DEVELOPMENT OF A SIMULATION LANGUAGE

IN CONJUNCTION WITH HYDRO-TURBINE MODELLING

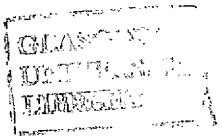
A Thesis
submitted to the Faculty of Engineering
of the University of Glasgow
for the Degree of

Doctor of Philosophy

by

Kenneth Hugh Aitken B.Sc.

January 1982



SUMMARY

Two main areas of work are covered in this thesis, namely the development of a digital simulation language and the modelling of a hydro-turbine generator.

The simulation language (GUILDS) is based upon a FORTRAN simulation package. The use of the STAGE2 macroprocessor to translate the language statements into FORTRAN is described along with the modifications and additions to the FORTRAN package. A User's Guide for GUILDS, prepared in the course of the project, has been included for reference.

The theoretical and empirical relationships which combine to form the simulation model of a hydro-turbine generating set are presented. The validity of these relationships is discussed in conjunction with an extensive comparison of site test results with those from the simulation model. Site tests were carried out at Loch Sloy Power Station during the commissioning of an operational microprocessor governor for one of the hydro-turbines.

The importance of a simulation model as a design tool for governor development is illustrated by a simulation study carried out on a proposed non-linear governor for the hydro-generator.

As an additional area of work, of a model of a thermal generating station was implemented, using GUILDS. The validity of the model is established and a study of the effect on the grid system of starting pumps in a pumped storage hydro-station is described.

CONTENTS

GUILDS USER'S GUIDE

The User's Guide for the Simulation Language GUILDS has been included in a pocket at the rear of the Thesis.

ACKNOWLEDGEMENTS

The work described in this thesis was funded jointly by the Science Research Council and the North of Scotland Hydro Electric Board and the author would like to thank them for their financial support. In addition the author is indebted to many others for their help and co-operation throughout the duration of the project.

Professor J. Lamb, of the Department of Electronics and Electrical Engineering at Glasgow University, is thanked for the provision of equipment and laboratory facilities.

The guidance of Dr. D.J. Winning, as supervisor, has been greatly appreciated since the start of the project, as has his general advice, consideration and encouragement. Dr. H. Davie, Dr. T.R. Foord and fellow research students Dr. N.F. Grant and Mr. A.P.B. Halley are thanked for their valuable advice and assistance, and the support of the computing and technical staff of the Department is gratefully acknowledged. Also, thanks go to Dr. D.G.E. Findlay, a former research student, currently with YARD Ltd., for his continued interest and assistance, and for the use of his plotting package.

The author is indebted to the NSHEB, not only for the funding provided, but also for the facilities made available in Sloy Power Station. Mr. A. G. Marshall, of the NSHEB, is thanked for his advice and support and particularly for his valuable contribution to the work on site. The co-operation of the personnel at Sloy Power Station has been appreciated and is gratefully acknowledged.

The author would like to thank the Directors and staff of YARD Ltd. (his present employer) for facilities made available and help received during the preparation of this Thesis.

Finally, the author would like to thank his family and friends, particularly his wife, for their continued support and encouragement during the project.

LIST OF SYMBOLS

a	Wave velocity
A	Cross-sectional area
A_{cv}	Area of water control valve
A_e	Effective area of water control valve ($A_{cv}C_d$)
A_{rv}	Area of relief valve
A_1	Governor valve actuator position
A_2	Governor valve position
b_p	Temporary Droop Double Derivative governor parameter
b_t	Temporary Droop governor parameter
B	Bulk modulus of fluid
B_1	Interceptor valve actuator position
B_2	Interceptor valve position
C_d	Coefficient of discharge of water control valve
C_{drv}	Coefficient of discharge of relief valve
D	Internal diameter of conduit
e	Wall thickness of conduit
e_n	Load self regulation factor
E	Modulus of elasticity of conduit material
E_{m1}	Change in fuel demand
E_m	E_{m1} delayed by T_d seconds
f	System frequency or stress factor of pipe
f_d	Frequency signal to derivative terms in governor
f_e	Frequency error signal (f_s-f)
f_1	Isolated load frequency
f_s	Frequency set point
f_t	Frequency transducer output

F	Coefficient of pipe section (as specified by subscript - see below)
F_a	Accelerating torque
F_d	Fuel density
F_e, F_{e1}	Electrical torque (generated)
F_g	Gross torque developed by runner
F_i	Fuel input to boiler
F_L	Electrical torque (load)
F_l	Turbine loss torque
F_m	Turbine mechanical torque
F_w	Water torque
g	Gravitational acceleration
G	Servo open loop gain
H	Head (as specified by subscript - see below)
H_R	Reaction head
I_a	Alternator moment of inertia
k_n	Isolated load simulator parameter (= e_n - 1)
K	Coefficient of pipe section (as specified by subscript - see below)
K_p, K_i, K_d	Proportional, integral and derivative coefficients for master pressure controller
K_1, K_2	Double Derivative governor parameters
K_{12}, K_{22}	Switched values of K_1 and K_2 respectively
L	Length of conduit (as specified by subscript - see below)

M	Master pressure control signal (M_x between limits)
M_i	Steady state fuel flow
M_x	Output of master pressure controller
n	Efficiency
P	Power
P_b	Boiler pressure
P_e	Electrical power (Generated) or pressure error signal ($P_s - P_b$)
P_L	Electrical power (Load)
P_{Lo}	Steady state electrical load power
P_m	Turbine mechanical power
P_s	Pressure set point
P_w	Water power
P_1	High pressure cylinder pressure
P_2	Reheat cylinder pressure
P_{m1}	Power output of high pressure cylinder
P_{m2}	Power output of reheat cylinder
P_t	Total turbine power output ($P_{m1} + P_{m2}$)
P_{t1}	Power output of regulating thermal plant
$q_1 - q_7$	Frequency transducer/governor parameters
Q	Flow (as specified by subscript - see below)
Q_i	Heat input to boiler
r_c	Servo closing rate limit
r_m	Relief valve maximum reclosure rate
r_o	Servo opening rate limit
r_t	Relief valve threshold opening rate
r_u	Relief valve unforced reclosure rate
r_1, r_2	Turbine runner outer and inner radii

R	Universal gas constant
s	Laplace operator
s_d	Derivative coefficient select
T	Temperature (absolute)
T_a	Alternator time constant
T_b	Boiler time constant
T_c	Fuel feed time constant
T_d	Temporary Droop governor parameter or fuel feed delay time
T_g	Governor valve time constant
T_L	Double Derivative governor parameter
T_m	Mill storage time constant
T_r	Relief valve time constant or reheater time constant
T_s	Servo time constant or mill start delay time
T_w	Water time constant
T_y	Temporary Droop Double Derivative governor parameter
T_3, T_4	Double Derivative governor parameters
u_1, u_2	Velocity of runner at outer and inner radii
v	Volume (of storage element)
w	Steam flow
w_1, w_2	Whirl velocity of fluid at outer and inner radii of runner
W_1	High pressure cylinder steam flow
W_2	Reheat cylinder steam flow
$x_1 - x_5$	Intermediate variables in simulation model

y	Servo position
y_c	Water control valve position
y_d	Desired servo position ($y_g + y_s$)
y_e	Servo error signal ($y_d - y$)
y_f	Frequency Disturbance System output
y_g	Governor output
y_l	Relief valve linkage position
y_L	Load limit set point
y_r	Relief valve position
y_s	Servo set point (desired MW)
z	Rate of change of servo position (dy/dt)
z_f	Derivative of frequency
z_g	Derivative of governor output
z_l	Rate of change of relief valve servo position
z_1, z_2	First and second derivatives in Double Derivative governor
α	Dimensionless constant in turbine transfer function
γ	Adiabatic index for steam
Δh	Heat drop (across a storage element)
ΔA	Per-unit deviation in area A
ΔH	Head drop (across an orifice)
ΔP	Per-unit deviation in power P
ΔP_L	Per-unit deviation in load power P_L
$\Delta \omega$	Per-unit deviation in speed
η_g	Generator efficiency
η_t	Turbine efficiency
ρ	Density of fluid
ω	Turbine speed

Pipeline Subscripts

1, 2, 3	Position in general pipe section representation
a	Conduit from surge shaft to first bifurcation
b	Conduit from bifurcation to sets 1 and 2
c	Conduit from bifurcation to set 3
d	Conduit from reservoir to surge shaft
r	Reservoir
rv	Relief valve
s	Surge shaft
t	Turbine

CHAPTER 1

GENERAL INTRODUCTION

For some years the Department of Electronics and Electrical Engineering at Glasgow University, in collaboration with the North of Scotland Hydro Electric Board (NSHEB) has been carrying out research into the design of improved speed governors for hydro-generators. In particular, a hydro-generator at Loch Sloy Power Station has been used for experimental studies and the implementation of new governor types.

References 1 and 2 describe the initial stages of this work during which some experimental equipment was installed on site and an electronic governor was developed and tested. References 3 and 4 describe the further development of the experimental equipment on site and the implementation of a microprocessor based governor including improvements to the governor algorithm.

The authors of References 1 and 3 used simulation facilities, initially analogue and hybrid[5] and latterly, digital, to assist in the development and testing of governors before testing on site.

The aims of the present project were the further development of the digital simulation facilities[17], resulting in a Simulation Language, and the expansion of the hydro turbine model to permit more detailed studies to be carried out. This latter section of work also involved modelling a thermal power station with a view to studying the effects of operating large hydro generators in a combined hydro-thermal power system.

The experimental work on site continued throughout this project, partly to permit more detailed information to be obtained from site tests, but also to enable the installation of an operational microprocessor governor to replace the existing station governor. Most of this work was carried out in collaboration with a contemporary reseach student who was primarily responsible for the implementation of the operational governor, as described in References 6 and 7.

1.1 Governing

The frequency of an interconnected power system, such as the National Grid, is controlled by the action of the governors on some of the plant in the system. When a drop in frequency is detected the governors increase the power to the turbines and hence the power output from the generators to maintain the frequency within the desired limits.

Thermal plant responds rapidly to governor action because the energy stored in the boiler can be released quickly. However, as the boiler pressure collapses due to the increased steam flow, the power output and hence the system frequency will fall. If the firing to the boiler is not increased, or additional plant not brought into service, the fall in output power will not be arrested and the system frequency will collapse.

Hydro plant, on the other hand, does not respond initially as quickly as thermal plant as it takes some time to accelerate the water supplying the turbine, but once the desired power output has been reached it can be maintained indefinitely. The response of the hydro plant thus complements that of the thermal plant and the advantages of operating hydro plant in an interconnected power system are apparent.

Most of the hydro plant in the United Kingdom is in the North of Scotland and was originally intended to supply areas of isolated load. The governors on this type of plant were designed to remain stable under these conditions at the expense of speed of response. Thus, the governors on most existing plant tend to be rather slow and unresponsive and any improvement in speed of response while grid connected would be benifical. More importantly, the techniques learned in these stations would be relevant to the design of governors for modern hydro stations, with installed capacities in the region of 1000MW[8,9,10,11].

1.2 Site Description

Loch Sloy Power Station[12,13] was built at Inveruglas on the north west shore of Loch Lomond in 1952. The water supply for the station comes from an upper reservoir, Loch Sloy, some 270m (880ft) above Loch Lomond. Despite a large dam and an increased catchment area the capacity of the resevoir is insufficient for continuous generation and thus the station is used only for peak loading. The general layout of the power station and the hydraulic system is shown in Figures 1.1 and 1.2 and the station and penstock are shown in Plate 1.

The station has four vertical axis Francis turbines rated at 32.5MW and operating at 428.6 r.p.m. The water for the turbines flows from the reservoir through a rock tunnel some 2750m (9000ft) in length, two concrete tunnels 180m (600ft) long and finally four steel pipes of about 490m (1600ft). A surge shaft is situated at the junction of the rock and concrete tunnels to prevent excessive pressure fluctations in the pipeline.

There is a main valve at the bottom of each pipeline adjacent to the turbine which allows the water from the pipeline to enter the spiral casing around the turbine. The flow of water from the spiral casing to the turbine is controlled by 24 guide vanes or gates arranged around the periphery of the turbine runner. The guide vanes are attached to a control ring which is operated by an hydraulic servomotor through a mechanical linkage shown in Plate 2.

The guide vanes are shaped so that the water passing through them is accelerated and the resulting kinetic energy is transferred to the turbine blades on impact. The water is further accelerated by the turbine blades as it passes through the runner providing an opposite reaction force on the runner. Thus the power output from the turbine is a combination of impulse and reaction effects.

After passing through the runner the water flows into the tailrace via a draft tube situated below the turbine and maintained at a slight vacuum.

A relief valve is provided for each turbine to divert the flow of water into the tailrace instead of through the turbine if for any reason the guide vanes have to be closed rapidly. This prevents damage to the pipeline due to large pressure surges which would be caused by attempting to decelerate the water in the pipeline too quickly. The relief valve operates on the rate of change of servomotor position and a threshold is set such that the valve does not operate under normal conditions. The actuating mechnaism of the relief is highly non-linear.

A detailed description of the construction and operation of the English Elect ric Governors used at Sloy is given in Reference 14. The governor is of a mechanical hydraulic design and operates the water control valve through the servomotor and associated linkages.

The speed of the set, with respect to a speed reference, is controlled by the governor. When the generator is connected to the grid, the speed remains constant and the speed reference is used to control the power output of the set.

During early experimental work on site[1] it was found to be impracticable to position the water control valve using the existing hydraulic servomotor, which is controlled by the mechanical speed governor, as there was no suitable electrical interface. As a result, an entirely new servomotor was installed, which operated on the same shaft as the existing servomotor, controlled by an electro-hydraulic distribution valve. Plate 3 shows a plan view of the hydraulic system with the new high pressure servomotor on the left painted a darker colour. (References 15 and 16 discuss some of the developments in hydro generator controllers and actuating mechanisms.)

Changeover from one hydraulic system to the other is effected by manual operation of by-pass and isolation valves and an electrical selector switch, as described in Appendix 1.

1.3 Simulation

Conventional methods of controller design involve the use of analytical techniques, for example Bode, Root Locus and Nyquist, to find the best structure for the controller and approximate settings for the controller parameters. Final parameter settings are then obtained by tuning the controller on site to give a satisfactory response. In earlier work[2], this type of approach was used for governor development but significant non-linearities in the plant reduced the effectiveness of this technique on its own and, as a result, a simulation model of the plant was developed to assist in obtaining more accurate parameter settings before undertaking final tuning on site.

Initially the simulation studies were carried out using an hybrid facility[5] but the problem size, set-up time and operational problems associated with the analogue computer available at that time have led to the almost exclusive use of digital simulation. A full simulation model of the hydro-generator, pipeline and servo system with associated non-linearities has been developed as part of this project, based on the work described in References 1 and 3.

This simulation model was validated by the comparision of simulated results with those obtained from site tests using both the existing governor and the new controllers. The model was updated as more information became available from site tests and as confidence in the model developed the simulation was used as a design tool to predict the performance of the new controllers.

The Real-Time Interactive Simulation Package (RISP)[17], a FORTRAN based system developed at Glasgow University for the simulation of continuous systems, was originally used for modelling the plant at Sloy on a PDP-11 computer. However, the use of this simulation package involved the user in a certain amount of programming not directly associated with the model.

To minimise the programming necessary, a simulation language, based on a non-realtime version of the simulation package was developed using a general purpose macroprocessor to translate the Model Description into FORTRAN. This language, known as GUILDS (Glasgow University Interactive Language for Dynamic Simulation) permits the user to describe the model in block diagram or differential equation form, greatly reducing the overheads involved, in terms of time and effort, when writing or developing a simulation model.

1.4 Thesis Outline

The development and implementation of the Simulation Language are presented in the following three chapters of this thesis: Chapter 2 outlines the principles behind the macroprocessor translator; Chapter 3 describes in detail the final form of the translator; and Chapter 4 discusses the FORTRAN section of the Language. The use of the Simulation Language and the facilities offered are described in the GUILDS User's Guide which has been included at the rear of this Thesis.

The model of the hydro-generator in Sloy Power Station is presented in Chapters 5 to 8: Chapter 5 gives the theoretical expressions on which the model is based; in Chapter 6 the per-unit representation is derived from the theoretical expressions; Chapter 7 presents the results of the simulation studies and associated site tests; and Chapter 8 describes a simulation study using the hydro-generator model to investigate a proposed type of non-linear governor.

In Chapter 9 a general simulation of a thermal generating plant is developed and the results of some simulation studies using the model are presented. The conclusions arising from the work described in these chapters are given in Chapter 10 which also contains some recommendations for further work.

Appendix 1 contains a paper based on some of the work carried out in Sloy Power Station which has subsequently been published in a shortened form by the IEE[18] as a technical note relating to the work discussed in Reference 4. The Model Description for all the simulation studies referred to in this thesis are collected together in Appendix 2 and the Documentation Files describing specific simulation runs are to be found in Appendix 3 along with a brief description of the conditions on site pertaining to site test results reproduced in the thesis. The GUILDS User's Guide, prepared as part of the work on the Simulation Language, has been included at the rear of the Thesis.

Part of the work in this thesis has been published, References 7 and 18, as it has related to the work carried out by the co-authors of the publications.

The sections of work which are considered to be original are as follows:-

(a) the use of a macroprocessor as a translator for a simulation language;

(b) the Simulation Language (GUILDS), including revision of the FORTRAN package to permit new ideas at the language level;

(c) the detail with which the hydro-generator is modelled;

(d) most of the work on the non-linear governor (Chapter 8) although some of the ideas had been previously suggested;

(e) certain of the aspects of the thermal plant model, particularly in relation to the combined hydro-thermal simulation.

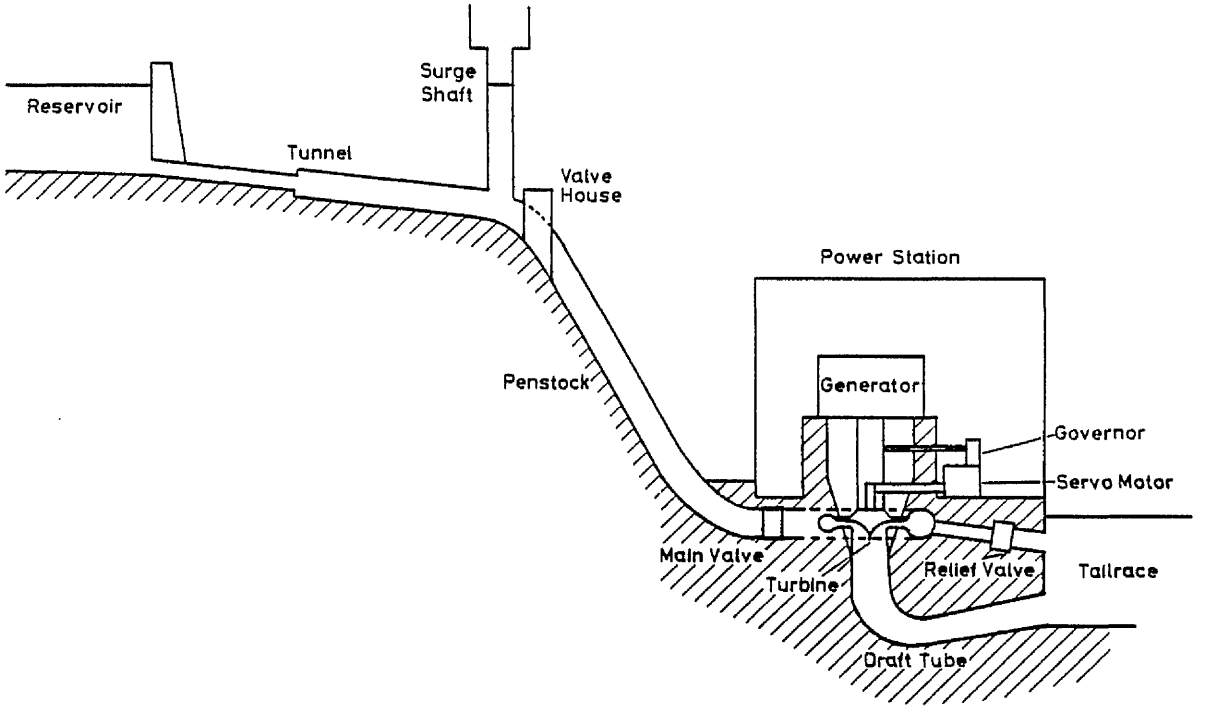


Figure 1.1 - Sloy Power Station - General Layout
(Not to scale)

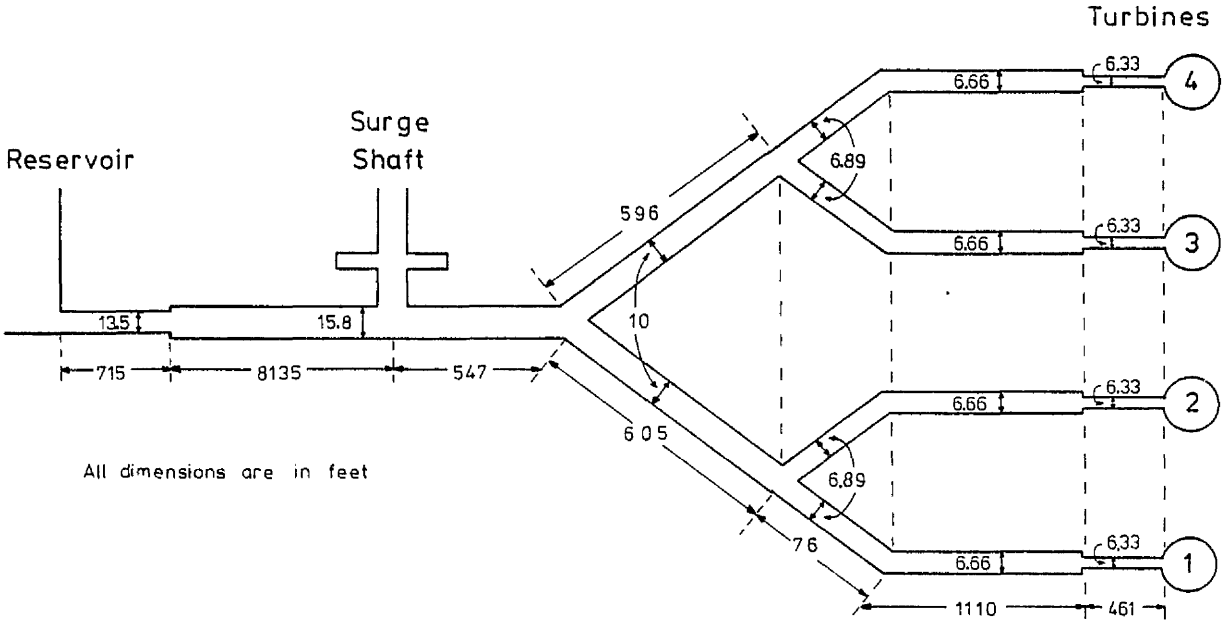


Figure 1.2 - Sloy Power Station - Hydraulic System Layout
(Not to scale)

Plate 1 - Loch Sloy Power Station

Plate 2 - Water Control Valve Actuator

Plate 3 - Hydraulic Servo System

CHAPTER 2

STAGE2 AS A CSSL TRANSLATOR

2.0 Introduction

Simulation is a means of predicting the performance of a system under specific operating conditions, testing and evaluating a proposed system or identifying those portions of a system which require further investigation. The use of a mathematical model of a system in conjunction with a computer is one of the most widely used methods of performing simulation studies.

Simulation of continuous systems (that is, systems which can be represented as a series of differential equations and associated algebraic expressions with variables which change continuously with the independent variable, usually time) was for many years the province of the analogue computer. However, with the advent of larger and faster digital machines there has been a tendancy to use analogue and hybrid (combined analogue and digital) simulation for specialised applications and to use digital techniques for general purpose simulation.

The advantages of digital simulation, over analogue techniques, are really advantages of the digital computer, for example, ease of implementation, repeatability of results, storage facilities for data, and almost unlimited mathematical operations. The main advantage of an analogue machine is the inherent high speed of operation due to the effectively parallel solution of the differential equations.

Initially, in the field of digital simulation, all studies were carried out in batch mode. The resulting poor turn-round was one of the main disadvantages of digital simulation, which persisted until the development of the interactive mini-computer. This permitted the

user of a digital simulation package to interact with the simulation during the run in a similar way to the "hands-on" approach used in analogue simulation.

To utilise the advantages of an interactive digital computer, and because there were no suitable packages available, an interactive digital simulation package, written in FORTRAN, was developed at Glasgow University for use on a PDP-11 computer[17]. However, using this package involved some detailed programming in FORTRAN to describe the model and the facilities offered to the user were somewhat limited. Thus, it was decided to develop a simulation language, based on this package which would permit the user to describe a model more easily and provide a wider range of facilities. The development of the simulation language is discussed in the following sections of this Chapter and the implementation of the language, finally named GUILDS (Glasgow University Interactive Language for Dynamic Simulation) is described in detail in Chapters 3 and 4. The GUILDS User's Guide, which has been included at the rear of this Thesis, describes the use of the Language and gives details of all the facilities provided as part of the GUILDS package.

2.1 Initial Development

As is the case with most simulation languages, there are three stages involved in creating and running a simulation using GUILDS.

(a) Input of the equations describing the model, with the values of constants and parameters - the MODEL DESCRIPTION;

(b) Translation of the Model Description into a computer orientated language - FORTRAN

(c) Compilation, linking and execution of the FORTRAN routines with a set of run-time routines which provide the integration methods and output facilities.

In the initial stages of the development of the simulation language, it was decided to use statements of a similar form to those of the IBM simulation language CSMP[19,20] as a basis for the language structure. These statements could then be translated into FORTRAN subroutines which would be compatible with the existing simulation package. Also, it was decided to investigate the possibility of using STAGE2[21,22], a general purpose macro-processor particularly suited to non-numeric applications, for writing the translator.

STAGE2 had already been used successfully by Dr. H. Davie within the Department for writing cross-compilers and assemblers and, an implementation, STAGE2B, supplied by CERL, was available on the PDP-11 computer. Some initial work was carried out to gain experience of using this somewhat different language and to find out if STAGE2 could be used to generate a sequence of FORTRAN statements from one or more input statements of the type that would be used for the simulation language. This was found to be relatively straightforward and thus it was decided to proceed with the use of STAGE2 for the simulation language translator.

The way in which STAGE2 operates is quite different from conventional programming languages and in addition, the abbreviations and symbols used, although extremely powerful, complicate the initial understanding. The operation of STAGE2 is controlled by a macro file which contains the definitions of the STAGE2 macros used for the translation process. STAGE2 attempts to match each line of the input file to a macro definition and the translation of that line is determined by the code body of the macro to which it is matched.

For example, a line of input may simply be output directly or parsed and output in a different form, passed to another macro for further translation or used to control the operation of other macros. In a typical application, such as the GUILDS translator, a large number of the STAGE2 macros in a file are not matched by lines in the input file but by lines output from other macros.

The advantages of using a language specifically developed for text handling, compared to a computational language like FORTRAN, became apparent as the translator was developed. It was possible to proceed with the writing of the translator before the language was fully defined as the facilities of the language could be extended by including additional STAGE2 macros in the translator macro file corresponding to the new input statements with little or no change to the existing macros.

2.2 STAGE2[21,22]

STAGE2 is a general-purpose macroprocessor which accepts character strings as input, transforms them according to a set of rules and produces character strings as output. The rules are presented to STAGE2 as a set of patterns or templates (the macro definition) each of which is followed by a sequence of lines (the macro code body). A generalised pattern matching process is used to match the input lines to the macro definitions and the processing of each line is determined by the code body of the macro to which the line is matched.

Since it is possible that an input line could be matched to more than one template, a set of rules are defined, within STAGE2, which attempt to eliminate any ambiguity and achieve the best match for each input line. To do this, the templates supplied by the programmer are organised into a tree structure and this tree is used for matching the input statements to the templates. In most cases the user need not

be familiar with the matching process but, if required, a detailed description can be found in references 21 and 22.

The macro definition, or template is a sequence of character strings separated by formal parameters (indicated by a parameter flag, %). When a match occurs between an input line and a template, each parameter flag corresponds to some substring of the input line (possibly the null string) and this substring becomes the actual parameter, corresponding to the formal parameter in the macro code body. In addition, it is possible, within the macro code body, to define a string as a parameter.

Parameter definitions, from either the template or code body are local to the macro and do not exist outwith the macro. It is necessary to use variable names for global references in order to transfer information from one macro to another. These variable names are in fact used as string addresses to access memory and the "value" of a variable is the string stored at that address. Initially all addresses contain a null string, which, for arithmetic operations is taken as zero although the null string and the zero string are not equal. If an attempt is made to use a variable in an arithmetic operation which contains a character string, as opposed to a numeric string, an error is produced.

When a template has been matched, STAGE2 uses the code body associated with the template to construct one or more lines of text. The "constructed line" is formed from strings drawn from the code body itself, from substrings of the input line which match parameter flags in the template associated with the code body and from the STAGE2 memory accessible to all code bodies, through references to variable names. The entire code body is considered as a single string of elements, the image of the code body in memory containing four different types of elements:-

TYPE 0 Single, literal character

TYPE 1 End-of line marker

TYPE 2 Paramter conversion

TYPE 3 Processor function

Type 2 elements specify a parameter (in the range 1 to 9) and the type of transformation that is to be performed on the parameter. These parameter conversions are listed in Figure 2.1.

Type 3 elements specify a processor function as listed in Figure 2.2. Processor functions are used for operations which cannot be conviently expressed as string transformations using the parameter conversions.

STAGE2 recognises certain characters as having a special significance. These characters are specified in a Flag Line which must be the first line of any macro file. The characters used in the flag line for all the STAGE2 work associated with the Simulation Language are listed in Figure 2.3. (It should be noted that the % symbol is used as the dummy parameter in a template and is also used as the escape character. No confusion should arise, however, as the first usage is confined to the template and the second to the macro code body.)

The significance of the escape character (%) is noteworthy, as it controls much of the STAGE2 processing. If an escape character is encountered while the code body is being read, the next characted is checked. If this character is another escape or the end-of-line flag (#) it is simply treated as a type 0 element and no special significance is attached to the character. If, however the character which follows the escape is a digit, STAGE2 recognises an instruction to undertake a parameter conversion. The first character after the escape specifies the parameter number and the next character defines

the conversion type (see Figure 2.1). For example, %10 specifies a type 0 conversion on parameter 1 and this conversion places an exact copy of the parameter string into the constructed line.

If the character following the escape is neither a digit nor one of the two special characters (% or #) a processor function is recognised, the particular function type being specified by the second character after the escape. An F is normally used as the processor function call and, for example, %F1 would be recognised as a call to a type 1 processor function which would output the constructed line directly without further processing.

During the processing of the code body, when a type 0 element is encountered, the literal character is copied into the constructed line. A type 2 element causes the specified parameter, appropriately transformed to be copied into the constructed line and a type 3 element initiates the execution of the specified processor function. When a type 1 element is encountered, a carriage return is added to the constructed line. Processing of the code body is temporarily suspended and the constructed line is presented to STAGE2 for further matching as though it had been read from the input device.

STAGE2 is normally configured with four channels or files for input and/or output. These channels have different attributes, as listed below, and tend to have specific uses as indicated.

CHANNEL 1	READ ONLY	INPUT FILE
CHANNEL 2	READ/WRITE	TEMPORARY FILE
CHANNEL 3	WRITE ONLY	OUTPUT FILE
CHANNEL 4	WRITE ONLY	ERROR FILE

Channel 2 is the only channel which can be rewound, although additional channels having the attributes of Channel 2 are available in some configurations. Type 1 and 2 processor functions can be used, in conjunction with the channel numbers, to direct the input and output.

Consider, as an example, the macro

```
%=INTGRL(%);
N%96#
INCR N#
%10=VAR(%91)%F13#
DER(%91)=%20%F13#
#
```

The first line of this macro is the template for the macro with two parameters which, in the code body, are referred to as parameters 1 and 2 numbered from left to right. If an input line A1=INTGRL(B1) is matched to this template then parameter 1 is string A1 and parameter 2 is string B1.

When execution of the code body begins, the constructed line is null. The type 0 element (N) from the start of the code body is copied into the constructed line and then a type 2 element is recognised. This element (%96) specifies a type 6 parameter conversion of parameter 9 which defines the constructed line (N) as parameter 9. The element following %96 is ignored and scanning of the code body resumes with the next element. Construction of a new line begins and the six type 0 elements (INCR N) are copied into the constructed line and then a type 1 element is recognised. STAGE2 adds a carriage return to the end of the line and outputs it for further matching as though read from the input channel.

If this constructed line matches another template, the code body associated with that template will be processed. This code body will construct other lines, which may cause further code body processing. If the constructed line does not match any template it is output as it stands. Eventually STAGE2 will return to the code body of %=INTGRL(%). At this point the constructed line has been completely processed and construction of a new line begins with the next element in the code body.

The next element is in fact type 2 (%10) and as a result of this conversion actual parameter 1 (A1) is copied into the constructed line. The next five elements (=VAR( ) are type 0 and thus are copied directly into the constructed line. Following these elements is a type 2 element (%91) which initiates a type 1 conversion on parameter 9 (N). This conversion copies the current value of parameter 9 (N), say 2, into the constructed line and then another type 0 element ()) is added to the line. The next element is of type 3 (%F1) and is recognised as a type 1 processor function, which sends the constructed line (A1=VAR(2)) to the current output channel without releasing the line for further matching. The element following a type 1 processor function is ignored and a new line is constructed beginning with the next code body element. The line DER(2)=B1 is constructed and output without being rematched when the type 3 element %F1 is recognised. Processing of the macro code body terminates when the type 1 element is recognised at the start of the next line being constructed.

This example illustrates the way in which the STAGE2 macroprocessor operates and shows how some of the processor functions and parameter conversions might be used. A more comprehensive example is to be found in the following section which makes use of many of the principles outlined above for the development of a simple translator for a simulation language.

Conversion Digit	Action
0	Copy the parameter to the constructed line
1, 2	Copy a string from memory to the constructed line
3	Copy the break character to the constructed line
4	Copy the value of the parameter, treated as an arithmetic expression, to the constructed line
5	Copy the length of the parameter to the constructed line
6	Reset the value of the parameter
7	Initiate a context-controlled iteration
8	Copy an integer equivalent to a single character into the constructed line (ASCII code)

Figure 2.1 - Parameter Conversions

Function Digit	Action
0	Terminate processing
1	Output a line without re-scanning
2	Change I/O channels and copy text
3	Store information in memory
4	Set skip counter unconditionally
5	Set skip counter conditionally on string equality
6	Set skip counter conditionally on the relative values of two expressions
7	Initiate a count controlled iteration
8	Advance an iteration
9	Escape from current macro
E	Generate error traceback (illegal function)

Figure 2.2 - Processor Functions

Character	Meaning
;	Marks the end of template or source line
%	Specifies a dummy parameter in template
#	Marks the end of a code body line
%	The escape character for a code body line
0	Zero - defines all digits
	Space - used as padding character
(or [	Open parenthesis
+	Addition operator
-	Subtraction operator
*	Multiplication operator
/	Division operator
) or]	Close parenthesis

Figure 2.3 - Flag Line Characters

2.3 An Example

Consider, as an example of both the operation of STAGE2 and the initial development of the simulation language, the equation for a first order lag, or real pole, given below.

$$Y = \frac{1}{1 + sT} * X$$

If it were necessary to use a simulation language, like GUILDS, to find the time response of a first order lag with a time constant T to a step of magnitude C1 then a series of FORTRAN-like statements, called the Model Description, could be written, as below:-

```
YDOT = (X-Y)/T
Y = INTGRL(YIC,YDOT)
```

with, for example `X = C1 * STEP(5.)`

where INTGRL and STEP are simulation language functions and C1 and T are parameters (fixed values).

The basis of a digital simulation language is an integration routine to calculate the values of the integrated variables at discrete intervals of the independent variable, usually TIME, and another routine to re-calculate the values of the derivatives, and associated variables. (For a fuller explanation of the techniques of digital simulation the reader is referred to section 2.1 of the GUILDS User's Guide.) The first of these routines is part of the simulation package and the second is produced by translation of the user-supplied Model Description. The integration routine calls the Model routine once (for first order integration methods) at each integration interval, passing the calculated values of the integrated variables to the model routine, and receives back the new values of the derivatives. The integration routine also requires to have, at the start of the computation, initial values for all the integrated variables.

In the equations given above, Y is defined as an integrated variable with an initial value YIC and YDOT is the derivative. Assuming that the values of Y and YDOT are passed to and from the integration routine as subroutine parameters and, as the integration routine is fixed and the parameters in the subroutine call cannot be varied, it is necessary to generate assignments in the Model routine to the parameters in the subroutine call. In the above example there is only one integrated variable but in general there would be many, so arrays are used to transfer the values of the integrated variables and the derivatives between the integration routine and the model routine. In the existing simulation package, the arrays VAR and DER are used for the integrated variables and derivatives respectively. Similarly, an array WW contains the values of the initial conditions of the integrated variables and the variable N is used to specify the number of INTGRL statements in the Model Description.

STAGE2 Macros

```
% = INTGRL(%,%);                 Macro template
N%96#                            Define N as parameter 9
INCR N#                          Increment N
%10 = VAR(%91)%F13#
WW(%91) = %20%F13#               Output equivalences
DER(%91) = %30%F13#              for DER, WW and VAR
#                                End of macro

%=%;                             Macro template
%10 = %20%F13#                   Output matched statement
#                                End of macro
```

Input

```
X = C1 * STEP(5.)
YDOT = (X-Y)/T
Y = INTGRL(YIC,YDOT)
```

Output

```
X = C1 * STEP(5.)                Input lines unchanged
YDOT = (X-Y)/T
Y = VAR(1)
WW(1) = YIC                      Lines generated by STAGE2
DER(1) = YDOT
```

Figure 2.4 - Example of the Translation of an INTGRL Statement

Thus, each INTGRL statement must generate three assignment statements and also, a count must be kept of the INTGRL statements as they are encountered. A STAGE2 macro which could accomplish this, and the resulting output are shown in Figure 2.4. Also shown, is a second macro required to process the other input statements. The output produced by these macros is not computationally satisfactory, as will be discussed below, but this example serves to show the main features of the translation process.

In the STAGE2 macros given, the use of several different parameter conversions and a processor function is illustrated. Referring to the INTGRL macro, the type 6 conversion defines the variable N as parameter 9 and all subsequent type 1 conversions on parameter 9 copy the value of the variable N into the constructed line. N is used in this macro as a global variable which can be accessed, through parameter 9, each time the macro is called. This is necessary since the values of the parameters are lost when the macro is exited. Parameters 1, 2 and 3 are defined in the macro template and the type 0 conversions on these parameters copy the actual parameter string into the constructed line. The type 1 processor function is used to output the constructed line to channel 3, the output file, without further processing.

The statement INCR N, when copied into the constructed line, is not output directly but is released for further template matching when the end-of-line flag (#) is encountered. Three additional STAGE2 macros, as shown in Figure 2.5, are used to process this line and increment the value of the variable N. These macros are "internal" or "system" macros as they are only matched by lines output from other macros and not by statements in the input file.

The EQU macro uses a type 3 processor function to store actual parameter string 2 at an address given by parameter string 1. The SET macro performs a type 4 parameter conversion on parameter 2 before calling the EQU macro and thus effectively stores the value of parameter string 2 at the address given by parameter string 1. Thus, the constructed line INCR N, where N has a value of 2 say, would match to the macro INCR % and a line N SET N+1 would be constructed. This line would, in turn, match to the % SET % macro which would construct the line N EQU 3. On matching to the % EQU % macro, the type 3 processor function would store the new value of N (3) at location N. Hence, all further references to N using a type 1 parameter conversion would return the value 3.

```
INCR %;
%10 SET %10+1#
#

% SET %;
%10 EQU %24#
#

% EQU %;
%F3#
#
```

Figure 2.5 - Some Additional STAGE2 Macros

The output from the macros shown in Figure 2.4 is deficient in the following ways:-

(a) the computational sequence is incorrect in that, although the value of VAR(1) has been calculated in the integration routine, which is executed before the model routine, it is not assigned to the variable Y until after this variable has been used in the expression for YDOT;

(b) the assignment to WW(1) will be repeated unnecessarily, since YIC is a constant, at each integration interval;

(c) no assignment for the variable N has been generated, although a count of the INTGRL statements has been kept, as there is no indication in the input file of when the last INTGRL statement has been encountered;

(d) finally, the output lacks many of the statements required to form a legal FORTRAN subroutine.

The latter two points can be dealt with by specifying additional statements, "translation control" statements, as the first and last statements in any input file and defining STAGE2 macros which will match these statements and generate the necessary output. Figure 2.6 lists the additional macros which would be required along with the corresponding input and output.

Despite the use of translation control statements deficiencies (a) and (b) above still exist and, in addition, the assignment for N, which has now been included, is also repeated unnecessarily at every integration interval. By including a separate subroutine which is executed at the start of the simulation run which contains the assignments for WW(1) and N, these statements can be removed from the Model subroutine. (This new subroutine could also be used for assigning values to parameters and constants, for example C1 and T, as described later.) However, to generate a second subroutine, it would first be necessary to read the whole of the input file so that, in cases where there are several INTGRL statements, all the initial conditions could be gathered and the number of INTGRL statements counted.

STAGE2 Macros

```
START;
     SUBROUTINE MODEL(VAR,DER)%F13#
     DIMENSION VAR(10), DER(10), WW(10)%F13#
#

END;
N%96#
     N = %91%F13#
     RETURN%F13#
     END%F13#
     #
```

Input

```
START
X = C1*STEP(5.)
YDOT = (X-Y)/T
Y = INTGRL(YIC,YDOT)
END
```

Output

```
     SUBROUTINE MODEL(VAR,DER)
     DIMENSION VAR(10), DER(10), WW(10)
     X = C1 * STEP(5.)
     YDOT = (X-Y)/T
     Y = VAR(1)
     WW(1) = YIC
     DER(1) = YDOT
     N = 1
     RETURN
     END
```

Figure 2.6 - Producing a FORTRAN Subroutine

This could be accomplished, using STAGE2, by storing all the statements which have to be output until the END statement is encountered and then the output file could be generated in the correct sequence. STAGE2 has the facilities for doing this, but the disadvantage of this approach is that the amount of storage required would very rapidly limit the size of problem which could be dealt with by the translator.

An alternative approach would be to use a multi-pass method where the input file is read more than once and the STAGE2 macros use logical branching to execute different sections of code body for each pass. Also, by using this method it is possible to group all the

assignments for the integrated variables (e.g. Y = VAR(1)) at the start of the Model subroutine, thus producing the correct computational sequence. The STAGE2 macros required for translating the example given above are shown in Figure 2.7 and the input and output are shown together in Figure 2.8.

Two FORTRAN subroutines, INIT and MODEL, are produced by the STAGE2 macros from four passes of the the input file. The sections of output required are:-

(1) statements for the start of subroutine INIT;

(2) assignments to the WW array of the initial conditions for each INTGRL statement;

(3) assignment to N of the number of INTGRL statemets;

(4) statements for the end of subroutine INIT and the start of subroutine MODEL;

(5) assignments to the integrated variables from the VAR array;

(6) assignments to the DER array of the derivatives and any other assignment statements;

(7) statements for the end of subroutine MODEL.

These sections of output cannot be produced by a single pass and in particular sections 2 and 3, section 5 and section 6 all require separate passes as the information being output in these sections is collected from throughout the file.

Initially the pass counter, RERUN, is null, arithmetically zero, but each time the START macro is called this variable is incremented. As the INCR RERUN statement is at the beginning of the START macro the first pass is pass 1, the second, pass 2 and so on.

The first pass, which only operates in the START macro, is used to copy the input file from Channel 1 to a temporary file on Channel 2. It is necessary to use a temporary file because the input file on Channel 1 cannot be rewound and each pass of the translator requires the input to start at the beginning of the file.

The copy operation is performed by initially calling a type 1 processor function to output the START statement to Channel 2 (START%F12#, the character following the %F1 specifying the channel) and then using a type 2 processor function (%F22) which copies lines of input from the input channel (1) to the output channel (2) until a string equal to parameter 1 is encountered at the start of a line of input. As parameter 1 has been set equal to END (END%16#) the complete input file (excluding the START and END statements) is copied to Channel 2. It then remains to write the END statement to Channel 2 (END%F12#) and rewind the the temporary file. The REWIND macro is called which also uses the type 2 processor function (2R%F23#) but, as parameter 1 is null in this macro, no copying takes place. The operation of the type 2 function in this macro is to set channel 2, rewound (2R), as the input channel and Channel 3 as the output channel.

After the copy and rewind operation, the second pass begins, with input being read from the temporary file, and the first statement input, START, is again matched to the START macro. The pass counter, RERUN, is now 2 and the start of subroutine INIT is output. During this pass the INTGRL macro outputs the assignments to the WW array and the END macro outputs the assignment for N, the number of INTGRL statements. (As N is used to specify the array subscripts for WW, DER, and VAR during different passes it is necessary to reset N at the start of each pass so that the correct correspondence between the array elements is obtained, for example the first INTGRL statement generates references to WW(1), DER(1) and VAR(1).)

In pass three, the START macro outputs the end of subroutine INIT and the start of subroutine MODEL. The INTGRL macro outputs the assignments of the integrated variables to the VAR array but no other processing takes place. During the fourth pass, the assignments for the DER array are output by the INTGRL macro and the other assignment

statements in the input file are output by the % = % macro. The END macro outputs the final statements of the MODEL subroutine and terminates the STAGE2 processing.

Some additional internal macros are required to perform the logical branching in the STAGE2 macros. To permit branching STAGE2 uses a "skip counter" to facilitate the skipping of lines in the macro code body or the input file. The skip counter can be set unconditionally or conditionally by using the type 4, 5 or 6 processor functions. The value of the skip counter is checked before each line of the code body is executed and before each input line is read. If the skip counter is non-zero the line is ignored and the counter is decremented.

The two macros used for logical branching in the example given (Figures 2.7 and 2.8), IF % = % SKIP% and IF % <> % SKIP% use the type 6 processor function. The character following the %F6, either 0 or 1 for the macros given, determines whether the skip counter is set, to the value of parameter 3, on the equality or inequality of the values of the parameter strings 1 and 2. If the specified condition is not satisfied the skip counter is set to zero.

STAGE2 Macros

```
START;
INCR RERUN#
N EQU 0#
IF RERUN <> 1 SKIP6#
START%F12#
END%16#
%F22#
END%F12#
REWIND#
%F9#
IF RERUN <> 2 SKIP4#
     SUBROUTINE INIT%F13#
     DIMENSION WW(10)%F13#
     COMMON/A/WW,N
%F9#
IF RERUN <> 3 SKIP4#
     RETURN%F13#
     END%F13#
     SUBROUTINE MODEL(VAR,DER)%F13#
     DIMENSION VAR(10), DER(10)%F13#
#

% = INTGRL(%,%)
N%96#
INCR N#
IF RERUN <> 2 SKIP2#
     WW(%91) = %20%F13#
%F9#
IF RERUN <> 3 SKIP2#
     %10 = VAR(%91)%F13#
%F9#
     DER(%91) = %30%F13#
#

% = %;
IF RERUN <> 4 SKIP1#
     %10 = %20%F13#
#

END;
IF RERUN = 3 SKIP3#
IF RERUN = 4 SKIP4#
N%96#
     N = %91%F13#
REWIND#
%F9#
     RETURN%F13#
     END%F13#
%F0#
#

REWIND;
2R%F23#
#
```

Figure 2.7 - STAGE2 Macros for Multi-pass Translation Example
(Part 1 of 2)

```
IF % = % SKIP%;
%F60#
#

IF % <> % SKIP%;
%F61#
#

INCR %;
%10 SET %10+1#
#

% SET %;
%10 EQU %24#
#

% EQU %;
%F3#
#
```

Figure 2.7 - STAGE2 Macros for Multi-pass Translation Example
(Part 2 of 2)

Input

```
START
X = C1*STEP(5.)
XDOT = (X-Y)/T
Y = INTGRL(YIC,YDOT)
END
```

Output

```
      SUBROUTINE INIT
      DIMENSION WW(10)
      COMMON/A/VW,N
      WW(1) = YIC
      N = 1
      RETURN
      END

      SUBROUTINE MODEL(VAR,DER)
      DIMENSION VAR(10), DER(10)
      Y = VAR(1)
      X = C1*STEP(5.)
      YDOT = (X-Y)/T
      DER(1) = YDOT
      RETURN
      END
```

Figure 2.8 - Input and Output for Multi-pass Translation

To complete this translation example it is necessary to have some means of defining the values of parameters in the Model Description. In particular values for YIC, C1 and T are required. Assignment statements could be used, which, if there are many parameters, would involve a large number of statements. As the values of these parameters are essentially data which does not change with time a more convenient method would be to include a series of assignments in one statement which could be readily translated into FORTRAN DATA statements.

As shown in Figure 2.9, a CONSTANT statement has been included in the input file to specify the values of YIC, C1 and T, and a CONSTANT macro has been defined which, in addition to the macros given in Figure 2.7, generates the FORTRAN subroutines also shown in this Figure. The CONSTANT macro produces a COMMON block in both the FORTRAN subroutines and a DATA statement in subroutine INIT which assigns the values of the parameters to the variable names.

Two additional macros are required for use in the CONSTANT macro and these macros are also shown in Figure 2.9. The IF % EQ % SKIP% macro is similar to the IF % = % SKIP% macro except that in this case the skip counter is set on the result of the comparison of parameters 1 and 2 considered as strings, using a type 5 processor function. The SKIP% macro uses the type 4 processor function to set the skip counter unconditionally to the value of parameter 1.

The CONSTANT macro illustrates the use of type 3 and 7 parameter conversions and a type 7 processor function for carrying out a "context controlled iteration". Initially, when the input line CONSTANT YIC=0.,C1=2.,T=0.1 is matched by the macro CONSTANT %=%,% parameter 1 is YIC, parameter 2 is 0. and parameter 3 contains C1=2.,T=0.1. The variables LIST1 and LIST2 are defined as parameters 9 and 8 respectivley.

With the pass counter, RERUN, equal to two, the CONSTANT macro assigns parameter 1 to LIST1 and parameter 2 to LIST2 and then a context controlled iteration is initiated on the constructed line, actual parameter 3, (%30%37,=#). The context controlled iteration is initialised by saving the specified parameter (3) and defining the characters following the parameter call (, and =) as break characters. The iteration is advanced by assigning to the parameter the string of characters from the start of the constructed line up to, but excluding, the first break character. This string, and the break character which follows, are deleted from the constructed line and the execution of the code body continues. In the following line, the type 3 parameter conversion is used to obtain the break character and if this character is a ',' then, a skip takes place and the next line executed adds the string in parameter 3 to that in LIST2. If, on the other hand the break character is not a ',' but an '=' then the skip does not take place and the string contained in parameter 3 is added to that in LIST1.

The context controlled iteration is advanced by the type 8 processor function (%F8) and the cycle described above is repeated until the constructed line is null. At the end of the cycle, for the example given, LIST1 will contain YIC,C1,T and LIST2 will contain 0.,2.,0.1. Thus, after the iteration is complete, these parameters can be used to generate the COMMON block and DATA statement required for subroutine INIT.

On the third pass, the iteration does not require to be repeated as the contents of LIST1 have not been altered and thus, the COMMON block for subroutine MODEL can be output directly. No action is taken by this macro during the fourth pass.

The scope of this example is, of necessity, somewhat limited but an attempt has been made to illustrate as many of the features of STAGE2 as possible and the way in which they are used in the translator for the simulation language. The input and output shown have been simplified for the sake of clarity (more detail is given in Chapter 3) but the principles adopted in the developing this example have followed, as closely as possible, those used in developing the Simulation Language. As the language was being used for simulation studies throughout the development, many of the features included in the language were included to facilitate these studies. The following Chapter describes the final version of the translator and discusses in some detail the implementation of the facilities offered.

STAGE2 Macros

```
CONSTANT %=%,%;
LIST1%96#
LIST2%86#
IF RERUN <> 2 SKIP11#
LIST1 EQU %10#
LIST2 EQU %20#
%30%37,=#
IF %33 EQ , SKIP2#
LIST1 EQU %91,%30#
SKIP1#
LIST2 EQU %81,%30#
%F8#
     COMMON/B/%91%F13#
     DATA %91/%81%F13#
%F9#
IF RERUN <> 3 SKIP1#
     COMMON/B/%91%F13#
%F9#
#

IF % EQ % SKIP%;
%F50#
#

SKIP%;
%F4#
#
```

Figure 2.9 - Translation of a CONSTANT Statement
(Part 1 of 2)

Input

```
START
CONSTANT YIC=0.,C1=2.,T=0.1
X = C1*STEP(5.)
YDOT = (X-Y)/T
Y = INTGRL(YIC,YDOT)
END
```

Output

```
      SUBROUTINE INIT
      DIMENSION WW(10)
      COMMON/A/WW,N
      COMMON/B/YIC,C1,T
      DATA YIC,C1,T/0.,2.,0.1
      WW(1) = YIC
      N = 1
      RETURN
      END

      SUBROUTINE MODEL(VAR,DER)
      DIMENSION VAR(10), DER(10)
      COMMON/B/YIC,C1,T
      Y = VAR(1)
      X = C1*STEP(5.)
      YDOT = (X-Y)/T
      DER(1) = YDOT
      RETURN
      END
```

Figure 2.9 - Translation of a CONSTANT Statement
(Part 2 of 2)

CHAPTER 3

THE SIMULATION LANGUAGE TRANSLATOR

3.0 Introduction

Glasgow University Interactive Language for Dynamic Simulation (GUILDS) is the end product of the simulation language development outlined in Chapter 2. The use of the language and the facilities available are described in the GUILDS User's Guide (which is to be found at the rear of this Thesis), thus the discussion in this Chapter is confined to the implementation of the translator. Chapter 4 describes the FORTRAN routines provided by the simulation language and some of the modifications to the FORTRAN simulation package. The STAGE2 macro files which form the GUILDS translator are somewhat numerous and thus have not been included in this Thesis. Figures 3.1 to 3.5, 3.7, 3.8 and 3.15 list all the macros used by the translator and give a brief description of the operation of the macros.

