PERFORMANCE EVALUATIONS OF GRIORYAN FFT AND COOLEY-TUKEY FFT ONTO XILINX VIRTEX-II PRO AND VIRTEX-5 FPGAS

Narayanam Ranganadh[1] and Bindu Tushara D[2]

Department of Electronics & Communications Engineering
Bharat Institute of Engineering & Technology
Mangalpally (V), Ibrahimpatnam (M)
Hyderabad, A.P., India – 501 510
`ranganadh.narayanam@gmail.com, rnara100@uottawa.ca`
`rnara100@gmail.com`

ABSTRACT

A large family of signal processing techniques consist of Fourier-transforming a signal, manipulating the Fourier-transformed data in a simple way, and reversing the transformation. We widely use Fourier frequency analysis in equalization of audio recordings, X-ray crystallography, artefact removal in Neurological signal and image processing, Voice Activity Detection in Brain stem speech evoked potentials, speech processing spectrograms are used to identify phonetic sounds and so on. Discrete Fourier Transform (DFT) is a principal mathematical method for the frequency analysis. The way of splitting the DFT gives out various fast algorithms. In this paper, we present the implementation of two fast algorithms for the DFT for evaluating their performance. One of them is the popular radix-2 Cooley-Tukey fast Fourier transform algorithm (FFT) [1] and the other one is the Grigoryan FFT based on the splitting by the paired transform [2]. We evaluate the performance of these algorithms by implementing them on the Xilinx Virtex-II pro [3] and Virtex-5 [4] FPGAs, by developing our own FFT processor architectures. Finally we show that the Grigoryan FFT is working fatser than Cooley-Tukey FFT, consequently it is useful for higher sampling rates. Operating at higher sampling rates is a challenge in DSP applications.

1. INTRODUCTION

In the recent decades, fast orthogonal transforms have been widely used in areas of data compression, pattern recognition and image reconstruction, interpolation, linear filtering, and spectral analysis. The suitability of unitary transforms in each of the above applications depends on the properties of their basis functions as well as on the existence of fast algorithms, including parallel ones. Since the introduction of the Fast Fourier Transform (FFT), Fourier analysis has become one of the most frequently used tool in signal/image processing and communication systems; The main problem when calculating the transform relates to construction of the decomposition, namely, the transition to the short DFT's with minimal computational complexity.

The computation of unitary transforms are complicated and time consuming process. Since the decomposition of the DFT is not unique, it is natural to ask how to manage splittings and how to obtain the fastest algorithm of the DFT. The difference between the lower bound of arithmetical operations and the complexity of fast transform algorithms shows that it is possible to obtain FFT algorithms of various speed [2]. One approach is to design efficient manageable split algorithms. Indeed, many algorithms make different assumptions about the transform length. The signal/image processing related to engineering research becomes increasingly dependent on the development and implementation of the algorithms of orthogonal or non-orthogonal transforms and convolution operations in modern computer systems. The increasing importance of processing large vectors and parallel computing in many scientific and engineering applications require new ideas for designing super-efficient algorithms of the transforms and their implementations [2].

In this paper we present the implementation techniques and their results for two different fast DFT algorithms. The difference between the algorithm development lies in the way the two algorithms use the splitting of the DFT. The two fast algorithms considered are radix-2 and paired transform [2] algorithms. The implementation of the algorithms is done both on the Xilinx Viretx-II Pro [3] and Virtex-5 [4] FPGAs. The performance of the two algorithms is compared in terms of their sampling rates and also in terms of their hardware resource utilization.

Section 2 presents the paired transform decomposition used in paired transform in the development of Grigoryan FFT. Section 3 presents the implementation techniques for the radix-2 and paired transform algorithms on FPGAs. Section 4 presents the results. Finally with the Section 5 we conclude the work and put forward some suggestions for further sampling rate improvements.

