2011

DSP implementation and study of sliding mode control algorithms for power factor correction in AC-DC converters

Elise Goff

Recommended Citation

Goff, Elise, "DSP implementation and study of sliding mode control algorithms for power factor correction in AC-DC converters" (2011). *Theses*. 513.
https://louis.uah.edu/uah-theses/513

DSP IMPLEMENTATION AND STUDY OF SLIDING MODE CONTROL ALGORITHMS FOR POWER FACTOR CORRECTION IN AC-DC CONVERTERS

by

ELISE GOFF

A THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in
The Department of Electrical and Computer Engineering
of
The School of Graduate Studies
of
The University of Alabama in Huntsville

HUNTSVILLE, ALABAMA

2011

In presenting this thesis in partial fulfillment of the requirements for a master's degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department or the Dean of the School of Graduate Studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

	10/24/11
(student signature)	(date)

THESIS APPROVAL FORM

Submitted by Elise Goff in partial fulfillment of the requirements for the degree of Master of Science in Engineering and accepted on behalf of the Faculty of the School of Graduate Studies by the thesis committee.

We, the undersigned members of the Graduate Faculty of The University of Alabama in Huntsville, certify that we have advised and/or supervised the candidate on the work described in this thesis. We further certify that we have reviewed the thesis manuscript and approve it in partial fulfillment of the requirements for the degree of Master of Science in Engineering.

10/24/11 Committee Chair
(Date)

10/24/11

10/26/11

Robert Department Chair

10/31/11 College Dean

Rhonda Kay Gaede 12/2/11 Graduate Dean

ABSTRACT

The School of Graduate Studies
The University of Alabama in Huntsville

Degree Master of Science in Engineering College/Dept. Engineering/Electrical and Computer Engineering

Name of Candidate Elise Goff

Title DSP Implementation and Study of Sliding Mode Control Algorithms for Power Factor Correction in AC-DC Converters

A DSP implementation of sliding mode control algorithms for power factor correction of a three-phase full bridge AC-DC boost converter is studied. The control algorithms were developed from previous studies and include classical sliding mode controls, super twisting sliding mode controls and sliding mode observers/differentiators. A co-simulation between a DSP with encoded sliding mode control algorithms and a computer which emulates the power block is accomplished. The controller is tested in this configuration under varying conditions. The studies of the embedded processor implementation of the power factor control system include observing the effects of varying source inductance of the converter, exploring the limitations on input frequency values, and testing the control's ability to adapt to varying load resistances. The validity of the DSP implementation of the sliding mode algorithms is confirmed by comparing the results of this co-simulation to the results of previously implemented computer-only simulations studies.

Abstract Approval: Committee Chair Yuri Shtessel 10/24/11
(Date)

Department Chair

Graduate Dean Rhonda Kay Gaede 12/2/11

ACKNOWLEDGMENTS

I would like to acknowledge everyone who provided me support while I pursued a Master's degree. First, I would to thank Dr. Yuri Shtessel for his instruction and advice throughout my work as well as providing graduate research assistanceship. Also I would like to thank Dr. Lindquist and the ECE department for providing a teaching assistanceship that provided funding and experience. I would like to extend thanks to Dr. Haik Biglari and the other engineers of Fairchild Controls who helped me and allowed me to travel to their facilities in Maryland to learn from them and gain from their experience. Lastly, I would like to thank my family and friends for always supporting me in my efforts.

TABLE OF CONTENTS

Page

LIST OF FIGURES viii

LIST OF TABLES xi

LIST OF SYMBOLS xii

LIST OF ACRONYMS xiii

Chapter

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Problem Overview 2

1.3 Literature Review 3

1.4 Thesis Organization 5

2 OVERVIEW OF AC-DC POWER CONVERTERS 6

2.1 Hardware Topology and Math Models of AC-DC Boost Converter 6

2.1.1 Single Phase AC to DC Boost Converter 7

2.1.2 Three Phase AC to DC Boost Converter Topology 9

2.1.3 Three Phase AC to DC Boost Converter Math Model 10

2.2 Power Factor 12

2.3 Summary 15

3 FUNDAMENTALS OF SLIDING MODE CONTROL 16

3.1 Classical Sliding Mode Control 17

3.1.1 General Case of Classical Sliding Mode Control and Definitions 17

3.1.2 An Example Implementing Classical Sliding Mode Controls 19

3.2 Second Order Sliding Mode Control (2-SMC) 26

3.2.1 Twisting Controls 26

3.2.2 Prescribed Convergence Law 27

3.2.3 Super Twisting Sliding Mode Controls 28

3.3 Sliding Mode Observers and Differentiators 31

3.4 Summary 36
4 POWER FACTOR CORRECTION USING SLIDING MODE TECHNIQUES 38
4.1 Assumptions and Control Objectives... 38
4.2 Current Error Tracking and Ideal Current Generation... 41
4.3 Development of Classical Sliding Mode Controller43
4.4 Load Resistance Observer... 45
4.5 Complete Mathematical Model of the Control Algorithm 46
4.6 Summary 47
5 DSP SIMULATION ENVIRONMENT DEVELOPMENT AND COMPOSITION .. 49
5.1 Overview of Co-Simulation Preparation 49
5.2 DSP Selection 51
5.3 Physical Structure of the Experimental Setup 55
5.4 Simulink Environment and Code Composer Studio IDE 57
5.4.1 SimPowerSystems and Simulink Coder Setup 57
5.4.2 Code Composer Studio Configuration.. 62
5.5 Running a PIL Simulation 63
5.6 Summary 67
6 DSP IMPLEMENTATION AND CO-SIMULATION STUDY RESULTS 68
6.1 Nominal Mode 69
6.2 Effect of Varying Source Inductance... 77
6.3 Effect on System Performance from Varying Phase Voltage Frequency............... 82
6.4 Accommodation of Varying Load Resistance 86
6.5 Summary 91
7 CONCLUSION... 94
REFERENCES 97

LIST OF FIGURES

Figure Page

2.1 Single phase boost converter [11]. 7

2.2 Single Phase Rectifier. 8

2.3 Three phase full bridge AC-DC boost converter [1]. 9

2.4Three Phase Boost Converter. 10

3.1 Uncontrolled sliding variables with bounded disturbance. 23

3.2 State variables controlled using the classical SMC technique. 24

3.3 The sliding variable σ. 24

3.4 The control signal u. 25

3.5 A closer view of the sliding variable during the sliding phase. 25

3.6 State variable behavior controlled using the super twisting algorithm. 29

3.7 Sliding variable from the super twisting control. 30

3.8 The control signal u of the super twisting sliding mode controller. 30

3.9 Observation error e. 33

3.10 Close inspection of observation error with chattering. 33

3.11 Observer output tracking velocity. 34

3.12 Super twisting parameter observer output. 35

3.13 Observation error from super twisting observer. 35

3.14 Comparison of the classical sliding mode observer and the super twisting sliding mode observer 36

5.1 A computer and a DSP in a co-simulation loop. 49

5.2 An MPC555 DSP. 53

5.3 ezDSP F28335 by Spectrum Digital. 54

5.4 Physical interface between the DSP and the computer. 56

5.5 SimPowerSystems three phase AC-DC converter model. 58

5.6 SimPowerSystems interface to other model simulation components. 58

5.7 Control block with data type conversions at the interface. 59

5.8 Under the mask of the control subsystem including target preferences block. 60

5.9 Workspace variables and plotting blocks. 61

5.10 Power Factor Calculators. 61

5.11 Code Composer Studio GUI after the device is connected. 63

5.12 Build window for control subsystem in Simulink. 64

5.13 Controller project displayed in CCS File View pane. 65

5.14 Typical C code generated in CCS. 65

5.15 PIL block generated by building the control block in Real Time Workshop. 66

6.1 Phase current and ideal phase current for nominal mode. 70

6.2 Phase voltage and current for nominal mode. 71

6.3 Phase currents in nominal mode. 71

6.4 Load resistance observer in nominal mode. 72

6.5 DC output voltage tracking in nominal mode. 73

6.6 Power factors with only phase term in nominal mode. 74

6.7 Power factor with harmonic distortion term in nominal mode. 74

6.8 One control signal in nominal mode 75

6.9 Control signals in nominal mode. 76

6.10 Corresponding switching signals in nominal mode. 76

6.11 Phase current and ideal current with 5 mH phase inductance. 78

6.12 Phase voltage and current with 5 mH phase inductance. 78

6.13 DC output voltage with 5 mH phase inductance. 79

6.14 Power factors with harmonic distortion term and 5 mH phase inductance. 80

6.15 Phase voltage and current with 10 mH phase inductance. 81

6.16 Power factors with harmonic distortion term and 10 mH phase inductance. 81

6.17 Phase and corresponding ideal current with phase voltage frequency at 200Hz. 83

6.18 Power Factors with only phase term, phase voltage frequency 200 Hz. 83

6.19 Phase voltage and current with phase voltage frequency of 240 Hz. 84

6.20 Power factors with only phase term, phase voltage frequency 240 Hz. 84

6.21 Phase voltage and current, phase voltage frequency 240 Hz and 1MHz control evaluation rate. 85

6.22 Power factors with only phase term, phase voltage frequency 240 Hz and 1 MHz control evaluation rate. 86

6.23 Model additions for load resistance evaluations. 87

6.24 Observed varying load resistance. 88

6.25 Phase voltage and current when load resistance varies. 89

6.26 DC output voltage when load resistance varies. 90

6.27 Power factors with harmonic distortion term when load resistance varies. 90

LIST OF TABLES

Table Page

6.1: Nominal Mode Parameters 69

LIST OF SYMBOLS

C	Output capacitance
E	Voltage magnitudes
f	Input frequency
i_j	Input phase currents
i_d	Desired phase current profile
L	Phase inductance
M	Lipchitz constant
r	Parasitic phase resistance
R	Load resistance
R_o	Nominal load resistance
σ	Sliding Variable
U_j	Control Signal
U_0	Output voltage
U_{gj}	Input phase voltages
v	Sliding mode injection term

LIST OF ACRONYMS

AC	Alternating Current
CCS	Code Composer Studio
DC	Direct Current
DSP	Digital Signal Processor
GUI	Graphical User Interface
IDE	Integrated Development Environment
JTAG	Joint Test Action Group
PC	Personal Computer
PIL	Processor In the Loop
PWM	Pulse Width Modulation
RAM	Random Access Memory
RMS	Root Mean Square
SMC	Sliding Mode Control
TCP/IP	Transmission Control Protocol/Internet Protocol
USB	Universal Serial Bus

CHAPTER 1

INTRODUCTION

Elimination of energy waste is one of the most significant discussions amongst engineers and scientists in the current decade. Many people have dedicated their time to the task of creating products that run on less power and even more importantly, do not waste energy or create harmful emissions. Power factor is the primary measure of efficiency of a system. If the power factor is said to be at unity then the system does not suffer from power losses. In power converters, distortions in the energy source cause the power factor to be depleted [2]. The goal of many researchers in this field is to eliminate these distortions and increase the power factor to as close to unity as possible.

1.1 Motivation

Any control algorithm that is designed for high power three phase AC-DC boost converters must be examined and tested carefully before it directly implemented. These converters contain sensitive circuits which must be interfaced precisely. A single misstep in wiring or switching of such a circuit can result in harm to the operator of the device

and to the device itself [1, 2]. Control algorithms have been designed in previous reports [3, 4] implementing sliding mode controls to perform this correction. These controls have never been thoroughly tested by implementation on an embedded processor. Encoding the controls to a digital signal processor and carefully examining the effectiveness and limits to such a control design is the next major design step in implementing the control in a hardware setting.

1.2 Problem Overview

The problem presented in this thesis is to implement the sliding mode control algorithms developed in [3, 4] on a DSP and test the encoded device with a simulated model of the plant system. The simulated plant model will run on a computer which will monitor the outputs of the DSP as well as the plant model parameters in order to verify that the control algorithms work as desired. The other objective is to test the system by varying parameters in the plant model to improve the results of the control or to strain the controller in order to test its theoretical boundaries.

The control algorithms which were designed and implemented on the DSP should produce the same results seen in the previous studies. The AC-DC converter is analyzed and the control algorithms are redefined in order to provide background and ensure the control algorithms are implemented correctly. These control algorithms include classical sliding mode control [5], super twisting sliding mode control [6], and sliding mode parameter observers [7]. Sliding mode controls are a natural choice for AC-DC boost converters circuits since the converters contain many nonlinear elements and the circuit is

controlled via switching signals. It is desired that the control will bring the power factor to near unity and it should be able to conform to varying load resistances and to tolerate high phase voltage frequencies [3].

1.3 Literature Review

Many documents have been published in the area of power factor correction for boost converters. Textbook controllers for boost converters implement pulse width modulation in order to generate the switching signals for the transistors. Controls that strictly use PWM are linear in nature and the nonlinear switching elements will cause distortion on the input current, leading to a lower power factor [1]. The researchers [8] addressed this problem by compensating for the parasitic phase capacitances of the switching elements. The solution did not require a lot of extra hardware but only yielded fair improvements to the overall power factor.

Reference [9] is a very interesting study focused on power factor correction of a single-phase boost converter. The researchers in this study use an algorithm which predicts the required duty cycles to achieve unity power factor. This algorithm was both simulated and experimentally implemented using DSPs and can operate at high switching frequencies. Although the results of both the simulation and experiment seemed to produce high power factors, the control is not based on feedback and may be sensitive to disturbances.

The researchers in [10] tackle the power factor correction by dividing the three phase module into three single phase modules with the outputs connected in

parallel. This is more cost and space effective. These modules are controlled using what is called an active clamp technique and a soft switching topology. The proposed control method should work under variable load conditions as with the sliding mode controller. The simulations for this control method produced high power factors and fairly low total harmonic distortion. Implementation of this method for control does not work on standard three phase rectifiers and requires extra hardware. Also, the control does not work with variable phase voltage frequency which is could be very important in some applications.

Sliding mode control algorithms were researched in [3, 4, 11]. The research in [11] focused on single phase correction using a time varying sliding surface. The research in [4] explores the classical sliding mode control, higher order sliding mode controls, and other nonlinear control algorithms such as feedback linearization. Some of the control algorithms presented in [4] did not improve on the classical method, such as the use of transformed sliding variables. The use of higher order sliding mode controls required pulse width modulation to return the control from a continuous signal to a switching signal. The content of [3] described only the classical sliding mode control as well as the sliding mode parameter observers. The research in [3] proved in computer-only simulations that the sliding mode controls produced near unity power factor for the three phase boost converter even without the use of pulse width modulation.

1.4 Thesis Organization

The material in this thesis is ordered so the knowledge presented in each chapter will support each subsequent chapter. Chapters 2 and 3 contain the background information necessary to understand the plant model and the sliding mode control algorithms implemented on the embedded processor. Chapter 2 presents a breakdown of AC-DC boost converters, including diagrams and mathematical models of circuit behavior. Chapter 3 contains the theoretical structure and generic examples of the types sliding mode control algorithms used in the control design. Using the information in the preceding chapters, the controller which is implemented on the DSP is presented in Chapter 4. This control was developed primarily from the algorithms presented in [3]. The process for developing the DSP and computer co-simulation is presented in Chapter 5. The implementation was completed using MathWorks' MATLAB with Simulink, and Texas Instruments' Code Composer Studio. The results of the co-simulation are presented in Chapter 6. These results include a nominal mode simulation and several simulations that test the theoretical performance of the control under varying conditions.

CHAPTER 2

OVERVIEW OF AC-DC POWER CONVERTERS

This chapter introduces the plant model of the system, a three phase full bridge AD to DC boost converter. Hardware topology and math models will be presented first for both single phase and three phase converters. The calculation of power factor is discussed in both a traditional sense and a more in depth sense. After the power factor is presented, the general outline for the control described in the next chapter is given.

2.1 Hardware Topology and Math Models of AC-DC Boost Converter

A boost converter, also known as a step-up converter, is a power converter in which the output voltage is higher than the input voltage. The hardware in a boost converter must contain a switch, a diode, and an energy storage element (i.e., an inductor) [1]. Boost converters can be cumbersome to breakdown in three phases without any former knowledge of how such converters operate, so a single phase converter is described initially.

2.1.1 Single Phase AC to DC Boost Converter

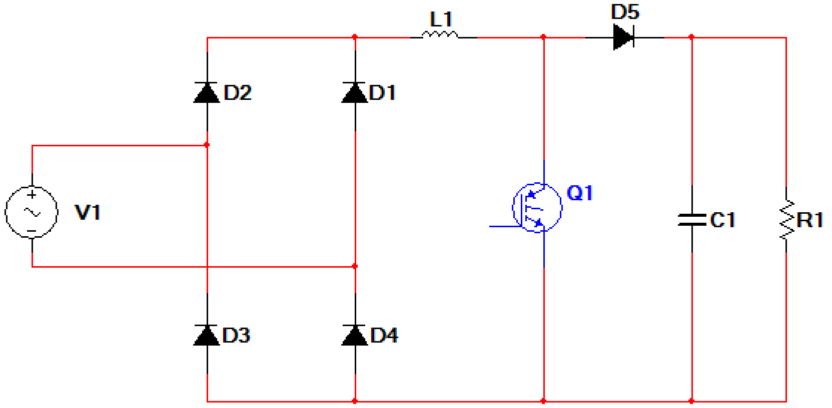


Figure 2.1 Single phase boost converter [11].

A single phase AC/DC boost converter contains an input voltage source which is fed through a full wave rectifier. This rectifier is then placed in series with an inductor. These elements are placed in parallel with a switch and a series combination of a diode and the load. Figure 2.1 shows the topology of a single phase boost converter [11]. The load consists of a basic RC filter which reduces ripple at the output. The switch is a transistor which should either be completely on or off at any time. When the switch is closed, the load will be cut off and the inductor will be shorted. During this time, the inductor acts as a load and will store the energy. When the switch is opened the energy is released and the inductor will add to the energy of the source. The capacitor can then be charged to a higher level than the input source. The diode on the output prevents flyback, a voltage spike produced when the source of energy is suddenly removed from the circuit when the switch closes. It is often referred to as the snubber [2].

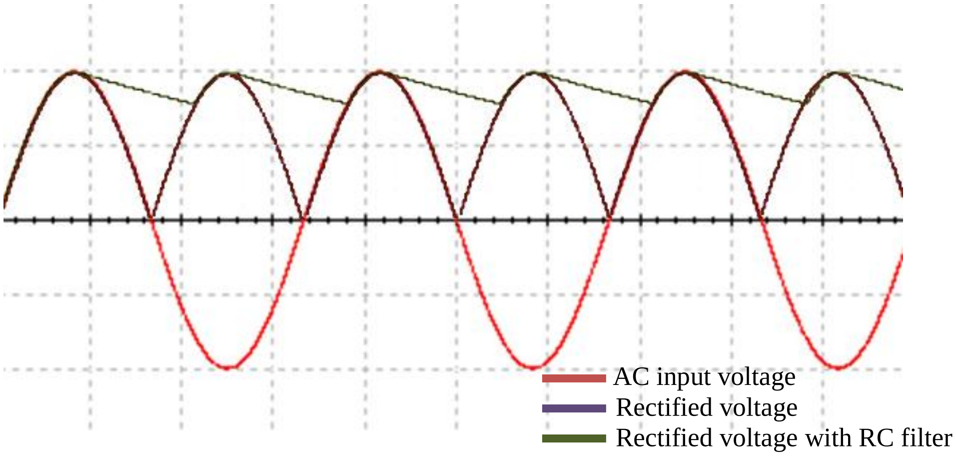


Figure 2.2 Single Phase Rectifier.

Figure 2.2 shows how a single phase AC-DC rectifier works. The input AC voltage, shown in red, is rectified across the full bridge created by the diodes, shown in purple. The voltage then flows through the RC filter where the capacitor is charged and slowly discharges to create the DC voltage, shown in green. The ripple of this voltage can be reduced by increasing the capacitance of the RC filter. The boost converter will have a DC output voltage higher than the input voltage due to the inductor and the switching element. The output will not appear as in the regular rectifier in 2.2. The DC output voltage will be higher than the input voltage and will contain high frequency rippling due to the switching element.

2.1.2 Three Phase AC to DC Boost Converter Topology

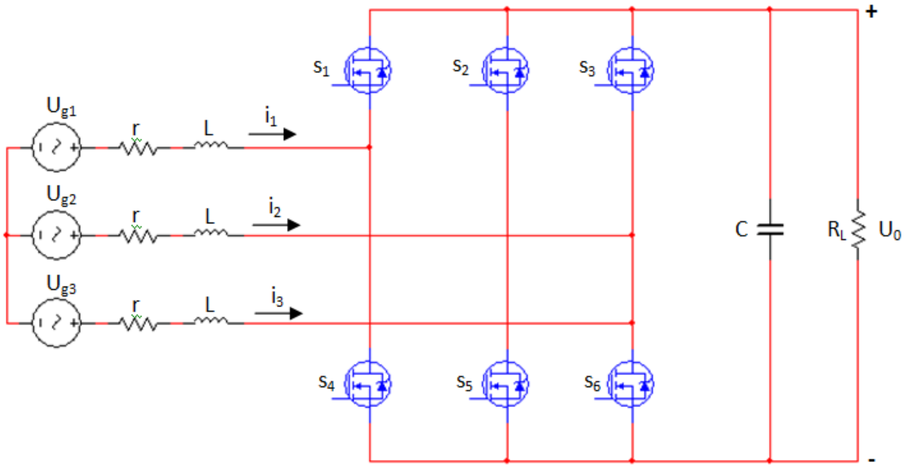


Figure 2.3 Three phase full bridge AC-DC boost converter [1].

The three phase converter operates on the same principles as the single phase. The three phase input allows for even higher DC output voltage levels to be attained. The energy storing elements in the three phase circuit are inductors, just as in the single phase. The switches in the three phase circuit are composed of a transistor wired anti-parallel with a diode. This transistor and diode combination will perform the task of a rectifier as well. The output still consists of an RC filter; the snubber is no longer needed [2].

Due to the arrangement of the switches, the switches that share the same phase should always be in the opposite position of each other. For example, if s1 is on then s4 should be off. When a switch is closed, the diode is shorted and the current is allowed to flow in either direction through the transistor. In practice and in the co-simulation a value of 0 represents an open switch and a value of 1 represents an closed switch. The actual voltage required to open or close a switch on a physical circuit may vary.

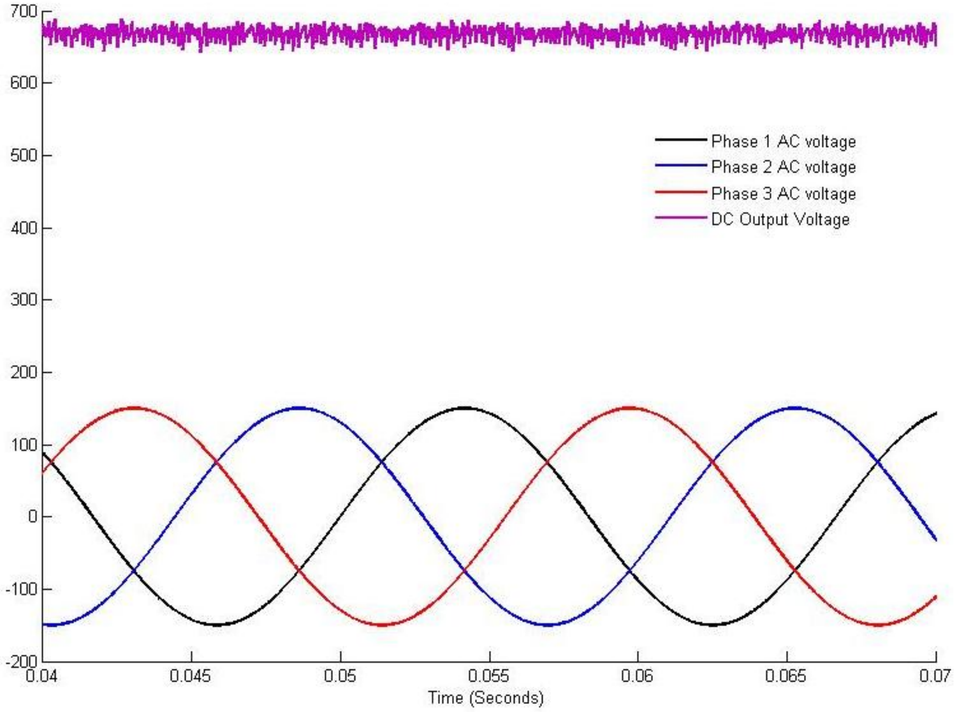


Figure 02.4Three Phase Boost Converter.

