High Efficiency DC/AC Power Converter for Photovoltaic Applications

by

Aleksey Trubitsyn

B.S., Boston University (2008)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

Massachusetts Institute of Technology

June 2010

Author_____
Department of Electrical Engineering and Computer Science
May 21, 2010

Certified by_____
David J. Perreault
Associate Professor of Electrical Engineering
Thesis Supervisor

Accepted by_____
Terry P. Orlando
Chairman, Department Committee on Graduate Students

High Efficiency DC/AC Power Converter for Photovoltaic Applications

by

Aleksey Trubitsyn

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2010, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

This thesis presents the development of a microinverter for single-phase photovoltaic applications that is suitable for conversion from low-voltage (25-40V) DC to high voltage AC (e.g. $240V_{AC,RMS}$). The circuit topology is based on a full-bridge series resonant inverter, a high-frequency transformer, and a novel half-wave cycloconverter. The operational characteristics are analyzed, and a multidimensional control technique is utilized to achieve high efficiency, encompassing frequency control and inverter and cycloconverter phase shift control. An experimental prototype is demonstrated in DC/DC conversion mode for a wide range of output voltages. The proposed control strategy is shown to allow for accurate power delivery with minimal steps taken towards correction. The prototype achieves a CEC averaged efficiency of approximately 95.1%. Guidelines for optimization are presented along with experimental results which validate the method.

Thesis Supervisor: David J. Perreault
Title: Associate Professor of Electrical Engineering

Acknowledgements

I want to thank Brandon Pierquet for answering all of my questions about this project, at absolutely any time, and no matter how trivial or off-point they were. I sincerely appreciate his endless willingness to help me understand the concepts of this project and I certainly would not have been able to get through this thesis without his help. I want to thank Alexander Hayman for all the practical information that he taught me in the several months that we worked together. I learned most of what I know about dealing with hardware from him and this was a crucial skill to have for advancing the prototype. I want to thank Professor David Perreault for bringing me onto this project and being very guiding, knowledgeable and extremely supportive during the last two years. Also very noteworthy is his willingness to stay late in lab, beyond regular hours, and helping us debug our projects. Many thanks go out to Enphase Energy and Dr. Landsman for their funding of my education. I want to thank the many members of LEES that made the lab a fun, coherent and productive work environment: George Hwang, Grace Cheung, Jackie Hu, Makiko Wada, Ben Cannon, Ian Smith, and Justin Burkhart. I also appreciate the numerous times that Anthony Sagneri and Robert Pilawa offered very helpful advice on my work.

Many thanks to Allan Hutchison-Maxwell for proofreading a large portion of this thesis!

Finally, I want to extend **special** thanks to my girlfriend, Maria, and my family, Anton, Mariam and Yuriy for their continued love and support throughout my college career!!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Solar inverters

THE market for rooftop photovoltaic (PV) panel installations is growing rapidly, and with it grows the demand for inverters to interface with the grid [1–3]. The inverter is placed between a PV array and the grid, as shown on Fig. 1.1. The inverter serves the dual purposes of drawing maximum DC power from the array (by imposing an optimal operating condition onto the solar panel through maximum power point tracking (MPPT)) and delivering this power into the AC voltage of the power line (transforming voltage levels and converting from DC to AC). The power loss in the inverter reduces the total energy extraction of the solar installation as a whole. Minimizing this loss as well as minimizing the inverter size and cost are important directions in the process of making solar energy a viable source for the future.

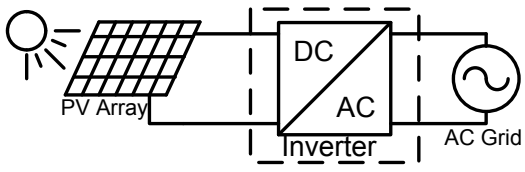


Figure 1.1: Solar inverter placement.

Multiple inverter system architectures exist, of which two are the most widely considered. The first approach involves a single grid-tie inverter connected to a series string of PV panels. The string has enough cells to provide a high DC voltage suitable for interfacing to the grid without requiring a large voltage transformation from the inverter, simplifying inverter design. Moreover, it controls power conversion to a single large power circuit (e.g. several kW or more) which can be loss effective. However, there are at least two limitations to this approach. First is that the maximum power point tracking is performed for the entire series string of panels, which is not optimal for the individual panels given variations among panels and variations in illumination of each panel [2, 8]. Secondly, a permanent defect or even a temporary shade to a single panel in a series string which is controlled by a single inverter limits the performance of the entire installation and can even damage the working cells (through current limit and over voltage respectively) [2, 8, 9]. Overall the decrease in energy extraction due to partial shading can exceed 10% over a yearly average [9].

Another approach to managing solar arrays is through the use of Module Integrated Converters (also known as microinverters) - power converters that are rated for only a few hundreds of watts each, and directly tie an individual low-voltage panel to the AC grid. Connecting each solar panel via its own micro inverter can improve the overall performance of an installation. One advantage comes from MPPT of each panel's output, which yields greater energy extraction than centralized MPPT of a series-connected string of modules can achieve. Additionally, the performance of the entire installation is no longer limited by failures in individual modules. Although the obvious disadvantage of needing more (although physically smaller) inverter modules, each providing much greater voltage transformation, exists for this approach, the trends in residential installations are expected to continue heading in that direction [1, 2, 8, 10]. In this application, efficiency and compactness are the driving design considerations [10].

1.2 Thesis scope and organization

This thesis will describe a microinverter design suitable for high efficiency DC/AC power conversion from a low-voltage (25-40V) DC input to a single-phase 240V AC grid connection. The basic concept of this design was originally presented in [14]. However, the focus of that work was slightly different: the goal was mainly to get the first prototype working with a reasonable efficiency and less focus was placed on the optimization. Here, we explore optimization of the design for high efficiency and present more thorough experimental results of a more complete prototype [1]. The specific design of the MPPT controller will not be taken into account, although it is readily implemented with the proposed converter design. Also not addressed are many of the practical issues that must be solved in order to make this a commercial product (such as those mentioned in [2]). The thesis is organized as follows. Chapter 2 will present the circuit topology along with some background topics which may be useful for understanding the design and the requirements on it. Chapter 3 will describe all components in more detail, including the available control input and the ways by which the losses can be estimated. Chapter 4 will present the method which is used to control the inverter and show the procedure for optimizing the design given certain constraints. Chapter 5 will present the experimental results, including overall efficiency and the power loss distribution. Chapter 6 will conclude the thesis and present the next logical steps for this project.

[1] It is more complete in that Hayman's work described a prototype with some components shorted out. It is still incomplete in that it does not demonstrate complete DC/AC conversion into the grid, but rather a series of static DC/DC operating conditions, which span the entire AC voltage range. This is, nevertheless, sufficient to demonstrate the capability and efficiency of the approach.

Chapter 2

System Overview

THIS chapter will present the inverter design requirements and describe the strategies by which high efficiency is achieved. Additionally, the inverter topology will be presented and briefly discussed, while setting the stage for the more detailed analysis of chapter 3. The discussion will begin by considering a general design technique, known as soft switching, which can used in switching power converters to increase efficiency. Next, the specifics of this system's goals will be defined and the circuit topology that satisfies these requirements will be presented. Lastly, the implications of soft switching will be considered with respect to the circuit.

2.1 Soft switching

There exists an extensive body of work on DC to AC power converters specifically for grid tied PV applications. A thorough overview and a topology classification is provided in [2, 4, 10, 13]. Topologies for different power levels and numbers of phases at the output are also presented in [7, 8, 11, 12], for example. Among techniques to increase efficiency, soft switching can significantly reduce the loss associated with switching a semiconductor device. Zero voltage and zero current switching (ZVS and ZCS respectively) are two such techniques, which provide a means of substantially

reducing loss associated with semiconductor device switching [6]. The idea behind these methods is to toggle a device when it is blocking (instantaneously) close to zero volts (for ZVS) or when it is conducting (also instantaneously) very little current (for ZCS). This is compared to the hard switching case where by a device changes states when it is either blocking the full voltage (e.g. typically an input or an output voltage in a circuit), or when it is conducting the full current seen in operation of the circuit. As switching loss results from an instantaneous overlap of voltage across and current through a device, forcing one of those terms to be close to zero serves to minimize the overlap product. On the other hand, soft switching often increases the complexity of determining the timing in a circuit and could require snubbers to satisfy certain constraints.

A review of common soft switched topologies for DC-AC conversion is presented in [5]. Several of the presented topologies utilize resonant conversion to achieve soft switching. The general approach is to use a resonant network of passive components such that switching in a certain frequency range will naturally support soft switching (although typically a few other operating constraints need to be met). Some difficulties arise when utilizing resonant conversion and these include: ensuring proper operation over a wide load range (if that is a needed feature), possibly increased device stresses due to resonant elements' voltage swings, and increased conduction losses [5, 15]. However, the overall tradeoffs can favor resonant topologies, especially since they often are able to absorb (and in fact rely on) the parasitic properties of the semiconductor devices.

Parameter	Value
Input Voltage	25-40V, DC
Output Voltage	240 V_{RMS}, AC 60Hz (1 phase)
Average Power	0-175 W
Efficiency	$\geq$ 96% CEC average
Power Factor	≈ 1
Switching Frequency	desirable to be above 60 kHz for size considerations

Table 2.1: Design specifications

Quantity	Value	Symbol
Switching frequency	$\geq$ 60kHz	f_{SW}
Switching angular frequency	$2\pi f_{SW}$ rad/s	ω_{SW}
Switching period	$1/f_{SW}$ s	T_{SW}
Line frequency	60 Hz	f_{line}
Line angular frequency	$2\pi f_{line}$ rad/s	ω_{line}
Line period	$1/f_{line}$ s	T_{line}
Instantaneous time	s	t, time

Table 2.2: Notations for frequencies and times of interest.

2.2 Design targets

The development of this inverter will be basis for a commercial product and the design goals are dictated by its market. Table 2.1 summarizes the input, output, and operational requirements. Although galvanic isolation between input and output is not required, it could be a desireable feature for conforming to safety standards for commercial inverters (as some requirements are simpler for isolated designs). The efficiency target is given using the California Energy Commission (CEC) metric for averaging efficiency across a range of output powers (discussed in 2.2.2). Table 2.2 summarizes the notation that will be used throughout the rest of this thesis.

2.2.1 Grid tie power delivery

The output voltage is fixed by the utility grid (given by (2.1)) and the inverter has to synchronize to it.

$$V_{out}(t) = V_{RMS}\sqrt{2}\sin(\omega_{line}t) = V_{peak}\sin(\omega_{line}t) \tag{2.1}$$

We principally consider the case $V_{RMS} = 240V$. Operating at unity power factor means that the current and the voltage are in phase with each other as if the line is a resistor of some equivalent value. The instantaneous output power (power into the line) is then given by (2.2).

$$P_{out}(t) = \frac{(V_{peak}\sin(\omega_{line}t))^2}{R_{equivalent}} = 2P_{ave}\sin^2(\omega_{line}t) \tag{2.2}$$

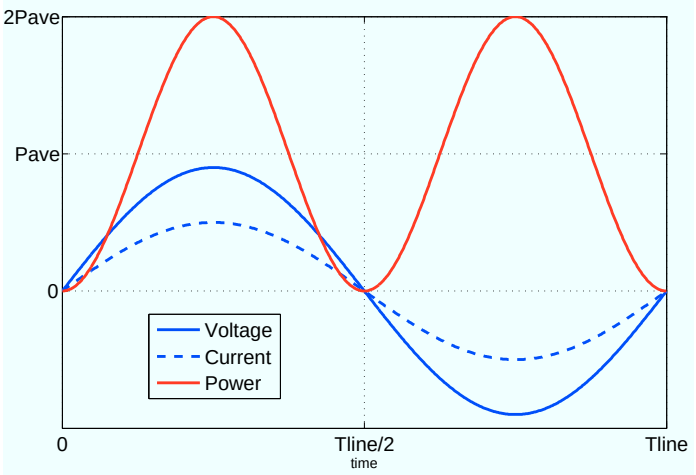


Figure 2.1: AC grid power and voltage

The quantity P_{ave} represents the power delivered into the grid, averaged over a line cycle. This quantity can change (e.g. based on solar panel insolation and shading and determined by MPPT control) and one of the design requirements is that this average value can change in the range between 0 and 175W. Figure 2.1 shows the relationship between the AC line voltage, current, and power. The maximum value of P_{out} is equal to twice its average value.

2.2.2 CEC efficiency

The CEC calls for a comprehensive reporting procedure for these types of inverters. Table 2.3 shows the efficiency test matrix - combinations of input and output voltages and output powers for which efficiency must be measured. For the purposes of this project, **input** voltages are defined as: $V_{min} = 25$, $V_{max} = 40V$ and their average is $V_{nom} = 32.5V$. The nominal **output** AC RMS voltage is defined as $240V_{RMS}$ and this will also be equal to output V_{max} and V_{min}. The case of 100% power level corresponds to $P_{ave} = 175W$. There are certain test procedures that have to be obeyed for testing (see [16]), but after the entire test matrix has been obtained the following data is reported to characterize the efficiency of an inverter:

- The *test matrix* itself.
- The *peak efficiency* - the maximum value found in the text matrix.
- The *nominal average efficiency* - the average of the 9 efficiency values taken for tests A-C and average power levels of 50-100 %.
- The *weighted efficiency* - the efficiency computed by taking the weighted average of test case A. The weighting coefficients are given in 2.4.

Test	V_{DC}	V_{AC}	P_{ave} level, percent of maximum						
			100	75	50	30	20	10	5
A	V_{nom}	V_{nom}							
B	V_{max}	V_{nom}							
C	V_{min}	V_{nom}							
D	V_{min}	102% V_{max}							
E	V_{max}	98% V_{max}							

Table 2.3: CEC test matrix. Table source: [16].

Average Power (%)	100	75	50	30	20	10	5
Weighting coefficient	.05	.53	.21	.12	.05	.04	0.00

Table 2.4: CEC weighting coefficients for high-insolation environments. Table source: [16].

2.2.3 Switching cycle averaged quantities

The switching frequency is at least 1000 times greater than the line frequency. There are two main consequences of this for the purposes of analysis of the circuit at the switching time scale. The first is that the instantaneous (low frequency - f_{line}) power delivery to the line can be well approximated by the switching cycle-averaged high frequency (f_{SW}) power delivery, given by (2.3). As such, the term 'instantaneous power delivery' will hereafter refer to the average power over a switching cycle, or either side of (2.3). The second consequence is that the 60Hz output voltage is nearly constant during a switching cycle, and using a constant DC value during a cycle is a good approximation (this is really a result of averaging (2.1) over 1 switching cycle).

$$P_{out,f_{line}}(t) \approx \frac{1}{T_{SW}} \int_{t-T_{SW}/2}^{t+T_{SW}/2} P_{out,f_{SW}}(\tau) d\tau \tag{2.3}$$

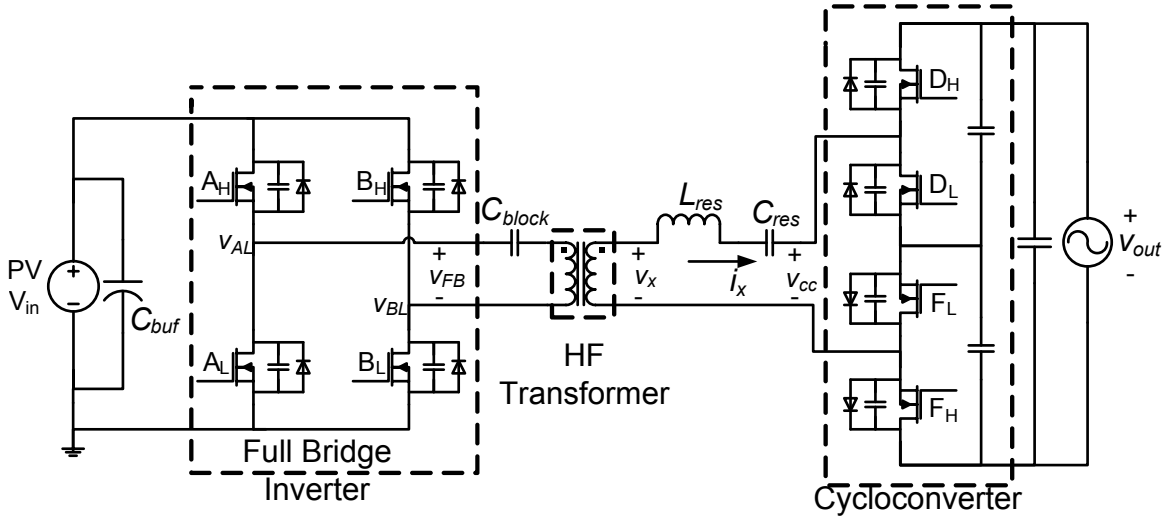


Figure 2.2: Circuit topology.

2.3 Circuit topology and overview

The architecture of the inverter is shown on fig. 2.2. It comprises a resonant inverter (switching at a high frequency), followed by a transformer to provide voltage level conversion and isolation, and a soft switched cycloconverter to down-convert to line frequency. Although this general architecture has long been known (e.g., [7]), it is perhaps the least explored of known approaches (e.g., see the topology review in [2, 4, 10, 13]). Control is achieved both through switching frequency control and timing (phase shift) control. Design decisions embodied in this topology focus on achieving high efficiency at small size while meeting the large voltage transformation requirement. The switching frequency determines the overall size of the system and the specified range keeps the dimensions to what is appropriate for an inverter in this application, while also meeting the power delivery constraints.

The resonant inverter, which consists of the full-bridge inverter and a series resonant LC network, converts the DC input voltage to a high-frequency quasi-sinusoidal waveform. The transformer provides voltage amplification and galvanic isolation, and the

cycloconverter steps down the frequency of the current from f_{SW} to f_{line}, to deliver power at the desired power factor into the output voltage source. The resonant inductor L_{res} and capacitor C_{res} may be placed on either side of the transformer. The capacitor (C_{block}) placed on the primary side of the transformer is not a resonant element, but merely ensures that no DC current is present that can possibly saturate the transformer. This capacitor is placed on the side of the transformer opposite to that of the resonant tank. Full-bridge inverter and half-wave cycloconverter topologies are selected because together they reduce the required transformer turns ratio (e.g., as compared to using a half-bridge inverter or a full-wave cycloconverter), thus improving achievable efficiency. Each MOSFET in the figure shows a capacitor and a diode connected in parallel with it. These represent the parasitic drain to source capacitance and the body diode of each device. They are shown as their role is vital to proper ZVS operation. By eliminating diode drops from main operation of the converter, and by operating all of the FET devices under ZVS, one can achieve low loss by scaling device areas up beyond that which is optimum for hard-switched topologies. Moreover, given an absence of diode drops in the main conversion path, the efficiency achievable with this topology is also expected to improve as the characteristics of resistive-drop devices continue to improve. Finally, we are able to absorb transformer leakage as part of the resonant inductor, facilitating at least partial component integration.

2.4 Operating a half-bridge under ZVS

Fig. 2.3 shows the setup of an idealized 'half-bridge' circuit. The grounded node is there to serve as a voltage reference for notational convenience and this node can be floating. Examination of fig. 2.2 reveals that there are four of these structures in the main system. The general case of what it means to have ZVS on such a structure

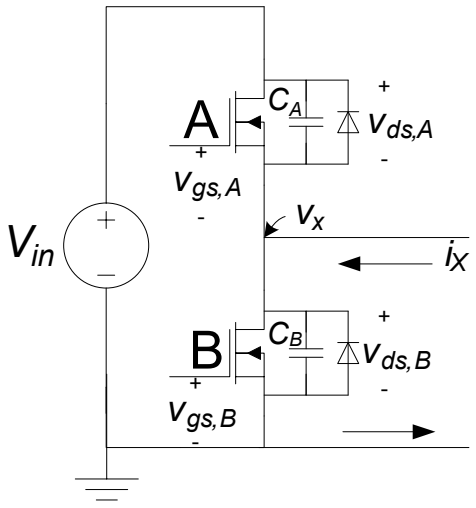


Figure 2.3: A MOSFET half-bridge.

is considered here. Current i_X will be defined positive as shown by the arrow, and v_X is the middle node voltage. Let $v_{gs,A}$ and $v_{gs,B}$ be the controlling (gate to source) voltages of MOSFETs A and B respectively and let $v_{ds,A}$ and $v_{ds,B}$ be the drain to source voltages of A and B respectively (therefore $v_{ds,A} = V_{in} - v_X$ and $v_{ds,B} = v_X$). Also let the parallel capacitances be of values C_A and C_B. For convenience, the FET which has its drain connected to the voltage source V_{IN} is going to be called the high side device of a half-bridge and the other FET will be called the low side.

Letting current be an ideal sinusoid for simplicity, with the positive direction indicated by the arrow of i_x, we are interested in what happens near the zero crossings of the current waveform. Figures 2.4 and 2.5 visualize what is happening to the voltages in this idealized half-bridge (an actual transition in an experimental circuit is shown in Fig. 2.6). Shortly before the zero crossing (interval 1), device B is fully on and is conducting the full current i_x. This means that $v_{gs,B}$ = high, $v_{ds,B} = 0$, $v_{gs,A}$ = low, $v_{ds,A} = V_{IN}$ (the last equality comes from assuming a negligible on-state resistive voltage drop in B, which is fine for the ZVS discussion). Interval 2 is identical to interval 1 in terms of voltages, however the current has crossed zero and became positive on 2. Interval 2 is concluded when device B is switched off. On interval 3,

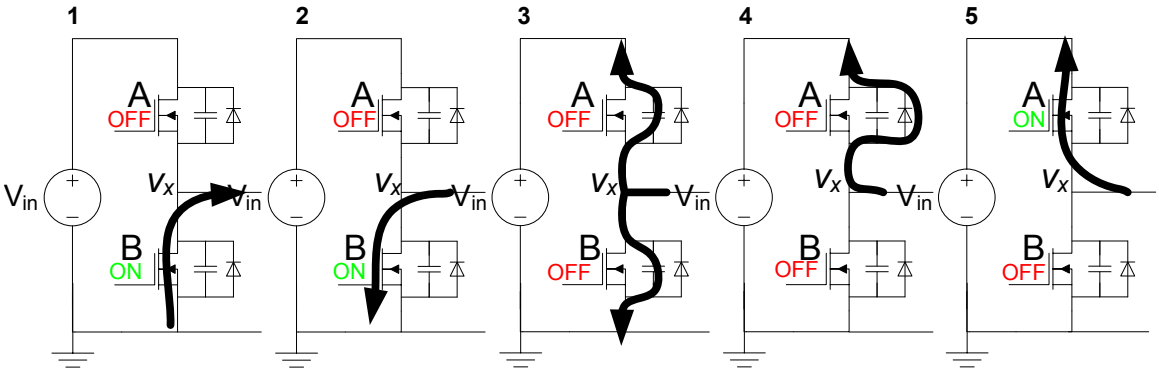


Figure 2.4: Half-bridge commutation period. Current flow is shown in dark curve with the arrow indicating positive direction. Figure adapted from [6].

both devices are off (so $v_{gs,B} = 0$ and $v_{gs,A} = 0$) and the current begins to travel through the capacitances C_A and C_B. At the start of interval 3, the voltage across C_A is V_{IN} and the voltage is 0 across C_B. Therefore, the positive current discharges C_A and charges C_B (from an AC perspective, capacitors C_A and C_B are effectively in parallel). The drain to source voltages are shown as having linear rise and fall on 2.5 to exaggerate the charging and discharging process and the actual shape might be different. This interval continues until C_A is discharged to a low voltage (nearly zero Volts) and C_B is charged to about V_{IN}. At this point the diode of device A starts to conduct and this marks the end of current commutation from B to A and the beginning of interval 4. At any point along interval 4 the entire current is carried by the diode of A with a small voltage drop (especially if compared to V_{IN}) and device A may be turned on with minimal voltage-current product overlap loss. Interval 5 begins when A is finally turned on and it is marked by the entire current being conducted through device A. Although resistive voltage drops are ignored on Fig. 2.4, it is often the case that the voltage drop due to diode conduction is higher than that due to FET conduction, and in that case the drain to source voltages are slightly different in intervals 4 and 5. The positive to negative transition of current is a very similar situation, except that the polarity of the current and the commutation order of the devices are reversed.

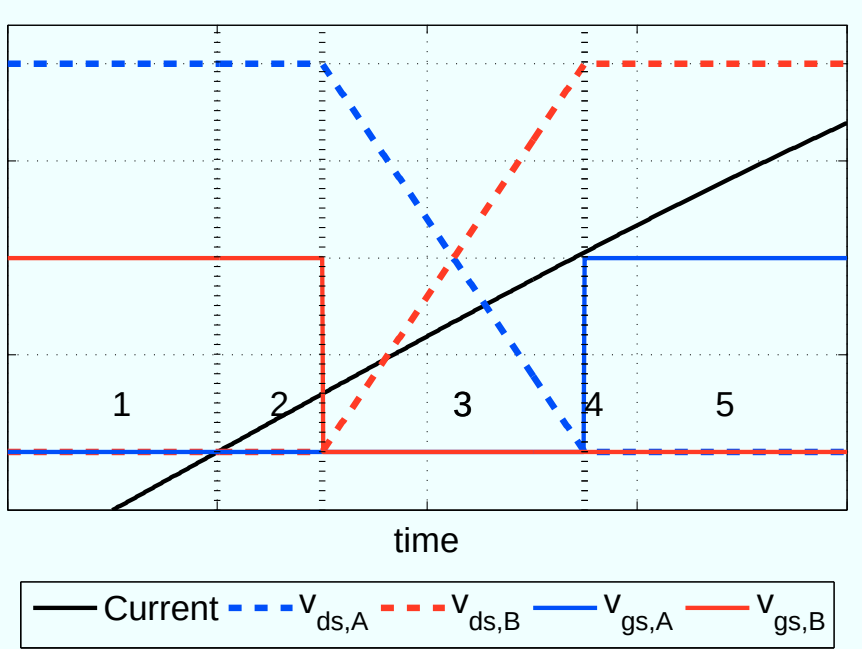


Figure 2.5: Voltages and current corresponding to each interval of Fig. 2.4.

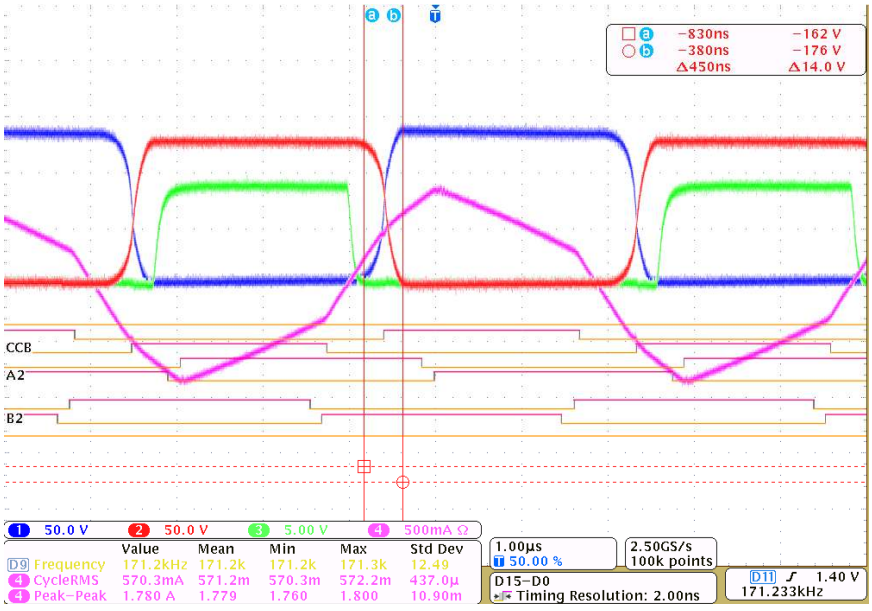


Figure 2.6: The waveforms of Fig. 2.5 as measured in a real circuit. Current is shown in purple, $v_{ds,A}$ is in red, $v_{ds,B}$ is in blue, $v_{gs,B}$ is in green.

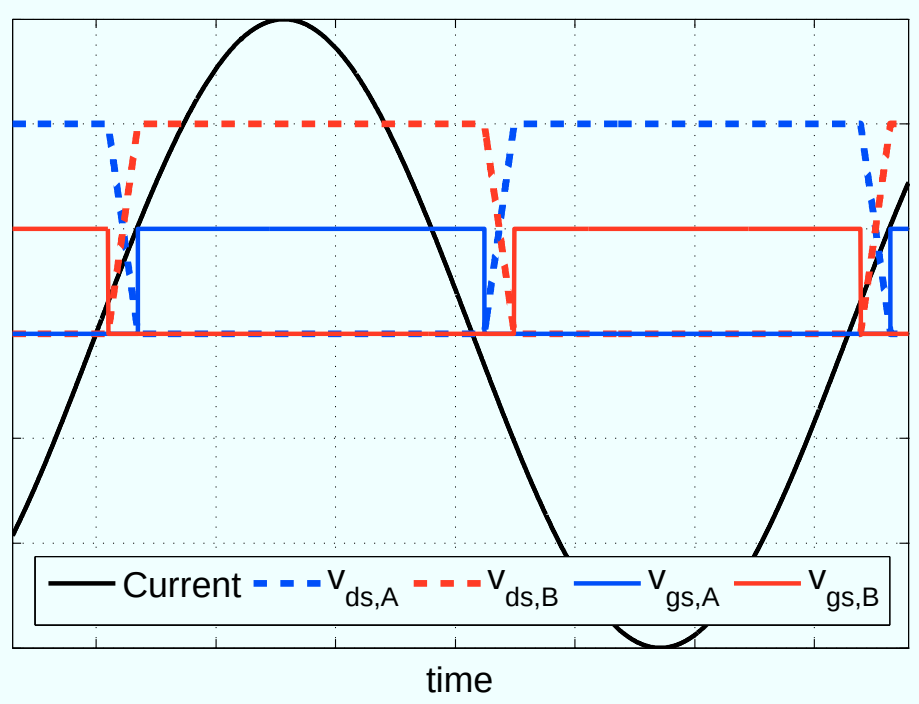


Figure 2.7: View over a switching period. Interval 4 is reduced to 0.

Theoretically, interval 4 is not needed for proper ZVS operation. Device A may be turned on at any time after C_B has discharged completely and any non-zero time interval during which the diode of A conducts results in an energy loss that is typically much higher than if the actual device A were conducting for the same time interval. Figure 2.7 shows the entire period of the current with both zero crossings' transition voltages. It also shows what ideal timing for turning on device A would look like. However, if some fault (even a slight miscalculation) in timing of the circuit causes device A to turn on before C_B is discharged, ZVS will be lost (or at least not utilized to its potential) and the overlap loss product and capacitive discharge loss will increase. In addition, depending on how far off mark the timing is, various kinds of ringing may result in the circuit which will contribute to loss in other components, and in some cases may damage the gate drivers and lead to system failure. Therefore it is often desirable to leave a non zero duration for interval 4 for 'safety' reasons. Another point to note is that interval 2 may begin earlier than shown. As long as the polarities in

each of the transitions are not reversed, the magnitude of the current at beginning of interval 2 may be anywhere from 0 to its peak. While ZVS is not explicitly effected by that, higher instantaneous current has several consequences:

- The loss in the diode increases (interval 4).
- The capacitances will be charged/discharged faster. The commutation time (interval 3) will decrease.
- The speed at which device B turns off becomes more important: higher current means higher losses due to the same amount of overlap in time.

The total duration of intervals 3 and 4 is referred to as 'deadtime' of this half-bridge. In general, to achieve highest efficiencies, interval 4 portions of the deadtime should be minimized (one of the FETs should be conducting instead of a diode whenever possible), with the minimal possible deadtime being equal to interval 3. Finally, the instantaneous power going into the half-bridge block is given by (2.4). Since v_X is also the voltage across device B, we will often look at low side FET's drain to source voltage and current into the drain to analyze the power flow between the various components of the inverter.

$$P_{half-bridge}(t) = v_X(t) \times i_X(t) \tag{2.4}$$

Chapter 3

Design

THIS chapter will present a detailed description of each component in this inverter. The associated loss mechanisms will also be described. Component sizing and tradeoffs will be considered briefly along with any practical issues, which may not be obvious from theory. Finally, an introduction to controlling power delivery will be presented. As will be seen in chapter 4, analyzing the circuit at the switching (fundamental) frequency (without considering the harmonic content) is sufficient for sizing considerations, and this will be used several times throughout this chapter. The notation of Fig. 3.1 will be used throughout this chapter. As the terminology is interlinked, discussing each component in detail may require definitions which originate in the other blocks' descriptions. Lastly, whenever parasitic components are discussed, the corresponding schematics will omit the parasitics which were found not to have an important effect on the design of the circuit.

3.1 Component description

The resonant inverter consists of the full-bridge inverter, the transformer, and the resonant tank. Although they make up a single functional block, the operation of these three components can be described independently.

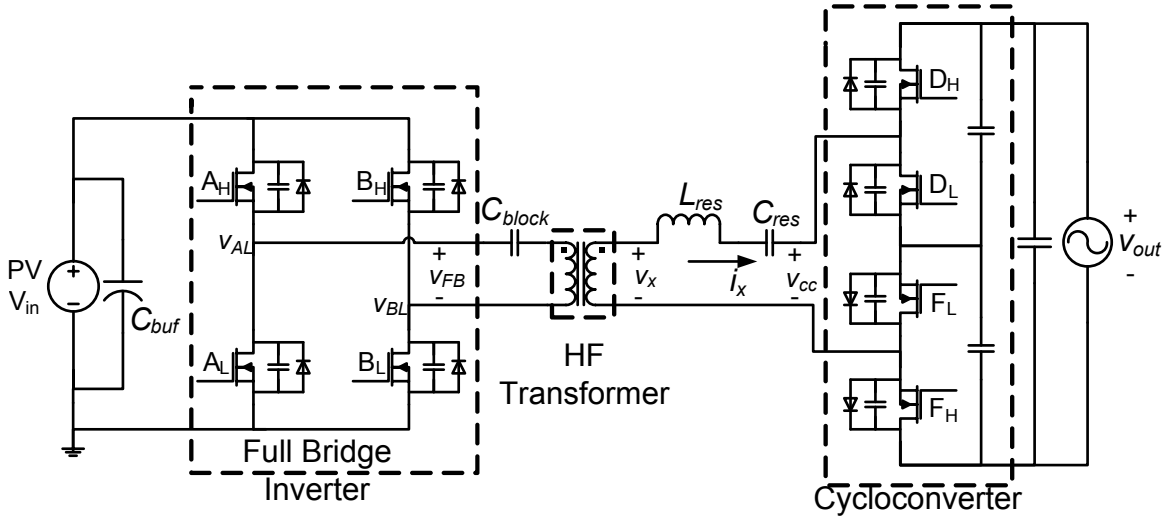


Figure 3.1: Circuit topology shown again for convenience.

3.1.1 Full-bridge

The full-bridge inverter consists of two half-bridge 'legs'. For convenience, the leg containing FETs A_L and A_H in Fig. 3.1 will be referenced as the leading leg and the other leg will be the lagging leg. The full-bridge output voltage v_{FB} is the voltage at the drain of the low side of the leading leg measured with respect to the drain of the low side of the lagging leg ($v_{FB} = v_{AL} - v_{BL}$). Each leg of the inverter is operated at just below 50% duty ratio (accounting for deadtime) under ZVS, as described in chapter 2. Figure 3.2 shows a typical timing diagram for the full-bridge, over several switching periods of length T_{SW}. The two legs are phase shifted with respect to each other, providing one way to control power transfer in this circuit.

When the high side of the leading leg is on and high side of the lagging leg is off, the full-bridge voltage is V_{IN}. When the lagging high side turns on, the differential voltage between the center nodes of the two legs drops to zero. When the leading leg switches next, the output voltage will drop to $-V_{IN}$, and so on for the rest of the switching cycle. The result is that v_{FB} is a three level stepped wave, of levels equal to

0, V_{IN}, and -V_{IN} volts, although at extreme cases of phase shift the stepped wave can become a square wave (-V_{IN} to V_{IN}) or a constant zero volts waveform. Describing the phase shift of the two legs can be done in several equivalent ways. A useful one involves defining a 'pulse width' variable δ as being the ratio of the positive voltage level (V_{IN}) of the full-bridge output voltage to half of the switching period. Thus, as δ ranges from 0 to 1, the full-bridge output ranges from a constant zero to a square

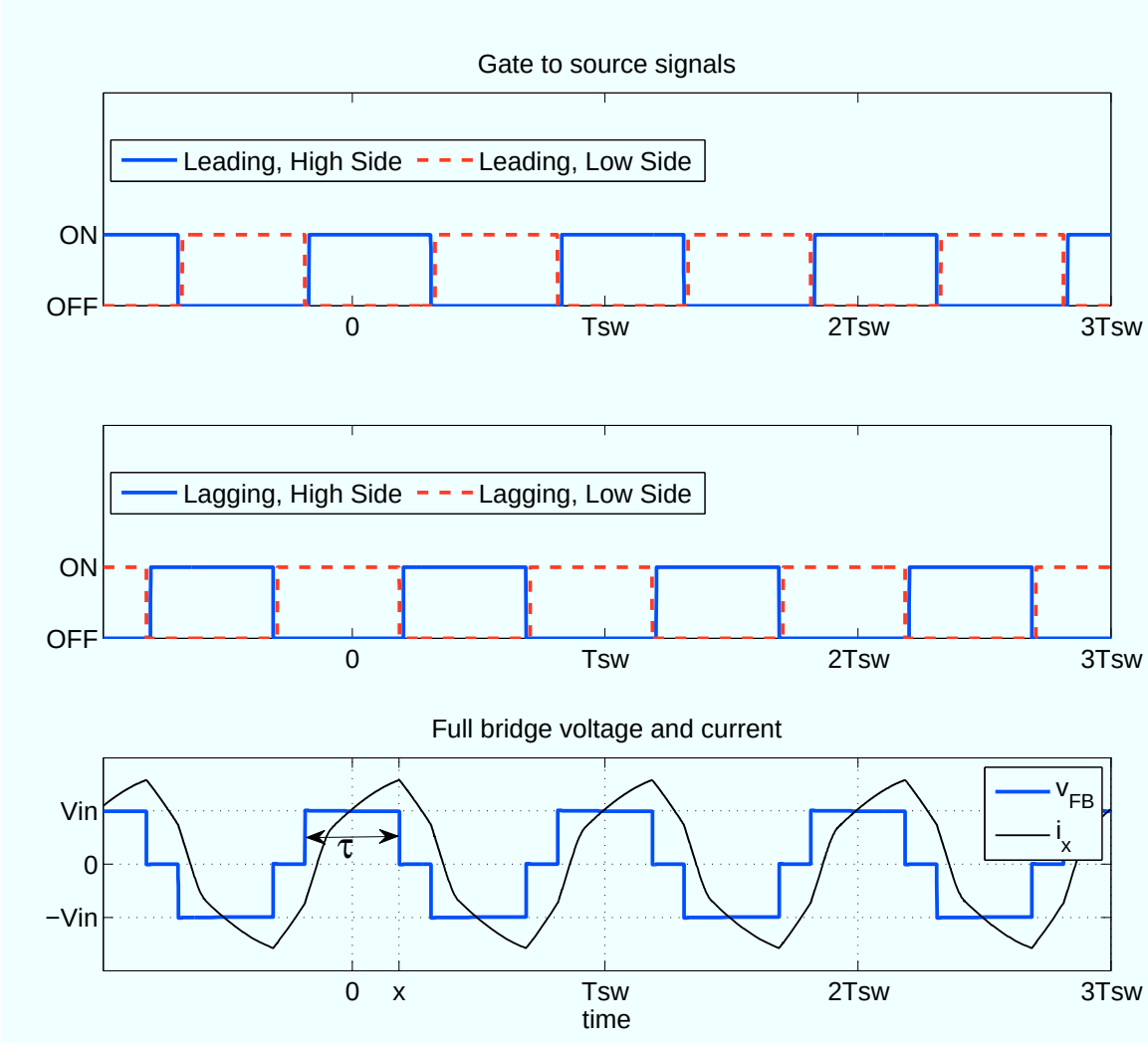


Figure 3.2: Full-bridge timing and output voltage v_{FB} of Fig. 3.1. The top two subfigures show the gating signals of the leading and lagging legs. Drain to source voltages (which can be inferred from the gating signals with the help of Fig. 2.7) combine to produce the waveform shown on the bottom subfigure.

wave. The time labeled by 'x' and the time interval τ on Fig. 3.2 are related to δ according to (3.1).

$$\delta = \frac{t(v_{FB} > \frac{V_{IN}}{2})|_{[t_0, T+t_0]}}{T/2} = \frac{2\tau}{T} = \frac{4 \times x}{T} \tag{3.1}$$

The harmonic content of waveform v_{FB} in Fig. 3.2 is given by (3.2). In that equation, 'n' is the harmonic number, with 0 being the DC component and 1 being the fundamental frequency. The frequency of the n$^{\text{th}}$ harmonic is $f_n = nf_{SW}$. Half-wave symmetry dictates that the magnitude of even harmonics is zero. Additionally, due to even symmetry as shown on the diagram, the phase of the odd harmonics is either 0 or π radians.

$$\langle v_{FB} \rangle_n = \begin{cases} 0, & n = 0,2,4,\ldots \\ \frac{4V_{IN}}{n\pi} \sin n\delta\pi/2, & n = 1,3,5,\ldots \end{cases} \tag{3.2}$$

Figure 3.2 also shows the resonant current in the circuit. The necessary condition for ZVS on the full bridge is for the zero-to-positive rising edge of v_{FB} to occur before the negative-to-positive zero crossing of the resonant current. Because of the periodicity and symmetry of the waveforms, this condition guarantees zero voltage transitions for all 4 MOSFETs of the full bridge.

One important difference between the leading and the lagging legs is that the leading leg transitions typically occur near the zero crossing of the current while the lagging leg typically switches close to the peaks of the current. While ZVS is maintained, as discussed in 2.4, certain situations may arise whereby the resonant current charges the drain to source capacitance in a shorter time than it takes for the gate driver to completely turn off the MOSFET (e.g. for v_{gs} to fall below the threshold voltage).

This is a function of the gate driver speed, the resonant current at that time, and the value of C_{oss} for the device. This condition leads to overlap loss, and there are at least four solutions to it:

- Change timing in the circuit such that the lagging leg does not switch under as high of a current. This is not always consistent with overall efficiency optimization. As will be seen later, the overlap loss is undesirable but not to the extent that it would offset altering the otherwise optimal operation of the implemented circuit.
- Obtain a faster gate driver. This is not always an option in terms of what is procurable on the market.
- Use MOSFETs with a higher output capacitance or a better tradeoff between input and output capacitance. This is not always desired as higher output capacitance typically means higher input capacitance, which increases gating loss. Obtainability may also be an issue.
- Add discrete low loss capacitors across drain to source of the lagging leg devices. This will serve to delay the rising edges in the lagging leg half-bridge, allowing more time to turn off completely under ZVS. This is the option that was utilized in the actual half-bridge, and the details will be shown in chapter 5.

3.1.1.1 Full-bridge losses

There are four main loss mechanisms in the full-bridge inverter, of which two are dominant. The first major loss comes from the MOSFET conduction and is approximated by (3.3). It is an approximation because while the equation assumes that at any time one FET in each leg is conducting, this assumption is wrong by the total deadtime in each leg. Chapter 4 will show that this is a good approximation. The

factor of 4 comes from the 4 devices in the full-bridge. The secondary side resonant current i_X needs to be scaled by the turns ratio of the transformer (see (3.6)) because the full-bridge is on the primary side. The second major loss component comes from gating each MOSFET - toggling the state between off and on requires the gate driver to charge the input capacitance of each device once per switching cycle. The total charge of $Q_{g,FB}$ (found in data sheets under gate charge) has to be delivered while charging the gate to V_{gs} with respect to the source (typically 5-12 volts). The loss associated with this is given by (3.4). The third loss mechanism is the aforementioned overlap loss, primarily in the lagging leg. This loss is hard to estimate but realistically it is not necessary to do so as care should be take to avoid the overlap condition altogether. The fourth loss mechanism occurs due the on-state drop of the body diode, as discussed in 2.4. This loss is given by (3.5), with V_{diode} being the voltage drop of the body diode given in the data sheet. However, the time that the diode conducts ($t_{diodeON} - t_{diodeOFF}$) is extremely small and this loss is effectively non-existent in proper operation.

$$P_{FB,cond} = 4 \times (Ni_{x,RMS})^2/2R_{ds,ON,FB} = 2 \times (Ni_{x,RMS})^2 R_{ds,ON,FB} \tag{3.3}$$

$$P_{FB,gating} = 4 \times Q_{g,FB} V_{gs,FB} f_{SW} \tag{3.4}$$

$$P_{FB,diode} = 4 \times \int_{t_{diodeON}}^{t_{diodeOFF}} V_{diode,FB} N i_x(\tau) d\tau \tag{3.5}$$

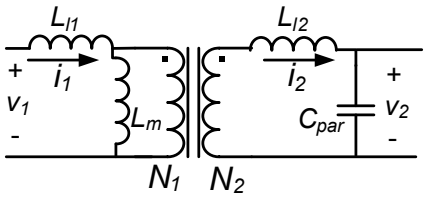


Figure 3.3: Transformer with the important parasitic elements.

3.1.1.2 Component selection

The 4 MOSFET devices need to be able block the maximum input voltage and be able to carry the maximum current of $N \times i_{x,max}$. To reduce losses, they need to have a low value of $R_{ds,ON,FB}$ and a low value of $Q_{g,FB}$ which tend to be contradictory requirements for real devices. Chapter 4 will provide a description of how to choose the right balance.

3.1.2 Transformer

The transformer is placed in this topology for two reasons: voltage transformation and galvanic isolation. Figure 3.3 shows the transformer with its major parasitics. It has N_1 turns of wire on the primary side and N_2 on the secondary. Assuming for now that leakage inductances L_{l1} and L_{l2} are very small and the magnetizing inductance L_m is very large, the relationship for voltage and current between the two sides are given by (3.7a), (3.7b).

$$\text{Turns Ratio} : N = \frac{N_2}{N_1} \tag{3.6}$$

$$i_1 = \frac{N_2}{N_1} \times i_2 = N \times i_2 \tag{3.7a}$$

$$v_1 = \frac{N_1}{N_2} \times v_2 = \frac{v_2}{N} \tag{3.7b}$$

Disregarding the parasitic elements is a good approximation for power transfer analysis (as will be seen in chapter 5). Therefore v_{FB} may simply be scaled by N and the transformer may be omitted for that purpose. The minimum turns ratio N is found by looking at the fundamental components of voltage waveforms v_{FB} and v_{CC}. The minimum turns ratio results from the need to amplify the minimum input voltage with a minimum full bridge phase shift, to the maximum fundamental of v_{CC}. This is given by equation (3.8). Finally, the leakage inductances L_{l1} and L_{l2} are effectively absorbed into the resonant inductance, according to (3.9).

$$N_{min} = \frac{\langle v_{CC} \rangle_{1,v_{out}max}}{\langle v_{FB} \rangle_{1,V_{in}min}} = \frac{\frac{2}{\pi}240\sqrt{2}}{\frac{4}{\pi}25} \approx 6.8 \tag{3.8}$$

$$L_{res,eff} = L_{res} + N^2 \times L_{l1} + L_{l2} \tag{3.9}$$

3.1.2.1 Transformer losses

The transformer design calls for a core volume on the order of a Ferroxcube RM14 pot core. This size is selected for compactness, and switching in the aforementioned frequency range allows for this choice. The major losses in the transformer arise due to current conduction, core loss, and parasitic capacitance C_{par}. Tables 3.1 and 3.2 list the properties of the transformer windings and the pot core, respectively, which are needed for loss estimation when Litz wire is used. The method of [17] is used for the conduction loss prediction. In overview, the loss due to conduction of alternating current is a scaled value of the resistive loss of the same wiring configuration carrying a

Symbol	Parameter
N_t	Number of wire turns
d	Diameter of one strand of Litz wire [m]
ρ	Resistivity of wiring material [Ωm]
N_s	Number of (Litz) wire strands in a winding turn
N_l	Number of winding layers. May be fractional
$XX_{\_,1}$ $XX_{\_,2}$	Subscripts used to refer to primary and secondary side windings respectively
K_l	Winding window utilization factor Relates to the bobbin geometry
K_p	Litz wire packing factor

Table 3.1: Important winding parameters

Symbol	Parameter
A_e	Magnetic core cross sectional area [m^2]
V_e	Magnetic core volume [m^3]
A_L	Inductance per turn [H]
MLT	Bobbin mean length per turn [m]

Table 3.2: Important core parameters

direct current. The scaling coefficient depends on the frequency, the material, and the number and arrangement of wires, and physically arises from the skin and proximity effects of high frequency current being carried in a closely spaced arrangement of wires [17]. Equations (3.10) show the steps taken in estimating this loss.

$$\text{Skindepth of copper(Cu)} : \delta_{cu}(f) = \sqrt{\frac{\rho_{cu}}{\pi \mu_{cu} f}} \tag{3.10a}$$

$$\text{Parameter x} : x(f) = \frac{d}{2\delta}\sqrt{\pi K_l} \tag{3.10b}$$

$$\text{DC to AC resistance scaling coefficient} : F_R(f) = 1 + \frac{5N_l^2 N_s - 1}{45} x^4(f) \tag{3.10c}$$

$$\text{DC resistance of n}^{th}\text{ side winding } : R_{DC,n} = \text{MLT}\frac{2\rho N_{t,n}}{\pi d_n^2 N_{s,n}} \tag{3.10d}$$

$$\text{Total winding loss } : P_{X,cond} = \left(N i_{x,RMS}\right)^2 R_{DC,1} F_{R,1}(f) + i_{x,RMS}^2 R_{DC,2} F_{R,2}(f) \tag{3.10e}$$

The presence of a finite magnetizing inductance L_m gives rise to flux density in the core and core loss [18]. The process of estimating this loss is shown in eq. 3.11. In overview, the magnetizing inductance is calculated based on the winding configuration. This, along with the fundamental of the full bridge voltage, allows for estimation of the magnetizing current, which yields to a finite magnetic flux density in the transformer core. Given the peak flux density (assuming a sinusoidal current), the core loss is calculated using the Steinmetz parameters [19].

$$\text{Magnetizing inductance} : L_m \approx A_L N_{t,1}^2 \tag{3.11a}$$

$$\text{Magnetizing current amplitude} : \langle i_m \rangle_1 \approx \frac{\langle v_{FB} \rangle_1}{\omega_{SW} L_m} \tag{3.11b}$$

$$\text{Magnetic flux peak} : B_{X,pk} \approx \frac{A_L N_{t,1} \langle i_m \rangle_1}{A_e} \tag{3.11c}$$

$$\text{Total core loss } : P_{X,core} = C_m C_t f^{x(f)} B_{X,pk}^{y(f)} V_e \tag{3.11d}$$

The parameters C_m, C_t, $x(f)$, $y(f)$ are typically reported by the magnetic material manufacturer, as seen in [19]. For a properly designed transformer, the magnetizing inductance is typically large enough that core loss is very small. The peak magnetic flux density given by equation (3.11c) has to be less than some B_{sat}, which is a material property. Exceeding this value will void the terminal relationships given by equations (3.7). The frequency dependence of magnetizing current given by equation (3.11b) sets the core saturation limit on how low the switching frequency can be.

