Design of FIR Filter using Genetic Algorithm

Samrat Banerjee, Sriparna Dey, Supriya Dhabal

Abstract— **Digital filters constitute an essential part of DSP. Actually, their extraordinary performance is one of the main reasons which have made DSP so popular. The purpose of the filters is to allow some frequencies to pass unchanged, while completely blocking others. The digital filters are mainly used for two purposes: Separation of signals that have been combined, and restoration of signals that have been distorted in some way. Here, FIR filter is designed using Genetic Algorithm (GA) in MATLAB. The response is studied by keeping values of fixed order, crossover probability and mutation probability.GA offers a quick, simple and automatic method of designing low pass FIR filters that are very close to optimum in terms of magnitude response, frequency response and in terms of phase variation. The number of operations in the design process is reduced and coefficient calculation is easily realized with the help of GA.**

Index Terms— **Digital Filters, FIR, GA, DSP.**

I. INTRODUCTION

Digital Signal Processing (DSP) is one of the most powerful technologies that are shaping science and engineering in this century. Revolutionary changes have already been made in a broad range of fields: communications, medical imaging, Radar and Sonar, and high fidelity music reproduction, to name just a few. Each of these areas has developed a comprehensive DSP technology, with its own algorithms, mathematics, and specialized techniques. Analog (electronic) filters can be used for these tasks, as these are cheap, fast, and have a large dynamic range in both amplitude and frequency; however, digital filters are vastly superior in the level of performance. In this work, a type of digital filter i.e., FIR filter is used to separate one band of frequencies from another. The primary attribute of FIR filters is their stability. This is because they are carried out by convolution rather than recursion. FIR filters are linear phase filters and both phase delay and group delays are constant in these filters [1].

II. FILTER AND DESIGN TECHNIQUES

Filtering is a process by which the frequency spectrum of a signal can be modified, reshaped or manipulated to achieve some desired objectives. These are as under [2]:
To eliminate noise that may contaminate the signal, to remove signal distortion which may be due to imperfection in the transmission channel, to resolve the signal into its frequency component, to demodulate the signal which was modulated at the transmitter side, to convert digital signals into analog signals and to limit the bandwidth of a signal.

Manuscript received June 04, 2015

Samrat Banerjee, Department of E.C.E, Netaji Subhash Engineering College, Kolkata (West Bengal University of Technology, Kolkata), West Bengal, India, Mobile No.: 9804758232, 9883719891
Sriparna Dey, Department of E.C.E, Netaji Subhash Engineering College, Kolkata (West Bengal University of Technology, Kolkata), West Bengal, India, Mobile No.: 8013878757, 8132800603
Supriya Dhabal, Associate Professor, Department of E.C.E, Netaji Subhash Engineering College, Kolkata (West Bengal University of Technology, Kolkata), West Bengal, India.

A. Finite Impulse Response (FIR) Filter

A Finite Impulse Response (FIR) digital filter is one whose impulse response is of finite duration [3]. The impulse response is "finite" because there is no feedback in the filter. If we put in an impulse (that is, a single "1" sample followed by many "0" samples), zeroes will eventually come out after the "1" sample has made its way in the delay line past all the coefficients. FIR (Finite Impulse Response) filters are implemented using a finite number "n" delay taps on a delay line and "n" computation coefficients to compute the algorithm (filter) function. The above structure is non-recursive, a repetitive delay-and-add format, and is most often used to produce FIR filters. This structure depends upon each sample of new and present value data. The number of taps (delays) and values of the computation coefficients ($h_0, h_1, \dots h_n$) are selected to "weight" the data being shifted down the delay line to create the desired amplitude response of the filter. In this configuration, there are no feedback paths to cause instability. The calculation of coefficients is not constrained to particular values and can be used to implement filter functions that do not have a linear system equivalent. More taps increase the steepness of the filter roll-off while increasing calculation time (delay) and for high order filters, limiting bandwidth. This can be stated mathematically as:

$$y(n)=\sum_0^{N-1} h(k)\,x(n-k) \qquad .....(1)$$

where, y(n) = Response of Linear Time Invariant (LTI) system.
x(k) = Input signal
h(k) = Unit sample response
N = No. of signal samples

FIR filters are simple to design and they are guaranteed to be Bounded Input-Bounded Output (BIBO) stable. By designing the filter taps to be symmetrical about the centre tap position, an FIR filter can be guaranteed to have linear phase response. This is a desirable property for many applications such as music and video processing.

