ECE Senior Design Final Report
For

Scalable Regulated Three Phase Power Rectifier

May 10, 2004 Rev. 1.0

Sponsors:  Dr. Herb Hess (University of Idaho)
Dr. Richard Wall (University of Idaho)

Instructor:  Dr. Jim Frenzel (University of Idaho)

Tao Nguyen                  Tyler Budzianowski
nguy1484@uidaho.edu              budz6104@uidaho.edu
Table of Contents

Abstract 1

Project Description 2
  Problem Statement 2
  Solution Method 2

Status 2
  What is Designed and Working 2
  What is Designed but Not Working 2
  What is Designed but Not Tested 2

Method of Solution 3
  Technical description (Reference block and schematic diagrams) 3
  Theoretical basis and fundamental relationships 3

Validation Procedure 4
  Test Plan 4

Results 4
  Operating Procedures 4, 5
  Validation Results 5
  Cost Analysis 5

Appendix A – PIC16C74B Data 6, 7, 8
Appendix B – MATLAB Diagrams 9, 10, 11
Appendix C – MATLAB Results 12
Appendix D – PLL Simulation Results 13, 14
Appendix E – Components List 15
Abstract

A three-phase SCR power rectifier converts an AC input line voltage to a rectified DC output using silicon-controlled rectifying (SCR) diodes. Six individual diodes are used to control the magnitude of the DC output voltage by means of firing (turning on and off) at a specified delay angle and locked period. This firing scheme is controlled by a microcontroller that determines the necessary SCR gate firing times based detection of the input voltage zero crossings (signal period). A comparator circuit attached to the AC line produces a digital signal to the processor (based on signal zero crossings) that is captured and used to predict the next zero crossing capture time and fire the SCR’s based on this prediction. For accuracy and precision, a phase locked loop algorithm is implemented in software that works to improve the accuracy of the next predicted zero crossing and locks to produce the desired output voltage. An implementation of this rectification system was designed and outlined by Richard Wall and Herb Hess and utilizes components and techniques that are currently obsolete and need to be upgraded.
Project Description

ii. Problem Statement
The problem at hand is to eliminate as many inaccuracies present in the existing power converter system and upgrade the components and software to conform to more modern technology and methods. The system software shall also be more easily transferable to others seeking to produce similar results.

i. Solution Method
After a thorough review of the existing system design, a MATLAB system software model was first developed for the entire system. The 80C196KD microcontroller and its assembler code in the existing system has been replaced by a modern Microchip PIC series microcontroller utilizing ‘C’ programming code and input/output control ports. The upgrading of the software components will produce a more accurate zero crossing time, incorporating new zero-crossing detection technology. Hardware components that will improve accuracy over the original design have been chosen as well. This includes the dynamic hystereses comparator system that produces less signal noise and therefore greater accuracy than the originally implemented optoisolator configuration.

Status

i. What is Designed and Working
The coding plan and theoretical software development has been completed for the PIC16C74B microcontroller that will accept a single digital pulse input and produce 6 digital firing pulses that are 60 degrees out of phase with each other. This software scheme is not completely coded yet, as it is not fully functional at this time. However, success has also been made in downloading example programs (written in ‘C’ code) to the PICDEM2 Plus development board and PIC processor. One particular program was designed to fire blinking LED’s and was functioning properly. For the MATLAB model, the system does produce an output signal based on an input period but does not yet accurately predict the next calculated zero crossing time.

ii. What is Designed but Not Working
The phase locked loop implementation in MATLAB has been created and continually modified for accuracy, but does not utilize all the necessary components of the PLL and needs further development to achieve a more accurate prediction and signal period lock. The algorithm for the microcontroller that accepts a single input and produces six firing outputs currently does not function as intended but is under improvement. This is mainly due to some issues with controlling I/O registers and is being addressed immediately.