2. DECOMPOSITION ALGORITHM OF THE FAST DFT USING PAIRED TRANSFORM

The print In this algorithm the decomposition of the DFT is done by using the *paired transform[2]*. Let $\{ x(n) \}$, n = 0:(N-1) be an input signal, N>1. Then the DFT of the input sequence $x(n)$ is

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

which is in matrix form

$$X = [F_N]x \qquad (2)$$

where X(k) and $x(n)$ are column vectors, the matrix $F_N = \left\| W_N^{nk} \right\|_{n,k=(0:N-1)}$, is a permutation of X.

$$F_N = diag\{[F_{N1}],[F_{N2}],....[F_{Nk}]\}[\overline{W}][\chi_N^{'}] \qquad (3)$$

which shows the applying transform is decomposed into short transforms F_{Ni}, i = 1: k. Let S_F be the domain of the transform F the set of sequences f over which F is defined. Let (D; σ) be a class of unitary transforms revealed by a partition σ. For any transform $F \in$ (D; σ), the computation is performed by using paired transform in this particular algorithm. To denote this type of transform, we introduce *"paired functions [2]."*

Let $p, t \in$ period N, and let

$$\chi_{p,t}(n) = \begin{cases} 1; np = t(\text{mod } N); \\ 0; otherwise. \end{cases} \quad n = 0: (N\text{-}1) \quad (4)$$

Let L be a non trivial factor of the number N, and $W_L = e^{2\Pi/L}$, then the complex function

$$\chi'_{p,t} = \chi'_{p,t;L} = \sum_{k=0}^{L-1} W_L^k \chi_{p,t+kN/l} \quad (5)$$

$$t = 0: (N/L-1), \text{ p}\in \text{ to the period 0: N-1}$$

is called L-paired function [2]. Basing on this paired functions the complete system of paired functions can be constructed. The totality of the paired functions in the case of $N = 2^r$ is

$$\{\{\chi_{2^n,2^n t}; t = 0: (2^{r-n-1} - 1), n = 0: (r-1)\}, 1\} \quad (6)$$

Now considering the case of $N = 2^r N_1$, where N_1 is odd, $r \geq 1$ for the application of the paired transform.

a) The totality of the partitions is
$$\sigma' = (T'_{1;2}, T'_{2;2}, T'_{4;2}, \ldots, T'_{2^r;2})$$

b) The splitting of F_N by the partition σ' is
$$\{F_{N/2}, F_{N/4}, \ldots, F_{N/2^r}, F_{N1}\}$$

c) The matrix of the transform can be represented as
$$[F_N] = \left(\bigoplus_{n=1}^{r} [F_{Ln}]\right) [\overline{W}][\chi'_n]$$
$$L_n = N/2^{n+1}, L_r = N_1.$$

where the $[\overline{W}]$ is diagonal matrix of the twiddle factors. The steps involved in finding the DFT using the paired transform are given below:

1) Perform the L-paired transform g = $\chi'_N(x)$ over the input x

2) Compose r vectors by dividing the set of outputs so that the first L^{r-1} elements $g_1, g_2, \ldots, g_L^{r-1}$ compose the first vector X_1, the next L^{r-2} elements $g_{Lr-1}+1, \ldots, g_{Lr-1+Lr-2}$ compose the second vector X_2, etc.

3) Calculate the new vectors Y_k, k=1:(r-1) by multiplying element-wise the vectors X_k by the corresponding turned factors $1, W_t, W_t^2, \ldots, W_t^{t/L-1}, (t = L^{r-k}), (t=Lr-k)$. Take $Y_r = X_r$

4) Perform the L_{r-k}-point DFT's over Y_k, k=1: r
5) Make the permutation of outputs, if needed.

3. IMPLEMENTATION TECHNIQUES

We have implemented various architectures for radix-2 and paired transform processors on Virtex-II Pro and Virtex-5 FPGAs. As there are embedded multipliers [3] and embedded block RAMs [3] available, we can use them without using distributed logic, which economize some of the CLBs [3]. Virtex-5 [4] is having DSP48E slices As we are having DSP48E slices on Virtex-5 FPGAs, to utilize them and improve speed performance of these 2 FFTs and to compare their speed performances on Virtex-5 FPGAs and Virtex-II pro FPGAs. As most of the transforms are applied on complex data, the arithmetic unit always needs two data points at a time for each operand (real part and complex part), dualport RAMs are very useful in all these implementation techniques.

