Multi Mode IEEE 754 Floating Point Unit (FPU)

V M Ranjith[1], Vithal Reddy[2], Dhanush R[3]

[1]Dept. of Electronics and Communication, BMS College of Engineering
[2]Dept. of Electronics and Communication, BMS College of Engineering
[3]Dept. of Electronics and Communication, BMS College of Engineering, Bengaluru

---

Abstract - *In this paper, the implementation of floating point ALU is presented and designed. The Floating Point Unit acts as a co-processor. It carries out arithmetic operations of floating numbers. The design is made to operate in 3 modes, 16-bit half-precision, 32-bit single precision and 64-bit double precision ALU for better precision and accuracy. The design is based on high performance, and is done after functional and timing simulation. The simulation tool used is Xilinx Vivado. The tool for synthesis and implementation is Quartus.*

***Key Words*: Floating point ALU, Adder, Subtractor, Multiplier, Half precision, Single precision, Double precision.**

1.INTRODUCTION

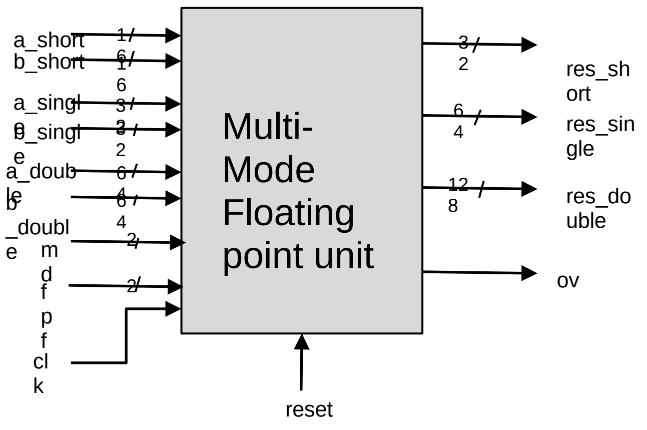

Fig-1 Block diagram of Multi-Mode Floating Point Unit

Half-precision floating numbers occupy 16-bits. The IEEE 754 format specifies that 16 bits are divided into 3 parts

1) sign bit
2) exponent
3) mantissa

The MSB represents a sign bit. If it is '1' then it is considered as a negative number. If it is '0' then it is considered as a positive number. Bits 14-10 represents exponent. The exponent is in offset binary representation with offset 15. Minimum exponent that can be represented is -14. Maximum exponent that can be represented is 15. So numbers with exponent part 2^-14 to 2^15 can be represented through this IEEE 754 half precision floating point representation. Bits 9-0 represents the fraction part. As all the floating point fractions are in the form of 1+ fraction format, to save bits only the fraction part is saved.

Subnormal numbers: The numbers that are obtained when the exponent part is 00000 but fraction part is non zero these numbers are considered as subnormal numbers.

The smallest subnormal number that can be represented using half precision floating point representation is 0.000000059604645.[6] Largest number that can be represented using half precision floating point representation is 65504.[6]

Single precision floating point numbers are represented using 32 bits. The MSB represents a sign bit. If it is '1' then it is considered as a negative number. If it is '0' then it is considered as a positive number. The following 8 bits represent exponent. Minimum exponent that can be represented is -126. Maximum exponent that can be represented is 127. So numbers with exponent part 2^-126 to 2^127 can be represented through this IEEE 754 single precision floating point representation. The remaining 23 bits represent a fraction.

The smallest subnormal number that can be represented using half precision floating point representation is 1.4×10^{45}. Largest number that can be represented using single precision floating point representation is $3.4 \times 10^{38[}$.

[1]

Double precision floating point numbers are represented using 64 bits. The MSB represents a sign bit. If it is '1' then it is considered as a negative number. If it is '0' then it is considered as a positive number. The following 11 bits represent exponent. Minimum exponent that can be represented is -1023. Maximum exponent that can be represented is 1024. So numbers with exponent part 2^-1023 to 2^1024 can be represented through this IEEE 754 single precision floating point representation. The remaining 52 bits represent a fraction.

