Designing a Real-Time Grid Simulator for use in Market and A.G.C. Studies

by

Oliver Romaniuk

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2008

© Oliver Romaniuk 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I understand that my thesis may be made electronically available to the public.

Abstract

Market based generation dispatch is becoming the industry norm in advanced electrical jurisdictions. Due to the continuous evolution of markets and their potential impacts on system operation, studies are performed from economic and social perspectives in order to gauge the effects of any changes before implementation into live systems. In addition, it is essential to verify the effect of changes in market design on power systems from a technical perspective.

The main objective of this thesis is to develop a real-time power system simulator for use in the investigation of market designs and automatic generation control schemes. The scope of this thesis is the mathematical algorithms used in the simulator, hardware and software implementation, and validation of the implemented simulator.

The simulator is based on a modified version of the power flow calculation using an innovative combination of a standard numerical technique implemented on a readily available computing hardware platform. The result is a significant decrease in computation time.

The power flow is performed repeatedly, with frequency being calculated between time steps to provide system dynamics. Frequency is calculated using a modified version of the generic generator swing equation. Generator and load models and their respective control systems are provided for the purposes of simulator validation and testing, although do not fall within the scope of the simulator itself.

Acknowledgements

Before we dive headfirst into this technical rant I call my thesis, I would like to say:

"Thank you! I could not have done this without you."

The list of people that statement relates to is long and distinguished, but I would like to individually thank the following people who have helped me in the following ways:

- To my loving and patient parents, Krystyna and Alexander, for creating me and being forced to live with the consequences of that decision for the past 31 years. Some have said that I'm turning out just fine, and I owe it all to you.
- To Associate Professor Ehab El-Saadany, for taking a chance on an unknown quantity, and for all the help and guidance you've provided in the past two years.
- To Aden Seaman, you've been indispensible in turning an idea into reality.
- To Erin Young, this all started with you and I'll never forget that.
- To Mario Chiarelli, the Yoda to my Luke Skywalker.
- To Vito Casola, the Emperor to my Darth Vader.
- To Lily Milanovitch, for calling me a lepton, knowing full well I would spend hours in the high school library trying to figure out what you meant by that.
- To my friends, because without you all I would have lost my mind long ago.
- To Julio Iglesias, Willie Nelson and *'To All the Girls I've Loved Before'*.

And finally, to non-persons who have played a role in my quest for knowledge:

- To the Flying Spaghetti Monster, for touching me with His Noodly Appendage.
- To caffeine, in all its wonderful forms. Jet fuel for the thinking machine.
- To the Universe, for existing and giving us all the opportunity to just be.
- To Humanity, for taking advantage of that opportunity and continuously working towards something better. Every day I wake up and smile, knowing we're one day closer to finally getting it right.

Contents

Appendices

List of Figures

List of Tables

Chapter 1.
Introduction

1.1 Background

In the 1990s, a number of electrical jurisdictions around the world moved from the vertically integrated utility model to that of competitive markets. While the reasoning behind such decisions is outside the scope of this discussion, it can be said that this move introduced a number of new intricacies to the operation and control of power systems [1]. One of the primary changes was the use of wholesale electricity markets as the method for determining which generators would be dispatched. For the first time competing generators were asked to submit bids to be ranked and cleared by an independent third party.

The methods for ranking bids, setting market prices, settling finances and providing some ancillary services were developed using economic supply demand theory and referred to as market design. Due to the nature of electricity as a commodity, part of the standard economic theory used in commodity markets required modification, with examples including the inability to warehouse or queue for the commodity. Electricity markets currently in use have been repeatedly modified and honed in order to obtain higher economic efficiency.

The United States Federal Energy Regulatory Commission (FERC) has issued a suggested Standard Market Design (SMD), with some electrical jurisdictions adopting it and others moving towards it. The result is that many of the numerous electrical jurisdictions have differing market designs. The success of these market designs varies, with arguably positive and negative aspects to each. The evolution of electricity markets is a continuous process in both academia and industry, with some methods eventually being implemented into physical electrical systems.

In order to be able to estimate the effects of a particular change in market design, in-depth studies are often completed before any changes are implemented. Many of the studies are based in social science and attempt to gauge the effects on stakeholders and the market place. While this is a highly useful exercise, the electrical commodity is quite different from others and drawing parallels with other commodities may not always prove accurate.

Effects of this were seen in the California energy crisis of 2000-2001. Due to a fundamental disconnect between the financial market and the physical aspects of the grid, the market provided great potential for gaming, to the benefit of many generators and financial players, but to the detriment of society [2].

For this reason there should be additional studies based on simulations that include both the financial and physical aspects of a power system. Physical aspects include items typically absent from social and economic studies, primarily effects on frequency and voltage levels. By observing the two fundamental metrics of a stable power system during the simulation of a market, one could identify whether the market operates to the benefit or detriment of the power system.

1.2 Objectives

The main objective of this thesis is to develop a real-time power system simulator used in the investigation of market designs and Automatic Generation Control (AGC) schemes. The scope of this thesis is the mathematical algorithms used in the simulator, hardware and software implementation, and validation of the implemented simulator.

Simulators have been widely used in the industry to measure, analyze and control power systems. Currently, there is a wide range of applications where simulators, or more generally, solution calculators, are being used. These include steady-state power flow problems, optimal power flows, transient and steady-state stability analysis and short circuit studies [3].

Each application has requirements on accuracy that dictate the complexity of the models used within their respective simulators. In turn, the complexity of the calculations dictates the speed at which the calculations can be performed, neatly dividing the applications into two groups. Simulations that can be performed in real-time or faster can be used in on-line applications, that is, have the potential to be used to control the power system itself. Simulations that take longer than real-time, while still highly useful, are used in off-line applications where the results can be applied to the power system with some latency. This latency can be on the order of minutes, hours, days or months, depending on the calculation and its application.

The purpose of limiting this simulator to the study of markets and AGC stems from the desired requirement of real-time simulation, coupled with the compromise between model complexity and calculation time. Power systems contain a wide range of time constants stemming from the different components of electrical power systems, primarily generation, load and protection. By identifying and eliminating portions of the system model not required to simulate the desired result complexity and, therefore, calculation time can be reduced.

Protection systems and electromagnetic transients operate at timescales orders of magnitude faster than that of the deliberate generation and load changes associated with markets and AGC. During a contingency such as a short circuit, protection systems should operate to isolate the fault, typically within hundreds of milliseconds. Once the fault is extinguished, the protection system will either be open or closed.

If the protection is closed, the system is at its original state and the simulation can continue as if the fault never occurred. A limitation to that statement is loss of angular stability in which the fault causes a generator to lose synchronism and be taken offline. The magnitude and potential negative results of such an occurrence are dealt with in dedicated stability studies. For the purposes of market and AGC simulation the assumption is that either the generator recovers and the system is stable or if taken offline the resulting system configuration is once again stable and a reserve market or AGC operates to replace the generation over a longer timescale, within the capabilities of the simulator being described.

In the case that the protection system is open after the fault, the power system is, therefore, in a reconfigured state and the simulator is required to reflect the change in the topology of the system. This requires a modification to the system model itself and can be accomplished through a second simulation using a 'new' power system model. Once again, transient stability in the transition between the two models is addressed using different tools. The assumption is that the system maintains stability and recovers using a market or AGC based system of some form. The question of power system dynamics is further addressed below.

Individual loads can operate quickly, although when aggregated tend to become quite predictable within a few percent of peak demand. The simulator should be able to reflect changes in load, although once again at a timescale much larger than that of the protection, and within the timescale of the capabilities of a market or AGC. In the case that a large portion of load is lost, (the probable case when dealing with large changes in load) a market or AGC would once again operate to modify generation over a longer timescale within the capabilities of the simulator.

To summarize, a simulator for the purposes of studying the operation of markets or AGC schemes should recognize and be capable of handling relatively slow changes in generation, load, voltage and frequency. Discontinuities such as the operation of protection devices, major contingencies, forced outages, loss of transmission or load and similar are assumed to leave the system in a stable state where a market or AGC can act to find the long term equilibrium point. The transients associated with such discontinuities are contained and dealt with in the larger field of power system stability, which has other tools for use in its analysis.

1.3 Overview

1.3.1 Purpose of Power System Simulation

Power system simulators, state estimators and economic optimal power flow calculators are core to the reliability of modern electricity systems and are used to predict and control the state of every major electricity jurisdiction. The prevalent operational model utilizes a centralized system, where Supervisory Control and Data Acquisition (SCADA) information from across the jurisdiction is transmitted to one or more Grid Control Centres (GCC), often owned and operated by the Independent System Operator (ISO). At the GCC, high powered computers perform complex mathematical calculations to determine the current state of the system and any control actions required to maintain stability. In [4], a detailed breakdown is given of the hierarchy and levels of control required to ensure this stability, including a number of both local and wide area controls. To summarize, local controls typically operate to maintain protection at the load, generation and protection component level, while wide area controls operate to maintain the stability of the system as a whole.

In order to properly control the power system, the ISO needs to know the current state of the system. Based on the current state of the system, decisions are made as to what actions will be required to ensure stability over the near-term time frame, i.e. minutes to hours.

1.3.2 The Power Flow Calculation

The basis for many of the calculations integral to the stability of the system is the Power Flow (PF). Given the physical characteristics of the power system, as well as inputs regarding generation and load, the PF calculates the voltages on every bus in the power system, as well as the required real and reactive power levels required to balance supply and demand and keep voltages near nominal values. The calculated values can be compared to actual measurements in order to verify the state of the system, and the calculated power levels can be used to dispatch actual generation across the electrical jurisdiction. The decision as to which generation is dispatched is often made using market economics, constrained by physical system metrics such as system congestion and security.

Since the power flow is a simulated version of the power system, alternate (i.e. theoretical) values can be used as inputs to provide operators with estimates of the potential state of the power system under hypothesized conditions. This is an extremely useful analysis for contingency planning, such as generator forced outages, transmission line failures and rapid load changes. These potential system states are put into the PF to estimate the condition of the power system after such changes. System operators can make decisions that ensure a stable system taking into account the current state of the system, as well as potential system configurations after hypothetical system disturbances.

The power flow is a static method, providing a snapshot of the power system at any given moment in time. In its basic form it does not contain any time based information and does not provide any information regarding the dynamics of the power system. Additional tools for such purposes are available to system operators but are outside the scope of the current discussion.

The power flow is a highly used and valued tool in the system operator’s toolkit, therefore there has been significant research into reducing the time required to perform the calculation and obtain the results. Industrial grade systems largely use the Gauss-Seidel and Newton-Raphson algorithms, as well as a number of modifications to, and variations of, these algorithms. Due to the mathematical nature of the methods, the speed with which the solution can be obtained is highly coupled with the method of hardware and software implementation. These methods are introduced and discussed in depth in Chapter 2 - Literature Survey.

1.3.3 Required Modifications to the Power Flow

The power flow is a tool used to find steady-state system voltages and required real and reactive power outputs. Alternatively, the application of power system simulation attempts to find the state of the system, based on the generation and load inputs. The state of generation and load are functions of their own respective or central control systems.

In other words, ISOs want to know what generation dispatch orders are required in order to stabilize the system and, therefore, use the power flow calculation to obtain that information. The application of the power flow calculation to power system simulation requires that the ‘control and dispatch’ mechanism be removed from the algorithm and dealt with separately.

In the case of real power, if generation and load are not in balance, then the PF provides the required dispatch in order to balance the system. In simulation, the imbalance manifests itself as a frequency deviation, upon which some as-yet undefined control system would act to correct the imbalance.

This is also true for reactive power output. The PF calculates required reactive power to maintain a desired (pre-set) bus voltage. In simulation, the bus voltage is largely a function of the reactive output of the generator at that bus. The duty of maintaining the desired bus voltage lies with the as-yet undefined control system.

The PF will therefore be modified as required and referred to as the Raw Power Flow (RPF). Raw implies the removal of the control aspect of the power flow algorithm. RPF provides the solution to the bus voltages, as well as a value known as the mismatch. With the control aspect removed, the RPF no longer attempts to balance the system, and the resulting error manifests itself in the mismatch value. It is this value that will further be used in the dynamic calculations. Inputs to the RPF are limited to the actual generation and load without any predefined voltages, save the reference bus.

To summarize, the goal is to identify what the system wants to do given the inputs, as opposed to what needs to be done to get the system to the desired operating state. That function is left to the control systems, often markets and AGC schemes in the case of real power or direct closed loop control systems in the case of reactive power.

1.3.4 Adding System Dynamics

The power flow is a static algorithm providing a snapshot of the state of the system. Under the assumption that the simulator is restricted to slow system dynamics, it is possible to utilize the power flow to provide a large number of successive snapshots, each slightly different from the previous to provide a time based simulation. An analogy would be a film or television broadcast, which is a series of static frames which when played successively provide the sensation of movement. Such modelling concepts are described in [5] and applied to voltage stability analysis. The method allows for higher computational efficiency by using an equilibrium model for the network while providing the level of accuracy required for the events under study.

The concept contains two issues to overcome in order to be practically implemented. The first being the large number of calculations required to perform the PF and the second being the PF does not contain any information regarding the frequency, a key indicator of power system state.

In order to provide a time varying simulation in a reasonable amount of time, the speed at which the PF is performed must be increased. There are two primary techniques described in this thesis that attempt to provide this functionality, both encapsulated within the method of calculation and its associated implementation. The first is the use of a long-ago discarded method for finding the power flow solution, the Jacobi method. The second lies in the use of recent advances in parallel computing technology.

Once a single power flow calculation can be performed in a short enough time to provide fast successive ‘frames’, the required system dynamics must be added to the simulator, i.e. the frequency component. Turning to the area of transient generator stability provides a suitable tool, the generator swing equation. Certain modifications are made in order to apply the equation to a time based simulation.

The novelty in this implementation of a power system simulator is implementing a neglected numerical method on a recently available hardware platform. This allows for fast computation time such that the steady-state power flow calculation can be used as the foundation for a dynamic real-time power system simulator.

1.3.5 Generator and Load Models

While the focus of the thesis is the development of a power system simulator, it would not be possible to test the simulator without the inclusion of generator and load models. For the purposes of preliminary testing the models developed were basic, in order to focus development efforts on the simulator.

Load models are constant power, both real and reactive. It is expected that when used in further research or simulations, the load models will be further developed to include sensitivity to voltage and frequency changes, elasticity to power prices, typical changes in load level over time as well as stochastic changes to reflect short term fluctuations.

Each generator is modeled as a simple rotating mass with variable real output power. The frequency of the system is calculated using the rotational inertia of the entire system, which is obtained through the summation of the rotational inertia constants of each of the generators. A variation of Newton's Second Law uses the system inertia and real power imbalance to calculate the system frequency.

Since perfect power balance is improbable, each generator model contains a control system that varies real power output according to frequency, known as droop control. The generator(s) reduce output linearly if the frequency is above 60Hz, and increase output linearly if the frequency drops below 60Hz.

Similar to load models, when the simulator is to be used for further research and simulation, the generator models would be further developed to include transient dynamics, market based generation set-points and voltage control.

1.3.6 Validation

With the solution algorithm and equations defined, the hardware and software implementation complete, and generator and load models designed and implemented, the task turns to validating the results to ensure that the simulator produces reliable output.

Academic and industry grade power flow programs are readily available and can be used to provide third party results against which to compare the simulator results. In order to place both systems on the same level, comparisons are only made to the simulator once it reaches steady-state values after a disturbance. If the two steady-state results match, the simulator will be considered to be producing accurate results.

For the purposes of validating the dynamic calculations, the equation is rather rudimentary and literature has shown that similar approximations have been implemented for use in other areas of study. A series of manual calculations given the same inputs for comparison should be sufficient to ensure numerical accuracy and validate the results.

1.3.7 Results

Design metrics for considering the operation of the simulator a success are described in Chapter 3 - Theory. To summarize, the simulator will use generation and load power levels as inputs, perform a single power flow calculation in less than 0.033 seconds and provide system voltages and frequency as outputs. The system size is that of typical power systems in the 3000 bus range.

In order to test the performance of the simulator, a number of potential scenarios are developed and tested. By comparing the scenarios, the effectiveness of each portion of the simulator can be estimated.

To measure dynamic response, three scenarios are used. The first is two step changes in generation, the second a slow continuous ramp of generation to observe response to a constant change, and the final scenario two generation ramps in different directions. As loads are considered to be constant power, the simulations will not include changes in load levels. The ability to vary load will be developed in more advanced load models as described previously.

In order to measure the success of the parallel Jacobi implementation, the simulator is run using larger systems with different numbers of processors. Small systems do not require enough computations in order to be able to take advantage of parallel processing, and are of little practical significance.

Chapter 2.
Literature Survey

The literature survey begins with a general review of power system simulation with the purpose of identifying and justifying the scope of the simulator under development. After reviewing the major categories of simulation currently in use, the desired level of approximation and the inputs and outputs are selected.

The second section of the survey focuses on parallel processing implementations of power flow algorithms, which are inherently faster than serial algorithms. The three algorithms for solving the power flow problem are Newton-Raphson, Gauss-Seidel and Jacobi. The survey reviews the state of research for parallel implementation of each algorithm and assesses the potential maximum speedup using current parallel processing technology.

The final section presents a paper that provides an example of the type of simulation decided upon in the balance of the survey. While not identical to the proposed implementation, the paper does illustrate a method for modifying the power flow to include the dynamics required for market and AGC simulation as described in the first section.

2.1 Power System Simulation

Power system operators and planners use a wide range of tools to study and control electrical power systems. The majority of power system tools fall into one of two major categories, those used to analyze a spectrum of possibilities for a given system or event, and those used simulate a given sequence of events or the state of a system. Each has different implementations and objectives.

Examples of tools that analyze spectrums of possibilities include the continuation power flow for voltage stability and bifurcation analysis for voltage and angular stability. To generalize, such tools give the user an indication of where the system is, and how far a certain variable can be changed before the system becomes unstable. The purpose of the simulator being designed is to perform a time based simulation using markets or AGC as the control system. Therefore, the use of such tools to examine the potential stability consequences resulting from the operation of the market and/or AGC under test would be an interesting exercise, but are irrelevant to the design of a time varying simulator.

Tools that simulate events or states typically fall into one of four categories, steady-state, long term dynamics, transient conditions, and fast transients. Each category requires different levels of accuracy to obtain the desired result. This is reflected in the complexity and level of abstraction in the mathematical models used.

2.1.1 Fast Transient Simulation

Fast transients are the most complex of the four categories and use full models for all the components in the system under study. The component models take into account all the electrical and physical properties of the device, whose equations reflect all electromagnetic transients with time constants in the millisecond range and below.

Examples of such simulations include lightning strikes, capacitor switching overvoltage, and fault and circuit switching transients. In many cases such events are damped rather quickly and have little lasting effect on the power system in the long term, provided that protection systems have not operated, stability is maintained and no damage to the devices has occurred.

While highly accurate [6], such simulations require significant computing power to perform in a reasonable timeframe [7]. Such systems are generally known to be quite expensive.

For the purposes of simulating a market or AGC, a number of assumptions are made to simplify the calculation, one of which is occurrences such as those listed above can be ignored. For this reason, the use of the fast transient level of simulation would be overkill for the application and would only serve to intensify the computation, increase the calculation time, and/or require additional hardware to attain the design metrics. While cost is not strictly a design metric, optimizing the cost to benefit ratio is an overarching goal in most activities.

2.1.2 Transient Simulation

The next level of simulation is that of transients. Examples include generator speed control, electromechanical oscillations and primary voltage control. The system model used in such studies includes individual models for each component, with larger mathematical abstraction and, therefore, less accuracy than used in fast transient studies.

Electrical phenomena with time constants less than 0.1s, such as travelling waves on transmission lines or switching transients, are ignored or considered to be instantaneous [8]. By abstracting such phenomenon the models are simplified, resulting in sets of differential equations describing the simplified electromechanical characteristics of the components. The calculations required to perform such simulations are still quite complex, although much simpler than fast transient analysis. Note that transient models are also utilized in bifurcation analysis as described earlier [9].

While the action of market control or AGC will produce slight generator oscillations and other transient phenomenon, an assumption is made that the changes occur on a timescale much longer than the damping time of such events and can therefore be ignored. In addition, it is assumed that the transient phenomenon are in fact transient and do not lead to system instability. The stability of the system after such control actions is another matter dealt and with in stability studies.

2.1.3 Long Term Dynamic Simulation

Long term dynamic simulation reflects slowly changing conditions corresponding to normal system operation. This includes normal variations in load, ramping due to generation dispatch, automatic generation control, and voltage and frequency regulation activities.

The system and component models are significantly less complex than either form of transient simulation. The primary assumption is that of 'tied-rotors', that is, there is a single system frequency and no generator oscillations. While the frequency can change, it does so uniformly around the system.

The models for generation take into account the output set points, limits and delays, but much (if not all) of the transient dynamics are removed depending on the purpose of the simulation [10]. Load models can vary widely, from simple constant power to time, frequency and voltage dependant. Once again, the nature and complexity of the model is largely defined by the objectives of the simulation.

Long term dynamics are adequate for simulating the actions of markets and automatic generation control [11]. It is assumed that the system is in a stable operating condition and that generation and load will change but not affect system stability. The use of power flow based simulators is often used in stability studies as is discussed in [5]. Such functionality is a possible avenue for further development, although outside the scope of the use of the simulator described in this thesis.

2.1.4 Steady-state Simulation

In steady-state, it is assumed that all the variables are constant. Under this assumption, one can remove all the dynamic portions of the system and component models. This greatly simplifies the calculations required to obtain the solution and increases the speed at which the solution can be found. Examples of such tools include the power flow and optimal power flow calculations.

For the purposes of market and AGC simulation, steady-state techniques do not provide the dynamics required to observe the desired phenomenon of varying voltages and frequency. Fortunately, the network models for steady-state power flow calculations are the same as that used in long term dynamic simulations [12]. The dynamics come as a result of changing the models of the generators and loads to provide variable output and include frequency and voltage deviations.

