A Photovoltaic Test Platform Realized with Multiple Independent Outputs

Thesis

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Kevin Patrick Crawford, M.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2011

Thesis Committee:

Jin Wang, Advisor

Longya Xu

Donald Kasten
Abstract

In this thesis, a project to design and build a photovoltaic test platform is discussed. Essentially, it is a photovoltaic simulator designed to have multiple outputs. A photovoltaic simulator, then, is a device that produces the electrical output as a photovoltaic panel would, but without the photovoltaic panel. This device, based on a full bridge architecture, utilizes a phase shifting control algorithm to apply current control through the transformer and to the outputs. Additionally, using the current-voltage photovoltaic characteristic curve and a second control loop, the output current and voltage levels can be controlled to operate at points along the desired curve. The power flow analysis shows that the current in the transformer can be calculated quite specifically, and by increasing the phase angle of an output, the current to that output increases accordingly. Analysis on the transformer and control algorithm supports this conclusion. Preliminary experimental results were collected that demonstrate successful tracking of the photovoltaic characteristic over a range of loads. The controllable region attainable with the current device was partially limited, but solutions are proposed for future work.
Acknowledgments

I would like to thank my advisor, Prof. Jin Wang, for all of his guidance and assistance. I would also like to thank fellow student Mark Scott for his multiple contributions and general assistance.
Vita

June 2004.................................................St. Thomas Aquinas High School

June 2009.................................................B.S. Electrical and Computer Engineering,
The Ohio State University

Fields of Study

Major Field: Electrical and Computer Engineering
# Table of Contents

Abstract ................................................................. ii

Acknowledgments ...................................................... iii

Vita ................................................................. iv

List of Figures ....................................................... vii

Chapter 1: Introduction ............................................. 1

Chapter 2: Background Information ............................. 3

Chapter 3: System Design .......................................... 10
  Circuit Design ...................................................... 10
  Control Design ................................................... 28

Chapter 4: System Analysis ....................................... 37
  Power Flow Analysis ............................................... 37
  Transformer Analysis ............................................... 54
  Control Analysis .................................................. 58
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Current-Voltage Characteristic Curve of a Photovoltaic Panel [6]</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>Current-Voltage Curves by Irradiance Level</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>Full Bridge DC/DC Converter Topology</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>MOSFET Snubber Circuit</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>Gate Drive Circuit</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>One Half of Transformer Core [1]</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>Shoot-Through Protection Circuit</td>
<td>22</td>
</tr>
<tr>
<td>10</td>
<td>Under-Voltage Lock-out Circuit</td>
<td>23</td>
</tr>
<tr>
<td>11</td>
<td>Voltage Sensor Circuit</td>
<td>24</td>
</tr>
<tr>
<td>12</td>
<td>Current Sensor Circuit</td>
<td>25</td>
</tr>
<tr>
<td>13</td>
<td>Interface Board Voltage Followers</td>
<td>26</td>
</tr>
<tr>
<td>14</td>
<td>Control Block Diagram for One Output</td>
<td>28</td>
</tr>
<tr>
<td>15</td>
<td>PWM Counter</td>
<td>32</td>
</tr>
<tr>
<td>16</td>
<td>Interrupt Timing</td>
<td>33</td>
</tr>
<tr>
<td>17</td>
<td>Standard Transformer Model</td>
<td>37</td>
</tr>
<tr>
<td>Page</td>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>18</td>
<td>Simplified Transformer Model</td>
<td>38</td>
</tr>
<tr>
<td>19</td>
<td>Primary and Secondary Transformer Voltage Waveforms</td>
<td>39</td>
</tr>
<tr>
<td>20</td>
<td>Transformer Current Waveform</td>
<td>41</td>
</tr>
<tr>
<td>21</td>
<td>Simplified Three-Winding Transformer Model</td>
<td>42</td>
</tr>
<tr>
<td>22</td>
<td>Delta Model of Transformer</td>
<td>43</td>
</tr>
<tr>
<td>23</td>
<td>Three Winding Transformer Voltage Waveforms</td>
<td>44</td>
</tr>
<tr>
<td>24</td>
<td>Current Waveforms for Matching Loads</td>
<td>48</td>
</tr>
<tr>
<td>25</td>
<td>Current Waveforms for Unbalanced Loads</td>
<td>49</td>
</tr>
<tr>
<td>26</td>
<td>Updated Current Waveforms for Unbalanced Loads</td>
<td>51</td>
</tr>
<tr>
<td>27</td>
<td>Simulation Circuit for Three Winding Transformer Power Flow</td>
<td>52</td>
</tr>
<tr>
<td>28</td>
<td>Simulated Current Waveforms for Matching Loads</td>
<td>53</td>
</tr>
<tr>
<td>29</td>
<td>Simulated Current Waveforms for Unbalanced Loads</td>
<td>53</td>
</tr>
<tr>
<td>30</td>
<td>MOSFET voltage waveforms with a Zero Degree Phase Angle</td>
<td>64</td>
</tr>
<tr>
<td>31</td>
<td>MOSFET voltage waveforms with a Thirty Degree Phase Angle</td>
<td>65</td>
</tr>
<tr>
<td>32</td>
<td>I-V Operating Points</td>
<td>66</td>
</tr>
<tr>
<td>33</td>
<td>I-V Operating Points with Manipulation of Input Voltage</td>
<td>68</td>
</tr>
<tr>
<td>34</td>
<td>Output Current and Voltage with Load Change</td>
<td>69</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

With the current energy needs, along with extensive considerations for environmental protection, it is no surprise that renewable and clean energy sources are increasing in demand. The global energy demands are only increasing, and without clean energy sources, the protection of the environment will continue to fall behind. As such, a great deal of research has been devoted to the field of renewable energy. This thesis discusses a project designed to further that effort.

The simplest description of this project is the design of a photovoltaic simulator. Essentially, this is a power supply that operates in such a way that the power produced by it has the same characteristics as the power produced by a photovoltaic panel. Without using any photovoltaic materials, the device produces an output that matches the electrical characteristics of a typical photovoltaic panel as exactly as possible through specific control methods. In this way, a photovoltaic panel is effectively simulated in a laboratory environment. Such devices have been created before, though, and so this project expands on those designs. In the initial design of the simulator, two separate outputs are available for use, and each one independently simulates a separate photovoltaic panel.

There are several motivating factors for this project. First, creating a photovoltaic
simulator significantly enhances the ability to test and develop many projects revolving around solar energy research. As this research has been increasing in popularity, it is likely that such a device would see considerable use. Additionally, having multiple outputs allows the simulation of multiple panels simultaneously. Obviously, the intended purpose for having multiple outputs is so that they can be used simultaneously, but it is worth mentioning that this is particularly worthwhile due to the typical architecture of photovoltaic systems. Due to the fact that individual photovoltaic panels do not produce a large amount of power, many systems tend to have multiple panels connected together to yield more practical amounts of power. Also, the multiple outputs allows all of this while at the same time maximizing power density and also limiting the space taken up in a laboratory. Finally, the multiple outputs from a single unit also allows the two panels to be controlled through a single interface, simplifying its use.

In this thesis, the project is discussed in its entirety. The circuit design, from selection of circuit architecture to transformer design, along with detailed aspects of the system, is explained. The control algorithm for the system is explained, along with its implementation. Circuit analysis, most significantly including a power flow analysis to evaluate the basic principles through which the device functions, follows the design section. The control method of the system is also analyzed. Finally, experimental results, recorded from operating the device under various conditions, is shown. At the end of the thesis, a brief section details future work to further advance the project.
Chapter 2: Background Information

The purpose of a photovoltaic simulator, simply put, is to generate an electrical output that matches what a photovoltaic panel would generate. However, the electrical characteristic that must be matched is fairly specific as well as somewhat complex. The concept of how the voltage and current relate to each other is straightforward, but the details of its behavior make simulating it more complicated. Thus, a deeper understanding of the behavior is necessary.

The electrical characteristic of photovoltaic panels is quite consistent. The shape of the voltage and current relationship, or I-V curve, has a very uniform shape. In addition, for many photovoltaic panels, the values of voltage and current at which the panels operate are all reasonably close, only varying within a small range. This simplifies the goal, as photovoltaic panels all match each other in electrical output for the most part. A typical I-V curve can be seen in Figure 1.

This I-V curve in Figure 1 has a few obvious characteristics. It is immediately noteworthy that the behavior of the photovoltaic panel changes dramatically from one end of the curve to the other. This behavior has been described through classifying the curve by three regions. The first region would be the area of the curve where the current remains mostly constant over a wide range of voltage. This region starts at
0 V. The second region follows the constant current trend, and can be characterized by a roughly constant change of voltage corresponding to a roughly constant change of current. In other words, the second region maintains a pattern of something closer to a linear change of resistance. Finally, the third region changes trends again. A relatively constant voltage over a wide current range characterizes the third region. As seen in Figure 2, the regions of approximately constant current, linear resistance, and constant voltage characteristics comprise the entire I-V curve of a typical photovoltaic panel [3].

Figure 1: Current-Voltage Characteristic Curve of a Photovoltaic Panel [6]
The shape of the I-V curve, and thus the behavior in each of the three regions, are dictated by the structure of photovoltaic panels. This is true for essentially all photovoltaic panels, regardless of voltage and current ratings. Thus, while the voltage and current values depicted in Figure 2 are different from those in Figure 1, the shapes are the same, as the basic structure of the panels is the same. This structure enables the process of converting solar energy to electrical energy. The structure of a typical photovoltaic panel can be seen in Figure 3. As can be seen, the photovoltaic panel primarily consists of a semiconductor material in two layers. This is a key factor in how a photovoltaic panel functions. The semiconductors used, most frequently...
silicon, display a property called the photovoltaic effect.

The photovoltaic effect refers to the process where, upon contact from a photon, electrons in the valence band of the semiconductor are excited and elevated to the conduction band. This, in turn, causes a voltage to develop, or a current to flow, across the electrodes of the photovoltaic panel as more and more electrons are excited. Additionally, the electron may only be excited when the photon that is absorbed contains more energy than the semiconductor material’s bandgap energy. This relatively simple process is the basis for which the photovoltaic panel was developed.

Multiple factors affect the ability of the photovoltaic panel to absorb these photons. First, the intensity of the light source, which can be quantified by irradiance, indicates the amount of energy in each photon. Obviously, for a greater irradiance, photons will have more energy. This causes more photons to have the necessary energy to bridge the bandgap, thus exciting more electrons. A second factor is the wavelength of the photons. Semiconductors in photovoltaic panels have a different response to photons depending on the wavelength of the light. If the frequency is
too low, the photovoltaic effect may not appear at all. However, even above such a threshold, there is a certain range of wavelengths for which the semiconductor material will respond most efficiently [2]. The effect this has on the photovoltaic panel is illustrated in Figure 4, where different levels of irradiance generate different quantities of power.

Figure 4: Current-Voltage Curves by Irradiance Level

As previously mentioned, the I-V characteristics of a photovoltaic panel depend on this photovoltaic effect. The three regions as previously described, then, can be explained based on this effect. First, the constant current region can be described
as a current limit. The current limit is equivalent to the total current produced by photovoltaic excitation in the panel. Along with a large enough load, the current limit can be reached. This operating point occurs at a voltage of 0 V. As the load is decreased, the operating point along the curve will change. The current decreases very slightly over this region, but the voltage increases very rapidly. Thus, the whole region operates at or near the current limit, created by the maximum electron excitation.

The opposite region, the approximately constant voltage region, behaves similarly. In this case, the voltage limit is equivalent to total excitation of the photovoltaic panel in terms of voltage potential. In this case, the operating point is with no load, and thus no current. Under these conditions, the photovoltaic excitation is clearly applied to developing a net voltage across the photovoltaic panel. However, it is notable that this limitation is different than that of the current limit. Also, as the load is increased, the voltage decreases slowly relative to the current change, but faster than the current change in the first region.

The last region, where a change in current is typically proportional to its corresponding change in voltage, operates in a different manner. This region operates near the power limit that a photovoltaic panel can produce. In essence, it is a balance of the current and voltage limitations. At maximum power output, a photovoltaic panel will again, of course, operate at maximum electron excitation. However, as this region operates below both current and voltage limits, the excitation is used for both effects. The load is small enough that, in addition to producing the necessary current, a considerable voltage is also produced. It is in this region that the photovoltaic maximum power can be achieved. Unlike the other two regions, where one
effect dominates, this balanced output can more effectively yield electrical power.

Finally, common uses for photovoltaic panels all serve the same basic purpose, though the degree to which they are used varies significantly. Photovoltaic panels are always used for their ability to generate electrical power. Sometimes they are used individually, for small, isolated devices. Other times, photovoltaic panels are used in extensive grids as part of a solar power plant to generate massive amounts of power. In these cases, or anywhere in between, the manner in which the panels are connected is significant. Connecting the panels in series provides a higher maximum voltage. Also, connecting panels, or strings of panels, in parallel allows for a greater maximum current. However, the combination of both series and parallel connections must be used to effectively generate large, and practical, power outputs.