Figure 2.4 shows the input and output of a three phase boost converter. The voltage is rectified similarly to the single phase, however the ripple from the capacitor is not as apparent because of the high phase switching frequencies in the DC output voltage. The three phase DC output voltage is very apparently boosted to approximately three times the magnitude of the individual AC input voltages.

2.1.3 Three Phase AC to DC Boost Converter Math Model

The math model for the three phase circuit is described in [3, 4]. The switches are modeled as follows:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{bmatrix}. \qquad (2.1)$$

Each U_i value represents the current state of the switches on the corresponding phase. The value of each U_i will be either -1 or 1. U_1, for example, corresponds to the switches s_1 and s_4. If $U_1 = 1$, then $s_1 = 1$ and $s_4 = 0$. The mathematical model of the circuit behavior is described by [3]:

$$\begin{aligned} \frac{di_1}{dt} &= -\frac{r}{L} i_1 + \frac{1}{3L}\left(2U_{g1} - U_{g2} - U_{g3}\right) - \frac{U_o}{6L}\left(2U_1 - U_2 - U_3\right) \\ \frac{di_2}{dt} &= -\frac{r}{L} i_2 + \frac{1}{3L}\left(2U_{g2} - U_{g1} - U_{g3}\right) - \frac{U_o}{6L}\left(2U_2 - U_1 - U_3\right) \\ \frac{di_3}{dt} &= -\frac{r}{L} i_3 + \frac{1}{3L}\left(2U_{g3} - U_{g1} - U_{g2}\right) - \frac{U_o}{6L}\left(2U_3 - U_1 - U_2\right) \\ \frac{dU_0}{dt} &= -\frac{U_0}{RC} + \frac{1}{2C}\left(i_1 U_1 + i_2 U_2 + i_3 U_3\right). \end{aligned} \qquad (2.2)$$

In (2.2) *r* is the parasitic phase resistance which includes the internal resistance of the voltage source and the impedance of the diodes. *R*, the load resistance corresponds to the resistor labeled R_L in Figure 2.2. *C* is the output capacitance. U_0 is the DC output voltage which is measured across the load. The phase values of current, voltage, and the switching signal are represented by i_j, U_{gj}, and U_j respectively, where j = {1, 2, 3}. The magnitude of each voltage will be labeled E_j. Each phase is not assumed to have the same input voltage magnitude. The equation for the voltage of a single phase is as follows:

$$U_{gj} = E_j \sin(\omega t + \theta), \qquad (2.3)$$

where θ is the phase of the input voltage; $\theta = \left\{0,\ \frac{2\pi}{3},\ -\frac{2\pi}{3}\right\}$ for each phase respectively. The equations in (2.2) can also be presented in the following vector format [3]:

$$\frac{d}{dt}\mathbf{i} = -\frac{r}{L}\boldsymbol{i} + \frac{1}{3L}\boldsymbol{B}\boldsymbol{U}_{\mathrm{g}} - \frac{U_o}{6L}\boldsymbol{B}\boldsymbol{U}$$

$$\frac{d}{dt}U_0 = -\frac{U_0}{RC} + \frac{1}{2C}\boldsymbol{U}^T\mathbf{i}, \tag{2.4}$$

where

$$\mathbf{i} = \{i_1,\ i_2,\ i_3\}^T$$

$$\mathbf{U} = \{U_1,\ U_2,\ U_3\}^T \tag{2.5}$$

$$\boldsymbol{U}_{\mathrm{g}} = \left\{U_{g1},\ U_{g1},\ U_{g1}\right\}^T.$$

B is described by the following singular gain matrix:

$$\mathbf{B} = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}. \tag{2.6}$$

2.2 Power Factor

The power factor is a numerical value between zero and one which quantifies the efficiency of a system. The mathematical formulation of the power factor has been defined two ways. The traditional method for calculating the power factor is to find the ratio of real power flowing into the load to the apparent power in the system. This is calculated using the following equation [12]:

$$PF = \cos(\theta - \psi), \tag{2.7}$$

where θ is the phase of the reference signal, which should be in the correct phase and free of undesired harmonics, and ψ is the phase of the signal which will contain the discrepancy in the phase. In an AC voltage source the phase of the voltage generally serves as the reference and the phase of the current will be ψ. This calculation of the power factor takes into account only the difference of phase between the voltage and the current.

The second method for calculating power factor is more thorough. A signal which is out of phase will not transfer power effectively to the load but harmonic distortion will also affect the efficiency of a system. In the second definition of power factor, a term which quantifies harmonic distortion is also calculated as a signal to noise ratio [3]. The power factor with this added term is defined as follows:

$$PF = PF_h \cdot PF_d. \tag{2.8}$$

PF_d is the phase displacement term defined in (2.7) which quantifies how out of phase the current is with the voltage. PF_h is the harmonic distortion term, which quantifies the effect of undesired harmonics in the current:

$$PF_h = \frac{RMS(i_1)}{RMS(i)}. \tag{2.9}$$

RMS stands for root mean square, a measure of signal energy. The RMS equation for calculating the energy of a signal $f(t)$ is defined as follows [12]:

$$RMS\big(f(t)\big) = \sqrt{\frac{1}{T}\int_0^T f(t)^2 dt}. \tag{2.10}$$

The $RMS(i_1)$ term in (2.6) is the energy of the main harmonic. $RMS(i)$ is the total signal energy. If there are extraneous harmonics in the signal the total signal energy will be

higher than the energy of the main harmonic resulting in a loss of power factor. From (2.7), (2.8), and (2.9) the complete equation for the power factor calculated with the harmonic distortion ratio is as follows [3]:

$$PF = \frac{RMS(i_1)}{RMS(i)} \cdot \cos(\theta - \psi).$$

The power factor is a decimal number between zero and one. In a very efficient system, the power factor will be close to one which is called unity power factor. It is often expressed as a percentage (e.g., a system with a 0.98 power factor is said to have 98% efficiency).

An ideal three phase rectifier would produce a DC output current which is completely smooth and power would be translated exactly between the AC voltage source and the DC output of the circuit. This means that the DC output source would be equal to the active power of the three phase system. This active power would result only from the fundamental harmonic components of voltage and current and there would be no phase difference between the voltage and current [13]. The power factor of an ideal system is one, unity. Of course, ideal systems such as this do not exist in reality.

AC-DC boost converters belong to the family of power supplies known as switched mode power supplies. Switched mode power supplies are designed to be very power efficient compared to traditional linear rectifiers. The inclusion of a switching regulator allows the operator to design controllers which produce a steady, regulated output DC voltage. Although these systems are much more energy efficient than linear rectifiers, the nonlinear elements can still cause the power factor to be less than unity. Also, the switching causes rippling on the output, resulting in a loss of power factor due to harmonic distortion. Another disadvantage to these circuits is that they are very

sensitive to electromagnetic interference so circuit designs including these power blocks must be very carefully designed [1].

2.3 Summary

Power factor signifies the efficiency of a system. The control designed for this application should bring the power factor as close to one as possible. The three phase boost converter is capable of being controlled using the transistor switches. This allows for a control to be designed which will perform the task of power factor correction. Because the converter contains many nonlinear elements and the control signal is naturally a high frequency switching signal, sliding mode controls are very good candidate controls for meeting this objective.

CHAPTER 3

FUNDAMENTALS OF SLIDING MODE CONTROL

Sliding mode control is a nonlinear control which applies high-frequency switching to force a system's state variables to follow what is called a sliding surface in state space. This surface is developed in such a way that the system follows some desired behavior. Sliding mode control is very robust which means the system dynamics are insensitive to matched, unknown, bounded disturbances in the sliding mode. This form of control is attractive for many applications such as robotics, flight control, and electric motor control [5]. In the case of switching mode power conversion described in Chapter 2, high frequency switching is natural to the system since the control signal is designed to switch the transistors on and off. Sliding mode controls are also applicable to controls which are not naturally switching. If high frequency switching is not compatible with the system, high order sliding mode controls, which produce continuous control signals, may be adopted. This chapter includes the background information of the sliding mode control techniques relevant to this project which include: classical sliding mode control, second order sliding mode control, and sliding mode parameter observers.

3.1 Classical Sliding Mode Control

Classical sliding mode controls implement the most general sliding mode design. State feedback is used to stabilize a system by driving the state variables to some desired position to state space. This method of controls can also be used for output tracking, where the output of the system will follow some command value. Sliding mode controls are robust, meaning the control will work properly even in the presence of unknown bounded disturbances and uncertainties. These disturbances and uncertainties must be matched by the control. For information on matching conditions see [5].

3.1.1 General Case of Classical Sliding Mode Control and Definitions

Since the algorithm implemented in the later chapters is used for output tracking, the generalized case described in this section will be similar. The following is a generic plant model for a system to be controlled with a classical sliding mode algorithm:

$$\begin{aligned} \dot{x} &= F(x) + B(x)u + f(x,t) \\ y &= h(x,t) \\ \|f(x_1,x_2,t)\| &\leq L(x,t), \end{aligned} \tag{3.1}$$

where $F(x) \in \mathbb{R}^n$, $B(x) \in \mathbb{R}^m, m < n$ and $h(x,t) \in \mathbb{R}^m$. The norm of $f(x_1,x_2,t)$ is bounded. The desired compensated dynamics for this system will cause y to follow a prescribed command value y_c. The control u must be designed such that this occurs and in such a way that the system is unaffected by the matched disturbance $f(x, t)$. For a more in depth description of matching conditions see [5]. The design process begins with

defining the desired compensated dynamics for the system. In the case of output tracking, the compensated dynamics will be defined so that the system error will be driven to zero as follows [5]:

$$e = y_c - y \rightarrow 0. \tag{3.2}$$

The sliding variable [5] is designed to be a function of these compensated dynamics, as the overall goal of the control will be to drive the sliding variables to zero in finite time. By driving the sliding variable to zero the system trajectory will be driven to a sliding surface in state space where it will remain for all time thereafter. The time in which it takes the sliding variable to reach zero is known as the reaching time.

$$\sigma = \sigma(x) \rightarrow 0, \sigma \in \mathbb{R}^{\mathrm{m}}. \tag{3.3}$$

In the case of this output tracking problem the sliding variables will be defined as follows:

$$\sigma = \sigma(\mathrm{e}, \dot{\mathrm{e}}, \ldots \mathrm{e}^{(\mathrm{r})}). \tag{3.4}$$

In (3.4) *r* is the known as the relative degree of the system [5]. If the disturbances are all matched, they should not appear in the compensated dynamics. First it must be determined that such a control even exists by the existence condition (also called the reachability condition [5] defined by the following Lyapunov analysis:

$$\begin{gathered} v = \frac{1}{2}\|\sigma\|^2 \\ \dot{v} = \sigma^{\mathrm{T}}\dot{\sigma} < 0. \end{gathered} \tag{3.5}$$

The control *u* should be designed in such a way that the compensated dynamics will be enforced. This is done by using the following derivation of (3.5):

$$\dot{v} = \sigma^{\mathrm{T}}\dot{\sigma} \leq -\rho v^{1/2}. \tag{3.6}$$

The control design is split using a concept known as equivalent control [5]. System behavior in the time after which the sliding variable reaches zero (known as the sliding phase) is analyzed in order to determine how the control needs to perform during this time to keep the sliding variable at zero. This control is called the equivalent control. The overall control will be defined as follows:

$$u_i = u_1 + u_{eqi}. \tag{3.7}$$

The control u_1 is the part of the control which drives the sliding variable to zero, this takes affect during the reaching phase [5]. The general format for a classical sliding mode controller is as follows:

$$u_i = u_{eqi} + r_i \text{sign}(\sigma_\text{i}), \tag{3.8}$$

where the coefficients r_i are tunable constants. For a multiple input multiple output system such as the one described above, it is important to keep in mind that each control does not necessarily correspond to a single sliding variable. Optimally, the controls may be decoupled through methods discussed in [5]. This is not always possible however. In the following section, a simple example of classical sliding mode controls is reviewed in order to show some of the basic dynamics.

3.1.2 An Example Implementing Classical Sliding Mode Controls

For the purpose of explaining how sliding mode controls work, a classical sliding mode control will be designed for the following simple double integrator:

$$\begin{aligned} &\dot{x}_1 = x_2 \\ &\dot{x}_2 = f(x_1, x_2, t) + u \qquad (3.9) \\ &|f(x_1, x_2, t)| \leq L. \end{aligned}$$

$f(x,t)$ is an unknown disturbance bounded by L. The classical sliding mode design starts with designing a control function u which will drive x_1, x_2 to zero in finite time in the presence of the disturbance $f(x,t)$. The desired compensated dynamics will be defined as follows:

$$x_2 + cx_1 = 0. \qquad (3.10)$$

With these dynamics there will be no residual effects from the disturbance and the state variables will be driven to zero asymptotically. In order to design the control the state variable [5] will be defined as follows:

$$\sigma = x_2 + cx_1. \qquad (3.11)$$

This variable is designed in such a way that the system will reach the desired position in state space when the sliding variable reaches zero and then the variable will stay at zero perpetually.

$$\sigma = \sigma(x) \rightarrow 0.$$

The system trajectory is designed in such a way that it will arrive at the sliding surface in finite time. If the proposed sliding variable is a valid solution, it will conform to the sliding mode existence condition. The existence condition is defined by the following equation [5]:

$$\sigma\dot{\sigma} < 0. \qquad (3.12)$$

For this example (3.12) can be rewritten:

$$\sigma\dot{\sigma} \leq -\rho|\sigma|, \ \rho > 0. \qquad (3.13)$$

Many factors can be determined by assessing the system behavior during the sliding phase (any time after the reaching time, when $\sigma = 0$). The reaching time can also be derived from this condition. In order to determine the reaching time, equate the inequality in (3.13) and solve for $\dot{\sigma}$ as follows:

$$\dot{\sigma} = -\rho. \tag{3.14}$$

The variable ρ is the bounded constant from (3.13), therefore integrating of (3.14) produces the following:

$$\sigma(\mathrm{t}) - \sigma(0) = -\rho(t - 0). \tag{3.15}$$

$\sigma(0)$ is a constant which is dependent on the initial conditions of the system and $\sigma(t)$ will equal zero at the reaching time. Solving (3.15) for the reaching time will produce the following result:

$$t_r \leq \frac{|\sigma(0)|}{\rho}. \tag{3.16}$$

From (3.16) it can be seen that the reaching time will depend on the initial conditions of the system and the bounded constant ρ.

The next step in the design process is to determine the necessary control behavior in terms of the sliding variable in order to design the control u. For this, the equivalent control analysis is used [5]. This control is derived from the analysis of the system during the sliding phase, the sliding mode dynamics, and the existence condition. During the sliding phase the following events occur:

$$\begin{gathered} \sigma = x_2 + cx_1 = 0 \\ x_2 = -cx_1. \end{gathered} \tag{3.17}$$

This means the system of equations in (3.9) can be defined as follows:

$$\begin{gathered} \dot{x}_1 = -cx_1 \\ x_2 = -cx_1. \end{gathered} \tag{3.18}$$

A very profound property of the sliding phase is that the order of the system has been reduced. Also, since disturbance $f(x_1, x_2, t)$ no longer appears in the state equations, it will not affect the output (the control is robust). The sliding mode dynamics are defined by taking the first derivative of the sliding variable:

$$\dot{\sigma} = f(x_1, x_2, t) + u + cx_2. \tag{3.19}$$

Substituting (3.19) into the existence condition (3.14) yields

$$\sigma(f(x_1, x_2, t) + u + cx_2) \leq -\rho|\sigma|. \tag{3.20}$$

The equivalent control is defined using (3.19). The disturbance will not affect the control in the sliding phase but the cx_2 term must be compensated for. Therefore, the equivalent control is defined as follows:

$$u_{eq} = -cx_2. \tag{3.21}$$

The overall control u will be defined as the equivalent control plus another term which will cause the trajectory to reach the sliding phase in finite time. This extra term is will be called v. The overall control is then defined as follows:

$$u = u_{eq} + v. \tag{3.22}$$

Substituting (3.20) and (3.21) into (3.13) yields

$$\sigma f(x_1, x_2, t) + \sigma v \leq |\sigma||f(x_1, x_2, t)| + \sigma v \leq -\rho|\sigma|. \tag{3.23}$$

The magnitude of the unknown disturbance $f(x_1, x_2, t)$ is bounded by L according to (3.9). Substituting L and factoring out the magnitude of σ simplifies (3.23) to

$$|\sigma|\big(L + v\,\mathrm{sign}(\sigma)\big) \leq -\rho|\sigma|. \tag{3.24}$$

Solving (3.24) for v yields

$$v = -(\rho + L)\mathrm{sign}(\sigma). \tag{3.25}$$

To solve for u substitute (3.18) and (3.22) into (3.19) to obtain the following control equation:

$$u = -cx_2 - (\rho + L)\text{sign}(\sigma). \tag{3.26}$$

The control in (3.26) will drive the state variables to the sliding surface in the finite reaching time. Since the system will behave according to (3.13), σ will equal zero but $\dot{\sigma}$ will not. As long as the first derivative (the trajectory velocity) is not zero, σ will continue to move after it reaches the sliding surface, causing what is known as chattering [5]. The added trajectory is due to the sign function, this function will change values once the trajectory crosses the sliding surface. Therefore, the chattering motion pushes the trajectory back and forth across the sliding surface on a very small scale (i.e., 10^{-4}) as opposed to the desired behavior in which the trajectory follows the sliding surface exactly. Chattering may have some affect on the output of the system but may be attenuated or completely eliminated using second order sliding mode techniques [6, 14].

The example is illustrated using the system model in (3.9) with initial conditions $x_1 = 2,\ x_2 = -1$ and $|f(x_1, x_2, t)| = 2\sin(t)$, therefore $|f(x_1, x_2, t)| \leq L = 2$. Uncontrolled, the state variables will exhibit the following behavior.

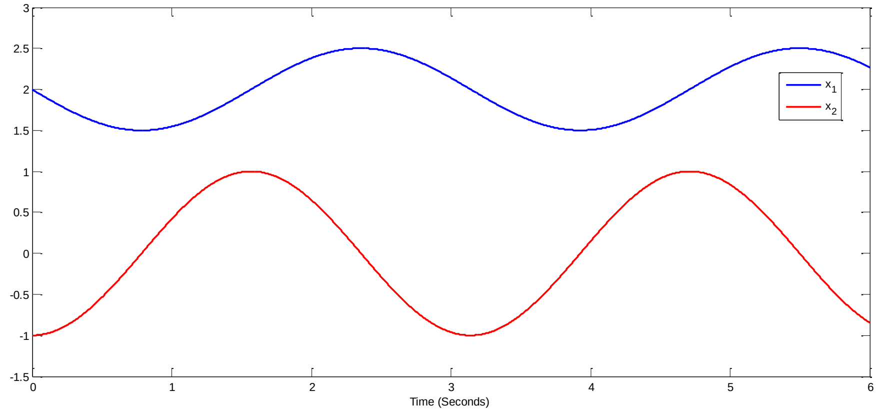


Figure 3.1 Uncontrolled sliding variables with bounded disturbance.

Then applying the control from (3.26) with $\rho = 4$ and $c = 2$ yields the following.

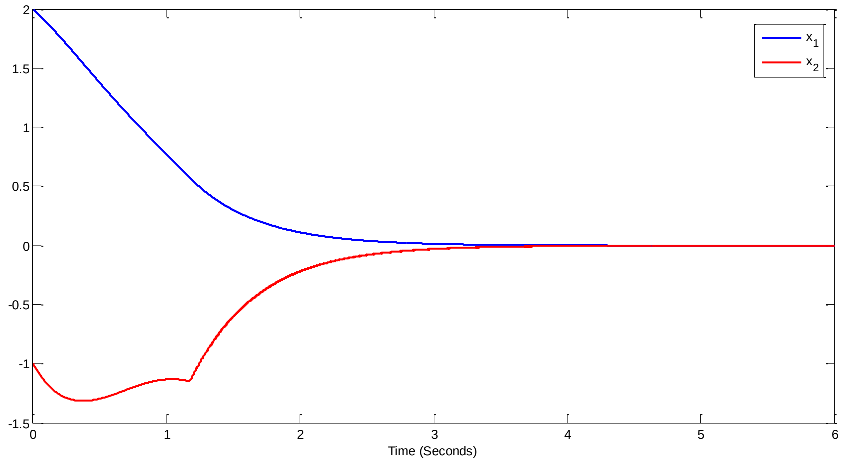


Figure 3.2 State variables controlled using the classical SMC technique.

In Figure 3.2 the state variables go to zero asymptotically during the reaching phase. The transition from the reaching phase to the sliding phase is apparent in x_2. This transition occurs shortly after the one second mark. The signals go to zero with no apparent residual effects from the disturbance.

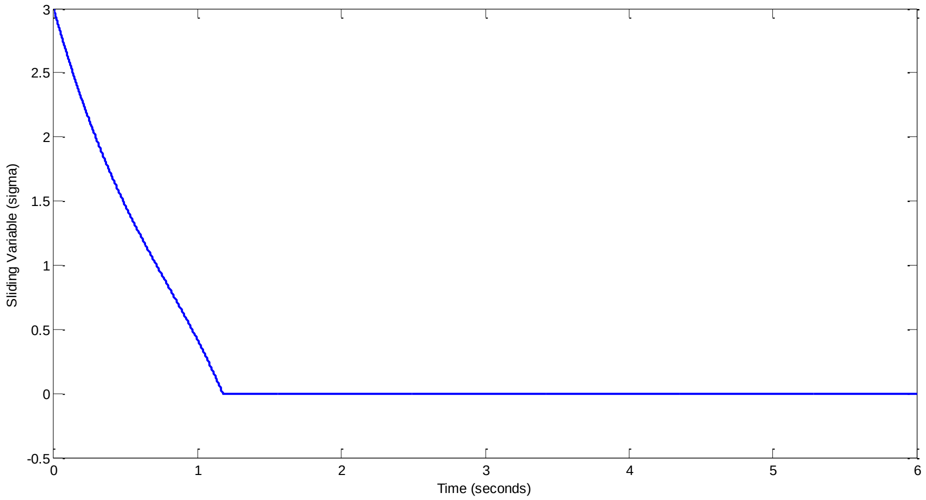


Figure 3.3 The sliding variable σ.

Figure 3.3 shows that the sliding variable reaches zero shortly after one second (the reaching time) and remains very close to zero for the remaining time (the sliding phase).

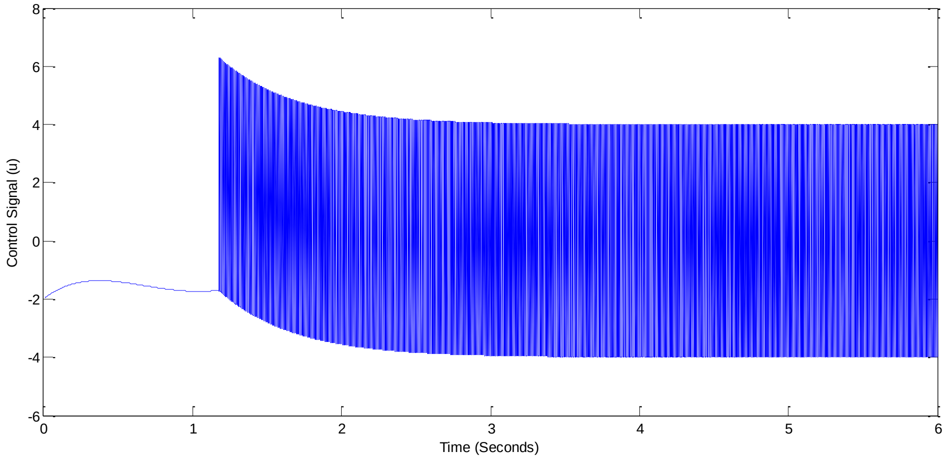


Figure 3.4 The control signal *u*.

