DESIGN OF A RESONANT SOFT SWITCHING POWER SUPPLY FOR STABILIZED DC IMPULSE DELIVERY

by

Andrew H. Seltzman

A thesis submitted in partial fulfillment of the

requirements for the degree of

Master of Science

(Electrical Engineering)

at the

UNIVERSITY OF WISCONSIN-MADISON

2012

APPROVED

By

Adviser Signature: _____

Adviser Title:_____

Date:_____

Abstract

This thesis addresses the issues involved in the design and construction of a multi-phase resonant switching power supply for delivery of a high voltage, high current stabilized DC impulse. Such a power supply may be used in place a pulse forming network (PFN) to drive a high power klystron amplifier, which typically requires voltages near -100kV at 10s of amps of current. Unlike an LC PFN, a switchmode power supply (SMPS) allows greater control over pulse duration while still allowing generation of longer duration pulses on the order of 10ms with constant output voltage by use of feedback regulation.

Specifically, the thesis documents the results from the design of a loosely coupled boost transformer with a parallel LC resonator on the secondary, a microcontroller based control system for feedback stabilization and techniques of harmonic mitigation to reduce switching noise on the output waveform.

Acknowledgements

I would like to thank Giri Venkataramanan, Paul Nonn, Bob Ganch, Jason Kauffold and Jay Anderson for assisting with this project. I would also like to thank the UW Madison Department of Physics, and Los Alamos National Lab (LANL) for providing support for this research.

Table of Contents

List of Figures

List of Tables

Nomenclature

ADC	Analog to Digital Converter
DSP	Digital Signal Processing
ESL	Effective Series Inductance
IGBT	Insulated Gate Bipolar Transistor
IRQ	Interrupt Request
ISR	Interrupt Service Routine
MIPS	Million Instructions Per Second
OFC	Oxygen Free Copper
PFN	Pulse Forming Network
PWM	Pulse Width Modulation
SCR	Silicon Controlled Rectifier
SMPS	Switch Mode Power Supply
UART	Universal Asynchronous Receiver and Transmitter
ZCS	Zero Current Switching
ZVS	Zero Voltage Switching

Introduction

0.1 Overview

The use of resonant circuits in high voltage switching power converters allows the voltage boost ratio of the transformer to exceed the turns ratio, allowing for more compact designs and reduction of copper usage in the secondary winding. Further, the boost ratio's strong dependence on switching frequency allows the power supply to regulate output voltage by shifting switching frequency. By switching at full duty cycle near resonance, the primary voltage and current will be in phase, allowing for zero current switching (ZCS), and almost complete elimination of switching losses. As switching frequency moves away from resonance, the IGBT current at the switching event increases from zero, however with only two switching events per cycle, switching losses are considerably lower than if PWM feedback were used for voltage control. Additional benefits of resonant topologies include the strong dependence of power transfer on a matched output load; in the event of a short circuit, the resonant circuit will be de-Qed and power transfer will automatically reduce without damage to the supply or load.

The power supply described herein is a resonant three phase pulsed power converter capable of a nominal output of -80kV at 40A for 10ms. The input dc link of the power supply is connected to an electrolytic capacitor bank with a nominal starting voltage of 900V and sufficient capacitance to drive the supply for 10ms. The capacitor bank is connected to the

input of an h-bridge module that drives the boost transformers. Each transformer primary is independently driven by a dedicated IGBT H-bridge circuit that is fiber optically coupled to a microcontroller based control system. The transformers are driven with a full duty cycle square wave of varying frequency between 18.5kHz and 26kHz and have microcrystalline iron cores for low loss while providing adequate volt-seconds. Each resonant transformer has a parallel LC resonant circuit on the secondary winding. The secondary windings are configured in a Y configuration and connected to a voltage doubling three phase rectifier, where the output of the rectifier feeds a center tapped capacitor bank with the center tap connected to the Y point of the transformer secondaries. The output of the rectifier is also connected to a harmonic filter tuned to the 6^{th} harmonic of the average operating frequency to greatly reduce ripple on the power supply output. Additionally, an LC low pass filter may be connected in series with the output to further reduce ripple at the expense of a slightly increased rise time of the HV pulse.

The power supply is connected to a klystron tube load with a nominal impedance of 1800ohm through an RG-8 solid dielectric coaxial cable. An RC snubber consisting of a 400uH inductor in parallel with a 50ohm resistor prevents an HV pulse from being reflected back up the coaxial cable in the event the output of the cable is shorted to ground by an internal HV arc. Additionally, a triggerable spark gap is placed in parallel with the klystron tube to crowbar the voltage across the cathode in the event of an internal arc in order to prevent arc damage to the cathodes emissive surface.

0.2 Research Contributions

The main contributions of this research address the design and construction of a three phase resonant power supply. Specifically the contributions can be classified into the following groups:

i. Design and analysis of a loosely coupled resonant transformer with a boost ratio that significantly exceeds the turns ratio with the capability of high voltage, high current output. The transformer in question includes a nano-crystalline iron core for low losses, and a specialized oil tank enclosing only the secondary winding for insulation and corona suppression.

ii. Design of a suitable microprocessor based control system to generate switching signals to the IGBT module over a fiber optic link and receiving feedback signals from capacitor bank current, capacitor bank voltage, and output voltage. Maintaining good resistance to electromagnetic interference from high current switching events, and providing ground loop isolation to input signals. Providing computer control to adjust the output pulse parameters. Programing of a feedback control system to stabilize output voltage as the capacitor bank voltage decreases by adjusting switching frequency.

iii. Mitigation of harmonic noise and ripple in the output through use of harmonic filters tuned to certain multiples of the switching frequency, inductive filers to make the output current stiff or low pass filters to selectively attenuate higher harmonics while still allowing rapid rise time at the beginning of the pulse.

iv. Design of a safety system for driving klystron tubes, that in the event of an internal arc, will cut output power from the supply, rapidly remove power from the load by use of a crowbar spark gap and prevent high voltage transients on the coaxial cable connecting the power supply to the load from being reflected back and damaging the supply.

0.3 Overview of Chapters

Chapter one presents a review of state of the art designs of high power SMPS design, topologies, and soft switching. A review of existing high power resonant SMPS designs is presented along with results from their testing. Techniques for harmonic mitigation are reviewed.

Chapter two presents the requirements and design constraints for construction of the power supply. While later chapters describe the design of the power supply, this chapter provides insight into why the power supply was designed in this particular manner based on the components available. The power supply was constructed to replace an LC based PFN for delivering power to a microwave amplifier. Requirements on the power supply include output voltage and current capabilities, fault tolerance, output voltage range, output stability, pulse duration, serviceability, safety, tolerance for voltage droop on the capacitor bank, and external control of parameters. Design constraints include the use of certain transformer cores, IGBTs, rectifier diode stacks, and capacitor banks which were either donated to the project or were available as surplus.

Chapter three presents the design of the power supply's power electronic components, transformers, filtering system and safety systems. Components include the capacitor bank's

capacitor trays, bus bars and interconnecting wires, the IGBT switching network's mechanical relay, stiffening capacitors, low inductance bus plates, IGBTs, gate drivers and transformer connections, a detailed description of the resonant transformer, it's design, testing and comparison to numerical and analytical models, the three phase rectifier and it's doubling capacitor configuration, techniques of harmonic mitigation, and safety systems to prevent damage in the event of an arc fault.

Chapter four presents the design and construction of the power supplies control system. This chapter covers the microcontroller, its connecting circuits, i/o optical and galvanic isolators, and the design of the operating code.

Chapter five presents results from testing of the power supply and comparison to spice and analytical models. Models of the transformer, rectifiers, and filters are presented and compared to simulation. Open loop testing of the power supply is presented and compared to steady state spice simulations. Testing of the feedback control system and stabilization of output voltage during drooping capacitor bank in presented. Testing of the crowbar spark gam and associated LR snubber system is presented and compared to simulation.

Chapter six presents a brief summary of the research, presents conclusions and recommendations for resonant SMPS design, and outlines future research.

Chapter 1 State of the Art Review

This chapter presents an overview of current research and development of pulsed power systems utilizing switched mode power supply (SMPS) systems for modulation and pulse stabilization. The first section presents several designs of high power SMPS, and high voltage SMPS topologies. The second section presents designs and characteristics of loosely coupled transformers. The third section covers resonant SMPS power converters, specifically a very similar design of a three phase resonant converter built at Los Alamos National Lab and results of its testing. The fourth section presents techniques of harmonic mitigation. The final section presents a summary of the chapter.

1.1 High power SMPS Design and Topologies

Design of high power SMPS converters is challenging due to the high voltages and currents involved. Power electronics must be capable of multi-megawatt power transfer, sometimes at thousands of volts and thousands of amps. The majority of high voltage power converters utilize either boost, flyback, or tightly or loosely coupled transformer based topologies. Boost and flyback converters are usually limited to low power operation, while tightly coupled high voltage systems may present safety hazards in the event of an output short due to the low impedance between the primary and secondaries of the transformer.

Poly-phase resonant power converters are a new method to generate high voltages with high power while maintaining a small physical size of 10 times smaller than previous methods. Such power converters are capable of producing 10s of MWs at 100s of kV. Additional benefits include inherent fault tolerance; the power converter must be designed to be matched to a given load such that in the event of a fault, the resonant circuit will be de-Qed thereby preventing power transfer. In the event of an arc fault, the system can safely run through the fault without damage to the load or supply, while the reduction in power transfer may be sufficient to clear the arc fault. Resonant power converters with amorphous nano-crystalline iron cores maintain the high permeability of iron cores, while allowing efficient use at higher frequencies in the 10s of kHz usually reserved for ferrite materials. Consequentially, a nano-crystalline transformer can provide 300 times the power transfer capability as a 60Hz transformer for a given size and weight [1]. For comparison, a 100kV, 60Hz system carrying 20Arms and utilizing soft iron cores will be on the scale of 35 tons and

have about 30kW losses while the transformer in a polyphase resonant converter operating at 140kV and 20kHz carrying 20Arms utilizing a nano-crystalline core will weight 450lbs and have about 3kW losses. Efficiency of as high as 97% may be realized with such a system. Such resonant systems can achieve significantly greater power levels with a given set power electronics in part to the resonant nature of the secondary which allows soft switching thereby reducing junction heating in the IGBTs. Given the exact nature of the system, either zero voltage switching or zero current switching may be utilized by switching the IGBTs during the period of reverse commutation of the anti-parallel diodes in the IGBT module, or by operating near resonance such that the IGBTs switch near the zero crossing period of primary current.

1.2 Loosely Coupled Transformers

In contrast to closely coupled inductive systems, such as a tightly wound transformer with low leakage inductance, loosely coupled transformers provide a high leakage inductance that may be used to form a resonant circuit. In many cases efficient power transfer may only be attained if either or both the primary and secondary windings are capacitively compensated [2], described herein as a resonant transformer. In closely coupled reactive power systems, the reactive power is usually less than the real power transferred, while in certain loosely coupled systems, the reactive power can be up to 50 times the real power.

The majority of transformers used in the power supplies herein utilize a tightly wound uncompensated primary and a loosely coupled secondary with large leakage inductance and parallel capacitive compensation. Series compensation of the secondary leads to voltage source like characteristics while parallel compensation leads to current source like characteristics. Series compensation of the primary is utilized when the amplitude of the input waveform must be low, while parallel compensation is used when the primary winding must be concentrated into a small volume, thereby requiring high currents. The Q factor of a compensated winding is given by (1.1) as and is typically between 2 and 10.

$$Q = \frac{VAR}{P} \tag{1.1}$$

Larger secondary Q factors allow greater power transfer at the expense of a higher secondary VA utilization. The power transfer capability of a resonant transformer can be increased by increasing the VA rating of the secondary by adding more copper or core cross section, increasing the primary current or varying the coupling of the windings.

1.3 Resonant SMPS Power Converters

Research in high power resonant power converters has been carried out at Los Alamos National Labs. The resonant power supply was designed to produce a high voltage pulse train of 140kV, at 1MW average, and 11MW peak that was used to drive klystron amplifiers. The output waveform was a chirp of 1.5ms pulses. The power supply drives the resonant transformers at a fixed 20kHz frequency using PWM to regulate the output waveform through feed forward and feedback techniques as shown in Figure 1.1 [3]. The supply used a resonant transformer with a nano-crystalline iron core and secondary LC resonator to achieve voltage boost to the required value. Zero voltage switching of the IGBTs was used to minimize switching losses.

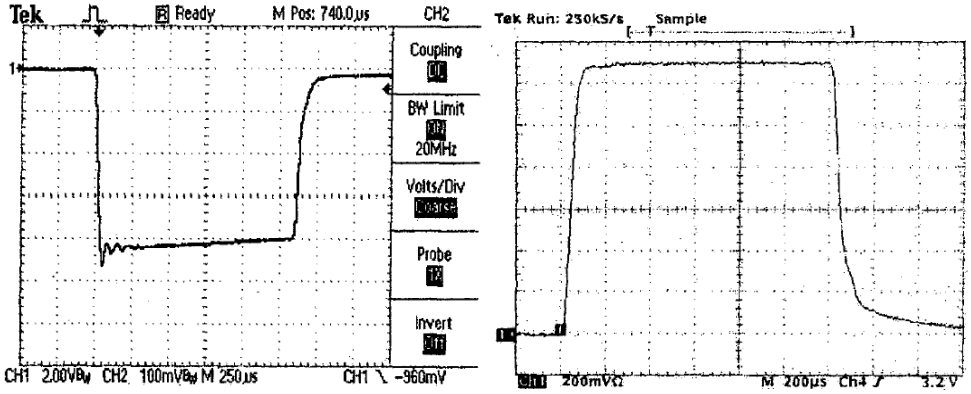


Figure 1.1 Power supply output pulse without and with feedback control.

The input DC link of the power converter used a bipolar capacitor bank charged to a nominal +/- 1250 V through a three phase SCR regulator connected to a 2100V substation. The capacitor bank utilized a series of custom low inductance metalized hazy polypropylene traction capacitors, designed to clear any internal short that may occur, thereby allowing

direct connection in parallel banks without individual capacitor safety fuses, which would increase inductance of the input DC link.

Each transformer was driven by an independent full h-bridge of IGBTs. The IGBTs used in this power supply were rated at 3.3kV and 1.2kA. The boost transformers utilized an amorphous nano-crystalline iron core operated at 20kHz with a bidirectional magnetic field swing of 1.6 Tesla. A parallel LC resonator was used on the secondary winding to achieve a high boost ratio. The transformers were measured to have 320W of loss per core during full power operation. The loosely coupled resonant secondary allows a boost ratio of 60:1 while using a turns ratio of 19:1 [4]. The turns ratio of the transformer is chosen to provide the required leakage inductance on the secondary to achieve a 20kHz resonant frequency with the parallel resonant capacitors.

The transformer cores were developed by National Arnold Magnetics and are constructed of 0.0008” nano-crystalline laminates for low loss at high frequency operation while maintaining a high magnetic permeability. The characteristics of the core material are presented in Table 1-1.

Table 1-1 Nano Material Characteristics [4]

Mu	50,000
Lamination Thickness	.0008"
Lamination Insulation	1 µM Namlite
Stacking Factor	~90%
Bsat	12.3 kG
Core Loss (our use)	~300 W
Core Weight (our use)	~95 lbs
Power (each core)	330 kW

The IGBT network consists of a set of 12 IGBTs configured in a set of three independent full h-bridges, with the top of each bridge connecting to the positive capacitor bank at +1250V and the bottom connecting to the negative capacitor bank at -1250V as shown in Figure 1.2.

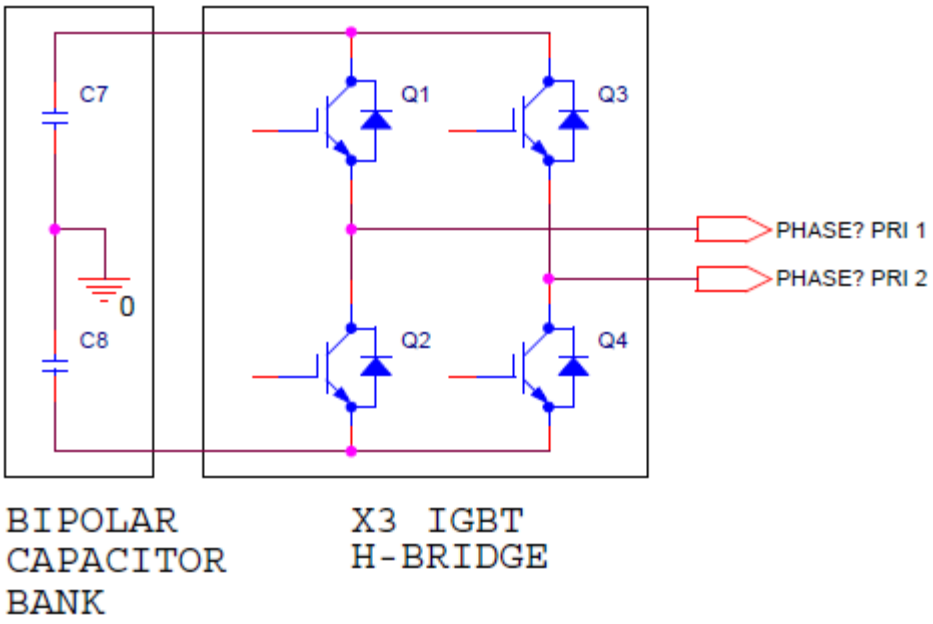


Figure 1.2 Bipolar bank and IGBT h-bridge

The IGBTs are mounted on a low inductance buss plate with a rail to rail inductance of 4nH, allowing snubberless operation of the IGBTs by preventing overvoltage conditions generated by high frequency ringing during switching events, where dI/dt can exceed 10kA/uS [5]. Total inductance of the input DC link is further reduced by using low ESL stiffening capacitors mounted directly to the IGBT bus plates providing a total input inductance on the order of 7nH [6].

The power supply uses zero voltage switching to reduce switching losses, where primary winding current is carried by the IGBTs freewheeling diode during the switching event. The

output voltage of the supply is controlled by PWM of the duty cycle of the switching period between 2.5us and 55us per half cycle which provides approximately 10% control range on the voltage output which is sufficient to stabilize output voltage during capacitor bank droop. PWM period is controlled by an adaptive feed forward / feedback system which provides very low ripple on the order of 150V. The voltage waveform on the transformer primary is shown in Figure 1.3.

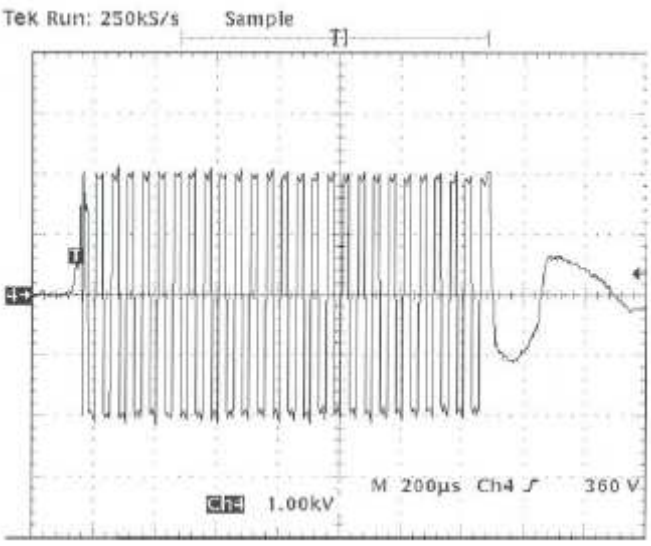


Figure 1.3 Transformer primary waveform [4].

The secondaries of the transformers are connected in a Y configuration, each transformer with a parallel resonant capacitor as shown in Figure 1.4.

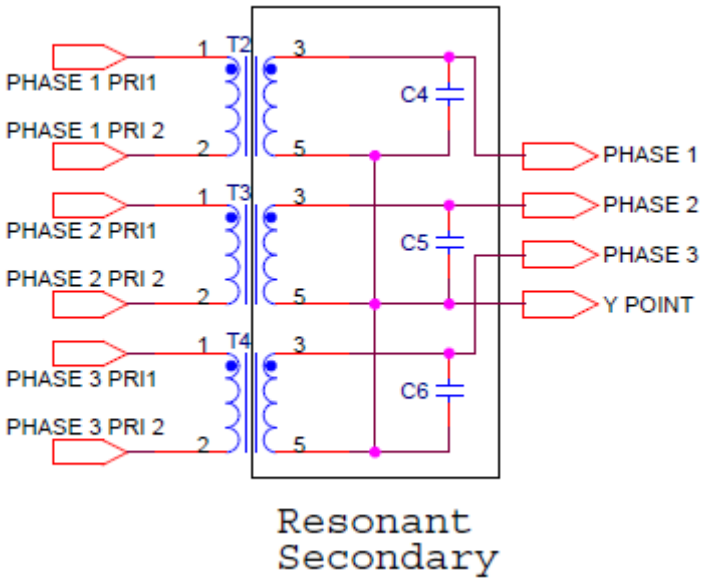


Figure 1.4 Three phase resonant transformer system.

The three output phases are connected to a three phase rectifier feeding a center tapped capacitor bank with the center tap connected to the Y point on the transformer secondaries as seen in Figure 1.5. The rectifier consisted of a string of 1.4kV, 75A diodes with a 50ns reverse recovery time [7].

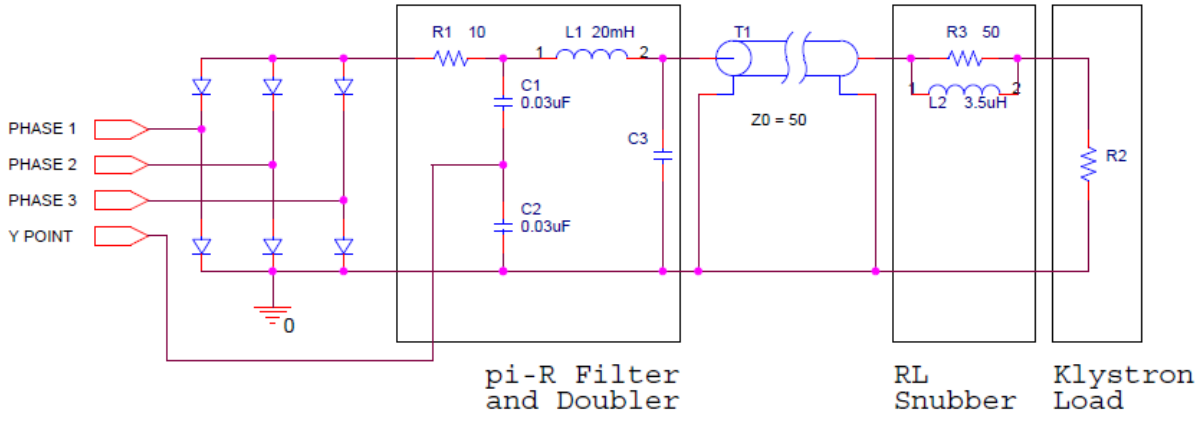


Figure 1.5 Rectifier, doubler, filter and RL snubber.

A resistor is placed in series, forming part of a pi-R filter, along with a series inductor and shunt capacitor after the center tapped doubling capacitor. The power supply is connected to the klystron load through a coaxial transmission line and a series RL snubber.

Testing of a crowbar system demonstrated de-Qing of the resonant transformers, automatically interrupting power transfer to the klystron tube. In the event that modulation on the transformer primary continued, only a slight increase in total energy dissipated into the klystron, on the order of 10J, was observed.

Other resonant transformer topologies have been explored for high voltage, high power modulators [8]. A long pulse modulator capable of producing a 25kV, 10A pulse of 1-2ms has been developed at E2V technologies. In this system, a unipolar electrolytic capacitor bank powers three independent full h-bridges that drive a resonant LC tank circuit connected across the primaries of three ferrite core transformers. The secondaries of the transformers are connected in a Y topology with the center tap floating. The output of the transformers is then rectified with a three phase rectifier and connected to an LC low pass filter to reduce harmonics and ripple.

In this particular power supply, the input DC link is powered of a unipolar capacitor bank charged from a three phase IGBT PWM rectifier designed to maintain the capacitor bank in a suitable voltage range while drawing power from the AC mains at unity power factor to comply with harmonic and flicker regulations. The size of the capacitor bank is minimized by utilizing intelligent charging methods and designing the power converter with a suitable dynamic boost range to compensate for increased voltage droop during each output pulse. With the bank used in this power supply, a voltage droop of 25% is expected during each

pulse. The capacitor bank is comprised out of electrolytic capacitors connected to a low inductance bus plate in a unipolar configuration, and powering three independent h-bridges as shown in Figure 1.6.

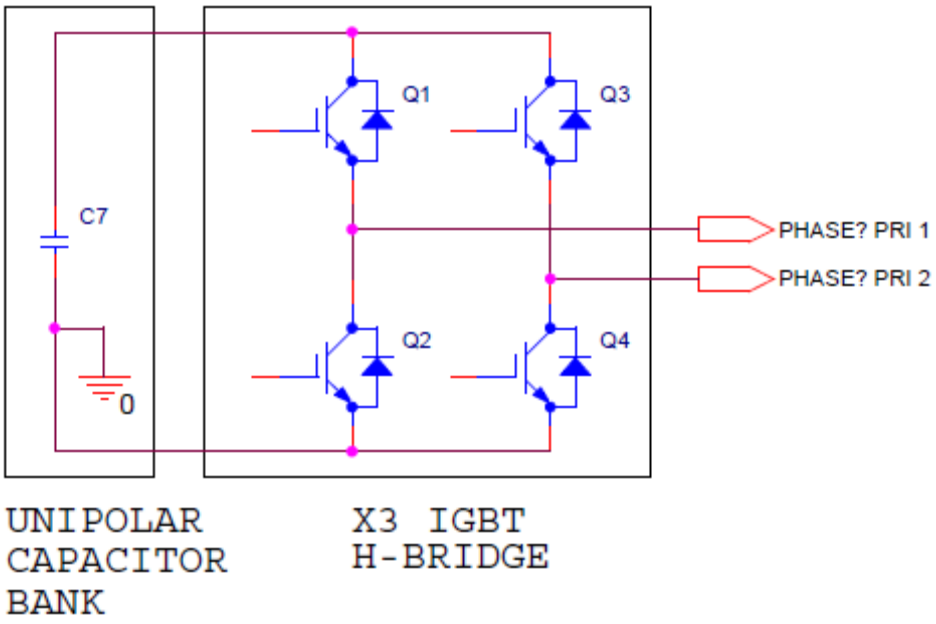


Figure 1.6 Unipolar bank and IGBT h-bridge

Each h-bridge drives a resonant boost transformer, configured with a resonant LC tank circuit on the primary as shown in Figure 1.7 with each transformer primary in parallel with the capacitor in the tank circuit. The transformer is constructed out of ferrite I-bars and has single layer primary and secondary windings. The cores are magnetically independent.

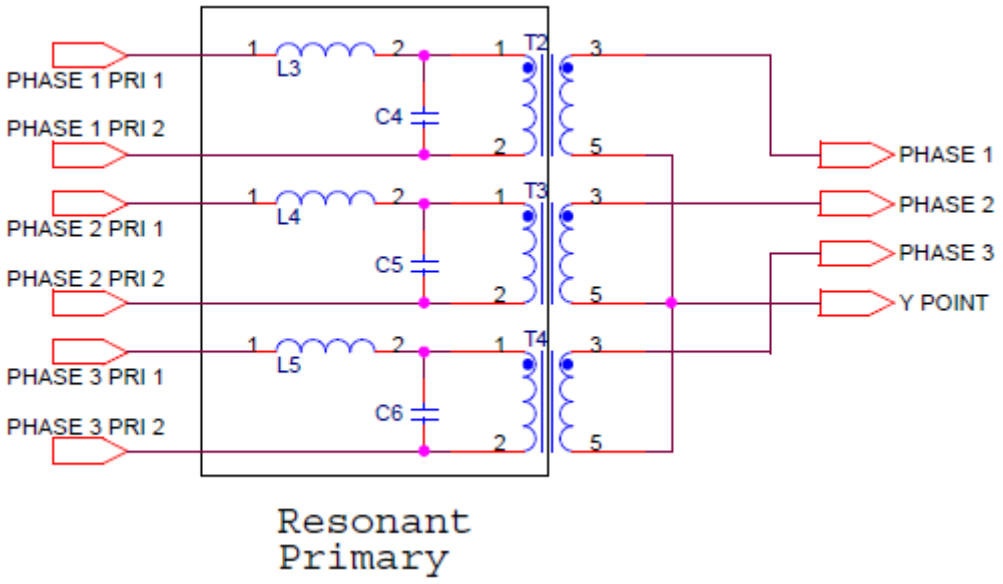


Figure 1.7 Resonant primary tank circuit

The transformer secondaries are connected in a Y topology, with the Y point floating, and are connected to a three phase rectifier. Output voltage is controlled by adjusting both the phase between the primary voltage and current as well as the duty cycle of the input waveform. In order to minimize switching losses, soft switching is obtained by using ZCS during IGBT turn on and ZVS during IGBT turn off. ZCS at the leading edge is ensured by phase shifting the drive waveform so the IGBT turns on during current zero cross, while output voltage control is established by PWM of the width of the pulse. Input dc link capacitor bank voltage is adjusted so that the lagging edge of the IGBT pulse occurs when the primary current is being commuted by the antiparallel diode and zero switching occurs as shown in Figure 1.8.

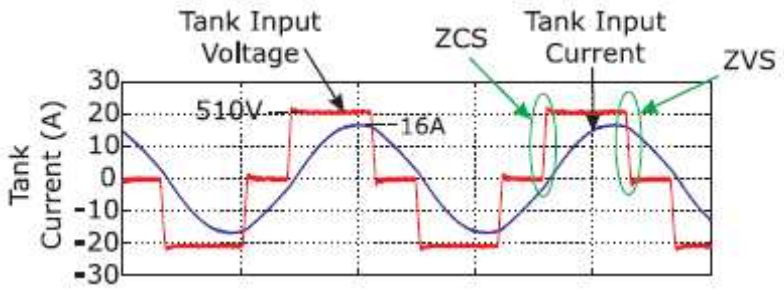


Figure 1.8 IGBT current and tank voltage[9].

The output of the rectifier feeds an LC lowpass filter as shown in Figure 1.9 that acts both to reduce harmonic noise and ripple but also as the output DC link storage capacitor. The klystron load is connected to the supply by a coaxial transmission line.

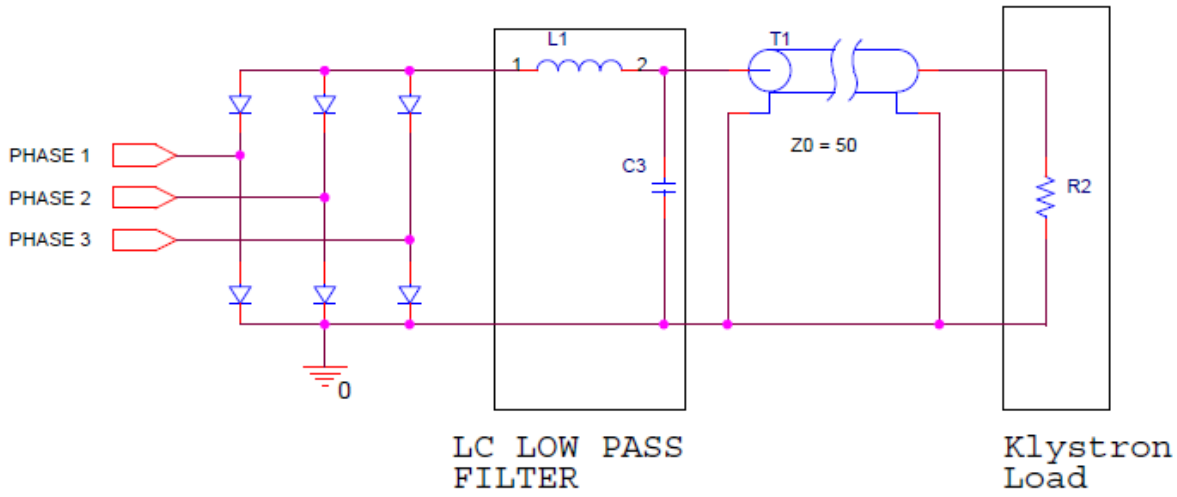


Figure 1.9 Output LC filter.

Research at E2V technologies has shown that klystron cathode damage can occur if deposited energy during an arc exceeds approximately 20J [9]. A crowbar circuit has been connected across the power supply output to shunt stored energy in the LC filter's capacitor in the event of a tube arc. An unfortunate side effect of using an LC filter is that the filter's capacitor is directly connected in parallel across the klystron tube with minimal series

inductance between the capacitor and the load, thereby increasing the fraction of energy that can be dissipated into the tube before the crowbar circuit fires.

1.4 Harmonic Mitigation

Reduction of switching noise and ripple in the output waveform is an important consideration when driving klystron tubes, since the tube's gain is a function of input voltage and current: such fluctuations in input voltage will result in fluctuation in RF gain generating a noisy output. Several methods of harmonic mitigation may be implemented to reduce noise and output ripple in an SMPS. The main source of harmonic noise in a three phase SMPS is the 6^{th} harmonic of the switching frequency, generated by the three phase full wave rectifier. Elimination of the 6^{th} harmonic will greatly reduce output voltage ripple. Further it is important to insure that the amplitudes of the secondary voltages are equal of the system will produce 1^{st} and 2^{nd} harmonic ripple as well [4].

The most basic methods of reducing ripple involves increasing parallel capacitance on the output DC link, however this introduces many undesirable characteristics to a SMPS driving a klystron that requires rapid output pulses. The increased capacitance provides more stored energy on the output bus that may damage the klystron in the event of an output arc [9]. Further the increased capacitance increases the rise and fall time of the pulse, wasting power and generating unnecessary heating of the klystron collector.

Another basic form of ripple control is an LC low pass filter [8]. Such filters rapidly reduce harmonic noise above their cutoff frequency, however to provide sufficient ripple reduction, the size of the parallel capacitor would increase rise time or may damage a klystron tube in the event of an internal arc, thereby requiring the use of an output crowbar circuit.

A shunt LC harmonic filter may be connected across the output and tuned to selectively filter the 6^{th} harmonic, however this filter losses effectiveness away from its resonant frequency, potently increasing noise if frequency is varies to stabilize output voltage. If the input capacitor bank is sufficient large, the switching frequency range may be placed near to the filter resonance at all times allowing the filter to reduce 6^{th} harmonic noise effectively.

Reduction in harmonics and ripples was obtained by utilizing a pi-R filter in the output, constructed by placing a 6 ohm resistor in series with the output of the rectifier stack and an LC pi filter on the output. Pi-R filters have been shown to greatly reduce output ripple while allowing fast rise times [3]. An additional benefit of pi-R filters is the ability to limit dI/dt during an output arc [7] thereby protecting the klystron from damage and increasing the probability that the fault will self-clear before the resonator is de-Qed.

1.5 Summary

Techniques for designing high voltage, high power resonant SMPS have been presented in this chapter. The first section presented several designs of high power SMPS, and high voltage SMPS topologies. It was generally concluded most high voltage high power systems utilize either closely coupled transformers operating at 60Hz or loosely coupled resonant transformers with nano-crystalline cores operating at frequencies around 20kHz. The higher operating frequency allows a considerably smaller power conversion system that operates at higher efficiencies due to reduced core losses. The utilization of a resonant transformer system allows a boost ratio higher than the turns ratio, and the capability to utilize soft switching to greatly reduce switching losses.

The second section presented designs and characteristics of loosely coupled transformers, different varieties of primary and secondary compensation, and methods of increasing power transfer.

The third section covers an in depth review of resonant SMPS power converters, including a designs of three phase resonant converters built at Los Alamos National Lab and E2V technologies, and results of their testing.

The fourth section presented techniques of harmonic mitigation, their benefits and drawbacks. It was concluded that increasing parallel output capacitance is not a feasible option due to the decreased rise time and potential to damage attached klystron loads in the event of an arc fault. Ideal candidates for harmonic reduction include pi-R filters and LC harmonic filters due to their low energy storage.

Chapter 2 Power Supply Design Overview

This chapter presents an overview of the design requirements and components available to construct the power supply presented herein. The first section presents the requirements and design constraints for construction of the power supply. Requirements on the power supply include output voltage and current capabilities, fault tolerance, output voltage range, output stability, pulse duration, serviceability, safety, tolerance for voltage droop on the capacitor bank, and external control of parameters. Design constraints include the use of certain transformer cores, IGBTs, rectifier diode stacks, and capacitor banks which were either donated to the project or were available as surplus. The second section presents a summary of the chosen design and the reasons certain features were selected.

2.1 Design Requirements and Constraints

The power supply presented in this thesis is designed to power a klystron amplifier for a fusion power research experiment. The klystron in question requires a cathode potential of -75kV and draws 40A of current, yielding a nominal impedance of 1875ohms. The klystron presents to a good approximation, a constant, purely resistive load with current linear to voltage within the rated operating range. Due to space constraints within the experimental area the power supply is required to have a compact design and be located approximately 30ft from the klystron tube which it powers. For safety the power supply and associated capacitor banks must be located in a caged area within the experimental area; however the system's output must be variable necessitating remote control over a computer terminal. To be placed into the requisite location in the engineering bay, the power supply must be lifted into position using a ceiling crane, favoring a light weight, modular design.

Due to the presence of high strength pulsed magnetic fields in the area, the power supply must not generate any ground loops when electrically connected to the experiment, requiring that all connections maintain galvanic isolation. Due to this requirement, all i/o lines connecting the power supply to the control system must be fiber optic, the secondary side of the power supply must get its ground from the klystron tube, and all voltage and current sensors must be galvanically isolated from the control system.

The klystron tube being powered is designed for sub-millisecond pulses, however for this experiment; the tube will be required to produce a 10ms pulse, in excess of its design specifications. Although this is permissible when using short duty cycles, it is probable that

the tube will occasionally arc internally. The stored energy in the power supply secondary must be sufficiently small so that no damage will occur to the tube. Further the power supply must be able to tolerate an arc fault that reflects a high voltage pulse back up the transmission line without damage. As the klystron gain is sensitive to voltage fluctuations, output ripple and harmonics must be minimized.

Due to budget constraints, a number of pats on the power supply were recycled from previous experiments, or donated to the project by LANL. The main capacitor bank is comprised of 450V electrolytic capacitors, each with capacitances of 1.8mF, 2mF or 2.4mF restricting the bank voltages to multiples of 450V. The capacitors were mounted in racks, with each rack containing 8 trays, with each tray containing 35 electrolytic capacitors. The bank has a total capacitance of 0.3F when configured in a 900V arraignment. Due to the configuration of the bus bars connecting the trays, it is easiest to configure the bank in the 900V configuration. A hipotronics high voltage power supply was obtained to charge the dc link input capacitor bank, capable of charging the bank to 900V.

A set of 20 matching Mitsubishi CM1200HB-66H IGBTs was donated, each rated at 3.3kV and 1.2kA with a rated pulse current of 2.4kA [A], of these 12 are used to construct a set of 3 full h-bridges. In addition, a number of 50kV 0.05uF Mylar foil capacitors, four low ESL IGBT bypass capacitors rated at 4kV, 10uF and 10nH ESL, three nano-crystalline iron cores with a length and width of 9.5” by 14” with a cross section of 2.5” by 1.75”, and a three phase full wave rectifier assembly were also donated.

2.2 Summary

This chapter presented the available parts, requirements, and constraints on the power supply described herein. These components and requirements formed the basis of the power supply design and governed the physical layout, dimensions and electrical parameters chosen. An electrolytic capacitor bank powered three phase resonant topology with independently driven primaries and Y connected parallel resonant secondaries was chosen. The secondaries are connected to a doubling three phase rectifier, with a pi and 6^{th} harmonic shunt LC filters to reduce output ripple. The klystron load is connected to the power supply through RG-8/U coaxial cable and has an LR snubber in series and a triggerable spark gap snubber in parallel. The system is controlled with a fiber optically coupled microcontroller with galvanically isolated analog inputs for feedback control and fiber optic communication to the control computer.

Chapter 3 Power Supply Design

This chapter presents the design of the resonant power supply including power electronics, magnetics, filtering, crowbar and snubber. The first section presents an overview and block diagram of the power supply hardware. The second section presents the connection and control of the capacitor charging power supply. The third section presents the design and construction of a smaller short pulse capacitor bank for safely testing the power supply and conditioning the klystron tube. The fourth section describes the design and construction of the long pulse capacitor bank including the electrolytic capacitors, capacitor fuses and connection to the bus plates, and the interconnection to the power supply. The fifth section presents the design of the IGBT switching network, its topology, the connection from the capacitor bank, the low ESL stiffening capacitors, a low inductance switching relay designed to disconnect the capacitor bank from the IGBT network, the low inductance bus plates and their insulation, the IGBTs and their gate drivers, and the connection to the resonant transformer primaries. The sixth section provide an overview of the transformer design, its nano-crystalline iron core, the design of its windings, the enclosure and insulation of its secondary winding within dedicated oil tanks, the feedthrough and cable connections to the transformers, the connection of the parallel resonant capacitors, and testing of the transformer's transfer function. The seventh section presents an overview of the doubling three phase rectifier configuration. The eight section presents filter networks for harmonic

mitigation. The ninth section presents safety systems to protect the klystron load including an LR snubber and crowbar spark gap. The tenth section summarizes the chapter.

3.1 Overview and Block Diagram

The power supply is arraigned in an easily serviceable, modular design, consisting of a charging supply, a capacitor bank, three h-bridges, three resonant transformers, a doubling three phase rectifier, an output filter and a control system as shown in Figure 3.1.

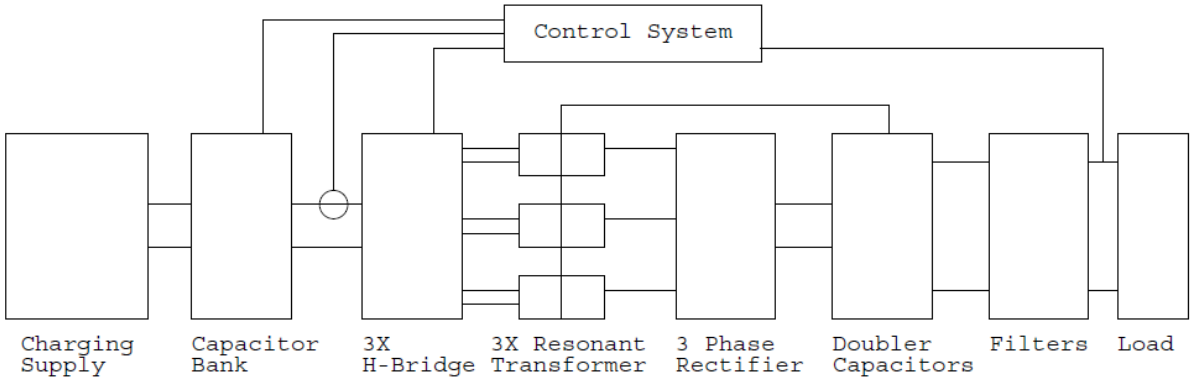


Figure 3.1 Power Supply Block Diagram.