The smallest subnormal number that can be represented using half precision floating point representation is 4.9×10^{-324}. Largest number that can be represented using double precision floating point representation is 1.8×10^{308}[1].

2. Arithmetic Operations

2.1 Addition

The input fpf (floating point function) bits represent the function to be performed. When the fps bits are set to 00 then the addition of two operands takes place. When the fps is 01 subtraction takes place. The operands consist of a sign bit which depicts whether the number is negative or positive. The fraction part is set according to the exponents of two operands by the comparator and shifting module. When both the exponents are equal then the both fractions are added and the result is normalized. When the exponent of a_short is greater than the exponent of b_short by value 'e', then the fraction part of b_short is shifted right 'e' times by adding 1 to the MSB. Similarly when the exponent of b_short is greater than the exponent of a_short by value 'e', then the fraction of a right-shifted by 'I times by adding the value 1. The corrected fraction parts of both the operands are added. Figure (2) shows a block diagram of the adder/subtractor. m1, m2 represents mantissa of operand 1,2. e1, e2 represents exponents of operands 1,2. Where the difference of both exponents is represented by e. The corrected mantissa are added to the adder/ subtractor unit.

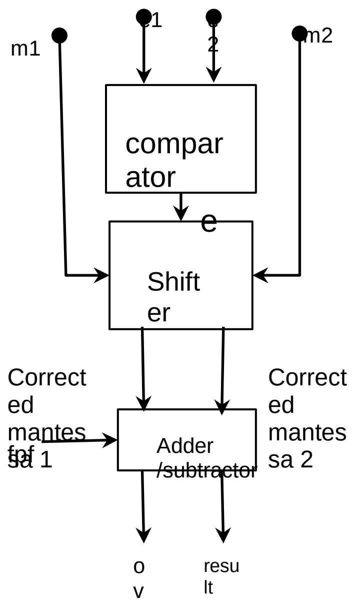

Fig-2 Block diagram of adder/subtractor

2.1.1 Addition of half-precision floating points

When the mode bits are set to 00 the half-precision floating-point unit is enabled. According to IEEE 754 format, it has 5 bits of exponent and 10 bits of the fraction part

The two operands are taken as a_short and b_short. Which are 16 bits long. The exponents of both operands are compared and shifted accordingly. The shifted inputs are now either added or subtracted depending upon 'fps' bits. When the result exceeds the maximum value of representation the overflow flag is set.

2.1.2 Addition of Single precision floating points

When the mode bits are set to 01 the single-precision floating point unit is enabled. IEEE 754 format has 8 bits of exponent and 23 bits of the fraction part

The two operands are taken as a_single and b_single. Which are 32 bits long. The exponents of both operands are compared and shifted accordingly. The shifted inputs are now either added or subtracted depending upon 'fps' bits. When the result exceeds the maximum value of representation the overflow flag is set.

2.1.3 Addition of Double precision floating points

When the mode bits are set to 11 the single-precision floating point unit is enabled. IEEE 754 format has 11 bits of exponent and 52 bits of the fraction part

The two operands are taken as a_single and b_single. Which are 64 bits long. The exponents of both operands are compared and shifted accordingly. The shifted inputs are now either added or subtracted depending upon 'fps' bits. When the result exceeds the maximum value of representation the overflow flag is set.

2.2 Multiplication

Multiplication of IEEE 754 floating-point numbers can be computed by multiplying the normalized mantissa(24-bit mantissa in Full-Precision and 11-bit mantissa in Half-precision), adding the biased bit exponent(8-bit mantissa in Full-Precision and 5-bit mantissa in Half-precision) and calculated the sign by XOR the input sign bits [4].

The multiplier for the IEEE 754 floating-point numbers can be divided into four sections:

1.Sign extraction

2.Addition of Exponent

3.Multiplication of Mantissa

4. Flag

Extract the sign bit of inputs and XORed to get the result sign bit. Add the exponents of input and subtract the bias(bias=[$2^{(n-1)}$-1] where n is the number of exponent bits) component from the summation.

