2006

Object-oriented shipboard electric power system library

Ram Praveen Saladi
West Virginia University

Object-Oriented Shipboard Electric Power System Library

by

Ram Praveen Saladi

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Electrical Engineering

Professor Muhammad A. Choudhry, Ph.D
Professor Ronald L. Klein, Ph.D
Professor Ali Feliachi, Ph.D., Chair

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2006

Keywords: Modelica, Power Electronics, AC Machine Drives, Faults, Petri Nets, Controls

Abstract

Object-Oriented Shipboard Electric Power System Library

by

Ram Praveen Saladi

The objective of this thesis is to explore the powerful capabilities of using an object-oriented modeling language to model and simulate an all electric Naval Shipboard Power System. Modelica has been used to model and simulate the shipboard power system which acts as an alternative simulation tool. The shipboard system is developed using the concept of packages. Different components like the buck converter, inverter, and AC machines have been modeled as a part of the library to develop the power system. The shipboard system has been simulated as two decoupled systems, the AC and DC systems. This research further focuses on developing a networked protection system to detect and clear faults and protect the shipboard power system from complete breakdown. A discrete supervisory controller has been designed using Petri nets as part of the protection system to control the converters and clear faults. A communication network has also been modeled for communication. Two different case studies, the open circuit test, and short circuit test were performed to test the effectiveness of the protection system and the simulation results are presented. This thesis also gives an overview of different properties of Modelica along with its advantages over other simulation tools, a detailed survey of different types of object-oriented simulation tools available, a comparison of different power electronics simulation tools, and some of the previous work done in Modelica.

Acknowledgement

First of, I would like to thank my advisor Professor Ali Feliachi for showing faith in me and giving me an opportunity to work for him. The completion of this thesis is definitely because of his encouragement. I also thank Dr.Karl Schoder who has been instrumental in giving right directions to my research and guided me all the time during my thesis.

I'm also thankful to both Professor Dr.Muhammad A. Choudhry and Professor Ronald L. Klein for their reviews and recommendations for my thesis, and also for guiding me at different stages of my Masters. I'm greatful to the entire APERC team members for helping me complete my thesis.

My family, though not present here, have always been my source for inspiration. I'm greatful to my father, mother, and brother for encouraging and supporting me all the time. I also thank my friend and fiancé Neelima who has been very understanding and supportive. I'm greatful to all the my friends who encouraged me and helped me complete my thesis.

Funding for this work was provided by the US DEPSCoR and Office of Naval Research (ONR) under grant (DOD/ONR N000 14-031-0660).

Contents

List of Figures

List of Tables

CHAPTER

1

Introduction

1.1 Introduction and Motivation

A large variety of software packages that support the development of models and simulation tools for studying the dynamic behavior of physical systems are available today from both research organizations and specialized commercial software vendors. By automating the process of simulating certain problem classes, these generic simulation systems enable practitioners without substantial programming experience to perform simulations easily. Physical systems are defined either by a mathematical model (e.g., a set of ordinary differential equations), or the model is interactively composed from predefined building blocks representing entities of the application domain using a graphical user interface. By drastically reducing the efforts for development of models, these packages considerably broadened the range of applications in several fields of engineering.

The increasing trend towards performing simulations of more complex and heterogeneous technical systems confronts these tools with substantial problems. In most cases, the models can be re-used only in the same system. The main cause for this problem is the absence of a state-of-the-art, standardized model representation. Model description languages, also known as general purpose modeling languages like MATLAB/Simulink [26] and ACSL [2], where employed, often do not adequately support the structuring of large, complex models. Also the process of model evolution due to their causal modeling nature, i.e., committing ports to function as inputs or output role [30]. The development of complex models is often provided by sophisticated graphical user interfaces, but at the price of specialization to a certain modeling formalism or application domain and uniqueness to a specific package like PSPICE [38] for electrical circuits, ADAMS [29] for multi-body systems, and ASPEN [5] for chemical processes.

To overcome model evolution, and causal modeling problems, general software engineering methods like object-oriented concepts are applied to the development of models to directly support the hierarchical structuring, reuse, and evolution of large and complex models. This method, called object oriented physical systems modeling, works independent of targeted application domain and of any graphical formalism used. Combined with another innovative modeling concept called non-causal modeling, the object-oriented modeling approach provides model reusability and general model evolution support.

1.2 Contributions of the Thesis

The objective of this research is to develop an object-oriented library of shipboard electric power system (SPS) components using Modelica. Modelica acts as an alternative modeling approach.

The components developed for the SPS library are designed to provide support for developing large and complex models. The list of components include electro-mechanical devices like the AC machines and power electronic converter models like the buck converter and rectifiers. A package for control models is also developed to aid in implementing control algorithms.

This research also develops a networked protection system to gaurd against different types of faults through fault detection and clearing. The protection component is designed using Petri net theory. Different case studies were performed to validate its concept and performance.

1.3 Outline of the Thesis

This section gives an outline of the remaining chapters presented in this thesis:

In Chapter 2 a review of different object-oriented modeling languages (Section 2.1), a comparison of different power electronics simulation tools (Section 2.2), some of the previous work done in Modelica (Section 2.3), and a brief discussion of the modeling and simulation of shipboard power systems (Section 2.4) are given.

In Chapter 3 properties of the Modelica language are presented (Sections 3.1 to 3.6) and different Modelica simulation environments (Section 3.7) are reviewed.

Chapter 4 gives an overview of libraries present in standard Modelica which have been used in this thesis. The libraries used are Electrical (Section 4.1), Blocks (Section 4.2), and Petri Nets (Section 4.3).

Chapter 5 provides details concerning the shipboard power system model components and the library structure (Section 5.1), the AC Generation and Propulsion System library (Section 5.2), the Power Supply Module (Section 5.3), the DC Distribution System Test Bed library (Section 5.4), the test examples package (Section 5.5), the power electronics library (Section 5.6), and machines library (Section 5.7).

In Chapter 6 a networked protection system is modeled to detect and clear faults. A communication network is modeled for communication of signals (Section 6.2), a model to create faults is presented (Section 6.3), protective device to detect the faults is presented (Section 6.4), a controller is designed which generate signals to clear the faults (Section 6.5), and case studies demonstrating the protection system are presented (Section 6.6).

In Chapter 7 a summary of the work done in this thesis is given (Section 7.1) and the possible directions for future work discussed (Section 7.2).

CHAPTER

2

Literature Survey and Related Work

This chapter gives an overview of object-oriented physical modeling languages in the first section, discusses various power electronics specific software tools in the second section, presents the previous work done in Modelica in third section, and the modeling and simulation of the shipboard power system in fourth section.

2.1 Object-Oriented Tools

The idea of **object-oriented modeling paradigm** was first introduced by Hilding Elmqvist [14] by proposing an equation based modeling language **DYMOLA** for design and implementation of physical systems. The main properties of the paradigm include

- Topographical interconnectivity.
- Hierarchical modeling.
- Encapsulation of knowledge.
- Class inheritance and Object instantiation.

Some of the common object-oriented modeling languages are:

Omola [4], [28] is based on the concept of model structuring, the ability to handle large and complex models. Some of the other features of Omola include equation based modeling, model reuse capability, model parameterization, and efficient handling of differential-algebraic equations, difference equations, and discrete events. Omola is implemented using OmSim, a simulation environment for translating and simulating models. OmSim has one disadvantage: differentiated variables cannot be given initial values.

The Neutral Model Format (NMF) [46] was developed along the lines of Omola and Dymola. The objective is to provide a common format of model expression for a number of existing and emerging simulation environments. The advantage of NMF over other languages is the way the details of a model are expressed, making them suitable for different simulation environments. Another advantage is the availability of good documentation.

SMILE [51] is similar to other object-oriented modeling languages like Omola and NMF. In SMILE equations can be specified symbolically or as procedures and also has a dedicated experiment description language and components for interactive experimenting and optimization. Different translators are available with a number of numeric solvers for simulating the models. The main application of SMILE is in design optimization of solar heating systems [6].

ALLAN [18] was developed based on the concept of reusability, and hierarchical nature. ALLAN is efficient to handle hybrid systems. The translators for the modeling language are written using FORTRAN or C language. ALLAN is used for modeling energy systems [19].

Modelica was developed by the Modelica association [31] pioneered by Martin Otter. It evolved from the object-oriented languages described above. The main idea behind Modelica was to develop a universal modeling language with the abilities present in different object-oriented modeling languages.

2.2 Power Electronics Simulation Tools

The simulation of power electronics circuits presents many challenges due to the nonlinear nature of the devices and because of the many switching events that take place. Some of the qualities a simulation tools ought to have to simulate power electronic circuits [32] are:

- Ability to handle non-linearities due to switching in an efficient manner.
- Capability of performing simulations for a small time step.

Many widely used commercial and freely available simulation tools allow simulation of power electronics circuits:

MATLAB/Simulink [26] is used for power electronics circuit simulations. The advantage is the presence of an extensive control library and being fast. The disadvantage is the tedious task of decomposing the entire system into state-space form and this becomes very difficult when the system is too complex. Power System Blockset (PSB) [17] is a commercial toolbox added for MATLAB/Simulink used for simulating the power electronic circuits. Using PSB can present convergence problems [50]. The PSB is a closed system and the models cannot be modified by the user to suit his needs.

PSpice [38] is used to perform simulation for both analog and digital circuits. It is well suited for modeling power electronic circuits but PSpice's control features are limited, does not allow data visualization during simulation, and simulation parameter setting is difficult in order to avoid convergence problem [50] making PSpice a poor simulation alternative.

EMTP [13], [20] is a simple, fast, stable simulation tool, and can perform both time and frequency domain simulations. The disadvantages of EMTP are: time step is fixed, the steady state solutions are limited to linear networks, and there are inherent time delays.

Modelica is a young and developing language capable of performing modeling and simulation of power electronic circuits. Advantages of Modelica are: models can easily be built based on acausal modeling appraoch, large and complex systems are easily modeled, physical appearance of the system is preserved, multi-domain modeling capabilities, hybrid event systems are handled efficiently, and control algorithms can easily be modeled. Standard, public domain Modelica library is available. More information about Modelica is given in Chapter 3.

2.3 Previous Work

The increasing demand for quality voltage and power in electrical circuits is leading to the development of new tools, and one such package is **P**ower **Q**uality **Lib**rary (PQLib) [47] developed using Modelica. The package consists of 3-phase passive electrical elements like resistors, capacitors and inductors, active elements like voltage and current sources. The package also has transmission lines, power factor correction devices like the filters and measuring instruments like rms voltmeters and ammeter and the digital frequency analyzer.

The core model used for the development of the PQLib is the 3ϕ connector pin called the **Pin3Ph**, which carries both voltage and current of all the phases. The Modelica code for the connector is given in Listing 2.1 . More information about the detailed modeling and implementation of the PQLib can

Listing 2.1: *3 Phase Pin*

```
connector Pin3Ph
Voltage1stPhase va;
Voltage2ndPhase vb;
Voltage3rdPhase vc;
flow Current1stPhase ia;
flow Current2ndPhase ib;
flow Current3rdPhase ic;
end Pin3Ph;
```

be found in [47].

A SPICELib library [24] was developed in Modelica along the lines of PSPICE. The library provides the models of linear resistors, inductors, and capacitors, controlled current and voltage sources, semiconductor devices like P-N junction diodes and MOS devices along with the ability to model circuits and perform different studies like bias point,component AC sweep, and transient analysis. The results of the SPICELib are compared to the ORCAD PSPICE version 9.1 in [38].

The ObjectStab library [27] was developed in Modelica for performing transient analysis of electric power systems at the transmission system level. The library consists of generators, transmission lines, reactive power compensation devices like shunt capacitors, reactors, fixed ratio transformers, tap changers, static and dynamic loads, buses and faulted lines. The voltage and currents in the library are represented in complex notation given by the equations $I = i_a + ji_b$ and $V = v_a + jv_b$.

The main component of the ObjectStab library is the connector pin with complex current and voltages components to directly represent the domain's fundamental information exchange model. The modelica code for the connector pin is in Listing 2.2.

This pin has been used as the basic building block for modeling the power system as a one-line diagram. Even though there are many commercial and efficient software tools present in the market for power system transient analysis like EuroStag [15], Simpow [48] and PSS/E [43], the advantage of the ObjectStab library is that it is easy to understand and the models can easily be changed and adopted according to the system under consideration and new components can be added.

A major step towards developing and implementing Modelica library for electrical circuits is the NCS library [21], modeled for designing and evaluating the NCS (National Combat Survivability) test-

Listing 2.2: *ObjectStab Pin*

```
connector Pin
  Base.Voltage va;
  Base.Voltage vb;
  flow Base.Current ia;
  flow Base.Current ib;
end Pin;
```

bed [40] of an all electric shipboard power system. The NCS library was modeled using the PQLibrary. The library explored new and different kind of hybrid controls like Petri nets [34] in designing both localized and supervisory controls for the shipboard power system and also explored the concepts of Networked Control Systems [8].

2.4 Modeling and Simulation

The shipboard power system is different from normal utility power system in many ways [7]. For example, very little rotational inertia relative to the load, both the electrical and mechanical systems are tightly coupled and faults need to be modeled consistently with the characteristics of the naval power system.

The increasing demand for reducing manning and costs while improving the survivability of a shipboard power system has led today's commercial and naval ships to change towards a completely electrical system. To meet these requirements and to develop different control techniques, a common reference system known as the Naval Combat Survivability Generation & Propulsion and DC Distribution Testbeds [40] [49] has been created.

Previously the shipboard AC power system was based on three-phase power generation and distribution in an ungrounded delta configuration [9] as a radial distribution system with either a single line of supply for less critical loads or two alternative paths for high priority loads.

The shipboard power system was simulated in [1], [54], [55] using different transient simulation tools like ATP [3], PSpice [38], and SABER [45]. A comparison of the tools suggested that ATP is a good tool for generating detailed component models, and PSpice and Saber have excellent graphical interfaces and flexibility in establishing monitor points and processing output data.

In [53] a test system of the shipboard power system was modeled and simulated using the Alternative Transients Program (ATP) [3] and a geographical information system. The test system electrical and geographical information was stored in the database. Different fault scenarios were studied with the test system based on geographical information stored and specific characteristics were used to enrich the shipboard power system database.

The Office of Naval Research has provided a basic MATLAB/Simulink model of the shipboard power system. The model is developed using basic simulink blocks and is based on state-space representation of the entire system. The ONR model requires for a time of one second approximately two and a half hours of CPU time without the Simulink Accelerator. The results of the simulation have been used for comparing the results of the simulation of the model developed in Modelica.

CHAPTER

3

Modelica

This chapter describes Modelica and features that make the modeling language different from other freely and commercially available tools.

3.1 Object-Oriented Modeling

Modelica emphasizes structured mathematical modeling through its object-oriented approach. This structuring concept helps handle large complex systems. A system is expressed in the form of equations known as **declarative style** of programming. In declarative style of programming, the focus is on what should be done and not how, as is the case in an algorithm and followed in traditional procedural languages.

The declarative style of object-oriented modeling is at a higher level than the usual object-oriented programming because the Modelica compiler generates code automatically to transmit data between objects and also code is not required for transmitting the data between objects [41].

3.2 Multi-Domain Modeling

Multi-Domain modeling means the ability to model different domains like mechanical, electrical, chemical and others. The advantage of Modelica over others is its multi-domain modeling capability, which makes it an ideal modeling package of electro-mechanical devices like robots. The models of different domains are connected together using different types of connectors and interface blocks. The standard libraries present in Modelica are “Electrical” used for modeling electrical circuits, “Mechanics” for mechanical components, “MultiBody” for modeling 3-dimensional mechatronic systems, “Thermal” for thermal systems, and “StateGraphs” to model discrete event systems.

3.3 Classes and Inheritance

Like any other object-oriented language, Modelica also provides the notion of classes and objects. A Modelica class consists of definition blocks where variables are specified, equations are written to specify the behavior of the class, and sometimes one class is a member of another class. The creation of an object, also known as instantiation, is different in Modelica from other languages like Java and C++ as the keyword **new** need not be specified for a creating a new object. Another advantage of object-oriented programming is the ability to extend the behavior and properties of a class called the **super class** or **base class** to another class called the **subclass** or **derived class**. This is called inheritance and is easily carried out in Modelica using the keyword **extends**.

3.4 Acausal Physical Modeling

One of the most interesting features of Modelica is **acausal modeling**. Acausal modeling means the input and output relationship between the variables need not be specified prior to the simulation. The causality of models is observed when the corresponding equations are solved by the translator. Due to this nature, acausal modeling is well suited for representing the physical structure of a system.

The advantage of acausal modeling over block diagram modeling can be further explained by the following example (Figure 3.1), which shows a simple electrical circuit model. Modelica code for the physical structure of the electrical circuit is given by the model **Simple Circuit** in Listing 3.1. The example demonstrates the use of which shows different classes and objects like the resistors R_1 and R_2, inductor L, capacitor C and the voltage source V_{ac}. These components have been connected together using the **connect** statement. The connect statement creates the equation for linking two components. For example the positive connector of V_{ac} is connected to the positive connector of R_1. The model is acausal since signal (information) flow has n’t been determined yet.

The same electrical circuit, when simulated using a block diagram modeling language like MAT-

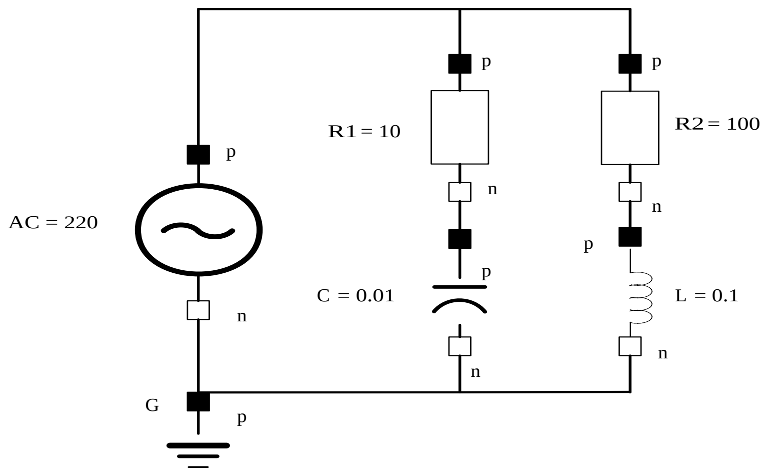


Figure 3.1: *Simple Electrical Circuit*

LAB/Simulink, the model looks as shown in Figure 3.2. The blocks completely change the appearance of the circuit and also the model is causal as the signal flow has been specified. Another disadvantage is that the resistors R1 and R2 definitions are context dependent making the reuse of the components of the model library difficult.

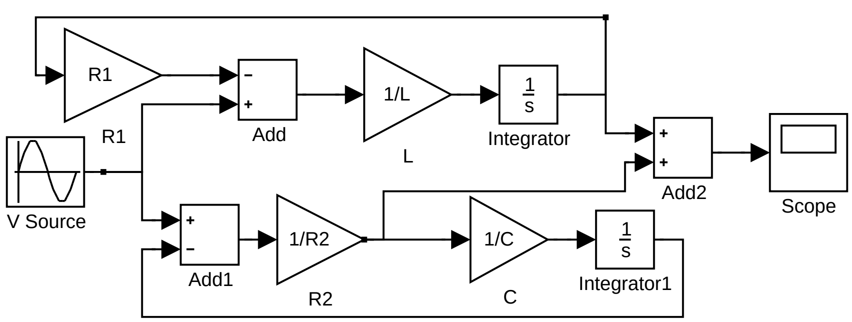


Figure 3.2: *Simulink Block Diagram*

3.5 Hybrid Modeling in Modelica

A hybrid system has both continuous and discrete components and modeling of such systems is called hybrid modeling. Hybrid modeling in Modelica is based on the synchronous data-flow principle [39]. According to which

“At every time instant, the equations which express relations between variables

Listing 3.1: *Simple Electrical Circuit*

```
model SimpleCircuit
Resistor R1(R = 10), R2(R=100);
Capacitor C1(C=0.01);
Inductor L (L = 0.1);
VsourceAC AC;
Ground G;
equation
connect(AC.p, R1.p);// Capacitor circuit
connect(R1.n, C.p);
connect(C.n, AC.n);
connect(R1.p, R2.p); // Inductor circuit
connect(R2.n, L.p);
connect(L.n, C.n);
connect(AC.n, G.p);  // Ground
end SimpleCircuit;
```

have to be fulfilled concurrently".

The synchronization principle is used to make hybrid modeling less error prone and is based on the following set of principles:

- An event is the time instant at which an action takes place. The computation time for the event is considered to be negligible.
- It is based on the single assignment principle according to which the number of unknown variables present in the model should be equal to the number of equations given in the model and if there are any mismatches either way the modelica translator gives an error.

The advantage of handling hybrid systems using synchronous data-flow principle is the synchronization between the continuous time and discrete event parts is automatic and leads to a deterministic behavior without conflicts.

3.6 Packages

The advantage of Modelica over other languages is the concept of **Packages** to build libraries as well as making a package part of another package. The packages concept helps in grouping models of similar nature together in a single package. Modelica being a multi-domain modeling language has a number of standard packages and these packages are structured as shown in Figure 3.3.

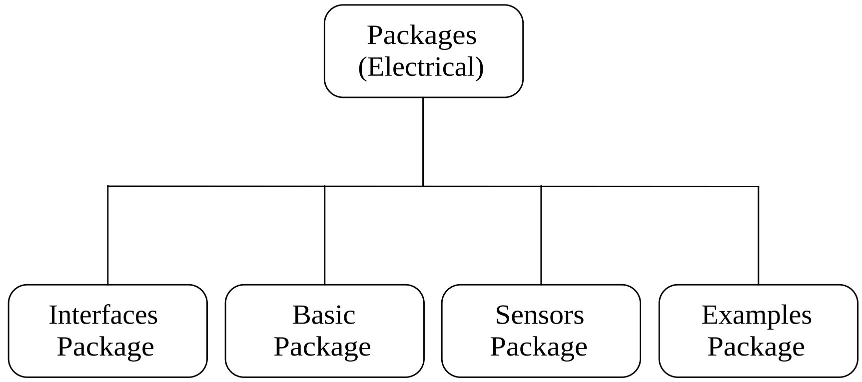


Figure 3.3: *Packages*

The structure of a library includes many nested packages like the interfaces, models, sensors, examples, and others explained below.

Interfaces: Interfaces is the basic building blocks of a package based on which the entire library is modeled. The interfaces package generally contains partial models and connectors used throughout the package.

Basic: The components present in this package are developed using the Interfaces package and also using other packages present in Modelica. The Basic package consists of basic elements used to develop systems more large and complex in the same domain.

Sensors: For a library that contains physical systems it is necessary to develop a package of sensors to measure different quantities. There are different types of sensors like the absolute sensor for the measurement of the absolute value at a point.

Examples: To show the working of the library and for the user to understand easily how the library works an Examples package is provided with some working examples of the models developed in the library, also some non-working examples of the library can be provided to show the user how the models should not be connected.

Types: Every package contains certain definitions useful for building the library, all these definitions are included in the Types sub-package.

Functions: This package is also similar to the type's package and the definitions in the functions package can be used throughout the package and there is also a possibility like the types package that there might not be any functions in the library.

Documentation: Once the library is developed a documentation of the entire library explaining each and every model of the library and also explain the limitations and applications of the library. The documentation of the library is generally preferred in the form of HTML format. Modelica has certain easy step by step commands using which the documentation of a library can be generated. The Figure 3.4 shows how the documentation of a library is presented in Modelica.

More details about Modelica are given in [41].

Name:Electrical
Path:Modelica.Electrical
Filename:C : /Dymola5/Modelica/Library/ Modelica 2.2/Electrical
Version:2.2
Description: Library for electrical models

This library contains electrical components to build up analog and digital circuits. The library is currently structured in the following sub libraries:

Package Analog for basic analog electrical components.
Package Digital for 2-, 3-, 4-, and 9-value logic of digital circuits.
Package MultiPhase for electrical components with 2, 3 or more phases.
Package Machines to model electrical motors and generators, especially three phase induction machines such as an asynchronous motor.

Figure 3.4: *Modelica Documentation Example*

3.7 Simulation Environments

To perform design and simulation of Modelica models a simulation environment is required. The simulation environment used for performing simulation of Modelica models should have the following properties:

- A *Modelica compiler or interpreter*, for translating Modelica code into machine executable code.
- An *execution and run-time module* to execute the transformed Modelica code having numeric solvers for handling discrete and hybrid models.
- A *text editor* for textual model editing.
- A *graphical model editor* for component based model design.

One simulation environment publicly available is the OpenModelica from Linkoping University [44]. It is an open source environment, but has lower performance and is not capable of handling large and complex systems yet another open source environment is SCICOS [35] with limited support to Modelica. MathModelica [25] and Dymola [10] are the commercially available simulation environments. Dymola has been used in this thesis because of better graphical and textual user interface, ability to generate models for use in MATLAB/Simulink, and ability to perform real-time hardware-in-the-loop simulations.

Dymola is developed by Dynasim in Lund, Sweden, and the name is an abbreviation of Dynamic Modeling Laboratory. The tool is designed to generate efficient code and has the capacity to handle variable structure Modelica models. The simulation of a modelica model in Dymola is a step-by-step process where initially the modelica source code is translated into a flat model with just equations and the object-oriented structure is lost. Afterwards, an analyzer sorts the equations according to the data-flow dependencies, the optimizer reduces the number of sorted equations by symbolic reduction methods, and these equations are finally converted into a C code using a code generator. This process is graphically represented in Figure 3.5.

Dymola has two different modes: the modeling and the simulation mode. In the modeling mode the models and model components are created by **drag and drop** from the Modelica libraries, equations and declarations are edited with the built-in text editor. The simulation mode makes it possible to do experiments on the model, plot results, and animate the model behavior.

In order to simulate the model, Dymola uses Dymosim, Dynamic Model Simulator. It is an executable, which is generated by Dymola and is used to perform simulations and compute initial values. It also contains the code that is required for continuous simulation and handling of events. These modelica models can also be exported to Simulink using Dymola when needed.

The result of the simulation can in turn be plotted or animated by Dymoview. Dymosim can be used in other environments too, though it is especially suited in combination with Dymola [11]. Dymola version 5.3a has been used for modeling the Modelica models in this thesis.

