DeverSYS

FREE IP CORE OVERVEW

Hierarchical Carry Save Algorithm.
HCSA Generic ALU.
Free VHDL IP Cores.

Copyright © DeverSYS
www.deversys.com

Hierarchical Carry Save Algorithm

It is known 3 possible adder implementations: **ripple carry**, **carry save** and **carry look ahead**. Here we propose a kind of carry save algorithm: “hierarchical carry save algorithm” (HCSA) and compare its properties with Synopsis’s DW01 ‘cla’(carry look ahead) adder implementation.

The idea of the algorithm is illustrated on Fig.1. Carry from low part of sum is used for choice of 2 possible high part sum results. This approach is used on every hierarchy level.

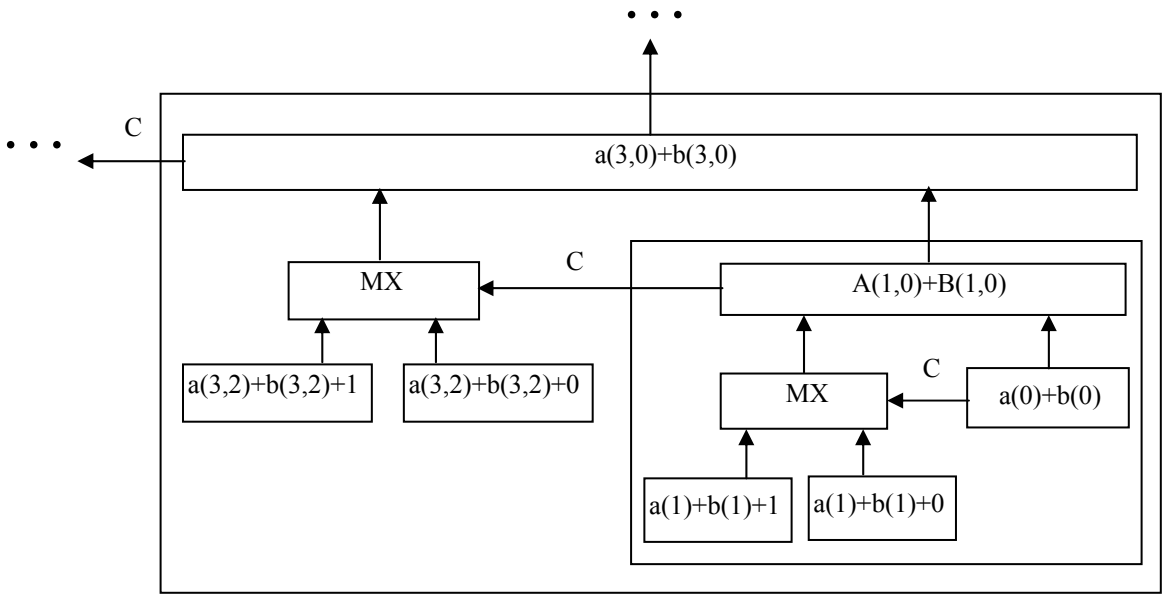

Fig.1. HCSA structure.

Implementation Results

Table 1 and Chart 1 represents HCSA synthesis and simulation results in comparing with standard Synopsis's DW01 'CLA' adder implementation (0.35u STD cell library).