B. Infinite Impulse Response (IIR) Filter

IIR filter is one whose impulse response is infinite [3]. Impulse response is infinite because there is feedback in the filter. This permits the approximation of many waveforms or transfer functions that can be expressed as an infinite recursive series. These implementations are referred to as Infinite Impulse Response (IIR) filters. The functions are infinite recursive because they use previously calculated values in future calculations to feedback in hardware systems. IIR filters can be mathematically represented as:

$$y(n)= \sum_0^{M-1} a_k y(n-k) + \sum_0^{N-1} h_k x(n-k) \quad ...(2)$$

Where a_k is the k^{th} feedback tap. M is the number of feed-back taps in the IIR filter and N is the number of feed-forward taps. IIR Filters are useful for high-speed designs because they typically require a lower number of multiply compared to FIR filters. IIR filters have lower side lobes in stop band as compared to FIR filters. Unfortunately, IIR filters do not have linear phase and they can be unstable if not designed properly. IIR filters are very sensitive to filter coefficient quantization errors that occur due to use of a finite number of bits to represent the filter coefficients. One way to reduce this sensitivity is to use a cascaded design. That is, the IIR filter is implemented as a series of lower-order IIR filters as opposed to one high-order.

C. *Advantages of FIR over IIR Filter*

FIR filters have the following advantages over the IIR filters:

1. FIR filters are linear phase filters, which is useful in speech processing.
2. FIR filters are always stable because all the poles are within the unit circle.
3. The designing methods are generally linear for FIR filters.
4. The start-up transitions have finite duration in FIR.
5. Round off noise can be made small by employing non-recursive technique of realization.

III. GENETIC ALGORITHM (GA)

Genetic algorithms are search algorithms based on the mechanics of natural selection and genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old; an occasional new part is tried for good measure.
Genetic algorithms have been developed by John Holland, his colleagues, and his students at the University of Michigan. The goals of their research were [4]:

1.) To abstract and rigorously explain the adaptive processes of natural systems, and
2.) To design artificial system software that retains the important mechanisms of natural systems.

In order for genetic algorithm to surpass their more traditional cousins to surpass in the quest for robustness, GA's must differ in some very fundamental ways. Genetic algorithms are different from more normal optimization and search procedures in four ways:

1.) GA's work with a coding of the parameter set, not the parameters themselves.
2.) GA's search from a population of points, not a single point.
3.) GA's use payoff (objective function) information, not derivatives or other auxiliary knowledge.
4.) GA's use probabilistic transition rules, not deterministic rules.

A. *Initialization*

In the initialization, the first thing to do is to decide the coding structure. Coding for a solution, termed a chromosome in GA, is usually described as a string of symbols from (0,1). These components of the chromosomes are then labelled as genes [5].
Figure shows a standard procedure of a Canonical Genetic algorithm.

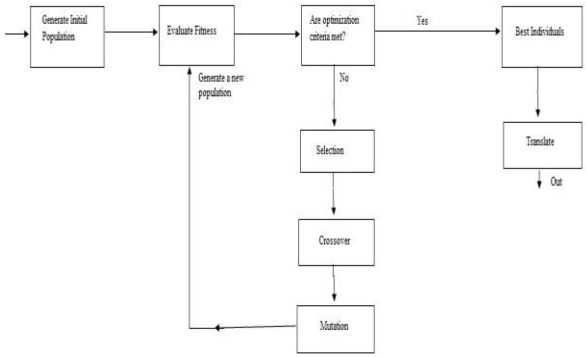

Fig.1: Standard procedure of Canonical Genetic Algorithm

B. *Crossover*

Crossover is an important random operator in CGA and the function of crossover operator is to generate new or child chromosomes by combining the information extracted from the parents. By one from two parent point crossover method, for a chromosome of length, l, a random number c between 1 and l is first generated. The first child chromosome is formed by appending the last l-c elements of the first parent chromosome to the first c elements of the second parent chromosome. The second child chromosome is formed by appending the last l-c elements of the second parent chromosome to the first c elements of the first parent chromosome. Probability of crossover ranges from 0.6 to 0.95.