2.1.5 Simulator Scope

Of the four levels of simulation, only long term dynamics are of interest in market and AGC studies. The goal of this thesis is then to take the power flow method that normally resides in the steady-state category and modify it to fall into the long term dynamics category. The other two categories provide significantly more accuracy than required at the cost of calculation time.

The simulator will utilize the power flow method to calculate the system voltages and the active power balance. Using the residual power balance, the frequency can be calculated using dynamic generator models operating under the 'tied-rotor' assumption.

2.2 Parallel Power Flow Algorithms

Power flow has been used extensively in academic and industry settings. It is a powerful tool used to calculate the voltages given generation and load inputs. Its typical use is to estimate the state of an electrical system given partial state information. Grid control centres use it as the first operational program when performing the calculations and operations required in finding the dispatch orders required to ensure steady-state system stability. As well, the power flow algorithm is an integral part of a number of other calculations such as voltage stability analysis and continuation power flows, security constrained optimal power flows and system optimization.

The amount of literature on the power flow is large but falls into a relatively small number of varieties. The method in which the calculation is performed can be split into six major headings using two criteria, the solution algorithm and the solution implementation. The three primary algorithms used to find the solution are the Jacobi, Gauss-Seidel and Newton-Raphson algorithms, each of which has various serial and parallel implementations.

As these are mathematical computations, the method of implementation plays a large part in determining the computational efficiency, and therefore the computation time of the algorithm. Due to the important nature of the power flow calculation, a significant amount of research has been performed in order to accelerate the computation.

The first digital power flow calculators were based on computers that had only one processor. Thus, serial algorithms and techniques were developed in order to reduce the number of computations and memory accesses required to perform the calculations. There are a number of methods in which this can be accomplished.

Since the relevant matrices are quite sparse, there are a number of mathematical and computational methods available in order to reduce their storage requirements (i.e. not storing large quantities of zeros) as well as removing the large numbers of multiplications of values by zero. In the case of the Jacobi and Gauss-Seidel algorithms, pre-processing and ordering techniques help to obtain these efficiencies. In the case of the Newton-Raphson algorithm, the largest computational portion consists of calculating and decomposing the Jacobian using sparse matrix techniques. Due to its sparseness there are a number of mathematical and computational techniques that assist in reducing the number of computations required.

There is extensive literature dealing with serial algorithms but parallel processing has made such significant advances in accelerating the algorithms to the extent that, for the purposes of this discussion serial algorithms will be omitted and the focus will remain on implementation of the three algorithms, keeping in mind that many methods for improving performance are standard in the field of matrix computation, and many have been incorporated into both serial and parallel implementations.

With the decrease in hardware costs and the development of faster inter-processor communication, the next major advance in the calculation of the power flow solution came in the form of parallel processing techniques, with the specific techniques for parallelization of each algorithm taking different forms due to inherent differences in their formulation. All three techniques attempt to solve the same set of equations or some mathematical rearrangement thereof, the power flow equations:

$$S_k = P_k + jQ_k = V_k\left[\sum_{n=1}^{N} Y_{kn}V_n\right]^* \qquad \text{k=1..N} \qquad (2.1)$$

Where: N is the number of buses

k refers to an individual bus

S, V and Y are complex numbers

In [13], an in depth explanation is provided of the intricacies of parallel processing with application to a number of power system related problems. The focus here is on the portion regarding the power flow problem.

2.2.1 Parallel Newton-Raphson Algorithms

Newton-Raphson is currently the industry choice for solving the power flow problem. The algorithm involves iteratively performing the following process:

Step 1: $$x(i)=\begin{bmatrix}\delta(i)\\V(i)\end{bmatrix} \tag{2.2}$$

Step 2: $$\Delta y(i)=\begin{bmatrix}\Delta P(i)\\\Delta Q(i)\end{bmatrix}=\begin{bmatrix}P-P[x(i)]\\Q-Q[x(i)]\end{bmatrix} \tag{2.3}$$

Step 3: $$\begin{bmatrix}\Delta\delta(i)\\\Delta V(i)\end{bmatrix}=\begin{bmatrix}J1(i) & J2(i)\\J3(i) & J4(i)\end{bmatrix}^{-1}\begin{bmatrix}\Delta P(i)\\\Delta Q(i)\end{bmatrix} \tag{2.4}$$

Step 4: $$x(i+1)=\begin{bmatrix}\delta(i+1)\\V(i+1)\end{bmatrix}=\begin{bmatrix}\delta(i)\\V(i)\end{bmatrix}+\begin{bmatrix}\Delta\delta(i)\\\Delta V(i)\end{bmatrix} \tag{2.5}$$

Where J1, J2, J3 and J4 constitute the Jacobian matrix based on the partial derivatives of P and Q with respect to V and δ. The most computationally intensive portion of the algorithm is constructing, calculating and decomposing the Jacobian using sparse matrix techniques, and limits the ability for the algorithm to be highly parallelizable.

A number of papers investigate accelerating the NR algorithm by parallelizing various portions. It was attempted to omit variations on the NR algorithm such as fast decoupled and DC power flow. These techniques are approximations and could potentially introduce significant error into the final solutions, causing the simulator to operate improperly. As a significant portion of the body of literature focuses on approximate algorithms for performing power flow calculations, a number of the papers investigated are done so under the assumption that the techniques may possibly be valid for application to the full NR algorithm.

In [13] the author describes the portions of the algorithm which can be performed in parallel, as well as some of the theoretical limitations to this process. A number of algorithms exampled in this paper attempt to apply mathematical re-ordering in order to find parallelism in an inherently sequential algorithm. While this does improve performance, it has inherent limitations as stated.

There are a number of papers that directly address the problem of NR parallelization and describe the difficulties in depth. In addition, the papers provide potential solutions.

In [14], the author utilizes re-ordering as mentioned above. The paper describes in depth the application of LU factorization to the matrices in solving non-linear systems of equations, as well as methods to find parallelization in the process. The implementation allows for a speed-up of 13-20 times the original calculation speed.

In [15], the author provides a method to parallelize totally the Newton power flow process on a distributed memory architecture with message passing between the computing nodes. This is a misleading statement, as the NR algorithm has inherent coupling between portions of the calculation. The statement should more accurately be phrased as the author has found a method to identify all steps of the NR method in a parallel manner, although within every step there are limitations to the amount of the algorithm that can be performed in parallel. The speedups are up to 6 times the original single processor calculation speed.

A good paper with which to draw a parallel is [16]. It uses the NR algorithm with a different technique applied, that of the Generalized Minimal Residual method (GMRes). Using this methodology, speed ups of twice the previous paper were attained.

The final paper in the NR category is [17], which focuses on transient simulation. While this is not the calculation being performed in this thesis, it nonetheless performs similar calculations and the results may provide insight. The algorithm is still the NR using successive over-relaxation. The parallelization method utilizes a 'travelling window', which means that portions of the next iteration can begin once portions of the current iteration are complete. This is not a powerful parallelization technique as the theoretical maximum parallelization is not great compared to inherently parallel algorithms, and the potential number of parallel processors is quite low. As in previous papers, speed up was in the range of 12 times the original speed.

Note that it is difficult to compare the papers above in terms of absolute time, as the systems under test were not necessarily the same. This is a significant drawback in comparing calculation speeds as few papers in the literature utilize a standard test system.

2.2.2 Parallel Gauss-Seidel Algorithms

Moving to the Gauss-Seidel algorithms, we return to [13]. The blanket statement by the author is that, to paraphrase, the GS algorithm is not suitable for parallel processing however variations exist which are. The GS algorithm requires the following calculation to be performed repeatedly until the system converges.

$$V_k(i+1) = \frac{1}{Y_{kk}}\left[\frac{P_k - jQ_k}{V_k^*(i)} - \sum_{n=1}^{k-1} Y_{kn}V_n(i+1) - \sum_{n=k+1}^{N} Y_{kn}V_n(i)\right] \tag{2.6}$$

It can be seen in the structure of formulation that the calculation of the present voltage has dependence on a portion of the voltages, namely those between 1 and k. It is this portion of the GS algorithm that limits its ability to be a fully parallel algorithm.

The first example of an attempt to find parallel components of the GS algorithm is contained in [18]. This paper does a thorough review of previous GS algorithms and addresses the bottlenecks in computation. The first method for finding parallel portions is known as colouring, or ensuring that portions of the calculation sent to each processor do not have relationships with portions sent to other processors. This is similar to the re-arranging method applied to the NR algorithm. Once again, these are procedures for finding available parallelism in an inherently sequential algorithm. This leaves us with speed ups on larger systems of approximately 9 times.

The next paper [19] provides another example of an ordering technique. The method of rearranging the calculation in order to find parallelism is repeated, this time with a two-step system instead of the colouring technique utilized in the previous example. This allows the author to attain speed ups of approximately 12 times, similar to the balance of the papers described.

2.2.3 Parallel Jacobi Algorithms

The final of the three primary power flow algorithms is the Jacobi, a variation, or more properly, a simplification of, the GS algorithm. As is shown, the formulation is almost identical, with the dependency between rows removed by the use of voltage values from the previous iteration only.

$$V_k(i+1) = \frac{1}{Y_{kk}}\left[\frac{P_k - jQ_k}{V_k^*(i)} - \sum_{n=1}^{k-1} Y_{kn}V_n(i) - \sum_{n=k+1}^{N} Y_{kn}V_n(i)\right] \qquad (2.7)$$

This can be re-written as:

$$V_k(i+1) = \frac{1}{Y_{kk}}\left[\frac{S_k^*}{V_k^*(i)} - \sum_{n=1,n\neq k}^{N} Y_{kn}V_n(i)\right] \qquad (2.8)$$

As the formulation shows, for every bus k of N buses, the calculation does not depend on any other bus during the same iteration. The caveat, and the primary reason that the Jacobi algorithm has fallen to the wayside in recent years, is that it has more difficulty converging than the GS.

The concept of parallelising the Jacobi algorithm was first attempted long ago. In [20], the author utilizes complex mathematical methods in order to find the Eigenvalues of a matrix. While not directly applicable to the power flow, the paper shows the power of parallelization in the Jacobi algorithm.

Another example of the utilization of Jacobi to find Eigenvalues is described in [21]. The author provides a method for partitioning the problem and in some cases utilizes as many as 256 processors, the first example of highly parallel execution. While speed up values are not provided the calculation is that of Eigenvalues and, therefore, not directly applicable to our application.

An example of parallel Jacobi to the power flow problem is discussed in [22]. While the title indicates that a Gauss-Seidel algorithm is being utilized, further investigation shows that the algorithm is in fact the Jacobi algorithm. This paper describes a novel method of using dedicated hardware for performing a number of the calculations. A Field Programmable Gate Array (FPGA) is implemented to perform the complex math required to solve the problem. In addition, the author performs optimisation in order to gain efficiency in the parallelisation of the complex calculations themselves, which often have additional parallelism internally. For example, the multiplication of two complex numbers consists of four parallel multiplications, followed by a parallel addition and subtraction:

$$(a + jb)(c + jd) = (ac - bd) + j(bc + ad) \qquad (2.9)$$

The paper continues to describe the speed up of approximately 5 times, which falls below the speed up achieved by the parallel NR and GS algorithms described previously. With the additional iterations required for the Jacobi algorithm, it appears that the implementation does not provide significant gains. This is addressed by the author, stating that the comparison is between a 1.5GHz processor versus a 100MHz FPGA. If a 1.5GHz FPGA hardware solution could be found, the speedup would be on the order of 5x(1500/100), or 75 times. This would provide significant speed up over other algorithms.

Given the advances of parallel computing hardware and software, it was decided that the potential for the Jacobi algorithm outweigh the potential for the NR or GS algorithms, without the advancement of the complex mathematical methods and their implementation on both hardware and software. The natural parallelism inherent to the Jacobi algorithm allows simple and effective implementation on parallel hardware, and it seems that the potential speed up is limited by hardware technology. Since the publishing of the previous paper in 1996, a number of new technologies have emerged in the area of high performance parallel computing. These technologies will be discussed in Chapter 4 - Implementation.

2.2.4 Example of Dynamic Parallel Power Flow

Only one paper was found that attempted to provide a simplified dynamic model utilizing a similar method as the one developed for this thesis, although the implementation focused on using one of the dishonest NR algorithms known as normal fast decoupled NR power flow.

In [23], the author performs a complex variation of the power flow (referred to in the paper as a load flow) and breaks the problem down into three portions, the load, the transmission losses, and the generation/load imbalance. By using participation factors of the individual generators and assigning portions to each, the imbalance is distributed across the generators, and referred to as the net accelerating or decelerating power.

The author states that using standard power flow techniques, the power imbalance would manifest itself at the reference bus, and potentially result in incorrect results for system imbalance. This may be true and the practice of distributing the imbalance could provide greater accuracy, but the complexity of the solution may also outweigh any additional accuracy. In addition, a single slack generator is the standard implementation for power flow solutions to find the required slack generator output. If such an implementation is utilized in industry with great success, it is interesting that the author believes it would not work for the calculation of system imbalance.

The paper is a good attempt to add dynamics to the power flow, although fails to actually utilize the power imbalance to calculate the system dynamics, that is, the frequency.

From the results of the literature survey, it is concluded that this thesis will utilize the Jacobi power flow solution algorithm to perform a sequence of snapshots of a dynamic power system, using power imbalance as described above to calculate deviations in frequency between the snapshots. The process will thereby produce a discrete time power system simulator that uses generation and load as inputs to produce bus voltages and a system wide frequency as outputs. As mentioned previously transient phenomena are the realm of transient and fast transient power system simulation and, therefore, not part of the design criterion.

Chapter 3.
Theory

As described in Chapter 1 - Introduction, the main objective of this thesis is to develop a real-time power system simulator used in the investigation of market designs and AGC schemes.

In order to attain the main objectives, the simulator is required to calculate the voltages and frequency of the simulated system at every time step. While the generation and load levels are also required, these are considered inputs to the simulator and the models of each are not intrinsic to the simulator itself.

This chapter describes the mathematical theory and thought process used during the development of the simulator. It begins with modifications performed to the standard Jacobi power flow algorithm and reasons for doing so. The second section describes the addition of system dynamics. The third and final section discusses areas for possible future improvement to the simulator.

3.1 Standard Jacobi Power Flow Algorithm

From the results of the preceding literature survey, it was decided that the Jacobi algorithm would be implemented due to the potential of using recent advances in parallel processing technology to increase the speed of calculation.

The increase in calculation speed is required as typical large scale power flow calculations may take as long as a few seconds. If the purpose of the simulator is to simulate the market and AGC, then the time step used in the simulation must be significantly smaller then the events under observation. From a system perspective, changes in load and generation occur on a longer term timeframe such as on the order seconds to minutes. With this in mind, it would be reasonable to state that the simulator must have time steps on the order of less than one second.

If the goal of the simulator is to be able to provide real-time performance, the desired time step is selected to be less than 1/30, or 0.033, seconds. This decision results from the video industry, where such a refresh rate allows the human brain to visualize a sequence of stationary frames as motion without any visual motion artifacts. While not important for a numerical solution, this becomes important in the case of real-time visualization. Note that this is a goal for perfect visualization and not reaching the full refresh rate does not indicate failure of the system. Refresh rates much slower than 0.033 seconds are typical in real-time power system simulations.

In order to ensure that the simulator can attain the 0.033 seconds refresh rate all actual simulations referred to in Chapter 5 - Results are done using a time step of 0.01 seconds, increasing the calculation rate by three times.

The normal use of a power flow solution is to find the state of an electrical system given the inputs of generation and load. Typically, the algorithm also finds the real power generation required at the slack bus such that the system is balanced in steady-state, as well as the reactive power generation required in order to hold the voltage at certain buses within limits. In short, a power system under normal steady-state operation typically meets the following criteria:

1) The generation must equal the load plus all losses.
2) Bus voltage magnitudes should remain close to rated values.
3) Generators operate within specified real and reactive power limits.
4) Transmission lines and transformers are not overloaded.

Standard power flow algorithms calculate the complex nodal voltages for each bus in a power system under study. Once complete, the real and reactive power flows can be calculated.

In large power systems, input data is normally given in terms of real and reactive powers instead of impedances and/or currents. For that reason, the system of equations becomes non-linear and requires a numerical solution method in order to obtain the solution.

The system of equations required to be solved for the power flow are as follows. First, the real and reactive powers at any given bus k are defined as the difference between the generation and load at the bus:

$$P_k = P_{Gk} - P_{Lk} \qquad (3.1)$$

$$Q_k = Q_{Gk} - Q_{Lk} \qquad (3.2)$$

In a typical power flow simulator, the buses are divided into one of three types, swing, load and voltage, also referred to as the reference, PQ and PV buses.

There is only one swing bus and is used as the voltage reference bus. At this bus, the voltage magnitude and angle are fixed and do not change during the solution process. This is required in order to obtain meaningful results. Without the inclusion of a reference voltage angle the entire system of equations would be under-constrained and contain an extra degree of freedom. Holding one of the buses at a constant voltage angle provides the system with a voltage phasor reference.

Another name for the reference bus is the swing bus. In standard power flow calculations, the swing bus contains the swing generator. In power flow studies, the loads and desired generator set points are typically known in advance and used within the power flow. What is unknown is the losses and imbalance in the system. For this reason, the real power output of the generator at the reference bus is allowed to change (i.e. swing) and absorb the losses and imbalance to produce a balanced steady-state solution for the system. At this bus, the algorithm will calculate P_k and Q_k, or the real and reactive powers.

At load buses, P_k and Q_k are given as inputs and the algorithm calculates the complex voltage, that is, the voltage magnitude (V) and angle (δ).

Buses with generators or other sources of controllable reactive power (Q) are considered to be PV buses. At these buses, the bus voltage magnitude is held constant and the algorithm solves for Q_k and the voltage angle.

The three types of buses and their parameters are listed in Table 3.1.

Table 3.1 - Standard Power Flow Bus Types

Bus Type	Given	Solved
Swing (Reference)	V δ	P Q
Load (PQ)	P Q	V δ
Voltage (PV)	P V	Q δ

In order to be able to calculate the power flow solution, the physical characteristics of the system are required. This is given in the Ybus matrix, which contains the admittances of every static device on the grid. This includes transmission lines as well as any series or shunt admittances. The Ybus matrix is typically symmetric, although may be unsymmetrical if certain devices are present within the system, an example being phase-shifting transformers.

The admittance model used for transmission lines is the π model. Using this model, short, medium and long steady-state models can be incorporated into the simulator, as well as passive shunt and series devices. As noted in the literature survey, steady-state and long term dynamic network models are identical. Devices such as synchronous condensers would be modeled as generators and are therefore outside the scope of this discussion.

With the Ybus matrix defied, the nodal equations for the network are determined:

$$I = Y_{Bus}V \qquad (3.3)$$

$$I_k = \sum_{n=1}^{N} Y_{kn}V_n \qquad (3.4)$$

$$S_k = P_k + jQ_k = V_k I_k^* \qquad (3.5)$$

Which results in the power flow equations for a general system:

$$S_k = P_k + jQ_k = V_k\left[\sum_{n=1}^{N} Y_{kn}V_n\right]^* \qquad \text{where k=1..N} \qquad (3.6)$$

In many power system simulators the Gauss-Seidel or Newton-Raphson algorithms would be applied, but in this implementation the Jacobi iterative substitution algorithm is applied to obtain the following:

$$V_k(i+1) = \frac{1}{Y_{kk}}\left[\frac{S_k^*}{V_k^*(i)} - \sum_{n=1,n\neq k}^{N} Y_{kn}V_n(i)\right] \qquad (3.7)$$

Where: N is the number of buses

k refers to an individual bus

S, V and Y are complex numbers

When computing the power flow calculations, there is one instance of the above formula for every bus in the power system. For example, a 2000 bus system would have 2000 equations, labelled $V_1(i+1)$ to $V_{2000}(i+1)$. Each of the equations is independent of each other during a single iteration and can be calculated simultaneously. As will be shown in Chapter 4 - Implementation, the efficiency of the Jacobi algorithm on parallel processing hardware is the ability to compute every equation simultaneously. In theory, the example 2000 bus system could use 2000 processors to complete the entire calculation in the same time as a single calculation.

The voltages are iteratively calculated until the stopping criterion is met. Often, the stopping criterion is the mismatch, calculated using the following formula.

$$\Delta S = S_k - V_k\left[\sum_{n=1}^{N} Y_{kn}V_n\right]^* \qquad (3.8)$$

Simply put, the mismatch for a bus is the difference between the known and calculated complex powers at the bus. If there is a significant difference between the two, then the system has yet to converge. Normally, the stopping criterion is that the magnitude of the mismatch on every bus must be below a desired threshold.

The standard power flow solution includes a number of additional steps in order to deal with occurrences such as real and reactive power limits on generators, but for the purposes of this thesis the discussion of standard power flow calculation algorithms is concluded, and the development of the power flow system under discussion begins.

3.2 Raw Jacobi Power Flow

The goal of the thesis is to develop a simulator and not a power flow calculator, and therefore requires a number of modifications to the PF that must be made in order to suit the application.

The first is the embedded 'control systems' within the algorithm. The standard algorithm actively changes the output of the real and reactive power set points of certain generators at the reference and PV buses.

At the reference bus the power flow algorithm maintains the voltage of the bus to the desired value. In order to allow the required degrees of freedom in the equations, it allows P and Q to change. As the goal is to modify the power flow algorithm for the purposes of simulation, having the swing generator modify its output based on the results of a calculation is not realistic. For this reason the constant voltage magnitude and angle criteria is maintained, although the portion which calculates the real and reactive powers is removed. In essence the swing bus is being modified into a load bus, but maintaining the voltage reference portion.

At first glance, it may seem that removing the variable generation at the slack bus would result in removing a degree of freedom from the system of equations, thereby over-constraining them. In fact, the swing bus power values come from reducing another portion of the calculation, the mismatch at the reference bus, to zero. By constraining the power and removing the mismatch constraint on the reference bus, the over-constraint on the system of equations is relieved. At the completion of the calculation, the mismatch at the reference bus will not be zero, and reflects the generation/load imbalance in the power system. This value is later used in the dynamic portion of the simulation to calculate the frequency deviation.

