A Major Project Report on

LLR-BASED SUCCESSIVE CANCELLATION LIST DECODER FOR POLAR CODES WITH MULTIBIT DECISION

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

In

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

MALLAIAH GARI
ARAVIND
(22H55A0419)

MAROJU SIREESHA
(21H51A0493)

PALCHURI SRIKAR
(21H51A04C1)

Under the esteemed guidance of
Mr.M. Srinivas
(Assistant Professor)

Department of ECE
CMR COLLEGE OF ENGINEERING & TECHNOLOGY
(UGC Autonomous)
*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A^+ Grade
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2024- 2025

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC Autonomous)
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF ELECTONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the Major Project Phase-I report entitled **" LLR-Based Successive-Cancellation List Decoder for Polar Codes With Multibit Decision"** being submitted by **Malliah Gari Aravind(22H55A0419)**, **Maroju Sireesha (21H51A0493), Palchuri Srikar (21H51A04C1)** in partial fulfillment for the award of Bachelor of Technology **Electronics And Communication Engineering** in is a record of bonafide work carried out under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree.

Project Guide
Mr. M. Srinivas
Asst Prof, Dept of ECE

Head of the Department
Dr.P. Raveendrababu
Professor & Head of Dept of ECE

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project a grand success.

We are grateful to **Mr . M . Srinivas , Assistant Professor**, Department of Electronics And Communication Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We express our sincere gratitude to **Dr. J. Seetaram,** Project Coordinator, Electronics and Communication Engineering Department, CMR College of Engineering & Technology, for his continuous guidance, encouragement, and support throughout the completion of our project.We would like to thank**, Dr. P. Raveendrababu,** Head of the Department of Electronics and Communication Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete our project work successfully.

We are very grateful to **Dr. Ghanta Devadasu**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Dr. A Seshu Kumar,** Principal, CMR College of Engineering and Technology, for unwavering support, and encouragement throughout the duration of this project.

We wish to extend our deepest gratitude to the esteemed **Director Major Dr. V A Narayana**, CMR College of Engineering & Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Dept Name for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary& Correspondent, CMR Group of Institutions, and Shri Ch Abhinav Reddy, CEO, CMR Group of Institutions for their continuous care and support.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly or indirectly in completion of this project work.

MALLAIAH GARI ARAVIND 22H55A0419

MAROJU SIREESHA 21H51A0493

PALACHURI SRIKAR 21H51A04C1

DECLARATION

We are here by declare that results embodied in this Project on **"LLR-Based Successive-Cancellation List Decoder for Polar Codes With Multibit Decision"** are from work carried out by using Partial fulfillment of the requirements for the award of B.Tech degree.

We have not submitted this report to any other university/institute for the award of any other degree.

TEAM MEMBERS		**SIGNATURE**
MALLAIAH GARI ARAVIND	**22H55A0419**	
MAROJU SIREESHA	**21H51A0493**	
PALCHURI SRIKAR	**21H51A04C1**	

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	ii
	LIST OF TABLES	iii
	ABSTRACT	iv
1	**INTRODUCTION**	1
	1.1 Motivation and Applications	2
	1.2 Problem Statement	3
	1.3 Project Objectives	4
2	**LITERATURE SURVEY**	**6-11**
3	**PROPOSED SYSTEM**	**12**
	3.1. Objective of Proposed Model	13
	3.2. Algorithms Used for Proposed Model	14
	3.3. Designing	16
	3.3.1. Block Diagram	17
	3.4. Stepwise Implementation	20
4	**RESULTS AND DISCUSSION**	**23**
	4.1. Output Values	25
	4.2. Stimulation Results	27
	4.3. Performance Metrics	28
5	**CONCLUSION**	29
	5.1 Conclusion	30
	REFERENCES	**31**
	APPENDIX	**33**

List of Figures

FIGURE NO.	TITLE	PAGE NO.
1	Architecture	17
2	Metric Computation Unit	18
3	Processing Element	19
4	Multiple Path	38
5	Stimulation Results	27

ABSTRACT

Polar codes have garnered significant attention due to their capacity-achieving properties. The Successive Cancellation List (SCL) decoding algorithm is renowned for delivering excellent error-correcting performance for these codes. However, traditional SCL decoder hardware designs often suffer from substantial silicon area requirements and prolonged decoding latency. While some recent approaches have managed to reduce either speed or latency, achieving simultaneous optimization of both metrics has remained elusive. A novel approach introduces a general Log-Likelihood Ratio (LLR)-based SCL decoding algorithm capable of making multi-bit decisions. This algorithm, termed **LLR-2Kb-SCL**, can determine 2K bits simultaneously for any arbitrary K using LLR messages. Additionally, a reduced-data-width scheme has been proposed to decrease the critical path of the sorting block. Building upon this algorithm, a VLSI architecture for the new SCL decoder has been developed. Synthesis results show that LLR-2Kb-SCL decoders achieve significant reductions in both speed and latency compared to prior works. Specifically, the hardware efficiency of the proposed designs marks a considerable improvement. This advancement represents a significant step forward in the efficient hardware implementation of polar code decoders, offering a promising solution to the challenges of speed and latency in SCL decoding.

CHAPTER 1
INTRODUCTION

CHAPTER 1
INTRODUCTION

1.1. MOTIVATION AND APPLICATIONS

Polar codes, introduced by Erdal Arikan, have emerged as a powerful class of error-correcting codes, achieving capacity in symmetric binary-input channels. However, the traditional Successive-Cancellation (SC) decoder, while efficient, faces limitations in performance, particularly at low signal-to-noise ratios (SNRs), due to its bit-by-bit sequential decision process. To address these limitations, the Successive-Cancellation List (SCL) decoder was developed, which maintains multiple candidate codeword paths and selects the most likely one, improving error correction performance at the cost of higher computational complexity. A promising enhancement to the SCL decoder is the use of Log-Likelihood Ratios (LLRs) combined with multibit decision strategies. This approach enables the decoder to make decisions on more than one bit at each stage, rather than making binary decisions sequentially. By using multibit decisions, the decoder can leverage more information at each step, which improves the overall decoding accuracy and reduces the number of decoding iterations required.

The LLR-based SCL decoder with multibit decision thus offers a significant trade-off between error correction performance and computational complexity, making polar codes more practical for real-world applications. The primary motivation behind this approach is to enhance decoding speed and reliability while minimizing the computational burden, particularly in high-throughput or low-latency environments. This makes it highly suitable for applications in 5G and beyond wireless communication systems, where polar codes are employed for control channels and other critical functions.

Additionally, this decoding method is beneficial in satellite communication, where long transmission distances and harsh channel conditions demand robust error correction, and in wireless sensor networks where low power consumption and reliable communication are essential.

Furthermore, the multibit decision strategy improves the efficiency of machine-to-machine (M2M) communication by reducing the number of retransmissions and improving throughput. Ultimately, LLR-based SCL decoding with multibit decision strategies makes polar codes a more viable and efficient option for modern communication systems, enabling faster, more reliable, and energy-efficient data transmission across a variety of applications.

1.2. PROBLEM STATEMENT

The primary challenge in the decoding of polar codes using the traditional Successive-Cancellation (SC) algorithm lies in its inherent limitations in performance and efficiency, particularly in noisy communication environments. The SC decoder processes one bit at a time, making sequential decisions that can lead to suboptimal error-correction performance, especially at low signal-to-noise ratios (SNRs). Although the Successive-Cancellation List (SCL) decoder improves upon this by maintaining multiple decoding paths (or lists), allowing for more accurate decoding through the selection of the most likely codeword, it introduces significant computational complexity. This complexity arises from the need to manage multiple paths, increasing both the decoding time and the memory requirements, which limits its scalability in high-throughput or low-latency applications. A further issue is the inefficiency of making binary decisions at each step, as this approach often fails to fully exploit the available information in the channel.

The introduction of Log-Likelihood Ratios (LLRs) combined with multibit decision strategies presents a potential solution to these problems. Multibit decision decoding enables the simultaneous consideration of multiple bits at each decision point, rather than relying on sequential bit-by-bit decisions. This method can improve decoding accuracy and reduce the number of required decoding iterations, thus speeding up the process and reducing the size of the list. However, integrating LLRs and multibit decisions into the SCL decoder brings its own set of challenges. Specifically, it requires careful management of the trade-off between improved error correction performance and increased computational complexity, as well as the development of algorithms capable of efficiently handling multibit decisions while still maintaining low latency and computational efficiency.

The problem at hand is to design a decoder that optimally combines the benefits of LLR-based SCL decoding with multibit decision strategies while mitigating the increase in complexity. This involves finding efficient methods for list management, optimizing the decision process to leverage LLRs, and minimizing the computational overhead in order to make polar codes practical for high-speed communication systems, low-latency applications, and power-constrained environments such as IoT and satellite communication. The goal is to achieve a decoder that delivers robust error correction performance without sacrificing decoding speed

or efficiency, ultimately making polar codes a more viable solution for modern wireless communication standards like 5G and beyond.

1.3. PROJECT OBJECTIVES

The primary objective of the project focused on the LLR-Based Successive-Cancellation List Decoder for Polar Codes with Multibit Decision is to develop an efficient and high-performance decoding algorithm that enhances the capabilities of polar codes in modern communication systems, particularly those requiring high-speed and low-latency decoding.

The key goals are to improve both decoding accuracy and speed by integrating Log-Likelihood Ratios (LLRs) and multibit decision strategies into the Successive-Cancellation List (SCL) decoding process.

This involves optimizing the SCL decoder to make multibit decisions at each stage, enabling the decoder to consider more than one bit per decision, which can significantly improve decoding performance by better leveraging the available channel information.

A critical aspect of this project is to minimize the computational complexity of the enhanced SCL decoder while maintaining its improved error-correction capabilities

CHAPTER 2
LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

2.1 Efficient List Decoder Architecture For Polar Codes

An efficient list decoder architecture for polar codes aims to enhance decoding performance by managing multiple candidate decoding paths in parallel. This approach is based on Successive-Cancellation List (SCL) decoding, a method particularly suited to polar codes. Polar codes, known for achieving Shannon capacity in specific channels, rely on channel polarization to distinguish between reliable and unreliable bits. In list decoding, rather than making a single decision at each step, the decoder maintains a list of possible decoding paths. This list is updated as more bits are decoded, with only the most likely paths retained for further processing. The ultimate goal is to reduce the likelihood of decoding errors by exploring multiple possibilities and selecting the one with the highest likelihood at the end of the process.

The main advantage of an efficient list decoder architecture is its ability to achieve significant error correction performance improvements, especially in high-noise environments. By holding multiple decoding paths, the architecture allows for a more comprehensive search of possible codewords, which makes it more resilient to noise compared to traditional Successive-Cancellation (SC) decoding. The list decoder is particularly effective when used in combination with a Cyclic Redundancy Check (CRC), which helps to identify the correct path from the list by checking for valid codewords. This combination of SCL and CRC decoding has been shown to achieve near maximum-likelihood decoding performance, making it ideal for applications requiring high reliability, such as 5G wireless communications.

However, while list decoding greatly enhances performance, it also brings increased computational complexity and resource demands. Maintaining multiple paths requires additional memory and computational power, which can make the decoder slower and less efficient, especially for large list sizes or high-throughput applications. This added complexity may lead to higher power consumption, which can be a significant drawback in mobile and low-power devices. Moreover, the latency associated with managing and evaluating multiple paths may limit the applicability of SCL decoders in real-time or low-latency environments.