The last major loss in the transformer results from the parasitic capacitance across the secondary. The energy that is used to charge this capacitance is lost during every switching cycle. Equation (3.12) gives an estimate for this loss. In general, C_{par} can be made very small for an ordinary design. However, its presence may give rise to very high frequency ringing in the circuit waveforms, which may contribute to the losses in other components. Chapter 5 will discuss a way to deal with this.

$$P_{X,C_{par}} = C_{par} \left(2N \left\langle v_{FB} \right\rangle_1\right)^2 f_{SW} \tag{3.12}$$

The total loss in the transformer is approximated by the sum of (3.10e), (3.11d), and (3.12). For a real design, the total loss is typically dominated by winding losses. It is also desireable that the magnetizing current is minimized because as it takes on larger values, the total primary side current increases for a given secondary side resonant current (as in a current divider). This contributes to losses in the full-bridge. The

selection of turns ratio and the actual numerical parameters of the real transformer will be discussed in chapters 4 and 5.

3.1.3 Resonant tank

The inductor L_{res} and capacitor C_{res} placed in series result in a second order filter. The resulting complex impedance seen by the full-bridge inverter is given by (3.13).

$$Z = \frac{1}{j\omega C_{res}} + j\omega L_{res} + R \tag{3.13}$$

Ignoring the resonant inverter load (which consists of the cycloconverter and the voltage output) for now, $R = R_{par}$ is the parasitic resistance of the full bridge and the resonant tank. This involves the on-state resistances of the FETs and the equivalent series resistances of the passive components. The unloaded tank quality factor Q is defined by (3.14). This is the ratio of the peak AC energy stored to energy dissipated per one period, scaled by 2π. The resonant angular frequency ω_0 is defined by (3.15). This is the frequency at which the impedance of the tank is at its minimal value of a purely resistive value R_{par}. This frequency is also important because operating under ZVS as shown on Fig. 3.2 requires that switching frequency always be above this value.

$$Q_{unloaded} = \frac{1}{R_{par}}\sqrt{\frac{L_{res}}{C_{res}}} \tag{3.14}$$

$$\omega_0 = 2\pi f_0 = \frac{1}{\sqrt{L_{res} \times C_{res}}} \tag{3.15}$$

As was mentioned earlier, the entire resonant tank may be placed on either the primary or the secondary of the transformer as long as the inductance and capacitance are scaled by the square of the turns ratio of the transformer. The proper scaling has to be consistent with equations (3.16), which are valid for scaling any reactive component across the transformer (e.g. the blocking capacitor).

$$L_{secondary} = L_{primary} \times \left(\frac{N_2}{N_1}\right)^2 \tag{3.16a}$$

$$C_{secondary} = C_{primary} \times \left(\frac{N_1}{N_2}\right)^2 \tag{3.16b}$$

While the inductor is generally able to handle the voltages seen in the inverter, discrete capacitors have a voltage rating. Care must be taken to ensure that the voltage across the capacitor never exceeds its rated value. Equation (3.17) provides the minimal voltage rating based on maximum resonant current that can be present in the inverter. The sizing of the resonant tank (picking the inductance and the capacitances) will be discussed in chapter 4 as part of optimization of the inverter.

$$V_{rating} > \left|\frac{i_{x,MAX}}{\omega_{SW} C_{res}}\right| \tag{3.17}$$

3.1.3.1 Resonant tank losses

The inductor loss analysis is similar to that of the transformer. The main difference is that the inductor has only one winding 'side', which slightly simplifies the analysis (all of the secondary side terms can be dropped in (3.11) and (3.10)). The inductor core size can be the same as for the transformer, although with a different A_L value. The important core and winding parameters are still given by tables 3.1 and 3.2 but now

they apply to the inductor (the actual numerical values will be different). The steps for calculating the core loss are similar to the transformer case. The difference is that the magnetizing current becomes the full resonant current in the equation for peak flux. The final expression for core loss is given by (3.18). Using the intermediate definitions of (3.10), the inductor winding conduction losses are given by 3.19. If the inductor is placed on the primary side of the transformer, the resonant current becomes scaled by N, as the corresponding change needs to be made in these loss estimates. In the inductor, the core is usually gapped to provide energy storage in the gap. As a result, core to conduction loss ratio is different from that of the transformer. Being a magnetic material, the inductor core is also susceptible to saturation. Although the current through the inductor is much higher than the magnetizing current of the transformer, its A_L value is typically much smaller. Care must be taken to ensure that the inductor design never allows the magnetic core to saturate, given the highest resonant current seen in the inverter.

The only loss in the resonant capacitor comes due to its ESR. This will be discussed in section 3.1.6.4

$$\text{Magnetic flux peak} : B_{L,pk} \approx \frac{A_L N_t i_{x,pk}}{A_e} \tag{3.18a}$$

$$\text{Total core loss} : P_{L,core} = C_m C_t f^{x(f)} B_{L,pk}^{y(f)} V_e \tag{3.18b}$$

$$\text{Total winding loss} : P_{L,cond} = i_{x,RMS}^2 R_{DC} F_R(f) \tag{3.19}$$

3.1.4 Cycloconverter

A half-wave (as opposed to a full-bridge structure) cycloconverter operates under ZVS to down-convert the high-frequency AC current to line frequency. Operation of the cycloconverter can be summarized as follows: when the line voltage is positive (referenced to the indicated polarity), the bottom two switches of the cycloconverter (F_H and F_L in Fig. 3.1) remain on, while the top two switches (D_H and D_L) form a ZVS half-bridge that modulates the average current (over a switching cycle) that is delivered into the AC line. Likewise, for negative output (line) voltages, the top two devices remain on while the bottom two devices modulate the power delivery. Each half-bridge operates at a nearly 50% duty ratio (accounting for deadtime). Note that polarity of the output voltage solely determines which devices need to be turned on because those devices would offer no blocking capability under that polarity (due to the body diodes). Operating in this manner and switching at the same frequency as the full-bridge inverter, the cycloconverter can down-convert to any frequency which is smaller than the switching frequency. Essentially, it synchronizes to the frequency of the load. Finally, the compactness of the layout of the high-frequency AC paths in this cycloconverter gives it substantial practical advantage over 'back-to-back switch' cycloconverter topologies.

Figure 3.4 shows the resonant current and the idealized voltage waveform v_{CC} for both a positive and a negative output voltage. The high side of each respective half-bridge can turn on under zero voltage when v_{CC} levels out to v_{out}, as indicated by the controlling signals on the figure. Angle ϕ is defined as the difference between the angle along the switching cycle at which $v_{CC} = \frac{1}{2}v_{out}$ and the angle at which the resonant current crosses zero while rising, referenced to the latter (on Fig. 3.4 $\frac{\phi T_{SW}}{2\pi} = \alpha - \beta$). The resulting power transfer to the line is visualized in Fig. 3.5 for this case. Each continuous shaded region represents the total energy delivered to the

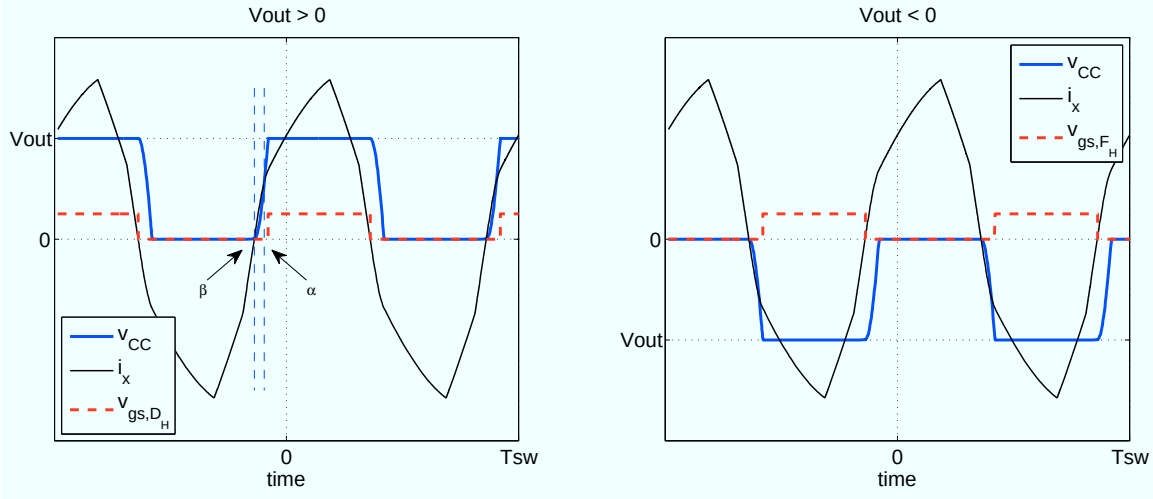


Figure 3.4: Cycloconverter operation for different output voltage polarity. As previously mentioned, v_{out} is essentially constant over a switching cycle.

line per switching cycle and the instantaneous output power is the average of that area over a switching cycle.

Although a more detailed analysis will be considered in chapter 4, looking at the power transfer through the fundamental component of resonant current will provide

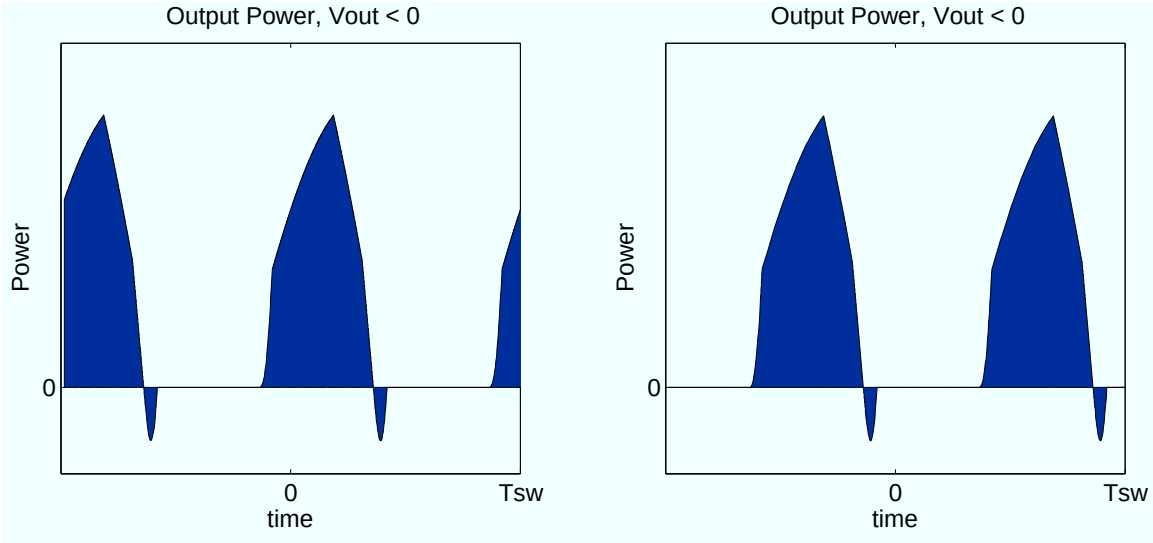


Figure 3.5: Power transfer to the line corresponding to the timing of Fig. 3.4. Positive net power is delivered.

a valid qualitative description (and a reasonable quantitative approximation). Equation (3.20) shows the relationship between power delivery per switching cycle, the line voltage, the amplitude of the fundamental of the resonant current, and angle ϕ. The average value of the shaded area of Fig. 3.5 is the graphical interpretation of this equation. When ϕ is 0, the shaded area is entirely non-negative, as the total duration of the negative area region is $\frac{\phi T_{SW}}{2\pi}$. As ϕ increases from 0, the negative portion of the shaded area increases as well. When ϕ is equal to $\pi/2$ the positive area is equal to the negative area and the net power delivery is 0. Increasing ϕ any further makes the net power delivered to the line negative. Finally, when ϕ is equal to π power is only drawn from the line in a switching cycle, with no positive component present in the shaded area. The necessary condition for ZVS on the cycloconverter is that ϕ be greater than 0. The upper limit for the value of ϕ is somewhere below π if ZVS is to be kept.

$$P_{out} = \frac{1}{2\pi}\int_0^{2\pi} i_x(\tau) v_{CC} d\tau \approx \frac{1}{\pi}\left\langle i_x \right\rangle_1 v_{out}\cos\phi \tag{3.20}$$

Equation (3.20) provides a means for defining an equivalent load impedance, based on fundamental harmonic analysis. The following expressions were derived in the original work on this inverter in [14] and will be restated here without a detailed derivation. In overview, when $\phi = 0$, the cycloconverter acts as an ideal rectifier with v_{CC} being completely in phase with i_x. For a given power delivery, the equivalent load resistance is defined by equation (3.21a). As ϕ increases from 0, the ideal rectifier behavior changes such that the cycloconverter and the line can be modeled as a series resistor-capacitor combination. The equivalent capacitance (C_x) and resistance (R_x) are given by equations (3.21b) and (3.21d) for a given output power and angle ϕ. This allows us to define the loaded tank quality factor, in which the resistive element is replaced with the load resistance and the resonant capacitance changes by being in

series with C_x. This is given by equation (3.22). Since a large quality factor means that there is significant energy 'sloshing' in the reactive components, it is desireable to keep the loaded tank quality factor to a minimum for high efficiency. Typically the loaded tank quality factor will be less than unity for this system (since the power delivery requirement spans a wide range of values, the equivalent load resistance will change with the operating condition and thus a single quality factor cannot be defined).

$$R_{ld} = \frac{2v_{out}^2}{\pi^2 P_{out}} \tag{3.21a}$$

$$R_x = R_{ld} \cos\phi^2 \tag{3.21b}$$

$$X_x = -R_{ld} \cos\phi \sin\phi \tag{3.21c}$$

$$C_x = -\frac{1}{\omega X_x} \tag{3.21d}$$

$$Q_{loaded} = \frac{1}{R_x}\sqrt{\frac{L_{res}}{\frac{1}{1/C_{res}+1/C_x}}} \tag{3.22}$$

3.1.4.1 Cycloconverter losses

Similar to the full bridge, the losses in the cycloconverter arise from resonant current conduction and from gating the MOSFETs. Given a low-loss bypass capacitor across each pair of half-bridges in the cycloconverter, the two switches which are on in a given half cycle are effectively in parallel, and so contribute reduced conduction loss compared to a single switch. The total conduction loss is given by equation (3.23). At any given time cycloconverter conduction loss owing to 1.5 device on-state resistances is obtained, as compared to two on-state resistances for a conventional 'back-to-back

switch' half-wave cycloconverter. Only one of the two half-bridges modulates at any given time and the gating loss associated with that is given by equation (3.24). Overlap loss is generally not a problem with the cycloconverter, owing to the small current present on the secondary side of the transformer. The final loss mechanism is the body diode conduction (given by equation (3.25)), but as is the case with the full-bridge, instead of estimating this loss, proper timing should be used to bring it down to negligible levels.

$$P_{CC,cond} = 1.5 \times i_{x,RMS}^2 R_{ds,ON,CC} \tag{3.23}$$

$$P_{CC,gating} = 2 \times Q_{g,CC} V_{gs,CC} f_{SW} \tag{3.24}$$

$$P_{CC,diode} = 2 \times \int_{t_{diodeON}}^{t_{diodeOFF}} V_{diode,CC} i_x(\tau) d\tau \tag{3.25}$$

3.1.5 Component Selection

The four MOSFET devices need to be able block the maximum output voltage and be able to carry a current of $i_{x,max}$. To reduce losses, they need to have a low value of $R_{ds,ON,CC}$ and a low value of $Q_{g,CC}$, which tend to be contradictory requirements for real devices. Additionally, output capacitance needs to be just big enough to allow for ZVS, yet keep ϕ to a minimum. This requirement is more important here than for the full-bridge because the current amplitude is much lower. Chapter 5 will provide a description of how to choose the right balance.

3.1.6 Capacitor banks

3.1.6.1 Input

The instantaneous (averaged over a switching cycle) output power varies at $2f_{line}$ or 120 Hz (see Fig. 2.1). However, the PV array provides a nearly constant power of P_{ave} over one line cycle. This instantaneous energy difference has to be made up in the bulk capacitance across the PV array (C_{buf} in Fig. 3.1). Defining the allowed capacitor voltage ripple ratio by equation (3.26), the size of this capacitance can be calculated according to (3.27) for a nominal input voltage and average power output. Larger capacitance is needed for smaller allowed ripple. Although MPPT design will not be considered, the allowed ripple ratio would be controlled by it. Owing to the large values of needed capacitance, these components would generally be electrolytics.

$$k = \frac{V_{IN} - v_{ripple}}{V_{IN} + v_{ripple}} \tag{3.26}$$

$$C_{buf} \approx \frac{P_{ave}}{2\omega_{line} V_{IN} (1 - k)} \tag{3.27}$$

3.1.6.2 Blocking

The blocking capacitor ensures that no DC current is present in the circuit, as this would saturate the core of a non-ideal transformer. Theoretically, since the full-bridge voltage waveform ideally has no DC component this capacitor is needed only on the secondary side of the transformer (and thus only if the resonant tank is placed on the primary). However, even a slight asymmetry in the full bridge waveform would

create a DC component, and practically the blocking cap should be present regardless of where the resonant tank is placed. The value of the blocking capacitor should be much larger (factor or 7-10) than that of the transformer scaled equivalent of the resonant capacitor, so as to have minimal influence on the quality of the resonant tank. These components can be ceramic or film capacitors for loss considerations (and the fact that the capacitance does not have to be large, as will be seen in chapter 4).

3.1.6.3 Output

The output capacitor bank, placed across the load, filters the switching frequency ripple out of the output current. This is needed to maintain acceptable power quality at the output. This capacitance had to be chosen such that the impedance at switching frequency is very low and impedance at the line frequency is very high (and given the large difference between the two frequencies, only the former needs to be satisfied). The appropriate condition is given in equation (3.28). This capacitance is generally of a small enough value to be satisfied by ceramic or film components.

$$\frac{1}{\omega_{SW} C_{out}} < 1 \tag{3.28}$$

3.1.6.4 Capacitor losses

Each capacitor has an equivalent series resistance (ESR) rating, which is the only contribution to the loss in the circuit. For high efficiency, capacitors with low ESR should be chosen whenever possible (e.g. NP0 type for ceramic, small value for loss

tangent for electrolytic). Placing several capacitors in parallel reduces the net ESR of the bank. The associated loss is a simple resistive drop, given by (3.29).

$$P_{cap} = R_{ESR} i_{cap,RMS}^2 \tag{3.29}$$

Chapter 4

Control and Optimization

THE task of optimizing the inverter is inherently related to the control law governing its operation. This chapter will present the different ways to control the inverter and demonstrate the general control algorithm which optimizes efficiency. From there on, a detailed procedure for obtaining control inputs will be described along with its limitation and the models which were used to derive it. As this procedure will be deemed too computationally intensive to be practical in the optimization process, a simplified inverter model will be used to find the optimal circuit parameters.

4.1 Inverter control inputs

As mentioned in chapter 3, the output voltage is essentially constant over a switching cycle and this was illustrated on every relevant figure presented so far. Although the ultimate task on this inverter is to output to an AC voltage, the low grid frequency allows for approximating the entire AC line cycle operation as a series of DC-DC operating conditions, with the instantaneous output voltage being $V_{out} = v_{out}(t)$. The term 'operating point' will hereafter refer to a triplet $(V_{IN}, V_{out}, P_{out})$ as valid average values over a switching cycle. These values uniquely specify the instantaneous input/output properties of the inverter, treated as a 'black box'. The task of con-

trolling DC/AC conversion is significantly simplified if the DC/DC approximation is used. Yet, not much is lost as theoretically it is easy to effectively output low frequency AC by stepping through DC points with a fine step size and at a high frequency (in fact this is exactly what a microcontroller could do in such an application). Therefore, the rest of this work will be concerned with operating the inverter in the DC/DC mode with the output voltage picked from the allowed (needed) range.

Summarizing the main points regarding operation of the inverter and the ZVS-imposed constraints, the main waveforms of interest are shown on Fig. 4.1 along with the notation which will be used through out this chapter. For simplicity, the waveforms have been scaled to fit and are missing their ordinate values as their relative timing is of greater importance. The figure illustrates waveforms on the time scale of the switching period. The voltage waveform labeled by v_x represents a scaled waveform v_{FB}. Note that a constant positive scaling factor does not change the timing constraints and angle definitions, and the reason for this scaling will become clear later. Angle θ is the angle (normalized to the switching period) by which the zero-crossing of resonant current i_x lags the rise of the voltage of the full bridge. Angle ϕ is the angle by which the zero-crossing of current leads the point at which $v_{CC} = 0.5V_{out}$. For ZVS, θ and ϕ are positive at minimum. Ratio δ (and its relation to τ) remains as defined in chapter 3. The quasi-sinusoidal shape of the resonant current emphasizes the preference to operate at a low loaded tank quality factor.

There are four main ways to control the amount of power transferred from the input voltage source V_{IN} to the output voltage source V_{out}. They are as follows:

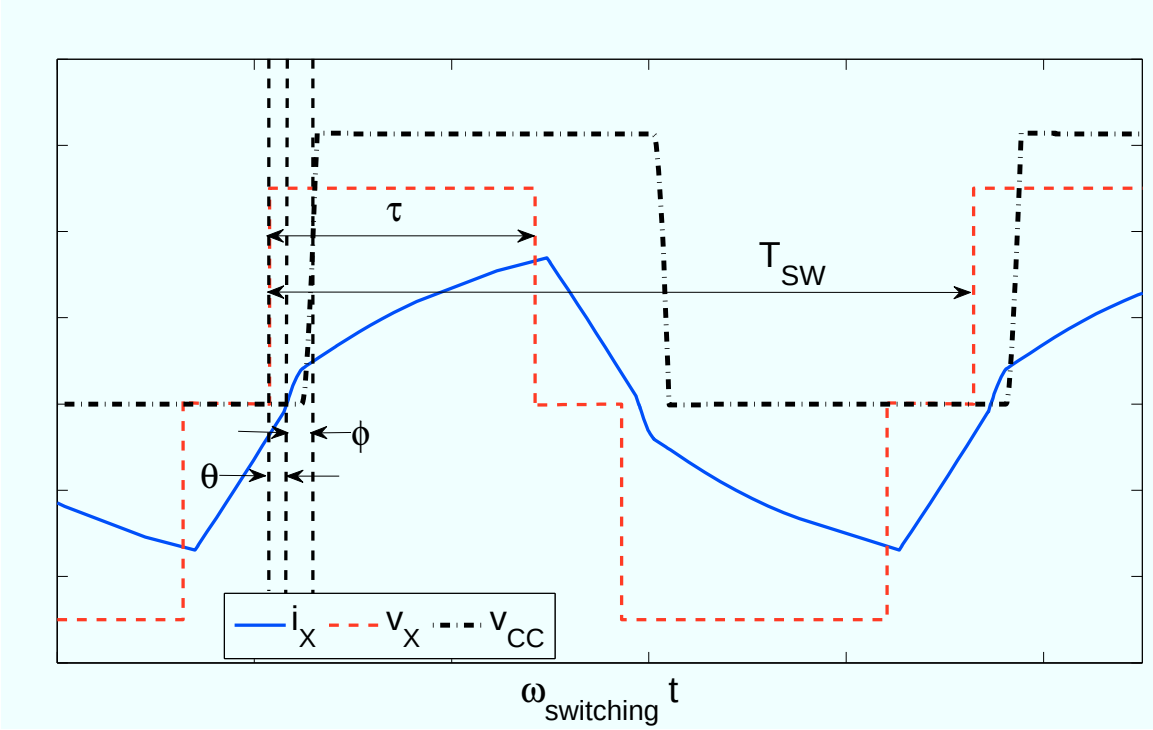


Figure 4.1: Waveforms and angles of interest.

4.1.1 Full bridge phase shift

Shifting the phase of the two legs of the full-bridge changes the overall 'width' of the positive and negative voltage pulses of v_{FB}. As phase shift decreases from $pi/2$ to 0 (which corresponds to the ratio δ changing from 1 to 0), the fundamental component of v_{FB} decreases, as shown by equation (3.2). In the limiting case of zero phase shift, the full-bridge outputs a constant voltage of 0 and no power is delivered from the input.

4.1.2 Switching frequency

The impedance presented by the series resonant LC network is a function of frequency, as indicated by equation (3.13). Starting just above resonance (as a minimal ZVS

requirement), the magnitude of the impedance increases as the frequency increases. It was shown in chapter 3 that the cycloconverter and the line output can be modeled as some equivalent impedance, which may be incorporated into the overall impedance of the resonant tank. As the tank impedance increases with frequency, the resonant current magnitude decreases. Thus, fixing everything else constant, the power transfer to the load goes down with increased frequency.

4.1.3 Cycloconverter phase shift

As was shown in equation (3.20), for a given resonant current amplitude and output voltage, increasing the cycloconverter phase shift ϕ decreases the instantaneous output power. An alternative, although slightly different, way to look at this is to realize that for a given output power, operating at larger phase angle ϕ means having larger peak (and RMS) resonant currents.

4.1.4 Burst mode

Burst mode control involves turning the converter on and off at a frequency which is much higher than line frequency, yet lower than the switching frequency. Since operating conditions with high output powers tend to be more efficient than the low power ones, burst mode can improve the overall averaged efficiency by 'bursting' high power at a high efficiency over a shorter time period, while delivering the same amount of energy as a continuous operation at a lower power and efficiency. Although this control method will not be considered in any more detail in this thesis, it remains a valid control method that can be easily incorporated later.

4.2 Optimal control strategy

The sum of all loss estimates presented in chapter 3 can be lumped into one equation (see (4.1)), which will hereafter be referenced as 'the loss function'. The loss function has some direct dependence on the switching frequency and the resonant current RMS value. The goal of an efficient control law is to minimize the loss function while satisfying the power delivery requirement given by equation (3.20) and the appropriate ZVS constraints (otherwise the loss function is invalid).

$$Loss = A\left(f_{SW}, i_x\right) f^{\alpha} + B\left(f_{SW}, i_x\right) {i_x}^{\beta} \tag{4.1}$$

The maximum amount of power which the inverter can deliver (ignoring ZVS requirements) occurs when it is operated at $f_{SW} = f_{res}$, $\delta = 1$, and $\phi = 0$. For ZVS, there exists a restriction on angles θ and ϕ - each has to be greater than some $\theta_{critical}$ and $\phi_{critical}$ respectively (the details of this will be discussed in the next section). Decreasing power delivery can be done by increasing f_{SW} or ϕ, or by decreasing δ. Starting with the output voltage and power requirements, the minimum required resonant current is fixed (e.g. based on equation (3.20) in the unrealizable case of $\phi = 0$). A necessary increase in current amplitude comes from the need to have at least a minimal phase shift ($\phi_{critical}$) on the cycloconverter. Accounting for this results in the true minimum achievable current which will satisfy power delivery.

Let it be assumed for now that the sensitivity of the loss function is the same for both f_{SW} and i_x. That is, the increase in loss is the same if f_{SW} or i_x increased by an identical percentage. Then, any extra cycloconverter phase shift will result in extra loss as a consequence of higher current. The task is then transformed into picking the appropriate switching frequency and full-bridge phase shift parameter δ, such that the required resonant current is produced in the circuit. An infinite number

of combinations of δ and f_{SW} can be found to satisfy these conditions. The loss function suggests that loss goes up with frequency but there is virtually no loss penalty associated with changing of parameter δ. Therefore, out of all possible combinations (δ,f_{SW}), the lowest loss occurs for the minimal f_{SW}. Imposing the $\theta_{critical}$ constraint, uniquely determines this pair. This mode of operation will hereafter be referred to as the minimal current mode.

In reality, the sensitivity of the loss function is generally not the same for increases in frequency as it is for increases in current. That means that at certain points it may be more efficient (the loss function result is smaller) to operate at a higher than minimal ϕ (and thus higher than minimal i_x) which allows the switching frequency to drop below the frequency corresponding to $\phi_{critical}$. Similarly, it may be desirable to put an upper limit on the switching frequency or, as previously discussed, increase ϕ to $\pi/2$ and beyond to provide zero power delivery (at the zero crossing of the line) or reactive power compensation, respectively. All of those actions will force this mode of operation, which will hereafter be referred to as the minimal frequency mode. In the minimal frequency mode, above minimal energy is 'sloshed' around in the resonant current as a consequence of operating at lower frequency, either for efficiency or other practical concerns. The loosely defined boundary between the two modes (stated in terms of $f_{boundary}$ and/or $\phi_{boundary}$) will be referred to as the sloshing boundary. The sloshing boundary depends entirely on the loss function, which in turn heavily depends on the component parameters and the validity of the loss models of chapter 3. It is not a precise boundary because the analytical loss models may not be accurate enough to exactly predict which direction is best and experimental verification will often be needed. The discussion presented in the rest of this chapter comes as a result of experimentally determining that most of the operating points of CEC interest occur in the minimal current mode of operation, with the minimal frequency mode mostly relevant for the very low output powers,

which have small CEC weighting coefficients. Additionally, the gain in efficiency due to sloshing is typically very small, so the minimal current mode is often very close to the most efficient control law.

In summary, efficient control of the converter is realized as a combination of the techniques presented in section 4.1, with power controlled by regulating a combination of switching frequency, full-bridge phase shift, and cycloconverter phase shift. The nominal control strategy which minimizes losses looks to operate at the lowest frequency and consequently the narrowest pulse width τ on the full bridge, while cycloconverter phase shift ϕ is kept to a minimum, as this results in the lowest RMS current. Essentially the combination of frequency/phase shift is more favorable in terms of loss than cycloconverter modulation, with further preference going to lower frequency. This is true for most operating points of interest, except for at extremely low output powers, for which current sloshing may provide a slightly more efficient mode of operation. The frequency has to be high enough such that θ and ϕ are greater than some $\theta_{critical}$ and $\phi_{critical}$ respectively, a requirement that comes from ZVS considerations. The values for these critical angles will be discussed in the next section, but the reason that they are non-zero is that a finite amount of time is needed for the resonant current to completely commutate through the parasitic capacitances of each FET near the switching transitions.

4.3 Equivalent circuit model

Considering only the minimal current mode for now, under the proposed algorithm, the cycloconverter is controlled to act as close to a diode rectifier as possible. The only deviation from the ideal diode behavior results from the output capacitance. Moreover, the entire circuit analysis may be significantly simplified for power trans-

fer modeling purposes. As mentioned in chapter 3, the action of a well designed transformer may be replaced by a voltage scaling coefficient equal to the turns ratio. Utilizing this assumption and modeling the cycloconverter as a ideal diode rectifier with a capacitor across one of the diodes, the model shown on Fig. 4.2 preserves all of the important characteristics of the inverter from a controller design point of view, while simplifying the analysis. The only new symbols in this model are v_x, which is the full-bridge voltage scaled by N (this is an AC waveform), and C_{par}, which is some equivalent capacitance across the cycloconverter. The waveforms of Fig. 4.1 are valid for this model. Note that C_{par} is placed across only one of the diodes because any $C_{par,D2}$ placed across diode D2 would be in parallel with C_{par} under AC analysis. Therefore the total effective capacitance may be lumped into one circuit element (as long as its value is the sum of whatever physical circuit parasitics which it represents). Since all MOSFETs are replaced with lossless components, the parasitic resistance term R_{par} should include on-state resistance of all FETs conducting at any time (2 in the full-bridge and 1.5 in the cycloconverter). Additionally R_{par} should absorb any ESR or other losses of the reactive components if they can be modeled via equivalent ESR. The output voltage source is DC of value equal to the line voltage at that time (utilizing the assumption that the output is constant on the scale of the switching period). Only the positive half of the AC line cycle needs to be considered due to symmetry of the cycloconverter (that is, only the case of $V_{out} > 0$ will be solved). This model will be used to calculate the appropriate control inputs which will satisfy power delivery requirements at the highest possible efficiency.

4.3.1 Governing equations

An even simpler model replaces the loaded cycloconverter with an AC voltage source equal to v_{CC}, as shown on Fig. 4.3. This is simply an RLC circuit excited by the

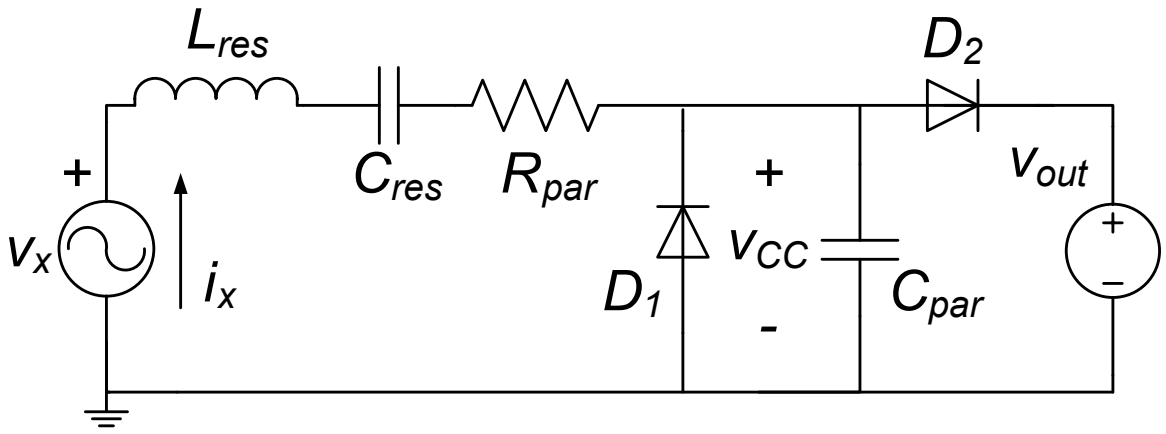


Figure 4.2: A simplified inverter model.

difference of the two voltage sources. Phasor analysis provides an easy way to solve for current in the main loop. Since multiple switching harmonics are present in both AC sources, the reference frequency of nth Phasor equation is $n \times f_{SW}$ for $n = 0, 1, 2 \ldots$. Fourier series coefficients of both AC waveforms are the complex amplitudes in the Phasor equations. The Fourier series coefficients of v_x are the coefficients of v_{FB}, given by equation (3.2), scaled by N. The phase reference will be chosen as the fundamental of v_x, or equivalently, such that the pulse τ of v_x on Fig. 4.1 is centered about $\omega_{SW} t = 0$. Consistent with this phase reference, the time domain expression for v_x is given by equation (4.2). The $2n + 1$ use of index indicates that the sum is taken over odd harmonics (as the even ones are zero).

$$v_x(t) = \sum_{n=0}^{\infty} N \frac{4V_{IN}}{(22n+1)\pi} \sin \frac{(n+1)\delta\pi}{2} \cos(2n+1)\omega_{SW} t \tag{4.2}$$

The impedance of the RLC network is given by equation (3.13). For convenience it is repeated here for the nth switching harmonic. Finally, the harmonic content of the cycloconverter waveform needs to be found in order to solve for the resonant current. This will be done in two steps - once for a simplified case, and once for a more realistic case. In both cases, however, the solution to the RLC problem is given

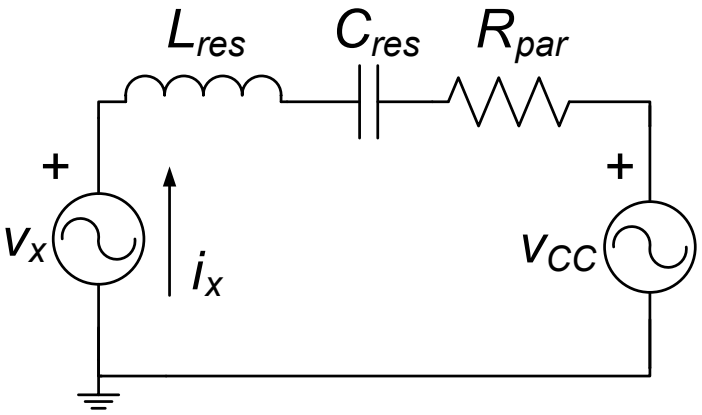


Figure 4.3: The simplest model for the inverter.

by equation (4.4). Once the complex amplitudes of each waveform are obtained, the rule for reconstructing the time domain waveform (in general and particularly for current) is given by equation (4.5).

$$Z_n = \frac{1}{jn\omega_{SW}C_{res}} + jn\omega_{SW}L_{res} + R_{par} \tag{4.3}$$

$$\langle v_{FB}\rangle_n - \langle v_{CC}\rangle_n = \langle i_x\rangle_n \times \langle Z_{equivalent}\rangle_n \tag{4.4}$$

$$i\,(t) = \sum_n \langle i\rangle_n \cos n\omega_{SW}t = \sum_n \|\langle i_{FB}\rangle_n\| \cos\left(n\omega_{SW}t + \angle\langle v_{FB}\rangle_n\right) \tag{4.5}$$

4.3.2 Ideal diode rectifier

The case of $C_{par} = 0$ will be considered first. Under this condition, the ideal diodes start conducting under any positive current. Thus for $i_x > 0$, $V_{CC} = V_{out}$ and for $i_x < 0$, $V_{CC} = 0$. The waveform v_{CC} is a square wave which varies between levels of

0 and V_{out} with instantaneous rising and falling edges. The transitions occur at the zero crossings of i_x. Defining θ_1 as the angle at which the resonant current crosses zero while rising (all with respect to the zero reference chosen above). The Fourier series coefficients of v_{CC} are then given by equation (4.6).

$$\langle v_{CC} \rangle_n = \begin{cases} \frac{V_{out}}{2}, & n = 0 \\ 0, & n = 2,4,6\ldots \\ \frac{2V_{out}}{n\pi}\left(-\sin n\theta_1 - j\cos n\theta_1\right), & n = 1,3,5,\ldots \end{cases} \tag{4.6}$$

The angle θ_1 is unknown. However, by its definition, evaluating the time domain waveform for current at $\omega_{SW}t = \theta_1$ results in a net current of zero. Thus the problem can be reduced to two expressions which must be satisfied for consistent solutions. These are given by equations (4.7) for odd n. Note that the DC case of n=0 is not considered as the tank impedance is infinite at DC due to the capacitor and thus no DC current is conducted. Additionally all even harmonics are not considered as the waveforms of interest have zero even harmonic content.

$$\langle i_x \rangle_n = \frac{N\frac{4V_{IN}}{n\pi}\sin\frac{n\delta\pi}{2} - \frac{2V_{out}}{n\pi}\left(-\sin n\theta_1 - j\cos n\theta_1\right)}{\frac{1}{jn\omega_{SW}C_{res}} + jn\omega_{SW}L_{res} + R_{par}} \tag{4.7a}$$

$$\sum_n \left\|\langle i_x \rangle_n\right\| \cos\left(n\theta_1 + \angle\langle i_x \rangle_n\right) = 0 \tag{4.7b}$$

$$P_{in} = \sum_n \frac{1}{2}\left\|\langle v_x \rangle_n\right\| \left\|\langle i_x \rangle_n\right\| \cos\left(\angle\langle v_x \rangle_n - \angle\langle i_x \rangle_n\right) \tag{4.8a}$$

$$P_{out} = \sum_n \frac{1}{2}\left\|\langle v_{CC} \rangle_n\right\| \left\|\langle i_x \rangle_n\right\| \cos\left(\angle\langle v_{CC} \rangle_n - \angle\langle i_x \rangle_n\right) \tag{4.8b}$$

Essentially, an arbitrary number of otherwise independent frequency domain equations needs to be solved under a time-domain imposed constraint. These equations can be solved using a numerical procedure. An error function can be defined as the difference between an assumed θ_1 and the zero crossing of the current waveform which is the result of solving equation (4.7a) given that θ_1 and substituting the coefficients into (4.5) to solve for the time domain current. Obtaining the input and output power, given the Phasor solutions can be done using equations (4.8). This procedure has been implemented in MATLAB and the built-in solver 'fsolve.m' was used to converge onto the consistent value of θ_1. The code can be found in Appendix A.

To validate this approach, the circuit was simulated in LTspice IV (see Appendix B for the listing). Switching frequency, circuit component values, input and output voltages were changed to compare the two results for different output powers (although the presented data does not rely on a systematic variation of parameters). Figure 4.4 shows the error in power transfer predicted by the numerical solver referenced to the values obtained in LTspice, plotted versus the number of odd harmonics which was considered in the model (e.g. 1 refers to just the fundamental, 2 refers to the fundamental and the 3^{rd} harmonic, and so on). Although the abscissa values are discretized, the point are connected for visual clarity. The fact that the error converges to zero as the number of harmonics is increased attests to the fact that this approach is probably valid. However, several other things are of interest on this figure.

- Error in power accuracy can be above ten percent when considering only the fundamental harmonic. This is a function of the quality factor of the tank, but since it did not vary significantly from the value expected in the actual inverter, it is safe to assume that ignoring higher order harmonics will give significant error in predicting power transfer. Adding the third harmonic already ready reduces the error to within five percent. As will be seen in chapter 5, it is not practical to

consider any harmonic beyond the fifth in the real circuit, and the error at that point is typically capped to within two percent.

- The error seems to be worse for four odd harmonics than for three. This phenomenon can be explained when looking at how the current zero crossing angle θ_1 converges (which is the main variable of iteration in this model). Figure 4.5 shows the error of this model in predicting the angle, referenced to LTspice, in degrees. The error converges to zero degrees for large number of harmonics, but it does so in an oscillatory manner. Overshoot and undershoot of θ_1 is what causes the larger deviation in power on the seventh harmonic.

- Another observation which emphasizes the importance of accuracy of θ_1 is the plot of the power transfer as a function of the harmonic number, shown on Fig. 4.6 for the case of 50 odd harmonics. The power transfer at any harmonic beyond the third is approximately zero. The error in power estimate for low number of harmonics comes from the misestimation of the angle, and not from ignoring the power transfer through higher order harmonics. The sensitivity of power transfer to errors in the angle directly depends on the current magnitude and, as a consequence of this, the error is higher at higher levels.

4.3.3 Diode rectifier with non-zero capacitance

Increasing the capacitance from zero to a finite value changes the shape of v_{CC} [20]. The rise time is no longer zero and the rising and falling edges have curvature, due to the capacitive charging. The model of Fig. 4.3 is still valid, however a new frequency decomposition of v_{CC} is needed. Assuming that capacitance C_{par} is linear, the waveform v_{CC} may be approximated by a quasi-square wave which rises some time after resonant current crosses zero while rising. Specifically this is the time during

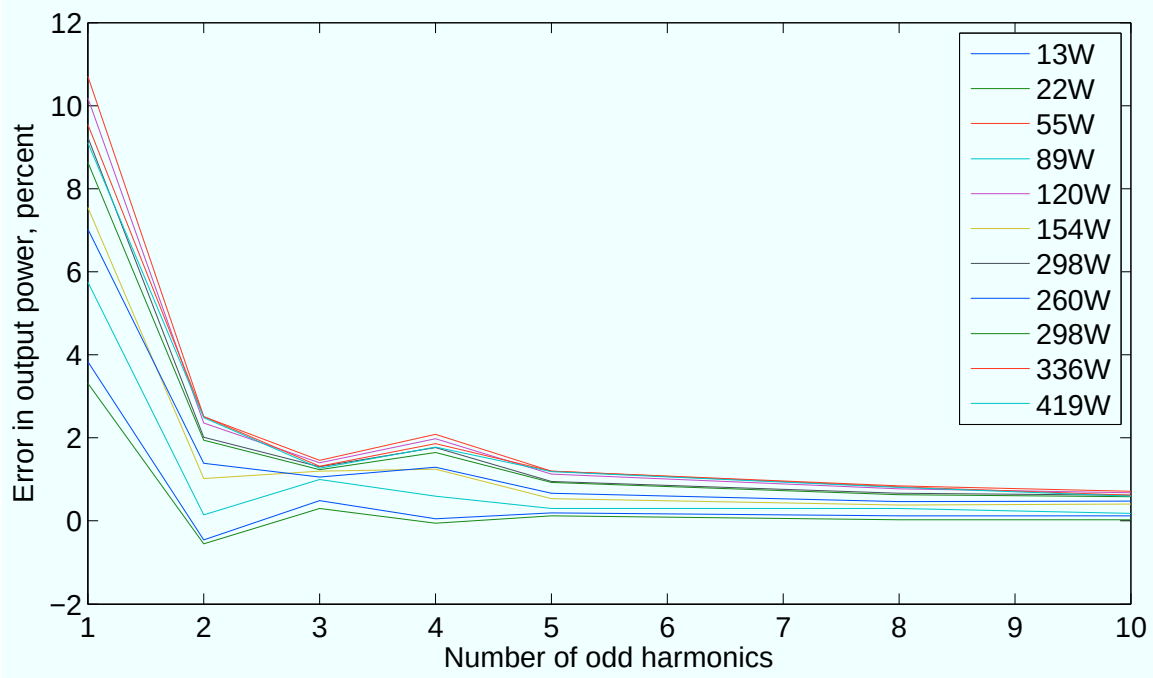


Figure 4.4: Error in power transfer predicted by the model for $C_{par} = 0$.

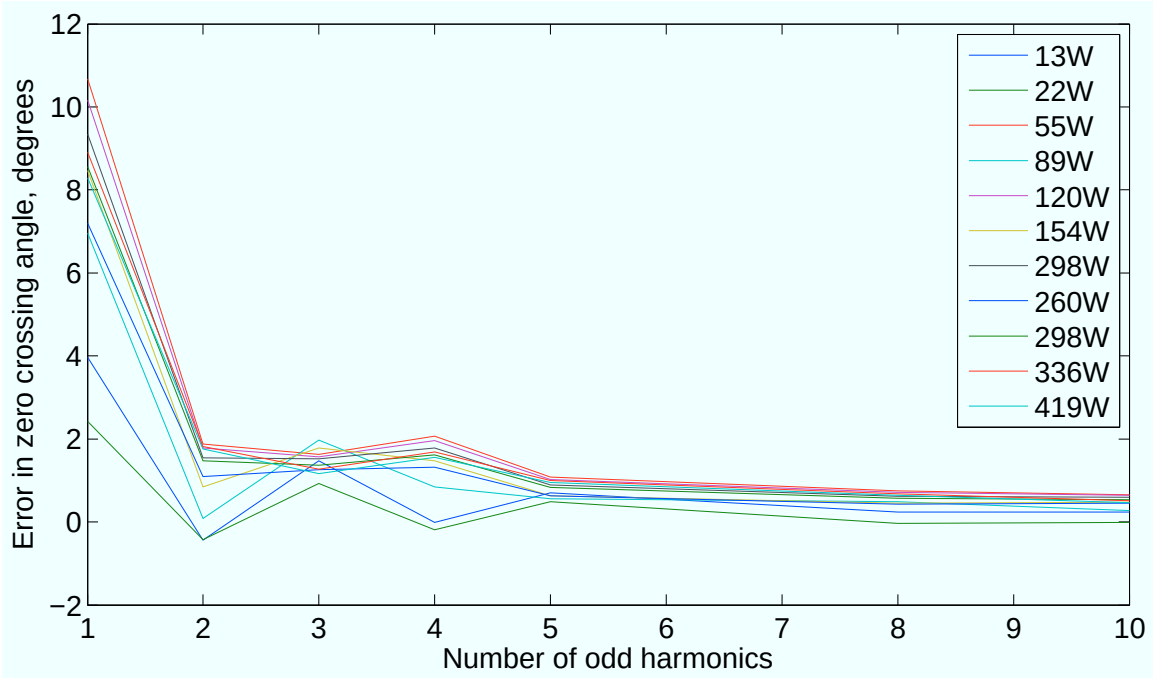


Figure 4.5: Error in zero crossing angle predicted by the model for $C_{par} = 0$.

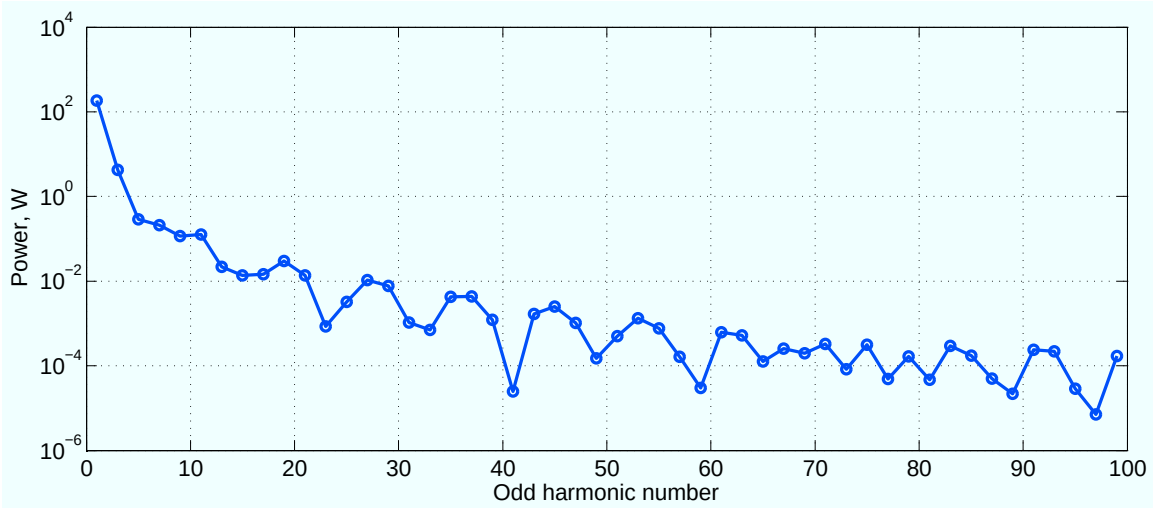


Figure 4.6: Power transfer over 50 odd harmonics for $C_{par} = 0$. Total power output of 190W.

which the current delivers half of the charge that is stored in C_{par} when charged to V_{out}. Defining $\theta_{i_x,0}$ as the angle along the switching period at which this zero crossing occurs, equation (4.9) presents a relationship for θ_Q as the angle at which half of the required charge is delivered. Although for the diode rectifier $\phi_s = 0$, adding this term will make the procedure applicable for the minimum frequency mode control (it will be shown later that $\phi \approx \phi_s$).

$$\frac{1}{2} C_{par} V_{out} = \frac{1}{\omega_{SW}} \int_{\theta_{i_x,0}+\phi_s}^{\theta_Q} i_x(\tau)\, d\tau \tag{4.9}$$

The procedure to arrive at the solution is similar to the case of no capacitance. Given all of the voltage levels (input and output) and all of the control inputs, the equations (4.10) need to be solved numerically for a consistent set of values (θ_Q and $\theta_{i_x,0}$). Doing so allows to find the Fourier coefficients of v_{CC} and i_x and equations (4.8) can be used to find the power flow in the circuit.

$$\langle i_x \rangle_n = \frac{N\frac{4V_{IN}}{n\pi}\sin\frac{n\delta\pi}{2} - \frac{2V_{out}}{n\pi}\left(-\sin n\theta_Q - j\cos n\theta_Q\right)}{\frac{1}{jn\omega_{SW}C_{res}} + jn\omega_{SW}L_{res} + R_{par}} \tag{4.10a}$$

$$\sum_n \|\langle i_x \rangle_n\| \cos\left(n\theta_{i_x,0} + \angle\langle i_x \rangle_n\right) = 0 \tag{4.10b}$$

$$\frac{1}{\omega_{SW}} \int_{\theta_{i_x,0}+\phi_s}^{\theta_Q} \left(\sum_n \|\langle i_x \rangle_n\| \cos\left(n\tau + \angle\langle i_x \rangle_n\right) \right) d\tau = \frac{1}{2} C_{par} V_{out} \tag{4.10c}$$

Note that for the case of $C_{par} = 0$, the two sets of equations become identical. The MATLAB implementation of these models is thus the same (see Appendix A for how to switch between the two). Likewise, the same LTSpice listing can be used for verification (see Appendix B) but now the value of C_{par} is changed from zero. Figure 4.7 shows the error in power delivery for several different levels of output power and C_{par} (which was selected to be in the general range of the cycloconverter MOSFETs). The error is much larger than in the ideal rectifier case. Furthermore, it does not converge to zero as more harmonics are used in the model. However, the error is capped to within seven percent when the third harmonic is added, and gets slightly better after the fifth. The percentage error also varies with the value of C_{par}. As shown on Fig. 4.8, the error increases for larger deviation from ideal rectifier behavior (e.g. as C_{par} increases).

