UNIVERSITY OF IOANNINA
DEPARTMENT OF INFORMATICS & TELECOMMUNICATIONS

Πανεπιστήμιο
Ιωαννίνων

MSc DEGREE IN INFORMATICS AND NETWORKS

MSc THESIS

DESIGN AND IMPLEMENTATION OF SCAN CHAINS IN AN INTEGRATED CIRCUIT

Skaltsonis Leonidas
PN:148
Email:leonskalts@gmail.com

Supervisor: Vartziotis Fotios
PhD, Assistant Professor

Arta, September, 2024

ΣΧΕΔΊΑΣΗ ΚΑΙ ΥΛΟΠΟΊΗΣΗ ΑΛΥΣΊΔΩΝ ΕΛΈΓΧΟΥ ΣΕ ΟΛΟΚΛΗΡΩΜΈΝΟ ΚΎΚΛΩΜΑ

Approved by three-member examination committee

Arta, 27/09/2024

EXAMINATION COMMITTEE

1. **Supervisor:**

   Fotios Vartziotis,

   PhD, Assistant Professor

2. **Committee Member:**

   Gregory Doumenis,

   PhD, Assistant Professor

3. **Committee Member:**

   Constantinos T. Angelis,

   Department Dean, Professor

© Skaltsonis, Leonidas, 2024.
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Non plagiarism statement

I declare responsibly and knowing the sanctions of Law 2121/1993 on Intellectual Property, that the present master's thesis is entirely the result of my own research work, is not a product of copying nor comes from assignment to third parties. All the sources used (of any kind, form, and origin) for its writing are included in the bibliography.

Skaltsonis, Leonidas

Signature

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my professor and thesis supervisor, Mr. Fotios Vartziotis, for his invaluable guidance, encouragement, and expertise throughout the development of this thesis. His support and insights were instrumental in shaping this research, and I am truly grateful for his mentorship.
I would also like to extend my sincere thanks to my family and friends for their unwavering support, patience, and encouragement throughout my studies. Their constant belief in my abilities and emotional support have been vital to my success, and I am deeply grateful for their continued presence and support.

ΠΕΡΙΛΗΨΗ

Η παρούσα διατριβή εξετάζει τις προόδους στον τομέα του manufacturing testing με έμφαση στον έλεγχο αλυσίδων ελέγχου και στο Design-for-Testability. Η έρευνα υποκινείται από τις σημαντικές προκλήσεις που παρουσιάζονται στον έλεγχο αλυσίδων σάρωσης, οι οποίες είναι κρίσιμες για τη βελτίωση της απόδοσης και της αξιοπιστίας των σύγχρονων κατασκευαστικών διαδικασιών. Μέσω μιας εκτενούς ανασκόπησης της βιβλιογραφίας, η μελέτη εντοπίζει τις βασικές τεχνολογίες στον έλεγχο αλυσίδων ελέγχου και αξιολογεί τα τρέχοντα εργαλεία στην αντιμετώπιση αυτών των προκλήσεων.

Η έρευνα εφαρμόζει ένα ισχυρό μεθοδολογικό πλαίσιο, ενσωματώνοντας εργαλεία όπως το Qflow και προσαρμοσμένα Ολοκληρωμένα Περιβάλλοντα Ανάπτυξης (IDEs), μαζί με την ανάπτυξη και εφαρμογή κυκλωμάτων αναφοράς και αλγορίθμων ελέγχου. Αυτή η προσέγγιση δοκιμάζεται αυστηρά μέσω μιας σειράς πειραμάτων που όχι μόνο επικυρώνουν τις μεθοδολογίες, αλλά συμβάλλουν και σε θεωρητικές και πρακτικές προόδους στον τομέα.

Τα πειραματικά αποτελέσματα αποδεικνύουν την αποτελεσματικότητα των προτεινόμενων μεθοδολογιών στη βελτιστοποίηση του ελέγχου αλυσίδων ελέγχου, αντιμετωπίζοντας ζητήματα όπως η κατανάλωση ενέργειας, οι περιπλοκότητες δρομολόγησης και η αποδοτικότητα του ελέγχου. Τα ευρήματα επικυρώνουν τα πρακτικά οφέλη αυτών των τεχνικών. Η έρευνα παρέχει πολύτιμες γνώσεις για την εφαρμογή βελτιωμένων στρατηγικών Design-for-Testability, υποστηρίζοντας την ανάπτυξη πιο αξιόπιστων και αποδοτικών διαδικασιών ελέγχου VLSI.

Η διατριβή καταλήγει με μια λεπτομερή ανάλυση των κύριων αποτελεσμάτων, τονίζοντας την επίδρασή τους στον ευρύτερο τομέα του manufacturing testing. Προτείνονται μελλοντικές κατευθύνσεις για την περαιτέρω βελτίωση αυτών των μεθοδολογιών, με έμφαση στην ενίσχυση της επεκτασιμότητας και της προσαρμοστικότητας του ελέγχου αλυσίδων για να ανταποκριθεί στις εξελισσόμενες τεχνολογίες ημιαγωγών.

ABSTRACT

This thesis explores advancements in manufacturing testing with a focus on scan-chain testing and design for testability. The research is motivated by significant challenges in scan chain testing, which are crucial for enhancing the efficiency and reliability of modern manufacturing processes. Through a comprehensive literature review, this study identifies key technologies in scan-chain testing and evaluates current tools in addressing these challenges.

The research employs a robust methodological framework, integrating tools like Qflow and customized Integrated Development Environments (IDEs) alongside the development and application of benchmark circuits and test algorithms. This approach is rigorously tested through a series of experiments that not only validate the methodologies but also contribute to theoretical and practical advancements in the field.

Experimental results demonstrate the effectiveness of the proposed methodologies in optimizing scan chain testing by addressing issues such as power consumption, routing complexities, and test efficiency. The findings validate the practical benefits of these techniques. This research provides valuable insights into the implementation of enhanced Design-for-Testability strategies, supporting the development of more reliable and efficient VLSI testing processes.

The thesis concludes with a detailed analysis of the key outcomes, highlighting their impact on the broader field of manufacturing testing. It proposes future research directions aimed at further refining these methodologies, with a focus on enhancing the scalability and adaptability of scan chain testing in response to evolving semiconductor technologies.

Contents

1 Introduction **1**
1.1 Background . . . 1
1.2 Research Motivation . . . 1
1.3 Objectives and Scope of the Thesis . . . 2
1.4 Thesis Contributions . . . 2
1.5 Organization of the Thesis . . . 2

2 Literature Review **3**
2.1 DFT Key Technologies . . . 3
2.2 Scan Chain Key Methodologies . . . 5
2.3 Current Tools and Methodologies . . . 7
2.3.1 Overview of Current Tools . . . 7
2.3.2 Limitations of These Tools . . . 9
2.4 Register-Transfer Level (RTL) Design . . . 10
2.4.1 RTL Design . . . 10
2.4.2 Selection of RTL Design Tools . . . 12
2.4.3 Integration and Customization . . . 12
2.5 Qflow and its Role in Testing . . . 13
2.5.1 Introduction to Qflow . . . 13
2.5.2 How Qflow Addresses Existing Problems . . . 15
2.5.3 Qflow in Comparative Context . . . 17
2.6 Problem Statement . . . 18
2.6.1 Detailed Definition of the Problem . . . 18
2.6.2 Importance of Addressing This Problem . . . 20
2.6.3 Expected Innovations from Addressing the Problem . . . 21

3 Research Methodology and Framework **22**
3.1 Overview of Methodological Approach . . . 22
3.1.1 Rationale Behind Chosen Methods . . . 22
3.1.2 Research Design . . . 24
3.1.3 Methodological Framework . . . 27
3.2 Qflow Nandgate 45nm Libraries . . . 28
3.2.1 Technical Specification of Libraries . . . 28
3.2.2 Application in Research . . . 29
3.2.3 Evaluation of Performance . . . 30
3.3 Tool Implementation (2D Chip ScanChain Creator) . . . 30
3.3.1 Integration and Functionality in the Research Context . . . 31
3.3.2 Case Study Applications . . . 35
3.4 Benchmark Circuits and Test Algorithms . . . 36
3.4.1 Selection Criteria for Benchmark Circuits . . . 36
3.4.2 Description and Justification of Algorithms Used . . . 36
3.4.3 Testing and Validation . . . 38

4 Experiments and Results **39**
4.1 Experimental Setup . . . 39

4.1.1 Software and Tools Used 39
4.1.2 Hardware Configuration 42
4.2 Data Collection and Analysis 42
4.3 Zone-Based and Non-Zone-Based Scan Chains 45
4.4 Visualization of Results 46
4.5 Detailed Presentation of Results 49
4.6 Evaluation and Interpretation 56
4.6.1 Analysis of LeoSolveTSP Performance Compared to Other Algorithms 56
4.6.2 Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance 59

5 Conclusions and Future Work 61
5.1 Findings 61
5.1.1 Contextualizing Results with Existing Literature 61
5.1.2 Identifying and Addressing Gaps 63
5.2 Applications and Impact 63
5.2.1 Practical Applications 63
5.2.2 Impact on the Field 64
5.3 Limitations 64
5.4 Recommendations for Future Research 65
5.5 Final Thoughts 66

List of Tables

1 Comparison of Random Path and TSP Path Distances 26
2 Algorithms Complexity 38
3 Number of DFFs in Different Benchmark Circuits 43
4 Benchmark Circuit: s5378, Non-Zone-Based 49
5 Benchmark Circuit: s5378, Zone-Based 50
6 Benchmark Circuit: s13207, Non-Zone-Based 50
7 Benchmark Circuit: s13207, Zone-Based 51
8 Benchmark Circuit: s38417, Non-Zone-Based 51
9 Benchmark Circuit: s38417, Zone-Based 52
10 Benchmark Circuit: s38584, Non-Zone-Based 52
11 Benchmark Circuit: s38584, Zone-Based 53
12 Benchmark Circuit: ac97_top, Non-Zone-Based 53
13 Benchmark Circuit: ac97_top, Zone-Based 54
14 Benchmark Circuit: aes_cipher_top, Non-Zone-Based 54
15 Benchmark Circuit: aes_cipher_top, Zone-Based 55
16 Benchmark Circuit: md5, Non-Zone-Based 55
17 Benchmark Circuit: md5, Zone-Based 56
18 Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance (Part 1) 59
19 Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance (Part 2) 60

List of Figures

1 Comparison of Non-Scan and Scan Sequential Circuits.[1] 1
2 DFT architecture[2] 3
3 BIST Design[3] 4
4 Scan-Chains[4] 6
5 Synopsys Design Compiler[5] 7
6 Cadence Encounter[6] 8
7 Xilinx Vivado Design Suite[7] 9
8 RTL Design Example: D Flip-Flop in VHDL and Verilog[8] 11
9 Quartus[9] 12
10 Qflow[10] 14
11 Raven RISC-V microprocessor[10] 14
12 Detail of a layout generated by qrouter[11] 16
13 Viewing, extraction, DRC checks, GDS Generation with Magic[12] . . . 17
14 Scan Chain Connection[13] 19
15 Transition of a normal flop to scan flop[14] 20
16 Overview of Methodological Approach for Algorithm Development and Testing 22
17 Traveling Salesman Problem[15] 23
18 Area with 30 nodes 25
19 Random path 25
20 TSP Nearest Neighbour path 26
21 Euclidean distance calculation formula 27
22 FreePDK45-Placement-and-Routing with Virtuoso from Cadence Design system[16] 28
23 LT Spice Alternating Current(AC) Analysis[17] 29
24 2D Chip ScanChain Creator 31
25 Floorplan 33
26 Advanced Floorplan 33
27 Point-Based Scan Pins Graph 34
28 Display ScanChains 34
29 Christofides Algorithm[18] 36
30 Visual Studio 2022[19] 39
31 Visual Studio 2022 IDE[20] 40
32 PyCharm 41
33 PyCharm IDE 41
34 Tool Workflow in 2D Chip ScanChain Creator 44
35 Comparison of Zone-Based and Non-Zone-Based Scan Chain Configurations 45
36 Line Chart 47
37 Bar Chart 48
38 Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance for 4 scanchains 59

TABLE OF ABBREVIATIONS (ACRONYMS)

Abbreviation	Definition
AC	Alternating Current
ASICs	Application-Specific Integrated Circuits
BIST	Built-In Self-Test
CAD	Computer-Aided Design
CPU	Central Processing Unit
DDR5	Double Data Rate 5
DEF	Design Exchange Format
DEF Files	Design Exchange Format Files
DFM	Design for Manufacturability
DFT	Design for Testability
DTCO	Design Technology Co-Optimization
DUT	Device Under Test
EDA	Electronic Design Automation
FPGA	Field-Programmable Gate Array
FUMV	Full Universe Model Verification
GDSII	Graphic Data System II

Abbreviation	Definition
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HLS	High-Level Synthesis
IC	Integrated Circuit
LEF	Library Exchange Format
LVS	Layout Versus Schematic
ML	Machine Learning
MST	Minimum Spanning Tree
NN	Nearest Neighbor
NP-hard	Non-deterministic Polynomial-time hard
PDK	Process Design Kit
Qflow	Quality Flow
RTL	Register-Transfer Level
SCI	Scalable Coherent Interface
SPICE	Simulation Program with Integrated Circuit Emphasis
SSD	Solid State Drive
STA	Static Timing Analysis

Abbreviation	Definition
TSP	Traveling Salesman Problem
VHDL	VHSIC Hardware Description Language
VLSI	Very-Large-Scale Integration

1 Introduction

1.1 Background

Manufacturing testing is a critical aspect of the production process for integrated circuits (ICs), ensuring that each device operates as intended before reaching the consumer. This process involves a series of tests to identify potential faults in the manufacturing process. The evolution of manufacturing testing has seen significant advancements, from simple functional tests to sophisticated methodologies such as scan-chain testing, which allows for the detection of faults within a circuit by shifting test patterns through a chain of flip-flops embedded within the design [1].

Design for Testability (DFT) is an essential concept that integrates testing considerations into the design phase of IC development. By incorporating DFT techniques, designers can create circuits that are easier to test, thus enhancing the efficiency and reliability of the manufacturing testing process. The importance of DFT has grown with the increasing complexity of ICs, as it helps in reducing test costs and improving fault coverage [2].

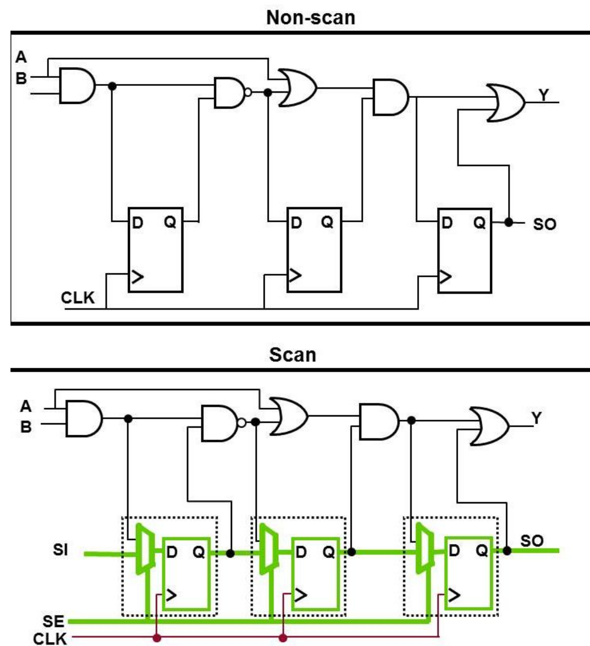

Figure 1: Comparison of Non-Scan and Scan Sequential Circuits.[1]

1.2 Research Motivation

The primary motivation for this research is to address key challenges in scan chain testing, particularly focusing on reducing power consumption, design complexity, and routing costs [3]. Efficiently managing scan chain lengths is crucial for lowering power usage and improving overall test efficiency. [4]. This research aims to optimize existing methodologies to achieve the shortest possible routing paths, thereby enhancing test speed and aligning with industry needs for more reliable and energy-efficient VLSI designs.[5].

1.3 Objectives and Scope of the Thesis

This thesis aims to address the identified challenges in scan-chain testing by developing and implementing advanced methodologies and tools. The specific objectives of the research are to:

- Enhance existing methodologies for scan-chain testing to improve efficiency and performance.
- Integrate these methodologies into a robust testing framework.
- Validate the effectiveness of the proposed solutions through comprehensive experiments.

The scope of the research encompasses the development, implementation, and validation of these methodologies, with a focus on their practical applications in the industry. Delimitations and boundaries of the study are defined to ensure a targeted approach to addressing the research questions.

1.4 Thesis Contributions

This thesis introduces targeted enhancements to scan-chain testing in VLSI circuits, focusing on optimizing existing methodologies to improve testing processes. By developing a practical framework that integrates these optimized techniques, the research demonstrates significant advancements in test efficiency and reliability. The validated methodologies provide a robust foundation for further innovations in scan-chain testing, supporting both academic exploration and practical applications in semiconductor design.

1.5 Organization of the Thesis

This thesis is organized into five chapters, each focusing on a specific aspect of the research.

Chapter 1 provides an introduction to the study, including the background, research motivation, objectives, and scope of the thesis.

Chapter 2 presents a review of the existing literature on scan-chain testing and Design for Testability (DFT), identifies key technologies and methodologies, and defines the research problem.

Chapter 3 outlines the research methodology, describing the selection and configuration of tools, as well as the development and implementation of test algorithms.

Chapter 4 details the experiments conducted and presents the results, including the experimental setup, data collection methods, and the findings of the research.

Finally, Chapter 5 discusses the findings in relation to the research problem and existing literature, explores their practical and theoretical implications, and concludes with a summary of key findings and suggestions for future research.

2 Literature Review

Scan-chain testing remains a fundamental technique in digital system testing, involving the embedding of test logic within the circuit to facilitate fault detection and diagnosis. This section reviews critical technologies that have enabled advancements in scan-chain testing and summarizes significant research efforts that have influenced current practices.

2.1 DFT Key Technologies

Design for Testability (DFT)

Design for Testability (DFT) involves integrating testing features into the design phase of integrated circuits (ICs) to simplify the testing process. DFT techniques include methods like boundary scan, built-in self-test (BIST), and scan chains, which help achieve high fault coverage and reduce test time. These techniques are crucial for managing the increasing complexity of modern ICs and ensuring their reliability. DFT enables easier fault detection and diagnosis by incorporating features that facilitate testing into the design itself. Lee et al. proposed a comprehensive DFT methodology that integrates advanced scan chain configurations and automated test pattern generation. Their work highlights the importance of incorporating testability features during the design phase, which significantly enhances the efficiency and reliability of the testing process [1, 6].

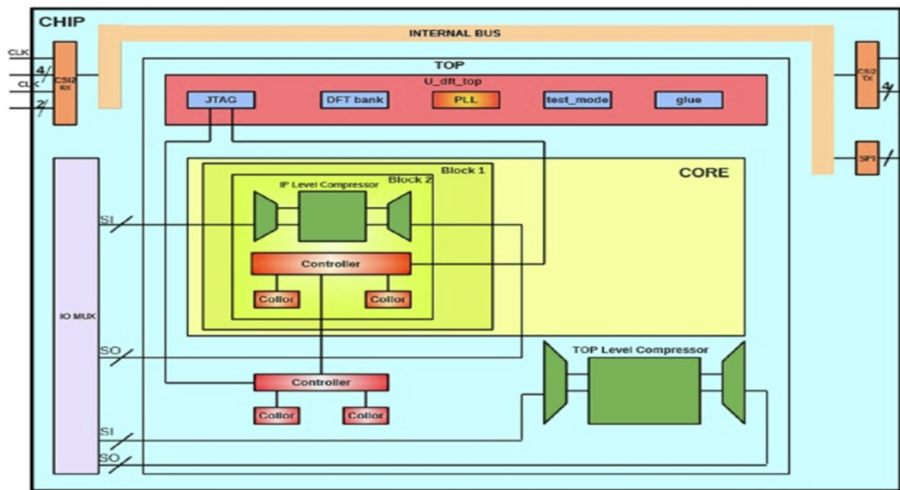

Figure 2: DFT architecture[2]

Built-In Self-Test (BIST)

Built-In Self-Test (BIST) is a technique that allows circuits to test themselves autonomously. This approach embeds test generation and response evaluation mechanisms within the IC, reducing the need for external test equipment. BIST is widely used in various types of circuits, including digital, analog, and mixed-signal circuits. It enhances fault coverage and reliability by enabling continuous monitoring and testing during the operation of the IC, thereby identifying and addressing faults more efficiently. Patel and Singh introduced a hybrid BIST approach that combines traditional methods with machine learning-based techniques to improve fault detection accuracy. Their research demonstrated the effectiveness of using ML algorithms to enhance the BIST process, resulting in reduced test time and higher fault coverage [2, 7].

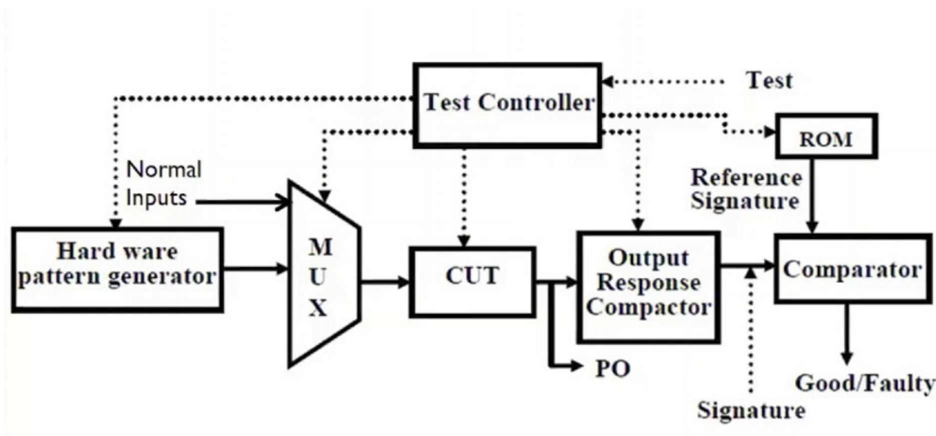

Figure 3: BIST Design[3]

Machine Learning for Test Pattern Generation

Machine learning (ML) techniques have been increasingly applied to generate more efficient test patterns for IC testing. ML algorithms can predict fault-prone areas and generate targeted test patterns, improving fault coverage and reducing test time. By leveraging large datasets of test patterns and fault responses, ML models can optimize the testing process, making it more accurate and efficient. Liu et al. explored the application of neural networks for optimizing test pattern generation. Their approach utilized deep learning techniques to predict fault-prone areas within the IC, enabling the generation of targeted test patterns. This research highlighted the potential of ML in improving test efficiency and fault coverage, providing a framework for integrating advanced data-driven techniques into traditional testing methodologies [11].

Low Power Scan Testing

As power consumption becomes a critical concern in IC design, low power scan testing techniques have been developed to reduce the power overhead during testing. Methods such as clock gating and test vector reordering are employed to minimize power consumption during scan operations. These techniques ensure that the testing process does not significantly impact the overall power budget of the IC, making them particularly important for battery-operated and energy-efficient devices. The novel low-power scan architecture introduced by Kim et al. focused on reducing power dissipation

during the test phase. By implementing clock gating and test vector reordering, their design achieved significant power savings while maintaining high fault coverage. This work addresses the growing concern of power consumption in IC testing, providing practical solutions for power-efficient test methodologies [12].

2.2 Scan Chain Key Methodologies

Scan Chains

Scan chains are a critical component of Design for Testability (DFT) strategies used in the testing of sequential circuits within Integrated Circuits (ICs). They work by connecting flip-flops in a serial shift register configuration, allowing test patterns to be shifted into the circuit and output responses to be shifted out. This method effectively converts complex sequential elements into more manageable combinational elements, significantly simplifying the testing process of complex digital designs [4].

How Scan Chains Work

Scan chains convert sequential elements, like flip-flops, into combinational ones during the testing phase. This is achieved by linking flip-flops into a chain where test data can be shifted in, applied to the circuit during a test mode, and then shifted out to observe the results. This approach helps identify faults in both the combinational logic and sequential elements of the circuit. The basic operations involved in scan chain testing are:

- **Scan-In:** Loading test data into the chain of flip-flops by shifting it serially through the chain. This operation prepares the circuit for testing by inserting specific test patterns designed to trigger faults.
- **Test Application:** Applying the test data to the circuit while the system is in a test mode to trigger faults. During this phase, the circuit operates normally, but with the inserted test data influencing its behavior, allowing faults to manifest.
- **Scan-Out:** Shifting the output data from the flip-flops to observe the results and diagnose faults. This step involves extracting the results of the test application phase, enabling the identification and diagnosis of any faults present within the circuit.

