LITERATURE REVIEW REPORT

IMPLEMENTATION OF LEAST MEAN SQUARE ALGORITHM USING VERILOG

(Submitted in partial fulfilment of the course INSTR/EEE F367 under Dr.K.Solomon Raju,CEERI)

Submitted by:

Aditi Jain-2012A3PS031P

Gitika Meher-2012A8PS283P

AIMS AND OBJECTIVES OF THE REVIEW:

- To learn about the theory and working of the LMS (Least mean square) algorithm.
- To understand the related mathematics behind the LMS algorithm.
- To know about the different VLSI implementations of the algorithm.
- To apply and implement the LMS algorithm using ModelSim.
- To learn VHDL and implement the algorithm using VHDL.

BACKGROUND TO THE REVIEW TOPIC:

An adaptive filter may be understood as a self-modifying digital filter that adjusts its coefficients in order to minimize an error function according to present conditions, as shown in figure 1.

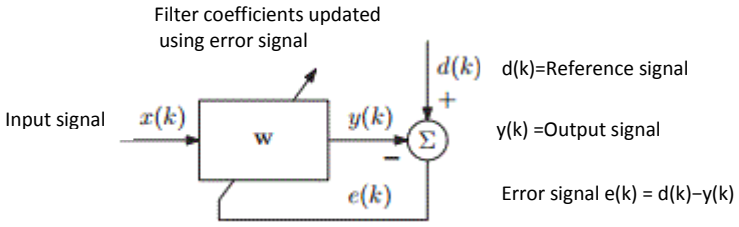

Figure 1: Basic block diagram of adaptive filter

Least mean squares (LMS) [1] algorithm is one such algorithm used to mimic a desired filter by finding the filter coefficients calculated using the least mean squares of the error, as shown in figure 2. It was invented in 1960 by Stanford University professor Bernard Widrow and his first Ph.D. student, Ted Hoff. The filter adjusts the filter weights to pass the desired input signal while reducing the noise portion of the signal with little to no filter roll-off up to the Nyquist rate (Fs/2). An adaptive filter can alter its own frequency response in order to improve the filter's performance.

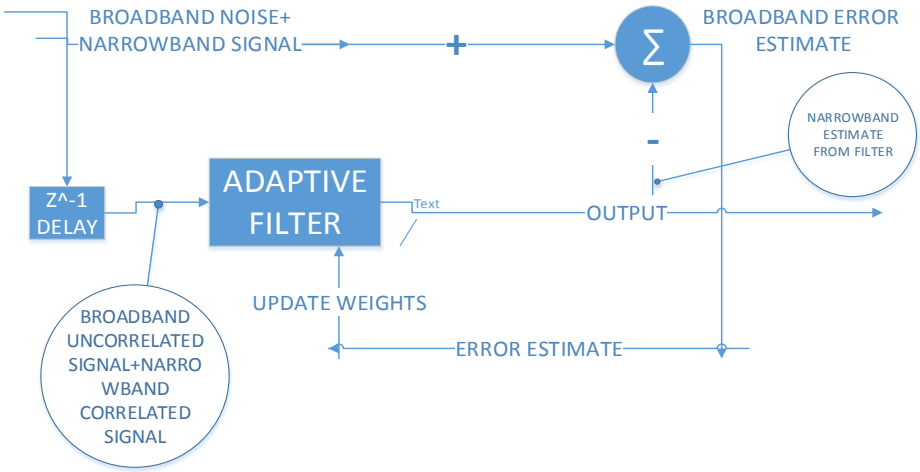

Figure 2: Flowchart showing LMS Algorithm

RELATED MATHEMATICS: [2]

Consider the transversal (FIR) filter with input x (n) i.e. vector of the M (filter length) most recent input samples at sampling point n and d(n) be the sequence of desired response samples.

$$x(n) = [x(n), x(n-1), ..., x(n-M+1)] \quad \text{------ (1)}$$

and w(n) i. e. vector of filter coefficients as

$$w(n) = [w_0(n), w_1(n), ... w_{M-1}(n)] \quad \text{------ (2)}$$

where $w_0 - w_{M-1}$ are weights of filter coefficients.

Note: x(n) and w(n) are generally input and filter coefficients of a defined filter, provided where LMS algorithm is applied. If not, x(n) and w(n) are both assumed equal to zero as starting condition.

