**Modeling and Simulation of Series Parallel Hybrid Electrical Vehicle (SPHEV)**

By

TAQIUDDIN BIN MOHD NASIR

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronic Engineering)

DECEMBER 2006

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

**CERTIFICATION OF APPROVAL**

**Modeling and Simulation of Series Parallel Hybrid Electrical Vehicle (SPHEV)**

By

TAQIUDDIN BIN MOHD NASIR

A project dissertation submitted to the
Electrical & Electronic Engineering Program
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL & ELECTRONIC ENGINEERING)

Approved by,

Supervisor,
Miss. Noor Hazrin Hany Bt. Mohd Hanif

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
DECEMBER 2006

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in references and acknowledgements and that the original works contained herein have not been undertaken or done by unspecified sources or persons.

TAQIUDDIN BIN MOHD NASIR

## ACKNOWLEDGEMENT

First of all, Alhamdulillah for the completion of this final year project entitled Modeling and Simulation of Series Parallel Hybrid Electrical Vehicle (SPHEV). I would like to express my appreciation to my supervisor, Miss Noor Hazrin Hany Mohd Hanif for giving me opportunity and trust to handle this project. A big thank you also goes to my co-supervisor, Mr.Syaifuddin Mohamad for his support, idea and guidance which are really helpful throughout the semester. A big thank you for the help and tips from TECHSOURCE stuffs especially Mr.Ong Weng Yung. I also would like to thank all the technicians of EE Department especially Mdm.Siti Hawa, Mr.Azhar, Mr.Isnani and Mr.Musa for their assistance during the completion of this project. My appreciation also goes to other lecturers of EE department, UTP stuff which are very supportive throughout the semester. Finally, I would like to give my biggest token of appreciation to my parent especially Mr.Ahmad Sabri Mohamad and Puan. Rahana Mat Hussin as they are the most supportive and understanding persons when I'm having a lot of difficulties during my completion of this project.

# ABSTRACT

The project was about modeling and simulation of Series Parallel Hybrid Electrical Vehicle using the MATLAB/Simulink as the based software. Based on the vehicle's real specification and right components sizing, a SPHEV Simulink model was developed and simulated to observe and study the operations of SPHEV under various driving conditions. The model developed was a forward looking model utilizing the driver model controller to request for power demand. The controller would execute the power demand and choose which mode to be used. Since the SPHEV was a combination of series and parallel modes, a good control strategy was required to ensure the vehicle was operating at the right mode depends on the driving situation. The simulation of SPHEV model was based on Urban Dynamometer Driving Schedule (UDDS) reference drive cycle and it was successfully simulated.

xPC Target as a powerful tool is used to rapidly implementing real time control system on a DC Motor. The xPC Target control system consist of three major components which are Host PC, Target PC and hardware. The motor is used as hardware of xPC Target as well as to represent the SPHEV system. The motor controller which has been designed in Host PC is compiled and downloaded to the Target PC which is running in real time operating system. The operating system will execute the controller and control the hardware. Variation of motor's load which also corresponds to variations of driving range in the SPHEV system will contribute to different speed of motor. The acquired speed is used by the controller to compensate with errors occur during the implementation.

TABLE OF CONTENTS

## LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 Background

Automobiles are an integral part of our everyday lives. Yet, conventional automobiles are the major cause of urban pollution in the $21^{st}$ century. The world will eventually encounter an acute energy crisis if we do not focus on alternatives energy sources and transportation mode. Current environmental concerns are motivating factor that lead to development of low emission (hybrid vehicles) or zero emission (electric) vehicles to replace the current conventional internal combustion engine vehicles. The subject of hybrid vehicles is becoming increasing important with intense drive from government, environmental activist and associated industries to develop the hybrid technologies. Several auto industries led by Japanese car manufacture's company Toyota have already started marketing the hybrid vehicles. The next generation of conventional automobiles will experience a gradual replacement of hydraulically driven actuator by electrically driven actuators. [1]

The term 'hybrid vehicle' implies the bringing together of two forms of vehicle propulsion, normally mechanical and electrical drawing of the best features of each to produce a vehicle which is called as Hybrid Electric Vehicle (HEV) [2]. HEVs have an electric motor as well as the internal combustion engine (ICE) to provide extended range and to 'curve down' the pollution problem. There are two hybrid configurations which are series and parallel. Detailed of both configurations will be explained in Chapter 2.

## 1.2 Problem Statement

The focus of the first semester of Final Year Project is to develop and simulate forward looking model of SPHEV. Utilizing the MATLAB/Simulink software, the model is developed based on the real specification of the components used in the SPHEV. The specifications of the components such as Internal Combustion Engine (ICE), Electric Motor, Generator and Battery which contribute to a complete SPHEV model are available in MATLAB/ADVISOR software. Simulation of non-real time SPHEV Simulink model is done for the purpose of implementing the suitable control strategy for the model. The principle of the control strategy is to optimize the operational efficiency of all components considered as one SPHEV system. Modeling and Simulation of SPHEV is targeted on various driving pattern and parameters to optimize the power needed in all condition.

Real-time rapid control prototyping using xPC Target will be the main concern for the second semester of Final Year Project. The SPHEV Simulink model is modified to interface with prototype hardware or actual plant which will be a DC motor. The simulation of SPHEV in real-time will be done using the xPC Target implementing the DC Motor as the replacement to the vehicle. The motor is chosen because of its availability and small specification since the xPC Target only allow for less specification hardware.

## 1.3 Objectives

The objectives of the Final Year Project can be divided into two parts which is the first semester of Final Year Project and second Semester of Final Year Project. The objectives for the first semester are:

- To determine the vehicle's components specifications based on the desired vehicle parameters which is typically a sedan sized Series Parallel HEV.
- To develop a forward looking model of Series Parallel Hybrid Electrical Vehicle based on the real specifications of the components in the system.
- To simulate the Series Parallel Hybrid Electrical Vehicle in non Real-Time application in various driving pattern.
- To apply a good control strategy to the system in order to have better operational efficiency of all the components.

For the second semester, the objectives are:

- To re-modify the Simulink SPHEV model in order to meet the requirements of xPC Target.
- To implement real-time rapid control prototyping using xPC Target utilizing the DC Motor as the hardware.
- To test the controller in real-time application.

## 1.4 Scope of Study

The scopes of the study during the whole Final Year Project can be described as below:

- Study behavior of Series Parallel Hybrid Electric Vehicle (SPHEV) under various operating conditions and drive cycles.
- Identify the best parameters of the components that will optimize the system performance.
- Develop the Simulink forward looking model of SPHEV and simulate the model in various operating conditions.
- Design and development of an optimal control strategy for the energy management system of SPHEV.
- Implement the real-time rapid control prototyping using the xPC Target.

# CHAPTER 2

# LITERATURE REVIEW

The chapter will explain about the Hybrid type presently together with its advantages and disadvantage, the Hybrid drive train components and also the MATLAB based software that is used through out the project. The software uses for this project are ADVISOR, Simulink, Real-Time Workshop and xPC Target. The ADVISOR is used as a source of components specification such as ICE and Motor while Simulink is an environment that is utilized to develop the SPHEV model. Real-Time Workshop can generate and compiles source code from Simulink models to create real-time software application. The xPC Target is an environment for rapid control prototyping in real-time application.

## 2.1 Types of Hybrid

The basic configurations of Hybrid Electrical Vehicles are series and parallel.

### 2.1.1 Series Hybrid

Series HEV is a configuration that has only one energy converter to provide the propulsion power to the drive train. The engine acts as prime mover to drive an electric generator which can either deliver power to the battery or drive the electric motor. The electric motor provides all the propulsion power .The components arrangement is shown in Figure 1. [3][6]

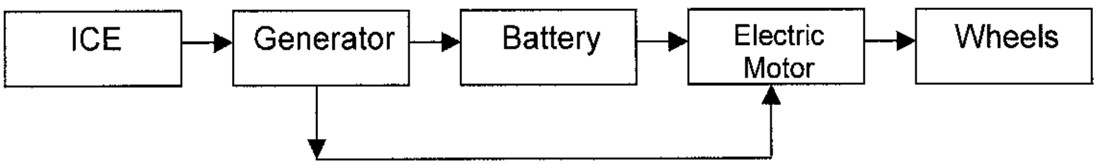



Figure 1: Series Hybrid System

The advantages and disadvantages of series hybrid are summarized in the following. The advantages of series hybrid are:

- Flexibility of location of engine-generator set.
- Simplicity of drivetrain.
- Suitability for short trips.

The disadvantages of series hybrid are:

- It needs three propulsion components: ICE, generator and motor.
- The motor must be designed for maximum sustained power that the vehicle may required such as when climbing a high grade. However, most of the time the vehicles operates below the maximum power.
- The battery will exhaust fairly quickly, leaving the ICE to supply all the power through the generator all the time.

### 2.1.2 Parallel Hybrid

A parallel HEV has more than one energy sources that can provide the propulsion power to the drive train. The ICE in this system is mechanically connected to the transmission system. The battery is added parallel to the ICE to provide the electrical energy to the electric motor and supplement the torque of ICE. So, the propulsion power in this configuration may be supplied by the heat engine, by the battery-motor set or by both systems in combination. The configuration is shown in Figure 2. [3]

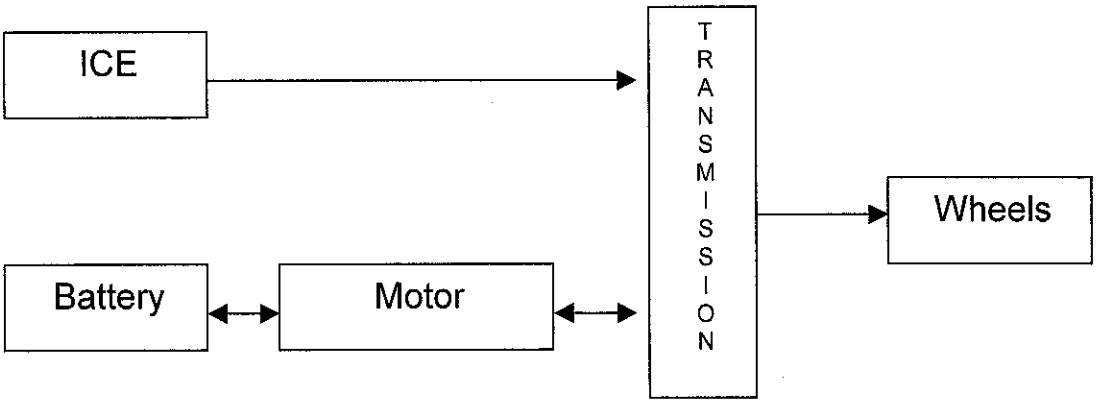



Figure 2: Parallel Hybrid System

The following are the advantages of parallel hybrid:

- It needs only two propulsion components: ICE and motor/generator. In parallel motor can be used as generator or vice versa.
- The combination of ICE and motor to give the propulsion power reduced the engine and motor sized. Since, both of the components can be rated at either half the maximum power or full maximum power depending on driving situation.

The disadvantages of parallel hybrid are:

- The control complexity increased significantly because power flow has to be regulated and blended from two parallel sources.
- The power blending from the ICE and the motor drive to a complex mechanical device.

### 2.1.3 Series-Parallel Hybrid

The series parallel components configuration is shown in Figure 3. The schematic is based on Toyota Prius hybrid design. The system features the characteristics of both Series and Parallel Hybrid system. The ICE mechanical power can directly drive the transmission system (parallel path) or being split by Power Split Device (PSD) into the generator (series path) depending on the driving conditions. The power through the generator will either provide the electrical energy to the motor or charge the battery through the Power Control Unit (PCU). The motor can also deliver power to the transmission system. The inverter is bidirectional and is used to charge the batteries from the generator or to condition the power for electric motor. [4]

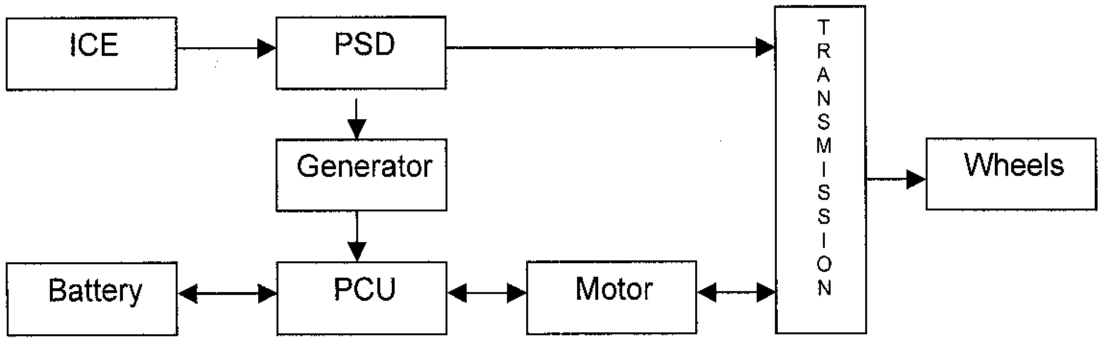



Figure 3: Series Parallel Hybrid System

The advantages of Series Parallel configuration are as follows:

- Both the series and parallel characteristics can be used depending on the driving situations. For example, the series elements ensure that the battery remains charged in prolonged period such as at traffic light or in traffic jam while the parallel elements will be useful during acceleration mode where more power is needed.
- Engine efficiency is optimized by using the generator. At certain driving speed, the engine will operate inefficiently which lead to high consumptions of fuel. So, the used of generator will help to optimized the engine efficiency thus reduce the fuel consumptions.
- The batteries act more as a power sources rather than energy sources. Thus the battery weight can be reduced significantly.

The disadvantages of Series Parallel hybrid are as follows:

- Energy is lost during the process of charging and discharging off battery and during power flow through the inverter.
- More complexity as all the propulsion components are needed: ICE, motor, and generator. Furthermore, inverter and power split device components are added to provide smooth operation conditions.

However, research of SPHEV is good to be done since it offer both of hybrid configuration.

## 2.2 Drive Train Components

The drive train components are the components which make up the whole Hybrid system. A Hybrid system can be different to one another depending on each component. It can be either a Series Hybrid or Parallel Hybrid that obviously depends on the position of a Electric Motor. The components that form a Hybrid system are Internal Combustion Engine (ICE), Electric Motor, Generator, Battery and Power Split Device (PSD).

### 2.2.1. Internal Combustion Engine (ICE)

The internal combustion engine is a heat engine in which fuel's burning occurs in a combustion compartment. The fuel's burning creates gases of high temperature and pressure, which need to be expanded. The expanding hot gases will act directly to movement part, for example by acting on pistons and rotors and moving the entire engine itself.

The engine is used for base loading power. It is required to consistently supply the steady state power requirements which correspond to rolling resistance and aerodynamic drag. Aerodynamic drag is referred as air resistance which flowing around the forward moving vehicle. It creates the air turbulence at the back of the car which will generate air drag. Low aerodynamic vehicle will cause more power to move it on. Rolling resistance is produced by the hysteresis of the tire at the contact surface with the roadway. Thus, the total demanded power by the vehicle is expressed as the following equation (1):

$$P_{required} = 0.5\rho C_d A V^3 + mgC_{rr}\cos\theta V + mg\sin\theta V + maV \qquad (1)$$

where

| | |
|---|---|
| p | air density (1.23 kg $m^{-3}$ ) |
| Cd | coefficient of drag of the vehicle in the direction of travel |
| Af | frontal Area (cross-sectional area of the vehicle) |
| V | Velocity (m/s) |
| M | Car mass |
| g | gravitational force |
| $C_{rr}$ | Coefficient of rolling resistance between the tires and the road |
| Θ | angle of inclination of the road surface upon which the vehicle is traveling |
| a | acceleration of the vehicle |

### 2.2.2 Electric Motor

An electric motor converts electrical energy into kinetic energy. The fundamental principle upon which electromagnetic motors are based is that there is a mechanical force on any current-carrying wire contained within a magnetic field. The force is described by the Lorentz force law and is perpendicular to both the wire and the magnetic field. Most magnetic motors are rotary, but linear motors also exist. In a rotary motor, the rotating part (usually on the inside) is called the rotor, and the stationary part is called the stator. The rotor rotates because the wires and magnetic field are set so that a torque is developed about the rotor's axis. The motor contains electromagnets that are wound on a frame which is called as armature. The armature is that part of the motor across which the input voltage is supplied.

### 2.2.3. Electrical Generator

The term 'generator' is used when power flow is in the opposite direction, with the machine converting mechanical energy into electrical energy. In the Series Parallel HEV configuration, the generator is used as a path to transfer the output power of the engine to either charging the battery or run the motor. [1]

### 2.2.4. Battery

A basic requirement for HEV is a source of electrical energy which is converted to mechanical energy in the electric motor for vehicle propulsion. Electrical energy is typically obtained through conversion of chemical energy stored in devices such as batteries. The battery pack should be capable of delivering the motor peak power and it also need to keep its State of Charge (SoC) within a specific limit in order to give maximum energy at every drive cycles. The SoC is the present capacity of the battery. It is the amount of capacity that remains after discharge from a top charge condition. [1]

### 2.2.5. Power Split Device (PSD)

The power split devices (shown in Figure 4) splits the engine power of the drive train system to two directions, the transmission systems and as the input to the generator. The power split device uses a planetary gear. The rotational shaft of the planetary carrier inside the gear mechanism is directly linked to the engine, and transmits the drive power to the outer ring gear and the inner sun gear via pinion gears. The rotational shaft of the ring gear is directly linked to the motor and transmits the drive force to the wheels, while the rotational shaft of the sun gear is directly linked to the generator. [2]

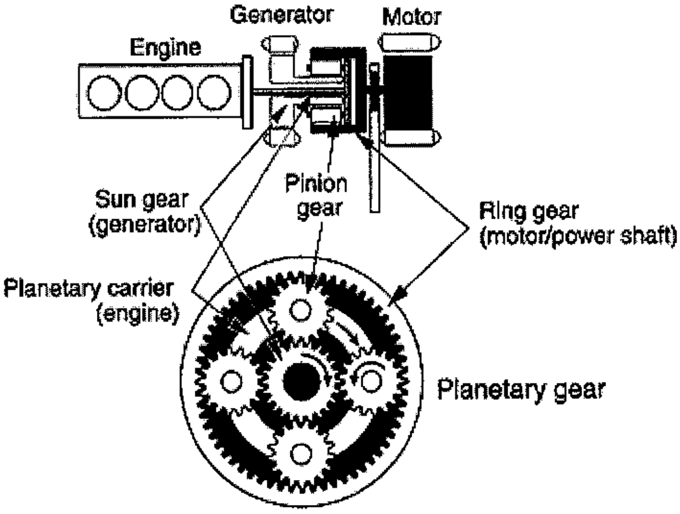



Figure 4: Power Split Device

### 2.3 MATLAB Based Software

The softwares used through out the project are ADVISOR and Simulink. The other such as Real-Time Workshop and xPC Target provide the environment for rapid control prototyping.

#### 2.3.1 Advanced Vehicle Simulator (ADVISOR)

ADVISOR 2.1 is the latest software version of National Renewable Energy Laboratory's advanced vehicle simulator. It is develop to support the US Department of Energy hybrid propulsion system program and is designed to be precise, fast, flexible, easily sharable and user friendly. It is the main source to obtain several components specifications which then will be used to model the system. Figure 5 below shows the vehicle input screen of the ADVISOR.

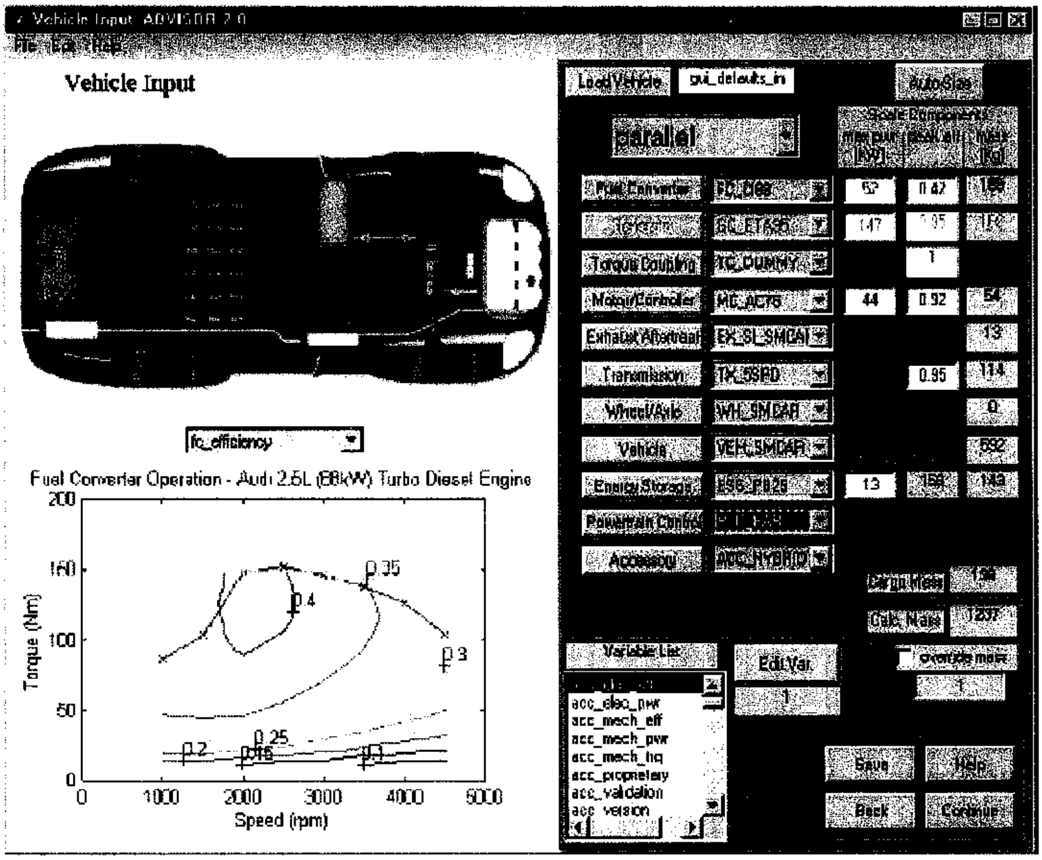


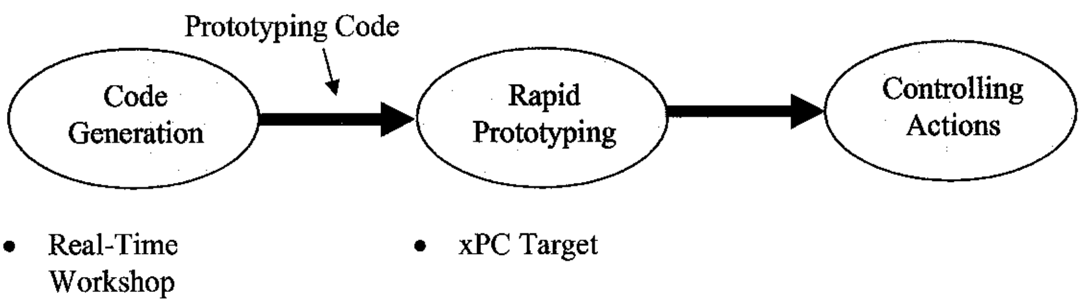

Figure 5: ADVISOR vehicle input screen

### 2.3.2 Simulink

Simulink is a software package for modeling, simulating and analyzing dynamic systems. It supports linear and nonlinear systems modeled in continuous time, sampled time or combination of both. For modeling, Simulink provide a graphical user interface (GUI) for building models as block diagrams, using click and drag mouse operations. After the model is defined, it can be simulate using a number of integration methods either from Simulink menus or by entering the commands in the MATLAB command windows. The simulation result can be put in the MATLAB workspace for post processing and visualization.

### 2.3.3 Real-Time Workshop

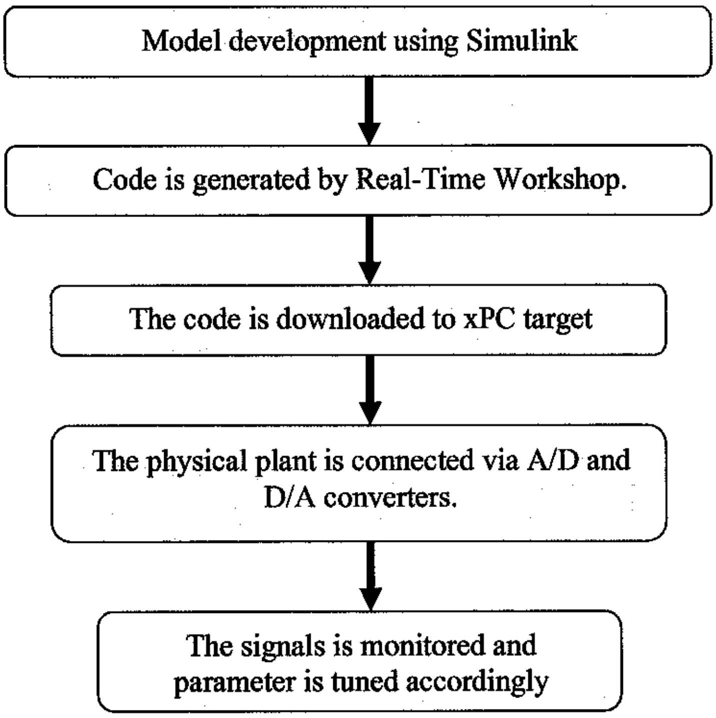
Real-Time Workshop is an extension capability of Simulink and MATLAB that automatically generates packages and compiles source code from Simulink models to create real-time software applications. One of the real-Time Workshop application areas is rapid prototyping applications. [9]




The Real-Time Workshop provides:

- Automatic code generation tailored for a variety of target platforms.
- A rapid and direct path from system design to implementation
- Seamless integration with MATLAB and Simulink.
- A simple graphical user interface.
- An open architecture and extensible make process.

