An Implementation of Pipelined Radix-4 FFT Architecture on FPGAs

Jiang Wang and Leif Arne Ronningen

***Abstract*—Design and functional implementation of a 16-point pipelined FFT architecture is presented. The architecture is based on the radix-4 algorithm. By exploiting the regularity of the algorithm, butterfly operation and multiplier modules were designed. The architecture adopts four butterflies, and the pipeline stage is optimized to balance the processing speed and the area. It was modeled by VHDL, and synthesized in FPGA. By adopting this architecture, the data throughput could be 2M/s. It is extensible for high point FFT.**

***Index Terms*—Fast Fourier transform (FFT), modular architecture, pipeline, VLSI design.**

I. Introduction

The fast Fourier transform (FFT) class of algorithms [1] is widely used in communication and digital signal processing. The FFT algorithm is considered one of the basic algorithms in many DSP projects. Nowadays, FFT is the key building block for the mobile communications; especially for the orthogonal frequency division multiplexing (OFDM) transceiver systems [2].Implementation of FFT of different architectures, for fast and efficient computational schemes, has attracted many researchers. The methodology of FFT simulation, implementation, and verification plays a key role in the industrial or consumer electronics areas, for example, the FFT image or acoustic processing, encoding and decoding, harmonic analysis in renewable energy and so on. The FFT is a typical computation where the memory access intensively and the high parallelism is needed. VLSI realization of FFT algorithm, should have pipelined architecture and/or parallelism, be regular and modular [3]. At algorithm level, it should achieve the multiplicative complexity as low as possible. At the architecture level, use the delay-feedback buffering strategy to minimize the memory size. It should have modular and regular modules, local routing, and low control complexity.

The discrete Fourier transform (DFT) X(k) of an N-point sequence x(n) is defined by

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{nk} \qquad \text{k=0,1,…,N-1} \tag{1}$$

$$W_N^{nk}=\cos\left(\frac{2\pi\cdot nk}{N}\right)-j\cdot\sin\left(\frac{2\pi\cdot nk}{N}\right) \tag{2}$$

In (2), the W_N^n is usually referred to as twiddle factor.

Manuscript received April 10, 2013; revised June 14, 2013.

The authors are with the Department of Telematics, NTNU, Trondheim 7491, Norway (e-mail: jiangw@item.ntnu.no).

Selecting an FFT radix is the first step on the algorithmic level. It is mainly a trade-off between the speed, power and area for the number of transistors. High-radix FFT algorithms, such as radix-8, often increase the control complexity and are not easy to implement. And to radix-2 FFT, there is the increase in the number of butterfly elements compared with radix-4. So the radix-4 was selected in this paper.

To understand the radix-4 FFT algorithm intuitively, we can examine the signal flow graph (SFG) as shown in Fig. 1.

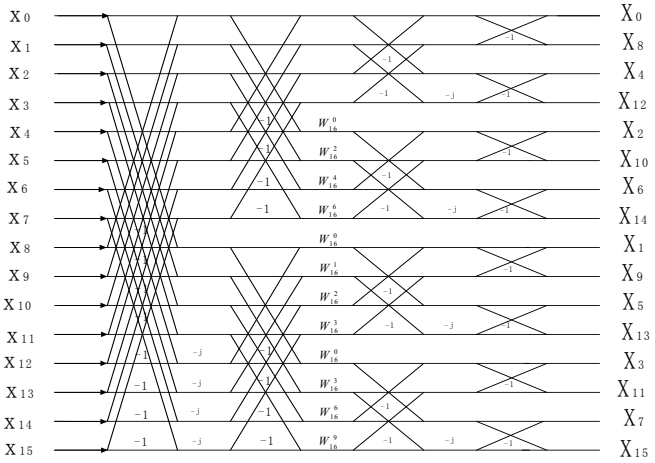

Fig. 1. Signal flow graph of 16-point radix-4 FFT

We could learn from Fig. 1. that –j terms are extracted. The complex multiplication with –j are accomplished by exchanging the real and the imaginary parts of the incoming data and then inverting the sign of the imaginary part. The SFG is also the base for designing a butterfly operation. It could be mapped to the architecture shown in Fig. 2.

The fundamental principle of Cooley and Turkey's algorithm for computation of N-point DFT, is that to decompose a given Discrete Fourier Transform (DFT) problem into successively smaller DFTs. The algorithm for the High Speed Pipelined DIT FFT architecture is based on the following equations:

$$\left[X(k),X\left(k+\frac{N}{4}\right),X\left(k+\frac{N}{2}\right),X\left(k+\frac{3N}{4}\right)\right]^T=\begin{bmatrix}1&1&1&1\\1&-j&-1&j\\1&-1&1&-1\\1&j&-1&-j\end{bmatrix}\begin{bmatrix}W_N^0F_0(k)\\W_N^kF_1(k)\\W_N^{2k}F_2(k)\\W_N^{3k}F_3(k)\end{bmatrix} \tag{3}$$

here

DOI: 10.7763/JOCET.2014.V2.100

$$F_{n1}(k)=\sum_{n_2=0}^{(N/4)-1} x(n_1+4n_2)W_{N/4}^{n_2\cdot k}$$

for n =0, 1, 2, 3; k=0,1,…, $\frac{N}{4}$ -1.