This discussion holds for the balance of the voltage constrained buses (PV) where standard power flow algorithms modify the reactive power output of a generator or synchronous condenser to obtain the desired bus voltage. As a simulation, not a power flow calculation, is being performed it is desired that the bus voltage reflect the actual reactive output of the generator. Therefore, all buses on the power system are considered to be load (PQ) buses and therefore have set real and reactive power levels. As such, the duty of regulating the generator reactive power output falls onto the voltage controller, part of the generator or grid control system and therefore outside the scope of the simulator. The modified list of bus types is given in Table 3.2.

Table 3.2 - Raw Jacobi Power Flow Bus Types

Bus Type	Given	Solved
Reference	V δ	Mismatch
Load (PQ)	P Q	V δ

With the power flow portion of the simulator complete, the development of the dynamic portion of the simulator can begin.

3.3 System Dynamics

As stated in the previous section, the power flow was modified to remove the slack bus concept, resulting in a residual mismatch at the reference bus. Numerically, this residual is equivalent to the amount that the slack generator would have modified its output to allow the system to converge.

In this section, a method is described to provide the simulator with dynamic response through the addition of frequency. As described in the introduction and literature survey, generator oscillations and fast dynamics will be omitted.

The fundamental equation describing a rotating machine is the swing equation.

$$\frac{2H}{\omega_{syn}}\omega_{p.u.}(t)\frac{d\omega(t)}{dt} = p_{mp.u.}(t) - p_{ep.u.}(t) = p_{ap.u.}(t) \qquad (3.9)$$

Where: H is the normalized generator rotational inertia constant

ω_{syn} is the synchronous frequency of the system

$\omega_{p.u.}(t)$ is the per unit system frequency at time t

$p_{p.u.}$ are the per unit mechanical, electrical and accelerating powers

This equation is typically used in studying the motion of synchronous machine rotors in transient generator stability studies. Similar equations could be used to study inter-generator dynamics and oscillations. For the purpose of long term dynamic simulation the equation is modified slightly.

The H in the swing equation relates to the rotational inertia of the generator in question. As stated in the literature survey, long term dynamic studies operate under the 'tied-rotor' assumption, neglecting inter-generator oscillations and the entire system contains only a single frequency. Therefore, H is used to reflect the rotational inertia constant of the entire system. The method for obtaining this value is to simply sum the H constants of each of the generators together [24].

The accelerating power refers to the system wide imbalance between generation and load, inclusive of any losses. From the results of the modified power flow, the reference bus real power mismatch provides this value. This is an approximation as the imbalance in a physical power system does not manifest itself at a single bus, but is rather spread out among the generators in the system. For the purposes of coarse system control such as markets and AGC, such an approximation makes little difference in the results and, as described in the literature survey, is often used in power flow calculations. Admittedly, dividing the imbalance among the generators and removing the tied-rotor assumption is an area for further development. This would allow for a greater accuracy in calculating the imbalance as viewed from a system perspective as opposed to the perspective of the reference bus, as well as allow for the simulator to simulate inter-generator oscillations and local variations in frequency.

For discrete time simulations, the use of full differential equations is not required. The equation is modified removing the derivatives and replacing them with division.

$$\frac{2H}{\omega_{syn}}\omega_{p.u.}(t)\frac{\Delta\omega(t)}{\Delta t} = re\left(\Delta S_{Bus1}\right) \qquad (3.10)$$

and rearranging for $\Delta\omega(t)$,

$$\Delta\omega(t) = \frac{\omega_{syn}\left\{re\left(\Delta S_{Bus1}\right)\right\}\Delta t}{2H\omega_{p.u.}(t)} \qquad (3.11)$$

By applying the equation at the end of every raw power flow calculation, the simulator will be able to modify the frequency of the system and reflect the required dynamics.

In order for the generators to change their real power output to halt frequency deviations due to real power imbalance, each is modeled with basic droop control. Droop control uses frequency as the input to increase or decrease generator output accordingly. With frequency above nominal, the droop controller will reduce the generators output linearly with the magnitude of the frequency deviation as shown in Figure 3.1.

The slope of the line in Figure 1 dictates the magnitude to which the generators output will be changed with respect to frequency and is denoted the droop regulation constant, R. The values of R used in the generators models for the simulations can be found in Appendices A, B and C for the 5, 14 and 30 bus models, respectively.

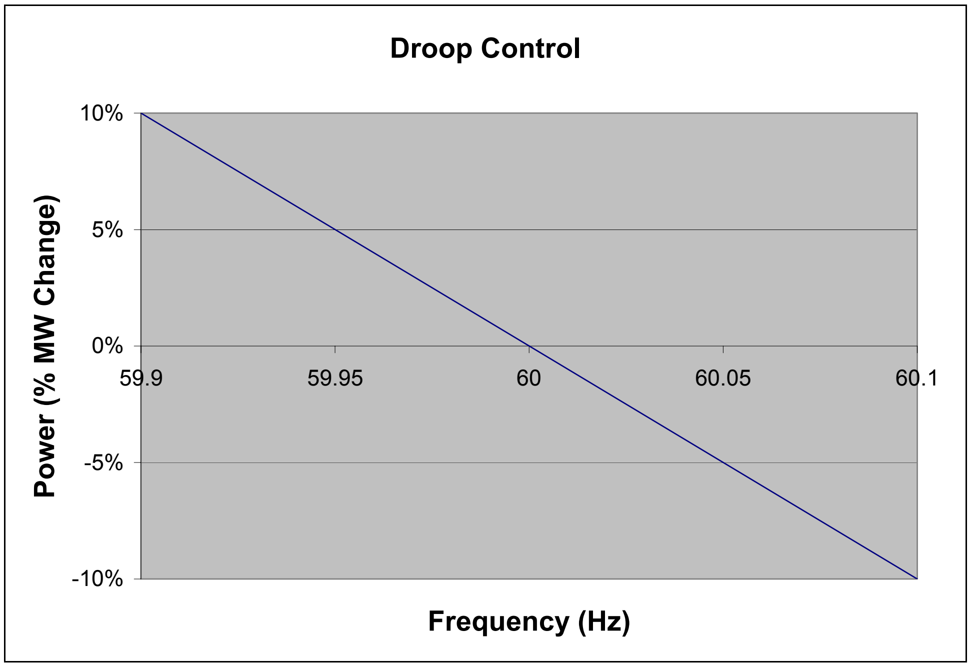


Figure 3.1 - Generator Droop Control Characteristics

Chapter 4. Implementation

The previous chapter introduced the Jacobi algorithm and the modifications implemented to the algorithm with the intent of meeting the design requirements. The formulation of the algorithm lends itself naturally to parallel processing as each of the equations are independent of each other during each iteration.

While it is true that other algorithms have faster rates of convergence, only the Jacobi has such potential utilizing recent advances in parallel processing technology. If the parallelisation can be performed to a great extent, the increased number of iterations may be outweighed by decreased time per iteration.

This chapter introduces and discusses the hardware and software implementations of the parallel Jacobi simulator. The first section includes a discussion of the hardware originally chosen for the implementation, as well as a reasonable substitute eventually chosen as the desired hardware was unavailable. The second section explains the method of software implementation by introducing the three stage approach used to ensure accuracy.

4.1 Hardware Implementation

In the industry of high performance computing, parallel processing has ubiquitously taken over as the dominant method for solving large complex problems [25]. While the specific implementations differ based on the nature of the problem, the basic unit of computation used in such massively parallel machines is the multi-core microprocessor.

Moore's Law (or more accurately, the popular culture version of it) has continued to be proven correct as the 'floating point operations per second' (flops) of microprocessors increases. This has, however, not been due to increasing clock frequencies as it was in the past. It has been primarily due to the emergence of another type of processor, the Graphics Processing Unit (GPU) [26].

4.1.1 Graphics Processing Units

The general trend for GPU design has been increasing the simultaneous number of calculations the processor can perform. This functionality stems directly from the mathematical requirements of graphics processing, which are embarrassingly parallel problems [27]. A significant portion of the calculations required to display an image consist of large vector and matrix manipulations, often requiring the same calculation performed on a large number of values. These calculations can be performed in a serial fashion, but due to their independence from each other, can be performed in parallel.

Current versions of graphics processors have up to 240 processing cores, allowing for 240 simultaneous independent calculations. In addition, vendors are developing platforms which contain multiple chips. For example, the NVIDIA Tesla Series will release the S1070 in Fall of 2008. The unit occupies 1U in a standard computer rack mount enclosure and contains four GPU's with a total of 960 cores, capable of up to 4 teraflops. Multiple units can be stacked together and chained to provide large numbers of simultaneous calculations [28].

From a computing perspective, the ability to perform a large number of simultaneous calculations requires that the processing core be able to obtain the data it requires with low latency, otherwise clock cycles are wasted as the processor sits idle waiting for data. For this reason GPUs have a different architecture than typical general purpose processors. The main design changes consist of locating greater amounts of high speed memory closer to the cores, reducing the memory latency, and increasing the size of the pipelines between the off-chip memory, on-chip memory and processing cores [29].

4.1.2 Equivalent Processing Architectures

Unfortunately, at the time of implementation for the simulator for this thesis, no GPU based machines were available. Therefore, other architectures were investigated in order to find a substitute that possesses similar computing characteristics. The goal was to implement the simulator on an architecture that reflects many of the same characteristics with the intent that the hypothesis of decreased calculation time could be achieved. In order to find a viable substitute, the characteristics of the GPU based architecture were reviewed and the most important aspects selected.

In parallel computing, the primary objective is to speed up the computation time of a process by finding available parallelism, and using multiple processing cores to perform parts of the calculation simultaneously. The theoretical maximum speed up is linear with the number of processors, that is, in a infinitely parallelisable program, doubling the number of processors would half the computation time. While theoretically accurate, there are a number of real world reasons this does not occur.

The primary reason for this is communication time between processors. When comparing the time of moving data from high speed memory located on the processor itself with the time of moving information from local memory or across even the highest speed networks, the difference between the three can be orders of magnitude. For this reason, a large part of the optimization involved in parallel programming is finding the optimal balance between available parallelism and communication delays.

The GPU implementation desired consists of a large number of processors on the same chip, with local memory available for moving information between the processing cores. Similar systems were developed before the emergence of the GPU and are known as Shared Memory Processors (SMP). While not on the same chip, SMP systems use a number of processors located in the same machine and share relatively high speed local memory. In this hardware implementation the largest part of the communication delay, the use of an external network, is removed.

The Shared Hierarchical Academic Research Computing Network (SharcNet) is a consortium of academic institutions that share high performance computing platforms. On this network is a shared memory system with 128 cores and 256GB of memory. Given the same number of processors as currently available GPUs, with little communication overhead compared to cluster systems using network connections, it was deemed that this machine was the closest in functionality that was available for the purpose of system implementation, it was therefore originally chosen as the implementation platform for the simulator.

A portion of the way through the software development for the parallel implementation, another available machine was found with greater similarities to the GPU implementation desired.

It consisted of a dual processor quad core Intel Xeon machine. In total, the simulation would have access to eight processing cores. With four cores having high speed access to shared, on-chip memory, the implementation should perform better than on the SharcNet SMP machine. At this point, software implementation was directed to continue on the Xeon machine. Fortunately, little code modification was required as the utilized parallel programming language, OpenMP, is a widely recognized standard for parallel programming.

It is important to note that even the SMP and Xeon implementation still fall victim to significant processor-memory latency issues. The difference between using on-chip and off-chip memory can be two orders of magnitude in typical processors [30]. Therefore there is a possibility that the results of the Xeon simulations would still be significantly slower than those of a GPU based implementation.

4.2 General Software Implementation

Once the hardware platform was decided, the process of implementing the software to simulate the algorithm needed to be performed. Due to a lack of familiarity with parallel programming and the possibility of computational error, it was decided that a three staged process for development with multiple system models would be used to ensure accuracy and validate the results.

The first stage was the use of third party software to produce verifiable power flow results. The second was a serial processing version of the simulator, and the third a parallel processing implementation. The three electrical system models used were a 5 [31], 14 [31] and 30 [32] bus system. Diagrams of the systems can be found in Appendices A, B and C respectively. In addition, once the parallel system was verified, two larger systems were used to observe the operation of the parallel simulator under large system scenarios. The results of the three implementations and discussion are found in Chapter 5 - Results and Discussion.

4.2.1 Third Party Power Flow Verification

The third party software chosen was PowerWorld Corporation's PowerWorld Simulator Version 8.0, Glover and Sarma Edition, a highly functional academic and industrial grade power flow calculator. The three test systems were implemented and simulated. The results were not dynamic simulations, but instead, sets of individual power flow solutions, i.e. snapshots of the systems at certain points in time. Under steady-state conditions, the power flow solutions should coincide with the results produced by both the serial and parallel implementations of the simulator.

4.2.2 Serial Equivalent Implementation

The second implementation of the simulator was developed in Microsoft Visual Basic 2005 on a standard PC using a single processor. Using the same algorithm and processing each of the Jacobi equations in a serial fashion instead of in parallel should theoretically produce identical solutions.

Since the only difference between the two implementations is the sequence of calculations, it was expected that the results will not only be numerically equal, but the number of iterations should be identical as well. Having the two implementations agree should provide a high degree of confidence that the parallel simulation is performing as expected.

4.2.3 Parallel Implementation

The final implementation was performed in C++ with OpenMP multiprocessor extensions in order to take advantage of the multiple cores on the Xeon machine. The parallel implementation is considered the final version used to run simulations. In addition, timing analyses were performed to validate the potential for speedup using the parallel Jacobi and dynamic simulation concepts.

As mentioned in the first section, the hardware on which the simulator was implemented was not the desired hardware but had similar characteristics. As parallel hardware has a number of different forms, often the method of software implementation changes with the hardware. Fortunately, the method of parallel software implementation for both the GPU and Xeon machines are quite similar. The largest difference between the two is the level of parallelism involved. The GPU implementation has a much higher level and the internal hardware architecture is optimized for such calculations.

In the programming language being used (C++ with OpenMP), the user implicitly states the processes that can be performed in parallel, specifies the number of processors to be used, and the program complies. Since each processor only has 4 processing cores, the amount of work dedicated to a single core will be approximately 25% of the entire processor workload.

With an NVIDIA GPU implementation using C++ with CUDA parallel programming extensions, a similar process is performed, although the processor itself has in excess of 128 processing cores per processor, instead of the 4 available on the Xeon. As was shown in Chapter 3 - Theory, since the Jacobi is an embarrassingly parallel algorithm its efficiency stems from the ability to perform each equation in parallel. For larger systems of thousands of buses, the GPU implementation could perform 128 of the calculations simultaneously instead of the 4 on the Xeon.

With the ability to process over 128 calculations simultaneously using on-chip memory, it is expected that the GPU implementation will show significantly less delays due to off-chip memory access latency. For this reason, it is once again stated that there is a possibility that the results of the Xeon simulations would still be significantly slower than those of a GPU based implementation.

Chapter 5.
Results and Discussion

The chapter begins with the process used to validate the simulator to ensure accuracy. The development and validation process used a three stage approach, with third party power flow software being used to validate the serial implementation and the serial implementation being used to validate the parallel implementation. An analysis is provided on the accuracy of the implemented simulators.

The second section provides and discusses a number of simulations performed using the parallel implementation. Two large scale systems are simulated to gauge the performance of the simulator on larger systems. Results are presented and a discussion is provided to interpret the results.

The third section discusses the outcome of the simulations and attempts to analyze any discrepancies or divergences from the original hypotheses. As well, the section compares the resultant simulator to the original design scope in an effort to gauge the successfulness of the implementation. Finally, the discussion turns to areas for improvement that could be pursued in order to attain the desired performance.

5.1 Simulator Validation

To ensure accuracy of the final simulator, a three stage approach was taken in implementation. The first stage consisted of a steady-state simulation in a verifiable industry-used software package. The second was simulation in the serial implementation, third followed by simulation in the parallel implementation. The three models used for the validation procedure were 5, 14 and 30 bus systems. The details of each system can be found in Appendices A, B and C respectively.

5.1.1 Establishing the Baseline Values

The first step of validation was to input the systems into the PowerWorld simulator to establish the baseline conditions. For each of the three systems, the PowerWorld simulator was used to find the steady-state equilibrium conditions. Since PowerWorld uses the standard power flow algorithm, the input values were the generation levels and reference bus voltage. PowerWorld then found the unknown voltages and slack bus powers required to meet system equilibrium. As the amount of information regarding the details of the systems is lengthy, it has been omitted here with full details found in Appendix D.

The resulting equilibrium PowerWorld generation values were used as inputs to the serial implementation of the simulator. Although PowerWorld and the simulator perform different calculations to find different variables, the systems are identical and should therefore produce the same results.

The serial simulator was provided with the reference bus voltage and all the generation and load values. The resulting solution was extremely close to that of PowerWorld, with the largest error in the initial condition simulations being 0.01% of the PowerWorld results.

5.1.2 Validating Steady-state Values

With the initial condition comparison complete, the serial simulator was subjected to a disturbance to vary the generation and voltage levels from the initial conditions. Once the simulator completed its reaction to the disturbances and came to the new steady-state equilibrium, it was subjected to a second disturbance and once again came to equilibrium.

The disturbances were an increase of real power generation at bus 1 of 1.0pu at time t=5 seconds, followed by a reduction of generation at bus 1 of 0.5pu at time t=20 seconds. The result of the serial 5 bus simulation is shown in Figure 5.1.

The voltages and generation levels at each steady-state equilibrium point were used as inputs to PowerWorld to compare the accuracy of the solutions after a disturbance. Once again it was found that the simulator voltages were within 0.01% of the PowerWorld equivalent values.

For the real power comparison, the results were the same as voltage, in that both systems came to the same results with less than 0.01% deviation. The major deviation was found to be in the reactive power comparison.

As the simulator no longer has a slack generator to relieve any mismatch in real or reactive power mismatch, the reference bus is left with residual mismatch at the end of the raw power flow algorithm. The real portion is used to calculate the system frequency change, which moves the generation control system (market or AGC, in this case droop control) to modify real power output to compensate. Note that there are no limits on the reactive power output at the reference bus.

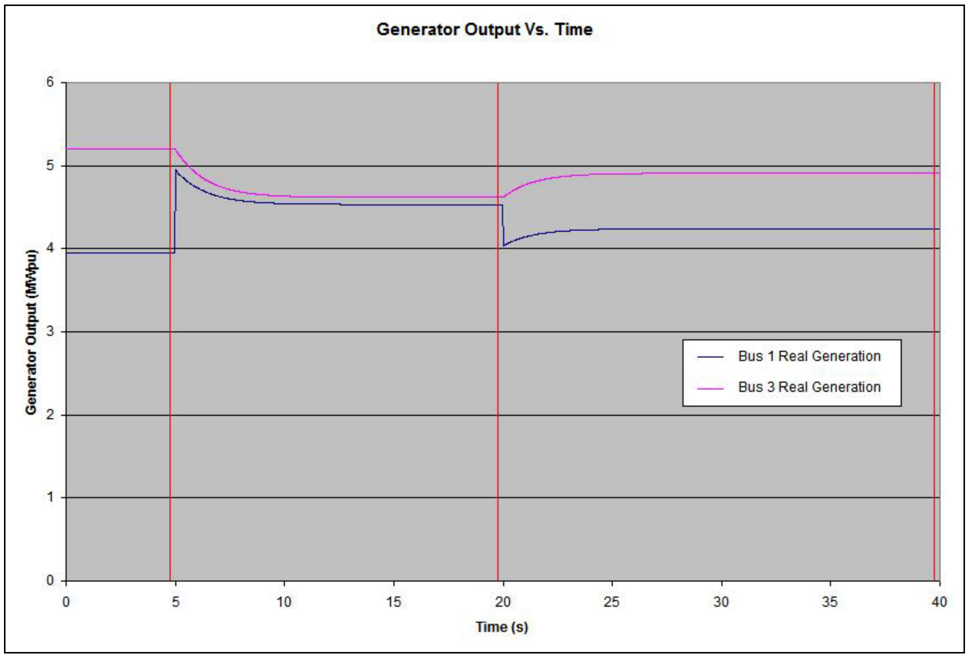


Figure 5.1 - System Disturbances and Steady-state Points Used for Comparison

5.1.3 Validating System Dynamics

Satisfied that the steady-state values of the simulation to this point were accurate, the procedure moved to validate the dynamic portion of the simulation. Long term dynamic simulations work under the base assumption that all the generators move in synchronism with a single system frequency, referred to as the ‘tied-rotor’ assumption.

Under this assumption, calculation of system frequency is performed using the swing equation which describes the acceleration and deceleration of generators under mechanical input and electrical output imbalances. It is based on Newton’s Second Law with modifications for the purposes of using power system inputs such as system frequency, normalized generator inertia constants, and per unit powers.

Three scenarios were simulated on each system to validate the simulator dynamics using the modified swing equation. A sample scenario is shown in Figure 5.2. In all scenarios the systems were allowed to come to equilibrium after an imbalance in initial conditions of the generators. The initial conditions are described in the generator files (Gx.txt) of Appendices A, B and C. The following sequence of steps was followed:

Scenario 1 - Impulse Test

- At time t=5 seconds, bus 1 real power generation increases by 1.0pu.
- The system is allowed to settle for 15 seconds until time t=20 seconds.
- At time t=20 seconds, bus 2 real power generation decreases by 0.5pu.
- The system is allowed to settle for 20 seconds until time t=40 seconds.
- Simulation completes at time t=40 seconds.

Scenario 2 - Slow Ramp Test

- At time t=5 seconds, bus 1 real power generation increases by 0.0001pu per time step of 0.01s, i.e. bus 1 real power generation increases at a rate of 0.01pu/second.
- The ramp continues until time t=25 seconds, at which the generation increase ceases and the system is allowed to settle until time t=40 seconds.
- Simulation completes at time t=40 seconds.

