Design Synthesis Comparative Analysis of Binary Adders

Guilherme B. Manske
CI-Inovador - UFRGS

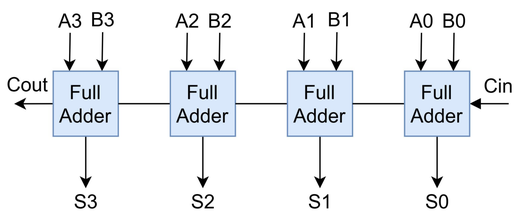
Paulo F. Butzen
PGMICRO - UFRGS

Renato P. Ribas
PGMICRO-UFRGS

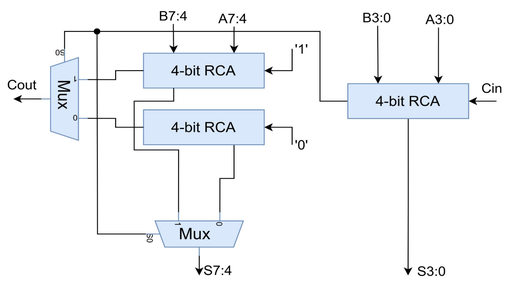
1. Introduction

- Computational systems are increasingly reliant on efficient digital arithmetic to meet the growing demands of modern applications.
- Addition forms the basis of more complex operations like multiplication, division, and comparison.
- The design and efficiency of adders significantly impact the overall performance, area, and power consumption of a digital system.
- Adder takes two binary numbers and optionally a carry input (Cin) to produce a sum (S) and a carry output (Cout).From basic half and full adders to more complex architectures.
- Multibit addition is implemented by cascading multiple full adders, forming the basis for more complex carry chain based adder structures.
- Carry propagate/generate based adder architectures were introduced to overcome the limitations of simple carry chain designs, reducing the carry propagation delay.
- To evaluate each adder architecture, it is important to apply both logical and physical synthesis.Metrics such as area, frequency, and power are used to compare designs.
- The designs are implemented in SystemVerilog and subjected to different methodologies. The analysis highlights how each design performs under real-world ASIC implementation constraints.

