Design and Implementation of Pipelined Floating Point Multiplier using Wallace Algorithm

Shilpa Ambiger
Dept of Electronics and Communication
KLE Technological University
Hubblli, India

Sanjay Eligar
Dept of Electronics and Communication
KLE Technological University
Hubballi, India

Abstract:- A digital circuit in which the changes in the state of the memory elements are synchronized to a clock (single-phase) signal is known as Synchronous Network. The optimization of the network is an important factor to reduce the overall manufacturing cost and increase productivity. The network parameters such as speed, area, power could be optimized. Lower power consumption and lesser area are some of the very important factors to be consider in the fabrication of DSP systems as well as good performance systems. Multipliers have comprehensive importance in both digital signal processors and microprocessors. So design of such high speed multiplier is very essential. Optimizing the speed as well as area of any multiplier is a prime design issue. Many methods are available to speed up a multiplier. This work incorporates with floating point pipeline technique for improving multiplier's performance.

***Keywords**:- Wallace algorithm; floating point; multiplication; sparten 3; ISE 14.2; verilog.*

I. INTRODUCTION

The increasing complexity of the VLSI circuits has led circuit designing to become a very complex task. There is need for the design techniques to be fast and also consume less power with minimum area for systems with increasing complexity. The timing optimization is becoming one of the most important factors in this domain. The great changes in the design methodologies of the future are required is pipelining technique. In computing, floating point gives an account about a way of representing an approximation of any real number in such a way that can support a wide range of values. In pipeline technique instructions and arithmetic operations are executed in overlapping. The pipeline consists of number of processing stages of instruction pipelining or arithmetic pipelining. These series of processing stages consists of combinational logic circuits to perform arithmetic or logic operations also to generate partial products. Series of processing stages are separated by clocked latches, the group of these clocked latches is called register. These registers are used to hold the intermediate results between series of pipeline stages. A data can be latched to registers by clock signal, this common clock signal is applied to all the registers presented in the pipeline stages simultaneously called as synchronous clock.

II. LITERATURE SURVEY

Various methods have been proposed for designed multipliers they are higher speed, power consumption will be less and less area. The steps involved in multiplication are explained here are partial product generation, reduction and last addition [1]. Multipliers with good performance will provide a higher speed and lower power utilization. Multipliers are the core part of all arithmetic processing units with having demand on their speed and lower power consumption [2]. Many methods are available to speed up of multipliers by improving the radix, number of compressors used and usage of fast adder for addition. Another method for improving of speed is pipeline technique with 3 stages. A comparison between pipeline technique and non pipeline technique done with respect to different parameters [3].

Decimal multiplication plays a vital role in most of commercial applications. Several improvements are introduced to the design a last carry propogation adder will be implemented. A representation of floating point number is done in 3 fields named as sign bit if zero means positive, one means negative, the method of representation is known as magnitude. Second field denoted as exponent and the last field fraction field denoted by F [4]. Floating point representation gives about real number approximation with consideration of wide range of values. Wallace algorithm and block diagram are presented here. Introducing about mantissa, sign bit and normalization is done [5].

III. TYPES OF MULTIPLIERS

A. *Array Multiplier*

Array multipliers are originated from parallelogram multiplication, the name parallelogram because the structure of array multiplier looks like parallelogram. Parallel adders are constructed in array multiplier as shown in fig 1 each stage of these will receive inputs in the form of partial product and will generate carry out into the next row. All partial products are generated at the same time in this case. By seeing the structure of array multiplier we can observe two parts that are vertical and horizontal. There are two types of delays full adders delay and gate delay. These vertical and horizontal have some delay like full adder's delay and gate delays.

The array multiplier have the simple and regular structure. As having the regular structure and also small size it can be layout easily.

As having small size time taken to design this multiplier will be much less as compared to other multipliers like wallace tree multiplier. Pipelined architecture also can be done because of its ease of design. By having these advantages it also has disadvantages which is worst case delay and slow speed.

