Hardware Simulation of Diesel Generator and Microgrid Stability

by

Michael M. Zieve

S.B., E.E. M.I.T., 2011

Submitted to the Department of Electrical Engineering

and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

May 2012

Author: ______________________________

Department of Electrical Engineering and Computer Science
May 21, 2012

Certified by: ______________________________

Professor James L. Kirtley Jr., Thesis Supervisor
May 21, 2012

Accepted by: ______________________________

Professor Dennis M. Freeman, Chairman, Masters of Engineering Thesis Committee

Hardware Simulation of Diesel Generator and Microgrid Stability
by
Michael M. Zieve

Submitted to the
Department of Electrical Engineering and Computer Science

May 21, 2012

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

Over the last few years, people have begun to depend less on large power plants with extensive distribution systems, and more on local distributed generation sources. A microgrid, a local collection of distributed generators, has the potential to offer a more flexible and customizable power generation system, while significantly improving its effect on the environment. In order to properly deploy and scale microgrids to meet diverse energy needs, there must be more study on their stability. This paper details the process and design of the modeling of a diesel generator. With the constructed diesel generator as a component of the microgrid project, the voltage and power stability of the modeled microgrid can be tested under various load conditions and faulted islanded conditions to help design the future of the electrical grid.

Thesis Supervisor: James L. Kirtley, Jr., Professor of Electrical Engineering

Acknowledgements

I would like to thank Professor James L. Kirtley Jr. for his advice and supervision over the course of this project. I would also like to thank my fellow students Jared Monnin, Jorge Martinez, Gavin Darcy, Zhyong Whang, Alejandro Ojeda and Webb Horn who, with their hard work, have enabled this project to grow from a mere idea one year ago, to a lab scale hardware model of a microgrid. Lastly, I would like to thank my parents, Sandie and Peter, and my sister Olivia for their constant support and encouragement throughout both my undergraduate and master's studies at MIT.

Table of Contents

Table of Figures

Background

Much of our electricity today comes from large generators that power the central utility. In the United States in 2010, 44.5% of power production was from coal-fired power plants, 24.1% from natural gas power plants, 19.4% from nuclear power plants and 6.1% from hydroelectric power plants (US Energy Information Administration). These generation sources are almost exclusively used in large generators that have become pervasive and a dominant part of the electrical grid because of their initial economies of scale. These large generating facilities are commonly located near the resources they require; for example, hydroelectric power plants are in areas with large water flow, and coal power plants are located near coal mines. These generating facilities are also normally located far from dense population centers in order to minimize pollution exposure. These limiting factors place large power plants far from the actual demand, thus requiring a substantial grid throughput to larger metropolitan areas and a complicated distribution grid to reach remote power customers. While the generators are seemingly functional and effective, the grid distribution system is vulnerable to many potential failures.

Over the last few years, technological developments have allowed for an increase in the use of various forms of distributed generation, which has transformed the utility distribution grid. The dependence on a few larger centralized power generation plants no longer meets reliability standards, as one fault can have a large impact on the power distribution to entire regions of the country. This consequence of centralized power distribution has changed the dependence on the traditional distribution grid and has led to reinforcement at a more local level by microgrids.

Microgrid

A microgrid is a local power source that is comprised of a collection of various forms of distributed generators. Many of these newer distributed generators are more environmentally friendly than traditional power generation sources because of their use of solar and wind energy, which has led to their popularity and development. A typical microgrid can also contain diesel generators, fuel cells and smaller turbines. A standard microgrid can rely on both distributed generators that produce alternating current as well as direct current. Typically, the alternating current and alternating voltage from a diesel generator, gas turbine or wind turbine is scaled using a transformer to meet all utility grid magnitude requirements and enable it to connect to the similarly alternating voltage of the grid. Connection to the central utility is more difficult for direct current and direct voltage generators, such as photovoltaic cells or other fuel cells. The direct current sources must be passed through a standard one or three phase inverter to properly interface with the alternating voltage of the central utility and standard wave form.

Microgrids are predominantly built to run in parallel with the central utility grid and designed to supplement the grid to help better provide for local electricity needs. However, microgrids are also designed to disconnect and run independently from the central grid while maintaining standard voltage and frequency levels. When a microgrid is in this independent state, it is said that it is "islanded". Microgrids are considered the way of the future because they are so versatile. Generally, the most valued attribute is a microgrid's ability to become islanded so power can be provided on a local level, even if the central utility were to fail or be disconnected. A microgrid allows for less dependence on the central utility where a fault can lead to enormous blackouts. One such blackout was in 2005 on two islands, Java and Bali. On the islands, "a failure in 500 kilovolt (kv) transmission line between Cilegon and Saguling in West Java cut electricity supplies" which in turn, "caused several power plants units to disengage from the network... with a combined capacity of 4,000 megawatts (MW) shut down" (Jakartapost). The blackout affected over 100 million people and is recognized as the largest blackout in history. While central utilities as a whole are extremely reliable, especially in the United States, in 2003 "a high-voltage power line in northern Ohio brushed against some overgrown trees and shut down... 50 million people lost power for up to two days... [and] the event contributed to at least 11 deaths and cost an estimated $6 billion" (Scientificamerican). This dependence on a complex central infrastructure means that a well-placed fault can affect all loads downstream. Microgrids can satisfy the need for more locally independent power generation, which will increase the service reliability and power quality on a local level.

Value of the Microgrid Model

It is important to understand that the development of the microgrid model is beneficial not only to the consumer but also to the existing utility. Microgrids supplementing the utility will minimize demand during the peak load times, which cause the most stress and costs on the utility infrastructure. The ability to purchase additional green energy from microgrids will also allow for the utility to meet its Renewable Portfolio Standards requirements, as well as meeting carbon reduction goals without assuming risk or making investments. Additionally, microgrids allow for greater reliability on the electrical system during adverse conditions, and can help minimize effects and costs from central outages.

Besides helping the utility, the real benefit of the microgrid model is that, because of the localization of the power sources, communities are able to specialize and construct a system that meets their specific needs, resources and infrastructure. The customization of a microgrid can allow for specific environmental requirements during design and construction so as to minimize specific pollutants or environmental damages. The designing of a microgrid can also take advantage of a specific locally abundant resource; for example, an extremely windy valley can construct a microgrid that is predominantly powered by a wind farm. The value in using a microgrid is that it can incorporate non constant generation sources, such as solar farms or wind turbines that fluctuate their power generation. When the wind slows, or it is either cloudy or night time, the microgrid can purchase power from the utility. When the wind is in full force or the sun is shining, the microgrid can provide all the power to its local community and even sell an excess to the utility. This customization can also apply to the specific strategy implemented to control the microgrid's production levels and loads. One can design a microgrid to supply prioritized power for various loads that have been deemed most important in the case of an islanded situation from a central utility fault. An example of this type of specific control is a university prioritizing the power to its animal testing facilities and prominent labs over student housing or administrative buildings. A microgrid can provide for specialized power needs and resources, more environmentally friendly power, and the ability to be less dependent on the central utility; thus, a microgrid is an important fixture in our current and future demand for power.

Motivation

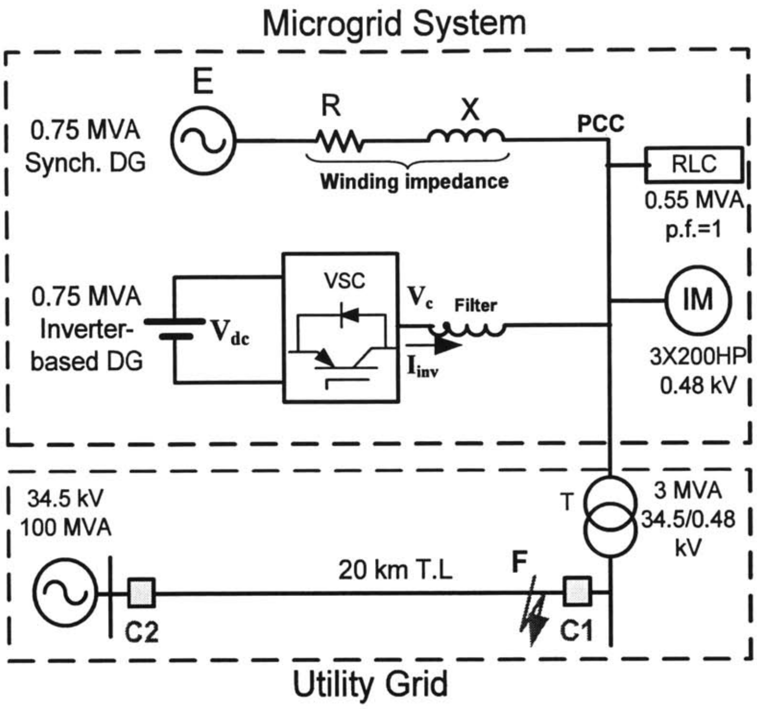


Figure 1 Microgrid model. Source: Impact of DG Interface

The focus of my research group, led by Professor James L. Kirtley, Jr., is to create a simulation of a microgrid that can allow for actual hardware tests to be performed. The most interesting area of study surrounding a microgrid is its behavior when disconnected from the central power grid due to a transient fault. In this islanded condition, the microgrid is suddenly forced to support a significantly increased load from the local system, as it is no longer supplemented from the central utility. Our microgrid hardware simulation will be on the kilowatt scale and will encompass a photovoltaic cell emulation, a wind turbine emulation and a diesel generator emulation. The microgrid will have various resistive, capacitive, inductive and phase controller loads with the most prominent being a large induction motor that will simulate realistic loads a typical microgrid would have to support. The overview of our microgrid model is seen in Figure 1. Through software simulations in Simulink in Matlab, different islanding conditions have been tested by varying fault durations and the types of loads on the system. Software simulations suggest that the system recovery from fault induced islanding conditions is heavily dependent on the specific loading of the system. With resistive, capacitive, inductive and phase controller loads on the system, the microgrid is able to maintain stable power generation through permissible voltage and frequency power supply levels. However, the software simulations also suggest that the stability of the system can be substantially affected with large direct connected induction motor loads, which make up a large portion of the realistic loads from commercial facilities (Alaboudy, Zeineldin and Kirtley 2011). Our area of study is to determine the effects of these various loads on voltage stability with variable fault durations in actual hardware. From our hardware model, we hope to determine the transient stability of the microgrid. More specifically, I will be focusing my thesis on the designing of the diesel generator emulator and how its modeled characteristics affect the overall microgrid hardware simulation.

Methodology

Beyond helping construct the overall project, which can be seen in more detail in the Reference Manual for Microgrid Hardware Simulation System, I am responsible for the designing and building of the hardware simulation of the diesel engine. Due to the obvious difficulties of actually incorporating a diesel engine into the simulation, my job is to recreate the behavior and response of a diesel generator using standard laboratory hardware. I will be using a 1.5 HP DC motor to behave as the diesel engine, which will have its shaft mechanically coupled to that of a 1 HP wound-rotor induction motor. This induction motor will be used to drive the system in the form of a synchronous generator. The basis of the simulation is to control the torque of the DC motor in accordance with the behavior of a diesel generator.

Diesel Generator Governor Model in CT

The basis for which the DC motor is used as a diesel generator is with its control strategy through the diesel generator governor model, which is seen in Figure 2.

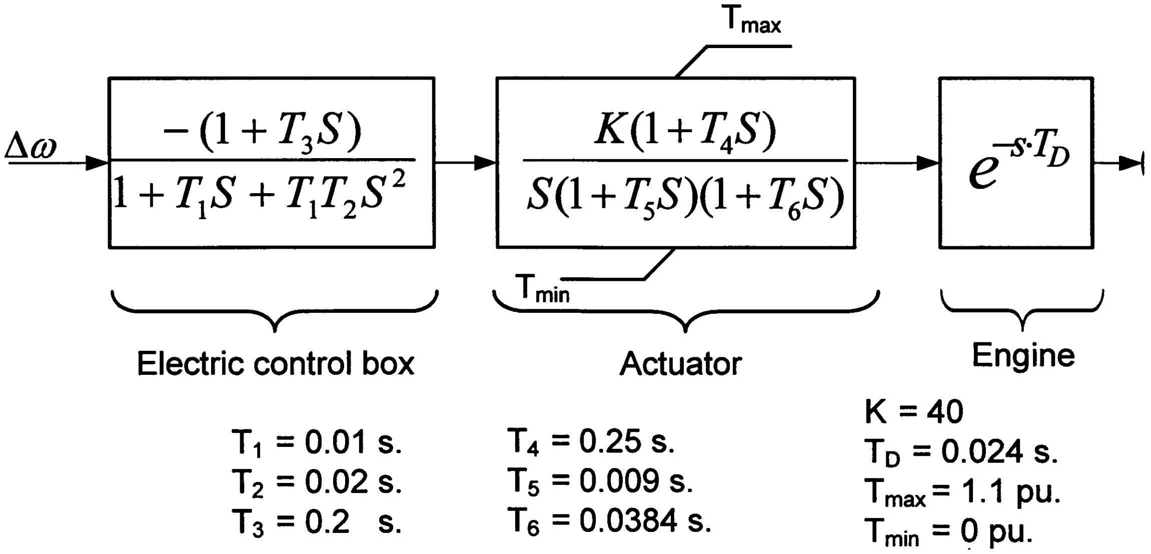


Figure 2 Diesel generator governor model. Source: Impact of DG Interface

When combining the effects of the Electrical control box and the Actuator, the System Function that describes the diesel generator governor model can be written as follows:

$$\frac{Y(s)}{X(s)} = \frac{-(1+T_3 s)K(1+T_4 s)}{(1+T_1 s+T_1 T_2 s^2)s(1+T_5 s)(1+T_6 s)}$$

In the system function, the input X represents the input Δω which represents the difference between the desired speed ω^* and the actual speed of the motor. The output Y represents the allowable torque output of the transfer function. While the Engine delay has not yet been incorporated, it will be enacted at the end in the actual implementation. The system function can now be expanded out as seen below:

$$\frac{-K - KT_3 s - KT_4 s - K(T_3 T_4)s^2}{s(1 + T_6 s + T_5 s + T_5 T_6 s^2 + T_1 s + T_1 T_5 s + T_1 T_6 s^2 + T_1 T_5 s^2 + T_1 T_5 T_6 s^3 + T_1 T_2 s^2 + T_1 T_2 T_6 s^3 + T_1 T_2 T_5 s^3 + T_1 T_2 T_5 T_6 s^4)}$$

After the System Function has been expanded, it is now sorted out by powers of "s":

$$\frac{-K - K(T_3 + T_4)s - K(T_3 T_4)s^2}{s + (T_6 + T_5 + T_1)s^2 + (T_5 T_6 + T_1 T_5 + T_1 T_6 + T_1 T_2)s^3 + (T_1 T_5 T_6 + T_1 T_2 T_6 + T_1 T_2 T_5)s^4 + T_1 T_2 T_5 T_6 s^5}$$

The constants that scale the transfer function and the different powers of "s" are as follows:

In the numerator:

$$b_0 = -K$$

$$b_1 = -K(T_3 + T_4)$$

$$b_2 = -K(T_3 T_4)$$

In the denominator:

$$a_0 = 0$$

$$a_1 = 1$$

$$a_2 = T_6 + T_5 + T_1$$

$$a_3 = T_5T_6 + T_1T_5 + T_1T_6 + T_1T_2$$

$$a_4 = T_1T_5T_6 + T_1T_2T_6 + T_1T_2T_5$$

$$a_s = T_1T_2T_5T_6$$

Each subscript of the constant represents the power of "s" that it scales. The transfer function can now be represented as:

$$a_5Y(s)s^5 + a_4Y(s)s^4 + a_3Y(s)s^3 + a_2Y(s)s^2 + a_1\text{Y(s)s} = b_0X(s) + b_1X(s)s + b_2X(s)s^2$$

This system can be better understood by performing a pole-zero plot in Matlab.

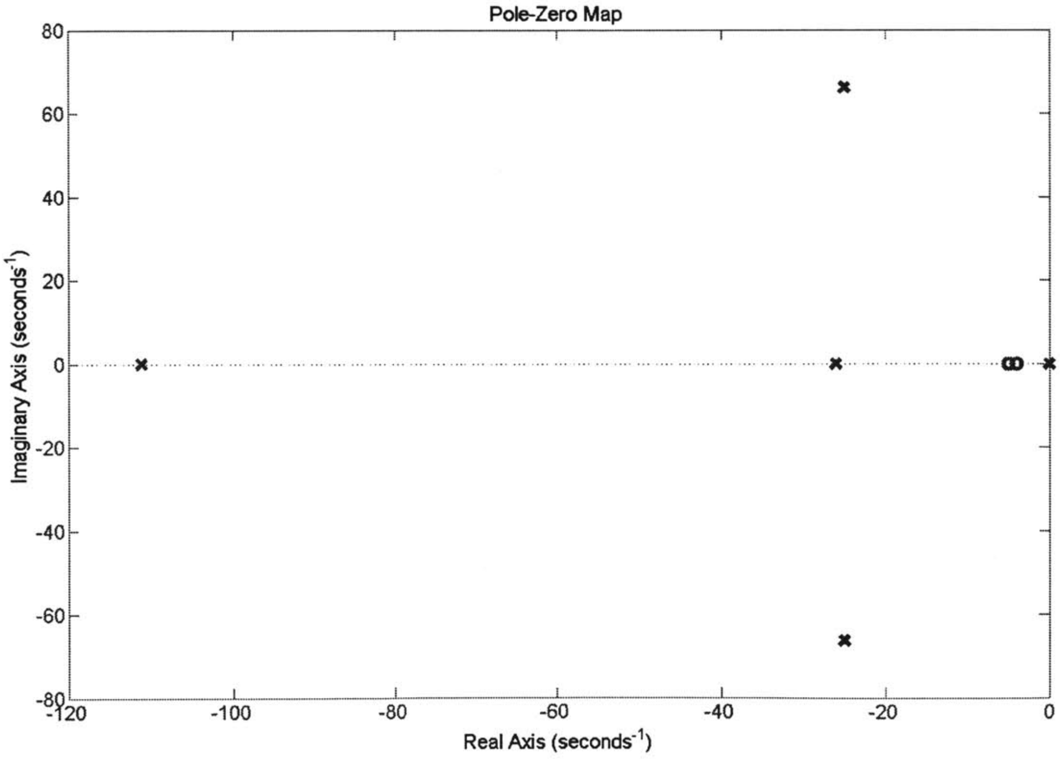


Figure 3 Pole-Zero Plot of Diesel Generator Governor Model

The poles are located at (0, 0), (-26, 0), (-26, 66.1), (-26, -66.1) and (-111, 0) and the zeros at (-4, 0) and (-5, 0). The poles that are expected to dominate are the ones at (0, 0) and (-26, 0) while the dominant zero is at (-4, 0). The plot in Figure 4 demonstrates the magnitude and phase response of the system as it varies with a changing frequency.

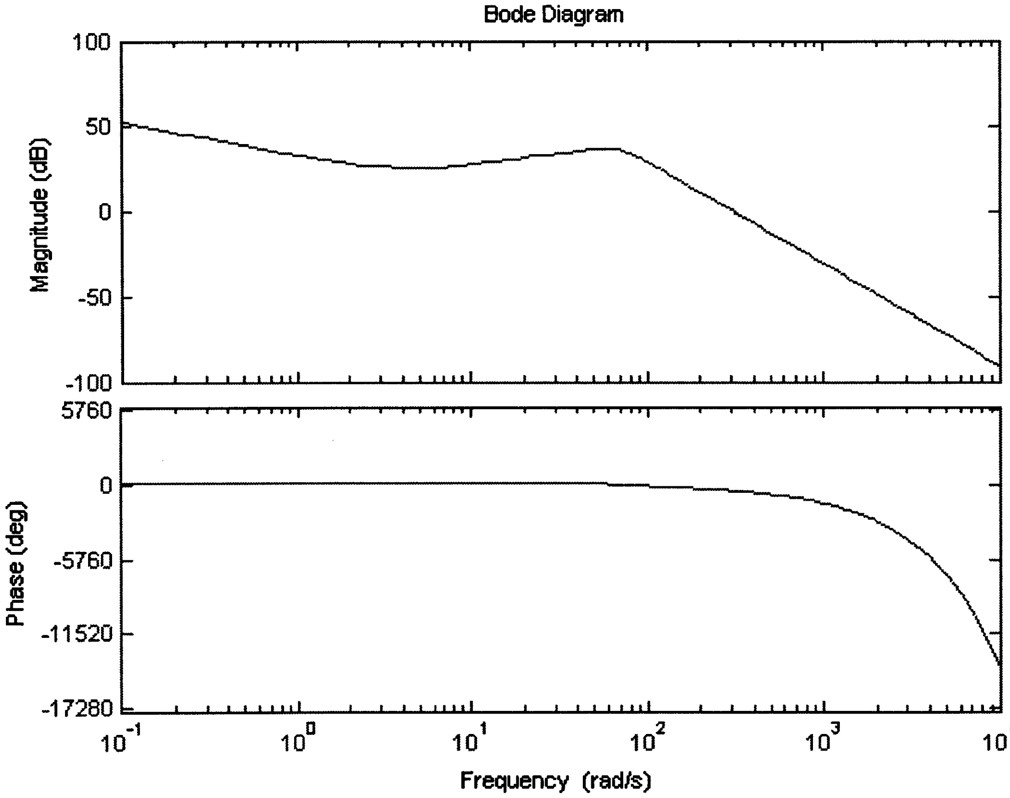


Figure 4 Bode Diagram of Diesel Generator Governor Model

This system function in the Laplace domain can be represented by a differential equation in the time domain. Each power of "s" represents a derivative $\frac{d}{dt}$ in the time domain. The differential equation of the diesel generator governor model can be written as:

$$a_5 y(t)^{(5)} + a_4 y(t)^{(4)} + a_3 y(\dddot{t}) + a_2 y(\ddot{t}) + a_1 y(\dot{t}) = b_0 x(t) + b_1 x(\dot{t}) + \ b_2 x(\ddot{t})$$

Diesel Generator Governor Model in DT

The continuous time diesel generator model, if implemented, would be an exact copy of how a real diesel generator would behave with the given characterizations. However, the dsPIC microcontroller can only operate in discrete time, so the continuous time model needs to be discretized to be used so it can be processed and implemented on the dsPIC microcontroller. In order to properly discretize the continuous time function while using available information, I used the backwards-Euler approximation. The backwards Euler theorem uses the definition of the derivative to take the change in the variable over the time.

$$z(\dot{n}T) \approx \frac{z[n] - z[n-1]}{T}$$

$$z(\ddot{n}T) \approx \frac{\frac{z[n] - z[n-1]}{T} - \frac{z[n-1] - z[n-2]}{T}}{T} = \frac{z[n] - 2z[n-1] + z[n-2]}{T^2}$$

$$z(\dddot{n}T) \approx \frac{\frac{z[n] - z[n-1]}{T} - 2\frac{z[n-1] - z[n-2]}{T} + \frac{z[n-2] - z[n-3]}{T}}{T^2}$$

$$= \frac{z[n] - 3z[n-1] + 3z[n-2] - z[n-3]}{T^3}$$

$$z(nT)^{(4)}$$

$$\approx \frac{\frac{z[n] - z[n-1]}{T} - 3\frac{z[n-1] - z[n-2]}{T} + 3\frac{z[n-2] - z[n-3]}{T} - \frac{z[n-3] - z[n-4]}{T}}{T^3}$$

$$= \frac{z[n] - 4z[n-1] + 6z[n-2] - 4z[n-3] + z[n-4]}{T^4}$$

$$z(nT)^{(5)} \approx \frac{\frac{z[n] - z[n-1]}{T} - 4\frac{z[n-1] - z[n-2]}{T} + 6\frac{z[n-2] - z[n-3]}{T}}{T^4}$$

$$\frac{-4\frac{z[n-3]-z[n-4]}{T}+\frac{z[n-4]-z[n-5]}{T}}{T^4}$$

$$=\frac{z[n]-5z[n-1]+10z[n-2]-10z[n-3]+5z[n-4]-z[n-5]}{T^5}$$

The backward-Euler approximation has allowed for the discrete implementation of the continuous time differential equation through the sampling of values at a sampling period of T. The differential equation from the diesel generator governor model can now be expressed discreetly when the backwards-Euler approximations are plugged in.

$$\frac{a_1y[n]}{T}-\frac{a_1y[n-1]}{T}+$$

$$\frac{a_2y[n]}{T^2}-2\frac{a_2y[n-1]}{T^2}+\frac{a_2y[n-2]}{T^2}+$$

$$\frac{a_3y[n]}{T^3}-3\frac{a_3y[n-1]}{T^3}+3\frac{a_3y[n-2]}{T^3}-\frac{a_3y[n-3]}{T^3}+$$

$$\frac{a_4y[n]}{T^4}-4\frac{a_4y[n-1]}{T^4}+6\frac{a_4y[n-2]}{T^4}-4\frac{a_4y[n-3]}{T^4}+\frac{a_4y[n-4]}{T^4}+$$

$$\frac{a_5y[n]}{T^5}-5\frac{a_5y[n-1]}{T^5}+10\frac{a_5y[n-2]}{T^5}-10\frac{a_5y[n-3]}{T^5}+5\frac{a_5y[n-4]}{T^5}-\frac{a_5y[n-4]}{T^5}$$

$$=$$

$$b_0x[n]+$$

$$\frac{b_1x[n]}{T}-\frac{b_1x[n-1]}{T}+$$

$$\frac{b_2x[n]}{T^2}-2\frac{b_2x[n-1]}{T^2}+\frac{b_2x[n-2]}{T^2}$$

Combining all scalars on each specific sample of time n, another set of constants can be derived to help simplify the discretized differential equation:

$$cy_0 = \frac{a_1}{T} + \frac{a_2}{T^2} + \frac{a_3}{T^3} + \frac{a_4}{T^4} + \frac{a_5}{T^5}$$

$$cy_1 = -\frac{a_1}{T} - 2\frac{a_2}{T^2} - 3\frac{a_3}{T^3} - 4\frac{a_4}{T^4} - 5\frac{a_5}{T^5}$$

$$cy_2 = \frac{a_2}{T^2} + 3\frac{a_3}{T^3} + 6\frac{a_4}{T^4} + 10\frac{a_5}{T^5}$$

$$cy_3 = -\frac{a_3}{T^3} - 4\frac{a_4}{T^4} - 10\frac{a_5}{T^5}$$

$$cy_4 = \frac{a_4}{T^4} + 5\frac{a_5}{T^5}$$

$$cy_5 = -\frac{a_5}{T^5}$$

$$cx_0 = b_0 + \frac{b_1}{T} + \frac{b_2}{T^2}$$

$$cx_1 = -\frac{b_1}{T} - 2\frac{b_2}{T^2}$$

$$cx_2 = \frac{b_2}{T^2}$$

In the above constants, each subscript represents the number of delays the samples it modifies has undergone. In the end, the pre engine delay discrete version of the diesel generator governor model can be described by the following equation.

$$cy_0y[n] + cy_1y[n-1] + cy_2y[n-2] + cy_3y[n-3] + cy_4y[n-4] + cy_5y[n-5] = cx_0x[n] + cx_1x[n-1] + cx_2x[n-2]$$

This is the equation that is run to implement the diesel generator governor model control strategy. However, as mentioned before, the engine part of the Diesel Generator governor model is represented as e^{-sT_D}which, in the laplacian domain, represents a delay of T_D which is specified as being .024 seconds. When implementing this delay, once the proper value of y[n] is determined from the equation,

$$y[n] = \frac{-cy_1y[n-1] - cy_2y[n-2] - cy_3y[n-3] - cy_4y[n-4] - cy_5y[n-5] + cx_0y[n] + cx_1y[n-1] + cx_2y[n-2]}{cy_0}$$

it is now implemented as the actual output for .024 seconds. This allows for the discrete model to incorporate the effects of the engine and therefore be complete.

