# Design and Implementation of a 1024-point High-speed FFT Processor Based on the FPGA

Sheng Zhou, Xiaochun Wang, Jianjun Ji, Yanqun Wang
Institute of Biomedical Engineering
Chinese Academy of Medical Sciences
Tianjin, China

***Abstract: To design a Fast Fourier Transform (FFT) processor to meet the needs for high-speed and real-time signal processing. A 1024-point, 32-bit, fixed, complex FFT processor is designed based on a field programmable gate array (FPGA) by using the radix-2 decimation in frequency (DIF) algorithm and the pipeline structure in the butterfly module and the ping-pone operation in data storage unit. When the primary clock is 100 MHz, the 1024-point FFT calculation takes about 62.95 us. The processor is fast enough for processing the high-speed and real time signals. The result provides reference values that theoretical study of the FFT algorithm can be applied into the adaptive dynamic filter of an ultrasonic diagnostic system and an ultrasonic Doppler flow measurement system.***

***Key words— field programmable gate array; 1024-point FFT; Butterfly; Ping-pong operation; Verilog HDL***

## I. INTRODUCTION

FFT (Fast Fourier Transform Algorithm) is a fast algorithm of DFT (Discrete Fourier Transformation). A real time processing of digital signal was becomes realization when the FFT was appeared, and which was used in many domains such as the ultrasound, the radar and the communication. In the process of ultrasonic echo signals processing, the FFT is a most important part. Its computing speed and accuracy are directly influence the performance of the system. FFT is a centre part of the Doppler blood flow spectrum analysis, and is the most commonly used signal processing method in the Doppler imaging system.

Although the Altera and Xilinix have developed the corresponding FFT IP (Intellectual Property) core, the price of them is so high that they are not used extensively. As a result, many researchers now design the FFT processors which are more suitable for themselves. In the engineering realization of FFT, the most generally hardware realization methods are included of DSP (Digital Signal Processor), FFT dedicated chip and FPGA [1]. The FFT operation in DSP not only occupies much more time, but also reduces the throughput rate of the whole system, and furthermore, the realization flexibility of the DSP software can not make well. By using dedicated FFT processing chip, the speed can achieve the requirement, but has bad expansibility. With the rapid development of the FPGA recent years, and comparing with the DSP technique, the FPGA is suitable for the high-speed signal processing system owing to its parallel signal processing architecture.

Combining the advantages of flexible programming in the software and high-speed by the dedicated integrated circuit, FPGA is suitable for the FFT algorithm and has superiorities in performance, costing and power consumption. It is realized by the related EDA (Electronic Design Automation) software and hardware description language. In modern signal processing, the requirements of high-speed and high-reliability are becoming a hot research point. Furthermore, many researchers are researching in combining the real time requirement of FFT with the flexibility design by the FPGA, realizing the optimal configuration of the parallel algorithm and hardware structure, and improving the FFT processing speed.

According to the actual requirements, this paper presents a method which realizes the FFT operation based on the FPGA. The FFT synthesis and simulation are realized on the Quartus II 6.0 and Modlesim 6.0 software. Although at last we only realize the 1024-point fixed-point arithmetic, this method can be also applied to 2M (M=2，3…) points arithmetic.

## II. THE BASIC PRINCIPLE OF THE RADIX-2-DIF FFT ARITHMETIC

The FFT arithmetic can be divided into two types basically, which is the decimation-in-time (DIT) and the decimation-in-frequency (DIF). This radix-2-DIF is adopted in this paper. The discrete Fourier transformation of the sequences x(n) is written as,

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} \qquad k,n=0,1,2\ldots,N-1 \quad W_N=e^{-j\frac{2\pi}{N}}$$

In which N is the sequence length. Then the x(n) is divided into two parts, which is include of the odd number part and the even number part. By this method, the N/2-point DFT can be continued to divided into the N/4-point DFT, the N/8-point DFT and lastly obtained the 2-point DFT [1]. Figure 1 the frequency decimation operation diagram where N=8.