In the Fast Fourier Transform process the butterfly operation is the main unit on which the speed of the whole process of the FFT depends. So the faster the butterfly operation, the faster the FFT process. The adders and subtractors are implemented using the LUTs (distributed arithmetic). The inputs and outputs of all the arithmetic units can be registered or non-registered.
Various possible implementations of multipliers we considered are:

Embedded multiplier:

a) With non-registered inputs and outputs
b) With registered inputs or outputs, and
c) With registered inputs and outputs.

Distributed multiplier: Distributed multipliers are implemented using the LUTs in the CLBs. These can also be implemented with the above three possible ways. Various considerations made to implement butterfly operation for its speed improvement and resource requirements. Basing on the availability of number of Embedded multipliers and design feasibility we have implemented both multiplication processes.

The various architectures proposed for implementing radix-2 and paired transform processors are single memory (pair) architecture, dual memory (pair) architecture and multiple memory (pair) architectures. We applied the following two best butterfly techniques for the implementation of the processors on the FPGAs [3].

1. One with Distributed multipliers, with fully pipelined stages. (Best in case of performance)
2. One with embedded multipliers and one level pipelining. (Best in case of resource utilization)

Single memory (pair) architecture (shown in Figure 1) is suitable for single snapshot applications, where samples are acquired and processed thereafter. The processing time is typically greater than the acquisition time. The main disadvantage in this architecture is while doing the transform process we cannot load the next coming data. We have to wait until the current data is processed. So we proposed dual memory (pair) architecture for faster sampling rate applications (shown in

Figure 2). In this architecture there are three main processes for the transformation of the sampled data. Loading the sampled data into the memories, Processing the loaded data, Reading out the processed data. As there are two pairs of dual port memories available, one pair can be used for loading the incoming sampled data, while at the same time the other pair can be used for processing the previously loaded sampled data. For further sampling rate improvements we proposed multiple memory (pair) architecture (shown in Figure 3). This is the best of all architectures in case of very high sampling rate applications, but in case of hardware utilization it uses lot more resources than any other architecture. In this model there is a memory set, one arithmetic unit for each iteration. The advantage of this model over the previous models is that we do not need to wait until the end of all iterations (i.e. whole FFT process), to take the next set of samples to get the FFT process to be started again. We just need to wait until the end of the first iteration and then load the memory with the next set of samples and start the process again. After the first iteration the processed data is transferred to the next set of RAMs, so the previous set of RAMs can be loaded with the next coming new data samples. This leads to the increased sampling rate.

Coming to the implementation of the paired transform based DFT algorithm, there is no complete butterfly operation, as that in case of radix-2 algorithm. According to the mathematical description given in the Section II, the arithmetic unit is divided into two parts, addition part and multiplication part. This makes the main difference between the two algorithms, which causes the process of the DFT completes earlier than the radix-2 algorithm. The addition part of the algorithm for 8-point transform is shown in Figure 4. The architectures are implemented for the 8-point 64-point,128-point,256-point transforms for both Viretx-II Pro and Virex-5 FPGAs. The radix-2 FFT algorithm is efficient in case of resource utilization and the paired transform algorithm is very efficient in case of higher sampling rate applications.

4. THE IMPLEMENTATION RESULTS

Results obtained on Virtex-II Pro and Virtex-5 FPGAs: The hardware modeling of the algorithms is done by using Xilinx's system generator plug-in software tool running under SIMULINK environment provided under the Mathworks's MATLAB software. The functionality of the model is verified using the SIMULINK Simulator and the MODELSIM software as well. The implementation is done using the Xilinx project navigator backend software tools.

Table 1 shows the implementation results of the two algorithms on the Virtex-II Pro FPGAs. Table 2 shows the implementation results of the two algorithms on Virtex-5 FPGAs. From Tables 1, 2 we can see that Grigoryan FFT is always faster than the Cooley-Tukey FFT algorithm. Thus paired-transform based algorithm can be used for higher sampling rate applications. In military applications, while doing the process, only some of the DFT coefficients are needed at a time. For this type of applications paired transform can be used as it generates some of the coefficients earlier, and also it is very fast.