4.3.4 Extension to the real inverter

The error in power delivery resulting from the capacitive diode rectifier model is too high to allow for a completely open loop operation based only on the predictions of this simplified model. However, if the error is indeed capped by about seven percent,

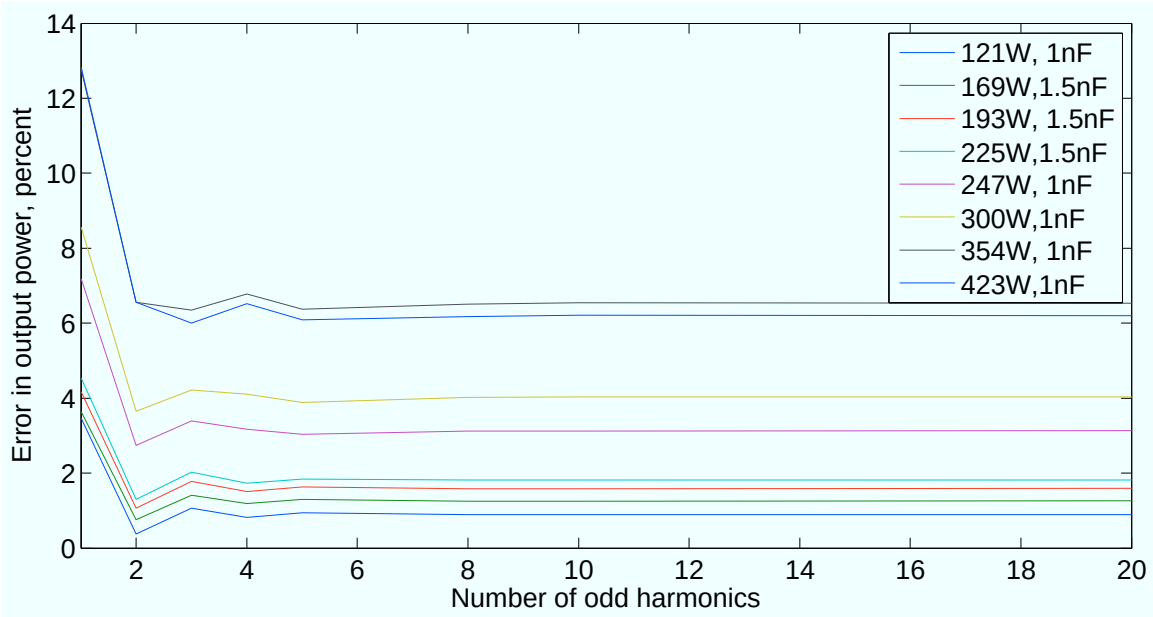


Figure 4.7: Error in power transfer predicted by the model for $C_{par} > 0$.

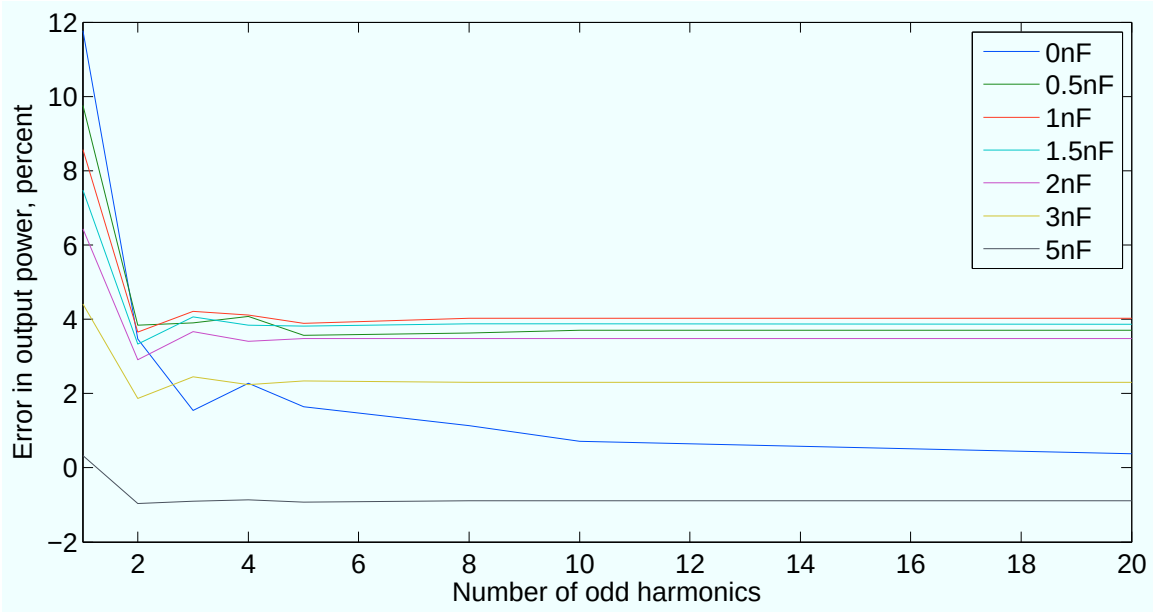


Figure 4.8: Changing C_{par} while keeping everything else constant. The nominal power is 300W (it changes slightly with C_{par}).

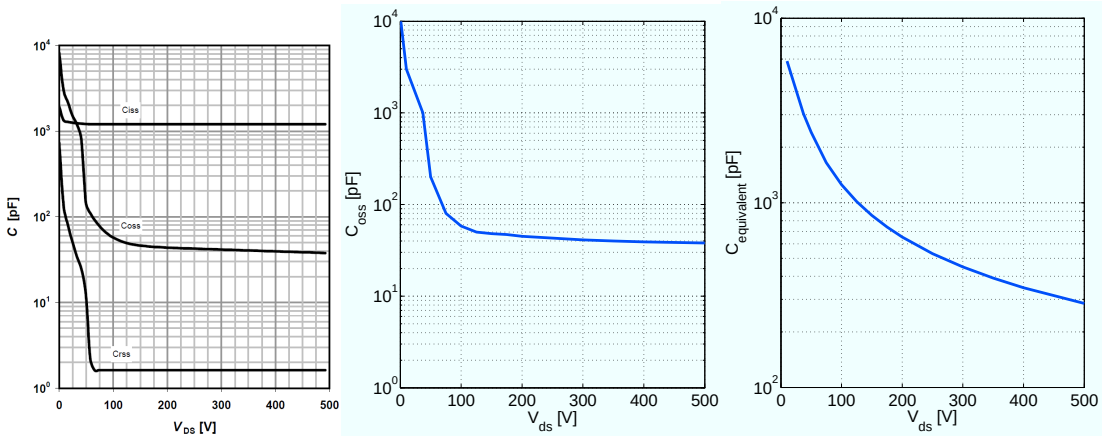
a feedback loop can provide acceptable power delivery within a few correction steps through the use of this model. Furthermore, fluctuations in input and output voltages and component values affect the prediction to the extent that feedback will be likely be necessary for any model that is used. In order to be able to use the model for the first order control input prediction given a desired operating point, several additional constraints need to be imposed on the capacitive rectifier model.

4.3.4.1 Equivalent charge

It is imperative that the correct value for the parasitic capacitance is used. The equivalent concept for a MOSFET is the output capacitance C_{oss}. It is a nonlinear capacitance and is typically given as a function of the drain to source voltage. For use in the model, the equivalent linear capacitance value which was found to work well experimentally is given by equation (4.11). The instantaneous capacitance $C = C_{oss}(V_{ds})$ was determined to be a poor choice. Some manufactures also specify the output energy as a function of drain to source voltage. Extracting the equivalent capacitance using the relationship $C = \frac{2E_{oss}(V_{ds})}{V_{ds}^2}$ turned out to be a less accurate prediction than equation (4.11). This is true for FETs on both the full bridge and the cycloconverter. For the full bridge $V_{ds,FB} \approx V_{IN}$ and for the cycloconverter $V_{ds,CC} \approx V_{out}$.

$$C_{equivalent}(V_{ds}) = \frac{1}{V_{ds}} \int_0^{V_{ds}} C_{oss}(\nu) d\nu \tag{4.11}$$

$C_{oss}(\nu)$ typically needs to be obtained by curve fitting to the datasheet plot of output capacitance versus voltage, for example as shown on Fig. 4.9. Also shown on the



(a) C_{oss} is in center (b) C_{oss} curve fit (c) Equivalent capacitance

Figure 4.9: Deriving capacitance for Infineon IPP60R250CP MOSFET. Plot (a) is taken from the datasheet.

figure is the resulting equivalent capacitance as a function of voltage. Due to Miller effect, it is typical for it to decrease with voltage.

4.3.4.2 Critical angles for ZVS

The aforementioned $\phi_{critical}$ is the angle during which $C_{equivalent}$ of the cycloconverter MOSFETs is charged to v_{out}. Solving equations (4.10) with $C_{par} = C_{equiv,CC}$ automatically determines $\phi_{critical}$, as given by equation (4.12). The subscript 'CC' refers to the fact that the cycoconverter capacitance is used. This is the minimum cycloconverter phase shift which is still consistent with ZVS. This definition gives rise to the angle ϕ_s (see equation (4.13)) as any cycloconverter phase shift beyond what is minimally required (subscript 's' denotes sloshing). For the lowest current mode, ϕ_s should be set to zero. For the lowest frequency mode, ϕ_s is positive.

$$\phi_{critical} = \theta_{Q,CC} - \theta_{i_x,0} - \phi_s \tag{4.12}$$

$$\phi = \phi_{critical} + \phi_s \tag{4.13}$$

Equation (4.10c) is equally applicable to the full bridge. Having obtained the solution to equations (4.10) using the cycloconverter capacitance, solving equation (4.10c), with $C_{par} = C_{equiv,FB}$ and a V_{IN} in place of V_{out}, results in $\theta_Q = \theta_{Q,FB}$ as the angle during which the capacitance of the full-bridge FET is charged. Note that technically the equation solves for the angle that is after the zero crossing of the current, which would not be consistent with ZVS as shown on Fig. 4.1. However, due to symmetry of current waveform, the net amount of charge delivered before the zero crossing is the same as it is after the zero crossing in the same time interval. The polarity is the only difference and this is consistent with ZVS on the full-bridge. Angle $\theta_{critical}$ is then given by equation (4.14).

$$\theta_{critical} = \theta_{Q,FB} - \theta_{i_x,0} \tag{4.14}$$

The value C_{par} of the model is the total capacitance of the two diodes. In all of these equations, an equivalent capacitance is used for each half-bridge. Because there are two MOSFETs, the total capacitance may be approximated[1] by doubling the single device equivalent capacitance, which is derived from the datasheet.

[1]Approximated (and not exactly) because the values of each FET's C_{oss} start out at different voltages: one charges and the other discharges.

4.3.4.3 Deadtime requirements

Selecting proper deadtime is a critical step in balancing efficient (in terms of minimizing diode conduction loss) and robustness of ZVS switching in the inverter. As mentioned in chapter 3, for high efficiency it is desirable to conduct resonant current through FETs in their on state, which means minimizing deadtime. However, too little deadtime may result in hard switching which causes overlap loss and voltage spikes. This is particularly important on the cycloconverter, where the FETs interact with high voltage and spikes due to hard switching may cause failure. Even if the FETs do not fail due to voltage, the increase in switching loss is typically higher than the body diode conduction loss. Therefore, it is better to overestimate deadtime than to underestimate it. The deadtimes are given by equations (4.15). The safety factors 'SF' are typically greater than 1 and are meant to absorb the shortcomings of the model. One such shortcoming comes from the fact that the value of $\theta_{Q,CC}$ was calculated as the angle during which only half of the total half-bridge charge is delivered. This is valid when estimating Fourier coefficients but for deadtime purposes the entire amount of charge needs to be delivered. Given the solution to the model (e.g. i_x and v_{CC} are known), equation (4.10c) may be solved for $\theta_{Q,CC}$ using the full equivalent capacitance of the half-bridge. Alternatively, a safety factor of $\approx$ 1.4-2 may be used in equation (4.15b). This factor is an approximation to how much extra deadtime is needed to commutate through two FETs' parasitic capacitances. The high end value of 2 is the most conservative case as the total deadtime has to be less than twice the time it takes to bring v_{CC} to $\frac{1}{2}V_{out}$, since current is rising at the transition. The safety factor for the full-bridge may be much less conservative, as the large current amplitude allows for fast commutation in those devices.

$$\text{deadtime}_{FB} = \text{SF}_{FB} \times \frac{\theta_{Q,FB}}{\omega_{SW}} \tag{4.15a}$$

$$\text{deadtime}_{CC} = \text{SF}_{CC} \times \frac{\theta_{Q,CC}}{\omega_{SW}} \tag{4.15b}$$

4.3.4.4 Cycloconverter turn on time

Since all timing was referenced to the center of the positive pulse of the full-bridge, the timing of those FETs is well defined (it will be quickly mentioned in experimental setup description of chapter 5). Timing is less clear for the cycloconverter. It is extremely important that it not be turned on too early as that will cause hard switching. For the positive half of the line cycle, the high side of the modulating leg (D_H on Fig. 2.2) turns on at an angle given by equation (4.16). The safety factor $\text{SF}_{CC,on}$ may be chosen to be greater than 1 to ensure that the cycloconverter never turns on too early. For the negative half of line ($V_{out} < 0$), the turn on of the modulating leg's high side (F_H on Fig. 2.2) is given by equation (4.17). The symmetry of the problem allows for computation of timing over only half of the line cycle while shifting by half of a switching period for the other half.

$$\theta_{ON,HS,pos} = \theta_{i_x,0} + \phi_s + \text{SF}_{CC,on} \times \text{deadtime}_{CC} \tag{4.16}$$

$$\theta_{ON,HS,neg} = \theta_{ON,HS,pos} + \pi \tag{4.17}$$

4.3.5 Other considerations

As was seen in chapter 3, both conduction and magnetic core losses are functions of frequency. Therefore, modeling all loss via a fixed resistor loses accuracy when operating over a wide frequency range. This emphasizes the need for feedback as an

alternative to an overly complicated model for loss, which would need to converge not only on the control input and phase angle but also on the equivalent resistance, which would become a function of frequency and current in the circuit.

The approximation of a square v_{CC} is somewhat crude and technically not necessary. Removing curvature was a way to simplify the computation of the Fourier series coefficients. Adding a linear capacitive charging curvature certainly allows for a closed form solution for the Fourier series. However, even for the case of fundamental and third harmonic, the equations become very convoluted, and if the fifth is to be considered, it is even worse. This is unnecessary for two reasons. First is that the capacitance is really non-linear and the presented equivalent capacitance method is arguably just as accurate. Second is that since a feedback controller is necessary anyways, small gain in accuracy of the initial estimate is not very important.

4.3.6 Validation

The final model implementation in MATLAB is shown in Appendix A. Appendix B provides an LTspice model for the inverter operating in the minimal current mode. This model includes full bridge FETs, a transformer, and a capacitive diode rectifier for the cycloconverter. The MATLAB model is in good agreement with LTspice. Adjusting the resistor value in the LTspice model may be helpful for better agreement. Experimental evaluation of these models will be presented in chapter 5.

4.4 Optimization

The goal of optimization is to select the parameters for the main components of the inverter such that the CEC efficiency is maximized when operating using the previously-described control scheme. These are: the resonant inductance and capacitance, the transformer turns ratio and the selection of MOSFETs. The most accurate optimization is achieved by thoroughly calculating control inputs for a representative set of operating points, using the methods described above, for several sets of circuit parameters. Efficiency can be found using the loss estimated formulas of chapter 3 and the best set of parameter will be found. However, this task turned out to be very resource intensive and thus impractical for optimizing over a wide range of parameters. For example, it could take up to two hours on a modern computer to calculate control inputs for one set of parameter values for all the necessary power levels at twenty volt output voltage increments. A much faster way of performing optimization is by analyzing the circuit only at the switching frequency. Fig. 4.6 validates this approach in terms of power transfer. Since the exact timing in the circuit is not important, a good approximation to the appropriate control inputs can be obtained this way. The biggest source of error will be in the cycloconverter phase shift, as higher order harmonics affect the commutation speed. However, the point of optimization is to select a set of parameters which will be better than others, and a consistent source of error in estimation will not significantly affect this choice.

A simplified version of the main control script was developed (see Appendix C for the MATLAB implementation) to calculate the approximate switching frequency and full-bridge and cycloconverter phase shifts for an arbitrary number of operating points along the AC line cycle, and at different average power levels, given all of the defining circuit parameters. A dramatic decrease in computation time came from not having to use a numerical solver to estimate the angle ϕ. Given an estimate for ϕ, the third

harmonic amplitude is easily derived and can be used in estimating the losses in the inverter. The loss models of chapter 3, evaluated at the fundamental and the third harmonic, were incorporated such that a CEC average efficiency number could be calculated.

4.4.1 Resonant components

The most important circuit parameters are the resonant inductor and capacitor values. Since the power output requirements determine the nominal resonant current level under minimal current control, the switching frequency is the only thing that can be changed in optimizing for efficiency. The frequency range is determined almost entirely by the resonant components. To study this tradeoff, values of L_{res} and C_{res} were swept over a wide range and CEC efficiency was computed for each case. Since the inductor is often a custom-made component, an optimal inductor configuration was picked for each inductance value. This included picking the core gap, the wire gauge, and the winding geometry for an RM14 core inductor for a set of frequencies, defined through a typical location away from resonance, and a set peak currents at those frequencies. Appendix C (file 'optimize.m' provides the specifics). This was done in an attempt to ensure that the optimal inductor model is used in each case. The transformer geometry was fixed at an ungapped RM14 core with a certain winding pattern and no parasitic capacitance. Devices which were used for loss analysis were the same ones as used on the prototype of chapter 5. A parasitic resistance of 50 mΩ was included in the circuit.

The CEC efficiency was calculated for the average power levels of [100, 75, 50, 30, 20] percent with the efficiency weighting coefficients of [0.05, 0.53, 0.21, 0.12, 0.09]. This method combines the two lowest power levels of Table 2.4 into a single number,

overestimating the efficiency in each case (which is appropriate given both the low weight and the small amount of energy delivered in that region of operation). For each average power case, ten output voltages were chosen, equally spaced in energy delivery along one quarter of the line cycle. Note that due to symmetry in the case of unity power factor, operation over a quarter of the line cycle is representative of operation everywhere else along the line cycle. The details of how to pick an arbitrary number of voltages equally spaced by energy, see chapter 5 or Appendix C. In finding control inputs, the frequency resolution was chosen to be 0.5 kHz. The operating points were calculated to within 3 percent power output accuracy.

Figure 4.10 shows the estimated CEC efficiency for a 32.5 Volts input, plotted versus the resonant component values. The missing values near the peak are due to either that the magnetic components saturating in certain regions of operation or that the resolution of the control inputs did not yield power output to within a specified accuracy. The 'bumps' in the surface indicate that the optimal inductor geometry changes with the size of the resonant tank. In general, efficiency increases with both inductance and capacitance values. This is directly related to the corresponding decrease in the frequency of operation. Like the CEC average, the efficiency for all the operating points may be averaged using the same weight for each. This defines a kind of an average frequency of operation, which can be used to gauge the effect of the resonant component change. This average frequency is shown on Fig. 4.11. Efficiency increase correlates well with the decrease in average frequency. Operating at the lowest frequency and the lowest quality factor results in highest efficiency. Similar plots may be constructed for any combination of inverter parameters. Without presenting the figures, it was determined that the inverter is more efficient for low input voltage than for high input voltage. The reason for that is also related to frequency: since the output and resonant currents are nominally fixed, at a higher

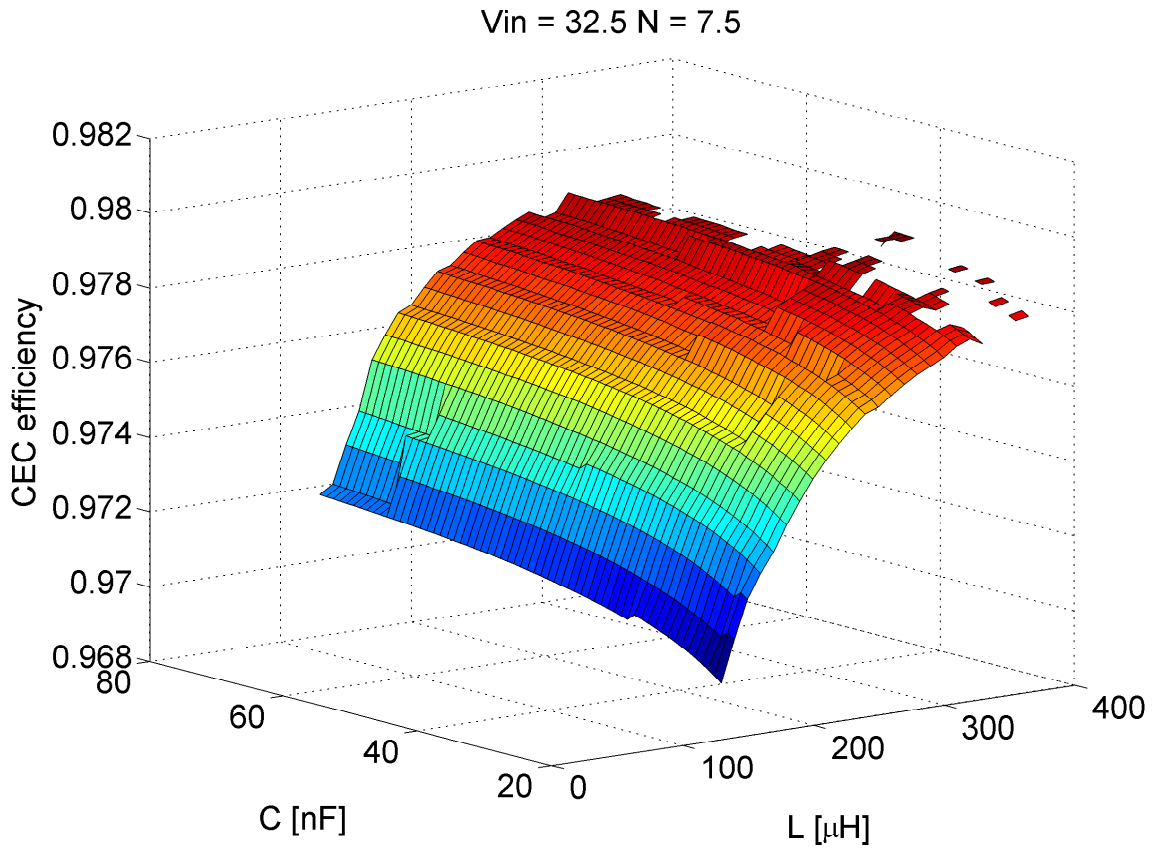


Figure 4.10: CEC efficiency estimate for $V_{IN} = 32.5V$ and $N = 7.5$.

input voltage it is necessary to switch at a frequency which is further away from resonance to produce the same current.

For a fixed inductance, Fig. 4.10 indicates that increasing capacitance has a diminishing impact on efficiency. Moreover, for a characteristic impedance ($\sqrt{\frac{L_{res}}{C_{res}}}$) in the range of about 50-65 Ω the change in efficiency is within about 0.2 percent. This can be attributed to the increased loss through the third harmonic at lower quality factors. This effect slightly reduces the benefit of going to lower frequency. Additionally, lower quality factor means a wider frequency range of operation under minimal current mode, which causes the average frequency not to decrease as fast as it would for an increase in both resonant components while keeping the quality factor constant.

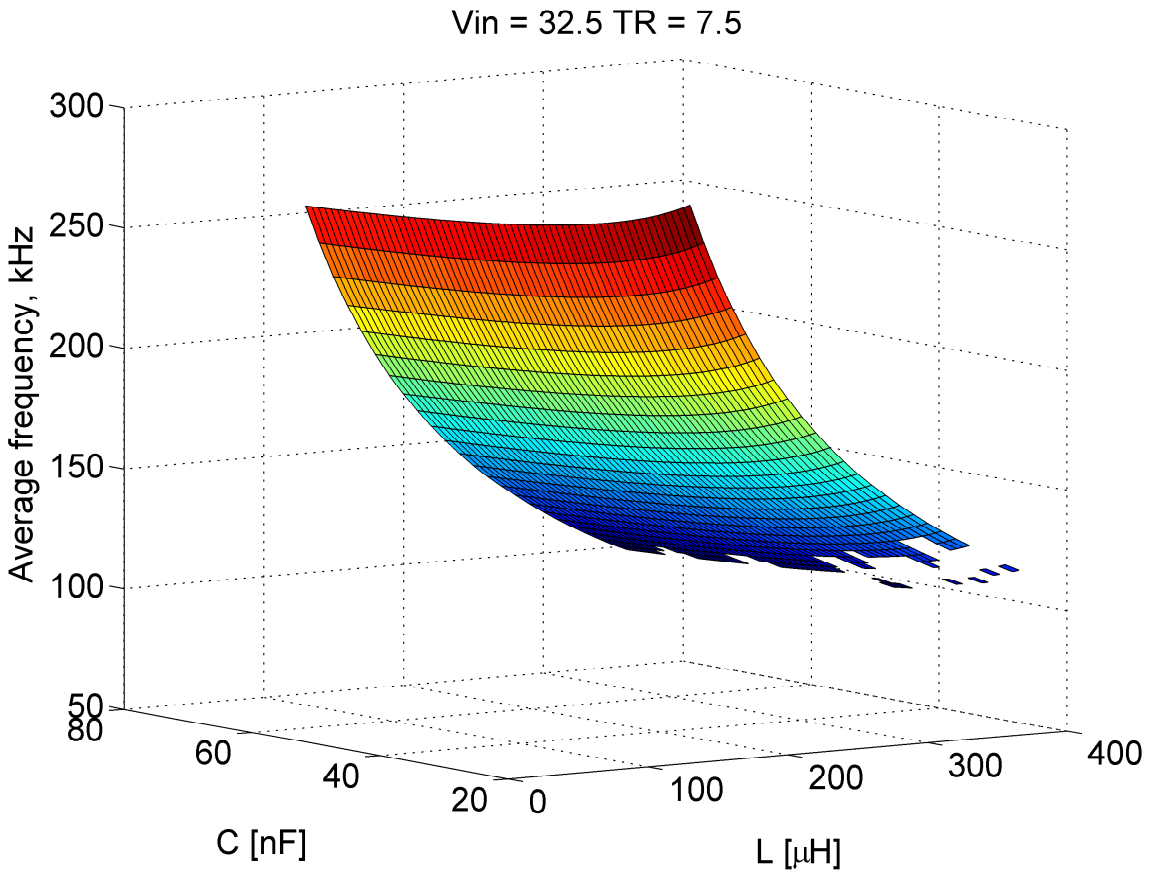


Figure 4.11: Average frequency for Fig. 4.10.

The physical size and cost of passive components put a limit of how low the frequency can be. Assuming the transformer has some fixed component size, independent of the resonant tank values, there is a lower limit on frequency before the core saturates, as discussed in chapter 3. A larger inductance fitted into a fixed core size requires either a larger number of turns or a larger A_L value, both of which bring the core closer to saturation for the same current level, as given by equation (3.18a). Therefore, increasing the physical size of the magnetic components is necessary beyond a certain inductance value. A similar trend is seen for capacitors. Although there is no saturation limit, larger values for capacitance require more physical circuit components or a larger package size. The low loss capacitors required for the resonant tank are expensive and bulky as compared to higher loss ones. In addition, the voltage

rating limits have to be considered, as described in chapter 3. This might motivate the choices of staying a slightly higher quality factor than what is optimal. The size of the capacitive filter bank also has to increase as frequency goes down (this was also mentioned in chapter 3). Overall, the goals of compactness and low cost go directly against the goal of highest efficiency at lowest frequency. Data for different sized resonant tanks will be presented in chapter 5. The general range of the components will be: 150-200 μH for the inductor, 20-45 nF for the capacitor, and a frequency range of 60-450 kHz.

4.4.2 Transformer turns ratio

Since the purpose of the transformer is to amplify the full-bridge voltage, the effect of N on efficiency is similar to that of the input voltage. Lower switching frequency calls for the lower input voltage or v_x on the simplified model. Therefore that N should be chosen to be close to its minimum allowed value (see equation (3.8) for the minimum turns ratio derivation). One difference between transformer amplification and input voltage is the fact that the full-bridge conducts the resonant current of the secondary scaled by N. Apart from frequency considerations, higher N means that every current-dependent loss on the primary side is higher for the same loss (current RMS) on the secondary. This is another reason supporting low N. To provide some margin of safety (e.g. accounting for power losses and fluctuations in output voltage), the turns ratio is chosen to be about ten percent higher than the allowed minimum for a final value of: $N = 7.5$.

4.4.3 MOSFET selection

If the resonant inductance and capacitance can take on virtually any value, the selection of MOSFETs is much more limited by the commercially available products. The important parameters in MOSFET selection are: the gate charge Q_g, the output capacitance $C_{oss}(V_{ds})$ and the on-state resistance $R_{ds,on}$. As mentioned in chapter 3, low Q_g and low $R_{ds,on}$ are typically opposing requirements. The output capacitance typically increases with Q_g. It is desirable to have low output capacitance and the optimal case would be just to have enough so that its discharge will always lag the device turn off with minimum margin. Any extra capacitance gives rise to extra phase shift will results in higher RMS current, leading to higher losses in every component. Due to the low current peaks seen on the secondary, this is particularly important for the cycloconverter. Once the resonant components have been selected, the parameters of several MOSFETs are entered into the simplified loss script (Appendix C) to select the best device. To illustrate the process, a comparison between two cycloconverter MOSFETs is carried out here for L_{res}= 220 μH and C_{res} = 42 nF.

The main parameters of two high voltage devices are shown in Table 4.1. Figure 4.12 shows C_{oss} as found in the datasheet and the equivalent capacitance computed with equation (4.11). One device has over two and half times the on-state resistance of the other while also having less than half of the equivalent output capacitance. The results for comparing the two devices using the simplified script are shown on Fig. 4.13. The abscissa of all subfigures represents the average power level, varied between 1 and 0.1. For each average power 'interval', ten points were taken along the first quarter of the AC line cycle, equally spaced in energy delivery (the peak, from left to right, of each interval corresponds to $240\sqrt{2}$V at the output). For the '99' device, higher cycloconverter phase shift is required as consequence of higher output capacitance, as shown on Fig. 4.13c. The cycloconverter phase shift decreases

Device	Infineon IPP60R099CP	Infineon IPP60R250CP
$R_{ds,on}$ [Ω]	0.099	0.250
Q_g at V_{gs} = 10V, [nC]	60	26
C_{oss} nominal [pF]	130	54
$V_{ds,max}$	650	650
Label on Fig. 4.12	650	650

Table 4.1: Comparing two Infineon devices.

towards the peaks of the line for two reasons: the switching frequency is typically lower in that region and the currents are higher due to the higher power delivery requirement. The extra phase shift results in higher peak resonant current in all operating points (see Fig. 4.13e). Even as the reduced on-state resistance decreases the loss in the cycloconverter (Fig. 4.13f), the total inverter loss is only lower at the highest current levels, which occur near the peaks of the line (Fig. 4.13a). The efficiency for each operating point is shown on Fig. 4.13b and 4.13d. Estimating the overall CEC efficiency for this data results in 97.35% and 97.59% for the '99' and the '250' device respectively. This suggests that the higher on-state resistance devices are slightly better for overall efficiency. However, since the difference in the two efficiencies is very small, smaller than the accuracy of the loss models, the only way to truly confirm this trend is via experimental measurements. It is reasonable to postulate that since the loss models for the magnetic components likely underestimate the total loss, and since loss in these components is sensitive to overall current, the tradeoff will be more favorable to the '250' FETs than the current estimate suggests. Note that the cycloconverter loss distribution is different between the two cases as the '99' device's total has a higher proportion of gating loss (delivering the gate charge to the FET) as seen on Fig. 4.14.

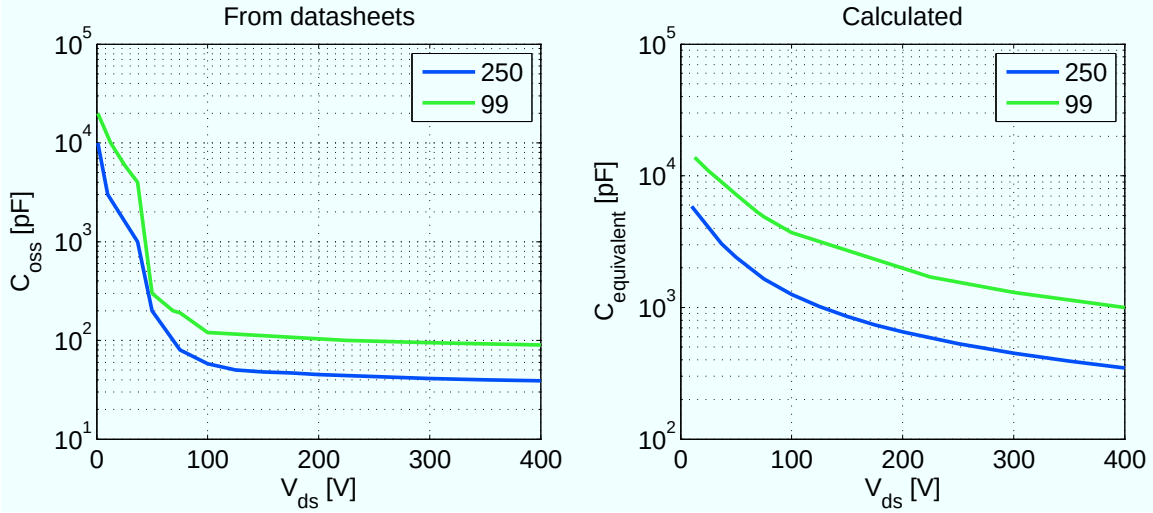


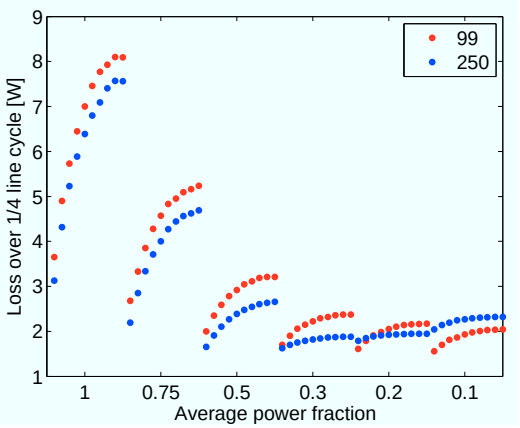
Figure 4.12: Capacitance estimation for FETs of Table 4.1.

4.4.4 Loss breakdown and efficiency

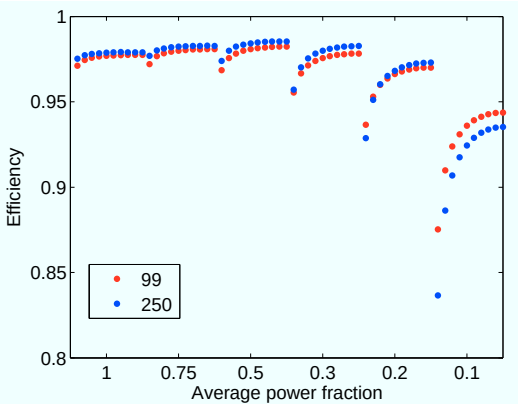
Considering just the case of '250' devices, Fig. 4.15 shows the balance between full-bridge and cycloconverter conduction and gating losses and a fraction of total inverter loss. Fig 4.16 the total loss distribution with some parasitic impedance of 50 mΩ as a realistic ESR. The dominant losses are related to MOSFETs. This is the benefit of the presented topology: as semiconductor devices become better, the efficiency could increase. The peak average efficiency exceeds 97%. This number was obtained based on somewhat realistic component values. Although this is based on simple loss models, assuming that they are within 50% of the correct value, efficiency of over 95% should be possible.

4.4.5 Sloshing boundary

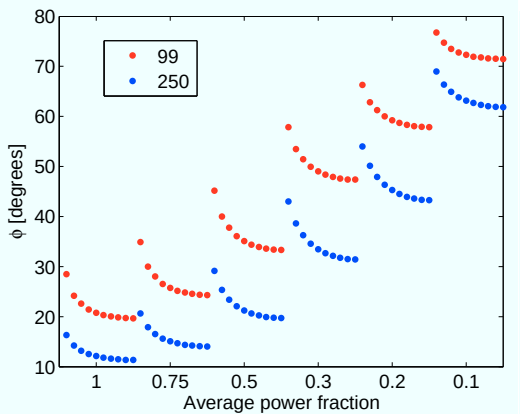
All of the models used so far were looking for control inputs based on the minimum current mode. The original assumption was that there is very little practical need for



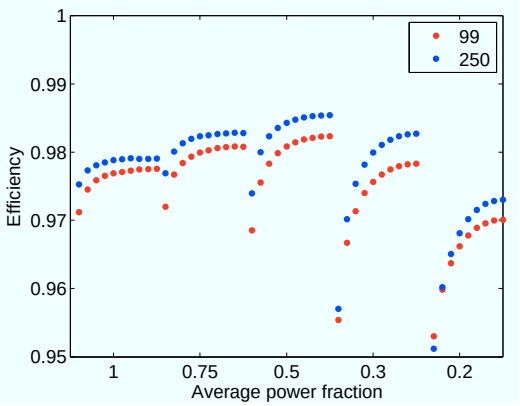
(a) Power loss.



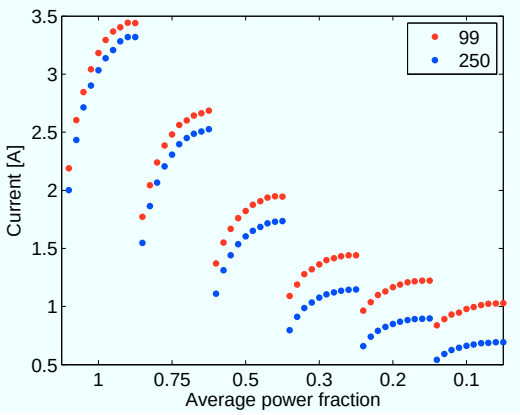
(b) Efficiency.



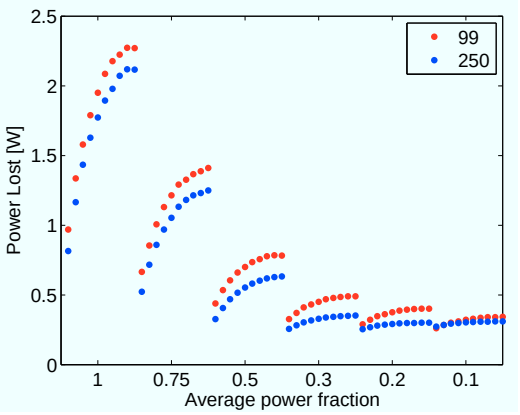
(c) Angle ϕ.



(d) Efficiency close up.



(e) Magnitude of $\langle i_x \rangle_1$.



(f) Cycloconverter total loss.

Figure 4.13: Comparing two MOSFETs in detail. $V_{IN} = 32.5$ V.

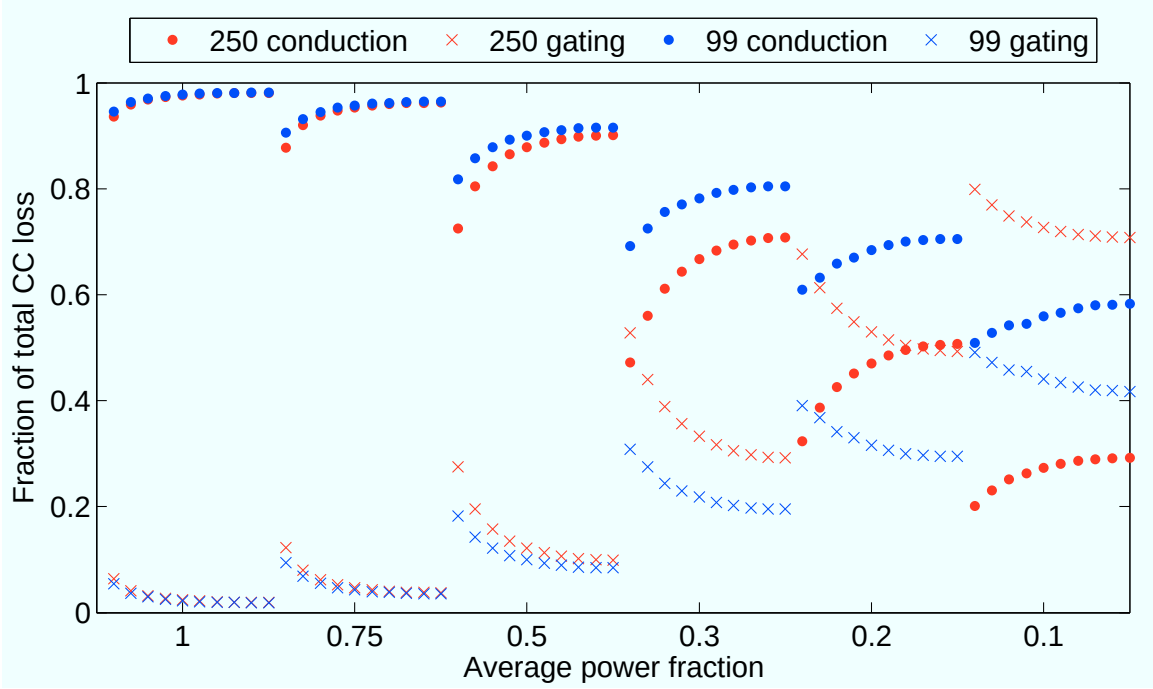


Figure 4.14: Cycloconverter loss breakdown.

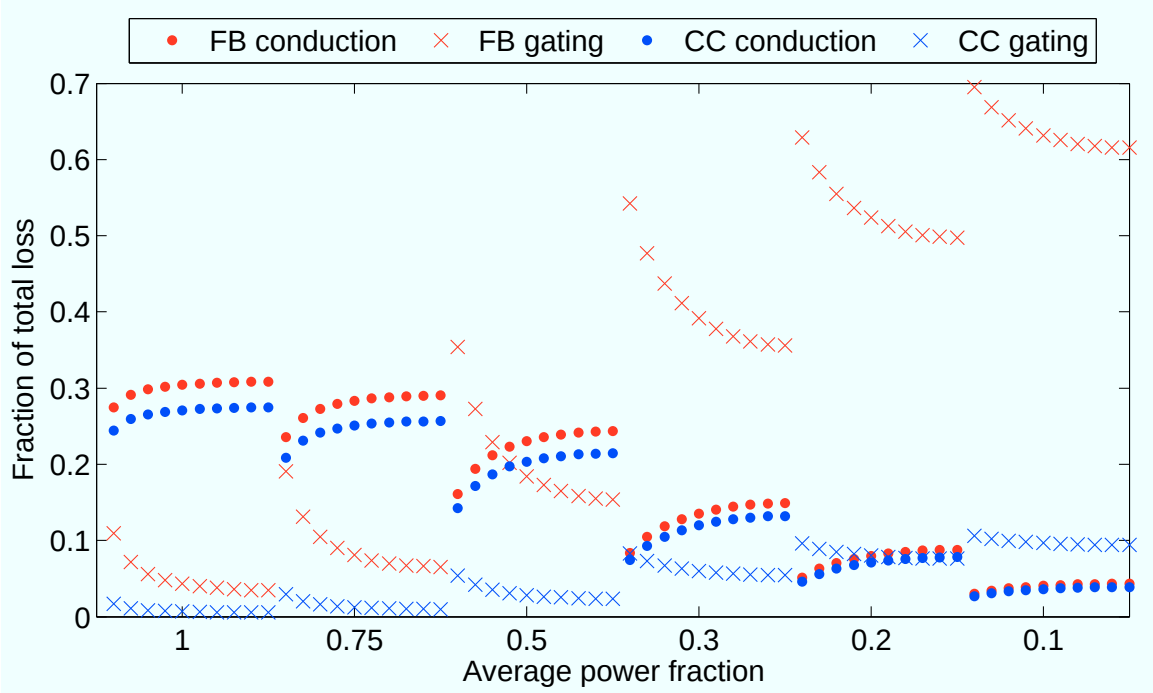


Figure 4.15: MOSFET related loss breakdown.

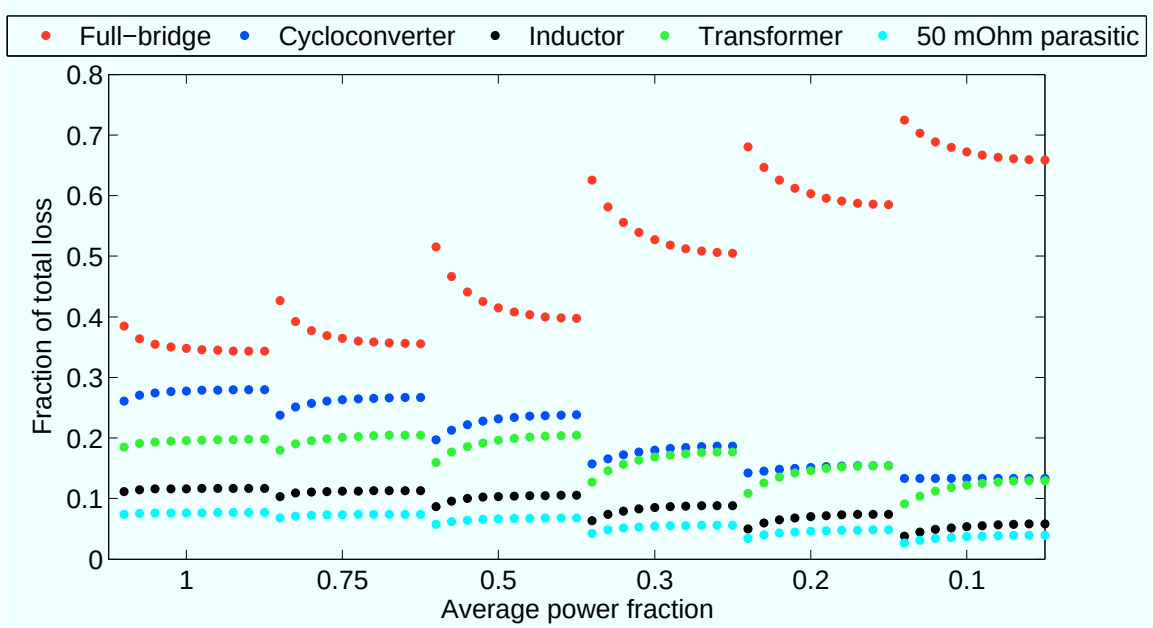


Figure 4.16: Total loss breakdown.

the minimum frequency mode. One way to validate this assumption is by comparing the losses that result from running the same operating point under different control inputs. The case of minimal cyclcoconverter phase shift is compared against larger phase shifts but lower frequencies of operation. In all cases the full-bridge pulse width is kept at the minimum allowed for maintaining ZVS. A MATLAB script to do this and pick the phase shift corresponding to best efficiency was implemented (see Appendix C). The results depend heavily on component parameters and loss models, and implementing an iterative numerical procedure is one of the only ways of making this comparison. To visualize what is happening, Fig. 4.17 shows the efficiency curves for operating points in the .3-.1 average power range. The highest input voltage is chosen because current sloshing would be most effective there (due to higher frequencies). The efficiency is plotted against frequency and the peak of the fundamental of resonant current. The red dot indicates the highest efficiency on each curve. As power output increases, the highest efficiency point shifts toward

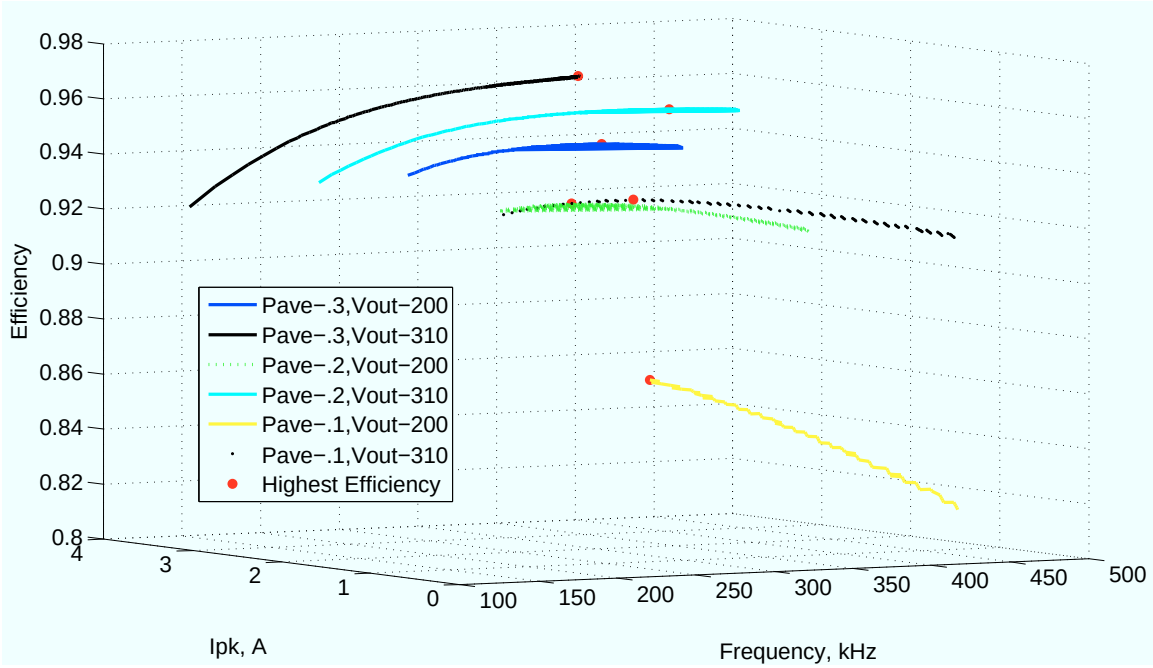
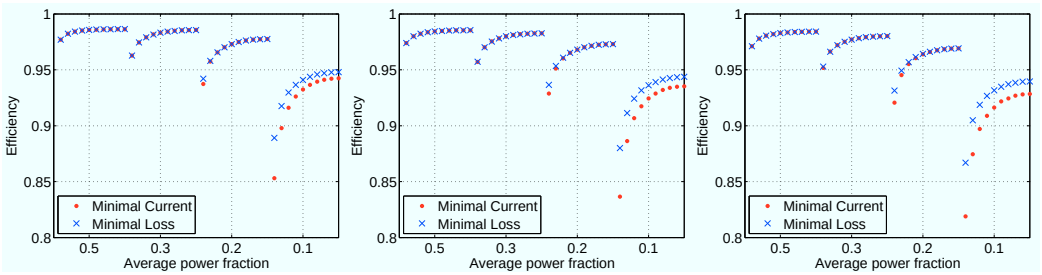


Figure 4.17: Illustration of efficiency versus current and frequency for several operating points. V_{IN}=40V.

the lowest current. Minimum frequency mode provides higher efficiency only at the lowest output powers.

