International Journal of Control and Automation
Vol.7, No.6 (2014), pp.177-188
http://dx.doi.org/10.14257/ijca.2014.7.6.18

Design and Implementation of 1-D and 2-D Mixed Architecture FFT Processor in Heterogeneous Multi-core SoC based on FPGA

Duo-li Zhang, Lu Huang, Yu-kun Song and Gao-ming Du

Institute of VLSI Design Hefei University of Technology, Hefei 230009, China

sqmiracle@126.com

Abstract

A novel architecture FFT processor which can carry on 1-D FFT algorithm or 2-D FFT algorithm corresponding different size of FFT is proposed in this paper. The architecture is served as a scalable IP Core which is suitable for the heterogeneous multi-core SoC on chip application. The mixed architecture FFT processor achieves balance between high processing speed and resources. Compared with a conventional 1-D FFT processor, this FFT processor is characterized by having smaller resources consumption; compared with a 2-D FFT processor which is mapped onto a long point FFT architecture, it has higher processing speed. It employs the Radix-2 DIT-FFT algorithm and fixed addressing structure. It takes two ways to hide time consumed on data-path, one is read-ahead operation of dual-port RAM to hide the delay introduced by 12-stages pipeline of butterfly-unit and Memory access delay; the other is Ping-Pong operation of 3 groups of RAM to conceal data transmitting time. It also adopt twiddle factor compression algorithm to reduce ROM space. It can carry out 2^n (n is from range of 5 to 14) single-precision floating-point FFT/IFFT directly. As a result, we have implemented the mixed architecture FFT processor based on FPGA, and successfully applied it into the heterogeneous multi-core SoC.

Keywords: *FFT processor; Radix-2; twiddle factor compression algorithm; read-ahead operation; mixed architecture; two-dimensional FFT operation*

1. Introduction

Fast Fourier Transform (FFT) is the fast algorithm of Discrete Fourier Transform (DFT)[7], it increases DFT's operational efficiency 1~2 orders of magnitude and plays most important roles in digital signal processing (DSP) [9]. It has promoted the development of DSP technology greatly and has gotten extensive application in Engineering.

Nowadays, Fast Fourier Transform (FFT) has been widely applied in many fields such as digital communications, image processing, speech recognition and radar processing. And its field-programmable gate array (FPGA) implementation method in the heterogeneous multi-core SoC has important research value [10]. Compared with design flow of general DSP or ASIC, designs based on FPGA have the advantages of flexibility, low power-consumption, short development time, low cost and high performance price ratio [11]. This paper introduces the design and implementation of a one-dimensional (1-D) and two-dimensional (2-D) mixed architecture FFT processor based on FPGA.

There are many kinds of algorithm pattern, such as Radix-2 FFT algorithm, Radix-4 FFT algorithm, Radix-8 FFT algorithm, Split-Radix FFT algorithm and so on[6][12]. Radix-2 FFT algorithm is suitable for hardware implementation, and it is usually employed to process 2^n-point FFT operation. This design selects Radix-2 DIT-FFT algorithm. Owing to computing

ISSN: 2005-4297 IJCA

cycle is double compared to Radix-4 FFT algorithm, the design employs dual parallel structure of two butterfly operation units. Because of its wide range of uses and own particularity, Radix-2 FFT algorithm not only requires its hardware implementation to achieve fast operation speed, but also takes into account the hardware area, there must be a good balance between them.

References [13-16] have introduced a traditional faster 1-D FFT processor. It has idle processing efficiency , but it needs larger memory. References [1-5] research2-D FFT algorithm to calculate the 1-D long point FFT. It greatly compressed internal storage resources, while the speed is slower than 1-D method. On this basis, this paper presents a 1-D and 2-D mixed architecture FFT processor. The practice shows that mixed architecture FFT processor achieves balance between speed and resources.

This paper is organized as follows. In the following two sections, a traditional 1-D FFT processor and a 2-D long-point FFT processor are introduced to discuss their technical defects. Twiddle factor compression algorithm, the principle of two-dimensional long point FFT algorithm and the read-ahead operation are also described in these two sections. In section IV, the 1-D and 2-D mixed architecture FFT processor is proposed and the detail design of 1-D FFT mode and 2-D FFT mode are given. Section V is the performance comparison of the three FFT processors, and hardware resources and operation speed are explicitly listed out. The last section gives the conclusion.

