VLSI Implementation of High Speed and High-Resolution FFT Algorithm Based on Radix 2 for DSP Application
Nooshin Mahdavi, Rozita Teymourzadeh, Masuri Bin Othman

To cite this version:
Nooshin Mahdavi, Rozita Teymourzadeh, Masuri Bin Othman. VLSI Implementation of High Speed and High-Resolution FFT Algorithm Based on Radix 2 for DSP Application. IEEE Student Conference on Research and Development (SCoReD 2007), Dec 2007, Selangor, Malaysia. IEEE, 2007 5th Student Conference on Research and Development, 2008, <10.1109/SCORED.2007.4451381>. <hal-01800728>

HAL Id: hal-01800728
https://hal.archives-ouvertes.fr/hal-01800728
Submitted on 17 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
VLSI Implementation of High Speed and High Resolution FFT Algorithm Based on Radix 2 for DSP Application

N. Mahdavi, R. Teymourzadeh, IEEE Student Member, Masuri Bin Othman

Abstract—Using Fast Fourier Transform (FFT) is indispensable in most signal processing applications. Designing an appropriate algorithm for the implementation of FFT can be efficacious in digital signal processing. Sophisticated techniques such as pipelining and parallel calculations have potential impacts on VLSI implementation of FFT algorithm. Furthermore, a mathemathic approach such as floating point calculation achieves higher precision. In this paper, an efficient algorithm with using parallel and pipelining methods is proposed to implement high speed and high resolution FFT algorithm. Latency reduction is an important issue to implement the high speed FFT on FPGA. The Proposed FFT algorithm shows the latency of 5131 clock pulse when N refers to 1024 points. The design has the mean squared error (MSE) of 0.0001 which is preferable to Radix 2 FFT.

Index Terms—Radix, FFT, Butterfly, VLSI, Floating point.

I. INTRODUCTION

Fast Fourier transform is a faster version of DFT which was invented by Cooley & Tuky in 1965 [1]. There are several types of FFT algorithms. Radix-2 is a practical algorithm to calculate FFT by using decimation in time (DIT) method.

The main idea of decimation in time is to sunder the input string of \( h_n \) with length of \( N = 2^L \) into two substrings with odd and even indexes. The N-point DFT of \( h_n \) can be achieved by suitable combining to \( N/2 \)-point DFT of each substring. Each one of them can be reduced to \( N/4 \)-point DFTs. This procedure will continue till only 2-point DFTs remains [2]. The final result is the formation of FFT algorithm. In 2006 petrovsky [3] implemented Radix 2-4 based on parallel-pipeline FFT-processors at structural level. He achieved 0.00201 MSE on his FFT algorithm. This paper shows the implementation of high speed and high resolution FFT based on parallel pipeline and floating point Radix-2 with the minimum MSE of 0.0001.

The next section describes the principle of the FFT structure. The mathematical formulation and block diagram of pipeline FFT algorithm is described in section III. Section IV shows implementation and design result in brief. Finally, conclusion is expressed in section V.

II. PRINCIPLE OF THE FFT STRUCTURE

The computation of N-point FFT via decimation-in-time algorithm requires \( O(N \log_2 N) \) complex multiplication time [7] and it will be increased if basic butterfly computation is used for adder and multiplication in decimation-in-time FFT algorithm. Additionally a complicated controller is also required. Furthermore, implementing fixed point FFT algorithm, results the output differs significantly in comparison to the expected output.

To obtain the high speed and high resolution FFT algorithm, implementation of the floating point pipeline FFT is applied.

The proposed method shows high performance of the FFT algorithm. The principle architecture is based on using a memory to keep input and output data. Thus, it reduces hard ware complexity. All the blocks are designed to operate with the same clock frequency. These stages implement as pipeline and parallel to reduce FFT calculations time.

III. FFT BLOCK DIAGRAM DESCRIPTION

Figure 1 shows the main block diagram of the FFT algorithm. Each block will be investigated separately.

A. Radix-2 Butterfly Algorithm

First, the RAM is initialized by external microprocessor and the data are loaded to the RAM by bit reserve address pin. The Radix 2 Butterfly block is the primary prefatory step taken in DIT FFT [8,9]. This block calculates the complex number in equation 1 and 2 as follow.