The capacitor bank charging supply charges the capacitor bank to 900V between pulses, ensuring acceptable voltage bounds during the pulse. The input DC link capacitor bank is mounted in three racks, configured as a 900V, 0.3F bank with each electrolytic capacitor connecting to the bank's bus plates using a stainless wire fuse that will open in the event of an internal short. The bank connects to the power supply using twisted pair wires which plug into the low inductance bus plates holding the IGBTs. The bus plate is connected directly to the low ESL capacitors, providing the IGBTs a stiff DC voltage source on the poles of the h-bridges. The bus plate holds three full h-bridges, each driving the primary of a resonant transformer. The primaries of the transformers are electrically independent and the cores are magnetically independent. The secondaries are designed with high leakage inductance, a

parallel resonator capacitor and are connected in a Y topology to a doubling three phase rectifier. The output of the supply is optionally filtered by a pi and harmonic filter to reduce ripple and harmonic noise. The klystron load is connected to the supply by a 20-30ft long RG-8U coaxial cable. An RL snubber is placed in series with the klystron to protect the tube in the event of an internal arc and prevent HV transients from being reflected down the transmission line. A triggerable spark gap is placed in parallel with the klystron to crowbar the output in the event of an internal arc. The power supply is controlled by a microcontroller based control system that varies switching frequency to stabilize voltage output. The microcontroller varies switching frequency as a function of capacitor bank and output voltage. A picture of the power supply is presented in Figure 3.2.

Figure 3.2 Resonant power supply in construction.

3.2 Capacitor Charging Power Supply

The capacitor bank charging supply charges the capacitor bank to 900V between pulses, which will be taken at very low duty cycle; one 10ms pulse approximately every 2-4 minutes. Due to the low duty cycle, the charge rate may be very slow, and the use of a linear power supply using hysteresis control is acceptable. A hipotronics 805-1A power supply as shown in Figure 3.3 capable of 1A output in the kV range will be used to charge the capacitor bank.

Figure 3.3 Hipotronics power supply.

The power supply is capable of being run directly from the AC mains and contains an internal control system to regulate output voltage. The power supply features operation off of

a 208/230VAC three phase input, 10% regulation and under 5% ripple [B]. Given that the power supply's voltage rating exceeds the voltage required on the bank, and the supply has current limiting, constant current charging may be utilized. The time to charge the capacitor bank is given by (3.1).

$$T = \frac{CV}{I} \tag{3.1}$$

For a 900V, 0.3F capacitor bank and 1A charging current, the power supply will be able to charge the bank from 0V to full voltage within 270s or 4.5 minutes. During cycling, the voltage droop on the bank will be a small fraction of the bank voltage allowing faster charge times between shots.

3.3 Testing Capacitor Bank

In order to safely test the power supply during development, it was necessary to construct a smaller, lower capacitor bank to minimize stored energy. A testing bank consisting of 24 450V, 6.2mF capacitors configured into a 900V, 37mF bank as shown in Figure 3.4.

Figure 3.4 Testing capacitor bank.

The bank is charged by an internal rectified microwave oven transformer, with bank voltage being controlled by a hysteresis controller. The controller uses a comparator circuit to turn on the microwave oven transformer if bank voltage is below the preset voltage value. The system is designed to have 10V hysteresis.

The capacitors are interconnected with copper bus bars and the bank is connected to the power supply with four 10 gauge wires twisted together to reduce inductance. The

voltage droop on the capacitor bank may be solved for from the dissipated pulse energy, stored bank energy and power supply efficiency as seen in equation (3.2).

$$V_f = V_i \sqrt{1 - \frac{E_{pulse}}{\eta E_{bank,i}}} = V_i \sqrt{1 - \frac{V_{out} I_{out} T_{pulse}}{\eta (1/2) C V_i^2}} \tag{3.2}$$

Assuming output voltage and current are constant due to feedback control, and that the rise time is negligible, the voltage droop on the capacitor bank can be plotted vs pulse time, as shown in Figure 3.5. See matlab code in [G];

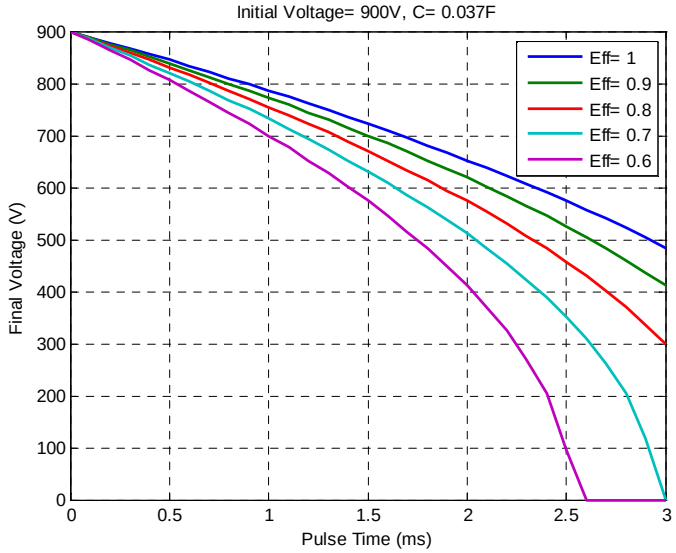


Figure 3.5 Testing bank voltage droop.

3.4 Long Pulse Capacitor Bank

An increased capacitance will be required to extend the pulse duration to 10ms and beyond in order to reduce the voltage droop during the pulse to acceptable levels. The main capacitor bank consists of 450V electrolytic capacitors, each with capacitances of 1.8mF, 2mF or 2.4mF. The capacitors are mounted in racks, with each rack containing 8 trays, with each tray containing 35 electrolytic capacitors. The bank has a total capacitance of 0.3F when configured in a 900V arraignment. Each tray is connected to a set of bus bars on the back of the rack, and each capacitor is connected to the trays through a stainless wire fuse to prevent the bank charge from being shunted through a single capacitor in the event of an internal short. Assuming output voltage and current are constant due to feedback control, and that the rise time is negligible, the voltage droop on the capacitor bank can be plotted vs pulse time, as shown in Figure 3.6.

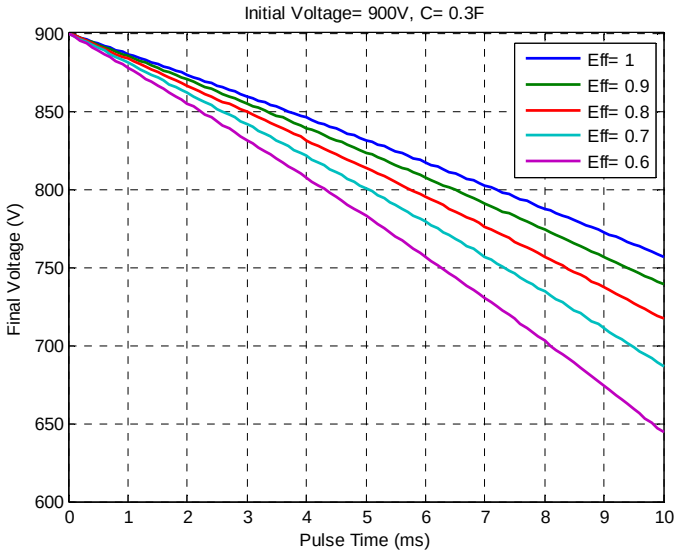


Figure 3.6 Main bank voltage droop.

3.4.1 Electrolytic Capacitors

The main capacitor bank consists of 450V electrolytic capacitors, each with capacitance of 1.8mF, 2mF or 2.4mF as shown in Figure 3.7. These capacitors were arraigned into trays of 35 capacitors in three racks, with two racks containing all capacitors connected in parallel and the third rack containing two electrically separate banks. During construction it was ensured that each of the two larger banks and the two smaller banks has equal capacitance. The two half rack banks are placed in parallel with the two full rack banks, and then placed in series with each other forming a 900V 0.3F capacitor bank.

Figure 3.7 Main bank capacitors.

3.4.2 Capacitor Fuses

Due to the possibility of a capacitor failure and the parallel connection of the capacitors in the bank it is necessary to individually fuse each capacitor such that in the event of a

capacitor developing an internal short, the energy of the entire bank is not shunted through the failed capacitor.

Each capacitor is connected to an aluminum tray with one terminal and a bus bar above the tray through the wire fuse. The capacitor fuses consist of a stainless connecting wire between each capacitor and the bus bars on each tray that will melt open in the event of an internal short, as shown in Figure 3.8.

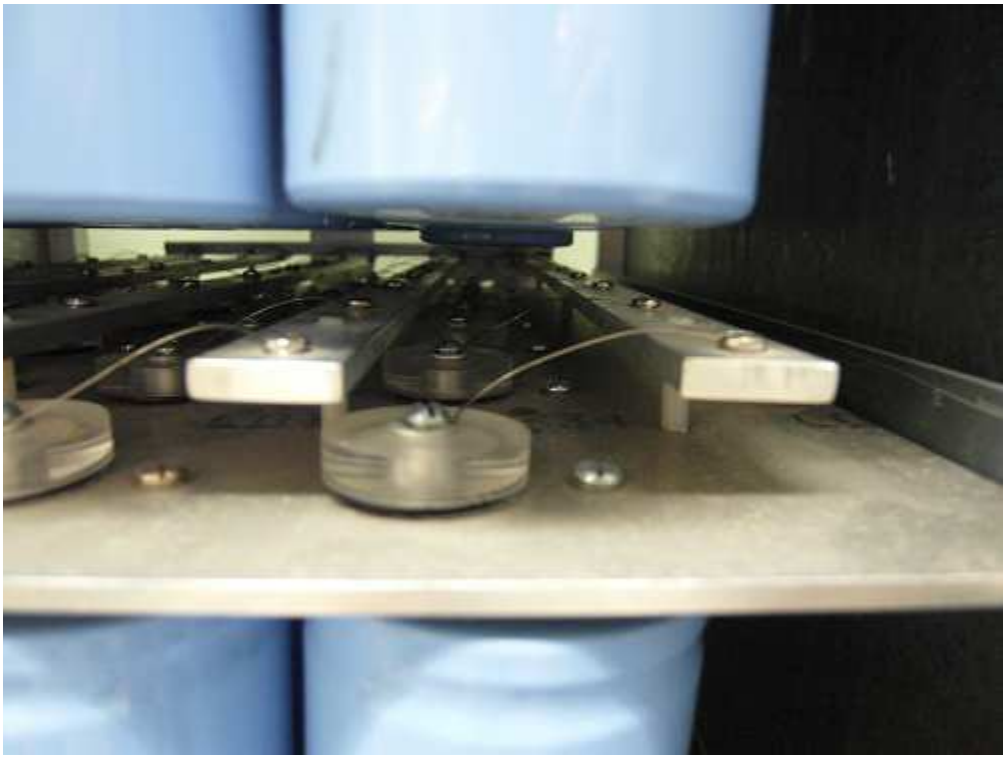


Figure 3.8 Capacitor bank fuses.

3.4.3 Capacitor Bank Bus Plates

Each capacitor tray is connected to a set of vertical bus bars in the rear of the rack as shown in Figure 3.9 allowing the trays to be electrically connected in parallel. One bus bar is connected to the base of the aluminum trays, while the other vertical bus bar connects to the

bus bars on each tray. Each bus bar is connected to low gauge interconnect cable that connects the capacitor bank racks to each other and to the power supply.

Figure 3.9 Capacitor bank bus bars.

3.4.4 Capacitor Bank Interconnect Cables

The capacitor banks are connected to the power supply's low inductance buss plate over a set of four twisted wires in order to minimize inductance while allowing flexibility in the location of the banks with respect to the power supply. The effects of any inductance contributed by the interconnect cables are minimized by the DC link stiffening capacitors on the power supply.

3.5 Switching Network Design

The switching network for this power supply consists of 12 Mitsubishi CM1200HB-66H IGBTs arraigned in three independent h-bridges, each riving the primary of a resonant transformer. The use of a low inductance bus plate to mount the IGBTs and the use of low ESL DC link stiffening capacitors allows snubberless operation near the transformer's resonant frequency

3.5.1 Topology

The IGBTs on the bus plate are arraigned in a full h-bridge configuration with each transformer being independently driven as shown in Figure 3.10.

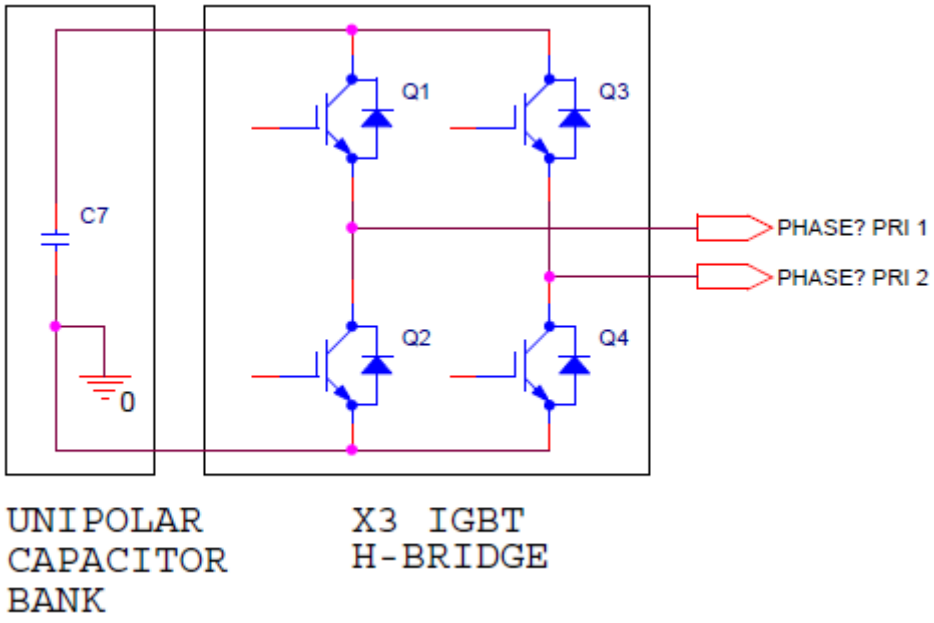


Figure 3.10 H-Bridge Topology.

Each IGBT has an antiparallel diode capable of commuting current during zero voltage switching during certain operating modes. Due to the low inductance of the bus plates that the IGBTs are mounted on and the presence of the DC link stiffening capacitors, snubberless operation is possible without ringing or high voltage spikes during switching events.

3.5.2 Cable Interconnects From Capacitor Bank

Copper blocks located at the terminals of the low inductance bus plate allow connection of the capacitor bank cables to the power supply as shown in Figure 3.11.

Figure 3.11 Capacitor bank interconnects.

The copper blocks have holes drilled to match the oversized banana plugs on the end of the cables. The testing bank connects with two wires per terminal while the main bank connects with eight as shown in Figure 3.12.

Figure 3.12 Main capacitor bank interconnect block.

3.5.3 Low ESL Stiffening Capacitor Bank

Transition between the higher inductance electrolytic capacitor bank and the low inductance buss plates on the power supply requires the use of low ESL stiffening capacitors on the input DC link. A set of four General Atomics 37547 low ESL capacitors rated at 4kVDC, 10uF, and 10nH previously used at LANL were used as bypass capacitors [10]. The capacitors are each rated to handle high transient currents of up to 4kA, and high rms currents of 100+ amps. The capacitors are connected directly to the ends of the low inductance buss plates. In addition, a small 900V, 2mF capacitor bank is added to the buss plates to further increase capacitance as shown in Figure 3.13

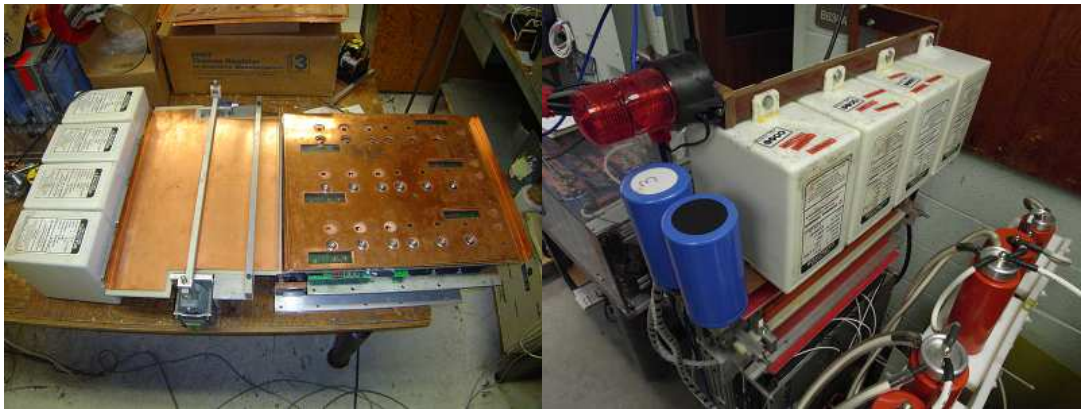


Figure 3.13 IGBT bypass capacitor connection.

3.5.4 Low Inductance Relay

A low inductance relay has been designed to serve as a master connect/disconnect switch capable of carrying the full current while maintaining a low inductance current path. The relay consists of a pair of 16.5” wide bus plates separated by a 1/16” of insulating G10-FR4 fiberglass. The top bus plate consists of a split input and output segment that can be connected and disconnected by physically pulling the overlapping segments of the busbar together as shown in Figure 3.14.

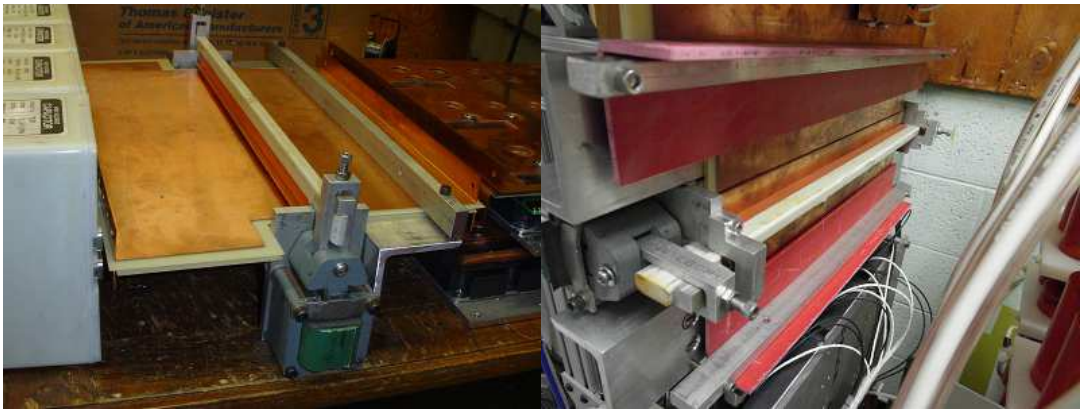


Figure 3.14 Low inductance relay.

The output busbar is rolled into a curved shape, so that with no pressure on it, it will disconnect from the input busbar. A pair of high strength mechanical AC relays connects the circuit by pulling a fiberglass bar against the top bus plate. The fiberglass bar is machined into a curved shape along its length and jacketed with a silicone boot to compensating for any deflection under stress and providing equal pressure to the busbar along its contact area. In this design the input and output busbars overlap by approximately 0.75” allowing the relay to carry the full power supply current. Dimensions of the relay busbar plates are given in [D].

3.5.5 Low Inductance IGBT Busbars

A voltage stiff source must be provided across the terminals of the IGBTs requiring a low inductance DC link. The IGBT busbars are designed to provide the minimum possible inductance, while allowing high current capability, physical rigidity, capability to connect to both the transformer primaries and low inductance relay with minimal added inductance and a large safety margin on insulation voltage standoff. Dimensions of the busbar plates are given in [E].

The IGBTs are arraigned into two modules, with each module containing three half h-bridges. Each module consists of an aluminum plate onto which the six IGBTs are attached. The IGBT gate drives connect directly to the IGBT modules and the associated wiring is shielded from the input and output busbars with a 1/16” polycarbonate sheet. A set of 0.75” OD, 0.375” ID copper standoffs connect to the output terminals of the IGBTs through the polycarbonate insulating plates and connect to the output busbars which are then bolted down

to the IGBT modules as shown in Figure 3.15. All busbar components are made out ov 1/16” thick oxygen free copper (OFC).

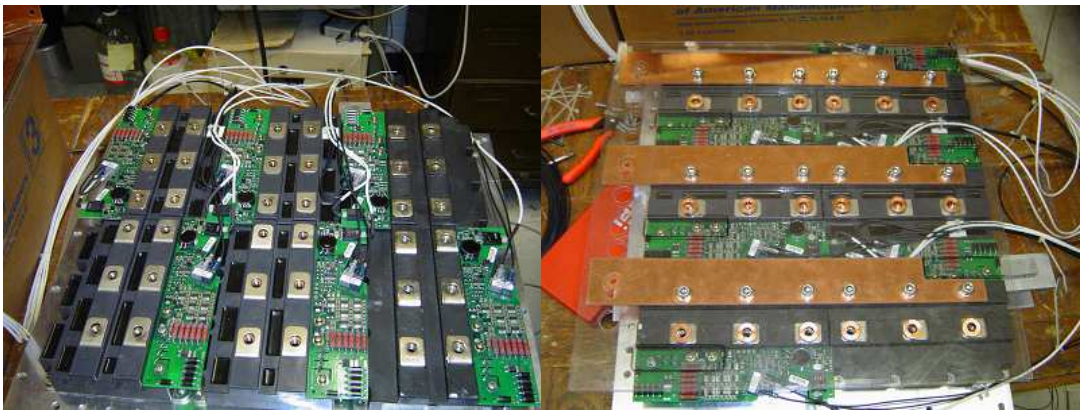


Figure 3.15 IGBT output busbars.

A second layer of 1/16” polycarbonate insulation is placed on top of the output busbars followed by the positive input busbar, which connects to the IGBTs with copper standoffs, followed by a third layer of polycarbonate insulation and the negative busbar as shown in Figure 3.16.

Figure 3.16 Input busbars.

Additional insulation between the busbars is accomplished by placing silicone o-rings between the layers of polycarbonate insulation so that when the busbars are tightened down,

the o-ring is compressed between the insulation layers forming an air tight seal. Additionally, overhang of the insulation layers increases the path between conductors, thereby reducing the possibility of arc tracking as shown in Figure 3.17.

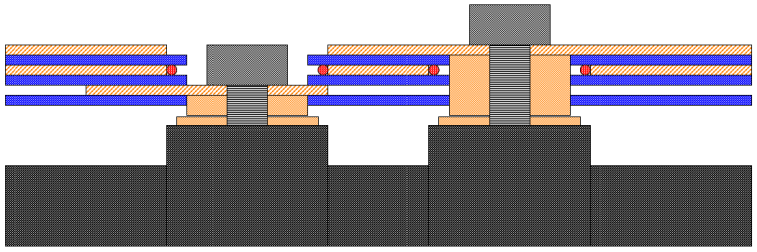


Figure 3.17 Busbar layer cross section.

The two IGBT modules bolt together face to face with the positive busbars connecting with flanges near the output busbar terminals and the negative busbars connecting near the low inductance relay. The two output busbars for a given transformer are now located very close together, allowing a low inductance connection. The completed IGBT module bolts to the low inductance relay with flanged connections, allowing a low inductance connection. The insulation layers of the relay and IGBT module overlay by 0.25” preventing arc tracking at the flange connection as shown in Figure 3.18. All flange connections use pre-stressed aluminum compression bars to provide uniform force across the flange.

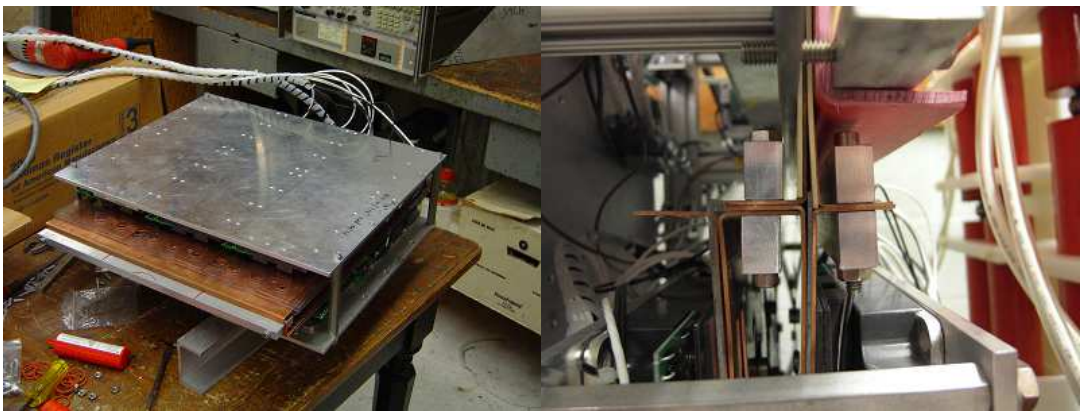


Figure 3.18 IGBT module connection flanges.

3.5.6 IGBTs and Gate Drivers

The IGBTs used to assemble the h-bridges are Mitsubishi CM1200HB-66H modules rated at 3.3kV, 1.2kA continuous current 2.4kA pulsed current as shown in Figure 3.19. Testing at LANL has shown that these IGBTs can exceed the pulsed current rating for short duty cycle pulses used in these resonant power supplies when soft switching is used to limit junction power dissipation.

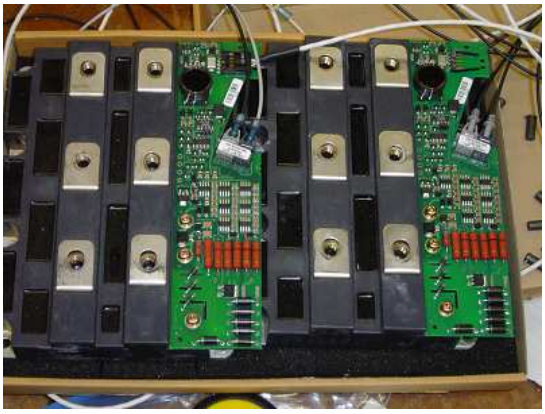


Figure 3.19 Mitsubishi CM1200HB-66H.

The IGBT gate drive signal was provided by a set of 1SD536F2-CM1200HB-66H gate drive modules manufactured by CT concepts. These gate drivers are matched to the electrical

requirements of the IGBT and offer a plug and play solution. Each module can provide +-36A of gate drive current and had 6kV isolation between any IGBT terminal and the DC power connection. Further, control and feedback of these modules is accomplished with a fiber optic interface, allowing galvanic isolation and immunity to electrical noise. The drivers allow the IGBT to achieve 15ns rise time and 20ns fall time. Full specifications are given in [F].

3.5.7 Transformer Connections

A set of intermediate busses connect the IGBT inverter module to the resonant transformers. The transformer bus connections are insulated with a sheet of fiber coated Mylar “whitepaper”, allowing for a low inductance path to be maintained as shown in Figure 3.20.

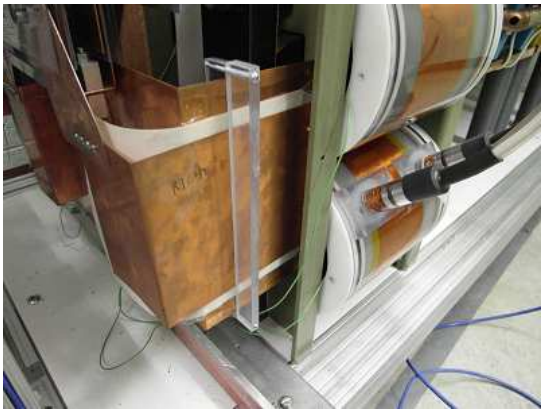


Figure 3.20 Transformer connections.

3.6 Transformer Design

A high power step up resonant transformer system has been designed for use on the power supply. The transformer is configured with electrically independent primaries and resonant secondaries connected in a Y configuration. Each secondary has a dedicated resonator capacitor connected in parallel across the winding as shown in Figure 3.21.

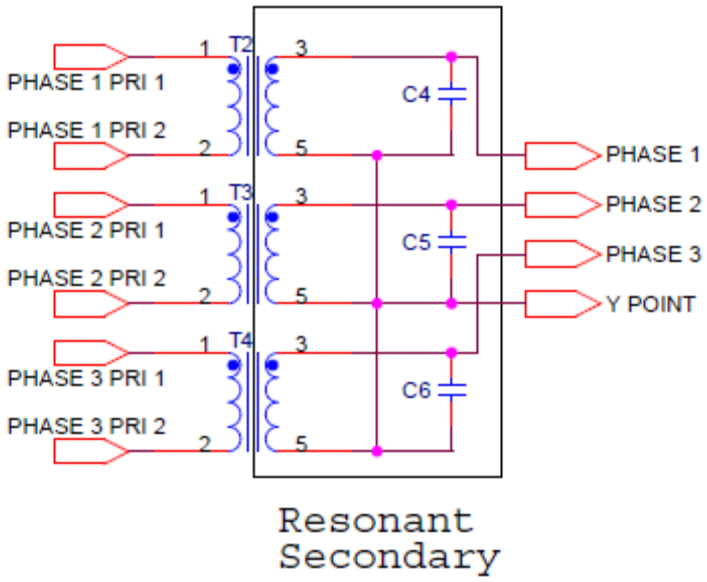


Figure 3.21 Resonant transformer configuration.

3.6.1 Overview

The resonant transformer is designed to provide a sufficient secondary leakage inductance to use in a resonant circuit by designing a loosely coupled secondary winding. The transformer is based on a nano-crystalline iron core that has a large magnetic permeability, thereby providing sufficient volt seconds for high power transfer while allowing low loss at high frequency operation. A tightly wound 10 turn primary is connected

to the IGBT inverter while a loosely wound secondary has enough privatized magnetic flux to provide a sufficiently large leakage inductance. Each secondary winding is enclosed in a dedicated oil tank for insulation and corona suppression. The transformer is pictures in Figure 3.22 and its mechanical design is presented in Figure 3.23.

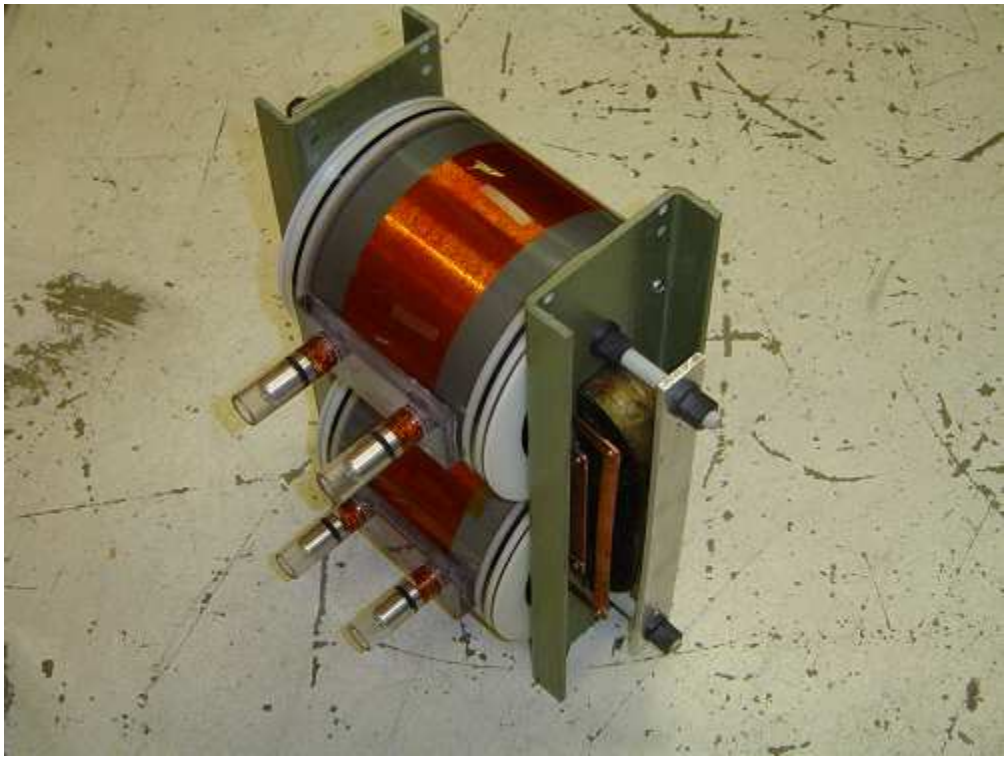


Figure 3.22 Resonant transformer.

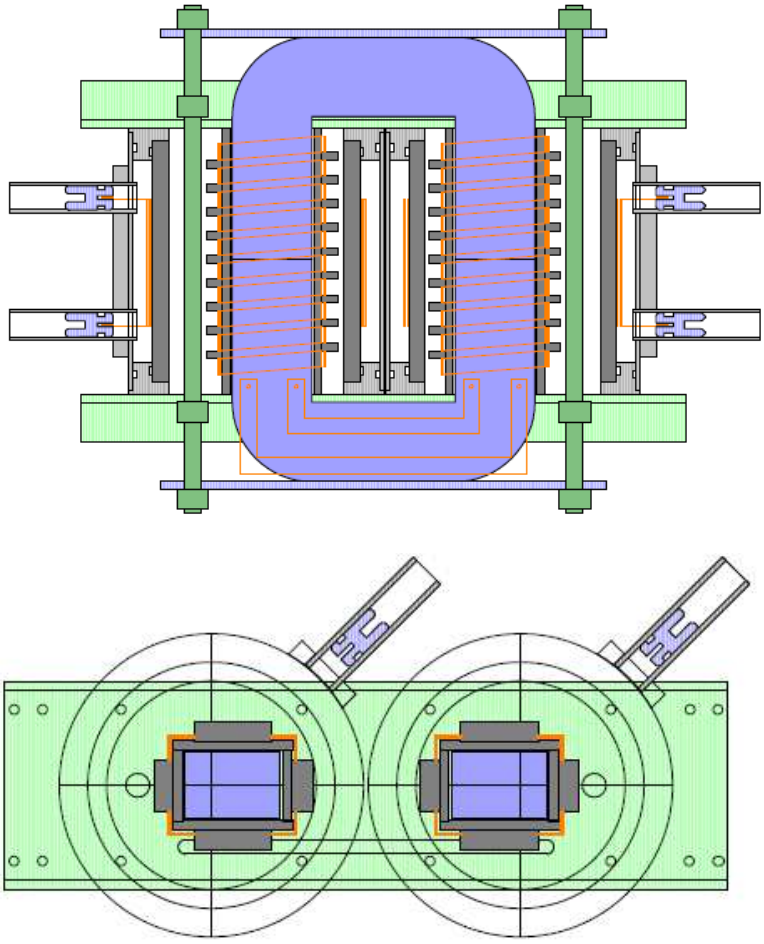


Figure 3.23 Resonant transformer design.

3.6.2 Nano-crystalline Iron Core

A set of three identical nano-crystalline iron cores were donated to the project by LANL. Each core has properties as given in Table 3-1 and dimensions as given in Figure 3.24. Testing determined that the cores saturate at 5.4E-3 Volt Seconds per turn full swing, sufficient for operation at 900V, 20kHz with a ten turn primary, requiring 4.5E-3 Volt Seconds per turn.

Table 3-1 Transformer Core Characteristics [4]

Mu	50,000
Lamination Thickness	.0008"
Lamination Insulation	1 µM Namlite
Stacking Factor	~90%
Bsat	12.3 kG
Current Saturation	5.4E-3 VoltSec/turn

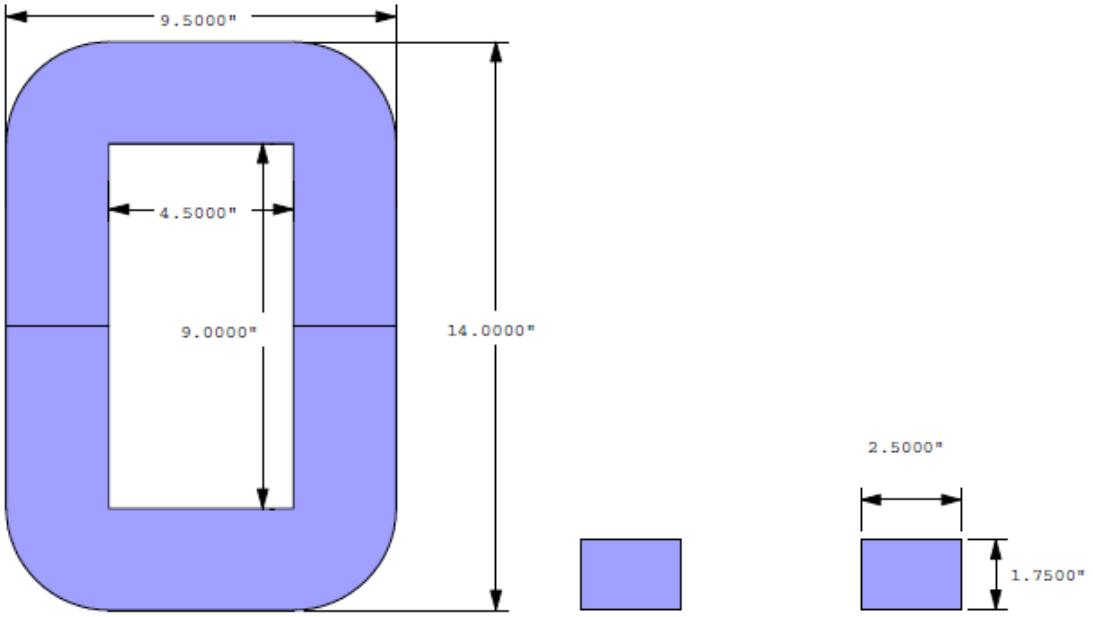


Figure 3.24 Transformer core dimensions.

3.6.3 Primary Winding

The transformer primary consists of a pair of 10 turn windings on each side of the core. Each winding used a 1/32” thick, 0.5” wide copper strap, wound in a helical manner around a 0.25” thick polycarbonate coil form. The primary is pictured in Figure 3.25 and has mechanical dimensions as given in Figure 3.26.

Figure 3.25 Transformer primary.

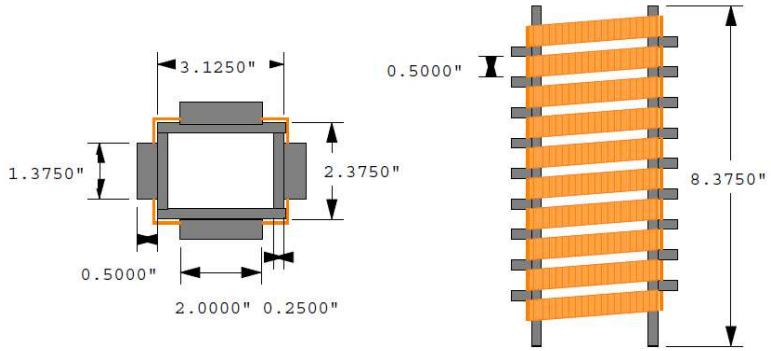


Figure 3.26 Transformer primary dimensions.

3.6.4 Secondary Winding Enclosure

Due to the high frequency, high voltage operation, the secondary is susceptible to corona formation, potentially leading to arcing. To prevent this, each secondary in enclosed in a dedicated oil tank with dimensions as shown in Figure 3.27. The oil tanks may be evacuated during oil filling to prevent air bubbles from being trapped in the windings.

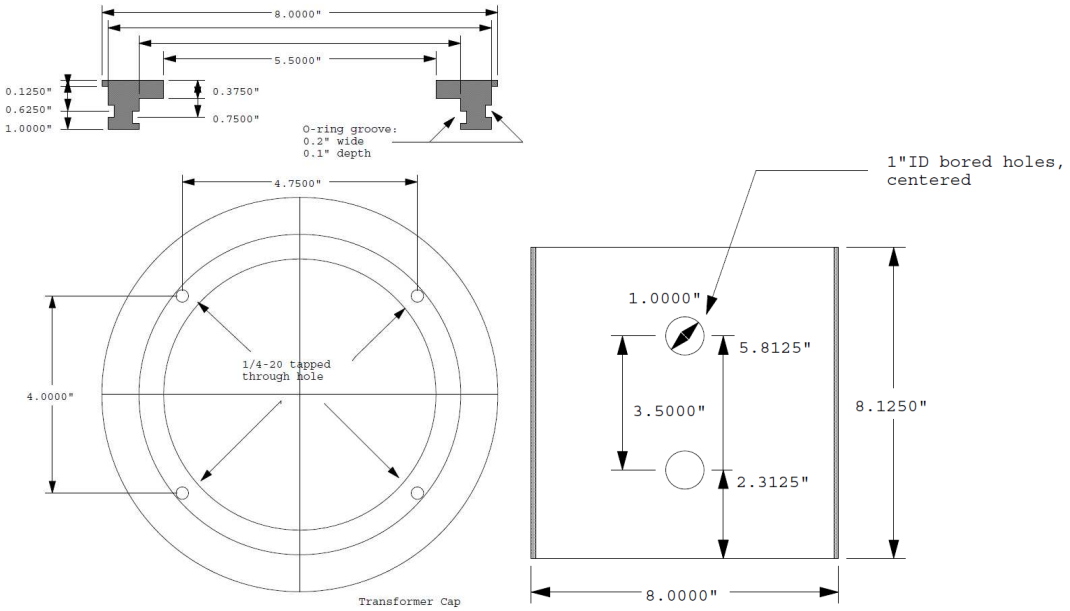


Figure 3.27 Oil tank dimensions

3.6.5 Secondary Winding

Each transformer has two loosely coupled secondaries, connected electrically in parallel. Each secondary winding is wound around a 6.5” diameter PVC coil form and consists of two layers of 22ga copper magnet wire connected electrically in parallel with a layer of Mylar foil insulation between them. Each secondary has 136 turns. The secondary winding dimensions and a picture are shown in Figure 3.28. The interior of the secondaries are lined with several axial lines of conductive copper tape which are then grounded to prevent capacitive coupling through the coil form from generating corona in the interior of the secondary oil tank.

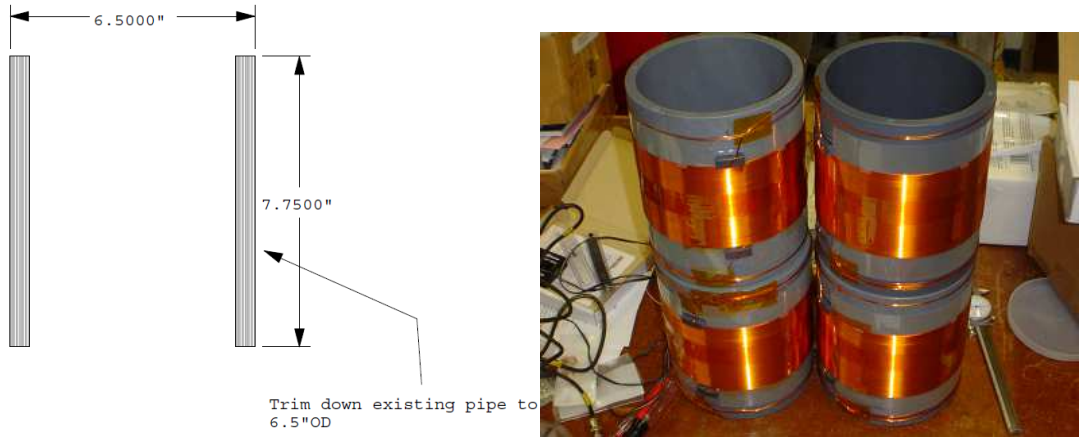


Figure 3.28 Transformer secondaries.

3.6.6 Oil Tank Feedthroughs

A set of high voltage oil tank feedthroughs have been designed to connect the transformer secondaries to the resonator capacitor banks. The feedthroughs must provide adequate high voltage insulation between the two terminals, and other conductive objects in near proximity. The oil tank feedthroughs are shown in Figure 3.29 and a CAD of the assembly is shown in Figure 3.30.