2. One-dimensional FFT processor

We have designed 1-D FFT processor which employs the radix-2 Decimation-In-Time (DIT) FFT algorithm and fixed addressing structure and gets 32-bit single precision floating point operation from 32 points to 16K points as in [13]. For N-point FFT requires 2N data memory and N / 2 twiddle factor coefficient memory, someone uses twiddle factor compression algorithm to compress the coefficient memory [1]. As shown in Figure 1, it's the diagram of 1-D FFT processor. It integrates FFT controller, memory controller, FFT computing unit and two 4 * 4K on-chip data memory. And, FFT computing unit integrates address generating unit of read-write, two butterfly operation units and the twiddle factor generator.

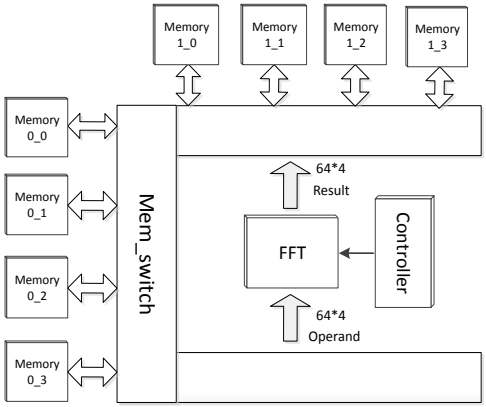

Figure 1. Block Diagram of 1-D FFT Processor

In each clock cycle, the processor will fulfill two butterfly operations parallel, that is to say, it reads four operands and two coefficients from on-chip memory simultaneously. Thus, 4 results are generated and written into on-chip memory.

2.1 The compression algorithm of twiddle factor

For the 16K-point FFT operation requires 8K twiddle factors which need a very large memory. It wastes a lot of resources in the FPGA.As to solve the above problems, reference [1] proposes the twiddle factor compression algorithm in the following equation.

$$W_N^k = W_N^{(\alpha+\beta)} = W_N^\alpha * W_N^\beta \quad (1)$$

We can resolve the twiddle factor's address value into the sum of two parts and calculate each part respectively. Then, we can achieve the ultimate factor generation by complex multiplication. When the number of FFT points N = 128 × 128, the variation range of k will be [0, 8191], k is expressed as:

$$k = 128\alpha + \beta \ (\alpha=0,\ 1,\ \cdots 63,\ \beta=0,\ 1,\ \cdots 127) \quad (2)$$

Then：

$$W_{128\times128}^k = W_{128\times128}^{(128\alpha+\beta)} = W_{128\times128}^{128\alpha} * W_{128\times128}^\beta = W_{128}^\alpha * W_{128\times128}^\beta \quad (3)$$

The first term will be stored in ROM1 which address is αand corresponding stored value is W_{128}^{α}. This table has a size of 64x64 bit. The second term will be stored in ROM2 which address is β and corresponding stored value isW_{N}^{β}. This table has a size of 128x64 bit. In this way, the number of memory storage space is only 192 which are much smaller than 8K.

3. Two-dimensional FFT Processor

3.1 The principle of two-dimensional long point FFT algorithm

The FFT operation of long N-point FFT can be broken into stage cascading of two smaller points FFT operation as in [1-3].

Assuming N = L * C, the input point of N can be described as matrix form by L-row and C-column. Similarly, the output point of N can be expressed as matrix form by C-row and L-column. The details are shown in Figure 2.