Key Advancements in Scan Chain Technologies

Key advancements in scan chain technologies have been developed to address the limitations of traditional static scan chains.

Adaptive Scan Chains dynamically adjust their configuration during operation based on real-time data analysis. Unlike static scan chains, which have a fixed configuration, adaptive scan chains can reconfigure to optimize test efficiency, reduce test time, and minimize power consumption. This adaptability is particularly useful in scenarios where power efficiency and rapid testing are critical, such as in high-performance or low-power devices [8].

Segmented Scan Chains divide a long scan chain into smaller, independently controllable segments. This segmentation reduces power consumption, minimizes test

time, and enhances diagnostic capabilities. By allowing parallel testing of segments, segmented scan chains can significantly speed up the testing process compared to traditional single-chain configurations [9].

Multi-Frequency Scan Chains operate at different frequencies within the same IC, allowing segments of the circuit to be tested under conditions that closely match their operational environment. This approach helps to uncover timing-related issues that might not be visible under single-frequency testing conditions.

Low Power Scan Chains address the challenge of power consumption during testing, which is especially significant for battery-powered devices. These scan chains utilize techniques such as clock gating, test data compression, and minimized switching activity to reduce power usage during testing, ensuring that the process does not excessively drain power resources while maintaining high fault coverage [10].

X-Tolerant Scan Chains are designed to manage unknown logic states (X-states) that can disrupt the accuracy of test results. These scan chains employ specialized techniques, such as masking or filtering X-values during testing, to improve the robustness and accuracy of the scan test by reducing the impact of unknown states on fault detection.

Hybrid Scan Chains combine features from various technologies, such as adaptive and segmented approaches, to leverage the strengths of each method. This combined strategy results in a more flexible and efficient testing process, capable of addressing a wider range of testing challenges.

Finally, modern scan chains incorporate **Robust Scan Chain Repair Techniques** that detect and reroute around faults within the scan chain. These built-in repair mechanisms, including automatic fault detection and bypass circuits, allow scan chains to continue functioning even when some cells are defective, enhancing the overall resilience and reliability of the testing process.

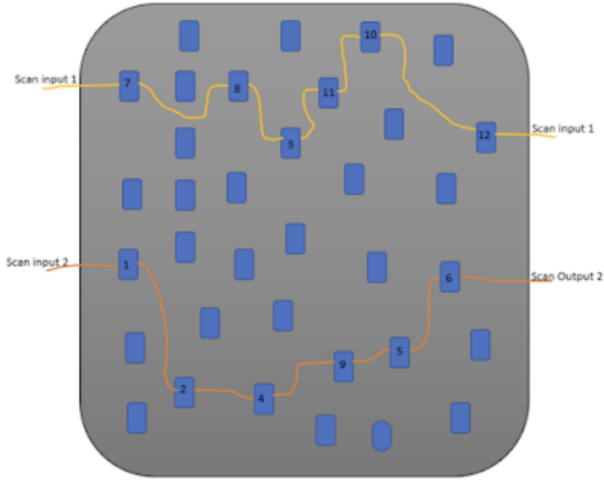

Figure 4: Scan-Chains[4]

Graph-Based Scan Chain Stitching

Graph-based scan chain stitching is an advanced technique that optimizes the connections between scan cells within an IC by applying graph algorithms to the circuit's

design descriptions. This method significantly enhances the efficiency of scan chain stitching, reducing test time, minimizing the physical length of the scan paths, and improving fault coverage. The process involves modeling the IC as a graph where nodes represent scan cells and edges represent possible connections. The objective is to find the optimal configuration that minimizes the total length of the scan chain while ensuring robust fault detection.

This graph-based approach is particularly advantageous in large and complex ICs, where traditional stitching methods can become inefficient. By leveraging graph algorithms, designers can determine the best way to connect scan cells, avoiding unnecessary long paths and reducing the overall complexity of the test setup. Research by Zaourar et al. introduced a novel graph-based scan chain stitching method that utilizes RTL (Register Transfer Level) design descriptions. Their study demonstrated substantial improvements in stitching efficiency and fault coverage, making this approach highly effective for modern IC testing [13].

2.3 Current Tools and Methodologies

2.3.1 Overview of Current Tools

This subsection reviews the current tools and methodologies used in scan-chain testing, detailing their functionalities and applications.

Synopsys Design Compiler

The Synopsys Design Compiler is widely used for synthesis and DFT insertion. It automates the creation of scan chains and integrates various DFT features like boundary scan and BIST. The tool ensures that designs meet testability requirements while optimizing performance and area. One of its key features is the ability to automatically generate and insert scan chains, which significantly reduces the time and effort required for manual intervention. The Design Compiler also supports test compression techniques to minimize test data volume and test time [14].

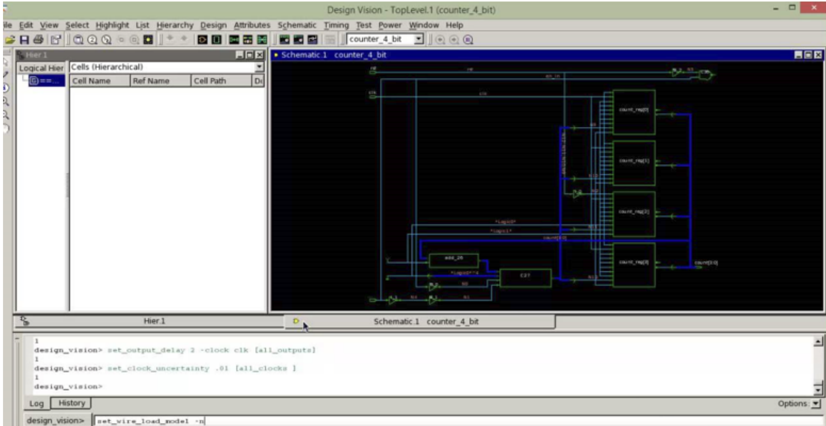

Figure 5: Synopsys Design Compiler[5]

Cadence Encounter Test

Cadence Encounter Test is a comprehensive suite for DFT and ATPG (Automatic Test Pattern Generation). It supports scan insertion, pattern generation, and fault simulation. The tool is known for its efficiency in handling large designs and its advanced diagnostic capabilities. Encounter Test provides robust fault coverage by leveraging multiple test algorithms and techniques. It also includes sophisticated diagnostic tools that allow engineers to pinpoint faults with high precision, thereby enhancing the overall reliability of the testing process [15].

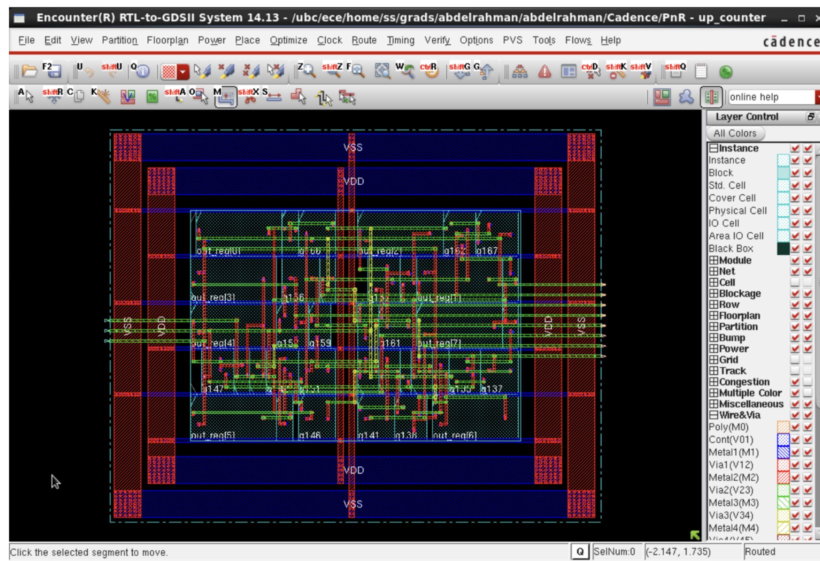

Figure 6: Cadence Encounter[6]

Mentor Graphics Tessent

Mentor Graphics Tessent provides a robust set of tools for DFT and silicon debug. It includes modules for scan insertion, BIST, and diagnostic test. Tessent is particularly noted for its high fault coverage and ease of integration into existing design flows. The Tessent suite offers advanced features such as logic BIST, memory BIST, and boundary scan, making it a comprehensive solution for IC testing. Its diagnostic capabilities are enhanced by its ability to provide detailed fault analysis and repair suggestions [16].

Xilinx Vivado Design Suite

The Vivado Design Suite by Xilinx offers comprehensive DFT capabilities, including built-in support for scan chains and BIST. It integrates well with Xilinx FPGA flows, providing a seamless environment for design and test. Vivado's DFT features are particularly useful for FPGA designs, where testability can be a significant challenge. The suite supports various DFT techniques and provides tools for automatic scan insertion, test pattern generation, and fault simulation [17].

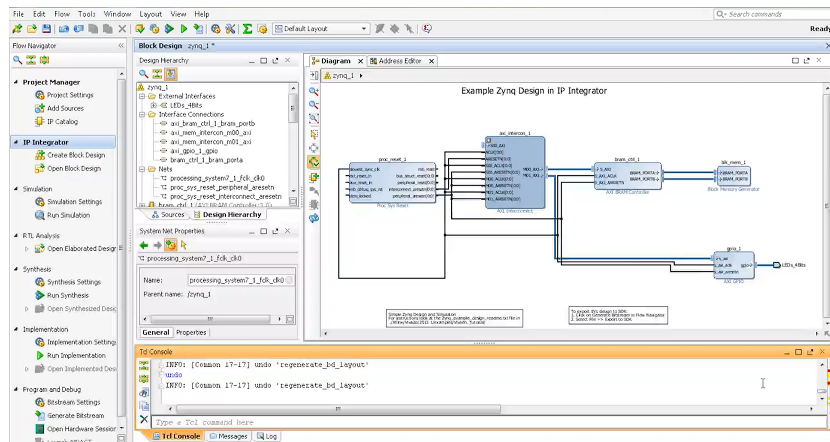

Figure 7: Xilinx Vivado Design Suite[7]

DFTMax

DFTMax, developed by Synopsys, is focused on optimizing testability in complex IC designs. It offers features for scan insertion, test compression, and ATPG, aiming to reduce test cost and time while maintaining high fault coverage. DFTMax is designed to work seamlessly with the Synopsys Design Compiler, providing a comprehensive solution for DFT implementation. The tool supports advanced test compression techniques that significantly reduce test data volume, making it an efficient solution for large-scale designs [14].

2.3.2 Limitations of These Tools

While the current tools provide significant advantages in scan-chain testing, they also have limitations that need to be addressed to further improve the testing process.

Scalability Issues

As IC designs become more complex, scalability becomes a significant challenge. Tools like Synopsys Design Compiler and Cadence Encounter Test may struggle with very large designs, leading to increased computation times and resource consumption. These scalability issues often manifest in longer synthesis times and increased memory usage, which can be prohibitive for very large designs. Optimizing the algorithms and improving the efficiency of these tools can help mitigate these scalability issues.

Integration Complexity

Integrating DFT tools into existing design flows can be complex and time-consuming. Each tool has its proprietary formats and requirements, making seamless integration a challenging task. This complexity is particularly noted with tools like Mentor Graphics Tessent and Xilinx Vivado Design Suite, which may require significant customization and configuration to fit into an existing workflow. This issue can be mitigated by improving tool interoperability and developing standardized interfaces [18].

High Costs

The cost of licensing and maintaining these tools can be prohibitively high, especially for smaller companies and academic institutions. High costs are associated with tools like Synopsys Design Compiler and DFTMax. The cost factor is a significant barrier

to entry for many organizations, limiting their ability to implement advanced DFT methodologies. Research suggests that developing cost-effective solutions and offering flexible licensing options can help address this issue.

Learning Curve

The steep learning curve associated with mastering these tools can be a barrier to their effective use. Comprehensive training and experience are required to utilize their full capabilities, which can be a hindrance for new users. This challenge is often mentioned in user feedback and technical training materials. To address this, tool vendors could provide more intuitive user interfaces and comprehensive training programs to help users get up to speed more quickly [19].

Limited Support for Emerging Technologies

While current tools are powerful, they may not fully support emerging technologies like machine learning-based test pattern generation and advanced low-power testing techniques. This limitation can hinder innovation and the adoption of new methodologies. As new testing techniques and technologies emerge, tool vendors need to update their tools to support these advancements. This requires continuous investment in research and development to stay ahead of technological trends.

2.4 Register-Transfer Level (RTL) Design

2.4.1 RTL Design

RTL (Register-Transfer Level) design is a critical phase in digital circuit design, bridging the gap between high-level abstract descriptions and low-level physical implementations. This chapter delves into the fundamentals of RTL design, its selection, configuration, and integration with other tools.

Register-Transfer Level (RTL) design is an abstraction used in the design and description of digital circuits. It focuses on how data moves between registers and how logical operations are performed on that data during each clock cycle. This level of abstraction is crucial as it allows designers to define the precise behavior of digital systems before moving to the gate-level design, which deals with individual logic gates and their interconnections.

A key aspect of RTL design involves the incorporation of scan chains, which are essential for testing and debugging digital circuits. Scan chains enable the serial shifting of test data through flip-flops in a circuit, allowing for the detection and diagnosis of faults during the testing phase. Integrating scan chains at the RTL level ensures that the design is prepared for effective testability, which is crucial for identifying and rectifying potential issues early in the design process.

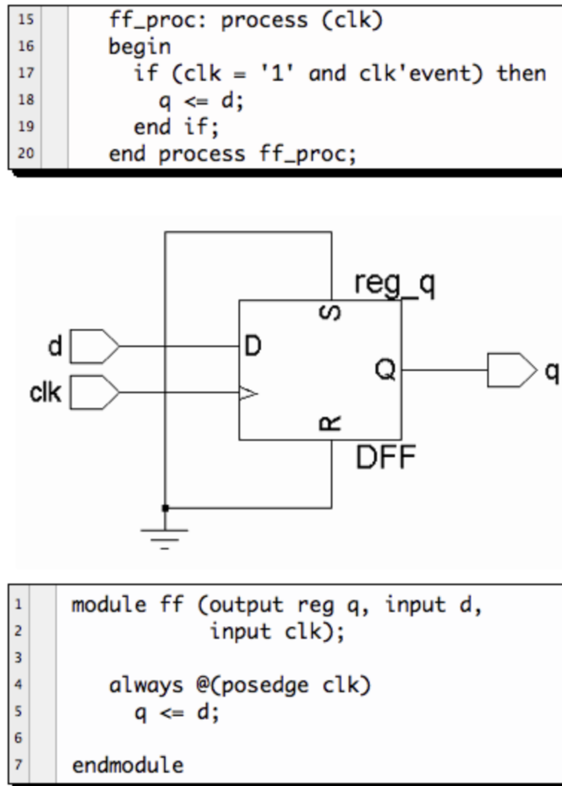

Figure 8: RTL Design Example: D Flip-Flop in VHDL and Verilog[8]

RTL design describes the flow of data within a digital system at the register level. It uses hardware description languages (HDLs) like Verilog and VHDL to define the system's functionality, allowing for precise simulation and verification before hardware synthesis. By working at the RTL level, designers can optimize the system for performance, power consumption, and area.

Importance of RTL Design

RTL design plays a pivotal role in the development of complex integrated circuits (ICs) and application-specific integrated circuits (ASICs). It provides a balance between high-level functional modeling and detailed hardware implementation, enabling early detection and correction of design errors, thus saving time and reducing costs in the development cycle [20] [21].

RTL Design Flow

The RTL design flow involves several key steps: Specification: Define the system's functionality, performance requirements, and constraints. Modeling: Use HDLs to create an abstract model of the system. Simulation: Verify the model's functionality through simulations. Synthesis: Convert the RTL model into a gate-level netlist. Implementation: Map the netlist onto physical hardware, such as FPGAs or ASICs. Verification: Perform post-synthesis and post-implementation verification to ensure correctness.

2.4.2 Selection of RTL Design Tools

Choosing the right tools for RTL design is crucial for successful digital system development. This section outlines the criteria for selecting RTL design tools and discusses some of the most widely used tools in the industry.

Criteria for Selection

When selecting RTL design tools, several factors must be considered: Compatibility: The tool should be compatible with existing design environments and other tools used in the development process. Ease of Use: The tool should have an intuitive interface and comprehensive documentation to facilitate learning and usage. Simulation Capabilities: Effective simulation is essential for verifying the functionality of RTL designs before synthesis. Synthesis Quality: The tool should produce optimized gate-level netlists that meet design specifications for performance, power, and area.

Popular RTL Design Tools

- Verilog: Known for its simplicity and ease of use, widely utilized for RTL design and verification.
- VHDL: Offers robustness and strong typing, suitable for complex design verification.
- SystemVerilog: Extends Verilog with advanced verification features, providing a comprehensive environment for design and testbench development.
- Vivado: A tool by Xilinx that offers a complete environment for RTL design, simulation, and synthesis.
- Quartus: Developed by Intel (formerly Altera), it provides a comprehensive suite for FPGA design, including RTL simulation, synthesis, and timing analysis [22] [23] [24].

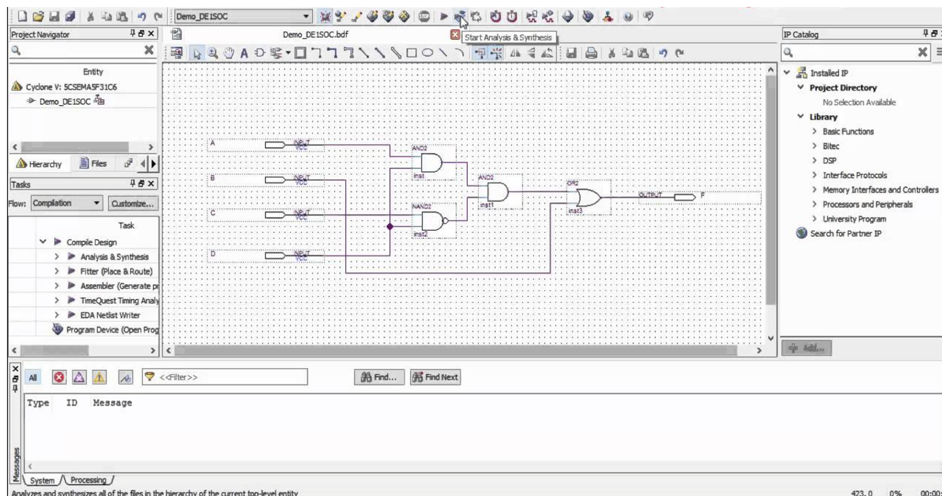

Figure 9: Quartus[9]

2.4.3 Integration and Customization

Integration of RTL design with other tools and customization for specific project needs are crucial for optimizing the design workflow and achieving the desired performance

targets.

Integration with Other Tools

RTL design tools often need to integrate with various other tools in the electronic design automation (EDA) workflow. This integration ensures a seamless design process from specification to implementation: Synthesis Tools: Convert RTL designs into gate-level netlists. Examples include Synopsys Design Compiler and Cadence Genus. Simulation Tools: Verify the functionality of the design before synthesis. Examples include ModelSim and VCS. Timing Analysis Tools: Ensure that the design meets timing requirements. Examples include Synopsys PrimeTime and Cadence Tempus [24] [25].

Customization for Specific Projects

Customizing RTL designs involves tailoring the design and tools to meet specific project requirements. This includes: Parameterization: Making the design adaptable to different configurations through parameters. Optimization: Tweaking the design to optimize for power, performance, or area based on the project's goals. Automation: Using scripts and automation tools to streamline repetitive tasks in the design flow.

Advanced RTL Design Techniques

To enhance the RTL design process, various advanced techniques can be employed: High-Level Synthesis (HLS): Converts high-level algorithmic descriptions directly into RTL code, speeding up the design process. Low Power Design Techniques: Implementing strategies like clock gating and power gating to reduce power consumption. Design for Testability (DFT): Incorporating features that make it easier to test the manufactured hardware for defects [26] [27].

2.5 Qflow and its Role in Testing

2.5.1 Introduction to Qflow

Qflow is an open-source digital synthesis flow tool that is extensively used in VLSI design for managing the entire flow from Register Transfer Level (RTL) to Graphic Data System II (GDSII). It integrates various other open-source tools such as Yosys for logic synthesis, Graywolf for placement, Qrouter for routing, Magic for layout, and Netgen for layout versus schematic (LVS) checks. This comprehensive toolchain facilitates the design and testing of Application-Specific Integrated Circuits (ASICs) and Field-Programmable Gate Arrays (FPGAs) by providing a seamless workflow for digital synthesis [28, 29].

Figure 10: Qflow[10]

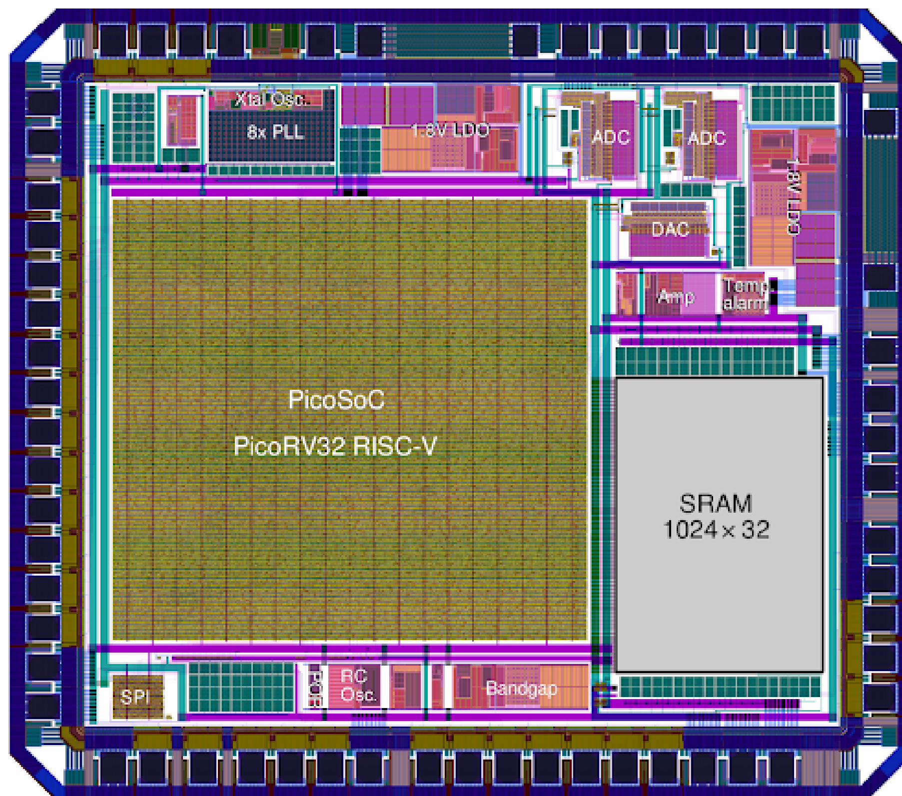

Figure 11: Raven RISC-V microprocessor[10]

Qflow's ability to manage the entire digital synthesis process makes it an invaluable resource for both academic and industry applications. It supports a variety of design technologies and can be customized to meet specific project requirements. The integration of multiple tools into a single framework simplifies the design process, reduces errors, and enhances productivity [30].

Application to the Raven RISC-V Microprocessor

One notable application of Qflow is in the design and testing of the Raven RISC-V microprocessor, a small, efficient, and open-source implementation of the RISC-V

architecture. Raven serves as a practical example of how Qflow can be used to manage the digital synthesis of complex processor designs. By utilizing Qflow, designers can take the Raven RISC-V microprocessor from RTL to a physical layout, ensuring that the processor adheres to design specifications and operates efficiently.

The Qflow toolchain, with its integrated synthesis, placement, and routing tools, enables the streamlined development of the Raven microprocessor. The flexibility and open-source nature of both Qflow and Raven highlight the potential for innovation in processor design without the need for costly proprietary tools. This synergy between Qflow and Raven fosters a more accessible approach to microprocessor design, particularly in educational and research environments where resources may be limited.

