COMPUTER SIMULATION OF POWERTRAIN COMPONENTS WITH METHODOLOGIES FOR GENERALIZED SYSTEM MODELING

by

SCOTT A. MUNNS

A thesis submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE
(MECHANICAL ENGINEERING)

at the
UNIVERSITY OF WISCONSIN-MADISON
1996

Acknowledgments

I would like to thank Professor John J. Moskwa, whose understanding of dynamic systems and controls has continually challenged me throughout my undergraduate and graduate careers. I also appreciate his willingness to support my research through the U. S. Army's Automotive Research Center initiative. Guy Babbitt, Richard Bonomo, Zack Rubin, James Wai, and the Powertrain Control Research Lab technicians have been extremely helpful and have managed to provide more than a few lighthearted moments which make the day go by much more quickly.

Co-workers have made significant contributions to the powertrain simulation described in this thesis. I thank Scott Chen for developing a cylinder-by-cylinder engine model in the SIMULINK environment. A modified version of this model is shown in Figures 2.4 - 2.8. Zack Rubin has played an integral part in the development of many drivetrain component models. The drivetrain system model was used to generate many of the simulation results in Chapters 3 and 4.

I thank my fiancee, Roxanne Smith, for her love and support as I put in many late night study sessions. I look forward to sharing a lifetime in the "real world" with her. I also thank my roommates, Carl Schinke and Dave Purvis, for their patience as I have been absent from our apartment (and negligent in my house-keeping duties) for an extended period. Thanks to the rest of the Nav crowd (especially Dave and Heidi Gering) for their friendship and support. Finally, I thank J. C., who has graciously given me meaningful relationships with these friends and co-workers.

Table of Contents

Page

List of Figures

List of Tables

Nomenclature

Symbol	Description	Units
α	Angular Acceleration	[rad/s^2]
γ	Ratio of Specific Heats	[ ]
η	Efficiency	[ ]
χ	Normalized Pressure Coordinate	[ ]
ω	Angular Velocity	[rad/s]
A	Area	[m^2]
c	Damping	[N-m/(rad/s)]
C_p	Constant Pressure Specific Heat	[J/(kg-K)]
f, g	General Functions	[ ]
F	Force	[N]
h	Enthalpy	[J/kg]
J	Inertia	[kg-m^2]
L_i	Length of Generalized Coordinate Vector i	[ ]
m	Mass	[kg]
$\dot{m}$	Mass Flow Rate	[kg/s]
M	Mach Number	[ ]
M_i	i^{th} Mode of Operation	[ ]
N	Speed	[rad/s or rpm]
N_s, N_r	Number of Gear Teeth on Planetary Sun and Ring	[ ]
P	Pressure	[Pa]
q_i	Generalized Coordinate Vector i	[ ]
$\Re$	Ratio of Gear Radii	[ ]
R	Gas Constant	[J/(kg-K)]
R_i	Radius of ith Gear	[m]

s	Entropy	[J/(kg-K)]
t	Torque	[N-m]
T	Temperature	[K]
$\mathbf{u}$	Vector of Generic Input Variables	[ ]
v	Velocity	[m/s]
V	Volume	[m^3]
$\dot{W}$	Power	[W]
$\mathbf{x}$	Vector of Generic State Variables	[ ]
$\mathbf{y}$	Vector of Generic Output Variables	[ ]

Subscript	Description
0	Stagnation or Total Thermodynamic Property
comp	Compressor
corr	Corrected
dep	Dependent Variable
ds	Driveshaft
gc	Generalized Coordinate
pc	Planet Carrier
r	Ring
s	Sun
std	Standard
turb	Turbine

Abstract

A computer simulation of a powertrain system was developed as a tool for design engineers in the automotive industry. Commercial dynamic system simulation software (SIMULINK) was utilized which had a graphical environment based on nonlinear state block diagrams. Methodologies were developed to deal with the complexities of modeling large systems and to provide a flexible, user-friendly simulation environment.

Two components, automatic transmissions and turbochargers, were investigated in detail due to challenges they provide to the modeler. The dynamic equations of automatic transmissions change with time, forcing the modeler to provide a framework for dealing with these changes. The turbocharger model requires iterative solutions to nonlinear algebraic equations at each time step to provide estimates of total pressures and temperatures. Simulation results are given for both components which demonstrate the potential usefulness of powertrain simulations.

Chapter 1: Introduction

1.1 Purpose of Research

In today's global economy, importance is placed on delivering products to market quickly. Fast product development cycles allow companies to adopt new technologies and to fulfill customers' desires before a competitor does so. Corporations have made adjustments in their methods to reduce cycle times, leading to concepts such as concurrent engineering and flexible manufacturing.

Product cycle times can be reduced significantly through the use of computer simulation tools by reducing costly, time-consuming hardware prototyping. Currently, most tools have a specific area of application such as fluid mechanics (CFD) or structural analysis (FEA). Many of these specific tools are already being used in the automotive industry; however, few tools are able to perform system-level simulation of vehicle and powertrain behavior. Since system-level, transient performance (acceleration, system response, handling, load carrying capability) are of interest to many customers, it is critical to create useful analytical tools to meet this need.

A properly designed powertrain simulation package would have several advantages. First, it would be possible to easily test different powertrain configurations. Second, different versions of a component within the same configuration could be tested for their effects on the overall powertrain behavior. Therefore, optimization and sensitivity analyses could be performed on the component and system levels. Third, many sources of experimental error would be eliminated because testing could be performed on an extremely repeatable basis, although it is recognized that simulation cannot entirely replace experimentation (because of modeling errors and simplifications). Finally, testing may be performed under conditions which would cause catastrophic failures. Weaknesses in the system can then be diagnosed and corrected.

A powertrain simulation was designed to meet the needs of vehicle designers, attempting to maximize the benefits of simulation mentioned above while creating a user-friendly environment. This thesis provides methodologies for designing large system simulations and presents methods for modeling specific powertrain components.

1.2 Overview of Thesis

Chapter 1 is an introductory chapter which indicates the purpose for the thesis research. More detailed introductions are included at the beginning of Chapters 2, 3, and 4 for readers with specific areas of interest.

In Chapter 2, methodologies for the modeling of large powertrain systems are introduced and explained. The methodologies are designed around the use of commercially available dynamic system simulation software with a graphical environment based on nonlinear state block diagrams. Emphasis is made on the creation of a simulation that is adaptable, portable, and easy to use. Issues covered include the concepts of hierarchy, flexibility, modularity, and causality. Initialization file structures and component-level pre-processors are also discussed.

The dynamic equations used to describe automatic transmissions and other hardware can change with time. Therefore, the modeler must take this into account and provide a means of switching the structure of the dynamic equations. Chapter 3 contains step-by-step methods for modeling automatic transmissions. A simple transmission is modeled first to provide additional insight into the method. Then, the same process is followed for a more complicated and realistic case, the Allison HT-740 4-speed automatic transmission. Simulation results for both transmissions are supplied and discussed.

Turbochargers are extremely useful devices for increasing the power density of diesel engines. Many diesel engines in commercial and military applications utilize turbochargers; therefore, turbochargers are crucial elements in a general powertrain system simulation. Chapter 4 discusses the modeling of turbochargers and provides two different algorithms for component models. The first algorithm is simple, but neglects the kinetic energy of the inlet and outlet gas streams. The second model incorporates the kinetic

energy effects of these streams through the iterative solution of nonlinear algebraic equations. Benchmark simulations are run to compare the two algorithms, and results of the simulations are discussed.

Chapter 5 provides a brief summary of findings, and makes suggestions for future work.

Chapter 2: Powertrain System Modeling

2.1 Introduction

In today's global economy, increased importance is placed on delivering products to market quickly. Fast product development cycles allow companies to keep abreast of new technologies and to be more receptive to customers' changing desires. Corporations have made many adjustments in their methods to reduce cycle times, leading to concepts such as concurrent engineering and flexible manufacturing.

If the number of hardware prototypes required before production begins is reduced, a significant source of delay and cost in the design process is eliminated. Therefore, corporations have placed increasing emphasis on the development and use of computer tools to achieve a minimal number of hardware prototypes. These tools typically use technologies such as solid modeling, finite element analysis (FEA), and computational fluid dynamics (CFD). The majority of these tools are used to predict detailed behavior of specific devices. Different tools must be used if the less detailed behavior of a complex, interconnected system is to be determined.

The automotive industry requires a tool for computer simulation of vehicle systems in order to reduce the required number of hardware prototypes. Knowledge of the dynamic behavior of a vehicle's mechanical, thermodynamic, and electrical components is essential for a successful vehicle design; therefore, the tools of choice must be able to simulate the time-varying, or *transient* behavior of each component and of the overall system. Fortunately, it is possible to mathematically describe the dynamic behavior of many devices in terms of nonlinear ordinary differential equations. Several computer simulation packages (SIMULINK, SystemBuild, EASY5, and others) are designed to solve systems of nonlinear ordinary differential equations and can be utilized to create a dynamic vehicle system simulation.

Before a vehicle system can be simulated, each component must be described in terms of nonlinear ordinary differential equations. Then, the components must be connected together in the desired configuration, and external inputs (accelerator position, road characteristics, etc.) must be specified. The system can be solved by a variety of numerical methods defined by the simulation program. Finally, the numerical results can be displayed in many different graphical formats, and data analysis can be performed.

Even though a simulation may correctly predict the dynamic response of a vehicle, poor design of that simulation could render the software virtually unusable by a design engineer, the intended customer. A design engineer could desire several attributes in a vehicle simulation:

- The ability to easily change system-level configurations (powertrain layouts, etc.) and predict the results of those changes.
- The ability to easily test different versions of a component within the same system configuration to compare effects on overall system performance.
- The ability to understand the simulation without having to know the details of each component model.
- The ability to readily view output data in a number of forms, and to limit the output data to only the desired parameters.
- The ability to add new component models if a new device is encountered or if a new modeling method is found for an existing device.
- The ability to run simulations on a personal computer with acceptable simulation times.
- The ability to easily change the fidelity, or accuracy, of a device model, knowing that a more accurate device model generally makes the system simulate more slowly.

The primary focus of this chapter is to illustrate the issues involved in designing a vehicular powertrain system simulation. Particular emphasis is placed on designing the simulation around the needs of the design engineer. First, dynamic system simulation software packages are described. Second, methods for creating user-friendly powertrain system dynamic models are given. Finally, the issue of causality and its effect on simulation

design is explored. All figures generated and terms defined in this chapter are based on SIMULINK dynamic system simulation software. However, the comments in this chapter also apply to other simulation programs.

2.2 Dynamic System Simulation Software

2.2.1 Dynamic Equation Formulation

As described in the introduction, it is necessary to create component models as systems of first-order, nonlinear ordinary differential equations (nonlinear state space form) [16]:

$$\{\dot{\mathbf{x}}\} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{2.1}$$

$$\{\mathbf{y}\} = \mathbf{g}(\mathbf{x}, \mathbf{u}) \tag{2.2}$$

where **{x}** is a vector of states, or integrated variables, **{u}** is a vector of input values, and **{y}** is the vector of output values. It is usually possible to reduce an order n ordinary differential equation to n first-order ordinary differential equations. Therefore, (2.1-2.2) provide a general mathematical framework about which to design powertrain component models.

2.2.2 Graphical Representation of Dynamic Equations

In most commercial dynamic system simulation packages, the nonlinear differential equations are represented in a graphical form which permits the user to see the structure of the differential equations. Two graphical representations are bond graphs [14] and nonlinear state block diagrams. Nonlinear state block diagrams are extensions of state diagrams and block diagrams [12], and are closely related to analog computer diagrams [2]. Since SIMULINK uses nonlinear state block diagrams to represent nonlinear differential equations, they are used throughout the chapter.

A nonlinear state block diagram representing a mass-spring-damper system with a nonlinear spring is shown in Figure 2.1. Blocks representing gains, summing junctions, integrators, and nonlinear functions are connected together by *signals*. As in state

diagrams, the outputs of the integrators are the states {**x**} [12]. This format is more clear than a standard block diagram containing transfer functions, since it is possible to represent transfer functions by an infinite number of combinations of states.

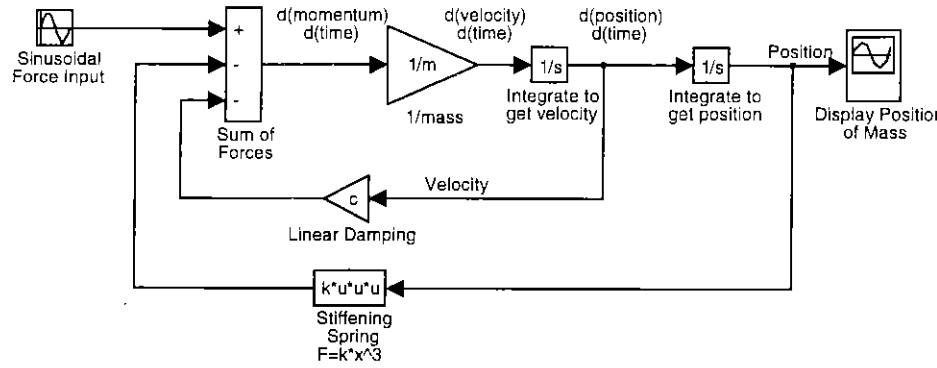

Figure 2.1: Nonlinear State Block Diagram of a Mass-Spring-Damper System

In addition to solving continuous, nonlinear systems of ordinary differential equations, many dynamic system simulation programs can also simulate discrete and mixed discrete-continous systems. This feature is useful for implementing digital controllers with continuous systems, but will not be discussed further.

Most blocks used in a simulation come from a standard library of basic blocks. These blocks usually include linear, nonlinear, discrete, and logical elements. Blocks are also provided for generating input signals (signal generator) or displaying results (oscilloscope).

It is obvious from Figure 2.1 that a large system may become a messy, undecipherable jumble of blocks and connections. Therefore, data structures have been designed to allow hierarchies of grouped blocks. Hierarchies reduce the number of blocks and signals seen at any one time, but allow all blocks to be seen and modified if desired. Hierarchies will be discussed further in Section 2.3.1.

2.2.3 Programming Language Representation of Dynamic Equations

While the nonlinear state block diagram is a convenient and easily understood representation of many dynamic systems, it is not always the best representation. Therefore, provision has been made in SIMULINK for the inclusion of code written in C or FORTRAN. The code is written in a specific format for compatibility with the simulation program, and includes several important functions. Some of these functions are:

- An initialization function to determine the number of inputs, outputs, and states in the dynamic system, and to obtain initial values for the states
- A function to compute the state derivatives in terms of the states and inputs, **f(x,u)**
- A function to compute the output vector in terms of the states and inputs, **g(x,u)**

The compiled code in the SIMULINK format is then inserted into a special "S-function" block so it can be included in the nonlinear state block diagram like any other standard or user-specified block [16]. An example of an S-function block representing a turbocharger compressor is shown in Figure 2.2.

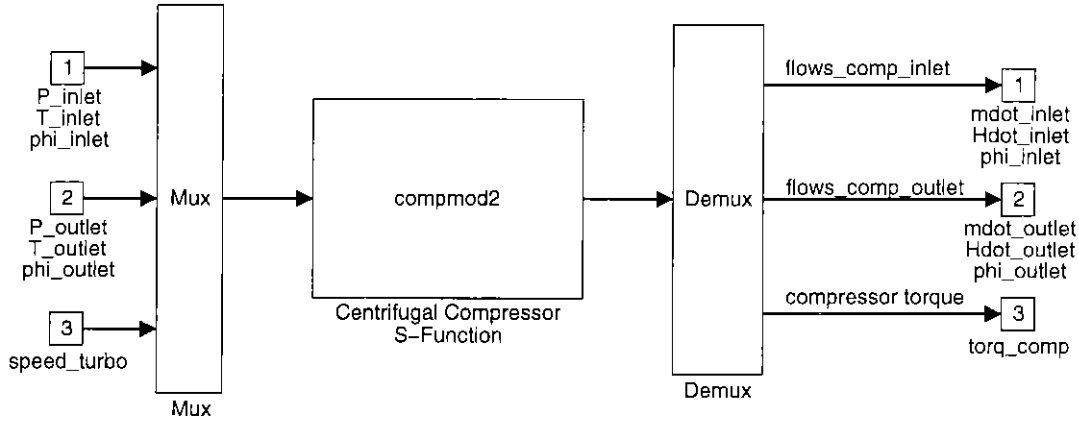

Figure 2.2: S-Function Block for a Turbocharger Compressor

2.2.4 Choosing an Appropriate Representation

Since two different representations of the dynamic equations are available, it is important to know when each representation should be used. There are no definitive rules for these decisions, but experience has shown that:

- Most dynamic systems are best represented by nonlinear state block diagrams
- Systems involving many logical decisions are often best represented by S-functions
- Systems which require an iterative (nonlinear algebraic) solution are best represented by S-functions

Many logical expressions (if-then-else constructs, etc.) may be implemented using logical blocks from the standard block library, although they are typically complicated and hard to understand at a first glance. On the other hand, in an S-function the high-level programming language directly states the conditional relationships. Therefore, simple logical expressions are often constructed with logical blocks, but S-functions are used for more complicated expressions.

Nonlinear algebraic solutions typically require initial guesses and several iterations at each integration time step. Since the simulation software is not set up to do this, the iteration should be accomplished internally by an S-function each time it is called by the integration algorithm. See Chapter 4 for examples of iterative turbocharger models implemented with S-functions.

Nevertheless, decisions to use S-functions must be made carefully, since S-functions must be recompiled on each target computer platform. Therefore, a compiler must be available on that platform that is compatible with SIMULINK. This significantly reduces the portability of the software. Additionally, nonlinear state block diagrams are generally much easier to read and understand than the code used to implement an S-function. Meticulous documentation of code must be maintained to aid in understanding.

2.2.5 Integration Routines

Most dynamic system simulation packages provide a choice of several different numerical integration algorithms. Each algorithm is ideally suited for a specific application. Runge-Kutta algorithms are useful for general nonlinear system analysis, while the Gear algorithm is best suited for stiff systems, or systems containing a mixture of fast and slow dynamics [16]. These algorithms have performed well for powertrain simulation. Both algorithms have variable step sizes and user-defined values for relative error tolerance, maximum step size, and minimum step size.

2.3 Construction of Large System Models

While small systems are very easy to simulate with SIMULINK or other programs, large systems demand careful planning at all stages of the simulation design process. In the following sections, guidelines and examples are given for the effective design of large powertrain system simulations.

2.3.1 Hierarchy

A large powertrain simulation can contain tens of component models and hundreds, if not thousands, of interconnected blocks. For manageability and ease of understanding, it is essential to be able to group these blocks into meaningful sets. Therefore, the concept of *hierarchy*, or nested groups of blocks, has been developed.

In SIMULINK, the basic unit of hierarchy is the *subgroup*. A subgroup may contain standard library components or other subgroups. Subgroups, like many other blocks, may have multiple inputs and outputs. Generally, subgroups are defined to encapsulate a portion of the physics of a device or system. However, subgroups may also be used simply to hide details of a complicated operation, reducing the visual complexity of the nonlinear state block diagram.

The best way to see hierarchy is through example. Figures 2.3-2.8 show several levels of hierarchy in a powertrain simulation. Often, five or more levels of hierarchy are necessary to move from the top level to standard library components.

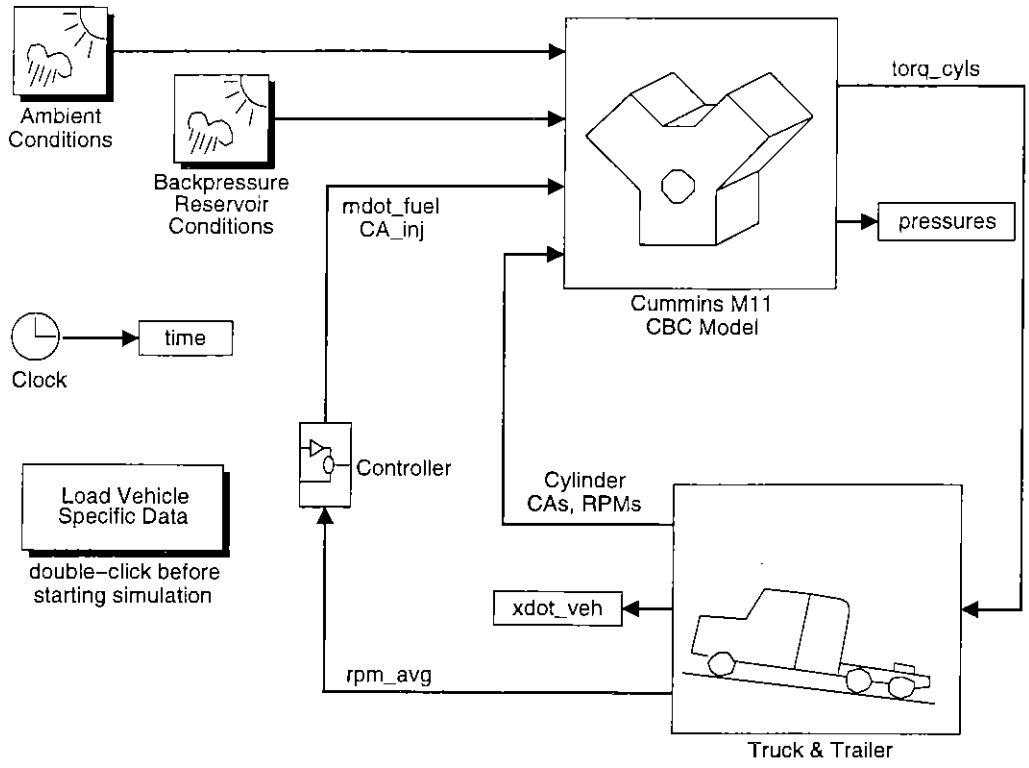

Figure 2.3: Top Level of a Powertrain Simulation

Notice that each level in the hierarchy holds specific types of component models, increasing in detail as one moves downward through the hierarchy. Most grouping decisions are based on a need to limit the number of blocks and signals in a subgroup for reduced complexity.

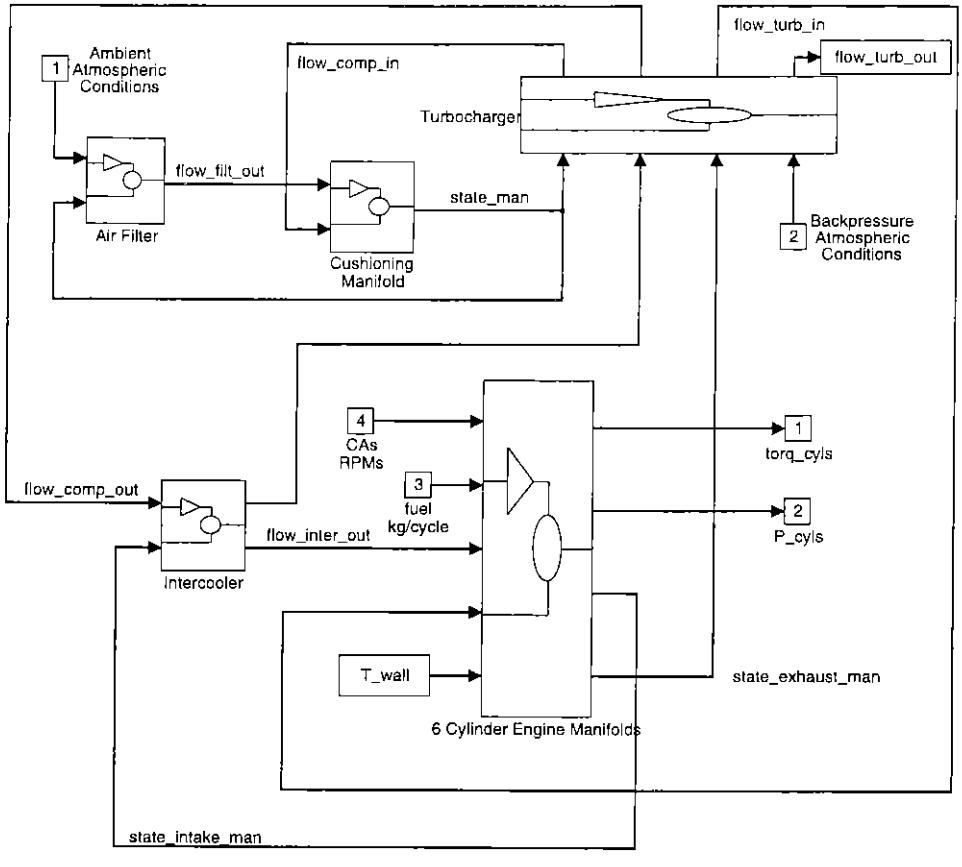

Figure 2.4: Engine Compartment Component Models

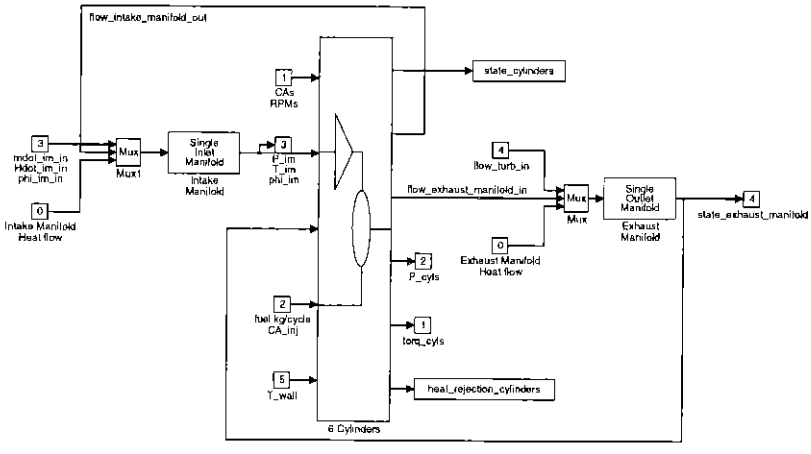

Figure 2.5: Engine Model with Manifolding

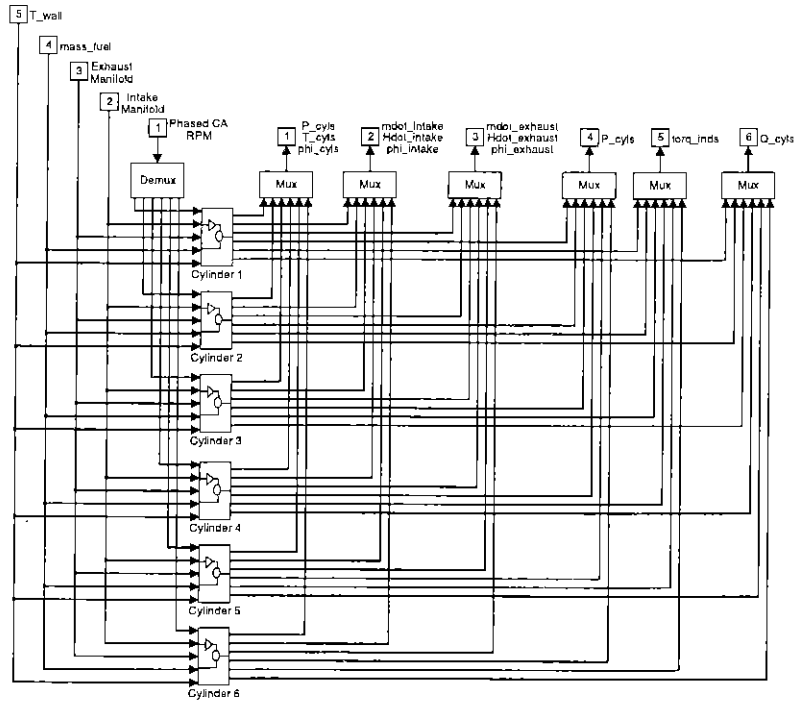

Figure 2.6: Six Cylinders of an Engine Model

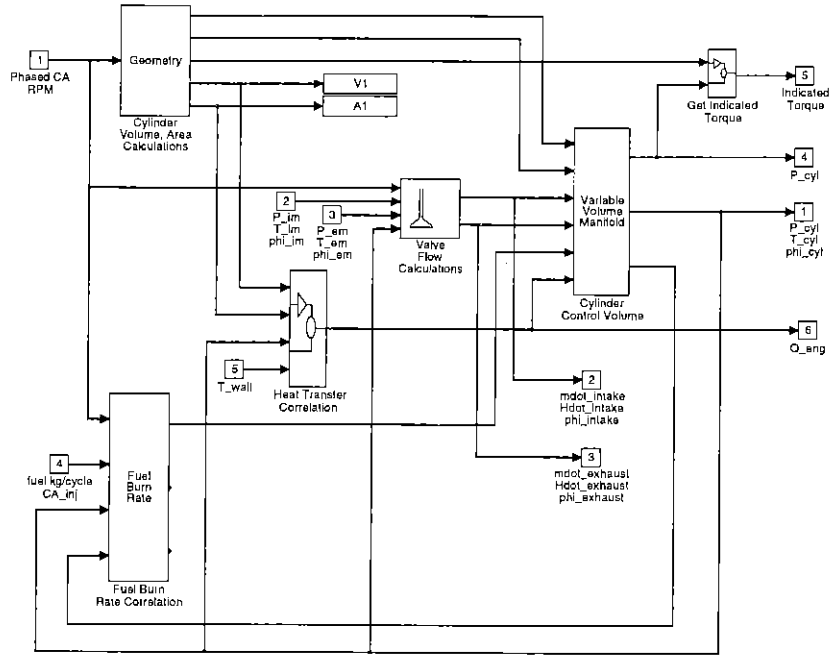

Figure 2.7: One Cylinder of an Engine Model

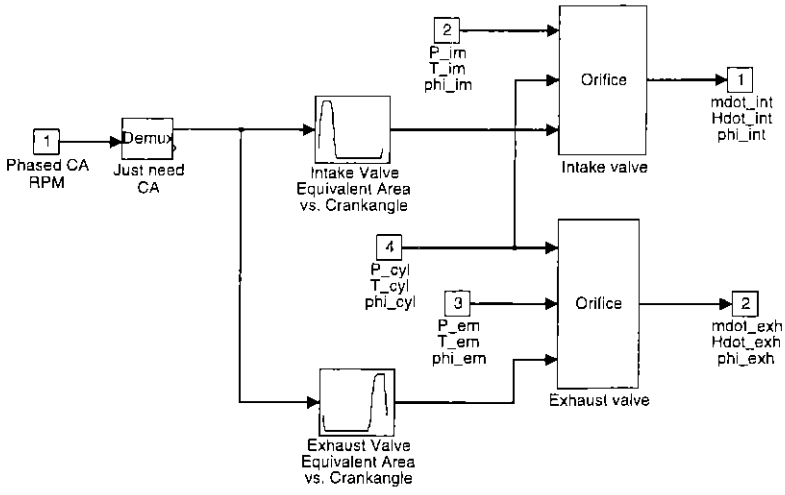

Figure 2.8: Intake and Exhaust Valves in a Cylinder Model

2.3.2 Flexibility

As mentioned in the introduction, an important feature for a design engineer is the ability to easily change powertrain configurations for testing of alternative layouts. This ability is directly related to the development of a standardized powertrain component model library. Figure 2.9 shows one such library.