Scenario 3 - Ramp Test

- At time t=5 seconds, bus 1 real power generation increases by 0.0005pu per time step of 0.01s, i.e. bus 1 real power generation increases at a rate of 0.05pu/second.
- The ramp continues until time t=10 seconds, at which the generation increase ceases and the system is allowed to settle until time t=10 seconds.
- At time t=10 seconds, bus 1 real power generation decreases by 0.001pu per time step of 0.01s, i.e. bus 1 real power generation increases at a rate of 0.1pu/second.
- The ramp continues until time t=30 seconds, at which the generation decrease ceases and the system is allowed to settle until time t=40 seconds.
- Simulation completes at time t=40 seconds.

Each of the three scenarios was chosen to investigate a different aspect of the simulator's ability. The impulse scenario was used to observe system response to step changes in generation. The slow ramp scenario was used to observe the system response to a long term ramp with the intent to have the droop control eventually equal the generator ramping. The ramp scenario was used to observe the system response to faster ramps of generation in different directions.

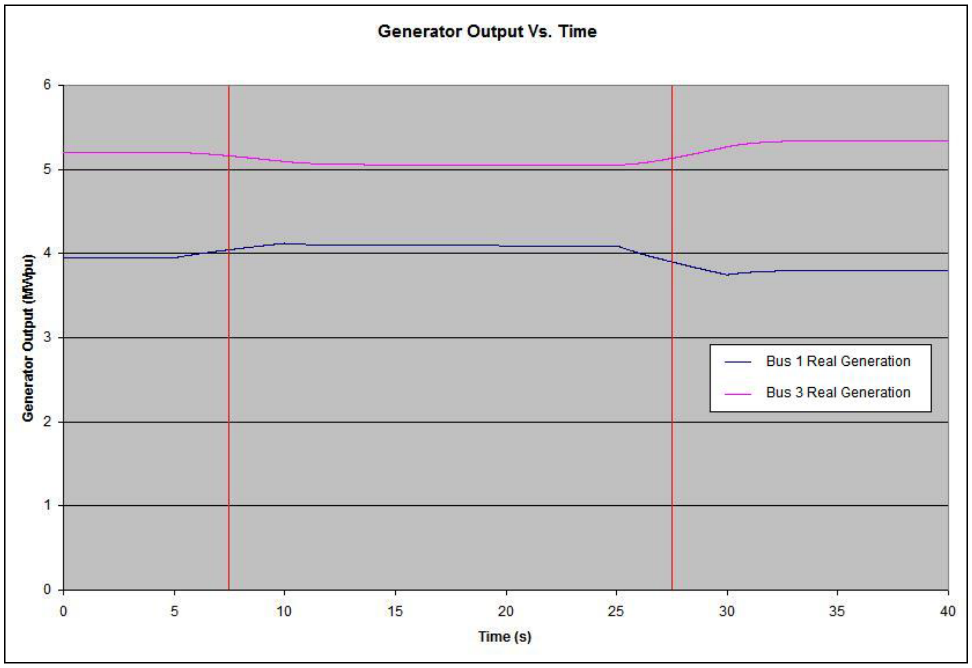


Figure 5.2 - Ramping Dynamic Sampling Points

At the points indicated on Figure 5.2 (in addition to points on the other scenarios), the real power mismatch values were taken from the simulation results and the frequency deviations were calculated manually to ensure that the simulation results were correct. It was found that the frequency deviations for the times given were indeed identical and therefore proved the integrity of the calculation. The full results of the calculations can be found in Appendix E.

Having completed validation of both the steady-state and dynamic portions of the serial simulator, the implementation is considered to operate properly and can be used to validate the operation of the parallel implementation.

5.1.4 Validating the Parallel Implementation

Once the serial simulator was validated, it was used as the benchmark for the parallel implementation. Using the three system models of 5, 14 and 30 buses and the three scenarios for each, the output of the parallel implementation was compared to the serial version in order to compare accuracy.

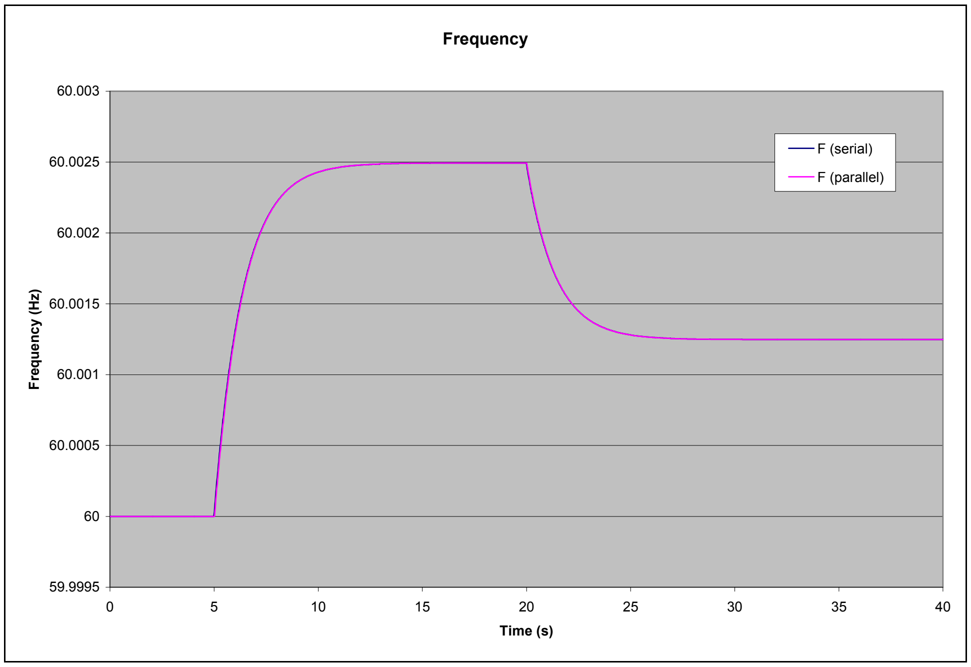


Figure 5.3 - Serial / Parallel Frequency Comparison

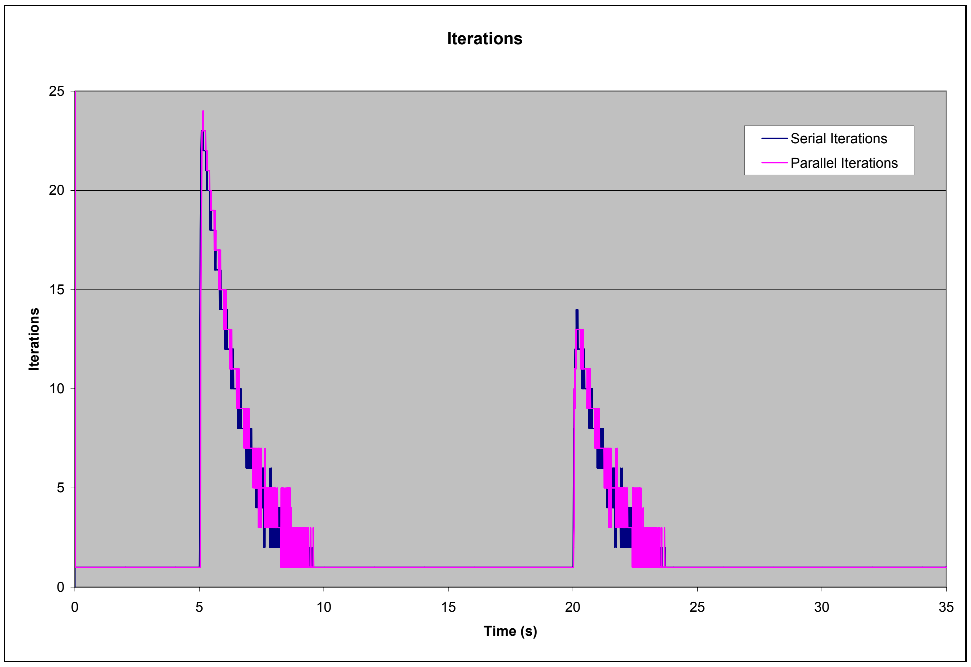


Figure 5.4 - Serial / Parallel Iteration Comparison

The three scenarios were simulated using the Xeon parallel processing implementation. Figure 5.3 and Figure 5.4 show sample results obtained for the 30 bus impulse scenario.

Note that on Figure 5.4 the scale has been altered in order to increase the resolution. The Jacobi algorithm requires a large number of iterations in the first time step in order to converge from a flat start (voltage of magnitude 1 and angle 0) to the first solution. In the small systems described, this was on the order of one to five hundred iterations. This is an initial condition issue and once the system is solved initially, the values from the previous time step are used as the starting point for the next time step. With little dynamics occurring in the system, it often converges in a single iteration. With large changes in the system, such as in the impulse scenario, the number of iterations is seen to increase from 20 to 50 iterations.

Visually, all of the simulations appear to be almost identical. From a numerical perspective, virtually all of the parallel calculations are within 0.1% of their serial equivalents. The balance of the values have rather large percentage differences, although they occur during or just after the generation changes applied during the scenarios.

Further observation indicates that these significant deviations may be a result of a time step offset. As an example, in the serial implementation the step increase in generation may occur at time t=5 seconds, that is, at the beginning of the t=5 second time step. In the parallel implementation the step increase may occur at the end of the t=5 second time step, that is, at the beginning of the t=5.01 time step.

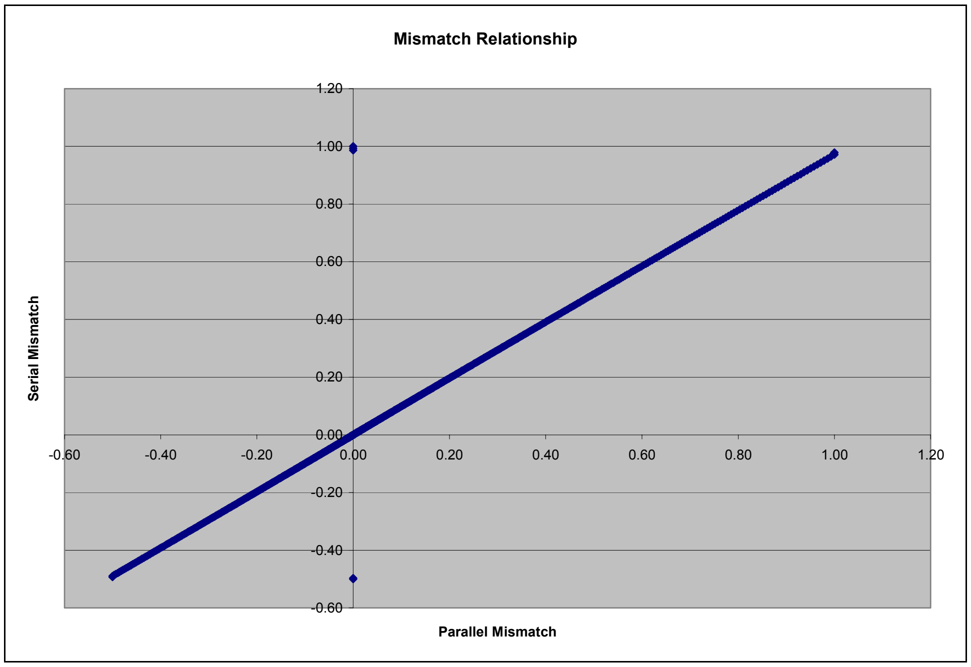


Figure 5.5 - Serial Parallel Mismatch Relationship

From a percentage perspective, the deviations may be numerically significant but there is a much more important relationship between the values, correlation. By plotting the same data points from both simulations against each other on the same graph should show a linear relationship between the two. Any outliers should be readily visible. An example of one such graph is given in Figure 5.5.

It can be seen that the significant majority of the data sets have a strict relationship. The three points that lie on the y-axis are examples of the time step offset described. They occur at the time at which the impulses are applied.

As the simulator is used for long term dynamic studies in real-time, such minute discrepancies are of little significant interest and do not materially affect the results. Constant offsets of 0.01 seconds are trivial in real-time simulation scenarios that may last tens of minutes and where the stimuli and their associated responses are measured in second to minutes. A number of additional serial / parallel result relationships were studied and came to similar results.

In terms of meeting the functional objectives of power system simulation, the simulator is considered validated. That is, the results of all the simulations have come to nearly identical results and exhibit nearly identical behaviour during the simulation of long term dynamics in a power system.

5.2 Parallel Simulator Timing Analysis

With a validated long term dynamic power system simulator, a performance analysis can be performed. The primary goal of the thesis is to be able to implement a real-time power system simulator, with time steps of less than 0.033 seconds. This should be true for a number of system sizes, such that the simulator is scalable for use in real world power system studies. For the actual simulations, time steps of 0.01 seconds were used in order to ensure that the simulator could achieve or surpass the required goal.

This section performs a number of simulations using the parallel Xeon implementation. Different systems are simulated using anywhere from 1 to 8 processing cores. Comparisons are made with respect to system size and number of cores used.

The 5, 14 and 30 bus test systems were considered too small to be of any computational significance to the parallel processing implementation. Additional systems were developed that contained 100, 500, 1000 and 2000 buses. As the version of PowerWorld being used to generate the Ybus matrices was limited to 30 buses, a random power system generator was developed to create larger systems. A description of the program is included in the following section, with the timing analysis following.

5.2.1 Synthesis of Large Systems

The models used for the 5, 14 and 30 bus systems were originally obtained as IEEE Common Format files and imported into the PowerWorld software package. The version of PowerWorld is a limited version meant for academic purposes. While the software remains full functional, it is limited to the simulation of systems of 40 buses or less. The full version is capable of over 60,000.

Since PowerWorld was used to provide the Ybus matrix, system sizes greater than 40 buses were not able to be imported to PowerWorld or their associated Ybus matrices exported for use within the simulator. In order to test the performance characteristics of the parallel simulator with greater system sizes, a program was developed in order to produce random electrical systems.

In order to produce a random system, the following information is required:

1) The number of buses in the system.

2) The network layout in the form of the Ybus matrix.

3) The location and power output of generators.

4) The normalized rotational inertia constant (H) of each generator.

5) The droop characteristics (R) of each generator.

6) The location and demand of all the loads.

In the random system generator developed the only user defined value is the number of buses, the balance of the information is calculated using randomly chosen values with limits. For the systems used in the testing of the simulator, the following system design parameters were implemented.

For the layout of the network, it was decided that approximately 2% of the buses would be connected to another bus through a transmission line. For every possible connection a random number between 1 and 0 was chosen. If the number was 0.02 or less, a randomly chosen admittance was added to the Ybus matrix. The value of the admittance was a randomly chosen value within typical limits of actual transmission lines as shown in Table 5.1. This procedure was repeated until the upper triangular portion of the Ybus matrix was filled in.

Once the random procedure was complete, the program went through the system to ensure that no electrical islands were produced during the random process. If an island was found, a random bus in the island was connected through a transmission line to a random bus in the island that contained the reference bus. This ensured that all the buses were part of a single electrical network.

Once the upper triangular portion of the Ybus matrix was complete, the diagonal values were calculated using the upper triangular portion and then the upper triangular portion was symmetrically copied to the lower triangular portion.

Generators were randomly scattered throughout the system with a generator to bus ratio of 1 to 5, i.e. one generator for every 5 buses. The power output and H constants of the generators were chosen randomly within limits, as shown in Table 5.2. The droop control constant, R, was chosen such that all generators could participate equally in responding to frequency deviations.

Loads were assigned to every bus, with randomly assigned values between the limits as shown in Table 5.2.

Table 5.1 - Transmission Line Characteristics

	Ymax(re)	Ymin(re)	Ymax(im)	Ymin(im)
Transmission Line	-25	-1	25	1

Table 5.2 - Transmission Line Characteristics

Component	Pmax	Pmin	Qmax	Qmin	Hmax	Hmin
Generator	50	10	50	0	3	1
Load	100	1	50	1	-	-

5.2.2 Large System Timing Analysis

Each of the four large systems was simulated on the parallel Xeon implementation using 1 to 8 processing cores. Each simulation consisted of 40 seconds of simulation, with time steps of 0.01s. Therefore, the results are the time required to perform 4000 sequential power flows for the system size given, as well as the frequency calculation in between each power flow.

The result of each set of simulations is presented in Figures 5.6 to 5.9. Each figure contains two sets of data, User and Real. The Real line refers to the actual time taken for the simulation to complete, taken from a human's perspective sitting at the simulator. The User line refers to the total time taken by all processors to complete the simulation, that is, the sum of each processors computation time. The performance of the simulator is solely gauged on the Real time results.

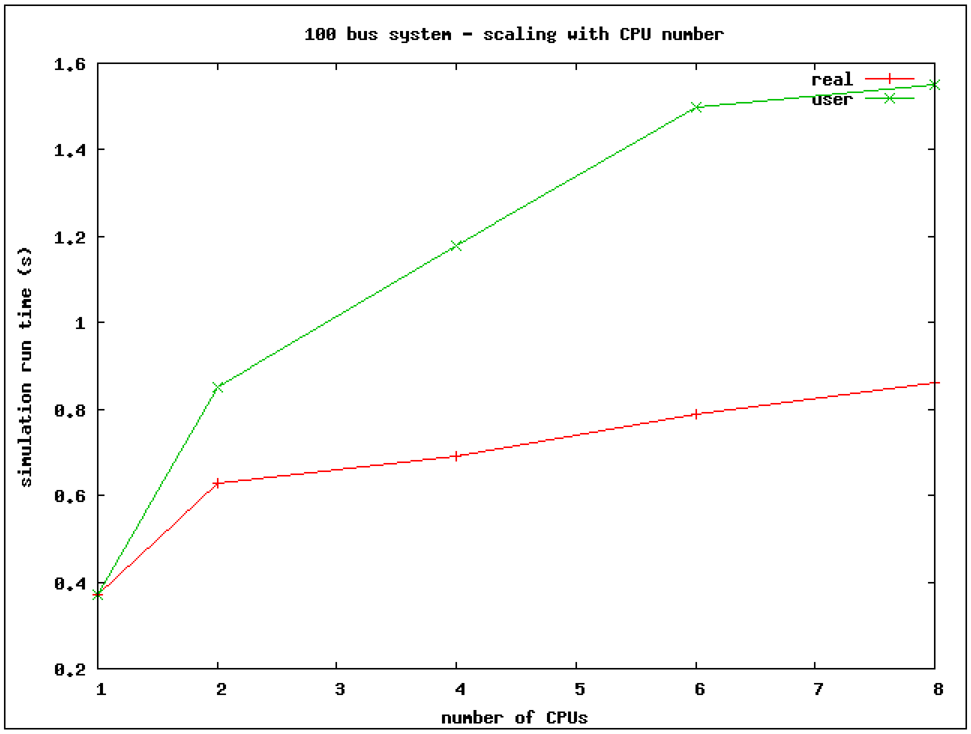


Figure 5.6 - 100 Bus Parallel Timing Comparison

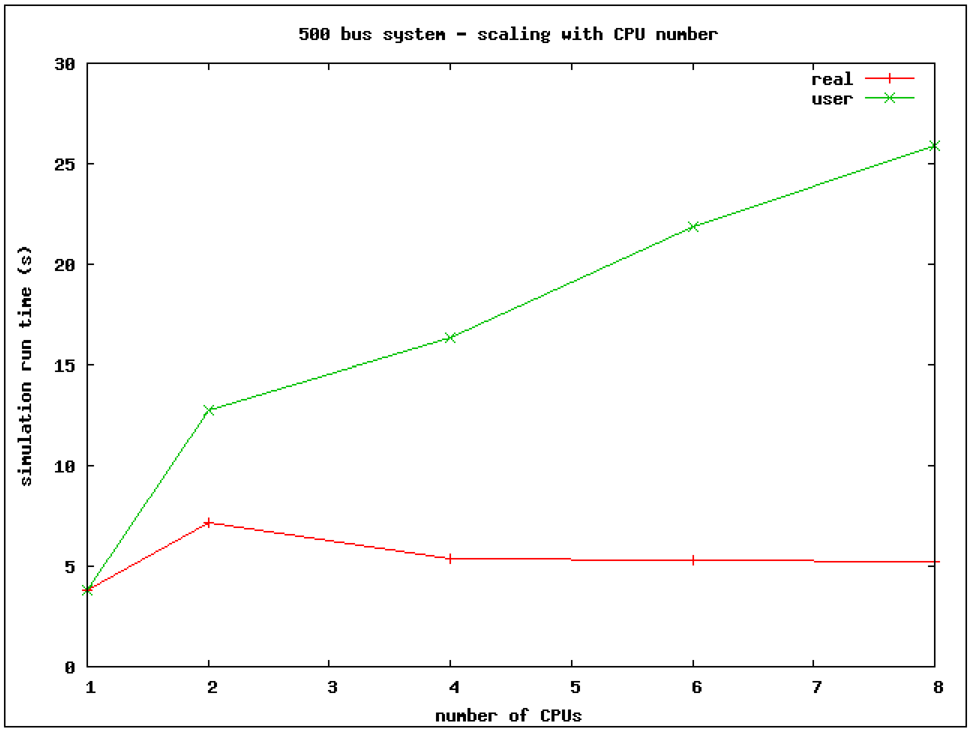


Figure 5.7 - 500 Bus Parallel Timing Comparison

From observation, it can be see that for the 100 and 500 bus systems the best performance is obtained from the use of a single processing core. This is entirely attributed to the large communication delays between the processing cores. This is largely seen in the change from a single to dual processing cores. Once the largely constant communication delay is introduced, the use of more cores once again decreases the simulation time as more operations are being done simultaneously. The fact remains that the simulator cannot achieve the peak performance obtained from a single core.