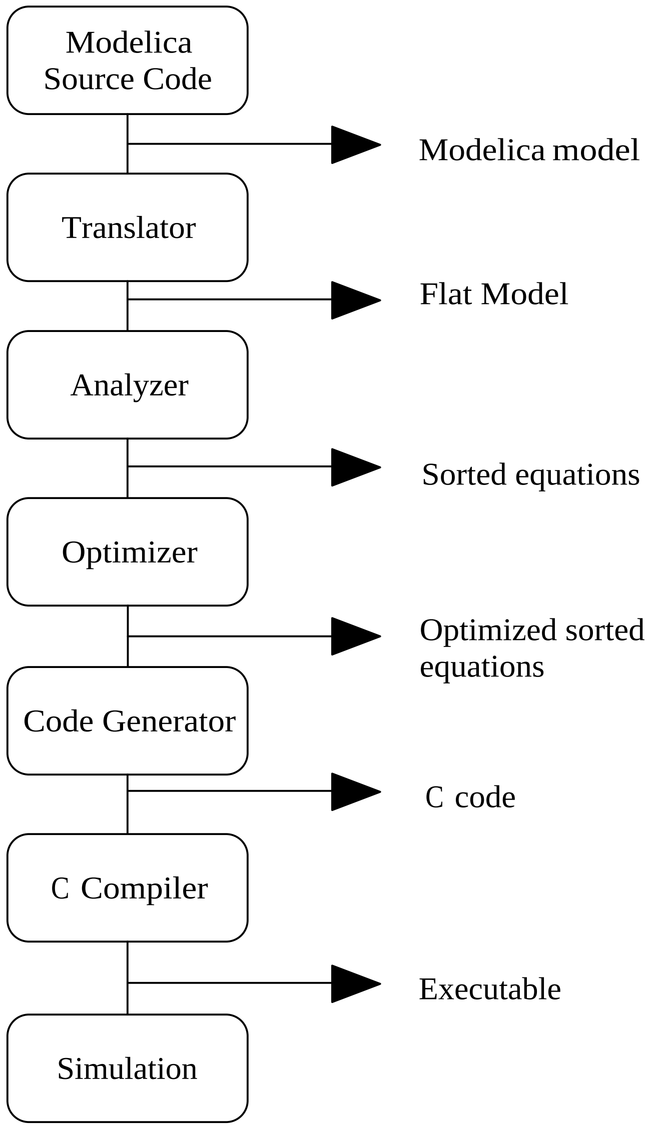


Figure 3.5: *Simulation in Modelica*

CHAPTER

4

Standard Libraries

The chapter gives an overview of the standard libraries present in Modelica used for modeling the Shipboard Power Systems (SPS) library.

4.1 Electrical

The Electrical library as the name suggests is used for modeling of electrical circuits both single and multiple phases. The library can also be used for modeling both AC and DC drives. The Electrical library has been used extensively for developing the shipboard power systems library. The Electrical library can be further classified into different libraries shown in the Figure 4.1, an overview of these libraries is given in the following sections.

4.1.1 Analog

This package was the first one available in the Electrical library which came along with the earliest version of Modelica. The library can be used to build and simulate electrical circuits. The connector **pin** present in the Interfaces sub-library acts as the base for building the components of the library. The connector pin carries two variables, the voltage and current. There are two types of pins, the **Positivepin** and the **Negativepin** which differ from each other only in their graphical annotation as shown in the Figure 4.2. The modelica code for the pin is shown in Listing 4.1.

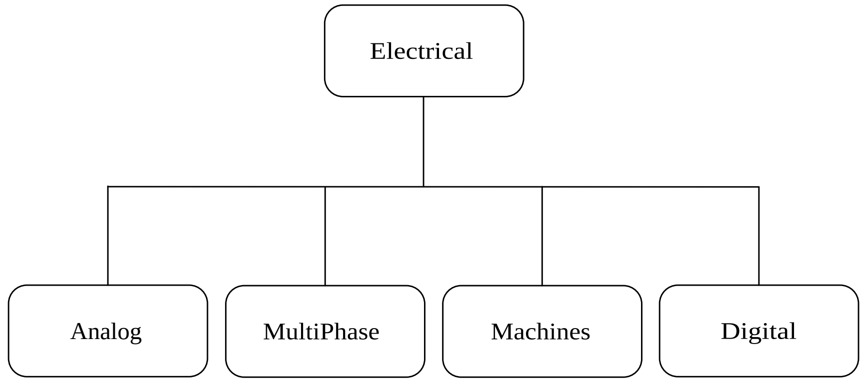


Figure 4.1: *Electrical Libraries*

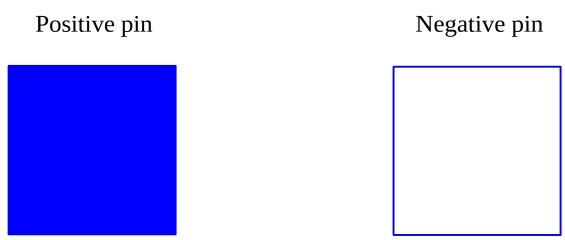


Figure 4.2: *Electrical Pins*

All the passive elements like the single phase resistors, capacitors, inductors and also the active elements like the AC and DC voltage sources and others are inherited from the superclass OnePort. The superclass OnePort has two electrical pins PositivePin p and NegativePin n such that a current **i** flows between the two pins along with a voltage drop **v** between them. The OnePort model can be seen in Figure 4.3.

The library is useful for modeling and simulating power electronic circuits due to the presence of the ideal power semiconductor devices like the diodes, thyristors, gate turn-off devices and the insulated gate bipolar transistors modeled as ideal opening and closing switches. Figure 4.4 shows

Listing 4.1: *Electrical Pin*

```
connector Pin "Pin of an electrical component"
  SI.Voltage v "Potential at the pin";
  flow SI.Current i "Current flowing into the pin";
end Pin;
```

Figure 4.3: *One Port*

Listing 4.2: *One Port model*

```
partial model OnePort
  SI.Voltage v "Voltage drop between the two pins (= p.v - n.v)";
  SI.Current i "Current flowing from pin p to pin n";
equation
  v = p.v - n.v;
  0 = p.i + n.i;
  i = p.i;
end OnePort;
```

an ideal closing switch having a boolean signal input which is the control signal. If the input to the control signal is true the switch closes and allows the current to flow between the pins. If the signal is false the switch acts as an open circuit and there is no flow of current between the pins.

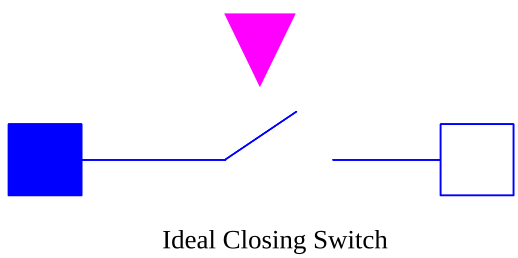


Figure 4.4: *Ideal Switch*

The Sources package in the library contains the different kinds of sources like the constant DC Voltage source, current source, sine voltage, and the variable voltage and current sources. The library also has a Sensors library which contains a voltage sensor and a current sensor.

Listing 4.3: *Ideal Switch*

```
model IdealClosingSwitch "Ideal electrical closer"
 extends Modelica.Electrical.Analog.Interfaces.OnePort;
 parameter SI.Resistance Ron(final min=0) = 1.E-5
 parameter SI.Conductance Goff(final min=0) = 1.E-5
 Modelica.Blocks.Interfaces.BooleanInput control
 protected
      Real s "Auxiliary variable";
 equation
      v = s*(if control then Ron else 1);
      i  = s*(if control then 1 else Goff);
end IdealClosingSwitch;
```

4.1.2 MultiPhase

The MultiPhase library was made part of the Modelica Electrical standard library when Modelica 2.1 Beta version was released. The MultiPhase library is similar in structure to the Analog library but the models are extended to multiple phases. The advantage of using the MultiPhase library over the Analog library is that the 3-phase and multiphase systems can be easily modeled using the multiphase components. Another advantage of MultiPhase library is the models developed look less complex because of the one line diagram connections between the components.

In MultiPhase library the core model used for developing the entire library is the **Plug**, an electrical connector for connecting different models. The Plug is developed using an array of *m* **Electrical.Analog.Interfaces.pins**. The variable m is given as input to the Plug. The default value for *m* is 3. The graphical annotation of the Plug is shown in Figure 4.5.

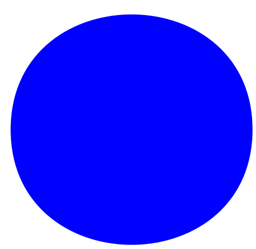


Figure 4.5: *MultiPhase Plug*

The MultiPhase library, apart from having all the components present in the Analog library also

Listing 4.4: *Plug Connector*

```
connector Plug "Plug with m pins for an electric component"
  parameter Integer m(final min=1) = 3 "number of phases";
  Modelica.Electrical.Analog.Interfaces.Pin pin[m] ;
end Plug;
```

has some additional models for interfacing with the Analog library. The **PlugtoPin** model has only one pin of the plug connected to the analog pin, another model is the **Star** connector with a Plug and a Negative Pin where all the pins present in the plug are connected to the Negative Pin. The Star connector is used for two purposes as it can be used to interface the analog library with the Multiphase library or can be used to form a wye connected electrical circuit. Analogous, **Delta** model exists.

4.1.3 Machines

The Machines library developed by Dr. Anton Haumer [22] was released along with Modelica version 2.1. The library showcases a perfect example of the multi-domain modeling capabilities of Modelica as both electrical components like resistors and inductors and mechanical components like the flanges and inertia components have been used for building the library. The library has models of machines like the DC machines, 3-phase asynchronous machines and the permanent magnet synchronous machine and the reluctance machine. The machines are modeled by considering the physical units of the components rather than per unit values. The modeling of the components of the library is explained by taking a closer look of the asynchronous machine in the following.

The model of the squirrel cage induction machine shown in Figure 4.6 is based on the Space Phasor theory according to which $3-\phi$ instantaneous voltages or currents can be transformed into phasor form (indicated by underline) and vice versa given by equations (4.1) - (4.5).

$$\underline{v} = \frac{2}{3}\left(v_1 + e^{j2\pi/3}v_2 + e^{-j2\pi/3}v_3\right) \tag{4.1}$$

$$v_0 = \frac{1}{3}\left(v_1 + v_2 + v_3\right) \tag{4.2}$$

$$v_1 = v_0 + Re\left(\underline{v}\right) \tag{4.3}$$

$$v_2 = v_0 + Re\left(e^{j2\pi/3}\underline{v}\right) \tag{4.4}$$

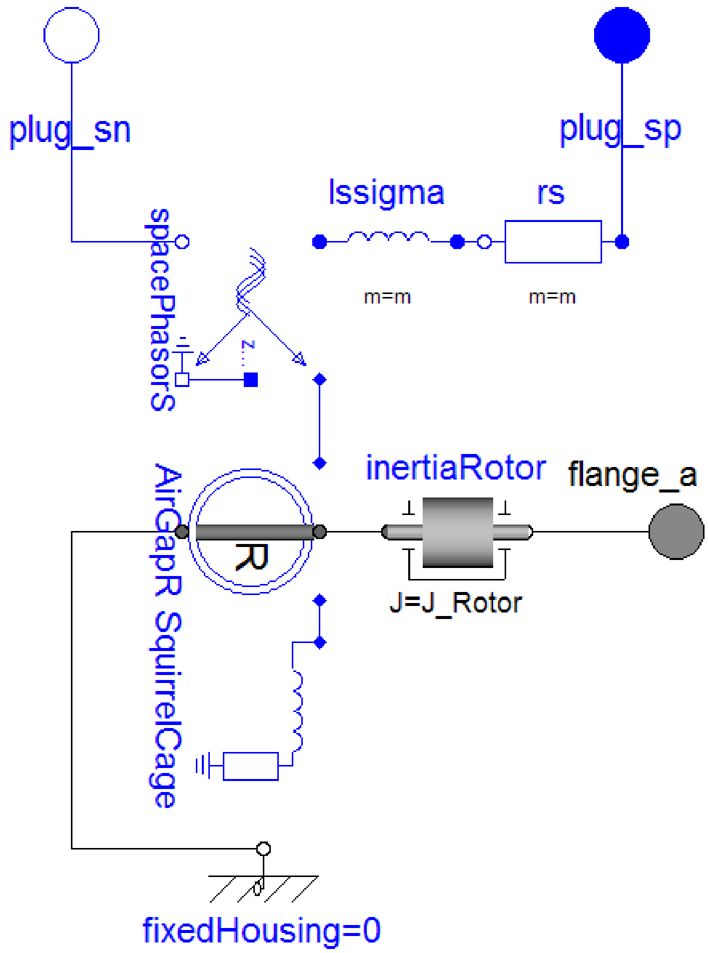


Figure 4.6: *Asynchronous Machine*

$$v_3 = v_0 + Re\left(e^{-j2\pi/3}\underline{v}\right) \tag{4.5}$$

The stator voltages of the three phases of the induction motor is given by the equation (4.6)

$$v_{si} = R_s i_{si} + \frac{d\psi_{si}}{dt} \tag{4.6}$$

Equation (4.6) is represented in the model by a MultiPhase Resistor R_s connected to the supply mains and a MultiPhase Inductor L_{ssigma} connected to the SpacePhasorS model. The SpacePhasorS model transforms the $3-\phi$ instantaneous stator voltages into space phasor form using equations (4.1) - (4.5). The stator voltage space phasor equation in stator fixed co-ordinate system transformed by the SpacePhasorS is given by eqaution (4.7).

$$\underline{v}_{S(s)} = R_S \underline{i}_{S(S)} + L_{S\sigma} \frac{d\underline{i}_{S(S)}}{dt} + \frac{d\underline{\Psi}_{m(s)}}{dt} \tag{4.7}$$

The squirrel cage model, which acts as the rotor of the model, is given by the transformed voltage equation (4.8).

$$\underline{v}_{R(s)} = R_R \underline{i}_{R(s)} + L_{R\sigma} \frac{d\underline{i}_{R(S)}}{dt} + \frac{d\underline{\Psi}_{m(R)}}{dt} \tag{4.8}$$

The transformed voltage and current from the stator and rotor are sent to the **AirGap** model to calculate flux linkages and the electromagnetic torque developed in the machine using equations (4.9)-(4.10).

$$\underline{\Psi}_m = L_m \left\{ \underline{i}_S + \underline{i}_R \right\} \tag{4.9}$$

$$T_{el} = \frac{2}{3} p I_m \left\{ \underline{i}_S \underline{\Psi}_S^* \right\} \tag{4.10}$$

The AirGap models are the basic elements of the Machines library. The AirGapS is connected by a space phasor connector called the **SpacePhasor**. The connector carries the transformed voltages and currents as an array of voltage v_[2] and current i_[2] shown in Figure 4.7.

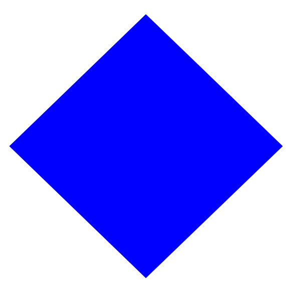


Figure 4.7: *Space Phasor Connector*

The induction motor model is modeled either referring to stator or rotor using AirGapS or AirGapR models respectively. The DC machines in the library are modeled using the AirGapDC model. The PMSM is also modeled in the same way as the induction motor along with an additional model **PermanentMagnet** connected to the squirrel cage model for providing the constant flux for the machine.

Listing 4.5: *Space Phasor*

```
connector SpacePhasor
  Modelica.SIunits.Voltage v_[2];
  flow Modelica.SIunits.Current i_[2];
end SpacePhasor;
```

4.2 Blocks

Once the models are developed, the controllers to control the power electronic devices are designed using the Blocks library. The transmission of signals between the controller and the plant is carried out using the **Real** and **Boolean** input and output connectors present in the Blocks library are shown in Figure 4.8.

Figure 4.8: *Real and Boolean Connectors*

The Real and Boolean connectors shown in Figure 4.8 form the base of the Blocks library. The library is used for modeling different control algorithms using the sublibraries present, which includes the **continuous** library having integrators, state-space blocks, a **discrete** library having the discrete form of continuous library. The Blocks library also has Logical sub-library with all the logical models like the AND, OR, NOT. The blocks library also has a Math sub-library to perform mathematical operations, a non linear library with a limiter and a delay block. The library also has Multiplexers and De-multiplexers in the Routing sub-library for real signals and different kinds of sources like the step signal, constant signal are present in the Sources sub-library.

4.3 Petri Nets

The concept of **Petri Nets** was first introduced by Carl Adam Petri in his dissertation [42]. Petri Nets are a widely used graphical and mathematical formalism for modeling and analyzing discrete event systems and are well suited to handle concurrent, asynchronous, parallel, distributed, non deterministic and stochastic behavior of systems [34].

Petri nets primarily consists of two different nodes **Places** (depicted as circles) and **Transitions** (depicted as vertical bars). The transitions and places are connected to each other and form a directed bipartite graph, which means no two places or transitions are connected together. Places have a **marking** (state) consisting of a nonnegative integer k called **token**. The state of the place changes when the tokens present in the place shifts according to the following transition firing rule.

1. A transition t is enabled if markings of each input place p connected to t has k tokens.
2. Enabled transition may or may not fire depending on whether an event takes place or not.
3. A transition t, if fired, removes tokens from each input place and adds the tokens to each output place.

Some of the application areas where Petri nets can be efficiently used are finite-state machines, communication protocols, and dataflow computations. More information about Petri nets and their properties and applications can be found in [34].

Petri nets library is a part of the standard Modelica library [33]. The initial Petri nets library in Modelica was modeled based on the following properties:

- The Petri nets modeled are *Normal* Petri nets as the number of tokens present in each place is at most one.
- The Petri nets are deterministic in nature. This is achieved by assigning priorities to transitions. If several transition are enabled at a time, the one with highest priority will fire. These are known as *Priority* Petri nets.
- The transitions are fired immediately without any delay.

The initial library was later extended to include transitions with deterministic or stochastic time delays and places having more than one token [16]. In each place and transition a set of boolean signals are present indicating the state of the place and if the transition has fired or not. Both the initial and the extended Petri net libraries have been used in this thesis. The libraries have been used with a slight modification. The current state of the places components present in the petri net library is taken as output using a boolean signal connector. The graphical annotation of places and transitions of the Petri net library are shown in Figure 4.9.

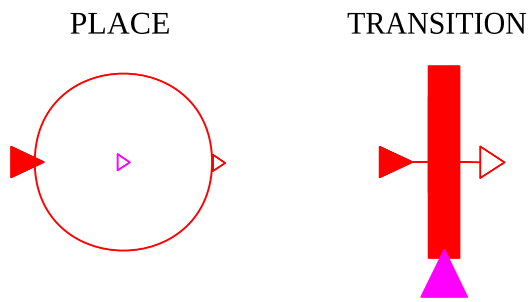


Figure 4.9: *Petri nets*

The working of the Petri net library in Modelica can be further explained by the following example shown in Figure 4.10. The example illustrates the connections between the places and transitions. The initial state of the place **initialstage** is true and changes to false when transition **T1** fires for time $t > 0$ seconds. The transition T1 also changes the state of the **Stage1** place from false to true. The transition **T2** fires at time $t = 0.2$ seconds and changes the states from true to false and false to true for the places Stage1 and Stage2 respectively. The time delay transition **TDelay** fires after a time delay of $\delta t = 0.2$ seconds at t = 0.4 and changes the state of **FinalStage** from false to true. Figure 4.11 show the states of the of the places initialstage which always remains false at zero level, Stage1 becomes true at the start of the simulation and becomes false at t = 0.2 seconds, Stage2 remains true from t = 0.2 to 0.4 seconds, and finalstage becomes true at t = 0.4 seconds.

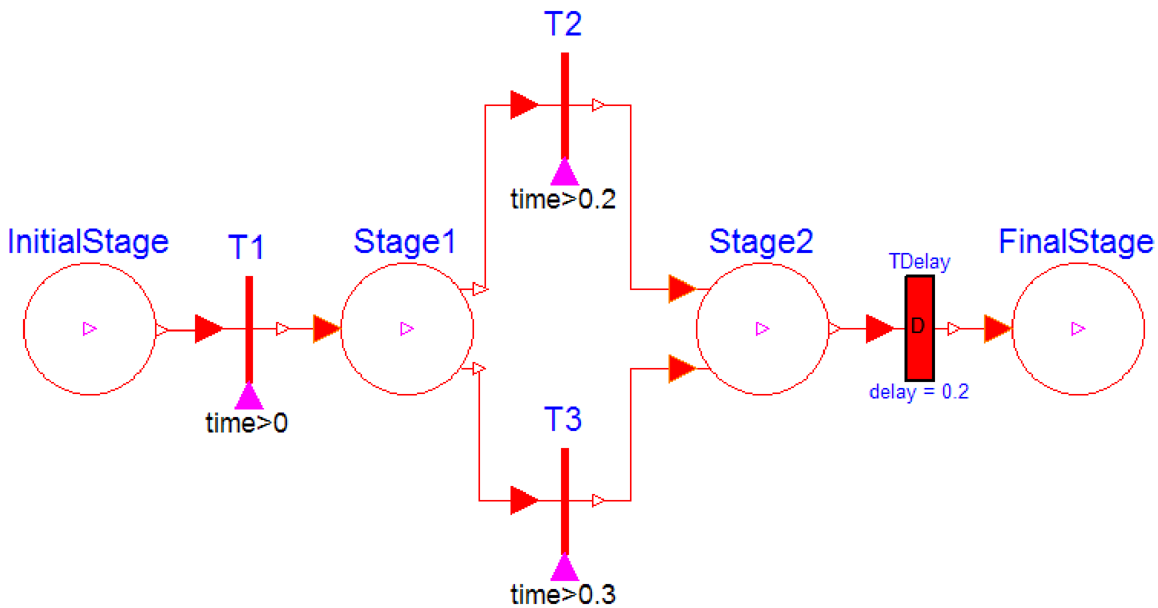


Figure 4.10: *Petri nets Example*

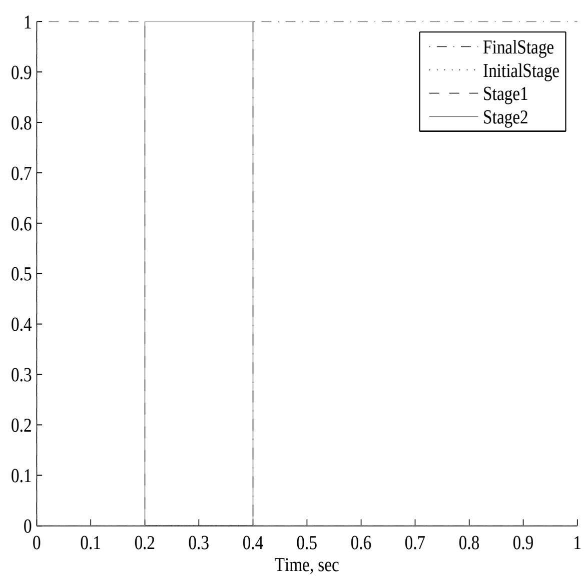


Figure 4.11: *Petri Net Example: Places' States*

CHAPTER

5

Shipboard Power System Library (SPS)

5.1 Introduction

The modeling of the components of the shipboard power system is based on the Naval Combat Survivability Generation and Propulsion and DC distribution testbeds [40] [49] shown in Figure 5.1.

The testbed is fed by two synchronous generators. The two propulsion systems are connected to the generator using a propulsion converter designed using power electronics. The AC is converted to DC for distribution by the power supply modules. The DC electric power is supplied by two DC buses to the loads, the port bus and starboard bus. The loads are grouped into different zones and for each zone the power is supplied from both of the DC buses by the DC-DC converters. The survivability of the ship is increased by supplying the loads from both the buses. The loads include induction motors, three-phase AC-loads, and constant power loads. The AC generator along with the propulsion system is called as the AC Generation and Propulsion System Testbed and the DC system is called the DC Distribution Testbed.

The idea behind developing a shipboard power system library in Modelica is take advantage of object-oriented design and investigate simulation performance. The performance of the shipboard power system is analyzed by decoupling the system and simulating as two separate models the AC Generation and Propulsion system and the DC Distribution System. The library uses the concepts of

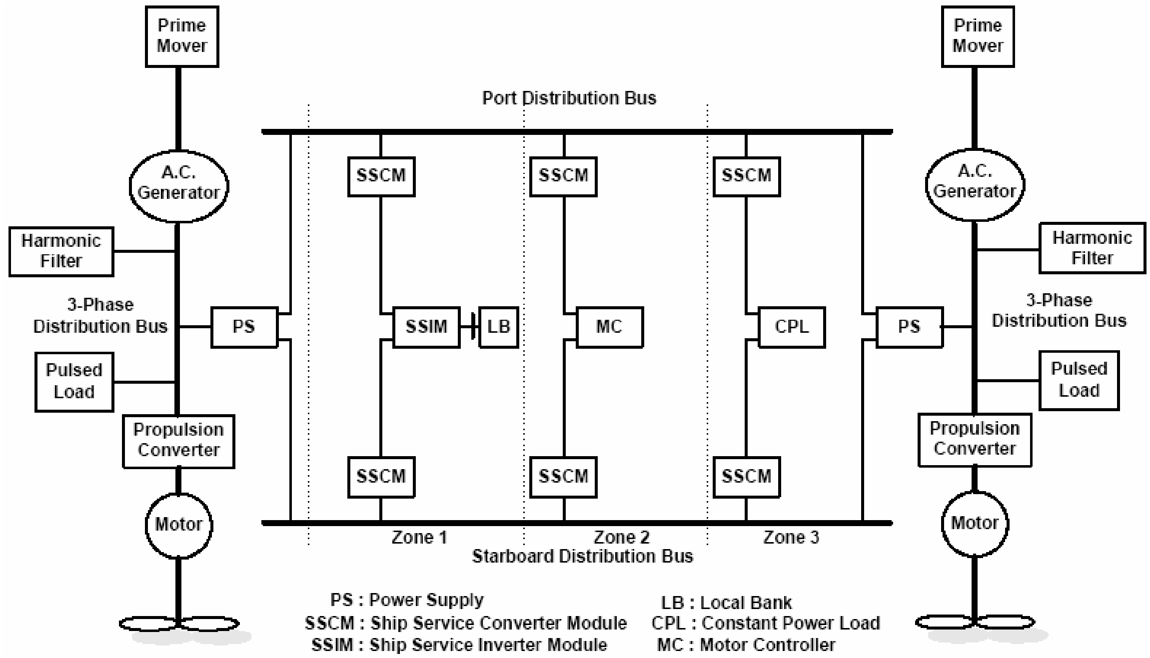


Figure 5.1: *Shipboard Power System*

Modelica Packages as explained in Section 3.6.

The structure of the shipboard power system **SPS** library is shown in Figure 5.2. The library is divided into different packages each having a group of related components. These packages are explained in detail in the following sections.

5.2 AC Generation and Propulsion System

The AC generation and propulsion system has a 59 kW generator connected to 37 kW propulsion system converter with an induction motor load through an AC bus. A wye-connected harmonic filter is connected to the AC bus. The entire system has been modeled as **ACSYS** package. A detailed modeling of the ACSYS library is explained in the coming sections.