In order to do this, though, other circuitry is often required to maintain efficiency. Variations from one panel to the next may produce a slightly different maximum voltage or maximum current. These imbalances get magnified in large grids of panels. With one string capable of providing a larger voltage than another, connecting them in parallel without any interface considerably reduces the potential power output from the combined total. One string of panels would constantly be flowing current into the other string. Thus, to avoid this, circuitry is often used to regulate the voltage or current levels. These interface elements are often the subject of research today, and photovoltaic simulators are thus worthwhile for this purpose, among others.
Chapter 3: System Design

Circuit Design

Designing the circuits in this device was a fairly extensive process. Once an overall concept of what the device would be was decided upon, the design process then broke down into many separate, smaller circuits. However, the overall circuit design began with specifying the operating conditions of the device. In order to maintain a relatively small device size, the switching frequency was selected to be 200 kHz. Likewise, to match photovoltaic behavior, the range of output current and voltage was set as slightly larger than what was required to enable the device to operate across the whole required range. Typical photovoltaic panel operating ranges go up to about 45 V, 8.5 A, and 250 W, as determined by a survey of current available units. From this, the maximum output voltage was selected as 60 V, the maximum output current was chosen to be 12.5 A, and the maximum output power was chosen to be 300 W. Additionally, the minimum input voltage that would maintain the power rating required by the device was determined to be 22 V. While the device might never be operated at those maximum levels, it ensured that the components used in the device would be capable of operating over the entire range needed. Given the
basic concept, then, the circuits were designed from a large scale approach initially. As the circuits became more distinct, detailed design procedures were applied to each circuit element.

Of course, there were many separate circuit elements of the system to be designed. The first design consideration, and probably the most significant, was the overall switching architecture. The circuit architecture selected for this device was a full bridge DC/DC converter with synchronous rectification. This constitutes circuits called H-bridges at both the input and output sides of a transformer. An H-bridge is a connection of four switches, typically MOSFETs, as two sets of two switches connected in series. The middle point of each set of switches would then be connected to either end of a transformer winding. By controlling the switches, the voltages applied to the transformer windings can be manipulated, and thus the power flowing through the transformer can be controlled.

There were several reasons for this choice. First, other photovoltaic simulators with just one output had utilized this approach in the past. The architecture worked very effectively in those cases. Second, synchronous rectification has multiple benefits, including the removal of diode voltage drops as a source of power loss. Also, synchronous rectification enables the use of controlling the power flow through phase shifting of the secondary switches relative to the primary. This phase shifting control is discussed in greater detail in later sections.

Additionally, this topology allows for a modular circuit design and future addition of more outputs. Each input or output H-bridge, as well as other components, was designed such that the board layout for each could be the same. Thus, the input
board has the exact same layout as an output board. The only differences are a few components that are or are not populated depending on which board is being used. Other than this, the input circuit is the same as an output circuit.

The transformer in the full bridge architecture serves two purposes. First, it isolates the output side from the input electrically. This prevents undesired interaction between the two sides, particularly in terms of electrical noise. Second, it is used to step up the voltage. A turns ratio of 1:3 was selected to allow a smaller voltage input to achieve the higher voltage range required by the photovoltaic electrical characteristic. This turns ratio, in an ideal circuit, produces a voltage at the output that is
exactly three times larger than the input voltage, and so an input of 15 V would reach
the 45 V output that is required to simulate a photovoltaic panel.

The switches on both input and output were selected to be MOSFETs. These
devices have demonstrated the necessary performance required by this device, and
they are simple to use. The switches are controlled such that two MOSFETs in series
will never be turned on at the same time, preventing short circuits. Additionally,
when switches one and four from Figure 5 are turned on, a positive voltage will be
applied to the primary transformer winding. Likewise, switches two and three on at
the same time apply a negative voltage to the transformer primary winding. Also,
during operation of the device, the output capacitors maintain reasonably constant
voltages over the duration of a given switching cycle. Due to this, similar switch
controls on the secondary side allow the application of positive or negative voltages
to the transformer secondary winding.

Not pictured in Figure 5 are the snubber circuits connected to the switches. Due
to the behavior of MOSFETs, snubber circuits are frequently required to remove
noise from the switching waveforms and transmit the desired voltage to the trans-
former. The snubber circuits pictured in Figure 6 were selected due to their ability
to specifically reduce oscillation on switching turn-off transitions. After this point,
selecting the component values started based on the oscillation frequency and then
were modified through simple trial and error. Once the switching waveforms were
improved to acceptable levels, the parameters of the snubber circuits were considered
accurately tuned.

The final aspect of the circuit design dealing with the switches in the full bridge are
the gate drive circuits required. The switches in the full bridge carry a large amount of current and switch from essentially 0 V to a large voltage value. As such, there is the potential for significant switching power loss. When the switches transition from on to off or from off to on, the voltage across the switch and current through the switch rise or fall fairly smoothly during the switching time duration. This overlap of a non-zero voltage with a non-zero current causes what is referred to as switching power loss. In order to minimize this, the length of time over which the switches make this transition must be minimized. This is a primary reason for gate drive circuits.

The gate drive circuits are themselves essentially very small power supplies that
Figure 7: Gate Drive Circuit

can provide a high current to the gates of the switches. A high current makes a MOSFET transition from one state to the other very rapidly, thus minimizing power loss. In order to maintain a reasonable efficiency, then, gate drive circuits are required. Also, though, the MOSFETs would transition far too slowly without the gate drive circuits such that the full bridge architecture would almost certainly fail to operate at all.

The gate drive circuits, as shown in Figure 7, are isolated from the control signals through an opto-isolator. This is done in order to electrically separate the digital signal from the power connections. The gate drive circuits share a ground with the MOSFETs, and this ground is used for the high power signal. This ground, then, is kept separate from the digital control ground. Were the two to be connected directly, it would cause significant problems for the digital circuitry. The isolation at the gate drive circuits prevents this at this location in the circuit.

Another significant design element of the device was the design of the transformer itself. Phase shifting control requires the transformer to be appropriately designed in
order to function well. In particular, phase shifting control requires that the transformer have a reasonably large mutual inductance, reasonably small leakage inductances, and leakage resistances that are much smaller than the leakage inductances. Additionally, it helps considerably for the leakage inductances of the outputs to be as similar as possible. Finally, in order to minimize the effect the outputs have on each other, it helps for the input side leakage inductance to be smaller than the output leakage inductances. The reasons for all of this will be explained in detail in the power flow analysis. Thus, the transformer was designed with these factors in mind, along with other considerations such as the maximum power requirement and physical size.

The process of designing the transformer was very likely the most mathematically driven design problem. The design process began by first defining the circuit’s operating parameters. The maximum input and output voltage, current, and power ratings, as well as operating frequency, provided a baseline for the transformer selection. With this in mind, the first step was calculating the turns ratio.
\[
N = \frac{V_{in-min} \cdot 0.9}{V_{out-max}} = \frac{22V \cdot 0.9}{60V} \approx \frac{1}{3}
\] (1)

This equation gives a simple estimate for the turns ratio, based on input and output voltage. The input voltage is the minimum voltage the input may drop to during operation. The output voltage is the maximum it may require. The 0.9 factor in the equation is an estimate of the effect of voltage regulation of the transformer, allowing room for some voltage drop. From the numbers that were known, this gave a turns ratio of roughly 1:3. This ratio was selected, in part, due to the potentially low number of turns. A low number of turns would be helpful as this would keep the leakage inductances small. However, the ratio selection does not specify the actual number of turns. Nevertheless, once this ratio was found, the next step was to determine the area product, \(W_aA_c\) and from it, select a set of cores to choose from.

\[
W_aA_c = \frac{P_{in} \cdot C}{4 \cdot B \cdot f \cdot K} = \frac{700W \cdot 5.07 \cdot 10^{-3} cm^2/A}{4 \cdot 1000G \cdot 200kHz \cdot 0.30} = 1.48 cm^4
\] (2)

From (2), the area product was found. The variables used were \(P_{in}\), the input power, \(C\), the current capacity per area, \(B\), the flux density, \(f\), the frequency, and \(K\), the winding factor. Most of the numbers were known, but some had to be estimated as typical values, such as the flux density and current capacity. Regardless, the area product narrowed down the core options to a few varieties. The three leading candidates out of the options found had area products of 2, 4, and 10 \(cm^4\), which were the closest values to the calculated area product. They were from a class of transformer cores called E cores, due to the shape of half of a single core resembling
the letter ‘E’.

After this, the surface area of the transformer core was calculated, based on specifications of the potential core choices. The dimensions of the cores were thoroughly specified in their datasheets, allowing easy calculation of surface area over the entire core. From the values found, a calculation of the thermal resistance of the core was carried out.

\[
R_{\text{therm}} = \frac{800 \ (°C \cdot cm^2/W)}{A_{\text{surface}}} = \frac{800}{95.302} = 8.56 \frac{°C}{W} \quad (3)
\]

The thermal resistance, \( R_{\text{therm}} \), was calculated for the three primary core options, and one was found to be considerably lower than the other two. This was due to a significantly larger size, and thus, surface area. This initially suggested that the larger core might be the best choice. However, this by itself was not enough to make a selection of the core. The next step was to calculate, based on the core parameters and maximum flux density, what the number of turns on the secondary windings would be for each core.

\[
N_S = \frac{V_{\text{out}}}{2 \cdot B_{\text{max}} \cdot A_E \cdot f_T} = \frac{45 \ V}{2 \cdot 0.1 \ T \cdot 0.000353 \ m^2 \cdot 200 \ kHz} = 3.19 \quad (4)
\]

Using the rated output voltage for a given output, parameters of the given transformers, an estimated maximum flux density of 1000 G, and a frequency of 200 kHz for the rate at which the transformer would oscillate, the number of turns for the secondary winding corresponding to each of the three cores were found. The smaller two cores provided values of about eight and six, while the largest had a value of
about three. As the turns ratio had already been set at one to three, the number of secondary turns was selected to be three for the case of using the third core, and either three or six for either of the others. Three turns rather than six would be preferred, as this would help to reduce leakage inductance. After that, the maximum flux density was calculated with the number of secondary turns known.

\[
B_{\text{max}} = \frac{V_{\text{out}}}{2 \cdot N_S \cdot A_E \cdot f_T} = \frac{45 V}{2 \cdot 3 \cdot 0.000353 \, m^2 \cdot 200 \, kHz} = 1062.3 \, G
\]  

Equation (5) is simply a reorganization of (4). However, the calculations for the three cores once again showed that the largest core had the best performance, as its maximum flux density was the smallest of the three. In fact, it was almost exactly the estimated value at 1062.3 G. This was under the condition of three turns for all three cores. Calculating the smaller two cores with six windings results in roughly equivalent maximum flux densities.

Next, the core loss and temperature rise were calculated for the three cores. The core loss was given in the datasheets in terms of power per unit volume as an absolute maximum value, so the volume of the cores was used as well.

\[
P_C = P_L \cdot Vol = 134 \, mW/cm^3 \cdot 44 \, cm^3 = 5.896 \, W
\]  

The core loss value, \( P_C \), can then be utilized, along with the thermal resistance, to find the temperature rise of the transformers.

\[
T_R = P_C \cdot R_{\text{therm}} = 5.896 \, W \cdot 8.56^\circ C/W = 50.47^\circ C
\]  

19
Not only did the largest transformer core have the smallest core loss, but its thermal resistance was also the smallest. Thus, this transformer had the smallest temperature rise, which was calculated to be 50.47 °C. Additionally, the other two transformer cores had temperature rises calculated to be larger than 110 °C, and this temperature rise would be unacceptably large. Thus, the largest core, labeled as 45528, was selected.

There were still a couple more steps to complete in designing the transformer, though. A transformer material had to be chosen from a variety of materials that had varying core losses depending on temperature. Given the temperature rise calculated, and assuming an ambient temperature of 25 °C, the P-material was chosen. It had the smallest core loss at around 75 °C.

Next, a suitable air gap had to be designed, if necessary. The flux density for the primary side was calculated to determine the need for an air gap.

\[
B = \frac{\mu \cdot N \cdot I}{l_{\text{eff}}} = \frac{4\pi \times 10^{-7} \cdot 2500 \frac{H}{m} \cdot 1 \cdot 50 A}{0.124 m} = 1.2667 T
\]  

(8)

Equation (8) comes from Ampere’s Law, using \(\mu\) for permeability, \(N\) for number of turns, \(I\) for current, and \(l_{\text{eff}}\) for effective length through the transformer core. The permeability in this case was 2,500 times the permeability of air. The value found for maximum flux density was clearly far too large to not have an air gap. Thus, the length of the air gap was found for a desired flux density limit of about 1000 G.

\[
l_{ag} = \frac{\mu \cdot N \cdot I}{B} = \frac{4\pi \times 10^{-7} \cdot 1 \frac{H}{m} \cdot 1 \cdot 50 A}{0.1 T} = 0.63 mm
\]  

(9)
Finally, in order to select the material used to fabricate the windings, the skin depth was calculated.

\[ D_{pen} = \frac{7.6}{\sqrt{f}} = \frac{7.6}{\sqrt{200,000}} = 0.017 \text{ cm} = 0.0067 \text{ in} \quad (10) \]

In order to maximize the use of the winding material, it is desired to have the thickness of the material be as close to the same as the skin depth as possible. This ensures that current flows through all of the winding material and not just the surface of the material. This reduces losses and fully utilizes the transformer windings. Given the skin depth of 0.0067 in, copper foil with a thickness of 0.01 in was chosen as the winding material. The thickness is close to, but slightly larger than, the skin depth, making effective use of the windings. Thus, with the turns ratio, the number of primary and secondary turns, the type of core, the air gap, and the winding material selected, the transformer was then completely specified.

