Performance Evaluation of Different Averaging based Filter Designs Using Digital Signal Processor and its Synthesis on FPGA

S. K. Shome, S. R. K. Vadali, U. Datta, S. Sen and A. Mukherjee

CSIR-Central Mechanical Engineering Research Institute, Durgapur, India
saikatkshome@cmeri.res.in, srk_vadali@cmeri.res.in, enc@cmeri.res.in

Abstract

Noise has been a primary deterrent in signal transmission which results in faulty information after signal processing, reducing their usability. It is important that signals should be free from unwanted, random variations so that the errors caused due to faulty representation of the original signal can be minimized. Digital filter describes a linear time invariant system used to perform spectral shaping or frequency selective filtering. Design of digital filter using averaging technique is a simple type of FIR filter without assigning any value of cut off frequency. In our work, Simple Moving Average (SMA) and Exponentially Weighted Moving Average (EWMA) based signal filtering techniques are applied on corrupted signal, having different signal to noise ratio (SNR, to filter out noise from a discretely sampled signal. Performance evaluation of these techniques has been carried out using DSP Starter Kit (DSK) having processor TMS320C6713 with Code Composer Studio and is verified through simulation using MATLAB. The algorithm of EWMA technique is synthesized on FPGA platform to estimate resource utilization and timing constraints. The motivation for this research work was derived from the quest to develop a FPGA based simple filtering technique as compared to the conventional FIR filters to reduce design cycle of the system.

Keywords: Signal Averaging, Simple Moving Average, Exponential Moving Average, Noise Variance, DSP, RMS Error, FPGA

1. Introduction

Noise is comprised of unwanted and interfering voltages and currents generated by motors in residential as well as industrial scenario. These random variations mask the required signal and do not have a strong correlation with the original one. This is due to many simultaneous processes acting at once, each adding a small component of the whole. If seen in isolation, it may appear that each noise is periodic in nature. However, the summation of many of these components in an unsynchronized fashion causes the overall noise to be random in nature. Often the combined effect of all noise sources acting together causes noise components to be comparable with the desired signal, even at times completely obscuring it. So there is a need to extract the desired signal from the noisy background.

In signal processing, the function of a filter is to remove random noise from the signal. Filtering is used in digital signal processing in a variety of ways, ranging from removal of undesired noise in desired signals, spectral shaping such as equalization for communication channels, signal detection in radar, sonar to performing spectral analysis of signals. A digital

filter uses a digital processor to perform numerical calculations on sampled values of the signal. The processor may be a general-purpose computer such as a PC or a specialized Digital Signal Processor (DSP) chip. It is used to reveal the true signal by suppressing noise, the random and persistent disturbance that obscures the true signal.

Digital filters make practical many designs that are impractical or impossible in analog filters. In practice, all DSP filters are implemented using finite-precision arithmetic, that is, a limited number of bits which sometimes make it preferable to an infinite impulse response (IIR) filter [1]. A finite impulse response filter has a number of useful properties like linear phase characteristics, design flexibility, higher operational speed and system stability which makes it widely used in digital audio assay, image processing, data transmission and bio-medical parlance. The finite impulse response (FIR) filter is generally implemented on DSP by generating coefficients with magnitude less than 1.0 which depends upon cut-off frequency, order of filtering and type of window. Shyu et al [2] have successfully designed a variable 2-D FIR digital filter using McClellan transformation with adjustable frequency characteristics. In [3], a different approach of filter implementation is presented. They use Field Programmable Gate Arrays (FPGAs) for a 16 order, constant coefficient FIR filter through VHDL. In a similar platform [4], constant shifts method and programmable shifts method based reconfigurable FIR filter architectures having low complexity have been proposed. FPGAs distinguished by being very fast and highly reconfigurable, allow for the development of scalable parallel architectures and real time analysis. As such, a computationally simple FIR filter in FPGA for real time analysis is highly desirable. Averaging technique is a simple FIR filter, providing satisfactory performance in time domain. Different types of signal averaging can be used effectively to separate a repetitive signal from noise without distorting the former using digital signal processing. If the noise is random with zero mean and is uncorrelated with the signal, averaging will improve the SNR. Since the averaging is done on a point-by-point basis, more the number of repetitions or sweeps that are averaged together, the better will be the final true noise free wave form [5]. However, the load on a process computer can become quite significant when we consider the fact that hundreds, and perhaps even thousands, of measurements are made on a typical process. An improvement in computational efficiency can be achieved if we perform the calculation of the mean in a recursive fashion. This is done by moving average technique [6]. It places equal emphasis on all data points. Thus a value in the past will have the same influence as the current measurement when calculating the filtered signal. This may be a desirable feature when the mean value of the measurement is almost constant. In dynamic systems with high level of noise, the most current values do not tend to reflect better the state of the process. A filter that places less emphasis on the most recent data would therefore be more useful. The Exponentially Weighted Moving Average filter seems to perform better in this situation [7].

Implementation of digital filter on digital signal processor has its own advantages like programmability as well as some disadvantages like delay, depending upon the speed of the processor. Application Specific Integrated Circuits (ASICs) [8] have been used traditionally. However, they imply a limited functionality due to their predefined architecture. Reconfigurable computing architectures [9] are sufficiently flexible so that new operations

can be implemented on the existing hardware besides being quite quick for real-time execution. Moreover, the price/performance ratio of these systems makes them a broadly competitive alternative to ASICs. FPGAs [10] have been identified as the natural platform for Custom Computing Machines (CCMs) [11] due to their re-programmability [12].

In this research, the performance of different averaging techniques of digital filtering has been investigated. Contributions of the present paper are as follows:

1. Development of algorithm for the generation of the following using DSP (Code Composer Studio):

   - Noise having normal distribution.

   - Sinusoidal signal equivalent to uncorrupted signal.

   - Corrupted signals having different SNRs.

2. Extraction of original uncorrupted signal by means of different techniques of digital filtering using DSP.

3. Simulations in MATLAB platform.

4. Synthesis of EWMA filter in FPGA

5. Quantitative measurements of statistical parameters of extracted signal and performance comparison of different filtering techniques

2. System Model

2.1. Generation of Random Noise having Gaussian Distribution

Random noise having Gaussian distribution is generated from Uniform distribution. Eq. 1 describes the algorithm for generating uniformly distributed random numbers between zero and one [13]

$$U_i = (aU_{i-1} + b) \ modulo \ c \tag{1}$$

where,

U_i : Random number generated

U_0 : Seed of the algorithm, an arbitrary constant.

a, b, and c   : Appropriately chosen constants.

