OPTIMIZING PARALLEL PREFIX ADDERS FOR USE IN DIGITAL FILTERS

V. M. Ionescu[1], L. Ioan, D. Visan, B. Cioc
University of Pitesti, Faculty of Electronics, Communications and Computer Science, Romania
[1]valeriu.ionescu@upit.ro

Keywords: parallel prefix adder, digital filter, optimization

Abstract: *Digital filters can be implemented using FPGAs in order to benefit from their architectural advantages by using registers, multipliers and adders. This paper proposes methods for using the modular structure of parallel prefix adders in order to increase the design flexibility of digital filters.*

1. INTRODUCTION

Digital filters are used in communications for filters in receiver and transmitters and improving the digital filter speed is recommended as these filters are slow compared to analog filters sometimes with a few orders of magnitude. Speeding the process would make digital filters available for new processing areas that were unavailable due to the lack of processing speed or high latency. [1].

Addition is an important part of the digital filtering process as a digital filter is composed of registers, adders and multipliers. Speeding the process of adding two numbers consists of finding as early as possible the carry that influences the higher order bits. Associative operations can be parallelized. This can be done by computing in parallel the operation's carry through segmentation of the operation in smaller pieces.

Usually digital filters need to implement a high level of parallelism in order to compete with analog filters. This means that operations use many bits and the circuits from the filter also need to process many bits in parallel (as serial operations give a high latency to the device).

Parallel prefix adder architectures are used in designing adders in excess of 32 bits that have a too increased design complexity for manual optimal design. Therefore it is important to design filters that can be used for low power and high performance while maintaining a as small as possible implementation area. Parallel prefix adders have a 3-tier architecture based on a limited number of modular structures:

-Tier 1: Preprocessing of the propagate and generate terms.

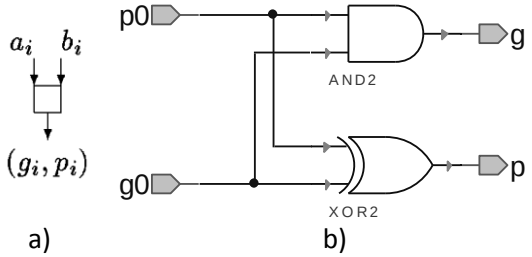

Fig. 1. Tier 1 - Preprocessing block: a) symbol; b) implementation

-Tier 2: Carry computation. This is the parallelizable part and is the most important and varies in implementation from one parallel adder model to another. It has the largest size of the three tiers. It includes the processing component for propagate and generate terms but also buffering components.

-Tier 3: Sum generation: This is a simple adder for the propagate and computed carry values and generate the final sum value.

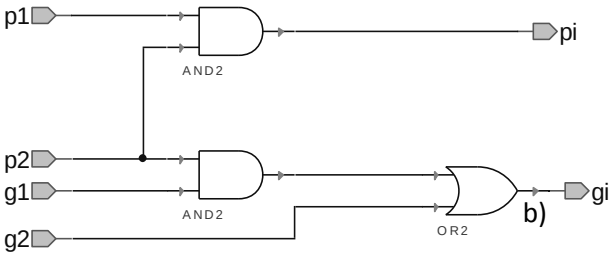

Fig. 2. Tier 2 processing component implementation that gives the generate (gi) and propagate (pi)

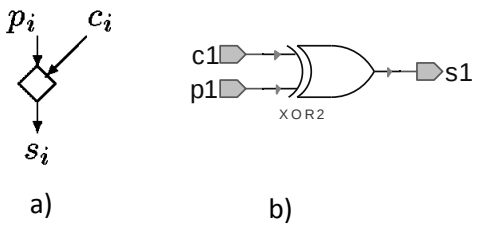

Fig. 3. Tier 3 - Postprocessing block: a) symbol; b) implementation

This paper analyses the gate re-use in architecture implementation for common parallel prefix adders that are used in digital filters.

2. PARALLEL PREFIX ADDER ARCHITECTURES

The differences in the architectures for parallel prefix adders target the tier 2 layer, parallel carry computation. The variations are related to: structure size; wiring complexity; fan-out and structure depth.

A classification was proposed [2] that visually represents the directions that these architectures were optimized.