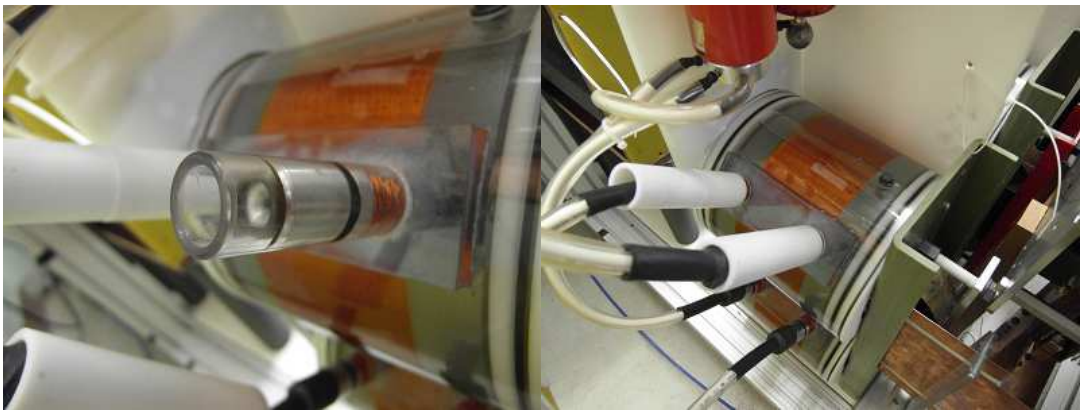


Figure 3.29 Oil tank feedthroughs.

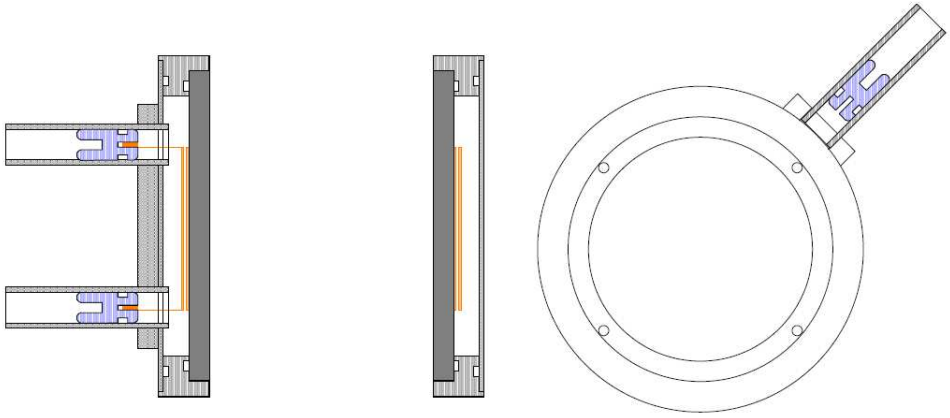


Figure 3.30 Assembled oil tank components.

Each feedthrough consists of a polycarbonate tube housing an aluminum plug as shown in Figure 3.31. The aluminum plug maintains an oil seal to the polycarbonate tube with an o-ring and has a 6-32 thread on the interior for attaching to the secondary winding and a banana plug socket for connecting to the associated cabling.

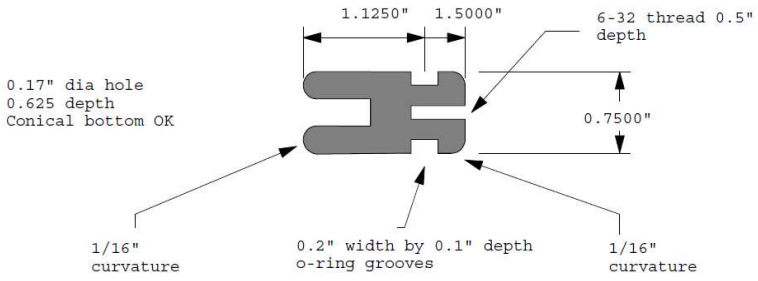


Figure 3.31 Aluminum plug.

The polycarbonate tubes are physically supported by a polycarbonate block with a matching radius to the oil tank that allows the block to be glued to the surface as shown in Figure 3.32. All polycarbonate components are welded together using ethylene dichloride solvent.

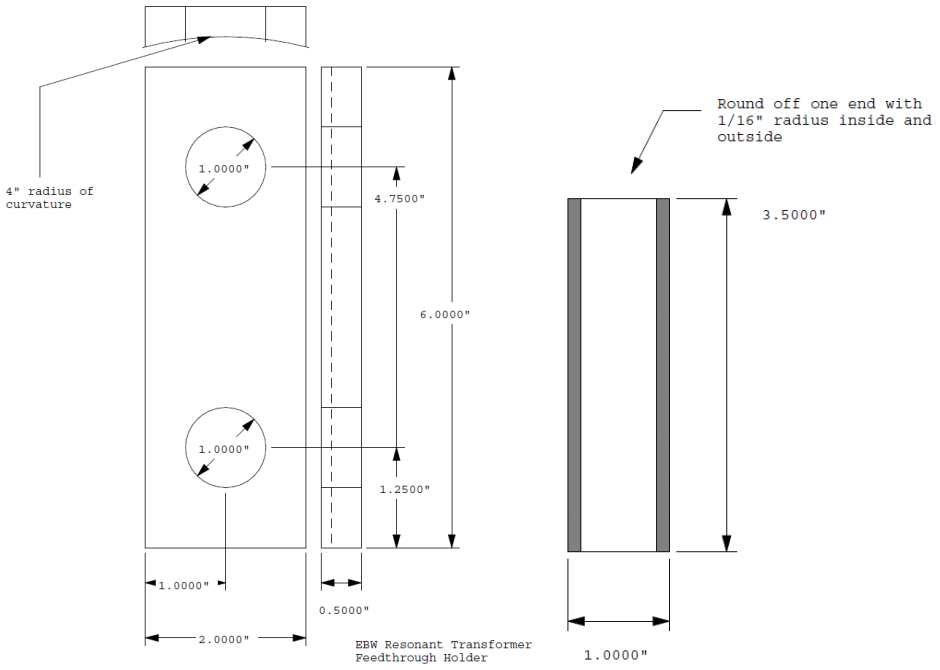


Figure 3.32 Polycarbonate tubes and supports.

The transformers are assembled between a pair of fiberglass support plates which bolt to the secondary oil tanks and hold the cores in place. Dimensions for the support plate are given in Figure 3.33.

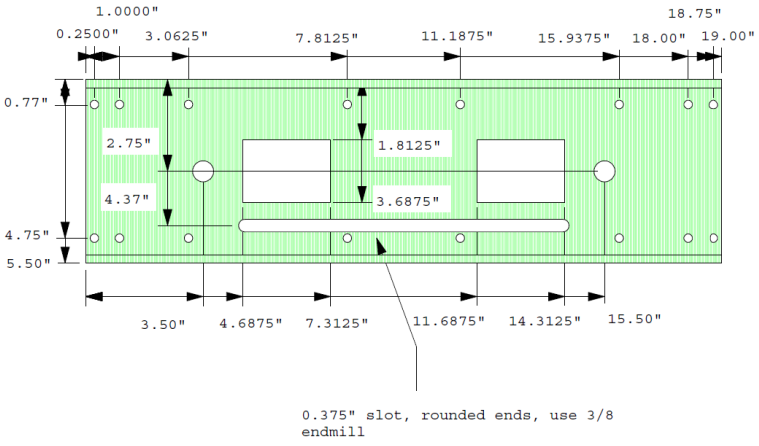


Figure 3.33 Transformer support plate.

3.6.7 Parallel Resonator Capacitor

The secondaries of each transformer are connected in parallel with a 0.05uF resonator capacitor assembly. Each assembly consists of four 50kV, 0.05uF Mylar foil capacitors connected in a series-parallel configuration to obtain a 100kV rating as shown in Figure 3.34. A set of rounded corona disks as shown in Figure 3.35 may be placed on the ends of the capacitors to reduce the risk of arcing.

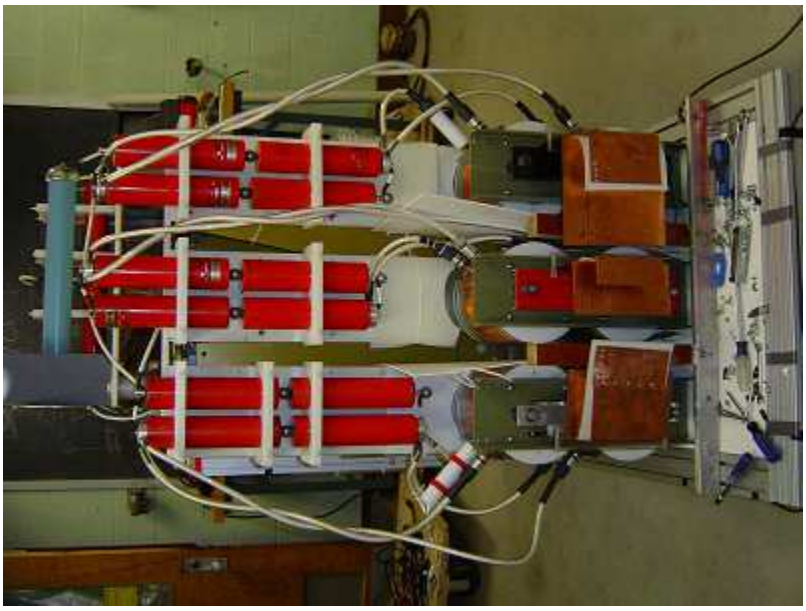


Figure 3.34 Resonator capacitor assembly.

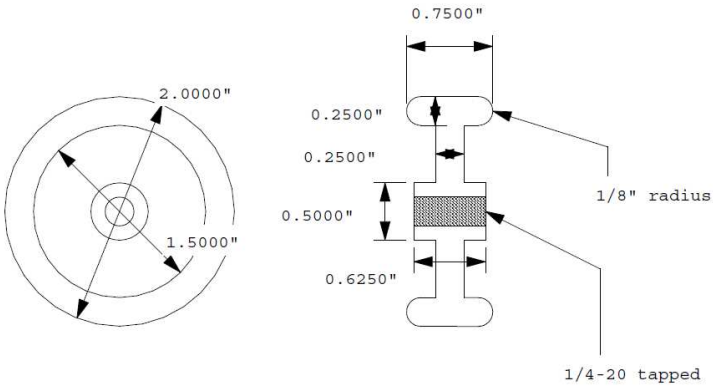


Figure 3.35 Corona disk.

3.6.8 Variation of Frequency, Turns Ratio, and Load Resistance

Several iterations of transformer secondaries were tested during the design process. Secondary turn numbers between 76 and 156 were tested between 1kHz and 30kHz with load resistances between 1ohm and 60kohm. The frequency responses for transformers with a 1.8kohm load are presented in Figure 3.36, and frequency responses for transformers with a 820ohm load are presented in Figure 3.37. Far away from the resonant frequency, the boost ratio equals the turns ratio, while near resonance, the boost ratio is significantly higher than the turns ratio. Plots are generated with the matlab code found in [G].

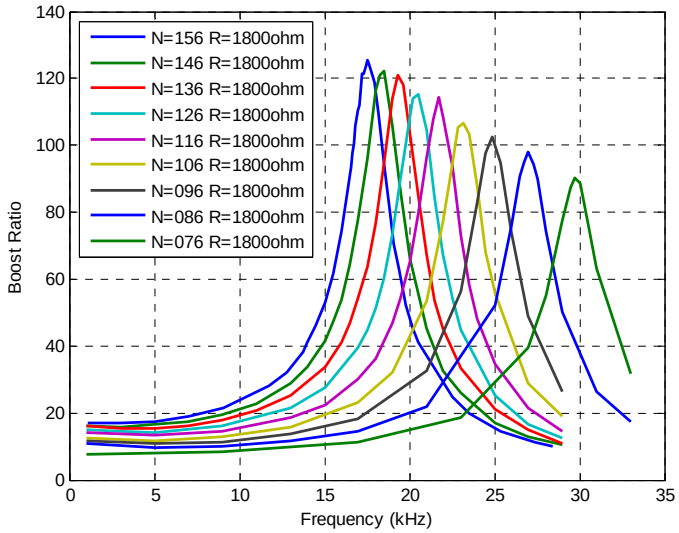


Figure 3.36 1.8kohm frequency responses.

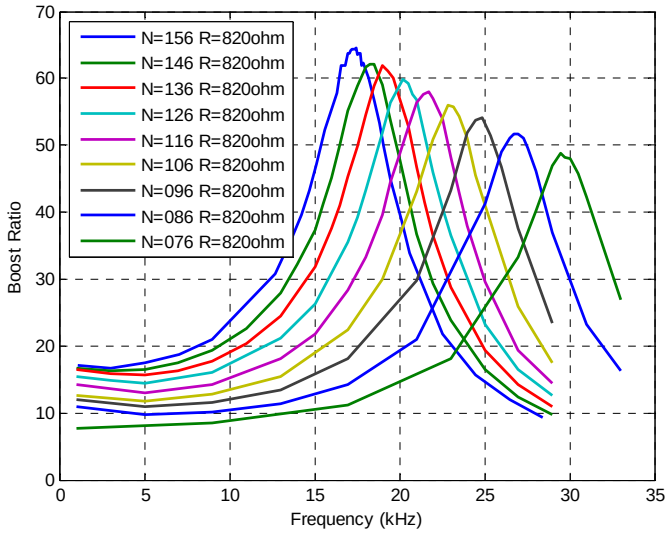


Figure 3.37 820ohm frequency responses.

Several analytical models for leakage inductance of the secondary were compared to measurements from the transformer. These analytical models include the long solenoid approximation (3.3), the long solenoid approximation with area compensation (3.4) where

the area of the magnetic core is subtracted from the area of the secondary, making the approximation that magnetic flux is excluded from the core when measuring leakage inductance with a shorted primary. A second model for the inductance of a short solenoid was derived by wheeler in [11] and [12] is given by (3.5), with the corresponding area compensated version given by (3.6). The length of the coil, h, is given by (3.7) so that the inductance formulas may be evaluated in terms of N.

$$L_{long} = \frac{\mu_0 N^2 A}{h_{coil}} \tag{3.3}$$

$$L_{long} = \frac{\mu_0 N^2 \left(A - A_{core}\right)}{h_{coil}} \tag{3.4}$$

$$L_{wheeler} = \frac{10\mu_0 N^2 A}{\left(9r_{coil} + 10h_{coil}\right)} \tag{3.5}$$

$$L_{wheeler} = \frac{10\mu_0 N^2 \left(A - A_{core}\right)}{\left(9r_{coil} + 10h_{coil}\right)} \tag{3.6}$$

$$h_{coil} = N \bullet D_{wire} \tag{3.7}$$

The evaluation of these formulas is computed in matlab and plotted against measured values of inductance in Figure 3.38. The wheeler formula with area compensation is a nearly an exact match to the measured values of inductance. Note that in the matlab code, the formulas are multiplied by 0.5 since there is a parallel connection of the two separate secondary coils, thus halving the inductance.

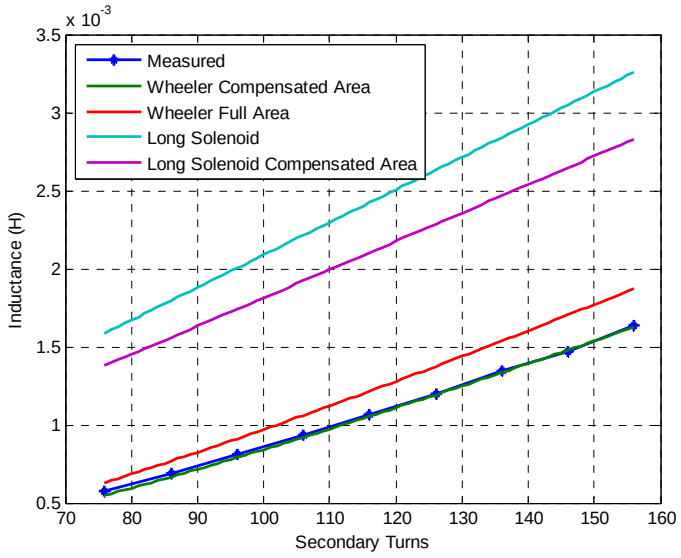


Figure 3.38 Leakage inductance models.

Analytical and numerical models for the transformer's transfer function were derived and compared to measured data. The final version of the transformer was chosen to have 136 turns, resulting in a primary leakage inductance of 8uH, primary magnetizing inductance of 1.69mH, secondary leakage inductance of 1.36mH, and a secondary series resistance of 1.2 ohms. A spice model of the transformer is presented in Figure 3.39.

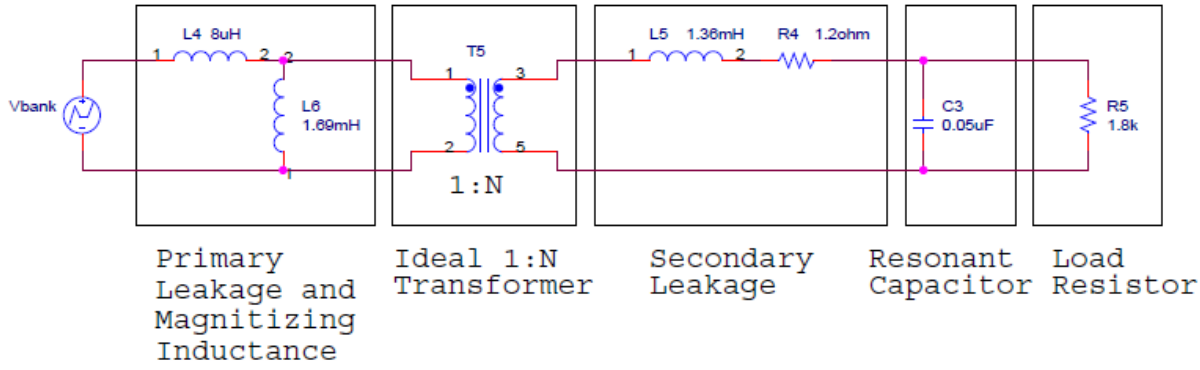


Figure 3.39 Transformer model

A simplified model of the transformer may be obtained by approximating that the magnetizing inductance is infinite, which remains valid as long as the transformers are run without saturating the cores, that primary resistance and leakage inductance is negligibly, and that the cores are magnetically lossless. The system may then be modeled as a resonant LC tank circuit driven by a voltage equal to the primary voltage times the turns ratio as shown in Figure 3.40. The spice simulation of this model is presented in Figure 3.41.

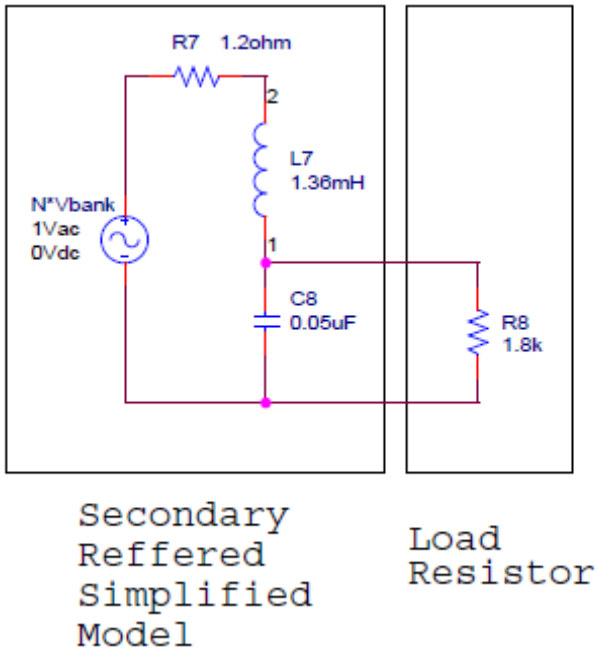


Figure 3.40 Simplified transformer model.

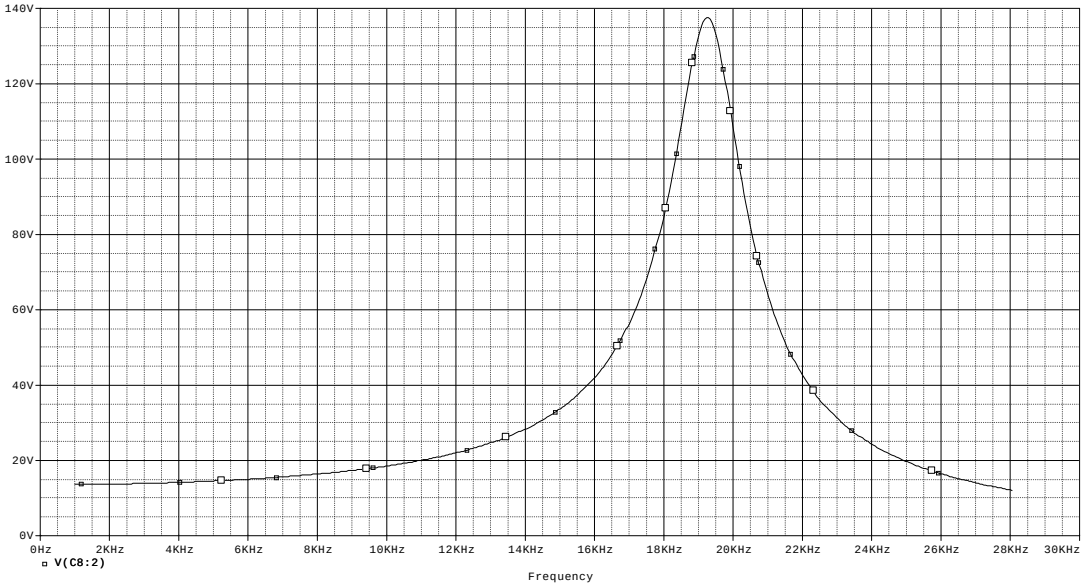


Figure 3.41 Spice simulation of simplified model.

The transfer function for the simplified model may be found analytically, with the total impedance seen by the voltage source given by (3.8) and the voltage gain given by (3.9).

$$Z(\omega) = \left[R_{inductor} + j\omega L + \frac{\frac{R_{load}}{j\omega C}}{R_{load} + \frac{1}{j\omega C}} \right] \tag{3.8}$$

$$G(\omega) = \frac{V_{\sec}}{V_{pri}} = \frac{N}{Z(\omega)} \left[\frac{\frac{R_{load}}{j\omega C}}{R_{load} + \frac{1}{j\omega C}} \right] \tag{3.9}$$

$$G(\omega) = \frac{V_{\sec}}{V_{pri}} = \frac{N}{\left[1 + \left(R_{inductor} + j\omega L \right) \left(\frac{1}{R_{load}} + j\omega C \right) \right]} \tag{3.10}$$

The plot of the transfer function closely matches the measured data as plotted in Figure 3.42 with the only difference being a slightly higher boost ratio at resonance which may be

attributed to the transfer function neglecting losses. This is further illustrated in a plot of the maximum boost ratio at resonance for a given turn number as shown in Figure 3.43 and the boost ratio at resonance for varying load resistance, as shown in Figure 3.44.

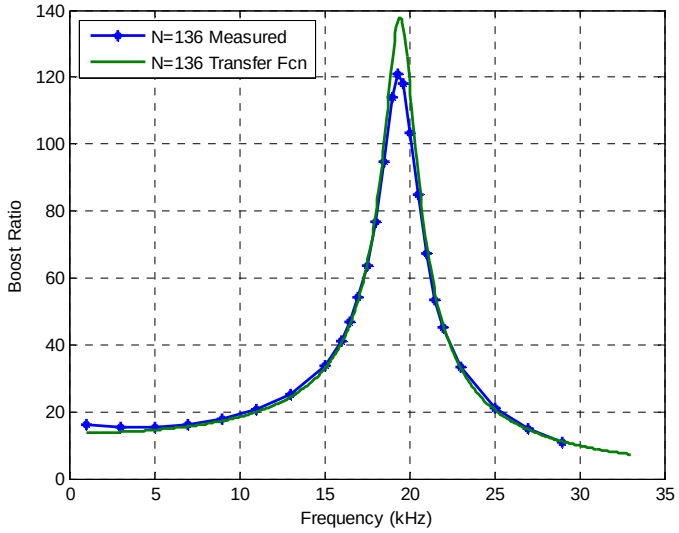


Figure 3.42 Comparison of transfer function and measured data.

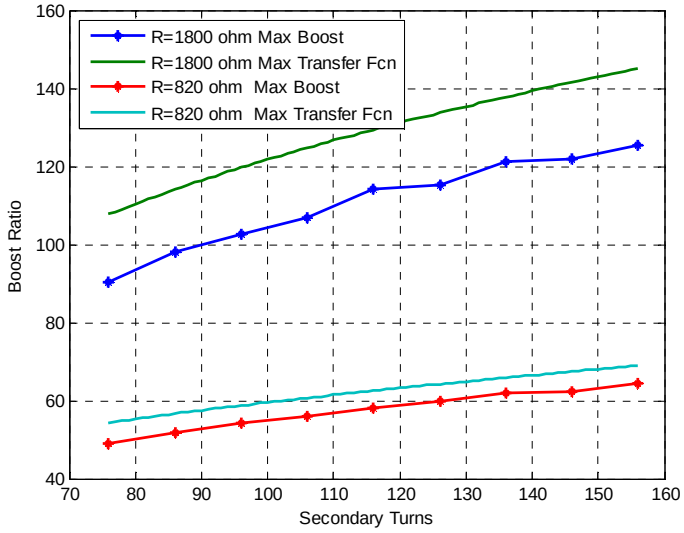


Figure 3.43 Maximum boost ratio vs turns.

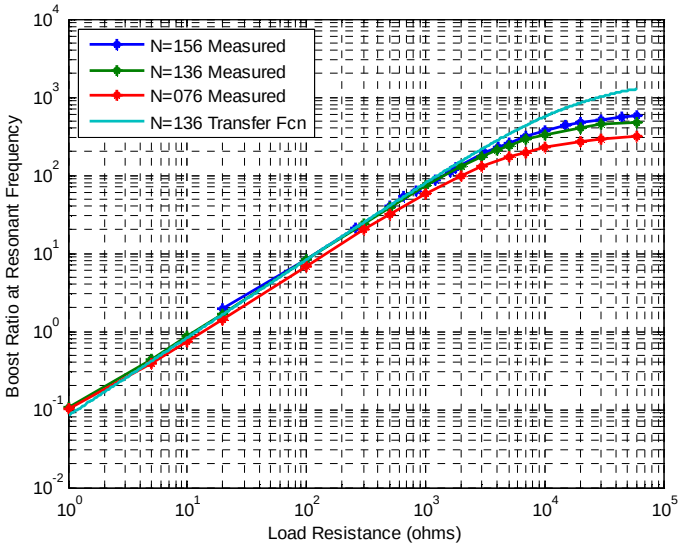


Figure 3.44 Boost ratio at resonance for varying load resistance.

The resonant frequency for the simplified system may be modeled analytically as (3.11). The analytical model matches the measured resonant frequencies with great accuracy as plotted in Figure 3.45.

$$F_{res} = \frac{1}{2\pi}\sqrt{\frac{1}{LC} + \frac{1}{(RC)^2}} \tag{3.11}$$

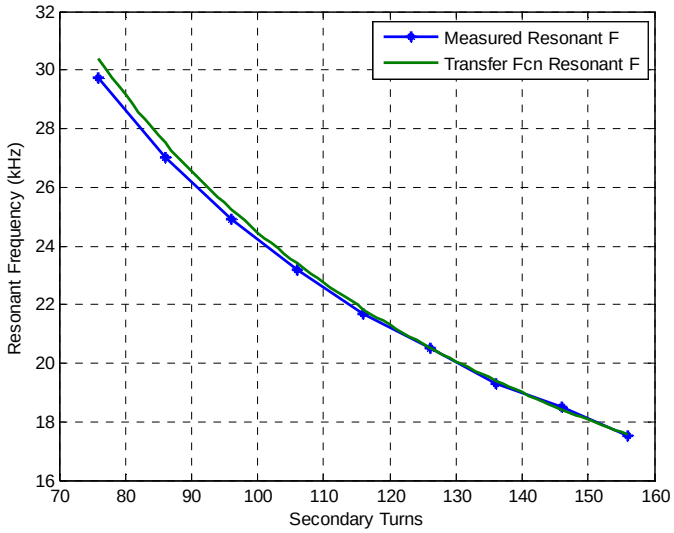


Figure 3.45 Measured and calculated resonant frequency.

The analytical models found for the loosely coupled transformer system accurately match measured values and numerical simulations and may be used to design the turns ratios of such systems in the future. Such models will greatly reduce the design time of such systems by allowing the elimination of trial and error approaches to loosely coupled resonant transformer design.

3.7 Doubling 3 Phase Rectifier

A doubling three phase rectifier is used to further increase the boost ratio of the system, compensating for the reduction in boost ratio that occurs when connecting the transformers to a capacitively loaded rectifier. A diagram of the rectifier configuration is presented in Figure 3.46 where the three transformers are connected in a Y configuration with the Y point connected to the midpoint of a capacitor bank across the output.

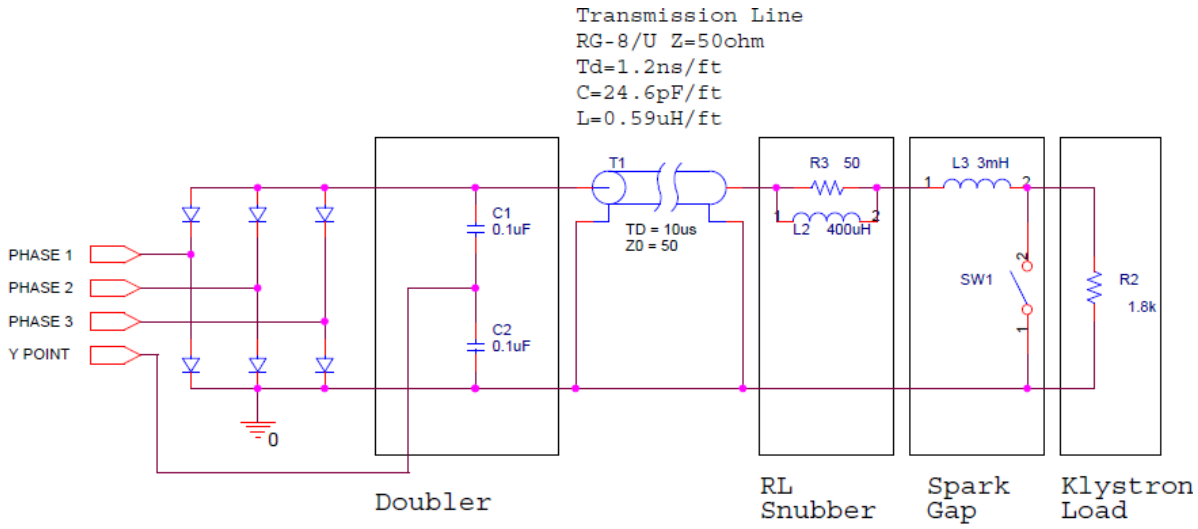


Figure 3.46 Doubling three phase rectifier.

3.7.1 Rectifier Stack Design

The rectifier stack is comprised of a series connection of a number of 1.4kV, 75A diodes providing a nominal rating of 160kV. The rectifier is capable of continuous operation when immersed in oil; however for the low duty cycle operation of this supply, they are capable of running in air. The set of rectifiers is presented in Figure 3.47.

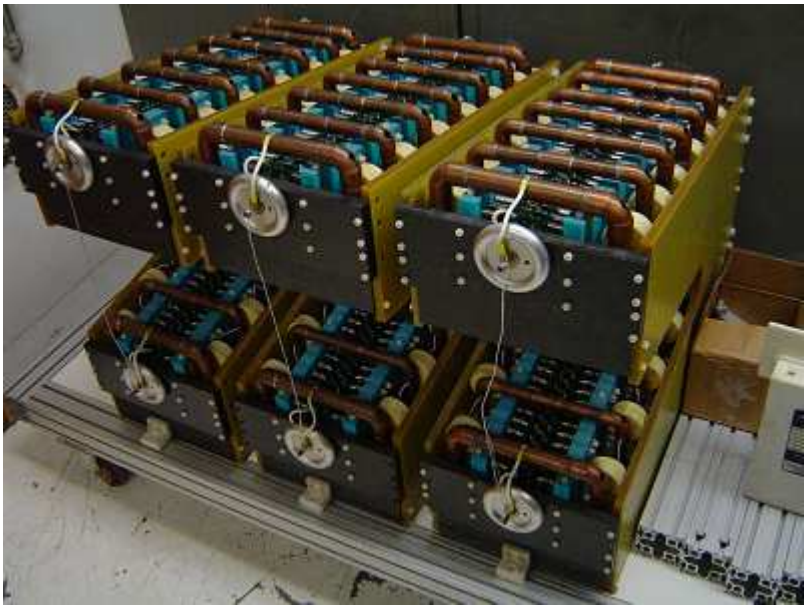


Figure 3.47 Rectifier stack.

3.7.2 Doubling Configuration

The output of the rectifier connects to a 0.05uF, 150kV capacitor bank consisting of four 75kV, 0.05uF capacitors connected in a series parallel configuration as shown in Figure 3.48. Previously utilized Mylar foil capacitors with a total bank rating of 100kV were damaged by reflected HV pulsed on the transmission line during klystron arcs; the newer capacitors provide greater resistance to damage.

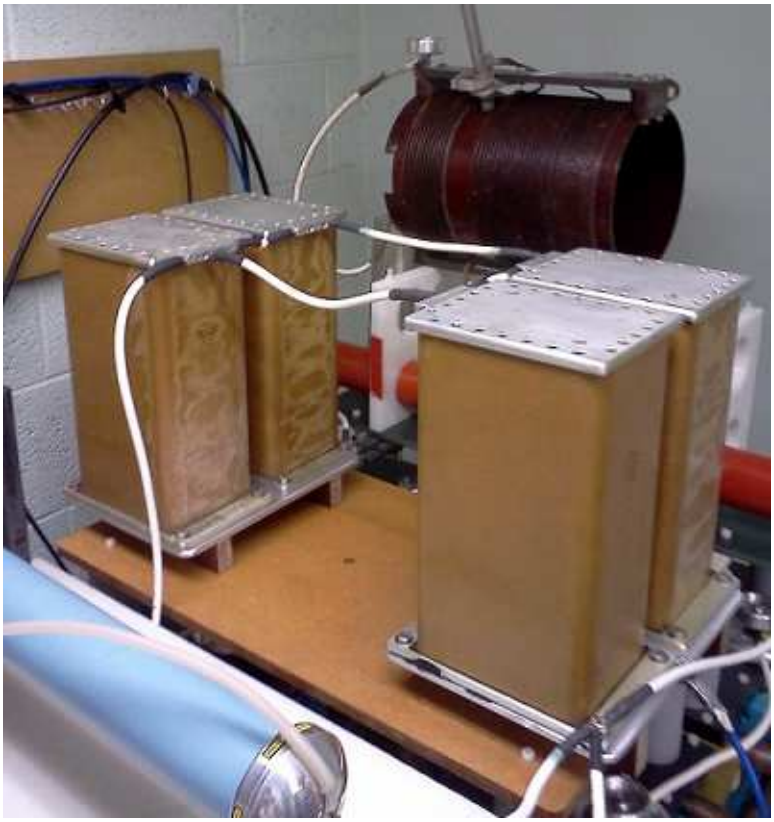


Figure 3.48 Doubler capacitors.

3.8 Harmonic Mitigation

The power supply is required to produce low ripple on the output due the klystron's gain dependence on input voltage. Methods of harmonic mitigation

3.8.1 Capacitive Loading

Capacitive loading of the supply output by the doubler capacitors provides some mitigation of ripple, however the capacitance cannot be increased without adversely effecting pulse rise time and potentially damaging the klystron tube during an arc due to the increase in stored energy.

3.8.2 Parallel LC Harmonic Filter for 6th Harmonic Mitigation

The primary source of ripple is 6^{th} harmonic from the three phase rectifier. A series LC circuit tuned to the switching frequency, and placed across the power supply output may be used to selectively null out the sixth harmonic. The circuit may be tunable to a given harmonic frequency by use of a variable inductor as shown in Figure 3.49.

Figure 3.49 LC harmonic filter.

3.9 Safety Systems

Due to the possibility of a klystron arcing, several safety systems were added to the supply to prevent damage to the klystron tube and power supply. An RL snubber placed in series with the tube will prevent an HV pulse from being reflected on the coaxial cable during an arc, and to some extent will limit energy dissipated into the tube during such a fault. A triggerable spark gap in parallel with the tube will rapidly crowbar tube voltage during a fault, preventing arc damage from occurring.

3.9.1 LR Snubber

The LR snubber is constructed with a 50ohm resistor in parallel with a 400uH inductor. The system is constructed by winding copper magnet wire around a large carborundum resistor, allowing the volume of both circuit elements to be minimized as shown in Figure 3.50. The snubber is enclosed a PVC insulation pipe for safety and mounted within the klystron cabinet, close to the cathode terminals.

Figure 3.50 LR snubber.

3.9.2 Crowbar Spark Gap

An innovative crowbar spark gap has been designed with three modes of triggering. The spark gap consists of two 4” metal balls enclosed within a polycarbonate tube. Both balls have trigger electrodes down the center with the grounded ball’s trigger electrode connected to an external source and the top trigger electrode connected to across a 3mH inductor in series with the klystron tube. In event of an internal arc, the rapid di/dt will fire the top trigger electrode by providing a high voltage difference across the inductor. A 5nF capacitor in series with the trigger electrode will prevent current from being shunted through the trigger electrode. In the event of overvoltage, the gap will breakdown without a trigger source. In the event that an arc is detected first by external equipment monitoring the klystron RF output power, the bottom trigger electrode may be fired. The HV trigger pulse is generated by an SCR discharging a capacitor through a step up transformer. The SCR is

triggered by a fiber optic receiver. When the gap fires, a current sense transformer sends an output pulse on a fiber optic cable connected to the supply's control system, which will then terminate power transfer. A schematic of the gap is presented in Figure 3.51 and a picture of the gap is presented in Figure 3.52

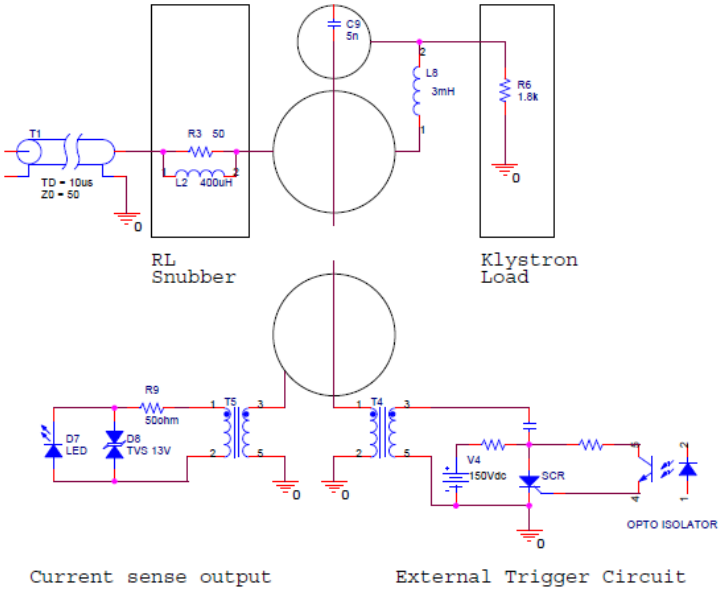


Figure 3.51 Spark gap schematic.

Figure 3.52 Spark gap.

3.10 Summary

This chapter presented the design of the resonant power supply including power electronics, magnetics, filtering, crowbar and snubber. A capacitor charging power supply was chosen to provide constant current charging ability to bring a 0.3F electrolytic capacitor bank up to 900V. The capacitor bank is sized appropriately such that the voltage droop during a 10ms pulse is acceptable. Each capacitor is individually fused for safety in the event of an internal short. The power supply drives the transformers through a set of three independent h-bridges and utilizes low inductance busbars and DC link stiffing capacitors. A low inductance relay is designed to disconnect the capacitor bank from the IGBT network when not pulsing. Gate drivers providing fiber optic isolation and control as well as rapid switching time were chosen for their plug and play functionality. The final design of a resonant transformer, its nano-crystalline iron core, the design of its windings, the enclosure and insulation of its secondary winding within dedicated oil tanks is described. Numerical and analytical models of loosely coupled resonant transformers were developed. Configuration of a coupling three phase rectifier configuration and filter networks for harmonic mitigation are presented. Several safety systems to protect the klystron load including an LR snubber and crowbar spark gap have been developed and installed in the system.

Chapter 4 Control System Design

This chapter presents the design of the resonant power supply's control system. Section one presents an overview of the control system design. Section two presents the specifications and capabilities of the microcontroller used in the control system. Section three presents the analog isolators for the feedback control signals. Section four presents the feedback voltage dividers that measure voltage at the klystron tube, power supply output and capacitor bank. Section five presents current measurement systems for capacitor bank input current. Section six presents the fiber optic I/O isolators. Section seven describes the microcontrollers operating code. Section eight summarizes the chapter.

4.1 Overview

The control system for the power supply is designed around a dsPIC30F2020 microcontroller due to the controller's built in peripherals. The microcontroller was specifically designed for SMPS control, including such features such as phase locked PWM generators, high speed analog to digital converters, and general purpose peripherals such as UARTs, timers, interrupt controllers, and oscillators. The microcontroller board is mounted in a shielded metal box with internal power supplies, analog isolator modules fro ground loop isolation and a fiber optic I/O system allowing the system to interface with the IGBT drivers and receive trigger or fault signals. A picture of the control system is presented in Figure 4.1.

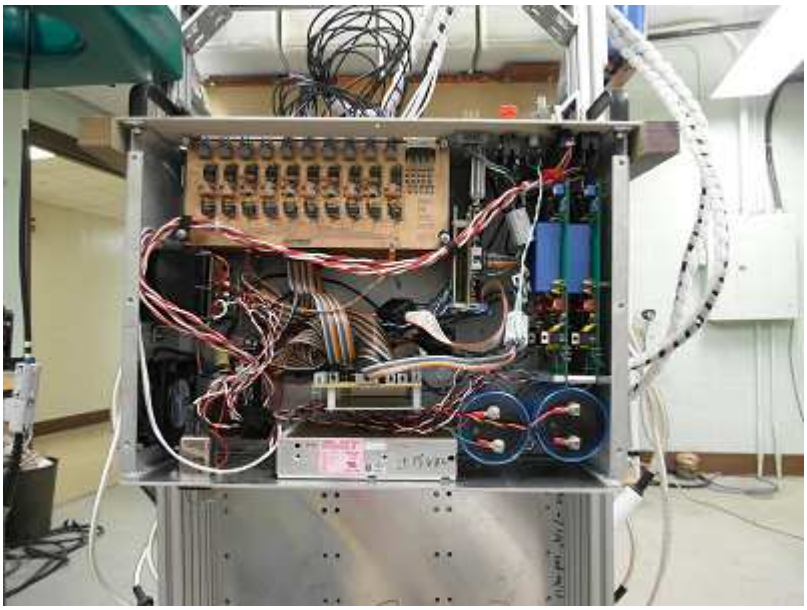


Figure 4.1 Control system.

4.2 dsPIC Microcontroller

Use of a high performance microcontroller specifically designed for SMPS operation allows a small compact system to be designed with minimal external equipment. A dsPIC30F2020 microcontroller manufactured by Microchip was chosen as the basis of the power supply's control system due to its small size, low cost, high processing speed of 30MIPS, built in DSP functionality, high speed ADC, and built in SMPS control modules. A datasheet for the microcontroller is given in [M].

The dsPIC microcontroller has several features designed to optimize performance in SMPS designs. The pinout of the microcontroller is given in Figure 4.2. The microcontroller is available in a 28 pin DIP package allowing easy integration and prototyping.