\[
\text{output} 1 = \text{input} 1 + W^k \times \text{input} 2 \quad (1)
\]
\[
\text{output} 2 = \text{input} 1 - W^k \times \text{input} 2 \quad (2)
\]
Hence, one complex multiplication and two complex additions are required in this calculation. If \( \text{output1} = X_{o1} + iY_{o1} \) , \( \text{output2} = X_{o2} + iY_{o2} \) and \( W^k = X_{w^k} + iY_{w^k} \) we have:

\[
X_{o1} + iY_{o1} = \left( X_{i1} + \left( X_{w^k} \times X_{i2} - Y_{w^k} \times Y_{i2} \right) \right)
+ \left( Y_{i1} + \left( X_{w^k} \times Y_{i2} + Y_{w^k} \times X_{i2} \right) \right)  \tag{3}
\]

\[
X_{o2} + iY_{o2} = \left( X_{i1} - \left( X_{w^k} \times X_{i2} - Y_{w^k} \times Y_{i2} \right) \right)
+ \left( Y_{i1} - \left( X_{w^k} \times Y_{i2} + Y_{w^k} \times X_{i2} \right) \right)  \tag{4}
\]

Thus, each butterfly requires four real multiplications and six real additions. If the above equations are implemented using fixed point calculations, the error possibility of the result will increase. These errors [4] consist of round-off, overflow and coefficient quantization errors. To reduce the errors and provide high resolution, floating point adder/subtractor is replaced. This adder/subtractor operates in 3 stages including mantis alignment, mantis addition/subtraction and normalization. Thus, each of the above phases can be used as stages for pipeline implementation. We convert the first phase into two stages in order to achieve higher clock frequency [5], [6]. Figures 2 and 3 show the block diagram of the pipeline adder/subtractor and multiplier. It is not necessary to adjust the exponents in multiplier. Since the normalized inputs are loaded to the multiplier, the mantises are multiplied and the exponents are added and amended. The result may be shifted one bit leftward. Thus the exponent will be decremented one unit.

Figure 4 shows the internal schematic of the pipeline butterfly algorithm. To improve speed calculations in the Radix-2 butterfly algorithm, the pipeline registers are located after each addition, subtraction and multiplication blocks. Hence, the pipeline butterfly algorithm keeps the final result in the register to transfer it to the RAM by the next clock cycle. This FFT algorithm is exactly executed after \( \left( N/2 \log_2 N \right) + 11 \) clock pulses. The 11 clock pulses delay is created by 11 pipeline registers in adder, subtractor and multiplier in a serial butterfly block. Additionally, parallel design of the FFT algorithm decreases the calculation time significantly.

B. ROM and RAM Blocks

The ROM in Radix-2 FFT algorithm has two look-up tables to save \( W^k \) coefficients. These coefficients are inherently complex numbers so the real & imaginary parts are separately kept on that. Since the amplitude of the sine and cosine are the same in the four quarters, (they only differ in the signs) \( W^k \) are calculated just for \( k = 0 \) to \( k = N/4 \) and related sign are saved for \( k = 0 \) to \( N \) into the ROM.
Thus, it avoids using a large number of gates in the FPGA board.

There is a RAM in the FFT block diagram. According to the system workflow, two complex data must be read from the RAM simultaneously and loaded to the butterfly block. Meanwhile, the two outputs of the butterfly block have to be written in the RAM at the same time. Figure 5 shows the internal structure of the RAM with the capability of reading and writing the two complex data simultaneously.

**IV. IMPLEMENTATION RESULT**

The new architecture of the Radix-2 FFT algorithm Verilog code was written and simulated by Matlab software. The design code is downloaded to the Virtex-2 FPGA board. From Xilinx ISE synthesize report, it was shown minimum clock period is 9.94 ns (Maximum Frequency is 100.6 MHz). The chip layout on Virtex 2 FPGA board is shown in Figure 6.

Figure 7 shows the output digital signal of the new Radix-2 FFT algorithm which simulated by Modelsim software.

Figure 8 shows the measured signal spectrum achieved by Radix-2 FFT algorithm for 1024 floating point complex data.

The output result appeared after 5131 clock pulses which proved the complex calculation of \( \left( \frac{N}{2} \log_2 N \right) + 11 \) in the system. The System MSE was measured by Matlab software. It shows the mean squared error (MSE) of 0.0001.

**V. CONCLUSION**

The new architecture of the high speed and high resolution of the parallel, pipeline and floating point Radix-2 FFT algorithm was designed and investigated. High Speed FFT architecture was obtained by two methods. The pipeline structure and parallel design lead us to have high speed FFT algorithm. Additionally, using a internal RAM makes the design compatible with different type of FPGA board. The implementation result shows the maximum throughput of the 100.6 MHz in Virtex 2 FPGA board. Meanwhile, the resolution was increased by floating point calculation during the FFT process. The proposed FFT algorithm proves the latency of \( \left( \frac{N}{2} \log_2 N \right) + 11 \), 5131 clock pulses for \( N = 1024 \) with the mean squared error (MSE) of 0.0001.

**VI. ACKNOWLEDGMENT**

We wish to thank our friend, Dr. Mohamad Amir Amini who has assisted us for completion of this dissertation.
VII. Reference