Figure 4.18 shows the efficiency of the minimum current mode compared to the best efficiency when allowing for extra energy sloshing. The four lowest average power levels and three input voltage levels are shown. Energy sloshing helps only near the zero crossings of the AC line cycle at low average power levels, and is more effective for higher input voltage. For the chosen resonant tank, the sloshing boundary is the average power level of 10%. This boundary may occur at a higher average power level, depending on the frequency range of operation. With the presented loss models, it was determined that when switching at above approximately 350 kHz it might be beneficial to consider energy sloshing and operating at a lower frequency. For typical quality factors, this concerns designs with a resonant frequency of about 200 kHz and higher, when using 3F3 magnetic material. Given this information, it was justifiable



(a) $V_{IN} = 25V$. (b) $V_{IN} = 32.5V$. (c) $V_{IN} = 40V$.

Figure 4.18: Comparing impact of energy sloshing.

to combine the 10 and the 20 percent average power levels together for the purposes of Fig. 4.10.

Chapter 5

Verification

THIS chapter will present the experimental results obtained with a prototype of the inverter. After a brief description of the experimental setup, the results will be presented in two parts. The design with the best efficiency will be shown first, along with discussion of accuracy of the control model. As this was not the initial design which was considered, the design evolution will then be presented. Consistency with the optimization theory, presented in chapter 4, will be discussed and major factors contributing to higher efficiency will be pointed out. Finally a method for estimating power losses will be shown.

5.1 Hardware

The prototype board[1] is shown on Fig. 5.1. A detailed schematic and bill of materials for the board are provided in Appendix F for reference. One important non-ideality of this board is trace resistance. Figure 5.2 shows the parasitic resistance of the primary side loop (interconnect from the inverter bridge to the transformer) measured[2] across frequency. This measurement was performed on an unpopulated board with the

[1]Board design and layout performed by Brandon Pierquet of MIT.
[2]This measurement was done by Garet Gamache of Darmouth University.

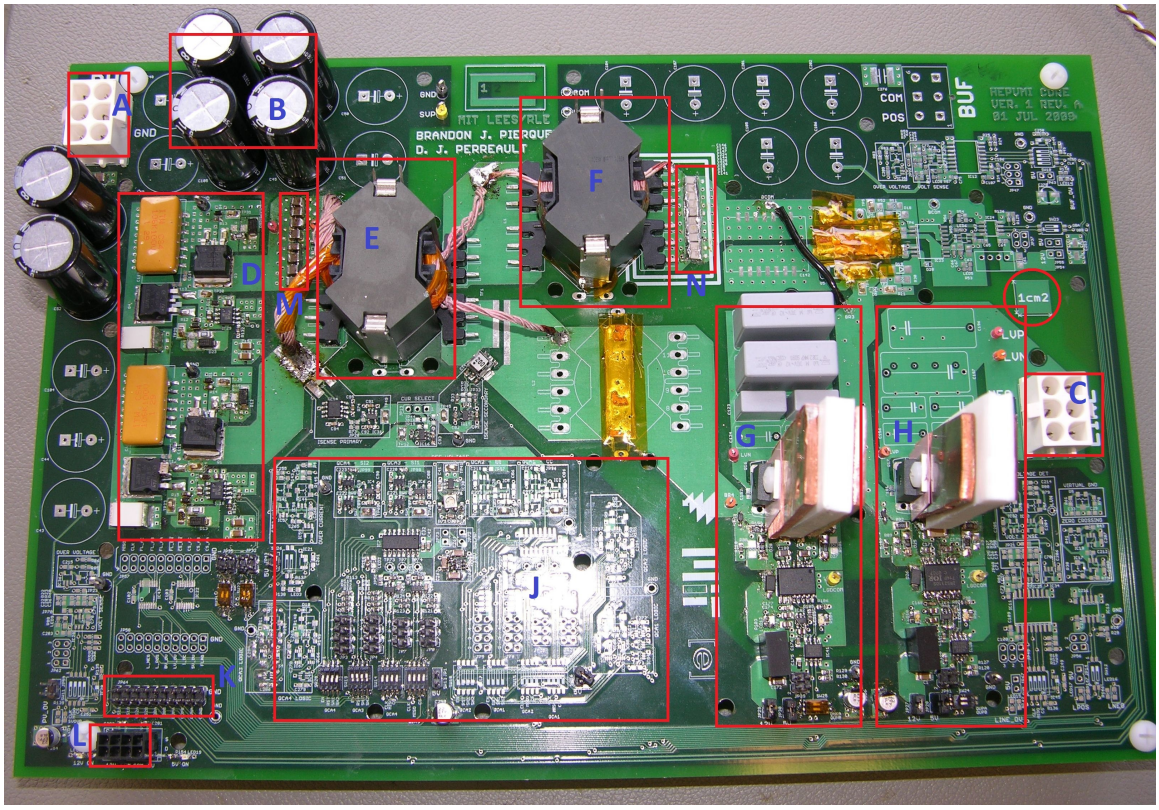


Figure 5.1: Prototype board with major components labeled. A:Input power source; B:Energy storage block; C: Output power connection; D: Full-bridge; E: RM14 core transformer; F: RM14 core inductor; G: Cycloconverter 'negative' leg; H: Cycloconverter 'positive' leg; J: Unused feedback control components; K: Timing signal connection; L: Gating power supply input; M: Blocking capacitor bank; N: Resonant capacitor bank. The red circle circumscribes a 1cm×1cm rectangle for scale reference.

appropriate connections shorted. This resistance causes a non-negligible power loss due to high RMS AC current on the primary.

5.1.1 Experimental setup

The inverter was tested at a series of DC/DC operating points. A diagram and a photo of the experimental setup are shown in Fig. 5.3. The DC power supply at the output (typically a Sorensen DCR 600-3B or a HP6015A) is connected to a large

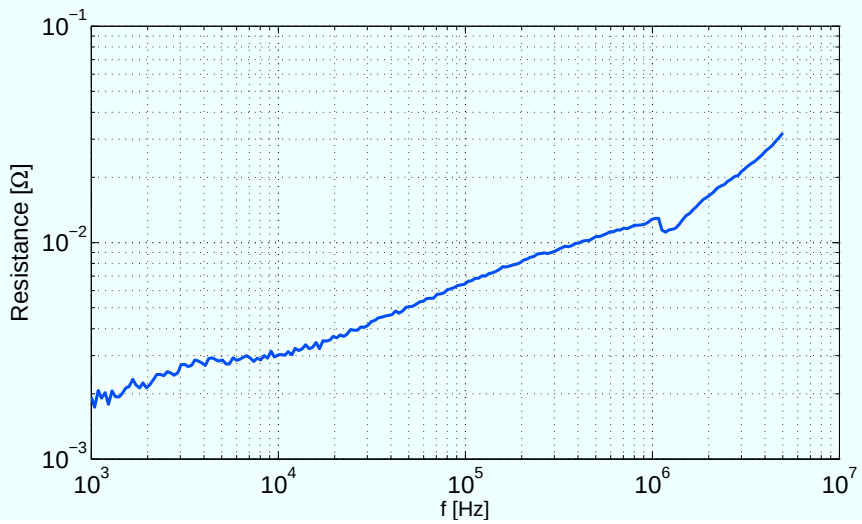


Figure 5.2: Parasitic resistance of the primary side trace loop. This includes the interconnect from the full-bridge to the transformer.

(about 200Ω) power resistor to set up the proper output voltage over a wide range of currents. The power from the inverter and the output supply is dissipated in this ballast resistor. The resistor and the supply are connected via a diode so as to prevent current flowing back into the inverter. The power for gating of all devices is provided by an auxiliary power supply (Tektronix PS280). The input power supply is typically a HP6030A or a HP6643. An estimate for efficiency is obtained by taking current and voltage measurements at all terminals of the inverter (input and output with multimeters[3] and gating power from the front panel display). This estimate ignores the power consumed by the separate FPGA board (Xilinx Spartan-3E Starter Kit from Digilent Inc.). The efficiency measurement is automated, with all of the multimeters connected to the computer (Appendix D shows the code to control the multimeters). Note that the output stage is not grounded, which requires significant common mode chokes to be placed at the output[4] (for a more accurate efficiency measurement and a possibly to prevent damage to the inverter). The input common

[3]The multimeters are: 1 Keithley 179A TRMS for input current and 3× HP 34401a for the other measurements.

[4]A series connection of two 20 mH common mode chokes was used.

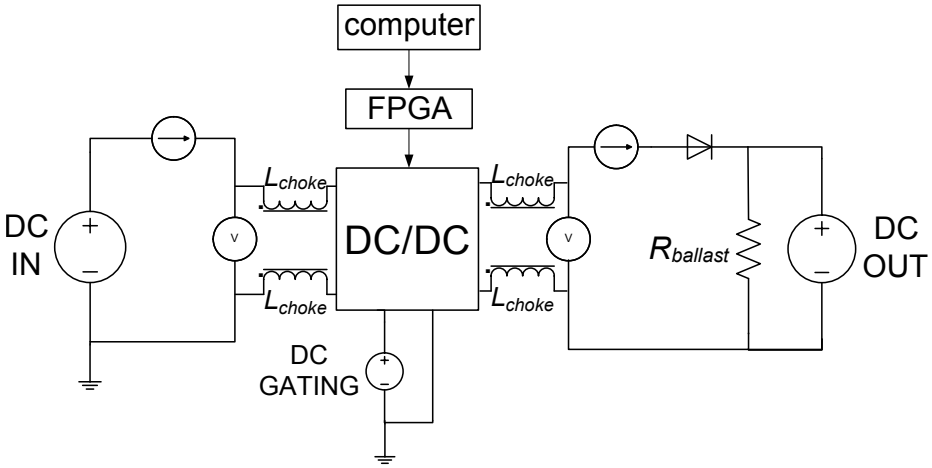
	Full-bridge	Cycloconverter
Turn on delay [ns]	70	260
Turn off delay [ns]	100	290
Turn on gate resistance [Ω]	0	8
Turn off gate resistance [Ω]	0	8

Table 5.1: Turn on delays and gate resistances.

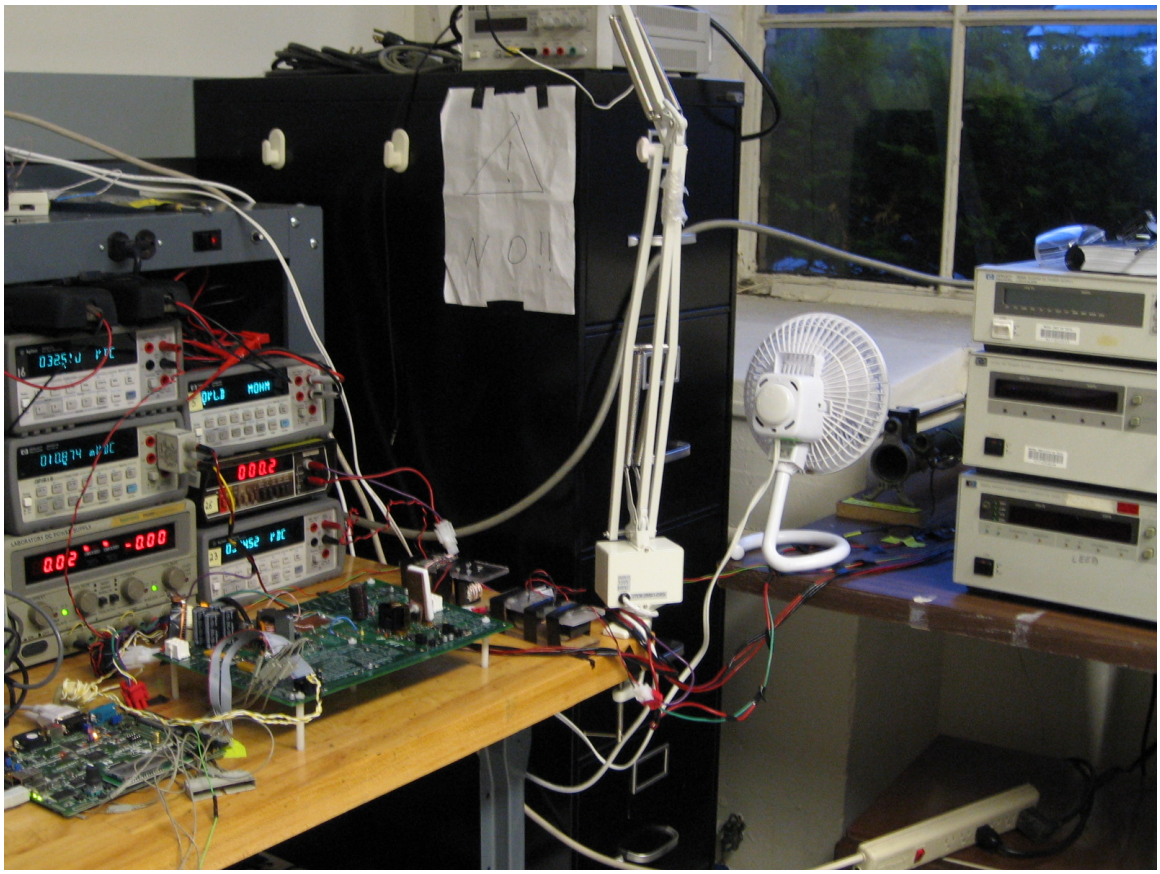
mode chokes are also added to enable a more accurate efficiency measurement. To test an operating point at a negative line voltage, the output connection to the inverter may simply be reversed. In each case, the non-switching leg of the cycloconverter has to be switched on and left in that state for the entire test.

The inverter timing is controlled via a 16-bit FPGA (see Appendix D for details), which is controlled by a computer. The 50 MHz clock of the FPGA development board limits the allowed frequencies of operation to those given by equation (5.1) where **J** is an integer. The clock also limits the timing accuracy to 20 ns. For proper switching of the MOSFETs, it is important to account for the delay between the digital signal command and the gate to source voltage of each device crossing its threshold value. This delay is at a minimum the delay of the gate driver and that due to the input capacitance of each FET. Beyond that, any resistance added between the gate driver and the gate contributes to the delay (adding resistance may help if excessive voltage oscillations occurs at the switching transitions). Table 5.1 summarizes the measured delays and the gate resistor which were used on the final prototype. The cycloconverter gate driver presented significant ringing which required some damping. Note that using a resistor-diode network in series with the gate allows for different turn on and turn off gate resistances.

$$f_{allowed} = \frac{50\text{MHz}}{\mathbf{J}} \tag{5.1}$$

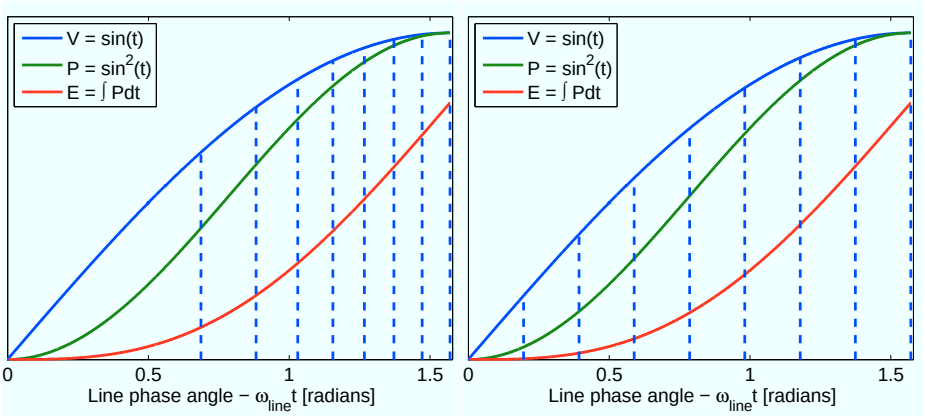


(a) Diagram of the setup.



(b) Photo of the setup. From left to right: FPGA, auxiliary power supply, prototype board, multimeters, common mode chokes (wrapped in black tape), ballast resistor and cooling fan, input and output DC power supplies.

Figure 5.3: Experimental setup.



(a) Equal spacing in energy. (b) Equal spacing in line phase.

Figure 5.4: Graphical view of spacing of 8 operating points in a quarter line cycle.

5.1.2 Selection of operating points

The DC/DC operating points to test may be selected in several ways, three of which are: equal spacing in voltage, equal spacing in line phase angle and equal spacing in energy. When only a few (ex: 4 per quarter of a line cycle) points are to be tested, it is appropriate to space them out by energy, to provide the best approximation to efficiency over that time. When more points are tested (about 7 and more), spacing in phase angle is more appropriate to accurately gauge the performance of the inverter across a wide output range. The reason for this difference arises from the fact that for unity power factor, the equal energy spacing of points favors the points towards the peaks of the line, as power increases as the square of voltage. Efficiency is typically higher near the peak of the line (due to lower frequency) and may thus be overestimated in that case. Spacing in phase angle captures the lower efficiency modes of operation at the lower output voltages. Figure. 5.4 illustrates the two types of spacing, both for 8 points along the quarter line cycle (see Appendix D obtaining similar data for arbitrary number of points).

After obtaining a set of DC/DC efficiencies, the approximation to the overall efficiency is given by equation (5.2) for a general selection of operating points. In the case of equal spacing in line phase angle, the coefficients $\delta(\omega_{line}t)$ cancel out from numerator and denominator. The subscript 'P_x' emphasizes that this is the efficiency for one average power level. Utilizing this for results across several average power levels will yield the data necessary to compute the CEC average efficiency using equation (5.3) with the weights F_x obtained from Table 2.4.

$$\eta_{P_x} = \frac{\int_0^{\pi/2} P_{out}(\omega_{line}t)\, d\omega_{line}t}{\int_0^{\pi/2} P_{in}(\omega_{line}t)\, d\omega_{line}t} \approx \frac{\sum_{(\omega_{line}t)=\theta_1}^{\theta_n} P_{out}(\omega_{line}t)\,\Delta(\omega_{line}t)}{\sum_{(\omega_{line}t)\theta_1}^{\theta_n} P_{in}(\omega_{line}t)\,\Delta(\omega_{line}t)} \tag{5.2}$$

$$\eta_{CEC} = \sum_x \eta_{P_x} F_x \tag{5.3}$$

5.2 Results I

The final inverter design has its resonant tank placed on the primary side of the transformer. This was done to mitigate the loss due to the parasitic capacitance across the secondary of the transformer. Table 5.2 summarizes the main circuit components. Appendix F shows details for supporting circuitry as well as part numbers. The inductor is implemented in a Ferroxcube ETD54 core instead of an RM14 core. The resonant capacitor bank is composed of film capacitors (polyester film dielectric). The components hang off the board (see Fig. 5.5) as the board was originally designed for a secondary-side resonant tank. The inductor value was measured separately from the circuit, and seen from the secondary L_{res} and C_{res} should be 208.6 μH and 44.4 nF, respectively (the actual nominal values for the physical components are listed in Table 5.2). To account for the parasitics in the circuit, the secondary side loop was

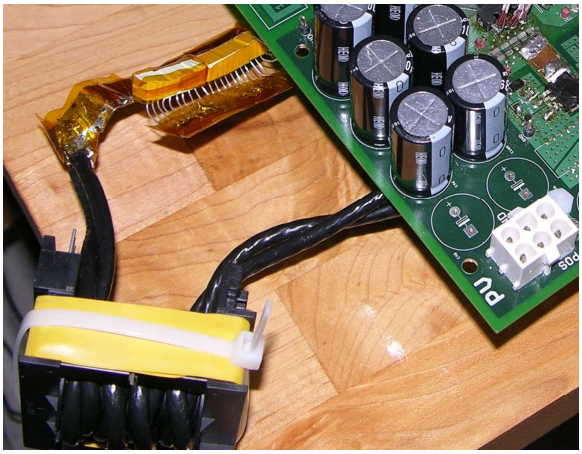


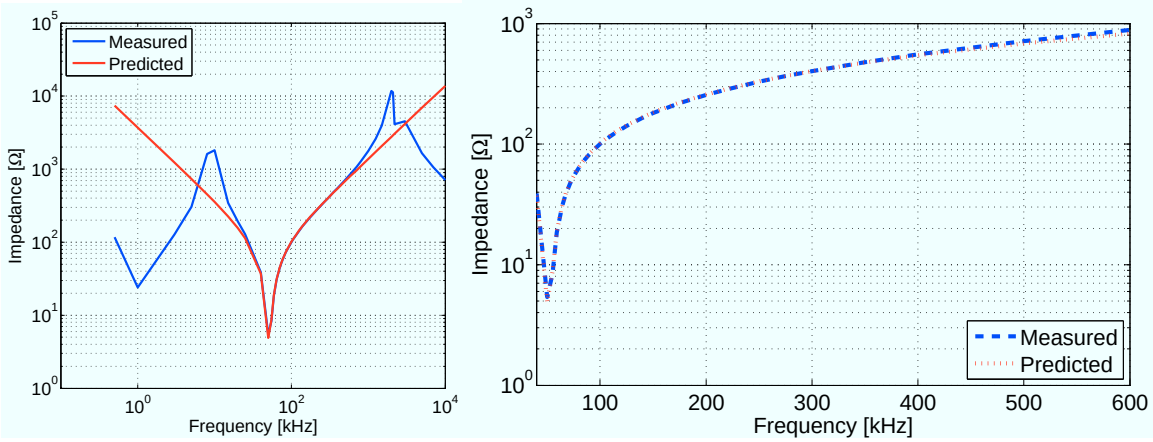
Figure 5.5: Photo of the inductor and capacitor bank.

broken and the board impedance was measured at those terminals on an impedance analyzer (all FETs were soldered short for this). Using the secondary side RLC model, the entire inverter is best modeled by a resonant tank of size 220 μH and 42 nF for L_{res} and C_{res} respectively and a 2 Ω parasitic resistance. The impedance measurement and these fitted values are shown on Fig. 5.6. Although a poor fit over a wide frequency range due to parasitic inductance and capacitance of the resonant components, the fixed parameters are a good match for the range of 50 kHz to almost 1 MHz. This puts a limit on how many harmonics can be considered in the analysis to no higher than the five (and thus only three odd ones).

Given this model of the inverter, a comprehensive table of operating conditions was computed. For an input voltage of 32.5V, the switching frequency, full-bridge pulse width and output voltage were varied. For each combination, the resonant current, output power and gate timing were calculated under minimum current control (using the model of chapter 4). Given an operating point to test, the input and output conditions can be looked up in the table to obtain the corresponding control inputs. If it is determined that the output power is significantly different from desired, a different set of control inputs may be selected such that power output moves in the right direction.

Component	Description
C_{res}	2.5 μF, 25× 0.1 μF, film
L_{res}	3.7 μH, ETD54-3C90 core, 1mm gap N_t=4×8400×AWG44
C_{block}	1.9 μF, 8× 0.22 μF, ceramic NP0
Transformer	N=7.5, RM14-3C95 core, ungapped; N_{t1}=4×600×AWG40, N_{t2} = 32× 80×AWG40
Full Bridge MOSFET	ST STB160N75F3; $R_{ds,on}$=3.7mΩ Q_g=85 nC at V_{gs}=10V, C_{oss}=1080 pF V_{max}=75V I_{max}=120A 2 nF extra C_{ds} on lagging leg
Cycloconverter MOSFET	Infineon IPP60R250CP; $R_{ds,on}$=250mΩ Q_g=26 nC at V_{gs}=10V, C_{oss}=54 pF V_{max}=650V I_{max}=8A

Table 5.2: Final prototype parameters.



(a) Wide frequency range.

(b) Narrow frequency range.

Figure 5.6: Measured vs. fitted impedance of the resonant tank as seen from the secondary side of the transformer (i.e. looking 'through' the transformer from the secondary side). Fitted: (L_{res},C_{res},R_{par})=(220 μH,42 nF,2 Ω).

In balancing performance with safe and efficient operation of the inverter, the following safety factors were chosen: $SF_{CC,on} = 1.5$, $SF_{CC} = 2$, $SF_{FB} = 1$, full-bridge ZVS margin = 0.2 radians. A set of efficiency data was collected by running the inverter to deliver between one and ten percent more power than what is desired by each operating point. This deviation was varied based on the instantaneous power output. About eight output voltages, equally spaced in line phase, were tested along the quarter line cycle for each average power level. Figure 5.7 shows the accuracy in delivering the desired output power, ignoring the average power level. The necessity for controlling to deliver more power than what is necessary (indicated by curve 'Predicted/Desired') is higher at instantaneous power output of below 100W. Above this number, the prediction was much closer to actual inverter operation. For powers below 20W and above 250W, one to three attempts had to be made to achieve the presented power delivery accuracy. In almost all other cases, the first attempt was sufficient.

In analyzing the source of error of this model, Figure 5.8 shows the accuracy in tracking the input power as well as accuracy of the loss estimate based on the single parasitic resistor[5]. As accuracy of this loss model increases (which occurs at lower frequencies/higher powers), the input power matches well with the prediction. At lowers power and thus higher frequencies, the loss estimate is significantly off and the input power is not predicted with nearly the same accuracy. Part of the deviation is that the constant parasitic resistor model is not accurate across frequency. The value which was used was fitted to experimental data for powers of above 100W, as conduction loss is more significant there. As power decreases, this effective resistance would increase (as the efficiency and the RMS current both drop, the equivalent resistance will increase), which would improve the accuracy of the loss model and

[5]Although it was determined that the populated board ESR is about 2Ω, a value of 2.9Ω was used in the control model as it provided a better 'fixed resistor' match to some previously collected data.

thus the power tracking accuracy. Note that the plot of the impedance of the inverter will be virtually unaffected by small changes in the parasitic resistance.

Figure 5.10a shows a plot of efficiency of the inverter as a function of nominal output power. At the very low powers efficiency drops off due to high frequency. Fig 5.10b shows a measure of efficiency which ignores the power used for gating the devices. Since this is the dominant loss mechanism at high frequency, the efficiency is substantially different. Assuming that the inverter's efficiency is 0 when delivering 0W in every average power case and approximating the efficiency of the 10% average power case to be the same as the 20% case, the CEC averaged efficiency is about 95.1%. When gating losses are ignored, this number rises to 96.2%. Figure 5.9 shows a plot of the predicted total gating loss and the value measured to be sourced by the 'gating power supply'. The prediction accounts for only about 60% of the gating loss. Fig. 5.11 shows a typical set of waveforms in the inverter. The time intervals from the current zero crossing to the full-bridge and cycloconverter rises is close to minimal.

5.3 Results II

The first inverter design was different from the one presented above. Part of the reason for this, is that the optimization and control schemes evolved along with the design, and not necessarily before the initial prototype. The main difference comes from the resonant tank sizing. The original design had the resonant tank on the secondary side. The inductor was a RM14-3C95 core gapped to 3.8 mm and the inductance was adjusted with the number of turns (generally on the order of 40). The original transformer N was equal to 8. Figures 5.12 show the effect of changing of the resonant components on the efficiency. Note that this data ignores the gating loss, which was not measured at the time, so the net efficiency is even lower. The

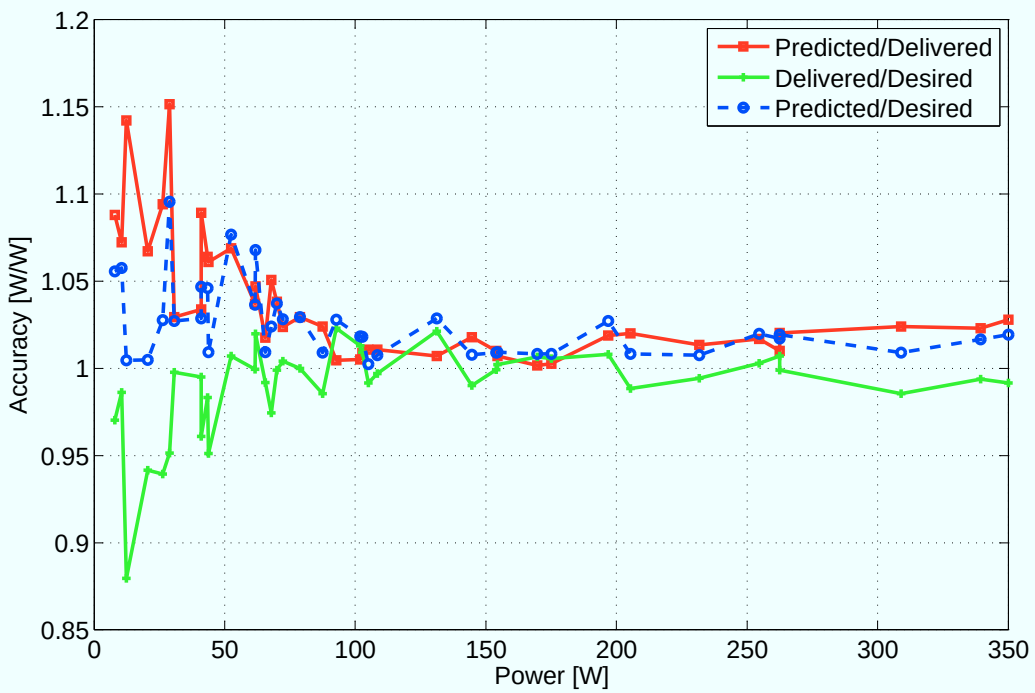
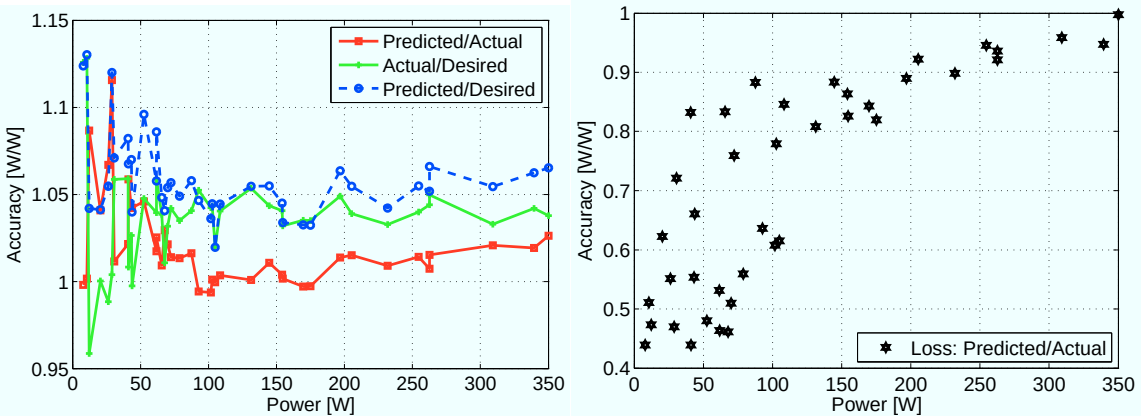


Figure 5.7: Accuracy in power delivery.



(a) Accuracy in input power.

(b) Accuracy in loss.

Figure 5.8: Comparing input power accuracy.

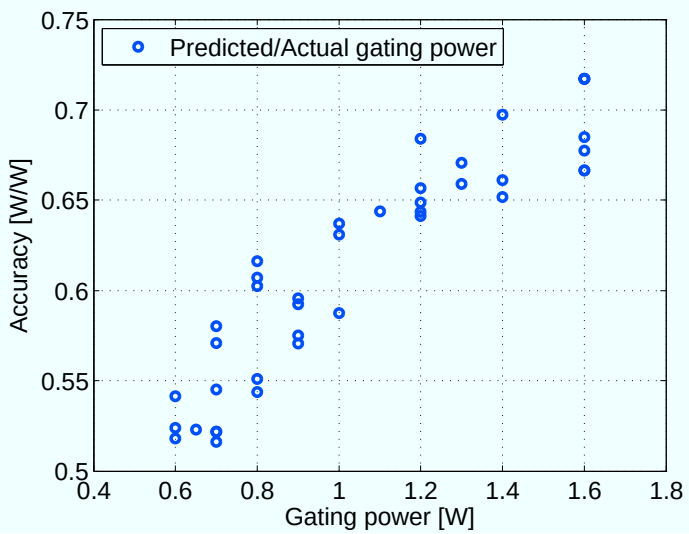
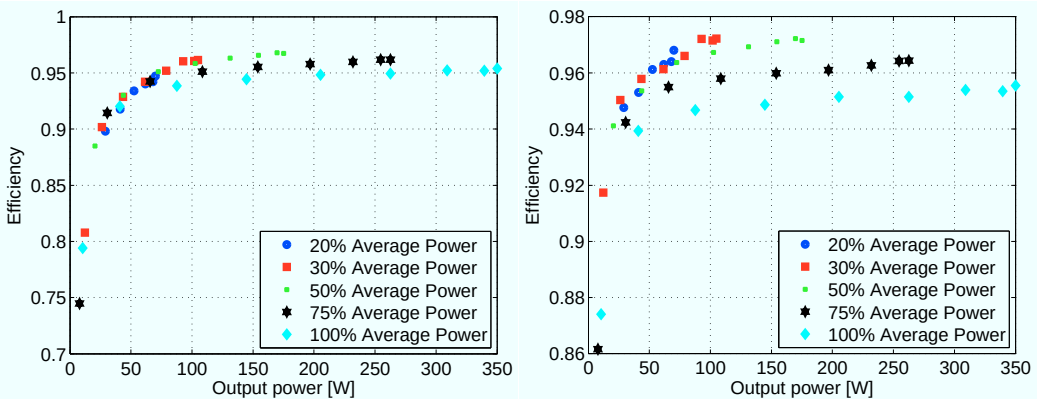


Figure 5.9: Comparing predicted to measured gating loss.

efficiency across the entire range of power increases when the resonant components increase in size. The case of higher input voltage is less efficient than the low input. All of this is due to the changes in switching frequency, and is consistent with the theory of chapter 4. Figure 5.13 shows the effect of decreasing the transform turns ratio from 8 to 7.5. Again, as predicted, this has a positive impact on efficiency, due to both lower frequency and lower current on the primary.



(a) Overall efficiency. CEC average - 95.1% (b) Ignoring gating loss. CEC average - 96.2%

Figure 5.10: Efficiency for the final prototype.

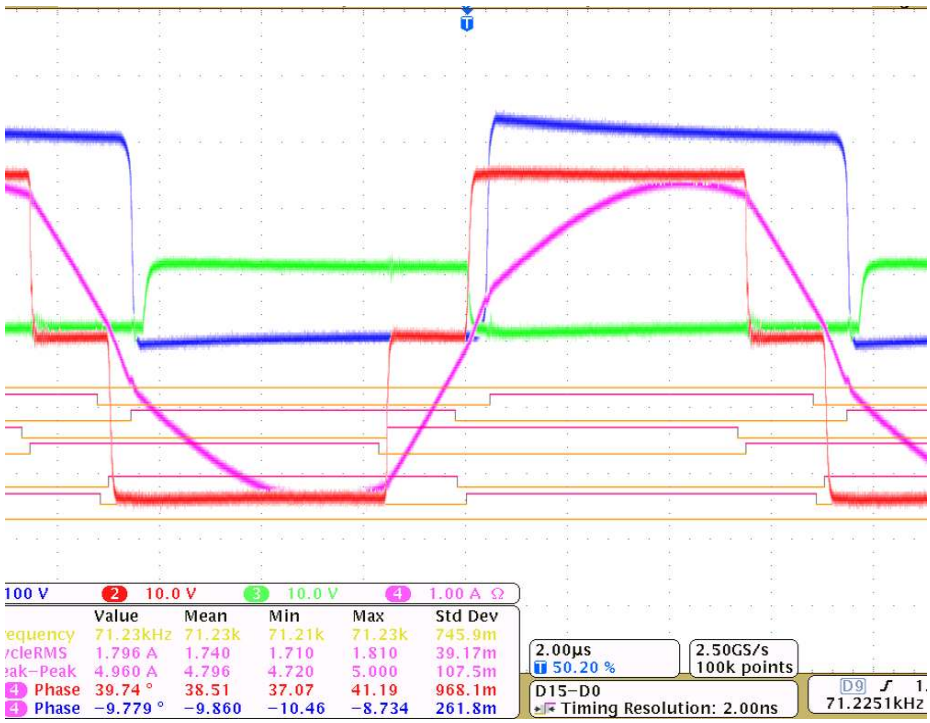
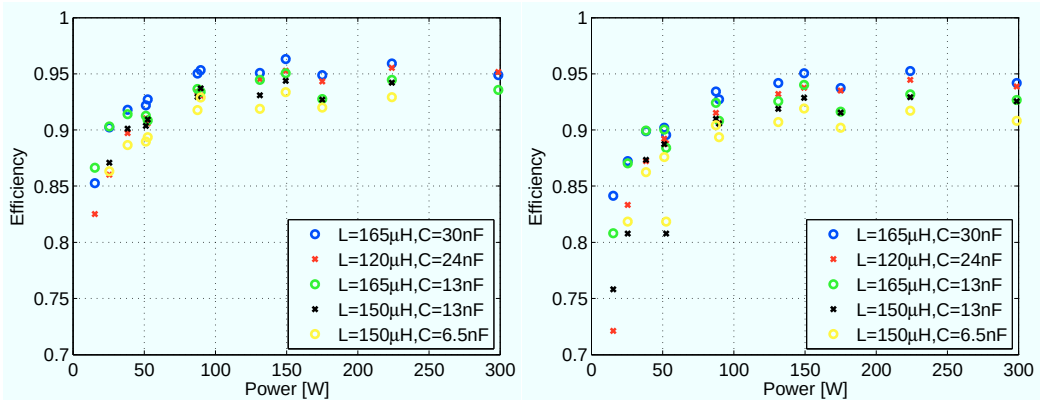
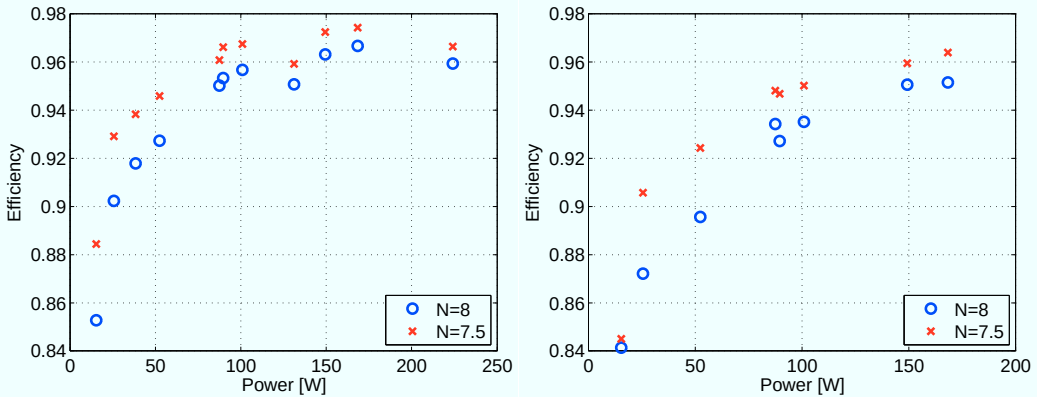


Figure 5.11: Typical inverter waveforms. 250W is outputted from 25V in to 305V. $-v_{CC}$ is in blue, i_x is in purple, v_{FB} is in red, v_{gs,F_H} is in green, The non flat top of blue is an artifact of the differential probe used to take the measurement.



(a) V_{IN}=25V. (b) V_{IN}=34V.

Figure 5.12: Changes in efficiency due to resonant tank size.



(a) V_{IN}=25V. (b) V_{IN}=34V.

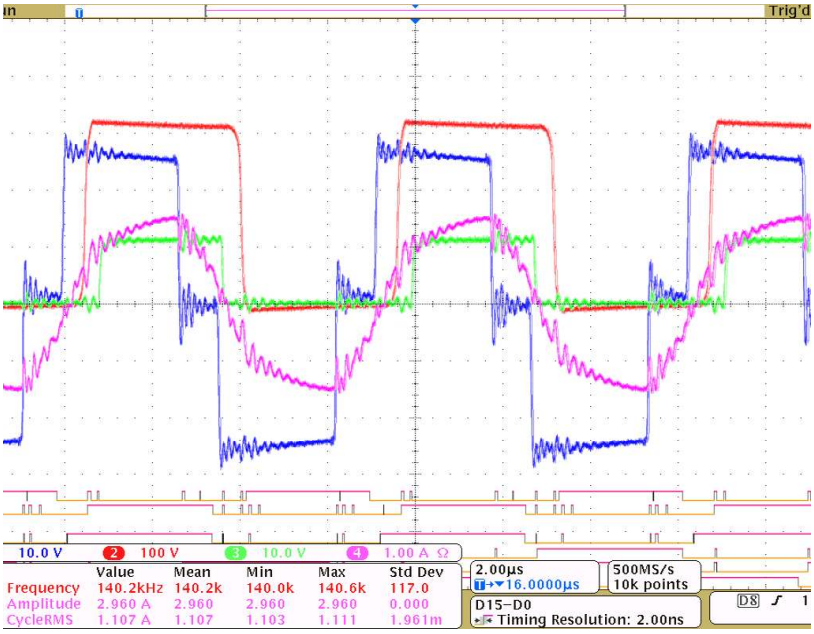
Figure 5.13: Changes in efficiency due to transformer turns ratio. For this test, L_{res}=165 μH, C_{res}=32nF.

5.3.1 Resonant tank placement

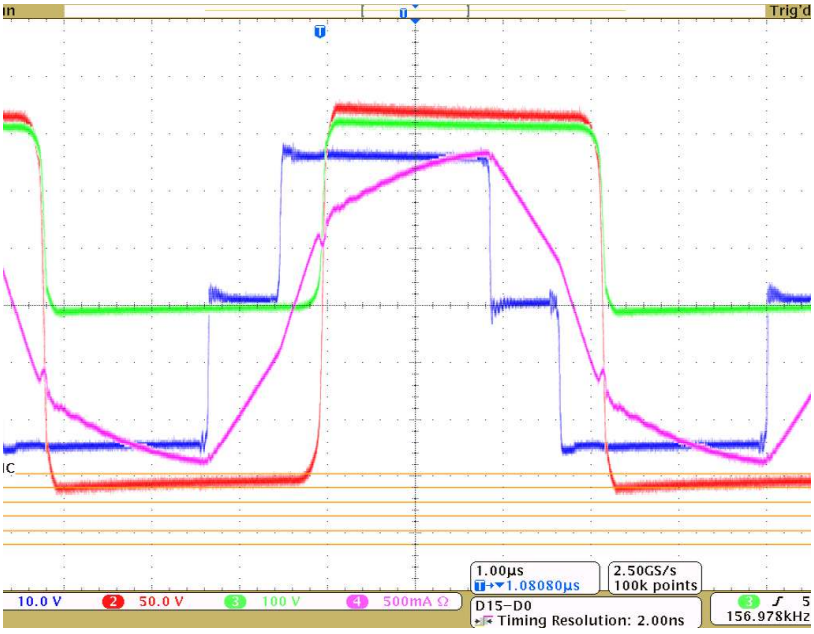
The parasitic capacitance of the transformer was measured to be about 12.5 pF across the secondary. This is partly responsible for the ringing in the waveforms, which contributes to loss and could cause failure of components. When the resonant tank is placed on the primary, this capacitance becomes absorbed into the cycloconverter parasitic (and yet it is small enough to have negligible effect on timing). The switch to the primary side inductor resulted in much cleaner waveforms. Figure. 5.14 shows a side by side comparison of two nearly identical operating points for the different resonant tank positions. The change was observed using both an ETD and an RM core inductor.

5.4 Loss estimates using thermal measurements

Any power loss in the inverter is dissipated as heating of the circuit components. One way to estimate the losses in the inverter is by measuring the difference in temperature



(a) Secondary side.



(b) Primary side.

Figure 5.14: Effect of resonant tank placement on the waveforms.

of its components between the off state and the steady state of operation. The general relationship between the column vector of power losses in each component of interest and the column vector of those components' temperature rises is given by equations (5.4) in terms of the thermal impedance matrix. This matrix couples power loss in any component to temperature rise in any other component (including itself). It has units of $°C/W$ and has to be derived experimentally by, for example, dissipating DC power in every component individually and measuring temperature rise in all components.

$$\Delta T = R_{TH} P_{\text{steady state}} \tag{5.4a}$$

$$P_{\text{steady state}} = R_{TH}^{-1} \Delta T \tag{5.4b}$$

This thermal characterization of the inverter was performed using K-Type thermocouples, which were attached using either solder or thermally conductive glue. The details (final thermal impedance matrix and the thermal characterization data) are found in Appendix E. Measured loss distribution data for several operating points is presented on Fig. 5.15. This data was taken for the RM14 inductor of 165 μH, a transformer N of 7.5 and a resonant capacitance of 32 nF. The inverter was allowed to run for 80-120 minutes to reach steady state before taking the temperature readings. The figure also presents the predicted losses for those operating points. All analytical estimates of underpredict the actual power loss. Although the source is this error is still unclear, scaling all of the loss models by some factor to account for inaccuracy reduces the maximum achievable efficiency prediction, presented in chapter 4. Note that the unaccounted power was found to be repeatable - the case of 28.8% unaccounted power was tested again to yield 27.6%.

	Vin/Vout/Pout [V]/[V]/[W]	Losses [W]						
		Total	Full-bridge	Cyclo-converter	Inductor	Transfor-mer	Conduc-tion	Unac-counted
Predicted	25/240/87	3.15	1.10	0.40	0.24	0.64	-	-
Measured	"	4.54	1.29	0.70	1.01	0.75	0.33	0.46
Predicted	34/313/150	4.27	1.26	0.54	0.39	0.92	-	-
Measured	"	7.02	2.06	1.23	1.94	0.97	0.50	0.33
Predicted	34/240/87	3.17	1.13	0.40	0.25	0.61	-	-
Measured	"	5.83	0.83	1.63	0.83	1.53	0.67	0.34
Predicted	25/313/150	3.75	1.09	0.54	0.31	0.62	-	-
Measured	"	4.94	1.32	0.81	1.25	1.03	0.48	0.04
Predicted	25/129/25	2.16	1.17	0.25	0.14	0.32	-	-
Measured	"	3.29	1.15	0.30	0.44	0.31	0.14	0.95

(a) Loss distribution, watts.

	Vin/Vout/Pout [V]/[V]/[W]	Losses [% of total]						
		Total	Full-bridge	Cyclo-converter	Inductor	Transfor-mer	Conduc-tion	Unac-counted
Predicted	25/240/87	100	34.9	12.7	7.6	20.3	-	-
Measured	"	100	28.5	15.3	22.3	16.6	7.2	10.1
Predicted	34/313/150	100	29.5	12.6	9.1	21.5	-	-
Measured	"	100	29.3	17.5	27.6	13.9	7.1	4.6
Predicted	34/240/87	100	35.6	12.6	7.9	19.2	-	-
Measured	"	100	14.3	27.9	14.3	26.3	11.5	5.8
Predicted	25/313/150	100	29.1	14.4	8.3	16.5	-	-
Measured	"	100	26.8	16.5	25.2	20.8	9.8	0.9
Predicted	25/129/25	100	54.2	11.6	6.5	14.8	-	-
Measured	"	100	35.0	9.2	13.3	9.4	4.2	28.8

(b) Loss distribution, percent.

Figure 5.15: Loss distribution for several operating points.

Chapter 6

Conclusion

6.1 Summary

This thesis presents the development of a microinverter for single-phase PV applications that is suitable for conversion from low-voltage (25-40V) DC to high voltage AC (e.g. $240V_{AC,RMS}$). The topology is based on a full-bridge series resonant inverter, a high-frequency transformer, and a novel half-wave cycloconverter. The operational characteristics are analyzed, and a multidimensional control technique is utilized to achieve high efficiency, encompassing frequency control, inverter phase shift control and cycloconverter phase shift control. An experimental prototype is demonstrated in DC/DC conversion mode for a wide range of output voltages. The proposed control strategy is shown to allow for accurate power delivery with minimal steps taken towards correction. The prototype achieves a CEC averaged efficiency of approximately 95.1%. Guidelines for optimization are presented along with experimental results which validate the method. The work concludes by pointing out that the original loss estimates are based on models which significantly differ from experimental measurements. As result, the projected efficiency is not achieved. However, the trends in designing and controlling for low loss remain qualitatively valid.

6.2 Next steps

Besides increasing efficiency, there are at least two logical next steps in continuing the development of the inverter. The first is to match the inverter with a true AC voltage source at the output. In theory, DC/AC conversion is a simple extension of the DC/DC case over a range of output voltages. Since DC/DC conversion was demonstrated with power accuracy maintained through a look up table, sensing the output voltage and controlling for the equivalent instantaneous DC operating condition will produce AC conversion. Higher switching frequency will improve the AC tracking accuracy via a pre-computed table. The distinguishable AC voltage variation limited is given by (6.1). For a given switching frequency it is not possible to track a variation in voltage above this limit.

$$\Delta v_{out} = \left\lfloor V_{out,MAX} \times \frac{f_{SW}}{f_{line}} \right\rfloor \tag{6.1}$$

When the output voltage crosses zero, the transition between the modulating cycloconverter legs may occur by keeping both low side devices conducting and the high side devices off. After the output polarity changes, the high side of the non-modulating leg is turned on and switching begins on both devices of the other leg.

The second step is to settle on an optimal size/frequency combination. The demonstrated prototype's parameters were chosen for higher efficiency rather than smaller size (both the resonant components and other passives), which may not necessarily be the best choice for a real product. Additionally, no dynamic response consideration were taken into account in selecting the frequency of the prototype.

6.3 Efficiency

More work needs to be done on tracing the losses and/or explaining the deviations between the loss models and experimental results. Accounting for various ringing and higher order harmonics as well as imperfections in control timing may bring the two closer, although this would be a hard value to estimate. It is perhaps more useful to look for immediate areas of improvement in efficiency. Several possible directions are now presented.

The efficiency may benefit from a different layout and selection of different devices. It was measured that the primary side loop ESR accounts for up to 9% of the loss at the higher current/power levels. It was possible to reduce this loss by almost one half by soldering conductive copper tape in parallel with the traces of the full bridge. Utilizing a different package (in a different layout) allows for use of the MDmesh MOSFET family by STMicroelectronics for the cycloconverter. These devices provide lower output capacitance and lower gate charge (while at slightly higher on-state resistances), and are projected to give higher efficiency (due to lower current) than what was used in the prototype.

It was observed, in chapter 5, that the semiconductor gating loss exceeds the projected value, based on the gate charge limit, by more than 50%. This deviation is caused partly by the inefficiencies in the discrete parts which were used to supply and regulate power to the gate driver. Since ignoring gating loss altogether increases the CEC efficiency by about 1.1% there is potential for increase in efficiency via more efficient gate drive circuitry. Relating to that, an approach which would likely be most effective in balancing minimum margin for ZVS while avoiding hard switching situations would implement a gate driver that turns on a MOSFET when the drain to source voltage is detected to be below a certain threshold. Such a gate driver can rely on pre-computed

deadtime to turn off while turning on at the most appropriate time, which may reduce losses in due to mistiming and certainly guarantee soft switching everywhere.

Although not explicitly presented in chapter 5, the tradeoff in the MOSFET on state gate to source voltage V_{gs} has been explored. In general the equivalent gate charge of a MOSFET increases with V_{gs}, both of which cause increase in gating loss. The benefit of using a higher V_{gs} is that the devices switch faster, which may improve the efficiency in the circuit overall. It has been found that the overall efficiency improves when using 10V instead of 12V for V_{gs} of the tested devices. There was evidence to suggest that operating at even lower V_{gs} will result in further gains in efficiency. However, a limit of about 9.6V was set by one of the discrete components used in the gate drive circuit. Switching to devices which are designed for a V_{gs} of 5V (and thus designing the gate drive circuit for that voltage) is expected to yield higher efficiency.

Finally, burst mode control may need to be explored for the very low powers, where the minimal current and the minimal frequency modes of operation are approximately equally lossy.

