IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
ISO 3297:2007 Certified
Vol. 3, Issue 7, July 2016

FPGA Implementation of CORDIC Multiplier based Redundant Floating-point Butterfly Architecture for Signal Processing Applications

A. Ramakrishna Raju[1], **N. Samba Murthy**[2], **Dr. M. Kamaraju**[3]

Department of ECE, Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh, India[1, 2, 3]

Abstract: The main objective of this paper is to design a area efficient and high speed floating-point (FP) butterfly architecture for a signal processing applications. The butterfly architectures used in Fast Fourier Transform (FFT) algorithms. Most of the butterfly architectures uses fixed point arithmetic than the FP arithmetic because of lesser speed but main advantage is the high dynamic range. The limitation of FP arithmetic is overcome by using a FDPA unit but it consume more area. This draw back overcome by using a Coordinate Rotation Digital Computer (CORDIC) algorithm. In this paper we propose a CORDIC Multiplier based redundant floating point butterfly architecture for a signal processing applications. The proposed work synthesis is carried out by using XILINX ISE synthesis tool on the XILINX 14.7 platform and simulated using Model Sim simulator. The performance of this design is evaluated on XILINX Spartan3 family device.

Keywords: CORDIC algorithm, FDPA unit, Floating-point, Fixed –point

I. INTRODUCTION

FFT algorithms use butterfly architectures to compute the Discrete Fourier Transform (DFT) of input signal. The butterfly circuit consists of several multipliers and an adder over complex number [1] is shown in Fig.1. Most of the butterfly architectures have been using fixed-point arithmetic, until recently that butterfly architectures based on FP operations grown [1]. The main advantage of FP arithmetic is the high dynamic range, but at the expense of higher delays [1].

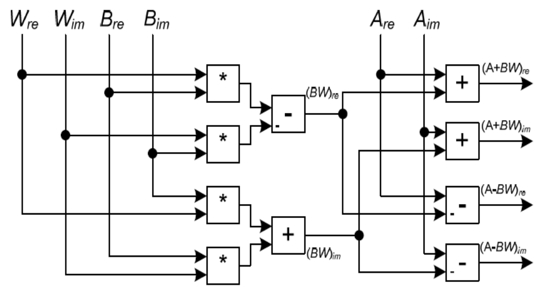


Fig.1. DIT Butterfly architecture expanded with complex numbers

As Discussed, The main limitation of the FP operation is their high delay in comparison with the fixed – point counter parts, is as a way to reducing delay of the FP arithmetic is to merge the several operations in a single floating – point unit. Here are the few techniques for merging the operations. They are Fused-Multiply-Add, Dot-Product and Fused Dot-Product-Add.

The time and area will be saved for butterfly architecture [2][5] by using the FP Fused-Multiply-Add technique, but the butterfly architecture cannot be implemented directly. In order to circumvent this problem Dot-Product unit is required. The Dot-Product unit is an extension of Fused-Multiply technic.

The Dot-Product unit calculates $AB + CD$ (or) $AB - CD$. The Dot-Product unit eliminates more intermediate Normalization and Rounding operations because of it combines more FP operations hence saving more area and time[3] than Fused-Multiply-Add.

Fused-Dot-Product-Add(FDPA) unit is the extension to the Dot-Product unit. By this unit combining more FP operations than the Dot-Product unit. FDPA unit calculates $AB \pm CD \pm E$ over FP operands [1]. The butterfly unit implemented using the FDPA unit is as follows. By using FDPA units achieves higher speed but area consumption is more. In order to achieve higher speed with lesser area CORDIC algorithm has been introduced in FDPA unit [1].

