ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

# IMPLEMENTATION OF DOUBLE PRECISION FLOATING POINT RADIX-2 FFT USING VHDL

**Tharanidevi.B[1], Jayaprakash.R[2]**

Assistant Professor, Dept. of ECE, Bharathiyar Institute of Engineering for Woman, Salem, TamilNadu, India [1]

Assistant Professor, Dept. of ECE, Muthayammal College of Engineering, Namakkal, TamilNadu, India [2]

**ABSTRACT**: The Discrete Fourier Transform (DFT) can be implemented very fast using Fast Fourier Transform (FFT). It has various numbers of applications in the field of signal processing. The FFT can be designed by radix-2 butterfly algorithm which requires needless computations and data storage. It consumes more power. Using IEEE-754 single precision and double precision floating-point format the Fast Fourier Transform (FFT) for real numbers can be computed which is implemented in hardware FPGA. This paper describes two new architectures for floating-point addition, subtraction and product which are repeatedly used in radix-2 butterfly algorithm. This new architecture reduces computation complexity, data storage, area, and power consumption. The algorithms are simulated using VHDL language.

**Keywords**: Fast Fourier Transform (FFT), Floating-Point, Radix-2 Butterfly, VHDL.

## I. INTRODUCTION

In computation, floating point format represents another way to describe the real number which can support wide range of values. It can be represented in the form as

$$\text{Significant digits} \times \text{base}^{\text{exponent}}$$

Since the real numbers cannot be implemented directly in the hardware, so floating point number representation is used. Commercial floating point algorithm consumes more power and requires more memory. The two new architectures for addition, subtraction and product for floating point number namely Fused DP and Fused AS unit. This new architecture reduces computation complexity, data storage, area, and power consumption. Section 2 describes IEEE-754 Standard. Section 3 describes Fast Fourier Transform. The next two sections describe the Fused DP and Fused AS units. Subsequent sections describe the use of these two operations to implement FFT butterfly units.

## II. IEEE-754 STANDARD

The IEEE Standard for Floating Point Arithmetic (IEEE 754) is a technical standard for floating point computation established in 1985 by the Institute of Electrical and Electronics Engineers (IEEE). Many hardware floating point units use the IEEE 754 standard. The current version, IEEE 754-2008 published in August 2008, includes nearly all of the original IEEE 754 1985 standard and the IEEE Standard for Radix-Independent Floating-Point Arithmetic (IEEE 854-1987).

The advantage of floating-point representation over fixed point and integer representation is that it can support a much wider range of values. For example, a fixed point representation that has seven decimal digits with two decimal places can represent the numbers 12345.678, 123.45, 1.23456 and soon, whereas a floating-point representation (such as the IEEE 754 decimal 32 format) with seven decimal digits could in addition represent 1.234567, 0.00001234567, 1234567000, and so on. The floating-point format needs slightly more storage (to encode the position of the radix point), so when stored in the same space, floating-point numbers achieve their greater range at the expense of precision.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

IEEE floating point numbers have three basic components: the sign, the exponent, and the mantissa. The mantissa is composed of the *fraction* and an implicit leading digit. The exponent base (2) is implicit and need not be stored. The following Table 1 shows the layout for single (32-bit) and doubles (64-bit) precision floating-point values. The number of bits for each field are shown (bit ranges are in square brackets).

*A) Sign Bit*

The sign bit is as simple as it gets. 0 denotes a positive number, 1 denotes a negative number. Flipping the value of this bit flips the sign of the number.

*B) Exponent*

The exponent field needs to represent both positive and negative exponents. To do this, a bias is added to the actual exponent in order to get the stored exponent. For IEEE single precision floats, this value is 127. Thus an exponent of zero means that 127 is stored in the exponent field. For double precision, the exponent field is 11 bits, and has a bias of 1023.

*C) Mantissa*

The mantissa, also known as the significant, represents the precision bits of the number. It is composed of an implicit leading bit and the fraction bits.

TABLE I

| Precision | Sign | Exponent | Fraction | Bias |
|---|---|---|---|---|
| Single Precision | 1[32] | 8 [30-23] | 23 [22-00] | 127 |
| Double Precision | 1 [63] | 11 [62-52] | 52 [51-00] | 1023 |

## III. FAST FOURIER TRANSFORM

The number of complex multiplication and addition operations required by the simple forms both the Discrete Fourier Transform (DFT) and Inverse Discrete Fourier Transform (IDFT) is of order $N^2$ as there are $N$ data points to calculate, each of which requires $N$ complex arithmetic operations.

