# Single Precision Floating Point Divider Design

**[1]Serene Jose, [2]Sonali Agrawal**
[1,2]Department of Electronics and Communication, Amrita School of Engineering
Kasavanahalli, Bangalore- 560035, India

**Abstract**— Growth in Floating Point applications and mainly its usage in reconfigurable hardware have made it critical to optimize floating point units. Divider is of particular interest because the design space is large and divider usage in applications varies widely. The design presented in this paper covers a range of performance, area and throughput constraints. Floating point numbers can be represented by single and double precision respectively. A design for single precision floating point divider was done in Verilog and was synthesized using Xilinx and Synopsys tool. The path delay, device utilization was also determined successfully.

**Keywords—** Divider, Floating Point Unit (FPU), and Single precision

## I. INTRODUCTION

Reconfigurable hardware mainly FPGAs and DSP processors use floating point operations in configuring the real time data. This paper presents the implementation of a 32 bit single precision IEEE 754 standard floating point divider which is designed with the power of handling rounding and exceptional cases of division like Not A Number along with special cases of floating point numbers like de-normal, real zero, infinite positive and negative values.

In recent FPUs, importance has been given in developing fast adders, subtractors and multipliers with very less emphasis on dividers. But to deal with the upcoming new technologies, an efficient implementation of division is inevitable. As such many algorithms were developed for dividers including subtractive methods, functional iterations and high radix algorithm for faster computation which further required multipliers and thus consumed large area and power. To account for this drawback, digit recurrence algorithm[1],[2],[3],[4] which uses subtraction along with non-restoring method for computation as it consumes much less area and power.

In the beginning, floating-point divider dealt with obtaining the value of denominator inverse from the LUT which is then multiplied with numerator[5],[6]. But this method was favourable only for small dividers. Following this, divider design with newton-Raphson, Taylor series iterations was developed which had the drawback of high latency[7],[8].To speed up the dividing process, an algorithm based on higher radix SRT [9],[10]was introduced which is complex in nature and is commonly employed for microprocessors.

This paper gives architecture for floating point divider where the divider core employs a fixed point division based on binary digit recurrence algorithm .The area, power and delay parameters are also found out using Synopsys design compiler and IC compiler and was also synthesised in Xilinx.

## II. FLOATING POINT REPRESENTATION

Floating point is a method of representing a wide range of real values. A floating point number can be represented in the form: $\pm m\times b^e$ ,where e the exponent, m the mantissa and b the base or radix. The IEEE Standard 754 for binary floating point number representation which includes single precision with 32 bit wide and double precision with 64 bit wide format.

This section briefly examines the single precision floating point representation. The MSB starts from the left which is one bit width sign, following eight bit width exponent and 23 bit width mantissa (significand) or fractional part.

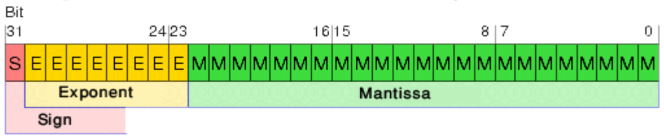


Fig 1.1 Single Precision Format

The value can be represented as:
Normalized value: $(-1)^s\times 2^e\times(1.f)$
Denormal or Subnormal value: $(-1)^s\times 2^{-126}\times(0.f)$ where,
s= 0 for positive and 1 for negative
$f = (b_{23}^{-1}+b_{22}^{-2}+b_{21}^{-3}+b_i^{n}+\ldots\ldots\ldots+b_0^{-23})$ ; $b_i^n$= 0 or 1

When the leading bit of mantissa is 1, commonly referred to as hidden bit followed by the radix point and the 23 bit fractional part, then the representation is said to be normalized.

e= unbiased exponent

The exponent can hold both positive and negative value. The value range of unbiased signed integer is [-128,127] or unsigned integer [0,255]. A bias of 127 is added to actual exponent e, to make negative exponents possible without using a sign bit.

For the biased exponent E, $E_{max}$=255 and $E_{min}$=0 are reserved for special quantities: The number zero which can be positive or negative is represented with E=0 and f=0 .The hidden bit is always 0 ; Denormalized numbers which can also be positive or negative are represented with E=0 and f≠0, which implies the value of e= -128 and mantissa M=0.f.The hidden bit is 0 ; Positive or negative Infinity have E=255 and a fraction f=0 including the hidden bit ; Finally, an exponent E=255 and a fraction f≠0 represents the symbolic unsigned entity NaN (Not a Number),which can be QNaN (Quiet NaN) produced by operations like 0/0,∞/∞ and SNaN (Signalling NaN) is signalled if operations like division by 0 or negative square roots are encountered. The exceptional cases of underflow and overflow property should be also encountered. .i.e. whenever the floating point division result exceeds the minimum or maximum value of the restricted exponent range,

To increase the precision of the floating point division result, rounding is necessary. For this, three bits known as guard, round and sticky bits were temporarily added internally to the actual fraction. While guard and round bits are the normal storage holders, the sticky bit is turned '1', whenever a 1 is shifted out of range. Here, the standard default rounding employed is 'Round to nearest even mode'. The value is rounded up or down to the nearest infinitely precise result. If the value is exactly halfway between two infinitely precise results, then it should be rounded up to the nearest infinitely precise even.

