Chapter 6
Simulation of Control Circuit for Active Power Filters

The instantaneous space vectors, e and i are set on a, b and c axis. These space vectors are easily transformed into alpha-beta-zero coordinates using relation 1 and 2

$$\begin{bmatrix} e\alpha \\ e\beta \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & \frac{-1}{2} & \frac{-1}{2} \\ 0 & \frac{\sqrt{3}}{2} & \frac{-\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} ea \\ eb \\ ec \end{bmatrix} \quad \text{--- 6.1}$$

$$\begin{bmatrix} i\alpha \\ i\beta \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & \frac{-1}{2} & \frac{-1}{2} \\ 0 & \frac{\sqrt{3}}{2} & \frac{-\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} ia \\ ib \\ ic \end{bmatrix} \quad \text{--- 6.2}$$

Where the α and β axes are the orthogonal coordinates. Necessarily, $e\alpha$ and $i\alpha$ are on the α axis, $e\beta$ and $i\beta$ are on the β axis. Their amplitude and (+,-) direction vary with passage of the time. Relation 3 defines the conventional instantaneous power and reactive power on the three-phase.

$$\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} e\alpha & e\beta \\ -e\beta & e\alpha \end{bmatrix} \begin{bmatrix} i\alpha \\ i\beta \end{bmatrix} \quad \text{--- 6.3}$$

The instantaneous currents on the α-β coordinates, i_α and i_β are divided into two kinds of instantaneous current components, respectively:

$$\begin{bmatrix} i\alpha \\ i\beta \end{bmatrix} = \begin{bmatrix} e\alpha & e\beta \\ -e\beta & e\alpha \end{bmatrix}^{-1} \begin{bmatrix} p \\ 0 \end{bmatrix} + \begin{bmatrix} e\alpha & e\beta \\ -e\beta & e\alpha \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ q \end{bmatrix} \quad \text{--- 6.4}$$

$$= \begin{bmatrix} i\alpha p \\ i\beta p \end{bmatrix} + \begin{bmatrix} i\alpha q \\ i\beta q \end{bmatrix} \quad \text{--- 6.5}$$

The sum of the instantaneous power $p_{\alpha p}$ and $p_{\beta p}$ coincides with the instantaneous real power in the three-phase circuit. $p_{\alpha p}$ and $p_{\beta p}$ represent instantaneous active power. The instantaneous powers $p_{\alpha q}$ & $p_{\beta q}$ cancels each other and make no contribution to the instantaneous power flow from source to the load, therefore $p_{\alpha q}$ and $p_{\beta q}$ represent instantaneous reactive power. Fig 6.1 depicts APF controller using p-q theory.

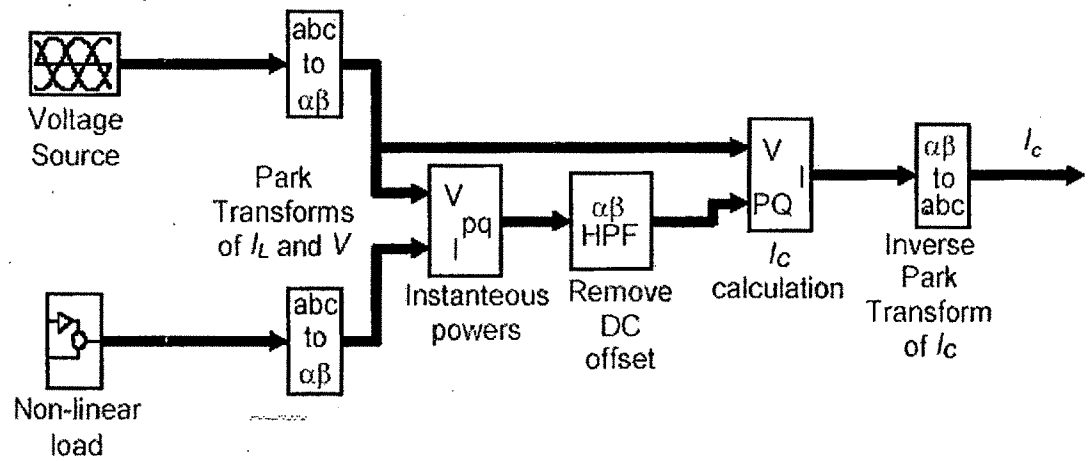


Fig 6.1 p-q theory based APF Controller

6.1. Simulation Tools

The chapter describes tools used for development of a model for control circuit and carry out simulation study

6.1.1 MATLAB

MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C. The name MATLAB is an abbreviation of matrix laboratory.

MATLAB features a family of add-on application-specific solutions called **toolboxes**. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others. The MATLAB system consists of five main pans. When MATLAB is started the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications associated with MATLAB. The first time MATLAB starts, the desktop of Fig.6.2 appears.