Rapid prototyping is a method for developing and testing an algorithm with actual hardware with minimal effort. The process of rapid prototyping can be simplified as follows:



### 2.3.4 xPC Target

xPC Target provides a high performance, host-target prototyping environment that enables the connection between the Simulink models to the physical systems and execute them in real-time on PC compatible hardware. The Host PC will be a standard usual PC which runs the MATLAB, Simulink and Real-Time Workshop while the Target PC will be an industrial computer which is xPC TargetBox. [8]

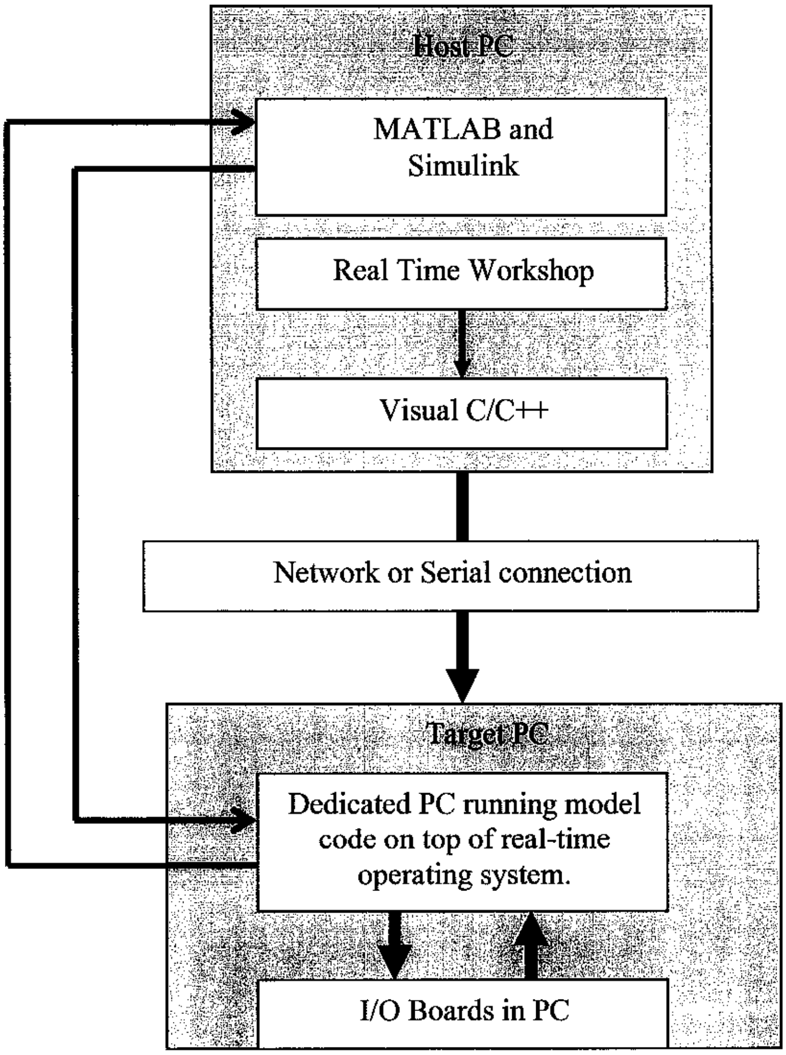



Figure 6: Block Diagram of Host PC and Target PC

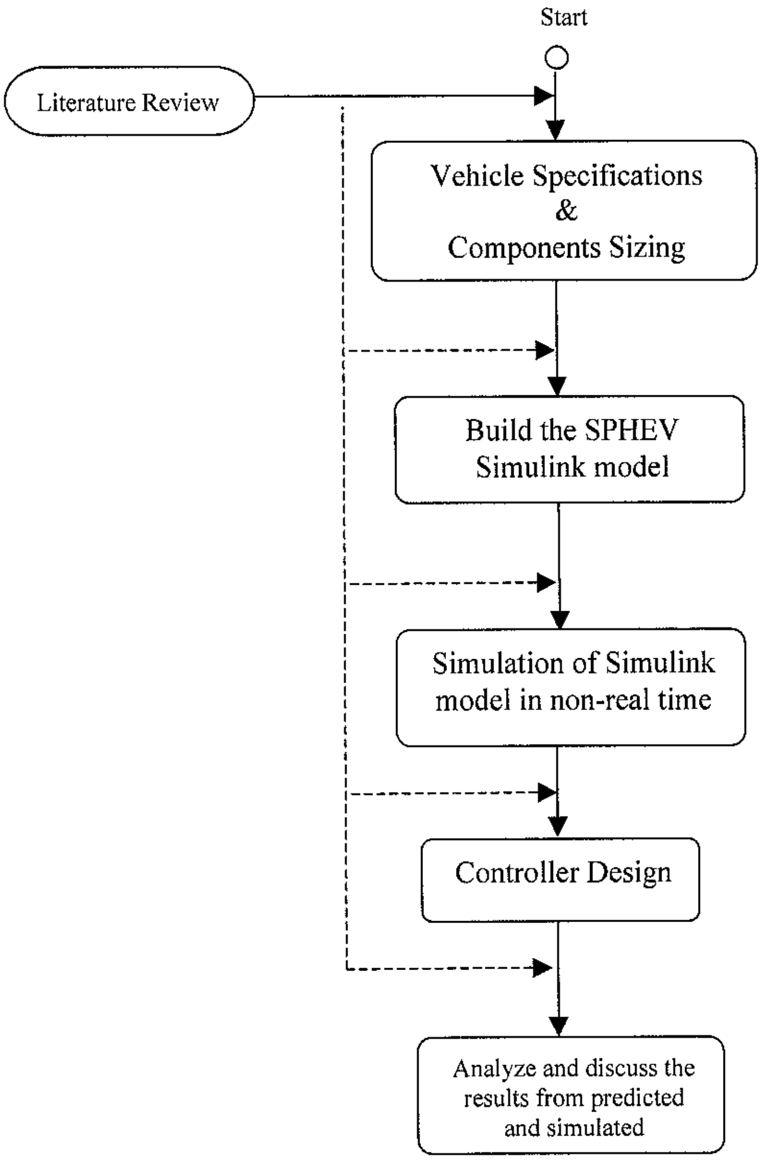
# CHAPTER 3

# METHODOLOGY

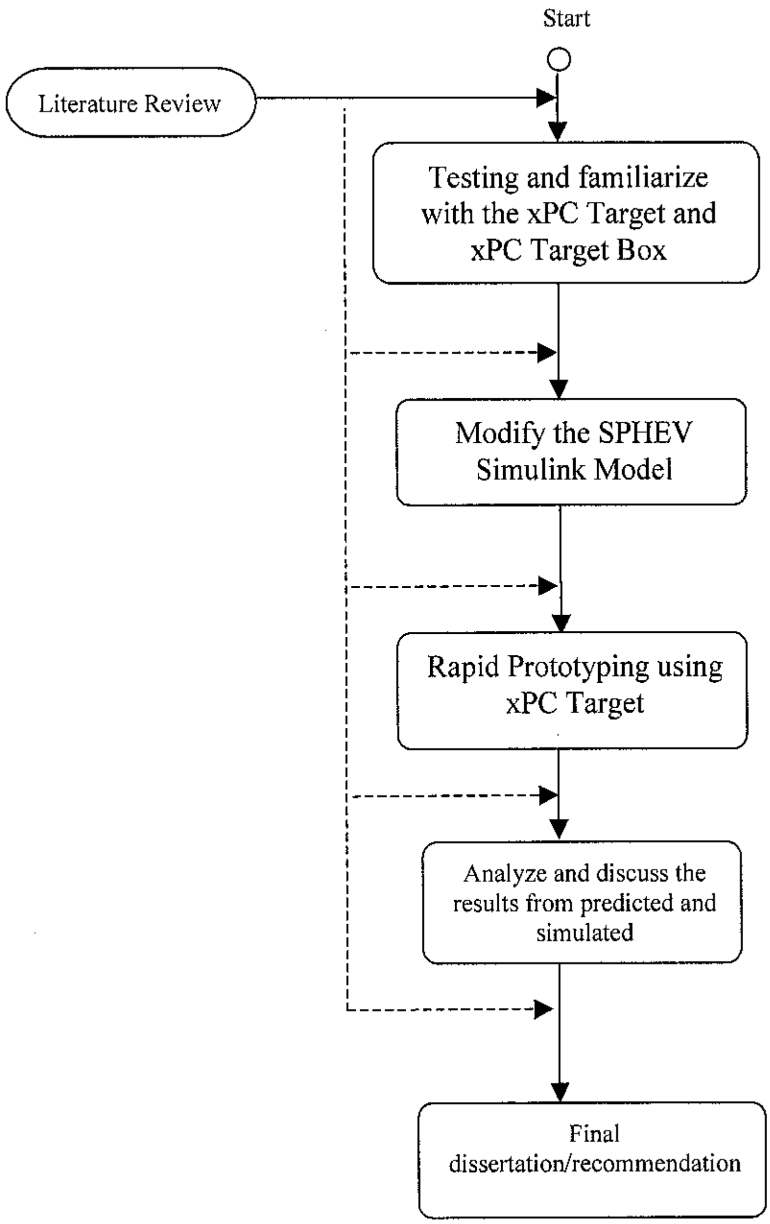
A lot of steps or methods need to be accomplished in order to successfully complete the Modeling and Simulation of Series Parallel HEV. The first half of the project, the objective was to build a SPHEV Simulink model and simulate it in offline. The model built was based on the real specifications of the vehicle and the components. A control strategy was developed in order to control the energy management system in the SPHEV drive train system.

On the second half of the project, focus is given on real-time rapid control prototyping using xPC Target. The Simulink model of SPHEV is modified to run in real-time, on dedicated PC using xPC target. A controller will be developed so that it can control the simulation in real-time application utilizing the DC Motor as the hardware for rapid prototyping. More realistic control can be experienced since the controller can be tested during the simulation.

## Final Year Project 1 Methodology



**Final Year Project II Methodology**



### 3.1 Vehicle Specifications

The sizing of the components of electrical system and mechanical system starts once the driving train architecture is laid out based on the parameters of the vehicle. The drive train architecture and control technique for the HEV depends on the desired requirements, including but not limited to performance, range and emission. The performance requirements of initial acceleration, cruising velocity, maximum velocity and grad ability dictate the design of power and energy requirements of the engine and motor. Development of the drive train will be directed towards developing the energy efficiency and cost effective propulsion system. But, automobiles performance is not depend on the drive train alone but also on the driving patterns such as journey conditions and driving behavior of the driver.

User governed specifications and ecological issues are the factors of designing the HEV. User governed specifications means the vehicle's performance mostly depends on consumer's demand. Acceleration performance, maximum speed and fuel consumptions are kind of examples that consumers are always demanding on. This kind of specifications dictates the sizing of the vehicles components such as the engine, electric motor and transmission systems. The second category is based on ecological issues whereby the manufacturer has to follow the regulations that have been standardize throughout the whole globe by environmental associations such as Environmental Protection Agency (EPA) and California Air Regulatory Board (CARB). Most car manufacturer chooses to follow both of the standardized specifications which is developing a very reliable, good performance car without effecting the environmental.

### 3.1.1 Road Load Power Requirements

The vehicles drive train parameters need to be determined first before proceeding to do the components sizing. The vehicle's parameters are based on a standard sedan size car in Malaysia. Based on the desired vehicle's parameter which is shown in Table 1, energy and power requirements of the vehicle can measured and calculated from a set of dynamic equation (1). Table 2 shows the energy and power requirement of the drive train.

| | |
|---|---|
| Top Speed | 160km/h |
| Acceleration | 0 – 100km/h in 15 sec |
| EV Range | 75 – 100 km |
| Grade ability | 7% at 90km/h |
| Mass (M) | 1250 kg |
| $C_D$ | 0.3 |
| $A_f$ | 2 $m^2$ |
| $C_{rr}$ | 0.01 |

Table 1: Vehicles Parameters and Specification

| Driving Conditions | Power Requirement (kW) |
|---|---|
| Cruising at 60 km/h | 3.8 |
| Cruising at 60 km/h at 7% grade | 18.1 |
| Cruising at 90km/h | 8.9 |
| Cruising at 90km/h at 7% grade | 30.4 |
| Cruising at 110 km/h | 14.4 |
| Cruising at 110 km/h at 7% grade | 40.6 |
| Top Speed (160km/h) | 38.4 |
| 0-100 km/h acceleration in 15 sec | 80.25 |

Table 2: Energy and Power Requirement

The graph shown in Figure 7 below is developed from the power requirements calculated in Table 2.

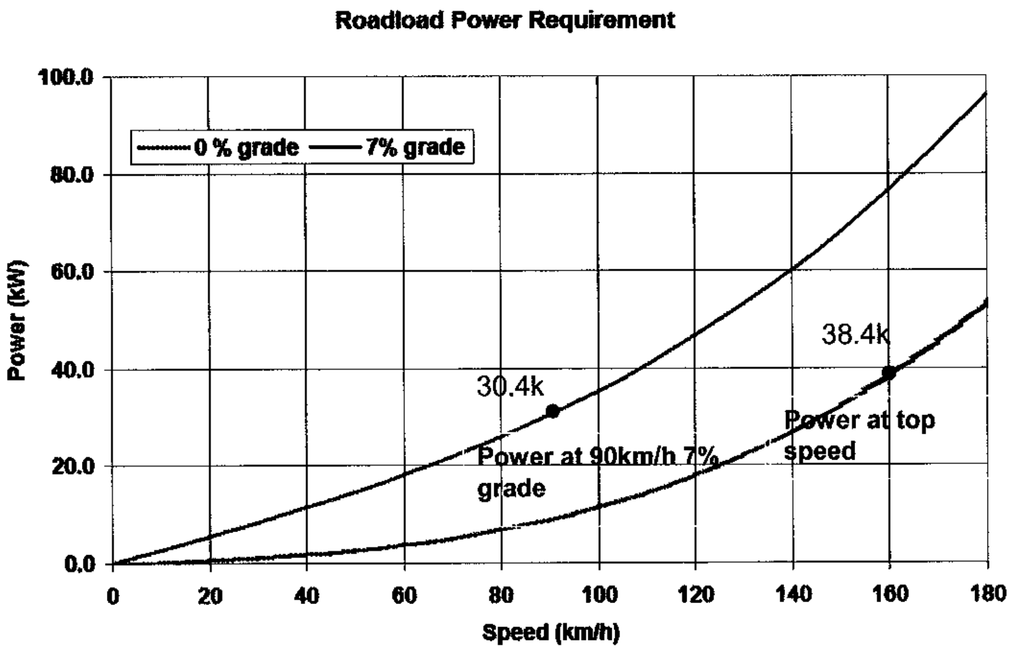



Figure 7: Road Load Power Requirements graph

### 3.2 Components Sizing

Based on the calculated road load power requirements, the size and specifications of the components could be determined and compared to the components available in the MATLAB/Advisor software. The component which is nearer to the specifications will be taken to be modeled.

#### 3.2.1 Engine Power Sizing

The engine is required to supply the steady state power requirements to the propulsion system. Required power can be calculated for different driving conditions with the given formulae in (1) applied. IC engine delivers the power for cruising at top speed which is at 160km/h with power required 38.4kW. So, in the MATLAB/Advisor software, the available engine which is within the needed specifications is:

**GEO 1.0 L (41kW) SI Engine**
**Max Power - 41kW at 5700 rpm**
**Peak Torque 81Nm 3477 rpm**

#### 3.2.2 Electric Motor

The electric motor primarily serves to meet the acceleration requirements. The motor chosen is Permanent Magnet (PM) Synchronous motor. During acceleration, the required power is 80.5kW. Since the motor is used to meet acceleration, about 39kW is needed in order for the vehicle to accelerate. So, from data acquisition in MATLAB/Advisor, electric motor specifications will be:

**Honda 49kW PM Motor**
**Torque range -176.4 N-m to 274.4 N-m**
**Speed ranges 0 to 8500 rpm.**

### 3.2.3 Electric Generator

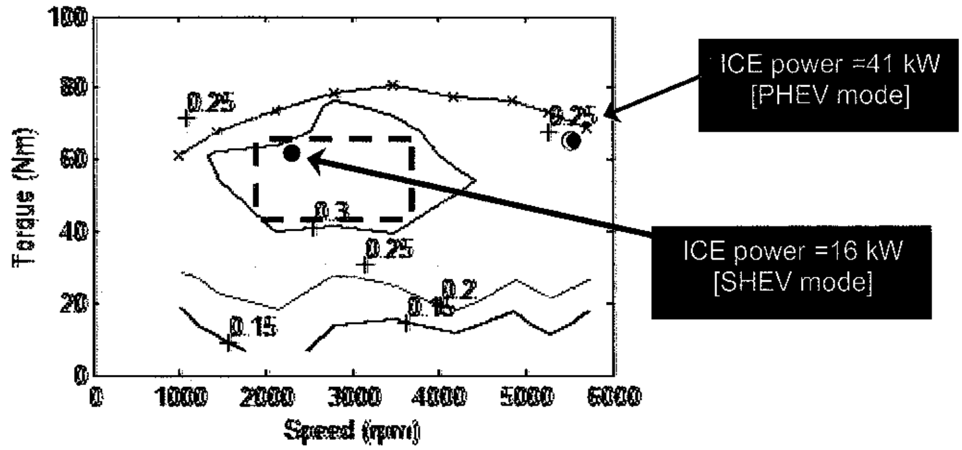



Figure 8: Performance characteristics of 41kW engine.

Generator sizing will be based on engine optimization during Series HEV mode. The maximum speed during Series HEV mode is at 110km/h which required 14.4kW. From the equation:

$$P_{electrical} = \eta P_{mechanical}$$

The generator efficiency ($\eta$) is assumed at 90% and $P_{electrical}$ 14.4kW, $P_{mechanical}$ will give about 16kW. This power is needed in order to drive the generator and supplied the power needed during Series HEV mode. So, from the MATLAB/Advisor software, the generator specification available is:

**32kW PM Generator**

**Torque range 0 to 520 N-m**

**Speeds range 0 to 7000 rpm.**

### 3.2.4 Battery Sizing

The battery will be based on Electrical Vehicle (EV) Mode which the conditions will be:

**Minimum EV range at 70km at 90km/h**

From the calculation, battery capacity is at 9.92kW/h which can give constant supply to above conditions. Maximum efficiency of the battery is obtained if battery's SOC is kept between 0.3 and 0.9 where the battery will be Lead Acid battery.

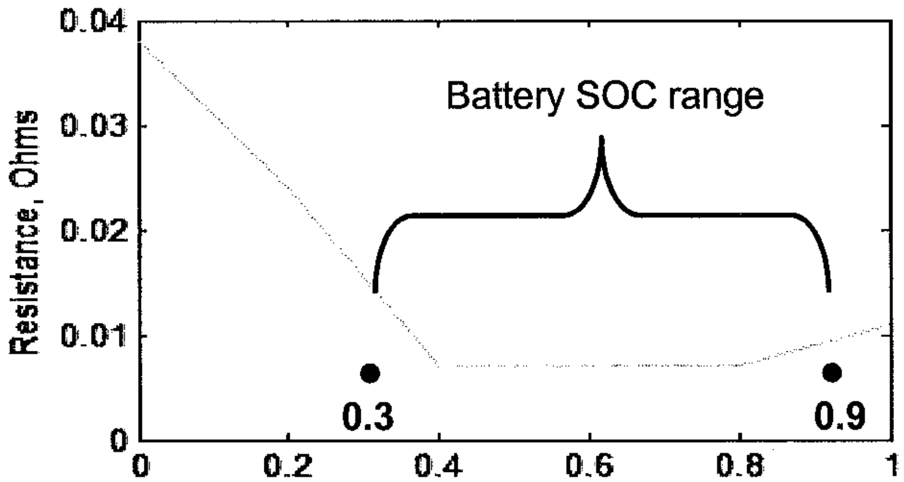



Figure 9: Battery State of Charge vs Resistance

### 3.3 Building the SPHEV Simulink Model

The SPHEV Simulink model is developed based on the Hybrid Electrical Vehicle (HEV) backward looking model which is available in MATLAB/Advisor software. Based on the backward looking model shown in Figure 10 below, a forward looking model shown in Figure 11 is built. The difference between a backward looking model and forward looking model is that the backward looking model requires no model of driver behavior. Without the driver model, the force that is required to accelerate the vehicle through the time step is calculated directly from the required speed trace. On the other hand, simulation of vehicle using the forward looking model approach includes the driver model which considered the required speed and the present speed.

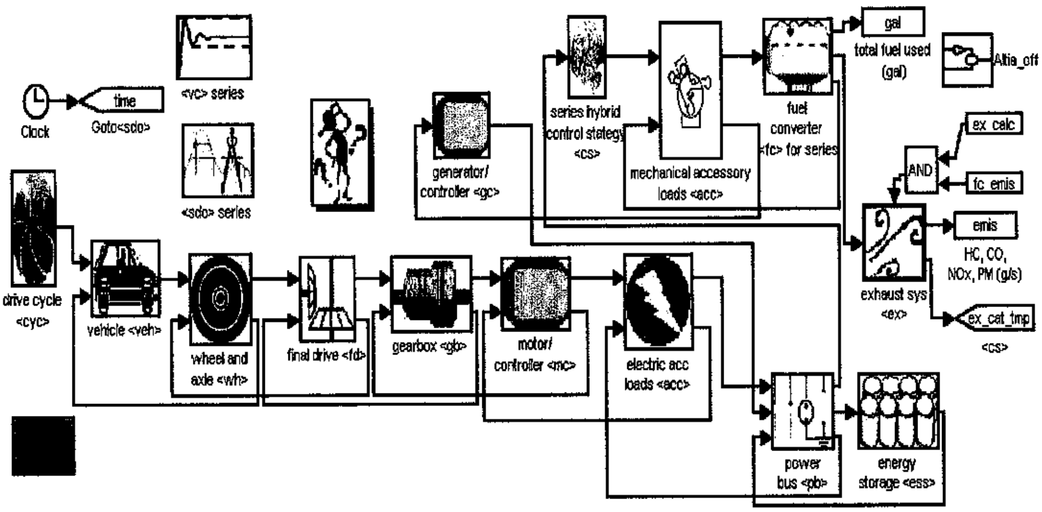



Figure 10: Backward Looking Model

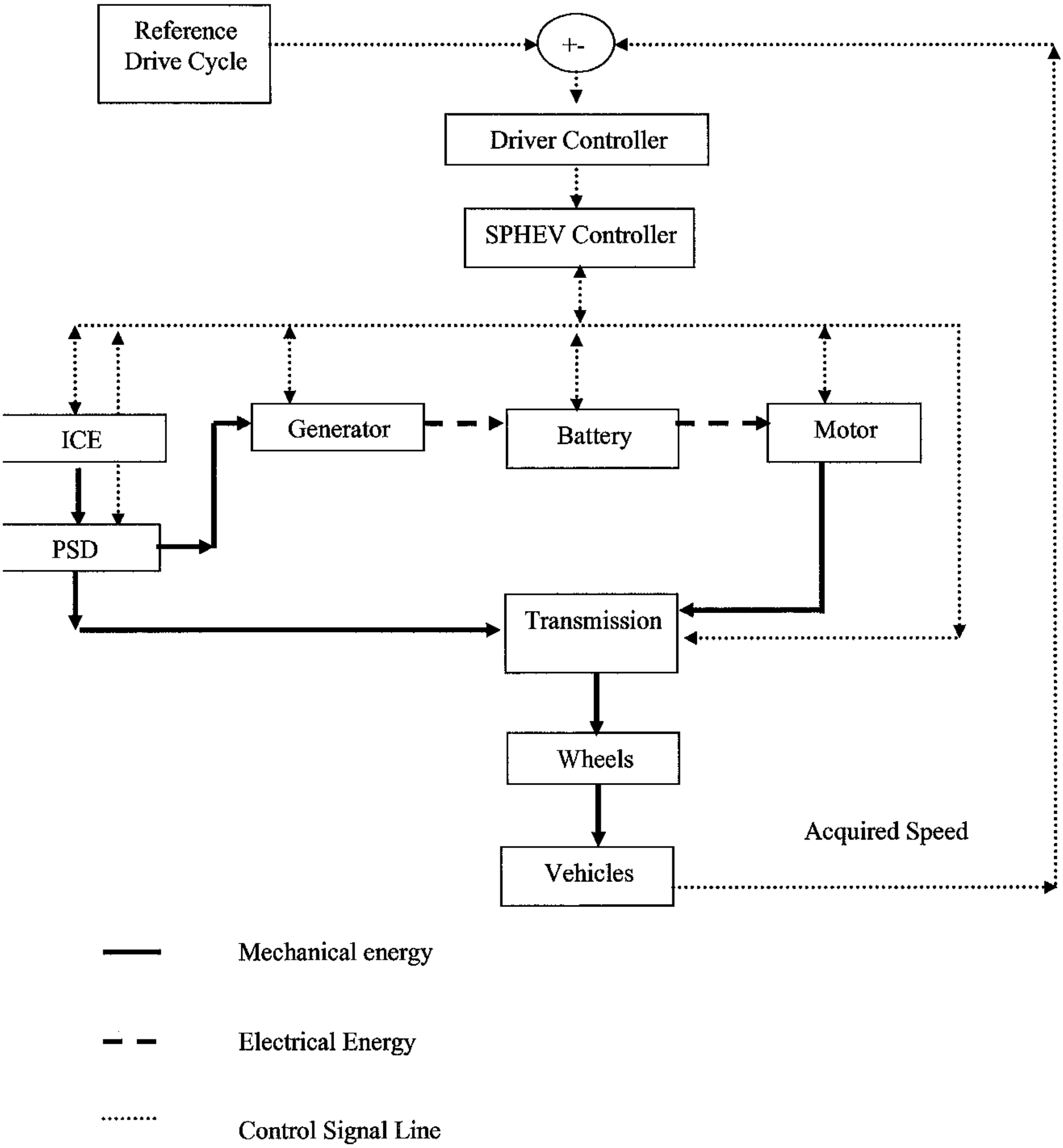



Figure 11: SPHEV Forward Looking Model

Based on the SPHEV model in Figure 11 above, reference input will be the reference drive cycle which must be followed by the driver model. Driver model will receive an acquired speed from the vehicle model. The acquired speed is compared with the required speed and the error between both of them will be used by driver model to give further command to the SPHEV controller. The controller will determine which mode to be used based on the power demand receive.

