International Journal of Research in Engineering, Science and Management
Volume-2, Issue-3, March-2019
www.ijresm.com | ISSN (Online): 2581-5792

A Literature Survey on Hardware Multiplication and Hardware Division Algorithm

Prem Kumar[1], B. Hyvallika[2], S. Vyshnavi Jyothi[3]
[1]*Assistant Professor, Department of ECE, Saveetha School of Engineering, Chennai, India*
[2,3]*Student, Department of ECE, Saveetha School of Engineering, Chennai, India*

***Abstract*: The computers can have made benefit for a greater investment in hardware to mechanize multiplication and division. Most normal division is suitable for serial and will not be suitable for highly parallel multiplier. Only 3 iterative steps are needed to produce a 40-bit reciprocal. The advantages of using the approximate multipliers is twofold. Multiplication execute lager time and to a lesser extent division. The algorithm is based on montgomery method for multiplication and division. The modular multiplication algorithms are modified and combined in all most all hardware computation. Division algorithm have been reducing latency and to improve the computational efficiency, area, hardware cost and power of the processor. This paper compares the different division and multiplication algorithms.**

***Keywords*: modular multiplication, modular division, montgomery method, FPGA technique.**

1. Introduction

Arithmetic unit of such a machine computation is necessary for the multiplication and division. peripheral equipment and controls may be advantageous to the economy of the machine to increase the hardware investment in the operation of multiplication and division. Many algorithms have been developed for division implementation in hardware algorithm. The algorithms are based on different aspects as quotient converge rate, fundamental hardware primitives and mathematic formulation. Multiplicative methods use hardware integrated with the floating point multiplier and have low to moderate latencies. Hardware designers frequently perceive divisions as infrequent, low priority operations and they allocate design effort and chip resources. The hardware architecture for the unified divider/multiplier that implements the UDMA efficiently supports all the computations. The simplest and widely used and implemented class of division algorithm is digit recurrence. Normally division algorithm consists of divisor, quotient and remainder. Peripheral equipment and controls may be advantageous to the economy of the machine to increase the hardware investment in the operation of multiplication and division. Computers have evolved rapidly since their creation of multiplication and division.

2. Literature survey

Arithmetic unit of such a machine computation is necessary for the multiplication and division. Fast multiplication is possible. Modular multiplication is hard. High speed up processing. Hardware algorithms are redundant. Highly power consuming. Highly accurate iterative computation. Multiplicative methods use hardware integrated with the floating-point multiplier and have low to moderate latencies, while subtractive methods generally employ separate circuitry and have low to high latencies [1]. Method for multiplying two integers. Modulo N while avoiding division by N. Useful for several computations. Addition and subtraction are unchanged. Multiplication Speed modular. Required extensive modular arithmetic. Addition algorithm is unchanged. Time consuming process. Representation is normally used. Many stages are there [2]. Numbers are representing in redundant. Modular additions are performed. There are no propagations. Suitable for VLSI implementation. Require additional processing. Additional processing is negligible several modular multiplications have to be perform. More on squaring and multiplying large integers [3]. Speed up processor. Requires clock cycle. High radices are used. Loop are reduced. Addition algorithm is unchanged. Time consuming process. Representation is normally used. Many stages are there [4]. Methods used that are faster than FFT. Involves the simplest methods of multiplication. C and assembly language is used. Integer also written in assembly. Discussion on squaring and multiplying. Methods are quite large numbers. No presence of iterative procedure. Problem is how to find the best ways [5].

Highly accurate initial approximations. High operating frequency. Independent of floating point. An iterative implementation. Does not use logarithmic approximations. Loss of accuracy. The error in multiplication [6]. Montgomery multiplication speed up multiplication. High speed. Space efficient algorithm. Analyze time and space requirement. High memory storage. SOS and CIOS methods are used. Time consuming tasks. CIOS operates faster than other montgomery multiplication. Many stages are there [7]. Improving the efficiency. Less clock rate. The implementation is done on FPGA. Length is intermediate result. Several implementations are used. Implementation is difficult [8]. A simple and efficient logarithmic multiplier. coding by using VHDL for the FPGA. MATLAB is used in kernel values. many algorithms are used. Coding is difficult. Programming can be used. Algorithms is difficult [9]. Based on standard cell methodology. Increase 20% area. Low latency for division. Good area performance. High performance of floating point. Power consumption. Size is large. Chip area [10].

Implementing the modular exponentiation in RNS. Computation time is reduced. RNS has benefits in large numbers. Execution rounds are decreased by 33%. Extension technique algorithm is implemented. Does not use logarithmic approximations. Loss of accuracy. The error in multiplication result [11]. Fast montgomery algorithm. To implement modular design. Time consuming for large operands. Dominant part of the computation. Computation is high. Demand for development [12]. Hardware complexity and latency. Based on data flow. Algorithms that are fast. Suitable for power sensitive environment. Limitation in chip. Power consumption. size is large. Chip area [13]. Improving the efficiency. Less clock rate. The implementation is done on FPGA. Length is intermediate result. Full address, save address are used. 1024 bit RSA exponentiation is used. Modular multiplication can be done. We cannot use the subtraction. Slow clock cycles. High power consumption [14]. Suitable for VLSI implementation. Algorithm is based on montgomery method. Used to avoid carry propagation. Length is independent of N. Uses shifts additions and subtraction. Need for managing. Large amount of computation. Demand for implementation [15].