Another aspect of the circuit design was protection circuits. There were two primary protection circuits implemented, and they function together as well. First, there was a shoot-through protection circuit created. This circuit is basically a logic circuit tied into the gate drive system that allows only one output to be active in any pair of MOSFETs connected in series. This directly prevents every series pair of MOSFETs from turning on at the same time and creating a short circuit that would destroy the MOSFETs and cause other potential damage to the system. As can be seen from Figure 9, the outputs of the protection circuit that go to the gate drive feed back to the logic NOR gates. If an output is high, this causes the output of the other NOR gate to go to a logic low if it is not already. Additionally, a fault signal is
in place that, if high, would force both switches to an off state. This fault signal is separate from the shoot through protection, however.

The other protection circuit is based on the concept of under-voltage detection. Should the voltage on the gate drive drop below a percentage value of what it should be, this would indicate that something is drawing more power than it should. This would only occur in the case of a fault, or a short circuit. When this is detected, the under-voltage lock-out is activated. Once activated, the gate drive enable signal, labeled GD\_3\_EN, turns off, forcing the gate drive of that switch to be low.

Additionally, the under-voltage circuits activate a fault control signal that, when activated, disables all four switches in the associated H-bridge. This is the same fault signal seen in Figure 9, and thus the fault action of stopping power flow is performed through that circuit. The result is that current very quickly ceases to flow through the MOSFETS in the forward direction. However, along with this fault
signal, the under voltage lockout signals and general fault signals generated are sent to the microcontroller. Those signals can be used to initiate a fault algorithm in the microcontroller to shut down the gate drive signals.

These two protection circuits are not the only protection for the system, though. A separate protection algorithm is implemented in the code that monitors current overflow. More detail on this algorithm is included in the control design section. Regardless, the combination of the code protection along with the physical protection circuits provides a layered protection for the system. Should one protection circuit fail, another aspect of the system protection can and should still activate.

The next circuit design element are the sensor circuits. The circuits are fairly simple, as they are mostly just interfacing between the sensor devices and providing appropriate signals to the system. The voltage sensor is a linear optocoupler that is set up to generate a proportional voltage signal relative to the sensed voltage. The device displays some nonlinearity at low voltage, but as this voltage range has the
exact same current level for photovoltaic characteristics, the poor sensor behavior in that range does not matter. The voltage sensor is shown in Figure 11. In order to control the output voltage level relative to the input, the resistors in the circuit must be specified. The resistors on the input reduce the input voltage level by a factor of five through the voltage divider. After this, the third resistor, R173, and the output resistor, R175, generate the gain for the voltage sensor transfer function. The input current is required to keep the voltage level at the inverting input at 0 V, and the output current proportionally mirrors this. This dictates the transfer function of the sensor.

\[
\frac{V_{out}}{V_{in}} = \frac{R175}{R173} \cdot \frac{R172}{R171 + R172}
\]  

(11)

The signal generated is, thus, an analog signal, and it is converted to a 0-3 V range so that the voltage sensors can be read by the analog to digital converters in
The current sensor, on the other hand, has more design features. First of all, though two current sensors are pictured, only one is populated on each board. This allows for the current sensor to be oriented in either direction, and thus any board can be set up as an input or an output, as previously mentioned for the modular design approach. Next, the current sensor operation is very simple. It outputs a voltage between 0.5 V and 4.5 V corresponding directly to a current range of 0 A to 30 A. The behavior is consistent and very linear over this range. However, in order to improve upon the granularity of the sensed current for the microcontroller, the range is modified by the amplifier circuit shown in Figure 12. The amplifier adjusts the sensor voltage range of 0.5 V to 2.5 V, corresponding to 0 A to 15 A of sensed

Figure 12: Current Sensor Circuit
current, to a practical output of 0 V to 3 V for the microcontroller. Resistors R176 and R180 provide the gain that increases the 2 V range of interest to the 3 V range for the microcontroller. Resistors R178 and R179 shift the output of the sensor to start at 0 V instead of 0.5 V. The conversion function can be seen below, where $V_{out}$ goes directly to the microcontroller, and $V_{sense}$ comes from the current sensor.

$$V_{out} = V_{dd} \frac{R_{179}}{R_{178} + R_{179}} \left(1 + \frac{R_{180}}{R_{176}}\right) - V_{sense} \frac{R_{180}}{R_{176}}$$  \hspace{1cm} (12)$$

Finally, the last aspect of the circuit design was the implementation of the DSP microcontroller. The microcontroller selected for this project was the TMS320F2812 microcontroller from Texas Instruments. This controller was selected as it was readily available in the laboratory, and it contained all of the capability required, especially its inclusion of twelve specific PWM signals, as well as analog to digital converters for analog sensors. The controller was already provided in a TMS320F2812 eZdsp board, providing connectors to the relevant signals. However, in addition to this board, an interface board was created to connect between the system and the eZdsp board.

![Figure 13: Interface Board Voltage Followers](image)

The interface board contains several features. First, and perhaps most obviously,
it has multiple connectors to provide a bridge between the system and the controller. However, some additional circuitry was included beyond this. To add additional stability to the signals coming from the sensors on the simulator, buffers were added between the inputs to the analog to digital converters of the microcontroller and the outputs of the sensor circuits. The buffers are simply op-amp based voltage follower circuits. This extra circuit helps to maintain the voltage levels coming out of the sensors due to the high impedance of the input to the voltage follower. Additionally, for signals coming from the microcontroller, buffers were put in place to provide additional current capability for the microcontroller signals. This utility was provided by a line driver chip.
Control Design

The control concept, as stated before, was to implement phase shifting control to directly control the power flow. The method chosen in order to do this was to use the phase shifting to achieve current control on the outputs. By this simple method, power flow can be controlled. However, one additional step had to be included, as current control by itself would not accurately simulate the electrical characteristic of a photovoltaic panel. Thus, an additional control loop was included based on the output voltage. The output voltage was then used to control the current reference in the current controller. Given the changing nature of the electrical characteristic of photovoltaics, the easiest way to perform this extra voltage control loop was through the use of a look-up table. The voltage value of the output feeds back into the table, and from this data table, the current reference is selected. Then, with current control, implemented by the phase shifting of the waveforms in the transformer, the output current tracks the correct current value corresponding to the output voltage. Thus, the power flow is controlled, and it is controlled to a point on the photovoltaic characteristic waveform.

Figure 14: Control Block Diagram for One Output
From Figure 14, the implementation of the control algorithm was actually fairly straightforward. The inner loop is the main control loop. This control loop implements the current control. Current sensors provide readings of the output currents. These values are then compared to the internal current references. The current error, defined as the difference between the actual, measured current and the desired current reference, is then manipulated by a PI controller. The PI controller adjusts the error value by proportional and integral terms. The proportional term multiplies the error signal by a set, constant value. This term accounts for the majority of the final value. The integral term multiplies the cumulative error as measured over the course of operation. The integral term is used primarily to ensure that there is no steady state error, as any small offset is reduced to zero over time. The two terms, added together, give an output value that, while driven by the error signal, is no longer truly an error value. In this case, the output from the PI controller directly provides the phase, as a proportion, that the secondary voltage signal must be shifted in the phase shifted control algorithm. This is achieved by tuning the PI controller to have the necessary multipliers. These values are found almost always by an empirical approach, and this was the method used for this project.

The outer loop, the voltage control loop, is in practice much simpler. In this control loop, the readings from the output voltage sensors are taken and used much like the current sensor readings in the inner loop. However, rather than finding an error or operating on the voltage value, the reading is instead simply fed into the look-up table. For each voltage value, with a precision of every 0.5 V, a corresponding current value is stored in this data table. Thus, for whatever voltage reading is
found, the corresponding current value is taken from the table and used as the current reference for its output. The current reference is used as indicated previously in the current control loop. In this way, the electrical characteristic behavior of photovoltaic panels is stored, and followed, within the voltage control loop.

Some additional factors went into the implementation of both control loops. While the algorithm is followed in both, they need to interact appropriately in order to function correctly. One primary detail is that the inner control loop, the current control loop, must update faster than the outer control loop. If the outer control loop updated too quickly, the reference for the current control loop would change too quickly. The current control could not function, then, as it would not have the time necessary to track to a specific current. In the actual implementation of the control, the voltage control loop updates only once for every ten times the current control loop is used. However, both loops operate at a high frequency relative to the signals involved. In fact, as the signals are, in typical operation, constant values with minimal fluctuation, there is plenty of speed for the control. The only significant control value changes would occur on load changes.

Another factor was the fact that there are two outputs to control. However, this was again fairly simple to accomplish. The control loops implemented for one output were duplicated exactly for a second output. The sensors and control values remain completely separate. The look-up table is reused for the second output, but this has no impact on the state at which each output is operating. Thus, the two outputs, having completely separate controllers, separate variables, and separate sensors, are controlled completely independently. This is particularly important for operating the
device and using both photovoltaic outputs in the same circuit.

The actual implementation of the entire control algorithm was carried out on the
DSP microcontroller. The code was written in the C programming language, as this
can be used directly with the microcontroller selected. Several aspects of the chosen
microcontroller dictated the manner in which the control would be implemented.
First of all, there are twelve pulse width modulation (PWM) signals built into the
microcontroller. They are designed to operate in pairs, as is appropriate for two
switches in series with each other. The two signals are forced to be complementary,
and each pair has just one comparison value that is compared to a timer. The
features of the PWM signals are controlled through two event manager units in the
microcontroller. The switching frequency can be set, the polarity of the PWM signals
can be set, and indeed even a dead time can be specified for the switching pairs.
However, the most powerful aspect of the event managers is their ability to update
the PWM signals twice in one PWM clock cycle. The event managers use specific
timers and counters for PWM, and in the desired mode the counters count up and
down in each clock cycle. In this mode, then, the counter will reach both the maximum
and minimum values once every clock cycle. The maximum value is reached halfway
within each clock cycle as well.

When the counter reaches the maximum counter value, it is referred to as timer
overflow. Likewise, when the counter reaches the minimum value of zero, it is referred
to as timer underflow. At both of these points, the PWM comparison values are
automatically updated based on a value stored in a comparison value mask register.
In addition, both of these events can be used to trigger interrupts. Thus, both the
rising and falling edge of every cycle for every PWM waveform can be exactly specified in this manner. By controlling these transitions, as well as maintaining a 180 degree separation between the transitions, the phase of PWM waveforms can be effectively controlled. Thus, this adjusting of PWM comparison values is the method by which phase shifted control is implemented.

Figure 16 shows the timing of the interrupts used in the program. The two interrupts for underflow and overflow are used only to update the PWM comparison values, and thus take little time. The third interrupt is the analog to digital conversion (ADC) interrupt. Its timing is set by a second timer that runs synchronously with
the main PWM timer. The position during the PWM clock cycle where it activates is simply set by a variable value. More time is allotted for this interrupt than the others. Not only does the ADC process take longer than the variable updates in the other interrupts, but more happens during the ADC interrupt as well. The first thing that happens in the ADC interrupt is the reading of signals on the analog to digital converters. The time it takes to read all six sensor inputs is about 240 $\mu$s. Thus, all of the values are sensed at a stable position, away from switching transition noise. This helps to improve the accuracy of the signals sensed. However, in order to make sure both the rise time and fall time values stay updated together, the rest of the control algorithm must be included within the ADC interrupt.

Because of this, the current and voltage values read in through the ADC process are used immediately in both control loops. To dictate the speed with which the control loops run, simple counters are used such that the control loops operate only
once every time the counter reaches its maximum. The values are used on these cycles to find error from the reference values, and the PI controllers for both outputs calculate new phases. The phases update the PWM values for the next cycle. Thus, because the whole process is included in the ADC interrupt, it ensures that the rise time value will not update without the fall time value, and vice versa.

Another interesting aspect of the code is the manner in which the PI controller was implemented. The microcontroller itself does not operate on floating point math directly, and thus can only approximate it through fixed point operations. This results in a very slow process. Since the PWM is operated at 200 kHz, the delay caused by floating point math took too long, and the ADC interrupt lasted beyond the point where the PWM compare values needed to be updated. Thus, the PI controller was modified to work with only integer math. This resulted in a large boolean structure, handling all the cases that would normally be handled by floating point math. Negative numbers in particular required special handling. However, with this extended code section, the overall speed of the PI controller was dramatically improved, allowing for its practical use.