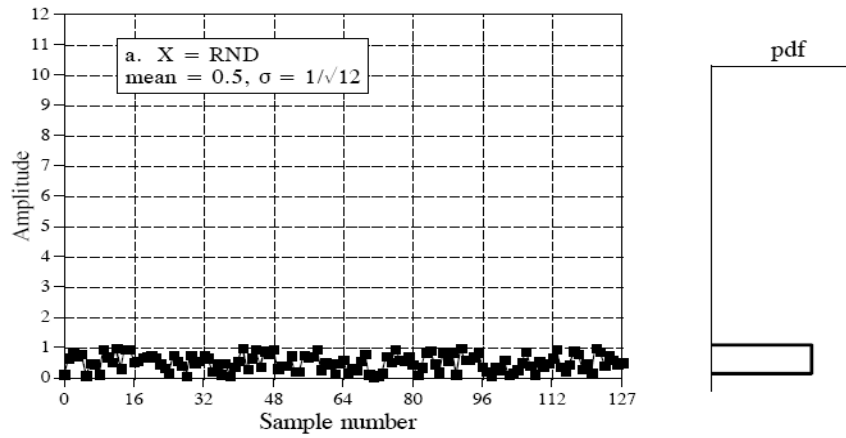The nature of the noisy signal and its pdf are shown in Figure 1.

Figure 1. Noise having Uniform Distribution

Algorithm for generating normally distributed random numbers G'_i (with non zero mean) based on Central Limit Theorem [14], from a uniformly distributed random number sequence is given by

$$G'_i = \sum_{j=1}^{p} U_{ij} \tag{2}$$

where, U_{ij}: uniformly distributed random number sequence.

The nature of the noisy signal and its pdf are shown in Figure 2.

For practical implementation, the value of "p" is considered to be twelve. The resulting sequence of (2) having mean six is then shifted to generate random sequence having Gaussian distribution G_i with zero mean and is obtained from (2) by
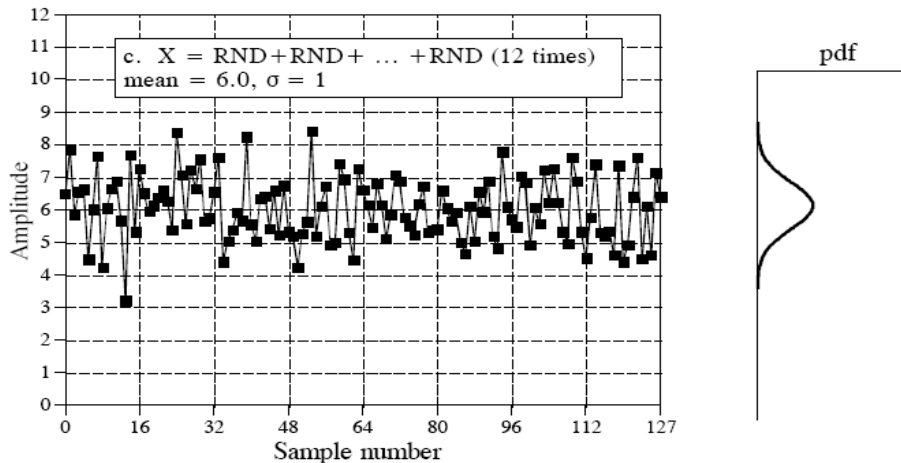
$$G_i = G'_i - 6 \tag{3}$$

Figure 2. Noise having Gaussian Distribution

2.2. Generation of Corrupted Sinusoidal Signal with Predetermined SNR using Gaussian Noise

RMS value of Gaussian noise is

$$G_{rms} = \sqrt{\sum_{i=0}^{n} G_i{}^2} \tag{4}$$

where,

n : Total number of noise samples

i : Sample number or index.

For signal to noise ratio of 'SNR' dB, RMS value of original signal 'S' is

$$S_{rms} = G_{rms} 10^{\left(\frac{SNR}{20}\right)} \tag{5}$$

Considering sinusoidal signal as the uncorrupted signal, to maintain the above mentioned SNR, the peak value (amplitude) of the sinusoidal signal is given by:

$$S_{PK(sin)} = 1.414 \left(\sqrt{\frac{1}{n}\sum_{i=0}^{n} G_i{}^2} \right) 10^{\left(\frac{SNR}{20}\right)} \tag{6}$$

Thus, samples of the sinusoidal signal are given by the equation

$$S_i = 1.414 \left(\sqrt{\frac{1}{n}\sum_{i=0}^{n} G_i{}^2} \right) 10^{\left(\frac{SNR}{20}\right)} sin\left(\frac{2\pi i}{N}\right) \tag{7}$$

where,

S_i : Sample value of pure sinusoidal signal for a desired SNR.

N : Number of samples per cycle

Corrupted signal y_i is expressed by the equation below:

$$y_i = S_i + G_i \tag{8}$$

Samples of corrupted sinusoidal signal with specified SNR are obtained by adding signal with noise, as given by:

$$y_i = 1.414 \left(\sqrt{\frac{1}{n}\sum_{i=0}^{n} G_i{}^2} \right) 10^{\left(\frac{SNR}{20}\right)} sin\left(\frac{2\pi i}{N}\right) + G_i \tag{9}$$

2.3. Extraction of Original Uncorrupted Signal

2.3.1. Simple Moving Average Filter: The principle of Simple Moving Average (SMA) filter is to compute the mean from a set of data points over a specific span of time. The mean is taken from an equal number of data either side of a central value which ensures that the variations in the mean are aligned with the variations in the data.