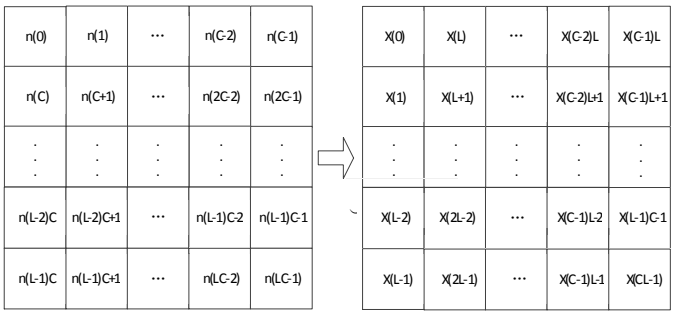

Figure 2. 2-D FFT Operation Transform Matrix

According to the Discrete Fourier Transform (DFT) algorithm formula as in [8]:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad (4)$$

In (4), k can be expressed by k=Lk1+k0 and n can be expressed by n=Cn1+n0.Therefore, we can transform the product x (k) easily as the following equation:

$$X(k_1,k_0)=\sum_{n=0}^{N-1}x(n_1,n_0)W_N^{(Cn_1+n_0)(Lk_1+k_0)}=\sum_{n_0=0}^{C-1}\sum_{n_1=0}^{L-1}x(n_1,n_0)W_N^{Cn_1Lk_1}W_N^{Cn_1k_0}W_N^{Ln_0k_1}W_N^{n_0k_0}$$
$$=\sum_{n_0=0}^{C-1}[\sum_{n_1=0}^{L-1}x(n_1,n_0)W_{\mathrm{L}}^{n_1k_0}]W_N^{n_0k_0}W_C^{n_0k_1}=\sum_{n_0=0}^{C-1}[X_1(k_{0,}n_0)W_N^{n_0k_0}]W_C^{n_0k_1} \quad (5)$$
$$=\sum_{n_0=0}^{C-1}[X_1^{'}(k_{0,}n_0)]W_C^{n_0k_1}=X_2(k_{0,}k_{1,})$$

(k1=0，…，C-1；k_0=0，…，L-1；n_1=0，…，L-1；n_0=0，…，C-1)

In (5), $X_1(k_{0,}n_0)$ expresses L-point FFT which is the column transformation of the matrix. $X_2(k_{0,}k_1)$ expresses C-point FFT which is the row transformation of the matrix. $X_2(k_{0,}k_1)$ expresses C-point FFT which is the row transformation of the matrix. (2)

Based on above formulas, we can divide the FFT operation of long N-point FFT into 1-D column transform of C groups L points and row transform of L groups C points. Thus, long N-point FFT is transformed into 2-D process which can greatly reduce the internal memory.

3.2 The design of two-dimensional FFT processor

Considering the 1-D FFT processor need very large data memory, references [1-5] research on the method which uses a 2-D FFT algorithm to operate the long point FFT. In this way, internal memory has been greatly compressed. On this basis, this paper has applied 2-D FFT algorithm into the1-D FFT processor to design two-dimensional FFT processor. It realizes the FFT/IFFT of any 2^n points of 32-bit single-precision floating-points from32 points to 16K points. 2-D FFT processor increases the transmission module to transfer the 1-D data. Assuming the maximum point of FFT is 16K, it requires 16K on-chip data memory. Considering 2K=16*128, the maximum point of 1-D FFT is 1K. If the number of points is less 2K, the input point of N can be described as matrix form by 1-row and N-column. In order to match computing bandwidth, the FFT processor uses three groups of 4 * 256 dual-port memory organized as Ping-Pong operation to fulfill the 1-D FFT. The 2-D FFT processor architecture is shown in Figure 3.

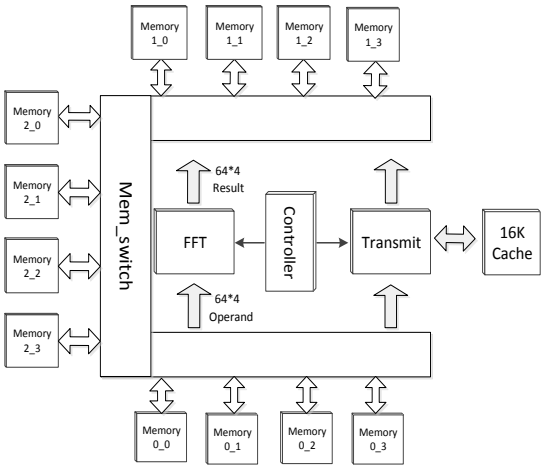

Figure 3. Block Diagram of 2-D FFT Processor

3.3 The principle of Read-ahead operation

In the 2-D FFT processor, the resource consumption of on-chip memory will be greatly reduced. 2-D FFT processor contains two independent butterfly units. N-point FFT operation run $\log_2 N$stages, each stage requires N / 2 butterfly operation, each butterfly unit runs N / 4 times, each time has butterfly unit Computing delay (C_d) and Memory access delay (M_d). Then, N-point FFT operation total delay period is:

$$\begin{aligned} \text{Total\_delay} &= \left[\mathrm{C}\left(\mathrm{L}/4*\log_2 L\right)+\mathrm{L}\left(\mathrm{C}/4*\log_2 C\right)\right]\left(M_d + C_d\right) \\ &= \mathrm{N}/4\left(\log_2 L+\log_2 C\right)\left(M_d + C_d\right) \\ &= \mathrm{N}/4\log_2 \mathrm{N}\left(M_d + C_d\right) \end{aligned} \tag{6}$$

In (6), M_d : Memory access delay; C_d : Butterfly unit computing delay. From the above equation, the total delay periods of N-point FFT operation are ($M_d + C_d$) times as many as the theoretical clock cycles of Radix-2 FFT algorithm with two butterfly units parallel operation. The extra clock consumption caused by the delay period must be reduced.

The 2-D FFT processor can employ read-ahead operation of dual-port memory to hide the pipeline delay of butterfly-unit and Memory access delay. After reading all operands from source memory, FFT computing unit needs to wait ($M_d + C_d$) clock cycles and then it can write all results back into destination memory. Immediately, it continues reading next stage operands. Read-ahead operation reads next stage operands in ($M_d + C_d$) clock cycles advance, that is to say, it writes this stage results back into destination memory from one port and reads next stage operands from the other port of destination memory simultaneously in ($M_d + C_d$) clock cycles. It must ensure that the next stage operands have been written into the destination memory, otherwise the FFT operation results will be wrong.

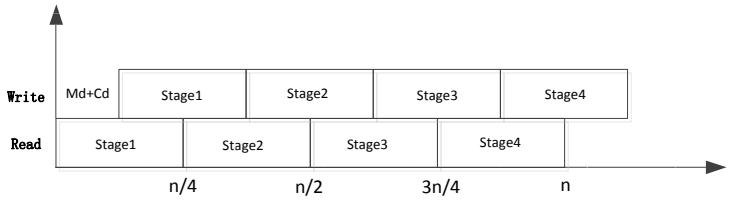

Figure 4. Flowchart of Read-ahead Operation

FFT processor employs Radix-2 DIT-FFT algorithm and fixed addressing structure. This algorithm structure is inputting in reverse order, outputting in sequence order. The topology structure is shown in Figure 4.

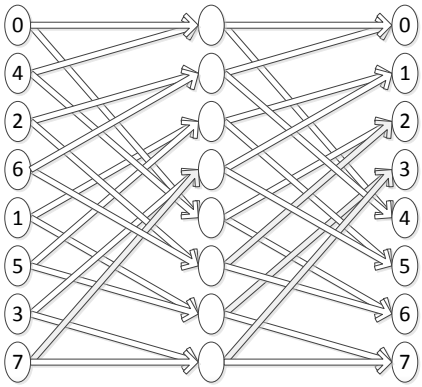

Figure 5. The Topology Structure of FFT Processor

The large memory will be divided into four blocks of smaller memory which are named as Memory x_0/1/2/3 and the storage is N/4 wherein N is the1-D FFT points. Original operation data of 1-D FFT will be stored in four blocks of smaller memory according to certain rules. As to N point FFT/IFFT, the conflict-free address of raw data for storage is shown in the listed table:

Table 1. Conflict-free Address Table

Memory x_0				N/2+3	N/2+7		N-5	N-1
Memory x_1				N/2+2	N/2+6		N-6	N-2
Memory x_2				N/2+1	N/2+5		N-7	N-3
Memory x_3				N/2	N/2+4		N-8	N-4
Address				N/8	N/8+1		N/4-2	N/4-1

In each clock, four operations are read from Memory x_0/1/2/3 respectively. The read-addresses of each operation are same for Memory x_0/1/2/3 and increase in sequence order. Then they will be loaded into two butterfly units and generated four results. And then the four results will be loaded into another four blocks of smaller memory.

