6-1-2008

An FPGA architecture design of a high performance adaptive notch filter

Michael James Szalkowski

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
An FPGA Architecture Design of a High Performance Adaptive Notch Filter

by

Michael James Szalkowski

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

Supervised by

Dr. Marcin Łukowiak
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
June 2008

Approved By:

Dr. Marcin Łukowiak
Assistant Professor, RIT Department of Computer Engineering
Primary Adviser

Dr. Daniel B. Phillips
Associate Professor, RIT Department of Electrical Engineering
Committee Member

Dr. Juan Cockburn
Associate Professor, RIT Department of Computer Engineering
Committee Member
Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title: An FPGA Architecture Design of a High Performance Adaptive Notch Filter

I, Michael James Szalkowski, hereby grant permission to the Wallace Memorial Library reproduce my thesis in whole or part.

Michael James Szalkowski

Date
Dedication

This thesis is dedicated to my family for the endless support they provide to me, up to the moon, the stars and the sky.
Thanks to my committee members Dr. Daniel B. Phillips and Dr. Juan Cockburn for providing me with such great advice and inspiring conversations. Thanks to my adviser Dr. Marcin Łukowiak for keeping me focused, not only throughout this process, but throughout my years at RIT. I would not have been able to achieve this if it were not for you.
Abstract

The occurrence of narrowband interference near frequencies carrying information is a common problem in modern control and signal processing applications. A very narrow notch filter is required in order to remove the unwanted signal while not compromising the integrity of the carrier signal. In many practical situations, the interference may wander within a frequency band, in which case a wider notch filter would be needed to guarantee its removal, which may also allow for the degradation of information being carried in nearby frequencies. If the interference frequency could be autonomously tracked, a narrow bandwidth notch filter could be successfully implemented for the particular frequency. Adaptive signal processing is a powerful technique that can be used in the tracking and elimination of such a signal.

An application where an adaptive notch filter becomes necessary is in biomedical instrumentation, such as the electrocardiogram recorder. The recordings can become useless when in the presence of electromagnetic fields generated by power lines. Research was conducted to fully characterize the interference.

Research on notch filter structures and adaptive filter algorithms has been carried out. The lattice form filter structure was chosen for its inherent stability and performance benefits. A new adaptive filter algorithm was developed targeting a hardware implementation. The algorithm used techniques from several other algorithms that were found to be beneficial.

This work developed the hardware implementation of a lattice form adaptive notch filter to be used for the removal of power line interference from electrocardiogram signals. The various design tradeoffs encountered were documented. The final design was targeted
toward multiple field programmable gate arrays using multiple optimization efforts. Those results were then compared.

The adaptive notch filter was able to successfully track and remove the interfering signal. The lattice form structure utilized by the proposed filter was verified to exhibit an inherently stable realization. The filter was subjected to various environments that modeled the different power line disturbances that could be present. The final filter design resulted in a $-3$ dB bandwidth of 15.8908 Hz, and a null depth of $-54$ dB. For the baseline test case, the algorithm achieved convergence after 270 iterations. The final hardware implementation was successfully verified against the MATLAB simulation results. A speedup of $\sim 3.8$ was seen between the Xilinx Virtex-5 and Spartan-II device technologies. The final design used a small fraction of the available resources for each of the two devices that were characterized. This would allow the component to be more readily available to be added to existing projects, or further optimized by utilizing additional logic.
# Contents

Dedication ................................................................. iii

Acknowledgments ....................................................... iv

Abstract ........................................................................... v

1 Introduction ................................................................. 1
   1.1 A Brief Review of Adaptive Filtering ............................ 1
      1.1.1 Design Considerations ......................................... 3
   1.2 A Brief Review of Electromagnetic Fields ...................... 4
      1.2.1 Maxwell’s Equations ............................................ 4
      1.2.2 Electromagnetic Waves ....................................... 5
   1.3 Overview of the Thesis Document ................................. 5

2 Theory ........................................................................... 7
   2.1 The Removal of Narrowband Interference ...................... 7
   2.2 Signal Environment of the Electrocardiogram ................. 9
      2.2.1 Characteristics of the ECG ................................. 9
      2.2.2 Sources of Interference .................................... 10
      2.2.3 Characteristics of Power Line Interference ............. 11
   2.3 Power Systems Stability and Power Line Conditioning .... 13
      2.3.1 Surges ......................................................... 14
      2.3.2 Voltage Variations ......................................... 15
      2.3.3 Harmonics ................................................... 15
      2.3.4 Underfrequency ............................................. 16
   2.4 Adaptive Notch Filter Components ............................... 18
      2.4.1 Review of current Update Algorithms ................. 18
      2.4.2 Review of current Filter Structures ..................... 27
6 Synthesis Results ................................................................. 92
  6.1 Testbench Verification ...................................................... 92
  6.2 Lattice Structure Component ........................................... 93
  6.3 Update Algorithm Components ........................................ 93
    6.3.1 Variance Estimator Component .................................. 94
    6.3.2 Step-size Updater Component .................................... 95
    6.3.3 Coefficient Updater Component .................................. 95
  6.4 Overall Analysis .......................................................... 96

7 Conclusions and Future Work ............................................... 100
  7.1 Future Implementation Work .......................................... 100
List of Figures

1.1 Adaptive filter block diagram representations .......................... 1
1.2 The overall process of adaptation consists of searching for the minimum of a performance surface. The mean-square-error ($\xi$) is plotted against the weights of the filter. The optimum weights result in a minimum MSE. If one of the weights is held constant, the performance surface becomes one-dimensional (right). ......................................................... 2
1.3 An electromagnetic wave is transverse: the $E$ and $B$ fields are perpendicular to the direction of propagation ............................................................... 5

2.1 Adaptive notch filter Prediction configuration for direct form realization .......................... 8
2.2 Performance surface of multiple sinusoidal input signal ................................. 8
2.3 An ECG trace provides the pattern of electrical activity across the heart .......... 10
2.4 PSD estimates of a narrowband interference signal with multiple harmonics 12
2.5 Cascading multiple second order filters allows for the removal of multiple sinusoids from the input signal ............................................................... 12
2.6 NERC Regions ........................................................................... 14
2.7 Pole-Zero Plot of Piloted Adaptive Notch Filter ................................. 24
2.8 Steiglitz and McBride Method Adaptive Notch Filter ................................. 26
2.9 Direct Form Filter Structure ................................................................ 27
2.10 One element of the normalized lattice structure ......................................................... 30
2.11 Details of the normalized lattice form using the rotation element block ................................. 31
2.12 Pole-Zero Plot of Direct Form Notch Filter ......................................................... 33
2.13 Frequency Response of 2nd Order Direct Form Notch Filter ................................. 34
2.14 Complementary notch/bandpass filter pair in lattice form is derived from the all-pass function ................................. 35
2.15 Pole-zero plot of the lattice form notch filter ......................................................... 36
2.16 Frequency response of the lattice form notch filter ......................................................... 37

3.1 No disturbances are present in the Baseline test case ................................. 42
3.2 Random variables ........................................................................... 44
3.3 TC2 – A surge presents itself in the PLI as a short duration spike in voltage 45
3.4 Voltage variations outside of Range A as defined by the ANSI C84.1 standard

3.5 As the distortion factor increases, each additional harmonic increasingly degrades the ECG signal

3.6 As the frequency decline increases, the system must take more drastic measures to restore a balanced load-generation

3.7 TC6 – Cumulative affects from the disturbance contributions outlined in Test Cases 1-5 on the interference signal

3.8 TC7 – A change in the magnetic flux density results in a change in the interference amplitude

4.1 The algorithm development method used feedback from each stage

4.2 Several different views of the Lattice-form adaptive notch filter performance surface with input $= 2.8 \sin(2\pi 60t)$

4.3 Approximations can be made when calculating the maximum step-size

4.4 Quantization tool developed to analyze quantization effects

4.5 Control logic for variable fraction length of step-size

4.6 The resulting steady-state MSE across the input dynamic range for a baseline interference signal

4.7 TC1b – Adaptive filter adjusts as the frequency changes randomly in the internal (left) and external (right) environments

4.8 TC2 – Adaptive filter correctly adjusts to a surge disturbance in the internal (left) and external (right) environments

4.9 TC3a:sag – Adaptive filter correctly adjusts to a sag voltage variation disturbance in the internal (left) and external (right) environments

4.10 TC3a:swl – Adaptive filter correctly adjusts to a swell voltage variation disturbance in the internal (left) and external (right) environments

4.11 TC4a – Cascaded adaptive filters correctly adjust to an acceptable level of harmonics disturbance in the internal (left) and external (right) environments

4.12 TC4b – Cascaded adaptive filters correctly adjust to a severe level of harmonics disturbance in the internal (left) and external (right) environments

4.13 TC5a – Adaptive filter correctly adjusts to a gradual decline underfrequency disturbance in the internal (left) and external (right) environments

4.14 TC5b – Adaptive filter correctly adjusts to a rapid decline underfrequency disturbance in the internal (left) and external (right) environments

4.15 TC6 – Cascaded adaptive filters adjust to a culmination of disturbances in the internal (left) and external (right) environments
4.16  TC7 – Adaptive filter correctly adjusts to a large alteration in the magnetic field in the external environment ........................................... 73
4.17  Tracking performance for a linear chirped sinusoid .......................... 73
4.18  An actual ECG signal is filtered using the proposed ANF .................. 74

5.1  Lattice filter component schematic symbol ...................................... 75
5.2  Lattice filter component RTL diagram ............................................. 77
5.3  Lattice filter component timing diagram .......................................... 78
5.4  Variance estimator component schematic symbol .............................. 80
5.5  Variance estimator component RTL diagram .................................... 82
5.6  Variance estimator component timing diagram .................................. 82
5.7  Step-size updater component schematic symbol .................................. 83
5.8  Step-size updater component RTL diagram ....................................... 85
5.9  State machine for step-size updater. Transitions to State 1 result in increasing $\mu$, while transitions to State 2 result in decreasing $\mu$. .................. 85
5.10  Step-size updater component timing diagram ..................................... 86
5.11  Weight updater component schematic symbol .................................... 87
5.12  Weight updater component RTL diagram ......................................... 89
5.13  State machine for writing weight values. Weights are written in the order $\cos \theta, \sin \theta, -\sin \theta$ ......................................................... 89
5.14  Weight updater component timing diagram ....................................... 90
5.15  Final design overview timing diagram for single block implementation ... 90
5.16  Final design overview timing diagram for cascaded implementation ...... 91

6.1  Testbench verification setup ............................................................. 93
# List of Tables

2.1 Regional Reliability Councils ........................................... 14
2.2 ANSI C84.1 (120 V Base) .............................................. 15
2.3 Load Shedding Programs .................................................. 17
3.1 Testing Environments ...................................................... 40
3.2 Test Cases ................................................................. 41
3.3 Harmonic component values ............................................. 46
4.1 Step-size state machine table .......................................... 55
4.2 Proposed lattice form adaptive notch filter characteristics .......... 61
4.3 Lattice form Adaptive Notch Filter MATLAB simulation results .... 63
5.1 Lattice filter component signal descriptions ....................... 76
5.2 Microcode control signals table ....................................... 79
5.3 Variance estimator component signal descriptions ................ 81
5.4 Step-size updater component signal descriptions ................. 83
5.5 Step-size state machine table ......................................... 86
5.6 Weight updater component signal descriptions .................... 88
6.1 Lattice filter synthesis report device utilization summary and timing report 94
6.2 Lattice filter post place and route report device utilization summary and timing report 94
6.3 Variance estimator synthesis report device utilization summary and timing report 95
6.4 Variance estimator post place and route report device utilization summary and timing report 95
6.5 Step-size updater synthesis report device utilization summary and timing report 96
6.6 Step-size updater post place and route report device utilization summary and timing report 96
6.7 Coefficient updater synthesis report device utilization summary and timing report 97
6.8 Coefficient updater post place and route report device utilization summary and timing report ................................................................. 97
6.9 Adaptive notch filter synthesis report device utilization summary and timing report ................................................................. 98
6.10 Adaptive notch filter post place and route report device utilization summary and timing report ...................................................... 99
6.11 Maximum input sample rate for each of the targeted implementations ................................................................. 99
1. Introduction

1.1 A Brief Review of Adaptive Filtering

Adaptive filtering is a powerful technique for signal processing and control systems [1–4]. It allows a filter to adjust its weights according to the signals it encounters. Figure 1.1 displays common block diagram representations of adaptive filter systems. The output of the programmable filter $\hat{y}[n]$, is subtracted from a reference sequence $y[n]$, which produces an error sequence $\epsilon[n]$. The error sequence and the input sequence $x[n]$ provide information to properly update the filter’s weights. For this application, the weights will determine the notch center frequency and the notch bandwidth. The depth of the null in the adaptive notch filter is superior to that of the fixed filter because the adaptive process maintains the correct phase relationships for canceling the undesired signal contents [1].
The overall process of adaptation can also be thought of in geometric terms, where adaptation is the procedure of searching for the minima of a performance surface [1]. The performance surface consists of the error surface plotted in the space of the filter weights. The performance surface function is described in Equation 1.1. The mean-square-error (MSE) is defined as the expected value of the error sequence ($\varepsilon[n]$). This provides statistical information relating filter performance to weight values, for a specific input signal. Adaptive processes cause the weight vector to seek the minimum of the performance surface. An example performance surface for this application is illustrated in Figure 1.2. For a sinusoidal input source centered at 60 Hz, it is clear what the optimum weight value for $\theta_1$ would be.

$$\text{MSE} = \xi = E[\varepsilon^2[n]] \quad (1.1)$$

Figure 1.2: The overall process of adaptation consists of searching for the minimum of a performance surface. The mean-square-error ($\xi$) is plotted against the weights of the filter. The optimum weights result in a minimum MSE. If one of the weights is held constant, the performance surface becomes one-dimensional (right).
1.1.1 Design Considerations

Adaptive filtering consists of a programmable filter and an update algorithm. Both of these components allow for optimizations. The update algorithm consists of minimizing a cost function, typically denoted as $J(n)$. Minimization of the cost function implies that the result of the error sequence will be closer to converging to zero after each iteration, resulting in continuously improved approximations $\{\hat{y}[n]\}$ of $y[n]$. The update algorithm for the filter weights has experienced a significant degree of research. Numerous theoretical derivations have been presented, some of which are based on the least mean squares (LMS) [5–9], the normalized least mean squares (NLMS) [10–15], and the Steiglitz-McBride methods (SMM) [16, 17]. Some algorithms use the gradient of the error sequence, some utilize phase, some techniques have constrained weight vectors and others have variable weighting factors. The Pilot Notch technique can be added to any existing algorithm in an attempt to improve performance [18]. The algorithms need to consider stability [19], correlation between random variables, and signal to noise ratios. It is important to know the various qualities and trade-offs between different algorithms, because the operation and performance is closely related to the statistical parameters of the signal environment.

Another important design feature is the filter structure. Traditional adaptive filters are implemented with finite impulse response adaptive line enhancers (FIR-ALE) [20]. However, an exorbitant number of taps would be needed in order to obtain a filter with a narrow notch bandwidth [20], especially at high sample rates. Infinite impulse response (IIR) implementations can achieve similar characteristics with substantially fewer taps. With the computational and performance advantages, IIR filters have shown significant performance benefits over their FIR counterparts [9, 16]. The theory of adaptive FIR filters is quite mature, especially when compared with adaptive IIR filtering.

There are several choices for the structure of the filter, the most popular being direct-form and lattice [3]. Extensions into the multiple sinusoid cancellation case often results
into cascade forms as well. Each of the structures has its own advantages and disadvantages, and the stability [21] of the structure needs to be considered, which requires a complete analysis of the overall system.

1.2 A Brief Review of Electromagnetic Fields

1.2.1 Maxwell’s Equations

Maxwell’s equations relate electric and magnetic fields and their sources (electric charges and current densities). Ampère’s law (Equation 1.2) originally explained how a current induces a magnetic field proportional to the magnitude of the current.

\[ \nabla \times \mathbf{B} = \mu_0 \mathbf{J} \quad (1.2) \]

A current induces a magnetic field.

However, when applied beyond magnetostatics (current no longer constant), an additional term (Equation 1.3a,1.3b) must be considered. From this equation we can see that a magnetic field (\( \mathbf{B} \)) can be induced by a current density (\( \mathbf{J} \)) as well as a changing electric field (\( \mathbf{E} \)). The constants are the permeability of free space (\( \mu_0 \)) and the permittivity of free space (\( \varepsilon_0 \)).

\[ \nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} \quad \text{Differential Form} \quad (1.3a) \]

\[ \oint \mathbf{B} \cdot d\mathbf{l} = \mu_0 I_{\text{enc}} + \mu_0 \varepsilon_0 \int \left( \frac{\partial \mathbf{E}}{\partial t} \right) \cdot d\mathbf{a} \quad \text{Integral Form} \quad (1.3b) \]

A changing electric field induces a magnetic field.

Similarly, a symmetry is observed in Faraday’s law (Equation 1.4). The changing magnetic field induced by an alternating current, induces an electric field.

\[ \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \quad (1.4) \]

A changing magnetic field induces an electric field.
1.2.2 Electromagnetic Waves

As explained by Maxwell’s equations, a time-varying electric field induces a magnetic field and vice versa. The electric and magnetic fields oscillate in phase and are perpendicular to each other and the direction of propagation, as depicted by Figure 1.3. This second property is characteristic of transverse waves, thus an electromagnetic wave is a transverse wave. Therefore particles do not move along with the electromagnetic wave, they will simply oscillate about their individual positions as the wave front passes by at the speed of light \( c \approx 3 \times 10^8 \text{ m/s} \).

