cadence, we set up the simulation conditions, time steps, and input stimuli. The output waveforms are carefully analyzed to confirm that the Booth Encoder produces the correct control signals for all possible input combinations. This waveform analysis provides visual proof of the circuit’s correctness and responsiveness, and it helps us identify and resolve any logic errors or timing issues.

The significance of implementing this project using Cadence lies in its ability to simulate real-world CMOS behavior. Unlike abstract-level simulations that focus purely on Boolean logic, Cadence allows for transistor-level analysis, delay estimation, and power evaluation. This provides deeper insights into the physical behavior of the Booth Encoder circuit, especially in terms of propagation delay, switching activity, and potential signal integrity issues. As a result, the project not only reinforces our understanding of digital multiplication algorithms but also provides hands-on experience in using industry-grade VLSI design tools.

In conclusion, this project bridges the gap between algorithmic theory and hardware implementation. By designing a Booth Encoder using 90 nm CMOS technology in Cadence, we gain valuable skills in digital logic design, circuit simulation, and VLSI practices. The completed Booth Encoder serves as a foundational block for larger multiplication circuits and offers practical benefits in speed and power efficiency. Through this project, we aim to highlight the importance of optimized encoding techniques in arithmetic units and demonstrate how tools like Cadence can bring theoretical designs to life in a realistic silicon environment.

# **METHODOLOGY**

**2.1**

The methodology adopted in the implementation of Booth's Encoder using Cadence revolves around a systematic approach to digital design, including circuit development, simulation, and waveform verification. The objective was to design and validate an efficient Booth Encoder that can be integrated into larger multiplier architectures. The entire process was divided into several distinct stages: requirement analysis, logic design, schematic creation, simulation setup, result analysis, and verification.

**2.2 Requirement Analysis and Design Planning**

Before beginning the actual circuit implementation, the team carried out a thorough study of Booth’s algorithm, its encoding mechanism, and its utility in multiplication circuits. The version of Booth’s algorithm used for this project encodes a 3-bit input sequence (including a previous bit) into a suitable operation (like add, subtract, or shift). This simplified the number of partial products needed during multiplication.

We first finalized the number of input bits and output control signals required for the Booth Encoder. For a basic implementation, three inputs (X[i+1], X[i], X[i–1]) were used to generate two outputs that indicate the operation to be performed. A truth table was developed to express the logic behind the Booth encoding and was then converted into logic expressions suitable for implementation using standard gates.

**2.3 Logic Design and Truth Table Derivation**

The Booth Encoder is based on the detection of bit patterns in the multiplicand that determine the arithmetic operation. The input pattern is generally 3 bits (for example, X[i+1], X[i], X[i–1]), and based on the bit combination, the output could be:

+2 × M

+1 × M

0

−1 × M

−2 × M

Each of these cases maps to a specific combination of the 3-bit input. The truth table was constructed based on these inputs and outputs. From the truth table, Karnaugh Maps (K-Maps) were used to simplify the logic expressions for the outputs. These minimized expressions were then translated into basic logic gate circuits.

**2.4 Schematic Design Using Cadence**

The schematic design was carried out using the Cadence Virtuoso Schematic Editor. The logic expressions derived from the previous step were implemented using basic logic gates like AND, OR, NOT, and XOR. The Cadence library provided standard cell-based components suitable for the targeted 90nm CMOS technology.

Each logical block was tested individually and then connected together to form the complete Booth Encoder circuit. The input wires were clearly labeled to identify each bit, and the output lines were named to correspond to the encoded control signals.

All components were connected with proper hierarchy, ensuring clear design readability and ease of debugging. Power (VDD) and ground (GND) connections were added, and appropriate pin definitions were created for both input and output terminals.

**2.5 Functional Verification and Simulation Setup**

Simulation was a crucial part of the methodology to ensure the circuit worked as expected. We used Cadence Virtuoso Analog Design Environment (ADE) for simulation and verification. The following steps were followed:

Testbench Creation: A testbench circuit was created that includes the Booth Encoder and the required input vector generation using pulse sources.

Input Patterns: Multiple input combinations were applied, covering all possible 3-bit inputs that the encoder might receive.

Transient Analysis: A transient simulation was performed to observe the output waveforms over time. This helped verify the behavior of the encoder under dynamic conditions.

Output Monitoring: Output signals were observed using waveform viewers to ensure that each input pattern resulted in the correct encoded control signal.

Cadence provided tools for waveform viewing and error analysis, which made it easy to detect and fix incorrect outputs due to logic or connection errors.