Furthermore, Qflow's support for various process technologies, including those used in the fabrication of the Raven RISC-V microprocessor, makes it a suitable choice for developing low-power and highly efficient processors. The Raven microprocessor's design goals align with Qflow's capabilities, demonstrating the effectiveness of this toolchain in achieving optimized VLSI designs[31]

Qflow also supports a wide range of process technologies and has been successfully used in educational and research settings to demonstrate the entire digital design flow. It enables users to create complete design layouts from RTL descriptions, making it a versatile and practical tool for VLSI design [32].

Furthermore, Qflow includes a set of examples and tutorials that make it easier for new users to get started. These resources are invaluable for educational purposes, providing step-by-step guidance through the design process [33]. The toolchain's compatibility with various design technologies, including the OSU035 and OSU018 process design kits (PDKs), further broadens its applicability in both academic and professional settings [34].

Qflow's significance also extends to its role in promoting the open-source ecosystem within the EDA community. By leveraging open-source tools, Qflow reduces the dependency on expensive proprietary software, making advanced VLSI design accessible to a broader audience. This democratization of technology fosters innovation and encourages collaborative development, which is vital for the continuous advancement of the field [28].

2.5.2 How Qflow Addresses Existing Problems

Qflow addresses several challenges in the digital synthesis flow by integrating various tools that automate different stages of the design process. One of the key problems in VLSI design is the complexity and time consumption associated with manual design and testing processes. Qflow mitigates these issues through automation and the use of open-source tools, which are cost-effective and highly customizable.

For example, Qflow's use of Yosys for logic synthesis allows for efficient translation of high-level design descriptions into gate-level representations. Yosys is known for its robust synthesis capabilities and supports a wide range of input formats, making it a versatile tool for different design requirements [35]. Graywolf's placement capabilities ensure that the synthesized logic is optimally placed on the chip, while Qrouter handles

the routing of interconnections between placed components. Magic provides a platform for layout design, enabling designers to visualize and refine the physical implementation of their circuits [30, 31].

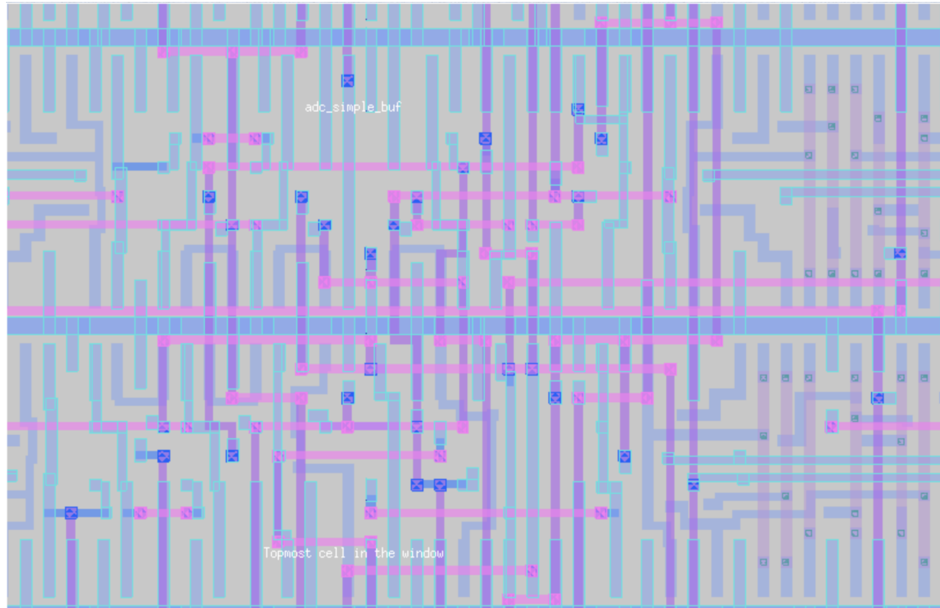

Figure 12: Detail of a layout generated by qrouter[11]

Qflow also enhances testing and verification processes through tools like Netgen, which performs LVS checks to ensure that the layout matches the schematic. This comprehensive approach to digital synthesis and verification reduces the likelihood of design errors and improves the overall reliability of the final product [30].

Moreover, Qflow's open-source nature allows for continuous improvement and community support. Users can contribute to the development and refinement of the tool, ensuring that it stays up-to-date with the latest advancements in VLSI design. This collaborative approach has led to the integration of new features and optimizations, making Qflow a robust solution for digital synthesis [33].

Qflow's ability to integrate with other tools and scripts also allows for the automation of repetitive tasks, which significantly reduces the time required for design iterations. This feature is particularly beneficial in educational settings, where students can focus more on learning design principles rather than getting bogged down by tedious manual processes [34].

Additionally, Qflow addresses the challenge of accessibility in VLSI design by providing a no-cost alternative to commercial EDA tools. This accessibility is crucial for academic institutions and small enterprises that may not have the budget for expensive software licenses. By offering a comprehensive suite of tools that cover the entire digital design flow, Qflow enables a wider range of users to engage in VLSI design and innovation [28].

2.5.3 Qflow in Comparative Context

When compared to other commercial digital synthesis tools, Qflow offers a significant advantage in terms of cost and flexibility. Commercial tools like Synopsys Design Compiler and Cadence Encounter Test are powerful but often come with high licensing fees and limited customization options. In contrast, Qflow, being open-source, is freely available and can be tailored to meet specific design needs.

Qflow's integration with other open-source tools further enhances its flexibility. For instance, Yosys and Magic are widely recognized for their robust performance in logic synthesis and layout design, respectively. The ability to use these tools within the Qflow framework provides a seamless and efficient workflow for VLSI designers [28, 30].

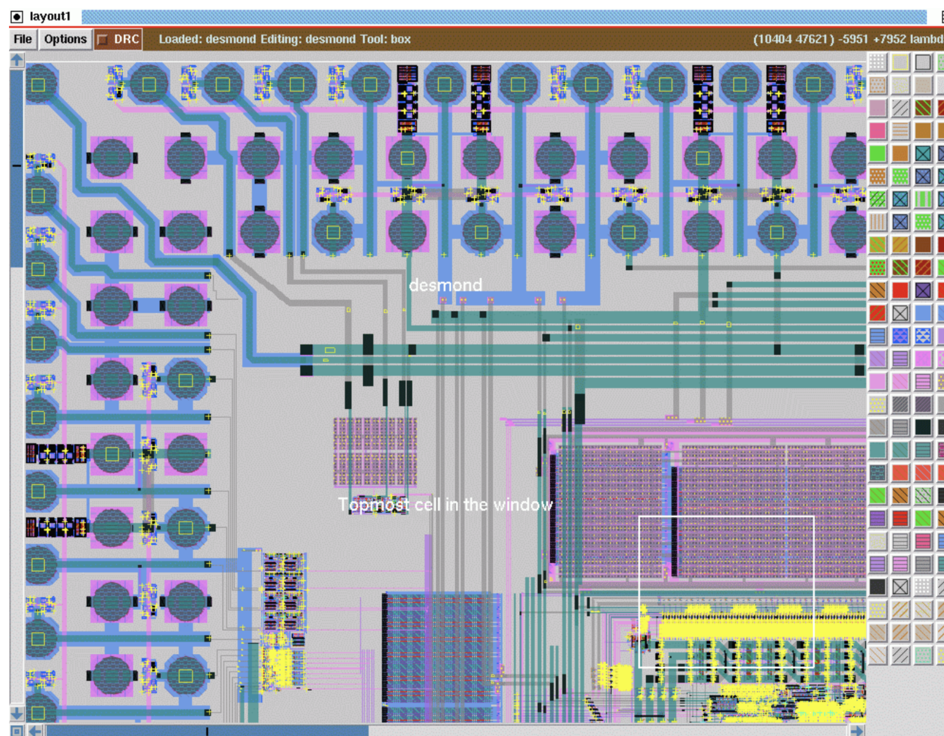

Figure 13: Viewing, extraction, DRC checks, GDS Generation with Magic[12]

Additionally, Qflow's support for various technologies and its adaptability to different design environments make it a valuable tool for both educational purposes and professional VLSI design projects. While commercial tools may offer advanced features and higher performance, Qflow provides a cost-effective and accessible alternative that can achieve comparable results for many applications [29].

Qflow also excels in educational settings, where the open-source nature of the tool allows students to understand the complete design flow from RTL to GDSII. This hands-on experience with a real-world toolchain prepares students for industry practices and enhances their learning outcomes [28].

Furthermore, the open-source community around Qflow ensures that the tool is continuously updated and improved. This community-driven development model allows for rapid incorporation of new features and bug fixes, keeping Qflow relevant and effective in the ever-evolving field of VLSI design [30].

In comparison to commercial tools, Qflow's transparency in terms of its underlying algorithms and processes provides users with a deeper understanding of the digital synthesis flow. This transparency is crucial for educational purposes, as it allows students and researchers to explore and modify the tool to suit their specific needs [28].

Moreover, Qflow's flexibility and adaptability extend to its compatibility with various process design kits (PDKs), enabling users to design for different manufacturing technologies. This versatility is essential for research and development projects that require experimentation with different process nodes and design rules [34].

In the professional context, Qflow's open-source nature allows companies to customize the toolchain to fit their specific workflows and design methodologies. This customization capability can lead to significant cost savings and efficiency improvements, as companies can tailor the tool to their unique needs without relying on vendor-specific solutions [35].

Overall, while Qflow may not have all the advanced features of high-end commercial tools, its open-source nature, flexibility, and comprehensive toolchain make it a valuable asset in the VLSI design ecosystem. It bridges the gap between accessibility and functionality, providing an effective platform for both learning and professional development in digital synthesis and design [28, 30].

2.6 Problem Statement

2.6.1 Detailed Definition of the Problem

The efficient integration of scan chains into VLSI designs is crucial for enhancing testability while minimizing negative impacts on circuit performance. As integrated circuits become increasingly complex, the challenges associated with scan chain implementation are becoming more pronounced, particularly in terms of power consumption, design complexity, routing costs, and achieving the shortest possible routing distance. Effective design strategies, such as graph-based approaches to scan chain stitching, are essential in addressing these issues [13].

Design-for-Testability (DFT) Challenges

DFT techniques, including scan chains, are fundamental in modern VLSI design, providing structured test methodologies that facilitate fault detection and diagnosis. Scan chains transform flip-flops into serially connected elements, allowing test patterns to be shifted in and out of the circuit to test its sequential and combinational logic. However, the integration of scan chains introduces additional complexity, particularly in managing the overhead of routing and ensuring that the scan paths do not interfere with normal circuit operation. Effective DFT implementation must balance testability with minimal impact on performance, area, and power consumption, presenting a significant challenge to designers.

Power Consumption

Power consumption during scan-based testing is a major concern, especially for large-scale integrated circuits. The shifting operations inherent to scan chains can lead to high switching activity, significantly increasing power dissipation during testing. This is particularly problematic in low-power and battery-operated devices where excessive

power consumption can impact performance and product longevity. Techniques such as clock gating, low-power test pattern generation, and scan chain segmentation are employed to reduce power overhead. However, optimizing these methods to achieve minimal power consumption while maintaining high fault coverage remains an ongoing challenge [36].

Design Complexity and Routing Costs

The integration of scan chains significantly increases the overall complexity of VLSI designs, as additional routing is required to connect sequential elements efficiently. Routing costs include increased layout area, potential congestion, and the need for additional routing layers to accommodate scan paths. Efficient routing strategies, such as the graph-based approaches highlighted by Zaourar et al., are essential to minimize these costs. By using graph models to infer logical proximity and employing TSP-based algorithms for optimal scan-stitching ordering, designers can reduce routing overhead and ensure that the scan chains do not degrade circuit performance [13].

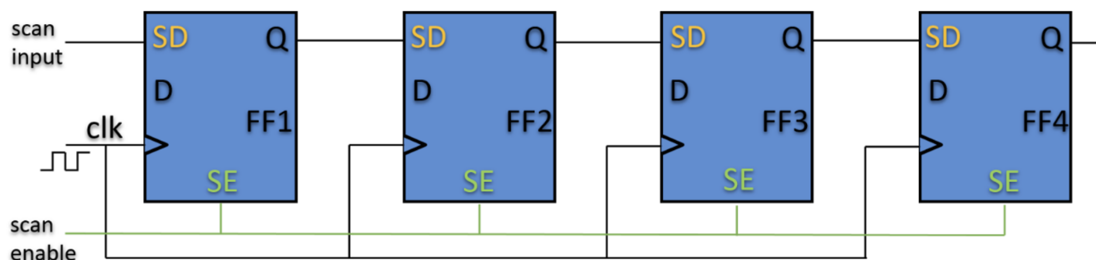

Figure 14: Scan Chain Connection[13]

Minimizing Scan Chain Route Distance

Minimizing scan chain route distance is critical for reducing test time, lowering power consumption, and improving overall test efficiency. Shorter routing paths facilitate faster data shifting and reduced power dissipation. Techniques based on the Traveling Salesman Problem (TSP) are used to optimize scan chain configurations, as they effectively minimize the total wirelength of scan chains, directly impacting the efficiency and cost of the testing process. The graph-based approach proposed by Zaourar et al. demonstrates significant reductions in routing costs by determining optimal scan-stitching orders using logical proximity information derived from RTL design descriptions [37].

Balancing Performance and Testability

The primary challenge in scan chain design is achieving a balance between circuit performance and robust testability. This involves managing power consumption, reducing routing complexities, and optimizing scan chain configurations to minimize overhead. Advanced DFT methodologies, such as those integrating graph-based stitching algorithms, play a critical role in meeting these objectives, enabling the integration of scan chains that enhance test coverage while preserving circuit performance.

Ensuring Reliable Integration of Scan Chains

Reliable integration of scan chains requires careful consideration of timing, signal integrity, and power distribution within the circuit. The graph-based stitching ap-

proach, which leverages logical proximity information, allows for more efficient routing and integration of scan chains without impacting the normal operation of the circuit. This method enhances the reliability and efficiency of scan chain designs, ensuring that potential timing and power issues are addressed before manufacturing [38].

2.6.2 Importance of Addressing This Problem

Addressing the specific challenges associated with scan chains is crucial for optimizing the testability and overall efficiency of VLSI circuits. The integration of advanced techniques, such as the Traveling Salesman Problem (TSP) combined with nearest neighbor heuristics, provides a structured approach to minimizing scan chain lengths, thereby reducing testing time and power consumption. This optimization is not about creating new algorithms but enhancing existing methods to address the growing complexities of modern integrated circuits [39].

Significance of Scan Chains: Scan chains are integral to VLSI testing methodologies, allowing access to internal circuit states that are critical for fault detection and debugging. Their effective implementation directly affects the quality of the testing process, reducing the likelihood of defects reaching production. By optimizing the scan path using heuristic techniques, the time and resources required for testing can be significantly reduced, improving both cost efficiency and reliability [41].

Power Management: Power consumption during scan testing is a critical issue, especially as IC designs become more compact and power-sensitive. Effective scan chain optimization can lower dynamic power usage by reducing the number of shifts required during testing. Techniques that minimize the scan chain path length, such as those using TSP with nearest neighbor, directly contribute to a more power-efficient testing environment [40].

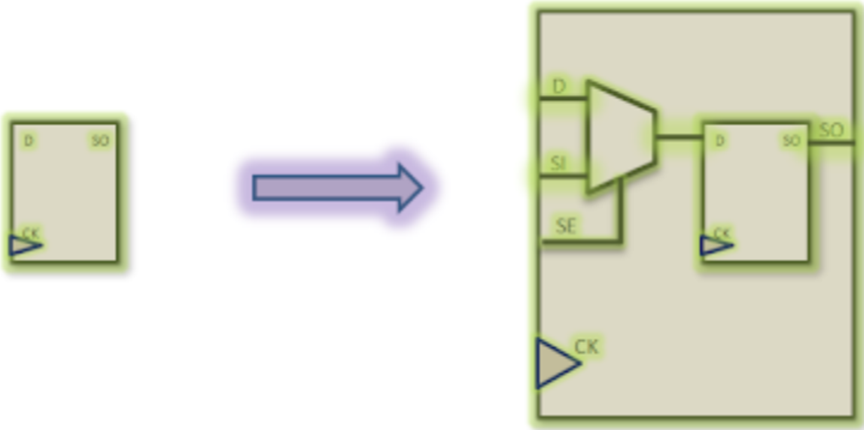

Figure 15: Transition of a normal flop to scan flop[14]

Design and Routing Complexity: Integrating scan chains into VLSI designs adds complexity, particularly in routing, which can increase congestion and affect signal integrity. Optimizing the scan chain layout with methods derived from TSP helps

streamline the routing process, making it easier to manage design constraints such as area and timing without compromising test coverage.

Reducing Scan Chain Route Distance: The primary objective of using TSP with nearest neighbor is to minimize the route distance of scan chains, which correlates with faster testing times and reduced overhead. Shorter scan paths allow for quicker data shifting, reducing both dynamic power consumption and the overall test duration. This approach directly addresses one of the key pain points in scan chain design: the balance between performance and testability [39].

Balancing Performance and Testability: One of the ongoing challenges in scan chain design is maintaining a balance between optimal test coverage and minimal performance impact. By employing heuristic optimization methods, designers can configure scan chains that are efficient without compromising the functionality or speed of the IC. This delicate balance is crucial for ensuring that high-performance circuits remain reliable and testable [41].

2.6.3 Expected Innovations from Addressing the Problem

The continuous refinement of scan chain optimization techniques, particularly through the use of TSP with nearest neighbor, is expected to yield several key advancements in the field of VLSI design and testing.

Improved Scan Chain Configurations: By leveraging heuristic optimization techniques, scan chains can be configured more effectively, reducing test times and improving power efficiency. This is not a novel algorithm but an adaptation of existing optimization strategies to better suit the constraints of modern VLSI designs [39].

Enhanced Power Management Techniques: The reduction of scan path lengths directly contributes to lower power consumption during testing. Future advancements may further refine these techniques, integrating more sophisticated power-aware optimizations that adapt dynamically to the testing environment [40].

Advanced Placement and Routing Strategies: The integration of TSP-based optimizations into the design workflow can lead to better placement and routing of scan chains, reducing the overall impact on design performance. Innovations in this area are expected to streamline the testing phase, making it more compatible with high-speed and high-density designs [41].

Increased Scalability and Flexibility: As VLSI designs continue to grow in complexity, the scalability of scan chain testing will be crucial. Techniques that minimize routing and power overheads will enable more extensive and complex designs to be tested efficiently, reducing bottlenecks in the design-to-production pipeline [39].

3 Research Methodology and Framework

3.1 Overview of Methodological Approach

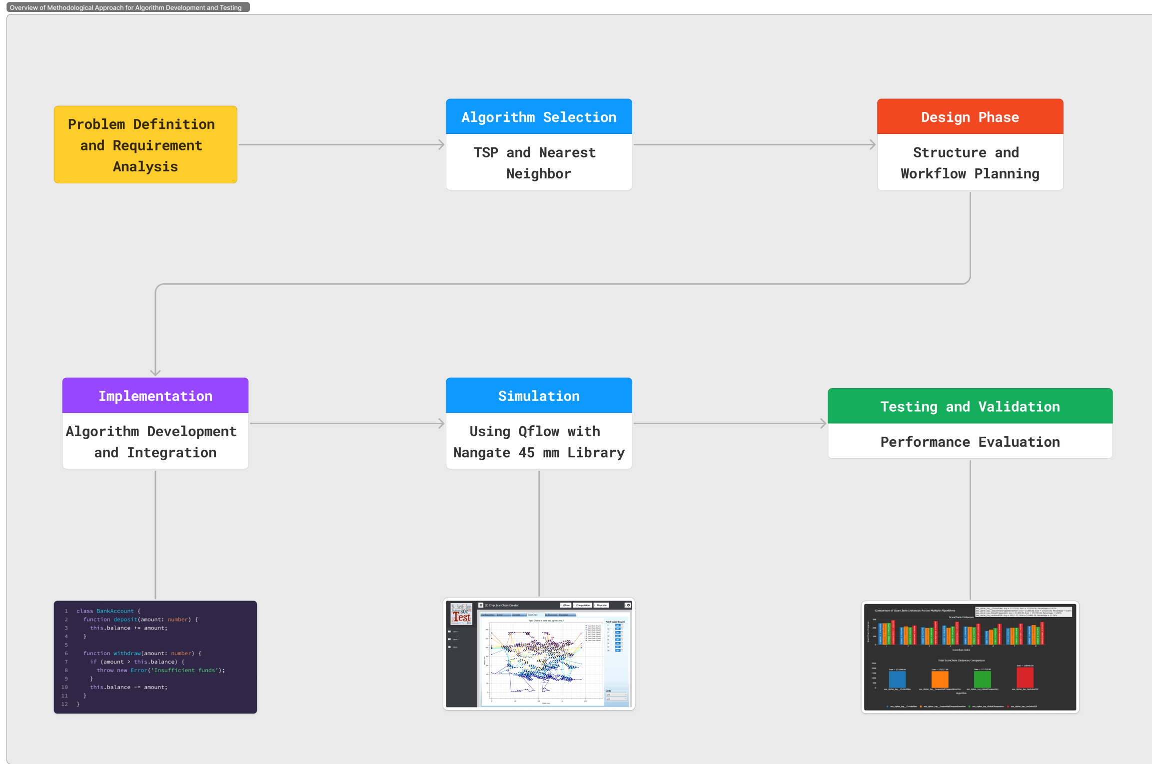

Figure 16: Overview of Methodological Approach for Algorithm Development and Testing

3.1.1 Rationale Behind Chosen Methods

The rationale for the methods employed in the LeoSolveTSP function centers on the primary objective of achieving an effective configuration of scan chains. This approach prioritizes minimizing the overall length of scan chains to enhance the efficiency and performance of the testing process. While solving the Traveling Salesman Problem (TSP) plays a crucial role, the emphasis lies in tailoring heuristic methods to specifically address the complexities of scan chain optimization. By arranging the scan chains in the shortest possible configuration, this methodology directly contributes to reduced test path lengths, ultimately leading to more efficient and reliable testing outcomes.

Understanding the Traveling Salesman Problem (TSP): The TSP is a well-known problem in combinatorial optimization, where the goal is to find the shortest possible route that visits each given point exactly once and returns to the starting point. This problem is NP-hard, implying that as the number of points increases, the time required to compute the exact solution grows exponentially. For large instances, finding an exact solution becomes computationally infeasible, necessitating the use of heuristic methods [43, 44].

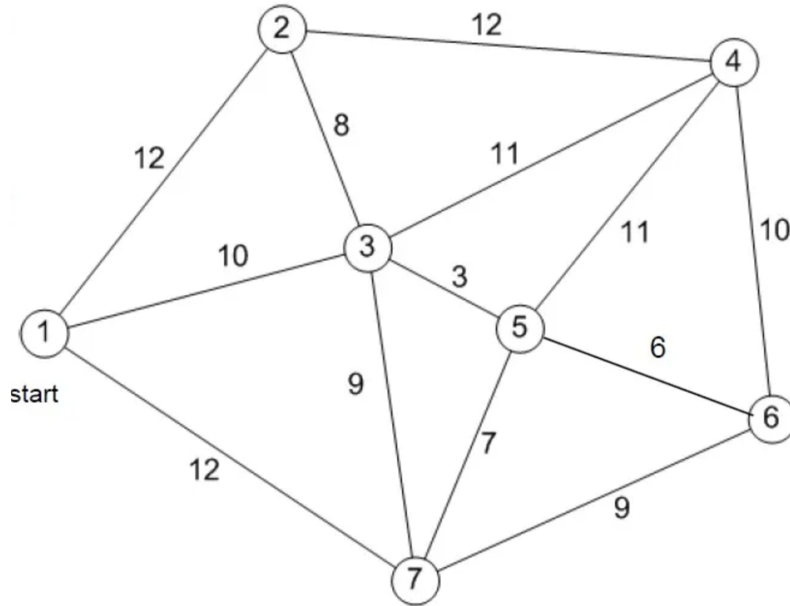

Figure 17: Traveling Salesman Problem[15]

Importance in Integrated Circuits: In the context of integrated circuits, optimizing scan chains is crucial for reducing overall testing time and enhancing the efficiency of the manufacturing process. Scan chains facilitate the testing of internal circuit elements by connecting them in a specific sequence that allows for efficient input and output of test data. The effectiveness of scan chain testing is highly dependent on the order in which these elements are connected; a poorly optimized sequence can result in longer paths, increased test times, and higher power consumption, negatively affecting both speed and cost-effectiveness. Optimizing the order of scan chain elements focuses on minimizing the physical and logical distances between connected components, directly improving the performance of the testing process. Shorter scan chains mean fewer clock cycles are needed to shift test data across the chain, significantly reducing the time required for testing and the dynamic power consumed during the process. This optimization problem is effectively addressed using techniques derived from the Traveling Salesman Problem (TSP), which seeks the shortest route connecting all nodes. By applying TSP-based strategies, such as heuristic approaches, the scan chain length is minimized, leading to substantial gains in testing efficiency. The benefits of this optimization extend beyond just time reduction. Shorter and more efficient scan chains reduce the toggling activity of circuit elements, which is particularly important in power-sensitive applications, where excessive switching can lead to increased heat and potential reliability issues. Furthermore, optimized scan chains improve the accuracy and coverage of the testing process, as reduced distances allow for more precise application of test patterns. In addition to performance improvements, the economic impact of scan chain optimization is significant. Faster testing cycles mean lower operational costs and reduced time-to-market, which are critical in the highly competitive semiconductor industry. As ICs continue to scale in complexity and size, with more nodes and elements to test, the ability to efficiently configure scan chains becomes increasingly important. Advanced TSP-based optimization techniques, such

as zone-based partitioning, further enhance scalability by dividing large circuits into smaller sections that are optimized locally before being globally integrated, ensuring that even the most complex IC designs can benefit from reduced testing overhead. [42].

