Adder Design and Analysis Report

Athav Vantan B (23B1306)
Santhosh Kumar B (23B1221)
Tamilaruvi S (23B1286)

12/12/2024

Introduction

Binary addition is a fundamental arithmetic operation in modern digital systems, critical for components like microprocessors, digital signal processors, and arithmetic logic units (ALUs). Efficient adders directly influence system performance by determining the minimum clock cycle time in processors. Modern adder architectures address the challenges of carry propagation delay using parallel prefix adder (PPA) structures, which divide the operation into pre-processing, carry look-ahead, and post-processing stages.

Key highlights:

- **Significance:** Binary adders are essential in ALUs, multipliers, and memory address generation.
- **Design Challenges:** Carry chain propagation delay increases with input width, impacting speed.
- **Solution:** Parallel Prefix Adders (PPAs) provide a structured approach to mitigate carry propagation issues, ensuring fast performance and efficient VLSI implementation.
- **Focus:** This report evaluates the performance of the following PPAs in terms of delay, area, and power across 8-bit, 32-bit, and 64-bit input widths:
  - Kogge Stone Adder
  - Brent Kung Adder
  - Han Carlson Adder
  - Sklansky Adder
  - Knowles Adder
  - Lander Fisher Adder

Timing Analysis

Timing Analysis was done using **Timing Analyser UI** of Quartus Software. **Input registers**(launch register) and **output registers**(latch register) were instantiated to store the inputs to and outputs from the adders, facilitating the use of a clock for timing analysis .Different clocks of suitable timeperiods were used for simulating different adders. The **Data Delay** corresponding to the path with least **slack value**, is measured(for each adder) which is the time taken for data to reach the input of latch register from the launch register - from the instant when clock reaches the launch register.

Details of board used for simulation
Board : Arrow MAX 10 DECA
Version : 0.9
Family : MAX 10
Device : 10M50DAF484C6GES
Vendor : Arrow
Total I/Os : 360

Logic Verification

RTL simulation(using **Modelsim Altera**) with **testbench and tracefiles** was performed for each adder to verify the correctness of logic.

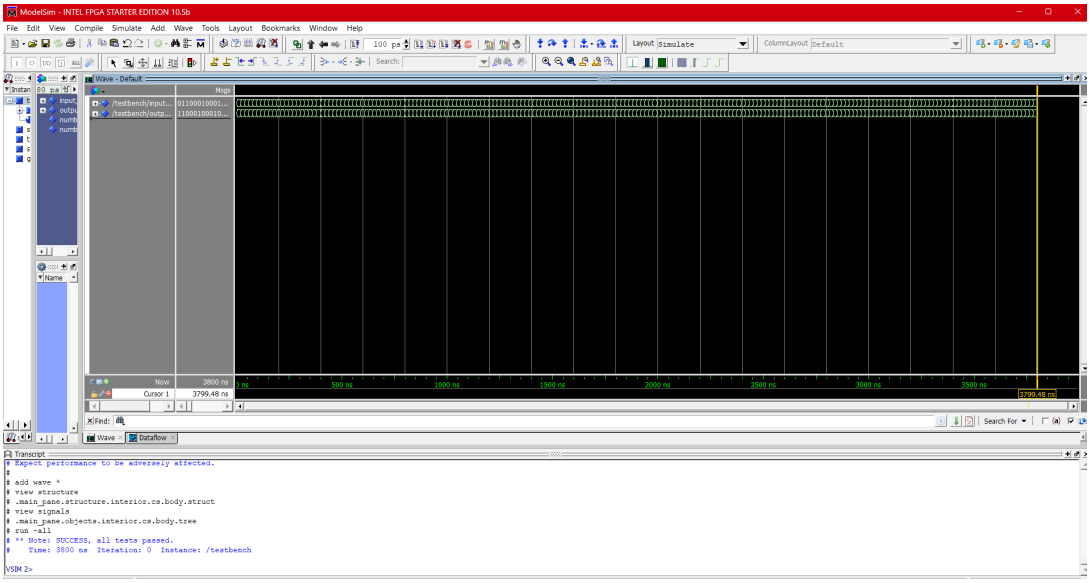

Figure 1: Eg. RTL simulation of 64 bit Kogge Stone Adder

Brent Kung Adder

The Brent Kung Adder is a parallel prefix form of the carry lookahead adder, designed to balance delay and area efficiently. In conventional carry lookahead adders, increasing the size of the input operands leads to increased delay. The Brent Kung Adder addresses this challenge by achieving a gate-level depth of $O(\log_2(n))$, which reduces delay while maintaining power efficiency. It requires less area and wiring congestion compared to other prefix adders like the Kogge Stone Adder. Instead of using a traditional carry chain to calculate the output, it employs a parallel prefix approach, minimizing delay without compromising the performance of the adder.

Block Diagram :

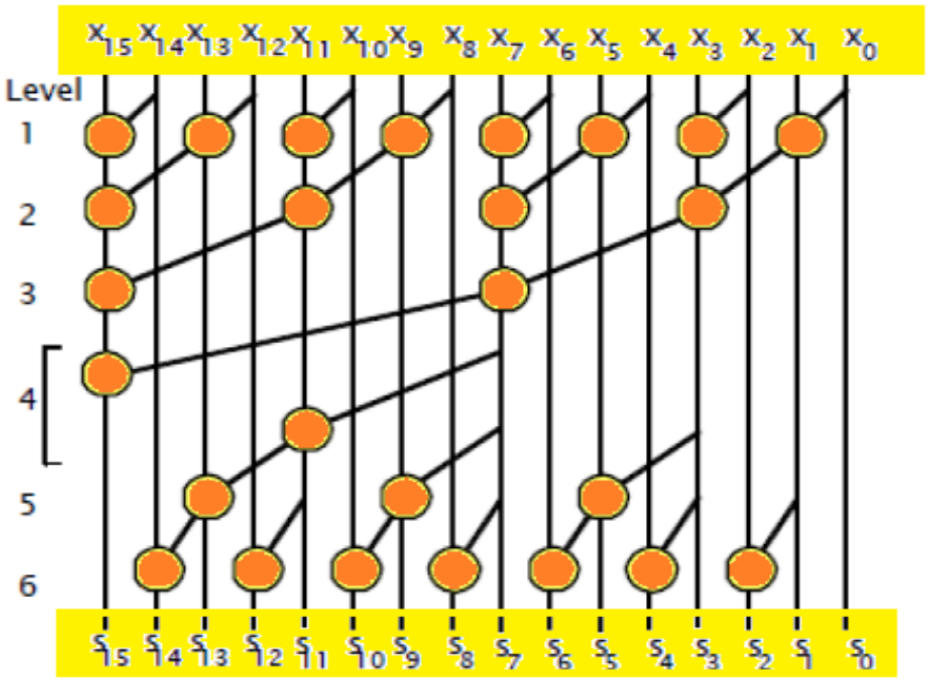

Figure 2: Block Diagram for Brent Kung Adder

8-bit Brent Kung Adder

Resource Utilization:

Table 1: Resource Utilization for 8-bit Brent Kung Adder

Metric	Value
Total Logic Elements	22
Total Registers	0
Total Pins	26

Timing Analysis:

Table 2: Timing Analysis for 8-bit Brent Kung Adder

Metric	Value
Setup Slack	0.961 ns
Data Delay	2.963 ns
Fmax	329.06 MHz

Waveform :

