International Journal of Computer Applications (0975 – 8887)
Volume 7– No.9, October 2010

Feasibility Analysis and Comparative study of FFT & Autocorrelation Algorithms

Abhishek Shukla
Department of Embedded System Design,
International Institute of Information Technology,
Pune

Suraj S. Jibhakate
Department of Embedded System Design,
International Institute of Information Technology,
Pune

ABSTRACT

FFT is one main property in any sequence being used in general. To find this property of FFT for any given sequence, many transforms are being used. The major issues to be noticed in finding this property are the time and memory management. Two different algorithms are written for calculating FFT and Autocorrelation of any given sequence. Comparison is done between the two algorithms with respect to the memory and time managements and the better one is pointed. Comparison is between the two algorithms written, considering the time and memory as the only main constraints. Time taken by the two transforms in finding the fundamental frequency is taken. At the same time the memory consumed while using the two algorithms is also checked. Based on these aspects it is decided which algorithm is to be used for better results.

General Terms

Transformation, Algorithms, Feasibility analysis, Comparative study

Keywords

FFT (fast fourier transform), Autocorrelation, MATLAB, C platform, Time management, Memory management.

1. INTRODUCTION

FFT is one main property in any sequence being used in general. To find this property of FFT for any given sequence, many transforms are being used. The major issues to be noticed in finding this property are the time and memory management. The main application of the FFT is in Analysazation of Umbilical Artery Doppler Signals to Fuzzy Algorithm, correlation analysis of near field microscopy measurements, synthesis of non-uniformly spaced arrays. In 1965 there was a remarkable breakthrough when a technique called the Fast Fourier Transform or FFT was invented to evaluate the Fourier Transform. This was *much* faster, taking a time proportional to $N \log(N)$ which is a lot smaller than $u(t)$. With the FFT available to calculate the Fourier transform quickly, its applications became almost unlimited

The autocorrelation function of a random process is defined as the average value of the product of the value of the signal sampled at time t1 and the value of the signal sampled at time t2 (where t2 = t1+dt and dt is an increment of time). Therefore, for a process described by some value x (where x could be acceleration or displacement etc.) varying with time t the autocorrelation function is the average of x(t)*x(t+dt) for many values of t and many values of dt.

The main application of the autocorrelation is analysis for interpreting acoustic emission in rock, for the interpretation of intestinal motility records, structural lineaments in radioactive sample elements, X-ray diffractionists to help recover the "Fourier phase information" on atom positions etc.

Comparison is between the two algorithms written, considering the time and memory as the only main constraints. Time taken by the two transforms in finding the periodicity is taken. At the same time the memory consumed while using the two algorithms is also checked. Based on these aspects it is decided which algorithm is to be used for better results.

In this paper we have given the description of FFT and AUTOCORRELATION alogorithm in section 2. Section 3. consist about the platform used. Section 4 about the implementation.Section 5 about the result. Section 6 consist of result.

2. OVERVIEW OF FFT & AUTOCORRELATION ALGORITHM

There are several methods to find fundamental period of audio signals. E.g. Fast Fourier Transform, Autocorrelation, FIR filter method of periodic detection, comb transformation, average magnitude differential function, zero-crossing detection.

The audio signal samples may be taken from Mp3 player, Generator (small), Generator (large), Laptop, AC Machine.

These sources are considered so as to obtain different real time frequencies. These frequencies are used in the algorithm for testing purpose.

2.1 Fast Fourier Transform (FFT):

The Fast Fourier Transform (FFT) is simply a fast (computationally efficient) way to calculate the Discrete Fourier Transform (DFT).

By making use of periodicities in the sines that are multiplied to do the transforms, the FFT greatly reduces the amount of calculation required. Here's a little overview.

Functionally, the FFT decomposes the set of data to be transformed into a series of smaller data sets to be transformed. Then, it decomposes those smaller sets into even smaller sets. At each stage of processing, the results of the previous stage are combined in special way. Finally, it calculates the DFT of each small data set. For example, an FFT of size 32 is broken into 2 FFT's of size 16, which are broken into 4 FFT's of size 8, which are broken into 8 FFT's of size 4, which are broken into 16 FFT's of size 2 for radix 2.