Multiply the mantissa of 2 inputs, if the resulting intermediate does not have a single '1' at MSB, then right-shift results so till we have '1' at MSB and the final output is truncated(LSB bits). Now Normalizing the mantissa by eliminating the most significant 1 and the value of the exponent is incremented corresponding to a right shift.

Flags are raised when Infinity, Zero, Underflow and Overflow cases are detected. The Infinity flag is raised if any exponent of inputs has all 1's and mantissa has all 0's. Zero flags are raised if the exponent is 0 and mantissa is 0. Underflow flag when the sum of both the exponent is less than bias. Overflow is raised when the exponent is beyond the range[5].

The general flowchart of multiplication is shown in Fig-3. where Sa and Sb are sign bits, Ea and Eb are Exponent bits, Ma and Mb are Mantissa of 2 inputs respectively. S, E, M are the sign, exponent, mantissa bits of the result respectively.

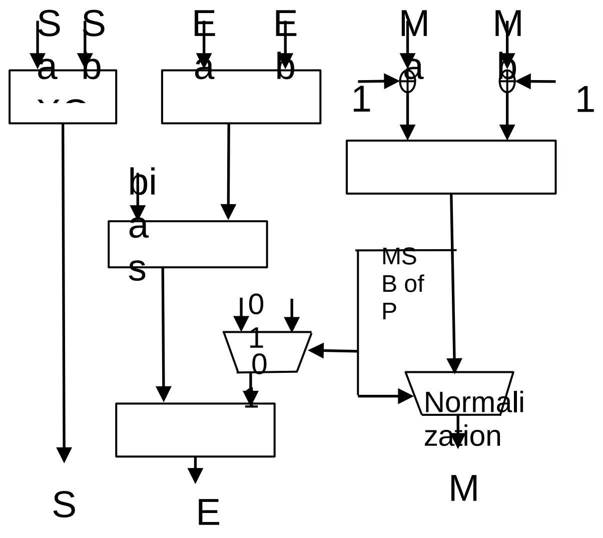

Fig -3 Block diagram of Multiplier.

3. CONCLUSIONS

The floating-point unit is considered a mathematical co-processing unit. Many programs use all short, single, double precision so there is a requirement of hardware that can perform all 3 modes of floating-point operations. The three mode floating-point units enable the option of choosing one of the three modes according to the program. It decreases hardware complexity. It uses simple but effective addition and multiplication algorithms. The experimental results show the functional and timing analysis for all the modules carried out using Intel Quartus Prime.

REFERENCES

[1] Pratik Singh & Kalyani Bhole, 'Optimized floating-point arithmetic unit'. 2014 Annual IEEE India Conference (INDICON). 05 February 2015, ISSN: 2325-9418.

[2] Nielsen, Asger & Matula, David & Lyu, C.N. & Even, Guy. (2000). An IEEE compliant floating-point adder that conforms with the pipeline packet-forwarding paradigm. Computers, IEEE Transactions on. 49. 33 - 47. 10.1109/12.822562.

[3] Thesis 'THE DESIGN OF AN IC HALF PRECISION FLOATING POINT ARITHMETIC LOGIC UNIT' by 'Balaji Navalpakkam Kannan' presented to 'The Graduate School of Clemson University

[4] Prachi Agrawal, Prof. Shravan Sable, Dr Rita Jain.A "Review on IEEE-754 Floating Point Multiplier Using Verilog", Volume-08, Number-03, 2015, ISSN: 2349-4689

[5] Aniruddha Kanhe, Shishir Kumar Ankit Kumar Singh "Design and Implementation of Floating Point Multiplier based on Vedic Multiplication Technique" Oct. 19-20

[6] https://en.wikipedia.org/wiki/IEEE_754

BIOGRAPHIES

V M Ranjith
Dept of Electronics and Communication, BMS College of Engineering, Bengaluru, Karnataka 560019

Vithal Reddy
Dept of Electronics and Communication, BMS College of Engineering, Bengaluru, Karnataka 560019

Dhanush R
Dept of Electronics and Communication, BMS College of Engineering, Bengaluru, Karnataka 560019