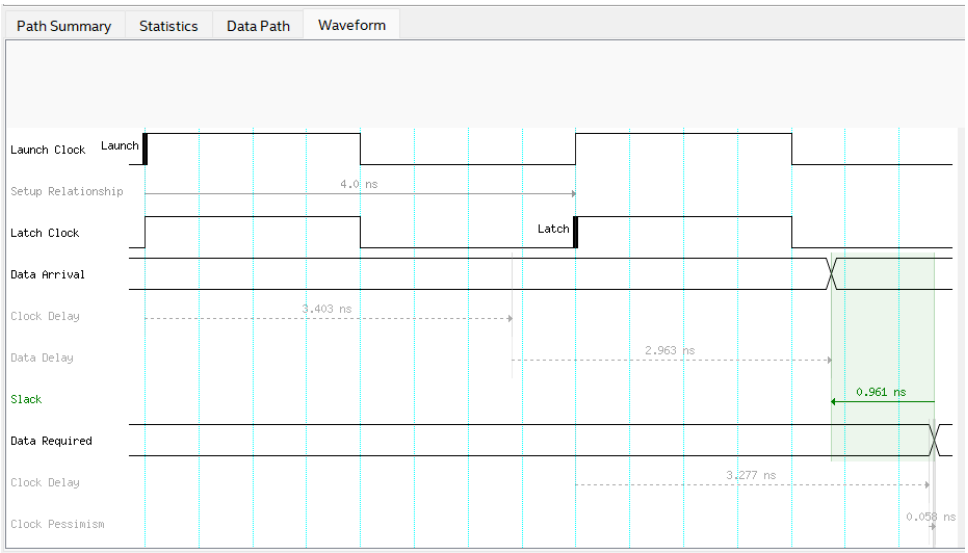

Figure 3: Timing Waveform for 8-bit Brent Kung Adder

32-bit Brent Kung Adder

Resource Utilization:

Table 3: Resource Utilization for 32-bit Brent Kung Adder

Metric	Value
Total Logic Elements	108
Total Registers	0
Total Pins	98

Timing Analysis:

Table 4: Timing Analysis for 32-bit Brent Kung Adder

Metric	Value
Setup Slack	0.034 ns
Data Delay	4.890 ns
Fmax	201.37 MHz

Waveform :

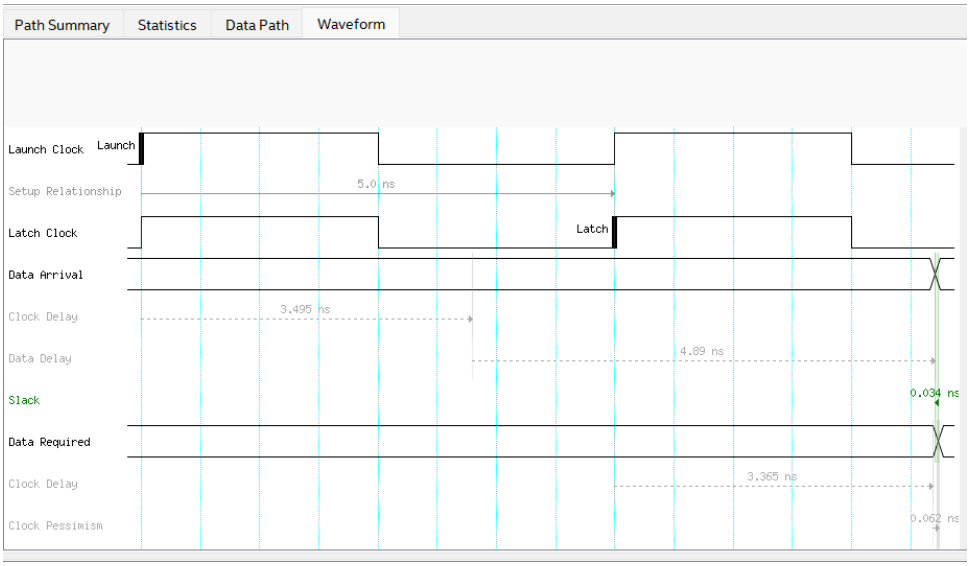

Figure 4: Timing Waveform for 32-bit Brent Kung Adder

64-bit Brent Kung Adder

Resource Utilization:

Table 5: Resource Utilization for 64-bit Brent Kung Adder

Metric	Value
Total Logic Elements	215
Total Registers	0
Total Pins	194

Timing Analysis:

Table 6: Timing Analysis for 64-bit Brent Kung Adder

Metric	Value
Setup Slack	0.814 ns
Data Delay	6.111 ns
Fmax	161.66 MHz

Waveform :

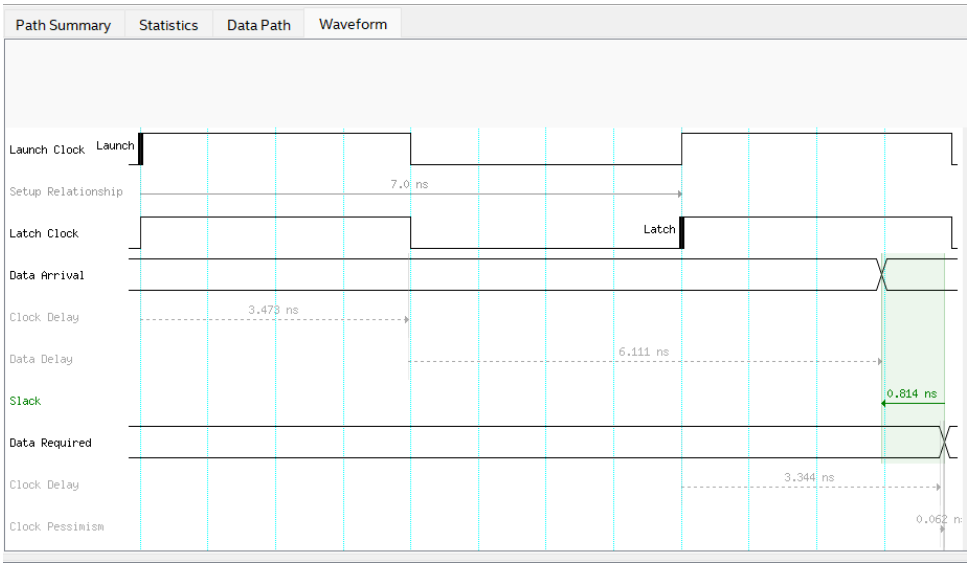

Figure 5: Timing Waveform for 64-bit Brent Kung Adder

Knowles Adder

The Knowles Adder is a type of parallel prefix adder designed to optimize speed by reducing fan-out in the carry propagation network. It achieves a more uniform distribution of logic levels, which helps in maintaining low delays for higher operand widths. The Knowles Adder is known for its balanced performance, making it suitable for applications requiring high-speed operations. Compared to other prefix adders, it provides a trade-off between delay and area, offering reduced delay at the expense of slightly higher wiring complexity. Its structured approach ensures efficient implementation in modern digital systems.