Implementation and Design

The actual implementation of the methodology was performed using a dsPIC30F6010, which is the brain of a Microchip M1 Motor Control Development Board. The program to control the whole system is written in C and is called DieselGenerator.C. This full program is appended at the end of this thesis. It is run using MPLAB IDE and compiled into Assembly language for the microcontroller to process it. This section will be a walkthrough of the general sections of the code and the important aspects of each section. For an understanding of the hardware that was used to implement the model, see the Reference Manual for Microgrid Hardware Simulation System.

Program Setup Section

Initially in the program, it is important to define all of the additional files that are needed to run the program. These files allow for the program to refer to registers by their name as opposed to by some arbitrary hexadecimal code. After the additional files have been imported, the pertinent variables are defined. These are all values that will remain constant throughout the whole program. The first set of these constants is the same constants used in the Diesel Generator Governor Model for which this whole system is defined. There are also the parameters, such as gains and constraints for the PID controller, that will adjust the current input into the synchronous generator, which in turn will adjust the three phase voltage output into the microgrid bus. Next, there are the DC Motor constants which have been measured from the actual motor. The field winding resistance R_F was easily measured with a multimeter and $R_F = 276.6\Omega$. With the intention to maintain the field voltage V_F at a constant 175 volts, the motor constant G remains the only constant the needs to be calculated. To do this, first I needed to know I$_F$, which is a constant equal to $I_F = \frac{V_F}{R_F} = \frac{200\, volts}{276.6\Omega} = .723\ amperes$. The motor constant G, when multiplied by I$_F$, is the slope of the Armature Voltage-RPM plot in the DC Motor Test figure. This is true under the assumption that R$_A$ is negligibly small. With the actual value of R$_A$ =3.3Ω being determined by measuring with a multimeter, it is small enough to be ignored for this approximation. The average value of G from my DC motor measurements with V$_F$ = 200VDC and V$_A$= 50, 100, 150 and 200VDC is .779 N-m/A^2. Finally, the constant LAG_ELEMENT_SIZE needs to be set to implement of the delay of $e^{-sT_D}$with T$_D$= .024 seconds. The basis for which the delay is implemented using LAG_ELEMENT_SIZE will be explained in the Main Program Section. However, it is important to understand how to set the value of LAG_ELEMENT_SIZE. Because the program will constantly be updated and have sections added to it, it is important to have the delay remain at .024 seconds. Later, in the Main Program Section when the PWM output is implemented, the LED D9 on the MC1 Motor Control Board will blink after each new PWM duty cycle update. By using an oscilloscope on the LED, I measure the blinking frequency

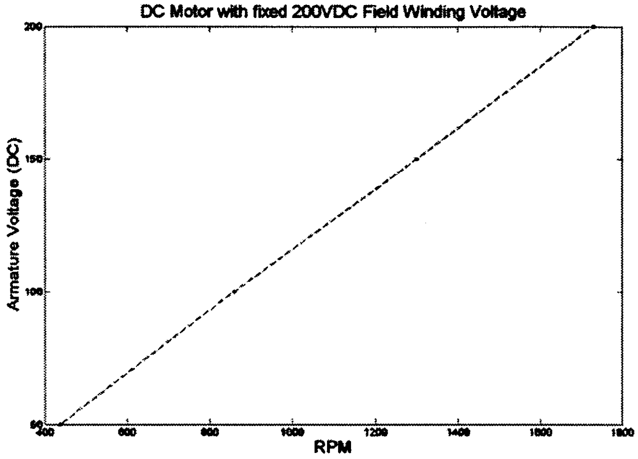
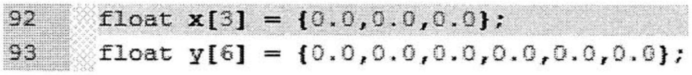


Figure 5 DC Motor Test

$$\text{LAG\_ELEMENT\_SIZE} = \text{frequency}_{\text{Blink}} * \text{T}_D$$

By setting it to the integer closest to this value, it will ensure that the output properly gets delayed .024 seconds. It is currently set to 10.

Finally, all the variables and functions are initialized. All these variables are given an initial value or set to zero either manually by default. Many of these variables match the variables that can be found in the derivations of the Diesel Generator Governor Model in Methodology section of this thesis. A few important variables are created such as X and Y, the input and output of the Diesel Generator Governor Model.

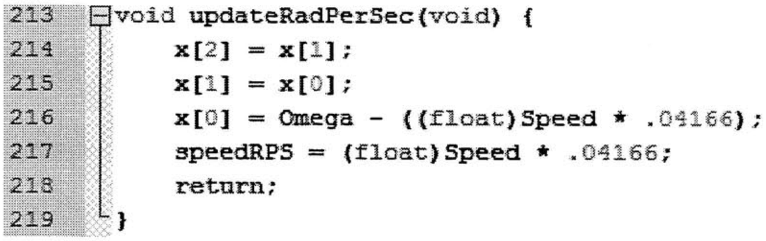


```
float x[3] = {0.0,0.0,0.0};
float y[6] = {0.0,0.0,0.0,0.0,0.0,0.0};
```

Figure 6 Initializing Primary Variables for TF Implementation

These are each specified to be a list of float elements with each additional element representing a sample shift (ie. the left most float in x is x[n] while the right most is x[n-2]). In additional to the transfer function variables, currentTSOut is initialized. This variable represents the desired current and the manipulation of the desired torque torqueTSOut of the implemented version of the Diesel Generator Governor Model. The value of currentTSOut is what the entire control system hopes to emulate with the actual current flowing to the DC Motor.

Main Program

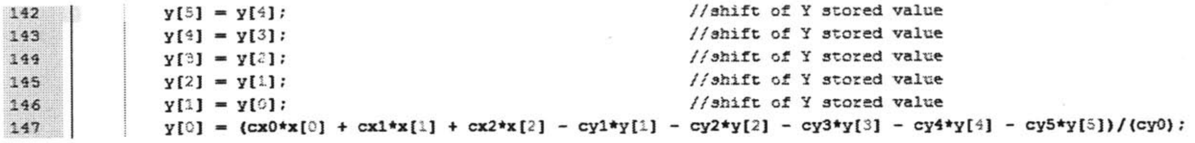


```
void updateRadPerSec(void) {
    x[2] = x[1];
    x[1] = x[0];
    x[0] = Omega - ((float)Speed * .04166);
    speedRPS = (float)Speed * .04166;
    return;
}
```

Figure 7 UpdateRadPerSec Function Implementation

The main program is where the majority of the computations in the system take place. Initially, the external measurements such as actual current and motor speed get updated by calling their given functions. The updateRadPerSec function, while located in the function section of the code, is called at the beginning of the main program to update the value of the X variable. For the values of the Δω, the .04166 scalar was based on a manual correction by using a tachometer on the motor shaft to ensure that the readout matched the value of X.

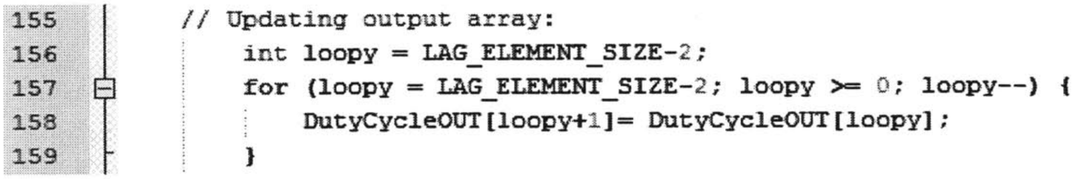
Next, the output torque variable y[n] is updated by performing a shift through all its values and updating the current output based on the previous values as designated by the DT Diesel Generator Governor Model.



```
y[5] = y[4];                                              //shift of Y stored value
y[4] = y[3];                                              //shift of Y stored value
y[3] = y[2];                                              //shift of Y stored value
y[2] = y[1];                                              //shift of Y stored value
y[1] = y[0];                                              //shift of Y stored value
y[0] = (cx0*x[0] + cx1*x[1] + cx2*x[2] - cy1*y[1] - cy2*y[2] - cy3*y[3] - cy4*y[4] - cy5*y[5])/(cy0);
```

Figure 8 Torque Output Update Implementation

To implement the desired lag as specified by the Engine section of the Diesel Generator Model, a list called DutyCycleOUT is specified as being the length of the constant LAG_ELEMENT_SIZE. Here, it can be seen that before setting the next value of the DutyCycleOUT, all the previous values need to be appropriately shifted.



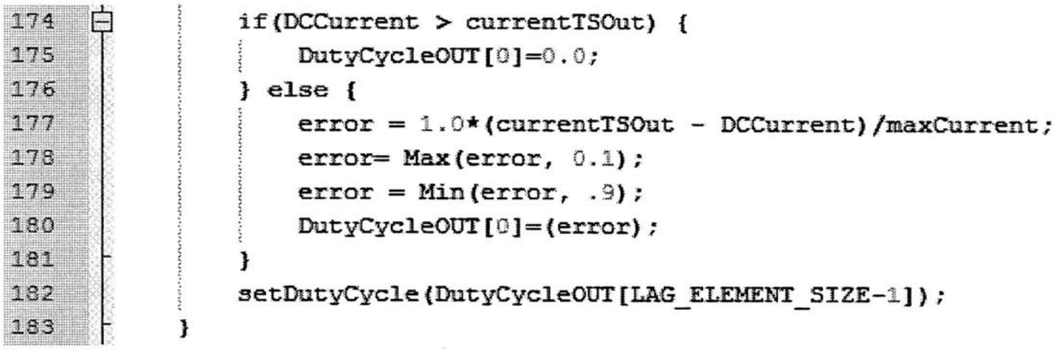
```
// Updating output array:
    int loopy = LAG_ELEMENT_SIZE-2;
    for (loopy = LAG_ELEMENT_SIZE-2; loopy >= 0; loopy--) {
        DutyCycleOUT[loopy+1]= DutyCycleOUT[loopy];
    }
```

Figure 9 Duty Cycle Shift Implementation

Because the value of LAG_ELEMENT_SIZE has been calibrated for the proper delay of .024 seconds with the frequency in which the next duty cycle is updated, this allows for the lag in the actual implementation of the calculation. It can be seen that if the actual current measured from the DCCurrent variable is larger than the allowable current from the transfer function via currentTSOut, the duty cycle is set to the minimum (close to zero) which cuts off additional current being delivered to the armature from the source. It is important to note that there is still freewheeling current going through the motor but this will decay relatively quickly. Once the measured current is smaller than what it allowed, the error is calculated as follows:

$$error = \frac{currentTSOut - DCCurrent}{maxCurrent}$$

This error is what sets the duty cycle for the PWM output from the dsPIC and is added to the end of the DutyCycleOUT list to ensure it gets delayed. The error is bounded between some range to ensure a more steady operation and also to ensure that there is no shoot through which is when the source is shorted to ground.



```
        if(DCCurrent > currentTSOut) {
            DutyCycleOUT[0]=0.0;
        } else {
            error = 1.0*(currentTSOut - DCCurrent)/maxCurrent;
            error= Max(error, 0.1);
            error = Min(error, .9);
            DutyCycleOUT[0]=(error);
        }
        setDutyCycle(DutyCycleOUT[LAG_ELEMENT_SIZE-1]);
    }
```

Figure 10 setDutyCycle Implementation

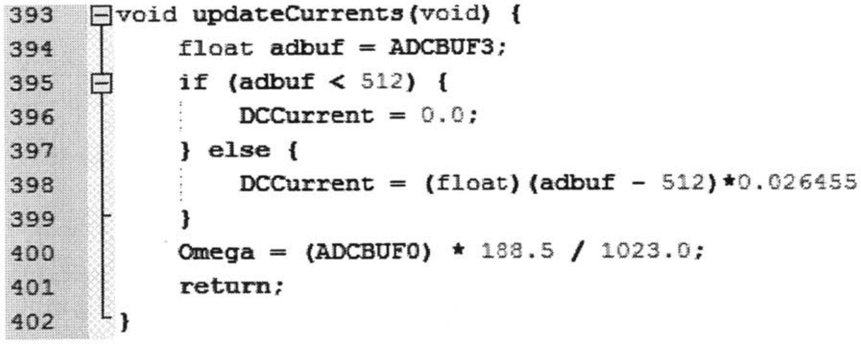
Functions

Within the functions section of the code there are initializing, processing and helper functions. The majority of the initializing functions are for the encoder, PWM output, analog to digital input conversion and the current sensor. The initializing functions are designed to make the appropriate settings on the dsPIC to enable the needed functions for the desired use. The PWM function is enabled to output sets of complementary PWM logic in a half-bring configuration. The Init_Encoder function enables the QEI logic and the 2500 pulses per revolution per input. The ADC (analog to digital conversion) function initializes which channels to make the analog to digital conversions and when to perform the conversion. The current sensor outputs a nominal 2.5 volts if there is no current flowing and a .185 volt per ampere on top of that for positive flowing current. The ADC stores the 0 to 5 volt input in 10 bits over a span of two registers. The 10 bits have a binary range from 0 to 1023 (2^10 -1) so the 2.5 volt input is at the half way point or a digital value of 512. If the current sensor is outputting 5 volts than that implies that there is:

$$\frac{5\ volts - 2.5\ volts}{.185\ volt\ per\ ampere} = 13.513\ amperes = digital\ input\ with\ a\ value\ of\ 1023$$

To maintain that each incremental increase in the input corresponds with the appropriate current increase, $\frac{13.513\ amperes}{(1023-512)} = .0264\ amperes\ per\ digital\ incremental\ increase$

The updateCurrents function first establishes the register where the ADC input value is stored and checks to see if it is under 2.5 volts or a 512 value; this would imply either zero or negative current (which will be treated as a zero current). If there is positive current flow, the input it appropriately scaled and the value of the actual current being delivered to the armature of the DC motor is stored as the DCCurrent variable.



```
void updateCurrents(void) {
        float adbuf = ADCBUF3;
        if (adbuf < 512) {
            DCCurrent = 0.0;
        } else {
            DCCurrent = (float)(adbuf - 512)*0.026455
        }
        Omega = (ADCBUF0) * 188.5 / 1023.0;
        return;
}
```

Figure 11 updateCurrents Implementation

Through the code written in DieselGenerator.C and the proper hardware connections, the DC motor coupled to the synchronous motor will behave like the Diesel Generator Governor Model.

Conclusion

Future Work

While the basis for the Diesel Generator emulator have been programmed, designed and constructed, there is still more work that can be done to perfect the system. The future work would incorporate labview into the control system of the motors. With the labview program helping with the control of the diesel generator, the generator could be made to actively change its frequency in order to synchronize with the other generators and the utility. Currently there is only a passive approach to synchronization, meaning that labview will only monitor the system until synchronization is possible and then connect the two systems.

Summary

As the demand for clean energy and independence from the central utility grows, the use of various alternative energy sources in the form of microgrids will become increasingly more important. My construction and implementation of the diesel generator will allow for the modeling of the effects of various fault durations and loads on the voltage stability of a microgrid. Furthermore, my thesis will help explain the basis for the modeling of diesel generators and how it can be done in a low cost, concise way. Because of my construction of the diesel generator emulator, the research group through the continuation of the project will be able to conduct real time tests in hardware to better understand the effects of islanding on the transient stability of a microgrid. This research will lead to enhanced design and control strategies to improve quality power generation from microgrids in the future.

Works Cited

Alaboudy, A. H. Kasem, H. H. Zeineldin, and J. Kirtley. "A Simple Control Strategy for Inverter Based DG to Enhance Microgrid Stability in the Presence of Induction Motor Loads." Masdar Institute of Science and Technology, Massachusetts Institute of Technology, 2011.

Alaboudy, A. H. Kasem, H. H. Zeineldin, and J. Kirtley. "Impact of DG Interface Control, Microgrid Control Strategy and Load Type on Microgrid Stability." Masdar Institute of Science and Technology, Massachusetts Institute of Technology, 2011.

Aurora, Leony. "Massive Blackout Hits Java, Bali | The Jakarta Post." *Home | The Jakarta Post*. 19 Aug. 2005. Web. 08 May 2011. <http://www.thejakartapost.com/news/2005/08/19/massive-blackout-hits-java-bali.html>.

Kirtley, James, and Hatem Zeineldin. "Construction of a Micro-Microgrid." Research Proposal, MIST/MIT Executive Committee, 2010.

Microchip. "TC9400" Vers. DS21483B, 3, January, 2002.
Microchip Catalog Search, 8 May 2011
http://ww1.microchip.com/downloads/en/DeviceDoc/21483d.pdf

Minkel, Jr. "The 2003 Northeast Blackout--Five Years Later: Scientific American." *Science News, Articles and Information | Scientific American*. 13 Aug. 2008. Web. 08 May 2011. <http://www.scientificamerican.com/article.cfm?id=2003-blackout-five-years-later>.

"U.S. Energy Information Administration - EIA - Independent Statistics and Analysis."*Electricity*. Jan. 2012. Web. 28 Apr. 2012. <http://205.254.135.7/electricity/annual/pdf/tablees1.pdf>.