The control signal u begins to exhibit the high frequency switching behavior in the sliding phase, as expected. The effect of this switching can be seen as chattering. This can only be seen by a much closer inspection of the sliding variable or the output.

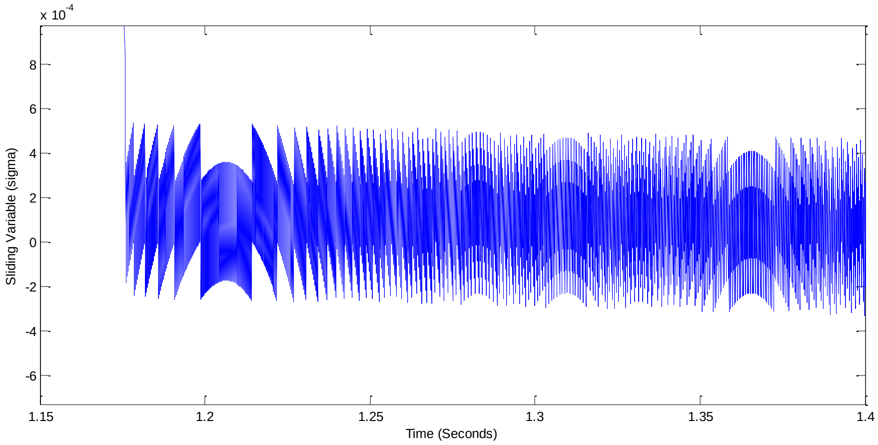


Figure 3.5 A closer view of the sliding variable during the sliding phase.

Figure 3.5 shows a closer inspection of the sliding variable. The chattering is more apparent but it can be seen that the magnitude of the chattering signal is very small, in the range of 10^{-4}. Using higher order sliding mode techniques would improve these results.

3.2 Second Order Sliding Mode Control (2-SMC)

Second order sliding mode controls can be used to produce a more accurate controller, attenuate the effects of chattering, and/or generate continuous controls. Continuous controls are very often more desirable in many systems. These controls are designed around the concept of driving *both* $\dot{\sigma}$ and σ to zero in finite time [7, 14]. The sliding variable designed will be non-linear, leads to the creation of a sliding manifold (as opposed to a sliding surface). As with the classical control, the trajectory of the system will ideally reach this manifold in finite time and remain there for all time thereafter. These controls are also robust to bounded disturbances.

3.2.1 Twisting Controls

The twisting controller was the first proposed 2-SMC controller [7]. Using the example in (3.9), the sliding variable would be defined as follows:

$$\sigma = x_1. \tag{3.27}$$

The dynamics are then derived as follows:

$$\dot{\sigma} = \dot{x}_1 = x_2 \tag{3.28}$$

$$\ddot{\sigma} = f(x_1, x_2, t) + u.$$

From (3.28) it can be seen that the second derivative is proportional to the control. The disturbance $f(x_1, x_2, t)$ is unknown and bounded. Since the control appears in the second derivative, the relative degree of the system is 2. Using twisting control [6], the sliding variable σ and $\dot{\sigma}$ are both driven to zero in finite time. The state variables are also driven to zero in finite time, which is even better than asymptotic convergence. The control algorithm for this method is defined as follows [6]:

$$u = -\left(\alpha_1 \mathrm{sign}(\sigma) + \alpha_2 \mathrm{sign}(\dot{\sigma})\right),\ \alpha_1 > \alpha_2 > 0.$$

$$(\ \alpha_1 +\ \alpha_1)K_m - C > (\ \alpha_1 -\ \alpha_2)K_M + C\ ) + C \tag{3.29}$$

$$(\alpha_1 \text{ - } \alpha_1)K\mathrm{m} > C.$$

The *K* constants are related to the boundaries on the disturbances and uncertainties. Generally, it is difficult to design the α values to match these specifications exactly, so they may be tuned for optimal performance. This algorithm provides the finite time convergence of the state variables and provides very accurate control. This method is insensitive to disturbances, which is even stronger than the robustness of classical sliding mode controls. However, it does not produce a continuous control [6].

3.2.2 Prescribed Convergence Law

Another second order technique is called the prescribed convergence law [6]. This control also drives σ and $\dot{\sigma}$ are both driven to zero in finite time. Like the twisting control, in order to implement the prescribed convergence law, the system must be of

relative degree two (the dynamics in (3.28) must hold, $f(x_1, x_2, t)$ is unknown and bounded by L). The control for the prescribed convergence law is written as follows:

$$u = -\alpha \text{sign}\left(\dot{\sigma} + \lambda|\sigma|^{1/2}\text{sign}\sigma\right),\ \alpha, \lambda > 0. \tag{3.30}$$

$$\alpha K_m - C > \lambda^2/2.$$

The parameters λ, α are tuned similarly to the twisting algorithm. The prescribed convergence algorithm also produces a discontinuous control and is insensitive to disturbances [6]. There are methods for creating continuous controls using these 2-SMC methods which are discussed in [6, 14]. For systems which are not of a relative degree two, there is a continuous sliding mode control algorithm which can be applied. This is known as the super twisting control algorithm.

3.2.3 Super Twisting Sliding Mode Controls

The second order sliding mode control method which will be implemented in this thesis is known as the super twisting control, which is one type of second order sliding mode control discussed in [6, 14]. This control may be used on a system with relative degree one and will produce a continuous control signal. In order to use super twisting control, the dynamics of $\dot{\sigma}$ must also be taken into account. Considering the double integrator from the (3.9), the sliding mode dynamic can be rewritten:

$$\dot{\sigma} = \varphi(x_1, x_2, t) + u. \tag{3.31}$$

If $\varphi(x_1, x_2, t) \equiv 0$, then the following continuous control would drive σ to zero in finite time:

$$u = -c|\sigma|^{1/2}\text{sign}(\sigma),\ c > 0. \tag{3.32}$$

Assume instead that φ is a smooth function with a first derivative bounded as follows:

$$|\dot{\varphi}(x_1, x_2, t)| \leq M. \tag{3.33}$$

With this new condition, the following control algorithm can be used [6]:

$$\begin{aligned} u &= -\alpha|\sigma|^{1/2}\mathrm{sign}(\sigma) - \dot{\xi} \\ \xi &= \beta\mathrm{sign}(\sigma) \\ \alpha &= \frac{1}{2}\sqrt{M} \\ \beta &= 1.1M. \end{aligned} \tag{3.34}$$

After the reaching phase, the first term of the control, $\alpha|\sigma|^{1/2}\mathrm{sign}(\sigma)$, will become zero since σ goes to zero and therefore is continuous during the sliding phase. The second term, $\beta \int \mathrm{sign}(\sigma)\mathrm{d}t$, will become equal to $\varphi(x_1, x_2, t)$. The discontinuous sign term is concealed by the integral. Therefore the chattering, as observed in the classical control, will be attenuated and both σ and $\dot{\sigma}$ are driven to zero in finite time [6].

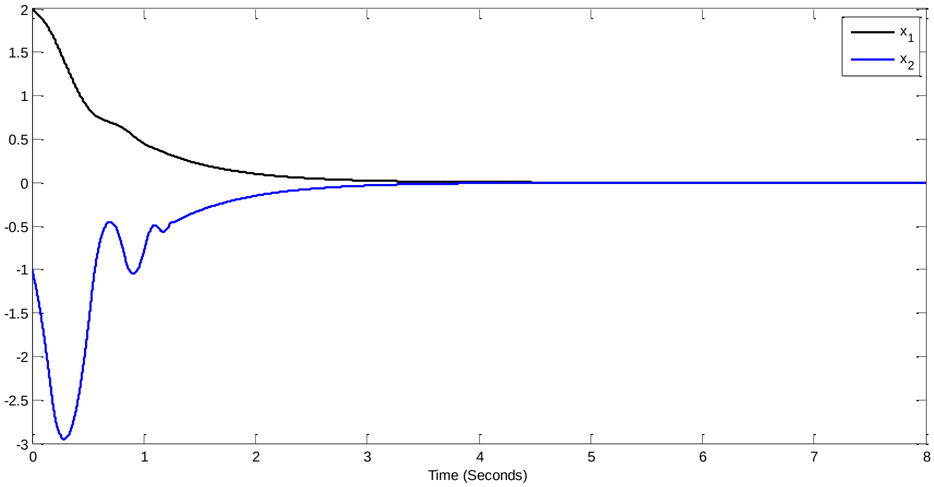


Figure 3.6 State variable behavior controlled using the super twisting algorithm.

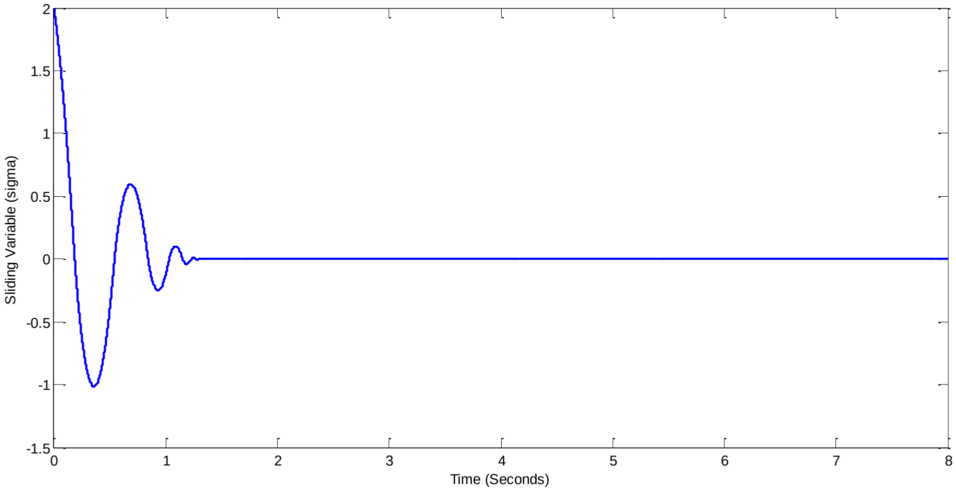


Figure 3.7 Sliding variable from the super twisting control.

Figure 3.6 shows that the state variables will still go to zero in finite time as desired. Figure 3.7 shows the sliding variable. The sliding variable will still exhibit some chattering but on a much smaller scale (10^{-6}). The magnitude of the chattering in the sliding phase is proportional to the evaluation step size. The reaching phase behavior is slightly different but the system stabilizes quickly.

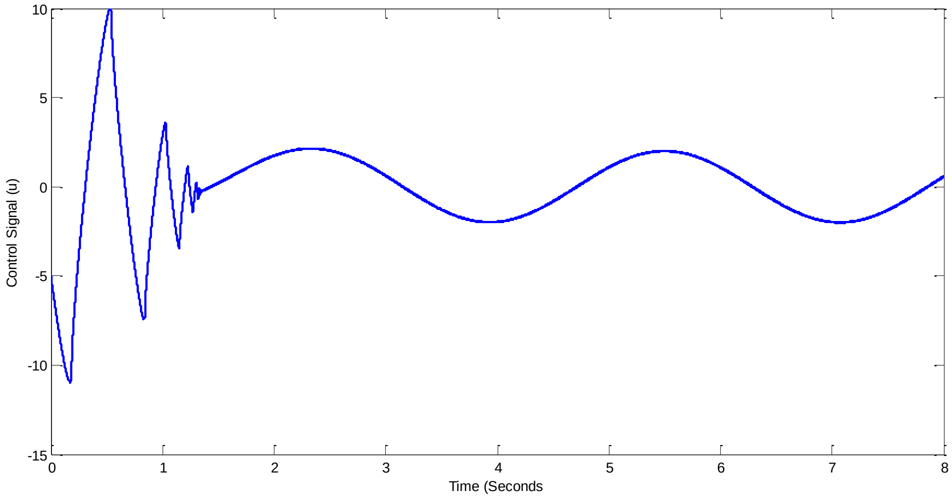


Figure 3.8 The control signal u of the super twisting sliding mode controller.

The control signal of a super twisting algorithm is continuous, as shown in Figure 3.8. This is because the switching signal in the equivalent control is masked by an integral. Continuous behavior is desirable in many applications in which the actuator is not compatible with a switching signal.

3.3 Sliding Mode Observers and Differentiators

Sliding mode observers [7] are used to determine the value of a parameter which may not be measured directly. These observers are beneficial because they exhibit the same robustness as sliding mode controllers, which means they will work effectively in the presence of bounded disturbances and bounded measurement uncertainties. A sliding mode differentiator exhibits this quality as well. The most basic sliding mode observer is known as the Utkin observer, which is described in [5, 7]. The design process begins similarly to a Luenburger observer, except that when the injection term algorithm is designed sliding mode algorithms are implemented. Using the double integrator example from (3.9), assume that x_1 (position) can be measured but x_2 (velocity) cannot. The first step in building an observer for x_2 is to define an observation algorithm, as follows:

$$\dot{\hat{x}}_1 = v. \tag{3.36}$$

The expression v in (3.36) is known as the injection term. The injection term is designed to drive the observation error to zero in finite time and thus $\hat{x}_1 \rightarrow \mathrm{x}_1$ and $\hat{x}_2 \rightarrow \mathrm{x}_2$. It works similarly to the control term in previous examples. The observer error is defined as follows:

$$e = \hat{x}_1 - x_1. \tag{3.37}$$

Taking the first derivative of (3.37) and substituting in the value of $\dot{x}_1$ from (3.9) yields

$$\dot{e} = -x_2 + v. \tag{3.38}$$

In order to drive e to zero, v must be designed such that $\hat{x}_1$ will become exactly equal to x_1. The injection term can be defined using either classical or second order sliding mode techniques.

Using a traditional sliding mode control algorithm the injection term will be designed similarly to the control u as follows [7]:

$$v = -(\rho + L)\operatorname{sign}(e), \ \ |x_2| \leq L\,. \tag{3.39}$$

According to the concept of equivalent control and from (3.38), the sliding mode dynamics are as follows:

$$\dot{e} = -x_2 + v_{eq}. \tag{3.40}$$

The observed variable x_2 will then be exactly equal to v_{eq} which can be found by using a low pass filter on the term v. The low pass filter removes the high frequency terms caused by the switching of the sign function [5]. This example system would require only a very basic low pass filter as follows:

$$LPF = \frac{1}{\tau s + 1}. \tag{3.41}$$

More complicated systems such as the AC-DC converter described in this thesis require higher order low pass filters. Revisiting the double integrator example, assume the velocity (x_2) cannot be measured and no control is used. The same initial conditions are used as in the first example. A value of $\tau = 0.01$ will be used for the filter. A classical sliding mode observer will be used to find the velocity.

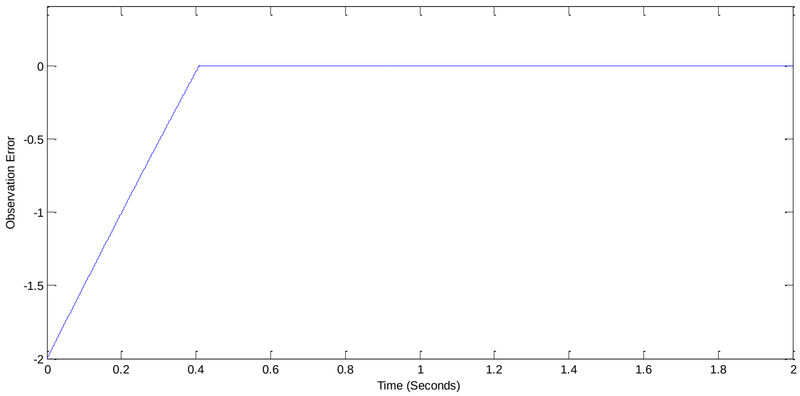


Figure 3.9 Observation error *e*.

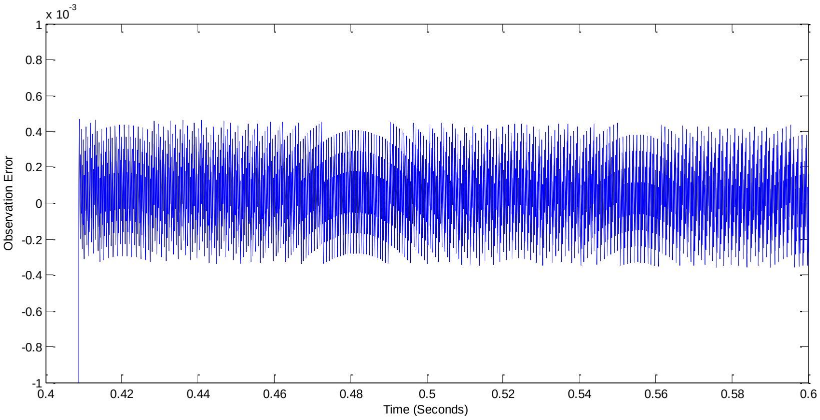


Figure 3.10 Close inspection of observation error with chattering.

Figure 3.9 and Figure 3.10 show the observer error, which acts as the sliding variable in the sliding mode observer [7]. There is some chattering which will affect the performance of the observer.

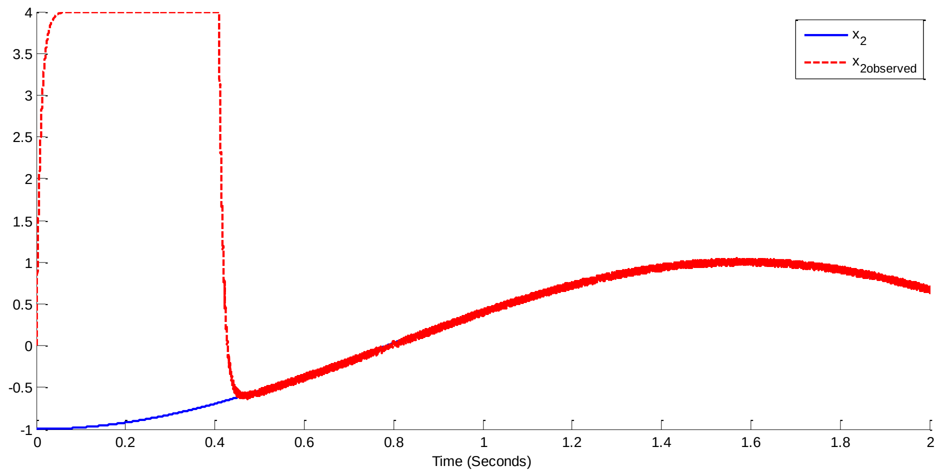


Figure 3.11 Observer output tracking velocity.

Figure 3.11 shows the output of the observer, after reaching time, the observer starts tracking the actual velocity very quickly. There are higher frequencies present which may be removed by using a better filter or by using a second order sliding mode technique [6].

Using a super twisting algorithm will remove the need for a low pass filter, and improve quality and accuracy of the observer output by driving e and $\dot{e}$ to zero in finite time. This super twist is used within the already defined system simply by redefining the injection term. Using the double integrator example, all equations will be the same through (3.38). The injection term v is then defined as follows using the super twisting control algorithm [6]:

$$\begin{aligned} v &= -\alpha|e|^{1/2}\mathrm{sign}(e) - \dot{\xi} \\ \xi &= \beta\mathrm{sign}(e) \\ \alpha &= \frac{1}{2}\sqrt{M} \\ \beta &= 1.1M. \end{aligned} \tag{3.42}$$

The injection term v will now be continuous and exactly equal to v_{eq} without the use of low pass filtering.

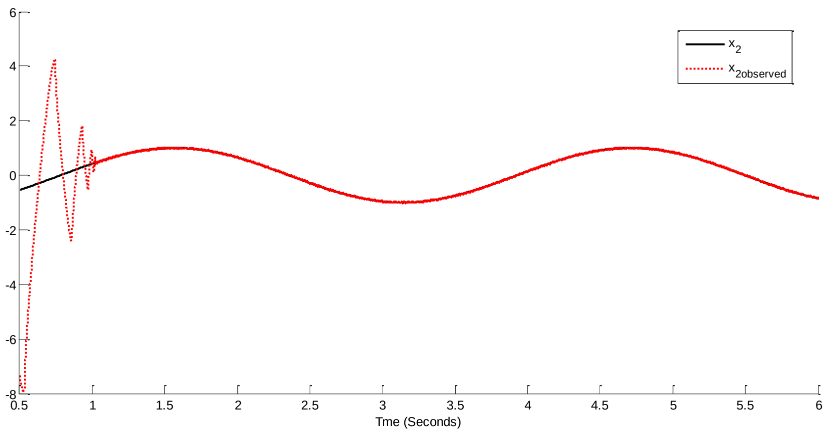


Figure 3.12 Super twisting parameter observer output.

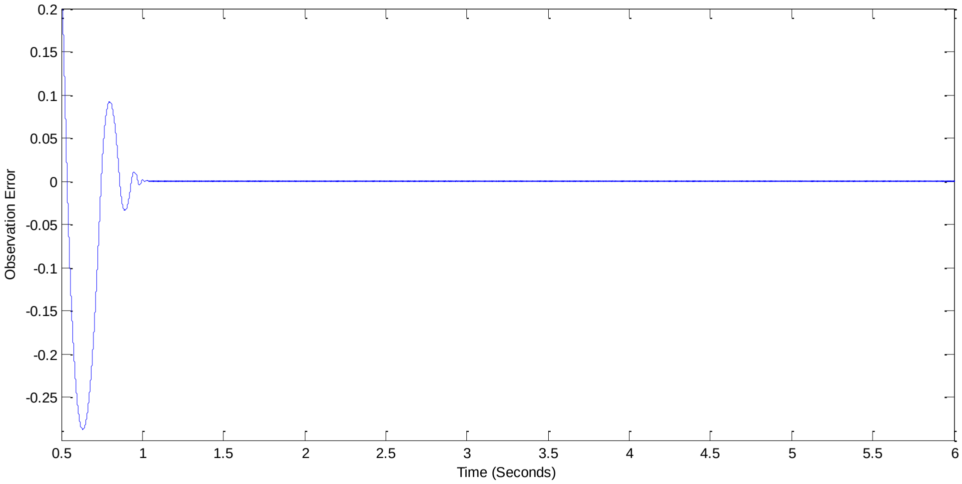


Figure 3.13 Observation error from super twisting observer.

Figure 3.12 shows the observed velocity using the super twisting algorithm. Once the sliding phase is reached, the observed variable is tracked very closely. The

observation error is shown in Figure 3.13. The chattering is very attenuated since it is proportional to the evaluation step size.

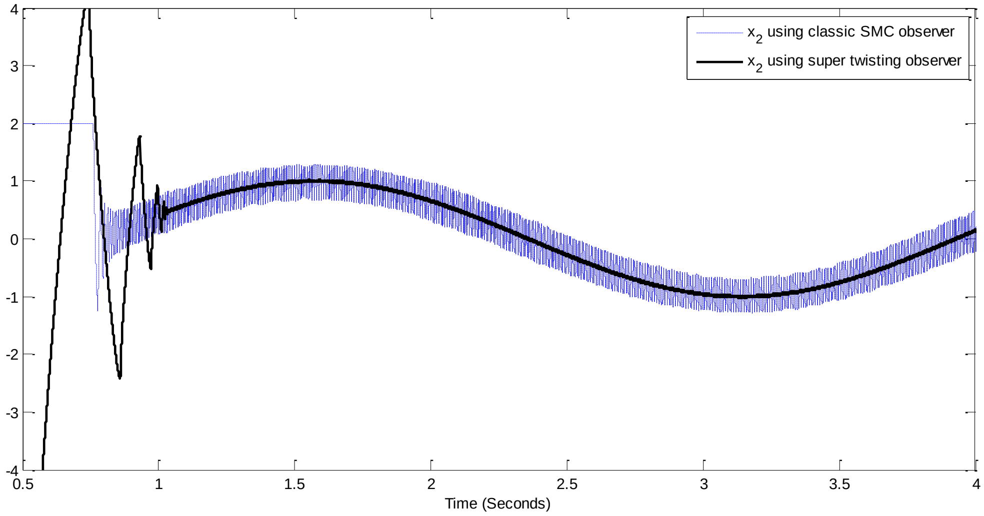


Figure 3.14 Comparison of the classical sliding mode observer and the super twisting sliding mode observer

Figure 3.14 shows a direct comparison of the classical sliding mode observation method, using the low pass filter to calculate the equivalent injection term and the super twisting observer which does not require a filter. The super twisting observer tracks the actual value much more accurately than the classical method without the effects created by high frequency switching [6, 7].