![Figure 1.3: An electromagnetic wave is transverse: the E and B fields are perpendicular to the direction of propagation](image)

1.3 Overview of the Thesis Document

Chapter 2 introduces the problem addressed by this thesis and describes the paradigm used to design the adaptive notch filter. The signal environment of the Electrocardiogram (ECG) is presented and the basic principles of Electrocardiography are described. The source of the power line interference (PLI) observed in ECG recordings and the importance of its removal are explained. The interference is then further classified in order to make important engineering decisions later on in the design process, and develop an accurate set of test
cases. The two design components of the adaptive notch filter are the update algorithm and filter structure. The current research efforts of both areas are reviewed.

The test cases that were developed are presented in chapter 3. A set of test cases based on the theory presented in chapter 2 are detailed, as well as two test environments. A set of requirements was created to assess the performance of the ANF throughout the algorithm development process.

The development effort utilizing MATLAB software is presented in chapter 4. An incremental approach to the design was used, starting with a basic design and adding more complexity. The basic designs allowed for a better understanding of the subject matter, while more robust testing allowed for optimizations and refinements to be made. Final efforts were guided toward developing an algorithm with the most efficient utilization of hardware resources. These results were used as a baseline to verify the hardware implementation.

A robust hardware design can be accomplished by taking a systematic approach. A clear understanding of each of the components allow for a more concise final design. The VHDL design process for the adaptive notch filter is presented in chapter 5. Each of the components are explained with the aid of register transfer level (RTL) diagrams, timing diagrams, state machine diagrams, and microcode control signals tables.

In chapter 6, the synthesis results are presented. The final design targeted two Xilinx device families, the Spartan-II and a Virtex-5. The Spartan-II device was representative of a low cost implementation, while the Virtex-5 device represented a high end implementation. An overall analysis of the results are presented.

Chapter 7 presents conclusions drawn from this project. Future implementation directions are offered that would allow for a practical implementation that could be used in an educational environment to facilitate student learning.
2. Theory

2.1 The Removal of Narrowband Interference

The introduction of interference into a system near frequencies carrying information requires a very narrow notch filter. Prediction is the underlying mechanism when applying an adaptive notch filter. It requires the identification of filter weights to force the sequence \( \hat{y}[n] \) to resemble the sequence \( y[n] \), the equivalent of minimizing the error sequence \( e[n] \).

In the context of this application, \( y[n] \) would be the reference signal of the sinusoidal interference that is to be tracked and removed, and \( \zeta \) refers to any measurement and background noise that is assumed to be independent of \( y[n] \), and Gaussian in nature. The objective is to estimate the signal frequency \( (\omega_0) \) so that \( \hat{H}(z) \) describes a notch filter centered at \( \hat{\omega}_0 \). After some manipulations (Equation 2.1), a prediction configuration (Figure 2.1) can be applied, which will allow for the correct weight values to be found in order to estimate a notch filter \( (\hat{H}(z)) \) centered at the signal frequency \( (\omega_0) \) of \( y[n] \). The parameter \( r, (0 < r < 1) \), describes the notch bandwidth. As the bandwidth parameter approaches 1, the notch bandwidth becomes more narrow.

\[
\hat{H}(z) = \frac{1 + \hat{\omega}_0 z^{-1} + z^{-2}}{1 + \hat{\omega}_0 r z^{-1} + r^2 z^{-2}} = \frac{N(z)}{D(rz)} = 1 - \frac{D(rz^{-1}) - N(z^{-1})}{D(rz)} = 1 - z^{-1}\frac{D(rz) - N(z)}{D(rz)}
\]

Identification of the filter weights of \( \hat{H}(z) \) excited by the input \( y[n] \) such that the error sequence goes to zero would result in successfully obtaining a notch filter from the reference signal. In theory, algorithms have updated the rational transfer function of degree matching that of the system [3]. This is known as the sufficient order case. For this application, two
Figure 2.1: Adaptive notch filter Prediction configuration for direct form realization

degrees of freedom would be needed for a single sinusoidal input [4]. In Equation 2.1, \( \hat{H}(z) \) corresponds to a second-order transfer function, which will allow for the identification of a single sinusoid. However, it should be noted that in practice the system order is either unknown, or unrealizable, and a finite order filter can rarely model a signal environment exactly [3]. This is known as the undermodelled case, and the IIR adaptive notch filter implemented by this project will fall into this category.

For an input signal consisting of multiple sinusoids, there exist many local minima in the performance surface (Figure 2.2) for the second order filter described. Each minimum corresponds to the removal of one of the sinusoids [22]. The local minimum that the filter would converge to would depend on the starting point.
2.2 Signal Environment of the Electrocardiogram

An important engineering decision made in adaptive filtering involves determining what type of model to use based on the conditions of the signal environment. The signal environment for this project is that of the electrocardiogram (ECG), which is an instrument that records the electrical activity of the heart over time. A basic knowledge of the ECG will enhance the understanding of what artifacts need to be removed and what an acceptable level of removal would be.

2.2.1 Characteristics of the ECG

The ECG is one of the oldest and most enduring tools used by cardiologists [23]. The principle behind the ECG consists of the electrical activity that is transmitted throughout the body when the heart is depolarized in order to trigger its contraction. This occurs for every beat and can be recorded via electrodes on the skin: one on each limb and six across the chest. This allows for 12 different electrical views of the heart, each reflecting different angles at which the electrodes “look” at the heart. This way, a three-dimensional electrical picture of the heart can be put together.

The basic ECG trace can be seen in Figure 2.3, and is comprised of three main waves (P, QRS and T). The P wave represents atrial depolarization. The QRS wave complex represents ventricular depolarization and can be broken down into its three components. The Q wave corresponds to the depolarization of the interventricular septum, the R wave corresponds to the depolarization of the main mass of the ventricles and the S wave corresponds to the final depolarization of the ventricles at the base of the heart. The T wave represents ventricular repolarization.

The shape, rate and rhythm of the waves are utilized when reading ECGs. Therefore, it is important to provide a clear and accurate representation in order for cardiologists to correctly interpret the ECG. However, a cardiologist may not always be present, especially in the case of an out-of-hospital sudden cardiac arrest. Automatic external defibrillators
(AEDs) are often available for these situations which could result in an untrained person using the machine [24]. For these cases, the AED needs to correctly determine whether or not the operator should shock the patient. Therefore, automated diagnosis of life-threatening cardiac arrhythmias is of high importance. Due to the nature of the situation, the environment is no longer controlled like it could be in a hospital. The ECG recording becomes more susceptible to high levels of interference which would compromise automated detection [25]. The suppression of interference becomes pivotal to allow an algorithm in software to correctly diagnose life-threatening cardiac arrhythmias such as ventricular fibrillation and asystole (no contraction activity).

2.2.2 Sources of Interference

There are many sources of interference for electrocardiogram recordings, such as alternating current (AC) power supplies [26]. This type of interference is commonly referred to as power line interference (PLI). Maxwell’s equations [27] can be used to show that power line interference can enter the ECG recordings by way of electromagnetic induction. Electromagnetic fields are naturally induced by current-carrying electrical devices, and are proportional to the magnitude of the electrical current as explained by Ampère’s law with Maxwell’s correction (Eq. 2.2). From this equation we can see that a magnetic field ($B$) can be induced by a current density ($J$) as well as a changing electric field ($E$). The well-known
constants are the permeability of free space ($\mu_0$) and the permittivity of free space ($\varepsilon_0$).

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$ \hspace{1cm} (2.2)

Applying Faraday’s law (Eq. 2.3a) to the ECG recording circuitry, the changing magnetic field induced by the AC current induces an electric field in the closed loop formed by the ECG lead wires and measurement circuitry. The same electromotive force (emf, $\mathcal{E}$) is observed when the ECG equipment is moved through the magnetic field. This is an example of motional emf, expressed by the flux rule (Eq. 2.3b) which states that the emf is equal to the rate of change of the flux ($\Phi$). This is the same result that can be derived from Faraday’s law (Eq. 1.4).

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$ \hspace{1cm} (2.3a)

$$\mathcal{E} = -\frac{d\Phi}{dt}$$ \hspace{1cm} (2.3b)

Utilizing results from power quality research, the most realistic and robust testing conditions can be developed in terms of potential interference. As made evident through Maxwell’s equations, each of the potential disturbances that a power system can experience can also be coupled into the ECG signal. These disturbances are outlined in Section 2.2.3. By introducing these disturbances into the interference that the adaptive notch filter could encounter, a more robust set of tests can be performed. The expected voltage and frequency of the power system is used as a baseline. The various disturbances power systems experience will deviate from this baseline.

2.2.3 Characteristics of Power Line Interference

The power line interference of interest in the United States is made up of a dominant 60 Hz fundamental, a second harmonic (from rectifying power supplies), a third harmonic (from nonlinearities in motors and transformers), and perhaps some low-energy higher harmonics [4]. An estimation of the Power Spectral Density (PSD) of this noise will provide information used to decide what order filter will be necessary for its elimination. Figure 2.4
illustrates an example for the PSD estimates of the interference with additional harmonics. The average power decreases for higher harmonics. The estimate uses Welch’s Method, an averaged, modified periodogram spectral estimation method.

![Power Spectral Density Using Welch Method](image)

Figure 2.4: PSD estimates of a narrowband interference signal with multiple harmonics

As previously mentioned, two degrees of freedom are needed for each sinusoid that composes the interference. For the multiple sinusoid case, \( \hat{H}(z) \) would either need to be extended to a multi-notch configuration by choosing \( N(z) \) and \( D(z) \) as higher order polynomials, or to cascade the basic second-order sections (Figure 2.5) into sequential stages [3]. Each cascaded block would converge to a separate minima of the performance surface, eliminating the corresponding sinusoid. The cascading scheme lends itself as a more practical implementation, and can also be pipelined using techniques in hardware.

![Cascading multiple second order filters](image)

Figure 2.5: Cascading multiple second order filters allows for the removal of multiple sinusoids from the input signal
2.3 Power Systems Stability and Power Line Conditioning

Electric utilities are responsible for the generation, transmission and distribution of reliable electricity. The job of the engineer is to identify potential disturbances and create suitable solutions in order to provide reliable and uninterrupted service to loads. The industry has had planning and operating criteria and guidelines for decades but compliance was always voluntary [28]. In order to ensure the reliable distribution of service across different systems, a set of standards has been adopted and enforced. These standards address how the power system should perform. Some aspects of interest for this research in the operation and planning of power systems include real-time transmission operations, balancing load and generation, emergency operation, system restoration and voltage control. These aspects are all related to the stability and recovery of the system from potential disturbances. Knowledge of these issues in detail will allow for a more robust testing and design of a filter that is intended to track and remove power line noise.

On July 20, 2006 the Federal Energy Regulatory Commission (FERC) issued an order certifying the North American Electric Reliability Corporation (NERC) as the United States Electric Reliability Organization (ERO) under Section 215 of the Federal Power Act. NERC’s reliability standards development process has been accredited by the American National Standards Institute (ANSI) [29]. As the National ERO, NERC successfully filed for the approval of regional delegations of authority for the purpose of proposing reliability standards to the ERO and enforcing them in their respective regions [30].

The Reliability Councils approved by the FERC are listed in Table 2.1 while their respective coverage areas can be seen in Figure 2.6 [31]. While providing as much uniformity as possible, regional variations in reliability standards are necessitated due to physical differences of the power systems in each region.

A disturbance is an unplanned event that produces an abnormal system condition. The ability of the electric system to supply the aggregate electrical demand of the end-use customers at all times, taking into account the various disturbances that could be presented, constitutes reliability [32]. The standards set forth and enforced by these councils to ensure
reliability often follow those proposed by ANSI and IEEE. Applicable standards are used as guidelines to provide the most realistic testing conditions. Because many worst case scenarios are assessed, the testing conditions are as robust as necessary. The standards of most interest to this research are those that provide criteria for system recovery and stability.

<table>
<thead>
<tr>
<th>Table 2.1: Regional Reliability Councils</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERCOT</td>
</tr>
<tr>
<td>FRCC</td>
</tr>
<tr>
<td>MRO</td>
</tr>
<tr>
<td>NPCC</td>
</tr>
<tr>
<td>RFC</td>
</tr>
<tr>
<td>SERC</td>
</tr>
<tr>
<td>SPP</td>
</tr>
<tr>
<td>WECC</td>
</tr>
</tbody>
</table>

![Map of NERC Regions](image)

**Figure 2.6: NERC Regions**

### 2.3.1 Surges

A surge is a transient variation of current, voltage, or power flow across an electric system which can have extremely short duration and high magnitude [32–34]. Utility switching operations and lightning are possible causes for surges which could cause serious damage
to equipment. In most cases, surge arrestors/suppressors/protectors have sufficient energy absorbing capability to damp harmful overvoltages [32]. Power conditioners are also an option to reduce the effects of surges.

2.3.2 Voltage Variations

The ANSI C84.1 standard (Table 2.2) establishes two voltage ranges for service and utilization [35]. Service voltages outside of range A should be infrequent, and utilization equipment shall be designed to perform throughout range A. Range B includes voltages outside of range A due to practical design and operating conditions. A momentary decrease or increase in voltage outside of range A is classified as a voltage sag or swell, and those lasting longer than a few seconds are considered undervoltage and overvoltage respectively [33, 34]. Corrective measures should be taken to improve voltages to meet the requirements of Range A. As can be noted in Table 2.2, the difference in the minimum service and utilization voltages is intended to allow for a voltage drop at the utilization end.

Table 2.2: ANSI C84.1 (120 V Base)

<table>
<thead>
<tr>
<th></th>
<th>Range A: ± 5%</th>
<th>Range B: +6%/-13%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Voltage</td>
<td>114-126</td>
<td>108-126</td>
</tr>
<tr>
<td>Utilization Voltage</td>
<td>108-126</td>
<td>110-127</td>
</tr>
<tr>
<td>Service Voltage</td>
<td>104.4-127.2</td>
<td></td>
</tr>
</tbody>
</table>

2.3.3 Harmonics

The utility frequency used in the distribution of AC power varies from country to country. For the purpose of testing, the United States utility frequency standard of 60 Hz is assumed. Harmonics are sinusoidal components of a periodic wave with frequencies that are integral multiples of the fundamental frequency. Harmonics are generally not produced by the utility [33, 36]. Nonlinear loads connected to each circuit could generate harmonics that affect other loads. Therefore, other customers could create excessive harmonics that could interfere with others electronic equipment until the utility employs corrective actions. Utilities
reference the IEEE Standard 519-1992 to determine whether corrective action is necessary.

The practices outlined in IEEE Standard 519-1992 [36] address the requirements for harmonic control in electric power systems. The standard is recognized by ANSI and implemented by Regional Reliability Councils when addressing harmonics problems. The recommended practices are to be used as guidelines in the design of power systems with nonlinear loads and recommend ”worst case” conditions [36]. As defined in [36], the distortion factor (DF) is the ratio of the root-mean-square (rms) of the cumulative harmonic content amplitudes ($A_k$) to the rms value of the fundamental quantity ($A_0$), as described in Equation 2.4. The distortion factor is used to gauge the total harmonic distortion in a system from acceptable (5%) to severe ($\geq 20$%), while no individual harmonic distortion should be more than 3% [36].

$$DF = \sqrt{\sum_{k=1}^{N} \frac{A_k^2}{A_0^2}} \cdot 100\%$$

(2.4)

2.3.4 Underfrequency

As defined by the NERC operating manual[32], Frequency Regulation is the ability of a Balancing Authority to maintain Scheduled Frequency (60 Hz), and Frequency Response is the ability of a system to respond to a change in system frequency. Stable operation of a power system relies upon the generating units supplying the loads with constant voltage and frequency. If a condition arises where the load power requirement exceeds the supply power generation, a drop in frequency will occur. The mechanical equations that describe the dynamic behavior of a generator are well established, and the equation of motion (Equation 2.5 [37, 38]), also known as the swing equation, which relates machine frequency to changes in input and output power, explains the drop in frequency. If the total power of the electric load ($P_e$) exceeds the total mechanical power ($P_m$) generated, there will be a negative deviation in the speed (frequency, $\omega$) of the motor. The inertia constant $H$ is defined as
the kinetic energy in watt-seconds at rated speed divided by the VA base.

\[ 2H_\omega \frac{d\omega}{dt} = P_m - P_e \]  (2.5)

According to a 2004 report [39] prepared by the Frequency Task Force of the NERC Resources Subcommittee, Frequency Response is declining when it should be increasing due to increasing load and the associated increase in generation. Over a period from 1994 to 2002, there was an 18% decline in frequency response while load and generation grew nearly 20% over the same period [39].

The mechanism in place to avoid widespread system outages and correct for significant underfrequency situations is load shedding [38, 39]. To rapidly balance load and available generation, shedding specific amounts of load will bring the frequency back up to reasonable levels and retain the integrity of the power system. As described in the NPCC Reference Manual [40], the Automatic Underfrequency Load Shedding program provides the guidelines and procedures to stabilize the system frequency in an area during an event of declining frequency. The goal of the program is to maintain a Frequency Response such that the system frequency returns to at least 58.5 Hertz in ten seconds or less and to at least 59.5 Hertz in thirty seconds or less, for generation deficiencies up to 25% of the load [40]. The Mid-America Interconnect Network (MAIN), now part of the MRO, has similar guidelines [41]. Further details of these two sets of guidelines can be seen in Table 2.3.