Let at any instant k, the average of the neighboring "p" samples on either side of the central point of any data sequence, y_i, is given by:

$$y_s^{(k)} = \frac{1}{2p+1}\sum_{i=k-p}^{k+p} y_i \tag{10}$$

$$y_s^{(k)} = \frac{1}{2p+1}\left[y_{k-p} + y_{k-p+1} + \cdots + y_{k+p}\right] \tag{11}$$

$$y_s^{(k)} = \frac{1}{2p+1}\left[\sum_{i=k-p}^{k} y_i + \sum_{i=k+1}^{k+p} y_i\right] \tag{12}$$

where, $y_s^{(k)}$ is the smoothened filtered value for the kth data point, "p" is the number of neighboring data points on either side of $y_s^{(k)}$, with 2p+1 as the data span, as shown in Figure 3.
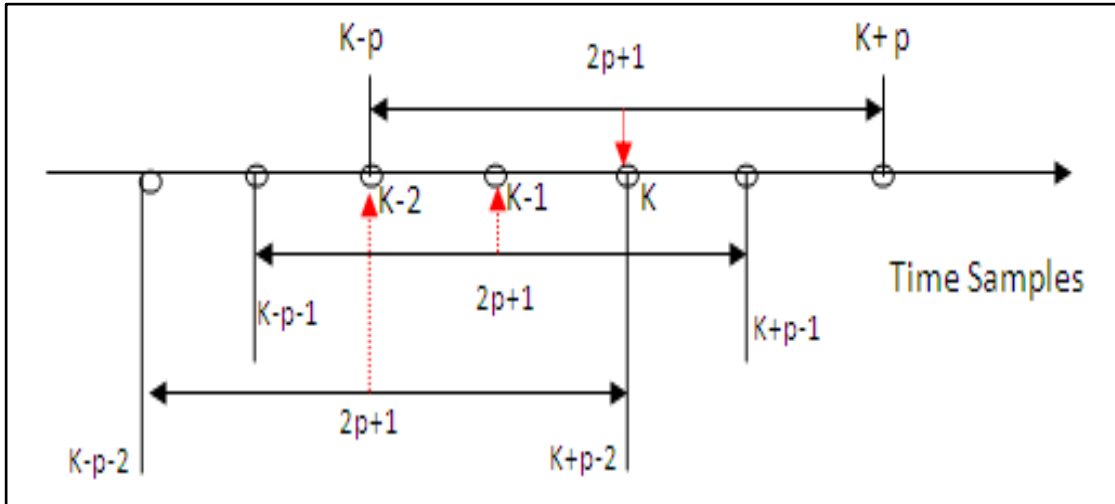

Figure 3. At Any Instant, a Moving Window of (2p+1) Values are used to Calculate the Average of the Data Sequence

Similarly, at the previous time instant, *k-1*, the average of the latest *(2p+1)* samples is:

$$y_s^{(k-1)} = \frac{1}{2p+1}\left[\sum_{i=k-p-1}^{k-1} y_i + \sum_{i=k}^{k+p-1} y_i\right] \tag{13}$$

Therefore,
$$y_s^{(k)} - y_s^{(k-1)} = \frac{1}{2p+1}\left[y_{k+p} - y_{k-p-1}\right] \tag{14}$$

$$y_s^{(k)} = y_s^{(k-1)} + \frac{1}{2p+1}\left[y_{k+p} - y_{k-p-1}\right] \tag{15}$$

Equation (15) represents the recursive form of simple moving averaging filter because the value of $y_s^{(k)}$ is calculated using its previous value, $y_s^{(k-1)}$ as reference. Regardless of the data window (2p+1), this technique places equal emphasis on all data points. Thus a value in the past will have the same influence as current measurement when calculating the filtered signal. The sequential logic of the simple moving average filter design is shown in Figure 4.

2.3.2. Exponentially Weighted Moving Average Filter: The moving average filter regards each data point in the data window to be equally important when calculating the average (filtered) value. In dynamic systems, however, the most current values tend to reflect better the state of the process. A filter that places more emphasis on the most recent filtered data

would therefore be more useful, especially in an online instrumentation system [15]. Such a filter can be designed by following the procedure, as explained below.

At any instant k, the average of the previous "p" samples of any data sequence y_i, is given by:

$$y_s^{(k)} = \frac{1}{p}\sum_{i=k-p+1}^{k} y_i \tag{16}$$

Considering the mean with one additional point

$$y_s^{(k+1)} = \frac{1}{p+1}\sum_{i=k-p+1}^{k+1} y_i = \frac{1}{p+1}\left[y_{k+1} + \sum_{i=k-p+1}^{k} y_i \right] \tag{17}$$

```
                    ┌─────────────┐
                    │    Start     │
                    └─────────────┘
                           │
                  ┌─────────────────┐
                  │ Initialize filter│
                  │     output       │
                  └─────────────────┘
                           │
     ┌──────────────────────────────────────┐
     │ • Select a window with (2p+1) data    │
     │   point from the input sequence.      │
     │ • Data point w.r.t. kᵗʰ instant is    │◄──────┐
     │   [yₖ₋ₚ,yₖ₋ₚ₊₁,.....yₖ,......yₖ₊ₚ₋₁, yₖ₊ₚ] │      │
     └──────────────────────────────────────┘       │
                           │                         │
     ┌──────────────────────────────────────┐       │
     │ Filtered output at the kᵗʰ instant is  │       │
     │                                        │       │
     └──────────────────────────────────────┘       │
                           │                         │
                    ◇ End of data ◇ ──── NO ─────────┘
                      sequence
                      reached?
                           │ Yes
                    ┌─────────────┐
                    │    Stop      │
                    └─────────────┘
```

Filtered output at the kth instant is

$$y_s^{(k)} = \frac{1}{2p+1}\sum_{i=k-p}^{k+p} y_i$$
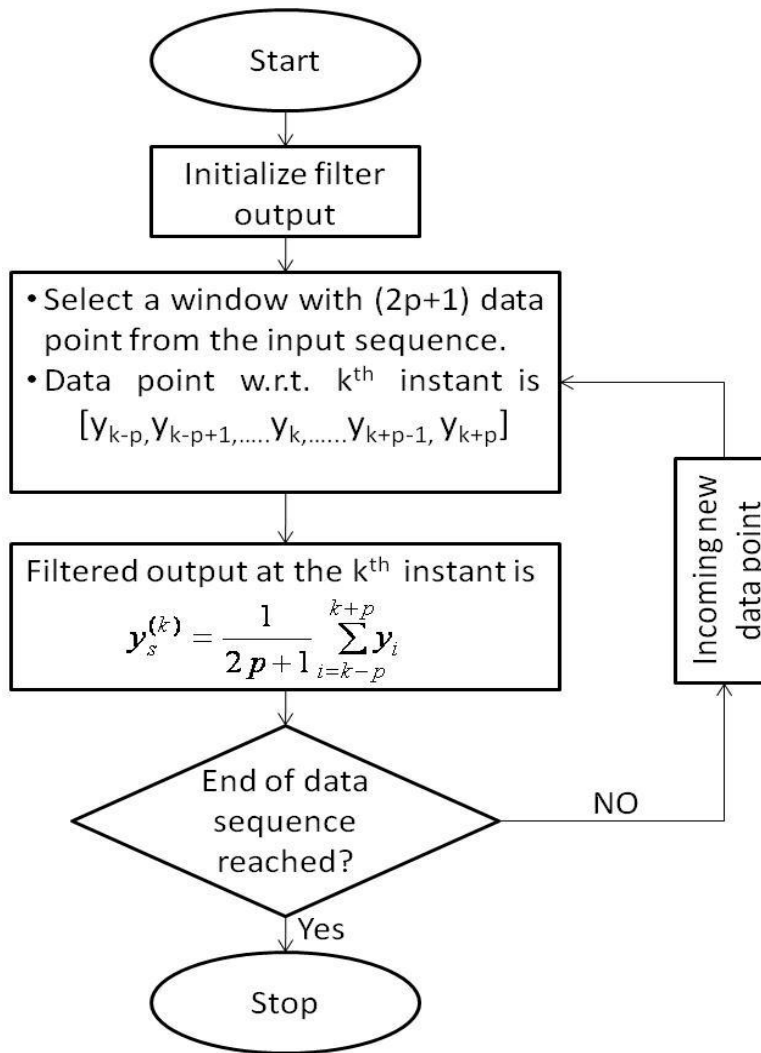
Incoming new data point

