Implementation and performance evaluation of paired transform based Faster FFT: Grigoryan FFT on Xilinx FPGAs and TMS DSPs using MATLAB: SIMULINK and CC Studio

Ranganadh Narayanam¹, Artyom M. Grigoryan², Parimal A. Patel³, Bindu Tushara D⁴.

¹Associate Professor, Department of ECE, Aurora’s Scientific and Technological Institute, Hyderabad, India.
ranganadh.narayanam@gmail.com

²Associate Professor, The university of Texas at San Antonio, TX, USA.

³Xilinx Key Global trainer and consultant, Xilinx corporation, California, USA.

⁴Assistant Professor, Department of ECE, Vignan Institute of Technology and Science, Hyderabad, India.

Abstract– Discrete Fourier Transform is a principal mathematical method for the frequency analysis and has wide applications in Engineering and Sciences. Because the DFT is so ubiquitous, fast methods for computing DFT have been studied extensively, and continuous to be an active research. 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 [6], Virtex-4[9] and Virtex-5[7] FPGAs, by developing our own FFT processor architectures. We have evaluated the performances also by implementing on Texas Instruments fixed point DSP processors: TMS320C5416[17], TMS320C6748[17], TMS320C5515[17]. Finally we show that the Grigoryan FFT is working faster than the Cooley-Tukey FFT, consequently it is useful for higher sampling rates. Operating at higher sampling rates is a challenge in DSP applications. We proved that on Xilinx FPGAs and TMS DSPs, the Grigoryan FFT is performing at most 1.358 and 1.7 times faster than the Cooley-Tukey FFT respectively. We also confirm that for the same architectures Virtex-5 platform is better platform for implementing the Grigoryan FFT.

Index Terms–frequency analysis, fast algorithms, DFT, FFT, paired transforms, SIMULINK, CC Studio.

I. INTRODUCTION

In the recent decades DFT has been playing several important roles in advanced applications such as image compression and reconstruction in biomedical images, audiology research for analyzing biomedical brain-stem speech signals, sound filtering, data compression, partial differential equations, and multiplication of large integers. The fast algorithms for DFT always look for DFT process to be fast, accurate and simple. Fast is the most important [10]. FFT is universal in signal processing, but it can also be used to compress image and audio files, solve differential equations and price stock options, among other things.

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 is complicated and time consuming process. Since the decomposition of the DFT is not unique, it is natural to ask how to manage splitting 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 [2]. 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 the current age there are some algorithms and implementations that are coming for the Faster FFT, than Cooley-Tukey FFT for several different applications. One of them is “University of Michigan FFT algorithm”: Anna Gilbert, Martin Strauss. The second one is “MIT FFT Algorithm”: Dina Katabi, Piotr Indyk, Eric Price, Haitham Hassanieh. Ours is also one of them, it is the paired transform-based Grigoryan FFT which can be essentially applicable in military applications in Aviation and electronic warfare antennas, medical nano-robots and nano-enhanced reconnaissance and communication devices, where faster speed of FFT operation, requirement of some of the FFT coefficients earlier than waiting until the final stage of FFT; are highly useful. The Paired transform based Grigoryan FFT is a highly powerful tool.

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 (Cooley-Tukey FFT) and paired transform [2] (Grigoryan FFT) algorithms. Implementation is done both on Xilinx FPGAs and Texas Instruments DSP processors. For FPGAs the modeling and simulations are done on SIMULINK of MATLAB with XSG: Xilinx System Generator (a high level visual tool for hardware generation). The implementation is done using the Xilinx project navigator backend software tools. We have developed specific C programs for TMS DSP processors using the Code Composer Studio Integrated Development Environment (CC Studio IDE), with implicit utilization of MAC engines.