3.4 Summary

Classical sliding mode control is a robust control law that forces a system to follow a predesigned trajectory in state space. There is some chattering introduced by this form of control but it is very useful for systems that require switching type control

inputs such as the AC-DC power converters. Super-twisting controls, a second order sliding mode control, reduce chattering and produce a continuous output. A sliding mode observer can be created to determine a variable which cannot be directly measured. An observer may use a low pass filter, or the filter may be substituted with a super-twisting algorithm which produces a smoother and more accurate result.

CHAPTER 4

POWER FACTOR CORRECTION USING SLIDING MODE TECHNIQUES

This chapter includes the derivation of the sliding mode control algorithms used for correcting the power factor in a three phase AC-DC boost power converter. These control algorithms were constructed in a model in Simulink and loaded to the embedded processor for the tests explained the following chapters. Several sliding mode control designs were tested in a full computer simulation in [3, 4]. In the DSP implementation the classical sliding mode control method [5] was applied for the main controller, using only a sign function instead of pulse width modulation. The load resistance estimator was constructed using a super twisting algorithm [7, 6].

4.1 Assumptions and Control Objectives

For convenience, the math model of the three phase AC-DC boost power converter presented in (2.2) and (2.4) will be shown below:

$$\begin{aligned}
\frac{di_1}{dt} &= -\frac{r}{L}i_1 + \frac{1}{3L}\left(2U_{g1} - U_{g2} - U_{g3}\right) - \frac{U_o}{6L}(2U_1 - U_2 - U_3) \\
\frac{di_2}{dt} &= -\frac{r}{L}i_2 + \frac{1}{3L}\left(2U_{g2} - U_{g1} - U_{g3}\right) - \frac{U_o}{6L}(2U_2 - U_1 - U_3) \\
\frac{di_3}{dt} &= -\frac{r}{L}i_3 + \frac{1}{3L}\left(2U_{g3} - U_{g1} - U_{g2}\right) - \frac{U_o}{6L}(2U_3 - U_1 - U_2) \\
\frac{dU_0}{dt} &= -\frac{U_0}{RC} + \frac{1}{2C}(i_1 U_1 + i_2 U_2 + i_3 U_3).
\end{aligned} \tag{2.2}$$

$$\begin{aligned}
\frac{d}{dt}\mathbf{i} &= -\frac{r}{L}\boldsymbol{i} + \frac{1}{3L}\boldsymbol{B}\boldsymbol{U}_{\mathbf{g}} - \frac{U_o}{6L}\boldsymbol{B}\boldsymbol{U} \\
\frac{d}{dt}U_0 &= -\frac{U_0}{RC} + \frac{1}{2C}\boldsymbol{U}^T\mathbf{i}.
\end{aligned} \tag{2.4}$$

In order for the sliding mode control to be implemented, several assumptions will be made about the three phase converter and the control output [4]:

1. The three input voltages must be of the same frequency. If the voltage frequency is increased or decreased all the voltage frequencies must be increased and decreased concurrently.
2. The three phases must have the same inductance and impedance and the three phases evenly contribute the output power.
3. The control signals $\mathbf{U} = \{U_1, U_2, U_3\}^T$ must never share the same value. That is, the situations $\mathbf{U} = \{1,\ 1,\ 1\}$ and $\mathbf{U} = \{-1, -1, -1\}$ must never occur. The vector **U** only appears in the system equations in (3.5) through the gain matrix B, described in (3.7). This gain matrix is singular; therefore the occurrence of the above situations will result in the loss of controllability of the system. It is

assumed that this situation does not occur and this assumption has been backed up through simulation analysis and the DSP implementation in this thesis [3].

4. The phase voltages and currents, the output DC current, and the control signals U are measurable. The values of the phase inductance and the capacitor in the RC filter on the output are known and do not change without restarting the system and changing control parameters. The value of the parasitic phase resistance can be measured by means of a resistance estimator similar to that of the output resistance but is not measured as such in this implementation [3]. This value is generally known and not expected to change. The phase voltage frequency, which could vary in real time, may also be measured using a frequency detector; in this implementation it is assumed to be known.

The primary goal of the control is to correct the power factor in all three phases and to drive the DC output voltage to a prescribed value. In order to do this the following objectives must be met:

1. The input currents must be completely in phase with their corresponding voltages. These currents should contain minimal extraneous harmonics. Ideally, they should contain only the main harmonic of the phase voltage.
2. The output DC voltage level must reach a desired voltage level and remain there even if the load resistance changes. The AC portion of the output voltage should be reduced as much as possible.

4.2 Current Error Tracking and Ideal Current Generation

The sliding mode algorithm chosen for implementation is the classical sliding mode control defined in [3]. The sliding control will be designed so that the current tracks a desired current profile which will be generated by the controller. Just as in the example in chapter three, the first step in control design is to define the sliding variables [ES]; in this case there will be three.

$$\sigma = \{\sigma_1,\ \sigma_2,\ \sigma_3\}^{\mathrm{T}}. \tag{4.1}$$

Since this is handled as an output tracking problem, each σ will represent the current tracking error for each phase. The control will drive these sliding variables and thereby, the error, to zero in finite time. The sliding variables are designed specifically as follows:

$$\sigma_j = i_j - i_{dj},\ j = \{1,\ 2,\ 3\}, \tag{4.2}$$

where i_j is the phase current and i_{dj} is the desired current profile generated by the controller. From this (4.2) it can be seen that the control of voltage has been reduced to simply controlling only the phase currents.

The phase currents in (4.2) are measurable based on previous assumptions. The ideal current must be generated by the controller. The frequency of each ideal phase current should be the same as the input voltage. This frequency may be measured but is assumed to be known in this implementation. The phase for each current should correspond to the phase; according to (2.3), $\theta_{\mathrm{j}} = \left\{0,\ \frac{2\pi}{3},\ -\frac{2\pi}{3}\right\}$ for $i_{dj} = \{i_{d1},\ i_{d2},\ i_{d3}\}$ respectively. The magnitudes of the currents need to be calculated by the controller. These magnitudes should be selected so that the DC output voltage will reach the desired level and be maintained at that level. Also the DC power should be balanced between the

input and the output sides of the converter [13]. Maintained power balance allows for formulation of the desired current levels based on the input and output DC power formulas. The formula for the input DC power of the jth phase $\langle IP_j \rangle_{DC}$, which could be calculated for any of the three phases, is derived as follows [3]:

$$\begin{aligned} \langle IP_j \rangle_{DC} &= \langle i_j U_{gj} - r i_j^2 \rangle_{DC} \\ &= \langle I_j E_j \sin\left(\omega t - \theta_j\right)^2 - r I_j^2 \sin\left(\omega t - \theta_j\right)^2 \rangle_{DC} \\ &= \tfrac{1}{2}\left(I_j + E_j - r I_j^2\right). \end{aligned} \tag{4.3}$$

I_d is the jth phase current magnitude. The input current should have the same magnitude as the desired current, therefore (4.3) can be rewritten as follows:

$$\langle IP_j \rangle_{DC} = \tfrac{1}{2}\left(I_{dj} + E_j - r I_{dj}^2\right). \tag{4.4}$$

The output side DC voltage can be calculated by the following:

$$\langle OP \rangle_{DC} = \frac{V_d^2}{R}. \tag{4.5}$$

V_d is the desired DC output voltage and R is the load resistance. Since it is assumed that each input phase contributes evenly to the output, the DC input power for one phase should be exactly equal to one third of the DC output voltage. The equations are related as follows:

$$\frac{1}{2}\left(I_{dj} + E_j - r I_{dj}^2\right) = \frac{1}{3}\frac{V_d^2}{R}. \tag{4.6}$$

Solving (4.6) for I_{dj} yields two real solutions as follows:

$$I_{dj} = \frac{E_j}{2R} \pm \sqrt{\frac{{E_j}^2}{4R} - \frac{2}{3}\frac{{V_d}^2}{rR}}. \tag{4.7}$$

In order for (4.7) to produce a real solution, the following restriction is placed on V_d:

$$V_d \le E_j \sqrt{\frac{3R}{8r}}. \tag{4.8}$$

Only the answer to (4.7) with the subtraction is optimal for minimal energy consumption. Therefore, the equation for the ideal current magnitude for each phase is as follows:

$$I_{dj} = \frac{E_j}{2R} - \sqrt{\frac{{E_j}^2}{4R} - \frac{2}{3}\frac{{V_d}^2}{rR}}. \tag{4.9}$$

Therefore the equation for the ideal current for the jth phase will be defined as follows [3]:

$$i_{dj} = I_{dj} \sin\left(\omega t + \theta_j\right). \tag{4.10}$$

4.3 Development of Classical Sliding Mode Controller

The sliding variables for the sliding mode controller have been defined in (4.1) and (4.2). The next step in designing the controller is to derive the sliding mode dynamics by taking the first derivative of the sliding variables [5]. In terms of the system vectors, this derivative yields the following [3]:

$$\begin{aligned} \dot{\sigma} &= \frac{d}{dt}\mathbf{i} - \mathbf{I_d}\boldsymbol{\omega}\mathbf{cos} \\ &= -\frac{r}{L}\mathbf{i} - \frac{U_o}{6L}\mathbf{BU} + \frac{1}{3L}\mathbf{BU_g} - \mathbf{I_d}\boldsymbol{\omega}\mathbf{cos} \\ &= \left(-\frac{r}{L}(\sigma + \mathbf{I_d}\mathbf{sin}) + \frac{1}{3L}\mathbf{BU_g} - \mathbf{I_d}\boldsymbol{\omega}\mathbf{cos}\right) - \frac{U_o}{6L}\mathbf{BU} \\ &= \varphi - \frac{U_o}{6L}\mathbf{BU}, \end{aligned} \tag{4.11}$$

where $\dot{\sigma} = \{\dot{\sigma}_1,\ \dot{\sigma}_2,\ \dot{\sigma}_3\}^T$ and $\varphi = \{\varphi_1,\ \varphi_2,\ \varphi_3\}^T$. Since the control appears after the first differentiation the relative degree of each phase is said to be one. The total relative degree of the system is three. Because **B** is a singular matrix, the controls cannot be decoupled without the use of transformed sliding variables [5]. The use of transformed sliding variables in this application was tested in [4] and proved only to degrade the overall performance of the controller, and therefore were not implemented in this study. φ_j is defined as follows:

$$\varphi_j = \frac{1}{3L}\left(3E_j - \sum_{k=1}^{3} E_k - 3rI_{dj}\right)\sin(\omega t + \theta_j) \tag{4.12}$$

$$-I_{dj}\omega\cos(\omega t + \theta_j) + \frac{U_o}{6L}\sum_{k=1}^{3} U_k.$$

The control signals U_j are defined as follows:

$$U_j = \operatorname{sign}(\sigma_j). \tag{4.13}$$

Since the control signals can only have a value of -1 or 1, ρ must be equal to 1 [3]. It was found in [4] that due to Kirchhoff's current law, only two phase currents have to be tracked. Therefore, the third sliding variable can be defined by the following formula:

$$\sigma_3 = \int_0^t (U_1 + U_2 + U_3)\, dt. \tag{4.14}$$

Combining (4.2) for j = 1, 2 and (4.13), the equations for σ_j can now be defined as follows:

$$\begin{aligned} \sigma_1 &= i_1 - i_{d1} \\ \sigma_2 &= i_2 - i_{d2} \\ \sigma_3 &= \int_0^t (U_1 + U_2 + U_3)\, dt. \end{aligned} \tag{4.15}$$

4.4 Load Resistance Observer

The calculation for ideal current magnitude (4.9) requires the value of the load resistor to be known. This value cannot be directly measured however, and is not assumed to remain at the same value at all times. Therefore, an observer must be constructed which will provide an accurate estimation of this load resistance. A sliding mode observer [7] can be designed to perform this task. In this study the load resistance is observed using a 2-SMC super twisting observer due to its accuracy and attenuated effects from chattering [7, 6]. As mentioned in Chapter two, an observer/differentiator should produce a continuous signal since the parameter being observed is continuous. If the second order method of super twisting control is not used, then a low pass filter would have to be used in order to produce this result. This application would require a higher order low pass filter. The super twisting control provides the continuous result with much higher accuracy and very few effects from chattering.

The load resistance only appears in the DC voltage equation of (2.2). This equation will be used to define the observer dynamics [3]:

$$\frac{d\widehat{U}_0}{dt} = -\frac{\widehat{U}_0}{R_0 C} + \frac{1}{2C}(i_1 U_1 + i_2 U_2 + i_3 U_3) + v_R, \tag{4.16}$$

where R_0 is a nominal resistance chosen by approximation, $\widehat{U}_0$ is the observer state and v_R is the injection term [7]. The observation error is then defined as follows:

$$e_u = U_0 - \widehat{U}_0. \tag{4.17}$$

The error in (4.17) will go to zero in finite time and has the following dynamics:

$$\dot{e}_u = -\left(\frac{U_0}{RC} - \frac{\widehat{U}_0}{R_0 C}\right) - v_R = \Psi - v_R \,. \tag{4.18}$$

Ψ is bounded since the DC voltage, the resistance, and the capacitance are bounded. Using the super twisting algorithm [6], the injection term can be defined as follows:

$$\begin{aligned} &v_R = v_{Req} = -\alpha|e|^{1/2}\mathrm{sign}(e) - \dot{\xi} \\ &\xi = \beta\mathrm{sign}(e) \\ &\alpha = \tfrac{1}{2}\sqrt{M} \\ &\beta = 1.1M. \end{aligned} \tag{4.19}$$

M is an arbitrary constant which is tuned for optimal performance. Using this injection term, the first derivative of the error will also go to zero and therefore (4.18) can be set to zero. Solving (4.18) equal to zero for v_{Req} yields the following result:

$$v_{Req} = \frac{\widehat{U}_0}{R_0 C} - \frac{U_0}{\hat{R}C}. \tag{4.20}$$

The estimated load resistance value can be calculated from (4.20) as follows [3]:

$$\hat{R} = -\frac{C v_{Req}}{\widehat{U}_0} + R_0. \tag{4.21}$$

4.5 Complete Mathematical Model of the Control Algorithm

A summarized version of the overall control system for power factor correction in three phase AC-DC boost converters can be created by combining (4.9), (4.10), (4.13) and (4.15) as follows:

$$U_j = \text{sign}(\sigma_j), j = 1, 2, 3$$

$$\begin{cases} \sigma_k = i_k - i_{dk}, k = 1, 2 \\ \sigma_3 = \int_0^t (U_1 + U_2 + U_3)\, dt \end{cases} \tag{4.22}$$

$$i_{dk} = I_{dk} \sin(\omega t + \theta_k)$$

$$I_{dk} = \frac{E_k}{2\hat{R}} - \sqrt{\frac{{E_k}^2}{4\hat{R}} - \frac{2}{3}\frac{{V_d}^2}{r\hat{R}}}.$$

The observer dynamics which track the load resistance can be summarized using (4.16), (4.17),(4.19), and (4.21) as follows:

$$\frac{d\hat{U}_0}{dt} = -\frac{\hat{U}_0}{R_0 C} + \frac{1}{2C}(i_1 U_1 + i_2 U_2 + i_3 U_3) + v_R$$

$$v_R = v_{Req} = -\alpha |e|^{1/2} \text{sign}(e) - \xi$$

$$\dot{\xi} = \beta \text{sign}(e)$$

$$\alpha = \tfrac{1}{2}\sqrt{M} \tag{4.23}$$

$$\beta = 1.1M$$

$$e_u = U_0 - \hat{U}_0$$

$$\hat{R} = -\frac{C v_{Req}}{\hat{U}_0} + R_0.$$

4.6 Summary

The control algorithm for correcting the power factor and producing a steady DC voltage level was defined using classical sliding mode controls [5]. The sliding mode

controller will work by the method of output tracking of the phase currents. An ideal current is generated by using known values of frequency and phase combined with a magnitude value generated by calculations formulated from the DC input and output power formulas. Only two of the phase currents have to be tracked, the third can be derived by equation (4.14). Other methods of sliding mode control can be applied to the three phase AC-DC boost converter and were researched in [4]. The higher order sliding mode controls have better accuracy but, since the resulting control signal will be continuous, they require the control signal to be pulse width modified. The other nonlinear methods researched in [4] did not provide the same level of accuracy as the sliding mode control algorithm

The resistance is estimated using a 2-SMC super twisting observer/differentiator algorithm [7, 6]. This estimation is done by estimating the value of the DC output voltage using the system equation that defines its behavior with a nominal value for the load resistance. The error between this estimated resistance and the actual measured value serves as the sliding variable in the estimator. The injection term is calculated using a super twisting formula [6]. Therefore, no low pass filtering is necessary and the result will be smoother and more accurate.

CHAPTER 5

DSP SIMULATION ENVIRONMENT DEVELOPMENT AND COMPOSITION

A processor in the loop setup requires that the embedded processor, in this case a Texas Instruments DSP, be integrated with a computer. The computer uses an Integrated Development Environment (IDE) between Simulink and Code Composer Studio (CCS), a Texas Instruments product. The simulation is built in Simulink, the code is generated though the IDE and uploaded to the device by CCS. This chapter will describe the setup of the DSP implementation of the controls described in previous chapters.

5.1 Overview of Co-Simulation Preparation

Figure 5.1 A computer and a DSP in a co-simulation loop.

Designing and running a processor in the loop co-simulation is a step by step process. This process is described in detail in this chapter. A basic overview of the individual steps follows:

Preliminary Setup: There are some steps which must be completed in order to run a co-simulation which must be done only once. First, an application appropriate DSP must be selected. This DSP must be compatible with Simulink Coder. Second, MATLAB with Simulink must be installed on the computer including the toolboxes SimPowerSystems and Simulink Coder. Finally, the supporting software for the DSP the drivers for the device must be installed. For TI processors this software is Code Composer Studio.

1. *Hardware setup:* Power the DSP and connect it to the computer using either JTAG or TCP/IP.
2. *Simulink setup:* Create the plant model for the three phase AC-DC boost converter in Simulink using SimPowerSystems [18] and configure the model to run at the desired plant model evaluation rate. Next, create the control model in Simulink using the standard block set or optimization blocks from the target support block set corresponding to the embedded device on which the code will be implemented [19]. This control model must be completely enclosed in its own subsystem. Interface the control model to the plant model based on the necessary word size of the processor, which may be less than the computer, using data type conversion blocks. Insert the target preferences block in to the control model subsystem, configuring the CPU speed, peripheral devices, and memory map parameters to match the DSP [17]. Finally, add the necessary analysis blocks to the Simulink model, including power factor calculators and workspace storage variables.

3. *Supporting software configuration:* Open Code Composer Studio (assuming necessary drivers have already been loaded) and connect to the DSP. Make the necessary configuration changes such as turning off the code security module.
4. *Co-simulation build process*: In Simulink, select the control subsystem and build using Real Time Workshop [19]. The code will be generated using the Simulink and CCS IDE. A project will be generated in CCS and a PIL control block will be generated in Simulink. Replace the control block in the main Simulink model with the PIL block generated by the build process. Set the desired simulation configuration including integration type, integration step size, and run time. PIL works with all three standard simulation modes. Normal mode was selected for this implementation.
5. *Running the Co-simulation*: Start the co-simulation by pressing the start simulation button. The project in CCS will be built and encoded to the processor. The co-simulation will begin after. All the data returned by the processor will be stored in Simulink variables for processing.

The presented steps for building, configuring, and running a co-simulation in the MATLAB/Simulink and CCS IDE are described in more detail throughout the following sections of this chapter.

5.2 DSP Selection

The preliminary setup process begins with the selection of a DSP. There are hundreds of DSP devices currently manufactured. The processor used in this

co-simulation was carefully selected based on several factors. The first factor is processing and bus speed. This speed should be fairly high since the control needs to be evaluated at a fairly high rate and the control signal produced should switch at the rate of at least 20 kHz. The next major factor is cost. These embedded devices are very sensitive and can possibly become unusable if connected incorrectly or exposed to static. Therefore, it is optimal to purchase a lower cost DSP. Another factor to take into consideration is peripheral devices. In this co-simulation the device is connected to a computer. Therefore, some form of connection must be available between the two devices. PIL simulations in Simulink use JTAG emulators by default. If the device does not have a built in JTAG emulator, the emulator must be purchased separately or the entire setup will have to be configured manually to use a different type of connection. Optimally, the DSP should be useable in experiments using the actual hardware version of the plant model. It may be convenient if the peripheral devices fit this application as well. Following is the review of three devices, including the device finally selected for use in this PIL implementation. Also, it is imperative that the device must be supported in Simulink Coder.

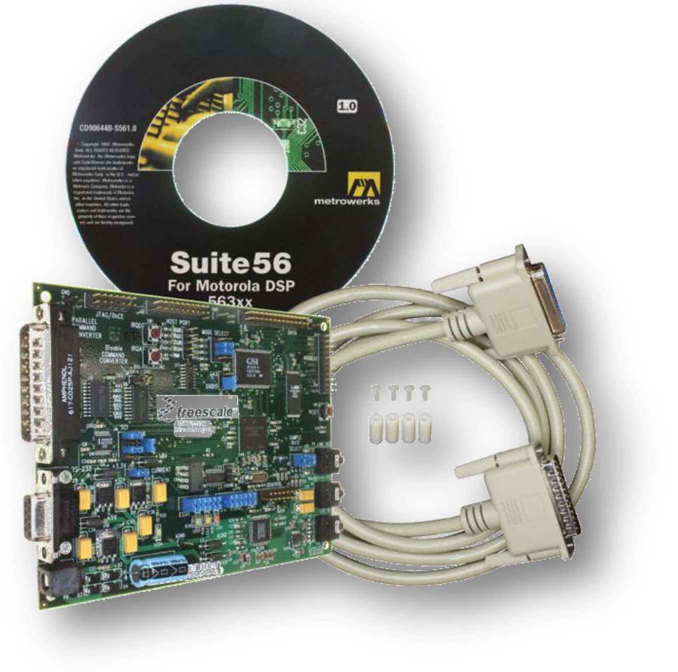


Figure 5.2 An MPC555 DSP.

The MPC555 manufactured by Freescale Semiconductor is a very widely used embedded device [15]. The device does not have a built in JTAG emulator but it does have CAN capability. Due to its availability and common use in control applications it is supported in Simulink. The device is also very inexpensive, with the price ranging around 100 dollars. The 32 bit processor included with this board is manufactured by Motorola. It is fairly fast, with a 40MHz max bus speed. However, this device is not well suited for this application. A faster processor speed is desirable and the inclusion of peripheral devices would be beneficial.

Figure 5.3 ezDSP F28335 by Spectrum Digital.

Texas Instruments manufactures a line of processors called the TMS320 family [16] which are intended for use in power electronics and power conversion applications such as the one studied in this thesis. The company Spectrum Digital manufactures several development boards with these processors. The processor chosen for this application is the F28335. It is also a 32 bit unit but the top processing speed is much faster at 150 MHz. It has built in PWM units, an RS232 connector and a CAN interface. The boards built by spectrum digital also include built in JTAG emulators which allow the device to be connected to the computer by USB. The processor is ideal for this application but the boards come with several options with a wide price range. The most basic and inexpensive (at around 300 dollars) board is the ezDSP F28335 [17] without a socketed processor. The addition of a socketed processor increases the price to around

500 dollars. If damage occurs to the board it is almost invariably direct damage to the processor. Having a socketed processor allows the processor to be replaced without having to replace the entire board. The last board manufactured by Spectrum Digital using this processor includes extra options for motor control development such as a fuzzy logic engine, a fuzzy logic processor, and some extra simulation options. This board runs close to 1000 dollars. The added cost would be unnecessary for the purposes of this co-simulation since none of those added peripherals would apply to this study. Also, since the co-simulation is implemented through the JTAG emulator, there is very little threat posed to the processor. This means that for this application it is also unnecessary to purchase a socketed processor. However, if the board were to be used in an actual test environment this added option is the optimal choice and therefore this version of the board was purchased for the purpose future development in hardware testing.