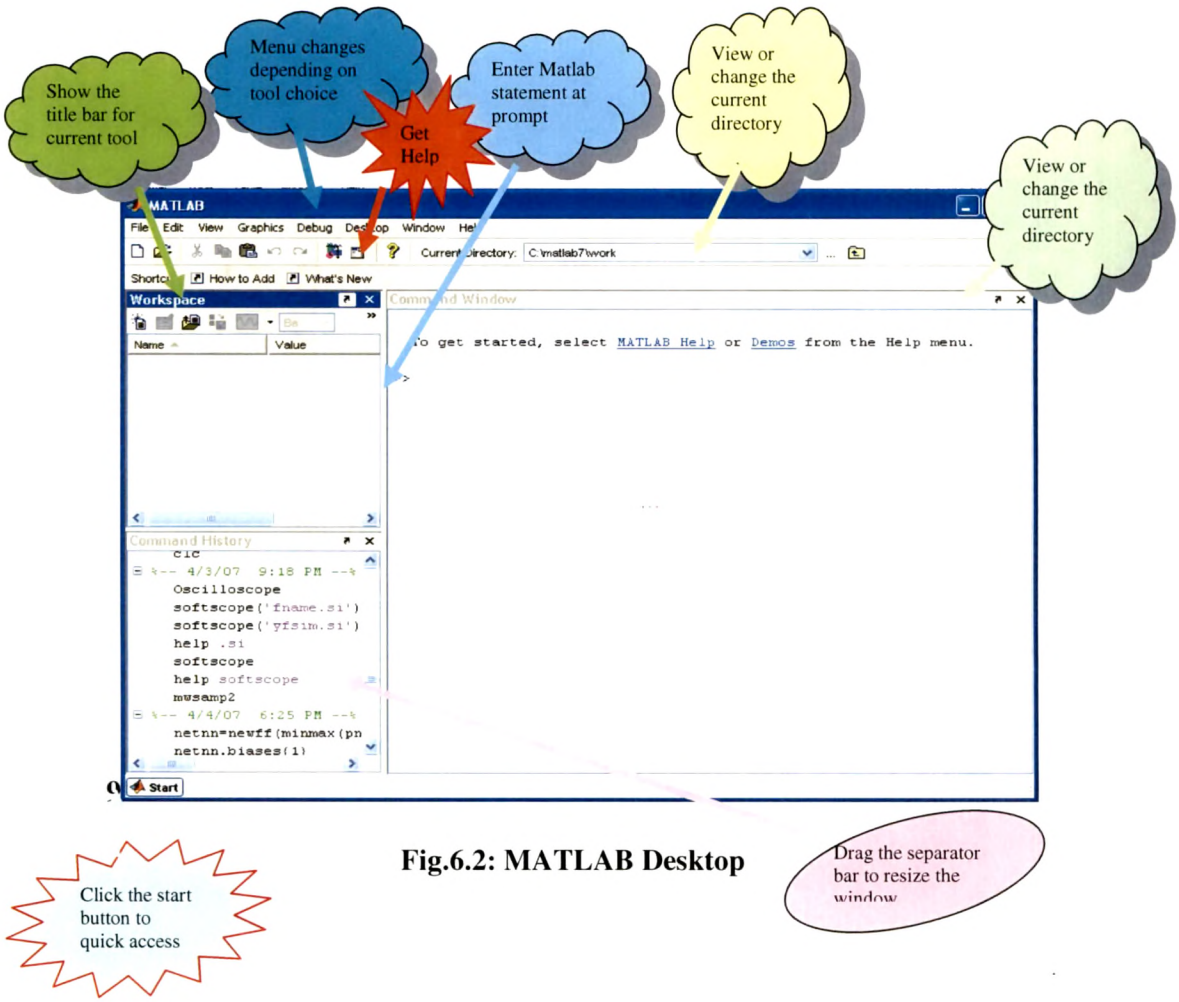


Fig.6.2: MATLAB Desktop

6.1.2 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 a hybrid of the two. Systems can also be multirate, i.e., have different parts that are sampled or updated at different rates.

For modeling, Simulink provides a graphical user interface (GUI) for building models as block diagrams, using click-and-drag mouse operations It includes a comprehensive block library of sinks, sources, linear and nonlinear components, and connectors. You can also customize and create your own blocks. For information on creating your own blocks, like separate Writing S-Functions.

Models are hierarchical, so you can build models using both top-down and bottom-up approaches. You can view the system at a high level, and then double-click blocks to go down through the levels to see increasing levels of model detail. This approach provides insight into how a model is organized and how its parts interact.

The model can be simulated using a choice of integration methods, either from the Simulink menus or by entering commands in the MATLAB Command Window. The menus are particularly convenient for interactive work, while the command-line approach is very useful for running a batch of simulations (for example Using scopes and other display blocks, you can see the simulation results while the simulation is running. The simulation results can be put in the MATLAB workspace for post processing and visualization.

Simulink provides a **library browser** (fig. 6.3) that allows you to select blocks from libraries of standard blocks and a graphical editor that allows you to draw lines connecting the blocks. You can model virtually any real-world dynamic system by selecting and interconnecting the appropriate Simulink blocks.

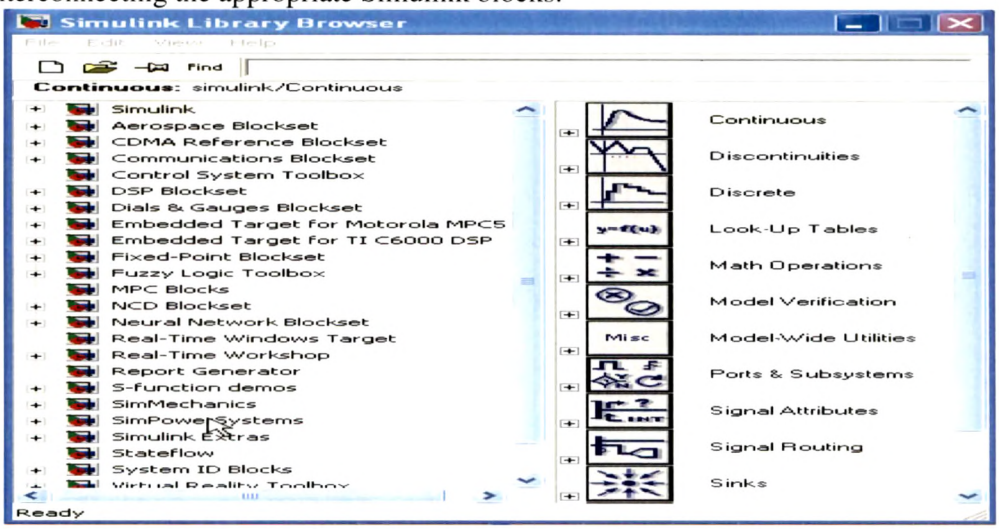


FIG. 6.3 SIMULINK: LIBRARY BROWSER WINDOW

The tree pane displays all the block libraries installed on the system. The icon pane displays the icons of the blocks that reside in the library currently selected in the tree pane. The documentation pane displays documentation for the block selected in the icon pane. You can locate blocks either by navigating the Library Browser's library tree or by using the Library Browser's search facility.

The library tree displays a list of all the block libraries installed on the system. You can view or hide the contents of libraries by expanding or collapsing the tree using the mouse or keyboard. To expand/collapse the tree, click the +/- buttons next to library entries or select an entry and press the +/- or right/left arrow key on your keyboard. Use the up/down arrow keys to move up or down the tree.