Great speedups can be achieved for these algorithms by efficient implementation in dedicated hardware such as Application-Specific Integrated Circuits (ASICs). However, high “time-to-market” has been a bottleneck for the ASICs. The evolution of Field Programmable Gate Arrays (FPGAs) along with high-level design tools such as from Altera, Xilinx System Generator have come as valuable and effective tool for high-level programmers to achieve better execution times in these reconfigurable hardware. The small time-to-market for FPGAs over VLSI models is the reason for popular choice of FPGAs in current market. FPGA expedite the time lag between hardware design and shipping time of the circuit from 2-3 years to a few weeks [16]. Advances in FPGA technology along with development of elaborate and efficient tools for modelling, simulation and synthesis have made FPGAs a highly useful platform. With a graphical environment based on SIMULINK and a pre-defined block set of Xilinx DSP cores, System Generator[14]-[15] meets the needs of both system architects who need to integrate the components of a complete design and hardware designers who need to optimize implementations. The salient features of FPGAs that make them superior in speed, over conventional general purpose hardware like Pentiums are their greater I/O bandwidth to local memory, pipelining, parallelism and availability of optimizing compiler [15]-[16]. Complex tasks, which involve, multiple image operators, run much faster on FPGAs than on Pentiums, in fact, some researches report an 800-time speed up by FPGA. There are several reasons for such large speed ups which FPGAs have over PCs. In comparison to an FPGA, hardware such as Pentium runs at memory speed, not at cache speed. So, even running at much higher clock frequency and having the facility of cache memory, it responds much slower than a comparable FPGA [16]. Frequency of operation in hardware such as Pentium can be increased up to a certain extent to increase the performance or the required data rate to process the image data, but increasing the frequency above certain limits causes system level and board level issues that become a bottleneck in the design. Considering all the advantages and to explore all these features we have chosen to implement on FPGAs. Implementing on DSP processors is a compulsory task for any DSP applications. The implementation of the algorithms is done in Hardware point of view on the Xilinx Virtex-II Pro [6], Virext-4 [9], and Virtex-5 [7] FPGAs. Then we have implemented on Texas Instruments DSP Processors: TMS320C5416 [17], TMS320C6748 [17], TMS320C5515 [17]. The performance of the two algorithms is compared in terms of their sampling rates and also in terms of their hardware resource utilization.

The paper is organized in the following way. Section 2 presents the paired transform decomposition used in paired transform in the development of Grigoryan FFT. In Section 3 we present the implementation techniques for the radix-2 and paired transform algorithms on Xilinx
FPGAs and TMS DSP processors. Section 4 presents the results. Finally with the Section 5 we conclude the work and further research.

2. DECOMPOSITION ALGORITHM OF THE FAST DFT USING PAIRED TRANSFORM

We consider the fast splitting of the discrete Fourier transform by the 1-D discrete paired transform (DPT) [2]-[5]. Let \( \{x_n; n = 0(N - 1)\} \) be the input signal of length \( N > 1 \). The \( N \)-point DFT of the signal \( x \) is defined as

\[
X_p = (F_N^p)x = \sum_{n=0}^{N-1} x_n W^{np}, \quad p = 0:(N - 1),
\]

where the number \( W = W_N = \exp(-2\pi i/N), i^2 = -1 \). This transform can be written in matrix form as \( X = [F_N]x \), where \( X \) and \( x \) are column-vectors for \( X \) and \( x \), and the matrix \( [F_N] = ||W^{np}||_{n,p=0,(N-1)} \). We consider the case of most interest, when the signal length is \( N = 2^r, r > 1 \). Unlike the Cooley-Tukey algorithm, on first stage of which \( F_N \) is calculated by two \( F_{N/2} \), Grigoryan splits the transform \( F_N \) by \( (r + 1) \) short transforms as \( \{F_{N/2}, F_{N/4}, F_{N/8}, \ldots, F_2, F_1, F_2\} \). Namely, the following matrix decomposition holds for the DFT:

\[
[F_N] = (\bigotimes_{k=0}^{r-1} [F_{N/2^k+1}]) \oplus 1 \cdot D_N \cdot [x_N]
\]

where \( \oplus \) denotes the operation of the Kronecker sum of matrices and the diagonal matrix

\[
D_N = \text{diag}\{1, W, W^2, W^3, \ldots, W^{N/2-1}, W^2, W^4, \ldots, W^{N/2-2}, 1, W^4, W^8, \ldots, W^{N/2-4}, 1, \ldots, 1\}.
\]