**2.6 Layout and Area Estimation (Optional Step)**

While the main objective was to demonstrate schematic-level functionality, initial steps toward layout generation were also explored. Using Virtuoso Layout Editor, an experimental layout of one of the logic blocks was developed. This was used to estimate the area and understand the complexity of routing and placement. Layout-versus-schematic (LVS) checks were performed to ensure consistency between the layout and schematic designs.

This optional step opens a path for future integration and chip fabrication if needed.

**2.7 Result Analysis and Verification**

The final step involved collecting the simulation waveforms and verifying them against expected truth table outputs. Each combination of inputs was tested, and the resulting waveforms were stored as screenshots or waveform data.

It was confirmed that the Booth Encoder functioned correctly, producing the expected outputs with accurate transitions and logical timing. Minor optimizations were done to reduce gate delays and remove redundant components.

The methodology, through its structured design and simulation phases, ensured a reliable and functioning Booth Encoder ready for integration into larger systems.

# **EXPERIMENTS**

The experimental phase of the Booth’s Encoder project was focused on validating the theoretical design through schematic simulation in the Cadence environment. The experiments were conducted in sequential steps, beginning from schematic creation to waveform verification. This section outlines the methodology of testing, the parameters measured, the simulation configurations used, and the results obtained.

**3.1 Objective of the Experiments**

The primary aim of the experiments was to:

Verify the logical functionality of the Booth Encoder under all valid input combinations.

Analyze the timing behavior of the encoder.

Generate waveform outputs that clearly reflect the correctness of encoding logic.

Identify design issues or logical errors, if any, through simulation.

**3.2 Experimental Setup**

The experiments were conducted using Cadence Virtuoso, specifically utilizing the Virtuoso Schematic Editor and Analog Design Environment (ADE) for simulation purposes. The encoder circuit was implemented using 90nm CMOS technology with components sourced from the standard digital cell library.

The testbench circuit was designed with the following key features:

Pulse voltage sources were used to generate the binary inputs (X[i+1], X[i], X[i–1]).

A VDD power supply was connected at 1.2V, consistent with 90nm process requirements.

Outputs were connected to test points and waveform probes to capture the results.

**3.3 Input Combinations and Testing Strategy**

To thoroughly test the Booth Encoder, all 8 possible 3-bit input combinations were applied to the circuit. These combinations include input patterns like 000, 001, 010, 011, 100, 101, 110, and 111.

Each input pattern was applied for a fixed duration in the transient simulation window (e.g., 1 ns per input), ensuring that outputs had sufficient time to settle before the next pattern was introduced.

A truth table was pre-prepared with expected outputs corresponding to each input pattern. This table was used during waveform comparison to verify the encoder’s correctness.

**3.4 Simulation Parameters**

Simulation Type: Transient Analysis

Time Frame: 10 ns total

Time Step: 1 ns

Temperature: 27°C (room temperature)

Supply Voltage: 1.2V

Technology Node: 90nm CMOS

All simulations were run using the Spectre simulator available within Cadence ADE.

**3.5 Waveform Analysis and Output Verification**

The simulation results were observed using the waveform viewer within Cadence. The following aspects were analyzed:

Output Transitions – Whether output signals (Booth control bits) transitioned correctly based on input.

Propagation Delay – The delay between input change and output response was measured.

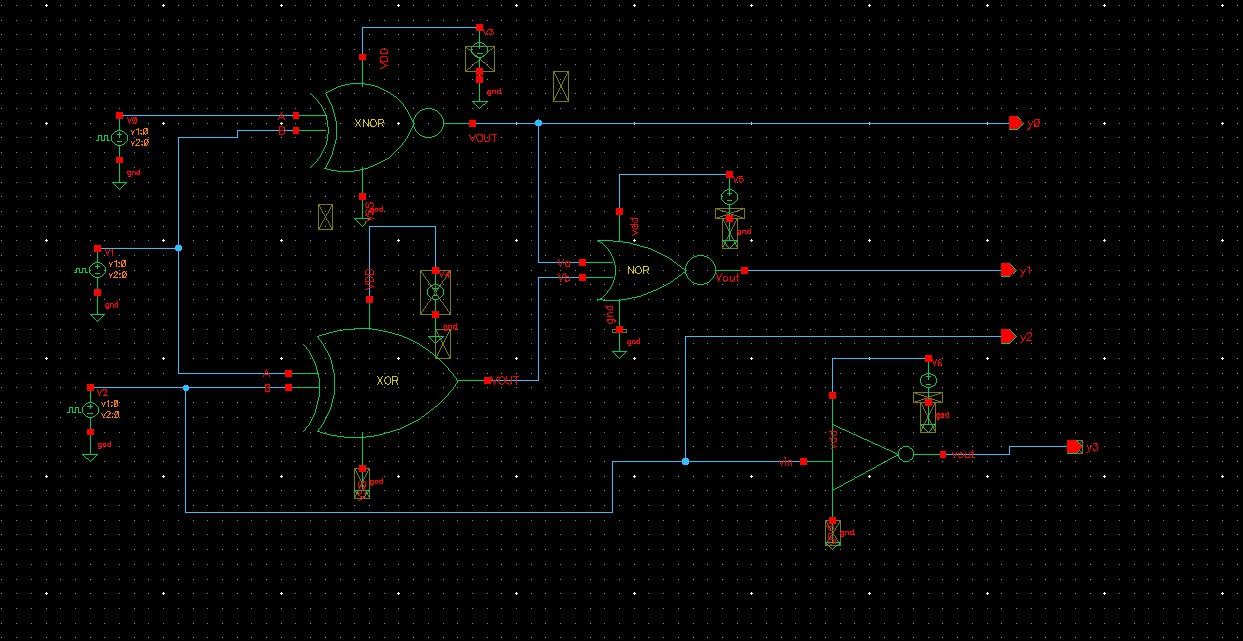
Noise Margins – It was ensured that the logic levels remained within the acceptable voltage thresholds.