Another aspect of the code implementation is simply a quality improvement on the signals measured in the ADC. In order to reduce noise on the incoming signals, filters were coded into the ADC interrupt. The code for the filters was very limited, operating very quickly. As the filters were only first order low pass filters, with a corner frequency between 400 and 500 Hz, the code implementation was very simple. Effectively, the previous value at the output of the filter is weighted heavily in the next value, thus dramatically reducing the rate of change of the value filtered. This
dramatically reduces the impact of single, errant readings due to noise. Therefore, the accuracy of the sensor signals was improved.

The next element of the code that was implemented was a soft start. This turned out to be quite simple to include. When the controller is reset at startup, the limit on phase angle is set at a much smaller level, effectively reducing the power limit. Once the voltage measured at a given output reaches a certain point, the phase limitation is moved back to its normal operating limit. This process essentially allows the output capacitors to charge more slowly, limiting in-rush current. This protects the components in the circuit from any potential damage that may cause. Once the capacitors are charged, then, the circuit can operate normally. The voltage value chosen for this threshold was 40 V, relative to the output maximum of 45 V.

Finally, there was a protection algorithm implemented purely through code in the microcontroller as well. The protection algorithm is an overcurrent sensor. On every cycle, the input current is measured. If the current goes over a preset current limit, a limit above normal operation range indicating a problem in the circuit, it is recorded. Should the input current remain above this limit for a set number of PWM cycles, this will trip the overcurrent protection, which updates the PWM registers, forcing all gate drive signals to a setting of low. So, in the event of a continuous current above a safe threshold, the circuit switches will deactivate, shutting off all power flow. The requirement for the overcurrent to be continuous is there to ensure that the overcurrent protection is not activated for a brief or abnormal reading. Thus, this protection method, in addition to the protection circuits implemented by hardware, provide the layers of system protection. Collectively, overcurrent detection in the
system, under-voltage detection on the switches, and shoot-through prevention for the switches are included in the overall protection scheme. In this way, the durability and reliability of the photovoltaic simulator are ensured.

The code explained in this section is attached in its entirety in Appendix A. A considerable number of comments appear in the code to indicate what the code is doing at each point.
Chapter 4: System Analysis

Power Flow Analysis

The power flow through the transformer can be better understood by first looking at the power flow through a similar system with only one output. In this scenario, the transformer simplifies and the equations reduce to a much quicker analysis. The power flow analysis begins with the model of the transformer. A standard transformer model can be seen in Figure 17.

![Standard Transformer Model](image)

Figure 17: Standard Transformer Model

Using this model, however, is prohibitively complex. Fortunately, several elements can be ignored without significant deviation from accuracy of results. Along with appropriate transformer design, the leakage resistances and mutual inductance can all be removed. The leakage resistances are small enough to not provide much impedance
effectively, while the mutual inductance is large enough, relative to the rest of the model, to appear as not much different from an open circuit. This means that all that is left is the leakage inductance of the primary and of the secondary. With both inductances referred to one side of the windings, this simplifies even further, to just a single inductance. This means that the transformer is modeled as nothing more than an inductor and a ground connection.

![Simplified Transformer Model](image)

Figure 18: Simplified Transformer Model

With the simplified model, the next step is to look at the voltage waveforms that are applied to the transformer windings. The input signal is a constant square-wave voltage coming out of the full bridge of switches. The assumption of this analysis is that the capacitors on the output side maintain a constant output voltage over the course of a switching cycle. This means that there is effectively a voltage applied to the secondary side by the full bridge switches there. However, there is the phase shift of the secondary switches, which means that this square-wave is shifted relative to the input voltage. In addition, in the ideal case, the voltage at the output would be identical to the input voltage, once referred to the same side. In reality, with power losses, the output voltage magnitude would be somewhat smaller. These waveforms
Given voltages across the transformer, along with a model of the transformer that is no more than a single inductor, the current through the transformer can be calculated from the basic equation governing inductor current. The equation is modified slightly to be in a simpler format with regards to a single switching cycle. A time duration is related in terms of phase angle and the switching frequency. Also, the current calculated by this equation is relative to the initial condition of the current. The current initial condition is given by $I_0$.

$$I = \frac{(\theta - \theta_0) \cdot V_L}{\omega \cdot L} + I_0$$

Based on this equation, the current through the transformer is easily calculated. However, it must be calculated for multiple sections of the switching cycle, as the $V_L$ value changes whenever there is a switching transition. The exact equations for this
case are below, where $V_P$ is the value of the DC primary voltage, and $V_S$ is the value of the DC secondary voltage. The specific phases, $\theta_N$, refer to those noted in Figure 19. Note that $\theta_0$ is zero in this case, and $\theta_2$ is $\pi$ as it is at the halfway point of the switching cycle. The $I_0$ term in each equation is relative to that segment only, and refers to the first point of the waveform over that section of time [5].

\begin{align*}
I &= (\theta - \theta_0) \cdot \frac{(V_P + V_S)}{\omega \cdot L} + I_0 \tag{14}
\end{align*}

\begin{align*}
I &= (\theta - \theta_1) \cdot \frac{(V_P - V_S)}{\omega \cdot L} + I_0 \tag{15}
\end{align*}

\begin{align*}
I &= (\theta - \theta_2) \cdot \frac{(-V_P - V_S)}{\omega \cdot L} + I_0 \tag{16}
\end{align*}

\begin{align*}
I &= (\theta - \theta_3) \cdot \frac{(V_S - V_P)}{\omega \cdot L} + I_0 \tag{17}
\end{align*}

The current waveform essentially indicates how power flows through the circuit. The AC current that flows through the transformer is generated due to the phase shift. Also, with a larger phase shift, a larger current waveform will be created. The current, then, directly translates to the power flow through the transformer. The current can then be converted to a root mean square (RMS) value. Along with the constant DC voltage at the output, which is identical to its RMS voltage, the average power is easy to calculate. The current over one switching cycle is enough to evaluate the RMS value.
Figure 20: Transformer Current Waveform

\[ I_{rms} = \sqrt{\frac{1}{2\pi} \int_0^{2\pi} [I]^2 d\theta} \]  

(18)

\[ P_{avg} = I_{rms} \cdot V_{rms} \]  

(19)

It should be noted that Equation (19) is not immediately clear. Normally, this would be apparent power, rather than active power. However, the RMS current flowing to the output is, in fact, essentially used to maintain the energy in the output.
capacitor. In the steady state, the RMS current flowing into the capacitor would exactly match the RMS current flowing out of the capacitor for the voltage to be maintained. Thus, the RMS current flowing in must be equivalent to the RMS current flowing out. Finally, since the output current and voltage are both DC values, the power from the output to the load is active power.

The extension to a second output adds more than a little complexity. First, the simplified transformer model changes to have three leakage inductances instead of two. In addition, those leakage inductances cannot be summed, and thus cause additional calculation.

\[ L_{12} = L_1L_2 + L_2L_3 + L_1L_3 \]

(20)

Figure 21: Simplified Three-Winding Transformer Model

The model seen in Figure 21, however, does not lend itself to easy analysis. In order to make the calculations both easier to perform and easier to understand, the transformer model can be converted using the Wye-Delta transform to the model in Figure 22.
In the new model, then, the analysis reverts to a straightforward approach, albeit with additional equations. The current can be broken down into that flowing from the input to the first output, $I_{12}$, the current from the input to the second output, $I_{13}$, and the current from the first output to the second output, $I_{23}$. These currents are defined as positive by the directions just stated in all following analysis. Also,
the voltages, now with three windings, are defined as $V_1$ for the input voltage, $V_2$ for the first output voltage, and $V_3$ for the second output voltage. As the voltages for all three windings are known, the current calculations are simple. However, there are more segments of each switching cycle over which the calculations must be made. Since there is a third winding, a third phase shifted voltage waveform appears on the transformer, creating additional sections [4].

![Three Winding Transformer Voltage Waveforms](image)

Figure 23: Three Winding Transformer Voltage Waveforms

The equations for current $I_{12}$ appear below:

$$I_{12} = \frac{(\theta - \theta_0) \cdot (V_1 + V_2)}{\omega \cdot L} + I_{12-0} \quad (23)$$

$$I_{12} = \frac{(\theta - \theta_1) \cdot (V_1 - V_2)}{\omega \cdot L} + I_{12-0} \quad (24)$$
\[ I_{12} = \frac{(\theta - \theta_2) \cdot (V_1 - V_2)}{\omega \cdot L} + I_{12-0} \]  
(25)

\[ I_{12} = \frac{(\theta - \theta_3) \cdot (-V_1 - V_2)}{\omega \cdot L} + I_{12-0} \]  
(26)

\[ I_{12} = \frac{(\theta - \theta_4) \cdot (-V_1 + V_2)}{\omega \cdot L} + I_{12-0} \]  
(27)

\[ I_{12} = \frac{(\theta - \theta_5) \cdot (-V_1 + V_2)}{\omega \cdot L} + I_{12-0} \]  
(28)

In this case, \( I_{12} \) is affected only by the voltage waveforms of \( V_1 \) and \( V_2 \). Similarly, the other currents are only affected by their two corresponding voltage waveforms. Thus, currents \( I_{13} \) and \( I_{23} \) are calculated in much the same way, though of course the voltages are different. Again, note that the initial condition term, \( I_{12-0} \) in the above equations, is relative to each segment of the switching cycle. Also, \( \theta_0 \) once again has a value of zero, but now it is \( \theta_3 \) that has a value of \( \pi \).

Next, the current going to each output or coming from the input can be calculated as well. Based on the model in Figure 21, it can be easily seen that the currents for the input and two outputs can be calculated as follows, based on the direction convention as previously stated:

\[ I_1 = I_{12} + I_{13} \]  
(29)
These simple calculations provide the currents going to each output. From this, then, RMS current can be found with Equation (18), and the power to each output is also easy to calculate:

\[
P_{2avg} = I_{2rms} \cdot V_2
\]

\[
P_{3avg} = I_{3rms} \cdot V_3
\]

The power to the first output is \( P_2 \) to match the current and voltage notation, and the power to output 2 is \( P_3 \).

In support of performing this power flow analysis, a script was written in MATLAB to generate the current waveforms for all three transformer windings over one switching cycle, with all three currents being referred to the primary side of the transformer. Parameters include the input and output voltages, the phase angles of the shifts of the two outputs, and the leakage inductance values. With this information, the equations previously provided were implemented over each segment of the waveform automatically. The behavior of the circuit can then be examined very quickly.
While any case can be chosen for analysis in this way, some cases are more pertinent than others. For example, the case of the two outputs being exactly balanced, or the case of one output having a larger load than the other. A few cases generated through the MATLAB program are shown below.

In Figure 24, it is shown that the two output currents are identical for having the same loads. In this test case, random numbers were selected to illustrate the situation. The primary side voltage was set to 15 V and the two secondary side voltages were set to 14 V as referred to the primary side. The inductance values were based on estimates of the real transformer used and converted to the equivalent delta values, and the phase angles were arbitrarily set to $\frac{\pi}{10}$, or 18$^\circ$. While this is purely based on equations and does not include any imperfections that would unbalance the two sides, it is still important to note that the current is split evenly between the two outputs. This behavior is very much desired, as it suggests that the circuit can operate the two outputs at the same level successfully.
The MATLAB script also calculates the RMS currents for each winding. Given a set of $n$ discrete points, the RMS calculation is quite simple.

\[ I_{\text{rms}} = \sqrt{\frac{\sum_{i=1}^{n} I(i)^2}{n}} \]  

(34)

This calculation provided current values for the waveforms seen in Figure 24. $I_{1\text{rms}}$ was 23.8 A, and $I_{2\text{rms}}$ and $I_{3\text{rms}}$ were 11.9 A.
Figure 25: Current Waveforms for Unbalanced Loads

Figure 25, on the other hand, shows an unbalanced system. The second output has a larger load than the first output. The settings changed were that the second output voltage was decreased to 11 V and its phase angle was increased to $\frac{\pi}{5}$, or 36°. Again, the values selected were arbitrary. The fact that the output voltage dropping corresponds with an increase in phase angle is, however, realistic. This is the way the system will typically operate.

As can be seen, then, the current going to each output is no longer the same. This,
of course, is also desired, as the currents should not be the same. However, what is interesting in this figure is that, while the voltage and phase angle for the first output did not change, the current going to this output did change. The calculated currents for this case were 33.3 A for $I_{1\text{rms}}$, 9.0 A for $I_{2\text{rms}}$, and 27.9 A for $I_{3\text{rms}}$. This is an effect of the two outputs being linked through the same transformer. They interfere with the control of each other. Nevertheless, this is acceptable. The control is designed to compensate for such interference, and the desired operating point can be found. The phase shift of the second output should not stay in the same position in that case, and the control would update it to the appropriate angle such that the second output sees the current level that it should. Meanwhile, this changed phase angle would also influence the first output, and thus the first phase angle would also be updated. The correct operating point appears with both phase angles being updated to positions that would not be selected independently.

