A new design and implementation of the butterfly unit on FPGA

Yang Jun Ding Jun Li Na Guo Yixiong
School of Information Science and Engineering, Yunnan University
Kunming, China
junyang@ynu.edu.cn

***Abstract*— This paper presents a new method to implement the butterfly unit of FFT processor, based on the coordinate transformation and CORDIC algorithm. The butterfly unit uses less resource and reaches higher speed, also can be configured by parameter and satisfied by the real time operations. Finally, it can work on the frequency of 50 MHz after the design is downloaded to a chip EP2C20F484C7.**

Key Words-butterfly unit; CORDIC algorithm; pipeline; FPGA

I. FOREWORD

The CORDIC (Coordinate Rotations Digital Computer) algorithm, which was first proposed by J.Volder, is able to complete multiplication, radication, looking up trigonometric function tables and complex computing such as anti-trigonometric function through simple bit shift and additional iterative operations. At the same time, it can decompose complex computing which is difficult to be directly achieve by using hardware circuits into unified simple bit shift and additional iterative calculation. Furthermore, its structure is regular and computing cycle is predictable and it is suitable for hardware implementation. On the domestic application, this algorithm is mainly used for computing the value of trigonometric function [1-3] and replacing the multiplication [4] and so on. In this paper, the research object is FFT butterfly computing unit. It converts the complex multiplication and addition in butterfly computing into addition (module 1) under polar coordinates and rectangular coordinates respectively, through using the advantages of CORDIC algorithm in the computing of vector and simplifying the computing of rotation factor. Compared with traditional methods of butterfly computing unit, it does not depend on the size of trigonometric functions' lookup table, and support FFT processing of high point and sample rate, and reduce computing complexity without needing multiplication.

II. ALGORITHM INTRODUCTION

CORDIC algorithm has two ways which are Rotation and Vectoring. And it also can be divided into three ways: circular coordinates, hyperbolic coordinates and linear coordinates. Under circular coordinates, the principle of plane coordinate rotation in the algorithm is stated as the following (In figure 1): after the vector (Xi, Yi) rotates angle θ, a new vector (Xj, Yj) is generated, and according to the trigonometric functions, how the vector (Xi, Yi) rotates to (Xj, Yj) can be expressed as followed [5]:

$$\begin{bmatrix} X_j \\ Y_j \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \quad (1)$$

According to formula (1), it is not difficult to deduce that it supports the iterative operation.

Moreover, equation (1) might be expressed as formula (2) by moving $\cos\theta$ in the right side of the equation.

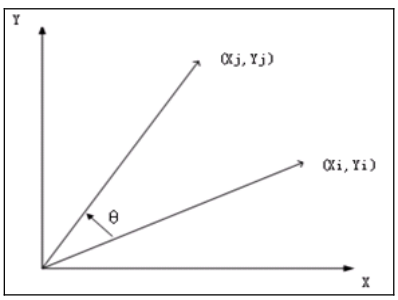

Figure 1. Vector Notation.

$$\begin{bmatrix} X_{n+1} \\ Y_{n+1} \end{bmatrix} = \cos\theta_n \begin{bmatrix} 1 & -\tan\theta_n \\ \tan\theta_n & 1 \end{bmatrix} \begin{bmatrix} X_n \\ Y_n \end{bmatrix} \quad (2)$$

If $\theta_n = \arctan\left(\frac{1}{2^n}\right)$ and put $\cos\theta_n$ on the right side of the equation as the final modified factor K, then the equation (2) can be expressed as (3), (4), (5).

$$\begin{cases} X_{n+1} = X_n - S_n \cdot 2^{-n} \cdot Y_n \\ Y_{n+1} = Y_n + S_n \cdot 2^{-n} \cdot X_n \\ Z_{n+1} = \theta - \sum_{i=1}^{n} \theta_i \end{cases} \quad (3)$$

$$S_n = \begin{cases} +1, Z_n \geq 0 \\ -1, Z_n < 0 \end{cases} \quad (4)$$

$$K = \frac{1}{P} = \prod_{n=0}^{\infty} \cos\left(\arctan\left(\frac{1}{2^n}\right)\right) \approx 0.607253 \quad (5)$$

Since the algorithm achieves converged state when iteration N is equal or grater than 16 [5], therefore, trigonometric functions $\sin\theta$ and $\cos\theta$ can be computed after the vector (1, 0) rotates angle θ.