The \( N \)-point unitary and binary discrete paired transform DPT \( [x_N'] \) is described in the following way [3]. Given frequency-point \( p = 0 \) and time \( t \in \{0, 1, \ldots, N - 1\} \), let \( x_{p,t}(n) \) be the function

\[
x_{p,t}(n) = \begin{cases} 1, & \text{if } np = t \mod N, \\ 0, & \text{otherwise,} \end{cases} \quad n = 0: (N - 1)
\]

The 2-paired, or shortly the paired function is defined as

\[
x'_{p,t}(n) = x_{p,t}(n) - x_{p,t+N/2}(n).
\]

The totality of the paired functions

\[
\{x'_{2^k}; k \in 01, 2; \ldots; N/2 - 1\}
\]

is the complete and orthogonal set of functions of the paired transform \( x_N' \). Figure 1 shows two matrices of the 16- and 32-point paired transform \( x_N' \) in parts a and b, respectively.

The number of operations of multiplication required to calculate the paired \( N \)-point FFT equals \( M_N = N/2(\log_2 N - 3) + 2, \ r > 2 \). The \( N \)-point discrete paired transform is fast and requires \( (2N - 2) \) operations of addition/subtraction. Figure 2 shows the signal-flow graph of the 8-point fast paired transform.

The double numbering of the paired functions refers to the frequency \( (p = 2^k) \) and time \( (t) \). The paired transform represents the discrete-time signal \( x_n' \) as the unique set of frequency-time signals (or splitting-signals),

\[
\left\{x_{2^k,0,0}'; x_{2^k,2,0}'; x_{2^k,2,2}'; \ldots; x_{2^k,2^{k-1},2^{k-1}}'; \ldots; x_{N/2^k,0}'; x_{N/2^k,2^k-1,2^k-1}\right\}_{k=0: (r-1)}
\]

The components of these splitting-signals are calculated by
Where \( t = 0: \left( \frac{N}{2^k+1} - 1 \right) \) In the paired transform of the signal, the first \( \frac{N}{2} \) components are the splitting-signal with \( k = 0 \) the next \( \frac{N}{4} \) components are the splitting-signal with \( k = 1 \), and so on. The last component of the transform is \( x_{k,0} = x_1 + x_2 + \cdots + x_{N-1} \). Each splitting-signal defines the \( \frac{N}{2^k} \)-point DFT of \( x_{k} \) at frequency-points of the corresponding subset 

\[
T'_p = \{(2m + 1)p \mod N; m = 0:(N/(2^p) - 1)\}
\]

Indeed, the following is valid:

\[
X_{(2m+1)p} = \sum_{p=0}^{N/(2^p)-1} \left(x'_p W_{N/p}^m \right) W_{N/(2^p)}^m \]

The set of \( N \) frequency-points \( \{0, 1, 2, \ldots, N - 1\} \) is divided by subsets \( T'_p \), where \( p = 2^k \), \( k = 0: (r - 1) \) and \( T'_0 = \{0\} \).

**Example 1:** Consider the signal \( X \) of length \( N = 256 \), which is shown in Figure 3 in part a. The paired transform of the signal, which is the set of nine splitting-signals \( \{x'_0, x'_1, x'_2, x'_3, x'_4, x'_5, x'_6, x'_7\} \) is shown in part b. The vertical dashed lines separate the first seven splitting-signals. Figure 4 shows the 256-point DFT \( X'_N \) of the signal in absolute mode in part a. In part b, the same DFT is shown, but in the order that corresponds to the partition of the frequency-points \( \{0, 1, 2, \ldots, 255\} \) by subsets \( T'_1, T'_2, T'_4, \ldots, T'_{128} \) and \( T'_0 \). The first part with 128 values corresponds to the 128-point DFT of the modified splitting-signal \( \{X'_{128} W_{128}^m\} \). The next part with 64 values corresponds to the 64-point DFT of the modified splitting-signal \( \{X'_{128} W_{128}^m\} \), and so on. Therefore, the following steps are involved in computing the DFT of the input signal \( x \), by using the paired transform:

1. Perform the paired transform \( X'_N[x] \) over the signal \( x \).
2. Compose \((r + 1)\) vectors by dividing the transform into the splitting-signals \( \{x'_p, t = 0: (N/(2^p) - 1)\} \), where \( p = 2^k \), \( k = 0: (r - 1) \) and \( p = 0 \).
3. Modify the splitting-signals by the corresponding twiddle factors,
4. Perform the \( N/(2^p) \)-point DFTs over the modified splitting-signals, when \( p \neq 0 \).
5. Perform the permutation of the output, if needed.
3. IMPLEMENTATION TECHNIQUES