Block Diagram :

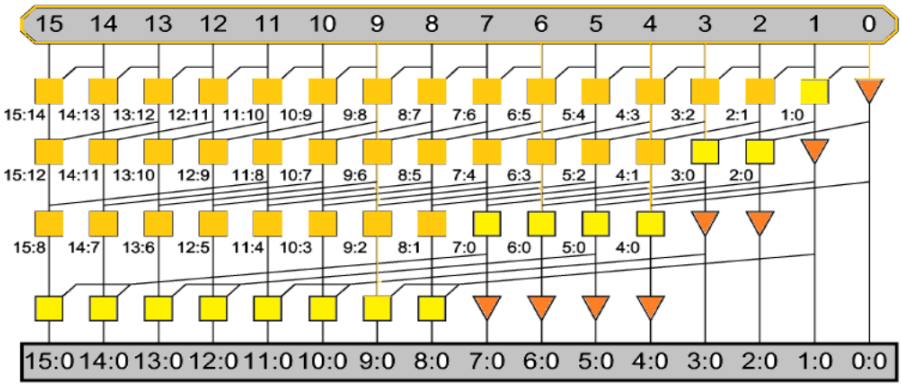

Figure 6: Block Diagram for Knowles Adder

8-bit Knowles Adder

Resource Utilization:

Table 7: Resource Utilization for 8-bit Knowles Adder

Metric	Value
Total Logic Elements	33
Total Registers	0
Total Pins	26

Timing Analysis:

Table 8: Timing Analysis for 8-bit Knowles Adder

Metric	Value
Setup Slack	0.336 ns
Data Delay	2.593 ns
Fmax	375.38 MHz

Waveform :

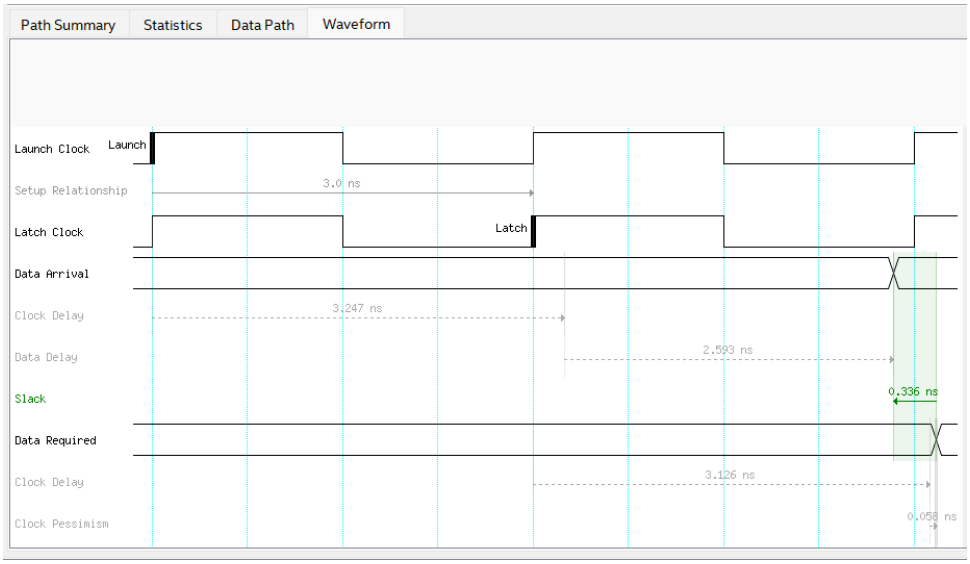

Figure 7: Timing Waveform for 8-bit Knowles Adder

32-bit Knowles Adder

Resource Utilization:

Table 9: Resource Utilization for 32-bit Knowles Adder

Metric	Value
Total Logic Elements	225
Total Registers	0
Total Pins	98

Timing Analysis:

Table 10: Timing Analysis for 32-bit Knowles Adder

Metric	Value
Setup Slack	0.482 ns
Data Delay	4.827 ns
Fmax	221.34 MHz

Waveform :

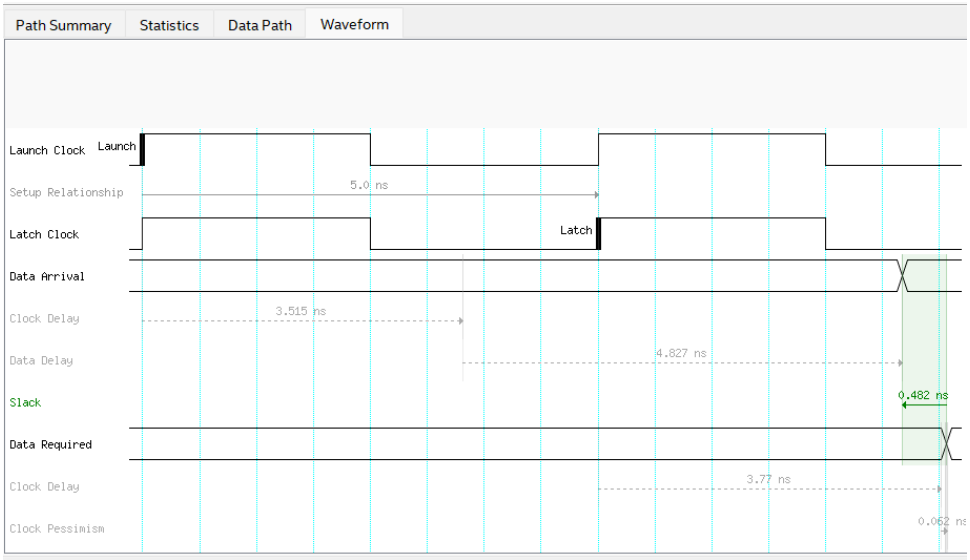

Figure 8: Timing Waveform for 32-bit Knowles Adder

64-bit Knowles Adder

Resource Utilization:

Table 11: Resource Utilization for 64-bit Knowles Adder

Metric	Value
Total Logic Elements	574
Total Registers	0
Total Pins	194

Timing Analysis:

Table 12: Timing Analysis for 64-bit Knowles Adder

Metric	Value
Setup Slack	0.671 ns
Data Delay	5.687 ns
Fmax	192.12 MHz

Waveform :

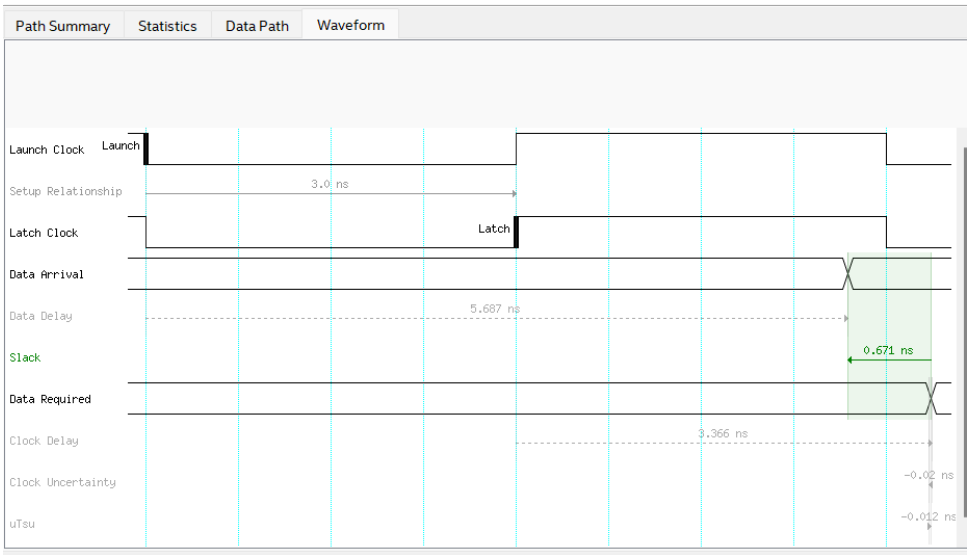

Figure 9: Timing Waveform for 64-bit Knowles Adder

Kogge Stone Adder

The Kogge Stone Adder is a parallel prefix adder widely used in high-performance applications due to its efficiency in handling large operand widths. It extends the basic concept of a carry lookahead adder by introducing a parallel carry prefix chain, which significantly reduces the delay associated with carry propagation.