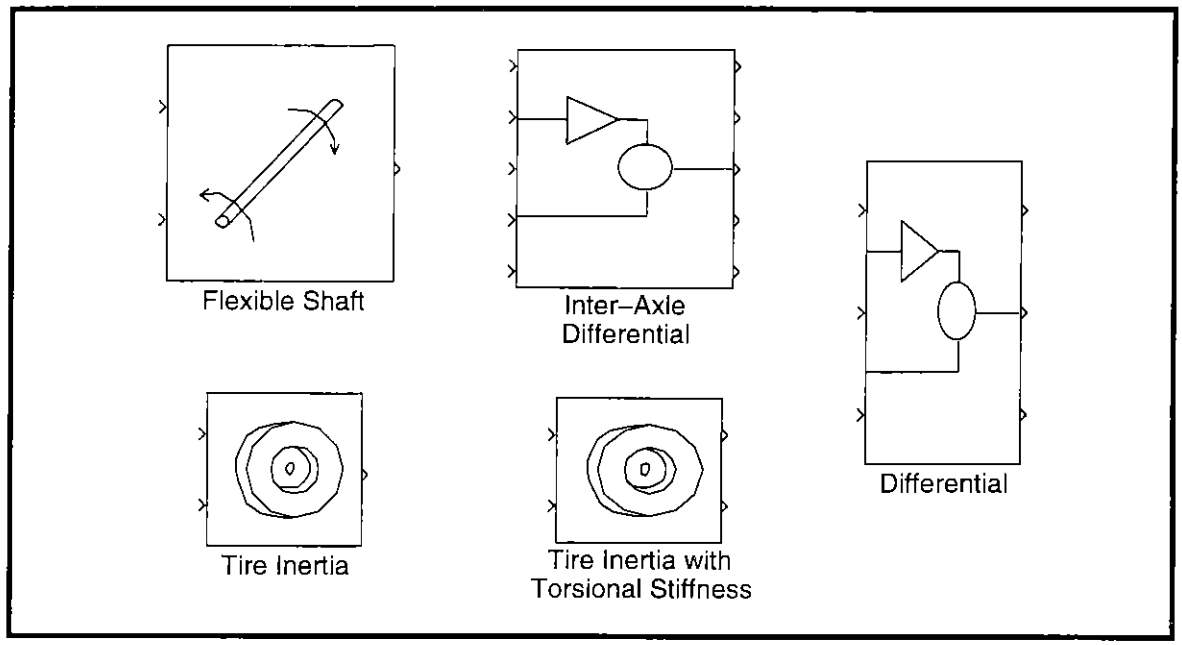

Figure 2.9: Powertrain Component Block Library

Several basic powertrain component models may be developed and placed into subgroups for inclusion in the block library. Blocks may be copied from the block library and placed into a new simulation whenever desired. However, the creation of standardized submodels does not guarantee the flexibility the design engineer requires. Several factors combine to assure success in this area.

The primary concern when developing standardized libraries is that the models correspond to real, intuitively understood components. Examples of such components include inertias, springs, orifices, and other physical-based hardware. If the submodels are not designed in this manner, extra effort is required by the design engineer to change configurations.

If the component models are not fully compatible with one another with respect to input and output variable choices, the models cannot be easily connected together in a proper cause-and-effect relationship. See Section 2.4 for a discussion of *causality* and its implications.

If a compatible set of standardized powertrain component models is utilized, it is a simple matter to alter many features of a powertrain layout. Figures 2.10-2.11 show two simple drivetrains utilizing the block library in Figure 2.9. Notice that the general formats of the drivetrain models are identical, even though one model has an additional axle. In addition, the wise choice of standardized component models allows the graphical model representation to resemble the actual drivetrain. This feature, although aesthetic in nature, makes the simulation easier to understand.

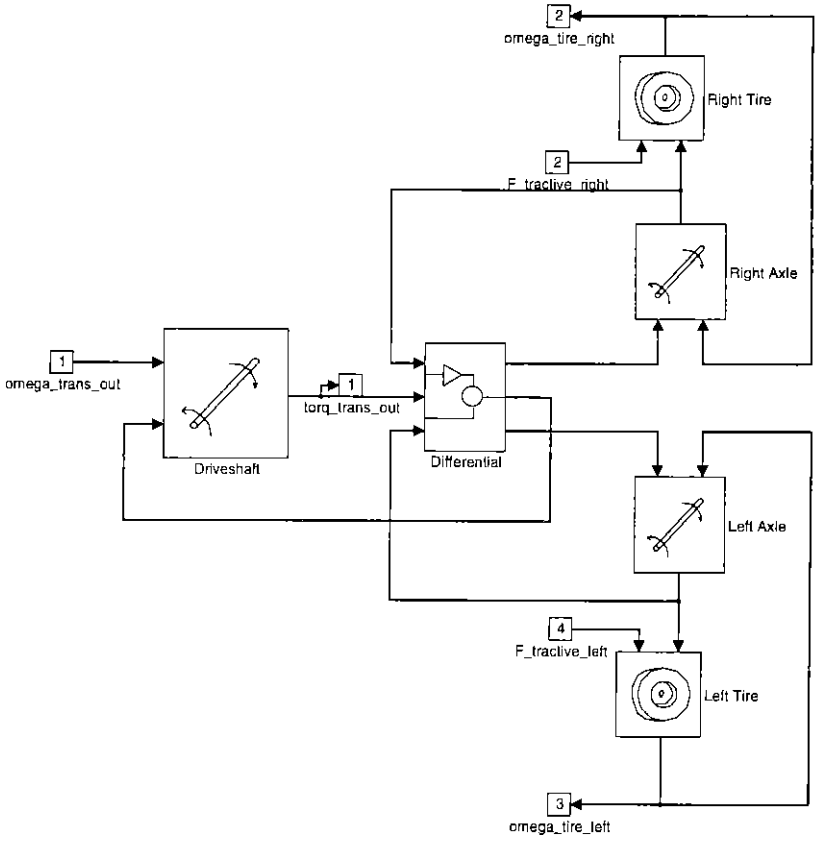

Figure 2.10: Simple Drivetrain Model

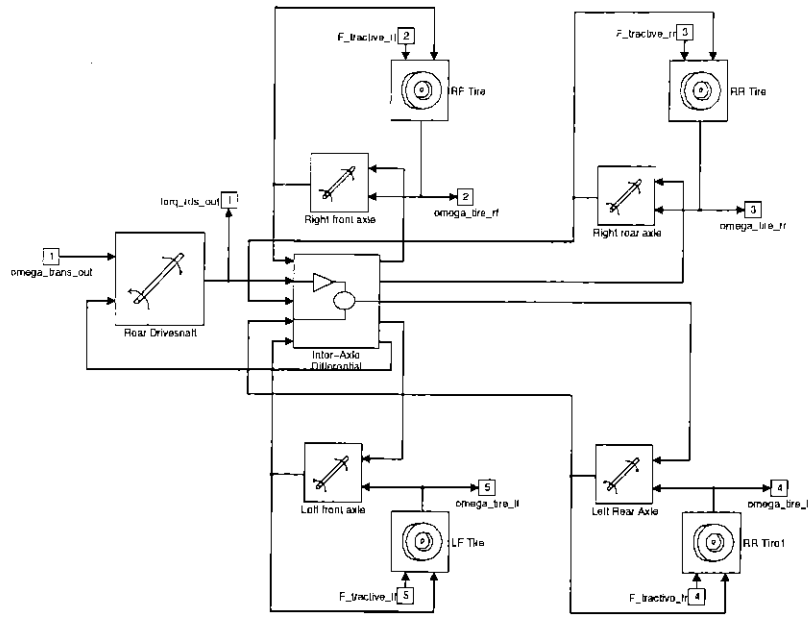

Figure 2.11: More Complicated Drivetrain Model

2.3.3 Modularity

Frequently in powertrain simulation design, it is desirable to have a choice between several different models of a given component. Each model may have specific advantages or disadvantages with respect to accuracy, simulation speed, and other criteria. The goal of modularity is to allow these different component models to be used interchangeably.

The modularity of a simulation depends upon the careful choice of inputs and outputs for given blocks. If the input and output arguments of two component models match exactly, the two models may be interchanged with no alterations to the rest of the system. More commonly, some small alterations have to be made; of course, the extent of these modifications should be minimized.

Ideal choices for inputs and outputs are very intuitive in nature, and can often be seen by placing a "black box" around the component in question so that its inner workings are ignored. For instance, internal combustion engines transform chemical energy of a fuel into work at an output shaft by burning the fuel in the presence of air. Therefore, atmospheric conditions and fueling rate are necessary inputs to an engine model. Work (or

torque) is an output of an engine model. Since an engine is not perfectly efficient, the burning of the fuel heats up the exhaust gas and rejects heat to the atmosphere. Thus, additional outputs include the thermodynamic state of the exhaust gas, its flow rate, and heat rejection to the outside world.

In a general example such as this, many details are overlooked. For example, does each cylinder of the engine have its own fuel flow rate (common in modern vehicles), or is the engine given one gross mass flow rate? These details must be examined one at a time, and conclusions must be reached. This becomes especially important when different teams are responsible for developing various versions of a component model; close communication between the teams is essential if a modular simulation is desired.

2.3.4 Initialization File Structure

Many blocks require the user to specify information before simulation begins. For example, a gain block needs a numerical value for its gain. Each integrator in the simulation also needs an initial value. In a large simulation, hundreds of *parameters* or *arguments* are required. In SIMULINK, which is a part of the MATLAB numerical computation program, all parameter values may be specified by MATLAB workspace variables. This flexibility allows parameters to be stored in MATLAB script text files or binary files. Script file name and parameter names should be very descriptive to aid understanding and to avoid duplication of variable names. Each parameter should be carefully documented, and units should be provided.The hierarchical nature of SIMULINK simulations suggests that a similar hierarchical structure be used to store parameter values. Such a structure is shown in Figure 2.12. In this structure, a master MATLAB initialization script runs other scripts containing the necessary parameters. Each of the second-level scripts may load other scripts, or more frequently, binary data (MAT) files.

2.3.5 Device Pre-Processors

Many component models, such as automatic transmission models, require over one hundred parameters to operate correctly. These parameters are often not intuitively understood by design engineers, since they are dependent on the exact type of model used.

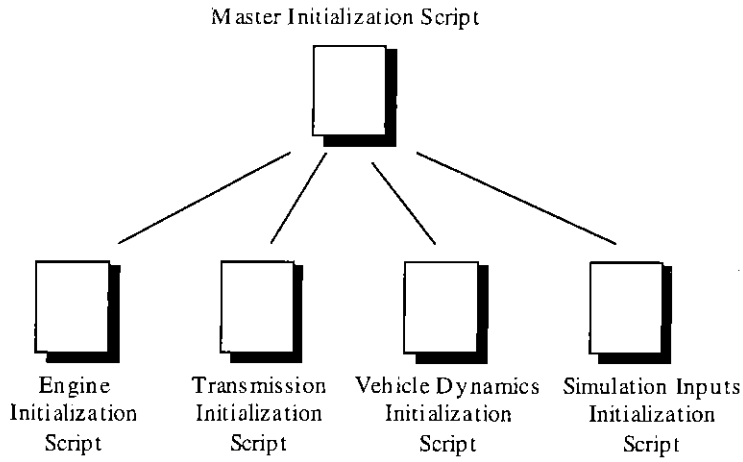

Figure 2.12: Hierarchical Initialization File Structure

However, in many cases, the parameters are functions of a few, intuitively understood design variables. For instance, in an automatic transmission simulation the design variables may include planetary gearset speed ratios, inertias of various internal parts, and clutch torque capacities.

Therefore, it is helpful if pre-processors are utilized to solve for required parameters in terms of the design variables. MATLAB functions and scripts may be written to perform the mathematical computations. These functions take in the design variable values given by the engineer, compute the necessary parameter values, and save the results to a binary data file. Since the parameters are saved, it is only necessary to run the pre-processor when a change in the design variables is desired. Then, when initialization files are run prior to a simulation, the binary data file is loaded into memory by an initialization script.

The use of pre-processors adds to the hierarchical initialization file structure of Figure 2.12. Now, some individual components have their own specialized pre-processors, each saving their data to binary data files. The extended hierarchical file structure is shown in Figure 2.13. Note that the pre-processors do not have to be written in MATLAB. Any programming language or commercially available engineering analysis package can be used as long as the software can output parameters into a data file compatible with MATLAB.

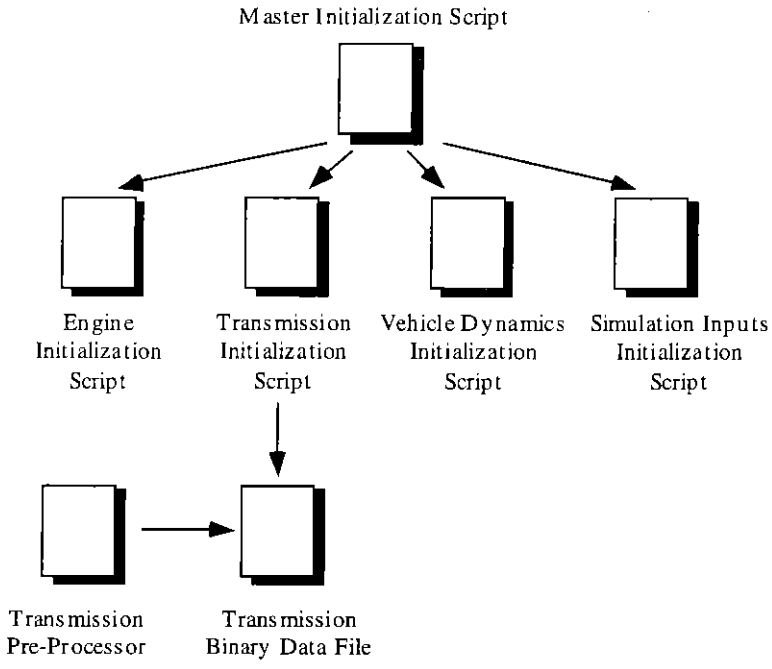

Figure 2.13: Extended Initialization File Hierarchy

One important feature of MATLAB is its ability to implement graphics features such as windows, menus, and pushbuttons. Therefore, it is possible to design Graphical User Interfaces (GUIs) for pre-processors. In a GUI pre-processor, the design engineer interacts with an environment very similar to most window-based operating systems. This is much more user-friendly to the engineer than the standard, MATLAB command-line functional syntax for a pre-processor. An example of a GUI pre-processor for an automatic transmission model is shown in Figure 2.14.

Unfortunately, GUI pre-processors require a substantial amount of effort to program. Therefore, their use should be limited to component models that may require many changes and have a complicated command-line functional syntax.

2.4 Causality

Causality, or the cause-and-effect relationship between component models, has a major influence on the layout of a powertrain simulation. Flexibility and modularity are important concepts, but a system with poor causality will simulate either slowly or not at

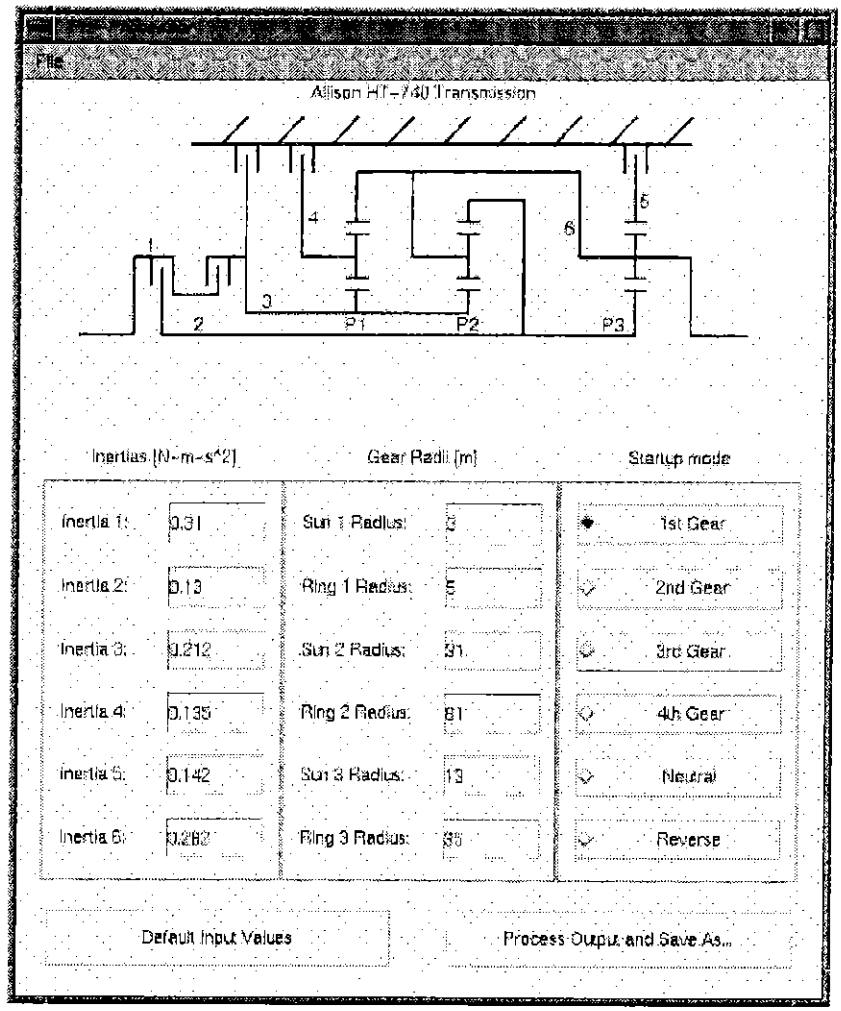

Figure 2.14: GUI Pre-Processor for an Automatic Transmission

all. Therefore, every decision regarding flexibility and modularity must be made taking causality considerations into account. In the following sections, methods will be outlined for designing simulations with proper causality.

2.4.1 Different Methods of Causality Assignment

Causality is an assigned characteristic of models within a system; the modeler (or the simulation software) must choose the causality before a system can be simulated. With bond graph methods, causality is assigned after the system model is constructed [14]. This is a significant advantage of simulation software based on bond graph system representations. However, other disadvantages led to the choice of a simulation program

based on nonlinear state block diagrams. With nonlinear state block diagrams, the cause-and-effect relationships between blocks are denoted by the directions of the signals between the blocks. Thus, causality is assigned as each individual block or subgroup is added to the system model. Since causality choices are left up to the simulation designer when using nonlinear state block diagrams, understanding the consequences of those choices is crucial, or descriptive flags should be added to the simulation to help the user.

2.4.2 Existence of Algebraic Loops

Causality is closely related to the order in which the simulation program solves equations. The software determines the equation update order through information contained in the state block diagram. This information includes the signal directions and knowledge about the *direct feedthrough* property of each block [16]. In a block with direct feedthrough, output values at a given time depend functionally on input values at the same time. Blocks with direct feedthrough include gain blocks and summing junctions. If a series of blocks with direct feedthrough are connected together from left to right, solution starts with a known value at the left and propagates to the right.

Not all blocks contain direct feedthrough. Examples of blocks without direct feedthrough include integrators and time delays. The output values of these elements at a given time do not depend on the input values at the same time; rather, they depend on input values from previous time steps. At a given time step, blocks without direct feedthrough function as known values and interrupt the feedthrough path. The values of these blocks are updated after the blocks with direct feedthrough have been solved.

Figure 2.15 shows a nonlinear state block diagram with a loop containing only direct feedthrough elements. Since there are no appropriate elements to interrupt the feedthrough path, the output of the left summing junction depends on its own value in an algebraic manner. Thus, an *algebraic loop* has been formed where iterations are needed at each time step to solve the loop. The simulator is capable of recognizing algebraic loops but is not always capable of solving them. Because the algebraic loops may be nonlinear, a Newton-Raphson algorithm is used to obtain a solution. Due to the number of iterations required

for a solution, simulation speeds often decrease markedly. This problem is exacerbated by the fact that the simulator does not iterate only on the elements in the loop; rather, it iterates on the whole model structure. While this does not have much of an effect on relatively small models, it is catastrophic for large powertrain models. Therefore, other means must be found for solving algebraic loops.

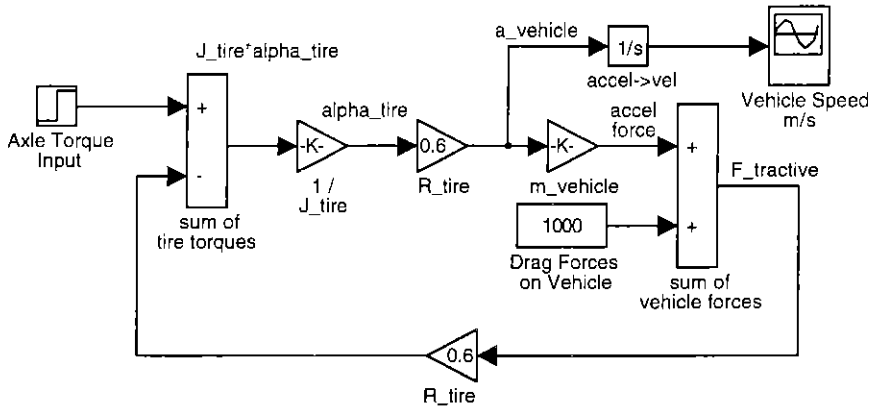

Figure 2.15: Algebraic Loop in a Simple Vehicle Model

2.4.3 Methods for Removing Algebraic Loops

Algebraic systems of equations described by algebraic loops may be solved several different ways. In this section, two methods are presented for linear systems and two methods are given for nonlinear systems. All of these methods involve replacing the algebraic loop with equivalent blocks having no such loops.

Linear Systems

A linear system of equations can be placed in the form:

$$[\mathbf{A}]\{\mathbf{y}\} = [\mathbf{B}]\{\mathbf{u}\} \tag{2.3}$$

where **[A]** and **[B]** are matrices, **{u}** is an input vector, and **{y}** is a solution vector. Provided that **[A]** is invertible, it is possible to solve for **{y}** as:

$$\{\mathbf{y}\} = [\mathbf{A}]^{-1}[\mathbf{B}]\{\mathbf{u}\} = [\mathbf{C}]\{\mathbf{u}\} \tag{2.4}$$

This leads to two potential solutions, depending on whether or not the **[A]** or **[B]** matrix coefficients are time invariant. If the coefficients are time invariant, it is possible to solve **[C]** numerically in a pre-processor prior to the simulation run. A typical solution method is Gaussian elimination with partial pivoting and back substitution. The **[C]** matrix can be placed in a "matrix gain" block for proper simulation with no algebraic loops. Figure 2.16 implements this solution for the algebraic loop of Figure 2.15.

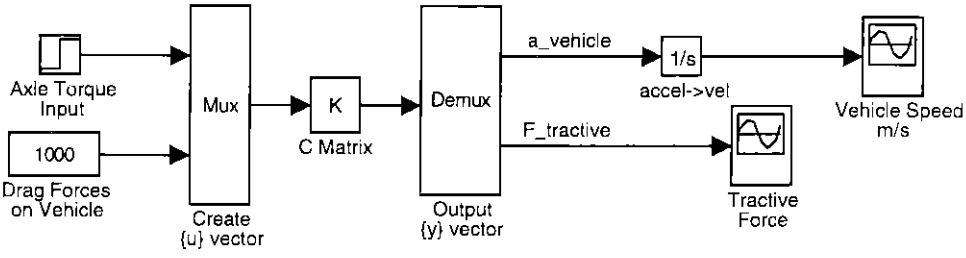

Figure 2.16: Removal of a Linear Algebraic Loop

If the **[A]** or **[B]** matrices contain time-varying coefficients, the coefficients must be re-computed at each time step and the solution must be updated. Because of the complex nature of this algorithm, an S-function block must be used to implement it. The resulting increase in simulation speed more than compensates for the complexity of the S-function.

Nonlinear Systems

If it is necessary to eliminate a nonlinear algebraic loop, two options are available. First, it may be possible to obtain an analytical solution for the outputs **{y}** in terms of the inputs **{u}**. If an analytical solution is unavailable, it is necessary to program an iterative solution of the loop in an S-function block. Many iterative solution algorithms are readily available in numerical algorithm libraries; examples include Newton-Raphson and Steepest Descent. Two ad hoc approaches for the iterative solution of compressor and turbine thermodynamics are given in Chapter 4.

2.4.4 Integral and Derivative Causality

The difference between integral and derivative causality is easily shown through example. Suppose there is a mass m sliding on a frictionless surface under the influence of external forces F. The velocity of the mass is denoted as v. Newton's equation of motion for the mass can be written in two different ways:

$$\frac{d}{dt}(m \cdot v) = \sum F \tag{2.5}$$

$$m \cdot v = \int F\,dt + C \tag{2.6}$$

Equation 2.5 corresponds to derivative causality, and implies that forces can be calculated by taking the derivative of the momentum. Equation 2.6 corresponds to integral causality, and implies that momentum is calculated by integrating the applied forces.

Simulation programs are generally designed to work with integral causality, and many robust, accurate algorithms are available for numerical integration. It is possible to take numerical derivatives of signals. However, results are extremely inaccurate if large time steps are used, and it is not possible to obtain an instantaneous derivative of a function. Therefore, derivative causality should be avoided [16].

2.4.5 Causality in Mechanical Systems

It is possible to predict the existence of algebraic loops before a model is implemented in nonlinear state block diagram format. In this section, examples from drivetrain mechanical systems will be given to explain a nonlinear state block diagram structure that contains no algebraic loops.

A typical model of an automotive drivetrain was shown previously in Figure 2.10. This model was obtained by the application of a "black box" approach, shown in Figure 2.17.

An interesting feature of Figure 2.17 is that there is an alternating structure of two different block types:

- Inertial blocks, which accept forces as inputs and pass speeds as outputs

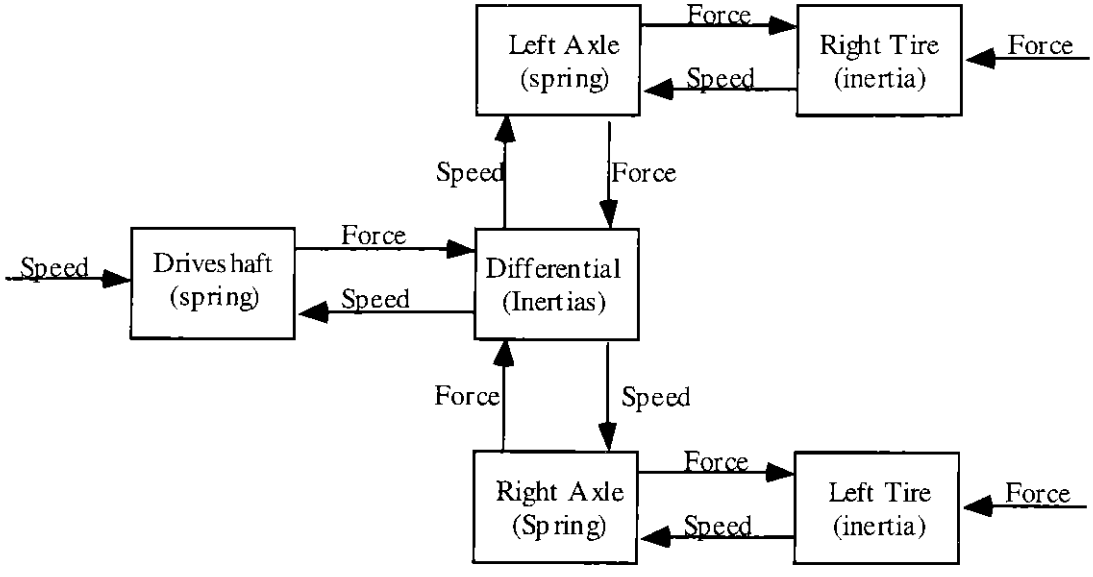

Figure 2.17: Black Box Version of a Drivetrain Model

- Compliant or resistive blocks, which accept speeds as inputs and pass forces as outputs

If such an alternating structure is used by the modeler, it can be shown that all of the devices in the system have integral causality as desired. This is because connections have not been made between two blocks of the same type. The addition of rigid connections reduces the number of degrees of freedom in the system, forces some devices to have derivative causality, and creates structures with algebraic loops [14].