### 3.4 Simulation of Simulink SPHEV model

The simulation of SPHEV model will be in non-real time environment. Two reference drive cycle will be used to test the simulation of SPHEV model. One of them is drive cycle of Urban Dynamometer Driving Schedule (UDDS) which is always being used for light duty vehicle testing. The UDDS cycle is shown in Figure 12 below.

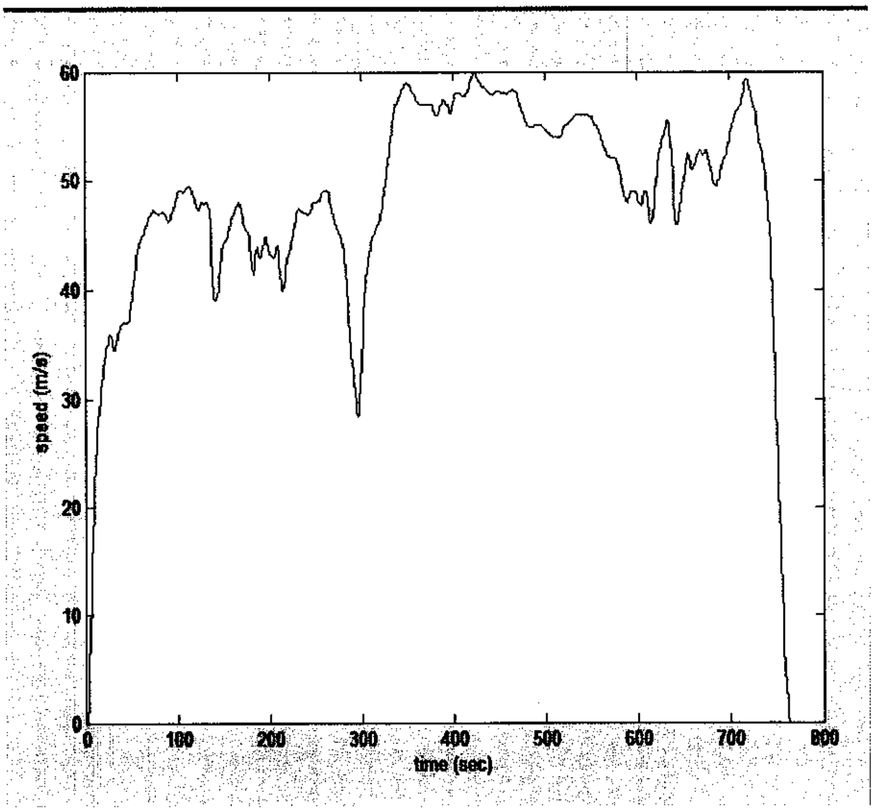



Figure 12: UDDS drive cycle

The other reference drive cycle is as shown in Figure 13 below. It is the drive cycle of acceleration from 0 to 100km/s in 15 seconds.

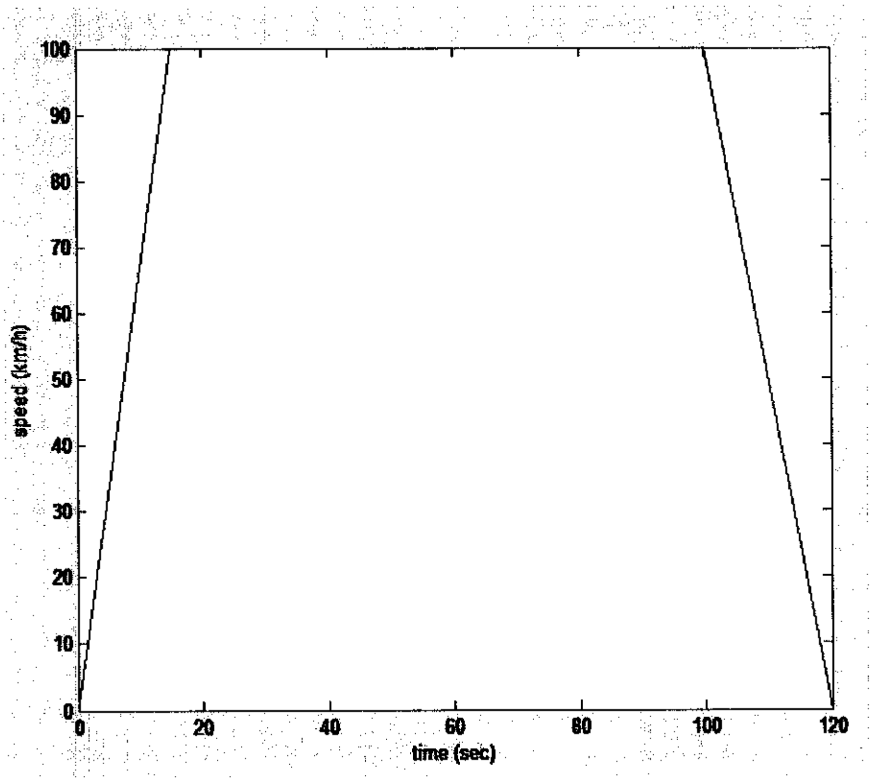



Figure 13: Second drive cycle

### 3.5 Controller Design

There will be two controller used in the SPHEV model which is shown in Figure 11. First controller is the driver model controller which consist only the PID controller. The driver model will respond to any change happen and will compensate the error by applying the right PID parameters. The other controller will be the SPHEV controller which controls the energy management system for the SPHEV drive train.

### 3.6 xPC Target and xPC Target Box

xPC Target is a powerful and user friendly tool for rapidly implementing real-time control systems on a digital computer. The software works through MATLAB Simulink, allowing control system to be designed in block diagram form in Simulink and then realized in the physical system. The block diagram is simply compiled to an executable and then loaded onto dedicated computer (in this case, the xPC TargetBox) for real-time execution.

There are three major components to xPC Target control system: The host PC, the xPC Target Box, and the plant. The host PC can be any PC running MATLAB, Simulink, and the Real-Time Workshop and xPC Target toolboxes. The Target Box is an industrial small PC-compatible computer that is designed specifically to run the xPC Target real-time operating system. It contains the computer hardware that executes the controller. This means that the entire machine is dedicated to run the controller program;

Through its I/O hardware, the Target Box will then interact with the plant. It will read sensor signals through its inputs and then output control signals based on the control law designed by the user. Figure 14 shows the typical configuration between the hardware and the xPC Target. The plant is connected through its internal DAC hardware.

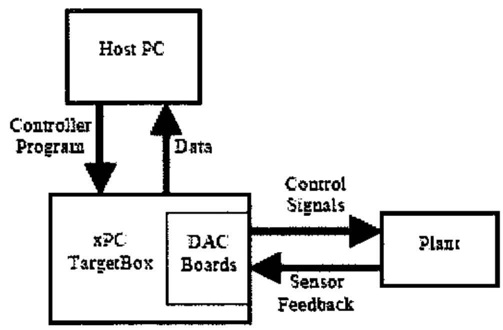



Figure 14: Typical xPC Target hardware Configuration

### 3.6.1 Host PC and Target PC Connection

There are two kind of connection between Host and Target PC which are as following:

1. Serial Connection

   - Host and target PC are connected directly with serial cable using RS-232 ports.The serial connection is shown in Figure 15.

2. Network Connection

   - The host and target are connected through a network. The network can be a LAN, Internet or direct connection using crossover Ethernet cable. The connection using the Ethernet cable has to use TCP/IP protocol for communication. The network connection is shown in Figure 16.

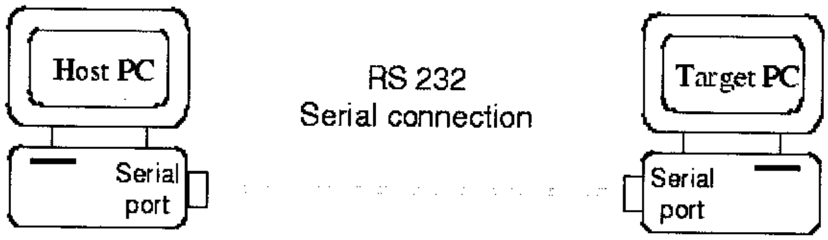



Figure 15: Serial Connection

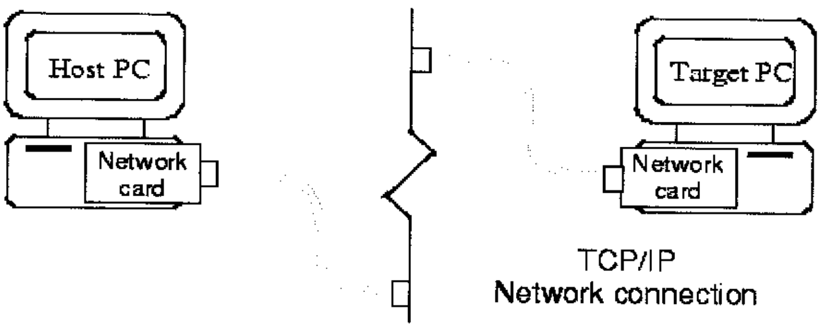



Figure 16: Network Connection

### 3.6.2 Building xPC Target

The steps taken in building the xPC Target can are as follows:

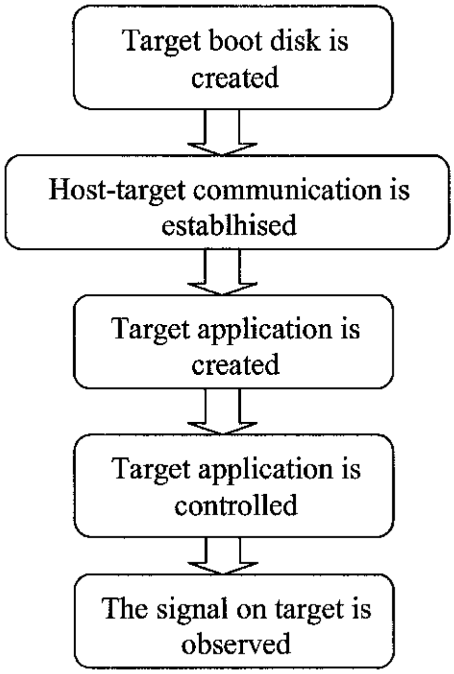



Figure 17: Building xPC Target steps.

#### 3.6.2.1 Creation of Target Boot Disk

The target boot disk is loaded and run into the xPC Target kernel. The target boot disk can be created by following the steps below:

1. A formatted disk is inserted into the Host PC disk drive.
2. In Matlab command window: 'xpcexplr' is typed

The xPC Target Explorer window will be shown as in Figure 18. From the TargetPC1 folder, "*configuration*" is chosen, then the "boot floppy' is chosen at the Target boot mode. Boot disk is created.

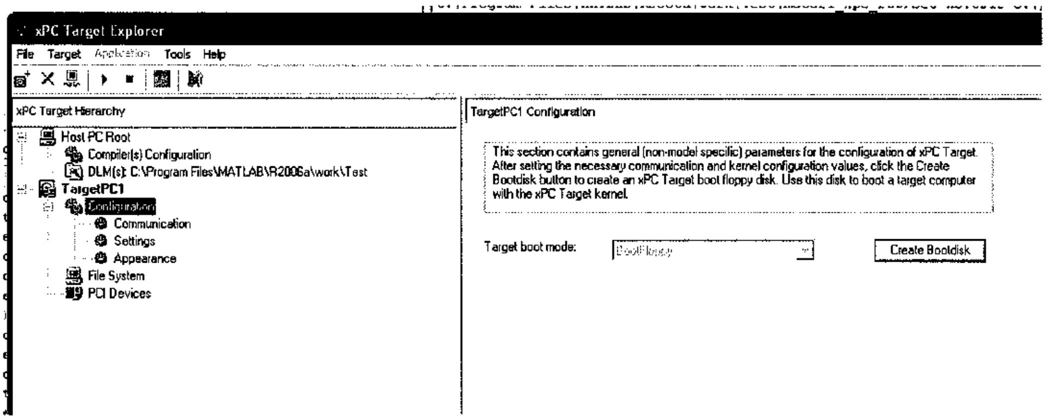



Figure 18: xPC Explorer for boot disk

### 3.6.2.2 Establishing Host-Target communication

Establishing host-target communication means develop the connection between the Host PC and the Target PC. There are two connections available which are TCP/IP and also RS232. From the xPC Target Explorer, *communication* set up is chosen and the diagrams in Figure 19 below will be displayed.

Choose communication method

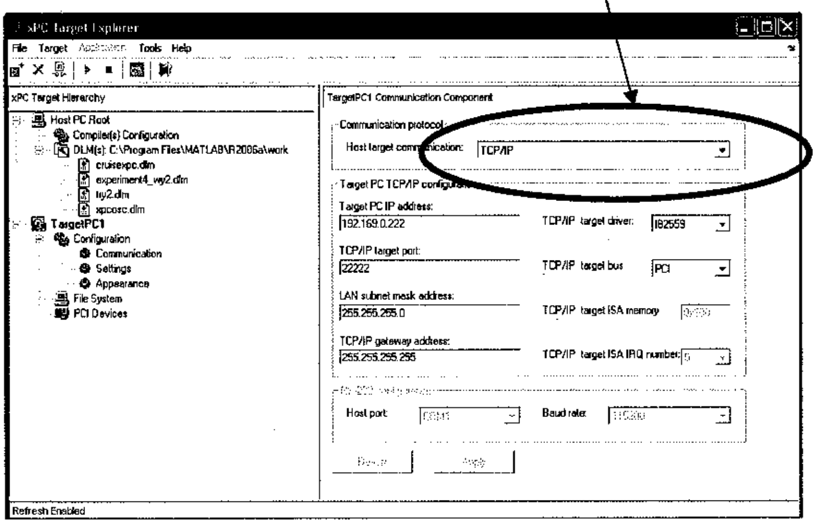



Figure 19: TCP/IP host communication

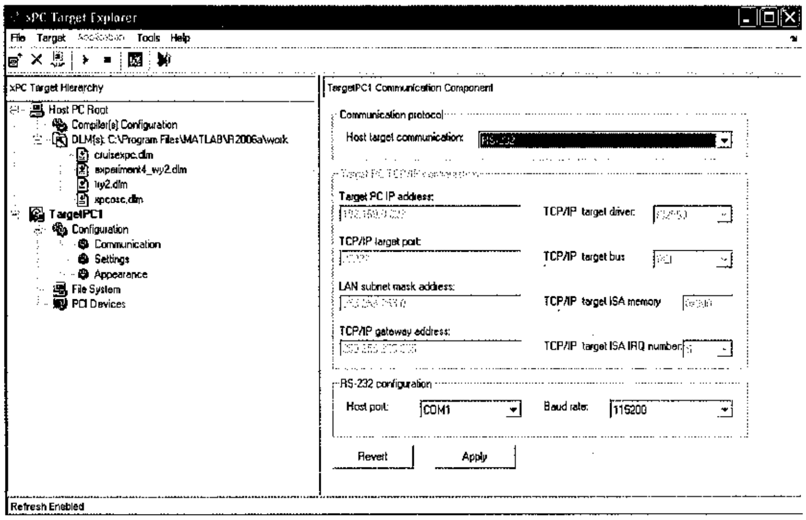



Figure 20: RS232 host communication

The host-target communication by network connection is preferred as compared to the RS232 connection because it gives faster transfer rate. Figure 19 and Figure 20 explain how the connection is built between the host-target PC.

The Local Area Connection (LAN) Status is set up as shown in Figure 21. The IP Address for LAN must not be the same as IP Address for Target PC which is shown in Figure 22. Once the connection setting was set up, the host PC and target PC is connected and an established connection is shown in Figure 23.

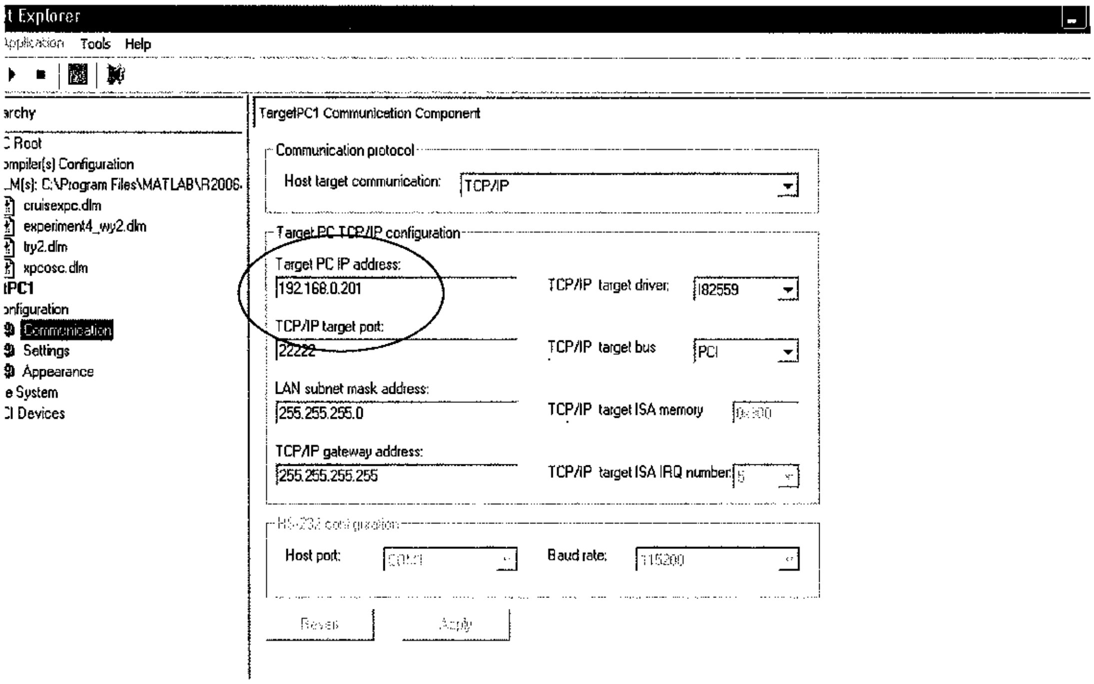



Figure 21: IP address for Target PC

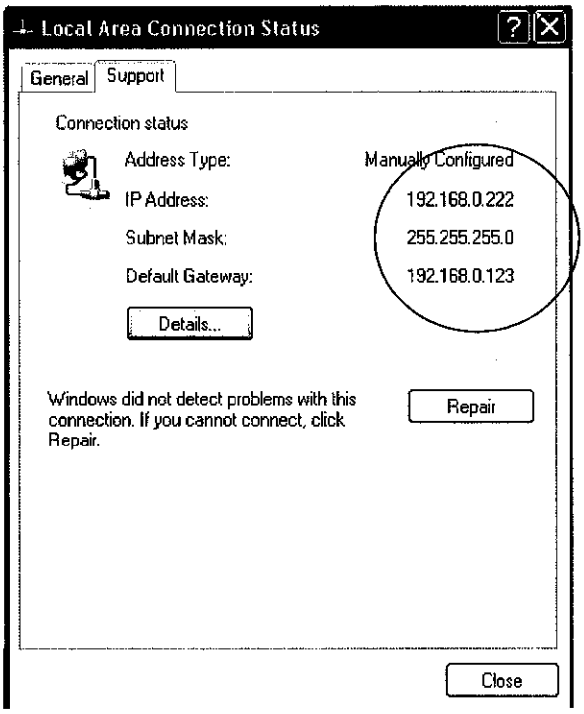



Figure 22: Local Area Connection setting

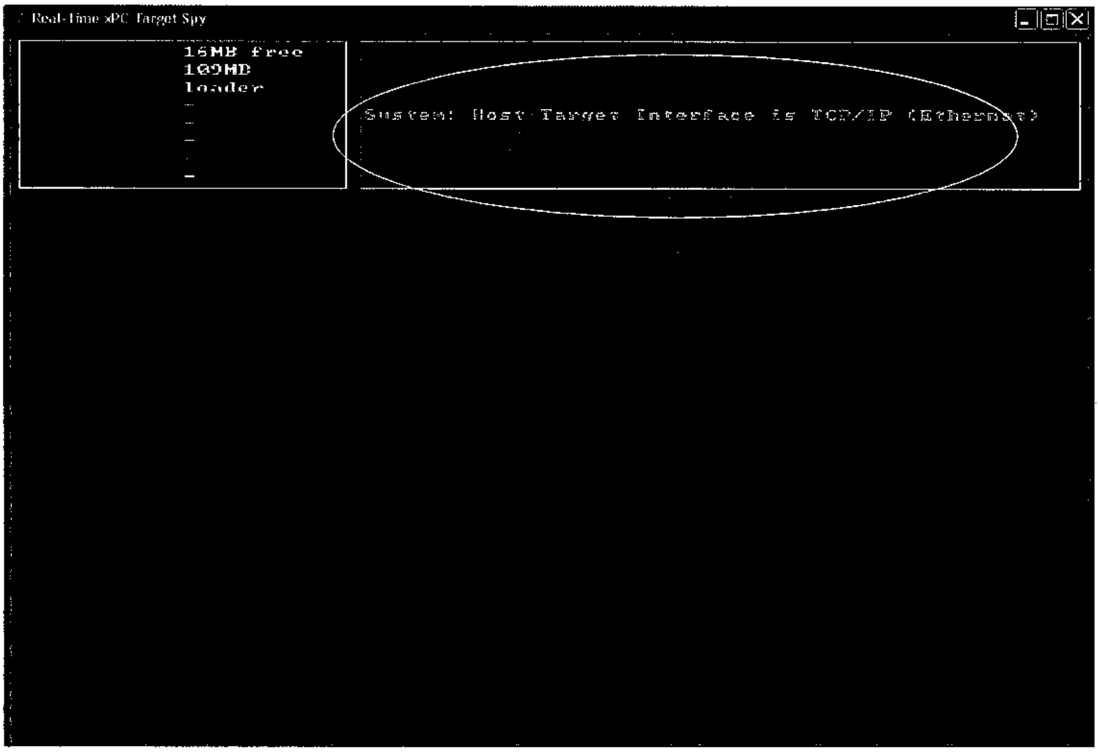



Figure 23: Prove of connection established

### 3.6.2.3 Creation of Target Application

This step is to build the real-time application and download it on the target. It will automatically download the built model into the target. The steps required are:

1. Respective Simulink model is opened.
2. From the menu bar, *tools* menu is chosen. From the *tools menu, Real Time Workshop* is selected and then *options* is preferred.
3. The information is as shown in the following Figures 24 and 25