3.1 Structure of the Model Description

Details of the structure of the Model Description, including, for example, syntax and statement ordering, are given in the GUILDS User's Guide, thus only a brief outline is given here. The structure is based on the SCi standard for CSSL's (Continuous Systems Simulation Languages)[23] and where this is imprecise the general form of the structure of CSMP[19,20] has been followed.

The Model Description is created using the text editor of the host computer and comprises three sections, INITIAL, DYNAMIC and TERMINAL (the first and last being optional) and some additional data and translation control statements. The INITIAL section contains statements giving the values of constants and initial conditions and any calculations which need to be performed before the simulation is

run. The DYNAMIC section describes the model in terms of first order differential equations and associated algebraic expressions. The TERMINAL section is for any calculations which may be required after the simulation, for example re-computing values for a subsequent run.

3.2 Processing of the Model Description

The input file is translated by three independent phases of STAGE2 into the FORTRAN subroutines INIT, MODEL and TERM corresponding to the sections of the input file. The first phase, which is optional, expands user defined Macros and removes the Macro Definitions from the input file. The second phase, also optional, is a Sorting Algorithm which, where necessary, orders the statements in the DYNAMIC section into the correct computational sequence. The third phase, the Translator, produces the FORTRAN subroutines from the input file, possibly pre-processed by the first two phases. The action of all three phases is described in detail in the following sections.

Three phases of processing are used, since each phase involves a different type of processing, independent of the other phases, which can be readily implemented as a separate STAGE2 process. This permits the user to select the phases of processing required for any given input file and the output from the Macro and/or Sorting phases can be used subsequently as the starting point for further modifications to the model. Also, by splitting the processing into three phases, the size of the STAGE2 macro file required for any one phase is reduced and thus a larger Model Description can be processed within the available memory on the PDP-11.

In addition to the three FORTRAN subroutines INIT, MODEL and TERM, part of a fourth subroutine is produced by the translator which is processed by a fourth STAGE2 macro file. This fourth subroutine is called by the FORTRAN package when a run-time documentation file is

being created (see section 3.2.4 and 4.2.3). Initially, the fourth subroutine was produced as part of the translation phase but it was decided to produce only the start of the subroutine in this phase and use a separate macro file to complete the translation to further reduce the size of the Translator macro file.

3.2.1 Macro Expansion Phase

The handler for user defined Macros, the first phase of the STAGE2 processing, removes the Macro Definition and Code Body from the start of the input file and replaces each Macro Call in the file using the appropiate Macro Expansion. The variables in the Macro Definition are formal parameters and are replaced by the real parameters in the Macro Call when the Macro is expanded.

The STAGE2 macros used for handling user defined Macros are shown in Figures 3.1 to 3.3. Figure 3.1 lists the general purpose STAGE2 macros required by all phases of the translator for arithmetic and string operations. Figure 3.2 lists the macros which are used to process lines of input and in Figure 3.3 the internal macros associated specifically with handling the user defined Macros are given. The macros given towards the end of Figure 3.2, for example <sp>% (where <sp> represents the space character), are required for handling the variety of spaces and tabs which may be used in the input file.

An example of a user defined Macro is given below:-

```
MACRO X1,X2 = ARITH [Y1,Y2,Y3,Y4]        Macro Definition
W1 = Y1 * Y3
X1 = Y1 + Y2 + W1                         Macro Code Body
X2 = Y3 + Y4 + W1
ENDMAC                                    End of Macro
```

A call to this Macro might be, for example

```
A1,A2 = ARITH [B1,B2,B3,B4]
```

which would be expanded, using the Macro Code Body, as

```
ZZM17 = B1 * B3
A1 = B1 + B2 + ZZM17
A2 = B3 + B4 + ZZM17
```

where ZZM17 is a typical generated variable, unique to this Macro Expansion.

When a Macro Definition is recognised by the STAGE2 macro MACRO %=%[%], a flag is set equal to 1, to indicate that, when a macro call is encountered, the macro has previously been defined. The name of the variable used for the flag is formed from the character string MAC followed by a number, resulting from a type 2 parameter conversion of the Macro name. The type 2 conversion uses the specified parameter to address memory and copies the contents of that memory location into the constructed line. If the location is undefined, that is the address has not previously been used, the current value of a variable, known as the Symbol Generator, is stored at that location and the Symbol Generator is incremented. Thus, for example, if the Macro Definition given above is encountered the location ARITH is addressed and as this will be the first reference to this address, the current value of the Symbol Generator, say 5, is stored at ARITH and the Code Body line MAC%22 EQU 1# stores 1 at the location MAC5.

Context controlled iteration is used to separate the formal parameters on both sides of the Macro Definition (i.e. %10 and %30). A type 2 conversion of each parameter is used as a suffix for the name PARAM and a number, giving the position of the parameter in the Macro Definition is stored at this location. These numbers are used subsequently when the Code Body lines are being saved for the Macro Expansion.

All lines read from the input file following the Macro Definition are assumed to be part of the Macro Code Body until an ENDMAC statement is encountered. These lines are stored, for use in the Macro Expansion, at a location given by suffixing the name LINE with a type 2 conversion of the Macro name and also with a line number. The number of lines stored is also saved in a variable given by suffixing MLCNT with a type 2 conversion of the Macro name.

If a Macro Call is recognised in the Code Body of the current Macro then the Macro Expansion of that Macro is stored as part of the Code Body of the current Macro. Thus, it is necessary that a Macro called from the Code Body of another Macro has previously been defined.

The lines of the Code Body are stored in such a way as to be easily used in the Macro Expansion. Each line of the Code Body is parsed using context controlled iteration and each occurance, in a line of the Code Body, of a variable which has appeared as a formal parameter in the Macro Definition, is replaced by 'n' where n is the number associated with that parameter in the Macro Definition. Similarly, any variable in the Code Body line which is not a parameter in the Macro Definition is replaced with "VAR", where VAR is the variable name. Such variables, local to the Macro Code Body, require to be replaced by unique global variables each time the Macro is expanded. Also, any FORTRAN statement labels in the Code Body line are replaced with ^xxx^, where xxx is the label, so that the statement label can be recognised and replaced with a uniquely generated label when the Macro is expanded. All other elements of the of the Code Body line, for example, operators, numeric operands and function names, are stored unchanged. The Macro Definition and Code Body are not sent to the output file.

The use of FORTRAN statements, other than assignment statements, in a Macro is only permitted in a PROCEDURE or if the Macro is called in a NOSORT section of the input file (see section 3.2.2). There are a group of STAGE2 macros, shown in Figure 3.2, which match any FORTRAN statements in the input file. If the statement matched is part of a Macro Code Body then the statement is parsed and stored as described above, otherwise the statement is output unchanged. FORMAT statements are not permitted in Macros thus, statement labels in READ and WRITE statements are not processed and each time the macro is expanded the same FORMAT statement is used. The responsibility lies with the user to ensure that a FORMAT statement is supplied and appears in a NOSORT section of the DYNAMIC segment.

When a Macro Call is recognised by the STAGE2 macro %=%[%], a check is made on the MAC flag to ensure that the Macro has been defined, and, if not, an error is signalled and processing terminated. If the Macro has been defined, the Macro Call is replaced by the Macro Expansion in the output file. The Code Body lines stored when the Macro Definition was encountered are retrieved to form the Macro Expansion.

In each line of the Macro Expansion each occurance of 'n' is replaced by the n'th parameter in the Macro Call and hence the formal parameters of the Macro Definition are replaced by the parameters of the Macro Call. Each occurance of "VAR" is replaced by a generated variable, of the correct type, that is IZMxx for integers and ZZMxx for reals, which is unique within the output file. Similarly, each occurance of ^xxx^ is replaced by a generated statement label, again unique within the output file. Two STAGE2 variables are used to generate these unique numbers, one for variable names and one for statement labels. The STAGE2 macro DUMMY %,% is used to determine if a local variable ("VAR") or statement label (^xxx^) has been encountered

in the current macro expansion. On the first occurance of a local variable or statement label, the current value of the appropriate STAGE2 counter is used and is then incremented. Subsequently, the value generated at the first occurance of the variable or statement label is returned.

A full example of the use of the Macro facility is to be found in Appendix B of the GUILDS User's Guide.

3.2.2 Sorting Phase

The second phase of the STAGE2 process is the Sorting Algorithm which orders the statements in the DYNAMIC section of the input file into the correct computational sequence. A statement can only be evaluated at the position allocated to it in the input file if all the inputs to the statement, that is all the variables on the right hand side, have been previously defined. Otherwise, the statement must be moved, relative to the other statements, to a position at which it can be evaluated.

The STAGE2 macros for the Sorting Algorithm are listed in Figures 3.4 and 3.5 and in addition the macros in Figure 3.1 are also used. Figure 3.4 lists those macros which match to lines read from the input file while those in Figure 3.5 are internal macros used in the sorting process. As was noted in section 3.2.1, a number of macros are required to handle the variety of spaces and tabs which can be used in the free format input permitted by GUILDS. These macros are grouped at the end of Figure 3.4. It should also be noted that although some macros appear to be capable of processing lines of input in each pass, the action taken by a macro during the first pass often prevents the macro matching any lines of input in subsequent passes. An example of this is the PROCEDURE %=%(%) macro which outputs %10=%20(%30) to Channel 2 without the PROCEDURE control word and thus no lines input during the second pass will match to the PROCEDURE template.

For the sorting algorithm, a variable is said to be defined if:-

(a) it has previously appeared on the left hand side of an assignment statement in INITIAL or DYNAMIC;

(b) it has appeared in a data statement of some type;

(c) it is a integrated variable (see below);

(d) it is a system variable, for example TIME.

An integrated variable is a variable which appears on the left hand side of an INTGRL statement and, since integration is predictive, the value of the variable is known at the start of the Model subroutine, before the statement has been encountered. There are some additional GUILDS functions, as indicated in Appendix C2.1 of the GUILDS User's Guide, which are translated into INTGRL statements and hence the variables on the left hand sides of these functions can be considered as integrated variables.

If an INTGRL (or similar) statement is used as part of an expression and not as a separate statement then the value of the variable on the left hand side of the statement is not known prior to the execution of the DYNAMIC segment. For example, the value of A in the statement

```
A = INTGRL(AIC,B)
```

is known at the start of the Model routine, whereas the value of C in the statement

```
C = D * INTGRL(AIC,B)
```

is not known until the statement has been encountered and then only if the value of D has already been defined, as described above.

By default the whole of the DYNAMIC segment is sorted but, by the use of the SORT and NOSORT statements, the sorting algorithm can be directed to act on only those sections specified. Similarly, a PROCEDURE can be used which allows a group of statements to be sorted as a group, the order of the statements in the group remaining unchanged. These features are particularly useful since, for example, sections containing logical branching would not make any sense if the statements were sorted. The responsibility for ordering the statements in the INITIAL and TERMINAL segments lies with the user as the computational sequence of these segments is not checked.

An example of the structure of a Procedure is given below:-

```
PROCEDURE A1,A2 = PROCNAM (B1,B2,B3)     Procedure Definition

     A1 = B1 * SQRT( B2 * B3 )
     IF(A1.LT.0) GOTO 10
     A2 = B2 * SQRT( A1 * B3 )           Procedure Statements
     GOTO 20
10   A2 = B2 * SQRT(( A1 * B3)/B1)
20   CONTINUE

ENDPROC                                  End of Procedure
```

where A1,A2 are the outputs from statements in the Procedure, B1,B2,B3 are inputs required by statements in the Procedure and PROCNAM is a dummy name associated with the Procedure. Thus, the Procedure statements must be positioned after statements defining the variables B1, B2 and B3 and before any statements that require the values of A1 or A2 as inputs. A full example of the use and translation of a Procedure is given in Appendix B of the GUILDS User's Guide.

In order to establish the integrated variables the whole input file must be read before the statements in the file can be sorted. Thus, the input file is copied to a temporary file on Channel 2 (see section 2.2) so that, after the first pass the file can be rewound for a second or subsequent pass. To reduce the number of passes required, the copy operation is not carried out directly

(as described in section 2.3) but is part of the first pass. Each line of the input file is read and matched to one of the STAGE2 macros listed in Figure 3.4.

If, during this copy operation, a PROCEDURE definition is recognised, a flag, formed by using a type 2 conversion of the procedure name as a suffix to the name PROC, is set equal to 1. The procedure definition is output to Channel 2, with the PROCEDURE control word removed and thus the statement resembles a function call. The Procedure is recognised on subsequent passes by using a type 2 conversion of the function name to reference the PROC flag. The lines of input following a PROCEDURE statement, up to but excluding the ENDPROC statement, are stored and not output to Channel 2. The ENDPROC statement is then deleted.

The Procedure statements are stored, in a similar way to those of a Macro Code Body (see section 3.2.1) at a location given by suffixing the name LINE with a type 2 conversion of the Procedure name and a line number. Similarly, the number of lines stored is also saved, at a location PLCNT, with a type 2 conversion of the Procedure name as a suffix.

During this pass, a search is made for INTGRL and other similar statements (see Appendix C2.1 of the GUILDS User's Guide) which define integrated variables. A flag, given by suffixing the name LIST with a type 2 conversion of the integrated variable name is set equal to 1 to indicate that the variable has been defined and the statement is output to Channel 2. Also, if a FORTRAN non-assignment statement, for example IF(X.GT.3.14) GOTO 10, is recognised in a SORT section of the input file, an error is signalled and processing terminated. All other lines of input are output directly to Channel 2.

When the END statement is encountered, END is written to Channel 2 and then the REWIND macro is called. This macro rewinds the file on Channel 2, specifies Channel 2 as the new input channel and Channel 3 as the new channel for output. If there were no Procedures in the original source file, then the input file on Channel 2 would be identical to the original source file.

On the second pass (with Channel 2 as input) all the variables in data statements and on the left hand side of assignment statements in the INITIAL segment are added to the list of defined variables since the values of these variables are known before the execution of the DYNAMIC segment. Data statements, for example CONSTANT A=1.,B=2., match to the macro % %=% and the variables in the data statements are defined by outputing the string %10=%20 for further processing. This string is matched to the macro %=%, as would any data continuation statements or assignment statements, but as a flag has been set by the % %=% macro, there is no output from the %=% macro and the only action of this macro is to parse the strings %10 and %20 and define any variables encountered.

The subsequent operation of the Sorting Algorithm (for the statements in the DYNAMIC segment during pass two) is shown in the flow chart in Figure 3.6 and the following comments refer to this figure.

Each statement is input in turn and matched to the appropriate macro. If the statement is in a NOSORT section then it is output to Channel 3 and the variable on the left hand side is defined. This is necessary since the variable may be used on the right hand side of a statement in a subsequent SORT section. If on the other hand the statement appears in a SORT section then the statement is passed to the SORT %=% macro for analysis.

The SORT macro uses a context controlled iteration to parse the expression on the right hand side of the statement and checks if each variable in the expression has been defined. If all the variables have been defined then the expression can be evaluated at the current position and the statement is output to Channel 3. Any undefined variables in an expression cause the statement to be stored in a list of undefined statements and a counter is incremented. Each time a statement is output and a new variable defined, this counter is checked and, if non-zero, statements are recalled from the list and passed to the sorting algorithm as though read from the input file. If the END (or TERMINAL) statement is encountered and there are still statements which have not been output, these statements are output and an error is signalled giving the first undefined variable in each statement.

If a function name is recognised as a Procedure, using the FPROC macro, the variables on the right hand side of the statement are checked, as for any other statement. If all the input variables have been defined then the output variables, on the left hand side, are defined and the Procedure statements are recalled and output to Channel 3. The Procedure statement is added to the list of undefined statements if any of the input variables are undefined when the Procedure statement is recognised.

3.2.3 Translation Phase

The third phase of the STAGE2 process is the translation of the input file, after Macro Expansion and Sorting if required, into the FORTRAN subroutines. The Translator has to make several passes through the input file because the order of the information in the input file is not necessarily as required for producing the output file. For example, the initial conditions for the integrated variables are passed to the integration routine from subroutine INIT but appear in the DYNAMIC section of the input file. Thus, a pass through the input file

is needed to collect all the initial conditions before they can be output as assignment statements in subroutine INIT. (The example given in section 2.3 explains in more detail the requirements for a multi-pass Translator.)

STAGE2 accesses the statements in the input file sequentially and, once the end of the file has been reached, it is necessary to rewind the file so that the statements can be input from the beginning again. As the input channel of STAGE2 cannot be rewound, the first operation of the Translator is to copy the input file to a temporary file on Channel 2 which can be rewound.

Each input statement is matched to the same STAGE2 macro in each pass thus a pass counter is used to control the operation of each macro for the different passes. Some macros may do nothing during certain passes whereas others may take the same action each time.

The output file is constructed sequentially with subroutine INIT first, followed by MODEL and then TERM if required. Statements are sent to the output file when the information necessary to construct the statement has been gathered from the input file and all preceding statements have been output. Statements common to all subroutines, for example COMMON blocks, are stored when first generated and output as required in the subroutines. By reference to the translation example shown in Figures 3.9 to 3.12 and the tables of STAGE2 macros in Figures 3.7 and 3.8 the operation of the translator can be followed in detail. Figure 3.7 lists all the macros required for matching lines of input with, at the end, three additional STAGE2 macros which are matched by lines output from other macros. The terms STORE, PRINT, and REWIND in Figures 3.7 and 3.8 are used to indicate calls to these macros. Figure 3.8 lists all the macros required to match GUILDS functions.

There are basically three different types of input statements, although some can be subdivided, as described in the following paragraphs. The operation of the translator and the output is determined by the macro to which the statement is matched. The macros for statements of the same type tend to perform similar functions (see Figures 3.7 and 3.8). Specific examples of the translation of the different types of statements may be found in the following section. Section 3.2 of the GUILDS User's Guide defines all the input statements which can be used in a Model description.

Data Statements

Data statements can be either assignment or interrogative in type. That is, a value can be specified when the Model Description is written (e.g. PARAMETER) or the user can be asked to specify a value at run-time (e.g. ASK). The first type is translated into FORTRAN DATA statements whereas the latter requires READ/WRITE statements; in both cases COMMON blocks are used for transfering the values of the variables between subroutines. The UPDATE statement is a combination of both these types in that a default value for the each variable is specified when the Model description is written, but the user has the opportunity to "update" the value at run-time.

Structure Statements

The structure statements of the language define the functional relationships between the variables. These statements are mainly FORTRAN assignment statements with some additional GUILDS functions. All the variables on the left hand sides of the structure statements in the DYNAMIC segment are included in a COMMON block (where the variable is integer or double precision a real single precision equivalent is included in the COMMON block and an additional assignment statement is inserted). This COMMON block is EQUIVALENCEd elsewhere in the FORTRAN package to an array DAT which is used for

communicating the values of the variables to the output routines of the Simulation Language. Another COMMON block, EQUIVALENCEd to an array HEADER, is generated which is used to pass the names of the variables to the output routines.

Apart from GUILDS functions (see Appendix C of the User's Guide and Figure 3.8), the statements in the Model Description are not translated and only require to be positioned in the correct place in the output file. Some of the GUILDS functions, for example the INTGRL statement, require to be translated and may generate a number of statements distributed through the FORTRAN subroutines. In the case of the INTGRL statement, the generated output consists of an assignment statement in INIT setting up the initial conditions and two assignment statements at different points in MODEL transferring values from and to the integrated variable (VAR) and derivative (DER) arrays respectively (see section 2.3).

Translation Control Statements

The translation control statements, for example INITIAL, are used to control the sequence of operation of the translator. In particular they set flags for use in other macros and output the statements formed from information gathered in other macros, for example, COMMON blocks formed from data statements by the PARAMETER macros are output at the start of the appropriate subroutines by the INITIAL DYNAMIC and TERMINAL macros.

An Example

An example of the translation process is given for a Model Description of the mass spring and damper system shown in Figure 3.9. The statements describing the dynamics of the system, five in all, first require to be sorted and then are translated into FORTRAN. Figure 3.10 gives a listing of the Model Description and Figure 3.11 shows the output from the sorting phase. Apart from ordering the

statements of the model description, the sorting processor also outputs all the statements in a standard format as shown. This is the format which is recommended for writing model descriptions and is used for the output from the macro expansion and sorting phases because leading spaces and tabs are deleted during the processing and hence any spaces or tabs inserted by the user would be lost.

The output from the translation process is a file containing concatenated FORTRAN subroutines. In the listing given (Figure 3.12), the FORTRAN file has been annotated to show how the translator composes the output from the input file. For each line of output the start of the corresponding source line is given along with the name of the macro which outputs the line and the number of the pass during which it is output. Certain lines of output do not correspond to any input lines but are necessary for operation with the FORTRAN package. These lines are output unchanged for every simulation module created, whereas the remainder of the output lines are determined by the statements in the input file.

For the translation phase the TITLE statement is used to initialise and reset flags and to control the multiple passes through the input file. When the TITLE statement is first encountered, the TITLE % macro, which matches to the statement, sets a pass counter to 1 and copies all the input file to a temporary file which is then rewound and becomes the input file for subsequent passes.

If no TITLE statement were present, the first legal statement (one of CONSTANT, INCON, PARAMETER, UPDATE, INITIAL or DYNAMIC) would write a default TITLE to the temporary file before copying the input file to the temporary file and rewinding, as before. This ensures that the translation process always starts in the same way with all the counters and flags set up correctly. Since STAGE2 starts off with all memory locations containing the null string (in arithmetic

terms, zero), the initialisation of counters is unnecessary and is only done for completeness. The setting of flags and control of the pass counter is, however, essential.

For the second pass, input starts again with the TITLE statement and the TITLE % macro outputs the start of subroutine INIT (lines 1-4), stores the text of the TITLE statement and increments the pass counter to 2. A variable NSP is set equal to 72 less the number of characters in the TITLE statement (for use in the package during run-time).

The PARAMETER and UPDATE statements match to macros which, during pass 2, parse the statements using context controlled iteration and store the variable names and numerical values of the assignments for subsequent output. If CONSTANT or INCON statements had been present these statements would have been processed in a similar way.

The INITIAL macro generates the next section of output for subroutine INIT (lines 5-25) including the specification statements and common blocks associated with the simulation package and the DIMENSION, DOUBLE PRECISION, EQUIVALENCE and COMMON statements from the UPDATE statement (lines 10,13,15 and 23). If more than one variable were present in the UPDATE statement these lines would be modified accordingly, but if more than one UPDATE statement were included multiple DIMENSION, DOUBLE PRECISION, EQUIVALENCE and COMMON statements would be generated, one for each UPDATE statement.

The ASK macro stores the name of the variable (A) specified in the ASK statement and outputs the COMMON block for this variable (line 24). The macro %=% matches to the statement B=0. and stores the name of the variable on the left hand side (B) for output in a COMMON block by the DYNAMIC macro (line 25). This COMMON block is output subsequently in subroutine MODEL and thus the values calculated in the INIT routine are passed to the MODEL routine.

A flag, INIT, is set to 1 in the INITIAL macro to indicate that, when the DYNAMIC macro is entered, an INITIAL statement has been encountered. If there were no INITIAL segment in the input file, the DYNAMIC macro would generate an INITIAL statement which effectively calls the INITIAL macro.

Although there is no further output during pass 2, additional processing takes place. The assignment statements in the DYNAMIC segment match to the %=% macro which, during this pass, stores the names of the variables on the left hand side of the equations for subsequent use and as each is stored increments a counter NV. The %=%INTGRL(%,%) macro stores the names of the variables on the left hand side of the INTGRL statements and also the initial condition string, parameter 3 in the INTGRL statement. A counter N is incremented each time the macro is matched to count the number of INTGRL statements. The END macro fills out the array of variables (DAT) with dummy variables and the array of variable names (HEADER) with spaces and rewinds the input file. Any other macros which may be matched during this pass are set to take no action.

During pass 3, the PARAMETER macro outputs a COMMON block for the names of the variables in the PARAMETER statement (line 26). The INITIAL macro outputs the FORTRAN DATA statements for (a) the literal array (HEADER) of the names of the variables on the left hand side of the assignment statements in the DYNAMIC segment stored in pass 2 by the %=% macro (line 27), (b) the literal array (TITLE) of the characters from the TITLE statement stored in pass 1 by the TITLE macro (line 28-30) and (c) the variables from the PARAMETER and UPDATE statements stored in pass 2 (lines 31-36). The READ and WRITE statements associated with the TITLE and UPDATE statements are also output by this macro (lines 37-48).

If no TITLE statement were present in the input file the default TITLE (SIMULATION PROGRAM) would be used at this point, thus lines 28-30, 37 and 38 are always output. However, if there were no UPDATE or PARAMETER statements in the input file, lines 31-36 and 39-48 would not appear. The ASK macro outputs the READ and WRITE statements for the ASK statement (lines 51-53). Multiple ASK statements would each generate blocks of READ, WRITE and FORMAT statements similar to those shown. The %=% macro outputs any assignment statements (line 56) for the INITIAL segment at this point.

Also during pass 3, the DYNAMIC macro outputs the FORMAT statements common to all the READ statements generated by the ASK statements in the INITIAL segment. Assignments for the system variables N, NV and NSP from the INTGRL, %=% and TITLE macros respectively, are output at this point (lines 62-64). The initial conditions for the INTGRL statements, stored during pass 2 are output as assignments to the array WW, from which they are passed to the integration routine (lines 66-67).

The RETURN and END statements (lines 69-70) for subroutine INIT are output by the DYNAMIC macro during this pass along with the start of subroutine MODEL. All the required specification statements and COMMON blocks associated with the PARAMETER, UPDATE and ASK statements, the assignment statements in the INITIAL segment and the remainder of the FORTRAN package (lines 71-96) are also output. COMMON/V/ is constructed from all the variable names on the left hand side of all the assignment and INTGRL statements stored in pass 2 and the dummy variables inserted by the END macro during pass 2. The %=%INTGRL(%,%) macro outputs the assignments to the integrated variable array (VAR) (lines 98-99) for the variable names on the left hand side of each INTGRL statement. There is no output by any of the other macros during this pass and the END macro again rewinds the input file.

As before, the TITLE macro increments the pass counter (to 4), but apart from setting flags the only macros to have any action during this pass are those matched by the assignment and INTGRL statements in the DYNAMIC segment. The assigments are output unchanged by the %=% macro and the INTGRL statements are translated and then output as assignments to the derivative array (DER) (lines 102-109). The END macro outputs the end of MODEL (lines 111-112) and terminates the STAGE2 translation process. In addition the END macro outputs the start of the run-time documentation file (not shown in Figure 3.12) as discussed in sections 3.2.4 and 4.2.3.

3.2.4 Run-time Documentation File

A facility has been included in the simulation language which allows the user to create a documentation file of a simulation run (section 4.2.3 and section 4.10 of the GUILDS User's Guide). This file contains all the data supplied by the user in the Model Description and details of the options selected by the user at run-time.

To produce this documentation file two FORTRAN subroutines are used. The first of these subroutines (RDOC1) is a fixed routine which calls the second routine (RDOC2) and then writes all the user options selected at run-time to the documentation file. The second routine writes the names and values of the variables in the data statements in the Model Description to the documentation file, along with the title of the simulation, the time, the date and the run number.

It was decided to use two separate subroutines to minimise the STAGE2 processing required to produce the FORTRAN code necessary to write the documentation files. The first routine handles Simulation Language variables which are known and therefore this routine can

remain unchanged from one model to the next. The second routine is dependent upon the user's Model Description and thus has to be generated by STAGE2 from the input file.

Initially, subroutine RDOC2 was produced entirely by the translation phase (section 3.2.2) but, in order to reduced the size of the STAGE2 macro file for the translator, it was decided to generate the subroutine in two stages. The translation phase (section 3.2.3) outputs the start of subroutine RDOC2 including all the COMMON, INTEGER, REAL and comment statements produced by translation of the Model Description. As all the necessary information for the model dependent section of the documentaion file is contained in these statements, a separate STAGE2 macro file can be used which completes the subroutine by inserting WRITE and FORMAT statements for all the user supplied variables.

An example of the FORTRAN subroutine RDOC2 is shown in Figure 3.13 and the corresponding documentation file is shown in Figure 3.14. This file corresponds to the simulation run given as an example in Appendix A of the GUILDS User's Guide and has been annotated to indicate those parts of the file produced by the fixed routine and those parts produced by the other routine. The STAGE2 macros files used to generate the documentation subroutine, RDOC2, are listed in Figure 3.15.

Macro Name	Function of Macro
% EQU %	Store actual parameter 2 at the address given by parameter 1.
% SET %	Store the value of parameter 2 evaluated as an arithmetic expression at the address given by parameter 1.
INCR %	Increment by 1 the value of parameter 1.
DCR %	Decrement the value of parameter 1 by 1.
SKIP %	Set the skip counter to the value of parameter 1.
IF % EQ % SKIP%	Set the skip counter to the value of parameter 3 if parameter strings 1 and 2 are identical.
IF % NE % SKIP%	Set the skip counter to the value of parameter 3 if parameter strings 1 and 2 are not identical.
IF %=% SKIP%	Set the skip counter to the value of parameter 3 if values of parameters 1 and 2 are equal.
IF %<>% SKIP%	Set the skip counter to the value of parameter 3 if values of parameters 1 and 2 are not equal.
IF %<% SKIP%	Set the skip counter to the value of parameter 3 if the value of parameter 1 is less than the value of parameter 2.
IF %>% SKIP%	Set the skip counter to the value of parameter 3 if the value of parameter 1 is greater than the value of parameter 2.

Figure 3.1 - STAGE2 Macros for Arithmetic and String Operations

Macro Name	Function of Macro
TITLE % INITIAL DYNAMIC TERMINAL SORT NOSORT *%	Output line unchanged.
END	Output statement unchanged and terminate processing.
MACRO %=%[%]	Set flag MAC to indicate a Macro Definition has been encountered; set flag MAC%22 to indicate that the Macro %20 has been defined; store the position of each parameter in the Macro Definition.
ENDMAC	Clear flag MAC.
%=%[%]	Check flag MAC%22 and if not set signal error and terminate processing. If set, save the actual parameters in the Macro Call and retrieve the lines of the Macro Code Body. As each line is retrieved it is passed to the STAGE2 macro FPAR % which replaces the special characters in the line, as described in the text.

%<tab>CONTINUE
GOTO %
GOTO %, (%)
GOTO (%), %
DO % %=%
IF(%) %
IF(%) %=%
IF(%) GOTO %
READ%
WRITE%
CALL %
CALL %(%)
%=%
PROCEDURE %=%(%)
ENDPROC
	If match occurs within a Macro Code Body store line. The line is stored, as described in the text, with statement labels, formal parameters and local variables indicated by special characters. Any of the parameters in these macros which match to expressions are passed to the STAGE2 macro PARSE % which uses a context controlled iteration to detect occurances of formal parameters, local variables and statement labels. Otherwise the line is output unchanged.

Figure 3.2 - STAGE2 Macros for Expanding User Defined Macros
(Part 1 of 2)

Macro Name	Function of Macro
% FORMAT%	If match occurs in a Macro Code Body signal an error and terminate processing. Otherwise output statement unchanged.
<sp>% <tab>%	Remove leading tabs and spaces and output line for further matching.
%<sp>% %<tab><sp>%	Where parameter 1 is a statement label (FNUM %) remove tabs and spaces between parameters, insert a tab and output line for further matching. Otherwise output line unchanged.
%<tab>%	If match occurs within a Macro Code Body output %10<tab>CONTINUE and %20 as separate lines for further matching. Otherwise output line unchanged.

Figure 3.2 - STAGE2 Macros for Expanding User Defined Macros
(Part 2 of 2)

Macro Name	Function of Macro
PARSE %	Parses parameter 1 (part of a Code Body Line) replacing each occurance of a formal parameter with 'n', where n is the position of the formal parameter in the Macro Definition, and replacing any local variables with "VAR" where VAR is the variable name. The STAGE2 macro FNUM % is used to determine if a string is numeric.
FPAR %	Parses parameter 1 (a code body line) and replaces 'n' with the n'th parameter in the Macro Call; replaces "VAR" with a generated variable unique within the output file; replaces ^xxx^ with a line number, also unique within the output file. The STAGE2 macro GPAR is called to get the actual parameter in the macro call corresponding to n. The macro FINT % is called to determine if VAR is integer or real so that a generated variable of the correct type is used. The macro DUMMY %,% is called to create a unique number for the generated variables.
FNUM %	Set the flag FNM if parameter string 1 is numeric.
FINT %	Set the flag FIN if the first character of parameter string 1 is one of I,J,K,L,M or N.
GPAR %	Get the actual parameter in the Macro Call in the position given by parameter 1.
DUMMY %,%	If parameter string 2 (VAR) has not been encountered in the current macro expansion then increment the counter, parameter string 1, store the value and return this value in a variable DUMNO. If the variable (VAR) has already been used then return the value stored on the first occurance.

Figure 3.3 - Internal STAGE2 Macros for Handling User Defined Macros

Macro Name	Functions of Macro Common to All Passes	Functions of Macro for First Pass	Functions of Macro for Second Pass
TITLE %	Initialise flags and increment run counter.	Output line to Channel 2.	Output line to Channel 3.
INITIAL	Set flag I	Output line to Channel 2.	Output line to Channel 3
DYNAMIC	Clear flag I and set flags SRT and D.	Output line to Channel 2.	Output line to Channel 3,
SORT	Set flag SRT.	Output line to Channel 2.	
NOSORT	Clear flag SRT.	Output line to Channel 2.	
TERMINAL	Clear flags D and SRT and set flag TER.	Output line to Channel 2.	
PROCEDURE %=%(%)	Set flags PROC%22 and PROC and output %10=%20(%30) to Channel 2.		
ENDPROC	Clear flag PROC		
END	Call REWIND	Output line to Channel 2.	If statements stored call UNDEF. Output line to Channel 3 and terminate processing.
% %=%	Where parameter 1 is a statement label (FNUM %) remove sqace, insert a tab and output line for further matching.	Output line to Channel 2.	Output line to Channel 3, set flag P and output %=% for further matching.
ASK %/% IASK %/%		Output line to Channel 2.	Output line to Channel 3 and call DEFSTR %20.
*% TERMINATE (%)		Output line to Channel 2.	Output line to Channel 3.
%=%INTGRL% %=%FOLAG% %=%CMPXPL%		Output line to Channel 2. If %20 is null (i.e. not part of an expression) call DEFSTR %20.	If SRT flag is set call SORT %10=%20INTGRL%30, if not call DEFSTR %10 and output line to Channel 3.
%=%		If PROC flag is set store line. Otherwise, output line to Channel 2.	If P flag is set, clear flag, parse line and define all variables. If I flag is set, define %10 and output line to Channel 3. If D flag and SRT flag are set call SORT %10=%20; if SRT is not set output line to Channel 3 and call DEFSTR %10.

Figure 3.4 - STAGE2 Macros for Sorting Model Description
(Part 1 of 2)

Macro Name	Functions of Macro Common to All Passes	Functions of Macro for First Pass	Functions of Macro for Second Pass
%<tab>CONTINUE GOTO % DO % IF(%) % % FORMAT READ% WRITE% CALL% % $%$		If PROC flag set, store line; if SRT flag set, call the ERROR macro. Otherwise, output line to Channel 2.	Output line to Channel 3.
IF(%) %=%		If PROC flag set, store line; if SRT flag set, call the ERROR macro. Otherwise, output line to Channel 2.	Output line to Channel 3 and call DEFSTR %20.
<sp>% <tab>%	Remove leading tabs and spaces and output line for further matching		
%<sp>%	Where parameter 1 is a statement label (number) remove space, replace with a tab and output line for further matching.	Output line to Channel 2	Output line to Channel 3
%<tab><sp>%	Remove space and output line for further matching.		
%<tab>%	Output %10 CONTINUE and %20 as separate statements for further matching.		

Figure 3.4 - STAGE2 Macros for Sorting Model Description
(Part 2 of 2)

Macro Name	Function of Macro
SORT %=%	If all the variables in parameter string 2 are defined, call DEFSTR %10 and output %10=%20 to Channel 3, unless a Procedure name is recognised; if there are any undefined statements stored call REPASS. If an undefined variable is found store parameters 1 and 2. The macro FSTR is called to search parameter string 2 for undefined variables and the macro FPROC is called to check parameter string 2 for a Procedure name and output the Procedure Statements.
REPASS	Check each stored statement, as in SORT above, and output statement if now defined.
FSTR %	Parse parameter string 1 and if all the variables in the expression are defined set flag FST. The macro FNUM is called to establish if any operand is numerical. If the flag UDF is set any undefined variable encountered is output to Channel 4 with an error message.
FPROC %	Parse parameter string 1 and if a Procedure name is recognised set the flag FPRC and output the statements stored when the Procedure Definition was encountered.
DEFSTR %	Parse parameter string 1 and add any variables enountered to list of defined variables.
FNUM %	Set the flag FNM if parameter string 1 is numeric.
UNDEF %	Set flag UDF, output stored statement to Channel 3 and to Channel 4 with an error message. Call FSTR to find first undefined variable in statement.
ERROR	Output error message and terminate processing.
REWIND	Set Channel 2 as input, Channel 3 as output and rewind Channel 2.

Figure 3.5 - Internal STAGE2 Macros Used by Sorting Algorithm.

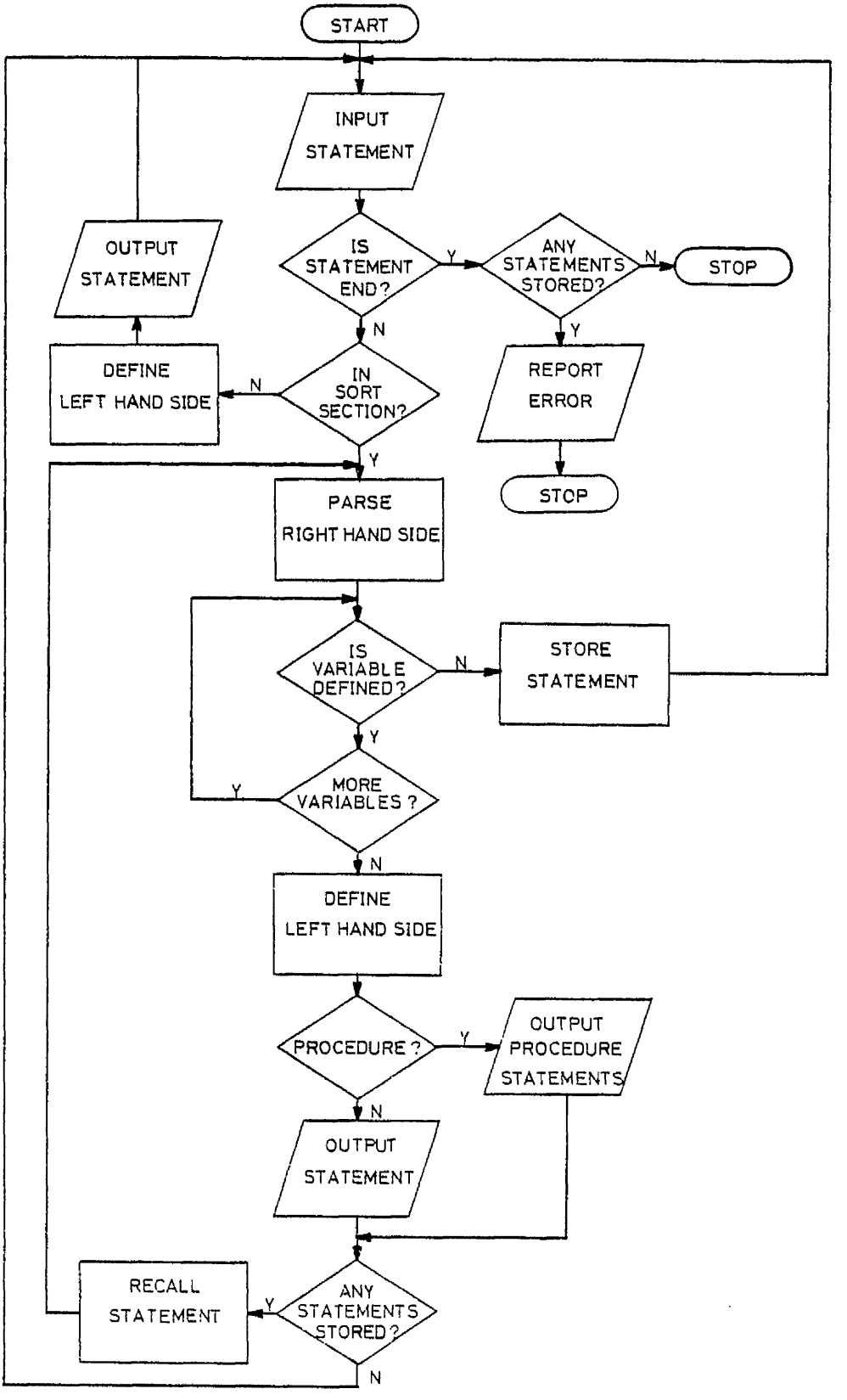


Figure 3.6 - Operation of Sorting Algorithm

Macro Name	Functions of Macro Common to All Passes	Functions of Macro for First Pass	Functions of Macro for Second Pass	Functions of Macro for Third Pass	Functions of Macro for Fourth Pass
TITLE %	Initialise variables Increment run counter	Copy I/P to Ch. 2 and rewind. STORE TITLE text as Hollerith. O/P start of INIT			
INITIAL	Set and clear flags	Generate TITLE statement if none present	PRINT user defined & system DIMENSION, DOUBLE PRECISION, INTGER, LOGICAL, REAL & COMMON statements and O/P DIMENSION, DOUBLE PRECISION, EQUIVALENCE & COMMON from UPDATE for INIT	O/P DATA statements from TITLE, UPDATE & PARAMETER and WRITE statements from TITLE & UPDATE for INIT	
DYNAMIC	Set and clear flags Call INITIAL if not already encountered	Generate TITLE statement if none present	O/P COMMON from %=% for INIT	O/P FORMAT from ASK, assignments for N & NV and IC's from integrators for INIT. O/P end of INIT and start of MODEL. PRINT specification and COMMON statements for MODEL as in INIT	Set flag
TERMINAL		Call END	Call END	Call END	O/P end of MODEL and start of TERM. PRINT specification and COMMON statements for TERM as in INIT
END	Call REWIND		Calculate NV. Fill out COMMON/V/ with dummy variables and COMMON/HEADER/ with blanks		O/P end of TERM and part of RDOC2 and terminate processing
PARAMETER %=%,%	Set flags	Generate TITLE statement if none present.	STORE LHS and RHS of assignments for DATA and COMMON	O/P COMMON of LHS for INIT	
CONSTANT % INCON % PARAMETER %=% PARAMETER %=%,... PARAMETER %=%,%,...	Call macro PARAMETER %=%,% Set flags				
%=%,%			STORE LHS and RHS of assignments for DATA and COMMON	O/P COMMON of LHS for INIT	
%=%,... %=%,%,...	Call macro %=%,%				
UPDATE %=%,% UPDATE %=%		Generate TITLE statement if none present	Store LHS and RHS of assignments for DATA COMMON & EQUIVALENCE		
ASK %/%			O/P COMMON from ASK for INIT	O/P R/W from ASK for INIT	
ASK PARAMETER/% IASK %/%	Call macro ASK %/% and set flags				

Figure 3.7 - STAGE2 Macros for Translating Model Description
(Part 1 of 2)

Macro Name	Functions of Macro Common to All Passes	Functions of Macro for First Pass	Functions of Macro for Second Pass	Functions of Macro for Third Pass	Functions of Macro for Fourth Pass
%=%	If part of a data statement, call %=%,% with parameter 3 as 0		STORE LHS, if not already encountered	O/P statement if for INIT	O/P statement if for MODEL or TERM
@%=% @%=%INTGRL%			Call %=% or %=%INTGRL(%,%)	Call %=% or %=%INTGRL(%,%)	
%=%INTGRL(%,%)			Increment N STORE LHS if not already encountered	O/P "VAR" assigments	O/P "DER" assignments
$%$ % % CONTINUE GOTO % IF(%) % DO % % FORMAT READ% WRITE% CALL %			O/P statement if for INIT		O/P statement if for MODEL or TERM
DIMENSION% DOUBLE PRECISION% INTGER% LOGICAL% REAL% COMMON%		Save user defined specification statements and COMMON blocks for O/P by PRINT macro			
COMMENT%		Save comment statements for O/P by PRINT macro			
*% *<sp>% *<tab>%	Call COMMENT% if before INITIAL else call $%$	Generate TITLE statement if none present			
<sp>% <tab>%	Strip leading spaces and tabs except in TERMINAL				
%<sp>% %<tab><sp>%	Call %<tab>%				
%<tab>%	Call %<tab>CONTINUE and output %20 for further matching				
%:% %=%:%	Separate statement and comment. Call *% and % or %=%				
%<sp>:% %<tab>:% %=%<sp>:% %=%<tab><sp>:%	Remove spaces and tabs. Call %:% or %=%:%				
REWIND (internal macro)	Rewind Ch. 2				
STORE% (%) %,%,% (internal macro)	Pack param. string 2 into list specified by other parameters				
PRINT %,% (internal macro)	Output stored statements as specified by parameters 1 & 2				

Figure 3.7 - STAGE2 Macros for Translating Model Description
(Part 2 of 2)

Macro Name	Functions of Macro Common to All Passes	Functions of Macro for First Pass	Functions of Macro for Second Pass	Functions of Macro for Third Pass	Functions of Macro for Fourth Pass
TERMINATE (%)					O/P IF statement
%=%FOLAG(%,%,%) %=%LEADLG(%,%,%,%) %=%DERLAG(%,%,%,%) %=%CMPXPL(%,%,%,%,%) %,%=PIPE(%,%,%,%,%)	Expanded form of function output for matching by %=% and %=%INTGRL(%,%) macros				
%=%LIMIT(%,%,%)		STORE LHS if not already encountered			O/P statement and limit output of integrator if RHS is an integrated variable
%=%DELAY(%,%) %=%ANDHYS(%,%,%) %=%HSTRSS(%,%,%,%) %=%RAMP2(%,%,%,%) %=%RATLIM(%,%,%,%)		STORE LHS if not already encountered			Insert function call number for dimension of storage array and O/P statement

Figure 3.8 - STAGE2 Macros for Translating GUILDS Functions

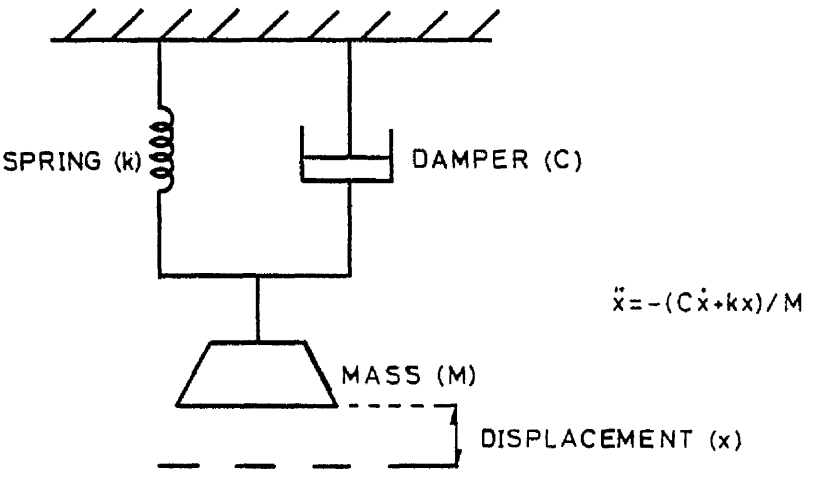


Figure 3.9 - Mass, Spring and Damper System

```
TITLE MASS, SPRING AND DAMPER SYSTEM
*EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS
*
CONSTANT C=1.,AK=2.
*
UPDATE AM=1.
*
INITIAL
*
ASK INITIAL DISPLACEMENT OF MASS/A
*
B=0.
*
DYNAMIC
*
  X2DOT=-(Y1+Y2)/AM      :ACCELERATION
  Y1=C*XDOT
  Y2=AK*X
  XDOT=INTGRL(B,X2DOT)   :VELOCITY
  X=INTGRL(A,XDOT)       :POSITION
*
END
```

Figure 3.10 - Example of a Model Description

```
TITLE MASS, SPRING AND DAMPER SYSTEM
*EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS
*
      CONSTANT C=1.,AK=2.
*
      UPDATE AM=1.
*
INITIAL
*
      ASK INITIAL DISPLACEMENT OF MASS/A
*
      B=0.
*
DYNAMIC
*
      Y1=C*XDOT
      Y2=AK*X
      X2DOT=-(Y1+Y2)/AM        :ACCELERATION
      XDOT=INTGRL(B,X2DOT)     :VELOCITY
      X=INTGRL(A,XDOT)         :POSITION
*
END
```

Figure 3.11 - Model Description After Sorting

LINE NO.	FORTRAN PRODUCED BY TRANSLATOR:	EQUIVALENT SOURCE LINE:	MACRO PRODUCING OUTPUT:	PASS NUMBER:
1	C MASS, SPRING AND DAMPER SYSTEM	TITLE...		
2	C	None		
3	SUBROUTINE INIT	None	TITLE %	1
4	C	None		
5	C EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS	Comment		
6	C	Comment		
7	C	Comment		
8	C	Comment		
9	DIMENSION W(30),WW(30)	None		
10	DIMENSION VAL1(1)	UPDATE...		
11	DOUBLE PRECISION HEADER(72)	None		
12	DOUBLE PRECISION TIME,H,PLT,FINTIM,STR,PRNT	None		
13	DOUBLE PRECISION ANAM1(1)	UPDATE...		
14	LOGICAL*1 TITLE(72)	None	INITIAL	
15	EQUIVALENCE (VAL1(1),AM)	UPDATE...		
16	COMMON/C/LY	None		2
17	COMMON/D/ITIN,ITOUT	None		
18	COMMON/H/TIME,H,PLT,FINTIM,STR,PRNT	None		
19	COMMON/L/HEADER	None		
20	COMMON/I/N,NV,NEW,LT	None		
21	COMMON/R/W,WW	None		
22	COMMON/T/TITLE,NSP	None		
23	COMMON/UD1/AM	UPDATE...		
24	COMMON/ASK1/A	ASK...	ASK %/%	
25	COMMON/INIT1/B	B=0.	DYNAMIC	
26	COMMON/BLK1/C,AK	PARAMETER...	PARAMETER %=%,%	
27	DATA HEADER/ 'Y1','Y2','X2DOT','XDOT','X',67*' '/	Dynamic statements		
28	DATA TITLE/ 'M','A','S','S',' ',' ',' ','S','P','R','I','N','G',	TITLE...		
29	1 ' ','A','N','D',' ','D','A','M','P','E','R',' ',			
30	1 'S','Y','S','T','E','M',42*' '/			
31	DATA C,AK/	PARAMETER...		
32	1 1.,2./			
33	DATA ANAM1/'AM'/	UPDATE...		
34	C	None		
35	DATA VAL1/1./	UPDATE...		
36	C	None		
37	WRITE(ITOUT,1003) TITLE	TITLE...		3
38	1003 FORMAT(1X,/,1X,72A1,/)	TITLE...	INITIAL	
39	WRITE(ITOUT,1005)	None		
40	1005 FORMAT(1X,'DO YOU WISH TO UPDATE ANY VARIABLES? ',$)	None		
41	READ(ITIN,1010) LUD	None		
42	1010 FORMAT(A1)	None		
43	IF(LUD.NE.LY) GOTO 1250	None		
44	C	None		
45	WRITE(ITOUT,1020)	None		
46	1020 FORMAT(1X,'VARIABLES FOR UPDATING: AM')	UPDATE...		
47	CALL UPDATE(VAL1,ANAM1,1)	UPDATE...		
48	1250 CONTINUE	None		
49	C	Comment	*%	
50	C	None		
51	1303 WRITE(ITOUT,1305)	ASK...		
52	READ(ITIN,1500,ERR=1303)A	ASK...	ASK %/%	
53	1305 FORMAT(1X,'INITIAL DISPLACEMENT OF MASS')	ASK...		
54	C	None		
55	C	Comment	*%	
56	B=0.	B=0.	%=%	
57	C	Comment	*%	
58	C	None		3
59	1500 FORMAT(10E13.6)	None		
60	1510 FORMAT(10I7)	None		
61	C	None		
62	N=2	INTGRL Statements		
63	NV=5	Dynamic Statements	DYNAMIC	
64	NSP=42	TITLE		
65	C	None		
66	WW(1)=B	XDOT=INTGRL...		
67	WW(2)=A	X=INTGRL...		
68	C	None		
69	RETURN	None		
70	END	None		

Figure 3.12 - FORTRAN Subroutines from Translation of Model Description (Part 1 of 2)

LINE NO.	FORTRAN PRODUCED BY TRANSLATOR:	EQUIVALENT SOURCE LINE:	MACRO PRODUCING OUTPUT:	PASS NUMBER:
71	C	None		
72	SUBROUTINE MODEL(VAR,DER,T,M)	None		
73	C	None		
74	C EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS	Comment		
75	C	Comment		
76	C	Comment		
77	C	Comment		
78	DOUBLE PRECISION TIME,H,PLT,FINTIM,STR,PRNT	None		
79	DIMENSION VAR(30),DER(30),DAT(72),W(30)	None		
80	REAL LIMIT	None		
81	COMMON/V/Y1,Y2,X2DOT,XDOT,X,ZZ0,ZZ1,ZZ2,	Dynamic Statements		
82	1 ZZ3,ZZ4,ZZ5,ZZ6,ZZ7,ZZ8,ZZ9,ZZ10,			
83	1 ZZ11,ZZ12,ZZ13,ZZ14,ZZ15,ZZ16,ZZ17,ZZ18,			
84	1 ZZ19,ZZ20,ZZ21,ZZ22,ZZ23,ZZ24,ZZ25,ZZ26,			
85	1 ZZ27,ZZ28,ZZ29,ZZ30,ZZ31,ZZ32,ZZ33,ZZ34,		DYNAMIC	
86	1 ZZ35,ZZ36,ZZ37,ZZ38,ZZ39,ZZ40,ZZ41,ZZ42,			
87	1 ZZ43,ZZ44,ZZ45,ZZ46,ZZ47,ZZ48,ZZ49,ZZ50,			3
88	1 ZZ51,ZZ52,ZZ53,ZZ54,ZZ55,ZZ56,ZZ57,ZZ58,			
89	1 ZZ59,ZZ60,ZZ61,ZZ62,ZZ63,ZZ64,ZZ65,ZZ66			
90	COMMON/D/ITIN,ITOUT	None		
91	COMMON/H/TIME,H,PLT,FINTIM,STR,PRNT	None		
92	COMMON/R/W	None		
93	COMMON/BLK1/C,AK	PARAMETER...		
94	COMMON/UD1/AM	UPDATE...		
95	COMMON/INIT1/B	B=0.		
96	COMMON/ASK1/A	ASK...		
97	C	None		
98	XDOT=VAR(1)	XDOT=INTGRL...	%=%INTGRL(%,%)	
99	X=VAR(2)	X=INTGRL...		
100	C	None	DYNAMIC	
101	C	Comment	*%	
102	Y1=C*XDOT	Y1=C*XDOT	%=%	
103	Y2=AK*X	Y2=AK*X	%=%	
104	C ACCELERATION	X2DOT=...	%=% :%	
105	X2DOT=-(Y1+Y2)/AM	X2DOT=...	%=%	
106	C VELOCITY	XDOT=INTGRL...	%=% :%	4
107	DER(1)=X2DOT	XDOT=INTGRL...	%=%INTGRL(%,%)	
108	C POSITION	X=INTGRL...	%=% :%	
109	DER(2)=XDOT	X=INTGRL...	%=%INTGRL(%,%)	
110	C	Comment	*%	
111	RETURN	None	END	
112	END	None		

Figure 3.12 - FORTRAN Subroutines from Translation of Model Description (Part 2 of 2)

```
C       MASS, SPRING AND DAMPER SYSTEM
C
        SUBROUTINE RDOC2
C
        LOGICAL*1 ZDATE(9),ZTIME(8)
C
C
C
C
        COMMON/BLK1/C,AK
        COMMON/UD1/AM
        COMMON/ASK1/A
        COMMON/INIT1/B
        COMMON/E/KRUN,KRR,LR
C
        WRITE(4,50)
50      FORMAT(/,
     1  /,1X,'EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS',/)
C
        CALL DATE(ZDATE)
        CALL TIME(ZTIME)
        WRITE(4,60)ZDATE,ZTIME,KRUN
60      FORMAT(1X,'DATE ',9A1,6X,'TIME ',8A1,6X,'RUN NUMBER ',I4,/)
C
        WRITE(4,100)
100     FORMAT(/,1X,'PARAMETERS, CONSTANTS AND INITIAL CONDITIONS',/)
        WRITE(4,105) C,AK
105     FORMAT(/,T2,'C = ',G,T25,'AK = ',G,/)
        WRITE(4,110)
110     FORMAT(/,1X,'UPDATE VARIABLES',/)
        WRITE(4,115) AM
115     FORMAT(/,T2,'AM = ',G,/)
        WRITE(4,120)
120     FORMAT(/,1X,'ASK VARIABLES',/)
        WRITE(4,125) A
125     FORMAT(/,T2,'A = ',G,/)
C
        RETURN
        END
```

Figure 3.13 - Example of Documentation Subroutine RDOC2

```
EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS

DATE 17-SEP-80      TIME 13:42:14      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
C =   0.5000000        AK =   0.5000000

UPDATE VARIABLES
AM =    2.000000

ASK VARIABLES
A =  -0.5000000
```

Output by RDOC2

```
INTEGRATION METHOD  EULER
INTEGRATION INTERVAL  0.100000E+00
FINISH TIME  0.100000E+02

PRINTING SELECTED
PRINT INTERVAL  0.100000E+01
PRINT VARIABLES:-

  TIME       X          XDOT

PLOTTING SELECTED
PLOT VARIABLES:-

  TIME       X

STORAGE SELECTED
FILENAME TEST.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED
POST-RUN PASS NUMBER    1

PLOTTING SELECTED
PLOT VARIABLES:-

  TIME       X          XDOT       X2DOT

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME TEST.PLT
STORED VARIABLES:-

  TIME       X          XDOT       X2DOT

STORAGE FOR FURTHER PRINTING SELECTED
FILENAME TEST.SPL
STORED VARIABLES:-

  TIME       X          XDOT       X2DOT
```

Output by Fixed Routine

Figure 3.14 - Example of a Run-time Documentation File

Macro Name	Function of Macro
%	Strip leading tabs
C %	Output start of RDOC2 to Channel 3 if matched for the first time. Otherwise output line to Channel 3.
%	Output line to Channel 3.
DOUBLE PRECISION % INTEGER % REAL %	Output line to Channel 3, parse parameter 1 for variables and store type at LIST%12
COMMON/BLK%/% COMMON/UD%/% COMMON/ASK%/%	Output line to Channel 3 and save parameter 2
RETURN	Output some more of RDOC2 to Channel 3 followed by WRITE and FORMAT statements for the variables in COMMON blocks above. The FIELD macro is called to construct the FORMAT statement.
FIELD %	Constructs one or more lines of FORMAT statement corresponding to the variable list in parameter string 1. The type of each variable is checked so that the correct FORMAT is used. The FINT macro is used to check for integer variables.
FINT %	Set a flag if the first character of parameter string 1 is one of I,J,K,L,M or N.
END	Output end of RDOC2 and terminate processing.