5.2.1 AC Generator

The power supplied to the shipboard power system is generated by a 59 kW, 560 V, line-line, rms synchronous generator modeled by **ACGenerator**. The Figure 5.3 shows a synchronous machine connected to a prime mover and a voltage source measuring the generator voltage given as input to

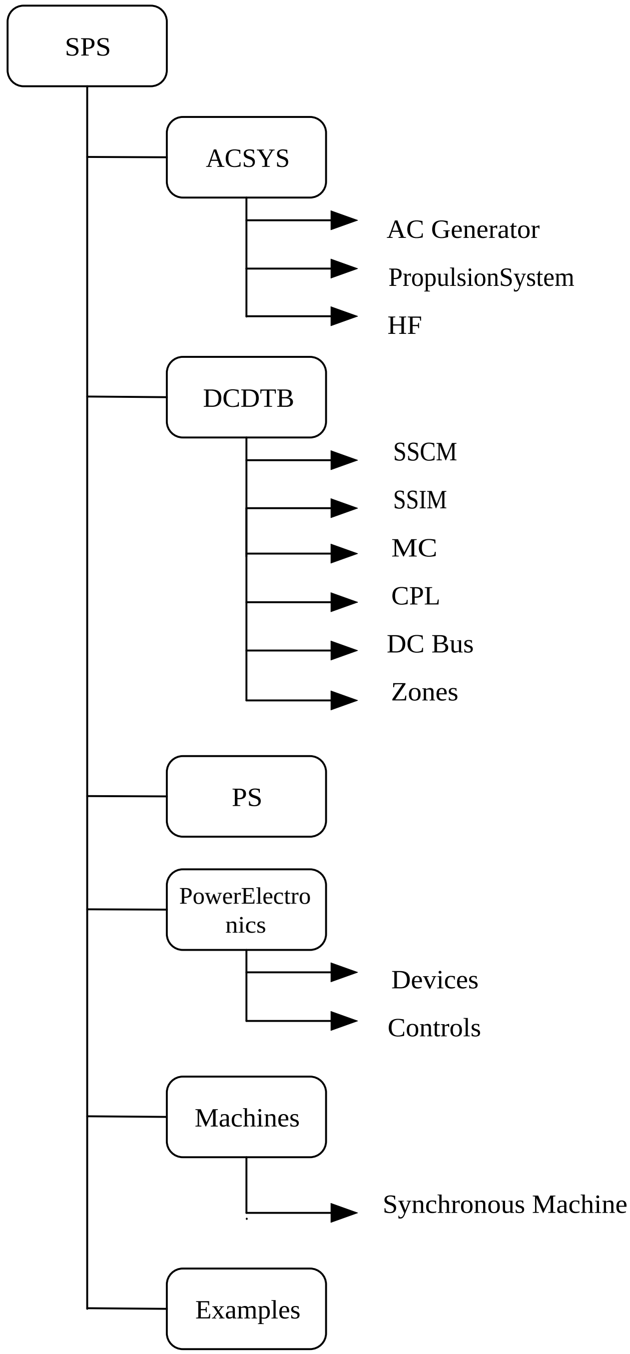


Figure 5.2: *SPS Library*

the excitation control.

The prime mover of the machine is **Modelica.Mechanical.Rotational.ConstantSpeed** a constant speed input model. The model generates the required torque to maintain a constant speed for

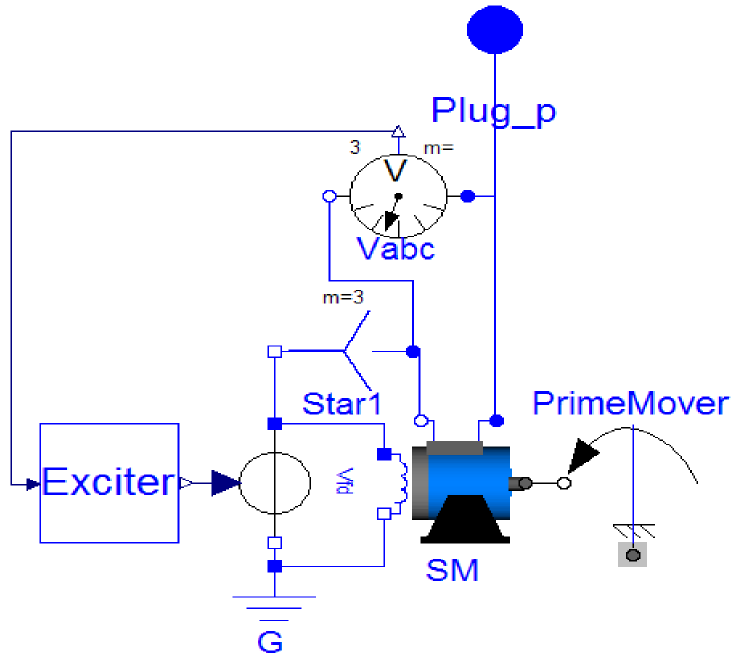


Figure 5.3: *AC Generator*

the generator. The constant speed at which the generator runs is given as parameters to the model.

5.2.2 Harmonic Filter

The generator, propulsion system, and the power supply module are connected using an AC bus. The AC bus is used to supply the voltage generated by the generator. The bus has a capacitor C and a shunt resistor R. A L-C wye-connected harmonic filter is connected to the AC bus to absorb the harmonics generated by the propulsion system. The AC bus with harmonic filter is modeled as **HF** .

Figure 5.4 shows the AC bus with harmonic filter. The AC bus is represented by the Multiphase resistor R_r and capacitor C_r, and the harmonic filter is represented by R_F, L_F, and C_F. The AC bus is connected externally to the generator, propulsion system, and the power supply module by the multiphase plug **PLug_P**.

5.2.3 Propulsion System

The propulsion system is a 37 kW, 1800 rpm, 460 V line-line rms induction motor drive. The system is divided into two parts: the power electronics part consisting of propulsion converter, DC link capacitor

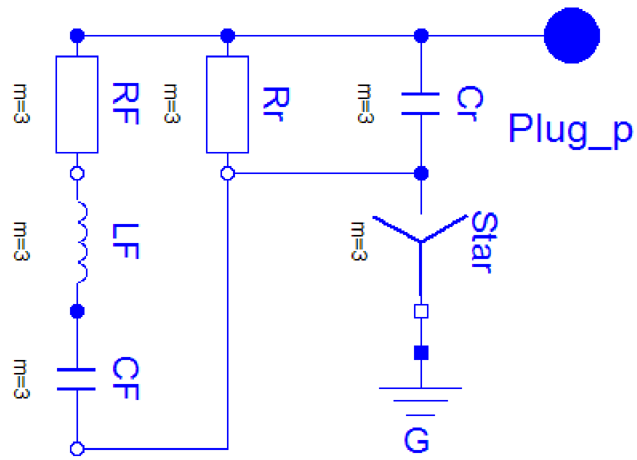


Figure 5.4: *Harmonic Filter*

and inverter and the induction motor part as shown in the Figure 5.5. The system is modeled as **PropulsionSystem** in Modelica. The input voltage to the system is from the voltage generated by the AC generator through the AC bus. The load for induction motor is given as

$$T_l = 1.6886 \times 10^{-3} \omega_{r,mech}^2 \tag{5.1}$$

The induction motor control is based on the constant slip-frequency control with torque reference input from [23]. The advantage of this control algorithm is the currents are readily limited making the drive robust, and, enables the use of less conservatism when the ratings of inverter semiconductors are chosen. The disadvantage is the controller requires phase current feedback which is expensive.

According to the control theory explained in [23], the reference currents are generated by transforming the abc currents to q-axis and d-axis current components. The q-axis current component is given by

$$\mathbf{I}_{qs} = \sqrt{\frac{2\,|T_e^*|\left(r_{r,est}'^2 + \left(\omega_s L_{rr,est}'\right)^2\right)}{3P\,|\omega_s|\,L_{M,est}^2 r_{rr,est}'}} \tag{5.2}$$

The d-axis and the zero sequence component of the current are made constant and set to zero and is given by equation (5.3).

$$\mathbf{I}_{ds} = 0,\ \mathbf{I}_{0s} = 0 \tag{5.3}$$

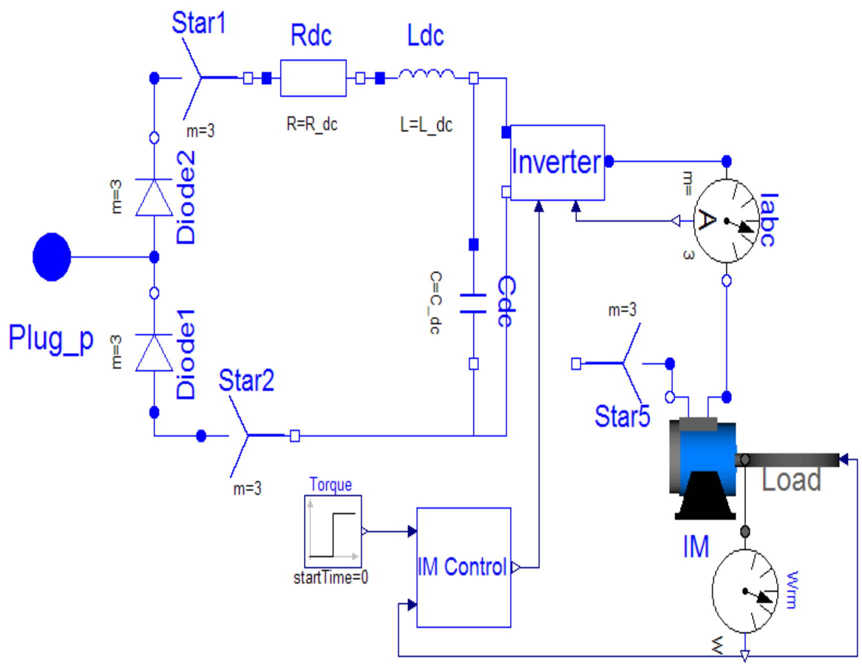


Figure 5.5: *Propulsion System*

The slip frequency is calculated as the ratio of the rotor resistance and inductance given by equation (5.4) and added to the rotor speed to calculate the reference frame components.

$$\omega_{s,set} = \frac{r'_{r,est}}{L'_{rr,est}} \tag{5.4}$$

The control block diagram shown in the Figure 5.6 is modeled using the equations (5.2) to (5.4). The inputs to the controller are rotor speed ω_r and the reference torque T_{ref}.

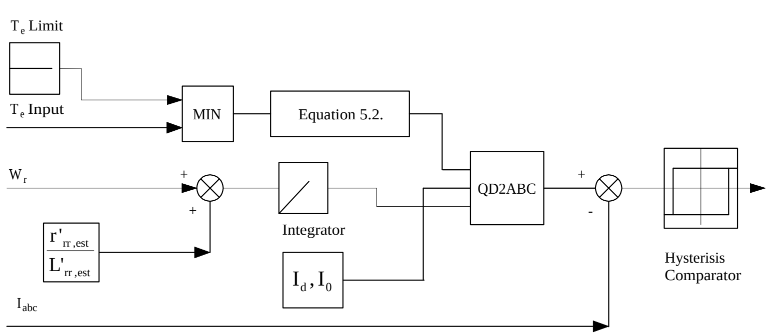


Figure 5.6: *Propulsion System Controller*

5.3 Power Supply Module

The Power Supply module is modeled as **PS**. The power supply module converts the voltage generated by the AC generator into a regulated DC voltage. The power supply module has an uncontrolled rectifier along with a buck converter to regulate the output of the rectifier. The output power of the model is a maximum of 15 kW. The input to the model is 560 V lline-line rms and the output of the converter is 500 V DC. Two power supply modules are present in the shipboard power system and each is connected to the starboard side DC bus and Port DC bus respectively to provide continuous supply of power to the zones. The power supply module shown in Figure 5.7 has a pair of **Modelica.Electrical.MultiPhase.Ideal.IdealDiodes** as an uncontrolled rectifier connected to the buck converter modeled (explained in section 5.6.1).

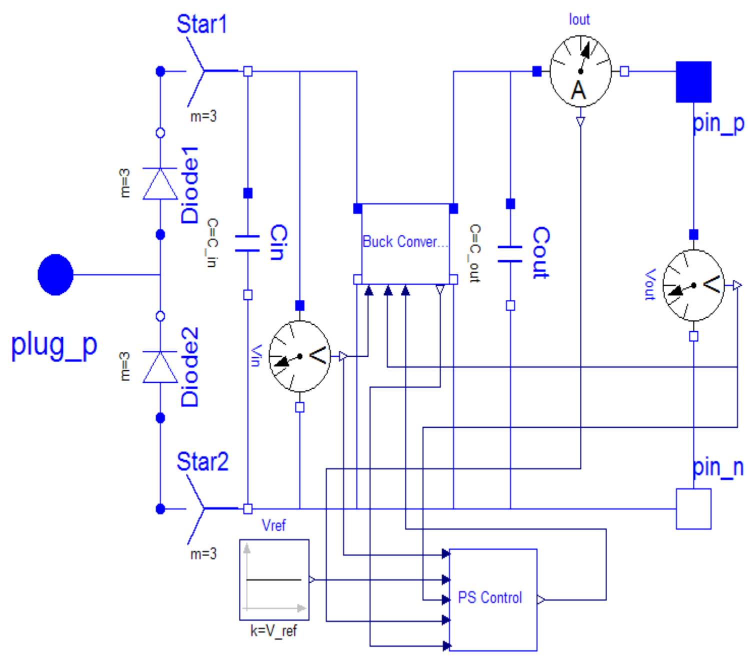


Figure 5.7: *Power Supply*

The output voltage of the Buck converter is regulated by the control block diagram shown in the Figure 5.8. The controller is modeled as **PS_Control**. The inputs to the controller are the reference voltage V_{ref}, the actual output voltage V_{out}, the DC output voltage from the uncontrolled rectifier V_{dc}, load current I_L and output current I_{out} from the converter. The voltages V_{ref} and V_{out} are compared using a PI controller and added to load current. The sum is multiplied by modulated output voltage of the controller to give the reference current. The reference current is compared with the actual current using a hysteresis comparator to control the buck converter.

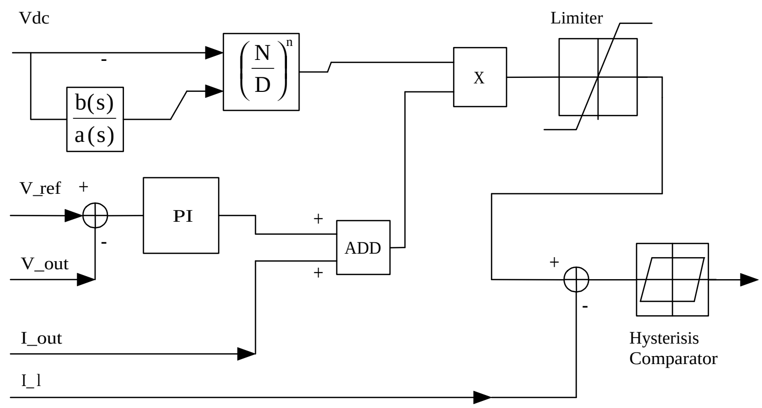


Figure 5.8: *Power Supply Controller*

5.4 DC Distribution Test Bed

The DC distribution test bed is modeled as **DCDTB** library as a part of the SPS library. All the components present in the DC Zonal distribution test bed have been modeled in the library. The models developed include the DC bus, ship service converter module, ship service inverter module, the motor controller and the constant power load model. All the models developed are explained in the following sections.

5.4.1 DC Bus

To supply voltage from the power supply module to the zones a DC bus of 500 V capacity is modeled. The DC bus is modeled as a lump of resistors, inductors, capacitors and conductances. Two similar models of the DC bus are modeled, the starboard bus **Stbd_Bus** and the port bus **Port_Bus** to supply voltage from both the power supply modules present in the shipboard power system. The DC bus model is shown in Figure 5.9

5.4.2 Ship Service Converter Module

The Ship Service Converter Module (SSCM) is modeled as **SSCM** and it is a buck converter along with an L-C filter as shown in Figure 5.10. The rated output power of the SSCM is 8 kW and converts

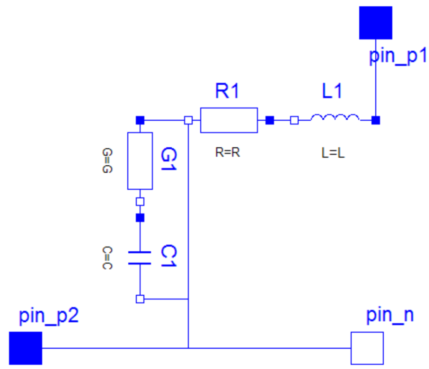


Figure 5.9: *DC Bus*

500 V DC input to a regulated 400 V DC output with a maximum current of 20 A. Each zone of the shipboard power system has two converter modules each connected to one of the starboard and port DC buses. The output from the converter modules is supplied to the load banks in each zone.

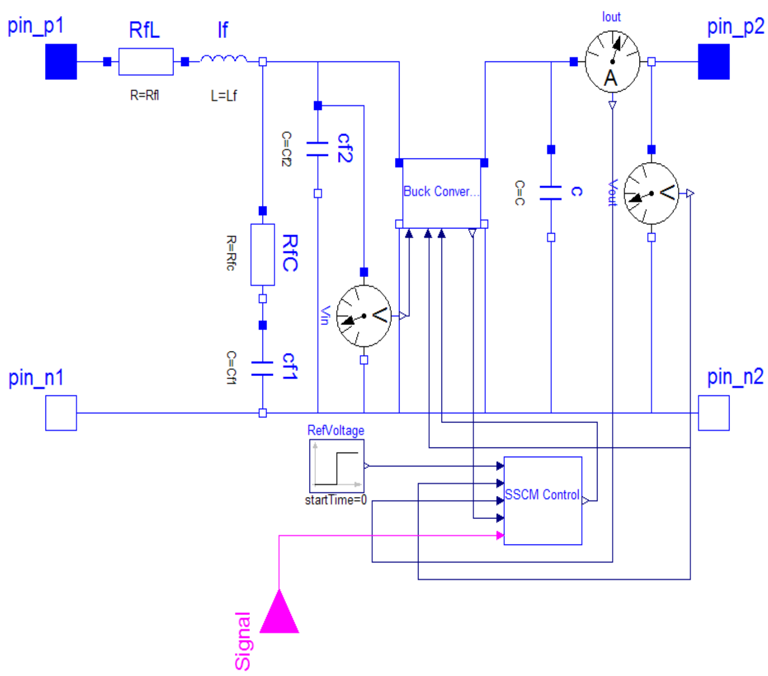


Figure 5.10: *Ship Service Converter Module*

The control of the converter module (**SSCM_Control**) is shown in the Figure 5.11. The inputs to the controller are the rated voltage V_{ref}, the output voltage V_{out}, load current I_L, and the output current I_{out} from the converter. The voltages are compared using a PI controller and added to the current error to generate signals for controlling the buck converter output.

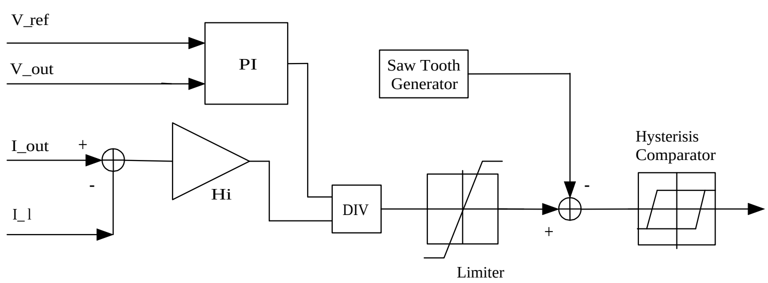


Figure 5.11: *Ship Service Converter Module Control*

5.4.3 Ship Service Inverter Module

The Ship Service Inverter Module modeled as **SSIM** is a 3ϕ DC-AC inverter. The inverter module acts as a load bank in Zone 1 of the shipboard power system. The input to the inverter is 400 V DC. Figure 5.12 shows the block diagram of the inverter module where the inverter is connected to a capacitive filter at the input and a delta connected R-C filter model **DeltaC** at the output. The load on the inverter is a delta connected R-L load modeled by **Delta_Load**.

The controller for the inverter module is modeled within **SSIM_Control**. Decoupling feedback control strategy [52] is used to control the output voltages by controlling the inverter currents. According to the control strategy the output voltages measured V_{an}, V_{bn}, and V_{cn} are transformed into synchronous reference frame voltages as V_q and V_d and compared with reference voltages V_{qref} and V_{dref} using a PI controller. The inverter load currents i_{aL}, i_{bL}, and i_{cL} The PI controller output is added to the feed forward current signals i^{e*}_{qF} and i^{e*}_{dF} to generate the reference currents for the inverter. The feed forward currents are generated by adding the transformed currents I_{qL} and I_{dL} of the measured load currents and the decoupled feedback signals. The decoupled feedback signals are given by equations 5.5 and 5.6.

$$i^{e*}_{qD} = \omega_e C_{eq} v^e_d \tag{5.5}$$

$$i^{e*}_{dD} = -\omega_e C_{eq} v^e_q \tag{5.6}$$

The decoupling is achieved by multiplying the d and q axis voltage components with the syn-

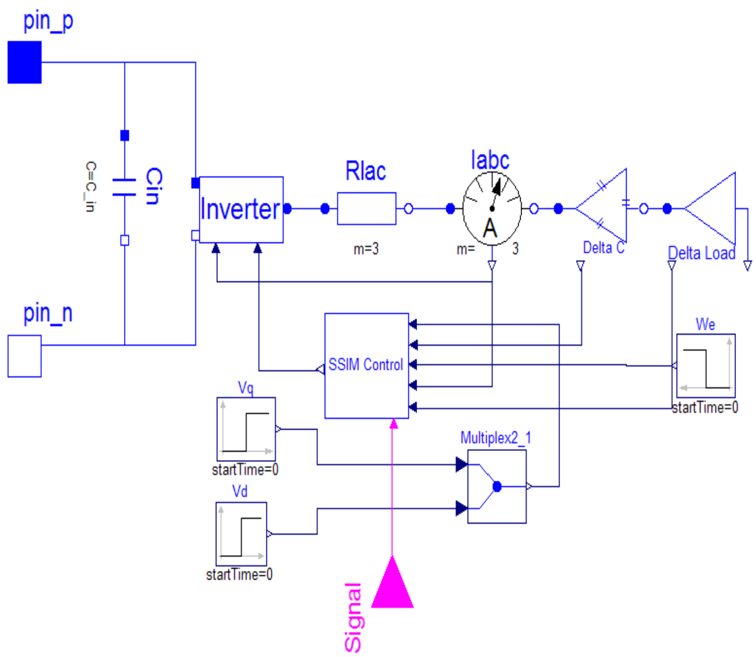


Figure 5.12: *Ship Service Inverter Module*

chronous reference frame and the effective capacitance of the output delta filter. These currents are added to the transformed measured currents given by equations 5.7 and 5.8

$$i_{qF}^{e*} = i_{qD}^{e*} + I_{qL} \tag{5.7}$$

$$i_{dF}^{e*} = i_{dD}^{e*} + I_{dL} \tag{5.8}$$

The advantage of using the decoupling feedback control is to eliminate the steady state error. The Figure 5.13 shows the controller block diagram along with the decoupling feedback control. The input to the controller are the reference voltages, the output capacitive filter voltages, the currents from the load model and the synchronous reference frame signal.

5.4.4 Motor Controller

The motor controller (MC) is a 5 hp induction motor drive and acts as load bank in Zone 2. The motor controller has an inverter connected to the induction motor shown in Figure 5.14. The input voltage to the inverter is 400 Volts DC. The load torque on the induction motor is given as square of the speed times a constant value $T_L = 2.8384e^{-4} \times \omega^2$.

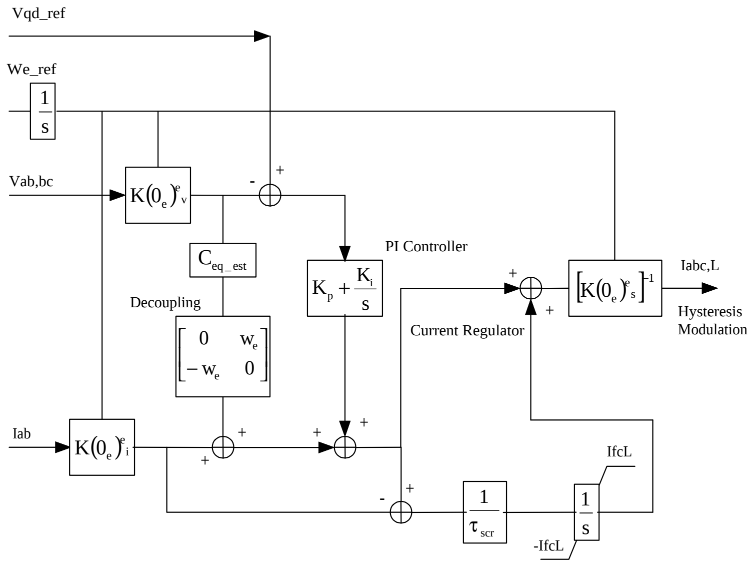


Figure 5.13: *SSIM Controller*

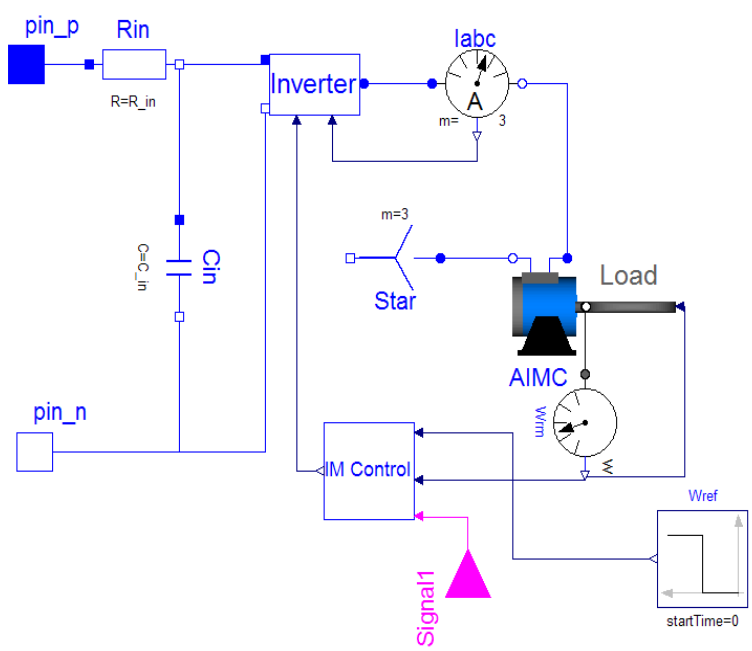


Figure 5.14: *Motor Controller*

The controller **MC_Control** designed for the induction motor is based on a speed reference input indirect field-oriented control explained in [23]. According to the control theory the torque is maximized during transient and steady states when the rotor flux and currents are perpendicular to each other. The equation (5.9) shows the rotor current and flux relation.

$$\lambda'^{e}_{qdr}.i'^{e}_{qdr} = \lambda'^{e}_{qr}.i'^{e}_{qr} + \lambda'^{e}_{dr}.i'^{e}_{dr} \tag{5.9}$$

The rotor current and flux are made orthogonal to each other using the conditions given by equation (5.10). These conditions are achieved first by placing all of the rotor flux linkage in d-axis using a suitable choice of reference frame and second by making the d-axis rotor current constant.

$$\lambda'^{e}_{qr} = 0;\ i'^{e}_{dr} = 0 \tag{5.10}$$

According to the control theory, the current control is achieved by comparing the reference currents $I_{abc,ref}$ with the measured currents using a hysteresis comparator. The reference currents are generated by transforming the q-axis and d-axis current components along the synchronous reference frame. The q-axis and d-axis currents along with reference frame are given by equations (5.11)-(5.14).

$$I_{qs} = \frac{3P}{4}\frac{T^{*}_{em}\omega_b X_{rr}}{X_m \psi_{ib}} \tag{5.11}$$

$$I_{ds} = \frac{\psi_{ib}}{X_m} \tag{5.12}$$

$$\omega_e = \omega_r + \omega_s \tag{5.13}$$

$$\omega_s = \frac{\omega_b R_r I_{qs}}{X_{rr} I_{ds}} \tag{5.14}$$

From the equations it can be seen that to maximize the torque during transients and steady state the d-axis current components is made constant to the ratio of flux to mutual impedance. The computed q-axis current component depends on the electromechanical torque. The reference torque is generated by comparing the measured speed and the reference speed input using a PI controller. The controller block diagram for the induction motor is shown in Figure 5.15. The parameters to the controller are the reference speed, and the PI controller values.