Figure 4. Sequential Logic for Moving Average

Since, $\sum_{i=k-p+1}^{k} y_i = p y_s^{(k)}$

therefore,

$$y_s^{(k+1)} = \frac{1}{p+1}\left[y_{k+1} + py_s^{(k)}\right] = \left(\frac{1}{p+1}\right)y_{k+1} + \left(\frac{p}{p+1}\right)y_s^{(k)} \qquad (18)$$

By shifting the time index back one time-step, we obtain the corresponding expression for $y_s^{(k)}$ as:

$$y_s^{(k)} = \left(\frac{1}{p+1}\right)y_k + \left(\frac{p}{p+1}\right)y_s^{(k-1)} \qquad (19)$$

Assuming $\alpha = \frac{p}{p+1}$, implies that $(1-\alpha) = \left(\frac{1}{p+1}\right)$

Hence, the exponential moving average filter can be written as:

$$y_s^{(k)} = \alpha y_s^{(k-1)} + (1-\alpha)y_k \qquad (20)$$

Equation (20) represents the Exponentially Weighted Moving Average Filter which is identical to a discrete first-order low-pass filter. When used as a filter, the value of $y_s^{(k)}$ is computed using the previous filtered value, $y_s^{(k-1)}$. The value of the filter constant, α, dictates the degree of filtering. Since $p \geq 0$, this means that $0 \leq \alpha < 1$. When a large number of points are being considered, $\alpha \rightarrow 1$, and $y_s^{(k)} \rightarrow y_s^{(k-1)}$. The Exponentially Weighted Moving Average filter places less importance to current data, which is generally noisy, and emphasizes more on the cumulative effect of older data. However, the weighting for each older data point decreases exponentially as shown in Figure 5. Flow chart for Exponentially Weighted Moving Average filter is illustrated in Figure 6.
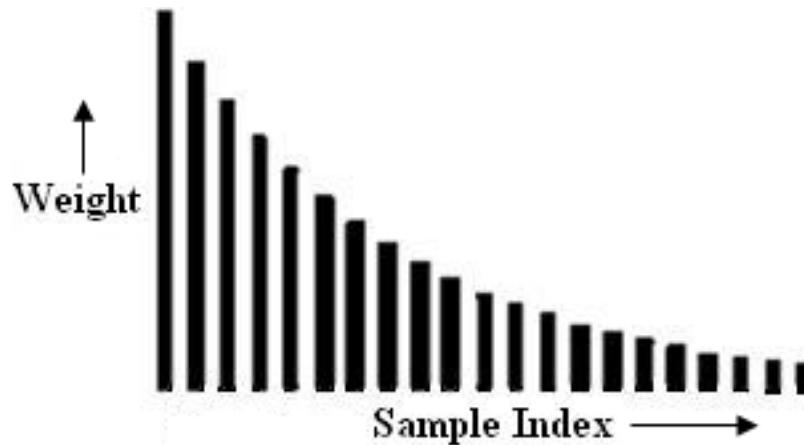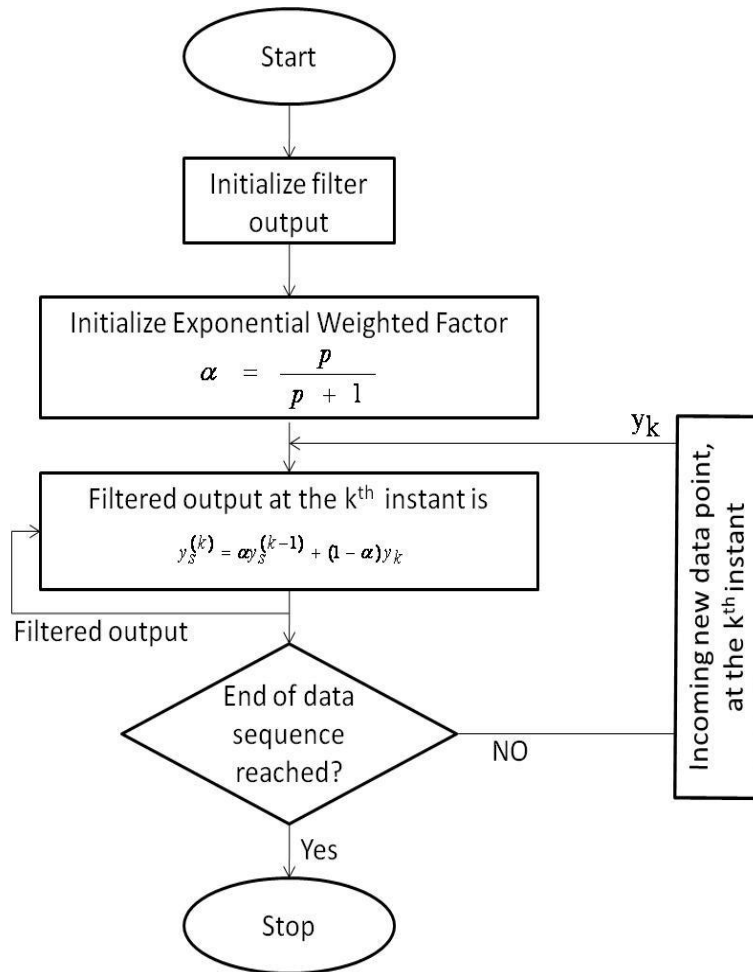

Figure 5. Exponential Weighting Effect

Figure 6. Sequential Logic for Exponentially Weighted Moving Average

3. Simulation Model

The simulation model is developed in MATLAB for validation of analytical results. A quantitative measurement of statistical parameters of extracted signal is also carried out and presented below.