Figure 3.15 - STAGE2 Macros for Run-time Documentation File

CHAPTER 4

FEATURES OF THE SIMULATION LANGUAGE

4.0 Introduction

The subroutines produced by the Simulation Language translator, as discussed in Chapter 3, are compiled and linked with the subroutines of the FORTRAN simulation package. The whole process of translation, compilation and linking (task building) is controlled by a Simulation Executive written using the PDP-11 Indirect Command File facility[47]. Sections 4.1 and 4.2 of the GUILDS User's Guide (which is to be found at the rear of this Thesis) describes the Simulation Executive in more detail and an example of the use of the Executive is given.

Although the Simulation Language is based on the package described in Reference 17, the FORTRAN subroutines have been extensively modified and a number of new routines added to provide more facilities. The use of these facilities is described in the User's Guide, but details of the structure of the FORTRAN program and of the implementation of some of the new features are given in this Chapter.

4.1 The FORTRAN Program

The basic structure of the FORTRAN program is shown in the flow chart given in Figure 4.1 and, in addition, a brief description of each FORTRAN subroutine is given in Figure 4.2, along with a list of the other subroutines called by each routine. The main program, FIRST, consists primarily of logical branching and subroutine calls controlled by variables set up by the user through an interactive dialogue, DIALOG. Not all the facilities selected by the user using this dialogue are shown in Figure 4.1, but the interaction of the different sections of the package can be clearly seen.

Initially, both INIT and DIALOG are called, although the Command String Interpreter (see section 4.2.1) can be selected as an alternative to DIALOG; for subsequent runs, either or both of these subroutines can be omitted. When the automatic rerun facility is selected, INIT is called, but only part of the subroutine is executed, DIALOG is omitted and subroutine TERM can be used to control the simulation runs. All the input/output statements in INIT associated with ASK and UPDATE statements in the Model Description are omitted and, in addition, the user can cause sections of the INITIAL segment of the Model Description to be by-passed when automatic rerun is selected (see section 4.8 of the GUILDS User's Guide).

The user can select one of several different integration techniques (see section 4.3 of the User's Guide): if a fixed step method is selected METHS is called; if a variable step method is selected VMETHS is called; if an integration routine supplied by the user is selected EXTERN is called. The integration routine is responsible for printed, plotted and/or stored output, as selected by the user. If stored output is selected the post-run output section (see section 4.2.2) can be used for further output from the stored data.

The principle section of the package is the integration routine which calculates the values of the integrated variables over the range of the independent variable, usually time, by the integration technique selected by the user. The values of the integrated variables are passed to subroutine MODEL where the corresponding derivatives and other associated variables are re-calculated.

The operation of the integration routine, with, as an example, plotted output selected, is shown in more detail in the flow chart given in Figure 4.3. This Figure represents the case for a first order integration routine, such as Euler; higher order routines, such

as Runge Kutta (fourth order), call MODEL and calculate intermediate values for the integrated variables several times within the integration interval.

The simulation time (TIME) and the plot time (PLTT) are set to zero initially and then subroutine MODEL is called to calculate the initial values of the derivatives and other variables, using the initial conditions for the integrated variables and the values of parameters supplied by the user through, for example, data statements or calculations in INIT. These values are used to calculate the values of the integrated variables at time H, where H is the integration interval, and then MODEL is called again to recalculate the values of the derivatives and other variables.

This cycle is repeated at each integration interval until the finish time (FINTIM) is reached. At each integration interval, PLTT is checked and if greater than or equal to the plot interval (PLT) then subroutine PLOT is called and the current values of the plot variables selected by the user are output.

4.2 Additional Features

The facilities offered by the FORTRAN simulation package were adequate, if somewhat limited. With the addition of the STAGE2 pre-processor, and the effective upgrade to a simulation language, it was decided to improve the FORTRAN package to provide the language with a more powerful execution phase. This involved a certain amount of rationalisation of the existing routines, for example changing all the package variables to start with ZZ, IZ or LZ to avoid confusion with variables supplied by the user, and also a number of new routines had to be included to provide the additional features required.

Of the 18 subroutines (excluding INIT, MODEL, TERM and EXTERN) which form the basis of the package, six are almost totally unchanged, seven have been altered substantially and seven new routines have been added. The original package provided the user with about six functions, whereas GUILDS offers a choice from more than twice this number. The real-time facilities of RISP[17] have not been included in GUILDS as non-real-time operation under a multi-user operating system was envisaged. However, if a real-time facility was required, for operating in conjunction with external equipment, then the inclusion of the clock routines in GUILDS would be relatively straightforward.

The following sections describe some of the features which have been added to the simulation package and the implementation of these features. A number of other more minor modifications have been made to the structure of the package and to the existing subroutines to facilitate the inclusion of these novel features.

4.2.1 Command String Interpreter

As noted above the options available to the user at run-time are selected, in response to various questions, through an interactive dialogue. As the facilities available within the package increased in number this dialogue became somewhat lengthy and, in particular, if a simulation was being repeated a number of times with only a few changes required, then answering all the questions each time was very tedious. Thus, it was decided, as an alternative to the interactive dialogue, to included the option of using a "command mode".

A command string interpreter was written which processes the commands issused by the user and thus permits selective changes to be made without repeating all of the dialogue. Commands are available which permit the user to select or de-select any of the options in the package (e.g. printing or plotting) or change any of the system variables set up by the user (e.g. integration interval or finish

time). Many of the commands result in the user being asked questions, similar to those in the interactive dialogue, after a particular command has been given (e.g. the user is asked for a file name after selecting storage by the STORE command). All the commands are listed with a brief explanation in response to the HELP command (see Appendix D of the GUILDS User's Guide).

For the first simulation run, of a series of repeated runs, it is usually more convenient to use the interactive dialogue to set up the run and thereafter use the command mode to make any changes required.

4.2.2 Post-run Output

In the original simulation package data could be stored during a run for only a limited number of variables (usually six) by assigning these variables to an array DAT. A separate program could then be used to plot these variables after the end of the simulation run. Since these assignments to the array DAT were part of the MODEL subroutine it was necessary to edit, compile, task build and then rerun the simulation in order to store and then plot any variables other than those originally specified.

This limitation was a particular hindrance when developing new simulation models as the response of almost any variable might have been required. Thus, it was decided to extend the array DAT (to nominally 72) so that the values of the variables on the left hand sides of all the expressions in the DYNAMIC section of the Model Description (unless specifically protected) could be stored during the simulation run.

In addition to this, a post-run output routine was written to access this data which permits the user to print, plot or store, repeatably, the responses of any of the variables in the DYNAMIC section of the Model Description. The facility for additional storage

was included so that the original data, stored in a binary format, could be written in ASCII format for output to a printer, or written in a format suitable for use with an external plotting package.

The user controls the post-run output section of the simulation through an interactive dialogue similar to that used for selecting the run-time options. Alternatively, the user can select "command mode" and a command string interpreter (similar to that described in section 4.1.2) is used. The data file that is to be used can be specified (the default being the most recent file created) and for each option (printing, plotting and storage) the user specifies the variables to be output. For plotting, mimimum and maximum values for each variable can be entered or calculated values can be used and also, the plot start and stop times can be specified. For stored output the user has to supply names for the files to which the data are to be written.

4.2.3 Run-time Documentation

A facility has been included in the simulation language whereby a documentation file of a simulation run can be written, if requested by the user. This facility is particulaly useful when repeated runs of a simulation are being carried out so that any output produced can be associated with a file containing details of the conditions pertaining to that run. This removes from the user the necessity to record in detail the changes made from one run to the next and thus all the information necessary to repeat a particular run is contained in the Model Description, which may be common to a number of runs, and the Documentation File for that run.

Although some potential users[24] consider that documentation should not be an optional feature, it has been made optional here to prevent an excessive number of files being created, particularly during the development of a model.

The Documentation File is written by two FORTRAN subroutines. The first routine is the same for every simulation module, and thus is part of the FORTRAN package. This routine calls the second routine which is dependent on the user's Model Description (see section 3.2.4). The second subroutine writes the simulation title, the date, the time and the run number to the file and the names and values of all the variables in the data statements in the Model description are added. The first subroutine then writes information concerning all the user selected options for the simulation run and for analysis of stored data using the post-run output routine.

An example of a run-time documentation file is given in Figure 3.14 for the simulation run shown in Figure A5 of the GUILDS User's Guide. This listing has been annotated to show which parts of the file are written by the fixed routine and which parts are written by the routine produced by the translator from the user's Model Description.

4.2.4 Keyboard Interrupt

During a simulation run it is sometimes desirable to be able to interrupt the run and, without actually aborting the program, end the run or start from the beginning again. This is a particularly useful feature when results are being plotted during the run and, from these results, it is realised that the model is incorrect or, sufficient information has been obtained from the start of the run and the run can then be stopped. Such a facility has been included in the simulation language whereby, when a key is struck on a specified terminal, the simulation pauses and the user is offered the choice of several options, as listed below:-

(1) the simulation run can be continued unchanged as though it had not been interrupted;

(2) the simulation run can be started from the beginning again with the option of passing through the INITIAL segment and the interactive dialogue section;

(3) the simulation run can be terminated and control returned to the computer operating system;

(4) the simulation run can be terminated with control passing to the user as if the DYNAMIC section had been completed normally;

(5) a disturbance, if specified by the DISTRB function, can be applied. (The DISTRB function permits the user to specify the value of a variable (or variables) before and after the disturbance is applied (see Appendix C2 of the User's Guide).)

Interfacing a FORTRAN program with the PDP-11 interrupt structure, under a multiuser system is difficult, thus it was decided to investigate the possibility of interrupting the simulation in some other way. As version 3.1 of the RSX-11M operating system[25] on the PDP-11 computer only allows half duplex terminal operation, that is information cannot be read from a terminal while data is being output to the terminal, it was not possible to use the same terminal to interrupt the simulation as was being used for output by the simulation program. Also, since a FORTRAN READ statement is effectively a READ and WAIT it was not possible for the simulation program to use a READ statement to watch for a character from the interrupt terminal, while, for example, plotting to another device, since the simulation run stops until the READ statement is satisfied.

This latter problem was overcome by using a separate program, started running from the main program, which reads the keyboard of a terminal, specified by the user as the interrupt device, and communicates with the main program by the use of Global Event Flags[26]. Global Event Flags are flags provided by the operating system which can be set, read and cleared by any user program. Thus, when an interrupt is detected by the secondary program reading a character from the specified interrupt keyboard, a flag is set which is checked by the main program once every integration cycle, and, if the flag is set the simulation run is suspended. At this point the user replies with one of the options, listed above, the event flag is cleared and execution of the simulation continues as requested.

Version 3.2 of the operating system allows full duplex terminal operation to be selected, thus, it is possible to interrupt a simulation run using the keyboard of a terminal that is being used for output.

4.3 Problem Size and Timing Considerations

There are various factors, as discussed in the following sections, which affect the size of Model Description which can be translated and executed, and the time required for each of these phases. It is worth noting that the restrictions on problem size, and the timings, are very much dependent on the computer being used, in this case a PDP11/45. With the advent of more powerful mini-computers with faster processors and virtual memory capabilities (for example, VAX 11/780) the execution times could be significatly reduced and the limits on problem size almost totally removed.

4.3.1 STAGE2 Memory Size

STAGE2 uses memory, firstly to store all the simulation language macro definitions and secondly for all the variables, or addresses generated during the translation process. If the memory requirements exceed the memory available an ERROR IN FULL is signalled and processing is aborted. As noted previously, the STAGE2 processing has been divided into four separate phases so as to permit translation of as large a Model Description as possible within the memory size available.

The size of Model Description that can be dealt with by the Macro Expansion Processor is very much dependent on the number and size of the user defined Macros as the STAGE2 macro file is relatively small and the memory requirements for the rest of the Model Description are minimal. In practice, these limits have never been reached and it is unlikely that any Model Description, which is within the limits given in section 4.3.2, would require sufficient memory to cause any problems.

Similarly, the memory requirements for the Sorting Algorithm depend on the number of statements that need to be repositioned and, to some extent the position the statement appears in the Model Description. Again, in practice, the STAGE2 memory limits have not been reached while sorting a Model Description. If, however, this were the case, the user could overcome the problem by attempting to sort some of the statements by hand or, by the use of SORT and NOSRT statements, cause the Model Description to be sorted in sections by several passes of the Sorting Algorithm.

The STAGE2 memory limits have been encountered during the translation phase with a number of large Model Descriptions. By increasing the amount of the memory available to STAGE2, within the PDP-11 address limits, and rationalising, where possible the STAGE2

coding this problem has, at present been overcome. It is possible, however, that this limit will be reached again if the size of the Model Description is increased much beyond the program limits given in the following section.

4.3.2 Program Limits

At present, due to the specified array dimensions in the FORTRAN subroutines, the number of differential equations is limited to 30 and the number of assignment statements in the DYNAMIC segment is limited to 72. This latter limit is in fact a limit on the number of variables, on the left hand sides of assignment statements, that can be used for output or storage (i.e. the dimension of the array DAT). The @ symbol (see section 3.2.4 of the GUILDS User's Guide) can be used to prevent the inclusion of certain variables in the array DAT and thus, the number of statements in the Model Description can be increased.

The dimensions of the arrays in the FORTRAN routines could be increased, if required. However, the STAGE2 memory limits outlined in section 4.2.1 would eventually restrict the size of Model Description that could be used.

The number of calls of some of the GUILDS subroutines (those listed in Appendix C2.2 of the User's Guide) is limited to 10. These functions require memory of past values and arrays are used, the subscript for a particular call being supplied by the translator, which are dimensioned as 10 in each subroutine. This is a nominal figure which could be easily changed, if required, by modifying the DIMENSION statements in the FORTRAN package.

4.3.3 Task Image Size

With the addition of a large number of subroutines to the FORTRAN simulation package it was found that the "task image" size, the size of the linked run-time module, exceeded the memory address limits of the PDP-11.

To overcome this difficulty an overlay structure[27] was used to reduce the overall size of the task image. When using an overlay structure it is necessary to divide the subroutines into groups, or segments, such that only one segment, in addition to a "root" segment containing the main program, requires to be in memory at any one time, the remaining segments being kept on disk. When a subroutine is called which is not in the overlay segment that is in memory, then the overlay segment containing the subroutine is copied from disk into memory overwriting the previous overlay segment. Thus, for example, the subroutines INIT and MODEL can be in separate segments as these subroutines are not called in the same section of the program.

The overlay segments used for the simulation language are such that it is unlikely that the task image size would exceed the memory address limits for a Model Description that is within the limits described in the previous sections.

4.3.4 Timing

Some typical figures for the time taken for the translation and execution of a simulation language Model Description are given below. These times are largely dependent upon the size of the Model Description but additional factors, for example other computer users, disk access time, type of compiler, can have a significant effect. The results given are for a Model Description having 72 assignments statements (including 24 differential equations) in the DYNAMIC segment, 15 assignments statements in the INITIAL segment and 30 other statements, mainly logical, data, type and translation control statements and comments. All times were recorded with no other users on the computer; version 3.1 of the RSX-11M operating system[25] was used with version 3.2 of FORTRAN IV[28]; storage was on an DD1180 disk. The figures given for Macro Expansion and Sorting are estimates as the Model Description did not require the use of either of these phases.

Macro Expansion	1 minute
Sorting	1.5 minutes
Translation	3.5 minutes
Compilation	30 seconds
Task-building	3.3 minutes
Execution (1)	1.3 minutes
Execution (2)	2.2 minutes
Execution (3)	3.5 minutes

The execution times are all for finish times of 50s using the Runge-Kutta fourth order integration technique with an integration interval of 0.05s. In the first case there is no output, in the second data is only stored whereas in the third case printed and plotted output are also selected. The first case would be appropriate for real-time operation with external equipment and, although this facility does not exist at present the timing has been included for completeness.

4.4 Discussion

In Chapters 2 and 3 and the current Chapter of this Thesis the need for, and the development of, a Continuous System Simulation Language have been presented. The use of the language has been illustrated by some examples, but the power of this design and development tool can be seen more clearly from the use of GUILDS in the simulation studies discussed in the subsequent Chapters of this Thesis.

To assess the value of GUILDS compared to other CSSLs a table (Figure 4.4) has been reproduced from Reference 24 in which the author lists some of the most important features of several languages. The original table has been amended by including the features of GUILDS in the first column (in place of a language which the author of

Reference 24 considered to be of little significance as the originator could not be traced) to enable a comparison to be made between GUILDS and other commercially avaliable simulation languages. Further details of these languages DARE-P, ISIS and ASCL can be found in References 29, 30, 31 and 32.

It can be seen from Figure 4.4 that the facilities offered by GUILDS compare well with those offered by the other languages. In particular ASCL, which is the closest language to GUILDS is superior in only a few of the features listed and GUILDS is superior to ASCL in some features. GUILDS offers only 5 integration routines, although user supplied routines can be included, but does not provide for stiff systems or vector integration. The user interaction and output facilities of GUILDS are better in some respects to those of ASCL, in particular graphical output during a simulation run is more readily available under GUILDS. The maximum problem size that can be handled by GUILDS is less than that of ASCL, but, it should be noted that ASCL was primarily a mainframe language although implementations for minicomputers are now common.

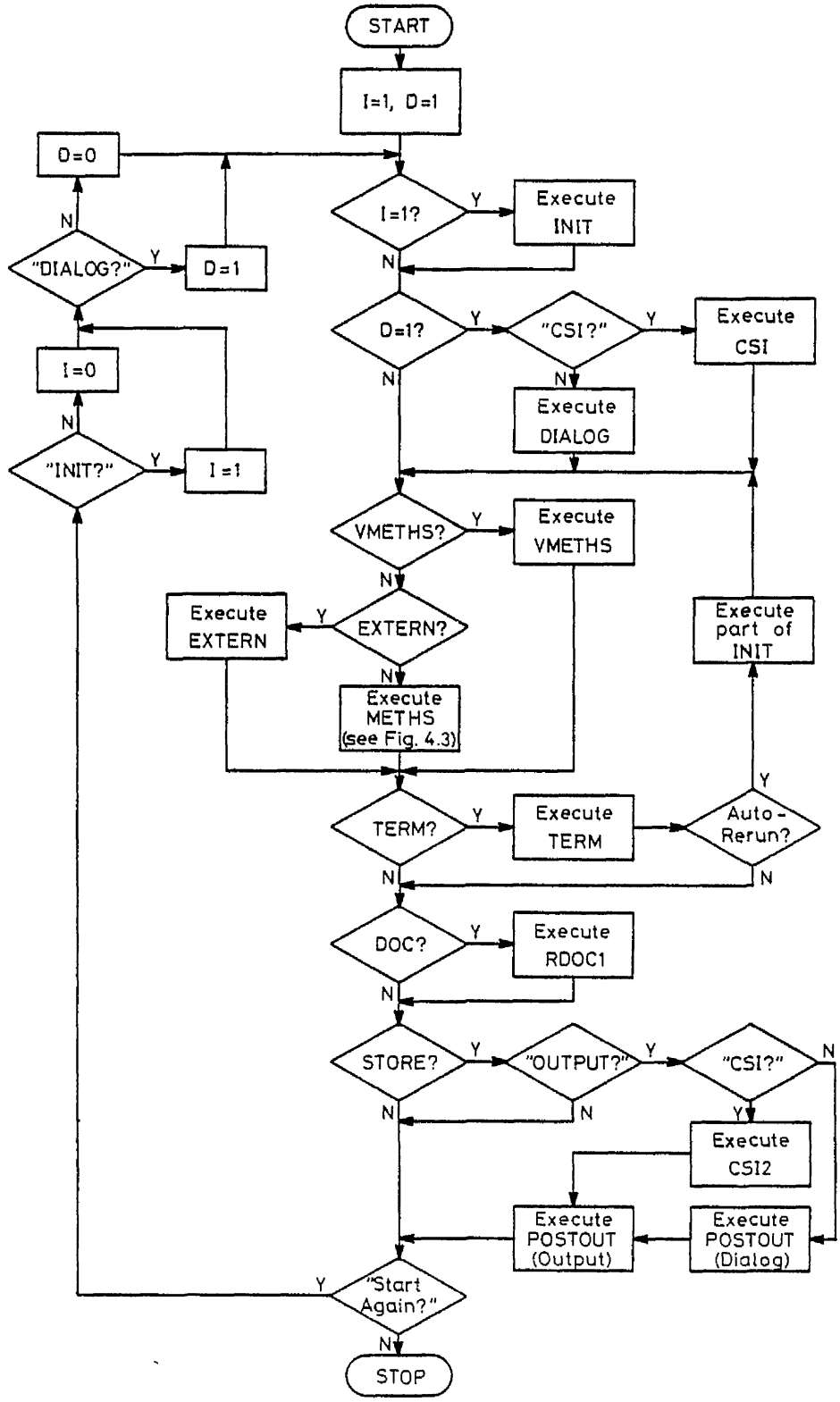
The other area where GUILDS does not compare well with ASCL and also the CSSL specification [23] is in error handling. The error checking facilities in GUILDS are limited and at present checks are made for undefined Macros, FORTRAN statements in SORT sections, undefined variables and array dimensions exceeded. GUILDS relies on the FORTRAN compiler for syntax checking and detection of other errors, although there is a certain amount of syntax checking in the STAGE2 template matching process.

For a user familiar with FORTRAN programming, finding errors detected by the compiler is reasonably straightforward but for other users the inclusion of debug and traceback facilites may prove useful.

Further work could also be done to extend the Keyboard Interrupt facility such the values of some of the variables of the simulation model could be changed during the simulation run. This could be achieved by extending the DISTRB function to include a parameter number or a variable name or, alternatively a facility similar to UPDATE for use at run-time could be implemented.

GUILDS is structured so that the language can be readily implemented on any computer which supports STAGE2 and FORTRAN, with the minimum of changes required. The areas where modifications may be necessary are the input/output routines including the device handling and the keyboard interrupt facility.

GUILDS is intended to be user modifiable, particularly the FORTRAN section of the language, since, an understanding of STAGE2 would be required before the translator could be altered. It would thus be possible to tailor the language to a particular user's requirements. The inclusion of the PIPE macro (section 6.3.4) for the hydro generator simulation is one example of a "user modification".



Quotation marks indicate questions asked of the user at run-time; in all other decision boxes variables set up in DIALOG are being tested.

Figure 4.1 - Operation of FORTRAN Program

Routine Name	Description of Routine	Routines Called
FIRST	Main program	START CSI DIALOG PLTSET HEAD VMETHS EXTERN METHS TERM RDOC1 POSTOU
START	Writes start-up message and calls INIT.	INIT
INIT	Translation of INITIAL segment of Model Description.	UPDATE
UPDATE	Permits the values of variables in UPDATE statements to be altered by the user at run-time.	
DIALOG	Interactive dialogue for controlling the simulation run.	
CSI	Command string interpreter to replace the above dialogue (4.2.1).	
PLTSET	Outputs axis etc. for plotted output.	
HEAD	Outputs headings for printed and stored (ACSII) output.	
VMETHS	Variable step length integration routine.	MODEL RK1 PRIN PLOT STORE

Figure 4.2 - Simulation Language Subroutines
(Part 1 of 3)

Routine Name	Description of Routine	Routines Called
EXTERN	User-supplied integration routine.	MODEL PRIN PLOT STORE
METHS	Fixed step length integration routines.	MODEL PRIN PLOT STORE
MODEL	Translation of DYNAMIC segment of Model Description.	GUILDS functions
PRIN	Routine for printed output.	
PLOT	Routine for plotted output.	
STORE	Routine for stored output.	
TERM	Translation of TERMINAL segment of Model Description.	
RDOC1	Writes part of run-time documentation file (4.2.3).	RDOC2 HEAD
RDOC2	Writes part of run-time documentation file (4.2.3).	
POSTOU	Post-run Output dialogue and output routines.	CSI2 HEAD PLTMM PLTSET
CSI2	Command string interpreter to repace above dialogue (4.2.1).	
PLTMM	Calculates mimimum and maximum value for plotted output.	

Figure 4.2 - Simulation Language Subroutines
(Part 2 of 3)

Routine Name	Description of Routine	Routines Called
ANDHYS, DELAY DISTRB, FNSW HYST, LIMIT RAMP, RAMP1 RAMP2, RATLIM STEP, STPLIM SWIN	GUILDS functions (see Appendix C the GUILDS User's Guide).	
RDTSK	Main program of separate task for handling Keyboard Interrupt (4.2.4).	

Figure 4.2 - Simulation Language Subroutines
(Part 3 of 3)

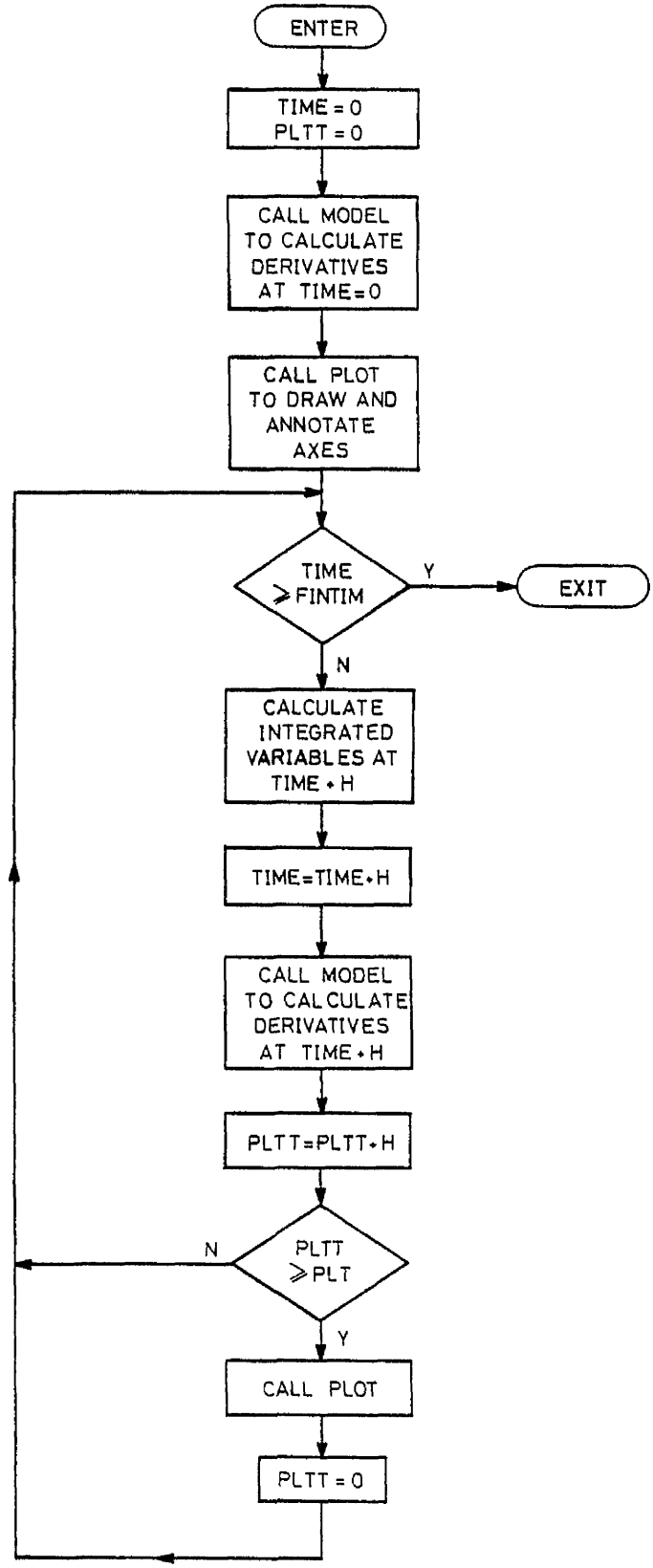


Figure 4.3 - Operation of Integration Routine

Features	GUILDS	DARE-P[29]	ISIS[30,31]	ACSL[32]
Graphics display of solution transient	Tektronics VDU and/or plotter	Print-plot & Calcomp	Print-plot & several VDUs	Print-plot & several VDUs
Interaction with model during run time	Yes, through keyboard interrupt	Not possible	Yes, through keyboard interrupt	Not possible
Graphics display of one or more key transients during run	Yes	Not available	Yes	Not available
Ability to call external sub-programs	Present	Present	Not possible	Present
Data able to be entered separately from model	Yes	No	Yes	Yes, between runs
Macro capability	Present	Not present	Subroutine facility is similar	Present
Flexibility of mode of presentation of results	Considerable (additional plot package available)	Excellent with cross-plot facility available	Considerable	Considerable
Graphics editing	Possible with use of interactive commands	Not readily available	Possible with use of interactive commands	Possible with use of interactive commands
Equation-orientated model description	Yes	Yes	Yes	Yes
Ability to branch between regions of simulation	Yes, but not allowed to branch into or out of derivative blocks. Usually between terminal and initial regions for re-run of simulation			

Figure 4.4 - Comparison of Simulation Languages
(Part 1 of 2)

Features	GUILDS	DARE-P[29]	ISIS[30,31]	ACSL[32]
Statement storing	Dynamic region only	Derivative blocks only	Not carried out	Derivative section only
Size of models	Limited by size of COMMON blocks containing variable names		Limited commonly by no. of lines of code for model definition	Limitations not immediately apparent through use of dynamic storage
Vector integration	Not possible	Possible in limited context of REPEAT	Not possible	Possible through use of INTVC operator
Output of translator	Set of FORTRAN subroutines	Set of FORTRAN subroutines	Binary work files only	Set of FORTRAN subroutines
Integration routines supplied with package	5 Routines, including Variable-step Runge-Kutta	12 routines including R-K-Merson Gear	Fixed step R-K and variable-step R-K-Merson	6 routines, 2 variable-step (Gear & Adams-Moulton) 4 fixed-step

Figure 4.4 - Comparison of Simulation Languages
(Part 2 of 2)

CHAPTER 5

HYDRO-GENERATOR THEORETICAL RELATIONSHIPS

5.0 Introduction

The model of the hydro-turbine generator and associated hydraulic system at Sloy Power Station used for simulation studies is described in the following sections along with the development from a relatively primative linear model to a full non-linear representation. The model is based on those described in References 1 and 3 and in particular some of the early development was carried out in collaboration with the author of Reference 3. The present model contains a more detailed representation of the pipeline characteristics and of the non-linear relationship between servo position and power than the previous models. The derivation of the per-unit form of the equations is left until Chapter 6 and the simulation studies with this model and the results of site tests are discussed in Chapter 7. The values of all the base quantities are given in Chapter 6 along with the values used for the constants and parameters of the model.

The system subdivides into sections as shown in Figure 5.1. The principal sections of the model, as discussed below are:-

(a) Generator

(b) Turbine and Pipelines

(c) Hydraulic Servo System

(d) Governor

In all the simulation studies only one generator in the station was simulated and all the other sets were assumed to be off and thus the effects of the other sets on the pipeline characteristics could be neglected. All experimental work carried out on site was thus conducted with only No. 3 set in operation. It should be noted that, with other sets running, the head available at each machine would be

reduced, due to increased losses in the pipeline, and also, any oscillations in the pipeline (between the reservoir and the surge shaft) would be more heavily damped. Thus, should it be necessary to carry out multiple machine studies, these effects would have to be included in the simulation.

5.1 Generator

In all the simulation studies the generator was assumed to be a lumped inertia and was modelled as a single integrator with a time constant equivalent to the inertia constant of the machine (Ta). The torque output from the machine (F_e) less the load torque (F_L) accelerates the machine thus the speed (ω) can be calculated from the expression for angular acceleration. That is:-

$$F_a = I_a\dot{\omega} \qquad 5.1$$

$$\Rightarrow \quad \omega = \frac{F_a}{sI_a} \qquad 5.2$$

where F_a is the accelerating torque ($F_a = F_e - F_L$),

I_a is the moment of inertia of the turbine/generator, and

s is the Laplace operator.

Using this simple model of the generator meant that the dynamics of the machine were not modelled and thus, for example, synchronising transients could not be simulated. However, these dynamic effects within the generator were considered to be of secondary importance when compared to the response of the governor and the turbine-generator to frequency disturbances. This model proved to be sufficiently accurate for the simulation studies carried out.

5.2 Turbine and Associated Hydraulics

5.2.1 Classical Transfer Function

The turbine and pipeline were originally modelled using the classical linear turbine transfer function derived by Woodward[33]:-

$$F_e = \frac{1 - sT_w}{1 + 0.5sT_w} * y \qquad 5.3$$

where T_w, the turbine water time constant, is a convenient way of representing the hydraulic system associated with the turbine. However, this transfer function is derived assuming that the head at the main inlet valve is discharged across the water control valve. This is the case for Pelton wheels which are impulse turbines, but for Francis turbines, where a proportion of the head is discharged across the runner, this transfer function is not strictly correct. Agnew[34] has developed a more accurate representation of the turbine characteristics taking account of the reaction head discharged across the runner. For the turbine operating near full load:-

$$\Delta P = \frac{1 - sT_w}{1 + sT_w/2\alpha} * \left[\Delta A - \left(\frac{1 - \alpha}{\alpha}\right) * \Delta\omega\right] \qquad 5.4$$

where ΔP, ΔA and $\Delta\omega$ are the per-unit deviations in power, control valve area and speed respectively and α is a dimensionless constant determined from the head and the design of the control valve.

The above transfer functions have been derived assuming small changes about an operating point, resulting in linear expressions. However, the hydraulic system at Sloy is highly non-linear and T_w is not constant but varies with flow and hence load. As a result, these transfer functions are only approximate and are not suitable for modelling large disturbances. Also, the effect of the pipeline and surge shaft dynamics on the response of the turbine have

been neglected. Thus, it was decided to implement a more explicit representation of the component parts of the system in preference to the linear models given above.

5.2.2 Pipeline Equations

To implement the pipeline dynamics in the simulation the equations developed by Wood[35] for simulating waterhammer on an analogue computer were also included. Other methods of analysis[36,37,38] were considered but the four first order differential equations resulting from Wood's equations were in a form suitable for inclusion in the simulation and thus this method was adopted. The method of characteristics recommended by Bryce[1] and discussed by Wylie and Streeter[39] was examined, but as this method relies on specified time intervals, computed from the wave velocity and pipe length, which, in general, are different for each pipe section and do not match the integration interval used by the simulation package, this method could not be conveniently implemented.

For a pipe section as shown in Figure 5.2, with one internal point the equations are:-

$$\frac{dQ_1}{dt} = -F^*(-3H_1 + 4H_2 - H_3) \qquad 5.5$$

$$\frac{dQ_2}{dt} = -F^*(H_3 - H_1) \qquad 5.6$$

$$\frac{dH_2}{dt} = -K^*(Q_3 - Q_1) \qquad 5.7$$

$$\frac{dH_3}{dt} = -K^*(3Q_3 - 4Q_2 + Q_1) \qquad 5.8$$

where $F = gA/L$ 5.9

$K = a^2/gAL$ 5.10

and A is the cross sectional area

L is the length and

a is the wave velocity

of the particular pipe section.

These equations define the flow into the pipe section and the head at the downstream end of the pipe, given the head at the upstream end of the section and the flow out of the pipe. The boundary conditions are determined externally from the conditions at the upstream and downstream ends of the section.

For a pipe section model representing several different conduits, effective areas and wave velocities were used, weighted with respect to the lengths of the different pipes. The wave velocities for the different conduits were calculated from the expression[40]:-

$$a = \frac{\sqrt{B/\rho}}{\sqrt{1 + f(B/E)}} \qquad 5.11$$

where a is the wave velocity

ρ is the density of the fluid

f is stress factor (see below)

B is the bulk modulus for the fluid

E is elastic modulus for the pipe.

The stress factor is determined from the construction of the pipe or tunnel: for thin-walled pipes f is given by the ratio of the internal diameter of the pipe to the wall thickness; for a tunnel, the limiting case of a thick-walled pipe, f is approximately equal to 2. The values of 'a' used in the simulation model are calculated in section 6.3.4.

Initially, the pipeline was modelled as a composite single section representing the pipeline from the surge shaft to No. 3 set (Figure 5.3), neglecting the branches to other sets. This was later expanded to a three pipe section model, as shown in Figure 5.4: section A representing the tunnel from the surge shaft to the first bifurcation; section B, a single composite section, representing the conduits from the bifurcation to sets No. 1 and 2; section C representing the conduits from the bifurcation to set No. 3 neglecting the branch to set No. 4.

In both these models, which gave similar results under all operating conditions simulated, the surge shaft was considered to be of infinite capacity, that is, effectively the reservoir. However, it was known that the U-tube oscillations between the surge shaft and the reservoir had some effect on the turbine during large disturbances. Thus the final model of the hydraulic system included equations to represent the surge shaft and the surge shaft to reservoir pipeline (Figure 5.5). The surge shaft, which has a free surface, is assumed to isolate the upstream conduits from any downstream transients and thus, the frequency of any oscillations is such that the conduit can be considered to be an inelastic water column[40]. As a result, the surge shaft can be modelled by the equations:-

$$\frac{dH_s}{dt} = \frac{Q_s}{A_s} \qquad 5.12$$

$$Q_s = Q_d - Q_a \qquad 5.13$$

and, in addition, the flow in the conduit between the reservoir and the surge shaft can be represented by the equation:-

$$\frac{dQ_d}{dt} = F_d*(H_r - H_s) \qquad 5.14$$

where the subscripts 's', 'r' and 'd' represent the surge shaft, the reservoir and the reservoir to surge shaft pipeline respectively and Q_a is the flow into the penstock represented by other pipe sections. The data used in the simulation model for this final hydraulic representation are to be found in section 6.3.3.

The turbine and reservoir impose boundary conditions on the pipeline: the reservoir head (H_r) is assumed to be constant for any given loch level over the test period; the flow at the downstream end (Q_t) is defined by the turbine equation. (If a relief valve is included in the model (section 5.2.5) the flow at the downstream end of the pipeline is the sum of the turbine flow (Q_t) and the relief valve flow (Q_{rv}).)

5.2.3 Turbine Hydraulic Equations

To define the boundary conditions at the downstream end of the pipeline it is necessary to have an expression for the flow through the turbine. In general, it is considered that the flow through a turbine, equivalent to the flow through the control valve, can be represented by the equation for flow through an orifice:-

$$Q = C_d A\sqrt{2g\Delta H} \qquad 5.15$$

where C_d is the coefficient of discharge of the orifice

A is the area of the orifice

ΔH is the head loss across the orifice.

For impulse turbines, such as Pelton wheels, it is assumed that the entire head at the main valve of the machine (H_t) is discharged across the control valve, thus:-

$$Q_t = C_d A_{cv} \sqrt{2gH_t} \qquad 5.16$$

where Q_t is the flow through the turbine

A_{cv} is the effective area of the control valve.

However, in the case of Francis turbines some of the head at the main valve is also discharged across the runner, the reaction head (H_R). The head at the main valve, effectively the total available potential energy, is converted to torque in the turbine, partly as the kinetic energy of the flow through the control valve and also as a result of reaction on the turbine as the water is accelerated through the runner. Thus, it is necessary to modify the flow equation as the reaction head effectively reduces the head discharged across the control valve:-

$$Q_t = C_d A_{cv} \sqrt{2g(H_t - H_R)} \qquad 5.17$$

It thus remains to derive an expression for the reaction head (H_R).

From Euler's Turbine Theorem[41,42] for axial flow in a turbine runner, the gross torque developed (F_g) is given by:-

$$F_g = \rho Q(r_1 w_1 - r_2 w_2) \qquad 5.18$$

where ρ is the density of the fluid

Q is the flow through the runner

r_1, r_2 are the outer and inner radii of the runner

w_1, w_2 are the whirl velocities of the fluid at the outer and inner radii of the runner (the whirl velocity is the circumferential component of the absolute velocity of the fluid)

and from this equation, it can be shown that:-

$$H_R = \frac{F_g}{\rho g Q} = \frac{u_1 w_1 - u_2 w_2}{g} \qquad 5.19$$

where u_1, u_2 are the velocities of the runner at the outer and inner radii of the runner.

From this equation, making some simplifying assumptions[41,42] it is possible to derive the expression:-

$$H_R = \frac{\omega^2(r_1^2 - r_2^2)}{2g} + f(Q^2) \qquad 5.20$$

where the function of Q^2 is considered to be sufficiently small to be neglected, and thus the reaction head can be evaluated from the speed and the physical dimensions of the runner.

Substituting this expresion in the flow equation gives:-

$$Q_t = C_d A_{cv}\sqrt{2gH_t-(r_1^2 - r_2^2)\omega^2} \qquad 5.21$$

and thus the boundary conditions at the downstream end of the pipeline have been defined, assuming that values can be obtained for all the variables in the above expression.

5.2.4 Turbine Mechanical Equations

The mechanical power output (P_m) from a hydro-turbine is given by the equation:-

$$P_m = \rho g Q_t H_t \eta_t \qquad 5.22$$

where ρ is the density of the fluid

g is the gravitational acceleration

Q_t is the flow through the turbine

H_t is the head loss across the turbine

η_t is the turbine efficiency.

The torque produced by the turbine is therefore given by:-

$$F_m = \frac{\rho g Q_t H_t \eta_t}{\omega} \qquad 5.23$$

where ω is the angular velocity of the turbine.

Assuming a linear relationship between mechanical and electrical torque, then:-

$$F_e = \eta_g F_m \qquad 5.24$$

where η_g is the generator efficiency.

In order to simulate run-up and load rejection conditions it was found to be necessary to include a constant loss term in equation 5.22 to represent the no load losses in the turbine (see section 6.3.2) but the linear relationships were used initially as a first approximation. The difference between the electrical load torque (F_L) and the electrical torque (F_e) is the accelerating torque (F_a) which causes speed and hence frequency changes in the generator. Under isolated load conditions it is necessary to include a load self-regulation factor (e_n) (i.e. the variation of the electrical load with frequency) in the simulation model (see section 6.4.1).

5.2.5 Relief Valve

During site tests it was observed that the relief valve effected the dynamics of the turbine, particularly under isolated load conditions, thus it was decided to include a simple representation of the relief valve in the simulation model. The relief valve is operated by the main servomotor through a mechanical linkage involving a piston/dashpot and spring arrangement. This gives rise to a highly non-linear operation, described in detail elsewhere[1,14], which, from observations on site, was found to be of an unrepeatable nature largely dependent on past history. A first order lag was used in the simulation to represent the operating mechanism.

During normal closing of the servo the relief valve is inoperative, however, if the servo closes faster than a certain threshold rate the relief valve starts to open. The rate of opening of the relief valve is approximately the same as the closing rate of the servo (less the threshold). The relief valve starts to close,

under the action of a spring, as soon as the main servomotor stops closing, at a rate determined by a timing screw on the dashpot[1]. If the servo starts to open while the relief valve is closing, the reclosure rate is increased by the rate at which the servo is opening. There is a throttle in the hydraulic pipework which limits the reclosure rate.

It was decided to model the hydraulic characteristics of the relief valve in a similar way to the main control valve. The manufacturer's discharge curves confirmed that the flow through the relief valve was approximately proportional to position and, since the valve is of an annular construction, area. Thus the expression for a sharp edged orifice could be used as a reasonable approximation to the flow through the valve:-

$$Q_{rv} = C_{drv} A_{rv} \sqrt{2gH_t} \qquad 5.25$$

where Q_{rv} is the flow through the relief valve

C_{drv} is the discharge coefficient of the valve

A_{rv} is the area of the valve (proportional to position).

The equations for the complete relief valve model are given in Chapter 6 and can be found in the Model Description of the hydro generator in Appendix 2.4.

5.3 Hydraulic Servo System

The hydraulic servo system, shown in block diagram form in Figure 5.6, consists on an electro-hydraulic distribution valve, which is represented by a first order lag, and an hydraulic servomotor which is modelled as an integrator with gain. The input to the integrator is limited to represent flow rate restrictions on the servomotor which exist in practice to satisfy operational limits on the speed of opening and closing the water control valve. The electrohydraulic servo valve

restricts the oil flow in the hydraulic pipework but an additional valve is required to limit the flow when the servo is opening as the opening rate is much lower than the closing rate. There is a position feedback loop round the servo system derived from a linear variable differential transformer (LVDT) mechanically connected to the servo.

During some early site tests under simulated isolated load conditions[3], limit cycles on the servo, and hence the frequency, were observed. These limit cycles (shown in Figure 5.7(a) and 5.7(b) for the Temporary Droop and Double Derivative governors respectively) were a result of non-linearities in the mechanical linkages between the servo and the water contol valve (see Plate 2). An hysteresis function was included in the simulation model to represent the backlash in the linkages and this was found to give a good representation of the limit cycles, although other effects, for example friction and the flexing of shear links, were known to be involved. Also, the force of the water on the guide vanes, which acts to close the water control valve in the event of linkage failure, has been neglected.

When the results of the simulation were compared with the site results at different load settings it was found that the amplitude and frequency of the limit cycles produced by the simulation were different from those observed on site. At first it was thought that this was because the model of the mechanical linkages was too simple and that a more detailed representation was required. However, further investigation using the simulation model indicated that by including a "gain factor" ($<<1$) in the flow equation for low flow conditions, limit cycles of a similar amplitude and frequency to those obtained on site could be produced. This was not a satisfactory solution as a different gain factor had to be found, largely by trial and error, for different load settings and also different heads.

On site it had been observed that the relationship between servo position and power output was not linear: a typical site result is given in Figure 5.8 recorded over long period, to minimise the effect of surge shaft oscillations, as the power output from the set was slowly increased. This curve can be seen to have a gradient less than unity at low loads which is as required by the simulation to give limit cycles of the correct frequency and amplitude. By including a non-linear function derived from this curve in the simulation, results similar to those obtained on site could be obtained without the need for a gain factor. The non-linear function used is derived in section 6.3.7 of Chapter 6 and a comparison of site and simulation results is given in of Chapter 7.

5.4 Governor

The governors used experimentally on site and represented mathematically in the digital simulation are described in detail in References 1 to 4 and in particular Reference 4 presents an excellent review of governing techniques. In addition to the governors, a controller is required to generate reference set points for the frequency, servo and load limit (see below) and to provide sequential control for run-up and shut down of the set. Only for certain simulation studies are these aspects of sequential control required, for example run-up simulation and load rejection, and thus all the models used do not included the servo and load limit set points.

Two different types of governors are of interest and the block diagrams and transfer functions for these governors are shown in Figures 5.9 and 5.10. The original mechanical governor in Sloy Power Station is a Temporary Droop (TD) governor (Figure 5.9) and the electronic and microprocessor governors developed by Bryce[1] and Findlay[3] from the work of Schlief[43] are Double Derivative (DD)

governors (Figure 5.10). As both these types of governors were used on site for experimental work it was necessary for the simulation be able to model both TD and DD governor characteristics.

It can be shown that the TD and DD can be represented by the same transfer function with different values for the parameters. Thus, it was only necessary to include one set of equations in the simulation model to represent both governor types.