ii. What is Designed but Not Tested
The hardware components have been acquired and inspected. As of yet they have not been prototyped or mounted to a breadboard, per the request of the customer. This is an immediate goal for the project in the future. These components will first be individually tested in the lab prior to mounting/implementing to ensure that they do in fact function properly.
Method of Solution

i. Technical Description

The existing system has been reviewed and researched as to how the overall system functions, ease of usability, as well as what necessary components are included and how each can be upgraded for efficiency and accuracy to meet the listed specifications.

a. MATLAB System Model

A model has been developed to effectively simulate and analyze the phase locked loop system. A main emphasis of this model is to ensure accurate phase lock, leading to an accurate predicted zero crossing time (signal period) and eventually an accurate output DC voltage in the system. The implemented model consists of six main block components including a clock, data file, check threshold, phase detector, low pass filter, and voltage controlled oscillator. Please see figures in appendix B for the subsystems and more descriptions of each component.

b. PIC 16C74B Microcontroller Design

The input voltage signal period, determined by the comparator configuration as outlined by Richard Wall, is detected and processed by means of a digital phase locked loop to locked-loop control system to ensure that the next predicted zero-crossing time is as accurate as possible. Once lock in the loop is achieved, the processor output fires the six SCR’s based on the future zero crossings predicted by the software algorithm and the desired firing angle delay (requested output DC voltage).

c. Hardware Components

All of the components necessary for the actual rectification circuit have been purchased and are ready to be tested and then assembled on a protoboard. A commercially available gate firing circuit (Enerpro FCO-AUX60) will be used to amplify the digital output pulses of the microcontroller to a usable gate triggering level as specified by the SCR requirements.

ii. Theoretical Basis

Because the phase-locked loop system is a digital implementation, its design is software based. This will allow for modifications to be made as needed to improve the overall system accuracy by making adjustments within this software, rather than changing hardware components. The PIC microcontroller has the capabilities necessary to accurately control the three-phase rectification system and execute the phase locked loop design. Because this phase locked loop system is rather complicated and hard to understand thoroughly, a simulation model has been implemented in MATLAB that can be adjusted and analyzed based on the desired results. This essentially models the software behavior within the processor and the phase lock mechanism. On behalf of the overall hardware, the components selected are adequate to meet our needs, and the use of a commercial gate firing circuit allows more attention to be focused on other design aspects.
Validation Procedure

i. Test Plan

a. PIC Microcontroller

To ensure that the microcontroller is operating as was designed and intended, a function generator that generates a digital periodic pulse will be attached to the input of the processor. Another generated pulse will be connected to the microcontroller as well, to simulate the second zero crossing. These two input pulses essentially simulate the output of the comparator circuit that simulates the zero crossing of the input signal. On the output of the microcontroller, an oscilloscope will be interfaced that can display the six output firing pulses. From here the output pulses and software algorithms can be adjusted such that they fire as necessary for the rectification process.

b. Rectifying Circuit Hardware

The rectifying circuit will be constructed on a prototyping circuit breadboard to ensure that the circuit and all of the components are functioning as designed. This will need to be connected to a hi-voltage source for testing and analysis, such as the three phase electric grid in the EE Power Lab.

c. MATLAB System Model

There are three parts to test in order to verify whether or not it is a complete working PLL model. The first test is to simulate the PLL model with without calling the data file from the m-file to check if the system can produce a signal with the same frequency of the reference signal. The second test is to run the data file alone and check the multiples frequencies signal that the program in data file supposes to produce (see figure 7 in appendix B). The final model test is including the data file and the threshold block in the PLL model (see figure 1 in appendix B). Now, the designed PLL should perform as a conventional PLL where output frequency should be equal to a random input reference frequency.