- Sine wave of unity magnitude with 64 samples per cycle has been generated as the pure signal.

- Randomly distributed Gaussian noise with zero mean has been generated and added with the pure signal to simulate noisy signal from the sensor. Signals of different SNRs are generated by varying the variance of the noise.

- The filtering capability of noisy signals having different SNRs is performed by Simple Moving Average and Exponential Moving Average technique by varying the length of neighboring samples (p).

- Performance evaluation of the filtering capability of both methods has been carried out by calculating the error value at all the sample points.

- • The Root Mean Square(RMS) Error is computed from these error values. Based on Monte Carlo method, the simulations have been carried out on 1000 such sets of noisy data samples and the resulting mean RMS Error has been quantified.

4. Synthesis of EWMA Filter on FPGA

Synthesis of EWMA filter is done on VLSI platform. The timing requirements and hardware resource utilization of the filter is determined. For practical implementation, FPGA platform was chosen for a SoC approach [16]. The architecture, as shown in Figure 7 has been designed targeting use of built in multiplier blocks of the Xilinx. The subtraction block generates $(1 - \alpha)$ by subtracting the input α from a constant 1. This is multiplied with the current sample value and finally added to the previous output samples scaled by a factor of α, to give the final output.
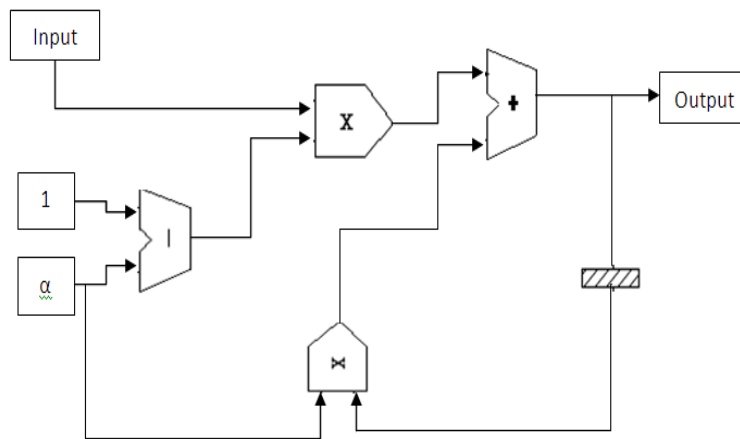
Figure 7. Architecture of the EWMA Block

5. Results and Discussion

5.1. DSP Platform

The simulations are then carried out using TMS320C6713 on DSP Environment [17, 18]. The parameters used for simulation are shown in Table1. Output signal is generated by AIC23 Codec of the DSK Board. Frequency of the output signal depends on the sampling rate (R) of the codec handler, total number of samples (N) in a cycle, and is governed by the following relation:

$$f = R/N \tag{21}$$

where, f: frequency of the output signal.

The waveforms of different signals in time and frequency domain are represented in the following figures. Required signal equivalent to uncorrupted original signal is shown Figure 8. Corrupted signal (signal added with noise) having SNR of 2 dB is shown in the Figure 9. The output of SMA and EWMA filtering techniques applied on the corrupted signals are shown in Figure 10 and Figure 11 respectively. From the qualitative observation, it can be stated that filter based on EWMA perform better than simple moving average technique.

Table 1. Statistical Parameters Used in Simulation

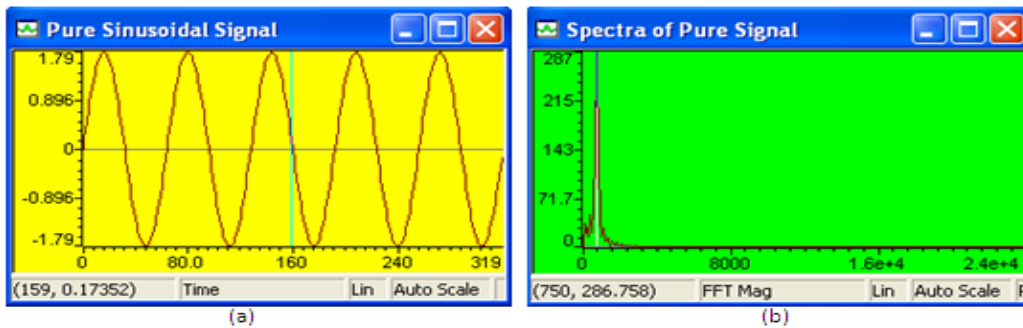| Parameters	Values
Frequency of generated signal (f)	750 Hz
Mean of noise signal (G)	0.00
Standard Deviation of noise	1.00
Number of neighboring samples (p)	10
Number of samples per cycle (N)	64
Total number of samples (n)	640
Codec sampling Frequency (R)	48 kHz
SNR	2 dB

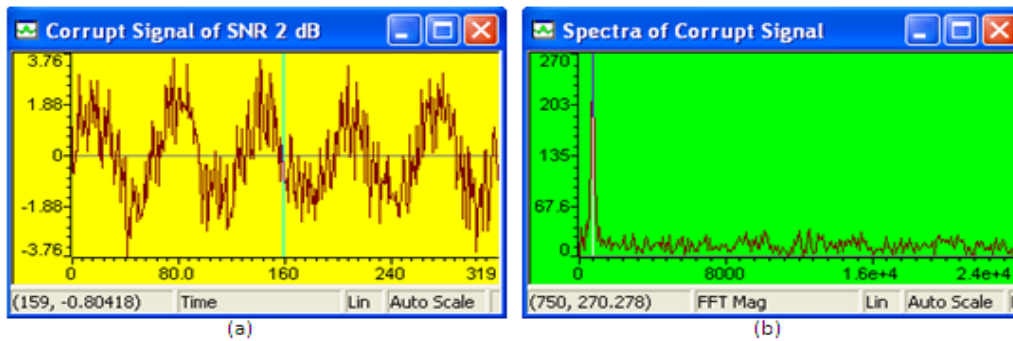**Figure 8. Generation of Pure Sinusoidal of 750 Hz with 64 Samples per Cycle**