5.4.5 Constant Power Load

The Constant Power Load (CPL) module shown in Figure 5.16 acts as a load bank in Zone 3. The Constant Power Load model is a classical DC-DC Buck converter with a resistive load R_load to

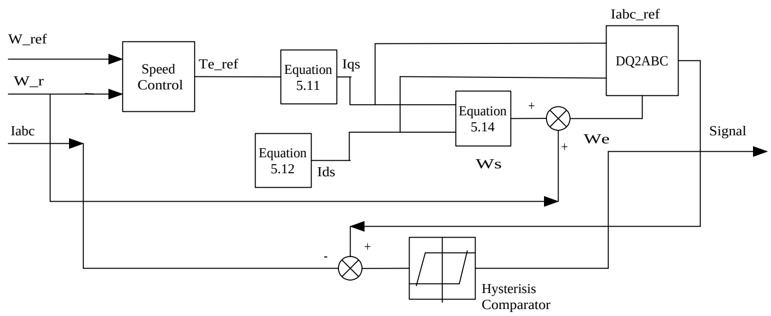


Figure 5.15: *Induction Motor Control*

maintain the output power at a rated value P_ref. The input voltage to the converter varies from 380-440 V DC. The load on the converter is $R\_load = 2\ \Omega$ and the rated output power is P_ref = 5 kW giving an output voltage of 100 V for the converter.

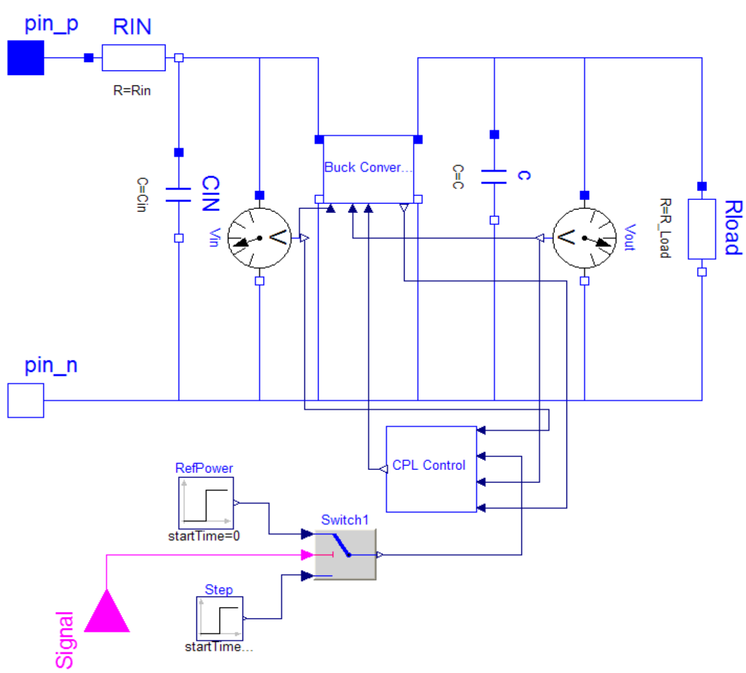


Figure 5.16: *Constant Power Load*

The controller block diagram of the CPL is shown in the Figure 5.17. The inputs are the measured input voltage V_in of the converter, reference output power P_ref, load voltage V_L and load current I_L. The reference power is first modulated by a Non-linear stabilizing control (NSC). The command current generated by the measured load voltage and the modulated reference power is used for hysteresis current control.

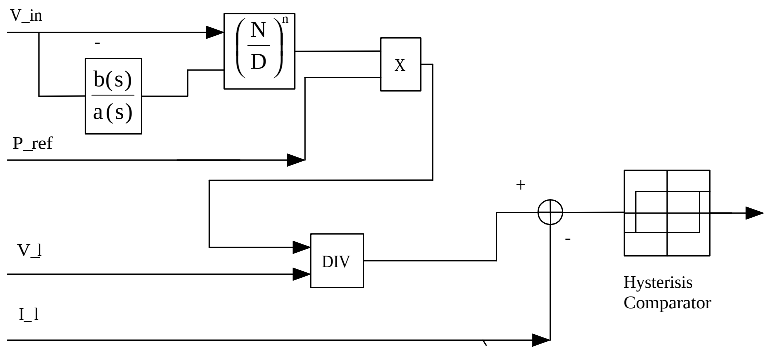


Figure 5.17: *Constant Power Load Controller*

5.4.6 Zones

The **Zones** library is a part of the DCDTB library. In the Zones library all the three zones of the shipboard power system are modeled. Each zone modeled is named as **Zone1**, **Zone2**, and **Zone3**. Each Zone model has a pair of SSCM modules each connected to the starboard side and the Port. The output of the converter modules is connected to the load bank by an Or ring diode circuit where in two diodes are connected to the output positive pins of the SSCM and connected to the load bank. The converter with higher voltage rating supplies the load while the other remains as a back-up supply. Figure 5.18 represents the block diagram of Zone 1. The block diagrams of Zone 2 and Zone 3 are similar to Zone 1 except the loads, which are different for each zone.

The input boolean signals present in the model are used to activate and deactivate the converters and load modules. The output boolean signals send signals to the circuit breakers to indicate a fault. The inputs of the converter modules are connected to the DC bus on either side using a pair of electrical pins.

5.5 Examples

The working of the components of shipboard power system library is demonstrated in the **Examples** library. The Examples library is made as a part of the SPS library. The Examples library is into three categories and modeled as three different libraries. The libraries are **Test_ACSYS**, **Test_PS**, and **Test_DCDTB**. Each library tests all the components present in each system. All the test models are

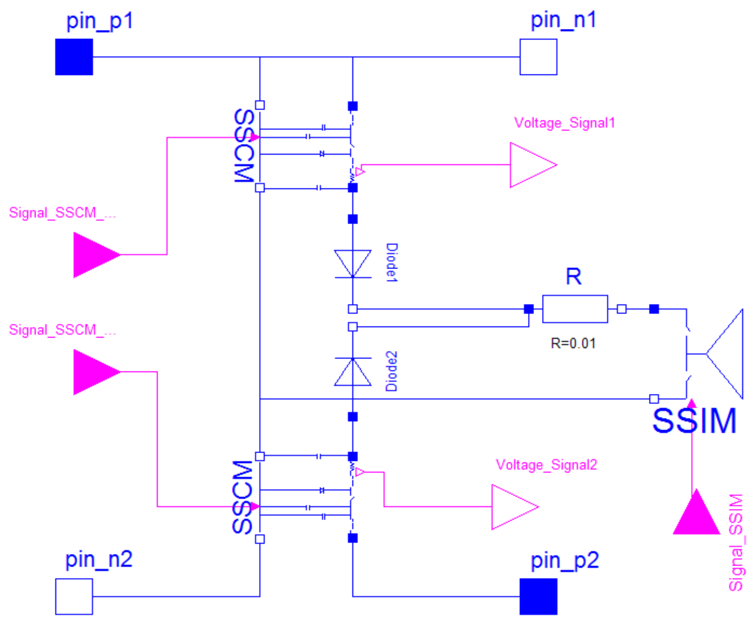


Figure 5.18: *Zone 1*

simulated for time **t = 1 sec**

5.5.1 Test_ACSYS

The Test_ACSYS tests all the models developed in the AC Generation and Propulsion System. Three components are modeled to test the AC system.

Test_ACGenerator: The synchronous machine modeled acts as generator for the shipboard power system. The synchronous machine is tested in this model. The generator is run at no-load to generate the rated voltage of the machine. The prime mover connected to the machine is a constant speed prime mover. The Figure 5.19 shows the no-load voltage generated by the generator. It can be seen that the voltage generated starts from zero and increases to reach the rated value of the machine after certain time of the simulation.

Test_PropulsionSystem: The input voltage to the propulsion system is 460 V, given by an ideal AC voltage source. A load torque of T_Load = 60 N.m. Load is applied on the system at time t = 0.1 s. Figure 5.20 shows the speed of induction motor drive. The speed of the machine starts at rated speed at the start of simulation, decreases gradually and comes back to the rated speed after

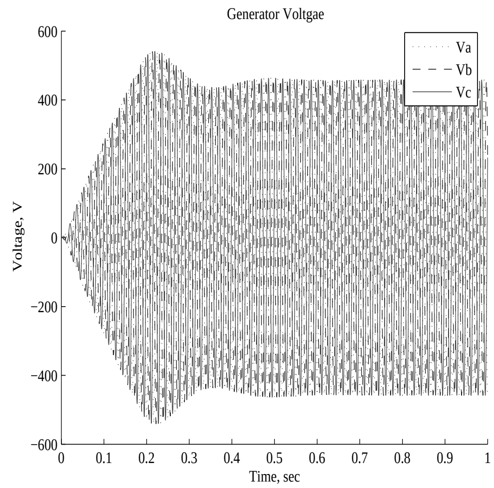


Figure 5.19: *Generator Voltage*

a certain time of simulation.

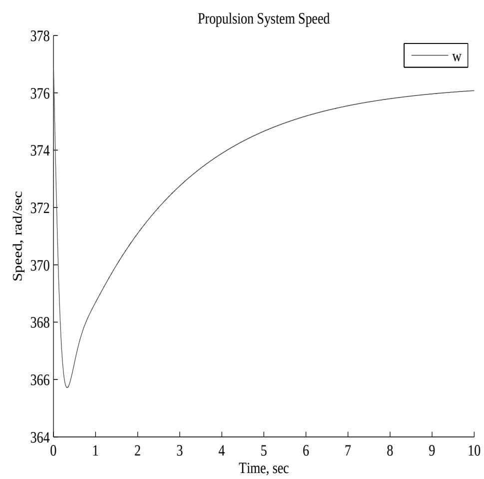


Figure 5.20: *Propulsion System Induction Motor Speed*

Test_ACSYS: The model Test_ACSYS is modeled to test the AC generation and propulsion system. The model has the AC generator model **ACGenerator** connected to the **PropulsionSystem** model using the AC bus model **HF**. An additional switch is used to connect the AC generator and propulsion system. The switch is used to allow the voltage generated from to reach the rated value reach Figure 5.21 shows the AC system. The switch is closed at time instant t = 0.1 s to supply the voltage to the propulsion system. The rated torque T_{rated} = 60 N.m is applied to the system at time t = 0.2 s. Figure 5.22 shows the voltages and currents of the generator. The voltage and current of the generator increases gradually till the load is applied on the system . The moment load is applied both the voltage and current increase instantaneously and reach steady state values of 460 V and 20 A respectively.

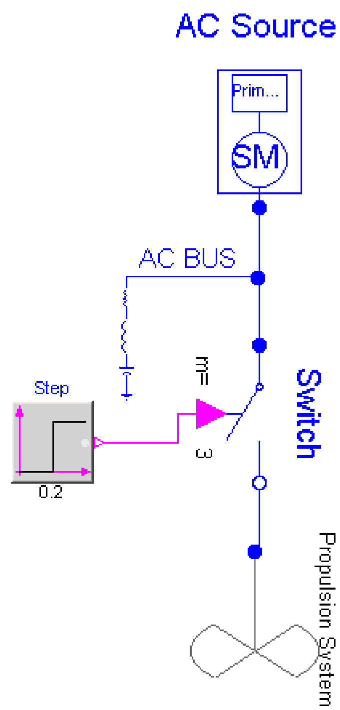


Figure 5.21: *AC Generation and Propulsion System*

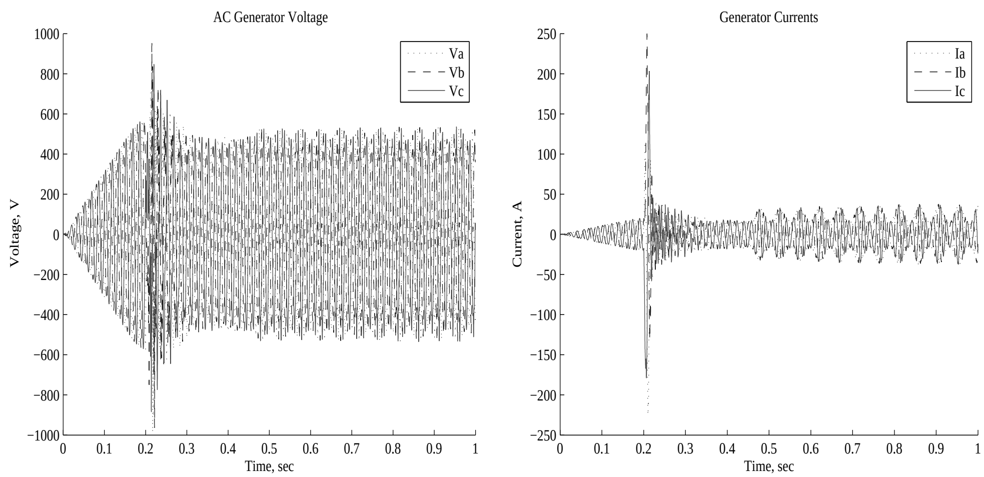
5.5.2 Test_PS

The power supply module is tested by applying an input voltage of 460 V line-line rms, and for a resistive load of R_{load} = 20 Ω. The Figure 5.23 shows the output voltage and current waveforms of the power supply module. The output voltage of the power supply module starts from zero at the start of simulation time and reaches the rated voltage very quickly.

5.5.3 Test_DCDTB

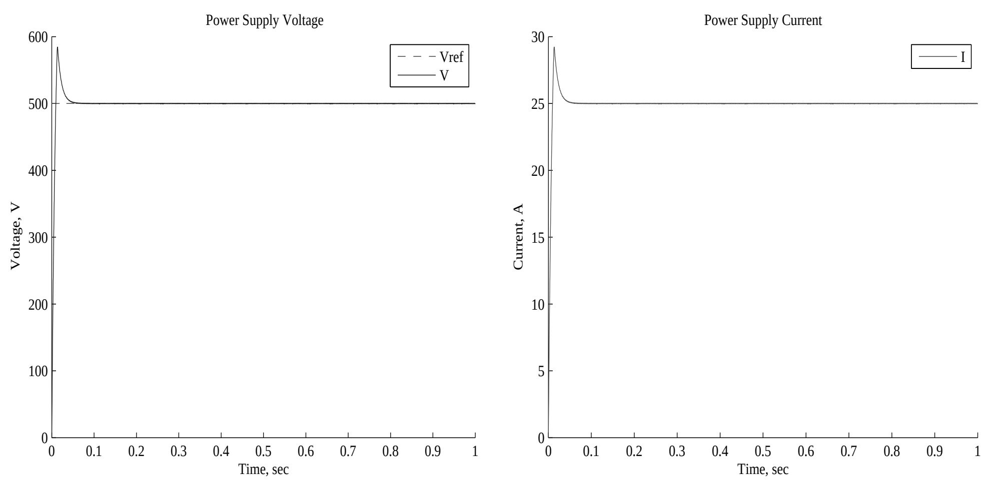
The Test_DCDTB library tests the models developed in the DCDTB library.

Test_SSCM: The working of the SSCM model is tested in this model. The converter module is



(a) ***Generator Voltage*** **(b)** ***Generator***

Figure 5.22: *AC Generator Voltages and Currents*

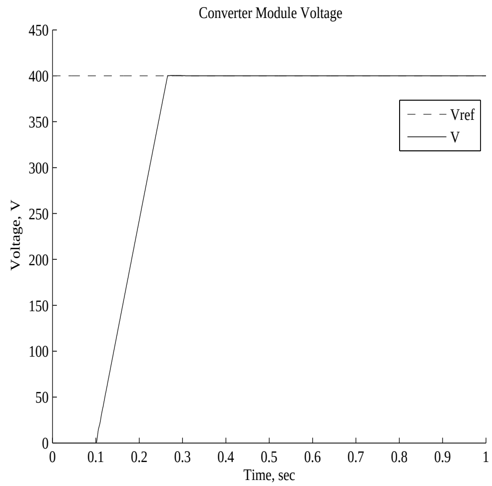


(a) ***PS Voltage*** **(b)** ***PS Current***

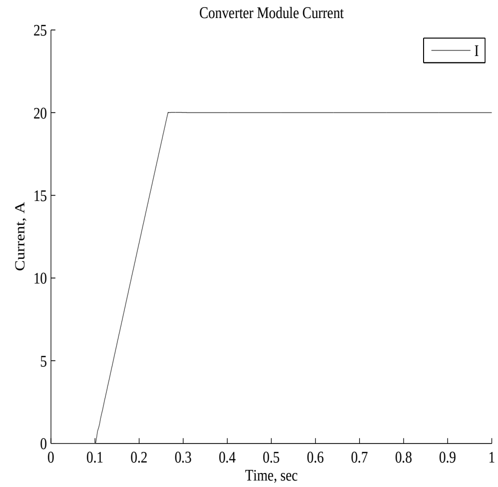
Figure 5.23: *Power Supply Voltages and Currents*

connected to a DC voltage source of 500 V and the output is connected to a resistor R_Load = 20 Ω. The converter is turned on by a boolean step signal at time t = 0.1 sec. The converter is simulated for a reference voltage of V_{ref} = 400 V. Figure 5.24(a) shows the reference voltage and the output voltage of the converter module, the voltage of the converter module is zero initially and gradually increases and reaches the reference voltage. Figure 5.24(b) shows the current through the load resistor.

Test_SSIM: The working of the inverter module is tested by using a varying DC supply input

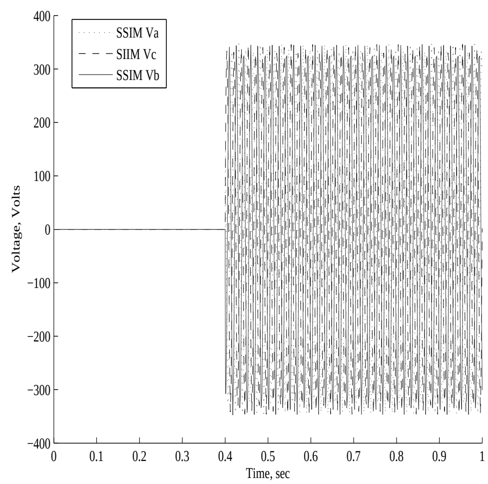


(a) ***SSCM_Voltage***

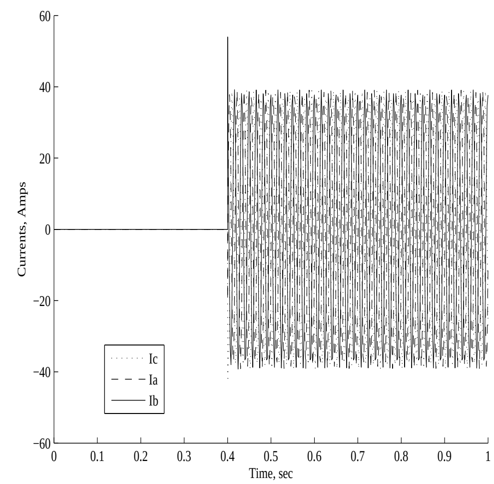


(b) ***SSCM_Current***

Figure 5.24: *Converter Module Voltages and Currents*



(a) ***SSIM_Voltage***



(b) ***SSIM_Current***

Figure 5.25: *Inverter Module Voltages and Currents*

voltage source. The converter is turned on at time t = 0.3 sec. The voltage and current waveforms of the inverter load are shown in Figure 5.25.

Test_MC: The motor controller is tested by supplying an input DC voltage of 400 V. The model is simulated for time t = 1.5 sec. The converter is turned on at time t = 0.1 sec. The speed of the induction motor is shown in Figure 5.26, the speed remains zero till the converter is off and increases and reaches the steady rated speed at around time t = 1.2 sec when the converter is turned on.

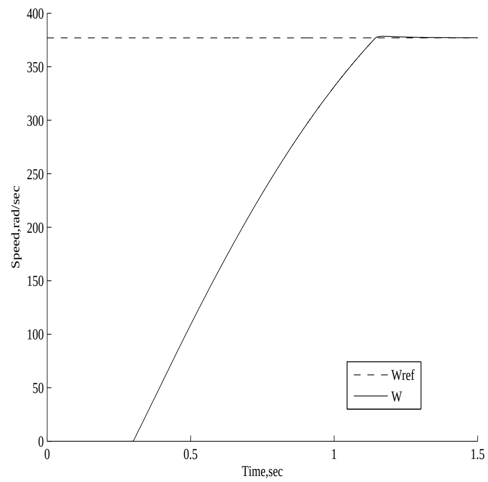
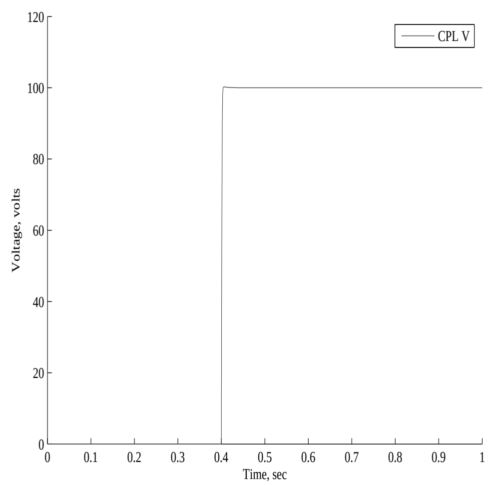
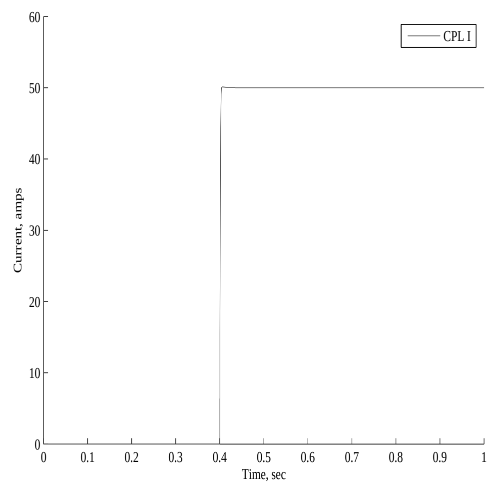


Figure 5.26: *Motor Controller Speed*



(a) ***CPL Voltage***



(b) ***CPL Current***

Figure 5.27: *Constant Power Load Voltages and Currents*

Text_CPL The constant power load model is tested for constant power of P = 5 kW and a resistive load R_{load} = 2 Ω. The voltage waveform of the output voltage of the converter (Figure 5.27) shows that the voltage reaches 100 V the moment the converter is turned on giving a constant output power of 5 kW.

Test_Zones: The Test_Zones tests the models Zone 1, Zone 2, and Zone 3. Each zone is connected to the power supply module and an ideal AC voltage sources both on starboard and Ports. The converter modules and the loads are turned on at time instants t = 0.1 sec and t = 0.3 sec

Test_DCDTB: The model Test_DCDTB tests the DC distribution system as a whole and the model is shown in the Figure 5.28. The simulation is run for time t = 1 sec. The converters present in the zones are turned on at time t = 0.1 sec and the loads in zone 1 and zone3 are turned on at t = 0.4 sec, while the load in zone 2 is turned on time t = 0.3 sec. Figure 5.29 show the voltages and currents of the power supply modules. The voltages of the power supply reach a maximum of 600 V and decrease once the load is applied and reaches a steady state. Observation of the current and voltage waveforms of the converter modules of all the zones shown in Figures 5.30 - 5.32 shows the converter module with higher voltage supply power to the loads and the other converter remains on standby.

5.6 Power Electronics Library

The **PowerElectronics** library has been modeled as a part of the SPS. The library consists of the basic models used for modeling the components of the shipboard power system. The PowerElectronics library is divided into different sub-libraries explained in the following sections.

5.6.1 Devices

The **Devices** library has the power electronics models used for modeling the components of the shipboard power system. The models developed are the buck converter and the 3-phase inverter. The average value models of the converters have been modeled to avoid the switching events and to make simulation run faster.

Buck Converter: A buck converter is a DC-DC converter which converts DC voltage from one level to the other. Converters are also known as DC equivalents of AC transformers due to their behavior. Some of the most common applications of the Buck converters are regulating the DC voltage to a desired level in switch-mode power supply and DC motor drive applications. The Buck Converter is modeled by **BuckConverter** and the graphical annotation is shown in Figure 5.33(a).

Inverters: Inverters are used for converting the DC input supply to AC. Ideal inverters produce sinusoidal waveforms but practically the output of the inverters is non sinusoidal and involves harmonics and some are square wave inverters. Some of the applications of the inverters include switch mode inverters used in the control of ac drives.The inverter is modeled as **Inverter** and the graphical annotation is shown in Figure 5.33(b).