According to the topology structure of FFT processor and conflict-free addressing mode, we can find the sequence of reading and storing as listed in the following table:

Table 2. The Sequence of Data Reading Address

Times / Memory	Address of Memory						
					… …	N/4-1	N/4
0-1					… …	N/4-2	N/4-1
2-3					… …	N/4-2	N/4-1

Table 3. The Sequence of Data Storing Address

Times / Memory	Address of Memory						
	1	2	3	4	… …	N /4-1	N /4
0-1	0	N /8	1	N/ 8+1	… …	N /8-1	N /4-1
2-3	N /8	0	N/ 8+1	1	… …	N /4-1	N /8-1

According to the principle of data reading and storing, it cannot make sure that the read-ahead operands of next stage have been already written into memory unless read-ahead times are less than the half of needed reading times in one stage when all results of this stage are completely written into destination memory. It must satisfy the formula:

$$M_d+C_d< n/8 \quad (7)$$

As to 2-D FFT processor, it must satisfy the formula:

$$M_d + C_d < min\{L, C\}/8 \quad (8)$$

In this design, $M_d=3$; $C_d=12$.Then, $\min\{L,C\}>120$ can be derived from (8).As above mentioned, if the point is less than 16K, it does not satisfy (8).It cannot employ the read-ahead operation to hide the pipeline delay of butterfly unit, which will cause a significant increase of operation speed.

4. One-dimensional and Two-dimensional Mixed Architecture FFT Processor

In this section, we will give the detail design for one-dimensional and two dimensional mixed architecture FFT processor which employs the Radix-2 DIT-FFT algorithm and fixed addressing structure. It takes two ways to hide time consumed on data-path, one is read-ahead operation of dual-port RAM to hide the delay introduced by 12-stages pipeline of butterfly-unit and Memory access delay; the other is Ping-Pong operation of 3 groups of RAM to conceal data transmitting time. It also adopt twiddle factor compression algorithm to reduce ROM space. The key components of FFT processor are controller, memory subsystem, data switch network (MUX、 Mem_switch_I and Mem_switch_O), transmitting module and computing module (FFT module). There are a complex memory access system to improve FFT efficiency, including transmitter, internal/external memory controlling unit, address generator, two groups data memory one of which consists four 2K memory labeled as Memory 0/1_x and three groups 4*32 temporary memory which are named Memory 2/3/4_x. Considering 16K=128*128, the maximum point of 1-D FFT is 128. In order to match computing bandwidth, the FFT processor uses three groups of 4 * 32 dual-port memory organized as Ping-Pong operation to fulfill the 1-D FFT. The mixed architecture FFT processor can calculate arbitrary 2^n point FFT directly, where values of n ranges from 5 to 14. According to the difference of FFT points, mixed architecture FFT processor can be configured 1-D FFT mode or 2-D mode. If the number of points is less 16K, the processor would carry on FFT/IFFT with 1-Dmode. Besides, the FFT processor is assigned 2-D FFT mode.

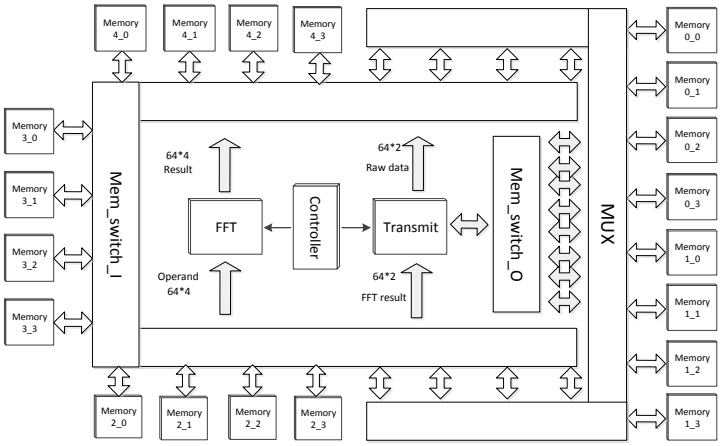

Figure 6. Mixed Architecture of 1-D and 2-D FFT

4.1. One-dimensional FFT operation mode

Given the FFT point is not more than 8K, FFT processor complete it by 1-D mode. Meanwhile, transmission module and temporary memory are passed by. FFT module accesses two groups data memory (Memory 0_x and Memory 1_x) at the same time; so as to it can read four operands from one and write four results into the other. But the next cascade, the source memory and destination memory would be shifted. The calculation flow is shown in Figure 7.