In Figure 26, the two phase angles for the outputs were modified while the output voltages remained the same. The phase angle for the first output was increased by $\frac{\pi}{60}$, or 3°, and the phase angle for the second output was decreased by the same amount. While the waveforms appear similar in shape to those in Figure 25, the RMS currents are changed once again. $I_{1\text{rms}}$ was now 33.8 A, $I_{2\text{rms}}$ became 11.9 A, and $I_{3\text{rms}}$ decreased to 25.0 A. Thus, the principle of modifying the phase angles to compensate for interactions is demonstrated. The first output maintained its current value, while the second output increased its current substantially. The operating point of the first output was maintained while the operating point of the second changed. This supports the idea that the outputs can be controlled independently.
The current waveforms seen in this program were themselves only created through the equations stated before. Therefore, in order to verify that this waveform generation was accurate, a simple simulation was run in order to verify the shape and magnitude of the current. The simulation circuit appears in Figure 27. The simulation results, seen in Figures 28 and 29, match those found in Figures 24 and 26. The simulation used exactly matching parameters, including voltages, leakage inductances, and phase angles. Thus, the MATLAB code was considered accurate.

The code written in MATLAB to generate these waveforms can be found in Ap-
Figure 27: Simulation Circuit for Three Winding Transformer Power Flow

Appendix B. The code is, like the main controller code, commented fairly extensively for clarity.
Figure 28: Simulated Current Waveforms for Matching Loads

Figure 29: Simulated Current Waveforms for Unbalanced Loads
Transformer Analysis

The power flow analysis explained in the previous section relies on a couple of assumptions in order to arrive at its mathematical representation. In particular, assumptions about the transformer, as seen by the simplification of the transformer model in Figure 21 and Figure 22, allow the power flow analysis to be carried out as it is. Obviously, the closer the transformer is to these simplified models, the more accurate the power flow analysis is. Thus, a brief analysis of the transformer parameters gives some general indication as to the accuracy of the power flow analysis.

The first assumption is that the mutual inductance, and in fact the core and magnetizing impedances in general, are large enough that they can be ignored in the simplified model without significant deviation. In order for this to be true, these impedances should be at least an order of magnitude larger than the series impedances of the transformer. Beyond that, the larger they are relative to the series impedances, the better for the model accuracy. The second assumption is that the series resistances, the elements that constitute the resistance of the windings, are very small. They should be insignificant next to the series leakage inductances. Once again, the smaller they are relative to the leakage inductances, the better. This is slightly counterintuitive, as the leakage inductances themselves are desired to be small in general. Nevertheless, the series resistances should be smaller still.

In order to verify that the transformer being used at least approximately follows these criteria, a few tests were performed to verify the transformer parameters. Typically, transformer parameters can be found through tests referred to as short circuit or open circuit tests. In these tests, the secondary winding is either left unconnected
or connected directly back to itself, and then rated voltage or rated current is applied to the input, depending on which test is being performed. By measuring the input voltage, input current, and input active power, the parameters can all be found. However, the tests had to be modified slightly to be performed at the operating frequency with the available equipment. This was considered acceptable, as relative values were all that were needed to determine the accuracy of the power flow analysis.

The first test performed was the open circuit test previously described. The two output windings were left unconnected, while the input winding received its rated voltage. Then, the voltage, current, and active power were measured. In this case, the test was carried out in the typical manner without alteration. The values found, here referred to as $V_{test}$, $I_{test}$, and $P_{test}$, provided the answers with some simple calculations.

\begin{align*}
R_{core} &= \frac{V_{test}^2}{P_{test}} = 32.5 \, \Omega \\
I_{real} &= \frac{P_{test}}{V_{test}} = 0.461 \, A \\
I_{im} &= \sqrt{I_{test}^2 - I_{real}^2} = 0.395 \, A \\
X_{mag} &= \frac{V_{test}}{I_{im}} = 38.0 \, \Omega \\
L_{mag} &= \frac{X_{mag}}{\omega} = 30.2 \, \mu H
\end{align*}

55
These values for core and magnetizing impedances reflect what was originally expected. As the leakage inductances were originally expected to be less than 1 \( \mu \)H, the mutual impedance would indeed be an order of magnitude larger. To verify this, though, the leakage inductances and series resistances still had to be calculated. In this test, the process differed from the typical short circuit test, though. Due to equipment limitations, a different method was used. First, the series resistances were estimated based on the resistivity of copper and the geometry of the windings. Each winding has a width of 1 inch and a thickness of 0.01 inches. Thus, the series resistances could be calculated for each winding based on its length. The resistivity used in these calculations is the value measured at a temperature of 25 °C.

\[
R = \frac{\rho l}{A} = \frac{16.8 \Omega \text{nm} \cdot l}{0.01 \text{in}^2} = \frac{6.614 \cdot 10^{-7} \Omega \text{m} \cdot l}{0.01 \text{in}^2}
\]  

The lengths of the windings are about 9 inches for the primary winding and about 24 inches for the two secondary windings. Thus, the resistances were found to be 5.953 \( \cdot 10^{-4} \) \( \Omega \) for the primary side and 1.587 \( \cdot 10^{-3} \) \( \Omega \) for the secondary sides.

Finally, to find the leakage inductances, the waveform of the transformer current and voltage were viewed simultaneously with an oscilloscope while a short circuit test was performed. A square wave of voltage at 200 kHz was applied to the primary winding with one output short circuited. The voltage was increased until the current reached a reasonable level. This resulted in a fairly close approximation of a triangle wave from the current with an average value of 0 A. At any point where the current was at 0 A, the voltage across the winding resistance would by necessity by 0 V. Thus, by looking at the slope of the current near a point where it crossed 0 A as it oscillated,
a condition could be found where the applied voltage of the square wave was applied entirely across the leakage inductances of the primary and secondary in series. The voltage for this short duration maintained a DC value. This made calculating the sum of the two leakage inductances very simple. The values calculated were values as referred to the primary side.

\[
L_{pri} + L_{sec} = \frac{V_{in} \cdot \Delta t}{\Delta I} = \frac{0.4 \cdot 800 \text{ ns}}{0.6 \text{ A}} = 0.53 \mu H \tag{41}
\]

The test was repeated two more times. Once, the primary was measured along with the other secondary winding short circuited. The third time, a secondary winding was used for the input while the other secondary winding was short circuited to measure the two secondary inductances without the primary winding. Given the three equations, the three leakage inductances could easily be found through algebraic substitution.

From these calculations, the series inductances were found to be 0.205 \(\mu H\) for the primary winding and 0.325 \(\mu H\) for the secondary windings. These winding values are all referred to the transformer primary side. Additionally, the impedance of these inductances are 0.258 \(\Omega\) and 0.408 \(\Omega\), respectively. Thus, the inductances are clearly more than an order of magnitude larger than the series resistances. The series resistances, then, can be reasonably ignored in the transformer model. Additionally, the leakage impedances are more than an order of magnitude smaller than the core and magnetizing impedances. This allows for the mutual impedances to be neglected reasonably as well. Therefore, the power flow analysis is acceptable for the transformer used for this device.
Control Analysis

The control concept for the PV simulator was fairly simple. Each output just utilized current control, based on the output voltage level, to operate at specific points along the I-V curve of the photovoltaic characteristic. Thus, as more current is required from an output, greater than the current that is flowing to that output, the voltage level of that output capacitor would decrease. This increases the current control setpoint to match the new voltage level, allowing the control algorithm to move to a new operating point on the I-V curve. Likewise, as less current is needed, the voltage would rise on the capacitor, causing the current setpoint to decrease. Tracking the I-V curve of a photovoltaic panel was, in principle, straightforward.

However, other factors made the control more demanding. Due to the fact that there are two outputs, the outputs need to operate independently in order to accurately track the photovoltaic I-V curve. Of course, as the two outputs are linked through the transformer, there is an interaction between the two. The current between the two outputs, $I_{23}$, must be non-zero whenever the two outputs are not operating at the exact same point. Additionally, following from the power flow analysis, the farther apart the two operating points of the outputs are, the greater $I_{23}$ will be. Thus, to maintain independent operation of the outputs, this current must be compensated.

Under many conditions, this current would not be a factor. Whenever the two outputs were operated at the same phase shift, aside from minor flaws in the system, the outputs would be exactly matched. Mathematically, for identical loads, this would indicate that current $I_{23}$ would be zero. This comes from the fact that under
identical loads, the two outputs would operate in the exact same manner, having identical output voltages, output currents, and phase angles. Thus, the two outputs, being exactly matched, would have no voltage difference at any point in time, and therefore would have no current between them. Realistically, the $I_{23}$ developed should maintain a value at least near to zero, depending on component mismatches. Under these conditions the control would work completely as normal, with no noticeable flaws. Additionally, the two outputs should be operating in this matched manner in several situations. When the outputs are connected to each other, either in series or in parallel, they should operate at the same point. They would be forced to have either the same current or the same voltage, and either way this should force them to the same operating point and the same phase shift. Thus, in these situations, $I_{23}$ would be at or near zero, and therefore it would not interfere with the control.

However, in the event where the two outputs are not operating at the same point, one will have a larger phase shift than the other. Additionally, based on the analysis assumption that both outputs appear as sources to the transformer, they will of course appear as sources to one another as well. Thus, the output that has a larger phase shift will draw current from the output with the smaller phase shift in addition to the current it draws from the primary side source. This combination of factors forces the current at that output to a level that is higher than desired. Meanwhile, the output with the smaller phase shift has less current going to its output because of the current being drawn by the other output. In order to adjust, the two phase shifts will have to move closer to one another. The smaller phase shift will increase while the larger phase shift will decrease.
This behavior works quite well for a couple of reasons. First, should the outputs be affected enough that the voltage levels change, the controller would naturally want to adjust the phase shifts in this manner anyways. The output with the smaller phase shift would be receiving too little current, so its capacitor would discharge as the load received the desired current. Thus, the voltage level would start to decrease and the current reference would increase, causing the phase shift to increase. Likewise, if the output with the larger phase shift continued to receive too much current, the capacitor would start to charge and increase its voltage level. At a higher voltage level, the current reference for that output would begin to decrease. This causes the phase shift to decrease. Thus, the control algorithm for each output does not fight against the interaction of the two outputs. Rather, the control algorithm will naturally want to adjust for that interaction.

Secondly, the fact that the two phase shifts would need to operate closer together than they would if they were not linked by the transformer also helps. When the two outputs operate at the same point, the interaction does not exist, and the current control functions as it would for a single output. However, as the two outputs move to operating points farther apart, the phase shifts move farther apart. The interaction between the two outputs, however, does not fight against this either. The interaction between the two outputs as represented by current $I_{23}$ would push the output currents to their operating points faster. Essentially, the phase shifts would need to change less than they normally would to reach the correct operating points. By this logic, any pair of operating points is reachable, as every pair of operating points exists within the range of correct operating points appropriate for a single output. This
essentially ensures that for most pairs of operating points, the appropriate phase shifts are achievable.

Furthermore, the operating points can be achieved without deviating from the I-V curve of photovoltaic panels. Despite the alteration of the phase angles of the outputs, the current tracking is still based on the output currents rather than the transformer currents. Thus, as the phase angles move to compensate for current $I_{23}$, the total current flowing to each output will include the correct output current in addition to the added or subtracted $I_{23}$ current. The current control simply moves to the correct phase angles for the two output voltages. The nature of the integral term of the controller forces it to compensate for the error injected by the $I_{23}$ term.

The limitations of this are certainly worth noting, however. In particular, the case where one output has no load and the other output has a non-zero load causes a small flaw in performance. In this case, there would be a phase angle between the two outputs, and a non-zero $I_{23}$ would develop. However, because the first output has no load, the current sensor would accurately detect 0 A of current at its output. Thus, the current at this output is correct, and the phase angle would not update to compensate for the $I_{23}$ that is developed. As such, the first output capacitor, acting as a source, would begin to decrease its voltage. Once the voltage decreased to 44.5 V, the next value in the data table below 45 V, a current would be directed to the output again. At this point, the phase angle would adjust to compensate for the appropriate current at 44.5 V in addition to the $I_{23}$ being drawn from this output. Additionally, as there is no load, the current indicated by the reduced voltage level would recharge the capacitor to 45 V. The effect, thus, is minimal, and would appear
as a minor fluctuation in the voltage level of the unloaded output. However, this effect would not exist in a real photovoltaic panel. While this effect may be negligible for many applications, especially considering there is no load at that point in time, it is noticeable regardless.
Chapter 5: Experimental Results

In order to verify the performance of the device, some experimental results were desired. Before the experiments could be carried out, though, other circuits needed verified. Thus, the secondary circuits were all tested before applying power to the primary power flow circuit. The protection circuits and coded protection algorithm, the gate drive signals, and the voltage and current sensors were all verified. In most cases, this was performed by applying specific voltage levels as input signals to the circuits. For the gate drive signals, the signals were simply monitored on an oscilloscope to verify the timing. Once all these circuits were shown to be functional, then, some preliminary tests could be carried out to test the primary circuit.