Appendix

```
// DG for dsPIC30F
// Michael Zieve (mzieve@mit.edu)
// Feb/24/2011

// Pin out notes:
// Updates values of global variables currentA, currentB, currentC and Omega
// CURRENT SENSOR: PIN AN2, read from ADCBUF3

//Program Setup

//Additional Files needed
////////////////////////////////////
//importing functions from additional files in same directory needed to run program
#include "p30F6010A.h"
#include "math.h"
#include "libpic30.h"

//Constants
////////////////////////////////////
#define PWM_PERIOD 1023          //resolution of A/D input from POT1
#define LAG_ELEMENT_SIZE 10  // this value should = frequency of LED D9 *.024 (value of
        Td)

// Time constants used in the transfer function.  Given from Diesel Generator Governor Model
from paper
#define T .0028
#define T1 .01
#define T2 .02
#define T3 0.2
#define T4 0.25
#define T5 0.009
#define T6 0.0384
#define K 40.0

//PID Controller Variables
#define epsilon 0.01  // PID allowable error
#define dt 0.01         //100ms loop time
#define MAX  4.0      //For Current Saturation
#define MIN -4.0       // Min Current
#define Kp  0.1        // proportional gain
#define Kd  0.01       //differnetial gain
#define Ki  0.005      //interal gain

// DC Motor constants
```

```
#define RF 276.6        // Ohms
#define VF 175.0        // Volts
#define G 0.779         // N-m / A^2


//Function Declarations
//////////////////////////////////
// Function declarations. dsPIC runs through these functions in this order. Initializing functions
first and then the main function
void init(void);
void Init_Encoder(void);              // Initialize Encoder information (registers, etc)
void Init_ADC(void);                  // Initialize the Analog/Digital converter
void Init_PWM(void);                  // Initialize settings for PWM
void setDutyCycle(float dc);          // Sets the duty cycle of PWM output to dc
void setDutyCycle2(float dc2);        // Sets the duty cycle 2 of PWM output to dc TEST
void updateCurrents(void);            // Updates values of DC current
void updateRadPerSec(void);           // Updates values of radians per second
void InitTMR1(void);                  // Initialize settings for interrupt timer (used to calculate
ang. velocity)
float Min(float a, float b);          // Returns the lesser of two floats. If equal, return a.
float Max(float a, float b);          // Returns the greater of two floats. If equal, return a.
int main(void);                       // Main function where the program is run

float PIDcal(float setpoint,float actual_position);        //PID function
float fabs(float value);                                   //PID ABS function

// Global Variables//
////////////////////////////////////
// Constants used in the transfer function Constant sclars
float a1 = (float)1.0;                          //scalar for S in denominator in TF
float a2 = (float)(T6 + T5 + T1);               //scalar for S^2 in denominator in TF
float a3 = (float)(T1*(T6 + T5 + T2)+T5*T6);    //scalar for S^3 in denominator in TF
float a4 = (float)(T1*(T2*T6+T5*T6+T2*T5));     //scalar for S^4 in denominator in TF
float a5 = (float)(T1*T2*T5*T6);                //scalar for S^5 in denominator in TF
float b0 = (float)(-K);                         //scalar for constant in numerator in TF
float b1 = (float)(-K*(T4+T3));                 //scalar for S in numerator in TF
float b2 = (float)(-K*(T3*T4));                 //scalar for S^2 in numerator in TF

// Constant sclars used in the transfer function scope. Initialized here and later defined in init()
float cy0;                                      //scalar for y[n]
float cy1;                                      //scalar for y[n-1]
float cy2;                                      //scalar for y[n-2]
float cy3;                                      //scalar for y[n-3]
float cy4;                                      //scalar for y[n-4]
float cy5;                                      //scalar for y[n-5]
float cx0;                                      //scalar for x[n]
float cx1;                                      //scalar for x[n-1]
float cx2;                                      //scalar for x[n-2]

int c;                                          // counter variable
float DCCurrent;
float x[3] = {0.0,0.0,0.0};     // Array stores x[n], x[n-1], and x[n-2] values for angular velocity
        error. Used in transfer function.
float y[6] = {0.0,0.0,0.0,0.0,0.0,0.0};         // Array stores y[n], y[n-1], y[n-2], y[n-3], y[n-4]
        and y[n-5] values for allowable torque. Used in transfer function.
float Omega;            // Between 0 and 1023, reflects angle of potentiometer VR2 on Motor
        Development Board
int Speed = 0;          // Speed of rotor in units of [Angles] per 0.015 [seconds] where [Angle] is
        between [0,2^15]
int AngPos[2] = {0,0};          // Current and previous angle values (in units of [angles] where
        angle is between [0,2^15]
int i;                          // Loop variable
int blink;                      //mz 1-10 test loop variable
int test = 1;                   //mz 1-10 test loop variable
int count2 = 0;

float desiredRadPerSec = 2*3.14;        // Setpoint variable for radians per second
float maxTorque = 2.92;                 // Maximum allowable torque for diesel generator.
float speedRPS = 0.0;
float currentTSOut = 0.0;
float torqueTSOut = 0.0;
float maxCurrent;                       // set to = maxTorque/(G*(VF/RF)) in init()
float error = 0.0;                      // difference between allowed corrent and real current.  sets
        duty cycle for proportional feedback

//PID Controller
float setpoint = 0.0;
float actual_position = 0.0;


// Array to implement output lag
float DutyCycleOUT[LAG_ELEMENT_SIZE];

//Main Program
//////////////////////////////////
// Program entrypoint. Execution always begins here after processor reset.
int main(void) {

        // First initialize all settings and global variables.
        init();
        Init_ADC();
        Init_Encoder();
```

```
    Init_PWM();
    InitTMR1();
    count2 = count2 +1;

    // Main program loop
    while(1) {
        // Housekeeping
        ClrWdt();       // need to periodically clear watchdog timer register
        c = c + 1;      // watch variable to count main loop cycles in watch window

        // Get and calculate current data (current and speed data)
        updateCurrents();       // Get current values from sensor and stores it in Current[]
        updateRadPerSec();  // Calculates and stores current and previous speed error
signals in x (input) array

        //Transfer function update.
        y[5] = y[4];            //shift of Y stored value
        y[4] = y[3];            //shift of Y stored value
        y[3] = y[2];            //shift of Y stored value
        y[2] = y[1];            //shift of Y stored value
        y[1] = y[0];            //shift of Y stored value
        y[0] = (cx0*x[0] + cx1*x[1] + cx2*x[2] - cy1*y[1] - cy2*y[2] - cy3*y[3] -
cy4*y[4] - cy5*y[5])/(cy0);  //updated value for TF output

        torqueTSOut = -1.0*(float)y[0];
        torqueTSOut = Min((float)torqueTSOut, (float)maxTorque);        // Make sure
y[0] <= maxTorque otherwise set output as maxTorque
        torqueTSOut = Max(0.0, (float)torqueTSOut);                 // Make sure y[0] >=
0. otherwise set output as 0

        currentTSOut = (float)torqueTSOut/(G*(VF/RF));              //convert TF torque
output to current


    // Updating output array:
        int loopy = LAG_ELEMENT_SIZE-2;
        for (loopy = LAG_ELEMENT_SIZE-2; loopy >= 0; loopy--) {
            DutyCycleOUT[loopy+1]= DutyCycleOUT[loopy];
        }

        TRISA = 0x0;            //blinking function to check output updating frequency
            if (test == 1) {
                blink = 2;
                test = 0;
            }
            if (blink%2 == 0) {     //if LED if off
```

```
                PORTAbits.RA9 = 1;          //turn LED on
                blink = blink +1;
            } else {                        //if LED is on
                PORTAbits.RA9 = 0;          //turn LED off
                blink = blink +1;
            }

        if(DCCurrent > currentTSOut) {      // if measured current is larger than
allowable current from TF
            DutyCycleOUT[0]=0.0;        //make the duty cycle 0
        } else {        // if measured current is less than allowable current from TF
            error = 1.0*(currentTSOut - DCCurrent)/maxCurrent;      //make error
variable = (Allowable Current - Actual Current)/(Max Current difference possible)
            error= Max(error, 0.1);     //ensure error variable doesn't go negative
            error = Min(error, .9);     //ensure error variable doesn't go above 1
            DutyCycleOUT[0]=(error); //add calculated duty cycle to output array
        }
        //setDutyCycle(DutyCycleOUT[LAG_ELEMENT_SIZE-1]);       //set the output
duty cycle to the last element in the array (this creates the lag)
        setDutyCycle(ADCBUF0/1023.0); // uncomment this to have POT controlled
PWM
    }
    //return 1;
}
//init variables
///////////////////////////////////
void init(void) {
    cy0 = (a1/T + a2/(T*T) + a3/(T*T*T) + a4/(T*T*T*T) + a5/(T*T*T*T*T));
    cy1 = (-a1/T - 2.0*a2/(T*T) - 3.0*a3/(T*T*T) - 4.0*a4/(T*T*T*T) -
5.0*a5/(T*T*T*T*T));
    cy2 = (a2/(T*T) + 3.0*a3/(T*T*T) + 6.0*a4/(T*T*T*T) + 10.0*a5/(T*T*T*T*T));
    cy3 = (-a3/(T*T*T) - 4.0*a4/(T*T*T*T) - 10.0*a5/(T*T*T*T*T));
    cy4 = (a4/(T*T*T*T) + 5.0*a5/(T*T*T*T*T));
    cy5 = (-a5/(T*T*T*T*T));
    cx0 = (b0 + b1/T + b2/(T*T));
    cx1 = (-b1/T - 2.0*b2/(T*T));
    cx2 = (b2/(T*T));

    maxCurrent = (float)maxTorque/(G*(VF/RF));      // setting maxCurrent allowed.  Done
in init() becuase relies on variables

    return;
}

void updateRadPerSec(void) {
    x[2] = x[1];                        //shift of X stored value
```

```
        x[1] = x[0];                               //shift of X stored value
        x[0] = Omega - ((float)Speed * .04166);    // (MZ change .0127832 to .04 from
measurements with tachometer)
        speedRPS = (float)Speed * .04166;          // (MZ change .0127832 to .04 from
measurements with tachometer)
        return;
}


// Initialize settings for the QEI (Quadrature Encoder Interface)
////////////////////////////////////
void Init_Encoder(void)
{
        ADPCFG |= 0x0038;                  // Configure QEI pins as digital inputs
        QEICONbits.QEIM = 0;               // Disable QEI Module
        QEICONbits.CNTERR = 0;             // Clear any count errors
        QEICONbits.QEISIDL = 0;            // Continue operation during sleep
        QEICONbits.SWPAB = 0;              // QEA and QEB not swapped
        QEICONbits.PCDOUT = 0;             // Normal I/O pin operation
        QEICONbits.POSRES = 1;             // Index pulse resets position counter
        DFLTCONbits.CEID = 1;              // Count error interrupts disabled
        //DFLTCONbits.QEOUT = 1;           // Digital filters output enabled for QEn pins
        //DFLTCONbits.QECK = 5;            // 1:64 clock divide for digital filter for QEn
        DFLTCONbits.INDOUT = 1;            // Digital filter output enabled for Index pin
        DFLTCONbits.INDCK = 5;             // 1:64 clock divide for digital filter for Index
        POSCNT = 0;                        // Reset position counter
        //MAXCNT = 9999;             // in x4 mode, QEI will receive 9999 pulses per revolution
        QEICONbits.QEIM = 0b110;           // X4 mode with position counter reset by Index ----
-----(111 to a 110)
        return;
}

// Initialize PWM settings
////////////////////////////////////
void Init_PWM(void) {
        // PWM Time Base Configurations
        PTCONbits.PTMOD = 0b00;            // Free-running mode
        PTCONbits.PTOPS = 0b0000;          // 1:1 Postscale
        PTCONbits.PTCKPS = 0b00;           // PWM time base input clock period is Tcy (1:1
prescale)
        PTCONbits.PTSIDL = 0;              // PWM time base runs in CPU Idle mode

        PTPER = PWM_PERIOD;                // Set the 15-bit period for the pwm period

        PWMCON1bits.PMOD1 = 0;             // PWM I/O pin pair is in complementary output
mode
```

```
    PWMCON1bits.PMOD2 = 0;          // PWM I/O pin pair is in complementary output
mode
    PWMCON1bits.PMOD3 = 0;          // PWM I/O pin pair is in complementary output
mode
    PWMCON1bits.PMOD4 = 0;          // PWM I/O pin pair is in complementary output
mode

    PWMCON2bits.SEVOPS = 0b0000; // PWM Special Event Trigger postcale 1:1
    PWMCON2bits.OSYNC = 0;          // Output overrides via the OVDCON register occur
on next Tcy boundary
    PWMCON2bits.UDIS = 0;           // Updates from duty cycle and period buffer
registers are enabled

    DTCON1bits.DTBPS = 0b01;        // Clock period for dead time unit B is 2 Tcy
    DTCON1bits.DTB = 5;             // Unsigned 6-bit dead time value bits for dead time
unit b
    DTCON1bits.DTAPS = 0b01;        // Clock period for dead tmie unit a is 2 Tcy
    DTCON1bits.DTA = 5;             // Unsigned 6-bit dead time value bits for dead time
unit a

    DTCON2bits.DTS4A = 0;               // Dead time provided from unit A (active)
    DTCON2bits.DTS4I = 0;               // Dead time provided from unit A (inactive)
    DTCON2bits.DTS3A = 0;               // Dead time provided from unit A (active)
    DTCON2bits.DTS3I = 0;               // Dead time provided from unit A (inactive)
    DTCON2bits.DTS2A = 0;               // Dead time provided from unit A (active)
    DTCON2bits.DTS2I = 0;               // Dead time provided from unit A (inactive)
    DTCON2bits.DTS1A = 0;               // Dead time provided from unit A (active)
    DTCON2bits.DTS1I = 0;               // Dead time provided from unit A (inactive)

    PDC1 = 100;                         //PWM Channel 1 being used for DC motor
of Diesel Generator (TP8 and TP18)
    PDC2 = 100;                               //PWM Channel 2 being used for DC
motor of inductive load         (TP7 and TP17)
    //PDC3 = 100;
    //PDC4 = 100;

    PTCONbits.PTEN = 1;             // Enable the PTEN bit (time base enable bit)
    //PWMCON1bits.PEN4H = 1;        // PWMxH pin is enabled for PWM output
    //PWMCON1bits.PEN3H = 1;        // PWMxH pin is enabled for PWM output
    PWMCON1bits.PEN2H = 1;          // PWMxH pin is enabled for PWM output
    PWMCON1bits.PEN1H = 1;          // PWMxH pin is enabled for PWM output
    //PWMCON1bits.PEN4L = 1;        // PWMxL pin is enabled for PWM output
    //PWMCON1bits.PEN3L = 1;        // PWMxL pin is enabled for PWM output
    PWMCON1bits.PEN2L = 1;          // PWMxL pin is enabled for PWM output
    PWMCON1bits.PEN1L = 1;          // PWMxL pin is enabled for PWM output
```

```
   LATD  = 0x0000;
   TRISD = 0xD70F;
        return;
}

//Set Duty Cycle
////////////////////////////////////
// Sets the PWM duty cycle to dc (a floating point value between 0 and 1)
void setDutyCycle(float dc) {
        PDC1 = (int)(PWM_PERIOD * dc)*2;
        return;
}

//testing for 2nd pwm channel
void setDutyCycle2(float dc2) {
        PDC2 = (int)(PWM_PERIOD * dc2)*2;
        return;
}

//EX#2/
//NOTES: MAX_COUNT_PER_REV = PULSES_PER_REV x COUNT_INC_PER_REV - 1 =
2500 × 4 – 1 = 9999
void PositionCalculation(void)
{
        AngPos[1] = AngPos[0];
        AngPos[0] = (float)POSCNT;

        return;
}

//EX#3
// December 2, 2011: Empiracally verified that the TMR1_interrupt gets called every 30 ms.
//NOTES:INTERRUPT_PERIOD = 60/(2 x MAX_SPEED_RPM) = 60/2*2000 = .015
void InitTMR1(void)
{
        TMR1 = 0;                   // Reset timer counter
        T1CONbits.TON = 0;          // Turn off timer 1
        T1CONbits.TSIDL = 0;        // Continue operation during sleep
        T1CONbits.TGATE = 0;        // Gated timer accumulation disabled
        T1CONbits.TCS = 0;          // use Tcy as source clock
        T1CONbits.TCKPS = 2;        // Tcy / 64 as input clock
        PR1 = 432;                  // Interrupt period = 0.015 sec with a 64 prescaler *******
---(864 gives 30 ms so changed to 432 for 15 ms)
        IFS0bits.T1IF = 0;          // Clear timer 1 interrupt flag
        IEC0bits.T1IE = 1;          // Enable timer 1 interrupts
        T1CONbits.TON = 1;          // Turn on timer 1
        return;
}

//EX#4
// Interrupt routine occurs every 0.015 seconds.
void __attribute__((__interrupt__)) _T1Interrupt (void)
{
        IFS0bits.T1IF = 0;                  // Clear timer 1 interrupt flag
        //ISF2bits.QEIIF = 0;                 //clear QEI interrupt bit  MZ add
        PositionCalculation();              // Store current and previous POSCNT values into
AngPos array
        Speed = AngPos[0] - AngPos[1];      // Calculate speed in terms of [Angles] per 0.015
[seconds]

        return;
}

// Initialize the settings the for the ADC converter to
// read current values from the hall effect sensors.
void Init_ADC(void) {
   /* ADPCFG: ADC Port Configuration Register */
   ADPCFG = 0xFFFF;                          // Set all ports digital
   ADPCFGbits.PCFG7 = 0;                     // AN7 analog
        ADPCFGbits.PCFG0 = 0;               //test 2-1-12
        ADPCFGbits.PCFG1 = 0;               //test 2-1-12
        ADPCFGbits.PCFG2 = 0;               //test 2-1-12
   //ADPCFGbits.PCFG0 = 0;                   // AN3 analog   //test comment
   //ADPCFGbits.PCFG1 = 0;                   // AN4 analog   //test comment
   //ADPCFGbits.PCFG2 = 0;                   // AN5 analog   //test comment

        ADCON2bits.VCFG = 0b000;            // Select voltage reference source to match range
(0V-5V)
        ADCON3bits.ADCS = 0b000000;         // Select analog conversion clock to Tcy / 2
        ADCON2bits.CHPS = 0b10;             // Use S/H channel CH0, CH1, CH2, CH3
        ADCON3bits.SAMC = 0b10000;          // Sample time 16 Tad
        ADCON1bits.SIMSAM = 1;              // Sample all channels simultaneously, then convert
sequentially
        ADCSSL = 0;                         // Skip ANx for x in <0:15> on input scan
        ADCHS = 0x0707;               // CH0: (+)AN7, CH1: (+)AN0, CH2: (+)AN1, CH3:
(+)AN2 //\\ (-)Vref
        //ADCHS = 0b0000011100100111;  // test comment
        ADCON1bits.SSRC = 0b111;            // Start conversion immediately after sampling
        ADCON1bits.ASAM = 1;                // Start sampling immediately after conversion
        ADCON1bits.FORM = 0b00;          // Present conversion results in buffer as unsigned
integer
        ADCON2bits.SMPI = 0b1111;        // Interrupt at completion of each 16th
sample/convert sequence

        ADCON1bits.ADON = 1;             // Turn on A/D module
        return;
}


// Updates values of global variables currentA, currentB, currentC and Omega
// CURRENT SENSOR: PIN AN2, read from ADCBUF3
void updateCurrents(void) {
        float adbuf = ADCBUF3;
        if (adbuf < 512) {
                DCCurrent = 0.0;
        } else {
                DCCurrent = (float)(adbuf - 512)*0.0264550;

        }
        Omega = (ADCBUF0) * 188.5 / 1023.0;
        return;
}

// Math helper function. Returns the min of two floats
float Min(float a, float b) {
        if(b < a) { return b; }
        return a;
}

// Math helper function. Returns the max of two floats.
float Max(float a, float b) {
        if(b > a) { return b; }
        return a;
}

float fabs(float value) {
        if (value < 0.0) {return -1.0*value;}
        else {return value;}
}
float PIDcal(float setpoint,float actual_position) {
        static float pre_error = 0;
        static float integral = 0;
        float error;
        float derivative;
        float output;
        //Caculate P,I,D
        error = setpoint - actual_position;

        //In case of error too small then stop integration
        if(fabs(error) > epsilon)
        {
                integral = integral + error*dt;
        }
        derivative = (error - pre_error)/dt;
        output = Kp*error + Ki*integral + Kd*derivative;

        //Saturation Filter
        output = Min(output, MAX);
        output = Max(output, MIN);

//Update error
        pre_error = error;

 return output;
}
```

Reference Manual for Microgrid Hardware Simulation System

by

Jared P. Monnin

S.B., E.E. M.I.T., 2011

and

Michael M. Zieve

S.B., E.E. M.I.T., 2011

May 20, 2012

Table of Contents

List of Figures

1. Purpose

The purpose of this document is to act as a reference manual for the microgrid hardware simulation system currently being constructed in the Laboratory of Electromagnetic and Electronic Systems at the Massachusetts Institute of Technology, under the supervision of Prof. James L. Kirtley, Jr. In the following pages, we seek to explain in detail all of the work that has been accomplished on this project, from January of 2011 to the time of writing, in May of 2012. We have included detailed written descriptions, photographs, circuit diagrams, part numbers, links to datasheets, and in several cases, our rationale behind certain design decisions. We divided this report into sections based on the major components of the overall system, and we aim for each section to be largely independent of the other sections. It is our hope that anyone who joins this project in the future will be able to reference this document to answer any question they might have about the system we have put in place, as it existed at the time of writing.

2. System Overview

This section is a broad overview of the electrical and mechanical aspects of the project, describing the various components of the project, their interconnections, and their place within the wider project. Each component described in this section is described in detail later in this document.

2.1 Electrical Connections

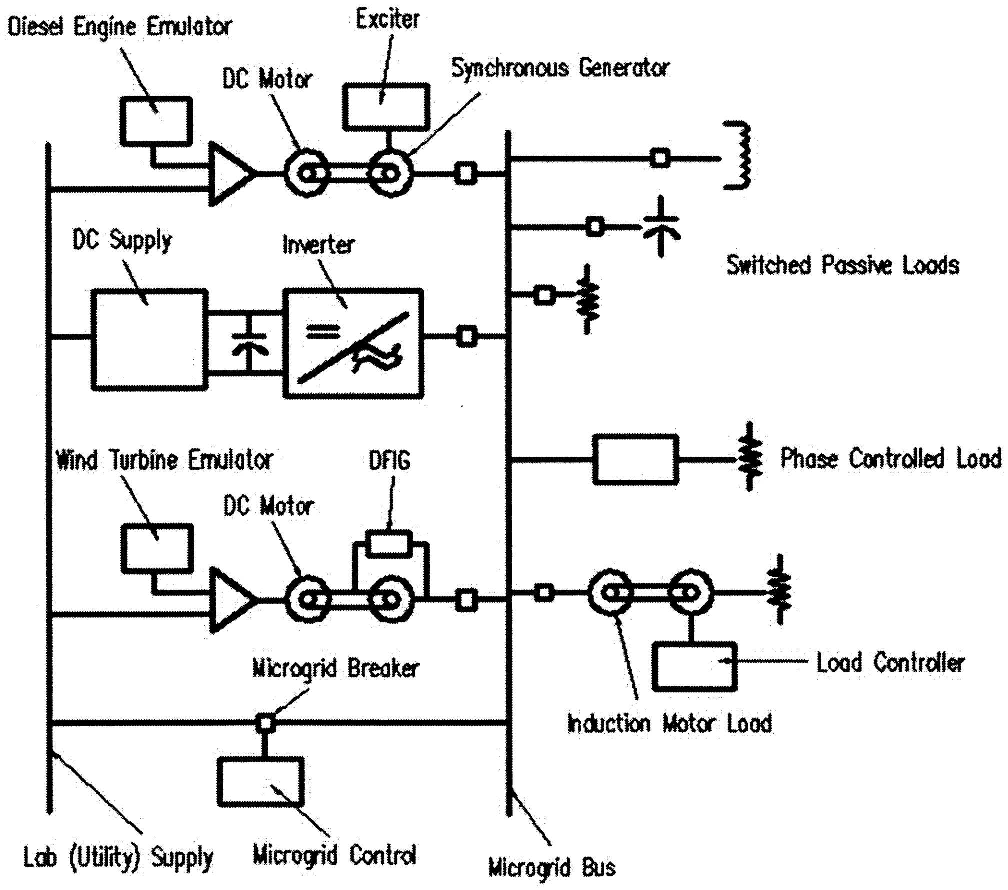


Figure 2-1 System Overview

The image above is a one-line diagram of the initial design of our project. At the time of writing, we have already implemented the microgrid bus, load and generator relays, resistive, capacitive, inductive, phase-controlled resistive, and induction motor loads, the solar farm emulator using the DC supply and inverters, the diesel generator emulation system, and a voltage and current measurement system.

2.1.1 Utility Connection

Power is supplied to the project from the lab utility supply through a rubber-insulated, three-phase, five-wire cable that should be connected only to the $120V_{l\text{-}n}/208V_{l\text{-}l}$ three-phase receptacle labeled L2 22. This receptacle is connected to breaker 22 in Load Center L2 to the right of the bus side of the project board, on the wall opposite room 10-097. The other end of this cable is connected to an emergency shut-off switch mounted on the project board under the project load center. Besides providing the ability to quickly disconnect power from the project, this box also contains three 30-ampere, 250-volt, Bussmann fuses. The emergency shut-off switch is then connected to the project load center, which contains the circuit breakers

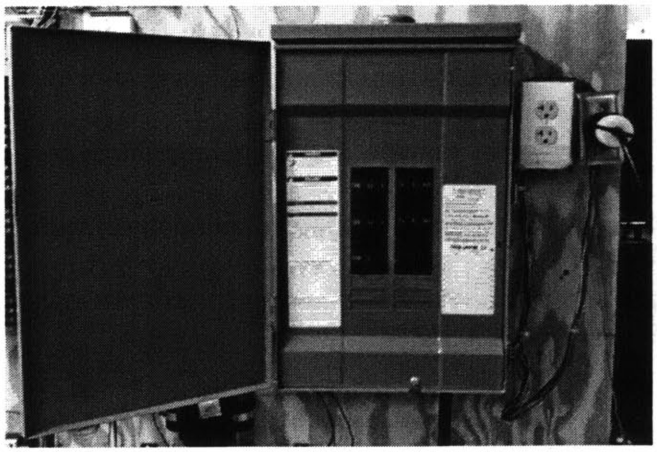


Figure 2-3 Load Center and Outlets

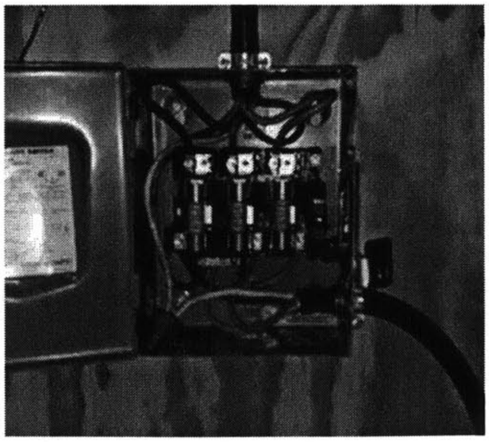


Figure 2-2 Emergency Shut-off Switch

that connect and disconnect different parts of the project. Besides a breaker connecting the utility to the microgrid bus switch, there are breakers connecting several single-phase and three-phase power outlets on the project board that are used for auxiliary power supplies and monitoring systems. The breaker diagram is included below:

Figure 2-4 Load Center Breaker Diagram

1	1-phase Outlet C	Not Connected	2
3	Not Connected	1-phase Outlets A & B	4
5	Not Connected	Not Connected	6
7	Not Connected	3-phase Outlets L1	8
9	Not Connected	3-phase Outlets L2	10
11	Not Connected	3-phase Outlets L3	12
13	Microgrid L1	No Breaker	14
15	Microgrid L2	No Breaker	16
17	Microgrid L3	No Breaker	18
19	No Breaker	No Breaker	20

As noted above, the bottom left breaker in the load center connects the utility supply to the main microgrid solid-state relay. The other side of this relay is connected to our microgrid bus, which is constructed from a line of interconnected terminal strips. All of our loads are connected to this bus through electromechanical relays, and all of our generation sources are connected to this bus through solid-state relays. More information on these individual components, their associated control systems, the measurement system, and the methods used to synchronize the microgrid bus to the utility supply are detailed in their respective sections below.