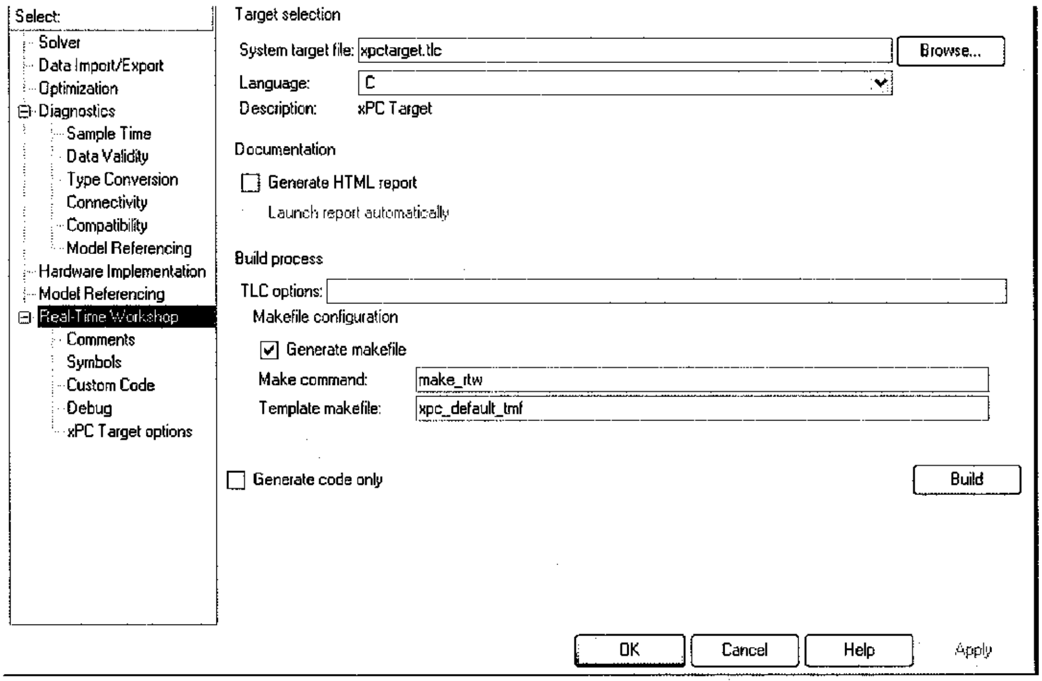



Figure 24: Real-Time Workshop Configuration setting

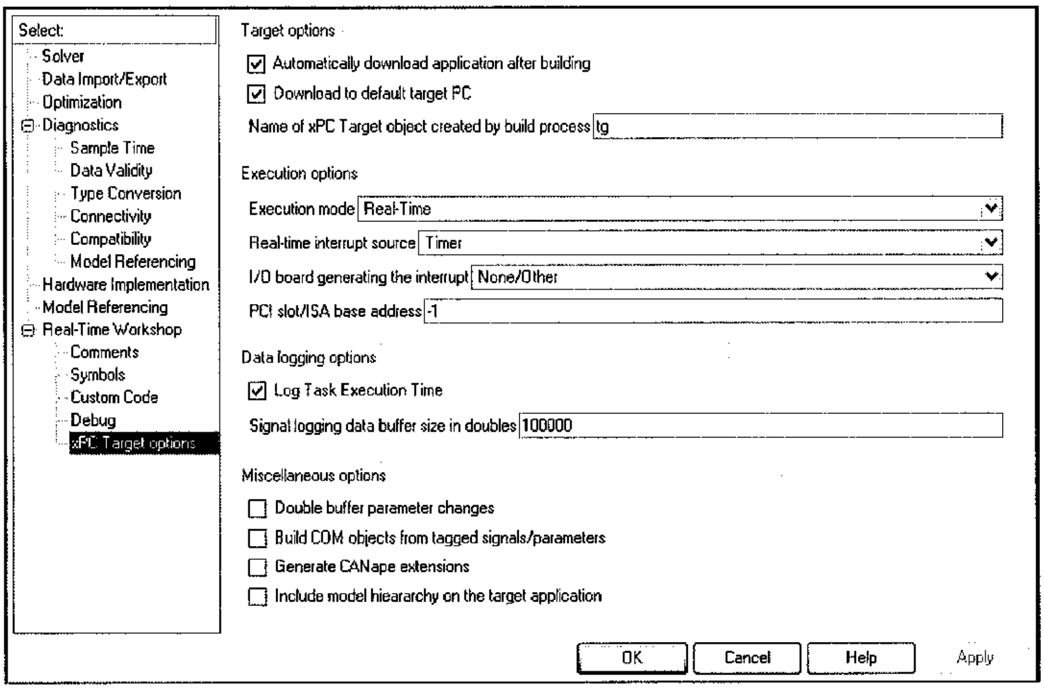



Figure 25: xPC target configuration setting

Once the set up is completed, the model is built and downloaded to the xPC Target by the following steps shown in Figure 26.

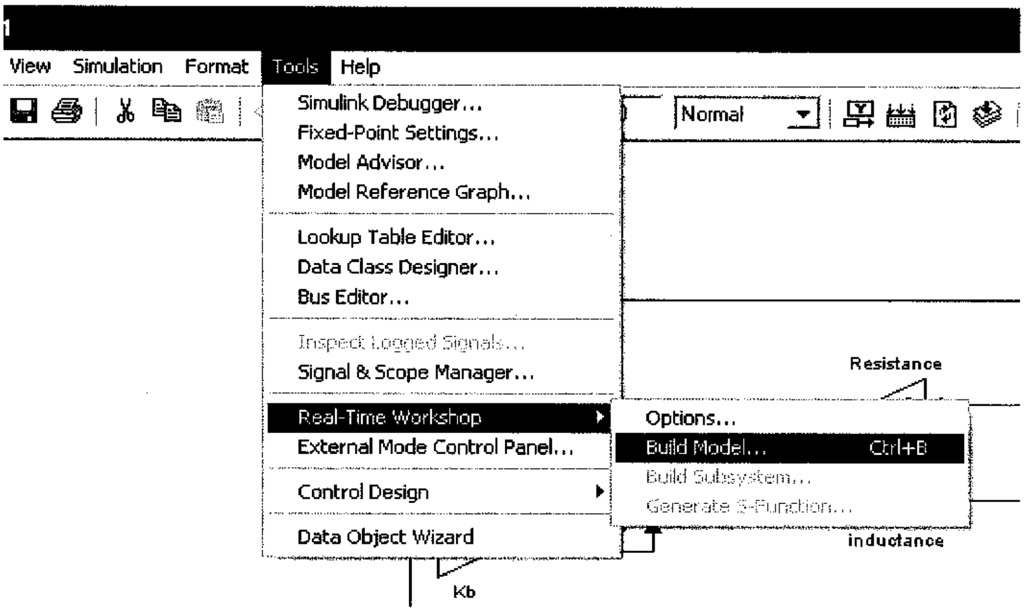



Figure 26: The model is built and downloaded into the xpC Target

Once the model is built, the diagram is set to run in "External" mode (the diagram will interface with I/O hardware) which the location of "External" mode is shown in Figure 27 below. The model can be Simulated in real-time once the "Play" button is pressed.

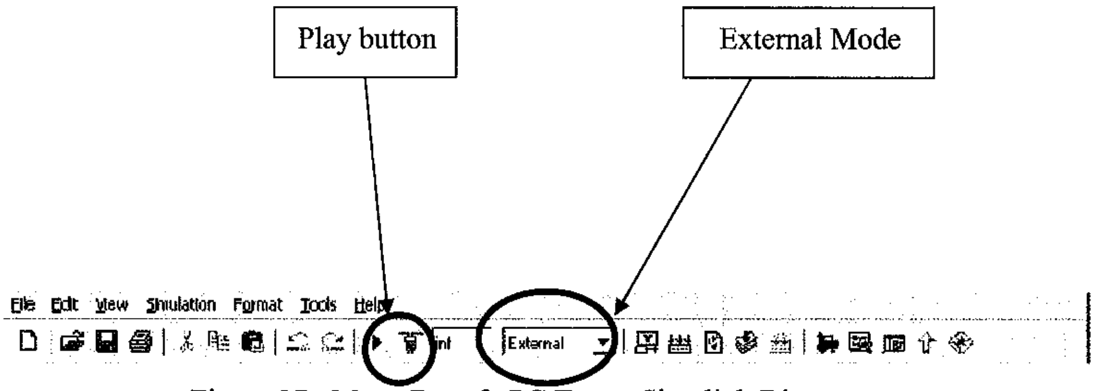



Figure 27 : Menu Bar of xPC Target Simulink Diagram

#### 3.6.2.4 Controlling the Target application

The target application can be controlled via the xPC Explorer. From the xPC explorer, the target can be connected to the Host PC, run the simulation and stop the simulation. Figure 28 below shows the clearer picture of xPC Explorer.

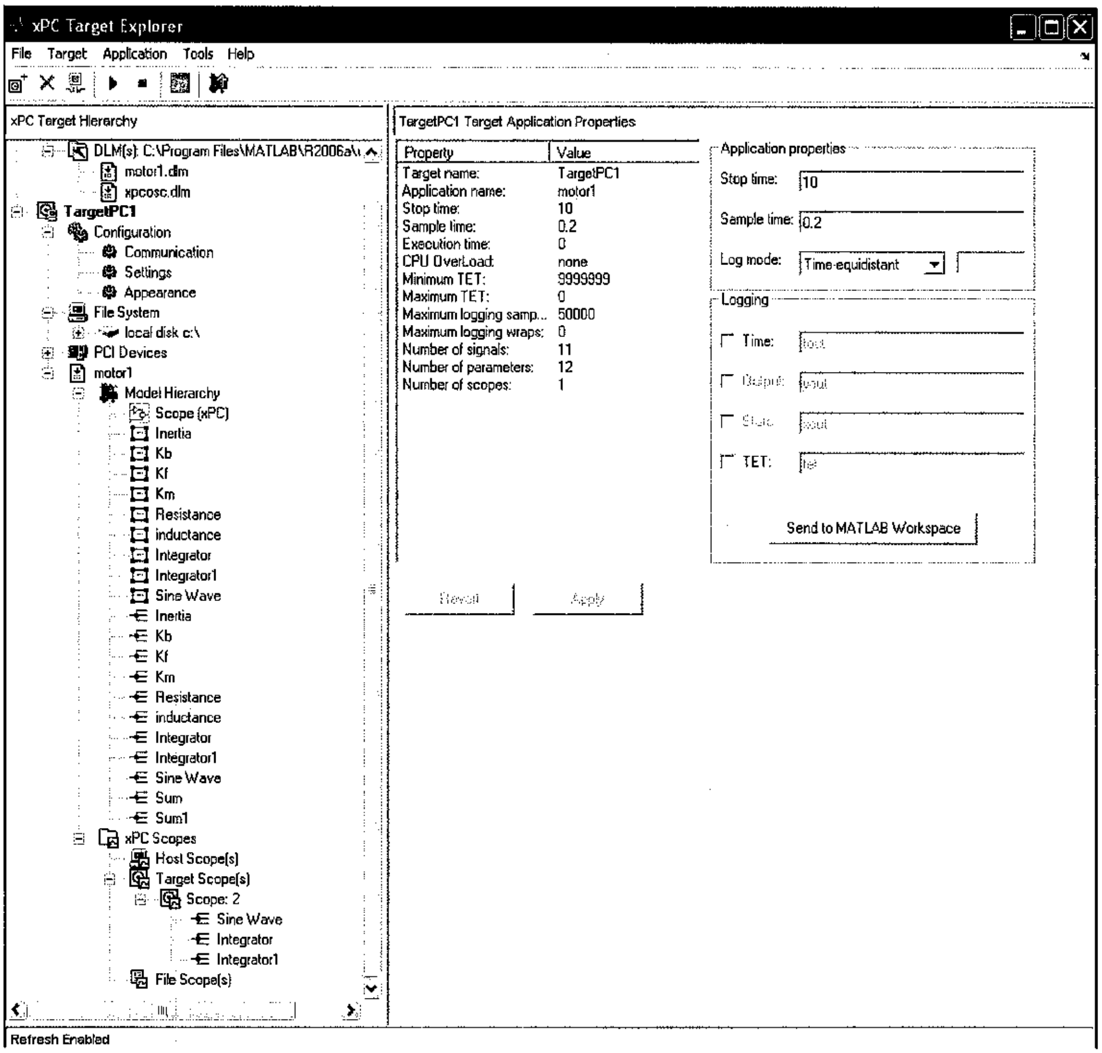



Figure 28: xPC Target Explorer windows

### 3.6.2.5 Observation of the signal on target.

Once the model is running in real-time, the simulation can be shown in the graph as shown in Figure 29.

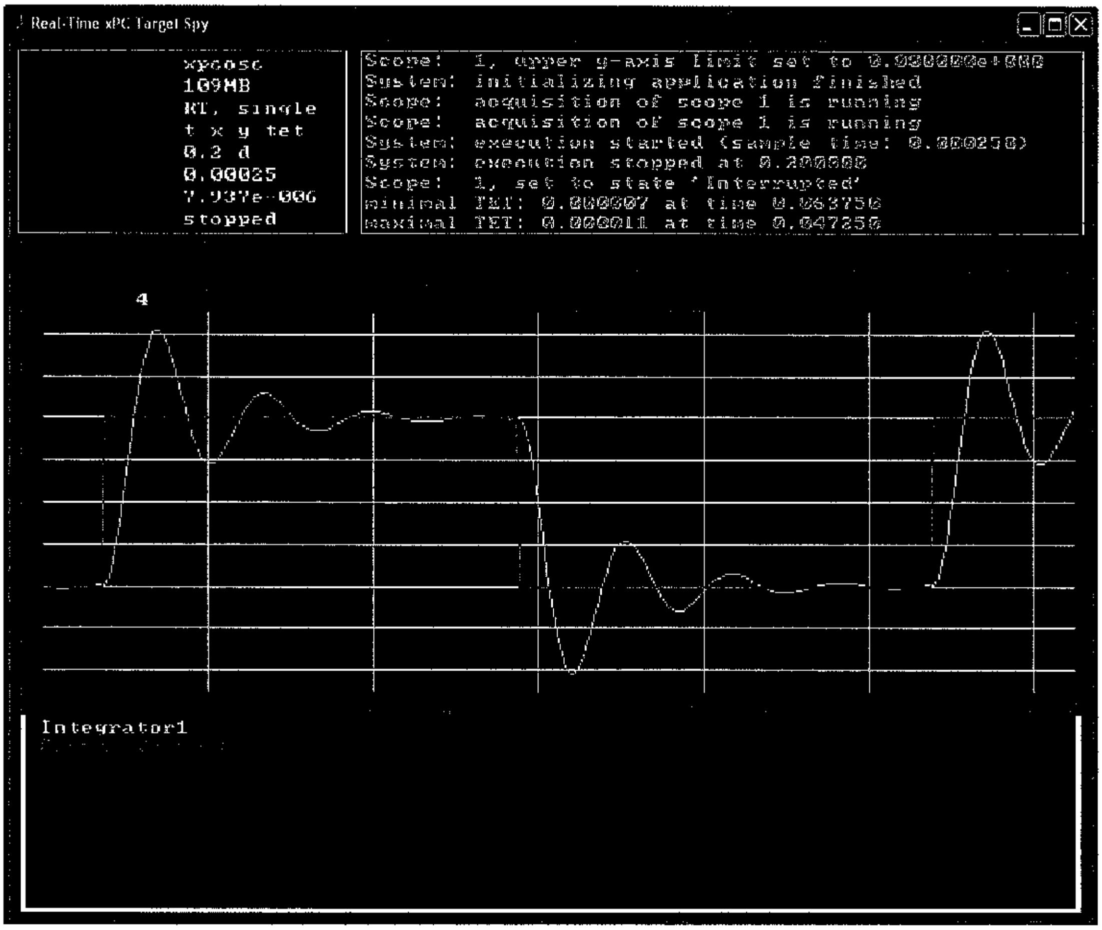



Figure 29: Real-Time xPC Target Spy

### 3.7 Target PC and Hardware Configuration

The system shown in Figure 30 consists of a motor with a tachometer being driven by a power amplifier and resisted by an unknown load. The amplifier is connected to an analog output channel in the Target Box, and the tachometer output is connected to an analog input channel. The shown system is implemented in order to run the rapid control prototyping using the xPC Target. [9]

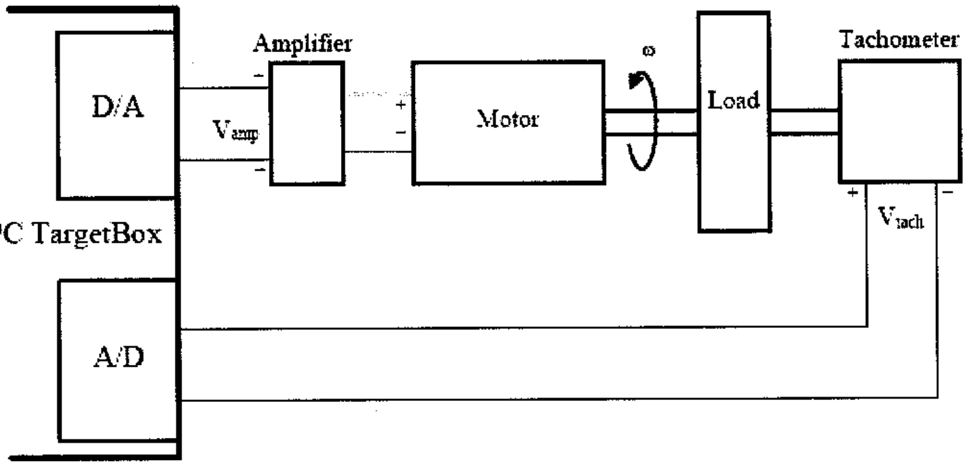



Figure 30: Connection between hardware and xPC Target Box

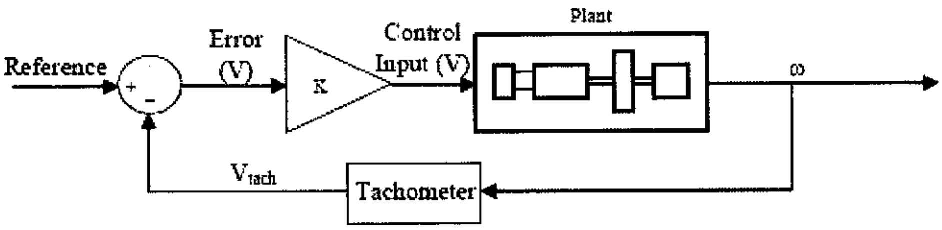



Figure 31: Controller Block diagram for motor tachometer system

In order to implement the real time rapid prototyping using the motor as the hardware, several special driver blocks are used which correspond to specific I/O hardware in the xPC Target Box. A summary of the hardware contained in the xPC Target Box that is used is shown in Table 3, along with the locations of the driver blocks

| Board | Base Address | Function | Driver Block Location |
|---|---|---|---|
| Diamond MM-32-AT Multifunction Board | 0x300 | Analog Voltage Output | xPC Target => D/A => Diamond => MM-32 |
| | | Analog Voltage Input | xPC Target => A/D => Diamond => MM-32 |
| | | Digital Input | xPC Target => Digital Input => Diamond => MM-32 |
| | | Digital Output | xPC Target => Digital Output => Diamond => MM-32 |

Table 3: Summary of I/O hardware in xPC Target Box

The analog voltage input and output functions on the Diamond MM-32 board are used to control the motor and read the tachometer voltage, respectively. The driver blocks for the analog input and analog output is added to the Simulink block diagram by simply locating them in the library browser window and dragging them into the blank diagram. The diagram should now look like Figure 32.

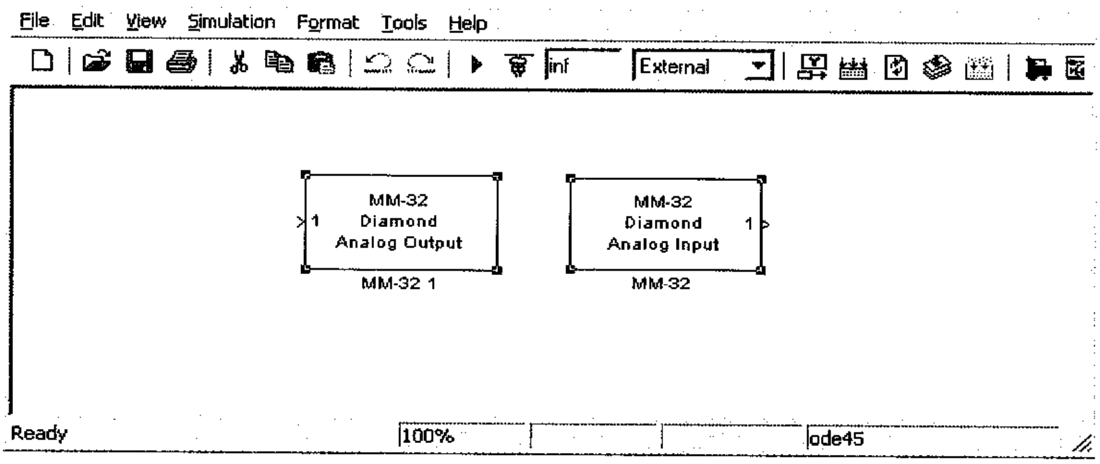



Figure 32 : Block Diagram Containing an Analog Input and Output Driver Block

The parameters of Analog Output and Analog Input are adjusted by double clicking on the block and both of Figure 33 and Figure 34. In Figure 33, the "Sample Time" parameter is set to match the fundamental step size parameter. The Base address for each board in the xPC Target Box is set in hardware, so it cannot be changed by the user. In Figure 34, the analog I/O board in the xPC Target Box has 32 individual analog input lines. Selecting "1-32SI" will make allow all 32 lines to be used, but they must all share a common ground. Selecting "1-16DI" will instruct the board to measure the voltage between two individual lines. For this case, the "1-16DI" option is used.

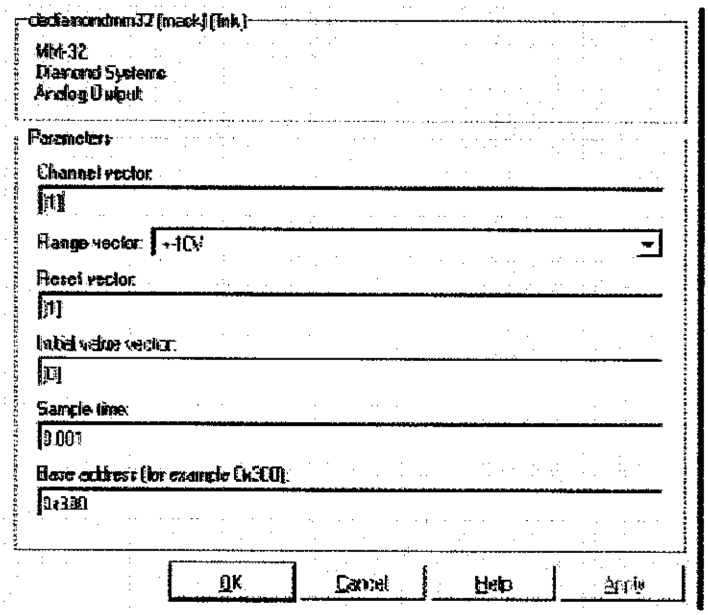



Figure 33 : Block Parameter for Analog Output

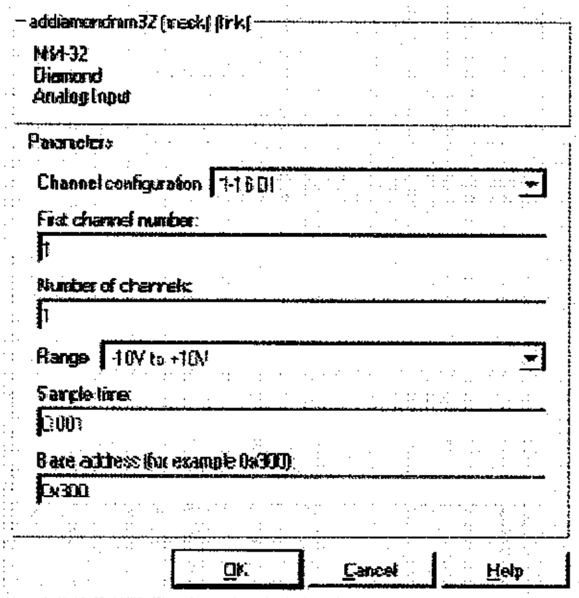



Figure 34 : Block Parameter for Analog Input

Once the set up of both analog output and analog input has completed, the reminder of the motor tachometer system can be filled as shown in Figure 35.