Support by the National Science and Technology Pillar Program (2012BAI13B02), the Fundamental Research Funds for the Central Universities and the PUMC Youth Fund

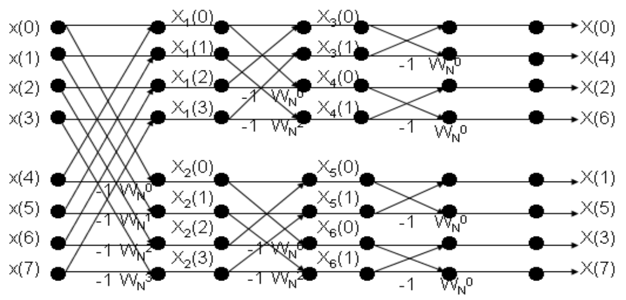


Fig1.The operation diagram of DIF-FFT where N=8

## III. DESIGN OF THE HARDWARE AND IMPLEMENTATION OF THE FFT ARITHMETIC

### A. The structure of the FFT processor

The whole structure of FFT processor is divided into several modules reasonably. Each module is worked parallel by the sequential control unit. The whole implementation diagram is shown in figure 2 which is included of a butterfly processing unit, a data storage RAM and a ROM unit, a address generating unit and a sequential control unit [2]. The design of the butterfly processing unit and the address generation unit is important for the design of fixed-point FFT. The implementation methods of each unit are introduced in the following section.

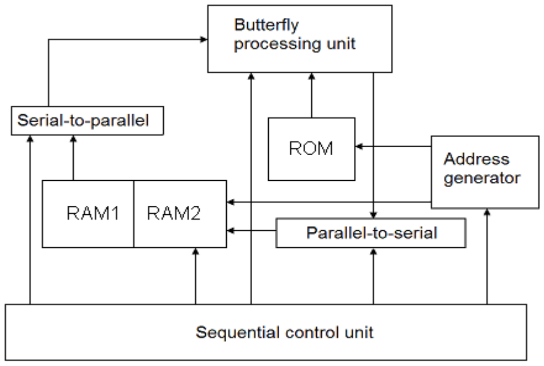
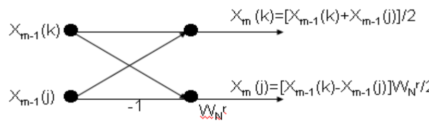


Fig.2 The whole system diagram

### B. The butterfly processing unit

The Butterfly processing unit is the core of the FFT arithmetic. The butterfly element is shown in figure 3. It is needed ten steps processing for 1024-point data, and each step is composed of N/2 butterfly processing elements. The operation of each butterfly processing element is as follows,

$$X_m(k) = X_{m-1}(k) + X_{m-1}(j)$$
$$X_m(j) = [X_{m-1}(k) - X_{m-1}(j)]W_N^r$$



$X_{m-1}(k)$
$X_{m-1}(j)$
$X_m(k)=[X_{m-1}(k)+X_{m-1}(j)]/2$
$X_m(j)=[X_{m-1}(k)-X_{m-1}(j)]W_N^r/2$
-1
$W_N^r$

Fig.3 The basic butterfly element

As shown in figure 3, the butterfly processing element is composed of a complex adder, a complex subtractor and a twiddle factor complex multiplier [3]. The twiddle factor multiplier is implemented by real number multiplication with 4 times and addition or subtraction with 2 times. Let the input data and twiddle factor are:

$$A = Are + jAim, B = Bre + jBim, W_N^r = Wre + jWim$$

Figure 4 is the hardware implementation diagram based on the FPGA.