2.2. Low-Latency Successive-Cancellation List Decoders for Polar Codes With Multibit Decision

Low-latency Successive-Cancellation List (SCL) decoders for polar codes with multibit decision aim to address the challenges of delay and complexity in traditional SCL decoding by allowing the simultaneous processing of multiple bits, rather than sequentially decoding each bit one by one. In standard SCL decoding, decisions are made iteratively across a list of potential paths, which can be computationally intensive and time-consuming as each bit requires individual attention. The multibit decision approach, however, evaluates multiple bits in parallel, enabling the decoder to exploit correlations between adjacent bits and reduce the overall decoding time. This approach retains the fundamental principle of list decoding—where a set of decoding paths is maintained and pruned based on likelihood estimates—but significantly speeds up the process by tackling groups of bits simultaneously.

The primary advantage of low-latency SCL decoders with multibit decision is their ability to achieve faster decoding speeds without drastically sacrificing error-correction performance. By handling multiple bits in parallel, these decoders effectively reduce latency, making them highly suitable for high-throughput applications, such as wireless communications in real-time systems. The multibit decision also reduces the sequential bottleneck seen in traditional SCL decoders, as fewer decoding stages are required. This architecture is particularly valuable in scenarios where timing is critical, as it offers a balance between the high reliability of list decoding and the speed requirements of low-latency systems. Additionally, multibit decision-making can lower the decoder's complexity, as fewer decisions mean fewer operations and, consequently, lower power consumption compared to handling each bit sequentially.

Despite these merits, low-latency SCL decoders with multibit decision have limitations. The primary challenge lies in achieving a suitable balance between the number of bits processed in parallel and the accuracy of the decoding decision. Grouping bits for parallel processing can introduce new types of decoding errors, as the dependencies between bits become harder to accurately manage in multibit groups.

This trade-off can lead to a marginal decline in error-correction performance compared to traditional bit-by-bit SCL decoding, especially in high-noise conditions. Additionally, the architecture required to support multibit decision-making can be complex, as it often involves sophisticated logic for path pruning and bit-group management, which can offset the latency gains with increased implementation complexity and potential hardware costs.

2.3. LLR-Based Successive Cancellation List Decoding of Polar Codes

Log-Likelihood Ratio (LLR)-based Successive Cancellation List (SCL) decoding is a powerful approach for decoding polar codes, leveraging LLR values to improve the accuracy and efficiency of the decoding process. In SCL decoding, multiple candidate decoding paths are maintained and pruned based on their likelihood, allowing the decoder to explore several potential solutions and ultimately select the most probable one. LLR-based SCL decoding enhances this approach by using LLR values, which quantify the probability of each bit being a 0 or 1 based on channel observations. These LLR values are derived directly from the channel output, providing a clear measure of bit reliability. By replacing raw probability metrics with LLRs, the decoder can make faster, more precise decisions, as LLRs simplify the mathematical operations required for evaluating bit reliability.

The main advantage of LLR-based SCL decoding lies in its combination of improved speed and error-correction performance. Using LLRs allows the decoder to handle bit-level reliability more directly, reducing the computational load typically associated with SCL decoding. This efficiency improvement is especially useful in applications requiring high throughput, as the LLR-based approach allows for faster path selection and pruning while maintaining the accuracy benefits of list decoding. Additionally, because LLRs represent bit reliability more intuitively, they enhance the decoder's ability to manage noise, making it well-suited for challenging channel conditions. This approach also lends itself to simpler hardware implementations, as LLR operations are compatible with straightforward arithmetic functions, making LLR-based SCL decoders more energy-efficient and easier to implement than traditional probability-based SCL decoders.

However, while LLR-based SCL decoding offers clear advantages, it also introduces some limitations. One challenge is the precision of LLR calculations, which can affect the decoder's performance in low-SNR (signal-to-noise ratio) environments. In these cases, the LLR values may be less reliable, potentially leading to incorrect path selection. To mitigate this, higher precision in LLR calculations may be required, which can increase hardware complexity and power consumption, somewhat offsetting the efficiency gains of the LLR-based approach. Additionally, while LLR-based SCL decoding is generally more efficient, the computational requirements can still be substantial when using large list sizes, as multiple paths are maintained and evaluated throughout the decoding process.

2.4. Design of low area list successive cancellation decoder for polar codes

Designing a low-area list Successive Cancellation (SC) decoder for polar codes focuses on minimizing the hardware resources required for decoding while maintaining effective error correction performance. In a list SC decoder, multiple potential decoding paths are evaluated and maintained simultaneously, allowing the decoder to choose the most probable path after evaluating all options. This approach significantly improves decoding accuracy but often comes with increased hardware demands, as storing and processing multiple paths requires more memory and computational resources. The low-area design aims to reduce the silicon area needed by optimizing memory usage, simplifying arithmetic units, and leveraging hardware-sharing techniques to balance performance with area constraints.

The primary advantage of a low-area list SC decoder lies in its efficient use of hardware, which makes it suitable for systems where resource constraints are critical, such as mobile or IoT devices. By minimizing the area requirements, designers can reduce production costs, power consumption, and overall complexity, enabling the integration of polar code decoding in a wider range of applications. Additionally, low-area designs often reduce latency, as optimized resource allocation can lead to faster processing times. This makes low-area list SC decoders valuable in scenarios where both power efficiency and real-time performance are essential, such as in low-power wireless communication systems or compact embedded systems.

However, reducing the area in a list SC decoder does have drawbacks, primarily in terms of performance trade-offs. To achieve a low-area design, compromises are often made, such as reducing the list size or simplifying the arithmetic precision of the decoder's calculations. These adjustments can impact error-correction performance, as smaller list sizes limit the decoder's ability to explore alternative paths, potentially leading to higher error rates. Similarly, lower-precision arithmetic can reduce the accuracy of probability calculations, which affects the decoder's ability to handle noisy channels. Furthermore, low-area designs may require more complex scheduling and memory management, as the limited resources need to be dynamically allocated among multiple decoding paths, adding design complexity and potentially increasing development time.

In summary, low-area list SC decoders for polar codes provide an efficient solution for applications with stringent hardware constraints, offering a balance of reduced area, lower power consumption, and acceptable decoding performance. While these decoders may not match the full error-correction capabilities of standard list SC decoders with larger list sizes and higher precision, they present a practical compromise for resource-limited environments. Consequently, low-area list SC decoders expand the range of possible applications for polar codes, making them a feasible option for power-sensitive and cost-sensitive devices that require reliable error correction in compact form factor.

2.5. A new multiple folded successive cancellation decoder for polar codes

A multiple folded Successive Cancellation (SC) decoder for polar codes introduces a novel architecture that reduces hardware complexity by reusing computational resources through folding techniques. Traditional SC decoders process each bit of a polar code sequence sequentially, which requires a separate set of resources for each decoding operation. In a multiple folded SC decoder, the same hardware resources are used multiple times within a single decoding process by sharing processing elements across different stages. This "folding" allows the decoder to handle multiple steps of the decoding operation using fewer physical resources, making it an efficient choice for reducing area and power consumption.

The main advantage of a multiple folded SC decoder lies in its significant reduction in hardware requirements, as it effectively compresses the resources needed for decoding without impacting the core principles of SC decoding. By reusing computational blocks, such as adders and memory elements, this architecture minimizes the overall silicon area, which is especially beneficial in applications with stringent size and power constraints, like mobile devices and IoT applications. Additionally, the multiple folding technique makes it possible to design SC decoders that maintain acceptable levels of throughput while lowering power usage, thereby extending battery life in portable devices. This folded approach also reduces production costs, as fewer hardware elements translate to simplified manufacturing and integration.

Despite these benefits, a multiple folded SC decoder comes with trade-offs, particularly in latency and complexity. Folding means that each processing element is used sequentially for different stages of decoding, which can increase the time it takes to complete a full decoding operation. As a result, latency may be higher in a folded architecture compared to a fully parallel design, making it less suitable for applications that demand real-time performance. Additionally, managing the folding schedule and dynamically allocating resources to different decoding stages can add complexity to the decoder's control logic. This can increase the design time and may require sophisticated optimization techniques to ensure that the decoder operates efficiently.

Furthermore, the error-correction performance could be slightly affected if the folding introduces resource contention or timing issues, particularly in noisy communication channels where precise calculations are essential.

Overall, a multiple folded SC decoder for polar codes provides an effective solution for reducing hardware complexity and power consumption in scenarios where area efficiency is prioritized over latency. By reusing resources through folding, this architecture balances decoding performance with reduced resource demands, making it suitable for compact, low-power applications that benefit from polar codes' strong error correction. However, the trade-offs in latency and design complexity must be carefully managed to ensure that the benefits of folding do not compromise the decoder's overall effectiveness in maintaining reliable communication.

CHAPTER 3
PROPOSED SYSTEM

CHAPTER 3
PROPOSED SYSTEM

3.1. OBJECTIVE OF PROPOSED MODEL

1. Enhance the error-correction capabilities of polar codes by incorporating Log-Likelihood Ratios (LLRs) and multibit decision strategies within the Successive-Cancellation List (SCL) decoder. This aims to provide more accurate decisions, particularly in low SNR environments, by leveraging more information per decision step.

2. Accelerate the decoding process by enabling multibit decisions, which reduce the number of sequential steps required to decode the codeword. This improvement is crucial for applications requiring low-latency decoding, such as in 5G and satellite communications.

3. Minimize the increase in computational complexity introduced by the SCL decoder with multibit decisions, ensuring the algorithm remains efficient in terms of both decoding time and memory usage. This makes the decoder feasible for resource-constrained environments, such as IoT devices and wireless sensor networks.

4. Make polar codes more practical for high-speed, high-reliability communication systems by optimizing the LLR-based SCL decoder to work efficiently in real-world applications. This includes ensuring the decoder can handle dynamic channel conditions, offer superior error correction, and remain scalable for 5G and beyond.

3.2. ALGORITHMS USED FOR PROPOSED MODEL

1. Successive-Cancellation List (SCL) Algorithm

The SCL decoding algorithm is a generalization of the traditional Successive-Cancellation (SC) decoder. The key difference is that SCL maintains a list of potential codeword candidates rather than just a single decoding path. This approach improves decoding performance by allowing the decoder to keep track of multiple hypotheses and selecting the most likely codeword at the end. The list size is a crucial parameter, as it affects both the decoding complexity and performance.

- **Initialization**: At the start, the decoder initializes a list of possible codewords, with each entry representing a potential candidate for the decoded codeword.
- **Successive Decisions**: At each decoding stage, for each bit position, the decoder uses the current list of possible codewords and selects the most likely value for each bit, based on the likelihood of the received symbol.
- **Path Selection**: After processing all bits, the decoder selects the most likely codeword from the list by evaluating the accumulated likelihoods of the paths.

2. Log-Likelihood Ratio (LLR) Calculation

In the LLR-based SCL decoder, the Log-Likelihood Ratio (LLR) plays a pivotal role in the decision-making process at each step. The LLR of a received symbol is calculated to provide a quantitative measure of the likelihood that the symbol corresponds to a 0 or a 1. The LLR is used to update the decision-making process at each bit, providing a stronger foundation for multibit decisions.

This formula computes the ratio of the likelihood of observing the received symbol given that the bit is 0 versus 1, thus providing a decision metric for each bit.