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Action</th>
<th>Frequency</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>59.3 Hz</td>
<td>Automatic load shedding of 10% of its load</td>
<td>59.3 Hz</td>
<td>Shed not less than 10% of system load</td>
</tr>
<tr>
<td>58.8 Hz</td>
<td>Automatic load shedding of 25% of its load</td>
<td>59.0 Hz</td>
<td>Shed not less than 20% of system load</td>
</tr>
<tr>
<td></td>
<td></td>
<td>58.7 Hz</td>
<td>Shed not less than 30% of system load</td>
</tr>
</tbody>
</table>
2.4 Adaptive Notch Filter Components

2.4.1 Review of current Update Algorithms

Least Mean Square

The Least Mean Square (LMS) algorithm is the focus of much study and is widely used in applications due to its computational simplicity and robustness [1, 42]. It selects the filter tap weights ($w_k[n]$), so that the error sequence ($\varepsilon[n]$, Equation 2.7) is minimized in the mean square sense. The error sequence is the difference between the desired output ($y[n]$) and the filter output ($\hat{y}[n]$, Equation 2.6). The optimal weight values ($W^*$) are analogous to the minimum of the performance surface.

\[
\hat{y}[n] = H^T[n]X[n] \quad (2.6)
\]
\[
\varepsilon[n] = y[n] - \hat{y}[n] \quad (2.7)
\]

The LMS algorithm uses a gradient-based method of steepest decent for descending toward the minimum of the performance surface [1]. The gradient ($\nabla$) of the performance surface is used to develop an adaptive algorithm using a gradient-based method. The negative of the gradient vector points in the direction where the greatest decrease in error with respect to a change in weight values will occur. Instead of directly computing the gradient of $\xi$, the LMS algorithm takes $\varepsilon^2[n]$ as an estimate of $\xi$, and the derivative of $\varepsilon$ with respect to the weights ($w$) is done. This results in a gradient estimate of the form (Equation 2.8) where $X[n]$ is the vector of input samples.

\[
\hat{\nabla}[n] = \begin{bmatrix} \frac{\partial \varepsilon^2[n]}{\partial w_0} \\ \vdots \\ \frac{\partial \varepsilon^2[n]}{\partial w_L} \end{bmatrix} = 2\varepsilon[n] \quad \begin{bmatrix} \frac{\partial \varepsilon[n]}{\partial w_0} \\ \vdots \\ \frac{\partial \varepsilon[n]}{\partial w_L} \end{bmatrix} = -2\varepsilon[n]X[n] \quad (2.8)
\]

The method of steepest descent (Equation 2.9a) can be used to update the weight vector ($W$). Using the simple estimate of the gradient the LMS update algorithm is constructed.
(Equation 2.9b), where $\mu$ is the adaptation step-size and regulates the convergence characteristics.

\[
W[n + 1] = W[n] - \frac{1}{2} \mu \nabla[n]
\]  
\text{(2.9a)}

\[
W[n + 1] = W[n] + \mu e[n] X[n]
\]  
\text{(2.9b)}

The adaptation step-size parameter controls the speed of adaptation and misadjustment [1]. The misadjustment is a dimensionless measure of the deviation from optimal performance. It can be defined by Equation 2.10 as a quantitative measure of the amount by which the excess MSE exceeds the minimum mean square error ($\xi_{min}$). In [43], expressions were derived which showed that the step-size inversely relates the misadjustment and convergence time: a smaller step-size reduced the steady state error, but at the cost of a longer convergence time.

\[
M = \frac{E[\xi_k - \xi_{min}]}{\xi_{min}} = \frac{\text{Excess MSE}}{\xi_{min}}
\]  
\text{(2.10)}

The adaptation step-size parameter can also cause the algorithm not to converge at all. To ensure convergence the step-size must be chosen in the range defined by Equation 2.11, where $\lambda_{max}$ is the maximum eigenvalue of the autocorrelation matrix ($R$) of the input signal [1]. The input signal power (trace[$R$]) is often used in replace of $\lambda_{max}$ because the signal power can generally be estimated more easily than the eigenvalues of $R$. Some research [42] suggests that the upper bound is inappropriate, and the range should instead be defined by Equation 2.12.

\[
0 < \mu < \frac{1}{\lambda_{max}}
\]  
\text{(2.11)}

\[
0 < \mu < \frac{1}{3\lambda_{max}}
\]  
\text{(2.12)}

In [43], the behavior of the LMS adaptive filter was extensively studied both during the initial transients and steady state. The steady state response after convergence was considered for both the stationary (time-invariant) and nonstationary (time-variant) environments. For a time-invariant system, any weight fluctuations would be due to large values of
\( \mu \), because the position of the performance surface remains constant. For the time-variant system, the MSE function varies in position, and thus contributes to the tracking error.

The derived expression for the tracking error can be seen in Equation 2.13. Any deviation of the individual weight vectors about the ensemble mean is due to gradient noise, while any difference between the ensemble mean of the adaptive weight vectors and the optimal value \( W^* \) is due to lag in the adaptive process.

\[
(\text{Tracking Error}) = (W - W^*) \\
\equiv (W - E[W]) + (E[W] - W^*) \\
\text{weight noise} \quad \text{weight lag}
\]  \hspace{1cm} (2.13)

Gradient noise occurs because the LMS algorithm uses a gradient estimate (\( \hat{\nabla} \)), resulting in a deviation from the true gradient (\( \nabla \)) by a zero-mean gradient estimation noise vector (N), seen in Equation 2.14. The noise vector is representative of the weight fluctuations related to the size of \( \mu \). Smaller values of \( \mu \) result in smaller tracking error due to gradient noise.

\[
\hat{\nabla} = -2e[n]X[n] = \nabla + N
\]  \hspace{1cm} (2.14)

Weight lag occurs in nonstationary environments. The algorithm’s ability to adjust to a moving performance surface is determined by the size of \( \mu \). Larger values of \( \mu \) result in smaller tracking error due to weight lag.

**Variable Step-Size Least Mean Square**

Ideally, a large step-size should be used early on in the adaptation, and a smaller step-size should be employed as the algorithm approaches convergence. This will allow for faster convergence while maintaining a small misadjustment. As a consequence, variable step-size algorithms should outperform a fixed step-size algorithm if the appropriate step-size values are utilized.

The variable step-size LMS (VSS) algorithm proposed in [5] adjusts the step-size according to the squared error (2.15). The step size is always positive and is controlled by the size of the prediction error and the positive parameters \( \alpha \) and \( \gamma \), where \( \alpha \) is constrained
to the range $0 < \alpha < 1$. The $\gamma$ parameter is usually small ($10^{-4}$) while $\alpha$, the exponential forgetting parameter, was found to work well closer to 1 (0.97). Intuitively it can be seen that the step size increases with large misadjustments, and decreases with smaller misadjustments.

$$\mu'[n + 1] = \alpha \mu[n] + \gamma e^2[n]$$  \hspace{1cm} (2.15)

The algorithm defines a maximum and minimum step-size, resulting in an update based on the conditional expression defined in Equation 2.16. The constant $\mu_{\text{max}}$ is chosen to ensure convergence, while $\mu_{\text{min}}$ is chosen to provide a minimum level of tracking ability. The initial step-size is usually taken to be $\mu_{\text{max}}$, although the behavior of the algorithm is independent of this.

$$\mu[n + 1] = \begin{cases} \mu_{\text{max}} & \text{if } \mu'[n + 1] > \mu_{\text{max}} \\ \mu_{\text{min}} & \text{if } \mu'[n + 1] < \mu_{\text{min}} \\ \mu'[n + 1] & \text{otherwise} \end{cases}$$ \hspace{1cm} (2.16)

**Modified Variable Step-Size Least Mean Square**

The modified variable step-size LMS (MVSS) algorithm proposed in [6] is derived from the VSS algorithm [5]. Their analysis showed that improvements could be made to make the step-size update a more accurate reflection of the state of adaptation, more specifically improving the steady-state performance.

The objective was to ensure large $\mu$ when the weights were far from optimal, and decreasing $\mu$ as the algorithm approached the optimum. This was achieved by using an estimate of the autocorrelation of the error sequence to control the step-size update. The estimate is described in Equation 2.17a, where $\beta$ is an exponential weighting parameter that governs the averaging time constant, and is constrained to the range $0 < \beta < 1$. Limits on $\mu[n + 1]$, $\alpha$, and $\gamma$ are the same as those of the VSS LMS algorithm.

$$p[n] = \beta p[n - 1] + (1 - \beta)e[n]e[n - 1]$$ \hspace{1cm} (2.17a)

$$\mu'[n + 1] = \alpha \mu[n] + \gamma p^2[n]$$ \hspace{1cm} (2.17b)
The parameters $\beta$ and $\gamma$ provide the algorithm with an extra degree of freedom to allow for simultaneous control of convergence speed and steady state misadjustment. A larger $\gamma$ results in a larger step-size during the initial stages of adaptation, allowing for faster convergence. The $\beta$ parameter provides a tradeoff between tracking speed and excess MSE.

A method for calculating the optimal parameters for a given nonstationary environment was discussed to achieve minimum misadjustment. The method assumes that the variance of the noise process that models the time-varying optimal weight vector is known.

**Normalized Least Mean Square**

A limitation of the original LMS algorithm is that the choice of the adaptation step-size is optimal only for certain environments. In order to ensure stability, the step-size must be sufficient over the dynamic range of the input signal. This results in an overly conservative step-size, causing unnecessarily slower convergences and a non-uniform rate of convergence based on the variance of the input signal.

The normalized LMS algorithm [12–15] replaces the fixed adaptation step-size $\mu$ by an input data-dependent step-size, resulting in the update algorithm seen in Equation 2.18. The step-size is statistically related to the variance of the current input vector ($X^T[n]X[n]$), resulting in an optimal and uniform rate of convergence across the dynamic range of the input signal.

$$W[n + 1] = W[n] + \frac{1}{X^T[n]X[n]}e[n]X[n]$$  \hspace{1cm} (2.18)

**Variable Step-Size Normalized Least Mean Square**

The variable step-size NLMS (VS-NLMS) algorithm proposed in [10] builds upon the properties of the NLMS algorithm to provide a variable adaptation step-size. This allows the algorithm to meet the conflicting requirements of fast convergence and low misadjustment.

The algorithm was developed as a special case of the Affine Projection Algorithm (APA), also derived in [10]. The updates to the step-size are given in Equation 2.19, where
\( \hat{p} \) is an estimate of the projection of the weight vector onto the input autocorrelation matrix. In Equation 2.19b, \( \alpha \) is a smoothing factor bounded by \( 0 \leq \alpha < 1 \), and \( C \) is a positive constant inversely related to the SNR of the input signal. For the APA algorithm, \( C = K / \text{SNR} \).

The special case when \( K = 1 \) is defined as the VS-NLMS algorithm. Stability is guaranteed when \( 0 < \mu_{\text{max}} < 2 \). The \( \| \cdot \| \) operator is the Euclidean norm of a vector.

\[
\begin{align*}
\mu'[n + 1] &= \mu_{\text{max}} \frac{\| \hat{p}[n] \|^2}{\| \hat{p}[n] \|^2 + C} \\
\hat{p}[n] &= \alpha \hat{p}[n - 1] + (1 - \alpha) e[n] \frac{X[n]}{\| X[n] \|^2}
\end{align*}
\] (2.19a)

(2.19b)

**Nonparametric Variable Step-Size Normalized Least Mean Squares**

When the presence of noise is considered, the classical NLMS algorithm will introduce the noise into the weight estimates. The nonparametric variable step-size NLMS (NPVSS-NLMS) algorithm developed in [11] intends to eliminate the noise from the weight estimates. To accomplish this, the authors propose to find the step-size parameter in such a way that the mathematical expectation of the error squared is equal to the power of the system noise \( (\sigma_v^2) \), as described by Equation 2.20.

\[
E[\epsilon^2(n)] = \sigma_v^2
\] (2.20)

This results in the solution defined in Equation 2.21. It can be seen that before the algorithm converges, the power of the error signal \( (\sigma_e^2) \) is larger than that of the system noise \( (\sigma_v^2) \); resulting in the second term in Equation 2.21b to be \( \sim 1 \). Therefore the algorithm behaves like the traditional NLMS with a large step-size. However, as the algorithm starts to converge, \( \sigma_e^2 \approx \sigma_v^2 \), and the step-size goes to zero.

\[
\begin{align*}
W[n + 1] &= W[n] + \mu[n] e[n] X[n] \\
\mu[n] &= \frac{1}{X^T[n] X[n]} \left[ 1 - \frac{\sigma_v^2}{\sigma_e^2[n]} \right]
\end{align*}
\] (2.21a)

(2.21b)
Piloted Notch Least Mean Square

The algorithm proposed in [18] uses additional “pilot” notches in equal quantity on either side of the zero of the main notch, as described by the pole-zero plot in Figure 2.7. This is used to provide additional information on the distance between the notch frequency and interference frequency. By examining the signs of the notches, a determination of step-size can be deduced. The following is a derivation considering two pilot notches. The weight of the notches can be updated as the main notch moves around. The 2\textsuperscript{nd} order transfer function can also be seen in Equation 2.22, where \( r \) is the distance from the origin to the pole, and \( W[n] \) is the weight vector of the main notch.

\[
H(z) = \frac{z^2 + W[n]z + 1}{z^2 + rW[n]z + r^2} \tag{2.22}
\]

The weight updates of the pilot notches (\( W_h[n], W_l[n] \)) are described in Equation 2.23. The parameters \( \alpha_h \) and \( \alpha_l \) have positive values and are functions of \( W[n] \). They represent the effect on the pilot notch weights from the difference in frequency between the main notch and the corresponding pilot notch. Larger values of \( \alpha \) would represent a pilot notch.
further from the main notch.

\[ W_h[n] = W(n) - \alpha_h \] \hspace{1cm} (2.23a)

\[ W_l[n] = W(n) + \alpha_l \] \hspace{1cm} (2.23b)

The corresponding error sequences of the pilot notches can be seen in Equation 2.24, where \( \varepsilon_h \) and \( \varepsilon_l \) correspond to the error sequences of the pilot notches. Because the notches only serve as rough indicators of the interference frequency, the values of \( \alpha \) can be selected as a convenient power-of-two so that the multiplication can be replaced with a shift operation.

\[ \varepsilon_h[n] = \varepsilon[n] - \alpha_h X[n-1] \] \hspace{1cm} (2.24a)

\[ \varepsilon_l[n] = \varepsilon[n] + \alpha_l X[n-1] \] \hspace{1cm} (2.24b)

The adaptation step-size is then determined by the signs of \( X[n-1] \varepsilon[n], X[n-1] \varepsilon_h[n], \) and \( X[n-1] \varepsilon_l[n] \). The signs provide information on the relative position of the interference frequency and that of the notches. If each of the signs are the same, it would imply that the interference must be at a higher frequency (if all positive) or at a lower frequency (if all are negative) and thus far away from the main notch. This would be a case where a large step-size would be desirable. If the interference is near the main notch, the signs will not all be equal, and thus a smaller step size would be employed. The authors in [18] used two arbitrary step-sizes for the small and large steps.

**Steiglitz and McBride Method**

Another family of algorithms are based on the ideas presented by Steiglitz and McBride [17]. The SM method (SMM) is designed to estimate the parameters of the adjustable polynomials which make up the transfer function \( H(z) \) of the filter, given by Equation 2.25.

\[ H(z) = \frac{N(z)}{D(z)} \] \hspace{1cm} (2.25)

The algorithm utilizes a prefilter, \( 1/D_k(z) \) which is applied to the observed input sequence \( x[n] \), and desired output (reference signal \( y[n] \)) of the filter. The output of each of
the prefilters (denoted with primes) are multiplied by adjustable polynomials, and the difference is then taken to construct the error sequence (Equation 2.26). After each iteration to minimize the squared error, the prefilter is replaced by \( \frac{1}{D_{k+1}(z)} \). It is important to note that the SMM works best when the measurement noise is white because the stationary point will remain unique. If the noise is colored, the SM estimate is usually biased.

\[
\varepsilon[n] = D_{k+1}(z)y'[n] - N_{k+1}(z)x'[n] 
\]

\[
= D_{k+1}(z)\frac{y[n]}{D_k(z)} - N_{k+1}(z)\frac{x[n]}{A_k(z)}
\]

Utilizing the general ANF from Figure 2.1, the block diagram when applying the SMM for notch filter identification can be seen in Figure 2.8. The SMM for notch filter parameter identification updates both \( D_k(z) \) and \( N_k(z) \) such that the MSE of \( \varepsilon[n] \) is minimized. The algorithm presented in [16] describes such a filter.
2.4.2 Review of current Filter Structures

As the update algorithm affects the filter performance, the filter structure also has an important impact on the filter. The filter structure becomes increasingly important when implementing IIR filters due to the stability issues inherent with the adaptive mode of operation.

Direct Form

Perhaps the most widely studied filter model structure is that of the direct form structure. They offer a tractable form of analysis, making them easier to understand. These structures are also widely implemented because of their straightforward design.