For length n input vector x, the DFT is a length n vector X, with n elements:

$$\sum_{n=0}^{N-1} x(n) e^{-j2\frac{\pi}{N}nk} k=0, 1, 2,..., N-1 \qquad (1)$$

As the name suggests, FFTs are algorithms for quick calculation of Discrete Fourier Transform of a data vector. The FFT is a DFT algorithm which reduces the number of computations needed for $N$ points from O ($N^2$) to O ($N$ log $N$) where log is the base-2 logarithm.

The Radix 2 algorithms are useful if $N$ is a regular power of 2 ($N=2^p$). There are two different Radix 2 algorithms, the so called Decimation in Time (DIT) and Decimation in Frequency (DIF) algorithms. Decimation in Time (DIT) computational elements consist of complex multiplications followed by a sum and difference network. Decimation in Frequency (DIF) computational elements consist of a sum and difference network followed by complex multiplications.

## IV. FUSED DOT PRODUCT UNIT

In many DSP algorithms the sum of the products of two sets of operands (dot-product) is a regularly used operation. It has some advantages over discrete floating-point multipliers in a floating point unit design. In usual

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

floating-point dot product is carried out with two multiplications and an addition. The multiplications are performed in parallel with two floating-point multipliers followed by a floating-point adder which is costly in teams of silicon area and in power consumption. But in fused floating point unit, the two floating point multipliers, adder and subtraction are performed in same unit which reduces the silicon area and power consumption. The implementation of a floating point fused dot-product unit shown in Fig.1.

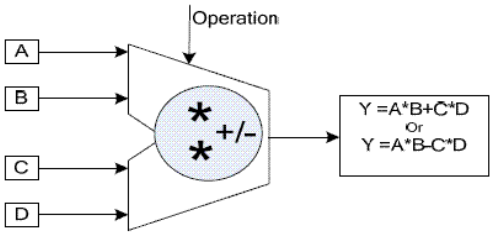


Fig.1. Fused Dot Product Unit

The floating-point two-term fused dot product (Fused DP) unit computes a two term dot product:

$$X = AB \pm CD \qquad (2)$$

The usual dot product adds the two products as shown in (2), but the Fused DP unit performing both addition and difference of the two products, so that the power and area is reduced when compared to a usual parallel discrete dot product implementation, since the rounding and normalization logic of both of the multipliers are eliminated. Multiplies two sets of operands and add/subtract the products. The two products need not to be normalized. Only the final result is normalized and rounded which reduces delay and silicon area.

## V. FUSED ADD-SUBTRACT UNIT

The usual parallel Add-Subtract unit which has separate add and subtract unit in parallel. This may increase the silicon area and consumes more power. Alternatively, in fused floating point Add-Subtract unit is used which reduce the silicon area and power consumption. It combines both addition and subtraction operations as a single unit to perform complex addition and subtraction operation. Sharing the exponent comparison and arrangement reduces complexity. The implementation of a floating point fused Add-Subtract unit shown in Fig.2.

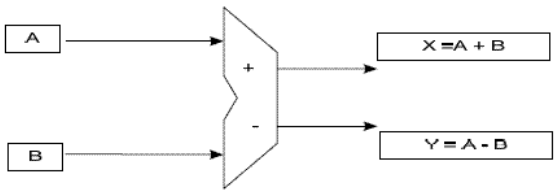


Fig.2. Fused Add-Subtract Unit

The floating-point fused add-subtract unit (Fused AS) performs an addition and a subtraction in parallel on the same pair of data as shown as:

$$X = A \pm B \qquad (3)$$

The both floating point fused dot product unit and fused add-subtract unit are designed based on behavioural modelling using VHDL.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

## VI. RADIX-2 FFT

To express the benefit of the Fused DP and Fused AS units for FFT implementation, FFT butterfly unit designs using both the discrete and the fused units have been made. First radix-2 decimation in time FFT butterfly is designed. It is shown in Fig. 3. All lines carry complex pairs of 32-bit IEEE-754 numbers or 64-bit IEEE-754 numbers and all operations are complex. A discrete implementation of complex adder uses two real adders. A discrete implementation of complex subtraction uses two real subtract and for discrete implementation of complex multiplication uses four real multipliers and two real adders. The complete butterfly consists of six real adders and four real multipliers. The complex add and subtract can be performed with two fused add subtract units and the complex multiplication can be performed with two fused dot product units.