At some discrete time n, the filter produces an output y(n) which is linear convolution sum given by

$$y(n) = \sum_{k=0}^{M-1} w(n)\ x(n-k) \quad \text{------ (3)}$$

Also can be represent in vector form as

$$y(n) = w^T(n)\, x(n) \quad \text{------ (4)}$$

The error signal e(n) is difference of this output with the primary signal d(n) given by ,

$$e(n) = d(n) - y(n) = d(n) - w^T x(n) \quad \text{------ (5)}$$

And by squaring error we get

$$e^2(n) = d^2(n) - 2d(n)x^T(n)w(n) + w^T(n)x(n)x^T(n)w(n) \quad \text{------ (6)}$$

To optimize the filter design, we choose to minimize the mean-square value (MSE), shown by

$$J= E [|e(n)|^2]=1/M(\sum_{k=0}^{M-1} [e^2(n-k)]) \quad \text{------} \quad (7)$$

For stationary inputs, the resulting solution is commonly known as Weiner filter, which is said to be optimum in the mean square error and the weights are chosen according to equation (7).

Now, let us define

$$e_w(n) = d(n)-\sum_{k=0}^{M-1} w_k x(n-k) = d(n) - w^T x(n) \quad \text{------} \quad (8)$$

Then

$$J_w = E[e_w(n)e_w^T(n)] = E[(d(n) - w^T x(n))(d(n) - x^T(n)w)] \quad \text{------} \quad (9)$$

The mean-squared error (J) is a second order function of the tap weights in the transversal filter. The dependence of the J on the unknown tap weights may be viewed in the form of a multi-dimensional parabaloid , as shown in figure 3, with a uniquely defined minimum point, also known as optimum Weiner solution.

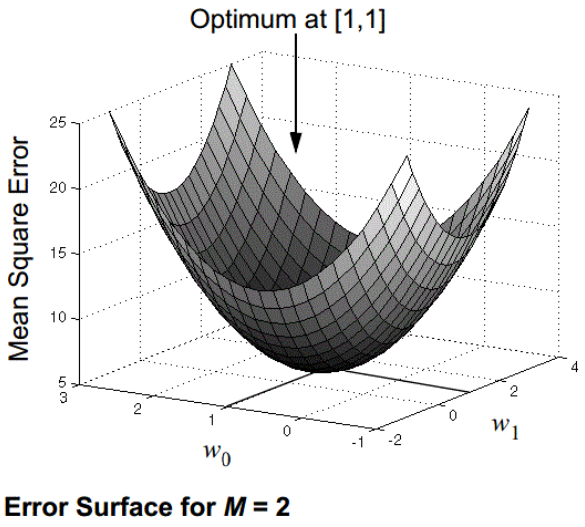

Figure 3: Parabaloid showing dependence of J on two tap weights

The optimal weights are found by setting

$$\partial/\partial w_k (E \{ e^2(n) \}) = 0 \qquad (k=0,1,.....,M-1)$$
$$= 2E [e(n) \partial e(n) /\partial w_k]$$
$$= -2E [e(n)x(n-k)] \qquad \text{(from equation 5)} \quad \text{------} \quad (10)$$

From the orthogonality principle we choose the weights such that the error is orthogonal to the observations (data), i.e.,

$$E\{ x(n-k) (d(n)-y(n)) \}=0 \qquad (k=0,1,.....,M-1) \quad \text{------} \quad (11)$$

This results in a filter that is optimum in the sense of minimum mean-square error.

The resulting system of equations

$$\sum_{k=0}^{M-1} w_k E\{ x(n-k)\, x(n-m) \} = E\{ x(n-k)\, (d(n)) \}$$

$$\sum_{k=0}^{M-1} w_k \varphi_{xx}(m-k) = \varphi_{xd}(-k) \qquad \text{------} \quad (12)$$

Equation (12) is known as the Wiener-Hopf or normal equations.

φ_{xx} is the autocorrelation function of and φ_{xd} is the cross-correlation function between x(n) and d(n)

In matrix form we can write

$$\varphi_{xx}\, w_o = \varphi_{xd} \qquad \text{------} \quad (13)$$

The optimal weight vector is given by

$$w_o = \varphi_{xx}^{-1}\, \varphi_{xd} \qquad \text{------} \quad (14)$$

Another approach of calculating corrected filter weights which does not involve the use of above mentioned matrices is as follows:

The gradient of the criterion J(n) with respect to the parameters w(n) is

$$\nabla_{w(n)} J(n) = \begin{bmatrix} \frac{\partial J(n)}{\partial w0} \\ \frac{\partial J(n)}{\partial w1} \\ \vdots \\ \frac{\partial J(n)}{\partial wn} \end{bmatrix} = -2E[x(n)e(n)] \qquad \text{-----} \quad (15)$$

Using steepest descent algorithm,

$$w(n+1) = w(n) - 1/2\mu \nabla w(n) J(n)$$

$$= w(n) + \mu E[u(n)e(n)] \qquad \text{-----} \quad (16)$$

$0 < \mu < 2\, \lambda_{max}$ for stability, λ_{max} being the maximum eigenvalue of autocovariance matrix R.