The $W_N^0, W_N^k, W_N^{2k}, W_N^{3k}$ also could be extracted and listed in Fig. 1. In the hardware implementation, they are stored in the RAM with 16-bit fix point.

II. Radix-4 Pipeline FFT Architecture

A. Delay Feedback Pipelined Architecture

There are two main delay buffering strategies of pipelined FFT architecture [4], [5] in the butterfly stages. One is delay-commutator (DC) architecture, and the other one is delay-feedback (DF) architecture. In above two architectures, FIFO is used to buffering the intermediate data. For the DC architecture, the utilization of each FIFO is 50%. For the DF architecture, the utilization of each FIFO is increased to 100%. The DF strategy is adopted in this implementation as shown in Fig. 2.

There are 4 stages shown in Fig. 2. Every stage has one butterfly element. To the butterfly operation, the input data and out put data could use the same address, the corresponding RAM could be divided into four groups for the four butterflies.

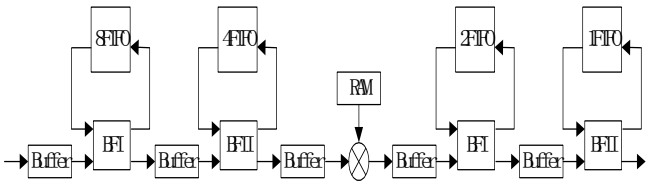

Fig. 2. Radix-4 pipeline architecture.

By observing the Radix-4 SFG, every butterfly stage has only one or two butterfly modes. They are denotes as BFI and BFII here. The BFI mode implements common butterfly operation. The BFII not only includes the common butterfly, but the butterfly whose input data will be multiplied by –j before normal butterfly operation. The overall structure is regular and suitable for VLSI implementation.

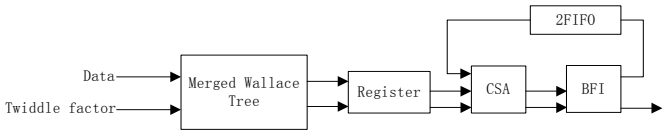

Fig. 3. Repartition scheme for pipeline stage III.

The butterfly operation and the twiddle factor multiplier both select modified-Booth encoding and Wallace tree. To the stage III, the latency of a complex multiplication is usually twice as long as that of a butterfly operation. So the pipeline should be repartitioned to balance the latency. To crack this problem, we merged the final adder (Carry Saved Adder) and the BFI into one stage. It is shown in Fig. 3.

B. Memory Design

There are three different types of memory (RAMs) in the architecture, which store the input values of data and coefficient, and the output value of the result. The RAM is designed to store one full set of data for 16-point FFT computation. Both the real and imaginary parts of the data are stored in fixed point representation as two different numbers. So for a radix-4 butterfly operation we have 4 numbers (2-real, 2-imaginary) to be stored and each number is 16-bit long. The data can be read serially to be processed in the radix computation element [6] [7]. The coefficient RAM and the output RAM are similar to the data RAM. The coefficient RAM that has 7 bytes denotes all of the twiddle factors listed in Fig. 1.

C. Memory Controllers

The RAM controller controls the READ and WRITE operations with proper address generation logic. The address generation unit provides the read pointer, and write pointer. The address generation is done using two modulo-4 counters. The number of unprocessed data items in the RAM, and overflow/underflow memory access, are controlled using an up/down counter. Every stage has a buffer，its function is to prepare the input data for the next stage. This buffering function could also be realized by the memory controllers. The counters produce the latency and signal the address generator to prepare the data.

III. Functional Implementation

The architecture proposed in the above section has been modeled in hardware description language VHDL with generic parameters for transform length and word-length, using fixed 16 point architecture. The presented architecture is regular and extensive for high point, 32-bit FFT which is used in 3D image processing systems.

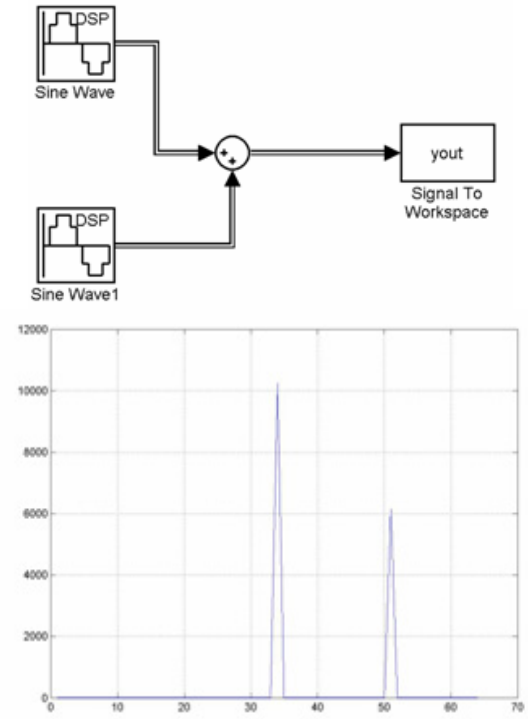

Fig. 4. Test vector generation for source signal with two frequency components.