Considering the DD transfer function shown in Figure 5.10 and letting $K_2 = 0$ this becomes:-

$$\frac{y_g}{f_e} = -\frac{1 + (T_3 + K_1)s}{b_p(1 + (T_3 + T_L)s + T_3T_Ls^2)} \qquad 5.26$$

and comparing this with the TD transfer function (Figure 5.9) gives:-

$$T_3 + K_1 = T_d \qquad 5.27$$

$$T_3 + T_L = \frac{(b_p + b_t)T_d + T_y}{b_p} \qquad 5.28$$

$$T_3T_L = \frac{T_dT_y}{b_p} \qquad 5.29$$

The values used for all the governor parameters are given in section 6.3.8 (Table 6.5) and by subsituting the values of T_d, T_y, b_p and b_t for the TD governor in equations 5.27 to 5.29 and solving, new values for T_3, T_L and K_1 can be obtained enabling the TD governor to be represented by the DD governor equations.

The power output from the set using the TD governor is controller by adjusting the frequency reference (f_s). However, as this causes any instructed load changes to be input to the governor, which has a dominant time constant of about 160s, the response of the set is somewhat slow. As a result, the DD governor was implemented with a

separate loading control which causes a signal, the servo set point (y_s), referred to as 'desired MW', to be added into the governor output (y_g) to give the desired servo position (y_d). There is also a load limit set point (y_L) which prevents the power output from the set rising above a certain level and is used to control the operation of the governor during the run-up sequence. The DD governor compares the desired servo position with the load limit and sets y_d equal to y_L if greater than y_L. Thus:-

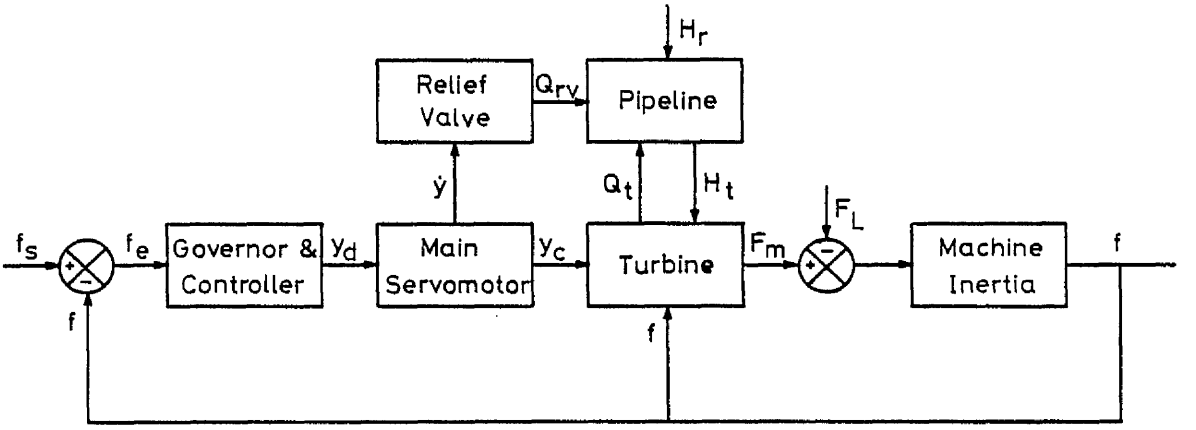
for the TD governor

$$y_d = y_g \qquad 5.30$$

and for the DD governor

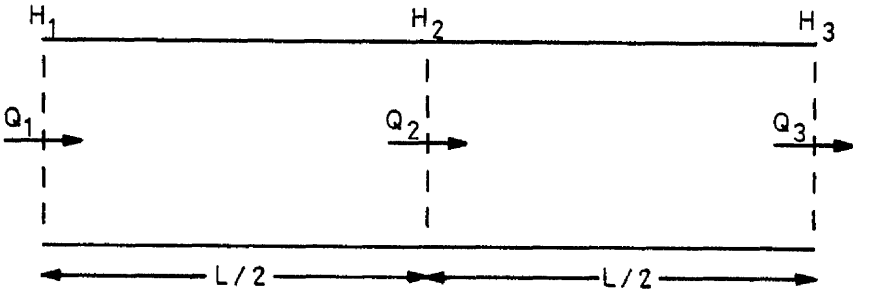
$$y_d = y_g + y_s; \quad y_d < y_L \qquad 5.31$$

$$y_d = y_L; \quad y_d > y_L \qquad 5.32$$



f	- Frequency	Q_{rv}	- Relief Valve Flow
f_s	- Frequency Reference	H_r	- Reservoir Head
f_e	- Frequency Error	Q_t	- Turbine Flow
y_d	- Desired Servo Position	H_t	- Turbine Head
y_c	- Control Valve Position	F_m	- Turbine Torque
$\dot{y}$	- Servo Rate	F_L	- Load Torque

Figure 5.1 - Hydro-generator Simulation Model



Q - Flow
H - Head

Figure 5.2 - Pipe Section Representation

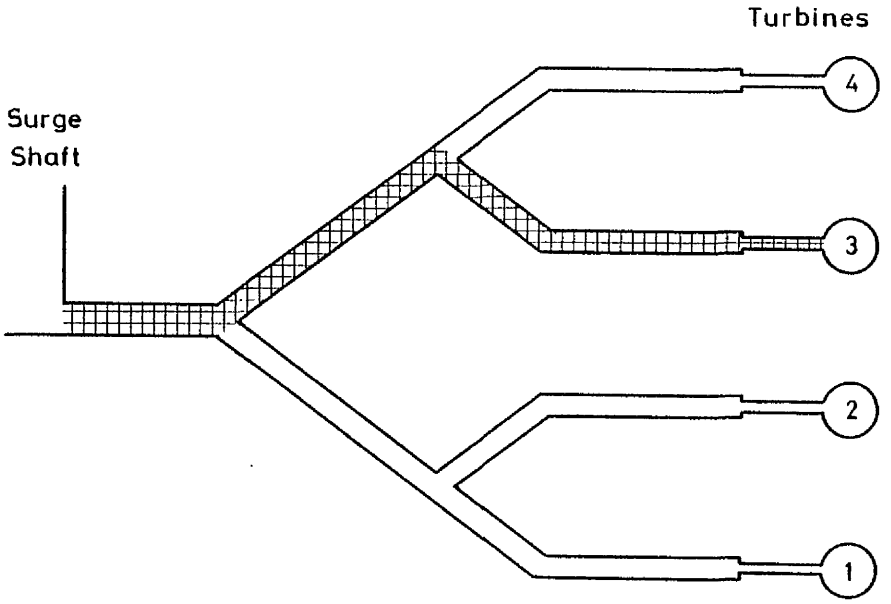


Figure 5.3 - Single Pipe Section Model

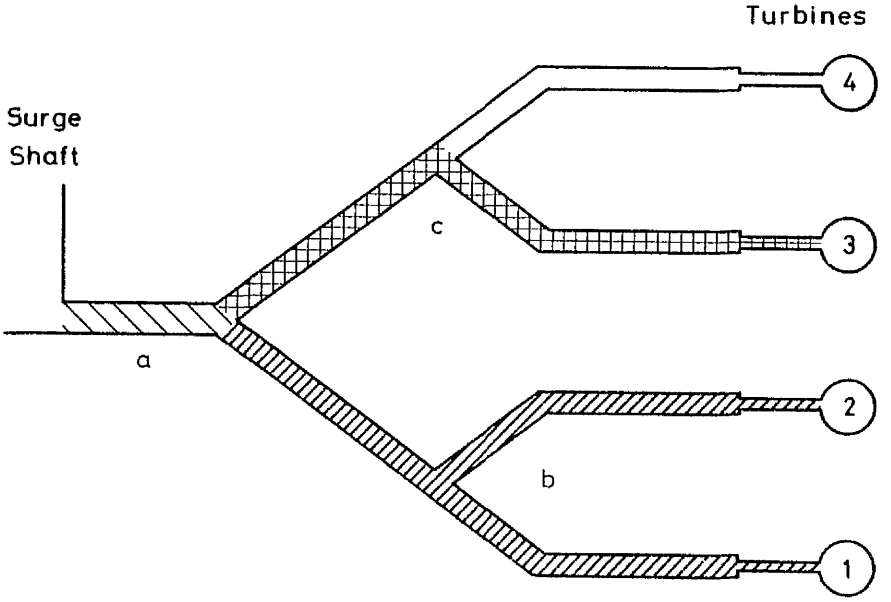


Figure 5.4 - Three Pipe Section Model

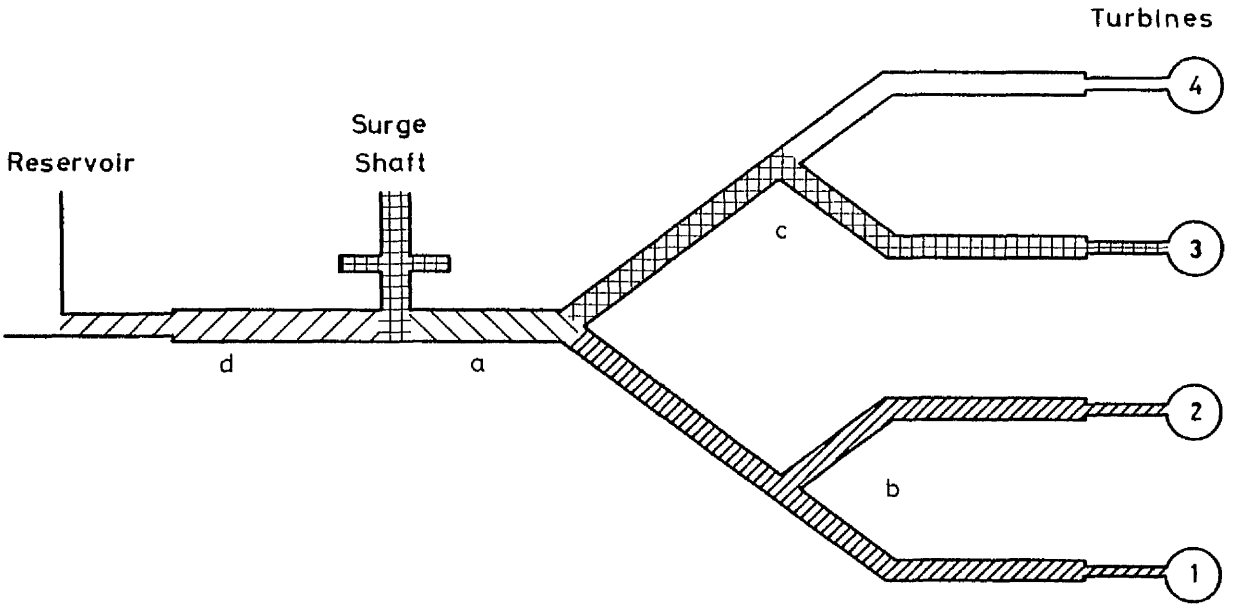
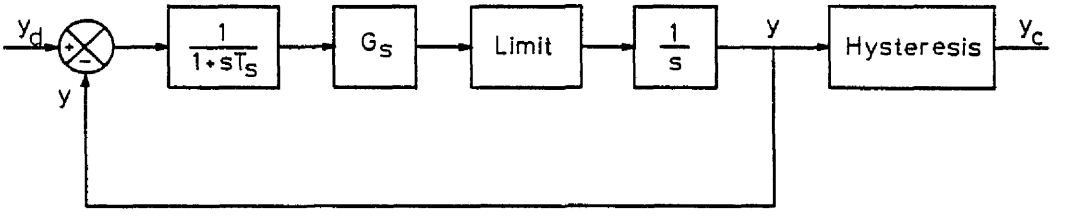
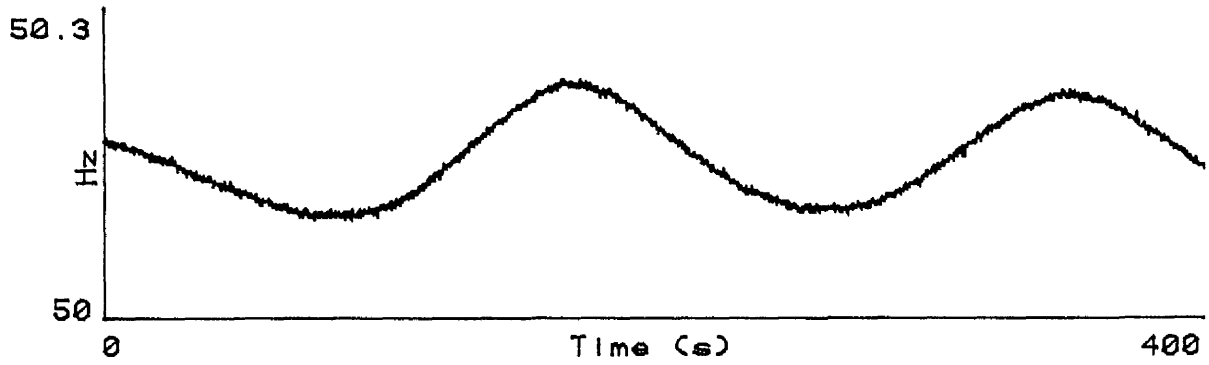


Figure 5.5 - Three Pipe Section Model with Reservoir and Surge Shaft

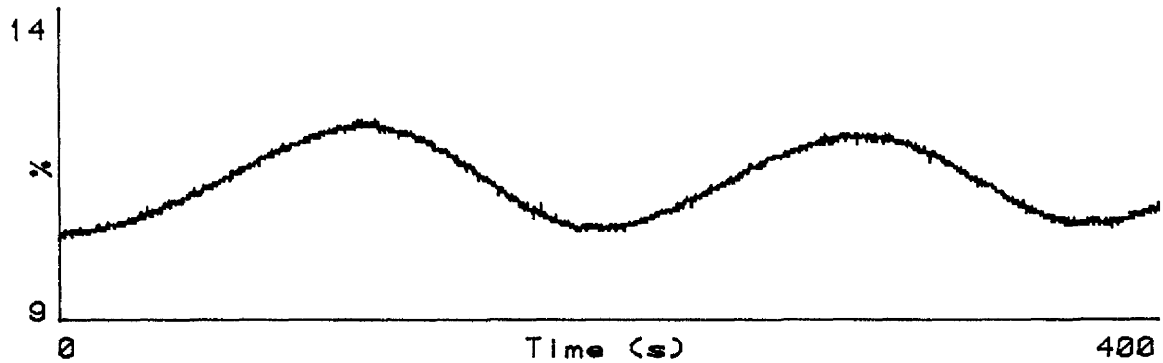


y_d - Desired Servo Position
y - Servo Position
y_c - Water Control Valve Position
G_S - Servo Gain
T_S - Servo Valve Time Constant

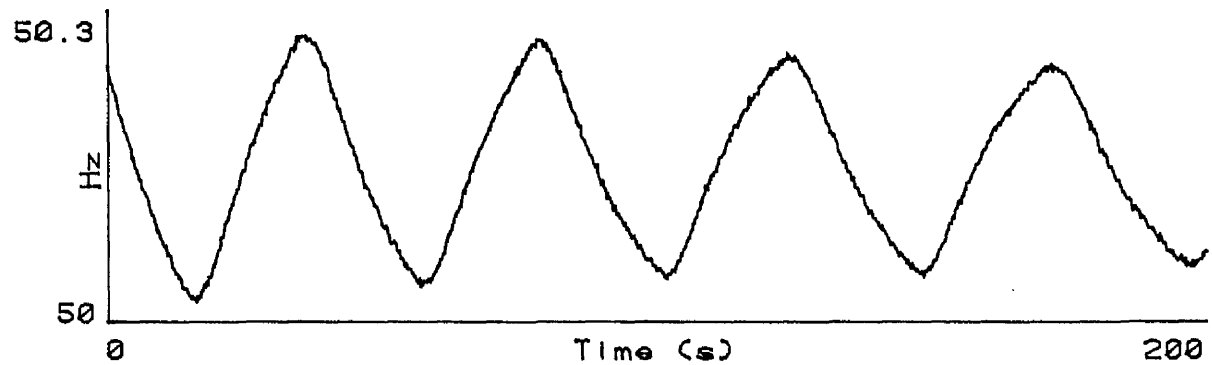
Figure 5.6 - Hydraulic Servo System Representation



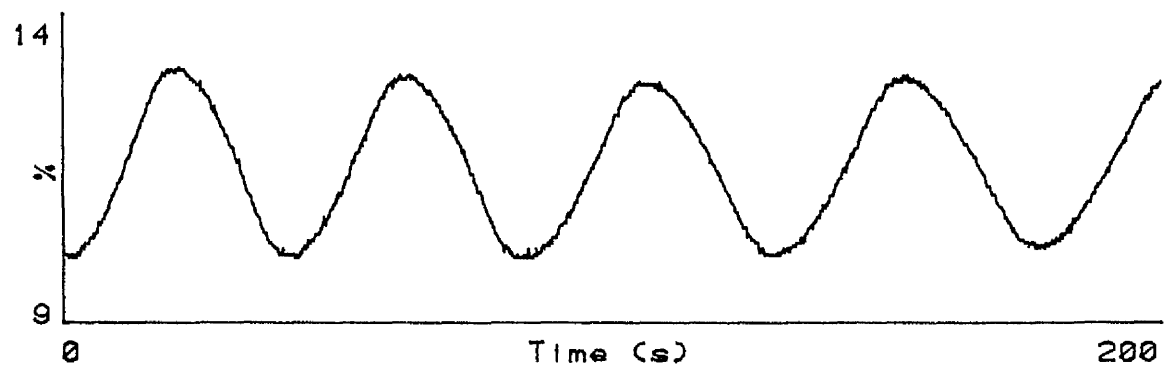
(a) TD Governor - Frequency



(a) TD Governor - Servo Position



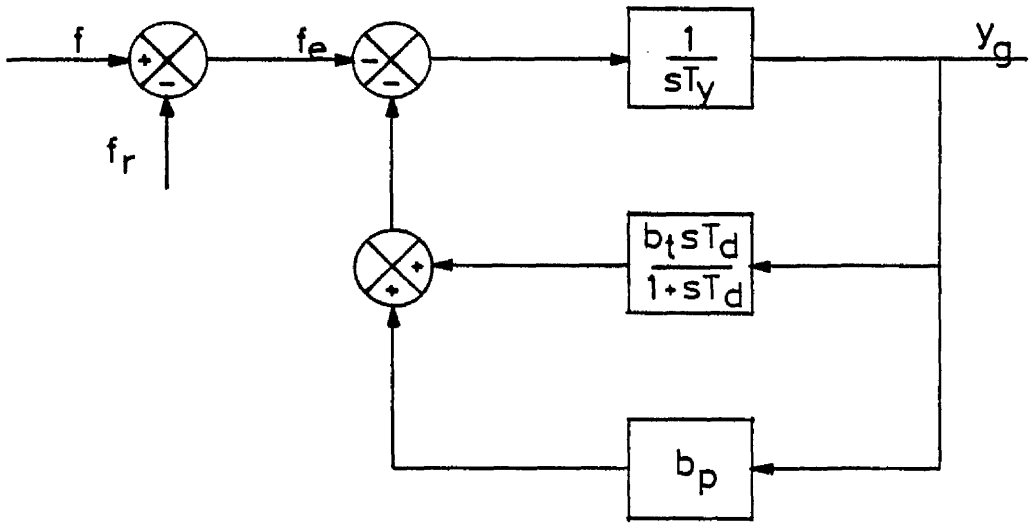
(b) DD Governor - Frequency



(b) DD Governor - Servo Position

Figure 5.7 - Limit Cycles

Figure 5.8 - Typical Non-linear Function Recorded on Site



$$\frac{y_d}{f_e} = -\frac{1 + sT_d}{b_p\left(1 + \left(\frac{(b_p + b_t)T_d + T_y}{b_p}\right)s + \frac{T_dT_y}{b_p}s^2\right)}$$

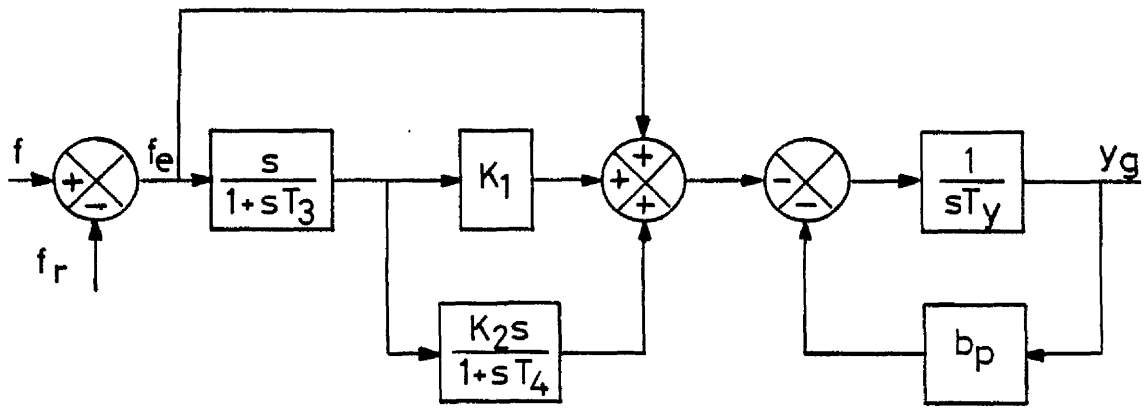
y_g = governor output

f = system frequency

f_r = frequency reference

f_e = frequency error

Figure 5.9 - Temporary Droop Governor



$$\frac{y_d}{f_e} = -\frac{1}{b_p + sT_y}\left(1 + K_1\frac{s}{1 + sT_3} + K_2\frac{s^2}{(1 + sT_3)(1 + sT_4)}\right)$$

y_g = desired servo position

f = system frequency

f_r = frequency reference

f_e = frequency error

$T_L = T_y/b_p$ = governor dominant lag

Figure 5.10 - Double Derivative Governor

CHAPTER 6

MODEL EQUATIONS FOR HYDRO-GENERATOR SIMULATION

6.0 Introduction

The model equations for the hydro-generator, given in Chapter 5, are presented in this Chapter in per-unit form as they are used in the simulation. Data for the simulation have been obtained from various sources including References 1 and 3 and the NSHEB and also as a direct result of tests carried out on site. The validity of the final model is discussed in Chapter 7 where the simulation results are compared with results of site tests.

6.1 Development of Site Facilities

The test facilities developed during the work of Bryce[1] and Findlay[3,4] in Sloy Power Station were adequate for interfacing experimental governors with the station equipment for evaluation of novel governor strategies. Tests under these conditions were always carried out under the close supervision of both University and Power Station personnel.

Following from this work, a more secure microprocessor governor was developed by Grant[6] and it was decided that, in order to further evaluate the governor, it would be necessary to gain some experience of operational service in the power station environment. Thus, it was necessary to provide a well defined and operationally satisfactory interface on site, to which experimental governors could be connected, along with all the relevant wiring diagrams, operating instructions and other documentation.

The re-engineering of the site equipment, including design and development, constructional work and documentation, was carried out in collaboration with Dr. Grant, a contemporary research student, and

other University personnel. Dr. Grant[6] gives a comprehensive description of the work involved and, as it was more directly related to his project, it was decided not to repeat this description here. Reference 18, and Appendix 1, describe the resulting "Controller Testing Facility" in some detail.

The installation of a permanent interface on site greatly reduced the time required to set up and execute site tests. The tests carried out during the commissioning of this equipment provided further information for the development of the simulation model and useful site results for comparision with the results of the simulation studies. Throughout this work, GUILDS was used to assist the development of the governor and, in particular, some unexpected results obtained on site during commissioning tests were reproduced and the cause traced, using the simulation model.

6.2 Base Quantities

A per-unit system was used in all the simulation studies, with base quanties as described below, for deriving the equations in the model from the theoretical expressions. Imperial units were used almost exclusively as all the site data and specifications are in imperial units. The principal quantities to which this applies are water power (P_w), mechanical power (P_m), electrical power (P_e), turbine head (H_t), turbine flow (Q_t), control valve area (A_{cv}) and servo position (y).

As the input to the turbine from the governor is servo position and the output from the generator is electrical power (and frequency) it was decided to define base quantites for these variables and to define the other variables in terms of these bases. Thus, the base electrical power was taken as 32.5MW, the nominal rating of the machine, and base servo positon was chosen to be the value of the servo

at maximum travel. The base value for frequency was taken as 50Hz with the equivalent base speed being 44.88 rad/s, since the generator is a 14 pole machine.

Following from this, the base area for the water control valve is that value which corresponds to 1 p.u. servo position (i.e the control valve fully open) and base flow and head are the values of flow and head which give 1 p.u. electrical power at 1 p.u. servo position.

The mechanical linkage which was added when the new high pressure servo was attached to the existing low pressure servo restricted the movement of the servo to about 9/10 of the full travel available before the addition was made. Since, for all practical purposes, 9/10 of full travel is the maximum travel of the servo, this position was taken to be 100% effective servo position and thus the base value for servo position was assumed to be 100%. The position of the servo is measured on site by a linear variable differential transformer and the output of this LVDT was adjusted to give 0 to 10V for 0 to 100% effective servo position. The base value for the control valve area was calculated from equation 5.21 with all the other variables assuming their maximum or base values (see section 6.3.3).

The head available at the turbine is given by the difference between the level of Loch Sloy (the reservoir) and that of Loch Lomond (the tailrace). The value for the base head, 266.7m (875ft), was arrived at after a series of measurements taken on site to establish at what head the output power was 32.5MW for 100% servo position.

By choosing base quantites in relation to electrical power, bases for mechanical power and water power are implicitly defined to be those values which, taking account of efficiency in the turbine and generator, give 1 p.u. electical power. The base mechanical power was taken to be 33.5MW (45000bhp) by assuming a generator efficiency of 97%, which is a typical value for machines of this size.

As no accurate measure of flow was available on site, the value of the base quantity for flow corresponding to 33.5MW (45000bhp) at a head of 266.7m was obtained from the Manufacturer's Predicted Turbine Performance Curves (reproduced in Figure 6.1). This value was 14.15m^3/s (500ft^3/s). The base value for water power was calculated from equation 5.22 to be 37.0MW, giving a turbine efficiency at full load of 0.91 (cf. References 3, 40 and 42). The variation of efficiency with load is discussed in section 6.3.2.

This deviation from the perhaps more normal method of using invariant power bases between the different parts of the system was adopted since, at rated power output, almost all the system quantites would be 1 p.u.

It was observed during loading tests on site that the head at the turbine fell by about 20ft (6m) from no load to full load due to frictional losses in the pipeline. As the model equations for the pipeline neglect the effect of friction it was not possilbe to include this head loss in the simulation without modifying the equations. It was decided not to modify the simulation model for the present work, but, it may prove necessary at some stage to include frictional effects. If this were done, the value of 266.7m (855ft) should be used as the base head for 32.5MW at 100% servo position, thus preserving the convention of all the variables being 1 p.u. at full load. In this case, however, the head at the reservoir end of the pipeline would be slightly in excess of 1 p.u. with the turbine head equal to 1 p.u. and non-zero flow through the turbine.

6.3 Model Equations

The equations derived in Chapter 5 for the hydro turbine model (Figure 5.1) are developed, in the following sections, into the per-unit expression used in the simulation model. The base quantities given above are used throughout and the subscript 'o' is used to denote the base value of a variable. Per-unit variables are denoted by a bar over the variable name (e.g $\overline{f}$) but except where an equation contains both physical and per-unit variables, the bar is generally omitted. Equations are derived using physical variables (Chapter 5) and are used in the simulation model in per-unit form.

6.3.1 Generator and Load Inertia

From equation 5.2, dividing by the appropriate base values:-

$$\frac{\omega}{\omega_o} = \frac{F_e - F_L}{F_{eo} s I_a} * \frac{F_{eo}}{\omega_o} \qquad 6.1$$

Now, as a convenient way of representing the generator and load inertia, define an alternator time constant T_a (dimensioned in seconds), as:-

$$T_a = \frac{I_a \omega_o}{F_{eo}} \qquad 6.2$$

thus:-

$$\overline{\omega} = \frac{\overline{F}_e - \overline{F}_L}{sT_a} \qquad 6.3$$

and since, in the per-unit system $\overline{\omega} = \overline{f}$ then:-

$$\overline{f} = \frac{\overline{F}_e - \overline{F}_L}{sT_a} \qquad 6.4$$

From this equation it can be seen that the value of T_a can be determined from the rate of change of frequency after a step change in load, that is:-

$$T_a = \frac{F_e - F_L}{sf} \qquad 6.5$$

and for the sets at Sloy the value of T_a is generally taken to be $7s^1$ although different values were used in some of the simulation studies (see Chapter 7).

6.3.2 Turbine Equations (Mechanical)

From equation 5.22, dividing by the appropriate base values:-

$$\frac{P_w}{P_{wo}} = \frac{\rho g H_o Q_o}{P_{wo}} * \frac{Q_t H_t}{Q_o H_o} \qquad 6.6$$

$$\Rightarrow \quad \overline{P}_w = \frac{\rho g H_o Q_o}{P_{wo}} * \overline{Q}_t \overline{H}_t \qquad 6.7$$

Now, since the base water power (P_{wo}) is defined at base head and flow as:-

$$P_{wo} = \rho g H_o Q_o \qquad 6.8$$

then:-

$$\overline{P}_w = \overline{Q}_t \overline{H}_t \qquad 6.9$$

Also from equation 5.22:-

$$P_m = P_w \eta_t \qquad 6.10$$

$$\Rightarrow \quad \frac{P_m}{P_{mo}} = \frac{P_w \, P_{wo}}{P_{wo} P_{mo}} \eta_t \qquad 6.11$$

and defining:-

$$\eta_t = \frac{P_{mo}}{P_{wo}} = \frac{33.5}{37.0} = 0.91 \qquad 6.12$$

then:-

$$\overline{P}_m = \overline{P}_w \qquad 6.13$$

Similarly, defining η_g as:-

$$\eta_g = \frac{P_{eo}}{P_{mo}} = \frac{32.5}{33.5} = 0.97 \qquad 6.14$$

gives:-

$$\overline{P}_e = \overline{P}_m \qquad 6.15$$

Equation 6.13 is derived assuming a constant turbine efficiency (equation 6.12) of 0.91 and similarly equation 6.15 assumes a constant generator efficiency (equation 6.14) of 0.97. These values are not precise but gave a good approximation when used in the simulation model. Equation 6.13, however, is only valid at rated conditions and, at other operating points, it is necessary to include the variation of efficiency with load[3,40,42], as derived below.

From the Predicted Turbine Performance Curves (Figure 6.1) values of mechanical power output (P_m) can be obtained for a given flow (Q_t) and head (H_t). Also, from equation 6.9, values of water power input (P_w) can be calculated for the same values of flow and head. At a constant head of 266.7m (1 p.u.), and neglecting the head losses in the pipeline (section 6.2), the values of P_m and P_w for varying flow can be plotted in per-unit terms, as shown on Figure 6.2. This figure represents the energy conversion effected by the turbine of the power in the water into mechanical power.

The Performance Curves only provide data from about half full load to full load, thus the lower portion of Figure 6.2 (shown as a broken line) is obtained by extrapolation. The intersection of this line with the P_w axis represents the loss power required to keep the turbine turning at rated speed with no mechanical power output. There is some justification for using a linear exptrapolation since the resulting value for the no load losses is about 10% and Bryce[1] suggested that the no load flow through the turbine was about 10% of the full load flow.

Figure 6.2 can be approximated to a straight line which can be expressed by the per-unit equation:-

$$\overline{P}_m = 1.11\overline{P}_w - 0.11 \qquad 6.16$$

and since $\overline{P}_e = \overline{P}_m$ (equation 6.15) and $\overline{F}_e = \overline{P}_e/\overline{f}$ then:-

$$\overline{F}_e = \frac{1.11\overline{P}_w - 0.11}{\overline{f}} \qquad 6.17$$

One implication of this equation is that the losses increase as the frequency falls below 1 p.u., which is not the case in practice. Assuming, as a first approximation, that the loss torque is proportional to frequency, then, equation 6.17 can be written as:-

$$\overline{F}_e = 1.11\overline{P}_w/\overline{f} - 0.11\overline{f} \qquad 6.18$$

which gives the same torque as equation 6.16 at 1 p.u. frequency.

Including this equation in the simulation gives a better representation of the turbine efficiency and enables simulations of run-up and load rejection characteristics to be carried out which otherwise would have been impossible. It should be noted, however, that this representation is still only valid at or near rated speed and some further assumptions were made in order to simulate the run-up characteristics, as discussed in section 6.4.3. The linear approximation to Figure 6.2 could be replaced by a more detailed representation but in the simulation studies carried out this was not found to be necessary.

6.3.3 Turbine Equations (Hydraulic)

Substituting the nominal (base) values for full load for all the variables in equation 5.21, with r_1 = 1.09m (3.57ft) and r_2 = 0.48m (1.58ft), gives:-

$$14.15 = C_d A_{cv}\sqrt{2*9.81*266.7 - (1.09^2 - 0.48^2)*44.88^2}$$

$$\Rightarrow \quad C_d A_{cv} = \frac{14.15}{\sqrt{5232.7 - 1929.0}} = 0.246 \qquad 6.19$$

Taking this value as the base value for the effective control valve area ($A_e = C_d A_{cv}$) since C_d is assumed to be constant, then equation 5.21 becomes, in per-unit terms:-

$$\overline{Q}_t = \frac{A_{eo}\overline{A}_e}{Q_o} * \sqrt{2gH_o\overline{H}_t - (r_1^2 - r_2^2)\,\omega_o^2\,\overline{\omega}^2}$$

$$\Rightarrow \quad \overline{Q}_t = \frac{0.246\overline{A}_e}{4.15} * \sqrt{2*9.81*266.7*\overline{H}_t - (1.09^2 - 0.48^2)44.88^2*\overline{\omega}^2}$$

$$\Rightarrow \quad Q_t = A_e\sqrt{1.584H_t - 0.584\omega^2} \qquad 6.20$$

and assuming that the control valve area is directly proportional to servo position (see section 6.3.6) then A_e in equation 6.20 can be replaced by y or, if hysteresis is included, by y_c (see section 6.3.7). In addition the per-unit speed (ω) can be replaced by per-unit frequency (f).

6.3.4 Pipelines

Figures 6.3 and 6.4 show the layout of the hydraulic system simulated by the single pipe section model and four pipe section model with reservoir and surge shaft. In the first case the single pipe section is a composite representing the different conduits from the surge shaft, which is assumed to be of infinite capacity, to set No. 3 ignoring the branches to the other sets. In the second case pipe section A represents the conduit from the surge shaft to the the first bifurcation; section B is a composite representing the branch to sets

No. 1 and 2 which are assumed to be off; section C is also a composite section representing the conduits to set No. 3; and section D represents the tunnel from the reservoir, assumed to be of infinite capacity, to the surge shaft which is modelled separately.

The three downstream pipe sections (A. B and C) each require the four differential equations (5.5 to 5.8) derived in Chapter 5. For any one of these sections, equation 5.5, dividing by the appropriate base values becomes:-

$$\frac{d\overline{Q}_1}{dt} = - \frac{FH_o}{Q_o} * (3\overline{H}_1 + 4\overline{H}_2 - \overline{H}_3) \qquad 6.21$$

and similar expressions can be obtained from equations 5.6 to 5.8.

The surge shaft is modelled by equations 5.12 and 5.13 which, in per-unit terms are:-

$$\overline{Q}_s = \overline{Q}_d - \overline{Q}_a \qquad 6.22$$

and:-

$$\frac{d\overline{H}_s}{dt} = \frac{Q_o}{A_s H_o} \overline{Q}_s \qquad 6.23$$

Similarly, the conduit between the reservoir and the surge shaft is represented by the per-unit equation derived from equation 5.14:-

$$\frac{d\overline{Q}_d}{dt} = \frac{F_d H_o}{Q_o} * (\overline{H}_r - \overline{H}_s) \qquad 6.24$$

Defining, for a specific pipe section:-

$$F_o = \frac{FH_o}{Q_o} \qquad 6.25$$

and, similarly defining:-

$$K_o = \frac{KQ_o}{H_o} \qquad 6.26$$

the complete hydraulic system can now be modelled using the above equations with the values of the parameters derived as below.

A GUILDS function (PIPE) is used in the simulation model to represent each pipe section described by equations 5.5 to 5.8. This function is translated by the STAGE2 processor into four first order differential equations. The inputs to the function are the head at the upstream end, the flow at the downstream end, the values of K_o and F_o for the section and initial conditions for the output variables. The output variables are the head at the downstream end and the flow at the upstream end of the section.

From equations 5.9 and 6.25:-

$$F_o = \frac{gAH_o}{LQ_o} \qquad 6.27$$

and, similarly, from 5.10 and 6.26

$$K_o = \frac{a^2Q_o}{gLAH_o} \qquad 6.28$$

thus values for F_o and K_o can be calculated from the length (L), cross-sectional area (A) and wave velocity (a) for each pipe section.

The wave velocity (a) for a particular conduit is dependent upon the construction of the conduit. The conduits being represented in the simulation are mainly rock tunnels and steel pipes and Table 6.1 gives the values for ρ, f, B, and E used in equation 5.11 to calculate the values of the wave velocity for these conduits.

Conduit	$\rho(Kgm^{-2})$	f	$B(GNm^{-2})$	$E(GNm^{-2})$	$a(ms^{-1})$
Tunnel	1000	2	2.1	11	1234
Pipe	1000	70	2.1	207	1108

Table 6.1 - Conduit Parameters for Wave Velocity Calculation

The stress factor (f) for the steel pipe is calculated assuming typical values for the diameter (D) and wall thickness (e) since the thickness of the pipe increases with the head down the pipeline and the internal diameter decreases accordingly:-

$$f = \frac{D}{e} = \frac{2.04}{0.029} = 70. \qquad 6.29$$

As the variation of the thickness and diameter of the pipes was not known exactly it was decided to use only one wave velocity for the steel conduits although three different diameters were used (see below).

The physical dimensions of the different conduits, indicated in Figure 6.5 are listed in Table 6.2. Where a pipe section represents different physical conduits, effective areas and wave velocities are calculated from the values for each conduit, weighted with respect to the length of the conduit. Section C, for example, referring to Figure 6.5, has a total length L_c given by:-

$$L_c = 181.6 + 23.2 + 338.3 + 140.5$$
$$= 683.6m$$

and the effective area is the sum of cross-sectional area of each conduit times the length of the conduit divided by the total length of the section:-

$$A_c = (181.6*7.31 + 23.2*3.46 + 338.3*3.24 + 140.5*2.93)/683.6$$
$$= 4.26m^2$$

and similarly, the effective wave velocity is given by:-

$$a_c = (1234*181.6 + 1108*502)/683.6$$

$$= 1141.5 ms^{-2}$$

Table 6.3 lists the effective values for area and wave velocity used in the simulation for the single pipe section model and for the multiple pipe section representation. Table 6.4 lists the values of K_o and F_o for each pipe section calculated from these figures.

Conduit	Construction	Length (m)	Diameter (m)	Area (m^2)	Wave Vel. (ms^{-1})
1	Lined Tunnel	218	4.12	13.33	1234
2	Rock Tunnel	2480	4.82	18.25	1234
3	Lined Shaft	-	7.92	49.32	-
4	Rock Tunnel	166.7	4.82	18.25	1234
5	Lined Tunnel	181.6	3.05	7.31	1234
6	Lined Tunnel	184.4	3.05	7.31	1234
7	Steel Pipe	23.2	2.1	3.46	1108
8	Steel Pipe	388.3	2.03	3.24	1108
9	Steel Pipe	140.5	1.93	2.93	1108

Table 6.2 - Physical Dimensions of Conduits

Pipe Section	Total Length (m)	Effective Area (m^2)	Effective Wave Vel. (ms^{-1})
Single Pipe	850.3	7.0	1159.6
A	166.7	18.25	1234.0
B	686.4	7.06	1141.9
C	683.6	4.26	1141.5
D	2689.	17.92	1234.0

Table 6.3 - Effective Dimensions for Pipe Sections

Pipe Section	K_o	F_o
Single Pipe	1.22	1.52
A	2.71	20.24
B	1.46	1.90
C	2.41	1.15
D	-	1.23

Table 6.4 - Pipe Section Parameters (K_o and F_o)

6.3.5 Relief Valve

The flow through the relief valve is given by equation 5.25. From the manufactures discharge curves for the relief valve the maximum area of the orifice is $1.3m^2$ and, at a head of 266.7m, the corresponding flow is $17.27m^3s^{-1}$. Subsituting these values in equation 5.25 gives:-

$$17.27 = 1.3C_{drv}\sqrt{2*9.81*266.7}$$

$$\Rightarrow \quad C_{drv} = 0.184 \qquad 6.30$$

Assuming C_{drv} is constant under all operating conditions, then:-

$$Q_{rv} = 0.184A_{rv}\sqrt{2gH_t} \qquad 6.31$$

and taking $A_{rvo} = 1.3m^2$ as the base value for the relief valve area then, in per-unit terms:-

$$\overline{Q}_{rv} = \frac{0.184*1.3}{14.15}\overline{A}_{rv}\sqrt{2*9.81*266.7\overline{H}_t}$$

$$\Rightarrow \quad \overline{Q}_{rv} = 1.22\overline{A}_{rv}\sqrt{\overline{H}_t} \qquad 6.32$$

and assuming that the position of the relief valve servo (y_r) is proportional to the area of the valve opening then:-

$$\overline{Q}_{rv} = 1.22\overline{y}_r\sqrt{\overline{H}_t} \qquad 6.33$$

The position of the relief valve is determined by the equations given below: if the rate of change of servo position (z) in the closing (negative) direction is greater than a threshold rate (r_t) then the relief valve linkage moves at a rate given by:-

$$z_1 = -z - r_t \qquad 6.34$$

and the relief valve linkage position, which is limited between 0 (closed) and 1 (fully open) is given by:-

$$y_1 = z_1/s \qquad 6.35$$

Representing the operating mechanism of the relief valve as a first order lag, with an effective time constant T_r, then the relief valve position is given by:-

$$y_r = \frac{1}{1 + sT_r} y_1 \qquad 6.36$$

If the servo closing rate falls below the threshold then the relief valve starts to close at a rate (r_u) determined by a timing screw in the operating mechanism. In addition, if the servo starts to open again, the rate of reclosure is increased by the rate at which the servo is opening. The reclosure rate is limited by a throttle valve in the hydraulic pipework. Thus:-

$$z_1 = -z - r_t \qquad \text{for } z < 0 \qquad 6.37$$

$$z_1 = -z - r_u \qquad \text{for } z > 0 \text{ and } z_1 > -r_m \qquad 6.38$$

and

$$z_1 = -r_m \qquad \text{for } z_1 < -r_m \qquad 6.39$$

As noted in Chapter 5, the relief valve operation is highly non-linear and largely unrepeatable but from observations made during site tests and also from Reference 1, values were arrived at as below:-

Threshold Opening Rate (r_t) = 0.025pu/s

Unforced Reclosure Rate (r_u) = 0.01pu/s

Maximum Reclosure Rate (r_m) = 0.025 pu/s

Operating Mechanism Time Constant (T_r) = 0.2s

6.3.6 Hydraulic Servo System

The equations used in the simulation of the hydraulic servo system were derived directly in per-unit terms from observations of the operation of the system and not from any theoretical relationships. Figure 6.6 shows, in block diagram form, the representation of the servo system used in the simulation model.

The error signal (y_e) resulting from the difference between the desired (y_d) and actual (y) servo position signals operates an electro-hydraulic distribution (Moog) valve controlling the flow of oil in the main servomotor which positions the water control valve accordingly. The main servomotor acts as an integrator and the system has an overall gain due to both electrical and hydraulic amplification. The Moog valve and the hydraulic pipework introduce lags in the system which are modelled as a lumped first order lag with a time constant T_s. The rate of change of servo position is restricted and this is represented by limits on the input to the main servomotor, r_o and r_c for opening and closing respectively. The backlash (see section 5.3) between the servo and the water control valve is represented by the GUILDS hysteresis function (HSTRSS) and the hysteresis is assumed to be symmetrical about any given servo position.

The per-unit equations resulting from this empirical approach are:-

$$y_e = y_d - y \qquad 6.40$$

$$x_4 = \frac{G_s}{1 + sT_s} y_e \qquad 6.41$$

$$z = x_4 \; ; \qquad r_c < x_4 < r_o \tag{6.42}$$

$$z = r_c \; ; \qquad x_4 < r_c \tag{6.43}$$

$$z = r_o \; ; \qquad x_4 > r_o \tag{6.44}$$

$$y = z/s \tag{6.45}$$

$$y_c = f(y) \tag{6.46}$$

where f(y) is the HSTRSS function.

From measurements on site and also with reference to Bryce[1] and Findlay[3] the values used in the simulation studies for the parameters in the above equations were:-

T_s = 0.1s

G_s = 3.0pu

r_o = 0.04pu/s

r_c = 0.25pu/s

Hyst = $\pm$0.007pu

The values of G_s and T_s were chosen so that the simulation results agreed with the response of the servo on site. By subjecting the servo to small steps, avoiding the rate limits, it would be possible, from the response of the servo, to obtain actual site values[14], if the values in the simulation were not sufficiently accurate.

The opening rate limit was set up using an adjustable flow restricter valve during commissioning tests on site[6] to the value given. This limit is required to prevent a vacuum being created in the pipeline when the servo is opened as there is no immediate increase in the flow into the pipeline from the surge shaft to match the increased

flow through the water control valve. The closing rate limit is a feature of the construction of the Moog valve and this rate was measured from the results of emergency stop tests carried out on site.

The magnitude of the hysteresis was measured by injecting a low frequency (0.1Hz) signal into the servo and increasing the amplitude of the signal until movement of the guide vanes of the water control valve was detected. Although this value proved satisfactory for the majority of the simulation studies carried out, it was found necessary to alter this value slightly for some studies, as noted in the appropriate Documentation Files.

6.3.7 Non-linear Function

As noted in Chapter 5, the power output from the generator (P_e) is related to the servo position (y) by a non-linear function. Figure 6.7 shows a typical site result recorded as the set was unloaded over a long period of time, to minimise the effect of surge shaft oscillations. The head on this occasion was 262.7m which is considered to be sufficiently close to the base value for this curve to be used to derive a per-unit non-linear function.

The non-linear relationship is most likely a combination of a number of non-linearities, for example, in the mechanical linkages between the servo and the guide vanes, in the discharge coefficient of the water control valve and in the efficiencies of the turbine and the generator. However, considering the non-linearities to be lumped together as a single non-linearity between the water control valve position (y_c) and the effective area of the valve ($A_e = A_{cv}C_d$) and using the linear expressions already derived it should be possible to obtain a expression of the non-linear relationship which could then be included in the simulation model.

Now, neglecting hysteresis, the x-axis of Figure 6.7 can be considered as equivalent to be water control valve position (y_c). The y-axis, per-unit electrical power (P_e), is also per-unit mechanical power (P_m) as a direct result of equation 6.15. Similarly, this axis can be recalibrated in terms of per-unit water power (P_w) through equation 6.16, which reflects the change in efficiency with flow, to give a graph of P_w against y_c as shown in Figure 6.8. Since, at constant (base) head, flow (Q_t) is equal to water power in per-unit terms (equation 6.9) and area (A_e) is equivalent to flow (equation 6.20), then Figure 6.8 is effectively a graph of A_e against y_c.

The site result was obtained by unloading the set to the no load point, thus, no data are available for the operation below this point. It was decided to obtain a best fit function for the main section of the curve and approximate the section below the no load point by a quadratic having the same gradient as the function at the no load point and passing through the origin (that is zero flow for zero servo position).

The points on the curve were supplied to a curve fitting program on the PDP-11 computer and the resulting third order function was:-

$$A_e = 0.0543 + 0.1351y_c + 1.986y_c^2 - 1.171y_c^3 \qquad 6.47$$

The gradient of this function (dA_e/dy_c) at the no load point ($y_c = 0.12$, $A_e = 0.1$) is 0.56. The quadratic function:-

$$A_e = A_0 + A_1y_c + A_2y_c^2 \qquad 6.48$$

for the section of the curve below the no load point is defined by the origin, the no load point and the gradient of the non-linear function at the no load point. Thus:-

$$A_e = 1.057y_c - 2.065y_c^2 \qquad 6.49$$

and equations 6.47 and 6.49 define the non-linear relationship between effective area and water control valve position for the complete range of the control valve position. Figure 6.9 shows the generated function; the section of the curve shown as a broken line corresponds to the quadratic function (equation 6.49).

To evaluate the initial condition for servo position (y_c) for a specified electrical load (P_L), the inverse of the non-linear function is required. To avoid any inaccuracies which could arise from using an inverse function, an iterative technique is used which forms part of the INITIAL segment of the Model Description.

6.3.8 Governor

The governor equations presented in Chapter 5 are dimensionless and thus can be used directly in the simulation. Table 6.5 lists the values of the parameters used in the simulation, most of the values being obtained from References 1 and 3. No attempt has been made to optimise the governor parameters during the simulation studies as the development of a model to reproduce and predict site results with the existing governors was considered of primary importance. It was found to be necessary to adjust the values of some of these parameters during the simulation studies in order to obtain agreement with the site results. The values used for a specific study can be found by reference to the appropriate Documentation File in Appendix 3.

Temporary Droop Governor Parameters

$T_y = 0.3$

$T_d = 16.0$

$b_t = 0.25$

$b_p = 0.03$

Double Derivative Governor Parameters

$T_3 = 0.2$

$T_4 = 0.2$

$T_y = 1.0$

$K_1 = 3.0$

$K_2 = 2.3$

$b_p = 0.03$

$T_L = T_y/b_p = 33.3$

Double Derivative Parameters for Temporary Droop Representation

$T_3 = 1.018$

$T_4 = 0.0$

$T_y = 4.75$

$K_1 = 14.99$

$K_2 = 0.0$

$b_p = 0.03$

$T_L = T_y/b_p = 158.29$

Table 6.5 - Governor Parameters

The double derivative governor transfer function, as a series of first order differential equations, with the servo set point (y_s) included, becomes:-

$$f_e = f_s - f \qquad 6.50$$

$$z_1 = (f_e - x_1)/T_3 \qquad 6.51$$

$$x_1 = z_1/s \qquad 6.52$$

$$z_2 = (z_1 - x_2)/T_4 \qquad 6.53$$

$$x_2 = z_2/s \qquad 6.54$$

$$x_3 = (K_1 z_1 + K_2 z_2 + f_e)/b_p \qquad 6.55$$

$$z_g = (x_3 - y_g)/T_L \qquad 6.56$$

$$y_g = z_g/s \qquad 6.57$$

$$y_d = y_g + y_s \qquad 6.58$$

On site, the desired servo position signal (y_d), as output to the servo, and the servo set point (y_s) and load limit (y_{LL}) signals as recorded on the data logger, are all scaled such that 0 to 100% is equivalent to -5 to 105%. This was done originally to ensure that the servo would not drift from the end-stops when either the fully open or fully closed position was demanded. However, this had the effect of introducing an additional gain into the governor loop and it was thus necessary to scale the variables in the simulation model in a similar manner.

The output of the governor is limited between $-y_s$ and $y_{LL}-y_s$ such that the desired servo position signal ($y_g + y_s$) is always in the range 0 to y_{LL}. This ensures that, under circumstances where the governor is demanding a large positive or negative output, any change in governor output in the opposite direction will have an immediate effect on the servo position. These limits were included in the simulation model to give as accurate a representation of the site governor as possible.

As noted in section 6.3.6 there are hydraulic rate limits on the main servomotor which are necessary for the safe operation of the station. As an additional precaution the output of the governor (y_d) is also rate limited to nominally the same values as in the hydraulic system. However as these values were not necessarily identical it was decided to include both sets of limits in the simulation model. The hydraulic rate limit was implemented using the GUILDS LIMIT function to limit the input to the main servo integrator and the function RATLIM was used to rate limit the governor output.

The servo set point signal (y_s) is rate limited in the closing direction so that, under normal operating conditions, the relief valve does not operate as the set is unloaded. In the opening direction the rate limit is approximately equal to the servo rate limit to permit the set to be loaded as quickly as possible without endangering the pipeline. On site, it was found that the relief operated not only when the rate of change of servo position exceeded the threshold but also when a constant rate (below the threshold) was maintained for more than about 50% of full stroke. As a compromise, the closing rate was set at 0.0167p.u./s (60s for 1 p.u., full stroke) but, as this rate was still sufficiently fast to operate the relief valve if the servo was closed from fully open, it was recommended that the set be unloaded in at least two stages.

The values given above are those currently used in the operational governor but, as most of the site results were gathered before or during the the commissioning tests in which the values were set up, various different values have been used in the simulation model. The documentation files for the simulation studies (Appendix 3) give the values used for these rates for each simulation run.

6.4 Additional Equations Required for the Simulation Model

In addition to the equations presented in the previous sections which form the basic model of the hydro-generator at Sloy Power Station there are a number of other equations which are necessary to permit the model to replicate certain specific conditions. These equations are given in the following sections.

6.4.1 Isolated Load Simulation

In order to test the closed loop response of a governor on site, without the need to arrange actual isolated load tests[1], an isolated load simulator (ILS) was installed on site as part of the experimental equipment[18]. The use of an ILS was first proposed by Schlief[45] and is also discussed by Causon[46]. References 3 and 4 describe, in detail, the ILS used on site in Sloy Power Station and thus only a briel outline is given here.

Figure 6.10 shows a block diagram of the isolated load simulator. From this Figure it can be seen that the ILS is similar to the representation of the generator and load used in the simulation model (sections 5.1 and 6.3.1) except that a load self-regulaltion factor (e_n) has been included. This takes account of the variation of the load power with frequency, which is important under isolated load conditions.