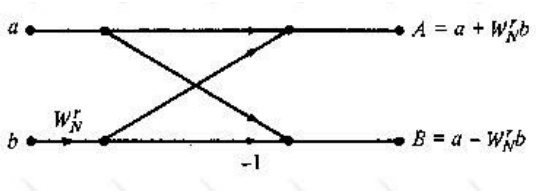


Fig.3. Butterfly for radix-2 DIT FFT

In comparing the discrete and fused radix-2 butterfly units, the fused architecture requires less area and is faster than the discrete implementation. Radix-2 FFT butterfly is performed for both single (32-bit) and double (64-bit) precisions and compared.

## VII. SIMULATION RESULT

### *A) 2-Point Fused FFT (Single Precision)*

$X_0$= (01000001000010110011001100110011, 01000000101100000000000000000000)
$X_1$= (01000000010000000000000000000000, 01000000111111000111101011100001)
$Y_0$= (00000000000000000000000000000000, 10000000100000000000000000000000)
$Y_1$= (00000000000000000000000000000000, 10000000100000000000000000000000)

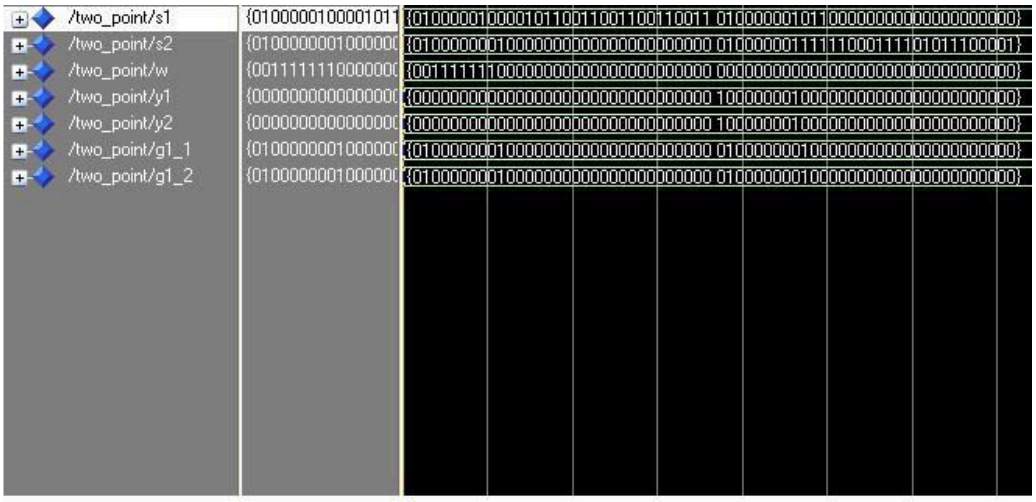


Fig.4. Waveform of 2-point Fused FFT (Single Precision)

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

*B) 2-Point Fused FFT (Double Precision)*

X0= (00111111100000000000000000000000000000000000000000000000000000000,
00000000000000000000000000000000000000000000000000000000000000000)
X1= (10111111100000000000000000000000000000000000000000000000000000000,
00000000000000000000000000000000000000000000000000000000000000000)
Y0= (00000000000000000000000000000000000000000000000000000000000000000,
10000000000000000000000000000000000000000000000000000000000000000)
Y1= (01000000000000000000000000000000000000000000000000000000000000000,
10000000000000000000000000000000000000000000000000000000000000000)

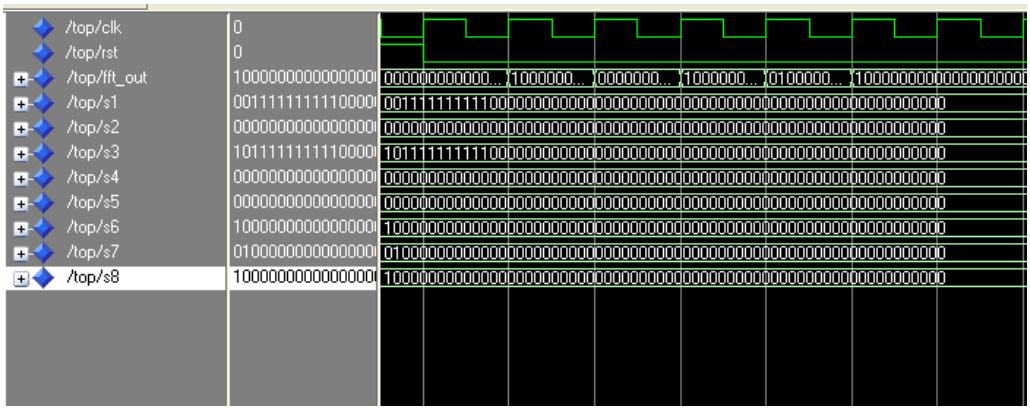


Fig.5. Waveform of 2-point Fused FFT (Double Precision)