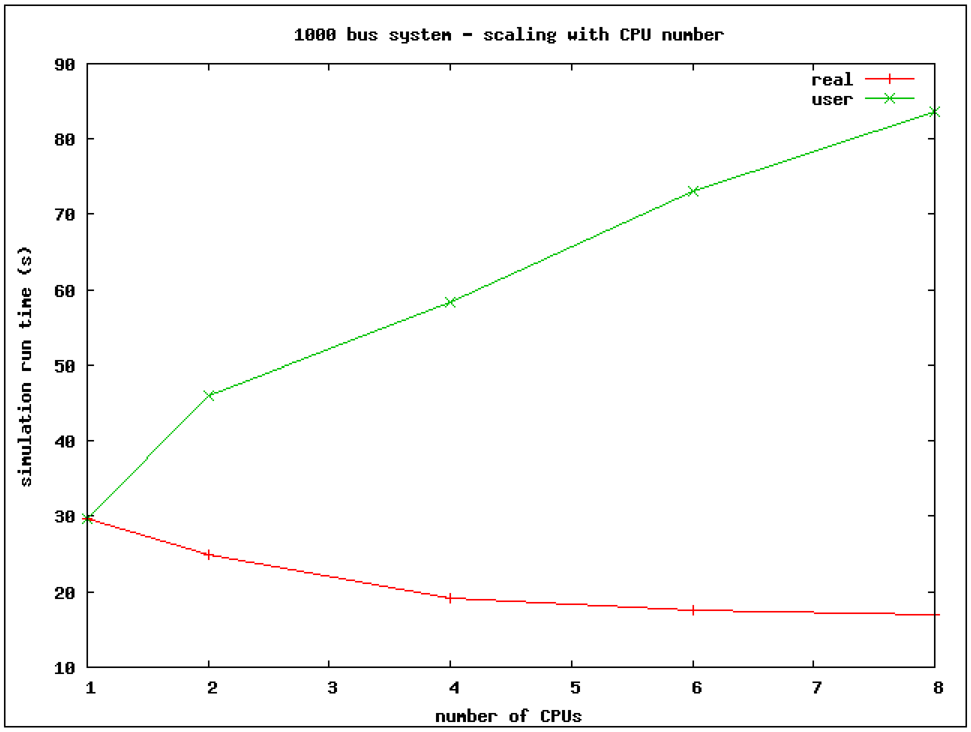


Figure 5.8 - 1000 Bus Parallel Timing Comparison

When moving to larger systems, there are significantly more calculations to be performed within a single iteration of the Jacobi power flow algorithm. For this reason, the use of multiple cores can outweigh the communication delays between cores. This is evident in the simulation results of the 1000 and 2000 bus systems of Figures 5.8 and 5.9.

As cores are added the processing time decreases in larger systems, but there is a point at which adding cores produces diminishing returns. This eventually results in no additional timing gains with more cores.

From the results of the timing analysis, it appears that larger systems can take advantage of more cores. This is exhibited in Figure 5.10 that shows the simulation time per bus. For the larger 2000 bus system, the simulation time per bus continues to decrease as the number of cores is increased. Figure 5.11 shows the relationship between the number of buses and the effect of adding cores. Using a single core the computation times increase in what seems to be an exponential manner. As cores are increased, the effect is to flatten out the graph, although more simulations of larger systems would be required in order to fully describe the true effect on the results.

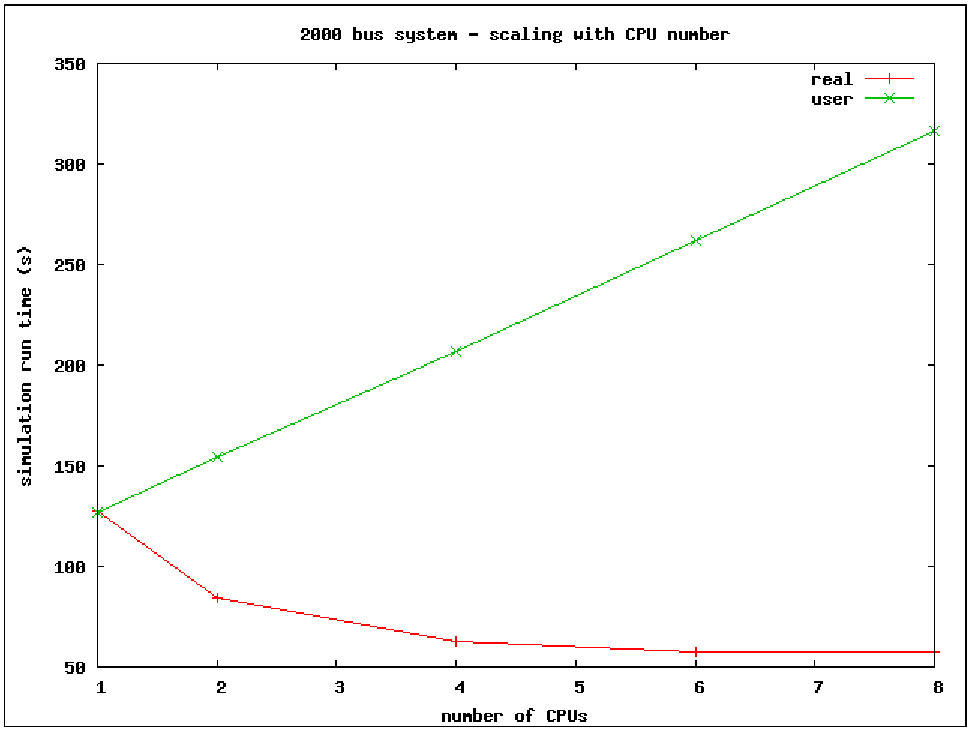


Figure 5.9 - 2000 Bus Parallel Timing Comparison

Regarding the original design metric of achieving real-time performance, it appears that the implementation does not quite meet the objective. In each of the large system simulations, the simulation time was 40 seconds with the impulse scenario under test. In the case of the 100, 500 and 1000 bus systems, the simulator managed to achieve the goal with a wide margin. In the 2000 bus simulation, the 8 cores version achieved a time of approximately 55 seconds, 15 seconds past the real time deadline.

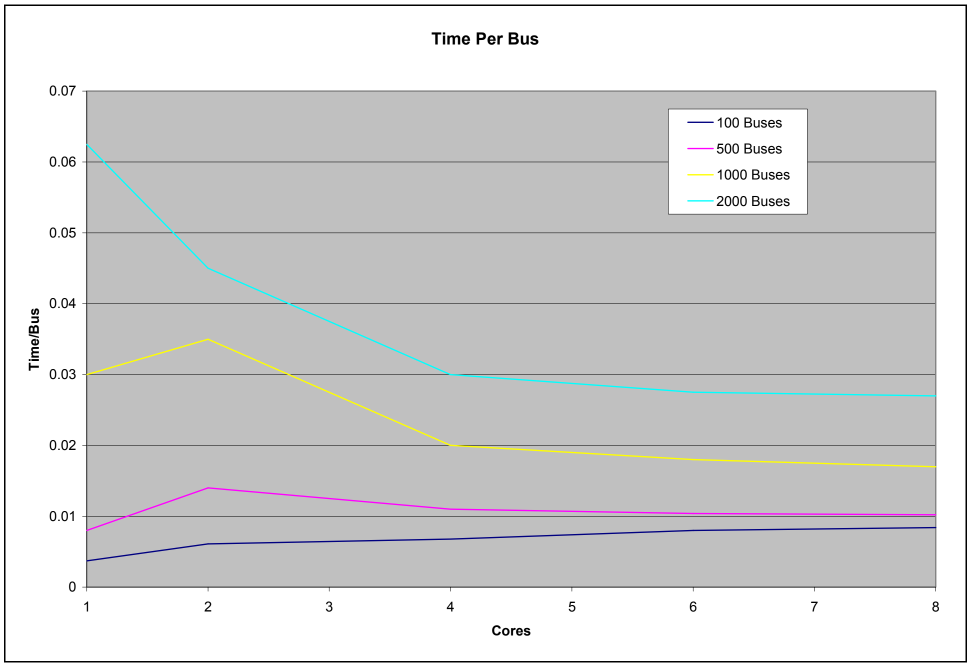


Figure 5.10 - Simulation Time per Bus

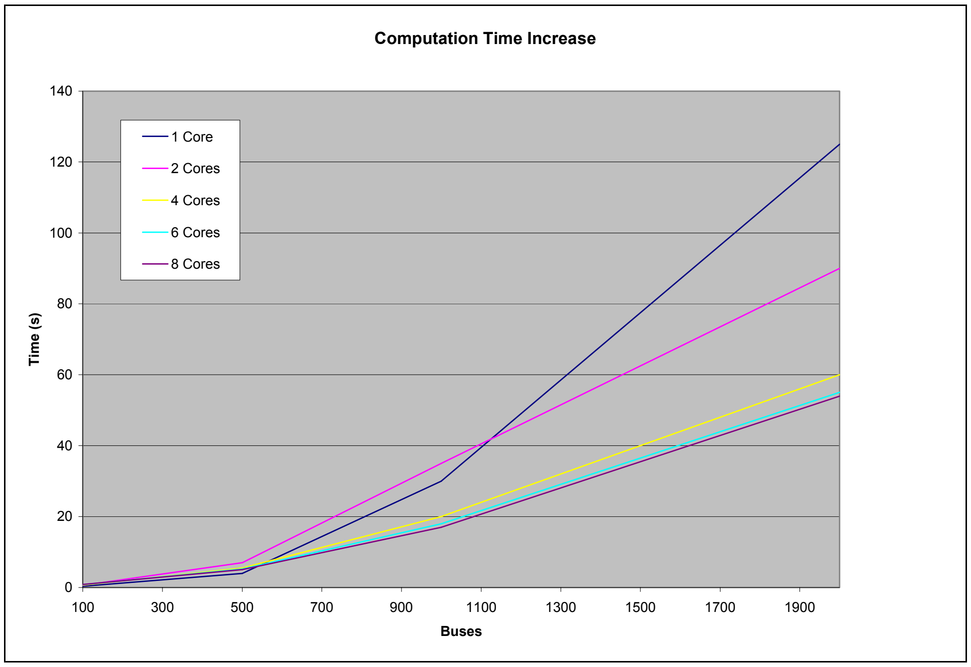


Figure 5.11 - Computation Time Increase

5.3 Discussion and Future Potential

The simulator has been validated, simulations performed and timing analysis completed. This section includes a discussion of observations made during the process, as well as possible explanation for the observations made.

5.3.1 Theoretical Limits of the Jacobi Algorithm

The Jacobi is an inherently embarrassingly parallel algorithm. The bulk of the computations are the calculation of the updated voltages and resulting bus mismatches during each iteration. For example, in a 2000 bus system there are 2000 independent voltage equations, one for each bus, and 2000 independent mismatch equations. In a theoretical implementation which does not include any communication delays, the number of serial computations to be performed could be brought down from 4000 (2000 voltage and 2000 mismatch) to only 2. In this respect the Jacobi algorithm is potentially very powerful if the correct parallel implementation can be found.

In current technology, the most powerful parallel processing technology for embarrassingly parallel algorithms is that of the GPU. The architecture is specifically designed and optimized for such calculations. Unfortunately, the technology was not available for the implementation of the simulator and a substitute technology was used.

From the results of the larger simulations, it was shown that inter-processor communication delays resulted in decreased performance. At the same time, it was generally shown that the Jacobi algorithm has potential for accelerating power flows.

5.3.2 Reduction in Jacobi Iterations

The largest drawback to the Jacobi algorithm lies in the large number of iterations required to solve the problem. In standard power flow applications the calculations are normally performed from a flat start, that is, a voltage of magnitude 1 and angle 0. This is shown in the initial iterations required during the first time step. In the systems simulated, this ranged from 500 for the smaller systems, to thousands for the larger systems.

As this thesis focuses on a simulator and not a power flow calculator, there is one major benefit that acts to increase the performance of the Jacobi algorithm. At the end of each time step, the next cycle of the Jacobi algorithm has a reasonably good estimate of the voltages in the next iteration. Long term dynamics are typically slow moving and therefore the system state does not typically make large changes. This has the effect of significantly reducing the number of iterations required in order to obtain convergence. If iteration count is the largest drawback of the algorithm, then it has been effectively negated when Jacobi is used to continuously simulate power systems.

This effect was described briefly in the parallel validation section. In the 30 bus system the initial iteration count was over 500. During portions of the simulation when little dynamics were occurring, the system often converged in only a single iteration. During severe changes such as an impulse increase in generation, the iteration count never exceeded 25.

It is expected that this feature will be of great benefit when simulating large systems where the differences between time steps are relatively small. If the number of simultaneous equations calculated can be increased and the iteration count reduced, the Jacobi algorithm can prove to be a powerful tool in power system simulation.

5.3.3 Additional Parallelism and Optimizations

In the timing analysis it was observed that as the number of processors increased, the additional benefits of parallelism showed diminishing return. This was primarily due to the nature of the hardware being utilized to implement the algorithm. As stated, the available parallelism in Jacobi is quite high from a theoretical standpoint, and that the GPU implementation could potentially harness these efficiencies.

In addition to the parallel nature of the Jacobi algorithm, there is significant parallelism available within the complex math required to perform the calculations. As shown in the literature survey, complex operations can be broken down into simpler real number multiplications and additions, many of which can be performed simultaneously. The nature of the GPU allows for this level of parallelism to be performed highly efficiently.

Another available level of parallelism available is that of the voltage and mismatch equations themselves. The summation terms in both the voltage and mismatch equations consist of a number of independent complex multiplications that can be performed simultaneously.

$$V_k(i+1) = \frac{1}{Y_{kk}}\left[\frac{S_k^*}{V_k^*(i)} - \sum_{n=1,n\neq k}^{N} Y_{kn}V_n(i)\right] \tag{5.1}$$

$$S_k = P_k + jQ_k = V_k\left[\sum_{n=1}^{N} Y_{kn}V_n\right]^* \tag{5.2}$$

When considering the Jacobi algorithm as a whole, it is observed that the new voltages are calculated, updated, and then the mismatch values calculated. It can be seen that the summation of the mismatch calculation is the same as the summation of the following voltage calculation, save the n=k term. By isolating the mismatch summation and reusing the calculation could effectively cut the computation time in half.

Such fine grained parallel optimizations require the correct hardware in order to be able to produce significant efficiency increases and could potentially be met using current GPU technology.

Chapter 6. Conclusions and Future Work

Market based generation dispatch is becoming the industry norm in advanced electrical jurisdictions. Due to the continuous evolution of markets and their potential impacts on system operation, studies are performed from economic and social perspectives in order to gauge the effects of any changes before implementation into live systems. In addition, it is essential to verify the effect of changes in market design on power systems from a technical perspective.

6.1 Conclusions

The main objective of this thesis is to develop a real-time power system simulator used in the investigation of market designs and AGC schemes. The scope of this thesis is the mathematical algorithms used in the simulator, hardware and software implementation, and validation of the implemented simulator.

Market and AGC studies operate in the long term dynamic time scale, with time constants varying from seconds to minutes. For this reason the simulator is restricted to these timescales, and operates under the 'tied-rotor' assumption. The maximum time-step was defined to be 0.033 seconds in order to allow for video quality refresh rates, although time steps of 0.01 seconds were used for the actual simulations.

A literature survey was performed to gauge the status of long term dynamic simulators, as well as the specific implementation desired for this thesis. It was found that, to the best of the author's knowledge, there have been no Jacobi based parallel processing implementations of a long term dynamic power system simulator.

The simulator is based on a modified version of the Jacobi power flow solution algorithm. The slack generator was removed to provide a generation load imbalance for the dynamic portion of the simulator. By performing a number of power flow calculations in quick succession, the resulting sequence provides the sensation of a fluid simulation.

As the power flow is a static algorithm, dynamics were added to the simulator using a variation of the generator swing equation. The reference bus mismatch is used in the equation to calculate the frequency deviation for every time step.

A number of hardware platforms were considered for use, with the GPU based systems being the optimal system. Unfortunately, none were available at the time of development, and a substitute system with similar qualities was found.

The development and validation was done using a three stage process. The first was the use of a third party commercial software package for verification purposes. The second was the development of a serial version of the simulator, and finally the parallel implementation.

The third party software produced steady-state results against which the serial implementation was compared for numerical accuracy. The dynamics of the serial implementation were compared to hand calculations in order to verify their accuracy. The accuracy of the steady-state simulations was high.

With the serial implementation validated, it was used as a benchmark for the parallel implementation. It was found that the simulations were not identical, but correlated such that the parallel simulator was considered to be an accurate representation of a dynamic power system.

A number of large systems were synthesized using a proprietary program. The large systems were used to test the performance of the simulator to see if it managed to meet the original performance specification. It was found that smaller systems did not take advantage of the parallel processing implementation. For larger systems significant gains were apparent.

From the timing analysis, it was hypothesized that a GPU hardware implementation could achieve significant performance gains over the Xeon multi-core processing architecture used for the parallel implementation.

In conclusion, the simulator achieved the basic functionality goals, but failed to meet the ultimate performance goals. Achieving the stated performance goals is addressed in the following section.

6.2 Future Work

The primary advancement that could be implemented is the use of a GPU based hardware platform. With the highly parallel nature of both the Jacobi algorithm and GPU's, it is believed that the simulator could make significant performance gains.

With the level of parallelism available in such hardware, a number of algorithm optimizations could be implemented as well. Examples include parallelizing the calculation of the summations within the voltages and mismatches and breaking down the complex mathematics into fundamental operations.

Beyond the concepts mentioned within the thesis, there are additional avenues for developing the simulator. During the literature survey, a paper suggested the concept of distributing mismatch in order to gain accuracy for the frequency calculation. While the implementation of the distributed mismatch in the paper was highly complex and 'after-the-fact', there is a possibility of having the Jacobi power flow algorithm converge such that the mismatch is distributed upon convergence. This would be a major modification and would require additional information from the generators in terms of their participation in the mismatch.

Additional functionality could also be found in providing a better modelling of the voltage controls such as PV buses and reactive power limits on the generator models. In addition, the simulator here is for simple power networks and the addition of additional controls such as transformer tap changers and phase shifters should be added in order to be able to better represent and simulate actual power systems.

References

[1] K. Bhattacharya, M. H. J. Bollen, J. E. Daalder, *Operation of Restructured Power Systems*. Norwell, MA: Kluwer Academic Publishers, 2001, pp. 1-2.

[2] P. L. Joskow, "*California's Electricity Crisis*," National Bureau of Economic Research, Cambridge, MA, Working Paper 8442, Aug. 2001.

[3] J. P. Barret, P. Bornard, B. Meyer, *Power System Simulation*. London, UK: Chapman and Hall, 1997, pp. 1.

[4] K. Tomsovic, D. E. Bakken, V. Venkatasubramanian, A. Bose, "Designing the Next Generation of Real-Time Control, Communication, and Computations for Large Power Systems," *Proceedings of the IEEE*, vol. 93, no. 5, pp. 965-979, May 2005.

[5] T. Van Cutsem, Y. Jacquemart, J.-N. Marquet, P. Pruvot, "A Comprehensive Analysis of Mid-Term Voltage Stability," *IEEE Transactions on Power Systems*, vol. 10, no. 3, pp. 1173-1182, Aug. 1995.

[6] W. Long, D. Cotche, D. Ruiu, P. Adam, S. Lee, R. Adapa, "EMTP - A Powerful Tool for Analyzing Power System Transients," *IEEE Computer Applications in Power*, vol. 3, no. 3, pp. 36-41, July 1990.

[7] R. Kuffel, J. Giesbrecht, T. Maguire, R. P. Wierckx, P. G. McLaren, "A Fully Digital Power System Simulator Operating in Real-Time," in *IEEE Canadian Conference on Electrical and Computer Engineering*, vol. 2, pp. 733-736, May 1996.

[8] J. P. Barret, P. Bornard, B. Meyer, *Power System Simulation*. London, UK: Chapman and Hall, 1997, pp. 99.

[9] IEEE Task Force on Power System Stabilizers, “Overview of Power System Stability Concepts,” in *IEEE Power Engineering Society General Meeting*, vol. 3, pp. 1762-1768, July 2003.

[10] V. Kola, A. Bose, P. M. Anderson, “Power Plant Models for Operator Training Simulators,” *IEEE Transactions on Power Systems*, vol. 4, no. 2, pp. 559-565, May 1989.

[11] D. R. Davidson, D. N. Ewart, L. K. Kirchmayer, “Long Term Dynamic Response of Power Systems: An Analysis of Major Disturbances,” *IEEE Transactions on Power Apparatus and Systems*, vol. 94, no. 3, pp. 819-826, May 1975.

[12] J. P. Barret, P. Bornard, B. Meyer, *Power System Simulation*. London, UK: Chapman and Hall, 1997, pp. 84-86.

[13] IEEE PES Power Systems Engineering Committee, “Parallel Processing in Power Systems Computation,” *IEEE Transactions on Power Systems*, vol. 7, no. 2, pp. 629-638, May 1992.

[14] W. Jun Qiang, A. Bose, “Parallel Solution of Large Sparse Matrix Equations and Parallel Power Flow,” *IEEE Transactions on Power Systems*, vol. 10, no. 3, pp. 1343-1349, Aug. 1995.

[15] T. Feng, A. J. Flueck, “A Message-Passing Distributed-Memory Parallel Power Flow Algorithm,” in *IEEE PES Winter Meeting*, vol. 1, pp. 211-216, Jan. 2002.

[16] T. Feng, A. J. Flueck, “A Message-Passing Distributed-Memory Newton-GMRES Parallel Power Flow Algorithm,” in *IEEE PES Summer Meeting*, vol. 3, pp. 1477-1482, July 2002

[17] C. Hong, C. M. Shen, "Implementation of Parallel Algorithms for Transient Stability Analysis on a Message Passing Multicomputer," in *IEEE PES Winter Meeting*, vol. 2, pp. 1410-1415, Jan. 2000

[18] G. Huang, W. Ongsakul, "Managing the Bottlenecks in Parallel Gauss-Seidel Type Algorithms for Power Flow Analysis," in *IEEE Power Industry Computer Application Conference*, pp. 74-81, May 1993.

[19] D. P. Koester, S. Ranka, G. C. Fox, "A Parallel Gauss-Seidel Algorithm for Sparse Power System Matrices," in *IEEE Supercomputing*, pp. 184-193, Nov. 1994.

[20] S. H. Ahmed, "On Jacobi and Jacobi-Like Algorithms for a Parallel Computer," *Mathematics of Computation*, vol. 25, no. 115, pp. 579-590, July 1971.

[21] D. Gimenez, R. van de Geijn, V. Hernandez, A. M. Vidal, "Exploiting the Symmetry on the Jacobi Method on a Mesh of Processors," in *IEEE Euromicro Workshop on Parallel and Distributed Processing*, pp. 377-384, Jan. 1996.