Similarly, if a given vector (X, Y) rotates to (Xn, 0), the Vectoring can be expressed as the following equations (5), (6), (7). The final computing results in formula (6) are the module and argument of initial vector (X, Y).

In this paper, there are two ways to simplify the butterfly computing unit in FFT processing. One is to achieve the

$$\begin{cases} X_{n+1} = X_n - S_n \cdot 2^{-n} \cdot Y_n \\ Y_{n+1} = Y_n + S_n \cdot 2^{-n} \cdot X_n \\ Z_{n+1} = Z_n - S_n \cdot \theta_n \end{cases} \quad (6)$$

$$S_n = \begin{cases} -1, Y_n \geq 0 \\ +1, Y_n < 0 \end{cases} \quad (7)$$

conversion from plane coordinates to polar coordinates through the vectoring and complete the complex multiplication under polar coordinates. The other is to achieve the conversion from polar coordinates to plane coordinate through the Rotation and complete the complex addition under plane coordinates. In the whole butterfly computation, the traditional four multiplication and six addition/subtraction operations are simplified into one addition operation (mode 1) under the polar coordinates and two addition/subtraction operations under rectangular coordinates. In the design of the butterfly computing unit, on the one hand, this paper needs to resolve the delay caused by iteration, and on the other hand, it also needs to deal with the accuracy of modified module because the revising enhance is related to the iterations.

III. SYSTEM DESIGN AND IMPLEMENTATION

A. The Overall Design

The function of butterfly computing unit is shown in Figure 2. To the inputs A and B and the rotation factor W, it can generate the outputs A + BW and A-BW .If A and B are the complex, the butterfly computing needs four multiplication and six addition /subtraction operations. As the complexity of butterfly computing unit, its realization needs to consume more hardware resources to achieve requirements in precision, speed and throughput. This paper proceeds from the complex operation rules, because the form of complex module is relatively simple in the complex multiplication and it only needs module multiply and arguments add up. And as a result of the form in real and imaginary parts is relatively simple in the complex addition and it only needs addition /subtraction between corresponding real parts and imaginary parts. Considering characteristics of the application in butterfly computing unit and the rotation factor W's mode is 1, butterfly computing unit is designed to be based on the CORDIC algorithm. Its structure is shown in Figure 3.

Suppose A=a+ib, B=c+id, the data flow in new butterfly computing unit is: first of all, input A is saved to the proprietary register and B enters the module R2P to convert from rectangular coordinates to polar coordinates. After dealing with the module R2P the complex B can be directly added up the argument with rotation factor W. Then the adding result between B and W enters module P2R to convert from the polar coordinates to rectangular coordinates. Finally,