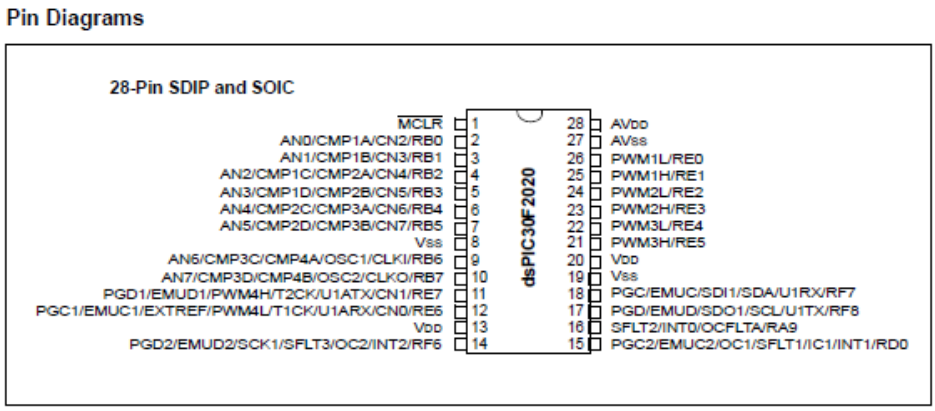


Figure 4.2 dsPIC30F2020 pinout.

The microcontroller has an internal fast RC oscillator allowing 30MIPS operation without use of an external crystal. A 10bit, 2msps ADC provides rapid sampling of analog inputs and allows the use of digital filters. The ADC can sample from 8 pins and can sample up to two inputs simultaneously. The system provides the option to synchronize ADC

sampling between switching events to prevent interference from switching transients. A set of three built in timers allows accurate timing of output pulses and repetitive triggering of interrupts for i/o operations. A built in UART allows communication over a serial bus for computer interfacing.

Most importantly, this series of microcontroller has a built in SMPS controller with 4 pairs of PWM outputs with the ability to synchronize the time bases and phase offsets between the different channels. Further, the PWM periods can be set to only update at the switching boundaries. These features are critical for use on this power supply since they ensure that the three separate phases will always remain 120 degrees apart, and that changes in PWM period will only occur near zero current, so that soft switching may be maintained.

A block diagram of the microprocessor hardware is shown in Figure 4.3.

FIGURE 1-2: dsPIC30F2020 BLOCK DIAGRAM

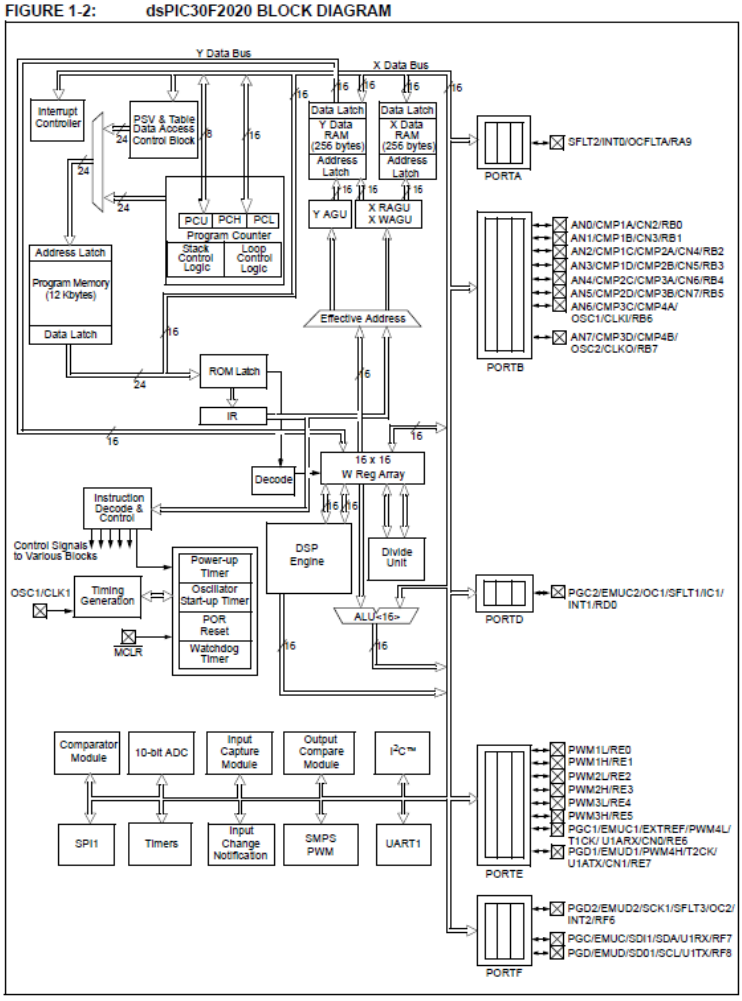


Figure 4.3 Microcontroller subsystem block diagram.

4.3 Analog Input Galvanic Isolation

Due to pulsed magnetic fields in the experiment area, isolation between the input and output of the power supply is required to prevent ground loops. As such the high voltage dividers providing voltage feedback to the control system must have their ground connection isolated. A set of four isolator boards using Analog Devices AD215BY isolator modules are connected between the voltage dividers and the microcontroller to provide noise filtering and ground loop isolation as shown in Figure 4.4. The isolator module has a +-15 volt dynamic range and a 120kHz bandwidth. A set of op-amp based analog filters limits the gain to 100kHz, while allowing signal amplification between a gain of 1 and 1000. A block diagram of the AD215 is provided in Figure 4.5. A data sheet for the AD215 is provided in [L].

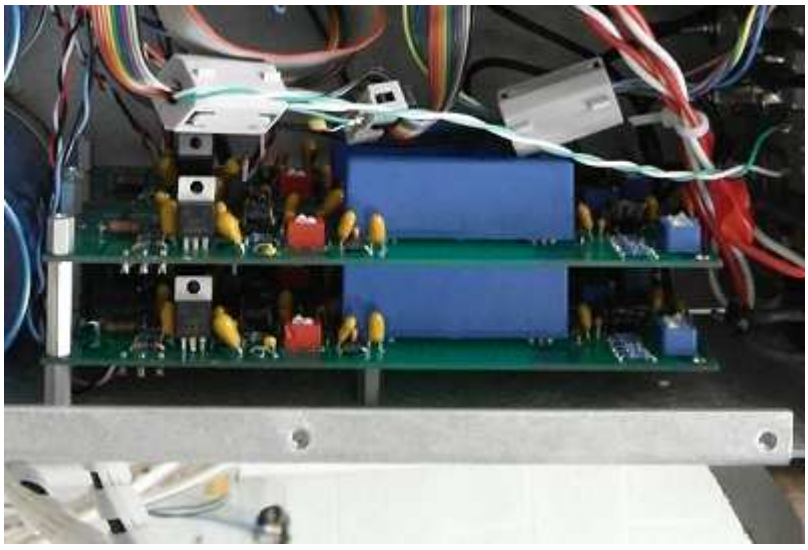


Figure 4.4 Isolation modules.

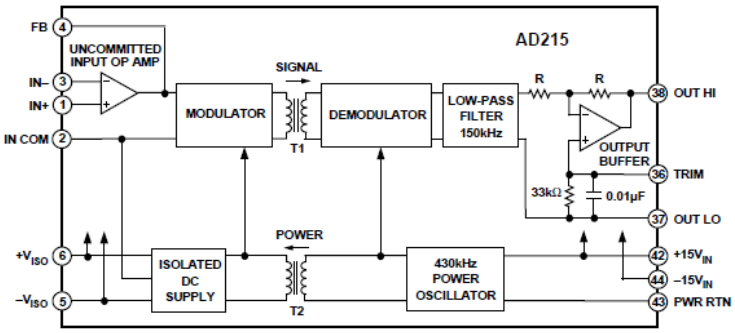


Figure 4.5 Isolator block diagram.

4.4 Feedback Voltage Dividers

A pair of voltage dividers is used to allow the control system to monitor voltage at the power supply output and at the klystron cathode. A 10000:1 Ross engineering voltage divider provides a high bandwidth measurement of output voltage while a custom voltage divider consisting of a high voltage resistor-capacitor divider with a Tektronix P6015A, 1000:1, 40kV high voltage probe connected to the half-way point extends the probe's range to 80kV and increases the division ratio to 2000:1 while maintaining the probe's original high bandwidth characteristics. Both probes are shown in Figure 4.6.

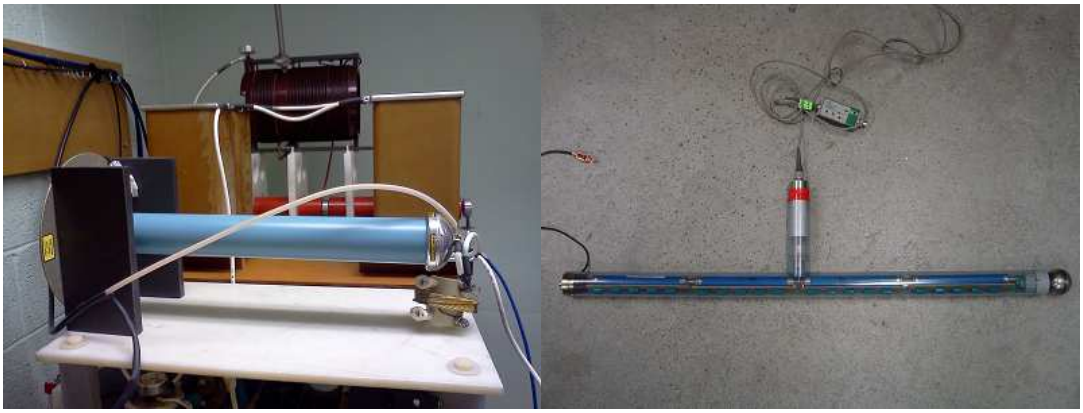


Figure 4.6 High voltage probes.

4.5 Current Measurement

Capacitor bank current is measured with a Pearson electronics 301X current transformer, providing a 0.01V/A output. The transformer mounts on the power supply frame next to the low inductance bus bars and the positive cable from the capacitor bank passes through the center as shown in Figure 4.7. The transformer is rated at 50kA peak current, with a droop rate of 3% per millisecond and has a current time product of 22kA-ms. The low and high frequency -3dB points are 5Hz and 2MHz. A complete data sheet is provided in [K]

Figure 4.7 Pearson transformer.

4.6 Fiber Optic I/O

In order to maintain galvanic isolation between the control system and the IGBT drivers, as well as peripheral equipment the control system uses a digital fiber optic isolator system as shown in Figure 4.8. The isolator converts fiber optic signals to TTL logic levels for interfacing with the microcontroller. The isolator consists of four boards, two transmitter boards and two receiver boards. Each board has 10 channels and provides an LED indicator of each channel's status to aid in diagnostics. The circuit diagram for the fiber optic interface is provided in Figure 4.9 (schematic courtesy of Mikhail Reyfman).

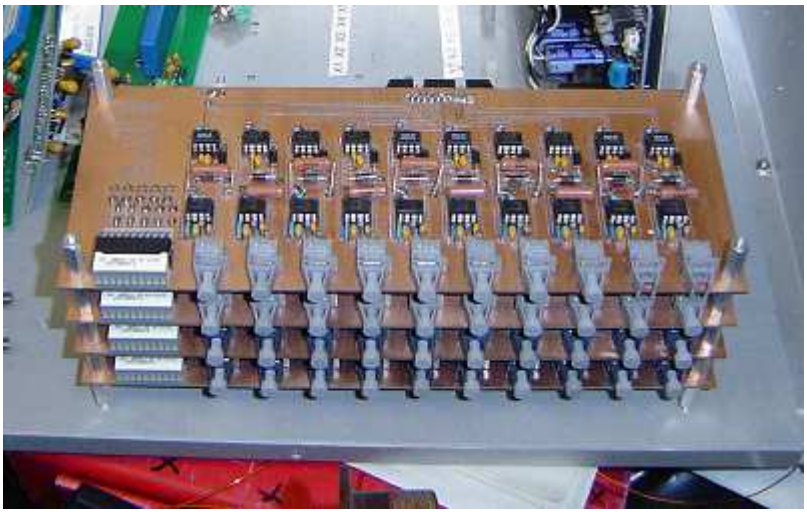


Figure 4.8 Fiber optic isolator.

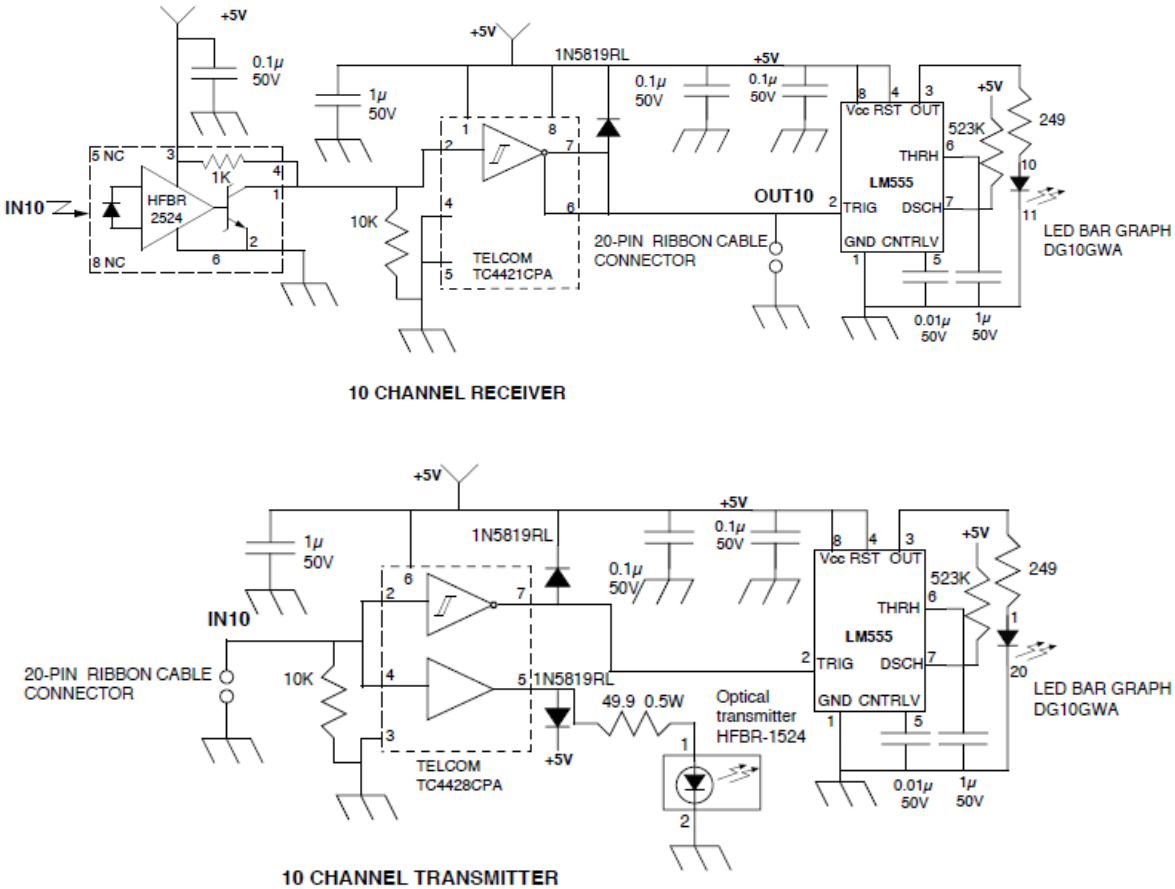


Figure 4.9 Fiber optic interface.

4.7 Microcontroller Code Design

The microcontroller code was designed to allow accurate, efficient feedback control, reliability and safety. The core allows for automated shutdown of the power supply in the event of voltage sensor failure, overvoltage condition, shorted output on startup, infinite loop conditions, and external fault inputs. Feedback control is accomplished by calculating the desired PWM period from the bank voltage measured with the ADC based on the equation found in Figure 5.14. The controller receives pulse start and external fault inputs over the fiber optic i/o module and pulse parameters are set by sending serial strings to the controller from a computer.

Complete microcontroller operating code is given in [N] with the leading number being the line of code. On startup, the processor in configured by lines 20-25 to use the fast RC internal oscillator, to run at the maximum execution speed of 30MIPS, and to enable the watchdog timer that will monitor operation during a pulse for safety. Variable values are initialized and stored into RAM by lines 100-196. The main function is located between lines 200-316.

Immediately on startup of the processor's code loop, lines 205-213 set all PWM outputs are overridden and turned off, and a pulse lockout timer bit is set to inhibit startup for a preset period, and feedback control is enabled. This is performed so that if a voltage transient caused by an external arc caused the microprocessor to reset but left the PWM peripheral on, the power supply would be prevented from receiving gate drive signals without the microcontroller realizing the pulse output was active. Line 225 calls the PWM lockout

subroutine, starting a 10ms countdown timer and inhibiting pulse activation for a set time. This was implemented early in testing when it was observed that after an arc causes the microcontroller to reset, residual transients and noise from ringing inductances occasionally triggered the input pin that calls the pulse start command, immediately triggering a second pulse.

The main code loop starts on line 252. In the main loop the processor checks bit flags controlling when collected date is sent to the connected computer over a serial port. If one of the timers that controls data output sets, the subroutine to transmit data is called on line 298 calling the subroutine on line 322 which transmits power supply status and ADC input values to a connected computer to allow the operator to monitor capacitor bank charging. Received commands from the operator are processed by the subroutines called between lines 303 and 312. Received serial characters are stored in RAM by the subroutine starting on line 358.

All power supply output pulse operations are handled by interrupts and synchronized by internal timers to ensure reproducible and accurate timing.

The interrupt service routine (ISR) on line 414 is triggered of an external fault is detected at any time. When triggered the subroutine to stop PWM output is called, the power supply is shut down, and an error message is sent to the user.

The ISR on line 450 is triggered of the fiber optic module receives a pulse start command. The system checks to see that pulse is not already running, the pulse start lockout timer has expired, and that the capacitor bank voltage is above a minimum threshold. If these conditions are met, the PWM start subroutine is called.

The ISR on line 472 controls ADC sampling rates and is triggered every time timer 1 expires. ADC sampling is automatically triggered when the timer expires, so the ISR only has to reset the timer.

The ISR on line 492 is used to terminate the power supply output pulse and set the PWM lockout timer. Timer 2 is started at the beginning of the pulse for the given pulse length, when it expires it triggers the ISR which calls the PWM stop subroutine and shuts down the pulse. The first time the timer calls the ISR, it is reset for 10ms and used as the PWM lockout timer, inhibiting another pulse from starting until it expires the second time.

The ISR on line 561 is triggered when timer 3 expires and sets a flag that is checked in the main loop to send the ADC readings back to the user over the serial port.

The PWM start subroutine starts on line 526. When called, it configures several variables used to acquire data during the pulse, sets timer 1 to a shorter period thereby increasing the ADC sampling rate, calculated the starting switching frequency based on capacitor bank voltage, configures the PWM generator for use, enables the watchdog timer that will reset the cpu if the control loop locks up, configures timer 2 to set the output pulse length and then enables the PWM output pins, starting the three phase gate drive signals.

The PWM stop subroutine starts on line 624. When called it resets timer 1 to a longer period thereby decreasing the ADC sampling rate, disables the PWM output pins, stopping the three phase gate drive signals, and turns off the watchdog timer.

The ISR on line 661 is triggered when the ADC conversion completes, meaning that the ADC, that was triggered when timer 1 expired, has completed digitizing the requested input channel and has it stored in its buffer. The ADC buffer values are stored in RAM, and

voltages, and currents are computed by multiplying by the given scale factors. If the pulse is active, line 689 calls the PWM control subroutine that uses the ADC measurements to update the PWM period in order to obtain the correct boost ratio and stabilize output voltage. Lastly, after the control routine returns, the watchdog timer is cleared, preventing it from reaching zero and resetting the cpu if the control routing did not hang.

The PWM control subroutine that starts on line 712, uses the ADC measurements to update the PWM period in order to obtain the correct boost ratio and stabilize the output voltage. If feedback control is turned on, line 769 uses the linearized model for PWM period found in Figure 5.14 to calculate the desired period. The period is in terms of timer register clock counts. Safety checks are performed between lines 775 and 800 limiting the requested PWM period into acceptable bounds and monitoring voltages and sensor inputs for problems. Lines 804 to 811 set the PWM generator module to the newly calculated period and calculate the phase offset ties to maintain 120 degree separation between the phases.

Other code between lines 818 and 1410 is used to configure peripherals, the ADC, timers, interrupts, the PWM generator, and print out recorded data to the terminal. The subroutine starting on line 1247 is used to parse received serial strings for commands and present the operator with on screen menus on the terminal.

The code described herein provides reliable, accurate feedforward control to stabilize output voltage based on capacitor bank voltage droop.

4.8 Summary

This chapter presented the design of the resonant power supply's control system. An overview of the control system design and components was presented along with the specifications and capabilities of the dsPIC30F2020 microcontroller used in the control system. Support equipment such as analog isolators for the feedback control signals, the associated voltage dividers that measure voltage at the klystron tube, power supply output and capacitor bank, capacitor bank input current sensors, and fiber optic I/O isolators have been presented. A description of the microcontrollers operating code describes methods for SMPS control and methods for closing the control loop.

Chapter 5 Power Supply Testing

This chapter presents the results from testing the power supply. Section one presents a spice simulation model for the power supply and predicted numerical results. Section two presents open loop testing of the power supply. Section three presents results from testing the feedback control system to compensate for capacitor bank droop and stabilize output voltage. Section four presents models and testing of several filters to reduce harmonics. Section five presents simulation and testing of arc faults, operation of the spark gap and installation of the snubber circuit. Section six summarizes the chapter.

5.1 Spice Simulation Model

A Spice simulation of the power supply has been created utilizing the simplified transformer model as shown in Figure 5.1. The secondary referred voltage is equal to the turns ratio of 1.36 times the maximum bank voltage of 900V. A doubling three phase rectifier feeds a simulated klystron load consisting of a 1.8kohm resistor without any output filtering.

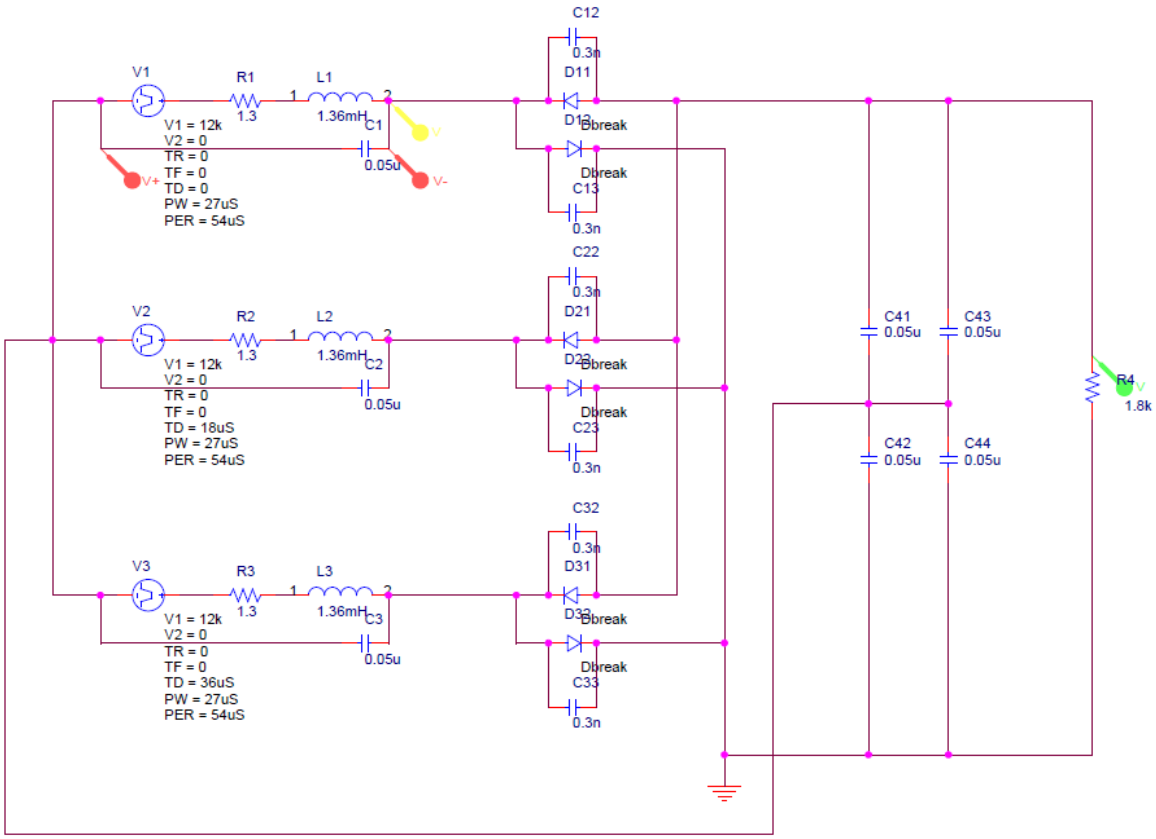


Figure 5.1 Simplified spice model.

A time domain simulation of output voltage and the phase voltage at a rectifier terminal with respect to ground was generated, simulating operation of the three phase rectifier and

transformer assembly. The simulated output voltage is lower in magnitude then the actual power supply output voltage measured at full bank voltage, however rise time, output waveform and ripple are almost identical with the exception that there is no capacitor bank droop on the simulation as seen in Figure 5.2.

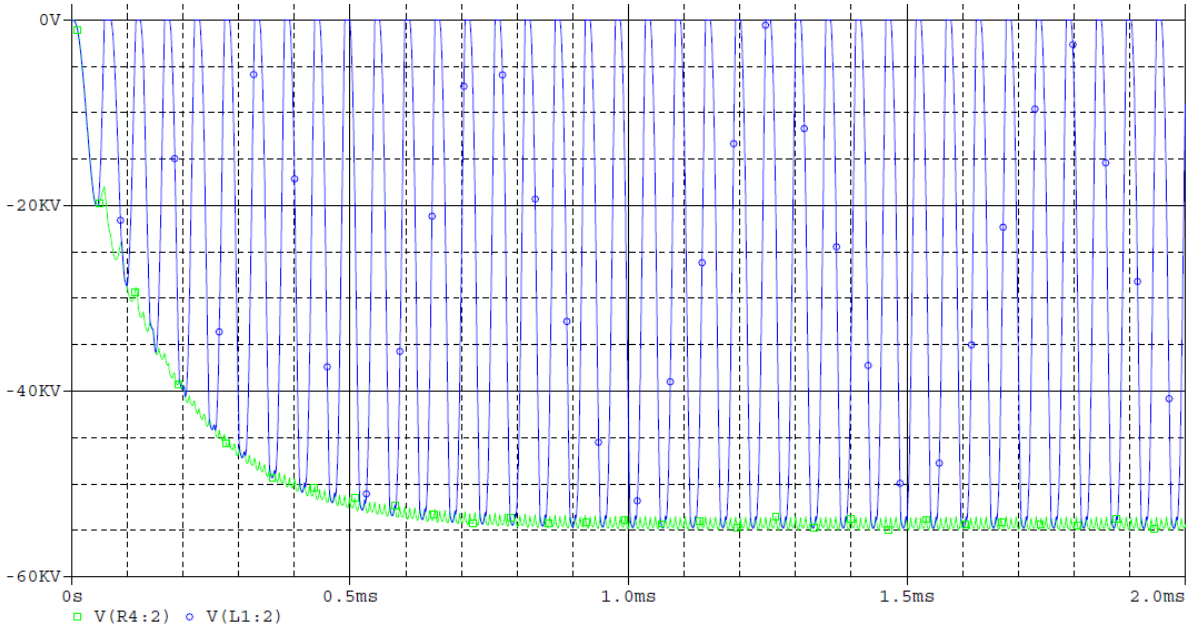


Figure 5.2 Time domain simulation of output voltage.

Plots of output voltage and the voltage on a transformer phase equivalent to that in the spice simulation are plotted in Figure 5.3. Plots are displayed before and after digital filtering with a median filter as seen in Figure 5.4 to reduce high frequency noise. A median filter is a non-linear filter that returns the median point in a sliding window. Median filters are excellent for extracting a desired signal from higher frequency noise. The oscilloscope used to capture the waveform digitizes at 5ksamp/ms, and a 50point median filter was applied to clean up the signal to better visualize the phase voltage and harmonic ripple on the output. The 6th harmonic was found to dominate the power spectrum.

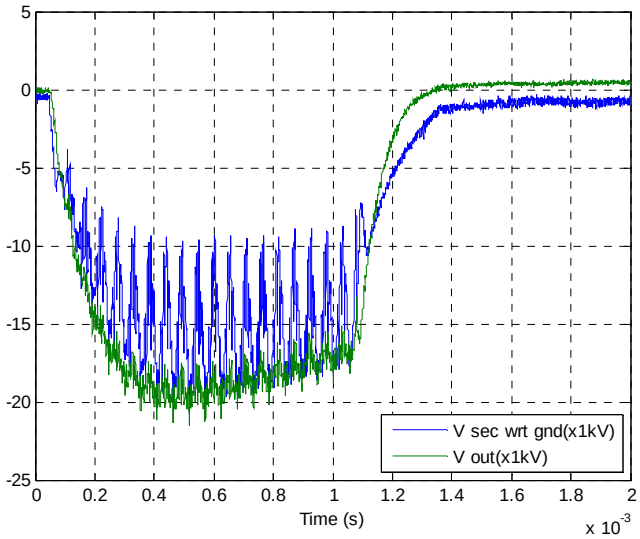


Figure 5.3 Output voltage and phase voltage before median filter.

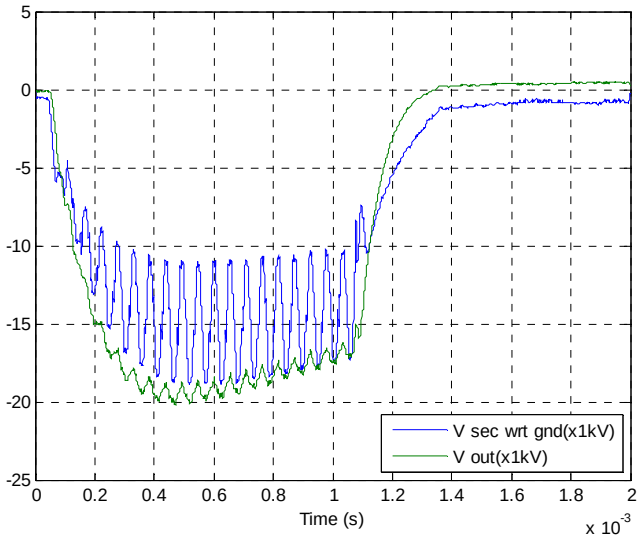


Figure 5.4 Output voltage and phase voltage after 50 point median filter.

Measurements of voltage across the primary of a resonant transformer were acquired. Near resonant frequency, soft switching occurs on both the leading and lagging edges of an IGBT switching event as shown in Figure 5.5. As switching frequency increases above

resonance, the voltage waveform begins to lead the current waveform causing a slight increase in switching current at the leading edge, but maintaining ZVS soft switching at the lagging edge due to the reverse diodes commutating the primary current as shown in Figure 5.6.

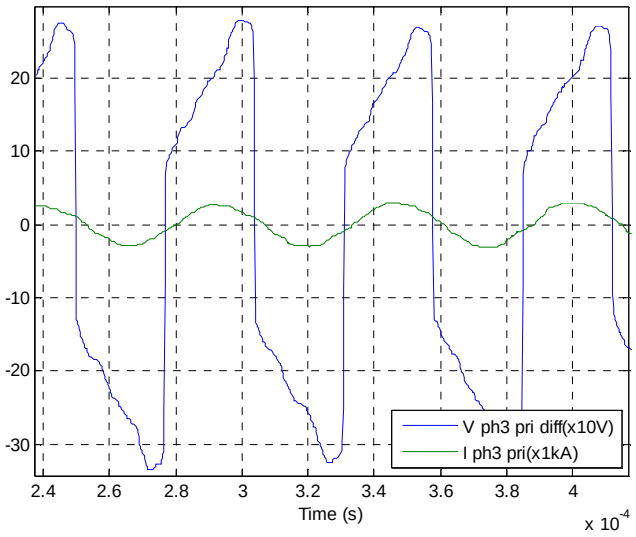


Figure 5.5 Transformer primary voltage and current at 18.5kHz.

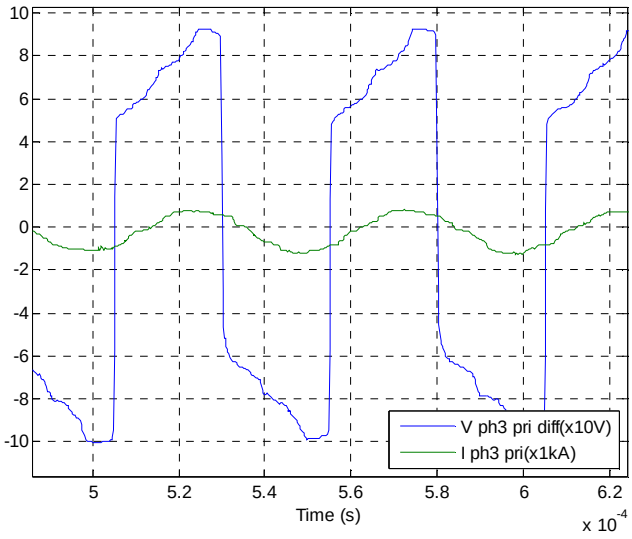


Figure 5.6 Transformer primary voltage and current at 20kHz.

5.2 Open Loop Testing

Initial testing of the power supply was carried out open loop to determine the proper linearized equation model for the feed forward segment of the control loop. Operation of the power supply was first verified by connecting the output voltage divider and gate drive signals to a mixed signal oscilloscope. In order to observe the relation between IGBT switching and voltage ramp up as seen in Figure 5.7.

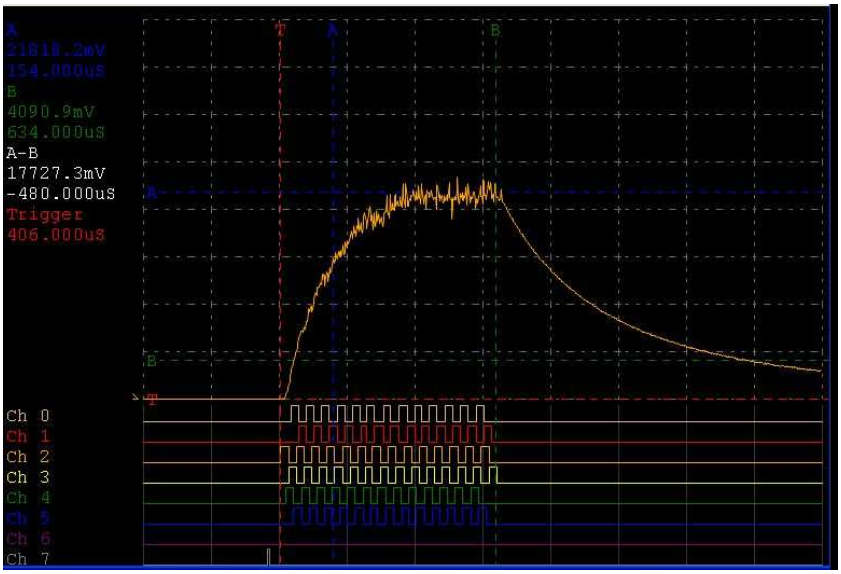


Figure 5.7 Output voltage waveform and h-bridge gate drive signals.

The testing capacitor bank voltage was increased as shown in Figure 5.8. The linear nature of the relationship between output and bank voltage indicated that the transformer cores do not saturate near full power operation.

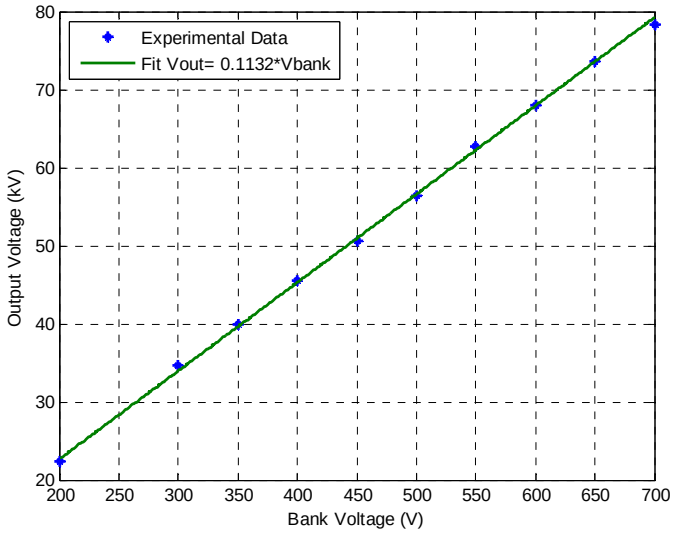


Figure 5.8 Output voltage vs bank voltage at 20kHz operation.

Testing of boost ratio at varying frequency was measured during open loop

Measurements of capacitor bank current, bank voltage, output voltage measured at the terminals of the power supply and load voltage measured at the terminals of the klystron tube were simultaneously acquired. As the pulse is generated, energy transfer from the capacitor bank causes its voltage to droop, leading to a decrease in output voltage. This effect is especially noticeable near resonance, where power transfer is maximized as seen in Figure 5.9. It should be noted that near resonance, a large amount of reactive power is stored in the resonator. After pulse termination this oscillation in the secondary coupled back to the primary and is rectified by the antiparallel diodes in the IGBTs, causing current flow back to the capacitor bank. As switching frequency increases, both output voltage and current draw from the capacitor bank decrease due to reduced power flow as seen in Figure 5.10, Figure 5.11, and Figure 5.12. The output voltage waveform becomes progressively flatter as the bank droop rate decreases.

Plots of bank voltage, current, and output and load voltage were generated by the matlab code found in [I].

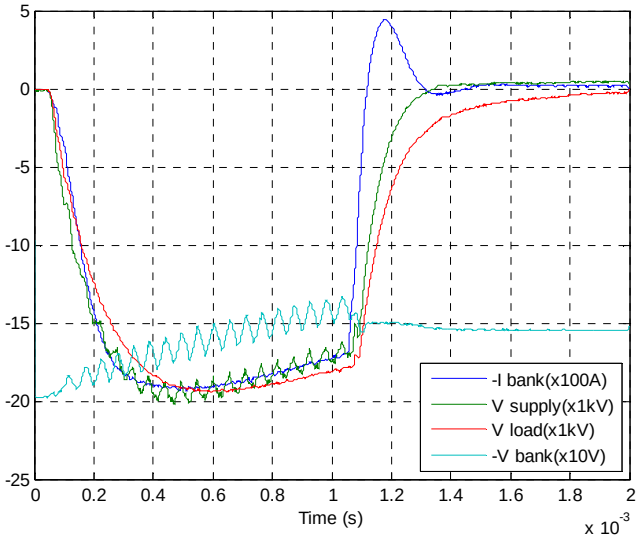


Figure 5.9 Open loop pulse at 18.5kHz and 200V bank charge.

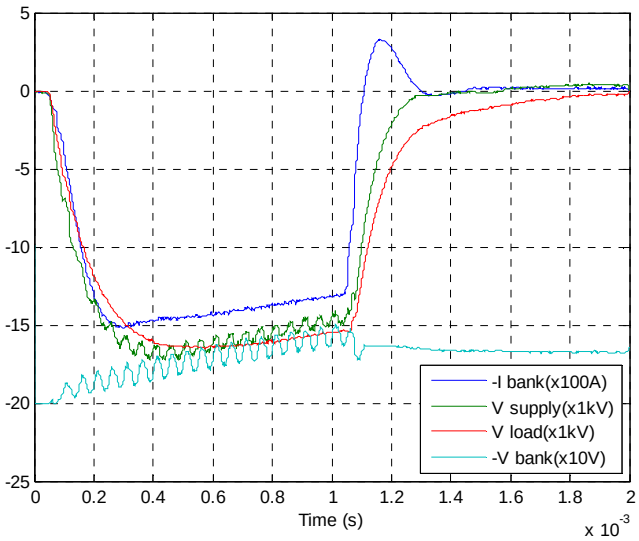


Figure 5.10 Open loop pulse at 20kHz and 200V bank charge.

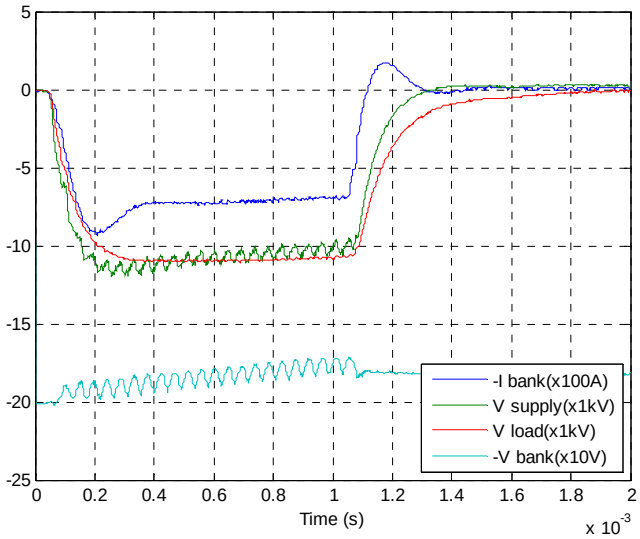


Figure 5.11 Open loop pulse at 22kHz and 200V bank charge.

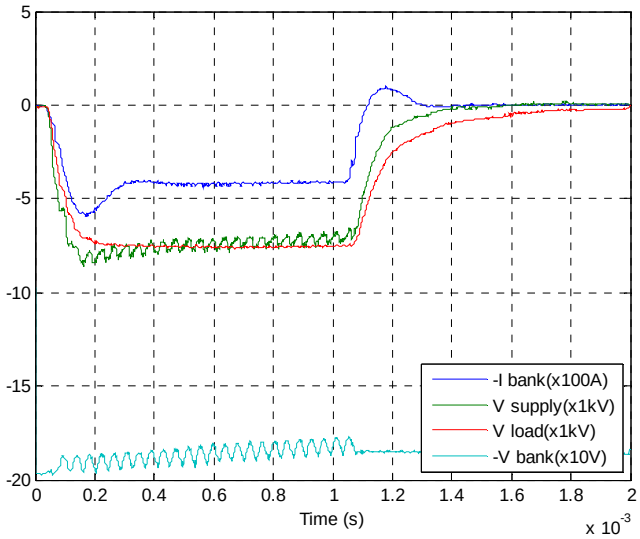


Figure 5.12 Open loop pulse at 24kHz and 200V bank charge.

Measurement of peak pulse voltage was obtained by similar methods and compared to capacitor bank voltage over the range of predicted operating frequencies to generate a plot of the power supplies effective boost ratio as a function of switching frequency. A linear

regression line was fit to this data to provide the microcontroller with a method to determine proper switching frequency for a given commanded output voltage and bank voltage using feed forward techniques. Te ploted boost ratio data and regression line are presented in Figure 5.13. Open loop plots of boost ratio vs switching frequency were generated by the matlab code found in [J].