## III. Single Precision Floating Point Division Architecture

This division architecture comprises of sign, exponent and mantissa computation sectors. Initially, the divider receives two 32 bit floating point numbers as its operands. The most significand steps in the calculation of the quotient Q of two normalized numbers A, the dividend and B, the divisor are as follows:

The primary aim in the computation process is to unpack these numbers by separating the floating point numbers into the corresponding sign bits, exponent bits and mantissa bits. The sign logic is a simple XOR operation. The exponent is calculated by adding the bias to the subtracted result of exponents.i.e. $E_Q=E_A-E_B+127$.Mantissa division block performs fixed point division using digit recurrence algorithm, which includes the following steps: First, the remainder of the division is set to the value of the dividend followed by subtracting the divisor value from the remainder. If the result is positive or zero, the MSB of the quotient is 1 or else 0.The remainder is shifted left each time after the subtraction process and the whole process continues for n times (equal to mantissa bit length) to obtain the quotient value. After this, the output of the mantissa has to be normalized i.e. mantissa part has to be left shifted so as to make the MSB 1 and correspondingly, changes has to be made to the exponent part also. After normalization, the output has to undergo the rounding control process, where the three bits namely 'guard', 'round' and 'sticky' bits are stacked to the LSB of the 24 bit significand part and the round to the nearest even is performed.

Along with this, corresponding changes are to be made through exponent adjustment block.

Finally, the output from sign block, exponent adjustment block and the rounding block are concatenated in the packing block to produce the final quotient.

To deal with the denormal inputs, a special block with a priority encoder and left shifter is included. The priority encoder functions to detect the position of leading one in the inputs and the left shifter shifts the mantissa part to so much value, so that the mantissa part is presently in normalized form and can be presented to the divider core.

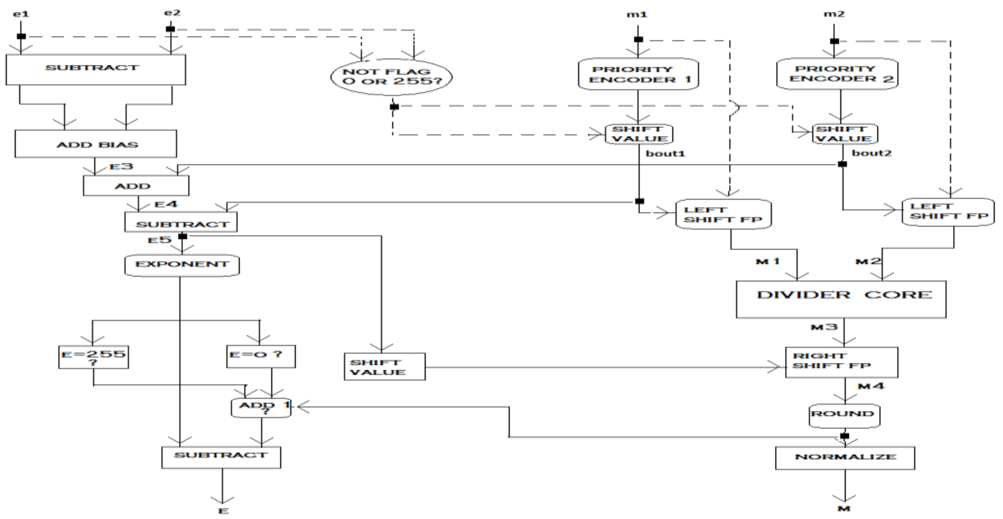
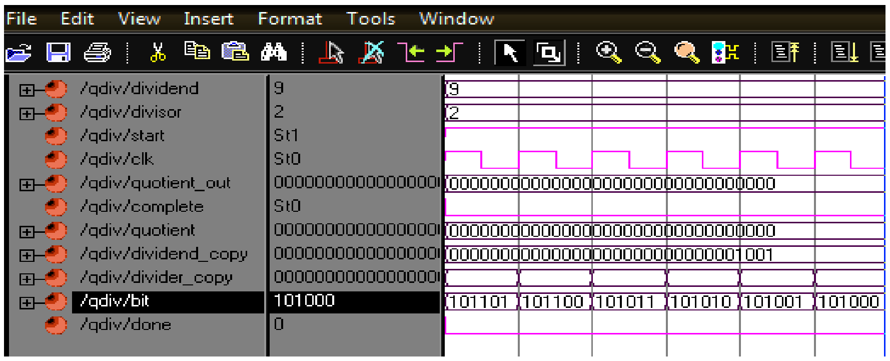