This also helps us identify which architectures can be combined by reusing the design blocks in order to obtain different adder characteristics without having to include both adders on the same die.

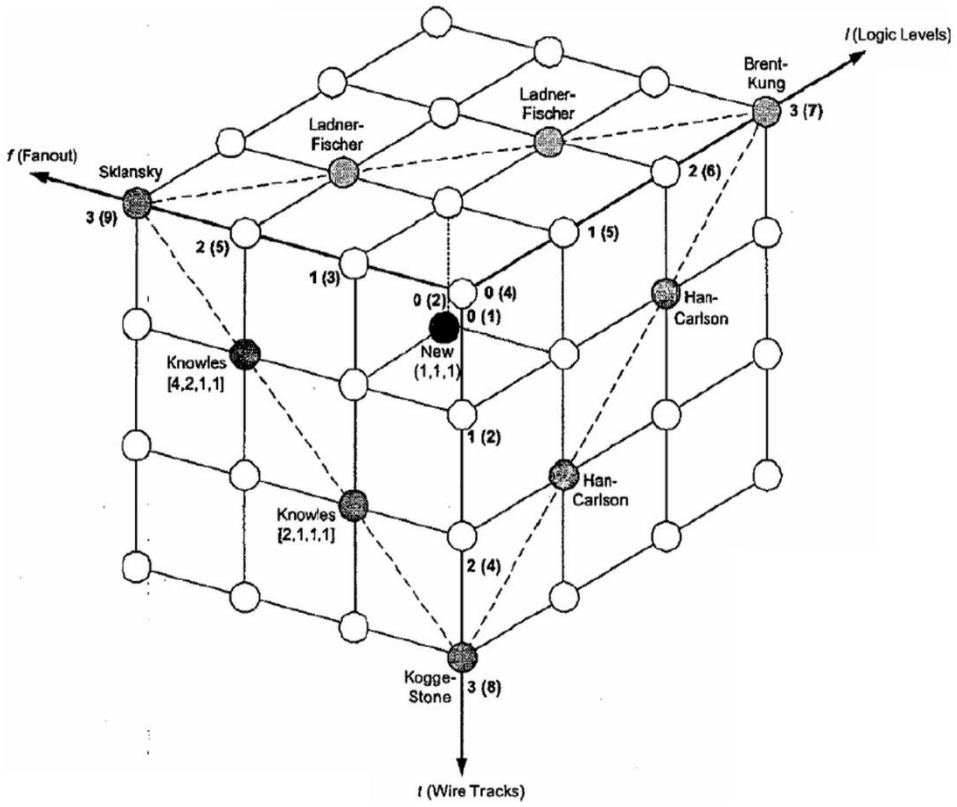

Fig. 4. Parallel prefix adder taxonomy that shows the 3 characteristics towards witch the architectures are optimized: fan-out, wiring length and logic depth [2]

3. BLOCK REUSE ANALYSIS FOR MIXED ADDER IMPLEMENTATION

The following adder architectures were analyzed: Ripple carry; J. Sklansky (proposed in 1960); Kogge-Stone (proposed in 1973); 1980: Ladner-Fisher adder (proposed in 1980); Brent-Kung adder (proposed in 1982); Han Carlson adder (proposed in 1987) and S. Knowles (proposed in 1999).

The Xilinx ISE implementations for some of these architectures are presented in Fig. 6 and 7.