It is often desirable to have rigid connections between inertial components. Then it is necessary to use only as many integrators as the number of degrees of freedom in the system, and the dependent speeds are solved algebraically in terms of the independent speeds (generalized coordinates). This leads to the use of the methods in Sections 2.4.3 to "hide" the algebraic loops from the simulation program.

2.4.6 Causality in Thermodynamic Systems

Although it is not as intuitively clear as with mechanical systems, a similar argument can be made for causality in thermodynamic systems. Consider the engine intake system shown in Figure 2.4. A portion of this system can be written in the "black box" form of Figure 2.18. In this figure, an alternating order of two different types of blocks is seen:

- Control volume blocks, which accept mass flow and energy flow as inputs and integrate to calculate the thermodynamic properties (state) of the control volume as outputs
- Flow control blocks, which accept thermodynamic properties (state) as inputs and calculate mass flow and energy flow as outputs

As in mechanical systems, if blocks of the same type are connected together directly, an algebraic loop is formed. Therefore, this practice should be avoided. However, if an algebraic loop cannot be avoided, the loop should be "hidden" by use of the methods in Sections 2.4.3.

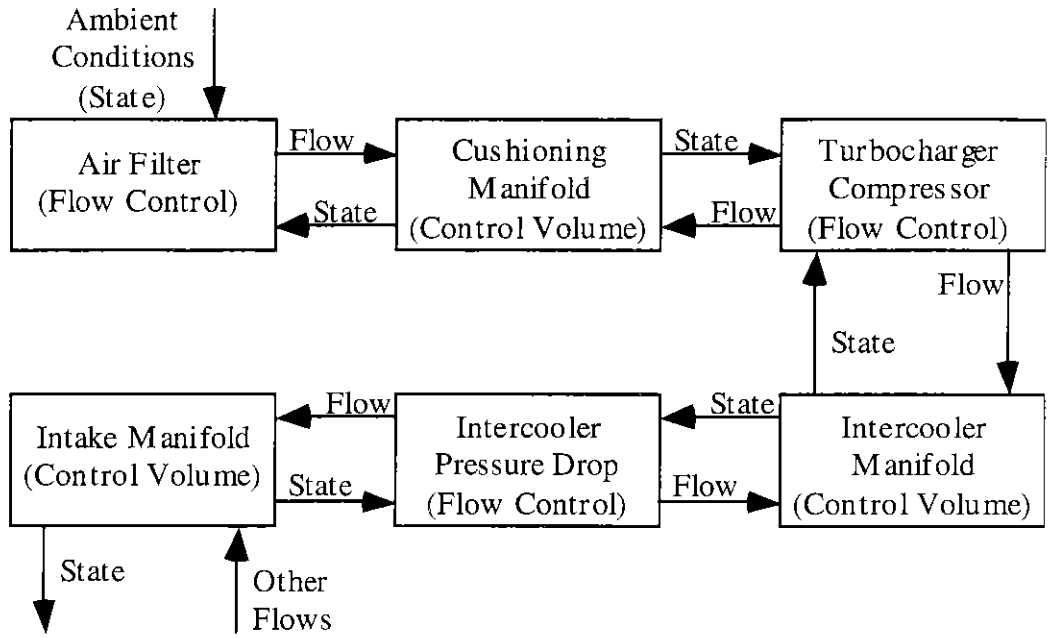

Figure 2.18: Black Box Structure for a Thermodynamic System

Chapter 3: Automatic Transmission Modeling

3.1 Introduction

In powertrain system modeling, the dynamic equations of motion frequently change during the course of a simulation. Therefore, a modeling method must be used which is able to change the system dynamics.

An automatic transmission is a specific example of a powertrain component which has changing dynamics. As the transmission is shifted from one gear to another, different components inside the transmission are connected together or disconnected. Due to the changing configuration, the equations of motion for the transmission change. The automatic transmission also provides an additional complication; the number of degrees of freedom of the mechanism often changes during a simulation. Thus, a successful modeling method must provide a means for dealing with varying dynamic equations and varying numbers of degrees of freedom.

In this chapter, a consistent method for dealing with these challenges is developed. First, background and theory are presented for simulation of time-varying dynamic systems. Next, a simple "automatic" transmission is simulated. Using this example as a reference, terminology for discussing automatic transmissions is explained, and the modeling method is described in a step-by-step manner. Then, the method is applied to the Allison HT-740 automatic transmission, used in the United States Army's M915A2 and M916A1 tactical vehicles. Finally, simulation results from a complete powertrain model including the Allison HT-740 transmission are presented.

3.2 Modeling Systems with Varying Dynamics

3.2.1 Prior Art

Before the development of the current modeling method, a literature search was performed to discover how other modelers dealt with systems which had time-varying dynamics. Several researchers have discussed modeling methods for devices exhibiting slip-stick

friction phenomena. Karnopp [13] maintained a state for each independent inertia, and algebraically solved for the frictional force necessary to keep the inertias coupled if the friction interface was in a sticking region. Logic was used to determine which set of algebraic equations should be used. In Karnopp's method, the logical complexity goes up rapidly as the number of frictional interfaces in the system increases. Haessig et al. [7] also maintained one state for each independent inertia, but frictional interfaces acted as dampers when slipping and as stiff springs when locked. The frictional interface model contained logic to determine whether the clutch should act as a damper or as a spring. Using Haessig's method, the logical complexity does not increase with the number of friction interfaces; if a friction interface model is maintained as a submodel in a library, the friction interface code is merely replicated for each separate friction interface. This feature is extremely advantageous when attempting to create a modular simulation.

In the papers by Karnopp and Haessig, the frictional interfaces had constant normal forces. Carey [5] extended their methods for use in manual transmission models, where clutches act as friction interfaces with varying normal forces. Additional logic was used to deal with potential problems when clutches transitioned from sticking to slipping or vice versa.

Runde [20], Pan and Moskwa [18], and Jamzadeh et al. [8] modeled automatic transmissions with different equations of motion for each realistic combination of locked and unlocked clutches that would be encountered. Logic was used to determine which set of equations should be used at each time step. Logical complexity in this method increases rapidly as the number of frictional interfaces and interconnected inertias increases.

3.2.2 Comparative Analysis of Methods

After considering the work of these researchers, the methods of Runde and Haessig were compared in a benchmark simulation using the transmission mechanism of Figure 3.1. Runde's method contained complicated logic, but simulated relatively quickly and accurately. Haessig's method had several advantages with respect to modularity and simplicity, but suffered because a set of stiff differential equations was created whenever clutches were locked. Thus, Haessig's method integrated more slowly than Runde's.

Haessig's method also exhibited an unrealistic ringing phenomenon at the shift points. From the results, it was decided to implement Runde's method for automatic transmission modeling while searching for ways to reduce its logical complexity.

3.2.3 Selection of Generalized Coordinates

Many different combinations of locked and unlocked clutches are possible in an automatic transmission. These combinations will now be called *modes* or *modes of operation*. Each mode has its own distinct set of dynamic equations. It is important to determine exactly which modes must be considered in the simulation. If a potential mode is unlikely to occur in practice or is unnecessary for the analysis, it should be discarded. Choosing too many modes of operation has the following negative effects:

- Logic necessary for switching between modes increases in complexity.
- A poorly chosen mode may have more degrees of freedom than the other modes being modeled. This will slow down the solution process for every mode of operation because extra integrators must be allocated by the simulation program for the extra degrees of freedom.

For a set of desired modes $\{M_1, M_2, \ldots, M_n\}$, there is a corresponding set of vectors of generalized coordinates $\{q_1, q_2, \ldots, q_n\}$. The lengths of the vectors are denoted by $\{L_1, L_2, \ldots, L_n\}$, and the modes are ordered by increasing degrees of freedom. Thus, M_n has the most degrees of freedom of $\{M_1, M_2, \ldots, M_n\}$. Then, if possible, q_n should be chosen such that $\{q_1, q_2, \ldots, q_{n-1}\}$ are all subsets of q_n. If this condition is met, it is guaranteed that no more integrators than L_n will be required to solve for the link speeds in terms of applied torques for every mode (the modeling objective). If an arbitrary mode M_a has fewer degrees of freedom than M_n, L_a integrators are necessary to model M_a. The remaining $(L_n - L_a)$ integrators not contained in q_a are linearly dependent upon q_a.

3.2.4 Determining Dynamic Equations of Motion

The dynamic equations of motion for the automatic transmission can be solved using methods based on bond graph theory [20] or free-body diagrams [18], among others. To simplify the solution process, the following assumptions are often made:

- All links, or rotating members in the transmission, are rigid.
- All links have only one degree of freedom.
- Planet gears have negligible inertia and inertial forces.
- Gears exhibit no backlash, and bearings have no play.
- Friction effects are negligible.

Some of these restrictions may be relaxed, but the complexity of the problem increases. Adding planet gear inertial effects significantly complicates the dynamic equations. Making certain links flexible in torsion is less difficult, but tends to increase the required number of degrees of freedom. High natural frequencies may also be created, which will significantly slow the simulation speed. Therefore, the addition of flexible links is not suggested unless it is believed that the assumption of rigid members will cause significant errors in the results.

To be compatible with simulation programs, it is necessary to cast the system dynamics as a set of first-order nonlinear differential equations [16, 21]:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{3.1}$$

$$\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{u}) \tag{3.2}$$

The inertial dynamics of an automatic transmission are linear in nature. Thus, equations (3.1-3.2) can be simplified to:

$$\{\dot{\mathbf{x}}\} = [\mathbf{A}]\{\mathbf{x}\} + [\mathbf{B}]\{\mathbf{u}\} \tag{3.3}$$

$$\{\mathbf{y}\} = [\mathbf{C}]\{\mathbf{x}\} + [\mathbf{D}]\{\mathbf{u}\} \tag{3.4}$$

where **[A]**, **[B]**, **[C]**, and **[D]** are matrices, **{x}** is the state vector, **{u}** is the input vector, and **{y}** is the output vector. The relationship between states and generalized coordinates is:

$$\{\mathbf{x}\} = \{\dot{\mathbf{q}}\} \tag{3.5}$$

If all the assumptions listed above are used, the transmission mechanism is an *inertial field* [14], where **{x}** are generalized coordinate rotational velocities, **{u}** are torques externally applied to the inertias, and **{y}** are other link rotational velocities or torques transmitted through rigidly connected coupling devices, solved in terms of **{u}** and **{x}**:

$$\{\dot{\mathbf{x}}\} = [\mathbf{0}]\{\mathbf{x}\} + [\mathbf{B}]\{\mathbf{u}\} \tag{3.6}$$

$$\left\{\begin{array}{c}\mathbf{y}_{speeds}\\ \hline \mathbf{y}_{torques}\end{array}\right\} = \left[\begin{array}{c}\mathbf{C}_{speeds}\\ \hline \mathbf{0}\end{array}\right]\{\mathbf{x}\} + \left[\begin{array}{c}\mathbf{0}\\ \hline \mathbf{D}_{torques}\end{array}\right]\{\mathbf{u}\} \tag{3.7}$$

Notice that for a linear inertial field, the link accelerations are linear algebraic functions of the input variables. Also, **{y}** speed elements are only dependent on **{x}**, and **{y}** torque elements are only dependent on **{u}**. Therefore, it is possible to solve algebraically for the elements of the matrices in (3.6-3.7) with dynamic, kinematic, and power conservation equations. Due to the complex nature of these equations, it is suggested that numerical (rather than symbolic) methods be used to solve these equations.

3.2.5 Numerical Solution of State-Space Matrices

The algebraic equations can be placed into the form:

$$[\mathbf{E}]\{\mathbf{z}\} + [\mathbf{F}]\{\mathbf{u}\} = \mathbf{0} \tag{3.8}$$

where **{z}** are the variables being solved for and **{u}** are the input variables to the state equations. Generally, **{z}** are link accelerations and torques transmitted through links. It is easily shown that:

$$\{\mathbf{z}\} = -[\mathbf{E}]^{-1}[\mathbf{F}]\{\mathbf{u}\} = [\mathbf{G}]\{\mathbf{u}\} \tag{3.9}$$

If **{z}** and **[G]** are partitioned such that:

$$\left\{\begin{array}{c}\alpha_{links}\\ \hline \mathbf{t}_{clutches,locked}\\ \hline \mathbf{t}_{intermediate}\end{array}\right\} = \left[\begin{array}{c}\mathbf{G}_{accel}\\ \hline \mathbf{G}_{clutches}\\ \hline \mathbf{G}_{intermediate}\end{array}\right]\{\mathbf{u}\} \tag{3.10}$$

it is possible to solve for the state-space matrices through the use of numerical methods such as Gaussian elimination with partial pivoting.

There will be a slightly different **[G]** matrix for each mode due to the inclusion of differing constraints on link accelerations. At this point, it is necessary to see if a given choice of generalized coordinates is realizable for each mode of operation. Fortunately, this is relatively simple to discover once the **[G]** matrices have been solved for each mode. First, it is necessary to test how many degrees of freedom are contained in each mode. Since $[\mathbf{G}_{accel}]$ contains information relating the input torques to the link accelerations, the tests are based on this matrix.

Solving for the degrees of freedom of a mechanism is equivalent to finding the number of independent rows in $[\mathbf{G}_{accel}]$. Thus, the number of degrees of freedom is equal to rank($\mathbf{G}_{accel}$). The rank of a matrix is easily found by performing Gaussian elimination with partial pivoting on $[\mathbf{G}_{accel}]$ until all rows below the pivot row contain only zeros. The number of nonzero rows is equal to the rank.

Once the number of degrees of freedom has been found for each mode, the programmer must choose a set of generalized coordinates with length L_n (Section 3.2.3). Numerical methods may once again be utilized to verify whether these coordinates are valid for each mode. If the rows of $[\mathbf{G}_{accel}]$ are reordered into the form:

$$[\mathbf{G}_{\mathbf{accel}}] = \left[\frac{\mathbf{G}_{\mathbf{accel,gc}}}{\mathbf{G}_{\mathbf{accel,dep}}}\right] \tag{3.11}$$

it is seen that the generalized coordinates are valid if the rows $[\mathbf{G}_{accel,gc}]$ span the row space of $[\mathbf{G}_{accel}]$. $[\mathbf{G}_{accel,gc}]$ spans the row space of $[\mathbf{G}_{accel}]$ if:

$$\text{rank}(\mathbf{G}_{\mathbf{accel}}) = \text{rank}(\mathbf{G}_{\mathbf{accel,gc}}) \tag{3.12}$$

If this is true for each mode, the given choice of generalized coordinates is valid. Then, the state-space matrices (3.6-3.7) can be solved in terms of the selected generalized coordinates. The most difficult matrix to solve for is $[\mathbf{C}_{speeds}]$, which requires that all desired link velocities be solved in terms of the generalized coordinate link velocities. However, $[\mathbf{C}_{speeds}]$ can be solved by noting that each link speed is a linear combination of the independent generalized coordinate speeds. Thus, it can be shown that:

$$\mathbf{G}_{accel,gc}^{T}\mathbf{C}_{speeds}^{T} = \mathbf{G}_{accel}^{T} \tag{3.13}$$

which is solved by a smart Gaussian elimination algorithm that can obtain an answer despite the fact that the matrices are not generally square. Once $[\mathbf{C}_{speeds}]$ is derived, the other matrices are derived as follows:

$$\mathbf{B} = \mathbf{G}_{accel,gc} \tag{3.14}$$

$$\mathbf{D}_{torques} = \mathbf{G}_{clutches} \tag{3.15}$$

This algorithm naturally lends itself to automation, where a computer takes the algebraic equations for each mode and finds all possible combinations of realizable generalized coordinates. Either the computer or the user can choose one of these valid sets of generalized coordinates. Then, the program can generate state-space matrices for the chosen generalized coordinates.

3.2.6 Logic for Changing Status of Coupling Devices

Equations (3.1-3.2) are typically called as separate functions by the simulation program, which uses an integration scheme to determine the results. However, for a system with time-varying dynamics, (3.1-3.2) will not be constant during the simulation. This complicates the problem of modeling the device. Somehow, the simulation program must know which mode is "current" at each time step.

It is clear that mode changing is dependent on the status of the coupling devices. If one of the coupling devices changes from a slipping to a sticking regime or vice versa, the mode will change. Thus, conditions must be developed for determining these transitions. The

conditions depend on the type of device being tested. In general, the status of each coupling device will be denoted as *locked* or *unlocked*. The outline shown below describes these conditions for friction clutches, band brakes, and overrunning clutches.

Friction Clutches and Band Brakes:

Criteria for Locking: The relative speed between the halves of the device changes sign, and if the two halves were to lock together, the torque transmitted through the device would be less than the current static capacity.

Criterion for Unlocking: The torque transmitted through the device becomes larger in absolute value than the current static clutch capacity.

Overrunning (One-Way) Clutch:

Criterion for Locking: The relative speed between the halves of the overrunning clutch changes sign, into the direction in which the clutch is not allowed to rotate. That sign could be positive or negative, and is determined by the particular clutch geometry and the sign convention used for the equations of motion.

Criterion for Unlocking: The torque transmitted through the clutch must change sign into the direction in which the clutch cannot transmit torque. Again, whether the sign must change to a positive or negative value depends on the particular clutch geometry and the sign convention used for the equations of motion.

3.2.7 Structuring Logic for Reduced Complexity

Now that the conditions for locking and unlocking several coupling devices have been given, it is possible to determine the new mode of operation at each integration time step. This problem fits into the realm of sequential logic design. One particular method for creating sequential logic involves the use of state machines. In this method, logical states

represent the specific tasks to be accomplished. For each logical state, specific conditions must be met before the logical state is allowed to change. An initial logical state is given at the start of the program.

While the state machine structure may seem rigid, it greatly simplifies the logic involved in picking the correct modes of operation for automatic transmission modeling. At each time step, the previous mode is recalled from memory. Then, conditions are tested to see if the mode should change. If none of the conditions are met, the mode remains the same. Otherwise, the mode changes to a new value corresponding with the condition that is met. With state machines, it is impossible to be in two logical states at once. Thus, if two conditions are met at the same time, only one is used to determine the new mode of operation.

A common graphical diagram for representing state machines is the State Transition Diagram (STD) [4]. In STDs, the different logical states are represented by bubbles. Allowable transitions between logical states are represented by arrows. Each arrow has a description of the conditions necessary to cause transition between logical states. This type of diagram allows the modeler to clearly lay out the logical states, possible transitions, and conditions associated with transitions while not getting caught in the details of those conditions. The exact details of the transition logic are written out separately. Examples of STDs are explained in Sections 3.3.2 and 3.4.2.

3.3 Example - Simple Transmission

3.3.1 Simulation Results and Terminology

In this section, a simple transmission is modeled using a systematic method. Simulation results are shown, and are used to explain terminology used when discussing automatic transmissions. For a more complete explanation of automatic transmissions, see [23].

A schematic of a simple transmission is shown in Figure 3.1. Note that gears are shifted by changing which clutch is engaged. This is the same method used to shift in an automatic transmission, although overrunning or one-way clutches may also be employed. Thus, the transmission in Figure 3.1 can accurately exhibit many of the same phenomena found in larger, more complicated automatic transmissions.

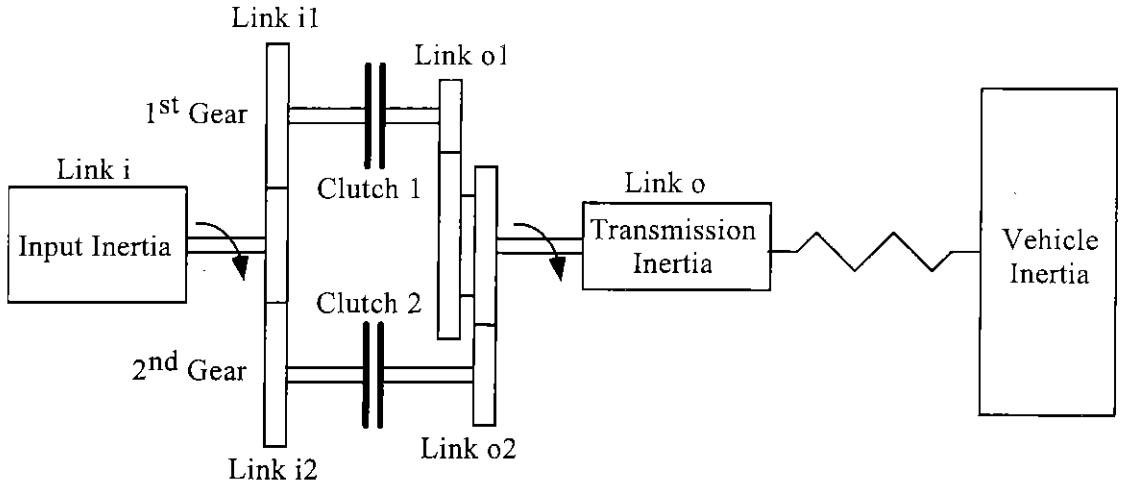

Figure 3.1: Simple Transmission Schematic

Parameters used in the simulation of a 1-2 upshift are shown in Table 3.1. Simulation results are displayed in Figures 3.2-3.7. From the diagrams, it can be seen that two distinct regions exist during the shift. The first region, called the *torque phase*, starts when the oncoming clutch (Clutch 2) is engaged (time=5 seconds). During the torque phase, the offgoing clutch (Clutch 1) remains locked. The increased torque transmitted through the oncoming clutch causes the torque transmitted through the offgoing clutch to decrease. The vehicle inertia, is much larger than the input inertia; therefore, the speeds of the shafts remain essentially constant during the torque phase. This differentiates the torque phase from the *inertia* or *speed phase*, where the offgoing clutch disengages, speeds change, and inertial forces become more significant. The speed phase is in progress from (time=5.24 seconds) to (time=5.63 seconds). The shift is completed when the oncoming clutch locks, synchronizing the input and output speeds in second gear.

Parameter	Value	Units
1st Gear Ratio	6.667:1	None
2nd Gear Ratio	3.636:1	None
Clutch 1 Static Capacity	1000	N-m
Clutch 1 Dynamic Capacity	850	N-m
Clutch 2 Static Capacity	2000	N-m
Clutch 2 Dynamic Capacity	1700	N-m
Input Inertia	2.9	$kg\text{-}m^2$
Transmission Inertia	1.0	$kg\text{-}m^2$
Output Inertia	100.0	$kg\text{-}m^2$
Driveshaft Stiffness	12000	N-m/rad
Driveshaft Damping	200	N-m/(rad/s)