Figure 9. Generation of Corrupt Signal with Gaussian Distributed Random Noise of SNR 2 dB
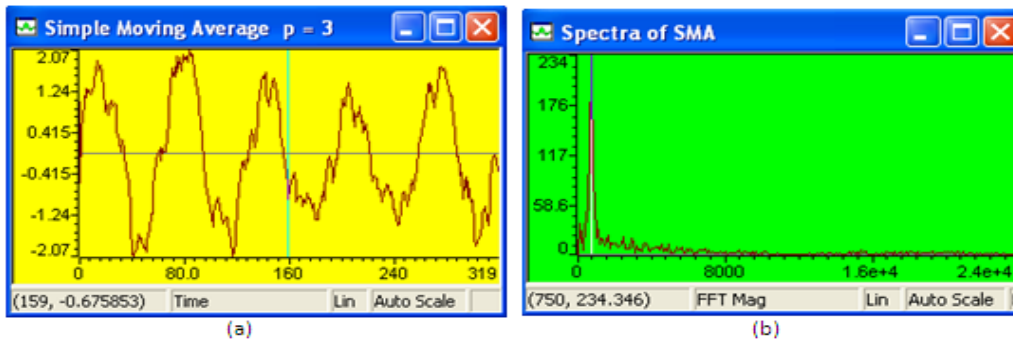
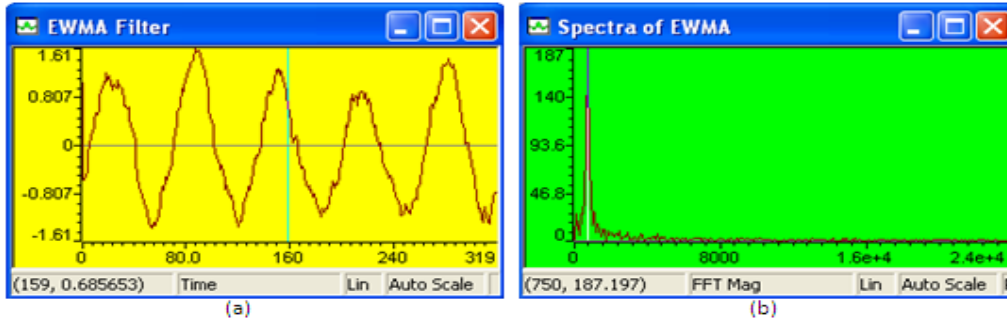**Figure 10. Filtered Output Using SMA Technique on the Signal, as Described in Figure 9(a)**

Figure 11. Filtered Output Using EWMA Technique on the Signal, as Described in Figure 9(a)

5.2. Matlab Platform

Performance evaluation of SMA and EWMA filter by varying noise variance and number of neighboring samples, 'p' is depicted in Table 2.

Table 2. Mean RMS Error Values for Different Noise Variance and Number of Neighboring Samples for SMA and EWMA Filter

Variance	Total Number of Neighboring Samples	Mean RMS Error $= \sqrt{\frac{\sum_{m=1}^{n}(y_s^{(m)} - y_i^{(m)})^2}{n}}$	
		Simple Moving Average	**Exponentially Weighted Moving Average**
2.0	3	0.2691	0.2206
	5	0.2365	0.2043
	11	0.1934	0.1991
2.5	3	0.3009	0.2453
	5	0.2644	0.2242
	11	0.2161	0.2096
3.5	3	0.3557	0.2890
	5	0.3129	0.2610
	11	0.2560	0.2323
4.5	3	0.4028	0.3265
	5	0.3541	0.2934
	11	0.2899	0.2546

- For Simple Moving Average where equal weightage is given to all the neighboring data samples, 'p', observations show a decrease in mean rms error value as 'p' is increased. This is because as we increase the number of neighboring samples, we get a better estimate of the new sample which in turn suggests better filtering performance.

- For Exponential Weighted Moving Average, the exponential weighing factor, α, determines the filtering capability. As the input values are corrupted with noise of considerably high variance, the filter dynamically places more weightage to the already filtered values as compared to the recent input data. Hence, the filter delivers a better performance.

- In high noise environments, the performance of Exponentially Weighted Moving Average filter is seen to be better than Simple Moving Average filter. For example, considering a variance of 3.5 and "p" to be 5, the mean rms error for SMA and EWMA is 0.3129 and 0.2610 respectively. EWMA filter in this case, results in a 16.59% reduced error as compared to SMA.
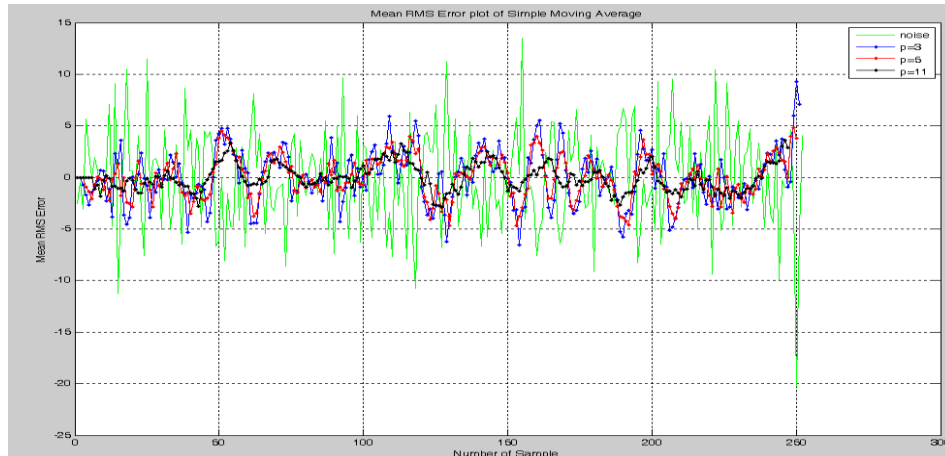