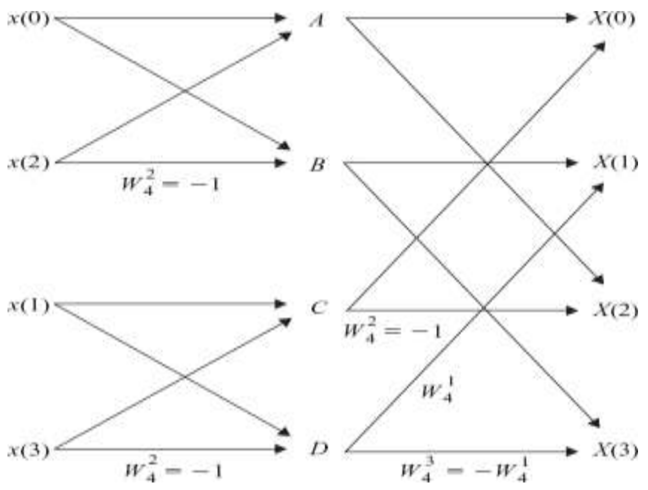


Fig 2.1 FFT Butterfly signal flow diagram, showing the example having N= 4.

Here's a slightly more rigorous explanation: It turns out that it is possible to take the DFT of the first N/2 points and combine them in a special way with the DFT of the second N/2 points to produce a single N-point DFT. Each of these N/2-point DFTs can be calculated using smaller DFTs in the same way. One (radix-2) FFT begins, therefore, by calculating N/2 2-point DFTs. These are combined to form N/4 4-point DFTs. The next stage produces N/8 8-point DFTs, and so on, until a single N-point DFT is produced.

The DFT takes N^2 operations for N points. Since at any stage the computation required to combine smaller DFTs into larger DFTs is proportional to N, and there are log $_2$(N).

N	10	100	1000	10^6
N2	100	10,000	10^6	10^{12}
$N\log_{10}N$	10	200	3000	$6 * 10^6$

The total computation is proportional to N * log2 (N). Therefore, the ratio between a DFT computation and an FFT computation for the same N is proportional to N / log2 (n). In cases where N is small this ratio is not very significant, but when N becomes large, this ratio gets very large. (Every time you double N, the numerator doubles, but the denominator only increases by 1.)

The "radix" is the size of FFT decomposition. In the example above, the radix was 2. For single-radix FFT's, the transform size must be a power of the radix. In the example above, the size was 32, which is 2 to the 5th power.

"Twiddle factors" are the coefficients used to combine results from a previous stage to form inputs to the next stage. An "in place" FFT is simply an FFT that is calculated entirely inside its original sample memory. In other words, calculating an "in place" FFT does not require additional buffer memory (as some FFT's do.)

2.2 Autocorrelation:

Autocorrelation is the cross-correlation of a signal with itself.. It is a mathematical tool for finding repeating patterns, such as the presence of a periodic signal which has been buried under noise, or identifying the missing fundamental frequency in a signal implied by its harmonic frequencies. It is often used in signal processing for analyzing functions or series of values, such as time domain signals.

The definition of the autocorrelation between any two time *s* and *t* is:

$$\mathbf{R(s,t) = (E[(X_t - \mu_t)(X_s - \mu_s)])/(\sigma_t \sigma_s)}$$

The following plot showing 100 random numbers with a "hidden" sine function, and an autocorrelation (correlogram) of the series on the bottom

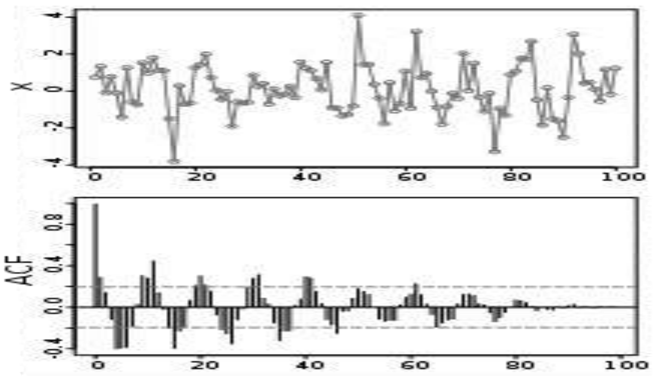


Fig 2.2 Graph between random numbers (0-100) and ACF

If X_t is a second-order stationary process then the mean μ and the variance σ^2 are time-independent, and further the autocorrelation depends only on the difference between *t* and *s*: the correlation depends only on the time-distance between the pair of values but not on their position in time. This further implies that the autocorrelation can be expressed as a function of the time-lag, and that this would be an even function of the lag $\tau = t - s$. This gives the more familiar form