In the first level, 2-bit generate (G) and propagate (P) signals are computed simultaneously. In subsequent levels, higher-order carry-out values are derived in parallel by using the results from previous levels. For instance, the carry-out of the 4th bit is calculated in the second level, the 8th bit in the third level, and so on. This parallel computation ensures that carry-out values for all bits are available with minimal delay.

The Kogge Stone Adder achieves this by using propagate and generate blocks, where the delay of each block is equivalent to two gate delays. This structured approach makes the Kogge Stone Adder one of the fastest adder designs, particularly suited for systems requiring high-speed arithmetic operations.

Block Diagram :

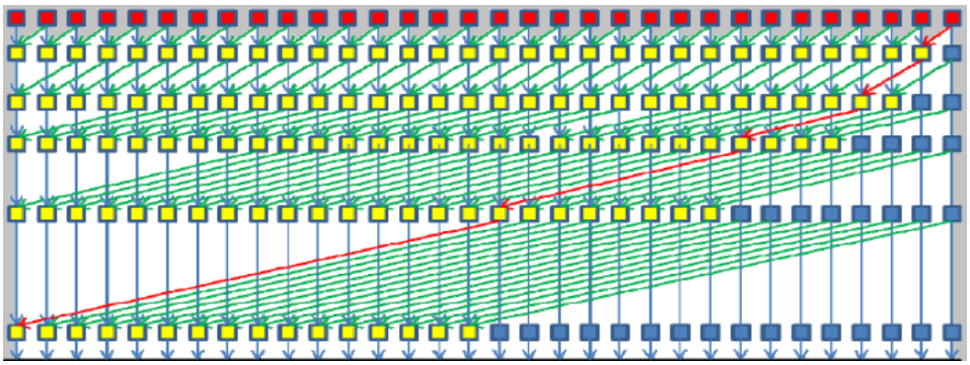

Figure 10: Block Diagram for Kogge Stone Adder

8-bit Kogge Stone Adder

Resource Utilization:

Table 13: Resource Utilization for 8-bit Kogge Stone Adder

Metric	Value
Total Logic Elements	27
Total Registers	0
Total Pins	26

Timing Analysis:

Table 14: Timing Analysis for 8-bit Kogge Stone Adder

Metric	Value
Setup Slack	0.130 ns
Data Delay	2.797 ns
Fmax	348.43 MHz

Waveform :

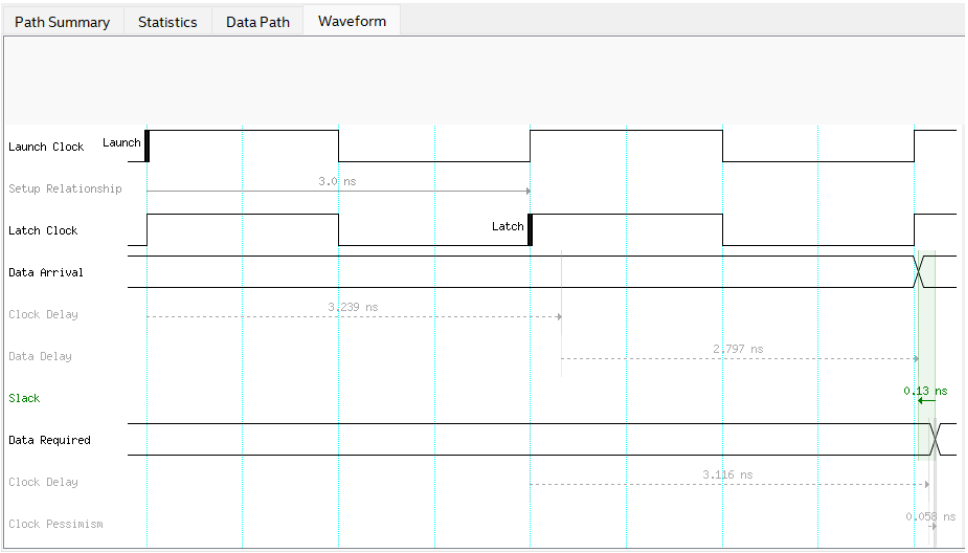

Figure 11: Timing Waveform for 8-bit Kogge Stone Adder

32-bit Kogge Stone Adder

Resource Utilization:

Table 15: Resource Utilization for 32-bit Kogge Stone Adder

Metric	Value
Total Logic Elements	218
Total Registers	0
Total Pins	98

Timing Analysis:

Table 16: Timing Analysis for 32-bit Kogge Stone Adder

Metric	Value
Setup Slack	0.512 ns
Data Delay	4.122 ns
Fmax	222.82 MHz

Waveform :

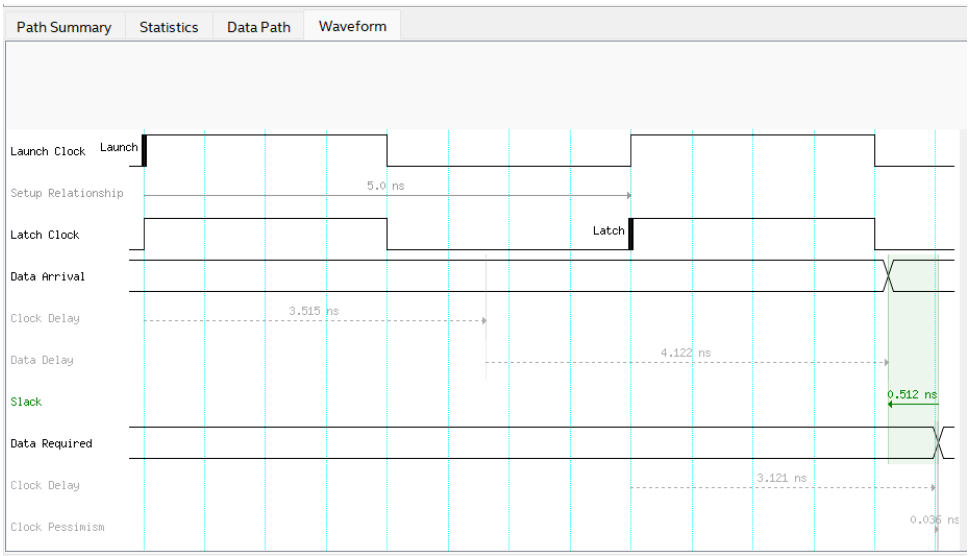

Figure 12: Timing Waveform for 32-bit Kogge Stone Adder

64-bit Kogge Stone Adder

Resource Utilization:

Table 17: Resource Utilization for 64-bit Kogge Stone Adder

Metric	Value
Total Logic Elements	570
Total Registers	0
Total Pins	194

Timing Analysis:

Table 18: Timing Analysis for 64-bit Kogge Stone Adder

Metric	Value
Setup Slack	0.103 ns
Data Delay	5.203 ns
Fmax	204.21 MHz

Waveform :

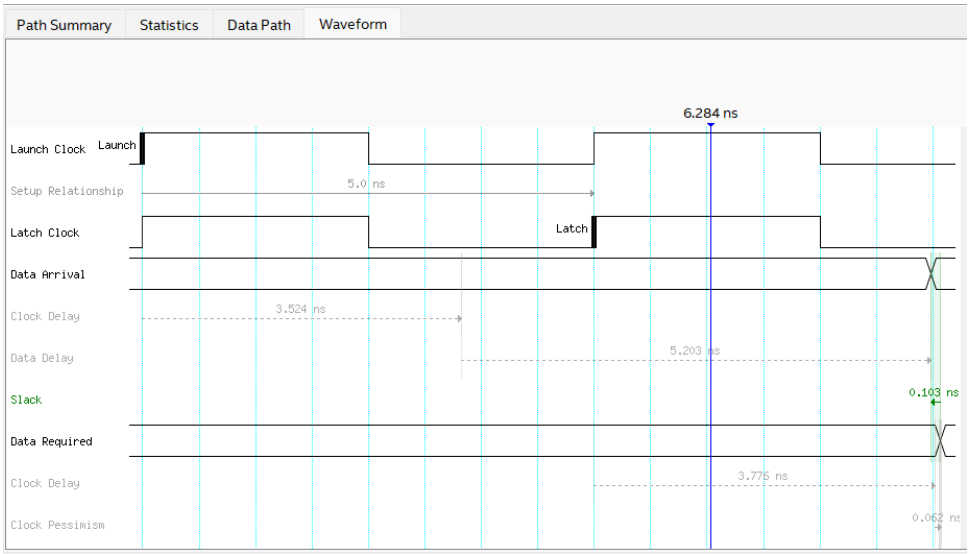

Figure 13: Timing Waveform for 64-bit Kogge Stone Adder

Sklansky Adder

The Sklansky Adder is a type of parallel prefix adder known for its simple and regular architecture. It uses a binary tree structure of propagate and generate cells to compute all the carry-in values (Cin) simultaneously. The design recursively constructs 2-bit adders, which are then combined to form 4-bit, 8-bit, 16-bit, and larger adders by merging smaller adders at each level.

While the Sklansky Adder offers a straightforward and efficient structure, it suffers from fan-out issues that may impact performance in certain implementations. However, in specific cases, it can achieve the same addition delay with fewer propagate and generate cells, making it a competitive option for applications requiring efficient area utilization.

Block Diagram :

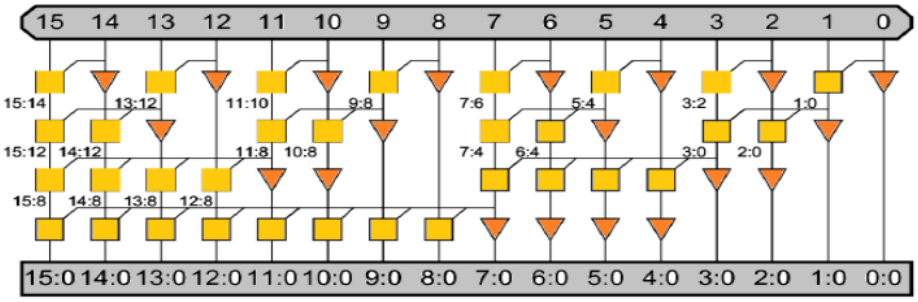

Figure 14: Block Diagram for Sklansky Adder

8-bit Sklansky Adder

Resource Utilization:

Table 19: Resource Utilization for 8-bit Sklansky Adder

Metric	Value
Total Logic Elements	26
Total Registers	0
Total Pins	26

Timing Analysis:

Table 20: Timing Analysis for 8-bit Sklansky Adder

Metric	Value
Setup Slack	0.336 ns
Data Delay	2.594 ns
Fmax	375.38 MHz

Waveform :

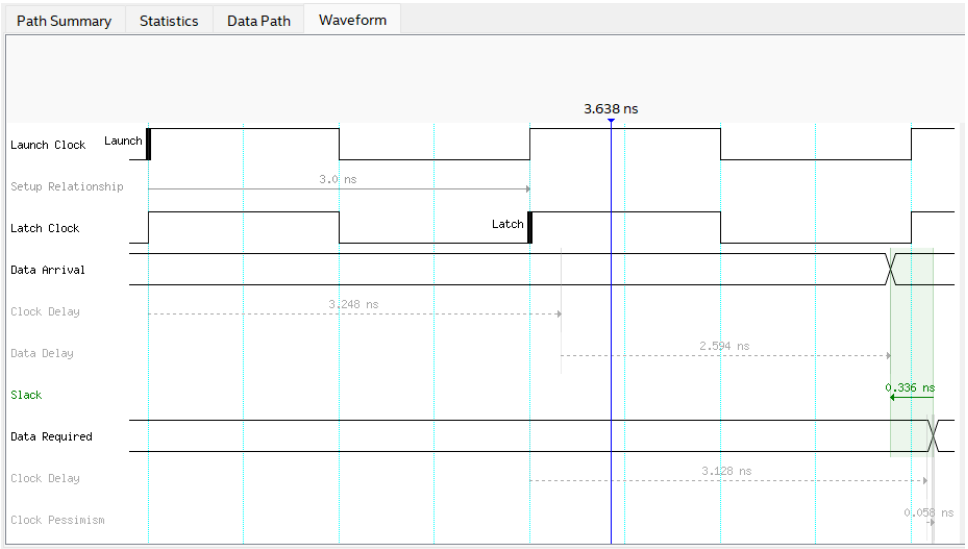

Figure 15: Timing Waveform for 8-bit Sklansky Adder

32-bit Sklansky Adder

Resource Utilization:

Table 21: Resource Utilization for 32-bit Sklansky Adder

Metric	Value
Total Logic Elements	135
Total Registers	0
Total Pins	98

Timing Analysis:

Table 22: Timing Analysis for 32-bit Sklansky Adder

Metric	Value
Setup Slack	0.943 ns
Data Delay	4.976 ns
Fmax	197.75 MHz

Waveform :

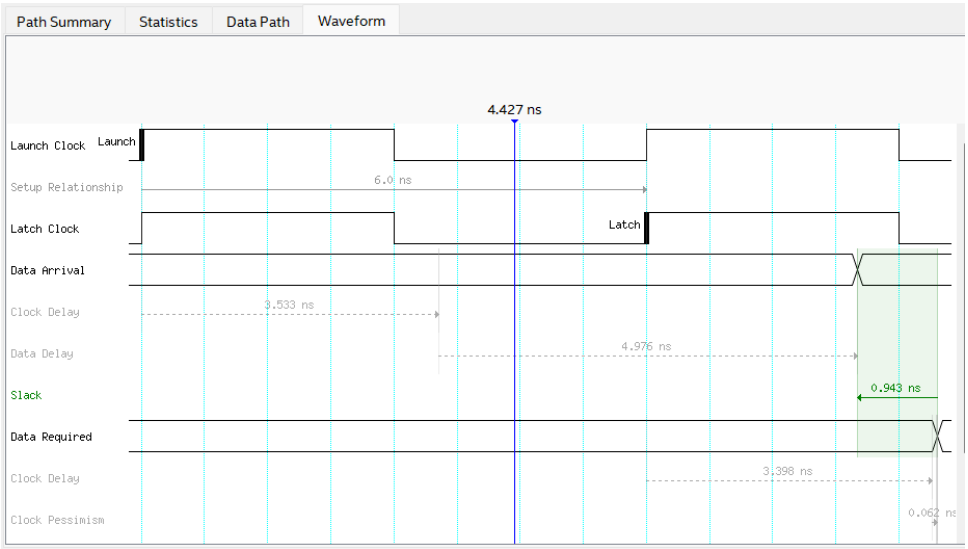

Figure 16: Timing Waveform for 32-bit Sklansky Adder

64-bit Sklansky Adder

Resource Utilization:

Table 23: Resource Utilization for 64-bit Sklansky Adder

Metric	Value
Total Logic Elements	283
Total Registers	0
Total Pins	194

Timing Analysis:

Table 24: Timing Analysis for 64-bit Sklansky Adder

Metric	Value
Setup Slack	0.763 ns
Data Delay	6.188 ns
Fmax	160.33 MHz

Waveform :

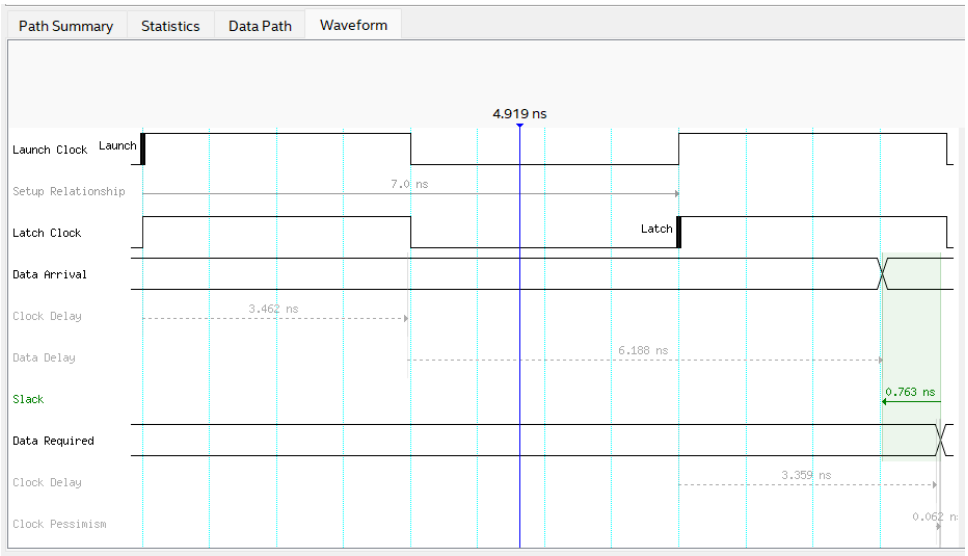

Figure 17: Timing Waveform for 64-bit Sklansky Adder

Han Carlson Adder

The Han Carlson Adder is a parallel prefix adder that combines features of the Brent Kung and Kogge Stone adders to create a hybrid design with reduced complexity. This adder performs carry-merge operations on even bits while propagating generate and propagate signals for odd bits down the prefix tree. At the final stage, the odd bits recombine with even bit carry signals to produce the true carry values.

This design reduces complexity compared to the Brent Kung Adder by leveraging a more efficient structure. However, the reduced complexity comes at the cost of adding an additional stage to the carry-merge path. The Han Carlson Adder is particularly useful for applications where a balance between area efficiency and performance is required.

Block Diagram :

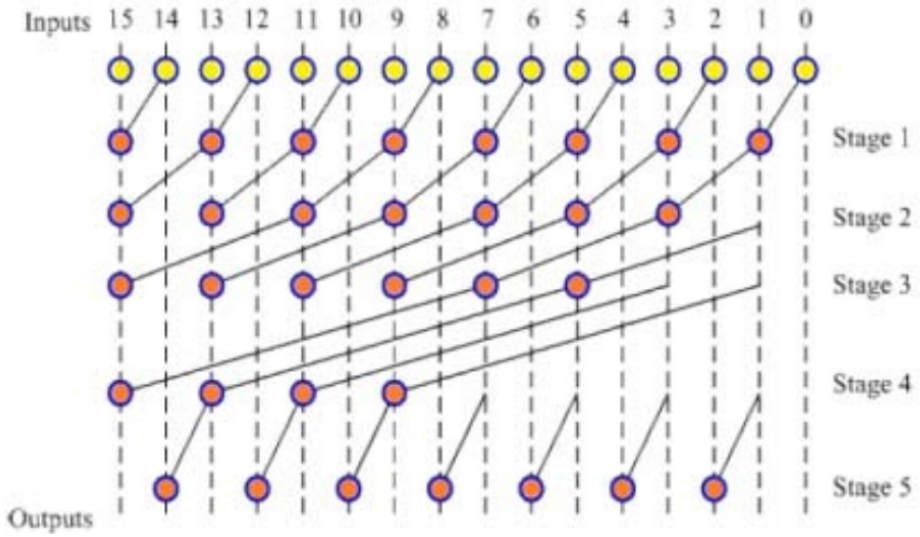

Figure 18: Block Diagram for Han Carlson Adder

8-bit Han Carlson Adder

Resource Utilization:

Table 25: Resource Utilization for 8-bit Han Carlson Adder

Metric	Value
Total Logic Elements	22
Total Registers	0
Total Pins	25

Timing Analysis:

Table 26: Timing Analysis for 8-bit Han Carlson Adder

Metric	Value
Setup Slack	0.061 ns
Data Delay	2.869 ns
Fmax	340.25 MHz

Waveform :

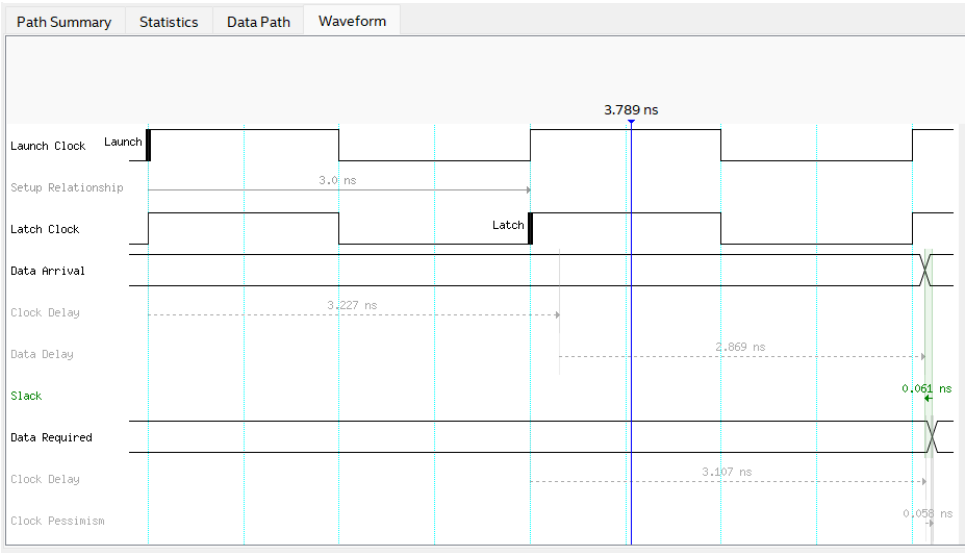

Figure 19: Timing Waveform for 8-bit Han Carlson Adder

32-bit Han Carlson Adder

Resource Utilization:

Table 27: Resource Utilization for 32-bit Han Carlson Adder

Metric	Value
Total Logic Elements	144
Total Registers	0
Total Pins	97

Timing Analysis:

Table 28: Timing Analysis for 32-bit Han Carlson Adder

Metric	Value
Setup Slack	0.298 ns
Data Delay	4.632 ns
Fmax	212.68 MHz

Waveform :

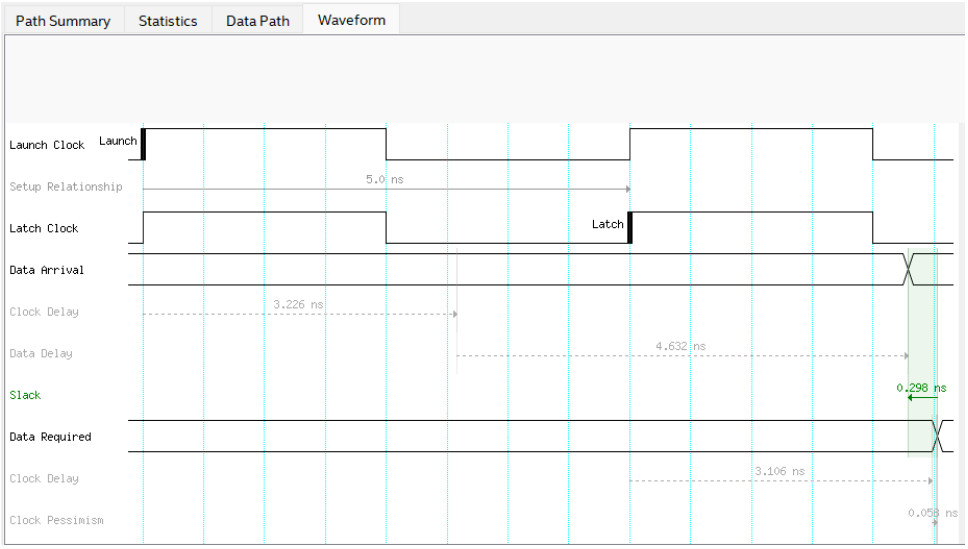

Figure 20: Timing Waveform for 32-bit Han Carlson Adder

64-bit Han Carlson Adder

Resource Utilization:

Table 29: Resource Utilization for 64-bit Han Carlson Adder

Metric	Value
Total Logic Elements	345
Total Registers	0
Total Pins	193

Timing Analysis:

Table 30: Timing Analysis for 64-bit Han Carlson Adder

Metric	Value
Setup Slack	0.428 ns
Data Delay	5.495 ns
Fmax	179.47 MHz

Waveform :

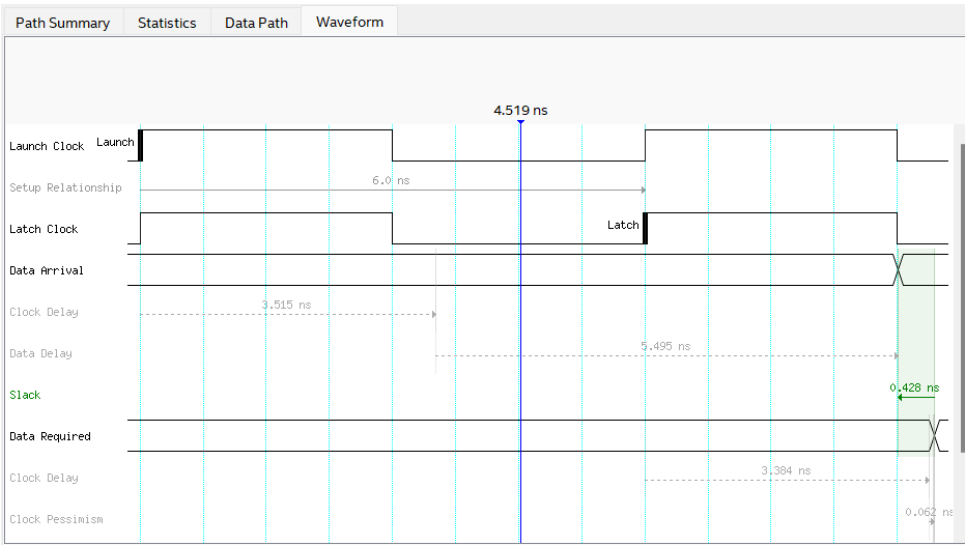

Figure 21: Timing Waveform for 64-bit Han Carlson Adder

Lander Fisher Adder

The Lander Fisher Adder is a parallel prefix adder that emphasizes efficient computation by strategically balancing delay and area. It employs a hierarchical prefix structure to calculate generate and propagate signals, ensuring faster carry generation. The design integrates multiple levels of carry computation, similar to other parallel prefix adders, but focuses on optimizing hardware complexity without significant performance trade-offs.

This adder is particularly effective in applications requiring minimal delay with reasonable area utilization. The Lander Fisher Adder's structure is tailored to meet the needs of modern digital systems, where both speed and resource efficiency are critical.

Block Diagram :

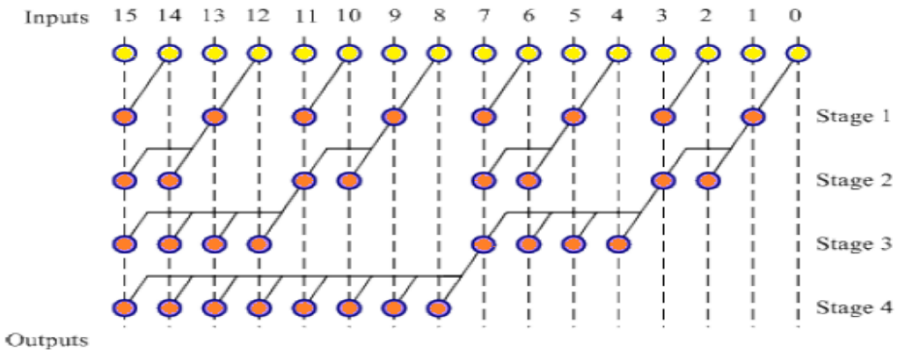

Figure 22: Block Diagram for Lander Fisher Adder

16-bit Lander Fisher Adder

Resource Utilization:

Table 31: Resource Utilization for 16-bit Lander Fisher Adder

Metric	Value
Total Logic Elements	55
Total Registers	0
Total Pins	49

Timing Analysis:

Table 32: Timing Analysis for 16-bit Lander Fisher Adder

Metric	Value
Setup Slack	0.714 ns
Data Delay	4.208 ns
Fmax	233.32 MHz

Waveform :