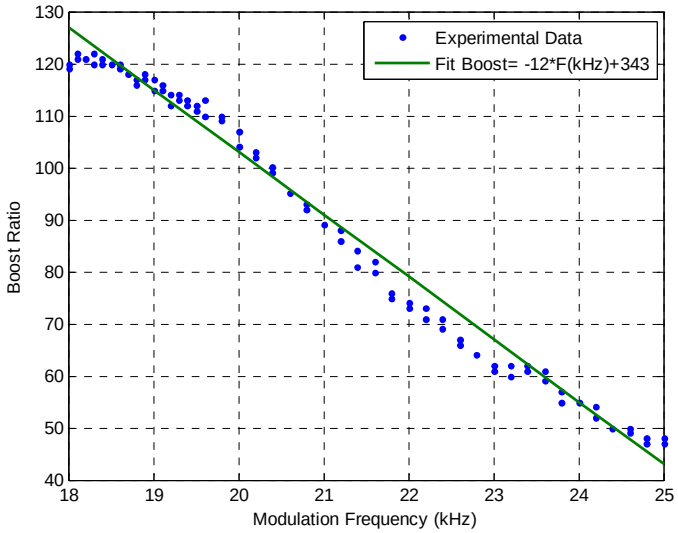


Figure 5.13 Boost ratio vs switching frequency.

5.3 Feedback Controller Testing

In order to close the control loop with feed forward techniques, the microcontroller measures the capacitor bank voltage and uses a linearized regression fit of boost ratio vs switching frequency to compute the required switching period. In practice, it is beneficial to fit a regression line to switching period vs required boost ratio as seen in Figure 5.14. This allows the removal of several divide operations from the control loop cycle; each divide operation required 16 instruction cycles to complete and is one of the more computationally intensive operations the microcontroller must complete. Removal of several divide operations was found to significantly enhance control loop execution speed, allowing an an increase from approximately 30 loops per ms to 120 loops per ms.

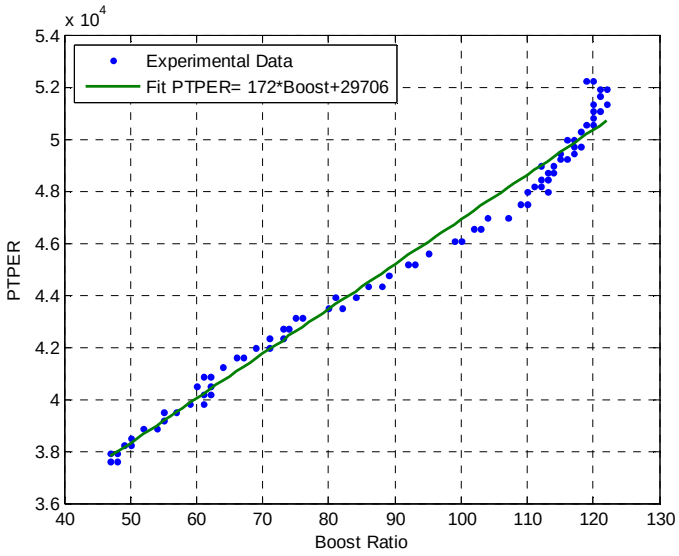


Figure 5.14 Microcontroller switching period vs required boost ratio.

Testing of the feed forward control system proved successful, allowing output voltage stabilization at -75kV output as shown in Figure 5.15. Note that the output voltage of the

power supply (light blue trace) is 10x the measurement and cursor readout. As capacitor bank voltage (bark blue trace) droops over the pulse, switching frequency in decreased toward resonance, thereby stabilizing output voltage. As expected, capacitor bank current (green trace) increases near to resonance compensation for the decreasing bank voltage to maintain constant power transfer to the load.

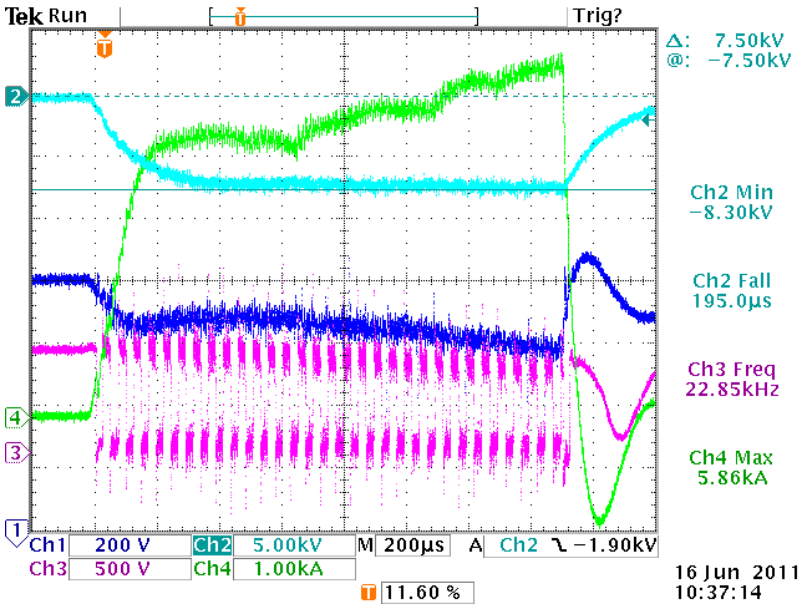


Figure 5.15 Feed forward control stabilized output voltage.

5.4 Filter Testing

A spice model of a pi and harmonic filter were examined and compared to experimental results. The spice simulation is seen in Figure 5.16 including a 0.37mH series inductor and 40nF shunt capacitor, added to the output to form a pi filter with the doubler capacitor. A series LC harmonic filter consisting of a 0.005uF capacitor and 0.41mH inductor tuned to the 6th harmonic was connected across the rectifier terminals.

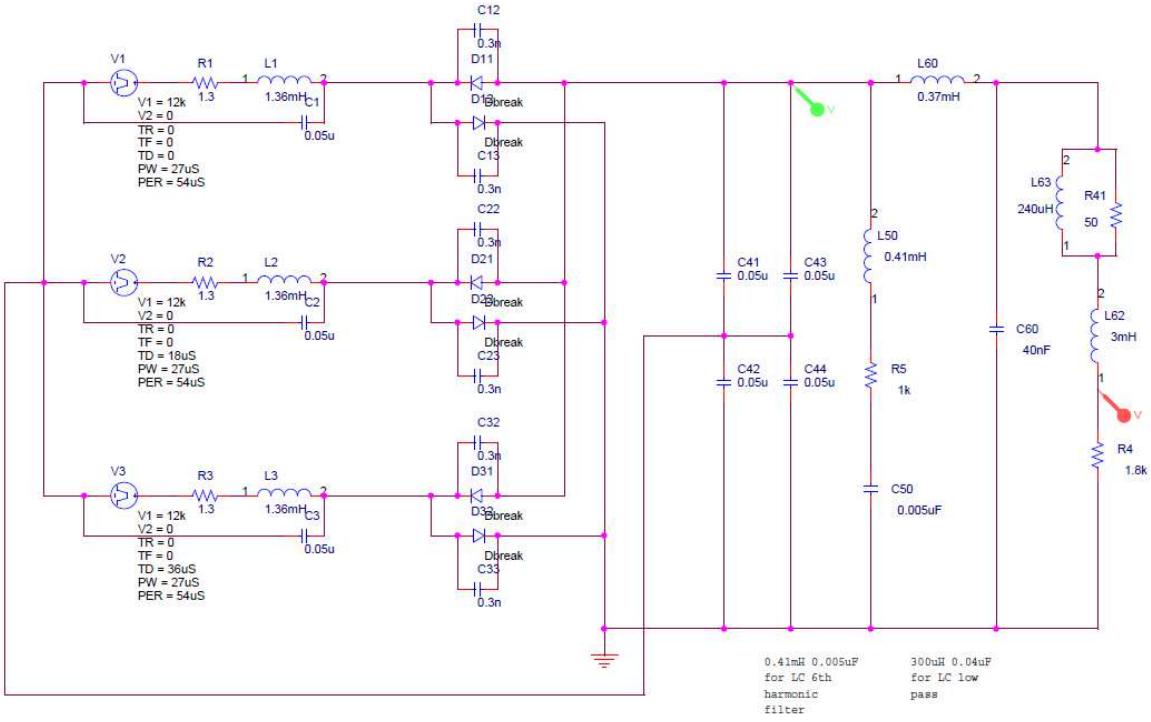


Figure 5.16 Spice model with pi and harmonic filters.

Time domain simulations seen in Figure 5.17 show an increase in rise time due to the increased capacitive loading of the output.

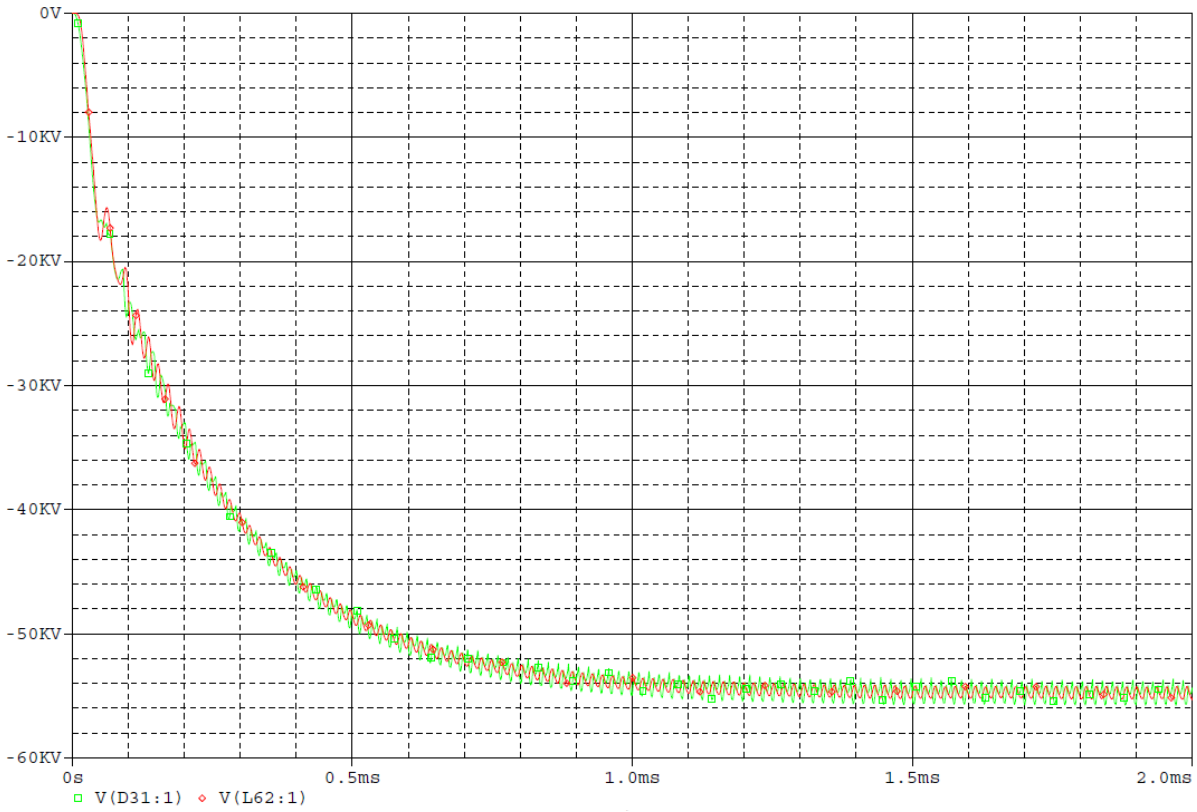


Figure 5.17 Spice time domain waveform with filters.

An FFT of the power supply output with no filters is presented in Figure 5.18 to provide a baseline for harmonic noise to compare to filtered outputs. Addition of a pi and harmonic filter greatly decreased 6th harmonic ripple as well as all other higher harmonics, as shown in Figure 5.19, however 3rd harmonic ripple was increased. Generation of 1st, 2nd, and 4th harmonics with unbalanced three phase excitation and no filter is demonstrated in Figure 5.20 by providing an approximately 10% mismatch in secondary phase voltages. Such a condition would result if the resonant frequencies of the secondaries are not equal due to variations in winding inductance.

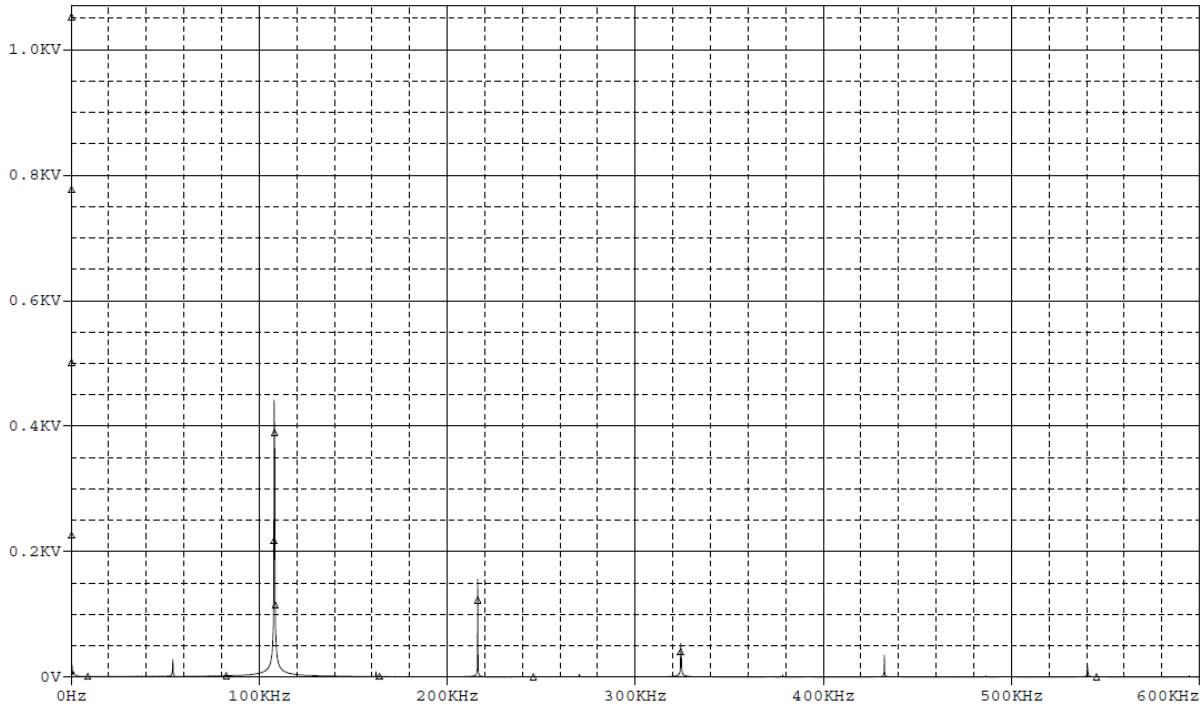


Figure 5.18 Balanced three phase operation at 70kV.

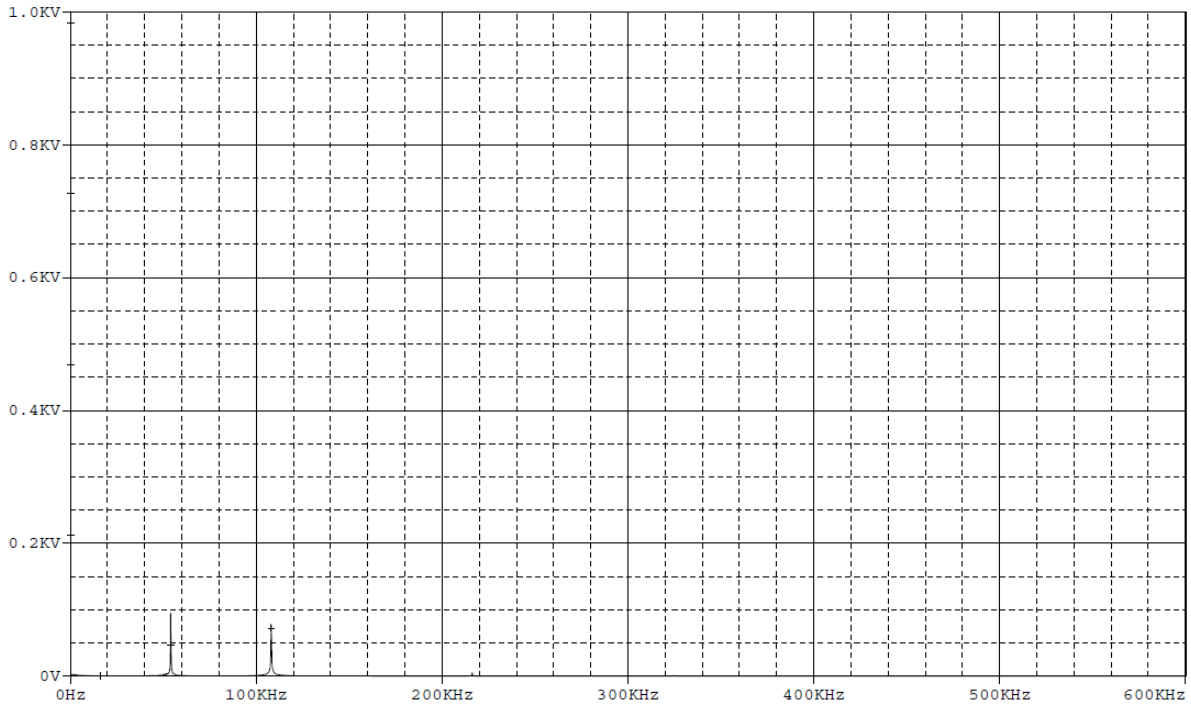


Figure 5.19 Pi and 6th harmonic filter.

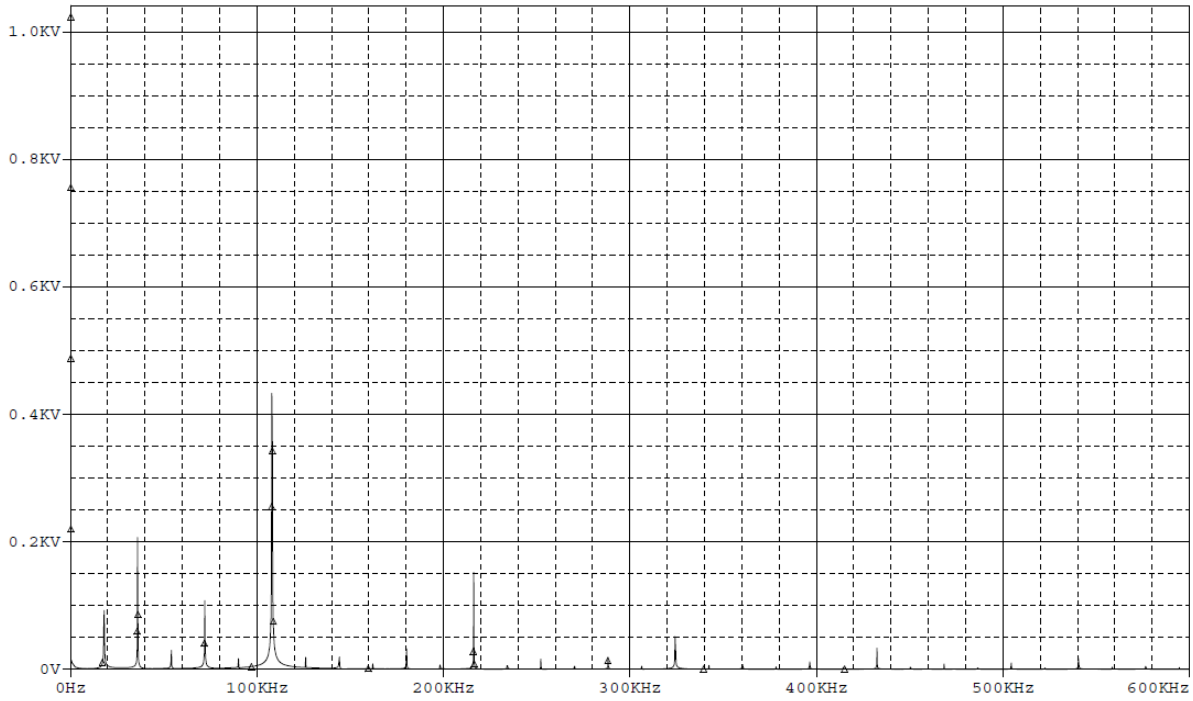


Figure 5.20 Unbalanced three phase operation.

Simulated results for the addition of filters are compared to experimental data. A baseline condition of an unfiltered output is seen in Figure 5.21. An LC harmonic filter consisting of a 0.005uF capacitor and adjustable inductor tuned to the 6th harmonic was connected across the rectifier terminals and harmonics were measured resulted in a decrease in 6th harmonic ripple as seen in Figure 5.22. With the addition of a pi filter by adding an 370uH series inductor, and 0.04uF shunt capacitor to the existing doubler capacitor eliminated 6th and higher harmonics but also increased 3rd harmonic as seen in Figure 5.23. This behavior is comparable to spice simulations.

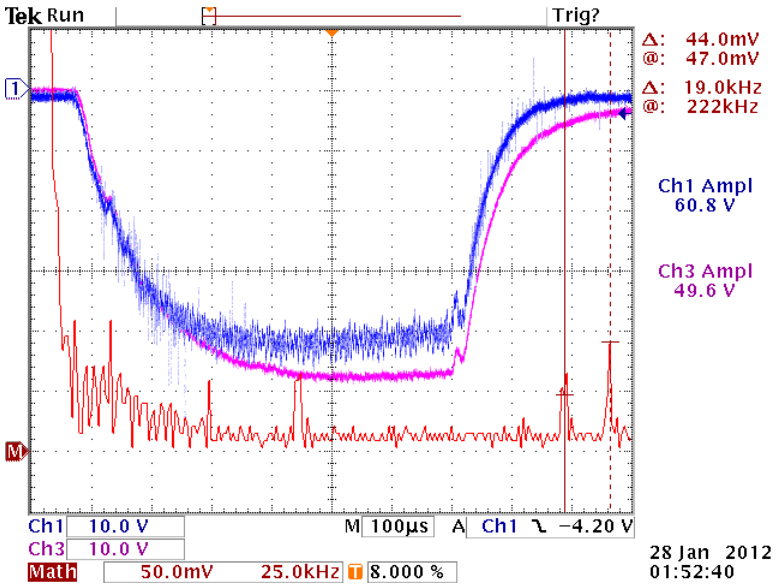


Figure 5.21 Unfiltered operation.

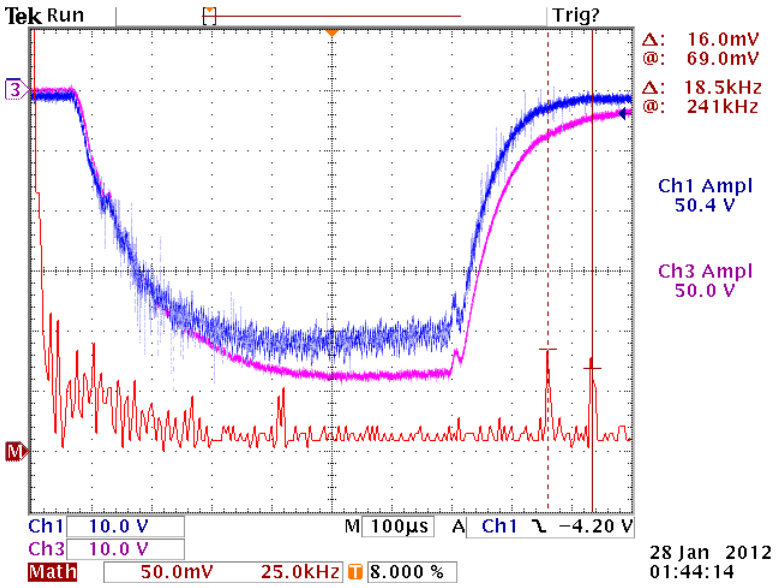


Figure 5.22 Operation with 6th harmonic filter.

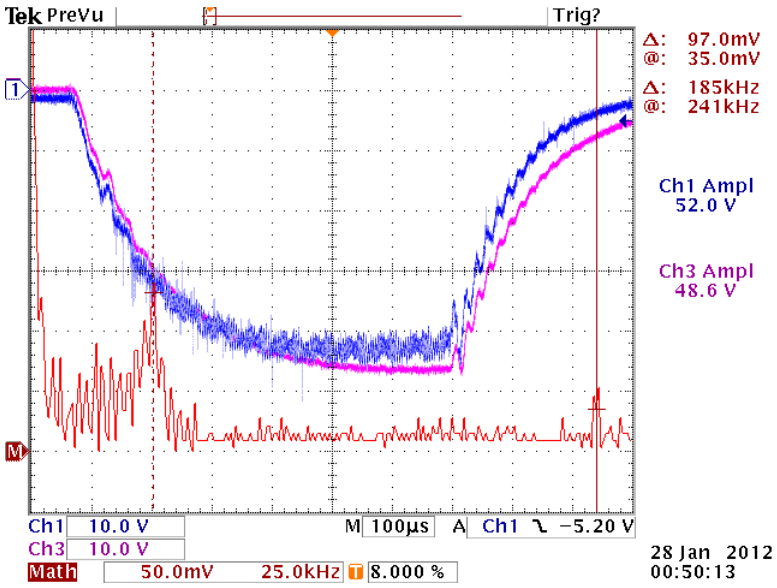


Figure 5.23 Operation with pi and 6th harmonic filters.

5.5 Output Arc Fault Simulation and Testing

A spice model to simulate crowbar or klystron arcing is shown in Figure 5.24. The model closes the switch across the terminals that would be connected to the spark gap.

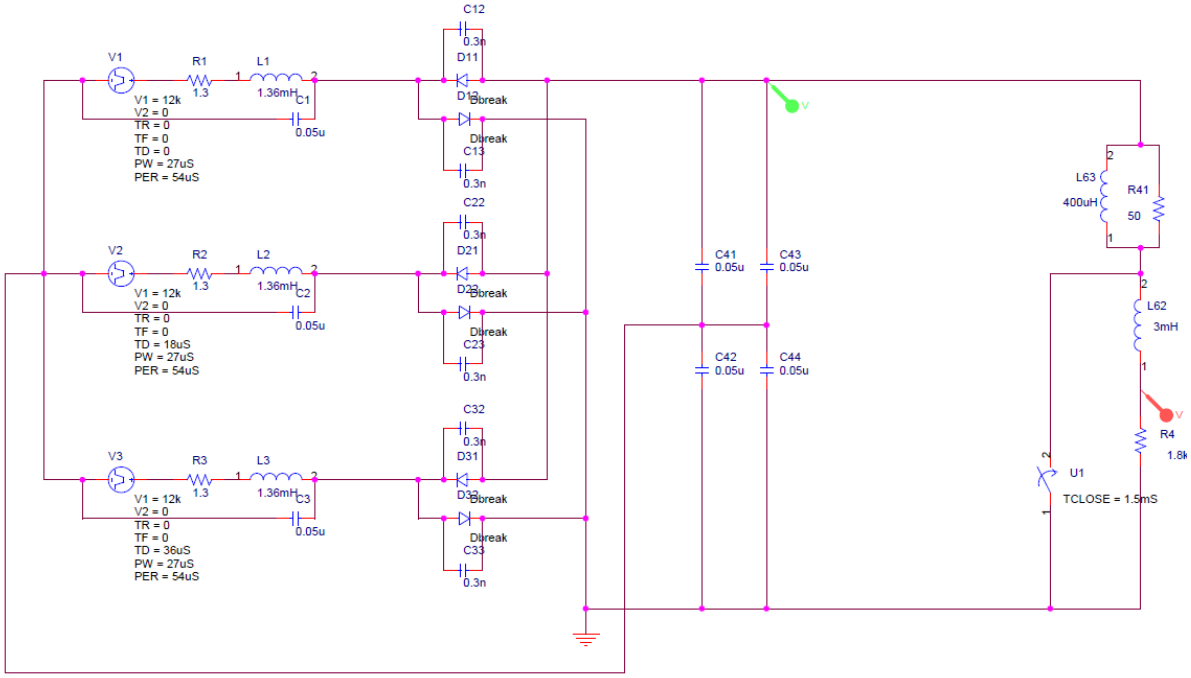


Figure 5.24 Spice model for spark gap simulation.

Time domain simulations of a spark gap firing event show almost instantaneous crowbarring of power supply output voltage as seen in Figure 5.25. In practice, stray inductance and the presence of a transmission line can generate a HV pulse of high amplitude traveling back on the coaxial cable in the event of a klystron arc as seen in Figure 5.26 (light blue trace). The amplitude of such a pulse exceeds the ratings of the doubler capacitors and can cause damage to the power supply.

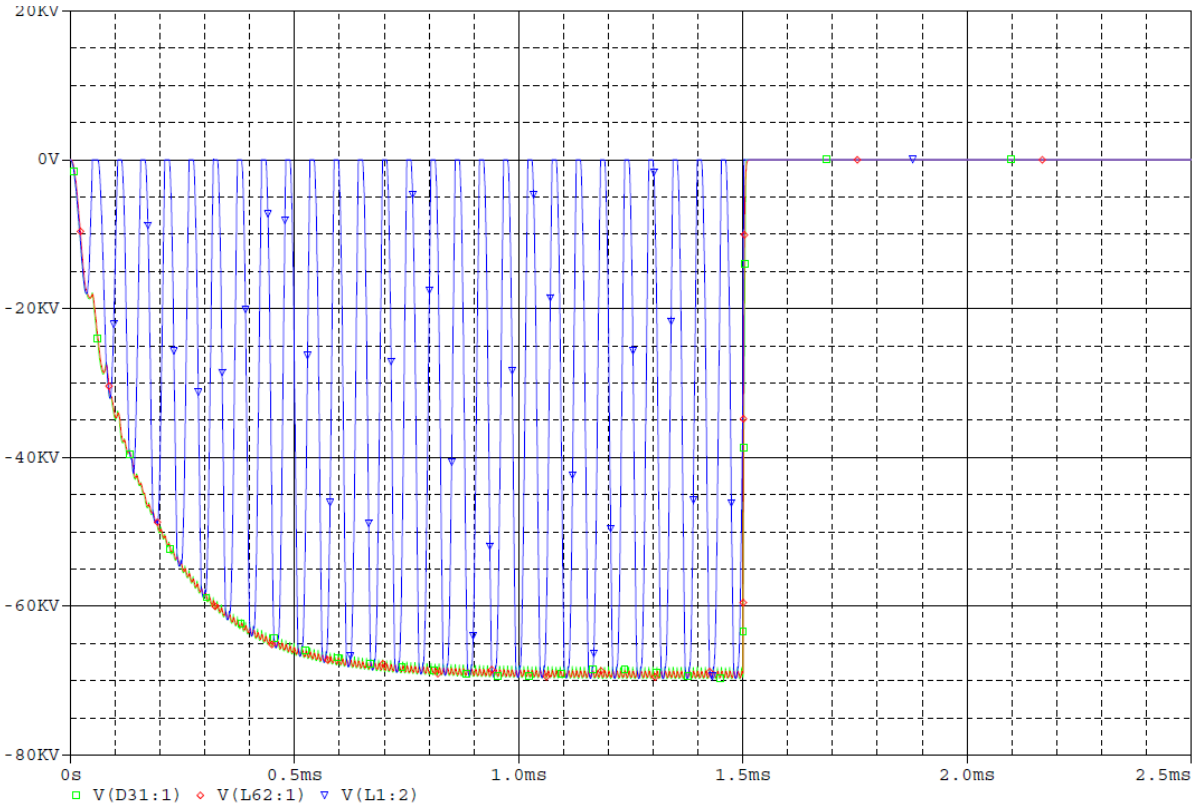


Figure 5.25 Simulation of voltages during spark gap firing.

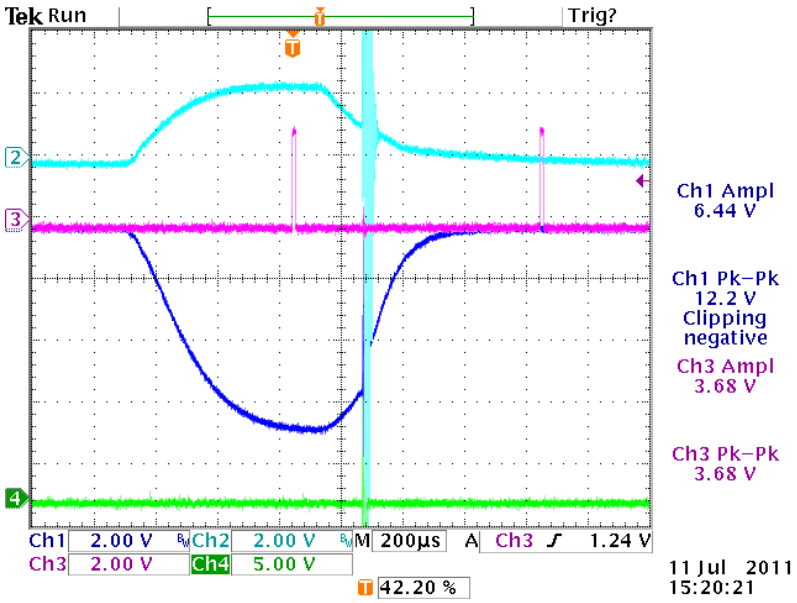


Figure 5.26 HV pulse caused by arc without snubber.

After the addition of an RL snubber circuit and associated spark gap allows safe operation of the system in the event of a power supply arc. A spark gap triggering event is shown in Figure 5.27. Upon firing of the spark gap, rapid crowbarring of the voltage across the klystron tube is observed, limiting energy that would be dissipate into the tube's cathode, potentially causing damage. An HV pulse is still reflected up the transmission line causing the voltage at the power supply terminals to swing from the previous negative value to a positive value of equal amplitude, however due to the presence of the snubber, the amplitude of the pulse is low enough to be safely tolerated by the system. Voltage on a transformer primary during a crowbar event is shown in Figure 5.28 demonstrating the rapid de-Qing of the secondary resonator observable as a reduction in primary current. Also visible is the cutoff of IGBT switching when the spark gap current sensor sends a fault condition pulse to the control system, cutting off the IGBT gate drive signals. A spark gap triggering event due

to an overvoltage condition is shown in Figure 5.29 and Figure 5.30, producing nearly identical effects.

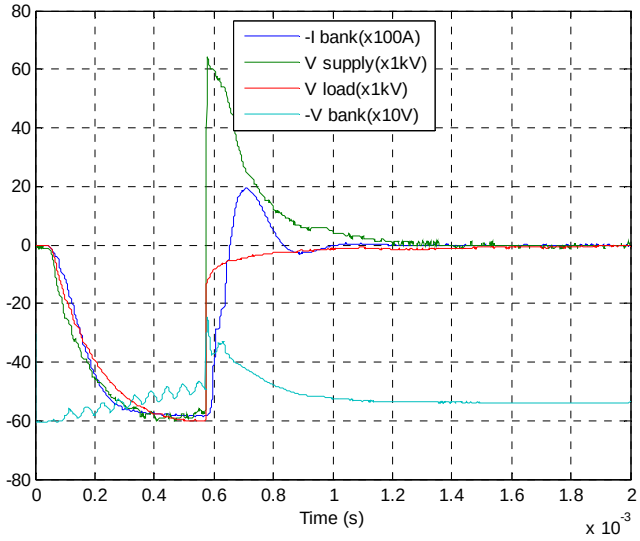


Figure 5.27 Voltages during spark gap remote trigger

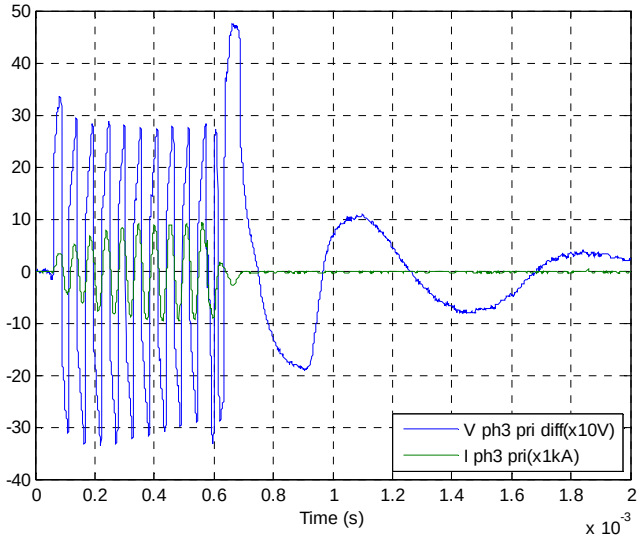


Figure 5.28 Phase current and voltages during spark gap remote trigger

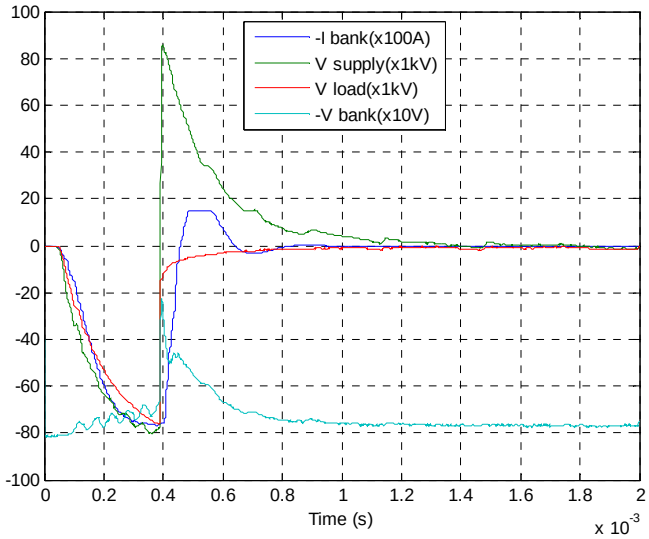


Figure 5.29 Voltages during spark gap overvoltage trigger

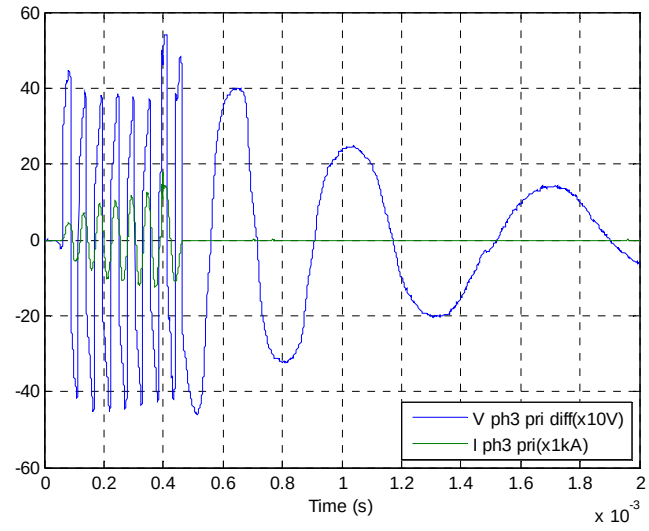


Figure 5.30 Phase current and voltages during spark gap overvoltage trigger

5.6 Summary

This chapter presented the results from testing the power supply. Numerical spice simulation models for the power supply and simulated results were compared to measured data from power supply operation and found to be comparable. Open loop testing of the power supply was used to generate a linearized model of power supply voltage boost as a function of switching frequency. Section three presents results from testing the feedback control system to compensate for capacitor bank droop and stabilize output voltage. Section four presents models and testing of several filters to reduce harmonics. Section five presents simulation and testing of arc faults, operation of the spark gap and installation of the snubber circuit. Section six summarizes the chapter.

Chapter 6 Conclusions and Recommendations

This thesis documents research on the design and construction of a three phase resonant power converter for driving klystron tubes. This chapter presents a summary of the research, provides conclusions and recommendations for the design of resonant power converters, and describes future research that will take place on this power supply.

6.1 Summary of Research

Chapter one presented a review of state of the art designs of high power SMPS design, resonant topologies, and soft switching. A review of three phase resonant power supplies constructed at LANL and E2V technologies

Chapter two presented the requirements and design constraints for construction of the power supply including available parts and constraints on integration into existing experiments. Requirements on the power supply include output voltage and current capabilities, fault tolerance, output voltage range, output stability, pulse duration, serviceability, safety, tolerance for voltage droop on the capacitor bank, and external control of parameters. Design constraints include the use of certain donated parts including transformer cores, IGBTs, rectifier diode stacks, and capacitor banks which were either donated to the project or were available as surplus.

Chapter three presented the design of the power supply's power electronic components, transformers, rectifier, filtering system and safety systems as well as comparison of experimental data to numerical simulations and the development of analytical models to provide the basis for future designs.

Chapter four presented the design and construction of the power supplies control system. This chapter covered the microcontroller, external electronics and the design of the operating code.

Chapter five presented results from power supply testing and comparison to numerical results from spice simulations. Models of the transformer, rectifiers, filters are presented and

compared to simulation. Open loop testing of the power supply is presented and compared to spice simulations. Testing of the feedback control system and stabilization of output voltage during drooping capacitor bank in presented. Testing of a crowbar spark gap and associated LR snubber system is presented and compared to simulation.

6.2 Conclusions

Design and analysis has been performed on a three phase resonant power supply utilizing loosely coupled resonant transformers with a boost ratio that significantly exceeds the turns ratio with the capability of high voltage, high current output. It was determined that the use of nano-crystalline iron core allowed low loss, high frequency operation while providing high permeability. Use of such transformers allows extremely high power density when compared to equivalently rated 60Hz systems using soft iron cores.

A successful design of a feedback control system has been produced. The control system utilizes a low cost, high performance microcontroller with integrated SMPS control functionality to stabilize output voltage as the capacitor bank voltage decreases by adjusting switching frequency. It was determined that feedforward control from capacitor bank voltage is sufficient to stabilize output voltage, however the addition of feedback off of the output voltage would improve stability.

Several designs of noise and ripple filters were tested. It was found that use of a harmonic filter provide excellent reduction 6th harmonic noise, however effectiveness decreases as switching frequency moves away from resonance. It was determined that the addition of a pi filter on the output greatly reduced higher harmonic noise and ripple but produces an anomalous increase in third harmonic ripple. The addition of a pi filter does not greatly effect rapid rise time at the beginning of the pulse and only adds minimal stored energy.

The design of several safety systems for driving klystron tubes has been shown to prevent damage to the power supply in the event of an internal arc. A snubber circuit has been shown to reduce high voltage transients on the coaxial cable connecting the power supply to the load to sufficiently low amplitudes, preventing damaging the supply. The design of an optically or di/dt triggered spark gap allows rapid crowbarring of klystron voltage and provides an optical signal that cuts output power from the supply. Use of the spark gap has been shown to successfully de-Q the secondary resonators, interrupting power transfer and preventing damage to the power supply.

6.3 Future Work

Future work will focus on further analysis of soft switching, control system capabilities, and noise reduction.

Topics on soft switching will include measurements of the power supply's waveforms using instrumentation with greater numbers of channels to simultaneously record all three phases at once, analytical models of the resonant transformer including losses and non-idealities, and use of phase shifting techniques to allow soft switching off of resonance.

Topics on improving control system capabilities will include integration of feedback and feedforward control techniques to actively monitor output voltage, improvement in control loop functionality including techniques such as sampling analog inputs between switching events, implementation of digital filters for improved noise immunity, and improved resistance to electromagnetic noise during arcs and faults.

Topics on noise reduction will include design of low stored energy output filters, balancing secondary amplitude to reduce any 1st, 2nd, and 4th harmonic noise, potential frequency dithering of the IGBT switching to spread out harmonics, and use of actively tunable harmonic filters by using power electronic techniques to vary effective inductance.

Bibliography

[1] W. A. Reass, D. M. Baca, and R. F. Gribble "DESIGN AND APPLICATION OF POLYPHASE RESONANT CONVERTER-MODULATORS" 4th CW and High Average Power Workshop LA-UR-06-3134

[2] O.H. Stielau, G.A. Covic "Design of loosely coupled inductive power transfer systems" International Conference on Power System Technology, 2000. Proceedings. PowerCon 2000.