The design methodology is from floating point model to fixed point model. The Matlab behavior model is established as a function model. Its PSNR is analyzed for specific applications. Then the hardware C model is established as a golden for VHDL verification. Fig. 4 is the test vector generation. Fig. 5 is the RTL simulation of the FFT core. It could be seen that the two frequency components are detected accurately by the FFT core.

The design flow adopts both top-to-bottom and bottom-to-top. In the top-to-bottom, the top level is constructed with high priority, such as control modules, data path and memory management. In the bottom-to-top, the fundamental computation elements are established for the more flexible modules construction, such as the adder and multiplier combined to establish the complex multiplier.

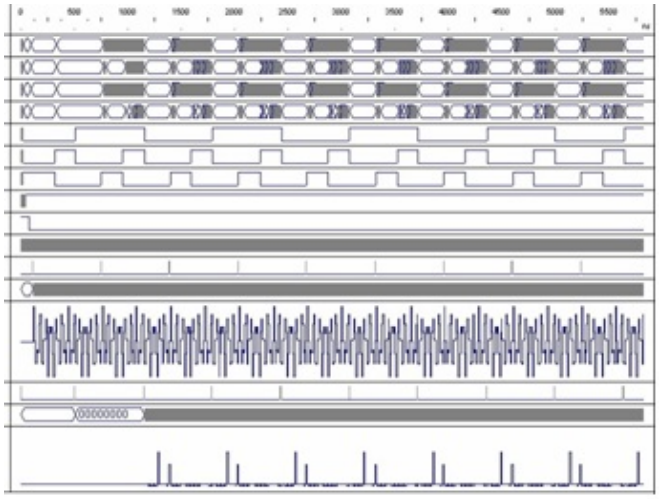

Fig. 5. Simulation of frequency detection by the FFT core.

The prototype of the presented FFT architecture has been fully synthesized by means of Altera FLEX10K (EPF10K130EQC240-2) [8]. Experimental results show throughput could be 2M/s.

The area/power consumption in the pipelined architecture is dominated by the FIFO register files and the complex multiplier. This is also considered as our future research direction.

IV. Conclusion

An architecture for pipelined processing 16-point FFT has been presented, which is regular and extensible for high point FFT. The pipeline performance was enhanced by the repartition of the multiplier and butterfly operation. The prototype of the architecture has been synthesized and verified by FPGA.

References

[1] A. V. Oppenheim and R. W. Schafer, *Digital signal processing*, NJ, Prentice-Hall, 1975. pp. 297-310.

[2] A. Sadat and W. B. Mikhael, "Fast Fourier transform for high speed OFDM wireless multimedia system, circuits and systems," in *Proc. the 44th IEEE 2001 Midwest Symposium on MWSCAS*, 2001, pp. 938.

[3] K. Sapiecha and R. Jarocki, "Modular architecture for high performance implementation of the FFT algorithm," *IEEE Trans. On Computers,* vol. 39, pp. 1464 – 1468, Dec. 1990.

[4] S. S. He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Proc. IEEE 1998 custom intergrated circuits conference,* 1998, pp. 131-134.

[5] B. Kang and J. Kim, "Low complexity multi-point 4-channel FFT processor for IEEE 802.11n MIMO-OFDM WLAN system," in *Proc. Green and Ubiquitous Technology 2012 international conference,* 2012, pp. 94-97.

[6] G. Bi and E. Jones, "A pipelined FFT processor for word-sequential data," *IEEE Trans. On Acoustics, Speech, and Signal Processing,* vol. 37, pp. 1982-1985, Dec. 1989.

[7] P. F. Stelling, C. U. Martel, V. G. Oklobdzija, and R. Ravi, "Optimal circuits for parallel multipliers," *IEEE Trans. Comput.,* vol. 47, pp. 273-285, Mar. 1998.

[8] Altera FPGA devices, Altera Corporation. [Online]. Available: http://www.altera.com/devices/fpga/fpga-index.html

Jiang Wang is a research fellow in Department of Telematics, NTNU, Norway. His main research focuses on the digital signal processing in industrial or consumer electronics, including multimedia, smart grid, renewable energy, signal processing platform and so on.

Leif Arne Ronningen is a professor in Department of Telematics, NTNU, Norway. His research interests includes multimedia technology, networked Collaboration Spaces, near-natural virtual quality, distributed heterogeneous processing architectures for real-time embedded systems, performance evaluation of quality shaping, as supported by DMP networks, the DMP Architecture, specification and philosophy, and application of DMP in arts.