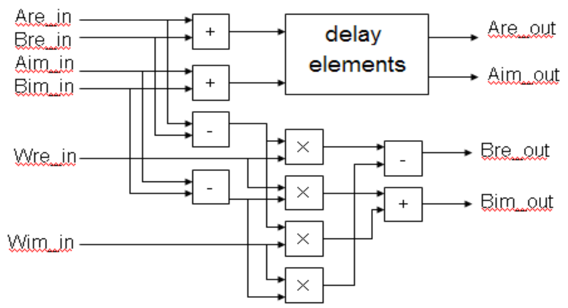


Fig.4 The hardware implementation diagram of butterfly element

The adder, the subtractor and the multiplier in this butterfly processing element are generated by the Megafunction in Quartus II. The operation of the butterfly arithmetic is in synchronization and needs 4 clock cycles. To avoid the data overflow, the output results are divided by the number 2 [4]. For the division operation consumes much more resources, in this design, we adopt the shifting method of binary numbers to implement the division operation. At the same time, the Xm(k) is delayed by a group of registers for data synchronization. The output result of the multiply-accumulate operation is remained 16-bit integer output [5-6].

### C. The data storage ROM/RAM

The data of the twiddle factor ROM are designed by the MATLAB. The real part and imaginary part of the input data A and B are all 16-bit signed binary data. The twiddle factor cos(2k∏ / N) and sin(2k∏ / N) are quantified to 16-bit signed binary data and saved to a ROM. The operation result is divided by 16'b0111111111111111 which is multiplied when quantization.

The data storage RAM is used to store input data and to buffer intermediate operation results. The input/output data are needed to write or read respectively form the RAM in each butterfly operation. Before the next transformation, the last result should be written to the RAM where read the data from. In order to achieve high speed FFT processing, in the FFT module, we use two RAM storage blocks to compose the typical Ping-pong operation module [7-8].RAM 1 and RAM 2 are used for storing the intermediate data in which one RAM is used as a reading block and the other is used as a writing block. In the first step of FFT, the data reading from RAM 1 are saved into RAM 2 after the butterfly operation element. The second step is just the opposite. In the same way until the tenth step is implemented. Figure 5 is the Ping-pong operation diagram.

The RAM we adopted is using True DualPort Memory module. The reading and writing operations with random addresses can be implemented under independent working clock in each side. It is meaning that, the data can be read out from two RAMs, for the butterfly operation and the results can write into two RAMs in the same time. This method greatly reduce the writing and reading cycle compared to single port RAM.

For using the dual-port RAM, when one storage element is used for data reading output, this storage element can save the data from the upper level. The pipeline structure is suitable for this FFT structure, and can realize the continuous operations to signal samples. The intermediate data are saved to a dual-port RAM. Then the two complex data from the RAM are split to four real data and then are inputted to the butterfly operation element. After the butterfly operation, the data are input into the selecting buffer to constitute two complex data and lastly to output. The serial-to-parallel and the parallel-to-serial are realized [4].

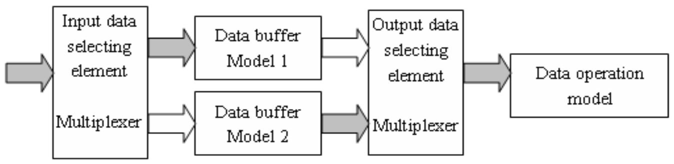


Fig.5 The ping-pong operation diagram

## D. *The addresses generator*

The addresses generator generate three group addresses, which is included of the reading addresses of RAM and ROM for butterfly operation data, the writing addresses of RAM for butterfly operation results and the lastly output addresses of FFT operation results. In actually, the reading addresses are the delay of the writing addresses. The delay time is the adder of the butterfly operation time and the parallel-to-serial time. The reading addresses are included of the real part and the imaginary part, as well as the twiddle factor addresses for butterfly operation. The addresses generator is composed of the counter and some logic parts [9-10].

The addresses generation is related to the step counter and the butterfly operation counter in each step, so we design two counters which are respectively used for the steps counting and the butterfly operation counting. The shifter registers are used to count the steps and the butterfly operation counter is in the general method. When the butterfly operation counter is full, there is a pulse for the steps counter, and the shifter is worked. If there is no derive pulse, the steps counter is remain unchanged. If the shifter work is finished, the butterfly operation addresses generating procedure for FFT is completed.