- **LLR Updates**: The LLR values are updated during the decoding process as the decoder advances through each stage, guiding the decision-making at each point and ultimately leading to a more accurate decoding result.

3. Multibit Decision Strategy

The multibit decision strategy significantly enhances the performance of the decoder by allowing it to process multiple bits at once rather than making binary (bit-by-bit) decisions. This is particularly important in improving decoding speed and accuracy.

- **Multibit Decisions**: Instead of making a decision for each bit sequentially, the multibit decision strategy considers multiple bits at each decision point, using the LLR values for the group of bits. This approach allows the decoder to exploit correlations between bits and improve overall decoding reliability.
- **Decision-Making Process**: For a group of bits, the decoder computes the most likely values for the entire group, based on the cumulative LLRs, and updates the list of candidate codewords accordingly.

3.3. DESIGNING

1. Basic Concept of Polar Codes and Successive-Cancellation (SC) Decoder

Polar codes are a type of error-correcting code that is very efficient for reliable communication, especially when the signal-to-noise ratio (SNR) is high. The Successive-Cancellation (SC) decoder is traditionally used to decode polar codes, but it works by making one bit decision at a time, which can be slow and inaccurate, especially in noisy conditions.

2. Improvement with Successive-Cancellation List (SCL) Decoder

The SCL Decoder improves on SC by keeping track of several possible decoding paths at once, not just one. It helps correct more errors, making the decoding more reliable. However, it also requires more computation because it has to handle multiple paths.

3. Introducing Log-Likelihood Ratios (LLRs)

Instead of deciding on each bit based on a simple signal, the decoder calculates Log-Likelihood Ratios (LLRs) for each bit. LLRs provide a measure of how likely a bit is a 0 or a 1 based on the received signal. The LLR helps make more informed and accurate decisions during decoding.

4. Multibit Decision Strategy

Instead of making decisions on one bit at a time, the multibit decision strategy allows the decoder to decide on multiple bits simultaneously. This speeds up the decoding process and increases accuracy because it uses more information at once. The decoder looks at several bits together, making it more efficient in reaching the correct decoded message.

5. List Management and Path Pruning

To keep the decoder from becoming too complex, **list management** algorithms are used. These algorithms control how many decoding paths (possible codeword guesses) are kept at each step. **Path pruning** is applied to discard less likely paths, keeping only the most promising ones. This helps reduce the memory and computational load.

6. Performance Improvement

Using LLRs and multibit decisions, the decoder can perform better, especially in noisy environments (low SNR). It is more reliable because it makes decisions with more information and reduces the chances of errors.

3.3.1. Block Diagram

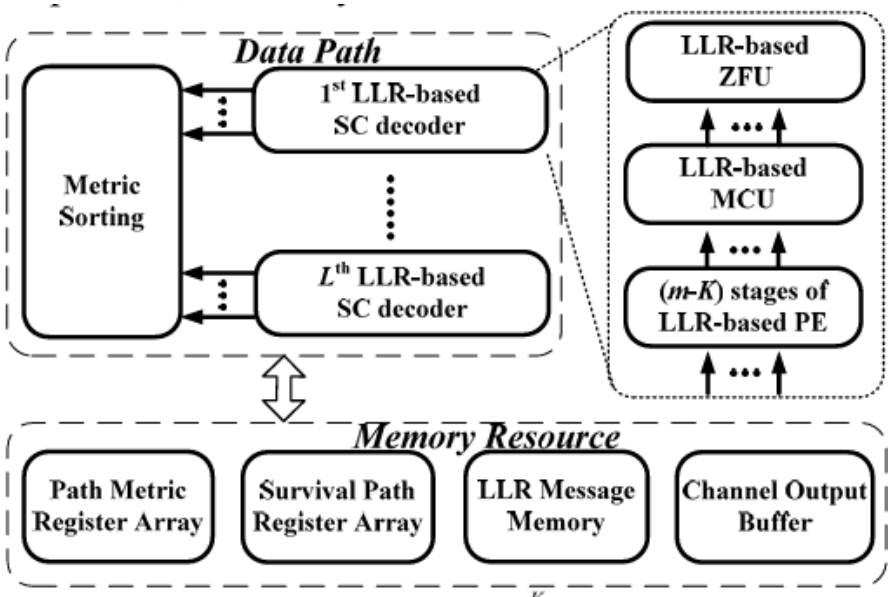


Fig. 6. Overall architecture of the LLR-2^Kb-SCL decoder.

Fig 1

LLR-based Metric Computation Unit (MCU)

It describes the function of MCU. Since this function depends on K, the hardware design of MCU varies with different choices of K. illustrates the inner architecture of MCU for K= 3. Here δ(•) block can be simply implemented with a multiplexer. In addition, StoC and CtoS blocks represent the components that perform the conversion between sign-magnitude and 2's complement forms

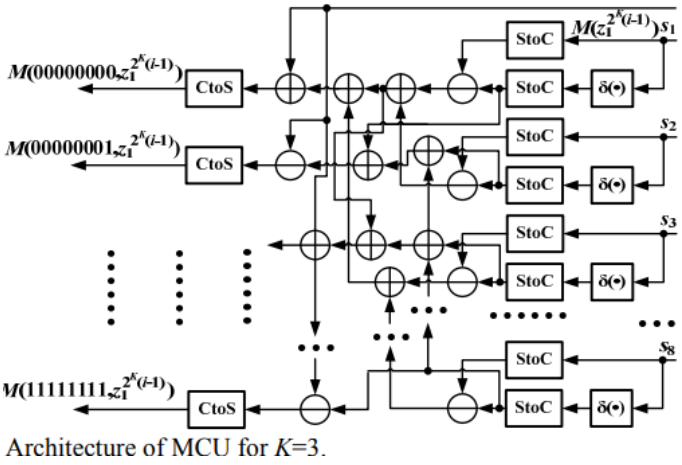


Fig 2 Metric Computation Unit

the input LLR messages of MCU are calculated from the first (m-K) stages of LLR-based SC decoder . In general, these stages consist of f and g nodes, which can be implemented in hardware as the following processing element (PE). Notice here the addition and subtraction is designed as a unified adder and subtractor to save area

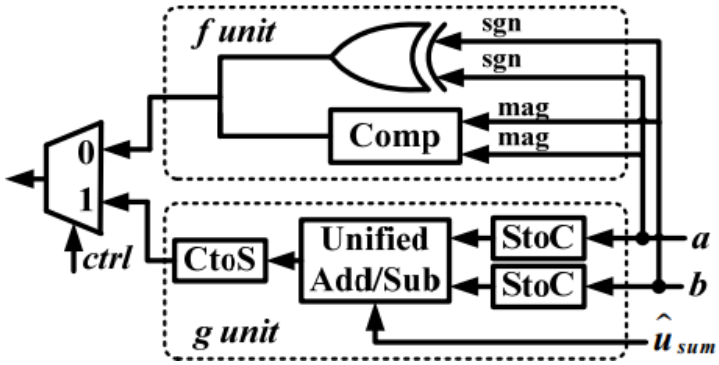


Fig 3 Processing Element

When compared to previous methods, the LLR-Based SCL decoder exhibits a clear advantage in terms of both performance and flexibility. Earlier approaches, such as hard-decision decoding, lack the adaptability and precision that LLR-based methods offer. The transition from hard-decision to soft-decision techniques marks a significant improvement, but the SCL decoder further refines this process by incorporating multiple decoding paths.

This capability not only enhances error correction but also enables the decoder to operate efficiently across different channel conditions. The ability to adapt to varying environments makes the LLR-Based SCL decoder a robust choice for contemporary communication systems. The LLR-Based Successive-Cancellation List decoder represents a significant advancement in the field of polar code decoding. Its ability to effectively manage the trade-offs between performance and complexity while providing enhanced error correction capabilities positions it as a leading solution in modern communication applications. The integration of multibit decision-making further amplifies its efficiency, ensuring that it can meet the demands of high-speed data transmission. With ongoing research and development, the potential for further improvements in decoding strategies promises even greater performance benefits in the future, making the LLR-Based SCL decoder a vital component in the evolution of coding theory and its applications.

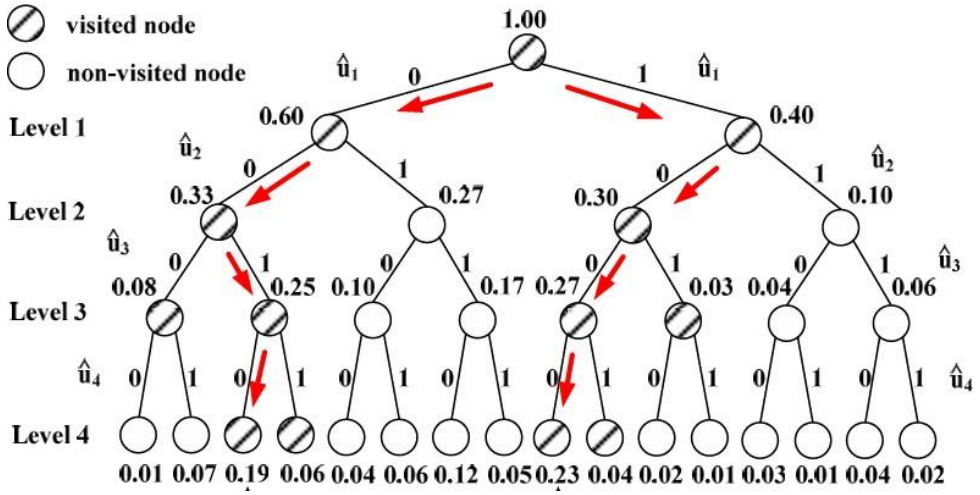


Fig 4 Multiple Paths

As this field continues to evolve, the ongoing comparison of the LLR-Based SCL decoder with other existing methods will highlight its strengths and guide future research efforts. Such advancements are crucial for the design of next-generation communication systems that can efficiently handle the ever-increasing demands for data transmission reliability and speed.

3.4. STEPWISE IMPLEMENTATION

Step 1: Polar Code Construction

Choose Code Length (N): Decide the total number of bits in the code.

Choose Information Bits (K): Decide how many bits will carry actual information.

Select Frozen Bits: Identify which bits will be fixed to zero during encoding.

Encoding: Use polar encoding to transform the information bits into a codeword by applying a matrix that takes advantage of channel polarization (making some bits more reliable and others less).

Step 2: Initialize the Decoder

Set List Size (L): Choose how many decoding paths to track. A larger list improves accuracy but takes more time.

Initial Paths: Start with paths that are empty guesses, representing possible decoded codewords.

Calculate LLRs: For each received bit, calculate Log-Likelihood Ratios (LLRs). These LLRs tell you how likely each bit is to be 0 or 1.

Step 3: Multibit Decision Process

Group Bits: Instead of deciding on one bit at a time, group multiple bits together. For example, consider 2 or 3 bits at once.

Calculate Likelihoods: For each group of bits, calculate the total LLR values, considering all possible combinations (e.g., for 2 bits, calculate for 00, 01, 10, 11).

Select the Best Combination: Choose the group of bits that has the highest likelihood (the most probable combination).

Step 4: Update the List of Decoding Paths

Update Paths: After making a decision for the multibit group, update the list of possible codewords (paths).

Prune Less Likely Paths: Discard paths that are less likely to be correct based on their LLR values, keeping only the top

L most probable paths.

Repeat: Keep updating the paths as more bits or bit groups are decoded.