5.3 Physical Structure of the Experimental Setup

The required components for a PIL simulation include a computer with MATLAB and Simulink installed as well as the required software for coding the embedded processor. Installing this software and the DSP drivers completes the preliminary setup. As stated previously, the DSP chosen for the implementation in this thesis is the eZdsp F28335, a DSP board built by Spectrum Digital. The processor on this board is a Texas Instruments TMS320F28335 digital signal controller. This DSP is compatible with CCS. More information on this device can be found in [10]. When choosing a DSP for a PIL application, the device must be compatible with Simulink Real Time Workshop as well

as compatible coding software. A complete list of compatible devices is available on MathWorks' website.

Figure 5.4 Physical interface between the DSP and the computer.

Figure 5.4 is a picture of the DSP device connected to the computer to run a PIL simulation. Setting up this hardware is the first major step in creating the co-simulation. A properly grounded antistatic mat and wristband are used for handling the device due to the exposed soldering and pins on the board that are directly connected to the processor. The DSP chosen for this research has a built in JTAG connector which allows the device to be connected to the computer via USB. This type of communication can be slow, but it is effective for passing data in simulation time. It is also possible with some devices to manually set up a TCP/IP type connection which will run faster. This DSP requires an external power source. In order to correctly boot the device, the power must be applied to the device approximately ten seconds or more before the USB cord is connected to the

computer. Once the DSP is connected, the next step is to build the plant and control models in Simulink.

5.4 Simulink Environment and Code Composer Studio IDE

The software environment on the computer consists of two main components. MathWorks' MATLAB with Simulink is the program in which the system design will take place. This includes the plant model, the control, and any other component in the simulation required for control design interface or analysis of the data. This is also where the data returned by the device will be stored for analysis. The other program component is Code Composer Studio. All code generation and passing of data to and from the DSP is handled by this program.

5.4.1 SimPowerSystems and Simulink Coder Setup

Building the model in Simulink is the second step of the co-simulation process. Two non-default Simulink toolboxes were used to implement the designs described in the previous chapters. The plant model was built using SimPowerSystems. This toolbox includes accurate models of common components in electrical power systems, including three phase AC-DC converters. All the parameters of this device can be customized manually, including resistor and inductance values, simulation precision, and device tolerances [11].

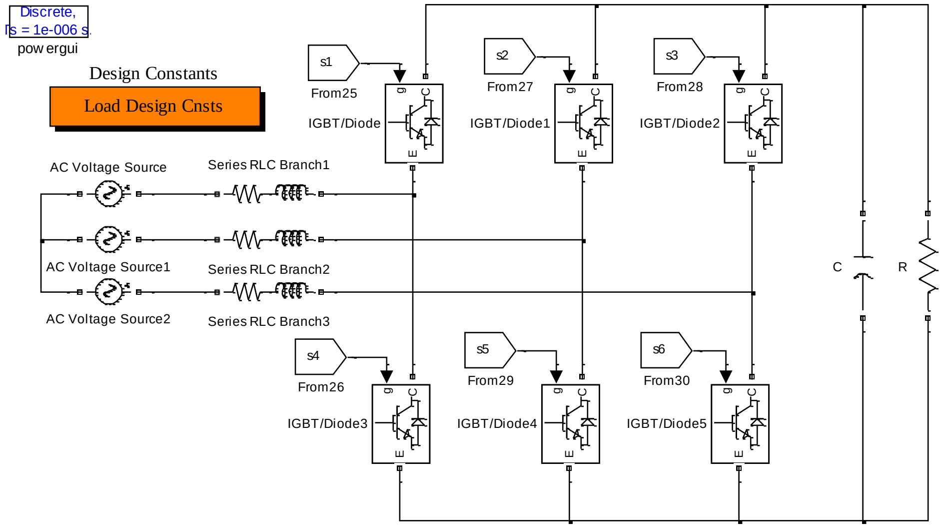


Figure 5.5 SimPowerSystems three phase AC-DC converter model.

If it is desirable to quickly change parameters and run more simulations, a database of program variables can be setup in the form of a MATLAB .m file and linked to a button in the Simulink GUI, as displayed in Figure 5.2.

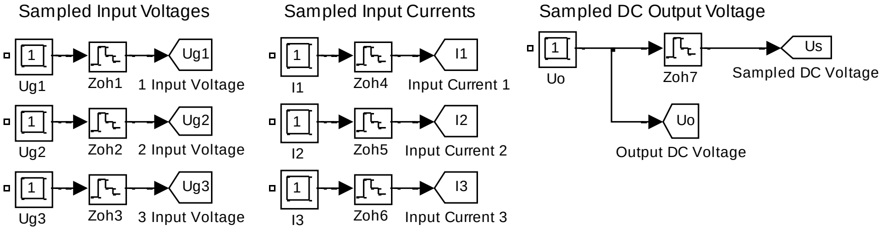


Figure 5.6 SimPowerSystems interface to other model simulation components.

Multimeter blocks are used in order to interface the data received from SimPowerSystems to the rest of the simulation, as in Figure 5.3. The sampling time on these blocks can be set individually. In this case they are set to simulation time, which is

the same as the plant model. The switch to control sampling time takes place at the interface to the controller.

The controller is implemented using the basic Simulink block set and Simulink Coder. Simulink Coder includes the target support packages for all the supported embedded and PC devices, Real Time Workshop, and the embedded coder. When starting to build a PIL simulation, the control model is first built from the default Simulink block set as it would be for a normal simulation, except that the entire controller must be contained within a single subsystem. If it is applicable, optimization blocks may be selected from the target support package for the embedded device to replace some of the basic math blocks [12].

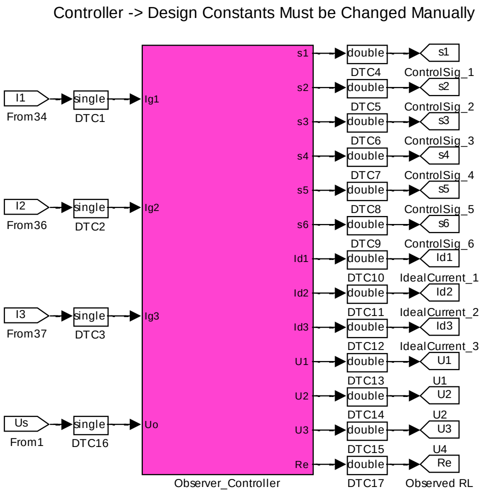


Figure 5.7 Control block with data type conversions at the interface.

If the word size of the embedded device is not the same as the computer, it is necessary to do data type conversion at all input and output interfaces to the device as shown in Figure 5.4. In the case of the simulation created for this research, the word size of the device is smaller than that of the computer, therefore a conversion to the

single data type was used at all inputs. The outputs required a data type conversion back to double. The incoming data is read as a double by the device, using the appropriate word size.

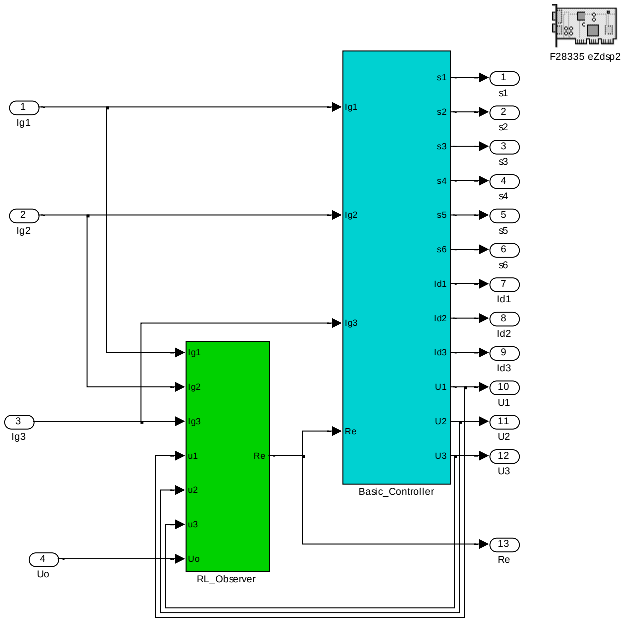


Figure 5.8 Under the mask of the control subsystem including target preferences block.

The next part of the Simulink modeling process is to add the target preferences block to the controller subsystem, as shown in Figure 5.5. This block will contain all the information about the DSP required to generate code for the device including: processor and internal clock speeds, a memory map, peripheral device settings, and the selection of which section of memory the program will be loaded into (i.e., RAM or flash). After this is in place, the configuration settings for Real Time Workshop must be properly adjusted for PIL settings, device recognition, and the CCS IDE.[1] For this particular configuration this means data import should be set for signal logging. The hardware implementation should be configured for the TI C2000 family of processors with no emulation hardware.

---

[1] Note that the version of CCS, device BIOS, CCS code generation tool, Simulink Coder, and MATLAB should all be in the latest available version or the PIL simulation may not yield any results.

The correct IDE system target file selection is idelink_ert.tfc using C as the programming language. The interface options should be configured for the TIC28x continuous time, non-inlined S-function and variable size signals should not be selected. SIL and PIL verification should be set to PIL and portable word sizes should be disabled. Finally, the embedded IDE configured so that the build format is Project and the build action is create processor in the loop project [19]. Any MATLAB simulation mode will work with PIL simulations, in this research normal simulation mode was used.

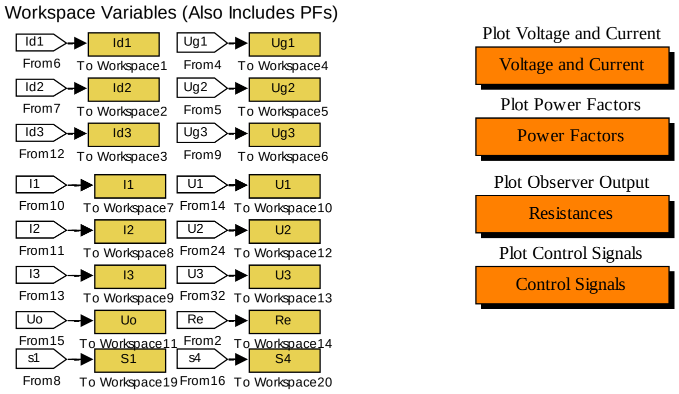


Figure 5.9 Workspace variables and plotting blocks.

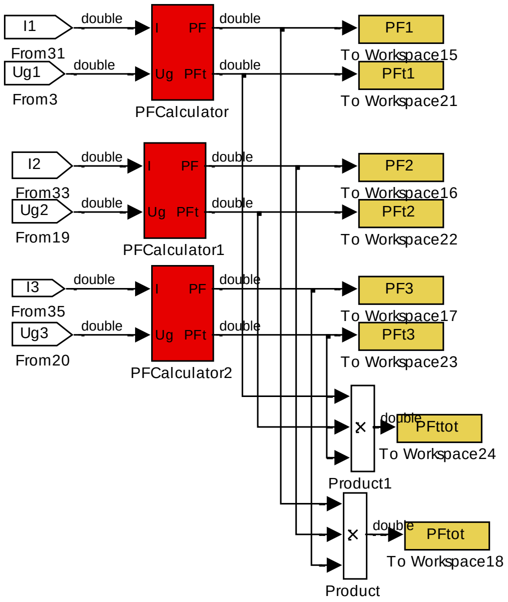
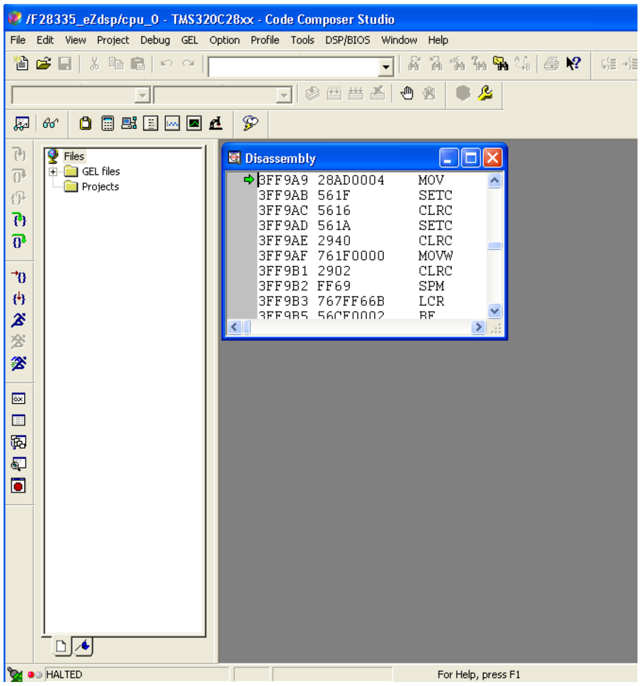


Figure 5.10 Power Factor Calculators.

There are other variables included in the Simulink GUI which are used for analysis of data. These include the blocks which carry the data received from SimPowerSystems and the DSP to the workspace, pictured in yellow in Figures 5.6 and 5.7. Writing plotting functions and mapping them to buttons is useful for simulations like this one where the same workspace variables will need to be plotted several times under varying system parameters. The orange blocks in Figure 5.6 represent these plotting functions. Figure 5.7 shows the power factor calculators, these subsystems contain the mathematical calculations for calculating the power factor.

5.4.2 Code Composer Studio Configuration

Code Composer Studio must also be configured to the particular device. This is the third major step of building the co-simulation. The appropriate drivers must be installed, with this device the drivers were included on a CD which came with the board but are also available online. The first time CCS is started the device must be selected from a list of available devices in the CCS setup menu. After the device settings are correct, the device may be connected. With the F28335, the device must first be connected to power then to the computer. CCS should not be started until the device is connected to the computer. Once CCS is opened the keyboard action ctrl-c will connect the device to CCS [17].



```
3FF9A9 28AD0004     MOV
3FF9AB 561F         SETC
3FF9AC 5616         CLRC
3FF9AD 561A         SETC
3FF9AE 2940         CLRC
3FF9AF 761F0000     MOVW
3FF9B1 2902         CLRC
3FF9B2 FF69         SPM
3FF9B3 767FF66B     LCR
3FF9B5 56CF0002     BF
```

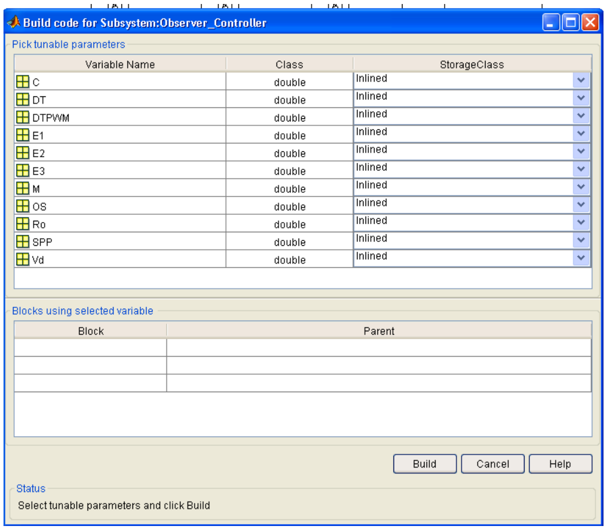
Figure 5.11 Code Composer Studio GUI after the device is connected.

If the device is connected properly, a disassembly should appear on the screen and GEL configurations will become available, as in Figure 5.8. The only necessary change in configuration within CCS is to turn off the code security module. This module is normally set to locked in order to prevent accidental overwrite of important sections of memory. If it is locked during the runtime of this simulation, it will prevent the control code from properly loading to the device.

5.5 Running a PIL Simulation

The fourth step in the process, after setting the configurations in both programs, is to generate the code for the control block. This is done by building selecting the control subsystem and using the build option using Real Time Workshop [19]. Real Time

Workshop was formerly its own Simulink package but is now included in Simulink Coder. A build window will appear which contains information about the inlined tunable parameters which will be included in the control model.



Variable Name	Class	StorageClass
C	double	Inlined
DT	double	Inlined
DTPWM	double	Inlined
E1	double	Inlined
E2	double	Inlined
E3	double	Inlined
M	double	Inlined
OS	double	Inlined
Ro	double	Inlined
SPP	double	Inlined
Vd	double	Inlined

Block	Parent

Select tunable parameters and click Build

Figure 5.12 Build window for control subsystem in Simulink.

Figure 5.12 shows this build window. There are several tunable parameters in this co-simulation. The only one not addressed previously is SPP, the sampling rate for the sine function used in the current generation profile. The code is generated using the Simulink and CCS IDE.

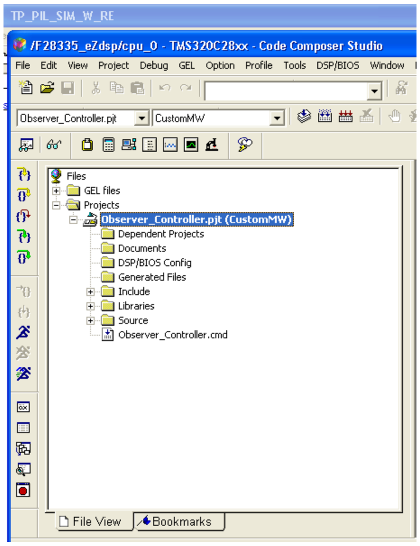
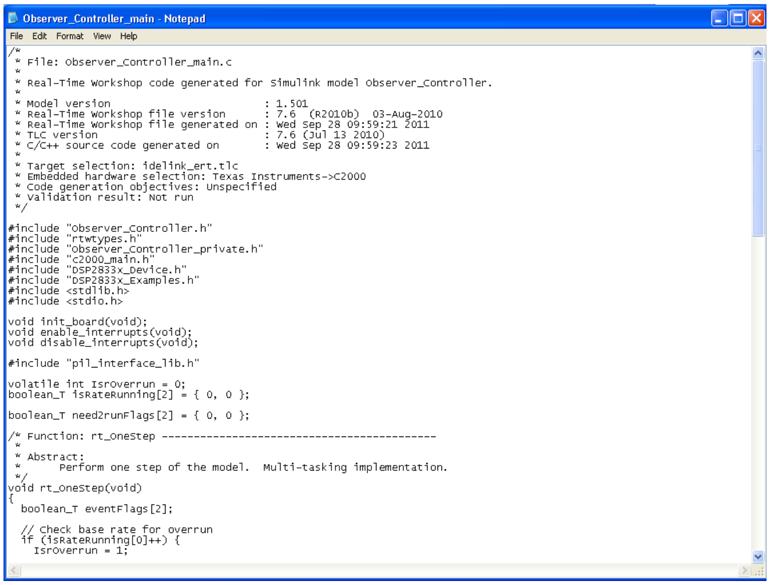


Figure 5.13 Controller project displayed in CCS File View pane.



```
/*
 * File: Observer_Controller_main.c
 *
 * Real-Time Workshop code generated for Simulink model Observer_Controller.
 *
 * Model version                        : 1.501
 * Real-Time Workshop file version      : 7.6  (R2010b)  03-Aug-2010
 * Real-Time Workshop file generated on : Wed Sep 28 09:59:21 2011
 * TLC version                          : 7.6 (Jul 13 2010)
 * C/C++ source code generated on       : Wed Sep 28 09:59:23 2011
 *
 * Target selection: idelink_ert.tlc
 * Embedded hardware selection: Texas Instruments->C2000
 * Code generation objectives: Unspecified
 * Validation result: Not run
 */

#include "Observer_Controller.h"
#include "rtwtypes.h"
#include "Observer_Controller_private.h"
#include "c2000_main.h"
#include "DSP2833x_Device.h"
#include "DSP2833x_Examples.h"
#include <stdlib.h>
#include <stdio.h>

void init_board(void);
void enable_interrupts(void);
void disable_interrupts(void);

#include "pil_interface_lib.h"

volatile int IsrOverrun = 0;
boolean_T isRateRunning[2] = { 0, 0 };

boolean_T need2runFlags[2] = { 0, 0 };

/* Function: rt_OneStep -------------------------------------------
 *
 * Abstract:
 *      Perform one step of the model.  Multi-tasking implementation.
 */
void rt_OneStep(void)
{
  boolean_T eventFlags[2];

  // Check base rate for overrun
  if (isRateRunning[0]++) {
    IsrOverrun = 1;
```

Figure 5.14 Typical C code generated in CCS.

The unnecessary information about the model such as subsystem definitions are removed by Simulink, the data is then passed into CCS where the C code is generated, as shown in 5.14, and formed into a project as shown in Figure 5.13. This project will later be compiled and encoded to the device by CCS during runtime.

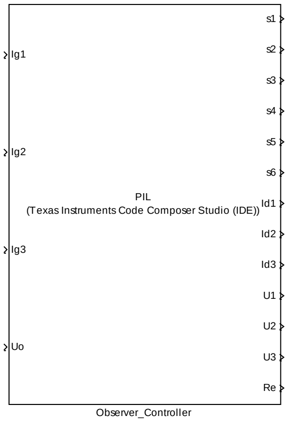


Figure 5.15 PIL block generated by building the control block in Real Time Workshop.

This code will be linked back to Simulink, where a PIL block that exactly resembles the controller will be built in a separate simulation window, as shown in Figure 5.15. This block will replace the control block in the simulation environment. Also, a dialog box will also appear with a list of links to all the generated code files for quick referencing.

Once the PIL block is in place in the simulation, the simulation process is started as with normal simulations. This is the fifth and final step of the co-simulation process. The progress of the simulation can be followed in the command window. The PIL project is first built and loaded to the device within CCS. During runtime, the data is passed one simulation step at a time to the embedded device where the appropriate calculations are made. The data is then returned to the computer and passed back into the Simulink simulation and/or stored into a workspace variable. This process is repeated until the specified simulation end time is reached.

5.6 Summary

The first step in building a PIL simulation in the Simulink/CCS IDE is to build the desired model in Simulink, also building any peripheral necessities such as the power factor calculators and the data storage blocks. The control block must be appropriately interfaced to the plant model, with data type conversions and the target preferences block, and the Real Time Workshop configurations must be adjusted appropriately. The driver for the device must be loaded and the GEL must be configured appropriately in CCS. To start the PIL simulation, build the control block using Real Time Workshop and replace the current control block with the PIL block that is generated by the build process. The simulation will then be run as a normal configuration would be.

CHAPTER 6

DSP IMPLEMENTATION AND CO-SIMULATION STUDY RESULTS

This chapter contains the results of the processor in the loop system described in chapter five under varying conditions. The first set of results is for what is called the nominal mode in which the system is tested with preselected base values of all system variables. This will serve as the basis for comparison in the later studies. The other three tests are intended to investigate the performance of the system by varying the phase inductance, phase voltage frequency, and load resistance respectively. The results in this chapter prove that the control algorithms work properly in simulation time when encoded into the DSP.

6.1 Nominal Mode

Table 6.1: Nominal Mode Parameters

Parameter	**Value**	**Description**
f_p	$1e^6$	Plant model simulation frequency, Hz
f_e	$1e^5$	Control evaluation frequency, Hz
f_s	$2e^4$	Sign function (or PWM) frequency, Hz
R_L	40	Load Resistance, Ω
R	0.2	Parasitic phase resistance, Ω
L	2	Phase inductance, mH
C	100	Output Capacitance, µF
E_i	150	Input AC voltage magnitudes, V
f_v	60	Phase voltage frequency, Hz
V_d	650	Desired DC output voltage, V

Table 6.1 contains the parameter values for the nominal mode DSP simulation. The plant simulation frequency was chosen to be very high in order to evaluate the plant at near-analog conditions. The control and sign function evaluation frequencies were chosen to represent realistic processing and PWM rates for the DSP. All other parameters were set to numbers that characterize standard applications of a three phase AC-DC converter.