[3] W.A. Reass, J.D. Doss, R.F. Gribble "A 1 MEGAWATT POLYPHASE BOOST CONVERTER-MODULATOR FOR KLYSTRON PULSE APPLICATION" Los Alamos National Laboratoy, PO. Box 1663, Los Alamos, NM87544, USA

[4] W. A. Reass, D. M. Baca, and R. F. Gribble "DESIGN AND APPLICATION OF POLYPHASE RESONANT CONVERTER-MODULATORS" 4th CW and High Average Power Workshop LA-UR-06-3134

[5] W. A. Reass, et.al. "DESIGN, STATUS, AND FIRST OPERATIONS OF THE SPALLATION NEUTRON SOURCE POLYPHASE RESONANT CONVERTER MODULATOR SYSTEM*" Proceedings of the 2W3 Particle Accelerator Conference

[6] W. A. Reass et.al "High-Frequency Multimegawatt Polyphase Resonant Power Conditioning" IEEE TRANSACTIONS ON PLASMA SCIENCE, VOL. 33, NO. 4, AUGUST 2005

[7] W.A. Reass, J.D. Doss, M.G. Fresquez, D.A. Miera, J.S. Mirabal, P.J. Tallerico "A PROOF-OF-PRINCIPLE POWER CONVERTER FOR THE SPALLATION NEUTRON SOURCE RF SYSTEM" Proceedings of the 1999 Particle Accelerator Conference, New York, 1999

[8] M. J. Bland, J. C. Clare, P. W. Wheeler, R. Richardson "A 25KV, 250KW MULTIPHASE RESONANT POWER CONVERTER FOR LONG PULSE APPLICATIONS" Pulsed Power Conference, 2007 16th IEEE International

[9] M. J. Bland, J. C. Clare1, P. W. Wheeler, J. S. Pryzbyla "A HIGH POWER RF POWER SUPPLY FOR HIGH ENERGY PHYSICS APPLICATIONS" Proceedings of 2005 Particle Accelerator Conference, Knoxville, Tennessee

[10] R.A. Cooper et.al. "DEVELOPMENT OF LOW INDUCTANCE HIGH POWER PLASTIC CASE CAPACITORS FOR THE SNS POLYPHASE RESONANT CONVERTER-MODULATOR" GENERAL ATOMICS ENERGY PRODUCTS Engineering Bulletin

[11] H. A. Wheeler "SIMPLE INDUCTANCE FORMULAS FOR RADIO COILS" Proceedings of the Institute of Radio Engineers 1928

[12] H. A. Wheeler "Formulas for the Skin Effect" Proceedings of the I.R.E. 1942

Appendix

A CM1200HB-66H IGBT

CM1200HB-66H

HIGH POWER SWITCHING USE
INSULATED TYPE

2nd-Version HVIGBT (High Voltage Insulated Gate Bipolar Transistor) Modules

MAXIMUM RATINGS (T_j = 25°C)

Symbol	Item	Conditions	Ratings	Unit
V_{CES}	Collector-emitter voltage	V_{GE} = 0V	3300	V
V_{GES}	Gate-emitter voltage	V_{CE} = 0V	±20	V
I_C	Collector current	DC, T_C = 100°C	1200	A
I_{CM}		Pulse (Note 1)	2400	A
I_E (Note 2)	Emitter current		1200	A
I_{EM} (Note 2)		Pulse (Note 1)	2400	A
P_C (Note 3)	Maximum collector dissipation	T_C = 25°C, IGBT part	15600	W
T_j	Junction temperature	—	–40 ~ +150	°C
T_{stg}	Storage temperature	—	–40 ~ +125	°C
V_{iso}	Isolation voltage	Charged part to base plate, rms, sinusoidal, AC 60Hz 1min.	6000	V
—	Mounting torque	Main terminals screw M8	6.67 ~ 13.00	N·m
		Mounting screw M6	2.84 ~ 6.00	N·m
		Auxiliary terminals screw M4	0.88 ~ 2.00	N·m
—	Mass	Typical value	2.2	kg

ELECTRICAL CHARACTERISTICS (T_j = 25°C)

Symbol	Item	Conditions			Limits			Unit
					Min	Typ	Max	
I_{CES}	Collector cutoff current	V_{CE} = V_{CES}, V_{GE} = 0V			—	—	15	mA
$V_{GE(th)}$	Gate-emitter threshold voltage	I_C = 120mA, V_{CE} = 10V			4.5	6.0	7.5	V
I_{GES}	Gate-leakage current	V_{GE} = V_{GES}, V_{CE} = 0V			—	—	0.5	µA
$V_{CE(sat)}$	Collector-emitter saturation voltage	T_j = 25°C	I_C = 1200A, V_{GE} = 15V	(Note 4)	—	3.80	4.94	V
		T_j = 125°C			—	4.00	—	
C_{ies}	Input capacitance	V_{CE} = 10V, V_{GE} = 0V			—	180	—	nF
C_{oes}	Output capacitance				—	18.0	—	nF
C_{res}	Reverse transfer capacitance				—	5.4	—	nF
Q_G	Total gate charge	V_{CC} = 1650V, I_C = 1200A, V_{GE} = 15V			—	8.6	—	µC
$t_{d(on)}$	Turn-on delay time	V_{CC} = 1650V, I_C = 1200A, V_{GE1} = V_{GE2} = 15V, R_G = 1.6Ω, Resistive load switching operation			—	—	1.60	µs
t_r	Turn-on rise time				—	—	2.00	µs
$t_{d(off)}$	Turn-off delay time				—	—	2.50	µs
t_f	Turn-off fall time				—	—	1.00	µs
V_{EC} (Note 2)	Emitter-collector voltage	I_E = 1200A, V_{GE} = 0V			—	2.80	3.64	V
t_{rr} (Note 2)	Reverse recovery time	I_E = 1200A, die / dt = –2400A / µs (Note 1)			—	—	1.40	µs
Q_{rr} (Note 2)	Reverse recovery charge				—	400	—	µC
$R_{th(j-c)Q}$	Thermal resistance	Junction to case, IGBT part			—	—	0.008	K/W
$R_{th(j-c)R}$		Junction to case, FWDi part			—	—	0.016	K/W
$R_{th(c-f)}$	Contact thermal resistance	Case to fin, conductive grease applied			—	0.006	—	K/W

Note 1. Pulse width and repetition rate should be such that the device junction temp. (T_j) does not exceed T_{jmax} rating.
2. I_E, V_{EC}, t_{rr}, Q_{rr} & die/dt represent characteristics of the anti-parallel, emitter to collector free-wheel diode.
3. Junction temperature (T_j) should not increase beyond 150°C.
4. Pulse width and repetition rate should be such as to cause negligible temperature rise.

B Hipotronics 8XX Series Power Supplies

5kW High Power DC Supplies

General	801-5A	825-200	8100-50	8150-33.3	8300-16.8
Input Voltage	208/230V 15A 50/60Hz Three Phase	208/220V 25A 50/60Hz Single Phase			
Output Voltage	0 – 1kV DC	0 – 25kV DC	0 – 100kV DC	0 – 150kV DC	0 – 300kV DC
Output Current	5A	200mA	50mA	33.3mA	16.8mA
Output Polarity	Positive or Negative output in respect to ground				
Metering	4.5" analog meters, ±2% full scale accuracy				
Regulation	10% No Load to Full Load		15% No Load to Full Load		
Ripple	5% rms (1% Optional)		2% rms		
Control Dimensions	49"H x 23"W x 24"D (1245mmx584mmx610mm)		17.5"H x 19"W x 17"D (445mmx483mmx432mm)		
Control Weights	Net 450lbs (204kg)		Net 105lbs (48kg)		
High Voltage Dimensions	In Control Section		32"Hx21"diameter (813mmx533mm)	60"H x 29"diameter (1524mmx737mm)	
High Voltage Weight	In Control Section		Net 950lbs (431kg)	Net 2800lbs (1270kg)	Net4700lbs (2132kg)

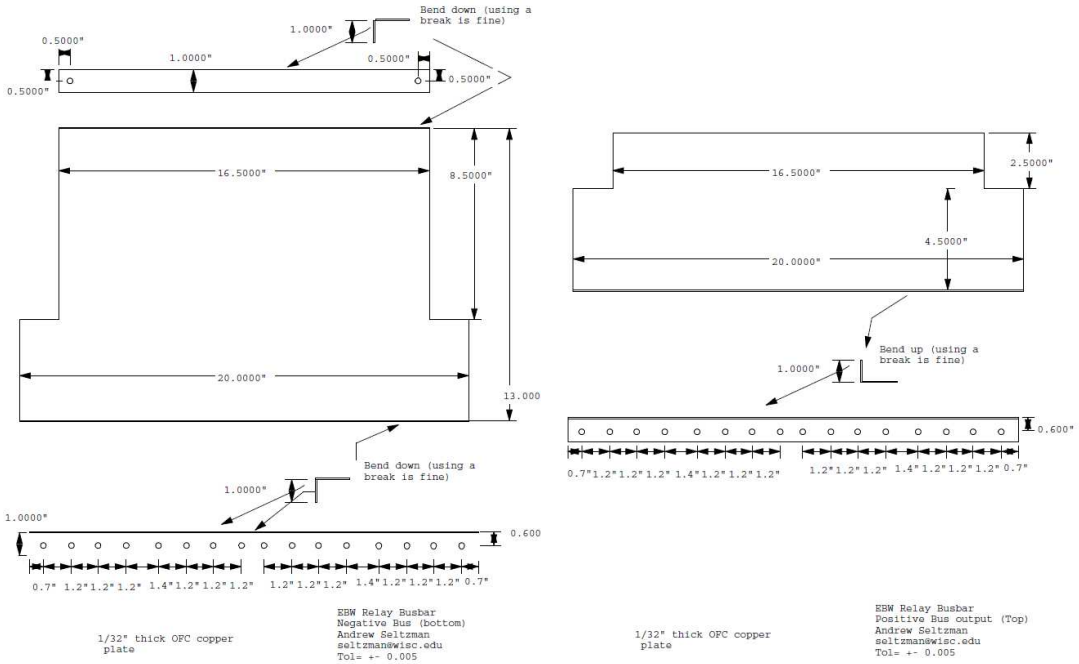
Note: Other Output Ratings Available, Consult Factory with Your Requirements

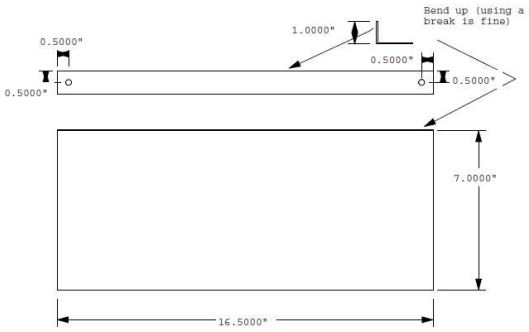
C Matlab code for calculation voltage droop

```
clear all;

Vi=900;
Vout=80E3;
R=1800;
Iout=Vout/R;
N=[1:-0.1:0.6];
C=0.3;

Tpulse=[0:0.1:10]/1E3;

Vfarr=[];
Tarr=[];
Legendarr=cell(length(N),1);    %prealocate cell array for legend

for i=[1:length(N)]
%calculate final voltage
Neff=N(i);
Vf=Vi.*sqrt(1-Vout.*Iout.*Tpulse./(Neff.*C.*Vi.^2/2));
Vfarr=[Vfarr;Vf];
Tarr=[Tarr;Tpulse];
%build cell array of legend strings
Legendarr{i}=['Eff= ', num2str(Neff)];

end
```

```
%plot voltage droop vs pulse time
figure(1)
plot(Tarr'.*1000,Vfarr','LineWidth',2)
xlabel('Pulse Time (ms)')
ylabel('Final Voltage (V)')
grid on
legend(Legendarr)
title(['Initial Voltage= ', num2str(Vi),...
    'V, C= ' num2str(C), 'F'])
```

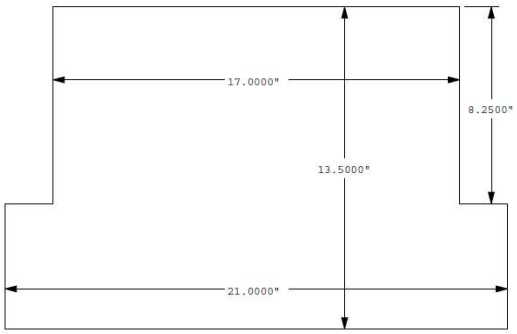
D Low inductance relay CAD designs





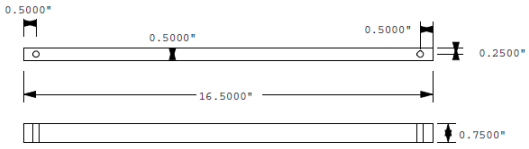
1/32" thick OFC copper plate

EBW Relay Busbar
Positive Bus (Top)
Andrew Seltzman
seltzman@wisc.edu
Tol= +- 0.005



1/16" G10-FR4 fiberglass

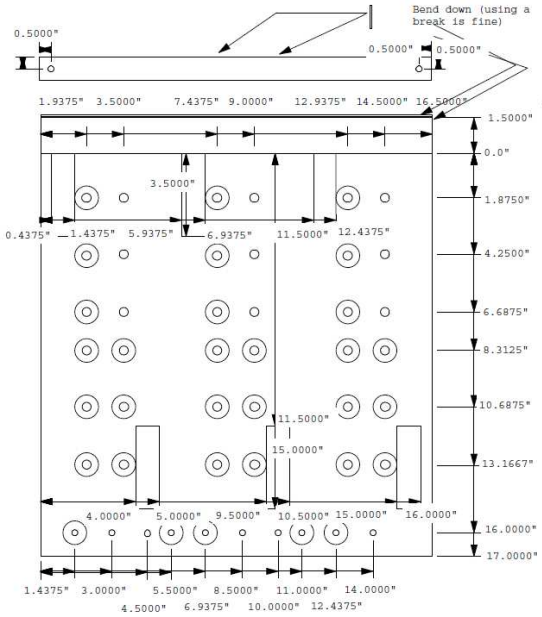
EBW Relay Busbar
Insulator
Andrew Seltzman
seltzman@wisc.edu
Tol= +- 0.005

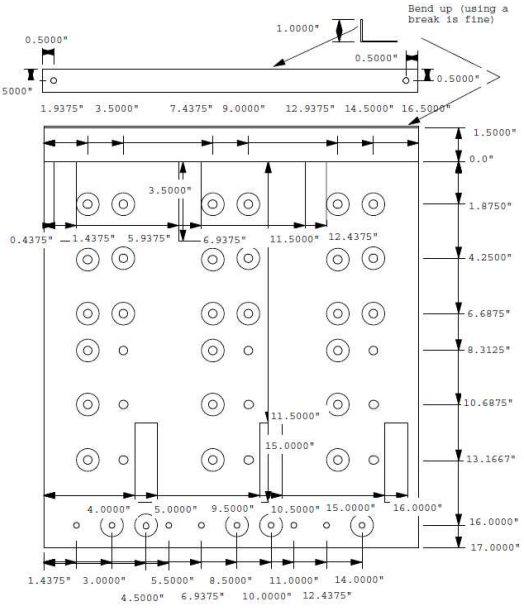


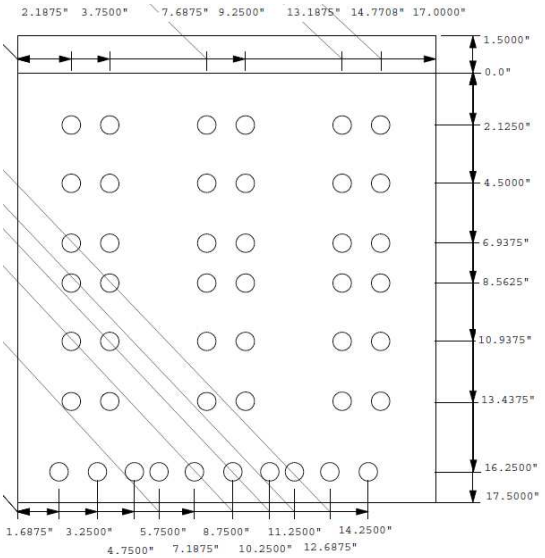
6061 Aluminum 0.5"x0.75"

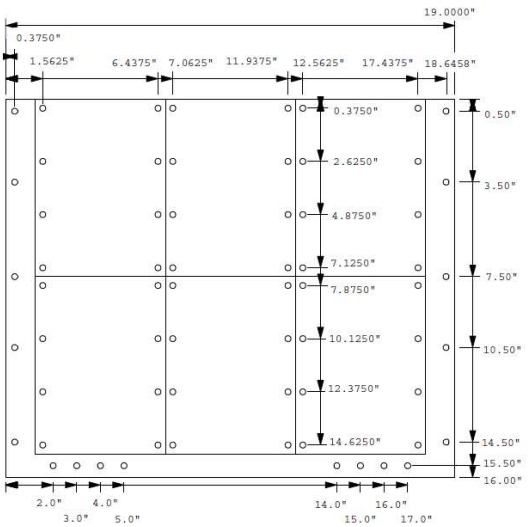
EBW Relay Busbar Clamp
Negative Bus (bottom)
Andrew Seltzman
seltzman@wisc.edu
Tol= +- 0.005

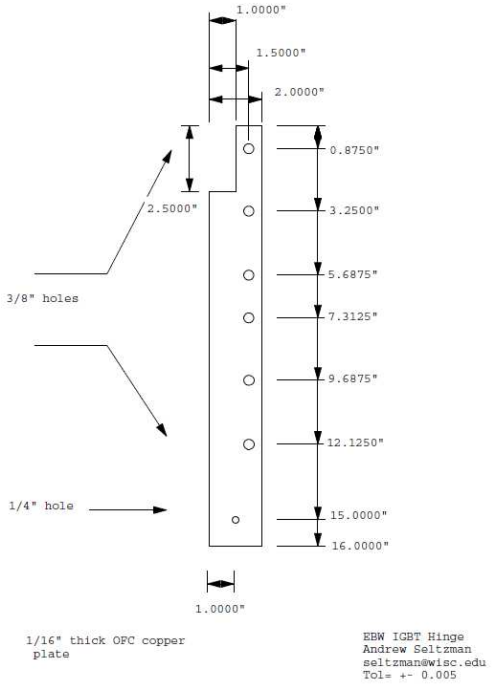
E IGBT busbar CAD designs











F CT concepts IGBT driver.

Absolute Maximum Ratings

Parameter	Remarks	Min	Max	Units
Supply voltage V_{DC}	VDC to GND (Note 1)	0	16	V
Gate peak current I_{out}	Note 8	-36	+36	A
Average supply current I_{DC}	Note 3		500	mA
Output power DC/DC converter	Notes 3,12		5	W
Switching frequency	Note 12		8.5	kHz
Test voltage (50Hz/1min)	Primary to output (Note 15)		6000	$V_{AC(eff)}$
DC-link voltage	Note 5		2200	V
Operating temperature	Note 12	-40	+85	°C
Storage temperature		-40	+90	°C

Electrical Characteristics

All data refer to +25°C and V_{DC} = 15V unless otherwise specified

Power supply	**Remarks**	**Min**	**Typ.**	**Max**	**Units**
Nominal supply voltage V_{DC}	VDC to GND (Note 1)	14.5	15	15.5	V
Supply current I_{DC}	Without load (Note 2)				
	Standard		120		mA
	Opt. 01		125		mA
Efficiency η	Internal DC/DC converter		85		%
Turn-on threshold V_{th}	Note 4		13		V
Hysteresis on/off	Note 4		0.6		V
Coupling capacitance C_{io}	Primary to output		15		pF
Short-circuit protection	**Remarks**	**Min**	**Typ.**	**Max**	**Units**
V_{ce} monitoring threshold	Between aux. terminals	50		60	V
Response time	3-level mode (Note 11)		8.5	9	µs
Response time	2-level mode (Note 6)		9.5	10	µs
Blocking time	2-level mode (Note 7)		1		s
Timing characteristics	**Remarks**	**Min**	**Typ.**	**Max**	**Units**
Turn-on delay $t_{pd(on)}$	Note 13		400		ns
Turn-off delay $t_{pd(off)}$	Note 13		550		ns
Output rise time $t_{r(out)}$	Note 9		15		ns
Output fall time $t_{f(out)}$	Note 9		20		ns
Acknowledge delay time	At status output (Note 14)		380		ns
Acknowledge pulse width	At status output	0.6		1.8	µs
Gate output	**Remarks**	**Min**	**Typ.**	**Max**	**Units**
Turn-on gate resistor $R_{g(on)}$	Note 8		2.2		Ω
Turn-off gate resistor $R_{g(off)}$	Note 8		3.7		Ω
Aux. gate capacitor C_{ge}			220		nF
Electrical insulation	**Remarks**	**Min**	**Typ.**	**Max**	**Units**
Operating voltage (Note 10)	Continuous or repeated			3300	V
Test voltage (50Hz/1min)	Primary to output (Note 15)			6000	$V_{AC(eff)}$
Partial discharge extinction volt.	IEC1287 / <10pC	2600			$V_{AC(eff)}$
Creepage distance	Primary to output	21			mm

G Matlab code for resonant transformer plots

```
clear all;


load('N156_R1800_F.txt');  %
load('N146_R1800_F.txt');  %
load('N136_R1800_F.txt');  %
load('N126_R1800_F.txt');  %
load('N116_R1800_F.txt');  %
load('N106_R1800_F.txt');  %
load('N096_R1800_F.txt');  %
load('N086_R1800_F.txt');  %
load('N076_R1800_F.txt');  %

load('N156_R820_F.txt');  %
load('N146_R820_F.txt');  %
load('N136_R820_F.txt');  %
load('N126_R820_F.txt');  %
load('N116_R820_F.txt');  %
load('N106_R820_F.txt');  %
load('N096_R820_F.txt');  %
load('N086_R820_F.txt');  %
load('N076_R820_F.txt');  %

load('N156_R.txt');  %
load('N146_R.txt');  %
load('N136_R.txt');  %
load('N126_R.txt');  %
load('N116_R.txt');  %
load('N106_R.txt');  %
load('N096_R.txt');  %
load('N086_R.txt');  %
load('N076_R.txt');  %

%turn number
N=[76,86,96,106,116,126,136,146,156];
Nrange=[76:156];

%resonant frequency at a given turn number
ResN=[N076_R1800_F(N076_R1800_F(:,4)==max(N076_R1800_F(:,4)),1),...
      N086_R1800_F(N086_R1800_F(:,4)==max(N086_R1800_F(:,4)),1),...
      N096_R1800_F(N096_R1800_F(:,4)==max(N096_R1800_F(:,4)),1),...
      N106_R1800_F(N106_R1800_F(:,4)==max(N106_R1800_F(:,4)),1),...
      N116_R1800_F(N116_R1800_F(:,4)==max(N116_R1800_F(:,4)),1),...
      N126_R1800_F(N126_R1800_F(:,4)==max(N126_R1800_F(:,4)),1),...
      N136_R1800_F(N136_R1800_F(:,4)==max(N136_R1800_F(:,4)),1),...
      N146_R1800_F(N146_R1800_F(:,4)==max(N146_R1800_F(:,4)),1),...
```

```
      N156_R1800_F(N156_R1800_F(:,4)==max(N156_R1800_F(:,4)),1)];

%maximum boost ratio at 1800 ohm load
MaxBoost1800=[
      max(N076_R1800_F(:,4)),...
      max(N086_R1800_F(:,4)),...
      max(N096_R1800_F(:,4)),...
      max(N106_R1800_F(:,4)),...
      max(N116_R1800_F(:,4)),...
      max(N126_R1800_F(:,4)),...
      max(N136_R1800_F(:,4)),...
      max(N146_R1800_F(:,4)),...
      max(N156_R1800_F(:,4))];

%Maximum boost ratio at 820 ohm load
MaxBoost820=[
      max(N076_R820_F(:,4)),...
      max(N086_R820_F(:,4)),...
      max(N096_R820_F(:,4)),...
      max(N106_R820_F(:,4)),...
      max(N116_R820_F(:,4)),...
      max(N126_R820_F(:,4)),...
      max(N136_R820_F(:,4)),...
      max(N146_R820_F(:,4)),...
      max(N156_R820_F(:,4))];

%theoretical inductance of the secondary wining
mu0=4*pi*1e-7;      %vacuum permiability
mur=50E3;           %relative permiability of core
Nsec=136;           %turns on secondary for this given plot
D22awg=0.644E-3;    %diameter of 22AWG wire used on secondary in meters
Radsec=6.5.*0.0254/2;   %Radius of secondary winding in meters (6.5"
diameter)
A=pi*Radsec.^2;         %area of secondary winding in m^2
Acore=(1.75*0.0254)*(2.5*0.0254);   %area of iron core in m^2
length136=D22awg.*Nsec;             %axial length of secondary winding
%inductance using long solonoid approximation
Lsec136th=(mu0.*(Nsec.^2).*A./length136).*(1/2)
%inductance using short solonoid (wheeler 1942 eq50)
Lsec136th2=10.*pi.*mu0.*Nsec.^2.*Radsec.^2./...
    (9.*Radsec+10.*length136).*(1/2)
%compensated for flu excluded from iron core area
Lsec136th3=10.*mu0.*Nsec.^2.*(pi.*Radsec.^2-Acore)./...
    (9.*Radsec+10.*length136).*(1/2)

%transfer function model of transformer
Lsec136=1.36E-3;  %1.36mH for N=135
Rsec=1.3;   %secondary winding resistance
            %(both cores in parallel, confirmed at 1.3ohm)
C=0.05E-6;  %0.05uF parallel resonant capacitor
R=1800;     %load resistance (measured and confirmed at 1800+-5ohm)
Nr=13.6;    %turns ratio
```

```
%transfer function frequency sweep
freq=logspace(log10(1000),log10(33000),1000);
w=2*pi*freq;
Zs=j*w*Lsec136th3+Rsec+(R./(j*w*C))./(R+1./(j*w*C));
%transfer function for transformer
Hw=Nr*abs(((R./(j*w*C))./(R+1./(j*w*C)))./Zs);

%transfer function load resistance sweep
Rsw=logspace(log10(1),log10(60000),1000);
freqR=N136_R1800_F(N136_R1800_F(:,4)==max(N136_R1800_F(:,4)),1).*1000;
w=2*pi*freqR;
ZsR=j*w*Lsec136+Rsec+(Rsw./(j*w*C))./(Rsw+1./(j*w*C));
%transfer function for transformer
Hr=Nr*abs(((Rsw./(j*w*C))./(Rsw+1./(j*w*C)))./ZsR);

%theoretical resonance vs turns
%Wheeler formula
%with compensation for magnetic flux excluded by shorted primary
%multiply by 1/2 since 2 coils in parallel
Larr=10.*mu0.*Nrange.^2.*(pi.*Radsec.^2-Acore)./...
    (9.*Radsec+10.*D22awg.*Nrange).*(1/2);
Fres=(1/(2*pi))*(sqrt(1./(Larr.*C)-1./((R.*C).^2)));

%measured inductance at a turns ratio
Lmeas=1./(((ResN.*(1000).*(2*pi)).^2+1./((R.*C).^2)).*C);

%models of inductance
%Wheeler formula
%multiply by 1/2 since 2 coils in parallel
LarrFullA=10.*mu0.*Nrange.^2.*(pi.*Radsec.^2)./...
    (9.*Radsec+10.*D22awg.*Nrange).*(1/2);
%simple long solonoid
%multiply by 1/2 since 2 coils in parallel
LarrLong=(mu0.*(Nrange.^2).*A./(D22awg.*Nrange)).*(1/2);
%simple long solonoid compensated area
%multiply by 1/2 since 2 coils in parallel
LarrLongCompA=(mu0.*(Nrange.^2).*((A-Acore))./(D22awg.*Nrange)).*(1/2);

%theoretical max boost
w=2*pi*(Fres);
%for 1800 ohm load
R=1800;
Zsmax1800=j.*w.*Larr+Rsec+(R./(j*w*C))./(R+1./(j*w*C));
 %transfer function for transformer
Hwmax1800=(Nrange./10).*abs(((R./(j*w*C))./(R+1./(j*w*C)))./Zsmax1800);
%for 820 ohm load
R=820;
Zsmax820=j.*w.*Larr+Rsec+(R./(j*w*C))./(R+1./(j*w*C));
%transfer function for transformer
Hwmax820=(Nrange./10).*abs(((R./(j*w*C))./(R+1./(j*w*C)))./Zsmax820);
```

```
%plot
figure(1)
loglog(N156_R(:,1),N156_R(:,5),'-*',N136_R(:,1),N136_R(:,5),'-*',...
       N076_R(:,1),N076_R(:,5),'-*',Rsw,Hr,'LineWidth',2)
xlabel('Load Resistance (ohms)')
ylabel('Boost Ratio at Resonant Frequency')
grid on
legend('N=156 Measured','N=136 Measured',...
    'N=076 Measured','N=136 Transfer Fcn','Location','NorthWest')

%plot resonant frequency vs turns
figure(2)
plot(N,ResN,'-*',Nrange,Fres/1E3,'LineWidth',2)
xlabel('Secondary Turns')
ylabel('Resonant Frequency (kHz)')
grid on
legend('Measured Resonant F','Transfer Fcn Resonant F',...
    'Location','NorthEast')

%plot resonant frequency vs turns
figure(3)
plot(N,MaxBoost1800,'-*',Nrange,Hwmax1800,N,MaxBoost820,'-*',...
    Nrange,Hwmax820,'LineWidth',2)
xlabel('Secondary Turns')
ylabel('Boost Ratio')
grid on
legend('R=1800 ohm Max Boost',...
       'R=1800 ohm Max Transfer Fcn',...
       'R=820 ohm  Max Boost',...
       'R=820 ohm  Max Transfer Fcn','Location','NorthWest')

%plot boost vs freq for n=136
figure(4)
plot(N136_R1800_F(:,1),N136_R1800_F(:,4),'-*',...
    freq/1000,Hw,'LineWidth',2)
xlabel('Frequency (kHz)')
ylabel('Boost Ratio')
grid on
legend('N=136 Measured','N=136 Transfer Fcn','Location','NorthWest')

%plot boost vs freq for n=all
figure(5)
plot(N156_R1800_F(:,1),N156_R1800_F(:,4),...
     N146_R1800_F(:,1),N146_R1800_F(:,4),...
     N136_R1800_F(:,1),N136_R1800_F(:,4),...
     N126_R1800_F(:,1),N126_R1800_F(:,4),...
     N116_R1800_F(:,1),N116_R1800_F(:,4),...
     N106_R1800_F(:,1),N106_R1800_F(:,4),...
     N096_R1800_F(:,1),N096_R1800_F(:,4),...
     N086_R1800_F(:,1),N086_R1800_F(:,4),...
     N076_R1800_F(:,1),N076_R1800_F(:,4),'LineWidth',2)
xlabel('Frequency (kHz)')
ylabel('Boost Ratio')
```

```
grid on
legend('N=156 R=1800ohm','N=146 R=1800ohm','N=136 R=1800ohm',...
    'N=126 R=1800ohm','N=116 R=1800ohm','N=106 R=1800ohm',...
    'N=096 R=1800ohm','N=086 R=1800ohm','N=076 R=1800ohm',...
    'Location','NorthWest')

%plot boost vs freq for n=all
figure(6)
plot(N156_R820_F(:,1),N156_R820_F(:,4),...
     N146_R820_F(:,1),N146_R820_F(:,4),...
     N136_R820_F(:,1),N136_R820_F(:,4),...
     N126_R820_F(:,1),N126_R820_F(:,4),...
     N116_R820_F(:,1),N116_R820_F(:,4),...
     N106_R820_F(:,1),N106_R820_F(:,4),...
     N096_R820_F(:,1),N096_R820_F(:,4),...
     N086_R820_F(:,1),N086_R820_F(:,4),...
     N076_R820_F(:,1),N076_R820_F(:,4),'LineWidth',2)
xlabel('Frequency (kHz)')
ylabel('Boost Ratio')
grid on
legend('N=156 R=820ohm','N=146 R=820ohm','N=136 R=820ohm',...
    'N=126 R=820ohm','N=116 R=820ohm','N=106 R=820ohm',...
    'N=096 R=820ohm','N=086 R=820ohm','N=076 R=820ohm',...
    'Location','NorthWest')

%plot inductance vs turns
figure(7)
plot(N,Lmeas,'-*',Nrange,Larr,Nrange,LarrFullA,...
    Nrange,LarrLong,Nrange,LarrLongCompA,'LineWidth',2)
xlabel('Secondary Turns')
ylabel('Inductance (H)')
grid on
legend('Measured','Wheeler Compensated Area','Wheeler Full Area',...
    'Long Solenoid','Long Solenoid Compensated
Area','Location','NorthWest')
```

H Matlab code for plotting boost ratio at varying frequency

```
clear all;
 %load open loop ps output at 200v on the bank
load('open_loop_200bank.txt');
fmod=open_loop_200bank(:,1);    %frequency in khz
vboost=open_loop_200bank(:,2)*1000/200; %kv out / 200v on the bank
ptper=235.*(40000./(fmod.*10));

%load open loop ps output at varying bank voltage
```

```
load('open_loop_vsweep_20khz.txt');
vbank=open_loop_vsweep_20khz(:,1);
vout=open_loop_vsweep_20khz(:,2);

Vin=[min(vbank):max(vbank)]; %
Vfit=0.1132*Vin;            %regression fit for vout vs vin

Freq=[min(fmod):max(fmod)]; %
BFfit=-12*Freq+343;    %regression line for boost to frequency plot

Boost=[min(vboost):max(vboost)];
PBfit=172*Boost+29706;  %regression line for PTPER to boost plot

%plot boost vs modulation frequency
figure(1)
plot(fmod,vboost,'.',Freq,BFfit,'LineWidth',2)
xlabel('Modulation Frequency (kHz)')
ylabel('Boost Ratio')
grid on
legend('Experimental Data','Fit Boost= -12*F(kHz)+343')

%plot ptper vs boost
figure(2)
plot(vboost,ptper,'.',Boost,PBfit,'LineWidth',2)
xlabel('Boost Ratio')
ylabel('PTPER')
grid on
legend('Experimental Data','Fit PTPER= 172*Boost+29706',...
    'Location','NorthWest')

%plot vout vs vbank
figure(3)
plot(vbank,vout,'*',Vin,Vfit,'LineWidth',2)
xlabel('Bank Voltage (V)')
ylabel('Output Voltage (kV)')
grid on
legend('Experimental Data','Fit Vout= 0.1132*Vbank',...
    'Location','NorthWest')
```

I Matlab code for date with differential voltage on transformer secondary.

```
clear all;
%close all;
```

```
fignum=0;

shots=[4];
channels=[1,2,3,4];

for shot=shots
for channel=channels

TOP=load(['ts',num2str(shot),'ch',num2str(channel),'.isf']);  %
BOT=load(['bs',num2str(shot),'ch',num2str(channel),'.isf']);  %
assignin('base',['t'],TOP(:,1));
assignin('base',['tch',num2str(channel)],medfilt1(TOP(:,2),50));
assignin('base',['bch',num2str(channel)],medfilt1(BOT(:,2),50));

end

%plot
fignum=fignum+1;
figure(fignum)
plot(t,tch1*3,t,tch2*100,t,tch3,t,tch4*500,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
legend('I ph3 pri(x1kA)','V ph3 pri(x10V)',...
       'V ph3 sec1(R)(x1kV)','V ph3 sec2(Y)(x1kV)','Location','SouthEast')

fignum=fignum+1;
figure(fignum)
plot(t,bch1,t,bch2*10,t,bch3*2,t,-bch4*20,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
legend('-I bank(x100A)','V supply(x1kV)',...
       'V load(x1kV)','-V bank(x10V)','Location','SouthEast')

fignum=fignum+1;
figure(fignum)
plot(t,tch3-tch4*500,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
legend('V secondary differential','Location','SouthEast')

fignum=fignum+1;
figure(fignum)
plot(t,tch3+tch4*500,t,bch2*10,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
legend('V sec wrt gnd(x1kV)','V out(x1kV)','Location','SouthEast')
```

```
end
```

J Matlab code for date with differential voltage on transformer primary.

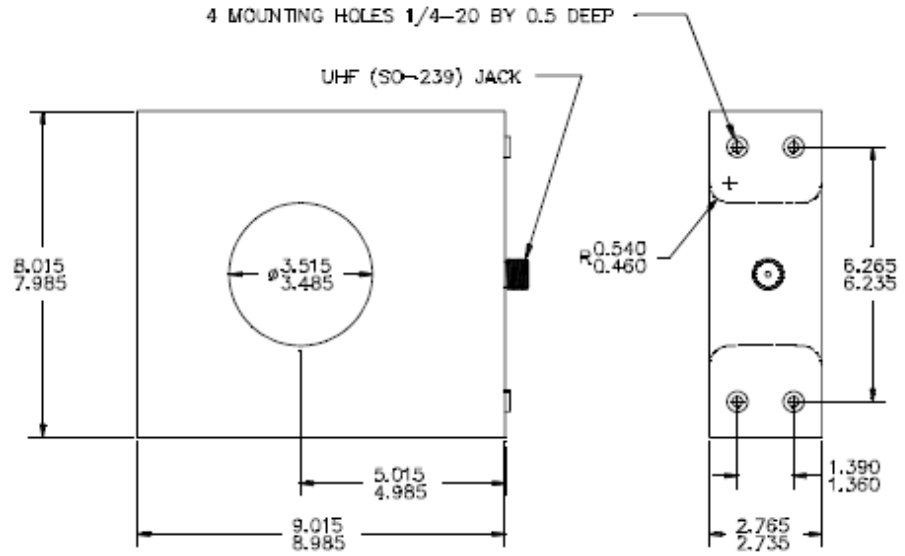
```
clear all;
close all;

fignum=0;

shots=[11];


for shot=shots

for channel=[1,2,3]
TOP=load(['ts',num2str(shot),'ch',num2str(channel),'.isf']);  %
assignin('base',['t'],TOP(:,1));
assignin('base',['tch',num2str(channel)],medfilt1(TOP(:,2),50));
end

for channel=[1,2,3,4]
BOT=load(['bs',num2str(shot),'ch',num2str(channel),'.isf']);  %
assignin('base',['bch',num2str(channel)],medfilt1(BOT(:,2),50));
end


%plot
fignum=fignum+1;
figure(fignum)
plot(t,tch1*3,t,tch2*100,t,tch3*100,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
legend('I ph3 pri(x1kA)','V ph3 pri1(x10V)',...
       'V ph3 pri2(x10V)','Location','SouthEast')

fignum=fignum+1;
figure(fignum)
plot(t,bch1,t,bch2*10,t,bch3*2,t,-bch4*20,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
```

```
legend('-I bank(x100A)','V supply(x1kV)',...
       'V load(x1kV)','-V bank(x10V)','Location','North')

fignum=fignum+1;
figure(fignum)
plot(t,-tch2*100+tch3*100,t,tch1*3,'LineWidth',1)
xlabel('Time (s)')
ylabel('')
grid on
legend('V ph3 pri diff(x10V)','I ph3 pri(x1kA)','Location','SouthEast')

fignum=fignum+1;
figure(fignum)
plot(t,10*(-tch2*100+tch3*100).*(tch1*3),...
     t,fastrms(10*((-tch2*100+tch3*100).*(tch1*3)),10000/(2*20)),...
     t,(bch1.*(-bch4*20)),t,(bch2*1E4).^2./(1800*1E3),'LineWidth',1)
xlabel('Time (s)')
ylabel('power')
grid on
legend('P ph3 (x1kW)','P ph3 rms(x1kW)',...
    'P in avg(x1kW)','P out (x1kW)','Location','SouthEast')

end
```

K Pearson transformer datasheet.

PEARSON™ CURRENT MONITOR MODEL 301X

Sensitivity	0.01 Volt/Ampere +1/-0%
Output resistance	50 Ohms
Maximum peak current	50,000 Amperes
Maximum rms current	400 Amperes
Droop rate	3.0 %/millisecond
Useable rise time	200 nanoseconds
Current time product	22 Ampere-second
Low frequency 3dB cut-off	5 Hz (approximate)
High frequency 3dB cut-off	2 MHz (approximate)
I/f figure	140 peak Amperes/Hz
Shielding	Double
Output connector	UHF (SO-239)
Operating temperature	0 to 65 °C
Weight	17.5 pounds



L AD215 datasheet

AD215–SPECIFICATIONS (Typical @ +25°C, V_S = ±15 V dc, 2 kΩ output load, unless otherwise noted.)

Parameter	Conditions	AD215AY/BY Min	Typ	Max	Units
GAIN					
Range[1]		1		10	V/V
Error	G = 1 V/V, No Load on V_{ISO}		±0.5	±2	%
vs. Temperature	0°C to +85°C		+15		ppm/°C
	–40°C to 0°C		+50		ppm/°C
vs. Supply Voltage	±(14.5 V dc to 16.5 V dc)		+100		ppm/V
vs. Isolated Supply Load[2]			+20		ppm/mA
Nonlinearity[3]					
AD215BY Grade	±10 V Output Swing, G = 1 V/V		±0.005	±0.015	%
	±10 V Output Swing, G = 10 V/V		±0.01		%
AD215AY Grade	±10 V Output Swing, G = 1 V/V		±0.01	± 0.025	%
	±10 V Output Swing, G = 10 V/V		±0.025		%
INPUT VOLTAGE RATINGS					
Input Voltage Rating	G = 1 V/V	±10			V
Maximum Safe Differential Range	IN+ or IN–, to IN COM		±15		V
CMRR of Input Op Amp			100		dB
Isolation Voltage Rating[4]	Input to Output, AC, 60 Hz				
AD215BY Grade	100% Tested[4]	1500			V rms
AD215AY Grade	100% Tested[4]	750			V rms
IMRR (Isolation Mode Rejection Ratio)	R_S ≤ 100 Ω (IN+ & IN–), G = 1 V/V, 60 Hz		120		dB
	R_S ≤ 100 Ω (IN+ & IN–), G = 1 V/V, 1 kHz		100		dB
	R_S ≤ 100 Ω (IN+ & IN–), G = 1 V/V, 10 kHz		80		dB
	R_S ≤ 1 kΩ (IN+ & IN–), G = 1 V/V, 60 Hz		105		dB
	R_S ≤ 1 kΩ (IN+ & IN–), G = 1 V/V, 1 kHz		85		dB
	R_S ≤ 1 kΩ (IN+ & IN–), G = 1 V/V, 10 kHz		65		dB
Leakage Current, Input to Output	240 V rms, 60 Hz			2	µA rms
INPUT IMPEDANCE					
Differential	G = 1 V/V		16		MΩ
Common Mode			2‖4.5		GΩ‖pF
INPUT OFFSET VOLTAGE					
Initial	@ +25°C		±0.4	±2.0	mV
vs. Temperature	0°C to +85°C		±2		µV/°C
	–40°C to 0°C		±20		µV/°C
OUTPUT OFFSET VOLTAGE					
Initial	@ +25°C, Trimmable to Zero	0	–35	–80	mV
vs. Temperature	0°C to +85°C		±30		µV/°C
	–40°C to 0°C		±80		µV/°C
vs. Supply Voltage			±350		µV/V
vs. Isolated Supply Load[2]			–35		µV/mA
INPUT BIAS CURRENT					
Initial	@ +25°C		300		nA
vs. Temperature	–40°C to +85°C		±400		nA
INPUT DIFFERENCE CURRENT					
Initial	@ +25°C		±3		nA
vs. Temperature	–40°C to +85°C		±40		nA
INPUT VOLTAGE NOISE					
Input Voltage Noise	Frequency > 10 Hz		20		nV/√Hz
DYNAMIC RESPONSE (2 kΩ Load)					
Full Signal Bandwidth (–3 dB)	G = 1 V/V, 20 V pk-pk Signal	100	120		kHz
Transport Delay[5]			2.2		µs
Slew Rate	±10 V Output Swing		6		V/µs
Rise Time	10% to 90%, ±10 V Output Swing		3		µs

Parameter	Conditions	AD215AY/BY Min	Typ	Max	Units
DYNAMIC RESPONSE (2 kΩ Load) Cont.					
Settling Time	to ±0.10%, ±10 V Output Swing		9		µs
Overshoot			1		%
Harmonic Distortion Components	@ 1 kHz		–80		dB
	@ 10 kHz		–65		dB
Overload Recovery Time	G = 1 V/V, ±15 V Drive		5		µs
Output Overload Recovery Time	G > 5		10		µs
RATED OUTPUT					
Voltage	Out HI to Out LO	±10			V
Current	2 kΩ Load	±5			mA
Max Capacitive Load			500		pF
Output Resistance			1		Ω
Output Ripple and Noise[7]	1 MHz Bandwidth		10		mV pk-pk
	50 kHz Bandwidth		2.5		mV pk-pk
ISOLATED POWER OUTPUT[8]					
Voltage	No Load	±14.25	±15	±17.25	V
vs. Temperature	0°C to +85°C		+20		mV/°C
	–40°C to 0°C		+25		mV/°C
Current at Rated Supply Voltage[2, 9]			±10		mA
Regulation	No Load to Full Load		–90		mV/V
Line Regulation			290		mV/V
Ripple	1 MHz Bandwidth, No Load[2]		50		mV rms
POWER SUPPLY					
Supply Voltage	Rated Performance	±14.5	±15	±16.5	V dc
	Operating[10]	±14.25		±17	V dc
Current	Operating (+15 V dc/–15 V dc Supplies)		+40/–18		mA
TEMPERATURE RANGE					
Rated Performance		–40		+85	°C
Storage		–40		+85	°C

NOTES

[1]The gain range of the AD215 is specified from 1 to 10 V/V. The AD215 can also be used with gains of up to 100 V/V. With a gain of 100 V/V a 20% reduction in the –3 dB bandwidth specification occurs and the nonlinearity degrades to ±0.02% typical.