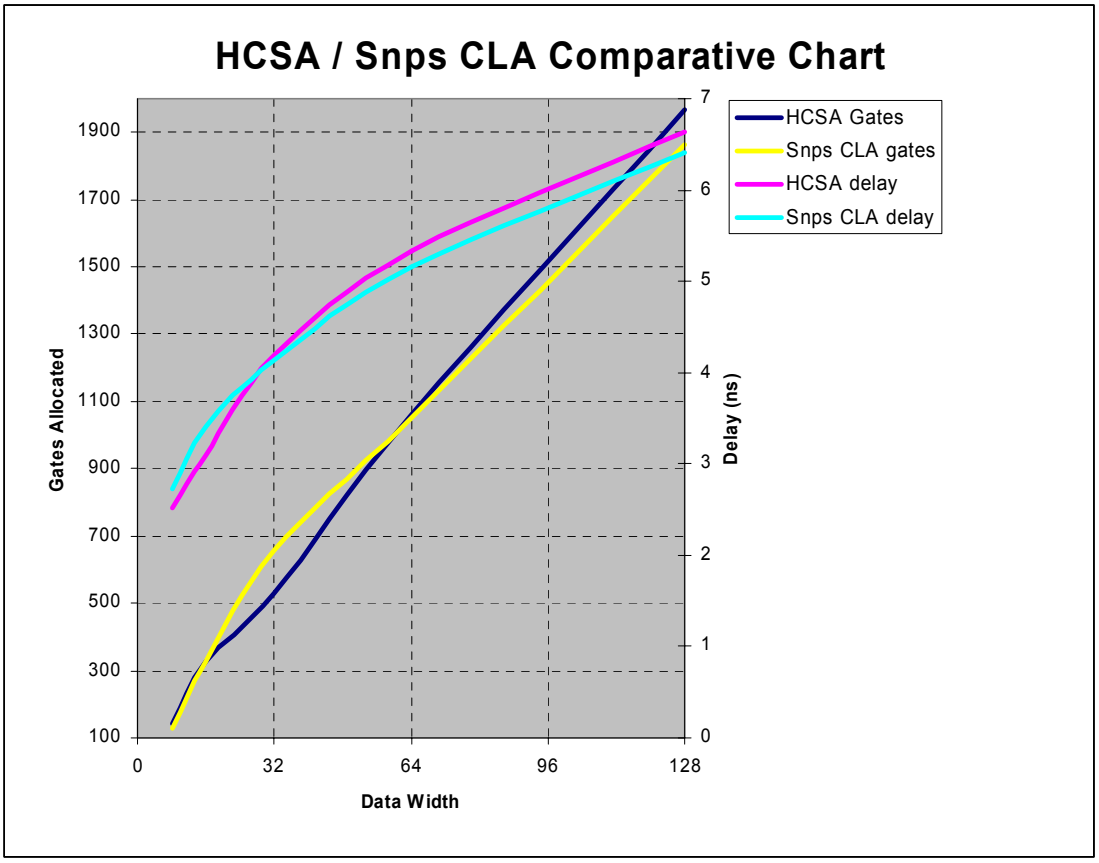

Chart 1.

	HCSA		Synopsis's 'cla'	
Data Width	**gates**	**delay**	**gates**	**delay**
8	143	2.51	127	2.72
16	327	3.09	327	3.40
32	527	4.18	655	4.14
64	1061	5.34	1053	5.16
128	1965	6.64	1865	6.41

Table 1. Synthesis and post synthesis simulation results.

Generic ALU based on HCSA adder

HCSA may be used within the different kind of tasks It behaves effective also for processor core parts. For example Generic ALU can be implemented applying HCSA methodology.
Basic idea is every **ALU bit** implemented as a combinational part and performs either logic or arithmetic operations. Since **bit carry/bit stealing** exists only in arithmetic operations (“+” or ”-“) the HCSA method can be also applied for the ALU bit. (for more details see ALU implementation).

In other words ALU structure is exactly the same as HCSA adder except missed logic parts there.

Table 2 represents ALU_HCSA (ALU with hierarchical carry save algorithm) implementation details (synthesis process: 0.35u library, worst case military conditions).

Data width	delay (ns)	combinational area (gates)	non-combinational area (gates)
8	4.70	230	235
16	5.66	385	400
32	6.99	800	715
64	8.39	1460	1345

Table 2. HCSA ALU Synthesis results.

Theme for debates.

HCSA ALU delay is greater than HSCA adders one, the reason is that ALU bit is more complex than adder one. Hardware expenses (combinational part) are a little bit greater in comparison with HCSA adder, because instruction decoder is included into combinational part additionally to ALU itself.

Please pay attention that delay grows proportional to data width logarithm as algorithms theory expected both for adder and ALU_HCSA cases.

Of course instruction decoder may be implemented using different ways, may be more effective. But the goal of this part is to present complete ALU only.