

A High Speed Carry Propagate Adder in a 1.5μ Process

Matthew Aldrich

Yale University Department of Electrical Engineering
matthew.aldrich@yale.edu

Abstract – a high speed Ling-Naffziger 8b adder has been designed and simulated in a 1.5μ process using dual rail domino logic. The adder is comprised of 640 transistors. An 8b pre-layout simulation yields a full add in 4 FO4 delays at a frequency of 333MHz with a dynamic power consumption of 200mW at 5V. (FO4 1.5μ ≈ 715ps)

I. INTRODUCTION

Addition time is critical to any CPU design. In most cases, adders occupy the critical path in many key areas of microprocessor design. A special family of adders are known as *tree adders*, or *logarithmic adders*. These adders improve speed on the critical path by looking across blocks.

This paper assumes that the reader is familiar and comfortable with the 64b Naffziger adder in [1] and in [2]. A simple and introductory 8b treatment of this type of adder, complete with simplified logic and schematics, can be found in [3]. In this paper, logic levels will only be given if needed.

The goal of this paper and simulation is two fold, 1) to verify the maximum performance of this adder configuration (no wire capacitance), that is create a well documented simulation to quickly and easily verify the domino logic transistor widths and 2) to analyze the performance of this adder in different processes as well as different logic families.

II. ADDER SIMULATION ARCHITECTURE

The simulation consists of 7 blocks which comprise the logic levels in the adder. By using Ling's equations [4], bitwise generate, propagate, and kill logic is simplified. These signals are then recursively combined to create the group generate, propagate, and kill signals. By using a pseudo carry signal that allows the adder to computer both possibilities of a signal, LSB are quickly available to higher bits. Finally the sum select gate computes the sum. Because the adder uses dual rail domino, complimentary signals are required.

Using Orcad Capture 9.1, the individual cells were drafted and verified using PSPICE. The cells were then made into sub-circuits and comprised the adder layout. In addition, a logic checker drafted in Excel allowed for easy debugging. A full 8b add is achieved in 3ns.

III. PERFORMANCE ESTIMATES

While performance estimates currently do not include interconnect, estimates can be given for adder performance at greater bit widths, different logic levels, and in different processes.

According to [2] the critical path in a 16 bit block is through the 11bit group generate. This is due to the carry

chain comprised of four 4bit H and I blocks. In a 32 bit design the critical path will be through the 27bit. However, to create 32 bits, the recursively combined H and I blocks must span 8 4bit H and I blocks. In the 8b simulation, the group generate only spans 7:0 rather than 15:0, so in this case the critical path is 4:0. It produces an output with a delay of 1.25ns. This is because the chain awaits an input from an earlier chain. Therefore in this simulation, we may expect that the 8bit and 12bit in the 16b design would yield the greatest delay.

A conservative estimate for the 16bit H and I signals is about .3ns of delay. The inputs from those blocks receive their results directly from the GPK cells. Reserve roughly .5ns for 8:0 and .5ns for 12:0. The final delay is added by the sum select gate, where in this case, produces a response in roughly .3ns. At this point the clocking period is roughly 4.6ns, an FO4 delay of 7. A worst case estimate would suggest that a 32b is computed with a delay of 7ns, an FO4 delay of 10. Table 1 shows the FO4 delays for both a 16b and 32b Naffziger adder. Wire capacitance can be estimated by adding 10% of the total delay to the adder. A comparison of this adders performance in Intel's .5μ process (FO4 delay: 140ps) is also given. Using the approximation for τ -delay [2] as

$$\tau = .2 * FO4 \quad (1)$$

the delay in the Intel process can be found. The τ for the MOSIS 1.5μ is 143ps. This yields a normalized delay of 21 (3ns/143ps). The τ in the Intel .5μ process is 28ps. By multiplying the Intel delay (28ps) by the normalized MOSIS delay, the delay of the MOSIS adder scaled to Intel's process is found. Intel's published 64b design [1] has an FO4 delay of approximately 7 which corresponds to a delay of roughly 930ps. Why does this design fall short?

The 16b and 32b are worst case estimates, they are based on the results of the 8b add with estimates of how the higher level logic performs. Therefore they are nothing but an educated guess.

It is also important to point out the shortcomings of the current simulation. Each cell in the library is based upon an individual child cell, for example, same carry schematic is used for bits 0:2 and for bits 4:6. In Intel's design, carry chains are individually crafted to ensure higher speed. In addition, the critical paths in this simulation are different than the critical paths pointed out in [1],[2]. In [5] Harris reports that domino logic can offer a 30% speedup (valency-2) versus a static logic implementation. In this case, the adder supports a larger look ahead (valency-4) and even reduces one stage of GPK generation by employing Ling's equations [4]. The percent speedup may be even greater in this case. In addition, the cell layout was done without clocked nmos transistors (footed design). The cells, as in [3], may need to be footed to ensure proper performance.

Table 1 – Delay in Multiple Processes (ns)

	Mosis 1.5 μ			Intel .5 μ		
	8b	16b	32b	8b	16b	32b
w = 0	3.0	4.6	7.0	.59	.90	1.3

IV. NEXT GENERATION TECHNIQUES: LVS

Intel's third generation Itanium design centers its ALU core around a different type of logic, LVS. In their 90nm process, Intel has achieved clocking rates 7Ghz [6]. The idea behind LVS logic is the reduction of the supply voltage to save power. However, as V_{dd} is scaled lower the delay of the circuit is made greater. To compensate the threshold voltage (V_t) is scaled as well [7]. LVS circuitry employs a form of pass transistor logic however, at the end of an operation a "sense amp" amplifies the signal. Basic GPK and Summing XOR are listed in [7]. These form the basis for LVS adder design.

V. CONCLUSIONS

A working pre-layout simulation of a valency-4 parallel prefix adder has been constructed and verified. Performance estimates of larger bit widths have also been estimated. In addition, this paper scales this adder's performance with that of Intel's 64b adder of the same design. Next generation logic techniques with application to integer operations have been reviewed and listed.

REFERENCES

- 1 S. Naffziger, "A subnanosecond 0.5 m 64b adder design," *Intl. Solid-state Circuits Conf.*, 1996, pp. 362-363.
- 2 N. Weste and D. Harris, *CMOS VLSI Design*, Addison-Wesley, 2004, pp. 350-353.
- 3 M. Aldrich and H. Waterford, "High Speed Adder Design: Constructing an 8b Naffziger Adder," 2003, pp. 1-13.
- 4 H. Ling, "High Speed Binary Adder," *IBM J. Research.*, vol. 25, no. 3, May 1981, pp. 156.
- 5 D. Harris and I. Sutherland, "Logical Effort of Carry Propagate Adders," *IEEE Transactions on Computers.*, 2003, pp. 873-878.
- 6 D. Delehanes *et al.*, "LVS Technology for the Intel Pentium 4 Processor on 90nm Technology." *Intel Technology Journal.*, vol. 8, no. 1, Feb. 2004, pp. 49-59.
- 7 T. Sakurai *et al.*, "Low Power CMOS Design through Vth Control and Low Swing Circuits," *Int. Symp. Low Power Electronics and Design.*, Aug. 1997, pp. 1-6.

ABOUT THE AUTHOR

Matthew H Aldrich is currently a senior in the Electrical Engineering Department at Yale University. He will receive an EE B.S. (ABET) degree in May 2004. He is advised by Richard Lethin. His interests are in mixed signal design and high speed digital circuits. He is fan of electronic music and enjoys swimming in his spare time. He is currently unemployed.