- To open the library right-click the library's entry in the browser. Simulink displays an **Open Library** button. Select the **Open Library** button to open the library. To create a model, select the **New** button on the Library Browser's toolbar. To open an existing model, select the Open button on the toolbar.
- To copy a block from the Library Browser into a model, select the block in the browser, drag the selected block into the model window, and drop it where you

want to create the copy. Blocks are the elements from which Simulink models are built.

- Simulink displays information about a block in a pop-up window when you allow the pointer to hover over the block in the diagram view.

Simulink blocks fall into two basic categories: nonvirtual and virtual blocks. Nonvirtual blocks play an active role in the simulation of a system. Virtual blocks, by contrast, play no active role in the simulation; they help organize a model graphically. Some Simulink blocks are virtual in some circumstances and nonvirtual in others. Such blocks are called conditionally virtual blocks.

Signals are the streams of values that appear at the outputs of Simulink blocks when a model is simulated. It is useful to think of signals as traveling along the lines that connect the blocks in a model diagram. The output of a Simulink block, by contrast, appears instantaneously at the input of the block to which it is connected. Simulink blocks can output one- or two-dimensional signals. A one-dimensional (1-D) signal consists of a stream of one-dimensional arrays output at a frequency of one array (vector) per simulation time step. A two-dimensional (2-D) signal consists of a stream of two-dimensional arrays emitted at a frequency of one 2-D array (matrix) per block sample time. The Simulink user interface and documentation generally refer to 1-D signals as *vectors* and 2-D signals as *matrices*. A one-element array is frequently referred to as a *scalar*. A *row vector* is a 2-D array that has one row. A *column vector* is a 2-D array that has one column.

Simulink blocks vary in the dimensionality of the signals they can accept or output during simulation. The values of Simulink signals can be complex numbers. A signal whose values are complex numbers is called a complex signal. A virtual signal is a signal that represents another signal graphically. Virtual blocks, such as a Bus Creator or Subsystem block, generate virtual signals. Like virtual blocks, virtual signals allow you to simplify your model graphically. For example, using a Bus Creator block, you can reduce a large number of nonvirtual signals (i.e., signals originating from nonvirtual blocks) to a single virtual signal, thereby making your model easier to understand. You can think of a virtual signal as a tie wrap that bundles together a number of signals. A control signal is a signal used by one block to initiate execution of another block, e.g., a function-call or action subsystem. When you update or start simulation of a block diagram, Simulink uses a dash-dot pattern to redraw Signal Basics lines representing the diagram's control signals.

The term **data type** refers to the way in which a computer represents numbers in memory. A data type determines the amount of storage allocated to a number, the method used to encode the number's value as a pattern of binary digits, and the operations available for manipulating the type. Most computers provide a choice of data types for representing numbers, each with specific advantages in the areas of precision, dynamic range, performance, and memory usage. To enable you to take advantage of data typing to optimize the performance of MATLAB programs, MATLAB allows you to specify the data types of MATLAB variables. Simulink builds on this capability by allowing you to specify the data types of Simulink signals and block parameters. Simulink supports all built-in MATLAB data types except int64 and uint64. The term *built-in data type* refers to data types defined by MATLAB itself as opposed to data types defined by MATLAB users. Besides the built-in types, Simulink defines a Boolean (1 or 0) type, instances of which are represented internally by uint8 values. Simulink also supports fixed-point data types.

When the complexity of model increases, the development of model can be simplified by grouping blocks into **subsystems** which can be created in two ways:

1. Add a Subsystem block to the model, then open that block and add the blocks it contains to the subsystem window.
2. Add the blocks that make up the subsystem, then group those blocks into a subsystem.

Table 6.1 lists the SIMULINK toolboxes used in this project work.

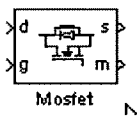
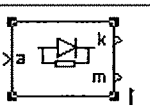
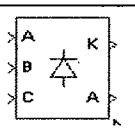
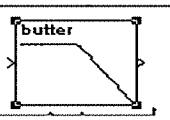
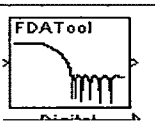
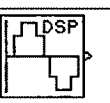
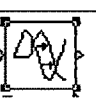
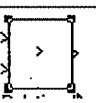
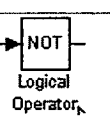
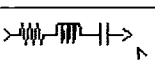
TOOL BOX	FUNCTION	SYMBOL
SIMPOWERSYSTEMS / POWER ELECTRONICS	MOSFET	
SIMPOWERSYSTEMS / POWER ELECTRONICS	DIODE	
SIMPOWERSYSTEMS / THREE PHASE LIBRARY	6-PULSE DIODE BRIDGE	
SIMPOWERSYSTEMS / ELECTRICAL SOURCES	A.C. VOLTAGE SOURCE	
DSP BLOCKSET / FILTERING	ANALOG FILTER DESIGN	
DSP BLOCKSET / FILTERING	DIGITAL FILTER DESIGN	
DSP BLOCKSET / FILTERING	SINE WAVE	
SIMULINK / CONTINUOUS	TRANSPORT DELAY	
SIMULINK / MATH OPERATIONS	RELATIONAL OPERATOR	
SIMULINK / MATH OPERATIONS	LOGICAL OPERATOR	
SIMPOWERSYSTEMS / ELEMENTS	SERIES RLC BRANCH	

TABLE 6.1 TOOLBOXES and Functional Blocks USED in the project

6.1.3 Running a SIMULINK model

It is generally a two-step process:

- Specification of various simulation parameters such as the solver used to solve the model, the start and stop time for the simulation, the maximum step size.