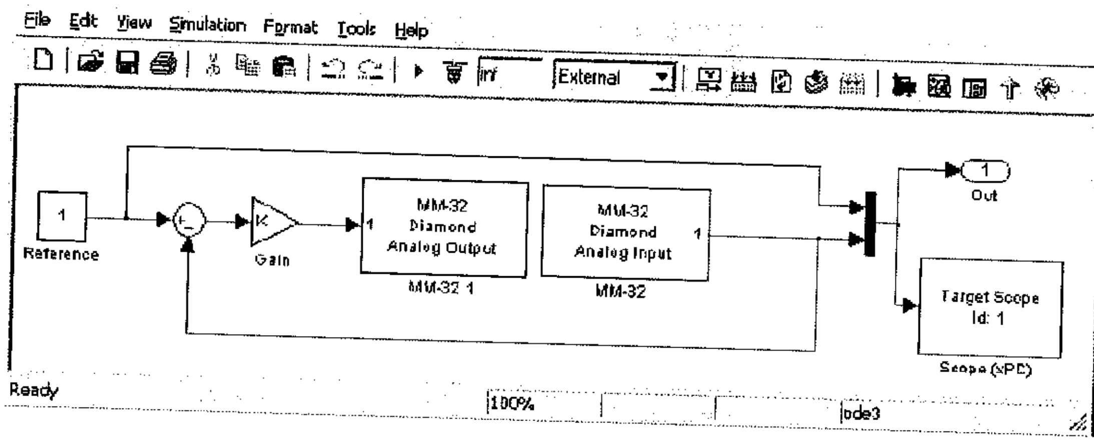



Figure 35: Proportional Motor tachometer Control system using xPC Target

The block diagram is now finished, and is ready to control the system and collect data. To run the diagram, it must first be compiled and uploaded to the xPC TargetBox. When the compilation process has completed, the message "Successful completion of xPC Target build procedure for model" will be displayed. If any errors occur, an error message dialog will appear that will provide information about the errors that have occurred.

# CHAPTER 4

# RESULTS AND DISCUSSION

The simulation of SPHEV will concentrate only finding the right control strategy to apply in the SPHEV system. Since the SPHEV is a very complicated system, therefore a good control strategy is very important in order to manage the power train system. The strategy is about choosing the right mode between either electrical, series, parallel or both series and parallel during certain driving conditions.

A controller for driver model will also be concentrated as it determine how much response it needs in order to compensate for the error occur between the acquired and required speed.

The test will use a reference drive cycle which is Urban Dynamometer Drive Scheduling (UDDS) for the purpose of light testing method (Figure 12). This reference drive cycle is a standardized cycle which has been used in testing the drive train of any light vehicles. The second reference drive cycle is a drive cycle of acceleration from 0 to 100km/h in 15 seconds (Figure 13). The main purpose for this drive cycle is to observe the respond given by the controller driver model.

## 4.1 SPHEV Simulink Model

Figure 36 shows the Simulink model for SPHEV. It consists of drive train components which are engine, motor, generator and battery.

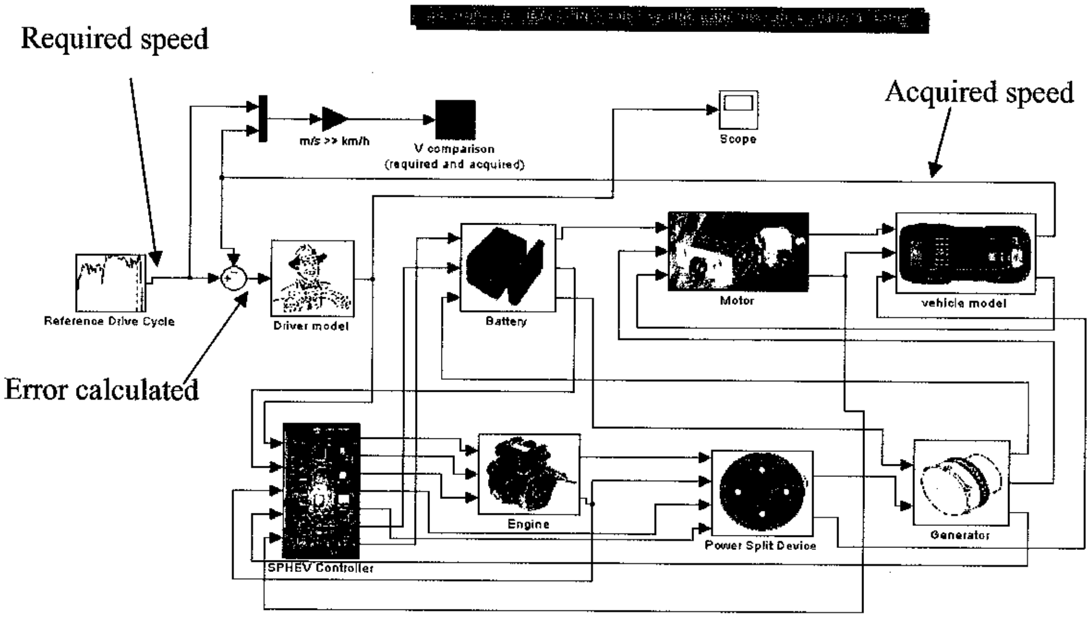



Figure 36: SPHEV Simulink model

From the model shown above, the reference drive cycle is the model input while the acquired speed as the model output. The system is a closed loop system where the output is feedback and being compared with the desired input (required speed). The error calculated will be compensated by corresponding with the right power demand actions taken by the driver. This power demand will be executed by the SPHEV controller and it will determine which mode will be used depending on the driving conditions.

### 4.2 Controllers

Two controllers for the system are driver model and the SPHEV controller.

#### 4.2.1 Driver model controller

The drive model controller is a PID controller. The controller will compare the acquired speed value with the required speed value. The difference (error) will be calculated and will respond to the power demand to compensate the errors occurred. Figure 37 below shows the PID controller used in this model.

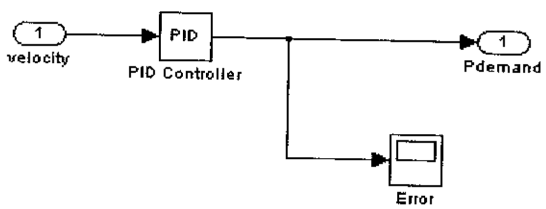



Figure 37: PID controller for driver model

#### 4.2.2 SPHEV Controller

Attached in Appendix VI is the coding for SPHEV controller. The controller receives signals from the battery, engine, generator and motor which state the conditions of its components at that particular time. All the signals will be executed so that the right command can be applied to the SPHEV system. Figure 38 shows the model for the SPHEV controller.

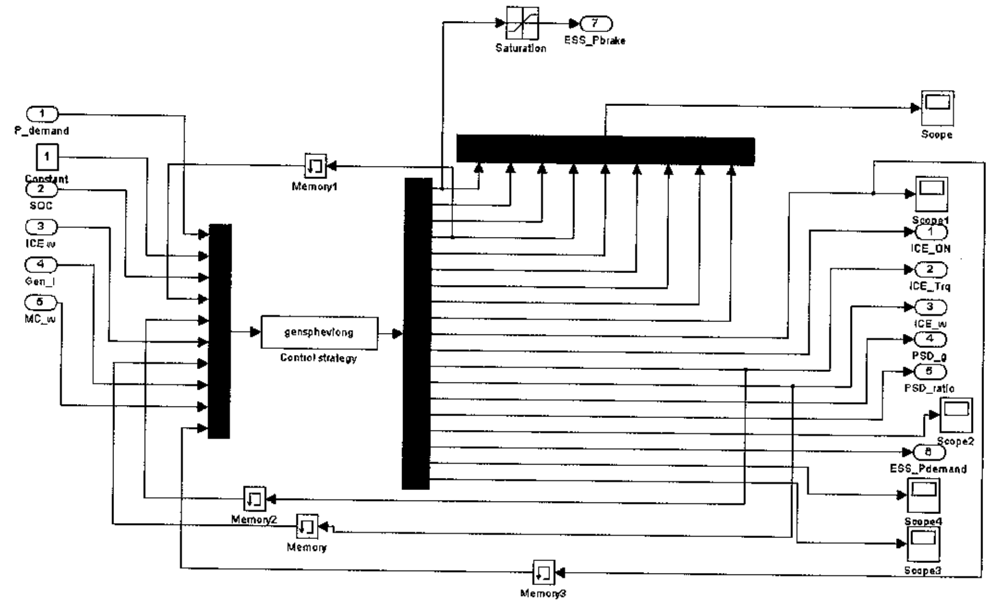



Figure 38: SPHEV controller

### 4.3 Simulation Results

Shown in Figure 39 is the general simulation result with the use of UDDS as the reference drive cycle. The y-axis represents the speed in m/s while the x-axis represents the time taken to finish the drive cycle. The result shows the comparison of required speed and acquired speed. It can be seen that the acquired speed follows the required speed starting from the 0 second until 750 seconds. From the simulation graph, an observation of increasing and decreasing of power demand was done at time between 270 seconds to 300 seconds. In real world, power demand increase when driver give more acceleration to the vehicle while reduction of power demand correspond to a time when the driver brake the vehicle. From the graph, an observation been made to the increasing and decreasing of the power demand request from the driver controller.

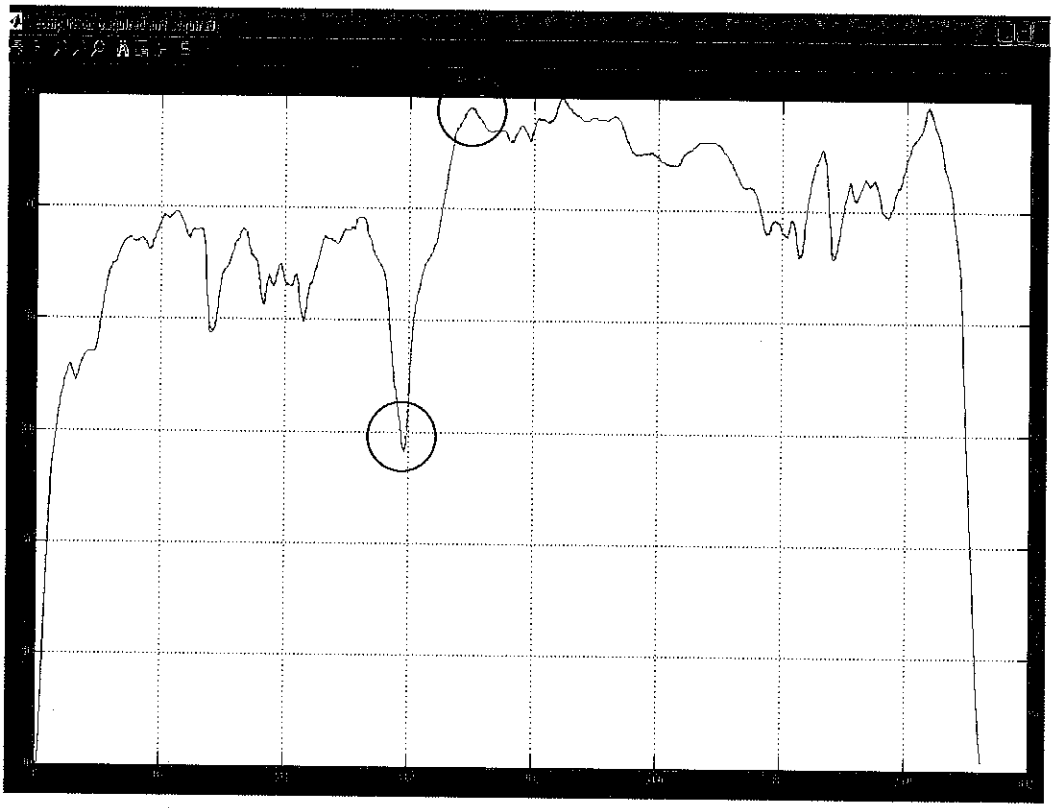


Figure 39: SPHEV simulation result

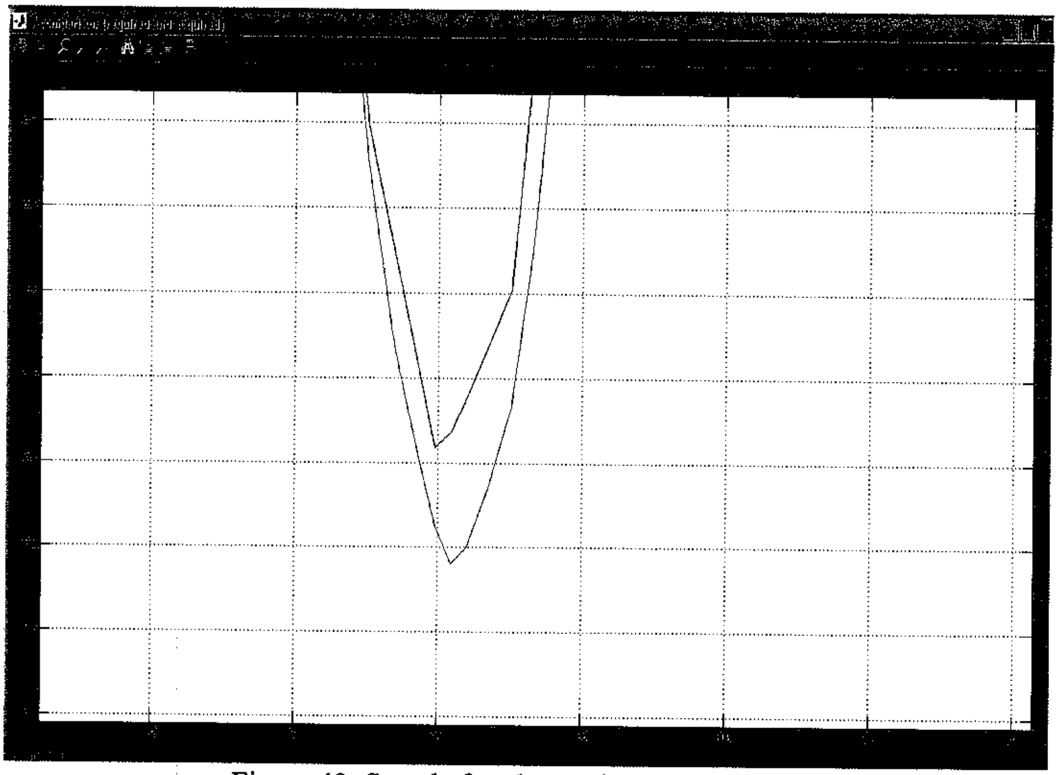


Figure 40: Speed after decreasing power demand

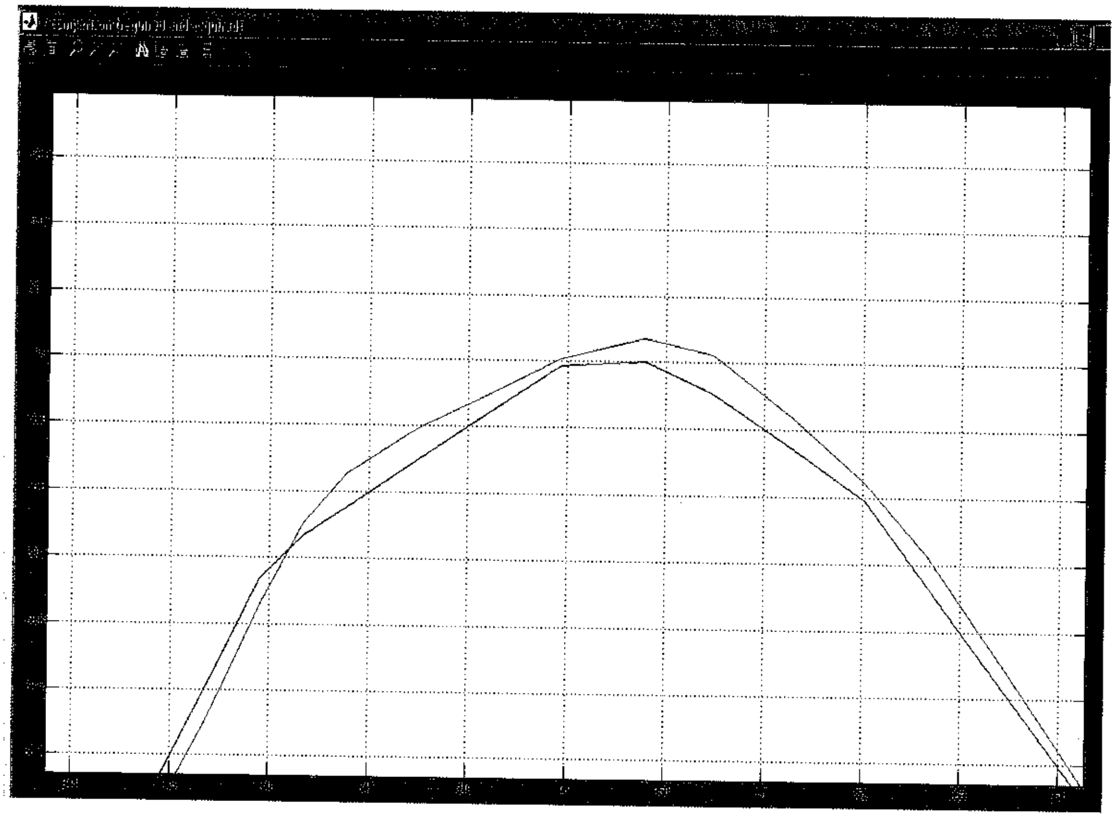


Figure 41: Speed after increasing power demand

In Figure 39, there are two circles point which is being highlight in Figure 40 and Figure 41. The green line is the reference speed while the blue line is the acquired speed. The difference of both speeds when calculated is around 0.2% which shown that the simulation is simulated successfully. The top speed at the circle points for each graph is 28km/h at 396 seconds and 60km/h at 349seconds.

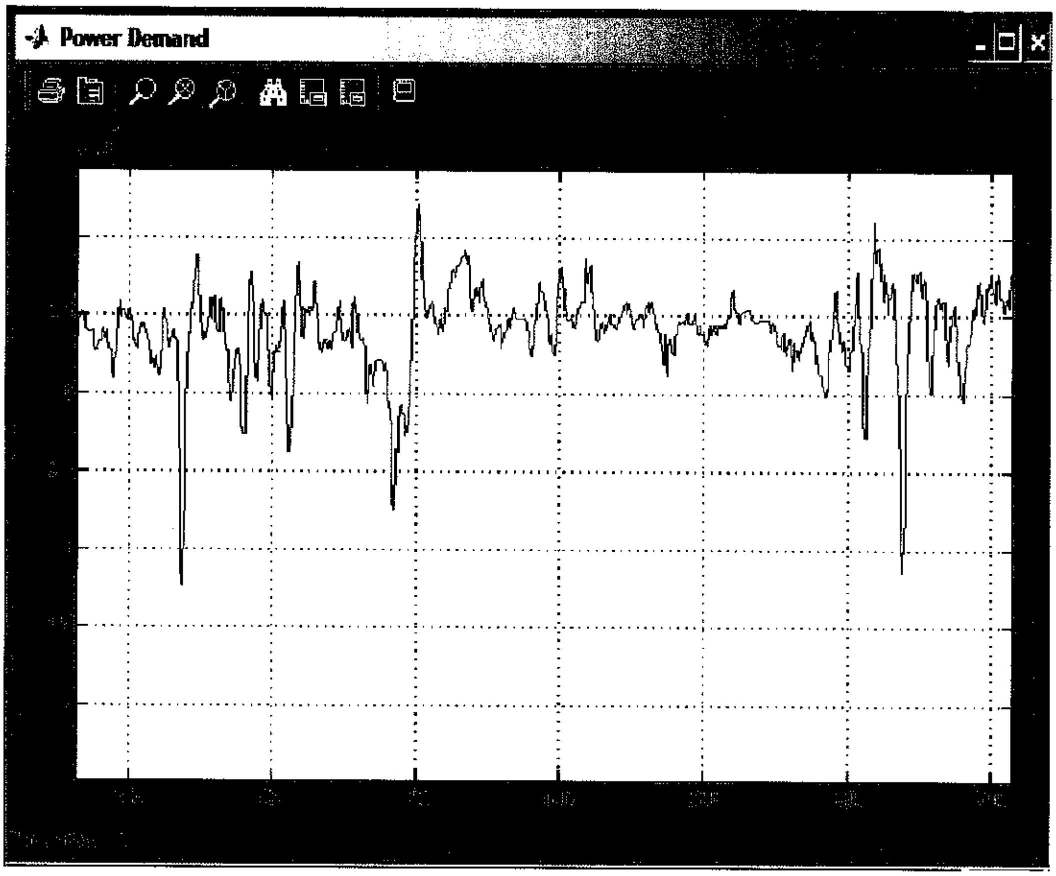



Figure 42: Power demand from the driver

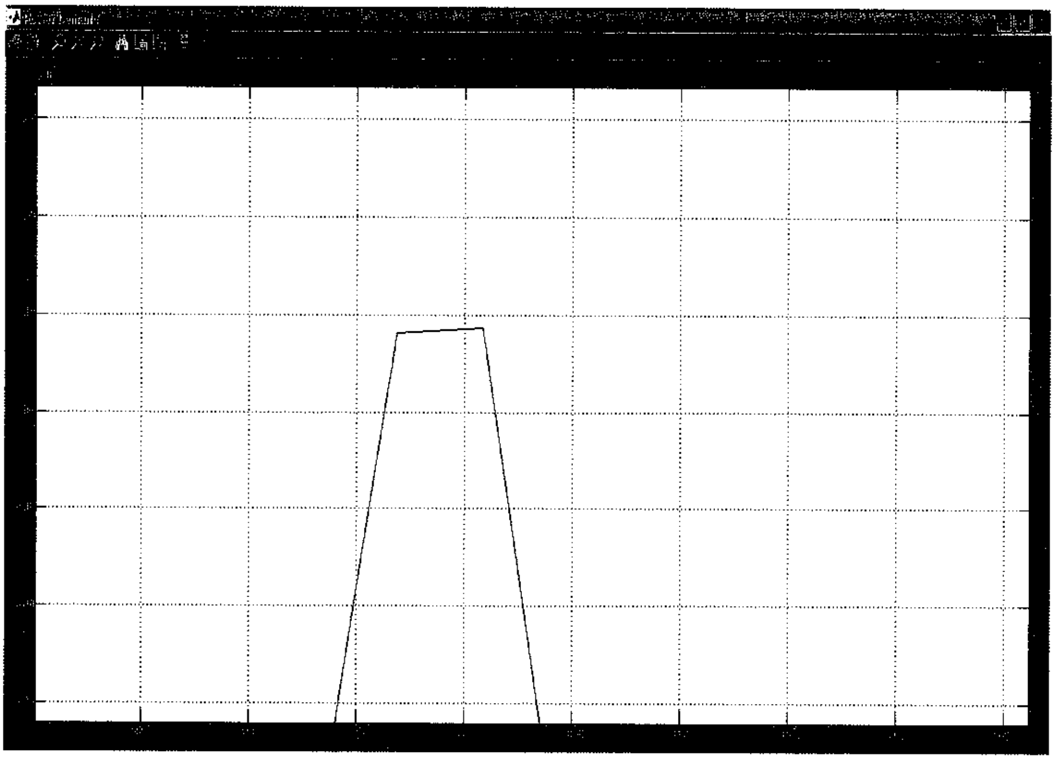


Figure 43 : Power demand at 300 sec

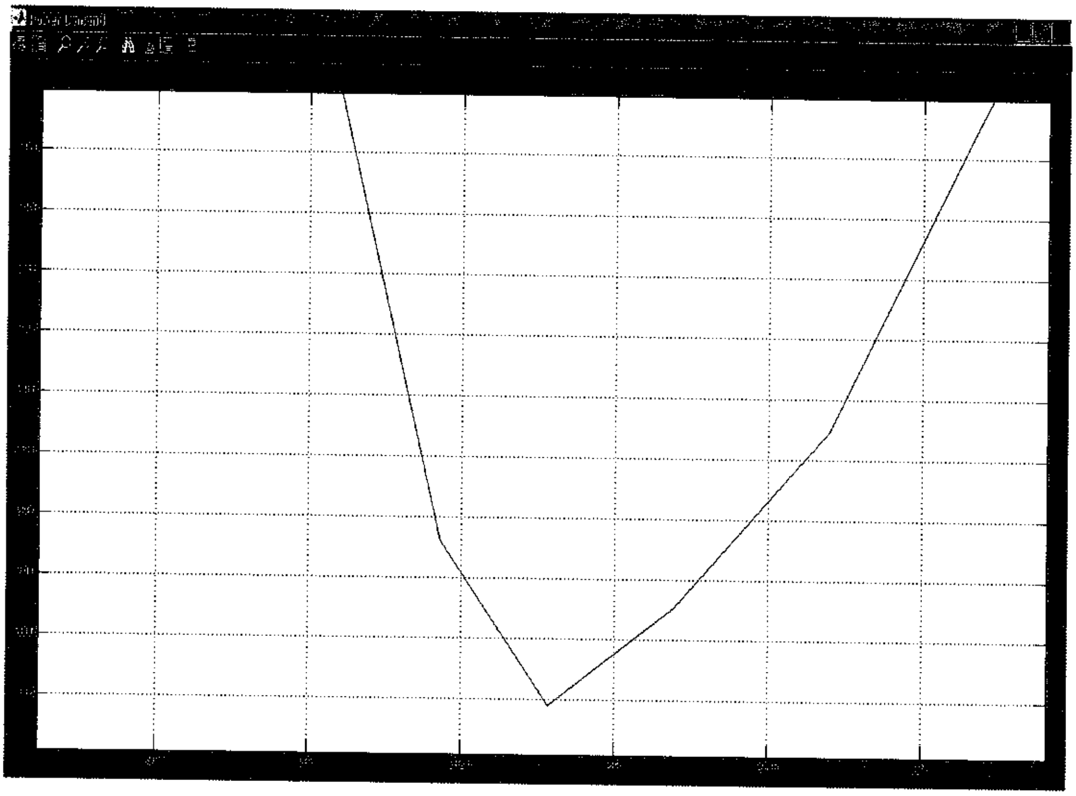


Figure 44 : Power demand after braking

Figure 42 shows the power demand requested by the driver model. Its fluctuating since the driver has to compensate for the error occurred. At 60km/h speed, the power demand required is 11.8kW which is shown in Figure 43. During braking mode which is shown in Figure 44, the required power demand decreased to less than zero since there is no additional power required to drive the propulsion system. Regenerative braking is conducted where the energy lost is transferred back to charge the battery.