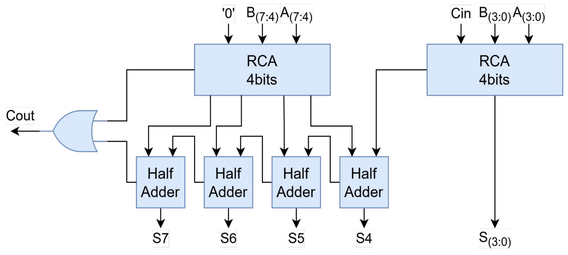
2. Adders



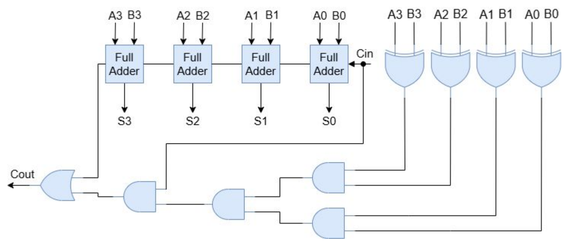
Ripple Carry Adder



Carry Select Adder

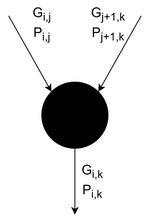


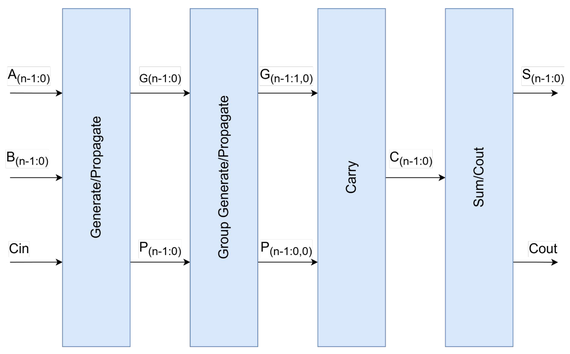
Carry Increment Adder



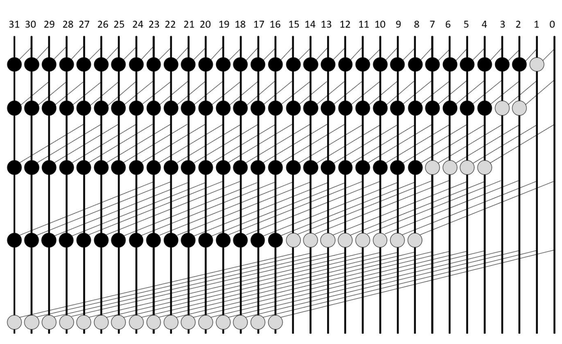
Carry Skip Adder

- Full Adder:
  - $S_i = A_i \wedge B_i \wedge C_{i-1}$
  - $C_i = A_iB_i + A_iC_{i-1} + B_iC_{i-1}$
- Generate/Propagate
  - $P_i = A_i \wedge B_i$
  - $G_i = A_iB_i$
  - $P_{i,0} = P_{i-1}P_{i-2}P_{i-3}\ldots P_0$
  - $G_{i,0} = G_i + P_iG_{i-1} + \ldots P_iP_0Cin$
  - $C_i = G_i + P_iC_{i-1}$
  - $S_i = A_i \wedge B_i \wedge C_{i-1} = P_i \wedge C_{i-1}$
  - $G_{7,4} = G_{7,6} + P_{7,6}G_{5,4}$
  - $P_{7,4} = P_{7,6} + P_{5,4}$





Parallel Prefix Adder Blocks



Kogge-Stone Prefix Tree

3. Methodologies

- Area Oriented Methodology - a very relaxed timing constraint was set, allowing Genus and Innovus to focus primarily on minimizing power consumption and area. This methodology allows the tools to select slower, more compact implementations, making it ideal for evaluating efficiency in low-performance, resource-constrained applications.
- Time Oriented Methodology - This method tries to determine the tightest achievable timing constraint for each adder architecture individually. This allows for a fair comparison of the maximum attainable speed for each architecture under aggressive timing requirements. The algorithm described in Fig. 1 is applied here for all designs.
- Reference Frequency Methodology - The time constraint of the Ripple Carry Adder, typically the slowest architecture, is optimized first. This constraint is then applied to all other designs, enabling a consistent and fair comparison of area and power under equal timing pressure. This methodology highlights how each architecture adapts when subjected to the same real-time performance target.
- Behavioral Adder Synthesis - A high-level adder using the '+' operator is synthesized to explore how aggressively the tools optimize behavioral descriptions. Additionally, this methodology evaluates whether other designs can benefit from similar optimizations by removing structural constraints. The analysis provides insights into tool-driven recognition and optimization of arithmetic patterns.

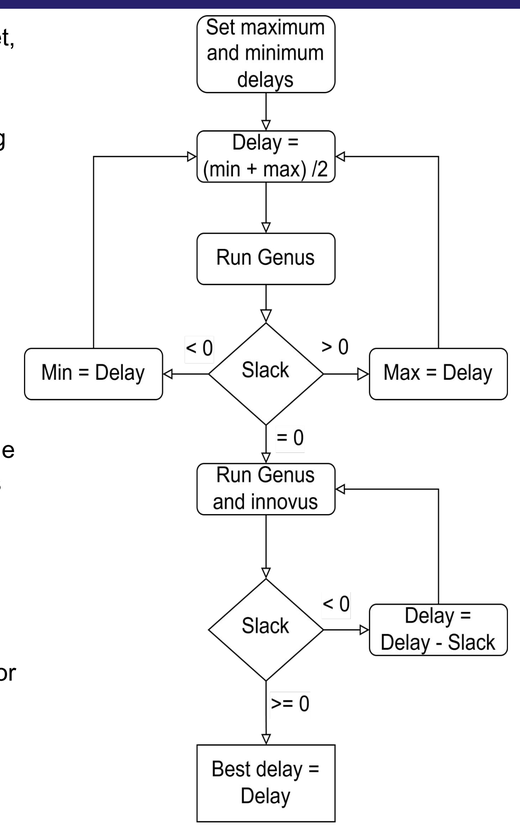
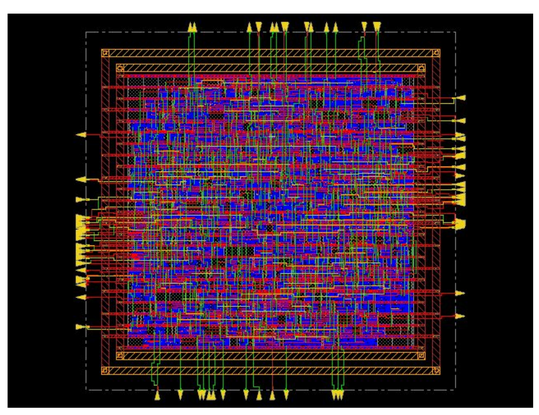
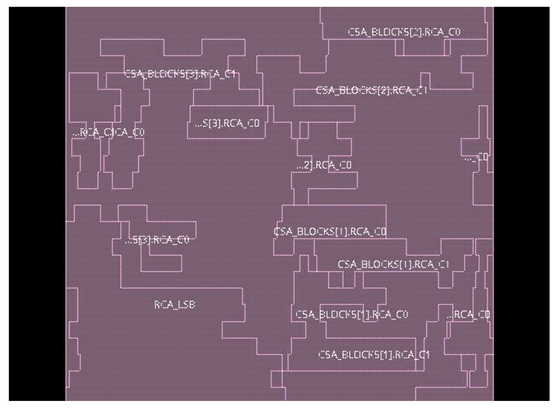