Appendix A

Converter models: MATLAB

A.1 Summary

The simplified inverter models and the complete control input selection scripts of chapter 4 made use of the scripts found in this appendix. All scripts are written in MATLAB, version 7.8.0 (R2009a) (which is the standard for all MATLAB scripts in the rest of this document). The diode rectifier inverter models, with or without the parasitic capacitance, is simulated using the script 'simpleModel.m'. The complete inverter model, which can be used to calculate a comprehensive control input table, is given by 'controlMain.m'. Both of these files rely on subroutines 'findTheta12Cds.m' and 'getVICds.m'. The four files are included below. All of these files execute as presented without errors, when placed in the same folder.

simpleModel.m

```
%this script is used to compare the RLC diode recitifier with or with-
%out parasitic capacitance across the diodes to LTSpice. Set all input
%parameters matching to Spice to get a true comparison
%If comparison is to be made with the FFT in LTSpice and the Fourier
%coefficiencts of this model, note that LTSpice divides the amplitude
%by sqrt(2)
%The angle reference is the fundamental of the full bridge waveform,
%which is always take to be of phase 0.

%%%% START: INPUTS%%%%%%%%%
%make sure the input and output voltages, the FB pulse width and the
%frequency match the Spice model to which the comparison is made
Vinput =25*7.5; %Vin
Vx = 305.6;     %Vout
f = 71.3e3;     %Frequency, switching
%resonant tank
Cres = 41.855e-9;
Lres = 219.9375e-6;
Rpar = 56.25*(52e-3)+1e-6;
RparStr = '0.2';
```

```
pdelta = 1;     %the base spice mode has a square wave input source,
                %making delta fraction = 1
%**********
%to switch between the two models of the diode rectifier
%set Cds = 0 for the no parasitic capacitance case,
%and Cds = some value > 0 for parasitice capacitance present case
Cds = .4339e-9;
%******************
%number of odd harmonics to consider
Nharm = [50];
%resolution in the zero crossing angle, radians
tstep = 0.001;
includeAll = 1; %set to 1 include all INPUT voltage harmonics (that is
                %if the source to be modeled is 1 ideal sinusoid). For
                %ex, to model a square or a 3 level wave set this to
                %one to model a single frequency source,set includeAll
                %to 0 and set begin to 1
                %the reason for this option is to allow to swap volta-
                %ge sources in Spice and quickly compare to this model
begin = 2;      %if includeAll = 0, this will be the first odd harmonic
                %ignored (2 = ignore 3rd and up, 3 = ignore 5th and up)

timeIt = 0;         %set to 1 to time the fsolve method
plotPower = 1;      %Plot power transfer as function of frequency

%%%% END: INPUTS%%%%%%%%%

w = 2*pi*f;
delta = pdelta*pi/2;
ThetaArray = (-pi:tstep:pi)';

fprintf(['Lres(uH)-%i, Cres(nF)-%i, Rpar-%s, Vinput-%i, f(kHz)-',...
    '.1f, delta-%.2f, Vout-%.3f, Cds(nF)-%.2f \n'],Lres*1e6,...
    Cres*1e9,RparStr,Vinput/TR, f/1e3,pdelta, Vx, Cds*1e9);

clear Vin
%For each number of harmonicin the Nharm vector:
for nh = 1:length(Nharm)

    Vin = zeros(1,Nharm(nh)); %preallocate the mem
    n = 1:1:Nharm(nh);
    H=2*n - 1;

    %solve for every needed harmonic of Vin
    Vin(n) = 4./H/pi.*Vinput.*sin(H.*delta);

    if(~includeAll) %if certain input harmonics are to be excluded
        Vin(begin:end) = 0;
    end
```

```
    options=optimset('Display', 'off');

    Q = Cds*Vx;  %output charge

    %solve for the angle at which charge Q is delivered after the cur-
    %rent zero crossing (which may be 0  for Q =0)
    fts = @(theta1)findTheta12Cds(theta1,w,Vin,Vx,Cres,Lres,H,Rpar,...
        ThetaArray,Q);
    [thetaSol, fval, exitflag] = fsolve(fts,-.6,options);

    if(timeIt)
        t2 = toc;
        disp(['Time for fsolve:', num2str(t2)])
    end

    %Given the solution, find the Fourier coefficients for the current
    %and the Vcc waveform
    [Vxs Izs thetaZeroCross thetaQ] = getVICds(thetaSol,Vin,Vx,w,...
        Cres,Lres,H,Rpar,ThetaArray,Q);
    %Compute the power in adn out
    Pout = sum(abs(Vxs).*abs(Izs).*cos(angle(Vxs)-angle(Izs))/2);
    Pin = sum(abs(Vin).*abs(Izs).*cos(angle(Vin)-angle(Izs))/2);

    %display the summary of results
    if(Nharm(nh)>3)
        fprintf(['%i, %.2f, %.2f, %.4f, %.4f, %.3f, %.2f, %.3f,'...
            ' %.3f,%.3f \n'],Nharm(nh), Pin, Pout,thetaZeroCross,...
            thetaSol,angle(Izs(1)),abs(Izs(1:4)./Izs(1)));
    elseif(Nharm(nh)>2)
        fprintf(['%i, %.2f, %.2f, %.4f, %.4f, %.3f, %.2f, %.3f,',...
            '.3f \n'], Nharm(nh), Pin, Pout,thetaZeroCross,...
            thetaSol,angle(Izs(1)),abs(Izs(1:3)./Izs(1)));
    elseif(Nharm(nh)>1)
        fprintf('%i, %.2f, %.2f, %.4f, %.4f, %.3f, %.2f, %.3f \n',...
            Nharm(nh), Pin, Pout,thetaZeroCross,thetaSol,...
            angle(Izs(1)),abs(Izs(1:2)./Izs(1)));
    else
        fprintf('%i , %.2f , %.2f , %.4f,  %.3f, \n',Nharm(nh),...
            Pin,Pout,thetaSol,angle(Izs(1)));
    end
end

%plot power transfer through each harmonic
if(plotPower)
    in = abs(Vin).*abs(Izs).*cos(angle(Vin)-angle(Izs))/2;
    out = abs(Vxs).*abs(Izs).*cos(angle(Vxs)-angle(Izs))/2;
    p3 = figure;
    semilogy(H, abs(out), 'o-b', 'LineWidth',2, 'MarkerSize',6);
    grid on;
    set(gca, 'FontSize',14)
```

```
    xlabel('Odd harmonic number');
    ylabel('Power, W');
end
```

controlMain.m

```
%this script steps in frequency, pulse width and specified input
%and output voltages and computes valid minimum current operating
%points that arise due to such timing  the results are stored in
%table opptsRe. Table opptsIm stores the harmonic content of ix
%and Vcc for the given number of harmonics. All circuit equations
%are set up as viewed from the SECONDARY of the transformer.
%The center of the full bridge pulse is the zero reference for all
%phase angles in this script
%Format of opptsRe (see code for description of value FF,
%typically=0.5): Vinput; Vout; PowerIn; PowerOut; FB pulse width
%fraction; Frequency; angle of Vcc rise to FF*Vout, angle current
%zero crossing; %phase angle needed to commutate FB device; Angle
%of margin for zvs on the FB; deadtime on FB ns; FF*deadtime CC.
%ns; FB rise time, radians; CC high side minimal turn on tim as
%fraction of Tsw (this value cant be used directly yet)
%for speed, the script only works for up to 5th harmonic

% profile on      %use profiler to see where the bottlenecks are
tic                     %start the timer

%%%%%%%%%%%%            INPUTS           %%%%%%%%%%%%%%%%%%%%%%%%
Cres = 41.855e-9;       %resonant capacitor, as seen from 2ndary
                        %has to account for Cblock in series
Lres = 219.9375e-6;     %resonant inductor, as seen from 2ndary
TR = 7.5;               %transformer turns ratio
Vinput = [25 32.5 40];  %input voltages to sweep

%phase angles in the AC quarter line cycle, in radians
%this is equivalent to setting the output voltages to sweep
VoutAngles = (70:10:90)./90*pi/2;
Nharm = 3;              %Number of odd harmonics to consider. In
                        %this case for Nharm=3: fundmtl, 3rd, 5th.
Vrms = 240;             %AC line RMS voltage
Pmax = 350;             %Max instantaneous output power
tstep = 0.01;           %radians; step in angle -determines the
                        %resolution of the zero crossing angle
                        %solution DEFAULT: 0.01
FF = 0.5;               %The fraction of total full-bridge charge
                        %that needs to commutate in order to say
                        %that the drain to source voltages became
```

```
                        %inverted.In chapter 4 this is given as 0.5
FPGAclk = 50e6;         %Frequency of the FPGA clock, HZ.
                        %Determines the frequency resolution

%the next 3 parameters determine the general range of output
%powers. the higher the max freq and the lower the min delta, the
% smaller output power is possible
fmax = 400e3;               %the maximum frequency of operation,Hz
deltamin = 0.6;             %the min FB pulse width
deltamax = 0.99;            %the max FB pulse width.
                            %FB step = 0.01

%Some parasitic resistance 45e-3 is obtained as an experimental fit
Rpar = 1*(2*0.00375+45e-3)*(TR^2)+1e-6;
RparStr = num2str(Rpar);    %convert to string to display later

%set the numeric solver preferences. TolFun=tolerance [of the
%zero crossing angle in radians]
options=optimset('Display', 'off','TolFun', 1e-3);

%%%%%%%%%%%    MOSFET DATA      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Data for Infineon IPP60R250CP taken from various points on the
%Coss vs Vds plot in the datasheet. This is the cycloconverter
%(CC) FET.
VdsCC = [1 10 37 50 75 100 125 150 175 200 250 300 350];
CdsCC = 1e-12*[1e4 3e3 1e3 2e2 80 58 50 48 47 45 43 41 40];
%Calculate the effective capacitance (see chapter 4)
CeffCC = 1./VdsCC.*cumtrapz(VdsCC,CdsCC);

%Data for ST STB160N75taken from various points on the Coss vs
%Vds plot in the datasheet. This is the full-bridge(FB) FET. The
%datasheet waveform is much smoother and thus a half-analytic fit
%is given
Vds = (0.1:0.1:50)';
n = find(Vds <= 5);
Coss(n) = 25000-Vds(n)*(25000-7500)/5;
m = find(Vds > 5 & Vds <= 10);
Coss(m) = 7.5e3+(Vds(m)-5)*(-7500+3750)/5;
k = Vds>10 & Vds <= 20;
Coss(k) = 3.75e3+(Vds(k)-10)*(-3750+1250)/10;
k = find(Vds>20);
Coss(k) = 1080;
Coss = Coss*1e-12;

%calculate the equivalent charge and output capacitance for FB
Qoss = cumtrapz(Vds,Coss)';
CeffFB = Qoss./Vds;
%%%%%%%%%%%  END  MOSFET DATA      %%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%  END  INPUTS      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%number of columns in the results array
Ncol = 10;
%preallocate the results vectors for speed
%~40 kPts per 1 output voltage level per 1 input voltage level
opptsRe = zeros(250000,Ncol+4);     %this will be the main
                                    %results array

%this will store the complex ix and vcc for each harmonic number
opptsIm = zeros(250000,2*Nharm)-1i*ones(250000,2*Nharm);

%make the odd harmonic index
n = 1:1:Nharm;
H=2*n - 1;

%generate the angle vectors to pass to the solver. This is faster
%than generating them in the solver every time it is called
ThetaArray = (-pi:tstep:pi)';
ThetaArray2 = (-pi:tstep:0.2)';
clear Vin

%Find the output voltages
Vline = sin(VoutAngles).*Vrms*sqrt(2);
pl = length(VoutAngles);
LV = length(Vinput);

%preallocate the mem
Vin = zeros(1,Nharm);
Vi = 0;
Qcc = 0;
Qfb = 0;
Vout = 0;
zvs = 0;
Pout = 0;
Pin = 0;

%calculate the lowest resonant frequency
Fres = 1./2/pi./sqrt(Lres.*Cres);
cnt0 = fix(FPGAclk./Fres);
cntMin = fix(FPGAclk./fmax);

%row counter for the results array
row = 1;
%step in input voltage
for vi = 1:LV
   Vi = Vinput(vi);
   %find the full bridge charge given an input voltage
   Qfb = Vi*interp1(Vds,CeffFB,Vi)/TR;
   %step in output voltage
   for np = 1:pl
        Vout = Vline(np);
```

```
        %find the equivalent CC charge based on output voltage
        Qcc = FF*2*interp1(VdsCC,CeffCC,Vout)*Vout;

%calculate the maximum amount of power that will need to be
%delivered at this voltage (ie at max average power). This is to
%save time and not calculate output conditions which will never be
%run. Factor of 1.1 because it tends to underpredict at certain
%low powerd
        Poutmax =  1.1*sin(VoutAngles(np))^2*Pmax;

        %step in frequency, starting at resonance
%       for f = Fres:.5e3:fend;  %may use any size frequency step
         for cnt=cnt0:-1:cntMin    %or use the FPGA clock
            f = FPGAclk./cnt;
            w = 2*pi*f;

            %reset the flag for too much power delivery
            tooMuchPowerAlready = 0;

            %then step in fraction of FB pulse width
            for pdelta = deltamin:0.01:deltamax
                delta = pdelta*pi/2;

            %find harmonics of Voltage across transformer secondary
                Vin(n) = 4./H/pi.*Vi*TR.*sin(H.*delta);

                %solve for the angle at which CC rises to FF*Vout
                fts = @(theta1)findTheta12Cds(theta1,w,Vin,...
                    Vout,Cres,Lres,H,Rpar,ThetaArray,Qcc);
                [thetaSol, fval, exitflag] = fsolve(fts,-.6,options);
                %thetaSol is the solution angle

                if(exitflag<=-1)
                    continue;   %if the solver failed
                end

        %given that thetaSol, find the current and CC voltage har-
        %monics (Vxs, Izs respecetively), the angle for zero
        %crossing of the current - iTheta, and the angle necessary
        %to commutate the FB current - thetaQ.
        [Vxs Izs iTheta thetaQ] = getVICds(thetaSol,Vin,Vout,w,...
                    Cres,Lres,H,Rpar,ThetaArray2,Qfb);

                %compute the resulting ZVS margin
                zvs = iTheta+delta;

                %see if the margin is at least the minimal phase
                %angle which still supports ZVS
                if(zvs < thetaQ)
                    %not enough ZVS margin so skip the rest
```

```
                    continue;
                end

                %calculate the expected output power
                Pout=sum(abs(Vxs).*abs(Izs).*cos(angle(Vxs)-...
                    angle(Izs))/2);

                if(Pout > Poutmax)
                    %if delivering more power than would ever be
                    %delivered at this voltage, stop increasing
                    %the pulse width and continue increasing
                    %frequency
                    tooMuchPowerAlready = 1;
                    break;
                else
                    %calculate the input power
                    Pin=sum(abs(Vin).*abs(Izs).*cos(angle(Vin)...
                        - angle(Izs))/2);

                    %Note that:
                    %thetaQ ~ FB deadtime in radians

                    %store the operating point in the tables
                    opptsRe(row,1:Ncol) = [Vinput(vi), Vout, ...
                        Pin, Pout,pdelta, f, thetaSol,...
                         iTheta, thetaQ, zvs];
                    opptsIm(row, 1:6) = [Vxs(1:3),  Izs(1:3)];
                    row = row+1;
                end
            end
        end
    end
end
toc
%deadtime FB. ns
opptsRe(:,Ncol+1) = fix(1e9*opptsRe(:,9)./opptsRe(:,6)./2./pi);
%FF*deadtime CC. ns. To arrive at deadtime, multiply by 1/FF
opptsRe(:,Ncol+2) = fix(1e9*(opptsRe(:,7)-opptsRe(:,8))...
    ./opptsRe(:,6)./2./pi);

%FB rise time, radians
opptsRe(:,Ncol+3) = -opptsRe(:,5)*pi/2;
%CC on, fraction of cycle
opptsRe(:,Ncol+4) = (opptsRe(:,7)-opptsRe(:,Ncol+3))./2./pi;

%since it tale a while to generate all this, save it!
save opptsReal.mat opptsRe
save opptsImag.mat opptsIm
```

getVICds.m

```
function [Vxn I thetaZero thetaQ] = getVICds(theta1, Vinput, Vx, w,...
    Cres, Lres, H, Rpar,theta, Qds)
%Given a consistent set of input (see description of each below) this
%function returns the results complex amplitude for current and
%cycloconverter voltage for each harmonic (Vxn and I respectively), as
%well as the current zero crossing angle (thetaZero) and the angle
% (thetaQ) during which a charge of Qds is delivered, starting at the
% zero crossing of current, while rising. This is similar to
% findTheta12Cds.m. The major difference is that findTheta12Cds is
% configured to be used by fsolve and thus must returns some measure
% that is desirable to bring to zero (like error in phase). Once the
% consistent solution has been found it is easy to repeat the procedure
% used in findTheta12Cds to solve for the other quantities of interest.
% Additionally, the main control script uses this function to calculate
% the minimum phases needed for commutation of the FB, which involves
% a different Q than the Cycloconv. Output/Inputs:
% Vxn = array of Vcc harmonics (complex)
% I = array of I harmonics (complex)
% thetaZero = zero crossing angle of I
% thetaQ = angle (absolute difference) from the zero crossing of I,
% during which it deliver charge Qds
% theta1 - the assumed angle at which Vcc rises to some portion of Vout
% w = 2*pi*fsw (switching frequency)
% Vinput - input voltage (for the real inverter this would be the
% voltage across transformer secondary.
% Vx = Vout (DC)
%Cres, Lres, Rpar = resonant tank capacitor, inductor and parasitic
%resistance
%H = index of harmonics. ex: [1,3,5]
%theta = vector with the pregenerated phase angle axis (to save time
%by nothaving to generate it in this function). literally -pi:0.01:pi;
%Qds- the charge of interest. Could be zero. Can not be negative


%create the differential voltage across the resonant tank, all as
%complex amplitudes at each harmonic.
Vxn = 2*Vx./pi./H.*(-sin(H*theta1) - 1i*cos(H*theta1));
%solve for complex current
Y = 1./(1./(1i*H*w*Cres)+1i*H.*w*Lres+Rpar);
I = (Vinput-Vxn).*Y;
%find the current peak and phase at each harmonic
Ipk = abs(I);
th = angle(I);

%assumming only 2 harmonics (1st and 3rd)
```

```
itotal = Ipk(1).*cos(theta+th(1))+Ipk(2).*cos(3*theta+th(2));
%and possibly 5th
if(H(end)==5)
   itotal = itotal+Ipk(3).*cos(5*theta+th(3));
end

%find the current zero crossing angle (of the index of it)
iZero = find(itotal>0);
n1 = iZero(1);

%start integrating current to give chare deliverd
qtotal = 0*itotal;  %itotal arrives blank
qtotal(n1:end) = cumtrapz(theta(n1:end), itotal(n1:end))/w;

%find the index at which more charge than necessary is delivered
%note that this is symmetric accross the current zero crossing - the
%negative charge before it is the same as the positive charge after it
%for the same time interbal. This allows the procedure to work on the
%full bridge as well
qZero = find(qtotal>Qds);

%interpolate for the angle
if(n1 > 1)
    thetaZero = interp1(itotal(n1-1:n1),theta(n1-1:n1),0);
else
    thetaZero = -pi;    %the very beginning
end
if(isempty(qZero))
    thetaQ = 6; %some large number
else
    n2 = qZero(1);
    thetaQ =  interp1(qtotal(n2-1:n2),theta(n2-1:n2),Qds) - thetaZero;
end
```

---

findTheta12Cds.m

```
function thetaDiff = findTheta12Cds(theta1,w,Vinput,Vx,Cres,Lres,H,...
    Rpar,theta,Qoss)
%This function computes the 'error' angle, defined as the difference
%between theta1 (the assumed angle at which a charge Qoss is delivered
%after current zero crossing) and the actual angle at which this occurs
%for the given inputs. That is theta1 and Vx define Vcc (the 'load'
%harmonics) and when frequency and other inputs are added, it is
%possible to solve for the resonant current and the new angle theta1
%which is the interval of interest.
%w = 2*pi*fsw (switching frequency)
```

```
%Vinput - input voltage (for the real inverter this would be the
%voltageacross transformer secondary.
%Vx = Vout (DC)
%Cres, Lres, Rpar = resonant tank capacitor, inductor and parasitic
%resistance
%H = index of harmonics. ex: [1,3,5]
%theta = vector with the pregenerated phase angle axis (to save time
%by not having to generate it in this function). literally -pi:0.01:pi;
%Qoss - the charge of interest. Could be zero. Can not be negative

%iterate about theta1 (zero crossing of current, rising)
%only works for up to 5th harmonic

%create the differential voltage across the resonant tank, all as
%complex amplitudes at each harmonic.
Vxn = 2*Vx./pi./H.*(-sin(H*theta1) - 1i*cos(H*theta1));
%solve for complex current
I = (Vinput-Vxn)./(1./(1i*H*w*Cres)+1i*H.*w*Lres+Rpar);
%find the current peak and phase at each harmonic
Ipk = abs(I);
th = angle(I);


%this is where increase in speed comes in - an arbitrary length for H
%may be assumed but stepping thru them in a for loop is much slower
%that the current version (even for 3 harmonics).

%assumming only 2 harmonics (1st and 3rd)
itotal = Ipk(1).*cos(theta+th(1))+Ipk(2).*cos(3*theta+th(2));

%and possibly 5th
if(H(end)==5)
   itotal = itotal+Ipk(3).*cos(5*theta+th(3));
end

%find the index in the angle vector which corresponds to the angle at
%which the resonant current crosses zero
iZero = find(itotal>0);
n1 = iZero(1);

%find the charge delivered as a function of angle, with 0 corresponding
%to the zero crossing of current. Q = INT{i, dAngle}
qtotal = 0*itotal;
qtotal(n1:end) = cumtrapz(theta(n1:end), itotal(n1:end))/w;

%see when delivered charge exceeds the required charge
qZero = find(qtotal>Qoss);

if(isempty(qZero))
    thetaDiff = -1; %if could not find it
```

```
else
    n2 = qZero(1);
    if(n1 > 1)
        %if such an angle is found-interpolate for the exact value
        theta1new = interp1(qtotal(n2-1:n2),theta(n2-1:n2),Qoss);
    else
        theta1new = -pi;
    end
    %the goal is to converge this 'error' to 0
    thetaDiff = theta1-theta1new;
end
```

Appendix B

Converter models: LTspice

B.1 Summary

The first level of validation of the MATLAB scripts of appendix A was performed by comparing the output to circuit simulations of LTspice IV. The simplified model consisting of an RLC circuit with a diode rectifier load, with or without the parasitic capacitance, can be simulated with the LTspice listing found in 'RLCdiodeRectifier.asc'. This model may be used to verify the output of 'simpleModel.m' ((with results shown in chapter 4). The full inverter schematic, with the cycloconverter replaced by two diodes is given in 'fullConverterPrimaryTankDiodeRectifier.m'. This model includes MOSFETs of the full-fridge and an idealized transformer and can be used to verify 'simpleModel.m' (provided the voltage in the MATLAB model is scaled in accordance with the) and 'controlMain.m'. In order to achieve good agreement, it is important to enter the same value for drain to source capacitance into both: the Spice model and the MATLAB script. The full inverter Spice listing also allows to change the deadtime on the full-bridge devices and to simulate modulation of the cycloconverter high side FET (this is given a pulsed voltage source, unconnected to the rest of the inverter). This provides a way to measure how much ZVS margin there exists given the control inputs. Overall the full inverter model was found to be in good agreement with experiment, provided that the equivalent parasitic capacitance model of chapter 4 is used in the Spice simulation.

Both LTspice model files are shown below. The full inverter models requires four other files to run properly (they are shown right afterwards and need to be placed into the same folder). All of these listings are valid and will simulate if pasted into text files and saved with the given names and extensions.

RLCdiodeRectifier.asc

```
Version 4
```

```
SHEET 1 3220 4916
WIRE 528 640 384 640
WIRE 384 704 384 640
WIRE 528 784 528 640
WIRE -160 864 -336 864
WIRE -80 864 -160 864
WIRE 64 864 0 864
WIRE 96 864 64 864
WIRE 384 864 384 768
WIRE 384 864 160 864
WIRE 384 880 384 864
WIRE 448 880 384 880
WIRE -336 928 -336 864
WIRE 384 928 384 880
WIRE 448 928 448 880
WIRE -336 1056 -336 1008
WIRE -208 1056 -336 1056
WIRE 384 1056 384 992
WIRE 384 1056 -208 1056
WIRE 448 1056 448 992
WIRE 448 1056 384 1056
WIRE 528 1056 528 864
WIRE 528 1056 448 1056
FLAG -208 1056 0
FLAG 528 640 lp
FLAG 64 864 V1
FLAG 384 864 Vc
FLAG -160 864 Vin
SYMBOL bv 528 768 R0
SYMATTR InstName Bline
SYMATTR Value V={Vo}
SYMBOL ind -96 880 R270
WINDOW 0 32 56 VTop 0
WINDOW 3 5 56 VBottom 0
SYMATTR InstName L1
SYMATTR Value {Lres}
SYMATTR SpiceLine Rser={Rp}
SYMBOL cap 160 848 R90
WINDOW 0 0 32 VBottom 0
WINDOW 3 32 32 VTop 0
SYMATTR InstName C1
SYMATTR Value {Cres}
SYMBOL diode 400 768 R180
WINDOW 0 24 72 Left 0
WINDOW 3 24 0 Left 0
SYMATTR InstName D1
SYMATTR Value Dfet250
SYMBOL diode 400 992 R180
WINDOW 0 24 72 Left 0
WINDOW 3 24 0 Left 0
```

```
SYMATTR InstName D2
SYMATTR Value Dfet250
SYMBOL voltage -336 912 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
WINDOW 3 16 104 Left 0
SYMATTR Value PULSE({Vin} {-Vin} {0.25/fsw} 1n 1n {0.5/fsw} {1/fsw})
SYMATTR InstName V1
SYMBOL cap 432 928 R0
SYMATTR InstName C2
SYMATTR Value 0
TEXT -144 768 Left 0 !.tran 0 {50/fsw} {40/fsw} 1n uic
TEXT 664 1160 Left 0 !.param Lres {225u}\n.param Cres {42n}
TEXT 336 1072 Left 0 !.model Dfet250 D(Ron=1u Vfwd=0)
TEXT 296 1152 Left 0 !.param Vo 180\n.param Vin = {40*7.5}
TEXT 664 1248 Left 0 !.param Rp {0.1}
TEXT 624 1224 Left 0 ;Lres had this as its ESR
TEXT -168 1080 Left 0 ;7.5 - turns ratio (to be)
TEXT -400 1184 Left 0 !.meas tran zr find time when I(L1) = 0 cross=2
TEXT 296 1288 Left 0 !.param F 0.5
TEXT -344 1352 Left 0 !.meas tran Irms rms I(L1)
TEXT -336 1320 Left 0 ;Some RMS and Peak values
TEXT 80 1376 Left 0 ;Power In and Out
TEXT -392 1152 Left 0 ;some angles of interest
TEXT 80 1400 Left 0 !.meas tran Pin avg V(Vin)*I(V1)
TEXT 456 1016 Left 0 ;change this value to add or remove capacitance
TEXT 296 1328 Left 0 ;F - the fraction of Vo that Vc has to reach
TEXT 296 1352 Left 0 ;to measure the time span. Only important to
TEXT 776 1352 Left 0 ;compare to Matlab
TEXT 528 1128 Left 0 ;INPUTS
TEXT -288 968 Left 0 ;can also try different voltage waveform
TEXT 296 1224 Left 0 !.param fsw =170k
TEXT -400 1208 Left 0 !.meas tran hv find time when V(Vc)={F*Vo} cross=2
TEXT -400 1232 Left 0 !.meas tran qa param {(hv/tsw-1)*2*pi}
TEXT -400 1256 Left 0 !.meas tran ia param {(zr/tsw-1)*2*pi}
TEXT 80 1424 Left 0 !.meas tran Pout avg V(lp)*I(Bline)
TEXT -344 1400 Left 0 !.meas tran Ipp pp I(L1)
TEXT -344 1448 Left 0 !.meas tran VinAve avg V(vin)
TEXT -344 1496 Left 0 !.meas tran VccAve avg V(Vc)
TEXT -344 1376 Left 0 !.meas tran Iave avg I(L1)
TEXT -344 1424 Left 0 !.meas tran VinRms rms V(vin)
TEXT -344 1472 Left 0 !.meas tran VccRms rms V(Vc)
TEXT 296 1256 Left 0 !.param tsw={1/fsw}
TEXT 600 832 Left 0 ;Output Voltage
TEXT -720 1184 Left 0 ;current zero crossing time
TEXT -720 1208 Left 0 ;time at which vc rose to f*V0
TEXT -704 1232 Left 0 ;convert the above two
TEXT -680 1256 Left 0 ;quantities to angles
RECTANGLE Normal 992 1376 288 1104
```

fullConverterPrimaryTankDiodeRectifier.asc

```
Version 4
SHEET 1 3220 4916
WIRE -1248 544 -1664 544
WIRE -816 544 -1248 544
WIRE -1248 592 -1248 544
WIRE -816 592 -816 544
WIRE 48 624 -16 624
WIRE 176 624 48 624
WIRE -1392 640 -1424 640
WIRE -1296 640 -1312 640
WIRE -960 640 -992 640
WIRE -864 640 -880 640
WIRE -16 656 -16 624
WIRE 48 656 48 624
WIRE -1424 688 -1424 640
WIRE -992 688 -992 640
WIRE 176 688 176 624
WIRE -1472 704 -1488 704
WIRE -1040 704 -1056 704
WIRE -1472 752 -1488 752
WIRE -1040 752 -1056 752
WIRE -16 768 -16 720
WIRE 48 768 48 720
WIRE 48 768 -16 768
WIRE -1424 784 -1424 768
WIRE -1248 784 -1248 688
WIRE -1248 784 -1424 784
WIRE -992 784 -992 768
WIRE -816 784 -816 688
WIRE -816 784 -992 784
WIRE 176 832 176 688
WIRE -1664 864 -1664 544
WIRE -1248 864 -1248 784
WIRE -720 864 -1248 864
WIRE -640 864 -656 864
WIRE -608 864 -640 864
WIRE -400 864 -528 864
WIRE -272 864 -304 864
WIRE -160 864 -272 864
WIRE -16 864 -16 768
WIRE -16 864 -96 864
WIRE -16 912 -16 864
WIRE 48 912 -16 912
WIRE -816 928 -816 784
WIRE -400 928 -816 928
```

```
WIRE -144 928 -304 928
WIRE -16 928 -16 912
WIRE 48 944 48 912
WIRE -1248 992 -1248 864
WIRE -816 992 -816 928
WIRE -16 1024 -16 992
WIRE 48 1024 48 1008
WIRE 48 1024 -16 1024
WIRE -1392 1040 -1424 1040
WIRE -1296 1040 -1312 1040
WIRE -960 1040 -992 1040
WIRE -864 1040 -880 1040
WIRE -144 1072 -144 928
WIRE -16 1072 -16 1024
WIRE -16 1072 -144 1072
WIRE 176 1072 176 912
WIRE 176 1072 -16 1072
WIRE -1424 1088 -1424 1040
WIRE -992 1088 -992 1040
WIRE -1472 1104 -1488 1104
WIRE -1040 1104 -1056 1104
WIRE -1472 1152 -1488 1152
WIRE -1040 1152 -1056 1152
WIRE -1424 1184 -1424 1168
WIRE -1248 1184 -1248 1088
WIRE -1248 1184 -1424 1184
WIRE -992 1184 -992 1168
WIRE -816 1184 -816 1088
WIRE -816 1184 -992 1184
WIRE -1664 1232 -1664 944
WIRE -1248 1232 -1248 1184
WIRE -1248 1232 -1664 1232
WIRE -816 1232 -816 1184
WIRE -816 1232 -1248 1232
WIRE -1664 1296 -1664 1232
FLAG -1056 704 OUTHINEG
IOPIN -1056 704 In
FLAG -1056 752 GND
IOPIN -1056 752 In
FLAG -1056 1104 OUTLONEG
IOPIN -1056 1104 In
FLAG -1056 1152 GND
IOPIN -1056 1152 In
FLAG -1488 704 OUTHIPOS
IOPIN -1488 704 In
FLAG -1488 752 GND
IOPIN -1488 752 In
FLAG -1488 1104 OUTLOPOS
IOPIN -1488 1104 In
FLAG -1488 1152 GND
```

```
IOPIN -1488 1152 In
FLAG -1664 1296 0
FLAG -1664 544 nvin
FLAG -816 928 bn
FLAG -1248 864 bp
FLAG 176 688 p
FLAG 176 1072 n
FLAG -1200 1792 0
FLAG -1088 1728 0
FLAG -960 1680 0
FLAG -816 1632 0
FLAG -1088 1648 OUTLOPOS
FLAG -1200 1712 OUTHIPOS
FLAG -960 1600 OUTHINEG
FLAG -816 1552 OUTLONEG
FLAG -272 864 V2
FLAG -16 864 Vc
FLAG -640 864 v1
FLAG -1392 1856 0
FLAG -1392 1776 VccGS
SYMBOL res -976 1056 R270
WINDOW 0 32 56 VTop 0
WINDOW 3 0 56 VBottom 0
SYMATTR InstName R7
SYMATTR Value 0.05
SYMBOL res -976 656 R270
WINDOW 0 32 56 VTop 0
WINDOW 3 0 56 VBottom 0
SYMATTR InstName R8
SYMATTR Value 0.05
SYMBOL res -1408 1056 R270
WINDOW 0 32 56 VTop 0
WINDOW 3 0 56 VBottom 0
SYMATTR InstName R10
SYMATTR Value 0.05
SYMBOL res -1408 656 R270
WINDOW 0 32 56 VTop 0
WINDOW 3 0 56 VBottom 0
SYMATTR InstName R11
SYMATTR Value 0.05
SYMBOL e -992 672 R0
SYMATTR InstName E8
SYMATTR Value 10
SYMBOL e -992 1072 R0
SYMATTR InstName E10
SYMATTR Value 10
SYMBOL e -1424 672 R0
SYMATTR InstName E11
SYMATTR Value 10
SYMBOL e -1424 1072 R0
```

```
SYMATTR InstName E12
SYMATTR Value 10
SYMBOL voltage -1664 848 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName Vpanel
SYMATTR Value {Vin}
SYMBOL ideal-transformer -352 896 R0
WINDOW 39 -20 71 Left 0
SYMATTR SpiceLine n=7.5
SYMATTR InstName X2
SYMBOL cap -160 880 R270
WINDOW 0 32 32 VTop 0
WINDOW 3 0 32 VBottom 0
SYMATTR InstName bk
SYMATTR Value {Cb}
SYMBOL powerfet -1296 592 R0
WINDOW 39 100 59 Left 0
SYMATTR SpiceLine Rds = {rdson} cds0 = {cdsfb}
SYMATTR InstName X3
SYMBOL powerfet -1296 992 R0
WINDOW 39 100 59 Left 0
SYMATTR SpiceLine Rds = {rdson} cds0 = {cdsfb}
SYMATTR InstName X9
SYMBOL powerfet -864 592 R0
WINDOW 39 100 59 Left 0
SYMATTR SpiceLine Rds = {rdson} cds0 = {cdsfb}
SYMATTR InstName X10
SYMBOL powerfet -864 992 R0
WINDOW 39 100 59 Left 0
SYMATTR SpiceLine Rds = {rdson} cds0 = {cdsfb}
SYMATTR InstName X11
SYMBOL voltage -1200 1696 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName V1
SYMATTR Value PULSE(10 0 {OTD-delta/2} {rt} {rt} {OTD+TdDH+TdDL} {tsw})
SYMBOL voltage -1088 1632 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName V2
SYMATTR Value PULSE(0 10 {OTD-delta/2+TdDH} {rt} {rt} {OTD} {tsw})
SYMBOL voltage -960 1584 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName V3
SYMATTR Value PULSE(0 10 {delta/2} {rt} {rt} {OTG} {tsw})
SYMBOL voltage -816 1536 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
```

```
SYMATTR InstName V4
SYMATTR Value PULSE(10 0 {delta/2-TdGL} {rt} {rt} {OTG+TdGL+TdGH} {tsw})
SYMBOL ind -624 880 R270
WINDOW 0 32 56 VTop 0
WINDOW 3 5 62 VBottom 0
SYMATTR InstName L1
SYMATTR Value {Lres}
SYMATTR SpiceLine Rser={Rp}
SYMBOL cap -656 848 R90
WINDOW 0 0 32 VBottom 0
WINDOW 3 32 32 VTop 0
SYMATTR InstName C1
SYMATTR Value {Cres}
SYMBOL diode 0 720 R180
WINDOW 0 24 72 Left 0
WINDOW 3 24 0 Left 0
SYMATTR InstName D1
SYMATTR Value Dfet250
SYMBOL diode 0 992 R180
WINDOW 0 24 72 Left 0
WINDOW 3 24 0 Left 0
SYMATTR InstName D2
SYMATTR Value Dfet250
SYMBOL cap 32 944 R0
SYMATTR InstName C2
SYMATTR Value {cds}
SYMBOL cap 32 656 R0
SYMATTR InstName C3
SYMATTR Value {cds}
SYMBOL voltage 176 816 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName V5
SYMATTR Value {Vo}
SYMBOL voltage -1392 1760 R0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName V6
SYMATTR Value PULSE(0 10 {CCon*tsw} {rt} {rt} {tsw/2-dtCC} {tsw})
TEXT -752 720 Left 0 !.tran 0 {50/fsw} {40/fsw} 1n uic
TEXT -384 528 Left 0 !.param wsw {2*pi*fsw}\n.param tsw {1/fsw}
TEXT -1608 1320 Left 0 !.param TdDH = {dtlead}\n.param TdDL = {dtlead}
TEXT -384 608 Left 0 !.param Lres {3.91u}\n.param Cres {2.4u}
TEXT -1280 1328 Left 0 ;leading leg = D
TEXT -488 808 Left 0 !.model Dfet250 D(Ron=1u Vfwd=0)
TEXT -656 1208 Left 0 !.param Vin = 32.5\n.param fsw =101.63k
TEXT -320 1160 Left 0 !.meas tran zcrR find time when I(L1)=0 cross=2
TEXT -1296 1384 Left 0 ;lagging leg = G
TEXT 216 1432 Left 0 ;Power In and Out
TEXT 200 1464 Left 0 !.meas tran Pin avg V(nvin)*I(Vpanel)
```

```
TEXT -680 1576 Left 0 ;CCon - fraction of tsw at which CC high
TEXT -680 1600 Left 0 ;Used to check to full matlab model
TEXT 208 1584 Left 0 !.meas tran Irms rms I(L1)
TEXT 208 1560 Left 0 ;RMS and Ave quantities
TEXT -592 1168 Left 0 ;INPUTS
TEXT -472 592 Left 0 ;INPUTS
TEXT -1608 1376 Left 0 !.param TdGH = {dtlag}
TEXT -1616 1464 Left 0 !.param OTD ={.5*(tsw-TdDH-TdDL)}
TEXT -1616 1496 Left 0 !.param OTG = {.5*(tsw-TdGH-TdGL)}
TEXT 208 1608 Left 0 !.meas tran Iave avg I(L1)
TEXT 208 1632 Left 0 !.meas tran Ipp pp I(L1)
TEXT 208 1656 Left 0 !.meas tran VinRms rms V(bp,bn)
TEXT 208 1680 Left 0 !.meas tran VinAve avg V(bp,bn)
TEXT 208 1704 Left 0 !.meas tran VccRms rms V(Vc,n)
TEXT 208 1728 Left 0 !.meas tran VccAve avg V(Vc,n)
TEXT 200 1488 Left 0 !.meas tran Pout avg V(p,n)*I(V5)
TEXT -320 1184 Left 0 !.meas tran y find time when V(Vc,n)={Vf} cross=2
TEXT -320 1208 Left 0 !.meas tran z find time when V(Vc,n)={Vo} cross=2
TEXT -320 1232 Left 0 !.meas tran qAF param {y/tsw - 1)*2*pi}
TEXT -320 1256 Left 0 !.meas tran qvAF param {(z/tsw - 1)*2*pi}
TEXT -320 1280 Left 0 !.meas tran icAngle param {(zcrR/tsw-1)*2*pi}
TEXT -656 1264 Left 0 !.param PW = .56\n.param CCon = .079
TEXT -656 1320 Left 0 !.param Vo= 218.2
TEXT -656 1376 Left 0 !.param dtCC = 544n\n.param dtlead = {dtlag}
TEXT -656 1432 Left 0 !.param cds = .61n\n.param cdsfb =4.4n
TEXT -656 1488 Left 0 !.param rdson = 3.75m\n.param rt = 10n
TEXT -368 664 Left 0 !.param Cb {2.2u}\n.param Rp {45m}
TEXT -384 1376 Left 0 ;deadtime Cyclo
TEXT -384 1352 Left 0 ;deadtime fb
TEXT 232 1160 Left 0 ;current zero crossing time
TEXT -240 1576 Left 0 ;side would turn on
TEXT -296 1600 Left 0 ;and does nothing in this circuit
TEXT 232 1184 Left 0 ;time when Vcc rises to ff*vout
TEXT 232 1208 Left 0 ;time when Vcc rises to vout
TEXT 232 1232 Left 0 ;convert the above three to angle
TEXT -1608 1400 Left 0 !.param TdGL = {dtlag}
TEXT -1608 1432 Left 0 !.param delta={PW*.5*tsw}
TEXT -128 1112 Left 0 ;fraction of Vout marking the rise
TEXT -656 1352 Left 0 !.param dtlag = 150n
TEXT -312 1112 Left 0 !.param F = 0.5
TEXT -320 1136 Left 0 !.param Vf = {F*Vo}
TEXT 232 1136 Left 0 ;the fraction of Vout of interest
RECTANGLE Normal -160 720 -384 576
RECTANGLE Normal -400 1552 -672 1184
```

ideal-transformer.asc

```
Version 4
SHEET 1 880 680
WIRE -80 80 -112 80
WIRE 16 80 -80 80
WIRE 16 96 16 80
WIRE 144 96 16 96
WIRE 304 96 224 96
WIRE 432 96 384 96
WIRE -80 128 -80 80
WIRE 16 128 16 96
WIRE 224 128 224 96
WIRE 144 144 144 96
WIRE 176 144 144 144
WIRE 176 192 144 192
WIRE 16 240 16 208
WIRE 144 240 144 192
WIRE 144 240 16 240
WIRE -80 256 -80 208
WIRE -80 256 -112 256
WIRE 16 256 16 240
WIRE 16 256 -80 256
WIRE 224 256 224 208
WIRE 432 256 224 256
WIRE 16 320 16 256
WIRE 80 320 16 320
WIRE 224 320 224 256
WIRE 224 320 160 320
FLAG -112 80 v1p
IOPIN -112 80 BiDir
FLAG -112 256 v1n
IOPIN -112 256 BiDir
FLAG 432 96 v2p
IOPIN 432 96 BiDir
FLAG 432 256 v2n
IOPIN 432 256 BiDir
SYMBOL e 224 112 R0
SYMATTR InstName E1
SYMATTR Value {n}
SYMBOL f 16 128 R0
WINDOW 39 24 116 Left 0
SYMATTR SpiceLine {n}
SYMATTR InstName F1
SYMATTR Value Vsense
SYMBOL res 176 304 R90
WINDOW 0 0 56 VBottom 0
WINDOW 3 32 56 VTop 0
SYMATTR InstName R1
SYMATTR Value 1Meg
SYMBOL res -96 112 R0
SYMATTR InstName R2
```

```
SYMATTR Value 1Meg
SYMBOL voltage 288 96 R270
WINDOW 0 -32 56 VBottom 0
WINDOW 3 32 56 VTop 0
WINDOW 123 0 0 Left 0
WINDOW 39 0 0 Left 0
SYMATTR InstName Vsense
SYMATTR Value 0
TEXT 96 56 Left 0 !.param n 1
```

ideal-transformer.asy

```
Version 4
SymbolType BLOCK
LINE Normal -4 28 -4 -28
LINE Normal 4 28 4 -28
LINE Normal -24 32 -24 26
LINE Normal -48 32 -24 32
LINE Normal 24 -32 24 -26
LINE Normal 48 -32 24 -32
LINE Normal -24 -32 -24 -26
LINE Normal -48 -32 -24 -32
LINE Normal 24 32 24 26
LINE Normal 48 32 24 32
CIRCLE Normal -7 -30 -12 -35
CIRCLE Normal 12 -30 7 -35
ARC Normal 10 -28 26 -12 24 -26 25 -15
ARC Normal 10 -18 26 -2 24 -16 25 -5
ARC Normal 10 -8 26 8 24 -6 25 5
ARC Normal 10 2 26 18 24 4 25 15
ARC Normal 10 12 26 28 24 14 25 25
ARC Normal -10 28 -26 12 -24 26 -25 15
ARC Normal -10 18 -26 2 -24 16 -25 5
ARC Normal -10 8 -26 -8 -24 6 -25 -5
ARC Normal -10 -2 -26 -18 -24 -4 -25 -15
ARC Normal -10 -12 -26 -28 -24 -14 -25 -25
TEXT 1 42 Center 0 1:n
PIN -48 -32 NONE 8
PINATTR PinName v1p
PINATTR SpiceOrder 1
PIN -48 32 NONE 8
PINATTR PinName v1n
PINATTR SpiceOrder 2
PIN 48 -32 NONE 8
PINATTR PinName v2p
PINATTR SpiceOrder 3
```

```
PIN 48 32 NONE 8
PINATTR PinName v2n
PINATTR SpiceOrder 4
```

POWERFET.asc

```
Version 4
SHEET 1 2276 812
WIRE 272 -16 272 -160
WIRE 384 -16 272 -16
WIRE 496 -16 384 -16
WIRE 496 112 496 -16
WIRE 272 128 272 -16
WIRE -48 144 -64 144
WIRE 224 144 -48 144
WIRE 384 160 384 -16
WIRE 224 192 208 192
WIRE 208 304 208 192
WIRE 272 304 272 208
WIRE 272 304 208 304
WIRE 384 304 384 224
WIRE 384 304 272 304
WIRE 496 304 496 176
WIRE 496 304 384 304
WIRE 272 448 272 304
FLAG 272 -160 D
IOPIN 272 -160 BiDir
FLAG 272 448 S
IOPIN 272 448 BiDir
FLAG -64 144 G
IOPIN -64 144 BiDir
FLAG -48 144 G
SYMBOL sw 272 224 M180
SYMATTR InstName S1
SYMATTR Value MyFET
SYMBOL diode 512 176 R180
WINDOW 0 24 72 Left 0
WINDOW 3 24 0 Left 0
SYMATTR InstName D1
SYMATTR Value DFW
SYMBOL cap 368 160 R0
SYMATTR InstName C1
SYMATTR Value {cds0}
TEXT 64 536 Left 0 ;.param cgs0 1000p
TEXT 64 768 Left 0 !.model MyFET SW(Ron={Rds} Roff=1Meg Vt=2 Vh=-1.5)
TEXT 64 736 Left 0 !.model DFW D(Vfwd=0.2 Ron=1n Vrev=1000)
```

```
TEXT 64 560 Left 0 ;.param cdg0 100p
TEXT 64 592 Left 0 ;.param cds0 5000p\n.param Rds 0
```

---

powerfet.asy

```
Version 4
SymbolType BLOCK
LINE Normal 48 48 48 96
LINE Normal 16 80 48 80
LINE Normal 40 48 48 48
LINE Normal 16 48 40 44
LINE Normal 16 48 40 52
LINE Normal 40 44 40 52
LINE Normal 16 8 16 24
LINE Normal 16 40 16 56
LINE Normal 16 72 16 88
LINE Normal 0 48 8 48
LINE Normal 8 16 8 80
LINE Normal 48 16 16 16
LINE Normal 48 0 48 16
LINE Normal 70 16 48 16
LINE Normal 70 80 48 80
LINE Normal 61 56 70 42
LINE Normal 79 56 70 42
LINE Normal 61 56 79 56
LINE Normal 70 56 61 56
LINE Normal 70 80 70 56
LINE Normal 70 16 70 42
LINE Normal 61 42 70 42
LINE Normal 79 42 61 42
WINDOW 0 100 31 Left 0
PIN 48 0 NONE 0
PINATTR PinName D
PINATTR SpiceOrder 1
PIN 0 48 NONE 0
PINATTR PinName G
PINATTR SpiceOrder 2
PIN 48 96 NONE 0
PINATTR PinName S
PINATTR SpiceOrder 3
```

Appendix C

Optimization scripts

C.1 Summary

The optimization discussion of chapter 4 made use of the MATLAB scripts found in this appendix. The file 'optimize.m' was used to size the resonant components and develop the three dimensional visualization of that trade-off. It makes calls to 'findL.m' and 'suboptG.m'. The latter calculates the approximate (using the fundamental-only analysis) control inputs for a wide range of operating points (spanning the entire CEC test space) and calculates the approximate losses and CEC average efficiency. That function may also be run as a standalone script (by commenting out the function header; see the comments in that file). As a script, it can be used to perform MOSFET selection studies (similar to that shown in chapter 4) and to analyze the various aspects of the minimum current control method and its effect on switching frequency and the resonant current (the function includes a convenient graphical representation of most quantities of interest). In all cases, 'loss4.m' is used to estimate losses given the component selection and operating parameters of the inverter. It makes calls to 'CCloss.m', 'FBloss.m', 'Lloss.m', 'Xloss.m' to estimate losses in each component. Finally, 'suboptS.m' can be used to estimate the control inputs when the cycloconverter phase shift is allowed to vary beyond the minimal. It explores a combination of minimal current and minimal frequency control methods to achieve the most efficient operating condition. In combination with 'suboptG.m', these can be used to study the current sloshing boundary, described at the end of chapter 4.

Note that, unlike 'controlMain.m' of Appendix A, 'suboptG.m' does not calculate a table of all operating conditions, but rather finds the control inputs which are consistent with certain (desired) operating points. The reason for this comes from the enormous difference in execution time of the two. While the latter looks to satisfy just one operating point, it may reject other valid solutions as it performs a partial grid search. It is fast enough (on the order of seconds) to allow for a complete restart of the grid search for every new operating point. Alternatively, 'controlMain.m' takes several hours to compute the same set of operating points. It is therefore more efficient not

to restart grid search at all and the full range of valid control inputs is performed first, before picking out the specific operating points of interest. Additionally, using this as a lookup table for feedback considerations makes this technique very appropriate.