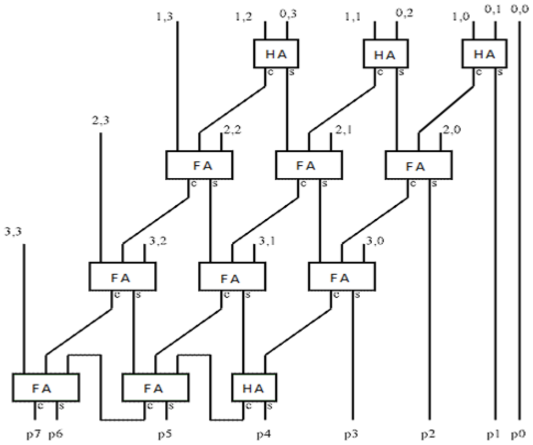

Fig 1:- Block diagram of array multiplier

B. *Wallace Tree Multiplier*

Wallace tree multiplier develops a speedy process for multiplication of two numbers. By seeing the structure of wallace tree multiplier we can say that a tree of carry save adder is constructed. By means presence of these trees of carry save adders the multiplicand-multiplies are summed up parallel as shown in the fig 2.

Carry save adder sums up 3 numbers at a time so there by reducing the delay and produce two binary numbers. So the number of logic levels also required in less number.

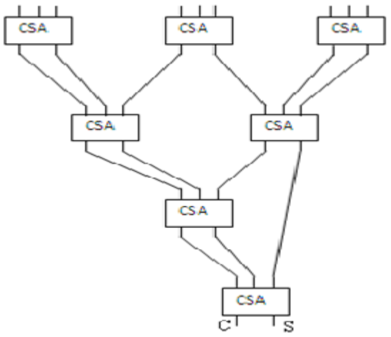

Fig 2:- Block diagram of Wallace tree multiplier

Signals with 3 bits are made to transmit a single bit full adder the output produced by that will supplied or passed to the lateral stage of full adder of carry save adder, this carry output then passed to the lateral stage of full adder of same bit thus creating a tree structure.

The carry output from each stage of full adder is then taken out to form 2nd result scalar rather than brought to the next bit thus the name comes carry save. As its having irregular structure the designing of circuit layout will not so easy and the velocity of operation will be high.

C. *Booth Multiplier*

Improving of multipliers can be done in many ways with the reduction in the number of partial products etc. One of the example for such type of multiplier is booth multiplier. Booth multiplier scans 3 bits at a time there by reducing the number of generated partial products. The 3 bits are from- two bits from the present available pair and the remaining one bit is from higher order bit of an intimate inferior order pair.

To speed up of the multiplier Booth encoding may perform many stages of multiplication at a time. The booth algorithm makes use of fast and small adder substractor. The power consumption will be more because of large number of adder cells required.

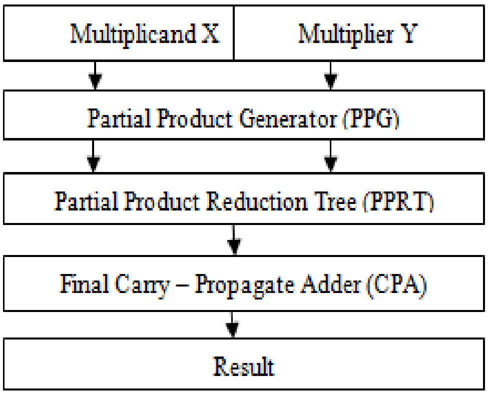
Multiplicand X
Multiplier Y
Partial Product Generator (PPG)
Partial Product Reduction Tree (PPRT)
Final Carry – Propagate Adder (CPA)
Result

Fig 3:- Block diagram of booth multiplier

4 bit multipliers	Array multiplier	Wallace tree multiplier	Booth multiplier
Delay	9.457ns	9.171ns	13.582ns
Area estimation			
1. No of slice LUT's	39	21	43
2. No of LUT flip flop pairs	39	21	43