It can be shown that[4]:-

$$F_L = P_{Lo}(1 + k_n f_e) + \Delta P_L \qquad 6.59$$

where F_L is the load torque

P_{Lo} is the steady state load power

ΔP_L is a small step change in load

f_e is the frequency error

and $k_n = e_n - 1$ 6.60

which results in the simplified block diagram shown in Figure 6.11.

When using the ILS, the generator remains connected to the grid but deadband is introduced into the frequency signal to prevent the governor responding to changes in grid frequency, which is assumed to remain constant at 1 p.u. throughout the tests (although this is not always the case). The frequency error signal (f_e) generated by the ILS is fed to the governor, notionally in place of the grid frequency error but, in fact, in addition to the frequency error signal output from the deadband, which is retained for operational safety.

It was necessary to simulate the operation of the set under simulated isolated load conditions in order to establish any differences between simulated and true isolated load operation and to confirm that, when using the ILS, satisfactory results could be obtained which would be valid under true isolated load conditions. Also, the behaviour of proposed governors under isolated load operation could be demonstrated before site tests were conducted.

Operation with the isolated load simulator differs from true isolated load operation in that the generator and turbine are operating at a constant speed which does not change with the frequency of the (simulated) isolated load. Thus, in the simulation of ILS operation, it is necessary to remove the speed dependence from equations 5.21 and 5.23 which, in per-unit terms become:-

$$\overline{F}_m = \overline{Q}_t \overline{H}_t \eta_t \qquad 6.61$$

$$\overline{Q}_t = \overline{A}_e \sqrt{1.584\overline{H}_t - 0.584} \qquad 6.62$$

Thus, by including equations 6.59, 6.61, and 6.62 in the simulation model, the performance of the hydro-generator operating with the ILS can be simulated.

6.4.2 Frequency Transducer

The frequency input to the governor is provided by a frequency transducer, described in detail in Reference 3 and Appendix 1. The input to the transducer is derived from the generator terminal voltage and current such that, even under fault conditions, a signal is always available. During the run up sequence, the residual magnetism of the rotor is sufficient to provide a signal for the transducer.

The output from the transducer is a highly non-linear signal derived from the period of the incoming waveform. Within the range 45 to 55Hz (0.9 to 1.1 p.u.) the signal is sufficiently linear to provide an accurate measurement of frequency[3]. Outwith this range the output of the transducer is held at one of two constant levels, depending on whether the frequency is above or below the linear region, and below 45Hz comparators are used to provide discrete "speed signals" which are used in the run up sequence. The design of the transducer was based on the mechanical TD governor which has similar features.

Originally, the output of the transducer was set at a voltage corresponding to 0Hz (0 p.u.) for frequencies below 45Hz and corresponding to 100Hz (2 p.u.) for frequencies above 55Hz thus ensuring rapid governor action in the event of under or over speed conditions arising. However, the step change in the frequency signal from 0 to 45Hz during the run up sequence caused the derivative terms in the governor to overwhelm the proportional action, resulting in a transient closing of the servo.

To overcome this problem the lower frequency limit was set to 35Hz (0.7 p.u.) and, in the governor, the action of the derivative terms was limited below about 47Hz. Subsequently, a further modification was made to the governor equations[6] whereby the derivative

terms were held at zero until the frequency reached 49Hz. Hysteresis was included so that the derivative terms would not be reset to zero until the frequency fell below 46.25Hz.

It was anticipated that similar problems would occur during a load rejection if the frequency exceed 55Hz. The derivative action resulting from the step change in frequency from 55 to 100Hz as the machine accelerated through 55Hz would aid the proportional action of the governor. However, as the machine decelerated and the frequency fell below 55Hz the step change from 100Hz to 55Hz would cause the derivative terms to overwhelm the proportional action as before, causing a transient opening of the servo in this case. To prevent this occuring the frequency signal to the derivative terms was limited within the governor to 54.75Hz.

As a direct result of the way in which the analogue to digital convertors in the microprocessor governor were used, the frequency signal into the governor was finally limited to 43.75Hz (0.905 p.u.) for frequencies below 45Hz and to 56.25Hz (1.095) for frequencies above 55Hz, although, within the governor, frequencies outwith this range are set to 35Hz (below 45Hz) and 100Hz (above 55Hz).

Three GUILDS functions (details of which are to be found in the GUILDS User's Guide) are used in the simulation model to represent the frequency transducer and the associated modifications to the governor algorithm: STPLIM is used to provide the overall characteristic (f_t) from the input frequency signal; ANDHYS is used to switch the derivative terms in (at 49Hz) and out (at 45.25Hz); LIMIT is used to limit the the frequency signal to the derivative terms (f_d) to 54.75Hz.

The additional equations required are:-

$$f_t = \text{STPLIM}(q_1, q_2, q_3, q_4, f) \quad 6.63$$

$$f_d = \text{LIMIT}(0., q_5, f_t) \quad 6.64$$

$$f_{ed} = f_s - f_d \quad 6.65$$

$$s_d = \text{ANDHYS}(0., f_t - q_6, f_t - q_7) \quad 6.66$$

$$K_{12} = K_1 s_d \quad 6.67$$

$$K_{22} = K_2 s_d \quad 6.68$$

where q_1 to q_7 are the frequency transducer break points. Equations 6.50, 6.51 and 6.55 require to be modified as below:-

$$f_e = f_s - f_t \quad 6.69$$

$$z_1 = (f_{ed} - x_1)/T_3 \quad 6.70$$

$$x_3 = (K_{12} z_1 + K_{22} z_2 + f_e)/b_p \quad 6.71$$

Since many of the site results were obtained as the modifications to the governor and the frequency transducer, detailed above, were being made the values of the parameters for these functions may differ from model to model, depending on the site conditions being simulated. The values for a particular simulation are to be found in the appropriate Documentation File in Appendix 3.

6.4.3 Run-Up Simulation

The run-up on site is determined by a sequential control unit and the governor only comes into action at or near rated speed. Thus, for governor development, it is only necessary that general agreement should be obtained between the simulation model and site results during run-up. However, it is more important that the behaviour of the model as the frequency approaches 50Hz is close to the that of the hydro generator.

Equation 6.18 gives the relationship between the electrical torque produced by the generator (F_e) and the power in the water (P_w) and whilst this expression is valid for operation around rated speed (f = 1 p.u.) it is clearly not valid near zero speed (since this implies an infinite torque). To obviate this problem, an arbitrary but qualitatively acceptable relationship was derived to replace equation 6.18.

For this arbitrary relationship, torque, at low speeds, was assumed to be proportional to flow, and to provide a simple representation of the effects of static friction, the torque was held at zero until the flow exceeded a threshold value (Q_1 = 0.1). Thus, at low speeds, equation 6.18 can be replaced by the relationship:-

$$F_{e1} = kQ_t - Q_1 \qquad 6.72$$

At some point in the run up sequence, as the speed of the set increases, it is necessary to transfer from equation 6.72 to equation 6.18 and at this point there should be no discontinuity in the torque. It was decided to make the transition at 40Hz (0.8 p.u.) so that the changeover would not occur within the normal operating region (45 to 55Hz) and, thus, equation 6.72 would only be required for run up simulations. Now, since equation 6.72 must give the same value of torque as equation 6.18 at 40Hz (f = 0.8) it can be shown that:-

$$F_{e1} = 1.388H_tQ_t - 0.0888 \qquad 6.73$$

and thus, the electrical torque (F_e) is given by:-

$$F_e = 0. \quad ;F_{e1} < 0., f < 0.8 \qquad 6.74$$

$$F_e = F_{e1} \quad ;F_{e1} > 0., f < 0.8 \qquad 6.75$$

$$F_e = 1.11P_w/f - 0.111f \quad ;f > 0.8 \qquad 6.76$$

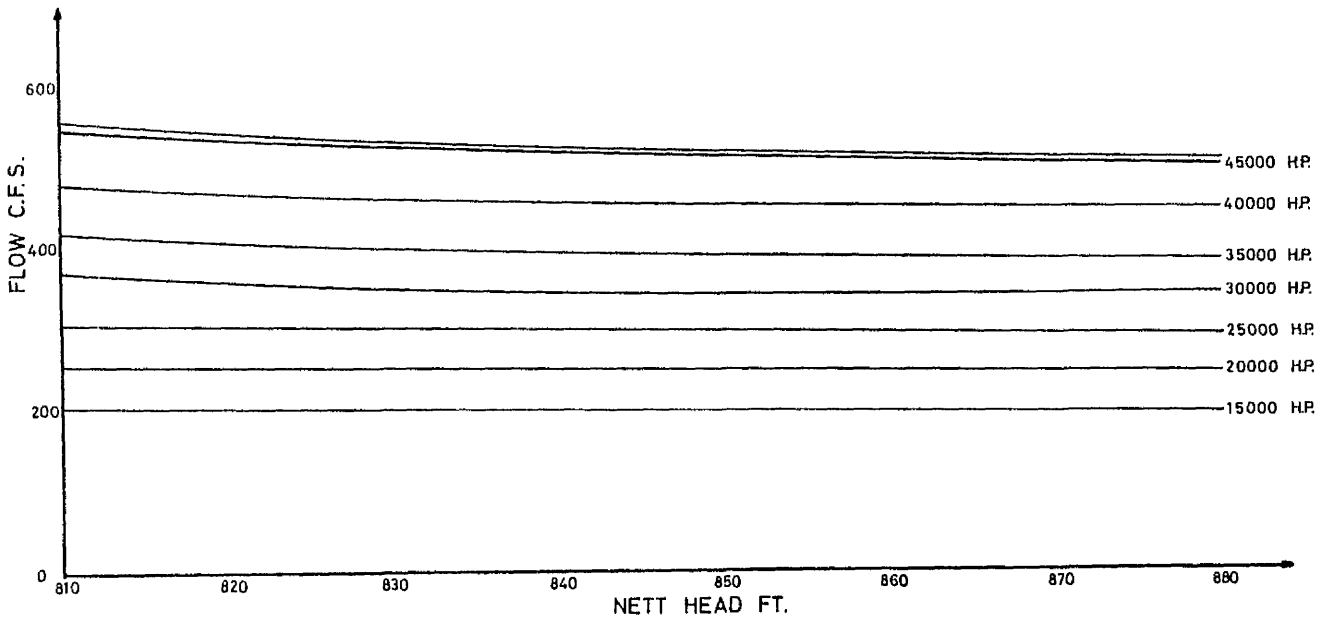


Figure 6.1 - Predicted Turbine Performance Curves

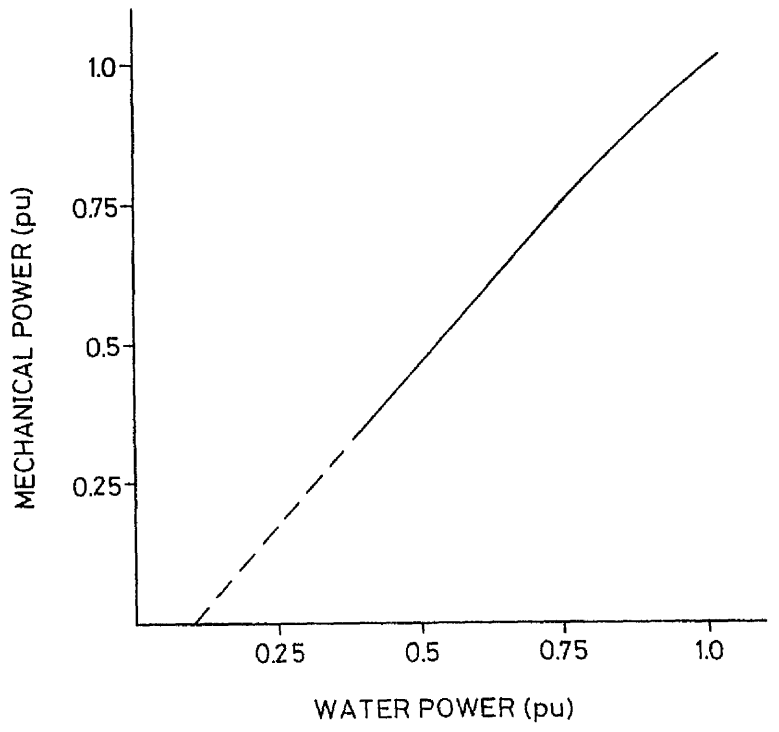


Figure 6.2 - Efficiency Characteristic at 1p.u. Head and Rated Speed

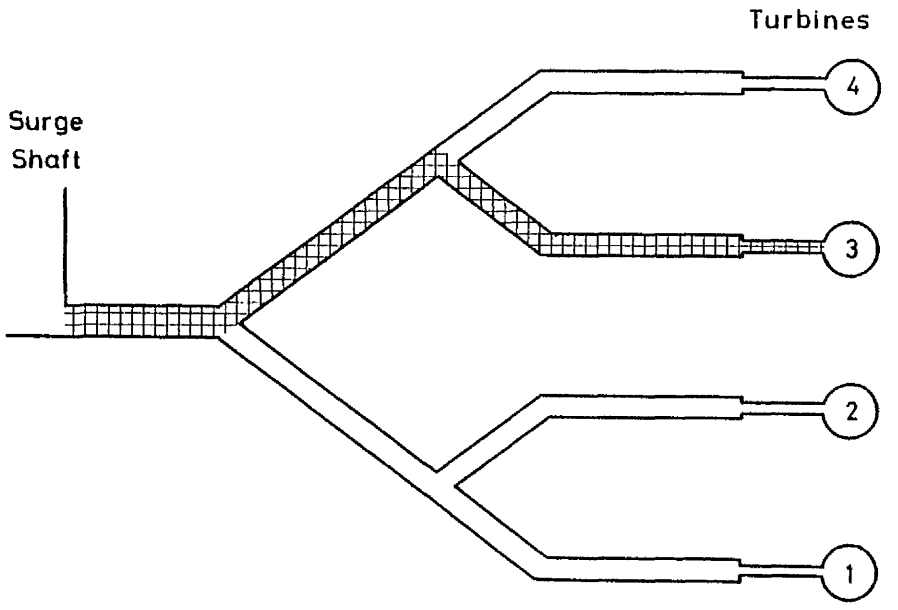


Figure 6.3 - Single Pipe Section Model

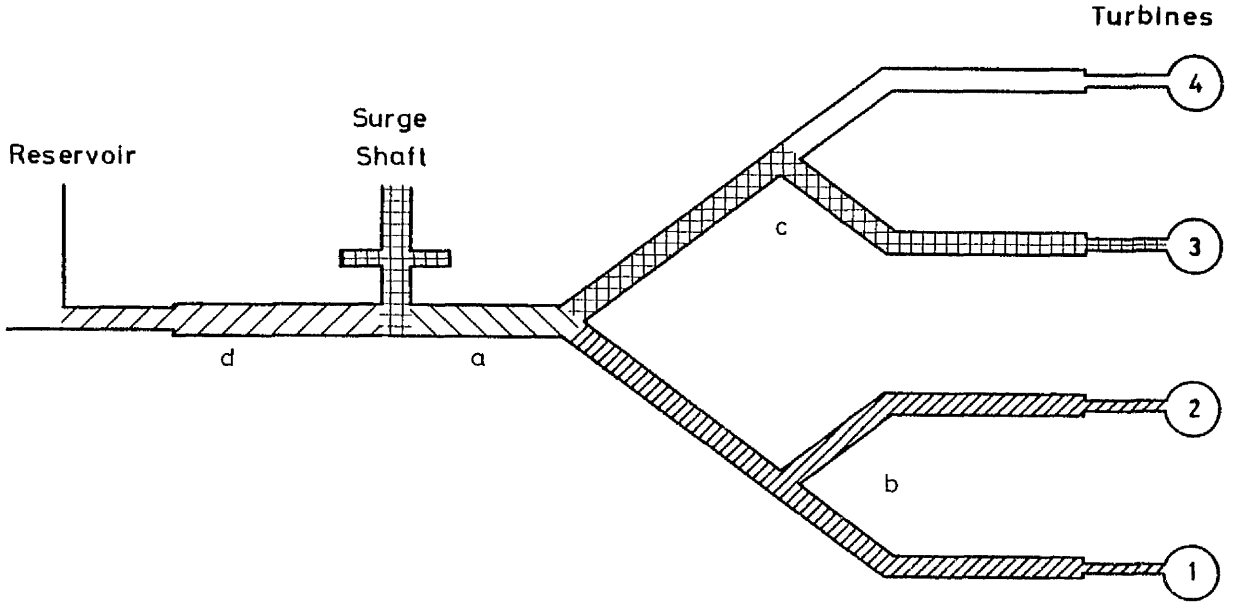


Figure 6.4 - Three Pipe Section Model with Reservoir and Surge Shaft

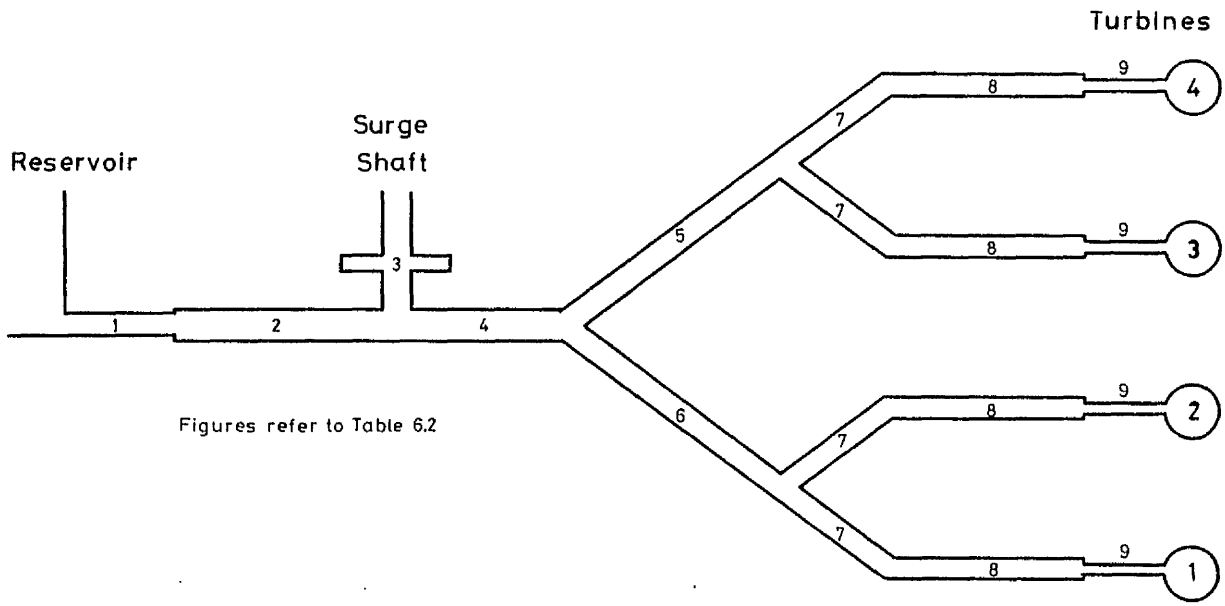
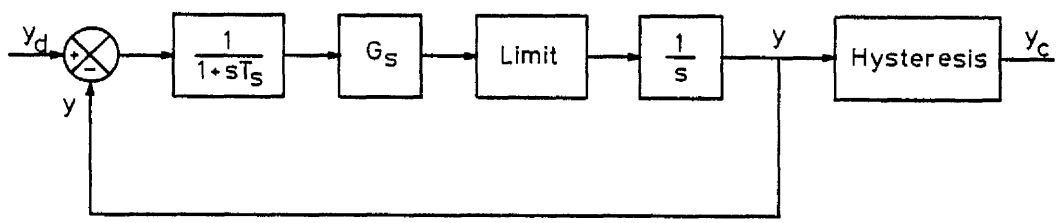


Figure 6.5 - Pipeline Arrangement



y_d - Desired Servo Position
y - Servo Position
y_c - Water Control Valve Position
G_s - Servo Gain
T_s - Servo Valve Time Constant

Figure 6.6 - Hydraulic Servo System Representation

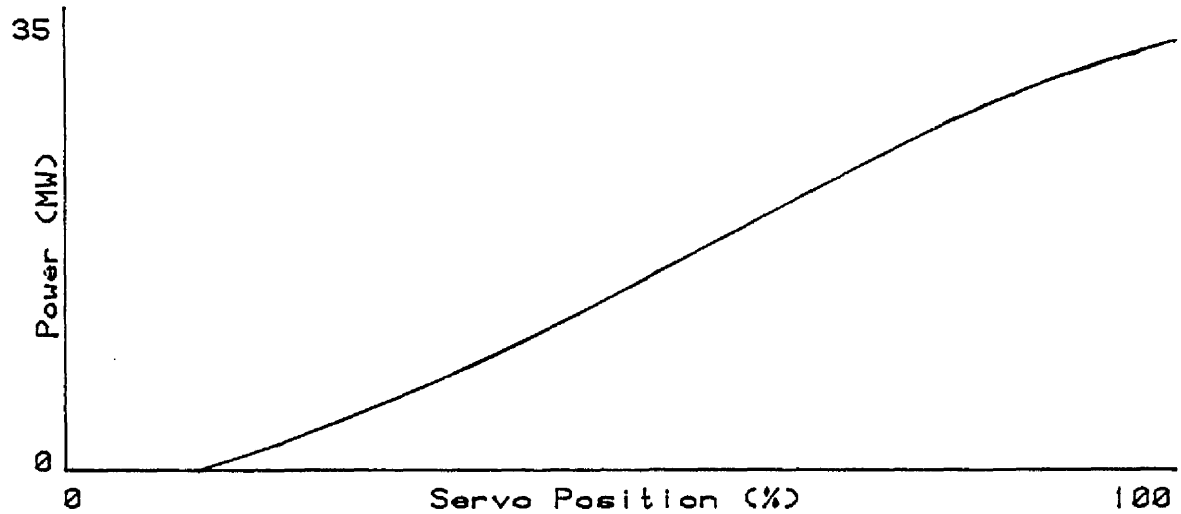


Figure 6.7 - Non-linear Relationship Recorded on Site

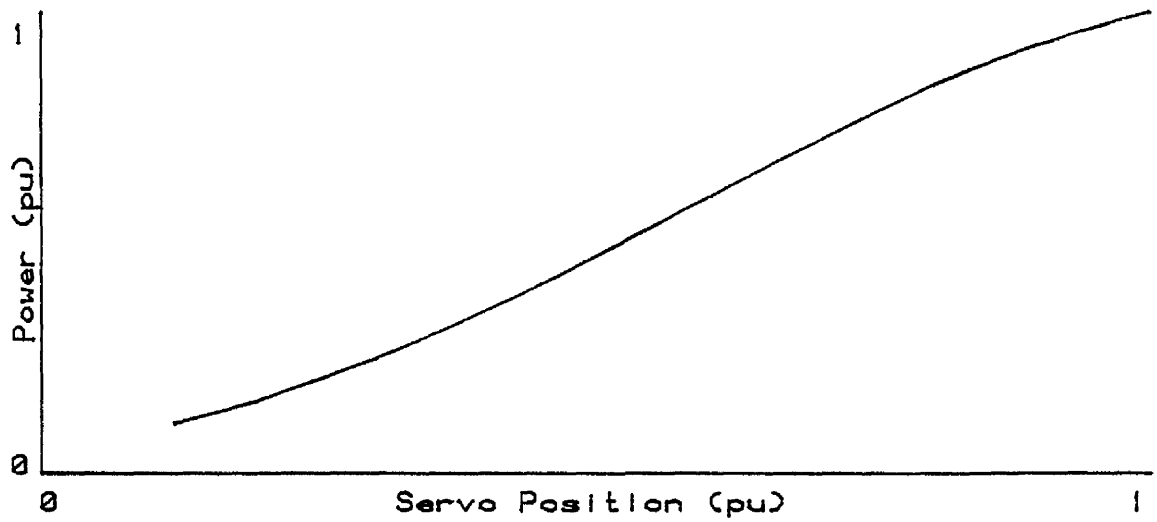


Figure 6.8 - Water Power/Servo Position Relationship

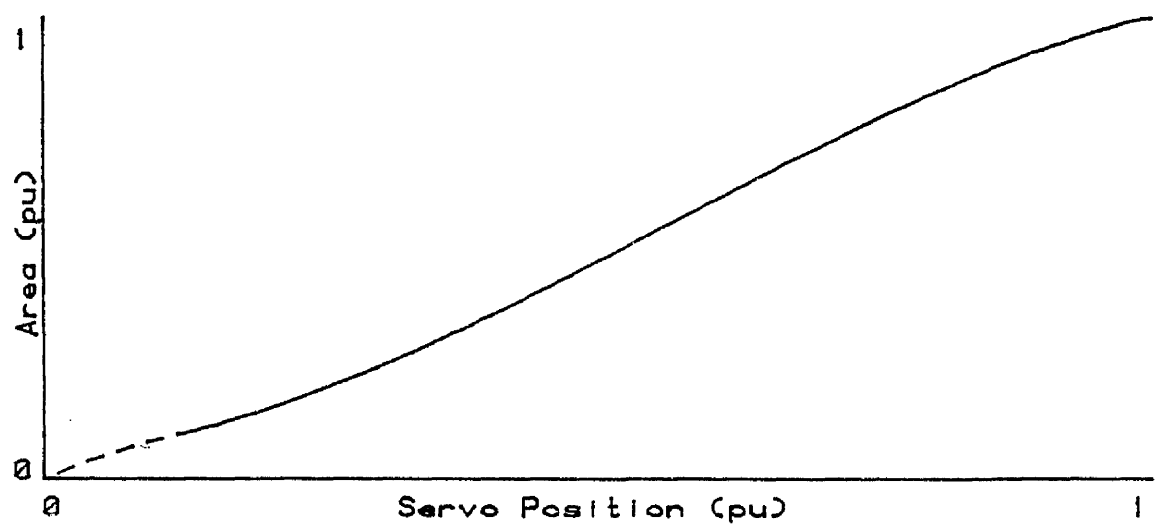
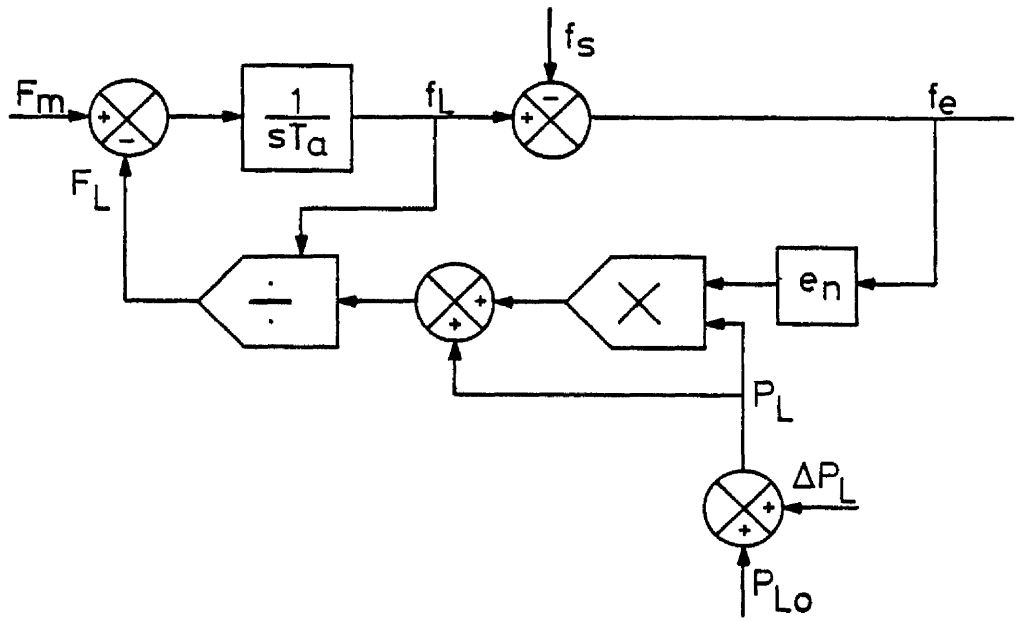


Figure 6.9 - Non-linear Function used in Simulation



f_L - Load Frequency
f_s - Frequency Reference
f_e - Frequency Error
F_m - Turbine Torque
T_a - Alternator Time Constant
e_n - Load Self-regulation Factor

P_L - Load Power
P_{Lo} - Steady State Load
ΔP_L - Change in Load
F_L - Load Torque

Figure 6.10 - Full Representation of Isolated Load Simulator

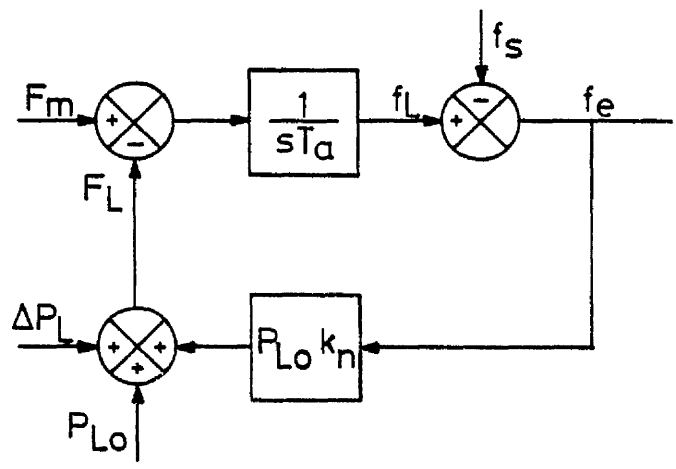


Figure 6.11 - Approximate Representation of ILS as used on Site
($k_n = e_n - 1$)

CHAPTER 7

HYDRO-GENERATOR SIMULATION STUDIES AND SITE TESTS

7.0 Introduction

The work of Bryce[1], Findlay[3] and Grant[6] was mainly concerned with the implementation and on-site testing of various governor types. Each of these authors made use of simulation to some extent, primarily as an aid to developing and proving governor strategies and implementations. Less emphasis was placed on the detailed behaviour of the simulation model than in the present project, the main thrust of which has been to obtain a more detailed agreement between the results of simulation studies and site tests. The resulting model should enable further governor developments be more throughly tested before being installed on site and also, the experience gained on this model will be valuable for modelling other stations, increasing the reliability of the simulation.

The theoretical relationships for the hydro-turbine derived in Chapter 5 and presented in a per-unit form in Chapter 6, along with some additional expressions, form the basis of the hydro generator model used for simulation studies. The results of various studies are presented in this Chapter and, where appropriate, a comparison is made with the results of site tests.

Site tests were carried out at various stages throughout the development of the operational microprocessor governor[5] and, as the governor and associated controller changed quite considerably during this period, several simulation models have been used. Also, it was not always necessary to include all parts of the system (for example the relief valve) in every simulation model. The exact model used for a particular study can be found by reference to the Model Descriptions and appropriate Documentation Files contained in Appendices 2 and 3.

The results presented do not necessarily reflect the chronology of the site tests or the the simulation work but are ordered to give the clearest overview of the development and use of the simulation model in conjunction with on-site testing. The simulation work, following on from Findlay[3,4], started with isolated load studies and frequency response tests (section 7.1). However, to assist with the development of the operational microprocessor governor[6] the run-up (sections 7.2 and 7.4) and load rejection (section 7.3) models were developed before the final isolated load simulations were carried out (section 7.5).

As noted eariler, the models used for simulating the different site results are basically the same although there may be some differences in detail. Inevitably, comparison of simulation and site results for one particular test revealed deficiencies in the model which, when rectified, affected the results of other simulation studies. Where the changes required to a model have had a significant effect on other results, these changes have been noted in the appropriate section.

7.1 Frequency Response Tests

To check the accuracy of the hydraulic system representation, frequency response tests were carried out, initially on the three pipe section model (Figure 6.3), then on the model including the reservoir, surge shaft and connecting tunnel (Figure 6.4). The Model Descriptions for these studies are to be found in Appendix 2.1 and 2.2 respectively.

The frequency of the lowest impedance peak is the most important resonant frequency, below which the governor must operate in order to maintain stability. The three pipe section model was shown early on to exhibit this frequency resonance but further tests were

carried out with the three pipe model and the more complex model, firstly to ensure that the technique being used gave similar results to those of Reference 3 and then to study the effect of the adding the reservoir and surge shaft to the model.

The frequency response tests were carried out by imposing a sinusoidal oscillation on the flow at the downstream end of the pipeline, in place of the turbine and water control valve, and harmonic analysis techniques were used to find the magnitude of the fundamental component of the resulting fluctuations in head. The hydraulic impedance of the pipeline could then be calculated from the ratio of the amplitude of the head to that of the flow.

The GUILDS automatic re-run facility (detailed in section 4.8 of the GUILDS User's Guide at the rear of this Thesis) was used so that the frequency of the forcing function could be scanned, over a prescribed range, and hence a plot of hydraulic impedance against frequency could be obtained. The DYNAMIC section of the Model Description contains a routine to sample the head and flow waveforms a specified number of times per cycle and these values were used subsequently in a TERMINAL section to obtain the harmonic coefficients, from which the amplitude of the fundamental frequency could be calculated, and hence the hydraulic impedance. It was necessary to wait for several cycles (typically five) before sampling to ensure that any initial transient had died down.

Figure 7.1 shows the frequency response for the three pipe section model and in Figure 7.2 the response of the more complex model is shown. As can be seen from a comparison of these two figures the addition of the surge shaft, reservoir and tunnel has little effect on the frequency response characteristic. At the frequencies of interest, the surge shaft isolates the upstream tunnel from the oscillations in

the pipeline and thus can be considered to be of infinite capacity, that is, as if it were the reservoir. Thus, the two models would be expected to exhibit a similar response.

An impedance resonance occurs when a reflected wave returns to the source (Set No. 3) in phase with the forcing function. The phase change experience by a travelling wave is proportional to the frequency of the disturbance and the distance travelled and thus, the longest path length (from Set No. 3 to Sets 1 and 2) would be expected to give rise to the lowest resonant frequency (as stated in Reference 1). However, the lowest frequency in fact corresponds to a reflection from the surge shaft, hence the agreement with the single pipe section model of Reference 3 which ignores the branch to sets 1 and 2.

The surge shaft can be considered as an hydraulic short circuit (a very low impedance compared to the pipeline) which causes the phase of the reflected wave to be changed by 180°. On the other hand, the reflection from Sets 1 and 2, which can be considered as an open circuit when the sets are off, does not involve a phase reversal. Thus, the necessary 360° phase shift is produced by the reflection from the surge shaft at a lower frequency than the reflection from the other sets.

The three lowest frequency response peaks obtained from an impedance chart produced by the NSHEB are shown as a broken line in Figures 7.1 and 7.2. It can be seen that the results obtained from the simulation for the two lowest peaks are in close agreement with the NSHEB results. Higher impedance peaks are not produced by the simulation model due to the simplicity of the representation but, as the lower resonances are of interest, this limitation is not a hinderance. Hence, from a stability consideration, a single pipe representation of the pipeline would have sufficed for the simulation

model. However, the it was decided to maintain the more complex representation so that the effect of the U-tube oscillations between the reservoir and the surge shaft were included in the model and also to permit the addition of a second turbine should a study of the interaction between turbines become necessary.

7.2 Run-up Simulation

The run-up sequence for the generator, from standstill to rated speed is determined by a sequential controller which initially formed part of the analogue control electronics but was later included in the microprocessor governor[6]. The Temporary Droop governor is used during the run-up sequence because this governor has a longer dominant time constant than the Double Derivative governor and thus the frequency of the limit cycles, resulting from backlash in the water control valve linkage, is lower. This enables the set to be synchronised to the grid more quickly by the automatic synchronising unit as the rate of change of frequency is lower.

The broken traces in Figure 7.3 show the results of a typical run-up on site, recorded during tests on site on the 19th of February 1980, details of which are given in Appendix 3.1, while the solid lines represent the corresponding simulation results. The Figure shows the frequency transducer output, the servo position signal (y) and the servo set point (y_s). The desired servo position signal (y_d, the sum of the servo set point and the governor output) was also recorded on site but as this is very similar to the servo position signal it has not been shown.

The servo and load limit set points (y_s and y_{LL}) ramp together from -5% (rest position) to approximately 28% and, although the governor is demanding a very high servo position, the governor output is limited to zero ($y_{LL} - y_s$). As the frequency reaches 25 Hz

the servo and load limit set points are pulled back by the controller to about 22% to give a better transient response as the frequency nears 50 Hz. At about 50Hz, the governor output becomes negative which reduces the desired servo position, closing the servo back and halting the rise in frequency just above 50 Hz.

The frequency transducer, described in Appendix 1, has a discontinuity below 45Hz[3,6] and, since the response of the derivative terms in the governor to this discontinuity gave rise to some problems, the derivative terms were supressed during the run-up. Both the discontinuity and the derivative supression were thus included in the simulation model (section 6.4.2). The frequency at which the derivative terms were activated (49Hz) was arrived at after a series of tests on site to establish the best transient performance as the frequency approached 50Hz. The simulation model of the run-up characteristic had not been developed at this stage, or it would have been possible to "tune" the system without the need for the site tests.

The results of a simulated run-up are shown in Figure 7.3 along with the site results which are represented by the broken lines. The Model Description for the simulation is given in Appendix 2.3 and the corresponding Documentation File can be found in Appendix 3.2.

As can be seen from this figure good agreement has be achieved between the site and simulated run-up characteristics. The discrepancy between the site and simulated servo set point signals is most probably due to errors in scaling in the on-site data logging equipment since the corresponding servo signals are in agreement. The most likely cause of the differences in the frequency transients after the first overshoot is the hysteresis model used in the simulation. This is discussed further in section 7.5 which deals with simulated isolated load simulations.

Agreement between the site and simulation results was only obtained, however, after some initial discrepancies between the two sets of results were resolved. It was noted initially that the simulated run-up took longer than that on site and, it was found that the rate of change of frequency from the simulation, around 45Hz, was considerably lower than that obtained from the site result.

The overall time for the run-up from the simulation model would not necessarily be expected to be in agreement with the site result due to the simplified model used for the torque in the early stages on the run-up (below 40Hz). However, the response at around 50Hz is important and, it was hoped that the simulation results would have been closer to the site response. Inspection of the equations given in Chapter 6 yielded a solution to this problem.

From equation 6.4 the rate of change of frequency (sf) is proportional to the accelerating torque (F_a) and, since the electrical load torque is zero, F_a is equal to the mechanical power output of the turbine which, in per-unit terms is the equal to the electrical torque F_e. Consequently, a low value of sf from the simulation implies a lower value of accelerating torque than on site. From equation 6.18, the accelerating torque is given by:-

$$F_a = 1.11F_w - F_l f \qquad 7.1$$

and thus, for a low value of F_a either the loss torque (F_l) is too high or the water torque (F_w) is too low.

It was quickly discovered that it was not possible to change the loss torque in order to give the correct accelerating torque as the resulting value of loss torque would have been unrealistically low. Thus, it was concluded that the water torque was too low. Now, considering equation 6.9, it can be seen that a low value of torque corresponds to a low value of flow and/or head. Again, it was found to be unrealistic to adjust the head, even allowing for possible errors in

site measurements of loch levels, to give the necessary torque as the required increase in head would have been considerable. From this it was concluded that the differences between the site and simulation were a result of the calculated value of flow being too low.

Considering equations 6.20 and 6.47, it can be seen that the flow is proportional to the effective area of the water control valve (at constant speed) which in turn is a non-linear function of the water control valve position (y_c). It was decided that the most probable source of error resulting in the reduced torque developed by the simulation model would be the non-linear function relating the area of the water control valve to the valve position.

The steady state values of servo position calculated using the simulation model, for given values of power output, were compared with the values obtained from observations on site. It was found that, at low loads, the simulation consistently yielded higher values of servo position than were observed on site. It was thus concluded that the lower portion of the non-linear function, which represents the the electrical power to servo position relationship, was incorrect.

Further analysis of the site data recorded on the day of the test resulted in a new non-linear function, as below. When the run-up was simulated with this function the results, as noted earlier (Figure 7.3) were found to be in agreement with the site result.

$$A_e = 0.0648 + 0.1561y_c + 1.929y_c^2 - 1.166y_c^3 \qquad y_c > 0.095$$

$$A_e = 1.253y_c - 2.852y_c^2 \qquad y_c < 0.095$$

As a further test of the simulation model, it was decided to repeat the simulation for comparison with the run-up recorded on site on 30th of August 1979, as shown by the broken lines in Figure 7.4 and detailed in Appendix 3.3. Note that on this occasion the settings on

the site governor for the servo set point during the run-up were different and also that the derivative terms in the governor were not activated until 50Hz. The head was 261.5m (858ft).

Initial attempts at simulating this site test with these changes in data were unsatisfactory and again, the fault was found to lie in the non-linear function. The function used for the previous simulation was therefore replaced by one derived from power and servo position data recorded on site on the day of the test, as below:-

$$A_e = 0.0598 + 0.2449y_c + 1.707y_c^2 - 1.002y_c^3 \qquad y_c > 0.11$$

$$A_e = 1.256y_c - 2.685y_c^2 \qquad y_c < 0.11$$

The results obtained from the simulation using this non-linear function are also shown in Figure 7.4, and as can be seen, good agreement with the site results (broken lines) has been achieved.

From this study it was concluded that it was not feasible to derive a single non-linear function in per-unit terms, as in section 6.3.7, which would be applicable to all site conditions. Further work would be required to investigate the differences in the power/servo position relationship from one occasion to the next. This would have involved a more detailed and rigorous collection of data on site over an extended period and it was not possible within the time available.

Having developed a simulation model which gave results in close agreement with the observed performance on site it was intended to use the model to tune the controller parameters to give a faster run-up thus enabling the set to be synchronised and loaded up more quickly. However, time did not permit this work to be carried out, and, it later transpired that a feature of the station auto-control scheme required that the turbine main valve be fully open before the

set could be loaded up. This restriction effectively determined the time required for the set to be run-up from standstill to full load and, as the present system enables the set to be synchronised before the main valve is fully open, little advantage could be gained by improving the run-up time. It should be noted that improved run up times obtained by optimising the run-up sequence could be advantageous for other stations where this restriction does not apply.

7.3 Load Rejection Tests

Most of the early work on site carried out by Bryce[1] and Findlay[3] was concerned with investigating the isolated load behaviour of new governor types. Initially, system splitting tests were carried out[1,10] and latterly an isolated load simulator (see section 6.4.1 and Reference 3) was used for testing on site.

Testing governor stability under isolated load conditions was thought to be the most severe test of the governor under worst-case conditions. During commissioning of the operational microprocessor governor, however, a series of load rejection tests were carried out which revealed some deficiencies in the governor[6]. It was thus decided that it would be worthwhile to simulate the load rejection to investigate these problems.

A load rejection occurs when the generator is suddenly disconnected from the load (for example the National Grid) to which it is supplying power. Due to the sudden loss of load, with no corresponding decrease in input power, the machine accelerates rapidly, at a rate determined by the amount of load rejected and the inertia of the machine. Under these circumstances the governor must control the overspeed and return the machine to rated speed as quickly as possible to enable the generator to be re-synchronised to the grid.

The problems observed on site were thought to be associated with the discontinuities in the frequency transducer and it was thus necessary to include these features in the load rejection simulation model, as in the run-up model. Also, the relief valve was known to have a significant effect on the frequency transient resulting from a load rejection and thus, the model of the relief valve derived in section 6.3.5, was included in the simulation.

As noted in Section 7.2, the TD governor is used during run-up, and only after synchronisation is the DD governor selected. A signal derived from the generator circuit breaker is used to select the governor type, thus, if the load rejection occurs as a result of this breaker opening, the governor reverts to TD and, in addition, the servo and load limit set points (y_S and y_{LL}) ramp back to the preset values corresponding to the end of the run-up sequence. If, on the other hand, the load rejection occurs as a result of a remote circuit breaker opening, the governor remains on DD and the set points do not change. As both these conditions were tested on site, the latter by replacing the signal from the generator circuit breaker to the governor by a permanently high signal, it was necessary to included both these options in the simulation. The GUILDS ASK facility was used to select the load rejection conditions,as shown in the Model Description in Appendix 2.4.

Three different load rejections were simulated for comparison with site results, as discussed in the following sections, and an additional site result is included to show the effect of one of the main problems on site which was resolved with the aid of the simulation.

7.3.1 15MW Load Rejection - DD Governor

The broken lines in Figure 7.5 show the results of a nominal 15MW load rejection site test at Sloy on the 19th of February 1980, as detailed in Appendix 3.5. Although this test was carried out by opening the generator circuit breaker, the signal from the breaker was held high to give the effect of a remote circuit breaker operating. Under these circumstances, the governor remains in the DD configuration and, as can be seen from the Figure, the servo set point (y_s) remains at the nominal 15MW setting.

The speed of the turbine, and hence the frequency rises immediately in response to the step change in load as the power in the water being delivered to the turbine is no longer balanced by the electrical load on the generator. The initial rate of change of frequency is determined by the inertia of the machine and the size of the load rejected (equation 6.4). Thereafter, the frequency transient is dependent upon the governor, which acts to close the water control valve in response to the rising frequency, and the relief valve which opens in response to the rapid movement of the main servo-motor. Once the water control valve is closed, the relief valve is considered to have little effect and, since there is no input power (no flow through the turbine) and no load, the frequency falls at a rate proportional to the losses in the machine. As the frequency falls towards 50Hz, the governor responds by opening the water control valve and limit cycling commences until, at some point, not shown in the Figure, the set is re-synchronised to the grid.

The maximum value of the frequency during the transient was 53.8Hz, thus the frequency transducer remains within the linear region (45 to 55Hz). As the frequency approaches the maximum value, however, the second derivative term reaches a positive maximum which tends to open the water control valve. This effect is seen in the transient

"flattening" of the servo and desired servo position signals at about 25%, but is more easily observed in the simulation results which follow.

The Model Description given in Appendix 2.4 was used for simulating load rejection tests. Initial results with this model were in broad agreement with the site results but three important differences were noted, as below:-

(a) the initial servo position for a given power did not correspond to that on site;

(b) the rate of change of frequency immediately after the load rejection was higher than on site;

(c) the rate of change of frequency with the water control valve closed was higher than on site.

Point (a) above was resolved by replacing the per-unit non-linear function derived in Chapter 6 by the function derived from the site data recorded on the day of the test (equation 7.2). Including this function in the simulation resulted in the initial values of servo position being closer to the site values for a given power output.

For the site results shown in Figure 7.5, the initial load setting was taken to be 15MW and the rate of change of frequency was measured to be 2.66Hz/s. Substituting these values in equation 6.4, results in an inertia constant T_a = 8.7s. Other site results, not included in this thesis, gave similar values for T_a.

Now, in Chapter 6, the inertia constant was stated as being 7s. This was the value used by Bryce[1] and Findlay[3] and satisfactory results were obtained using this value in the simulation models. However, no tests were carried out to verify this figure which was assumed to be correct, until this series of load rejections, which formed part of the commissioning tests for the operational governor, enabled an accurate value to be established.

Also from Figure 7.5, the rate of fall of frequency with the water control valve closed was measured to be 0.47Hz/s. Again, substituting in equation 6.4, with T_a = 8.7, the loss torque (F_l) was found, to be, F_l = 0.082pu. In Chapter 6, the loss torque was stated as 0.11pu, but, as was noted at the time, this value was only an estimate as no further information was available. However, it may be reasonable to assume that, with the water control valve closed and the turbine essentially de-watered, the overall losses would be smaller. It was decided, as a first approximation, to assume that the losses were constant at 8.2% during the run, since, the effect of changing from 11% to 8.2% is likely to be small, except when the set is operating at or near low load.

Figure 7.5 shows the response of the simulation model modified as discussed to take account of the points listed above. The Documentation File for this simulation run is to be found in Appendix 3.6.

Comparing the simulation results with those obtained on site, shown as a broken line in the Figure, it can be seen that good agreement between the two sets of results has been obtained. The residual differences in the servo position and servo set point signals are again most likely due to scaling errors in the on-site data logging equipment (in section 7.2) and the simplified model of the losses in the turbine. The problems of obtaining agreement between the site power/servo position data, site test initial conditions, and simulation initial conditions are dealt with in more detail in section 7.5.

In Figure 7.6, the simulated frequency and servo position responses from Figure 7.5 are repeated but, in this case, the first and second derivative terms in the governor have been shown. The effect of the positive peak in the second derivative term on the servo position and hence, the frequency, can be clearly seen from this Figure.

As noted earlier, the effect of the relief valve on the frequency transient during a load rejection is important. To illustrate this, the simulation run of Figure 7.5 was repeated with the relief valve inoperative (Appendix 3.7). Figure 7.7 shows the frequency and servo position transients and the head fluctuations for this case. For comparision, the corresponding variables from the previous simulation run (Appendix 3.6) have been displayed as broken lines in this Figure. Also shown is the relief valve position when operative.

From these two sets of results, it can be seen that the relief valve has the effect of reducing the head at the turbine in the period after the load is rejected. This removes the danger of high pressures building up in the pipeline and also limits the turbine overspeed by reducing the power available to accelerate the machine. It should be noted that on site the actual relief valve position and the head at the turbine were not logged, thus no comparisons with the simulated results are possible. It was observed on site however, that the maximiun opening of the relief valve during this load rejection was 3.5 on a scale from 0 to 10. This figure is close to the maximum value of 36.4% given by the simulation.

The frequency overshoot is reduced when the relief valve operates but, as can be seen from Figure 7.7, the remainder of the frequency transient is not substantially different. The main infulence of the relief valve is on the turbine head fluctuations, reducing the peak head from about 140% to just over 110%. The relief valve can also be shown to enhance the stability of the set by damping oscillations in the hydraulic system[3].

7.3.2 15MW Load Rejection - TD Governor

Figure 7.8 (broken lines) shows the results of a nominal 15MW load rejection carried out at Sloy on the 6th of February 1980, details of which can be found in Appendix 3.8. This test is similar to that discussed in the preceeding section except that, in this case, the actual signal from the generator circuit breaker was used and thus, as soon as the breaker opens the governor reverts to TD and the servo set point (y_s) ramps back to the no-load setting. The rate of the ramp and the no-load setting are both controller parameters[6] and on this occasion were set to 2%/s and 21.4% respectively.

The effect of the change in governor type, from DD to TD is not immediately obvious from the results shown, however, the following points should be noted: the magnitude of the frequency transient has been increased from 53.5Hz to about 55Hz as a result of the slower response of the TD governor; the transient "flattening" of the servo position signal, due to the second derivative in the DD case, does not occur; the fall in frequency below 50Hz during the transient has been increased from 0.5Hz in the DD case to 1.5Hz; the period of the limit cycles after the initial transient has been increased, although this in not shown in the Figure.

The effect of the servo set point ramp initially aids the action of the governor, although, the combined effect of the governor and the ramp is not sufficient to close the water control valve at the maximum rate. In the DD case, the water contol valve closes at the rate limit for the initial section the transient, that is for about half of the closing stroke. If the initial load setting is high enough, as in this case, the servo set point ramp is still active as the frequency approaches 50Hz. Thus, although the governor attemps to open the water control valve, the servo set point ramp effectively counteracts the governor action permitting the frequency to dip

below 50Hz. This can be seen more clearly in the next test (30MW load rejection) in which the water control valve starts to open under governor action but the effect is limited until the servo set point ramp has payed off.

The 15MW load rejection test was simulated using the same model as for the DD governor test in the previous section. Apart from the governor types and the slightly lower initial load setting, the model parameters are identical for the two cases. The non-linear function used was the same as for the DD governor case as no site data was recorded on the occasion of the TD tests from which a new non-linear function could be derived. The Model Description is given in Appendix 2.4 and the Documentation File for this run is to be found in Appendix 3.9.

The results of the simulation run are recorded in Figure 7.8 along with the corresponding site results which are shown as broken lines in this Figure. As can be seen, good agreement has been obtained between the two sets of results.

Some additional information from the simulation run is presented in Figure 7.9 to enable the effect of the servo set point ramp on the governor output to be observed. Along with the frequency signal from Figure 7.8, the governor output (y_g) is shown, which, when added to the servo set point (y_s), gives the desired servo position signal (y_d).

From this Figure it can be seen that, as soon as the frequency reaches its maximum value, the governor output, although still negative, starts to increase. However, the demanded servo position remains at zero until, as the frequency approaches 50Hz, the sum of the servo set point and the governor output becomes positive.

Note that, as the servo set point is decreasing, the governor output would have to increase at a greater rate for there to be any effect on the desired servo position signal before the set point reaches its no-load value.

Had the set point remained at its value before the load rejection the effect of the governor output on desired servo position would have resulted in the servo opening at an earlier point and hence the frequency undershoot would have been reduced. However, since the TD governor is inherently less responsive than the DD governor, the underspeed under these conditions would still have been lower than for the equivalent DD test.

The frequency error resulting from this test is such that the governor output has no effect on the desired servo position until the no-load setting is reached, at which point the servo starts to open. For comparision, the action of the governor during a 30MW load rejection should be noted, as shown in Figure 7.11.

7.3.3 30MW Load Rejection - TD Governor

The results of a nominal 30MW load rejection test, carried out at Sloy on the 19th of February 1980, are shown as broken lines in Figure 7.10 and details of the test are given in Appendix 3.10.

As for the test described in the previous section, the governor reverts to TD as soon as the generator circuit breaker opens, and the servo set point ramps back to the no-load setting, as shown in the Figure.