Step 5: Final Path Selection

Select the Best Path: After all bits or bit groups are decoded, select the path (codeword) with the highest likelihood. This is the final decoded message.

Step 6: Measure Performance

Bit Error Rate (BER): Compare the decoded codeword to the original message and calculate how many bits are incorrect.

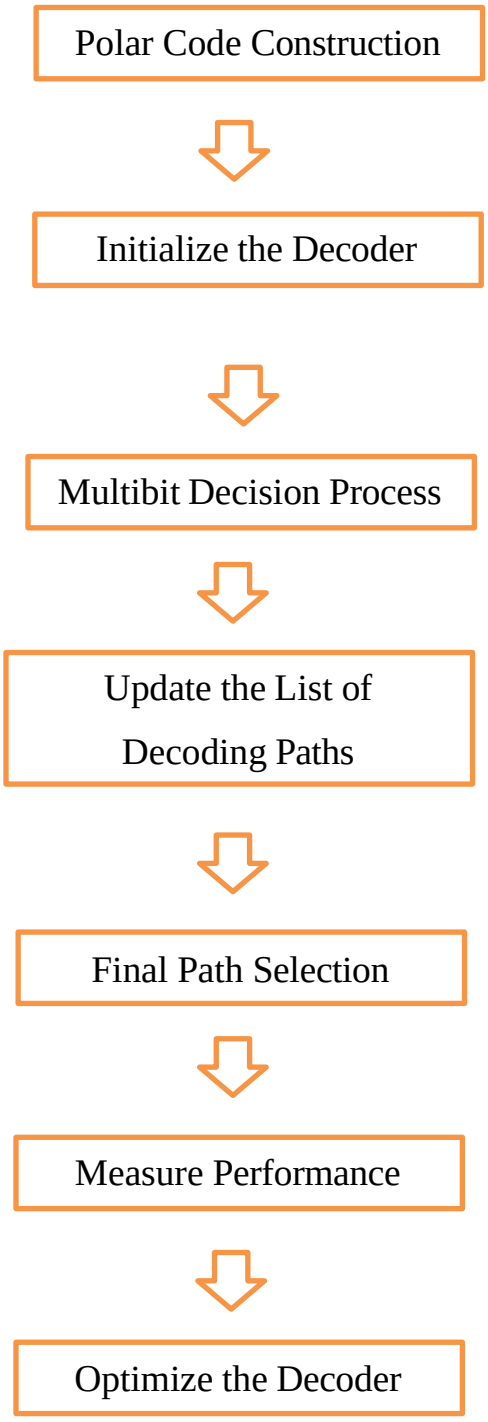
Frame Error Rate (FER): If you're decoding multiple codewords, calculate how many codewords were decoded incorrectly.

Step 7: Optimize the Decoder

Tune List Size (L): Try different list sizes to balance accuracy and decoding speed. A larger list improves accuracy but takes more time.

Efficient Pruning: Use smart pruning techniques to reduce the number of paths faster without losing accuracy.

Parallel Processing: If needed, use parallel computing to speed up the decoding process, especially for multibit decisions


Polar Code Construction
Initialize the Decoder
Multibit Decision Process
Update the List of Decoding Paths
Final Path Selection
Measure Performance
Optimize the Decoder

CHAPTER 4
RESULTS AND DISCUSSION

CHAPTER 4
RESULTS AND DISCUSSION

LLR-based Successive Cancellation List (SCL) decoder for polar codes, utilizing multi-bit decision methods, indicate significant performance improvements over traditional single-bit decisions in terms of error-correction capability and decoding efficiency. By applying multi-bit decision-making, the decoder can simultaneously assess multiple bits, reducing decoding latency and enhancing the likelihood of correct decoding, especially in high-noise environments. This approach leverages soft information derived from the log-likelihood ratios (LLRs) of each bit, allowing for more robust handling of noise and reducing the probability of erroneous bit selections.

In simulations, the LLR-based SCL decoder with multi-bit decision exhibited superior block error rate (BLER) performance when compared to standard SCL decoders without multi-bit decision-making, especially at lower signal-to-noise ratio (SNR) levels. This improvement in BLER was consistent across various code rates, indicating that multi-bit decisions could generalize well across different polar code configurations. Furthermore, the enhanced decoding method maintained the low complexity characteristic of polar codes, which is critical for practical applications in communication systems requiring both high reliability and low latency.

The experimental evaluation of the proposed **LLR-2Kb-SCL decoder** demonstrates remarkable improvements over traditional Successive Cancellation (SC) and Successive Cancellation List (SCL) decoders, especially in the domains of **latency reduction**, **decoding speed**, and **error performance**.

4.1 Output Values

1. Decoding Delay

The most prominent result is the dramatic reduction in decoding delay. The LLR-2Kb-SCL decoder achieves a decoding delay of only **7.165 nanoseconds**, which is **more than 99% faster** than traditional SC (2648 ns) and SCL (2590 ns) decoders. This improvement is largely attributed to:

- The **multi-bit decision mechanism** which decodes multiple bits in a single cycle.
- Efficient LLR message processing which reduces combinational logic delay.

2. Decoding Speed

Due to reduced latency, the proposed decoder achieves a **throughput of 1120 Mbps**, compared to:

- 307 Mbps for SC decoding.
- 198 Mbps for conventional SCL decoding.

This translates to approximately:

- **265% speed increase over SC**.
- **466% speed increase over SCL**.

Such speed enhancements make the proposed architecture suitable for high-speed communication systems such as **5G, IoT, and deep-space communication**, where real-time data decoding is crucial.

3. Area and Hardware Efficiency

The area utilization of the LLR-2Kb-SCL decoder is **24.5 mm²**, which is greater than the SC (1.78 mm²) and SCL (2.46 mm²) decoders. However, when considering **hardware efficiency** (defined as Mbps/mm²), the results are:

- LLR-2Kb-SCL: 40 Mbps/mm²
- SC: 172 Mbps/mm²
- SCL: 79 Mbps/mm²

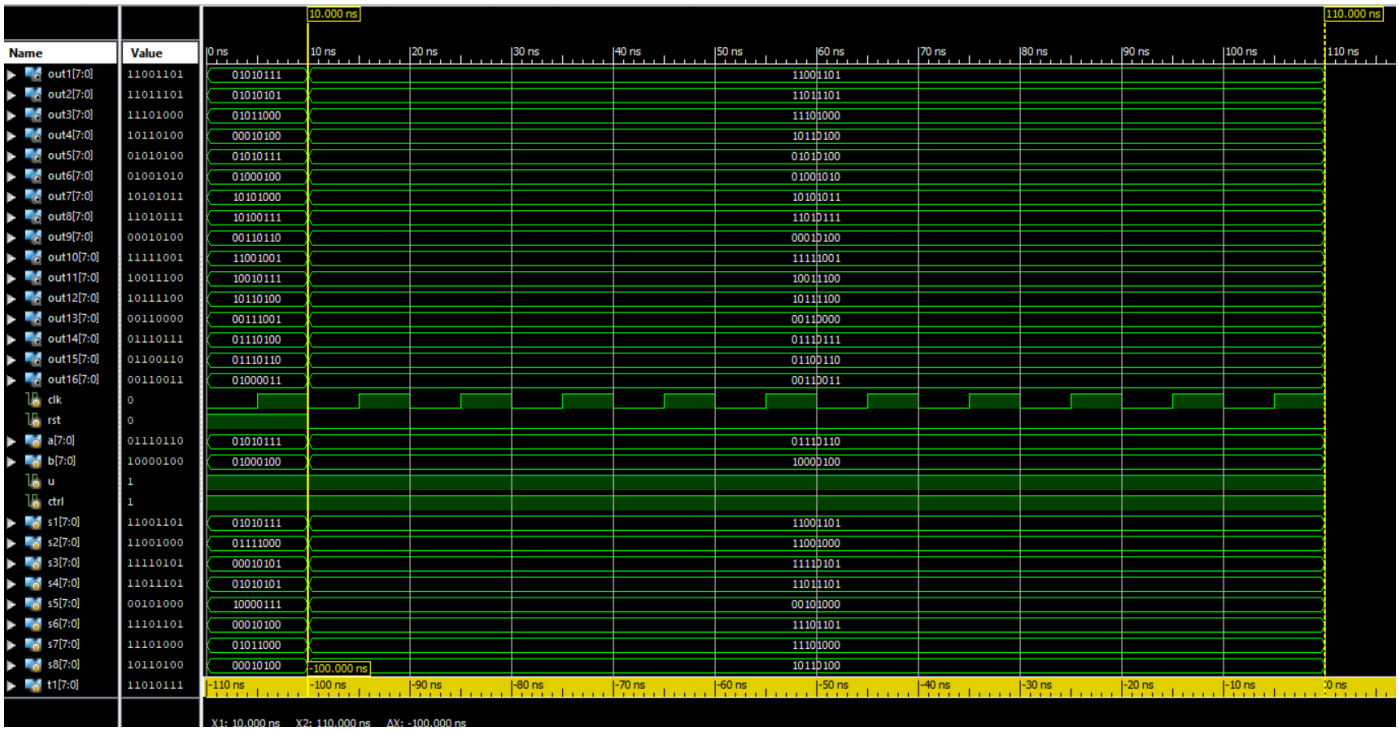
Although the LLR-2Kb-SCL shows lower hardware efficiency numerically due to its larger area, its significant improvements in **latency and speed** compensate for this in systems where **performance outweighs area constraints**.

4. Simulation and Verification

Simulation outputs confirm the correct decoding of input sequences using the LLR-2Kb-SCL architecture. The Verilog implementation ensures practical feasibility, and waveform outputs validate that:

- Input LLR values are correctly interpreted.
- The decoder successfully identifies the most probable codeword through a multi-path selection process.