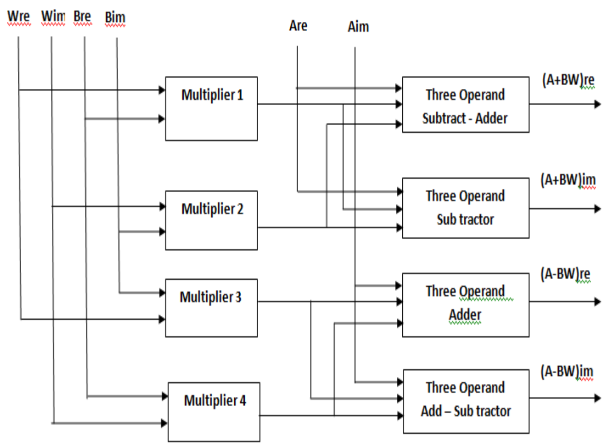


Fig.2 Butterfly architecture with FP Fused-Dot-Product-Add

In this paper CORDIC Multilier based butterfly architecture is proposed which has higher speed and lesser area. The paper structured as follows Section II explains the Proposed Butterfly Architecture, Section III Explains Experimental Results of Proposed Butterfly Architecture , Section IV Conclusion.

II. PROPOSED BUTTERFLY ARCHITECTURE

The proposed CORDIC Multiplier based butterfly architecture is shown in below Fig.3. Proposed butterfly architecture consists of CORDIC Multiplier , Three Operand Subtract-Adder, Three Operand Subtractor, Three Operand Adder and Three Operand Add-Subtractor.

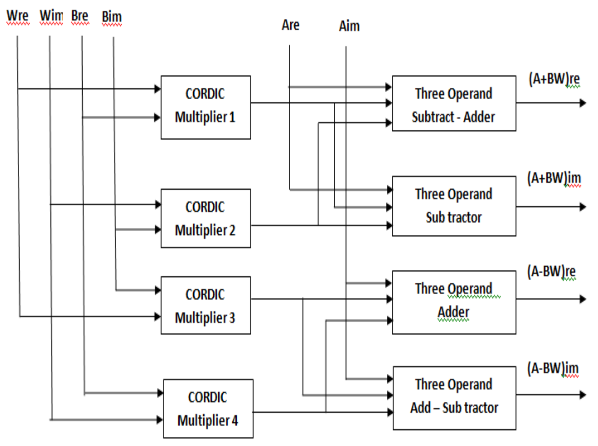


Fig.3. Proposed Butterfly Architecture

A. CORDIC Multiplier

The CORDIC algorithm uses a one-bit-at-a-time approachto make computations to an arbitrary precision. In the process, relatively small lookup tables are used for constants necessary for the algorithm. Typically these tables require only one to two entries per bit of precision. CORDIC algorithms also use only right shifts and additions, minimizing the computation time [10].

A CORDIC algorithm for Multiplication may be derived by using a series representation for $a$as follows:

$$c = a * b \qquad (1)$$

$$= b * \sum_{i=1}^{B} x_i * 2^{-i} \qquad (2)$$

$$= \sum_{i=1}^{B} b * x_i * 2^{-i} \qquad (3)$$

$$= \sum_{i=1}^{B} x_i * (b * 2^{-i}) \qquad (4)$$

From this it is seen that c is composed of shifted versions of b. The unknown coefficients, x_i, may be found by driving a to zero one bit at a time. If the ith bit of a is non-zero, b_i is right shifted by i bits and added to the current value of c. The ith bit is then removed from a by subtracting $2^{-i}$from a. If a is negative, the ith bit in the twos complement format would be removed by adding 2^{-i}. In either case, when a has been driven to zero all bits have been examined and $c$contains the signed product of a and b correct to B bits.

This algorithm is similar to the standard shift and add multiplication algorithm except for two important features. First, arithmetic right shifts are used instead of left shifts, allowing signed numbers to be used. Secondly, computing the product to B bits with the CORDIC algorithm is equivalent to rounding the result of the standard algorithm to the most significant B bits. The final algorithm is as follows:

```
Multiply (a.b) {
    for ( $j = 1$ ; $j =< R$ ; $j++$) {
        if( $a > 0$)
            $a = a - 2^{-j}$
            $c = c + b * 2^{-j}$
        else
            $a = a + 2^{-j}$
            $c = c + b * 2^{-j}$
    }
    return(c)
}
```

By using CORDIC Multiplication algorithm perform floating-point Multiplication on both signed and unsigned numbers.