3.1 Implementation on Xilinx FPGAs

Choosing an appropriate tool for FPGA design is of crucial importance as it affects the cost, development time and various other aspects of design. SIMULINK is a platform for multi-domain simulation and Model-Based Design for dynamic systems. It provides an interactive graphical environment and a set of block libraries, and can be extended for different specialized applications. Using SIMULINK one can quickly build up models from libraries of pre-built blocks. For high level design we have chosen Xilinx System Generator. It is a DSP design tool from Xilinx that enables the use of the Mathworks model-based design environment SIMULINK for FPGA design. Xilinx System Generator (XSG) for DSP is a tool which offers block libraries that plugs into SIMULINK tool (containing bit-true and cycle-accurate models of their FPGAs particular math, logic, and DSP functions) [14]-[16]. It is a system-level modeling tool in which designs are captured in the DSP friendly SIMULINK modeling environment using a Xilinx specific blockset. All of the downstream FPGA implementation steps including synthesis and place and route are automatically performed to generate an FPGA programming file. Over 90 DSP building blocks are provided in the Xilinx DSP blockset for SIMULINK. System Generator for DSP, is a system level design tool that is a Blockset for MATLAB SIMULINK. It has different levels of support for the DSP slices.

The hardware modelling 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. Many blocks of custom MATLAB code (.m files) were however needed for the design and the hardware generated for these blocks was not optimized. We used DSP IP with DSP slices support, and the DSP slice to build custom functions, the DSP slice macro to simplify sequential instructions.

We have implemented various architectures for radix-2 and paired transform processors on Xilinx Virtex-II Pro, Virtex-4, Virtex-5 FPGAs. As there are embedded dedicated multipliers and embedded block RAMs available, we can use them without using distributed logic, which economize some of the CLBs. As we are having Extreme DSP slices [9] on Virtex-4 FPGAs, DSP48E Slices [7] on Virtex-5 FPGAs we have utilized them to improve speed performance of these 2 FFTs and to compare their speed performances. 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), dual port 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.

We have considered the implementation of both embedded and distributed multipliers; the latter are implemented using the LUTs in the CLBs. The three considerations for inputs/outputs are with non-registered inputs and outputs, with registered inputs or outputs, and with registered inputs and outputs. To implement butterfly operation for its speed improvement and resource requirement, we have implemented both multiplication procedures basing on the availability of number of embedded multipliers (especially for Virtex-II Pro FPGAs), and design feasibility; and design feasibility using DSP Slices.

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.

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

Single memory (pair) architecture (shown in Figure 5) 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 6). 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.

![Single memory (pair) architecture](image1)

![Dual memory (pair) architecture](image2)

For further sampling rate improvements we proposed multiple memory (pair) architecture (shown in Figure 7). 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 2, 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 2.

![Multiple memory (pair) architecture](image3)

(Transform length = $N = 2^n$)

$((1,2);(3,4);(5,6))$ ---- (-,-) memory pairs for each iteration.

![SIMULINK models](image4)

The SIMULINK model diagrams for butterfly operation for both FFTs are given in Figure 8.

![SIMULINK models](image5)

(a) Cooley-Tukey FFT

(b) Grigorayn FFT

Figure 8. SIMULINK models (a) for butterfly diagram for N=8 Cooley-Tukey FFT (b) for the addition part shown in figure 2 of Paired transform based FFT: Grigorayn FFT.
The architectures are implemented for the 8-point, 64-point, 128-point and 256-point transforms for Xilinx Virtex-II Pro, Virtex-4 and Virtex-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 speed of operation and hence higher sampling rate applications.