The input-output system \( y(n) = H(z)u(n) \) can be rewritten as \( A(z)y(n) = B(z)u(n) \) which leads to the difference equation seen in Equation 2.28. The corresponding signal flow graph is sketched in Figure 2.9.

\[
y(n) + a_1 y(n-1) + \cdots + a_M y(n-M) = b_0 u(n) + b_1 u(n-1) + \cdots + b_M u(n-M) \quad (2.28)
\]

As can be seen in the flow graph, the weights are being manipulated directly under
adaptive techniques. This convenience leads to the rather tractable convergence analysis. However, there are numerical and stability problems associated with the direct form structure. The time-varying difference equation (Equation 2.28) is susceptible to unstable behavior during the adaptation process, and stability checks are normally incorporated to ensure the autoregressive part (feedback) is of minimum phase [44]. When using finite precision arithmetic, roundoff error accumulations can make the filter unreliable. Therefore, while straightforward, the limitations of the direct form structure for adaptive filtering motivates one to consider other options [3].

**Lattice Form**

The formulation of lattice form structures was presented by Gray and Markel [1, 45]. Techniques for synthesizing lattice form structure from the well known direct form transfer function were also presented. By relating the lattice form structure with that of the direct form structure, a greater basis for understanding is developed.

The direct form transfer function is presented in Equation 2.29, where the subscript $M$ is used to designate explicitly the filter size, and the leading coefficient $b_{M,0}$ is always assumed to be one. The $2M + 1$ parameters representing the filter are those of the $A_M$ and $B_M$ vectors, and these are manipulated directly under adaptive techniques.

$$H(z) = \frac{Y(z)}{X(z)} = \frac{A_M(z)}{B_M(z)}$$  \hspace{1cm} (2.29)

The parameters that represent the lattice filter are $M$ $k$-parameters $\{k_0, k_1, \ldots, k_{M-1}\}$ and $M + 1$ tap parameters $\{\nu_0, \nu_1, \ldots, \nu_M\}$. These parameters are recursively obtained from the direct form parameter vectors $A_M$ and $B_M$ as described by Equation 2.30-2.34, for $m = M, M - 1, \ldots, 1$ with $\nu_0 \doteq a_{0,0}$.
As evident by Equation 2.32, the $k$-parameters will always have a magnitude less than one for a stable filter. Should any $|k_m| \geq 1$, then $B_M(z)$ will not have all its roots within the unit circle, resulting in an unstable filter. In terms of the new variables, $A_M(z)$ is equivalent to Equation 2.35, resulting in the equivalent representation of the transfer function $H(z)$ seen in Equation 2.36.

\[
A_M(z) = \sum_{m=0}^{M} v_m z^m C_m(z) \quad (2.35)
\]

\[
H(z) = \sum_{m=0}^{M} v_m z^m C_m(z) z^{m-1} B_m(z) \quad (2.36)
\]

Further building upon their lattice structure in [45], Gray and Markel presented the normalized structure [46] based on an orthonormal polynomial structure. This normalized form is inherently limited to realizing stable functions because all of the coefficients are bounded by unity.

In order to generate different orthogonal polynomial structures, Gray and Markel defined a new set of pi-parameters which modified the polynomials and tap parameters (Equation 2.37). The normalized structure can be realized by choosing the pi-parameters such that the polynomials are made orthonormal.

\[
\hat{A}(z) = \pi_m A_m(z) \quad (2.37a)
\]

\[
\hat{B}(z) = \pi_m B_m(z) \quad (2.37b)
\]

\[
\hat{\nu} = \frac{v_m}{\pi_m} \quad (2.37c)
\]
Assuming that the filter \( H(z) \) is driven by the sequence \( u(n) \), the state vectors \( x^+_m(n) \) and \( x^-_m(n) \) are defined as having z-transforms \( \hat{A}_m(z)U(z)/A_m(z) \) and \( \hat{B}_m(z)U(z)/A_m(z) \), respectively. The output sequence \( y(n) \) is defined by Equation 2.38. The \( \pi \)-parameters that result in polynomials having unit form used to obtain the recursive equations of the normalized form can then be fully described in Equation 2.39. The implementation of these relations is depicted by the signal flow graph in Figure 2.10.

\[
y(n) = \sum_{m=0}^{M} \hat{v}_m x^-_m(n + 1) \tag{2.38}
\]

\[
x^+_m(n) = \sqrt{1 - k_m^2} x^+_{m+1}(n) - k_m x^-_m(n) \tag{2.39a}
\]

\[
x^-_{m+1}(n + 1) = k_m x^+_{m+1}(n) + \sqrt{1 - k_m^2} x^-_m(n) \tag{2.39b}
\]

Figure 2.10: One element of the normalized lattice structure

Furthermore, if an angle \( \theta_m \) is defined as the inverse sine of \( k_m \) (Equation 2.40a) then the cosine of \( \theta_m \) could be defined as Equation 2.40b. If Equation 2.40 is used for the normalized form, then each separate lattice element creates a rotation since each section takes the input vector and rotates it through an angle \( \theta_m \) to obtain a new vector. The significance of the rotation is also important for stability because it bounds the \( k \)-parameters to unity.

\[
\sin \theta_m = k_m \tag{2.40a}
\]

\[
\cos \theta_m = \sqrt{1 - k_m^2} \tag{2.40b}
\]
The filter parameters may now be taken as rotation angles \( \{\theta_0, \ldots, \theta_{M-1}\} \) plus the tap parameters \( \{\hat{\nu}_0, \ldots, \hat{\nu}_M\} \). The implementation of Equation 2.38 with the rotation blocks is shown in Figure 2.11.

![Diagram of filter structure](attachment:filter-diagram.png)

Figure 2.11: Details of the normalized lattice form using the rotation element block

The normalized lattice form was further studied by Regalia, who presented adaptive algorithms based upon the structure in [3, 44]. The normalized lattice structure is inherently limited to realizing stable and causal functions because all of the coefficients are bounded by unity, the roundoff noise accumulation is low irrespective of the poles of the filter, and the sensitivity to coefficient quantization is manageable.

**Cascade and Parallel Form**

An alternative to designing higher order direct form or lattice form filters would be cascade or parallel forms. Cascading can be accomplished by factoring the transfer function in terms of its poles \( p_k \) and zeros \( z_k \) (Equation 2.41). The smaller first and second-order filter sections are less sensitive to roundoff errors and weight quantization compared to higher order direct form filters.

\[
H(z) = \prod_{k=1}^{M} \frac{(z - z_k)}{(z - p_k)}
\] (2.41)
Parallel forms can be a bit more difficult to realize. Application of the Heaviside Formulas allows for the calculation of residues which are needed for each of the factored poles, and often leads to approximations of the intended filter. Also, when there are repeated poles, more complex methods are needed to be compute what are called residue chains. The general form can be seen in Equation 2.42.

\[
H(z) = \frac{R_1}{z - p_1} + \frac{\text{“Residue Chain”}}{(z - p_2)^2} + \frac{R_2}{z - p_2} + \cdots + \frac{R_M}{z - p_M} \tag{2.42}
\]

However, problems do arise because the mapping from the transfer function space to the parameter space is not unique, there will be many different combinations of weights that would map to the same squared error. The MSE with respect to the parameter value would no longer be convex. Thus, their nonuniqueness in the parameter space creates a distinct disadvantage toward adaptive implementations.

**Direct Form Notch Filter**

Suppose that the input signal contains a single sinusoid with some measurement noise \( \zeta(n) \), where the amplitude \((c_1)\), frequency \((\omega_1)\) and phase \((\phi_1)\) are unknown (Equation 2.43). The frequency response of the ideal notch filter would be 0 at \( \omega_1 \) and 1 everywhere else (Equation 2.44).

\[
u(n) = c_1 \sin(\omega_1 n + \phi_1) + \zeta(n) \tag{2.43}
\]

\[
|H(e^{j\omega})| = \begin{cases} 
0 & \omega = \pm \omega_1 \\
1 & \text{otherwise}
\end{cases} \tag{2.44}
\]

Of course this is not a rational function and cannot be implemented in practice. However, it can be shown that a second order filter can be a good approximation. The transfer function for the direct form filter can be seen in Equation 2.45, where \( \omega_0 \) is the notch frequency.

\[
\hat{H}(z) = \frac{1 + az + z^2}{1 + arz + r^2 z^2} \quad \omega_0 = \arccos \left( -\frac{a}{2} \right) \tag{2.45}
\]

The zeros of the filter are located at \( z = e^{\pm j\omega_0} \). Thus, the signal power of any sinusoid whose frequency is near \( \omega_0 \) will be reduced. The \( r \) parameter determines just how near a
signal must be to $\omega_0$ before it is affected. Made evident by the pole-zero map in Figure 2.12, as $r \to 1$ the poles of the system move closer to the zeros on the unit circle. Also, as $r \to 1$, the bandwidth of the filter decreases as seen in the frequency response in Figure 2.13.

![Pole-Zero Plot of Direct Form Notch Filter](image)

**Figure 2.12: Pole-Zero Plot of Direct Form Notch Filter**

**Lattice Form Notch Filter**

The lattice form notch filter presented in [3] is derived from the second order all-pass function, described by Equation 2.46. This representation implements the normalized structure using the rotation elements.

$$V(z) = \frac{\sin \theta_2 + \sin \theta_1 (1 + \sin \theta_2)z + z^2}{1 + \sin \theta_1 (1 + \sin \theta_2)z + \sin \theta_2 z^2}$$  \hspace{1cm} (2.46)

From the all-pass function, a complementary pair of transfer functions can be defined resulting in a notch filter (Equation 2.47a) and bandpass filter (Equation 2.47b). The corresponding signal flow graph is sketched in Figure 2.14. The transfer function of the notch filter is explicitly defined by Equation 2.48a.

$$H(z) = \frac{1}{2} [1 + V(z)]$$  \hspace{1cm} (2.47a)

$$G(z) = \frac{1}{2} [1 - V(z)]$$  \hspace{1cm} (2.47b)
Figure 2.13: Frequency Response of 2\textsuperscript{nd} Order Direct Form Notch Filter
As can be seen graphically by the flow graph and mathematically by the transfer function, the second order filter results in two rotation angle parameters: $\theta_1$ and $\theta_2$. These parameters can be related to their direct form counterparts: $\theta_1$ controls the notch frequency as the $a$ parameter does, and $\theta_2$ controls the notch bandwidth as the $r$ parameter does. These relationships are defined in Equation 2.48, where $\omega_0$ is the notch frequency, and $\beta$ specifically denotes the $-3$dB attenuation bandwidth.

$$\hat{H}(z) = \frac{1 + \sin \theta_2}{2} \frac{1 + 2 \sin \theta_1 z + z^2}{1 + \sin \theta_1 (1 + \sin \theta_2) z + \sin \theta_2^2} \quad (2.48a)$$

$$\omega_0 = \theta_1 + \frac{\pi}{2} \quad (2.48b)$$

$$\sin \theta_2 = \frac{1 - \tan(\beta/2)}{1 + \tan(\beta/2)} \quad \beta = 2 \tan^{-1} \left[ \frac{1 - \sin \theta_2}{1 + \sin \theta_2} \right] \text{[rad/s]} \quad (2.48c)$$

The zeros of the filter are controlled by $\theta_1$ and are located at $z = e^{\pm j\omega_0}$. The poles of the filter are controlled by $\theta_2$. Made evident by the pole-zero map in Figure 2.15, as $\theta_2 \to \pi/2$ the poles of the system move closer to the zeros on the unit circle. Also, as $\theta_2 \to \pi/2$ the bandwidth of the filter narrows as seen by the frequency response plotted in Figure 2.16.

Comparing the frequency response of the direct form realization with that of the lattice form realization, it can be seen that the lattice form has more favorable characteristics. The lattice form provides a uniform response of 1 (0 dB) in the area outside of the local vicinity of $z = e^{\pm j\omega_0}$, while the direct form filter does not. The response across the frequency spectrum of the direct form filter also varies based on the parameter value.
The ideal notch filter is intended to only affect the undesirable contents of the input signal while leaving the remaining frequency components unaltered. The lattice form realization will achieve this while the direct form will not. If the notch bandwidth parameter is to be tuned by adaptive processes, it is also desirable to have consistent behavior outside of the notch region as the parameter value changes. It is obvious that only the lattice form implementation can achieve this.

**Multiple Sinusoid Notch Filter**

While identifying a single sinusoid at a particular frequency is a good start, signals are typically composed of multiple sinusoids, as described by Equation 2.49. The background noise is denoted by $\zeta$, and the objective for an adaptive filter is to determine the signal frequencies $\omega_i$ of the sinusoids with the corresponding phase ($\phi_i$) and amplitude ($c_i$).

\[
u(n) = \sum_{i=1}^{M} c_i \sin(\omega_i n + \phi_i) + \zeta(n)\]  

(2.49)

In order to design a filter to identify and remove multiple sinusoids, two techniques can
Figure 2.16: Frequency response of the lattice form notch filter
be employed. One implementation would be to extend the filter to a multi-notch configuration, choosing higher order polynomials for the transfer function. The number of poles and zeros would equal the number of sinusoids composing the signal of interest. Another implementation would be to cascade the basic 2\textsuperscript{nd} order notch filter sections, in order to remove each of the frequency components one by one.
3. Testing

3.1 Test Cases

Disturbances in power systems can be produced from a wide range of sources: from an increase in load during warm summer months to geomagnetic storms that cause large fluctuations in the earth’s magnetic field and difference in potential between points of ground. Power systems engineers have designed mechanisms to handle disturbances effectively in order to provide consistent operating conditions. Knowledge of those mechanisms is equally as important as the disturbances themselves in order to accurately model the possible fluctuations of the power line interference that the filter may encounter. Appropriately modeling the PLI will allow for more realistic and robust testing conditions.

The nature of the interference depends on the environment. The amount of interference due to magnetic induction depends upon the area of the loop enclosed by the two electrode leads connecting the ECG recording equipment and the patient [26]. If the leads were twisted leading up to the body, the effective conductive loop area would decrease and thus the interference would also decrease. Also, the induced potential depends upon the orientation of the loop with respect to the magnetic field. More current is induced in the loop when the surface area of the loop is perpendicular to stronger magnetic field lines. Therefore, the amount of interference that could be reduced depends upon decisions made by the operator.

The environment presented in [26] was that of an ECG machine immediately adjacent to an air-conditioning unit that was turned on. The magnetic flux density (B-field) was calculated to be $\sim 3.2 \times 10^{-7}$ Wb/m$^2 \equiv 0.32 \, \mu$T. This resulted in an interference of $\sim 120 \, \mu$V for a loop area of 1 m$^2$. The environment presented in [25] was that of an AED in a train station in close proximity to the high voltage power lines used by the train. The measured magnetic field varied from 0.7 to 3.7 $\mu$T, depending on orientation of people near
the patient and whether a train was present. The maximum B-field was approximately an order of magnitude larger than the environment presented in [26]. Using the values obtained from [26] as a guideline, the maximum interference for this environment is assumed to be \( \sim 1.3875 \text{ mV} \) for a loop area of \( 1 \text{ m}^2 \). Related research has assumed similar levels of interference amplitude for this environment (\( 1 \text{ mV} \) [24] and \( 2 \text{ mV} \) [47]).

Therefore, this testing assumes two environments: internal and external, described in Table 3.1. The internal environment assumes a controlled setting with trained operators using the recording equipment. The resulting interference amplitude is \( 100 \mu\text{V} \). The external environment assumes an uncontrolled setting with untrained operators (i.e. users of a public access defibrillator such as an AED). The resulting interference amplitude is \( 1 \text{ mV} \). Both environments assume that the orientation of the patient with respect to the B-field remains constant.

<table>
<thead>
<tr>
<th>Environment</th>
<th>Baseline Interference Amplitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal</td>
<td>( 100 \mu\text{V} )</td>
</tr>
<tr>
<td>External</td>
<td>( 1 \text{ mV} )</td>
</tr>
</tbody>
</table>

A set of test cases was derived for all of the foreseeable disturbances (Section 2.2.3) that may be presented in the power line interference. The signal source for each of the test cases was a sinusoidal. An overview of each test case can be seen in Table 3.2. Test Case 0: Baseline, provides a baseline to compare the rest of the testing results against. Test Case 6: Cumulative, contains all the disturbance variables from tests 1-5 to form a “worst case scenario”. Test Case 7: B-field Alterations, presents the scenario described in [25], when a train went by and altered the B-field. A detailed description of each test case with PLI waveforms is provided in Sections 3.1.1-3.1.8. Test cases 0-6 are applied to both environments, internal and external. Test case 7 is only applicable for the external environment.