The frequency transient resulting from this load rejection rose to a maximum of 62Hz (calculated from the maximum speed observed on the revolution counter on the set) and the frequency transducer was therefore operating outwith its linear region (section 6.4.2). Only the frequency transducer output in the linear region was recorded on site, and not the actual frequency, thus the signal shown in the Figure

is limited at 55Hz. For frequencies above 55Hz, the frequency seen by the governor is set to 100Hz (i.e a frequency error of 50Hz) and as a result of this large error signal the water contol valve closes under governor action at the rate limit for most of its travel. It is just possible to observe a slightly slower closing rate at the beginning of the transient which corresponds to the governor action below 55Hz.

As the frequency falls towards 50Hz, the governor responds by demanding an opening of the water control valve, however, as the set point ramp is still active, the effect of the governor action is limited and the frequency continues to fall, although at a reduced rate. Note that in this case the governor output is large enough to have some effect on the servo position, even though the set point ramp is active (c.f. Figure 7.13). When the servo set point reaches the no-load setting, the water control valve opens further and the frequency starts to rise again towards 50Hz. The minimum value of frequency recorded during the test was 46.7Hz.

A simulation of this site test was carried out using the model given in Appendix 2.4. The results of the simulation are presented in Figure 7.10 along with the corresponding site results which are shown as broken lines. The Documentation File for the simulation run is given in Appendix 3.11.

As can be seen from Figure 7.10, the two sets of results are in close agreement for most of the test. There are, however, some discrepancies in the servo and frequency signals from the point at which the servo starts to open, but, given the simplicity of the model used to represent the flow conditions and the losses in the turbine, these differences are remarkably small.

As noted in section 7.3.1, the losses, in the turbine were set at 8.2% for the all the load rejection simulations, as this value gave the same rate of change of frequency for the period when the water

contol valve was closed as that obtained from site results (for both 15MW and 30MW tests). No attempt was made to adjust the losses as the servo opens or take account of the interaction between the water control valve and the relief valve, which is still open at this point. Also, as noted in Chapter 5, the relief valve model is empirical and would not be expected to give detailed agreement with the performace of the relief valve on site.

The frequency signal from the simulation is shown in Figure 7.11 along with the governor output, servo set point and desired servo position signals. The actual frequency, which corresponds to the speed of the set, rises to a maximum value of 61.8Hz which is close to the site value of approximately 62Hz.

The effect of the governor action on the desired servo position signal can be seen from this Figure. As before, when the frequency reaches its maximum value, the governor output starts to increase but it is effectively counteracted by the servo set point ramp. After about 25s, the governor output has risen sufficiently to give a positive desired servo position however the effect is limited until, after a further 12s, the servo set point reaches the no load setting.

7.3.4 Assistance to Governor Development

As noted earlier, load rejections were first carried out as part of the of the commissioning of the operational microprocessor governor. Figure 7.12 shows the results of one of the first load rejection test carried out at Sloy, on the 10th of January 1980. The results are for a 20MW load rejection and the governor type after the circuit breaker was opened was TD. Details of the test are given in Appendix 3.12.

During the test, it became apparent from the change in rate of servo movement that the governor was not responding correctly, resulting in an unacceptably high overspeed, when compared to the mechanical governor. Consequently, all subsequent commissioning test were suspended and efforts made to find out the source of the problem.

Since, in this test, the frequency transient exceeded 55Hz (outwith the linear region of the frequency transducer) for the first time during the commissioning , it was thought that the frequency transducer could be the source of the problem. The simulation model for the load rejection tests, at an early stage of development, was modified to include a detailed representation of the frequency transducer and governor characteristics. The results from the simulation model were found to be similar to the results that had been anticipated from the site governor and did not agree with the actual site behaviour. However, after a series of simulation runs during which a variety of possible errors were introduced, the results demonstrated that the change in the servo closing rate was in fact due to the action of the derivative term in the governor being suppressed as the frequency neared 55Hz. The cause of this problem transpired to be an error in the FORTRAN compiler used in generating the governor program, details of which are given in Chapter 3 of Reference 6.

Once the necessary modifications were made to the governor program, and tested in the simulation model, the governor was re-installed on site and the commissioning tests completed allowing the governor to enter operation service. This problem serves to illustrate the value of simulation as an aid to the development of, in particular, the governor but also, in general, any control system.

7.4 Run-up Simulation - 2

As a result of the load rejection studies discussed above the inertia of the turbine/generator set was taken to be 8.4s. However, the inertia had previously been assumed to be 7s and this value was used by Bryce[1], Findlay[3] and in some of the earlier work of this project. In particular, the run-up simulations discussed in section 7.2 were performed using this value and it was thus necessary to repeat the simulations to examine the effect of increasing the inertia.

Figure 7.13 shows the results of a run-up simulation using the same model as before (Appendix 2.3) but with an inertia constant of 8.7s. The site results of Figure 7.3 are shown in the Figure as broken lines and details of the site test and the simulation run are given in Appendices 3.1 and 3.13 respectively.

As can be seen from the Figure, the run-up characteristic now differs from the site result, both in terms of the time to reach 50Hz and the rate of change of frequency at 50Hz. It was noted in section 7.2 that, due to the simplicity of the torque model at low speeds, the time for run-up from the simulation was not considered to be of great importance. However, it had been hoped that agreement could be obtained between the two sets of results for the frequency transient at around 50Hz. This is no longer the case.

In the time available, no explanation was found for the apparent contradiction between the two different values of inertia required to simulate the run-up and load rejection tests. The techniques used in section 7.2 above were applied but the uncertainties associated with the loss torque at low speeds made this very difficult. It was decided that, within the scope of the present project, the aim of obtaining a single set of data which would be applicable to all operating conditions would have to be foregone.

7.5 Simulated Isolated Load Tests

One of the main areas of concern throughout the development of alternative hydro-turbine governors has been the closed loop stability of the governor should the set be required to supply an isolated load. For this reason, system splitting tests were carried out by Bryce[1,2], and the use of the Isloated Load Simulator (ILS) for testing isolated load operation was developed by Findlay[6]

Following on from the work of Findlay, further simulated isolated load tests were carried out on site to obtain more detailed results for comparision with the simulation model results. A series of step tests were performed at both high and low load settings (25MW and 5MW) with different values of load self-regulation (e_n) to study the effect on the transient response of the set and the subsequent "steady state" limit cycles. The tests were all carried out at Sloy on the 30th of August 1979 and details are given in Appendix 3.14 to 3.16.

Figures 7.14, 7.15 and 7.16 show the results of a 5% (of 32.5MW) step change in load from an initial setting of about 25MW. The values of k_n ($k_n = e_n - 1$) for these tests were 0, 0.5, and 1.0 respectively. The signals shown in the Figures are simulated frequency, the output of the ILS, servo position and the power output from the generator. The broken lines are the results obtained from the site tests, the other results coming from the simulation.

It can be seen from these Figures that increasing the load self-regulation increases the period of the limit cycles but has little effect on the amplitude. The effect of k_n on the frequency transient is less obvious as the magnituded of the excursion is dependent upon the point on the limit cycle at which the step is applied. This was not realised at the time of the tests or steps would have been taken to ensure that all the disturbances were executed at the same point.

The response of the turbine to a step change in load can be seen from these Figures, particularly Figure 7.14. Immediately the step is applied, the frequency starts to fall and the governor responds by opening the servo. However, the power output does not increase immediately but tends to fall at first and then recovers. This is due to the drop in head at the turbine caused by the water control valve opening without a corresponding increase in flow, necessary to maintain the power output. The delay in the increase in power corresponds to the time required to accelerate the water column in the pipeline.

Figure 7.17 shows the results of a similar test from an initial load setting of 5MW. It was found that the effect of k_n at low loads was negligable and thus the only result shown is for $k_n = 0$. This is as would be expected since the load self-regulation is proportional, not only to K_n, but also to the steady state load power (P_{Lo} in equation 6.59).

Comparing the results of this Figure with Figure 7.14 it can be seen that the period of the limit cycles is much greater (typically, 21.7s as against 12.2s) and that the frequency transient is more heavily damped. This is a result of the considerably lower gain of the power/servo position relationship (Figure 6.8) at low loads.

Details of the modifications required to the simulation model to include the ILS are given in Section 6.4.1 and the Model Description for these studies is to be found in Appendix 2.5. It should be noted that it was necessary to set up the initial conditions incorrectly in the model to induce limit cycling to enable the step change to be applied at the same point on the limit cycle as on site. By observing the limit cycles during the run, it was possible to use the GUILDS DISTRB function (detailed in the GUILDS User's Guide) to apply the step at the desired point on the limit cycle, thus enabling a closed comparision with the site results to be made.

As a result of the incorrect initial conditions, the limit cycles shown at the start of the simulation results would not be expected to agree with the limit cycles of the site results. Further, the limit cycles at the start of the some site results reflect transients due to the ILS being brought into operation or due to changes being made to the value of k_n prior to the step being executed.

The value of the inertia constant of the generator and load was set up on the ILS on site to be 7s, thus, it was necessary to use this value in the simulation model and not the value of 8.7s that had been measured from the load rejection tests (section 7.3.1). However, as a result of the studies carried out with site signals injected into the simulation model (see section 7.7) it was subsequently found necessary to modified this value to 6.45s so that agreement could be obtained between the site and simulation step responses.

Initially, simulations of the site tests gave responses which, although similar to the site results did not agree in detail. In the light of the run-up simulation studies, it was decided to replace the "per-unit" non-linear function (equation 6.47) with the function derived from data recorded on site on the day of the tests (equation 7.4). However, this did not result in the expected improvement in the response of the simulation model.

The main difference between the two sets of results was in the mean steady state values of servo position and power output; for a given servo position the simulation model consistently gave a value of power output that was lower than on site. By increasing the reservoir head in the simulation model, it was possible to obtain agreement, in the steady state, between the two sets of results. However, the increase in head which this demanded was outwith the limits of any possible errors in site measurements and, in addition, the increase in head had an adverse effect on the transient performace.

From a closer inspection of the site power/servo position data and the site transient responses it was found that the steady state values from the simulated isolated load step tests did not correspond to the power/servo position data and, in fact, the steady state values from the simulation were, as would have been expected, in agreement with the power/servo position data. Several possible sources of error have been identified which could have resulted in the discrepancies between the these two sets of site data: errors in the recording of one or other set of data; interaction between the ILS and the power transducer resulting in the power transducer giving erroneous results; or a physical phenomenon as yet unobserved. However, no explanation for the discrepancies was found and, as it was not possible to repeat the simulated isolated load tests, the ILS having been removed from site along with the analogue control rack[6], it was necessary to accept the results as they were.

As a consequence of this, to facilitate comparision between site and simulation results, it was decided to consider only changes in the power and servo position signals from their respective steady state values. As the gain of the servo position/power curve has a significant effect on the simulated isolated load response, the simulation operating point was chosen to give the best agreement with the site results. This involved some minor adjustments to the initial power level, the step size and, in some cases the frequency set point. It was also necessary to change the head slightly in order to avoid using an initial power setting which differed significantly from the nominal value recorded on site. One possible explanation for this, is the U-tube oscillation between the reservoir and the surge shaft which causes fluctions in the effective head at the turbine. Although this phenomenon is modelled in the simulation, the cumulative effect of the series of step tests carried out on site could not be included.

The site and simulation results also differed initially in the transient response to the change in load and in the period of the subsequent limit cycling. After some considerable effort, culminating in the use of the techniques discussed in section 7.7, it was found that the source of these differences was a discrepancy in the values of the governor parameters between the site governor and the simulation model. It had been overlooked that, during some early work on site, the governor constants T_3 and T_4 had been changed from 0.2 to 0.3 so that the FORTRAN governor[6], which used a slightly different integration technique, more closely resembled that of Findlay[3]. Also, it was discovered that the governor droop (b_p) which was nominally set at 3% was in fact about 3.4%.

The results of a simulation of a site test using the ILS are shown in Figure 7.14, for a 5.2% step change in load from 24MW with k_n equal to zero. The initial load setting and the step size were choosen to give the best agreement with the site test results, which are shown as broken lines in the Figure. As noted above, the servo position and power signals are shown as changes from mean levels rater than as absolute values. The values used for the results in this Figure were:-

	SERVO POSITION	POWER OUTPUT
SITE RESULTS	67.5%	75.0%
SIMULATION	74.0%	75.0%

The Documentation File for this simulation run can be found in Appendix 3.17.

It can be seen from this Figure, that the simulation results agree reasonably well with the site responses. In particular, the shape of the frequency transient and the amplitude and period of the susequent limit cycles are in close agreement, although, the overall shape of the limit cycles, especially in the power signal, are slightly

different. This difference reflects, at least in part, the assumption that the limit cycles result only from hysteresis in the water control valve linkages, whereas, there are almost certainly other non-linearities, for example friction, involved.

A similar comparison can be made between the simulation and site results for simulated isolated load tests with k_n = 0.5 and k_n = 1.0, as shown in Figures 7.15 and 7.16 respectively. Again good agreement has been obtained although the results for the k_n = 1.0 case are not as close. The mean levels for the servo position and power signals for the k_n = 0.5 case were:-

	SERVO POSITION	POWER OUTPUT
SITE RESULTS	69.0%	76.0%
SIMULATION	70.0%	70.0%

and for the k_n = 1.0 case, the values were:-

	SERVO POSITION	POWER OUTPUT
SITE RESULTS	68.0%	75.0%
SIMULATION	70.0%	70.0%

In Figure 7.17 the results of a simulation of a 4.8% step from an initial load of 3.5MW are presented for the case k_n = 0. The site results are shown again as broken lines in this Figure, and the mean levels for the servo position and power signals were:-

	SERVO POSITION	POWER OUTPUT
SITE RESULTS	28.0%	18.0%
SIMULATION	25.0%	14.0%

Once again, the simulation results can be see to follow closely the responses obtained on site, although, it was necessary to increase the amount of hysteresis in the model (from $\pm$ 0.7% to $\pm$ 1%) to

get these results. Again this is probably a reflection of the over-simplified representation the non-linearities in the servo/control valve linkages. As with the site tests, the effect of k_n at low loads was negligable and thus only one set of results has been included.

The agreement achieved between the site and simulation results at both high and low loads, for which the step responses are quite different, is a result of including the non-linear servo position/power output relationship in the model. Thus the ability to simulate the isolated load behaviour of the set over the full range of operation without changing the model in any way, supports the inclusion of this function in the model. As stated earlier, there are still some uncertainties in deriving the exact form of the function in per-unit terms and some further work would be required to enable a general function to be used.

7.6 Isolated Load Simulation

Having shown that the simulation model could be used to reproduce the results of ILS tests on site, it was decided to use the model to study briefly the differences between simulated isolated load and true isolated load behaviour. Figure 7.18 shows the results of one such study for a 5.2% step change in load from a steady state load of 24MW. The results from a simulated isolated load (ILS) site test with $k_n = 0$ (Figure 7.14), are shown as broken lines in this Figure. The model used for the true isolated load simulations is given in Appendix 2.6 and the simulation run and site test shown in Figure 7.18 are documented in Appendix 3.14 and 3.21.

The mean values of the servo position and power signals shown in the Figure were:-

	SERVO POSITION	POWER OUTPUT
SITE RESULTS	69.0%	75.0%
SIMULATION	77.0%	75.0%

Clearly, some differences between the two sets of results would be expected since, as noted earlier, the isolated load simulation is only an approximate representation of the true isolated load behaviour and ignores the effect of the frequency (speed) term in the flow and power equations (as the turbine is operationg at constant speed). However, from Figure 7.18, it can be seen that, although the two sets of results are different, the magnitude of the initial frequency transient and the period of the subsequent limit cycles are in close agreement. The main discrepancies lie in the response following the initial transient before the steady state limit cycling begins and in the magnitude of the limit cycles.

These results indicate that, at least in part, the ILS is a good representation of the true isolated load response of the turbine and generator, particularly with reference to stabiliy margins. Thus, the behaviour of the governor under thee conditions can be demonstrated without the need for extensive system splitting tests.

7.7 Model Response to Injected Signals

As an aid to the simulation studies discussed in section 7.5 a facility was developed whereby the actual signals recorded on site could be injected into the simulation model so that the responses of individual parts of the model could be compared with the site test results. Thus, sources of error within the model could be isolated to one particular block. The Model Description used for this work is to be found in Appendix 2.7.

For the simulated isolated load tests on site, the data for the simulated frequency, servo position and power output signals were recorded at a storage interval of 0.1s. As the simulation model used an integration interval of 0.05s the site data was read in at only every second integration interval.

Using the signals recorded on site as inputs to the model enables the model to be split into three separate functional blocks, that is:-

(a) the governor, which has frequency (simulated) as input and servo position as output;

(b) the turbine, which has servo position as input and power as out;and

(c) the load, with power as input and frequency as output.

Tests were carried out on these sections of the model by using the appropriate site signal as the input and comparing the output with the corresponding site response. The results of these tests were useful in solving some of the problems associated with modelling the simulated isolated load behaviour of the set.

A typical set of results obtained using this technique are presented in Figure 7.19, and the three simulation runs required to produce the Figure are documented in Appendix 3.22 to 3.24. The solid lines in the Figure show the responses of the three sections of the model to the appropriate input signals and the corresponding site signals are shown as broken lines. Of course, since the output of one section is, in reality, the input to the next, the site results shown for comparison with the outputs from the model are also the inputs to the subsequent sections of the model.

As before, due to the inconsistences between the model and site steady state values, the servo position and power signals are shown as deviations from a mean level. The values used in this instance were:-

	SERVO POSITION	POWER OUTPUT
SITE RESULTS	68.7%	75.0%
SIMULATION	75.0%	65.0%

From Figure 7.19 it can be seen that each section of the model behaves very similarly to the equivalent section of the real plant. The agreement between the two sets of results is slightly better than had been previously obtained with the complete model, probably as a result of cumulative errors. As noted earlier several differences in the model parameters were identified by the use of this technique. Had more time been available it would have been instructive to have adopted a similar approach to the run-up and load rejection simulations, resulting in, hopefully, similar improvements in response.

Another modification made to the model at this stage involved using different integration techniques within the one model. For speed of solution, the governor implementation on site uses Euler integration with an interval of 0.1s. However, due to numerical instability in the pipeline model, the simulation requires to use a Fourth Order Runge-Kutta technique with a step length of 0.05s. Thus, to ensure that the simulation model of the governor was as good a representation of the site governor as possible it was decided to include the option of solving the governor section of the model using Euler.

As a result of the way in which the integration subroutines of GUILDS were written, it was possible to obtain Euler integration from the Runge-Kutta method by evaluating the governor equations only every fourth pass through the dynamic section of the model, that is when the GUILDS variable M was set to 1 (see section 3.5 of the GUILDS User's Guide). Further, if the governor equation were only evaluated at every second integration interval of the Runge-Kutta method, the effective integration interval would be 0.1s, as required.

This approach was found to work sucessfully, although no differences between the Runge-Kutta/Euler model and the Runge-Kutta model were observed.

7.8 Conclusions

In general, it has been shown that good agreement can be obtained between the results of site tests and those from the simulation model. This justifies the confidence in the simulation model to predict the responses of different governor algorithms prior to testing on site. However, as noted earlier, some further work on site may be necessary to study in detail the non-linear relationship between power and servo position and to investigate the reasons for the different values of inertia constant required for run-up and load rejection simulations. Also, it may be valuable to monitor other parameters on site, such as flow through the turbine, the head at the turbine, the surge shaft level and the relief valve position.

It is worth noting that the simulation models used for the different studies presented in this Chapter are, to a large extent, consistent. The only parameters which have been adjusted to obtain agreement with site results, apart from the head (H_r), the frequency reference (f_s) and the coefficients of the non-linear servo position/power function, which would be expected to change with site

conditions, are the generator inertia constant (T_a), the no-load flow losses (F_1) and the hysteresis in the water control valve linkages. In addition, some of the governor parameters were adjusted to match those in use in the site governor when the specific tests were carried out.

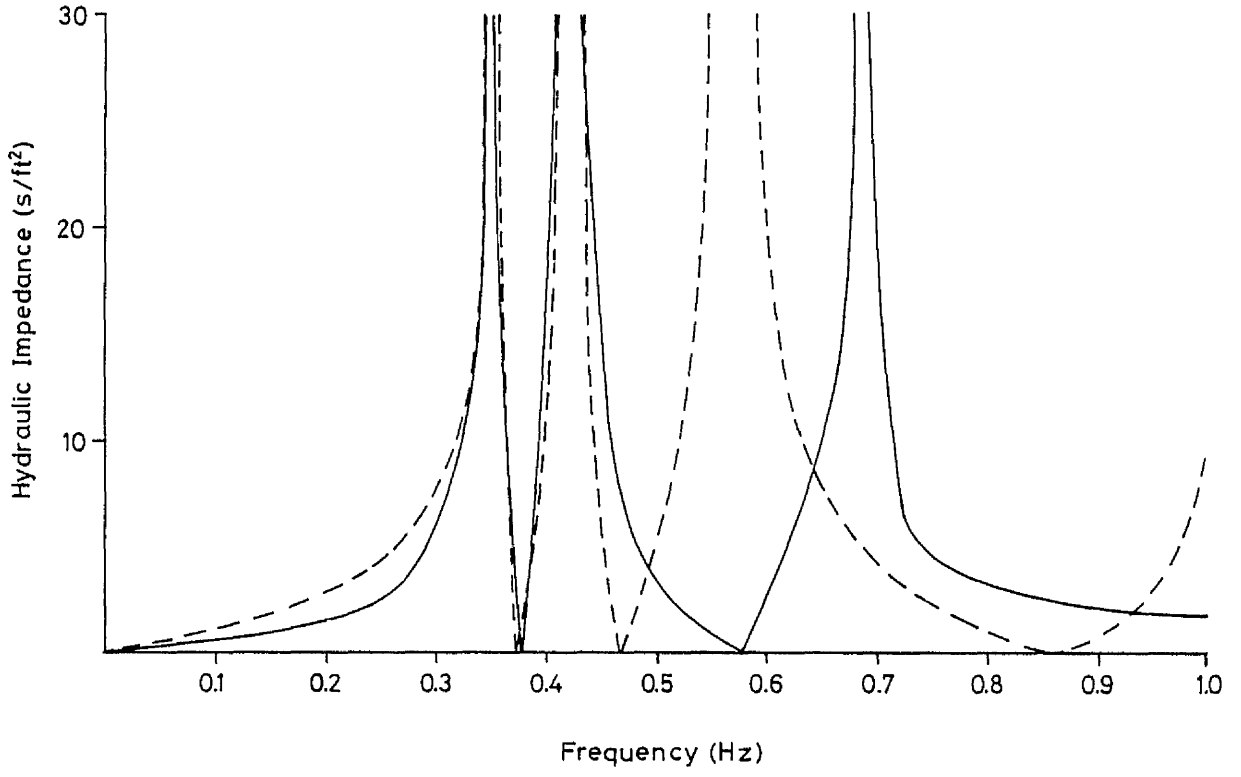


Figure 7.1 - Impedance Chart for Three Pipe Section Model

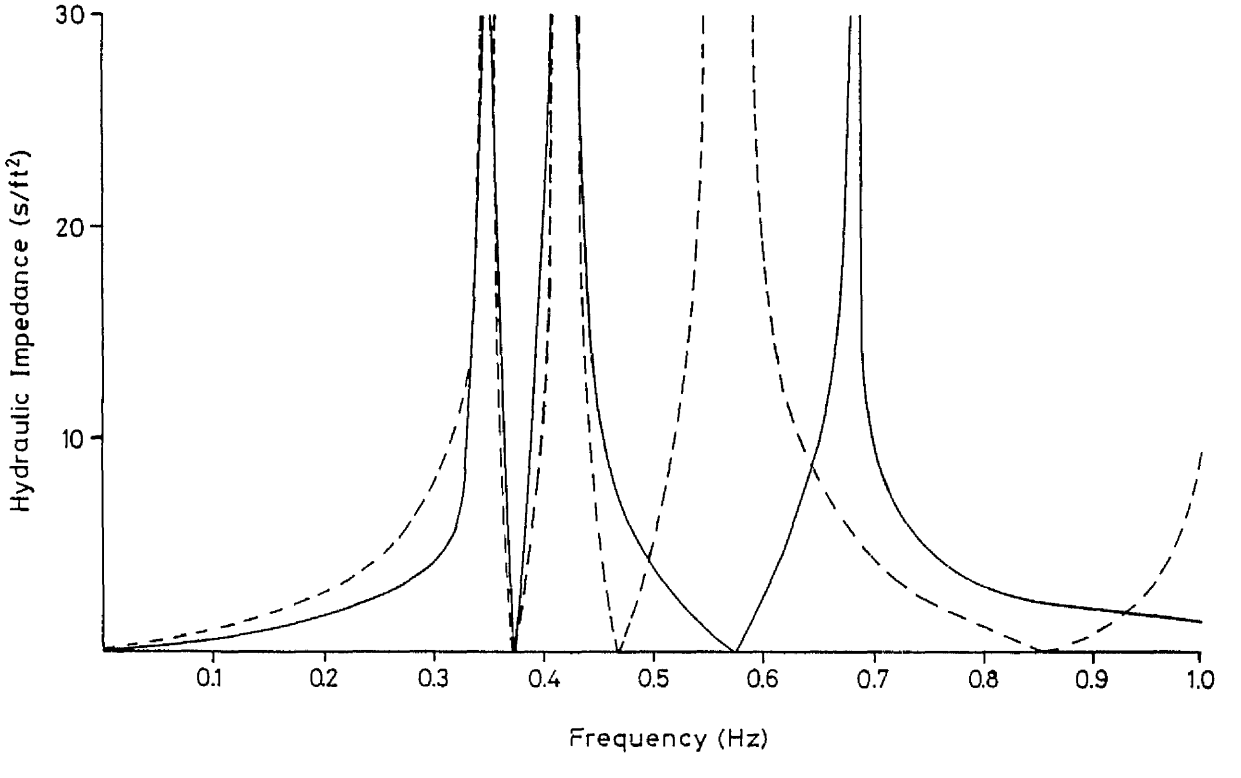
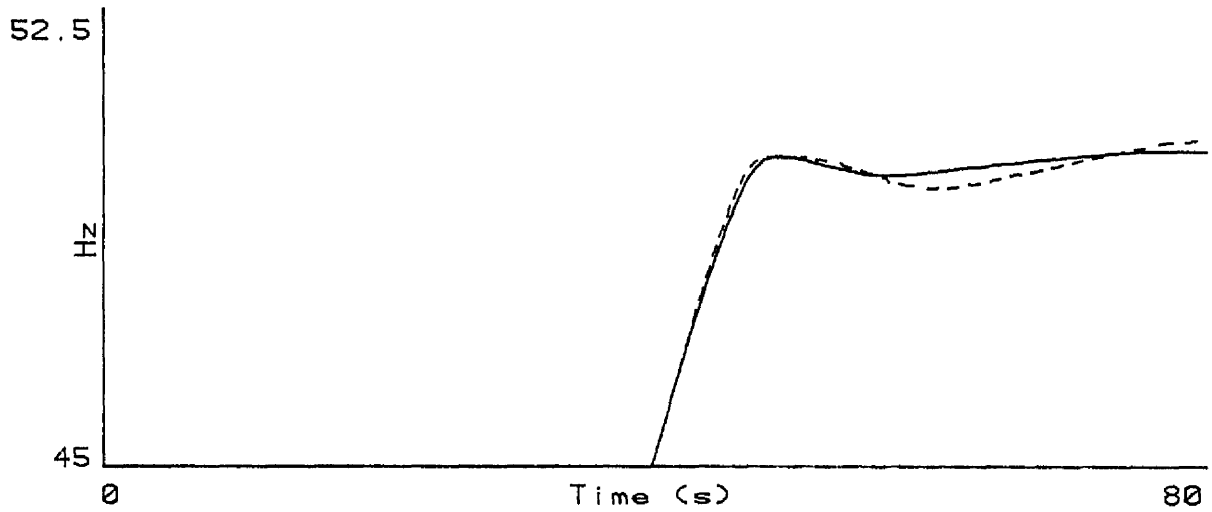
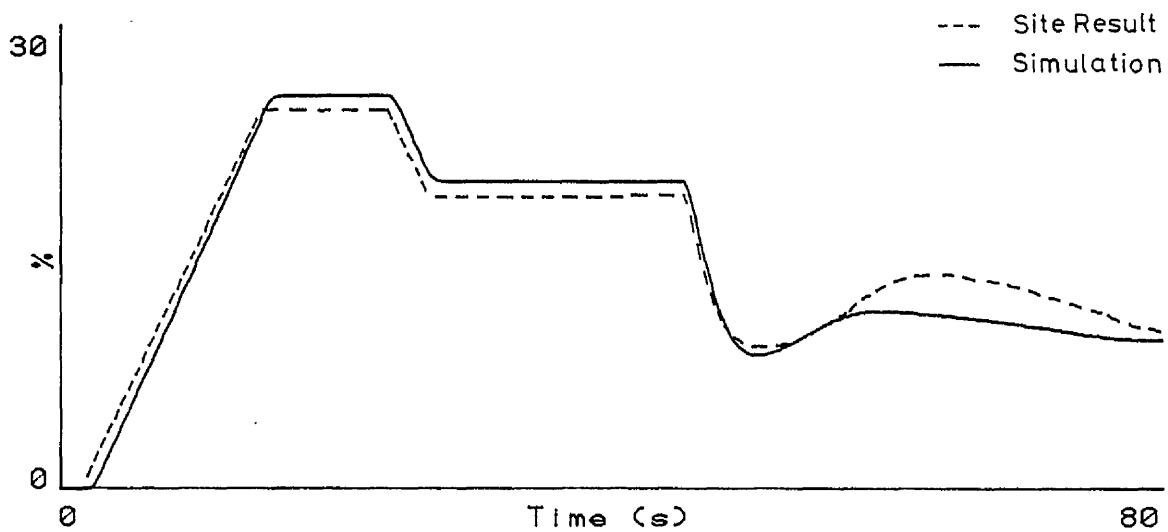


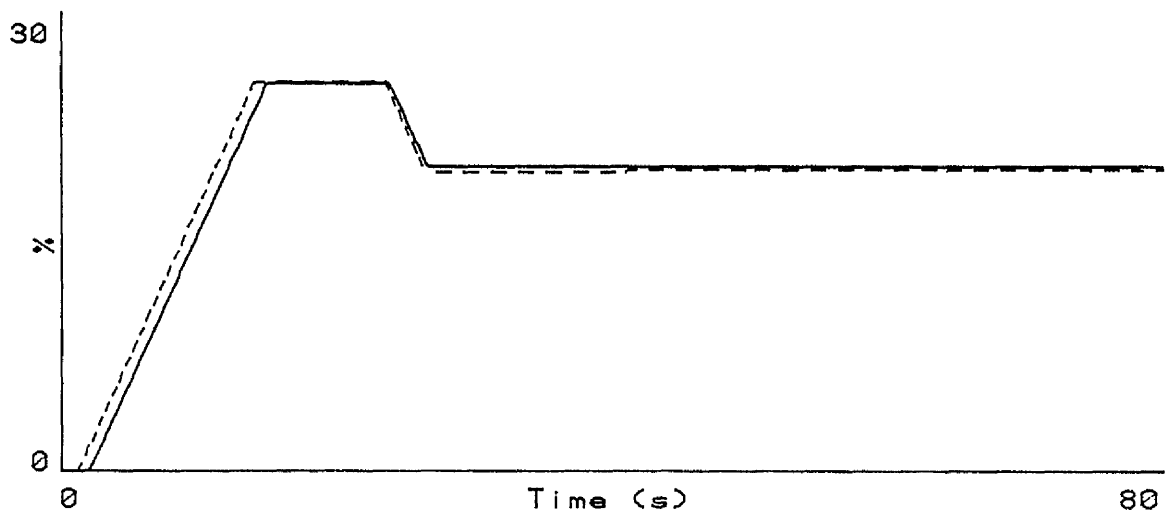
Figure 7.2 - Impedance Chart for Three Pipe Section Model with Reservoir and Surge Shaft