First, the microcontroller code was modified such that the full range of photovoltaic operation could be achieved at a lower power level. In this way, low power testing could be applied across, ideally, the full operating range. Next, with the microcontroller operating and the protection circuits ready, an initially low input voltage was applied. This allowed for the first results to be collected. By manipulating the phase angle of an output manually in the code, the MOSFET voltage waveforms could be viewed at specific phase angles. Viewing these waveforms would indicate that the switches were operating as desired.
Figures 30 and 31 show the voltage waveforms of two MOSFETs from the primary side and one secondary side. They refer to the voltages as measured from drain to source. As can be seen, the switches are operating at the desired phase angles. However, the waveform for switch 2 is abnormal. Given that the system still functioned correctly and that the other waveforms appear to be fine, it is likely this is simply due to a hardware flaw, such as a snubber element not functioning. The phase angles match up as expected, and most of the voltage waveforms are fairly clean square waves. This also suggests that the snubbers appear to be tuned appropriately, as the voltage overshoot and ripple is within acceptable limits. In addition, the gate
Figure 31: MOSFET voltage waveforms with a Thirty Degree Phase Angle

drive circuits are basically verified by this test.

With all aspects of the system seemingly operational, the next step was to test the system with its intended operation. By applying both an input voltage and a load, as well as recording the output voltage and current levels, the control algorithm could be tested. By recording the voltage and current operating points of the output for various loads, and then comparing those points to the I-V characteristic that the control algorithm is trying to match, the performance of the control algorithm can be analyzed.

The data points collected in Figure 32 match the desired I-V curve fairly closely,
Figure 32: I-V Operating Points

suggesting that the control algorithm is functioning correctly. However, the data collected only covers half of the I-V curve. This is because beyond this point, for lower voltages, the operating point begins to diverge from the desired curve. At the operating point measured with the lowest voltage, the control angle was already at its minimum. Thus, while the control algorithm was, in principle, functioning correctly, it could not control the current to stay as low as 8.33 A for a larger load that would reduce the voltage lower. This result uncovered a design flaw with the system. However, it can be rectified by redesigning the transformer. This is discussed
further in the section detailing future work.

To verify that the control algorithm would still function at lower voltage levels, a second test was performed. In this case, once the region where the control algorithm was already at its minimum was reached, the input voltage was manually reduced to bring the operating range of the device back to a controllable point. With the input voltage lower, the output voltage reduced as well. Correspondingly, using a resistive load, the output current reduced. This was lowered until the phase angle was automatically increased to maintain the current level of 8.33 A. Thus, the points on the curve in Figure 33 that appear where they did not in Figure 32 are artificially generated. Nevertheless, they still verify that the control algorithm can function in this range in principle, if not currently in practice.

Finally, to verify the control adjustment of the output when the load changes, dictating a current and voltage level change, the output was monitored during a load change. For a load increase, the current will increase, and it was expected to increase quickly due to the large output capacitors. Likewise, the output voltage was expected to decrease more slowly, again due to the large output capacitors. This transition, in general, though, should be relatively quick and as smooth as possible.

From Figure 34, it can be seen that the current does indeed increase very quickly for a load change of 0 A to a little less than 1 A. The transition for both current and voltage appears fairly smooth. Also, the speed with which the output current and voltage both change to the new, stable operating point is fairly quick. Within 10 ms, the output has reached its new operating point in this test.
Figure 33: I-V Operating Points with Manipulation of Input Voltage
Figure 34: Output Current and Voltage with Load Change
Chapter 6: Future Work

As mentioned in the experimental results, some revisions to the current system are required. The controllable range for the current system is not large enough to cover the full range of the desired I-V curve. This is largely due to the transformer design. First, the leakage inductances are very small. Based on the power flow analysis, smaller leakage inductances lead to larger currents, regardless of the phase angle. Thus, with larger leakage inductances, the current would naturally be lower for a given phase angle. This would mean that a larger phase angle would be required across the full range, and the minimum phase angle would not be reached until a much lower voltage. Second, the input voltage, when operating at rated conditions, would be about 15 V. Given that the low end of the desired output voltage range is 0 V or as close to that point as possible, the current input voltage is too high.

In order to fix these problems, the turns ratio of the transformer can be increased. This action would both increase the leakage inductance and decrease the input voltage. Both of these adjustments would help to bring more of the desired I-V curve within the operating range. However, this is limited by the fact that it would increase the necessary input current. This can be factored into the transformer design, though. Thus, redesigning the transformer for a larger turns ratio would be the first
major step in the future work of this project.

Additionally, multiple other developments for this project have already been considered. The first, and perhaps most obvious, extension would be to increase the number of outputs further. However, some aspects would need redesigned to accommodate more outputs, so this perhaps would be more along the lines of a redesign of the device as a whole, as a second generation of the project. However, having more outputs would be extremely useful. This would allow for, clearly, more panels to be connected in series or parallel. With enough outputs, series strings of panels could even be connected in parallel, more closely emulating practical installations. Taken even further, this could be utilized, with some more substantial modifications, to have enough outputs to effectively simulate a section of a photovoltaic power plant. As these power plants utilize a very large number of photovoltaic panels, it would be extremely impractical to try to test equipment for these devices without extensive test equipment such as described here.

Another development, and one that is certainly more immediately possible to realize, is the addition of a communication link to the controller. Through this, a computer connected to the controller, along with a graphical user interface, could modify the behavior of the photovoltaic simulator in real time. In such a way, a temporary shading effect could be illustrated simply by modifying the data table used in the control algorithm. With new data points, an entirely new I-V curve could be tracked by the device. Also, through this user interface, each output could be controlled separately, enabling the shading effect mentioned to affect only the specified panels simulated. This gives a great deal of flexibility to the test platform.
Chapter 7: Conclusion

In this project, a test platform based on photovoltaic simulators was proposed. The main concept was to utilize multiple outputs through one input on the same transformer to simplify control, minimize size, and at the same time provide multiple photovoltaic panel simulator outputs. This would be a versatile and compact photovoltaic test platform. Given the developing interest in photovoltaic research, the test platform should see substantial use.

The system was designed based on a full bridge architecture for several reasons. With this architecture, a phase shifting control algorithm could be implemented to control power flow. The power flow analysis supports the control method, even for multiple outputs, indicating that increasing the phase angle between the input and an output increases the current that flows to that output. The transformer analysis supports the accuracy of the power flow analysis, as well.

Experimental results uncovered a design flaw that limits the controllable range fairly significantly. However, at the same time, the experimental results indicate that the system design and control algorithm are valid. The system was able to successfully track the photovoltaic characteristic curve. The full bridge architecture and switching waveforms were verified. The only element of the design that needs signif-
icant adjustment in order to achieve full operation at this point is the transformer. Essentially, though the system did not achieve all the functionality desired, the results that were found do indicate that the concept is valid.
Bibliography


Appendix A: Microcontroller Code

// based off of the free file Example_281xEvPwm.c from Texas Instruments
#include “DSP281x_Device.h” // contains TI created data
#include “DSP281x_Examples.h” // contains TI created functions
void init_eva(void); // initialize event managers
void init_evb(void);
interrupt void T1PR(void); // period and underflow interrupts for PWM
interrupt void T1UN(void);
interrupt void ADCisr(void); // interrupt to read ADC’s and operate PI control

// Global variables
Uint16 Counter = 0; // used for timing
Uint16 Vin = 0;
Uint16 VoutA = 0;
Uint16 VoutB = 0;
Uint16 In = 0;
Uint16 IoutA = 0;
Uint16 IoutAPrev = 0; // additional Iout variables for filters
Uint16 IoutA0 = 0;
Uint16 IoutA0Prev = 0;
Uint16 IoutB = 0;
Uint16 IoutBPrev = 0;
Uint16 IoutB0 = 0;
Uint16 IoutB0Prev = 0;
Uint16 VoutAPrev = 0; // additional Vout variables for filters
Uint16 VoutA0 = 0;
Uint16 VoutA0Prev = 0;
Uint16 VoutB0 = 0;
Uint16 VoutB0Prev = 0;
Uint16 alpha1A = 0x005E; // phase angles. 0x005E is equivalent to zero degrees
Uint16 alpha2A = 0x005E;
Uint16 alpha1B = 0x005E;
Uint16 alpha2B = 0x005E;
Uint16 IrefA = 0; // current reference variables
Uint16 IrefB = 0;
Uint16 GainPA = 1; // effective value 1/3
Uint16 GainIA = 1; // effective value 1/30
Uint16 GainPB = 1; // effective value 1/3
Uint16 GainIB = 1; // effective value 1/30
Uint16 ErrA = 0; // PI controller variables
Uint16 Prev_ErrA = 0;
Uint16 ErrB = 0;
Uint16 Prev_ErrB = 0;
Uint16 PErrA = 0;
Uint16 IErrA = 0;
Uint16 PErrB = 0;
Uint16 IErrB = 0;
Uint16 PVCurve[91]; // data table for I-V curve of PV panel
Uint16 PosCheckA = 1; // used in PI controller
Uint16 PosCheckB = 1; // 1 for positive, 0 for negative
Uint16 IErrAPos = 1; // used in PI controller
Uint16 IErrBPos = 1; // integral error polarity
Uint16 DownShift = 5; // decrease Iout readings by 2^5 to prevent overflows
Uint16 IErrAMax; // limits integral error to prevent overflow
Uint16 IErrBMax;
Uint16 OverCurrent = 0; // variable for overcurrent protection
Uint16 alpha1AMax = 0x005E; // 0 degrees shift until rated voltage reached
Uint16 alpha1BMax = 0x005E;
void main(void)
{
    // TI function to initialize system control
    InitSysCtrl();
    // TI function to initialize GPAMUX and GPBMUX
    EALLOW;
    // Enable PWM pins
    GpioMuxRegs.GPAMUX.all = 0x00FF; // EVA PWM 1-6 pins
    GpioMuxRegs.GPBMUX.all = 0x00FF; // EVB PWM 7-12 pins
    EDIS;
    // Disable interrupts for initialization functions
    DINT;
    // Initialize PIE control registers
InitPieCtrl();
// Disable interrupts and clear flags:
IER = 0x0000;
IFR = 0x0000;
// Initialize the PIE vector table
InitPieVectTable();
EALLOW;
PieVectTable.T1PINT = &T1PR; // add used interrupts to PIE vector table
PieVectTable.T1UFINT = &T1UN;
PieVectTable.ADCINT = &ADCisr;
EDIS;
// PV Curve Characteristic data array
PVCurve[0] = 2274;
PVCurve[1] = 2274;
PVCurve[2] = 2274;
PVCurve[3] = 2274;
PVCurve[4] = 2274;
PVCurve[5] = 2274;
PVCurve[6] = 2274;
PVCurve[7] = 2274;
PVCurve[8] = 2274;
PVCurve[9] = 2274;
PVCurve[10] = 2274;
PVCurve[12] = 2274;
PVCurve[13] = 2274;
PVCurve[14] = 2274;
PVCurve[15] = 2274;
PVCurve[16] = 2274;
PVCurve[17] = 2274;
PVCurve[18] = 2274;
PVCurve[19] = 2274;
PVCurve[20] = 2274;
PVCurve[21] = 2274;
PVCurve[22] = 2274;
PVCurve[23] = 2274;
PVCurve[24] = 2274;
PVCurve[25] = 2274;
PVCurve[26] = 2274;
PVCurve[27] = 2274;
PVCurve[28] = 2274;
PVCurve[29] = 2274;
PVCurve[30] = 2274;
PVCurve[31] = 2274;
PVCurve[32] = 2274;
PVCurve[33] = 2274;
PVCurve[34] = 2274;
PVCurve[35] = 2274;
PVCurve[36] = 2274;
PVCurve[37] = 2274;
PVCurve[38] = 2274;
PVCurve[39] = 2274;
PVCurve[40] = 2274;
PVCurve[41] = 2274;
PVCurve[42] = 2274;
PVCurve[43] = 2274;
PVCurve[44] = 2274;
PVCurve[45] = 2274;
PVCurve[46] = 2274;
PVCurve[47] = 2274;
PVCurve[48] = 2274;
PVCurve[49] = 2274;
PVCurve[50] = 2274;
PVCurve[51] = 2274;
PVCurve[52] = 2273;
PVCurve[53] = 2273;
PVCurve[54] = 2273;
PVCurve[55] = 2272;
PVCurve[56] = 2272;
PVCurve[57] = 2271;
PVCurve[58] = 2270;
PVCurve[59] = 2269;
PVCurve[60] = 2267;
PVCurve[61] = 2265;
PVCurve[62] = 2262;
PVCurve[63] = 2259;
PVCurve[64] = 2254;
PVCurve[65] = 2248;
PVCurve[66] = 2240;
PVCurve[67] = 2230;
PVCurve[68] = 2217;
PVCurve[69] = 2201;
PVCurve[70] = 2180;
PVCurve[71] = 2154;
PVCurve[72] = 2122;
PVCurve[73] = 2082;
PVCurve[74] = 2034;
PVCurve[75] = 1976;
PVCurve[76] = 1908;
PVCurve[77] = 1830;
PVCurve[78] = 1740;
PVCurve[79] = 1640;
PVCurve[80] = 1528;
PVCurve[81] = 1406;
PVCurve[82] = 1273;
PVCurve[83] = 1131;
PVCurve[84] = 980;
PVCurve[85] = 821;
PVCurve[86] = 654;
PVCurve[87] = 480;
PVCurve[88] = 300;
PVCurve[89] = 114;
PVCurve[90] = 0;