Fig. 1. Time Optimization Fluxogram

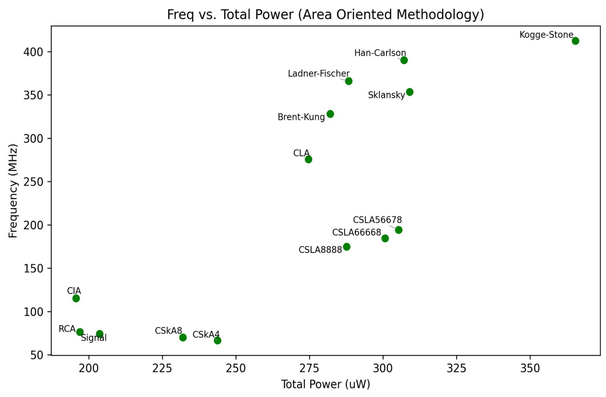
4. Resultados



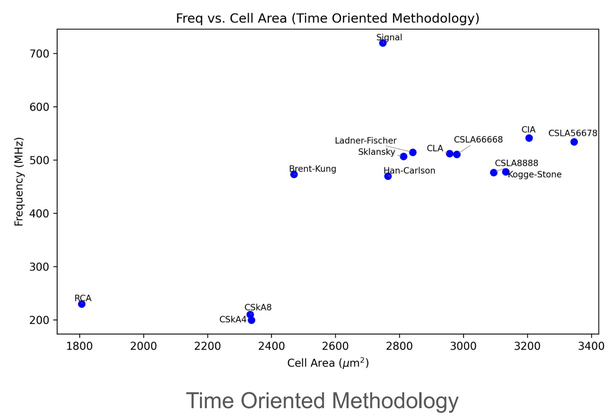
,32-bit Kogge Stone Adder



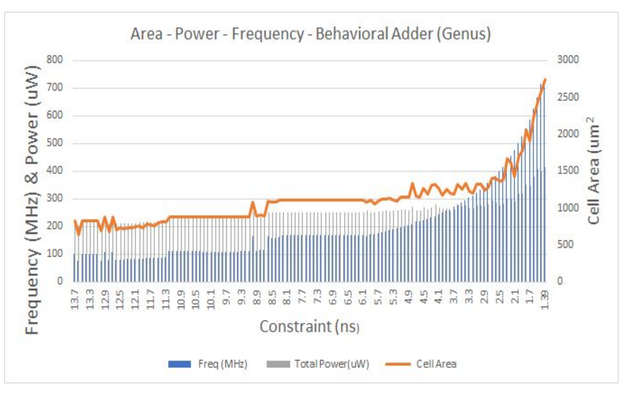
CSLA8888



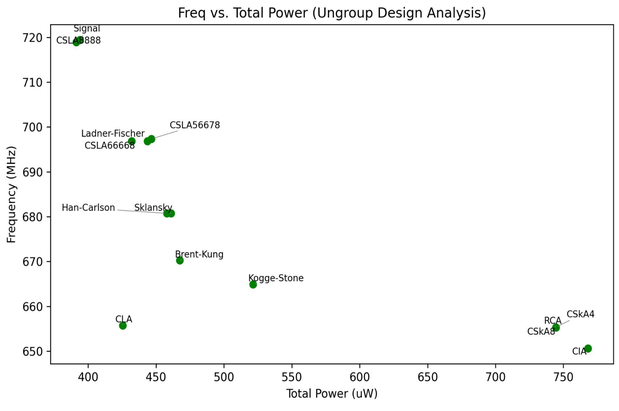
Area Oriented Methodology



Time Oriented Methodology



Behavioral Synthesis



Behavioral Synthesis

5. Conclusão

- The Ripple Carry Adder offers the smallest area footprint, the Carry Increment Adder balances area and power efficiently. Parallel prefix adders such as Kogge-Stone and Ladner-Fischer provide high performance but at the expense of increased area, power and routing complexity. The behavioral design using the '+' operator benefited from tool-level optimizations, highlighting the impact of synthesis attributes such as ungrouping on final performance.
- This study emphasize the importance of aligning adder architecture selection with specific design constraints, such as area, power, or timing, based on the intended application. The proposed flow and automated methodology also enable reproducible and fair comparisons of arithmetic blocks in ASIC design flows.

Linkedin

GitHub

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO
Softex