Frequency Transducer Output

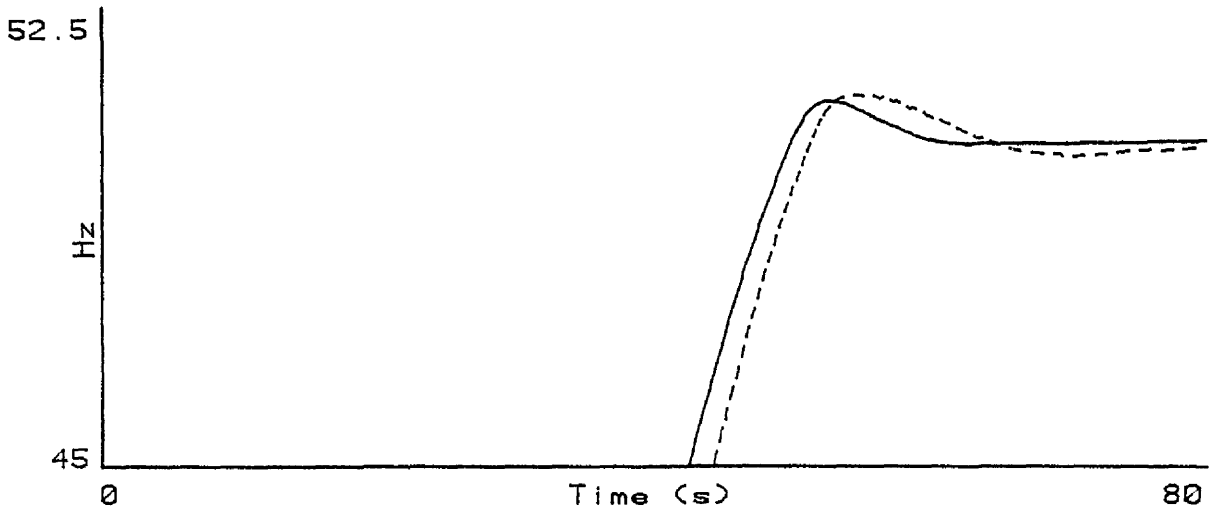


Servo Position

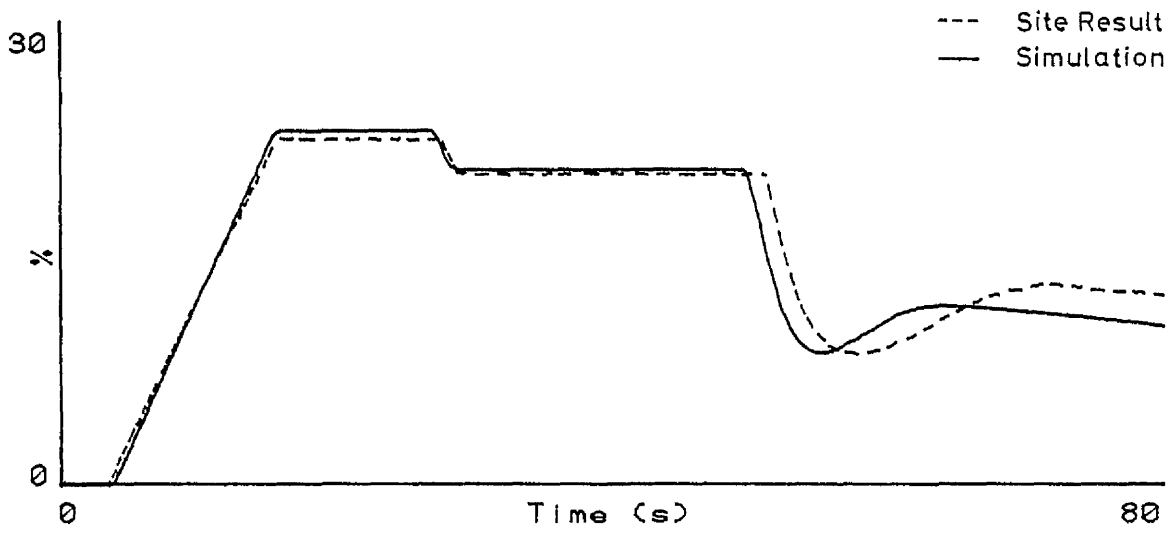


Servo Set Point

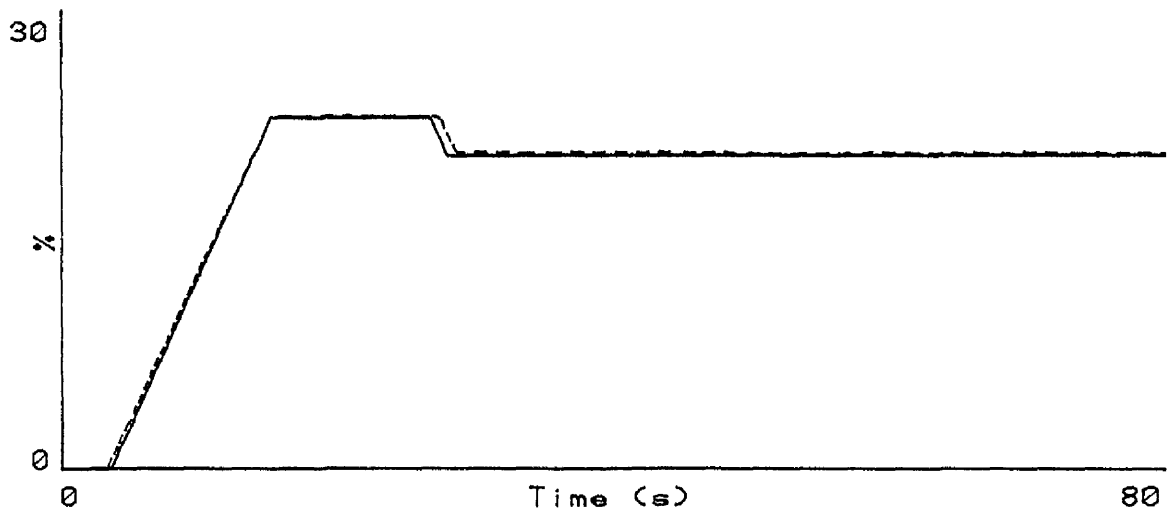
Figure 7.3 - Run-up Characteristic



Frequency Transducer Output

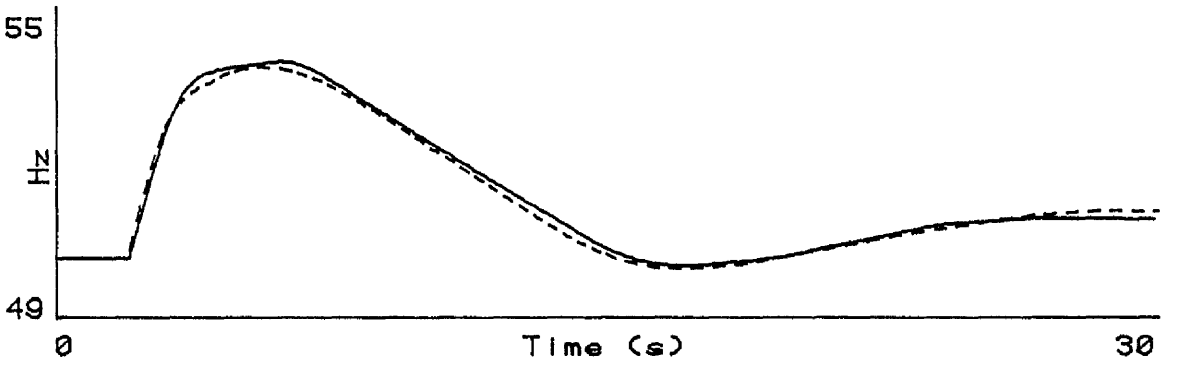


Servo Position

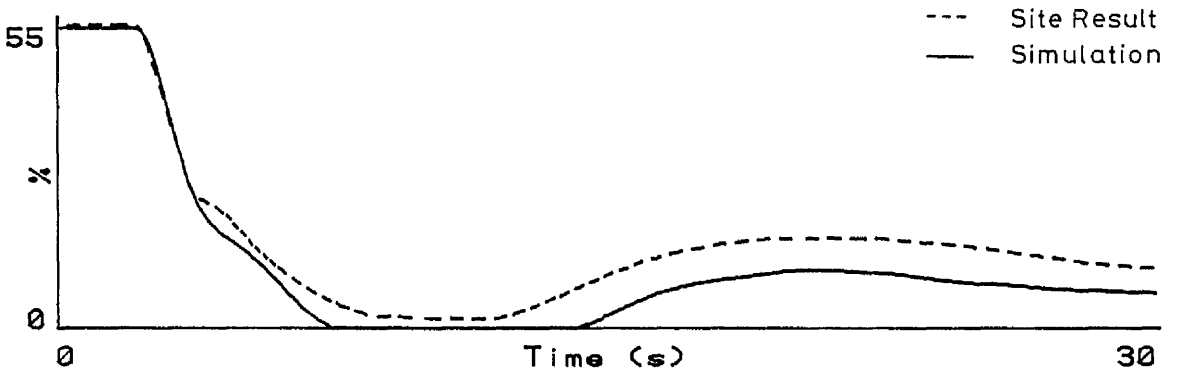


Servo Set Point

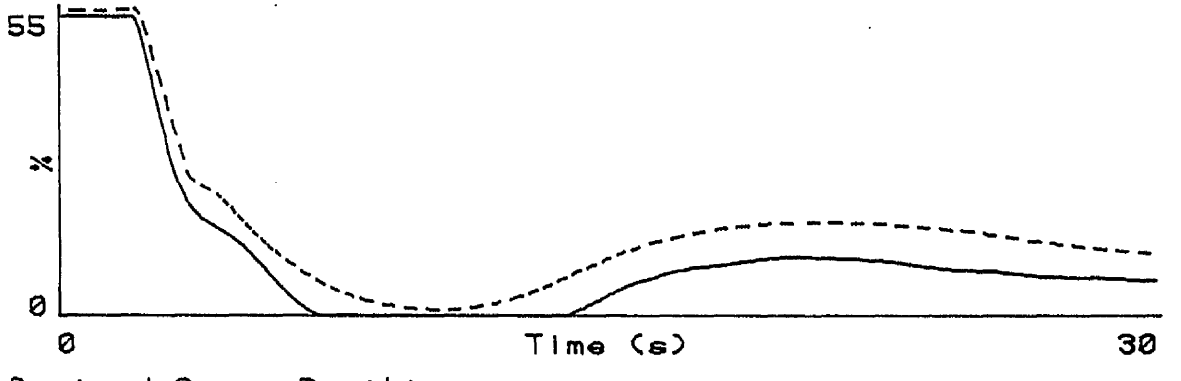
Figure 7.4 – Run-up Characteristic



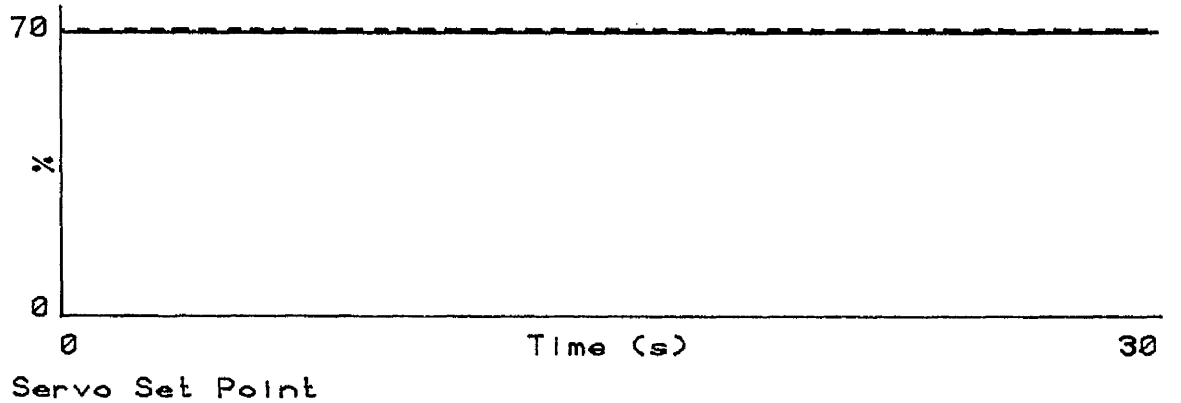
Frequency Transducer Output



Servo Position



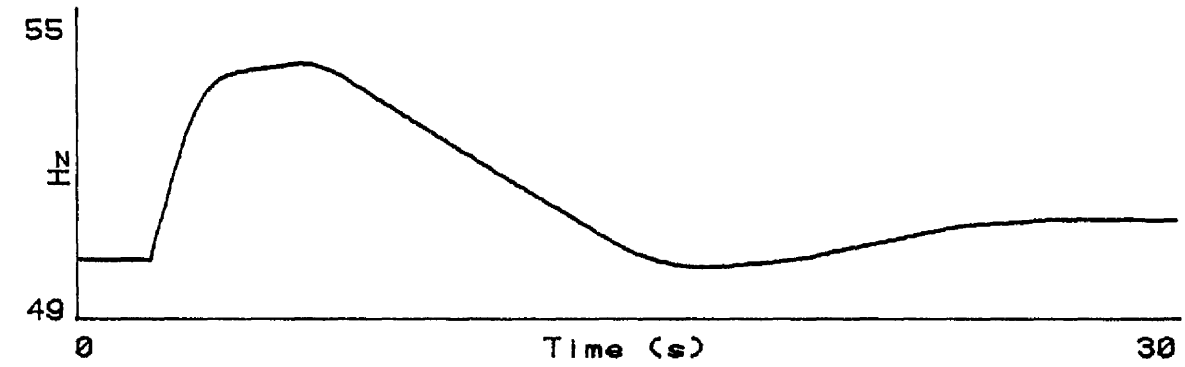
Desired Servo Position



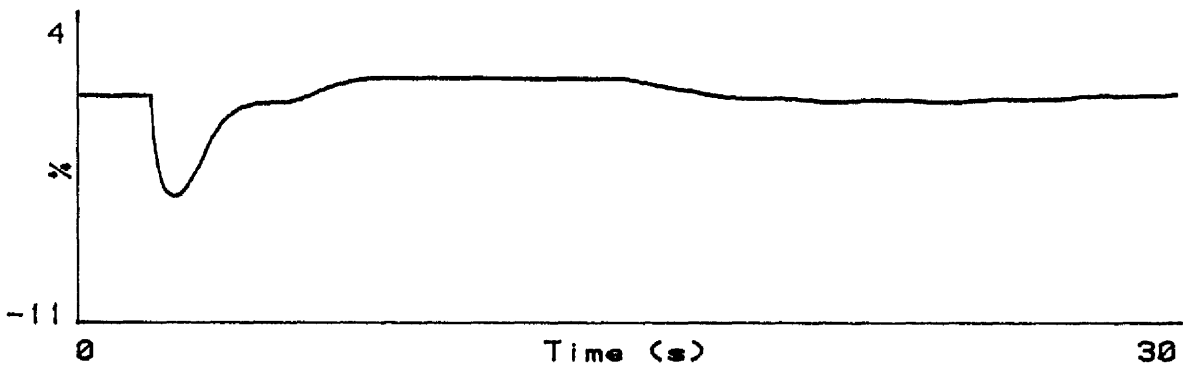
Servo Set Point

Figure 7.5 - 15MW Load Rejection

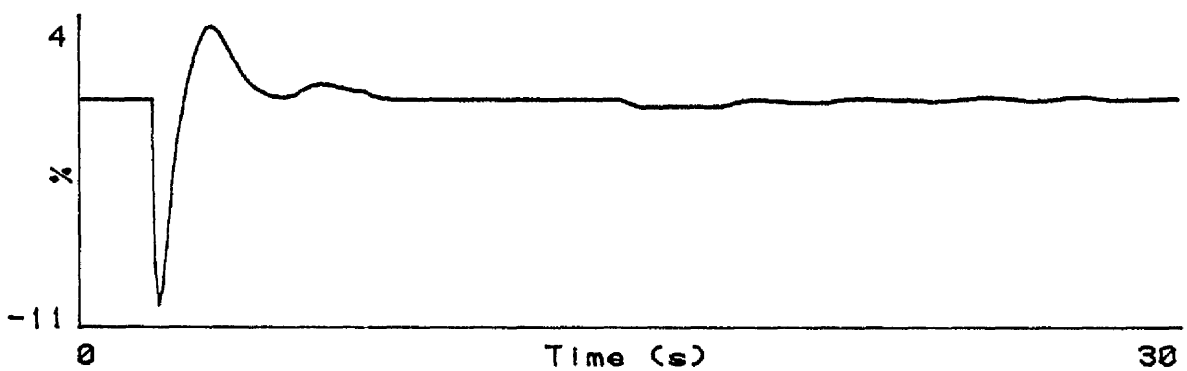
Double Derivative Governor



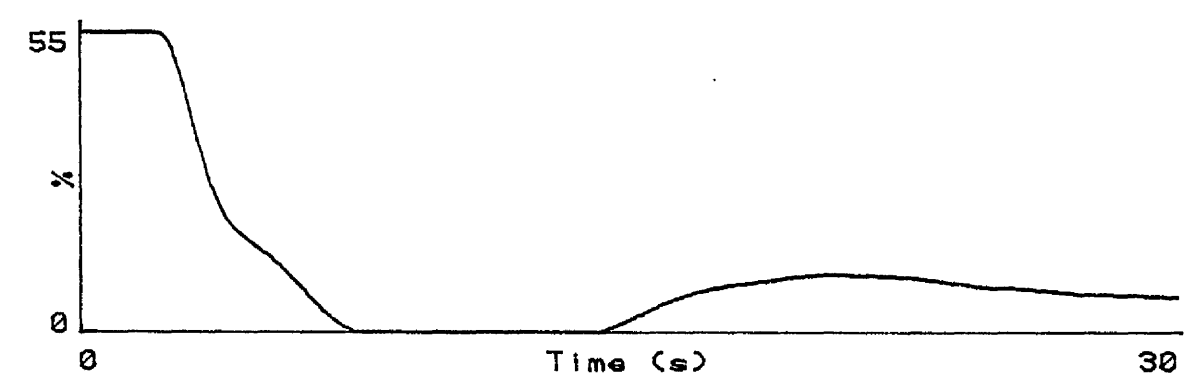
Frequency Transducer Output



First Derivative Term



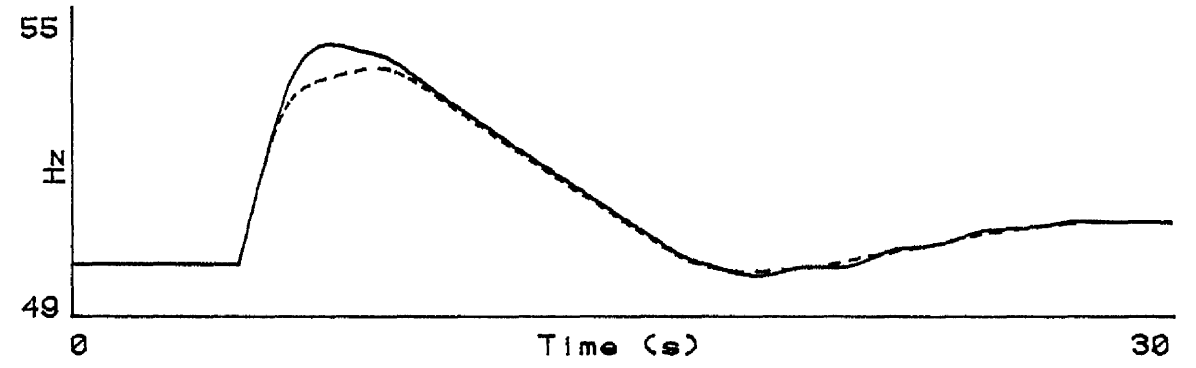
Second Derivative Term



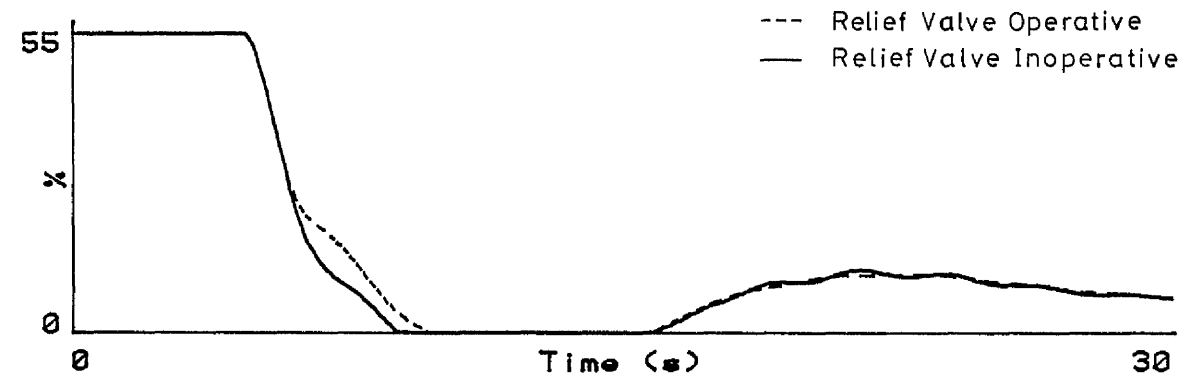
Servo Position

Figure 7.6 - Simulated 15MW Load Rejection

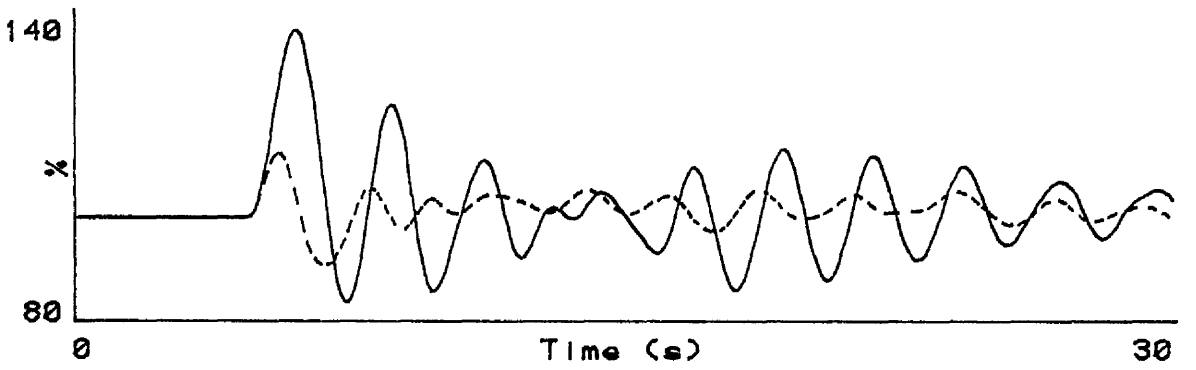
Double Derivative Governor



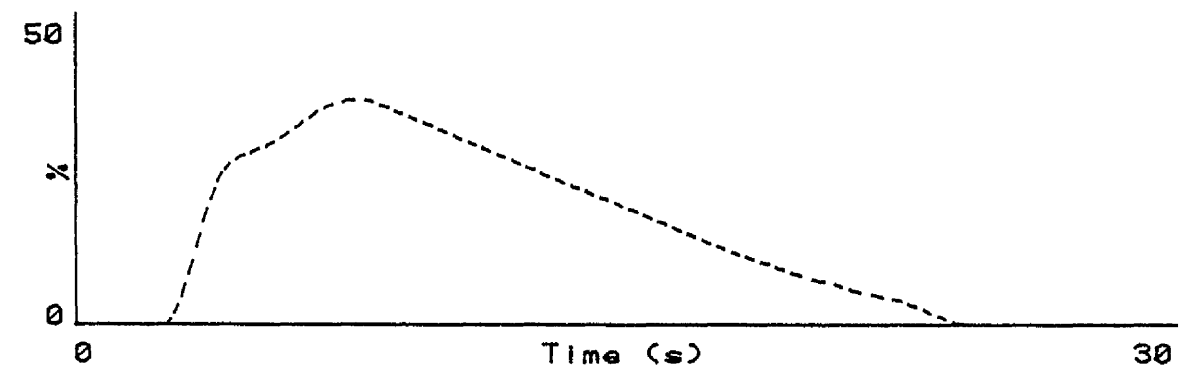
Frequency Transducer Output



Servo Position



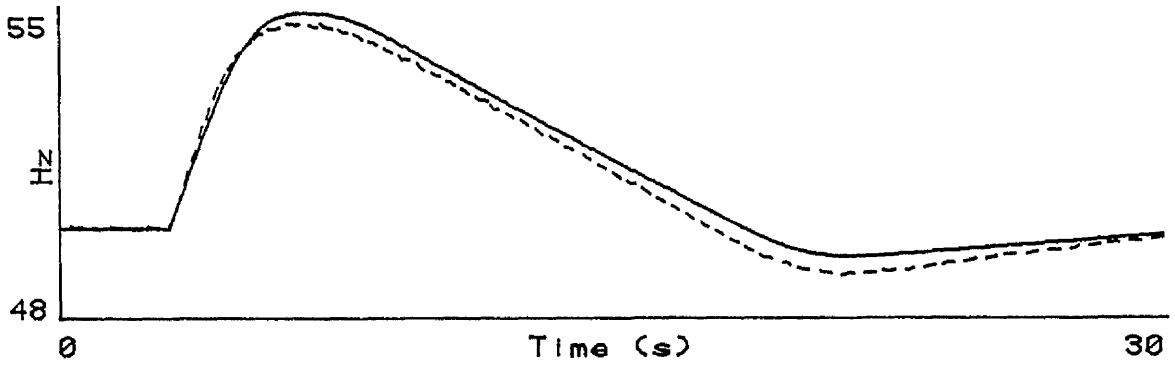
Turbine Head



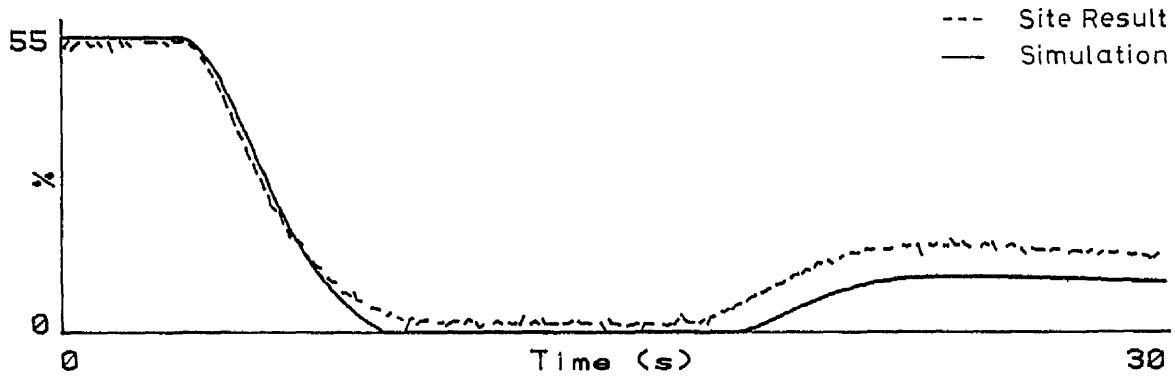
Relief Valve Position

Figure 7.7 - Simulated 15MW Load Rejection

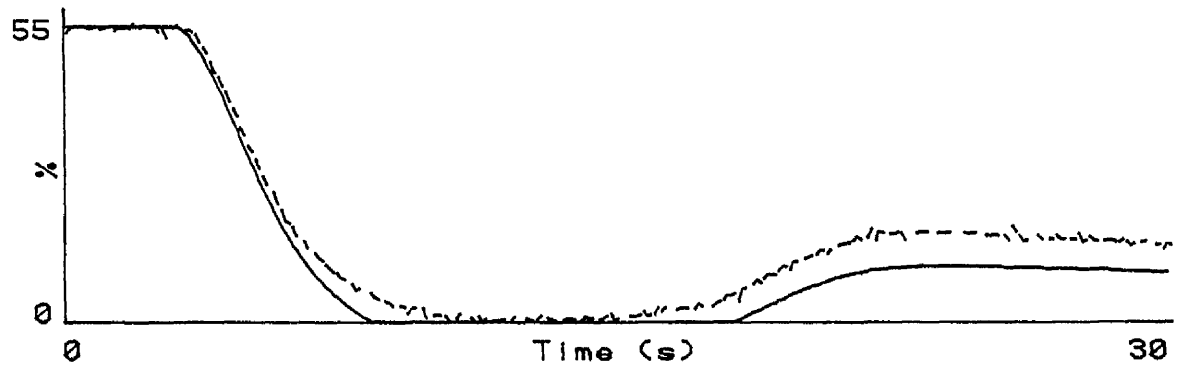
Effect of Relief Valve Action



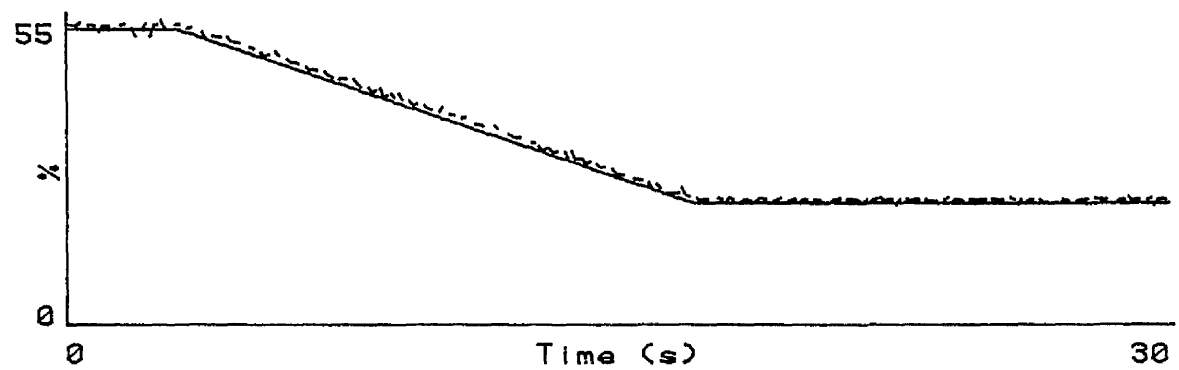
Frequency Transducer Output



Servo Position



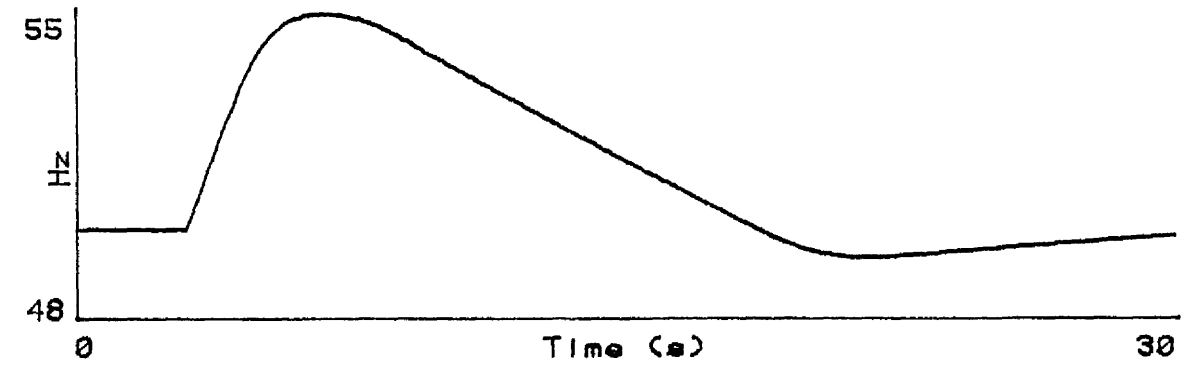
Desired Servo Position



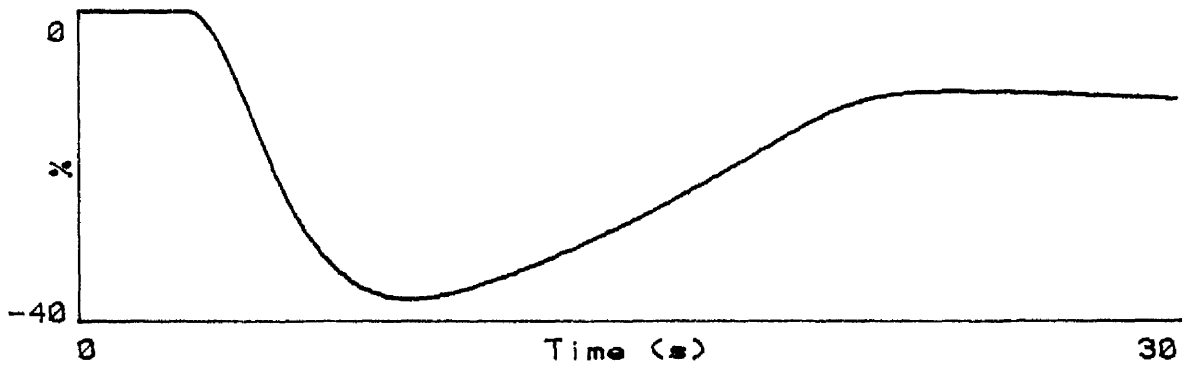
Servo Set Point

Figure 7.8 - 15MW Load Rejection

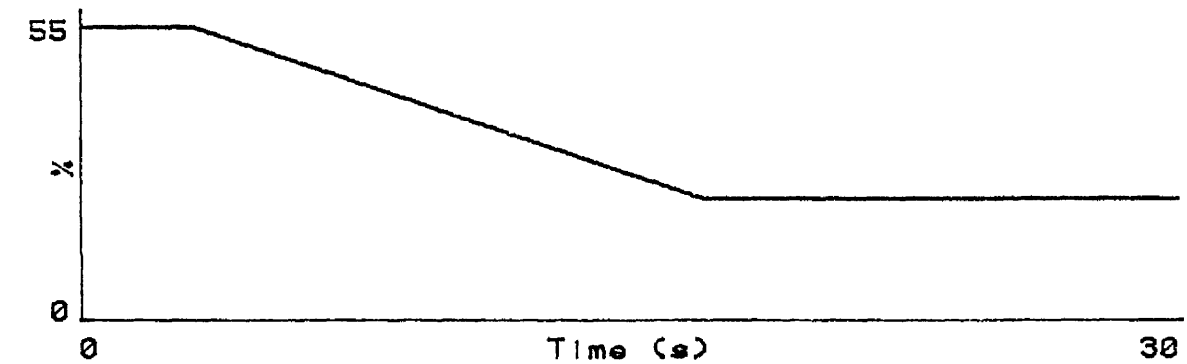
Temporary Droop Governor



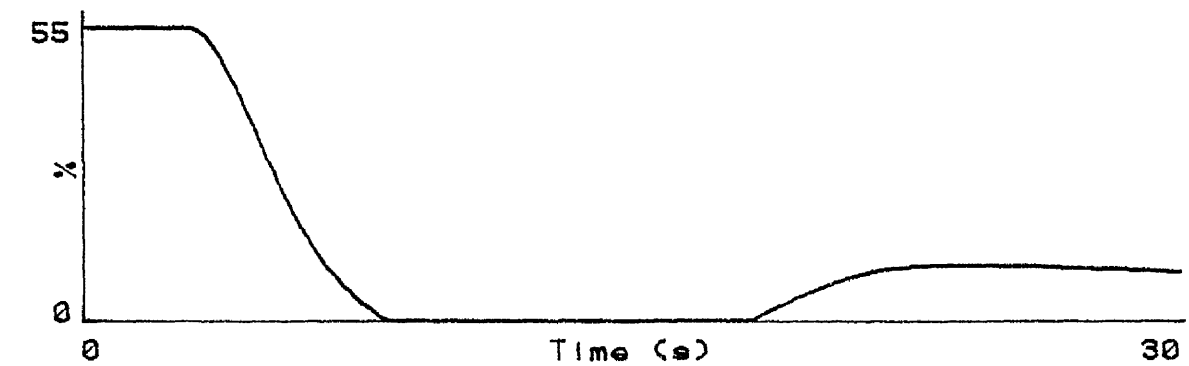
Frequency Transducer Output



Governor Output



Servo Set Point



Desired Servo Position

Figure 7.9 - Simulated 15MW Load Rejection

Temporary Droop Governor

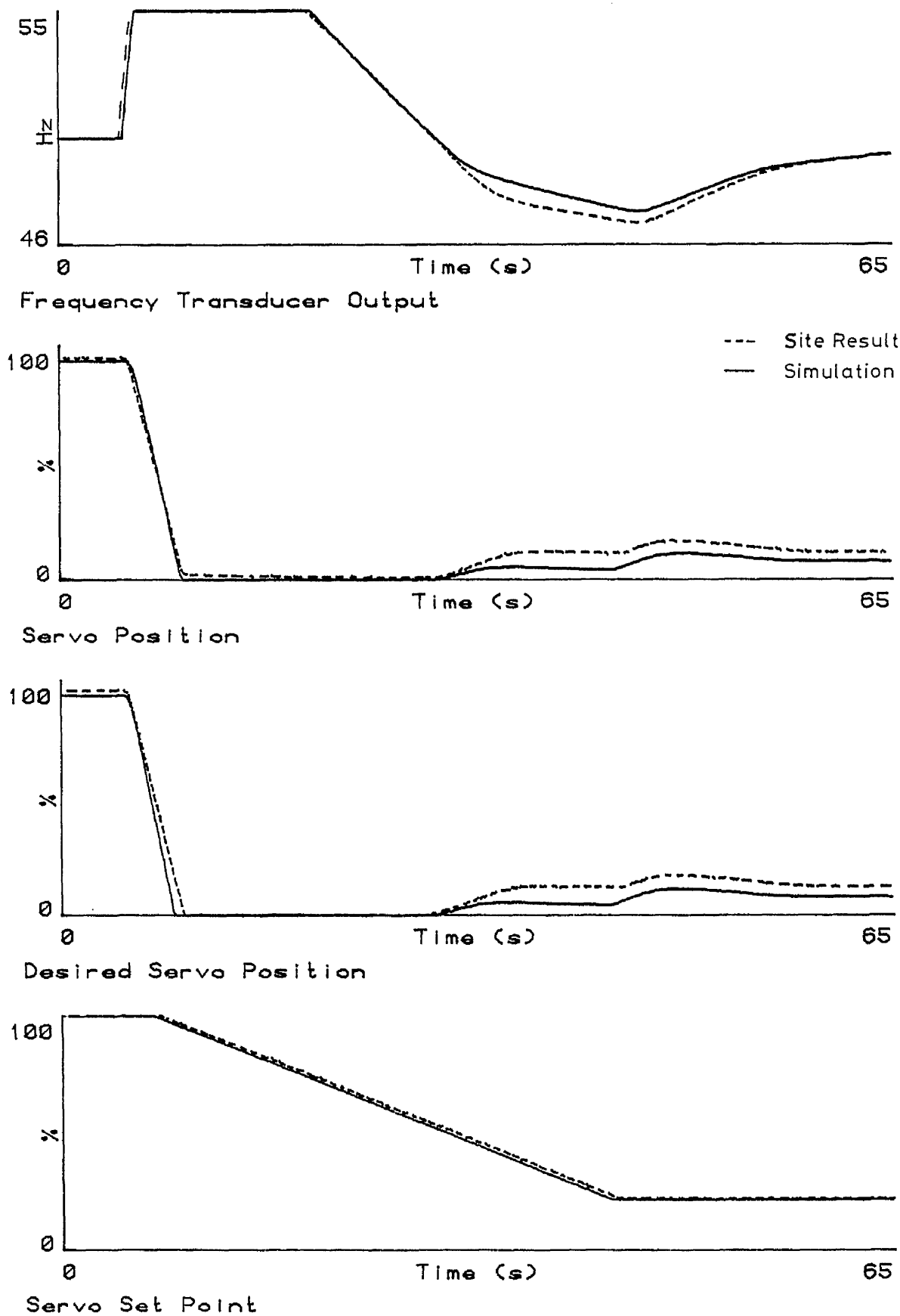


Figure 7.10 - 33MW Load Rejection

Temporary Droop Governor

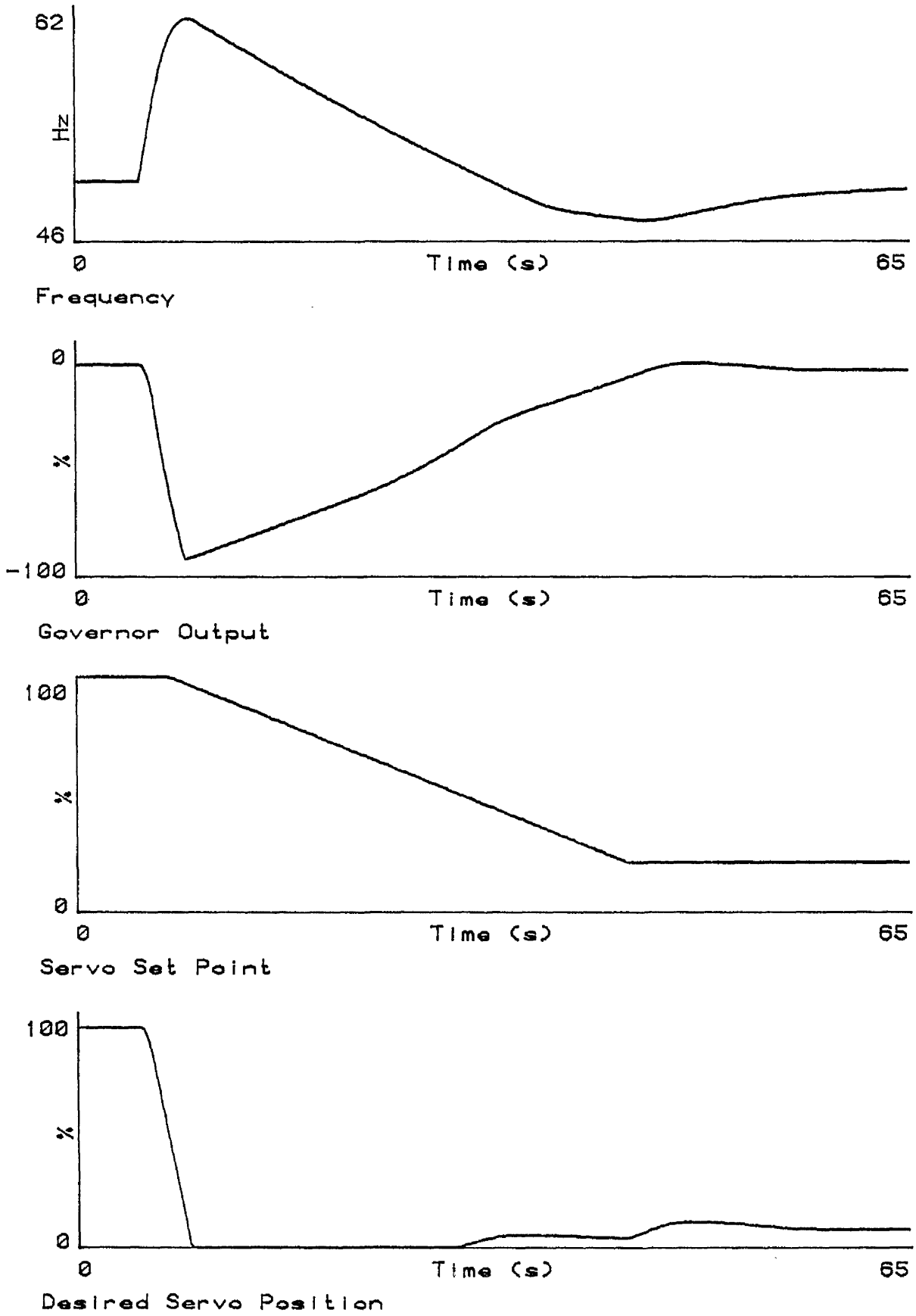
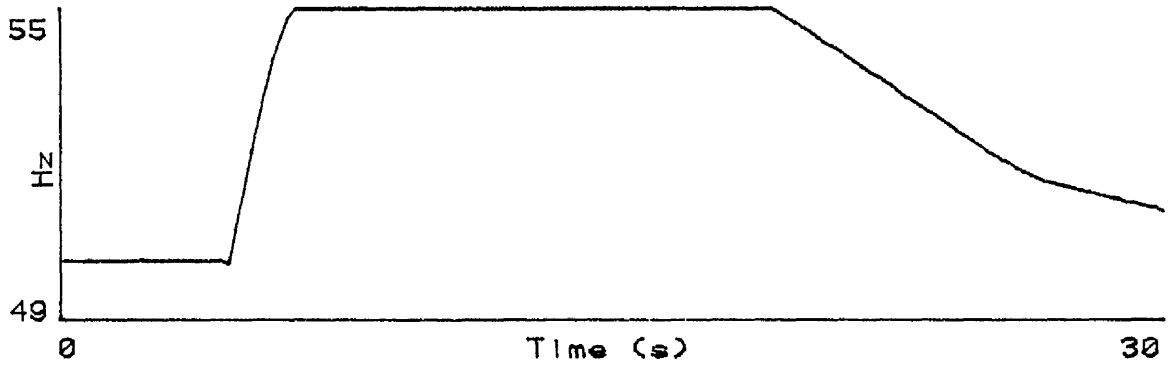
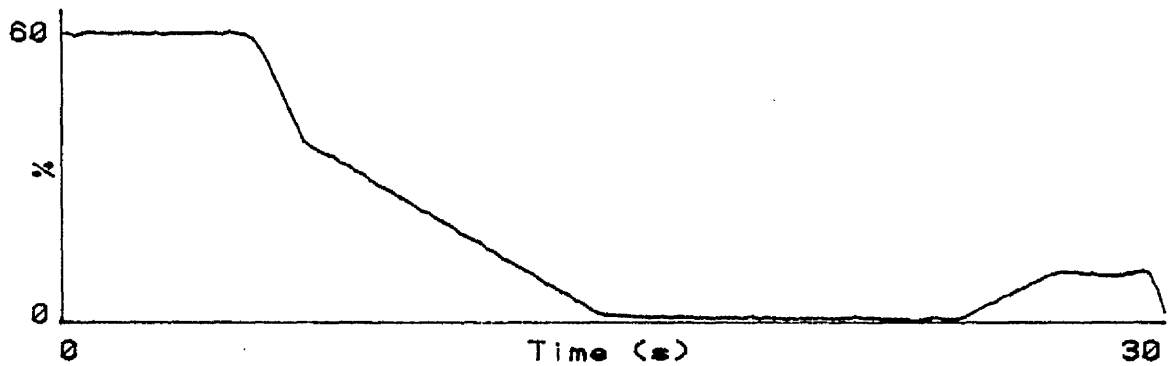


Figure 7.11 - Simulated 33MW Load Rejection

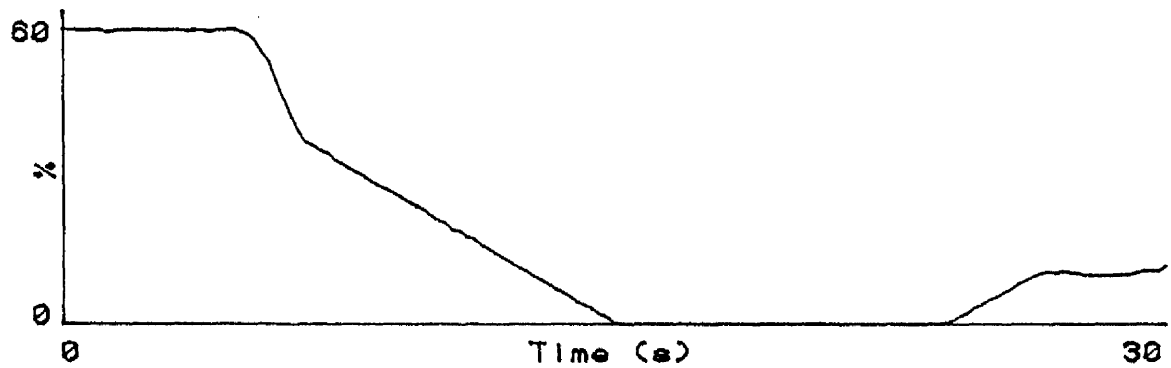
Temporary Droop Governor



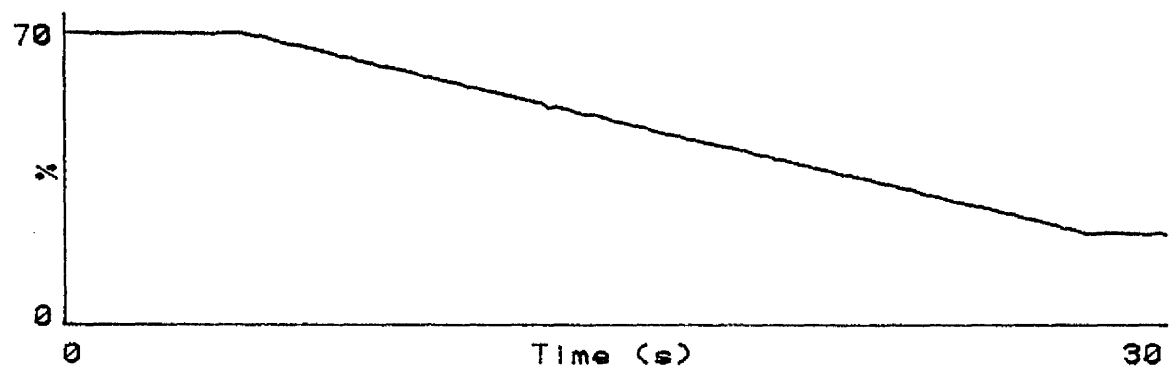
Frequency Transducer Output



Servo Position



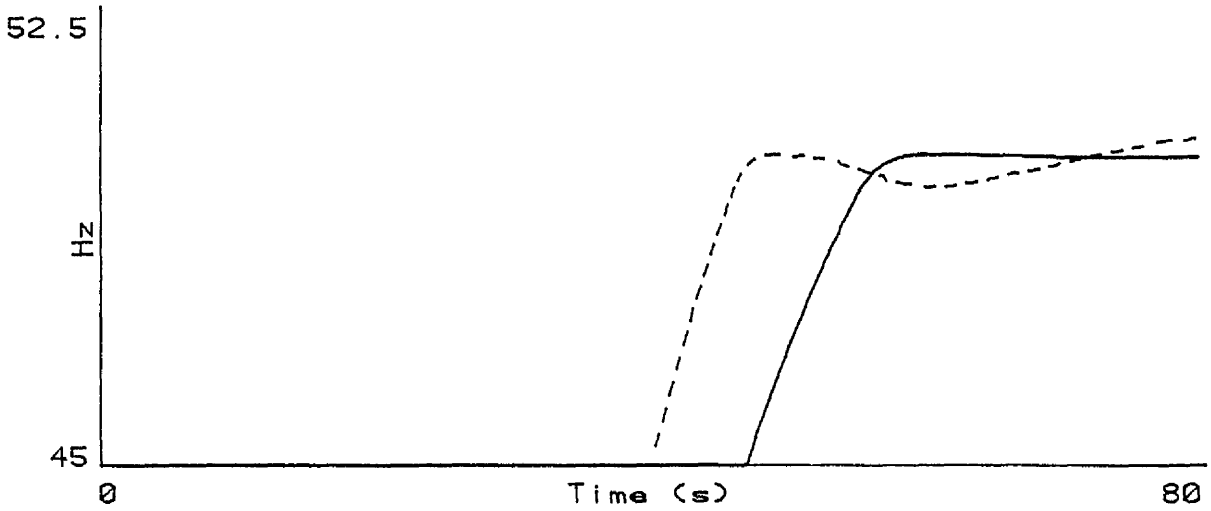
Desired Servo Position



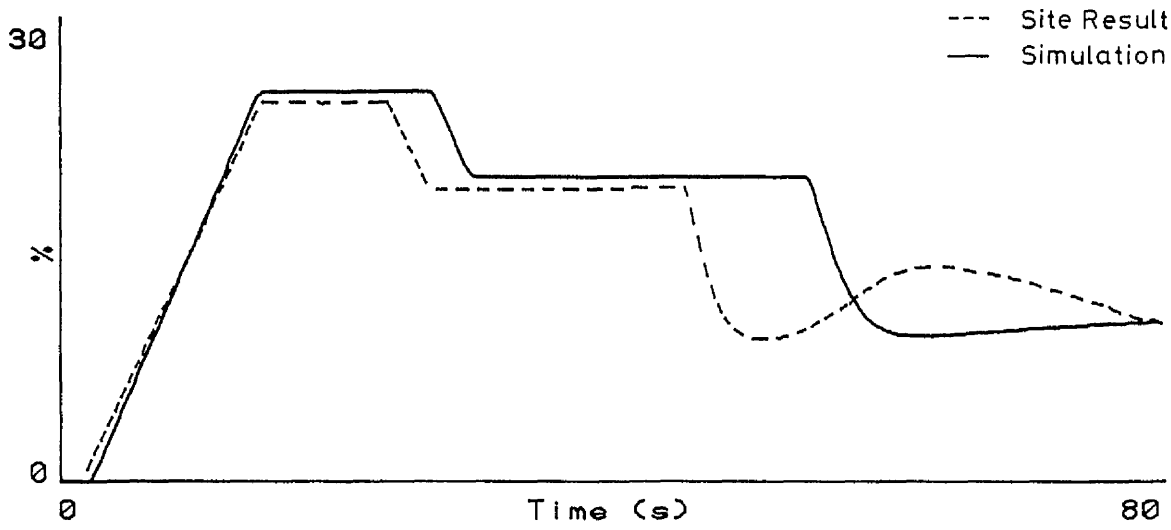
Servo Set Point

Figure 7.12 - 20MW Load Rejection

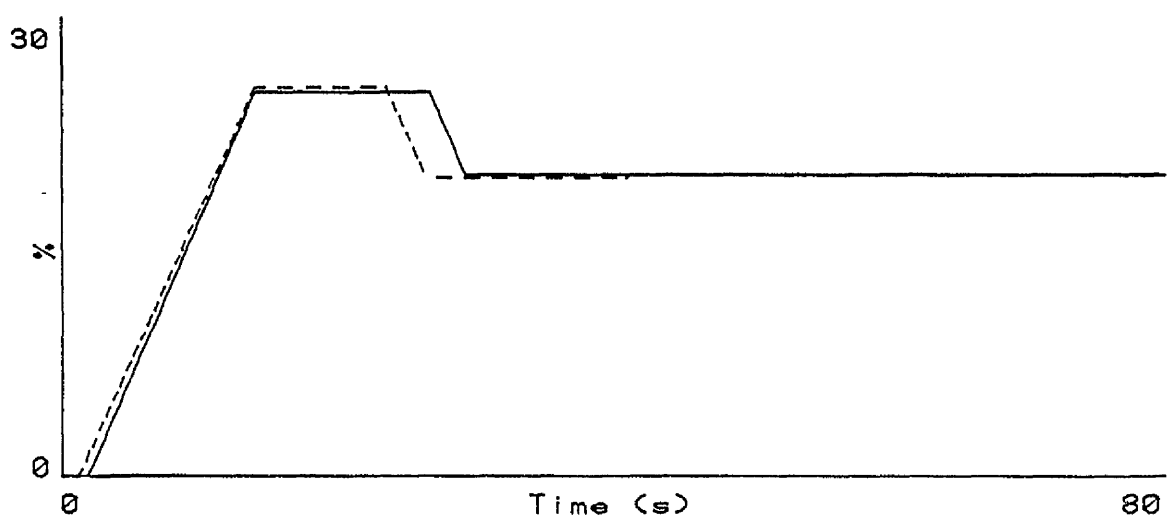
Faulty Governor



Frequency Transducer Output



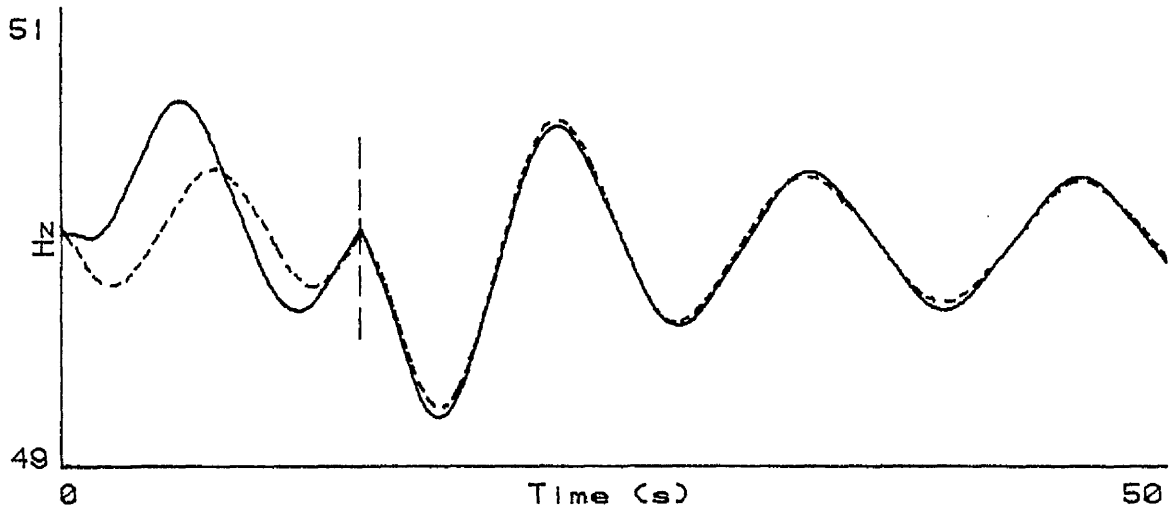
Servo Position



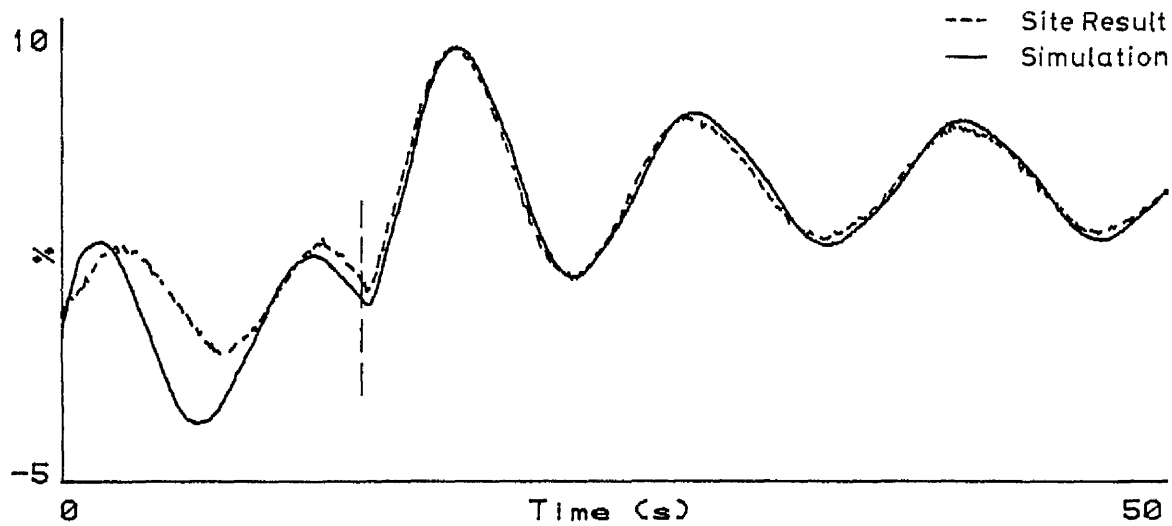
Servo Set Point

Figure 7.13 - Run-up Characteristic

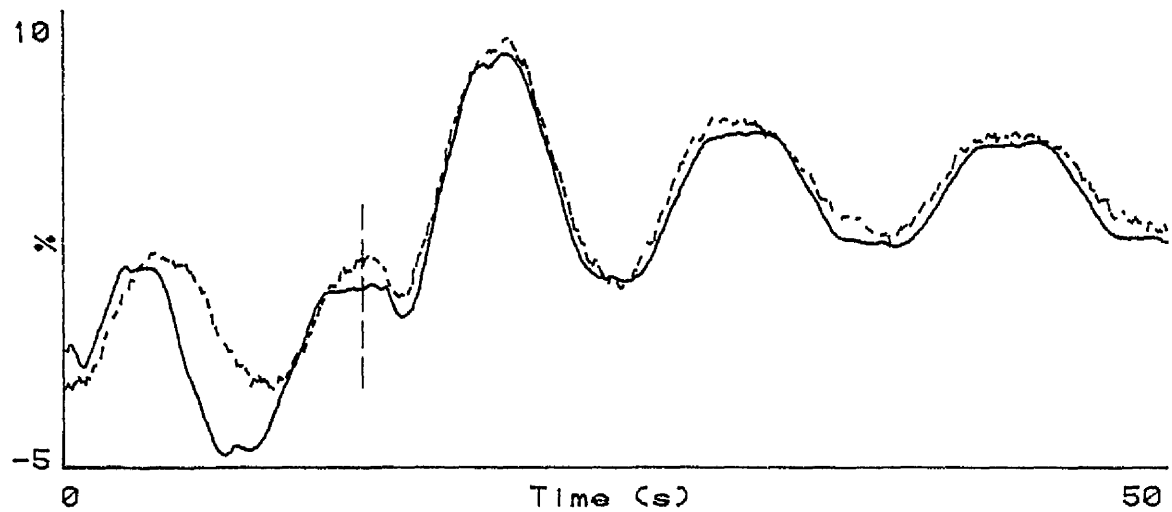
(Ta = 8.7s)



Frequency



Servo Position



Power Output

Figure 7.14 - Simulated Isolated Load

5% Step in Load from 25MW, kn=0.0

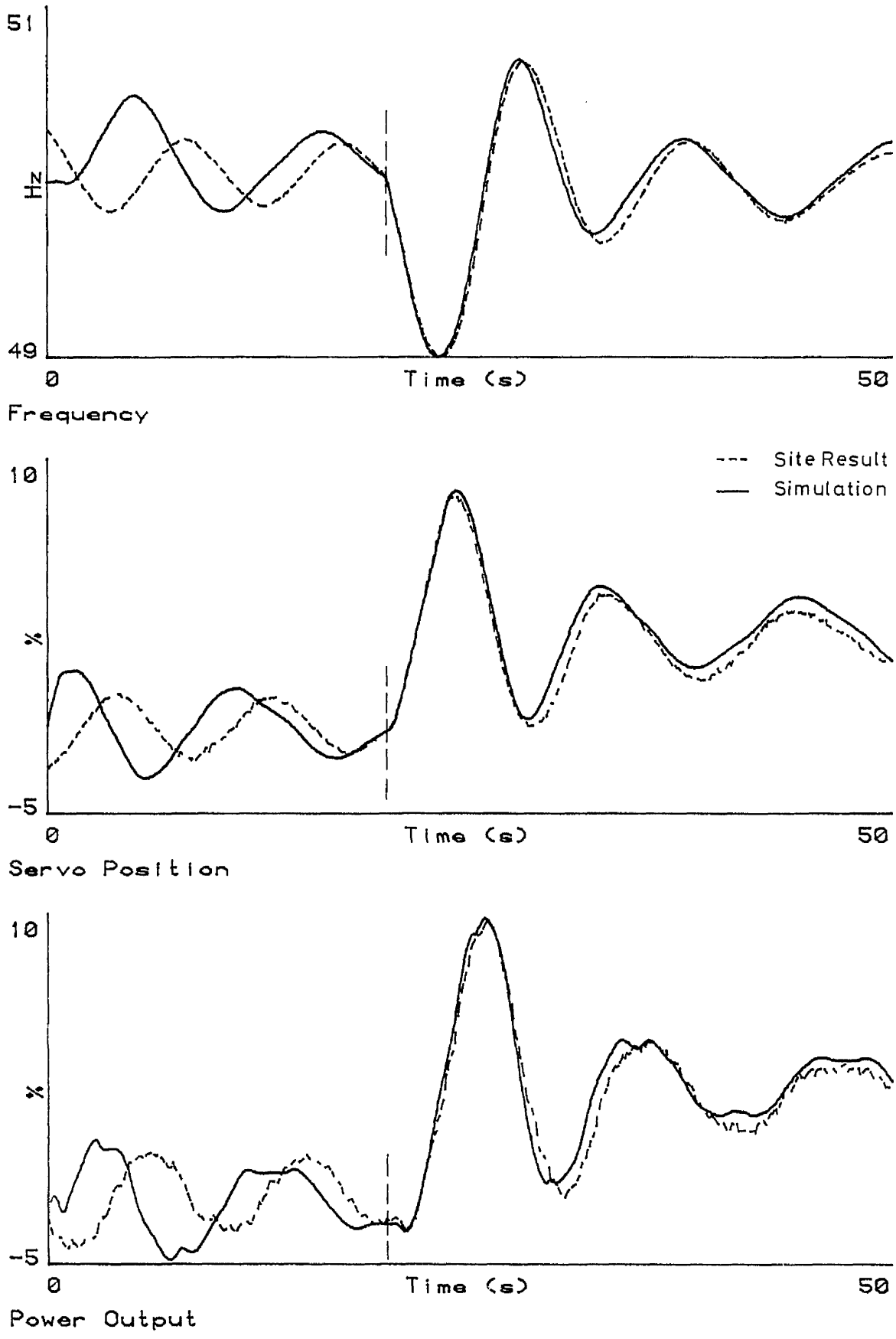
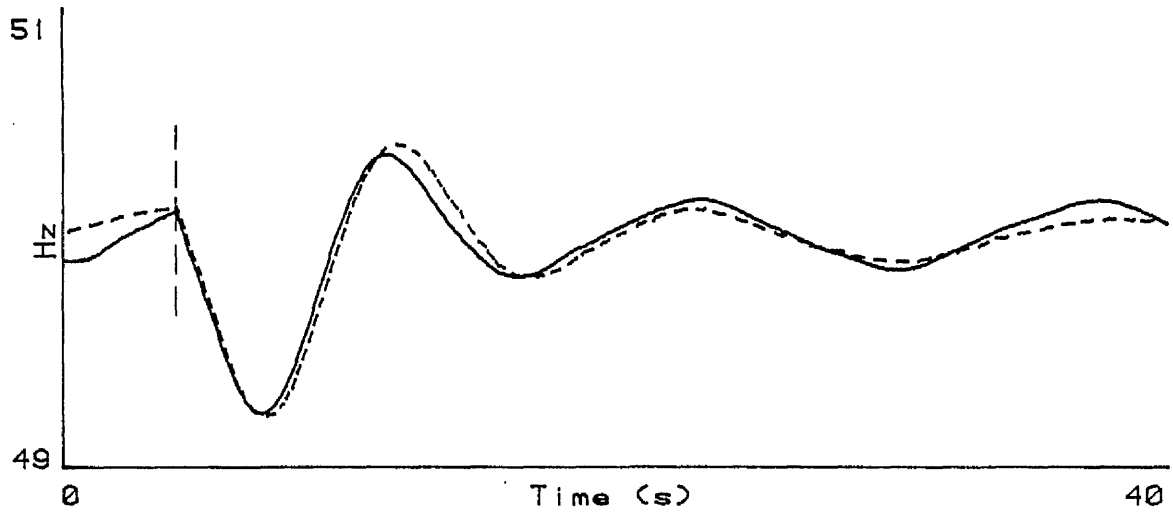
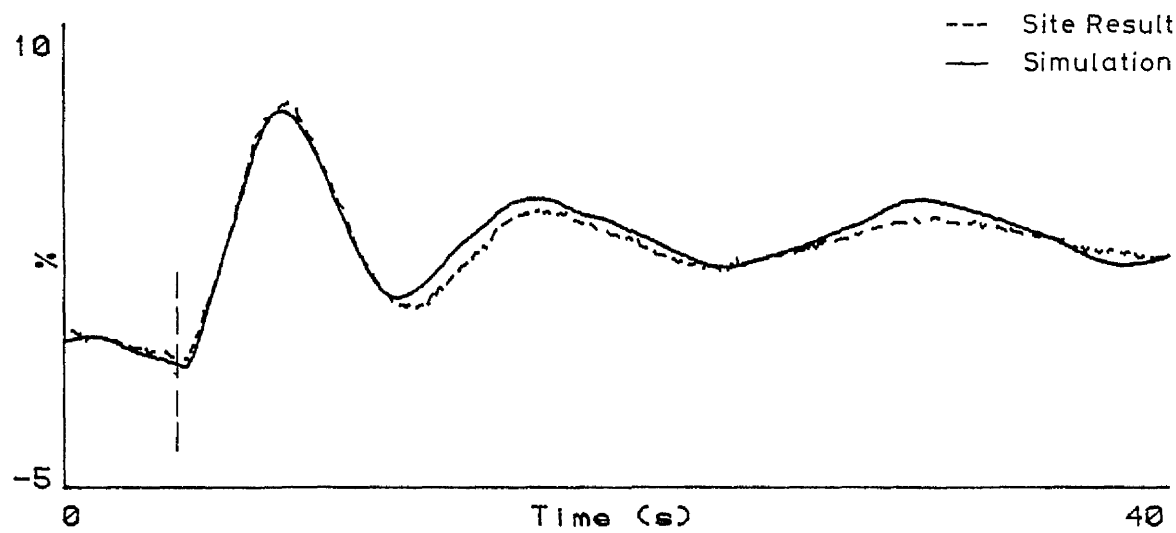