B. Three Operand Subtract- Adder

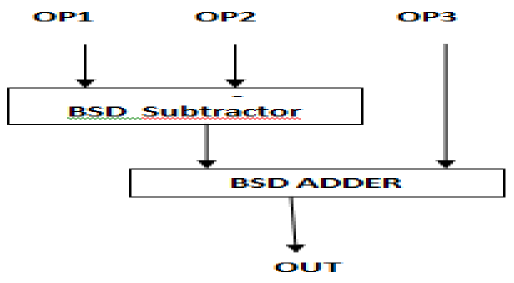


Fig.4. Block Diagram of Three Operand Subtract- Adder

Three Operand Subtract-Adder consists of Binary Signed Digit (BSD) subtractor and BSD adder. These are cascaded to form Three Operand Subtract-Adder is shown in Fig.4. It has three inputs, two are from multiplier 1 and 2 and the real part of A (Are) will be given as third input as shown in the Fig.3.

C. Three Operand Subtractor

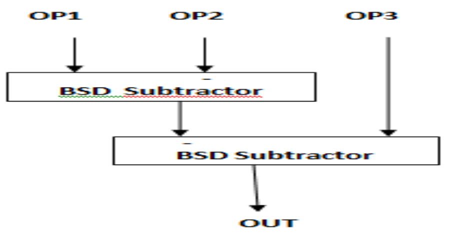


Fig.5 Block Diagram of Three Operand Subtractor

Three Operand Subtractor consists of two BSD subtractors. These are cascaded to form Three Operand Subtractor is shown in Fig.5. It has three inputs, two are from multiplier 1 and 2 and the real part of A (Are) will be given as third input as shown in the Fig.3.

D. Three Operand Adder

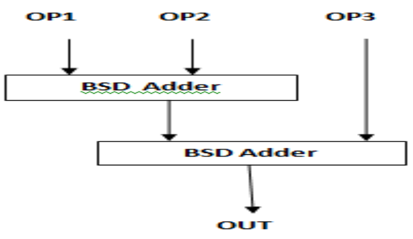


Fig.6. Block Diagram of Three Operand Adder.

Three Operand Adder consists of two BSD Adders . These are cascaded to form Three Operand Adder is shown in Fig.6. It has three inputs, two are from multiplier 3 and 4 and the imaginary part of A (Aim) will be given as third input as shown in the Fig.3.

E. Three Operand Add-Subtractor

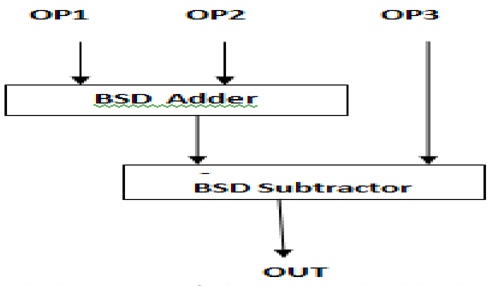


Fig.7. Block Diagram of Three Operand Add-Subtractor.

Three Operand Subtract-Adder consists of BSD Adder and BSD Subtractor. These are cascaded to form Three Operand Add-Subtractor is shown in Fig.7. It has three inputs, two are from multiplier 3 and 4 and the Imaginary part of A (Aim) will be given as third input as shown in the Fig.3.
The proposed CORDIC Multiplier based butterfly architecture is designed in Xilinx ISE and there RTL schematic diagram is shown in Fig 8 and Fig 9.