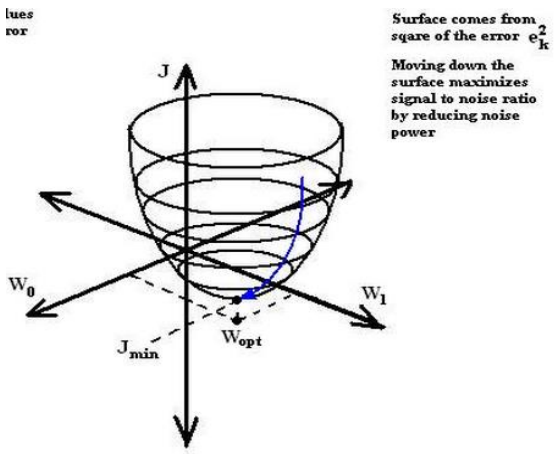

Figure 4: Steepest Gradient Algorithm

Ideally the optimal weight solution can be obtained by applying the steepest descent method to the error surface, but since the true gradient cannot be determined, a practical implementation involves estimating the gradient from the available data using the least-mean-square (LMS) algorithm

LMS algorithm will use an immediately available approximation

$$\hat{\nabla}_{w(n)}J(n) = -2x(n)e(n) \quad \text{-----} \quad (17)$$

Therefore,

$$w(n+1) = w(n) - 1/2\mu\hat{\nabla}_{w(n)}J(n) = w(n) + \mu x(n)e(n) \quad \text{-----} \quad (18)$$

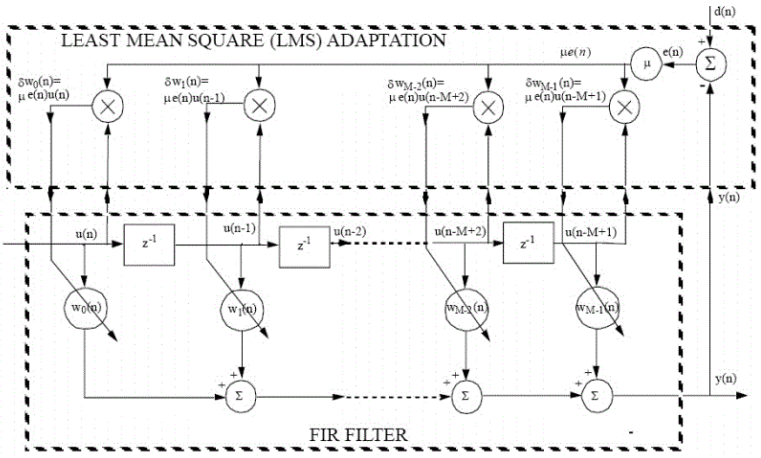

Figure 5: LMS Algorithm

Note: The complexity of the algorithm is 2M + 1 multiplications and 2M additions per iteration.

ADAPTIVE NOISE CANCELLATION CONFIGURATION:

There are four major types of adaptive filtering configurations; adaptive system identification, adaptive noise cancellation, adaptive linear prediction, and adaptive inverse system. All of the above systems are similar in the implementation of the algorithm and have the same signal requirements, but different in system configuration.

In our project, we were provided with accelerometer readings for 45 minutes at three different frequencies. Our task was to find the three unknown frequencies.

Since our readings had noise mixed with them, we made use of the second configuration, Adaptive Noise Cancellation Configuration, shown in figure 6 to eliminate the noise and then plot its fft (fast fourier transform) plot to find out the frequencies.