Figure 7. Flowchart of 1-D FFT Operation Mode

4.2. Two-dimensional FFT operation mode

If the scale of FFT would be 16K, the FFT processor configured 2-D mode. At present, the transmitter has two main jobs, one is to fetch raw data of row or column from Memory 0_x or Memory 1_x and writes them into one of three temporary memory and the other one is to send the FFT results from the temporary memory to data memory. In this mode, the FFT Controller can appoint one of temporary memory as source end and assign another as destination end. The FFT module can read four operands from source end and writes four results into destination end simultaneously. Under the management of Controller module, the transmission module and the FFT operation unit complete all row/column FFT process coordinately. Finally, it outputs the result of 16K points FFT/IFFT.

As shown in Figure 8, there three temporary memories are organized as Ping-Pong architecture. If Memory 2_x are used as operand memory, the Memory 3_x are assigned result memory. They are in turn to perform the other side unless current operand memory is empty. Now, Memory 4_x are served as data buffer in which the data are exchanged between FFT module and data memory. Three temporary memory are in turn to do three functions if the current operands would have been completed all operation. Repeating above operations, the mixed architecture FFT processor can complete the long point FFT/IFFT with several groups of small size memory.

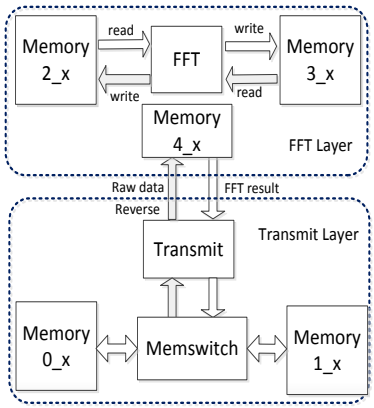

Figure 8. Flowchart of 2-D FFT Operation Mode

5. Performance comparison of the three FFT processors

Traditional one-dimensional FFT processor has idle processing speed, but it costs more storage resources. Because of additional time consumption of transmission and limitations of read-ahead, two-dimensional FFT processor operates slowly, but its storage resources can be greatly compressed. The mixed architecture FFT processor has excellent speed and far less storage resource. The details are compared as follow.

Here comes Table 4 which contains all the FFT computation cycles. If the number of points is less than 16K, the 2-D processor can't read any data from the current destination memory unless all result had been produced. For this constrain, the 2-D processor increases significantly the whole computation cycles compared with the theoretical value (T-V). Because of the same constrain, mixed architecture dealing with 32 or 64 point FFT can't hide butterfly unit computing delay and memory access delay by read-ahead operation. Therefore, the computation cycles of 32 and 64-point FFT are more than the theoretical value.

Table 4. Comparison of FFT Computation Cycles

Speed / FFT	FFT points									
	32	64	128	256	512	1k	2k	4k	8k	16k
T-V	40	96	224	512	1152	2560	5632	12288	26624	57344
1-D	120	192	240	528	1168	2576	5648	12304	26640	57574
2-D	180	284	396	808	1708	3628	13524	24760	39160	57656
Mix-D	120	192	240	528	1168	2576	5648	12304	26640	57656

It is easy to draw FFT relative acceleration ratio linear curve which is the ratio of theoretical value with FFT computation cycles. As shown in Figure 9, we can see that 1-D FFT processor and mixed architecture FFT processor have idle processing speed, and their relative acceleration ratiolinear curves coincide with each other. Their relative acceleration ratiocoefficientsare close to 1 when they deal with N-point (N is from range of 128 to 16K)

FFT. In 2-D FFT processor, the maximum point of 1-D FFT is 1K. If the number of point is less 2K,the input point of N can be described as matrix form by 1-row and N-column. Besides, it would be configured matrix form by L-row and C-column. As a result, relative acceleration ratio linear curve of 2-D FFT processor has a turning point appeared in point 2K. 2-D FFT processor operates slowly, but it's operating speed close to the theoretical speed of operation when it operates a long point FFT. For example, three linear curves intersect at point 16K.