C. *Mutation*

Mutation is another important operator in CGA, though it is usually considered as a background operator. It operates independently on each individual by probabilistic perturbing each bit string. A usual way to mutate used in CGA is to generate a random number v between 1 and l and then make a random change in the vth element of the string with probability p_m ε (0, 1)

Typically, the probability for bit mutation changes from 0.001 to 0.01

Parent#1 1 0 0 1 | 0 1 0 01 0 → Child#1 0 1 0 1 0 1 0 0 1 0

Parent#2 0 1 0 1 | 1 1 1 0 1 1 Child#2 1 0 0 1 1 1 1 01 1

Parent 1 1 0 1 0 1 0 0 1 0 → Child 1 1 0 1 0 1 0 1 1 0

Fig.2: One-point Crossover and Mutation operators

IV. DESIGNING TECHNIQUES OF FIR FILTERS

There are essentially three well-known methods for FIR filter design namely:

(1) The window method
(2) The frequency sampling technique
(3) Optimal filter design methods

A. Optimal Design of FIR Filter using Genetic Algorithm

The genetic algorithm loops over an iteration process to make the population evolve [6]. It consist the following steps:

1.) Selection: The first step consists in selecting individuals for reproduction. This selection is done randomly with a probability depending on the relative fitness of the individuals so that best ones are often chosen for reproduction than poor ones.
2.) Reproduction: In the second step, offspring are bred by the selected individuals. For generating new chromosomes, the algorithm can use both recombination and mutation.
3.) Evaluation: Then the fitness of the new chromosomes is evaluated.
4.) Replacement: During the last step, individuals from the old population are killed and replaced by the new ones. The algorithm is stopped when the population converges towards the optimal solution.

B. Application of Genetic Algorithm to FIR Filter Design

A digital FIR filter is characterized by the following transfer function,

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n} \qquad ...(3)$$

In the above expression, N is the order of the filter and *h(n)* represent the filter coefficients to be determined in the design process. Designing the FIR filters as minimum phase provides some important advantages. Minimum phase filters have two main advantages: Reduced filter length and Minimum group delay. Minimum phase filters can simultaneously meet delay and magnitude response constraints yet generally require fewer computations and less memory than linear phase. Recently, GA has been emerged into optimum filter designs. The characteristics of multi-objective, coded variables, and natural selection make GA different from other optimization techniques. Filters designed by GA have the potential of obtaining near global optimum solution [7].

FIR digital filter has a finite number of nonzero entries of its impulse response such as h[n], n=0,1,…,N. Generally assume implicitly that $h[n] \neq 0$, $h[0] \neq 0$.
The transfer function of the FIR filter is given in eq. (3) and the frequency response of form is:

$$H(e^{j\omega}) = \sum_{n=0}^{N-1} h(n) e^{-j\omega n} \qquad ... (4)$$

Consider the ideal frequency response $H_d(e^{j\omega})$ with the samples divided into equal frequency interval, Thus we can get,

$$H_d(e^{j\omega})|_{\omega = 2\pi k/N} = H_d(k) \qquad ...(5)$$

where, $H_d(k)$ is regarded as the frequency response of the filter to design. Equation (5) can be rewritten as

$$H_d(k) = H_d(e^{j\omega})|_{\omega = 2\pi k/N}, \ k=0,1,.............N-1 \qquad ...(6)$$

To design a linear phase FIR filter, we must minimize the error between actual and ideal output. There exist some forms of error function for the filter design. One of them is the least-squares method. We define the error function as the error between the desired magnitude and the actual amplitude at a certain frequency, that is

$$E(e^{j\omega}) = H_d(e^{j\omega}) - H(e^{j\omega}) \qquad ...(7)$$

Thus we can adopt the objective function for the minimization as total squared error across frequency domains as follows

$$E(e^{j\omega}) = \sum_{i=1}^{M} [\, |H_d(e^{j\omega_i})| - |H(e^{j\omega_i})| \,]^2 \qquad ....(8)$$

where, *M* is the number of frequency interval. From eq. (4) we can write the above equation as:

$$E(e^{j\omega}) = \sum_{i=1}^{M} [\, |H_d(e^{j\omega_i})| - |\sum_{n=0}^{N-1} h(n) e^{-j\omega_i n}| \,]^2 \qquad ...(9)$$

The problem is reduced to find out *h (n)* by minimizing the squared error *E*.

C. Coefficient Encoding

The filter impulse response coefficients, h (0) to h(N), are sufficient to represent a digital FIR filter. Thus, N+1 coefficients of the filter form the genome and the particle position in the GA and the PSO, respectively. Each coefficient is represented by a floating number in the range [-1, 1], inclusive.