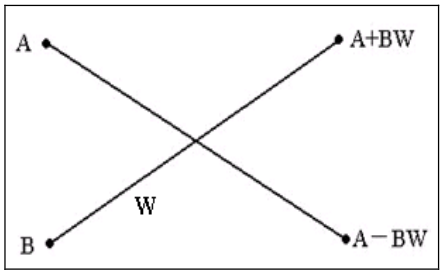

Figure 2. Butterfly Computing Unit

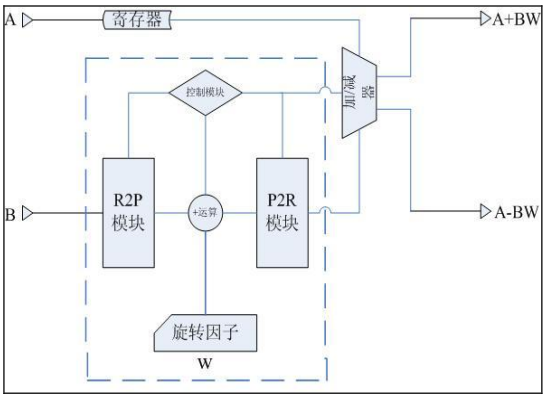

Figure 3. New Butterfly Computing Structure

under the control signals in the control module the addition and subtraction operations between A + BW and A-BW are completed by the addition/subtraction browser. As a result of its own characteristics of CORDIC algorithm, it only needs bit shifter and addition/subtraction browser in the design. In the view of requirements in throughput, the whole design uses pipeline technology. The following contents show the various design features of each sub-module.

B. Module CORDIC

Module CORDIC ， that is within the dotted line in Figure 3, mainly contains module R2P, module P2R and the module rotation factor. Furthermore, the module R2P and module P2R, which are corresponding to the Rotation and Vectoring in CORDIC algorithm, both use pipeline structures. The advantages of pipeline structure are improving data throughput and able to effectively cover up the delay caused by iterations in CORDIC algorithm, each cycle carries out a new operation to generate a computing result except that the first data computing cycle is a little longer. In every level pipeline design, to reduce the consumption of resources brought about by look-up table method, bit shift and angle θ are often constants. Taking into account the impact of the modified value caused by iterative end conditions in the Vectoring and Rotation two different ways, both the two modules require a pipeline state word to track the state of every level pipeline. At the same time, rotation factor module is in charge of generating an angle of iterative factor and replacing look-up table method in trigonometric function with argument generator, which makes the butterfly computing unit is independent of the size of lookup table of Trigonometric function, and hence it supports high point and sample rate.

Take the case of Module R2P (In figure 4). Its inputs are made up of the input clock CLK, enable signal ENA, reset signal RST, real part Xi and imaginary part Yi. The data-processing module is made up of the pretreatment unit r2p_pre, pipeline unit r2p_CordicPipe, and the modified processing unit r2p_post. Its outputs which are made up of module R and argument Z are the state signal ps, real part X, imaginary part Y and the argument Z after the initial data is processed by unit r2p_pre and they are delivered to the unit r2p_CordicPipe for further processing. The unit R2p_Cordic-Pipe is a 16 line pipeline and its processing are unmodified module X and argument Z and the pipeline state word State. These signals after the processing of unit r2p_post output the module R and argument Z of complex $Xi + i \cdot Yi$. The structure and data flow in module P2R are similar to those in the module R2P.

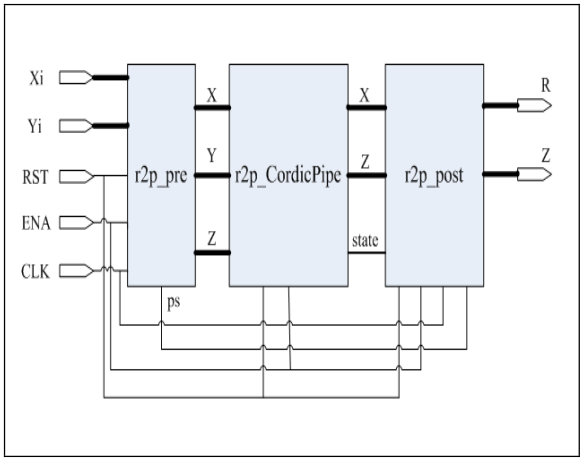

Figure 4. The Structure of Module R2P

C. The Pipeline State word

From the formula (5) we can see the revising enhance of modified module is related to the iterations. In particular, when the iterations are very small, the actual modified value is very different from the converged constant Kn=0.607253, seeing the statistical data in the table 1. It requires to track the states of all levels Pipeline, and know the iterations clearly to every input, so that it can reduce the error of modified value. Table 1 offers the ways of modified value and multiplication approximately carried out by bit shift and addition and subtraction [7].