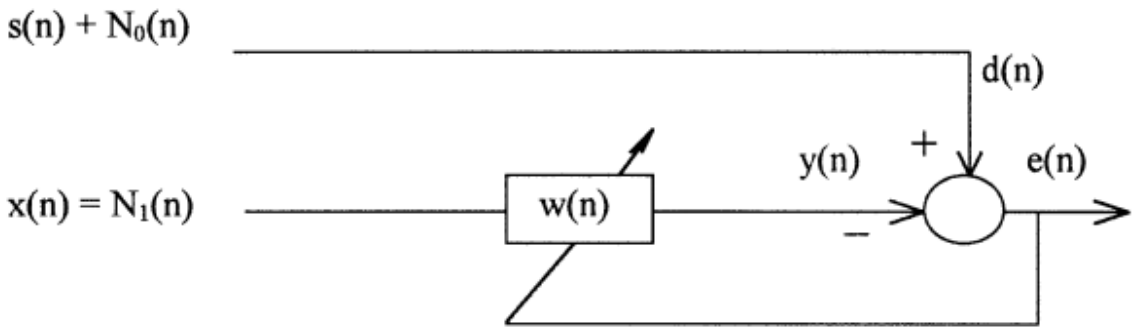

Figure 6: Adaptive Noise Cancellation Configuration

In this configuration the input x(n), a noise source N1(n), is compared with a desired signal d(n), which consists of a signal s(n) corrupted by another noise N0(n). The adaptive filter coefficients adapt to cause the error signal to be a noiseless version of the signal s(n). Both of the noise signals for this configuration need to be uncorrelated to the signal s(n). In addition, the noise sources must be correlated to each other in some way, preferably equal, to get the best results. The error signal should converge to the signal s(n), but not converge to the exact signal.

In our particular case, the accelerometer readings were input as the desired signal d(n) and x(n) was assumed to be a white Gaussian noise. This particular noise signal was chosen since it is a basic noise model used to mimic the effect of many random processes that occur in nature. We expected our error signal e(n) to converge to the true accelerometer readings.

The configuration was coded and tested in the following three different softwares:

- MATLAB
- ModelSim (using Verilog)
- Xilinx Ise

The results obtained in each of the three have been discussed below.

1. MATLAB:
   Shown below in figure 7 is a plot of the original accelerometer readings, used as desired signal in the lms configuration.

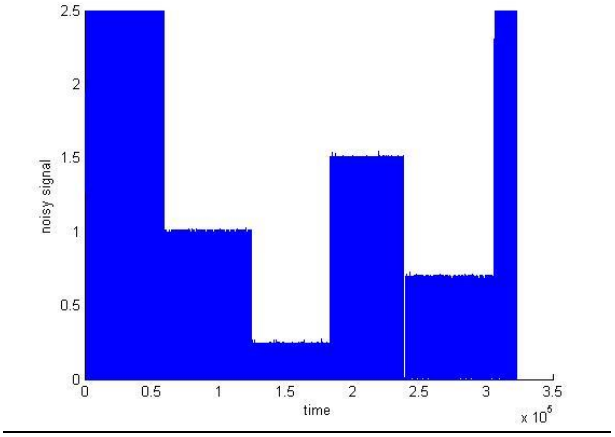

Figure 7: Desired Signal

Figure 8 shows the plot of error signal, which comprises of the true accelerometer readings.

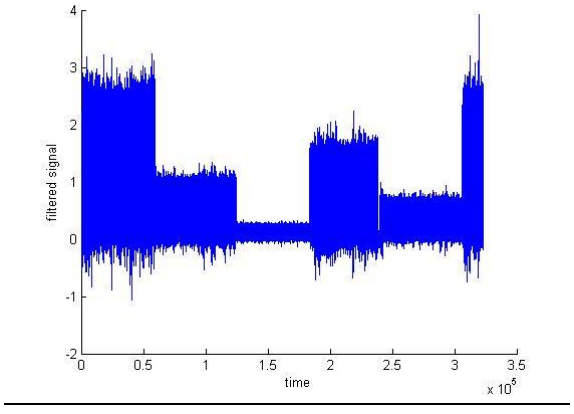

Figure 8: Error signal

Figures 9 and 10 show the fft plot of the desired signal and figures 11 and 12 show the fft plot of the error signal.