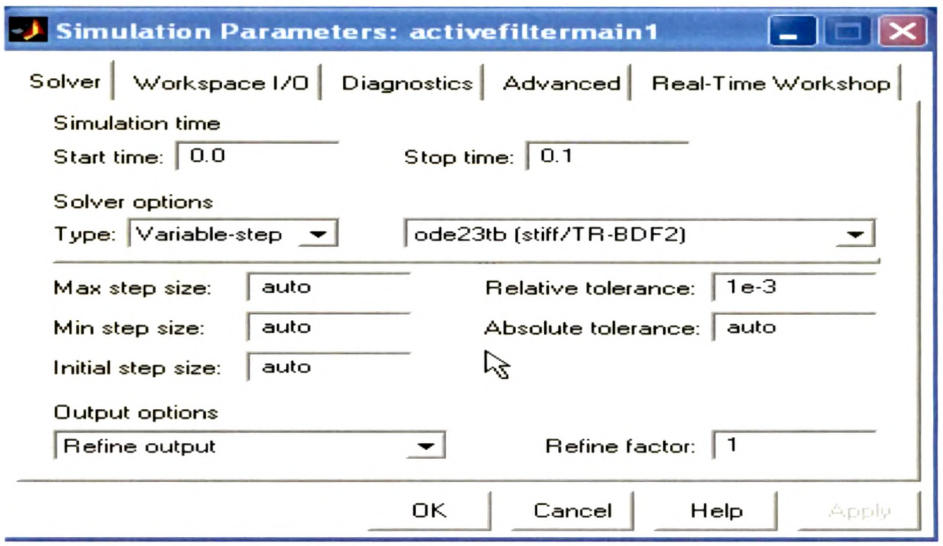


Fig 6.4 Dialog Box: SIMULATION PARAMETERS

To use the **Simulation Parameters** dialog box:

- Open or select the model whose simulation parameters you want to set.
- Select **Simulation parameters** from the model window's **Simulation** menu.
- Change the settings as necessary to suit your needs. You can specify parameters as valid MATLAB expressions, consisting of constants, workspace variable names, MATLAB functions, and mathematical operators. The **Simulation Parameters** dialog box appears. (Fig. 6.4)
- Click **Apply** to confirm the changes or **OK** to confirm the changes and dismiss the dialog box.

The **Solver** pane allows to: Set the simulation start and stop time, Choose the solver and specify its parameters and Select output options he simulation output can be directed to workspace variables and get input and initial states from the workspace. On the **Simulation Parameters** dialog box, select the **Workspace I/O** tab, fig. 6.5 pane appears.

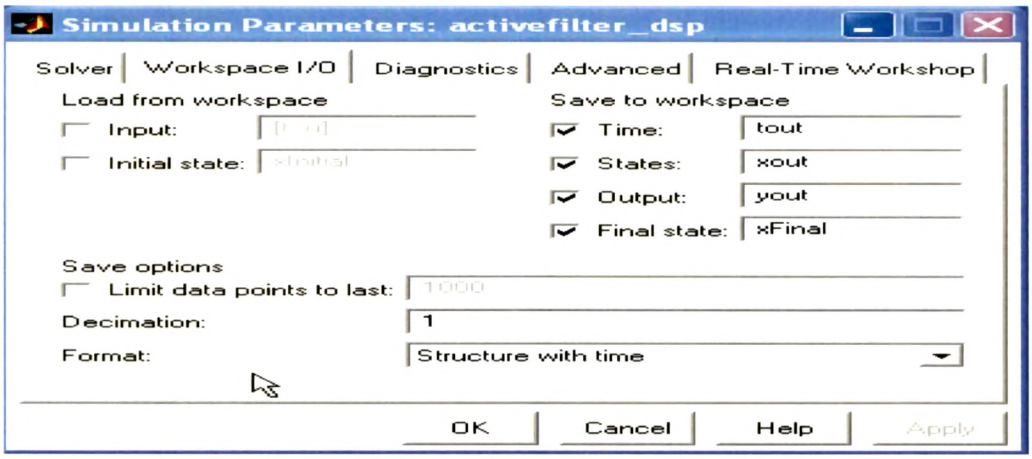


Fig 6.5 WORK SPACE I/O PANE

The **Simulink debugger** is a tool for locating and diagnosing bugs in a Simulink model. It enables you to pinpoint problems by running simulations step by step and displaying intermediate block states and input and outputs. The Simulink debugger has both a graphical and a command-line user interface. The graphical interface allows you to access the debugger's most commonly used features. The command-line interface gives you access to all the debugger's capabilities. Wherever you can use either interface to perform a task, the documentation shows you first how to use the graphical interface and then the command-line interface to perform the task.

The **Model Discretizer** selectively replaces continuous Simulink blocks with discrete equivalents. Discretization is a critical step in digital controller design and for hardware in-the-loop simulations. You can use this tool to prepare continuous models for use with the Real-Time Workshop Embedded Coder, which supports only discrete blocks.

6.2 Controller Simulation

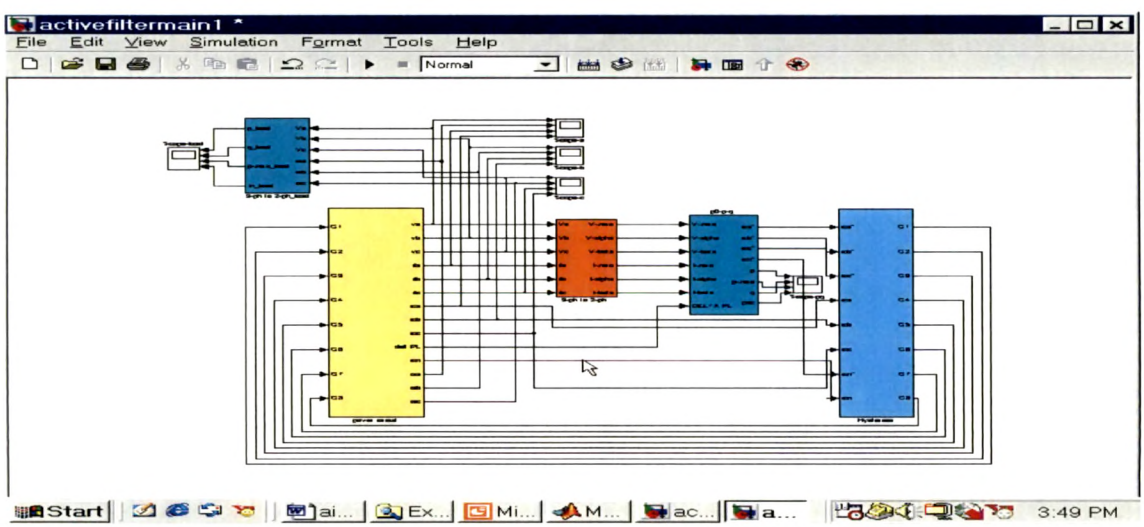


Fig 6.6 Simulation Setup for Control Signal Generation

Fig 6.6depicts a setup for generation of firing pulses using p-q theory is implemented using MATLAB/SIMULINK. The four blocks used correspond to the four stages of p-q theory correspond to the respective subsystem..