Nearest Neighbor Heuristic: The Nearest Neighbor (NN) heuristic is one of the simplest and most intuitive methods for finding a feasible solution to the TSP. It starts at an arbitrary node and repeatedly visits the nearest unvisited node until all nodes have been visited. Key benefits of using the NN heuristic include:

- **Simplicity:** The algorithm is easy to understand and implement, making it suitable for rapid development and testing.
- **Speed:** NN provides a quick solution, which is often good enough for practical purposes, especially with a moderate number of nodes.
- **Baseline for Improvement:** The solution obtained from NN can serve as a baseline that can be further refined using more sophisticated optimization techniques if necessary [44].

While the NN heuristic does not guarantee an optimal solution, it often produces a route that is reasonably close to the optimal one, particularly when the points are uniformly distributed.

3.1.2 Research Design

The research design for developing the LeoSolveTSP function involves a systematic approach to ensure that the algorithm is effective, efficient, and integrates seamlessly with the existing ScanChainTool. The objective is to optimize the scan chain configuration by minimizing the total length of connections, which directly impacts testing speed and power efficiency in integrated circuits.

Phases of Research Design include Problem Definition and Requirement Analysis, where specific requirements for the scan chain optimization tool are defined. Constraints related to the hardware environment and the testing process are identified, including considerations such as computational limitations and the need for high-speed data shifting. Criteria for evaluating the performance of the optimization algorithm are established to ensure that the solution meets both speed and quality standards.

Algorithm Selection and Justification involves reviewing various algorithms for solving the TSP, including exact methods such as dynamic programming and branch-and-bound, as well as heuristic methods like Nearest Neighbor (NN), genetic algorithms, and simulated annealing. The NN heuristic is selected based on its balance between computational efficiency and solution quality, making it particularly suitable for the dynamic and iterative nature of scan chain optimization [44].

The figures presented (Figures 16-18) illustrate the core concept of the LeoSolveTSP function. Figure 16 shows a generated area with 30 nodes, representing a typical set of scan cells in a circuit. Figure 17 demonstrates a randomly connected path, highlighting the inefficiencies in traversal distance with a total path length of 459.26 centimeters. In contrast, Figure 18 shows the path optimized using the TSP Nearest Neighbor heuristic, reducing the total distance to 167.74 centimeters. This significant reduction underscores

the effectiveness of TSP-based optimization in minimizing scan chain lengths, directly leading to faster testing times and reduced power consumption.

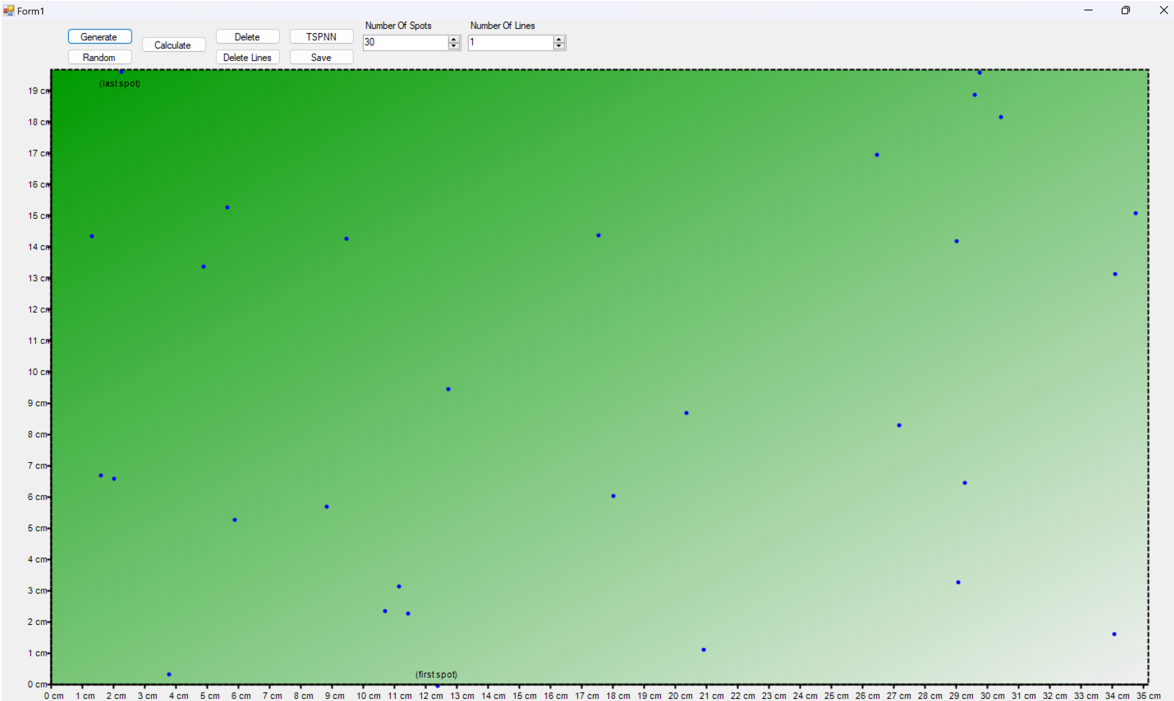

Figure 18: Area with 30 nodes

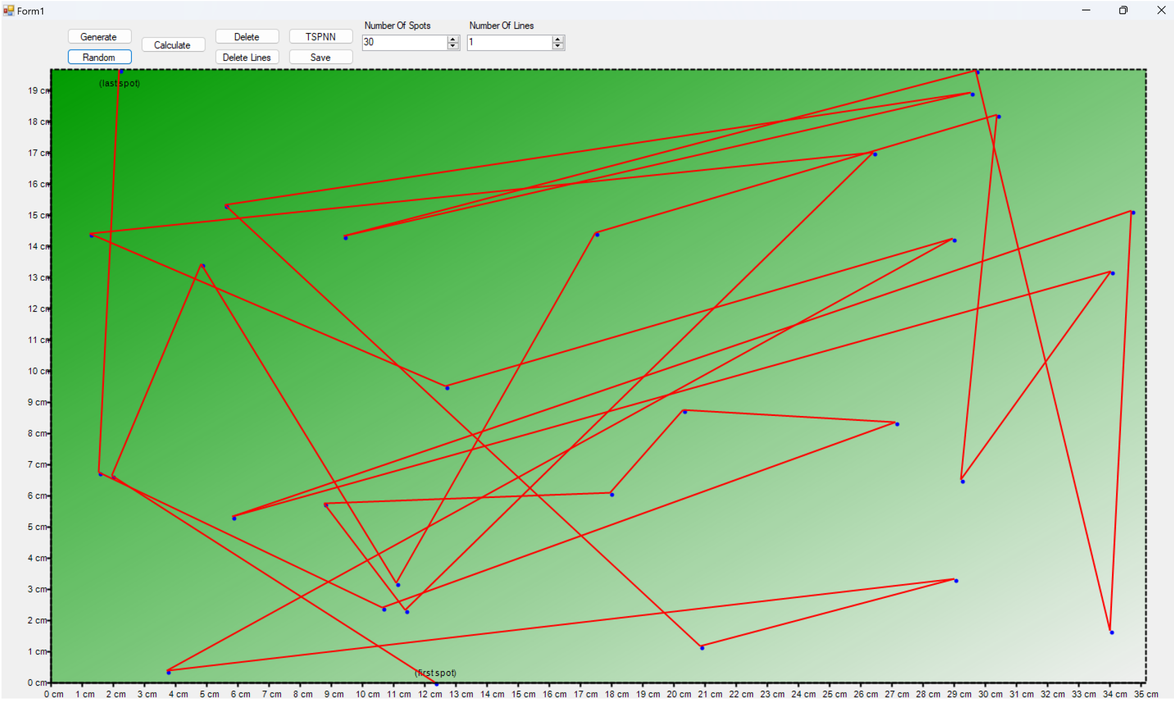

Figure 19: Random path

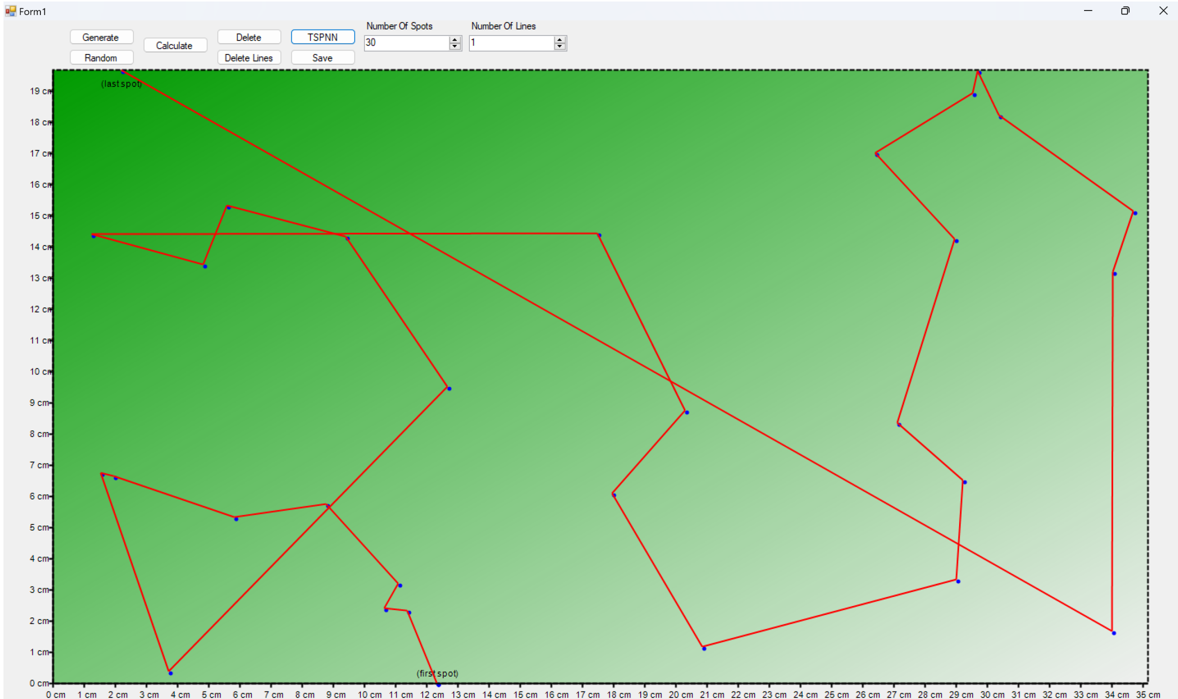

Figure 20: TSP Nearest Neighbour path

Path Type	Distance in centimeters
Random Path	Distance = 459.26
TSP Path	Distance = 167.74

Table 1: Comparison of Random Path and TSP Path Distances

During the Detailed Design phase, the architecture of the LeoSolveTSP function is meticulously outlined, including key components such as data structures for nodes, distances, and the solution path. These components are developed with modularity in mind to facilitate testing and future enhancements. The implementation phase involves coding the LeoSolveTSP function in C#, incorporating auxiliary functions such as CalculateDistance and CalculateTotalDistance, which use the Euclidean distance formula.

$$\text{Distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

Euclidean distance calculation formula

This formula calculates the direct distance between two points, ensuring that the algorithm accurately assesses and minimizes the total path length between scan chain elements.

The comparative analysis in Table 2 specifically highlights the LeoSolveTSP algorithm's effectiveness in reducing path lengths compared to a random path configuration. This comparison illustrates the algorithm's ability to significantly shorten scan chains, serving as an initial validation of its performance in optimizing scan chain lengths.

Testing and Validation involve a structured approach that includes performance evaluations to assess the LeoSolveTSP algorithm in comparison to other established algorithms. The algorithm, based on the Traveling Salesman Problem (TSP) and employing the Nearest Neighbor heuristic, is tested within a program that applies it to standard benchmark circuits commonly used in the field. To ensure consistency, the LeoSolveTSP algorithm is directly compared to other algorithms under the same conditions on the same benchmark circuits.

3.1.3 Methodological Framework

The methodological framework combines structured and iterative approaches to ensure a robust and flexible development process.

Hybrid Waterfall and Agile Approach: Waterfall elements include Requirement Analysis, where a detailed analysis of the requirements is conducted at the outset to ensure a clear understanding of the problem and the expected outcomes, and Design, where a comprehensive design is developed before proceeding with implementation, ensuring that all aspects of the algorithm and its integration are well thought out.

Agile elements include Iterative Development, where the algorithm is implemented in small, manageable increments, allowing for continuous feedback and refinement. Continuous Testing is conducted at each stage of development to identify and address issues early, ensuring the final product is of high quality. User Feedback is incorporated throughout the development process to ensure the tool meets their needs and expectations.

Detailed Methodological Steps involve Requirement Analysis, where stakeholders are engaged to gather detailed requirements and the requirements are documented to create a specification document. In Design, high-level and detailed designs are developed, including flowcharts and pseudocode, and the design is reviewed with stakeholders to make necessary adjustments. During Implementation, the code for the LeoSolveTSP function and auxiliary methods is written, and version control is used to manage the codebase and track changes.

In the Testing phase, unit tests for individual components are created, integration testing is performed to ensure components work together as expected, and performance testing is conducted to evaluate the algorithm's efficiency with large datasets. Finally, in Documentation and Training, user manuals and technical documentation are developed,

and training sessions and materials are provided to help users understand and use the tool effectively.

Advantages of the Hybrid Approach include comprehensive planning, where initial planning and design phases ensure that all requirements are considered, reducing the risk of major issues during implementation. Flexibility and responsiveness are achieved through iterative development, allowing for quick responses to changing requirements or unforeseen challenges. Continuous improvement is enabled by regular testing and user feedback, leading to a more robust and user-friendly tool [45].

3.2 Qflow Nandgate 45nm Libraries

3.2.1 Technical Specification of Libraries

The 45nm NanGate Open Cell Library is a comprehensive, open-source standard cell library designed to emulate a realistic 45nm process. Developed using the FreePDK45 process design kit from North Carolina State University, it is available under the Apache 2.0 license. This library includes various standard cells such as inverters, NAND, NOR, XOR gates, and different types of flip-flops and multiplexers, essential for creating complex digital circuits.

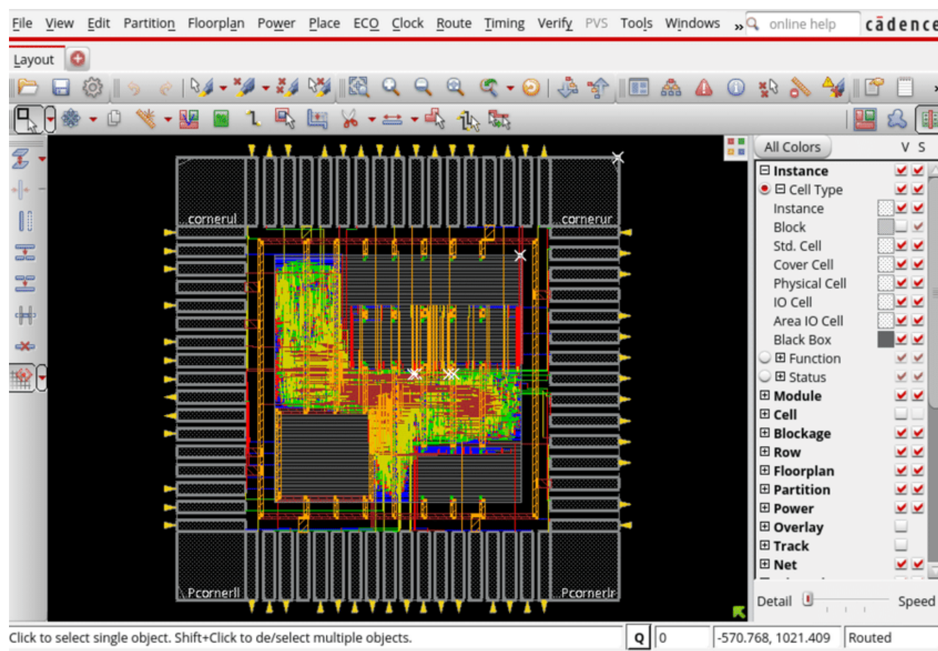

Figure 22: FreePDK45-Placement-and-Routing with Virtuoso from Cadence Design system[16]

Cell Architecture: Standard cells are designed with consistent height and well-defined pin locations, optimizing for area, performance, and power.

LEF/DEF Files: These files describe the layout and abstract views of the cells, which are crucial for placement and routing during physical design.

Liberty Timing Files: These files provide timing, power, and functional descriptions of the cells, essential for timing analysis and synthesis.

SPICE Models: Used for detailed circuit simulation, allowing for accurate power and performance characterization.

GDSII Files: Contain the final mask layout data of the cells, useful for detailed physical verification and manufacturing rule checks.

The layout data includes directories such as './lef' for technology and macro LEF files, './lib' for standard cell and macro liberty files, and './qrc' for the QRC technology file. This detailed organization helps in maintaining the integrity of the design and ensuring efficient workflow integration [46] [47].

3.2.2 Application in Research

The Qflow toolkit, integrated with the NanGate 45nm library, serves as a powerful platform for VLSI design education and research. Researchers and students utilize Qflow for several key applications:

Digital Circuit Synthesis: Tools like Yosys are used for synthesis, transforming Verilog descriptions into gate-level netlists using the 45nm standard cell library. This process involves logical optimization and technology mapping to the target 45nm cells [46].

Placement and Routing: Tools such as Graywolf for placement and Qrouter for routing ensure that the physical layout of the circuits meets design specifications. Placement determines the optimal positions for cells, while routing connects them electrically [48].

Timing Analysis: Integration with tools like QTiming allows for detailed timing analysis to ensure the design meets required performance specifications. This step verifies that the circuit operates correctly at the desired clock speed by analyzing setup and hold times [49].

Power Analysis: SPICE simulations with the provided models help evaluate the power consumption of designs. This includes dynamic, leakage, and short-circuit power, providing insights into the energy efficiency of the design [47].

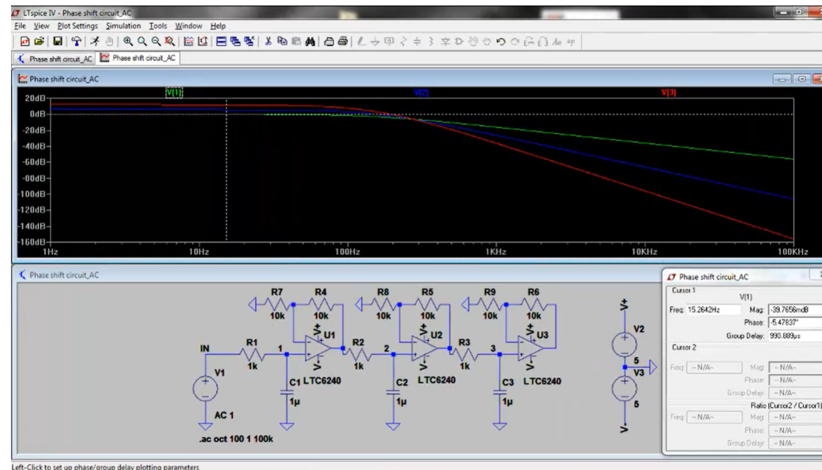

Figure 23: LT Spice Alternating Current(AC) Analysis[17]

Educational Projects: Qflow serves as an educational tool in VLSI design courses, providing students with hands-on experience with a complete open-source EDA flow. Students can experiment with different design approaches and gain practical insights into the digital design process [50].

In addition to these primary applications, Qflow's integration with the NanGate 45nm library enables detailed analysis and optimization of power, performance, and area, which are critical metrics in VLSI design. This integration supports advanced research topics such as low-power design techniques and high-performance computing architectures.

3.2.3 Evaluation of Performance

Designs implemented using the Qflow toolkit and the 45nm NanGate library are evaluated based on several performance criteria:

Timing: Accurate timing models in the Liberty files allow for precise timing closure, ensuring critical paths meet the required clock frequency. This is essential for ensuring that the design can operate at the intended speed without timing violations [47].

Power Consumption: Detailed power analysis using SPICE simulations helps understand the power efficiency of designs, including dynamic, leakage, and short-circuit power. This analysis is crucial for battery-powered devices and other power-sensitive applications [48].

Area Efficiency: The compact layout of standard cells in the 45nm technology node ensures high-density designs, which is critical for modern VLSI circuits. Efficient use of area can reduce manufacturing costs and improve performance [49].

Design Flexibility: The comprehensive set of standard cells and flexible Qflow tools support the implementation of various digital designs, from simple combinational circuits to complex sequential systems. This flexibility is important for adapting the tools to a wide range of applications.

Educational Value: The toolkit's open-source nature and detailed documentation make it invaluable for educational purposes, providing a realistic design experience without the costs associated with commercial tools. This hands-on experience is invaluable for preparing students for careers in semiconductor design and EDA tool development [50].

These performance metrics are critical for both academic research and practical design projects, providing a solid foundation for further advancements in digital design methodologies.

3.3 Tool Implementation (2D Chip ScanChain Creator)

Mr. Vartziotis tool, named "2D Chip ScanChain Creator," is designed to facilitate the integration and functionality of scan chains in the context of VLSI design and testing. The tool provides a comprehensive suite of features that enable efficient design-for-test (DFT) insertion, configuration, and analysis.

3.3.1 Integration and Functionality in the Research Context

The 2D Chip ScanChain Creator integrates seamlessly with the Qflow digital synthesis flow and other standard VLSI design tools. Its primary functionalities include DFT insertion, scan chain configuration, and advanced floorplanning. The tool is structured into several key modules:

Overall Architecture

The 2D Chip ScanChain Creator is structured to provide a comprehensive environment for scan chain design and optimization. The architecture consists of several interconnected modules, each responsible for specific aspects of the design process:

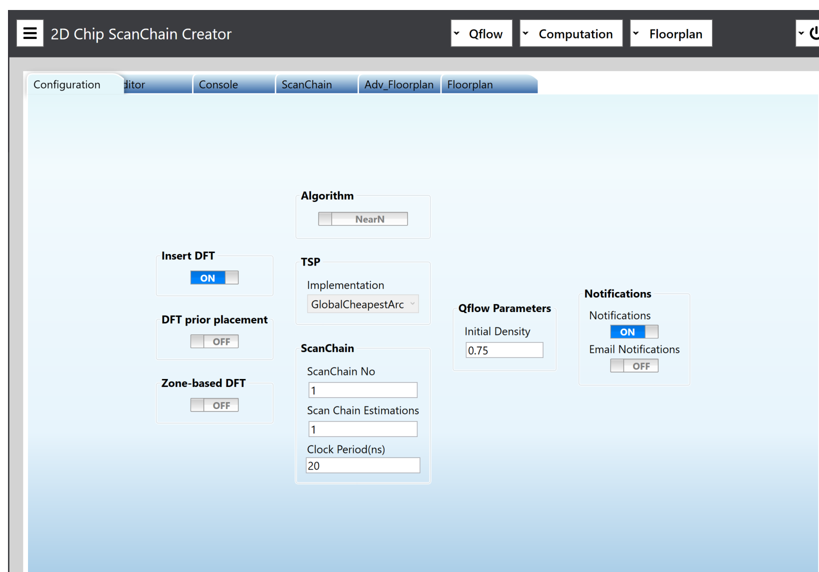

Figure 24: 2D Chip ScanChain Creator

Configuration Module: This module allows users to set the initial parameters for the design process. It includes options for enabling DFT insertion, selecting test algorithms, and configuring Qflow parameters. Users can also toggle specific features such as zone-based DFT, DFT prior placement, and notification settings. This flexibility ensures that the tool can adapt to various design requirements and constraints, providing a customizable environment for different project needs.