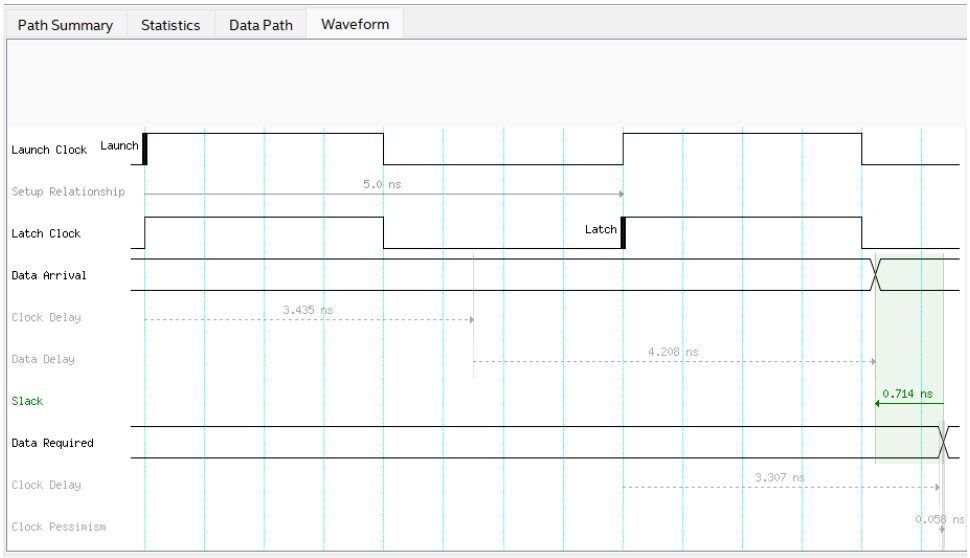

Figure 23: Timing Waveform for 16-bit Lander Fisher Adder

32-bit Lander Fisher Adder

Resource Utilization:

Table 33: Resource Utilization for 32-bit Lander Fisher Adder

Metric	Value
Total Logic Elements	130
Total Registers	0
Total Pins	97

Timing Analysis:

Table 34: Timing Analysis for 32-bit Lander Fisher Adder

Metric	Value
Setup Slack	0.563 ns
Data Delay	4.769 ns
Fmax	225.38 MHz

Waveform :

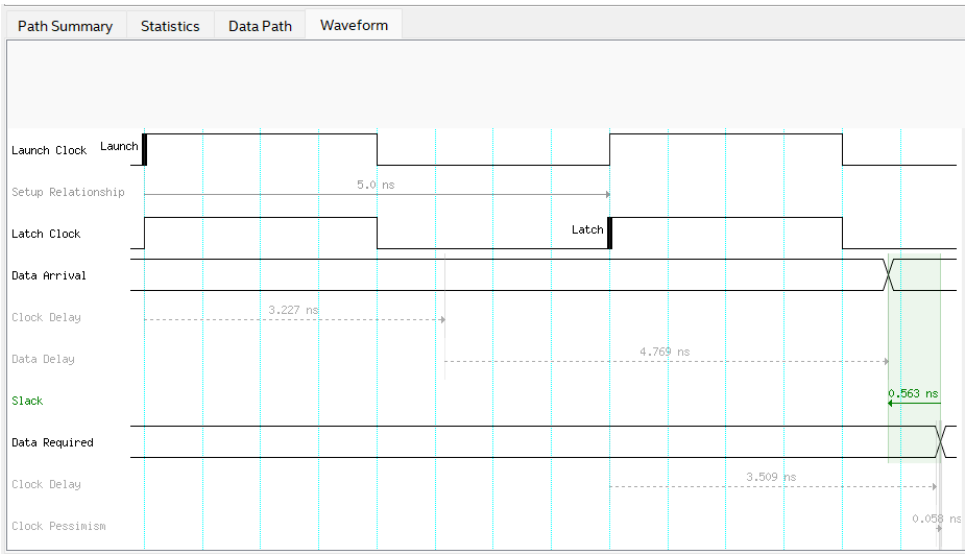

Figure 24: Timing Waveform for 32-bit Lander Fisher Adder

64-bit Lander Fisher Adder

Resource Utilization:

Table 35: Resource Utilization for 64-bit Lander Fisher Adder

Metric	Value
Total Logic Elements	282
Total Registers	0
Total Pins	193

Timing Analysis:

Table 36: Timing Analysis for 64-bit Lander Fisher Adder

Metric	Value
Setup Slack	0.793 ns
Data Delay	6.118 ns
Fmax	161.11 MHz

Waveform :

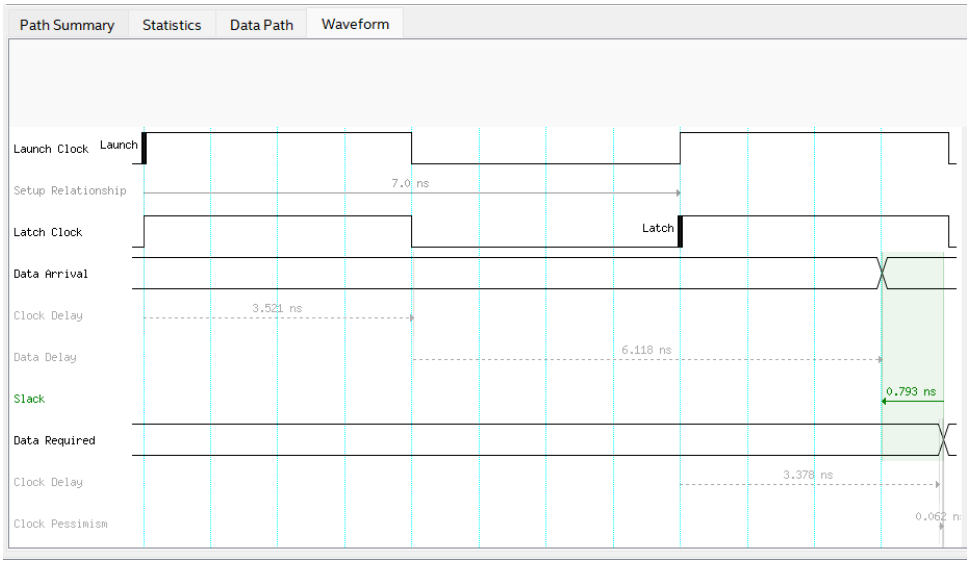

Figure 25: Timing Waveform for 64-bit Lander Fisher Adder

Summary

Table 37: Performance Comparison of Adder Architectures

Adder Architecture	No of bits	Data Delay(ns)	Fmax(MHz)	Logic Elements
Brent-Kung	8	2.963	329.06	22
Brent-Kung	32	4.890	201.37	108
Brent-Kung	64	6.111	161.66	215
Knowles	8	2.593	375.38	33
Knowles	32	4.827	221.34	225
Knowles	64	5.687	192.12	574
Kogge-Stone	8	2.797	348.43	27
Kogge-Stone	32	4.122	222.82	218
Kogge-Stone	64	5.203	204.21	570
Sklansky	8	2.594	375.38	26
Sklansky	32	4.976	197.75	135
Sklansky	64	6.188	160.33	283
Han Carlson	8	2.869	340.25	22
Han Carlson	32	4.632	212.68	144
Han Carlson	64	5.495	179.47	345
Lander Fisher	16	4.208	233.32	55
Lander Fisher	32	4.769	225.38	130
Lander Fisher	64	6.118	161.11	282

Work Distribution

1. Athav Vantan B (23B1306) : Brent Kung, Knowles
2. Santhosh Kumar B (23B1221) : Kogge Stone, Sklansky
3. Tamilaruvi S (23B1286) : Han Carlson Adder, Lander Fisher Adder

Github Link

Link to Github Repo:
`https://github.com/santhoshkumarbalan/EE-224-Adders`