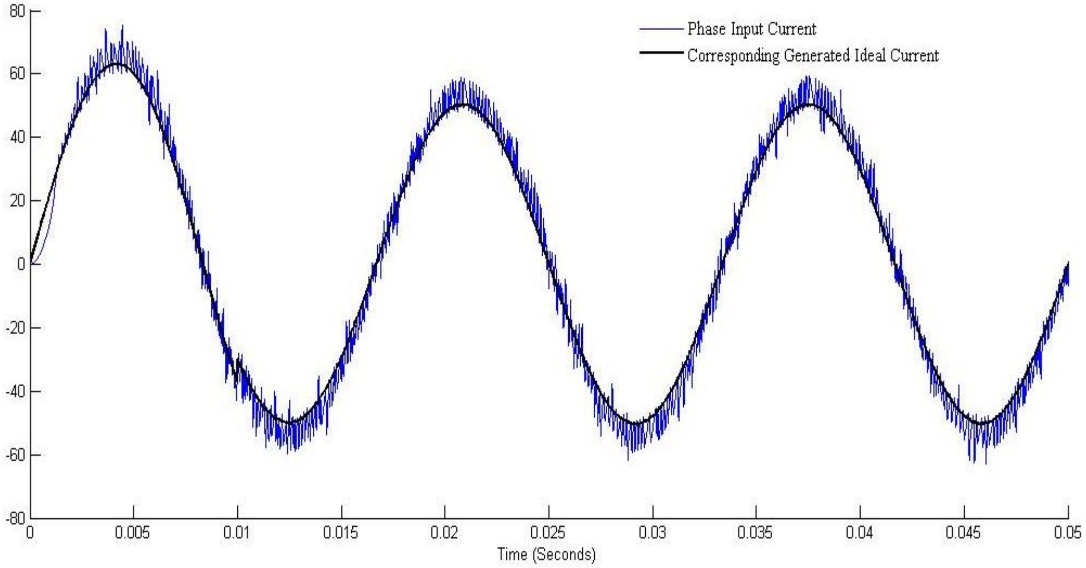


Figure 6.1 Phase current and ideal phase current for nominal mode.

Figure 6.1 shows the phase current from the first phase and the ideal current generated by the control. The phase current tracks the ideal current closely in the nominal mode co-simulation, however the phase current contains harmonics other than the main harmonic. This is due to the switching nature of the controller which is caused by the sign function. This can be corrected by using a higher control evaluation rate or by increasing the phase inductance.

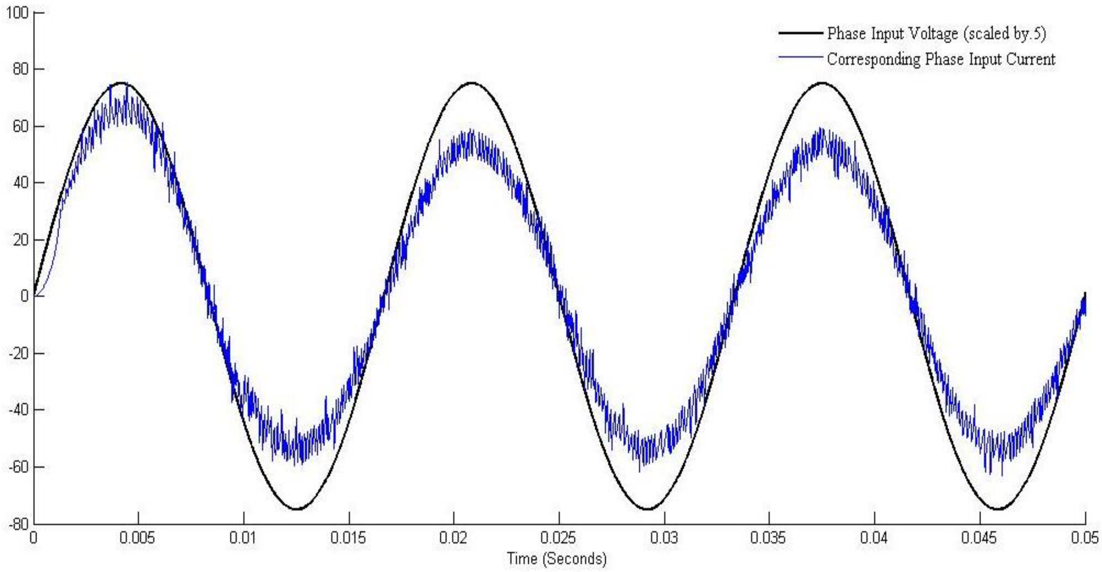


Figure 6.2 Phase voltage and current for nominal mode.

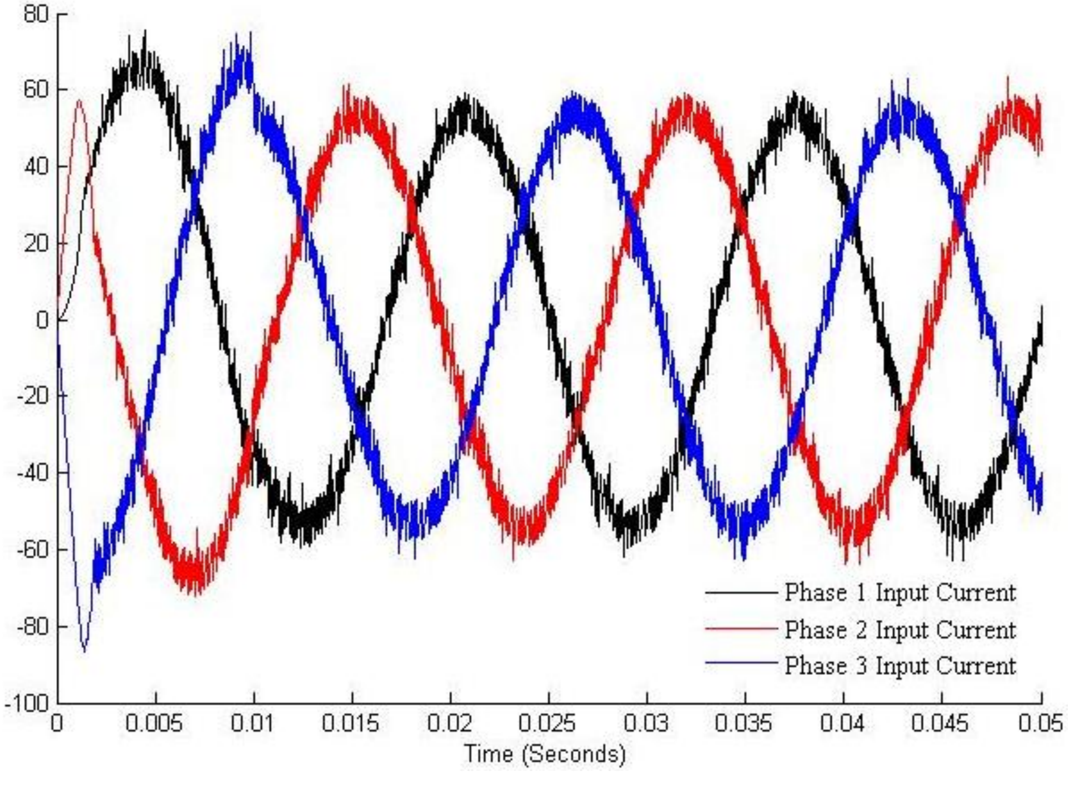


Figure 6.3 Phase currents in nominal mode.

Figure 6.2 shows the phase one voltage and current. In the picture it is clear to see that they are correctly in phase, which will result in a very high phase component in the power factor. Figure 6.3 shows that the three phases have a 120° phase difference from each other. Since the control has no effect on the input voltages, it can be determined that each phase current will align correctly with its corresponding phase voltage and each phase will have the same power factor. The only problem is caused by the chattering effect in each phase. The high harmonics that exist in each phase will cause distortion power which slightly increases the current to be sent to the load, increasing the output power to higher than desired levels [13].

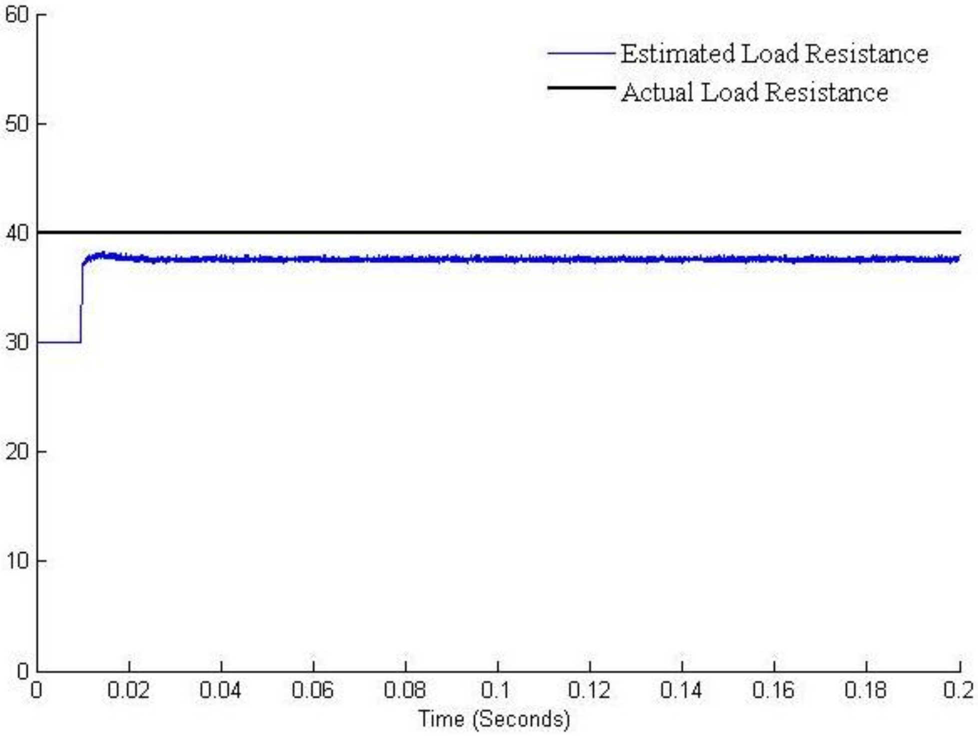


Figure 6.4 Load resistance observer in nominal mode.

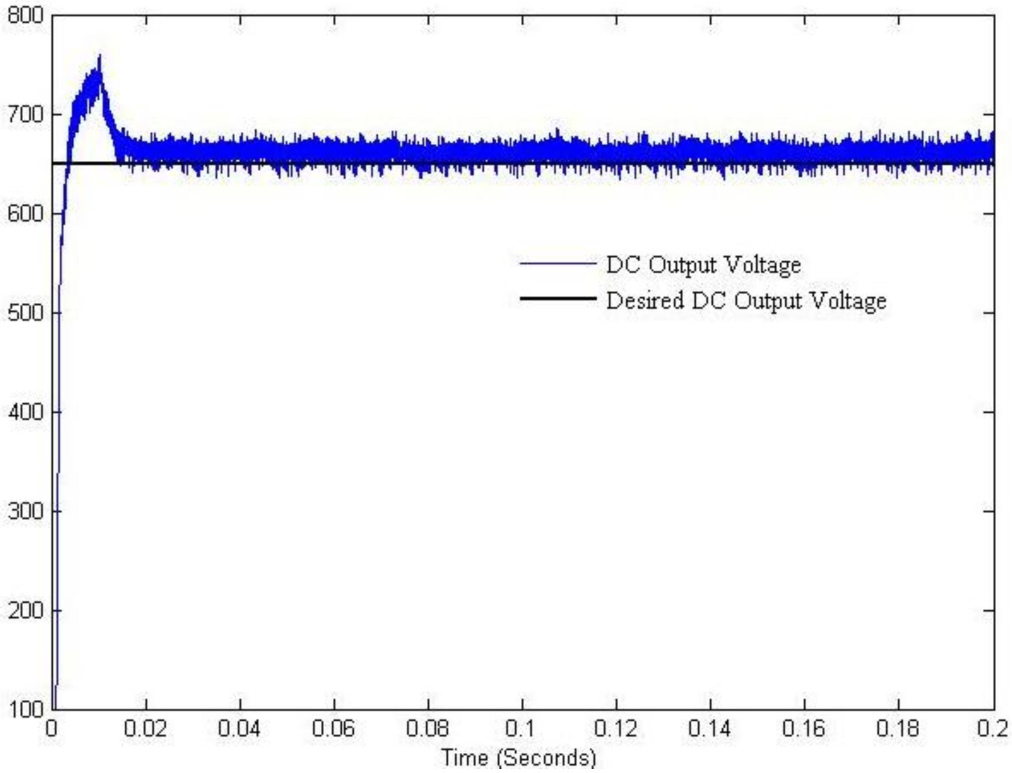


Figure 6.5 DC output voltage tracking in nominal mode.

Figure 6.4 shows the performance of the load resistance observer. The observer is approximately one ohm less than the actual load resistance of forty ohms. This is due to the effect of chattering on the input current causing excessive power at the output. This also has an effect the DC output voltage, shown in Figure 6.5. The excessive power at the output causes the DC voltage to be higher than the desired voltage. The load resistance observer uses the DC output voltage to estimate the resistance; therefore, if the output voltage is slightly higher than expected, the resistance will appear slightly lower. There is not as much apparent rippling in the load resistance observer because the super twisting control was used. Using a classical sliding mode observer and a standard low pass filter would have resulted in rippling in the observer similar to that in the controller.

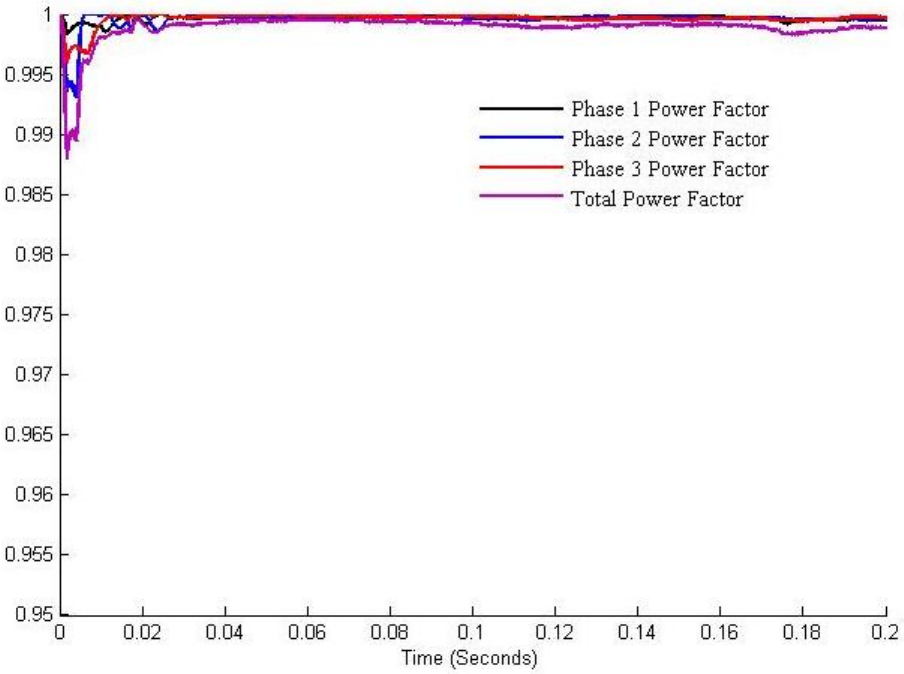


Figure 6.6 Power factors with only phase term in nominal mode.

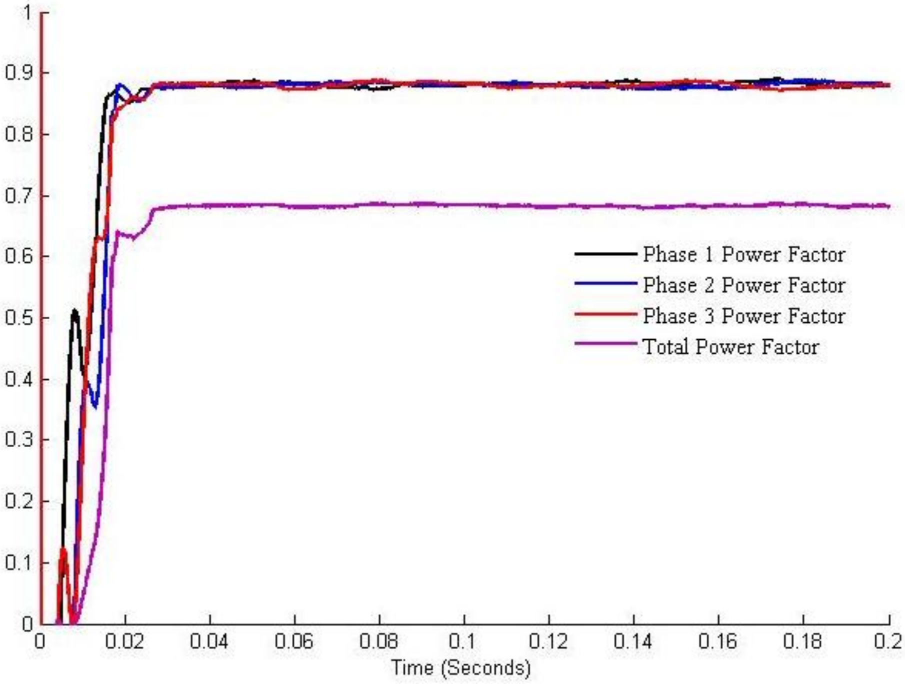


Figure 6.7 Power factor with harmonic distortion term in nominal mode.

The power factor can be calculated two different ways, as discussed in Chapter two. Using the traditional method of only taking into account the phase, the power factor for the sliding mode control is extremely close to one, as shown in Figure 6.6. When the harmonic distortion is also taken into account, the power factor appears to be lower but still very close to nine for each phase. If the higher harmonics were eliminated, the power factor could be increased closer to the phase-only power factor levels.

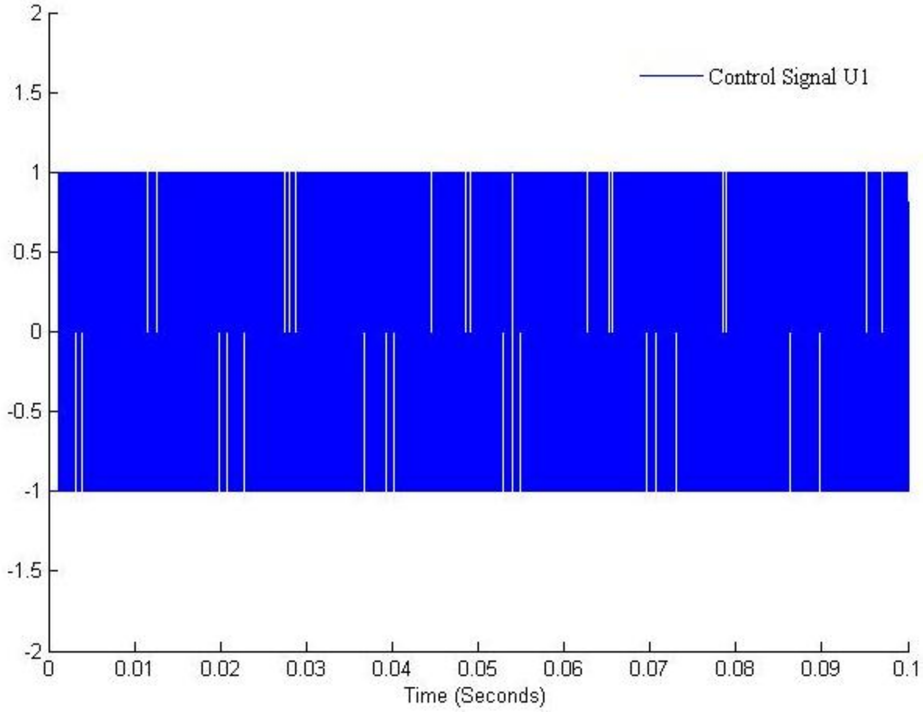


Figure 6.8 One control signal in nominal mode

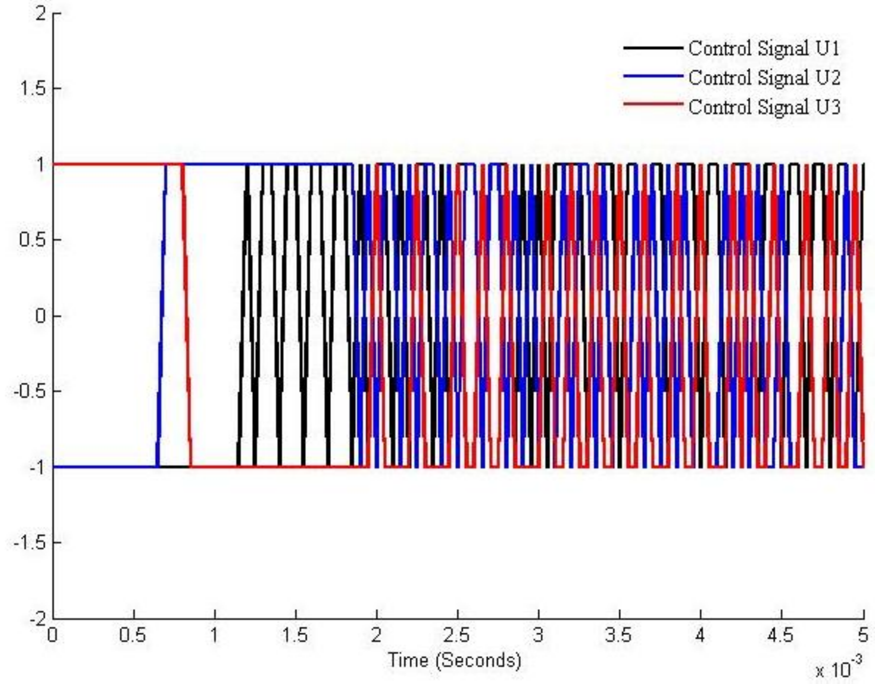


Figure 6.9 Control signals in nominal mode.

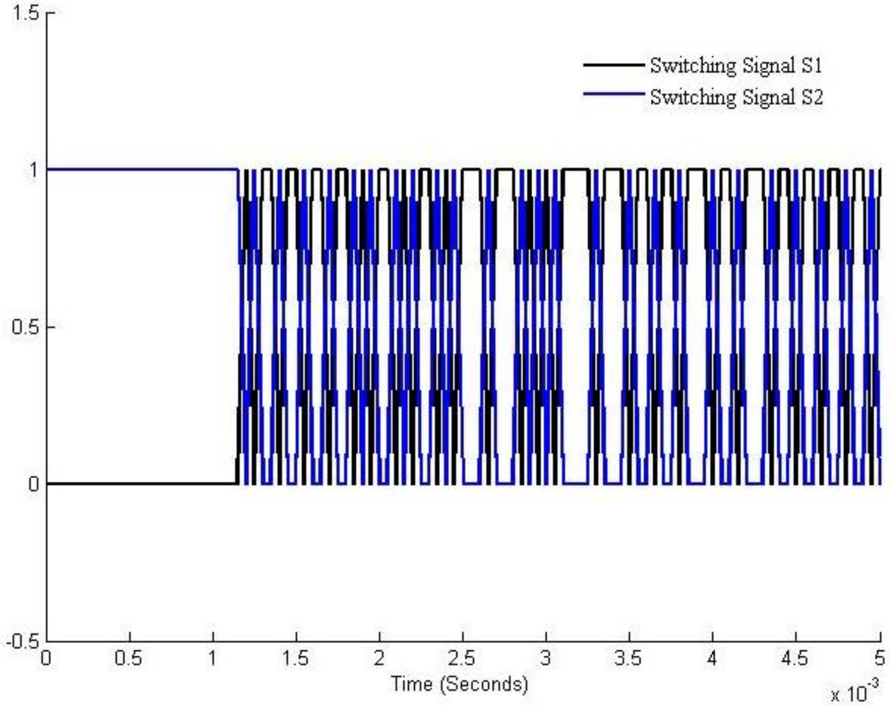


Figure 6.10 Corresponding switching signals in nominal mode.

Figures 6.9 and 6.10 demonstrate the control signals generated by the control before they are converted to switching signals. Figure 6.8 shows the control U_1, which is the control generated for the first phase. Figure 6.9 shows all three control signals. Looking at all three signals together is important in application because the controls cannot all share the same value at any given time or the control will fail. Figure 6.10 shows the switching signals for the first phase which control the corresponding switches S1 and S4. It is also important to be sure that these signals never share the same on or off value before the control is tested on the actual three phase converter. An occurrence of corresponding phase switches being on or off at the same time could potentially result in dangerous conditions for the operator and irreparable damage to the three phase converter.

6.2 Effect of Varying Source Inductance

The only noticeable problem in the nominal mode tests is the chattering in the phase input current. This causes problems when trying to reach a desired output voltage level and has a negative impact on the power factor due to added phase harmonics. The chattering in the controller also causes error in the load resistance observer. It is not very practical to assume the control evaluation rate can be continuously increased in order to accommodate the control since the control runs on a digital device. Therefore, other ways to reduce this ripple must be investigated. It can be shown that increasing the phase inductance will reduce the rippling in the phase current and therefore positively affects

overall control performance. In this test the phase inductance was increased to 5 mH then to 10 mH to show these effects.