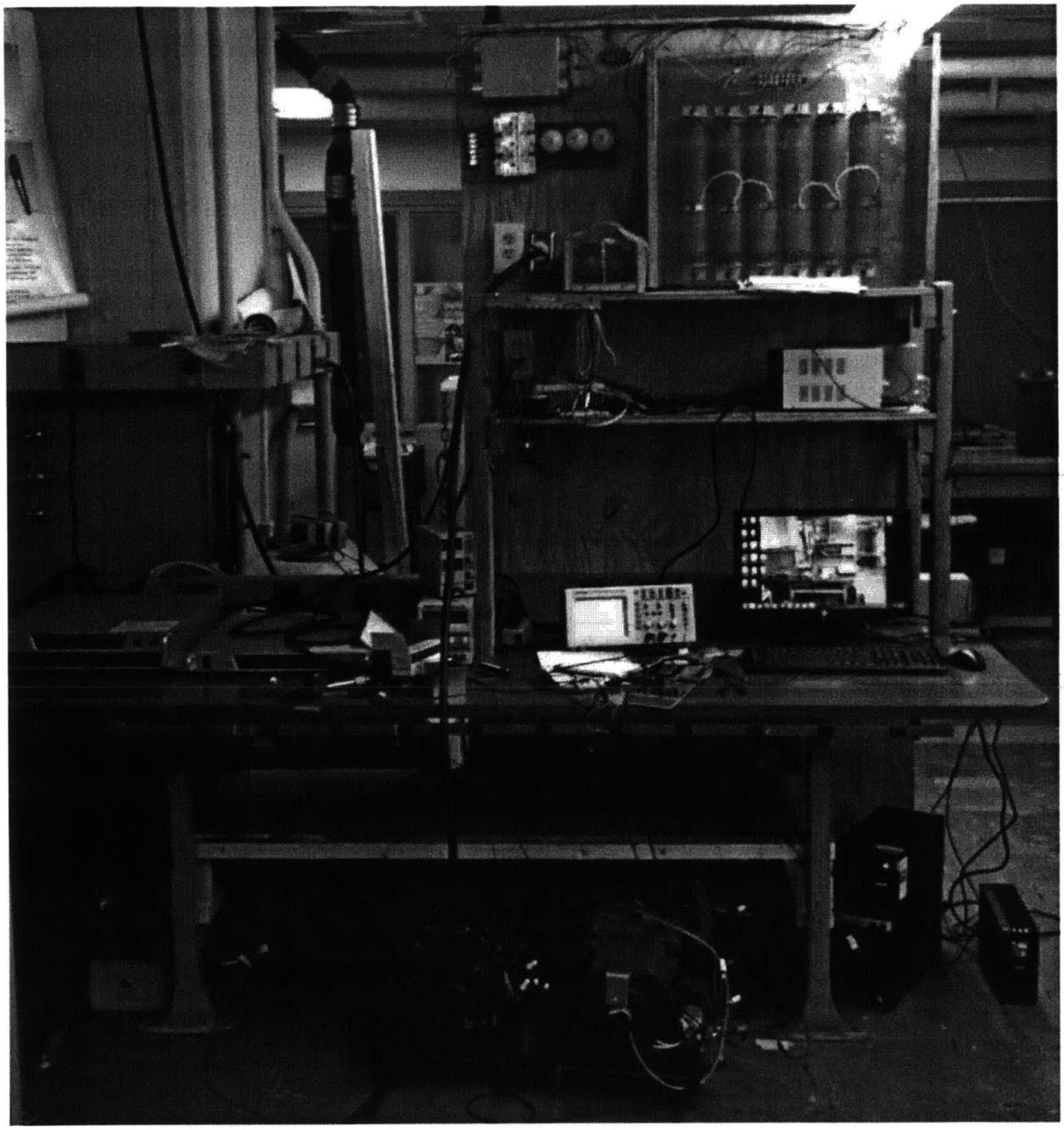


Figure 2-5 Load Side of Project Board

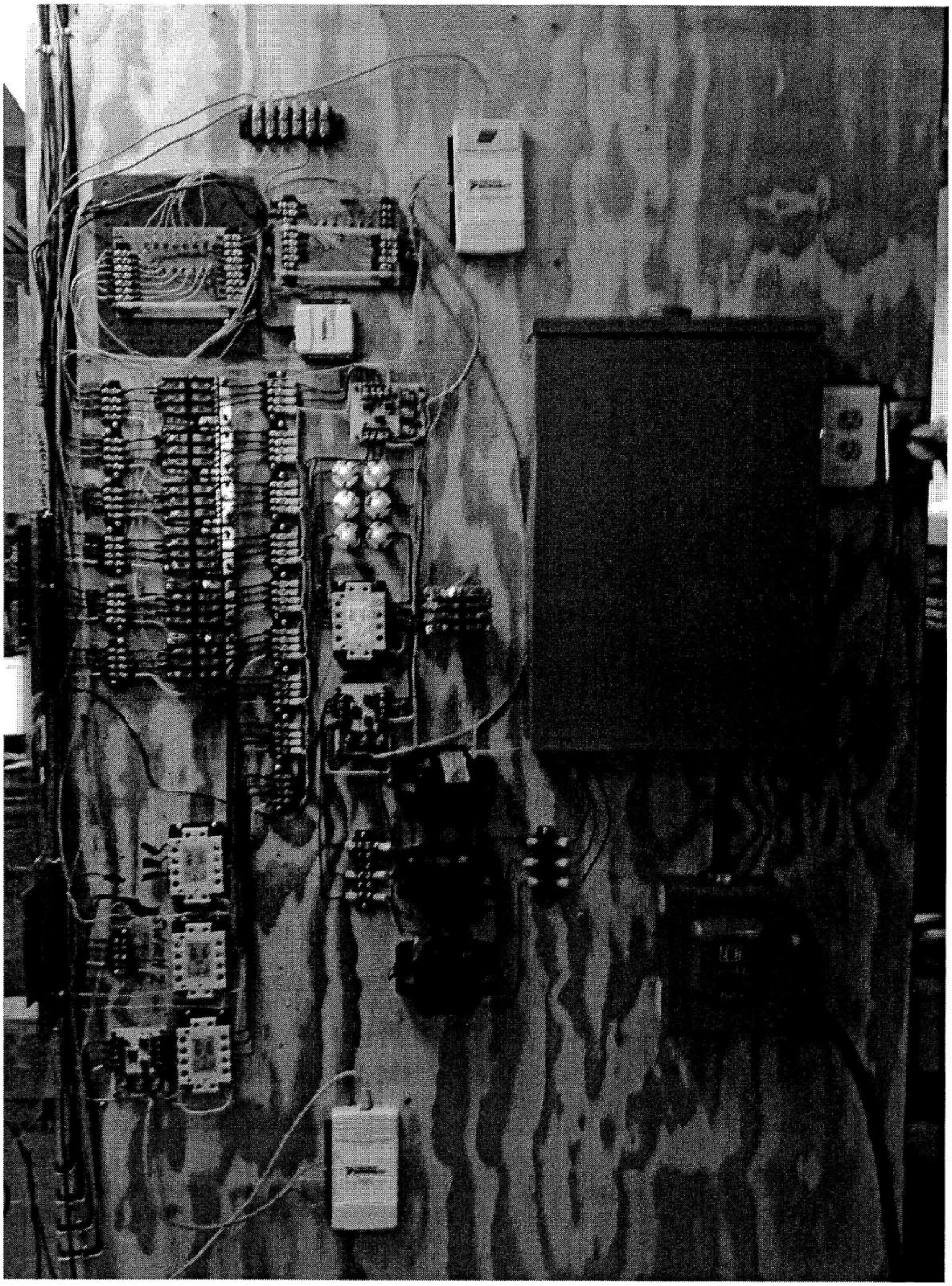


Figure 2-6 Bus Side of Project Board

2.1.2 Project Computer

Figure 2-7 Project Computer

Many of the components of our system described in later sections rely on our project computer, which is located under the desk on the load side of the project board. We purchased an HP desktop computer, the HP 505B Minitower Business PC with Windows 7, a monitor, and licenses for LabVIEW and other useful software. This computer interfaces with the measurement system DAQs and the Matsusada power supplies via several USB ports. Our main LabVIEW control interface runs on this computer, and all of the power supply control programs and dsPIC code versions can be found on this computer. As of the time of writing, the login information is:

Username:	Programmer
Password:	picogrid

We noticed that our computer was sensitive to the switching transients induced by our reactive loads, especially the induction motor load. During these transients, the computer would lose its connection to the control and measurement DAQs before quickly reconnecting to them, which interfered with data collection. This problem persisted even when we plugged the computer into a power outlet connected to a different load center, so we purchased an APC uninterruptible power supply and voltage regulator, which largely solved the transient problem.

2.1.3 Starting Procedure

The following checklist describes the steps in which the microgrid system should be started to ensure safe operation:

1. Turn on the computer and connect the DAQs via USB. With the DAQs powered up, launch LabVIEW, and run the control program.
2. Only after the DAQs are powered up and the LabVIEW program is running, turn on the power supply to the switches. **DO NOT TURN ON THE POWER SUPPLY WITHOUT LABVIEW RUNNING AND THE DAQS POWERED UP.** Turning on the power supply in this condition will turn on every switch on the board, with potentially catastrophic results.
3. Test each switch by clicking on the corresponding button in LabVIEW. The electromechanical switches will make an audible click. All switches have an LED indicator that turns on when voltage is applied to the switch.
4. Check the load center to ensure that the breakers are on or off as desired.
5. Connect the project to the utility power supply by plugging the main cable into the 3-phase outlet (L2 22) next to the project.
6. Turn the emergency off switch to the on position to power up the load center. At this point, if the bottom left breaker is on, the utility is connected to the main bus relay.
7. If generation sources are running and connected to the microgrid bus, **see the synchronization section before connecting the utility to the microgrid bus.** Failure to do so can result in large currents and torques on the motors, with potentially disastrous results.
8. Loads can now be connected and disconnected through the LabVIEW interface.

2.2 Hardware Mounting

As shown in Figure 2-5 and Figure 2-6 above, the emergency shut-off switch, load center, microgrid bus, and electromechanical and solid-state relays are all mounted on a vertical plywood board which is fastened to the back of a heavy workbench. We refer to this side of the project board as the "bus side". On the workbench side of the project board, which we call the "load side", we have constructed several shelves above and below the workbench. These shelves support several of our loads, some auxiliary power supplies, and important spare parts for the system. The shelf beneath the desk is intended to hold all of the control boards and power supplies for the motor sets used in the diesel generator, wind farm emulator, and induction motor load systems.

2.3 Safety Measures

To prevent accidental contact with the electrical terminals on the bus side of the project board, we have constructed a clear fiberglass cover that latches to the load center with a weak permanent magnet and can be held shut more securely with a hook and rubber band connected to the cover. This cover should be latched securely before any power is connected to any part of the microgrid system. On the load side of the project board, we have constructed wood and metal screening covers for all accessible components that might present an electrical or thermal hazard if accidentally touched. In the event of any unintended fault on the system or any electrical accident, power to the microgrid should immediately be disconnected by shutting off the emergency shut-off switch. Utility power and all microgrid generation sources and loads can be isolated from the microgrid bus by shutting off the switch driver board power supply.

3. Generators

There are currently two generation sources in our system, with the possibility of adding a third source in the near future. These generation sources are described in this section.

3.1 Diesel Generator

The diesel generator is the master generator in the microgrid because it sets the voltage, phase and frequency of the system in the absence of the utility. The diesel generator is designed to supply between 500 watts and 1 kilowatt of power to the microgrid. The diesel generator in our system is comprised of a 1.5-horsepower DC motor and a 1.0-horsepower wound-rotor induction motor. The two motor shafts are coupled together using a 22-mm Lovejoy Sintered Iron Jaw Coupling with a LOVEJOY Buna-N Insert (size A/L075) and are mounted together on a steel plate. Note for future applications, the actual shaft size is a standard 7/8-inch shaft and the metric 22-mm shaft coupling was bored out another 0.023 inches in order for it to fit properly on the shaft.

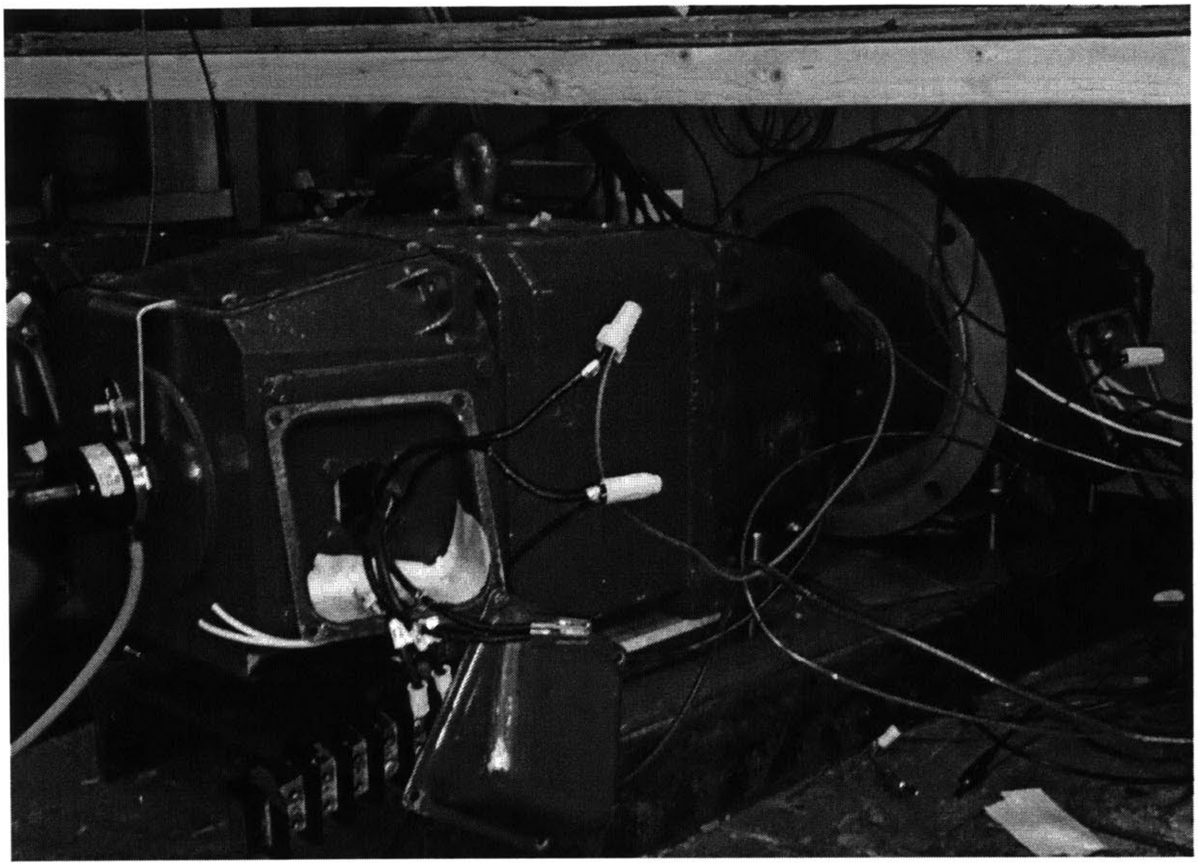


Figure 3-1 Diesel Generator (DC Motor on left and Induction Motor on right)

3.1.1 Accessories

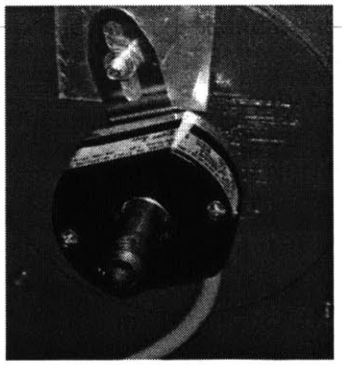


Figure 3-2 Red Lion ZPJ

Mounted on the secondary shaft of the DC motor is a Red Lion ZPJ large through-bore rotary pulse generator. This incremental encoder has three NPN Open Collector Transistor outputs that provide standard A, B and Z encoder quadrature outputs. Output B leads output A by 90 degrees for counter-clockwise rotation of the motor shaft and each of these outputs provides 2500 pulses per revolution. The Z output provides an index pulse and therefore outputs one pulse per revolution. There are 2500*2 + 1 = 5001 pulses per revolution, which provide a high resolution measurement for the direction, speed and position of the rotor. Note

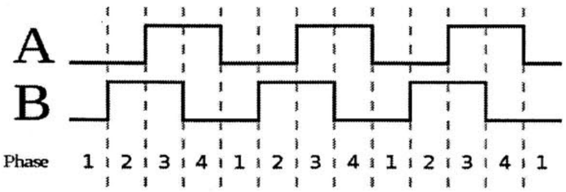


Figure 3-4 Counter-Clockwise Quadrature Encoder Logic

that while position can be calculated, only the direction and speed of the rotor shaft are used. It is important that all of these outputs are connected to the desired output voltage, in our case +5VDC, through pull-up resistors or else there will be no output. Additionally, the encoder is mounted to the secondary shaft side of the DC Motor such that it is centered on the secondary shaft. To ensure a firm connection between the internal coupling of the encoder and the DC motor secondary shaft, a 12-mm insert is used. Note for future use, the secondary shaft is actually 0.5 inches, or 12.7 mm.

There is also an Allegro Microsystems Inc. ACS712 current sensor board, which we used to measure the current driving the DC motor. The 05B version of the board has a current measuring range for both alternating and direct current within the range of ±5.0 amperes. A zero ampere measurement results in a nominal 2.5-V output with a 185-millivolt per ampere sensitivity in measurement. For example, if one ampere is flowing in the positive direction through the current sensor, it will output 2.5 + 0.185 = 2.685 volts. Note that we use a few varieties of Allegro Microsystems current sensor boards, all with different current ranges and sensitivities. The 05B model is used in the diesel generator system because it provides the highest resolution in the desired current range.

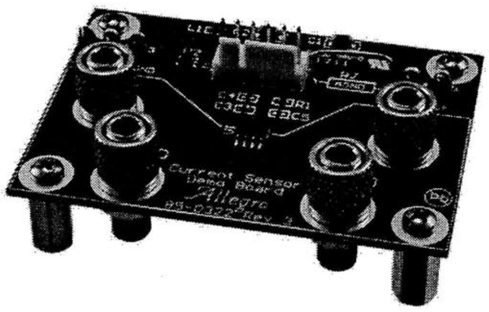


Figure 3-3 ACS712 Current Sensor Board

Additionally, there is a dsPICDEM MC1 Motor Control Development Board, which we use as the controller for the system. The board uses a Microchip dsPIC30F6010 High Performance Digital Controller as its central processor. The dsPIC30F6010 has a 16-bit architecture and has 80 pins, 68 of which have I/O functionality. It also has functionality for 8 pulse width modulation channels. The dcPIC30F6010 can be removed and replaced by another microcontroller if needed simply by replacing the mounted red card, as shown below.

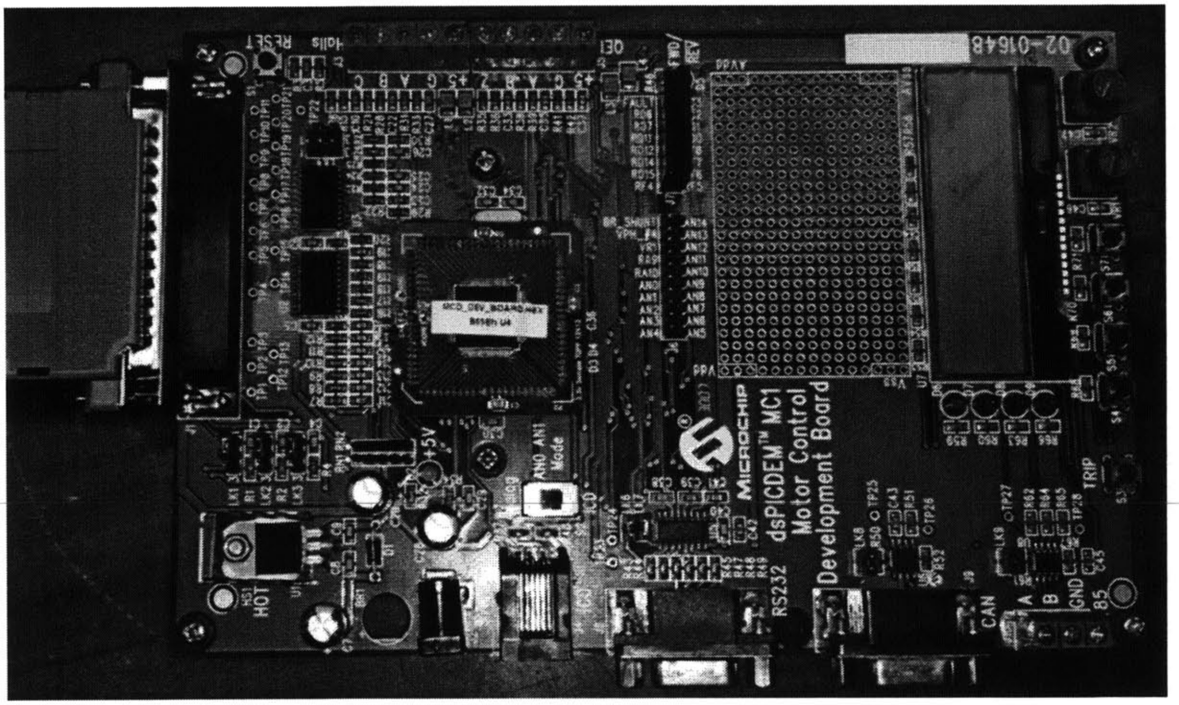


Figure 3-5 MC1 Motor Control Development Board

The Motor Control board and dsPIC are programmed using the free MPLAB Integrated Development Environment software that can be downloaded from www.microchip.com. This software supports programming in both Assembly and C. To debug and program the dsPIC, the MPLAB ICD 3 In-Circuit Debugger 3 System is used. This system connects to the computer via USB and programs the dsPIC via an Ethernet cord connected to the ICD port on the board.

Note that to properly use the ICD 3, the AN0 Mode switch must be switched from its Analog position to ICD.

Figure 3-6 MPLAB ICD 3

3.1.2 Model

While the DC motor acts as the prime mover and the coupled induction motor acts as the three-phase generator, what makes this system emulate a diesel generator is our use of the diesel generator governor model shown below:

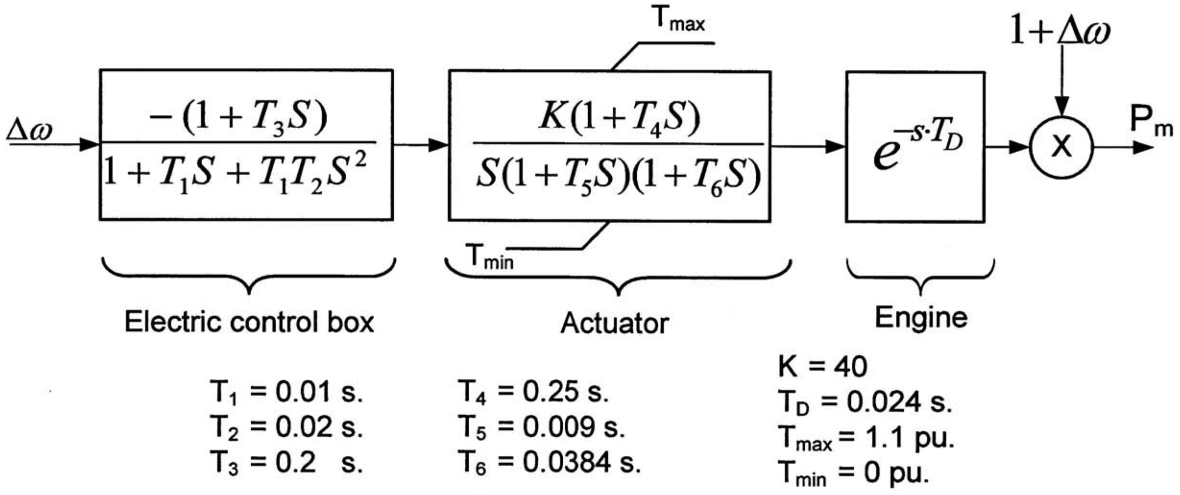
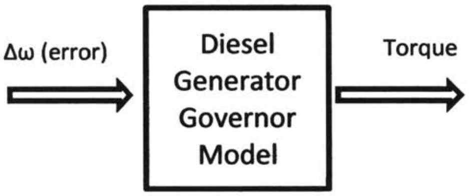


Figure 3-7 Diesel Generator Governor Model

The diesel generator governor model represents a continuous time transfer function that models the behavior of a diesel generator. It takes an input of a mechanical frequency error and outputs a power command. Because torque multiplied by mechanical frequency is power, $P_m = T * \omega$, the torque output of the system is what is measured in our model. For a DC motor, torque is proportional to armature current, so we measure the current used to drive the DC motor and scale it such that it represents the torque output of the system. Similarly, we measure the rotational speed of the motor shaft using the encoder to determine the mechanical speed, which allows us to determine the frequency error. For simplicity, the diesel generator system that we implemented can be thought of as a simple block diagram:



3.1.3 I/O

The diesel generator system combines many different parts and boards. Descriptions of the various pin-outs are listed below for each motor and component.

DC Motor

DC Motor Wires	Color	Connection
F1	Black	V+ (Field Power Supply)
F2	Black	F3 (on DC Motor)
F3	Black	F2 (on DC Motor)
F4	Black	V- (Field Power Supply)
A1	Black	IP- (on Current Sensor Board)
A2	Black	(Power) Ground
Temperature Sensor (2)	White	Not Connected

This DC motor field configuration results in a high voltage connection where the field windings are connected in series.