3.2 Implementation on Texas Instruments DSP Processors.

We have developed Programs for the Cooley-Tukey FFT and Grigoryan FFT algorithms in C language specific for the Texas Instruments DSP processors by using the software Texas Instruments Code Composer Studio Integrated development environment. The C programming is done for the implementation on TMS fixed point DSP Processors: TMS320C6748, TMS320C5416, and TMS320C5515. The features of DSP specific capabilities, advanced break points, the conditional or hardware break points for C expressions, local variable and registers are utilized to the best for simple and efficient implementations. The advanced memory window is used for the observation of memory at each level. Using the CC studio we can be able to easily and quickly measure code performance and ensure the efficient use of the DSP target’s resources during debug and development sessions. Detailed simulations are observed using the features of the CC Studio for finding the number of CCs and finding the speed of operation of the two algorithms while implementing on the three DSP processors.

4. PRIMILINARY IMPLEMENTATION RESULTS

4.1 Results on Xilinx FPGAs

We have implemented all 3 different memory pair architectures explained in Section 3. But here we are providing the results for multiple memory pair architecture in Figure 7 as it is the most efficient of all. We are showing the results and efficiency comparison of both Cooley-Tukey and Grigoryan FFT on Xilinx Virtex-II Pro, Virtex-4 and Virtex-5 FPGAs for the same architectures. Then we are showing the % improvement of the Grigoryan FFT over the Cooley-Tukey FFT and providing how many times the Grigoryan FFT is faster than the Cooley-Tukey FFT. Then we are comparing the three FPGA platforms to verify which one is better platform for implementing our architectures. From the results we can easily identify that the Grigoryan FFT is much faster than the Cooley-Tukey FFT.

Tables 1,2,3 and 4 show the implementation results of the two algorithms on all the considered Xilinx FPGAs. From these results we can see that the 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. As of verification of better platform on which we can implement our architectures it is found that Virtex-5 FPGAs are better platform for our architectural implementations. By observing the results the Grigoryan FFT is performing at most 24.09 % faster. If we observe the number of times speed, it is clear that the Grigoryan FFT is working at most 1.358 times faster than the Cooley-Tukey FFT.

Table 1. Efficient performance of the Grigoryan FFT over Cooley-Tukey FFT, on Xilinx 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.
Table 2. Efficient performance of the Grigoryan FFT over the 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.

Table 3. Efficient performance of the Grigoryan FFT over the Cooley-Tukey FFT, on Virtex-4 FPGAs. Table showing the sampling rates and the resource utilization summaries for both the algorithms, implemented on the Virtex-4 FPGAs. We have utilized Extreme DSP slices in this, which is making us much faster than Virtex-II Pro FPGAs.

Table 4. The percentage improvement in speed of operation over Virtex-II Pro FPGAs, of Virtex-5 and Virtex-4; of both Cooley-Tukey and Grigoryan FFT algorithms. It shows clearly that Virtex-5 platform is better than Virtex-4 and also Virtex-II pro, in terms of speed of operation.

4.2 Results on Texas Instruments DSP processors

After implementing on the three TMS DSP processors we have observed the performances for N = 16, 32, 64, 128, 256, 512, 1024 point FFTs for both Cooley-Tukey FFT and Grigoryan FFT. The implementation results are given in Tables 5-10 for all the three DSP processors. By observing on all the three TMS DSP processors, the % speed improvement of Grigoryan FFT over Cooley-Tukey FFT it is clear that the Grigoryan FFT is performing much better and can be utilized for higher sampling rates of operation. The Grigoryan FFT is performing at most 70% faster. If we observe the number of times speed, it is clear that the Grigoryan FFT is working at most 1.7 times faster than the Cooley-Tukey FFT. An example graphical representation, for TMS320C6748 DSP, of the speed improvement of the Grigoryan FFT over the Cooley-Tukey FFT is given in the Figure 9.

Table 5 Performance comparison of the two algorithms on DSP processor TMS320C6748 (fixed point) processor.
Table 6: Table showing the sampling rate of both the algorithms (starting form N = 16 to N = 1024) for TMS320C6748.