D. Fitness Function

We use the total squared error as the fitness function of FIR digital filter, that is

$$E(e^{j\omega}) = \sum_{i=1}^{M} [\, |H_d(e^{j\omega_i})| - |\sum_{n=0}^{N-1} h(n) e^{-j\omega_{in}}| \,]^2 \qquad ... (10)$$

V. PROPOSED TECHNIQUE

The method applied through MATLAB is to design a low pass FIR filter with ideal magnitude response, zero phase and small phase variation. Consider that a low pass FIR filter is to be designed with the initial conditions described in the table:

Filter Type	Low Pass
Order of Filter	30
No. of sample Point	200
Stop Band Frequency (ω_s)	0.25
Pass Band Frequency (ω_p)	0.45
Population No.	200
Generation No.	500
Crossover Probability (P_c)	0.9
Mutation Probability (P_m)	0.1

Table1: Initial conditions for designing low pass FIR Filter

Ideal Low pass filter passes all the signals that are below the cut off frequency and stop all others. Here, there is a flat pass band below pass band frequency ($\omega_p = 0.45$) and flat attenuation band above stop band frequency. ($\omega_s = 0.25$) When we are using only two parents, we get the magnitude response versus frequency curve as shown in Fig.3, Fig.5, Fig.7, Fig.9. But, when we are using three parents, we get a better magnitude response versus frequency curve as shown in Fig.4, Fig.6, Fig.8, Fig.10.

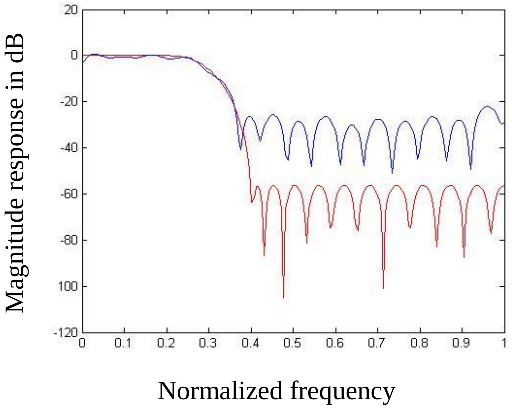

Fig.3: Magnitude Response of FIR Filter using two parents at 500 generations with 3 attempts

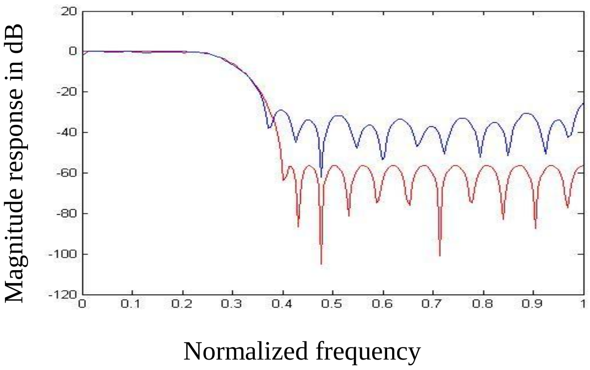

Fig.4: Magnitude Response of FIR Filter using three Parents at 500 generations with 3 attempts

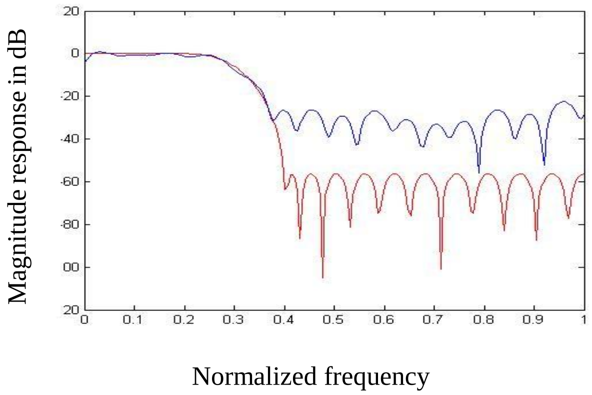

Fig.5: Magnitude Response of FIR Filter using two Parents at 500 generations with 5 attempts

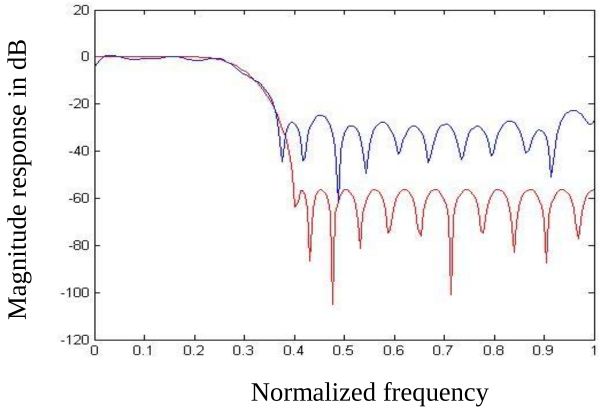

Fig.6: Magnitude Response of FIR Filter using three Parents at 500 generations with 5 attempts