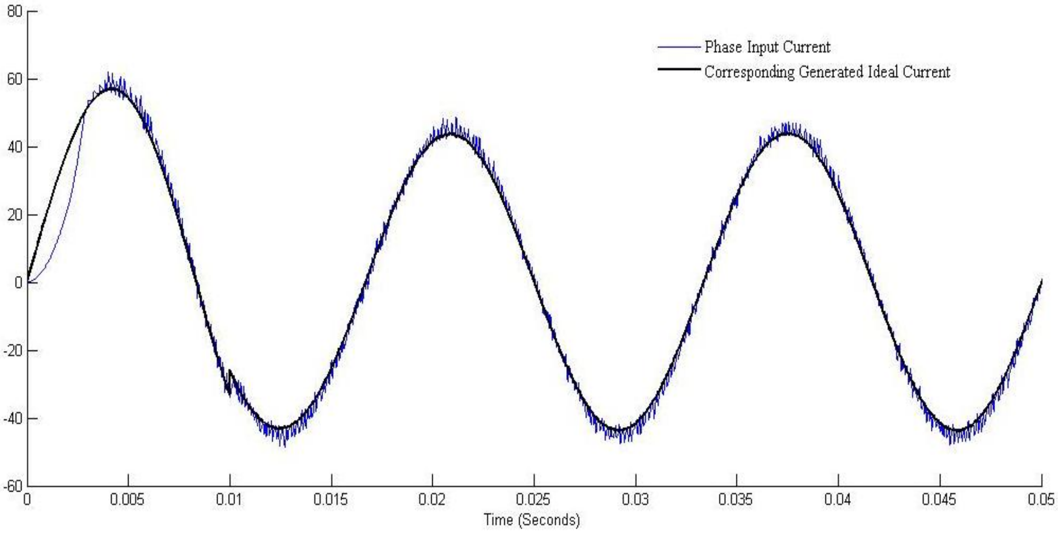


Figure 6.11 Phase current and ideal current with 5 mH phase inductance.

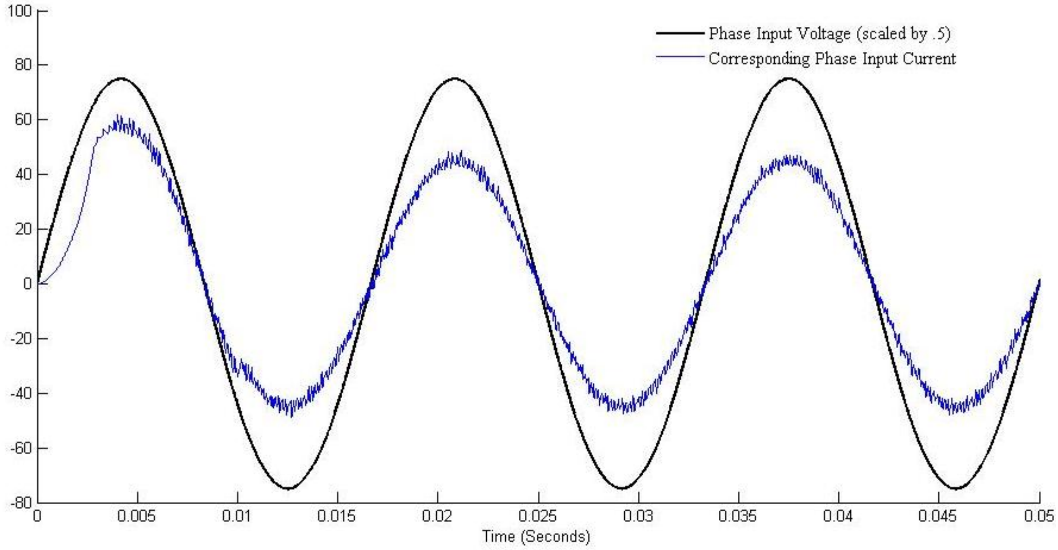


Figure 6.12 Phase voltage and current with 5 mH phase inductance.

Figure 6.11 shows the phase current for the first phase and the control-generated ideal current that corresponds to the first phase. With the phase inductance increased to 5 mH, the undesired harmonics are significantly reduced. Figure 6.12 shows the phase voltage and current for the first phase. This shows that increasing the inductance has no effect on the overall phase of the current, which would have a negative impact on the power factor.

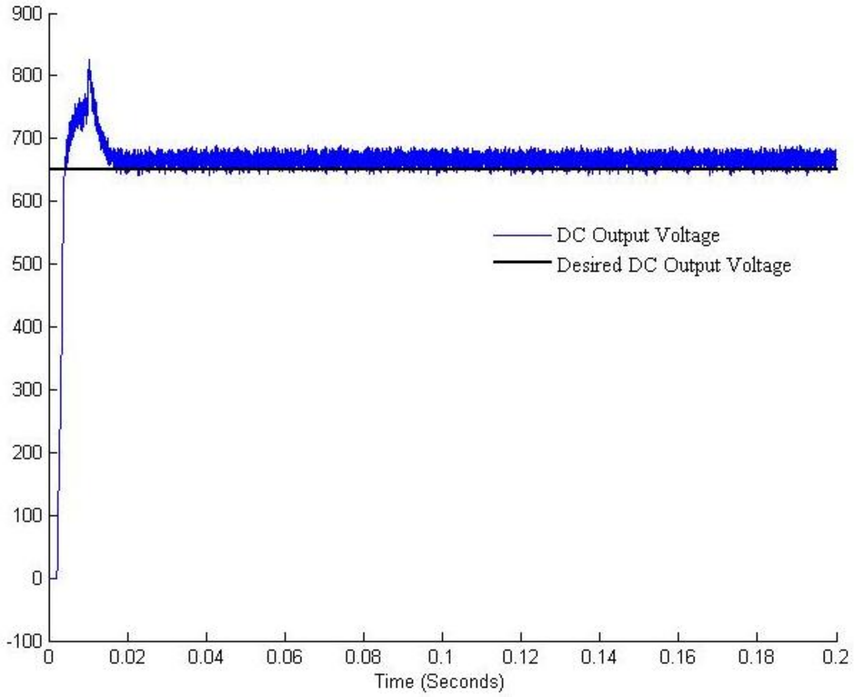


Figure 6.13 DC output voltage with 5 mH phase inductance.

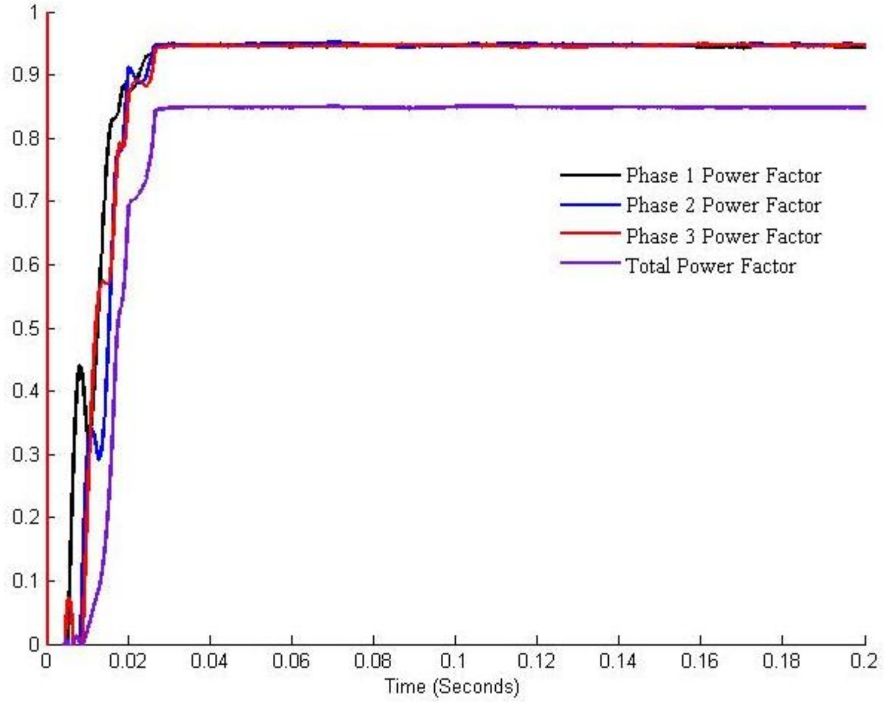


Figure 6.14 Power factors with harmonic distortion term and 5 mH phase inductance.

The DC output voltage, shown in 6.13, seems to have slightly less rippling than the DC output voltage in the nominal mode test, shown in 6.5. The clear effect of increasing the input inductance is shown in Figure 6.14. The power factor calculated with the harmonic distortion term is significantly higher. Each phase has a power factor of well over 0.9.

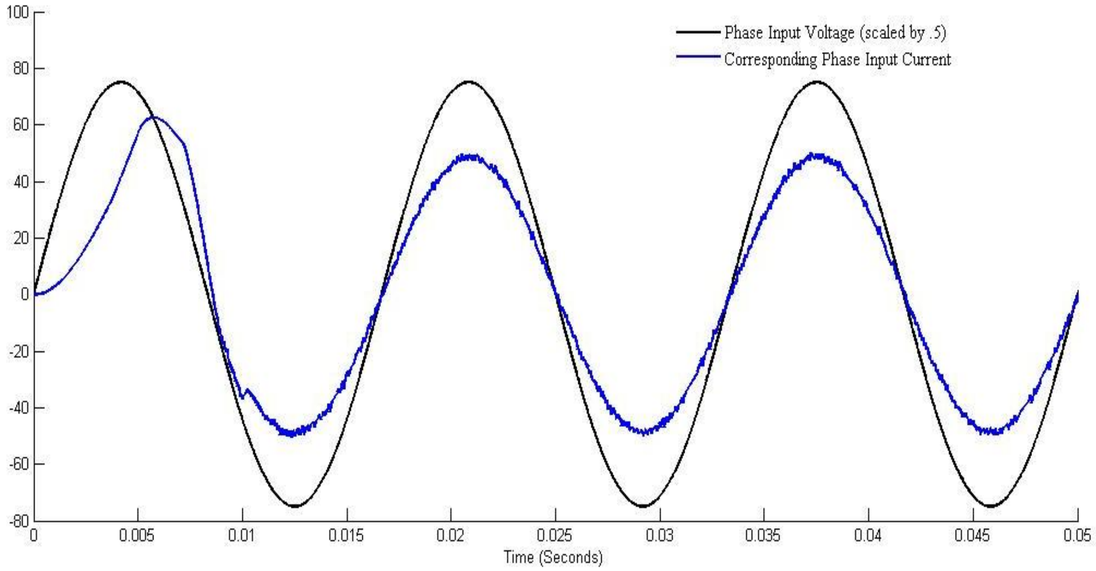


Figure 6.15 Phase voltage and current with 10 mH phase inductance.

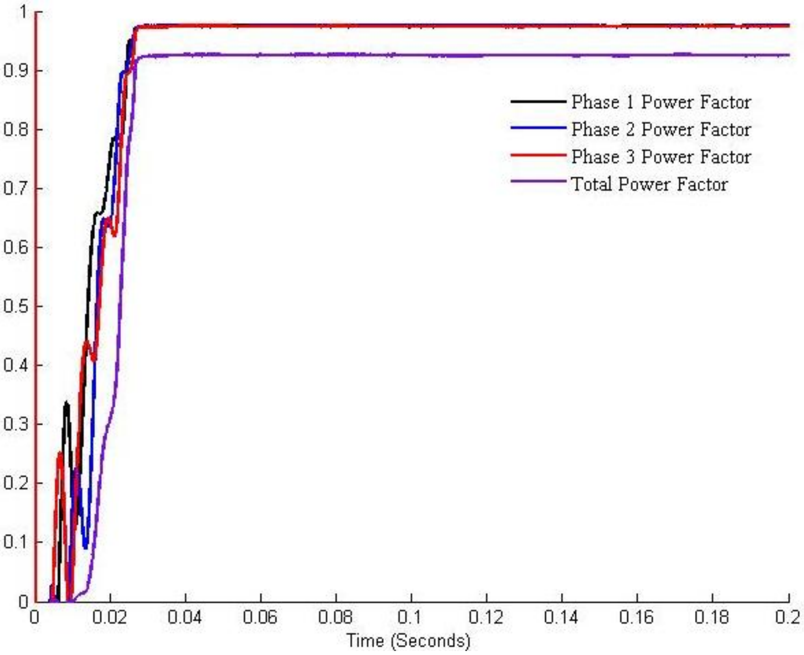


Figure 6.16 Power factors with harmonic distortion term and 10 mH phase inductance.

By increasing the inductance again to 10 mH the rippling can be reduced even farther, as shown in Figure 6.15. There is still no effect on the phase of the current

overall, but the reaching time is slightly longer. As shown in Figure 6.16, the power factors, including the overall power factor, are all above 0.9 even with the harmonic distortion term included in the calculation. Increasing the phase inductance to 5 mH could possibly be a reasonable act in application. However, increasing the inductance to 10 is unlikely a good idea. The problem encountered with actual inductors is that they can become significantly larger physically at higher inductance values. Depending on the application, the extra hardware volume and weight may be very unwelcomed. For the rest of the control implementations in this chapter, the phase inductance will be set to 5 mH so that the effects of varying other parameters will be more obvious.

6.3 Effect on System Performance from Varying Phase Voltage Frequency

The operator of the AC-DC converter may need to use a higher phase voltage frequency in certain applications. This study is very significant in certain implementations of the three phase converter. The study of sliding mode controls for the purpose of power factor correction in [3, 4] was originally intended for use in an aircraft. The application was to be used to drive a component of the engine. The converter operated at high frequencies which vary depending on the changing rotational speed of the component. The implementations in this section test the upper bound on frequency by increasing the frequency until the control performance starts to deteriorate. The variables used were the same as the nominal mode test with the exception of phase inductance, which was 5 mH. The first phase voltage frequency tested that showed moderate change to performance was 200 Hz.

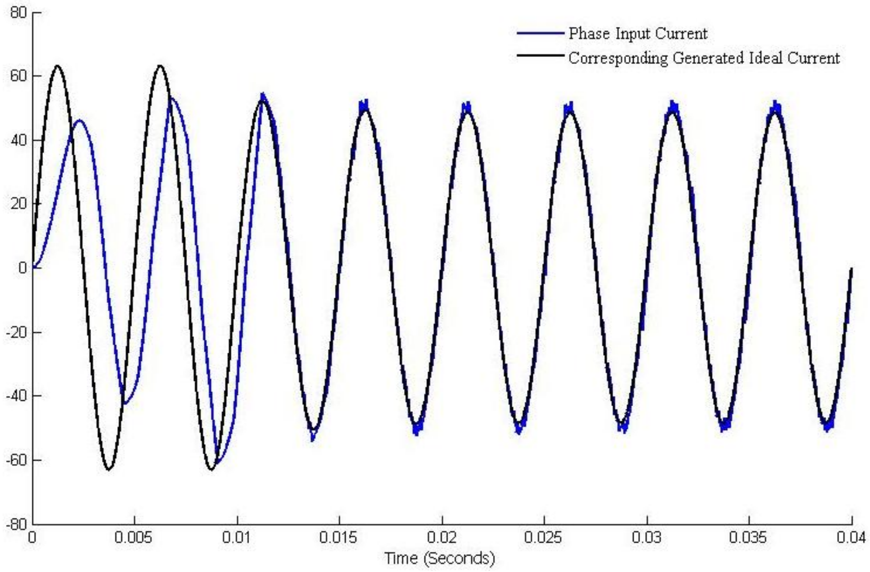


Figure 6.17 Phase and corresponding ideal current with phase voltage frequency at 200Hz.

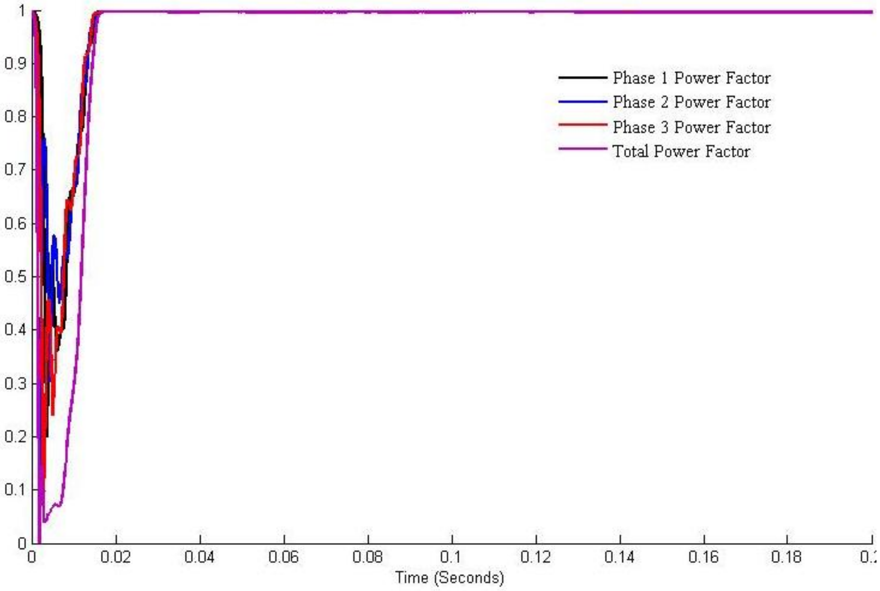


Figure 6.18 Power Factors with only phase term, phase voltage frequency 200 Hz.

Figures 6.17 and 6.18 show that at 200 Hz, the system will still work properly except that the settling time of the control is longer. There are no extra harmonics in the phase current. This is shown in Figure 6.17 by comparing the current to the control-

generated ideal current. There is also no major effect on the phase, as shown in Figure 6.18 using the phase term only power factors. A similar result was achieved for 220 Hz, with the settling time of the control being slightly longer. The control begins to fail at an phase voltage frequency of 240 Hz.

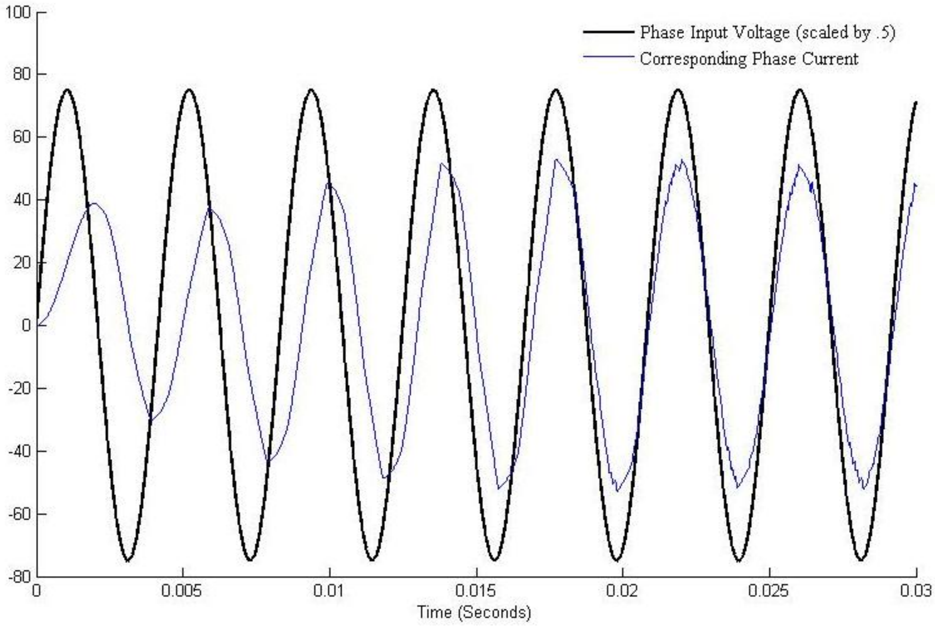


Figure 6.19 Phase voltage and current with phase voltage frequency of 240 Hz.

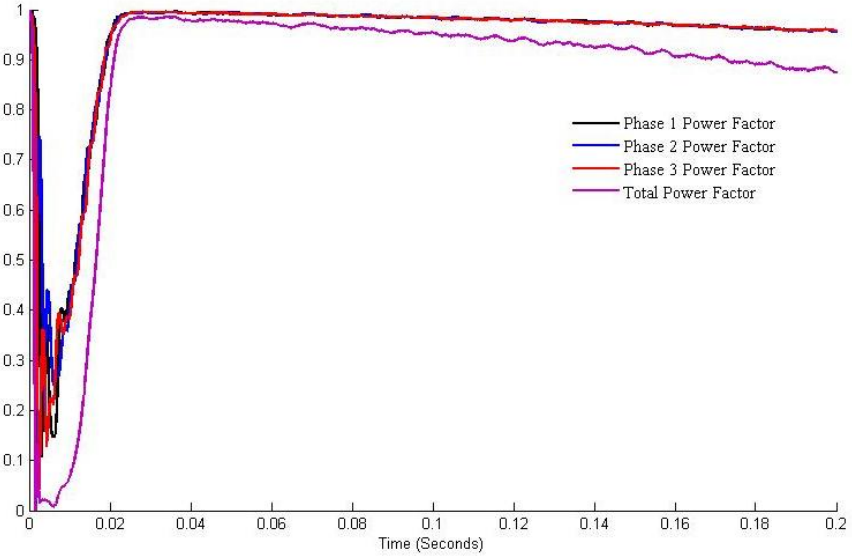


Figure 6.20 Power factors with only phase term, phase voltage frequency 240 Hz.

Figure 6.19 shows the first phase voltage and the corresponding phase current. The current amplitude eventually reaches the desired level, but the shape of the waveform is slightly incorrect and the current is out of phase with the voltage. Figure 6.20 shows the power factor calculated with only the phase term. The control seems work correctly initially, but then slowly starts degrading. This is because the control evaluation step size will not work at this frequency. The sign wave that generates the ideal current will have a slightly different frequency than the phase voltage, which will cause a gradually increasing discrepancy between the phase currents and phase voltages. This can be corrected by decreasing the step size of the control evaluation (increasing the control evaluation frequency).

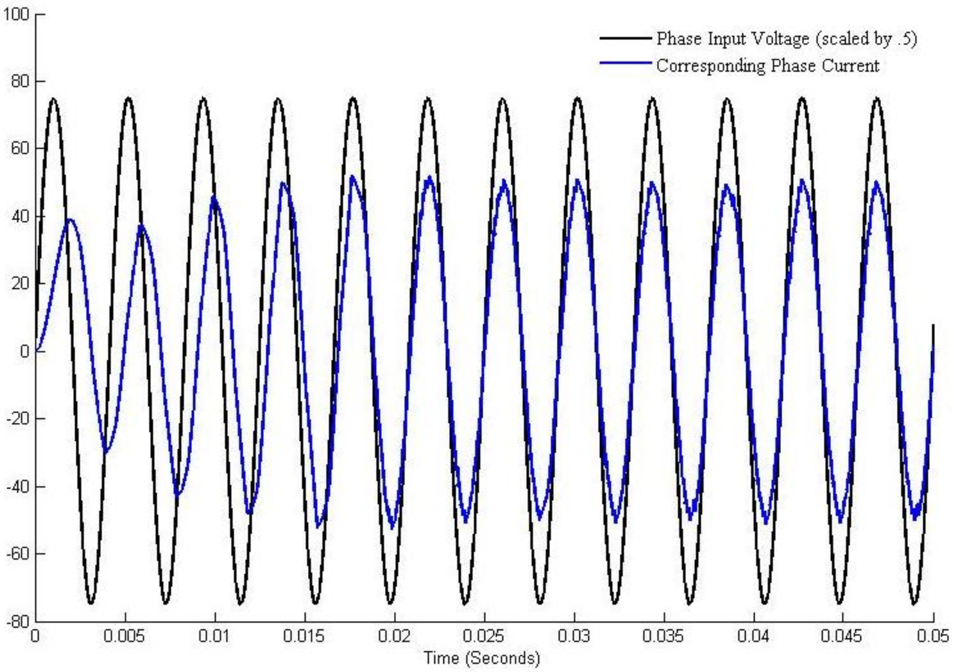


Figure 6.21 Phase voltage and current, phase voltage frequency 240 Hz and 1MHz control evaluation rate.