<table>
<thead>
<tr>
<th>No. of Samples</th>
<th>Radix-2 FFT</th>
<th>Paired transform</th>
<th>%Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>101580</td>
<td>92346</td>
<td>10</td>
</tr>
<tr>
<td>32</td>
<td>275363</td>
<td>222067</td>
<td>24</td>
</tr>
<tr>
<td>64</td>
<td>489229</td>
<td>391383</td>
<td>25</td>
</tr>
<tr>
<td>128</td>
<td>1542693</td>
<td>1224360</td>
<td>26</td>
</tr>
<tr>
<td>256</td>
<td>2589026</td>
<td>1646276</td>
<td>53</td>
</tr>
<tr>
<td>512</td>
<td>4807175</td>
<td>3004609</td>
<td>60</td>
</tr>
<tr>
<td>1024</td>
<td>12682383</td>
<td>9220491</td>
<td>70</td>
</tr>
</tbody>
</table>

Table 7: Performance comparison of the two algorithms on DSP processor TMS320C5416 (fixed point) processor.

<table>
<thead>
<tr>
<th>No. of CCs</th>
<th>Sample rate (MHz)</th>
<th>Number of times over Cooley-Tukey FFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.65</td>
<td>18.99</td>
<td>1.076</td>
</tr>
<tr>
<td>11.99</td>
<td>16.42</td>
<td>1.360</td>
</tr>
<tr>
<td>9.18</td>
<td>17.34</td>
<td>1.223</td>
</tr>
<tr>
<td>9.25</td>
<td>12.54</td>
<td>1.336</td>
</tr>
<tr>
<td>10.23</td>
<td>16.58</td>
<td>1.621</td>
</tr>
<tr>
<td>11.65</td>
<td>17.89</td>
<td>1.536</td>
</tr>
<tr>
<td>8.55</td>
<td>12.19</td>
<td>1.426</td>
</tr>
</tbody>
</table>

Table 8: Table showing the sampling rate of both the algorithms (starting form N = 16 to N = 1024) for TMS320C5416.

<table>
<thead>
<tr>
<th>Number of CCs</th>
<th>Sample rate (MHz)</th>
<th>Number of times over Cooley-Tukey FFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.75</td>
<td>17.33</td>
<td>1.000</td>
</tr>
<tr>
<td>11.63</td>
<td>14.41</td>
<td>1.239</td>
</tr>
<tr>
<td>13.08</td>
<td>16.33</td>
<td>1.250</td>
</tr>
<tr>
<td>8.70</td>
<td>10.45</td>
<td>1.361</td>
</tr>
<tr>
<td>9.929</td>
<td>15.18</td>
<td>1.520</td>
</tr>
<tr>
<td>10.65</td>
<td>17.64</td>
<td>1.600</td>
</tr>
<tr>
<td>6.55</td>
<td>11.11</td>
<td>1.696</td>
</tr>
</tbody>
</table>

Table 9: Performance comparison of the two algorithms on DSP processor TMS320C5515 (fixed point) processor.

<table>
<thead>
<tr>
<th>No. of Samples</th>
<th>Radix-2 FFT</th>
<th>Paired transform based FFT</th>
<th>%Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>101600</td>
<td>90235</td>
<td>12</td>
</tr>
<tr>
<td>32</td>
<td>275160</td>
<td>220690</td>
<td>25.02</td>
</tr>
<tr>
<td>64</td>
<td>461111</td>
<td>330273</td>
<td>39.61</td>
</tr>
<tr>
<td>128</td>
<td>1540680</td>
<td>1201321</td>
<td>28.25</td>
</tr>
<tr>
<td>256</td>
<td>2280014</td>
<td>1609342</td>
<td>41.67</td>
</tr>
<tr>
<td>512</td>
<td>4803439</td>
<td>3004099</td>
<td>59.89</td>
</tr>
<tr>
<td>1024</td>
<td>15230721</td>
<td>9002381</td>
<td>69.18</td>
</tr>
</tbody>
</table>

Figure 9: For TMS320C6748 DSP. On Y-axis the number of clock cycles taken to process the FFT of size N = 16,32,64,128,256,512,1024. N on X-axis. Series 2,3 are for the number of clock cycles taken for the Cooley-Tukey FFT and Grigoryan FFT respectively, it is clear that the Grigoryan FFT is going faster than the Cooley-Tukey FFT.
5. CONCLUSION AND FURTHER RESEARCH