According to Huhta and Webster in [26], if the interference is reduced to 1 percent of the
<table>
<thead>
<tr>
<th>Test Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>0 Baseline</strong></td>
<td>no variations, 60 Hz sine wave</td>
</tr>
<tr>
<td><strong>1 Random Variables</strong></td>
<td></td>
</tr>
<tr>
<td>a. Amplitude</td>
<td>can vary ±5% (Range A)</td>
</tr>
<tr>
<td>b. Frequency</td>
<td>can vary ±0.03Hz</td>
</tr>
<tr>
<td>c. Gaussian Noise</td>
<td>SNR = 20 dB</td>
</tr>
<tr>
<td><strong>2 Surges</strong></td>
<td>short duration (&lt; 0.1 s) spike in voltage</td>
</tr>
<tr>
<td><strong>3 Voltage Variations</strong></td>
<td></td>
</tr>
<tr>
<td>a. Sag/Swell</td>
<td>momentary (2 – 3 s) voltage outside of Range A</td>
</tr>
<tr>
<td>b. Under/Overvoltage</td>
<td>extended (&gt; 3 s) voltage outside of Range A</td>
</tr>
<tr>
<td><strong>4 Harmonics</strong></td>
<td></td>
</tr>
<tr>
<td>a. Acceptable</td>
<td>low distortion factor (5%)</td>
</tr>
<tr>
<td>b. Severe</td>
<td>high distortion factor (≥ 20%)</td>
</tr>
<tr>
<td><strong>5 Underfrequency</strong></td>
<td></td>
</tr>
<tr>
<td>a. Gradual Decline</td>
<td>generation deficiency of &lt; 10%</td>
</tr>
<tr>
<td>b. Rapid Decline</td>
<td>generation deficiency ≥ 25%</td>
</tr>
<tr>
<td><strong>6 Cumulative</strong></td>
<td>culmination of tests 1-5</td>
</tr>
<tr>
<td><strong>7 B-field Alterations</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>order of magnitude change in B-field</td>
</tr>
</tbody>
</table>
desired signal, it will not significantly degrade the recording. Since a typical ECG potential is along the order of 1 mV, a test will be deemed successful if the interference is reduced to 10 µV. This is reaffirmed as an appropriate value in [25] with respect to an automated external defibrillator (AED) correctly identifying asystole. An AED will generally define a recording as asystole when a 2-3 s segment amplitude is less than 100 µV, thus the interference needs to be less than this value.

3.1.1 Test Case 0: Baseline

The Baseline test provides the environment for nominal operating conditions. The ECG recording equipment is in the presence of electromagnetic fields generated by an AC power source. There are no disturbances present in the PLI and the amplitude remains constant. The resulting interference waveform can be seen in Figure 3.1. The results of this test can be used to compare against other tests.

![Figure 3.1: No disturbances are present in the Baseline test case](image)

3.1.2 Test Case 1: Random Variables

Random signals and noise are present in any practical engineering problems. Random variables are an important consideration when designing a signal processing systems. Three random variables were considered: amplitude, frequency and noise. The amplitude random
variable sample space was ±5% of the nominal amplitude. The frequency random variable sample space was ±0.03 Hz. The noise random variable was assumed to be Additive White Gaussian in nature and provided a signal to noise ratio of 20 dB. The resulting interference waveforms can be seen in Figure 3.2.

### 3.1.3 Test Case 2: Surges

There are three disturbances in the interference, allowing the test to present surges of varying magnitude and duration. The first surge occurs at \( t = 1 \) s with a duration of 1 ms, resulting in a disturbance magnitude of 1 mV. The second surge occurs at \( t = 1.3 \) s with a duration of 500 \( \mu \)s, resulting in a disturbance magnitude of 300 \( \mu \)V. The third surge occurs at \( t = 1.7 \) s with a duration of 5 ms, resulting in a disturbance magnitude of 700 \( \mu \)V. The resulting interference waveform can be seen in Figure 3.3.

### 3.1.4 Test Case 3: Voltage Variations

Voltage variations are important in determining the number of necessary bits to be used in the VHDL models. This test presents two cases, one for momentary fluctuations in voltage (sag/swell), and one for an extended decrease/increase in voltage (under/overvoltage). The voltage sag/swell duration is 2 s, while the under/overvoltage lasts the duration of the test. The resulting waveforms can be seen in Figure 3.4.

### 3.1.5 Test Case 4: Harmonics

As harmonics introduce themselves in the PLI, additional filter structures must be cascaded to properly remove the additional frequency components of the PLI. This testing assumes only 2 harmonics are distorting the fundamental.

The IEEE Standard 519-1992 [36] defines total harmonic distortion in a system based on a distortion factor (DF, Equation 2.4): from acceptable (< 5%) to severe (≥ 20%). This test presents two cases, one for acceptable harmonic levels, and one for severe harmonic levels.
Figure 3.2: Random variables
Figure 3.3: TC2 – A surge presents itself in the PLI as a short duration spike in voltage

Figure 3.4: Voltage variations outside of Range A as defined by the ANSI C84.1 standard

(a) TC3a – A momentary decrease/increase in voltage is known as a Sag/Swell

(b) TC3b – An extended decrease/increase in voltage is known as an Under/Overvoltage
Using Equation 2.4, amplitude values for the two harmonic components were determined, resulting in a **DF** of 4.24% for the acceptable case, and 20% for the severe case. The individual harmonic component values can be seen in Table 3.3, and the resulting waveforms can be seen in Figure 3.5.

### Table 3.3: Harmonic component values

<table>
<thead>
<tr>
<th>Component</th>
<th>Percent of fundamental ((f_0))</th>
</tr>
</thead>
<tbody>
<tr>
<td>4a. Acceptable</td>
<td>(A_1 (2f_0)) (3%)</td>
</tr>
<tr>
<td></td>
<td>(A_2 (3f_0)) (3%)</td>
</tr>
<tr>
<td>4b. Severe</td>
<td>(A_1 (2f_0)) (17.885%)</td>
</tr>
<tr>
<td></td>
<td>(A_2 (3f_0)) (8.944%)</td>
</tr>
</tbody>
</table>

#### 3.1.6 Test Case 5: Underfrequency

Underfrequency conditions can be widespread or can occur in isolated areas referred to as islands. It is important for the affected area to reestablish a load-generation balance quickly. This is accomplished by first arresting the frequency decline, and then entering a period of system restoration.

For gradual declines in frequency (Figure 3.6(a)) manual corrective actions are taken by utilizing all operating reserves and if necessary shedding loads. If corrective actions cannot be implemented to restore balance in time and the frequency has rapidly declined past the nominal set point of 59.3 Hz, automatic load shedding programs are tripped into action (Figure 3.6(b)). The goal of automatic underfrequency load shedding (UFLS) programs is to arrest the frequency decline and return the frequency to nominal operating conditions by shedding loads within the low frequency area. Additional loads are continuously shed until the system has been restored.

For this testing, the UFLS procedures set forth by the Northeast Power Coordinating Council (NPCC) are assumed. The procedures are described in the NPCC Reference Manual [40] and briefly explained in Section 2.3.4. The two trip points (59.3 Hz, 58.8 Hz) are labeled in Figure 3.6.
Figure 3.5: As the distortion factor increases, each additional harmonic increasingly degrades the ECG signal
(a) TC5a – A generation deficiency of 10% drops the frequency to a low of 59.3 Hz

(b) TC5b – A generation deficiency of 25% drops the frequency to a low of 58.6 Hz

Figure 3.6: As the frequency decline increases, the system must take more drastic measures to restore a balanced load-generation
3.1.7 Test Case 6: Cumulative

The cumulative testing consists of the worst case conditions presented in test cases 1-5. At $t=1$ s, the system undergoes a rapid decline in frequency, resulting in automatic load shedding to arrest the decline and restore the system frequency. Harmonics are present and produce a severe harmonic distortion factor of 20%. There is a 10 s long overvoltage beginning at $t=1$ s, and a 10 s long undervoltage beginning at $t=15$ s. A surge occurs at $t=1$ s with a duration of 1 ms. Random variables are added for amplitude, frequency and noise. Figure 3.7 presents the resulting interference signal.

![Cumulative interference signal](image)

(a) Cumulative interference signal

![Cumulative interference signal frequency component](image)

(b) Cumulative interference signal frequency component

Figure 3.7: TC6 – Cumulative affects from the disturbance contributions outlined in Test Cases 1-5 on the interference signal
3.1.8 Test Case 7: B-field Alterations

Large alterations in the magnetic field were observed in the environment presented in [25]. The magnitude of the field changed depending on whether the train was present in the terminal or not. The magnitude of the \( B \)-field corresponds directly to the amplitude of the observed interference. Therefore, an order of magnitude drop in the \( B \)-field will result in an order of magnitude drop in the interference signal amplitude. The resulting waveform can be seen in Figure 3.8. This test will only be performed for the external environment condition.

![Figure 3.8: TC7 – A change in the magnetic flux density results in a change in the interference amplitude](image-url)

Figure 3.8: TC7 – A change in the magnetic flux density results in a change in the interference amplitude
3.1.9 Test Case: Tracking

To perform tracking analysis on the adaptive notch filter, a linear chirped sinusoid was provided as an input [48]. An expression for the linear chirp signal can be seen in Equation 3.1a, where $\beta$ is the chirp rate, and $f(n)$ is the input frequency. As the chirp rate is increased, the algorithm has fewer samples to use in order to correctly adjust the weights of the filter. The frequency was linearly varied from 60 Hz to 59.97 Hz. The frequency delta of 0.03 Hz was chosen based on the power systems research (Section 2.3).

\begin{align*}
    u(n) &= \sin \phi(n) \\
    \phi(n) &= \phi(n - 1) + 2\pi f(n) \\
    f(n) &= f(0) + \beta \cdot n \\
    &= f(n - 1) + \beta
\end{align*}
4. Algorithm Development

The development method used to design the adaptive notch filter can be seen in Figure 4.1. The method is similar to the software development method, the waterfall model [49].

![Algorithm Development Diagram](image)

Figure 4.1: The algorithm development method used feedback from each stage

The first step involves researching the problem and analyzing the various approaches others have taken to solve the problem. The various update algorithms and filter structures uncovered by the research (Section 2.4) were modeled in MATLAB to further analyze the performance and characteristics. This was accomplished by running the algorithms through testing verification, and a greater understanding of the theory behind the various algorithms and structures was achieved. This process resulted in the second order normalized lattice form notch filter structure [3] and the nonparametric variable step size normalized least mean square update algorithm [11] to be chosen as the basis for the design and implementation.

After the structure and algorithm were chosen, the individual parameters were adjusted to optimize the performance around a set of generic requirements. The details of this process are explained in Section 4.1. The adjustments were then cycled backed to the first step where they were implemented in MATLAB and any research was done relating to the changes. The updated implementation was then tested and more adjustments were made.
as necessary. The development cycle was repeated until an acceptable design was reached that met the requirements.

4.1 Proposed Notch Filter Design

The proposed adaptive notch filter was designed around a set of requirements. The filter requirements were as follows: narrow notch bandwidth, deep null depth, fast rate of convergence, low misadjustment, high frequency of operation and minimization of resource utilization. Many of these requirements counteract one another, so decisions were made to balance the various tradeoffs.

A tradeoff exists between the notch filter bandwidth and the depth of the null. As the notch bandwidth decreases, the depth of the null is adversely affected. Also, the impulse response of the filter is longer in the time domain as the notch bandwidth decreases. This results in more iterations needed to reduce the interference. These tradeoffs can be seen in the performance surface. Multiple views can be seen in Figure 4.2. The views are adjusted by the azimuthal angle ($\phi$) and the vertical elevation ($EL$), both in degrees, represented as $view(\phi, EL)$ in the subfigure captions. The performance surface represents the amount of attenuation in dB after 250 iterations for a range of weight values on a sinusoidal input that represents the maximum amplitude interference.

The notch bandwidth is controlled by the $\theta_2$ weight, and is constant in the proposed design. The notch bandwidth chosen for this design was the smallest value that would still effectively reduce the interference signal in as few iterations as possible. It is represented by the thick black line in the figures. The minimum attenuation is represented by a pass/fail (P/F) threshold plane in the figures. It represents the interference signal power being reduced to 5E-5 mV$^2$.

A tradeoff exists between the rate of convergence and the misadjustment after convergence. For larger adaptation step-sizes, the algorithm will converge faster, but also have a larger misadjustment after convergence. A smaller adaptation step-size allows for a smaller misadjustment to be maintained after convergence, but the rate of convergence is slower.
Figure 4.2: Several different views of the Lattice-form adaptive notch filter performance surface with input = 2.8 sin(2π60t)
A normalized variable step-size was chosen for this design to allow for a fast rate of convergence and low steady-state misadjustment. The step-size would be dependent upon the input signal variance ($\sigma_x^2$) and the output error signal variance ($\sigma_e^2$). The output error will decrease as the notch center frequency approaches that of the interference signal. Thus the adaptation step-size will decrease as the output error decreases in comparison to the input. The variable step-size is updated according to Table 4.1. These values were determined empirically by analyzing the performance of the filter for different transition conditions and assignment values.

<table>
<thead>
<tr>
<th>Transition</th>
<th>Condition</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$\sigma_e^2 &gt; \frac{3\sigma_x^2}{4}$</td>
<td>$\mu = \frac{5\mu}{4}$</td>
</tr>
<tr>
<td>B</td>
<td>$\sigma_e^2 &lt; \frac{3\sigma_x^2}{4}$</td>
<td>$\mu = \frac{\mu}{2} + \frac{\mu_{\text{max}}}{32}$</td>
</tr>
<tr>
<td>C</td>
<td>$\sigma_e^2 &lt; \frac{\sigma_x^2}{4}$</td>
<td>$\mu = \frac{\mu}{2} + \frac{\mu_{\text{max}}}{64}$</td>
</tr>
<tr>
<td>D</td>
<td>$\sigma_e^2 &lt; \frac{3\sigma_x^2}{4}$</td>
<td>$\mu = \frac{\mu}{2} + \frac{\mu_{\text{max}}}{32}$</td>
</tr>
<tr>
<td>E</td>
<td>$\sigma_e^2 &gt; \frac{3\sigma_x^2}{4}$</td>
<td>$\mu = 2\mu$</td>
</tr>
</tbody>
</table>

In order to reduce FPGA resource utilization and have a high frequency of operation, the logic needs to be simplified. By using approximations and decreasing bit lengths, the number of logic gates needed for implementation is reduced. Approximations and quantization could change the behavior of the filter from that of the ideal design, resulting in a decrease in performance. These effects were studied in order to reduce resource utilization but still maintain performance.

### 4.1.1 Approximations and Quantization

The multiplication or division of a number by an integer power-of-two is a very simple process in binary arithmetic. The computationally expensive multiplication or division
can be replaced with a shift operation. Simplifications like these come at the expense of approximation error. When multiplications and divisions cannot be avoided, decreasing the number of bits the arithmetic operators need will result in reduced resource utilization. These adjustments come at the expense of quantization errors.

An approximation could be made with regard to the maximum step-size. It is important to calculate the maximum step-size value to ensure the stability of the update algorithm. In order to do this, the eigenvalues of the autocorrelation matrix needed to be found. For a 2\textsuperscript{nd} order filter, the autocorrelation matrix is 2x2 symmetric, resulting in two real eigenvalues. The maximum eigenvalue (Equation 4.1a) is used to determine the maximum step-size (Equation 4.2a). Plotting $\sigma_x^2$ and $\lambda_{\text{max}}$ for a range of interference signal amplitudes, a simple correlation was found. First the trendlines were found for each (Figure 4.3(a)) and the ratio between the two was found. Using this ratio, a new approximation could be made for $\lambda_{\text{max}}$ (Equation 4.1b). The resulting approximation can be seen in Figure 4.3(b). The approximation is also shown to give an accurate representation of $\mu_{\text{max}}$ (Figure 4.3(c)), now defined by (Equation 4.2b). The impact of this approximation on the filter’s ability to adapt was negligible.

\[
\lambda_{\text{max}} = \max(eig(R_{xx})) \quad (4.1a)
\]
\[
\hat{\lambda}_{\text{max}} = 1.896986\sigma_x^2 \quad (4.1b)
\]
\[
\hat{\lambda}_{\text{max}} \approx 2\sigma_x^2
\]

\[
\mu_{\text{max}} = \frac{1}{2\lambda_{\text{max}}} \quad (4.2a)
\]
\[
\hat{\mu}_{\text{max}} \approx \frac{1}{2(2\hat{\sigma}_x^2)} \quad (4.2b)
\]

The algorithm is stable as long as the step-size is less than this maximum value. Further simplifications can be made if a factor of $1/8$ is introduced to the calculation. This would cause the factor in front of the variance estimate ($\hat{\sigma}_x^2$) to be 32. This factor can then
(a) Power series trendlines for $\lambda_{\text{max}}$ and $\sigma_x^2$

(b) Approximation for $\lambda_{\text{max}}$ using scaled $\sigma_x^2$

(c) Approximation for $\mu_{\text{max}}$ using $\lambda_{\text{max}}$ approximation

Figure 4.3: Approximations can be made when calculating the maximum step-size
be absorbed by the variance calculation, and cancel out a divide by 32 in its calculation. Therefore, the final equation used to calculate the maximum step-size can be seen in Equation 4.3. This simplification also had minimal impact on the filter’s ability to respond to a changing environment. If the maximum step-size value was decreased too much, the rate of convergence would be adversely affected. However, there would be no benefit to decrease the value any further, as the sole purpose of this simplification was to eliminate a division and a multiplication.

\[
\hat{\mu}_{\text{max}} \approx \frac{1}{32(\hat{\sigma}_x^2)}
\]  

(4.3)

Three main sources of quantization error are introduced by the implementation of digital filters: input quantization, weight quantization, and quantization in arithmetic operations [50]. A GUI tool was created in MATLAB to visualize the quantization effects of the proposed ANF design. Examples of the tool can be seen in Figure 4.4. The input signal amplitude and quantization parameters (word length and fraction length) could be changed using sliders, and the plot would update based on the new user inputs automatically. A set of buttons on top was used to switch between different parameters. The quantization values associated with each parameter were stored as variables, allowing the user to set different values for each parameter because many parameters are used in the calculation of other parameters. The tool was created to assist in sorting out these dependencies.