From table 1 we can see in the first several times the modified values are very different from what they are under the iterative end condition, particularly in the first 5 times, so the pipeline in the first 5 levels is designed to an additional output bit which is used to mark arriving iterative ending condition. At the same time it designs a 5-bit outputting synchronous signal of pipeline state, whose relationship with iterations is shown in table 1. So the circuit of pipeline state word could be made up of a set of selector，the trigger D and a AND gate. The circuit of tracking the pipeline state word is shown in figure 5.

TABLE 1. ITERATE TIMES AND REVISES VALUE

iterate times	revises value	method	Pipeline State
0	1	R	111112
1	0.7071	R((1/2+1/4)-(1/32+1/128+1/512))	111102
2	0.6325	R(1/2+1/8+1/128)	111002
3	0.6136	R((1/2+1/8)-(1/128+1/256))	110002
4	0.6089	R((1/2+1/8)-(1/64+1/1024))	100002
5	0.6072	R((1/2+1/8)-(1/64+1/512))	000002

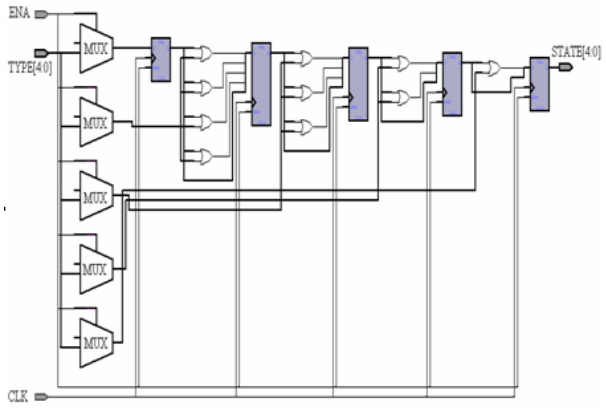

Figure 5. Pipeline State Word

IV. SYNTHESIS AND SIMULATION

A. Synthesis

The parameter in this design is set to a 16 fixed-point. This design takes Altera EP2C20F484C7 as the target chip and carries out the synthesis and layout and routing on the platform of Quartus II. Its main results are shown in table 2. From the point of highest-frequency clock this butterfly computing unit simplicity and efficiency on CORDIC algorithm.

B. Simulation

For the butterfly computing unit, post-simulation is carried on the platform of ModelSim. The I/O are expressed as complement code and used a 16 bit fixed-points decimal fraction, and the weights of low eight bits are 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256. Simulation data and waveform are shown in table 3 and in figure 6.

The Simulation result (in table 3) contains input and output. The Input part includes input complex which is composed by real part and imaginary part and rotation factor which is becoming the argument. The output part is the same type To make the test data representative, we select typical

TABLE 2. PERFORMANCE INDICATOR OF BUTTERFLY UNIT

Total logic elements	3709
Total memory bits	82
Fmax	126MHz

TABLE 3. SIMULATION RESULT

input			output	
real part	Imaginary part	argument	real part	imaginary part
hex			hex	
0100	0000	−45°	00B6	FF4A
0100	0100	−45°	016B	0000
0000	0100	−45°	00B6	00B6
FF00	0100	−45°	0001	016A
FF00	0000	−45°	FF4A	00B6
FF00	FF00	−45°	FE96	0001
0000	FF00	−45°	FF4A	FF4A
0100	FF00	−45	FFFF	FE96
decimal			decimal	
1	0	−45°	0.7109375	-0.7109375
1	1	−45°	1.41796875	0.0
0	1	−45°	0.7109375	0.7109375
-1	1	−45°	0.00390625	1.4140625
-1	0	−45°	-0.7109375	0.7109375
-1	-1	−45°	-1.4140625	0.00390625
0	-1	−45°	-0.7109375	-0.7109375
-1	-1	−45°	-0.00390625	-1.4140625

coordinate point and make rotation factor as a constant. Similarly, table 3 also can be divided into top part expressed as hexadecimal and bottom part expressed as decimal. It is not difficult to see the result error is very small. For example, the result is $\sqrt{2}$ ≈ 1.41421356 after 1+ i rotates 45-degree clockwise, and its calculation result is 1.41796875. Their absolute error is just 0.00375519 and the relative error is about 0.00266.