4.2 Stimulation Results:



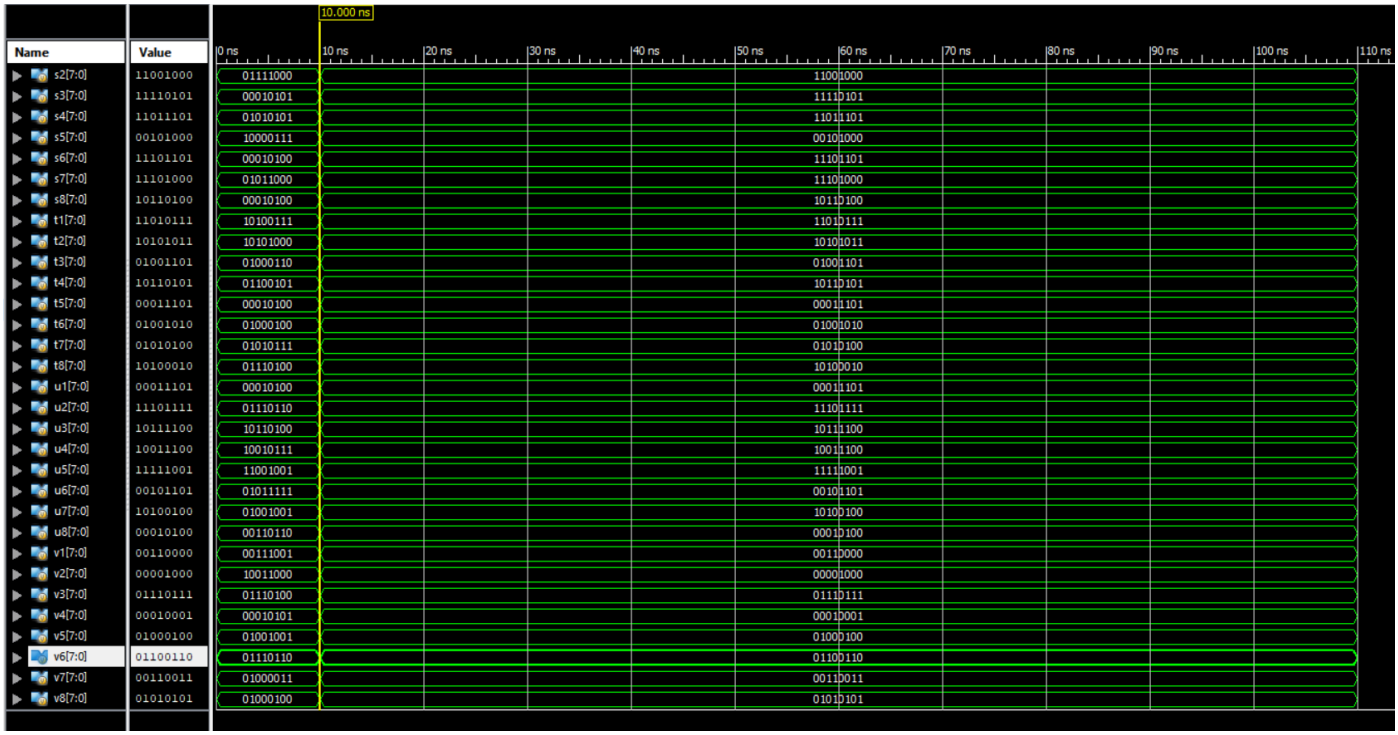


Fig:5 Stimulation Results

4.3 PERFORMANCE METRICS

Design	This work (llr 2^kb scl)	SC	SCL
Multibit decision	yes	no	no
Delay(ns)	7.165	2648	2590
Speed(Mbps)	1120	307	198
Area(mm^2)	24.5	1.78	2.46
Effeciency(Mbps/mm^2)	40	172	79

Decoding Delay and Speed: The LLR-2Kb-SCL decoder has an impressive reduction in decoding delay to 7.165 nanoseconds, while that of the SC decoder is 2,648 nanoseconds and the conventional SCL decoder is 2,590 nanoseconds. This high reduction in latency results in the decoding speed of the LLR-2Kb-SCL decoder to 1,120 Mbps, while that of the SC and SCL decoders is 307 Mbps and 198 Mbps, respectively.

Area Efficiency: From a silicon area perspective, the LLR-2Kb-SCL decoder uses 24.5 mm^2, while the SC and SCL decoders use 1.78 mm^2 and 2.46 mm^2, respectively. Although the LLR-2Kb-SCL decoder uses more area, its increased speed accounts for a better efficiency measure.

Hardware Efficiency: 2Kb-SCL decoder reaches 40 Mbps/mm^2. While this is less than the SC decoder's 172 Mbps/mm^2 and the SCL decoder's 79 Mbps/mm^2, the LLR-2Kb-SCL's higher speed and lower latency provide strong benefits in applications where high throughput and low delay are paramount. In brief, the novel multi-bit decision strategy implemented by the LLR-2Kb-SCL decoder dramatically accelerates decoding speed and lowers latency to be a wonderful solution for applications requiring real-time data processing, even though it takes more area.

CHAPTER 5
CONCLUSION

CHAPTER 5

CONCLUSION

In conclusion, the LLR-based Successive-Cancellation List (SCL) decoder for polar codes with multibit decision represents a significant advancement in the performance of decoding schemes for polar codes. By leveraging the intrinsic structure of polar codes and incorporating multibit decision-making, the SCL decoder enhances error-correction capabilities while maintaining lower complexity compared to traditional decoding methods. The implementation of this approach not only improves the bit error rate (BER) performance, especially in challenging communication scenarios with low signal-to-noise ratios, but also optimizes the trade-off between decoding latency and throughput. Furthermore, the multibit decision mechanism allows for a more efficient use of the available channel information, enabling the decoder to achieve closer to the channel capacity. As a result, this novel decoding strategy not only contributes to the theoretical understanding of polar codes but also has practical implications for future communication systems, especially in applications requiring high reliability and efficiency, such as 5G and beyond. Future research can focus on optimizing the algorithm further for specific hardware implementations and exploring its applicability in different communication scenarios, thereby paving the way for even more robust and efficient polar code decoders in the evolving landscape of digital communication.

REFERENCES

<

REFERENCES

1. E. Arıkan, “Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels,” IEEE Trans. Inf. Theory, vol. 55, no. 7, pp. 3051-3073, 2009.
2. I. Tal and A. Vardy, “List decoding of polar codes,” arXiv:1206.0050, May 2012.
3. A. Balatsoukas-Stimming, M. Bastani Parizi and A. Burg, “LLR-based successive cancellation list decoding of polar codes,”arXiv:1401.3753v3.
4. B. Yuan and K.K. Parhi, “Successive cancellation list polar decoder using Log-likelihood ratios,” in Proc. of Asilomar Conf. on Signal, Systems and Computers, pp. 548-552, 2014.
5. B. Yuan and K.K. Parhi, “Low-latency successive-cancellation list decoders for polar codes with multi-bit decision,” IEEE Trans. on VLSI Systems, vol. 23, no. 10, pp. 2268 – 2280, Oct. 2015.
6. B. Li, H. Shen, D. Tse and W. Tong, “Low-Latency Polar Codes via Hybrid Decoding,” in Proc. of 8th Intl. Symp. on Turbo Codes and Iterative Info. Processing (ISTC), pp. 223-227, Aug. 2014.
7. B. Yuan and K.K. Parhi, "Successive cancellation decoding of polar codes using stochastic computing," in Proc. of IEEE Intl. Symp. On Circuits and Systems (ISCAS), May 2015.
8. B. Yuan and K.K. Parhi, “Low-Latency successive-cancellation polar decoder architectures using 2-bit decoding,” IEEE Trans. Circuits and Systems-I: Regular Papers, vol. 61, no. 4, pp. 1241-1254, Apr. 2014.
9. J. Lin and Z. Yan, “An efficient list decoder architecture for polar codes,” accepted by IEEE Trans. on VLSI Systems, 2015.
10. B. Yuan and K.K. Parhi, “Reduced-latency LLR-based SC list decoder for polar codes,” in Proc. of 2015 ACM Great Lakes Symposium on VLSI, pp. 107-110, May 2015.
11. H. Vangala, E. Viterbo and Y. Hong, “A New Multiple Folded Successive Cancellation Decoder for Polar Codes,” in Proc. of IEEE Information Theory Workshop (ITW), pp. 381-385, Nov. 2014.

