

# Novel FPGA-based pipelined floating point FFT processor

Li Wei<sup>a)</sup> and Wang Jun

School of Electronics and Information Engineering, Beijing University of Aeronautics and Astronautics, Beijing 100083, China

a) [windriver@126.com](mailto:windriver@126.com)

**Abstract:** Two novel architectures for pipelined floating point fast Fourier transform on FPGA are presented. The new radix-2<sup>2</sup> two-path delay feedback (R2<sup>2</sup>TDF) architecture leads to 50% area saving for floating point complex adders compared with the radix-2<sup>2</sup> single-path delay feedback (R2<sup>2</sup>SDF) architecture. Besides a new hybrid architecture is presented which mixes the R2<sup>2</sup>TDF and R2<sup>2</sup>TDF butterfly structures and is flexible and efficient for FPGA implementation.

**Keywords:** FFT, pipelined, floating point, FPGA

**Classification:** Science and engineering for electronics

## References

- [1] A. V. Oppenheim and R. W. Schaffer, “Discrete-Time Signal Processing, 2nd ed.,” New Jersey, Prentice-Hall, 1999.
- [2] L. R. Rabinar and B. Gold, “Theory and Application of Digital Signal Processing,” PHI, 1999.
- [3] S. He and M. Torkelson, “A new approach to pipeline FFT processor,” *Proc. IEEE 10th Int. Parallel Process. Symp.*, pp. 766–770, 1996.
- [4] E. E. Swartzlander, W. K. W. Young, and S. J. Joseph, “A radix-4 delay commutator for fast Fourier transform processor implementation,” *IEEE J. Solid-State Circuits*, vol. 19, no. 5, pp. 702–709, Oct. 1984.
- [5] Y. N. Chang and K. K. Parhi, “An efficient pipelined FFT architecture,” *IEEE Trans. Circuits Syst. II*, vol. 50, no. 6, pp. 322–325, June 2003.
- [6] S. K. Nag and H. K. Verma, “An efficient parallel design of FFTs in FPGAs,” *Proc. International Conference on Signal processing applications and technology (ICSPAT 98)*, 1998.
- [7] L. Mintzer, “Large FFTs in a single FPGA,” *Proc. International Conference on Signal processing applications and technology (ICSPAT 96)*, pp. 895–899, 1996.
- [8] Xilinx Incorporated, 2008. [Online] <http://www.xilinx.com>

## 1 Introduction

Fast Fourier Transform (FFT) is one of the most important algorithms in digital signal processing and is necessary for many real-time requirements [1]. Since the complexity of FFT implementation grows with the FFT depth and precision different kinds of processor architectures have been developed to increase processing speed and decrease hardware cost. There are mainly two

common FFT architectures in use [2, 3, 4, 5]. The first is the memory-based architecture. It accepts a burst of data for a short time, after which the stream must stall until the data is processed. This architecture has one or two butterfly units. It is considered area efficient however it requires many computation cycles. The other is the pipeline architecture. It consists of a pipeline capable of processing a stream of data at a constant rate of throughput. Data sample can be accepted every clock cycle. The pipelined architecture gives high throughput, since the data stream is never stalled. Thus, the maximum performance will be limited by the maximum achievable clock frequency. This architecture requires number of processing elements related with the length of FFT and the radix which consumes relatively large area compared with memory architectures

The data precision format is a key factor for FFT processor. There are two types of data format. The first is the fixed-point format which is used in most FFT processors. The problem with a fixed point FFT is to maintain the accuracy and preserve the dynamic range at the same time. To meet such requirement, studies have been done on floating-point FFT processors.

In the past twenty years, Field Programmable Gate Array (FPGA) and Programmable Logic Device' (PLD) have developed rapidly and at current stage digital signal processors based on FPGA are applied in most areas of signal processing. Compared with traditional ASIC design flow, design based on FPGA has the advantages of flexibility and time to market objective. A lot of pipelined FFT processors based on FPGA have been presented by researchers. However most of them concentrate on fix-point data format. The move from fixed point data to floating-point significantly changes the design space. Floating-point arithmetic requires much more area per operation and, more importantly, it requires almost as much area for an adder as a multiplier.