All of these scripts are executable, as shown, without errors, if all are placed in the same folder.

optimize.m

```
%This script produces a 3D plot of approximate CEC efficiency as a
%function of resonant inductance and capacitance. This can be used to
%pick the optimal resonant tank values (those which produce the
%highest CEC efficiency).
%the scripts makes calls to sup_optG.m, which must be configured to
%run in function mode (see that file for details), and to findL.m

tic;    %start timing the script (for interest)

%generate the range of inductances and capacitance over which to test
[tryL, tryC] = meshgrid((80:10:400)*1e-6, (10:1:90)*1e-9); %[H], [F]

%clear out the vairable that will be used
clear j cec ave_freq Qunl;
%preallocate for memory              %will store:
cec = -1*ones(size(tryL));           %CEC efficiency
ave_freq = -1*ones(size(tryL));      %average frequency
aveIpk = -1*ones(size(tryL));        %average peak resonant current
Qunl = -1*ones(size(tryL));          %characteristic impedance
                                     %of the tank

%some typical Al 3f3 rm14 inductor values (nH/turn^2)
[Al]=[87 101 126 166 200 263 315 400 518 700];

%approximate vector of peak currents for the IPP60R250CP devices at
%frequencies corresponding to Fvec. These were found using sub_optG.m
Ipks = [3.2 2.4 1.7 1.1 .8].';

%Input voltage
Vin = 32.5;
%Turns ratio
TR = 7.5;

%Some large blocking cap
Cblock = 200e-6/TR/TR;

%make sure to change sub_opt into a function
```

```
for l = 1:length(tryL(1,:))
    for c = 1:length(tryC(:,1))
       %generate the vector of frequencies, a certain position away
       %from resonance. For this range of qualify factors,
       Fvec = [1.5, 1.7, 2.3, 3.8, 5.4  ].'*1./2/pi*1./sqrt(...
           tryC(c,l)*tryL(c,l));

       %find the 'best' inductor configuration for this inductance
       %(usually the lowest Al value wins)
       [Nturns,Nstrands,Nbundles,ml,awg,Al2,Loss,Lcond,Lcore,...
           klayer,wirediam, wirearea]=findL(Al,tryL(1,l),Fvec,Ipks,0);
       %find the CEC efficiencies and average frequencies and currents
       [cec(c,l),ave_freq(c,l),aveIpk(c,l)]=sub_optG(Vin,tryL(c,l),...
           tryC(c,l),Cblock,3,Nturns,Nstrands,ml,Al2,wirediam,...
           wirearea,klayer,TR,0,1,1);

       %Find the characteristic impedance of the tank
       Qunl(c,l) = sqrt(tryL(c,l)./tryC(c,l));

       %if saturated or unable to compute the efficiency, it will
       %return -1 so insert nan everywhere at that location
       if(cec(c,l) < 0)
           cec(c,l) = nan;
           ave_freq(c,l) = nan;
           aveIpk = nan;
           Qunl = nan;
       end
    end
end
toc

%Produce all of the figures
figure;
surf(tryL*1e6,tryC*1e9,cec)
set(gca, 'FontSize',16)
xlabel('L [\muH]')
ylabel('C [nF]')
zlabel('CEC efficiency')
% zlim([.96 .978]) %1.01*max(max(cec))
title(['Vin = ',num2str(Vin),' N = ',num2str(TR)])
fcec2 = figure;
surf(tryL*1e6,tryC*1e9,ave_freq/1e3)
set(gca, 'FontSize',16)
xlabel('L [\muH]')
ylabel('C [nF]')
zlabel('Average frequency, kHz')
title(['Vin = ',num2str(Vin),' TR = ',num2str(TR)])
ceceq = figure;
surf(tryL*1e6,tryC*1e9,Qunl)
set(gca, 'FontSize',16)
```

```
xlabel('L [\mu H]')
ylabel('C [nF]')
zlabel('(L/C)^{0.5} [Ohm]')
title(['Vin = ',num2str(Vin),' TR = ',num2str(TR)])
figure;
surf(tryL*1e6,tryC*1e9,aveIpk)
set(gca, 'FontSize',14)
xlabel('L [\mu H]')
ylabel('C [nF]')
title(['Vin = ',num2str(Vin),' TR = ',num2str(TR)])
```

findL.m

```
function [Nturns,Nstrands,Nbundles,ml,awg,Al2,Loss,Lcond,Lcore,...
    klayer,wirediam,wirearea]=findL(Al,L,Fvec,Ipks,plots)
%find the optimal inductor core and winding given the list of avail-
%able Al values, frequencies and current of interest. Assumer 3f3 Rm14
%core and litz wire of AWG 40, 42, 44. Set plots = 1, to plot the
%analysis, set it to 0 otherwise.
%the most optimal parameters are returned.

%calculate the necessary number of turns for each Al value
N = sqrt(L./(Al*1e-9));

%some data on wires: AWG40, 42, 44
%the klayer (packing factors) were determined by actually winding the
%available wire into the core
wirediam40 = .0799*1e-3; %wire diameter, m awg = 40;
wirearea40 = 0.00501*1e-6; %m^2
klayer40 = 0.25;
DCesr40 = 0;

wirediam42 = 0.064e-3;
wirearea42 = 0.00322e-6;
klayer42 = 0.35;
DCesr42 = 0;

wirediam44 = 0.051e-3; %mm AWG44
wirearea44 = 0.00204e-6; %mm^2 AWG44
klayer44 = 0.45;
DCesr44 = 0;

%calculate the number of strands bundle and layers needed for each
%wire gauge for each number of turns
maxStrands40 = 5100./N;
bundles40 = fix(maxStrands40/40);
```

```
Nstrands40 = bundles40*40;
Nlayers40 = N/6;
maxStrands42 = 7900./N;
bundles42 = fix(maxStrands42/100);
Nstrands42 = bundles42*100;
Nlayers42 = N/8;
maxStrands44 = 10700./N;
bundles44 = fix(maxStrands44/27);
Nstrands44 = bundles44*27;
Nlayers44 = N/10;
Temperature = 27;
%find the loss for each configuration for each litz gauge
for n = 1:length(Al)
   [Ltotal40(n), Lwind40(n), Lcore40(n)]=Lloss(Fvec,Ipks,L,Al(n),...
       N(n),Nstrands40(n),Nlayers40(n),wirediam40,wirearea40,...
       klayer40,Temperature,DCesr40);
end
for n = 1:length(Al)
   [Ltotal42(n), Lwind42(n), Lcore42(n)]=Lloss(Fvec,Ipks,L,Al(n),...
       N(n),Nstrands42(n),Nlayers42(n),wirediam42,wirearea42,...
       klayer42,Temperature,DCesr42);
end
for n = 1:length(Al)
   [Ltotal44(n), Lwind44(n), Lcore44(n)]=Lloss(Fvec,Ipks,L,Al(n),...
       N(n),Nstrands44(n),Nlayers44(n),wirediam44,wirearea44,...
       klayer44,Temperature,DCesr44);
end
%find the minimal total loss
min44 = min(Ltotal44);
min42 = min(Ltotal42);
min40 = min(Ltotal40);
%find the configuration which gave that minimum loss
if(min44 < min40 && min44 < min42)
   awg = 44;
   m = min44;
   n = find(Ltotal44==m);
   Nturns = N(n);
   Nbundles = bundles44(n);
   Nstrands = Nstrands44(n);
   ml = Nlayers44(n);
   Loss = m;
   Lcond= Lwind44(n);
   Lcore = Lcore44(n);
   Al2 = Al(n);
   klayer = klayer44;
   wirediam = wirediam44;
   wirearea = wirearea44;
elseif(min42 < min40 && min42 < min44)
   awg = 42;
   m = min42;
```

```
   n = find(Ltotal42==m);
   Nturns = N(n);
   Nbundles = bundles42(n);
   Nstrands = Nstrands42(n);
   ml = Nlayers42(n);
   Loss = m;
   Lcond= Lwind42(n);
   Lcore = Lcore42(n);
   Al2 = Al(n);
   klayer = klayer42;
   wirediam = wirediam42;
   wirearea = wirearea42;
else
   awg = 40;
   m = min40;
   n = find(Ltotal40==m);
   Nturns = N(n);
   Nbundles = bundles40(n);
   Nstrands = Nstrands40(n);
   ml = Nlayers40(n);
   Loss = m;
   Lcond= Lwind40(n);
   Lcore = Lcore40(n);
   Al2 = Al(n);
   klayer = klayer40;
   wirediam = wirediam40;
   wirearea = wirearea40;
end

if(plots == 1)
    %if this option is selected plots the losses for all cases
    figure;
    plot(Al, Ltotal44, Al, Ltotal42, Al, Ltotal40);
    hold on;
    plot(Al, Lwind44, '--',Al, Lwind42, '--', Al, Lwind40, '--');
    plot(Al, Lcore44,'.-', Al, Lcore42, '.-', Al, Lcore40, '.-');
    hold off;
    legend('AWG 44 total','AWG 42 total','AWG 40 total',...
        'AWG 44 wind','AWG 42 wind','AWG 40 wind','AWG 44 core',...
        'AWG 42 core','AWG 40 core');
    xlabel('Al');
    ylabel('Loss');
    title(['L=',num2str(L*1e6),'\muH']);
end
```

suboptG.m

```
%this file may be run as either a script or a function. In script mode
%it calculates the approximate minimum current control inputs for spe-
%cified operating points and stores the results in array 'oppts'.
%It also estimates the losses in the inverter for each operating
%point and calculates the CEC efficiency in cecEff.
%In function mode, it makes the same computations but returns only the
%efficiency, and the average of the resonant current on the secondary
%and the switching frequency.
%Everything is found for [1 .75 .5 .3 .2] average power levels.
%The CEC efficiency estimate will give an error if more than 1 input
%voltage is specified and to average over several input voltages, the
%script has to be run for each case.(control inputs are still found).
%for script mode, inputs start at line 66
%The range of output voltages in set on line 63
%Uncomment line 20 and 21 if you want this to be a function (so that
% optimize.m call call it) and then set func = 1
%Line 421 presents an option to plot the results of this script (in
%the scipt mode). Line 353 shows formar of oppts.

function [cecEff, fave, Ipkave]=sub_optG(Vin,Lres,Cres,Cblock,Rpar,...
   N,Nstrands,ml,Al,wirediam,wirearea,klayer,TR,DCesr,func,lossCalc)
func = 1;

%calculate losses. typically leave this as 1
lossCalc = 1;
FET = 250;
clear pi i j
clear Coss
%Full Bridge MOSFET datasheet fit
VdsFB = (0.1:0.1:50)';
n = find(VdsFB <= 5);
Coss(n) = 25000-VdsFB(n)*(25000-7500)/5;
m = find(VdsFB > 5 & VdsFB <= 10);
Coss(m) = 7.5e3+(VdsFB(m)-5)*(-7500+3750)/5;
k = VdsFB>10 & VdsFB <= 20;
Coss(k) = 3.75e3+(VdsFB(k)-10)*(-3750+1250)/10;
k = find(VdsFB>20);
Coss(k) = 1080;
Coss = Coss*1e-12;
Qoss = cumtrapz(VdsFB,Coss)';
CeffFB = Qoss./VdsFB;
%cycloconverter MOSFET datasheet fit
if(FET == 250)
    VdsCC = [1 10 37 50 75 100 125 150 175 200 250 300 350 400 500];
    CdsCC = 1e-12*[1e4 3e3 1e3 2e2 80 58 50 48 47 45 43 41 40 39 38];
    CeffCC = 1./VdsCC.*cumtrapz(VdsCC,CdsCC);
else %assume the 99 mOhm series FETs otherwise
    VdsCC = [1 12.5 25 37 50 68.7 75 100 225 300 400];
    CdsCC = 1e-12*[2e4 1e4 6e3 4e3 300 200 190 120 100 95 90];
    CeffCC = 1./VdsCC.*cumtrapz(VdsCC,CdsCC);
```

```
end

%the fraction of Vout to which the cycloconverter voltage has to rise
%to say that the zero crossing of Vcc fundamental occurs there.
%Typically leave this is 0.5
FF = 0.5;
%set the phase angles of Vout to test (0 to pi/2)
%this sets the output voltages of interes
VoutPhaseAngles = [0.6345,0.8134,0.9458,1.0566,1.1549,1.2454,...
    1.3306, 1.4124 ,1.4921 ,1.5707];

if(func == 0)
    TR = 7.5;           %Transformer turns ratio
    Lres = 220e-6;      %resonant components as seen on the primary
    Cres = 4.2e-8;

    Cblock = 130e-6;     % this is Cblock on primary
    Cblock = Cblock/(TR^2); %so it has to be scaled to the secondary
    %selecting the percent average power level.
    PercentPower =[1 .75 .5 .3 .2];
    %selecting the input voltage
    Vin = 25;
    Rpar = 2.81;        %some parasitic resistance. This makes it
                        %more accurate in needed to deliver more power
                        %than what a lossless inverter would see

    %current inductor parameters (for non-function mode, the inductor
    % geometry is fixed)
    %for function mode - the inductor parameters are passed to it

    N = 44;
    Nstrands = 243;
    ml = 3.6;
    Al = 88;
    wirediam = 0.051e-3; %mm AWG44
    wirearea = 0.00204e-6; %mm^2 AWG44
    klayer = 0.45;
    DCesr = 0;

else
    %function mode. L,C,and everyhings else is paased here. Just
    %select the ave power levels
    PercentPower =[1 .75 .5 .3 .2];
end

fend =600e3;    %do not try to go above this frequency
Pave_max = 175; %max average power
Vrms = 240;     %RMS voltage good
Vline = sin(VoutPhaseAngles).*Vrms*sqrt(2);
```

```
%the lowest resonant frequency (it can increase when the parasitic
%capacitance of the cycloconverter goes up
Fres0 = 1./2/pi./sqrt(Lres.*1./(1./Cres+1./Cblock));

%transformer magnetizing inductance on the primary
%a fixed value since the transformer is fixed
Lu = 130e-6; %H
clear j;
oppts = [];
%the minimum equivalent output resitor: when cycloconverter has zero
%parasitic capacitance and operates at the highest average power
Rlprime = 66./PercentPower;
%compute variation in power given the specified output voltages
PowerPercentLineCycle = sin(VoutPhaseAngles).^2;
%for each percentage average power
for pm = 1:length(PercentPower)
    %find the maximum instantaneous power output
    Pout_max = 2*Pave_max.*PercentPower(pm);
    Rldprime = Rlprime(pm);

    %find the needed instantaneous output power for that average power
    %level and given angles in the line cycle
    power = PowerPercentLineCycle.*Pout_max;
    pl = length(power);

    %for each input voltage
    for vi = 1:length(Vin)
        %compute the fundamental of the input voltage
        Vinput = Vin(vi)*TR*4/pi;
        Qfb = Vin(vi)*interp1(VdsFB,CeffFB,Vin(vi))/TR;

        %step through the line cycle positions
        for n = 1:pl
            %Get the instanataneous output power at that point
            Pout_rqrd = power(n);

            %0 = Roughy adjust power for what efficiency is expected
            %to be (to get closer to actual power delivery
            %requirement. 1 = assume efficiency is 1
            if(1)
                eta = 1;
            else
                if(Pout_rqrd>80)
                    eta = .96;
                elseif(Pout_rqrd>50)
                    eta = .94;
                elseif(Pout_rqrd>20)
                    eta = 0.9;
                else
                    eta = 0.85;
```

```
    end
end
Pout_rqrd = Pout_rqrd./eta;

%Get the output voltage at this point
Vout = Vline(n);
%Estimate the charge needed to be delivered to the
%cycloconverter FETs
Qcc = FF*2*interp1(VdsCC,CeffCC,Vout)*Vout;

%Adjust the accuracy of the power delivery. Anything above
%100V at the output, make it to within 1 percent, and so
%on
if(Vout > 100)
    cc = .01;
elseif(Vout > 50)
    cc = 0.02;
elseif(Vout > 25)
    cc = 0.03;
else
    cc = 0.04;
end

%now that the output voltage and power are known, find the
%combination of frequency and full bridge pulse width that
%satisfies that
for f = Fres0:0.25e3:fend
    w = 2*pi*f;
%assume phi (cycloconverter phase shift) is zero at first
    phi = 0;
%then converge onto a consistent value in twenty steps of
%less (a limits is placed for speed).
    for tr = 1:1:20
       %On the first iteration this will give the lowest
       %concievable current - for that case of phi = 0
       Ipk_rqrd_min = Pout_rqrd*pi./Vout/cos(phi);
       %Assuming a sinuisodal current
       %Q = integral_{0}^{w*t}{Ipk_rqrd_min*sin(w*t)dt}
       % solve for when Q > Qcc
       if(Qcc*w/(Ipk_rqrd_min) > 1)
        %This occurs when the minimum current is too small
        %and the gues need to move towards the maximum
        %phase shift.
           phi_min = mean([phi,pi/2]);
       else
           %this will find lowest resonant current at
           %the lowest frequency
           phi_min = acos(1-Qcc*w./(Ipk_rqrd_min));
       end
        %phi shall be accurate to within 5 mRadians
```

```
    phidiff = phi_min-phi;
    if(abs(phidiff) > 0.005)
        phi = phi + 0.5*(phidiff);
    else
        break;
    end
end

%the minimum possible current has been found
%set required to equal minimum to achieve lowest loss
Ipk_rqrd = Ipk_rqrd_min;
%estimate the necessary margin for FB switching,
%in degrees. Impose and arbitrary minimum of 3 deg
FB_ZVS = max(3,180/pi*acos(1-Qfb*w./(Ipk_rqrd_min)));
opptstemp = [];
amin = 0;

%model the load at this frequency (see Hayman 2009)
%find 1/Cload. might not be meaningful yet
OneOverCload = w*Rldprime*sin(phi)*cos(phi);
%find corresponding Rload (adjust Rldprime for phi)
Rload = Rldprime*cos(phi).^2;
%find equivalent total Cres of tank
Ceq = (1./Cres + 1./Cblock + OneOverCload ).^-1;
%new Fres, accounting for the change in effective Cres
Fres = 1./2/pi./sqrt(Lres.*Ceq);

%skip this f if the total impedance is not inductive
%or if f is too close to resonance
if(f < 1.01*Fres)
    continue;
end

%Total complex admittance at the fundamental
Y = 1./(1./(1i*w*Ceq)+1i*w*Lres+Rload+Rpar);
%should be negative for inductive load
Itheta = angle(Y);

%calculate the magnetizing impedance of the
%transformer at the fundamental
Zu = w*Lu*TR.^2;

%Calculate the minimum FB pulse width
%for this angle of current and frequency
%(minimum current mode)
%delta is the phase angle
delta = FB_ZVS*pi/180+Itheta+pi/2;
%percent delta is the 'duty ratio'.
percent_delta = delta*2/pi;
```

```
%setup the impedance equation for the third harmonic
%for the sole purpose of calculating LOSS, and not
%considering it for the power transfer.
%(this is needed to impose some condition against
%going to extremely low quality factor
I3 = (Vinput*sin(3*delta)/3-(2*Vout./pi./3.*...
  (-sin(3*(Itheta-phi)) - 1i*cos(3*(Itheta-phi)))))...
  ./(1./(1i*3*w*Cres)+1i*3.*w*Lres+Rpar);
Ipk3 = abs(I3);

%the the duty ratio is too small or too large
%(arbitrary imposed limits)
if(percent_delta>=0.99 || percent_delta < 0.3)
    continue;
else
    %make the actual Ipk out of admittance. This is
    %the total available resonant current to go into
    %the secondary
    Itotal = abs(Y)*Vinput*sin(delta);

    %Calculate the resonant current after the current
    %divider that is the magnitizing inducrance (that
    %is part of the available goes into the core of
    %the transformer
    Ipk = Itotal.*(Zu./(Zu+1./abs(Y)));
    Iu = Itotal-Ipk;

    %compute output power
    Pactual = (Vout*Ipk*cos(phi)/pi);
    %see if it is within accuracy limit
    if(abs(Pactual-Pout_rqrd*eta)<cc*Pout_rqrd*eta)
         %compute the final Q
         Q = 1./Rload*sqrt(Lres./Ceq);
         %compute the input current
         Iin = Pout_rqrd./Vin(vi);

     %----Supplemental Block-----
     %If it is desired to move the resonant tank
     %onto the primary side, this will aid in
     %determining the voltages V1 and V2 in setup:
     %  __----__---Cres---Lres---______-----
     %  FB    V1                 _Xfrmr_   V2
     %  ------------------------       ----
     % V1 and V2 magnitude and phase are computed
     %This needs values that are primary referenced
     %Cres as seen on the primary
     Cres1 = Cres*TR^2;
     %Ipk as seen on the primary
     Ipk1 = TR*Ipk;
     Ac = Ipk1./w./Cres1;
```

```
        A1 = Vin(vi)*4/pi*sin(delta);
        ThetaC = -Itheta-pi/2;

    %the fundamental component of the full bridge
    %output waveform represents the reference phase

%theta1 = angle of the fundamental of voltage waveform
%of the primary side of the would-be informer (L+Xfrmr)
Theta1 = atan((A1+Ac*sin(ThetaC))./(A1+Ac*cos(ThetaC)));
%theta2 = angle of the fundamental of voltage waveform
%of the secondary side of the would-be informer (L+Xfrmr)
        Theta2 = Itheta-phi;
        %--end--Supplemental Block-----

        if(lossCalc ~= 0)
        %calculate loss
        [Ploss,FBtotal,FBcond,FBsw,FBdiode,CCtotal,...
        CCcond,CCsw,CCdiode,Ltotal,Lwind,Lcore,Xtotal,...
        Xwind1,Xwind2,Xcore,Xpar,LossParasitic]...
        =loss4([f,3*f],[Ipk,Ipk3],Iu,percent_delta,...
        Vin(vi),Vout,Lres,TR,N,Nstrands,ml,wirediam,...
        wirearea,klayer,Al,DCesr,30);
        if(Ploss==-1)
           aa = -1;
           fave= -1;
           Ipkave= -1;
           if(func)
               return;
           else
               Ploss = 1000;
           end
        end
        if(Ploss > 0.4*Pout_rqrd*eta)
        %if the estimate loss is more that 40 % of
        %delivered power, skip this (need to aim to
        %deliver more power)
           continue;
        end
        else  %do not compute loss
            Ploss=0;FBtotal=0;FBcond=0;FBsw=0;...
            FBdiode=0;CCtotal=0;CCcond=0;CCsw=0;...
            CCdiode=0;Ltotal=0;Lwind=0;Lcore=0;...
            Xtotal=0;Xwind1=0;Xwind2=0;Xcore=0;...
            Xpar=0; LossParasitic=0;
        end
       %real is taken for prettier output. Since
       %imaginary crept into the equations, i is
       %carried thru. But it is always 0000*i so this
       %is fine
       opptstemp = real([opptstemp;...
```

```
                    [PercentPower(pm),Vin(vi),Vout,eta,...
                    Pout_rqrd*eta,f,Fres,Q,percent_delta,...
                    phi*180/pi,FB_ZVS,Itheta*180/pi,...
                    Itotal,Iu,Ipk,Ipk3,Iin,Ceq,Rload,...
                    Theta1, Theta2, Theta2-Theta1,...
                    Ploss+Lcore,FBtotal, FBcond, FBsw, FBdiode,...
                    CCtotal, CCcond, CCsw, CCdiode,...
                    Ltotal, Lwind, Lcore, Xtotal, Xwind1,...
                    Xwind2, Xcore, Xpar,LossParasitic],...
                    phi_min, 0, Pactual, 1-Ploss./(Pactual),...
                    Pactual/(Pout_rqrd*eta)]);
                    %this will pick the lowest frequency that works
                       break;
                   else
                       continue;
                   end
               end
           end
           %once the operating point is found, save to the main array
           oppts = [oppts; opptstemp];
       end
   end
end

%based on how many average powers are calculated, pick the CEC
%weighting coefficients, lumping the lowest ones togehter if needed
if(length(PercentPower) == 4)
    coeffs = [.05 .53 .21 .21];
elseif(length(PercentPower) == 5)
    coeffs = [.05 .53 .21 .12 .09];
else
    coeffs = [.05 .53 .21 .12 .05 .04];
end

%sampled time points in a quarter of a line cycle
time = VoutPhaseAngles;
vpts = length(PowerPercentLineCycle);

t = [0;time'];
[r,c] = size(oppts);
eff = [];

%proceed only if every voltage at every power level was calculated
if(r == vpts*length(coeffs))
    for p = 1:length(PercentPower)
        for k = 1:vpts
            elost = trapz(t,[0;oppts(1+(p-1)*vpts:p*vpts,23)]);
            edeliv = trapz(t,[0;oppts(1+(p-1)*vpts:p*vpts,43)]);
             %eff = 1-integral(Plost)/integral(Pin)
             %both itegrate over 1/4 of line cycle
```

```
            eff(p,1) = 1 - elost/edeliv;
        end
    end
end

%this check is probably not needed
if(length(eff) ~= length(coeffs))
    cecEff = -1;
else
    cecEff = coeffs*eff;
end

%Literally average the resulting frequencies and current
fave = mean(oppts(:,6));
Ipkave = mean(oppts(:,15));

%Set this to if(1) to shows summary plots of the operating points
if(0)
%generate the tick labels for the xaxis to plot the results
nPts = length(PowerPercentLineCycle);
regs = length(PercentPower);
%put an x tick in the middle of each average power
xticks = nPts/2+(0:regs-1)*nPts;

figure;
subplot(2,3,1)
plot(oppts(:,23),'.b','MarkerSize',15);
set(gca, 'FontSize',14)
ylabel('Loss over 1/4 line cycle [W]')
set(gca,'XTick',xticks)
if(length(xticks)==6)
  set(gca,'XTickLabel',{num2str(PercentPower(1)),...
    num2str(PercentPower(2)),num2str(PercentPower(3)),...
    num2str(PercentPower(4)),num2str(PercentPower(5)),...
    num2str(PercentPower(6))});
elseif(length(xticks)==5)
  set(gca,'XTickLabel',{num2str(PercentPower(1)),...
    num2str(PercentPower(2)),num2str(PercentPower(3)),...
    num2str(PercentPower(4)),num2str(PercentPower(5))});
elseif(length(xticks)==4)
  set(gca,'XTickLabel',{num2str(PercentPower(1)),...
    num2str(PercentPower(2)),num2str(PercentPower(3)),...
    num2str(PercentPower(4))});
end

subplot(2,3,2)
plot(oppts(:,44),'.b','MarkerSize',15);
set(gca, 'FontSize',14)
xlabel('Average power fraction')
ylabel('Efficiency')
```

```
ylim([.95 1])
%to save some paper - leaving this blank for all but 1 subplot
set(gca,'XTickLabel',{''});
subplot(2,3,3)
plot(oppts(:,10),'.b','MarkerSize',15);
set(gca, 'FontSize',14)
xlabel('Average power fraction')
ylabel('\phi [degrees]')

subplot(2,3,4)
plot(oppts(:,6),'.b','MarkerSize',15);
set(gca,'XTickLabel',{''});
set(gca, 'FontSize',14)
xlabel('Average power fraction')
ylabel('Frequency')

subplot(2,3,5)
plot(oppts(:,15),'.b','MarkerSize',15);
set(gca, 'FontSize',14)
set(gca,'XTickLabel',{''});
xlabel('Average power fraction')
ylabel('Current [A]')


%plots the loss distribution
loss8 = figure;
plot(oppts(:,24)./oppts(:,23),'.r','MarkerSize',15);
hold on
plot(oppts(:,28)./oppts(:,23),'.b','MarkerSize',15);
plot(oppts(:,32)./oppts(:,23),'.k','MarkerSize',15);
plot(oppts(:,35)./oppts(:,23),'.g','MarkerSize',15);
plot(oppts(:,36)./oppts(:,23),'.c','MarkerSize',15);
set(gca, 'FontSize',14)
xlabel('Average power fraction')
ylabel('Fraction of total loss')
legend('Full-bridge','Cycloconverter', 'Inductor','Transformer',...
    '50 mOhm parasitic','Orientation','Horizontal','Location',...
    'NorthOutside');
hold off
set(gca,'XTick',xticks)
if(length(xticks)==6)
  set(gca,'XTickLabel',{num2str(PercentPower(1)),...
    num2str(PercentPower(2)),num2str(PercentPower(3)),...
    num2str(PercentPower(4)),num2str(PercentPower(5)),...
    num2str(PercentPower(6))});
elseif(length(xticks)==5)
  set(gca,'XTickLabel',{num2str(PercentPower(1)),...
    num2str(PercentPower(2)),num2str(PercentPower(3)),...
    num2str(PercentPower(4)),num2str(PercentPower(5))});
elseif(length(xticks)==4)
```

```
  set(gca,'XTickLabel',{num2str(PercentPower(1)),...
    num2str(PercentPower(2)),num2str(PercentPower(3)),...
    num2str(PercentPower(4))});
end
end
```

loss4.m

```
function [Ploss,FBtotal, FBcond, FBsw, FBdiode,CCtotal, CCcond,...
    CCsw, CCdiode,Ltotal, Lwind, Lcore, Xtotal, Xwind1, Xwind2,...
    Xcore, Xpar,LossParasitic]=loss4(Fvec, Ipks, Im, duty, Vin,...
    Vout,Lres,TR, Nturns, Nstrands, Nlayers,wirediam,wirearea,...
    klayer,Al,DCesr,Temperature)
%given phaseshifts,power,voltages,l,c, esr and xfromer and inductor
%structure. compute the expected losses. The parameters for all FETs
%are hardcoded into this function. Board ESR is also in this function
%as some fixed value
%-1 is returned if saturation is detected in any magnetic component
%%
%parasitics
Rcres = 0.05;        %ESR Cres
Rcblock = 0.000;     %ESR Cblock
RboardPrimary = 0.000; %Esr board
LossParasitic=(Rcres+(Rcblock+RboardPrimary)*TR.^2)*0.5*sum(Ipks.^2);
%%
% Full Bridge - ST Mosfets
Rdsfb = 0.00357; %ohm
QgFB = 85e-9; %85 at 10V, 100 at 12V.
Cissfb = 6750e-12;
Crssfb = 40e-12;
Vgs = 10;
deadtimefb = 150e-9;    %a typical deadtime
[FBtotal, FBcond, FBsw, FBdiode] = FBloss(Vin, TR, Vgs, Fvec,...
    (Im+Ipks)./sqrt(2),Rdsfb, Crssfb, Cissfb, QgFB, deadtimefb);
%%
%Cycloconverter Infenion IPP60R250CP Mosfets
RdsCC = 0.25;
CissCC = 1200e-12;
CrssCC = 1.2e-12;
QgCC = 26e-9;   %26V at 10V 30 at 12V
Vgs = 10;
deadtimecc = 700e-9; %typical deadtime
[CCtotal, CCcond, CCsw, CCdiode] = CCloss(Vout, Vgs, Fvec,...
    Ipks./sqrt(2), RdsCC, CissCC, CrssCC, QgCC, deadtimecc);

%%
```

```
%Inductor
[Ltotal, Lwind, Lcore] = Lloss(Fvec,Ipks,Lres,Al,Nturns,Nstrands,...
    Nlayers,wirediam, wirearea,klayer, Temperature,DCesr);

if(Ltotal == -1)    %the case of saturation
    Xtotal = -1;
    Xwind1= -1;
    Xwind2= -1;
    Xcore= -1;
    Xpar= -1;
    Ploss= -1;
    return
end
%%
%Xformer parameters, static
AlX = 8130;
Cpar =0;% 12.5e-12;

wire1diam = .0799*1e-3; %wire diameter, m awg = 40;
wire1area = 0.00501*1e-6; %m^2
k1layer = .45;
N1strands = 600;
N1layers = 1;
N1turns = 4;
DCesr1 = 0;

N2strands = 80;
N2layers = 3;
N2turns = 30;
wire2diam = .0799*1e-3; %mm AWG40
wire2area = 0.00501*1e-6; %mm^2 AWG40
k2layer = 0.45;
DCesr2 = 0;
Lu = 130e-6;
[Xtotal, Xwind1, Xwind2, Xcore, Xpar] = Xloss(TR,Fvec,Ipks,Vin,...
    duty,AlX,N1turns,N1strands,N1layers,wire1diam,wire1area,...
    k1layer,N2turns,N2strands,N2layers,wire2diam, wire2area,...
    k2layer, Temperature,DCesr1, DCesr2, Lu, Cpar);
Ploss = Xtotal+Ltotal+CCtotal+FBtotal+LossParasitic;
```

FBloss.m

```
function [FBtotal,FBcond,FBswQ,FBdiode]=FBloss(Vin,TR,Vgs,f,Irms,...
    Rds, Ciss, Crss, Qg,deadtime)
%Compute the total, conducation, gating, and diode voltage power loss
%in 1 full bridge
```

```
%f = vector of frequencies to consider. 1st entry = switching
%frequency. other entries are optional, Hz
%Ipk = vector of peak current, each corresponding to a frequency in f,
%at the same index, A
%TR = turns ratio
%Vin = input voltage (25-40V), V
%Vgs = gate to source (12V), V
%Rds = On state resistance, ohm
%Ciss = input capacitance, F,
%Crss = drain-gate capactiance, F
%deadtime in each leg for diode power loss calculation, s
%gating loss
FBswQ = 4*Qg*Vgs*f(1);
%a different way to calculating gating loss. Not used now
%FBswC = 2*f(1)*((Ciss)*Vgs^2 + (Crss)*(Vin-Vgs)^2);
%conduction loss
FBcond = 2*Rds.*TR.^2*sum(Irms.^2);  %x2 for 2 legs
%diode loss:
%a very poor esitmate. Included in hopes of making a better estimate
%later and is not included in the total loss.
%about half of peak current * 2 legs * ~0.5V
FBdiode = 0.5*TR.^2*sum(Irms)*deadtime;
%add gating an conduction
FBtotal = FBcond + FBswQ;
```

CCloss.m

```
function [CCtotal,CCcond,CCswQ,CCdiode]=CCloss(Vout,Vgs,f,Irms,Rds,...
    Ciss, Crss, QgCC,deadtime)
%Compute the total, conducation, gating, and diode voltage power loss
%in the cyclconverter
%f = vector of frequencies to consider. 1st entry = switching
%frequency. other entries are optional, Hz
%Ipk = vector of peak current, each corresponding to a frequency in f,
%at tthe same index, A
%Vout = output voltage (0-339V), V
%Vgs = gate to source (12V), V
%Rds = On state resistance, ohm
%Ciss = input capacitance, F,
%Crss = drain-gate capactiance, F
%deadtime in each leg for diode power loss calculation, s
%conduction loss
CCcond = 1.5*Rds.*sum(Irms.^2);
%gating loss
CCswQ = 2*QgCC*Vgs*f(1);
%a different way to calculating gating loss. Not used now
```

```
% CCswC = f(1)*((Ciss)*Vgs^2 + (Crss)*(Vout-Vgs)^2);
%diode loss:
%a very poor esitmate. Included in hopes of making a better estimate
%later and is not included in the total loss.
%deadtime occurs at ~0.1*Ipk of current, 0.45V drop
CCdiode = 0.45*0.1*sum(Irms)*deadtime;
%total loss
CCtotal = CCcond + CCswQ ;%+ CCdiode;
```

Lloss.m

```
function [Ltotal, Lwind, Lcore] = Lloss(f,Ipks,L,Al,Nturns,Nstrands,...
    Nlayers,wirediam, wirearea,klayer, Temperature,DCesr)
%Compute the total, conduction, and core loss for the inductor in rm14
%given all of its properties. Assumes copper winding
%f = vector of frequencies (for core loss, only f(1) is used)
%Ipks = vector of peak currents those frequencies
%L = Inductance, currently not used
%Al = nH/Turn^2
%Nturns = Number of turns
%Nstrands = Number of total litz strands in a turn
%Nlayers = [can be fractional] number of layers
%wirediam = wire diameter, m
%wirearea = wire ares, m^2
%klayer = packing factor
%DCesr = if there is a measurement for ESR at dc, set it to this value
%to%use it otherwise set this to 0 and theoretical DC esr will be used
%Temperature = degrees C
%Returns -1 for all three if the core flux is 300 mT (saturation)

%for rm14: geometric parameters
mlt = 71e-3; %average length per turn, m
Ae = 198e-6; % Ac is effective core area in m^2
Vc = 13900e-9; % Vc is effective core volume in m^3
Wa = 112e-6;    % wa is bobbin winding area in m^2
Ww = 18.4e-3;   % ww is bobbin winding width in m

%resistivity at a temperature, ohm-meters
rhocu = 1.72e-8*(1 + .0039*(Temperature-20));
mu0 = 4*pi*1e-7;   %N/m
%skindepth of copper at that Frequency.
skindepth = 1/sqrt(pi*0.999994*mu0).*sqrt(rhocu./f);

%the parameters needed for AC resistance. Follow Gu's paper for the
%method
x = wirediam.*sqrt(pi.*klayer)./(2*skindepth);
```

```
fr = 1+((5.*Nlayers^2.*Nstrands-1)/45).*x.^4;
%if DC esr is not passed to the function, estimate it here
%R = rho*Length/Area
if(DCesr == 0)
    Rdc = rhocu*mlt.*Nturns./(wirearea.*Nstrands);
else
    Rdc = DCesr;
end

%Estimate the AC resistance
Rac = fr.*Rdc;
Lwind = sum(Rac.*0.5.*Ipks.^2);

%%
%Core loss
%find Bpeak - formulas for loss are based on Bpk, not Bave
Bpks = Al*1e-9*Nturns*Ipks(1)./Ae;       %in Tesla
Bmax = max(Bpks)*1000;                   %in mT
%check for saturation.
if(Bmax > 300)
   warning('Inductor is close to saturation flux, Bpk = %d', Bmax);
   Ltotal = -1;
   Lwind = -1;
   Lcore = -1;
   return;
end

%Two ways to find core loss. (Un)comment as needed
% % Steinmetz parameters: version 1 - Ferroxcube App note.
% for n = 1
%     if(f(n) < 300e3)
%         %this is in mW/cm^3
%         Lcore(n) = .25e-3*(f(n))^1.63*Bpks(n)^2.45*(1.26e4).^-1*...
%             (.790e-4 - 1.05e-2*Temperature + 1.26*Temperature.^2);
%     elseif(f(n) < 500e3)
%         %this is in mW/cm^3
%         Lcore(n) = 2e-5*(f(n))^1.8*Bpks(n)^2.5*(1.28e4).^-1*(.770e-4
%         - 1.05e-2*Temperature + 1.28*Temperature.^2);
%     else
%            %this is in mW/cm^3
%         Lcore(n) = 3.6e-9*(f(n))^2.4*Bpks(n)^2.25*(.670e-4 - ...
%           0.81e-2*Temperature + 1.14*Temperature.^2);
%     end
% end
% Lcore = Lcore*Vc*1e3; %0.001 for mW to W x 1e6 for m^3 to cm^3 = 1e3
% Lcore = sum(Lcore);

% %Steinmetz parameters: version 2 - this texbook
% http://books.google.com/books?id=p3KHPUhaOaUC&pg=PA74&lpg=PA74&dq=3f
% 3+steinmetz+parameters&source=bl&ots=nel4cI2uxB&sig=dfbnzWbwaVctqwK1
```

```
% gksq8jXmnO0&hl=en&ei=LjrqS4GzOIH38QbwnfTxAw&sa=X&oi=book_result&ct=r
% esult&resnum=9&ved=0CD4Q6AEwCA#v=onepage&q&f=false
Lcore = 47.434*(f(1)/1e3)^1.3*Bpks(1)^2.5*13.9*.001;

%results are similar for both
Ltotal = Lcore+Lwind;
```

Xloss.m

```
function [Xtotal, Xwind1, Xwind2, Xcore, Xpar] = Xloss(TR,f,Ipks,...
    Vin,delta,Al,N1turns,N1strands,N1layers,wire1diam,wire1area,...
    k1layer,N2turns,N2strands,N2layers,wire2diam,wire2area,...
    k2layer,Temperature,DCesr1,DCesr2,Lu,Cpar)
%Compute the total, conduction and core (primary and secondary - 1
%and 2) and the loss due to parasitic capacitance across the secondary
%for RM14 3C95 core transformer given all of its properties.
%Assumes copper winding
%TR = turns ratio
%f = vector of frequencies (for core loss, only f(1) is used)
%Ipks = vector of peak currents those frequencies
%L = Inductance, currently not used
%Al = nH/Turn^2
%delta = full bridge pulse width fraction (0 to 1)
%for primary and secondary (1and 2):
%Nturns = Number of turns
%Nstrands = Number of total litz strands in a turn
%Nlayers = [can be fractional] number of layers
%wirediam = wire diameter, m
%wirearea = wire ares, m^2
%klayer = packing factor
%DCesr = if there is a measurement for ESR at dc, set it to this value
%to use it.otherwise set this to 0 and theoretical DC esr will be used
%Lu = magnetizing inductance. Set it to 0 to let the function estimate
%it.
%Cpar = parasitic capacitance across the transformer, can be 0 for no
%such loss mechanism
%Temperature = degrees C
%Returns -1 for all losses if the core flux is 300 mT (saturation)

%Al = nH/Turn^2 of the core
%Al = 8130; %for 3C95
%Al = 5700; %for 3f3

%Rm 14 geometric parameters
mlt = 71e-3; %average length per turn, m
Ae = 198e-6; % Ac is effective core area in m^2
```

```
Vc = 13900e-9; % Vc is effective core volume in m^3
Wa = 112e-6;    % wa is bobbin winding area in m^2
Ww = 18.4e-3;   % ww is bobbin winding wind in m

%resistivity at a temperature, ohm-meters
rhocu = 1.72e-8*(1 + .0039*(Temperature-20));
mu0 = 4*pi*1e-7;   %N/m
%skindepth
skindepth = 1/sqrt(pi*0.999994*mu0).*sqrt(rhocu./f);
%like the inductor, follow Gu's paper
x1 = wire1diam.*sqrt(pi*k1layer)./(2*skindepth);
x2 = wire2diam.*sqrt(pi*k2layer)./(2*skindepth);

fr1 = 1+((5*N1layers^2*N1strands-1)/45).*x1.^4;
fr2 = 1+((5*N2layers^2*N2strands-1)/45).*x2.^4;

%if DC esr is not passed to the function, estimate it here
%R = rho*Length/Area
if(DCesr1 == 0)
    Rdc1 = rhocu*mlt*N1turns./(wire1area*N1strands);
else
    Rdc1 = DCesr1;
end

if(DCesr2 == 0)
    Rdc2 = rhocu*mlt*N2turns./(wire2area*N2strands);
else
    Rdc2 = DCesr2;
end
Rac1 = fr1.*Rdc1;
Rac2 = fr2.*Rdc2;
Xwind1 = sum(TR.^2*Rac1.*0.5.*Ipks.^2);
Xwind2 = sum(Rac2.*0.5.*Ipks.^2);
%calculate the fundamental of the fullbridge
Vfb1 = Vin*4/pi*sin(delta*pi/2);
%loss due to parasitic cap across secondary
Xpar = Cpar*(Vfb1*2*TR)^2.*f(1);
%to find core loss, find the magnetizin inductance if it were not
%passed to the function
if(Lu == 0) %if the measured Lu is not passed to the function, find it
    Lu = Al*N1turns.^2*1e-9;
end
Ipks = Vfb1./Lu./(2*pi*f);
%find Bpeak - formulas for loss are based on Bpk, not Bave
Bpks = Al*1e-9*N1turns*Ipks(1)./Ae;
Bsat = .300;     %T, saturation flux density
Bmax = max(Bpks);
%check for saturation
if(Bmax > Bsat)
   warning(['Transformer is over the saturation (volt seconds), '...
```

```
        'Bpk = %d, f=%d'], Bmax,f(1));
   Xtotal = -1;
   Xwind1 = -1;
   Xwind2 = -1;
   Xcore = -1;
   Xpar = -1;
   return;
end
fmin = 1/2/pi*Vfb1./N1turns./Ae./(Bsat);
%check for saturation again
if(f < fmin)
   warning(['Transformer is over the saturation (volt seconds), '...
        'Bpk = %d, f=%d'], Bmax,f(1));
   Xtotal = -1;
   Xwind1 = -1;
   Xwind2 = -1;
   Xcore = -1;
   Xpar = -1;
   return;
end
%Convert frequency to kHz
F = f(1)/1000;
%Manual fit to the 3c95 datasheet (maybe exp and quadratic is not the
%best choice but it seemed to give the best fit)
a = 1.3489e-5*exp(1.6266e-2*F);
c = -4.4833e-6*F.^2 + 7.0500e-4*F + 2.8573;
%Ploss = kW/m^3 when Bpk is in mT
Xcore = sum(a.*(Bpks(1).*1000).^c);
%convert to W lost
Xcore = 1000*Xcore.*Vc;
%total transformer loss
Xtotal = Xcore+Xwind1+Xwind2+Xpar;
```