Figure 6 shows the simulation waveforms corresponding to table 3. The xx and yy are the output data, respectively represents the real part and imaginary part of table 3. Considering the actual validation clock of the ultimate target chip, the simulation clock cycle is set to 50 MHz. It is not difficult to see that there are some delays and glitches in the output waveform, but they don't affect the calculation results. This also reflects the characteristics of the circuit.

V. VALIDATE

Download the document of layout and routing to the Cyclone the II EP2C20F484C7 chip, and derivate the output signal xx and yy. Check the outputs which are data signal and timing signal whether satisfy the requirements or not through the Tek logic analysis instrument (TLA5200). The timing result is shown in figure 7, among which the CK0 is the global clock (50 MHz), C3 represents eight high bits of XX and C2 represents eight low bits of XX, A3 represents eight high bits of YY and A2 represents eight high bits of XX. Compared figure 6 with figure 7, we can see simulation waveform is in accord with the timing waveform that is the butterfly computing unit can be in proper working order under the clock frequency is 50 MHz.

The feature of fixed-point butterfly computing unit in this design is that the data bit can be under control of parameter. Compared with the butterfly computing unit in [8], its advantage of high sample rate in the FFT processing is very obvious because it doesn't need to save the value of rotation factor. Furthermore, compared with the butterfly computing unit in [9], its advantage is in hardware resources consumption because it doesn't directly use multiplier. The following is comparing the consumed logic units (in figure8) between this design and those in [9] and the statistical data is