Table 3.1: Parameters For Simple Transmission Simulation

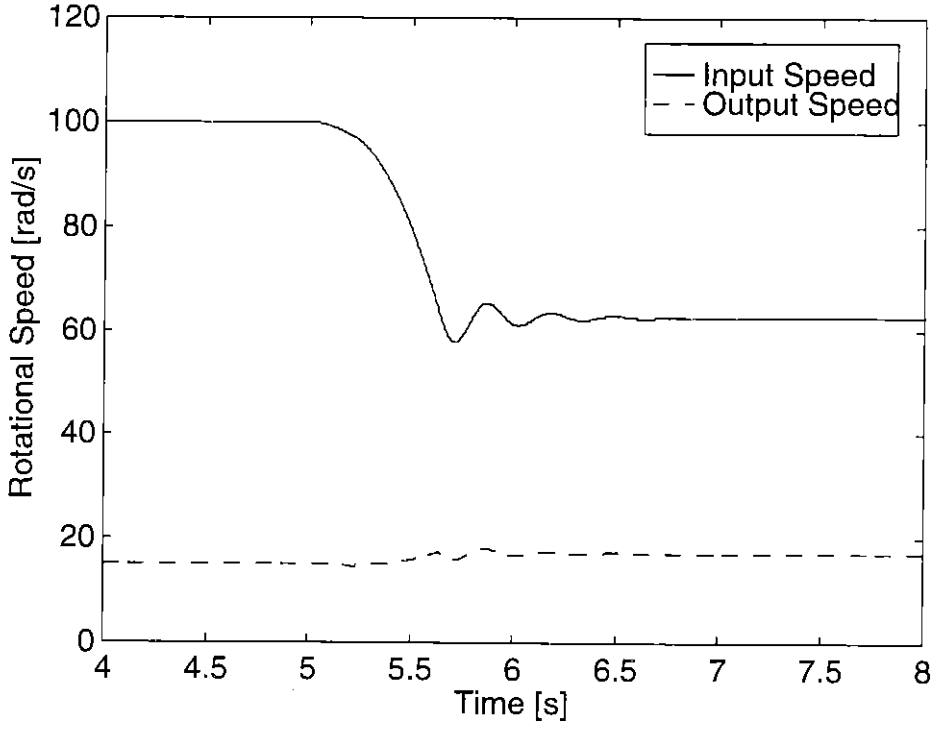

Figure 3.2: Input and Output Speeds for Simple Transmission

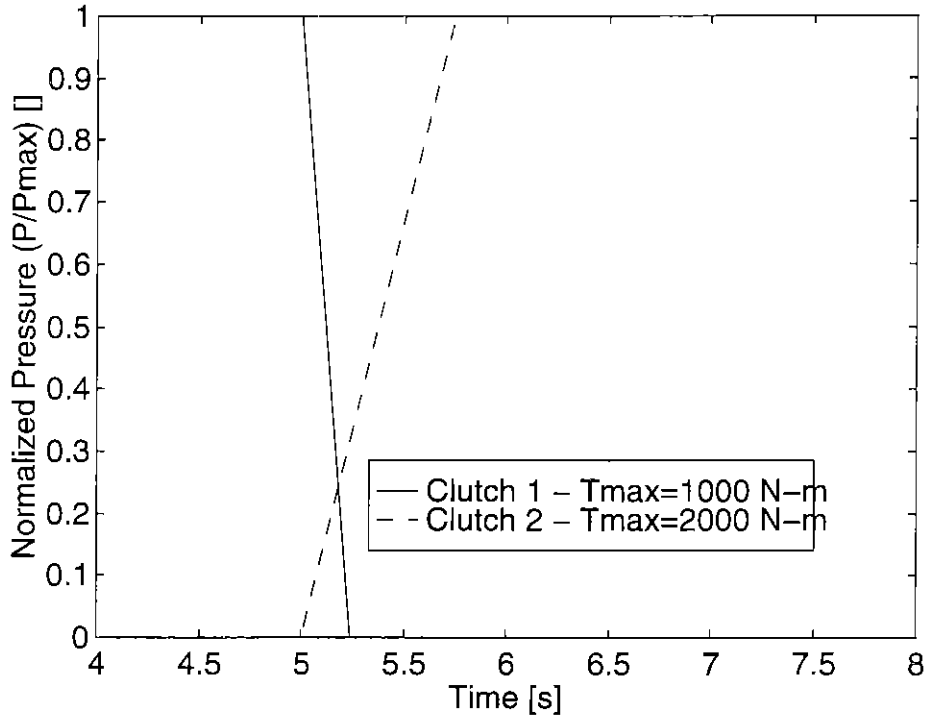

Figure 3.3: Clutch Pressure Profiles for Simple Transmission

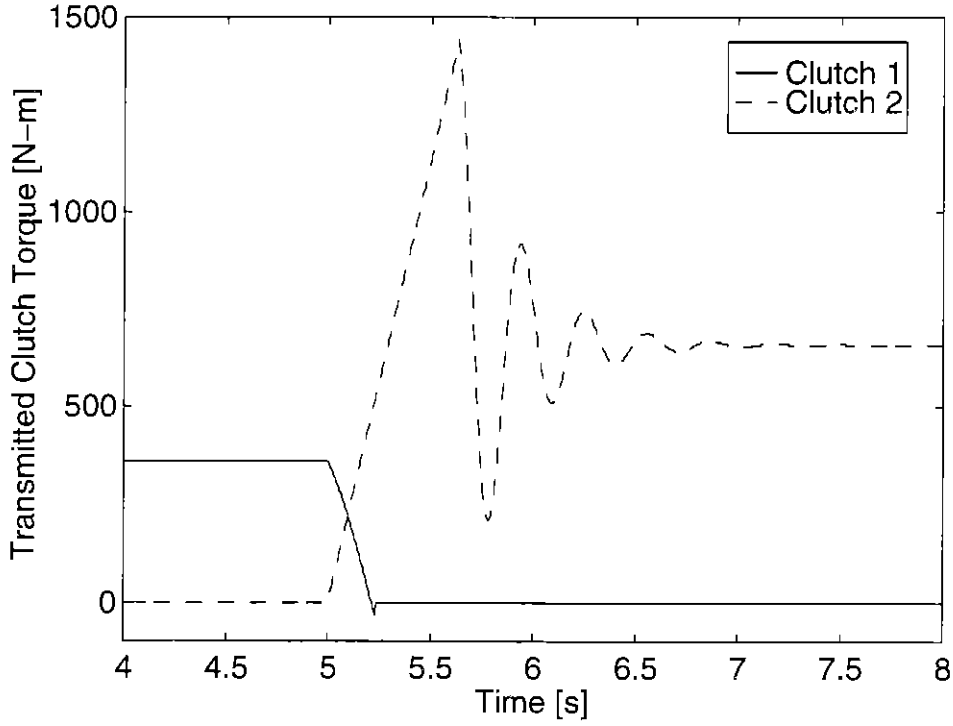

Figure 3.4: Transmitted Clutch Torque for Simple Transmission

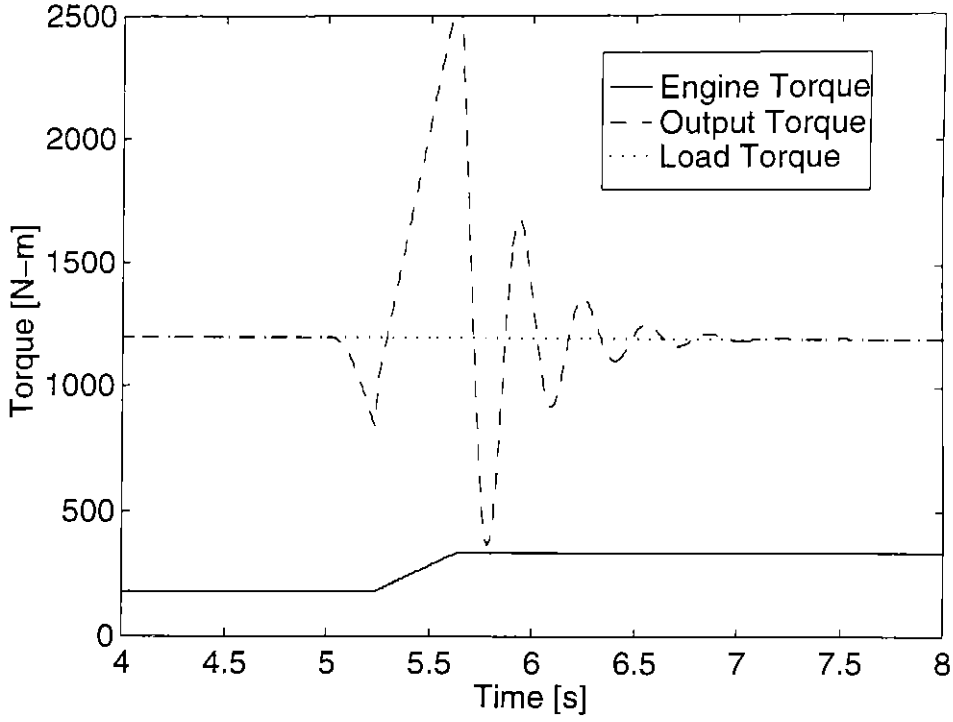

Figure 3.5: Input and Output Torques for Simple Transmission

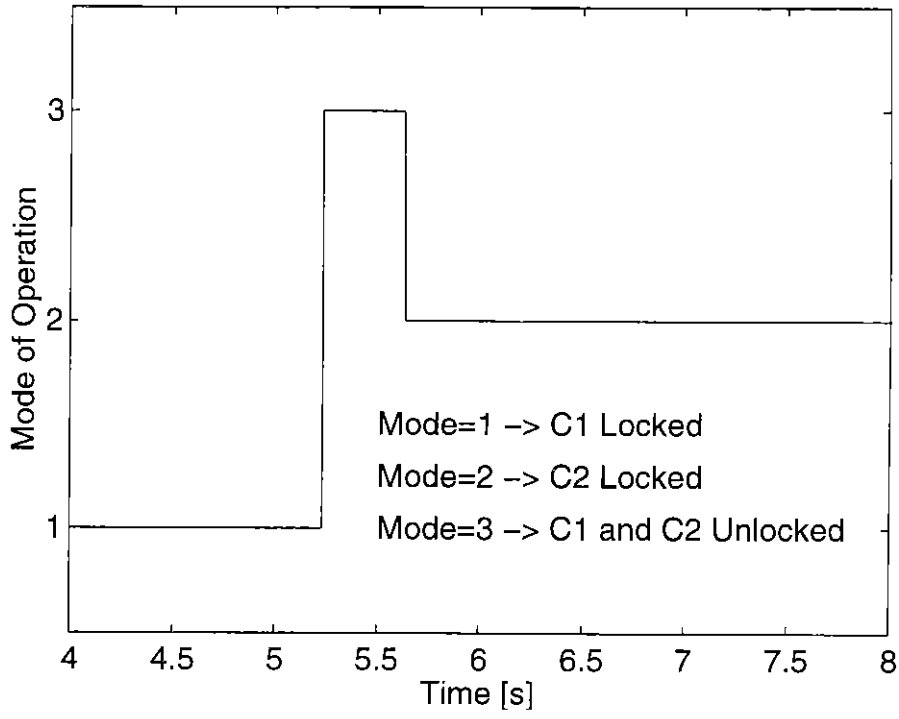

Figure 3.6: Mode of Operation of Simple Transmission

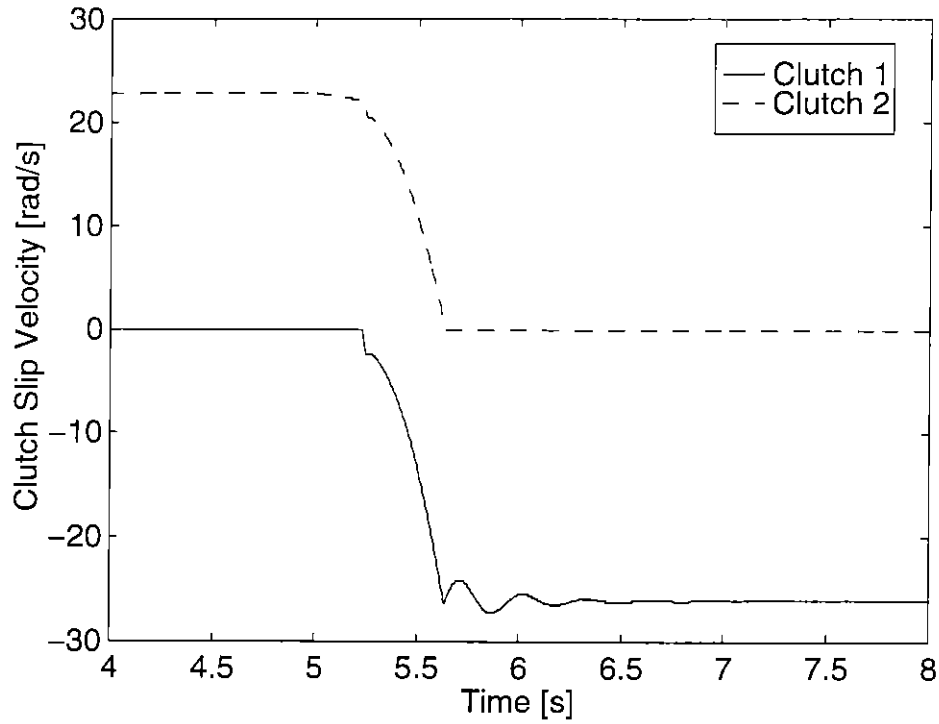

Figure 3.7: Clutch Slip Velocities for Simple Transmission

Notice that the output torque decreases significantly during the torque phase of the shift. This is caused by the transfer of torque from a higher gear ratio (1^{st}) to a lower gear ratio (2^{nd}), and is commonly referred to as the *torque hole.*

Ideally, the oncoming clutch is disengaged at the exact moment when its transmitted torque transitions from a positive to a negative value. This *perfect shift* occurs naturally when overrunning or one-way clutches are used in the mechanism. If friction clutches are used, a perfect shift is difficult to accomplish, but is closely approximated in many modern automatic transmissions. If the offgoing clutch is disengaged too late, the clutches fight each other, and *bindup* occurs. Bindup also enlarges the size of the torque hole more than necessary, giving the driver a feeling of reduced power. If the offgoing clutch is disengaged too early, the capacity of the oncoming clutch is not sufficient to balance the engine torque. The engine accelerates noticeably, and *flare* occurs. Bindup and flare are both undesirable, although flare is more noticeable to a driver. One use of powertrain

simulations is to create good trajectories for the oncoming and offgoing clutch pressures so that nearly perfect shifts can be realized without adding the expense of multiple one-way clutches.

3.3.2 Mode Selection, Switching, and Generalized Coordinates

The STD for the simple transmission of Figure 3.1 is shown in Figure 3.8. Each mode is represented by a bubble, and conditions required to change modes are represented by arrows. A clutch that is disengaged or slipping is referred to as being *unlocked*, while a sticking clutch is referred to as being *locked*. From the diagram, it is clear that three potential modes have been considered:

- Mode 1: Clutch 1 locked, Clutch 2 unlocked
- Mode 2: Clutch 1 unlocked, Clutch 2 locked
- Mode 3: Clutch 1 unlocked, Clutch 2 unlocked

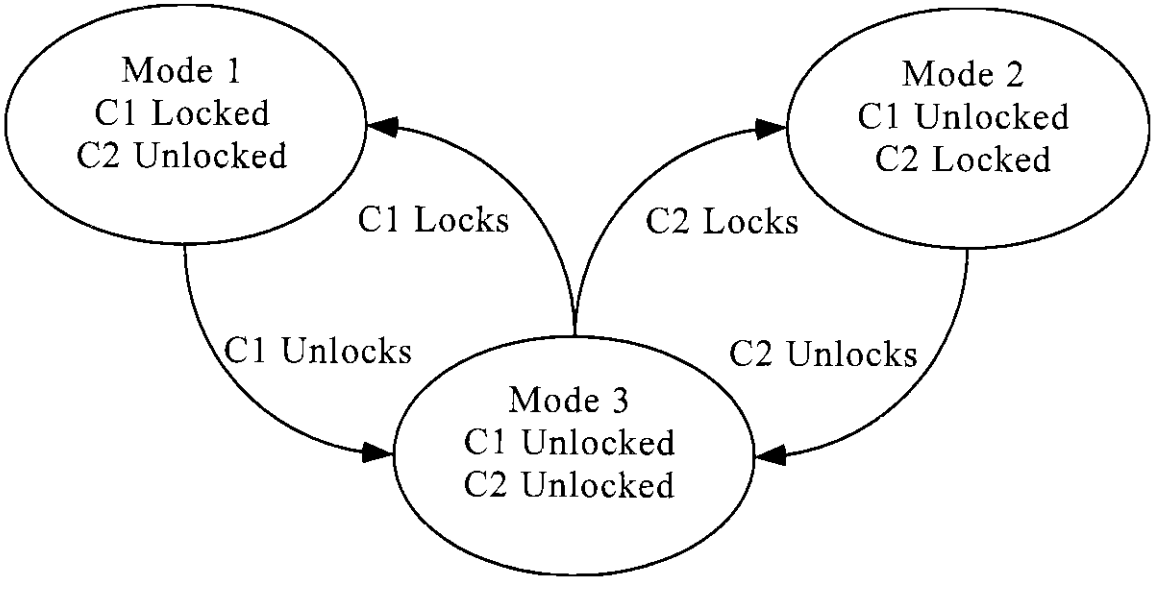

Figure 3.8: State Transition Diagram for Simple Transmission

By looking at the arrows, the conditions for changing modes may be determined:

- If currently in Mode 1 and Clutch 1 unlocks, change to Mode 3.
- If currently in Mode 2 and Clutch 2 unlocks, change to Mode 3.
- If currently in Mode 3 and Clutch 1 locks, change to Mode 1.
- If currently in Mode 3 and Clutch 2 locks, change to Mode 2.

In this simple example, it is obvious that the mechanism has one degree of freedom in Modes 1 and 2, and two degrees of freedom in Mode 3. It is convenient to choose generalized coordinates based on the position of the input and output inertias. Thus, the states for the dynamic equations become:

$$\{\mathbf{x}\} = \begin{Bmatrix} \omega_{input} \\ \omega_{output} \end{Bmatrix} \tag{3.16}$$

In Modes 1 and 2, which only have one degree of freedom, only the first state is used. The second state is linearly dependent on the first state.

3.3.3 Development of Dynamic Equations

At this point, it is helpful to see how the dynamic equations of the simple transmission are developed. The same general process is used with more complicated epicyclic mechanisms such as the Allison HT-740 transmission. The bond graph method was used to determine the dynamic equations for the simple transmission. This bond graph is shown in Figure 3.9. A step-by-step process is used to generate the bond graph. First, it is necessary to understand the kinematic relationships created by the various gear meshes. In bond graph form, these relationships can be represented as multiport transformer elements. The transformer elements have no inertia, but serve to define kinematics and to maintain power conservation. Once the transformers have been defined, inertias and clutches are added to the bond graph. Clutches act as flow sources (motion constraints) when locked, and as effort sources (torque generators) when unlocked. Causality may be assigned to the bond graph in order to obtain a better understanding of the cause-and-effect relationships in the transmission.

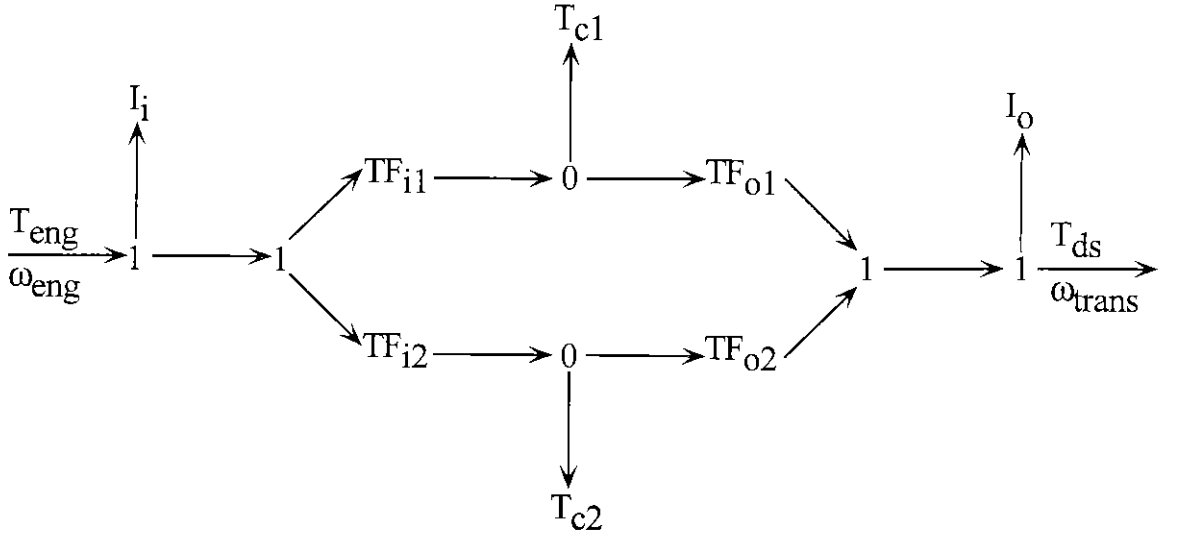

Figure 3.9: Bond Graph for Simple Transmission

At this point, the bond graph is converted to equation form. External torques and unlocked clutch torques serve as inputs, and are treated as known values. In the kinematic equations for the mechanism, link accelerations are substituted for the link velocities. The overall process yields a set of solvable linear algebraic equations. For this simple transmission, symbolic solutions are generated. As is seen from (3.25-3.34), the equations become complicated even for very simple mechanisms. Finally, actual inertias and gear radii are substituted into the symbolic equations, and the results are translated into the state-space matrix elements. Below, equations are given for the simple transmission of Figure 3.1.

Basic Equations: Kinematics and Free Body Diagrams

Figure 3.10 shows the transmission separate from any other springs or inertias, and defines many of the terms used in the following equations:

$$t_i = \frac{1}{\Re_{i1}} t_{C1} + \frac{1}{\Re_{i2}} t_{C2} \tag{3.17}$$

$$t_o = \Re_{o1} \cdot t_{C1} + \Re_{o2} \cdot t_{C2} \tag{3.18}$$

$$t_{eng} - t_i = J_i \cdot \alpha_i \tag{3.19}$$

$$t_o - t_{ds} = J_o \cdot \alpha_o \tag{3.20}$$

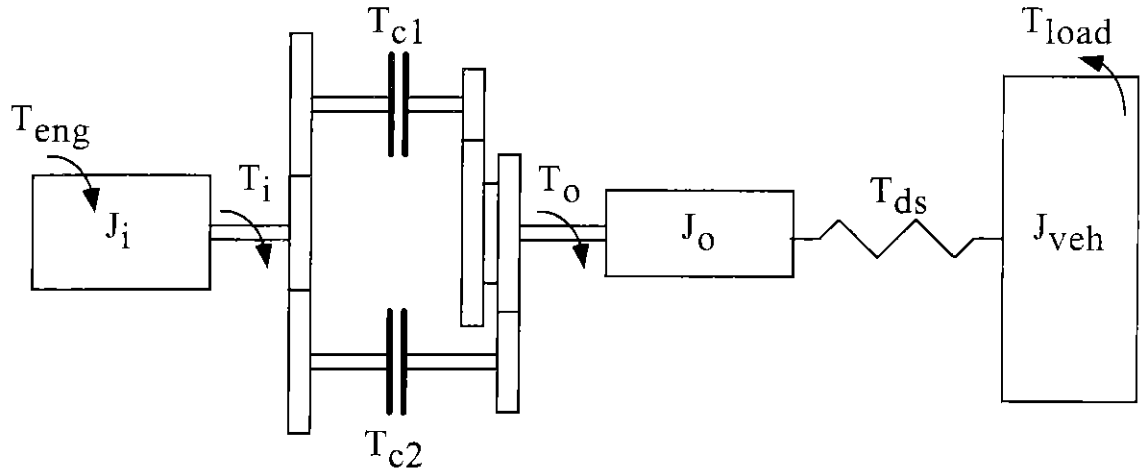

Figure 3.10: Equation Nomenclature for Simple Transmission

where the following substitutions are made in terms of gear radii:

$$\Re_{i1} = \frac{R_{i1}}{R_i} \tag{3.21}$$

$$\Re_{i2} = \frac{R_{i2}}{R_i} \tag{3.22}$$

$$\Re_{o1} = \frac{R_{o,1}}{R_{o1}} \tag{3.23}$$

$$\Re_{o2} = \frac{R_{o,2}}{R_{o2}} \tag{3.24}$$

Mode 1 Results (Clutch 1 Locked)

$$\alpha_{i1} = \alpha_{o1} \tag{3.25}$$

$$\alpha_i = \frac{\left(\Re_{i1}^2\Re_{o1}^2\right)t_{eng} + \left(\frac{\Re_{i1}\Re_{o1}}{\Re_{i2}}\right)\left(\Re_{i2}\Re_{o2} - \Re_{i1}\Re_{o1}\right)t_{C2} - \left(\Re_{i1}\Re_{o1}\right)t_{ds}}{J_o + \Re_{i1}^2\Re_{o1}^2 J_i} \tag{3.26}$$

$$\alpha_o = \frac{\left(\Re_{i1}\Re_{o1}\right)t_{eng} + \left(\frac{1}{\Re_{i2}}\right)\left(\Re_{i2}\Re_{o2} - \Re_{i1}\Re_{o1}\right)t_{C2} - t_{ds}}{J_o + \Re_{i1}^2\Re_{o1}^2 J_i} \tag{3.27}$$

$$t_{C1} = \frac{\left(\Re_{i1} J_o\right) t_{eng} - \left(\frac{\Re_{i1}}{\Re_{i2}}\right)\left(\Re_{i1}\Re_{i2}\Re_{o1}\Re_{o2} J_i + J_o\right) t_{C2} + \left(\Re_{i1}^2 \Re_{o1} J_i\right) t_{ds}}{J_o + \Re_{i1}^2 \Re_{o1}^2 J_i} \tag{3.28}$$

Mode 2 Results (Clutch 2 Locked)

$$\alpha_{i2} = \alpha_{o2} \tag{3.29}$$

$$\alpha_i = \frac{\left(\Re_{i2}^2 \Re_{o2}^2\right) t_{eng} + \left(\frac{\Re_{i2}\Re_{o2}}{\Re_{i1}}\right)\left(\Re_{i1}\Re_{o1} - \Re_{i2}\Re_{o2}\right) t_{C1} - \left(\Re_{i2}\Re_{o2}\right) t_{ds}}{J_o + \Re_{i2}^2 \Re_{o2}^2 J_i} \tag{3.30}$$

$$\alpha_o = \frac{\left(\Re_{i2}\Re_{o2}\right) t_{eng} + \left(\frac{1}{\Re_{i1}}\right)\left(\Re_{i1}\Re_{o1} - \Re_{i2}\Re_{o2}\right) t_{C1} - t_{ds}}{J_o + \Re_{i2}^2 \Re_{o2}^2 J_i} \tag{3.31}$$

$$t_{C2} = \frac{\left(\Re_{i2} J_o\right) t_{eng} - \left(\frac{\Re_{i2}}{\Re_{i1}}\right)\left(\Re_{i1}\Re_{i2}\Re_{o1}\Re_{o2} J_i + J_o\right) t_{C1} + \left(\Re_{i2}^2 \Re_{o2} J_i\right) t_{ds}}{J_o + \Re_{i2}^2 \Re_{o2}^2 J_i} \tag{3.32}$$

Mode 3 Results (Clutch 1 and Clutch 2 Unlocked)

$$\alpha_i = \frac{t_{eng} - \left(\frac{1}{\Re_{i1}}\right) t_{C1} - \left(\frac{1}{\Re_{i2}}\right) t_{C2}}{J_i} \tag{3.33}$$

$$\alpha_o = \frac{\left(\Re_{o1}\right) t_{C1} + \left(\Re_{o2}\right) t_{C2} - t_{ds}}{J_o} \tag{3.34}$$

3.4 Example - Allison HT-740 Automatic Transmission

In this section, the Allison HT-740 automatic transmission is modeled using the same general method outlined in the previous sections and shown by the development of the dynamic equations for the simple transmission of Figure 3.1.

3.4.1 Layout of the Transmission

The layout of the Allison HT-740 transmission is shown in Figure 3.11. As is seen by the labeling of the diagram, this transmission has six rigid links, five clutches, and three planetary gearsets.

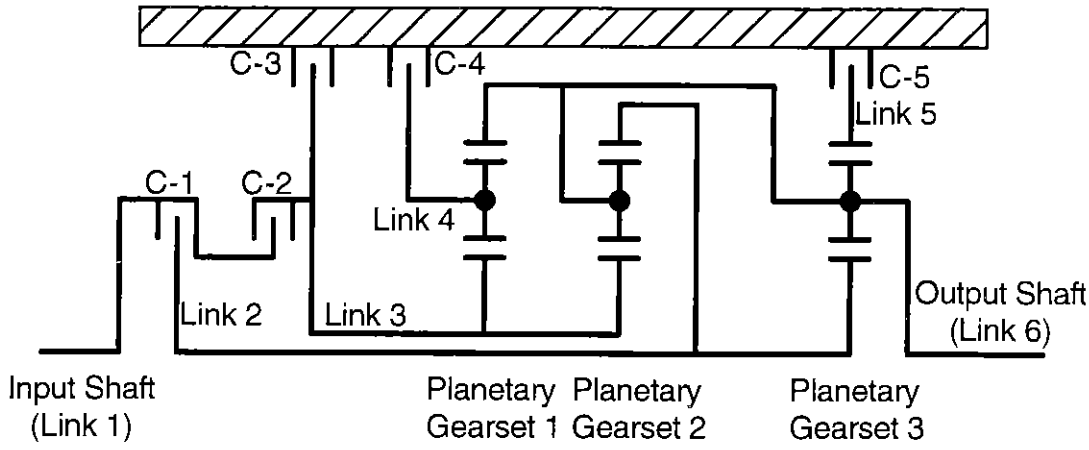

Figure 3.11: Stick Diagram of Allison HT-740 Transmission

The HT-740 has four forward gears and one reverse gear. Table 3.2 explains the clutch engagement sequence. From the diagram, it is clear that three clutches are involved in each shift, but one clutch remains locked during the shift. Thus, there is one oncoming and one offgoing clutch for each shift.

Gear	Clutch 1	Clutch 2	Clutch 3	Clutch 4	Clutch 5
1st Gear	X				X
2nd Gear	X			X	
3rd Gear	X		X		
4th Gear	X	X			
Reverse		X			X

Table 3.2: Clutch Sequences for Allison HT-740 Transmission

3.4.2 Modes of Operation and Mode Switching

Based on the stick diagram of Figure 3.11 and the clutch sequencing of Table 3.2, it is possible to determine which modes of operation should be included in the model. If all clutches are disengaged, it is seen that the mechanism has three degrees of freedom. If this mode is used, three integrators must be used by the simulation program to simulate the mechanism. A decision must be made at this point whether or not the mode should be used. For this analysis, it was assumed that one clutch would always be locked. The additional constraint means the system has at most two degrees of freedom. This assumption saves an integrator, speeds up the simulation, and simplifies the logic required to sense transitions between modes.

Based on this decision and on Table 3.2, the modes were developed. The potential list of modes involves any combination of one or two locked clutches. However, many extraneous modes can be deleted from the list of potential modes. For the HT-740, the following modes were defined:

- Mode 1: 1st Gear - Clutch 1 Locked, Clutch 5 Locked
- Mode 2: 2nd Gear - Clutch 1 Locked, Clutch 4 Locked
- Mode 3: 3rd Gear - Clutch 1 Locked, Clutch 3 Locked
- Mode 4: 4th Gear - Clutch 1 Locked, Clutch 2 Locked
- Mode 5: Forward Gear Speed Phase - Clutch 1 Locked, all other clutches unlocked
- Mode 6: Neutral - Clutch 5 Locked, all other clutches unlocked
- Mode 7: Reverse - Clutch 2 Locked, Clutch 5 Locked

After the modes were defined, it was possible to draw the State Transition Diagram (STD) for the transmission model, shown in Figure 3.12. This diagram clearly shows the chosen modes and the conditions for switching modes in high-level terms. The details of these conditions are given in Section 3.2.6.

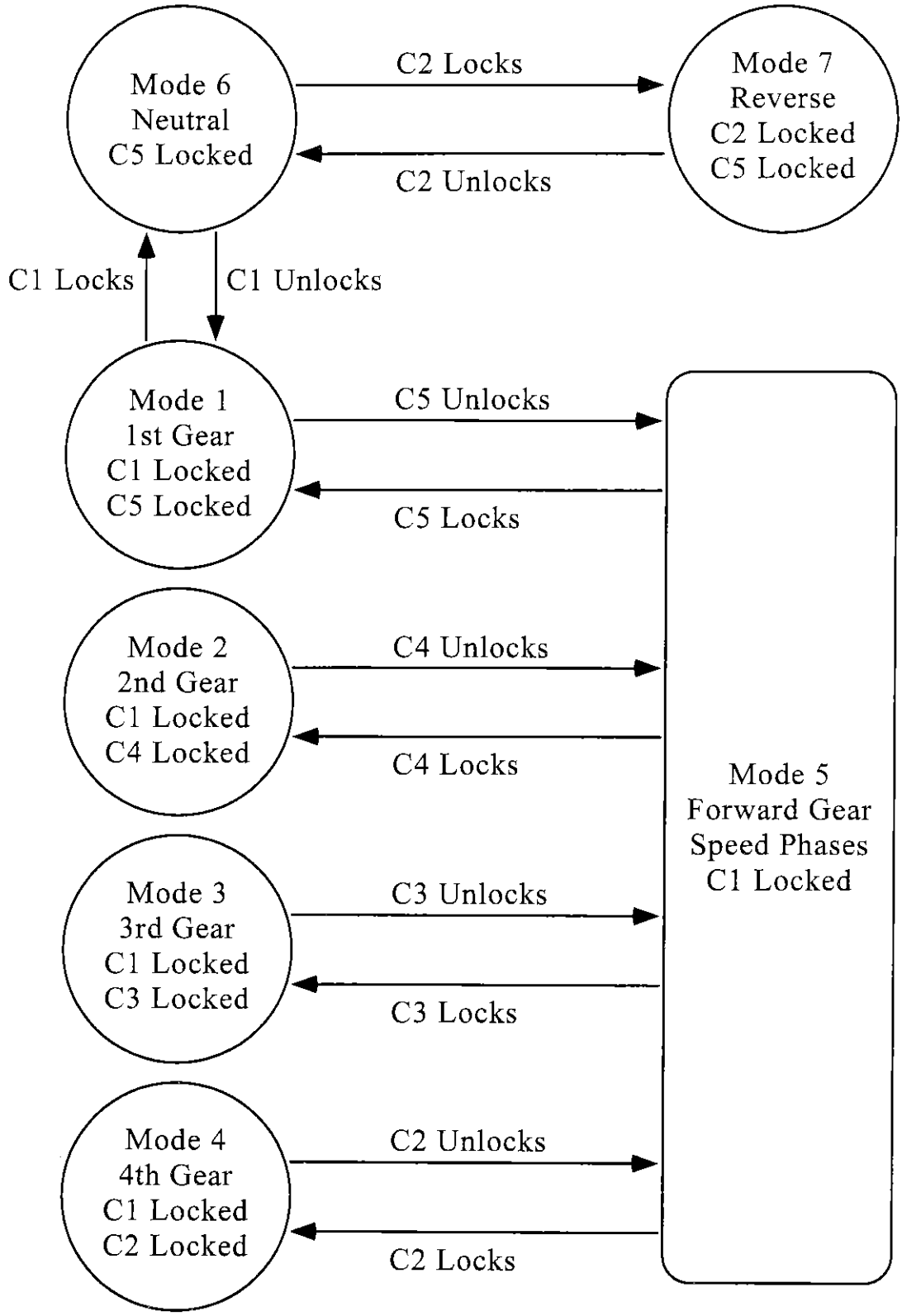

Figure 3.12: State Transition Diagram for Allison HT-740 Transmission

3.4.3 Dynamic Equation Development

The dynamic equations for the HT-740 were developed using bond graphs [20]. The planetary gearset kinematics were modeled as multiport transformers. For each multiport transformer, it is possible to choose certain ports as inputs and solve for the output speeds in terms of the input speeds:

$$\{\omega_{output}\} = [\mathbf{PL}]\{\omega_{input}\} \tag{3.35}$$

where **[PL]** is the multiport transformer modulus. Since multiport transformers have no energy storage capability, the algebraic sum of all power flows into the transformer must be zero. If the correct sign convention is used for power flows into the transformer, it can be proven that [20]:

$$\{\mathbf{t}_{\mathbf{input}}\} = -[\mathbf{PL}]^T\{\mathbf{t}_{\mathbf{output}}\} \tag{3.36}$$

For a simple planetary gearset with one ring gear, one sun gear, and one planet carrier, **[PL]** can be derived as:

$$\{\omega_{carrier}\} = [\mathbf{PL}]\begin{Bmatrix}\omega_{sun}\\ \omega_{ring}\end{Bmatrix} = \begin{bmatrix}\dfrac{N_s}{N_s+N_r} & \dfrac{N_r}{N_s+N_r}\end{bmatrix}\begin{Bmatrix}\omega_{sun}\\ \omega_{ring}\end{Bmatrix} \tag{3.37}$$

$$\begin{Bmatrix}t_{sun}\\ t_{ring}\end{Bmatrix} = -[\mathbf{PL}]^T\{t_{carrier}\} = \begin{bmatrix}\dfrac{-N_s}{N_s+N_r}\\ \dfrac{-N_r}{N_s+N_r}\end{bmatrix}\{t_{carrier}\} \tag{3.38}$$

A completed bond graph, showing power flows but having no causality assignments, is shown in Figure 3.13.