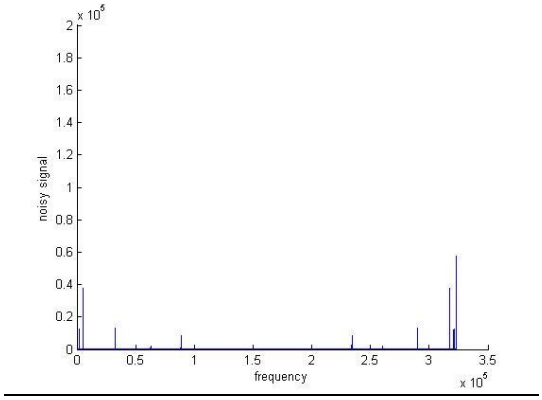

Figure 9: FFT plot of desired signal

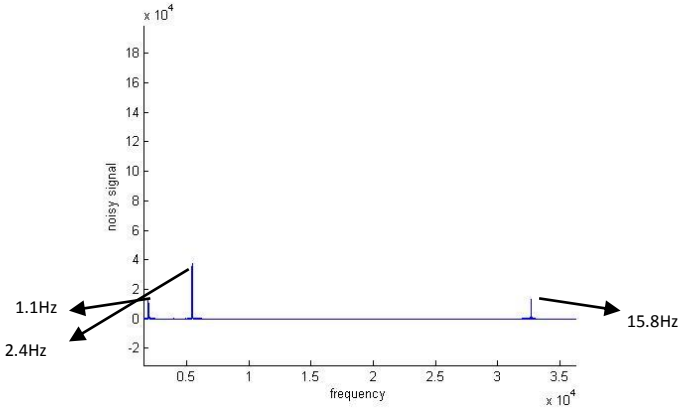

Figure 10: FFT plot of desired signal (zoomed)

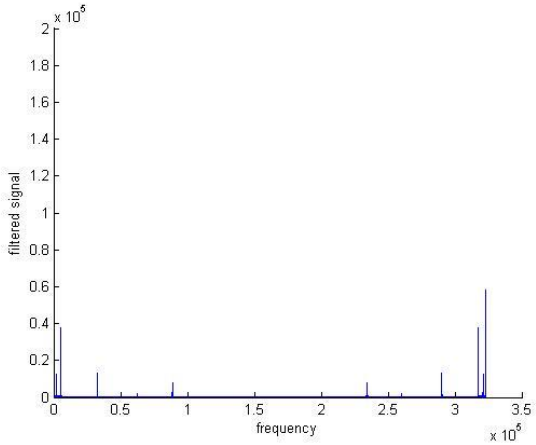

Figure 11: FFT plot of error signal

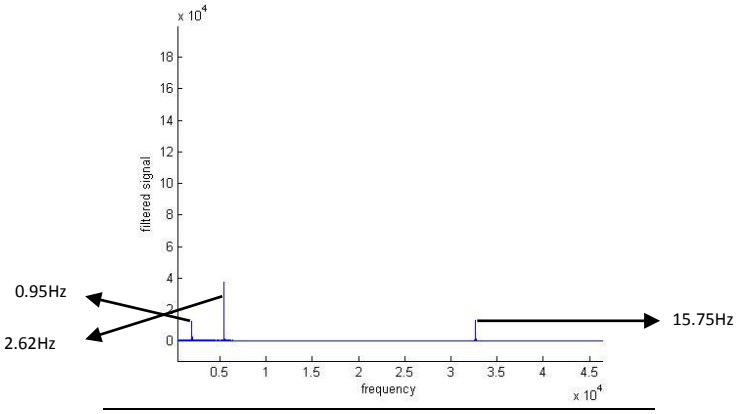

Figure 12: FFT plot of error signal (zoomed)

2. VERILOG:

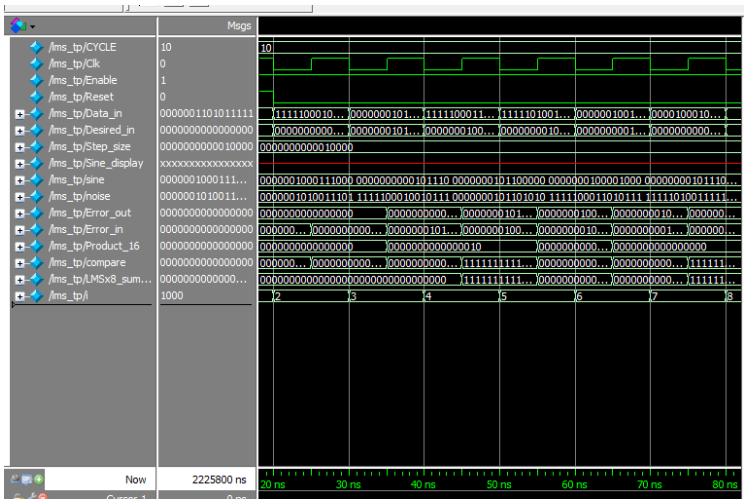

Figure 13: Output waveforms in ModelSim