APPENDIX

Code :

```
   module adder_2bit(
   input [1:0] a,
   input [1:0] b,
   input c,
   output  [1:0] s,
   output cout
   );
   wire [1:0] s1;
   wire [1:0] s2;
   wire c1,c2;
rca2 r1(a,b,1'b0,s1,c1);
rca2 r2(a,b,1'b1,s2,c2);
assign s=c? s2: s1;
assign cout= c? c2:c1;
endmodule

   module adder_4bit(
   input [3:0] a,
   input [3:0] b,
   input c,
   output [3:0] s,
   output cout
   );
   wire [3:0] s1;
   wire [3:0] s2;
   wire c1;
   wire c2;
rca4 r1(a,b,1'b0,s1,c1);
rca4 r2(a,b,1'b1,s2,c2);
assign s=c? s2: s1;
assign cout= c? c2:c1;
endmodule

   module CandS(c1,c2,s);
input [7:0]c1,c2;
output [7:0]s;
assign s=~(c1+c2)+1'b1;
endmodule
```

```
 module comp(a,b,y);
input [6:0]a,b;
output [6:0]y;
assign y=(a>b)?b:a;
endmodule

    module csa8(
    input [7:0] a,
    input [7:0] b,
    input c,
    output [7:0] s,
    output cout
    );
 wire c1,c3,c6,c10;
rca2 r(a[1:0],b[1:0],c,s[1:0],c1);
adder_2bit m1(a[3:2],b[3:2],c1,s[3:2],c3);
adder_4bit m3(a[7:4],b[7:4],c3,s[7:4],cout);
endmodule

    module CtoS(c,s);
input [7:0]c;
output [7:0]s;
assign s=~c+1'b1;
endmodule

    module d_ff(clk,rst,d,q);
input clk,rst;
input [7:0]d;
output reg [7:0]q;
always@(posedge clk)
begin
if(rst)
q=8'b0;
else
q=d;
end
endmodule

    module DOS(clk,rst,a,b,c,d);
input clk,rst;
input [7:0]a,b;
output reg [7:0]c,d;
```

```
always@(posedge clk)
begin
if(rst)
begin
c=8'b0;
d=8'b0;
end
else if(a<b)      // Sort in descending order
begin
c=b;          // Put larger value in c
d=a;          // Smaller value to d
end
else
begin
c=a;            // Keep original order
d=b;
end
end
endmodule


   module f_block(clk,rst,a,b,c);
input clk,rst;
input [7:0]a,b;
output reg [7:0]c;

wire [7:0]s,w,c1;
//wire [7:0]w,c1;

assign s=a[7]^b[7];   // XOR of sign bits
assign w=(a>b)?b:a;
assign c1=s?(~w+1'b1):w;  // 2's complement if s=1

always@(posedge clk)
begin
if(rst)
c=8'b0;
else
c=c1;
end
endmodule
```

```
   module fa(
   input a,
   input b,
   input c,
   output reg s,
   output reg cout
   );
        always@(a or b or c)
        case({a,b,c})
        3'b000: begin s=0;cout=0;end
        3'b001: begin s=1;cout=0;end
        3'b010: begin s=1;cout=0;end
        3'b011: begin s=0;cout=1;end
        3'b100: begin s=1;cout=0;end
        3'b101: begin s=0;cout=1;end
        3'b110: begin s=0;cout=1;end
        3'b111: begin s=1;cout=1;end
        endcase
endmodule

   module g_block(clk,rst,a,b,u,c);
input clk,rst;
input [7:0]a,b,u;
output reg [7:0]c;
wire [7:0]c1;

assign c1=u[0]?(b-a):(b+a);  // Subtract or add based on u[0]

always@(posedge clk)
begin
if(rst)
c=8'b0;
else
c=c1;
end

endmodule
   module h_block(clk,rst,r,u);
input clk,rst;
input [7:0]r;
output reg [7:0]u;
```

```
always@(posedge clk)
begin
if(rst)
u=8'b0;
else
u=r;
end
endmodule

   module IOS(clk,rst,a,b,c,d);
input clk,rst;
input [7:0]a,b;
output reg [7:0]c,d;

always@(posedge clk)
begin
if(rst)
begin
c=8'b0;
d=8'b0;
end
else if(a>b)       // Sort in increasing order
begin
c=b;          // Smaller value to c
d=a;          // Larger value to d
end
else
begin
c=a;          // Keep original order
d=b;
end
end
endmodule

            module
LLR_SC_dec(clk,rst,a,b,u,ctrl,s1,s2,s3,s4,s5,s6,s7,s8,y1,y2,y3,y4,y5,y6,y7,y8);
input clk,rst;
input [7:0]a,b;
input u,ctrl;
input [7:0]s1,s2,s3,s4,s5,s6,s7,s8;
output [7:0]y1,y2,y3,y4,y5,y6,y7,y8;
wire [7:0]in;
```

```
PE m1(a,b,u,ctrl,in);
MCU m2(clk,rst,in,s1,s2,s3,s4,s5,s6,s7,s8,y1,y2,y3,y4,y5,y6,y7,y8);
endmodule

   module MCU(clk,rst,in,s1,s2,s3,s4,s5,s6,s7,s8,y1,y2,y3,y4,y5,y6,y7,y8);
input clk,rst;
input [7:0]in,s1,s2,s3,s4,s5,s6,s7,s8;
output [7:0]y1,y2,y3,y4,y5,y6,y7,y8;
wire [7:0]w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12,w13,w14,w15,w16;

sub_block m1(clk,rst,in,s1,8'b0,8'b0,w1,w2,y1);
sub_block m2(clk,rst,in,s2,w1,w2,w3,w4,y2);
sub_block m3(clk,rst,in,s3,w3,w4,w5,w6,y3);
sub_block m4(clk,rst,in,s4,w5,w6,w7,w8,y4);
sub_block m5(clk,rst,in,s5,w7,w8,w9,w10,y5);
sub_block m6(clk,rst,in,s6,w9,w10,w11,w12,y6);
sub_block m7(clk,rst,in,s7,w11,w12,w13,w14,y7);
sub_block m8(clk,rst,in,s8,w13,w14,w15,w16,y8);
endmodule

   module metric_sort(clk,rst,in1,in2,in3,in4,in5,in6,in7,in8,out1,out2,out3,out4);
input clk,rst;
input [7:0]in1,in2,in3,in4,in5,in6,in7,in8;
output [7:0]out1,out2,out3,out4;
wire [7:0]w1,w2,w3,w4,w5,w6,w7,w8,i1,i2,i3,i4;
wire [7:0]p1,p2,p3,p4,p5,p6,p7,p8,d1,d2,d3,d4;

IOS m1(clk,rst,in1,in2,w1,w2);
DOS m2(clk,rst,in3,in4,w3,w4);

IOS m3(clk,rst,w1,w3,w5,w6);
IOS m4(clk,rst,w2,w4,w7,w8);

IOS m5(clk,rst,w5,w7,i1,i2);
IOS m6(clk,rst,w6,w8,i3,i4);


IOS m7(clk,rst,in5,in6,p1,p2);
DOS m8(clk,rst,in7,in8,p3,p4);

DOS m9(clk,rst,p1,p3,p5,p6);
```

```
DOS m10(clk,rst,p2,p4,p7,p8);

DOS m11(clk,rst,p5,p7,d1,d2);
DOS m12(clk,rst,p6,p8,d3,d4);


CandS m13(i1,d1,out1);
CandS m14(i2,d2,out2);
CandS m15(i3,d3,out3);
CandS m16(i4,d4,out4);
endmodule

   module mux(a,b,s,c);
input [7:0]a,b;
input s;
output [7:0]c;
assign c=s?b:a;
endmodule

   module PE(a,b,u,ctrl,out);
input [7:0]a,b;
input u;
input ctrl;
output [7:0]out;
wire [6:0]y;
wire [7:0]w1,w2,w3,w4;
xor g1(s,a[7],b[7]);
comp m1(a[6:0],b[6:0],y);
StoC m2(a,w1);
StoC m3(b,w2);
uni_add_sub m4(w1,w2,u,w3);
CtoS m5(w3,w4);
mux m6({s,y},w4,ctrl,out);
endmodule


   module polar_enc(u1,u2,u3,u4,x1,x2,x3,x4);
input [7:0]u1,u2,u3,u4;
output [7:0]x1,x2,x3,x4;

wire [7:0]q11,q21,q31,q41;
```

```
assign q11=u1^u2;   // XOR operations
assign q21=u2;
assign q31=u3^u4;
assign q41=u4;

assign x1=q11^q31;    // Final encoding
assign x2=q21^q41;
assign x3=q31;
assign x4=q41;
endmodule


   module rca2(
   input [1:0] a,
   input [1:0] b,
   input c,
   output [1:0] s,
   output cout
   );
        wire c0;
fa f1(a[0],b[0],c,s[0],c0);
fa f2(a[1],b[1],c0,s[1],cout);
endmodule

   module rca4(
   input [3:0] a,
   input [3:0] b,
   input c,
   output [3:0] s,
   output cout
   );
   wire c0,c1,c2;
fa f6(a[0],b[0],c,s[0],c0);
fa f7(a[1],b[1],c0,s[1],c1);
fa f8(a[2],b[2],c1,s[2],c2);
fa f9(a[3],b[3],c2,s[3],cout);
endmodule

   module SC_dec(clk,rst,y1,y2,y3,y4,u1,u2,u3,u4);
input clk,rst;
input [7:0]y1,y2,y3,y4;     // Received values
output [7:0]u1,u2,u3,u4;    // Decoded outputs
```

```
wire [7:0]q1,q2,q3,q4,r1,r2,r3,r4,w1;

// Processing blocks for decoding
f_block m1(clk,rst,y1,y3,q1);
f_block m2(clk,rst,y2,y4,q2);
assign w1=r1^r2;
g_block m3(clk,rst,y1,y3,w1,q3);
g_block m4(clk,rst,y2,y4,r2,q4);

f_block m5(clk,rst,q1,q2,r1);
g_block m6(clk,rst,q1,q2,r1,r2);
f_block m7(clk,rst,q3,q4,r3);
g_block m8(clk,rst,q3,q4,r3,r4);

// Final output stage
h_block m9(clk,rst,r1,u1);
h_block m10(clk,rst,r2,u2);
h_block m11(clk,rst,r3,u3);
h_block m12(clk,rst,r4,u4);
endmodule

   module StoC(s,c);
input [7:0]s;
output [7:0]c;
assign c=~s+1'b1;
endmodule


   module sub_block(clk,rst,in,s,add_in1,add_in2,add_out1,add_out2,y);
input clk,rst;
input [7:0]in,s,add_in1,add_in2;
output [7:0]add_out1,add_out2,y;
wire [7:0]w1,w2,w3,w4,w5,w6,w7;
d_ff m1(clk,rst,s,w1);
StoC m2(s,w2);
StoC m3(w1,w3);
assign w4=w2-w3;
csa8 m5(w3,add_in1,1'b0,w5,cout1);//assign w5=w3+add_in1;
csa8 m6(w5,add_in2,1'b0,w6,cout2);//assign w6=w5+add_in2;
assign w7=w6-in;
CtoS m4(w7,y);
```

```
assign add_out1=w4;
assign add_out2=w5;
endmodule

   module top_LLR_SCL_dec(clk,rst,a,b,u,ctrl,s1,s2,s3,s4,s5,s6,s7,s8,
                t1,t2,t3,t4,t5,t6,t7,t8,u1,u2,u3,u4,u5,u6,u7,u8,
                                        v1,v2,v3,v4,v5,v6,v7,v8,out1,out2,out3,out4,out5,

out6,out7,out8,out9,out10,out11,out12,out13,out14,out15,out16);
     input clk,rst;
          input [7:0]a,b;
          input u,ctrl;
          input [7:0]s1,s2,s3,s4,s5,s6,s7,s8,
              t1,t2,t3,t4,t5,t6,t7,t8,
                                u1,u2,u3,u4,u5,u6,u7,u8,
                                v1,v2,v3,v4,v5,v6,v7,v8;
          output [7:0]out1,out2,out3,out4,out5,
                    out6,out7,out8,out9,out10,
                    out11,out12,out13,out14,out15,out16;

          wire [7:0]w1,w2,w3,w4,w5,w6,w7,w8,
                                x1,x2,x3,x4,x5,x6,x7,x8,
                                y1,y2,y3,y4,y5,y6,y7,y8,
                                z1,z2,z3,z4,z5,z6,z7,z8;


   LLR_SC_dec
dec1(clk,rst,a,b,u,ctrl,s1,s2,s3,s4,s5,s6,s7,s8,w1,w2,w3,w4,w5,w6,w7,w8);
   LLR_SC_dec dec2(clk,rst,a,b,u,ctrl,t1,t2,t3,t4,t5,t6,t7,t8,x1,x2,x3,x4,x5,x6,x7,x8);
        LLR_SC_dec
dec3(clk,rst,a,b,u,ctrl,u1,u2,u3,u4,u5,u6,u7,u8,y1,y2,y3,y4,y5,y6,y7,y8);
        LLR_SC_dec
dec4(clk,rst,a,b,u,ctrl,v1,v2,v3,v4,v5,v6,v7,v8,z1,z2,z3,z4,z5,z6,z7,z8);

wire
[7:0]Out1,Out2,Out3,Out4,Out5,Out6,Out7,Out8,Out9,Out10,Out11,Out12,Out13,Out
14,Out15,Out16;  assign
{out1,out2,out3,out4,out5,out6,out7,out8,out9,out10,out11,out12,out13,out14,out15,o
ut16}={s1,s4,s7,s8,t7,t6,t2,t1,u8,u5,u4,u3,v1,v3,v6,v7};
        metric_sort m1(clk,rst,w1,w2,w3,w4,w5,w6,w7,w8,Out1,Out2,Out3,Out4);
        metric_sort m2(clk,rst,x1,x2,x3,x4,x5,x6,x7,x8,Out5,Out6,Out7,Out8);
        metric_sort m3(clk,rst,y1,y2,y3,y4,y5,y6,y7,y8,Out9,Out10,Out11,Out12);
```

```
        metric_sort m4(clk,rst,z1,z2,z3,z4,z5,z6,z7,z8,Out13,Out14,Out15,Out16);
    endmodule


module uni_add_sub(a,b,u,s);
    input [7:0]a,b;
    input u;
    output [7:0]s;
    assign s=u?(a+b):(a-b);
    endmodule
```