$$\mathbf{R_{(\tau)} = E[(X_t-\mu)(X_{t+\tau}-\mu)] / \sigma^2}$$

2.3 Difference Between FFT & Autocorrelation

The difference between FFT and Autocorrelation is shown in below table:

Sr. No.	FFT	Autocorrelation
1	It's frequency domain analysis	It's a time domain analysis
2	Buffer size depends on desired frequency accuracy. Buffer size increases with higher accuracy.	Buffer size depends on previous fundamental frequency and changes with same .
3	Fixed length buffer is used to achieve frequency accuracy of 0.5 Hz	Varying length buffer is used to compute fundamental period
4	Usually large size of buffer is needed.	Buffer size is comparatively small.
5	Less accurate than autocorrelation in computing fundamental frequency.	More accurate than FFT in computing fundamental frequency.
6	FFT has less number of computations and so it is faster.	Autocorrelation FFT has more number of computations and hence it is computationally slower

3. PLATFORM USED

The platform used for this work is MATLAB & DEV C++.

Audio file recording is also done using Matlab function which records the sound at defined sampling rate (e.g. 8000 samples /second). Matlab has functions such as read, write, play, record .wav files. Plotting of various graphs and evaluating them is very easy using Matlab. MATLAB is a numerical computing environment and fourth-generation programming language. Developed by The MathWorks, MATLAB allows matrix manipulation, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs in other languages. Although it is numeric only, an optional toolbox uses the MuPAD symbolic engine, allowing access to computer algebra capabilities. An additional package, Simulink, adds graphical multidomain simulation and Model-Based Design for dynamic and embedded systems.

MATLAB ("Matrix Laboratory") was created in the late 1970s by Cleve Moler, then chairman of the computer science department at the University of New Mexico. He designed it to give his students access to LINPACK and EISPACK without having to learn Fortran. It soon spread to other universities and found a strong audience within the applied mathematics community. Jack Little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded The MathWorks in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK.

DEV C++ is a free integrated development environment (IDE) distributed under the GNU General Public License for programming in C and C++. It is bundled with MinGW, a free compiler. The IDE is written in Delphi.

The project is hosted by Source Forge. Dev-C++ was originally developed by programmer Colin Laplace. Dev-C++ runs exclusively on Microsoft Windows.

Bloodshed Dev-C++ is a full-featured Integrated Development Environment (IDE) for the C and C++ programming languages. It uses the MinGW port of the GCC (GNU Compiler Collection) as its compiler. Dev-C++ can also be used in combination with Cygwin or any other GCC-based compiler.[1]

One additional aspect of Dev-C++ is its use of DevPaks, packaged extensions on the programming environment with additional libraries, templates, and utilities. DevPaks often contain, but are not limited to, GUI utilities, including popular toolkits such as GTK+, wxWidgets, and FLTK. Other DevPaks include libraries for more advanced function use.Dev-C++ is generally considered a Windows-only program. There is also a Linux version available, but it is in alpha and has not been updated since July 2002.

4. IMPLEMENTATION

This module computes fundamental frequency of the input audio signal with frequency resolution of 0.5 Hz. Matlab function fft ( ) is used to compute FFT of audio signal which implements Radix 2 algorithm. Output spectrum contains all frequency components and their harmonics along with noise. Out of all these frequency components, identification of fundamental period is done using an algorithm which is discussed in section 4.3

Buffer size depends on the frequency resolution required.