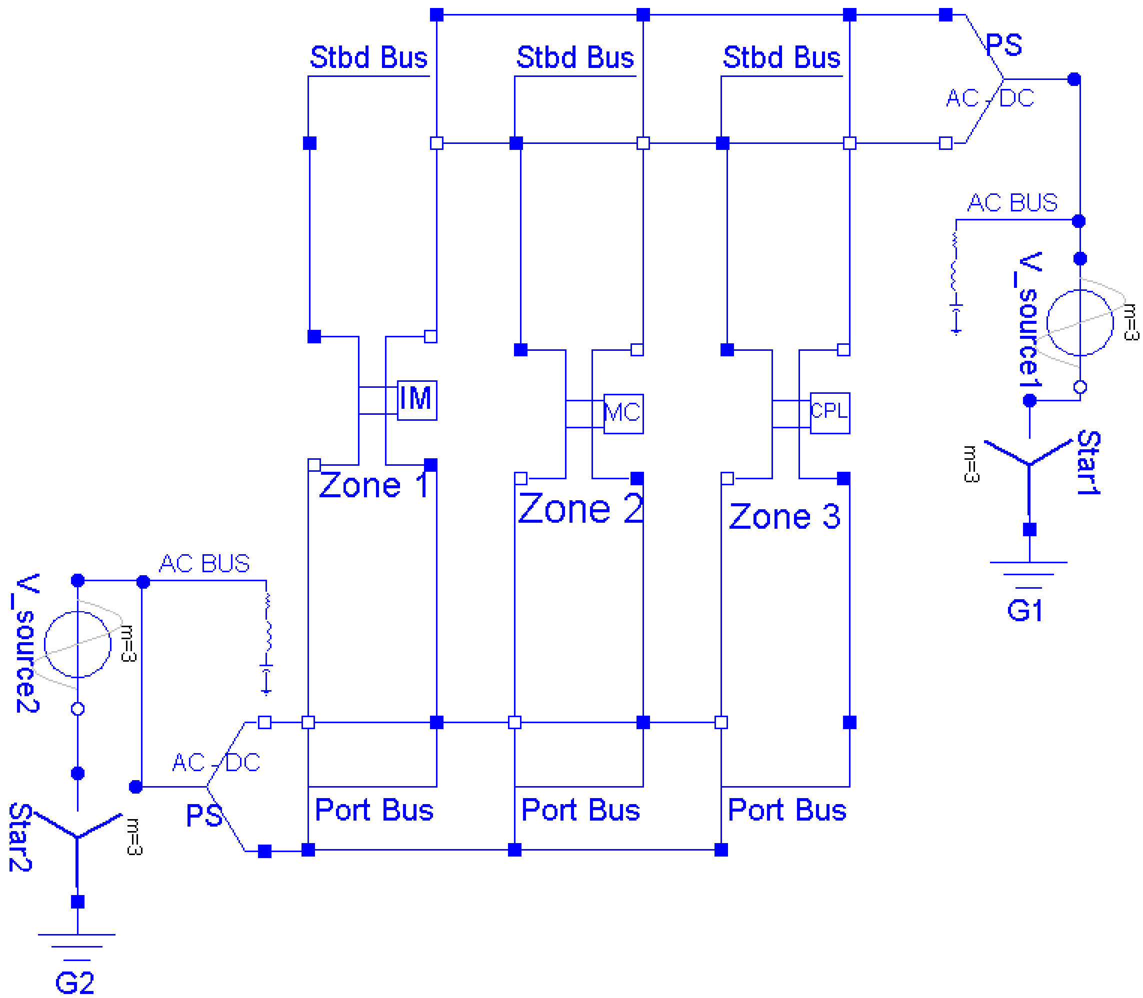
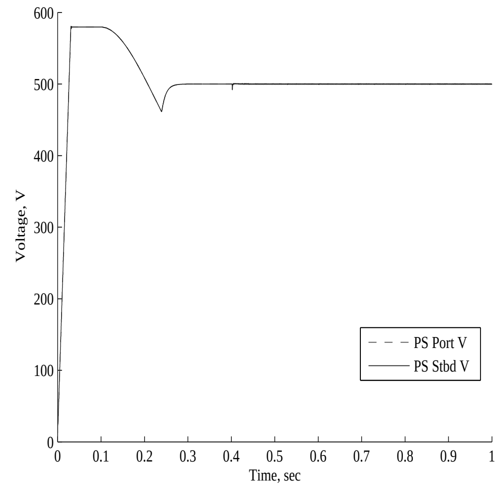


Figure 5.28: *DC Distribution System Test Bed*

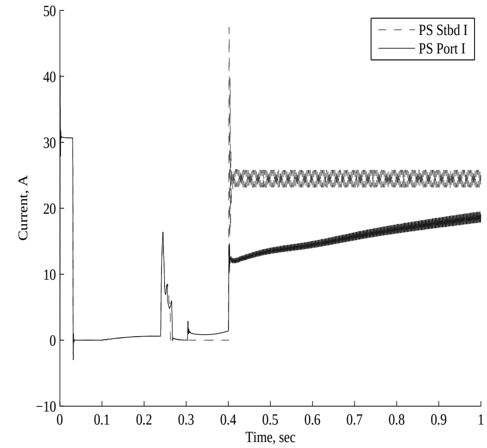
5.6.2 Controls

The **Controls** library provides control algorithms for the converters modules present in the shipboard power system. The models developed include the PI controller and mathematical transformations used in the advanced control of electric drives.

PI Controller: PI controller (Proportional Integral controller) modeled as **PIControl** is one of the most common feedback loop controller used for performing simulations. The controller compares a measured value from a process with a reference set point value. The difference or error signal is then processed to calculate a new value, which brings the measured value to the reference set point

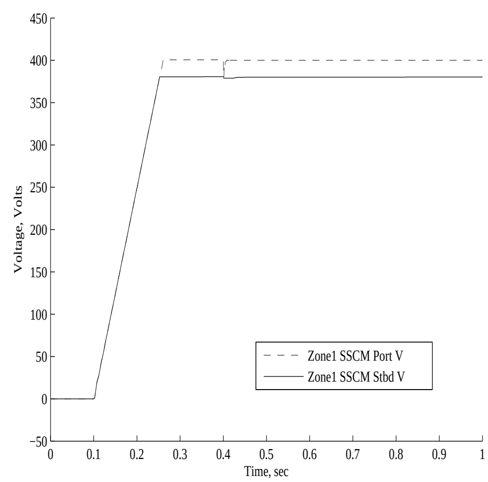


(a) ***PS Voltage***

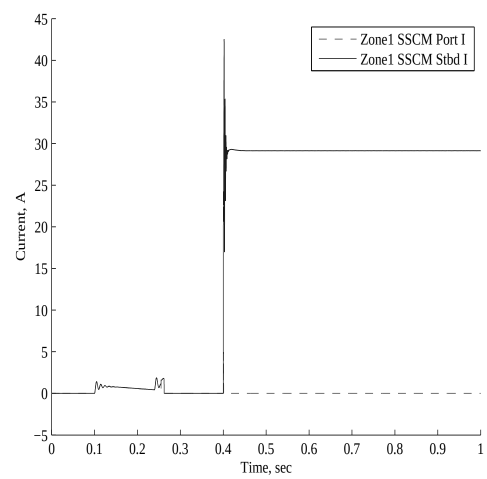


(b) ***PS Current***

Figure 5.29: *Power Supply Voltages and Currents*



(a) ***Zone 1 SSCM Voltage***

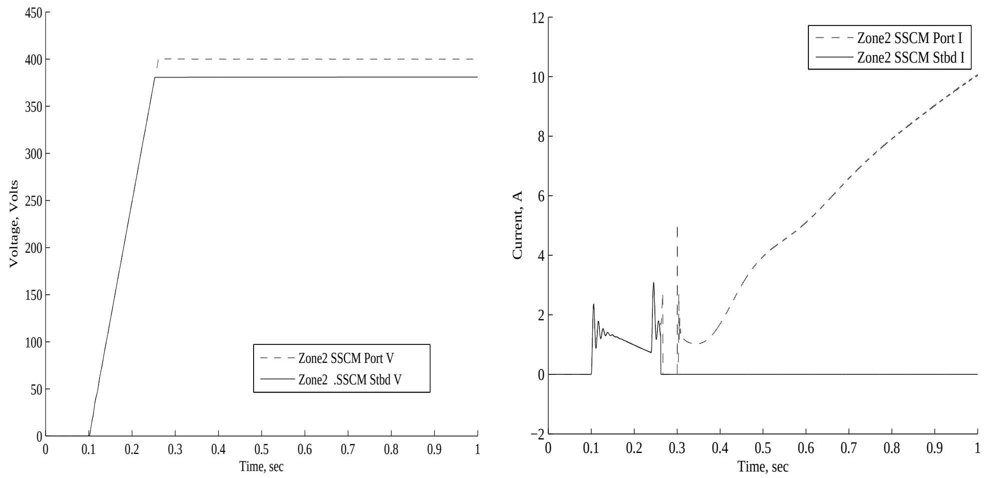


(b) ***Zone 1 SSCM Currents***

Figure 5.30: *Zone 1 SSCM Voltages and Currents*

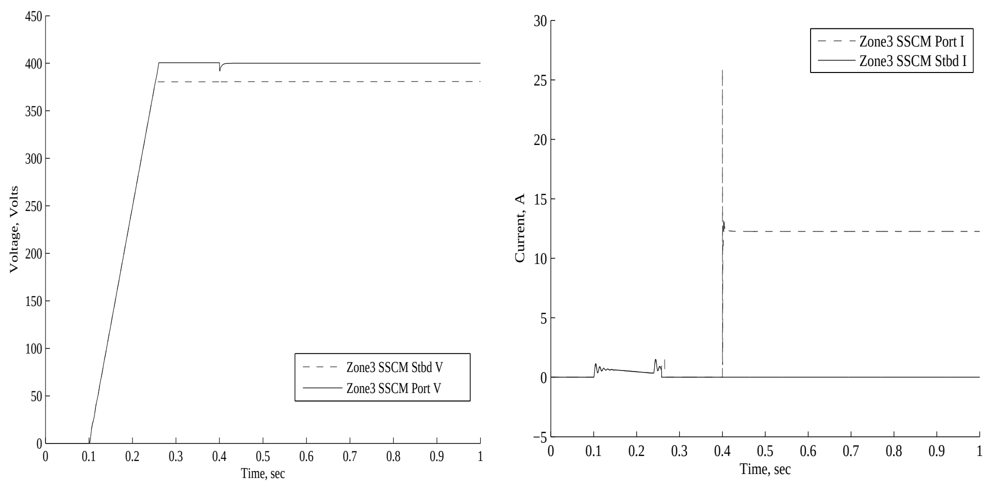
value. The PI controller modeled is shown in Figure 5.34. The parameters to the controller are K_p proportional gain and K_i integral gain, and the maximum and minimum values for the limiter.

Transformations: Mathematical transformations used are known as Park's transformations [23]. They are useful in machine analysis and advanced control of electric drives. The transformations convert 3ϕ instantaneous time varying quantities to direct and quadrature axes along with the zero quantity referred to a reference frame. The change of variables from 3ϕ to d-q axis along a reference



(a) ***Zone 2 SSCM Voltage*** **(b)** ***Zone 2 SSCM Currents***

Figure 5.31: *Zone 2 SSCM Voltages and Currents*



(a) ***Zone 3 SSCM Voltage*** **(b)** ***Zone 3 SSCM Currents***

Figure 5.32: *Zone 3 SSCM Voltages and Currents*

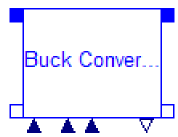
frame is given as

$$\mathbf{f}_{qdos} = \mathbf{K}_s \mathbf{f}_{abcs} \tag{5.15}$$

where

$$\left(\mathbf{f}_{qdos}\right)^T = \left[\begin{array}{ccc} f_{qs} & f_{ds} & f_{0s} \end{array} \right] \tag{5.16}$$

$$\left(\mathbf{f}_{abcs}\right)^T = \left[\begin{array}{ccc} f_{as} & f_{bs} & f_{cs} \end{array} \right] \tag{5.17}$$



(a) ***Buck Converter***

(b) ***Inverter***

Figure 5.33: *Converters*

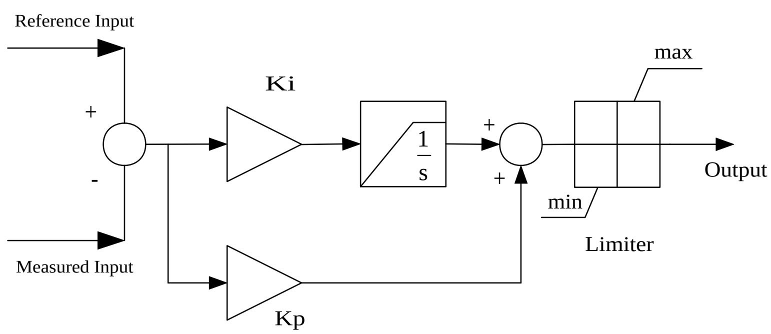


Figure 5.34: *PI Controller*

$$\mathbf{K}_s = \frac{2}{3}\begin{bmatrix} \cos\theta & \cos\left(\theta-\frac{2\pi}{3}\right) & \cos\left(\theta+\frac{2\pi}{3}\right) \\ \sin\theta & \sin\left(\theta-\frac{2\pi}{3}\right) & \sin\left(\theta-\frac{2\pi}{3}\right) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{5.18}$$

$$\omega = \frac{d\theta}{dt} \tag{5.19}$$

Equation (5.18) is the transformation matrix and its inverse is used for transforming the d-q components to instantaneous quantities. The transformation is given as

$$\mathbf{f}_{abcs} = (\mathbf{K})^{-1}\mathbf{f}_{qdos} \tag{5.20}$$

where the inverse transformation is given as

$$(\mathbf{K}_s)^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 1 \\ \cos\left(\theta-\frac{2\pi}{3}\right) & \sin\left(\theta-\frac{2\pi}{3}\right) & 1 \\ \cos\left(\theta+\frac{2\pi}{3}\right) & \sin\left(\theta+\frac{2\pi}{3}\right) & 1 \end{bmatrix} \tag{5.21}$$

The 3ϕ instantaneous components can be either currents or voltages. Three transformation models have been implemented. The transformation from abc to dq0 is modeled by abc2dq0 and the inverse transformation is modeled in dq02qbc. Apart from these models, abbc2dq0 transformation model given by the equations below is available.

$$\mathbf{f}_q = \mathbf{f}_{ab}\frac{2}{3}\cos\theta + \mathbf{f}_{bc}\left(\frac{1}{3}\cos\theta + \frac{1}{\sqrt{3}}\sin\theta\right) \tag{5.22}$$

$$\mathbf{f}_d = \mathbf{f}_{ab}\frac{2}{3}\cos\theta + \mathbf{f}_{bc}\left(\frac{1}{3}\sin\theta - \frac{1}{\sqrt{3}}\cos\theta\right) \tag{5.23}$$

The transformations available are with respect to an arbitrary reference frame. The transformations can be referred to a particular reference frame and, accordingly, the value of ω changes and also the notation of the components change. The different reference frames present are given in Table 5.1.

Table 5.1: *Different Reference Frames*

Reference Frame	Speed	Notation
Stationary Reference Frame	0	$\mathbf{f}^s_{qd0s}, \mathbf{K}^s_s$
Rotor Reference Frame	ω_r	$\mathbf{f}^r_{qd0s}, \mathbf{K}^r_s$
Synchronously Rotating Reference Frame	ω_e	$\mathbf{f}^e_{qd0s}, \mathbf{K}^e_s$

5.7 Machines

The shipboard power system has two different types of AC machines. The asynchronous machine for the propulsion system and the motor controller, and the synchronous machine for generating power for the ship. The asynchronous machine model from the Machines library available in Modelica standard library is used for modeling the propulsion system and the motor controller.

The synchronous machine is modeled as **SynchronousMachine**. The machine modeled is based on the reference frame or space phasor theory explained in [23] and the Machines library explained in Section 4.1.3. The stator side of the synchronous machine has 3ϕ symmetrical windings of resistance R_s and N_s turns and the rotor has field winding (f_d) with R_{fd} resistance and N_{fd} windings and 3 damper windings K_{q1}, K_{q2} aligned along the q-axis and K_d aligned along the d-axis. The synchronous machines equations when expressed in rotor reference frame are given as

$$v_{qs}^{r} = -r_s i_{qs}^{r} + \omega_r \lambda_{ds}^{r} + p\lambda_{qs}^{r} \tag{5.24}$$

$$v_{ds}^{r} = -r_s i_{ds}^{r} - \omega_r \lambda_{qs}^{r} + p\lambda_{ds}^{r} \tag{5.25}$$

$$v_{os} = -r_s i_{0s} + p\lambda_{0s} \tag{5.26}$$

$$v_{kq1}^{`r} = -r_{kq1}^{'} i_{kq1}^{'r} + p\lambda_{kq1}^{'r} \tag{5.27}$$

$$v_{kq2}^{`r} = -r_{kq2}^{`} i_{kq2}^{'r} + p\lambda_{kq2}^{'r} \tag{5.28}$$

$$v_{fd}^{`r} = -r_{fd}^{`} i_{fd}^{'r} + p\lambda_{fd}^{'r} \tag{5.29}$$

$$v_{kd}^{`r} = -r_{kd}^{`} i_{kd}^{'r} + p\lambda_{kd}^{'r} \tag{5.30}$$

The equation for the electromagnetic torque in terms of the rotor reference frame is given as

$$T_e = \left(\frac{3}{2}\right)\left(\frac{P}{2}\right)\left(\lambda_{ds}^{r} i_{qs}^{r} - \lambda_{qs}^{r} i_{ds}^{r}\right) \tag{5.31}$$

The synchronous machine is modeled using the equations (5.24) - (5.30) and the components of Machines library as shown in Figure 5.35.

The model is similar to the induction motor model shown in Figure 4.6. The stator side of both the machines is modeled in a similar manner. The rotor of the synchronous machine is modeled using two components. First one is the **Damperwindings** modeled for K_{q1} and K_d damper windings using equations (5.27) and (5.30), and the second one is the **Fieldwinding** modeled for damper winding K_{q2} using equation (5.28). The Fieldwinding model has a pair of electrical pins connected to the field winding resistor R_{fd} and inductor L_{fd}. The DC excitation voltage is supplied by the external pins connected to the field winding resistor and the Fieldwinding model. The torque generated in the machine is calculated in the AirGap model using equation (5.31).

The input parameters to the Synchronous Machine model are the rated voltage, current, power, the frequency at which the power is generated, the number of poles of the machine, rotor inertia, resistance and inductance of the stator, field winding, and damper windings.

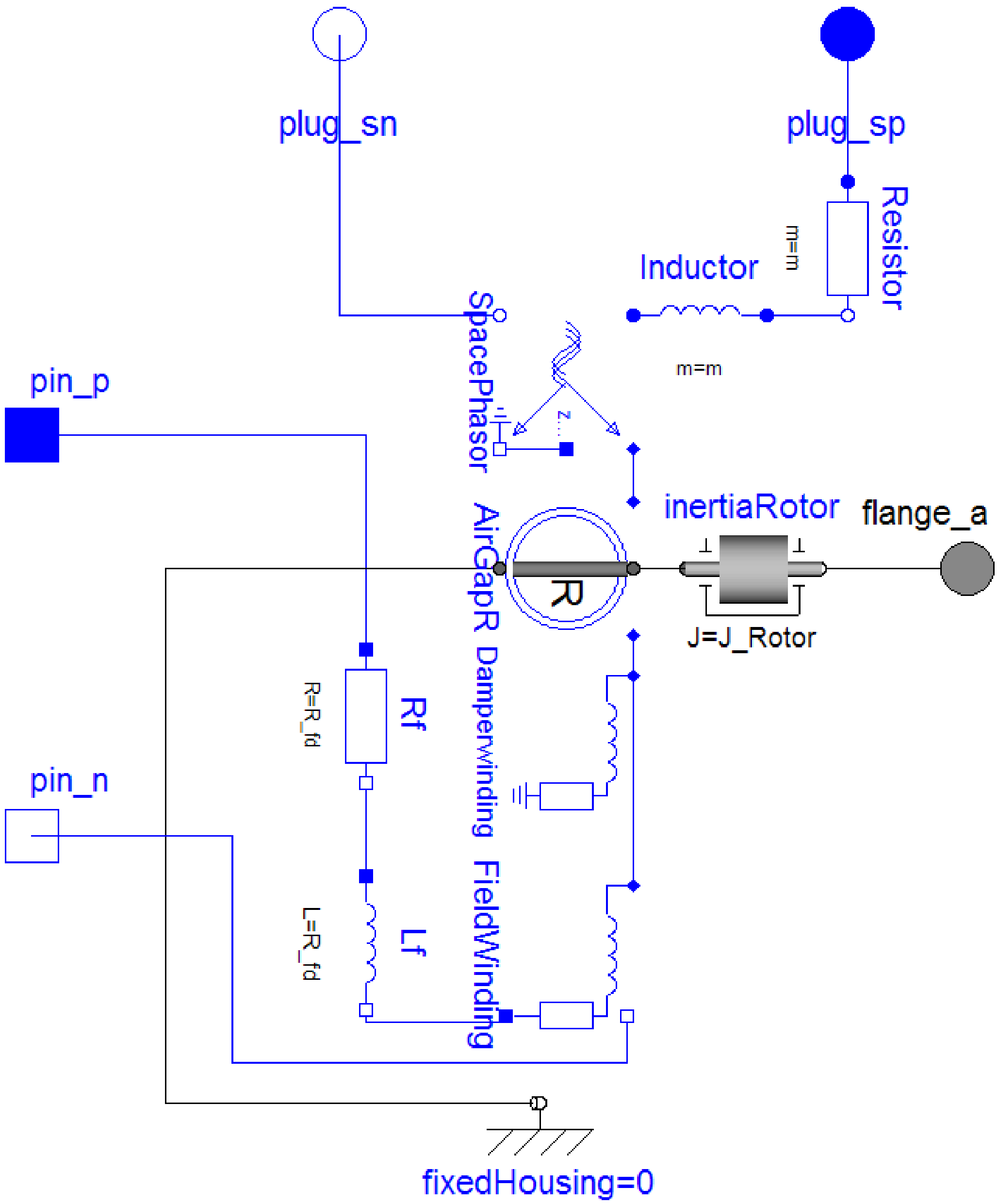


Figure 5.35: *Synchronous Machine*

CHAPTER

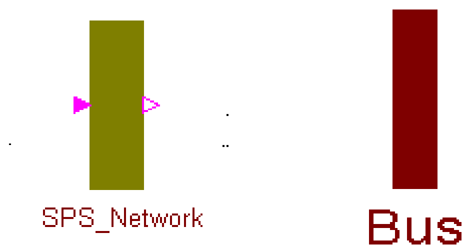
6

Networked Protection System

6.1 Introduction

As in the case of a utility power system, the shipboard power system can also incur faults. The protective system of the ship should be able to detect and clear the faults in no time and restore the supply back to normal operation. This is very important for the survivability of the ship.

This thesis develops a protective system called the Networked Protection System. The protection system developed detects and clears the faults that occur in the system. The faults are detected using the circuit breaker modeled. A controller has been designed as a part of the protection system to clear faults, control the converters and loads turn on/off, and control the breakers. The circuit breakers clear the fault when it recieves a signal from the controller clear the fault. The controller designed is a discrete hybrid event based controller modeled using Petri nets. The signals between the controller, circuit breakers, and the converters are transmitted using a communication network component developed. A time delay is introduced in the communication network between the inputs ans outputs. The reason behind introducing a delay in the network is, in real world the controller and the plant are physically at a distance from each other and this is called a distributed control system. The communication between the plant and the controller is not instant, a delay is introduced. To resemble this scenario and considering the system as a distributed control system the delay is



(a) ***SPS_Communication*** **(b)** ***Bus_Communication***

Figure 6.1: *Communication Network*

introduced.

A **ProtectiveSystem** library is developed in Modelica to implement the networked protection system. Different components are developed as a part of the library, a communication network, fault model, protective devices, and the command and control center. Two case studies are performed to prove the working of the protective system.

6.2 Communication Network

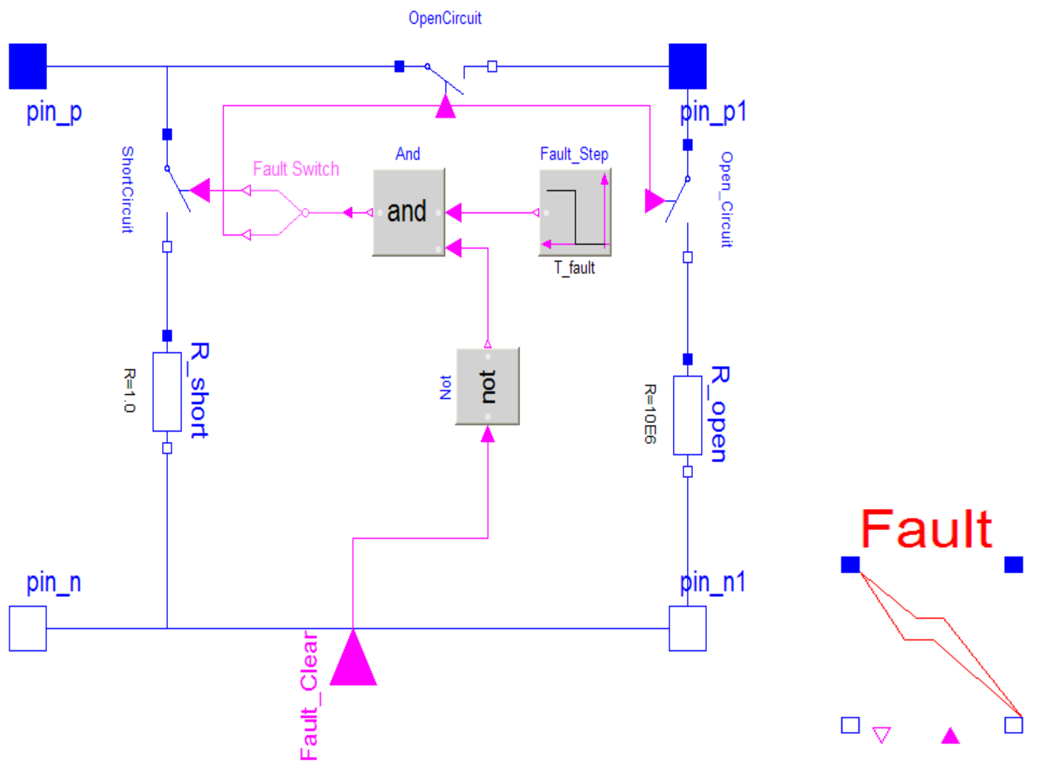
Two different communication networks are modeled in the library:

- **SPS_Communication**

  The model was developed for communication between the physical system to the command and control center. The network communicates the switch on/off signals for the converter modules and loads in the zones and also signals to the circuit breaker. A constant time delay **T_Delay** is introduced in the network between the input and output channels. The time delay T_Delay is given as parameters to the communication network. The default value for T_delay = 1 ms. The graphical annotation of the communication network is shown in Figure 6.1(a).

- **Bus_Communication**

  The model was developed for communication between the zone converter modules and the circuit breakers. No time delay is introduced in the network model as the network is assumed to be a separate network used specifically for communication between the zones and circuit breakers. The bus communication network is shown in the Figure 6.1(b).



(a) ***Fault_Model*** **(b)** ***Graphical Annotation***

Figure 6.2: *Fault Model*

6.3 Fault_Model

To create a fault in the system, a **Fault_Model** component is implemented. The model developed can be used to create a fault in DC system. Two types of faults can be created: open circuit and short circuit faults with adjustable resistance.

The Fault_Model is shown in Figure 6.2(a). An **IdealOpeningSwitch Open_Circuit** (initially closed) is connected in series with the DC circuit and an **IdealClosingSwitch Short_Circuit** (initially open) connected in parallel with the DC circuit. Depending on the type of fault, the **FaultSwitch** directs the input boolean signal to the control of Open_Circuit or Short_Circuit and the switches either **open** or **close** accordingly. The input to the FaultSwitch is a boolean signal which is true at the time of fault and false when the **Fault_Clear** signal from the controller becomes true. The inputs to the Fault_Model are the type of fault and the time **T_Fault** at which the fault occurs.

The graphical annotation of the Fault_Model is shown in Figure 6.2(b). The low impedance

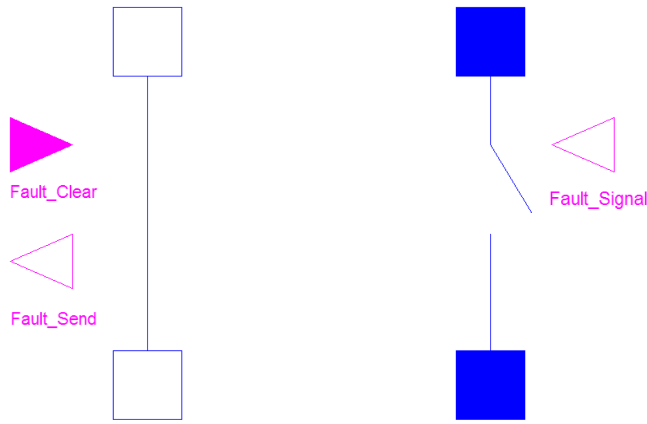


Figure 6.3: *Circuit Breaker*

R_Short and high impedance **R_Open** resistors are present in the model to limit the short circuit currents and modeling high-impedance to ground component.