Table 1 Efficient performance of Grigoryan FFT over Cooley-Tukey FFT, on Virtex-II Pro FPGAs. Table showing the sampling rates and the resource utilization summaries for both the algorithms, implemented on the Virtex-II Pro FPGAs.

No. of points	Cooley-Tukey radix-2 FFT			Grigoryan Radix-2 FFT		
	Max. freq. MHz	No. of mult.	No. of Slices	Max. Freq. MHz	No. of mult	No. of slices
8	35	4	264	35	8	475
64	42.45	4	480	52.5	8	855
128	50.25	12	560	60	16	1248
256	55.40	24	648	68.75	48	1985

Table 2 Efficient performance of Grigoryan FFT over Cooley-Tukey FFT, on Virtex-5 FPGAs. Table showing the sampling rates and the resource utilization summaries for both the algorithms, implemented on the Virtex-5 FPGAs. We have utilized DSP48E slices in this, which is making us much faster than Virtex-II Pro FPGAs.

No. of points	Cooley-Tukey radix-2 FFT			Grigoryan Radix-2 FFT		
	Max. freq. MHz	No. of Slices	No. of DSP48E slices	Max. freq MHz	No. of Slices	No.of DSP48E slices
8	48	200	4	52	350	4
64	55.25	375	6	60	700	8
128	60.50	450	11	80	1000	26
256	75	560	14	85	1645	82

5. CONCLUSIONS AND FURTHER RESEARCH

In this paper we have shown that both on on Virtex-II Pro and Virtex-5 FPGAs the paired transform based Grigoryan FFT algorithm is faster and can be used at higher sampling rates than the Cooley-Tukey FFT at an expence of high resource utilization.

1. In all implementations on FPGAs, the number of bits used for the data is 16-bits. So all the multipliers here are used as 16-bit multipliers. The size of the multipliers used were 18-bit multipliers. For instance, if there are some applications using only 8-bit data, then one can use the 40 dedicated multipliers as 80 multipliers, as two multiplications can be implemented by using a single embedded multiplier as long as the sum of the two products bits is less than 36 bits.

2. We are planning to implement these algorithms onto Latest TMS DSP processors, and also by utilizing MAC engines on them; and onto Some more Xilinx FPGAs for extensive verification of these two FFT algorithms. Then we would like to standardize Grigoryan FFT and apply for real time applications.

ACKNOWLEDGEMENTS

REFERENCES

[1] B James W. Cooley and John W. Tukey, "An algorithm for the machine calculation of complex Fourier Series", Math. comput. 19, 297-301 (1965)

[2] B Split Manageable Efficient Algorithm for Fourier and Hadamard transforms Artyom M. Grigoryan and Sos S. Agaian, Signal Processing, IEEE Transactions on, Volume: 48, Issue: 1, Jan.2000 Pages: 172 - 183

[3] Virtex-II Pro platform FPGAs: detailed description
http://www.xilinx.com/support/documentation/data_sheets/ds083.pdf

[4] Virtex-5 platform FPGAs: detailed description http://www.xilinx.com/support/documentation/virtex-5_user_guides.htm

[5] Case study of Grigoryan FFT onto FPGAs and DSPs, Narayanam Ranganadh, Parimal A Patel and Artyom M. Grigoryan. IEEE proceedings, ICECT – 2012.