The logic parts of the addresses generator is composed of upper and lower notes and the address logics of twiddle factors which are A_address, B_address and rom_address respectively.

## E. *The sequential control unit*

The sequential control unit is used to generate control signals for each model. Its main task is exactly to coordinate work and consequently accomplish the whole procedure of the FFT operation. In according to many processing elements related to FFT operation and the complex controlling, in this unit, we adopt the modeling of Mealy limited state machine.

The function of the sequential control unit is described as: first, enter the initial state, and when the external reset signal is valid, all of the functional modules are reset; when the enable signal is in valid, the inputting control unit is in inputting state and generate the enable signal to receive the data which is intend to operate; the following is the butterfly operation state; when the fix-point operation is completed, enter the output control state and send out the output enable signal; when the intermediate operation data are read from the RAM, the completed signal is generated if all of the data is outputted and changed to initial state; the following is the delay operation and prepare for the next step operation.

## IV. THE WAVEFORM SIMULATION VERIFICATION AND THE SEQUENTIAL ANALYSIS

Each functional module is analyzed and realized above mentioned. The FFT processor is synthesized on the top module as figure 6.

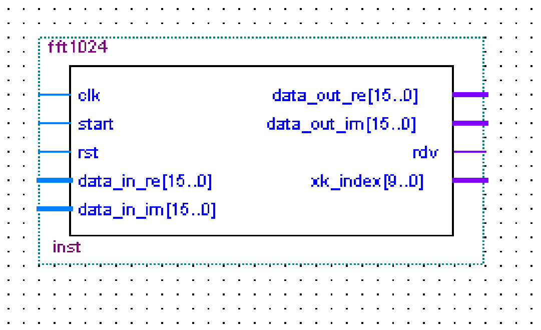
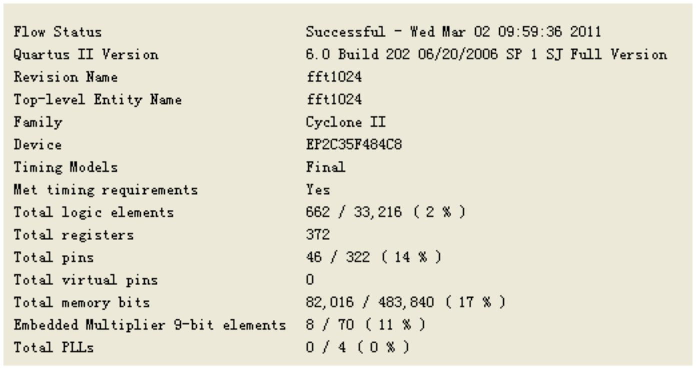


Fig.6 The top module of the FFT

The system is developed on the software QuartusⅡ6.0 offered by ALTERA. The chip we selected is EP2C35F484C8. Figure 7 is the compiling result.



| | |
|---|---|
| Flow Status | Successful - Wed Mar 02 09:59:36 2011 |
| Quartus II Version | 6.0 Build 202 06/20/2006 SP 1 SJ Full Version |
| Revision Name | fft1024 |
| Top-level Entity Name | fft1024 |
| Family | Cyclone II |
| Device | EP2C35F484C8 |
| Timing Models | Final |
| Met timing requirements | Yes |
| Total logic elements | 662 / 33,216 ( 2 % ) |
| Total registers | 372 |
| Total pins | 46 / 322 ( 14 % ) |
| Total virtual pins | 0 |
| Total memory bits | 82,016 / 483,840 ( 17 % ) |
| Embedded Multiplier 9-bit elements | 8 / 70 ( 11 % ) |
| Total PLLs | 0 / 4 ( 0 % ) |

Fig.7 The compiling result by the Quartus II 6.0

From the figure 7, we can see that the total logic elements are only used 2% and the total memory bits are only occupied 15%. The saving use of logic elements meets the system requirement of rational utilization of FPGA resource.