6.4 Circuit_Breaker

The circuit breaker model developed is **Circuit_Breaker**. The Circuit_Breaker appears in the system on the DC buses connected to SSCM converter module in each zone and in total 6 circuit breakers are present in the system. The circuit breaker modeled is shown in Figure 6.3. The breaker has a current sensor connected in series and a voltage sensor connected in parallel to the circuit. Parameters of the circuit breaker is the rated power **P_rated** of the converter module. The circuit breaker has an IdealOpeningSwitch **Breaker** (initially closed) connected in series with the converter module. The limiting current **I_limit** of the breaker is calculated by the P_rated power and voltage **V** measured across breaker by the voltage sensor. The signal **Fault_Send** is connected to the controller and becomes true when the circuit breaker becomes active and the circuit breaker is again deactivated when **Fault_Clear** signal from the controller is true.

The circuit breaker becomes active when the measured current **I** exceeds the limiting current I_limit or the signal **Fault_Signal** from the converter modules.When an open circuit or short circuit fault occurs, the signals either **Short_Circuit** or **Open_Circuit** are enabled respectively. The **Fault_Signal** is from the converter module connected to the circuit breaker. The signal is generated when the input voltage to the converter is within a pre-specified range after the output voltage of the converter reaches the rated value.

6.5 Command and Control Center

To control the converters modules and the loads present in the zones, control the circuit breakers and generate signals to clear the faults a controller has been designed as **CCC** (Command and Control Center). The CCC acts as a supervisory controller and is designed using petri nets library in Modelica. The inputs to the command and control center are signals from the circuit breakers present on both the starboard DC bus and port DC bus.

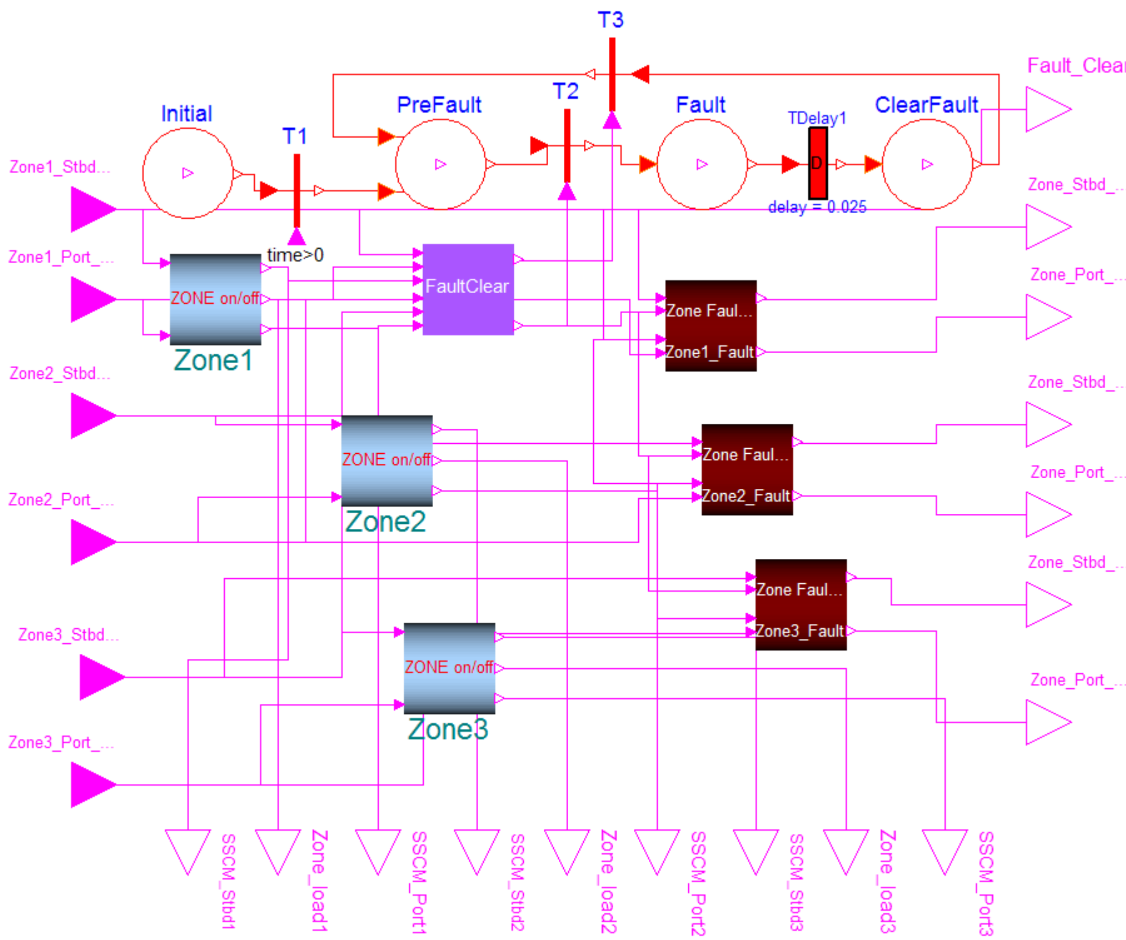


Figure 6.4: *Command Control Center*

Figure 6.4 shows the command and control center designed. The controller is divided into three sections

- Fault control section to generate fault clearing signals and includes places Initial, Prefault, Fault and ClearFault, zone control section, and breaker control section.

- Zone control section to generate signals to control the turn on/off the converters, and loads. The Zone control section is represented by components **Zone1**, **Zone2**, and **Zone3** in the command and control center.
- Breaker control section for generating signals for the circuit breakers present on the DC buses. **Zone1_Fault**, **Zone2_Fault**, and **Zone3_Fault** represent the breaker control section.

Figure 6.7(a) represents a flow diagram for the fault control section. At the time of start of simulation the state of Initial place is true and the states for remaining places is false. The moment simulation starts the the states of the initial and pre-fault change to false and true respectively by the firing of the transition T1 connecting initial and Pre-Fault. The state of Fault place is true when a fault occurs in the system and fault signal is generated by **FaultClear** model present in the command and control center. The inputs to the FaultClear model are signals from circuit breakers. Fault clearing signal is the state of Clear Fault place. The state of FaultClear place is true when the time delay transition **TDelay** fires after a time delay **T_Delay** has elapsed. The time delay transition has been made a part of the controller because, in a real world scenario the controller cannot generate the output instantaneously, a certain time delay is present. When a new fault occurs in the system the Fault Clear state changes to false and the the pre-fault state again becomes true.

Figure 6.5 shows the block diagram of zone control section present in Zone1, Zone2, and Zone3 components of the command and control center. The inputs are **Stbd** and **Port** signals from the circuit breakers present on both starboard and port DC buses connected to each zone. The input signals indicate the status of the breakers and the signals are true when fault occurs in the system. The output signals are **SSCM_Stbd**, and **SSCM_Port**, and **Zone_Load** to control the turn on/off the converters and loads. Figure 6.7(b) shows the flow diagram for converter turn on/off which is same for both starboard and port sides. The state of Initial is true and the states of remaining places is false. The state of SSCM is true when transition Zone_On fires after time greater than Zone_Turn_On. The state of SSCM1 changes from false to true and SSCM changes from true to false when the transition is fired when there is no fault or the fault is cleared in the system. The time delay transition fires after a pre-specified time delay and changes the state of SSCM on/off to true. The state of SSCM on/off changes to false if a false occurs in the system. The state of the place SSCM on/off is used to turn on/off the converters. The output **Zone_load**, to turn on the loads is true when the transition **Load_Turn_On** is fired at the time instant **Load_Time_On**. The time delay transition is used in the controller to induce a time delay to allow the capacitors connected to the power supply module to be

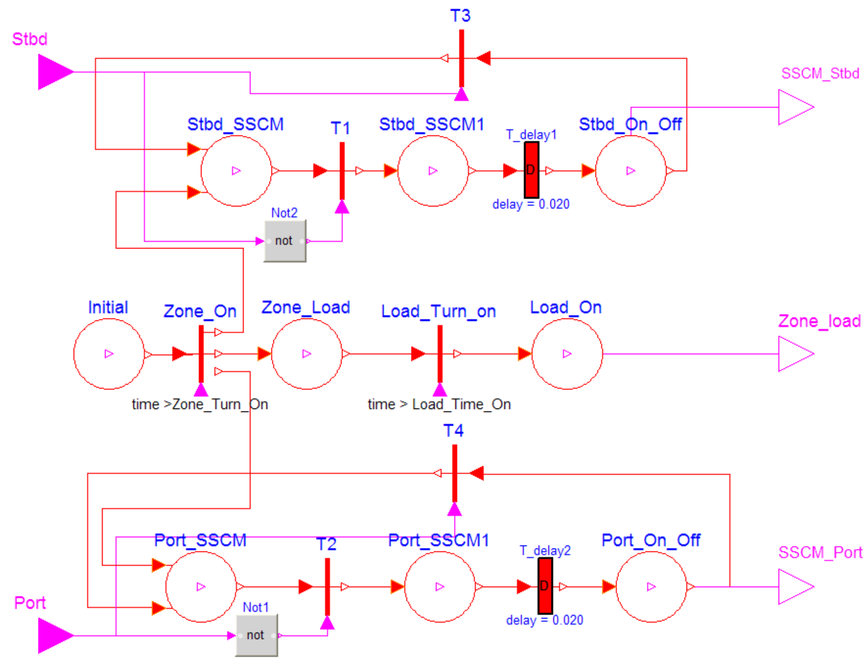


Figure 6.5: *Zone Controls*

charged after the fault has been cleared and before the load is applied.

The block diagram of the components present in the breaker control section of the command and control center is shown in Figure 6.6. All the three models of the breaker control section have the same structure. There are four inputs signals two of which are from the circuit breakers present on the DC buses connected to each zone, one signal is the fault signal from the FaultClear model, and the other one is fault clearing signal generated in the fault control section. The outputs **Zone_Stbd_Clear** and **Zone_Port_Clear** are signals to control the circuit breakers. Figure 6.7(c) shows the flow diagram of breaker control section. At the start of simulation the state of Initial is true and the states of remaining places is false. The state of the Breaker place changes to true the moment simulation starts and remains true till a fault in the system. The fault in the system is indicated by the signals from the breakers and changes the states of places Breaker from true to false and Fault from false to true. The state of Clear place is true when the transition connecting the Breaker and Clear is fired by the fault clearing signal.

The input parameters to the command and control center are the time instants SSCM_Zone1, SSCM_Zone2, and SSCM_Zone3 to switch on converter modules and Load_Zone1, Load_Zone2, and Load_Zone3 to switch on the load modules in the zones.

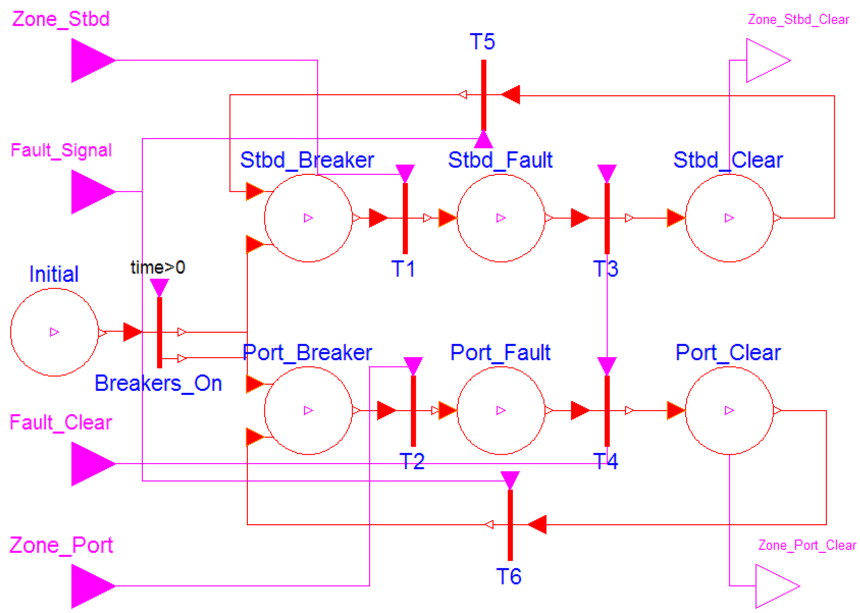


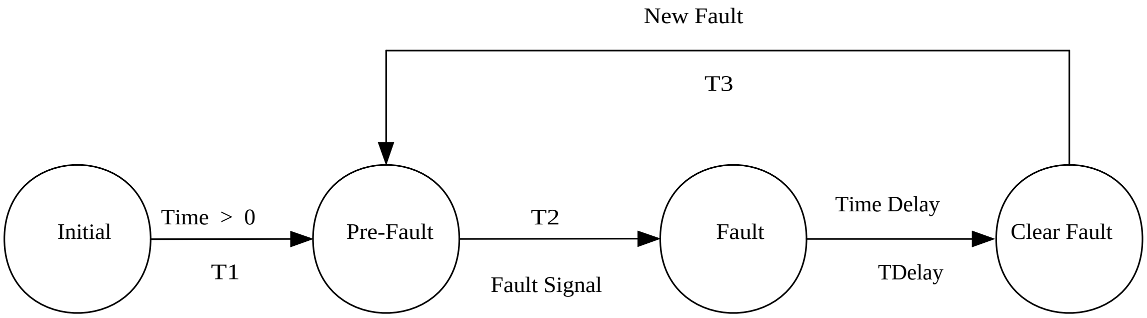
Figure 6.6: *Zone Fault Clear*

6.6 Case Studies

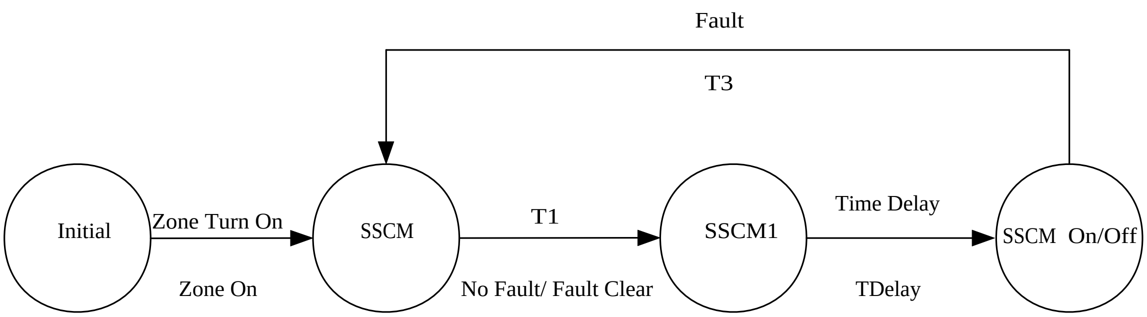
Two case studies are performed on the DC distribution system test bed to test the effectiveness of the protection System. In each case study a different type of fault is applied on the system. The Figure 6.8 shows the test system block diagram. The circuit breakers in the system are placed on the DC buses on both starboard and Ports. The signals between the circuit breakers and the command control center are transmitted through the communication network SPS_Communication. The communication between the converters and the circuit breakers is done using the Bus_Communication network. The Fault occurs near the output of the power supply module on the starboard side in both the cases.

6.6.1 Open Circuit Fault Test

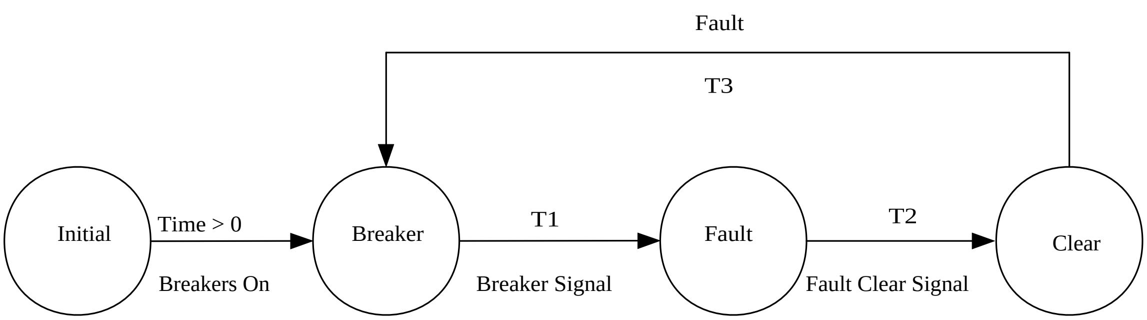
In this test case an open circuit fault occurs in the test system. The fault occurs at time t = 0.45 sec after all the loads in the zones are switched on. The current and voltage outputs of the power supply modules on both starboard and Ports are shown in Figure 6.9. The current in the power supply module on the starboard side becomes zero at the time of fault and the current on the Port increases at the time of fault, but the voltages remain the same. The currents in the starboard and Port power supply modules again come back to the initial values after the fault is cleared. The voltage of the



(a) ***Fault Control Flow Diagram***



(b) ***Zone Control Flow Diagram***



(c) ***Breaker Control Flow Diagram***

Figure 6.7: *Command Control Center Flow Diagrams*

power supply module on starboard side shown in Figure 6.9(c) decreases slightly the moment the fault is cleared because the load is present on the module at the time the fault is cleared.

The circuit breakers connected to the converter modules on the starboard side become active the moment fault is detected. The breakers immediately break the supply to the converter modules. The circuit breaker inform the controller of the occurrence of a fault by sending the signal shown in Figure 6.10(a). The circuit breakers remain active till the fault is cleared by the controller using the

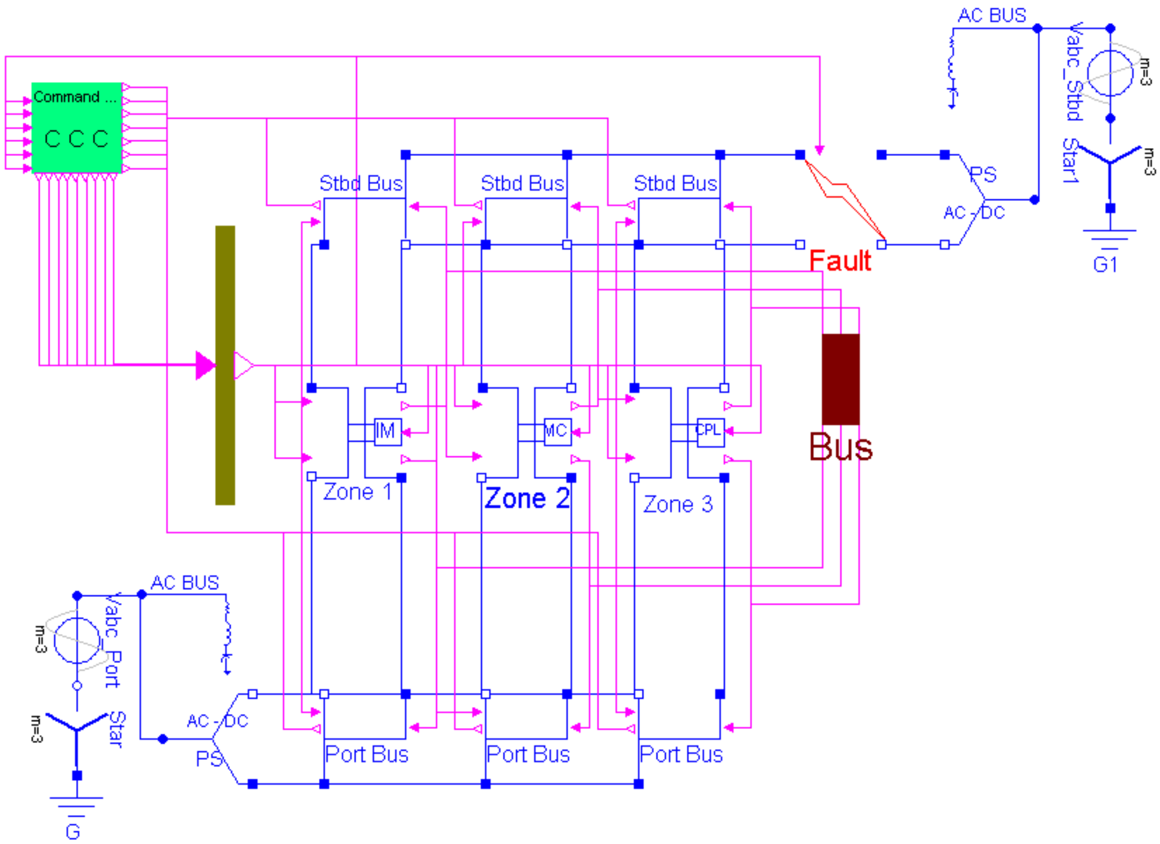
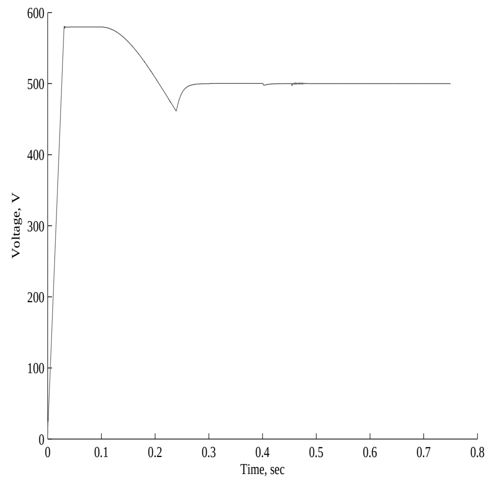


Figure 6.8: *Network Protective Test System*

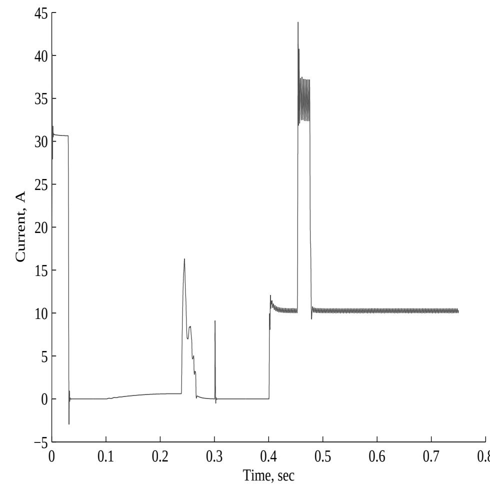
signal shown in Figure 6.10(b) at time t = 0.475 sec.

The moment the controller receives signal from the circuit breaker, the controller deactivates the effected converter modules by sending the turn-off signal to the converter shown in Figure 6.11(a). The signal is same for all the controllers effected by the fault. After the fault is cleared the converters are not immediately turned on, a time delay t = 20 m sec is introduced to switch back on in order to give sufficient time for the capacitors in the converters to be recharged. The converters not effected by the fault are not turned off as shown in Figure 6.11(b) by the converter turn on/off signal.

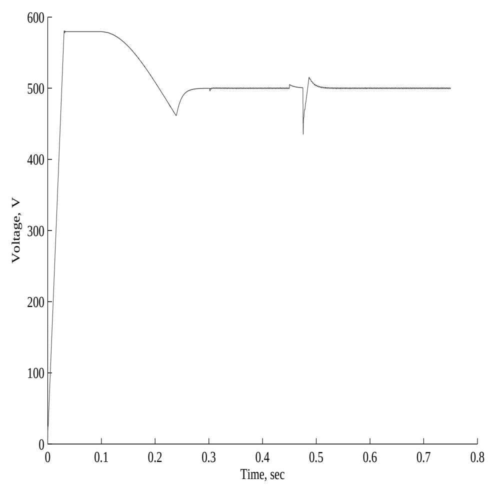
The voltages and currents of the converter modules are shown in the Figures 6.12 - 6.14. The voltages of the converters modules on starboard side in Zone1 and Zone2 decrease because of the fault. The current of the converter modules on the starboard side reduces and the current of the converter increases during the fault and come back to the initial values after the fault is cleared to provide continuous supply to the loads even during the time of fault. The voltages and currents of the



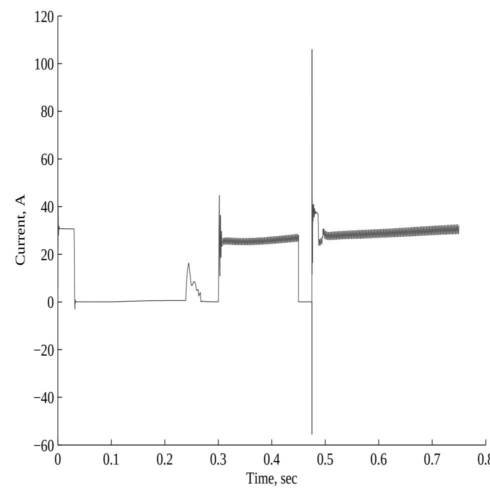
(a) ***PS Port Voltage***



(b) ***PS Port Current***



(c) ***PS Stbd Voltage***

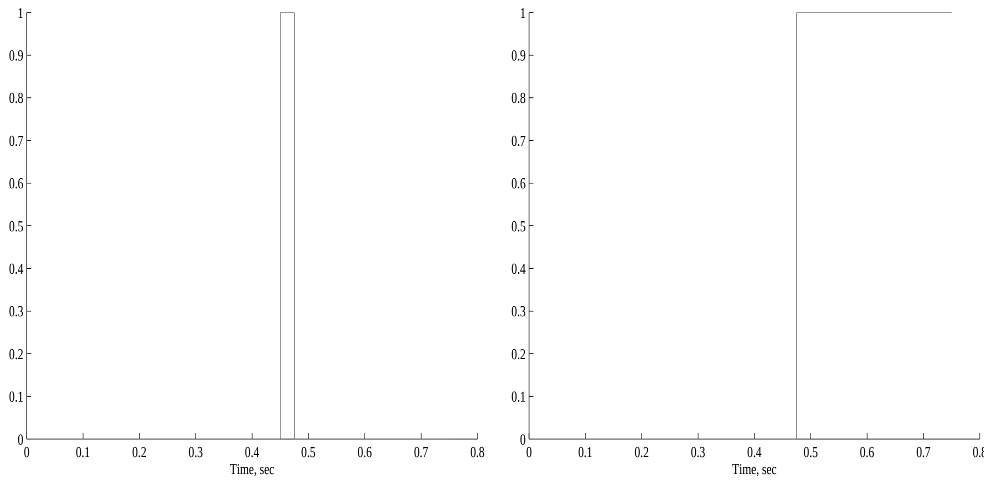


(d) ***PS Stbd Current***

Figure 6.9: *Power Supply Voltages and Currents for Open Circuit Test*

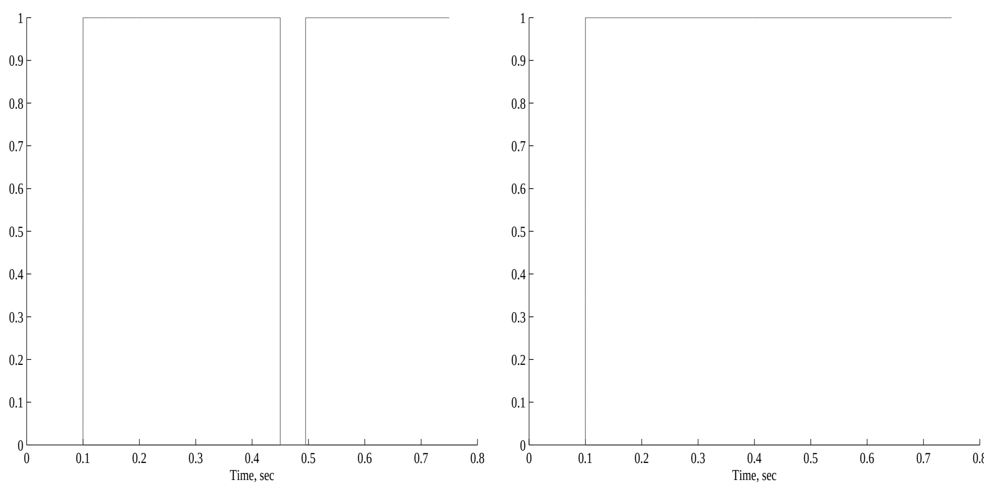
converter modules in Zone 3 are not effected by the fault. The reason behind this is the load in Zone 3 is supplied by the Port converter module rather the starboard side converter module.