// IErr limit - prevent overflow
// minimum IErrAMax must be able to completely counteract P part of PI controller
IErrAMax = (0x005E + GainPA * (2274 >> DownShift)/3)*30/GainIA;
IErrBMax = (0x005E + GainPB * (2274 >> DownShift)/3)*30/GainIB;

// initialize the event managers and ADCs
init_eva();
init_evb();
InitAdc();

// enable desired interrupts
PieCtrlRegs.PIEIER1.all = M_INT6; // Enable ADCINT
PieCtrlRegs.PIEIER2.all = (M_INT4 | M_INT6); // T1PR and T1UN int enable
IER |= (M_INT1 | M_INT2 | M_INT4); // Enable CPU Interrupts 1, 2, and 4
EINT; // Enable Global interrupt INTM
ERTM; // Enable Global realtime interrupt DBGM

// Configure ADC
AdcRegs.ADCMAXCONV.all = 0x0005; // Setup 6 conv’s on SEQ1
AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x00; // Setup ADCINA0 as 1st SEQ1 conv.
AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 0x1; // Setup ADCINA1 as 2nd SEQ1 conv.
AdcRegs.ADCCHSELSEQ1.bit.CONV02 = 0x2; // Setup ADCINA2 as 3rd SEQ1 conv.
AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3; // Setup ADCINA3 as 4th SEQ1 conv.
AdcRegs.ADCCHSELSEQ2.bit.CONV04 = 0x4; // Setup ADCINA4 as 5th SEQ1 conv.
AdcRegs.ADCCHSELSEQ2.bit.CONV05 = 0x5; // Setup ADCINA5 as 6th SEQ1 conv.
AdcRegs.ADCTRL3.bit.ADCCLKPS = 0x3; // ADC prescaler to give 25MHz clock
AdcRegs.ADCTRL2.bit.EVA_SOC_SEQ1 = 1; // Enable EVASOC to start SEQ1
AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1 = 1; // Enable SEQ1 interrupt (every EOS)
//Main Loop
for(;;); //interrupt driven code, so main loop is an infinite loop
}
void init_eva()
{
// EVA Configure T1PWM, T2PWM, PWM1-PWM6
// Initialize EVA Timer1
EvaRegs.T1PR = 0x00BC; // Timer1 period
EvaRegs.T1CMPR = 0x000A; // Timer1 compare
EvaRegs.T1CNT = 0x0000; // Timer1 counter
EvaRegs.GPTCONA.bit.T1TOADC = 1; //ADC interrupts from GPT1
EvaRegs.EVAIMRA.bit.T1PINT = 1; //Enable interrupt
EvaRegs.EVAIFRA.bit.T1PINT = 1;
EvaRegs.EVAIMRA.bit.T1UFINT = 1; //Enable interrupt
EvaRegs.EVAIFRA.bit.T1UFINT = 1;
// TMODE = continuous up/down
// Timer enable
// Timer compare enable
EvaRegs.T1CON.all = 0x0842;
// Initialize EVA Timer2
EvaRegs.T2PR = 0x00BC; // Timer2 period
EvaRegs.T2CMPR = 0x005E; // Timer2 compare
EvaRegs.T2CNT = 0x0000; // Timer2 counter
// TMODE = continuous up/down
// Timer enable
// Timer compare enable
EvaRegs.T2CON.all = 0x0842;
// Setup T1PWM and T2PWM
// Drive T1/T2 PWM by compare logic
EvaRegs.GPTCONA.bit.TCMPOE = 1;
// Polarity of GP Timer 1 Compare = Active low
EvaRegs.GPTCONA.bit.T1PIN = 1;
// Polarity of GP Timer 2 Compare = Active high
EvaRegs.GPTCONA.bit.T2PIN = 2;
// Enable compare for PWM1-PWM6
EvaRegs.CMPR1 = 0x005E; // zero degree phase for PWM1-PWM6
80
EvaRegs.CMPR2 = 0x005E;
EvaRegs.CMPR3 = 0x005E;
// output pin 1 CMPR1 - active high (Switch 1)
// output pin 2 CMPR1 - active low (Switch 2)
// output pin 3 CMPR2 - active high (Switch 4) 4 and 3 swapped for deadband functionality
// output pin 4 CMPR2 - active low (Switch 3)
// output pin 5 CMPR3 - active high (Switch 5)
// output pin 6 CMPR3 - active low (Switch 6)
EvaRegs.ACTRA.all = 0x0666; //PWM 1,3,5 active high; 2,4,6 active low
EvaRegs.DBTCONA.all = 0x04E0; // Enable deadband (roughly 0.04 us)
// additional deadband not needed due to delay in hardware
EvaRegs.COMCONA.all = 0xA600;
}
void init_evb()
{
// EVB Configure T3PWM, T4PWM and PWM7-PWM12
// Initialize EVB Timer3
// Timer3 controls T3PWM and PWM7-12
EvbRegs.T3PR = 0x00BC; // Timer3 period
EvbRegs.T3CMPR = 0x005E; // Timer3 compare
EvbRegs.T3CNT = 0x0023; // Timer3 counter - delay to sync with T1
EvbRegs.EVBIMRA.bit.T3PINT = 1; //Enable interrupt
EvbRegs.EVBIFRA.bit.T3PINT = 1;
EvbRegs.EVBIMRA.bit.T3UFINT = 1; //Enable interrupt
EvbRegs.EVBIFRA.bit.T3UFINT = 1;
// TMODE = continuous up/down
// Timer enable
// Timer compare enable
EvbRegs.T3CON.all = 0x0842;
// Initialize EVB Timer4
// Timer4 controls T4PWM
EvbRegs.T4PR = 0x00BC; // Timer4 period
EvbRegs.T4CMPR = 0x005E; // Timer4 compare
EvbRegs.T4CNT = 0x0000; // Timer4 counter
// TMODE = continuous up/down
// Timer enable
// Timer compare enable
EvbRegs.T4CON.all = 0x0842;
// Setup T3PWM and T4PWM
// Drive T3/T4 PWM by compare logic
EvbRegs.GPTCONB.bit.TCMPOE = 1;
// Polarity of GP Timer 3 Compare = Active low
EvbRegs.GPTCONB.bit.T3PIN = 1;
// Polarity of GP Timer 4 Compare = Active high
EvbRegs.GPTCONB.bit.T4PIN = 2;
// Enable compare for PWM7-PWM12
EvbRegs.CMPR4 = 0x005E;
EvbRegs.CMPR5 = 0x005E;
EvbRegs.CMPR6 = 0x005E;
// Compare action control. Action that takes place
// on a compare event
// output pin 1 CMPR4 - active high (Switch 8) 8 and 7 swapped
// output pin 2 CMPR4 - active low (Switch 7)
// output pin 3 CMPR5 - active high (Switch 9)
// output pin 4 CMPR5 - active low (Switch 10)
// output pin 5 CMPR6 - active high (Switch 12) 12 and 11 swapped
// output pin 6 CMPR6 - active low (Switch 11)
EvbRegs.ACTRB.all = 0x0666;
EvbRegs.DBTCONB.all = 0x04E0; // Enable deadband, same as EVA
EvbRegs.COMCONB.all = 0xA600;
}
interrupt void T1PR(void)
{
    EvaRegs.CMPR3 = alpha1A; // update compare values for rising edge
    EvbRegs.CMPR4 = alpha1A; // switches 5-8
    EvbRegs.CMPR5 = alpha1B; // switches 9-12
    EvbRegs.CMPR6 = alpha1B;
    EvaRegs.EVAIMRA.bit.T1PINT = 1; //Enable more interrupts
    EvaRegs.EVAIMRA.all = BIT7; //Clear interrupt
    PieCtrlRegs.PIEACK.all = PIEACK_GROUP2; //acknowledge interrupt return;
}
interrupt void T1UN(void)
{
    EvaRegs.CMPR3 = alpha2A; // update compare values for falling edge
    EvbRegs.CMPR4 = alpha2A; // switches 5-8
    EvbRegs.CMPR5 = alpha2B; // switches 9-12
    EvbRegs.CMPR6 = alpha2B;
    EvaRegs.EVAIMRA.bit.T1UFINT = 1; //Enable more interrupts
    EvaRegs.EVAIFRA.all = BIT9; //Clear interrupt
    PieCtrlRegs.PIEACK.all = PIEACK_GROUP2; //acknowledge interrupt return;

return;
}
interrupt void ADCisr(void)
{
  Vin = AdcRegs.ADCRESULT0 >>4; // read values through ADCs
  VoutA0 = AdcRegs.ADCRESULT1 >>4; // digital low pass filter initial values
  VoutB0 = AdcRegs.ADCRESULT2 >>4;
  Iin = AdcRegs.ADCRESULT3 >>4;
  IoutA0 = AdcRegs.ADCRESULT4 >>4;
  IoutB0 = AdcRegs.ADCRESULT5 >>4;
  IoutA0 = 4095 - IoutA0; // 0-4095 now corresponds to 0-15 A
  IoutB0 = 4095 - IoutB0;
  Iin = 4095 - Iin;
  VoutA = VoutA0/16 + VoutA0Prev/16 + (VoutAPrev*14)/16; // digital low pass filters
  VoutB = VoutB0/16 + VoutB0Prev/16 + (VoutBPrev*14)/16;
  VoutA0Prev = VoutA0;
  VoutB0Prev = VoutB0;
  VoutAPrev = VoutA;
  VoutBPrev = VoutB;
  IoutA = IoutA0/16 + IoutA0Prev/16 + (IoutAPrev*14)/16; // digital low pass filters
  IoutB = IoutB0/16 + IoutB0Prev/16 + (IoutBPrev*14)/16;
  IoutA0Prev = IoutA0;
  IoutB0Prev = IoutB0;
  IoutAPrev = IoutA;
  IoutBPrev = IoutB;
  if(IoutA >= 3276) || IoutB >= 3276) // output current larger than 12 A
  {
    if(OverCurrent >= 19) // requires 20 consecutive overcurrent readings
      { // to trigger
        EvaRegs.ACTRA.all = 0x0000; // shut down all gate drive signals
        EvbRegs.ACTRB.all = 0x0000;
        OverCurrent = 0;
      }
    else
      { // reset overcurrent protection, no triggering
        OverCurrent++;
      }
  }
  else
    { // reset overcurrent protection, no triggering
      OverCurrent = 0;
    }
}
if(Counter >= 10000) // occurs 20 times per second
{
    if(VoutA >= 2730) // VoutA larger than 40 V on 60 V scale
    {
        alpha1AMax = 0x007C; // increase maximum phase angle to 30 degrees
    }
    if(VoutB >= 2730) // VoutB larger than 40 V
    {
        alpha1BMax = 0x007C; // increase maximum phase angle to 30 degrees
    }
    if(VoutA >= 3060) // limit maximum voltage reading to 45 V
    {
        // higher readings do not alter control
        VoutA = 3060;
    }
    if(VoutB >= 3060)
    {
        VoutB = 3060;
    }
    IrefA = PVCurve[(VoutA/34)]; // scales VoutA to 0-90
    IrefB = PVCurve[(VoutB/34)]; // scales VoutB in 0-90
    Counter = 0; // reset counter
}
else
{
    Counter = Counter + 1;
}
if(Counter % 100 == 0) // occurs 2000 times per second
{
    Prev_ErrA = ErrA; // previous error updated
    Prev_ErrB = ErrB;
    // PI controller implemented in this manner to avoid floating point math
    // this increases the speed of the operation in the selected DSP chip considerably
    if(IoutA < IrefA) // positive error, proportional error will be positive
    {
        ErrA = (IrefA - IoutA) >> DownShift; // 0-127 int range, scaled down for int math
        PErrA = ErrA; // set proportional error
        if(PosCheckA == 1) // last cycle had positive error
        {
            if(IErrAPos == 1) // integral error is positive
            {