Figure 12. Mean RMS Error Plot of Simple Moving Average Filter for Different 'p'
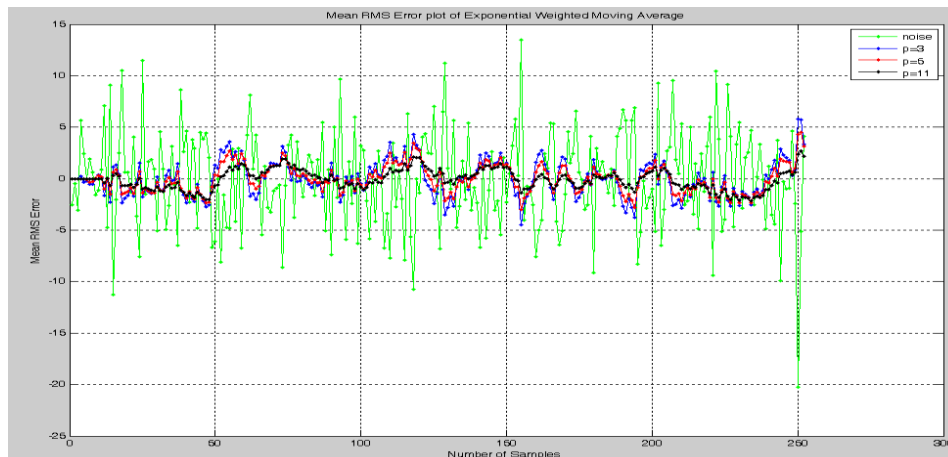

Figure 13. Mean RMS Error Plot of Exponential Moving Average Filter for Different 'p'

Quantitative measurement of mean RMS error using Simple Moving Average and Exponentially Weighted moving Average for noise variance of 4.5 is validated by qualitative observations as shown in Figure 12 and Figure 13 respectively. It is observed that on increasing the value of 'p', the RMS Error gradually decreases.

5.3. FPGA Platform

After having verified the efficacy of EWMA filter in Matlab and DSP Board, the same was synthesized on VLSI platform. The design tools for synthesis are listed in Table 3. The

proposed architecture has been coded using Verilog hardware description language (Verilog HDL) with Register Transfer Logic (RTL).

A schematic view of the synthesis result has been shown in Figure 14 which is technology independent and gives a basic logic representation of the circuit. The architecture has been mapped in Xilinx FPGA XC2S200 device with speed grade -5. The ISim timing diagrams results, verifying the outputs, are depicted in Figure 15. The device utilization summary along with partial HDL synthesis report are shown in Figure 16, Figure 17. The device utilization summary suggests that the architecture consumes minimal hardware resource. It was found that the architecture can operate at a minimum time period of 2.102 ns with a maximum frequency of 475.737 MHz, using a total memory of 229 kB.

Table 3. Design Tools for Synthesis

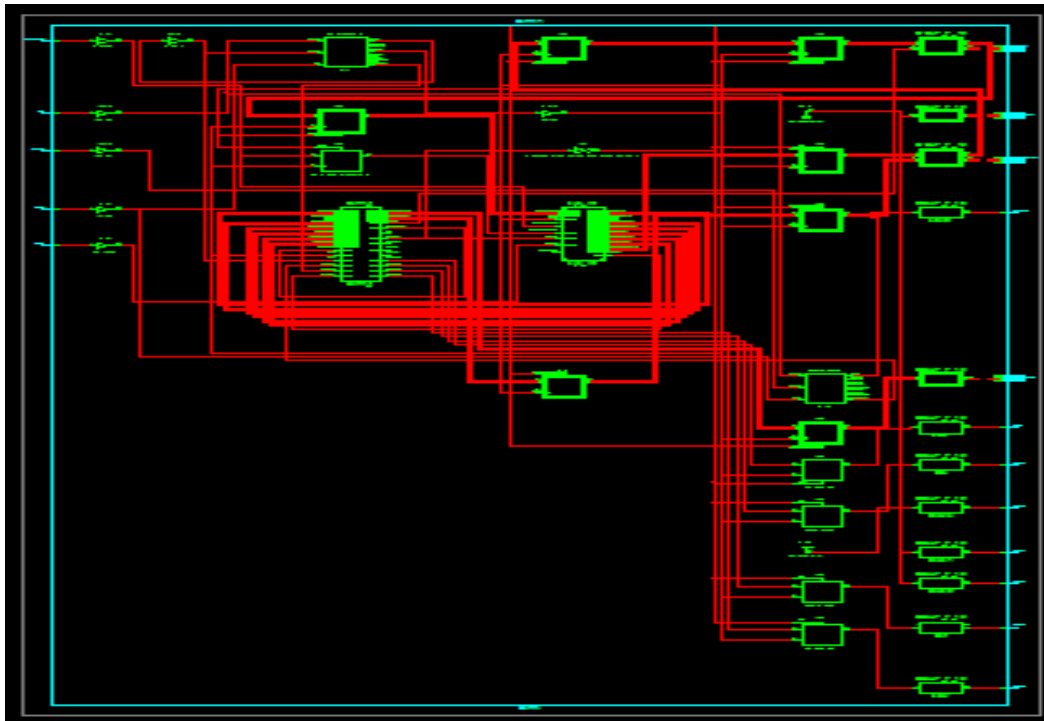| Platform	Xilinx ISE Design Suite 13.1
HDL Language	Verilog HDL
FPGA Family	Spartan 3E
Package	VQ100
Target Device	XC3S250E
Speed Grade	-5

Figure 14. Synthesis Result of the EWMA Block
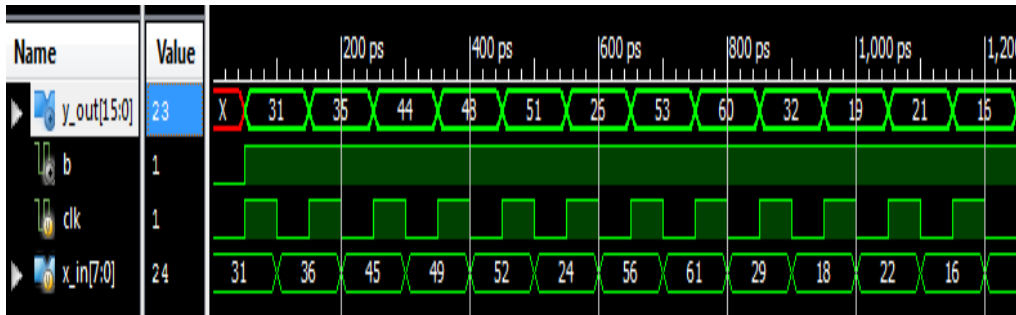
Figure 15. Test Bench (timing diagram) Result of EWMA with p = 10

Figure 16. Device Utilization Summary of Spartan 3E

```
==============================
HDL Synthesis Report

Macro Statistics
# Multipliers              : 1
  2x2-bit multiplier       : 1
# Adders/Subtractors       : 2
  1-bit adder              : 1
  2-bit adder              : 1
# Registers                : 3
  1-bit register           : 3


==============================
```

Figure 17. HDL Synthesis Report

5. Conclusion

Statistical analysis of different digital filtering techniques used for simple low-pass filter applied on corrupted signals having different SNRs has been carried out using MATLAB, DSP TMS320C6713 and FPGA. The research introduces some of the niche terminologies used in the area of signal processing and shows how signal averaging technique having less complexity than conventional digital filters can reduce the effects of noise. Two filtering algorithms based on averaging are considered and it is observed that the performance of EWMA filter is better in comparison to the SMA methodology when the signal is corrupted with more noise. EWMA based filter design is synthesized on FPGA considering real time issues. Further optimization may be carried out for applications requiring higher throughput.