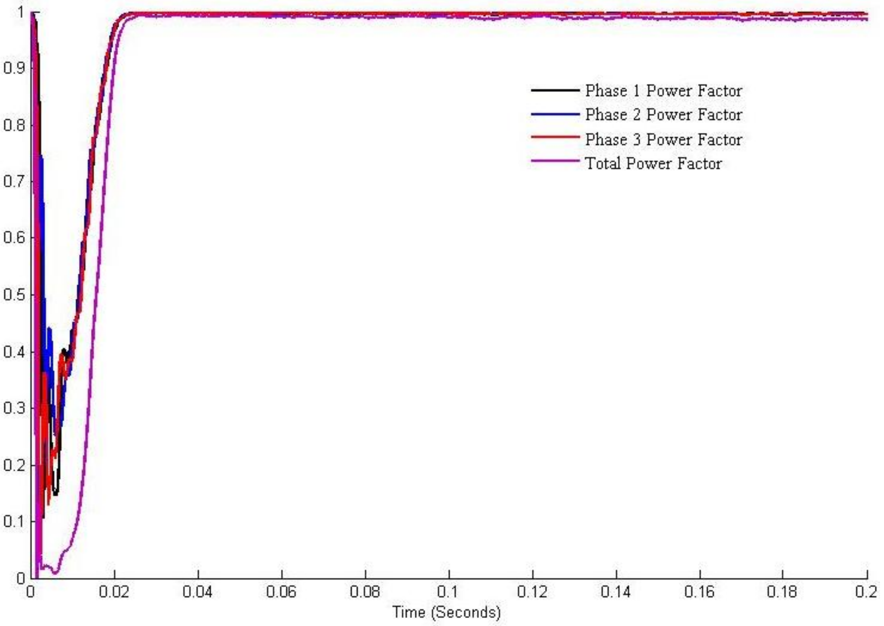


Figure 6.22 Power factors with only phase term, phase voltage frequency 240 Hz and 1 MHz control evaluation rate.

With the evaluation frequency increased to nearly the plant model evaluation frequency, the control once again begins to work properly. Figure 6.21 shows that the phase voltage and the current are in phase once again and the current appears to have the correct shape. The power factor is also restored, as shown in Figure 6.22. This is possible to do with a processor in the loop simulation since the control also runs in simulation time. With a real time application, it is unlikely this DSP could evaluate the control at this frequency.

6.4 Accommodation of Varying Load Resistance

Along with increasing the phase voltage frequency, the operator of the converter may also wish to accommodate a varying load resistance. If the load resistance never

changes and is known, it is not necessary to have an observer to calculate this resistance. Adding an observer allows the load resistance to remain unknown and also accommodates real time change of load resistance without restarting the plant system or the controller.

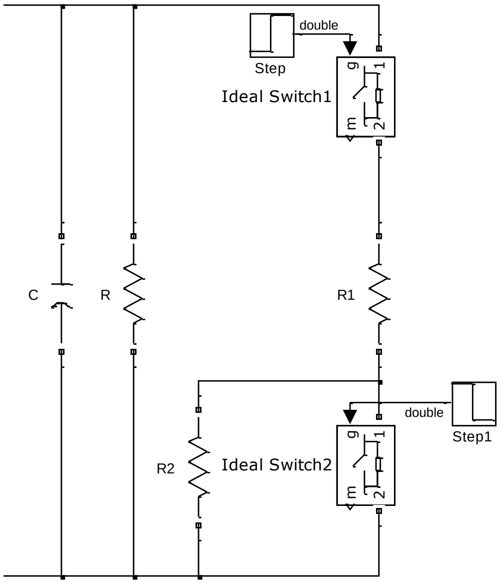


Figure 6.23 Model additions for load resistance evaluations.

In this implementation of the simulation, the nominal mode parameters were used with the exception of input inductance which was set to 5 mH. The load resistance starts at 40 Ω, at 0.1 seconds the resistance is dropped to 30 Ω, then at 0.2 seconds it is increased back to 40 Ω. Since the SimPowerSystems toolbox was used to build the plant model, a circuit had to be constructed to cause this resistance switching effect at the load. The plant model addition is shown in Figure 6.23. Resistor R has a value of 40 Ω, resistor R1 has a value of 120 Ω, and R2 has a value of 1 M Ω. The step function labeled only as step causes Ideal Switch1 to close at a time of 0.1 seconds, the 40 Ω resistor in

parallel with the 120 Ω resistor produces and overall load resistance of 30 Ω. The step function labeled step1 causes Ideal Switch2, which is normally closed, to open. The 1 M Ω resistor is the no longer shorted and a 40 Ω resistor in parallel with more than 1 M Ω produces almost exactly 40 Ω once again at the load.

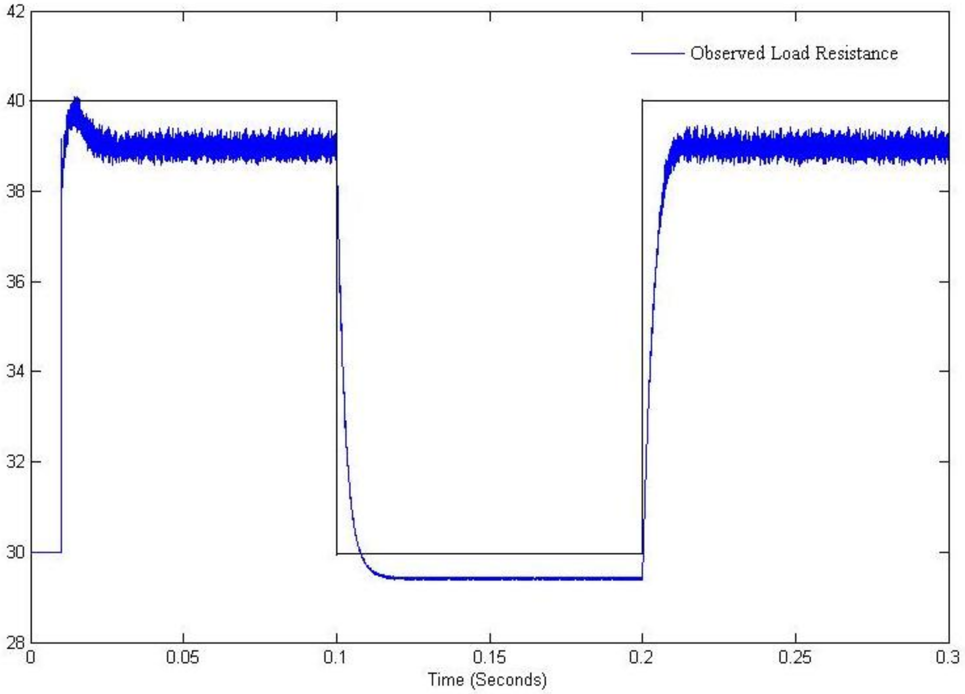


Figure 6.24 Observed varying load resistance.

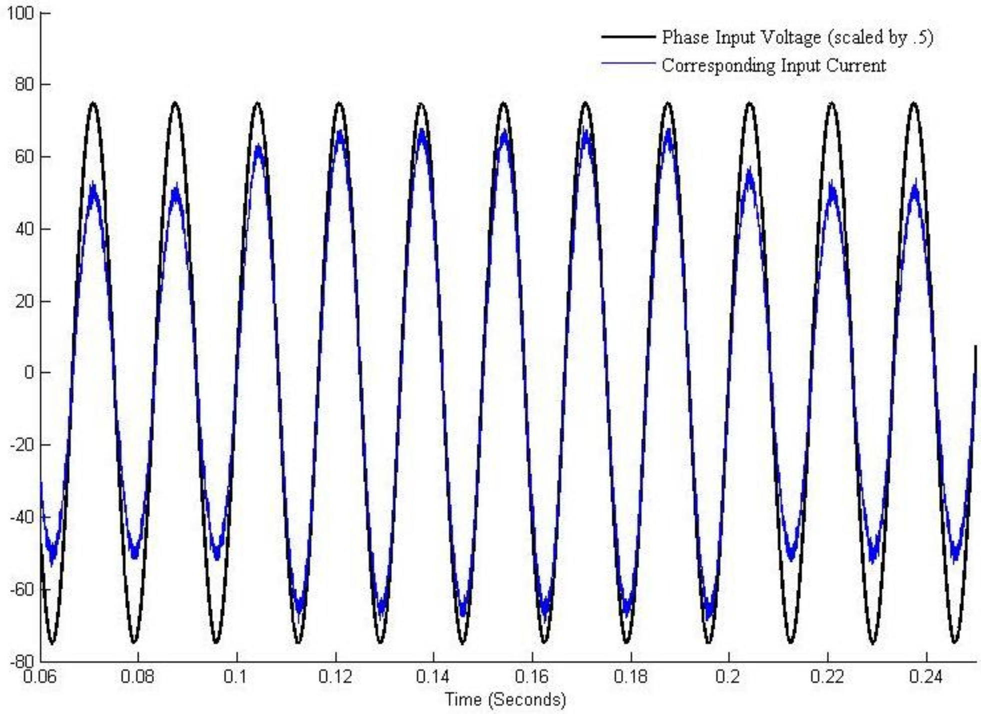


Figure 6.25 Phase voltage and current when load resistance varies.

The output of the observer can be seen in Figure 6.24. The black line in 6.24 represents the actual load resistance value. The observer still has the same slight bias due to the chattering of the controller. The important thing to note is that, when the load resistance changes at 0.1 and 0.2 seconds, the observer very quickly responds to the change and settles to the new resistance. The observer value is closer to the actual resistance when the load resistance is lower because when the phase current is higher the output power is naturally higher and the rippling does not affect as significantly. The first phase current is shown along with the first phase voltage in Figure 6.25. During the times when the load resistance is varying, there are no visible effects on the phase current. When the load resistance falls to 30 Ω, the current increases to accommodate the new resistance.

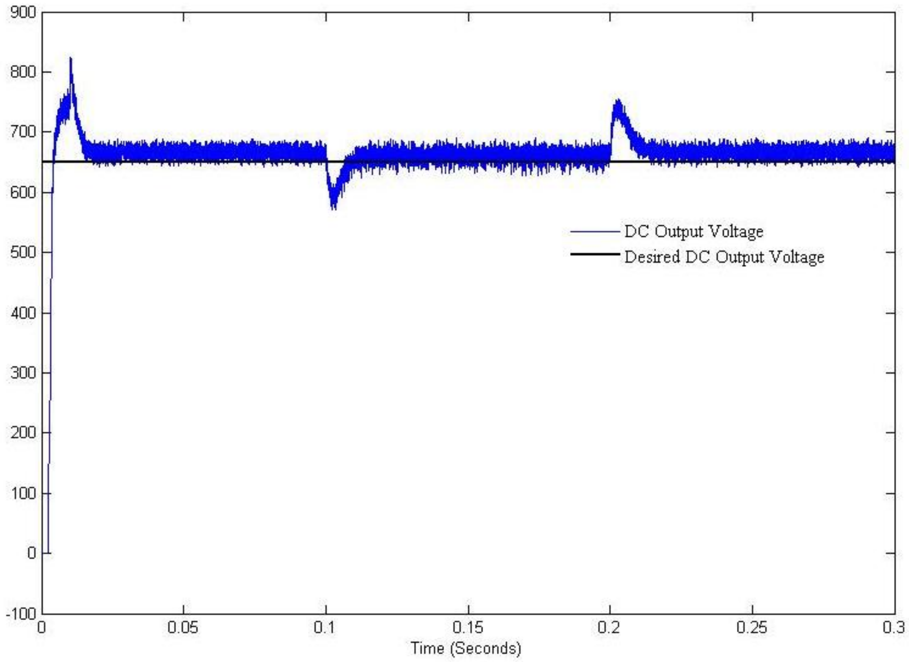


Figure 6.26 DC output voltage when load resistance varies.

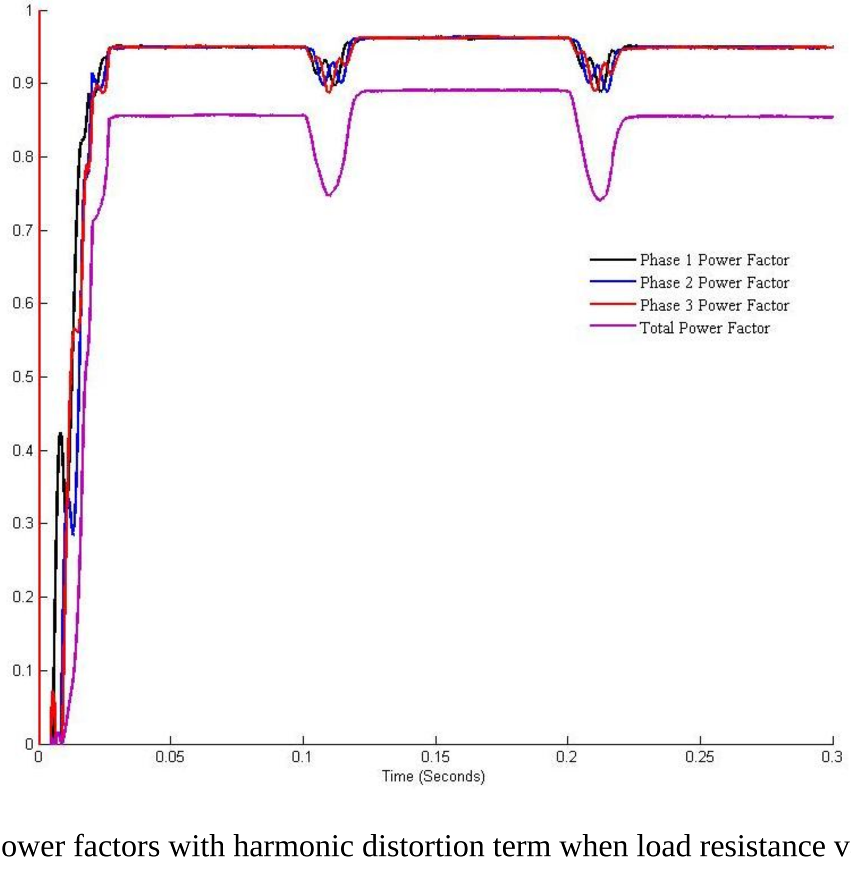


Figure 6.27 Power factors with harmonic distortion term when load resistance varies.

The change in the resistance shows up in the DC output voltage as small, short lived, positive or negative voltage spikes depending on whether the resistance is rising or falling. This is shown in Figure 6.26. The DC output voltage, like the observed load resistance, is closer to the desired value when the resistance is lower. Figure 6.27 shows the effect of varying resistance values on the power factors calculated with the harmonic distortion term. There is a short period in which the power factor is momentarily lowered, but then it quickly returns to a high value. Since the rippling is reduced at lower resistances (higher phase current), the power factor is also slightly higher.

6.5 Summary

The sliding mode control algorithms for power factor correction in three phase AC-DC converters have been implemented on the DSP. The plant model was simulated on a computer using SimPowerSystems in Simulink [18]. The two components communicated in a processor in the loop simulation developed using Simulink and CCS [19]. The results of the co-simulation, gathered using Simulink, were reported in this chapter. Four types of evaluations were made using this co-simulation.

The nominal mode study tested the controller under some pre-selected, baseline parameter values. In this test, significant rippling in the phase current due to chattering affected the performance of the resistance observer and had a negative impact on the power factor. The traditional power factor, based only on the phase component was very close to unity (0.99) for all three phases and the overall. With the harmonic distortion

term included, the power factors were considerably lower. Each phase had a power factor near 0.88, the overall was approximately 0.68.

The study of varying phase inductance was performed in order to correct the harmonic distortion term in the power factor. In order to reduce the harmonic distortion and increase the total power factor, the phase inductance was increased to 5 mH and then to 10 mH. Both inductances produced cleaner currents with reduced higher harmonics and elevated power factors. The 5 mH test produced individual phase power factors of approximately 0.96 and an overall power factor of 0.88 with the harmonic distortion term included. The 10 mH inductor increased the power factor even higher to 0.97 with an overall close to 0.92. Using a 10 mH inductor is physically impractical due to the size of the inductor. The 5 mH inductor has proved to show a major improvement over the 2 mH inductor and therefore was selected for use in the latter co-simulations.

The study of varying phase voltage frequency was completed in order to find the upper bound of the phase voltage frequency at which the control will still work without changing the control evaluation rate. This frequency was found to be approximately 220 Hz. Above this frequency, the control performance starts to degrade due to integration rates and sine function sampling rates. At 240 Hz, the input currents become out of phase with the input voltages, resulting in a steadily increasing degradation of power factor. The solution to this problem is increasing the control evaluation frequency. The limit to which the control evaluation frequency can be increased depends on the processing speed of the digital processing device on which the control is encoded.

The study of varying load resistance was the final study. The system was implemented under nominal conditions with an increased phase inductance to reduce

rippling and with varying load resistances. The study proved that the control handles varying loads very well, as expected. The observer started tracking the value of the new resistance very quickly, and the phase currents adjusted appropriately. Only a small amount of spiking occurred in the DC output voltage and the power factor was quickly restored.

The most significant outcome of this study can be drawn from the fact that the controls in all of the studies were implemented on a DSP. This means that the code for each part of the control was generated correctly and works properly under all the desired test conditions. Since the model used in the co-simulation was originally built as a model in Simulink, the control can be easily transferred to a separate model to be used for encoding the device for use with a physical plant model. In order to implement the control in application with a physical plant model only a few minor changes need to be made. These changes include removing the data type conversions at the control interface and addressing the control inputs and outputs to pins on the DSP board (which can be found in [17]). Optionally, control designs using the higher order sliding mode controls described in [4] may be implemented as well. The DSP has built in pulse width modulators which may be useful for implementing these algorithms.

CHAPTER 7

CONCLUSION

In this thesis, a control algorithm was implemented and studied on a DSP device using a processor in the loop co-simulation with a computer. The purpose of the control was to improve the power factor of a three phase AC-DC boost converter under varying loads. The type of algorithms used were all sliding mode controls, including a classical sliding mode control [5] for the main control and a super twisting observer/differentiator [7, 6] to observe the load resistance. The processes for creating the co-simulation using MATLAB with Simulink [18, 19] and Code Composer Studio was described in detail. Multiple tests were performed in order to be sure that the control algorithms implemented work properly on the DSP in simulation time and produce results comparable to a computer only simulation. Theoretical improvements and changes to the system such as increasing the phase inductance and increasing the phase voltage frequency were also evaluated using the DSP implementation of the control. The conclusion formed from the results of this implementation is that the control works on an embedded processor in simulation time.

The optimal conditions for the control were found during the testing phase. The phase inductance should be higher than 2 mH but less than 10 mH for practical physical

implementation. The phase voltage frequency should be less than 240 Hz, assuming a control evaluation frequency of 100 kHz. It is possible to make the controller work in theory at higher frequencies but this requires a higher control evaluation rate which may not be practical depending on the embedded processor chosen. In the load resistance tests the controller withstood changes to load resistance as expected.   When calculating the power factor in the traditional sense of using only the phase angle, all simulations except those where the phase voltage frequency was over 240 Hz produced power factors of 0.97 and higher. The higher harmonics present in the phase currents caused the power factors to be slightly lower if calculated with the harmonic distortion term. These harmonics also cause the output voltage level to be slightly higher than the desired value, and the resistance estimator to be inaccurate by a value of one ohm or less.

The processor in the loop implementation produced a reasonable result with nearly the same performance as the results of previous computer-only simulation studies [3, 4]. It has now been proven that the control code works when implemented on a DSP when operating in a loop with the computer through the JTAG emulator. This result is significant because all the control calculations were performed by the embedded processor instead of the computer. The precision of DSP is adequate enough to produce a result comparable to a computer. This further validates the concept of using sliding mode controls for power factor correction in switching power converters beyond computer only simulations.

Transferring the control model, originally built in Simulink, for this study to an application with an actual three phase AC-DC boost converter is a simple process. The control model can be implemented by first removing the data type conversion interfaces

and addressing the inputs and outputs to pins of the DSP then transferring the model to a blank model and using the Simulink embedded coder to encode the DSP [Coder]. A hardware interface between the embedded device and the converter block to reduce the power of the signals going to the device and boost the voltage of the signals coming from the device must also be developed in order to utilize this implementation. The switching control signal can be generated by using the sign function but it may be more advisable to use the built in PWM generators, which will generate the corresponding switching signals. This way, a dead zone can easily be set up within the signals to prevent control and plant system failure. The super twisting control algorithm may also be tested as designed in [4]. The super twisting algorithm will produce a slightly more accurate result but will also require the use of pulse width modulation.

REFERENCES

[1] C. Basso, *Switch-Mode Power Supplies: SPICE Simulations and Practical Designs*. Columbus, OH: McGraw-Hill, 2008.

[2] R. Tarter, *Solid-State Power Conversion Handbook*. New York, NY: Wiley, 1993.

[3] Y. Shtessel, S. Baev, and H. Biglari, “Unity Power Factor Control in 3-phase AC/DC Boost Converter using Sliding Modes,” *IEEE Transactions on Industrial Electronics*, Vol. 55, No. 11, 2008, pp. 3874–3882.

[4] R. Schaeffel, “Power Factor Control of a 3-Phase AC/DC Boost Converter Using Nonlinear Control Techniques,” Huntsville, AL, 2009.

[5] C. Edwards and S. Spurgeon, *Sliding Mode Control*. Bristol, PA: Taylor & Francis, 1998.

[6] A. Levant, “Higher-order Sliding Modes, Differentiation and Output-Feedback Control,” *International Journal of Control*, Vol. 76 (2003), Nos. 9/10, pp. 924–941.

[7] V. Utkin, J. Guldner, and J. Shi, *Sliding Mode Control in Electromechanical Systems.* Philadelphia, PA: Taylor and Francis, 1999.

[8] K. D. Gussem, D.M. Van de Sype, A.P. Van den Bossche, and J.A. Melkebeek, “Input Current Distortion of CCM Boost PFC Converters Operated in DCM,” *IEEE Transactions on Industrial Electronics,*Vol. 54, No. 2, pp. 858-865, Apr 2007.

[9] W. Zhang, G. Feng, Y. Liu, and B. Wu, “A Digital Power Factor Correction (PFC) Control Strategy Optimized for DSP,” *IEEE Transactions on Power Electronics*, Vol. 19, no. 6, pp. 1474–1485, Nov 2004.

[10] H. M. Suryawanshi, M. R. Ramteke, K. L. Thakre, and V. B. Borghate. “Unity-Power Factor Operation of Three-Phase AC–DC Soft Switched Converter Based On Boost Active Clamp Topology in Modular Approach,” *IEEE Transactions on Power Electronics*, Vol. 23, No. 1, Jan 2008.

[11] H. Yiwen, X. Weisheng, and W. Hongrun, “Sliding Mode Control of a Single Phase AC/DC Boost Converter Using Time Varying Sliding Surface,” *IEEE International Conference on Intelligent Computation Technology and Automation*. 2008

[12] Nilsson, Riedel, *Electric Circuits*. Upper Saddle River, NJ: Pearson Prentice Hall, 2008.

[13] A. Kloss, *A Basic Guide to Power Electronics*. New York, NY: Wiley,

[14] A. Levant, "Quasi-Continuous High-Order Sliding-Mode Controllers," IEEE Transactions on Automatic Control, Vol. 50, No 11, November 2005, pp. 1812–1815.

[15] *MPC555 / MPC556 User's Manual*, Motorola, 2000. [Online]. Available: www.freescale.com

[16] *TMS320F28335, TMS320F28334, TMS320F28332, TMS320F28235, TMS320F28234, TMS320F28232, Digital Signal Controllers (DSCs)* Data Manual. Texas Instruments, Dallas, TX, 2011.

[17] *ezDSP*^{TM} *F28335 Technical Reference*, Spectrum Digital Inc., Stafford, TX, 2007.

[18] *SimPowerSystems 5.4*, The MathWorks, Inc., Natick, MA, 2011. [Online]. Available: www.mathworks.com

[19] *Simulink Coder 8.0*, The MathWorks, Inc., Natick, MA, 2011. [Online]. Available: www.mathworks.com