Dynamic equations, including link inertial effects, were generated for each mode using the bond graphs as guides. The input-output relationships for the planetary gearsets had already been derived using multiport transformer elements as described above. These

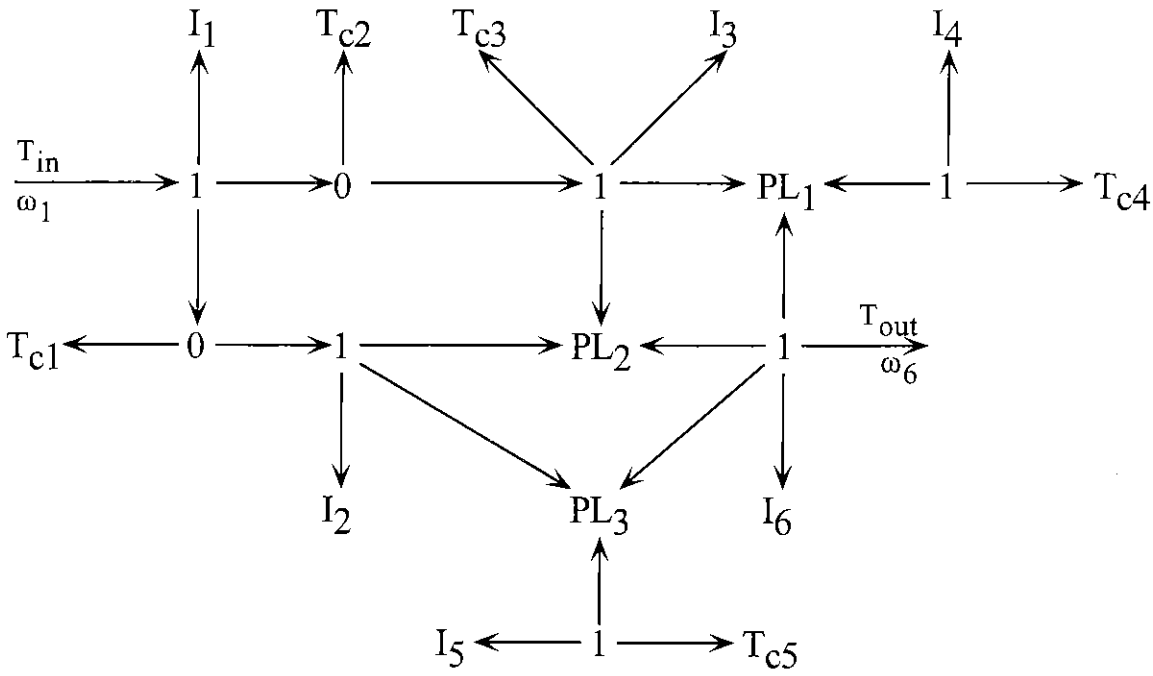

Figure 3.13: Bond Graph of Allison HT-740 Transmission

equations were solved numerically using MATLAB. As described earlier, link accelerations were solved in terms of the input, output, and slipping clutch torques. A review of the equations used is given below.

Dynamic Equations

$$t_{in} - t_{C1} - t_{C2} = J_1 \alpha_1 \tag{3.39}$$

$$t_{C1} - t_{r2} - t_{s3} = J_2 \alpha_2 \tag{3.40}$$

$$t_{C2} - t_{C3} - t_{s1} - t_{s2} = J_3 \alpha_3 \tag{3.41}$$

$$-t_{C4} - t_{pc1} = J_4 \alpha_4 \tag{3.42}$$

$$-t_{C5} - t_{r3} = J_5 \alpha_5 \tag{3.43}$$

$$-t_{out} - t_{r1} - t_{pc2} - t_{pc3} = J_6 \alpha_6 \tag{3.44}$$

Planetary Gearset Kinematic Relationships

$$\alpha_4 = \frac{N_{s1}}{N_{s1} + N_{r1}} \alpha_3 + \frac{N_{r1}}{N_{s1} + N_{r1}} \alpha_6 \tag{3.45}$$

$$\alpha_6 = \frac{N_{s2}}{N_{s2} + N_{r2}} \alpha_3 + \frac{N_{r2}}{N_{s2} + N_{r2}} \alpha_2 \tag{3.46}$$

$$\alpha_6 = \frac{N_{s3}}{N_{s3} + N_{r3}} \alpha_2 + \frac{N_{r3}}{N_{s3} + N_{r3}} \alpha_5 \tag{3.47}$$

Planetary Gearset Power Conservation

$$t_{s1} = \frac{-N_{s1}}{N_{s1} + N_{r1}} t_{pc1} \tag{3.48}$$

$$t_{r1} = \frac{-N_{r1}}{N_{s1} + N_{r1}} t_{pc1} \tag{3.49}$$

$$t_{s2} = \frac{-N_{s2}}{N_{s2} + N_{r2}} t_{pc2} \tag{3.50}$$

$$t_{r2} = \frac{-N_{r2}}{N_{s2} + N_{r2}} t_{pc2} \tag{3.51}$$

$$t_{s3} = \frac{-N_{s3}}{N_{s3} + N_{r3}} t_{pc3} \tag{3.52}$$

$$t_{r3} = \frac{-N_{r3}}{N_{s3} + N_{r3}} t_{pc3} \tag{3.53}$$

For each mode, different clutch constraints were used:

Mode 1: C1 Locked, C5 Locked

$$\alpha_1 = \alpha_2 \tag{3.54}$$

$$\alpha_5 = 0 \tag{3.55}$$

Mode 2: C1 Locked, C4 Locked

$$\alpha_1 = \alpha_2 \tag{3.56}$$

$$\alpha_4 = 0 \tag{3.57}$$

Mode 3: C1 Locked, C3 Locked

$$\alpha_1 = \alpha_2 \tag{3.58}$$

$$\alpha_3 = 0 \tag{3.59}$$

Mode 4: C1 Locked, C2 Locked

$$\alpha_1 = \alpha_2 \tag{3.60}$$

$$\alpha_1 = \alpha_3 \tag{3.61}$$

Mode 5: C1 Locked, all other clutches unlocked

$$\alpha_1 = \alpha_2 \tag{3.62}$$

Mode 6: C5 Locked, all other clutches unlocked

$$\alpha_5 = 0 \tag{3.63}$$

Mode 7: C2 Locked, C5 Locked

$$\alpha_1 = \alpha_3 \tag{3.64}$$

$$\alpha_5 = 0 \tag{3.65}$$

3.4.4 Simulation Results

In order to demonstrate the validity and usefulness of the Allison HT-740 transmission model, a full powertrain simulation was conducted. In the simulation, a large vehicle was accelerated from zero speed to approximately 27 miles per hour over twenty seconds. During this interval, the transmission was shifted sequentially from first gear through fourth gear (overdrive). In order to shift gears, clutch pressures were given time-based trajectories with behaviors similar to production transmissions. All shifts were made when the engine speed reached approximately 1800 revolutions per minute. Table 3.3 provides further data about the shift points.

Shift	Approximate Time [s]	Oncoming Clutch	Offgoing Clutch
1st Gear - 2nd Gear	4 - 5	C4	C5
2nd Gear - 3rd Gear	9 - 10	C3	C4
3rd Gear - 4th Gear	17 - 18	C2	C3

Table 3.3: Simulation Shift Point Data

Two different clutch pressure trajectories were chosen. In the first trajectory, pressures were given that forced a shift to take place, but little consideration was given to shift quality. Thus, potential problems inherent in poor shift trajectories are easily recognizable. Then, the pressure trajectories were modified to eliminate these problems. Several plots are exhibited for each trajectory, and comparisons between the two are made.

Original Clutch Pressure Trajectory

With the first clutch pressure trajectory, problems are immediately evident. Most importantly, it is clear that bindup is occurring at each shift. Bindup is obvious whenever the sign of the offgoing clutch transmitted torque changes during a shift. An effect of bindup, as described in Section 3.3.1, is a depressed torque hole. Due to fighting between the clutches, at each shift, the power dissipation at each shift is higher than necessary. This could cause the clutches to overheat unless extra cooling is provided. A solution to bindup is to lower the pressure on the offgoing clutch sooner; this change is implemented in the improved trajectory.

A second anomaly is seen in Figure 3.18. In the shift between third and fourth gear (time=17 seconds), both clutches are dissipating energy. If the slip velocities (Figure 3.15) and mode of operation (Figure 3.19) are examined, it is seen that the oncoming clutch is forcing the offgoing clutch to unlock; this is due to a mismatch in the static capacities of the two clutches. A great amount of energy is released during this shift, potentially causing overheating problems.

A final difficulty can be seen by examining Figure 3.19. In the clutch trajectories, the oncoming clutch pressure ramps up linearly between 4-5 seconds, 9-10 seconds, and 17-18 seconds. However, in the first two shifts, the speed phase extends past the end of this ramp. Therefore, if the driver were to suddenly increase engine torque during the shift, the oncoming clutch would have insufficient capacity. This would cause the engine to suddenly accelerate, which is clearly undesirable.

Improved Clutch Pressure Trajectory

With the improved clutch pressure trajectory, better performance was obtained by careful regulation of the offgoing clutch pressure. Given proper timing, almost all bindup is eliminated (Figure 3.22). Several other benefits of the improved timing are seen. First, the size of the torque hole is reduced due to the fact that the torque of the offgoing clutch is not allowed to become negative (Figure 3.23 vs. Figure 3.17). Second, power dissipation is reduced because the clutches do not fight each other (Figure 3.24 vs. Figure 3.18). Finally, the anomaly in the shift between third and fourth gear is eliminated. Clearly, simulation allows the designer to quickly examine the impact of various clutch pressure trajectories and to achieve desired results.