Throughput is one. Modular multiplication for every clock cycles. Used in RSA cryptosystem. Security for electronic banking transaction. Reduce latency. Multiplication of more than 500 bits done. Suffers from latency. Slow clock. High power bits to be used [16]. Implement high speed FPGA address. Improve the performance. Extensive experiments are done. 70% speed up. Low power consumption. High accuracy. Increased efficiency. High power consumption. Slow clock rate. Low performance rate [17]. Implementing the modular exponentiation in RNS. Computation time is reduced. RNS has benefits in large numbers. Execution rounds are decreased by 33%. Extension technique algorithm is implemented. Less storage requirements. High power consumption. Additional processing is negligible [18]. Enabling achievement of arbitrary accuracy. Uses the Mitchell's algorithm. Error % is small as required. Hardware involves address and shifter. Less power consuming. Parallel circuits are used. Does not use logarithmic approximations. Loss of accuracy. The error in multiplication results [19]. Algorithms are simple. Suitable for hardware implementations. Consumers less area. Used in digital communications. Barrette reduction. Large prime field. Efficiency is challenging. Time consuming process [20].

Highly accurate initial approximations. High operating frequency. Independent of floating point. An iterative implementation. Errors in the result. Highly accurate. High latency [21]. A simple and efficient logarithmic multiplier. coding by using VHDL for the FPGA. MATLAB is used in kernel values. many algorithms are used. Achieve arbitrary accuracy. Simulation is done using Xilinx. Iterative procedure. Errors in the result. Time and power consuming operation. Loss of accuracy [22]. Reduction in clock cycles. Maintain critical path delay. Verilog HDL and FPGA are used. Low power consumption. Less storage requirement. Better bandwidth utilization. High power consumption. Iterative accuracy. Coding is difficult [23]. Developed to reduce latency. Improve the computational efficiency. Hardware cost is high. Area and power of processor. High latency operations. High power consumption. High operating frequency [24]. Development of logarithm architecture. Design of logarithmic architecture. Low power consumption. 86% of data processing time. Solution for hardware efficient. Fast multiplication operation. FXP and FLP multipliers. we cannot achieve arbitrary accuracy [25].

3. Conclusion

In this paper we discussed about hardware multiplication and hardware division by using montgomery method. Several techniques are done for the hardware multiplication and hardware division like FPGA technique, FXP and FLP multipliers. The drawbacks of this techniques Less power consuming. Parallel circuits are used. Does not use logarithmic approximations. Loss of accuracy. The error in multiplication results. High power consumption and accuracy can be developed by using FPGA technique.

References

[1] C. S. Wallace, "A Suggestion for a fast multiplier", Sep 1962, IEEE Computer society.
[2] Peter. L, "Modular multiplication without trial division", April 1985, American mathematical society.
[3] Naofumi Takagi, Shuzo Yajima, "Modular multiplication hardware algorithms with redundant representation and their applications to RSA cryptosystem" July 1992, IEEE Transactions on computers.
[4] Dan Zuras, "More on squaring and multiplying large integers", Aug 1994, IEEE Transactions on computers.
[5] Holger Orup, "Simplifying quotient determination in high radix modular multiplication", Sep 1995, IEEE Transactions on computers.
[6] Cetinkayakoc,Tolga Acar, "Analyzing and comparing montgomery multiplication algorithms", Sep 1996, IEEE Transactions on computers.
[7] S. F. Oberman, Michael.j, "Division algorithms and implementation", Aug 1997, IEEE Transactions on computers.
[8] Peter Soderquist, Miriam Leeser, "Division and square root choosing the right implementation", July 1997, IEEE Transactions on computer.
[9] S. F. Oberman, "Floating point division and square root algorithm and implementation in the microprocessor "Oct 1999, IEEE Transactions on computers.
[10] Loi Ai, A, Tawalbeh. A. F. Tenca, "An algorithm and hardware architecture for integrated modular division and multiplication", Feb 2000, IJSCE.
[11] Nadia Nedjah, Luiza De Macedo Mourelle, "Hardware architecture for the montgomery modular multiplication", March 2001, IJSCE.
[12] Viktor Bunimor, Manfred Schimmler, "Area and time efficient modular multiplication of large integers", April 2003, IEEE Computer society.
[13] David Narh Amanor,Viktor Bunimov, "Efficient hardware architecture for modular multiplication on FPGAs", April 2005, Computer society IEEE .
[14] Marcelo E. Kaihara, "A hardware algorithm for modular" Jan 2005, IEEE Computer society.
[15] Taek-Jun Kwam,Jeft Drape, "Floating point division and square root implementation using a taylor -series expansion algorithm with reduced look-up tables", June 2008, IEEE Transactions on computers.

[16] Miaoing Huang, KrisGaj, “An optimized hardware architecture for the montgomery multiplication algorithm”, Aug 2008, IEEE Transactions on computer
[17] J. H. Yang, C. C Chang, “Efficient residue number system iterative modular multiplication algorithm for modular exponentiation”, Nov 2008, IEEE Transactions on computers.
[18] Zdenka Babic, Aleksej Auramouic, “An iterative logarithmic multiplier”, Nov. 2010 IEEE Computer society.
[19] Ingo Rust,Tobias G. Noll, “A Digit-set- interleaved radix-8 division/square root kernal for double precision floating point”, Mar 2010, IEEE Computer society.
[20] Kihwan Jun, Earl E. Swartzlander, “Modified non-restoring division algorithm with improved delay profile and error correction”, June 2012, IEEE Computer society.
[21] Deeksha R Shetty, “Improving accuracy in Mitchell’s logarithmic multiplication using iterative multiplier for image processing application”, July 2013, IEEE Computer society.
[22] Khalid javeed, “Serial and parallel interleaved modular multiplier on FPGA platform”, Sep 2015, IJSCE.
[23] Rohini. S. Hongal, Anita D. J., “Comparative study of different division algorithms for fixed and floating point arithmetic unit for embedded applications” Sep 2016, IJSCE.
[24] Durgesh Nandan, Jitendra Kanungu, “65 Years Journey of Logarithm Multiplier”, April 2018, IJSCE.