[2]When the isolated supply load exceeds ±1 mA, external filter capacitors are required in order to ensure that the gain, offset, and nonlinearity specifications are preserved and to maintain the isolated supply full load ripple below the specified 50 mV rms. A value of 6.8 µF is recommended.

[3]Nonlinearity is specified as a percent (of full-scale range) deviation from a best straight line.

[4]The isolation barrier (and rating) of every AD215 is 100% tested in production using a 5 second partial discharge test with a failure detection threshold of 150 pC. All "B" grade devices are tested with a minimum voltage of 1,800 V rms. All "A" grade devices are tested with a minimum voltage of 850 V rms.

[5]The AD215 should be allowed to warm up for approximately 10 minutes before any gain and/or offset adjustments are made.

[6]Equivalent to a 0.8 degrees phase shift.

[7]With the ±15 V dc power supply pins bypassed by 2.2 µF capacitors at the AD215 pins.

[8]Caution: The AD215 design does not provide short circuit protection of its isolated power supply. A current limiting resistor may be placed in series with the isolated power terminals and the load in order to protect the supply against inadvertent shorts.

[9]With an input power supply voltage greater than or equal ±15 V dc, the AD215 may supply up to ±15 mA from the isolated power supplies.

[10]Voltages less than 14.25 V dc may cause the AD215 to cease operating properly. Voltages greater than ±17.5 V dc may damage the internal components of the AD215 and consequently should not be used.

Specifications subject to change without notice.

M Microcontroller datasheet

MICROCHIP

dsPIC30F1010/202X

28/44-Pin dsPIC30F1010/202X Enhanced Flash SMPS 16-Bit Digital Signal Controller

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *"dsPIC30F Family Reference Manual"* (DS70046). For more information on the device instruction set and programming, refer to the *"dsPIC30F/33F Programmer's Reference Manual"* (DS70157).

High-Performance Modified RISC CPU:

- Modified Harvard architecture
- C compiler optimized instruction set architecture
- 83 base instructions with flexible addressing modes
- 24-bit wide instructions, 16-bit wide data path
- 12 Kbytes on-chip Flash program space
- 512 bytes on-chip data RAM
- 16 x 16-bit working register array
- Up to 30 MIPS operation:
  - Dual Internal RC
    - 9.7 and 14.55 MHz (±1%) Industrial Temp
    - 6.4 and 9.7 MHz (±1%) Extended Temp
  - 32X PLL with 480 MHz VCO
  - PLL inputs ±3%
  - External EC clock 6.0 to 14.55 MHz
  - HS Crystal mode 6.0 to 14.55 MHz
- 32 interrupt sources
- Three external interrupt sources
- 8 user-selectable priority levels for each interrupt
- 4 processor exceptions and software traps

DSP Engine Features:

- Modulo and Bit-Reversed modes
- Two 40-bit wide accumulators with optional saturation logic
- 17-bit x 17-bit single-cycle hardware fractional/integer multiplier
- Single-cycle Multiply-Accumulate (MAC) operation
- 40-stage Barrel Shifter
- Dual data fetch

Peripheral Features:

- High-current sink/source I/O pins: 25 mA/25 mA
- Three 16-bit timers/counters; optionally pair up 16-bit timers into 32-bit timer modules
- One 16-bit Capture input functions
- Two 16-bit Compare/PWM output functions
  - Dual Compare mode available
- 3-wire SPI modules (supports 4 Frame modes)
- I^2C™ module supports Multi-Master/Slave mode and 7-bit/10-bit addressing
- UART Module:
  - Supports RS-232, RS-485 and LIN 1.2
  - Supports IrDA® with on-chip hardware endec
  - Auto wake-up on Start bit
  - Auto-Baud Detect
  - 4-level FIFO buffer

Power Supply PWM Module Features:

- Four PWM generators with 8 outputs
- Each PWM generator has independent time base and duty cycle
- Duty cycle resolution of 1.1 ns at 30 MIPS
- Individual dead time for each PWM generator:
  - Dead-time resolution 4.2 ns at 30 MIPS
  - Dead time for rising and falling edges
- Phase-shift resolution of 4.2 ns @ 30 MIPS
- Frequency resolution of 8.4 ns @ 30 MIPS
- PWM modes supported:
  - Complementary
  - Push-Pull
  - Multi-Phase
  - Variable Phase
  - Current Reset
  - Current-Limit
- Independent Current-Limit and Fault Inputs
- Output Override Control
- Special Event Trigger
- PWM generated ADC Trigger

Analog Features:

ADC

- 10-bit resolution
- 2000 Ksps conversion rate
- Up to 12 input channels
- "Conversion pairing" allows simultaneous conversion of two inputs (i.e., current and voltage) with a single trigger
- PWM control loop:
  - Up to six conversion pairs available
  - Each conversion pair has up to four PWM and seven other selectable trigger sources
- Interrupt hardware supports up to 1M interrupts per second

COMPARATOR

- Four Analog Comparators:
  - 20 ns response time
  - 10-bit DAC reference generator
  - Programmable output polarity
  - Selectable input source
  - ADC sample and convert capable
- PWM module interface
  - PWM Duty Cycle Control
  - PWM Period Control
  - PWM Fault Detect
- Special Event Trigger
- PWM-generated ADC Trigger

Special Microcontroller Features:

- Enhanced Flash program memory:
  - 10,000 erase/write cycle (min.) for industrial temperature range, 100k (typical)
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Flexible Watchdog Timer (WDT) with on-chip low power RC oscillator for reliable operation
- Fail-Safe clock monitor operation
- Detects clock failure and switches to on-chip low power RC oscillator
- Programmable code protection
- In-Circuit Serial Programming™ (ICSP™)
- Selectable Power Management modes
  - Sleep, Idle and Alternate Clock modes

CMOS Technology:

- Low-power, high-speed Flash technology
- 3.3V and 5.0V operation (±10%)
- Industrial and Extended temperature ranges
- Low power consumption

Pin Diagrams

28-Pin SDIP and SOIC

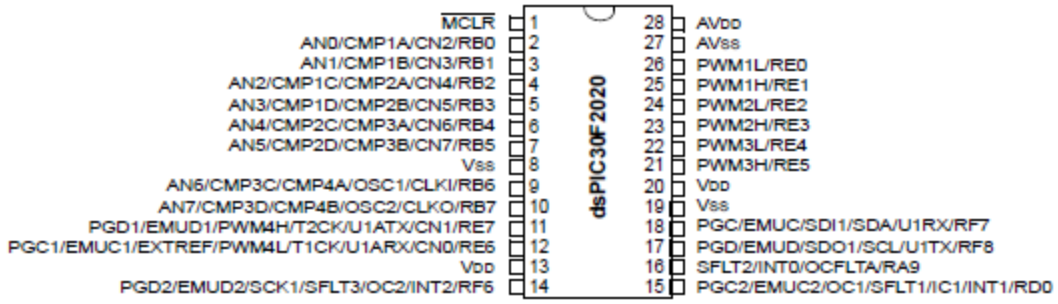


FIGURE 1-2: dsPIC30F2020 BLOCK DIAGRAM

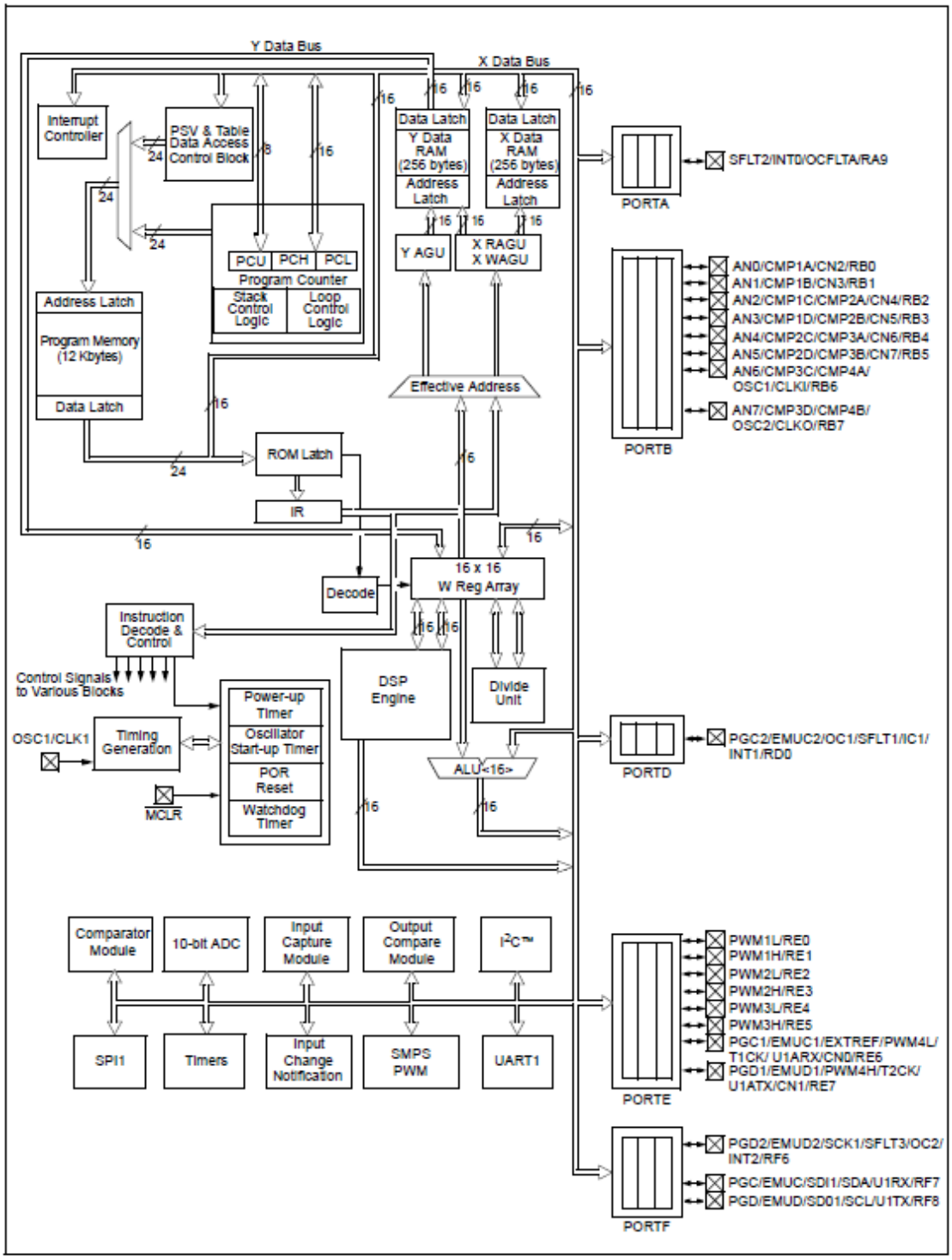


TABLE 1-2: PINOUT I/O DESCRIPTIONS FOR dsPIC30F2020

Pin Name	Pin Type	Buffer Type	Description
AN0-AN7	I	Analog	Analog input channels.
AVDD	P	P	Positive supply for analog module.
AVSS	P	P	Ground reference for analog module.
CLKI CLKO	I O	ST/CMOS —	External clock source input. Always associated with OSC1 pin function. Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in RC and EC modes. Always associated with OSC2 pin function.
EMUD EMUC EMUD1 EMUC1 EMUD2 EMUC2	I/O I/O I/O I/O I/O I/O	ST ST ST ST ST ST	ICD Primary Communication Channel data input/output pin. ICD Primary Communication Channel clock input/output pin. ICD Secondary Communication Channel data input/output pin. ICD Secondary Communication Channel clock input/output pin. ICD Tertiary Communication Channel data input/output pin. ICD Tertiary Communication Channel clock input/output pin.
IC1	I	ST	Capture input.
INT0 INT1 INT2	I I I	ST ST ST	External interrupt 0 External interrupt 1 External interrupt 2
SFLT1 SFLT2 SFLT3 PWM1L PWM1H PWM2L PWM2H PWM3L PWM3H PWM4L PWM4H	I I I O O O O O O O O	ST ST ST — — — — — — — —	Shared Fault Pin 1 Shared Fault Pin 2 Shared Fault Pin 3 PWM 1 Low output PWM 1 High output PWM 2 Low output PWM 2 High output PWM 3 Low output PWM 3 High output PWM 4 Low output PWM 4 High output
$\overline{\text{MCLR}}$	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low Reset to the device.
OC1-OC2 OCFLTA	O I	—	Compare outputs. Output Compare Fault pin
OSC1 OSC2	I I/O	CMOS —	Oscillator crystal input. Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in FRC and EC modes.
PGD PGC PGD1 PGC1 PGD2 PGC2	I/O I I/O I I/O I	ST ST ST ST ST ST	In-Circuit Serial Programming™ data input/output pin. In-Circuit Serial Programming clock input pin. In-Circuit Serial Programming data input/output pin 1. In-Circuit Serial Programming clock input pin 1. In-Circuit Serial Programming data input/output pin 2. In-Circuit Serial Programming clock input pin 2.

Legend: CMOS = CMOS compatible input or output; Analog = Analog input
ST = Schmitt Trigger input with CMOS levels; O = Output
I = Input; P = Power

TABLE 1-2: PINOUT I/O DESCRIPTIONS FOR dsPIC30F2020 (CONTINUED)

Pin Name	Pin Type	Buffer Type	Description
RB0-RB7	I/O	ST	PORTB is a bidirectional I/O port.
RA9	I/O	ST	PORTA is a bidirectional I/O port.
RD0	I/O	ST	PORTD is a bidirectional I/O port.
RE0-RE7	I/O	ST	PORTE is a bidirectional I/O port.
RF6, RF7, RF8	I/O	ST	PORTF is a bidirectional I/O port.
SCK1 SDI1 SDO1	I/O I O	ST ST —	Synchronous serial clock input/output for SPI #1. SPI #1 Data In. SPI #1 Data Out.
SCL SDA	I/O I/O	ST ST	Synchronous serial clock input/output for I^2C™. Synchronous serial data input/output for I^2C.
T1CK T2CK	I I	ST ST	Timer1 external clock input. Timer2 external clock input.
U1RX U1TX U1ARX U1ATX	I O I O	ST — ST O	UART1 Receive. UART1 Transmit. Alternate UART1 Receive. Alternate UART1 Transmit.
CMP1A CMP1B CMP1C CMP1D CMP2A CMP2B CMP2C CMP2D CMP3A CMP3B CMP3C CMP3D CMP4A CMP4B	I I I I I I I I I I I I I I	Analog Analog Analog Analog Analog Analog Analog Analog Analog Analog Analog Analog Analog Analog	Comparator 1 Channel A Comparator 1 Channel B Comparator 1 Channel C Comparator 1 Channel D Comparator 2 Channel A Comparator 2 Channel B Comparator 2 Channel C Comparator 2 Channel D Comparator 3 Channel A Comparator 3 Channel B Comparator 3 Channel C Comparator 3 Channel D Comparator 4 Channel A Comparator 4 Channel B
CN0-CN7	I	ST	Input Change notification inputs Can be software programmed for internal weak pull-ups on all inputs.
VDD	P	—	Positive supply for logic and I/O pins.
VSS	P	—	Ground reference for logic and I/O pins.
EXTREF	I	Analog	External reference to Comparator DAC

Legend: CMOS = CMOS compatible input or output; Analog = Analog input; ST = Schmitt Trigger input with CMOS levels; O = Output; I = Input; P = Power