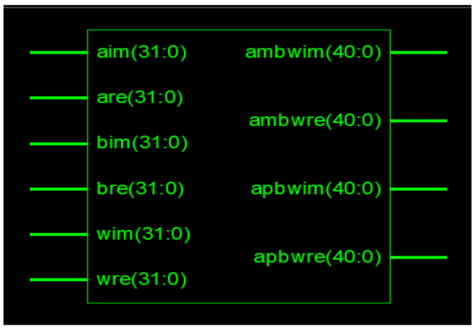


Fig.8. RTL Schematic Diagram of Proposed Butterfly Architecture

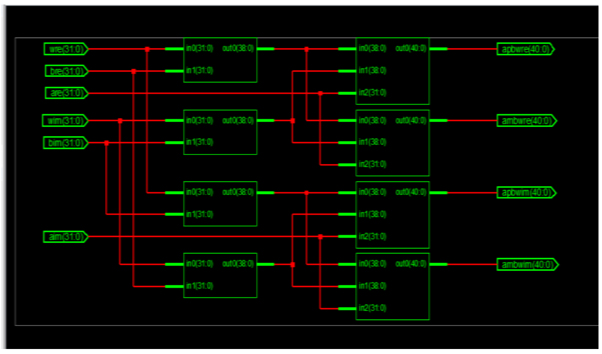
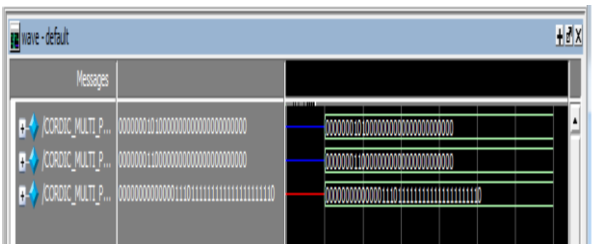


Fig.9. Internal RTL Schematic Diagram of Proposed Butterfly Architecture.

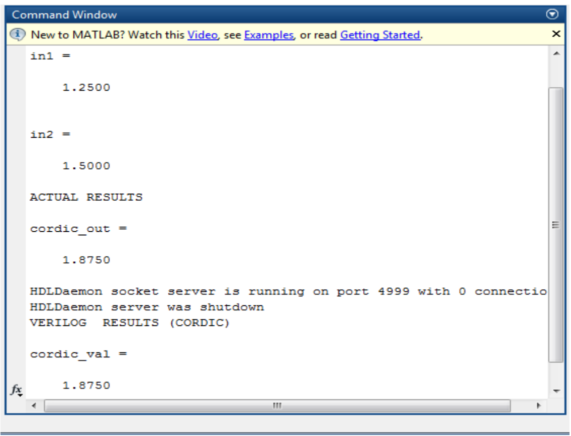
III.EXPERIMENTAL RESULTS

The proposed butterfly architecture is simulated using ModelSim and the results are verified by using MATLAB. The CORDIC Multiplier simulation results are obtained by giving the following data as input In0 = 00000001010000000000000000000 (1.25) and In2 = 00000001100000000000000000000 (1.5) and obtain the output as out= 00000001100000000000000000000 (1.5) is shown in Fig.10 and Fig.11.



```
wave - default
Messages
/CORDIC_MULTI_P...  0000000101000000000000000000
/CORDIC_MULTI_P...  0000000110000000000000000000
/CORDIC_MULTI_P...  00000000000001110111111111111111110
```

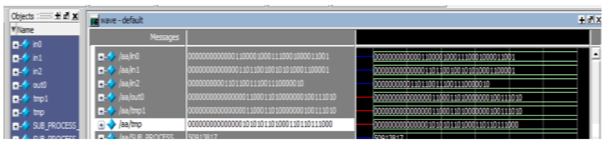
Fig.10. Simulation Results of CORDIC Multiplier



```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
in1 =

    1.2500


in2 =

    1.5000

ACTUAL RESULTS

cordic_out =

    1.8750

HDLDaemon socket server is running on port 4999 with 0 connectio
HDLDaemon server was shutdown
VERILOG  RESULTS (CORDIC)

cordic_val =

    1.8750
```

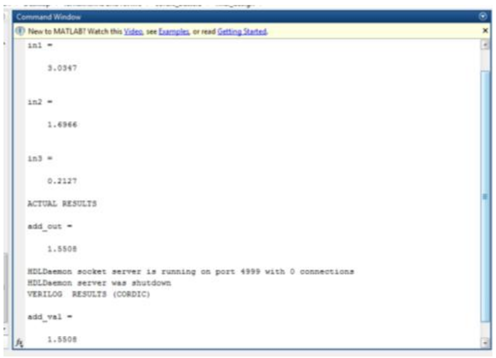
Fig.11. Simulation results of CORDIC Multiplier verification using MATLAB.