Buffer size = Sampling frequency/ accuracy

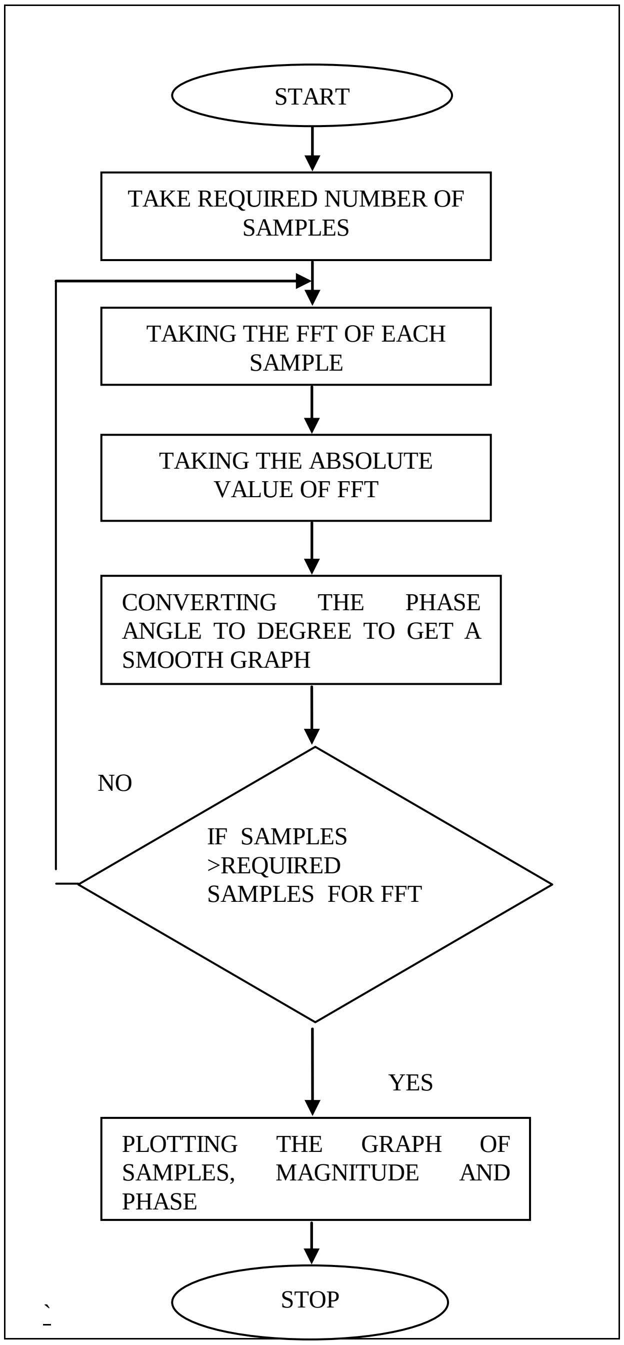
Input audio signal has sampling frequency = 8000 samples/sec and we compute the FFT with frequency resolution 0.5 Hz .Buffer size required = 8000/ 0.5 = 16,000

FFT implements radix-2 algorithm which requires samples in power of 2. So total number of samples required to achieve resolution of 0.5 Hz would be = 2 ^14 = 16384 samples