DFT Insertion: The tool supports the insertion of DFT structures at various stages of the design process. Users can enable or disable DFT insertion based on the specific needs of their design. This feature is crucial for ensuring that the design is testable and meets the required fault coverage metrics.

Algorithm Selection: Users can choose from a variety of algorithms for scan chain optimization, including GlobalCheapestArc, SequentialCheapestInsertion, Christofides, and FirstUnboundMinValue. Each algorithm offers distinct advantages depending on the specific design goals and constraints.

Scan Chain Configuration: Users can specify the number of scan chains, the number of estimations, and the clock period in nanoseconds.

Qflow Parameters: Initial density settings can be adjusted to control the placement density.

Notifications: Users can enable notifications and email updates.

Qflow Integration: The tool integrates tightly with the Qflow digital synthesis flow, providing a seamless interface for various stages of the design process. This integration includes:

- **Loading DUT:** The Design Under Test (DUT) can be loaded into the tool, allowing users to import and prepare their designs for further processing.
- **Running Synthesis:** The synthesis process converts high-level design descriptions into gate-level representations, optimizing the logic for the target technology.
- **Placement and Routing:** The placement phase determines the optimal positions for cells on the chip, while the routing phase connects the cells according to the design specifications. These steps are crucial for ensuring that the design meets performance and area constraints.
- **Static Timing Analysis (STA):** STA verifies that the design meets the required timing constraints, ensuring that the circuit operates correctly at the desired clock frequency.
- **Post-Route Timing Analysis:** This step re-evaluates the timing after routing to ensure that any changes during routing do not violate the timing constraints.
- **Migration:** This process adapts the design for manufacturing, ensuring that it meets all necessary fabrication requirements. Each of these steps is critical for the successful implementation and optimization of the design.

Functional Modules

Computation Module: This module handles the processing of DFT information and scan cells. Users can read DFT data from CAD files, process scan cells, and read layout files for further analysis. The computation module supports various formats and standards, making it compatible with a wide range of design environments. It includes functionalities such as:

- **Read DFT from CAD:** This function imports DFT information from CAD files, allowing for seamless integration with existing design data.
- **Read Layout File:** Users can import layout files for detailed analysis and optimization.
- **Process Scan Cells:** This function processes the scan cells, preparing them for integration into the design.

Floorplanning Module: The tool provides advanced floorplanning capabilities, allowing users to visualize and manage the placement of scan pins and scan chains. The floorplanning module includes options for:

- **Floorplan Graph:** Displays the overall floorplan of the design, providing a visual representation of the cell placements and interconnections.

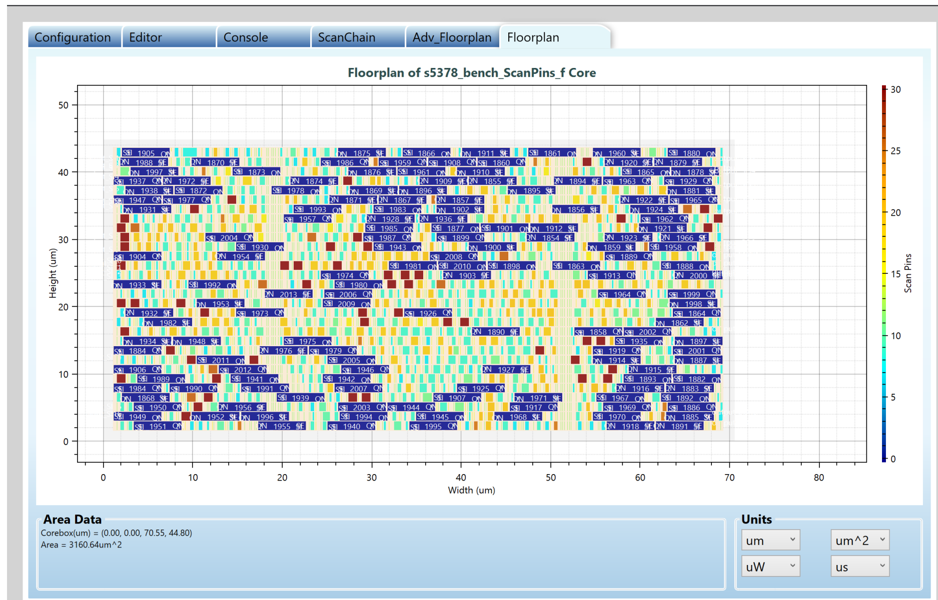

Figure 25: Floorplan

- **Advanced Floorplan Graph:** Offers more detailed visualizations and analysis tools for advanced floorplanning tasks.

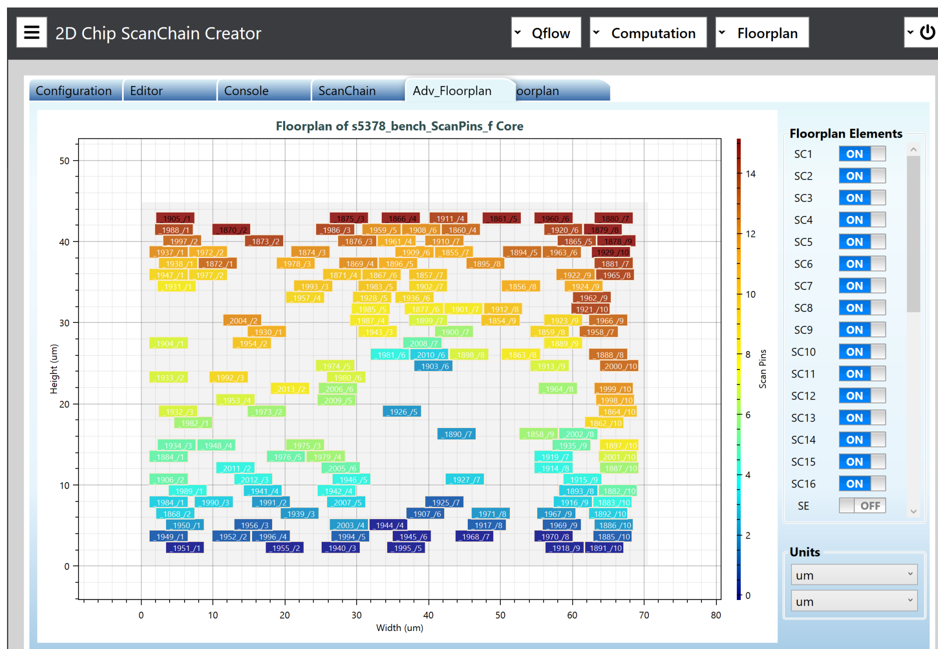

Figure 26: Advanced Floorplan

- **Point-Based Scan Pins Graph:** Focuses on the placement and routing of scan pins, ensuring optimal placement for testability.

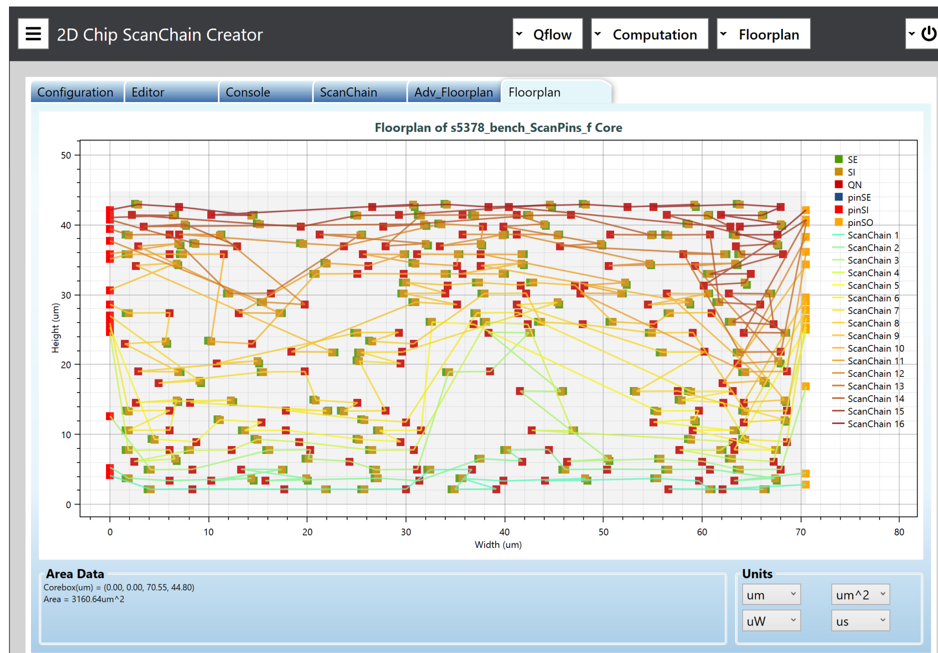

Figure 27: Point-Based Scan Pins Graph

- **Display ScanChains:** Visualizes the scan chains within the design, providing insights into their structure and optimization.

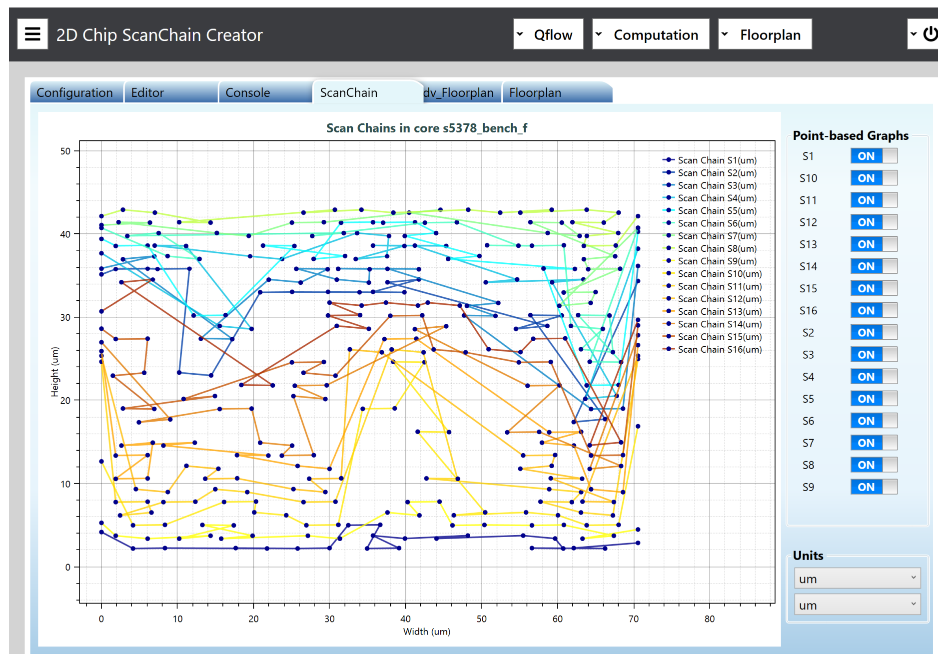

Figure 28: Display ScanChains

- **Display Core:** Displays the core layout of the design, highlighting critical areas and interconnections.

The 2D Chip ScanChain Creator features an intuitive user interface designed to facilitate ease of use and accessibility. Key aspects of the user interface include:

Modular Design: The tool's modular design allows users to easily navigate through different functionalities, configure settings, and visualize results. This modularity ensures that users can focus on specific tasks without being overwhelmed by unnecessary complexity. The interface consists of several tabs including Configuration, Editor, Console, ScanChain, Adv_Floorplan and Floorplan.

Customization Options: Users can customize the tool's settings to match their specific project requirements, providing a flexible and adaptable environment for various design scenarios.

Visual Feedback: The tool provides visual feedback through graphs, helping users to understand the impact of their design choices and optimizations. This visual feedback is crucial for identifying potential issues and making informed decisions.

3.3.2 Case Study Applications

To illustrate the effectiveness and functionality of the 2D Chip ScanChain Creator, several case study applications were conducted using benchmark circuits and advanced test algorithms.

Benchmark Circuits: The tool was applied to a variety of benchmark circuits, such as s5378, s13207, s38417, s38584, aes_cipher_top, md5, and ac97_top. These circuits vary in complexity and size, providing a comprehensive evaluation of the tool's capabilities.

Algorithm Implementation: The tool supports the implementation of several TSP algorithms for scan chain optimization, including GlobalCheapestArc, SequentialCheapestInsertion, Christofides, and FirstUnboundMinValue. Each algorithm was tested to assess its impact on scan chain length and overall performance.

Qflow Integration: Through its integration with Qflow, the tool was used to perform synthesis, placement, and routing on the benchmark circuits. The results demonstrated significant improvements in design efficiency and test coverage.

Performance Metrics: Key performance metrics, such as total scan path length were recorded and analyzed. The tool's ability to optimize these metrics highlights its practical applicability in real-world VLSI design and testing scenarios.

User Interface and Experience: The intuitive user interface of the 2D Chip ScanChain Creator was evaluated for usability. The tool's modular design allows users to easily navigate through different functionalities, configure settings, and visualize results, making it accessible to both novice and experienced VLSI designers.

These case studies underscore the tool's potential to enhance the design and testing processes in scan chains, providing valuable insights and optimizations that contribute to the overall efficiency and effectiveness of digital circuit design and testing.

3.4 Benchmark Circuits and Test Algorithms

3.4.1 Selection Criteria for Benchmark Circuits

The selection of benchmark circuits is critical for evaluating the performance of test algorithms and ensuring their relevance to practical applications. The following criteria were used to select the benchmark circuits:

Complexity and Size: Benchmark circuits vary in complexity and size to provide a comprehensive evaluation of the algorithms. For instance, circuits like s5378 are relatively small, while circuits like s38417 and s38584 are larger and more complex. This diversity helps in testing the scalability and robustness of the algorithms [51].

Relevance to Industry: The chosen circuits, such as aes_cipher_top and md5, are representative of common applications in cryptography, which is essential in modern electronics. Similarly, ac97_top is used in audio codec applications, demonstrating the algorithms' applicability across different domains [52].

Availability of Data: These benchmark circuits are well-documented and widely used in both academic and industry research. This ensures the availability of reference data for comparison, facilitating the validation of the proposed methods.

3.4.2 Description and Justification of Algorithms Used

Four primary algorithms are utilized in this research: Christofides, FirstUnboundMinValue, GlobalCheapestArc, and SequentialCheapestInsertion. Each algorithm has its unique approach and benefits:

Christofides Algorithm: This algorithm is known for providing a near-optimal solution to the Traveling Salesman Problem (TSP) within a factor of 3/2 of the optimal solution. It constructs a minimum spanning tree, finds a minimum-weight perfect matching, and combines them to form an Eulerian circuit, which is then converted into a Hamiltonian circuit by shortcutting repeated vertices. This method is particularly effective for ensuring high-quality solutions in reasonable computational time [43].

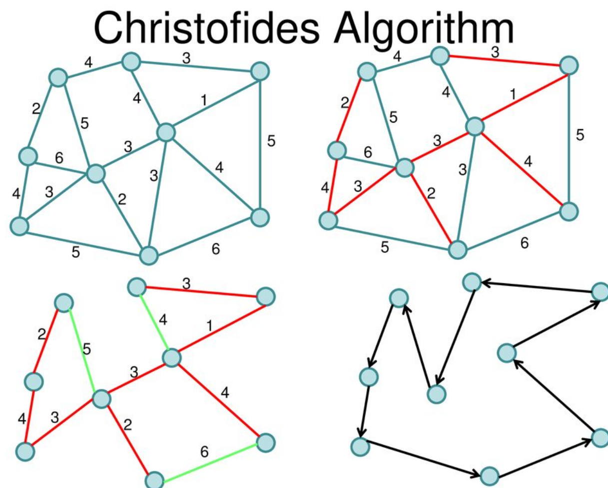

Figure 29: Christofides Algorithm[18]

- **Minimum Spanning Tree (MST)**: Construct an MST from the graph, ensuring all vertices are connected with the minimum total edge weight.
- **Perfect Matching**: Identify vertices with an odd degree in the MST and find a minimum-weight perfect matching for these vertices.
- **Eulerian Circuit and Hamiltonian Path**: Combine the edges of the MST and the perfect matching to form an Eulerian circuit. Convert this circuit into a Hamiltonian path by skipping repeated vertices to form the final tour.

FirstUnboundMinValue (FUMV): This heuristic selects the variable with the smallest domain and assigns it the minimum value. It is a part of constraint programming and is used for solving combinatorial optimization problems efficiently by minimizing the search space at each decision point [44].

- **Variable Selection**: Choosing the variable with the smallest domain, which means the variable that has the fewest possible values left to be assigned.
- **Value Assignment**: Assigning the minimum value from the domain to the chosen variable.

GlobalCheapestArc: This algorithm iteratively adds the cheapest arc (edge) to the solution, ensuring that no node is added more than once and that the resulting path forms a valid tour. It is suitable for large datasets where quick heuristic solutions are preferred over exact solutions [53].

- **Arc Selection**: Iteratively select the cheapest arc from the remaining unvisited nodes to the current node.
- **Tour Formation**: Ensure that the selected arcs form a valid tour without revisiting any node.

SequentialCheapestInsertion (SCI): This method builds the tour by repeatedly inserting the next node at the cheapest position in the existing partial tour. It is effective for constructing high-quality solutions incrementally and is widely used in various routing and scheduling applications [54].

- **Initial Tour**: Start with a small initial tour, usually consisting of two or three nodes.
- **Node Insertion**: Sequentially insert the remaining nodes at the position that results in the smallest increase in tour length.
- **Tour Update**: Update the tour after each insertion to ensure it remains valid.

Algorithm	Complexity
Christofides	$O(V^3)$
FirstUnboundMinValue	$O(d^n)$
Sequential Cheapest Insertion	$O(n^3)$
Global Cheapest Arc	$O(n^2 \log n)$
LeoSolveTSP	$O(n^2)$

Table 2: Algorithms Complexity

These algorithms, each with its strengths, provide a robust framework for solving the TSP in the context of scan chain optimization in VLSI design. The Christofides algorithm ensures near-optimal solutions, FUMV minimizes the search space, GlobalCheapestArc offers efficiency for large datasets, and SCI provides a method for incrementally building high-quality solutions. This diverse set of algorithms allows for a comprehensive evaluation of scan chain optimization across different types of benchmark circuits.

3.4.3 Testing and Validation

The testing and validation of the proposed algorithms involve several steps:

Algorithm Performance: The algorithms are evaluated based on their ability to optimize scan chain lengths and reduce testing time. Performance metrics include computational efficiency, solution quality, and scalability [51].

Benchmark Circuit Analysis: Each benchmark circuit is subjected to the proposed algorithms to assess their effectiveness in real-world scenarios. Metrics such as total scan path length are recorded and analyzed [52].

Simulation and Real-World Testing: The algorithms are implemented and tested within a simulation environment using software tools like Qflow, with the Nangate 45nm library selected as the cell library. This library is chosen to provide accurate cell models that closely represent the physical characteristics of the circuit components, ensuring that the simulation results are realistic and reliable. By integrating the Nangate 45nm library into Qflow, the simulation can effectively mimic real-world conditions, providing a strong basis for performance evaluation.

Documentation and Reporting: Detailed documentation of the testing process, results, and analysis is maintained. This includes recording any deviations from expected outcomes and providing explanations for such occurrences. Comprehensive reports are generated to summarize the findings and provide recommendations for future improvements.

4 Experiments and Results

4.1 Experimental Setup

The experimental setup for this research involved a detailed integration of software tools and hardware configurations tailored to the needs of VLSI design, testing, and optimization.

4.1.1 Software and Tools Used

The primary Integrated Development Environment (IDE) used for developing the Leo-SolveTSP function and the 2D Chip ScanChain Creator was Visual Studio 2022. This robust development platform offers a comprehensive suite of features essential for both novice and expert developers. Visual Studio 2022 provides advanced code editing capabilities with intelligent code completion, real-time error checking, and powerful refactoring tools that significantly enhance coding efficiency and accuracy. The IDE's integrated debugging tools, such as breakpoints, watch windows, and step-through debugging, allow developers to thoroughly examine code behavior and identify issues at an early stage. Performance profiling tools are available to analyze code performance, memory usage, and to identify bottlenecks, ensuring that the developed algorithms are optimized for speed and efficiency. Visual Studio 2022 also integrates seamlessly with version control systems like Git, enabling efficient code management, collaboration, and version tracking. Its extensibility supports a wide range of extensions and plugins, allowing customization of the development environment to suit specific project needs. Moreover, cloud integration with Azure facilitates cloud-based development, testing, and deployment, making it an ideal choice for developing complex VLSI design tools and applications [55].

Figure 30: Visual Studio 2022[19]

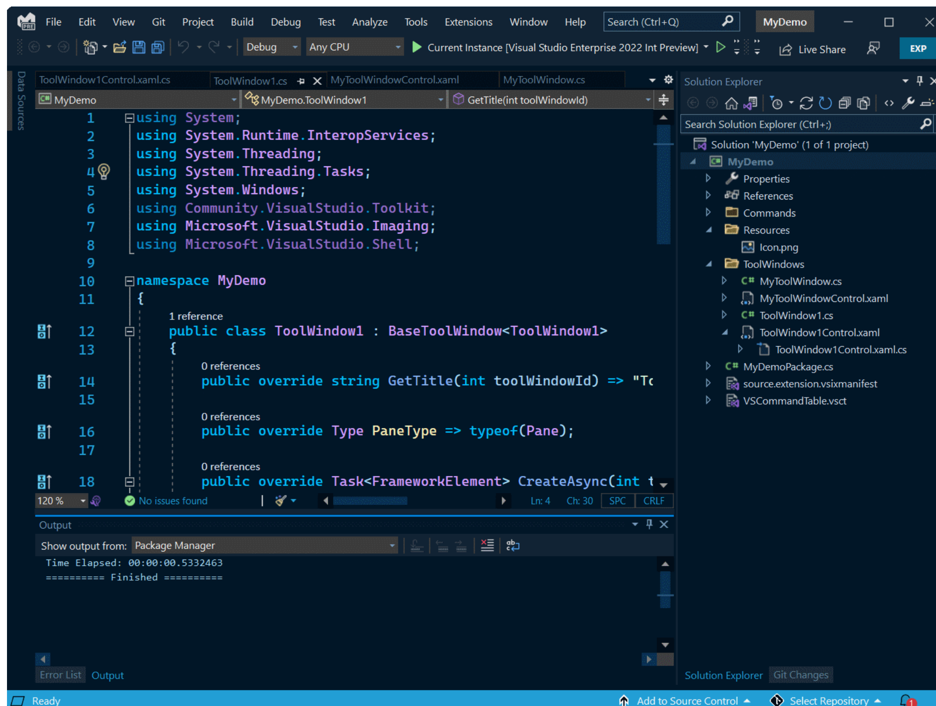

Figure 31: Visual Studio 2022 IDE[20]

For the results analysis program, PyCharm Community Edition 2022.3.3 was used. PyCharm is a widely recognized IDE specifically designed for Python development, providing advanced code editing, debugging, and testing capabilities. It offers powerful navigation tools, such as class, file, and symbol search, which make it easy to locate and modify code efficiently. The integrated debugger includes features like breakpoints, step-through execution, and variable inspection, crucial for identifying and resolving issues in Python code. PyCharm supports various testing frameworks such as pytest, unittest, and doctest, facilitating the creation and execution of unit tests to ensure code reliability. The IDE also manages virtual environments seamlessly, allowing the isolation of project dependencies and ensuring consistency across different development environments. Intelligent code assistance, with features like code completion, error detection, and quick-fix suggestions, significantly enhances productivity and reduces coding errors. PyCharm's integration with version control systems like Git provides tools for commit, push, pull, and branch management directly within the IDE. Additionally, PyCharm includes tools for managing and interacting with databases, supporting a range of database systems, and providing features like SQL editing and database browsing. Its rich feature set and focus on Python development make it a valuable tool for writing and maintaining the results analysis scripts used in this research [56].