Synchronous Generator (Wound-Rotor Induction Motor)

Generator Wires	Color	Connection
T1	Black	T7 (Induction Motor)
T2	Black	T8 (Induction Motor)
T3	Black	T9 (Induction Motor)
T4	Black	T5 and T6 (Induction Motor)
T5	Black	T4 and T6 (Induction Motor)
T6	Black	T4 and T5 (Induction Motor)
T7	Black	T8* and T9* (Induction Motor)
T8	Black	T7* and T9* (Induction Motor)
T9	Black	T7* and T8* (Induction Motor)
M1	White	V+ (Synchronous Generator Supply)
M2	White	V- (Synchronous Generator Supply)
M3	White	Not Connected

Note that connections with a * such as T7-T8 -T9 are all internally connected.

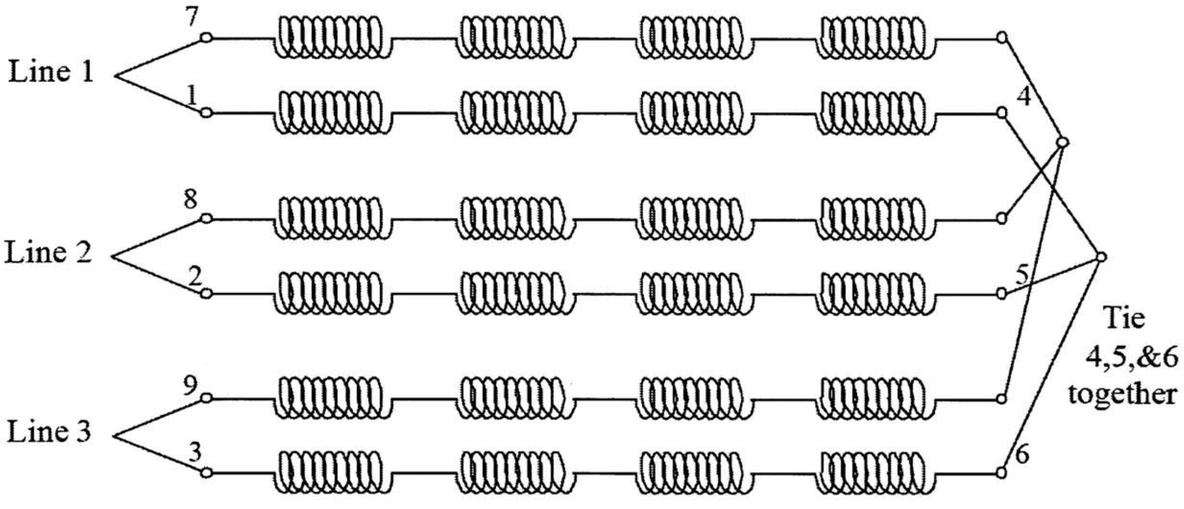


Figure 3-8 Synchronous Generator Armature Windings

This configuration of the armature windings allows for a three-phase AC output.

Encoder

Encoder Wires	Color	Connection	I/0	A/D
+VDC	Red	+5V on QEI connector (on Motor Control Board)	Input	Power
COM	Black	G on QEI connector (on Motor Control Board)	Input	Ground
A	White	A* on QEI connector (on Motor Control Board)	Input	Digital
B	Green	B* on QEI connector (on Motor Control Board)	Input	Digital
Z	Orange	Z* on QEI connector (on Motor Control Board)	Input	Digital

Note that connections with a * such as A, B and Z are all additionally connected through a 1000-ohm pull-up resistor to +5VDC.

Current Sensor Board

Current Sensor Board Pins	Color	Connection	I/O	Type
Vin	N/A	Not Connected	NC	NC
+5V	N/A	+5V (on Motor Control Board)	Input	Power
AGND	N/A	Ground (on Motor Control Board)	Input	Ground
Vout	N/A	AN2 (on Motor Control Board)	Output	Analog
CF	N/A	Not Connected	NC	NC
IP+	N/A	Motor Output 1 (from Inverter1)	Input	Analog
IP-	N/A	A1 (on DC Motor)	Output	Analog

Note that to connect to the Current Sensor Board, a 6 position WM4267-ND Connection Housing is needed for the board, and the incoming wires must be crimped with a WM2510-ND Female Connection Terminal.

dsPIC Board (dsPICDEM MC1 Motor Control Development Board)

The dsPIC board is powered by a +9VDC wall adaptor that plugs into the 9N IN connector. As mentioned earlier, the ICD3 Ethernet cord connects to the ICD port for debugging and programming, and the ANO switch must be switched to ICD for it to work. The dsPIC board has many different signal connection points, and therefore the listing is broken up by connection terminal.

dsPIC Board (J1 connector)	**Wire Color**	**Connection**	**Test Point**	**I/O**	**Type**
Pin 13 (PWM Phase 1 HIGH)	Yellow	Inverter input 1	TP18	Output	Digital
Pin 31 (PWM Phase 1 LOW)	Green	Inverter input 2	TP8	Output	Digital
Pin 12 (PWM Phase 2 HIGH)		Inverter2 input 1	TP17	Output	Digital
Pin 30 (PWM Phase 2 LOW)		Inverter2 input 2	TP7	Output	Digital
Pin 19 (+5V)			N/A	Output	Power
Pin 18 (Ground)			N/A	Output	Ground

Note that the J1 connecter is a Male D-subminiature 37 (37 pins).

dsPIC Board (QEI port)	**Color**	**Connection**	**I/O**	**Type**
+5	Red	+5V on Encoder	Output	Power
G	Black	COM on Encoder	Output	Ground
A	White	A on Encoder	Input	Digital
B	Green	B on Encoder	Input	Digital
Z	Orange	Z on Encoder	Input	Digital

dsPIC Board (J7 port)	**Color**	**Connection**	**I/O**	**Type**
AN2	Yellow	Vout (on Current Sensor Board)	Input	Analog
VR1		AN0 (on dsPIC Board)	Output	Analog
AN0		VR1 (on dsPIC Board)	Input	Analog

Note that to connect to the J7 port, a 16-position WM2525 Connection Housing is needed for the board and the incoming wires must be crimped with a WM2510-ND Female Connection Terminal.

3.2 Solar Farm

The solar farm hardware simulation in our project is meant to emulate an array of photovoltaic modules that can provide three-phase power to a microgrid using small power inverters. We built a basic version of this hardware simulation using three microinverters and a USB-controllable DC power supply connected to the project computer.

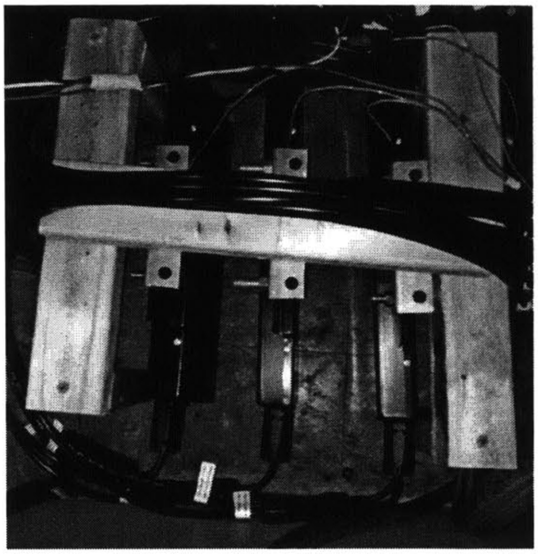


Figure 3-9 Enphase Microinverter Array

3.2.1 Inverter Array

Rather than build inverters from scratch, we purchased three Enphase D380 microinverters. These inverters are capable of outputting up to 380 watts each at $208V_{\text{line-line}}$ and can accept input voltages of 28V to 56V DC. These three inverters are mounted on a wooden frame, which sits to the left of the project board on the load side of the board. The microinverter input wires are terminated with MC4 connectors, so we bought six MC4 extension cables, cut them in half, and used them to connect the twelve microinverter inputs to a terminal strip, which we then connected to the V+ and V- wires of the DC power supply. Note that the Enphase D380 is actually a dual-pack module consisting of two Enphase M190 modules, so there are twice as many electrical connections as one might expect. Once we connected the module inputs to a terminal strip, we used an Enphase AC Branch Cable to connect the module outputs in a Y configuration with the neutral point connected to the neutral wire of our microgrid bus. This branch cable was then connected to the third solid-state relay (counting from the top of the board) on the bus side of the project board.

To start the inverters, the DC power supply should be connected and turned on first, outputting a voltage within the range of acceptable starting voltages. There are large capacitors in the microinverters, so there will be a large charging current for a few seconds when the DC power is first applied. Once the DC input has been applied for approximately one minute, the status LEDs will blink green six times, and the microinverters can be connected to the microgrid bus. Since the microinverters do not support voltage independently, no synchronization is necessary before closing this relay. Upon detecting rated voltage on the output, the microinverters have a 5-minute commissioning period before they will begin to output power. At the end of the commissioning period, the LED indicators should begin to blink orange, signifying that they are outputting power but are not synced up to an optional monitoring system (the Enphase Envoy) that we did not purchase. At this point, the DC power supply should be outputting full current if the voltage limit is set sufficiently high. If the microinverters are not outputting current at this point, but the voltage on the DC supply is experiencing transients, it will help to connect a resistive load and increase the current limit of the power supply. For troubleshooting issues, the LED indicators on the back of the modules will output troubleshooting codes that can be looked up in the datasheet (see appendix for datasheet link).

3.2.2 DC Power Supply

For the DC power supply in this system, we purchased a Matsusada RE45-45-LUs1 power supply, which is capable of outputting 45 amperes at 45 volts. This model also comes with a USB control module, which allows us to command voltage and current output using the project computer. This power supply currently sits on the left side of the lab table on the load side of the project board. It is connected to the microinverter array under the table with 4-gauge (AWG) cables, which allow us to take full advantage of its 45-ampere output.

We have written a basic C++ program that communicates with the power supply and can cycle through a vector of output voltages and currents, with an arbitrary time delay between samples. This program allows us to control the power supply to behave like a simple model of a photovoltaic module. Using data from the National Renewable Energy Laboratory's (NREL) Oahu Solar Measurement Grid (National Renewable Energy Laboratory 2011), we can command realistic outputs from the DC power supply with as little as one second of delay between samples. The data from NREL's dataset is a measurement of solar irradiance, but with the relations between solar irradiance and voltage and current approximated as

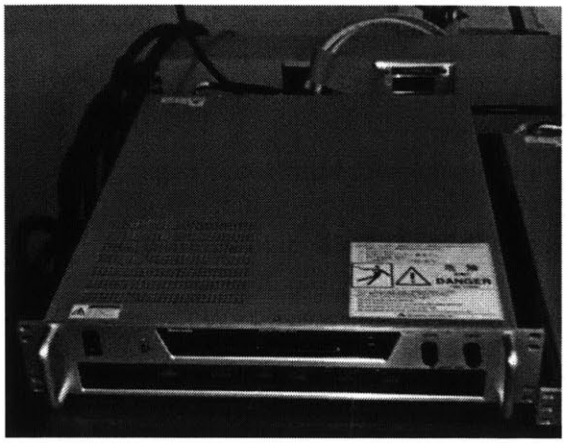


Figure 3-10 Matsusada DC Power Supply

$$I_{out} \propto \Phi \qquad \text{and} \qquad V_{out} \propto \ln(\Phi),$$

where Φ is the solar irradiance, it is simple to derive current and voltage pairs that fluctuate as a function of the solar irradiance.

3.2.3 Future Work

One of the biggest flaws in the current solar farm system is the inability of the microinverters to independently support an output voltage. In other words, our solar farm can only operate in conjunction with another generation source. This is less of a problem if we always have the utility connected or the diesel generator running, but it limits the ways in which the solar farm can contribute to the stability of the microgrid during transients. Future work could include replacing the current microinverter array with a set of inverters that can implement real and reactive power droop control or current injection control. This new system should be able to support voltage independently of the other generation sources and would potentially be able to greatly improve the stability of the microgrid during fault-induced islanding and other transients (Alaboudy, Zeineldin and Kirtley n.d.).

3.3 Wind Farm

At the time of writing, only the diesel generator and solar farm simulations have been developed to the point of operation; however, we have also made progress toward the development of a wind farm hardware simulation. The wind farm will emulate the electrical

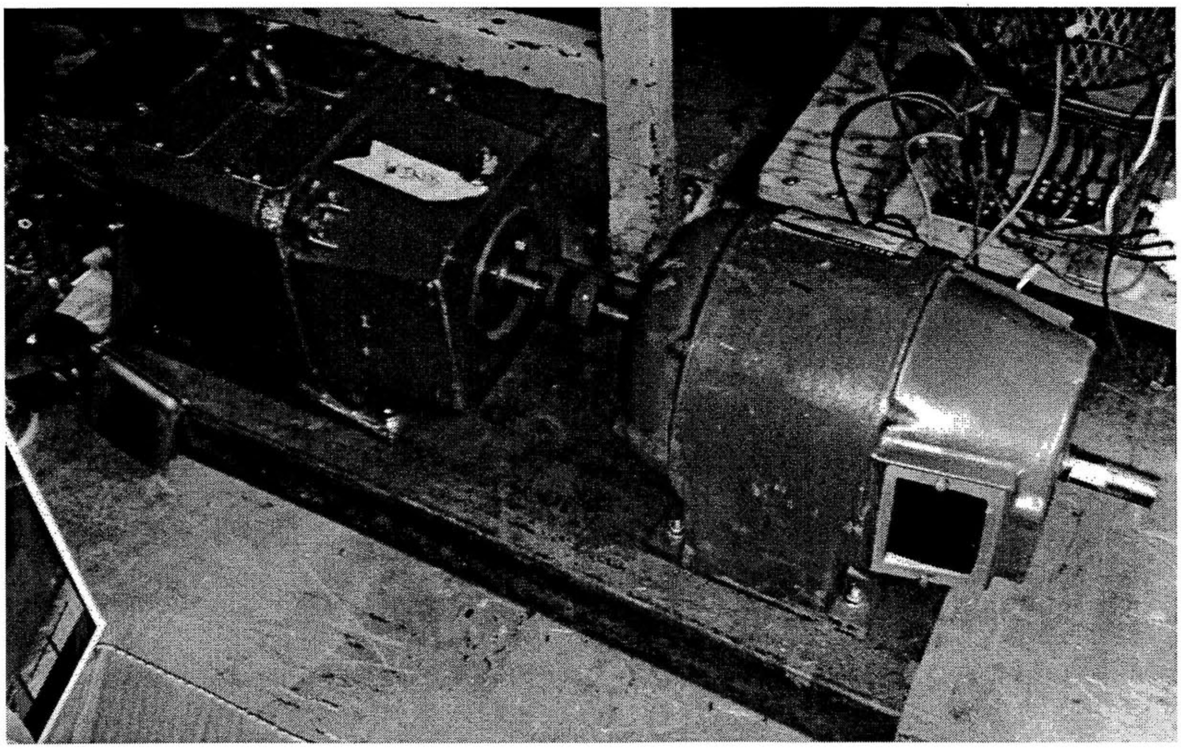


Figure 3-11 Wind Farm Motor Set (DC Motor on left, Induction Motor on right)

properties of a doubly fed induction generator driven by a wind turbine.

3.3.1 Hardware

The prime mover in this system, the wind turbine, will be modeled using a 1.5-horsepower DC motor identical to those motors used in the diesel generator described above and in the induction motor load described in the Loads section. This motor is mechanically coupled to a 1-horsepower wound rotor induction motor using the same Lovejoy jaw coupling described above, again with the 22-mm inner diameter bored out to fit the 7/8-inch shaft of the motors. The wound rotor induction motor will serve as the doubly fed induction generator thanks to the slip rings that provide access to the rotor windings. These two motors are mounted on a heavy steel base, and the motor set is currently stored under a desk in the corner of the project work area. Using the procedures outlined in IEEE Standard 112 and IEEE Standard 113, we found the parameters of the motors to be the following:

Table 1 - Wound Rotor Induction Motor and DC Motor Parameters

Induction Motor

X_1	4.0Ω
X_2	4.0Ω
X_m	69.3Ω
R_c	469.7Ω

DC Motor

Armature Resistance	0.7Ω
Armature Inductance	0.032H
Field Resistance	275Ω
Field Inductance	7.5H
Armature to Field Mutual Inductance	2.97H

3.3.2 Design and Future Work

To get the DC motor to emulate a wind turbine, we designed a simple buck converter circuit that would drive the armature of the DC motor at an arbitrary voltage. With a constant voltage on the field of the DC motor, we can control the shaft speed by adjusting the duty cycle of the buck converter with a simple microchip. Using wind data from the National Renewable Energy Laboratory, it would be straightforward to derive a set of desired duty cycles, which can then be stored on the microchip and passed to the buck converter at arbitrary time delays. In this way, we could control the shaft speed of the DC motor to emulate the mechanical characteristics of a wind turbine. We have purchased the components to build this circuit, including the microchip, but it has not yet been built.

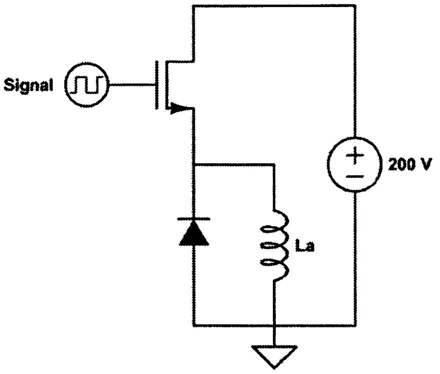


Figure 3-12 DC Motor Armature Buck Converter

On the generator side of the motor set, the present design calls for the construction of a back-to-back converter that can feed power to the rotor of the wound rotor induction generator when it is running below synchronous speed and can feed power from the rotor to the grid when it is running above synchronous speed. Controlling all twelve of these transistors

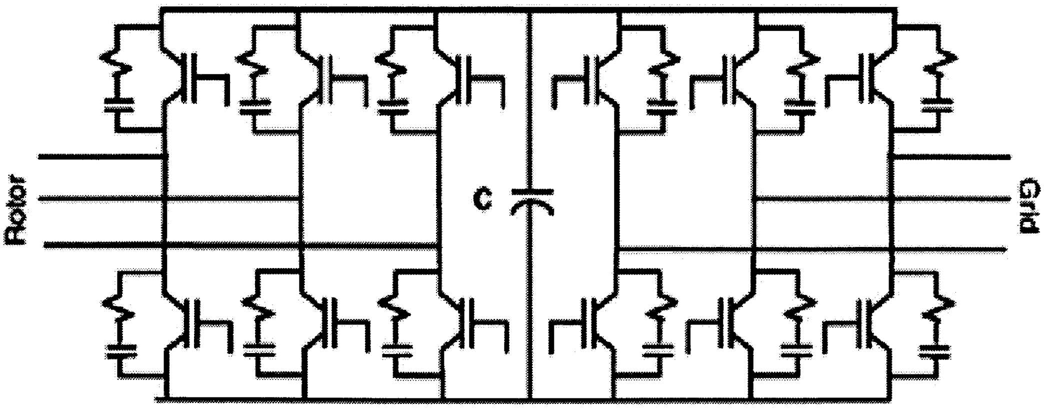


Figure 3-13 DFIG Back-to-Back Converter (Chowdhury and Chellapilla 2005)

is difficult and would have to be done with a digital signal processor, similar to the one used in the diesel generator controller. We derived an algorithm, based on a paper by B. H. Chowdhury (Chowdhury and Chellapilla 2005), to control the back-to-back converter, using an encoder and current sensors, as follows:

Rotor-side Vector Control:

1. Measure stator and rotor currents.
2. Convert these currents into the α-β reference frame.
3. Use the α-β currents to calculate the stator flux angle.
4. The active power set point and the available active power from the machine (i.e. the power being generated by the machine) are input to a P-I loop. The output is the q-axis rotor current.
5. The reactive power set point (set to zero) and the available reactive power from the machine (i.e. the reactive power being generated by the machine) are input to a P-I loop. The output is the d-axis rotor current.
6. The d-q axis currents are converted to the α-β reference frame using the stator flux angle calculated earlier.
7. The α-β currents are converted to a-b-c components.
8. The a-b-c components are used in a state vector modulation or hysteresis modulation scheme to generate gating signals for the rotor side converter.

Stator-side Vector Control:

1. Convert the grid side supply voltages into the α-β reference frame.
2. Calculate the supply angle.

3. Use the voltage across the DC link capacitor and its setpoint value to generate an error signal.
4. Feed the error signal to a P-I loop to get the d-axis current.
5. The q-axis current is forced to zero.
6. Convert the d-q axis currents to α-β frame using the voltage angle.
7. Convert the α-β currents to a-b-c frame.
8. Use state vector modulation or hysteresis modulation to generate gating signals for the supply side converter.

Successful implementation of this algorithm will produce a fully operational doubly fed induction generator. Many of the components required to implement this scheme have already been purchased and can be found in the project work area.

4. Loads

There are currently five different types of load in our system: resistive, capacitive, phase-controlled resistive, inductive, and induction motor. Our electromechanical relay switching system allows us to connect and disconnect the loads arbitrarily from the live microgrid bus, enabling rapid changes in load on the system. The individual loads are described in this section.

4.1 Resistive Load

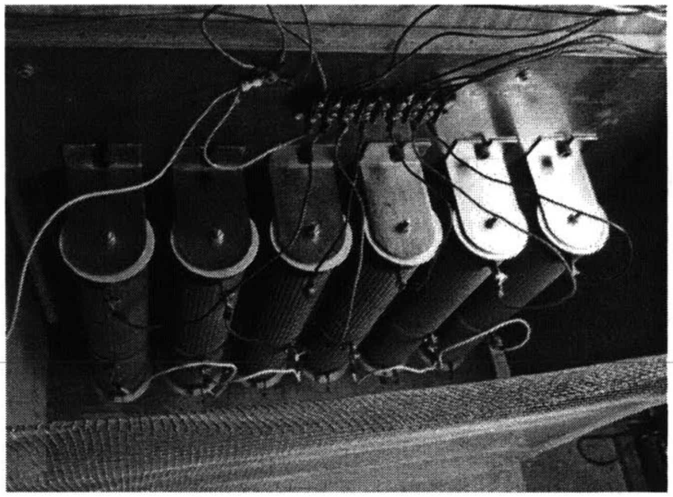


Figure 4-1 Resistive Load

There are currently two resistive loads in our system, consisting of six ceramic power resistors capable of high current and high temperature operation. The resistors are bolted to a large metal plate mounted at the top of the load side of the project board. The resistors and metal mounting plate are protected from accidental contact by two wooden side panels and a grounded wire screen. Each resistor has three connection points: one connection on each end and an adjustable band that allows the resistors to be tuned to an arbitrary resistance between 0 ohms and approximately 72 ohms. With these available connection points, it would be simple to connect the six resistors in any series or parallel combination of Y or delta configurations. There is a terminal strip at the top of the metal mounting plate that provides three-phase connections to the first and second electromechanical relays (counting from the top of the board) on the bus side of the project board, but there is currently no neutral wire connected to this terminal strip.

Each resistor is currently tuned to 48 ohms. At a nominal system voltage of 120 V_{RMS}, this tuning yields a pow.er dissipation of 300 watts per resistor or 900 watts per resistor array. With two resistor arrays, this allows us up to 1800 watts of resistive load on the system. If the resistors are tuned to higher power levels, care should be taken that the extra heat generated does not exceed the temperature rating of the wire insulation, or else the mounting board should be rotated so that the terminal strip and wires are below the resistive loads.

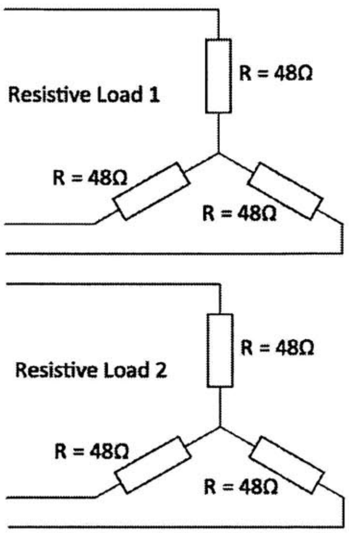


Figure 4-2 Resistive Load Diagram

4.2 Capacitive Load

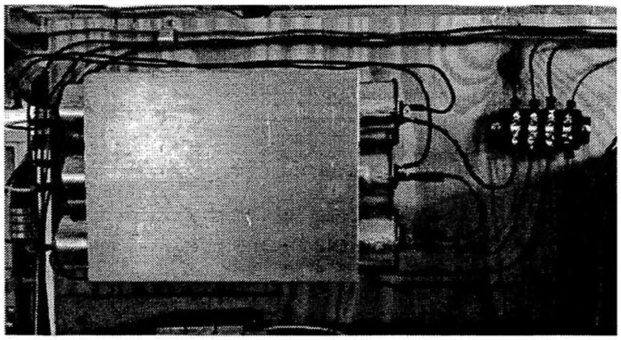


Figure 4-3 Capacitive Load

The capacitive load consists of six AC motor-start capacitors. The capacitors are mounted in an aluminum case at the top of the load side of the project board, to the left of the resistive load. There is currently no protective covering over this load as it is mounted high enough on the board to be out of the range of accidental contact. Each capacitor has two four-lead connection points, which we connected to the system using 0.25-inch quick connect/disconnect terminal connectors. With these available connection points, it would be simple to connect the six capacitors in any series or parallel combination of

Y or delta configurations. There is a terminal strip to the right of the capacitive load that provides access to the third electromechanical relay on the bus side of the project board, but there is currently no neutral wire connected to this terminal strip.