The simulation results of Three Operand subtract-adder obtained by giving input In0= 00000000000001100001000111000100011001 (3.0347), In1 = 00000000000001101100100101010001100001 (1.6966), In2 = 00000000001101100111001110000010 (0.2127) and obtain output out = 00000000000000011000110100000000100111010 (1.5508) is shown in Fig 12 and Fig 13.



```
/aa/in0   0000000000000110000100011100010000011001
/aa/in1   0000000000000011011001001010100011000001
/aa/in2   0000000000110110011100111000000010
/aa/out0  0000000000000001100011010000000100111010
/aa/tmp1  0000000000000001100011010000000100111010
/aa/tmp   0000000000000001010101101000110110111000
```

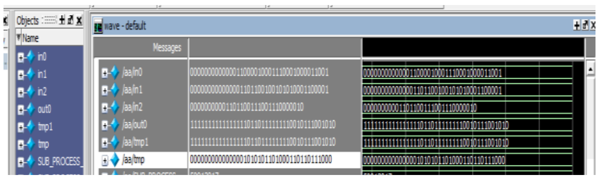
Fig.12. Simulation Results of Three Operand Subtract – Adder



```
in1 =

    3.0347

in2 =

    1.6966

in3 =

    0.2127

ACTUAL RESULTS

add_out =

    1.5508

HDLDaemon socket server is running on port 4999 with 0 connections
HDLDaemon server was shutdown
VERILOG  RESULTS (CORDIC)

add_val =

    1.5508
```

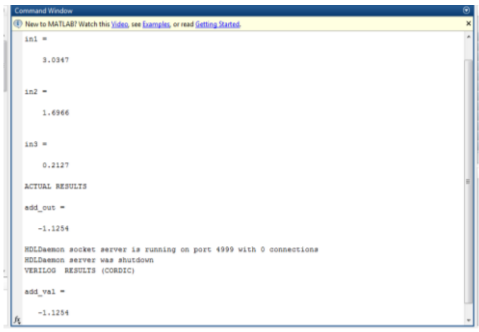
Fig.13. Simulation results of Three Operand Subtract-Adder verification using MATLAB.

The simulation results of Three Operand subtractor obtained by giving input In0= 0000000000000110000100011100010000011001 (3.0347), In1 = 0000000000000011011001001010100011000001 (1.6966), In2 = 0000000000110110011100111000000010 (0.2127) and obtain output out = 1111111111111110110111111110010111001010 (-1.1254) is shown in Fig. 14 and Fig. 15.



```
/aa/in0   0000000000000110000100011100010000011001
/aa/in1   0000000000000011011001001010100011000001
/aa/in2   0000000000110110011100111000000010
/aa/out0  1111111111111110110111111110010111001010
/aa/tmp1  1111111111111110110111111110010111001010
/aa/tmp   0000000000000001010101101000110110111000
```

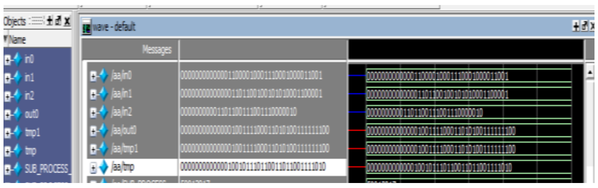
Fig.14. Simulation Results of Three Operand Subtractor



```
in1 =

    3.0347

in2 =

    1.6966

in3 =

    0.2127

ACTUAL RESULTS

add_out =

   -1.1254

HDLDaemon socket server is running on port 4999 with 0 connections
HDLDaemon server was shutdown
VERILOG  RESULTS (CORDIC)

add_val =

   -1.1254
```