Figure 32: PyCharm

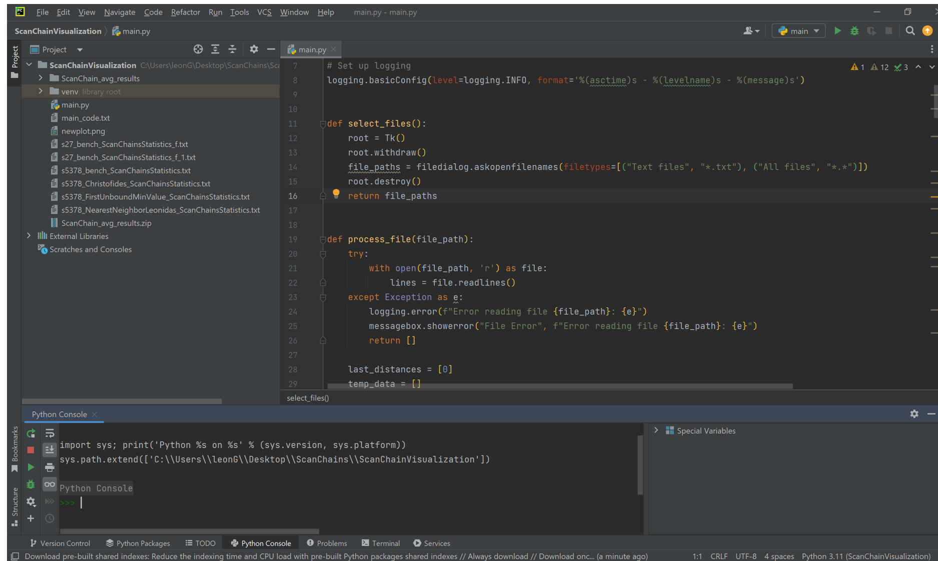

Figure 33: PyCharm IDE

The Python program developed for results analysis utilized several key libraries:

Plotly: Used for creating interactive plots and visualizations. Plotly provides a wide range of plotting functions and supports complex visualizations, making it ideal for displaying experimental results in a clear and interactive manner [57].

Tkinter: Employed for creating graphical user interfaces (GUIs). Tkinter is a standard Python library for creating GUIs, providing tools for building windows, dialogs, and various input controls [58].

OS: Used for interacting with the operating system, such as file handling and directory management.

Logging: Utilized for generating log messages to track the execution of the program and record any errors or important events.

The experiments also utilized Qflow and the NanGate 45nm Open Cell Library, which were previously discussed in detail. Qflow integrates several tools, including Yosys for synthesis, Graywolf for placement, and Qrouter for routing, providing a comprehensive workflow for digital design. The NanGate 45nm library includes a wide range of standard cells characterized for a 45nm technology node, essential for accurate design analysis and optimization. The integration of these tools with Visual Studio

2022 ensures a robust and efficient design and testing environment, facilitating seamless workflow and accurate results.

4.1.2 Hardware Configuration

The hardware configuration for this research was designed to support the execution of complex algorithms and handle extensive computational tasks associated with VLSI design and testing.

The experiments were conducted on a system powered by an Intel Core i5-1240P processor. This CPU features a base clock speed of 3.3 GHz and belongs to the Alder Lake (12th Gen) family. The Intel Core i5-1240P is equipped with advanced features such as hyper-threading and efficient power consumption, making it suitable for high-performance computing tasks in both development and testing phases [59].

The system uses Intel Iris Xe Graphics, which provides integrated graphics performance sufficient for handling graphical user interfaces and basic computational tasks. While it does not offer the same level of performance as discrete GPUs, the Iris Xe Graphics is optimized for energy efficiency and is capable of supporting high-resolution displays and moderate graphical workloads [60].

The system is equipped with 16 GB of DDR5 RAM. DDR5 memory offers higher bandwidth and efficiency. The ample 16 GB capacity ensures that the system can handle large datasets and run multiple applications simultaneously without performance degradation. This memory configuration is particularly beneficial for applications requiring intensive data processing and real-time analysis.

The hardware configuration includes a 512 GB SSD (Solid State Drive). SSDs provide fast read and write operations, which are crucial for quickly loading large design files and datasets. The high-speed storage ensures efficient data logging and analysis during the experiments, significantly reducing latency compared to traditional hard drives. The SSD's performance is vital for maintaining workflow efficiency and handling large volumes of data generated during VLSI design processes.

4.2 Data Collection and Analysis

The data collection and analysis process for this research involved systematic methods to ensure the accuracy and reliability of the results. This chapter outlines the approaches used for data collection.

Data collection in this research was meticulously planned and executed to gather relevant information for evaluating the performance of the developed algorithms and tools. The primary methods of data collection included:

Simulation Runs: Extensive simulation runs were conducted using the developed algorithms within the Visual Studio 2022. These simulations generated data on various performance metrics such as execution time, memory usage, and accuracy of the solutions.

Benchmark Circuits: A diverse set of benchmark circuits (s5378, s13207, s38417, s38584, aes_cipher_top, md5, and ac97_top) were utilized to evaluate the algorithms. These benchmarks provided a range of complexity and size, allowing for a comprehensive assessment of the algorithms' scalability and robustness. Data from these circuits were collected during the simulation runs.

Benchmark Circuit	Number of DFFs
s5378	164
s13207	229
s38417	1471
s38584	1197
aes_cipher_top	558
md5	916
ac97_top	2279

Table 3: Number of DFFs in Different Benchmark Circuits

Tool Workflow in 2D Chip ScanChain Creator: The data collection process for the benchmark circuits in the 2D Chip ScanChain Creator involves the following steps:

- **Press Qflow, then Load DUT:** The first step involves loading the Design Under Test (DUT) into the Qflow environment. This prepares the design for subsequent synthesis and placement operations.
- **Run Synthesis and Run Placement:** These steps involve synthesizing the design into a gate-level netlist and then placing the gates on the chip layout. The synthesis transforms the high-level design into a detailed, implementable format, while placement determines the optimal physical locations for the gates.
- **Insert DFT and DFT Prior Placement:** These steps incorporate Design-for-Testability (DFT) measures before the placement phase to ensure that test structures are properly integrated into the design. This is crucial for enhancing the testability of the circuit.
- **Zone-Based and Algorithm Selection:** The design is further optimized through zone-based partitioning, where the circuit is divided into manageable regions for

targeted optimization. An algorithm is then applied to sequence the scan chains effectively.

- **TSP Implementation:** The Traveling Salesman Problem (TSP) implementation is used to optimize the order of scan cells within the scan chain, minimizing the overall path length and improving testing efficiency.
- **ScanChain Number Selection:** This step involves defining the number of scan chains based on the design requirements and optimization results.
- **Go to Computation, press Process Scan Cells, and then File_bench_ScanPins:** The design is processed to identify and configure the scan cells. The **File_bench_ScanPins** command generates detailed files of the scan chains and their configurations.
- **Go to Floorplan, Display ScanChains, and select File_bench_ScanPins_f:** This final step involves visualizing the scan chains using the Floorplan tool. This visualization aids in assessing the layout and efficiency of the scan chain configuration.

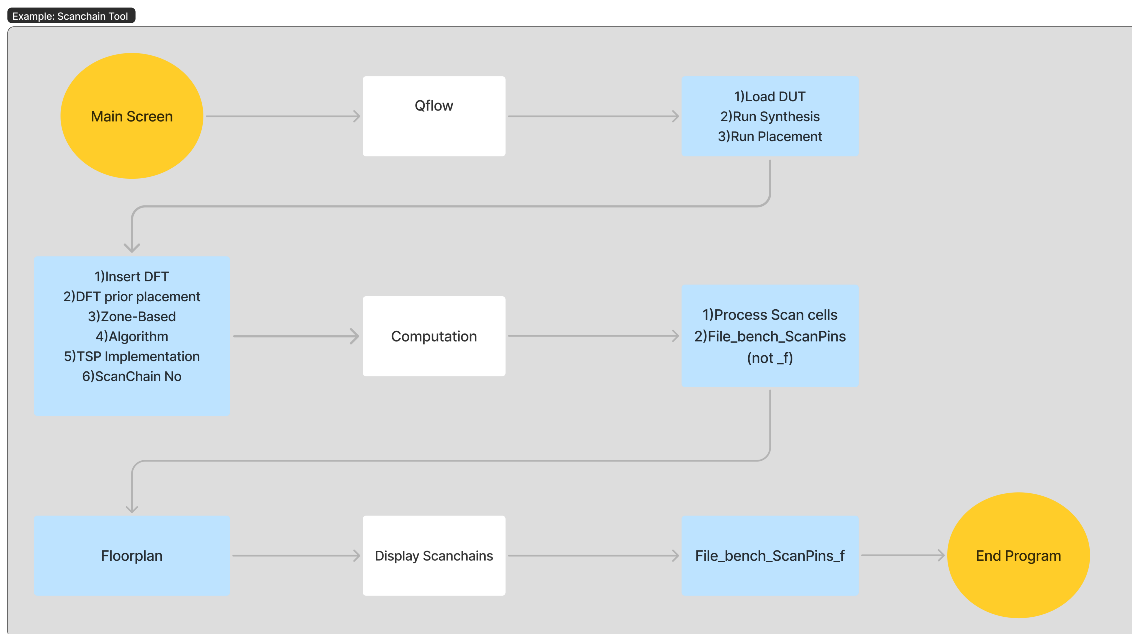

Figure 34: Tool Workflow in 2D Chip ScanChain Creator

The data that are collected are produced after the execution of these steps in the benchmark circuits (s5378, s13207, s38417, s38584, aes_cipher_top, md5, and ac97_top).

4.3 Zone-Based and Non-Zone-Based Scan Chains

Two primary approaches in scan chain design are zone-based and non-zone-based scan chains, each with its unique configuration and impact on testing efficiency.

Zone-Based Scan Chains

Zone-based scan chains involve dividing the circuit into localized regions, known as zones, where each zone contains a portion of the scan cells. The scan chains are then configured within these zones, which helps to keep the routing paths short and localized. This approach focuses on maintaining proximity between scan cells, thus reducing the need for long interconnections across different parts of the circuit.

Zone-based configurations are particularly advantageous in scenarios where managing timing and reducing power consumption are critical. By localizing the routing paths, zone-based scan chains minimize switching activities, which can significantly lower the power required during testing. This method also improves the overall efficiency of the testing process by enabling faster data shifting between scan cells due to their closer proximity.

Non-Zone-Based Scan Chains

Non-zone-based scan chains, on the other hand, do not utilize a structured partitioning of the circuit. In this approach, scan cells are linked without specific consideration of their physical locations, resulting in scan paths that can span across the entire chip. This can lead to longer routing paths, increased congestion, and a more complex layout management.

The absence of a localized structure in non-zone-based configurations often results in higher power consumption due to extended routing and increased switching activity during testing. Despite these challenges, non-zone-based scan chains are simpler to design initially and do not require the detailed partitioning required for zone-based configurations. However, this simplicity can come at the cost of efficiency, especially in larger and more complex circuit designs where routing and timing constraints are more stringent.

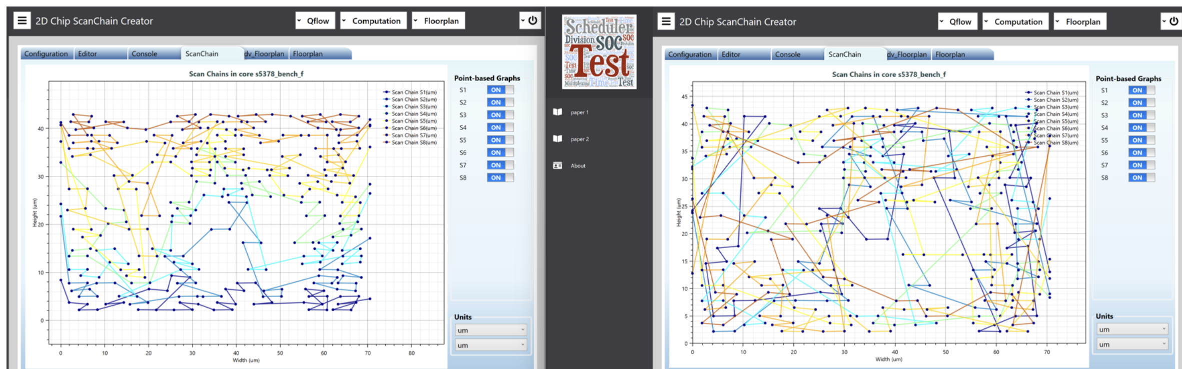

Figure 35: Comparison of Zone-Based and Non-Zone-Based Scan Chain Configurations

Application and Impact

Both zone-based and non-zone-based scan chains play essential roles in the testing and validation of VLSI circuits. Zone-based approaches are particularly suited for designs where power efficiency and routing optimization are critical, offering a structured method to enhance the performance of testing operations. In contrast, non-zone-based scan chains provide a straightforward approach that may be easier to implement but can lead to inefficiencies in terms of power and routing management in complex circuits.

Understanding these approaches allows designers to select the most appropriate scan chain configuration based on the specific requirements of the circuit, balancing simplicity, performance, and testing efficiency.

4.4 Visualization of Results

The presentation of results is a critical component of this research, offering insights into the performance and effectiveness of the developed algorithms and tools. This section encompasses a detailed exposition of the results, complemented by appropriate visualizations to elucidate key findings. The visualization of results was executed using a program developed in PyCharm Community Edition 2022.3.3, a highly regarded Integrated Development Environment (IDE) specifically designed for Python development.

Visualization Workflow and Results Analysis

The results analysis program leverages several Python libraries to process the data and generate comprehensive visualizations. Below is an overview of the workflow, the tools employed, and the insights derived from the visualizations.

Workflow Overview

- **File Selection**: The program initiates by prompting users to select multiple data files through a graphical file dialog box, enabled by the Tkinter library. This user-friendly interface simplifies the file selection process.
- **Data Processing**: Each selected file undergoes processing to extract relevant data points. The program reads the files line by line, parses the data, and stores it in a structured format. The OS library manages file operations such as reading, writing, and directory management, ensuring smooth and efficient program execution.
- **Visualization Creation**: The processed data is utilized to create visualizations. The program allows users to choose between line charts and bar charts, with the visualizations being rendered using the Plotly library. Plotly's extensive capabilities facilitate the creation of interactive and aesthetically pleasing visual representations.
- **Annotations and Comparisons**: The visualizations are annotated with detailed information, including average and total distances, and percentage differences from the best-performing algorithm. This enhances the interpretability of the visualizations, making it easier to compare results across different datasets.
- **Output and Logging**: The final visualizations are displayed to the user, with the results logged for future reference. The program also saves the results to a

file for further analysis. The Logging library plays a crucial role in tracking the program's execution, recording errors, and significant events.

Interactive Plots and Insights

Interactive plots form the cornerstone of the visualization process, allowing users to dynamically explore the data. These plots enable functionalities such as zooming, panning, and hovering over data points for detailed information, which are essential for in-depth data analysis and trend identification.

The visualizations generated include:

- **Line Charts**: These charts plot the distances for each scan chain, facilitating an easy comparison of algorithm performance. Different algorithms are represented by distinct colored lines, with the X-axis denoting the ScanChain index and the Y-axis representing the ScanChain distance. This visualization aids in identifying performance trends and variations across algorithms.

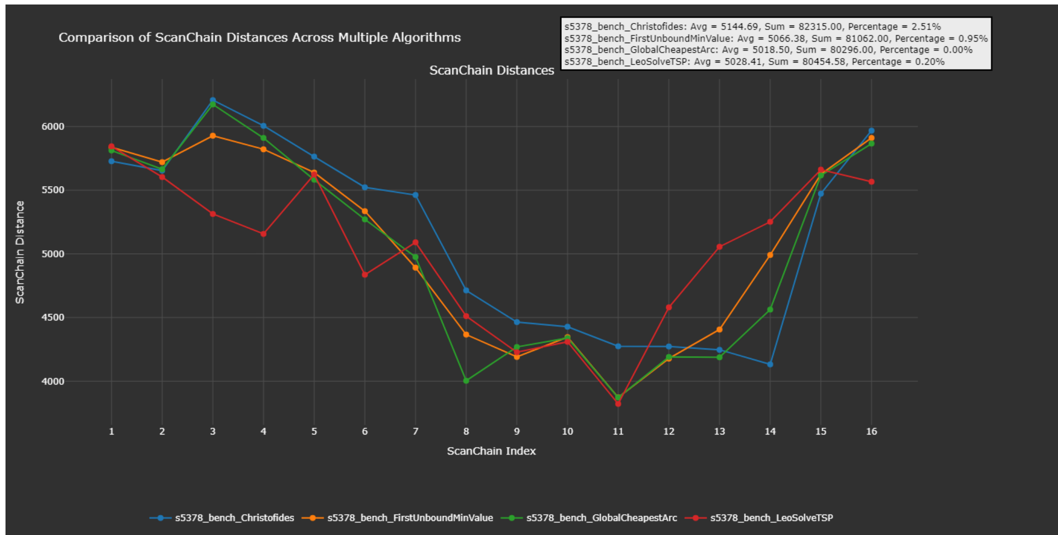

Figure 36: Line Chart

- **Bar Charts**: These charts provide a summary of the total distances for each algorithm, enabling a straightforward comparison of overall effectiveness. Each bar represents a different algorithm, with the bar height indicating the total ScanChain distance. This visualization succinctly conveys the cumulative performance of each algorithm.

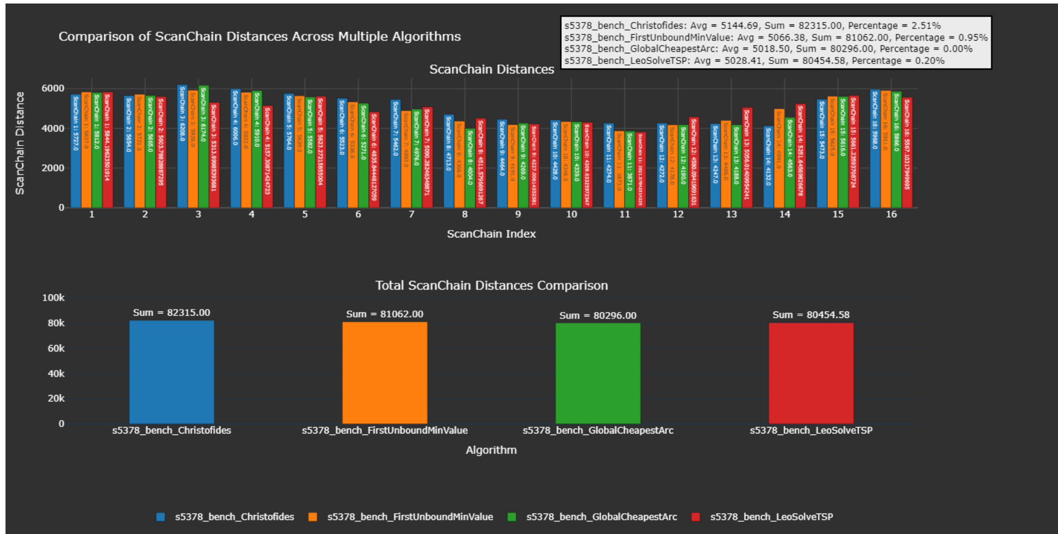

Figure 37: Bar Chart

Detailed Analysis of Results

The insights derived from the visualizations include:

- **Algorithm Performance**: The visualizations clearly demonstrate how different algorithms perform in minimizing ScanChain distances. Line charts, for example, reveal consistent performance trends and highlight algorithms that outperform others across various ScanChain configurations.
- **Statistical Analysis**: Metrics such as average and total distances, displayed in the annotations, offer quantitative measures of algorithm performance. These metrics are crucial for identifying the most efficient algorithms under different conditions.
- **Comparison Across Datasets**: The ability to compare results from multiple datasets underscores the robustness of the algorithms. Consistent performance across datasets indicates algorithm reliability, while significant variations may suggest the need for further optimization.

The application of PyCharm and the developed Python program significantly enhanced the research's capacity to visualize and analyze experimental results. The combination of interactive plots, user-friendly GUIs, efficient file handling, and robust logging provided a holistic solution for data analysis. The insights gained from these visualizations were instrumental in assessing the effectiveness of the developed algorithms and tools, thereby contributing substantially to the research's success.
The presentation of results is a critical part of this research, as it provides insights into the performance and effectiveness of the developed algorithms and tools. This section includes detailed results along with appropriate visualizations to illustrate key findings.

4.5 Detailed Presentation of Results

The results obtained from the simulation runs and benchmark circuit evaluations are presented in this section. The primary performance metrics considered include execution time, memory usage, and the quality of the solutions generated by the algorithms.

Scan Chain Number	Christofides	FirstUnboundMinValue	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 48358.00 AvgSCL = 12089.50 (+0.25%)	TL = 48237.00 AvgSCL = 12059.25 (0.00%)	TL = 48326.00 AvgSCL = 12081.50 (+0.18%)	TL = 63081.88 AvgSCL = 15770.47 (+30.77%)
8 Scan-Chains	TL = 70565.00 AvgSCL = 8820.62 (0.00%)	TL = 71752.00 AvgSCL = 8969.00 (+1.68%)	TL = 71269.00 AvgSCL = 8908.62 (+1.00%)	TL = 87875.91 AvgSCL = 10984.49 (+24.53%)
16 Scan-Chains	TL = 105551.00 AvgSCL = 6596.94 (+1.10%)	TL = 104402.00 AvgSCL = 6525.12 (0.00%)	TL = 104924.00 AvgSCL = 6557.75 (+0.50%)	TL = 121555.43 AvgSCL = 7597.21 (+16.43%)
32 Scan-Chains	TL = 154579.00 AvgSCL = 4830.59 (0.00%)	TL = 156956.00 AvgSCL = 4904.88 (+1.54%)	TL = 155031.00 AvgSCL = 4844.72 (+0.29%)	TL = 166400.73 AvgSCL = 5200.02 (+7.65%)

Table 4: Benchmark Circuit: s5378, Non-Zone-Based

TL = Total Length
AvgSCL = Average Scan Chain Length

Scan Chain Number	Christofides	FirstUnboundMinValue	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 33784.00 AvgSCL = 8446.00 (0.00%)	TL = 34083.00 AvgSCL = 8520.75 (+0.89%)	TL = 34657.00 AvgSCL = 8664.25 (+2.58%)	TL = 42806.95 AvgSCL = 10701.74 (+26.71%)
8 Scan-Chains	TL = 49831.00 AvgSCL = 6228.88 (+2.08%)	TL = 50491.00 AvgSCL = 6311.38 (+3.43%)	TL = 48817.00 AvgSCL = 6102.12 (0.00%)	TL = 60367.21 AvgSCL = 7545.90 (+23.66%)
16 Scan-Chains	TL = 82315.00 AvgSCL = 5144.69 (+2.51%)	TL = 81062.00 AvgSCL = 5066.38 (+0.95%)	TL = 80296.00 AvgSCL = 5018.50 (0.00%)	TL = 80454.58 AvgSCL = 5028.41 (+0.20%)

Table 5: Benchmark Circuit: s5378, Zone-Based

Scan Chain Number	Christofides	FirstUnboundMinValue	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 56982.00 AvgSCL = 14245.50 (0.00%)	TL = 57530.00 AvgSCL = 14382.50 (+0.96%)	TL = 57788.00 AvgSCL = 14447.00 (+1.41%)	TL = 67682.77 AvgSCL = 16920.69 (+18.78%)
8 Scan-Chains	TL = 85461.00 AvgSCL = 10682.62 (+1.28%)	TL = 86095.00 AvgSCL = 10761.88 (+2.03%)	TL = 84380.00 AvgSCL = 10547.50 (0.00%)	TL = 110538.06 AvgSCL = 13817.26 (+31.00%)
16 Scan-Chains	TL = 131314.00 AvgSCL = 8207.12 (+2.75%)	TL = 129485.00 AvgSCL = 8092.81 (+1.32%)	TL = 127797.00 AvgSCL = 7987.31 (0.00%)	TL = 153458.92 AvgSCL = 9591.18 (+20.08%)
32 Scan-Chains	TL = 185613.00 AvgSCL = 5800.41 (+2.88%)	TL = 180423.00 AvgSCL = 5638.22 (0.00%)	TL = 187101.00 AvgSCL = 5846.91 (+3.70%)	TL = 206240.65 AvgSCL = 6445.02 (+14.31%)