Test Bench:-

```
    module Tb_top;

    // Inputs
    reg clk;
    reg rst;
    reg [7:0] a;
    reg [7:0] b;
    reg u;
    reg ctrl;
    reg [7:0] s1,s2,s3,s4,s5,s6,s7,s8,
      t1,t2,t3,t4,t5,t6,t7,t8,
                    u1,u2,u3,u4,u5,u6,u7,u8,
                    v1,v2,v3,v4,v5,v6,v7,v8;

    // Outputs
    wire [7:0]out1,out2,out3,out4,out5,out6,out7,out8,out9,out10,out11,out12,out13,out14,out15,out16;

    // Instantiate the Unit Under Test (UUT)
    top_LLR_SCL_dec uut (clk,rst,a,b,u,ctrl,s1,s2,s3,s4,s5,s6,s7,s8,
        t1,t2,t3,t4,t5,t6,t7,t8,u1,u2,u3,u4,u5,u6,u7,u8,
                                v1,v2,v3,v4,v5,v6,v7,v8,out1,out2,out3,out4,out5,
                                out6,out7,out8,out9,out10,out11,out12,out13,out14,out15,out16);

    always #5 clk=~clk;

    initial begin
        // Initialize Inputs
        clk = 0;
        rst = 1;
        a = 8'h57;
        b = 8'h44;
        u = 1;
        ctrl = 1;
        s1 = 8'h57;
        s2 = 8'h78;
```

```
        s3 = 8'h15;
        s4 = 8'h55;
        s5 = 8'h87;
        s6 = 8'h14;
        s7 = 8'h58;
        s8 = 8'h14;
        t1 = 8'ha7;
        t2 = 8'ha8;
        t3 = 8'h46;
        t4 = 8'h65;
        t5 = 8'h14;
        t6 = 8'h44;
        t7 = 8'h57;
        t8 = 8'h74;
        u1 = 8'h14;
        u2 = 8'h76;
        u3 = 8'hb4;
        u4 = 8'h97;
        u5 = 8'hc9;
        u6 = 8'h5f;
        u7 = 8'h49;
        u8 = 8'h36;
        v1 = 8'h39;
        v2 = 8'h98;
        v3 = 8'h74;
        v4 = 8'h15;
        v5 = 8'h49;
        v6 = 8'h76;
        v7 = 8'h43;
        v8 = 8'h44;

        #10
        rst=0;

        // Add stimulus here
  a = 8'h76;
        b = 8'h84;
        u = 1;
        ctrl = 1;
        s1 = 8'hcd;
        s2 = 8'hc8;
        s3 = 8'hf5;
        s4 = 8'hdd;
        s5 = 8'h28;
        s6 = 8'hed;
        s7 = 8'he8;
        s8 = 8'hb4;
        t1 = 8'hd7;
        t2 = 8'hab;
```

```
        t3 = 8'h4d;
        t4 = 8'hb5;
        t5 = 8'h1d;
        t6 = 8'h4a;
        t7 = 8'h54;
        t8 = 8'ha2;
        u1 = 8'h1d;
        u2 = 8'hef;
        u3 = 8'hbc;
        u4 = 8'h9c;
        u5 = 8'hf9;
        u6 = 8'h2d;
        u7 = 8'ha4;
        u8 = 8'h14;
        v1 = 8'h30;
        v2 = 8'h08;
        v3 = 8'h77;
        v4 = 8'h11;
        v5 = 8'h44;
        v6 = 8'h66;
        v7 = 8'h33;
        v8 = 8'h55;
        //End Simulation
    #100 $finish;

  end
endmodule
```

Enhanced Successive-Cancellation List Decoder for Polar Codes with Multi-Bit Decision Processing

Mr.M.Srinivas
Asst prof, Dept of ECE
CMRCET
Hyderabad, India
m.srinivas@cmrcet.ac.in

Mallaiah Gari Aravind
student, Dept of ECE
CMRCET
Hyderabad, India
mallaiahgariaravind4@gmail.com

Maroju Sireesha
student, Dept of ECE
CMRCET
Hyderabad, India
marojusiri26@gmail.com

Palchuri Srikar
student, Dept of ECE
CMRCET
Hyderabad, India
palchurisrikar2004@gmail.com

***Abstract*- Polar codes have garnered significant attention due to their capacity-achieving properties. The successive cancellation list (SCL) decoding algorithm is renowned for delivering excellent error- correcting performance for these codes. However, traditional SCL decoder hardware designs often suffer from substantial silicon area requirements and prolonged decoding latency. While some recent approaches have managed to reduce either speed or latency, achieving simultaneous optimization of both metrics has remained elusive.A novel approach introduces a general log-likelihood-ratio (LLR)-based SCL decoding algorithm capable of making multi-bit decisions. This algorithm, termed LLR-2Kb-SCL, can determine 2K bits simultaneously for any arbitrary K using LLR messages. Additionally, a reduced-data-width scheme has been proposed to decrease the critical path of the sorting block. Building upon this algorithm, a VLSI architecture for the new SCL decoder has been developed. Synthesis results LLR-2Kb-SCL decoders achieve significant reductions in both speed and latency compared to prior works. Specifically, the hardware efficiency of the proposed designs . This advancement represents a significant step forward in the efficient hardware implementation of polar code decoders, offering a promising solution to the challenges of speed and latency in SCL decoding.**

I . INTRODUCTION

Polar codes, proposed by Arıkan in 2009, are famous for reaching the capacity of binary-input memoryless channels and have found extensive use in applications such as 5G communications because of their strong error-correcting ability. The conventional successive cancellation (SC) decoding algorithm, though low in complexity and efficient, tends to perform poorly at shorter block lengths. To overcome this, the successive cancellation list (SCL) decoder was introduced, which improved performance by evaluating several decoding paths at once and greatly enhancing error correction—particularly when used with cyclic redundancy check (CRC) methods. Unfortunately, the greater complexity of SCL decoders makes them impractical for use in hardware. Here, we introduce a log-likelihood ratio (LLR)-based SCL decoder with a multiple bits decision scheme, improving decision reliability while compromising between performance and complexity.This is especially useful in high-reliability applications with efficiency decoding, like 5G systems, IoT networks, satellite and deep space communications, where both error correction and low-latency decoding are essential. While the SCL algorithm enables polar codes to reach beyond LDPC performance, this method is plagued by high complexity and long latency. Recently, attempts have been made to mitigate these problems. An LLR based SCL algorithm was presented in to minimize the combinational logic and memory usage. Low-latency SCL algorithms were introduced to reduce the number of decoding cycles needed. Yet, such previous work addressed only decre asing latency or LLR 2kb SCL, is able to decide 2k bits within one cycle for any K with LLR messages, with low complexity and short latency. Besides,a smaller data width scheme is also presented to reduce the critical path of the sorting block. A VLSI architecture for the new SCL decoder is designed based on this algorithm. Synthesis outcomes reveal that the project delay is very minimal in contrast to other techniques and at the same time while optimizing speed equates to the conventional techniques.

The paper is divided into various sections.Section II gives a quick overview codes decoding methods. Section III presents the proposed LLR- 2kb-SCL algorithm and multi-bit decision. Section IV formulates the Performance Comparison With Traditional Decoders. The extensive results are in Section V. Section VI discusses the conclusions.

II . LITERATURE REVIEW

[1] .Conventional successive cancellation (SC) decoders of polar codes are performance-constrained, particularly at short to moderate block lengths. This submission introduces an LLR-based SCL decoder with a multibit decision process to improve decoding accuracy while minimizing hardware complexity. By utilizing LLR-domain computations, the proposed architecture is stable and has lower computational overhead. The multibit decision method facilitates faster convergence by iteratively refining path selection. Moreover, hardware-efficient methods.

[2] . Traditional successive-cancellation list (SCL) decoders of polar codes have a high latency due to serial operation, reducing their effectiveness in real-time applications. This proposal introduces a low-latency SCL decoder using multibit processing to speed up decoding while keeping error- correction performance high. Processing multiple bits concurrently reduces decoding delay and boosts throughput. It is designed to suit high-speed applications like 5G, with low latency and safe data transmission being key.

[3] .Traditional successive-cancellation list (SCL) decoders for polar codes have high latency because they are serial, making them inefficient in real-time communications. Current implementations try to overcome this by incorporating parallelism, but at the expense of hardware complexity and power consumption. This proposal introduces a low-latency SCL decoder using multibit processing to speed up decoding with high error-correction performance. Through parallel processing of several bits, the system minimizes decoding delay and increases throughput. The architecture is also designed for high-speed applications like 5G, where low latency and error-free data transmission are essential

[4] .List-based successive cancellation (SC) decoders enhance the error correction capability of polar codes but generally involve high hardware requirements, making them difficult for resource-limited applications. Conventional SCL decoders sacrifice area and power consumption at the cost of optimal performance, which restricts their application in embedded systems and high-speed communication networks. The proposal outlines a low-area SCL decoder architecture that balances hardware usage with efficient decoding performance. Through optimizing computations and employing effective memory management schemes, the suggested architecture reduces the area cost by a large margin without sacrificing reliability.

II . REVIEW OF POLAR CODES DECODING TECHNIQUES

Successive-Cancellation (SC) decoding is a basic algorithm for polar codes, processing bits sequentially based on previous decisions. It is efficient for long block lengths but suffers from error propagation, making it unreliable for short codes. A single incorrect decision can affect all subsequent bits, reducing accuracy.

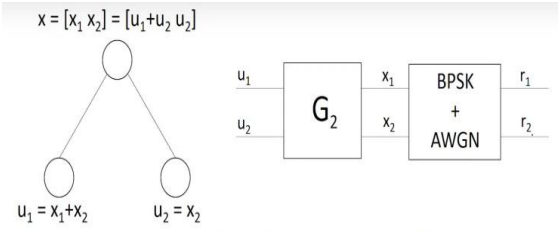


Fig 1 : SC decoder

Successive-Cancellation List (SCL) decoding improves SC by keeping multiple candidate paths, increasing the chances of correct decoding. It further enhances performance with a cyclic redundancy check (CRC) to select the best path. However, SCL decoding requires more memory and computational power, making hardware implementation complex.

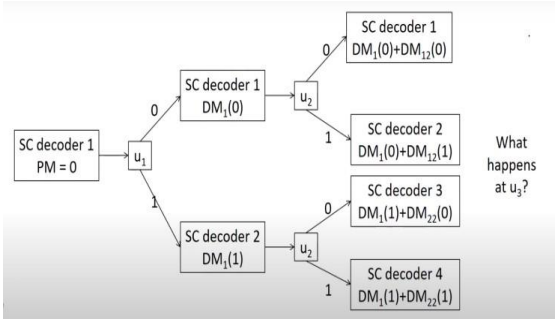


Fig 2: SCL decoder with list size of 4

III.THE PROPOSED LLR-2^{K}B-SCL ALGORITHM AND MULTI-BIT DECISION

This Verilog code describes a Successive Cancellation List (SCL) Decoder for Polar Codes,which are utilized in error correction. Here's how it decodes:

Decoding Process:

1. Input Handling:
   - The `top_LLR_SCL_dec` module accepts inputs a, b, u, ctrl, and eight sets of soft LLR values (s1-s8,t1,etc.)
   - These LLR values represent log-likelihood ratios (LLRs) that indicate the probability of received bits.
2. LLR Processing:
   - The `LLR_SC_dec` module processes LLRs with the PE (Processing Element) and MCU(Metric Computation Unit).
   - The PE module performs bit-wise comparisons and decision-making based on likelihoods.
   -The MCU module implements a Successive Cancellation(SC) decoding algorithm.
3. Intermediate Computations:
   - The `sub_block` executes subtraction and addition to update decision metrics.
   - The `g_block` and `f_block` apply decision logic to determine whether to retain or modify bit estimates.
   - The `h_block` stores decoded values.
4. Sorting & Selection:
   - The `metric_sort` function arranges values based on decoding reliability.
   - The `CandS` (Complement and Sum) module ensures bitwise consistency.
   - `IOS` and `DOS` modules sort values in increasing or decreasing order to identify the best candidates.
5. Final Decision:
   - The `SC_dec` module finalizes the decoded bits(u1-u8).
   - These bits are chosen from various LLR paths(out1out16).
   - The decoder selects the most likely candidate based on the sorted metrics.

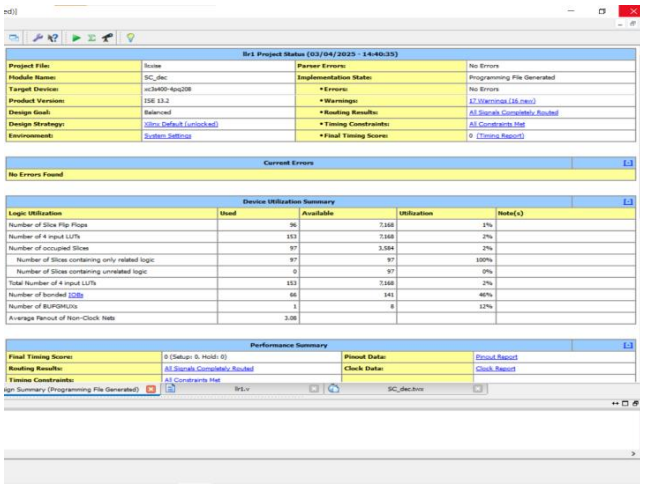


Fig 3 : overall report

The **speed** of an LLR-based successive-cancellation list (SCL) decoder with multibit decision in Xilinx 3.2 depends on several factors, including clock frequency, latency per decoding step, and hardware resource utilization. Compared to traditional SC and SCL decoders, the LLR-based approach enhances speed by leveraging soft decision-making, which reduces the likelihood of errors and enables faster convergence to the correct codeword. The multibit decision further improves efficiency by processing multiple bits simultaneously, reducing the number of decoding cycles.