Figure 4.4(a) gives an example of the tool being used to show the input quantization effects. Also displayed by the tool is how the roundoff error effects calculations further along in the algorithm like the input variance (Figure 4.4(b)) and step-size (Figure 4.4(c)), and how those parameters quantization values could compound the errors.

The variance estimator needed the most number of bits in the design. The 24 bits are necessary to correctly calculate the variance over the entire input dynamic range. This estimate is then used to calculate the maximum step-size. For a given \( \mu_{\text{max}} \), the adaptation step-size range is relatively small, and can thus be represented with fewer bits. However, \( \mu_{\text{max}} \) has a wide range of possible values. To maintain the small word length, the fraction length needed to be variable. By adjusting the input amplitude, the correct fraction length
(a) Input signal Analog to Digital converter (ADC) quantization effect

(b) Input signal variance parameter accumulated quantization effects

(c) $\mu_{\text{max}}$ (left) and $\mu_{\text{min}}$ (right) parameter accumulated quantization effects. The ! is an overflow indicator. The number is the value of $F_{\text{state}}$.

Figure 4.4: Quantization tool developed to analyze quantization effects
could be found over the input dynamic range. The fraction length is dependent on the input signal power, as indicated by Equation 4.4. The GUI tool was used to determine the fraction length values needed across the dynamic range. The thresholds were chosen in such a way as to limit the effect of changing the fraction length; either value of $F$ would produce the same step-size value. The control circuitry can be seen in Figure 4.5. The smaller word length allows for a smaller multiplier size to be used.

$$
F = \begin{cases} 
2 & 10 \mu V \leq A < 31 \mu V \\
4 & 31 \mu V \leq A < 62 \mu V \\
6 & 62 \mu V \leq A < 124 \mu V \\
8 & 124 \mu V \leq A < 248 \mu V \\
10 & 248 \mu V \leq A < 494 \mu V \\
12 & 494 \mu V \leq A < 987 \mu V \\
14 & A \geq 987 \mu V \\
F & otherwise 
\end{cases} \quad (4.4)
$$

Figure 4.5: Control logic for variable fraction length of step-size

A variable fraction length was also used for internal lattice values. The same thresholds were used so that the control circuitry would be reused.
4.1.2 Final Design Parameters

Table 4.2: Proposed lattice form adaptive notch filter characteristics

Performance characteristics:
\( F_s = 1 \text{ kHz} \)
\( \theta_2 = .36\pi, \quad \therefore \beta_{-3 \, dB} = 15.8908 \text{ Hz} \)
\( D_{null} = -54 \text{ dB} \)

Update algorithm computations:
\[
\hat{\sigma}^2(n) = \frac{1}{32} \sum_{i=0}^{31} x_{n-i}^2
\]
\[
\hat{\sigma}^2(n + 1) = \frac{1}{32} \left[ \hat{\sigma}^2(n) - x_{n-32+1}^2 + x_{n+1}^2 \right]
\]
\[
\hat{\mu}_{\max} = \frac{1}{32(\hat{\sigma}^2)}
\]

Filter computations:
\[
\begin{bmatrix} g_1 \\ w \end{bmatrix} = \begin{bmatrix} \cos \theta_2 & -\sin \theta_2 \\ \sin \theta_2 & \cos \theta_2 \end{bmatrix} \begin{bmatrix} u(n) \\ x_2(n) \end{bmatrix}
\]
\[
\hat{y}(n) = \frac{1}{2} [u(n) + w(n)]
\]
\[
\theta_1(n + 1) = \theta_1(n) - \mu\hat{y}(n)x_1(n)
\]
\[
\begin{bmatrix} x_1(n + 1) \\ x_2(n + 1) \end{bmatrix} = \begin{bmatrix} \cos \theta_1(n + 1) & -\sin \theta_1(n + 1) \\ \sin \theta_1(n + 1) & \cos \theta_1(n + 1) \end{bmatrix} \begin{bmatrix} g_1 \\ x_1(n) \end{bmatrix}
\]

Bit quantizations:

Input ADC (\( u \)): [12 9] range: 5 \( \mu \text{V} \rightarrow 4 \text{ mV} \)
\( \sigma^2 \): [22 14] range: 5 \( \mu \text{V} \rightarrow 2.8 \text{ mV} \)
\( \mu \): [12 \( F \)] \( F=\{2,4,6,8,10,12,14\} \)
\( \hat{y} \): [12 9]

filter states (\( g_n, x_n \)): [12 \( F \)] \( F=\{11,11,10,10,9,8,7\} \)
\( \mu \cdot x_1 \): [16 13]
\( (\mu \cdot x_1) \cdot \hat{y} \): [16 18]
\( \theta_1, \theta_2 \): [16 13]
CORDIC: [16 14]
4.2 MATLAB Simulation Results

After characterizing the interference, appropriate test cases were made. These test cases provided a realistic environment to properly assess the performance of the adaptive filter. The performance of an adaptive filter is evaluated based on its transient behavior, steady-state behavior and stability.

To measure the transient behavior, the rate of convergence is used. The convergence rate is the number of iterations required to converge “close enough” to the optimal mean-square-error. Knowing the expected weight values \textit{a priori}, this was done by visual inspection of the learning curve.

To measure the steady-state behavior, the steady-state misadjustment and MSE is used. The misadjustment (Equation 4.5) is a quantitative measure of the amount by which the excess MSE exceeds the minimum mean square error ($\xi_{\text{min}}$).

\[ M = \frac{E[\xi_k - \xi_{\text{min}}]}{\xi_{\text{min}}} = \frac{\text{Excess MSE}}{\xi_{\text{min}}} \quad (4.5) \]

To measure stability, the numerical robustness after quantization errors is used. The quantized filter should behave close to that of the non-quantized (analog) filter. The quantized filter is the realizable filter implementation in hardware.

The test cases outlined in Section 3.1 were simulated in MATLAB using the proposed notch filter design. The results can be seen in Table 4.3. The results include the steady state MSE ($\xi$), the pass/fail (P/F) indicator and the convergence rate (in number of iterations). Testing was performed for both the internal and external testing environments.

The P/F indicator is a measure of the MSE against a threshold. The threshold is the signal power of a 10 $\mu$V interference signal (5E-5 mV$^2$). This threshold value is based on the recommendation that the interference should be reduced to 1 percent of the desired signal[26]. The indication for success (✓) occurs if the MSE is less than the P/F threshold, and failure (✗) otherwise.

The following subsections explain the results of each test case in more detail. Figures are included to display the learning curve and error output of the filter. All of the tests were

62
Table 4.3: Lattice form Adaptive Notch Filter MATLAB simulation results

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Internal Environment</th>
<th>External Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Conv. Rate</td>
<td>ξ</td>
</tr>
<tr>
<td>0</td>
<td>270</td>
<td>3.33786E-7</td>
</tr>
<tr>
<td>1a</td>
<td>270</td>
<td>3.38912E-6</td>
</tr>
<tr>
<td>1b</td>
<td>270</td>
<td>2.41308E-5</td>
</tr>
<tr>
<td>1c</td>
<td>260</td>
<td>4.73738E-5</td>
</tr>
<tr>
<td>2</td>
<td>150</td>
<td>2.86102E-7</td>
</tr>
<tr>
<td>3a</td>
<td>swl: 32/32</td>
<td>2.67029E-7</td>
</tr>
<tr>
<td>3b</td>
<td>sag: 49/94</td>
<td>4.67300E-7</td>
</tr>
<tr>
<td>4a</td>
<td>B1: 267</td>
<td>9.62353E-6</td>
</tr>
<tr>
<td>4b</td>
<td>B2: off</td>
<td>–</td>
</tr>
<tr>
<td>5a</td>
<td>B3: off</td>
<td>–</td>
</tr>
<tr>
<td>5b</td>
<td>t1: –</td>
<td>3.82376E-7</td>
</tr>
<tr>
<td>6</td>
<td>t2: –</td>
<td>4.04090E-7</td>
</tr>
<tr>
<td>5b</td>
<td>t3: –</td>
<td>3.70264E-7</td>
</tr>
<tr>
<td>6</td>
<td>t1: –</td>
<td>5.59845E-6</td>
</tr>
<tr>
<td>7</td>
<td>t2: –</td>
<td>3.87102E-7</td>
</tr>
<tr>
<td>6</td>
<td>t3: –</td>
<td>3.70026E-7</td>
</tr>
<tr>
<td>7</td>
<td>t4: –</td>
<td>–</td>
</tr>
<tr>
<td>7</td>
<td>t1:</td>
<td>–</td>
</tr>
<tr>
<td>7</td>
<td>t2:</td>
<td>–</td>
</tr>
<tr>
<td>7</td>
<td>t3:</td>
<td>–</td>
</tr>
<tr>
<td>7</td>
<td>t4:</td>
<td>–</td>
</tr>
</tbody>
</table>

63
performed with an “analog” and “quantized” version of the proposed filter. The “analog” results represent the simulation without quantization effects, while the “quantized” results include quantization effects. The following nomenclature is consistent with all figure results: “analog” is represented with a dashed blue line, “quantized” is represented with a solid red line. Any other information present in a figure will be indicated accordingly by the figure’s legend.

4.2.1 Test Case 0: Baseline

The input dynamic range is 10 µV to 2.8 mV. This range was tested in 10 µV increments using a baseline interference signal. The results can be seen in Figure 4.6. The proposed filter design was able to successfully reduce the interference below the P/F threshold across the entire dynamic range. However, the minimum MSE ($\xi_{\text{min}}$) was found to be on the order of E-20, resulting in a large misadjustment for the filter. This is due to quantization error. Because the filter is designed to have a narrow notch bandwidth ($\beta \approx 0.1$ rad), the quantization errors become more significant. The quantization error will decrease as more bits are used in the design.

![Figure 4.6: The resulting steady-state MSE across the input dynamic range for a baseline interference signal](image)

Figure 4.6: The resulting steady-state MSE across the input dynamic range for a baseline interference signal
4.2.2 Test Case 1: Random Variables

Test case 1 consisted of using three random variables: amplitude (TC1a), frequency (TC1b) and AWGN (TC1c). The resulting steady-state MSE was calculated after the filter was initially able to converge. The proposed filter was able to sufficiently reduce the interference for each case for the internal environment. The filter failed to reduce the interference below the P/F threshold for all three cases for the external environment. For test case 1b, the frequency of the interference would randomly change on 1 second intervals. Initially the error would be large, but the filter would quickly adjust it’s null, and the interference would again be sufficiently reduced. The test fails to meet the passing requirements because the MSE calculation includes these initial large errors. The test case 1b results can be seen in Figure 4.7. The “reference” is the input interference frequency.

![Graphs showing weight values and output error for Test Case 1](image)

Figure 4.7: TC1b – Adaptive filter adjusts as the frequency changes randomly in the internal (left) and external (right) environments

4.2.3 Test Case 2: Surges

Test case 2 included 3 surges of varying length and amplitude, but only the surge with the longest length (5 ms) was present after sampling the input signal. The results can be seen
in Figure 4.8. The proposed filter design was able to successfully handle the disturbance in both environments, and remain stable.

Figure 4.8: TC2 – Adaptive filter correctly adjusts to a surge disturbance in the internal (left) and external (right) environments

### 4.2.4 Test Case 3: Voltage Variations

Test case 3 consisted of voltage variation disturbances that were momentary (TC3a) and extended (TC3b). The momentary disturbance (sag/swell) lasted 2 seconds. The convergence rate was recorded for both instances when the disturbance presented itself, and when it dispersed. The MSE was calculated during the disturbance, after the filter converged. The results of test case 3a can be seen in Figures 4.9-4.10. The figures include the learning curve and output error when the disturbance presents itself and when it disperses 2000 iterations (2 seconds) later. The proposed filter design was able to successfully handle each disturbance in both environments, and remain stable.

### 4.2.5 Test Case 4: Harmonics

Test case 4 consisted of harmonic disturbances that resulted in acceptable harmonic distortion (TC4a) and severe harmonic distortion (TC4b). In order to remove the fundamental interference signal and the two harmonics that were present, 3 filters were cascaded (B1,
Figure 4.9: TC3a:sag – Adaptive filter correctly adjusts to a sag voltage variation disturbance in the internal (left) and external (right) environments.

Figure 4.10: TC3a:swl – Adaptive filter correctly adjusts to a swell voltage variation disturbance in the internal (left) and external (right) environments.
B2 and B3). As can be seen in Table 4.3, the cascaded filters were able to successfully reduce the interference below the P/F threshold in both environments. A power saving feature was also observed during the internal environment tests. If the input signal power is calculated to be less than that of a 10 µV signal, the filter would not process the signal and pass it through. In this instance, the filter is said to be “off”. This feature was correctly observed in the internal environment. The results of test case 4a and 4b can be seen in Figures 4.11-4.12.

Figure 4.11: TC4a – Cascaded adaptive filters correctly adjust to an acceptable level of harmonics disturbance in the internal (left) and external (right) environments

Figure 4.12: TC4b – Cascaded adaptive filters correctly adjust to a severe level of harmonics disturbance in the internal (left) and external (right) environments
4.2.6 Test Case 5: Underfrequency

Test case 5 presented underfrequency conditions where the initial drop in frequency is gradual (TC5a) and rapid (TC5b). Three time periods were examined to model the filter’s behavior throughout the disturbance. For each of the time periods the MSE was calculated. Time period $t_1$ represented the time during the initial drop in frequency before corrective action was taken at the utility. Time period $t_2$ represented the time during system recovery. Time period $t_3$ represented the time when the system regained stability and the frequency value was near nominal. The results of test case 5a and 5b can be seen in Figures 4.13-4.14. The proposed filter design was able to successfully handle the disturbance in both environments, and remain stable. The filter did have more trouble correctly adjusting to the rapid decline (TC5b) in frequency than the gradual decline (5a) in frequency. This is evident from the MSE values during $t_1$ presented in Table 4.3.

4.2.7 Test Case 6: Cumulative

The cumulative testing presented a disturbance made up of the worst-case disturbances from test cases 1-5. Three cascaded blocks were utilized to remove each of the harmonic components. The results can be seen in Figure 4.15. The filter was unable to sufficiently reduce the cumulative interference below the P/F threshold in either environment, though it did remain stable. The third block had the most difficult time tracking the 2$^{nd}$ harmonic. This is a result of the first two blocks not sufficiently reducing the interference at the fundamental and 1$^{st}$ harmonic frequencies. The signal power of the 2$^{nd}$ harmonic is small compared to the fundamental ($\sim8.9\%$ of fundamental). Therefore, as the null of B3 approached the 2$^{nd}$ harmonic, the decrease in output error was not significant compared to the input. The output error variance did not decrease to less than 25% of the input variance, thus restricting the algorithm from decreasing the adaptation step-size to $\mu_{\text{min}}$. The larger step-size caused the filter’s weight values to change more substantially.
Figure 4.13: TC5a – Adaptive filter correctly adjusts to a gradual decline underfrequency disturbance in the internal (left) and external (right) environments
Figure 4.14: TC5b – Adaptive filter correctly adjusts to a rapid decline underfrequency disturbance in the internal (left) and external (right) environments.
4.2.8 Test Case 7: B-field Alterations

Large alterations of the magnetic field in an environment are directly observed as changes in the interference amplitude. These large variations were only tested in the external environment. Four time periods were examined to model the filter’s behavior throughout the disturbance. Time period $t_1$ represented the time when the $B$-field increased to its maximum. Time period $t_2$ represented the time when the $B$-field remained constant at its maximum. Time period $t_3$ represented the time when the $B$-field decreased back to its original value. Time period $t_4$ represented the time after the $B$-field alteration disturbance was over. The results can be seen in Figure 4.16. When the $B$-field was steady, the filter was able to successfully reduce the interference. However, as the $B$-field changed, the filter was thrown off track. This is because $\mu_{\text{min}}$ is a function of the calculated input signal power ($\sigma^2$). Because the calculation is not instantaneous, $\mu_{\text{min}}$ is delayed, and the notch frequency weight is incorrectly adjusted.

4.2.9 Test Case: Tracking

Stimulating the filter with a signal with a linearly varying frequency allowed tracking analysis to be performed. This was performed by using a linear chirped sinusoid. The chirp rate was increased until the algorithm could no longer adjust the weights before the input
Figure 4.16: TC7 – Adaptive filter correctly adjusts to a large alteration in the magnetic field in the external environment

frequency reached 59.97 Hz. The tracking results for various chirp rates can be seen in Figure 4.17. As can be verified from the results, the adaptive notch filter performed better as the chirp rate decreased.

Figure 4.17: Tracking performance for a linear chirped sinusoid

4.3 Filtered ECG Example

An example of the cascaded ANF can be seen in Figure 4.18. Actual ECG signals were obtained from physionet.org. This particular ECG signal is record 106 from the MIT-BIH Arrhythmia Database and displays a normal sinus rhythm. Other interesting signals were used for testing to see how the interference affected defining characteristics, such as the onset of atrial fibrillation, found in record 202.
Figure 4.18: An actual ECG signal is filtered using the proposed ANF
5. VHDL Model

5.1 Lattice Structure Component

For every sample a minimum of two lattice computations are computed: one for the reference signal, and one for the primary signal. The appropriate input to the filter is expected in that order. This component allows for the cascading of up to 4 filters. If the component is set up to cascade, the entire cascade block for the reference signal is computed first. Figure 5.1 illustrates the lattice component schematic symbol and the corresponding signal descriptions can be found in Table 5.1.