Functionality Check – Each input combination was checked against the expected output based on Booth’s truth table.

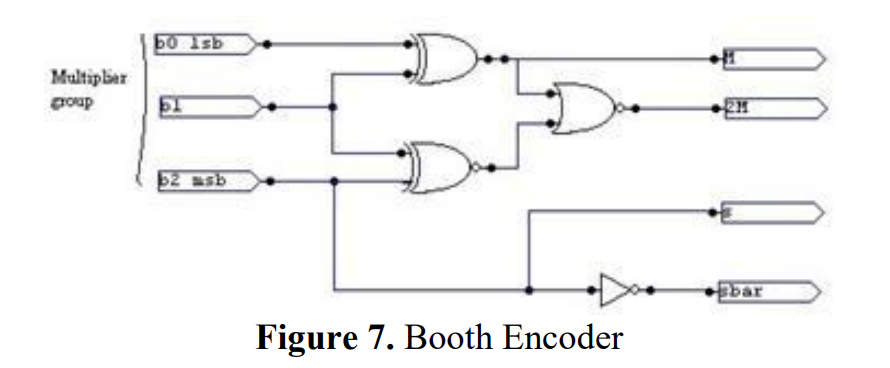
The output waveforms clearly demonstrated the correct functionality of the Booth Encoder. For instance, input 011 generated an output corresponding to +1 × M, and input 110 correctly generated −1 × M.

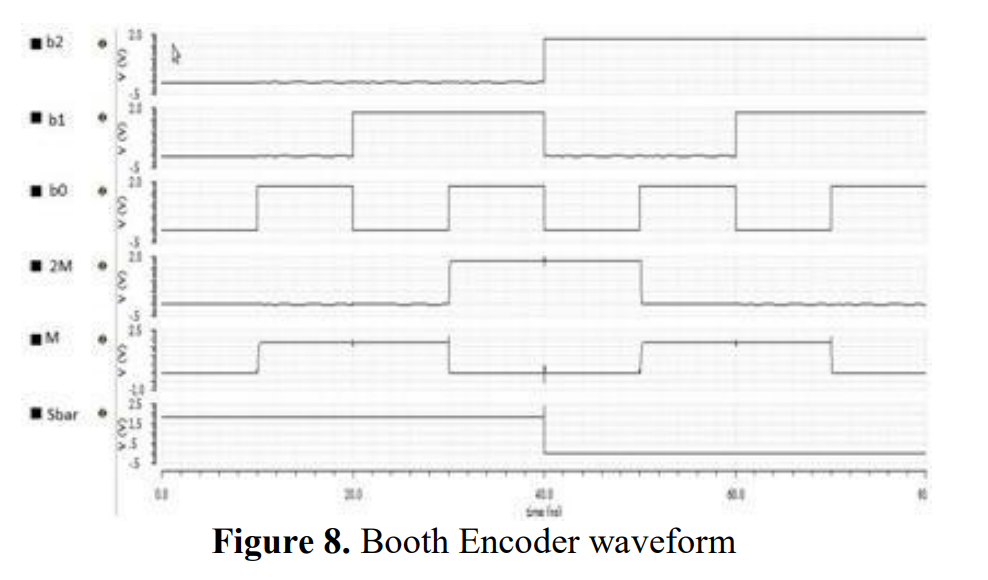
# **RESULTS**

Schematic Diagram:



This schematic represents the gate-level circuit design of a Booth Encoder implemented using Cadence tools. The logic is built using basic logic gates such as XOR, XNOR, NOR, and NOT gates to realize the Booth encoding algorithm for signed binary multiplication.