---

suboptS.m

```
%This script calculates the most efficient set of control inputs for
%a specified set of operating points. This script is very similar to
%sub_optG.m, however it also explores the use of cycloconverter phase
%shift as a control handle. In addition to minimal current mode,
%energy sloshing is also considered. The most efficient set of control
%inputs is stored in array 'opsMax'. All valid control inputs are
%also stored, in array 'opsAll'. There is only a script mode for this
%function, although due to extreme similarity with sub_optG.m it is
%very easy to extend it to a function mode, whic will make it be
%callable by optimize.m
%Line 71 sets the average power levels to calculated
```

```
%The CEC efficiency estimate will give an error if more than 1 input
%voltage is specified and to average over several input voltages, the
%script has to be run for each case. (control inputs are still found)
%for script mode, inputs start at line 62
%The range of output voltages in set on line 65
%column 42 of opsMax and opsAll indicated the cycloconverter phase
%shift beyond the minimum, which results in most efficient operation.
%(in sub_optG.m, column 42 of oppts was always 0 as the miminum phi
%was always used).

%calculate losses. always has to be 1
lossCalc = 1;
FET = 250;
clear pi i j Coss
%Full Bridge MOSFET datasheet fit
VdsFB = (0.1:0.1:50)';
n = find(VdsFB <= 5);
Coss(n) = 25000-VdsFB(n)*(25000-7500)/5;
m = find(VdsFB > 5 & VdsFB <= 10);
Coss(m) = 7.5e3+(VdsFB(m)-5)*(-7500+3750)/5;
k = VdsFB>10 & VdsFB <= 20;
Coss(k) = 3.75e3+(VdsFB(k)-10)*(-3750+1250)/10;
k = find(VdsFB>20);
Coss(k) = 1080;
Coss = Coss*1e-12;
Qoss = cumtrapz(VdsFB,Coss)';
CeffFB = Qoss./VdsFB;

%cycloconverter MOSFET datasheet fit
if(FET == 250)
    VdsCC = [1 10 37 50 75 100 125 150 175 200 250 300 350 400 500];
    CdsCC = 1e-12*[1e4 3e3 1e3 2e2 80 58 50 48 47 45 43 41 40 39 38];
    CeffCC = 1./VdsCC.*cumtrapz(VdsCC,CdsCC);
else %assume the 99 mOhm series FETs otherwise
    VdsCC = [1 12.5 25 37 50 68.7 75 100 225 300 400];
    CdsCC = 1e-12*[2e4 1e4 6e3 4e3 300 200 190 120 100 95 90];
    CeffCC = 1./VdsCC.*cumtrapz(VdsCC,CdsCC);
end

%the fraction of Vout to which the cycloconverter voltage has to rise
%to say that the zero crossing of Vcc fundamental occurs there.
%Typically leave this is 0.5
FF = 0.5;
%set the phase angles of Vout to test (0 to pi/2)
%this sets the output voltages of interes
VoutPhaseAngles = [0.6345,0.8134,0.9458,1.0566,1.1549,1.2454,...
    1.3306, 1.4124 ,1.4921 ,1.5707];

TR = 7.5;           %Transformer turns ratio
Lres = 220e-6;      %resonant components as seen on the primary
```

```
Cres = 4.2e-8;

Cblock = 130e-6;     % this is Cblock on primary
Cblock = Cblock/(TR^2); %so it has to be scaled to the secondary
%selecting the percent average power level.
PercentPower =[1 .75 .5 .3 .2 .1];
%selecting the input voltage
Vin = 25;
Rpar = 2.81;        %some parasitic resistance. This makes it
                    %more accurate in needed to deliver more power
                    %than what a lossless inverter would see

%current inductor parameters (for non-function mode, the inductor
% geometry is fixed)
%for function mode - the inductor parameters are passed to it

N = 44;
Nstrands = 243;
ml = 3.6;
Al = 88;
wirediam = 0.051e-3; %mm AWG44
wirearea = 0.00204e-6; %mm^2 AWG44
klayer = 0.45;
DCesr = 0;

fend =600e3;    %do not try to go above this frequency
Pave_max = 175; %max average power
Vrms = 240;     %RMS voltage good
Vline = sin(VoutPhaseAngles).*Vrms*sqrt(2);

%the lowest resonant frequency (it can increase when the parasitic
%capacitance of the cycloconverter goes up
Fres0 = 1./2/pi./sqrt(Lres.*1./(1./Cres+1./Cblock));

%transformer magnetizing inductance on the primary
%a fixed value since the transformer is fixed
Lu = 130e-6; %H
clear j;
opSall = [];
opMax = [];

%the minimum equivalent output resitor: when cycloconverter has zero
%parasitic capacitance and operates at the highest average power
Rlprime = 66./PercentPower;
%compute variation in power given the specified output voltages
PowerPercentLineCycle = sin(VoutPhaseAngles).^2;

%for each percentage average power
for pm = 1:length(PercentPower)
    %find the maximum instantaneous power output
```

```
Pout_max = 2*Pave_max.*PercentPower(pm);
Rldprime = Rlprime(pm);

%find the needed instantaneous output power for that average power
%level and given angles in the line cycle
power = PowerPercentLineCycle.*Pout_max;
pl = length(power);

%for each input voltage
for vi = 1:length(Vin)
    %compute the fundamental of the input voltage
    Vinput = Vin(vi)*TR*4/pi;
    Qfb = Vin(vi)*interp1(VdsFB,CeffFB,Vin(vi))/TR;

    %step through the line cycle positions
    for n = 1:pl
        %Get the instanataneous output power at that point
        Pout_rqrd = power(n);
        %0 = Roughy adjust power for what efficiency is expected
        %to be (to get closer to actual power delivery
        %requirement. 1 = assume efficiency is 1
        if(1)
            eta = 1;
        else
            if(Pout_rqrd>80)
                eta = .96;
            elseif(Pout_rqrd>50)
                eta = .94;
            elseif(Pout_rqrd>20)
                eta = 0.9;
            else
                eta = 0.85;
            end
        end
        Pout_rqrd = Pout_rqrd./eta;

        %Get the output voltage at this point
        Vout = Vline(n);
        %Estimate the charge needed to be delivered to the
        %cycloconverter FETs
        Qcc = FF*2*interp1(VdsCC,CeffCC,Vout)*Vout;

        %Adjust the accuracy of the power delivery. Anything above
        %100V at the output, make it to within 1 percent, and so
        %on
        if(Vout > 100)
            cc = .01;
        elseif(Vout > 50)
            cc = 0.02;
        elseif(Vout > 25)
```

```
    cc = 0.03;
 else
    cc = 0.04;
 end

%clear out the temp
 opptstemp = [];

 %now that the output voltage and power are known, find the
 %combination of frequency and full bridge pulse width that
 %satisfies that
 for f = Fres0:0.25e3:fend
    w = 2*pi*f;
%assume phi (cycloconverter phase shift) is zero at first
    phi = 0;
%then converge onto a consistent value in twenty steps of
%less (a limits is placed for speed).
    for tr = 1:1:20
       %On the first iteration this will give the lowest
       %concievable current - for that case of phi = 0
       Ipk_rqrd_min = Pout_rqrd*pi./Vout/cos(phi);
       %Assuming a sinuisodal current
       %Q = integral_{0}^{w*t}{Ipk_rqrd_min*sin(w*t)dt}
       % solve for when Q > Qcc
       if(Qcc*w/(Ipk_rqrd_min) > 1)
        %This occurs when the minimum current is too small
        %and the gues need to move towards the maximum
        %phase shift.
           phi_min = mean([phi,pi/2]);
       else
           %this will find lowest resonant current at
           %the lowest frequency
           phi_min = acos(1-Qcc*w./(Ipk_rqrd_min));
       end
        %phi shall be accurate to within 5 mRadians
        phidiff = phi_min-phi;
        if(abs(phidiff) > 0.005)
            phi = phi + 0.5*(phidiff);
        else
            continue;
        end
    end

    %the minimum possible current has been found
    %set required to equal minimum to achieve lowest loss
    Ipk_rqrd = Ipk_rqrd_min;

    if(~isreal(pi/2-phi_min))
       error('unreal phi') ;
    end
```

```
%identical to sub_optG.m up to here but now
%add a loop to iterate over phi
for phi_s = 0:0.01:pi/2-phi_min
   phi = phi_min+phi_s;
    %set required to equal minimum
   Ipk_rqrd = Ipk_rqrd_min.*cos(phi_min)./cos(phi_min+phi_s);
   if(Ipk_rqrd > 15*Ipk_rqrd_min)
       break;
   end
   FB_ZVS = max(3,180/pi*acos(1-Qfb*w./(Ipk_rqrd_min)));
end
%estimate the necessary margin for FB switching,
%in degrees. Impose and arbitrary minimum of 3 deg

%model the load at this frequency (see Hayman 2009)
%find 1/Cload. might not be meaningful yet
OneOverCload = w*Rldprime*sin(phi)*cos(phi);
%find corresponding Rload (adjust Rldprime for phi)
Rload = Rldprime*cos(phi).^2;
%find equivalent total Cres of tank
Ceq = (1./Cres + 1./Cblock + OneOverCload ).^-1;
%new Fres, accounting for the change in effective Cres
Fres = 1./2/pi./sqrt(Lres.*Ceq);

%skip this f if the total impedance is not inductive
%or if f is too close to resonance
if(f < 1.01*Fres)
    continue;
end

%Total complex admittance at the fundamental
Y = 1./(1./(1i*w*Ceq)+1i*w*Lres+Rload+Rpar);
%should be negative for inductive load
Itheta = angle(Y);

%calculate the magnetizing impedance of the
%transformer at the fundamental
Zu = w*Lu*TR.^2;

%Calculate the minimum FB pulse width
%for this angle of current and frequency
%(minimum current mode)
%delta is the phase angle
delta = FB_ZVS*pi/180+Itheta+pi/2;
%percent delta is the 'duty ratio'.
percent_delta = delta*2/pi;

%setup the impedance equation for the third harmonic
%for the sole purpose of calculating LOSS, and not
```

```
%considering it for the power transfer.
%(this is needed to impose some condition against
%going to extremely low quality factor
I3 = (Vinput*sin(3*delta)/3-(2*Vout./pi./3.*...
  (-sin(3*(Itheta-phi)) - 1i*cos(3*(Itheta-phi))))...
  ./(1./(1i*3*w*Cres)+1i*3.*w*Lres+Rpar);
Ipk3 = abs(I3);

%the the duty ratio is too small or too large
%(arbitrary imposed limits)
if(percent_delta>=0.99 || percent_delta < 0.3)
    continue;
else
    %make the actual Ipk out of admittance. This is
    %the total available resonant current to go into
    %the secondary
    Itotal = abs(Y)*Vinput*sin(delta);

    %Calculate the resonant current after the current
    %divider that is the magnitizing inducrance (that
    %is part of the available goes into the core of
    %the transformer
    Ipk = Itotal.*(Zu./(Zu+1./abs(Y)));
    Iu = Itotal-Ipk;

    %compute output power
    Pactual = (Vout*Ipk*cos(phi)/pi);
    %see if it is within accuracy limit
    if(abs(Pactual-Pout_rqrd*eta)<cc*Pout_rqrd*eta)
         %compute the final Q
         Q = 1./Rload*sqrt(Lres./Ceq);
         %compute the input current
         Iin = Pout_rqrd./Vin(vi);

     %----Supplemental Block-----
     %If it is desired to move the resonant tank
     %onto the primary side, this will aid in
     %determining the voltages V1 and V2 in setup:
     %  __----__---Cres---Lres---______-----
     %  FB    V1                 _Xfrmr_   V2
     %  ----------------------/       \---
     % V1 and V2 magnitude and phase are computed
     %This needs values that are primary referenced
     %Cres as seen on the primary
     Cres1 = Cres*TR^2;
     %Ipk as seen on the primary
     Ipk1 = TR*Ipk;
     Ac = Ipk1./w./Cres1;
     A1 = Vin(vi)*4/pi*sin(delta);
     ThetaC = -Itheta-pi/2;
```

```
    %the fundamental component of the full bridge
    %output waveform represents the reference phase

%theta1 = angle of the fundamental of voltage waveform
%of the primary side of the would-be informer (L+Xfrmr)
Theta1 = atan((A1+Ac*sin(ThetaC))./(A1+Ac*cos(ThetaC)));
%theta2 = angle of the fundamental of voltage waveform
%of the secondary side of the would-be informer (L+Xfrmr)
        Theta2 = Itheta-phi;
        %--end--Supplemental Block-----

        if(lossCalc ~= 0)
        %calculate loss
        [Ploss,FBtotal,FBcond,FBsw,FBdiode,CCtotal,...
        CCcond,CCsw,CCdiode,Ltotal,Lwind,Lcore,Xtotal,...
        Xwind1,Xwind2,Xcore,Xpar,LossParasitic]...
        =loss4([f,3*f],[Ipk,Ipk3],Iu,percent_delta,...
        Vin(vi),Vout,Lres,TR,N,Nstrands,ml,wirediam,...
        wirearea,klayer,Al,DCesr,30);
        if(Ploss==-1)
           aa = -1;
           fave= -1;
           Ipkave= -1;
           Ploss = 1000;
        end
        if(Ploss > 0.4*Pout_rqrd*eta)
        %if the estimate loss is more that 40 % of
        %delivered power, skip this (need to aim to
        %deliver more power)
           continue;
        end
        else  %do not compute loss
            Ploss=0;FBtotal=0;FBcond=0;FBsw=0;...
            FBdiode=0;CCtotal=0;CCcond=0;CCsw=0;...
            CCdiode=0;Ltotal=0;Lwind=0;Lcore=0;...
            Xtotal=0;Xwind1=0;Xwind2=0;Xcore=0;...
            Xpar=0; LossParasitic=0;
        end
       %real is taken for prettier output. Since
       %imaginary crept into the equations, i is
       %carried thru. But it is always 0000*i so this
       %is fine
       opptstemp = real([opptstemp;...
        [PercentPower(pm),Vin(vi),Vout,eta,...
        Pout_rqrd*eta,f,Fres,Q,percent_delta,...
        phi*180/pi,FB_ZVS,Itheta*180/pi,...
        Itotal,Iu,Ipk,Ipk3,Iin,Ceq,Rload,...
        Theta1, Theta2, Theta2-Theta1,...
        Ploss+Lcore,FBtotal, FBcond, FBsw, FBdiode,...
```

```
                    CCtotal, CCcond, CCsw, CCdiode,...
                    Ltotal, Lwind, Lcore, Xtotal, Xwind1,...
                    Xwind2, Xcore, Xpar,LossParasitic],...
                    phi_min, phi_s, Pactual, 1-Ploss./(Pactual),...
                    Pactual/(Pout_rqrd*eta)]);
                    %this will pick the lowest frequency that works
                       break;
                   else
                       continue;
                   end
               end
           end
           %sort by column 23 (loss) in asceding order
           if(~isempty(opptstemp))
               opptstempsorted = sortrows(opptstemp,-44);
                opSall = [opSall; opptstempsorted];
                opSmax = [opSmax; opptstempsorted(1,:)];%
           end
       end
   end
end

%based on how many average powers are calculated, pick the CEC
%weighting coefficients, lumping the lowest ones togehter if needed
if(length(PercentPower) == 4)
    coeffs = [.05 .53 .21 .21];
elseif(length(PercentPower) == 5)
    coeffs = [.05 .53 .21 .12 .09];
else
    coeffs = [.05 .53 .21 .12 .05 .04];
end
%sampled time points in a quarter of a line cycle
time = VoutPhaseAngles;
vpts = length(PowerPercentLineCycle);
t = [0;time'];
[r,c] = size(opSmax);
effS = [];
%proceed only if every voltage at every power level was calculated
if(r == vpts*length(coeffs))
    for p = 1:length(PercentPower)
        for k = 1:vpts
            elost = trapz(t,[0;opSmax(1+(p-1)*vpts:p*vpts,23)]);
            edeliv = trapz(t,[0;opSmax(1+(p-1)*vpts:p*vpts,43)]);
             %eff = 1-integral(Plost)/integral(Pin)
             %both itegrate over 1/4 of line cycle
            effS(p,1) = 1 - elost/edeliv;
        end
    end
end
```

```
%this check is probably not needed
if(length(eff) ~= length(coeffs))
    cecEffS = -1;
else
    cecEffS = coeffs*effS;
end
```

Appendix D

Experimental setup scripts

D.1 Summary

This appendix deals with the experimental testing of the prototype. Specifically, the software, which was used to supply timing signals to the inverter, and the scripts, which were used to measure efficiency, are presented and explained briefly.

Low power digital signals are sent to the gate drivers in the inverter, which in turn switch the MOSFETs. An example of these signals is shown on Fig. 5.11 in light yellow and purple, below the analog waveforms. The timing of these signals is controlled by a field programmable gate array (FPGA). The FPGA model which was used is the Xilinx xc3s500e and it is part of the Spartan-3E Starter Kit[1] (development kit) by Digilent Inc. The development board connects to a computer running Xilinx ISE Webpack 11.1 on Windows XP x86. The FPGA is initially loaded with the code, described in section D.2, via USB JTAG programming. Since the FPGA uses volatile memory, the software must be loaded using Xilinx software every time that the board is powered on. The code allows the FPGA to communicate with the computer via the serial interface of the development board. This allows any program to send commands to the FPGA if connected to it through a serial (USB) interface (this would not be possible via the JTAG interface even though both are connected physically through USB).

D.2 FPGA software description

Besides allowing communication (sending and receiving) through the serial interface, the FPGA software[2] implements an arbitrary number of pulse width modulation

[1] http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,792&Prod=S3EBOARD
[2] The Verilog code was written be Brandon Pierquet

Register Number	Format	Description
0	16 bit value Bit 0: 1=enable, 0=disable Bit 1: Value when disabled Bit 15: 1=reset, 0=not reset	Status register enables/disables the PWM channel and sets the default (disable) value; reset clears all registers
1	16 bit value all bits: off compare	Equal to value of 'cnt' at which channel output changes to 0
2	16 bit value all bits: on compare	Equal to value of 'cnt' at which channel output changes to 1

Table D.1: Description of PWM channel registers.

(PWM) channels. These are the basic timing blocks of the system. Each PWM channel outputs either a logical 1 or 0 and its operation is specified by three 16-bit registers, numbered 0-2, assigned to each PWM channel. The function of each register is specified in Table D.1 for PWM channels which are numbered starting from 1. PWM channel number 0 is reserved for the centralized counter and its 3 registers serve a different function. The 16 bit register 0 determines the enabled/reset state of all other PWM channels, with bit 1 being the enable bit (1=enable,0=disable) and bit 15 being the reset bit (1=held in reset). Note that reset clears all register to their original state, so ending reset requires sending new values into all PWM channels. The same is not true with disable: when disabled, all channels remain in their pre-specified default state and continue timing when enable again. Register 1 of PWM channel 0 is the 16 bit value 'cnt', which determines the switching frequency, as given by the equation (D.1). The development board has a 50 MHz crystal oscillator which is used as the main clock. Register 1 of PWM0 resets to 0 after every 'cnt' ticks of the main clock. The 'on' and 'off' values of all other PWM channels are compared to the value of 'cnt' to determine their instantaneous output. The timing diagram for each PWM channel is shown in Fig. D.1. Register 2 of PWM0 allows for prescaling of the clock: setting bit 0 to 1 enable the prescalet and bits 7:4 determine the prescaler value, with 000 being equal to half of the frequency. Prescaling was not needed in this work, as the 16 bit counter provided sufficient frequency range.

$$\text{cnt} = \left\lfloor \frac{50\text{MHz}}{f_{switching}} \right\rfloor \tag{D.1}$$

Each logical PWM channel may be tied to output to a pin on the development board. This routing is specified in the .ucf constraints file. In addition to PWM0, eight PWM channels were implemented - one for each MOSFET in the inverter. Sending status and timing updates (enabling, resetting, changing frequency, deadtimes, rela-

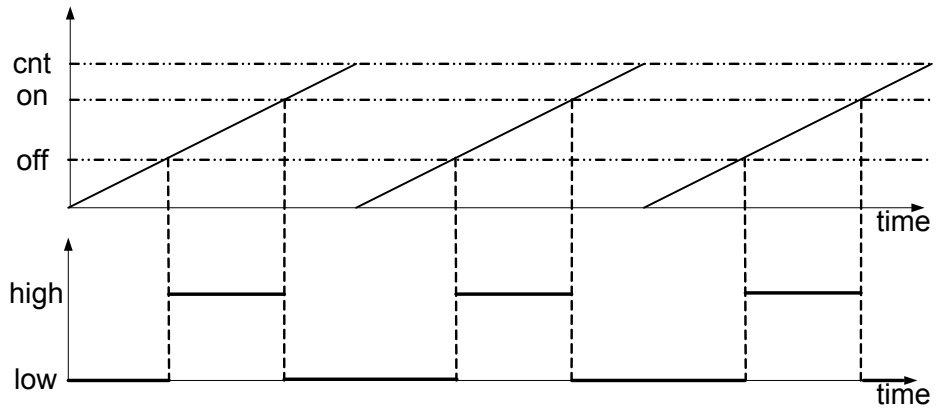


Figure D.1: The timing diagram for PWM0 channel 1 is shown on top. The timing diagram for PWM channels 1 and higher is shown on the bottom. Note that 'off' may precede 'on'.

tive phases) to PWM channels is accomplished via a serial connection. Each message comprises a string of format 'xAAnBBBB'. The value 'x' may be either 'w' for write or 'r' for read. 'AA' is the hexadecimal value for the PWM channel number and 'n' is the register of that PWM channel, to which the hexadecimal value 'BBBB' will be written. 'BBBB' is not needed in the read mode, when the content of the register 'AAn' is sent back from the FPGA. Given the setup of the PWM channels, the timing of FPGA may be changed while the inverter is running. This was confirmed to be safe across a wide range of operating powers.

D.3 MATLAB scripts for FPGA

The FPGA connected to a USB port is accessible from MATLAB to provide the correct timing. The script 'connect2FPGA.m' establishes the connection and sets up the PWM channels. The script 'fpgatesting.m' is the main script, which is used to test desired operating points on the inverter. It relies on the output of 'controlMain.m' (see Appendix A) to lookup the appropriate control inputs for operating points which are generated by 'fg.m'. The function 'getItAll.m' interlinks the three. Function 'fpga2.m' is used to send a given set of control inputs to the FPGA. Finally, 'switchHB.m' manually changes the state of PWM channels to a certain value, and 'fpgaON.m' and 'fpgaOFF.m' enable and disable the FPGA timing respectively. The calls to take efficiency measurements from 'fpgatesting.m' are directed to the scripts of the next section. The file 'points_in_line.m' illustrates the process of selecting operating points using various spacing metrics.

connect2fpga.m

```
function fpga_conn = connect2fpga()
%FPGA development board is connected to a USB port of the computer. This
%function initialized the connection to that USB port. It returns the fpga
%file handle, sets the frequency to 200 kHz and turns on the default
%channels of the CC (the nonmodulating side). See switchHB.m for which
%channels are the default.
fpga_conn = serial('COM5','BaudRate',115200,'DataBits',8);
fopen(fpga_conn);
fhex = dec2hex((50e6/200e3),4);
fprintf(fpga_conn,'w0020000');     %disable prescaler
fprintf(fpga_conn,'w0000000');     %disable all FPGA timing
fprintf(fpga_conn,['w001',fhex]);  %enter the frequency
fprintf(fpga_conn,'w0000001');     %enable all FPGA timing

%turn on the non-modulating half of CC
switchHB(fpga_conn, -1, 1);
```

fpgatesting.m

```
%get the fpga connection handle
if(~exist('fpga_conn','var'))
    fpga_conn = connect2fpga();
end

%conventions:
%CC HS start time as a function of the switching period
%dta, dtb = deadtime: A-after high side. B-after low side. *Probably
%can be set equal. Format [fbLEAD, fbLAG, CC]. Normalize to T. (either
%have %this as a fraction of T (0,1) or time in sec*f (=time/T). EX:
% dta = [260e-9, 220e-9, 750e-9]*f;
% dtb = [260e-9, 220e-9, 750e-9]*f;

%if increasing frequency on the fly - do it in 10 khz increments
%if decreasing - do it <5 khz steps. Otherwise the sudden changes in
%power spike result in unpleasent crackling noises. This has never led
%to failure, but conceivably could.

%%
% some values which may be used to test the connection
% Vin = 25 Vuot = 313        149 W out
% f = 94970.971;       %for clock imprecision
```

```
% delta = 0.78;
% CC_hs_start = 0.568;
% dta = [180e-9, 160e-9, 710e-9]*f;
% dtb = [180e-9, 160e-9, 710e-9]*f;
% useDelay = 0;
%
% dta = [160e-9, 160e-9, 500e-9]*f;
% dtb = [160e-9, 160e-9, 600e-9]*f;

%suppress the warning due to GPIB port time out (this is only used in
%testing efficiency)
warning('off','MATLAB:serial:fread:unsuccessfulRead');


%oppsRe is the output of controlMain.m. It stores all of the
%calculated timing information. PCT can be set to a different array
PCT = opptsRe;

% IPP60R250CP MOSFET DATA. It is here to interpolate for equivalent
% capacitance so that an LTSpice model can use this value to verify
% circuit performance
VdsCC = [1 10 37 50 75 100 125 150 175 200 250 300 350 400 500];
CdsCC = 1e-12*[1e4 3e3 1e3 2e2 80 58 50 48 47 45 43 41 40 39 38];
CeffCC = 1./VdsCC.*cumtrapz(VdsCC,CdsCC);

%run the script fg.m to compute the desired operating points to
%test. This will produce the array PCT
fg;

%pick a row of PCT which you want to test
ROW = 36;

%pick the desired accuracy of power delivery. A set of control inputs
%will be selected such that the inverter model predicts that it will
%deliver CO*desired power of row# ROW of PCT. If the bound is too
%small, the correct operating point might not be found and it might
%need to be loosed. Always above 1.
CO = 1.01;

%insert the minimum full-bridge ZVS margin that you want
zvsMarg = 0.2;  %RADIANS

%obtain the input parameters given these conditions
[f, delta, CC_hs_start, dtF, dtC, PowerIn, PowerOut]=getItAll(ROW,...
    PCT,PTT, CO,zvsMarg);
if(f==-1)
    error('No operating point was found');
end

%adjust the cc high side start (see chapter 4 for why)
```

```
CC_hs_start = 1.5*dtC*1e-9*f+CC_hs_start;
%adjust the deadtime (see chapter 4 for why)
dtC = 2*dtC;

%find the input and output voltages to set
setVout1 = PTT(ROW,3);

Ccc = interp1(VdsCC,CeffCC,setVout1);

fprintf(['freq=%.2f delta=%.3f ccStart=%.3f vout=%.1f V Pin=%.2f '...
    'Pout=%.2f W, dtFB=%i ns dtCC=%.2f ns Ccc=%.2f nF \n'],f*1e-3,...
    delta, CC_hs_start, setVout1,PowerIn, PowerOut,dtF,dtC,Ccc*1e9);

disp(['Set Vin = ' ,num2str(PTT(ROW,2)),', Vout = ' ,...
    num2str(setVout2),' PoutDesired =  ', num2str(PTT(ROW,4))]);
f = f/.999;

%FOR NEGATIVE LINE CYLCE - add 0.5 to CC HS switching time
CC_hs_start=CC_hs_start+.5;

LSadj = 0; %to subtract from FB low side deadtime. could be 20ns.
            %Empirical adjustment
%deadtimes
dta = [dtF, dtF-LSadj, dtC]*1e-9*f;
dtb = [dtF, dtF-LSadj, dtC]*1e-9*f;
useDelay = 1;
delayONstr = useDelay*[70, 70, 70, 70, 260, 260];       %rise delay, ns
delayOFFstr = useDelay*[100, 100, 100, 100, 290, 290];  %fall delay, ns
delayON = 1e-9*delayONstr*f;                            %rise delay
delayOFF = 1e-9*delayOFFstr*f;                          %fall delay

%directory where to store the efficiency data
dir = 'may18';
%generate the string which describes the FPGA inputs
FPGAstring = [datestr(now,'dd-mmm-yyyy-HH-MM-SS'),'-',num2str(fix(f)),...
    '-',num2str(delta),'-',num2str(CC_hs_start),'-',...
    num2str(fix(dta(1)/f*1e9)),'-',num2str(fix(dta(2)/f*1e9)),'-',...
    num2str(fix(dta(3)/f*1e9)),'-',num2str(fix(dtb(1)/f*1e9)),'-',...
    num2str(fix(dtb(2)/f*1e9)),'-',num2str(fix(dtb(3)/f*1e9)),'-',...
    num2str(delayONstr(1)),'-',num2str(delayONstr(2)),'-',...
    num2str(delayONstr(3)),'-',num2str(delayONstr(4)),'-',...
    num2str(delayONstr(5)),'-',num2str(delayONstr(6)),'-',...
    num2str(delayOFFstr(1)),'-',num2str(delayOFFstr(2)),'-',...
    num2str(delayOFFstr(3)),'-',num2str(delayOFFstr(4)),'-',...
    num2str(delayOFFstr(5)),'-',num2str(delayOFFstr(6)),'-',...
    num2str(PowerIn),'-',num2str(PowerOut)];

% put some basic checks here to make sure that no crap values pass thru
if(f > 500e3 || f < 30e3)
   error('Check frequency');
```

```
end
if(delta > .9 || delta < .1)
   error('Check pulse width');
end
if(CC_hs_start > 0.999 || CC_hs_start < 0)
   error('Check CC');
end
if(max(dta) > 0.45 || max(dtb) > 0.45 || min(dtb) < 0||  min(dta) < 0)
   error('Check deadtimes');
end
if(max(delayON) > 0.45 || max(delayOFF) > 0.45 || min(delayON) < 0 ||...
    min(delayOFF) < 0)
   error('Check delays');
end

%program the FPGA
fpga2(fpga_conn, f, delta,CC_hs_start, dta, dtb, delayON, delayOFF);

%turn off the fpga
fpgaOFF(fpga_conn);
fprintf('\nTrying full input and full output voltages\n')
%set input and output voltage
%this is power supply specific
setVin(32.5,4);
%for output power supply, scale the sent value by 100
setVin(setVout1/100,6);

%Wait for user to initiate the test
input('Press Enter to start');
%turn on the FPGA
fpgaON(fpga_conn);

%gating power supply current is measured by hand
sc = input('Press enter to stop or chip current continue:', 's');

if(isempty(sc))
    fpgaOFF(fpga_conn);
else
    PoutMeas=takedata([dir,'\\full-power-Row-v2-',num2str(ROW),'-',...
           num2str(PTT(ROW,4)),'-',FPGAstring,'-10-',num2str(sc),...
           '.txt'],10);
    input('Press Enter to stop');
    fpgaOFF(fpga_conn);
end

% To turn on and off the FPGA timing
% fpgaON(fpga_conn);
% fpgaOFF(fpga_conn);

if(0)
```

```
%%
    %to turn off and then on the non-modulating leg
    switchHB(fpga_conn, -1, 0)
    switchHB(fpga_conn, -1, 1)
end
```

getItAll.m

```
function [f, delta,cc_start,dtF, dtC,PowerIn,PowerOut]=getItAll(pRow,...
    PCT, PTT, CO, zvsMin)
%find the set of control inputs from array PCT, which satisfy the
%requirements given in row pRow of array PTT. See controlMain.m and
%fg.m for the format of those 2 arrays

oVout = 2;
oVin = 1;
oPout = 4;
clear TT TT2 TT3 TT4 TT5
%narrow down by Vin
a = find(((abs(PCT(:,oVin) - (PTT(pRow,2)))) < 1));
TT(1:length(a),:) = PCT(a,:);
%narrow down by Vout
b = find(abs(TT(:,oVout) - (PTT(pRow,3))) < 1);
TT2(1:length(b),:) = TT(b,:);
% CO = 1.03; %CO for the tolerance on power
c = find(TT2(:,oPout) >= PTT(pRow,4));
TT3(1:length(c),:) = TT2(c,:);
d = find(TT3(:,oPout) <= CO*PTT(pRow,4));
TT4(1:length(d),:) = TT3(d,:);
oZVS = 10;
c = find(TT4(:,oZVS) > zvsMin);
TT5(1:length(c),:) = TT4(c,:);
if(isempty(TT5))
    f=-1;
    delta =-1;
    cc_start=-1;
    dtF=-1;
    dtC=-1;
else
    f = TT5(1,6);
    delta = TT5(1,5);
    cc_start = 1.0*TT5(1,14);
    dtF = TT5(1,11);
    dtC = 1.43*TT5(1,12);
    PowerIn = TT5(1,3);
    PowerOut = TT5(1,4);
```

```
end
```

fg.m

```
%Calculates the operating points to be tested
%inputs:

%set the positions along the line (in degrees, from 0 to 90) to test
Vpos = (10:10:90)./90;
Vinput = [32.5];    %set the input votlages
PercentPower =[.3 .5 .75 1]; %set the power levels
Pave_max = 175;
Vrms = 240;
%see array PTT for output
PowerPercentLineCycle = sin(Vpos*pi/2).^2;
Rlprime = 66./PercentPower;
LPP = length(PercentPower);
pl = length(PowerPercentLineCycle);
LV = length(Vinput);
PTT = zeros(LPP*LV*pl,6);
row = 1;
for pm = 1:LPP
    Pout_max = 2*Pave_max.*PercentPower(pm);
    Rldprime = Rlprime(pm);
    LineCyclePosition = sqrt(PowerPercentLineCycle);
    Vline = LineCyclePosition.*Vrms*sqrt(2);
    Pout = PowerPercentLineCycle.*Pout_max;
    for vi = 1:LV
        for np = 1:pl
            Vout = Vline(np);
            %Perecent average power, Vin, Vout, Pout
            Iout = Pout(np)./Vout;
            Iin =  Pout(np)./Vinput(vi)/0.95;       %0.95 - some efficiency
            PTT(row,1:6) = [PercentPower(pm), Vinput(vi), Vout, Pout(np), Iin, Iout];
            row = row+1;
        end
    end
end
```

fpga2.m

```
function fpga2(fpga_conn, f, delta, CC_hs_start, dta, dtb, delayON, delayOFF)
```

```
%This function writes the timing information to the FPGA. The PWM channel
%numbers are hard coded into this file. Some simple error checking is
%performed (e.g. no negative times, no delays exceed 'reasonable' bounds.
%All timing is referenced to the center of the full-bridge.
%fpga_conn = handle to the FPGA connection
%f = frequency, Hz
%delta = fraction (0 to 1) of the full-bridge pulse width. 1 = half of
%switching period is positive, half is negative. 0.5 = a quarter is
%positive, a quarter is negative, and so on.
%CC_hs_start = start of high side of the cycloconverter, normalized to the
%switching period.
%dta, dtb = vectors of deadtime, each 1x3: A-after high side.
%B-after low side. Each index corresponds to a half bridge. All times are
%normalized to switching period (e.g. 0 to 1, but cannot exceed 0.45 as a
%sanity check). Order of HBs: [FullBridge Leading, FullBridge Lagging, Cyc
%loconverter High Side]
%delayOn, delayOFF - delays for on and off transitions for each half bridge
%same rules as deadtimes.
%Default PWM output channels:
%4 = HS, leading leg of FB
%3 = LS, leading leg of FB
%2 = HS, lagging leg of FB
%1 = LS, laggng leg of FB
%5 = HS, positive leg of CC
%6 = LS, negative leg of CC
%7 = HS, positive leg of CC
%8 = LS, negative leg of CC

%*********************************************
%** SPECIFY ALL TIMING AS NORMALIZED TO T
%*********************************************
%PWM output channel numbers:
chan_num = [3,4,1,2,7,8];

%number of half bridges. 3 and not 4 as one of the CC half-bridges
numHB = 3;

t0 = 0.25*delta;    %offset everythign to center the full bridge
                    %waveform at 0

Nodd = 1:2:2*numHB; %compute array indeces
Neven = Nodd + 1;

%Compute the array of start times for high sides of each half bridge
start(Neven) = [-t0, t0, -t0+CC_hs_start];

CLK = 50e6;     %FPGA clock is 50 MHZ
clk_cnt = fix(CLK/f);
fhex = dec2hex(clk_cnt,4); %want 4 0digits
```

```
% set the frequency LAST! otherwise it might take two tries to reset the
% counters

if(~isempty(find(dta>0.45,1)) || ~isempty(find(dtb>0.45,1)))
    warning('component:mnemonic','Deadtime might be too large!');
end

%each array is up to here is length numHB. From here on, each array is of
%length 2*numHB
%the ONTIME (duty cycle) of each half bridge. want this to be close to 0.5
%this will give the same value for each half bridge
thetaOT(Neven) = 0.5*(1 - dta - dtb);
thetaOT(Nodd) = 0.5*(1 - dta - dtb);
start(Nodd) = start(Neven)+thetaOT(Neven)+dta;
thetaON = start;
thetaOFF = thetaON + thetaOT;
thetaON = thetaON - delayON;
thetaOFF = thetaOFF  - delayOFF;
ON = mod((thetaON),1);
OFF = mod((thetaOFF),1);

%theta ON(x)=position in the switching period when channel x turns on
%theta OFF(x)=position in the switching period when channel x turns off
%position is really [0,1] not [0,2pi)
ON = floor(mod((clk_cnt*thetaON),clk_cnt));
OFF = floor(mod((clk_cnt*thetaOFF),clk_cnt));

if(sum(ON < 0) || sum(ON > clk_cnt) || sum(OFF < 0) || sum(OFF > clk_cnt))
    fpgaOFF(fpga_conn);
    error('Error in calculating ON/OFF times. FPGA is now disabled.')
end
channels = 2*numHB;
enable = ones(1,channels);          %enable the channels
reset = zeros(1,channels);          %reset = OFF
default = zeros(1,channels);        %defaults to 0 / off

for n = 1:channels
   if(n<10)
      chnum = ['0',num2str(chan_num(n))];
   else
      chnum = num2str(chan_num(n));
   end
%enable the channel: chNum -0(reg)-X(8/0=reset/not)-00-X(bit 1 = default)
   reg0 = ['0',num2str(8*reset(n)),'00',num2str(2*default(n)+enable(n))];
   reg1 = ['1',dec2hex(OFF(n),4)];
   reg2 = ['2',dec2hex(ON(n),4)];

   wr0 = ['w',chnum,reg0];
   wr1 = ['w',chnum,reg1];
   wr2 = ['w',chnum,reg2];
```

```
   fprintf(fpga_conn,wr0);
   fprintf(fpga_conn,wr1);
   fprintf(fpga_conn,wr2);
end
fprintf(fpga_conn,['w001',fhex]);  %enter the frequency
disp('Wrote to FPGA!');
```

switchHB.m

```
function switchHB(fpga_conn, chan_num, on)
%changes the state of a half bridge to value of on (either 1 for on or 0
%for off). chan_num are the PWM output channels [LowSide, HighSide]. LS is
% toggled first. fpga_conn = handle to fpga connection, obtained using
%connect2fpga.m for example.
%Set chan_num=[-1,-1] to use the default PWM channels (5,6) (currently it
%is the 'positive' half of the cycloconverter

%set the defaults
if(chan_num(1) == -1)
    chan_num = [5,6];
end

for n=1:length(chan_num)
   if(chan_num(n)<10)
      chnum = ['0',num2str(chan_num(n))];
   else
      chnum = num2str(chan_num(n));
   end
   if(on == 1) %if turn on
     reg0 = ['0','0002'];
     reg1 = ['1',dec2hex(2222,4)];
     reg2 = ['2',dec2hex(2223,4)];
   else %if turn off
     reg0 = ['0','0000'];
     reg1 = ['1',dec2hex(2223,4)];
     reg2 = ['2',dec2hex(2222,4)];
   end

   wr0 = ['w',chnum,reg0];
   wr1 = ['w',chnum,reg1];
   wr2 = ['w',chnum,reg2];
   %enable the channel: chNum -0(reg)-X(8/0=reset/not)-00-X(bit 1 = default)
   fprintf(fpga_conn,wr0);
   fprintf(fpga_conn,wr1);
   fprintf(fpga_conn,wr2);
```

```
   disp(['Wrote: ',wr0,'  ',wr1,'  ',wr2]);
   pause(0.1);
end
```

fpgaON.m

```
function fpgaON(fpga_conn)
%given a handle to the FPGA connection, enable all FPGA timing signals
fprintf(fpga_conn,'w0000001');
```

fpgaOFF.m

```
function fpgaOFF(fpga_conn)
%given a handle to the FPGA connection, disable all FPGA timing signals
fprintf(fpga_conn,'w0000000');
```

points_in_line.m

```
%this script computes and visually display points along the AC line cycle
%such that they are equally spaced in energy delivery.
%This is all done for unity power factor, for an arbitrary number of points
%method: assuming P proportional to V^2 and E = INT(Pdt).
%look at the total change in Energy (last-first), divide it into N steps,
%and then find time indeces where those values occur in energy. From there,
%back track where along power curve to test (and thus voltage)

%number of points to pick
N = 8;

%Assuming current is proportional to voltage^2 (PF = 1)
n = 0:0.0001:pi/2;
Energy = cumtrapz(n,sin(n).^2);
Power = sin(n).^2;
Voltage = sin(n);
E0 = Energy(1);               %should be 0 if V is in phase with I
Etotal = Energy(end) - E0;    %total energy delivered
                      %break up area in 4 equal energy regions
dE = Etotal/N;
```

```
ecurrent = E0;
clear ind ind2

for nn = 1:N
   temp = find(Energy>=ecurrent);
   ind(nn) = temp(1);
   ecurrent = ecurrent+dE;
end
ind(N+1) = length(n);
time = n(ind);
%%
figure;
plot(n,Voltage,n,Power,n,Energy);
legend('V = sin(t)','P = sin^2(t)','E = \int Pdt', 'Location','NorthWest')
hold on
%plot the voltages at which the energy made a step
for nn=1:N+1
    line([time(nn);time(nn)],[0;Voltage(ind(nn))],'LineStyle','--');
end

%First display the positions in the line cycle at which the energy
%delivered increases by 1/N*total
disp(['For increments in energy of every 1/',num2str(N),' of total']);
disp(['Angles in the line cycle:']);
n(ind)
disp(['Voltages in the line cycle:'])
240*sqrt(2)*Voltage(ind)

%to approximate using the mid-way point, find the voltages in the middle of
%each of the above.
ind2 = diff(ind);
ind2 = ind(1:N)+fix(0.5*(ind2));
time2 = n(ind2);
disp(['Angles in the line cycle to test (half way in between each):'])
n(ind2)
disp(['Voltages in the line cycle:'])
240*sqrt(2)*Voltage(ind2)

%plot the 'half way points' in bold
for nn=1:N
    line([time2(nn);time2(nn)],[0;Voltage(ind2(nn))],'LineWidth',3);
end
hold off
```

D.4 MATLAB scripts for efficiency measurements

The function 'takedata.m' gets called to collect a set of efficiency measurements. It makes calls to 'setupDMMs.m', 'getGPIBhandle.m', 'mygpib.m', and 'readEff.m'. The connection to GPIB instruments is established via a USB to GPIB converter from Prologix[3].

takedata.m

```
function Pout=takedata(filename,timesec)
%This function take efficiency data for timesec seconds and outputs the
%results in the file specified by filename. It calls mygpib.m and
%setupDMMs.m, which must be in the same folder. Multimeter GPIB addresses
%are hardcoded on line 14 and get sent to all the functions which use them
%from here.
%Returns: Pout = measured output power
fout = fopen(filename,'a');
fprintf(fout,['---',date,'---',num2str(timesec),'secs\n']);
disp(['Wait ',num2str(timesec),' secs while data is taken']);
fclose(fout);
%GPIB addresses: Vin, Iin, Vout, Iout multimeters
addr = [16,15,23,26];
%configure the multimeters and the handle to the instruments
sp = setupDMMs(addr);
tic;
a = toc;
while(a < timesec)
    a = toc;
    mygpib(0,filename,sp,addr);
end
%reads in and displays the average statistics for the efficiency files
%avearray = numsamples, ave for each column
%temparray = could return the array if needed
%returnData = 0 to return [], non zero to return the read in array
[avearray, duration, temparray] = readEff(filename, 0);
Pout = avearray(7);
%display the results
disp(['Took ',num2str(avearray(1)),' readings. Results: { Pin, Pout, Eff } = { ',...
    num2str(avearray(6)), 'W, ',  num2str(Pout),'W, ',num2str(avearray(8)), ' }']);
fclose(sp);
```

[3] http://www.prologix.biz

setupDMMs.m

```
function sp = setupDMMs(addresses)
%sets up the multimeters to be ready for triggering and measurements.
%all multimeters, except for a fixed address 26 are HP34401A or HP34410A.
%Address 26 is a Keithley TRMS 179 meter, with a static GPIB Address and
%only push button settings
%addresses = vector of GPIB addresses:
%address(1) = Vin
%address(2) = Iin - via a shunt, so set to voltage
%address(3) = Vout
%address(4) = Iout - typically address 26
%call to open the GPIB port
sp = getGPIBhandle(6,0.7);
%configure the Prologix GPIB box
fprintf(sp, '++eoi 1'); %turn on end-of-line character
fprintf(sp, '++eos 2'); %send a line-feed(ASCII 10)character at end of line
fprintf(sp, '++auto 0');
% fprintf(sp,'*RST'); %reset the meter to its power-on state
% fprintf(sp,'*CLS'); %clear the status registers
fprintf(sp,['++addr ',num2str(addresses(1))]);
fprintf(sp,'CONF:VOLT:DC');
fprintf(sp,'VOLT:DC:RANG:AUTO ON');
fprintf(sp,'SENS:VOLT:DC:RES MAX');
fprintf(sp,'SENS:VOLT:DC:NPLC 1');  %before: MIN
fprintf(sp,'TRIG:SOUR BUS'); %bus triggering
fprintf(sp,'TRIG:COUN 1'); %one trigger per init cycle
fprintf(sp,'TRIG:DEL:AUTO ON'); %automatic delay after trigger
fprintf(sp,'SAMP:COUN 1'); %samples per triggerr
fprintf(sp,['++addr ',num2str(addresses(2))]);
fprintf(sp,'CONF:VOLT:DC');
fprintf(sp,'VOLT:DC:RANG:AUTO ON');
fprintf(sp,'SENS:VOLT:DC:RES MAX');
fprintf(sp,'SENS:VOLT:DC:NPLC 1');
fprintf(sp,'TRIG:SOUR BUS'); %bus triggering
fprintf(sp,'TRIG:COUN 1'); %one trigger per init cycle
fprintf(sp,'TRIG:DEL:AUTO ON'); %automatic delay after trigger
fprintf(sp,'SAMP:COUN 1'); %samples per trigger
fprintf(sp,['++addr ',num2str(addresses(3))]);
% fprintf(sp,'*RST'); %reset the meter to its power-on state
% fprintf(sp,'*CLS'); %clear the status registers
fprintf(sp,'CONF:VOLT:DC');
fprintf(sp,'VOLT:DC:RANG:AUTO ON');
fprintf(sp,'SENS:VOLT:DC:RES MAX');
fprintf(sp,'SENS:VOLT:DC:NPLC 1');
fprintf(sp,'TRIG:SOUR BUS'); %bus triggering
fprintf(sp,'TRIG:COUN 1'); %one trigger per init cycle
```

```
fprintf(sp,'TRIG:DEL:AUTO ON'); %automatic delay after trigger
fprintf(sp,'SAMP:COUN 1'); %samples per trigger

% setup old current meter - addr 26,
fprintf(sp,['++addr ',num2str(addresses(4))]);
fprintf(sp,'T5');       %1 measurement, trigger on X
```

getGPIBhandle.m

```
function sp = getGPIBhandle(comPort, timeout)
%Returns sp - the handle to the serial port which is connected to the
%prologix GPIB to USB box: www.prologix.biz
%The multimeters and the power supplies are daisy chained together via GPIB

% delete(INSTRFINDALL)
sp = serial(['COM',num2str(comPort)]);
sp.Terminator = 'LF';
sp.Timeout = timeout;
fopen(sp);
```

mygpib.m

```
function mygpib(conf,filename,sp,addresses)
%Take 1 set of (Vin, Iin, Vout, Iout) and record it to a file
%conf = 1 or 0 based on whether or not the multimeter settings need to be
%set or if the GPIB handle has not be acquired
%filename - full (with path) file name of the file to which to append the
%measurement data
%sp - handle to the serial port to which the GPIB instruments connect (only
%meaningfull is conf is not set to 1)
%addresses - GPIB address vector for Vin, Iin, Vout, Iout in that order

%if need to obtain the GPIB handle, or if the multimeters are not set up
%yet
if(conf == 1)
    sp = setupDMMs(addresses);
end
clear Iin Iout Vin Vout
fprintf(sp,['++addr ',num2str(addresses(1))]);
fprintf(sp,'INIT');
fprintf(sp,'++trg');
fprintf(sp,'FETC?');
```

```
fprintf(sp,'++read 10');
Vin = str2num(char(fread(sp,16))');
fprintf(sp,['++addr ',num2str(addresses(3))]);
fprintf(sp,'INIT');
fprintf(sp,'++trg');
fprintf(sp,'FETC?');
fprintf(sp,'++read 10');
Vout = str2num(char(fread(sp,16))');
fprintf(sp,['++addr ',num2str(addresses(2))]);
fprintf(sp,'INIT');
fprintf(sp,'*TRG');
fprintf(sp,'FETC?');
fprintf(sp,'++read 10');
Iin = str2num(char(fread(sp,16))');
fprintf(sp,['++addr ',num2str(addresses(4))]);
fprintf(sp,'X');
fprintf(sp,'++read 12');
Iout = str2num(char(fread(sp,16))');
Iin = Iin*1000;     %1 mV/A shunt is used for input current sense
Iout = Iout*0.001;  %Keithley returns current in mA
%write to file
fout = fopen(filename, 'a');
fprintf(fout,[num2str(Iin),',',num2str(Vin),',',num2str(Iout),',',...
    num2str(Vout),',',num2str(Vin*Iin),',',num2str(Vout*Iout),',',...
    num2str(Vout*Iout./Vin./Iin),'\n']);
fclose(fout);
```

readEff.m

```
function [avearray,duration, temparray] = readEff(filename, returnData)
%reads in and displays the average statistics for the efficiency files
%avearray = number of samples, ave for each column
%temparray = could return the array if needed
%returnData = 0 to return [], non zero to return the read in array

fdata = fopen(filename, 'r');
temparray = [];
duration = [];
while 1
    line =  fgetl(fdata);
    if ~ischar(line), break, end
    commas = find(line == ',');
    if(length(commas)<2)
        duration = line;
        continue;
    end
```

```
    temp = str2num(line);
    if(length(temp)<7)
        continue;
    else
        temparray = [temparray;temp];
    end
end
[r,c]=size(temparray);
avearray = mean(temparray(1:end,:));
avearray = [r,avearray];
if(returnData == 0)
    temparray = [];
end
fclose(fdata);
```

setVin.m

```
function setVin(Vin, VinAddress)
%sets the output voltage (of the power supply at GPIB address VoutAddress
%to setVoutM. Works for HP power supplies
sp = getGPIBhandle(6,0.7);
fprintf(sp,['++addr ' num2str(VinAddress)]); %address the power supply
fprintf(sp, ['VSET ', num2str(Vin)]);  % set the voltage
fclose(sp);
```

D.5 Verilog code for FPGA

The FPGA Verilog software was written by Brandon Pierquet [21] but is included here for completeness. The files shown here are written in Verilog and may all be imported into a new project (in Xilinx Webpack for Xilinx FPGAs). The main file (and the project name) is 'serial_to_pwm.v'. The .ucf constraints file is FPGA/development board specific and needs to be edited unless and identical board is used.

serial_to_pwm.v

```
‘timescale 1ns / 100ps
module serial_to_pwm(input CLK,input CLR,input RXD,output TXD,
output [15:0]PWMOUT,output CLKOUT,output RXD_LED,output TXD_LED,
output FPGA_INIT_B);
assign FPGA_INIT_B = 1;
```

```
assign RXD_LED = ~RXD;
assign TXD_LED = ~TXD;
assign CLKOUT  = CLK;
wire deser_rx_data_valid;
wire [7:0] deser_rx_data;
// Instantiate the serial port reciever
serial_async_receiver deserial(.clk(CLK),.RxD(RXD),
.RxD_data_ready(deser_rx_data_valid),.RxD_data(deser_rx_data),
.RxD_endofpacket(),.RxD_idle() );
wire instr_cmd;
wire [15:0] instr_data;
wire [11:0] instr_addr;
wire instr_valid;
// Instantiate the protcol decoder
input_protocol_decode instr_decode (.CLK(CLK), .CLR(CLR),
.DIN_TICK(deser_rx_data_valid),.DIN(deser_rx_data),
.CMDOUT(instr_cmd),.DOUT(instr_data),.AOUT(instr_addr),.DECODED(instr_valid));
wire [7:0] tx_fifo_din;
wire [15:0] readback_data;
reg [2:0] readback_nibble_state;
reg [3:0] readback_nibble;
wire[7:0] readback_nibble_encoded;
reg readback_nibble_en;
reg [3:0] readback_data1;
reg [3:0] readback_data2;
reg [3:0] readback_data3;
reg [3:0] readback_data4;
// Instantiate the PWM root controller
pwm_root_controller pwm_control ( .CLK(CLK), .RST(CLR), .AIN(instr_addr),
.DIN(instr_data),.CMD(instr_cmd),.EXECUTE(instr_valid), .DOUT(readback_data),
.PWMOUT(PWMOUT)
);
//assign tx_fifo_din = readback_data[7:0];
wire [7:0] tx_data;
wire tx_fifo_empty;
wire tx_busy;
// Instantiate the serial port transmit FIFO
sync_fifo tx_fifo ( .din(tx_fifo_din),.wr_en(tx_fifo_we),.rd_en((~tx_busy)&
(~tx_fifo_empty)),.dout(tx_data),.full(),.empty(tx_fifo_empty),.clk(CLK),
.reset(CLR));  //reg tx_send;
// Instantiate the transmit serializer
serial_async_transmitter serout ( .clk(CLK),.TxD_start(~tx_fifo_empty),
.TxD_data(tx_data),.TxD(TXD),.TxD_busy(tx_busy));
// State machine-esque block to chop a 16bit number into 4 4-bit values
//  which are then translated into asciihex for serial transmission
always @(posedge CLK)
case(readback_nibble_state)
3'b100: readback_nibble_state <= 3'b011;
3'b011: readback_nibble_state <= 3'b010;
3'b010: readback_nibble_state <= 3'b001;
```

```
3'b001: readback_nibble_state <= 3'b000;
default:
if(instr_valid & (instr_cmd==1'b0))
readback_nibble_state <= 3'b100;
else
readback_nibble_state <= 3'b000;
endcase
always @(posedge CLK)
case(readback_nibble_state)
3'b100:
begin
readback_nibble <= readback_data4;
readback_nibble_en <= 1'b1;
end
3'b011: readback_nibble <= readback_data3;
3'b010: readback_nibble <= readback_data2;
3'b001: readback_nibble <= readback_data1;
default:
begin
{readback_data4,readback_data3,readback_data2,readback_data1}<=readback_data;
readback_nibble <= 4'h0;
readback_nibble_en <= 1'b0;
end
endcase
bin2asciihex output_encode(
.bin(readback_nibble),
.ascii(readback_nibble_encoded)
);
assign tx_fifo_we  = (readback_nibble_en | deser_rx_data_valid);
assign tx_fifo_din = (readback_nibble_en) ? readback_nibble_encoded : deser_rx_data;
endmodule
```

serial_to_pwm.ucf

```
NET "CLK" PERIOD = 20.0ns HIGH 50%;
NET "CLK" LOC = "C9" | IOSTANDARD = LVTTL;
NET "CLK" TNM_NET = "CLK";
OFFSET = IN 10 ns VALID 20 ns BEFORE "CLK";
OFFSET = OUT 20 ns AFTER "CLK";
NET "FPGA_INIT_B" IOSTANDARD = LVTTL;
NET "FPGA_INIT_B" SLEW = QUIETIO;
NET "FPGA_INIT_B" DRIVE = 4;
NET "FPGA_INIT_B" LOC = V13;
##############################################################################
# Discrete Indicators (LED)
NET "RXD_LED"        LOC = "F12"  |IOSTANDARD = LVTTL |DRIVE = 12 |SLEW = SLOW ;
```

```
NET "TXD_LED"        LOC = "E12"  |IOSTANDARD = LVTTL |DRIVE = 12 |SLEW = SLOW ;
##########################################################################
# Mechanical Switches (SW)
NET "CLR" LOC = L13 | IOSTANDARD = LVTTL;
#NET "SW<1>"         LOC = "L13"  | IOSTANDARD = LVTTL ;
#NET "SW<2>"         LOC = "L14"   | IOSTANDARD = LVTTL ;
#NET "SW<3>"         LOC = "N18"   | IOSTANDARD = LVTTL ;
##########################################################################
# Serial Ports (RS232)
NET "RXD" LOC = "R7"  |IOSTANDARD = LVTTL ;
NET "TXD" LOC = "M14" |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW;
##########################################################################
# Accessory Headers (J18, J19, J20)
NET "CLKOUT"        LOC = "C7" |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<0>"     LOC = "D7" |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<1>"     LOC = "B4" |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<2>"     LOC = "A4" |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<3>"     LOC = "D5"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW;
NET "PWMOUT<4>"   LOC = "C5"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW;
NET "PWMOUT<5>"     LOC = "A6"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<6>"    LOC = "B6"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW;
NET "PWMOUT<7>"     LOC = "E7"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<8>"     LOC = "F7"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<9>"     LOC = "F8"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
NET "PWMOUT<10>"    LOC = "E8"  |IOSTANDARD = LVTTL |DRIVE = 8 |SLEW = SLOW ;
```