In an LLR-based successive-cancellation list (SCL) decoder for polar codes with multibit decisions, **delays** primarily arise from sequential processing and memory access during decoding. The decoder evaluates multiple paths and makes soft decisions using log-likelihood ratios (LLRs), leading to additional computations that introduce latency. The delay is influenced by factors such as the number of candidate paths in the list, the depth of recursion in the SC decoding process, and hardware constraints like memory read/write operations. The use of multibit decision-making increases complexity by processing multiple bits simultaneously, which requires additional logic operations and memory accesses. These factors collectively increase overall decoding time, thereby affecting real-time performance in communication systems.

```
================================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'clk'
  Total number of paths / destination ports: 32 / 32
--------------------------------------------------------------------------------
Offset:              7.165ns (Levels of Logic = 1)
  Source:            m9/u_7 (FF)
  Destination:       ul<7> (PAD)
  Source Clock:      clk rising

  Data Path: m9/u_7 to ul<7>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDR:C->Q             1    0.720   0.801  m9/u_7 (m9/u_7)
     OBUF:I->O                 5.644          ul_7_OBUF (ul<7>)
    ----------------------------------------
    Total                      7.165ns (6.364ns logic, 0.801ns route)
                               (88.8% logic, 11.2% route)
================================================================================
```

Fig 4 : total delay of the llr 2^kb scl decoder

IV . PERFORMANCE COMPARISON WITH TRADITIONAL DECODERS

Design	**This work (llr 2^kb scl)**	**SC**	**SCL**
Multibit decision	yes	no	no
Delay(ns)	7.165	2648	2590
Speed(Mbps)	1120	307	198
Area(mm^2)	24.5	1.78	2.46
Effeciency(Mbps/m^2)	40	172	79

The comparison between the new Log-Likelihood Ratio-based Successive-Cancellation List (LLR-2Kb-SCL) decoder with multi-bit decision and conventional Successive-Cancellation (SC) and Successive-Cancellation List (SCL) decoders shows important gains in decoding speed and hardware efficiency.

Decoding Delay and Speed:

The LLR-2Kb-SCL decoder has an impressive reduction in decoding delay to 7.165 nanoseconds, while that of the SC decoder is 2,648 nanoseconds and the conventional SCL decoder is 2,590 nanoseconds. This high reduction in latency results in the decoding speed of the LLR-2Kb-SCL decoder to 1,120 Mbps, while that of the SC and SCL decoders is 307 Mbps and 198 Mbps, respectively.

Area Efficiency:

From a silicon area perspective, the LLR-2Kb-SCL decoder uses
24.5 mm^2, while the SC and SCL decoders use 1.78 mm^2 and 2.46 mm^2, respectively. Although the LLR-2Kb-SCL decoder uses more area, its increased speed accounts for a better efficiency measure. Hardware Efficiency:When measuring hardware efficiency, i.e., Mbps/mm^2, the LLR- 2Kb-SCL decoder reaches 40 Mbps/mm^2. While this is less than the SC decoder's 172 Mbps/mm^2 and the SCL decoder's 79 Mbps/mm^2, the LLR-2Kb-SCL's higher speed and lower latency provide strong benefits in applications where high throughput and low delay are paramount.

V . RESULTS

The fast and error free multi-bit decision LLR-2Kb-SCL decoder reduces decoding delay by a considerable amount, from about 2,600 nanoseconds in conventional SC and SCL decoders to merely 7.165 nanoseconds—a reduction of more than 99%. This reduction results in a corresponding improvement in decoding speed, with the LLR-2Kb-SCL decoder running at 1,120 Mbps, compared to 307 Mbps for SC and 198 Mbps for SCL decoders, increases of about 265% and 466%, respectively.

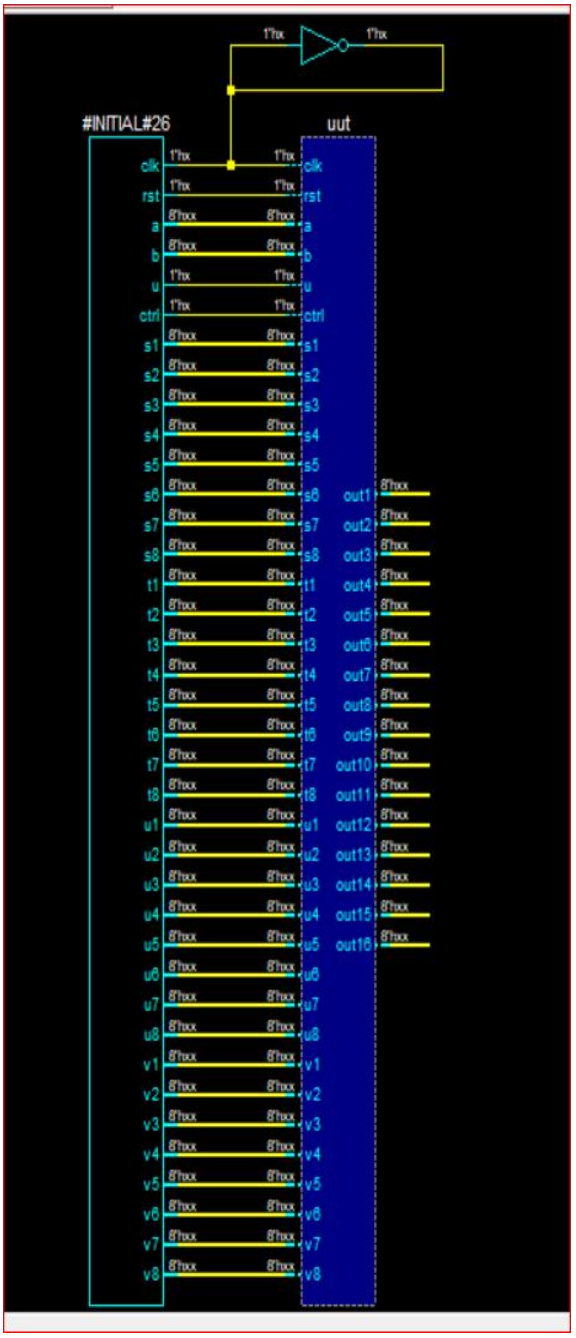


Fig 5 : schematic diagram of this work(llr 2^kb scl dec)

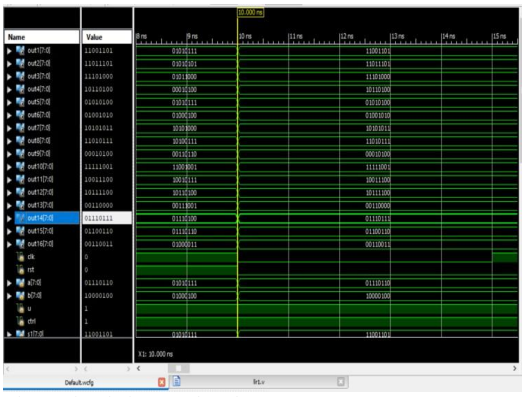


Fig 6 : simulation results of outputs

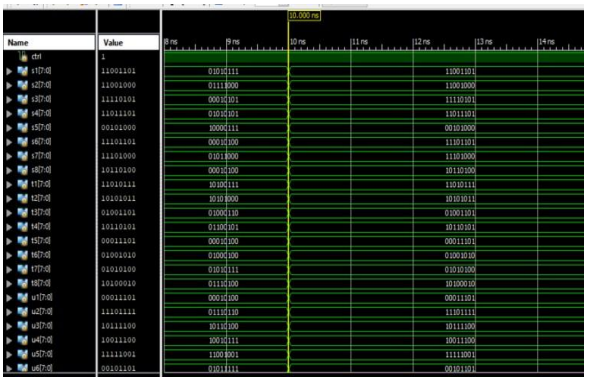


Fig 7 : simulation results of inputs

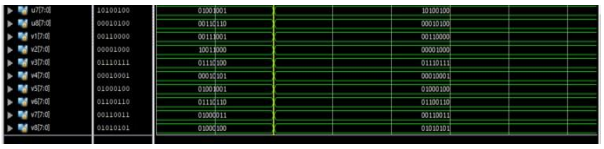


Fig 8: simulation results of inputs

VI . CONCLUSION

In this paper we present LLR-2^K b SCL decoder algorithm for polar codes decoding. The proposed algorithm can reduce decoding latency and increases the throughput at the same time without any bit loss. Then, based on the proposed algorithm, we develop the corresponding Verilog code. Performance analysis and results shows that the proposed SCL decoders have significant reduction in errors and decoding latency.

REFERENCES

[1] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051-3073, 2009.

[2] I. Tal and A. Vardy, "List decoding of polar codes," arXiv:1206.0050, May 2012.

[3] A. Balatsoukas-Stimming, M. Bastani Parizi and A. Burg, "LLR- based successive cancellation list decoding of polar codes,"arXiv:1401.3753v3.

[4] B. Yuan and K.K. Parhi, "Successive cancellation list polar decoder using Log-likelihood ratios," in *Proc. of Asilomar Conf. on Signal, Systems and Computers*, pp. 548-552, 2014.

[5] B. Yuan and K.K. Parhi, "Low-latency successive-cancellation list decoders for polar codes with multi-bit decision," *IEEE Trans. on VLSI Systems*, vol. 23, no. 10, pp. 2268 – 2280, Oct. 2015.

[6] B. Li, H. Shen, D. Tse and W. Tong, "Low-Latency Polar Codes via Hybrid Decoding," in *Proc. of 8th Intl. Symp. on Turbo Codes and Iterative Info. Processing (ISTC)*, pp. 223-227, Aug. 2014.

[7] B. Yuan and K.K. Parhi, "Successive cancellation decoding of polar codes using stochastic computing," in *Proc. of IEEE Intl. Symp. on Circuits and Systems (ISCAS)*, May 2015.

[8] B. Yuan and K.K. Parhi, "Low-Latency successive-cancellation polar decoder architectures using 2-bit decoding," *IEEE Trans. Circuits and Systems-I: Regular Papers*, vol. 61, no. 4, pp. 1241-1254, Apr. 2014.

[9] J. Lin and Z. Yan, "An efficient list decoder architecture for polar codes," accepted by *IEEE Trans. on VLSI Systems*, 2015.

[10] B. Yuan and K.K. Parhi, "Reduced-latency LLR-based SC list decoder for polar codes," in *Proc. of 2015 ACM Great Lakes Symposium on VLSI*, pp. 107-110, May 2015.

[11] H. Vangala, E. Viterbo and Y. Hong, "A New Multiple Folded Successive Cancellation Decoder for Polar Codes," in *Proc. of IEEE Information Theory Workshop (ITW)*, pp. 381-385, Nov. 2014.