In this paper we have shown that with our FFT processors architectures on Xilinx FPGAs and TMS DSPs the paired transform based Grigoryan FFT algorithm is faster and can be used at higher sampling rates than the Cooley-Tukey FFT at an expense of high resource utilization. It is observed that implementations for the Grigoryan FFT we are the first to design and we are the first and may be the best implementers using SIMULINK. After studying the solution of implementation method using SIMULINK we demonstrated its hardware feasibility and visual interface through Xilinx system generator. We have explored the feasibility of CC studio for TMS DSP processors for the Grigoryan FFT. We have proved that on Xilinx FPGAs and TMS DSPs, Grigoryan FFT is performing at most 1.358 and 1.7 times faster than the Cooley-Tukey FFT respectively. Which is a good improvement in speed over the Cooley-Tukey FFT. As a technological confirmation, for our implementations especially for the Grigoryan FFT (FFT of interest) out of the three FPGA platforms we conclude that Virtex-5 FPGAs is found to be better platform.

As a further research we are further extending these architectures with even best utilization of the features of DSP slices on FPGAs, by efficiently exploiting parallelism of FPGA; and explicit utilization of MAC engines on DSP processors, which can lead to most efficient implementations and the Grigoryan FFT can show even much better performance. We are developing a “neural data acquisition and processing and RF wireless transmitting” DSP processor design where we will be utilizing our Grigoryan FFT processors.

ACKNOWLEDGEMENTS

This is to acknowledge Dr. Parimal A. Patel, Dr. Artyom M. Grigoryan of The University of Texas at San Antonio for their valuable guidance for the first part of this research series as Mr. Narayanan’s Master’s Thesis. There is no order as first author among the three of us; we all are of equal most importance in this critical research. Mr. Narayanan also would like to acknowledge University of Ottawa research facility providers for providing essential tools while he was a biomedical engineering research student.

REFERENCES

[10] The scientist and engineer’s guide to Digital Signal Processing, Dr. Steven W. Smith.
Applications in Industry and Engineering, CAINF-2004, Orlando, Florida USA, Nov. 17-19, 2004


[17] TMS320C5416, TMS320C6748, TMS320C5515 manuals:www.ti.com


Author's Biographies

Mr. Ranganadh Narayanan is an associate professor in the department of Electronics & Communications Engineering in Aurora’s Scientific & Technological Institute( ASTI). Mr. Narayanan was a research student in the area of “Brain Stem Speech Evoked Potentials” under the guidance of Dr. Hilmi Dajani 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 has done some advanced learning in the areas of DNA computing, String theory and Unification of forces, Faster than the speed of light theory with worldwide reputed persons and world’s top ranked universities. Mr. Narayanam’s research interests include neurological Signal & Image processing, DSP software & Hardware design and implementations, neurotechnologies. Mr. Narayanan can be contacted at rmar100@gmail.com, mara100@uottawa.ca , ranganadh.narayanam@gmail.com

Dr. Artyom M. Grigoryan received the MS degrees in mathematics from Yerevan State University (YSU), Armenia, USSR, in 1978, in imaging science from Moscow Institute of Physics and Technology, USSR, in 1980, and in electrical engineering from Texas A&M University, USA, in 1999, and Ph.D. degree in mathematics and physics from YUS, in 1990. In 1990-1996, he was a senior researcher with the Department of Signal and Image Processing at Institute for Problems of Informatics and Automation, and Yerevan State University, National Academy Science of Armenia. In 1996-2000 he was a Research Engineer with the Department of Electrical Engineering, Texas A&M University. In December 2000, he joined the Department of Electrical Engineering, University of Texas at San Antonio, where he is currently an Associate Professor. He holds two patents for developing an algorithm of automated 3-D fluorescent in situ hybridization spot counting and one patent for fast calculating the cyclic convolution. He is the author of three books, three book-chapters, two patents, and many journal papers and specializing in the theory and application of fast one- and multi-dimensional Fourier transforms, elliptic Fourier transforms, tensor and paired transforms, unitary heap transforms, design of robust linear and nonlinear filters, image enhancement, encoding, computerized 2-D and 3-D tomography, processing biomedical images, and image cryptography.

Dr. Parimal A. Patel did his doctoral studies in UT Austin. He was a professor and Chair of the department of Electrical and Computer Engineering in UT San Antonio. He has been a Xilinx, California Consultant while being as the chair at UT San Antonio. He is currently a key Global Trainer & Consultant of Xilinx, California.

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