1. **Power Circuit (Fig 6.7)**

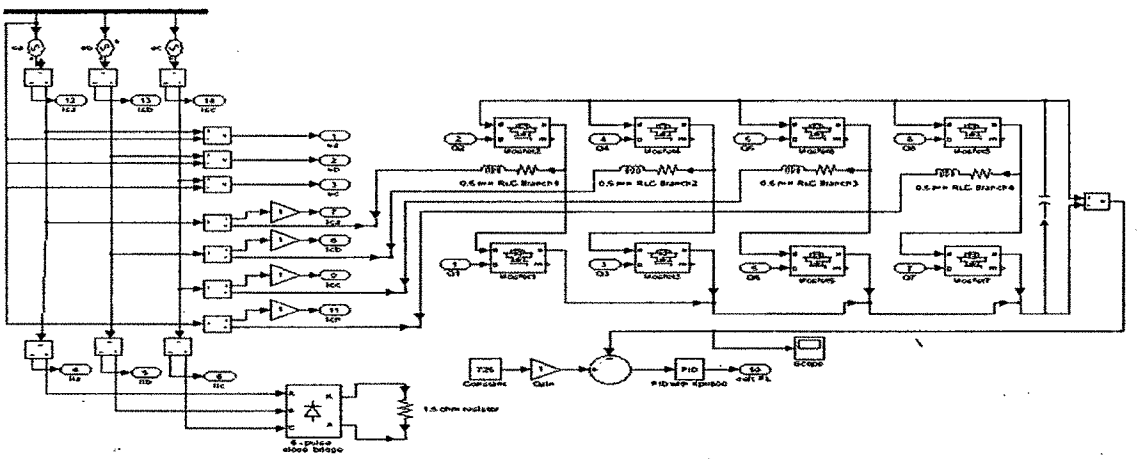


Fig 6.7 SIMULINK subsystem: Power Circuit

2 Three phase to two phase source converter (Fig 6.8) is realized using the following relations (6.6 to 6.8)

$$i_o = 0.577i_a + 0.577i_b + 0.577i_c \quad \text{---6. 6}$$

$$i_\alpha = 0.816i_a - 0.41i_b - 0.41i_c \quad \text{- - -6.7}$$

$$i_\beta = 0.707i_b - 0.707i_c \quad \text{--- 6.8}$$

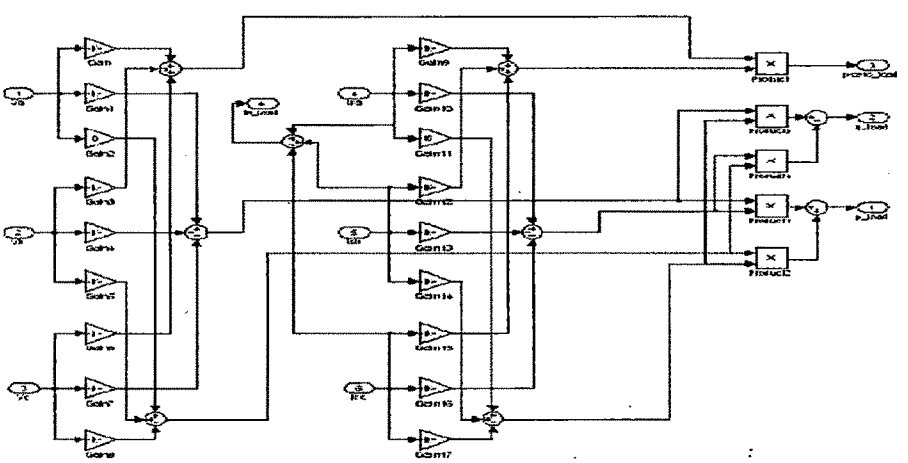


Fig 6.8 SIMULINK Subsystem: Three Phase – Two Phase Conversion

3 Computation of p, q component of power and compensating currents generator

(Fig 6.9) using the following relations (6.9 ....6.13).

$p_o = v_o i_o$ ---6. 9

$p = v_\alpha i_\alpha + v_\beta i_\beta$ --- 6.10

$q = v_\alpha i_\beta - v_\beta i_\alpha$ ---6.11

$$ic\alpha = \frac{1}{e^2\alpha + e^2\beta} * (-e\alpha * pac + e\beta * qac) \quad \text{--- 6.12}$$

$$ic\beta = \frac{1}{e^2\alpha + e^2\beta} * (-e\beta * pac - e\alpha * qac) \quad \text{---- 6.13}$$

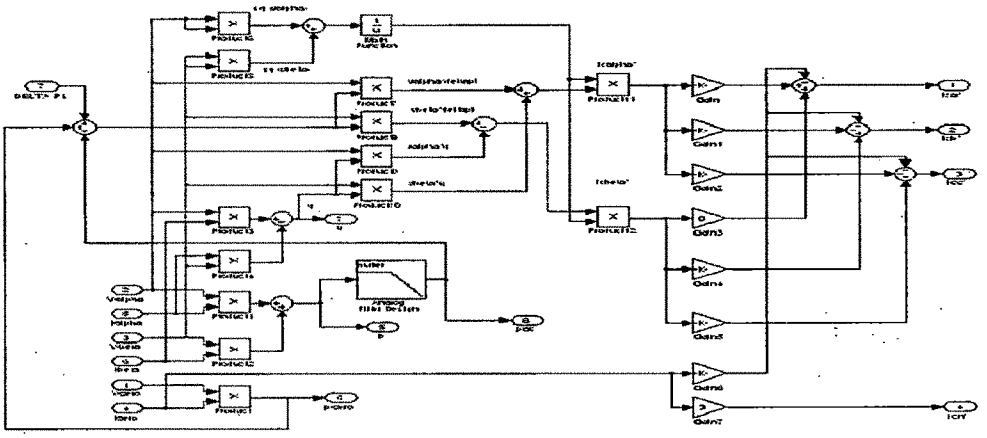


Fig 6.9 SIMULINK Sub System: p-q calculation and Compensating Current Generator

4. The i_{ca}^*, i_{cb}^*, i_{cc}, i_{cn}^* given by relations (6.14 ... 6.17) represents compensated three phase four wire reference currents.