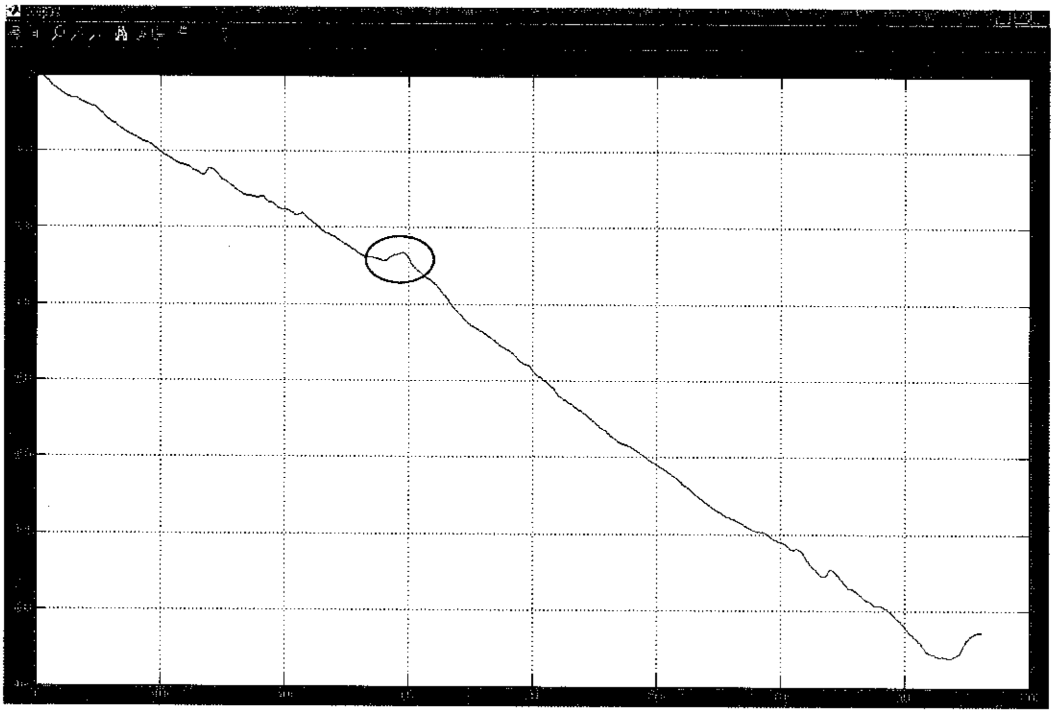


Figure 45: State of Charge (SOC) of battery

State of Charge (SOC) of battery shown in Figure 45 is decreasing since it is continuously used. The SOC is decreasing from 0.9 to 0.8 before it start to increase back. At 300 seconds (in circled), there is a little bit of spike of the SOC. It's due to the charging process of the battery since regenerative braking take place on that time.

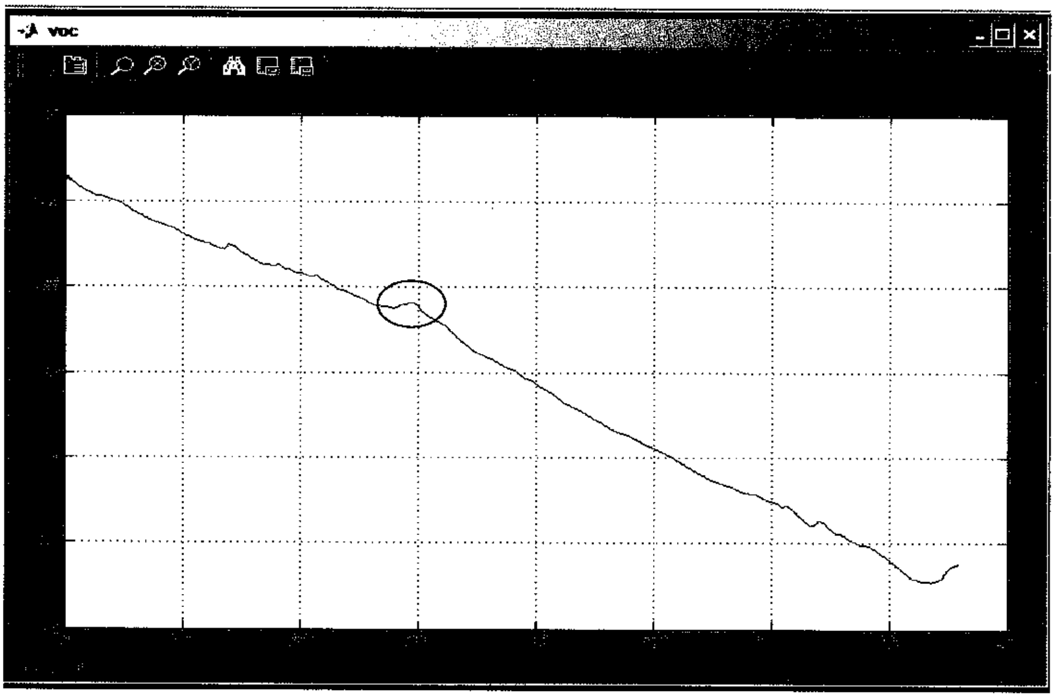


Figure 46: Voltage draw from the battery

The voltage of the battery is reducing from 430V to 405V. The voltage is also decreasing due to continuously used to run the propulsion system. During braking mode which is at 300 seconds time, the voltage surge a bit. It can be concluded that regenerative braking where the battery is charging occurred at the particular time.

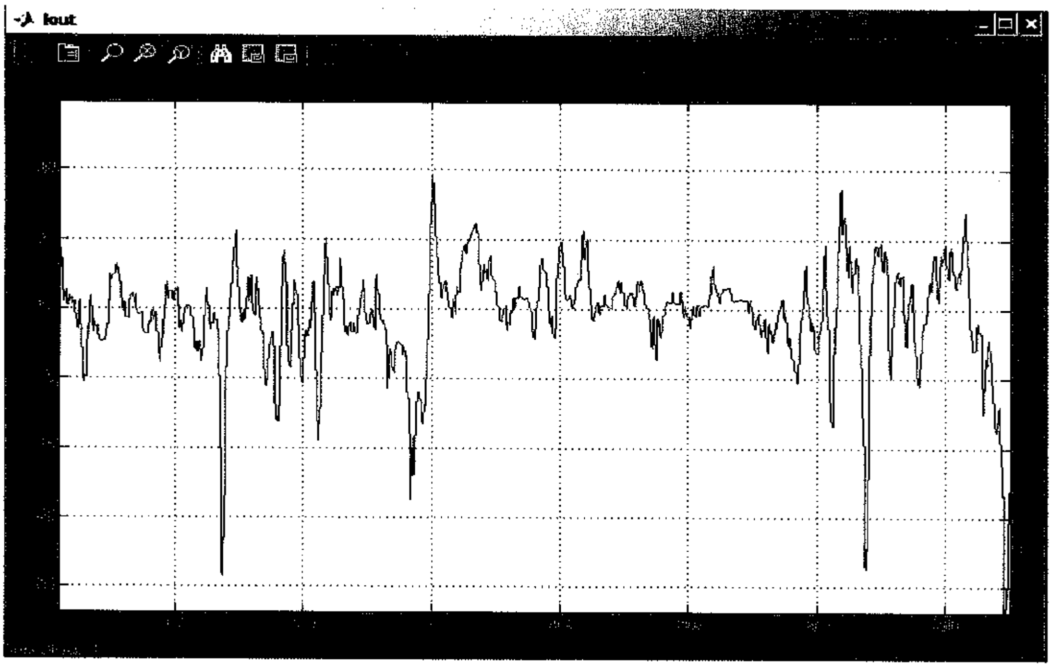



Figure 47: Current draw from the battery

Figure 47 shows the output current of the battery. It is fluctuating since it has to follow the request power demand by the driver model. During accelerating, more power demand is needed thus increased the output current from the battery. But, during braking mode, current is charge into the battery. Figure 48 and 49 explains the detail of the situation.

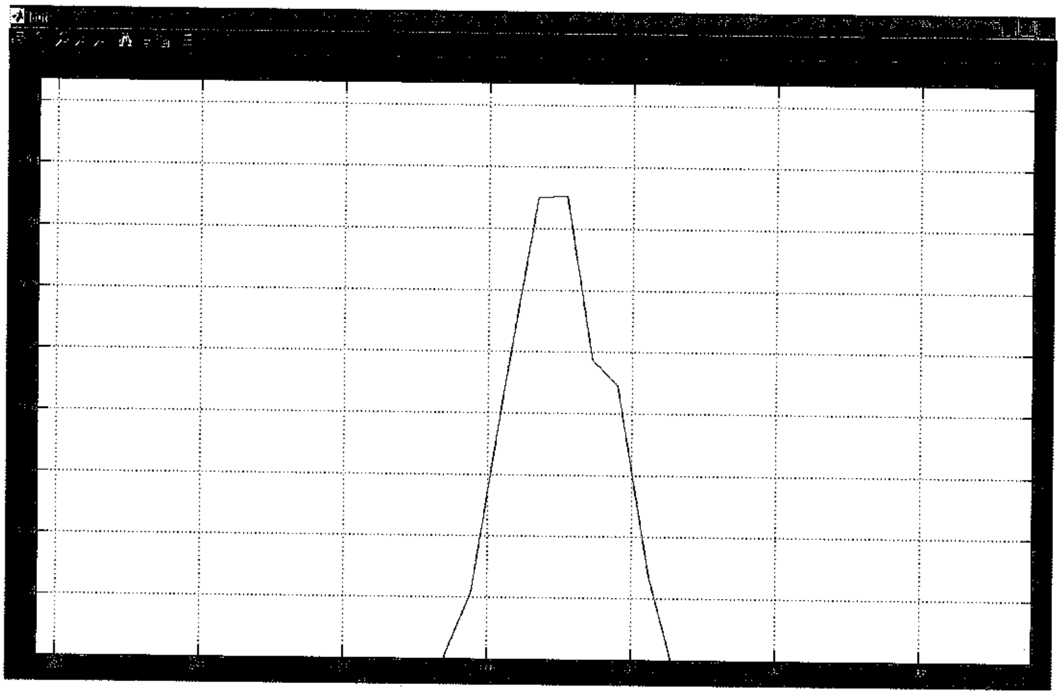


Figure 48 : Output current at high power demand

Shown in Figure 48 is the output current which is drawn from the battery. At 60km/h speed, the current drawn is about 28.5 A

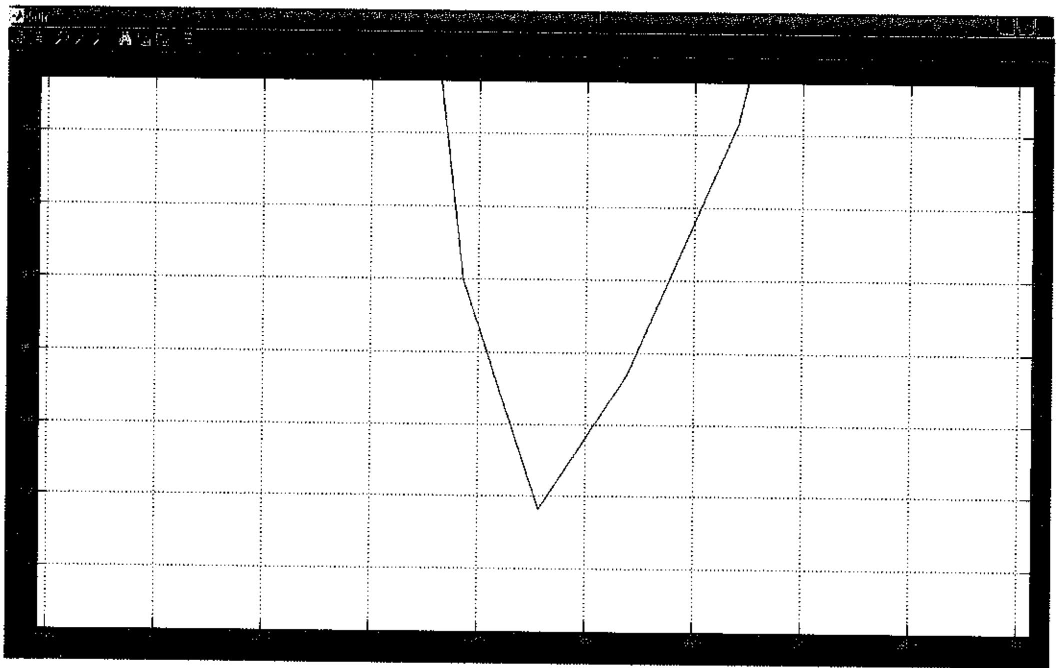


Figure 49 : Output current at lower power demand

In Figure 49, shown is the output current during braking mode. Since regenerative braking occurred, the current which is drawn from the lost energy is charge back into the battery where in the graph it is shown at -6A.

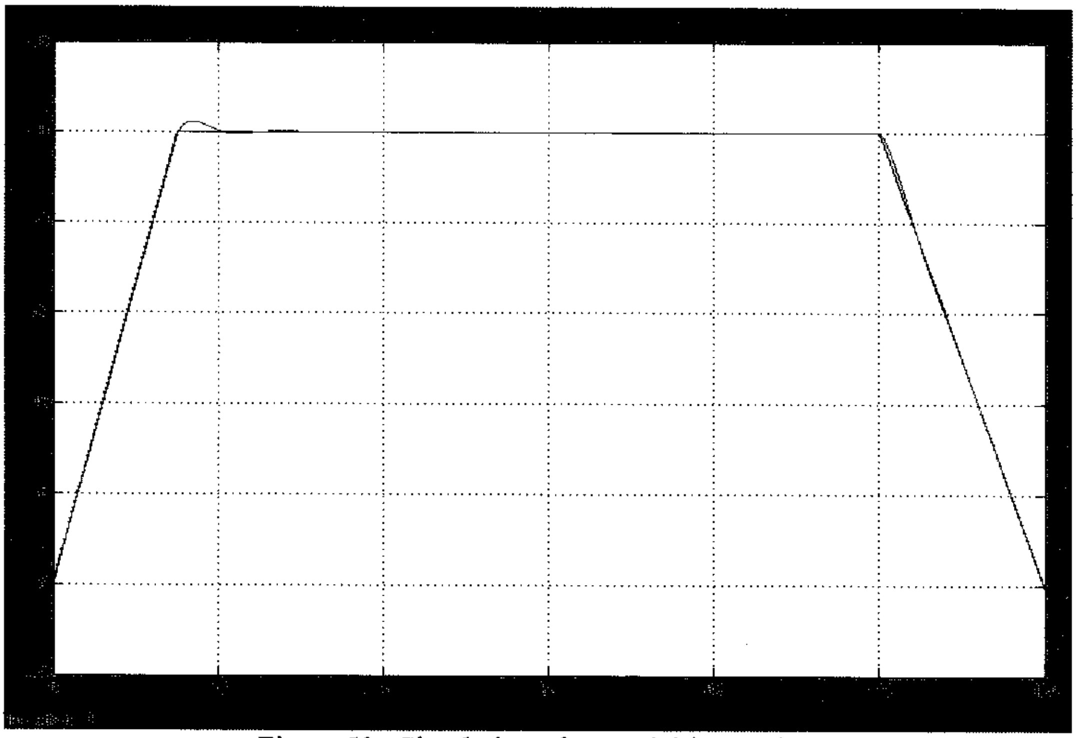


Figure 50 : Simulation of second drive cycle

The simulation result in Figure 50 shows that the driver model response to the sudden change of the vehicle acceleration. With almost zero percentage overshoot and fast settling time, the PID controller designed for the driver model is well developed.

### 4.4 Real-Time Rapid Prototyping

Ideally, the DC Motor is considered as the SPHEV for real-time rapid prototyping simulation. From Figure 30, the motor is driven by the amplifier. The amplifier is needed in order to amplify the signal from the xPC Target Box. Once the signal is amplified, the motor can drive the attached load. Various load attached will result in various speed produced. The speed is measured by tachometer and speed signal will be feedback into the xPC Target Box in term of voltage value. The controller designed will compare the acquired speed and the required speed and compensate for the errors occur between those two speeds. The motor speed will be varied depends on the load attached.

### 4.4.1 Amplifier

Amplifier circuit which is developed for the project is shown in Figure 50 below.

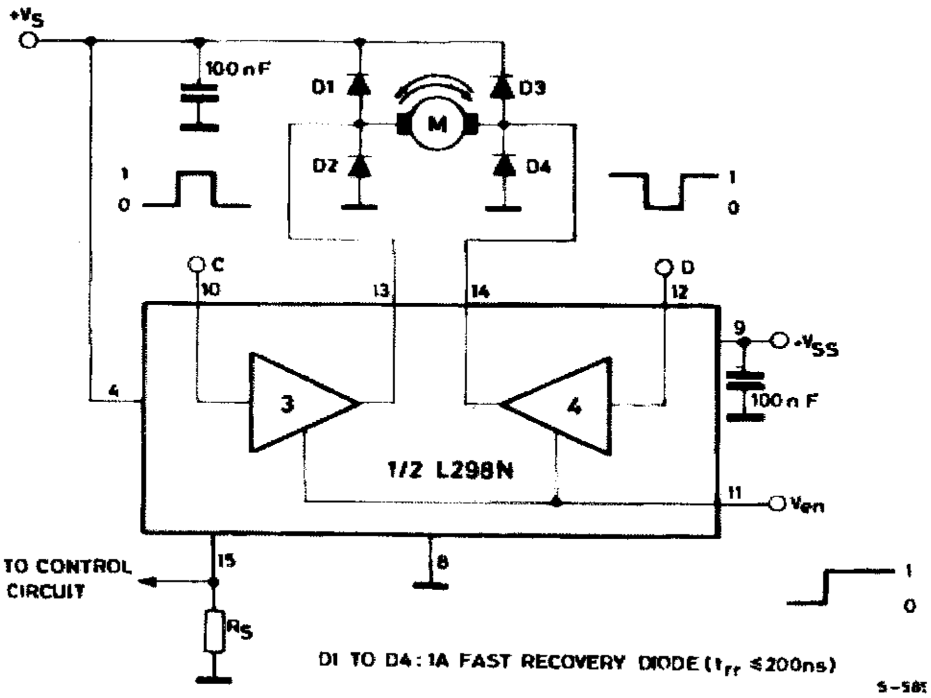



Figure 51: Amplifier for the DC Motor

The amplifier shown above requires Pulse Width Modulation (PWM) signal in order to drive the motor. Vs shown in the figure is for motor power supply while Vss is for IC power supply. The signals from xPC Target Box will coming through point C and D. For this particular amplifier, the motor will be activated based on the table below.

| Inputs | | Function |
|---|---|---|
| $V_{en}$ = H | C = H ; D = L | Forward |
| | C = L ; D = H | Reverse |
| | C = D | Fast Motor Stop |
| $V_{en}$ = L | C = X ; D = X | Free Running Motor Stop |

L = Low H = High X = Don't care

Table 3: Input tables for amplifier

### 4.4.2 Tachometer

The tachometer is driven by the motor. The motor speed will be measured by the tachometer and corresponding voltage value is feedback to the xpC Target Box. The acquired speed is compare to the required speed and the controller will compensate the error occur

## 4.5 Discussion

The SPHEV Simulink was well designed and developed based on the correct vehicle's and components' specification. The model was successfully simulated by using UDDS and acceleration cycle as a standard reference drive cycle which prove the effectiveness of SPHEV as compared to any others vehicles presently. Since it was well simulated, it shows that the right control strategy has been implemented in the SPHEV system. The project signifies the advantages of Hybrid Vehicle compares to the conventional vehicle. Hybrid vehicle contribute to a low fuel consumptions without having to trade off with the vehicle performance and it also contribute to the low emission of vehicle.

For real-time rapid prototyping, theoretically the speed of the DC motor can be controlled in Host PC through xPC Target. The controller is modified in real-time to compensate with the error occurs between acquired and required speed of motor. But, since the xPC Target does not have the ability to supply for PWM input signal which is required by the amplifier, the DC motor could not be run for the purpose of real-time simulation. Further work which is to determine the right amplifier need to be accomplished in order to have a well simulated real-time rapid prototyping.

# CHAPTER 5

# CONCLUSION AND RECOMMENDATION

## 5.1 Conclusion

The SPHEV Simulink was well designed and developed based on the correct vehicle's and components' specification. The model was successfully simulated by using UDDS as a standard reference drive cycle which prove the effectiveness of SPHEV as compared to any others vehicles presently. Since it was well simulated, it shows that the right control strategy has been implemented in the SPHEV system.

Implementation of rapid control prototyping using xPC Target as an interface between the software and hardware was also done successfully. The DC Motor had been used to represent the vehicle and being controlled in real-time application. The same control strategy was also being used to see the responded of the SPHEV system to the different driving conditions. Since the simulation result in real-time were similar as in non real-time, it can be considered that this Final Year Project was successfully completed.

The project signifies the advantages of Hybrid Vehicle compares to the conventional vehicle. Hybrid vehicle contribute to a low fuel consumptions without having to trade off with the vehicle performance and it also contribute to the low emission of vehicle.

## 5.2 Recommendation

The simulink SPHEV model was not a complete model yet. A lot of justifications, calculations and right modeling approach should be made to have a complete model. Due to a very short period of time, it was the best model that can be complete. Some recommendation that can be suggested here are:

- Components specifications and sizing could be made much better since it was roughly done without fully technical details.
- Control strategy of the SPHEV system could be made more specific with implementation of the real control strategy practice in automobile industries. This must be concentrated most since it gives a lot of knowledge and technical ability to the Instrument and control engineer to be.
- More simulation result should be made based on a lot of reference drive cycle.
- Rapid control prototyping hardware should include the others components which are available in small scale such as generator, battery and driver model.

## REFERENCES


[1] Iqbal Husain, "Electric and Hybrid Vehicles Design Fundamentals", CRC Press 2003,USA.

[2] C.M. Jefferson & R.H Barnard, " Hybrid Vehicle Propulsion", WIT Press 2002,UK.

[3] B.K Powell, K.E Bailey and S.R Cikanek, "Dynamic Modeling and Control of Hybrid Electric Vehicle Powertrain Systems", IEEE Control Systems Magazine, October 1998 pp 17-33

[4] A.Halvai Niasar, H.Moghbelli, A.Vahedi, "Design Methodology of Drive Train for a Series Parallel Hybrid Electrical Vehicle and its Powe.r Flow Control Strategy", IEEE.

[5] Keith B.Wipke, Matthew R.Cuddy and Steven D.Burch, "ADVISOR 2.1: A User- Friendly Advanced Power train Simulation using a Combined Backward/Forward Approach", IEEE Transactions on Vehicular Technology, Vol 48, No.6 November 1999.

[6] Salisa Abdul Rahman, "Modeling and Simulation of Energy Management System of Series Hybrid Electrical Vehicle", Universiti Teknologi Petronas, December 2005.

[7] S. Burch, M. Cuddy, V. Johnson, T. Markel, D.Rausen, S. Sprik, K. Wipke, 'ADVISOR 2.21 Documentation: Advanced Vehicle Simulator', National Renewable Energy Laboratory, 2000.

[8] MATLAB User's Guide for xPC Target and xPC Target Box.