In order to verify the accuracy of the system designed, the simulation result is simulated digitally in Modelsim6.0 [11], as seen in figure 8. The output is in valid when the signal rdv is high. When the clock frequency is 100 MHz, it needed 62.95 us to finish a 1024-point 32-bit fixed point complex FFT transformation.

The verification of algorithm is finished by the Matlab 7.0. The input signal is x(t)=20000*sin(2*∏*10*t). The output data from the Modelsim 6.0 are saved in .txt document and imported into the MATLAB. Then the data format is changed to the signed decimal from the binary complementary code. The comparison between the simulation data by the Modelsim and the FFT functional computational results by the MATLAB is seen as figure 9.

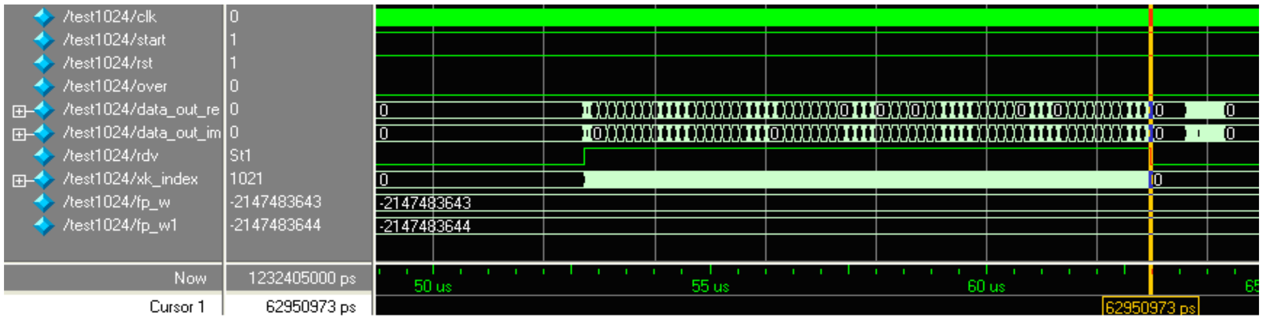


Fig.8 The simulation result by the Modelsim

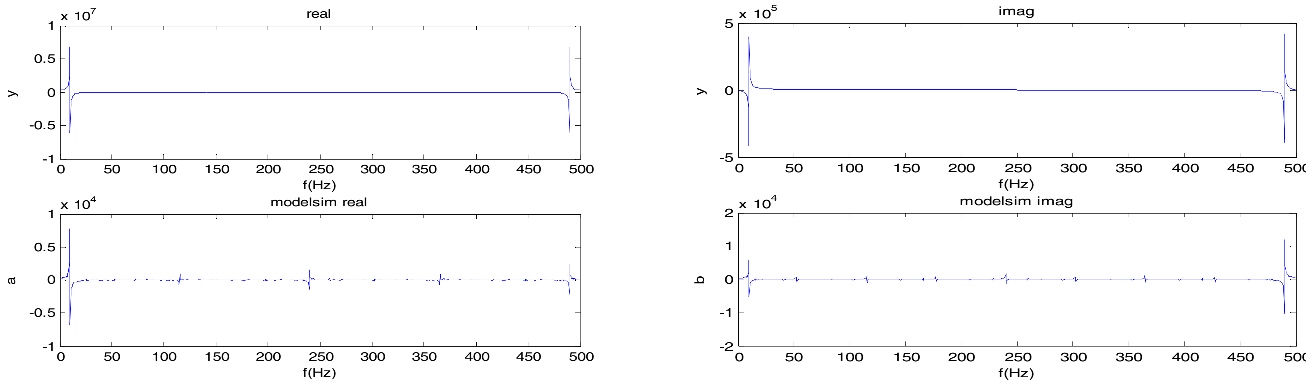


Fig.9 The comparing between the FFT processor result and the FFT functional computational result

Form figure 9, we can see that the FFT processor results and the computational results by the MATLAB are basically the same. The validity of the algorithm is verified. The statistical error analysis between 1024-point simulation results by the Modelsim and the testing data are worked on the MATLAB platform [12-13]. The total average relative error is:

$$\frac{\sum_{i=1}^{1024}\frac{|A(i)-B(i)|}{B(i)}}{1024} \approx 0.0062 = 0.62\%$$

It is seen that the deviation between the two platforms is very small. So the fixed point FFT processor based on the FPGA get a good application which is in the allowable error range.