[22] D. P. Chassin, P. R. Armstrong, D. G. Chavarria-Miranda, R. T. Guttromson, "Gauss-Seidel Accelerated: Implementing Flow Solvers on Field Programmable Gate Arrays," in *IEEE PES General Meeting*, June 2006.

[23] R. Ramanathan, H. Ramchandani, S. A. Sackett, "Dynamic Load-Flow Technique for Power System Simulators," in *IEEE PES Summer Meeting*, pp. 25-30, August 1986.

[24] J. P. Barret, P. Bornard, B. Meyer, *Power System Simulation*. London, UK: Chapman and Hall, 1997, pp. 84-86.

[25] Top 500 Supercomputer Sites (2008, July). Top 500 List - June 2008. [Online]. Available at http://www.top500.org/list/2008/06/100

[26] No Author, "*NVIDIA CUDA Programming Guide, Version 0.8*," NVIDIA Corporation, Santa Clara, CA, Dec. 2007.

[27] C Boyd, "Data Parallel Computing," in *ACM Queue*, vol 6, no. 2, pp. 30-39, March/April 2008.

[28] No Author, "*NVIDIA Tesla S1070 Datasheet*," NVIDIA Corporation, Santa Clara, CA, June 2008.

[29] K. Fatahalian, M. Houston, "GPU's - A Closer Look," in *ACM Queue*, vol 6, no. 2, pp. 30-39, March/April 2008.

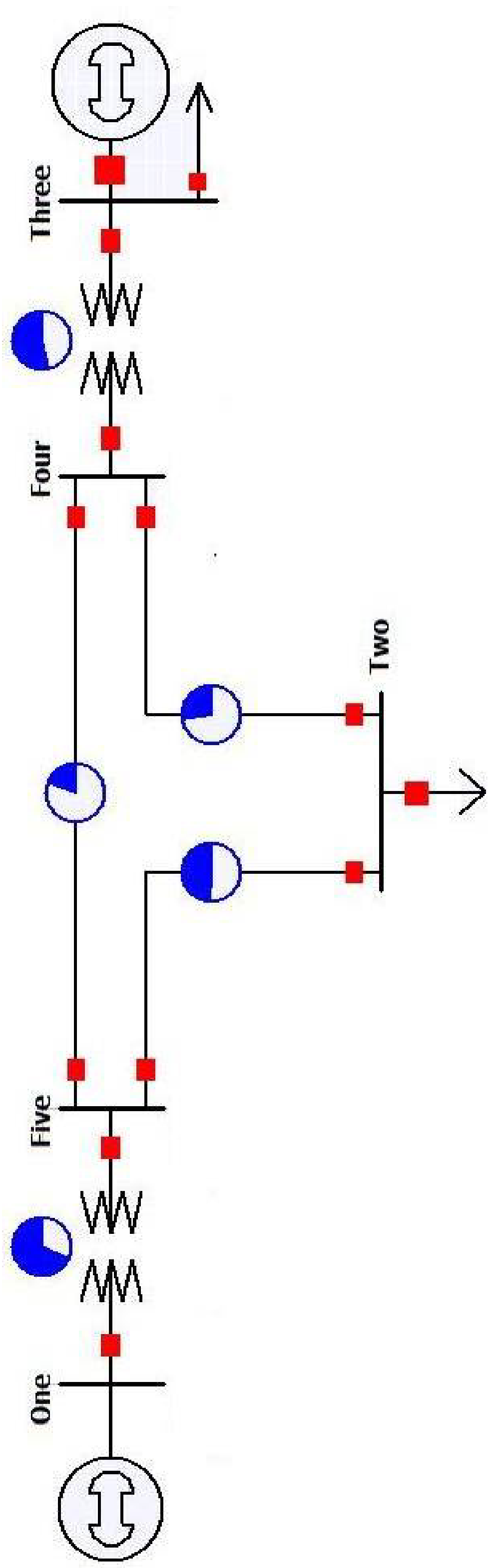
[30] C. C. Liu, I. Ganusov, M. Burtscher, S. Tiwari, "Bridging the Processor-Memory Performance Gap with 3D IC Technology," in *IEEE Design and Test of Computers*, pp. 556- 564, Nov/Dec 2005.

[31] Software, "PowerWorld Simulator Version 8.0, Glover and Sarma Edition," PowerWorld Corporation, Champaign, IL, 2001.

[32] University of Washington Department of Engineering (2008, July). Power Systems Test Case Archive. [Online]. Available at http://www.ee.washington.edu/research/pstca/

Appendix A - 5 Bus Power System Model

5 Bus System Diagram



5 Bus System Simulator Input Files

Bx.txt

Bus	BusID	Y(re)	Y(im)	VRef(re)	VRef(im)
1	B1	3.729	-49.7203	1	0
2	B2	2.6783	-28.459	1	0
3	B3	7.458	-99.4406	1	0
4	B4	11.9219	-147.959	1	0
5	B5	9.0856	-108.578	1	0

Tx.txt

Bus1	Bus2	Y(re)	Y(im)
1	5	-3.729	49.7203
2	4	-0.8928	9.9197
2	5	-1.7855	19.8393
3	4	-7.458	99.4406
4	2	-0.8928	9.9197
4	3	-7.458	99.4406
4	5	-3.5711	39.6786
5	1	-3.729	49.7203
5	2	-1.7855	19.8393
5	4	-3.5711	39.6786

Gx.txt

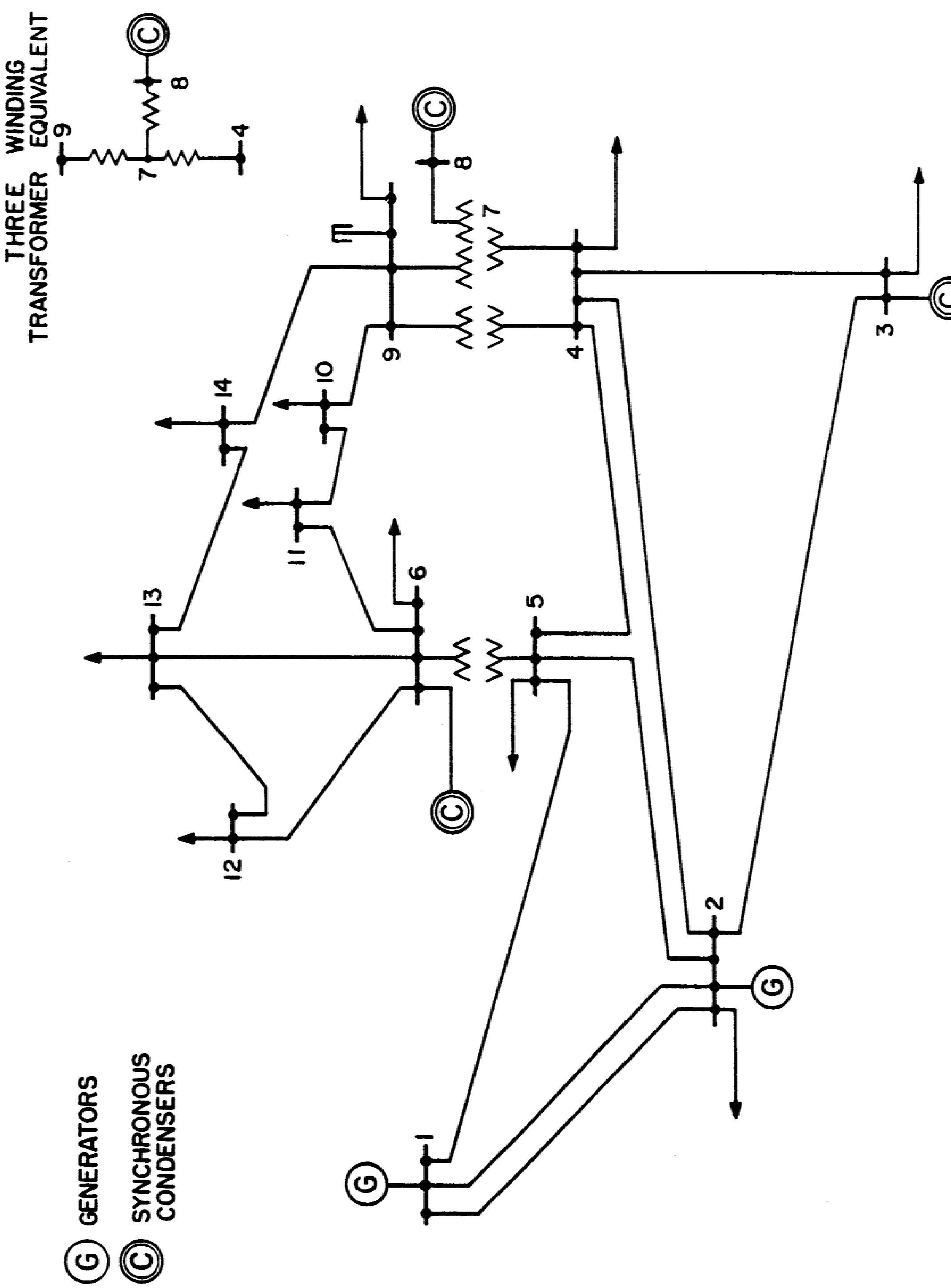
Bus	GenID	P	Q	PMax	H	%Reg
1	G1	394.84	114.29	500	942.5	0.0005
3	G3	520	337.48	700	1696.5	0.0005

Lx.txt

Bus	LoadID	P	Q
2	L2	800	280
3	L3	80	40

Appendix B - 14 Bus System Model

14 Bus System Diagram



AEP 14 BUS TEST SYSTEM BUS CODE DIAGRAM

14 Bus System Simulator Input Files

Bx.txt

Bus	BusID	Y(re)	Y(im)	VRef(re)	VRef(im)
1	Bus1	6.025	-19.4471	1.06	0
2	Bus2	9.5213	-30.2721	1	0
3	Bus3	3.121	-9.8224	1	0
4	Bus4	10.513	-38.6542	1	0
5	Bus5	9.568	-35.5336	1	0
6	Bus6	6.5799	-17.3407	1	0
7	Bus7	0	-19.549	1	0
8	Bus8	0	-5.677	1	0
9	Bus9	5.3261	-24.0925	1	0
10	Bus10	5.7829	-14.7683	1	0
11	Bus11	3.8359	-8.497	1	0
12	Bus12	4.015	-5.4279	1	0
13	Bus13	6.7249	-10.6697	1	0
14	Bus14	2.561	-5.344	1	0

Tx.txt

Bus1	Bus2	Y(re)	Y(im)
1	2	-4.9991	15.2631
1	5	-1.0259	4.235
2	1	-4.9991	15.2631
2	3	-1.135	4.7819
2	4	-1.686	5.1158
2	5	-1.7011	5.1939
3	2	-1.135	4.7819
3	4	-1.986	5.0688
4	2	-1.686	5.1158
4	3	-1.986	5.0688
4	5	-6.841	21.5786
4	7	0	4.8895
4	9	0	1.8555
5	1	-1.0259	4.235
5	2	-1.7011	5.1939
5	4	-6.841	21.5786
5	6	0	4.2574
6	5	0	4.2574
6	11	-1.955	4.0941
6	12	-1.526	3.176
6	13	-3.0989	6.1028
7	4	0	4.8895
7	8	0	5.677
7	9	0	9.0901

8	7	0	5.677
9	4	0	1.8555
9	7	0	9.0901
9	10	-3.902	10.3654
9	14	-1.424	3.0291
10	9	-3.902	10.3654
10	11	-1.8809	4.4029
11	6	-1.955	4.0941
11	10	-1.8809	4.4029
12	6	-1.526	3.176
12	13	-2.489	2.252
13	6	-3.0989	6.1028
13	12	-2.489	2.252
13	14	-1.137	2.315
14	9	-1.424	3.0291
14	13	-1.137	2.315

Gx.txt

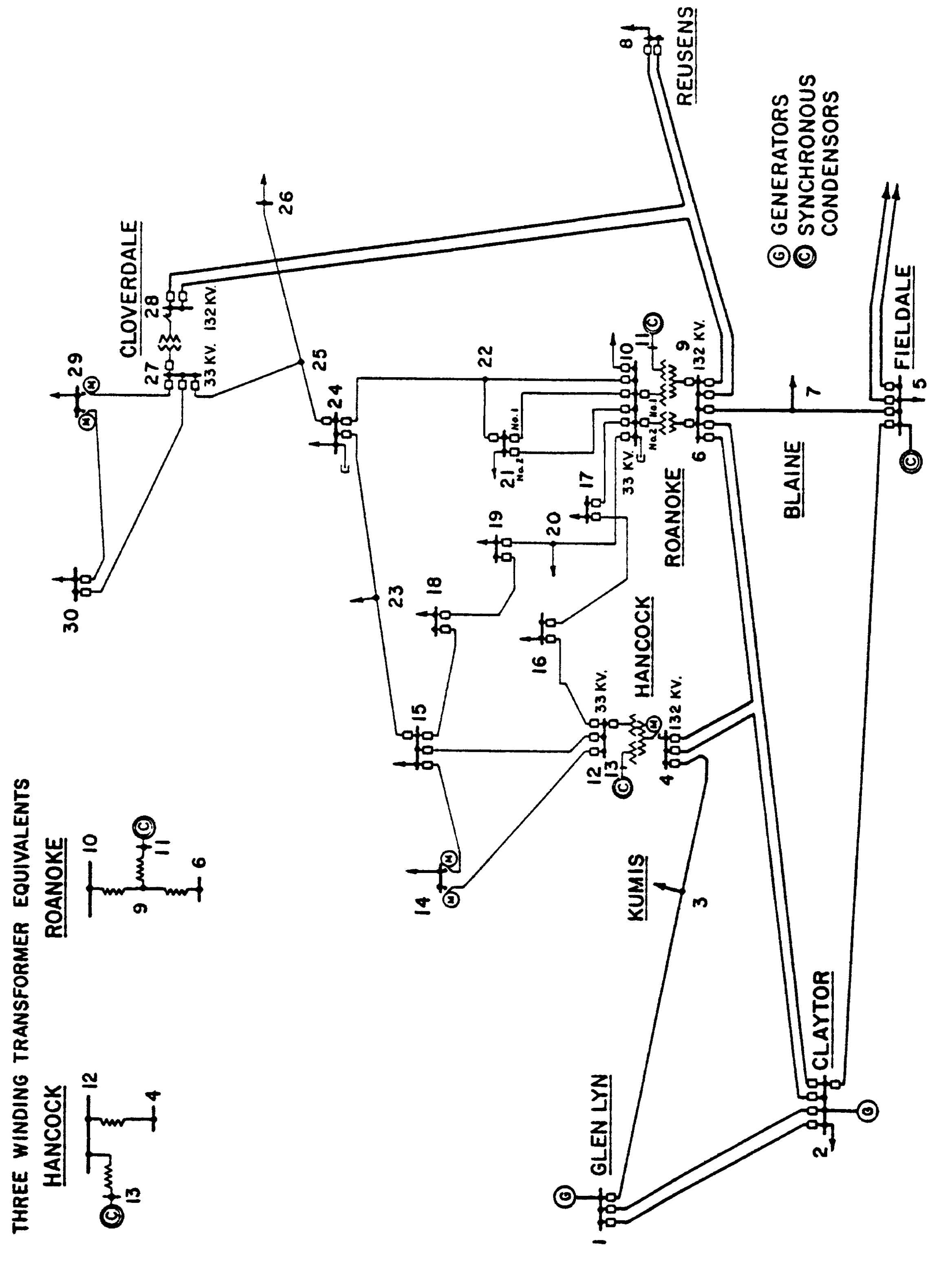
Bus	GenID	P	Q	PMax	H	%Reg
1	G1	232.38	-27.61	250	1319.5	0.0005
2	G2	40	50	50	1394.9	0.0005
3	G3	0	28	50	1583.4	0
6	G6	0	11.1	50	1470.3	0
8	G8	0	20.5	50	1508.0	0

Lx.txt

Bus	LoadID	P	Q
1	Load1	0	0
2	Load2	21.7	12.7
3	Load3	94.2	19
4	Load4	47.8	-3.9
5	Load5	7.6	1.6
6	Load6	11.2	7.5
9	Load9	29.5	16.6
9	Null9	0	0
10	Load10	9	5.8
11	Load11	3.5	1.8
12	Load12	6.1	1.6
13	Load13	13.5	5.8
14	Load14	14.9	5

Appendix C - 30 Bus System Model

30 Bus System Diagram



30 Bus System Simulator Input Files

Bx.txt

Bus	BusID	Y(re)	Y(im)	VRef(re)	VRef(im)
1	Bus1	6.7655	-21.2316	1.06	0
2	Bus2	9.7523	-30.6487	1	0
3	Bus3	9.7363	-29.1379	1	0
4	Bus4	16.3141	-55.5094	1	0
5	Bus5	4.09	-12.1906	1	0
6	Bus6	22.3416	-82.8291	1	0
7	Bus7	6.5442	-18.4567	1	0
8	Bus8	7.7333	-26.5275	1	0
9	Bus9	0	-18.7063	1	0
10	Bus10	13.4621	-41.3838	1	0
11	Bus11	0	-4.8077	1	0
12	Bus12	6.574	-24.4242	1	0
13	Bus13	0	-7.1429	1	0
14	Bus14	4.0175	-5.4243	1	0
15	Bus15	9.3655	-16.0116	1	0
16	Bus16	3.2711	-8.9451	1	0
17	Bus17	5.2751	-15.1582	1	0
18	Bus18	4.8865	-9.9062	1	0
19	Bus19	8.958	-17.9835	1	0
20	Bus20	7.6672	-15.7501	1	0
21	Bus21	21.8765	-45.1084	1	0
22	Bus22	21.9345	-43.4829	1	0
23	Bus23	3.4298	-6.9653	1	0
24	Bus24	5.3118	-9.1883	1	0
25	Bus25	4.4957	-7.865	1	0
26	Bus26	1.2165	-1.8171	1	0
27	Bus27	3.6523	-9.4604	1	0
28	Bus28	5.8068	-22.6715	1	0
29	Bus29	1.9076	-3.6044	1	0
30	Bus30	1.5995	-3.0173	1	0

Tx.txt

Bus1	Bus2	Y(re)	Y(im)
1	1	6.7655	-21.2316
1	2	-5.2246	15.6467
1	3	-1.5409	5.6317
2	1	-5.2246	15.6467
2	2	9.7523	-30.6487
2	4	-1.7055	5.1974
2	5	-1.136	4.7725
2	6	-1.6861	5.1165

3	1	-1.5409	5.6317
3	3	9.7363	-29.1379
3	4	-8.1954	23.5309
4	2	-1.7055	5.1974
4	3	-8.1954	23.5309
4	4	16.3141	-55.5094
4	6	-6.4131	22.3112
4	12	0	4.1913
5	2	-1.136	4.7725
5	5	4.09	-12.1906
5	7	-2.954	7.4493
6	2	-1.6861	5.1165
6	4	-6.4131	22.3112
6	6	22.3416	-82.8291
6	7	-3.5902	11.0261
6	8	-6.2893	22.0126
6	9	0	4.9158
6	10	0	1.8561
6	28	-4.3628	15.4636
7	5	-2.954	7.4493
7	6	-3.5902	11.0261
7	7	6.5442	-18.4567
8	6	-6.2893	22.0126
8	8	7.7333	-26.5275
8	28	-1.444	4.5408
9	6	0	4.9158
9	9	0	-18.7063
9	10	0	9.0909
9	11	0	4.8077
10	6	0	1.8561
10	9	0	9.0909
10	10	13.4621	-41.3838
10	17	-3.956	10.3174
10	20	-1.7848	3.9854
10	21	-5.1019	10.9807
10	22	-2.6193	5.4008
11	9	0	4.8077
11	11	0	-4.8077
12	4	0	4.1913
12	12	6.574	-24.4242
12	13	0	7.1429
12	14	-1.5266	3.1734
12	15	-3.0954	6.0973
12	16	-1.952	4.1044
13	12	0	7.1429
13	13	0	-7.1429
14	12	-1.5266	3.1734
14	14	4.0175	-5.4243

14	15	-2.491	2.2509
15	12	-3.0954	6.0973
15	14	-2.491	2.2509
15	15	9.3655	-16.0116
15	18	-1.8108	3.6874
15	23	-1.9683	3.9761
16	12	-1.952	4.1044
16	16	3.2711	-8.9451
16	17	-1.3191	4.8408
17	10	-3.956	10.3174
17	16	-1.3191	4.8408
17	17	5.2751	-15.1582
18	15	-1.8108	3.6874
18	18	4.8865	-9.9062
18	19	-3.0757	6.2188
19	18	-3.0757	6.2188
19	19	8.958	-17.9835
19	20	-5.8824	11.7647
20	10	-1.7848	3.9854
20	19	-5.8824	11.7647
20	20	7.6672	-15.7501
21	10	-5.1019	10.9807
21	21	21.8765	-45.1084
21	22	-16.7746	34.1277
22	10	-2.6193	5.4008
22	21	-16.7746	34.1277
22	22	21.9345	-43.4829
22	24	-2.5405	3.9544
23	15	-1.9683	3.9761
23	23	3.4298	-6.9653
23	24	-1.4614	2.9892
24	22	-2.5405	3.9544
24	23	-1.4614	2.9892
24	24	5.3118	-9.1883
24	25	-1.3099	2.2876
25	24	-1.3099	2.2876
25	25	4.4957	-7.865
25	26	-1.2165	1.8171
25	27	-1.9693	3.7602
26	25	-1.2165	1.8171
26	26	1.2165	-1.8171
27	25	-1.9693	3.7602
27	27	3.6523	-9.4604
27	28	0	2.6087
27	29	-0.9955	1.881
27	30	-0.6875	1.294
28	6	-4.3628	15.4636
28	8	-1.444	4.5408

28	27	0	2.6087
28	28	5.8068	-22.6715
29	27	-0.9955	1.881
29	29	1.9076	-3.6044
29	30	-0.9121	1.7234
30	27	-0.6875	1.294
30	29	-0.9121	1.7234
30	30	1.5995	-3.0173