48

84
IErrA += Prev_ErrA; // both errors positive, sum
} else if(IErrA >= Prev_ErrA) // integral error negative and greater than
    // last cycle error
    IErrA -= Prev_ErrA; // subtract smaller value, error stays negative
} else // integral error negative and less than last cycle error
    { IErrA = Prev_ErrA - IErrA; // subtract smaller value
      IErrAPos = 1; // integral error becomes positive
    }
else // last cycle had negative error
    { if(IErrAPos == 0) // integral error is negative
        { IErrA += Prev_ErrA; // both errors negative, sum
        }
    else if(IErrA >= Prev_ErrA) // integral error positive and greater than
        { // last cycle error
          IErrA -= Prev_ErrA; // subtract smaller value, error stays positive
        }
    else // integral error positive and less than last cycle error
        { IErrA = Prev_ErrA - IErrA; // subtract smaller value
          IErrAPos = 0; // integral error becomes negative
        }
    }
if(IErrA >= IErrAMax) // limit maximum integral error
    { IErrA = IErrAMax;
    }
if(IErrAPos == 1) // positive integral error
    { alpha1A = 0x005E + GainPA * PErrA / 3 + GainIA * IErrA / 30;
    } // integral error added with positive proportional error
else // negative integral error
    { if(GainPA * PErrA / 3 >= GainIA * IErrA / 30) // proportional error larger than
        { // integral error
          alpha1A = 0x005E + GainPA * PErrA / 3 - GainIA * IErrA / 30;
        }
} // positive proportional error plus negative integral error
else // negative integral error larger than positive proportional error
{
    alpha1A = 0x005E; // phase must be at minimum
}

PosCheckA = 1; // confirms there was positive error this cycle
else // negative error this cycle, but same process as above
{
    ErrA = (IoutA - IrefA) >> DownShift; // 0-127 int range, scaled down for int math
    PErrA = ErrA;
    if(PosCheckA == 1) // last cycle had positive error
    {
        if(IErrAPos == 1)
        {
            IErrA += Prev_ErrA;
        }
        else if(IErrA >= Prev_ErrA)
        {
            IErrA -= Prev_ErrA;
        }
        else
        {
            IErrA = Prev_ErrA - IErrA;
            IErrAPos = 1;
        }
    }
    else // last cycle had negative error
    {
        if(IErrAPos == 0)
        {
            IErrA += Prev_ErrA;
        }
        else if(IErrA >= Prev_ErrA)
        {
            IErrA -= Prev_ErrA;
        }
        else
        {
            IErrA = Prev_ErrA - IErrA;
        }
    }
}
IErrAPos = 0;
}
}
if(IErrA >= IErrAMax)
{
IErrA = IErrAMax;
}
if(IErrAPos == 1) // positive integral error
{
if(GainIA * IErrA / 30 >= GainPA * PErrA / 3) // integral error larger than
{  // proportional error
alpha1A = 0x005E + GainIA * IErrA / 30 - GainPA * PErrA / 3;
} // positive integral error plus negative proportional error
else // negative proportional error larger than positive integral error
{
alpha1A = 0x005E; // minimum phase
}
}
else // both errors negative
{
alpha1A = 0x005E; // minimum phase
}
PosCheckA = 0; // confirms negative error this cycle
}
if(IoutB < IrefB)
{
ErrB = (IrefB - IoutB)>>DownShift; // 0-127 int range, scaled down for int math
PErrB = ErrB;
if(PosCheckB == 1) // last cycle had positive error
{
if(IErrBPos == 1)
{
IErrB += Prev_ErrB;
}
else if(IErrB >= Prev_ErrB)
{
IErrB -= Prev_ErrB;
}
else
{
IErrB = Prev_ErrB - IErrB;
}
IErrBPos = 1;
}
else // last cycle had negative error
{
  if(IErrBPos == 0)
  {
    IErrB += Prev_ErrB;
  }
  else if(IErrB >= Prev_ErrB)
  {
    IErrB -= Prev_ErrB;
  }
  else
  {
    IErrB = Prev_ErrB - IErrB;
    IErrBPos = 0;
  }
  if(IErrB >= IErrBMax)
  {
    IErrB = IErrBMax;
  }
  if(IErrBPos == 1)
  {
    alpha1B = 0x005E + GainPB * PErrB / 3 + GainIB * IErrB / 30;
  }
  else
  {
    if(GainPB * PErrB / 3 >= GainIB * IErrB / 30)
    {
      alpha1B = 0x005E + GainPB * PErrB / 3 - GainIB * IErrB / 30;
    }
    else
    {
      alpha1B = 0x005E;
    }
  }
  PosCheckB = 1; // confirms positive error this cycle
}
else

88
ErrB = (IoutB - IrefB) >> DownShift; // 0-127 int range, scaled down for int math
PErrB = ErrB;
if(PosCheckB == 1) // last cycle had positive error
{
  if(IErrBPos == 1)
  {
    IErrB += Prev_ErrB;
  }
  else if(IErrB >= Prev_ErrB)
  {
    IErrB -= Prev_ErrB;
  }
  else
  {
    IErrB = Prev_ErrB - IErrB;
    IErrBPos = 1;
  }
} else // last cycle had negative error
{
  if(IErrBPos == 0)
  {
    IErrB += Prev_ErrB;
  }
  else if(IErrB >= Prev_ErrB)
  {
    IErrB -= Prev_ErrB;
  }
  else
  {
    IErrB = Prev_ErrB - IErrB;
    IErrBPos = 0;
  }
}
if(IErrB >= IErrBMax)
{
  IErrB = IErrBMax;
}
if(IErrBPos == 1)
{
if(GainIB * IErrB / 30 >= GainPB * PErrB / 3)
{
alpha1B = 0x005E + GainIB * IErrB / 30 - GainPB * PErrB / 3;
}
else
{
alpha1B = 0x005E;
}
else
{
alpha1B = 0x005E;
}
PosCheckB = 0; // negative error this cycle
}
if(alpha1A > alpha1AMax)
{
alpha1A = alpha1AMax;
}
if(alpha1A < 0x005E)
{
alpha1A = 0x005E;
}
if(alpha1B > alpha1BMax)
{
alpha1B = alpha1BMax;
}
if(alpha1B < 0x005E)
{
alpha1B = 0x005E;
}
a = 0x00BC - alpha1A;
alpha2B = 0x00BC - alpha1B;

// Reinitialize for next ADC sequence
AdcRegs.ADCCTRL2.bit.RST_SEQ1 = 1; // Reset SEQ1
AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1; // Clear INT SEQ1 bit
PieCtrlRegs.PIEACK.all = PIEACK_GROUP1; // Acknowledge interrupt to PIE
return;
Appendix B: MATLAB Analysis Code

% Controlled Power Flow

clear all;
close all;

%% Constants
V1 = 15; % constant voltages over one cycle
V2 = 14;
V3 = 11;
theta2 = pi/10 + pi/60; % first output phase angle
theta3 = pi/5 - pi/60; % second output phase angle
w = 200000*2*pi; % radians per second angular velocity
L12 = 0.3*10^-6; % estimated primary referred inductances
L13 = 0.3*10^-6;
L23 = 0.6*10^-6;

I120 = -1/(2*w*L12)*((V1+V2)*theta2 + (V1-V2)*(pi-theta2)); % This takes Ipeak-peak/2, giving starting negative peaks
I130 = -1/(2*w*L13)*((V1+V3)*theta3 + (V1-V3)*(pi-theta3));
I230 = -1/(2*w*L23)*((V3-V2)*theta2 + (V2+V3)*(theta3-theta2) + (V2-V3)*(pi-theta3));

I12A = (V1+V2)/(w*L12)*theta2 + I120; % values at theta2
I13A = (V1+V3)/(w*L13)*theta2 + I130;
I23A = (V3-V2)/(w*L23)*theta2 + I230;

I12B = (V1-V2)/(w*L12)*(theta3-theta2) + I12A; % values at theta3
I13B = (V1+V3)/(w*L13)*(theta3-theta2) + I13A;
I23B = (V2+V3)/(w*L23)*(theta3-theta2) + I23A;

I12pi = -I120; % corresponding values for second half of cycle
I13pi = -I130;
I23pi = -I230;
I12pA = -I12A;
I13pA = -I13A;
\[ I_{23pA} = -I_{23A}; \]
\[ I_{12pB} = -I_{12B}; \]
\[ I_{13pB} = -I_{13B}; \]
\[ I_{23pB} = -I_{23B}; \]

%%% Variables
theta = 0:pi/1800:2*pi; % cycle angle in tenths of degrees
II2 = zeros(1,length(theta)); % intiaze vector lengths
II3 = zeros(1,length(theta));
II3 = zeros(1,length(theta));

%%% Current Equations: Section 1
% calculate current between 0 and theta2
for i = 1:(find(theta > theta2,1)-1)
    II2(i) = (V1+V2)/(w*L12)*theta(i) + II20;
    II3(i) = (V1+V3)/(w*L13)*theta(i) + II30;
    II3(i) = (V3-V2)/(w*L23)*theta(i) + II30;
end

%%% Current Equations: Section 2
% calculate current between theta2 and theta3
for i = find(theta > theta2,1):find(theta > theta3,1)-1)
    II2(i) = (V1-V2)/(w*L12)*(theta(i)-theta2) + II2A;
    II3(i) = (V1+V3)/(w*L13)*(theta(i)-theta2) + II3A;
    II3(i) = (V2+V3)/(w*L23)*(theta(i)-theta2) + II3A;
end

%%% Current Equations: Section 3
% calculate current between theta3 and pi (half of cycle)
for i = find(theta > theta3,1):find(theta == pi,1)
    II2(i) = (V1-V2)/(w*L12)*(theta(i)-theta3) + II2B;
    II3(i) = (V1-V3)/(w*L13)*(theta(i)-theta3) + II3B;
    II3(i) = (V2-V3)/(w*L23)*(theta(i)-theta3) + II3B;
end

%%% Current Equations: Section 4
% calculate current between pi and pi + theta2
for i = find(theta == pi,1):find(theta >=(pi + theta2),1)-1)
    II2(i) = (-V1-V2)/(w*L12)*(theta(i)-pi) + II2pi;
    II3(i) = (-V1-V3)/(w*L13)*(theta(i)-pi) + II3pi;
    II3(i) = (V2-V3)/(w*L23)*(theta(i)-pi) + II3pi;
end

%%% Current Equations: Section 5
% calculate current between pi + theta2 and pi + theta3
for i = find(theta >=(pi + theta2),1):find(theta >=(pi + theta3),1)-1)
    II2(i) = (V2-V1)/(w*L12)*(theta(i)-pi-theta2) + II2pA;
\[ I_{13}(i) = \frac{(-V_1 - V_3)}{(w \cdot L_{13})} (\theta(i) - \pi - \theta_2) + I_{13pA}; \]
\[ I_{23}(i) = \frac{(-V_2 - V_3)}{(w \cdot L_{23})} (\theta(i) - \pi - \theta_2) + I_{23pA}; \]
\end

%% Current Equations: Section 6
% calculate current between \( \pi + \theta_3 \) and \( 2\pi \) (end of cycle)
for \( i = \text{find}(\theta > = (\pi + \theta_3), 1); \text{find}(\theta == 2\pi, 1) \)
\[ I_{12}(i) = \frac{(V_2 - V_1)}{(w \cdot L_{12})} (\theta(i) - \pi - \theta_3) + I_{12pB}; \]
\[ I_{13}(i) = \frac{(V_3 - V_1)}{(w \cdot L_{13})} (\theta(i) - \pi - \theta_3) + I_{13pB}; \]
\[ I_{23}(i) = \frac{(V_3 - V_2)}{(w \cdot L_{23})} (\theta(i) - \pi - \theta_3) + I_{23pB}; \]
end

%% Delta Currents Plot
% plot figure of currents corresponding to delta model
figure(1);
plot(theta, I_{12}, 'LineWidth', 2);
hold on;
plot(theta, I_{13}, 'r–', 'LineWidth', 2);
plot(theta, I_{23}, 'g:', 'LineWidth', 2);
grid on;
title('One Cycle of Delta Transformer Currents');
xlabel('Angle in cycle (rad)');
ylabel('Current Magnitude (A)');
legend('I_{12}', 'I_{13}', 'I_{23}');
set(gca, 'XTick', 0:pi/2:2*pi);
set(gca, 'XTickLabel', {'0', 'pi/2', 'pi', '3pi/2', '2pi'});

%% Real Currents Plot
% plot currents corresponding to wye model and seen at input/outputs
\[ I_1 = I_{12} + I_{13}; \]
\[ I_2 = I_{12} - I_{23}; \]
\[ I_3 = I_{13} + I_{23}; \]
\[ I_{1rms} = (\text{sum}(I_1.^2)/\text{numel}(I1)).^{0.5} \]
\[ I_{2rms} = (\text{sum}(I_2.^2)/\text{numel}(I2)).^{0.5} \]
\[ I_{3rms} = (\text{sum}(I_3.^2)/\text{numel}(I3)).^{0.5} \]
figure(2);
plot(theta, I_1, 'LineWidth', 2);
hold on;
plot(theta, I_2, 'r–', 'LineWidth', 2);
plot(theta, I_3, 'g:', 'LineWidth', 2);
grid on;
title('One Cycle of Actual Transformer Currents');
xlabel('Angle in cycle (rad)');
ylabel('Current Magnitude (A)');
legend('I1','I2','I3');
set(gca,'XTick',0:pi/2:2*pi);
set(gca,'XTickLabel',
{0,'pi/2','pi','3pi/2','2pi'});