Results

i. Operating Procedures

a. For demonstrational purposes, the following aspects of the design will be featured:

PIC16C73B Microcontroller

The resources associated with the microcontroller include the processor chip itself, the PICDEM2 Plus prototyping development board, the Microchip and IAR Systems development software. The system has successfully accepted and executed example software, and the development system has been used successfully. The intention for the demo was to design the system to accept a single digital input pulse and produce six separate firing pulses 60 degrees out of phase with each other. This implementation has not yet been completely successful. The design will incorporate a user interface (keypad) and LCD display to allow the user to specify the firing delay angle/output voltage in a user-friendly manner.
PLL Matlab Model

Data file program shows that it generates different frequencies. PLL matlab model (without including datafile block and threshold block) demonstrates the frequency locked when the phase errors between the reference signal and the feedback signal are zero. Lastly, PLL matlab model (with including the datafile block and threshold block) demonstrates the lock frequency ability between the output signal and the random input signal.

ii. Validation Results

As previously stated, an oscilloscope and function generator setup will be used to control the input signal period and will allow for monitoring the output signal pulses. For the PLL matlab model, Data file is compiled, run and generated different frequencies (see appendix C for the plot). PLL without datafile block and threshold block is run and locked the phase when the error is zero. This can be shown in the appendix D. PLL with the datafile block and threshold block is run but not working as expected, and this can be shown in appendix E.

iii. Cost Analysis

The bulk of the necessary components are not extraordinary expensive and were acquired for about $350, including a gate firing circuit. The PIC16C74B microcontroller chip itself was rather inexpensive ($13), however the development board and software kit was higher in cost ($105). The most expensive item necessary for the design is the Enerpro FCO-AUX60 gate firing circuit board. The purchase of this component was delayed until it was determined that it would be the best solution for the design due to its reliability and ease of use. The cost for the printed circuit board layout is a rough estimate due to the fact that it has not yet been acquired.

Table 1. Budget details for components in the Project

<table>
<thead>
<tr>
<th>Components</th>
<th>Price/unit</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Firing Circuit Board</td>
<td>$292.45</td>
<td>$292.45</td>
</tr>
<tr>
<td>1 Demo Board for PIC</td>
<td>$104.38</td>
<td>$104.38</td>
</tr>
<tr>
<td>1 PIC16C74 processor</td>
<td>$12.73</td>
<td>$12.73</td>
</tr>
<tr>
<td>1 Comparator circuit</td>
<td>$1.10</td>
<td>$1.10</td>
</tr>
<tr>
<td>1 12Vdc Power Supply</td>
<td>$14.95</td>
<td>$14.95</td>
</tr>
<tr>
<td>1 LCD</td>
<td>$14.91</td>
<td>$14.91</td>
</tr>
<tr>
<td>1 Key Pad</td>
<td>$14.95</td>
<td>$14.95</td>
</tr>
<tr>
<td>1 7805 Voltage Regulator</td>
<td>$0.99</td>
<td>$0.99</td>
</tr>
<tr>
<td>6 x SCRs</td>
<td>$1.80</td>
<td>$10.80</td>
</tr>
<tr>
<td>Total Cost</td>
<td></td>
<td>$467.26</td>
</tr>
</tbody>
</table>
Appendix A

Microchip PIC16C74B Specifications:

Resources:
- 4 K x 14 words of Program Memory,
- 192 x 8 bytes of Data Memory (RAM)
- High performance RISC CPU
- 22 I/O ports
- 8-bit timer/counter with 8-bit prescaler
- 16-bit timer/counter with prescaler
- DC - 20 MHz clock input
- PWM max. resolution is 10-bit

Electrical Ratings:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambient Operating Temperature</td>
<td>-55 deg C to 125 deg</td>
</tr>
<tr>
<td>Pin Voltage Range</td>
<td>-0.3 V to +7.5 V</td>
</tr>
<tr>
<td>Maximum Supply Output Current</td>
<td>300 mA</td>
</tr>
<tr>
<td>Maximum Supply Input Current</td>
<td>250 mA</td>
</tr>
<tr>
<td>Max Current Sunk by All Ports</td>
<td>200 mA</td>
</tr>
<tr>
<td>Power Consumption</td>
<td>&lt; 5 mA @ 5V, 4 MHz</td>
</tr>
<tr>
<td></td>
<td>23 mcA typical @ 3V, 32 kHz</td>
</tr>
<tr>
<td></td>
<td>&lt; 1.2 mcA typical standby current</td>
</tr>
</tbody>
</table>
PIC SOFTWARE IMPLEMENTATION (coding scheme)  
[pre design demo]

