A Low Power 32-bit Quaternary-Tree Adder

Nadida Raad, Mohammad M. Mansour
Electrical and Computer Engineering Department
American University of Beirut
Beirut 1107 2020, Lebanon
{ngr04, mmansour}@aub.edu.lb

Abstract—In this paper, we implement a 32-bit quaternary tree adder and simulate the design using HSPICE in 90-nm static CMOS logic. Simulation results show that the adder can operate at 3.4 GHz (measured at 1.8V, 25°C) and dissipates 8.9 mW of power. The sparse-tree architecture achieves low carry-merge fan-outs and inter-stage wiring complexity. Compared to other designs in the literature, the quaternary tree adder has fewer carry-merge blocks and faster carry generation after removing the sum generation from the critical path.

Keywords—component; tree adders; quaternary adders, low-power, ALU

I. INTRODUCTION

FAST binary adders with single cycle latency are essential to modern digital computers. Because of their high switching activity, they are considered as thermal hotspots and peak-current limiters in current microprocessors. These constraints motivate the need of high-performance and energy-efficient adders. For decades, carry look-ahead adders [3] have been the popular choice of fast parallel binary adders. The conditional-sum adders [4] are another important class of adder design. Conditional-sum adders suffer from fan-out limitation since the number of multiplexers that need to be driven increases exponentially. A modification of conditional-sum adders have been presented in [5]. These adders are called conditional carry adders since the conditional-sum adder principle applies to only the carry generation circuit. An embedded tree structure was introduced in [6] such that the number of fan-out for each multiplexer is a constant internally. In high-speed adder designs, fan-out limitation may seriously degrade the estimated addition speed. In this paper, we implement a 32-bit quaternary tree adder in Verilog and provide a full study of power vs. delay with previous designs through HSPICE simulations.

II. ADDER ARCHITECTURE

Figure 1 shows the main architecture of the 32-bit quaternary adder. It is divided into two main components: the quaternary-tree and the sum generator(s). The quaternary tree takes the two 32-bit inputs and generates 1 in 4 carries in parallel, i.e. C3, C7, etc., ...C31, which feed into the non-critical side path sum generator to produce the sum resulting in a 73% fewer carry-merge gates [7].

A. Sparse-tree Adder Design

The performance setting and power limiting block in prefix adder is the carry merge block. In order to build very fast adders, it is necessary to organize propagation and generation signals into sparse-tree. It is this tree that selects the appropriate sum from the conditional sum generator (Fig. 2).

The first stage of the sparse tree is the Propagate/Generate block that outputs propagate (P = A# AND B#) and generate (G = A# NOR B#) signals from the adder inputs A# and B#. Complementary version of the adder’s inputs are available at the front-end multiplexer. The use of complementary adder inputs results in single-transistor pull-up (GP = P NAND P1) and pull down (GP = P NOR P1). The critical path of the
adder’s tree (Fig. 3) is improved by retaining the same number of gates as the Kogge-stone architecture [1] with reduced fanouts/stage and reduced interconnect delay between stages. Moreover, the sparse-tree has 73% fewer carry-merge gates with generate/propagate fanouts of 1 and 2 respectively, on the majority of carry-merge gates.

Consequently, the critical path reduces to a pruned carry-merge tree with 33/50% reduction in P/G fanouts per stage and 25% reduction in maximum inter-stage interconnect length.

B. Conditional Sum Generator

In parallel with the Quaternary tree, 4-bit sum generators speculatively generate conditional sums corresponding to a carry-in of 0 or 1. The non-criticality of the sum generator permits the usage of a ripple carry-merge scheme to generate the conditional carries. Thus, as shown in Fig. 4, the carry in at the first level of each conditional carry rail is tied to 0 and 1 respectively, generating 2 rails of conditional carries. An XOR of the partial sum with the conditional carries generates the conditional sums. The carries from the sparse-tree (C3, C7,…C23 and C27) then select the appropriate 4-bit conditional sums using a 2:1 multiplexers, delivering the final 32-bit sum. In this way, logic traditionally implemented in the main carry-tree using expensive parallel prefix logic is implemented in the sparse-tree design using an energy-efficient architecture. Such an approach results in smaller area, reduced energy consumption and lower leakage.

III. DESIGN BENEFITS AND MEASUREMENTS

The 32-bit adder was implemented in Verilog, and the simulations were done using HSPICE, in 90-nm static CMOS logic. The features of lower dynamic power consumption and higher noise margin make static CMOS particularly attractive. Fig. 5 shows the simulation results for the static CMOS implementation of the adder.

IV. CONCLUSION

The design of an energy efficient 32-bit quaternary tree adder operating at 3.4 GHz in a 1.8 V, 90-nm CMOS technology consuming 8.9 mW total power has been described. This mitigates the power density issues and thermal hotspots challenges within the execution core, increasing reliability and reducing cooling costs.

ACKNOWLEDGMENT

This work was supported by Intel’s Middle East Energy Efficiency (MER) Research.

REFERENCES