Table 1. Comparison Between Multipliers

IV. IMPLEMENTATION

A. *Pipeline Technique*

The pipeline technique is one of the most followed technique in the digital design. This technique is done with breaking down of a sequential process into sub blocks. These sub blocks are again pushed into a series of hardware blocks in order to process it independent of the complete operation. In pipeline technique instructions and arithmetic operations are executed in overlapping. The pipeline consists of number of processing stages of instruction pipelining or arithmetic pipelining. These series of processing stages consists of combinational logic circuits to perform arithmetic or logic operations also to generate partial products. Series of processing stages are separated by clocked latches, the group of these clocked latches is called register. These registers are used to hold the intermediate results between series of pipeline stages. A data can be latched to registers by clock signal this common clock signal is applied to all the registers presented in the pipeline stages simultaneously called as synchronous clock.

In pipeline each step is carrying a single microinstruction and every step is linked to other every step. In this work we are preferring 3 stage pipeline. First stage is partial product, second stage is addition and the last stage is normalization.

B. Floating point multiplier

Floating point gives an account about a way of representing an approximation of any real number in the direction that will support wide range of values.

1 bit Sign	8 bit exponent	23 bits fraction
1	10000010	00110000000000000000000

Fig 4:- Floating point format

Fig 4 shows the format of floating point. Which consists of 24 bits. The first bit is represented as bit sign, second bit is represented as exponent bit which consists of 8 bits and the last bit is represented as fraction bit which consists of 23 bits.

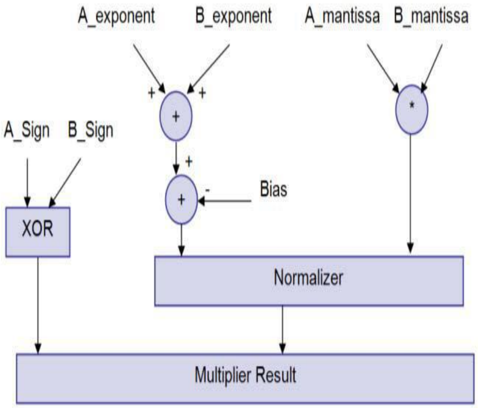

Fig 5:- Floating point block diagram

Fig 5 shows the floating point block diagram. A bit sign is passed to the XOR gate. A 8 bit exponent are added and the bias is applied to this addition then the result is passed to the normalizer. Lastly the 23 bits mantissa(fraction) adder together and passed to the normalizer. The XOR output and normalizer output passed to the multiplier result which gives final result.

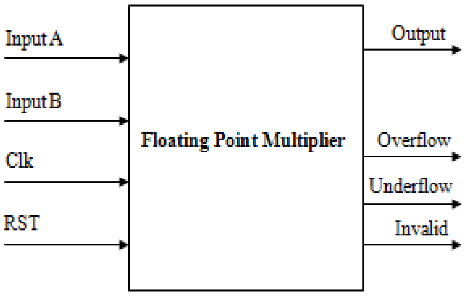

Fig 6:- Schematic representation of floating point multiplier

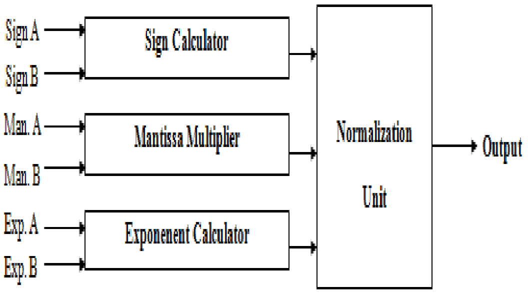

Fig 7:- Implementation of floating point multiplier

Fig 7 shows the implementation of floating point multiplier. The multiplication of two floating point numbers is done by

- By adding the exponent of given two numbers then subtracting the bias from the result of those two given numbers.
- Multiplying the mantissa of given two numbers
- Calculate the sing by XORing sign bits of given two numbers.