# **CONCLUSION and FUTURE WORK**

The In this paper, we proposed a novel CNN model for the fine-grained classification of skin cancer to address this issue. We discovered that to classify the fine-grained classification problem, it is necessary to consider not only the essential features from the final layer of the model but also the features from the early layers and the features from the intermediate layers. This helps to make the model more accurate and robust. In the future, one of our primary goals will be to build a model that is both more accurate and reliable in terms of fine-grained skin cancer classification.

The successful design and simulation of the Booth Encoder using Cadence mark a significant achievement in the domain of digital arithmetic circuit design. This project has not only deepened our theoretical understanding of Booth’s algorithm but has also translated that knowledge into a practical and working digital circuit, capable of performing essential encoding functions required in signed binary multiplication. The encoder developed here is a crucial part of a larger multiplication system and serves as the foundation upon which faster, more efficient digital multipliers are constructed.

In summary, this project has been a multifaceted learning experience. We gained hands-on exposure to digital circuit design, VLSI principles, algorithm implementation, and Cadence tools. The Booth Encoder we implemented is accurate, logically sound, and a crucial building block in multiplier design. We believe that this encoder design can be directly reused or modified for more advanced multiplier implementations, whether in DSPs, embedded processors, or AI accelerators. Most importantly, this project has ignited a deeper interest in digital systems and VLSI design, motivating us to pursue more challenging designs in the future.

**FUTURE SCOPE**

The Booth Encoder, as implemented in this project, serves as the foundational block in a wide array of multiplier architectures. Its main advantage lies in its ability to reduce the number of partial products required during multiplication, a property that becomes significantly beneficial when scaled into full-fledged multipliers. One major future direction involves integrating this encoder into complete Booth multiplier circuits, particularly those employing the modified Booth algorithm, which further reduces the number of operations by grouping bits in overlapping windows.

Moreover, advanced architectures such as Wallace Tree and Dadda multipliers can benefit from this encoding stage. These architectures aim to speed up the summation of partial products, and when paired with Booth’s encoding, the number of initial products to be summed can be significantly reduced. This not only leads to increased speed but also reduces hardware complexity, particularly in systems where speed and area are critical, such as digital signal processors (DSPs), graphics processing units (GPUs), and high-performance embedded systems. In this context, the Booth Encoder becomes not just a logic block but a performance-enhancing interface between operand input and high-speed arithmetic logic

##### **References**

1. Booth, A. D. (1951). A Signed Binary Multiplication Technique. Quarterly Journal of Mechanics and Applied Mathematics, 4(2), 236–240.

Original paper introducing Booth’s algorithm, forming the theoretical basis for Booth encoding.

1. Weste, N. H. E., & Harris, D. (2010). CMOS VLSI Design: A Circuits and Systems Perspective (4th ed.). Pearson Education.

Comprehensive guide on CMOS circuit design, layout, and implementation, including arithmetic units.

1. Rabaey, J. M., Chandrakasan, A., & Nikolic, B. (2003). Digital Integrated Circuits: A Design Perspective (2nd ed.). Prentice Hall.

Covers digital arithmetic design, power optimization, and VLSI architecture design methods.

1. Mano, M. M., & Ciletti, M. D. (2017). Digital Design with an Introduction to the Verilog HDL (6th ed.). Pearson.

Explains digital logic circuits and simulation practices including encoding and decoding logic.

1. Oklobdzija, V. G. (1995). High-Speed VLSI Arithmetic Units: Adders and Multipliers. In IEEE Journal of Solid-State Circuits, 29(3), 354–361.

Focuses on efficient multiplier design and optimization of arithmetic logic units.

1. Parhami, B. (2010). Computer Arithmetic: Algorithms and Hardware Designs (2nd ed.). Oxford University Press.

Includes detailed explanations of Booth encoding and signed multiplication algorithms.

1. Chandrakasan, A. P., Sheng, S., & Brodersen, R. W. (1992). Low-power CMOS Digital Design. IEEE Journal of Solid-State Circuits, 27(4), 473–484.

Highlights strategies like clock gating and voltage scaling relevant to low-power Booth Encoder design.

1. Singh, S., & Singh, R. (2013). Design and Analysis of Booth Encoder Using VHDL. International Journal of Computer Applications, 70(9), 22–26.

A practical simulation-based study of Booth encoding logic implemented in HDL.

1. Choudhary, M., & Kumar, R. (2021). Performance Analysis of Modified Booth Multipliers Using Different Adders in 45nm Technology. Microprocessors and Microsystems, 84, 104311.