![Lattice Filter Schematic](image)

**Figure 5.1: Lattice filter component schematic symbol**

The RTL diagram can be seen in Figure 5.2. The lattice computations utilize two distributed memory components, one multiply-accumulator (MAC), and one adder. The State
### Table 5.1: Lattice filter component signal descriptions

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>I</td>
<td>Global clock (needs to be ≥ 24x data_clk for each cascaded structure)</td>
</tr>
<tr>
<td>reset</td>
<td>I</td>
<td>Global reset tied to roc</td>
</tr>
<tr>
<td>en</td>
<td>I</td>
<td>Enable</td>
</tr>
<tr>
<td>data_clk</td>
<td>I</td>
<td>Reference and ECG data sample rate</td>
</tr>
<tr>
<td>data</td>
<td>I</td>
<td>Data Bus</td>
</tr>
<tr>
<td>num_casc</td>
<td>I</td>
<td>Number of filters to cascade</td>
</tr>
<tr>
<td>casc_num</td>
<td>O</td>
<td>Current cascade block being calculated</td>
</tr>
<tr>
<td>F_{state}</td>
<td>I</td>
<td>Fraction state of current cascade block</td>
</tr>
<tr>
<td>DPRA_{data}</td>
<td>I</td>
<td>Dual Port RAM Address for filter state variables</td>
</tr>
<tr>
<td>QDPO_{data}</td>
<td>O</td>
<td>Dual Port RAM Data Output for filter state variables</td>
</tr>
<tr>
<td>A_{coef}</td>
<td>I</td>
<td>Address Bus for filter weights</td>
</tr>
<tr>
<td>D_{coef}</td>
<td>I</td>
<td>Data Bus for filter weights</td>
</tr>
<tr>
<td>WE_{coef}</td>
<td>I</td>
<td>Write Enable control line for filter weights</td>
</tr>
<tr>
<td>result</td>
<td>O</td>
<td>Output of the lattice filter computations</td>
</tr>
<tr>
<td>rdy</td>
<td>O</td>
<td>Output ready</td>
</tr>
</tbody>
</table>

Variables memory contains the lattice state variable values, and the weight memory contains the sine and cosine weight values. The weight memory is a dual port distributed memory component, allowing other components to update the stored weight values. The MAC is configured to multiply and accumulate two values, and the output is registered.

The lattice component needs to operate at a higher frequency than the sample rate. The latency to complete one lattice computation is 12 cycles, therefore a minimum of 24 cycles are needed for the reference and primary lattice computations.

Figure 5.3 illustrates the timing diagram for one lattice computation. Assuming a sample rate of 1kHz (data_clk), this example (clk = 100 kHz) would allow for the cascading of 4 structures (96 total cycles). The MAC needs to have a registered output because an intermediate state variable (g) needs to be held for 2 clock cycles from 60 ns to 80 ns. This is possible because the MAC component is enabled by the New Data (ND) signal. The ND signal qualifies that the data on the A and B input ports is to be inserted into the MAC. If there is no new data for the MAC to work on, it remains idle, holding the previous result in the register.
Figure 5.2: Lattice filter component RTL diagram
All of the control signals are maintained by a microprogram micro-sequencer. The microcode control signals table is illustrated in Table 5.2. When enabled, the micro-sequencer will step through the addresses of the microcode stored in memory. Each address will output the necessary control signals to perform the correct operations. One complete run through the microprogram will result in one lattice computation to be completed. The lattice computations for the cascaded filters that the reference signal propagates through will be completed first. Then the computations for the primary signal are performed.

A program counter is used to control the microprogram. The *num_casc* signal determines how many structures to cascade, while *casc_cnt* and *fetch_typ* keep track of which cascaded structure is currently being processed and *ref_prm* maintains whether this is the reference or primary signal. There are separate memory locations for each of the individual filters that is to be cascaded. The memory locations are separated by the upper bits of the address line for both the State Variables and weight memory blocks, specified by *casc_cnt*. The weight memory blocks do not differ between the reference and primary signals, but the State Variables memory blocks do. Therefore, the address lines are separated by another bit (MSB) specified by *ref_prm*.
Table 5.2: Microcode control signals table

<table>
<thead>
<tr>
<th>Micro Address</th>
<th>$A_{data}[4:0]$</th>
<th>$WE_{data}$</th>
<th>$A_{coef}[4:0]$</th>
<th>MUX[1:0]</th>
<th>FD</th>
<th>ND</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>***10$^1$</td>
<td>0</td>
<td>**110$^2$</td>
<td>-3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0001</td>
<td>***</td>
<td>0</td>
<td>**100</td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0010</td>
<td>***</td>
<td>0</td>
<td>**110</td>
<td>0-</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0011</td>
<td>***10</td>
<td>0</td>
<td>**101</td>
<td>0-</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0100</td>
<td>***11</td>
<td>1</td>
<td>**</td>
<td>10</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0101</td>
<td>***01</td>
<td>0</td>
<td>**010</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0110</td>
<td>***10</td>
<td>1</td>
<td>**000</td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0111</td>
<td>***01</td>
<td>0</td>
<td>**001</td>
<td>11</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1000</td>
<td>***10</td>
<td>0</td>
<td>**010</td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1001</td>
<td>***10</td>
<td>1</td>
<td>**</td>
<td>10</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1010</td>
<td>***10</td>
<td>0</td>
<td>**</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1011</td>
<td>***01</td>
<td>1</td>
<td>**</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

1. Upper three bits of $A_{data}$ controlled by outside logic ($\text{ref_prm}$ & $\text{case_cem}$); Specifies individual filter
   Reference Cascaded: 000-011
   Primary Cascaded: 100-111
   Lower two bits determine state variable for specific filter
   01: $x_1$
   10: $x_2$
   11: $w$

2. Upper two bits of $A_{coef}$ controlled by outside logic ($\text{case_num}$); Specifies cascade number
   Individual cascaded filter weights: 00-11
   Lower three bits determine specific weight
   000: $\sin \theta_1$
   001: $-\sin \theta_1$
   010: $\cos \theta_1$
   100: $\sin \theta_2$
   001: $-\sin \theta_2$
   110: $\cos \theta_2$

3. Don’t care bits
5.2 Update Algorithm Components

5.2.1 Variance Estimator Component

The variance provides a statistical measure of a signal. The update algorithm utilizes the variance of the reference input signal and the filter error output signal. The equation for the variance can be seen in Equation 5.1a. After some manipulations, this component updates the variance by using Equation 5.1b. By maintaining a running cumulative sum, the oldest value can be subtracted and the newest value added to obtain the updated variance. Figure 5.4 illustrates the variance estimator component schematic symbol and the corresponding signal descriptions can be found in Table 5.3.

$$\hat{\sigma}^2(n) = \frac{1}{N} \sum_{i=0}^{N-1} x_{n-i}^2$$ (5.1a)

$$\hat{\sigma}^2(n+1) = \frac{1}{N} \left[ \sum_{i=0}^{N-1} x_{n-i}^2 - x_{n-N+1}^2 + x_{n+1}^2 \right]$$ (5.1b)

The RTL diagram can be seen in Figure 5.5. The design takes on a circular shift register form. A distributed memory component is utilized to retain the most recent N values. Designing N to be a power of 2, the division can subsequently be replaced by a simple shift operation, at the cost of accuracy. Including only the previous samples that made up one period of the input function would result in the most accurate estimate of the variance, but would require more circuitry to be sample synchronized. However, if N is made large
Table 5.3: Variance estimator component signal descriptions

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>I</td>
<td>Global clock (needs to be ≥ 24x data_clk for each cascaded structure)</td>
</tr>
<tr>
<td>reset</td>
<td>I</td>
<td>Global reset tied to roc</td>
</tr>
<tr>
<td>data</td>
<td>I</td>
<td>Data Bus</td>
</tr>
<tr>
<td>fd</td>
<td>I</td>
<td>First data signal</td>
</tr>
<tr>
<td>nd</td>
<td>I</td>
<td>New data signal, starts a new computation</td>
</tr>
<tr>
<td>(\hat{\sigma}^2)</td>
<td>O</td>
<td>Estimated variance (input or error) for current cascade block</td>
</tr>
<tr>
<td>rdy</td>
<td>O</td>
<td>Output ready</td>
</tr>
</tbody>
</table>

enough to contain enough samples to constitute more than one period of the function, the error is minimized. If N is made too large, the ability for the filter to quickly react to a changing environment decreases. Therefore, a knowledge of the signal environment and sample rate will allow one to make a better decision for the size of N. Assuming an interfering signal frequency of 60 Hz and a sample rate of 1000 Hz, there are approximately 16.7 samples per period. Therefore, an appropriate value for N would be 32.

To allow for the cascading of multiple structures, the most recent values need to be separately stored for each variance estimate. The top two bits of the distributed memory address line are utilized to ensure the correct values are utilized. These bits are controlled by the nd and fd control signals. The address changes every rising edge of nd. If fd is asserted high, the address will reset to “00”. If fd is not asserted, the address is incremented by one.

Figure 5.6 illustrates the timing diagram for one variance computation. According to Xilinx coregen, the optimum number of stages for the multiplier was 4. While that calculation is in process, the adder/subtractor preemptively subtracts the oldest value from the cumulative summation. Once the multiplication is complete, the result is then added to the temporary summation yielding the new variance estimate. The corresponding address pointer is then incremented to be prepared for the next computation. The latency of this component is 7 cycles.
Figure 5.5: Variance estimator component RTL diagram

Figure 5.6: Variance estimator component timing diagram
5.2.2 Step-size Updater Component

The update algorithm is the component in adaptive signal processing that updates the filter weights. In order to take advantage of fast convergence and low steady-state misadjustment, additional circuitry is needed to allow for a variable step-size. Figure 5.7 illustrates the step-size updater component schematic symbol and the corresponding signal descriptions can be found in Table 5.4.

![StepSizeUpdater](image)

Figure 5.7: Step-size updater component schematic symbol

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>I</td>
<td>Global clock (needs to be ≥ 24x data_clk for each cascaded structure)</td>
</tr>
<tr>
<td>reset</td>
<td>I</td>
<td>Global reset tied to roc</td>
</tr>
<tr>
<td>casc_num</td>
<td>I</td>
<td>Current cascaded filter number</td>
</tr>
<tr>
<td>F_state</td>
<td>I</td>
<td>Fraction state of current cascade block</td>
</tr>
<tr>
<td>nv</td>
<td>I</td>
<td>New variance signal, starts a new computation</td>
</tr>
<tr>
<td>var_type</td>
<td>I</td>
<td>Variance type, input ('1') or error ('0')</td>
</tr>
<tr>
<td>(\hat{\sigma}^2)</td>
<td>I</td>
<td>Estimated variance (input or error) for current cascade block</td>
</tr>
<tr>
<td>DPRA_data</td>
<td>I</td>
<td>Dual Port RAM Address for each structure’s step-size</td>
</tr>
<tr>
<td>QDPO_data</td>
<td>O</td>
<td>Dual Port RAM Data Output for each structure’s step-size</td>
</tr>
<tr>
<td>rdy</td>
<td>O</td>
<td>Output ready</td>
</tr>
</tbody>
</table>

The RTL diagram can be seen in Figure 5.8. The design can be broken up into three processes. The ControlLogic process controls the flow of data throughout the component.
It is responsible for reducing the 21-bit variance to a 12-bit value based on the \( F_{\text{state}} \) signal. It also stores the necessary information for the current calculation in a distributed memory component. The \( \text{UpdateMuMax} \) process calculates the maximum step-size (\( \mu_{\text{max}} \)). The reciprocal component has a latency of 24 cycles, so the process was designed to pipeline two inputs at a time. This ability is necessary when cascading filter blocks. The \( \text{UpdateMu} \) process calculates the step-size. It utilizes a state machine to select the data for the A and B input ports to be inserted into the adder. The input and output of the adder needs to be 14 bits in order to maintain precision. The process is responsible for correctly shifting and padding the input to 14 bits, and then shifting and zero-stripping the output back down to 12 bits. The state machine is utilized to control pre and post-processing of the data. The processes are run in parallel to maximize performance. While the \( \text{UpdateMu} \) process uses the current value of \( \mu_{\text{max}} \), the \( \text{UpdateMuMax} \) process is calculating the next value of \( \mu_{\text{max}} \).

Figure 5.9 illustrates the state machine used to adjust the step-size. The Idle State is utilized to allow for cascaded blocks. The state of each block is registered as it is updated. When a new update is triggered, the state of the current block is loaded up, the state is updated and registered, and the state machine goes back into the Idle State. Transitions to State 1 result in increasing \( \mu \), while transitions to State 2 result in decreasing \( \mu \). The conditions for each of the transitions can be seen in Table 5.5, with the corresponding step-size assignments. The conditions use a simple relational operator that compares the signal power (variance) of the input and output of the filter. When the notch frequency is far away from the input signal frequency, much of the signal will pass through the filter, resulting in an output variance close to that of the input variance. In this case a larger step-size would be more appropriate to allow for a faster convergence. When the notch frequency is close to that of the input signal frequency, the output variance will be much less than the input variance. In this case a smaller step-size would be more appropriate to allow for a smaller misadjustment. Each of the assignments purposefully utilize simple shift and add operations. This eliminates costly multiplications and reduces the complexity.
Figure 5.8: Step-size updater component RTL diagram

Figure 5.9: State machine for step-size updater. Transitions to State 1 result in increasing \( \mu \), while transitions to State 2 result in decreasing \( \mu \).

85
Table 5.5: Step-size state machine table

<table>
<thead>
<tr>
<th>Transition</th>
<th>Condition</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$\sigma_e^2 &gt; \frac{3\sigma^2_e}{4}$</td>
<td>$\mu = \frac{5\mu}{4}$</td>
</tr>
<tr>
<td>B</td>
<td>$\sigma_e^2 &lt; \frac{3\sigma^2_e}{4}$</td>
<td>$\mu = \frac{\mu}{2} + \frac{\mu_{max}}{32}$</td>
</tr>
<tr>
<td>C</td>
<td>$\sigma_e^2 &lt; \frac{\sigma^2_e}{4}$</td>
<td>$\mu = \frac{\mu}{2} + \frac{\mu_{max}}{64}$</td>
</tr>
<tr>
<td>D</td>
<td>$\sigma_e^2 &lt; \frac{3\sigma^2_e}{4}$</td>
<td>$\mu = \frac{\mu}{2} + \frac{\mu_{max}}{32}$</td>
</tr>
<tr>
<td>E</td>
<td>$\sigma_e^2 &gt; \frac{3\sigma^2_e}{4}$</td>
<td>$\mu = 2\mu$</td>
</tr>
</tbody>
</table>

Figure 5.10 illustrates the timing diagram for one step-size computation. The component starts in an idle state waiting for a new variance signaled by the \texttt{nv} control strobe, and behaves according to the \texttt{var_type} signal. The timing diagram provides examples for both variance types: input (\texttt{var_type} = 1) and error (\texttt{var_type} = 0). The step-size is updated when a new variance of type input is processed. As the \textit{ControlLogic} process writes the incoming values to memory, the \textit{UpdateMu} process uses the values that were previously stored to perform its calculations. The previous states input and error variances are used to update the state machine. The new value of $\mu$ is then written to memory after 7 cycles.

![Figure 5.10: Step-size updater component timing diagram](image-url)
5.2.3 Coefficient Updater Component

The weight updater component updates the lattice weight values. The component has two modes of operation specified by the type signal: to hard code a specified lattice weight, or to update the $\theta_1$ weights using the LMS equation. The LMS update equation is described in (5.2), where $\mu$ is the step-size described in previous sections, $\hat{y}$ is the output of the lattice notch filter, and $x_1$ is a lattice state variable. Figure 5.11 illustrates the weight updater component schematic symbol and the corresponding signal descriptions can be found in Table 5.6.

$$\theta(n + 1) = \theta(n) - \mu \hat{y} x_1$$  \hspace{1cm} (5.2)

![Figure 5.11: Weight updater component schematic symbol](image)

The RTL diagram can be seen in Figure 5.12. A multiplier and subtracter were needed to compute $\theta$ using the LMS update equation. The case_num signal is used to read the correct variables from the other components memory, and utilize the correct value of $\theta$ in the update. According to Xilinx Coregen, the optimum number of stages for the multiplier was 4. The latency for $\theta$ to update using the LMS equation is 10 cycles. While the next value for $\theta$ is being computed in the LMSUpdate process, the CORDIC (Coordinate Rotational Digital Computer) component is simultaneously calculating the sine and cosine.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>I</td>
<td>Global clock (needs to be ≥ 24x data clk for each cascaded structure)</td>
</tr>
<tr>
<td>reset</td>
<td>I</td>
<td>Global reset tied to roc</td>
</tr>
<tr>
<td>casc_num</td>
<td>I</td>
<td>Current cascade block number</td>
</tr>
<tr>
<td>F_state</td>
<td>I</td>
<td>Fraction state of current cascade block</td>
</tr>
<tr>
<td>nd</td>
<td>I</td>
<td>New data signal, starts a new computation</td>
</tr>
<tr>
<td>type</td>
<td>I</td>
<td>Update type, tells component what type of update to perform</td>
</tr>
<tr>
<td>θ</td>
<td>I</td>
<td>Theta, overwrites the specified lattice weight when type = “10”</td>
</tr>
<tr>
<td>DPRA_s</td>
<td>O</td>
<td>Dual Port RAM Address for filter state variables</td>
</tr>
<tr>
<td>QDPO_s</td>
<td>I</td>
<td>Dual Port RAM Data Output of filter state variables</td>
</tr>
<tr>
<td>DPRA_µ</td>
<td>O</td>
<td>Dual Port RAM Address for each structure’s step-size</td>
</tr>
<tr>
<td>QDPO_µ</td>
<td>I</td>
<td>Dual Port RAM Data Output of each structure’s step-size</td>
</tr>
<tr>
<td>ˆy</td>
<td>I</td>
<td>Output of current cascade block</td>
</tr>
<tr>
<td>A_coef</td>
<td>O</td>
<td>Address Bus for filter weights</td>
</tr>
<tr>
<td>D_coef</td>
<td>O</td>
<td>Data Bus for filter weights</td>
</tr>
<tr>
<td>WE_coef</td>
<td>O</td>
<td>Write Enable control line for filter weights</td>
</tr>
</tbody>
</table>

values of the previous value of θ. The CORDIC component is a simple and efficient way of calculating trig functions. For this design, the Word Serial Architecture was chosen. This implementation utilized 50% less area than the Parallel Architecture at the cost of increased latency. This decision was made because 24 cycles are needed to perform the lattice computations per cascaded block. As long as the latency did not exceed 24 cycles, there would be no decrease in throughput. The latency of the CORDIC component is 24 cycles. If the filter is set up to cascade multiple blocks, the input to the CORDIC is buffered.