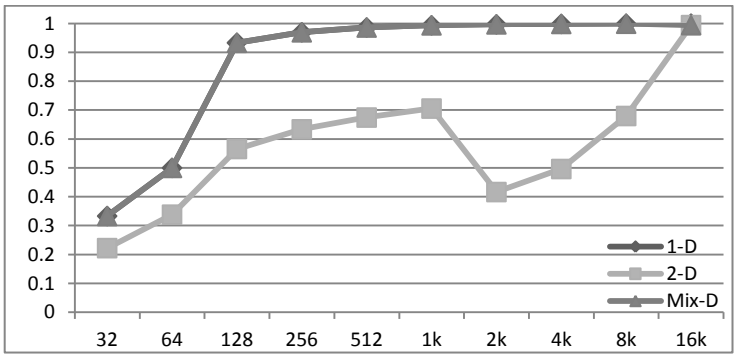

Figure 9. FFT Relative Acceleration Ratio Linear Curve

Here comes Table5which contains all the comprehensive resource and the maximum frequency. Due to data dependency, 1-D FFT processor has to have double capacity of max points.

Table 5. Comparison of Comprehensive Resource and the Maximum Frequency

FFT / Index	1-D	2-D	mix-D
LUTs	11154	15659	19364
DSP48E1s	64	88	88
Block RAM	68	35	44
MaxFrequency	198.169	182.249	186.898

It is possible to draw FFT relative acceleration ratio of unit storage resource linear curve by dividing the corresponding value of Block RAM and normalizing the results. As shown in Figure 10, we can see that mixed architecture FFT processor have idle relative acceleration ratio of unit storage resource compared with 1-D FFT processor and 2-D FFT processor. It cannot be ignored that 2-D FFT relative acceleration ratio coefficient of unit storage resource of 2-D FFT processor is larger than the one of mixed architecture FFT processor. The mixed architecture FFT processor considered as a whole has higher performance resource ratio when carrying out 2^n (n is from range of 5 to 14) single-precision floating-point FFT/IFFT.

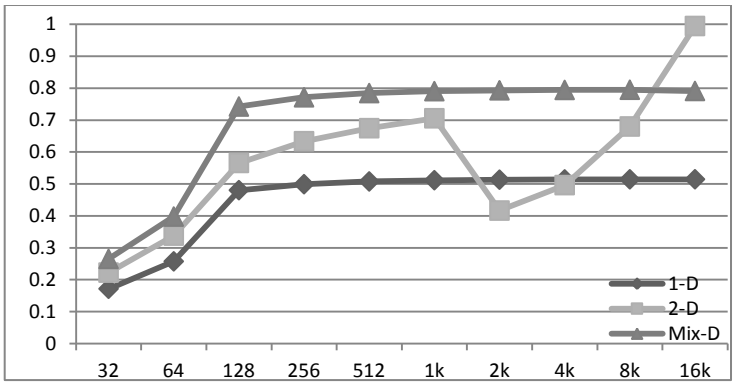

Figure 10. FFT Relative Acceleration Ratioof Unit Storage Resource Linear Curve

The above mentioned FFT processors have all been successfully implemented on Virtex-6 XC6VLX760 FPGA. We can see that achieves balance between high processing speed and resources.

6. Summary

Based on 1-D FFT algorithm and 2-D FFT algorithm, this paper introduces mixed architecture which can carry on 1-D FFT algorithm or 2-D FFT algorithm corresponding different size of FFT. The hardware implementation of the above-mentioned FFT processors is discussed independently. Finally the operation speed and the comprehensive resource between the above-mentioned FFT processors are compared. The description and the evaluation results depicted in this paper clarify the advantage of the proposed FFT architecture that it achieves balance between high processing speed and resources. It is worth notice that the mixed architecture FFT processor has been successfully implemented on XilinxVirtex-6 XC6VLX760 FPGA. Furthermore it is implemented on the heterogeneous multi-core SoC platform.