Gx.txt

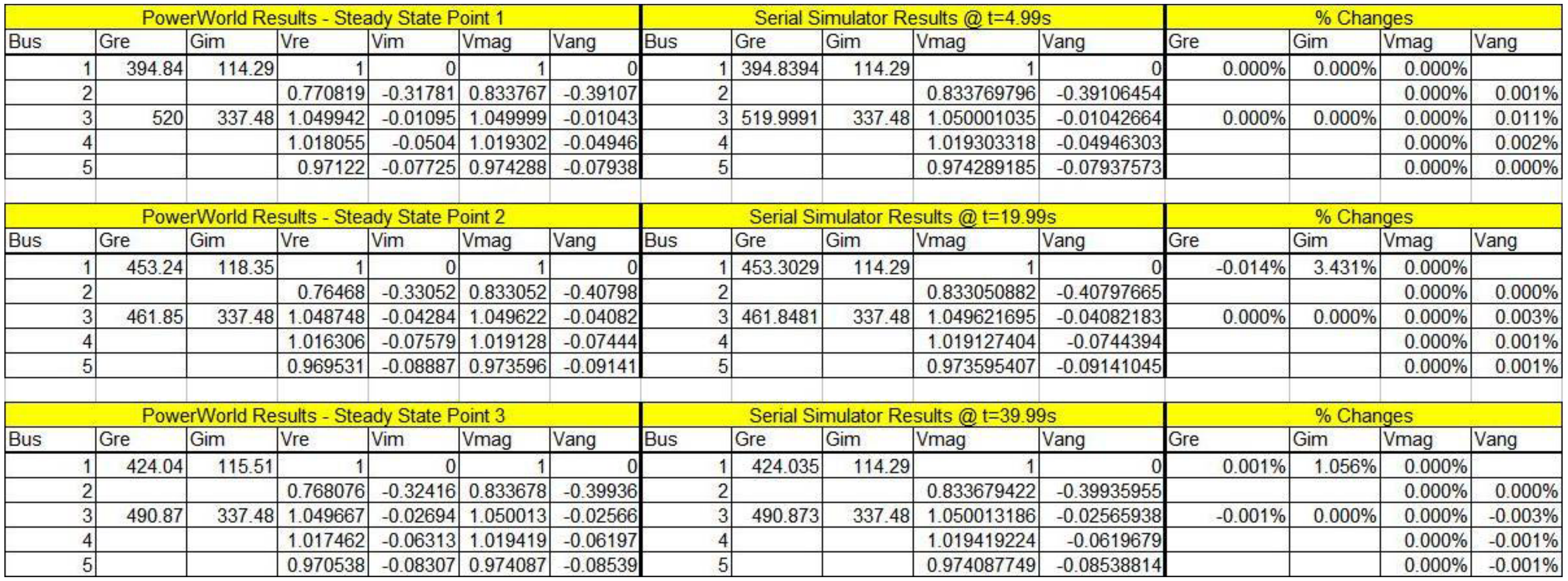
Name	GenID	P	Q	PMax	H	%Reg
1	GlenLyn	260.93	-17.22	350	942.5	0.0005
2	Claytor	40	50	50	1319.5	0.0005
5	Fieldale	0	37	50	565.5	0
8	Reusens	0	37.3	50	377.0	0
11	Roanoke	0	16.2	50	942.5	0
13	Hancock	0	10.6	50	754.0	0

Lx.txt

Bus	LoadID	P	Q
2	Claytor	21.7	12.7
3	Kumis	2.4	1.2
4	Hancock	7.6	1.6
5	Fieldale	94.2	19
7	Blaine	22.8	10.9
8	Reusens	30	30
10	Roanoke	5.8	2
12	Hancock	11.2	7.5
14	Bus14	6.2	1.6
15	Bus15	8.2	2.5
16	Bus16	3.5	1.8
17	Bus17	9	5.8
18	Bus18	3.2	0.9
19	Bus19	9.5	3.4
20	Bus20	2.2	0.7
21	Bus21	17.5	11.2
23	Bus23	3.2	1.6
24	Bus24	8.7	6.7
26	Bus26	3.5	2.3
29	Bus29	2.4	0.9
30	Bus30	10.6	1.9

Appendix D - PowerWorld and Serial Results

5 Bus System - Results Comparison



PowerWorld Results - Steady State Point 1							Serial Simulator Results @ t=4.99s					% Changes			
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	394.84	114.29	1	0	1	0	1	394.8394	114.29	1	0	0.000%	0.000%	0.000%	
2			0.770819	-0.31781	0.833767	-0.39107	2			0.833769796	-0.39106454			0.000%	0.001%
3	520	337.48	1.049942	-0.01095	1.049999	-0.01043	3	519.9991	337.48	1.050001035	-0.01042664	0.000%	0.000%	0.000%	0.011%
4			1.018055	-0.0504	1.019302	-0.04946	4			1.019303318	-0.04946303			0.000%	0.002%
5			0.97122	-0.07725	0.974288	-0.07938	5			0.974289185	-0.07937573			0.000%	0.000%

PowerWorld Results - Steady State Point 2							Serial Simulator Results @ t=19.99s					% Changes			
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	453.24	118.35	1	0	1	0	1	453.3029	114.29	1	0	-0.014%	3.431%	0.000%	
2			0.76468	-0.33052	0.833052	-0.40798	2			0.833050882	-0.40797665			0.000%	0.000%
3	461.85	337.48	1.048748	-0.04284	1.049622	-0.04082	3	461.8481	337.48	1.049621695	-0.04082183	0.000%	0.000%	0.000%	0.003%
4			1.016306	-0.07579	1.019128	-0.07444	4			1.019127404	-0.0744394			0.000%	0.001%
5			0.969531	-0.08887	0.973596	-0.09141	5			0.973595407	-0.09141045			0.000%	0.001%

PowerWorld Results - Steady State Point 3							Serial Simulator Results @ t=39.99s					% Changes			
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	424.04	115.51	1	0	1	0	1	424.035	114.29	1	0	0.001%	1.056%	0.000%	
2			0.768076	-0.32416	0.833678	-0.39936	2			0.833679422	-0.39935955			0.000%	0.000%
3	490.87	337.48	1.049667	-0.02694	1.050013	-0.02566	3	490.873	337.48	1.050013186	-0.02565938	-0.001%	0.000%	0.000%	-0.003%
4			1.017462	-0.06313	1.019419	-0.06197	4			1.019419224	-0.0619679			0.000%	-0.001%
5			0.970538	-0.08307	0.974087	-0.08539	5			0.974087749	-0.08538814			0.000%	-0.001%

14 Bus System Results Comparison

PowerWorld Results - Steady State Point 1							Serial Simulator Results @ t=4.99s					% Changes			
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	232.38	-27.61	1.06	0	1.06	0	1	2.32399	-0.2761	1.06	0	-0.008%	0.000%	0.000%	
2	40	50	1.045902	-0.09238	1.049973	-0.08809	2	0.400038	0.5	1.049992019	-0.08810372	-0.009%	0.000%	-0.002%	-0.013%
3			0.993356	-0.22472	1.018457	-0.22248	3			1.018489339	-0.22248368			-0.003%	-0.004%
4			1.007207	-0.18368	1.023819	-0.18038	4			1.023862661	-0.18040197			-0.004%	-0.010%
5			1.012564	-0.1566	1.024603	-0.15345	5			1.02464562	-0.1534633			-0.004%	-0.012%
6			1.041948	-0.26287	1.074597	-0.24713	6			1.074715757	-0.24714748			-0.011%	-0.006%
7			1.041742	-0.2473	1.070694	-0.23308	7			1.070776194	-0.23309701			-0.008%	-0.008%
8			1.073583	-0.25486	1.103419	-0.23308	8			1.103499913	-0.23309702			-0.007%	-0.008%
9			1.028104	-0.27388	1.063958	-0.26034	9			1.064058916	-0.26036131			-0.010%	-0.007%
10			1.022072	-0.27514	1.058457	-0.26296	10			1.058561481	-0.26297766			-0.010%	-0.006%
11			1.027979	-0.27056	1.062988	-0.25736	11			1.063100358	-0.25737555			-0.011%	-0.006%
12			1.023949	-0.27451	1.060107	-0.26193	12			1.06025001	-0.2619539			-0.013%	-0.009%
13			1.019147	-0.27487	1.055563	-0.26344	13			1.05570017	-0.26345581			-0.013%	-0.008%
14			1.002147	-0.28701	1.042437	-0.27893	14			1.042572141	-0.27895391			-0.013%	-0.008%

PowerWorld Results - Steady State Point 2							Serial Simulator Results @ t=19.99s					% Changes			
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	249.83	-24.23	1.06	0	1.06	0	1	2.499199	-0.2761	1.06	0	-0.036%	-13.950%	0.000%	
2	23.51	50	1.041898	-0.10006	1.046691	-0.09574	2	0.23508	0.5	1.04670947	-0.09575294	0.009%	0.000%	-0.002%	-0.014%
3			0.988496	-0.23155	1.015254	-0.2301	3			1.015287515	-0.23011045			-0.003%	-0.005%
4			1.003127	-0.18979	1.020923	-0.18699	4			1.020967294	-0.18700809			-0.004%	-0.010%
5			1.008966	-0.16218	1.021918	-0.15938	5			1.021960569	-0.1593978			-0.004%	-0.012%
6			1.037249	-0.26902	1.071568	-0.25377	6			1.071686665	-0.25378322			-0.011%	-0.006%
7			1.03705	-0.25366	1.067621	-0.23989	7			1.067704058	-0.23990668			-0.008%	-0.008%
8			1.068925	-0.26146	1.100436	-0.23989	8			1.100516502	-0.23990665			-0.007%	-0.008%
9			1.023117	-0.28014	1.060777	-0.26726	9			1.060878359	-0.26728073			-0.010%	-0.007%
10			1.017086	-0.28133	1.055278	-0.26986	10			1.05538265	-0.26987561			-0.010%	-0.006%
11			1.02311	-0.27671	1.059869	-0.26414	11			1.059981812	-0.26415585			-0.011%	-0.006%
12			1.019108	-0.28058	1.057028	-0.26867	12			1.057170418	-0.26869067			-0.013%	-0.009%
13			1.014272	-0.28092	1.052456	-0.27019	13			1.052593452	-0.27021454			-0.013%	-0.008%
14			0.997052	-0.29307	1.039232	-0.28589	14			1.039366974	-0.28590802			-0.013%	-0.008%

PowerWorld Results - Steady State Point 3							Serial Simulator Results @ t=39.99s					% Changes			
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	241.1	-25.97	1.06	0	1.06	0	1	2.411214	-0.2761	1.06	0	-0.009%	-6.315%	0.000%	
2	31.75	50	1.043933	-0.09623	1.048358	-0.09192	2	0.317483	0.5	1.048360159	-0.09192446	0.005%	0.000%	0.000%	-0.010%
3			0.990959	-0.22814	1.016882	-0.22628	3			1.016898461	-0.22629075			-0.002%	-0.004%
4			1.005196	-0.18674	1.022395	-0.18368	4			1.022424703	-0.18370071			-0.003%	-0.009%
5			1.010791	-0.1594	1.023282	-0.15641	5			1.023312401	-0.15642731			-0.003%	-0.011%
6			1.03963	-0.26595	1.073108	-0.25044	6			1.073211779	-0.25045978			-0.010%	-0.006%
7			1.039428	-0.25049	1.069184	-0.23648	7			1.069250481	-0.23649641			-0.006%	-0.008%
8			1.071286	-0.25816	1.101954	-0.23648	8			1.102018198	-0.23649641			-0.006%	-0.009%
9			1.025644	-0.27702	1.062395	-0.2638	9			1.062479412	-0.26381494			-0.008%	-0.007%
10			1.019612	-0.27824	1.056895	-0.2664	10			1.056982888	-0.26642054			-0.008%	-0.006%
11			1.025577	-0.27364	1.061456	-0.26074	11			1.061551821	-0.26075982			-0.009%	-0.006%
12			1.021561	-0.27756	1.058595	-0.26529	12			1.058720965	-0.26531634			-0.012%	-0.009%
13			1.016742	-0.2779	1.054037	-0.26681	13			1.054157627	-0.26682919			-0.011%	-0.008%
14			0.999632	-0.29005	1.040862	-0.2824	14			1.040980581	-0.28242446			-0.011%	-0.008%

30 Bus System - Results Comparison

PowerWorld Results - Steady State Point 1							Serial Simulator @ t=4.99s									
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	MMim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	260.93	-17.22	1.06	0	1.06	0	1	2.609747	-0.1722	-0.000341	1.06000000	0.00000000	-0.0171%	0.0000%	0.0000%	
2	40	50	1.038743	-0.097339	1.043294	-0.093436	2	0.400064	0.5		1.04328201	-0.09344979	-0.0160%	0.0000%	0.0011%	-0.0152%
3			1.012177	-0.133849	1.020989	-0.131476	3				1.02094672	-0.13150195			0.0041%	-0.0198%
4			0.998783	-0.163292	1.012043	-0.162057	4				1.01201450	-0.16208750			0.0029%	-0.0187%
5	0	37	0.979676	-0.247277	1.010401	-0.247243	5	0	0.37		1.01039253	-0.24726819			0.0009%	-0.0103%
6			0.9918	-0.193969	1.01059	-0.193135	6				1.01056563	-0.19316393			0.0024%	-0.0150%
7			0.977568	-0.223276	1.002742	-0.224548	7				1.00272348	-0.22457376			0.0018%	-0.0116%
8	0	37.3	0.988984	-0.206878	1.01039	-0.206209	8	0	0.373		1.01036503	-0.20623831			0.0025%	-0.0142%
9			1.019578	-0.256254	1.051288	-0.246233	9				1.05125618	-0.24627154			0.0030%	-0.0156%
10			1.006502	-0.282874	1.045497	-0.273979	10				1.04546793	-0.27402066			0.0028%	-0.0152%
11	0	16.2	1.04977	-0.263843	1.082419	-0.246234	11	0	0.162		1.08238732	-0.24627154			0.0029%	-0.0153%
12			1.02169	-0.272634	1.05744	-0.26077	12				1.05743980	-0.26080127			0.0000%	-0.0120%
13	0	10.6	1.035074	-0.276205	1.071292	-0.260769	13	0	0.106		1.07129215	-0.26080127			0.0000%	-0.0122%
14			1.003058	-0.284456	1.042612	-0.276333	14				1.04262070	-0.27635258			-0.0008%	-0.0069%
15			0.998186	-0.284805	1.038022	-0.277937	15				1.03801946	-0.27797228			0.0002%	-0.0125%
16			1.006622	-0.279615	1.044736	-0.270945	16				1.04473284	-0.27098745			0.0003%	-0.0157%
17			1.00067	-0.284279	1.040267	-0.276796	17				1.04024570	-0.27683817			0.0020%	-0.0152%
18			0.985954	-0.292782	1.028507	-0.288659	18				1.02850076	-0.28868462			0.0006%	-0.0089%
19			0.982675	-0.295051	1.026014	-0.291689	19				1.02600467	-0.29171343			0.0009%	-0.0084%
20			0.987599	-0.292844	1.030102	-0.288262	20				1.03008755	-0.28829118			0.0014%	-0.0101%
21			0.99238	-0.287189	1.0331	-0.281699	21				1.03306635	-0.28174398			0.0033%	-0.0161%
22			0.992961	-0.287093	1.033631	-0.281453	22				1.03359589	-0.28149616			0.0034%	-0.0153%
23			0.986158	-0.288661	1.027537	-0.284758	23				1.02751240	-0.28480183			0.0024%	-0.0154%
24			0.979911	-0.29012	1.021957	-0.287845	24				1.02191528	-0.28787510			0.0040%	-0.0104%
25			0.97797	-0.281624	1.017712	-0.280382	25				1.01767674	-0.28039300			0.0034%	-0.0039%
26			0.958939	-0.283757	1.000041	-0.287698	26				1.00000514	-0.28771263			0.0036%	-0.0052%
27			0.986197	-0.274251	1.02362	-0.271236	27				1.02361011	-0.27124584			0.0010%	-0.0036%
28			0.986278	-0.204056	1.007166	-0.204017	28				1.00714121	-0.20404534			0.0025%	-0.0141%
29			0.9611	-0.28962	1.003789	-0.292688	29				1.00380626	-0.29268869			-0.0017%	-0.0003%
30			0.945597	-0.300903	0.992319	-0.308083	30				0.99236546	-0.30807442			-0.0047%	0.0027%

PowerWorld Results - Steady State Point 2							Serial Simulator @ t=19.99s									
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	MMim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	274.17	-14.36	1.06	0	1.06	0	1	2.741967	-0.1722	-0.02901	1.06000000	0.00000000	-0.0098%	-19.9164%	0.0000%	
2	27.61	50	1.035667	-0.102921	1.040768	-0.099051	2	0.276095	0.5		1.04074414	-0.09905600	0.0017%	0.0000%	0.0023%	-0.0047%
3			1.00987	-0.137036	1.019125	-0.134873	3				1.01912621	-0.13488914			-0.0001%	-0.0121%
4			0.995918	-0.167189	1.009854	-0.166323	4				1.00985236	-0.16634249			0.0001%	-0.0115%
5	0	37	0.975793	-0.252258	1.007872	-0.252977	5	0	0.37		1.00786283	-0.25299053			0.0009%	-0.0052%
6			0.988562	-0.198268	1.008248	-0.197936	6				1.00824138	-0.19795491			0.0007%	-0.0096%
7			0.97401	-0.227825	1.0003	-0.229773	7				1.00029139	-0.22978902			0.0008%	-0.0069%
8	0	37.3	0.985669	-0.21119	1.00804	-0.211069	8	0	0.373		1.00803244	-0.21108703			0.0007%	-0.0084%
9			1.015862	-0.260709	1.048783	-0.251217	9				1.04877075	-0.25123989			0.0011%	-0.0093%
10			1.002556	-0.28727	1.042901	-0.279061	10				1.04289325	-0.27908619			0.0007%	-0.0091%
11	0	16.2	1.046084	-0.268464	1.079984	-0.251216	11	0	0.162		1.07997152	-0.25123989			0.0011%	-0.0097%
12			1.017945	-0.276963	1.05495	-0.26565	12				1.054973989	-0.265666462			-0.0022%	-0.0062%
13	0	10.6	1.031342	-0.280608	1.068834	-0.26565	13	0	0.106		1.068857885	-0.265666461			-0.0022%	-0.0062%
14			0.999195	-0.288737	1.040077	-0.281307	14				1.040109217	-0.281310404			-0.0031%	-0.0013%
15			0.994289	-0.289076	1.035459	-0.282937	15				1.035480497	-0.282952902			-0.0021%	-0.0058%
16			1.00275	-0.283935	1.042174	-0.275933	16				1.04219553	-0.275957			-0.0021%	-0.0086%
17			0.996722	-0.288628	1.037671	-0.281867	17				1.037672016	-0.281893156			-0.0001%	-0.0091%
18			0.981958	-0.297039	1.025901	-0.293746	18				1.025917487	-0.293755278			-0.0016%	-0.0033%
19			0.97864	-0.299318	1.02339	-0.296816	19				1.02340351	-0.296821405			-0.0013%	-0.0018%
20			0.98358	-0.29714	1.027483	-0.293383	20				1.02749067	-0.293393309			-0.0007%	-0.0036%
21			0.988377	-0.291521	1.030473	-0.286817	21				1.03046035	-0.286843691			0.0012%	-0.0094%
22			0.98896	-0.291423	1.031004	-0.286565	22				1.030991309	-0.286593194			0.0012%	-0.0097%
23			0.98218	-0.292906	1.024925	-0.289823	23				1.024924525	-0.289850465			0.0001%	-0.0094%
24			0.97587	-0.294372	1.019302	-0.292971	24				1.019284225	-0.292985757			0.0018%	-0.0052%
25			0.973988	-0.285919	1.015087	-0.285533	25				1.015073668	-0.285528981			0.0013%	0.0016%
26			0.954895	-0.287962	0.99737	-0.292891	26				0.997354969	-0.292886877			0.0015%	0.0014%
27			0.98229	-0.278615	1.021039	-0.276379	27				1.021048394	-0.276373899			-0.0010%	0.0019%
28			0.98296	-0.208341	1.004797	-0.208862	28				1.004789733	-0.208877263			0.0007%	-0.0075%
29			0.957044	-0.293891	1.001152	-0.297941	29				1.001189297	-0.297926726			-0.0037%	0.0049%
30			0.941439	-0.305124	0.98965	-0.313421	30				0.989716724	-0.313393946			-0.0067%	0.0087%