[9] MATLAB User's Help

## APPENDICES

### Appendix I:M-File for Urban Dynamometer Driving Schedule

```
% ADVISOR data file:  CYC_UDDS.m
%
% Data source: epa website
%
% Data confirmation:
%
% Notes:
% Represents the Urban Dynamometer Driving Schedule (UDDS), which is
% equivalent to the first two bags of the Federal Test Procedure
(FTP-75).
% In previous versions of ADVISOR, it was called FUDS.  It is
commonly called
% the "LA4", "FTP 72", "EPA II", or "the city test" and represents
% city driving conditions. It is used for light duty vehicle testing.

% Created on: 12-Jun-1998
% By:  SS of NREL
%
% Revision history at end of file.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE ID INFO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
cyc_description='EPA Urban Dynamometer Driving Schedule';
cyc_version=2002; % version of ADVISOR for which the file was
generated
cyc_proprietary=0; % 0=> non-proprietary, 1=> proprietary, do not
distribute
cyc_validation=0; % 0=> no validation, 1=> data agrees with source
data,
% 2=> data matches source data and data collection methods have been
verified
disp(['Data loaded: CYC_UDDS - ',cyc_description])


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SPEED AND KEY POSITION vs. time
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% load variable 'cyc_mph', 2 column matrix with time in the first
column
load CYC_UDDS.mat
% keep key in 'on' position throughout cycle ('1' in the 2nd column
=> 'on')
vc_key_on=[cyc_mph(:,1) ones(size(cyc_mph,1),1)];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OTHER DATA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Size of 'window' used to filter the trace with centered-in-time
averaging;
```

```
% higher numbers mean more smoothing and less rigorous following of
the trace.
% Used when cyc_filter_bool=1
cyc_avg_time=3;  % (s)
cyc_filter_bool=0;  % 0=> no filtering, follow trace exactly; 1=>
smooth trace
cyc_grade=0;     %no grade associated with this cycle
cyc_elevation_init=0; %the initial elevation in meters.

if size(cyc_grade,1)<2
   % convert cyc_grade to a two column matrix, grade vs. dist
   cyc_grade=[0 cyc_grade; 1 cyc_grade]; % use this for a constant
roadway grade
end

% A constant zero delta in cargo-mass:
% First column is distance (m) second column is mass (kg)
cyc_cargo_mass=[0 0
   1 0];

if size(cyc_cargo_mass,1)<2
   % convert cyc_grade to a two column matrix, grade vs. dist
   cyc_cargo_mass=[0 cyc_cargo_mass; 1 cyc_cargo_mass]; % use this
for a constant roadway grade
end

if exist('cyc_coast_gb_shift_delay')
    gb_shift_delay=cyc_coast_gb_shift_delay; % restore the original
gb_shift_delay which may have been changed by cyc_coast
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REVISION HISTORY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 23-Jun-1998 (MRC):  cosmetic changes
% 3/15/99:ss updated *_version to 2.1 from 2.0
% 11/3/99:ss updated version to 2.21
% 7/5/00: ss, tm: Was named fuds instead of udds. Updated data in
CYC_UDDS.mat to that from EPA site.
% 8/15/00 AB:  Updated description to inform FUDS to UDDS and other
"also known as" names.
% 8/20/01: mpo added special case code to reset gb_shift_delay if
changed by cyc_coast
```

## Appendix II:M-File for Energy Storage System ( Battery)

```
% ADVISOR data file:  ESS_PB28.m    where 28 is c/5-rate capacity
%
% Data source:
% Testing done by University of Illinois
%
% Data confirmation:
%
% Notes:
% These parameters describe the Johnson Controls 12-95 lead-acid
battery.  This
% data was provided by the University of Illinois Urbana/Champaign
under
% subcontract #XCB-5-15296-01 to NREL.
%
% Created on: 30-Jun-1998
% By:  MRC, NREL, matthew_cuddy@nrel.gov
%
% Revision history at end of file.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE ID INFO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_description='Johnson Controls lead-acid battery, tested by U of
Illinois';
ess_version=2002; % version of ADVISOR for which the file was
generated
ess_proprietary=0; % 0=> non-proprietary, 1=> proprietary, do not
distribute
ess_validation=0; % 0=> no validation, 1=> data agrees with source
data,
% 2=> data matches source data and data collection methods have been
verified
disp(['Data loaded: ESS_PB28 - ',ess_description])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Temperature range over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_tmp=[0 22 40];  % (C)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SOC RANGE over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_soc=[0:.2:1];  % (--)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LOSS AND EFFICIENCY parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Parameters vary by SOC horizontally, and temperature vertically
ess_max_ah_cap=[
   28
   28
   28
];  % (A*h), max. capacity at C/5 rate, indexed by ess_tmp
```

```
% average coulombic (a.k.a. amp-hour) efficiency below, indexed by
ess_tmp
ess_coulombic_eff=[
   .9
   .9
   .9
];  % (--);
% module's resistance to being discharged, indexed by ess_soc and
ess_tmp
ess_r_dis=[
   0.038 0.024 0.007 0.007 0.007 0.011
   0.038 0.024 0.007 0.007 0.007 0.011
   0.038 0.024 0.007 0.007 0.007 0.011
]; % (ohm)
% module's resistance to being charged, indexed by ess_soc and
ess_tmp
ess_r_chg=ess_r_dis; %no other data available

% module's open-circuit (a.k.a. no-load) voltage, indexed by ess_soc
and ess_tmp
ess_voc=[
   6.0 8.9 11.8 12.0 12.3 12.6
   6.0 8.9 11.8 12.0 12.3 12.6
   6.0 8.9 11.8 12.0 12.3 12.6
]; % (V) voc at low soc seems low compared with other lead acid
batteries, use with caution

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LIMITS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_min_volts=5.5;% caution, this value may be too low(compared with
other lead acid batteries, see also
%                                 ess_voc note.
ess_max_volts=16.5;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OTHER DATA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_module_mass=11.8;  % (kg), mass of a single ~12 V module

ess_module_num=25;  %a default value for number of modules

ess_cap_scale=1; % scale factor for module max ah capacity

% user definable mass scaling relationship
ess_mass_scale_fun=inline('(x(1)*ess_module_num+x(2))*(x(3)*ess_cap_s
cale+x(4))*(ess_module_mass)','x','ess_module_num','ess_cap_scale','e
ss_module_mass');
ess_mass_scale_coef=[1 0 1 0]; % coefficients in ess_mass_scale_fun

% user definable resistance scaling relationship
ess_res_scale_fun=inline('(x(1)*ess_module_num+x(2))/(x(3)*ess_cap_sc
ale+x(4))','x','ess_module_num','ess_cap_scale');
ess_res_scale_coef=[1 0 1 0]; % coefficients in ess_res_scale_fun

% battery thermal model
ess_th_calc=1;                              % --     0=no ess thermal
calculations, 1=do calc's
```

```
ess_mod_cp=660;                              % J/kgK  ave heat capacity
of module (typical Pb bat - from Optima)
ess_set_tmp=35;                              % C      thermostat temp
of module when cooling fan comes on
ess_area_scale=(ess_module_mass/11)^0.7;     % --     if module
dimensions are unknown, assume rectang shape and scale vs PB25
ess_mod_sarea=0.2*ess_area_scale;            % m^2    total module
surface area exposed to cooling air (typ rectang module)
ess_mod_airflow=0.01;                        % kg/s   cooling air mass
flow rate across module (20 cfm=0.01 kg/s at 20 C)
ess_mod_flow_area=0.005*ess_area_scale;      % m^2    cross-sec flow
area for cooling air per module (assumes 10-mm gap btwn mods)
ess_mod_case_thk=2/1000;                     % m      thickness of
module case (typ from Optima)
ess_mod_case_th_cond=0.20;                   % W/mK   thermal
conductivity of module case material (typ polyprop plastic - Optima)
ess_air_vel=ess_mod_airflow/(1.16*ess_mod_flow_area); % m/s  ave
velocity of cooling air
ess_air_htcoef=30*(ess_air_vel/5)^0.8;        % W/m^2K cooling air heat
transfer coef.
ess_th_res_on=((1/ess_air_htcoef)+(ess_mod_case_thk/ess_mod_case_th_c
ond))/ess_mod_sarea; % K/W  tot thermal res key on
ess_th_res_off=((1/4)+(ess_mod_case_thk/ess_mod_case_th_cond))/ess_mo
d_sarea; % K/W  tot thermal res key off (cold soak)
% set bounds on flow rate and thermal resistance
ess_mod_airflow=max(ess_mod_airflow,0.001);
ess_th_res_on=min(ess_th_res_on,ess_th_res_off);
clear ess_mod_sarea ess_mod_flow_area ess_mod_case_thk
ess_mod_case_th_cond ess_air_vel ess_air_htcoef ess_area_scale

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REVISION HISTORY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 3/4/98 (DR): converted W-hr values to A-hr values (in E_JC1295.m)
% 10/13/98:mc recomputed ess_max_ah_cap and renamed file (from
ESS_PB25)
%             for consistency with other files
% 02/09/99 (SDB):  added thermal model inputs
% 2/4/99 ss: added ess_module_num=25;
% 3/15/99:ss updated *_version to 2.1 from 2.0
% 8/5/99:ss deleted all peukert coefficient and data(no longer needed
in advisor model)
%           added limits 'ess_max_volt' and 'ess_min_volt'
%9/9/99: vhj changed variables to include thermal modeling (matrices,
not vector), added ess_tmp
% 11/03/99:ss updated version from 2.2 to 2.21
% 7/30/01:tm added user defineable scaling functions for
mass=f(ess_module_num,ess_cap_scale,ess_module_mass)
%           and
resistance=f(ess_module_num,ess_cap_scale)*base_resistance
% ADVISOR data file:  ESS_PB28.m    where 28 is c/5-rate capacity
%
% Data source:
% Testing done by University of Illinois
%
% Data confirmation:
%
% Notes:
% These parameters describe the Johnson Controls 12-95 lead-acid
battery.  This
```

```
% data was provided by the University of Illinois Urbana/Champaign
under
% subcontract #XCB-5-15296-01 to NREL.
%
% Created on: 30-Jun-1998
% By:  MRC, NREL, matthew_cuddy@nrel.gov
%
% Revision history at end of file.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE ID INFO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_description='Johnson Controls lead-acid battery, tested by U of
Illinois';
ess_version=2002; % version of ADVISOR for which the file was
generated
ess_proprietary=0; % 0=> non-proprietary, 1=> proprietary, do not
distribute
ess_validation=0; % 0=> no validation, 1=> data agrees with source
data,
% 2=> data matches source data and data collection methods have been
verified
disp(['Data loaded: ESS_PB28 - ',ess_description])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Temperature range over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_tmp=[0 22 40];  % (C)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SOC RANGE over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_soc=[0:.2:1];  % (--)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LOSS AND EFFICIENCY parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Parameters vary by SOC horizontally, and temperature vertically
ess_max_ah_cap=[
   28
   28
   28
];  % (A*h), max. capacity at C/5 rate, indexed by ess_tmp
% average coulombic (a.k.a. amp-hour) efficiency below, indexed by
ess_tmp
ess_coulombic_eff=[
   .9
   .9
   .9
];  % (--);
% module's resistance to being discharged, indexed by ess_soc and
ess_tmp
ess_r_dis=[
   0.038 0.024 0.007 0.007 0.007 0.011
   0.038 0.024 0.007 0.007 0.007 0.011
   0.038 0.024 0.007 0.007 0.007 0.011
]; % (ohm)
```

```
% module's resistance to being charged, indexed by ess_soc and
ess_tmp
ess_r_chg=ess_r_dis; %no other data available

% module's open-circuit (a.k.a. no-load) voltage, indexed by ess_soc
and ess_tmp
ess_voc=[
   6.0 8.9 11.8 12.0 12.3 12.6
   6.0 8.9 11.8 12.0 12.3 12.6
   6.0 8.9 11.8 12.0 12.3 12.6
]; % (V) voc at low soc seems low compared with other lead acid
batteries, use with caution

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LIMITS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_min_volts=5.5;% caution, this value may be too low(compared with
other lead acid batteries, see also
%                                  ess_voc note.
ess_max_volts=16.5;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OTHER DATA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ess_module_mass=11.8;  % (kg), mass of a single ~12 V module

ess_module_num=25;  %a default value for number of modules

ess_cap_scale=1; % scale factor for module max ah capacity

% user definable mass scaling relationship
ess_mass_scale_fun=inline('(x(1)*ess_module_num+x(2))*(x(3)*ess_cap_s
cale+x(4))*(ess_module_mass)','x','ess_module_num','ess_cap_scale','e
ss_module_mass');
ess_mass_scale_coef=[1 0 1 0]; % coefficients in ess_mass_scale_fun

% user definable resistance scaling relationship
ess_res_scale_fun=inline('(x(1)*ess_module_num+x(2))/(x(3)*ess_cap_sc
ale+x(4))','x','ess_module_num','ess_cap_scale');
ess_res_scale_coef=[1 0 1 0]; % coefficients in ess_res_scale_fun

% battery thermal model
ess_th_calc=1;                           % --      0=no ess thermal
calculations, 1=do calc's
ess_mod_cp=660;                          % J/kgK  ave heat capacity
of module (typical Pb bat - from Optima)
ess_set_tmp=35;                          % C       thermostat temp
of module when cooling fan comes on
ess_area_scale=(ess_module_mass/11)^0.7;  % --     if module
dimensions are unknown, assume rectang shape and scale vs PB25
ess_mod_sarea=0.2*ess_area_scale;        % m^2    total module
surface area exposed to cooling air (typ rectang module)
ess_mod_airflow=0.01;                    % kg/s   cooling air mass
flow rate across module (20 cfm=0.01 kg/s at 20 C)
ess_mod_flow_area=0.005*ess_area_scale;  % m^2    cross-sec flow
area for cooling air per module (assumes 10-mm gap btwn mods)
ess_mod_case_thk=2/1000;                 % m      thickness of
module case (typ from Optima)
```

```
ess_mod_case_th_cond=0.20;                    % W/mK   thermal
conductivity of module case material (typ polyprop plastic - Optima)
ess_air_vel=ess_mod_airflow/(1.16*ess_mod_flow_area); % m/s  ave
velocity of cooling air
ess_air_htcoef=30*(ess_air_vel/5)^0.8;       % W/m^2K cooling air heat
transfer coef.
ess_th_res_on=((1/ess_air_htcoef)+(ess_mod_case_thk/ess_mod_case_th_c
ond))/ess_mod_sarea; % K/W  tot thermal res key on
ess_th_res_off=((1/4)+(ess_mod_case_thk/ess_mod_case_th_cond))/ess_mo
d_sarea; % K/W  tot thermal res key off (cold soak)
% set bounds on flow rate and thermal resistance
ess_mod_airflow=max(ess_mod_airflow,0.001);
ess_th_res_on=min(ess_th_res_on,ess_th_res_off);
clear ess_mod_sarea ess_mod_flow_area ess_mod_case_thk
ess_mod_case_th_cond ess_air_vel ess_air_htcoef ess_area_scale


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REVISION HISTORY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 3/4/98 (DR): converted W-hr values to A-hr values (in E_JC1295.m)
% 10/13/98:mc recomputed ess_max_ah_cap and renamed file (from
ESS_PB25)
%              for consistency with other files
% 02/09/99 (SDB):  added thermal model inputs
% 2/4/99 ss: added ess_module_num=25;
% 3/15/99:ss updated *_version to 2.1 from 2.0
% 8/5/99:ss deleted all peukert coefficient and data(no longer needed
in advisor model)
%            added limits 'ess_max_volt' and 'ess_min_volt'
%9/9/99: vhj changed variables to include thermal modeling (matrices,
not vector), added ess_tmp
% 11/03/99:ss updated version from 2.2 to 2.21
% 7/30/01:tm added user defineable scaling functions for
mass=f(ess_module_num,ess_cap_scale,ess_module_mass)
%            and
resistance=f(ess_module_num,ess_cap_scale)*base_resistance
```

## Appendix III:M-File for Internal Combustion Engine

```
% ADVISOR Data file:  FC_SI41_emis.M
%
% Data source: J. Dill Murrel, JDM & Associates.
%
% Data confidence level:
%
% Notes: 1991 Geo Metro 1.0L SI engine.
% Maximum Power 41 kW @ 5700 rpm.
% Peak Torque 81 Nm @ 3477 rpm.

% WARNING:  This data comes from transient testing on the FTP and is
% only appropriate to model transient-operation engines.
%
% Created on:  06/22/98
% By:  Tony Markel, National Renewable Energy Laboratory,
Tony_Markel@nrel.gov
%
% Revision history at end of file.
%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE ID INFO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fc_description='Geo 1.0L (41kW) SI Engine - transient data';
fc_version=2002; % version of ADVISOR for which the file was
generated
fc_proprietary=0; % 0=> non-proprietary, 1=> proprietary, do not
distribute
fc_validation=0; % 1=> no validation, 1=> data agrees with source
data,
% 2=> data matches source data and data collection methods have been
verified
fc_fuel_type='Gasoline';
fc_disp=1.0;  % (L), engine displacement
fc_emis=1;      % boolean 0=no emis data; 1=emis data
fc_cold=0;      % boolean 0=no cold data; 1=cold data exists
disp(['Data loaded: FC_SI41_emis.M - ',fc_description]);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SPEED & TORQUE RANGES over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (rad/s), speed range of the engine
fc_map_spd=[104.5 149.2 220.9 292.5 364.1 435.7 507.4 552.2 596.9];

% (N*m), torque range of the engine
fc_map_trq=[6.8 13.6 20.4 27.2 33.8 40.6 47.4 54.2 61 67.8 74.6
81.4];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FUEL USE AND EMISSIONS MAPS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (g/s), fuel use map indexed vertically by fc_map_spd and
% horizontally by fc_map_trq
fc_fuel_map_gpkWh =[
```

```
635.7    635.7    541.4    447.2    352.9    332.2    311.4    322.4    333.5
333.5    333.5    333.5
678.4    500.1    443.8    387.4    331.1    301.8    297 283.4    269.8
358 358 358
463.4    463.4    407.6    350.1    294.3    280.8    267.3    253.9    269.8
303.2    336.7    336.7
699.1    567.9    500.3    432.7    301.4    283.9    266.3    248.7    258.8
268.8    271.9    317.9
592.9    592.9    494.6    393.4    295.1    279.4    263.6    247.9    255.2
262.5    295 322.6
667.9    524.8    381.6    351.9    322.2    304.9    287.5    270.8    290.8
310.9    330.9    330.9
630.6    630.6    522.5    411.1    303 304.4    305.8    304.2    314.5
324.8    327.7    327.7
698.4    500.5    428.6    392.7    356.8    337.9    328.4    319 328.8
338.6    333.7    333.7
751.1    637.8    521.1    407.8    393.1    378.4    363.3    348.2    318.8
340.2    340.2    340.2];
% fuel map in g/kWh

% (g/s), engine out HC emissions indexed vertically by fc_map_spd and
% horizontally by fc_map_trq
fc_hc_map_gpkWh =[
11.5    11.5    9.8 8.2 6.5 5.8 5.1 5.9 6.8 6.8 6.8 6.8
7.8 7   6.2 5.5 4.7 4.3 4.7 4.6 4.5 4.6 4.6 4.6
5.8 5.8 5.2 4.6 4   4   4   4   4.5 4.6 4.6 4.6
7.1 6   5.4 4.9 3.8 3.7 3.6 3.4 3.2 3   3.4 3.9
5.8 5.8 5   4.3 3.6 3.6 3.6 3.7 3.6 3.6 4   3.9
5.6 4.7 3.7 3.7 3.7 3.4 3.1 3   3.2 3.4 3.5 3.5
8.2 8.2 6.8 5.4 4.1 3.7 3.3 3.1 3.2 3.2 3.3 3.3
5.8 5.2 4.8 4.5 4.3 3.7 3.4 3.2 3.3 3.3 3.3 3.3
5.6 5.8 5.9 6.1 5.7 5.4 5   4.6 3.9 3.9 3.9 3.9];
% engine out HC in g/kWh

% (g/s), engine out CO emissions indexed vertically by fc_map_spd and
% horizontally by fc_map_trq
fc_co_map_gpkWh =[
71.8     71.8     58.8     45.7     32.7     27.5     22.4     82.9     143.3
143.3    143.3    143.3
104.4    68.3     56.8     45.3     33.9     23.3     25.7     24.2     22.8
268.6    268.6    268.6
48.3     48.3     42.9     37.2     31.8     28.6     25.4     22.2     22.8
152.9    283 283
103.1    82.7     72.2     61.8     41.5     36.9     32.3     27.8     31.1
34.4     178.5    279.9
88.1     88.1     74.2     59.9     46  41.9     37.8     33.7     34.8
35.8     158.8    264.6
96.1     74.5     52.9     51.2     49.5     45.9     42.4     34  117.9
201.8    285.7    285.7
114.6    114.6    92.1     69  46.5     60.7     74.8     129.6    195.5
261.4    277.8    277.8
60.1     63.8     64  64.1     64.3     108.2    130.2    152.2    216.7
281.2    278.1    278.1
51.8     75.2     99.3     122.8    134.9    147.1    159.7    172.2    196.6
286.6    286.6    286.6];
% engine out CO in g/kWh

% (g/s), engine out NOx emissions indexed vertically by fc_map_spd
and
% horizontally by fc_map_trq
```

```
fc_nox_map_gpkWh =[
5.8 5.8 9.3 12.8    16.3    16.1    15.9    13  10.2    10.2    10.2
10.2
5.2 8.8 9.2 9.7 10.2    8   8.9 13.2    17.5    4.6 4.6 4.6
8.1 8.1 8.8 9.6 10.4    10.8    11.3    11.7    17.5    11.6    5.7
5.7
4.2 5.6 6.3 7   8.4 8.9 9.5 10.1    13.9    17.7    8.1 3.1
5.8 5.8 7.2 8.7 10.1    11  11.8    12.6    15.9    19.2    9.3 6.8
14.9    16.4    17.8    19.4    21  20.6    20.3    19.1    14.6
10.2    5.7 5.7
28.7    28.7    26.8    25  23.1    22.4    21.7    16.5    12.1
7.8 6.5 6.5
31.1    27.9    26.7    26.2    25.6    20.9    18.6    16.3    12.1
7.8 6.8 6.8
35  31.1    27.1    23.2    21.1    19.1    17  14.9    10.9    7.4
7.4 7.4];
% engine out NOx in g/kWh

% (g/s), engine out PM emissions indexed vertically by fc_map_spd and
% horizontally by fc_map_trq
fc_pm_map_gpkWh=zeros(size(fc_fuel_map_gpkWh));

% (g/s), engine out O2 indexed vertically by fc_map_spd and
% horizontally by fc_map_trq
fc_o2_map=zeros(size(fc_fuel_map_gpkWh));


% convert g/kWh maps to g/s maps
[T,w]=meshgrid(fc_map_trq, fc_map_spd);
fc_map_kW=T.*w/1000;
fc_fuel_map=fc_fuel_map_gpkWh.*fc_map_kW/3600;
fc_hc_map=fc_hc_map_gpkWh.*fc_map_kW/3600;
fc_co_map=fc_co_map_gpkWh.*fc_map_kW/3600;
fc_nox_map=fc_nox_map_gpkWh.*fc_map_kW/3600;
fc_pm_map=fc_pm_map_gpkWh.*fc_map_kW/3600;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cold Engine Maps
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fc_cold_tmp=20; %deg C
fc_fuel_map_cold=zeros(size(fc_fuel_map));
fc_hc_map_cold=zeros(size(fc_fuel_map));
fc_co_map_cold=zeros(size(fc_fuel_map));
fc_nox_map_cold=zeros(size(fc_fuel_map));
fc_pm_map_cold=zeros(size(fc_fuel_map));
%Process Cold Maps to generate Correction Factor Maps
names={'fc_fuel_map','fc_hc_map','fc_co_map','fc_nox_map','fc_pm_map'
};
for i=1:length(names)
    %cold to hot raio, e.g. fc_fuel_map_c2h = fc_fuel_map_cold ./
fc_fuel_map
    eval([names{i},'_c2h=',names{i},'_cold./(',names{i},'+eps);'])
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LIMITS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (N*m), max torque curve of the engine indexed by fc_map_spd
fc_max_trq=[61  67.6 73.7 78.5 80.9 77.3 76.2 73.3 68.7];
```

```
% (N*m), closed throttle torque of the engine (max torque that can be
absorbed)
% indexed by fc_map_spd -- correlation from JDMA
fc_ct_trq=4.448/3.281*(-fc_disp)*61.02/24 * ...
   (9*(fc_map_spd/max(fc_map_spd)).^2 + 14 *
(fc_map_spd/max(fc_map_spd)));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DEFAULT SCALING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (--), used to scale fc_map_spd to simulate a faster or slower
running engine
fc_spd_scale=1.0;
% (--), used to scale fc_map_trq to simulate a higher or lower torque
engine
fc_trq_scale=1.0;
fc_pwr_scale=fc_spd_scale*fc_trq_scale;   % --  scale fc power

% user definable mass scaling function
fc_mass_scale_fun=inline('(x(1)*fc_trq_scale+x(2))*(x(3)*fc_spd_scale
+x(4))*(fc_base_mass+fc_acc_mass)+fc_fuel_mass','x','fc_spd_scale','f
c_trq_scale','fc_base_mass','fc_acc_mass','fc_fuel_mass');
fc_mass_scale_coef=[1 0 1 0]; % coefficients of mass scaling function

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% STUFF THAT SCALES WITH TRQ & SPD SCALES (MASS AND INERTIA)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fc_inertia=0.1*fc_pwr_scale;           % (kg*m^2), rotational inertia
of the engine (unknown)
fc_max_pwr=(max(fc_map_spd.*fc_max_trq)/1000)*fc_pwr_scale; % kW
peak engine power

fc_base_mass=1.8*fc_max_pwr;            % (kg), mass of the engine
block and head (base engine)
                                        %       mass penalty of 1.8
kg/kW from 1994 OTA report, Table 3
fc_acc_mass=0.8*fc_max_pwr;             % kg    engine accy's,
electrics, cntrl's - assumes mass penalty of 0.8 kg/kW (from OTA
report)
fc_fuel_mass=0.6*fc_max_pwr;            % kg    mass of fuel and fuel
tank (from OTA report)
fc_mass=fc_base_mass+fc_acc_mass+fc_fuel_mass; % kg  total
engine/fuel system mass
fc_ext_sarea=0.3*(fc_max_pwr/100)^0.67;       % m^2    exterior
surface area of engine



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OTHER DATA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fc_fuel_den=0.749*1000; % (g/l), density of the fuel
fc_fuel_lhv=42.6*1000; % (J/g), lower heating value of the fuel

%the following was added for the new thermal modeling of the engine
12/17/98 ss and sb
fc_tstat=96;                  % C      engine coolant thermostat set
temperature (typically 95 +/- 5 C)
```

```
fc_cp=500;                    % J/kgK  ave cp of engine (iron=500, Al
or Mg = 1000)
fc_h_cp=500;                  % J/kgK  ave cp of hood & engine
compartment (iron=500, Al or Mg = 1000)
fc_hood_sarea=1.5;            % m^2    surface area of hood/eng
compt.
fc_emisv=0.8;                 %        emissivity of engine ext
surface/hood int surface
fc_hood_emisv=0.9;            %        emissivity hood ext
fc_h_air_flow=0.0;            % kg/s   heater air flow rate (140
cfm=0.07)
fc_cl2h_eff=0.7;              % --     ave cabin heater HX eff (based
on air side)
fc_c2i_th_cond=500;           % W/K    conductance btwn engine cyl &
int
fc_i2x_th_cond=500;           % W/K    conductance btwn engine int &
ext
fc_h2x_th_cond=10;            % W/K    conductance btwn engine &
engine compartment

% calc "predicted" exh gas flow rate and engine-out (EO) temp
fc_ex_pwr_frac=[0.40 0.30];                        % --   frac of
waste heat that goes to exhaust as func of engine speed
fc_exflow_map=fc_fuel_map*(1+14.5);                % g/s  ex gas flow
map:  for SI engines, exflow=(fuel use)*[1 + (stoic A/F ratio)]
fc_waste_pwr_map=fc_fuel_map*fc_fuel_lhv - T.*w;   % W    tot FC
waste heat = (fuel pwr) - (mech out pwr)
spd=fc_map_spd;
fc_ex_pwr_map=zeros(size(fc_waste_pwr_map));       % W   initialize
size of ex pwr map
for i=1:length(spd)
 fc_ex_pwr_map(i,:)=fc_waste_pwr_map(i,:)*interp1([min(spd)
max(spd)],fc_ex_pwr_frac,spd(i)); % W  trq-spd map of waste heat to
exh
end
fc_extmp_map=fc_ex_pwr_map./(fc_exflow_map*1089/1000) + 20;  % W   EO
ex gas temp = Q/(MF*cp) + Tamb (assumes engine tested ~20 C)

%the following variable is not used directly in modelling and should
always be equal to one
%it's used for initialization purposes
fc_eff_scale=1;

% clean up workspace
clear T w fc_waste_pwr_map fc_ex_pwr_map spd fc_map_kW

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REVISION HISTORY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 06/23/98 (tm): created from a_dodg31.m
% 07/06/98 (MC): corrected max power calc. in mass calc.
%                renamed fc_init_coolant_temp to fc_coolant_init_temp
% 07/17/98 (tm): file renamed FC_SI102.M
% 07/16/98 (SS): added variable fc_fuel_type under file id section
% 07/17/98 (tm): fc_fuel_den changed from 0.737 to 0.749 and
fc_fuel_lhv changed from 42.7 to 42.6
% 07/30/98 (sb): added A/F ratio and split of waste heat variables
% 10/9/98 (vh,sb,ss): added pm and removed init conditions and added
new exhaust variables
% 10/13/98 (MC): added variable fc_disp under file id section
```

```
%                fc_ct_trq computed according to correlation from
JDMA, 5/98
% 10/13/98 (MC): updated equation for fc_ct_trq (convert from ft-lb
to Nm)
% 12/17/98 ss,sb: added 12 new variables for engine thermal
modelling.
% 01/25/99 (SB): modified thermal section to work with new BD,
revised FC mass calc's
% 2/4/99: ss,sb changed fc_ext_sarea=0.3*(fc_max_pwr/100)^0.67  it
was 0.3*(fc_max_pwr/100)
%        it now takes into account that surface area increases based
on mass to the 2/3 power
% 3/15/99:ss updated *_version to 2.1 from 2.0
% 7/6/99:tm removed clear statement for all *gpkWh data - now used in
plots
% 7/9/99:tm cosmetic changes
% 11/03/99:ss updated version from 2.2 to 2.21
% 01/31/01: vhj added fc_cold=0, added cold map variables, added +eps
to avoid dividing by zero
% 02/26/01: vhj added variable definition of fc_o2_map (used in NOx
absorber emis.)
% 7/30/01:tm added user definable mass scaling function
mass=f(fc_spd_scale,fc_trq_scale,fc_base_mass,fc_acc_mass,fc_fuel_mas
s)
```