In this paper we develop a pipelined floating point FFT processor. First the architecture of the pipelined radix-2<sup>2</sup> single-path delay feedback (R2<sup>2</sup>SDF) FFT architecture is studied. Then a new radix-2<sup>2</sup> two-path delay feedback (R2<sup>2</sup>TDF) FFT architecture is designed and implemented. Compared with R2<sup>2</sup>SDF architecture this processor has a low cost of addition operators however a higher memory cost. So we present another hybrid architecture to balance the cost for both resources of FPGA.

## 2 R2<sup>2</sup>SDF architecture

The pipelined FFT architecture typically falls into one of the two following categories [2, 3, 4, 5]. One is multipath delay commutator (MDC) and the other is single-path delay feedback (SDF), respectively. In general, the MDC schemes can achieve a higher throughput rate while need more hardware cost and additional memory to reorder the input data. In this paper the R2<sup>2</sup>SDF architecture [3] is selected because it has less requirement for multiplier and storage while has a high throughput. This makes it an ideal architecture for FPGA implementation of pipeline FFT processors.

Consider the input of FFT  $x(n)$ ,  $n = 0 \sim N - 1$ , where  $N$  being size

of the FFT. According to the R2<sup>2</sup>SDF architecture the first stage butterfly *BFI* first shifts the incoming data  $x(n)$  into shift register,  $k$  clock cycles later computes a two point DFT with the incoming data and the data stored in the shift register as eq. (1).

$$\begin{aligned} & (xr(n) + xr(n+k)) + j(xi(n) + xi(n+k)), \\ & (xr(n) - xr(n+k)) + j(xi(n) - xi(n+k)) \end{aligned} \quad (1)$$

Where  $xr(n)$  and  $xi(n)$  is the real and imaginary part of  $x(n)$ .

Similarly the second stage butterfly *BFII* first shifts the incoming data  $x(n)$  into shift register,  $k$  clock cycles later computes a two point DFT with the incoming data and the data stored in the shift register.

$$\begin{aligned} & (xr(n) + xi(n+k)) + j(xi(n) - xr(n+k)), \\ & (xr(n) - xi(n+k)) + j(xi(n) + xr(n+k)) \end{aligned} \quad (2)$$

According to (1) and (2) the butterfly is consisted of four real adders and which is free when waiting for the incoming data. The utilization rate of the adder operation is 50%. To implement the radix-2<sup>2</sup> single-path delay feedback architecture, we need  $\log_4 N - 1$  complex multipliers,  $2 \log_2 N$  complex adders and  $N - 1$  shift registers.

### 3 R2<sup>2</sup>TDF architecture

The butterfly structure of the new FFT processor is shown in Fig. 1. Consider the first butterfly of the FFT where the butterfly input being  $x(n)$  and  $x(n + N/2)$   $n = 0 \sim N/2 - 1$ . On the first  $N/2$  cycles  $x(n)$  is stored in RAM A with the control signal  $s = 0$ . On the next  $N/2$  cycles the butterfly computes  $Z_1(n) = x_A + x_B$  with  $s = 1$  where  $x_A$  being the data loaded from the RAM A and  $x_B$  being the current input data. The result is sent to the next butterfly. At the same time  $x_B$  is stored in RAM B.  $N/2$  clocks later  $x_A$  and  $x_B$  are loaded from RAMA and RAMB respectively and the butterfly computes  $Z_2(n) = x_A - x_B$ . At this time the current input data is stored in RAMA. The operation of the second butterfly is similar to that of the first one, except the “distance” of butterfly input sequence is just  $N/4$ .

The structure presented needs two memories for feedback storage and the total memory cost is twice than the R2<sup>2</sup>SDF so we name the structure R2<sup>2</sup>TDF (Two Data Feedback). It can be seen there is only one complex computing unit with  $Z_2(n)$  or  $Z_1(n)$ . Consider the stages of the FFT processor  $k = 0 \sim K - 1$ , when  $k$  is even the  $Z_1(n)$  is consisted with one real addition and one real subtraction operator and  $Z_2(n)$  is consisted with two real subtraction operating. When  $k$  is odd the  $Z_1(n)$  and  $Z_2(n)$  both need one real addition and one real subtraction operating. From the IEEE-754 floating point structure the positive and negative number is only different in the flag. So the subtraction can be easily performed by addition operator with the flag change. So it can be seen that the real floating point adder can be reduced 50% compared with the R2<sup>2</sup>SDF.