References

[1] J. G. Proakis and D. G. Manolakis, "Digital signal processing: principles, algorithms and applications", Pearson Prentice Hall, **(2007)**.

[2] J. -L. Shyu, S. -C. Pei and Y. -D. Huang, "Design of Variable Two-Dimensional FIR Digital Filters by McClellan Transformation", IEEE Trans on Circuit and System, vol. 56, no. 3, **(2009)**, pp. 574-582.

[3] X. Jiang and Y. Bao, "FIR Filter Design based on FPGA", IEEE International Conference on Computer Application and System Modeling, vol. 13, **(2010)**, pp. 621-624.

[4] R. Mahesh, "New reconfigurable architectures for implementing FIR filters with low complexity", IEEE Transaction on computer-aided design of integrated circuits and systems, vol. 29, no. 2, **(2010)**.

[5] L. A. Taylor, R. G. Bennett and T. J. Dorsett, "The application of signal averaging to the nibp oscillometric waveform", Clinical Engineering/Medical Informatics, IEEE-EMBC and CMBEC, **(1995)**, pp. 691-692.

[6] B. Baker, "The power of moving-average digital filters", EDN Magazine, **(2005)**, pp. 30.

[7] J. S. Hunter, "The exponentially weighted moving average", Journal of Quality Technology, pp. 203-209.

[8] M. J. S. Smith, "Application-Specific Integrated Circuits", Addison-Wesley, **(1997)**.

[9] J. Villasenor and W. H. Mangione-Smith, "Configurable Computing", Scientific American, vol. 276, no. 6, **(1997)**, pp. 54-59.

[10] S. Brown and J. Rose, "FPGA and CLPD Architectures: A Tutorial", IEEE Design & Test of Computers, vol. 13, no. 2, **(1996)**, pp. 42-57.

[11] D. A. Buell and K. L. Pocek, "Custom Computing Machines: An Introduction", Journal on Supercomputing, vol. 9, **(1995)**, pp. 219-230.

[12] S. Hauck, "The Roles of FPGAs in Reprogrammable Systems", Proc. of IEEE, vol. 86, no. 4, **(1998)**, pp. 615-638.

[13] S. W. Smith, "The scientist and engineer's guide to digital signal processing", California Technical Publishing, Second Edition.

[14] www.statisticalengineering.com/central_limit_theorem.htm.

[15] S. H. Hwang, H. Chen and C. Chang, "An exponentially weighted moving average method for identification and monitoring of stochastic systems", Industrial & Engineering Chemistry Research, vol. 47, no. 21, **(2008)**, pp. 8239-8249.

[16] W. Wolf, "FPGA-Based System Design", Prentice Hall, **(2004)**.

[17] P. V. Ingole, A. K. Sapkal, G. G. Sarate and S. R. Hirekhan, "Implementation of signal generator (DSP) using TMS320C6713 DSK", International Journal of Computer Science & Engineering Technology, vol. 1, Issue 2, **(2011)** March, pp. 95-98.

[18] R. Chassaing, "Digital signal processing and applications with the C6713 and C6416 DSK", John Wiley and Sons, **(2005)**.

Authors

S. K. Shome

Saikat Kumar Shome received his Bachelor of Technology Degree in Computer Science and Engineering from Future Institute of Engineering and Management, Kolkata, India in 2010. Presently, he is engaged as a Scientist Trainee at CSIR-Central Mechanical Engineering Research Institute, Durgapur, India and persuing M.Tech in Mechatronics. His research areas include Mechatronics, Bio-Medical Image Processing, Network Security and VLSI Signal Processing.

SRK Vadali

Siva Ram Krishna Vadali received his Bachelor of Technology in Electronics & Communication Engineering from Jawaharlal Nehru Technological University, Hyderabad and MS (By Research) from IIT, Kharagpur in Telecommunications Engineering. He is presently working as a Scientist, in Central Mechanical Engineering Research Institute,

Durgapur (A CSIR Laboratory), India. His research areas include Signal Processing, Digital Communications, and Image Processing.

U. Datta

Uma Datta received her B.Tech. degree in Electronics and Communication Engineering from Institute of Radio Physics and Electronics, Kolkata, India and M.Tech. degree in Industrial Electrical Systems from NIT, Durgapur, India respectively. She has been working as a scientist in the department of Electronics and Instrumentation in Central Mechanical Engineering Research Institute (a National Laboratory under Council of Scientific and Industrial Research), Durgapur since 1981 and is currently a Senior Principal Scientist there. Her research interests include Control System for Machine Automation, Wireless Ad Hoc and sensor networks and Energy Harvesting System to extend Lifetime of Sensor Network. As of today, she is associated with a number of government and private sponsor projects and has published thirty five (35) research papers in various journals and conferences. She is a member of IEEE (Communication Society) and is a reviewer of several IEEE conference papers.

S. Sen

Siddheswar Sen received his Dipl. in Electronics & Telecommunication Engineering from Birla Institute of Technology, Kolkata in 1997. From 1998 to 2005 he worked in ECIL Rapiscan Limited (A Govt. of India undertaking Joint Venture Organization). Currently he is associated with CSIR-Central Mechanical Engineering Research Institute, Durgapur, India. His work area includes design and development of Embedded System for Robotics and Control. As of today, he has published six research papers in various journals and conferences.

A. Mukherjee

Arpita Mukherjee received the B.E. degree in electrical engineering from Jalpaiguri Government Engineering College, Jalpaiguri, India, in 2001 and M.Tech degree in Electrical Engineering from the Regional Engineering College, Durgapur, India, in 2003. She obtained PhD in Electrical Engineering from Bengal Engineering and Science University, Shibpur, India in 2011. She is presently a CSIR Quick Hire Scientist at CMERI, Durgapur, India. Her areas of interest include control system, statistical signal processing, Bayesian filtering, target tracking.