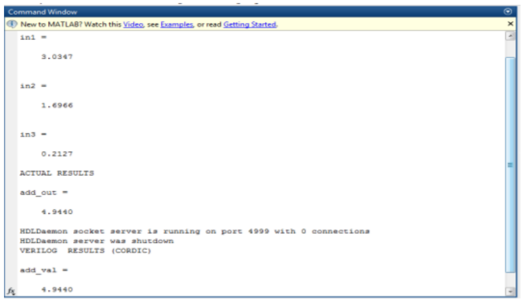
Fig.15. Simulation results of Three Operand Subtractor verification using MATLAB.

The simulation results of Three Operand adder obtained by giving input In0= 0000000000000110000100011100010000011001 (3.0347), In1 = 0000000000000011011001001010100011000001 (1.6966), In2 = 0000000000110110011100111000000010 (0.2127) and obtain output out = 0000000000000100111100011010100111111100 (4.9440) is shown in Fig 16 and Fig 17.



```
/aa/in0   0000000000000110000100011100010000011001
/aa/in1   0000000000000011011001001010100011000001
/aa/in2   0000000000110110011100111000000010
/aa/out0  0000000000000100111100011010100111111100
/aa/tmp1  0000000000000100111100011010100111111100
/aa/tmp   0000000000000100101110110011011001111010
```

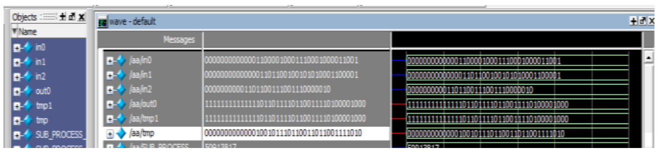
Fig.16. Simulation Results of Three Operand Adder



```
in1 =

    3.0347

in2 =

    1.6966

in3 =

    0.2127

ACTUAL RESULTS

add_out =

    4.9440

HDLDaemon socket server is running on port 4999 with 0 connections
HDLDaemon server was shutdown
VERILOG  RESULTS (CORDIC)

add_val =

    4.9440
```

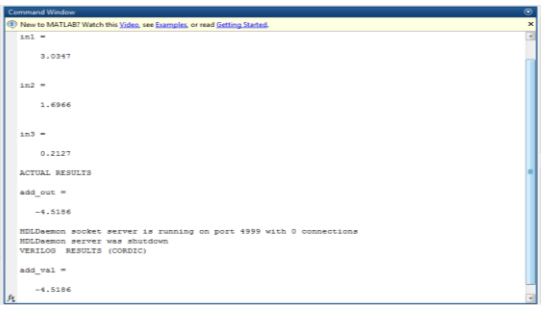
Fig.17. Simulation results of Three Operand Adder verification using MATLAB.

The simulation results of Three Operand Add-Subtractor obtained by giving input In0= 0000000000000110000100011100010000011001 (3.0347), In1 = 0000000000000011011001001010100011000001 (1.6966), In2 = 0000000000110110011100111000000010 (0.2127) and obtain output out = 1111111111111110110111101100111101000001000 (-4.5186) is shown in Fig 18 and Fig 19.



```
/aa/in0   0000000000000110000100011100010000011001
/aa/in1   0000000000000011011001001010100011000001
/aa/in2   0000000000110110011100111000000010
/aa/out0  1111111111111110110111101100111101000001000
/aa/tmp1  1111111111111110110111101100111101000001000
/aa/tmp   0000000000000100101110110011011001111010
```