Objectives:
- To receive 1 digital pulse input from I/O pins
- Output six individual firing pulses to I/O pins, each delayed based on the desired firing angle in an open loop configuration

Implementation:
System Resolution:
This value is based on the angle that will be entered by the user (in degrees)

- With a desired frequency of 60 Hz, our period is then 16.67 ms. Therefore at 1 degree of resolution (of 360) our instruction time length is 16.67 ms/360 deg = 46297 ns
- Because our processor takes 50 ns per instruction cycle, our corresponding timer value is 46297 ns/50 ns = 926 units
- This value can be written to TIMER0 in order to allow the timer to interrupt upon overflow.

Initialization
- Should set ports to zero to initialize (ensure no instabilities occur)
- Setup to poll PortA until an input signal is received (zero-cross).
  - Once a signal is received, interrupt to period calculation algorithm

Input
1. We will need to capture the timer value when the first pulse is received (1st zero crossing) and capture the timer value when the second pulse is received (2nd zero crossing). These will use Timer0 as the reference. From these two values, the period of the input signal can be computed
   \[ \text{PERIOD} = (\text{ZeroCross2} - \text{ZeroCross1}) \]
2. From the computed period we can force the software to interrupt every 46297 ns by writing the value 926 to TIMER0. This value ensures the processor will overflow at the desired time length (based on our degree of resolution). This overflow will cause an interrupt
3. This interrupt causes the processor to output one of the 6 outputs to the output PortB

Output
1. When Timer0 interrupts an output is passed to the PortB output port. The period of the output pulse should be 16.67 ms (for 60 Hz frequency)
2. The remaining consecutive pulses can be fired at the required delay. For example, I will base my firing pulses as being 60 degrees apart for each of the 6 output pulses.
   - For say, 1 degree resolution:
     - each consecutive pulse is based on the first output pulse (generated on Timer0 overflow interrupt) PLUS 60 deg
       \[ (60 * 926) = 55560 \]
     - \( \text{PULSE1} = 926 \)
     - \( \text{PULSE2} = \text{PULSE1} + 55560 \)
     - \( \text{PULSE3} = \text{PULSE2} + 55560 \)
     - \( \text{PULSE4} = \text{PULSE3} + 55560 \)
     - \( \text{PULSE5} = \text{PULSE4} + 55560 \)
     - \( \text{PULSE6} = \text{PULSE5} + 55560 \)
3. After the 6th output pulse is fired, the processor will break out of the interrupt routine and restart the process
PIC16C74B Pin Assignments and Resource Allocations (PreDemo)

**PORTA**  (bi-directional I/O port)
This is used by the PICDEM2 Plus board for the control of the LCD module
- The pins used for the LCD control lines are (RA3, RA2, RA1)

**PORTB**  (bi-directional I/O port)
This will be used for the digital input to the processor from the zero crossing detecting comparator circuit.
- The current design utilizes only one input port for demonstration purposes (RB1) but will utilize another port for the second zero-crossing digital signal (RB2)

**PORTC**  (bi-directional I/O port)
This will be used as the output of the processor to the gate firing circuitry.
- The output ports here will be the 6 output SCR gate firing pulses (RC1, RC2, RC3, RC4, RC5, RC6)

**PORTD**  (bi-directional I/O port)
This is used by the PICDEM2 Plus board for the data to the LCD module.
- The pins used are for the LCD data lines are (RD3, RD2, RD1, RD0)

**Demonstration Layout**

Digital Input $\rightarrow$ PIC16C74B $\rightarrow$ 6 Sequentially Firing Pulse Signal $\rightarrow$ Output Pulse Signals
Appendix B

Figure 1. Overall Simulink Matlab of Phase Locked Loop

**Threshold block** - helps the system to lock the phase when the small change in low pass filter output is equal to the threshold constant.