## Appendix IV:M-File for Electric Generator

```
% ADVISOR data file:  GC_PM32.m
%
% Data source:
% Unique Mobility specification sheet for the SR180p/CR20-300
% motor/controller combination at 195 V, dated 10/28/94
%
% Data confirmation:
%
% Notes:
% This is the same machine/inverter combination as is in MC_PM32
%
% Created on: 2-Sep-1998
% By:  MRC, NREL, matthew_cuddy@nrel.gov
%
% Revision history at end of file.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE ID INFO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
gc_description='Unique Mobility 32-kW permanent magnet
motor/controller';
gc_version=2002; % version of ADVISOR for which the file was
generated
gc_proprietary=0; % 0=> non-proprietary, 1=> proprietary, do not
distribute
gc_validation=0; % 0=> no validation, 1=> data agrees with source
data,
% 2=> data matches source data and data collection methods have been
verified
disp(['Data loaded: GC_PM32 - ',gc_description])


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SPEED & TORQUE RANGES over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (N*m), torque vector corresponding to columns of efficiency & loss
maps
% this is INPUT torque (>0 => running as a generator)
gc_map_trq=[0 40 80 120 160 200 240 320 400 480 520]*4.448/3.281/12;

% (rad/s), speed vector corresponding to rows of efficiency & loss
maps
gc_map_spd=[0 500 1000 1500 2000 2500 3000 4000 5000 6000
7000]*(2*pi)/60;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LOSSES AND EFFICIENCIES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
gc_eff_map=[
0.200   0.200   0.200   0.200   0.200   0.200   0.200   0.200   0.200
0.200   0.200
0.200   0.380   0.490   0.520   0.570   0.600   0.600   0.520   0.450
0.430   0.430
```

```
0.200   0.500   0.620   0.670   0.715   0.725   0.730   0.720   0.710
0.700   0.700
0.200   0.520   0.650   0.710   0.740   0.765   0.770   0.775   0.775
0.767   0.767
0.200   0.540   0.670   0.730   0.770   0.785   0.780   0.800   0.805
0.808   0.808
0.200   0.580   0.700   0.760   0.785   0.810   0.820   0.830   0.835
0.830   0.830
0.200   0.590   0.720   0.770   0.800   0.825   0.835   0.845   0.847
0.846   0.846
0.200   0.600   0.755   0.805   0.830   0.845   0.854   0.865   0.866
0.868   0.868
0.200   0.600   0.770   0.815   0.840   0.860   0.867   0.883   0.888
0.887   0.887
0.200   0.550   0.775   0.830   0.860   0.870   0.884   0.897   0.905
0.910   0.910
0.200   0.500   0.760   0.840   0.870   0.885   0.893   0.905   0.915
0.920   0.920
];% (--), efficiency of the machine/inverter when run as a motor

%
% convert to losses, assuming losses are symmetric about zero torque
%
[T1,w1]=meshgrid(gc_map_trq,gc_map_spd);
gc_mech_pwr_map=T1.*w1; % (W), output power (when motoring) for each
trq and spd
temp=gc_mech_pwr_map./gc_eff_map; % input power (when motoring)
gc_loss_map=temp-gc_mech_pwr_map; % (W) loss corresponding to each
trq and spd

%
% assume that losses at zero torque and speed are the same as nearest
neighbors
%
gc_loss_map(1,:)=gc_loss_map(2,:); % loss at zero spd = loss at
lowest +ive spd
gc_loss_map(:,1)=gc_loss_map(:,2); % loss at zero trq = loss at
lowest +ive trq

%
% convert loss map to output power map for machine running as a
generator
%
gc_outpwr_map=gc_mech_pwr_map-gc_loss_map;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LIMITS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
gc_max_crrnt=300;    % maximum current draw for motor/controller set,
A
gc_min_volts=60;     % minimum voltage for motor/controller set, V
% maximum continuous torque corresponding to speeds in gc_map_spd
gc_max_trq=[512 508.1 504.2 500.3 496.3 492.4 488.5 480.7 472.8 465
0]...
   *4.448/3.281/12; % (N*m)
% factor by which motor torque can exceed maximum continuous torque
for short
% periods of time
gc_overtrq_factor=320/220;  % (--)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DEFAULT SCALING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (--), used to scale fc_map_spd to simulate a faster or slower
running engine
gc_spd_scale=1.0;
% (--), used to scale fc_map_trq to simulate a higher or lower torque
engine
gc_trq_scale=1.0;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OTHER DATA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
gc_inertia=0.0226; % (kg*m^2), rotor's rotational inertia
gc_mass=23.6+14.5; % (kg), mass of machine and controller


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CLEAN UP
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear T1 w1 gc_inpwr_map temp gc_loss_map gc_mech_pwr_map


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REVISION HISTORY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 3/15/99:ss updated *_version to 2.1 from 2.0

% 11/03/99:ss updated version from 2.2 to 2.21
% Begin added by ADVISOR 3.2 converter: 30-Jul-2001
gc_mass_scale_coef=[1 0 1 0];

gc_mass_scale_fun=inline('(x(1)*gc_trq_scale+x(2))*(x(3)*gc_spd_scale
+x(4))*gc_mass','x','gc_spd_scale','gc_trq_scale','gc_mass');

% End added by ADVISOR 3.2 converter: 30-Jul-2001
```

## Appendix V:M-File for Electric Motor

```
%%%%% ADVISOR data file:  MC_PM49
%
% Data source:
% Honda R&D Americas
%
% Created on: 2/20/99
%
% By: Anil Paryani (Honda R&D Americas), aparyani@hra.com
% Revision history at end of file.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE ID INFO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mc_version=2002;
mc_description='Honda 49 KW (continuous), permanent magnet
motor/controller';
mc_proprietary=0; % 0=> non-proprietary, 1=> proprietary, do not
distribute
mc_validation=1; % 0=> no validation, 1=> data agrees with source
data,
% 2=> data matches source data and data collection methods have been
verified
disp(['Data loaded: MC_PM49 - ',mc_description]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SPEED & TORQUE RANGES over which data is defined
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (N*m), torque range of the motor
mc_map_trq=[-176.4 -156.8 -137.2 -117.6 -98 -78.4 -58.8 -39.2 -19.6
0.0...
    19.6    39.2    78.4    98.0    117.6 137.2 156.8 176.4 235.2
274.4];

% (rad/s), speed range of the motor
mc_map_spd=[0 500 1000 1500 2000 2500 3000 3500 4000 4500 5000 5500
...
      6000 6500 7000 7500 8000 8500]*(2*pi)/60;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LOSSES AND EFFICIENCIES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mc_eff_map = 0.01*[...
54.17   56.09   59.74   62.16   64.71   64.88   66.49   68.30   63.07
63.07   87.76   84.71   79.49   78.10   76.56   75.09   73.90   71.33
63.88   59.75
54.17   56.09   59.74   62.16   64.71   64.88   66.49   68.30   63.07
63.07   87.76   84.71   79.49   78.10   76.56   75.09   73.90   71.33
63.88   59.75
70.00   71.77   75.20   78.37   80.62   82.73   84.62   85.31   80.23
80.23   85.98   86.96   87.34   86.64   85.45   84.73   84.03   83.26
80.81   77.35
79.08   80.25   82.73   84.76   86.91   87.56   87.27   87.20   80.24
80.24   87.45   88.53   89.23   89.37   88.36   88.08   87.98   87.33
85.65   82.47
83.36   84.27   86.74   88.36   89.34   90.20   90.39   89.14   81.05
81.05   90.54   90.31   90.33   90.42   90.38   90.13   89.86   89.38
87.95   87.25
86.38   87.62   88.89   90.36   90.71   91.07   91.08   89.20   83.52
83.52   88.41   91.83   91.51   91.56   91.43   91.28   91.02   91.23
90.67   90.67
90.83   90.83   91.04   91.41   92.60   91.95   92.22   90.68   84.90
84.90   90.61   91.38   92.36   92.29   92.35   92.16   92.12   93.52
93.61   93.61
92.78   92.78   92.78   92.78   93.06   93.10   92.21   91.79   84.92
84.92   90.37   92.79   93.59   94.31   94.42   94.68   95.24   95.42
95.42   95.42
```

```
93.49   93.49   93.49   93.49   93.49   93.74   93.45   91.19   86.24
86.24   93.14   94.56   95.69   95.67   96.02   96.07   95.88   95.88
95.88   95.88
94.37   94.37   94.37   94.37   94.37   94.24   93.97   91.80   85.70
85.70   90.78   93.73   96.00   96.13   96.39   96.23   96.23   96.23
96.23   96.23
95.03   95.03   95.03   95.03   95.03   94.26   94.29   91.51   82.22
82.22   89.23   93.00   95.29   96.05   96.05   96.05   96.05   96.05
96.05   96.05
94.75   94.75   94.75   94.75   94.75   94.75   93.06   90.49   81.37
81.37   87.75   92.89   95.47   95.83   95.83   95.83   95.83   95.83
95.83   95.83
94.07   94.07   94.07   94.07   94.07   94.07   93.27   89.98   80.69
80.69   86.69   92.47   95.18   95.40   95.40   95.40   95.40   95.40
95.40   95.40
93.84   93.84   93.84   93.84   93.84   93.84   92.95   89.38   79.83
79.83   86.00   92.05   95.06   95.48   95.48   95.48   95.48   95.48
95.48   95.48
93.05   93.05   93.05   93.05   93.05   93.05   93.05   89.16   78.99
78.99   85.00   91.13   94.50   94.70   94.70   94.70   94.70   94.70
94.70   94.70
92.12   92.12   92.12   92.12   92.12   92.12   92.12   88.90   77.41
77.41   84.26   90.75   94.21   94.21   94.21   94.21   94.21   94.21
94.21   94.21
91.27   91.27   91.27   91.27   91.27   91.27   91.27   88.14   76.08
76.08   82.89   90.31   93.49   93.49   93.49   93.49   93.49   93.49
93.49   93.49
90.47   90.47   90.47   90.47   90.47   90.47   90.47   87.80   75.97
75.97   82.22   89.96   93.17   93.17   93.17   93.17   93.17   93.17
93.17   93.17
];

% CONVERT EFFICIENCY MAP TO INPUT POWER MAP
[temp_T,temp_w]=meshgrid(mc_map_trq,mc_map_spd);
temp_mc_outpwr_map=temp_T.*temp_w;
temp_mc_losspwr_map=(1./mc_eff_map-
1).*temp_mc_outpwr_map.*(temp_T>0)+...
   (mc_eff_map-1).*temp_mc_outpwr_map.*(temp_T<0);

%% to compute losses in entire operating range
%% ASSUME that losses at zero torque are the same as those at the
lowest
%% positive torque, and
%% ASSUME that losses at zero speed are the same as those at the
lowest positive
%% speed
temp_zti=find(mc_map_trq==0);
temp_zsi=find(mc_map_spd==0);
if ~isempty(temp_zti)
   temp_mc_losspwr_map(:,temp_zti)=temp_mc_losspwr_map(:,temp_zti+1);
end
if ~isempty(temp_zsi)
   temp_mc_losspwr_map(temp_zsi,:)=temp_mc_losspwr_map(temp_zsi+1,:);
end


%% compute input power (power req'd at electrical side of
motor/inverter set)
mc_inpwr_map=temp_mc_outpwr_map+temp_mc_losspwr_map;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LIMITS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mc_max_crrnt=400;    % maximum current draw for motor/controller set,
A

mc_min_volts=60;     % minimum voltage for motor/controller set, V

% maximum continuous torque corresponding to speeds in mc_map_spd
mc_max_trq=[274.4   274.4   274.4   274.4   233.8 187.0 155.9 133.6
116.9 103.9 93.5 85.0 77.9 71.9 66.8 62.3 58.4 55.0];% (N*m)

mc_max_gen_trq=-1*[170.0 170.0 170.0 170.0 170.0 170.0 143.1 122.7
107.4 95.4 85.9 78.1 71.6 66.1 61.3 57.3 53.7 50.5];
%EV Plus' regenerative torque is less than drive torque

% maximum overtorque (beyond continuous, intermittent operation only)
% below is quoted (peak intermittent stall)/(peak continuous stall)
mc_overtrq_factor= 49/49;  % (--)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DEFAULT SCALING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (--), used to scale mc_map_spd to simulate a faster or slower
running motor
mc_spd_scale=1.0;

% (--), used to scale mc_map_trq to simulate a higher or lower torque
motor
mc_trq_scale=1.0;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OTHER DATA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mc_inertia=0.0507;    % (kg*m^2), rotor's rotational inertia
mc_mass=(45 + 15);  % (kg), mass of motor and controller

% motor/controller thermal model
% Note: These values are estimates by NREL, based on Westinghouse
75kW.  Thermal model was not available in A2.0.2
%       at the time Honda entered the original data.
mc_th_calc=1;                              % --     0=no mc thermal
calculations, 1=do calc's
mc_cp=430;                                 % J/kgK  ave heat capacity
of motor/controller (estimate: ave of SS & Cu)
mc_tstat=45;                               % C      thermostat temp of
motor/controler when cooling pump comes on
mc_area_scale=(mc_mass/91)^0.7;            % --     if motor
dimensions are unknown, assume rectang shape and scale vs AC75
mc_sarea=0.4*mc_area_scale;                % m^2    total module
surface area exposed to cooling fluid (typ rectang module)

%the following variable is not used directly in modelling and should
always be equal to one
%it's used for initialization purposes
mc_eff_scale=1;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CLEAN UP
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear temp*


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REVISION HISTORY
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 3/11/99 (KW): converted data from A2.0.2 into A2.1 (added estimated
thermal characteristics)
% 3/15/99:ss made sure *_version was updated to  2.1 from 2.0
% 7/29/99:mc removed the T=-0.1 Nm column from the map and removed
code that
%            overrode efficiency in the -ive torque quadrant
% 11/03/99:ss updated version from 2.2 to 2.21
% Begin added by ADVISOR 3.2 converter: 30-Jul-2001
mc_mass_scale_coef=[1 0 1 0];

mc_mass_scale_fun=inline('(x(1)*mc_trq_scale+x(2))*(x(3)*mc_spd_scale
+x(4))*mc_mass','x','mc_spd_scale','mc_trq_scale','mc_mass');

% End added by ADVISOR 3.2 converter: 30-Jul-2001
```

## Appendix VI: M-File for SPHEV Controller

```
function [sys,x0,str,ts] = mode(t,x,u,flag);

% Description of Control Algorithm Program
%

% This Part(A) of program explains the set-up of s-function for
future modifications

%%%%%%%%%%%%%%%%%%%%%
% Start of PART (A) %
%%%%%%%%%%%%%%%%%%%%%
%TIMESTWO S-function whose output is two times its input.
%   This M-file illustrates how to construct an M-file S-function
that
%   computes an output value based upon its input.  The output of
this
%   S-function is two times the input value:
%
%     y = 2 * u;
%
%   See sfuntmpl.m for a general S-function template.
%
%   See also SFUNTMPL.

%   Copyright 1990-2002 The MathWorks, Inc.
%   $Revision: 1.7 $

%
% Dispatch the flag. The switch function controls the calls to
% S-function routines at each simulation stage of the S-function.
%
switch flag,;
  %%%%%%%%%%%%%%%%%%
  % Initialization %
  %%%%%%%%%%%%%%%%%%
  % Initialize the states, sample times, and state ordering strings.
  case 0
    [sys,x0,str,ts]=mdlInitializeSizes;

  %%%%%%%%%%%
  % Outputs %
  %%%%%%%%%%%
  % Return the outputs of the S-function block.
  case 3
    sys=mdlOutputs(t,x,u);

  %%%%%%%%%%%%%%%%%%%
  % Unhandled flags %
  %%%%%%%%%%%%%%%%%%%
  % There are no termination tasks (flag=9) to be handled.
  % Also, there are no continuous or discrete states,
  % so flags 1,2, and 4 are not used, so return an emptyu
  % matrix
  case { 1, 2, 4, 9 }
    sys=[];
```

```
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % Unexpected flags (error handling)%
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % Return an error message for unhandled flag values.
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end

% end timestwo

%
%=====================================================================
=========
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-
function.
%=====================================================================
=========
%
function [sys,x0,str,ts] = mdlInitializeSizes();

sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 19;  % sized outputs
sizes.NumInputs      = 10;  % sized inputs
sizes.DirFeedthrough = 1;   % has direct feedthrough
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
str = [];
x0  = [];
ts  = [-1 0];    % inherited sample time

% end mdlInitializeSizes

%
%=====================================================================
=========
% mdlOutputs
% Return the output vector for the S-function
%=====================================================================
=========

%%%%%%%%%%%%%%%%%
% End of PART(A)%
%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%
% Start of PART(B) %
%%%%%%%%%%%%%%%%%%%%

function sys = mdlOutputs(t,x,u)

%Set-up input%
% u(1) = Pdemand
sys(1) = u(1);
% u(2) = Pdemand_cont
```

```
sys(2) = u(2);
% u(3) = SOC
sys(3) = u(3);
% u(4) = ICE_ON
sys(4) = u(4);
% u(5) = ICE_Trq
sys(5) = u(5);
% u(6) = ICE_w
sys(6) = u(6);
% u(7) = PSD_g
sys(7) = u(7);
% u(8) = Gen_i
sys(8) = u(8);
% u(9) = MC_w
sys(9) = u(9);
% counter-- only for long distance mode
sys(10) = u(10);


% Start of Control Algorithm - Long Distance %
if (sys(3)>0.3) & (sys(10)==0);
    u(11)=0;
    u(12)=0;
    u(13)=0;
    u(14)=4;
    u(15)=1;
    u(16)=0;
        if sys(1)>0
            u(17)=sys(1);
            u(18)=0;
            u(19)=0;
        else
            u(18)=sys(1);
            u(17)=0;
            u(19)=0;
        end

elseif (sys(3)<=0.3);
    u(11)=1;
    u(14)=3;
    u(10)=1; %set counter to 1
    u(16)=0;
        if sys(1)>40000
            u(17)=sys(1)-40000*0.8;
            u(15)=2800/(sys(9)*(1/9)*60/(2*pi)); %gratio =
2800rpm/wheel speed rpm
            u(12)=(0.8*40000)/(2800*2*pi/60);
            u(18)=0;
            u(19)=0;
            u(13)=2800*2*pi/60;
        elseif (0 <= sys(1) <= 40000);
            u(15)=2800/(sys(9)*(1/9)*60/(2*pi)); %gratio =
2800rpm/wheel speed rpm
            u(12)=sys(1)/(2800*2*pi/60);
            u(18)=0;
            u(19)=0;
            u(13)=2800*2*pi/60;
            u(17)=0;
        elseif (sys(1)<0);
            u(18)=u(1);
```

```
            u(17)=0;
            u(19)=0;
            u(12)=50;%idling trq
            u(13)=1000*2*pi/60; %idling mode
            u(15)=2800/(sys(9)*(1/9)*60/(2*pi)); %gratio =
2800rpm/wheel speed rpm
        end

 end

%Return values to Simulink SPHEV model
sys = u;

% History of Program Files
%----------------------------------------------------------------------
------

% First created on 1/5/2006, author MSM
```