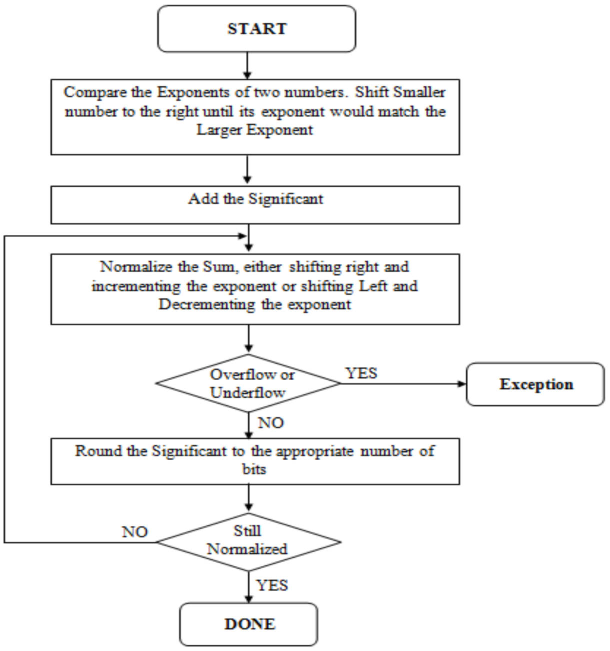
START
Compare the Exponents of two numbers. Shift Smaller number to the right until its exponent would match the Larger Exponent
Add the Significant
Normalize the Sum, either shifting right and incrementing the exponent or shifting Left and Decrementing the exponent
Overflow or Underflow
YES
Exception
NO
Round the Significant to the appropriate number of bits
NO
Still Normalized
YES
DONE

Fig 8:- Floating point multiplier algorithm

- Multiply the significand/fraction of given numbers (1.M1*1.M2)
- Place the decimal point in the result of multiplication

- Add the exponents and then substract the bias from that (Exp1+Exp2-bias).
- Getting the sign by XORing i.e, Sign1 XOR Sign2.
- Normalize the result i.e, by getting 1 at the MSB of the result's significand bit.
- By rounding the results in order to adjust in the available bits.
- Checking of overflow and underflow occurances.

C. Design of floating point pipelined multiplier

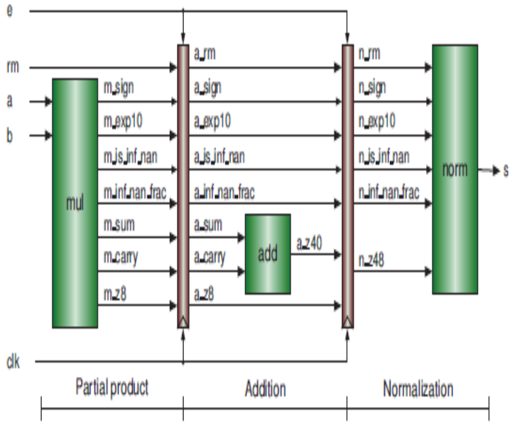

Fig 9:- Floating point pipelined multiplier block diagram

Fig 9 shows the pipelined floating-point multiplier's block diagram. Here the operation is divided as three stages: i)Partial product generated ii)Addition iii)Normalization. Pipeline registers are fitted in between many different stages and also having write enable signal which would be used in the design of CPU to crib the function of pipeline method. Here m, a, n are used as prefix to the signals of partial product generated, addition of signals and last normalization of circuit respectively. 'm' denotes multiplication, 'a' denotes addition, and 'n' denotes normalization.