Fig.18. Simulation Results of Three Operand Add-Subtractor

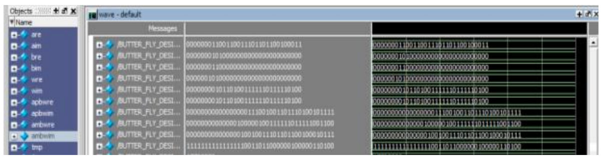


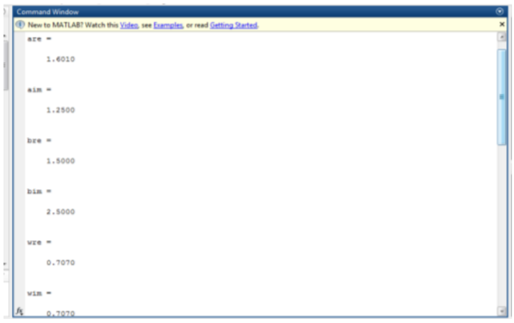
```
in1 =

    3.0347

in2 =

    1.6966

in3 =

    0.2127

ACTUAL RESULTS

add_out =

   -4.5186

HDLDaemon socket server is running on port 4999 with 0 connections
HDLDaemon server was shutdown
VERILOG  RESULTS (CORDIC)

add_val =

   -4.5186
```

Fig.19. Simulation results of Three Operand Add-Subtractor verification using MATLAB.

The simulation results of proposed butterfly architecture obtained by giving inputs are
are= 00000001100110011101101100100011 (1.6010),
aim = 00000001010000000000000000000000 (1.2500),
bre = 00000001100000000000000000000000 (1.5000),
bim = 00000010100000000000000000000000 (2.5000),
wre = 00000000101101001111110111110100 (0.7070),
wim = 00000000101101001111110111110100 (0.7070).
and obtain outputs are
apbwre=00000000000000000111001001101110100101111 (0.8940),

apbwim = 0000000000000001000001001111110111111001100 (4.0780), ambwre = 0000000000000001001001110110110010001011 (2.3080) ambwim = 11111111111111110011011000000010000001101 00 (-1.5780) is shown in Fig 20 and Fig 21 and Fig 22.


```
Objects                wave - default
Name                   Messages
are      /BUTTER_FLY_DESI...  00000001100110011101101100100011
aim      /BUTTER_FLY_DESI...  00000001010000000000000000000000
bre      /BUTTER_FLY_DESI...  00000001100000000000000000000000
bim      /BUTTER_FLY_DESI...  00000010100000000000000000000000
wre      /BUTTER_FLY_DESI...  00000000101101001111110111110100
wim      /BUTTER_FLY_DESI...  00000000101101001111110111110100
apbwre   /BUTTER_FLY_DESI...  0000000000000000111001001101110100101111
apbwim   /BUTTER_FLY_DESI...  0000000000000001000001001111110111111001100
ambwre   /BUTTER_FLY_DESI...  0000000000000001001001110110110010001011 1
ambwim   /BUTTER_FLY_DESI...  11111111111111110011011000000010000001101 00
tmp
```
Fig.20. Simulation Results of Proposed Butterfly Architecture


```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
are =

    1.6010

aim =

    1.2500

bre =

    1.5000

bim =

    2.5000

wre =

    0.7070

wim =

    0.7070
```
Fig.21. Simulation results input values of Proposed Butterfly Architecture verification using MATLAB

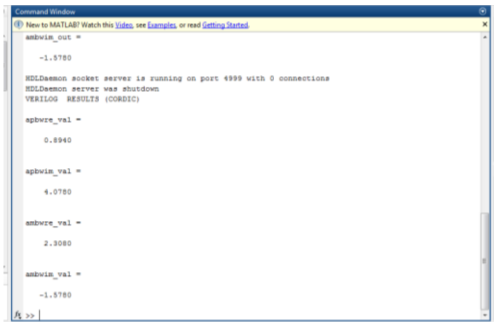

```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
ambwim_out =

  -1.5780

HDLDaemon socket server is running on port 4999 with 0 connections
HDLDaemon server was shutdown
VERILOG  RESULTS (CORDIC)

apbwre_val =

    0.8940


apbwim_val =

    4.0780


ambwre_val =

    2.3080


ambwim_val =

   -1.5780

>> |
```
Fig.22. Simulation result output values of Proposed Butterfly Architecture verification using MATLAB