The continuous supply of power to the loads is indicated by the current waveforms of the load module in the zones (Figure 6.15). It can be seen that the currents in the loads remain same even though there is a fault in the system thus increasing the survivability of the ship.



(a) ***Fault Signal*** **(b)** ***Fault Clear Signal***

Figure 6.10: *Fault Signal from the breakers and Fault Clearing Signal from the Controller for Open Circuit Test*



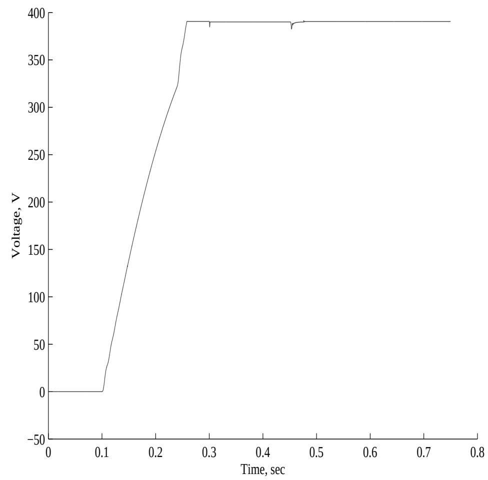
(a) ***Stbd Converter Switch*** **(b)** ***Port Converter Switch***

Figure 6.11: *Zone 1 Port and Stbd SSCM Turn On/Off Signals for Open Circuit Fault Test*

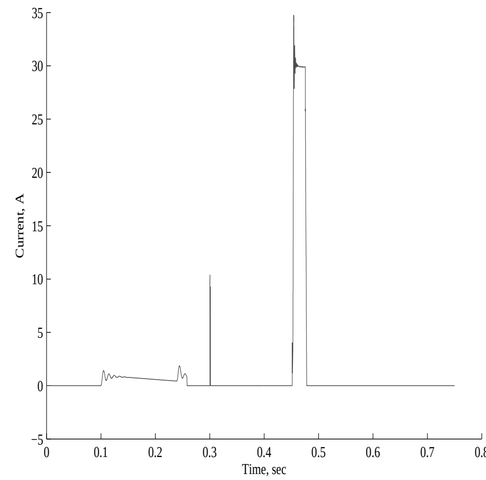
6.6.2 Short Circuit Fault Test

The short circuit fault test is performed on the same test (Figure 6.8). The fault model is connected at the output of the power supply module on the starboard side as in open circuit test and the fault occurs on the starboard side at time t = 0.45 sec.

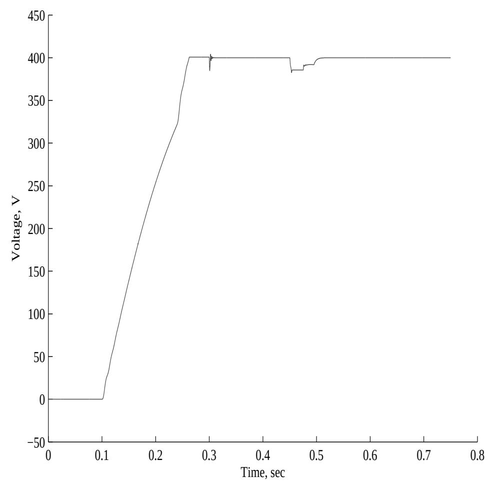
Figure 6.16 shows the voltage and currents of the power supply module on both starboard and



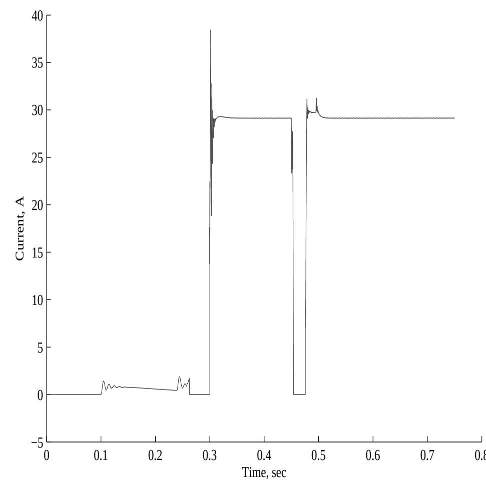
(a) ***SSCM Port Voltage***



(b) ***SSCM Port Current***



(c) ***SSCM Stbd Voltage***

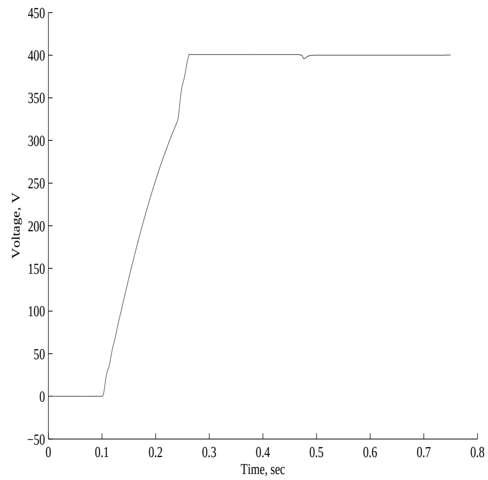


(d) ***SSCM Stbd Current***

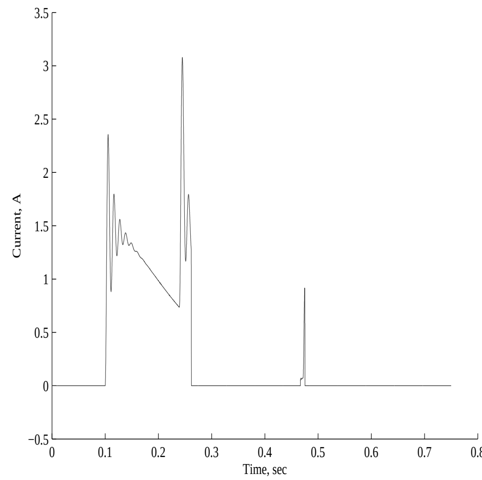
Figure 6.12: *Zone 1 SSCM Voltages and Currents for Open Circuit Fault Test*

Ports. The voltage of the power supply module decreases and the current increases instantaneously indicating a short circuit fault. The current in the power supply module on the starboard sides increases during the time of fault to supply the loads.

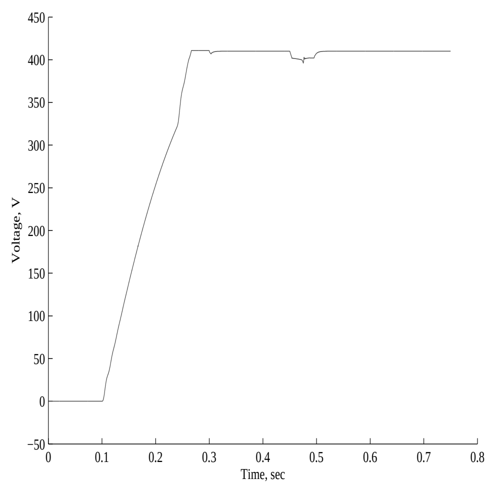
The circuit breakers are activated at the time fault as shown in Figure 6.17(a) and become in active when the Fault_Clear signal from the controller is enabled. The Fault_Clear signal shown in Figure 6.17(b) clears the fault at time t = 0.475 sec. The moment fault occurs the controller deactivates the converter modules and again activates after the fault is cleared. The turn on/off signals for the converters in Zone1 is shown in Figure 6.18. The signals are same for converters in Zone2



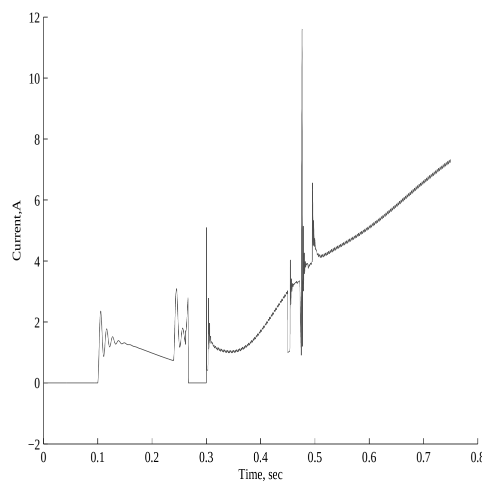
(a) ***SSCM Port Voltage***



(b) ***SSCM Port Current***



(c) ***SSCM Stbd Voltage***

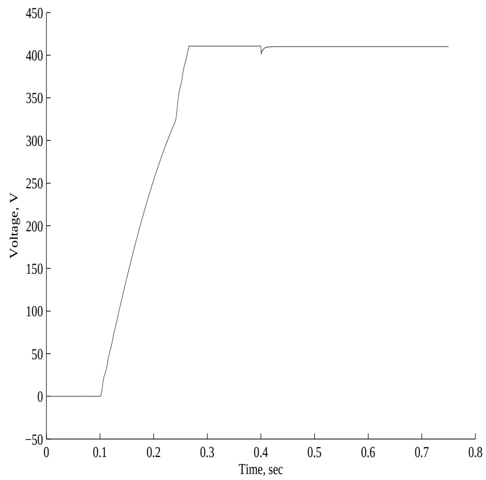


(d) ***SSCM Stbd Current***

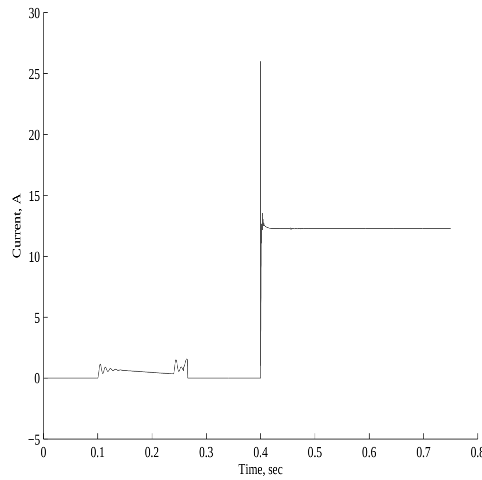
Figure 6.13: *Zone 2 SSCM Voltages and Currents for Open Circuit Fault Test*

and Zone3. The converters not effected by the fault are not de-activated.

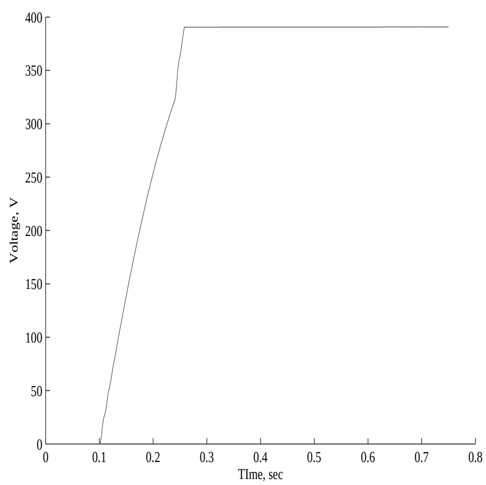
Figures 6.19 - 6.21 shows the voltages and currents of the converter modules present in the zones. The voltage and current of the converters on the starboard side in Zone1 and Zone2 decrease during the time of fault and come back to the initial values after th fault is cleared. The currents of the converters on the Port in zone1 increases to supply the load during the time of fault. The voltages and currents of the Zone3 converter modules are not effected by the fault on the starboard side as the converter on the Port supplies power to the load.



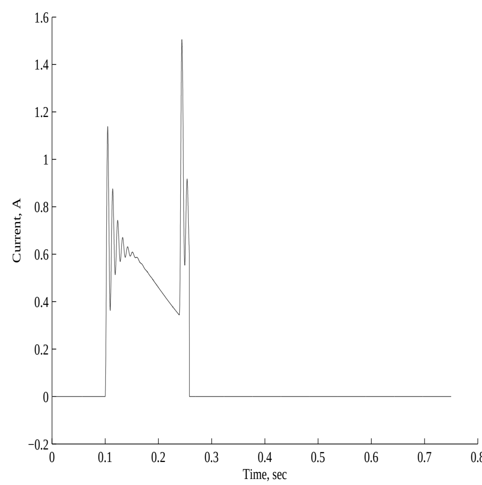
(a) ***SSCM Port Voltage***



(b) ***SSCM Port Current***



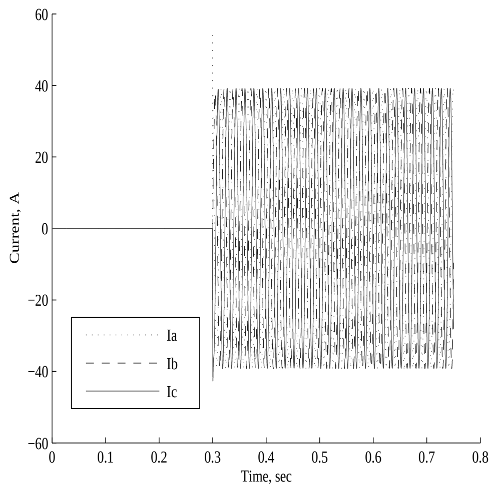
(c) ***SSCM Stbd Voltage***



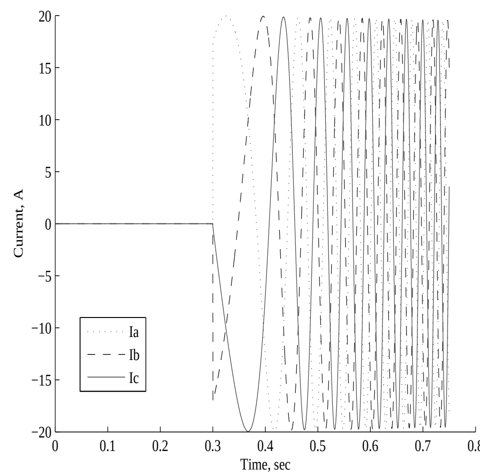
(d) ***SSCM Stbd Current***

Figure 6.14: *Zone 3 SSCM Voltages and Currents for Open Circuit Fault Test*

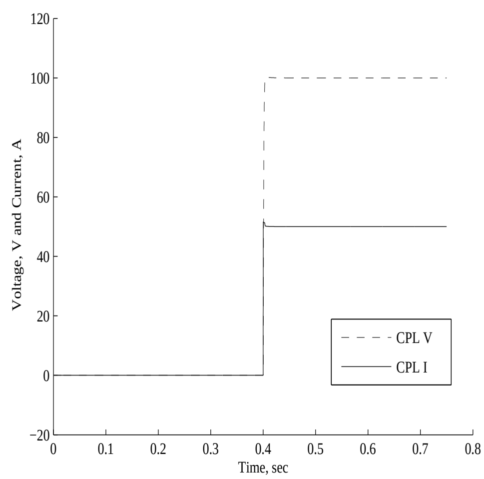
The continuous supply to the loads is maintained by the converter modules in zones and this can be verified by Figure 6.22 where no interupption in power supply to the loads in zones can be observed.



(a) ***Zone 1 Load Current***

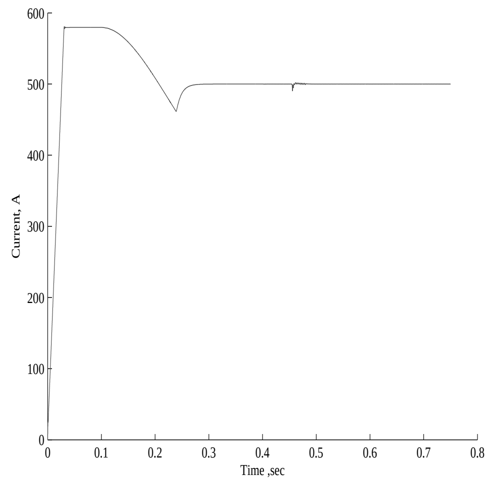


(b) ***Zone 2 Load Current***

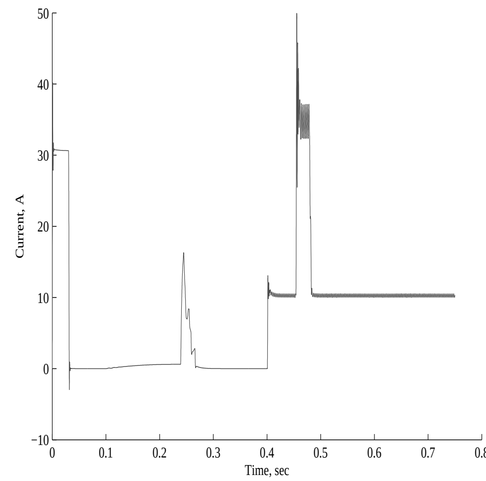


(c) ***Zone 3 Load Current***

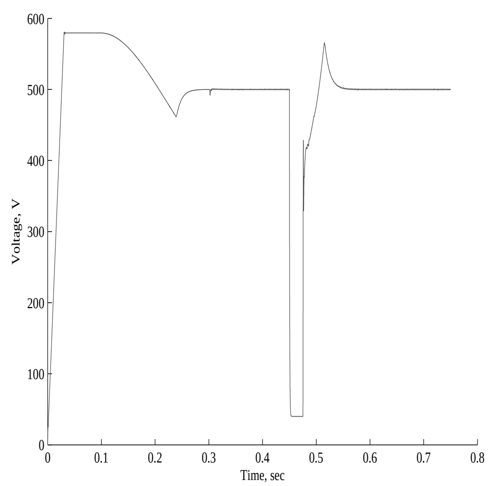
Figure 6.15: *Zones' Load Currents During Fault Simulation Test*



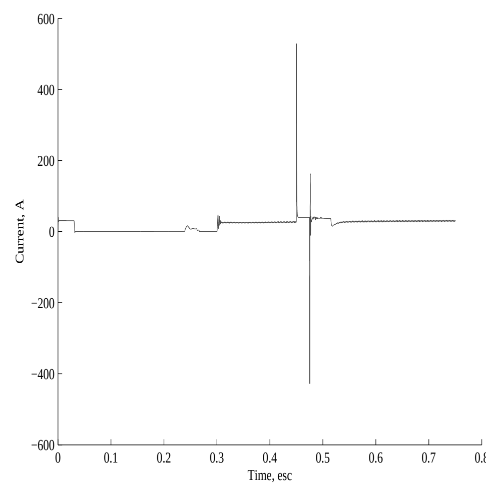
(a) ***PS Port Voltage***



(b) ***PS Port Current***

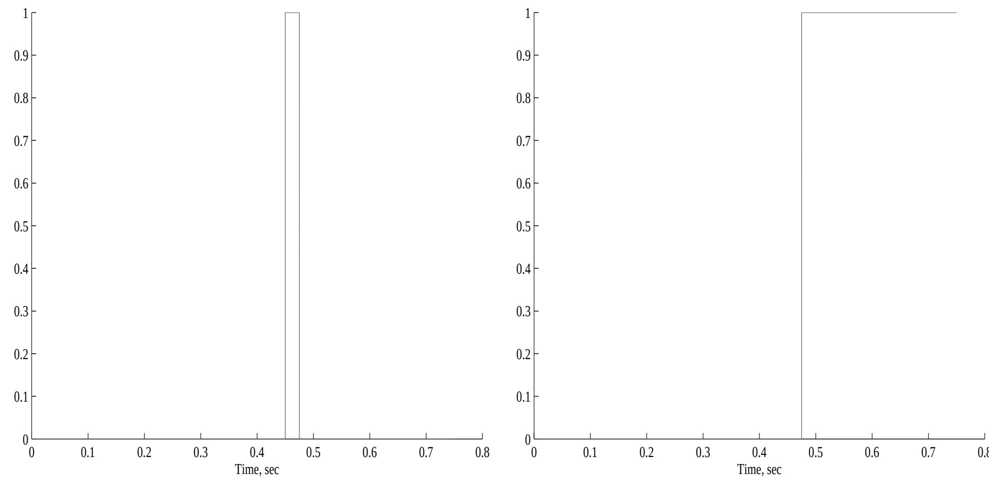


(c) ***PS Stbd Voltage***



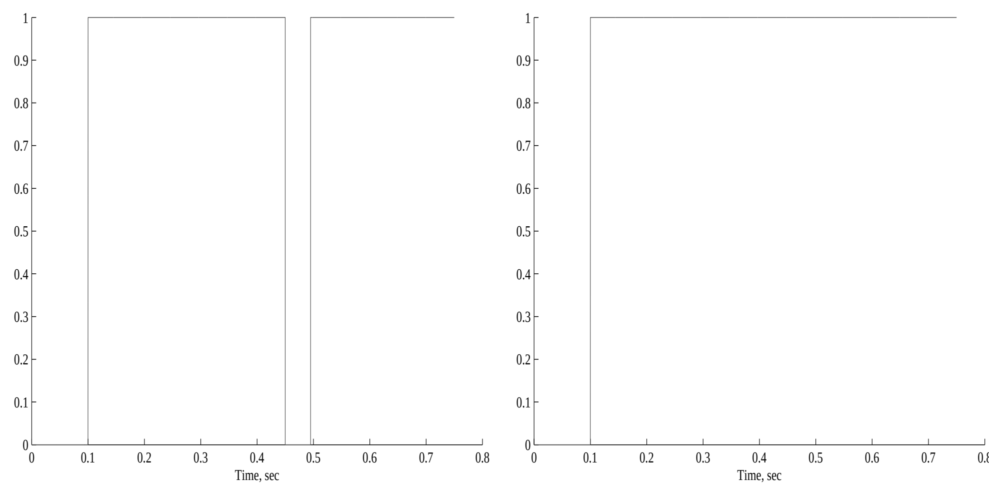
(d) ***PS Stbd Current***

Figure 6.16: *Power Supply Voltages and Currents for Short Circuit*



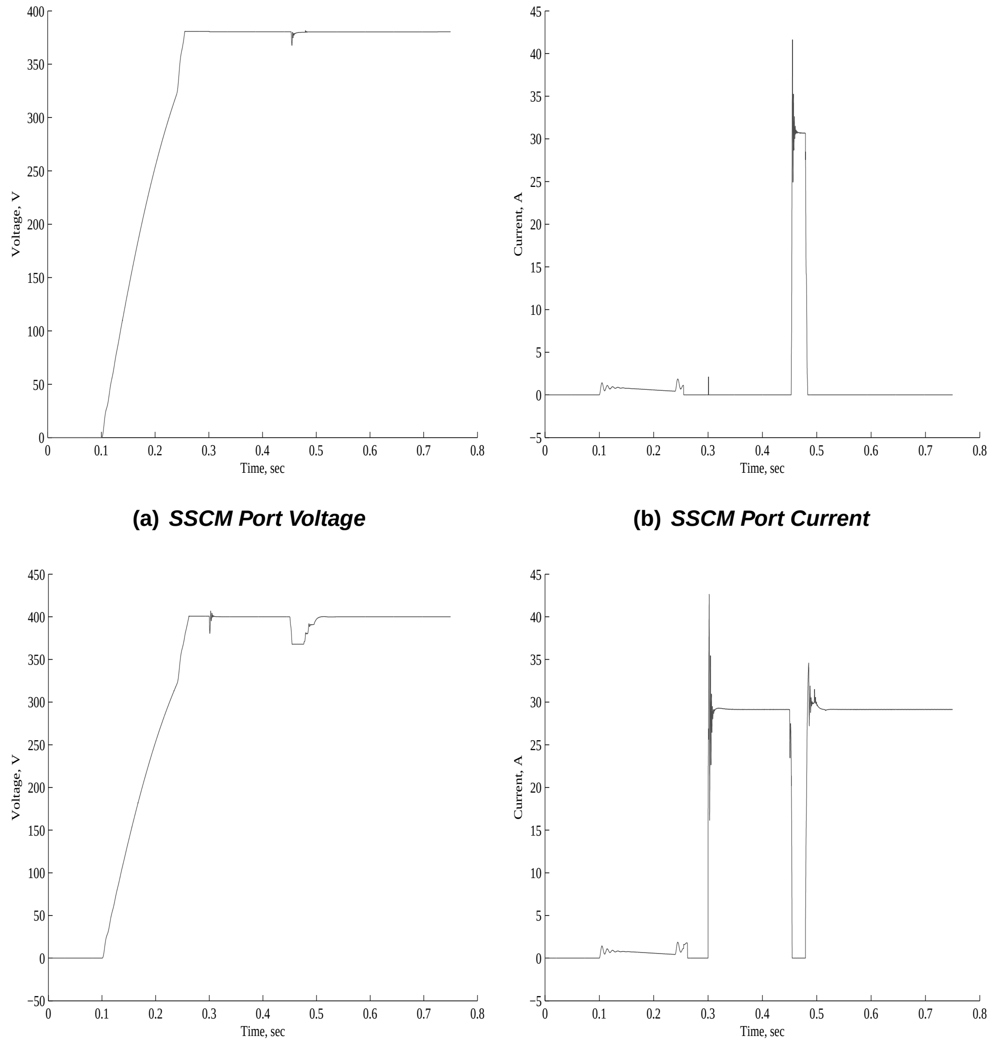
(a) ***Fault Signal*** **(b)** ***Fault Clear Signal***

Figure 6.17: *Fault Signal from the breakers and Fault Clearing Signal from the Controller for Short Circuit Test*



(a) ***Stbd SSCM Switch*** **(b)** ***Port SSCM Switch***

Figure 6.18: *Zone 1 Port and Stbd SSCM Turn On/Off Signals for Short Circuit Fault*