Fig1.2 Architecture of Single Precision floating point divider

## IV. Results

### *A.* Simulation results

The simulations were carried out in Model Sim 5.7g. The implementation of the single precision floating point divider was done using Verilog and result was verified with different values.



| | |
|---|---|
| /qdiv/dividend | 9 |
| /qdiv/divisor | 2 |
| /qdiv/start | St1 |
| /qdiv/clk | St0 |
| /qdiv/quotient_out | 0000000000000000 |
| /qdiv/complete | St0 |
| /qdiv/quotient | 0000000000000000 |
| /qdiv/dividend_copy | 0000000000000000 |
| /qdiv/divider_copy | 0000000000000000 |
| /qdiv/bit | 101000 |
| /qdiv/done | 0 |

Fig 1.3.Simulation result of fixed point divider using digit recurrence algorithm

### *B.* Device Utilization

The hardware amount utilization was obtained from Xilinx. This report gives the information that the divider design is FPGA synthesizable, with the efficient utilization of 1% of the device.

Fig 1.4.Device Utilization Report

| LOGIC UTILIZATION | USED | AVAILABLE |
|---|---|---|
| Number of slices | 166 | 14752 |
| Number of slice Flip Flops | 131 | 29504 |
| Number of 4 Input LUTs | 256 | 29504 |
| Number of Bonded IOBs | 125 | 376 |

***C.* Delay Calculation:** Divider compilation gave 10.305 ns of maximum combinational path delay.

### *D.* Area and Power

| | |
|---|---|
| COMBINATIONAL AREA USAGE | 4756.377nm$^2$ |
| TOTAL CELL AREA | 8439.099nm$^2$ |
| TOTAL DYNAMIC POWER | 45.2305µW |
| CELL LEAKAGE POWER | 27.1779µW |

Fig 1.5. Area and Power Chracteristics

## V. Conclusion

This paper presents a vibrant design for single precision floating point divider with exceptional cases handling capability. The characteristic features of the divider such as area, power, latency etc. were determined using Xilinx and Synopsys tools.

## V1.References

[1] Baesler M,Voigt S,Teufel T, *"FPGA implementation of Radix-10 Digit Recurrence Fixed and Floating-point dividers"*,IEEE international conference on Reconfigurable computing and FPGAs,December 2011,pp:13-19
[2] Rust I,NollnT.G,"*A radix-4 single-precision floating-point divider based on digit set interleaving*", Circuits and systems,proceedings of IEEE international symposium ,june 2010,pp:709-712.
[3] Thakkar A J,Einioui A, "*Design and implementation of double precision floating point division and square root on FPGAs*", IEEE, Aerospace conference on july 2006,pp:7
[4] Stavros Paschalakis ,Peter Lee, "*Double Precision Floating-point arithmetic on FPGAs"*, In proc.IEEE symposium on FPGA for custom computing machines,2004,pp:155-167.
[5] Xiaojun Wang, Braganza S, Leeser M, "*Advanced components in the variable precision Floating-point library*", Field-Programmable custom computing machines, IEEE symposium on Dec 2006, pp.: 249-268
[6] J.Dido ,N.Geraudie, "*A Flexible floating point format for optimizing data paths and operators in FPGA based DSPs*", in International Symposium on Field Programmable Gate Arrays, Feb. 2002, pp.50-55.
[7] Suhap Sahin, Adnan Kavak, Yasar Becerikli,H.Engin Demiray, "*Implementation of floating point arithmetic using an FPGA*", IEEE,2003,pp:12-17
[8] E.Roesler and B.E Nelson "*Novel optimizations for hardware floating point units in a modern FPGA architecture*", in 12th International Conference on Field-Programmable Logic, pp.637-646,2002.
[9] X.Wang and B.E. Nelson, "*Tradeoffs of designing Floating point division and square root on Virtex FPGAs*", in 11th IEEE Symposium on Field Programmable Custom Computing Machines, April 2003,pp.195-203.
[10] Mohamed Anane,Hamid Bessalah,Mohamed Issad,Nadjia Anane,Hassen Salhi,"*Higher Radix and Redundancy Factor for floating-point SRT Division*",IEEE transactions on VLSI,June 2008,vol 16,pp:774-779.

## About The Authors

**Serene Jose** received her Bachelor of Engineering in Electrical And Electronics Engineering from Mahatma Gandhi University, Kerala in the year 2010 and currently pursuing Master of Technology in VLSI design in Amrita University, Bangalore. Her main research interests include Low Power VLSI Design, Digital Design.

**Sonali Agrawal** graduated from Pandit Ravishankar Shukla University in 2002 .She completed her post-graduation from NIT, Rourkela in 2008 .Presently ,she is working as an Asst. Professor at Amrita Viswa Vidyapeetham and has been engaged with research in Low Power VLSI Design, Digital Design, Hardware Description languages,DSP.