Table I. Comparision of FP Butterfly Architectures

Parameters	Existing work	Proposed work
Number of Slice LUTs	2168	1044
Number of 4 input LUTs	2576	2024
Number of Bonded IOBs	944	310
Time Delay	60.4ns	53.296ns

IV CONCLUSION

We designed a high speed and area efficient Floating-Point butterfly architecture, which is faster than existing works. The reason for getting lesser area is Multiplier designed by using CORDIC algorithm which reduces the complexity, and reason for speed improvement is FDPA unit.

REFERENCES

[1] Amir Kaivani, SeokbumKo," Floating-Point Butterfly Architecture Based on Binary signed-DigitRepresentation", IEEE Transactions on very large scale integration systems, 2015.
[2] Swartzlander, E. E. Jr. and H. H. Saleh, "FFT Implementation with Fused Floating-Point Operations," IEEE Transactions on Computers, Vol. 61, No. 2, pp. 284-288, 2012.
[3] Sohn, J. and E. E. Swartzlander, Jr., "Improved Architectures for Floating-Point Fused Dot Product Unit," Proceedings of IEEE 21st Symposium on Computer Arithmetic, pp. 38-41, 2013.
[4] Swartzlander, E. E. Jr. and H. H. Saleh, "Fused Floating-Point Arithmetic for DSP," Proceedings of the 42nd Asilomar Conference on Signals, Systems and Computers, pp. 767-771, 2008.
[5] Min, J. H, S. W. Kim and E. E. Swartzlander, Jr., "A Floating-Point Fused FFT Butterfly Arithmetic Unit with MergedMultiple-Constant Multipliers," Proceedings of the 45th Asilomar Conference on Signals, Systems and Computers, pp. 520-524, 2011.
[6] Tao, Y.g. Deyuan, F. Xiaoya and R. Xianglong, "Three-Operand Floating-Point Adder," Proceedings of the 12th IEEE International Conference on Computer and Information Technology, pp. 1920196, 2012.
[7] Winograd, S., "On computing the discrete Fourier transform," Mathematics of Computation, Vol. 32, pp. 175–199, 1978.
[8] Cooley, J.W. and J.W. Tukey, " An algorithm for the Machine Calculation of Complex Fourier Series, " Mathematics of Computations , Vol. 19, No. 90, pp. 297-301, 1965.
[9] Jack E Volder," The CORDIC Trignometric Computing Technic," IRE Transactions on Electronic Computers, Vol. EC-8, NO. 15, PP.330-334 , 1959

BIOGRAPHIES

A. Ramakrishna Raju obtained his B.Tech from Vishnu Institute of Technology Bhimavaram. Currently Pursuing M.Tech in Embedded Systems. In Gudlavalleru Engineering College, Gudlavalleru.

N. Samba Murthy obtained his B.Tech from JNTU Hyderabad and M.Tech from Gudlavalleru Engineering College, Gudlavalleru and currently pursuing Ph.D from JNTU-K, Kakinada in the research area of VLSI Architecture Design. Areas of interest are Microcontrollers, Embedded System Design, Microprocessors and VLSI architecture design. He is a member of IE. Presently working as Assistant Professor in E.C.E Department, Gudlavalleru Engineering College, Gudlavalleru.

Dr. M. Kamaraju obtained his B.E and M.E from Andhra University and Ph.D from JNTU-H, Hyderabad, In the area of Low Power VLSI Design. Areas of Interest are Microprocessors, Microcontrollers, Digital System Design, Embedded System Design, Low Power VLSI Design. He Published 74 technical papers in national and international journals and conferences. He reviewed number of papers for international journal and conferences. He is a Fellow of IETE and IE and member of IEEE. Presently working as Professor and Head of the E.C.E Department, Gudlavalleru Engineering College, Gudlavalleru, India. He is a past chairman of IETE and present chairman of IE Vijayawada local centre, Andhrapradesh.