Figure 5.13 illustrates the state machine used to write the weight values into the dual port distributed memory of the lattice filter component. The value for − sin θ is obtained by calculating the two’s complement of sin θ. The transition A occurs when the rising edge of cordic Nd is detected. The remaining transitions occur on subsequent rising edges of clk.

Figure 5.14 illustrates the timing diagram for one LMS update computation. The component is signaled to perform an update by the nd control strobe, and behaves according to the type signal.
Figure 5.12: weight updater component RTL diagram

Figure 5.13: State machine for writing weight values. weights are written in the order \( \cos \theta, \sin \theta, -\sin \theta \)
5.3 Final Design Overview

The overall timing diagram for one complete iteration can be seen in Figure 5.15. Each of the components are working in parallel updating different parameters. Events indicating when important parameters are updated are labeled on the diagram.

Figure 5.15: Final design overview timing diagram for single block implementation

The overall timing diagram for a cascaded implementation can be seen in Figure 5.16.
To indicate pipelining, the process is shaded. An example of this is the $\text{UpdateMuMax}$ process.

Figure 5.16: Final design overview timing diagram for cascaded implementation
6. Synthesis Results

Development of each of the design components was carried out using Xilinx ISE 9.1i and ModelSim SE 6.1a. Simulations were run through ModelSim for each component to verify functionality. The Xilinx ISE tool was used to compare various implementations. Different devices were targeted with multiple optimization efforts. The targeted devices were representative of a low-end FPGA (Spartan-II) and a high-end FPGA (Virtex-5 LX Platform). The optimization efforts were area and speed. Logic utilization and timing measurements were taken from Xilinx ISE synthesis results. The following sections summarize the results for each of the components.

6.1 Testbench Verification

Hardware verification was performed individually for each component by comparing the output waveforms from ModelSim to the expected timing diagrams. The component timing diagrams are presented in the implementation details found in Chapter 5. Separate testbenches were created to stimulate each component with the necessary control signals and data busses.

After each component was verified, the overall interface was tested. To do this, the output from ModelSim was compared against the output of MATLAB. A testbench was utilized to emulate the synthetic environment used by the MATLAB software simulations. A text file was used to read in values to simulate the output of an ADC. Utilizing the same input from the MATLAB simulations allowed for a direct comparison between the software and hardware outputs for an extended number of iterations. A diagram of the verification setup can be seen in Figure 6.1.
6.2 Lattice Structure Component

The lattice filter is the main engine of the adaptive notch filter. It offers up to four configurable second order lattice-form notch filters to be cascaded. The configuration will effectively process two signals in parallel. The modular design leaves flexibility on how the weight values are either statically set or dynamically updated. The design focused on minimizing the total number of clock cycles necessary to complete a computation. The lattice filter device utilization summary from the synthesis report for each of the implementations can be seen in Table 6.1. The utilization percent is shown after the total logic used number for each logic category. The post place and route report summary can be seen in Table 6.2. Included in the summary is the more accurate timing model.

6.3 Update Algorithm Components

The update algorithm components are responsible for updating the lattice filter weight values. Each of the designs were based upon the design results for the lattice core. This resulted in each component having a latency less than or equal to that of the lattice core.
Table 6.1: Lattice filter synthesis report device utilization summary and timing report

<table>
<thead>
<tr>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>slices</td>
<td>287; 12% 271; 11%</td>
</tr>
<tr>
<td>slice FFs</td>
<td>164; 3% 162; 3%</td>
</tr>
<tr>
<td>4-input LUTs</td>
<td>624; 13% 615; 13%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>28.532 31.167</td>
</tr>
</tbody>
</table>

(b) Xilinx Virtex-5 xc5vLX85

<table>
<thead>
<tr>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>slice registers</td>
<td>163; 0% 162; 0%</td>
</tr>
<tr>
<td>slice LUTs</td>
<td>557; 1% 559; 1%</td>
</tr>
<tr>
<td>bit slices</td>
<td>622 579</td>
</tr>
<tr>
<td>- fully used</td>
<td>- 15% - 24%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>128.107 103.405</td>
</tr>
</tbody>
</table>

Table 6.2: Lattice filter post place and route report device utilization summary and timing report

<table>
<thead>
<tr>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>external IOBs</td>
<td>75; 53% 75; 53%</td>
</tr>
<tr>
<td>SLICEs</td>
<td>318; 13% 316; 13%</td>
</tr>
<tr>
<td>best case freq (MHz)</td>
<td>42.805 39.997</td>
</tr>
</tbody>
</table>

Because of the dependencies between the components, each had to work in parallel and implement a pipelined architecture to meet the latency requirement.

### 6.3.1 Variance Estimator Component

The variance estimator component uses a shift register technique to estimate the variance of a signal based upon the most recent 32 values. A majority of the resources of this component are the registers needed to retain the values to separately calculate four variance estimates. The variance estimator device utilization summary from the synthesis report for each of the implementations can be seen in Table 6.3. The utilization percent is shown after the total logic used number for each logic category. The post place and route report summary can be seen in Table 6.4. Included in the summary is the more accurate timing model.
Table 6.3: Variance estimator synthesis report device utilization summary and timing report

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Xilinx Spartan-II xc2s200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slices</td>
<td>364; 15%</td>
<td>277; 12%</td>
</tr>
<tr>
<td>slice FFs</td>
<td>367; 8%</td>
<td>335; 7%</td>
</tr>
<tr>
<td>4-input LUTs</td>
<td>500; 11%</td>
<td>500; 11%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>54.410</td>
<td>53.536</td>
</tr>
<tr>
<td>(b) Xilinx Virtex-5 xc5vLX85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slice registers</td>
<td>357; 1%</td>
<td>335; 1%</td>
</tr>
<tr>
<td>slice LUTs</td>
<td>385; 1%</td>
<td>372; 1%</td>
</tr>
<tr>
<td>bit slices</td>
<td>600</td>
<td>481</td>
</tr>
<tr>
<td>- fully used</td>
<td>- 23%</td>
<td>- 46%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>284.536</td>
<td>312.925</td>
</tr>
</tbody>
</table>

Table 6.4: Variance estimator post place and route report device utilization summary and timing report

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Xilinx Spartan-II xc2s200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>external IOBs</td>
<td>37; 26%</td>
<td>37; 26%</td>
</tr>
<tr>
<td>SLICEs</td>
<td>308; 13%</td>
<td>301; 13%</td>
</tr>
<tr>
<td>best case freq (MHz)</td>
<td>80.880</td>
<td>84.624</td>
</tr>
</tbody>
</table>

6.3.2 Step-size Updater Component

The step-size updater component is responsible for updating the step-size used by the LMS update algorithm. A state machine is implemented to allow for a variable step-size. The component can maintain up to four separate variable step-sizes, each to be associated with one of the lattice filter cascade blocks. The step-size updater device utilization summary from the synthesis report for each of the implementations can be seen in Table 6.5. The utilization percent is shown after the total logic used number for each logic category. The post place and route report summary can be seen in Table 6.6. Included in the summary is the more accurate timing model.

6.3.3 Coefficient Updater Component

The weight updater component is responsible for updating the weight values of the lattice filter utilizing the LMS algorithm. The component can maintain up to four separate weight
### Table 6.5: Step-size updater synthesis report device utilization summary and timing report

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Xilinx Spartan-II xc2s200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slices</td>
<td>301; 13%</td>
<td>334; 14%</td>
</tr>
<tr>
<td>slice FFs</td>
<td>358; 8%</td>
<td>354; 8%</td>
</tr>
<tr>
<td>4-input LUTs</td>
<td>463; 10%</td>
<td>514; 11%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>55.540</td>
<td>53.319</td>
</tr>
<tr>
<td>(b) Xilinx Virtex-5 xc5vLX85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slice regs</td>
<td>357; 1%</td>
<td>354; 1%</td>
</tr>
<tr>
<td>slice LUTs</td>
<td>432; 1%</td>
<td>429; 1%</td>
</tr>
<tr>
<td>bit slices</td>
<td>587</td>
<td>574</td>
</tr>
<tr>
<td>- fully used</td>
<td>- 34%</td>
<td>- 36%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>277.816</td>
<td>244.888</td>
</tr>
</tbody>
</table>

### Table 6.6: Step-size updater post place and route report device utilization summary and timing report

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Xilinx Spartan-II xc2s200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>external IOBs</td>
<td>46; 25%</td>
<td>46; 25%</td>
</tr>
<tr>
<td>SLICEs</td>
<td>389; 16%</td>
<td>370; 16%</td>
</tr>
<tr>
<td>best case freq (MHz)</td>
<td>68.027</td>
<td>49.648</td>
</tr>
</tbody>
</table>

values, each to be associated with one of the lattice filter cascade blocks. The weight updater device utilization summary from the synthesis report for each of the implementations can be seen in Table 6.7. The utilization percent is shown after the total logic used number for each logic category. The post place and route report summary can be seen in Table 6.8. Included in the summary is the more accurate timing model.

### 6.4 Overall Analysis

The lattice component represents the performance bottleneck in terms of limiting the maximum operating frequency. This was expected because the overall latency was intentionally minimized for the block. If the multiply-accumulator was pipelined, the component would be able to operate at a higher frequency. However, the sample rate would be adversely affected because more clock cycles would be necessary to complete one iteration of the adaptation algorithm.
Table 6.7: Coefficient updater synthesis report device utilization summary and timing report

(a) Xilinx Spartan-II xc2s200

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>slices</td>
<td>653; 28%</td>
<td>628; 17%</td>
</tr>
<tr>
<td>slice FFs</td>
<td>760; 16%</td>
<td>755; 16%</td>
</tr>
<tr>
<td>4-input LUTs</td>
<td>1148; 24%</td>
<td>1145; 24%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>49.928</td>
<td>49.928</td>
</tr>
</tbody>
</table>

(b) Xilinx Virtex-5 xc5vLX85

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>slice regs</td>
<td>760; 1%</td>
<td>755; 1%</td>
</tr>
<tr>
<td>slice LUTs</td>
<td>993; 2%</td>
<td>991; 2%</td>
</tr>
<tr>
<td>bit slices</td>
<td>1144</td>
<td>1120</td>
</tr>
<tr>
<td>- fully used -</td>
<td>-53%</td>
<td>-55%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>204.311</td>
<td>204.311</td>
</tr>
</tbody>
</table>

Table 6.8: Coefficient updater post place and route report device utilization summary and timing report

(a) Xilinx Spartan-II xc2s200

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>external IOBs</td>
<td>91; 65%</td>
<td>91; 65%</td>
</tr>
<tr>
<td>SLICEs</td>
<td>656; 27%</td>
<td>656; 27%</td>
</tr>
<tr>
<td>best case freq (MHz)</td>
<td>69.614</td>
<td>67.454</td>
</tr>
</tbody>
</table>
The next largest bottleneck was the computation of the variable $F_{state}$. This computation represented six levels of logic (Figure 4.5), resulting in a large propagation delay. The delay could be reduced if the logic was pipelined over several clock cycles. If the algorithm did not use variable length parameters, this propagation delay would be eliminated.

The synthesis report for the complete adaptive notch filter implementation can be seen in Table 6.9, and the post place and route report can be seen in Table 6.10. The resulting maximum sample rate for each of the targeted implementations can be seen in Table 6.11. The maximum sample rate was calculated using the synthesis report estimates in order to make a fair comparison between the Spartan-II and Virtex-5 device implementations. The maximum frequency of operation is $24\times$ the maximum sample rate. As expected, the technology of the Virtex-5 FPGA allowed for a much higher frequency of operation than the Spartan-II FPGA, resulting in a speedup of $\sim3.8$. Changing the optimization effort to speed, resulted in a speedup of $\sim1.2$. By changing the optimization effort to area, a more efficient use of resources was seen. This was most evident in the variance estimator component.

Table 6.9: Adaptive notch filter synthesis report device utilization summary and timing report

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Xilinx Spartan-II xc2s200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slices</td>
<td>1706; 72%</td>
<td>1543; 65%</td>
</tr>
<tr>
<td>slice FFs</td>
<td>1707; 36%</td>
<td>1702; 36%</td>
</tr>
<tr>
<td>4-input LUTs</td>
<td>2864; 60%</td>
<td>2819; 59%</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>30.122</td>
<td>25.217</td>
</tr>
<tr>
<td>(b) Xilinx Virtex-5 xc5vLX85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>slice regs</td>
<td>1702; 3%</td>
<td>1702; 3%</td>
</tr>
<tr>
<td>slice LUTs</td>
<td>2405; 4%</td>
<td>2420; 4%</td>
</tr>
<tr>
<td>bit slices</td>
<td>3050</td>
<td>2811</td>
</tr>
<tr>
<td>- fully used</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>max freq (MHz)</td>
<td>111.130</td>
<td>96.074</td>
</tr>
</tbody>
</table>

For the application of removing PLI from ECG signals, the sample rate is only 1 kHz. Therefore, the low-end FPGA would be suitable for the application. Also, the area optimization would more appropriate for the application. This would allow for more resources to be used by other components performing biomedical functions.

For other applications, such as the baseband processing used in RF radio links, a higher
Table 6.10: Adaptive notch filter post place and route report device utilization summary and timing report

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>external IOBs</td>
<td>15; 10%</td>
<td>15; 10%</td>
</tr>
<tr>
<td>SLICEs</td>
<td>1729; 73%</td>
<td>1704; 72%</td>
</tr>
<tr>
<td>best case freq (MHz)</td>
<td>38.127</td>
<td>32.861</td>
</tr>
</tbody>
</table>

Table 6.11: Maximum input sample rate for each of the targeted implementations

<table>
<thead>
<tr>
<th>Target Device</th>
<th>Optimization</th>
<th>Max Sample Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spartan-II</td>
<td>Speed</td>
<td>1.255 MHz</td>
</tr>
<tr>
<td>Spartan-II</td>
<td>Area</td>
<td>1.051 MHz</td>
</tr>
<tr>
<td>Virtex-5</td>
<td>Speed</td>
<td>4.630 MHz</td>
</tr>
<tr>
<td>Virtex-5</td>
<td>Area</td>
<td>4.003 MHz</td>
</tr>
</tbody>
</table>

A high-end FPGA optimized for speed would better meet these needs. If area is less of a concern, than an additional speedup of 2 is achievable by filtering the reference and primary signal in parallel.
7. Conclusions and Future Work

The proposed lattice form adaptive notch filter was able to successfully track and remove the interfering signal. The filter was subjected to various environments that modeled the different power line disturbances that could be present. The potential disturbances that could be present in the power line interference was modeled based on the results of the power systems research.

The final filter design resulted in a $-3$ dB bandwidth of $15.8908$ Hz, and a null depth of $-54$ dB. For the baseline test case, the algorithm achieved convergence after 270 iterations. The filter was able to successfully reduce the interference below the acceptable signal power level threshold of $5E-5$ mV$^2$ for a majority of the test cases. The filter was not successful in reducing the interference signal power below the acceptance threshold for the cumulative test case that accumulated all of the potential disturbances.

The final hardware implementation was successfully verified against the MATLAB simulation results. A speedup of $\sim 3.8$ was seen between the Xilinx Virtex-5 and Spartan-II device technologies. The final design used a small fraction of the available resources for each of the two devices that were characterized. This would allow the component to be more readily available to be added to existing projects, or further optimized by utilizing additional logic.

7.1 Future Implementation Work

Future work could include verifying the FPGA functionality using actual ECG signals, or function generators. After the register transfer language (RTL) has been synthesized, an Agilent 33120A 15 MHz Function/Arbitrary Waveform Generator could be used to verify functionality. This function generator has the ability to reproduce an arbitrary ECG signal, as well the sinusoidal power line interference with additive white Gaussian noise (AWGN).
The ECG test signal is one of the built-in arbitrary waveforms stored in non-volatile memory. Two function generators would need to be utilized. One to generate the reference signal, and the other to generate the contaminated input signal. Successful implementation can be verified visually by using an oscilloscope to view the output of the filter. An ADC with sample/hold circuitry could be used on the front end to provide the necessary data acquisition. Typical sampling frequencies for ECGs would be around 1 kHz, although 512 Hz would be high enough to satisfy the Nyquist condition [4].
Bibliography


103