*C) FPGA Implementation*

*1) Spartan-6 FPGA*

Unified Learning Kit is based on Texas Instruments OMAP3530 application processor & Spartan-6 FPGA. The Spartan-6 family provides leading system integration capabilities with the lowest total cost for high-volume applications. The thirteen-member family delivers expanded densities ranging from 3,840 to 147,443 logic cells, with half the power consumption of previous Spartan families, and faster, more comprehensive connectivity. Spartan-6 FPGAs are the programmable silicon foundation for Targeted Design Platforms that deliver integrated software and hardware components that enable designers to focus on innovation as soon as their development cycle begins.

Supporting Appliances : The Spartan-6 FPGA supports interfaces such as Ethernet, FPGA HDR2 20-pin Header, Keypad connector(4X4), FPGA Expansion connector, Mictor connector, ADC, DAC, LED, UART Transceiver, 7 segment LED, 16x2 Char LCD, Oscillators (10 & 100Mhz), DDR2 SDRAM, PROM, Dip Switches.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

Fig.6. Unified Learning Kit

*2) 70-Pin Expansion connector*

Most of FPGA signals are directly driven by FPGA which are 3.3V voltage level. For some signals, level translators are used to convert the 1.8V FPGA signals to 3.3V compatible signals.

Using 70-Pin Expansion connector leds are interfaced with FPGA to display the outputs of 32 bit FFT Processor is shown in figure 7.

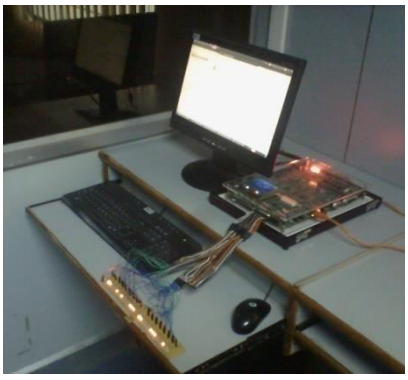

Fig.7. Implementing in UTLP Kit

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 2, Issue 10, October 2013**

*D) Comparison*

*1) 32-bit FFT*

| Architecture / Parameter | Parallel | Fused |
|---|---|---|
| **Area** | 81,029 Gates | 60,094 Gates |
| **Power** | 42mW | 37mW |
| **Delay** | 8.442ns | 7.210ns |

Fig.8. 32-bit FFT Comparison

*2) 64-bit FFT*

| Architecture / Parameter | Parallel | Fused |
|---|---|---|
| **Area** | 139,123 Gates | 93,624 Gates |
| **Power** | 43mW | 34mW |
| **Delay** | 8.833ns | 7.713ns |

Fig.9. 64-bit FFT Comparison

## VIII. CONCLUSION

This paper explains about the design of two new fused floating-point arithmetic units and their application to the implementation of FFT butterfly operations. Using this architectures area, delay and power is reduced for radix-2 FFT butterfly and compared. From the results it is proved that fused arithmetic implementation is better than parallel arithmetic implementation. By double precision it is shown that the result is more accurate than single precision implementation. Fig. 8 and 9 shows the comparison between single precision and double precision for various parameters. In future pipelined architecture of fused arithmetic can design.

## REFERENCES

[1] Earl E. Swartzlander Jr., and Hani H.M. Saleh, "FFT Implementation with Fused Floating-Point Operations," in IEEE Transactions on Computers, 2012.

[2] IEEE Standard for Floating-Point Arithmetic, ANSI/IEEE Standard 754-2008, Aug. 2008.

[3] Sneha N.kherde and Meghana Hasamnis, "Efficient Design and Implementation of FFT," in International Journal of Engineering Science and Technology, 2011.

[4] H.H. Saleh and E.E. Swartzlander, Jr., "A Floating-Point Fused Dot-Product Unit," Proc. IEEE Int'l Conf. Computer Design (ICCD), 2008.

[5] H.H. Saleh, "Fused Floating-Point Arithmetic for DSP," PhD dissertation, Univ. of Texas, 2008.

[6] H. Saleh and E.E. Swartzlander, Jr., "A Floating-Point Fused Add-Subtract Unit," Proc. IEEE Midwest Symp. Circuits and Systems (MWSCAS), pp. 519- 522, 2008.