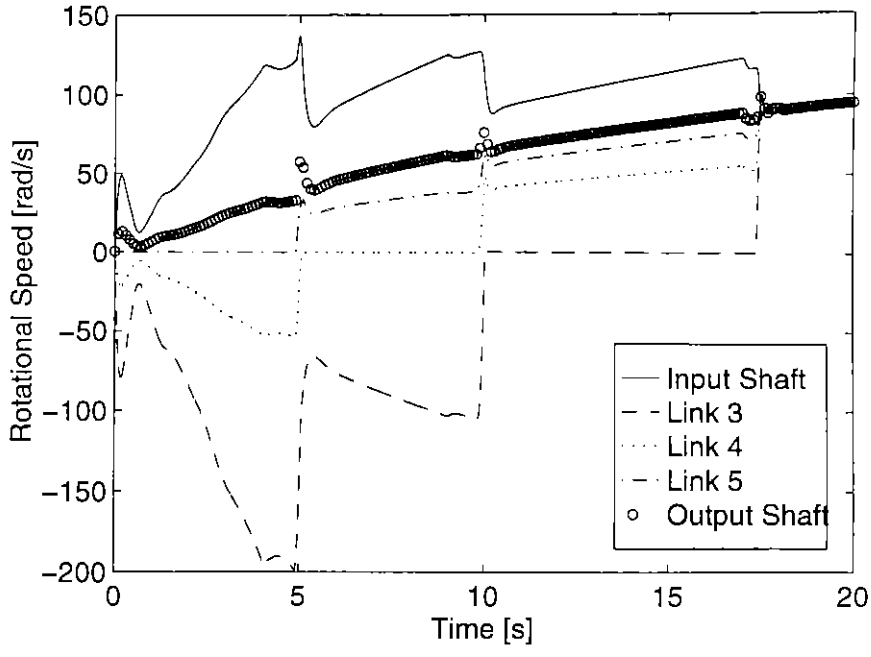

Figure 3.14: Link Speeds, Original Clutch Pressure Trajectory

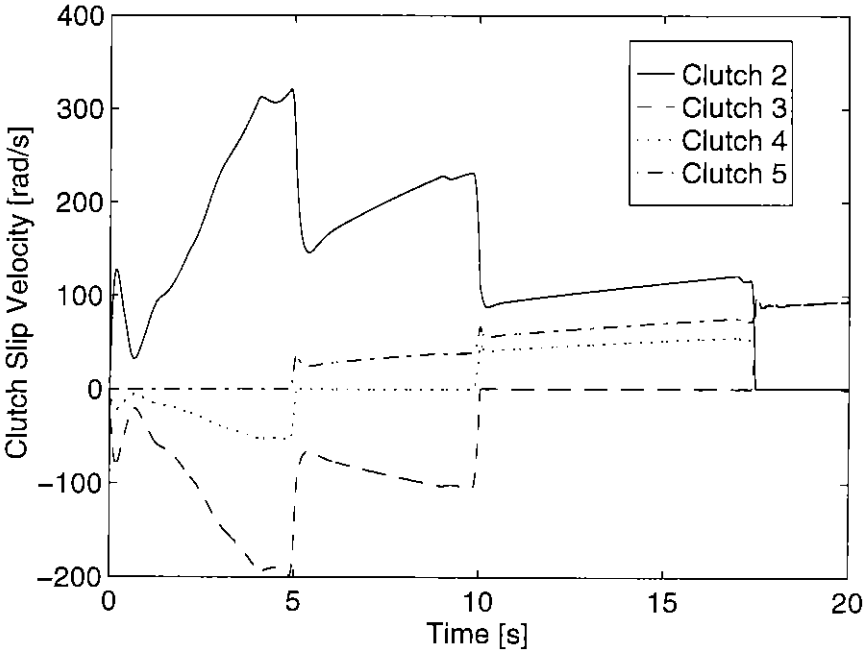

Figure 3.15: Clutch Slip Velocities, Original Clutch Pressure Trajectory

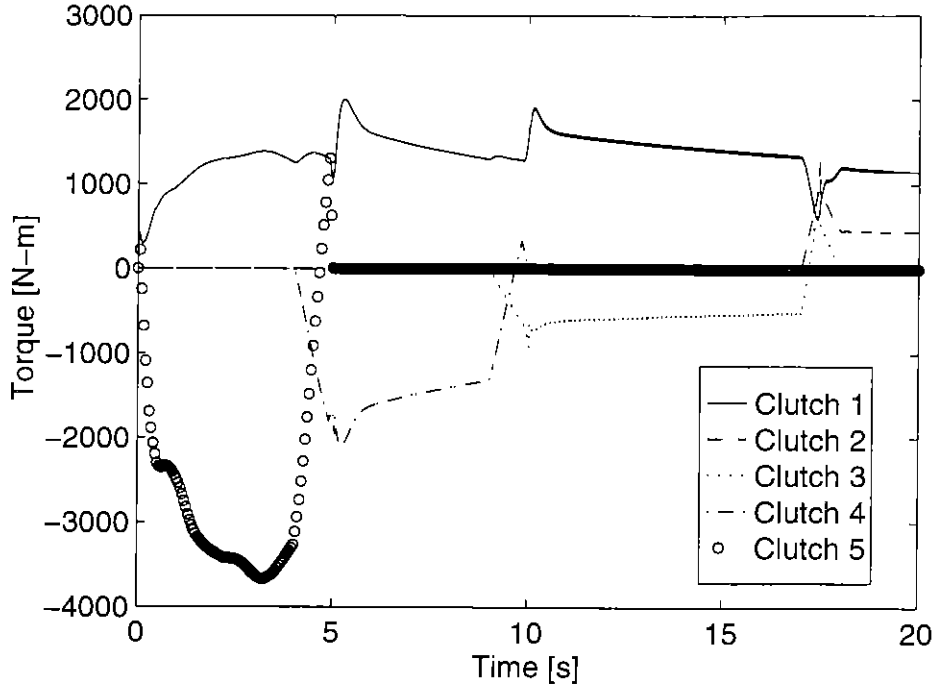

Figure 3.16: Transmitted Clutch Torques, Original Clutch Pressure Trajectory

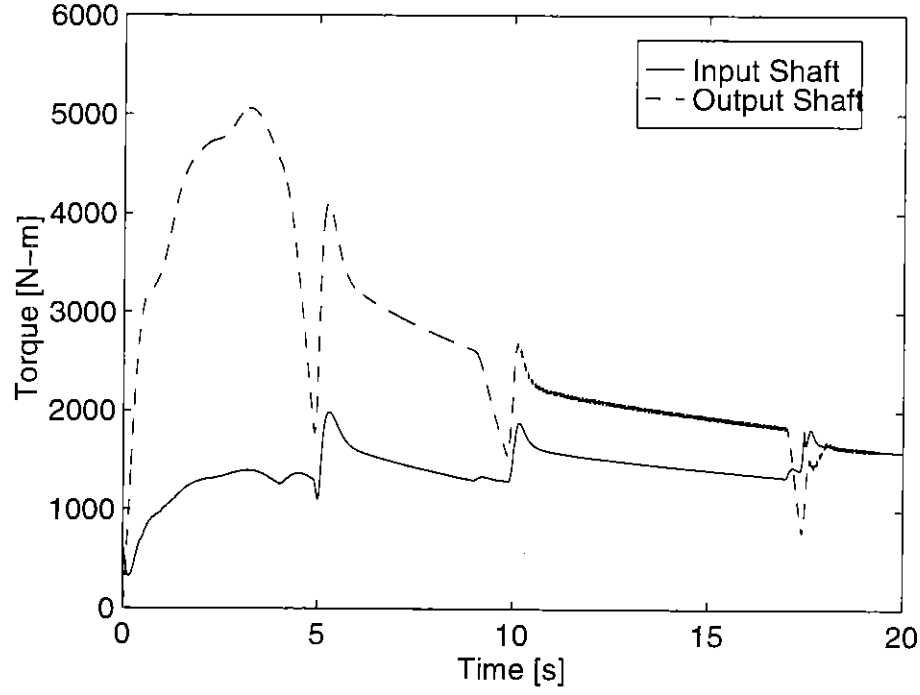

Figure 3.17: Input and Output Torques, Original Clutch Pressure Trajectory

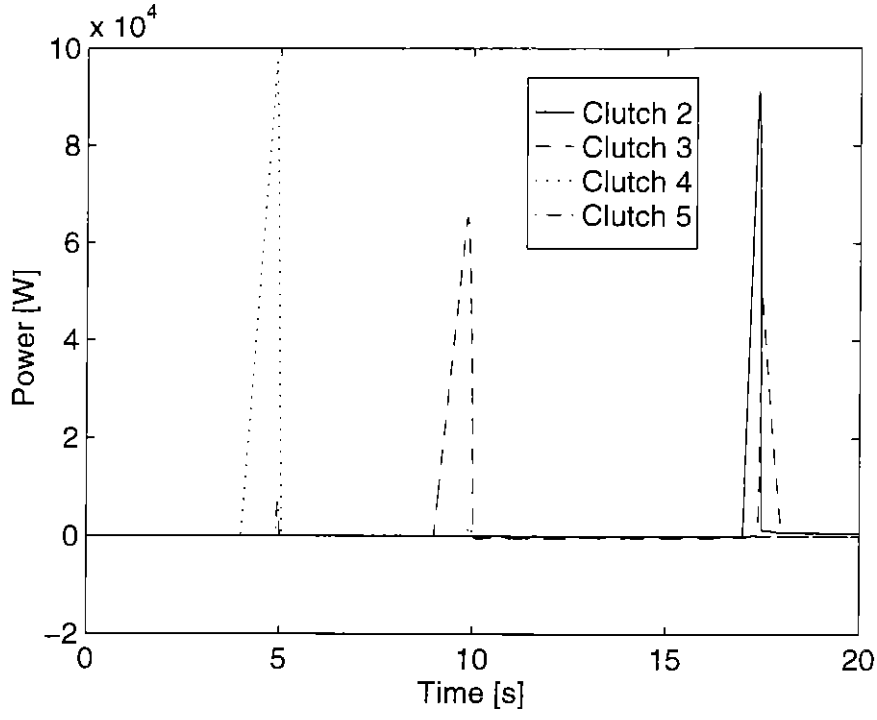

Figure 3.18: Clutch Power Dissipation, Original Clutch Pressure Trajectory

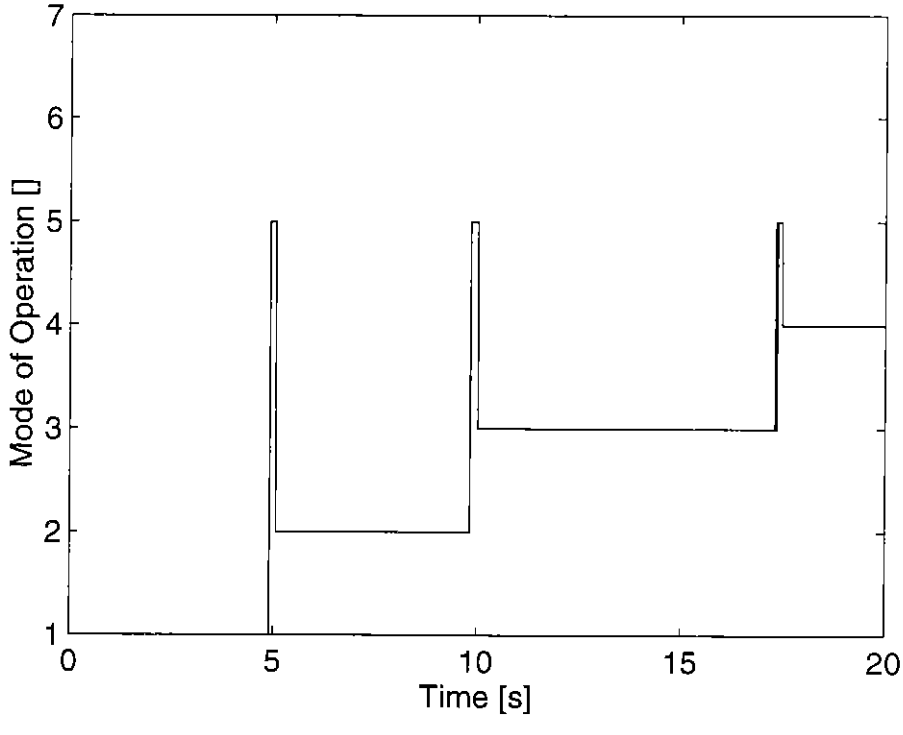

Figure 3.19: Mode of Operation, Original Clutch Pressure Trajectory

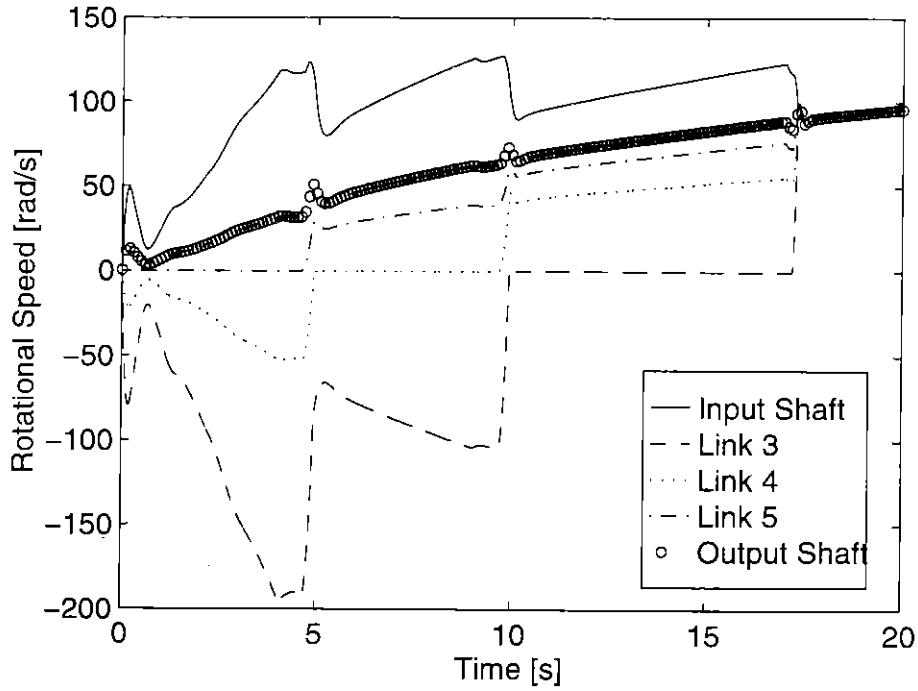

Figure 3.20: Link Speeds, Improved Clutch Pressure Trajectory

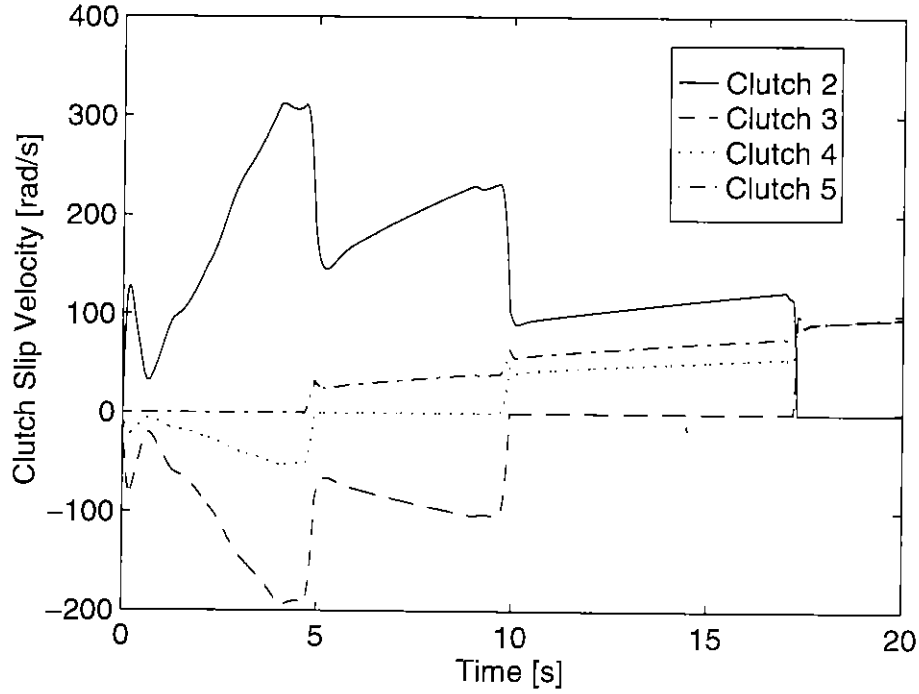

Figure 3.21: Clutch Slip Velocities, Improved Clutch Pressure Trajectory

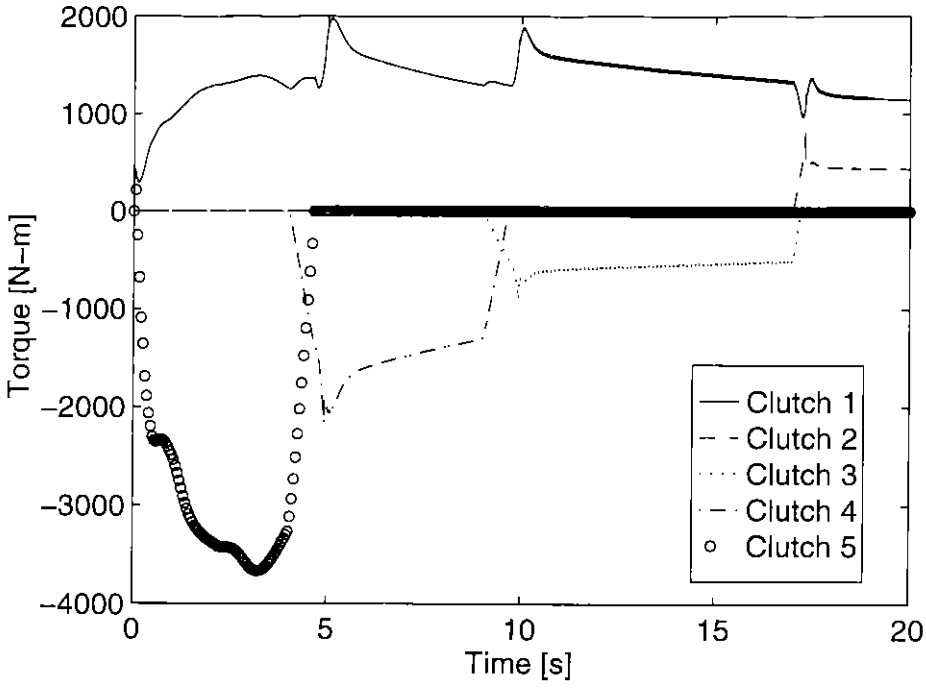

Figure 3.22: Transmitted Clutch Torques, Improved Clutch Pressure Trajectory

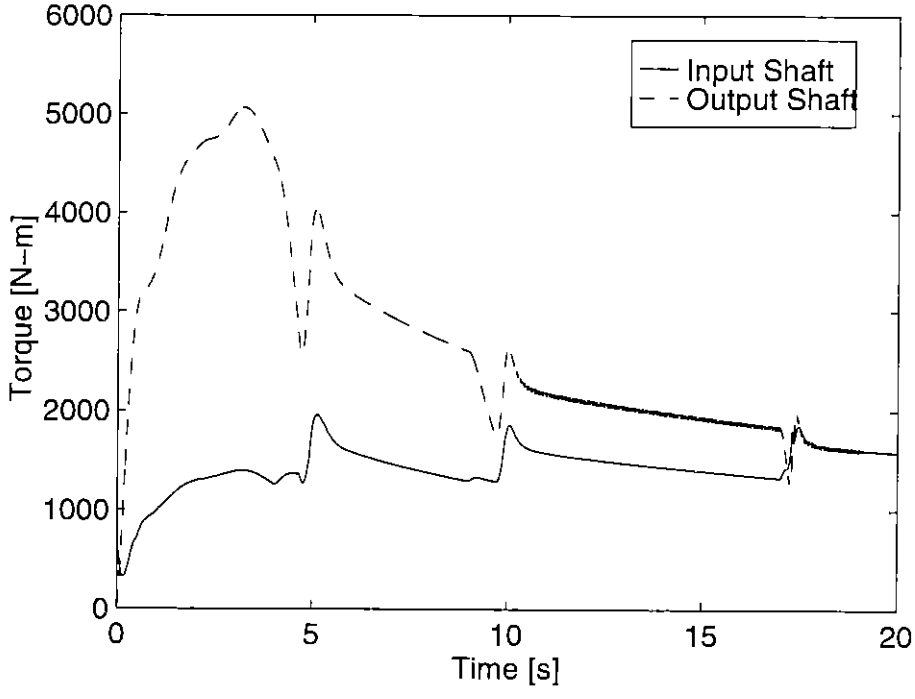

Figure 3.23: Input and Output Torques, Improved Clutch Pressure Trajectory

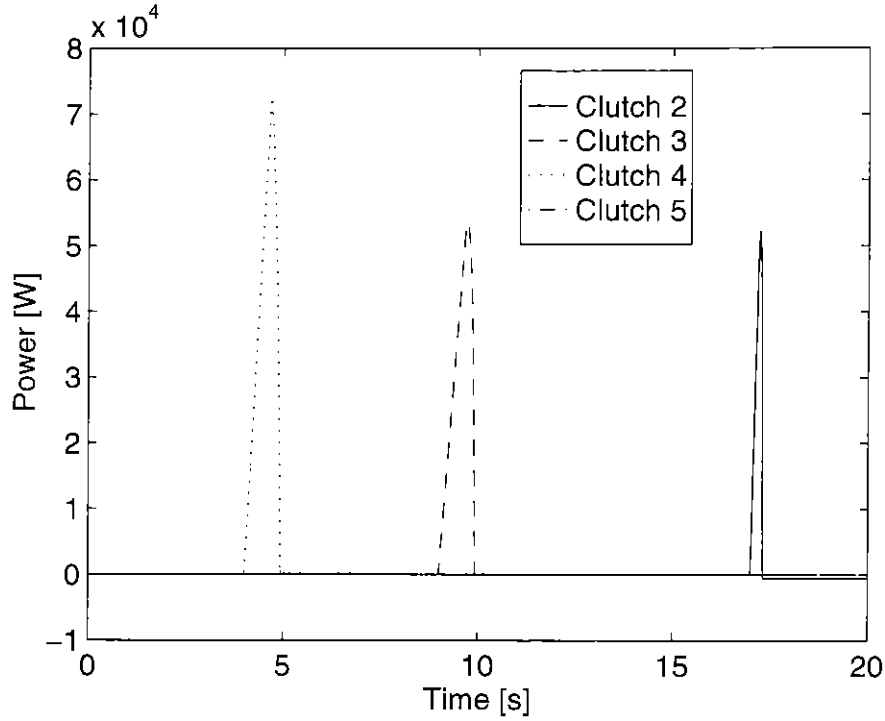

Figure 3.24: Clutch Power Dissipation, Improved Clutch Pressure Trajectory

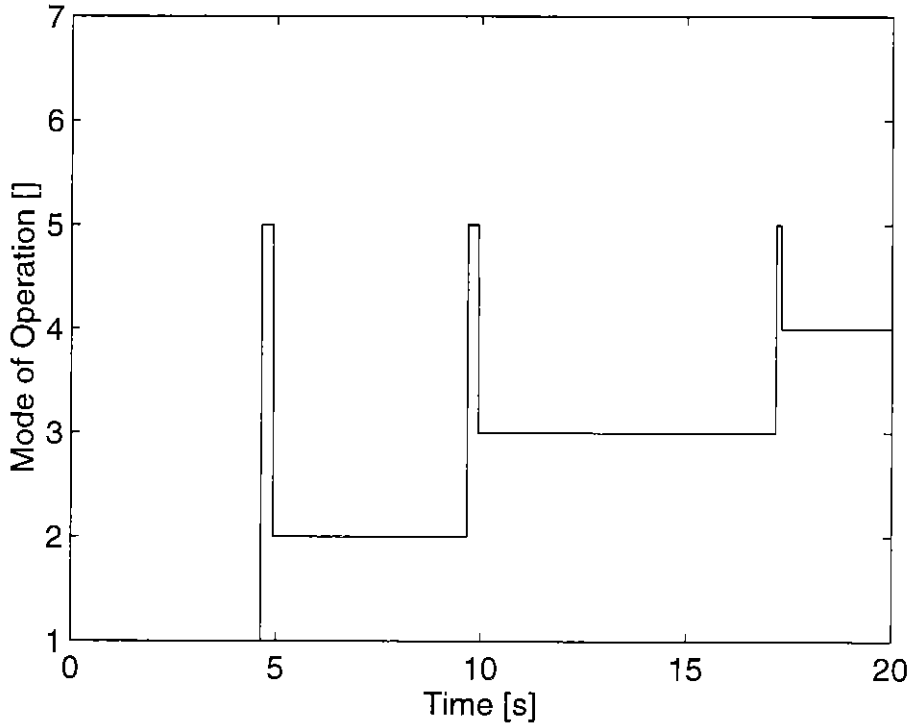

Figure 3.25: Mode of Operation, Improved Clutch Pressure Trajectory

Chapter 4: Turbocharger Modeling

4.1 Introduction

The turbocharger is an essential element in a high power density engine application. The turbocharger's duty is to provide a higher air mass flow rate to the engine by capturing some of the energy in the exhaust stream, which allows a proportionally greater amount of fuel to be added during each cycle. The additional fueling ability, for a given engine, will increase the maximum power output capability of the engine.

Turbochargers consist of two primary elements: a compressor and a turbine. Hot exhaust gas flows past the turbine, which converts some of the engine's exhaust gas energy into useful work at the turbine's output shaft. The compressor uses work applied to an input shaft to convert relatively low energy intake air into a higher energy flow. When the turbine's output shaft is connected to the compressor's input shaft, the turbine is able to provide the mechanical energy necessary to turn the compressor.

Turbines and compressors are typically divided into two categories, based on how the gas flows through the device. Axial flow devices have gas flowing parallel to the rotating shaft. Typical examples include the turbines and compressors in aircraft jet engines. Centrifugal flow devices have gas flowing radially between the shaft rotational axis and the circular rotor's periphery. An example of a centrifugal flow compressor rotor is shown in Figure 4.1. Most compressors and turbines used in diesel engines for automotive applications are of the centrifugal (or radial) flow variety. Thus, emphasis is made on modeling the radial flow components, which have different characteristics from axial flow devices.

If the turbine and compressor are properly matched, the use of these devices will significantly improve the overall engine performance as described above. It is possible to match turbines and compressors in a static or steady-state sense, or in a transient or time-varying sense. Since an automotive engine typically undergoes frequent transients, it is important to understand how the turbocharger and the engine as a whole operate under these conditions. The transient operation of the turbocharger has a significant impact on

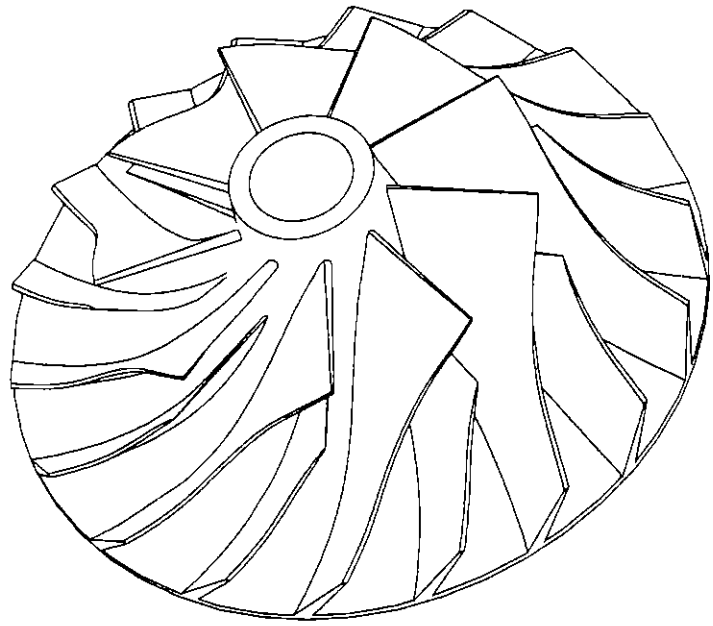

Figure 4.1: Centrifugal Compressor Rotor [23]

emissions, especially for diesel engines. If a turbocharger has a tendency to starve the engine of needed intake air flow during transients, a plume of black smoke may be emitted from the exhaust pipe due to excess fueling. The smoke is ugly and hazardous to the environment, and in military applications, the *signature* may also place troops in danger of attack by enemies who notice the smoke.

As described above, transient turbocharger models may be useful to engine designers. Other authors have noticed this need, and have described their techniques for modeling turbochargers under transient conditions [1, 3, 6, 9, 11, 15, 21, 24]. In the following pages, methods for transient turbocharger modeling will be described which have been implemented successfully in a graphical simulation environment.

4.2 Quasi-Steady Modeling

The concept of a *quasi-steady* model means that the forces acting on the rotor at each moment in time are approximately the same as if it were operated in the steady state with the same operating conditions. These forces are then integrated to estimate rotor speed. This simplifying assumption relies on knowledge that certain dynamics within the device

will occur much faster than the required model bandwidth. Thus, at each time instant in the simulation, it can be assumed that these "fast" dynamics have already settled to the steady-state values without causing significant modeling error.

For turbocharger quasi-steady modeling, the main assumptions are that:

- The internal volumes of the turbine and compressor are small and have little effect on the overall dynamics [25].
- The behavior of the turbocharger is not dependent on the rates of change of pressures, temperatures, or rotational speed.

When quasi-steady behavior is assumed, models can be based on steady-state data available from turbocharger manufacturers. Little transient data is available, and if needed for model validation, must be obtained from special experimental setups.

4.3 Basic Turbocharger Thermodynamics

Many standard thermodynamics textbooks provide a basic understanding of how compressors and turbines operate [17]. An overview of the basic thermodynamic equations will be provided, and special forms will be shown which facilitate the design of a quasi-steady turbocharger model.

4.3.1 Compressor Thermodynamics

If a control volume is drawn around the input and output ports of a compressor, energy and entropy balances can be performed on the compressor. Figure 4.2 shows the energy and entropy balances with appropriate terms included. Note that heat transfer is neglected; heat transfer is difficult to determine and is neglected since it is typically small in relation to the other energy flows.

For the energy and entropy balances, the following equations are derived:

$$\dot{m}_{comp} \cdot \left(h_1 + \frac{V_1^2}{2} \right) + \dot{W}_{comp} = \dot{m}_{comp} \cdot \left(h_2 + \frac{V_2^2}{2} \right) \tag{4.1}$$

Compressor efficiency is defined as:

$$\dot{m}_{comp} \cdot s_1 + s_{produced} = \dot{m}_{comp} \cdot s_2 \tag{4.2}$$

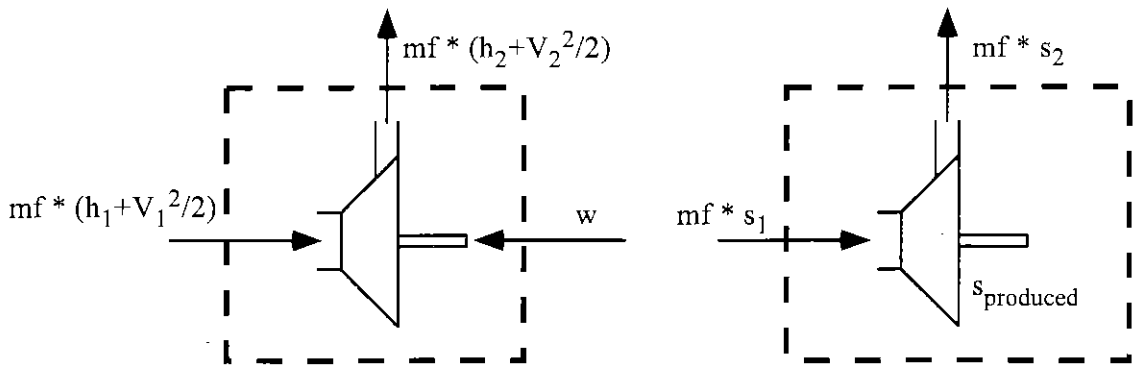

Figure 4.2: Compressor Energy and Entropy Balances

If ideal gases are assumed and kinetic energy effects are eliminated, the equations for compressor outlet flow temperature and compressor shaft torque can be computed [10]:

$$\eta_{comp} = \frac{\dot{W}_{comp,isentropic}}{\dot{W}_{comp,actual}} \tag{4.3}$$

$$\frac{T_{2,isentropic}}{T_1} = \left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}} \tag{4.4}$$

$$\dot{W}_{comp,isentropic} = \dot{m}_{comp} \cdot C_p \cdot \left(T_{2,isentropic} - T_1\right) \tag{4.5}$$

$$\dot{W}_{comp,actual} = \dot{m}_{comp} \cdot C_p \cdot \left(T_2 - T_1\right) \tag{4.6}$$

$$T_2 = T_1\left\{1 + \frac{1}{\eta_{comp}}\left[\left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]\right\} \tag{4.7}$$

$$\dot{W}_{comp,actual} = t_{comp} \cdot \omega_{comp} \tag{4.8}$$

$$t_{comp} = \frac{\dot{m}_{comp} C_p T_1}{\eta_{comp}\omega_{comp}}\left\{\left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}} - 1\right\} \tag{4.9}$$

4.3.2 Turbine Thermodynamics

Likewise, if a control volume is drawn around the input and output ports of a turbine, energy and entropy balances can be performed on the turbine. Figure 4.3 shows the energy and entropy balances with appropriate terms included. Heat transfer is again neglected.

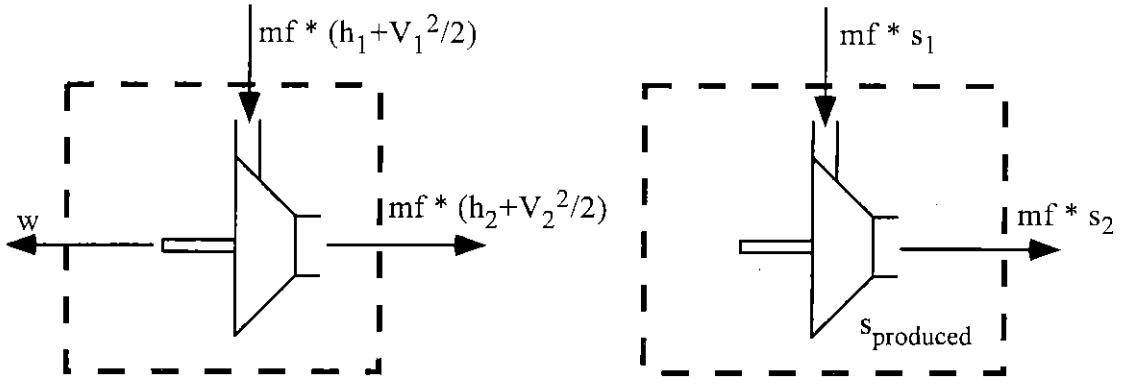

Figure 4.3: Turbine Energy and Entropy Balances

For the energy and entropy balances, the following equations are derived:

$$\dot{m}_{turb} \cdot \left(h_1 + \frac{V_1^2}{2} \right) = \dot{m}_{turb} \cdot \left(h_2 + \frac{V_2^2}{2} \right) + \dot{W}_{turb} \tag{4.10}$$

$$\dot{m}_{turb} \cdot s_1 + s_{produced} = \dot{m}_{turb} \cdot s_2 \tag{4.11}$$

Turbine efficiency is defined as:

$$\eta_{turb} = \frac{\dot{W}_{turb,actual}}{\dot{W}_{turb,isentropic}} \tag{4.12}$$

If ideal gases are assumed and kinetic energy effects are eliminated, the equations for turbine outlet flow temperature and turbine shaft torque can be computed [11]:

$$\frac{T_{2,isentropic}}{T_1} = \left(\frac{P_2}{P_1} \right)^{\frac{\gamma - 1}{\gamma}} \tag{4.13}$$

$$\dot{W}_{turb,isentropic} = \dot{m}_{turb} \cdot C_p \cdot \left(T_1 - T_{2,isentropic} \right) \tag{4.14}$$

$$\dot{W}_{turb,actual} = \dot{m}_{turb} \cdot C_p \cdot (T_1 - T_2) \tag{4.15}$$

$$T_2 = T_1 \left\{ 1 + \eta_{turb} \left[\left(\frac{P_2}{P_1} \right)^{\frac{\gamma-1}{\gamma}} - 1 \right] \right\} \tag{4.16}$$

$$\dot{W}_{turb,actual} = t_{turb} \cdot \omega_{turb} \tag{4.17}$$

$$t_{turb} = \frac{\dot{m}_{turb} C_p T_1 \eta_{turb}}{\omega_{turb}} \left\{ 1 - \left(\frac{P_2}{P_1} \right)^{\frac{\gamma-1}{\gamma}} \right\} \tag{4.18}$$

4.4 Initial Turbocharger Modeling Approach

The compressor and turbine models are based on steady-state lookup tables available from turbocharger manufacturers. In the following sections, procedures are described for the quasi-steady modeling of turbocharger thermodynamics with dynamic modeling of turbocharger rotor inertia. The initial models assume that kinetic energy effects are negligible.

4.4.1 Centrifugal Flow Compressor

The compressor data available from turbocharger manufacturers is often defined by contour maps of the form [11]:

$$N_{corr,comp} = f\left(\dot{m}_{corr,comp}, \frac{P_{02}}{P_{01}} \right) \tag{4.19}$$

$$\eta_{comp} = f\left(\dot{m}_{corr,comp}, \frac{P_{02}}{P_{01}} \right) \tag{4.20}$$

where

$$N_{corr,comp} = N_{comp} \cdot \sqrt{\frac{T_{std,comp}}{T_{01}}} \tag{4.21}$$

$$\dot{m}_{corr,comp} = \dot{m}_{comp} \cdot \frac{P_{std,comp}}{P_{01}} \sqrt{\frac{T_{01}}{T_{std,comp}}} \tag{4.22}$$

P_{01}, P_{02}, T_{01}, and T_{02} are total pressures and temperatures. If kinetic energy effects are neglected, static pressures and temperatures can be substituted for the total properties.

Corrected mass flow and compressor speed are used to compensate for different environmental conditions than those under which the steady-state experiments were made. An example of an actual compressor lookup table is shown in Figure 4.4. Two critical regions for the lookup table are labeled as *surge* and *stall lines*. Along the surge line, the rotor speed contours become nearly horizontal. To the left of the surge line, the speed contours drop with respect to pressure ratio. Thus, for a given rotor speed and pressure ratio, two different mass flows are available. This may create an unstable "surging" phenomenon which can destroy the compressor. Along the stall line, mass flow becomes limited or choked. At each rotor speed, a pressure ratio for which surge and stall occur can be identified. This fact will become useful in a later data transformation.

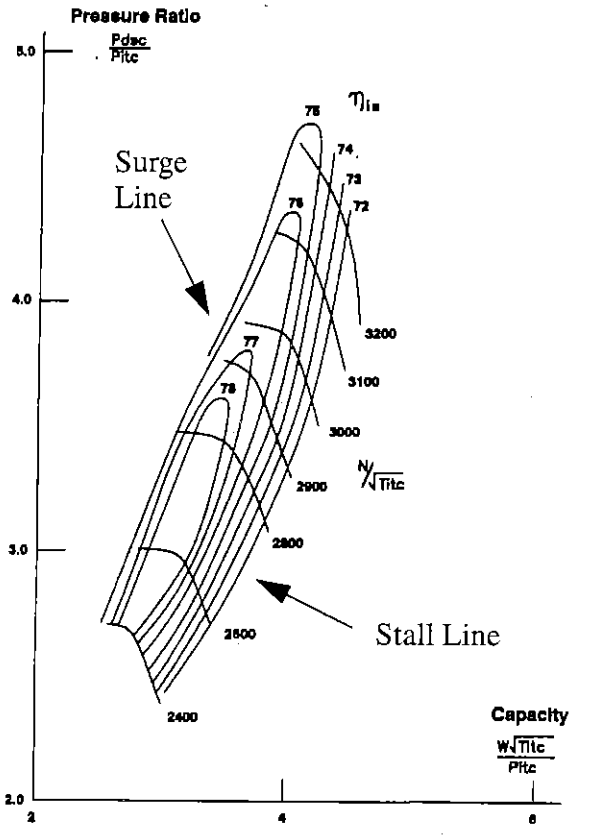

Figure 4.4: Typical Centrifugal Compressor Map [23]

Before defining a compressor algorithm, it is first necessary to determine the necessary inputs and outputs. To maintain an explicitly causal relationship between variables, it is convenient to situate the compressor between two filling-and-emptying manifolds as in Figure 4.5.

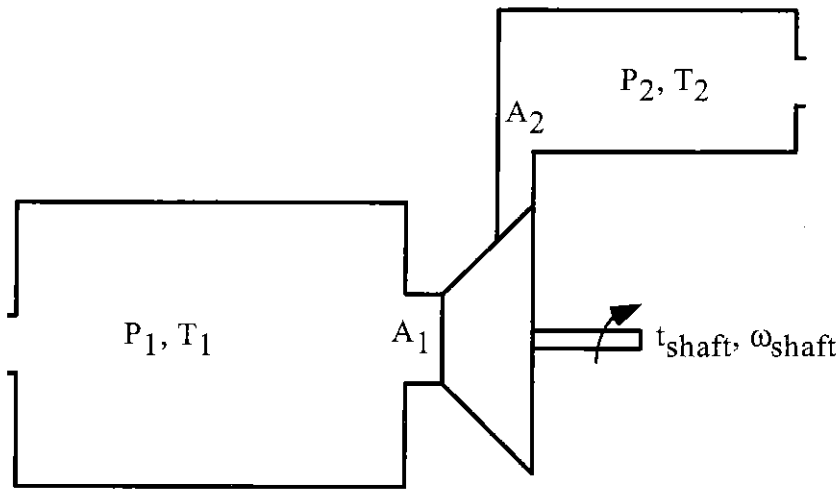

Figure 4.5: Schematic of Compressor Model with Nomenclature

The following are inputs to the compressor model:

- Pressure (1)
- Temperature (1)
- Pressure (2)
- Rotor speed

The outputs calculated by the compressor thermodynamic model are:

- Outlet temperature
- Mass flow rate
- Shaft torque

Notice that the inputs and outputs described above do not match the lookup tables shown in Figure 4.4 and described by equations (4.19) and (4.20). Using data analysis software, the data is reformulated into the two following functional relationships:

$$\dot{m}_{corr,comp} = f\left(N_{corr,comp}, \chi\right) \tag{4.23}$$

$$\eta_{comp} = f\left(N_{corr,comp}, \chi\right) \tag{4.24}$$

where

$$\chi = \frac{\left(P_{02} / P_{01}\right)\Big|_{actual} - \left(P_{02} / P_{01}\right)\Big|_{stall}}{\left(P_{02} / P_{01}\right)\Big|_{surge} - \left(P_{02} / P_{01}\right)\Big|_{stall}} \tag{4.25}$$

is the normalized pressure coordinate evaluated at the current corrected rotor speed. Values of χ between 0 and 1 indicate normal operation. Values greater than 1 indicate surge, and values less than 1 indicate stalled operation. Figure 4.4 shows that mass flow and efficiency are more sensitive to changes in pressure ratio at lower rotor speeds. Use of χ instead of the actual pressure ratio allows smaller lookup tables to be used while maintaining the same accuracy at low speeds. Actual surface plots of compressor mass flow and efficiency in the format of equations (4.23) and (4.24) are shown in Figures 4.6 and 4.7.

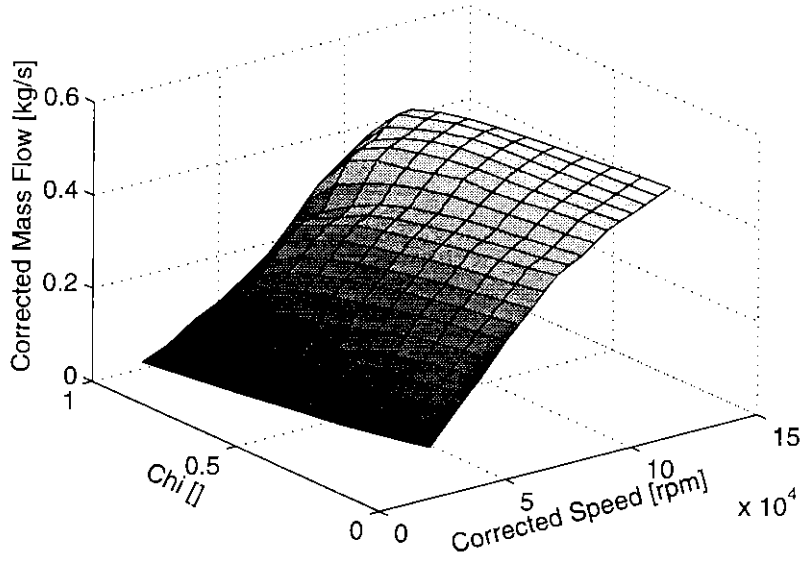

Figure 4.6: Compressor Mass Flow Lookup Table

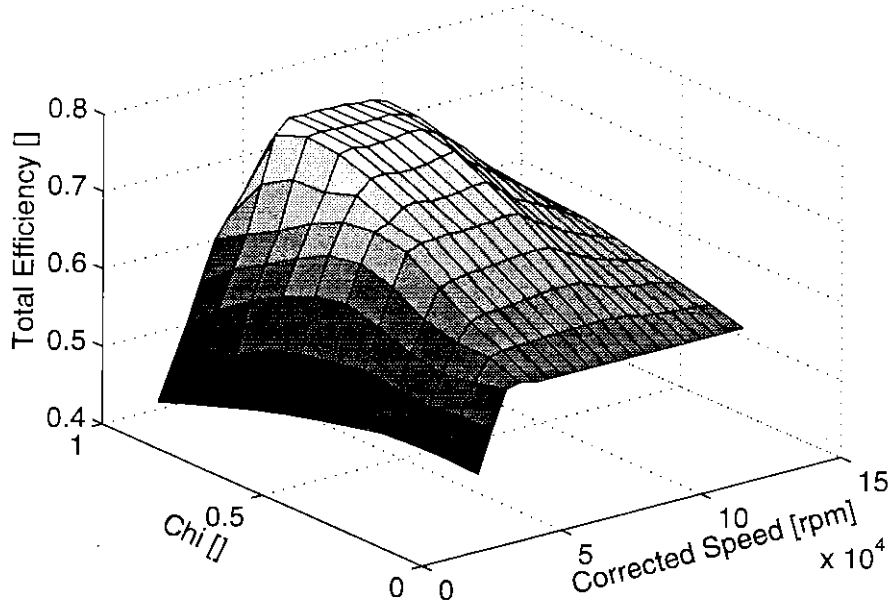

Figure 4.7: Compressor Efficiency Lookup Table

Now that the tables are structured to match the desired causality, an algorithm can be implemented. The algorithm, shown as a flow chart in Figure 4.8, is accurate if ideal gases and minimal kinetic energy effects are assumed.