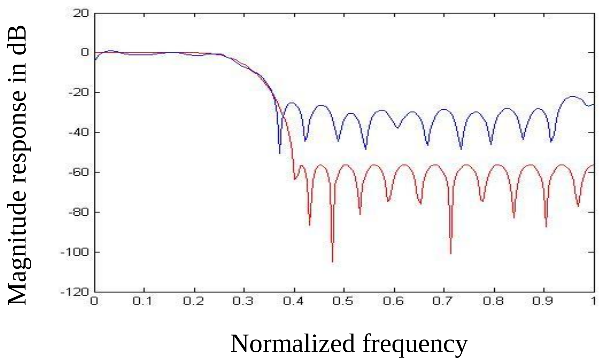

Fig.7: Magnitude Response of FIR Filter using two Parents at 400 generations with 5 attempts

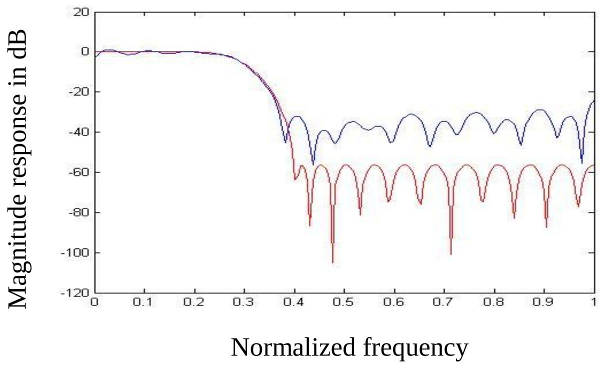

Fig.8: Magnitude Response of FIR Filter using three Parents at 400 generations with 5 attempts

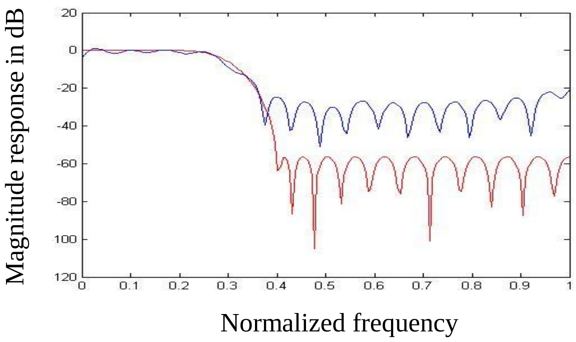

Fig.9: Magnitude Response of FIR Filter using two Parents at 600 generations with 3 attempts

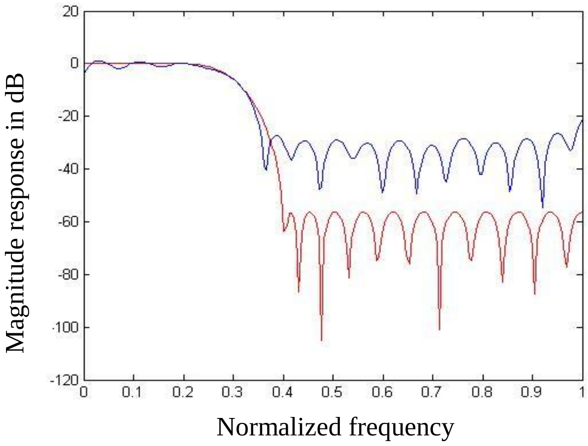

Fig.10: Magnitude Response of FIR Filter using three Parents at 600 generations with 3 attempts

VI. SIMULATION RESULTS FOR GENETIC ALGORITHM

h(0)=h(15)	0.9764
h(1)=h(16)	0.2608
h(2)=h(17)	-0.0321
h(3)=h(18)	0.3095
h(4)=h(19)	-0.0916
h(5)=h(20)	-0.4859
h(6)=h(21)	0.5461
h(7)=h(22)	-0.4989
h(8)=h(23)	0.4140
h(9)=h(24)	0.0971
h(10)=h(25)	0.3496
h(11)=h(26)	-0.0658
h(12)=h(27)	0.4673
h(13)=h(28)	0.8167
h(14)=h(29)	0.3127

Table 2: Filter coefficients of GA

GA is an optimizing method to design an FIR filter. Figure 4 shows that its magnitude response (Blue) is approximately same as ideal response (Red). But it has small amount of ripples in the pass band and very small in stop band. The transition bandwidth is near to optimal method. Best response is seen in Fig.4.