Figures
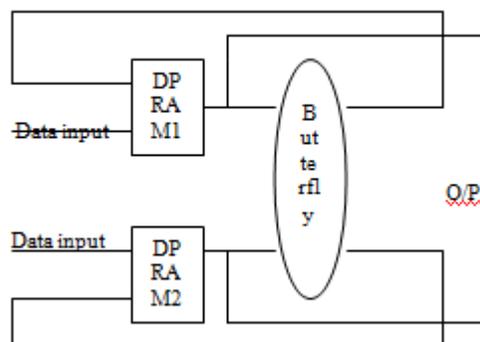
Figure 1 Single memory (pair) architecture
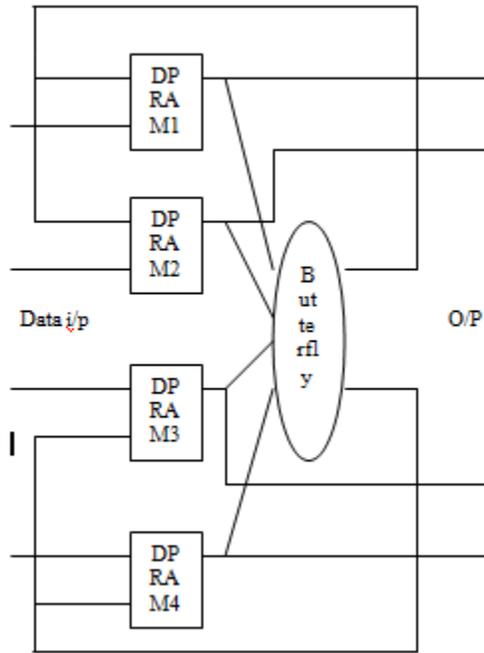
Figure 2 Dual memory (pair) architecture
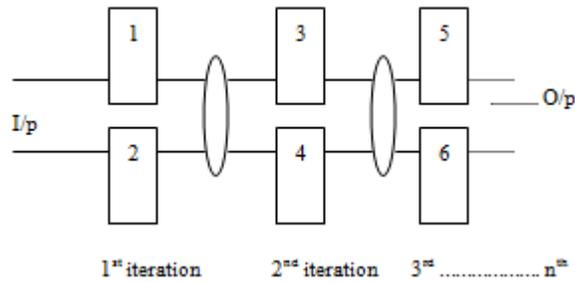
Figure 3 Multiple memory (pair) architecture

(Transform length = N = 2^n)
(1,2);(3,4);(5,6) ---- (-,-) memory pairs for each iteration.

----- Butterfly unit for each iteration.
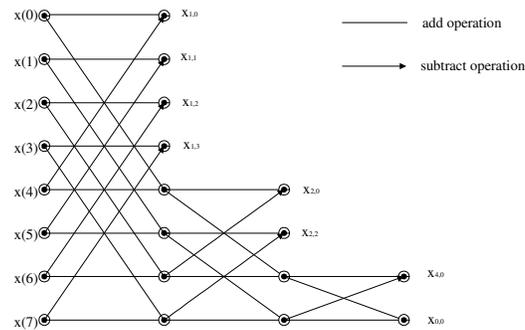
Figure 4 Figure showing the addition part of the 8-point paired transform based DFT

Authors

Mr. Ranganadh Narayanam is an Assistant professor in the department of Electronics & Communications Engineering in Bharat Institute of Engineering & Technology (BIET). This research is continuation of the research done in the Univeristy of Texas at San Antonio under the guidance of Dr. Parimal A. Patel, Dr. Artyom M. Grigoryan, as my master's thesis. This current research was partly funded by BIET. Mr.Narayanam, a research student in the area of "Brain Stem Speech Evoked Potentials" under the guidance of Dr. Hilmi Dajnai of University of Ottawa,Canada. He was also a research student in The University of Texas at San Antonio under Dr. Parimal A Patel,Dr. Artyom M. Grigoryan, Dr Sos Again, Dr. CJ Qian,. in the areas of signal processing and digital systems, control systems. He worked in the area of Brian Imaging in University of California Berkeley. Mr. Narayanam is having around 5+2 years of full time teaching & research experiences respectively; and more than 5 years of entry level research experience and more than 10 publications. Mr. Narayanam's research interests include neurological Signal & Image processing, DSP spftware & Hardware design and implementations, neurotechnologies.

Ms. Bindu Tushara D. is currently an Assistant Professor in Vignan Institute of Technology & Science (VITS). She is a double gold medalist of Jawaharlal Nehru Technological University as the best out going student of the year 2009. She has received medals from Governer of Andhara Pradesh. Her research interests are DSP and Wireless Communications.