$i_{ca}^* = -0.577\, i_o + 0.816\, i_{c\alpha}$ --- 6.14

$i_{cb}^* = -0.577\, i_o - 0.408\, i_{c\alpha} + 0.707\, i_{c\beta}$ --- 6.15

$icc^* = -0.577 i_o - 0.408\, i_{c\alpha} - 0.707\, i_{c\beta}$ --- 6.16

$icn^* = 1.73 i_o$ --- 6.17

The Hystersis band controller block (Fig 6.10) generates desired firing pulses.

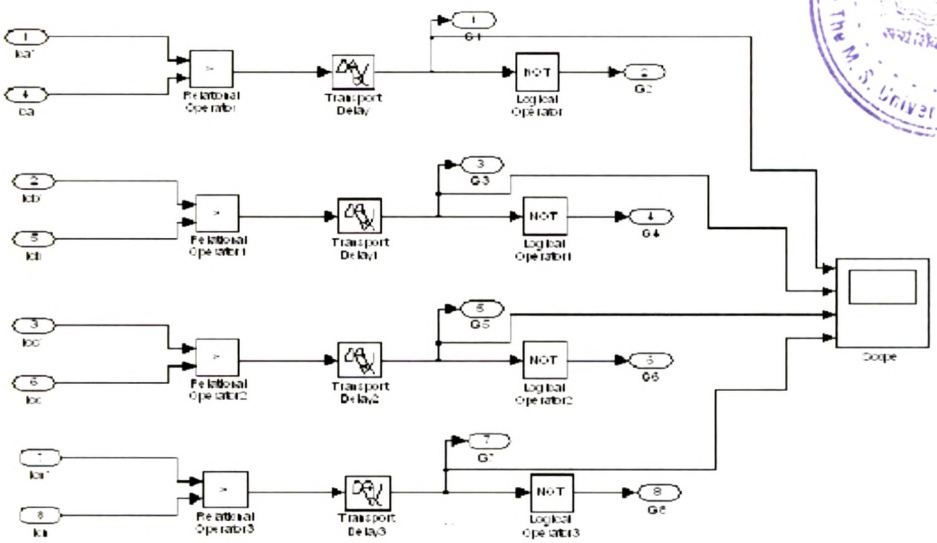
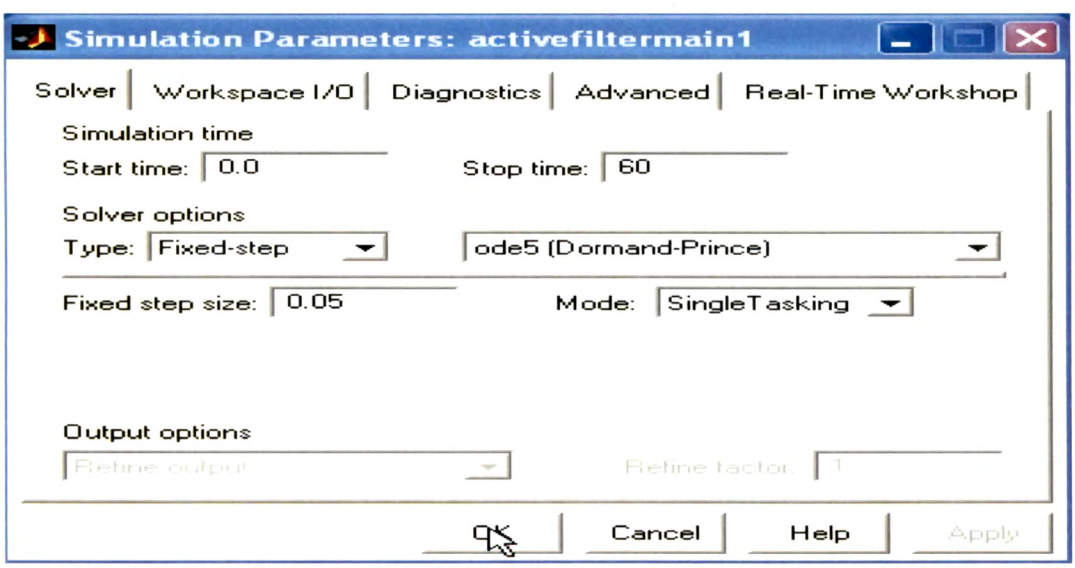


Fig 6.10 SIMULINK Sub System: Hysterisis Band controller

6.3 SIMULATION WAVEFORMS

The simulation parameters for the model of Fig 6.6 are set as shown in Fig.6.11 The response; simulation waveforms for source voltage, source current, load current and compensating current for a, b and c phases on running the SIMULINK model.



9

Fig 6.11 SELECTION of SIMULATION PARAMETER for APF controller

Fig 6.12 to 6.14 depicts the simulation waveforms for source voltage, source current, load current and compensating current for **a, b and c phases** respectively. Computational and load side waveforms are in Fig 6.15 and Fig 6.16 respectively.