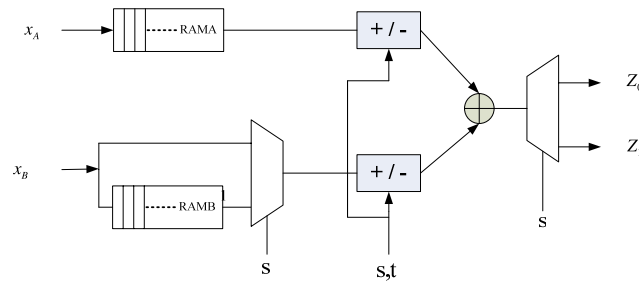


Fig. 1. Butterfly structure for  $R2^2TDF$  FFT processor

#### 4 Hybrid architecture

The floating point FFT processor presented is optimized with the cost of complex adders however the memory use is twice than the  $R2^2SDF$  architecture. A solution is to combine these two architectures and optimize the two kinds of resource at the same time. From the above description the data flow of the two architectures is the same so as the input and output relationship. Hence the hybrid architecture can be used with some stages using  $R2^2SDF$  and some using the new architecture  $R2^2TDF$ . It is known that the memory cost of the  $R2^2SDF$  structure of the DIF algorithm is decreased with the stages. So the memory is mainly cost in the first several stages which can be implemented using the  $R2^2SDF$  stage structure and the reset stages can be implemented using the  $R2^2TDF$  structure.

The hybrid FFT architecture for  $N = 1024$  is shown in Fig. 2 and the hardware requirement comparison is listed in Table. I. It shows that compared with the  $R2^2SDF$  the hybrid architecture needs about half of complex adders with only slightly more data memory.

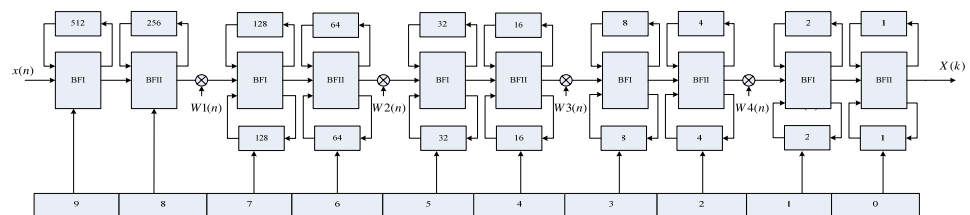


Fig. 2.  $R2^2SDF$  pipeline FFT architecture for  $N = 1024$

Table I. Hardware requirement comparison

	$R2^2SDF$	$R2^2TDF$	Hybrid
Memory	$N - 1$	$2N - 2$	$1.25N - 2$
Complex Adders	$4\log_4 N$	$2\log_4 N$	$2\log_4 N + 2$

## 5 Implementation

The floating point architecture of FFT processor is similar with the fixed point. The addition and multiplication operators and coefficient are changed using floating point format. The floating point arithmetics have been widely studied and some commercial vendors have developed floating-point intellectual property (IP) cores on FPGAs. In this paper the floating point operator is implemented using the single precision floating point IP of Xilinx which allows the operands to be applied on every clock cycles [8]. In this paper a 16384 point FFT processor is implemented using the Xilinx FPGA XC5VSX95 with ISE 10.1. The resource cost is listed in Table. II from synthesis tools with the default settings and the running clock speed is about 160 MHz. In this table the slice flip flops and LUTs are mainly used to implement the floating point adders. The BlockRAMs are used by the data memory and DSP48Es are used by the multipliers. It shows the resource cost is consistent with the hardware requirement comparison listed in Table. I.

**Table II.** FPGA cost of the FFT processor

	R2 <sup>2</sup> SDF	R2 <sup>2</sup> TDF	Hybrid
Slice Flip Flops	43387	29602	35536
Slice LUTs	39389	27597	32821
36k BlockRAMs	41	79	63
DSP48Es	72	72	72

## 6 Conclusion

In this paper two new floating point pipelined FFT processors are proposed. The R2<sup>2</sup>TDF architecture optimizes the R2<sup>2</sup>SDF butterfly structure to increase the floating point adder utilization and the hybrid architecture use two kinds of butterfly structures which can balance the cost for the memory use and adders. This makes it an ideal architecture for FPGA implementation of pipeline floating point FFT processors.