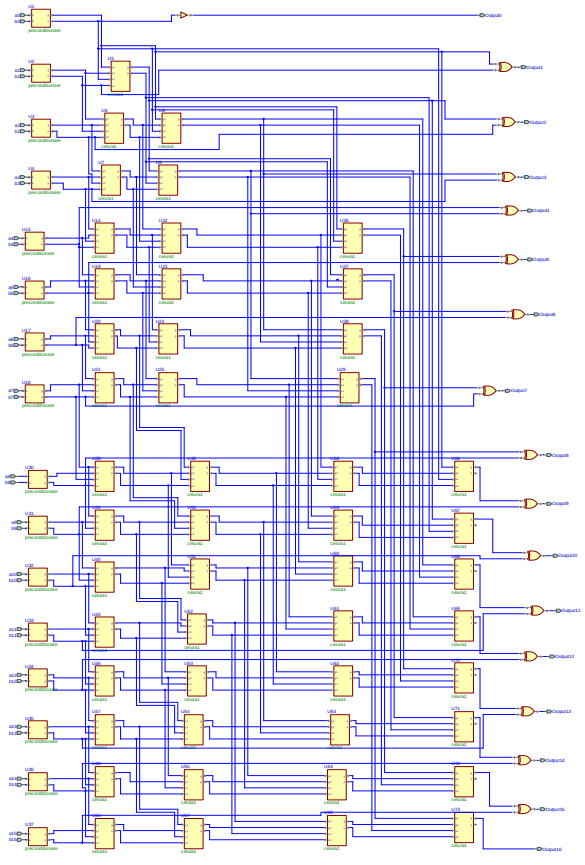

Fig. 6 Kogge-Stone adder, tier 2 implementation, 16 bits

By implementing in Xilinx ISE and analyzing the 16 bit adder implementations in these architectures, the following table presents the gate implementation of the tier 2 area of the parallel prefix adder (as Tier 1 and 3 are unaffected by the architecture change).