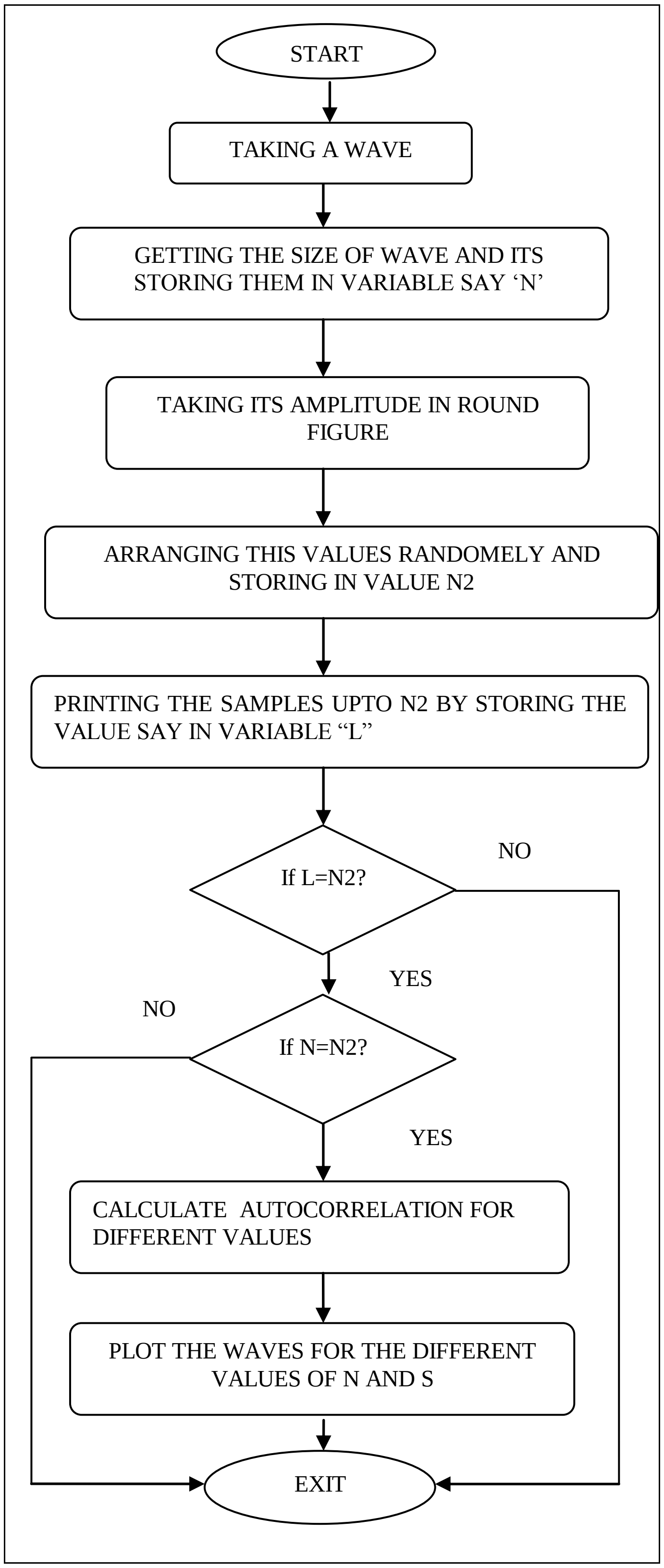
4.1 FLOWCHART

The step wise implementation undergone flowchart is as follows:

4.1.1 Flowchart for Magnitude & Phase


START
TAKE REQUIRED NUMBER OF SAMPLES
TAKING THE FFT OF EACH SAMPLE
TAKING THE ABSOLUTE VALUE OF FFT
CONVERTING THE PHASE ANGLE TO DEGREE TO GET A SMOOTH GRAPH
NO
IF SAMPLES >REQUIRED SAMPLES FOR FFT
YES
PLOTTING THE GRAPH OF SAMPLES, MAGNITUDE AND PHASE
STOP

4.1.2 FLOWCHART FOR IMPLEMENTATION OF AUTOCORELATION


START
TAKING A WAVE
GETTING THE SIZE OF WAVE AND ITS STORING THEM IN VARIABLE SAY 'N'
TAKING ITS AMPLITUDE IN ROUND FIGURE
ARRANGING THIS VALUES RANDOMELY AND STORING IN VALUE N2
PRINTING THE SAMPLES UPTO N2 BY STORING THE VALUE SAY IN VARIABLE "L"
If L=N2?
NO
YES
NO
If N=N2?
YES
CALCULATE AUTOCORRELATION FOR DIFFERENT VALUES
PLOT THE WAVES FOR THE DIFFERENT VALUES OF N AND S
EXIT

5. RESULT:

The graphical implementation from a given signal is shown below:

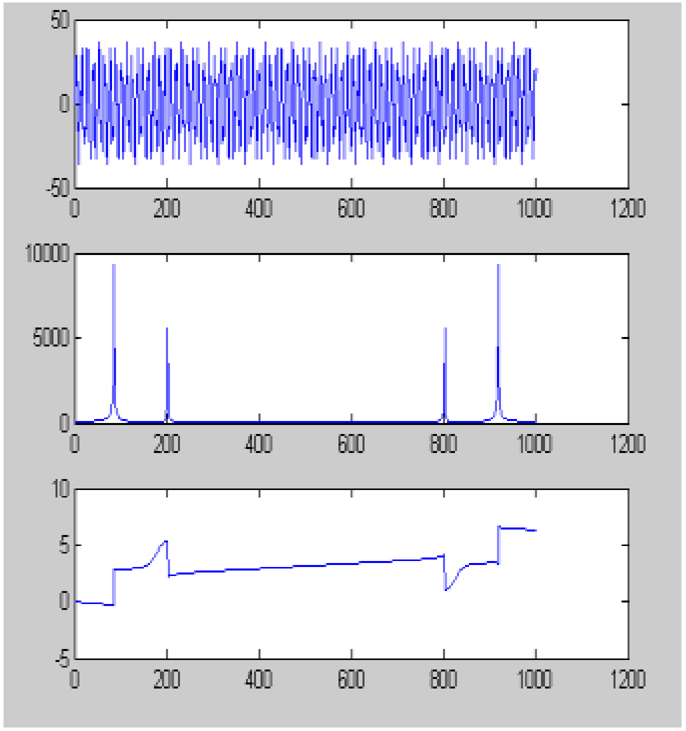


Fig 5.1 magnitude and phase plot for given input signal From the above figure the input signal is taken and its magnitude and phase is plotted.

5.2. IMPLMENTATION OF AUTOCORRELATION

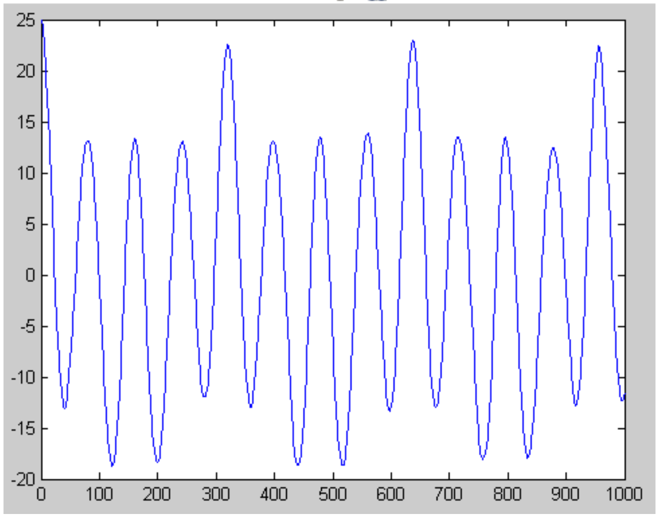


Fig 5.2 shows the implementation of autocorrelation using MATLAB

6. CONCLUSION

In this paper for performing feasibility analysis, the FFT and Autocorrelation algorithm has been implemented on two different platforms i.e. MATLAB & C. by plotting magnitude and phase graphs for any given input signal and performed both FFT & autocorrelation algorithms for different input signals & finally tried to found out the maximum peaks in the signal i.e. Fundamental frequencies

Thus after comparing two algorithms i.e. FFT and Auto Correlation regarding the time constraints and memory management the FFT algorithm can have maximum number of samples in minimum time.

7. REFERENCE

1. [paper] matlab implementation of an fft based algorithm for polynomial plus/minus factorization by martin hromcık, michael sebek
2. [paper] systematic generation of fpga-based fft implementations *hojin kee, newton petersen, jacob kornerup, shuvra s. bhattacharyya.*
3. [paper] a modified split-radix fft with fewer arithmetic operations by steven g. johnson* and matteo frigo.
4. [paper] reducing memory references for fft calculation by mokhtar a. aboleaze, ayman i. elnaggar
5. web resource: [online visited on aug 18,2010]: Pitch detection using time and freq. domain method

   [http://cnx.org/content/m11714/latest].
6. IEEE Oceanic Engineering Societyby ,MacInnes, C.S. Alameda El Espinel, Lima
7. hojin kee, newton petersen, jacob kornerup, shuvra s. bhattacharyya," systematic generation of fpga-based fft implementations", in proceedings of the international conference on acoustics, speech, and signal processing,las vegas, nevada, march 2008.
8. josé moura, carnegie mellon university" dsp formatlab™ and labview™ volume ii: discrete frequencytransforms", foresterw. isen 2008
9. web resource: [online visited on aug 28,2010] www.math.tamu.edu/~kfu/chapter10.pptx.
10. web resource: [online source visited on sept 07,2010] jorg arndt ,"algorithms for programmers"; this document is online at http://www.jjj.de/fxt/.