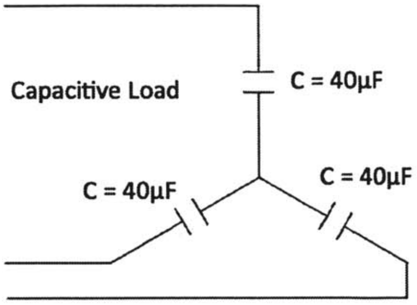


Figure 4-4 Capacitive Load Diagram

Each capacitor in the load is approximately 80µF. In their current configuration, the capacitors are connected in a Y arrangement with the neutral point unconnected. Each leg of this array consists of two capacitors in series, yielding an equivalent capacitance of 40µF per leg. At a nominal system voltage of 120 V_{RMS}, this configuration yields a reactive power of around 217 VAR per capacitor into the system or around 651 VAR into the system for the entire load. Relatively easy changes in connections can yield several different levels of load, but it is not possible to tune the power level of the capacitive load as finely as the resistive load; however, at the time of writing, there are several smaller motor-start capacitors in the project work area that could be added to the capacitive load to yield intermediate power levels.

4.3 Phase-controlled Resistive Load

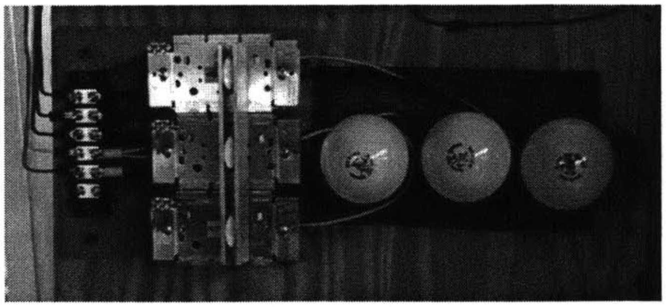


Figure 4-5 Phase-controlled Resistive Load

The phase-controlled resistive load consists of three light bulbs connected through three residential dimmer switches. This load is mounted at the top left of the load side of the project board, below the capacitive load and to the left of the resistive load. The dimmer switches are mounted on a separate board from the light bulbs so that the load connected to the dimmer switches can be changed in the future, if so desired. There is no protective covering over this load as most of the connections are completely enclosed and access to the dimmer switches is necessary if the power level is to be changed. We mounted the three dimmer switch boxes in parallel and constructed a metal bar connecting the three dimmer toggles, thereby allowing us to adjust the level of all three dimmer switches evenly. This bar is removable if an unbalanced load is desired. There is a terminal strip to the left of the load, mounted on the same board as the dimmer switches, which provides a connection to the fourth electromechanical relay on the bus side of the project board.

Operating at a nominal system voltage of 120 V_{RMS}, the light bulbs in this load are 60 watts each, providing a total resistive load of 180 watts. These light bulbs are connected in a Y configuration with the neutral point connected to the microgrid bus neutral. Each dimmer switch is connected in series with the live wire of its corresponding light bulb, and each dimmer is capable of full range dimming, from 0% to 100%.

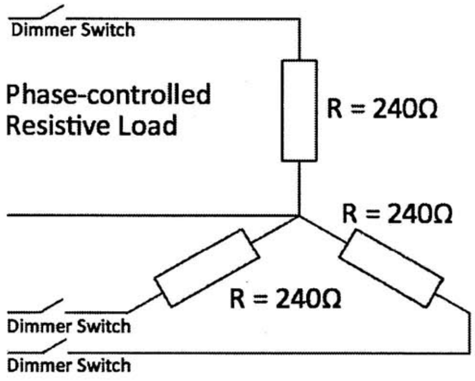


Figure 4-6 Phase-controlled Resistive Load Diagram

4.4 Inductive Load

The inductive load consists of three hand-wrapped inductors. These inductors are mounted on the top shelf on the load side of the project board, between the three-phase power outlet and the resistive load. We built a protective case for this load that consists of a three-sided wooden frame with the other three sides covered by a grounded wire screen. This allows the inductors to receive proper ventilation without the chance of accidental contact with electrical or thermal hazards. There is a terminal strip on the back of the protective case that provides a connection to the fifth electromechanical relay (counting from the top of the board) on the bus side of the project board.

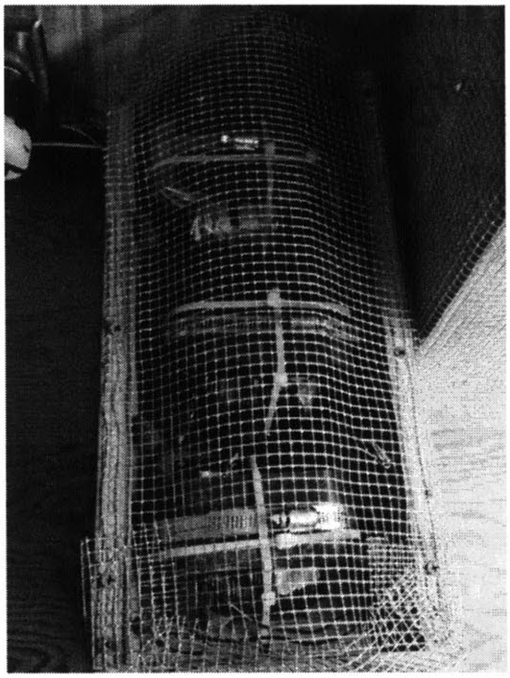


Figure 4-7 Inductive Load

The inductors were constructed using three split, laminated iron cores. Each inductor has a winding consisting of around 270 turns of 14-gauge magnetic wire wrapped around a rectangular plastic bobbin. Since we were working with a split core, we were able to place the wire-wrapped bobbin on one half of the core and then use appropriately sized pieces of FR-4 glass-epoxy laminate sheeting to maintain a constant gap size between the halves of the iron core. With the bobbin on the core and the FR-4 maintaining the gap size, we used worm-gear hose clamps to keep the two halves of the iron core pressed together. Constructed in this way, we were able to build three inductors of around 108mH each, which we connected in a Y configuration with the neutral point connected to the microgrid bus neutral. At a nominal system voltage of 120 V_{RMS}, this

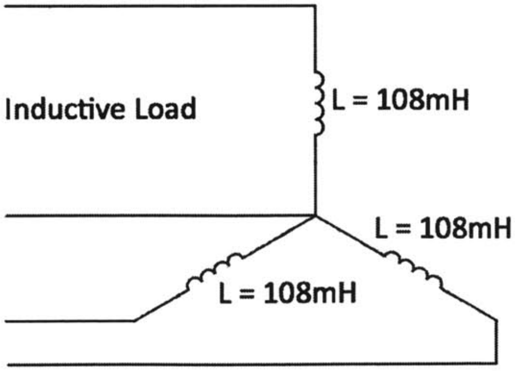


Figure 4-8 Inductive Load Diagram

configuration yields a reactive power of around 354 VAR per inductor out of the system or around 1061 VAR out of the system for the entire load. If smaller or larger load is desired, the gap size of each inductor can be adjusted to raise or lower the inductance within small bounds, or else more turns can be added to or removed from each winding.

4.5 Induction Motor Load

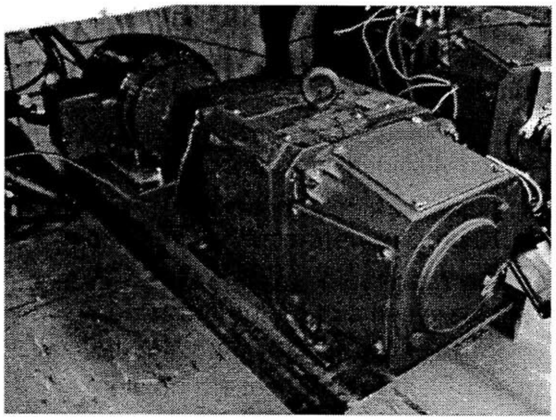


Figure 4-9 Squirrel-cage Induction Motor and DC Motor Set

The induction motor load in our system consists of a 0.75-horsepower squirrel-cage induction motor mechanically coupled to a 1.5-horsepower separately excited DC motor. The two motors are bolted to a heavy steel base under the table on the load side of the project board. To get the shafts to align for a proper coupling, we mounted the DC motor directly to the steel base but used aluminum shims to raise the smaller induction motor to the appropriate height. We coupled the shafts with the same Lovejoy jaw coupling described in the diesel generator section above, again boring out the 22mm inner diameter to fit the 7/8-inch motor shafts. The induction motor is electrically connected to the sixth electromechanical relay (counting from the top of the board) on the bus side of the project board. To have an adjustable load on the induction motor, we connected a power resistor across the armature of the DC motor and connected an adjustable DC power supply across the field of the DC motor. The power resistor is mounted inside a wood and wire screening enclosure to the left of the induction motor load, under the table on the load side of the project board. The DC power supply connected to the field of the DC motor currently sits on top of the table on the load side of the project board.

By adjusting the DC power supply connected to the field of the DC motor, we can achieve an adjustable mechanical load on the induction motor. This is possible because the power dissipated by the resistor on the armature of the DC motor, and by approximation, the mechanical power dissipated by the induction motor, is described by the equation:

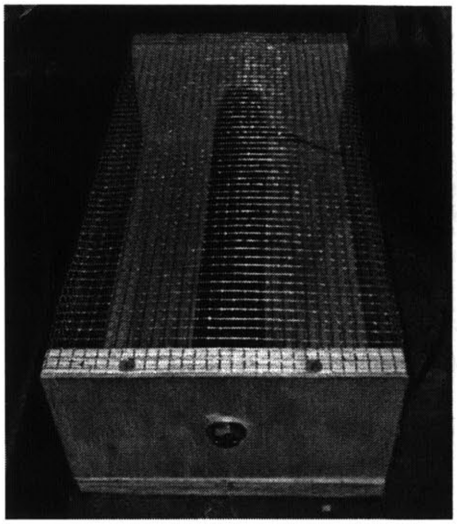


Figure 4-10 DC Motor Power Resistor

$$P_{diss} = \frac{E_a^2}{R_a}$$

where E_a is the generated voltage in the armature and R_a is approximately the resistance of the resistor connected to the armature, which is 50 ohms. The generated voltage is related to the field voltage by the equation:

$$E_a = \Omega \cdot M \cdot I_f = \Omega \cdot M \cdot \frac{V_f}{R_f}$$

for slowly varying field currents, I_f, where Ω is the mechanical speed of the shaft, M is the mutual inductance between the armature and field, V_f is the field voltage, and R_f is the field resistance. We can then describe the power dissipation as a function of field voltage with the equation:

$$P_{diss} = \frac{\left(\Omega \cdot M \cdot \frac{V_f}{R_f}\right)^2}{R_a}$$

Operating in this configuration, the no-load current (where V_f = 0) drawn by the induction motor is approximately 1.92 amperes at a nominal voltage of 208 $V_{RMS,line-line}$. Note that even with no voltage applied to the field of the DC motor, there may still be current flowing in the armature power resistor due to residual magnetism in the motor iron. The induction motor reaches its rated current of 3.4 amperes with an applied DC motor field voltage of approximately 185 volts. The power supply currently used for the field excitation is a Matsusada RE500-2.4-LUs1, which is capable of outputting 2.4 amperes at 500 volts, so care should be taken not to greatly exceed the ratings of the machines. This power supply also has a USB control module installed, which allows us to control the output of the supply using any desktop computer, if this is desired. We wrote a basic program in C++, saved on the project computer, which allows us to cycle through a vector of voltage values with an arbitrary delay between values, but this is not currently in use and the voltage is simply set by the knob on the power supply.

Using IEEE Standard 112, we found the parameters of the induction motor load to be the following, where the parameters in the table refer to the circuit elements in the accompanying schematic:

X_1	3.08Ω
X_2	3.08Ω
X_m	64.0Ω
R_1	2.65Ω
R_2	1.24Ω
R_c	362Ω

Figure 4-12 Induction Motor Parameters

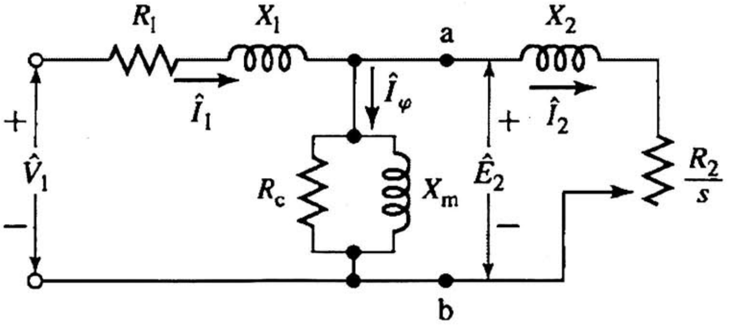


Figure 4-11 Squirrel-cage Induction Motor Single Phase Equivalent Circuit (Fitzgerald, Kingsley and Umans 2003)

Using IEEE Standard 113, we found the parameters of the DC motor to be the following:

Armature Resistance	0.7Ω
Armature Inductance	0.032H
Field Resistance	275Ω
Field Inductance	7.5H
Armature to Field Mutual Inductance	2.97H

The current, voltage, power, and speed ratings for the machines are clearly labeled on the machine nameplates.

5. Switching System

The switching system is an essential part of the project that allows us to dictate which specific loads and generators will be connected at any given time. We can also use the switching system to create an intentional fault on our system, as well as to connect and disconnect the microgrid to the central utility. To perform all of these operations, we used both magnetic contactor relays and solid-state relays.

5.1 Magnetic Contactor Relays

For the ability to individually switch on and off each of the loads, general-purpose magnetic contactor industrial relays were used. These relays are normally open and get switched on when a 24VDC supply sourcing 18mA is connected across the coil terminals. This on/off signal is supplied in our system by the white and yellow wires from the switch drivers. For a visual indication that the switch is powered on, a red LED is placed in series with a 24-kΩ resistor. When there is 24 volts across the coil terminals, the switch is closed and the red LED is illuminated. The relays are rated up to 25 amperes at full load and 30 amperes for a fully resistive load.

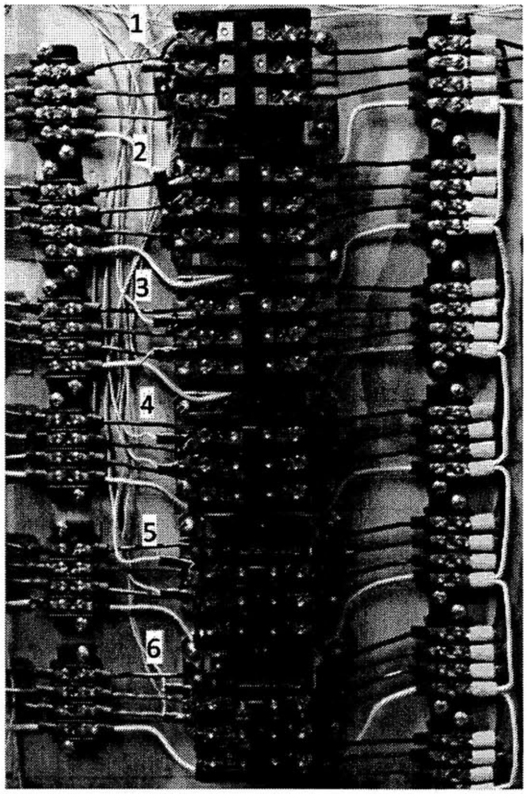


Figure 5-1 Load Relays

There are currently seven of these general-purpose industrial relays being used on the project, six of which are used to connect the microgrid bus to the various loads.

Relay #	Connects microgrid bus to:
1	Right Resistor Load
2	Left Resistor Load
3	Capacitor Load
4	Phase-Controlled Load
5	Inductor Load
6	Induction Motor Load

The seventh general-purpose industrial relay is being used as the faulting switch and has the utility on one side and the microgrid bus on the other side. This relay has a custom shorting bar (seen on the right of the switch in Figure 5-2) that can be placed across all three phases of the switch. When this switch is closed with the shorting bar in place, it will induce a three-phase line-to-line symmetric fault.

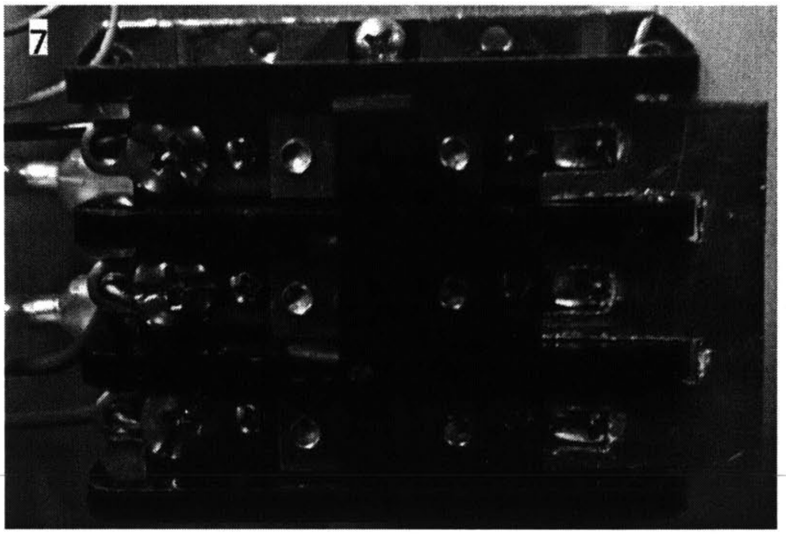


Figure 5-2 Shorting Relay

5.2 Solid State Relays

Along with the seven magnetic contactor relays, there are four solid-state relays being used on the microgrid. The solid-state relays are rated up to $480V_{RMS}$ and $55A_{RMS}$ with a built-in LED status indicator. The solid-state relays also have their on/off signal connected in our system by the white and yellow wires from the switch drivers and are switched on with 24 volts. The first three solid-state relays (pictured to the right) all switch on the various generators and connect them to the microgrid, as shown below:

Solid State Relay #	Connects microgrid to:
1	Solar Farm Generator
2	Wind Farm (in the future)
3	Diesel Generator

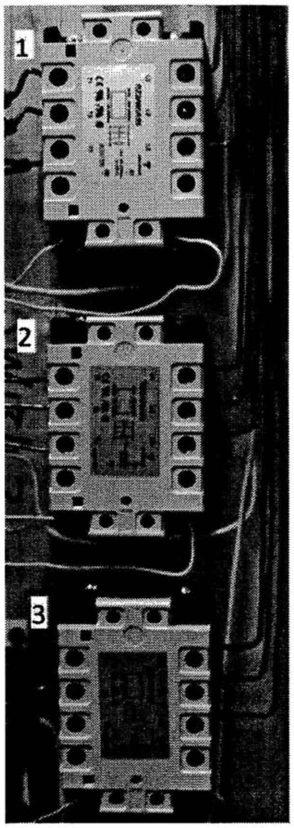


Figure 5-3 SSR Generator Relays

The fourth solid-state relay connects the three-phase microgrid bus to the three phases of the utility. This is the solid-state relay that will be in the open state when the microgrid is in the islanded condition. This is also the switch that needs to be closed at the proper time to synchronize the microgrid with the utility. As shown in Figure 5-4, there are two 12-watt 120VAC light bulbs in series across each phase, which will be illuminated with the switch open to indicate that the microgrid and utility are not synchronized.

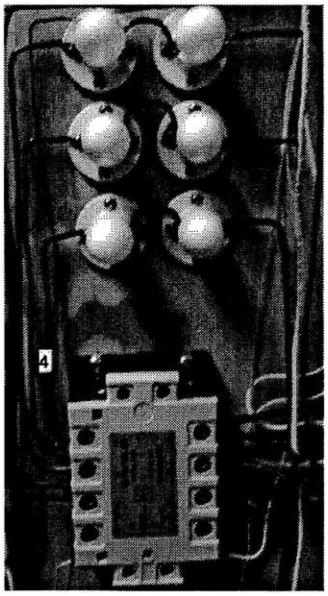


Figure 5-4 SSR Microgrid-Utility Relay

5.3 Driving Boards

In order to be able to control the on/off state of the relays from the computer, we used a National Instrument Multifunctional Data Acquisition System (DAQ). The NI USB-6008 has 12 TTL outputs, which we used to control the relays; however, the +5 volt (TTL logic) is not the 24 volts required to trigger on and off the relays. To remedy this, we built driving boards to properly amplify the 5 volts to the required 24 volts for the switching.

5.3.1 Magnetic Contactor Relay

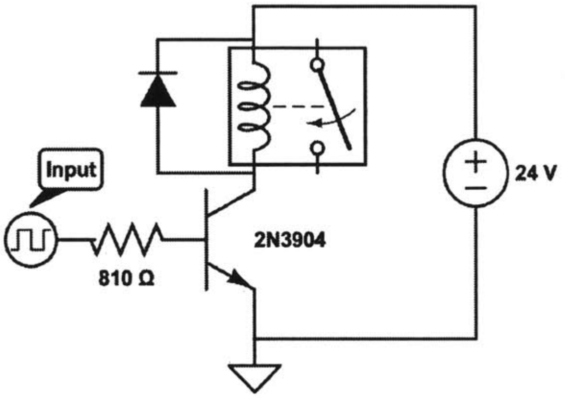


Figure 5-5 Circuit Diagram of Magnetic Contactor Relay Driving Board (1 of 7)

For the magnetic contactor relays, we designed a driving board with seven basic common emitter BJTs. For this circuit, as shown in Figure 5-5, with the BJT in cut-off, there is close to 0 volts across the coil terminals such that the relay is in the OFF state. When there is a 5 volt input from the DAQ, there is an 810-ohm resistor at the base of the BJT to provide current to turn on the device. With 5 volts being outputted from the DAQ, and an inherent voltage across the internal diode in the BJT between the base and the collector of around .7 volts, we expect $\frac{5\ volts - .7\ volts}{810\ ohms} \approx 5.3\ milliamperes$ which will turn the BJT on and make it approximately a short. With the switch in the ON state, there will be close to the full 24 volts across the coil terminals and it will thus be in the ON state as well. Note that a diode was placed in the reverse direction across the coil terminals to behave as a freewheeling diode in the case when the device is switched off. When switched off, the inductive coil cannot support a sudden change in current without imposing a large voltage because for an inductor, $V_L = L * \frac{di_L}{dt}$, so the freewheeling diode will help circulate current to minimize the strain on the BJT when switching.

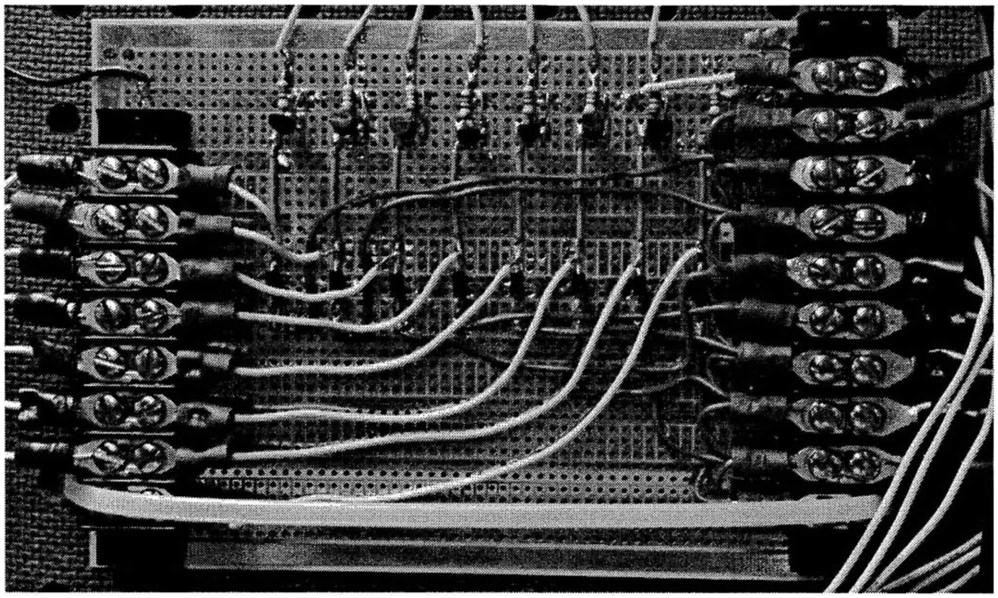


Figure 5-6 Magnetic Contactor Relay Driver Board

5.3.2 Solid State Relay

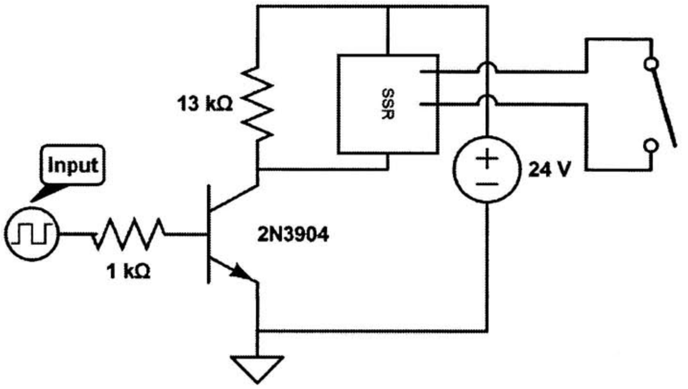


Figure 5-7 Circuit Diagram of the Solid State Relay Driver Board (1 of 4)

Like the magnetic contactor relay driving board, there are four standard common emitter BJTs used for the amplifier circuit on the driving boards for the solid-state relays (SSR). The predominant difference between the two boards is that the SSR input terminal has a small impedance but requires the entire 24 volts across it to turn on. If it were placed by itself in series with the BJT, there would be a voltage divider between its impedance and the impedance of the BJT that would prevent the full 24 volts from appearing across the SSR input terminals. Therefore, the SSR input terminals were put in parallel with a large resistor of 13000 ohms and now it has close to the full 24 volts across it when the BJT is in the ON state.