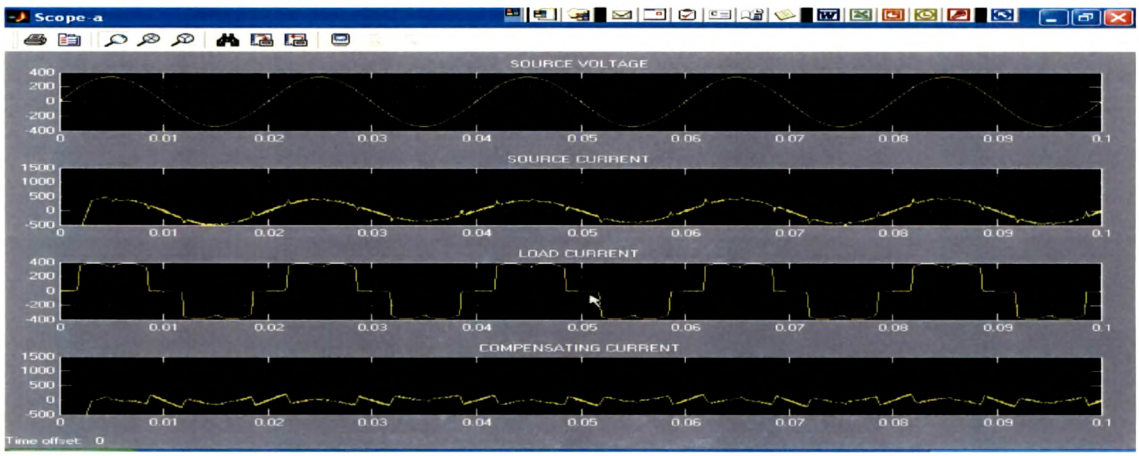


Fig 6.12: SIMULATION Responses: Phase A

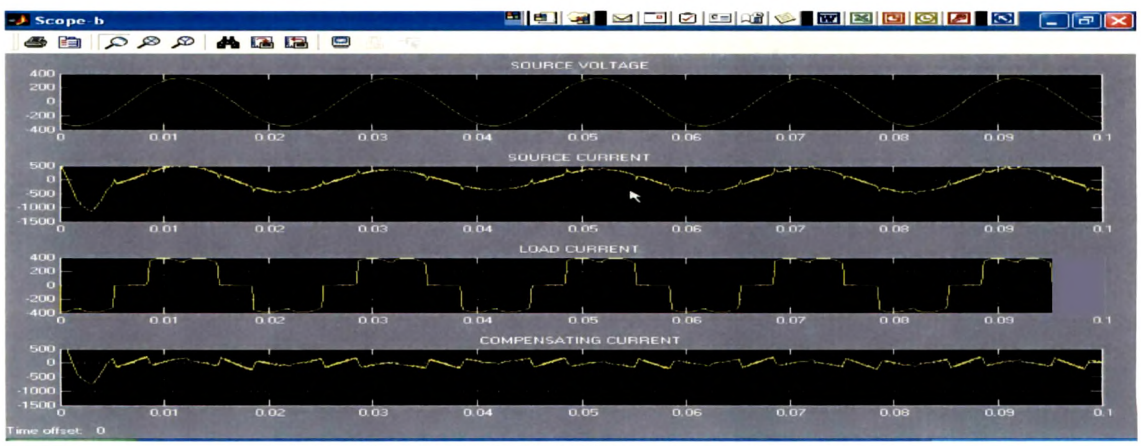


Fig 6.13 SIMULATION Responses: Phase B

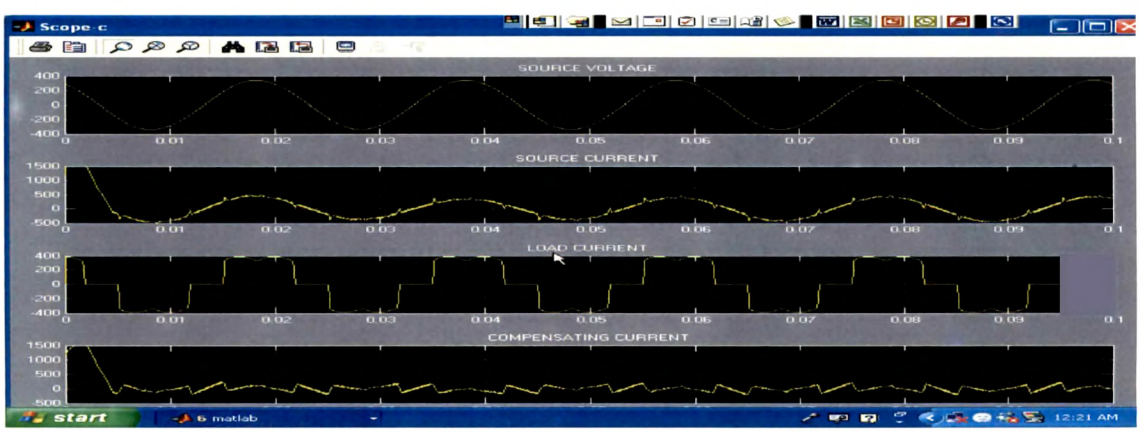


Fig 6.14 SIMULATION Responses: Phase C

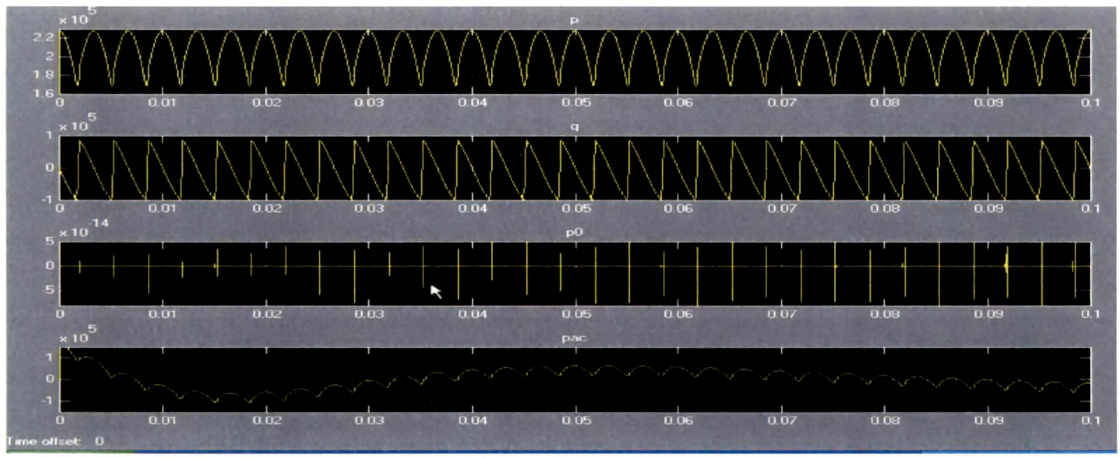


Fig. 6.15 Simulation Responses: Power Computations

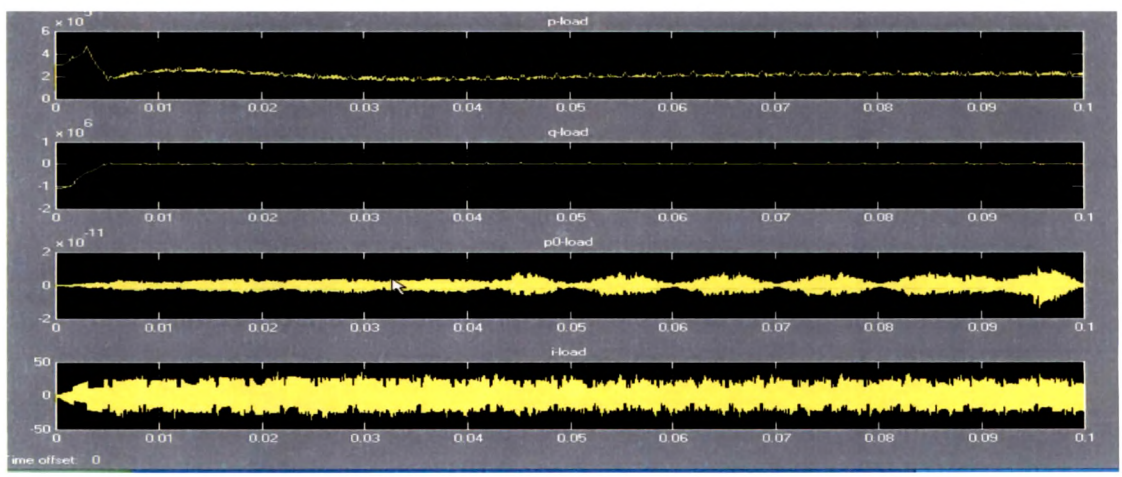


Fig 6.16: Simulation Responses: Load Side Waveforms

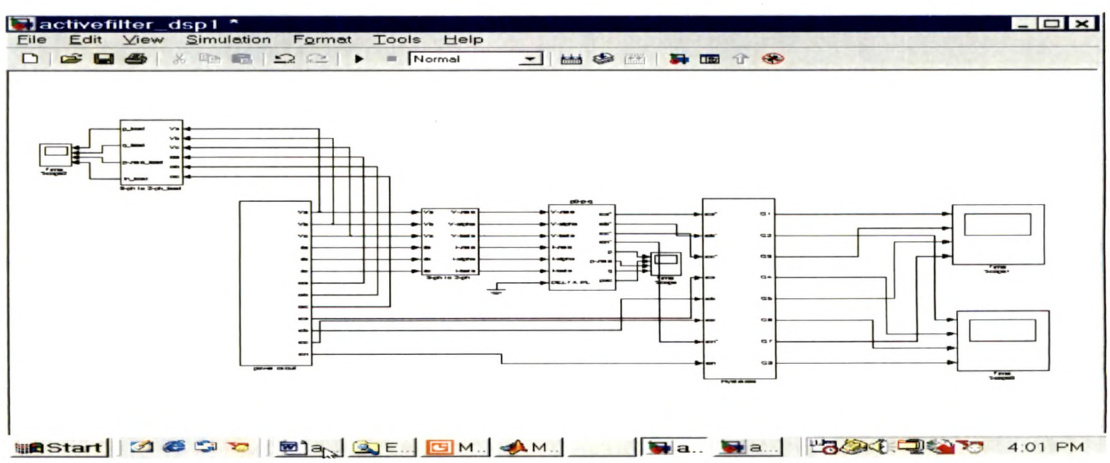


Fig 6.17 Simulation setup with and without loading

To check the simulated control circuit, the IGBT driver circuit is also simulated using MATLAB / SIMULINK. The firing