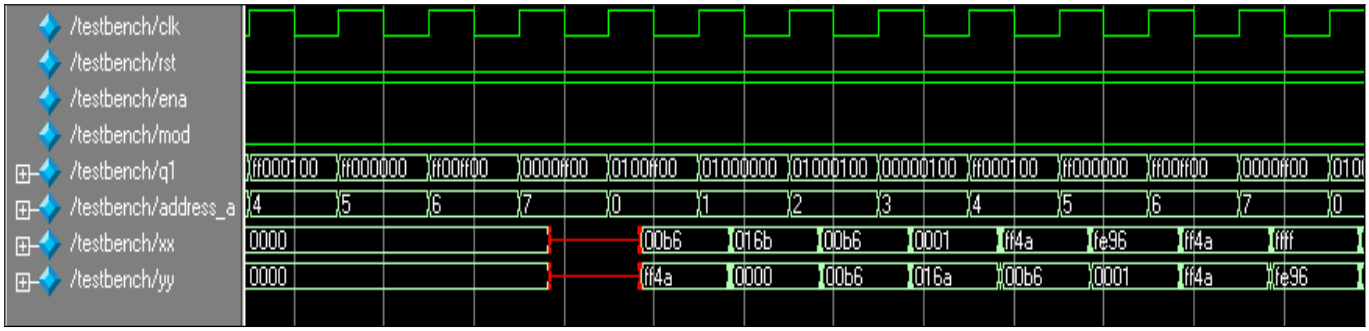

Figure 6. Simulation Result

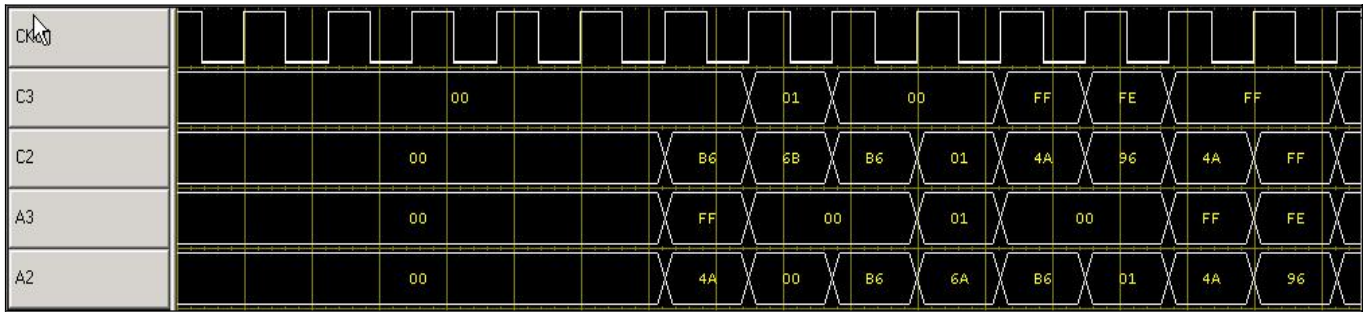

Figure 7. Measurement Result

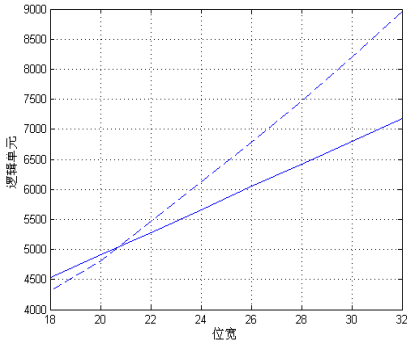

Figure 8. Comparison of Resource Consumption

obtained after synthesized by Quartus II. In the figure 8, the solid line represents the butterfly computing unit in this design and the dotted line represents those in [9]. Obviously, the design has advantage of consuming less resource in high bit. And the experiment's data provided a certain reference for further research float butterfly computing unit (in line with IEEE-754 standard).

VI. CONCLUSION

Based on the CORDIC arithmetic, this paper adopts parameterized control system for precision, and raises the throughput with pipeline technique, replaces lookup table of Trigonometric function with argument generator that makes butterfly computing unit in this design not to rely on its size. So it supports high point and high sample rate, its each part is designed briefly and its structure is regular, it is easily implemented by using hardware. It has a high processing speed and high throughput and the system is stable .This is a new method of hardware realization in butterfly computing unit.

REFERENCES

[1] Tian Shulin, Wang Houjun, Xu Hong-bing, “A Research of Signal Generator Technology Based on CORDIC Algorithm”, Chinese Journal of Scientific Instrument, pp.150-153, May 23th,2002.

[2] Yu Zhou, Shao Nan, “Design of Digital Filter Based on CORDIC Algorithm”, Microelectronics & Computer, pp.10-12, May 24th, 2007.

[3] Guo Lihao, Duan Zhemin, Bai Sen, “Design of a direct digital frequency synthesizer based on CORDIC algorithm”, Electronics Optics & Control, pp.77-79, May 13th,2006.

[4] Li Chengshi, Chu Jianpeng, Li Xinbing, “A High Speed Real-time and Fixed-point FPGA Realization of a CORDIC Based FFT Processor”, Microelectronics & Computer, pp.88-91,Apirl 21th,2004.

[5] Richard Herveille, “Cordic Core Specification". http://www.opencores.org/projects.cgi/web/cordic

[6] Ray Andraka, “A survey of CORDIC algorithms for FPGA based computers”, Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field programmable gate arrays, Monterey, CA, pp.191-200,February,1998.

[7] Xia yuwen, Verilog Digital System DesignTutorial, Beijing University of Aeronautics and Astronautics Press, 2004.

[8] Han Zeyao, HanYan, Zheng Weimin, “The Design of a High Speed Real-time and Fixed-point FFT Processor”, Journal of Circuits and Systems, pp.18-22, March 7th, 2002.

[9] Zhao Ming, “radix 4 complex fft”. http://www.opencores.com/projects.cgi/web/cfft