pwm_root_controller.v

```
module pwm_root_controller #(parameter PWM_CHANNELS=10,
parameter CHAN_REGISTERS=3,
parameter PWM_CHAN_A_BITS=4, // Num of bits to address PWM_CHANNELS+1
parameter CHAN_REG_A_BITS=2, // Num of bits to address channel regs, 4 max
parameter REG_STATUS=3'h0,parameter REG_CMP_OFF=3'h1,parameter REG_CMP_ON=3'h2,
parameter BIT_EN = 4'h0,parameter BIT_RST = 4'hF,parameter BIT_DEF=4'h1)
(input CLK,input RST,input [11:0] AIN, //11:4 are PWM channel, 3:0
//are internal PWM register
input [15:0] DIN,input CMD,input EXECUTE,output [15:0] DOUT,
output [PWM_CHANNELS:0] PWMOUT);
// Address breakdowns for pwm channel and internal mem
wire [PWM_CHAN_A_BITS-1:0] a_chan;
wire [CHAN_REG_A_BITS-1:0] a_reg;
assign a_chan = AIN[4+PWM_CHAN_A_BITS-1:4]; // AIN[15:4] is for PWM channel
assign a_reg  = AIN[0+CHAN_REG_A_BITS-1:0];//AIN[3:0] is for channel registers
// Start RAM definition; address registers to ensure a sync-read ram
reg [15:0] ram[PWM_CHANNELS:0][CHAN_REGISTERS-1:0];
reg [PWM_CHAN_A_BITS-1:0] read_a_chan;
```

```
reg [CHAN_REG_A_BITS-1:0] read_a_reg;
wire we;
assign we = CMD & EXECUTE;
// RAM access
always @(posedge CLK)
begin
if (we) ram[a_chan][a_reg] <= DIN;
read_a_chan <= a_chan;
read_a_reg  <= a_reg;
end
assign DOUT = ram[read_a_chan][read_a_reg];
// End RAM definition
parameter REG_CNTMAX = 8'h1;
parameter REG_CLKDIV = 8'h2;
parameter CHAN_ROOT = 8'h0;
wire [15:0] PWMCNT_Q;
wire PWMCNT_TC;
wire PWMCNT_EN;
wire [15:0] CLKDIV_Q;
wire [15:0] CLKDIV_BIT;
assign CLKDIV_BIT = ram[CHAN_ROOT][REG_CLKDIV];
assign CLKDIV_RST = ram[CHAN_ROOT][REG_CLKDIV][BIT_RST];
assign PWMCNT_TC = (PWMCNT_Q==ram[CHAN_ROOT][REG_CNTMAX]);
assign PWMCNT_EN  = ram[CHAN_ROOT][REG_STATUS][BIT_EN];
assign PWMCNT_RST = CLKDIV_RST | ram[CHAN_ROOT][REG_STATUS][BIT_RST];
wire PWMCNT_CE;      // Clock divider
counter_ce_sc clock_divider (.CLK(CLK),.CLR(RST|CLKDIV_RST|PWMCNT_CE),
.CE(1'b1), .Q(CLKDIV_Q)
);
assign PWMCNT_CE = CLKDIV_BIT[0] ? CLKDIV_Q[CLKDIV_BIT[7:4]] : 1'b1;
counter_ce_sc counter (           // Shared counter for PWM channels
.CLK(CLK),.CLR(RST|PWMCNT_TC|PWMCNT_RST),.CE(PWMCNT_EN&PWMCNT_CE),
.Q(PWMCNT_Q)
);
// Output assignments and dynamic PWM module instantiation
assign PWMOUT[0] = PWMCNT_TC;
generate
genvar i;
for (i=1; i <= PWM_CHANNELS; i=i+1) begin : PWMCHANS
set_reset_pwm_gen pwmch (
.CLK(CLK),
.EN(ram[i][REG_STATUS][BIT_EN]&PWMCNT_EN),
.DEF_VAL(ram[i][REG_STATUS][BIT_DEF]),
.COUNTER(PWMCNT_Q),
.CMP_ON_IN(ram[i][REG_CMP_ON]),
.CMP_OFF_IN(ram[i][REG_CMP_OFF]),
.Q(PWMOUT[i]));
end
endgenerate
endmodule
```

serial_to_pwm_pa_ports.v

```
‘timescale 1ns / 1ps
module netlist_1_EMPTY(PWMOUT,CLK,CLR,RXD,TXD,CLKOUT,
RXD_LED,TXD_LED,FPGA_INIT_B);
  output [15:0] PWMOUT;
  input CLK;
  input CLR;
  input RXD;
  output TXD;
  output CLKOUT;
  output RXD_LED;
  output TXD_LED;
  output FPGA_INIT_B;
endmodule
```

input_protocol_decode.v

```
module input_protocol_decode(input CLK,input CLR,input DIN_TICK,i
nput [7:0]DIN,output reg CMDOUT,   // low for read cmd, high for write
output reg [15:0] DOUT,output reg [11:0] AOUT,output reg DECODED);
reg[3:0] state;
parameter IDLE=4’b0000,
RWAIT=4’b0001,
           CMD=4’b1000,
         ADDR1=4’b1001,
ADDR2=4’b1010,
ADDR3=4’b1011,
         DATA1=4’b1100,
DATA2=4’b1101,
DATA3=4’b1110,
DATA4=4’b1111;
// Continuous decoding of incoming data for CMD characters
parameter CMD_WR=1’b1;
parameter CMD_RD=1’b0;
reg iscmdchar;
reg decodedcmdchar;
always @(DIN)
begin
case (DIN)
8’h72: {iscmdchar,decodedcmdchar} = {1’b1,CMD_RD};
8’h77: {iscmdchar,decodedcmdchar} = {1’b1,CMD_WR};
default: {iscmdchar,decodedcmdchar} = {1’b0,1’b0};
```

```
endcase
end
// Continuous decoding of incoming data for DATA/ADDR characters
wire ishexchar;
wire [3:0] decodedhexchar;
asciihex2bin input_decode(
.ascii(DIN),
.ishexchar(ishexchar),
.bin(decodedhexchar)
);
// State transitions
//  -> occurr only on DIN_TICK events (which are sync with CLK)
always @(posedge CLK or posedge CLR)
begin
if (CLR == 1)
state <= IDLE;
else
if (DIN_TICK)
case(state)
IDLE: if(iscmdchar) state <= CMD;
else state <= IDLE;
CMD: if(ishexchar) state <= ADDR1;
else state <= IDLE;
ADDR1: if(ishexchar) state <= ADDR2;
else state <= IDLE;
ADDR2: if(ishexchar) state <= ADDR3;
else state <= IDLE;
ADDR3: if(ishexchar) state <= DATA1;
else state <= IDLE;
DATA1: if(ishexchar) state <= DATA2;
else state <= IDLE;
DATA2: if(ishexchar) state <= DATA3;
else state <= IDLE;
DATA3: if(ishexchar) state <= DATA4;
else state <= IDLE;
DATA4: state <= IDLE;
default state <= IDLE;
endcase
else
case(state)
ADDR3: if(CMDOUT==CMD_RD) state <= IDLE;
DATA4: state <= IDLE;
default: state <= state;
endcase
end
// State Outputs
always @(posedge CLK)
      begin
case(state)
CMD:   AOUT[11:8]  <= decodedhexchar;
```

```
  ADDR1: AOUT[7:4]   <= decodedhexchar;
  ADDR2:
begin
AOUT[3:0]    <= decodedhexchar;
end
ADDR3:
begin
DOUT[15:12] <= decodedhexchar;
DECODED      <= (CMDOUT==CMD_RD) ? 1'b1 : 1'b0;
end
DATA1: DOUT[11:8]  <= decodedhexchar;
DATA2: DOUT[7:4]   <= decodedhexchar;
DATA3: DOUT[3:0]   <= decodedhexchar;
DATA4: DECODED     <= 1'b1;
default: // includes IDLE
begin
CMDOUT       <= decodedcmdchar;
AOUT         <= AOUT;
DOUT         <= DOUT;
DECODED      <= 1'b0;
end
endcase
      end
endmodule
```

counters.v

```
module counter_ce_sc(CLK, CLR, CE, Q);
parameter WIDTH = 16;
input CLK, CE, CLR;
output [WIDTH-1:0] Q;
reg    [WIDTH-1:0] Q;
always @(posedge CLK)
begin
if (CLR)
Q <= 16'b0;
else
if (CE)
Q <= Q + 1'b1;
    end
endmodule
```

hexascii.v

```
module bin2asciihex(input [3:0] bin, output reg [7:0] ascii );
// Continuous decoding of binary values into hex chars
always @(bin)
begin
case (bin)
4'h0: ascii = 8'h30;4'h1: ascii = 8'h31;4'h2: ascii = 8'h32;
4'h3: ascii = 8'h33;4'h4: ascii = 8'h34;4'h5: ascii = 8'h35;
4'h6: ascii = 8'h36;4'h7: ascii = 8'h37;4'h8: ascii = 8'h38;
4'h9: ascii = 8'h39;4'hA: ascii = 8'h41;4'hB: ascii = 8'h42;
4'hC: ascii = 8'h43;4'hD: ascii = 8'h44;4'hE: ascii = 8'h45;
4'hF: ascii = 8'h46;default: ascii = 8'h3F; // "?"
endcase
end
endmodule
module asciihex2bin(input [7:0] ascii,output reg ishexchar,output reg [3:0]bin);
// Continuous decoding of ascii hex chars into binary values
always @(ascii)
case (ascii)
8'h30: bin = 4'h0; 8'h31: bin = 4'h1; 8'h32: bin = 4'h2;
8'h33: bin = 4'h3; 8'h34: bin = 4'h4; 8'h35: bin = 4'h5;
8'h36: bin = 4'h6; 8'h37: bin = 4'h7; 8'h38: bin = 4'h8;
8'h39: bin = 4'h9; 8'h41: bin = 4'hA; 8'h42: bin = 4'hB;
8'h43: bin = 4'hC; 8'h44: bin = 4'hD; 8'h45: bin = 4'hE;
8'h46: bin = 4'hF; default: bin = 4'h0;
endcase

// Continuous decoding of ascii hex chars into binary values
always @(ascii)
case (ascii)
8'h30: ishexchar = 1'b1; 8'h31: ishexchar = 1'b1;
8'h32: ishexchar = 1'b1;8'h33: ishexchar = 1'b1;8'h34: ishexchar = 1'b1;
8'h35: ishexchar = 1'b1; 8'h36: ishexchar = 1'b1;
8'h37: ishexchar = 1'b1; 8'h38: ishexchar = 1'b1;
8'h39: ishexchar = 1'b1; 8'h41: ishexchar = 1'b1;
8'h42: ishexchar = 1'b1; 8'h43: ishexchar = 1'b1;
8'h44: ishexchar = 1'b1; 8'h45: ishexchar = 1'b1;8'h46: ishexchar = 1'b1;
default: ishexchar = 1'b0;
endcase
endmodule
```

serial_async_transmitter.v

```
// RS-232 TX module
// (c) fpga4fun.com KNJN LLC - 2003, 2004, 2005, 2006
//`define DEBUG   // in DEBUG mode, we output one bit per clock cycle
//(useful for faster simulations)
```

```
module serial_async_transmitter(clk, TxD_start, TxD_data, TxD, TxD_busy);
input clk, TxD_start;
input [7:0] TxD_data;
output TxD, TxD_busy;
parameter ClkFrequency = 50000000; // 50MHz
parameter Baud = 115200;
parameter RegisterInputData = 1; // in RegisterInputData mode, the input doesn't
// have to stay valid while the character is been transmitted
// Baud generator
parameter BaudGeneratorAccWidth = 16;
reg [BaudGeneratorAccWidth:0] BaudGeneratorAcc;
initial BaudGeneratorAcc = 0;
`ifdef DEBUG
wire [BaudGeneratorAccWidth:0] BaudGeneratorInc = 17'h10000;
`else
wire [BaudGeneratorAccWidth:0] BaudGeneratorInc =
((Baud<<(BaudGeneratorAccWidth-4))
+(ClkFrequency>>5))/(ClkFrequency>>4);
`endif
wire BaudTick = BaudGeneratorAcc[BaudGeneratorAccWidth];
wire TxD_busy;
always @(posedge clk) if(TxD_busy)
BaudGeneratorAcc <= BaudGeneratorAcc[BaudGeneratorAccWidth-1:0]
+ BaudGeneratorInc;
// Transmitter state machine
reg [3:0] state;
initial state = 0;
wire TxD_ready = (state==0);
assign TxD_busy = ~TxD_ready;
reg [7:0] TxD_dataReg;
always @(posedge clk)
if(TxD_ready & TxD_start) TxD_dataReg <= TxD_data;
wire [7:0] TxD_dataD = RegisterInputData ? TxD_dataReg : TxD_data;
always @(posedge clk)
case(state)
4'b0000: if(TxD_start) state <= 4'b0001;
4'b0001: if(BaudTick) state <= 4'b0100;
4'b0100: if(BaudTick) state <= 4'b1000;  // start
4'b1000: if(BaudTick) state <= 4'b1001;  // bit 0
4'b1001: if(BaudTick) state <= 4'b1010;  // bit 1
4'b1010: if(BaudTick) state <= 4'b1011;  // bit 2
4'b1011: if(BaudTick) state <= 4'b1100;  // bit 3
4'b1100: if(BaudTick) state <= 4'b1101;  // bit 4
4'b1101: if(BaudTick) state <= 4'b1110;  // bit 5
4'b1110: if(BaudTick) state <= 4'b1111;  // bit 6
4'b1111: if(BaudTick) state <= 4'b0010;  // bit 7
4'b0010: if(BaudTick) state <= 4'b0011;  // stop1
4'b0011: if(BaudTick) state <= 4'b0000;  // stop2
default: if(BaudTick) state <= 4'b0000;
endcase
```

```
// Output mux
reg muxbit;
always @( * )
case(state[2:0])
3'd0: muxbit <= TxD_dataD[0];
3'd1: muxbit <= TxD_dataD[1];
3'd2: muxbit <= TxD_dataD[2];
3'd3: muxbit <= TxD_dataD[3];
3'd4: muxbit <= TxD_dataD[4];
3'd5: muxbit <= TxD_dataD[5];
3'd6: muxbit <= TxD_dataD[6];
3'd7: muxbit <= TxD_dataD[7];
endcase
// Put together the start, data and stop bits
reg TxD;// register the
always @(posedge clk) TxD <= (state<4) | (state[3] & muxbit);
//output to make it glitch free
endmodule
```

sync_fifo.v

```
module sync_fifo #(parameter DATA_WIDTH = 8, parameter DEPTH = 16,
parameter ADDR_WIDTH = log2(DEPTH))
   (input [DATA_WIDTH-1:0]  din,input wr_en, input rd_en, output
   [DATA_WIDTH-1:0] dout,
      output reg full, output reg empty, input clk, input reset );
   function integer log2;
      input integer n;
      begin
         log2 = 0;
         while(2**log2 < n) begin
            log2=log2+1;
         end
      end
   endfunction
   reg   [ADDR_WIDTH : 0]   rd_ptr; // note MSB is not really address
   reg   [ADDR_WIDTH : 0]   wr_ptr; // note MSB is not really address
   wire  [ADDR_WIDTH-1 : 0] wr_loc;
   wire  [ADDR_WIDTH-1 : 0] rd_loc;
   reg   [DATA_WIDTH-1 : 0]   mem[DEPTH-1 : 0];
   assign wr_loc = wr_ptr[ADDR_WIDTH-1 : 0];
   assign rd_loc = rd_ptr[ADDR_WIDTH-1 : 0];
   always @(posedge clk) begin
      if(reset) begin
         wr_ptr <= 'h0;
         rd_ptr <= 'h0;
```

```
      end // end if
      else begin
         if(wr_en & (~full))begin
            wr_ptr <= wr_ptr+1;
         end
         if(rd_en & (~empty))
            rd_ptr <= rd_ptr+1;
      end //end else
   end//end always
   //empty if all the bits of rd_ptr and wr_ptr are the same.
   //full if all bits except the MSB are equal and MSB differes
   always @(rd_ptr or wr_ptr)begin
      //default catch-alls
      empty <= 1'b0;
      full <= 1'b0;
      if(rd_ptr[ADDR_WIDTH-1:0]==wr_ptr[ADDR_WIDTH-1:0])begin
         if(rd_ptr[ADDR_WIDTH]==wr_ptr[ADDR_WIDTH])
            empty <= 1'b1;
         else
            full <= 1'b1;
      end//end if
   end//end always

   always @(posedge clk) begin
      if (wr_en)
         mem[wr_loc] <= din;
   end //end always
   //comment if you want a registered dout
   assign dout = rd_en ? mem[rd_loc]:'h0;
endmodule
```

serial_async_receiver.v

```
// RS-232 RX module
// (c) fpga4fun.com KNJN LLC - 2003, 2004, 2005, 2006
module serial_async_receiver(clk, RxD, RxD_data_ready,
 RxD_data, RxD_endofpacket, RxD_idle);
input clk, RxD;
output RxD_data_ready;  // onc clock pulse when RxD_data is valid
output [7:0] RxD_data;
parameter ClkFrequency = 50000000; // 50MHz
parameter Baud = 115200;
// We also detect if a gap occurs in the received stream of characters
// That can be useful if multiple characters are sent in burst
//  so that multiple characters can be treated as a "packet"
output RxD_endofpacket;  // one clock pulse, when
```

```
//no more data is received (RxD_idle is going high)
output RxD_idle;  // no data is being received
// Baud generator (we use 8 times oversampling)
parameter Baud8 = Baud*8;
parameter Baud8GeneratorAccWidth = 16;
wire [Baud8GeneratorAccWidth:0] Baud8GeneratorInc=
((Baud8<<(Baud8GeneratorAccWidth-7))+
(ClkFrequency>>8))/(ClkFrequency>>7);
reg [Baud8GeneratorAccWidth:0] Baud8GeneratorAcc;
initial Baud8GeneratorAcc = 0;
always @(posedge clk) Baud8GeneratorAcc <=
 Baud8GeneratorAcc[Baud8GeneratorAccWidth-1:0]+Baud8GeneratorInc;
wire Baud8Tick = Baud8GeneratorAcc[Baud8GeneratorAccWidth];
//////////////////////////
reg [1:0] RxD_sync_inv;
initial RxD_sync_inv = 0;
always @(posedge clk) if(Baud8Tick) RxD_sync_inv <= {RxD_sync_inv[0], ~RxD};
// we invert RxD, so that the idle becomes "0",
// to prevent a phantom character to be received at startup
reg [1:0] RxD_cnt_inv;
reg RxD_bit_inv;
initial RxD_bit_inv = 0;
initial RxD_cnt_inv = 0;
always @(posedge clk)
if(Baud8Tick)
begin
if( RxD_sync_inv[1] && RxD_cnt_inv!=2'b11)
RxD_cnt_inv <= RxD_cnt_inv + 2'h1;
else
if(~RxD_sync_inv[1] && RxD_cnt_inv!=2'b00)
RxD_cnt_inv <= RxD_cnt_inv - 2'h1;
if(RxD_cnt_inv==2'b00)
RxD_bit_inv <= 1'b0;
else
if(RxD_cnt_inv==2'b11)
RxD_bit_inv <= 1'b1;
end
reg [3:0] state;
reg [3:0] bit_spacing;
initial state = 0;
initial bit_spacing = 0;
// "next_bit" controls when the data sampling occurs
// depending on how noisy the RxD is, different values might work better
// with a clean connection, values from 8 to 11 work
wire next_bit = (bit_spacing==4'd10);
always @(posedge clk)
if(state==0)
bit_spacing <= 4'b0000;
else
if(Baud8Tick)
```

```
bit_spacing <= {bit_spacing[2:0] + 4'b0001} | {bit_spacing[3], 3'b000};
always @(posedge clk)
if(Baud8Tick)
case(state)
4'b0000: if(RxD_bit_inv) state <= 4'b1000;  // start bit found?
4'b1000: if(next_bit) state <= 4'b1001;  // bit 0
4'b1001: if(next_bit) state <= 4'b1010;  // bit 1
4'b1010: if(next_bit) state <= 4'b1011;  // bit 2
4'b1011: if(next_bit) state <= 4'b1100;  // bit 3
4'b1100: if(next_bit) state <= 4'b1101;  // bit 4
4'b1101: if(next_bit) state <= 4'b1110;  // bit 5
4'b1110: if(next_bit) state <= 4'b1111;  // bit 6
4'b1111: if(next_bit) state <= 4'b0001;  // bit 7
4'b0001: if(next_bit) state <= 4'b0000;  // stop bit
default: state <= 4'b0000;
endcase
reg [7:0] RxD_data;
initial RxD_data = 8'hFF;
always @(posedge clk)
if(Baud8Tick && next_bit && state[3]) RxD_data <= {~RxD_bit_inv, RxD_data[7:1]};
reg RxD_data_ready, RxD_data_error;
always @(posedge clk)
begin
// ready only if the stop bit is received
RxD_data_ready <= (Baud8Tick && next_bit && state==4'b0001 && ~RxD_bit_inv);
 // error if the stop bit is not received
RxD_data_error <= (Baud8Tick && next_bit && state==4'b0001 &&  RxD_bit_inv);
end
reg [4:0] gap_count;
initial gap_count = 0;
always @(posedge clk) if (state!=0) gap_count<=5'h00; else if(Baud8Tick &
 ~gap_count[4]) gap_count <= gap_count + 5'h01;
assign RxD_idle = gap_count[4];
reg RxD_endofpacket; always @(posedge clk) RxD_endofpacket <= Baud8Tick &
 (gap_count==5'h0F);
endmodule
```

Appendix E

Thermal measurements

E.1 Thermal characterization

The thermal measurements were performed using K and T type thermocouples. The thermocouples may be attached to each components in different ways. For MOSFETs, the thermocouple may be attached to the drain tab using thermally conductive epoxy, preferably electrically insulating. Due to the relatively low melting point of the metals used in them, T type thermocouples may be soldered onto the drain tab. Alternatively, the thermal sensor may simply be secured between the drain tab and the heat sink. In each case the measured thermal impedance will be slightly different. For magnetic components, the thermocouple may be slid into the gap between the core and the winding. This was found to be the most convenient way to measure temperature changes in those components, while being repeatable. Alternatively the sensor can be glued or taped to the top of the magnetic core of inserted into the gap.

In performing the characterization of the populated board, thermocouples were attached to each MOSFET drain tab and inserted into the two magnetic components. Starting with the board being in ambient steady state, DC current was conducted through one of the devices at a time (either drain to source of a MOSFET, or the leads of the magnetic components) and the temperature rise was measured in all other components. The sensors were connected to a 12 channel thermocouple readout box[1], which recorded the 12 readings every 4 seconds and stored the results to a computer. The ambient temperature was also measured to ensure that no significant changes in ambient temperature occurred. The voltage drop across and the current through the conducting component were recorded. After the temperatures reached steady state, the current level was increased (typically a power supply in CC mode was used to run this test). This was done for three power levels for each component. Fig. E.1 shows a typical plot of temperature for four components that are thermally coupled (these are the four cycloconverter devices). The temperature rise is greatest in the conducting component, but the other components' temperatures follow the same trend, albeit with scaling factors. Due to the large amount of copper used on the board and the general physical scale of the components, it can take up to an hour for the temperature to reach approximate steady state. The other components' temperature curves are essentially unperturbed by power dissipation is the shown component. The measurement is consistent with the expectation that the complement FET in a half bridge would be the most thermally coupled component, with the other cycloconverter devices being about evenly, weakly coupled.

[1] Digi-Sense Thermocouple Scanning Thermometer with 12-Channels - Benchtop

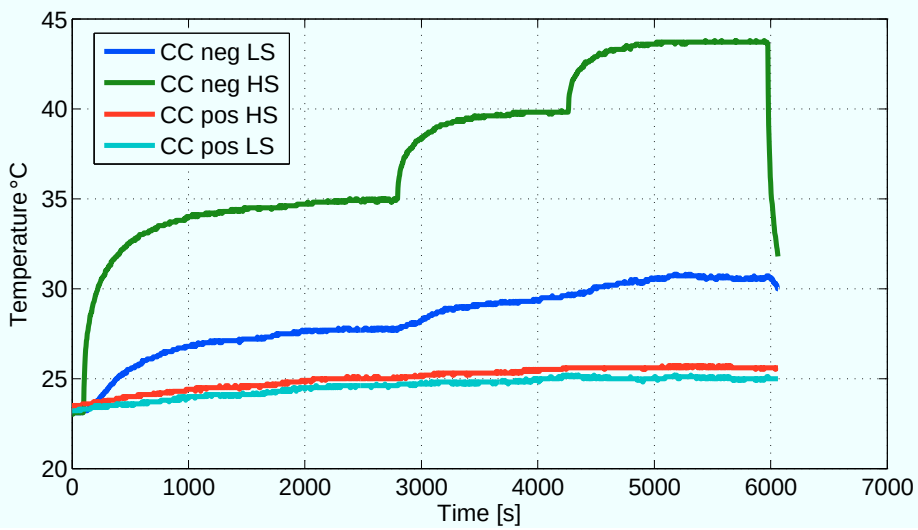


Figure E.1: Temperature changes in 3 of the cycloconverter FETs, as the power dissipation in the high side of negative cycloconverter leg steps between 0.5, 0.76 and 0.96W.

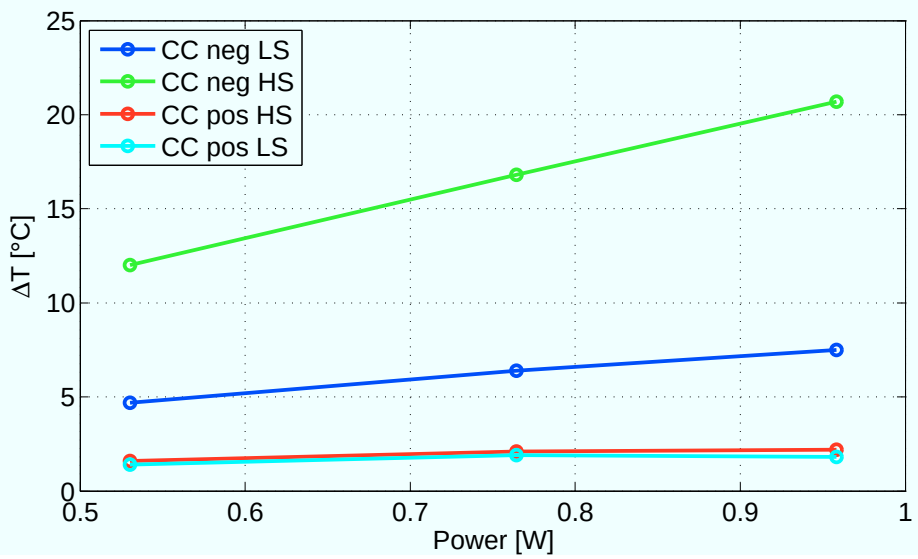


Figure E.2: The rise in temperature over steady state for each of the components due to power dissipation in the LS of negative leg of CC.

To derive the thermal impedance relationship from power dissipation in this component to temperature rise in every other component, the three steady state changes in temperature (from ambient) are plotted against power dissipation for all of these components. The result is shown on Fig. E.2. The resulting relationship is well fitted by a linear function (which agrees with theory) and the resulting slope of each line is the thermal impedance coefficient from one component to the other.

E.2 Thermal impedance matrices

This analysis was performed for all of the components with major losses. Defining the total loss and temperature changes in every component as quantities:

$$P_{loss} = \begin{pmatrix} P_{\text{FB,lag,ls}} \\ P_{\text{FB,lag,hs}} \\ P_{\text{FB,lead,ls}} \\ P_{\text{FB,lead,hs}} \\ P_{\text{Transformer}} \\ P_{\text{Inductor}} \\ P_{\text{CC,neg,hs}} \\ P_{\text{CC,neg,ls}} \\ P_{\text{CC,pos,hs}} \\ P_{\text{CC,pos,ls}} \end{pmatrix} W$$

and

$$\Delta T_{all} = \begin{pmatrix} \Delta T_{\text{FB,lag,ls}} \\ \Delta T_{\text{FB,lag,hs}} \\ \Delta T_{\text{FB,lead,ls}} \\ \Delta T_{\text{FB,lead,hs}} \\ \Delta T_{\text{Transformer,bobbin}} \\ \Delta T_{\text{Inductor,bobbin}} \\ \Delta T_{\text{CC,neg,hs}} \\ \Delta T_{\text{CC,neg,ls}} \\ \Delta T_{\text{CC,pos,hs}} \\ \Delta T_{\text{CC,pos,ls}} \end{pmatrix} C/W.$$

The resulting thermal impedance matrices are then given by:

$$R_{th1} = \begin{pmatrix} 22.6 & 11.8 & 4.0 & 5.7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10.1 & 20.9 & 4.4 & 6.6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4.2 & 4.4 & 22.7 & 11.6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5.6 & 6.3 & 9.2 & 19.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.9 & 1.6 & 1.7 & 1.1 & 9.9 & 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 22.4 & 5.6 & 1.6 & 3.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5.1 & 17.9 & 2.0 & 4.0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3.0 & 2.0 & 17.7 & 8.8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3.0 & 1.5 & 7.1 & 21.7 \end{pmatrix} C/W$$

and

$$R_{th2} = \begin{pmatrix} 18.0 & 9.2 & 4.2 & 5.8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 8.9 & 15.6 & 4.7 & 7.2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2.6 & 2.3 & 17.9 & 11.4 & 0.9 & 0 & 0 & 0 & 0 & 0 \\ 3.3 & 3.4 & 8.3 & 16.6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.1 & 0.6 & 1.7 & 2.5 & 9.9 & 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 21.9 & 6.3 & 0.7 & 1.6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 8.2 & 21.0 & 0.7 & 1.6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.2 & 1.3 & 14.9 & 6.3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.2 & 1.7 & 6.2 & 17.6 \end{pmatrix} C/W,$$

$$\Delta T_{all} = R_{th1,2} P_{loss} \tag{E.1}$$

such that the final expression for thermal characterization is given by equation (E.1). Thermal impedance matrix R_{th1} was obtained with the thermocouples soldered to the drains of all MOSFET devices. Matrix R_{th2} was measured from a setup that used thermally conductive epoxy called Arctic Silver[2] to attach the thermocouples to the full-bridge devices and the cycloconverter thermocouples were held in place by the tension provide by the screw to heat sink connection. As result of poorer thermal conductivity, the thermal sensitivity is lower in the second matrix for both sets of MOSFETs. Both matrices are nearly symmetric (which indicates low unidirectional heat transfer during the test). They are both invertible, which allows for the intended power loss calculation to take place.

As a final note, it was observed that measured the temperature in an operational inverter resulted in a significant amount of EMI pickup by the less than 0.7m long thermocouple wire. This resulted in a large enough common mode voltage at the output terminals that the read-out box was damaged at one point. It is thus recommended that the thermocouples be disconnected from the read out box for the duration of operation of the inverter and quickly connected back once the operation ends (although this introduces some error).

[2]http://www.arcticsilver.com/

Appendix F

Prototype hardware

F.1 Parts list

Table F.1 shows the part which were used in the prototype, except for the two magnetic components, which were custom wound as shown in chapter 5.

F.2 PCB Layout

The prototype is implemented on a four layer printed circuit board. They are shown on the next few pages in Fig. F.1, F.2, F.3, F.4.

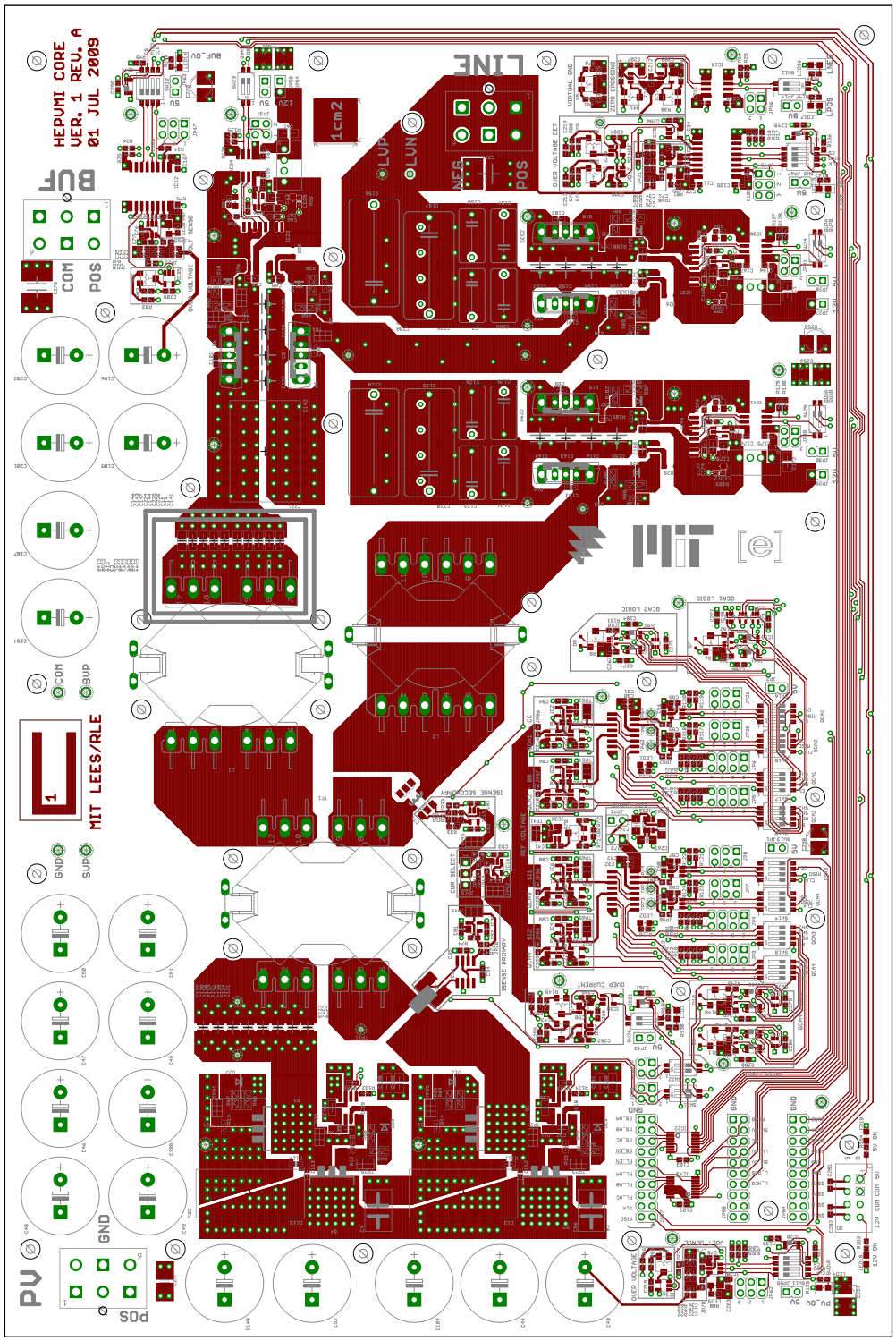
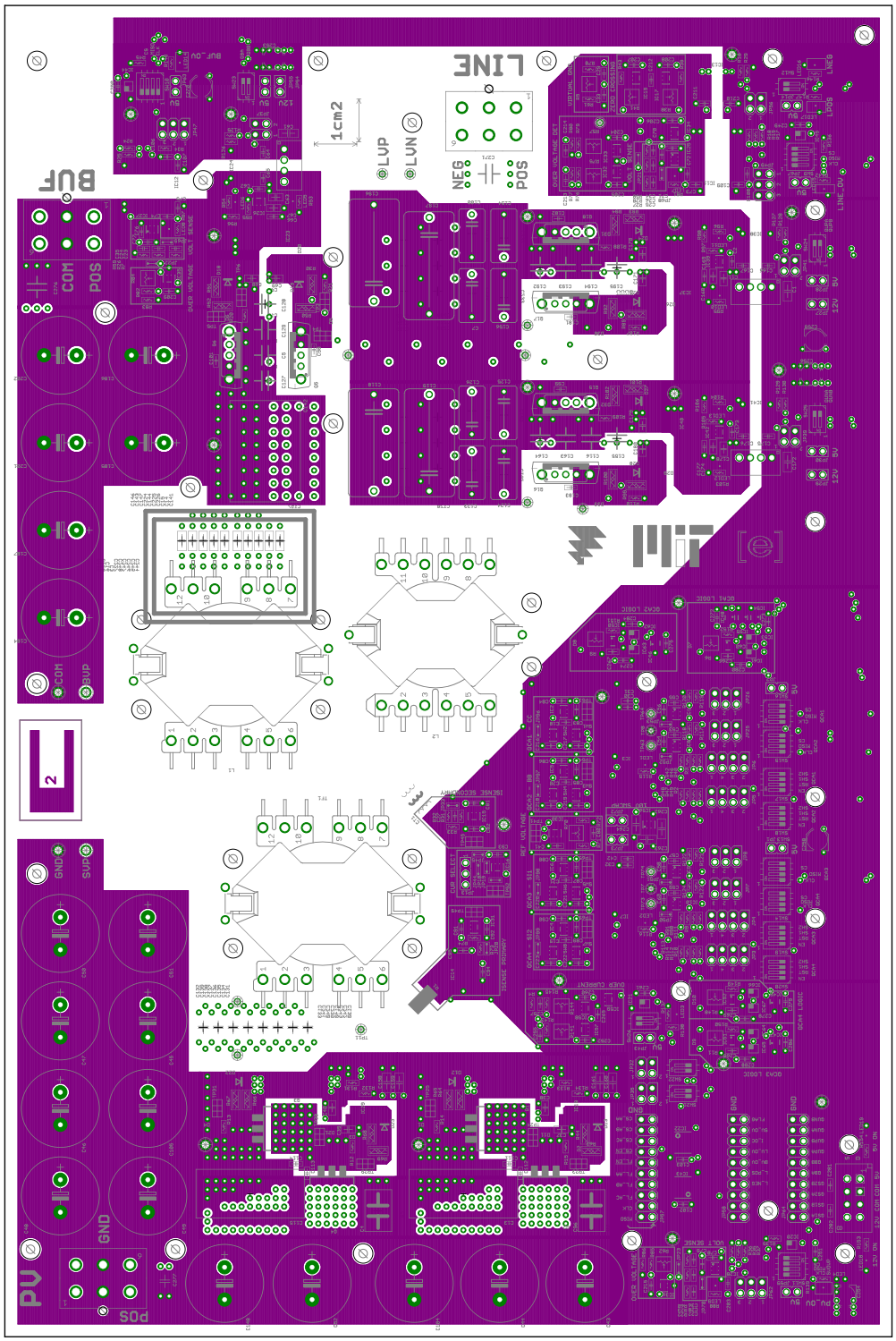


Figure F.1: PCB layer 1 of 4



5/21/2010 1:37:31 PM f=0.89 D:\Documents\eagle\HEPVMI-core-v1rA\main.brd

Figure F.2: PCB layer 2 of 4

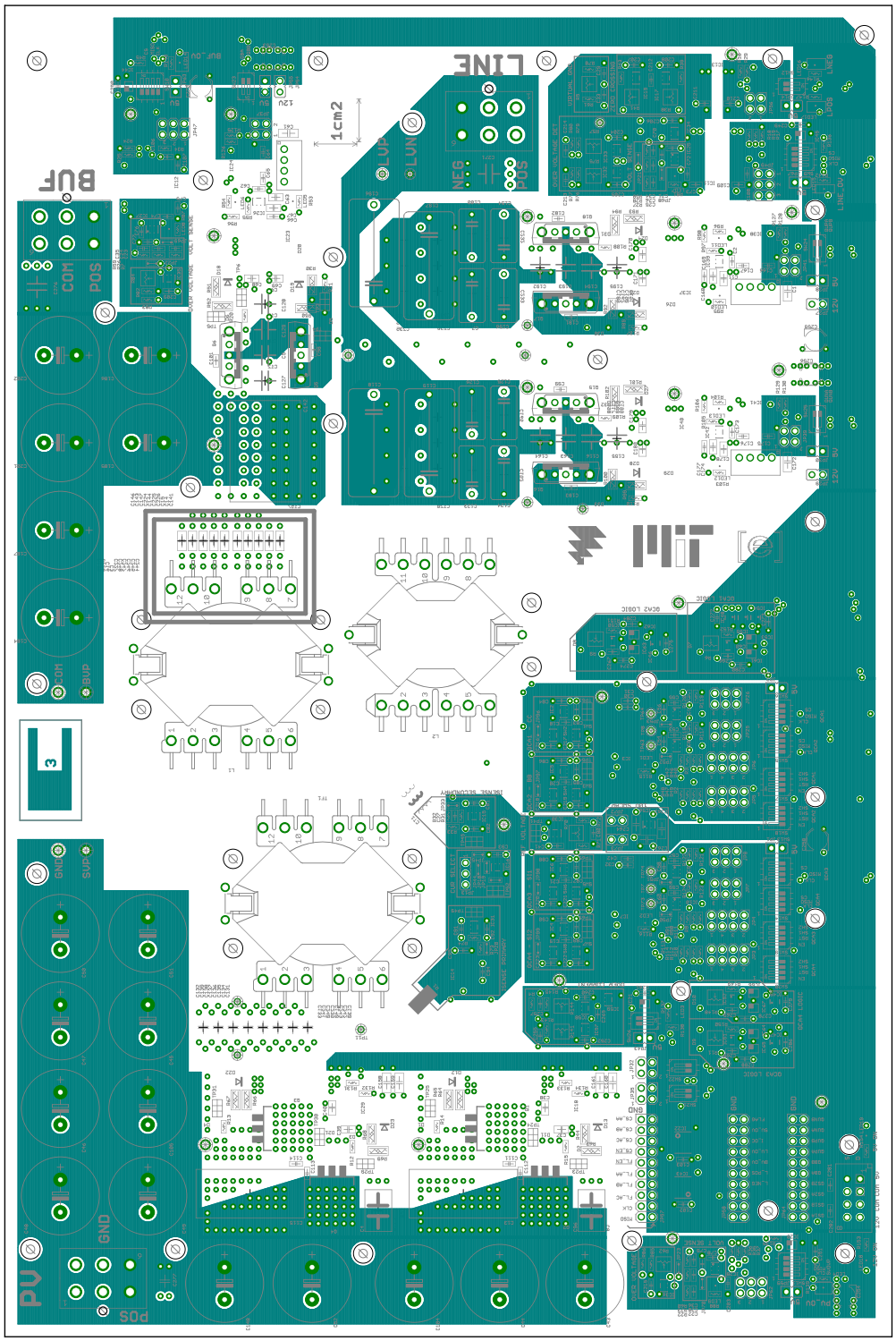


Figure F.3: PCB layer 3 of 4

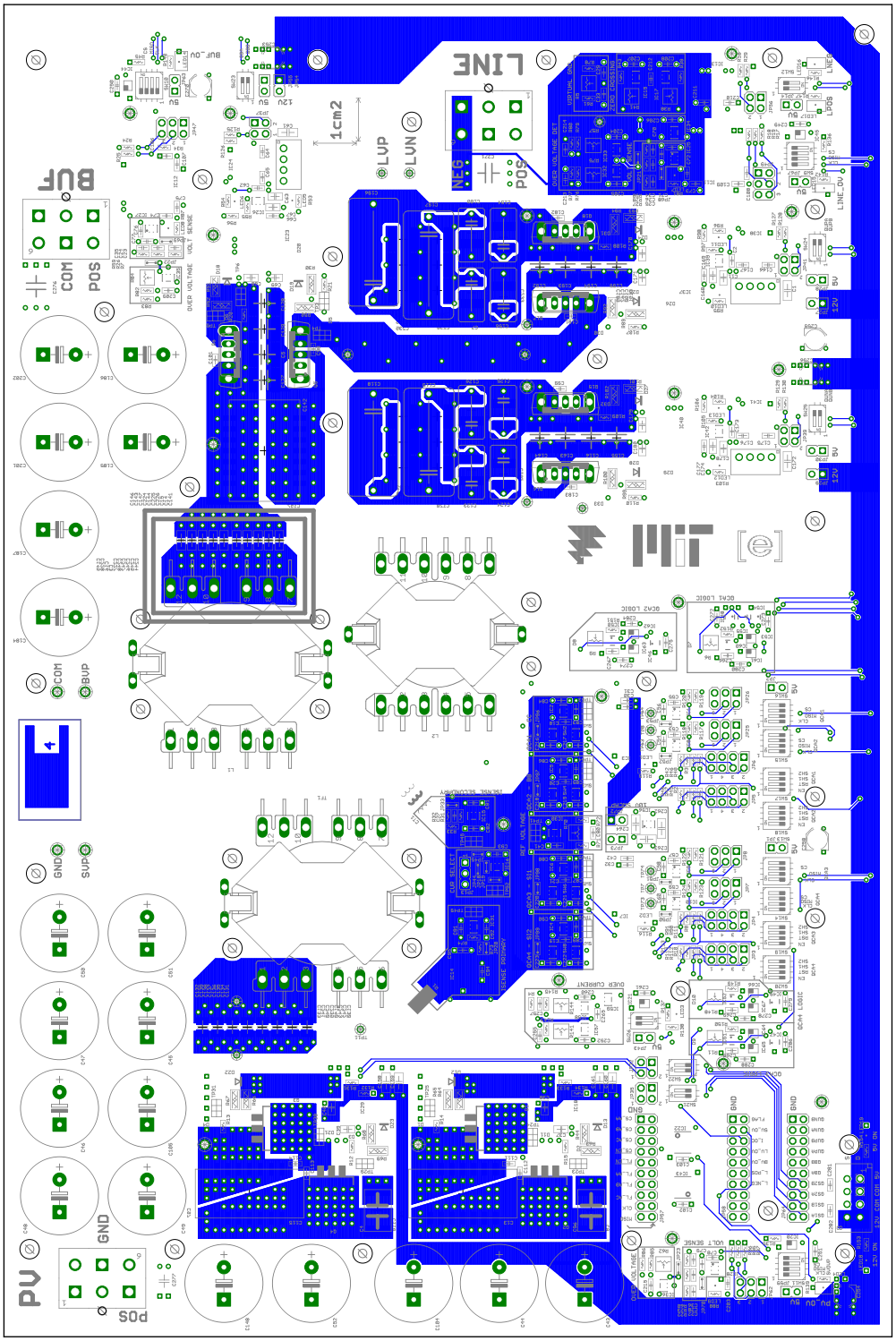


Figure F.4: PCB layer 4 of 4

Manufacturer and part number	Description	Quantity	Cost each [S]
Infineon IPP60R250CP	High voltage MOSFETs	4	3.18
International Rectifier IR2113STRPBF	High voltage gate driver	2	4.19
STMicroelectronics STTH1R06A	High voltage bootstrap diode	2	0.50
Linear Technology	Voltage regulator LT1761ES5-5	2	2.38
Analog Devices ADUM3210BRZ	Digital isolator	2	3.04
CUI Inc. VBSD1-S12-S12-SIP	Isolating DC/DC converter	2	4.66
STMicroelectronics STB160N75F3	Low voltage MOSFETs	4	5.55
National Semiconductor LM5101	Low voltage gate driver	2	3.72
Nichicon HE series 100V, 680μF	Bulk buffer capacitance	6	1.74
Capstick	Filter capacitance 405K100CS4	4	-
Vishay	Filter capacitance VJ2220Y224KBEAT4X	20	3.70
Vishay	Resonant capacitance, secondary VJ1210A222JXGAT5Z	20	1.37
Kemet	Resonant capacitance, primary R82EC3100DQ70J	30	0.21
TDK	Blocking capacitance, secondary C3225X7R2A225K	30	1.32
Murata Electronics	Various decoupling capacitance GRM31CR71E475KA88L	20	0.45
Taiyo Yuden	Various decoupling capacitance TMK212B7105KG	30	0.165
Murata Electronics	Various decoupling capacitance GRM188R71E105KA12D	20	0.29
Rohm	Resistor 100k MCR03EZPJ104	20	0.07
Vishay	Gate resistors CRCW12068R20FNEA	10	0.114
Diodes Inc BAV19WS	Clamping and low voltage boot strap diode	20	0.5

Table F.1: Prototype parts list.

Bibliography

[1] Xiaoming Yuan; Yingqi Zhang; , "Status and Opportunities of Photovoltaic Inverters in Grid-Tied and Micro-Grid Systems," Power Electronics and Motion Control Conference, 2006. IPEMC 2006. CES/IEEE 5th International , vol.1, no., pp.1-4, 14-16 Aug. 2006

[2] Kjaer, S.B.; Pedersen, J.K.; Blaabjerg, F.; , "A review of single-phase grid-connected inverters for photovoltaic modules," Industry Applications, IEEE Transactions on , vol.41, no.5, pp. 1292- 1306, Sept.-Oct. 2005

[3] Amirabadi, M.; Balakrishnan, A.; Toliyat, H.A.; Alexander, W.; , "Soft switched ac-link direct-connect photovoltaic inverter," Sustainable Energy Technologies, 2008. ICSET 2008. IEEE International Conference on , vol., no., pp.116-120, 24-27 Nov. 2008

[4] Yaosuo Xue; Liuchen Chang; Sren Baekhj Kjaer; Bordonau, J.; Shimizu, T.; , "Topologies of single-phase inverters for small distributed power generators: an overview," Power Electronics, IEEE Transactions on , vol.19, no.5, pp. 1305- 1314, Sept. 2004

[5] Bellar, M.D.; Wu, T.S.; Tchamdjou, A.; Mahdavi, J.; Ehsani, M.; , "A review of soft-switched DC-AC converters," Industry Applications, IEEE Transactions on , vol.34, no.4, pp.847-860, Jul/Aug 1998

[6] Henze, C.P.; Martin, H.C.; Parsley, D.W.; , "Zero-voltage switching in high frequency power converters using pulse width modulation," Applied Power Electronics Conference and Exposition, 1988. APEC '88. Conference Proceedings 1988., Third Annual IEEE , vol., no., pp.33-40, 1-5 Feb 1988

[7] Bhat, A.K.S.; Dewan, S.D.; , "Resonant inverters for photovoltaic array to utility interface," Aerospace and Electronic Systems, IEEE Transactions on , vol.24, no.4, pp.377-386, Jul 1988

[8] Lohner, A.; Meyer, T.; Nagel, A.; , "A new panel-integratable inverter concept for grid-connected photovoltaic systems," Industrial Electronics, 1996. ISIE '96., Proceedings of the IEEE International Symposium on , vol.2, no., pp.827-831 vol.2, 17-20 June 1996

[9] Quaschning, V.; Hanitsch, R.; , "Influence of shading on electrical parameters of solar cells," Photovoltaic Specialists Conference, 1996., Conference Record of the Twenty Fifth IEEE , vol., no., pp.1287-1290, 13-17 May 1996

[10] Quan Li; Wolfs, P.; , "A Review of the Single Phase Photovoltaic Module Integrated Converter Topologies With Three Different DC Link Configurations," Power Electronics, IEEE Transactions on , vol.23, no.3, pp.1320-1333, May 2008

[11] Ho, B.M.T.; Henry Shu-Hung Chung; , "An integrated inverter with maximum power tracking for grid-connected PV systems," Power Electronics, IEEE Transactions on , vol.20, no.4, pp. 953- 962, July 2005

[12] Kawabata, T.; Honjo, K.; Sashida, N.; Sanada, K.; Koyama, M.; , "High frequency link DC/AC converter with PWM cycloconverter," Industry Applications Society Annual Meeting, 1990., Conference Record of the 1990 IEEE , vol., no., pp.1119-1124 vol.2, 7-12 Oct 1990

[13] Jih-Sheng Lai; , "Power conditioning circuit topologies," Industrial Electronics Magazine, IEEE , vol.3, no.2, pp.24-34, June 2009

[14] A.K. Hayman, "Development of a High-Efficiency Solar Micro-Inverter," S.M. thesis, Department of Electriacal Engineering, Massachusetts Institute of Technology, MA, 2009.

[15] Guichao Hua; Lee, F.C.; , "Soft-switching techniques in PWM converters," Industrial Electronics, Control, and Instrumentation, 1993. Proceedings of the IECON '93., International Conference on , vol., no., pp.637-643 vol.2, 15-19 Nov 1993

[16] Bower, W.; Whitaker, C.; Erdman, W.; Behnke, M.; Fitzgerald, M.; , "Performance Test Protocol for Evaluating Inverters Used in Grid-Connected Photovoltaic Systems," Prepared for the California Energy Commission, Oct 2004.

[17] Gu, W.-J.; Liu, R.; , "A study of volume and weight vs. frequency for high-frequency transformers," Power Electronics Specialists Conference, 1993. PESC '93 Record., 24th Annual IEEE , vol., no., pp.1123-1129, 20-24 Jun 1993

[18] Kassakian, J.; Schlecht, M.; Verghese, G.; , "Principles of power electronics," Addison Welseley Publishing Company, 1991. pp. 594-596

[19] Ferroxcube, "Design of planar power transformers," Appl. Note, pp. 4

[20] Godoy, P.A.; Perreault, D.J.; Dawson, J.L.; , "Outphasing Energy Recovery Amplifier With Resistance Compression for Improved Efficiency," Microwave Theory and Techniques, IEEE Transactions on , vol.57, no.12, pp.2895-2906, Dec. 2009

[21] B.J. Pierquet, Designs for Ultra-High Efficiency Grid-Connected Power Conversion, Unpublished Ph. D. Dissertation., Dept. Elect. Eng., Mass. Inst. Tech., MA, 2010