pulses generated from the above circuit are applied given to the IGBT driver circuit to obtain desired output. The set up of Fig 6.17 was used for simulation of signals for driver circuit of Power circuit. Fig 6.18 and 6.19 depicts the firing pulses.

Fig 6.18 Simulated Firing Pulses: without Load

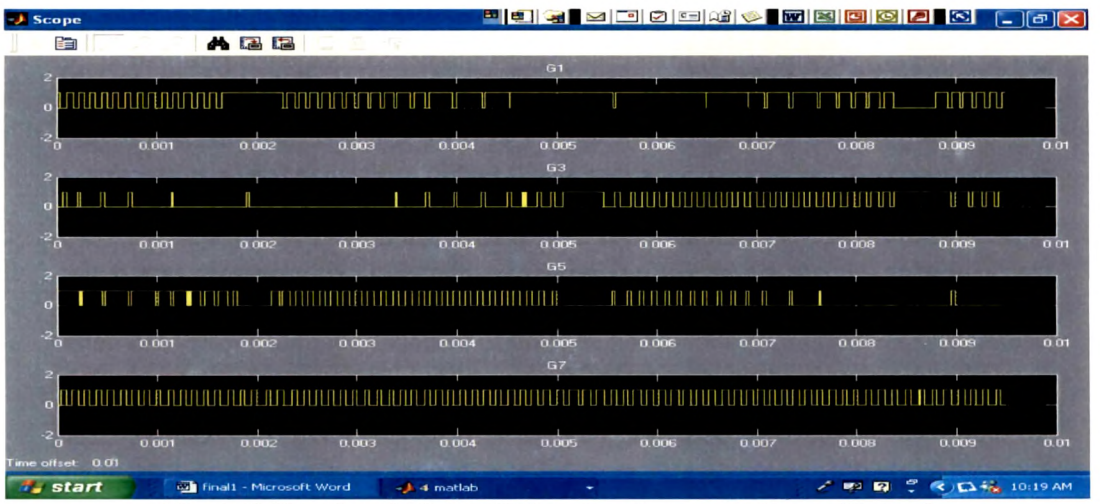


Fig 6.19 Simulated Firing Pulses for Compensation with load

The SIMULINK model developed here will be used to develop software for DSP implementation Using Real Time workshop for embedded Targets by MATHWORKS and CCS link for TI DSP 67xx series of processors. The software is described in the subsequent chapters.