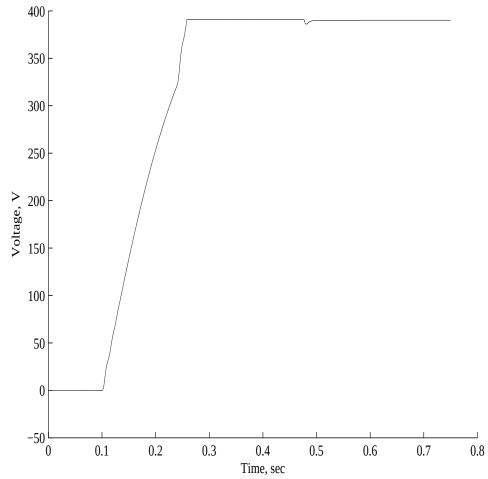
(a) ***SSCM Port Voltage***

(b) ***SSCM Port Current***

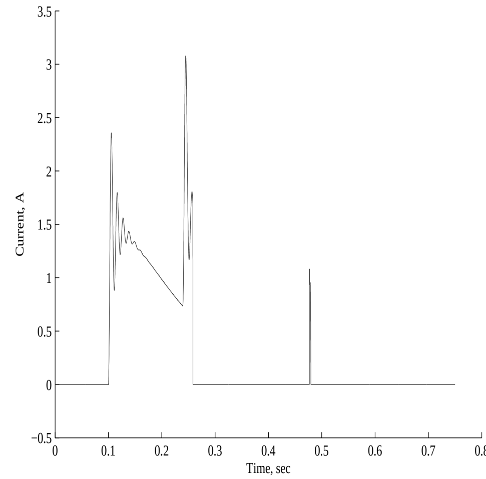
(c) ***SSCM Stbd Voltage***

(d) ***SSCM Stbd Current***

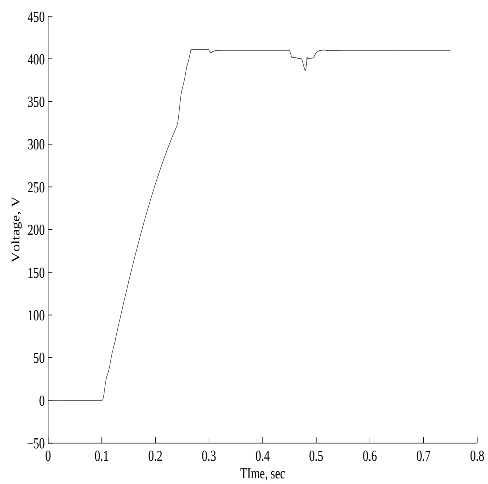
Figure 6.19: *Zone 1 SSCM Voltages and Currents for Short Circuit Fault Test*



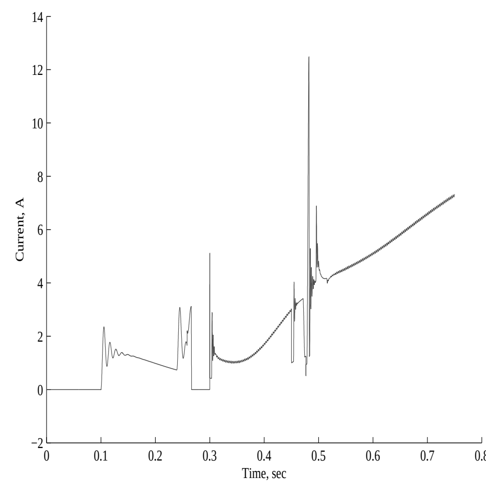
(a) ***SSCM Port Voltage***



(b) ***SSCM Port Current***

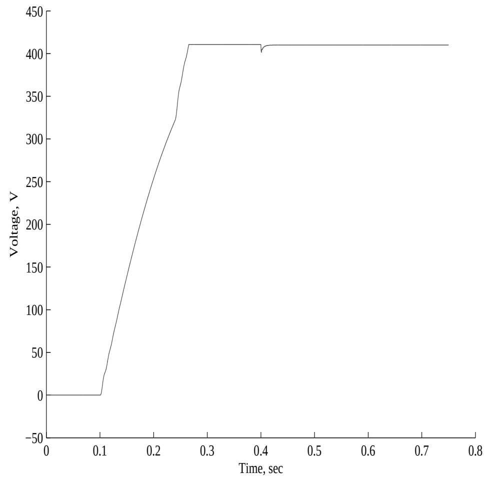


(c) ***SSCM Stbd Voltage***

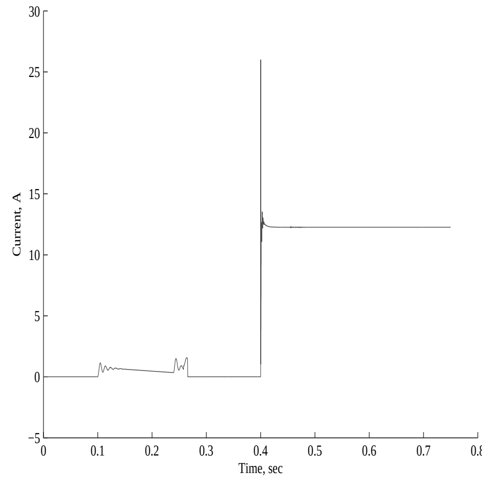


(d) ***SSCM Stbd Current***

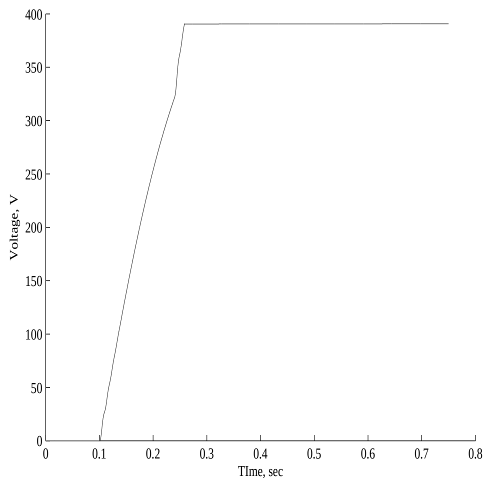
Figure 6.20: *Zone 2 SSCM Voltages and Currents for Short Circuit Fault Test*



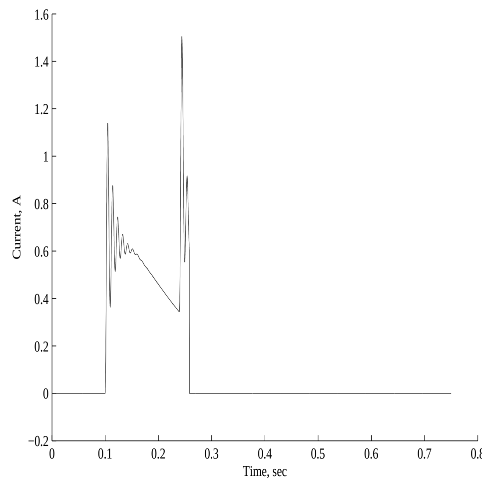
(a) ***SSCM Port Voltage***



(b) ***SSCM Port Current***

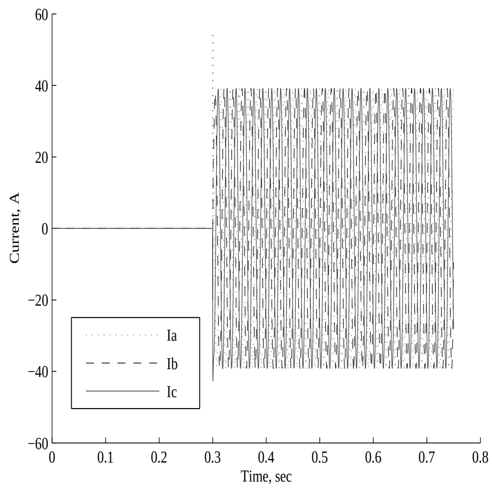


(c) ***SSCM Stbd Voltage***

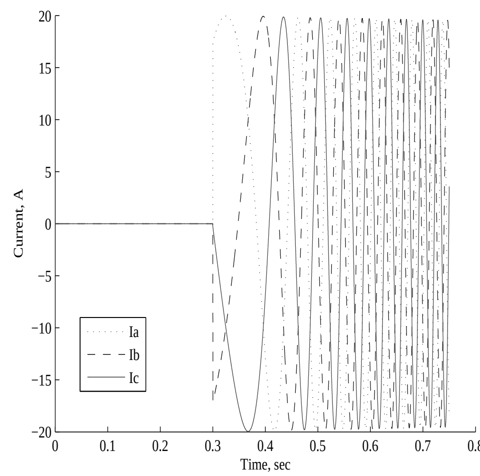


(d) ***SSCM Stbd Current***

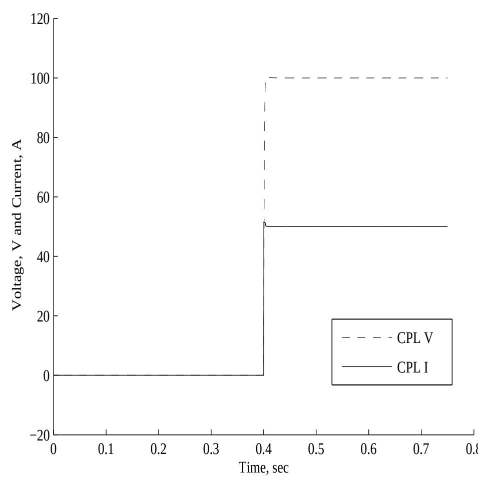
Figure 6.21: *Zone 3 SSCM Voltages and Currents for Short Circuit Fault Test*



(a) ***Zone 1 Load Current***



(b) ***Zone 2 Load Current***



(c) ***Zone 3 Load Current***

Figure 6.22: *Zones' Load Currents During Short Circuit Fault Simulation*

CHAPTER

7

Summary and Future Work

7.1 Conclusions

A shipboard power system (SPS) library was developed in Modelica using the concepts of packages. The library developed includes power electronics, machines, and control models used in modeling components. The shipboard power system was modeled and simulated as two decoupled systems, the AC generation and propulsion system and the DC distribution system test bed. The results of the simulations were compared with the MATLAB/Simulink ONR model available and was found to give the same results.

To protect the system from different faults in the DC distribution system test bed a Network Protection System was developed. The protection system model detects faults in the system using the circuit breaker modeled and clears the fault using the discrete controller designed. The communication of the signals between the controller and the plant is done using a constant delay communication network. Different case studies performed indicate that the protective system is working efficiently.

Modelica has been used to follow an object-oriented approach in modeling and simulating the shipboard power system. Modelica is a good alternativein modeling due to the efficient re-use of the models developed, a property of object-oriented languages that reduces the time for modeling. The graphical user interface provided by Dymola were customized for the library components developed to

Table 7.1: *Time Taken for Different Tests*

Description	Simulation Model	Simulation Time	CPU Time
AC Geneartion and Propulsion System	Test_ACSYS	1 sec	30 min
DC Distribution Test bed	Test_DCDTB	1 sec	5 hrs
Open Circuit Fault Test	Test_Open_DCDTB	0.75 sec	6 hrs 30 min
Short Circuit Fault Test	Test_Short_DCDTB	0.75 sec	5 hrs

facilitate data input. The simulation of components of the shipboard power system individually takes the least amount of time but when the system is simulated as a whole the simulation time increases drastically. The simulation time taken for simulating the components of the shipboard power system and for the test cases is shown in the Table 7.1.

7.2 Future Work

The time taken for simulation runs is very long. This can be reduced by making efforts towards implementing some of the parallel programming techniques presented in [36] to parallelize some part or the whole of the shipboard power system. The networked protection system developed is restricted only to the DC distribution system but can be extended to include the AC generation and propulsion system in future work. The accuracy of the results of shipboard power system simulations can be increased by tuning and optimizing the control parameters. The optimization can be performed in two different ways.

- Modelica scripts, external functions written to initialize models, set parameter values, perform simulations and plot variables.
- Dymola version 6.0 can be used. The newer version of Dymola has in built capabilities to perform optimization an tuning of model parameters which can be utilised in optimizing the shipboard power system.

APPENDIX

A

SPS Parameters

The parameters of the components of shipboard power system and their control parameters are given in the following.

Table A.1: *Synchronous Machine Parameters*

Parameter	Symbol	Value
Rated Power	P_b	59 KW
Base Voltage	V_b	570 V (l-l) rms
Base Frequency	ω_b	377 rad/sec
Base Impedance	Z_b	9.538 Ω
Number of Poles	P	4
Stator Winding Resistance	R_s	0.003 pu
Stator Winding Leakage Reactance	X_{ls}	0.19 pu
D-axis Reactance	X_d	0.18 pu
Q-axis Reactance	X_q	0.18 pu
Rotor Winding Resistance	R_{fd}	0.000929 pu
Rotor Winding Leakage Reactance	X_{lfd}	0.1414 pu
D-axis damper Resistance	R_{kd}	0.01334 pu
D-axis damper Leakage Reactance	X_{lkd}	0.08125 pu
Q-axis damper Resistance	R_{kq1}	0.00178 pu
Q-axis damper Leakage Reactance	X_{lkq1}	0.8125 pu
Q-axis damper Resistance	R_{kq2}	0.00841 pu
Q-axis damper Leakage Reactance	X_{lkq2}	0.0939 pu

Table A.2: *AC Bus with Harmonic Filter*

Parameter	Symbol	Value
Shunt Resistance	R	500 Ω
Shunt Capacitance	C	40 μF
Filter Resistance	R_f	39 m Ω
Filter Inductance	L_f	5.61 mH
Filter Capacitance	C_f	49.75 μF

Table A.3: *Propulsion System Induction Motor Parameters*

Parameter	Symbol	Value
Number of Poles	P	4
Base Frequency	ω_b	377 rad/sec
Stator Winding Resistance	R_s	0.087 Ω
Stator Winding Leakage Reactance	X_{ls}	0.302 Ω
Magnetizing Inductance	X_m	13.08 Ω
Rotor Winding Resistance	R_r	0.228 Ω
Rotor Winding Leakage Reactance	X_{lr}	0.302 Ω
Inertia	J	1.662 Kgm^2
Base Torque	T_b	198.0 N-m

Table A.4: *Propulsion System DC Link Parameters*

Parameter	Symbol	Value
Inductance	L_{dc}	11.4 mH
Resistance	R_{dc}	0.056 Ω
Capacitance	C_{dc}	1.988e-3 F

Table A.5: *PS Module Control Parameters*

Parameter	Symbol	Value
Reference Voltage	V_{ref}	500 V
Maximum Command Current	i_{max}	40 A
PI Controller, Proportional Gain	K_p	1
PI Controller, Integral Gain	K_i	100
Current Regulator, Integral Gain	K_{ih}	100
NSC exponent	n	1
NSC time constant	t	5.00e-03
Hysteresis Error Bandwidth	h	1.0

Table A.6: *Port and Starboard DC Bus*

Parameter	Symbol	Value
Inductance	L	2.2 μH
Resistance	R	0.034 Ω
Capacitance	C	0.5 μF
Conductance	G	10 Ω^-1

Table A.7: *SSCM Circuit Parameters*

Parameter	Symbol	Value
Switching Frequency	f_{sw}	20 KHz
Rated DC Input Voltage	V_{in}	500 V DC
Rated DC Output Voltage	V_{out}	400 V DC
Rated Output Power	P_{out}	8 kW
Input Filter Inductor	L_f	357 μH
Input Inductor Resistor	R_{fl}	0.2 Ω
Input Filter Capacitor	C_{f1}	500 μF
Input Filter Series Resistor	R_{fc}	1.0 Ω
Input Filter Additional Capacitor	C_{f2}	45 μF
Buck Converter DC Inductor	L	3.0 mH
Buck Converter DC Inductor Resistance	R_L	0.5 Ω
Buck Converter Output Capacitor	C	500.0 μF

Table A.8: *SSCM Control Parameters*

Parameter	Symbol	Value
Reference Voltage	V_{ref}	400 V DC
PI Proportional Integral Gain	K_i	0.1
PI Proportional Gain	K_p	23
Current Feedforward Proportional Gain	h_i	0.015

Table A.9: *SSIM Circuit Parameters*

Parameter	Symbol	Value
Rated DC Input Voltage	V_{in}	500 V DC
Input Filter Series Resistance	R_s	10.0 mΩ
Input Filter Capacitor	C_{in}	590 μF
Input Filter Capacitor Series Resistor	R_{cin}	127.0 mΩ
Output Filter ac Inductor	L_{ac}	550 μH
Output Filter ac Inductor Resistance	R_{lac}	38.0 mΩ
Output Filter, ac Capacitor	C_{ac}	50.0 μF
Output Filter, ac Capacitor Series Resistance	R_c	8.0 mΩ
Load Bank, Inductance	L	1.0 mH
Load Bank, Resistance	R_L	15.9 Ω

Table A.10: *SSIM Control Parameters*

Parameter	Symbol	Value
Upper Bound on Command Current	i_max	40.0 A
Current Regulator Integral Gain	K_i	46.9
Current Regulator Proportional Gain	K_p	0.255
Synchronous Current Regulator Time Constant	τ_{scr}	0.5 ms
Synchronous Current Regulator Turn-off Time Constant	τ_{scr_off}	1.0 ms
Synchronous Current Regulator saturation Limit	I_{fcl}	20

Table A.11: *MC DC Link Parameters*

Parameters	Symbol	Value
Rated DC Input Voltage	V_{in}	380-440 V
Input Filter, Series Resistance	R_{in}	10.0 mΩ
Input Filter, Capacitor	C_{in}	1000.0 μF

Table A.12: *MC Induction Motor Parameters*

Parameter	Symbol	Value
Rated Input Voltage	V_{in}	230 V(l-l) rms
Number of Poles	P	4
Base Flux	ψ_b	185 V
Base Frequency	ω_b	377 rad/sec
Stator Winding Resistance	R_s	0.315 Ω
Stator Winding Leakage Reactance	X_{ls}	0.546 Ω
Magnetizing Inductance	X_m	18.92 Ω
Rotor Winding Resistance	R_r	0.591 Ω
Rotor Winding Leakage Reactance	X_{lr}	0.546 Ω
Inertia	J	0.089 Kgm^2
Base Torque	T_b	19.8.0 N-m

Table A.13: *MC Control Parameters*

Parameter	Symbol	Value
Speed Control Proportional Gain	K_{sc}	5
Speed Control Integral Gain	τ_{sc}	1
Maximum torque Limit	T_{max}	25 N-m

Table A.14: *CPL Circuit Parameters*

Parameter	Symbol	Value
Rated DC Input Voltage	V_{in}	120 - 600 V DC
Rated DC Output Voltage	V_{out}	100 V DC
Rated Output Power	P_{out}	5 kW
Input Filter Capacitor	C_{in}	470 μF
Input Filter Series Resistor	R_{in}	1.0 mΩ
Buck Converter DC Inductor	L	2.0 mH
Buck Converter DC Inductor Resistance	R_L	0.01 Ω
Buck Converter Output Capacitor	C	470.0 μF
Nominal Load Resistance	R_l	2.0 Ω

Table A.15: *CPL Control Parameters*

Parameter	Symbol	Value
Reference Power	P_{ref}	5 kW
Lower Bound on Input Voltage	Vin_{min}	120 V
Upper Bound on Input Voltage	Vin_{max}	1000 V
Lower Bound on Output Voltage	$Vout_{min}$	50 V
Upper Bound on Output Voltage	$Vout_{max}$	1000 V
Maximum Command Current	i_max	51 A
Current Regulator, Time Constant	K_i	100
NSC exponent	n	1
NSC time constant	τ	5.00e-04
Hysteresis bound on Current Control	h	1

References

[1] Adediran, A., T., Xiao, H., and Butler, K., L.,“The Modeling and Performance Testing of a Shipboard Power System,” *Proceedings of the 33 Annual Frontiers of Power Conference*, Vol 1, 1-9, October 2000.

[2] Aegis Technological Inc., http:\\ www.acslsim.com\

[3] Alternative Transients Program (ATP), http:\\www.emtp.org.

[4] Andersson, M.,“OmSim and Omola :Tutorial and User’s Manual Ver. 3.4,”

[5] Aspen Technologies Inc, http:\\ www.aspentec.com\

[6] Behr, A., Jochum, A., and Becker, G., “Using The Dynamic Simulation Environment SMILE For The Design Optimization of Solare Heating Systems”

[7] Butler, K. L., Sarma, N. D. R., Whitcomb, C., Do Carmo, H., Zhang, H., “Shipboard Systems Deploy Automated Protection,” *IEEE Computer Applications in Power*, 11(2), 31-36, April, 1998.

[8] Cervin, A., Henriksson, D., Lincoln, B., Eker, J., Arzen, K., E.,“How Does Control Timing Affect Performance?,” *IEEE Control Systems Magazine*, 23(3), pages 16-30, 30 June, 2003.

[9] Ciezki, J. G., Ashton, R. W.,”Selection and Stability Issues Associated with a Navy Shipboard DC Zonal Electric Distribution System,” *IEEE Transaction on Power Delivery*, Vol. 15, No. 2, pp. 665-669, April 2000.

[10] Dymola, Homepage: http:\\www.dynasim.com\.

[11] Dynasim’s, “Dymola User’s Manual: Version 5.3a,” 2005

[12] Ebron, J.,“Modeling and Simulation of Thermal Power Plants,” *Thesis Dept. of Automatic Control, Lund Institute of Technology*, February 1998.

[13] Electromagnetic Transient Program, http:\\ www.emtp.com\

[14] Elmqvist, H.,“A Structured Model Language for Large Continuous Systems,” *Dept.of Automatic Control, Lund Institute of Technology*, Lund, Sweden 1978.

[15] EUROSTAG, http:\\www.eurostag.be\frameset.htm

[16] Fabricius, S., M., O., “Extensions to the Petri Net Library in Modelica,” Swiss Federal Institute of Technology Zurich, Switzerland.

[17] Hydro-Quebec,*Power System Blockset for use with Simulink*, Mathworks.Inc 2001

[18] Jeandel, A., Favret, F., Lapenu, L., Lariviere, E., “ALLAN Simulation, a general tool for model description and simulation,” *Proceedings of the IBPSA Conference*, 1993, (Adelaide, AUSTRALIA).

[19] Jeandel, A., Boudad, F., “Physical System Modeling Languages : From ALLAN to MODELICA,” *GAZ DE FRANCE - Research and Development Division*

[20] Juan A. Martinez, “Modeling of Custom Power Equipment Using Electromagnetic Transients Programs”

[21] Karl Schoder, Ali Feliachi, “Simulation and Control of Electric Shipboard Power Systems using Modelica and MATLAB/Simulink,” *Power Engineering Society General Meeting*, 1, pages 244-248, 6-10 June 2004.

[22] Kral, C., Haumer, A.,“Modelica Libraries for dc machines, three phase and polyphase machines,” *Proceedings of the* 4^{th} *International Modelica Conferenece*, pages 549-558, 7-8 March 2005.

[23] Krause, P., C., Wasynczuk, O. and Sudhoff, S., D.,*Analysis of Electric Machinery and Drive Systems*, John Wiley & Sons, 2002.

[24] Martin, C., Urquia, A., and Dormido, S.,“SPICELib - Modeling and Analysis of Electric Cirucits with Modleica,” *3nd International Modelica Conferenece*,161-170, 3-4 November 2003.

[25] Mathcore Engineering AB, Homepage: http:\\www.mathcore.com\.

[26] Mathworks Inc, http:\\ www.mathworks.com\

[27] Mats Larsson,“ObjectStab - A Modelica Library for Power System Stability Studies,” *Modelica Workshop*, pages 13-22, October 2000.

[28] Mattsson, S., E., and Anderson, M.,“Ideas Behind Omola,” *CACSD 92: IEEE Symposium Computer Aided Control System Design*, 23-29, 1992.

[29] Mechanical Dynamics Inc., http:\\ www.mscsoftware.com\

[30] Micheal M. Tiller,*Introduction to Physical Modeling in Modelica*, Kluwer Academic Publishers, 2001.

[31] Modelica Association, http:\\ www.modelica.org\

[32] Mohan, N., Undeland, T., M., Robbins, W., P., *Power Electronics Converters Applications and Design*, John Wiley & Sons, 2^{nd} edition, 1995.

[33] Mostermann, P., J., Otter, M., and Elmqvist, H., “Modeling Petri Nets as Local Constraint Equations for Hybrid Systems Using Modelica,”*Summer Computer Simulation Conference*, pages 314-319, 19-22 July 1998.

[34] Murata, T., “Petri Nets: Properties, Analysis and Applications,” *Proceedings of the IEEE*, 77(4), April 1989.

[35] Najafi, M., Furic, S., Nikoukhah, R., “SCICOS: A General Purpose Modeling and Simulation Environment,” *Proceedings of the* 4^{th} *International Modelica Conference*, Mrach 7-8 2005, Hamburg, Germany.

[36] Nystrom, K., Aronsson, P., Fritzon, P., " Parallelization in Modelica,"*Proceedings of the* 4^{th} *International Modelica Conference*, Mrach 7-8 2005, Hamburg, Germany.

[37] "ONR Control Challenge Problem (white paper)," http:\\ www.usna.edu\EPNES\Challenge Problem.htm, 2002.

[38] Orcad PSPICE, http:\\www.orcad.com\

[39] Otter, M., Elmqvist, H., Mattsson, S., E.,"Hybrid Modeling in Modelica Based on Synchronous Data Flow Principle," CACSD'99, 22-26 August 1999.

[40] Pekarek, S., Tichenor, J., Sudhoff, S., Sauer, J., D., Delisle, D., and Zivi, E., "Overview of a Naval Combat Survivability Program," 13^{th} *International Ship Control Systems Symposium*, Orlando, Florida, USA, 2003.

[41] Peter Fritzon, *Principles of Object-Oriented Modeling and Simulation With Modelica 2.1*, John Wiley & Sons, November 2003.

[42] Petri, C., A., "Communication with Automata", *PhD Dissertation*, New York: Griffiss Air Force Base.Tech. Rep. RADC-TR-65-377, vol. 1, SUPPI. 1, 1966

[43] Power System Simulator, http:\\www.pti-us.com\pti\software\psse\index.cfm

[44] *ROOMS - Research Center for Object Oriented Modeling and Simulation*, Homepage: http:\\www.ida.liu.se\labs\pelab\modelica, Linkoping University, Sweden.

[45] Saberbook Navigator, *Analogy Inc.*, Version 4.3.

[46] Sahlin, P., Bring, A., and Sowell, E., F.,"The Neutral Model Format for building Simulation: Version 3.02," *Dept. of Building Services Engineering, Royal Institute of Technology*, Stockholm Sweden, June 1996.

[47] Sergej N. Kalaschnikow,"PQLib - A Modelica Library for Power Quality Analysis in Networks," *Proceedings of the* 2^{nd} *International Modelica Conferenece*, March 2002.

[48] SIMPOW, http:\\www.stri.se\metadot\index.pl

[49] Sudhoff, S., D., Pekarek, S., Kuhn, B., Glover, S., Sauer, J., and Delisle, D.,"Naval Combat Survivability Testbeds Investigation Shipboard Power Electronics Based Power Propulsion," *IEEE Power Engineering Society Summer Meeting*, 347350, 2002.

[50] TechOnline, http:\\www.techonline.com\community\ed_resource\feature_article\14293

[51] Tummescheit, H., Klose, M., and Ernst, T.,"Modelica and Smile - A Case Study Applying Object-Oriented Concepts to Multi-Facet Modeling," *Proceedings of the 9th European Simulation Symposium ESS97*, Budapest, Hungary, October 1997.

[52] Wasynczuk, O., Sudhoff, S., D., Tran, T., D., Clayton, D., H., and Hegner, H., J.,"A voltage Control Strategy for Current Regulated PWM Inverters," *IEEE Tans. on Power Electronics*, 11(1) January 1996.

[53] Xiao, H., Adediran, A. T., Butler, K. L., "Fault scenario studies based on geographical information for shipboard power systems," *Proceedings of North American Power Symposium*, 436442, October 2001.

[54] Zhang, H., Butler, k., L., Sarma, N., D., R.,DoCarmoand, H., Gopalakrishnanand, S., and Adediran, A.,"Analysis of Tools for Simulation of Shipboard Electric Power Systems," *Electric Power Systems Research*,Vol 58, 111-122, 2001.

[55] Zhang, H., and Butler, K.,"Simulation of Ungrounded Shipboard Power Systems in PSpice," *Proceedings Midwest Symposium on Circuits and Systems*, 23-29, 1998.