D. Design of Wallace tree pipelined floating point multiplier

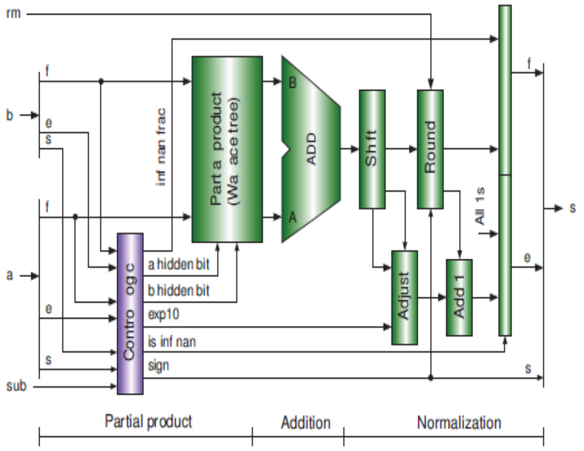

Fig 10:- Pipelined floating point Wallace tree multiplier block diagram

We have compared the multipliers and know that wallace tree multiplier is having the less delay compare to the booth multiplier and array multiplier. So wallace tree multiplier is considered as fastest multiplier.

Fig 10 shows the block diagram of pipelined floating point wallace tree multiplier. Which consists of 3 parts named as i) Wallace tree to produce the partial product generated (sum and carry) ii) Adder for adding the sum and carry iii) Circuit for normalization. In order to perform the significand multiplication two clock cycles are used. We used pipeline registers to execute multiplication of float numbers on each clock cycle so we have added a pipeline register to the Wallace tree.

V. RESULTS

A. Results of pipelined technique

Analysis is done by using the tool called Xilinx ISE design suite 14.2 used is SPARTAN3E. The device considered is XC3S500e-5ft256. The table 2 shows the delay analysis of wallace tree pipelined floating multiplier.

Stage	Delay
1	1.800ns
2	6.100ns
3	30.384ns

Table 2. Delay Analysis Of Wallace Tree Pipelined Floating Point Multiplier

Multiplier(wallace tree)	Delay
Without pipeline	55.020ns
With pipeline	38.334ns

Table 3. Comparison Between Pipelined And Non Pipelined Technique

VI. CONCLUSION

A comparison between different types of multipliers done in the earlier stage. We came to know that wallace tree multiplier is the fastest multiplier compared to booth and array multiplier. So in the next stage a comparison between pipelined technique and non pipelined technique is done in terms of area and also delay. By analysing the area and delay produced by both pipelined multiplier and non pipelined multiplier we can say that pipelined multiplier has high speed compared to non pipelined at the cost of area.

REFERENCES

[1] J. Sharma, S. Kumar, and B. Sampla, "Speed comparison of various
multipliers," International Journal of Engineering and Management
Research (IJEMR), vol. 4, no. 3, pp. 244–247, 2014.

[2] S. Vaidya and D. Dandekar, "Delay-power performance comparison of
multipliers in vlsi circuit design," International Journal of Computer
Networks & Communications (IJCNC), vol. 2, no. 4, pp. 47–56, 2010.

[3] J. M. Jose and R. Paul, “A novel booth wallace multiplier for dsp applications,”

[4] S. Khan and M. Z. Alam, “Implementation and simulation of ieee 754 Single -precision floating point multiplier,” 2014.

[5] C. V. Matey, D. S. Chede, and S. Sakhare, “Design & implementation of floating point multiplier using wallace and dadda algorithm,” International Journal of Application or Innovation in Engineering & Management, vol. 3, no. VI, 2014.

[6] Y. Li, Computer Principles and Design in Verilog HDL. John Wiley & Sons, 2015.

[7] B. S. Ganesh, J. Abhilash, and G. R. Kumar, “Design and Implementation of floating point multiplier for better timing performance,” International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 1, no. 7, pp. pp–130, 2012.

[8] M. R. Kumar and G. P. Rao, “Design and implementation of 32 bit high level wallace tree multiplier,” Int. Journ. of Technical Research and Application, vol. 1, pp. 86–90, 2013.