Figure 6. Threshold subsystem block diagram

**Phase Detector** - a sequential logic phase detector. The phase detector compares the phase at each input and generate the error signal, $V_e(t)$, proportional to the phase different between the two inputs. $K_d$ is the gain of the phase detector (V/rad).

Figure 3. Phase Detector subsystem block Diagram
The purpose of low pass filter in the PLL model is to integrate the pulsed output from the phase detector to produce a smoothed “DC” VCO control designed voltage.

\[
\frac{1}{0.1667s+1}
\]

Figure 4. Low pass filter transfer function

VCO oscillates at an angular frequency, \( W_{out} \). Its frequency is set to a nominal \( W_0 \) when the control voltage is zero. Frequency is assumed to be linearly proportional to the control voltage with a gain coefficient \( K_{VCO} \). VCO gain is a measure of the frequency swing of the VCO per unit change in the VCO control voltage.

Figure 5. Subsystem block diagram for Voltage Control Oscillator

The modulate integrator integrates the input constant to become a linear plot with a magnitude of one and with a new frequency. This frequency increases with a gain of \( K=1+\epsilon \), where \( \epsilon \) is equal to 0.22*2^16.

Figure 6. Subsystem block diagram of the Modulo Integrator
Appendix B continues

Figure 7 Subsystem of Convert to Square block

```matlab
data = 0;                      %initialize data, N, a, x, fr1
N=2^16;
a=1;
x=0;
fr1=0;

for i=1:2^N
    newfr=0;
    fr=58+4/(2^N)*i;                                                        %frequency of the current signal
    if i==1
        clk = fr;                                                                  %set clock equals first frequency
        fr1=fr1+fr;
        for j=1:5                                                                 %generate number cycles when fr equals 59.5Hz
            t=0:1/(fr1*fr1):1/fr1-1/(fr1*fr1)
            data(1+fr*(j-1):fr+fr*(j-1))=square(fr*2*pi*t);
            x=x+fr;
        end
    end
end

if i==a+1
    fnew=fr;                                                                  %set new frequency equals fnew
    z=(1/fnew)/(1/(fr1*fr1));                                        %define new number of samples in one cycle
    t=0:1/(z*frnew):1/frnew-1/(z*frnew);
    for j=1:5                                                                  %generate number of cycles for each signal
        data(1+x:x+z)=square(fr*2*pi*t);                      %each signal
        x=x+z;
    end
    a=a+1
end

plot(data,'b')
```

Figure 8. Following is the program codes that stored in the data file block:
Appendix C

Figure 8. Output plot of the datafile block, frequency does changed respect to time
Appendix D
The following are the output scope of reference signal, phase detector, and VCO of PLL that is not including the data file block and the threshold block.
Followings are the output scope of reference signal, phase detector, and VCO of PLL that is including the datafile block and the threshold block.
Appendix E
Components List

- 4025L Silicon Controlled Rectifiers
  - Used to control the output DC voltage magnitude in based on a user defined firing angle.

- 7805 Voltage Regulator
  - Was originally planned to be used to step up the output voltage signal pulse from the processor to the firing board (up to 12 V)

- Jameco Electronics 4x4 Matrix Keypad
  - Used for user input to the processor to define firing angle or desired voltage output level (DC)

- 12 Vdc Unregulated Power Supply
  - Was originally designed to be used to step up the output voltage signal pulse from the processor to the firing board (up to 12 V)

- LM393 Comparator Chip
  - Used in place of the original optoisolator implementation in a dynamic hysteresis configuration to detect the input signal zero crossings while eliminating as much signal line noise as possible

- Azonic LCD Screen
  - Used to display the firing angle, and/or the output voltage

- Microchip PIC16C74B Microcontroller
  - Used to replace the original Intel 196 microcontroller with an updated and non obsolete processor

- Microchip PICDEM2 Plus Demonstration Board
  - Used to allow for prototyping and demonstration of the PIC16C74B microcontroller’s features and functionality

- Enerpro FC0-AUX60
  - Commercially available firing circuitry used to amplify the output digital pulse form the microcontroller to a gate-level voltage to fire the SCRs.