N Microcontroller operating code

```
1 //*****************************************************************************
2 //* fiberpic.c
3 //*
4 //* Code to run dsPIC30F2020 with fiber optic serial modem to send ADC data
5 //* Requires C30 compiler
6 //*
7 //*****************************************************************************
8
9 #include <p30f2020.h>
```

```
10 //#include <stdio.h>
11 //#include <math.h>
12
13
14 //******************************************************************************
15 //* Configuration Bits
16 //******************************************************************************
17 //Fcy=(SOURCE OSCILLATOR FREQUENCY * PLL MULTIPLIER)/(PROGRAMMABLE POSTSCALER * 4)
18
19
20 _FOSCSEL(FRC_PLL)                                   //Select Fast RC oscillator with PLL
21 _FOSC(CSW_FSCM_OFF & FRC_HI_RANGE & OSC2_IO & HS_EC_DIS);   //if HI Fcy=30 MHz and Tcy=33.3 ns
22 //_FPOR(PWRT_128)
23
24 _FWDT(FWDTEN_OFF & WDTPRE_PR32 & WDTPOST_PS1);   //Turn OFF Hardware WDT enable and set config bits
25 _FGS(CODE_PROT_OFF & GWRP_OFF);                  // Code Protect OFF
26
27
28 //******************************************************************************
29 // Function prototypes
30 //******************************************************************************
31
32 #define testbit(data,bitno) ((data>>bitno)&0x0001)
33 #define setbit(data,bitno) data=(data&(0x0001<<bitno))
34
35 //Median filter definitions
36 #define PIX_SORT(a,b) { if ((a)>(b)) PIX_SWAP((a),(b)); }
37 #define PIX_SWAP(a,b) { int temp=(a);(a)=(b);(b)=temp; }
38
39 void initCORE(void);       //init core registers
40 void timersetup(void);     //setup timers
41 void adcsetup(void);       //setup ADC
42 void uartsetup(void);      //setup uart for serial output
43 void iosetup(void);        //setup io
44 void pwmsetup(void);       //setup PWM generator
45 void INTsetup(void);       //setup interupts
46 void TxData(void);         //print output data to terminal
47 void PWMstart(void);       //enable PWM pins and start pulse timer
48 void PWMstop(void);        //disable PWM pins and stop pulse timer
49 void PWMlockout(void);     //PWM lockout timer, prevents EMI during shutdown from triggering second pulse
50 void PWMcontrol(void);     //PWM control loop, updates frequency or duty cycle
51 void PSprintdata(void);    //Pring PWM pulse data to console
52 void clrRXdata(void);      //clear RX data buffer
53
54 void __attribute__((interrupt, auto_psv)) _T1Interrupt(void);
55 void __attribute__((interrupt, auto_psv)) _T2Interrupt(void);
56 void __attribute__((interrupt, auto_psv)) _T3Interrupt(void);
57 void __attribute__((interrupt, auto_psv)) _ADCInterrupt(void);
58 void __attribute__((interrupt, auto_psv)) _U1RXInterrupt(void);
59
60 void __attribute__((interrupt, auto_psv)) _INT0Interrupt(void); //interupt for error detection from i/o extender
61 void __attribute__((interrupt, auto_psv)) _INT1Interrupt(void); //interupt for push button
62 void __attribute__((interrupt, auto_psv)) _INT2Interrupt(void); //interupt for optical trigger in
63
64 void printSerial_int(unsigned int);            // print unsigned int
65 void printSerial_binary_int(unsigned int);     // print unsigned int as binary string
66 void printSerial_signed_int(signed int val);   // print signed int
67 void printSerial_str(char *);                  // print char string from ram
68 void printSerialROM_str(const char * str);     // print char string from rom
69 void printSerial_ASCII(char value);            // print single char value
70 void parse_string(char *);                     // parse string for data
71 int  str_to_int(char *);                       // convert ascii numeric string to an int
72 int  opt_med3(int * p);                        // execute median filter on given 3 element array
73
74 //******************************************************************************
75 // Variables / const (const puts value in program memory instead of ram)
76 //******************************************************************************
77 const char ResetString[]={"RESET\r\n\0"};      //
78 const char InvalidString[]={"\r\nInvalid Command, type \"help\" for commands\r\n\0"};   //
79 const char InvalidNumberString[]={"\r\nNumber out of bounds\0"};
80 const char ResetInitString[]={"\r\nInitiating dsPIC reset\r\n\0"};
81 const char ADCString[]={" ADCcnt=\0"};                                  //
82 //const char TimeString[]={" SYSt= \0"};                                //
83 //const char SpaceString[]={" \0"};                                     //
84 //const char RxUARTString[]={" RxUART=\0"};                             //
```

```
85 const char SbufString[]={" Sbuf=\0"};                                //
86 const char PulseString[]={"\r\nPulse trig by serial command\0"};
87
88 const char freqString[]={"\r\nPWM Frequency(00200=>20kHz)= \0"};
89 const char dtyString[]={"\r\nPWM Duty Cycle(00100=>100%)= \0"};
90 const char ptimeString[]={"\r\nPWM pulse time(00100=>10ms)= \0"};
91 const char vsetString[]={"\r\nVout set point(08000=>80kV)= \0"};
92 const char vlimitString[]={"\r\nVlimit set point(08000=>80kV)= \0"};
93 const char kpString[]={"\r\nKp= \0"};
94 const char kiString[]={"\r\nKi= \0"};
95 const char Ch0123String[]={"ADC[0123]= \0"};    // String lables for ADC ch 0,1,2,3
96
97 const char CLS[]={0x1b,0x5b,0x32,0x4a};        // clear terminal screen sequence for VT100 emulation < ESC [ 2 J >
98 const char NL[]={"\r\n\0"};                    // carage return and line feed
99 const char HOME[]={"\r\0"};                    // carage return without line feed
100
101 // raw value from ADC
102 int     ADC_Ch0=0;                  // literal ADC value
103 int     ADC_Ch1=0;                  // literal ADC value
104 int     ADC_Ch2=0;                  // literal ADC value
105 int     ADC_Ch3=0;                  // literal ADC value
106
107 char received;                      // UART Rx data element
108 char RxDataString[80];              // UART Rx data string
109 int  RxDataPoint = 0;               // UART Rx data string pointer
110
111 struct{
112     unsigned TxDataReady :1;          //transmit data
113     unsigned RxDataReady :1;          //process received data
114     unsigned PSDataReady :1;          //transmit PWM data
115     }ProgCON;                       // [15..0] program control flags
116
117 //counter variables
118 unsigned int MAINcount=0;           // main program loop counter
119 unsigned int ADC_INTcount  = 0;     // ADC conversion complete interupt counter
120 unsigned int TMR1_INTcount = 0;     // timer1 interupt counter
121 unsigned int TMR2_INTcount = 0;     // timer2 interupt counter
122 unsigned int TMR3_INTcount = 0;     // timer3 interupt counter
123 unsigned int U1RX_INTcount = 0;     // UART Rx interupt counter
124
125 //data variables for power supply operation
126 int IIN_DATA_MAX = 0;               // max current from bank
127 int IIN_DATA_MIN = 0;               // min current from bank
128 int VIN_DATA_MAX = 0;               // max voltage on bank at start of pulse
129 int VIN_DATA_MIN = 0;               // min voltage on bank at end of pulse
130 int IOUT_DATA_MAX = 0;              // max current to load
131 int VOUT_DATA_MAX = 0;              // max voltage to load
132 int TVSP_DATA = 0;                  // time to reach data setpoint
133 int TVMAX_DATA = 0;                 // time to reach max voltage
134
135 //control variables for power supply operation
136 struct{
137     //unsigned interlock    :1;     //must be set to 1 to enable triggering (not yet implimented)
138     unsigned dummypulse :1;     //set is dummy pulse is running(no PWM output)
139     unsigned ConFreq    :1;     //set if output regulation via freq control
140     unsigned PWMlockout :1;     //set if PWM lockout is enabled
141     unsigned PSerror    :1;     //error in power supply
142     unsigned aboveVlimit:1;     //power supply over voltage
143     unsigned extINTerror:1;     //external interupt error
144     unsigned lowVin:1;          //Vin reading too low (no charge or disconnected input)
145     unsigned lowVout:1;         //Vout not reading above set threshold within set time
146     }PSCONFIG;                  // [15..0] power supply operation control flags
147
148 //variables for PWM control
149 int PWMfreq = 220;              //PWMfreq=200 for 20kHz (used for user input for const freq operation)
150 int PWMper=47000;               //PWMper=47000 for 20khz (used in control loop for increaced speed)
151 int Dty = 100;                //PWM duty cycle 100 => 100%
152
153 //setpoints and limits
154 int Vset = 7500;                //output voltage setpoint 8000=80kV
155 int Vlimit=8500;                //output voltage upper limit (85kV)
156 int Vinmin=150;                 //input voltage minimum limit for power supply operation
157 int Ptime = 10;                 //pulse time setpoint in uS 100=10ms (duration of output pulse)
158 int LOCKtime=100;               //lockout time(post pulse trigger inhibit) setpoint in uS (100=10ms)
159 int Fmax=250;                   //max frequency limit(user input)
```

```
160 int Fres=185;                  //resonant frequency(user input)
161 int PWMpermin=37600;            //minimum PWM period(corosponds to maxumum frequency)
162 int PWMperres=50810;            //resonant PWM period(corosponds to resonant frequency)
163 //int Fmin=150;                 //min frequency limit(user input)
164 int Dtymax=100;                 //max duty cycle limit
165 int Dtymin=50;                  //mim duty cycle limit
166 int VoutminTH=2;                //Vout startup threshold(5=50%Vset, if not above threshold, at given time, shutdown)
167 int VsetTH=750;                 //Vout startup internal threshold (pre-calc value to eliminate devide operation in loop)
168 int VoutminTHtime=10;           //Vout startup threshold time(10=0.1ms)
169 int VoutminTHtimeTMR2=0;        //Vout startup threshold time converted to counts of TMR2
170
171 //PID variables
172 int Kpfreq=0;                   //preportinal term gain
173 int Kifreq=0;                   //integral term gain
174 int Kilimit=2000;               //limit on integral accumulator
175 signed int KiACC=0;             //integral term accumulator
176
177 int ADC_convINTcount=0;     //ADC interupts during the pulse
178 int ADC_trigINTcount=0;     //ADC trigger requests during pulse
179 //int ADCthcnt=0;               //ADC samples under a given threshold#########debuging
180 //int ADCth=200;                //ADC sample threshold#########debuging
181
182 //measured voltage and current
183 int Vout=0;                     //output voltage
184 int Iout=0;                     //output current
185 int Vin=0;                      //input voltage
186 int Iin=0;                      //input current
187 signed int Vouterror=0;         //voltage error
188
189 //pulse data storage
190 int VinBuff3[3];                //cyclic buffer for ADC input
191 int VinBuff3ptr;                //array element pointer
192 int MedianBuff[3];
193 //int timearr[40];
194 //int Voutarr[40];
195 //int Vinarr[40];
196
197 //--------------------------------------------------------------------------------
198 //   *********        MAIN function        *************
199 //--------------------------------------------------------------------------------
200 int main(void){
201
202 //for safety, the first thing on power up, make sure pwm output is stopped
203 //incase processor crashed and reset, leaving PWM generator in the on state
204     //Overide PWM outputs to shutdown pwm
205     IOCON1bits.OVRENH=1;              // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
206     IOCON1bits.OVRENL=1;              // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
207     IOCON2bits.OVRENH=1;              // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
208     IOCON2bits.OVRENL=1;              // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
209     IOCON3bits.OVRENH=1;              // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
210     IOCON3bits.OVRENL=1;              // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
211
212     PSCONFIG.PWMlockout=1;            //enable PWM lockout
213     PSCONFIG.ConFreq=1;               //set to 1 (Feedback on by default)
214
215 //initilize non-critical systems
216     initCORE();                     // init system
217     uartsetup();                    // setup UART for communication
218     adcsetup();                     // setup ADC
219     timersetup();                   // setup timers
220     iosetup();                      // setup i/o pins and ports
221     pwmsetup();                     // setup PWM system
222
223 //now that timers are setup lockout PWM for designated time on startup
224 //since the first PWMstop() is called before timer setup, it may cleared
225     PWMlockout();                   //start timer2 for PWM lockout MUST BE CALLED AFTER PWMstop()
226
227 //initilize interupts(after PWM lockout is started)
228     INTsetup();                     // setup INT 0,1,2 interupts
229
230 //init received character value and print out RESET to console
231     received='\0';
232     printSerialROM_str(ResetString);    //print out string that says "RESET"
233
234 //print out cause of reset and intcon1 configuration
```

```
235 //  printSerialROM_str(NL);
236     printSerialROM_str("RCON=    \0");  //print out RCOD data to display cause of reset
237     printSerial_binary_int(RCON);
238     printSerialROM_str(NL);
239     printSerialROM_str("INTCON1= \0");  //print intcon1 to display configuration of interupts
240     printSerial_binary_int(INTCON1);
241     printSerialROM_str(NL);
242     printSerialROM_str(NL);
243
244 //alert if watchdog timer timeout was cause of reset
245 if(RCONbits.WDTO==1){
246     printSerialROM_str("WDT RESET: Woof Woof \"The Watchdog\" did not get feed \r\n\0");
247     RCONbits.WDTO=0;
248 }
249
250
251 // Main Program Loop (entire program is interupt driven)
252 while(1){
253
254 MAINcount++;                        // increment loop counter
255
256 //Power supply error
257     if(PSCONFIG.PSerror==1){
258         printSerialROM_str("\r\n----------Power supply error:----------\0");
259         PSCONFIG.PSerror=0;
260
261     //output over voltage limit error
262         if(PSCONFIG.aboveVlimit==1){
263             printSerialROM_str("\r\nOutput overvoltage\0");
264             PSCONFIG.aboveVlimit=0;
265         }
266     //external interupt error
267         if(PSCONFIG.extINTerror==1){
268             printSerialROM_str("\r\nExternal error interupt\0");
269             PSCONFIG.extINTerror=0;
270         }
271
272     //Vin capacitor bank feedback sensor not conected or voltage too low
273         if(PSCONFIG.lowVin==1){
274             printSerialROM_str("\r\nVin too low or sensor disconected\0");
275             PSCONFIG.lowVin=0;              //DC link undervolt flag
276         }
277
278     //Vout feedback sensor not conected or voltage too low during startup
279         if(PSCONFIG.lowVout==1){
280             printSerialROM_str("\r\nVout too low or sensor disconected\0");
281             PSCONFIG.lowVout=0;             //Output undervolt flag
282         }
283
284     //Power supply error list newline to prevent last entery from being covered by ADC data output
285         printSerialROM_str(NL);
286         printSerialROM_str(NL);
287     }
288
289
290 //transmit PS data
291     if(ProgCON.PSDataReady==1){
292         PSprintdata();
293         ProgCON.PSDataReady=0;
294     }
295
296 //transmit data string
297     if (ProgCON.TxDataReady==1){
298         TxData();                       // transmit data
299         ProgCON.TxDataReady=0;
300     }
301
302 //parse recived data if ProgCON.RxDataReady is set by interupts
303     if (ProgCON.RxDataReady==1){
304
305         printSerialROM_str(NL);
306         printSerialROM_str(SbufString);
307         printSerial_str(RxDataString);          // print received string to terminal
308
309         parse_string(RxDataString);             // Parse received string for commands
```

```
310         clrRXdata();                        //clear RX data buffer
311         ProgCON.RxDataReady=0;              //clear program control flag to parse data
312       }//end if
313
314
315 }//end while
316 }//end main
317
318
319 //---------------------------------------------------------------------------
320 // Transmit data set as ASCII srting
321 //---------------------------------------------------------------------------
322 void TxData(void){
323
324         printSerialROM_str(HOME);
325
326       if (received=='#'){
327         printSerialROM_str(CLS);
328         received=0;
329       }//end if
330       else {
331
332         printSerialROM_str(Ch0123String);
333         printSerial_signed_int(ADC_Ch0);
334         printSerial_signed_int(ADC_Ch1);
335         printSerial_signed_int(ADC_Ch2);
336         printSerial_signed_int(ADC_Ch3);
337         printSerialROM_str(ADCString);
338         printSerial_int(ADC_INTcount);
339
340 //      printSerialROM_str(" ADCth=");
341 //      printSerial_int(ADCthcnt);
342
343 //      printSerialROM_str(ConfigString);
344
345         printSerialROM_str(SbufString);
346         printSerial_str(RxDataString);
347
348 //      printSerialROM_str(NL);
349
350       }    //end else
351
352 }
353
354
355 //---------------------------------------------------------------------------
356 // U1RX interrupt, triggered by incoming serial data
357 //---------------------------------------------------------------------------
358 void __attribute__((__interrupt__)) _U1RXInterrupt(void){
359     IFS0bits.U1RXIF = 0;            // clear interrupt flag
360     U1RX_INTcount++;                // increment interupt counter
361
362     while(U1STAbits.URXDA){         //while there is data in buffer
363
364     received = U1RXREG;                                 //received character into temp variable
365                                                 //(reading the character, pops it from buffer stack)
366
367     if(received == '$'){
368       clrRXdata();                          //clear RX data buffer
369
370       RxDataString[RxDataPoint]=received;       // store received char into Rx data string
371       RxDataPoint++;                        // incriment pointer
372
373     }//end if
374     else if(received == '\r'){              //if carage return
375       RxDataPoint++;                        //incriment pointer
376       RxDataString[RxDataPoint]='\0';           //write zero after end of data
377       RxDataString[80]='\0';                  //write zero to end of string buffer (backup overflow prevention)
378       RxDataPoint = 0;                      //reset pointer beck to beginning of string
379
380       ProgCON.RxDataReady=1;                    //set program control flag to parse data
381
382     }//end else if
383     else if(received == 0x08){                  //if backspace
384       if(RxDataPoint>0){RxDataPoint--;}         //decriment pointer if greater then zero, else just clear the first char
```

```
385         RxDataString[RxDataPoint]='\0';                     //write zero after end of data
386     }//end else if
387
388     else{
389         RxDataString[RxDataPoint]=received;                 // store received char into Rx data string
390         RxDataPoint++;                                      // incriment pointer
391     }//end else
392
393     }//end while
394
395 }
396
397
398 //---------------------------------------------------------------------------
399 // Clear RX data buffer
400 //---------------------------------------------------------------------------
401 void clrRXdata(void){
402     RxDataString[0]='\0';                                   //clear buffer (write zeros)
403     for(RxDataPoint=0; RxDataPoint!=80; RxDataPoint++){
404         RxDataString[RxDataPoint]='\0';                     //clear buffer (write zeros)
405     }//end for
406     RxDataString[80]='\0';
407     RxDataPoint = 0;                            // UART Rx data string pointer
408 }
409
410
411 //---------------------------------------------------------------------------
412 // INT0 interrupt, error detection from i/o extender
413 //---------------------------------------------------------------------------
414 void __attribute__((__interrupt__)) _INT0Interrupt(void)
415 {
416     IFS0bits.INT0IF = 0;                // clear(0) interrupt flag
417     PWMstop();                          // stop PWM output
418     PSCONFIG.PSerror=1;                 // set error flag
419     PSCONFIG.extINTerror=1;             // set external interupt error flag
420
421 }
422
423
424 //---------------------------------------------------------------------------
425 // INT1 interrupt, front panel push button
426 //---------------------------------------------------------------------------
427 void __attribute__((__interrupt__)) _INT1Interrupt(void)
428 {
429     IFS1bits.INT1IF = 0;                // clear(0) interrupt flag
430
431 //start PWM pulse
432     if((IOCON1bits.OVRENH==1)&(PSCONFIG.PWMlockout==0)){        // If the pulse isnt running and lockout disabled
433         if((Vin<Vinmin)&(PSCONFIG.dummypulse==0)){
434             PSCONFIG.PSerror=1;             //set power supply error flag
435             PSCONFIG.lowVin=1;              //set DC link undervolt flag
436             PWMstop();                      //shut down power supply
437         }
438         else{
439             PWMstart();             // start pulse timer and enable outputs
440         }
441     }// end of if statement
442
443 }
444
445
446
447 //---------------------------------------------------------------------------
448 // INT2 interrupt, optical trigger in
449 //---------------------------------------------------------------------------
450 void __attribute__((__interrupt__)) _INT2Interrupt(void)
451 {
452     IFS1bits.INT2IF = 0;                // clear(0) interrupt flag
453
454 //start PWM pulse
455     if((IOCON1bits.OVRENH==1)&(PSCONFIG.PWMlockout==0)){        // If the pulse isnt running and lockout disabled
456         if((Vin<Vinmin)&(PSCONFIG.dummypulse==0)){
457             PSCONFIG.PSerror=1;             //set power supply error flag
458             PSCONFIG.lowVin=1;              //set DC link undervolt flag
459             PWMstop();                      //shut down power supply
```

```
460         }
461         else{
462             PWMstart();             // start pulse timer and enable outputs
463         }
464     }// end of if statement
465
466 }
467
468
469 //-------------------------------------------------------------------------
470 // TMR1 interrupt, used to trigger ADC
471 //-------------------------------------------------------------------------
472 void __attribute__((__interrupt__)) _T1Interrupt(void)
473 {
474     IFS0bits.T1IF = 0;              // clear interrupt flag, set when timer=PR1
475     TMR1=0;                         // clear timer register
476
477 //###############for testing
478     if((IOCON1bits.OVRENH==0)){     //if the pulse is active
479         ADC_trigINTcount++;         //ADC interupts during the pulse
480     }
481
482 //ADC now triggered using speciel event trigger from TMR1 period match
483 //  ADCPC0bits.SWTRG0=1;            //Start conversion of AN0 and AN1 if trigger source is SWTRG
484 //  ADCPC0bits.SWTRG1=1;            //Start conversion of AN2 and AN3 if trigger source is SWTRG
485 }
486
487
488
489 //-------------------------------------------------------------------------
490 // TMR2 interrupt, used to time power supply output pulse
491 //-------------------------------------------------------------------------
492 void __attribute__((__interrupt__)) _T2Interrupt(void)
493 {
494     IFS0bits.T2IF = 0;              // clear interrupt flag, set when timer=PR2
495     T2CONbits.TON=0;                // 1=> start timer, 0=> stop timer
496     TMR2=0;                         // clear timer register
497     IEC0bits.T2IE = 0;              // disable timer interupt(timer not runing after pulse ends)
498
499 //trigger PWM lockout
500     if((IOCON1bits.OVRENH==0)){     //if PWM is running
501         PSCONFIG.PWMlockout=1;      //enable PWM lockout
502         PWMlockout();               //start timer2 for PWM lockout
503         ProgCON.PSDataReady=1;      //set flag to print PWM data to console
504     }
505     else{                           //if PWM lockout enabled and pulse not running
506         PSCONFIG.PWMlockout=0;      //clear lockout
507     }
508 //stop PWM pulse(called twice,at pulse end and lockout end for safety)
509     PWMstop();                      //stop PWM pulse
510 }
511
512
513 //-------------------------------------------------------------------------
514 // TMR3 interrupt, used to send data I/O
515 //-------------------------------------------------------------------------
516 void __attribute__((__interrupt__)) _T3Interrupt(void)
517 {
518     IFS0bits.T3IF = 0;              // clear interrupt flag
519     TMR3=0;                         // clear timer register (reset timer)
520     ProgCON.TxDataReady = 1;        //set program control flag to transmit data
521 }
522
523 //-------------------------------------------------------------------------
524 // PWMstart enable PWM outputs and start TMR2 to time pulse lenght
525 //-------------------------------------------------------------------------
526 void PWMstart(void){
527
528 //set control variables to default
529 //duty cycle = 100% (varies between 50-100%)
530 //PWMfreq controls PWM frequency 200 => 20kHz
531 PWMfreq=Fres;
532
533 //clear old pulse data
534     IIN_DATA_MAX = 0;               // max current from bank
```

```
535    IIN_DATA_MIN = 9999;            // min current from bank(set to large number, so less then operation will overwrite)
536    VIN_DATA_MAX = 0;               // max voltage on input
537    VIN_DATA_MIN = 9999;            // min voltage on input(set to large number, so less then operation will overwrite)
538    IOUT_DATA_MAX = 0;              // max current to load
539    VOUT_DATA_MAX = 0;              // max voltage to load
540    TVSP_DATA = 0;                 // time to reach data setpoint
541    TVMAX_DATA = 0;                // time to reach max voltage
542
543 //clear old variables
544    KiACC=0;                      //integral term accumulator
545    ADC_convINTcount=0;           //ADC interupts during the pulse
546    ADC_trigINTcount=0;           //ADC trigger requests during pulse
547    VsetTH=Vset/10;
548    VinBuff3[0]=Vin;              //cyclic buffer for ADC input
549    VinBuff3[1]=Vin;              //cyclic buffer for ADC input
550    VinBuff3[2]=Vin;              //cyclic buffer for ADC input
551    VinBuff3ptr=0;                //array element pointer
552
553
554 //set timer 1 to trigger ADC at higher sampling rate during pulse
555    T1CONbits.TCKPS=0b00;         // prescale: 0b00=>clk/1, 0b01=>clk/8, 0b10=>clk/64, 0b11=>clk/256
556    TMR1=0;                      // clear timer register
557    PR1=200;                     // Interrupt after this many cycles
558
559 //Toggle PWM generator enable to reset timebase to 0
560    PTCONbits.PTEN=0;             // Enable(1) or Disable(0) PWM timebase
561    PTCONbits.PTEN=1;             // Enable(1) or Disable(0) PWM timebase
562
563 //set PTCONbits.EIPU=1 during changes of preiod and duty cycle so they take place before the cycle starts
564    PTCONbits.EIPU=1;                    // Active period updates immediatly(1) or on cycle bounderies(0)
565
566    if(PSCONFIG.ConFreq==1){             //if in frequency control mode
567       PWMper=(Vset/(Vin+1))*1724+29706;   //faster calculation of PWM period(fewer devides required)
568    }
569    else{                                //manual frequency setpoint for testing
570       PWMper=235*(40000/PWMfreq);       // PTPER<15:3>: PWM Time Base Period Value bits (20kHz = 47000 ~(235*200) for complimentary,)
571    }
572
573
574    PTPER=PWMper;
575    MDC=PTPER/2;                  // MDC<15:0> PWM MASTER DUTY CYCLE REGISTER (0xFFFF to 0x0008)
576    DTR1=50+(PTPER/2)*(100-Dty);    // Dead time register
577    DTR2=DTR1;                    // Dead time register
578    DTR3=DTR1;                    // Dead time register
579    ALTDTR1=DTR1;                 // Alternate Dead time register
580    ALTDTR2=DTR1;                 // Dead time register
581    ALTDTR3=DTR1;                 // Dead time register
582    PHASE1=0;                    // PWM timebase phase shift time value
583    PHASE2=PTPER*1/3;             // PWM timebase phase shift time value
584    PHASE3=PHASE2*2;              // PWM timebase phase shift time value
585
586    PTCONbits.EIPU=0;             // Active period updates immediatly(1) or on cycle bounderies(0)
587
588 //-------------start watchdog timer before pulse-----------------
589 RCONbits.SWDTEN=1;     //turn on(1) or off(0) software watchdog timer enable
590
591 //---------------code to start output pulse------------------
592       // now config timer 2 to control power supply output pulse
593       //Ptime => pulse time setpoint 100=10ms
594       PR2=(460*Ptime)/10;        // (4692=>10ms) Interrupt after this many cycles timer runs ar Fosc/4
595       //VoutminTHtime=> Vout startup threshold time(10=0.1ms)
596       VoutminTHtimeTMR2=(46*VoutminTHtime)/10;    //set startup threshold timer
597
598       // start countdown
599       TMR2=0;                   // clear timer register
600       IEC0bits.T2IE = 1;         // 1=> enable interrupts, 0=> disable interupts
601       T2CONbits.TON=1;           // 1=> start timer, 0=> stop timer
602
603    //enable pwm pins
604       IOCON1bits.PENH=1;            // PWMxH pin is enabled(1, PWM) or disabled(0, general i/o)
605       IOCON2bits.PENH=1;            // PWMxH pin is enabled(1, PWM) or disabled(0, general i/o)
606       IOCON3bits.PENH=1;            // PWMxH pin is enabled(1, PWM) or disabled(0, general i/o)
607
608       IOCON1bits.PENL=1;            // PWMxL pin is enabled(1, PWM) or disabled(0, general i/o)
609       IOCON2bits.PENL=1;            // PWMxL pin is enabled(1, PWM) or disabled(0, general i/o)
```

```
610       IOCON3bits.PENL=1;              // PWMxL pin is enabled(1, PWM) or disabled(0, general i/o)
611
612    //Enable PWM outputs
613       IOCON1bits.OVRENH=0;            // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
614       IOCON1bits.OVRENL=0;            // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
615       IOCON2bits.OVRENH=0;            // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
616       IOCON2bits.OVRENL=0;            // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
617       IOCON3bits.OVRENH=0;            // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
618       IOCON3bits.OVRENL=0;            // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
619 }
620
621 //-------------------------------------------------------------------------
622 // PWMstop disable PWM outputs and stop pulse
623 //-------------------------------------------------------------------------
624 void PWMstop(void){
625
626 //set timer 1 (ADC trigger) to lower sampling rate after pulse
627    T1CONbits.TCKPS=0b01;          // prescale: 0b00=>clk/1, 0b01=>clk/8, 0b10=>clk/64, 0b11=>clk/256
628    TMR1=0;                        // clear timer register
629    PR1=10000;                      // Interrupt after this many cycles 16mhz
630
631 //----------------code to stop output pulse-------------------
632
633    //Overide PWM outputs to shutdown pwm(soft switching shutdown, activates at cycle boundary)
634    IOCON1bits.OVRENH=1;            // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
635    IOCON1bits.OVRENL=1;            // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
636    IOCON2bits.OVRENH=1;            // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
637    IOCON2bits.OVRENL=1;            // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
638    IOCON3bits.OVRENH=1;            // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
639    IOCON3bits.OVRENL=1;            // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
640
641 //--------------stop watchdog timer after pulse----------------
642    RCONbits.SWDTEN=0;              //turn on(1) or off(0) software watchdog timer enable
643
644 //----------------clear dummy pulse flag if it was a dummy pulse-----------
645    PSCONFIG.dummypulse=0;          //dummy pulse flag
646 }
647
648 //-------------------------------------------------------------------------
649 // PWMlockout starts timer2 for a time during which the pulse can't be re-triggered
650 //-------------------------------------------------------------------------
651 void PWMlockout(void){
652    PR2=(469*LOCKtime)/10;      // (1173=>10ms) Interrupt after this many cycles timer runs ar Fosc/4
653    TMR2=0;                     // clear timer register
654    IEC0bits.T2IE = 1;          // 1=> enable interrupts, 0=> disable interupts
655    T2CONbits.TON=1;            // 1=> start timer, 0=> stop timer
656 }
657
658 //-------------------------------------------------------------------------
659 // ADC interrupt, triggered when ADC conversion completes
660 //-------------------------------------------------------------------------
661 void __attribute__((__interrupt__)) _ADCInterrupt(void)
662 {
663    IFS0bits.ADIF=0;            // reset interupt flag
664    ADC_INTcount++;             // inc interupt counter
665
666 if(ADSTATbits.P0RDY==1){        //if ADC pair 0 converted
667    ADC_Ch0 = ADCBUF0;          // pin 2 (Vout)
668    ADC_Ch1 = ADCBUF1;          // pin 3 (Iout)
669    ADSTATbits.P0RDY=0;         // ADSTATbits.P#RDY Conversion Data for Pair # Ready(1)
670 }
671 if(ADSTATbits.P1RDY==1){        //if ADC pair 1 converted
672    ADC_Ch2 = ADCBUF2;          // pin 4 (Vin)
673    ADC_Ch3 = ADCBUF3;          // pin 5 (Iin)
674    ADSTATbits.P1RDY=0;         // ADSTATbits.P#RDY Conversion Data for Pair # Ready(1)
675 }
676
677 //----------------calculate voltages and currents from ADC inputs
678    Vout=(ADC_Ch0*10);          //assuming 10000=100kV (20000:1 probe)
679    Iout=(ADC_Ch1);             //assuming ?x probe (?:1)
680    Vin=(ADC_Ch2);              //assuming 200x probe(1kV in = 5V => ADC_Ch2=1013) 1000=1kV
681    Iin=(ADC_Ch3);              //assuming ?x probe (?:1)
682
683
684 //if(Vin<ADCth){ADCthcnt++;}       //#######threshold counter for testing
```

```
685
686 //---------------run control loop-----------------------------
687 if((IOCON1bits.OVRENH==0)){//if the pulse (or dummypulse) is active, so start control loop
688     ADC_convINTcount++;         //ADC interupts during the pulse
689     PWMcontrol();               // Use new ADC data to update Dty or Freq
690 }//end if
691
692 //---------------clear watchdog timer-------------------------
693 //for safety WDT is enabled during pulse, WDT will reset dsPIC in 2mS if not cleared in control loop
694     asm("CLRWDT");              //clear watchdog timer if control loop is running
695 }
696
697
698 //-------------------------------------------------------------------------
699 // Sort 3 element array and return median value
700 //-------------------------------------------------------------------------
701 int opt_med3(int * p){
702     PIX_SORT(p[0],p[1]);
703     PIX_SORT(p[1],p[2]);
704     PIX_SORT(p[0],p[1]);
705     return(p[1]) ;
706 }
707
708
709 //-------------------------------------------------------------------------
710 // Update PWM control variables
711 //-------------------------------------------------------------------------
712 void PWMcontrol(void){
713
714 //----------------filter data-----------------------------
715
716 //write to cyclic buffer
717 //  VinBuff3ptr++;                  //array element pointer
718 //if(VinBuff3ptr>2){                  //reset pointer
719 //  VinBuff3ptr=0;}
720 //  VinBuff3[VinBuff3ptr]=Vin;      //cyclic buffer for ADC input
721
722 //copy array
723 //  MedianBuff[0]=VinBuff3[0];      //Median filter buffer
724 //  MedianBuff[1]=VinBuff3[1];      //Median filter buffer
725 //  MedianBuff[2]=VinBuff3[2];      //Median filter buffer
726
727 //find median value
728 //Vin=opt_med3(MedianBuff);         //Overwrite Vin with filtered value
729 //Vin=(VinBuff3[0]+VinBuff3[1]+VinBuff3[2]+VinBuff3[VinBuff3ptr])>>2;
730
731 //----------------record data-----------------------------
732
733 if(Iin>IIN_DATA_MAX){
734     IIN_DATA_MAX = Iin;}            // max current from bank
735
736 if(Iin<IIN_DATA_MIN){
737     IIN_DATA_MIN = Iin;}            // min current from bank
738
739 if(Vin>VIN_DATA_MAX){
740     VIN_DATA_MAX = Vin;}            // max voltage on input
741
742 if((Vin<VIN_DATA_MIN)){
743     VIN_DATA_MIN = Vin;}            // min voltage on input
744
745 if(Iout>IOUT_DATA_MAX){
746     IOUT_DATA_MAX = Iout;}      // max current to load
747
748 if(Vout>VOUT_DATA_MAX){
749     VOUT_DATA_MAX = Vout;
750     TVMAX_DATA= TMR2;}      // max voltage to load
751 //note: nested if statments run faster then if with the & of two conditions in argument
752 if(TVSP_DATA==0){   // time to reach data setpoint
753     if(Vout>Vset){
754         TVSP_DATA = TMR2;
755     }
756 }
757
758 //-------------------------update PWM switching frequency-------------
759     if(PSCONFIG.ConFreq==1){                    //if in frequency control mode
```

```
760         //PWMfreq controls PWM frequency 200 => 20kHz
761 //      Vouterror=(Vout-Vset);                    //voltage error
762 //      KiACC=KiACC+Vouterror/64;                 //integrate error
763 //      if(KiACC>Kilimit){KiACC=Kilimit;}         //limit accumulator positive bound
764 //      if(KiACC<-Kilimit){KiACC=-Kilimit;}       //limit accumulator negative bound
765
766         //compute required PWM frequency
767         //(Vin+1) to eliminate devide by zero error if Vin=0
768 //      PWMfreq=( 275-((Vset/(Vin+1))*100)/14+(Vouterror*Kpfreq)/1024 +(KiACC*Kifreq)/128 );
769         PWMper=(Vset/(Vin+1))*1724+29706;
770     }//end if
771
772 //------------------perform safety checks-------------------------
773 //if(PWMfreq>Fmax){PWMfreq=Fmax;}   //limit commanded frequency in safe bounds
774 //if(PWMfreq<Fres){PWMfreq=Fres;}   //limit commanded frequency in safe bounds
775 if(PWMper<PWMpermin){PWMper=PWMpermin;} //limit commanded frequency in safe bounds
776 if(PWMper>PWMperres){PWMper=PWMperres;} //limit commanded frequency in safe bounds
777
778 if(Vout>Vlimit){                  //if over maximum safe voltage limit
779     PSCONFIG.PSerror=1;           //set power supply error flag
780     PSCONFIG.aboveVlimit=1;       //set over voltage flag
781     PWMstop();                    //shut down power supply
782 }
783 //note: nested if statments run faster then if with the & of two conditions in argument
784 if(Vin<Vinmin){
785 if(PSCONFIG.dummypulse==0){
786     PSCONFIG.PSerror=1;           //set power supply error flag
787     PSCONFIG.lowVin=1;            //set DC link undervolt flag
788     PWMstop();                    //shut down power supply
789 }}
790
791 //if((Vout<((Vset*VoutminTH)/10))&(TMR2>VoutminTHtimeTMR2)&(PSCONFIG.dummypulse==0)){
792 //note: nested if statments run faster then if with the & of two conditions in argument
793 if(Vout<(VsetTH*VoutminTH)){
794 if(TMR2>VoutminTHtimeTMR2){
795 if(PSCONFIG.dummypulse==0){
796     PSCONFIG.PSerror=1;           //set power supply error flag
797     PSCONFIG.lowVout=1;           //Output undervolt flag
798     PWMstop();                    //shut down power supply
799 }}}
800
801
802 //-------------------------update control variables-----------------
803 // PTPER<15:3>: PWM Time Base Period Value bits (20kHz = 47000 ~(235*200) for complimentary)
804
805 //  PTPER=235*(40000/PWMfreq);      // PTPER<15:3>: PWM Time Base Period
806     PTPER=PWMper;                   // PTPER<15:3>: PWM Time Base Period
807     MDC=PTPER/2;                   // MDC<15:0> PWM MASTER DUTY CYCLE REGISTER (0xFFFF to 0x0008)
808     PHASE1=0;                     // PWM timebase phase shift time value
809     PHASE2=PTPER*1/3;               // PWM timebase phase shift time value
810     PHASE3=PHASE2*2;                // PWM timebase phase shift time value
811
812 }
813
814
815 //--------------------------------------------------------------------------------
816 // Print PWM data to console
817 //--------------------------------------------------------------------------------
818 void PSprintdata(void){
819
820     printSerialROM_str(NL);
821     printSerialROM_str("----------Pulse Data:----------\r\n");
822     if(PSCONFIG.ConFreq==0){            //if in manual mode for frequency responce testing
823         printSerialROM_str(freqString);
824         printSerial_int(PWMfreq);
825     }
826     printSerialROM_str(dtyString);
827     printSerial_int(Dty);
828     printSerialROM_str(ptimeString);
829     printSerial_int(Ptime);
830     printSerialROM_str(vsetString);
831     printSerial_int(Vset);
832     printSerialROM_str(vlimitString);
833     printSerial_int(Vlimit);
834     printSerialROM_str("\r\nKp= ");
```

```
835     printSerial_int(Kpfreq);
836     printSerialROM_str(" Ki= ");
837     printSerial_int(Kifreq);
838
839     printSerialROM_str("\r\nADCtrig samp= ");
840     printSerial_int(ADC_trigINTcount);
841     printSerialROM_str("\r\nADCconv samp= ");
842     printSerial_int(ADC_convINTcount);
843
844     printSerialROM_str(NL);
845
846     printSerialROM_str("Data from last pulse:\0");
847     printSerialROM_str("\r\nVin MAX=   \0");
848     printSerial_signed_int(VIN_DATA_MAX);
849     printSerialROM_str(" Vin MIN=   \0");
850     printSerial_signed_int(VIN_DATA_MIN);
851     printSerialROM_str("\r\nIin MAX=   \0");
852     printSerial_signed_int(IIN_DATA_MAX);
853     printSerialROM_str(" Iin MIN=   \0");
854     printSerial_signed_int(IIN_DATA_MIN);
855     printSerialROM_str("\r\nVout MAX=  \0");
856     printSerial_signed_int(VOUT_DATA_MAX);
857     printSerialROM_str(" Iout MAX=  \0");
858     printSerial_signed_int(IOUT_DATA_MAX);
859
860     printSerialROM_str("\r\nT to Vset= \0");
861     printSerial_signed_int(TVSP_DATA);
862     printSerialROM_str(" T to Vmax= \0");
863     printSerial_signed_int(TVMAX_DATA);
864
865     printSerialROM_str(NL);
866
867 }
868
869
870 //-------------------------------------------------------------------------------
871 // Initilize system variables, stacks
872 //-------------------------------------------------------------------------------
873 void initCORE(void){
874     OSCCON=0x3300;                  // Select Primary Oscillator
875 //  CORCON=
876 }
877
878 //-------------------------------------------------------------------------------
879 // Initilize i/o ports
880 //-------------------------------------------------------------------------------
881 void iosetup(void){
882
883     TRISDbits.TRISD0=1;             // RD0 (pin 15) as input
884     TRISAbits.TRISA9=0;             // RA9 (pin 16) as output
885     TRISFbits.TRISF6=1;             // RF6 (pin 14) as input
886
887 }
888
889
890 //-------------------------------------------------------------------------------
891 // Initilize INT 0,1,2 interupts
892 //-------------------------------------------------------------------------------
893 void INTsetup(void){
894
895 //INT0 => int from i/o extender
896     INTCON2bits.INT0EP=1;           //INT0 Interupt on rising edge(0) of falling edge(1)
897     IPC0bits.INT0IP=6;              //INT0 Set priority level 7=>highest, 1=>lowest, 0=>disabled
898     IEC0bits.INT0IE = 1;            //INT0 interupt enable(1) or disable(0)
899
900 //INT1 => int from front panel button
901     INTCON2bits.INT1EP=1;           //INT1 Interupt on rising edge(0) of falling edge(1)
902     IPC4bits.INT1IP=4;              //INT1 Set priority level 7=>highest, 1=>lowest, 0=>disabled
903     IEC1bits.INT1IE = 1;            //INT1 interupt enable(1) or disable(0)
904
905 //INT2 => int from optical trigger
906     INTCON2bits.INT2EP=1;           //INT2 Interupt on rising edge(0) of falling edge(1)
907     IPC4bits.INT2IP=5;              //INT2 Set priority level 7=>highest, 1=>lowest, 0=>disabled
908     IEC1bits.INT2IE = 1;            //INT2 interupt enable(1) or disable(0)
909
```

```
910 }
911
912
913
914 //------------------------------------------------------------------------------
915 // Initilize PWM generator
916 //------------------------------------------------------------------------------
917
918 void pwmsetup(void){
919
920 //disable PWM controller if enabled at startup SET TO 0 BEFORE CONFIGURATION
921     PTCONbits.PTEN=0;               // Enable(1) or Disable(0) PWM timebase
922
923     //pwm timebase setup
924     PTCONbits.EIPU=0;              // Active period updates immediatly(1) or on cycle bounderies(0)
925     PTCONbits.PTSIDL=1;             // Halt(1) or Run(0) PWM timebase in CPU idle
926
927     PTPER=235*(40000/PWMfreq);      // PTPER<15:3>: PWM Time Base Period Value bits (20kHz = 47000 ~(235*200) for complimentary,)
928     MDC=PTPER/2;                   // MDC<15:0> PWM MASTER DUTY CYCLE REGISTER (0xFFFF to 0x0008)
929     DTR1=50+(PTPER/2)*(100-Dty);    // Dead time register
930     DTR2=DTR1;                     // Dead time register
931     DTR3=DTR1;                     // Dead time register
932     ALTDTR1=DTR1;                  // Alternate Dead time register
933     ALTDTR2=DTR1;                  // Dead time register
934     ALTDTR3=DTR1;                  // Dead time register
935     PHASE1=0;                     // PWM timebase phase shift time value
936     PHASE2=PTPER*1/3;              // PWM timebase phase shift time value
937     PHASE3=PHASE2*2;               // PWM timebase phase shift time value
938
939     PWMCON1bits.MDCS=1;             // MDC register provides timebase(1) or PDCx provides timebase(0)
940     PWMCON2bits.MDCS=1;             // MDC register provides timebase(1) or PDCx provides timebase(0)
941     PWMCON3bits.MDCS=1;             // MDC register provides timebase(1) or PDCx provides timebase(0)
942
943     PWMCON1bits.DTC=0;             // DTC<1:0> Dead time control Positive(0) Negative(1) or Disabled(2)
944     PWMCON2bits.DTC=0;             // DTC<1:0> Dead time control Positive(0) Negative(1) or Disabled(2)
945     PWMCON3bits.DTC=0;             // DTC<1:0> Dead time control Positive(0) Negative(1) or Disabled(2)
946
947     IOCON1bits.OVRDAT=0b00;        // PWM pin override output values <1:0> PWMH=OVRDAT<1> and PWML=OVRDAT<0>
948     IOCON2bits.OVRDAT=0b00;        // PWM pin override output values <1:0> PWMH=OVRDAT<1> and PWML=OVRDAT<0>
949     IOCON3bits.OVRDAT=0b00;        // PWM pin override output values <1:0> PWMH=OVRDAT<1> and PWML=OVRDAT<0>
950
951     IOCON1bits.OVRENH=1;           // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
952     IOCON1bits.OVRENL=1;           // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
953     IOCON2bits.OVRENH=1;           // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
954     IOCON2bits.OVRENL=1;           // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
955     IOCON3bits.OVRENH=1;           // PWMH pin override output value =OVRDAT<1> (1) or PWM gen output(0)
956     IOCON3bits.OVRENL=1;           // PWML pin override output value =OVRDAT<0> (1) or PWM gen output(0)
957
958     IOCON1bits.PENH=0;             // PWMxH pin is enabled(1, PWM) or disabled(0, general i/o)
959     IOCON2bits.PENH=0;             // PWMxH pin is enabled(1, PWM) or disabled(0, general i/o)
960     IOCON3bits.PENH=0;             // PWMxH pin is enabled(1, PWM) or disabled(0, general i/o)
961     IOCON1bits.PENL=0;             // PWMxL pin is enabled(1, PWM) or disabled(0, general i/o)
962     IOCON2bits.PENL=0;             // PWMxL pin is enabled(1, PWM) or disabled(0, general i/o)
963     IOCON3bits.PENL=0;             // PWMxL pin is enabled(1, PWM) or disabled(0, general i/o)
964
965     TRISEbits.TRISE0=0;            // RE0 (pin 26) as output(0)
966     TRISEbits.TRISE1=0;            // RE1 (pin 25) as output(0)
967     TRISEbits.TRISE2=0;            // RE2 (pin 24) as output(0)
968     TRISEbits.TRISE3=0;            // RE3 (pin 23) as output(0)
969     TRISEbits.TRISE4=0;            // RE4 (pin 22) as output(0)
970     TRISEbits.TRISE5=0;            // RE5 (pin 21) as output(0)
971
972     IOCON1bits.PMOD=0;             // PWMH/PWML pair mode configuration Complementery(0) Independent(1) push-pull(2)
973     IOCON2bits.PMOD=0;             // PWMH/PWML pair mode configuration Complementery(0) Independent(1) push-pull(2)
974     IOCON3bits.PMOD=0;             // PWMH/PWML pair mode configuration Complementery(0) Independent(1) push-pull(2)
975
976     IOCON1bits.OSYNC=1;            // PWM override update at end of PWM period(1) or CPU clock boundary(0)
977     IOCON2bits.OSYNC=1;            // PWM override update at end of PWM period(1) or CPU clock boundary(0)
978     IOCON3bits.OSYNC=1;            // PWM override update at end of PWM period(1) or CPU clock boundary(0)
979
980
981 //enable PWM controller(1) SET THIS LAST
982     PTCONbits.PTEN=1;               // Enable(1) or Disable(0) PWM timebase
983
984 }
```

```
985
986
987
988 //---------------------------------------------------------------------------
989 // Initilize UART for serial communication
990 //---------------------------------------------------------------------------
991 void uartsetup(void){
992     // Set Up UART TX side
993
994     TRISEbits.TRISE7=0;             // Alt UART TX as output(0)
995     TRISEbits.TRISE6=1;             // Alt UART RX as input(1)
996
997     U1BRG=194;                    // 9600Baud for tcy=30MHz (frc hi and 32x pll)
998
999     U1MODE=0x8000;                  // Enable, 8data, no parity, 1 stop
1000     U1STAbits.URXISEL=0b00;        // URXISEL<1:0>: Receive Interrupt Mode Selection bit
1001                                 // (Interupt when 11=>buffer full, 10=>buffer 3/4 full, 0x=>character is received)
1002     U1MODEbits.ALTIO = 1;          // Alt UART io pins
1003     U1MODEbits.UARTEN = 1;         // UART enabled(1) or disabled(0)
1004     U1STAbits.UTXEN =1;            // UART transmitter enabled(1) or disabled(0) (UARTEN=1 must be set prior to this)
1005
1006     //Receiver interupt initialize
1007     IFS0bits.U1RXIF = 0;           // set interupt flag to 0
1008     IPC2bits.U1RXIP = 4;           // set priority level 7=>highest, 1=>lowest, 0=>disabled
1009     IEC0bits.U1RXIE = 1;           // enable reciever interupt
1010 }
1011
1012
1013 //---------------------------------------------------------------------------
1014 // Setup ADC to scan analog inputs
1015 //---------------------------------------------------------------------------
1016
1017 void adcsetup(void) {
1018
1019     ADPCFG=0b1111111111110000;      // PCFG[15..0] A/D Port Configuration Control bits enable(0) pin as ADC or disable(1) for digital I/O
1020     TRISB =0b1111111111111111;      // TRISB[15..0] set to 1 to connect pin tri-state buffer to input
1021
1022     ADCONbits.ADSIDL=0;             // Stop(1) in Idle Mode or run(0)
1023     ADCONbits.FORM=0;               // Data Output Format bit fractional(1) or decimal(0)
1024     ADCONbits.EIE=0;                // Early Interrupt Enable bit after first conversion(1) or second(0)
1025     ADCONbits.ORDER=0;              // Conversion Order bit first chanels to convert are odd(1) or even(0)
1026                                 // in the dsPIC30f2020 ADC conversions are preformed in pairs 01,23,45 etc
1027                                 // after conversion they show up sequentialy in the ADCBUFx variable
1028                                 // this sets wether the odd or even channel is converted first
1029     ADCONbits.SEQSAMP=1;            // Sequential Sample Enable(1) or disable(0) controlles if set and holds of the ADC
1030                                 // pair are triggered at the same time(first conversion) or during seperate conversions
1031     ADCONbits.ADCS=0b011;               // ADCS[2..0]A/D Conversion Clock Divider Select bits(0-7) => Fadc/(4+2*ADCS)
1032     ADSTAT=0;                     // [5..0] Conversion Data for Pair #x Ready bits
1033
1034
1035 //  ADCPC0bits.TRGSRC0=0b00001;     //Trigger 0 Source Selection bits for channels AN0 and AN1
1036 //  ADCPC0bits.TRGSRC1=0b00001;     //Trigger 1 Source Selection bits for channels AN2 and AN3
1037     ADCPC0bits.TRGSRC0=0b01100;     //Trigger 0 Source Selection bits for channels AN0 and AN1
1038     ADCPC0bits.TRGSRC1=0b01100;     //Trigger 1 Source Selection bits for channels AN2 and AN3
1039                                 //ADC trigger selection
1040                                 //0b00001 = software trigger
1041                                 //      For SW trigger, set ADCPC0bits.SWTRG1=1;
1042                                 //0b00010 = Global software trigger selected
1043                                 //      For global SW trigger set ADCONbits.GSWTRG=1; must be cleared afer conversion
1044                                 //0b00011 = PWM Special Event Trigger selected
1045                                 //0b00100 = PWM generator #1 trigger selected
1046                                 //0b01100 = Timer #1 period match
1047                                 //0b01101 = Timer #2 period match
1048
1049     ADCPC0bits.IRQEN1=1;            //Interrupt Request Enable of channels AN3 and AN2 Enable(1) or disable(0)
1050     ADCPC0bits.IRQEN0=1;            //Interrupt Request Enable of channels AN1 and AN0 Enable(1) or disable(0)
1051
1052
1053     IFS0bits.ADIF=0;                // Clear(0) ADC Interrupt Flag
1054     IPC2bits.ADIP=6;                // set priority level 7=>highest, 1=>lowest, 0=>disabled
1055     IEC0bits.ADIE=1;                // ADC Interrupt Enable(1) or Disable(0)
1056
1057     ADCONbits.ADON=1;               // A/D Operating Mode bit Enable(1) or Disable(0) ADC operation
1058 }
1059
```

```
1060
1061
1062
1063 //--------------------------------------------------------------------------------
1064 // Sets up timers and timer interupts
1065 //--------------------------------------------------------------------------------
1066 void timersetup(void) {
1067
1068     SRbits.IPL = 3;             // CPU interupt priority levels
1069     INTCON1bits.NSTDIS=1;           // Disable(1) or enable(0) nested interupts
1070
1071 // Set up TMR1 for ADC trigger
1072     T1CONbits.TCS=0;            // 1=> external clock from TxCK, 0=> internal clk
1073     T1CONbits.TCKPS=0b01;       // prescale: 0b00=>clk/1, 0b01=>clk/8, 0b10=>clk/64, 0b11=>clk/256
1074     T1CONbits.TGATE=0;          // gated time acculation: 1=>enable, 0=>disable
1075     T1CONbits.TSIDL=1;          // stop timer in idle: 1=>disable, 0=>enable
1076     TMR1=0;                     // clear timer register
1077 // Set up TMR1 interupts
1078     PR1=10000;                  // Interrupt after this many cycles (testing)
1079     IPC0bits.T1IP = 6;          // set priority level 7=>highest, 1=>lowest, 0=>disabled
1080     IFS0bits.T1IF = 0;          // clear interrupt flag, set when timer=PRx
1081     T1CONbits.TON=1;            // 1=> start timer, 0=> stop timer
1082     IEC0bits.T1IE = 1;          // 1=> enable interrupts, 0=> disable interupts
1083
1084 // Set up TMR2 for PWM pulse time
1085     T2CONbits.TCS=0;            // 1=> external clock from TxCK, 0=> internal clk
1086     T2CONbits.TCKPS=0b10;       // prescale: 0b00=>clk/1, 0b01=>clk/8, 0b10=>clk/64, 0b11=>clk/256
1087     T2CONbits.TGATE=0;          // gated time acculation: 1=>enable, 0=>disable
1088     T2CONbits.TSIDL=1;          // stop timer in idle: 1=>disable, 0=>enable
1089     TMR1=0;                     // clear timer register
1090 // Set up TMR2 interupts
1091     PR2=469;                    // Interrupt after this many cycles (4692=10ms at clk/64, 1173=10ms at clk/256)
1092     IPC1bits.T2IP = 6;          // set priority level 7=>highest, 1=>lowest, 0=>disabled
1093     IFS0bits.T2IF = 0;          // clear interrupt flag, set when timer=PRx
1094     T2CONbits.TON=0;            // 1=> start timer, 0=> stop timer
1095     IEC0bits.T2IE = 0;          // 1=> enable interrupts, 0=> disable interupts
1096
1097 // Set up TMR3 for I/O control
1098     T3CONbits.TCS=0;            // 1=> external clock from TxCK, 0=> internal clk
1099     T3CONbits.TCKPS=0b11;       // prescale: 0b00=>clk/1, 0b01=>clk/8, 0b10=>clk/64, 0b11=>clk/256
1100     T3CONbits.TGATE=0;          // gated time acculation: 1=>enable, 0=>disable
1101     T3CONbits.TSIDL=1;          // stop timer in idle: 1=>disable, 0=>enable
1102     TMR3=0;                     // clear timer register
1103 // Set up TMR3 interupts
1104     IPC1bits.T3IP = 4;          // set priority level 7=>highest, 1=>lowest, 0=>disabled
1105     PR3=10000;                  // Interrupt after this many cycles
1106     IFS0bits.T3IF = 0;          // clear interrupt flag, set when timer=PRx
1107     T3CONbits.TON=1;            // 1=> start timer, 0=> stop timer
1108     IEC0bits.T3IE = 1;          // 1=> enable interrupts, 0=> disable interupts
1109
1110 }
1111
1112
1113 //--------------------------------------------------------------------------------
1114 // Prints signed int as ASCII string on serial port (-32768 to +32767)
1115 //--------------------------------------------------------------------------------
1116 void printSerial_signed_int(signed int val) {
1117     int index=0;
1118     char str6[6];                   //6 char string
1119
1120     //determins sign
1121     if(val<0){
1122         val=-val;
1123         str6[0]='-';
1124     }else{
1125         str6[0]='+';
1126     }
1127     //build num string
1128     for(index=5;index>=1;index--){
1129         str6[index]=(val%10 + '0');
1130         val=val/10;
1131     }
1132     //transmit string
1133     if((U1MODEbits.UARTEN == 1)&&(U1STAbits.UTXEN =1)){ // only transmit of uart and uart tx are enabled
1134     for(index=0;index<=6;index++){
```

```
1135        while(U1STAbits.UTXBF==1);
1136        U1TXREG=str6[index];
1137    }//end for
1138    }//end if
1139
1140 }
1141
1142 //-----------------------------------------------------------------------------
1143 // Prints unsigned int as ASCII string on serial port (0 to 65535)
1144 //-----------------------------------------------------------------------------
1145 void printSerial_int(unsigned int val) {
1146     int index=0;
1147     char str5[5];                  //5 char string
1148
1149     //build num string
1150     for(index=0;index<5;index++){
1151        str5[4-index]=(val%10 + '0');
1152        val=val/10;
1153     }
1154     //transmit string
1155     if((U1MODEbits.UARTEN == 1)&&(U1STAbits.UTXEN =1)){ // only transmit of uart and uart tx are enabled
1156     for(index=0;index<=4;index++){
1157        while((U1STAbits.UTXBF==1));//wait while buffer full or tx register not empty
1158        U1TXREG=str5[index];
1159        while((U1STAbits.TRMT==0));//wait while buffer full or tx register not empty
1160     }//end for
1161     }//end if
1162 }
1163
1164
1165 //-----------------------------------------------------------------------------
1166 // Prints literial ASCII value from variable on serial port
1167 //-----------------------------------------------------------------------------
1168 void printSerial_ASCII(char value) {
1169     if((U1MODEbits.UARTEN == 1)&&(U1STAbits.UTXEN =1)){ // only transmit of uart and uart tx are enabled
1170        while((U1STAbits.UTXBF==1));//wait while buffer full or tx register not empty
1171        U1TXREG=value;
1172        while((U1STAbits.TRMT==0));//wait while buffer full or tx register not empty
1173     }//end if
1174 }
1175
1176 //-----------------------------------------------------------------------------
1177 // Prints string of ASCII characters on serial port from RAM
1178 //-----------------------------------------------------------------------------
1179 void printSerial_str(char * str) {
1180     int i = 0;
1181
1182     if((U1MODEbits.UARTEN == 1)&&(U1STAbits.UTXEN =1)){ // only transmit of uart and uart tx are enabled
1183     while(str[i]!=0) {
1184        while((U1STAbits.UTXBF==1));//wait while buffer full or tx register not empty
1185        U1TXREG=str[i];
1186        i++;
1187        while((U1STAbits.TRMT==0));//wait while buffer full or tx register not empty
1188     }//end while
1189     }//end if
1190 }
1191
1192 //-----------------------------------------------------------------------------
1193 // Prints string of ASCII characters on serial port from program memory
1194 //-----------------------------------------------------------------------------
1195 void printSerialROM_str(const char * str) {
1196     int i = 0;
1197
1198     if((U1MODEbits.UARTEN == 1)&&(U1STAbits.UTXEN =1)){ // only transmit of uart and uart tx are enabled
1199     while(str[i]!=0) {
1200        while((U1STAbits.UTXBF==1));//wait while buffer full or tx register not empty
1201        U1TXREG=str[i];
1202        i++;
1203        while((U1STAbits.TRMT==0));//wait while buffer full or tx register not empty
1204     }//end while
1205     }//end if
1206 }
1207
1208 //-----------------------------------------------------------------------------
1209 // Convert int into a binary ascii string
```

```
1210 //---------------------------------------------------------------------------
1211 void printSerial_binary_int(unsigned int val){
1212     int index=0;
1213
1214     //transmit string
1215     if((U1MODEbits.UARTEN == 1)&&(U1STAbits.UTXEN =1)){ // only transmit of uart and uart tx are enabled
1216     for(index=15;index>=0;index--){
1217         U1TXREG=(testbit(val,index)+'0');
1218         while((U1STAbits.UTXBF==1)|( U1STAbits.TRMT==0));//wait while buffer full or tx register not empty
1219     }//endfor
1220     }//end if
1221 }
1222
1223
1224 //---------------------------------------------------------------------------
1225 // Convert numeric ascii string to an unsigned int
1226 //---------------------------------------------------------------------------
1227 int str_to_int(char * str){
1228 int result=0;
1229 int index=0;
1230 int digit=0;
1231
1232     for ( index = 0; (str[index]!='\0')&(index<=4); ++index ){
1233         digit=str[index] - '0';             //get the ascii char valua and convert to numeric digit
1234         result=10*result + digit;           //add the digit to the result and multiply old result by 10
1235     }
1236
1237 return(result);
1238 }
1239
1240
1241 //---------------------------------------------------------------------------
1242 // Parse received string for commands
1243 //---------------------------------------------------------------------------
1244 //  Dty=100;                        //duty cycle = 100% (varies between 50-100%)
1245 //  PWMfreq=200;                    //PWMfreq controls PWM frequency 200 => 20kHz
1246
1247 void parse_string(char * str){
1248 int i=0;                            //string index
1249 int cnum=0;                         //converted number
1250 char str6[6];                       //6 char string
1251
1252
1253 //if numeric input, then expect to set a variable
1254 if(str[0] == '$'){          //expected first char for numeric input
1255     for(i=0;(str[i]!='\0')&(i<=4);i++){
1256         str6[i]=str[(i+1)];
1257     }//end for
1258         str6[5]='\0';
1259
1260         cnum=str_to_int(str6);      //converted number
1261
1262         printSerialROM_str("\r\nNumber received= \0");
1263         printSerial_int(cnum);      //display conveted number
1264     //set pwm frequency
1265         if((str[7]=='f')&(str[8]=='r')&(str[9]=='e')&(str[10]=='q')&(str[11]=='\0')){
1266             if((cnum>=Fres)&(cnum<=Fmax)){
1267             PWMfreq=cnum;
1268             printSerialROM_str(freqString);
1269             printSerial_int(PWMfreq);
1270             printSerialROM_str(NL);
1271             }
1272             else{
1273             printSerialROM_str(InvalidNumberString);        // print string to terminal
1274             }
1275         }//end if
1276     //set duty cycle
1277         else if((str[7]=='d')&(str[8]=='t')&(str[9]=='y')&(str[10]=='\0')){
1278             if((cnum>=Dtymin)&(cnum<=Dtymax)){
1279             Dty=cnum;
1280             printSerialROM_str(dtyString);
1281             printSerial_int(Dty);
1282             printSerialROM_str(NL);
1283             }
1284             else{
```

```
1285            printSerialROM_str(InvalidNumberString);        // print string to terminal
1286            }
1287        }//end if
1288    //set pulse time
1289        else if((str[7]=='p')&(str[8]=='t')&(str[9]=='i')&(str[10]=='m')&(str[11]=='e')&(str[12]=='\0')){
1290            if((cnum>=1)&(cnum<=100)){
1291            Ptime=cnum;
1292            printSerialROM_str(ptimeString);
1293            printSerial_int(Ptime);
1294            printSerialROM_str(NL);
1295            }
1296            else{
1297            printSerialROM_str(InvalidNumberString);        // print string to terminal
1298            }
1299        }//end if
1300    //set voltage output setpoint
1301        else if((str[7]=='v')&(str[8]=='s')&(str[9]=='e')&(str[10]=='t')&(str[11]=='\0')){
1302            if((cnum>=0)&(cnum<=Vlimit)){
1303            Vset=cnum;
1304            printSerialROM_str(vsetString);
1305            printSerial_int(Vset);
1306            printSerialROM_str(NL);
1307            }
1308            else{
1309            printSerialROM_str(InvalidNumberString);        // print string to terminal
1310            }
1311        }//end if
1312    //set voltage limit setpoint
1313        else if((str[7]=='v')&(str[8]=='l')&(str[9]=='i')&(str[10]=='m')&(str[11]=='i')&(str[12]=='t')&(str[13]=='\0')){
1314            if((cnum>=0)){
1315            Vlimit=cnum;
1316            printSerialROM_str(vlimitString);
1317            printSerial_int(Vlimit);
1318            printSerialROM_str(NL);
1319            if(Vset>Vlimit){Vset=Vlimit;}                   //reset Vset to Vlimit in required
1320            }
1321            else{
1322            printSerialROM_str(InvalidNumberString);        // print string to terminal
1323            }
1324        }//end if
1325    //set Kpfreq
1326        else if((str[7]=='k')&(str[8]=='p')&(str[9]=='\0')){
1327            if((cnum>=0)){
1328            Kpfreq=cnum;
1329            printSerialROM_str(kpString);
1330            printSerial_int(Kpfreq);
1331            printSerialROM_str(NL);
1332            }
1333            else{
1334            printSerialROM_str(InvalidNumberString);        // print string to terminal
1335            }
1336        }//end if
1337    //set Kifreq
1338        else if((str[7]=='k')&(str[8]=='i')&(str[9]=='\0')){
1339            if((cnum>=0)){
1340            Kifreq=cnum;
1341            printSerialROM_str(kiString);
1342            printSerial_int(Kifreq);
1343            printSerialROM_str(NL);
1344            }
1345            else{
1346            printSerialROM_str(InvalidNumberString);        // print string to terminal
1347            }
1348        }//end if
1349    //if no valid input
1350        else{
1351            printSerialROM_str(InvalidString);              // print received string to terminal
1352        }//end else
1353
1354    }//end numeric set if
1355
1356 //manual pulse trigger
1357    else if((str[0]=='p')&(str[1]=='u')&(str[2]=='l')&(str[3]=='s')&(str[4]=='e')&(str[5]=='\0')){
1358        if((IOCON1bits.OVRENH==1)&(PSCONFIG.PWMlockout==0)){        // If the timer isnt running and lockout disabled
1359            printSerialROM_str(PulseString);
```

```
1360            if((Vin<Vinmin)&(PSCONFIG.dummypulse==0)){
1361                PSCONFIG.PSerror=1;             //set power supply error flag
1362                PSCONFIG.lowVin=1;              //set DC link undervolt flag
1363                PWMstop();                      //shut down power supply
1364            }
1365            else{
1366                PWMstart();             // start pulse timer and enable outputs
1367            }
1368        }
1369    }//end pulse trigger else if
1370
1371 //software reset
1372    else if((str[0]=='r')&(str[1]=='e')&(str[2]=='s')&(str[3]=='e')&(str[4]=='t')&(str[5]=='\0')){
1373        RCON=0x0000;
1374        printSerialROM_str("\r\n\r\nRCON=    \0");
1375        printSerial_binary_int(RCON);
1376        printSerialROM_str(ResetInitString);
1377        asm("RESET");
1378    }//end else if
1379
1380 //display data
1381    else if((str[0]=='d')&(str[1]=='a')&(str[2]=='t')&(str[3]=='a')&(str[4]=='\0')){
1382        ProgCON.PSDataReady=1;      //print PWM data to console
1383    }//end  else if
1384
1385 //dummy pulse
1386    else if((str[0]=='d')&(str[1]=='p')&(str[3]=='\0')){
1387        PSCONFIG.dummypulse=1;      //set dummypulse flag
1388        printSerialROM_str("\r\nDummypulse selected, saftey checks disabled \0");
1389    }//end  else if
1390
1391 //help menu
1392    else if((str[0]=='h')&(str[1]=='e')&(str[2]=='l')&(str[3]=='p')&(str[4]=='\0')){
1393    printSerialROM_str("\r\nValid Commands:\r\n$xxxxx freq  (sets PWM freq 00200=20kHz, 00250>=freq>=00150)\0");
1394    printSerialROM_str("\r\n$xxxxx dty   (sets PWM duty cycle 00100=100%, 00100>=dty>=00050)\0");
1395    printSerialROM_str("\r\n$xxxxx ptime (sets pulse time 00100=10ms, 00100>=ptime>=00001)\0");
1396    printSerialROM_str("\r\n$xxxxx vset  (sets output voltage 08000=>80kV, 00000<=vset<=Vlimit)\0");
1397    printSerialROM_str("\r\n$xxxxx vlimit(sets crowbar voltage limit 08500=>85kV, 00000<=Vlimit)\0");
1398    printSerialROM_str("\r\npulse (triggers output pulse)\r\ndp(enables dummy pulse)\0");
1399    printSerialROM_str("\r\ndata  (prints out power supply operation data)\0");
1400    printSerialROM_str("\r\nreset (dsPIC hardware reset)\r\n\0");
1401    }//end help else if
1402
1403        //for all other input
1404    else {
1405        printSerialROM_str(InvalidString); // print string to terminal
1406    }//end else
1407
1408    printSerialROM_str(NL);
1409 }//end of sub
1410
```