Design of High Speed Digital CMOS Comparator Using Parallel Prefix Tree

N. Prasanna¹, H. Sumitha²

¹Department Of ECE, Srinivasa Ramanujan Institute Of Technology : Anantapuramu
²Assistantprofessor, Department Of ECE, Srinivasa Ramanujan Institute Of Technology : Anantapuramu

Abstract: This paper Presents a new comparator design is proposed by using parallel prefix tree. Energy efficient and high speed operation of comparators is needed for high speed digital circuits. The comparison outcome of the most significant bit, proceeding bitwise toward the least significant bit only when the compared bits are equal. In existing system, the parallel prefix structure is designed for 16, 32 and 64 bit architectures and the reports from the Xilinx tool concludes that for every bit range doubles the delay, memory, LUT and power has not doubled up to the mark. But In the proposed design of my project, each and every element in the parallel prefix structure will be replaced by universal logic (multiplexer). By performing this modification in the architecture will leads to reduction in POWER CONSUMPTION and DELAY.

Keywords: Parallel prefix tree structure high fan in, high fan out, Bitwise competition logic(BCL).

1. Introduction

A high speed comparator is a very basic and useful arithmetic component of digital systems. Comparator is a major fundamental element in most digital circuits. The main advantages of this design are high speed and power efficiency, maintained over a wide range. Comparators are key design element for a wide range of applications like parallel testing, signature analyzer, built- in self- test circuits, graphics and image/signal processing[1]-[3]. The design of high-speed, low power, and area-efficient comparators. Comparison is a fundamental operation digital processors.

Comparator designs improve scalability and reduce comparison delays using a hierarchical prefix tree structure composed of 2-b comparators [17]. These structures require log2N comparison levels, with each level consisting of several cascaded logic gates. However, the delay and area of these designs may be prohibitive for comparing bit operands. The prefix tree structure’s area and power consumption can be improved by leveraging two-input multiplexers(instead of 2-b comparator cells) at each level and generate-propagate logic cells on the first level (instead of 2-b adder cells)[18].

Furthermore, the structure can perform only “greater-than” or “less-than” comparisons and not equality. To improve the speed and reduce power consumption[20],[21] several designs rely on pipelining and power-down mechanisms to reduce switching activity with respect to the actual input operands’ bit values. A 64-b comparator requires only three pipeline cycles using a multiphase clocking scheme[23]. However, such a clocking scheme may be unsuitable for high-speed single-cycle processors because of several heavily loaded global clock signals that have high-power transition activity. Additionally, race conditions and a heavily constrained clock jitter margin may make this design unsuitable for wide-range comparators. An alternative architecture leverages priority encoder magnitude decision logic with two pipelined operations that are triggered at both the falling and rising clock edges[24] to improve operating speed and eliminate long dynamic logic chains. Other architectures use a multiplexer-based structure to split a 64-b comparator into two comparator stages[25]: the first stage consists of eight modules performing 8-b comparisons and the modules’ outputs are input into a priority encoder and the second stage uses an 8-to-1 multiplexer to select the appropriate result from the eight modules in the first stage.

Similarly, other energy-efficient designs leverage schemes to reduce switching activity. Compute-on demand comparators compare two binary numbers one bit at a time, rippling from the most significant bit (MSB) to the least significant bit (LSB). The outcome of each bit comparison either enables the comparison of the next bit if the bits are equal, or represents the final comparison decision if the bits are different. Thus, a comparison cell is activated only if all bits of greater significance are equal. This scheme detects the larger operand by determining which operand possesses the leftmost 1 bit after pre-encoding before supplying the operands to bitwise competition logic (BCL) structure. The BCL structure partitions the operands into 8-b blocks and the result for each block is input into a multiplexer to determine the final comparison decision.

This structure consists of two basic modules: Comparison resolution module and decision module. The comparison resolution module divides two input N- bit arrays to be compared into two buses namely left bus and right bus each of N bits wide respectively. The decision module in turn decides whether equal, less than or greater than relationship exists between applied inputs for comparison. To overcome some of the drawback present in the above designs (such as higher power consumption, multi cycle computation, unsuitable custom structures for scaling, irregular VLSI structures, and irregular transistors), parallel prefix structure based comparator design provides fast, scalable, wide range, and power efficient algorithm.

Volume 4 Issue 10, October 2015
Figure 1: Block diagram of our comparator architecture, consisting of a comparison resolution module connected to a decision module.

The comparison resolution module is a novel MSB-to-LSB parallel prefix tree structure that performs the bitwise comparison of two N bit operands (A & B) entered into the comparator. The parallel structure encodes the bitwise comparison results to two N bit buses called left bus and right bus. The bitwise comparison of equal bits sets '0 in both the buses. If the bitwise comparison of unequal bits occur, any of the buses (A or B) sets to '1 and the bitwise comparison stops immediately by setting '0 in the remaining bits present in the buses. The decision module produces the result of comparison of the input operands based on the signals from the left and right buses. The possible results from the decision module are A=B, A>B, A<B.

2. Comparator Design Details

We partition the structure into five hierarchical prefixing sets, where as each set performs a exact function whose output serves as input to the next set, in hope of the fifth set produces the output on the left bus and the right bus Every part of cells components within each set operate in parallel were as it’s a key feature to increase operating speed while minimizing the transitions to a minimal set of left most bits needed for a correct decision. This prefixing set structure bounds the components fan-in and fan-out regardless of comparator bit-width and eliminates heavily loaded global signals with parasitic components, thus improving the operating speed and reducing power consumption.

3. Architecture of 16-Bit Comparator Using Parallel Prefix Tree

In this project the switching logic and the main block design is carried out by using mux logic to perform low power operations because In electronics, a multiplexer is a device that selects one of several analog or digital input signals and forwards the selected input into a single line. A multiplexer of 2n inputs has n select lines, which are used to select which input line to send to the output. A multiplexer is also
called a data selector. An electronic multiplexer can be considered as a multiple-input single output.

4. Advantages by Using Multiplexer Based in Parallel Prefix Tree

Low power consumption by replacing the needed logics by multiplexer, because multiplexer operates at very low power switching transitions compared to another logical gates. Low delay compared to normal based comparator, less area and less LUT compared to existing system.

5. Conclusion

In this project a Design Of High Speed CMOS Comparator Using Parallel Prefix Tree using regular digital hardware structures consisting of two modules: the comparison resolution module and the decision module. This regularity allows simple prediction of comparator characteristics for arbitrary bit widths and is attractive for continued technology Scaling and logic synthesis. These modules are structured as parallel prefix trees by using a normal flow.

Future work will include additional circuit optimizations to further reduce the power dissipation by adapting dynamic and analog implementations for the comparator resolution module and a high-speed zero-detector circuit for the decision module. Given that our comparator is composed of two balanced timing modules, the structure can be divided into two or more pipeline stages with balanced delays based on a set structure, to effectively increase the comparison throughput.

References