VII. CONCLUSION

In this present work, FIR filter is designed using GA in MATLAB. The response is studied by keeping values of fixed order, crossover probability and mutation probability. From the outputs obtained it is clear that GA offers a quick, simple and automatic method of designing low pass FIR filters that are very close to optimum in terms of magnitude response, frequency response and in terms of phase variation. (Here, only magnitude response has been shown). A technique of using three parents has been proposed and outputs are compared with the outputs obtained using two parents for different no. of generations taking various no. of attempts. It has been observed that a better response is achieved when three parents are used instead of two. Best response is obtained when 500 generations have been taken using 3 attempts (Fig.4). FIR filter design using Blackman window also provides good magnitude response but transition bandwidth is very high, large phase deviation and lack of control of critical frequencies ω_p and ω_s. To overcome this problem, Parks McClellan can be used. But as the order of the filter increases, this method is not suitable. Therefore, to solve all these problems, GA is used. With the help of GA, the number of operations in design process is reduced and coefficient calculation is easily realized. Here, the work has been restricted to low pass filters, it could be extended to high pass, band pass and band stop filters.

ACKNOWLEDGMENT

The authors would like to acknowledge and thank the professors from their college, especially the guide for the project, **Mr. Supriya Dhabal**, for his support and contribution. Also, the authors wish to express their gratitude to their respective parents for their support through the course of the project, and their prayers helped to complete the project in time.

REFERENCES

[1] Rabiner Lawrence R., “Techniques for Designing Finite-Duration Impulse-Response Digital Filters”, IEEE Transactions on Communication Technology, Vol. com -19, April 1971
[2] McClellan James H. and Thomas W. Parks, “A Unified Approach to the Design of Optimum FIR Linear-Phase Digital Filters”, IEEE proceeding-circuit theory, Vol. 20, pp. 697 – 701, Nov 1973.
[3] L. R. Rabiner, M.T. Dolan and J.F. Kaiser, “Some Comparisons between FIR and IIR Digital Filters”, Vol- 53, Feb.1973
[4] “Genetic Algorithms in search, optimization and Machine Learning” By David E. Goldberg.
[5] Teaching Genetic Algorithm using MATLAB, Y.J. Cao and Q.H. Wu. Int. J. Elect. Enging. Educ., Vol. 36, pp. 139–153. Manchester U.P., 1999. Printed in Great Britain
[6] Ajoy Kumar Dey, “A Method of Genetic Algorithm (GA) for FIR Filter Construction”,Vol. 1, pp. 87-90, Dec 2010
[7] A. Antoniou, Digital Filters: Analysis, Design, and Applications, McGraw-Hill, Singapore, 1993.
[8] A comparison between GA and PSO for Linear Phase FIR Filter Design, Meisam Najjarzadeh and Ahmad Ayatollahi. IEEE2008.
[9] Rabiner L.R. and B. Gold, “Theory and Application of Digital Signal Processing”, Prentice-Hall, Englewood Cliffs, NJ, 1975.
[10] S.K. Mitra, Digital Signal Processing: A Computer-based Approach, McGraw-Hill, 2001.
[11] A.V Oppenheim, R.W. Schafer, Digital Signal Processing, Prentice-Hall, 1975.

BIOGRAPHIES

Samrat Banerjee is doing his M.Tech. in Electronics & Communication Engineering, in the Department of E.C.E., from Netaji Subhash Engineering College, Kolkata (West Bengal University of Technology, Kolkata, West Bengal, India). He completed his B.Tech. in Electronics & Instrumentation Engineering from Maharana Institute of Professional Studies, Kanpur (Uttar Pradesh Technical University, Lucknow, Uttar Pradesh, India) in 2012.

Sriparna Dey is doing her M.Tech. in Electronics & Communication Engineering, in the Department of E.C.E., from Netaji Subhash Engineering College, Kolkata (West Bengal University of Technology, Kolkata, West Bengal, India). She completed her B.Tech. in Electronics & Communication Engineering from Bharathiyar College of Engineering & Technology, Karaikal (Pondicherry University, Pondicherry, India) in 2013.

Supriya Dhabal is Associate Professor in the Department of Electronics & Communication Engineering at Netaji Subhash Engineering College, Kolkata, West Bengal, India.