Multipliers and Dividers: Insights on Arithmetic Circuit Verification (Extended Abstract) *

Randal E. Bryant

Carnegie Mellon University, Pittsburgh PA 15213, USA

Abstract. We consider methods for verifying multiplier and divider circuits based on symbolic function representations. Verification can be performed at either the bit-level, where individual signals are represented as Boolean functions, or at the word-level, where signal vectors are represented as "pseudo-Boolean" functions mapping Boolean variables to numeric values. These two classes of functions can be represented and manipulated as ordered Binary Decision Diagrams (BDDs), and Binary Moment Diagrams (BMDs), respectively.

It is impractical to verify multiplier or divider circuits entirely at the bit-level, because the BDD representations for these functions grow exponentially with the word size. It is possible, however, to analyze individual stages of these circuits using BDDs. Such analysis can be helpful when implementing complex arithmetic algorithms. As a demonstration, we show that Intel could have used BDDs to detect and even correct erroneous table entries in the Pentium floating point divider.

Abstracting to a word level offers two advantages over bit-level verification. First, it allows much more abstract and concise specifications in terms of arithmetic expressions. Second, we can verify complete multiplier circuits in polynomial time. Future extensions promise to enable word-level verification of divider circuits, as well.

Much of the research in formal hardware verification has focussed on systems having complex control, but relatively simple data operations. When verifying such systems, the desired behavior is specified as a set of desirable properties, e.g., in the form of temporal logic formulas, rather than as a complete specification of the functionality. Tools such as symbolic model checkers [5] have achieved notable success for this form of verification.

Arithmetic circuits have just the opposite characteristics—their complexity arises from the extensive data manipulation, while the control is relatively straightforward. Furthermore, one can give a precise specification of the desired functionality in terms of arithmetic expressions. This class of circuit has not received as much attention from the verification community, except by those using methods based on theorem proving, e.g., [8]. This inattention is due to two main reasons. First, many perceive that arithmetic circuit design is fairly straightforward—the same implementation techniques

* This research is sponsored by the Wright Laboratory, Aeronautical Systems Center, Air Force Materiel Command, USAF, and the Advanced Research Projects Agency (ARPA) under grant number F33615-93-1-1330.

have been used for years, and designers are confident of their ability to detect errors using conventional simulation. Intel's recent experience with its Pentium floating point divider[7] has exposed the error in this thinking. There are many places one can make mistakes in designing these circuits, some of which may be very hard to detect with the limited number of cases that can be tested by simulation. Second, these circuits are particularly troublesome for methods based on ordered Binary Decision Diagrams (BDDs), the most popular alternative to theorem proving [3]. The BDDs representing the outputs of a multiplier grow exponentially with the word size [2], making them impractical for word sizes much beyond 16 bits. Other arithmetic functions such as division also seem to be intractable using BDDs, although this has not been proved formally.

We describe recent progress in verifying arithmetic circuits, particularly multipliers and dividers. Using BDDs, one can analyze circuits at the *bit level*, i.e., representing each signal as a Boolean function. Even though it may be impractical to represent the complete circuit using BDDs, useful results can be obtained for individual circuit stages. We demonstrate this by showing the desired behavior for one iteration of radix-4 SRT division [1], as used in the Pentium divider, can be specified and verified using BDDs. This verification will detect incorrect table entries such as occurred in the Pentium, and can even be used to generate the correct values.

Our analysis of an SRT divider exposes a second shortcoming of bit-level analysis, namely generating a bit-level specification of the desired behavior. For our analysis, we constructed logic circuits to check range constraints on stage inputs and outputs, and to check the functional relation between the inputs and outputs. These circuits are as complex as the logic for the actual stage, and there is no good way to verify their correctness.

Abstracting the functionality to a word level allows more abstract specifications and makes it possible to verify complete arithmetic circuits. With this approach, we view a set of signals as forming a "word," having a numeric interpretation, e.g., according to a two's complement integer representation. The specification of a circuit consists of definitions of the word encodings for the inputs and outputs, and arithmetic expressions describing the intended functionality. The word-level operation of a circuit can be represented symbolically as "pseudo-Boolean" functions mapping Boolean values (corresponding to individual signals), to numeric values (corresponding to signal vectors). Such an approach was formulated by Lai and Vrudhula [6], where they used Edge-Valued Binary Decision Diagrams (EVBDDs) to represent this class of function. We have recently shown that Binary Moment Diagrams (BMDs) can also be used, having superior performance for representing functions such as word-level multiplication [4].

References

1. D. E. Atkins, "Higher-radix division using estimates of the divisor and partial remainder," *IEEE Transactions on Computers*, Vol. C-17, No. 10 (October, 1968), pp. 925–934.
2. R. E. Bryant, "On the complexity of VLSI implementations and graph representations of Boolean functions with application to integer multiplication," *IEEE Transactions on Computers*, Vol. 40, No. 2 (February, 1991), pp. 205–213.

3. R. E. Bryant, "Symbolic Boolean manipulation with ordered binary decision diagrams," *ACM Computing Surveys*, Vol. 24, No. 3 (September, 1992), pp. 293–318.
4. R. E. Bryant, and Y.-A. Chen, "Verification of arithmetic circuits with binary moment diagrams," *32nd Design Automation Conference*, 1995.
5. E. M. Clarke, J. R. Burch, K. L. McMillan, and D. L. Dill, "Sequential circuit verification using symbolic model checking," *27th Design Automation Conference*, June 1990.
6. Y.-T. Lai, and S. Sastry, "Edge-valued binary decision diagrams for multi-level hierarchical verification," *29th Design Automation Conference*, June, 1992, pp. 608–613.
7. H. P. Sharangpani, M. L. Barton, "Statistical analysis of floating point flaw in the Pentium processor(1994)," Intel Technical Report, Nov. 30, 1994.
8. D. Verkest, L. Claesen, and H. DeMan, "A proof of the nonrestoring division algorithm and its implementation on an ALU," *Formal Methods in System Design*, Vol. 4, No. 1 (January, 1994), pp. 5–32.