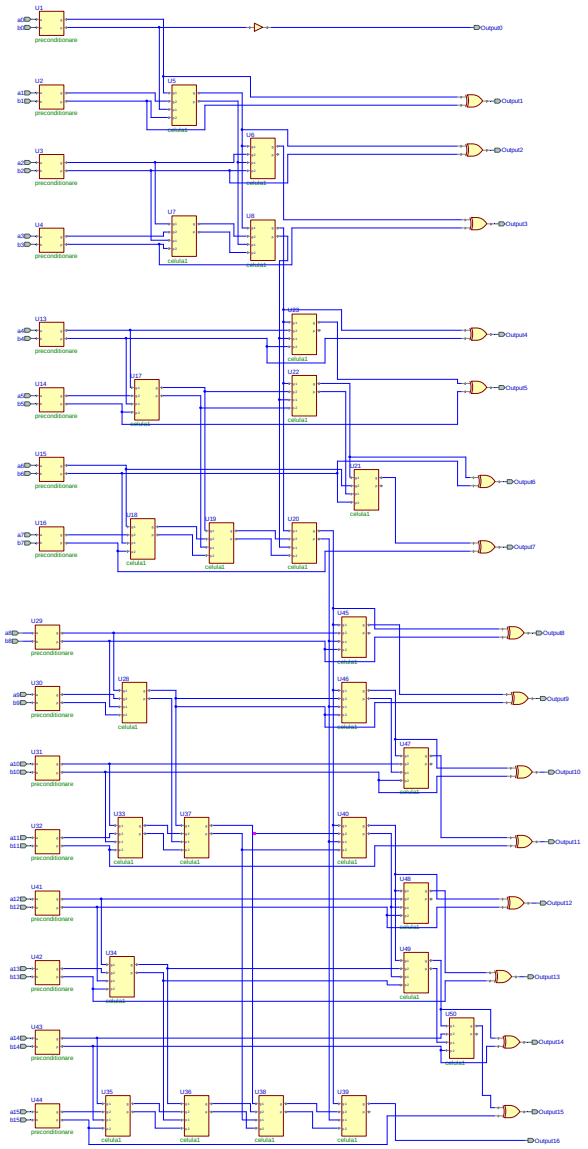

Fig. 7 Brent-Kung adder, tier 2 implementation, 16 bits

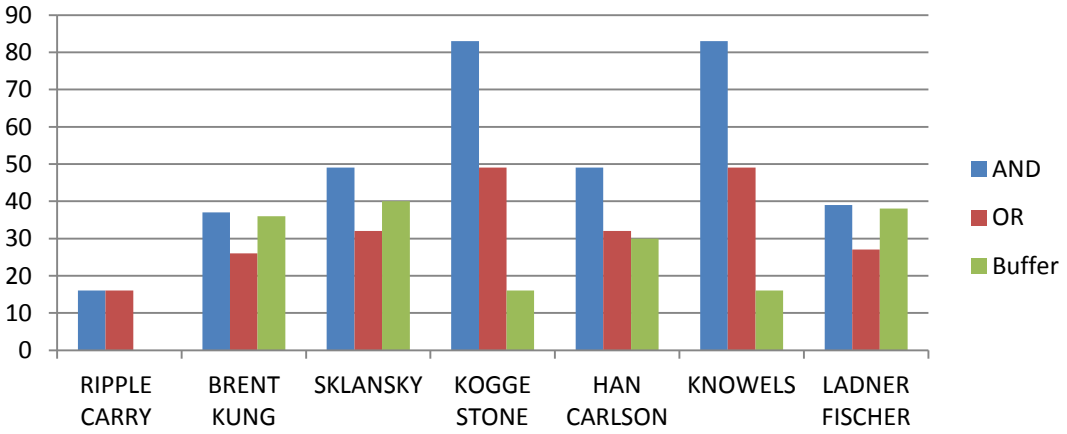

Fig. 8 Gate usage for different parallel 16 bit parallel adder architectures

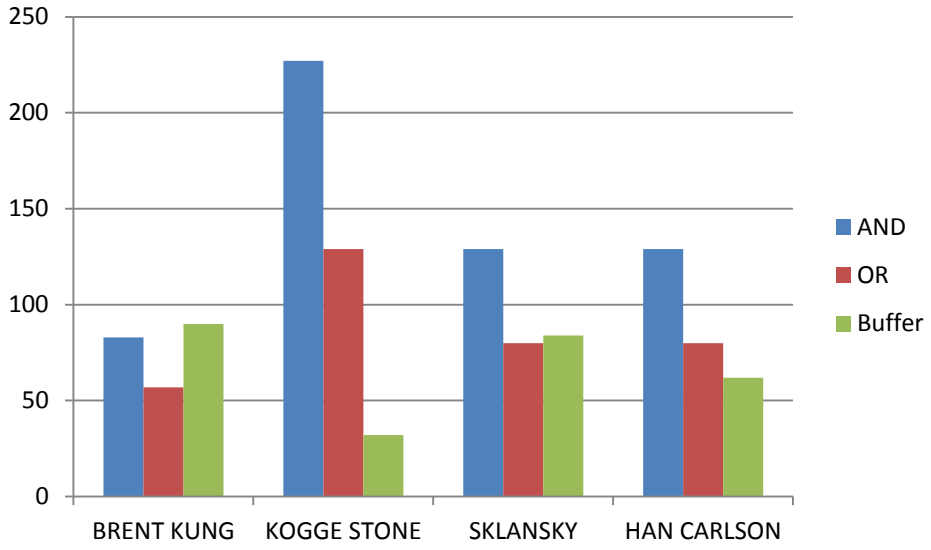

Fig. 9 Gate usage for different parallel 32 bit parallel adder architectures

Table 1. Gate count for the tier 2 structure, 16 bit adders

Adder type	AND gates	OR gates	Buffers
RIPPLE CARRY	16	16	0
BRENT KUNG	37	26	36
SKLANSKY	49	32	40
KOGGE STONE	83	49	16
HAN CARLSON	49	32	30
KNOWELS	83	49	16
LADNER FISCHER	39	27	38

Table 2. Gate count for the tier 2 structure, 32 bit adders

Adder type	AND gates	OR gates	Buffers
BRENT KUNG	83	57	90
KOGGE STONE	227	129	32
SKLANSKY	129	80	84
HAN CARLSON	129	80	62

The results show that it is easy to implement relatively with the same logical circuits different parallel prefix adder architectures.

For example it is easy to reuse the implementation blocks of the Sklansky adder to also implement the Brent-Kung adder. Similarly the Kogge-Stone adder, known for its high wiring, can reuse most of its blocks and implement the Brent-Kung adder that has minimal amount of wires required and the logical overhead is minimal consisting in the addition of buffer gates.

4. CONCLUSION

This implementation results shows that it is possible to implement different architectures on the same die, with the only supplementary implementation being the control circuit for switching the wire connectivity for the architectures.

This may create some latency that may impact circuit performance that will be analyzed in the future.

5. REFERENCES

[1]. Steven W. Smith - Digital Signal Processing: A Practical Guide for Engineers and Scientists 2002 ISBN 0-7506-7444-X, accessed 01.02.2012: http://www.dspguide.com/ch21/1.htm

[2]. D. Harris, "A taxonomy of parallel prefix networks", Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on (Volume:2), ISBN: 0-7803-8104-1, 2003