Table 6: Benchmark Circuit: s13207, Non-Zone-Based

Scan Chain Number	Christofides	FirstUnboundMinValue	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 38643.00 AvgSCL = 9660.75 (+4.34%)	TL = 37036.00 AvgSCL = 9259.00 (0.00%)	TL = 39626.00 AvgSCL = 9906.50 (+6.99%)	TL = 48657.12 AvgSCL = 12164.28 (+31.38%)
8 Scan-Chains	TL = 54601.00 AvgSCL = 6825.12 (+0.78%)	TL = 56624.00 AvgSCL = 7078.00 (+4.51%)	TL = 54180.00 AvgSCL = 6772.50 (0.00%)	TL = 64711.94 AvgSCL = 8088.99 (+19.44%)
16 Scan-Chains	TL = 80795.00 AvgSCL = 5049.69 (0.00%)	TL = 81642.00 AvgSCL = 5102.62 (+1.05%)	TL = 81576.00 AvgSCL = 5098.50 (+0.97%)	TL = 81295.36 AvgSCL = 5080.96 (+0.62%)

Table 7: Benchmark Circuit: s13207, Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
8 Scan-Chains	TL = 527978.00 AvgSCL = 65997.25 (0.00%)	TL = 536264.00 AvgSCL = 67033.00 (+1.57%)	TL = 530561.00 AvgSCL = 66320.12 (+0.49%)	TL = 646908.83 AvgSCL = 80863.60 (+22.53%)
16 Scan-Chains	TL = 771595.00 AvgSCL = 48224.69 (0.00%)	TL = 789129.00 AvgSCL = 49320.56 (+2.27%)	TL = 780994.00 AvgSCL = 48812.12 (+1.22%)	TL = 952691.85 AvgSCL = 59543.24 (+23.47%)
32 Scan-Chains	TL = 1137915.00 AvgSCL = 35559.84 (+0.45%)	TL = 1154833.00 AvgSCL = 36088.53 (+1.95%)	TL = 1132777.00 AvgSCL = 35399.28 (0.00%)	TL = 1380913.46 AvgSCL = 43153.55 (+21.91%)

Table 8: Benchmark Circuit: s38417, Non-Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 215962.00 AvgSCL = 53990.50 (+1.72%)	TL = 221306.00 AvgSCL = 55326.50 (+4.23%)	TL = 212316.00 AvgSCL = 53079.00 (0.00%)	TL = 266451.09 AvgSCL = 66612.77 (+25.50%)
8 Scan-Chains	TL = 264081.00 AvgSCL = 33010.12 (+1.37%)	TL = 261916.00 AvgSCL = 32739.50 (+0.54%)	TL = 260521.00 AvgSCL = 32565.12 (0.00%)	TL = 344882.01 AvgSCL = 43110.25 (+32.38%)
16 Scan-Chains	TL = 373714.00 AvgSCL = 23357.12 (0.00%)	TL = 380394.00 AvgSCL = 23774.62 (+1.79%)	TL = 401622.00 AvgSCL = 25101.38 (+7.47%)	TL = 509039.24 AvgSCL = 31814.95 (+36.21%)

Table 9: Benchmark Circuit: s38417, Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 311809.00 AvgSCL = 77952.25 (0.00%)	TL = 315091.00 AvgSCL = 78772.75 (+1.05%)	TL = 315293.00 AvgSCL = 78823.25 (+1.12%)	TL = 386583.74 AvgSCL = 96645.93 (+23.98%)
8 Scan-Chains	TL = 450153.00 AvgSCL = 56269.12 (0.00%)	TL = 466253.00 AvgSCL = 58281.62 (+3.58%)	TL = 457116.00 AvgSCL = 57139.50 (+1.55%)	TL = 560194.54 AvgSCL = 70024.32 (+24.45%)
16 Scan-Chains	TL = 653053.00 AvgSCL = 40815.81 (0.00%)	TL = 668118.00 AvgSCL = 41757.38 (+2.31%)	TL = 669070.00 AvgSCL = 41816.88 (+2.45%)	TL = 802117.37 AvgSCL = 50132.34 (+22.83%)
32 Scan-Chains	TL = 938931.00 AvgSCL = 29341.59 (0.00%)	TL = 959157.00 AvgSCL = 29973.66 (+2.15%)	TL = 949028.00 AvgSCL = 29657.12 (+1.08%)	TL = 1116398.35 AvgSCL = 34887.45 (+18.90%)

Table 10: Benchmark Circuit: s38584, Non-Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 187646.00 AvgSCL = 46911.50 (0.00%)	TL = 191290.00 AvgSCL = 47822.50 (+1.94%)	TL = 188335.00 AvgSCL = 47083.75 (+0.37%)	TL = 233645.46 AvgSCL = 58411.37 (+24.51%)
8 Scan-Chains	TL = 239228.00 AvgSCL = 29903.50 (+0.12%)	TL = 242106.00 AvgSCL = 30263.25 (+1.33%)	TL = 238939.00 AvgSCL = 29867.38 (0.00%)	TL = 299696.13 AvgSCL = 37462.02 (+25.43%)
16 Scan-Chains	TL = 337390.00 AvgSCL = 21086.88 (+0.36%)	TL = 349268.00 AvgSCL = 21829.25 (+3.89%)	TL = 336196.00 AvgSCL = 21012.25 (0.00%)	TL = 441659.39 AvgSCL = 27603.71 (+31.37%)

Table 11: Benchmark Circuit: s38584, Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
16 Scan-Chains	TL = 1169343.00 AvgSCL = 73083.94 (+0.40%)	TL = 1196635.00 AvgSCL = 74789.69 (+2.75%)	TL = 1164627.00 AvgSCL = 72789.19 (0.00%)	TL = 1438232.20 AvgSCL = 89889.51 (+23.49%)
32 Scan-Chains	TL = 1692631.00 AvgSCL = 52894.72 (0.00%)	TL = 1739034.00 AvgSCL = 54344.81 (+2.74%)	TL = 1717412.00 AvgSCL = 53669.12 (+1.46%)	TL = 2068231.20 AvgSCL = 64632.23 (+22.19%)

Table 12: Benchmark Circuit: ac97_top, Non-Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
8 Scan-Chains	TL = 410881.00 AvgSCL = 51360.12 (0.00%)	TL = 420289.00 AvgSCL = 52536.12 (+2.29%)	TL = 410881.00 AvgSCL = 51360.12 (0.00%)	TL = 512988.58 AvgSCL = 64123.57 (+24.85%)
16 Scan-Chains	TL = 540460.00 AvgSCL = 33778.75 (0.00%)	TL = 571944.00 AvgSCL = 35746.50 (+5.83%)	TL = 566908.00 AvgSCL = 35431.75 (+4.89%)	TL = 707618.62 AvgSCL = 44226.16 (+30.93%)
32 Scan-Chains	TL = 935636.00 AvgSCL = 29238.62 (+2.41%)	TL = 940075.00 AvgSCL = 29377.34 (+2.90%)	TL = 913590.00 AvgSCL = 28549.69 (0.00%)	TL = 1157177.75 AvgSCL = 36161.80 (+26.66%)

Table 13: Benchmark Circuit: ac97_top, Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 195237.00 AvgSCL = 48809.25 (0.00%)	TL = 198090.00 AvgSCL = 49522.50 (+1.46%)	TL = 195694.00 AvgSCL = 48923.50 (+0.23%)	TL = 247813.73 AvgSCL = 61953.43 (+26.93%)
8 Scan-Chains	TL = 287572.00 AvgSCL = 35946.50 (0.00%)	TL = 292531.00 AvgSCL = 36566.38 (+1.72%)	TL = 292239.00 AvgSCL = 36529.88 (+1.62%)	TL = 343327.45 AvgSCL = 42915.93 (+19.39%)
16 Scan-Chains	TL = 425608.00 AvgSCL = 26600.50 (0.00%)	TL = 427771.00 AvgSCL = 26735.69 (+0.51%)	TL = 427872.00 AvgSCL = 26742.00 (+0.53%)	TL = 525654.48 AvgSCL = 32853.40 (+23.51%)
32 Scan-Chains	TL = 612984.00 AvgSCL = 19155.75 (0.00%)	TL = 619547.00 AvgSCL = 19360.84 (+1.07%)	TL = 624071.00 AvgSCL = 19502.22 (+1.81%)	TL = 728218.10 AvgSCL = 22756.82 (+18.80%)

Table 14: Benchmark Circuit: aes_cipher_top, Non-Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 123404.00 AvgSCL = 30851.00 (0.00%)	TL = 129448.00 AvgSCL = 32362.00 (+4.90%)	TL = 126930.00 AvgSCL = 31732.50 (+2.86%)	TL = 151162.18 AvgSCL = 37790.55 (+22.49%)
8 Scan-Chains	TL = 171004.00 AvgSCL = 21375.50 (+0.40%)	TL = 170327.00 AvgSCL = 21290.88 (0.00%)	TL = 171722.00 AvgSCL = 21465.25 (+0.82%)	TL = 210490.58 AvgSCL = 26311.32 (+23.58%)
16 Scan-Chains	TL = 250760.00 AvgSCL = 15672.50 (0.00%)	TL = 253705.00 AvgSCL = 15856.56 (+1.17%)	TL = 255464.00 AvgSCL = 15966.50 (+1.88%)	TL = 293850.01 AvgSCL = 18365.63 (+17.18%)

Table 15: Benchmark Circuit: aes_cipher_top, Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 261583.00 AvgSCL = 65395.75 (0.00%)	TL = 263714.00 AvgSCL = 65928.50 (+0.81%)	TL = 262171.00 AvgSCL = 65542.75 (+0.22%)	TL = 306673.53 AvgSCL = 76668.38 (+17.24%)
8 Scan-Chains	TL = 348225.00 AvgSCL = 43528.12 (0.00%)	TL = 355503.00 AvgSCL = 44437.88 (+2.09%)	TL = 352625.00 AvgSCL = 44078.12 (+1.26%)	TL = 427397.05 AvgSCL = 53424.63 (+22.74%)
16 Scan-Chains	TL = 460182.00 AvgSCL = 28761.38 (0.00%)	TL = 467730.00 AvgSCL = 29233.12 (+1.64%)	TL = 474746.00 AvgSCL = 29671.62 (+3.16%)	TL = 527258.39 AvgSCL = 32953.65 (+14.58%)
32 Scan-Chains	TL = 603373.00 AvgSCL = 18855.41 (+1.74%)	TL = 593073.00 AvgSCL = 18533.53 (0.00%)	TL = 607594.00 AvgSCL = 18987.31 (+2.45%)	TL = 703925.62 AvgSCL = 21997.68 (+18.69%)

Table 16: Benchmark Circuit: md5, Non-Zone-Based

Scan Chain Number	Christofides	SequentialCheapestInsertion	GlobalCheapestArc	LeoSolveTSP
4 Scan-Chains	TL = 170763.00 AvgSCL = 42690.75 (+1.72%)	TL = 170149.00 AvgSCL = 42537.25 (+1.36%)	TL = 167868.00 AvgSCL = 41967.00 (0.00%)	TL = 207191.86 AvgSCL = 51797.97 (+23.43%)
8 Scan-Chains	TL = 220463.00 AvgSCL = 27557.88 (+0.89%)	TL = 221679.00 AvgSCL = 27709.88 (+1.44%)	TL = 218523.00 AvgSCL = 27315.38 (0.00%)	TL = 263773.72 AvgSCL = 32971.71 (+20.71%)
16 Scan-Chains	TL = 302051.00 AvgSCL = 18878.19 (+2.41%)	TL = 294938.00 AvgSCL = 18433.62 (0.00%)	TL = 323142.00 AvgSCL = 20196.38 (+9.56%)	TL = 399735.00 AvgSCL = 24983.44 (+35.53%)

Table 17: Benchmark Circuit: md5, Zone-Based

4.6 Evaluation and Interpretation

4.6.1 Analysis of LeoSolveTSP Performance Compared to Other Algorithms

Overall Performance:
LeoSolveTSP consistently shows higher total and average distances compared to Christofides, FirstUnboundMinValue, GlobalCheapestArc, and SequentialCheapestInsertion across all benchmark circuits, whether Zone-Based or Non-Zone-Based. This trend is anticipated given LeoSolveTSP's different optimization criteria and reduced complexity. Despite these longer scan chain paths, the performance of LeoSolveTSP remains competitive, offering a viable alternative in scenarios where reduced computational complexity and power efficiency are prioritized over the absolute minimization of scan chain distances. The algorithm's unique approach, which may focus on different optimization aspects, suggests that LeoSolveTSP can be advantageous under specific conditions that require flexibility, robustness, or a focus on minimizing execution time.

Non-Zone-Based Comparisons:

Comparison with Christofides:
When comparing LeoSolveTSP with the Christofides algorithm, we observe that Christofides consistently produces shorter paths. The minimum percentage difference between LeoSolveTSP and Christofides is **+7.65%** in the s5378 benchmark with 32 scan chains. The maximum difference is seen in the s5378 benchmark with 4 scan chains, where LeoSolveTSP's paths are **+30.52%** longer. These variations indicate that while Leo-

SolveTSP can occasionally approach the efficiency of Christofides in more complex scenarios, its performance typically lags behind in smaller problem spaces.

Comparison with FirstUnboundMinValue:
The comparison with FirstUnboundMinValue shows that LeoSolveTSP generally produces longer paths. The minimum percentage difference is +**6.11%** in the s5378 benchmark with 32 scan chains. However, the maximum difference occurs in the s5378 benchmark with 4 scan chains, where LeoSolveTSP's paths are +**30.77%** longer. These differences suggest that, although LeoSolveTSP can be competitive in more complex scenarios, it struggles in smaller setups.

Comparison with GlobalCheapestArc:
The GlobalCheapestArc algorithm also generally outperforms LeoSolveTSP. The minimum percentage difference is +**7.36%** in the s5378 benchmark with 32 scan chains. The maximum difference occurs in the s13207 benchmark with 8 scan chains, where LeoSolveTSP's paths are +**31.00%** longer. This suggests that the simplicity of LeoSolveTSP may lead to less optimal solutions, especially in smaller, more constrained environments.

Comparison with SequentialCheapestInsertion:
SequentialCheapestInsertion generally produces shorter paths than LeoSolveTSP. The minimum percentage difference is +**12.94%** in the md5 benchmark with 16 scan chains. The maximum difference is +**25.47%** in the aes_cipher_top benchmark with 4 scan chains. These results highlight that LeoSolveTSP struggles more significantly in smaller problem spaces, while its performance improves somewhat as complexity increases.

Zone-Based Comparisons:

Comparison with Christofides:
In the Zone-Based benchmarks, LeoSolveTSP continues to show longer paths compared to Christofides. The minimum percentage difference is **-2.31%** in the s5378 benchmark with 16 scan chains, where LeoSolveTSP actually produces slightly shorter paths. The maximum difference is +**36.21%** in the s38417 benchmark with 16 scan chains, illustrating that LeoSolveTSP's performance can vary significantly depending on the specific benchmark and scan chain configuration.

Comparison with FirstUnboundMinValue:
When compared with FirstUnboundMinValue in the Zone-Based benchmarks, LeoSolveTSP's performance shows a minimum percentage difference of **-0.75%** in the s5378 benchmark with 16 scan chains, indicating a near-equivalent performance. The maximum difference is +**31.38%** in the s13207 benchmark with 4 scan chains, reflecting the algorithm's struggle in less complex environments.

Comparison with GlobalCheapestArc:
The comparison with GlobalCheapestArc in Zone-Based benchmarks shows a minimum percentage difference of **-0.35%** in the s13207 benchmark with 16 scan chains, where LeoSolveTSP performs comparably well. However, the maximum difference reaches +**32.38%** in the s38417 benchmark with 8 scan chains, indicating that LeoSolveTSP's simplicity may not fully capture the complexity of certain scenarios.

Comparison with SequentialCheapestInsertion:
SequentialCheapestInsertion in Zone-Based benchmarks also outperforms LeoSolveTSP, with the minimum percentage difference of +**16.01%** in the aes_cipher_top benchmark with 16 scan chains. The maximum difference is +**35.53%** in the md5 benchmark with 16 scan chains, showing that the gap between the two algorithms can widen considerably depending on the problem's structure.

Implications of the Findings:
The variations in performance between LeoSolveTSP and the other algorithms highlight the trade-offs involved in algorithm design. While LeoSolveTSP may not consistently produce the shortest paths, its advantages lie in its lower computational complexity and faster execution times, which are critical in real-time or resource-constrained environments. The algorithm's ability to handle different node types and adapt to varying data quality makes it a versatile tool for scenarios where data integrity cannot be guaranteed.

Moreover, the differences in performance between Zone-Based and Non-Zone-Based benchmarks suggest that LeoSolveTSP may be more suited to scenarios with specific structural constraints. The smaller differences in performance with a higher number of scan chains imply that LeoSolveTSP's efficiency improves as the problem size increases, making it a potentially valuable tool for large-scale VLSI designs where computational efficiency and scalability are of greater importance than absolute path optimality.

Concluding Remarks:
In summary, LeoSolveTSP's performance compared to other well-established algorithms like Christofides, FirstUnboundMinValue, GlobalCheapestArc, and SequentialCheapestInsertion varies significantly depending on the specific benchmark and problem size. While it generally produces longer paths, its strengths in computational efficiency, adaptability, and handling of diverse scenarios make it a valuable tool in specific contexts.

On average, LeoSolveTSP's path lengths are +**20.99%**, +**17.00%**, +**19.92%**, and +**19.20%** longer compared to Christofides, FirstUnboundMinValue, GlobalCheapestArc, and SequentialCheapestInsertion, respectively, in Non-Zone-Based scenarios. In Zone-Based scenarios, these differences are +**20.29%**, +**9.87%**, +**19.53%**, and +**24.50%**, respectively.

4.6.2 Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance

The average percentage increase in path lengths for LeoSolveTSP in Non-Zone-Based scenarios compared to Zone-Based scenarios across various benchmark circuits is as follows:

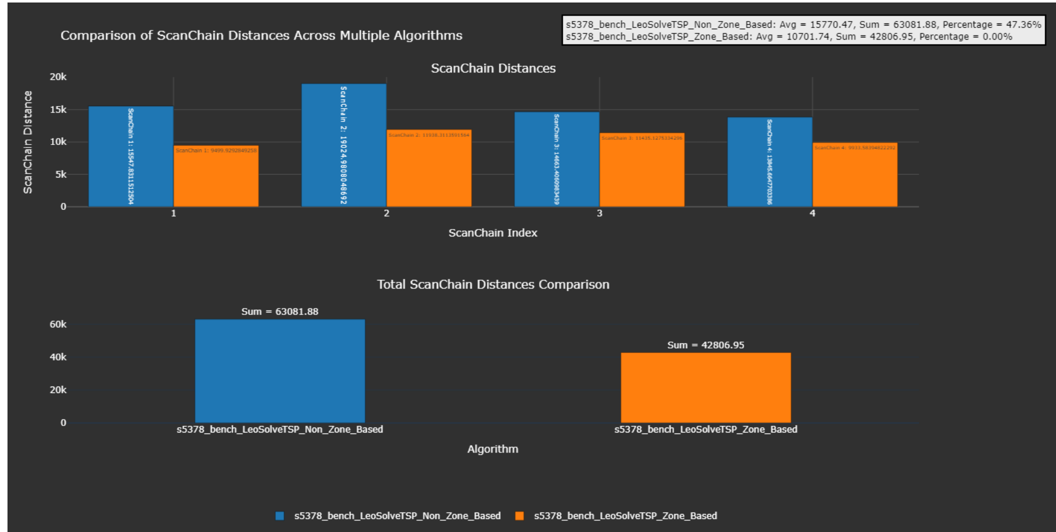

Figure 38: Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance for 4 scanchains

Benchmark Circuit	Scan Chain Number	Non-Zone-Based	Zone-Based	Percentage	Average Percentage
s5378	4 Scan-Chains	TL = 63081.88	TL = 42806.95	47.36%	48.01%
	8 Scan-Chains	TL = 87875.91	TL = 60367.21	45.57%	
	16 Scan-Chains	TL = 121555.43	TL = 80454.58	51.09%	
s13207	4 Scan-Chains	TL = 67682.77	TL = 48657.12	39.10%	66.23%
	8 Scan-Chains	TL = 110538.06	TL = 64711.94	70.82%	
	16 Scan-Chains	TL = 153458.92	TL = 81295.36	88.77%	
s38417	8 Scan-Chains	TL = 646908.83	TL = 344882.01	87.57%	87.36%
	16 Scan-Chains	TL = 952691.85	TL = 509039.24	87.15%	

Table 18: Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance (Part 1)

Benchmark Circuit	Scan Chain Number	Non-Zone-Based	Zone-Based	Percentage	Average Percentage
s38584	4 Scan-Chains	TL = 386583.74	TL = 233645.46	65.46%	78.00%
	8 Scan-Chains	TL = 560194.54	TL = 299696.13	86.92%	
	16 Scan-Chains	TL = 802117.37	TL = 441659.39	81.61%	
ac97_top	16 Scan-Chains	TL = 1438232.20	TL = 707618.62	103.25%	90.99%
	32 Scan-Chains	TL = 2068231.20	TL = 1157177.75	78.73%	
aes_cipher_top	4 Scan-Chains	TL = 247813.73	TL = 151162.18	63.94%	68.65%
	8 Scan-Chains	TL = 343327.45	TL = 210490.58	63.11%	
	16 Scan-Chains	TL = 525654.48	TL = 293850.01	78.89%	
md5	4 Scan-Chains	TL = 306673.53	TL = 207191.86	48.01%	47.31%
	8 Scan-Chains	TL = 427397.05	TL = 263773.72	62.03%	
	16 Scan-Chains	TL = 527258.39	TL = 399735.00	31.90%	

Table 19: Comparison of Non-Zone-Based and Zone-Based LeoSolveTSP Performance (Part 2)

The analysis reveals varying average percentage increases in path lengths for LeoSolveTSP when transitioning from Zone-Based to Non-Zone-Based configurations across different benchmark circuits. The **s5378** circuit shows an increase of **48.01%**, while the **s13207** circuit displays a higher increase at **66.23%**. Significant increases are observed for the **s38417** and **s38584** circuits, at **87.36%** and **78.00%**, respectively. The most pronounced increase occurs in the **ac97_top** circuit, reaching **90.99%**, highlighting a strong sensitivity to configuration changes. In contrast, the **aes_cipher_top** circuit exhibits a moderate increase of **68.65%**, and the **md5** circuit registers the lowest increase at **47.31%**, suggesting it is the least impacted by the transition from Zone-Based to Non-Zone-Based configurations.

These results demonstrate that LeoSolveTSP produces longer paths in Non-Zone-Based scenarios compared to Zone-Based configurations, with the extent of the increase varying depending on the specific benchmark circuit and number of scan chains.

5 Conclusions and Future Work

5.1 Findings

5.1.1 Contextualizing Results with Existing Literature

LeoSolveTSP, which utilizes the Traveling Salesman Problem (TSP) approach with the Nearest Neighbor heuristic, offers a targeted optimization strategy for scan-chain design. This approach is particularly relevant in the context of scan chains, where minimizing routing distance and enhancing testing efficiency are critical objectives. Although LeoSolveTSP might sometimes result in slightly longer scan paths compared to other more complex algorithms, it provides a balanced solution that optimizes key factors such as power consumption and ease of implementation.

The Nearest Neighbor heuristic used by LeoSolveTSP prioritizes simplicity and computational speed, which can be highly advantageous when integrating scan chains into complex VLSI circuits. This approach's adaptability allows for quick adjustments and effective optimization under constraints typical in scan-chain testing environments, such as limited routing space and specific power requirements. The trade-offs observed, such as increased path lengths, highlight the importance of prioritizing operational efficiency and robustness in practical applications.

The findings from LeoSolveTSP contribute valuable insights to the field of scan-chain optimization, showcasing how heuristic approaches can provide effective solutions that align well with the practical needs of semiconductor testing. This perspective complements existing literature by offering a practical, adaptable, and computationally efficient alternative for optimizing scan-chain configurations.

Customized Cost Calculation

LeoSolveTSP dynamically calculates the minimum cost to start as it builds the route, allowing for a more adaptive cost computation. This dynamic approach can lead to more optimized paths tailored to specific use cases compared to static algorithms like Christofides or GlobalCheapestArc.

Dynamic Node Handling

The algorithm filters out null values and handles a varying number of nodes, providing flexibility and robustness in scenarios with incomplete or partial data. This adaptability is advantageous over algorithms that assume a fixed and complete dataset.