Acknowledgments

This research was sponsored by the National Natural Science Foundation of China under Contracts61106020, 61002400, 61179036, and Doctoral Fund of Ministry of Education of China (Grant No. 20100111120009).

References

[1] W. Xiaojun, L. Teng and Z. Xiyuan, "Research on Implementation of 2-dimension Long FFT Processor Based on Cascade-pipelined Structure", Signal and Information Processing, vol. 11, no. 40, **(2010)**.
[2] H. Weidong, D. Zhemin and G. Cheng, "2D-parallel method for ultra long FFTs in FPGA", Electronic Measurement Technology, vol. 11, no. 30, **(2010)**.
[3] S. Tao and D. -J. Zhuang, "Improvement and Implementation of FFT Algorithm for Long Sequences", Modern Radar, vol. 7, no. 27, **(2005)**.
[4] O. Nibouche, S. Boussakta, M. Darnell and M. Benaissa, "Algorithms and pipeline architectures for 2-D FFT and FFT-like transforms", Digital Signal Processing, vol. 4, no. 20, **(2010)**.
[5] S.-I. Satake, Y. Hiroi, Y. Suzuki, N. Masuda and T. Ito, "Special-purpose computer for two-dimensional FFT", Computer Physics Communications, vol. 6, no. 179, **(2008)**.
[6] S. Bouguezel, M. O. Ahmad and M. N. S. Swamy, "New radix-(2×2×2)/(4×4×4) and radix-(2×2×2)/(8×8×8) DIF FFT algorithms for 3-D DFT", IEEE Transactions on Circuits and Systems, vol. 2, no. 53, **(2006)**.

[7] A. V. Oppenheim, R. W. Schafer and J. R. Buck, "Discrete Time Signal Processing", 2nd ed, Prentice Hall Publishers, Upper Saddle River, New Jersey, (**1998**).
[8] C. Peiqing, "Digital Signal Processing Tutorial3rded, Tsinghua University Publishers", Beijing, **(2006)**, pp.143-175.
[9] A. Haveliya, "Design and Simulation of 32-Point FFT Using Radix-2 Algorithm for FPGA Implementation", 2012 Second International Conference on Advanced Computing & Communication Technologies, Rohtak, Haryana, **(2012)** January 7-8.
[10] H. He and H. Guo, "The Realization of FFT Algorithm based on FPGA Co-processor", Second International Symposium on Intelligent Information Technology Application, Shanghai, **(2008)** December 20-22.
[11] B. Wang, Q. Zhang, T. Ao and M. Huang, "Design of Pipelined FFT Processor Based on FPGA", 2010 Second International Conference on Computer Modeling and Simulation, Sanya, Hainana, **(2010)** January 22-24.
[12] R. Airoldi, F. Garzia and J. Nurmi, "FFT Algorithms Evaluation on a Homogeneous Multi-Processor System-on-Chip", 2010 39th International Conference on Parallel Processing Workshops, San Diego, CA, **(2010)** September 13-16.
[13] D. -L. Zhang, X. -P. Yang and Y. -K Song, "Design and implementation of a large points FFT acceleration unit in multi-processor system based on FPGA", Proceedings of the 2nd International Symposium on Computer, Communication, Control and Automation (ISCCCA-13), Published by Atlantis Press, Paris, France, Shenyang, Lianing, China, **(2013)** May 25-26.
[14] X. -P. Li, C. Long and W. Shihu, "The Implementation of High-speed FFT processor based on FPGA", 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE), Changchun, **(2010)** August 24-26.
[15] S. Mou and X. Yang, "Design of a High-speed FPGA-based 32-bit Floating-point FFT Processor", Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, Qingdao, **(2007)** July30-August 1.
[16] S. K. Shome, A. Ahesh, D. K. Gupta and S. R. K. Vadali, "Architectural design of a highly programmable Radix-2 FFT processor with efficient addressing logic", 2012 International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, **(2012)** March 15-16.