## V. CONCLUSIONS

Compared to the theoretical values, the errors from the simulation results are quite small. Using the abundant logic resources in FPGA is the superiority of this system. The pipeline architecture and the Ping-pong operation are based on the embedded M4K logic elements and are finally finished the high-speed fixed-point FFT operation. Under the clock of 100 MHz, the 1024-point FFT operation is only used 62.95 us which is reached the high processing requirement. With the improving of the FPGA resources, the superiority of using this structure to realize the FFT operation is more and more significantly. Furthermore, by this structure, it is easy to expand the FFT operation by increasing the depth of RAM and ROM, changing the twiddle factor table and adding the times of the ping-pong operation.

In the design of dynamic filter in the traditional ultrasonic imaging, when to determine the center frequency, it is only taken into consideration that the frequency is attenuated with the depth increased and neglected the nonlinear relationship between depth and frequency. The assumption of fixed attenuation coefficient and the frequency is changed linearly from near field to far field leads to the differences between the frequency characteristics of real echo signals and the theoretical model. Consequently, this way weakens the matched effect of dynamic filter. The fast Fourier transform processor designed in this paper can analysis the frequency characteristics of each small section of the echoes in real time and determine the dynamic filter whose coefficients are according to the center frequency. By this way, it can attain the optimal effect of the matched filter and raise the signal-to-noise ratio. This self-adaptive dynamic filter is very useful for digital ultrasonography and the ultrasonic Doppler flow measurement system.

## REFERENCES

[1] Hu Guangshu. Digital signal processing-theoretical algorithm and realization [M]. Peking: Tsinghua University Press, 2005

[2] Chu Chao, Zhang Qin, Xie Ying-ke, et al. Design of a high performance FFT processor based on FPGA [C]. Proceeding of 2005 Asia and South Pacific Design Automation Conference, Shanghai, January 18-21, 2005

[3] Uwe Meyer-Baese. Digial signal processing based on FPGA (second edition)[M]. Peking: Tsinghua University Press. 2006

[4] Zhu Binglian, Liu Xuegang. FPGA Implementation of Pipelined FFT [J]. Journal of Chongqing University. 2004,27(9): 33-36

[5] Sun Fei, Zhou Ning, Sun Yannan, et al. Hardwire Logic Implementation of a 8-point FFT Algorithm [J]. Microelectronics and Computer, 2002, 19(11): 5-17

[6] Ren Bingyu, Zhan Yinwei. Design of 64-point FFT Processor Based on FPGA [J]. Modern Electronics Technique, 2009, 32(14): 1-3

[7] Liu Guodong, Chen Boxiao, Chen Duofang. Design of FFT Processor based on FPGA [J]. AERONAUTICAL COMPUTER TECHNIQUE, 2004, 34(3): 101-104

[8] Xia Yuwen. Digital System Design Tutorial [M]. Peking: Beijing University of Aeronautics and Astronautics Press, 2004.1

[9] Ma Qiang. Implementation of fast fourier transform on FPGA [D]. Nanjing University of Science and Technology, 2005

[10] Zhang Yu, Fang Kangling. Design of General FFT Processor Based on FPGA [J]. Computer Technology and Development, 2010, 20(8): 87-90

[11] Qiao Lufeng. Verilog HDL digital system design and verification [M]. Peking: Publishing House of Electronics Industry, 2009.4

[12] Shousheng He， Mats Torkelson. Design and Implementation of a 1024-point Pipeline FFT Processor [C]. Custom Integrated Circuits Conferenc.1998

[13] Levent Aksoy, Ece Olcay Gunes. Area optimization algorithms in High-speed digital FIR filter synthesis [M] //SBCCI2008. New York: ACM, 2008: 64-69.