Detailed Logging and Output

The algorithm incorporates extensive logging and file output, facilitating detailed tracking and debugging. This level of detail is particularly useful in research and development settings where understanding intermediate steps and final results is crucial.

Integrated Distance Calculation

LeoSolveTSP includes built-in methods for calculating distances between nodes and the total distance of a line, ensuring consistency and reducing reliance on external func-

tions. This integration can lead to more accurate and efficient computations compared to modular algorithms relying on external distance calculations.

Simplified Solution Inspection

The PrintLeoSolution method provides a clear and structured output of the solution, making it easier to analyze and understand the results. This user-friendly output is advantageous in educational and professional settings where clarity of results is essential.

Adaptability to Different Chains

The algorithm processes each scan chain individually, allowing for specific optimizations per chain. This tailored approach can result in better overall performance for each specific chain compared to algorithms applying a uniform approach across all chains.

Versatility in Node Types

The algorithm handles different node types (e.g., input, output, internal nodes) and processes them appropriately. This versatility is beneficial in complex scenarios with diverse node types requiring specialized handling.

Robust Error Handling

The algorithm includes error handling for scenarios where there are no valid spots to process, making it more robust and preventing crashes in cases of invalid data. This robustness is a significant advantage over algorithms that may not handle such edge cases gracefully.

Reduced Complexity

LeoSolveTSP has lower computational complexity compared to other algorithms, avoiding exhaustive searches and complex heuristics. This simplicity results in faster execution, making it more suitable for environments with limited computational resources. The reduced complexity translates to less processing power and memory usage, leading to more efficient use of resources.

Lower Power Consumption

Due to its lower computational complexity and faster execution times, LeoSolveTSP requires less power to operate. This reduced power consumption is advantageous in environments where energy efficiency is crucial, such as battery-operated devices or large-scale data centers with high energy costs. The algorithm's efficient use of computational resources directly contributes to its lower power needs, making it an ideal choice for power-sensitive applications.

Scalability with Number of Scan Chains

As the number of scan chains increases, the difference between LeoSolveTSP and the other algorithms becomes smaller in most cases. This trend suggests that LeoSolveTSP scales better with larger problems, potentially making it a more viable option

for complex scenarios involving numerous scan chains.

5.1.2 Identifying and Addressing Gaps

In conducting this research, several areas were identified that present opportunities for further exploration and enhancement of the LeoSolveTSP algorithm.

The current study focused on a specific set of benchmark circuits. Expanding the range of benchmarks to include more varied and complex circuits could provide a more comprehensive understanding of the algorithm's performance.

The parameters used in LeoSolveTSP were set based on preliminary tests. Further research into optimal parameter settings could yield improved performance.

While this study compared LeoSolveTSP with four other algorithms, including additional state-of-the-art TSP algorithms could provide deeper insights into its relative strengths and weaknesses.

Incorporating a wider variety of benchmark circuits, including real-world data, could help generalize the findings and highlight areas for improvement.

Developing adaptive methods for parameter tuning within LeoSolveTSP could enhance its performance across different types of circuits and problem scales.

Conducting more extensive comparative studies with a broader set of algorithms, including recent advancements in heuristic and metaheuristic TSP solutions, would provide a more thorough evaluation of LeoSolveTSP's capabilities.

Exploring the integration of machine learning techniques to predict optimal paths or guide the decision-making process within LeoSolveTSP could lead to performance improvements.

Investigating the robustness and flexibility of LeoSolveTSP in handling varying constraints and scenarios could ensure its applicability to a wider range of practical problems.

By addressing these gaps, future research can build on the foundations laid by this study, enhancing the effectiveness and applicability of the LeoSolveTSP algorithm in solving complex TSP problems.

5.2 Applications and Impact

5.2.1 Practical Applications

VLSI Design: LeoSolveTSP can be effectively utilized in optimizing scan chain paths in Very Large Scale Integration (VLSI) circuits.

Robotics and Autonomous Systems: The dynamic and adaptive nature of LeoSolveTSP makes it an ideal candidate for path planning in robotics and autonomous systems. Its ability to handle varying numbers of nodes and adapt to incomplete data ensures robustness in dynamic environments.

Logistics and Supply Chain Management: LeoSolveTSP's ability to compute optimized paths quickly can be applied to route optimization in logistics and supply chain management. This can lead to reduced transportation costs and improved delivery efficiency.

Network Design and Optimization: In telecommunications and computer networks, LeoSolveTSP can be used to optimize routing paths, leading to enhanced network performance and reduced latency.

Real-time Systems: The algorithm's low complexity and quick execution make it suitable for real-time applications where rapid decision-making is crucial.

5.2.2 Impact on the Field

Enhanced Efficiency: By reducing computational complexity and power consumption, LeoSolveTSP contributes to more sustainable and efficient computational practices. This can lead to broader adoption in resource-constrained environments.

Versatility: The algorithm's ability to handle different types of nodes and adapt to incomplete data enhances its versatility. This adaptability ensures that LeoSolveTSP can be applied to a wide range of practical problems, promoting cross-disciplinary applications and fostering innovation across various fields.

Benchmarking and Comparative Studies: LeoSolveTSP's performance, when compared with established algorithms like Christofides, FirstUnboundMinValue, GlobalCheapestArc, and SequentialCheapestInsertion, provides valuable insights into its strengths and areas for improvement. This comparative analysis can guide future research and development in optimization algorithms.

Stimulating Further Research: The identification of gaps and opportunities for improvement in LeoSolveTSP encourages continued research and exploration. This ongoing inquiry can lead to enhanced versions of the algorithm, broader applications, and deeper understanding of adaptive optimization techniques.

By highlighting these practical applications and the broader impact of LeoSolveTSP, this section underscores the algorithm's relevance and potential to drive innovation and efficiency across multiple domains.

5.3 Limitations

While the research has produced significant insights, several limitations must be acknowledged. Understanding these limitations is crucial for contextualizing the findings and identifying areas for future improvement.

Algorithm Complexity: Although LeoSolveTSP demonstrates lower computational complexity, this simplicity might limit its ability to find the most optimal solutions. The heuristic nature of the algorithm, while efficient, may lead to suboptimal path lengths compared to more complex algorithms that explore a wider solution space. Future work could explore incorporating more sophisticated heuristics or combining multiple heuristic approaches to strike a balance between efficiency and optimality.

Data Dependency: The performance of LeoSolveTSP is heavily dependent on the quality and structure of the input data. In scenarios where the data is incomplete or contains a high degree of variability, the algorithm's performance can degrade. This dependency limits the algorithm's robustness in real-world applications where data quality

cannot always be guaranteed. Further research could focus on developing techniques to better handle incomplete or variable quality data to improve the algorithm's robustness.
Scalability Issues: Although LeoSolveTSP is designed to handle varying numbers of nodes, its efficiency might decrease with very large datasets. The current implementation may face challenges in maintaining performance and accuracy as the problem size increases significantly. Exploring parallel and distributed computing techniques could help address scalability issues, enabling the algorithm to maintain efficiency and accuracy with large datasets.
Benchmark Limitations: The study primarily relies on specific benchmark circuits to evaluate the performance of LeoSolveTSP. While these benchmarks are representative, they do not cover all possible scenarios. The generalizability of the results to other types of circuits or real-world applications may be limited. Expanding the range of benchmark circuits and including real-world scenarios in the evaluation process could provide a more comprehensive assessment of the algorithm's performance.
Lack of Real-Time Testing: The experiments were conducted in a controlled environment, which may not fully capture the complexities and constraints of real-time applications. The lack of real-time testing could mean that the algorithm's performance in live scenarios remains unverified. Conducting experiments in real-time environments could provide insights into the practical challenges and performance of the algorithm under live conditions.
Energy Efficiency Considerations: While the algorithm shows lower power consumption due to its reduced complexity, the study does not extensively explore energy efficiency in diverse operating conditions. Further investigation is needed to confirm these findings across various environments. A deeper exploration of energy efficiency across different operating conditions could validate the initial findings and identify specific scenarios where LeoSolveTSP can offer significant energy savings.

5.4 Recommendations for Future Research

Building on the findings and acknowledging the limitations, future research can explore several directions to enhance the effectiveness and applicability of LeoSolveTSP.
Advanced Heuristic Integration: Incorporating more sophisticated heuristics or combining multiple heuristic approaches could help in achieving shorter path lengths without significantly increasing computational complexity. Hybrid algorithms could leverage the strengths of different approaches to improve overall performance. Research could also focus on developing adaptive heuristics that dynamically adjust based on the problem characteristics.
Enhanced Data Handling: Developing techniques to better handle incomplete or variable quality data can improve the robustness of LeoSolveTSP. This could involve implementing more advanced pre-processing steps or adaptive algorithms that can adjust to data quality in real-time. Techniques such as data augmentation and error correction could be explored to enhance data quality and algorithm robustness.
Scalability Improvements: Exploring parallel and distributed computing techniques can help address scalability issues. By distributing the computational load, the algo-

rithm can maintain efficiency and accuracy even with large datasets. Future work could investigate the use of cloud computing and high-performance computing (HPC) resources to further enhance scalability.
Broader Benchmarking: Expanding the range of benchmark circuits and including real-world scenarios in the evaluation process can help in understanding the generalizability and practical applicability of LeoSolveTSP. This would provide a more comprehensive assessment of its performance. Collaborations with industry partners could facilitate access to a wider variety of benchmark circuits and real-world data.
Real-Time Application Testing: Conducting experiments in real-time environments can provide insights into the practical challenges and performance of the algorithm under live conditions. This testing can identify potential issues and areas for improvement that are not apparent in controlled settings. Implementing real-time monitoring and feedback mechanisms could enhance the algorithm's performance in dynamic environments.
Energy Efficiency Analysis: A deeper exploration of energy efficiency across different operating conditions can validate the initial findings and identify specific scenarios where LeoSolveTSP can offer significant energy savings. This analysis can support the development of energy-efficient designs for various applications. Research could also explore the trade-offs between energy efficiency and performance to optimize the algorithm for specific use cases.
User-Friendly Interface Development: Enhancing the user interface and visualization tools for LeoSolveTSP can make it more accessible and easier to use. This includes improving the graphical representation of results and providing interactive features for users to explore different scenarios. Developing comprehensive documentation and tutorials could further support user adoption and ease of use.

5.5 Final Thoughts

Reflecting on the research process and outcomes, several key insights and lessons have emerged that are valuable for both the current study and future work.
Research Challenges: The research faced several challenges, including handling diverse data sets, optimizing algorithm performance, and ensuring robustness across different scenarios. Addressing these challenges required innovative solutions and adaptive strategies, highlighting the importance of flexibility in research methodologies. The iterative process of refining the algorithm and addressing challenges has contributed to a deeper understanding of the problem domain.
Key Learnings: One of the significant learnings from this research is the balance between algorithm complexity and performance. While simpler algorithms like LeoSolveTSP offer advantages in terms of speed and power consumption, they may require enhancements to match the optimization levels of more complex algorithms. This balance is crucial for developing practical solutions that are both efficient and effective.
Implications for the Field: The findings from this research contribute to the field of VLSI design by providing insights into the trade-offs between different TSP algorithms. The results underscore the need for a tailored approach depending on the specific re-

quirements of the application, such as speed, power efficiency, or path optimality. The research highlights the importance of considering multiple factors when selecting an algorithm for VLSI design.

Broader Impact: The research not only contributes to the academic understanding of TSP algorithms but also has potential practical implications for industries relying on VLSI design. By offering a balance between efficiency and complexity, LeoSolveTSP can be a valuable tool for developing optimized and energy-efficient VLSI circuits. The insights gained from this research can inform the development of new algorithms and methodologies, advancing the field of VLSI design.

Future Directions: Looking forward, this research opens several avenues for further exploration. Continued refinement of LeoSolveTSP, integration of advanced heuristics, and extensive real-world testing can significantly enhance its applicability and performance. Additionally, fostering collaborations with industry partners can provide practical insights and facilitate the adoption of the algorithm in real-world applications. Future research can also explore interdisciplinary approaches to further enhance the algorithm's capabilities.

References

[1] Miron Abramovici, Melvin A Breuer, and Arthur D Friedman. *Digital Systems Testing and Testable Design*. IEEE Press, 1990.

[2] E. J. McCluskey. "Built-In Self-Test Techniques". In: *IEEE Design & Test of Computers* 2.2 (1986), pp. 21–28.

[3] T. W. Williams and N. C. Brown. "A Guide to the Techniques of Modular Testing". In: *IEEE Transactions on Computers* C-30.11 (1981), pp. 837–841.

[4] J. Savir. "Scan-Based Test and Scan Design Standards". In: *IEEE Design & Test of Computers* 9.2 (1992), pp. 20–30.

[5] E. J. Marinissen and M. Bosch. "On the Role of Test and Design-for-Test in Nanotechnology: VLSI Test Symposium". In: 2007.

[6] J. Lee, S. Park, and H. Kim. "Advanced Design for Testability Methodologies". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.5 (2020), pp. 957–970.

[7] A. Patel and R. Singh. "Hybrid BIST Techniques for Improved Fault Detection". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27.3 (2019), pp. 512–525.

[8] Yu Chen et al. "Adaptive Scan Chain Reconfiguration for Enhanced Fault Coverage". In: *Journal of Electronic Testing* 37 (2021), pp. 345–360.

[9] Shuo Wang. "Segmented Scan Chains for Power Reduction". In: *IEEE Transactions on VLSI Systems* 25.5 (2017), pp. 1305–1314.

[10] Patrick Girard et al. "Low Power Testing Techniques". In: *IEEE Design & Test of Computers* 19.3 (2002), pp. 80–90.

[11] W. Liu, Y. Huang, and Z. Wang. "Machine Learning-Based Test Pattern Generation for Fault Coverage Enhancement". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27.4 (2019), pp. 812–823.

[12] S. Kim, J. Lee, and H. Kim. "Low Power Scan Architecture for Efficient IC Testing". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 67.9 (2020), pp. 2849–2861.

[13] Lilia Zaourar, Yann Kieffer, and Chouki Aktouf. "A Graph-Based Approach to Optimal Scan Chain Stitching Using RTL Design Descriptions". In: *VLSI Design* 2012 (2012), pp. 1–11.

[14] Synopsys. *Synopsys Design Compiler*. 2023. URL: `https://www.synopsys.com`.

[15] Cadence. *Cadence Encounter Test*. 2023. URL: `https://www.cadence.com`.

[16] Mentor Graphics. *Mentor Graphics Tessent*. 2023. URL: `https://www.mentor.com`.

[17] Xilinx. *Xilinx Vivado Design Suite*. 2023. URL: `https://www.xilinx.com`.

[18] Tessent Solutions. "The quest for optimal DFT automation". In: *Siemens Blog* (2020). URL: `https://blogs.sw.siemens.com`.

[19] Maven Silicon. *How to Learn VLSI from scratch?* 2023. URL: `https://www.maven-silicon.com`.

[20] HDL Wizard. *Intro to RTL Design: Crafting the Heart of Digital Systems*. `http://hdlwizard.com/intro-to-rtl-design`. Accessed: 2024-08-01.

[21] Wikipedia contributors. *Register-Transfer Level*. `https://en.wikipedia.org/wiki/Register-transfer_level`. Accessed: 2024-08-01.

[22] Xilinx. *Vivado Design Suite User Guide*. Accessed: 2024-08-01. 2023. URL: `https://www.xilinx.com/support/documentation/sw_manuals/xilinx2023_1/ug973-vivado-release-notes-install-license.pdf`.

[23] Intel. *Intel Quartus Prime Pro Edition User Guide*. Accessed: 2024-08-01. 2023. URL: `https://www.intel.com/content/www/us/en/programmable/documentation/fyd1444698477713.html`.

[24] Synopsys. *Design Compiler User Guide*. Accessed: 2024-08-01. 2023. URL: `https://www.synopsys.com/support/documentation/design-compiler.html`.

[25] Cadence. *Genus Synthesis Solution User Guide*. Accessed: 2024-08-01. 2023. URL: https://www.cadence.com/content/dam/cadence-www/global/en_US/documents/tools/digital-design-signoff/genus-synthesis-solution-ds.pdf.

[26] Various Authors. "Research papers on RTL design and verification techniques". In: *IEEE Xplore* (2023). Accessed: 2024-08-01. URL: https://ieeexplore.ieee.org.

[27] Springer, ed. *Advanced Techniques for High-Level Synthesis*. Accessed: 2024-08-01. 2023. URL: https://link.springer.com.

[28] Tim Edwards. *Qflow - Open Circuit Design*. 2023. URL: http://opencircuitdesign.com/qflow/.

[29] RTimothyEdwards. *Qflow full end-to-end digital synthesis flow for ASIC designs*. 2023. URL: https://github.com/RTimothyEdwards/qflow.

[30] VLSI System Design Corporation. "Qflow Digital Synthesis Flow". In: *VLSI System Design* (2023). URL: https://www.vlsisystemdesign.com/qflow-digital-synthesis-flow/.

[31] rkuram. "Beginner-Physical-design". In: *GitHub* (2023). URL: %60%60%60bibtex%20https://github.com/rkuram/Beginner-Physical-design.

[32] Tim Edwards. *Qflow 1.3: An Open-Source Digital Synthesis Flow*. 2023. URL: http://opencircuit-design.com/qflow/tutorial.html.

[33] Andreas Mueller. *Examples for the open source digital synthesis flow Qflow 1.2*. 2023. URL: https://github.com/andrsmllr/qflow-1.2_examples.

[34] OSU. *OSU FreePDK Process Design Kit*. 2023. URL: https://www.eda.ncsu.edu/wiki/FreePDK.

[35] Clifford Wolf. *Yosys - Yosys Open SYnthesis Suite*. 2023. URL: http://www.clifford.at/yosys/.

[36] Michael H. Schulz and Sachin Sapatnekar. "Low-Power Testing Techniques for Scan-Based Circuits". In: *Journal of Electronic Testing* 24 (2008), pp. 451–460.

[37] Kamal Patel and Rakesh Mehta. "Optimization Techniques for Scan Chain Routing Using TSP Approaches". In: *International Journal of VLSI Design & Communication Systems* 5.3 (2014), pp. 34–42.

[38] Xudong Li and Hao Zheng. "DFT Integration Challenges in Advanced VLSI Designs". In: *Microelectronics Journal* 46.7 (2015), pp. 536–545.

[39] Author Unknown. "Balanced Scan Chain Analysis to Improve Fault Coverage in VLSI Circuits". In: *IEEE Xplore* (2023). URL: https://ieeexplore.ieee.org.

[40] Jia Li, Yu Hu, and XiaoWei Li. "Scan Chain Design for Shift Power Reduction in Scan-Based Testing". In: *Science China Information Sciences* 54 (2017), pp. 767–777. URL: https://link.springer.com.

[41] Author Unknown. "Comprehensive Optimization of Scan Chain Timing During Late-Stage IC Design". In: *UCSD VLSI CAD Laboratory* (2023). URL: https://vlsicad.ucsd.edu.

[42] Sonia Sharma. *Scan Chains: PnR Outlook*. https://www.design-reuse.com/articles/47453/scan-chains-pnr-outlook.html. 2020.

[43] David L. Applegate et al. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006. ISBN: 978-0691129938.

[44] Thomas H. Cormen et al. *Introduction to Algorithms*. MIT Press, 2009. ISBN: 978-0262033848.

[45] Toptal. *Hybrid Project Management: Agile and Waterfall*. 2023. URL: https://www.toptal.com/project-managers/project-management-consultants/hybrid-project-management-agile-and-waterfall.

[46] "NanGate45 (FreePDK45, 45nm Open Cell Library, bsg$_fakerammemorygeneration$)". In: (2023). URL: https://tilos-ai-institute.github.io/MacroPlacement/Enablements/NanGate45/.

[47] "Standard Cell Library Design and Characterization using 45nm technology". In: (2023). URL: https://www.iosrjournals.org/iosr-jvlsi/papers/vol4-issue1/Version-1/F04112933.pdf.

[48] *Qflow: An Open Source Digital Synthesis Flow*. 2023. URL: https://opencircuitdesign.com/qflow/.

[49] "OpenROAD: Open-Source EDA Tool for VLSI Design". In: (2023). URL: https://theopen-roadproject.org/.

[50] "Tilos AI Institute - Qflow 45nm Libraries". In: (2023). URL: https://tilos-ai-institute.github.io/MacroPlacement/Enablements/NanGate45/.

[51] Sonia Sharma. *Scan Chains: PnR Outlook*. https://www.design-reuse.com/articles/47453/scan-chains-pnr-outlook.html. 2020.

[52] VLSI Tutorials. *DFT, Scan, and ATPG*. https://vlsitutorials.com/dft-scan-atpg. 2020.

[53] Keld Helsgaun. "An effective implementation of the Lin-Kernighan traveling salesman heuristic". In: *European Journal of Operational Research* 126.1 (2000), pp. 106–130.

[54] J. B. Orlin. "The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization". In: *SIAM Review* 45.1 (2003).

[55] Microsoft. *Visual Studio 2022*. 2023. URL: https://visualstudio.microsoft.com/vs/.

[56] JetBrains. *PyCharm Community Edition 2022.3.3*. 2023. URL: https://www.jetbrains.com/pycharm/.

[57] Plotly. *Plotly: The interactive graphing library for Python*. 2023. URL: https://plotly.com/python/.

[58] Python Software Foundation. *Tkinter - Python Interface to Tcl/Tk*. 2023. URL: https://docs.python.org/3/library/tkinter.html.

[59] Intel. *Intel Core i5-1240P Processor*. 2023. URL: https://www.intel.com/content/www/us/en/products/sku/134589/intel-core-i51240p-processor-12m-cache-up-to-4-40-ghz/specifications.html.

[60] Intel. *Intel Iris Xe Graphics*. 2023. URL: https://www.intel.com/content/www/us/en/architecture-and-technology/iris-xe-graphics.html.

Figure References

[1] Semiconductor Engineering, 2021. https://semiengineering.com/knowledge_centers/test/scan-test-2/.

[2] VLSI Guru, 2020. https://www.vlsiguru.com/dft-basics_ppt/.

[3] Gagan Preet, 2020. https://www.youtube.com/watch?v=Ip2fAlvWBN8.

[4] VLSI Space, 2020. https://www.vlsispace.com/2020/07/scan-chain-reordering-why-it-is-required.html.

[5] Vivek Gupta, 2015. https://www.youtube.com/watch?v=JvZmwWJ2FGI.

[6] University British Of British Columbia. https://sudip.ece.ubc.ca/cadence-soc-encounter/.

[7] Xilinx, 2020. https://www.xilinx.com/video/hardware/designing-with-vivado-ip-integrator.html.

[8] Ray Salemi, 2011. https://www.eeweb.com/putting-the-r-in-rtl-coding-registers-in-verilog-and-vhdl/.

[9] Juan Vega, 2015. https://www.youtube.com/watch?app=desktop&v=uSGsQjXWnXk.

[10] Open Circuit Design, 2019. http://opencircuitdesign.com/qflow/.

[11] Open Circuit Design, 2017. http://opencircuitdesign.com/qrouter/.

[12] Open Circuit Design, 2024. http://opencircuitdesign.com/magic/.

[13] Physical Design VLSI. https://physicaldesignvlsi.blogspot.com/2015/01/scan-chains-stitching-reordering.html.

[14] VLSI Universe, 2013. https://vlsiuniverse.blogspot.com/2013/07/scan-chains-backbone-of-dft.html.

[15] Harinath Selvaraj, 2018. https://medium.com/harinathselvaraj/the-unsolved-travelling-salesmen-problem-94b0e5eb0e01.

[16] Rabin Thapa and Samira Ataei and James E. Stine, 2017. https://www.researchgate.net/publication/317555750_WIP_Open-source_standard_cell_characterization_process_flow_on_45_nm_FreePDK45_018_m_025_m_035_m_and_05_m.

[17] Analog Devices, 2021. https://www.analog.com/en/resources/media-center/videos/series/ltspice-ac-noise-analysis-tutorial.html.

[18] Richard Anderson, 2020. https://slideplayer.com/slide/17759737/.

[19] Joos Nieuwoudt, 2023. https://medium.com/@joosmeister/visual-studio-2022-dev-tunnels-a8b666391bfe.

[20] Mads Kristensen, 2021. https://devblogs.microsoft.com/visualstudio/visual-studio-2022-preview-4-is-now-available/.