Additionally, the 13000-ohm resistor acts as a pull-up resistor to prevent the lower input terminal from floating in the BJT OFF state. For the BJT to be in an ON state, the DAQ outputs 5 volts, which when put across the 1000-ohm resistor to the .7 volts on the base of the BJT, results in a similar ≈ 5 milliamperes.

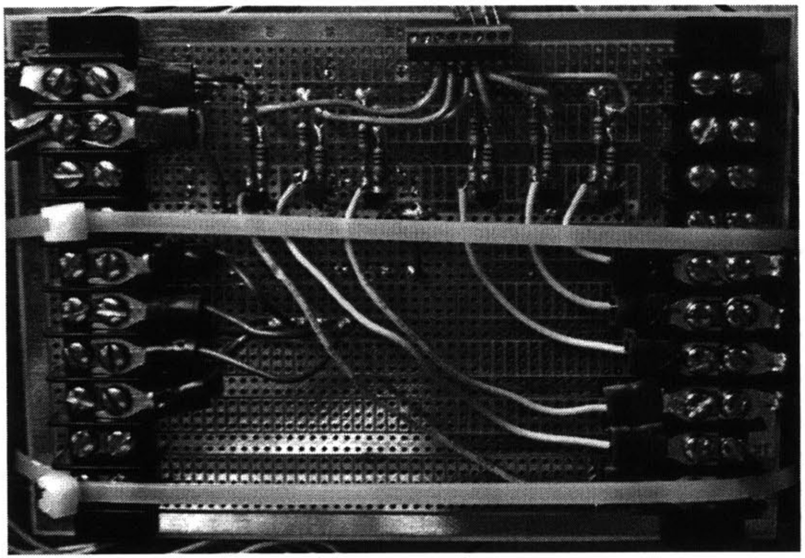


Figure 5-8 Solid State Relay Driver Board

5.4 Faulting System

One of the most important tests we want to perform on the microgrid is to see how the microgrid behaves during islanding conditions. The most important of these islanding conditions is when the microgrid is disconnected from the central utility because of a fault on the utility. Without an easy way to properly simulate a fault, we will actually impose a three-phase line-to-line symmetric fault on our system. It is important to note that during the fault, none of the lines are shorted to neutral or ground.

5.4.1 Faulting Bar

In order to induce a fault, we have constructed a faulting bar that is mounted onto the seventh mechanical contactor relay. This is just a piece of aluminum that was machined into an "E" shape (see Figure 5-9) to fit the terminals of the relay. While we are interested in measuring and observing the effects that this fault will have on the system, there are a few things that we expect to happen. Once the seventh mechanical contactor relay is engaged and the fault is induced, we expect the fourth solid-state relay to trip and become an open after a few cycles. Once this is opened, the microgrid will be in the islanded state. Similarly, it is possible, and probably likely, that the breakers for the three phases in the load center will trip after a few cycles and thus disconnect the fault from the central utility. However, before this fault is disconnected from the central utility, there will be a few cycles with very large current being drawn because of the fault. To help minimize this current, and hopefully prevent the entire lab's lights from dimming too much, we have added some current limiting inductors to the system.

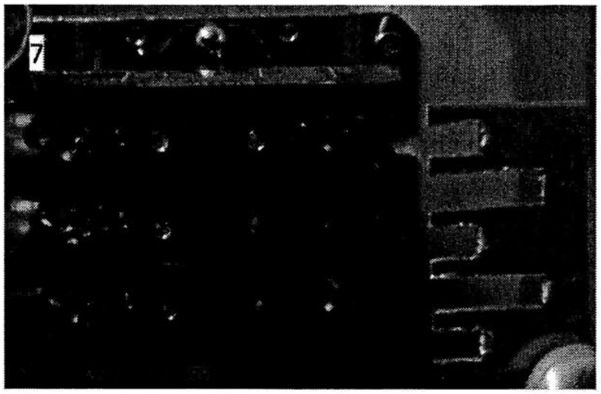


Figure 5-9 Shorting Bar

5.4.2 Current Limiting Inductors

An inductor was placed in series with each phase to provide an impedance for when the three phases are shorted together line-to-line. This impedance limits the current from the utility. Without this series impedance, there would be very little impedance limiting the utility from delivering excessive current until the breaker was flipped a few cycles later. The inductor fights the surge in current due to the fault to help protect the utility and it also does not dissipate much real power, mostly reactive power. These inductors were constructed using three split, laminated iron cores. Each inductor has a winding of 12-gauge magnetic wire wrapped around a rectangular plastic bobbin that acts as a shield between the wire and the core. This helps with heating and also ensures that the insulation around the magnetic wire is not scraped off from the edges of the inductor. The rectangular bobbin was first wrapped independently and, since we were working with a split core, we were able to place the wire-wrapped bobbin on one half of the core and then use appropriately sized pieces of FR-4 glass-epoxy laminate sheeting to maintain a constant gap size between the halves of the iron core. With the bobbin on the core and the FR-4 maintaining the gap size, we used worm-gear hose clamps to keep the two halves of the iron core pressed together. To size the inductors, we decided that we wanted it to have an impedance of approximately 5 ohms at 60 hertz. The 5 ohms implies that if the 120 volts are shorted across the inductor from a fault, there will be an absolute max of 24 amps flowing through it. $|Z| = 5\ ohms = |j\omega L|$. So to solve for the desired inductance:

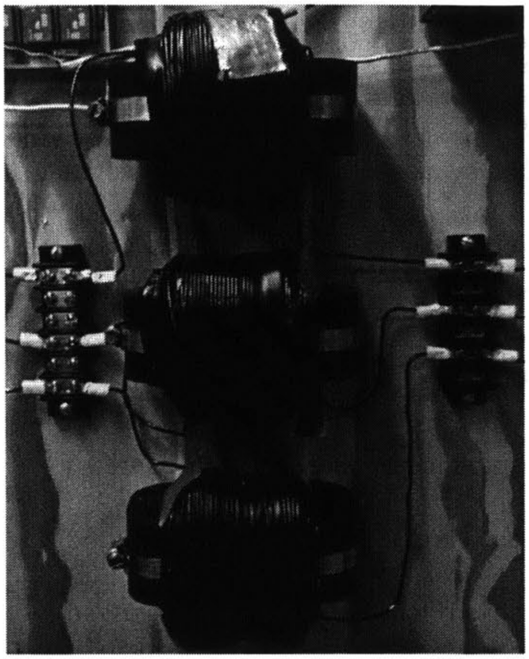


Figure 5-10 Shorting Inductors (all 3 phases)

$$L = \frac{5\ ohms}{2 * \pi * 60\ hertz} = .01326\ henries$$

To determine the required number of turns we would need for this inductance,

$$L = \frac{N * B_{sat} * Area}{I} \; so \; N = \frac{L * I}{B_{sat} * Area}.$$

Upon measuring the core, its cross sectional area was 1 inch by 2 inches and it was determined that we would use a conservative estimate for B$_{sat}$ = 1 Tesla.

$$N = \frac{.01326\; henries * 24\; Amps}{1\; Tesla * 2\; inches^2} \approx 247\; turns$$

Now, to solve for the appropriate gap size (g), we used Ampere's Law:

$$H * 2 * g = N * I$$

We were then able to solve for the appropriate gap size:

$$g = \frac{N * I}{2 * H} = \frac{247\; turns * 24\; Amperes}{2 * \frac{1\, Tesla}{\mu_o}} = 3.725\; mm$$

While we made these calculations, we left additional magnetic wire so that we could add a few turns to have our measurements meet the expected inductance. Additionally, we needed to add some extra FR-4 to increase the gap size. In the end, we used 270 turns and had a gap size around 1.25 cm. When measured on the LCR meter, our series inductors are now the desired 5 ohms. Note that because the inductors are in series with the loads, there is a voltage divider and therefore the loads (when connected) do not have the full 120 volts across them without generation sources supporting the microgrid bus voltage.

5.5 Synchronization

Our microgrid is designed to run both in parallel with the central utility as well as independent of it. A microgrid can be islanded (disconnected) from the central utility because a forced fault caused the breakers connecting the two systems to open or because it was intentionally islanded. Additionally, a microgrid can be intended to run independently of the

$$L = \frac{N*B_{sat}*Area}{I} \; so \; N = \frac{L*I}{B_{sat}*Area}.$$

Upon measuring the core, its cross sectional area was 1 inch by 2 inches and it was determined that we would use a conservative estimate for B_{sat} = 1 Tesla.

$$N = \frac{.01326 \; henries * 24 \; Amps}{1 \; Tesla * 2 \; inches^2} \approx 247 \; turns$$

Now, to solve for the appropriate gap size (g), we used Ampere's Law:

$$H * 2 * g = N * I$$

We were then able to solve for the appropriate gap size:

$$g = \frac{N * I}{2 * H} = \frac{247 \; turns * 24 \; Amperes}{2 * \frac{1 \, Tesla}{\mu_o}} = 3.725 \; mm$$

While we made these calculations, we left additional magnetic wire so that we could add a few turns to have our measurements meet the expected inductance. Additionally, we needed to add some extra FR-4 to increase the gap size. In the end, we used 270 turns and had a gap size around 1.25 cm. When measured on the LCR meter, our series inductors are now the desired 5 ohms. Note that because the inductors are in series with the loads, there is a voltage divider and therefore the loads (when connected) do not have the full 120 volts across them without generation sources supporting the microgrid bus voltage.

5.5 Synchronization

Our microgrid is designed to run both in parallel with the central utility as well as independent of it. A microgrid can be islanded (disconnected) from the central utility because a forced fault caused the breakers connecting the two systems to open or because it was intentionally islanded. Additionally, a microgrid can be intended to run independently of the central utility. Essentially, disconnecting a microgrid from the central utility is not a difficult problem to solve assuming that the microgrid can support all the loads independently and the voltage and frequency regulation of the system is sufficiently robust. However, the reconnecting of the microgrid back to the central utility can be a very difficult problem to solve. In order to properly reconnect, it is essential that the two systems have nearly identical frequencies in their waveforms and that the three microgrid phases not have a significant phase shift from the corresponding phase of the utility. Additionally, the phases must be rotating in the same direction. The two systems should have similar voltages although it is possible the stronger utility can pull up a drooping microgrid voltage. When these conditions are met, the microgrid and the utility can be synchronized.

5.5.1 Synchronizing Light Bulbs

For our system, we hope to eventually be able to synchronize the microgrid with the utility through the LabVIEW interface. However, we have established a visual method for ensuring that the two systems are properly synchronized before they are reconnected together. Two standard 120-VAC, 7-watt light bulbs were connected in series across the fourth solid-state relay that connects the microgrid with the utility. Two light bulbs were needed instead of one because if the two phases in each system are phase shifted by 180 degrees, there will be a maximum of 240V_{RMS} across the bulbs. The light bulbs will be illuminated if there is a voltage across

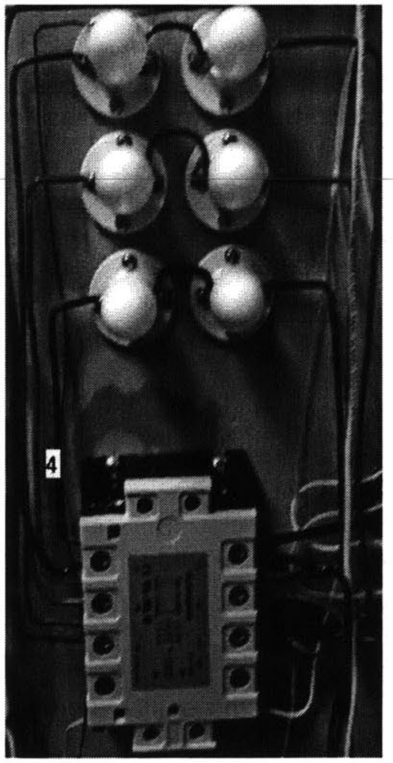


Figure 5-11 Synchronizing Light Bulbs

their terminals: the brighter the bulbs, the higher the voltage. If there is a voltage difference between the microgrid and the utility for a given phase, the light bulbs for that phase will be illuminated thus implying that the system cannot be synchronized. All of the light bulbs must be off, to insure that the system is synchronized, before it is safe to reconnect the microgrid with the central utility.

6. Monitoring System

The monitoring system is an essential part of the project as it allows us to capture data on important variables from experiments for real time visualization. The monitoring system also stores the data, which can later be analyzed. The monitoring system is made up of various National Instruments Data Acquisition (DAQ) devices as well as custom built voltage and current measurement boards.

6.1 DAQs

Currently, there are three DAQs being used on the project, two of the NI USB-6211 variety and one NI USB-6008. The NI USB-6211 has 16 analog inputs, 4 digital inputs, 2 analog outputs and 4 digital outputs each with 16 bits of resolution. The NI USB-6008 has 8 analog inputs, 2 analog outputs and 12 interchangeable digital I/O each with 12 bits of resolution. There will be 6 analog inputs from each measurement board and a digital output for each relay. In the end, just for these applications, we will use a total of 30 analog inputs and 11 digital outputs and therefore need a few DAQs for the microgrid. DAQs 1 and 2 (from top to bottom) connect the measurement boards and their pin-outs can be seen in the measurement boards section. The pin-out for DAQ 3 (the NI USB-6008), which controls the switches, is as follows:

DAQ #3 (Relay Control DAQ)	Wire Color	Connection	I/O	Type
P0.0	Green	SSR1 Driver Board BJT Base	Output	Digital
P0.1	Green	SSR2 Driver Board BJT Base	Output	Digital
P0.2	Green	SSR3 Driver Board BJT Base	Output	Digital
P0.3	Green	SSR4 Driver Board BJT Base	Output	Digital
P0.4	Yellow	MCR1 Driver Board BJT Base	Output	Digital
P0.5	Yellow	MCR2 Driver Board BJT Base	Output	Digital
P0.6	Yellow	MCR3 Driver Board BJT Base	Output	Digital
P0.7	Yellow	MCR4 Driver Board BJT Base	Output	Digital
P1.0	Yellow	MCR5 Driver Board BJT Base	Output	Digital
P1.1	Yellow	MCR6 Driver Board BJT Base	Output	Digital
P1.2	Yellow	MCR7 Driver Board BJT Base	Output	Digital
GND	Green	Ground on Driver Board	Output	Ground

6.2 Measurement Boards

The measurement boards were designed by one of our group members, Jorge Elizondo, to allow for an integrated measurement and analysis of main performance variables such as current and voltage of the system at various targeted nodes. The boards themselves consist of three (one for each phase) Pulse Electronic Corporation Transformers, which take a primary voltage up to 230VAC and output a secondary voltage up to 6 VAC with an approximate turns ratio of 38. These transformers have a large input impedance so that they have a minimal impact on the system they are monitoring. The intention is to get the high line voltage into a range for which it

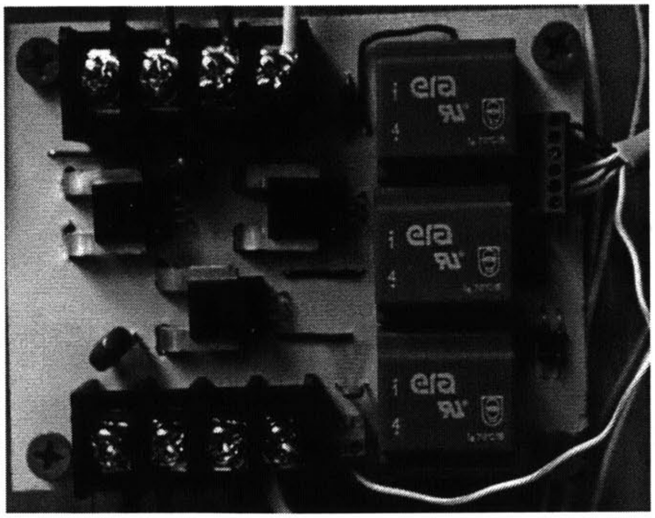


Figure 6-1 Beta Prototype of Measurement Board

is within the tolerance of the input range of the DAQs. There are also three Allegro Current Sensors, which measure the current of each phase of the system and output an analog voltage to the DAQs.

There are fuses on the neutral of the primary side of each transformer to help protect the transformer in the case of a current spike. Additionally, there are EPCOS Inc. Varistors, which will protect the transformers if there is a large spike in line to neutral voltage on any of the phases. At the time of writing, there are currently four measurement boards on the microgrid, with another being built:

Measurement Board #	Measurement Node
1	Diesel Generator Emulator
2	Solar Farm Emulator
3	Utility main grid
4	Microgrid Bus
5*	Wind Turbine Emulator (not shown)

Note that the measurement board on the Wind Turbine Emulator will be added in the future.

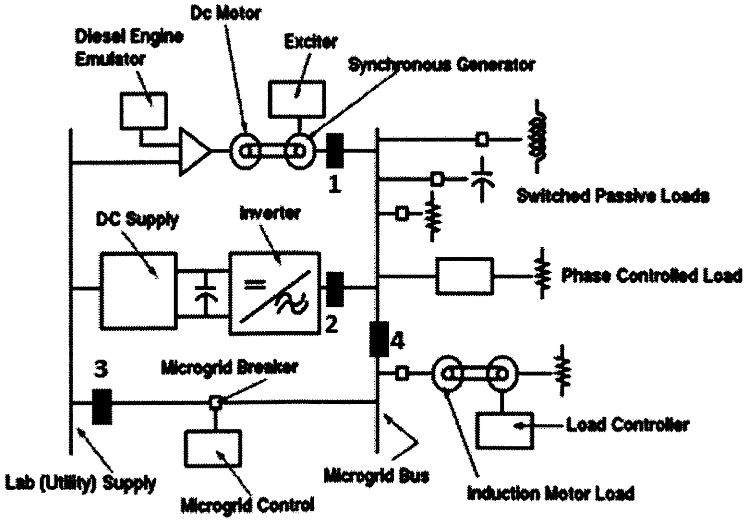


Figure 6-2 Measurement Board Locations

Measurement Board 1 (Diesel Generator)	**Wire Color**	**Connection**	**I/O**	**Type**
V1	Blue	AI0 (DAQ 2)	Output	Analog
V2	Brown	AI1 (DAQ 2)	Output	Analog
V3	Green	AI2 (DAQ 2)	Output	Analog
I1	White/Blue	AI3 (DAQ 2)	Output	Analog
I2	White/Brown	AI4 (DAQ 2)	Output	Analog
I3	White/Green	AI5 (DAQ 2)	Output	Analog
+5V	Red	Supply +5 V	Input	Power
GND	White/Orange	GND (DAQ 2) and Supply V-	Input	Ground

Measurement Board 2 (Solar Farm Emulator)	**Wire Color**	**Connection**	**I/O**	**Type**
V1	Blue	AI6 (DAQ 2)	Output	Analog
V2	Brown	AI7 (DAQ 2)	Output	Analog
V3	Green	AI8 (DAQ 2)	Output	Analog
I1	White/Blue	AI9 (DAQ 2)	Output	Analog
I2	White/Brown	AI10 (DAQ 2)	Output	Analog
I3	White/Green	AI11 (DAQ 2)	Output	Analog
+5V	Red	Supply +5 V	Input	Power
GND	White/Orange	GND (DAQ 2) and Supply V-	Input	Ground

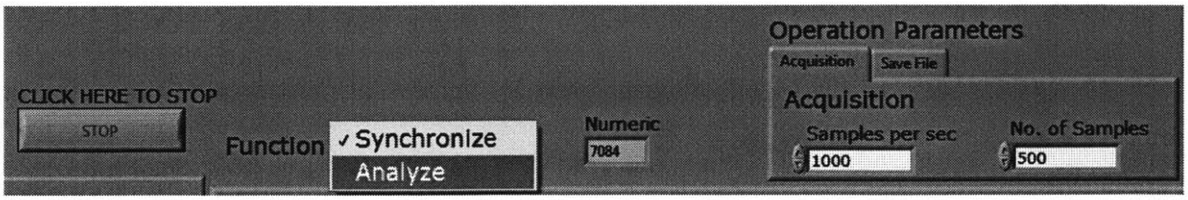
Measurement Board 3 (Utility)	**Wire Color**	**Connection**	**I/O**	**Type**
V1	Blue	AI0 (DAQ 1)	Output	Analog
V2	Brown	AI1 (DAQ 1)	Output	Analog
V3	Green	AI2 (DAQ 1)	Output	Analog
I1	White/Blue	AI3 (DAQ 1)	Output	Analog
I2	White/Brown	AI4 (DAQ 1)	Output	Analog
I3	White/Green	AI5 (DAQ 1)	Output	Analog
+5V	Red	Supply +5 V	Input	Power
GND	White/Orange	GND (DAQ 1) and Supply V-	Input	Ground

Measurement Board 4 (Microgrid Bus)	**Wire Color**	**Connection**	**I/O**	**Type**
V1	Blue	AI6 (DAQ 1)	Output	Analog
V2	Brown	AI7 (DAQ 1)	Output	Analog
V3	Green	AI8 (DAQ 1)	Output	Analog
I1	White/Blue	AI9 (DAQ 1)	Output	Analog
I2	White/Brown	AI10 (DAQ 1)	Output	Analog
I3	White/Green	AI11 (DAQ 1)	Output	Analog
+5V	Red	Supply +5 V	Input	Power
GND	White/Orange	GND (DAQ 1) and Supply V-	Input	Ground

Note that for each of the measurement boards, the 6 V/I pins are in order from the top of the board to the bottom and the ground pin is to the right of the +5V pin when the board is properly oriented.

6.3 LabVIEW

LabVIEW is installed on the project computer and is the overall monitoring and control software that is used for the microgrid project. Within the current LabVIEW program (version 3.1), there is a page for analyzing the system and one for synchronizing the system. These two pages make up the user interface that is used to control the microgrid. Throughout the program, there is a constant top bar that helps the user navigate between pages and make common selections. The farthest left button is the STOP button that acts as the software emergency stop. When pressed, it will turn off all measurement boards and open all relays. To



Function ✓ Synchronize

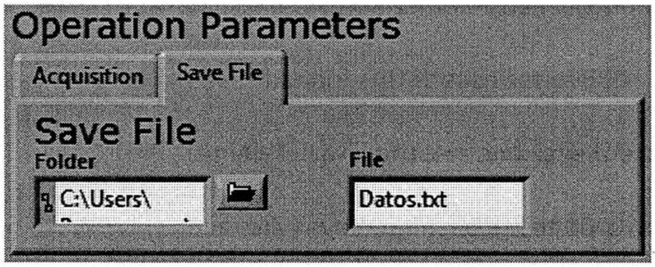
Numeric 7084

Samples per sec 1000

No. of Samples 500

Figure 6-3 LabVIEW Top Control Bar

switch between each of these pages, there is a "Function" tab in the middle of this top bar that, when clicked, shows a drop down menu to select the desired screen. The screen that is checked represents the current screen. There is also the "Numeric" box that acts as a counter for the number of loops the system has performed through the software. Finally, to the right of the top control bar is the Operation Parameters. Within this menu and under the "Acquisition" tab, the user can set the number of Samples per second that each DAQ will collect per channel. Additionally, the user can set the Number of Samples per file. In the current example, the DAQ will collect 1000 samples per second from each channel. After each set of 500 samples, it will process and save the data. Basically, every half-second at 1000 samples per second the DAQ will save and process the data. Under the "Save File" tab, the user can select the location of the file and what to name it as can be seen in the figure. The default is to save the data as a text file.