4.4.2 Radial Flow Turbine

The algorithm for the turbine is similar to that of the compressor, but is simplified due to the fact that the turbine has less complicated characteristics. The turbine data available from turbocharger manufacturers is typically defined by contour maps of the form:

$$N_{corr,turb} = f\left(\dot{m}_{corr,turb}, \frac{P_{01}}{P_{02}}\right) \tag{4.26}$$

$$\eta_{turb} = f\left(\dot{m}_{corr,turb}, \frac{P_{01}}{P_{02}}\right) \tag{4.27}$$

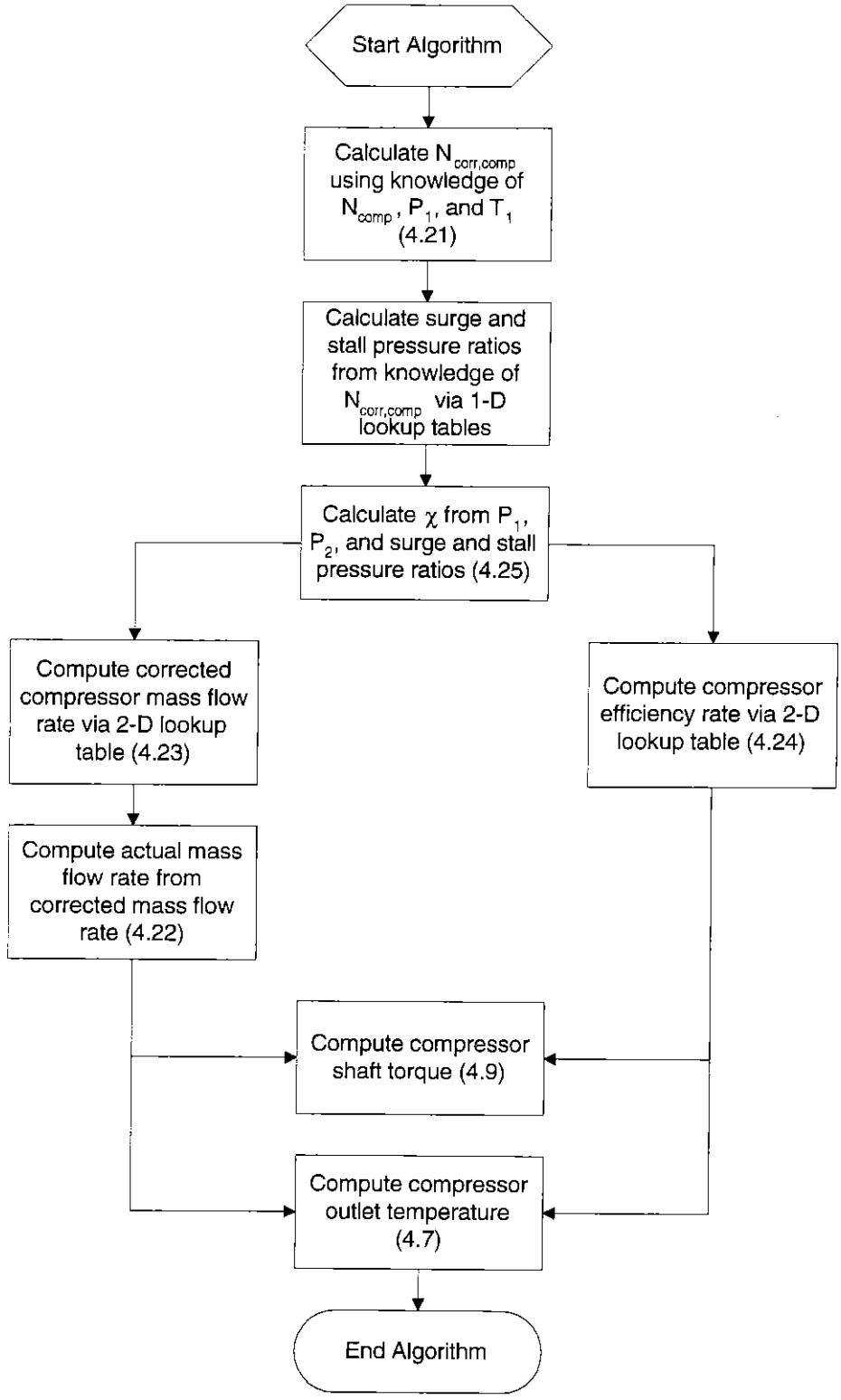

Figure 4.8: Flow Chart for Initial Compressor Model

where

$$N_{corr,turb} = N_{turb} \cdot \sqrt{\frac{T_{std,turb}}{T_{01}}} \tag{4.28}$$

$$\dot{m}_{corr,turb} = \dot{m}_{turb} \cdot \frac{P_{std,turb}}{P_{01}} \sqrt{\frac{T_{01}}{T_{std,turb}}} \tag{4.29}$$

P_{01}, P_{02}, T_{01}, and T_{02} are total pressures and temperatures. If kinetic energy effects are neglected, static pressures and temperatures can be substituted for the total properties. Corrected mass flow and speed are used in the same manner as for the compressor.

The turbine model's inputs and outputs must be defined in such a way as to maintain an explicitly causal relationship between the variables. Like the compressor, the turbine is situated between two filling and emptying manifolds as in Figure 4.9.

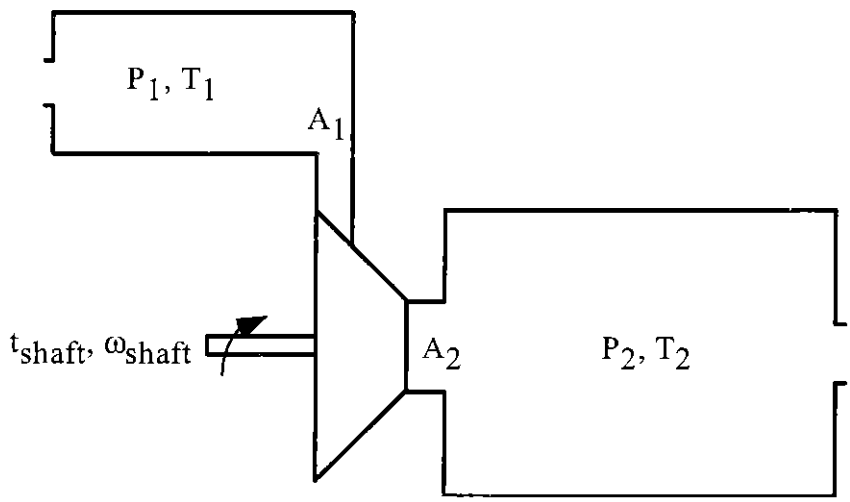

Figure 4.9: Turbine Model Schematic with Nomenclature

Inputs to the turbine thermodynamic model are the same as for the compressor:

- Pressure (1)
- Temperature (1)
- Pressure (2)
- Rotor speed

The outputs calculated by the turbine thermodynamic model are also the same as for the compressor:

- Outlet temperature
- Mass flow rate through the turbine
- Shaft torque

The inputs and outputs described above do not match equations (4.26) and (4.27). Using data analysis software, the data is reformulated into the two following functions:

$$\dot{m}_{corr,turb} = f\left(N_{corr,turb}, \frac{P_{01}}{P_{02}} \right) \tag{4.30}$$

$$\eta_{turb} = f\left(N_{corr,turb}, \frac{P_{01}}{P_{02}} \right) \tag{4.31}$$

Actual surface plots of turbine mass flow and efficiency in the format of equations (4.30) and (4.31) are shown in Figures 4.10 and 4.11. Now that the tables are structured to match the causality, an algorithm can be implemented. The algorithm, shown as a flow chart in Figure 4.12, is accurate if ideal gases and minimal kinetic energy effects are assumed.

4.4.3 Turbocharger Dynamics

Now that the compressor and turbine thermodynamics have been defined and necessary outputs have been calculated, the two components can be connected by a common shaft and inertial dynamics can be examined. The dynamics are computed as:

$$\omega_{comp} = \omega_{turb} = \omega_{shaft} \tag{4.32}$$

$$\left(J_{comp} + J_{turb} + J_{shaft} \right) \cdot \dot{\omega}_{shaft} = t_{turb} - t_{comp} - c\left(\omega_{shaft} \right) \tag{4.33}$$

with a nonlinear damping function included. Now, whenever a torque imbalance occurs between the compressor and turbine, the turbocharger inertia is accelerated or decelerated until a new equilibrium point is reached. The turbocharger inertia, together with the manifold volumes, can have a large effect on the turbocharger lag. Lag is an extremely

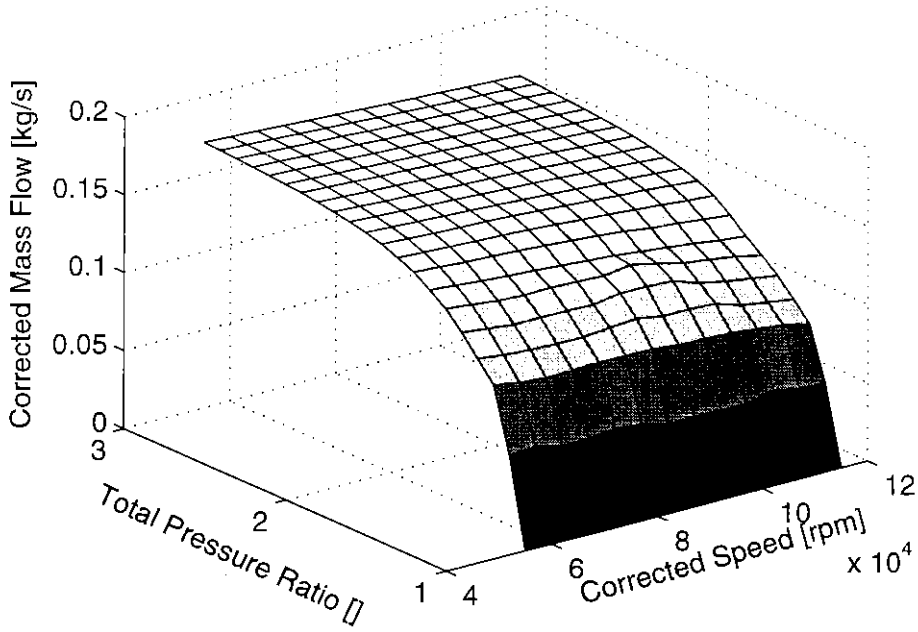

Figure 4.10: Turbine Mass Flow Lookup Table

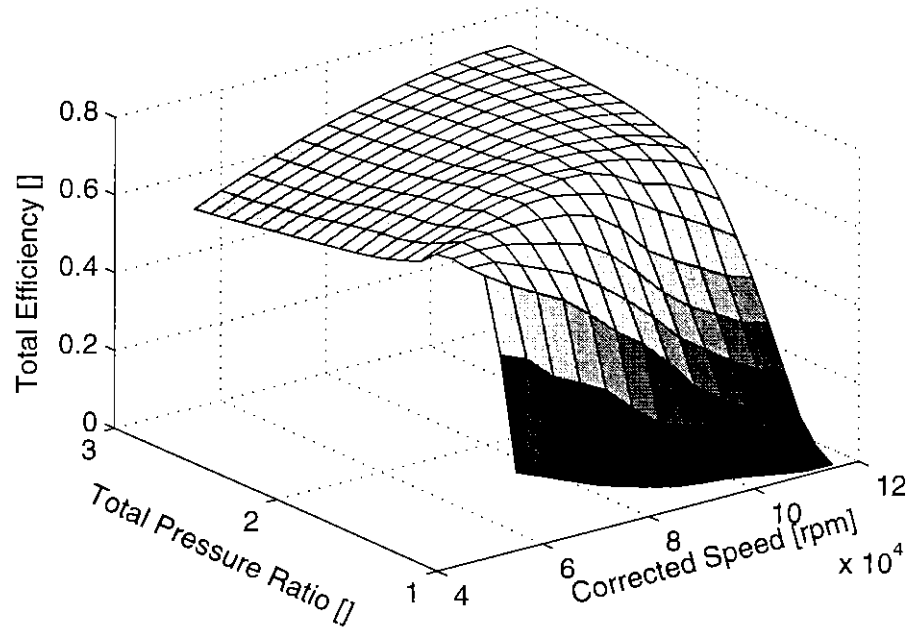

Figure 4.11: Turbine Efficiency Lookup Table

noticeable (and unpleasant) phenomenon to the engine user, and if a simulation can correctly estimate the lag, problematic design concepts can be eliminated before large sums of money are spent on prototypes.

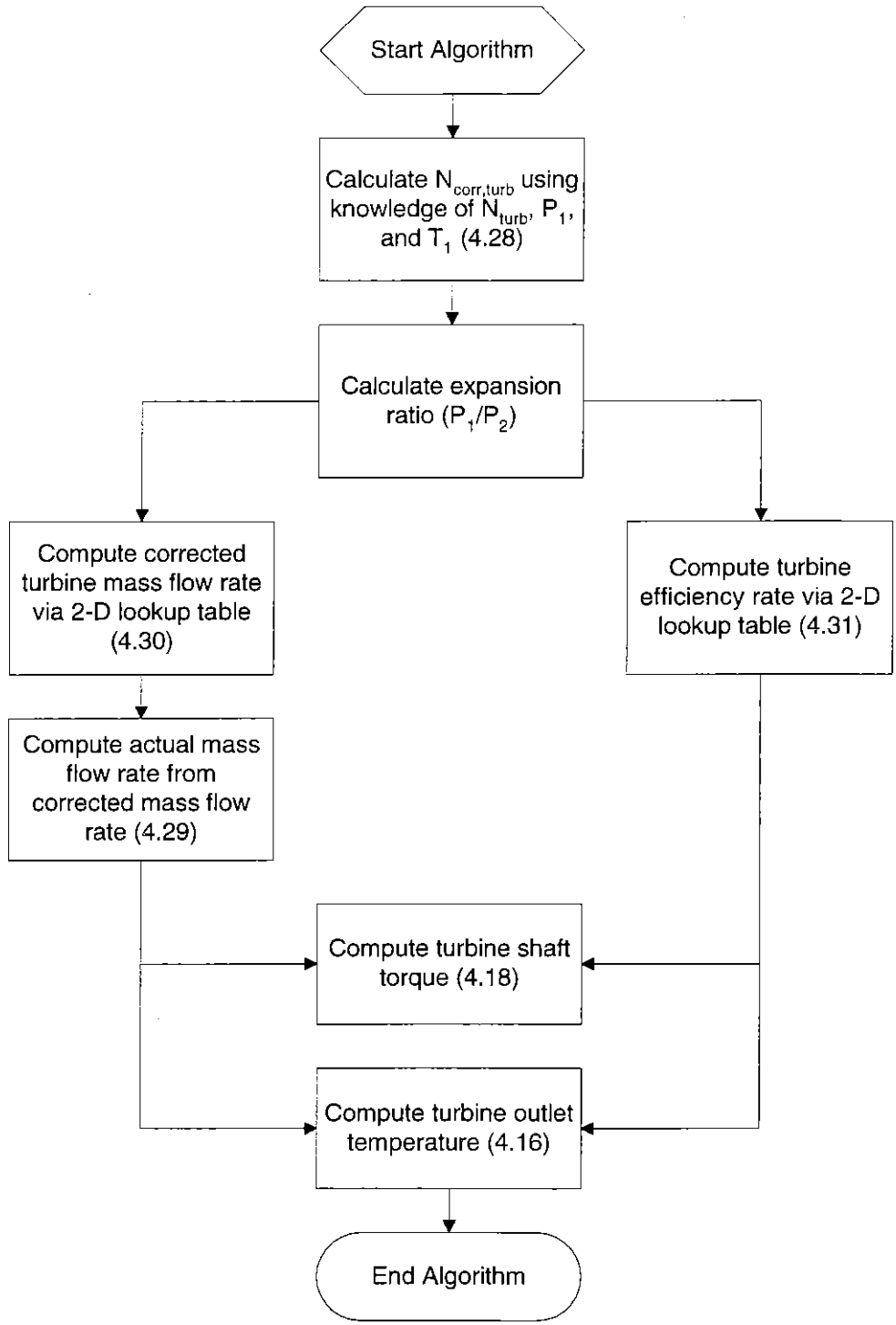

Figure 4.12: Flow Chart for Initial Turbine Model

4.5 Modifications to Initial Turbocharger Modeling Approach

In the sections above, it was assumed that kinetic energy effects are negligible. Unfortunately, this is not always a valid assumption. Often, high flow velocities are encountered at the inlet and outlet ports of the turbine and compressor. Thus, an attempt was made to derive estimates of the kinetic energy effects from the static pressures and temperatures of the filling and emptying manifolds to either side of the compressor and turbine. The following sections will show equation derivations and algorithms implemented to include kinetic energy effects for the compressor and turbine.

First, it is important to define *total* and *static* property values. Static properties values are measured as if the measuring device is moving at the same speed as the fluid. Total property values are measured as if the flow is isentropically decelerated to zero speed [19]. Thus, total property values include kinetic energy effects, which are important if the flow velocity is high. Total temperatures, pressures, and enthalpies will be denoted by a "0" subscript, such as T_{02}. Total property values can be related to the static property values if the Mach number, M, of the flow is known:

$$T_0 = T\left(1+\frac{\gamma-1}{2}M^2\right) \tag{4.34}$$

$$P_0 = P\left(1+\frac{\gamma-1}{2}M^2\right)^{\frac{\gamma}{\gamma-1}} \tag{4.35}$$

If there is a certain mass flow through a pipe of known cross-sectional area, and if the state of the gas is known, a useful expression for the Mach number is:

$$M = \left(\frac{\dot{m}}{PA}\right)\sqrt{\frac{RT}{\gamma}} \tag{4.36}$$

Another useful expression, obtained by rearrangement of (4.36), is:

$$T = \frac{\gamma}{R}\left(\frac{PAM}{\dot{m}}\right)^2 \tag{4.37}$$

The equations shown above, in conjunction with device-specific equations shown below, are the basis for the modified compressor and turbine algorithms. Note that the same causal relationships are used for the basic and modified algorithms. Thus, the modified algorithm block can directly replace the basic algorithm block in simulations.

4.5.1 Centrifugal Flow Compressor

In order to better understand the compressor thermodynamics, it is helpful to see the device's enthalpy-entropy (h-s) diagram, shown in Figure 4.13. States labeled (s) denote results if the gas compression were an isentropic process.

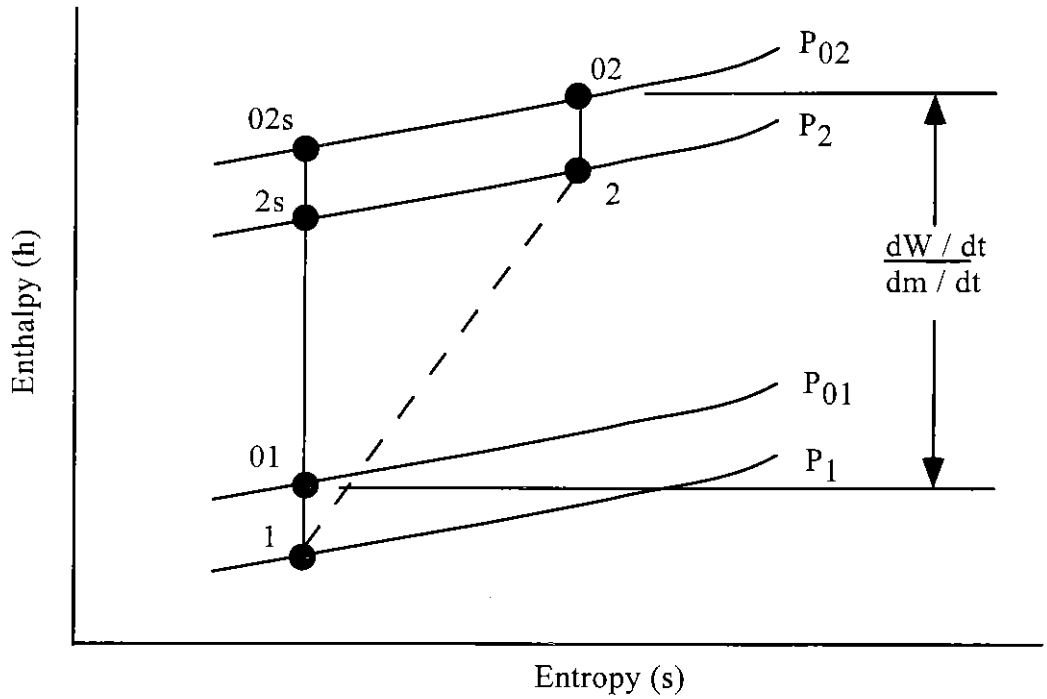

Figure 4.13: Enthalpy-Entropy Diagram for Compressor

From this diagram, it is seen that the compressor efficiency can be expressed in terms of total enthalpies:

$$\eta_{comp} = \frac{h_{02s} - h_{01}}{h_{02} - h_{01}} \tag{4.38}$$

If ideal gases are assumed, C_p is constant and (4.38) can be simplified to:

$$\eta_{comp} = \frac{T_{02s} - T_{01}}{T_{02} - T_{01}} \tag{4.39}$$

Now, it is possible to define an algorithm for the compressor model. The algorithm is iterative, but convergence is obtained with only a few repetitions. A flow chart for the algorithm is shown in Figure 4.14. Note that the algorithm needs an estimate of the inlet and outlet Mach numbers at time=0. At each following time step, the answer from the previous time step is used for the initial guess. Since conditions change very little between time steps, it is extremely unlikely that the initial guesses for Mach numbers could get far enough away from the actual values to cause a convergence problem. In practice, convergence problems have not been seen.

4.5.2 Radial Flow Turbine

An enthalpy-entropy (h-s) diagram for the turbine is shown in Figure 4.15. Again, states labeled (s) indicate results if the expansion process were isentropic.

Similar to equation (4.38), the turbine efficiency can be expressed in terms of total enthalpies:

$$\eta_{turb} = \frac{h_{01} - h_{02}}{h_{01} - h_{02s}} \tag{4.40}$$

Since ideal gases are assumed, (4.40) can be simplified to:

$$\eta_{turb} = \frac{T_{01} - T_{02}}{T_{01} - T_{02s}} \tag{4.41}$$

With these definitions complete, the turbine model algorithm is defined as in Figure 4.16. As for the compressor, no problems have been seen with convergence. In fact, little difference has been found between the basic and modified algorithms for the turbine. The turbine is a less complicated device functionally, and is less sensitive to changes in pressure ratio.