PowerWorld Results - Steady State Point 3							Serial Simulator @ t=39.99s										
Bus	Gre	Gim	Vre	Vim	Vmag	Vang	Bus	Gre	Gim	MMim	Vmag	Vang	Gre	Gim	Vmag	Vang
1	267.55	-15.75	1.06	0	1.06	0	1	2.675807	-0.1722	-0.01446	1.06000000	0.00000000	-0.0115%	-9.3333%	0.0000%	
2	33.81	50	1.037197	-0.100124	1.042018	-0.096235	2	0.338072	0.5		1.04201926	-0.09624889	0.0081%	0.0000%	-0.0001%	-0.0144%
3			1.010992	-0.135438	1.020024	-0.133173	3				1.02004193	-0.13319336			-0.0018%	-0.0156%
4			0.997325	-0.165236	1.01092	-0.164188	4				1.01093966	-0.16421173			-0.0019%	-0.0146%
5	0	37	0.977719	-0.249763	1.009116	-0.250106	5	0	0.37		1.00913442	-0.25012346			-0.0018%	-0.0070%
6			0.990159	-0.196113	1.009393	-0.195532	6				1.00941012	-0.19555525			-0.0017%	-0.0121%
7			0.975769	-0.225544	1.001496	-0.227155	7				1.00151421	-0.22717637			-0.0018%	-0.0092%
8	0	37.3	0.987304	-0.209029	1.009189	-0.208636	8	0	0.373		1.00920538	-0.20865829			-0.0016%	-0.0107%
9			1.017695	-0.258477	1.050006	-0.248724	9				1.05002063	-0.24875066			-0.0014%	-0.0109%
10			1.004503	-0.285067	1.044169	-0.276519	10				1.04418806	-0.27654788			-0.0018%	-0.0105%
11	0	16.2	1.047902	-0.266149	1.081172	-0.248724	11	0	0.162		1.08118634	-0.24875066			-0.0013%	-0.0109%
12			1.01979	-0.274794	1.056164	-0.26321	12				1.05621414	-0.26322870			-0.0047%	-0.0072%
13	0	10.6	1.033181	-0.278402	1.070033	-0.263209	13	0	0.106		1.07008215	-0.26322870			-0.0046%	-0.0073%
14			1.001099	-0.286591	1.041313	-0.278819	14				1.04137236	-0.27882602			-0.0057%	-0.0024%
15			0.996209	-0.286937	1.036709	-0.280438	15				1.03675744	-0.28045708			-0.0047%	-0.0067%
16			1.004657	-0.281772	1.043423	-0.273441	16				1.04347160	-0.27346682			-0.0047%	-0.0096%
17			0.998669	-0.286449	1.038938	-0.279332	17				1.03896635	-0.27936009			-0.0027%	-0.0102%
18			0.983928	-0.294906	1.027173	-0.291203	18				1.02721671	-0.29121419			-0.0043%	-0.0039%
19			0.98063	-0.297181	1.024672	-0.294254	19				1.02471171	-0.29426157			-0.0039%	-0.0027%
20			0.985562	-0.294988	1.028762	-0.290823	20				1.02879670	-0.29083646			-0.0034%	-0.0046%
21			0.990351	-0.289351	1.031755	-0.284258	21				1.03177094	-0.28428815			-0.0015%	-0.0106%
22			0.990933	-0.289253	1.032287	-0.284009	22				1.03230119	-0.28403900			-0.0014%	-0.0106%
23			0.984141	-0.290779	1.0262	-0.287291	23				1.02622607	-0.28732048			-0.0026%	-0.0103%
24			0.977863	-0.29224	1.020598	-0.290407	24				1.02060745	-0.29042467			-0.0009%	-0.0062%
25			0.975952	-0.283766	1.016369	-0.282957	25				1.01638277	-0.28295526			-0.0014%	0.0005%
26			0.956889	-0.285856	0.998674	-0.290296	26				0.99868779	-0.29029387			-0.0014%	0.0006%
27			0.984218	-0.276428	1.0223	-0.273807	27				1.02233665	-0.27380425			-0.0036%	0.0008%
28			0.984596	-0.206194	1.005955	-0.206437	28				1.00597219	-0.20645696			-0.0017%	-0.0099%
29			0.959046	-0.29175	1.002441	-0.295313	29				1.00250538	-0.29530166			-0.0065%	0.0040%
30			0.943491	-0.303009	0.990954	-0.310752	30				0.99104881	-0.31072782			-0.0096%	0.0079%

Appendix E - Dynamic Serial Simulation

5 Bus System Validation

Fsyn	ωsyn	Δt	System Base
60	376.991	0.01	100

Generator	Gen Pmax	Gen Hnom	H - System Base
1	500	2.5	12.5
3	700	4.5	31.5
		System H	44

Impulse Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
7.50	60.00209386	377.0042745	0.162040244	0.000293052	0.000293052	0.000%
7.51	60.00209677	377.0042928	0.160861599	0.000290921	0.000290921	0.000%
7.52	60.00209966	377.004311	0.159691521	0.000288805	0.000288805	0.000%
7.53	60.00210252	377.004329	0.158529948	0.000286704	0.000286704	0.000%
7.54	60.00210537	377.0043469	0.157376817	0.000284618	0.000284618	0.000%
7.55	60.0021082	377.0043646	0.156232068	0.000282548	0.000282548	0.000%
7.56	60.002111	377.0043822	0.15509564	0.000280493	0.000280493	0.000%
7.57	60.00211379	377.0043997	0.153967471	0.000278453	0.000278453	0.000%
7.58	60.00211655	377.0044171	0.152847504	0.000276427	0.000276427	0.000%
7.59	60.00211929	377.0044343	0.151735677	0.000274416	0.000274416	0.000%

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
22.50	60.00144802	377.0002166	-0.080901294	-0.000146313	-0.000146313	0.000%
22.51	60.00144657	377.0002075	-0.080312822	-0.000145249	-0.000145249	0.000%
22.52	60.00144513	377.0001984	-0.079728631	-0.000144192	-0.000144192	0.000%
22.53	60.00144369	377.0001894	-0.079148692	-0.000143143	-0.000143143	0.000%
22.54	60.00144227	377.0001805	-0.078572972	-0.000142102	-0.000142102	0.000%
22.55	60.00144086	377.0001716	-0.078001442	-0.000141068	-0.000141068	0.000%
22.56	60.00143946	377.0001628	-0.07743407	-0.000140042	-0.000140042	0.000%
22.57	60.00143807	377.0001541	-0.076870827	-0.000139024	-0.000139024	0.000%
22.58	60.00143669	377.0001454	-0.076311682	-0.000138012	-0.000138012	0.000%
22.59	60.00143532	377.0001368	-0.075756606	-0.000137009	-0.000137009	0.000%

Slow Ramp Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
25	60.00046592	376.9940459	0.013708895	2.48E-05	2.47934E-05	0.000%
25.01	60.00046617	376.9940475	0.013609249	2.46E-05	2.46132E-05	0.000%
25.02	60.00046641	376.994049	0.013510061	2.44E-05	2.44338E-05	0.000%
25.03	60.00046665	376.9940505	0.0134114	2.43E-05	2.42554E-05	0.000%
25.04	60.0004669	376.994052	0.013313315	2.41E-05	2.4078E-05	0.000%
25.05	60.00046713	376.9940535	0.013215842	2.39E-05	2.39017E-05	0.000%
25.06	60.00046737	376.994055	0.013119005	2.37E-05	2.37266E-05	0.000%
25.07	60.00046761	376.9940565	0.01302282	2.36E-05	2.35526E-05	0.000%
25.08	60.00046784	376.994058	0.012927298	2.34E-05	2.33799E-05	0.000%
25.09	60.00046807	376.9940594	0.012832446	2.32E-05	2.32083E-05	0.000%

Ramp Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
7.5	60.00016928	376.992182	0.057879407	0.000104679	0.000104679	0.000%
7.51	60.00017033	376.9921886	0.057960728	0.000104826	0.000104826	0.000%
7.52	60.00017138	376.9921952	0.058041457	0.000104972	0.000104972	0.000%
7.53	60.00017243	376.9922018	0.0581216	0.000105117	0.000105117	0.000%
7.54	60.00017348	376.9922084	0.05820116	0.000105261	0.000105261	0.000%
7.55	60.00017454	376.9922151	0.058280141	0.000105404	0.000105404	0.000%
7.56	60.00017559	376.9922217	0.058358548	0.000105546	0.000105546	0.000%
7.57	60.00017665	376.9922283	0.058436384	0.000105687	0.000105687	0.000%
7.58	60.00017771	376.992235	0.058513655	0.000105826	0.000105826	0.000%
7.59	60.00017877	376.9922416	0.058590363	0.000105965	0.000105965	0.000%

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
27.5	60.00028527	376.9929108	-0.11567871	-0.000209213	-0.000209213	0.000%
27.51	60.00028317	376.9928977	-0.11583559	-0.000209497	-0.000209497	0.000%
27.52	60.00028107	376.9928845	-0.11599577	-0.000209787	-0.000209787	0.000%
27.53	60.00027897	376.9928713	-0.116155886	-0.000210076	-0.000210076	0.000%

14 Bus System Validation

Fsyn	ωsyn	Δt	System Base
60	376.991	0.01	100

Generator	Gen Pmax	Gen Hnom	H - System Base
1	250	3.5	8.75
2	50	3.7	1.85
3	50	4.2	2.1
6	50	3.9	1.95
8	50	4	2
		System H	16.65

Impulse Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
7.50	60.00696598	377.034887	0.298598905	0.001426967	0.001426966	0.000%
7.51	60.00698018	377.0349762	0.297192293	0.001420244	0.001420244	0.000%
7.52	60.00699431	377.035065	0.295748441	0.001413344	0.001413344	0.000%
7.53	60.00700838	377.0351534	0.294347518	0.001406649	0.001406648	0.000%
7.54	60.00702238	377.0352413	0.292882306	0.001399646	0.001399646	0.000%
7.55	60.00703631	377.0353288	0.291500475	0.001393042	0.001393042	0.000%
7.56	60.00705017	377.0354159	0.290082075	0.001386264	0.001386263	0.000%
7.57	60.00706397	377.0355026	0.288706622	0.00137969	0.00137969	0.000%
7.58	60.00707769	377.0355889	0.28726754	0.001372813	0.001372813	0.000%
7.59	60.00709136	377.0356747	0.285911186	0.001366331	0.00136633	0.000%

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
22.50	60.00641457	377.0314224	-0.14915923	-0.000712819	-0.00071282	0.000%
22.51	60.00640748	377.0313778	-0.148409042	-0.000709234	-0.000709235	0.000%
22.52	60.00640042	377.0313335	-0.147701747	-0.000705854	-0.000705855	0.000%
22.53	60.0063934	377.0312893	-0.146962485	-0.000702322	-0.000702322	0.000%
22.54	60.00638641	377.0312454	-0.146291634	-0.000699116	-0.000699116	0.000%
22.55	60.00637945	377.0312017	-0.145557778	-0.000695609	-0.000695609	0.000%
22.56	60.00637253	377.0311582	-0.144864551	-0.000692296	-0.000692296	0.000%
22.57	60.00636564	377.0311149	-0.144198421	-0.000689113	-0.000689113	0.000%
22.58	60.00635878	377.0310718	-0.143469936	-0.000685631	-0.000685632	0.000%
22.59	60.00635196	377.031029	-0.142784514	-0.000682356	-0.000682356	0.000%

Slow Ramp Test

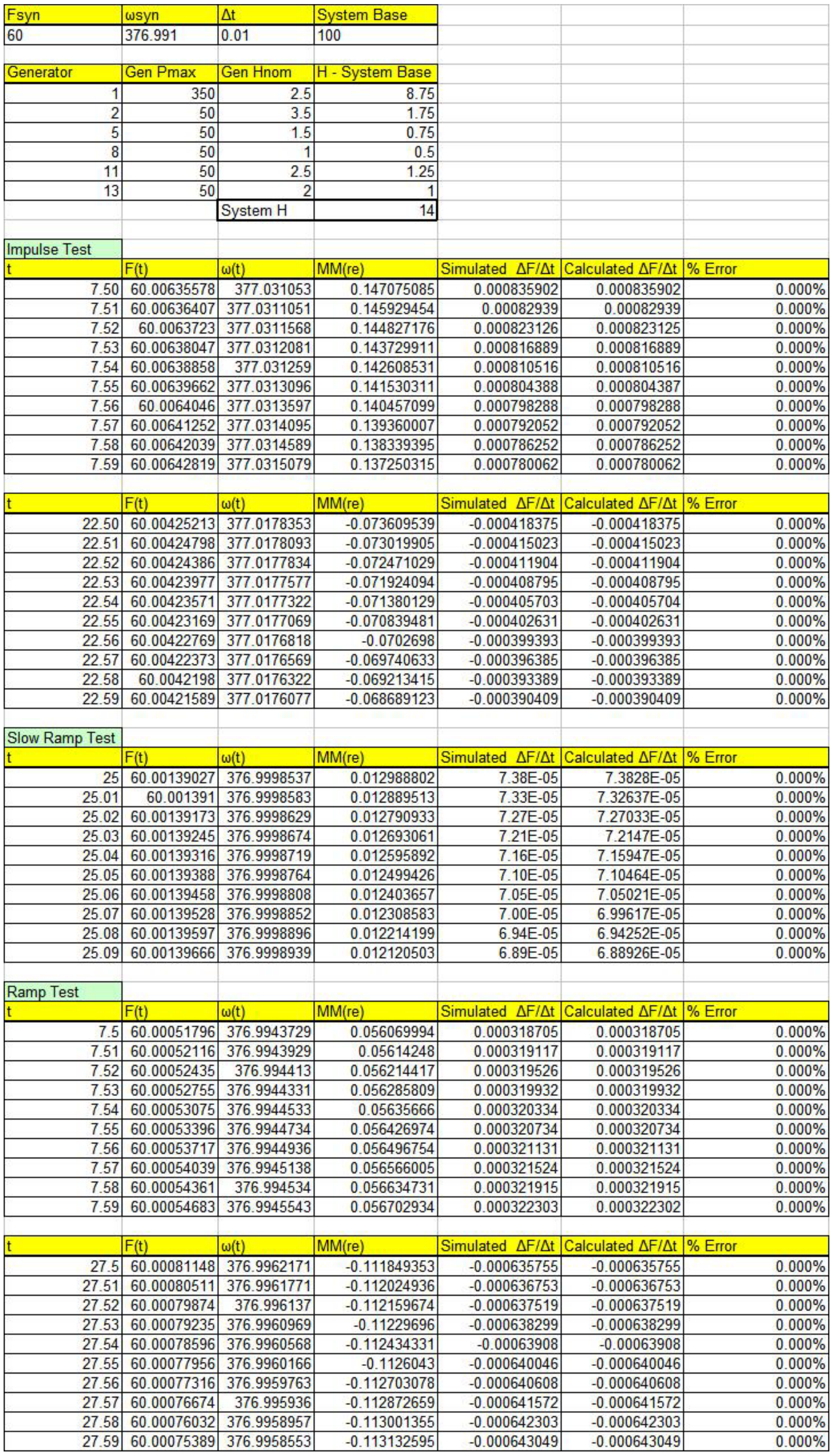
t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
25	60.00177653	377.0022807	0.020628903	9.86E-05	9.85914E-05	0.000%
25.01	60.00177751	377.0022869	0.020529346	9.81E-05	9.81156E-05	0.000%
25.02	60.00177849	377.002293	0.020430228	9.76E-05	9.76419E-05	0.000%
25.03	60.00177946	377.0022991	0.020331553	9.72E-05	9.71703E-05	0.000%
25.04	60.00178043	377.0023052	0.020233322	9.67E-05	9.67008E-05	0.000%
25.05	60.00178139	377.0023112	0.020135536	9.62E-05	9.62335E-05	0.000%
25.06	60.00178235	377.0023172	0.020038196	9.58E-05	9.57683E-05	0.000%
25.07	60.0017833	377.0023232	0.019941301	9.53E-05	9.53052E-05	0.000%
25.08	60.00178425	377.0023292	0.019844851	9.48E-05	9.48442E-05	0.000%
25.09	60.00178519	377.0023351	0.019748844	9.44E-05	9.43854E-05	0.000%

Ramp Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
7.5	60.00052183	376.9943972	0.073050252	0.000349135	0.000349135	0.000%
7.51	60.00052533	376.9944192	0.073169288	0.000349704	0.000349704	0.000%
7.52	60.00052883	376.9944412	0.073319348	0.000350421	0.000350421	0.000%
7.53	60.00053234	376.9944632	0.073471481	0.000351149	0.000351149	0.000%
7.54	60.00053586	376.9944853	0.07362111	0.000351864	0.000351864	0.000%
7.55	60.00053939	376.9945075	0.073769627	0.000352573	0.000352573	0.000%
7.56	60.00054292	376.9945297	0.073917005	0.000353278	0.000353278	0.000%
7.57	60.00054646	376.9945519	0.074063339	0.000353977	0.000353977	0.000%
7.58	60.00055001	376.9945742	0.074208753	0.000354672	0.000354672	0.000%
7.59	60.00055356	376.9945965	0.074353266	0.000355363	0.000355363	0.000%

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
27.5	60.00142248	377.0000562	-0.145648698	-0.000696101	-0.000696101	0.000%
27.51	60.00141551	377.0000123	-0.145978097	-0.000697675	-0.000697675	0.000%
27.52	60.00140852	376.9999684	-0.146239539	-0.000698925	-0.000698925	0.000%
27.53	60.00140151	376.9999244	-0.146540892	-0.000700365	-0.000700365	0.000%
27.54	60.0013945	376.9998803	-0.146805937	-0.000701632	-0.000701632	0.000%
27.55	60.00138747	376.9998361	-0.147108995	-0.00070308	-0.000703081	0.000%
27.56	60.00138042	376.9997919	-0.147433429	-0.000704631	-0.000704631	0.000%
27.57	60.00137336	376.9997475	-0.14769102	-0.000705862	-0.000705862	0.000%
27.58	60.00136629	376.9997031	-0.147986865	-0.000707276	-0.000707276	0.000%
27.59	60.0013592	376.9996586	-0.148246159	-0.000708516	-0.000708516	0.000%

30 Bus System Validation



Fsyn	ωsyn	Δt	System Base
60	376.991	0.01	100

Generator	Gen Pmax	Gen Hnom	H - System Base
1	350	2.5	8.75
2	50	3.5	1.75
5	50	1.5	0.75
8	50	1	0.5
11	50	2.5	1.25
13	50	2	1
		System H	14

Impulse Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
7.50	60.00635578	377.031053	0.147075085	0.000835902	0.000835902	0.000%
7.51	60.00636407	377.0311051	0.145929454	0.00082939	0.00082939	0.000%
7.52	60.0063723	377.0311568	0.144827176	0.000823126	0.000823125	0.000%
7.53	60.00638047	377.0312081	0.143729911	0.000816889	0.000816889	0.000%
7.54	60.00638858	377.031259	0.142608531	0.000810516	0.000810516	0.000%
7.55	60.00639662	377.0313096	0.141530311	0.000804388	0.000804387	0.000%
7.56	60.0064046	377.0313597	0.140457099	0.000798288	0.000798288	0.000%
7.57	60.00641252	377.0314095	0.139360007	0.000792052	0.000792052	0.000%
7.58	60.00642039	377.0314589	0.138339395	0.000786252	0.000786252	0.000%
7.59	60.00642819	377.0315079	0.137250315	0.000780062	0.000780062	0.000%

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
22.50	60.00425213	377.0178353	-0.073609539	-0.000418375	-0.000418375	0.000%
22.51	60.00424798	377.0178093	-0.073019905	-0.000415023	-0.000415023	0.000%
22.52	60.00424386	377.0177834	-0.072471029	-0.000411904	-0.000411904	0.000%
22.53	60.00423977	377.0177577	-0.071924094	-0.000408795	-0.000408795	0.000%
22.54	60.00423571	377.0177322	-0.071380129	-0.000405703	-0.000405704	0.000%
22.55	60.00423169	377.0177069	-0.070839481	-0.000402631	-0.000402631	0.000%
22.56	60.00422769	377.0176818	-0.0702698	-0.000399393	-0.000399393	0.000%
22.57	60.00422373	377.0176569	-0.069740633	-0.000396385	-0.000396385	0.000%
22.58	60.0042198	377.0176322	-0.069213415	-0.000393389	-0.000393389	0.000%
22.59	60.00421589	377.0176077	-0.068689123	-0.000390409	-0.000390409	0.000%

Slow Ramp Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
25	60.00139027	376.9998537	0.012988802	7.38E-05	7.3828E-05	0.000%
25.01	60.001391	376.9998583	0.012889513	7.33E-05	7.32637E-05	0.000%
25.02	60.00139173	376.9998629	0.012790933	7.27E-05	7.27033E-05	0.000%
25.03	60.00139245	376.9998674	0.012693061	7.21E-05	7.2147E-05	0.000%
25.04	60.00139316	376.9998719	0.012595892	7.16E-05	7.15947E-05	0.000%
25.05	60.00139388	376.9998764	0.012499426	7.10E-05	7.10464E-05	0.000%
25.06	60.00139458	376.9998808	0.012403657	7.05E-05	7.05021E-05	0.000%
25.07	60.00139528	376.9998852	0.012308583	7.00E-05	6.99617E-05	0.000%
25.08	60.00139597	376.9998896	0.012214199	6.94E-05	6.94252E-05	0.000%
25.09	60.00139666	376.9998939	0.012120503	6.89E-05	6.88926E-05	0.000%

Ramp Test

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
7.5	60.00051796	376.9943729	0.056069994	0.000318705	0.000318705	0.000%
7.51	60.00052116	376.9943929	0.05614248	0.000319117	0.000319117	0.000%
7.52	60.00052435	376.994413	0.056214417	0.000319526	0.000319526	0.000%
7.53	60.00052755	376.9944331	0.056285809	0.000319932	0.000319932	0.000%
7.54	60.00053075	376.9944533	0.05635666	0.000320334	0.000320334	0.000%
7.55	60.00053396	376.9944734	0.056426974	0.000320734	0.000320734	0.000%
7.56	60.00053717	376.9944936	0.056496754	0.000321131	0.000321131	0.000%
7.57	60.00054039	376.9945138	0.056566005	0.000321524	0.000321524	0.000%
7.58	60.00054361	376.994534	0.056634731	0.000321915	0.000321915	0.000%
7.59	60.00054683	376.9945543	0.056702934	0.000322303	0.000322302	0.000%

t	F(t)	ω(t)	MM(re)	Simulated ΔF/Δt	Calculated ΔF/Δt	% Error
27.5	60.00081148	376.9962171	-0.111849353	-0.000635755	-0.000635755	0.000%
27.51	60.00080511	376.9961771	-0.112024936	-0.000636753	-0.000636753	0.000%
27.52	60.00079874	376.996137	-0.112159674	-0.000637519	-0.000637519	0.000%
27.53	60.00079235	376.9960969	-0.11229696	-0.000638299	-0.000638299	0.000%
27.54	60.00078596	376.9960568	-0.112434331	-0.00063908	-0.00063908	0.000%
27.55	60.00077956	376.9960166	-0.1126043	-0.000640046	-0.000640046	0.000%
27.56	60.00077316	376.9959763	-0.112703078	-0.000640608	-0.000640608	0.000%
27.57	60.00076674	376.995936	-0.112872659	-0.000641572	-0.000641572	0.000%
27.58	60.00076032	376.9958957	-0.113001355	-0.000642303	-0.000642303	0.000%
27.59	60.00075389	376.9958553	-0.113132595	-0.000643049	-0.000643049	0.000%