Folder
C:\Users\

File
Datos.txt

Figure 6-4 LabVIEW "Save File" Tab

In the main section of the screen below the control tab, regardless of whether you are on the "Synchronize" or "Analyze" page, the left of the screen is always occupied by the "SWITCH CONTROL" display. As can be seen from the figure, each relay in the system is represented as a square green button. When clicked by the user, the green button will light up indicating that the relay is closed (in the ON state). Additionally, there is also a circular green button for each measurement board that is illuminated when the measurement board is clicked active. There is also a "Clear all" and "Select all" button that will turn on and off all of the measurement boards. Because the computer has a hard time processing all of the measurement board data at once, it works best if only the important measurement boards are active during the appropriate test.

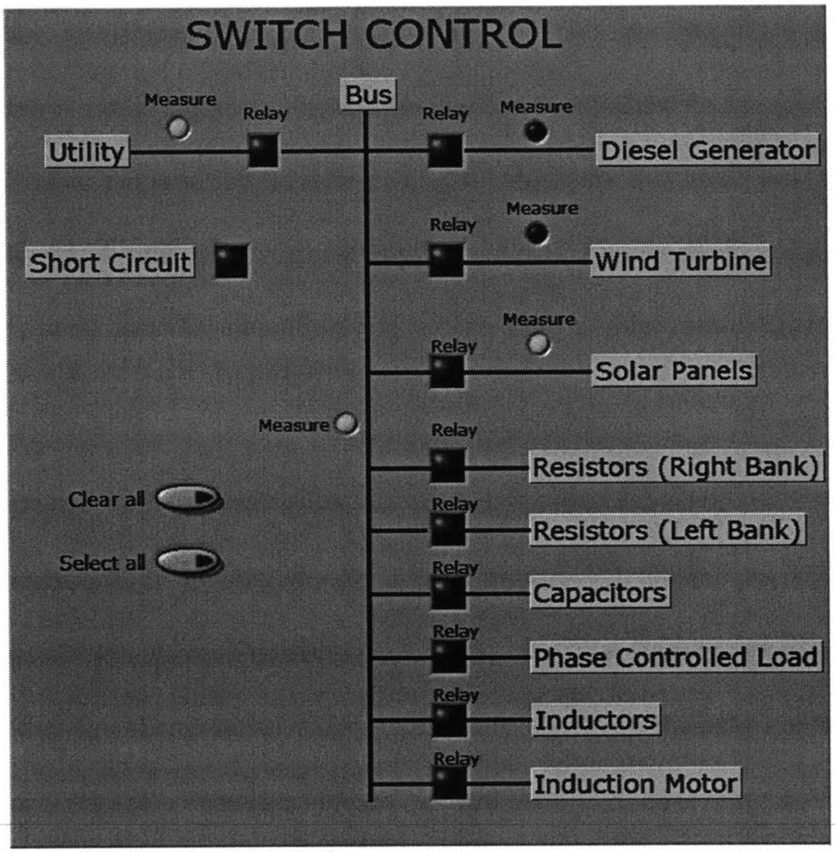
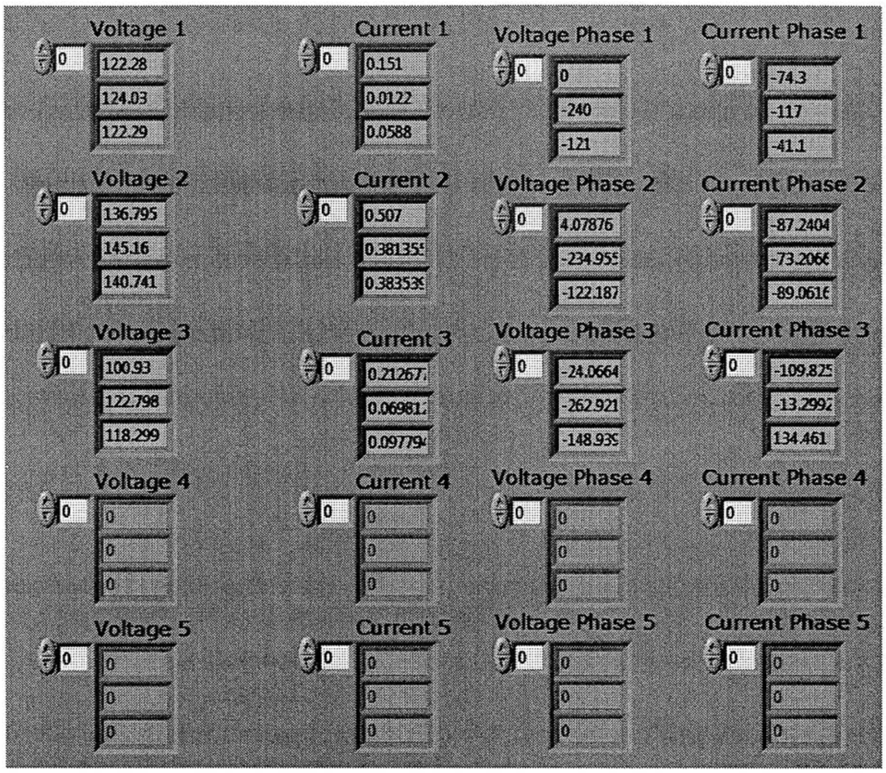


Figure 6-5 LabVIEW Switch Control

The second section of the "Analyze" screen on the LabVIEW program is the measurements section. This section lists all measurements from each measurement board (five total) as a column for each of the voltages (RMS), currents (RMS), voltage phase angles and current phase angles. Within each measurement criterion for each measurement board, there are three values displayed representing the measurement for each phase.



Voltage 1	Current 1	Voltage Phase 1	Current Phase 1
0	0	0	0
122.28	0.151	0	-74.3
124.03	0.0122	-240	-117
122.29	0.0588	-121	-41.1

Voltage 2	Current 2	Voltage Phase 2	Current Phase 2
0	0	0	0
136.795	0.507	4.07876	-87.2404
145.16	0.38135	-234.955	-73.2066
140.741	0.38353	-122.187	-89.061

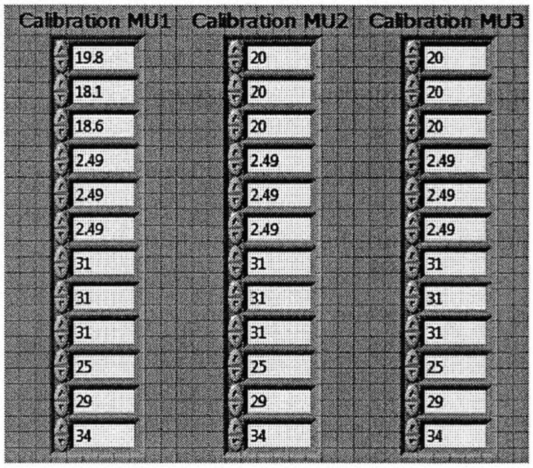
Voltage 3	Current 3	Voltage Phase 3	Current Phase 3
0	0	0	0
100.93	0.21267	-24.0664	-109.82
122.798	0.06981	-262.921	-13.299
118.299	0.09779	-148.93	134.461

Voltage 4	Current 4	Voltage Phase 4	Current Phase 4
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Voltage 5	Current 5	Voltage Phase 5	Current Phase 5
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Figure 6-6 LabVIEW Measurements Section

For now, the third section of the "Analyze" page (note it is also the third section of the "Synchronize" page) is the calibration section. This section is more of the back end adjustments that the user can make to change the gains on each measurement that the measurement boards make. This should hopefully be done automatically in the future to calibrate all the sensors and should never need to be adjusted manually.



Calibration MU1	Calibration MU2	Calibration MU3
19.8	20	20
18.1	20	20
18.6	20	20
2.49	2.49	2.49
2.49	2.49	2.49
2.49	2.49	2.49
31	31	31
31	31	31
31	31	31
25	25	25
29	29	29
34	34	34

Figure 6-7 LabVIEW Calibration Section

The differentiating section in the

"Synchronize" page is the middle section currently called "Micro-grid points to be synchronized". This section requires you to enable two and only two of the measurement boards from section one and it will clearly display the voltages of each measurement point. In most cases, the utility will be used as the reference point (Point 1) and the microgrid bus as Point 2. This interface shows the measured frequency and plots a phasor diagram representation of the phase A of each measurement point. If the two points are synchronized, the two blue arrows in the diagram (the lines are not shown in the figure) will overlap.

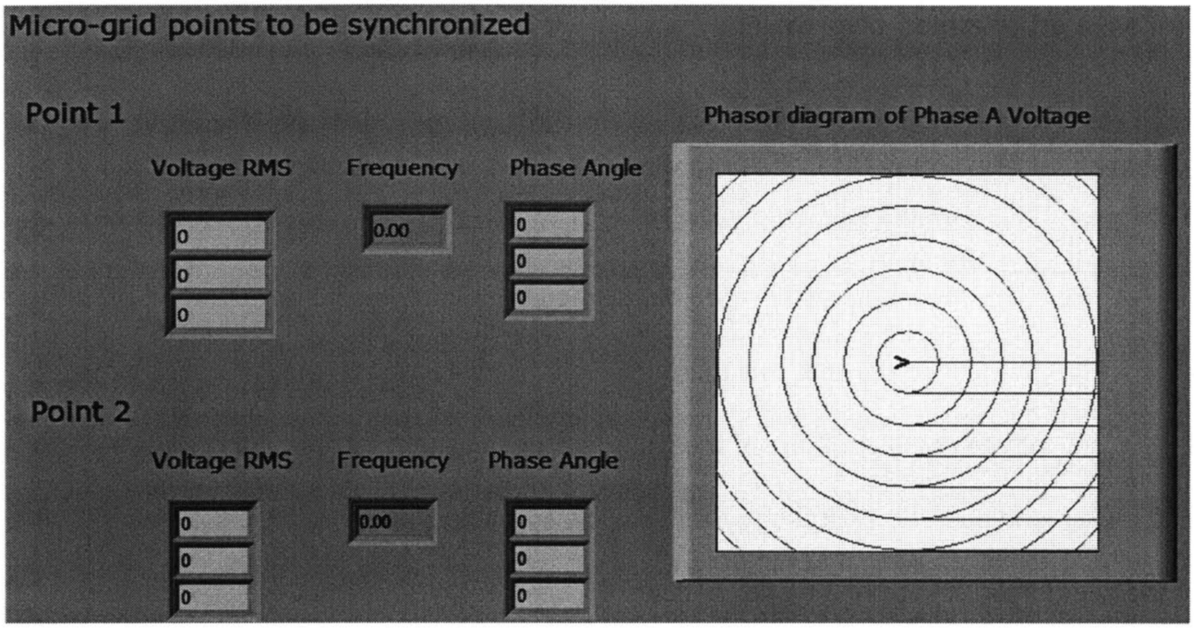


Figure 6-8 LabView Synchronize Page

7. Future Work

This section discusses future work that could improve the present capabilities of the project or implement new systems that expand on current capabilities. The system could be considered complete without the following systems, but these suggestions would all be interesting projects for the system.

7.1 Diesel Generator Excitation System

The present excitation system used on the diesel generator model, as described above, is a rather simple PI controller that sets the voltage on the field based on the difference between the output voltage of the armature of the wound-rotor induction machine and the line voltage of the microgrid bus. This excitation system has the advantage of being simple and easy to implement, but it is far from the optimal control scheme for the system. Developing a more complex excitation system that relies on more than just the output voltage of the generator would help make the system more stable during faults, fault-induced islanding, increases or decreases in load, and other system transients. The dsPIC used to control the DC motor is currently being used well below its full potential and could be very useful in implementing a more complex control scheme.

7.2 Solar Farm Inverter Array

The Enphase microinverters currently used in our system have the advantage of being modular and easy to use; however, they have the major disadvantage of being unable to support voltage independently of other generation sources. The solar farm system could be improved by replacing these inverters with a six-pack of MOSFETs or IGBTs with associated gate drivers and output filters that could be controlled in a more flexible way than the Enphase microinverters. Such a configuration could implement P-Q droop control or current injection control that would allow the group to study the impact of inverter-based distributed generation on system transients, like faults and fault-induced islanding.

Another improvement in the solar farm system involves changes to the control of the DC power supply. In its present form, the program on the project computer that controls the DC power supply outputs voltage commands that change after a set time interval. A better implementation of this program would give current commands to the power supply instead and would be a more accurate emulation of the electrical characteristics of a photovoltaic cell as it would allow the Enphase inverters to use their built-in maximum power point tracking capabilities.

7.3 Wind Farm

As described in a previous section, some progress has already been made toward constructing a wind farm simulation. This design uses current and voltage measurements as well as an encoder to measure rotor speed and angle. While we believe it would be best to pursue this design and see it to completion before any other design is attempted, a similar design was proposed by a visiting scientist working in our group. This design, which can be found on the project computer, forgoes the use of an encoder and instead uses only current and voltage measurements to implement a doubly fed induction generator. It would be an interesting addition to the project if this second design could also be implemented and the results compared with the original design.

7.4 Induction Motor Load Supply Control

As noted above, the voltage on the field of the DC motor in the induction motor load system is manually controlled by the voltage knob on the supply. However, this power supply is USB controllable, so it would be possible to adapt our simple C++ program on the project computer to implement a more advanced control algorithm that could implement arbitrary load profiles. In this way, it would be possible to implement an induction motor load with periodically varying load that behaves similarly to a refrigerator, on the small scale, or industrial machinery, on a larger scale.

7.5 Microgrid Central Controller

At present, the diesel generator system sets the voltage and frequency of the microgrid bus without any input from the utility bus or other generation sources. After the completion of the wind farm generator and any upgrade of the solar array, it would be a beneficial and interesting project to implement a central control system for the microgrid bus that better coordinates the generation sources with the utility bus. Such a system has the potential to improve the stability of the microgrid during transients like faults and fault-induced islanding.

7.6 Load-shedding Scheme

As described previously, the present system for controlling the load switches is fairly simple. The switch driving board is controlled by the project computer through a DAC, and our LabVIEW interface consists of a few buttons that are either on or off. As a result, a load is either connected to the system or it is not, and switching the state of any particular load must be done by manually clicking the corresponding button. It would be an interesting addition to the project to implement a load-shedding scheme in LabVIEW that could selectively disconnect and reconnect loads during and after faults and other transients. Nearly all of the required hardware is already in place for this system, and our current LabVIEW control interface would be ideal for implementing a load-shedding algorithm.

Bibliography

Alaboudy, A. H. Kasem, H. H. Zeineldin, and J. Kirtley. "A Simple Control Strategy for Inverter Based DG to Enhance Microgrid Stability in the Presence of Induction Motor Loads." Masdar Institute.

Chowdhury, Badrul H., and Srinivas Chellapilla. "Double-Fed Induction Generator Control for Variable Speed Wind Power Generation." *Electric Power Systems Research*, no. 76 (2005): 786-800.

Fitzgerald, A. E., Charles Kingsley, and Stephen D. Umans. *Electric Machinery.* 6th Edition. New York: McGraw-Hill, 2003.

IEEE Power Engineering Society. *Standard 112 - IEEE Standard Test Procedure for Polyphase Induction Motors and Generators.* The Institute of Electrical and Electronics Engineers, Inc., 2004.

IEEE Power Engineering Society. *Standard 113 - IEEE Guide: Test Procedures for Direct-Current Machines.* The Institute of Electrical and Electronics Engineers, Inc., 1985.

National Renewable Energy Laboratory. "Oahu Solar Measurement Grid." *NREL.* 2011. http://www.nrel.gov/midc/oahu_archive/ (accessed March 29, 2012).

Appendix A – Part Numbers and Datasheets

This appendix lists useful information about the specific hardware used in each section above and lists links to datasheets and other useful documents.

A.1 System Overview

	Part	**Manufacturer**	**Part Number**
	Load Center	Square D	QO320L125GRB
Datasheet	http://products.schneider-electric.us/products-services/product-detail/?event=datasheet&partnumber=QO320L125GRB&countrycode=us		
	Breakers	Square D	QO310
Datasheet	http://products.schneider-electric.us/products-services/product-detail/?event=datasheet&partnumber=QO310&countrycode=us		
	Emergency Shutoff Switch	Square D	D-321N
Datasheet	http://products.schneider-electric.us/products-services/product-detail/?event=datasheet&partnumber=D321N&countrycode=us		
	Desktop Computer	HP	505B
Datasheet	http://h10010.www1.hp.com/wwpc/pscmisc/vac/us/product_pdfs/HP_505B_BusinessPC_Datasheet_Sept2010.pdf		
	Voltage Regulator	APC	Pro 700
Datasheet	http://www.apcmedia.com/salestools/DFAH-8GLRDV_R0_EN.pdf		

A.2 Generators

A.2.1 Diesel Generator

	Part	**Manufacturer**	**Part Number**
	DC Motor	Reliance	DC0189ATY
Datasheet	http://www.reliance.com/prodserv/motgen/motinfcnt.htm		
	Induction Motor	Reuland	7660466-1
Datasheet	http://reuland.com/Brochures/Wound%20Rotor%20Motor.pdf		
	Current Sensor	Allegro MicroSystems	ACS712ELC-05B-T
Datasheet	http://www.allegromicro.com/en/Products/Part_Numbers/0712/0712.pdf		
	Encoder	Red Lion	ZPJ2500
Datasheet	http://www.redlion.net/Products/Groups/Thru-BoreEncoders/ZPJ/Docs/11033.pdf		

	Part	Manufacturer	Part Number
	Encoder Bore Sleeve	Red Lion	RPGBIM01
Datasheet	http://www.redlion.net/Products/Groups/Thru-BoreEncoders/ZPJ/Docs/11033.pdf		
	Digital Signal Processor	Microchip	dsPICDEM MC1
Datasheet	http://ww1.microchip.com/downloads/en/DeviceDoc/70119E.pdf		
	Coupling	Lovejoy	L075-22mm
Datasheet	http://www.lovejoy-inc.com/products/jaw-type-couplings.aspx		
	Coupling Insert	Lovejoy	68514410621
Datasheet	http://www.lovejoy-inc.com/products/jaw-type-couplings.aspx		
	dsPICDEM MC1 Motor Control Development Board	Microchip	DM300020
Datasheet	http://ww1.microchip.com/downloads/en/DeviceDoc/70098A.pdf		
	6 position Connection Housing	Molex Connector Corporation	WM4267-ND
Datasheet	http://www.molex.com/pdm_docs/sd/050579506_sd.pdf		
	Crimps	Molex Connector Corporation	WM2510-ND
Datasheet	http://www.molex.com/pdm_docs/sd/016020102_sd.pdf		
	16 position Connection Housing	Molex Connector Corporation	WM2525
Datasheet	http://www.molex.com/pdm_docs/sd/022552161_sd.pdf		
	Butt Splice Nylon	Panduit Corporation	298-9995-ND
Datasheet	http://www.panduit.com/groups/MPM-NL/documents/PartDrawing/076693.pdf		

A.2.2 Solar Farm

	Part	Manufacturer	Part Number
	Microinverter	Enphase	D380-72-2LL-S12
Datasheet	http://enphase.com/wp-uploads/enphase.com/2011/03/Enphase_D380_Datasheet.pdf		
	MC4 Cables	Wholesale Solar	9991075
Datasheet	http://www.wholesalesolar.com/products.folder/cable-folder/mc4-extension-cables.html		
	AC Branch Cable	Enphase	ET3C-G2-06
Datasheet	http://enphase.com/wp-uploads/enphase.com/2011/05/Enphase_Field_Wiring_Diagram_D380_208v.pdf		
	DC Power Supply	Matsusada	RE45-45-LUs1
Datasheet	http://www.matsusada.com/pdf/RE.pdf		

A.2.3 Wind Farm

	Part	Manufacturer	Part Number
	Inverter	Powerex	PM30CSJ060
Datasheet	www.pwrx.com/pwrx/docs/pm30csj060.pdf		
	Current Sensor	Allegro MicroSystems	ACS712ELC-05B-T
Datasheet	http://www.allegromicro.com/en/Products/Part Numbers/0712/0712.pdf		
	Encoder	US Digital	E3-1024-875-I-H-T-B
Datasheet	http://www.usdigital.com/assets/general/76 e3 datasheet 3.pdf		
	Coupling	Lovejoy	L075-22mm
Datasheet	http://www.lovejoy-inc.com/products/jaw-type-couplings.aspx		
	Coupling Insert	Lovejoy	68514410621
Datasheet	http://www.lovejoy-inc.com/products/jaw-type-couplings.aspx		

A.3 Loads

A.3.1 Resistive Load

	Part	Manufacturer	Part Number
	Resistor	Milwaukee Resistor Corp.	8/91
Datasheet	http://www.milwaukeeresistor.com/pdf/roundwire.pdf		

A.3.2 Capacitive Load

	Part	Manufacturer	Part Number
	Capacitor	GE / Genteq	97F9391
Datasheet	http://www.richardsonrfpd.com/resources/RellDocuments/SYS 6/GEC001-MotorRun20080618.pdf		

A.3.3 Phase-controlled Resistive Load

	Part	Manufacturer	Part Number
	Dimmer Switch	Lutron	GL-600P-WH
Datasheet	http://www.lutron.com/TechnicalDocumentLibrary/Spec%20Guide%20Volume%201%20Glyder.pdf		

A.3.4 Induction Motor Load

	Part	Manufacturer	Part Number
	DC Motor	Reliance	DC0189ATY
Datasheet	http://www.reliance.com/prodserv/motgen/motinfcnt.htm		
	Induction Motor	Marathon Electric	143TTFS8026AB W
Datasheet	http://www.marathonelectric.com/MMPS/details.jsp?item=056T17F5322		

	Part	Manufacturer	Part Number
	DC Power Supply	Matsusada	RE500-2.4-LUs1
Datasheet	http://www.matsusada.com/pdf/RE.pdf		
	Resistor	Milwaukee Resistor Corp.	4/98
Datasheet	http://www.milwaukeeresistor.com/pdf/ribwound.pdf		
	Coupling	Lovejoy	L075-22mm
Datasheet	http://www.lovejoy-inc.com/products/jaw-type-couplings.aspx		
	Coupling Insert	Lovejoy	68514410621
Datasheet	http://www.lovejoy-inc.com/products/jaw-type-couplings.aspx		

A.4 Switching System

	Part	Manufacturer	Part Number
	Electromechanical Switch	TE Connectivity	P25P42D22P1-24
Datasheet	http://www.te.com/catalog/pn/en/2-1393132-6?RQPN=P25P42D22P1-24		
	Solid-state Switch	Carlo Gavazzi	RZ3A60D75P
Datasheet	http://www.gavazzionline.com/pdf/RZ3A.pdf		
	BJT	Fairchild Semiconductor	2N3904
Datasheet	http://www.fairchildsemi.com/ds/2N/2N3904.pdf		
	Diode	Fairchild Semiconductor	1N4002
Datasheet	http://www.fairchildsemi.com/ds/1N/1N4002.pdf		
	DAC	National Instruments	NI-USB 6008
Datasheet	http://sine.ni.com/ds/app/doc/p/id/ds-218/lang/en		

A.5 Monitoring System

	Part	Manufacturer	Part Number
	DAQ	National Instruments	NI-USB 6008
Datasheet	http://sine.ni.com/ds/app/doc/p/id/ds-218/lang/en		
	DAQ	National Instruments	NI-USB 6211
Datasheet	http://sine.ni.com/ds/app/doc/p/id/ds-9/lang/en		
	Transformer	Zettler Magnetics	AHI02512
Datasheet	http://www.alliedelec.com/images/products/datasheets/bm/ZETTLER_MAGNETICS/70037377.pdf		
	Current Sensor	Allegro Microsystems	ACS756KCA-050B-PFF-T
Datasheet	http://www.allegromicro.com/en/Products/Part_Numbers/0756/0756.pdf		
	Varistor	EPCOS Inc.	S07K385
Datasheet	http://www.epcos.com/inf/70/db/var_08/SIOV_Leaded_StandarD.pdf		
	Fuse	Littelfuse Inc.	0251.062MXL
Datasheet	http://www.littelfuse.com/data/en/Data_Sheets/Littelfuse_251_253.pdf		

	Part	Manufacturer	Part Number
	Terminal Block Connector	TE Connectivity	796949-7
Datasheet	http://www.te.com/catalog/pn/en/796949-7		
	Terminal Block Connector	TE Connectivity	796949-2
Datasheet	http://www.te.com/catalog/pn/en/796949-2		
	Terminal Block Connector	TE Connectivity	796949-4
Datasheet	http://www.te.com/catalog/pn/en/796949-4		