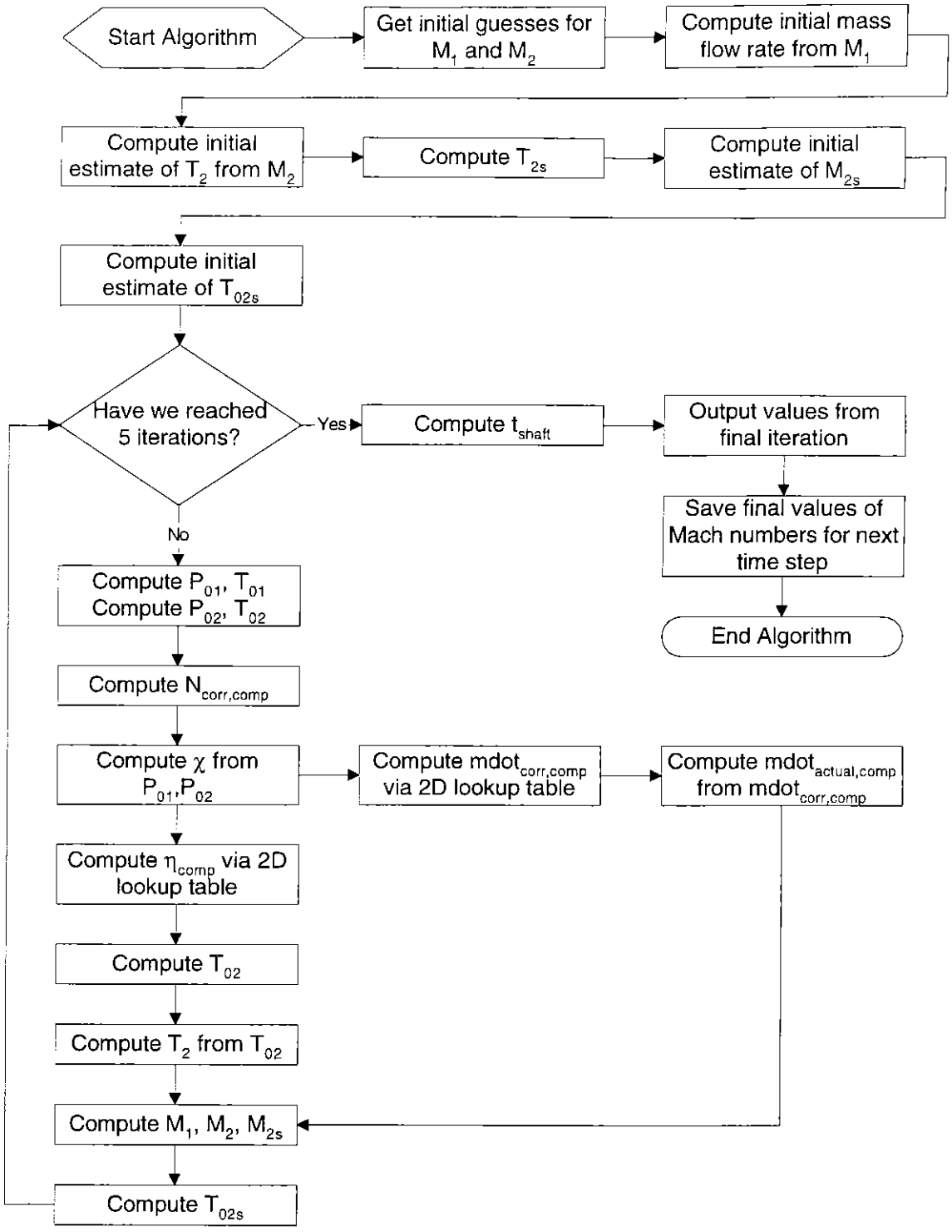

Figure 4.14: Flow Chart for Iterative Compressor Model

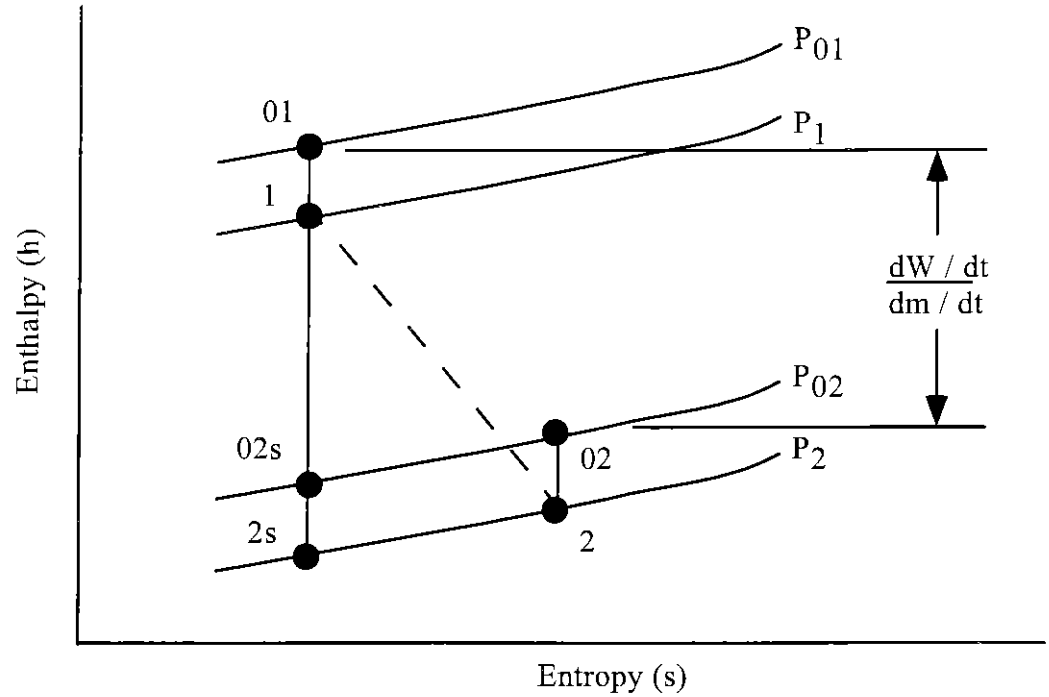

Figure 4.15: Enthalpy-Entropy Diagram for Turbine

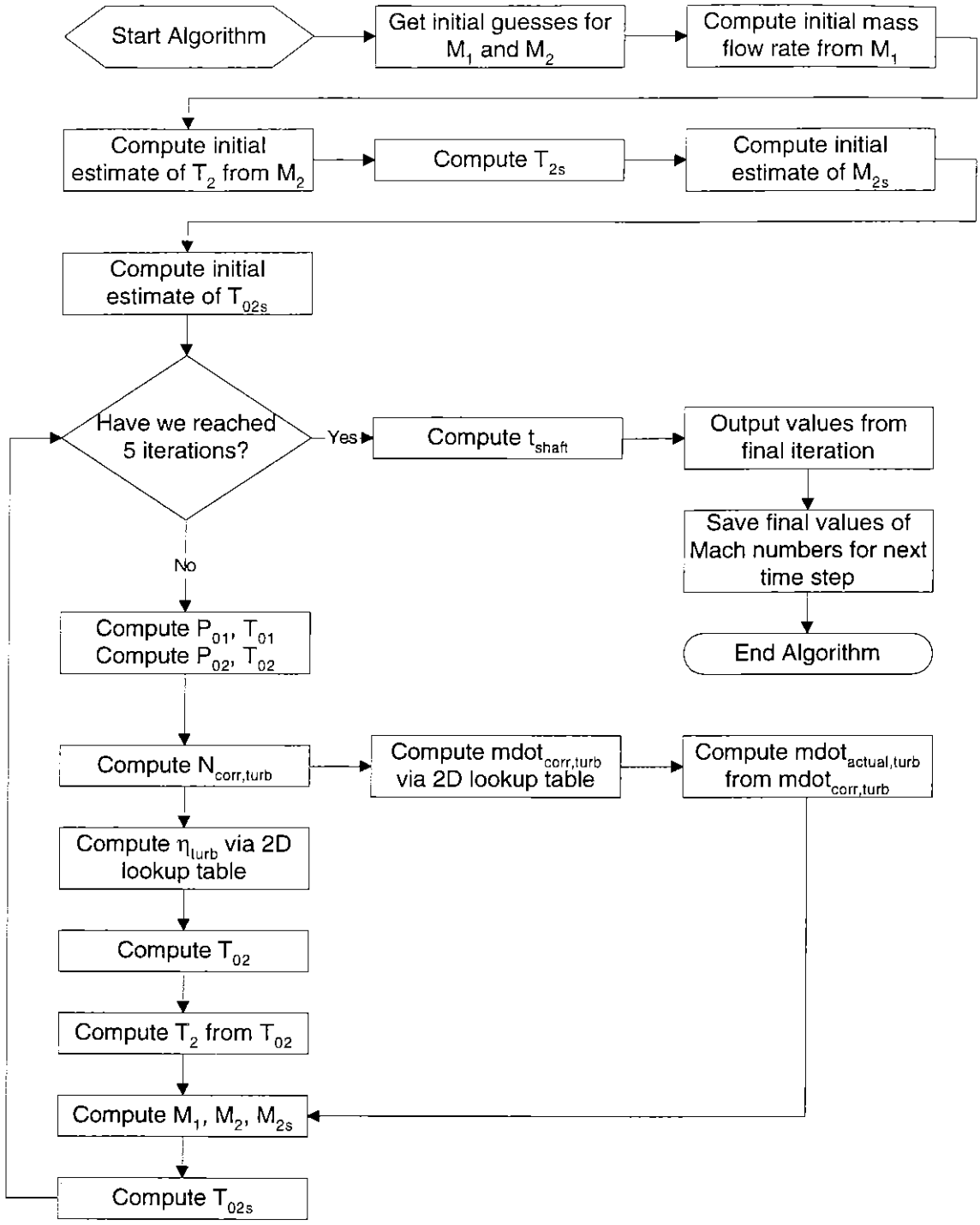

Figure 4.16: Flow Chart for Iterative Turbine Model

4.6 Simulation Results and Comparison of Methods

A benchmark SIMULINK simulation was designed to compare the two turbocharger algorithms. To make the results more realistic, the mean torque turbocharged engine model [11] was connected to a full powertrain system model including vehicle dynamics. Two versions of the engine model were tested, with the only difference being the turbocharger algorithms used. A fuel mass flow rate trajectory was given to the engine model which would accelerate it from a nominal speed of 800 rpm to approximately 2000 rpm. The automatic transmission in the powertrain model was shifted to limit maximum engine speed; shifts occurred at (time=4 seconds), (time=9 seconds), and (time=17 seconds).

Results from this simulation for both algorithms are shown in Figures 4.17-4.24. The non-iterative model is denoted as the "original" model, and the iterative model is denoted as the "improved" model. Clearly, a difference is seen between the original and improved algorithms with respect to turbocharger speed (Figure 4.17). The improved algorithm consistently predicts a lower rotor speed. By plotting χ and turbocharger speed (Figure 4.18), one can get a feel for where the compressor model is operating on the lookup tables for mass flow and efficiency, Figures 4.6-4.7. The improved compressor algorithm has lower values for χ, meaning that it is predicting higher values for corrected mass flow at a given turbocharger speed. This effect was expected from test runs of the iterative compressor algorithm before it was fully implemented in SIMULINK.

Figures 4.19-4.24 show similar small changes between the original and improved models. In a global sense, it can be seen that larger differences between the models occur at higher mass flow rates, where Mach numbers are more significant. As expected, at low pressure ratios and correspondingly low mass flow rates, Mach numbers are less significant and the two algorithms predict virtually the same results. This is an important check on the operation of the improved model. Unfortunately, it is difficult to determine more specific differences between the algorithms from simulations; the compressor, engine, and turbine models are linked together mechanically and thermodynamically, and feedback effects

between the components obscure these details. It may be possible to use experimental test data to "decouple" the engine from the turbine and compressor models, which would make analysis of the algorithms much easier; Section 5.2 discusses this work in more detail.

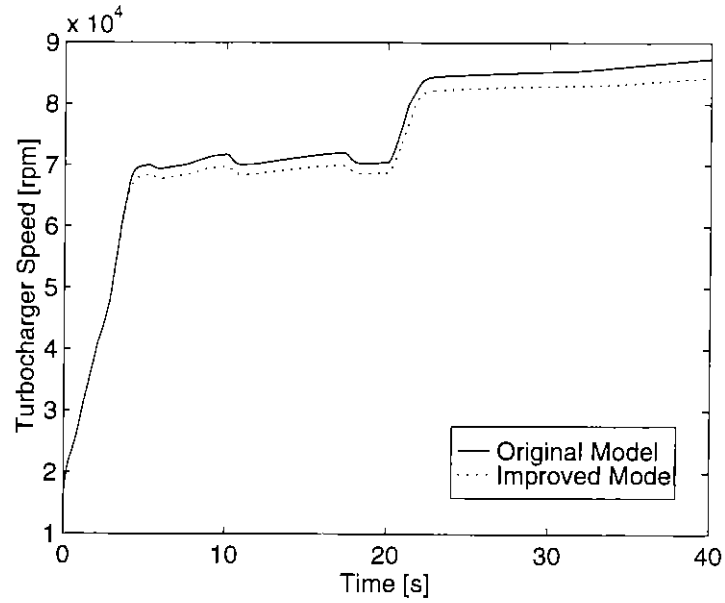

Figure 4.17: Simulation Results - Turbocharger Speed

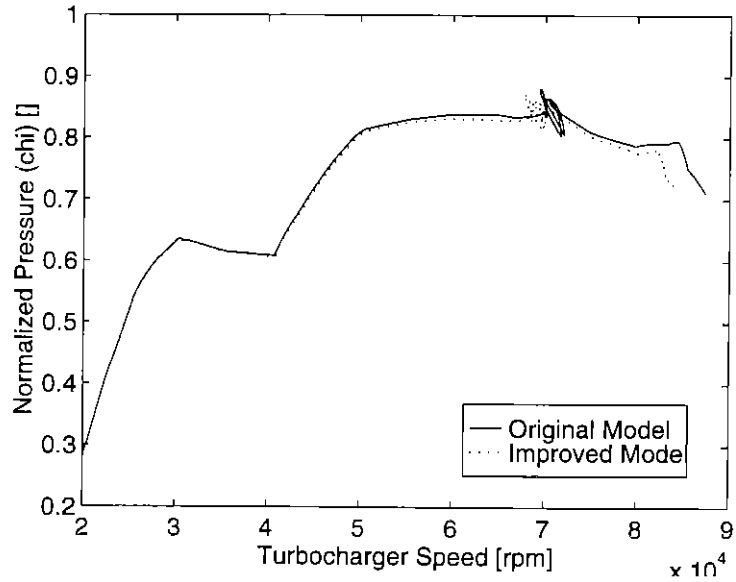

Figure 4.18: Simulation Results - Normalized Pressure

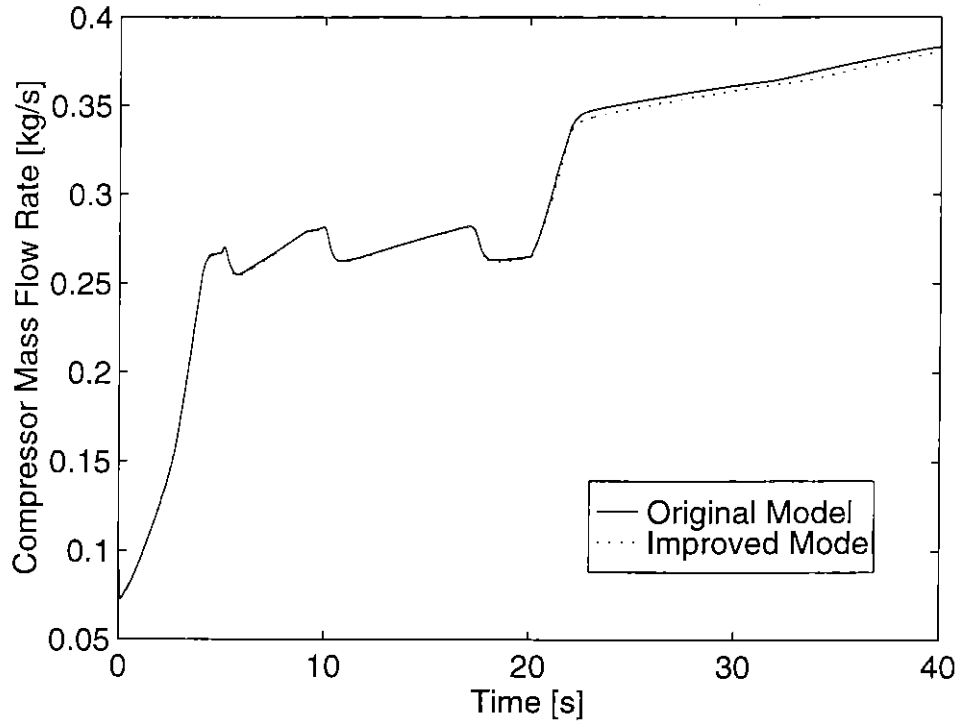

Figure 4.19: Simulation Results - Compressor Mass Flow Rate

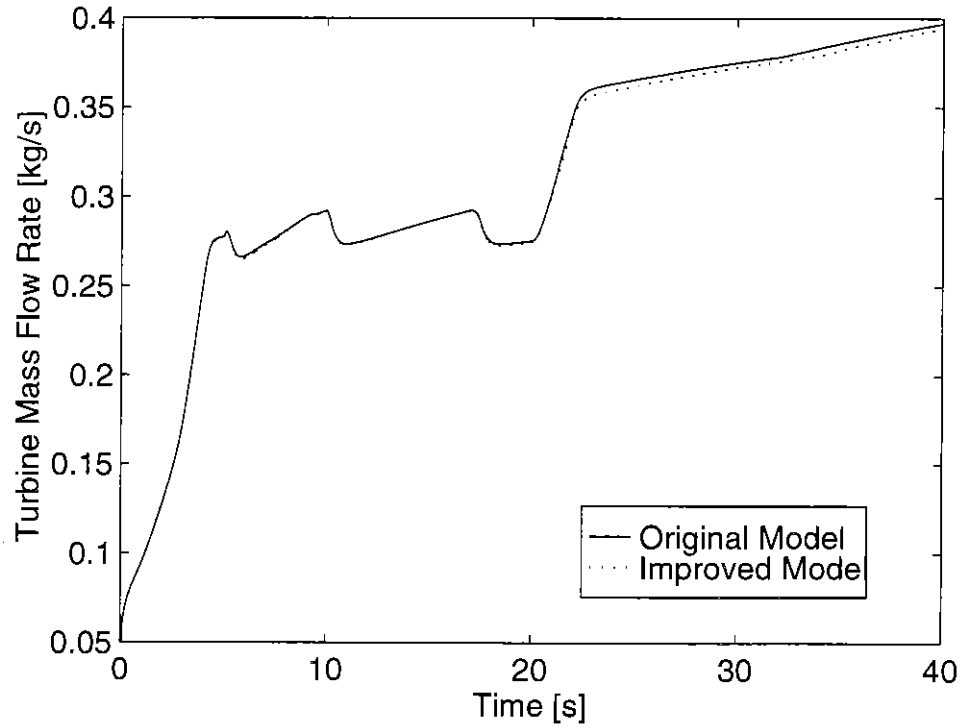

Figure 4.20: Simulation Results - Turbine Mass Flow Rate

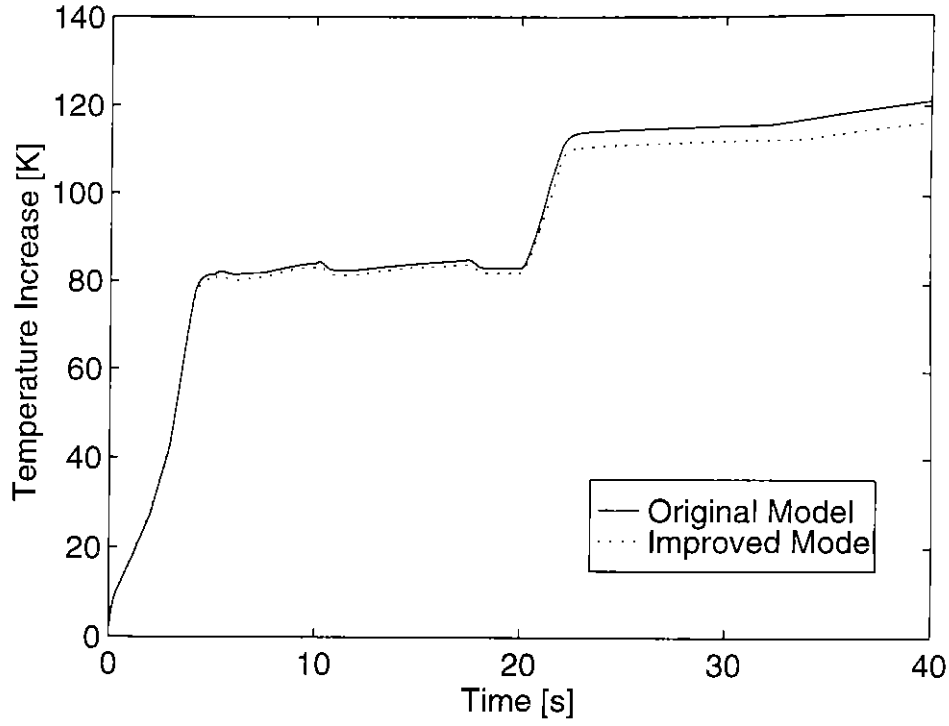

Figure 4.21: Simulation Results - Compressor Gas Temperature Increase

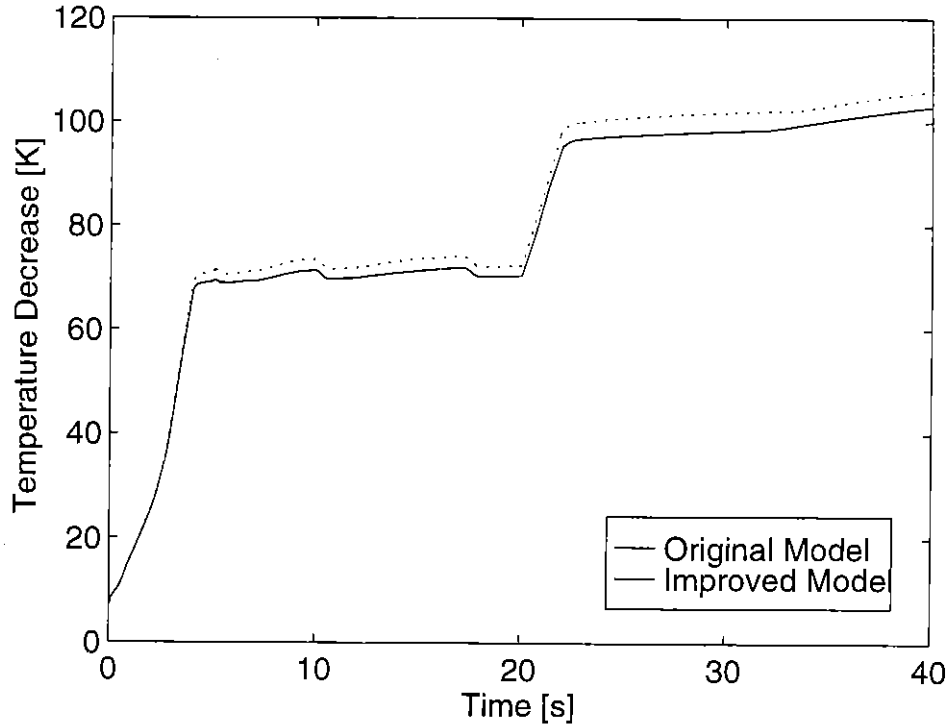

Figure 4.22: Simulation Results - Turbine Gas Temperature Decrease

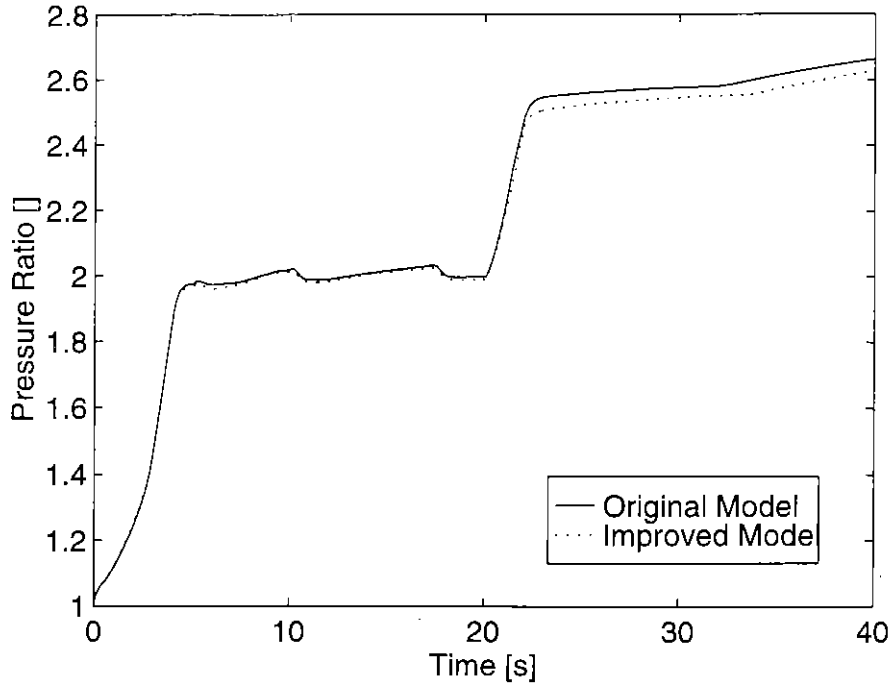

Figure 4.23: Simulation Results - Compressor Pressure Ratio

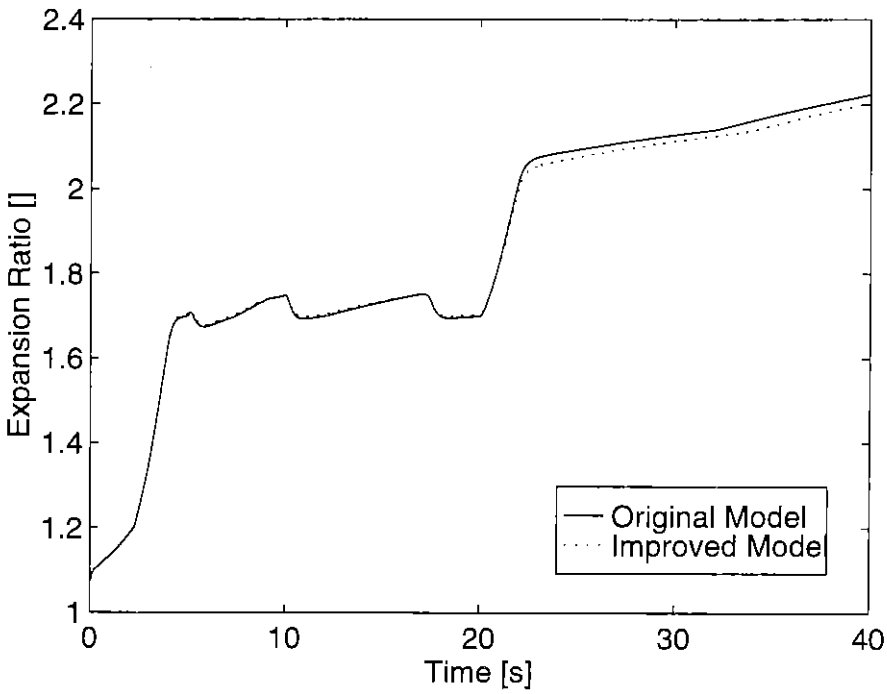

Figure 4.24: Simulation Results - Turbine Expansion Ratio

Chapter 5: Conclusions

5.1 Summary

A powertrain simulation was implemented in SIMULINK to help engineers predict the transient, system-level performance of a powertrain. Design of this large simulation required careful planning and adherence to methodologies which enhanced simulation usability and understandability. Concepts such as hierarchy, flexibility, modularity, and causality were explained; when implemented, these concepts guide the simulation developer into an intuitive, physics-based realization of the nonlinear ordinary differential equations which define the system behavior. Additional features, such as initialization files and component pre-processors, significantly reduced user workload and were utilized whenever feasible.

Methods for automatic transmission model design were outlined. The methods were based on the use of linear algebraic relationships, and resulted in the generation of state space matrices for each different mode of operation of the transmission. Sequential logic in the form of state diagrams was utilized to make decisions for changing the dynamic equations. Two example transmissions were evaluated. The first, a simple two-speed transmission, was used to explain the methods and terminology commonly used when discussing automatic transmissions. The second transmission was the Allison HT-740 automatic transmission, used in the U. S. Army's M915A2 and M916A1 vehicles. Simulation results for the Allison HT-740 were used to show that automatic transmission models can be used to optimize clutch pressure trajectories and eliminate clutch bindup.

Two algorithms for the modeling of turbochargers were explained. Both algorithms were based on the quasi-steady assumption, and used lookup tables generated from manufacturer data to predict the turbocharger behavior. The first algorithm was relatively simple, and neglected kinetic energy effects of the gas flowing through the compressor and turbine. The second algorithm accounted for the kinetic energy effects, but had to solve a system of nonlinear algebraic equations by iteration. The two algorithms were tested in benchmark

simulations, and the results were shown to be very similar. However, differences were noted when Mach numbers increased, showing that the second algorithm may have advantages in regions of the performance map requiring large mass flow rates.

5.2 Future Work

It is necessary to obtain some transient experimental data for verification of assumptions about turbocharger performance. First, it is important to verify the validity of the quasi-steady assumption, or show that the turbocharger responds almost instantaneously to changes in its environment. Second, it is important to take a closer look at the two turbocharger algorithms and determine any necessary modifications to them. Acquiring useful test data is not a trivial task. Since results must be obtained for transient conditions, all sensors must have fast response times. While high bandwidth thermocouples and pressure transducers are fairly easy to find and implement, it is difficult to find mass flow rate sensors which provide accurate results at the bandwidth necessary for transient turbocharger tests. However, attempts at acquiring this data will be made soon.

Additional research needs to be done with respect to determining values for the compressor lookup tables at low rotor speeds and pressure ratios; typically, experimental data from manufacturers is not available in this regime, and the maps must be estimated. It may be possible to obtain computer code from turbocharger manufacturers to perform the predictions off-line in a pre-processor. This would signficantly improve the accuracy of the turbocharger model at low engine speeds and loads.

With respect to automatic transmission modeling, significant contributions may yet be made. A prevalent source of difficulty is human error while generating the algebraic equations solved to get the state space matrices. However, with the numerical approach shown, it is quite feasible to write software which reads in a graphical or textual description of the transmission mechanism and automatically formulates the state space matrices. This would decrease errors in the modeling process and ease the workload of the modeler

significantly. Some programs already exist for this purpose (e. g., ENPORT for bond graphs). For the broader powertrain model additional elements will be added of varying fidelity, depending upon design needs.

Bibliography

[1] Assanis, D. N. and Heywood, J. B. "Development and Use of a Computer Simulation of the Turbocompounded Diesel System for Engine Performance and Component Heat Transfer Studies." Society of Automotive Engineers Paper 860329.

[2] Beachley, N. H. and Harrison, H. L. Introduction to Dynamic System Analysis. New York: Harper and Row, 1978.

[3] Berglund, Sixten. "Comparison Between Measured and Simulated Transients of a Turbocharged Diesel Engine." Society of Automotive Engineers Paper 942323.

[4] Bollinger, J. G. and Duffie, N. A. Computer Control of Machines and Processes. New York: Addison-Wesley, 1988.

[5] Carey, D. M. Private Communications, July 1994. Cummins Engine Company, Inc.

[6] El-Gammal, A. M. "An Algorithm and Criteria for Compressor Characteristics Real Time Modeling and Approximation." ASME Journal of Engineering for Gas Turbines and Power **113**, January 1991.

[7] Haessig, D. A. Jr. and Friedland, B. "On the Modeling and Simulation of Friction." ASME Journal of Dynamic Systems, Measurement, and Control **113**, September 1991.

[8] Jamzadeh, F. et. al. "Dynamic Simulation Modeling for Heavy Duty Automatic Transmission Control Development." Society of Automotive Engineers Paper 922441.

[9] Jensen, J.-P. et. al. "Mean Value Modeling of a Small Turbocharged Diesel Engine." Society of Automotive Engineers Paper 910070.

[10] Kao, M. Model-Based Turbocharged Diesel Engine Control and Diagnostics Using Nonlinear Sliding Control and Observers. Ph. D. Thesis, University of Wisconsin-Madison, 1994.

[11] Kao, M. and Moskwa, J. J. "Turbocharged Diesel Engine Modeling for Nonlinear Control and State Estimation." ASME Journal of Dynamic Systems, Measurement, and Control **117/1**, March 1995.

[12] Kuo, B. C. Automatic Control Systems, 6th ed. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[13] Karnopp, D. "Computer Simulation of Stick-Slip Friction in Mechanical Dynamic Systems." American Society of Mechanical Engineers, Journal of Dynamic Systems and Control **107**, March 1985.

[14] Karnopp, D. et. al. System Dynamics: A Unified Approach, 2nd ed. New York: John Wiley & Sons, Inc., 1990.

[15] Ledger, J. D. and Walmsley, S. "Computer Simulation of a Turbocharged Diesel Engine Operating Under Transient Load Conditions." Society of Automotive Engineers Paper 710177.

[16] The Math Works, Inc. SIMULINK User's Guide, 1994.

[17] Myers, G. E. Engineering Thermodynamics. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[18] Pan, C. H. and Moskwa, J. J. "Dynamic Modeling and Simulation of the Ford AOD Automatic Transmission." Society of Automotive Engineers Paper 950899.

[19] Roberson, J. A. and Crowe, C. T. Engineering Fluid Mechanics, 4th ed. Boston: Houghton Mifflin, 1990.

[20] Runde, J. K. Modelling and Control of an Automatic Transmission. M. S. Thesis, Massachusetts Institute of Technology, 1986.

[21] Streit, E. E. and Borman, G. L. "Mathematical Simulation of a Large Turbocharged Two-Stroke Diesel Engine." Society of Automotive Engineers Paper 710176.

[22] Winchell, F. J. and Route, W. D. "Ratio Changing the Passenger Car Automatic Transmission." Society of Automotive Engineers Paper 610407.

[23] Winn, K. R. et. al. "Turbocharger Development: A Look at the Future." ICE - Vol. 22, Heavy Duty Engines: A Look at the Future. American Society of Mechanical Engineers, 1994.

[24] Winterbone, D. E. et. al. "A Wholly Dynamic Model of a Turbocharged Diesel Engine for Transfer Function Evaluation." Society of Automotive Engineers Paper 770124.

APPROVED

Adviser Signature: .....

Adviser Title: Assoc. Prof. of M. E.

Date: 5-15-96