Figure 7.15 - Simulated Isolated Load

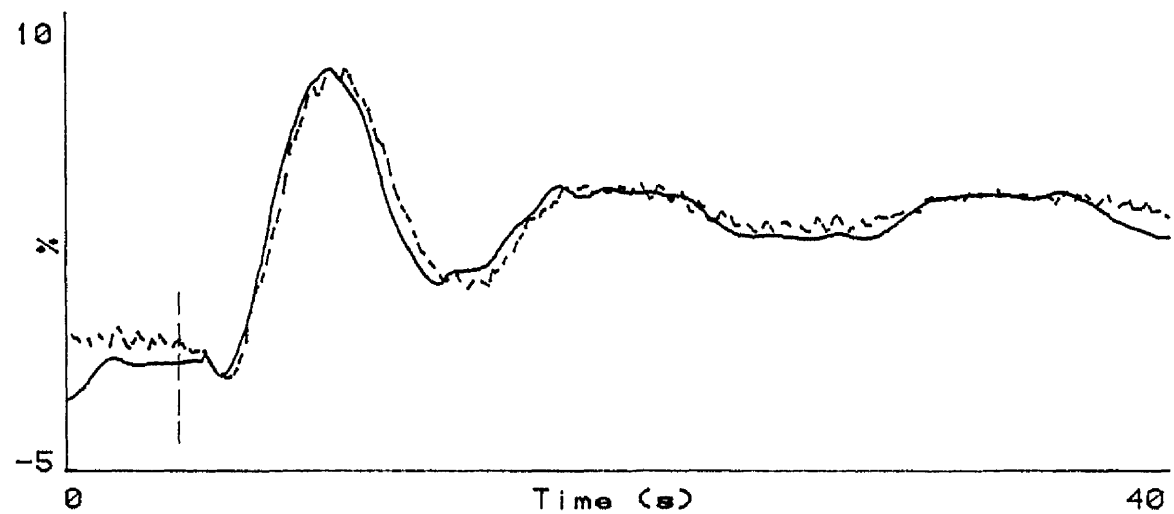
5% Step in Load from 25MW, kn=0.5



Frequency



Servo Position



Power Output

Figure 7.16 - Simulated Isolated Load

5% Step in Load from 25MW, kn=1.0

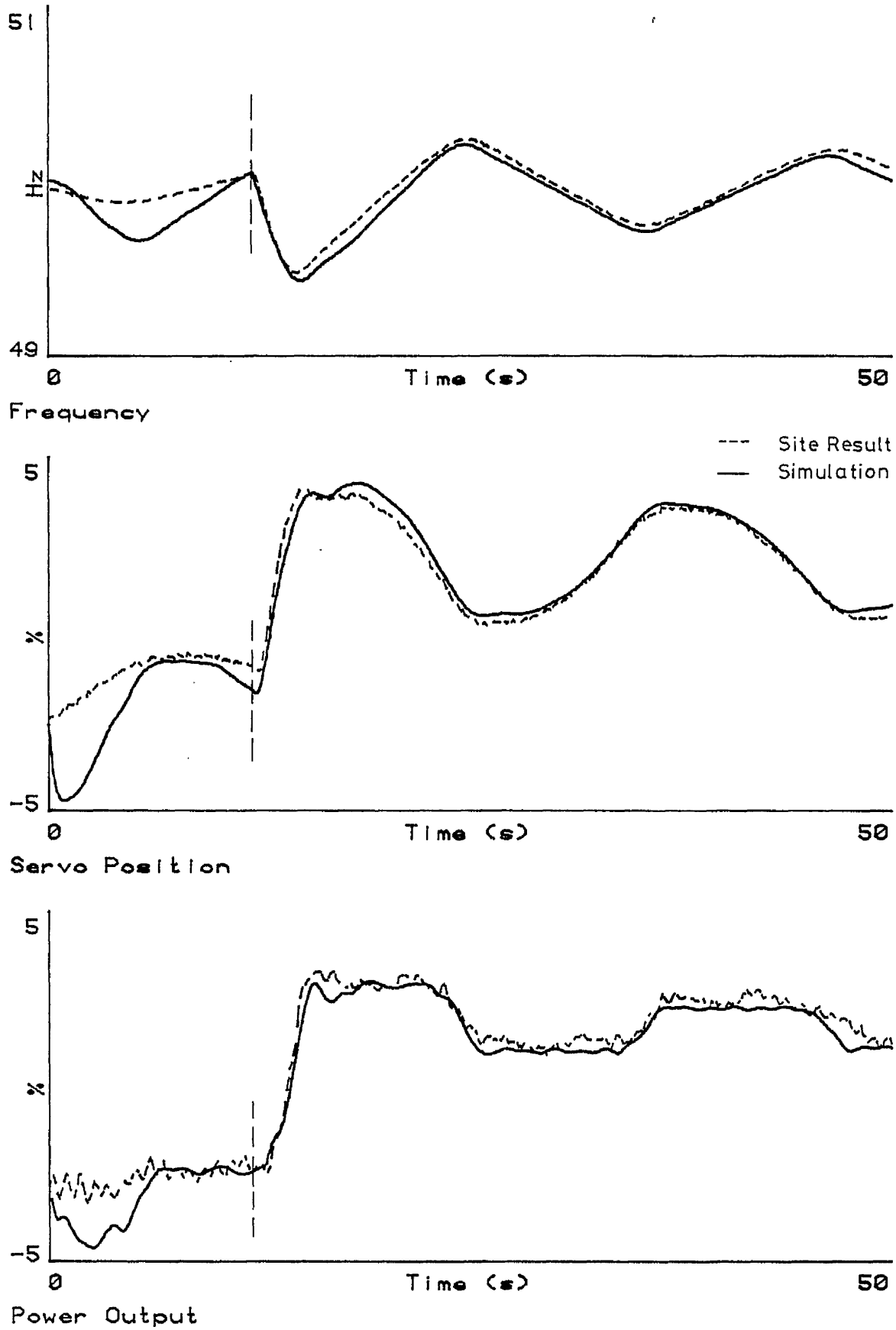


Figure 7.17 - Simulated Isolated Load

5% Step in Load from 5MW, kn=0.0

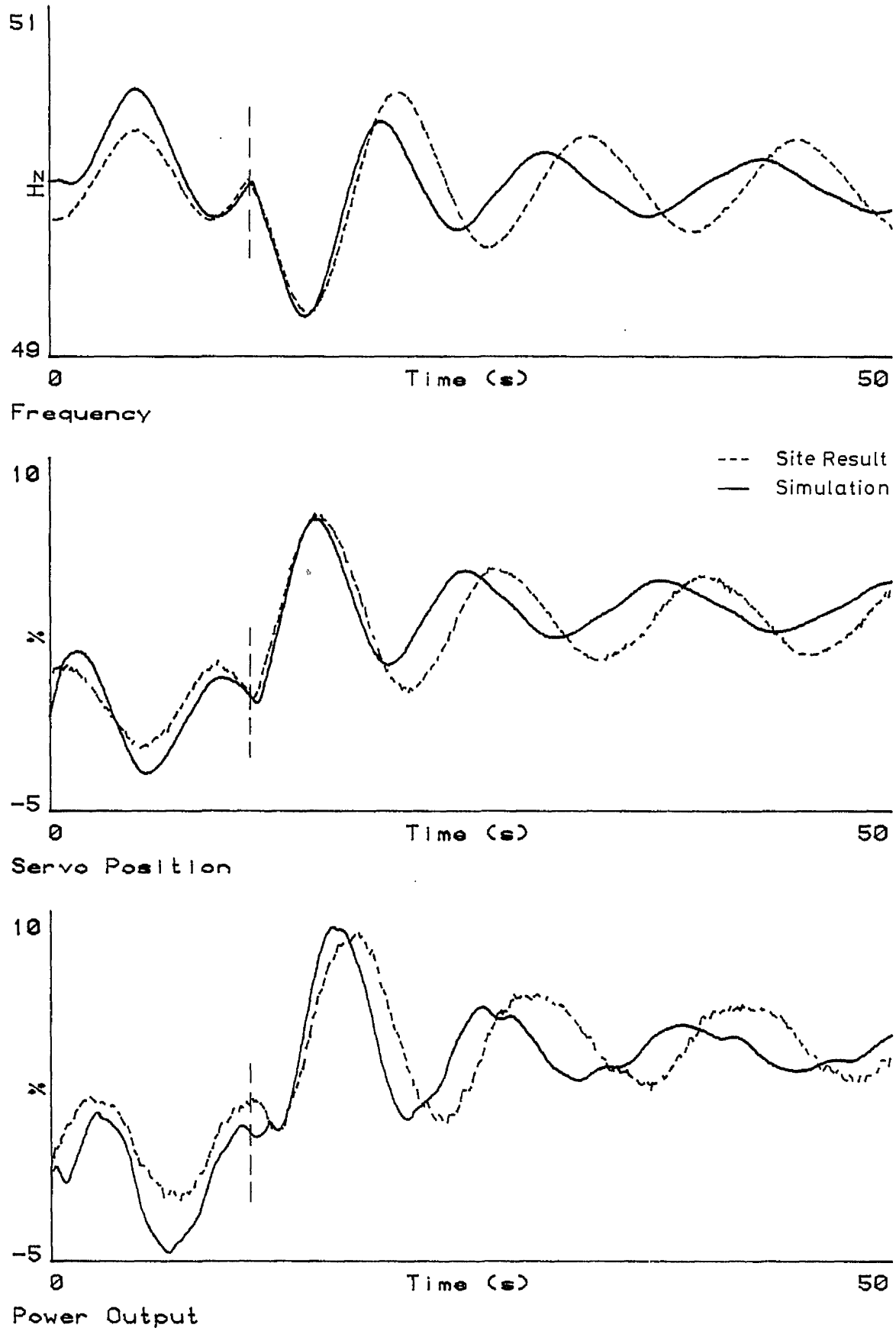


Figure 7.18 - Isolated Load Simulation

5% Step in Load from 25MW

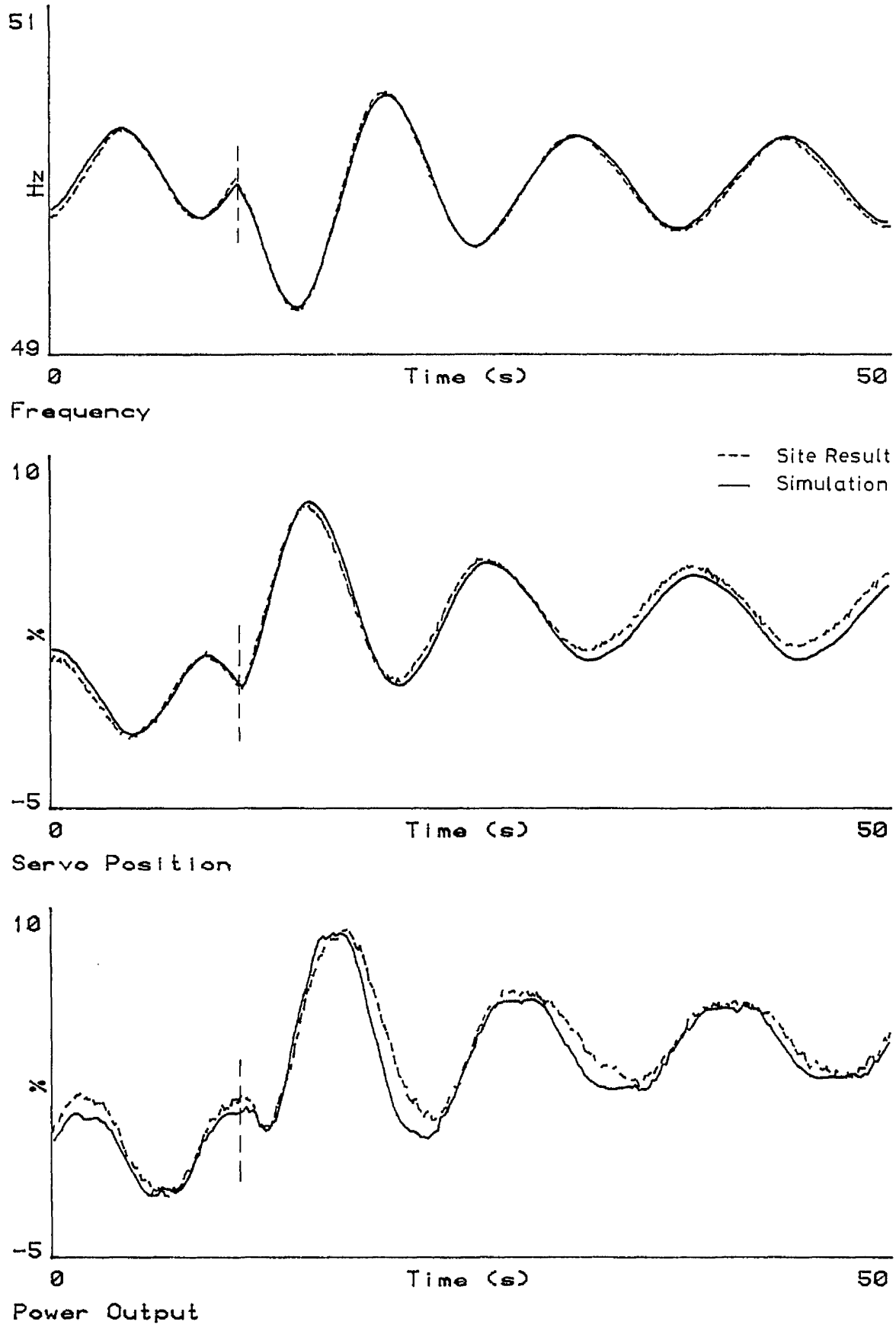


Figure 7.19 - Injected Site Signals

5% Step in Load from 25MW, kn=0.0

CHAPTER 8

AN INVESTIGATION OF A NON-LINEAR GOVERNOR

8.0 Introduction

The development of a fast acting governor for controlling hydro-generators has been fully documented[1,3,6] and in particular Reference 3 describes a microprocessor-based adaptive governor which has been tested experimentally at Loch Sloy Power Station.

Throughout all this work, the speed of response of new types of governors has been restricted by the need to maintain stability in the event of the generator being required to supply an isolated load. A governor which would respond quickly to grid connected frequency disturbances, particularly those associated with the loss of generating plant, and yet preserve stability under isolated load conditions, would have obvious advantages.

A "Rapid Response System" had been proposed[48] by Dr. D.J. Winning for use with the existing mechanical Temporary Droop Governor at Loch Sloy and some initial tests had been carried out. It was decided to investigate this system, renamed a Frequency Disturbance System (FDS), to see if some of the principles could be used in conjunction with the Double Derivative governor, to provide a governor which was both fast acting when grid connected and yet stable under isolated load conditions.

8.1 Original Proposal

The original proposal for the Rapid Response System involved interfacing with the existing mechanical governor through a speeder motor, and, by the use of an inverse governor algorithm, the relatively slow Temporary Droop dynamics could be bypassed. This would have resulted in a purely proportional governor action but under such

conditions the hydro generators at Loch Sloy are unstable and thus it was necessary to constrain the proportional action (as described in section 8.2.8).

To prevent inducing instability, the system was designed to detect abnormal rates of change of frequency and to discriminate between types of frequeny transients before taking any action. The types of frequency transients which might trigger the system, and the desired responses are outlined in section 8.3.

The action of the system, once triggered, was that of a constrained proportional governor acting on only short term frequency changes during the operational period (typically 10s). After this time, the system was reset and prevented from taking further action for a further period (typically 100s). During this latter period the output of the system was allowed to decay exponentially to zero to complement the action of the governor which would be responding over this period, the net output of the governor and the RRS together remaining unchanged. The basic structure of the system as it was implemented using analogue electronics is shown in Figure 8.1.

8.2 Description of the Frequency Disturbance System

Although this work is based on the original proposal for an RRS outlined above the development and the implementation of the FDS differ significantly in several ways. The RRS was built using analogue electronics as an addition or enhancement for the existing mechanical governor in Sloy Power Station (Figure 8.1), whereas the FDS would be implemented digitally as an integral part of the microprocessor controller at present in service in the Power Station. Thus, some of the ideas and techniques associated with the RRS are not applicable to the FDS and also the realistaion of the FDS using digital techniques requires the functions of the system to be expressed in a different

manner. In addition, although some initial site tests were carried out, many of the ideas behind the original proposal had not been fully investigated and no conclusions were reached as to the feasability, or otherwise of the RRS.

It was decided initially to design the FDS to give the best response for the type of frequency transients that would be experienced on an interconnected power system, bearing in mind the requirements for isolated load operation, and then test the system using a digital simulation of a hydro generator based on the model developed in Chapters 5 and 6. Subsequently the behaviour of the FDS under isolated load conditions could be checked using a slightly different simulation model.

The principal elements of the FDS, as implemented initially, are shown in the block diagram in Figure 8.2. The following sections outline the functions of the various blocks but details of the implementation are left until section 8.4 where the equations used for a digital simulation of the FDS are developed. The parameters of the system can all be varied as required and the values used in the simulation studies are given in the Documentation Files as noted in section 8.4. Initially the values were based, where appropriate, on those proposed for the RRS.

8.2.1 Overall Operation

The input to the FDS is frequency (f) and the output of the system (y_f) is a signal which adds into the governor output (y_g), along with a steady state loading signal, desired MW (y_s), to give the desired servo position signal (y_d). The processing of the input signal and the corresponding response of the system are detailed below.

8.2.2 Derivative and Rate Detectors

The FDS is triggered by rates of change of frequency outwith certain limits. The derivative block gives the rate of change of frequency and the rate detectors are used to determine if the rate of change exceeds the specified thresholds.

8.2.3 Timer and Phase Selector

If a rate of change of frequency which exceeds the specified limits is detected, the System Timer is started. This timer runs throughout the active period of the system response and is used to determine the phase of operation to be executed at any particular time.

(i) Run Phase

The first phase of operation is the RUN phase during which the peak followers are enabled and the output of the proportional governor (Average and Droop) is selected.

(ii) Payoff Phase

Following the Run phase, the next phase of operation is the PAYOFF phase during which the output of the FDS ramps from the value reached at the end of the Run phase to zero. The peak followers and the proportional governor are disabled during this phase.

(iii) Inhibit Phase

After the end of the Payoff phase the FDS is inhibited from retriggering for a specified time. During this phase the outputs of the peak followers and the proportional governor are set to zero.

(iv) Reset Phase

Following the Inhibit phase the FDS is RESET to the quiescent state and triggering is re-enabled. At the end of this phase the System Timer is set to zero.

8.2.4 Washout

The frequency used by the peak followers is first passed through a "washout" filter. This is a derivative with a lag, the output of which is zero during slow frequency changes but follows any rapid changes in frequency. This has the effect of removing the steady state frequency and thus the FDS responds only to transient frequency errors and not steady state errors.

8.2.5 Peak Followers and Proportional Governor

In a purely proportional governor, the demanded servo position is linearly related to frequency error. All frequency oscillations are thus transmitted to the servo and with the generator operating with an isolated load, this type of governor has been shown to result in unstable conditions. The proportional governor however, does offer the advantage of a very rapid response to variations in system frequency and the FDS has been designed to take advantage of the rapid response without inducing oscillations.

A positive and a negative peak follower are set to track the minimum and maximum value of frequency during any disturbance which has started the System Timer and the proportional governor acts on the average of the outputs of the peak followers. The average of the output of the peak followers is dependent on the shape of the frequency wave form and is not simply the mean of the waveform. The peak followers in effect provide a measure of non-linear filtering which removes any superimposed oscillations and yet transmits an approximate frequency error to the proportional governor. The gain (or droop) of the proportional governor determines the relative effect of the FDS on the desired servo position signal compared to the contribution from the Double Derivative governor.

Initially the peak followers were set such that only the one follower tracked the initial transient, the other remaining at zero until, if at all, the frequency signal passed through the initial level, travelling in the opposite direction. This had the effect of halving the nominal gain of the system for any frequency transient resulting in a steady state offset in servo position. Thus, it was considered that an alternative approach might be to allow both peak followers to track the initial frequency transient and then follow the minimum and maximum values of the disturbance. This would have resulted in a system gain close to the nominal value as the average of the peak followers would have been approximately the mean of the frequency transient. However, as shall be seen later, the need to detect a zero crossing of the frequency signal, required that one of the peak followers remain at zero thus this alternative was not implemented.

8.3 Frequency Disturbances

To investigate the behaviour of the FDS using digital simulation techniques, it is first necessary to simulate the types of frequency transients to which the system may have to respond.

By considering typical system frequency disturbances and the behaviour of the set at Sloy, it was decided that there were basically five different types of frequency disturbances which a FDS would have to detect and discriminate between.

8.3.1 Generation Loss

The type of frequecy transient caused by the loss of generating plant somewhere on the system is shown on Figure 8.3(a). This Figure is derived from the results of a frequency transient caused by the loss of a 300MW generator at Longannet Power Station in July 1972.

This is the type of frequency disturbance that the FDS is required to react to so that hydro plant can quickly pick up load and restore the system frequency in the event of the loss of generating plant. (The behaviour of thermal plant under these conditions and the suitability of hydro plant to meet generation deficits is discussed in Chapter 9 and additionally in Reference 49.)

To simulate this type of frequency transient two second order systems were used, one (system B) for the overall shape of the inital response and the other (system A) for the superimposed oscillation. A ramp function was used to represent the long term fall in frequency after the inital transient. The values of the natural frequency (ω_n) and the damping (ξ) were chosen from analysis of the response in Figure 8.3(a) and are given in Figure 8.4. The simulated frequency disturbance is shown in Figure 8.3(b) for comparison with Figure 8.3(a). The Model Description is given in Appendix 2.8 and the Documentation File in Appendix 3.25.

To test the FDS with this frequency transient a Frequency Signal Generator section was included in a simulation of the Sloy hydro-generator (as developed in Chapter 6) and a section was added to represent the FDS. By a choice of suitable values for the parameters the Frequency Signal Generator could be used for this and the following two frequency disturbance simulations. In addition, a special case of the generation loss disturbance with no superimposed oscillation could be simulated by the this Frequency Signal Generator.

The Model Description used initially for these simulation studies is given in Appendix 2.9. It should be noted that, as this work was completed before the hydro-generator model was finalised, some of the equations may differ slightly from those used in Chapter 7.

8.3.2 Synchronising Transient

The synchronising transient at Sloy is a lightly damped oscillation which could also be caused by line swtching, lightning and other disturbances on the system. Since, in all these cases, the disturbances are not associated with loss of generation the FDS should ideally take no action as under certain circumstances this could result in the oscillation being sustained. The frequency of this oscillation is a function of the load on the generator, decreasing as the load increases.

A second order system with natural frequency (ω_n) and damping (ξ) as given in Figure 8.4 was used to simulate this type of frequency disturbance and the simulated output is shown in Figure 8.5. The frequency of the oscillation used for the simulation studies was that observed during the Longannet Tests when the machine was loaded to 25MW.

8.3.3 Heavily Damped Oscillation

Another type of frequency transient which could occur, although unlikely at Sloy where the damping is light, is a heavily damped oscillation, as shown in Figure 8.6. Again the FDS should not respond to this type of disturbance.

A second order system, with parameters as given in Figure 8.4, was used to model this transient.

8.3.4 Isolated Load Operation

It is essential that the FDS does not induce instability during frequency disturbances when the generator is supplying an isolated load. A typical isolated load frequency disturbance is shown in Figure 8.7 recorded during site tests at Loch Sloy using an isolated load simulator (section 6.4.1) and the Double Derivative governor (Appendix 3.14).

To test the performance of the FDS developed for the frequency transients given above the equations for the Sloy hydro generator operating grid connected were modified to represent isolated load operation and a series of step tests were carried out (see section 8.4.3). The Model Description for this simulation study is given in Appendix 2.11.

8.3.5 Load Rejection

The final type of frequency transient to which the FDS may be required to react is that which occurs during a load rejection, i.e. when the generator is disconnented from the system by, for example, a circuit breaker being opened. Use of the FDS could be beneficial in reducing the overspeed caused when the generator becomes isloated from the system. A typical load rejection frequency transient recorded on site during commissioning tests using the Double Derivative governor is shown in Figure 8.8.

To investigate the behaviour of the FDS during these conditions the equations of the FDS were included in a simulation model which had been used for load rejection studies (Chapter 6). The resulting Model Description is given in Appendix 2.12.

8.4 Implementation and Development of the Frequency Disturbance System

The basic implementation of the FDS is shown in the flow chart in Figure 8.9 and the equations for this implementation can be found in the Model Description in Appendix 2.9. To ensure the FDS System Timer is only updated at the integration interval (H) and not at intermediate times when using a Runge-Kutta integration technique the Simulation Language variable M is tested and only if this variable is 1 is the Timer incremented. (Further details of this feature can be found in the GUILDS User's Guide at the end of this Thesis.)

The results of simulation studies using this implementation in conjunction with the frequency disturbances given above are presented in the following sections along with the modifications made to the FDS in the light of some of the results. In all the simulation studies, except where otherwise stated, the disturbances applied were kept as small as possible (typically 0.1%) to avoid the servo rate limits which would otherwise obscure the true response of the FDS.

8.4.1 Performance of FDS with a Generation Loss Transient

The first simulation study was carried out using the Model given in Appendix 2.9 and the Documentation Files containing the values of the variables for the study are given in Appendix 3.28 and 3.29. The Frequency Signal Generator was used to model a generation loss transient (as discussed in section 8.3.1) and the results of the simulation study are shown in Figure 8.10.

Figure 8.10(a) shows the frequency transient produced by the Frequency Signal Generator. The output of the FDS and the Double Derivative governor are shown together in part (b) of the Figure. Part (c) shows the desired servo position signal (the sum of the governor output, the FDS output and the desired MW signal) both with and without the FDS active. The desired servo position signal is repeated in part (d) of the Figure along with a plot of the action that would have been taken by a purely proportional governor. Since the proportional governor is, in effect, an "ideal" governor the FDS should emulate this response as closely as possible.

From these results is can be seen that the FDS offers a significant improvement over the Double Derivative governor for this type frequency disturbance as the servo response is much faster. In addition it should be noted that the response of the servo with the FDS active is close to that produced by the proportional governor governor, without the undesirable oscillations being transmitted to the servo.

The response of the hydro-generator in this simulation has no effect on the frequency transient as the the model is effectively open loop. This is the case at Sloy and all other hydro stations in the NSHEB area where the power output of the generator is small compared to the size of the grid system and thus the sets would have relatively little effect on the system frequency. It would be possible, using a combined hydro-thermal simulation (as discussed in Chapter 9) to model a closed loop system where a large hydro-generator, for example a 400MW set at Craig Royston, could have a significant effect on the frequency transient. It may be necessary to carry out some further studies if it were proposed to use the FDS with larger hydro-generators.

8.4.2 Performance of the FDS with Oscillatory Transients

Further simulation studies were carried out with the model described above using the Frequency Signal Generator to simulate oscillatory transients as described in sections 8.3.2 and 8.3.3. The relevant Documentation Files for these studies are to be found in Appendix 3.30 to 3.35.

For the first case, with a lightly damped synchronising transient, the results of the simulation study are given in Figure 8.11. The frequency disturbance is shown in part (a) of the Figure; the outputs of the FDS and the governor (broken line) are shown in part (b); and the desired servo position both with and without (broken line) the FDS active is shown in part (c).

From these results it can be seen that the FDS considerably degrades the respose of the governor to the frequency transient. The servo movement is increased about five times on the first peak of the transient and the subsequent position of the servo is offset from the steady state position by about 1.2%.

To detect this type of frequency disturbance it was proposed that an escape mechanism be included in the FDS algorithm whereby, if the magnitude of the second peak was greater than a certain fraction (typically 0.5) of the first peak and in the opposite direction, the output of the FDS would be reset to zero and the Inhibit phase entered.

The results of the simulation with this escape mechanism included are presented in Figure 8.12. The effect of the escape mechanism on the servo movement can be seen from the results. The performance after the first peak has been considerably improved, but the initial transient is still much larger with the FDS than without.

The FDS developed above was also used in a simulation with the Frequency Signal Generator producing a heavily damped oscillation. The results of this study are presented in Figure 8.13. As the second peak of this oscillation does not rise to half the magnitude of the first, the escape mechanism is not activated and a large transient and subsequent offset in servo position is produced. This is obviously undesirable and thus it was decided to change the FDS such that the escape mechanism was activated by the first zero cross-over of the (washout) frequency signal after the FDS had been activated.

The effect of this change can be seen by comparing the results shown in Figure 8.14 with those in Figure 8.13. As there is now no net offset in the desired servo position signal it is obvious that the response has been improved but the initial transient is still considerably larger that that produced by the governor acting without the FDS.

To eliminate this problem it was decided to hold the output of the FDS at zero for a certain time until it was known whether the transient was oscillatory (indicated by a zero cross-over) or due to a generation loss. As the frequency of the oscillations, which is characteristic of the set at sloy, was 0.5Hz it was decided to hold the

output at zero for a time greater than the half period of the oscillations (typically 1.5s). If a zero cross-over is detected within this time the escape mechanism is activated and there is no output from the FDS. Otherwise the output of the FDS assumes the value it would have reached, had the output not been held. It may be necessary to increase this timing for loadings other than that being simulated or for sets other than at Sloy (see above).

The effect of the delay in output for the three frequency transients discussed above are shown in Figure 8.15. Part (a) of this Figure shows the desired servo position with the FDS active for a generation loss transient. By comparison with Figure 8.10 (c) it can be seen that no appreciable difference has been made by holding the output at zero for the first 1.5s of the transient.

Parts (b) and (c) of Figure 8.15 show the response of the servo to the two types of the oscillatory transients. In both cases the FDS escape mechanism is activated before the hold time is passed and thus there is no output from the FDS. The response is these cases is therefore the same as shown in Figures 8.11 and 8.13 with the FDS not active.

A flow chart of this improved FDS algorithm is shown in Figure 8.16 and the equations for this system can be found in the Model Description in Appendix 2.10. The Documentation Files for the results shown in Figure 8.15 are given in Appendix 3.36 to 3.38.

8.4.3 Performance of the FDS with an Isolated Load

Having obtained an FDS which was acceptable under grid connected operation, a series of simulation studies were carried out with this system to investigate the behaviour of the FDS under isolated load conditions. The Model Description for these studies is to be found in Appendix 2.11.

Figures 8.17, 8.18 and 8.19 ahow the response of the hydro generator to step changes in the isolated electrical load both with and without the FDS active. In all the figures the broken lines represent the response without the FDS. The corresponding Documentation Files for these figures are in Appendix 3.37 to 3.45.

Under isolated load conditions the frequency of the generator output and the water control valve exhibit limit cycles due to backlash (hysteresis) in the linkages to the water control valve (see Chapter 6 and also Reference 3). In Figure 8.17 the step change in load was 0.1% and the initial frequency transient was not fast enough to activate the FDS. The FDS is in fact triggered by the start of the limit cycling although normally the limit cycles would not activate the FDS. As soon as the zero crossing of the frequency signal is detected, the FDS escape mechanism is activated and the transient caused by the FDS action dies out in about one period of the limit cycles. The transient on the frequency signal would be unlikely to cause any problem but the rapid movement of the servo in the closing direction may give rise to a transient opening of the relief valve, which may not be acceptable.

In Figure 8.18 the response of the hydro generator to a 1% step is shown. In this case the initial frequency transient triggers the FDS with the result that the size of the initial transient is reduced but the subsequent recovery transient is larger. Again the effect on frequency is not substantial but the movement of the servo has been increased by about 5%. The effect of the servo rate limits on the response can be seen in the Figure.

In the third study, the step size was increased to 10% as shown in Figure 8.19. In this case the servo rate limits restrict the action of the FDS such that there is virtually no difference in the responses with or without the FDS active. Figure 8.20 shows the same

test repeated but this time the rate limits have been removed to represent other stations where, because the pipelines are much shorter than at Sloy, the rate limits are less severe. The action of the FDS reduces the size of the initial transient, compared to that in Figure 8.19, and the recovery transient has also been improved. However, this latter point is a direct result of the relief valve opening due to the rapid closing of the servo when the FDS escape mechanism is activated on the zero crossover. The onset of limit cycling can be seen towards the end of the simulation run as the relief closes and the flow conditions return to the steady state.

The results of these studies show that the action of the FDS under isolated load conditions, although marginally degrading the governor response, does nothing to impair the stability. It may be possible to improve the isolated load behaviour of the FDS by adjusting some of the system parameters, for example the 'hold' time, but this would almost certainly impair the response of the FDS to grid connected frequency transients. It would be necessary to test the FDS on site to find out if the isolated load would be acceptable under operational conditions, given that the likelihood of such circumstances arising in practice is somewhat remote.

8.4.4 Performance of the FDS during a Load Rejection

Figure 8.21 shows the results of a simulation of a 50% load rejection both with and without (broken lines) the FDS active. The Documentation Files for this study are given in Appendix 3.46 and 3.47.

It is clear from the Figure that the FDS reduces the initial frequency overshoot but, as a result, the subsequent recovery transient is more oscillatory. As in the case of the isolated load operation, it would be desirable to study the behaviour of the FDS on site before deciding if this response would be acceptable under operational conditions.

8.5 Conclusions

From the results of the simulation studies presented in this Chapter it can be seen that the FDS, with the improvements outlined in section 8.4.2, would enhance the response of a hydro generator to frequency disturbances of the type caused by the loss of generating plant. This would enable the hydro plant to pick up load faster and either reduce the size of frequency transients experienced by the power system or permit the amount of spare thermal plant on the system to be reduced.

In addition, it has been shown that the performance of the FDS for other frequency transients, either grid connected or isolated is acceptable in that stability of the hydro generator is not effected. However, it is recommended that some on-site tests are carried out so that, particularly under isolated load conditions, the simulation results can be confirmed and assesed in terms of any additional wear or strain imposed on the mechanical/hydraulic equipment.

Site tests could be readily carried out using the controlled testing facilities developed at Sloy Power Station (Reference 18). The grid connected response could be tested by using simulated frequency transients, as above, from either digital or analogue equipment which could be injected into the FDS and the existing Double Derivative microprocessor governor[3,6]. The Isloated Load Simulator (section 6.4.1) could be used for testing the isolated load response of the FDS and the load rejection tests could be carried out with no additional equipment required, as for the Commissioning Tests[6].

The value of GUILDS as a design tool is illustrated by this work. The original proposal for a Rapid Response Governor has been implemented as a simulation model and fully tested, modifications to the design have been made, some of the tests repeated and proposals for site work drawn up. This has all been effected without the need to

invest in any equipment or site facilities until the proposal has been shown to be feasible. A certain amount of optimisation of the design could still be carried out but this would now be facilitated by some site data which would shown up any deficiencies in the model.

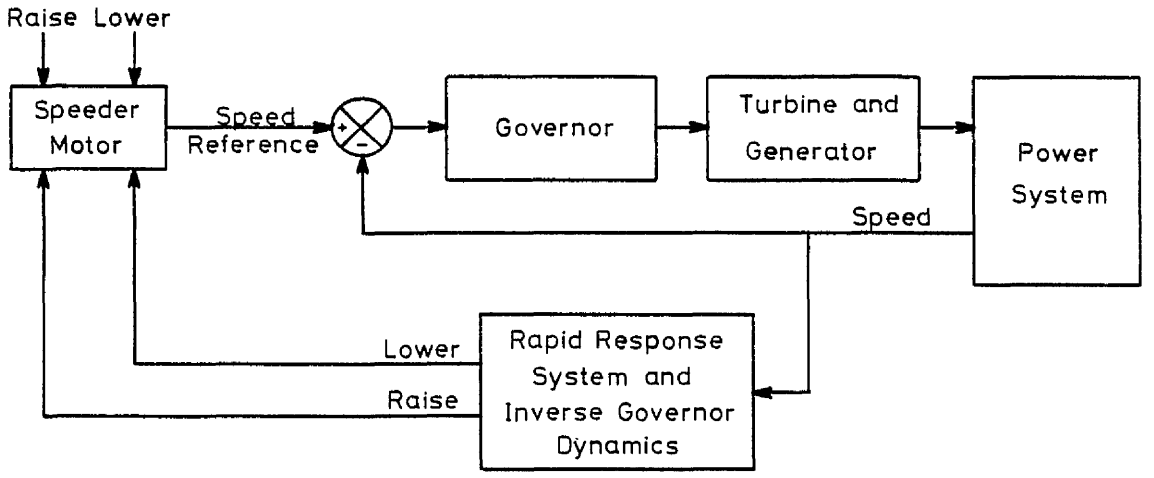


Figure 8.1 - Practical Implementation of Rapid Response System

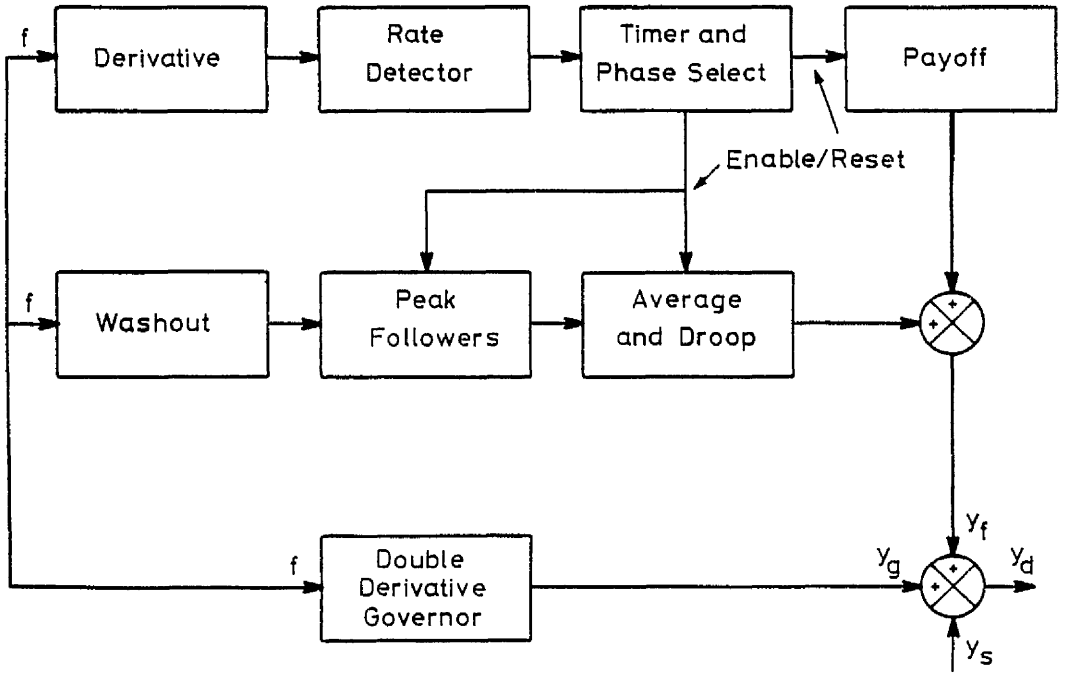
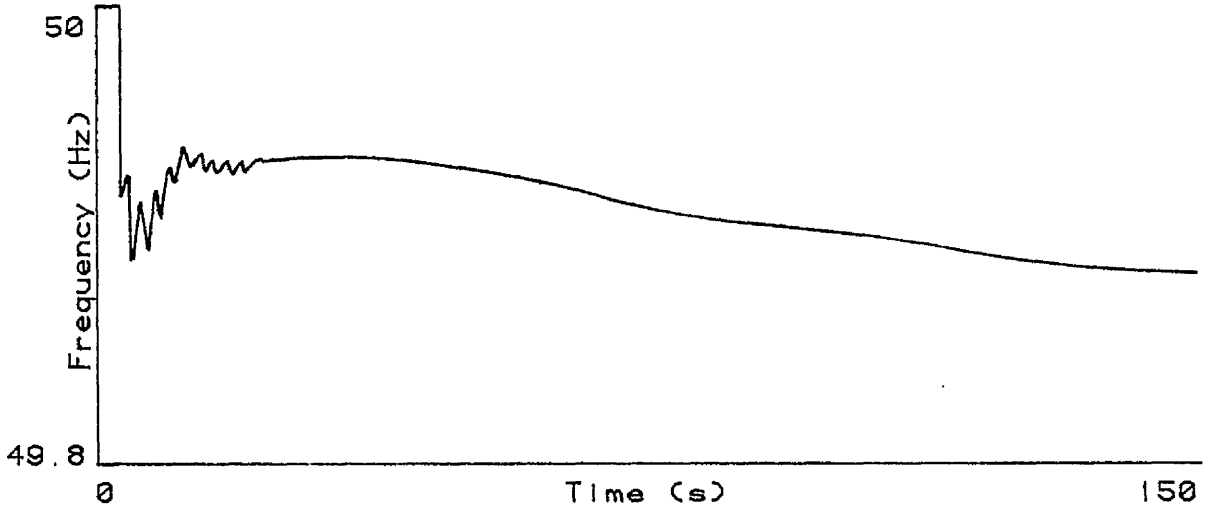
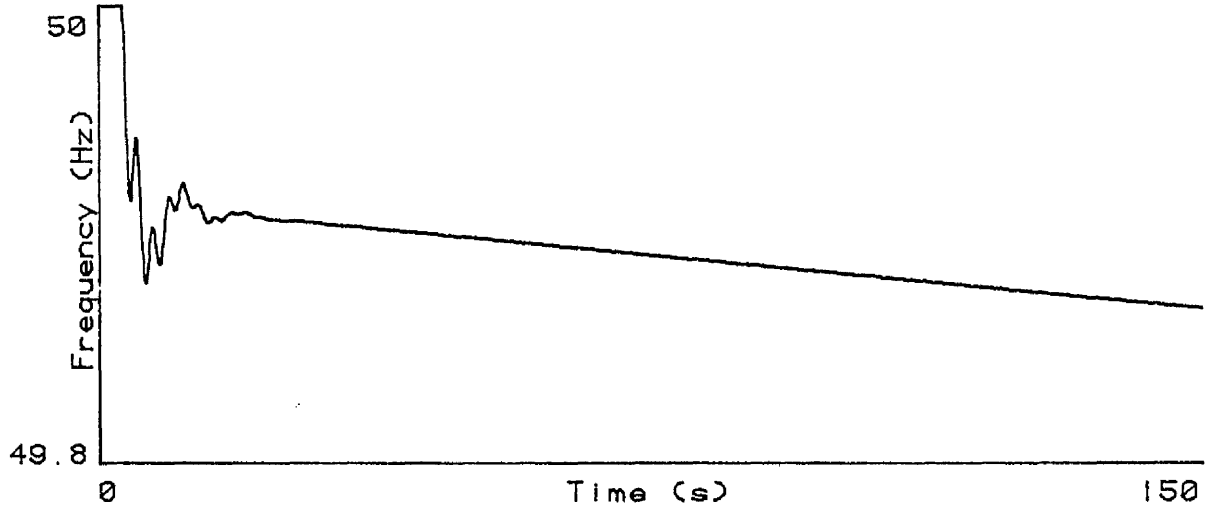


Figure 8.2 - Principal Elements of Frequency Disturbance System



(a) Transient from Longannet Trip



(b) Simulated Generation Loss Transient

Figure 8.3 - Generation Loss Transients

Model Equations for Frequency Signal Generation

```
AK1=2*DA/WA
AK2=WA*WA
BK1=2*DB/WB
BK2=WB*WB

X=ST*STEP(5.)
YA=INTGRL(0.,YDOTA)
Y2DOTA=(X-AK1*YDOTA-YA)*AK2
YDOTA=INTGRL(0.,Y2DOTA)
YB=INTGRL(0.,YDOTB)
Y2DOTB=(X-BK1*YDOTB-YB)*BK2
YDOTB=INTGRL(0.,Y2DOTB)
YC=RAMP2(TIME-13.,RR,1.,0.)
YD=PA*YA+PB*YB+PC*YC+PD*YDOTA+PE*YDOTB+PF
```

Model Parameters for Frequency Signal Generation

PARAMETER	TRANSIENT		
	Generation Loss	Synchronising	Heavily Damped
PA	-0.8	0.0	0.0
PB	-1.0	0.0	0.0
PC	1.0	0.0	0.0
PD	0.0	1.0	1.0
PE	0.0	0.0	0.0
PF	0.0	1.0	1.0
WA	3.0	3.0	3.0
DA	0.09	0.09	0.5
WB	0.8	-	-
DB	0.28	-	-
ST	0.001	0.001	0.001
RR	-6.2E-6	-	-

WA - Natural frequency of sub-system A
DA - Damping of sub-system A
WB - Natural frequency of sub-system B
DB - Damping of sub-system B
ST - Step size
RR - Ramp rate

Figure 8.4 - Frequency Signal Generator

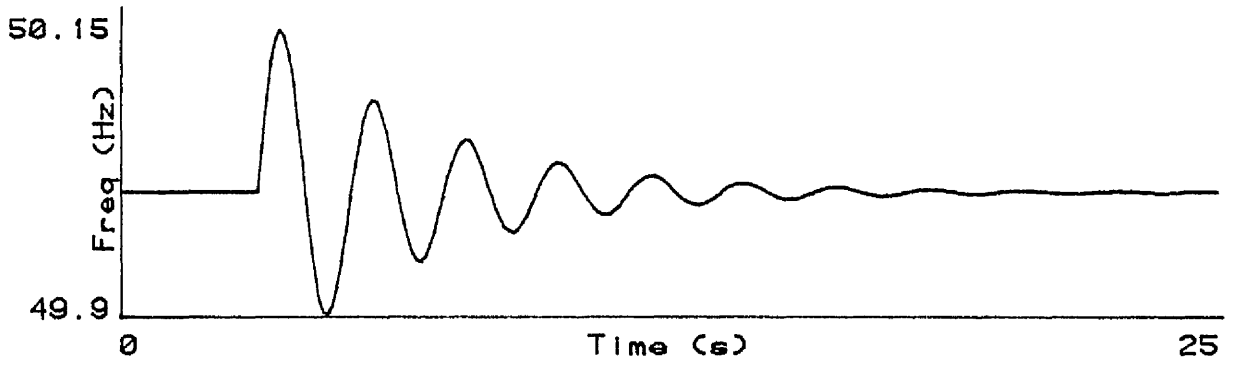


Figure 8.5 - Synchronising Transient

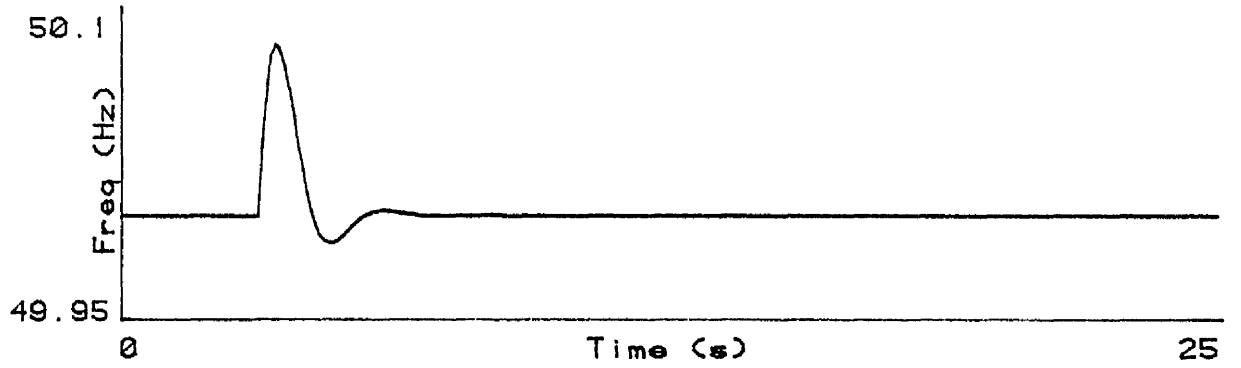


Figure 8.6 - Heavily Damped Transient

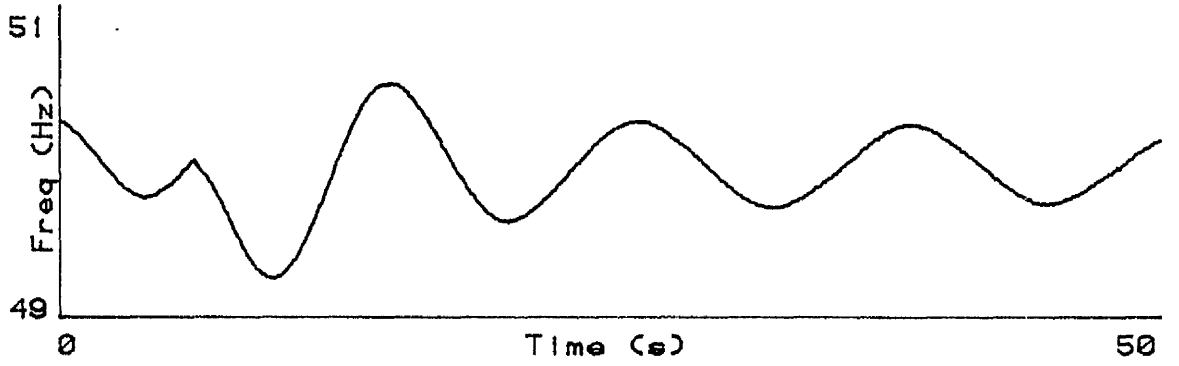


Figure 8.7 - Simulated Isolated Load Test (5% Step)

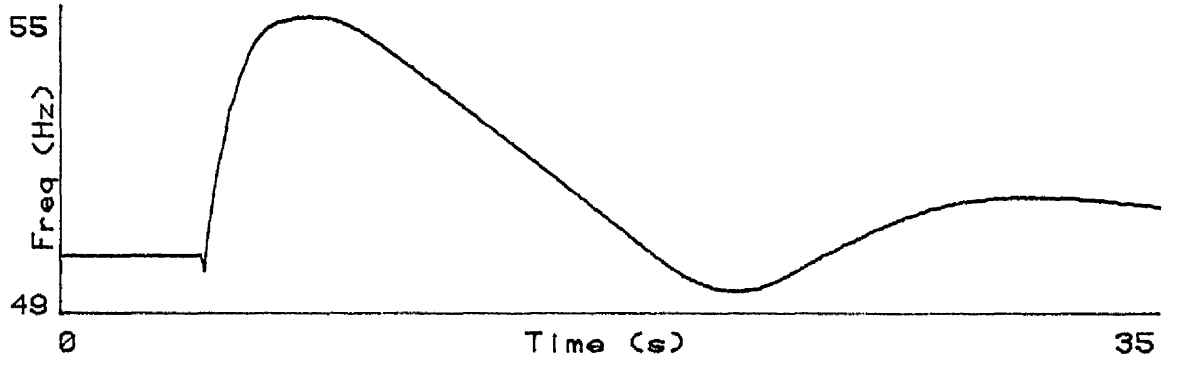
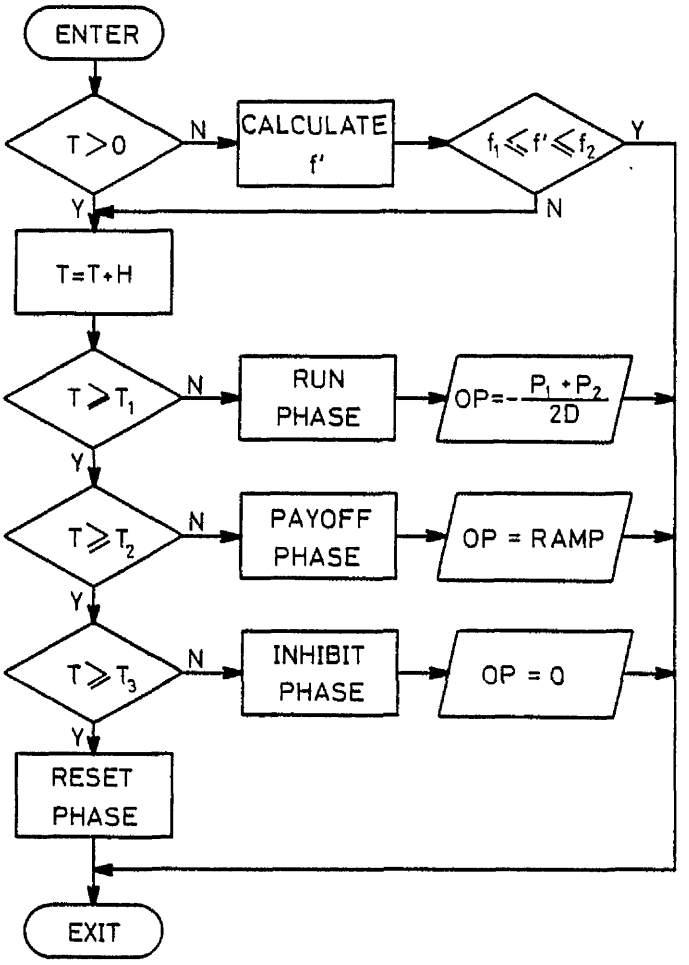


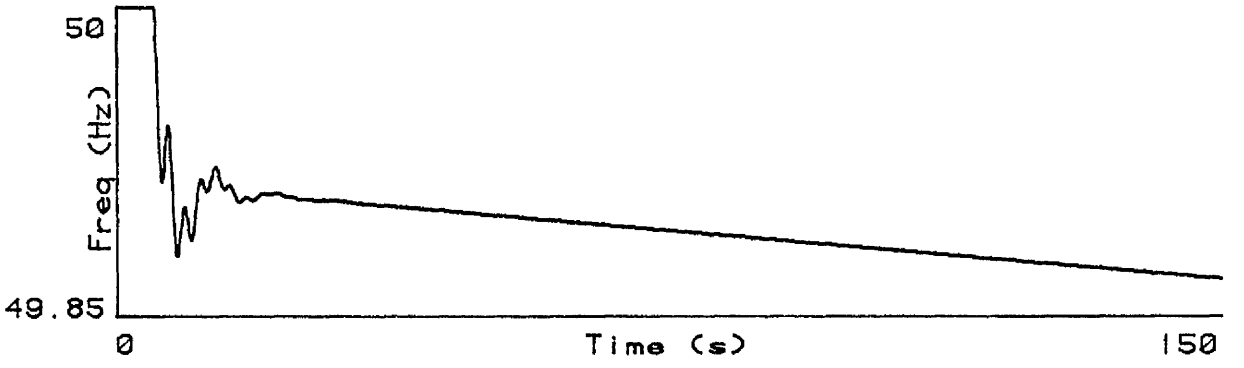
Figure 8.8 - Load Rejection Site Test (15MW)



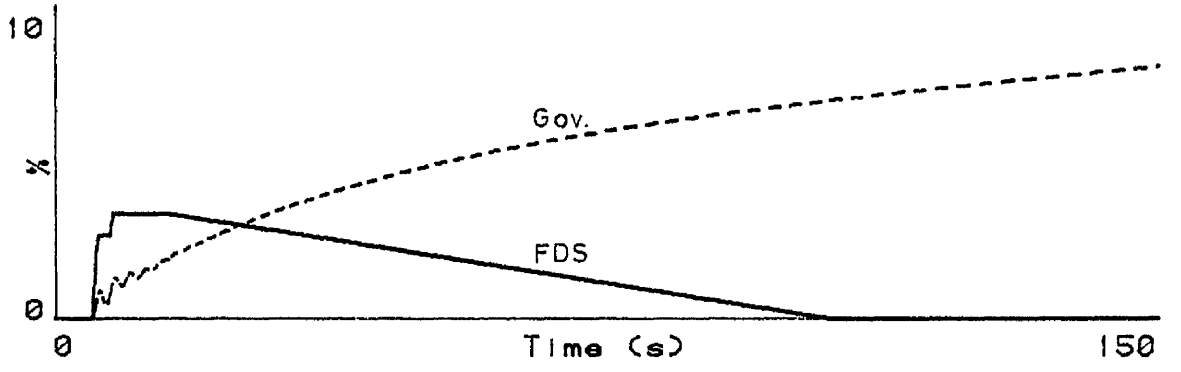
T - Timer
T_1 - Run Phase Time
T_2 - Payoff Phase Time
T_3 - Inhibit Phase Time
OP - Output of FDS

H - Integration Interval
f´ - Derivative of Frequency
f_1, f_2 - Limits on f´
P_1, P_2 - Peak Followers
D - Droop

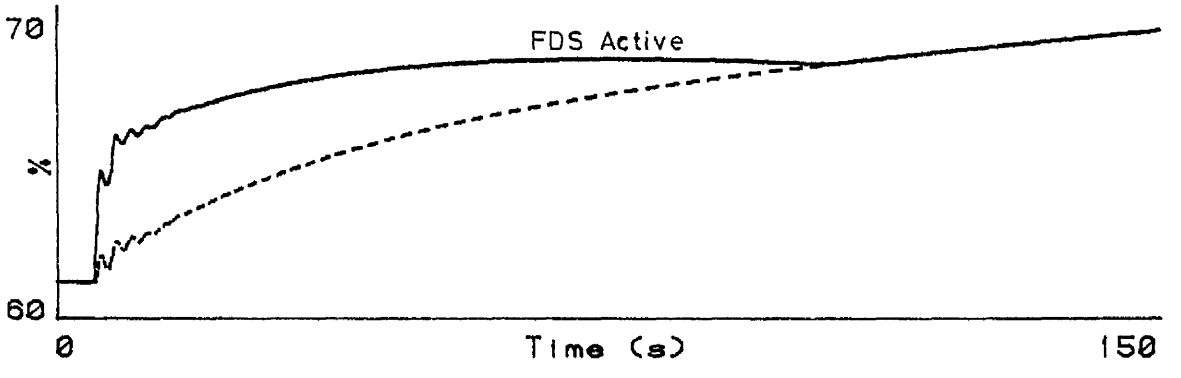
Figure 8.9 - Initial Implementation of Frequency Disturbance System



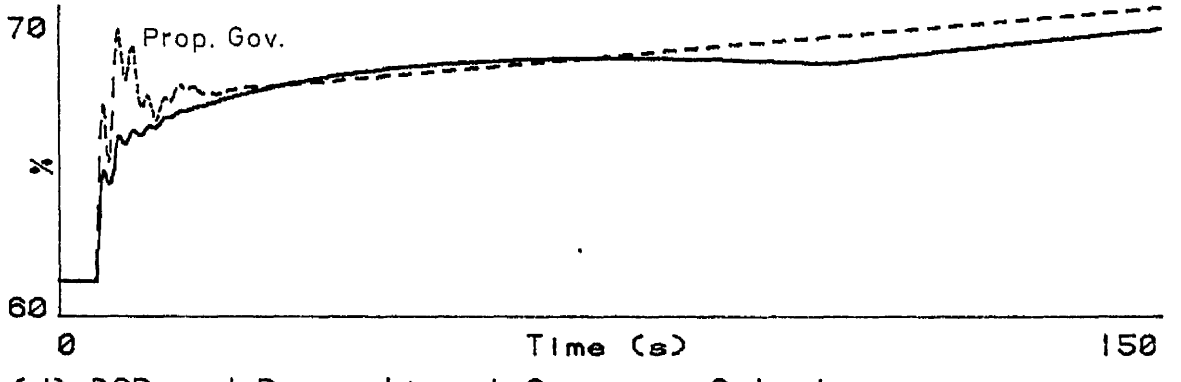
(a) Frequency Transient



(b) FDS Output and Governor Output



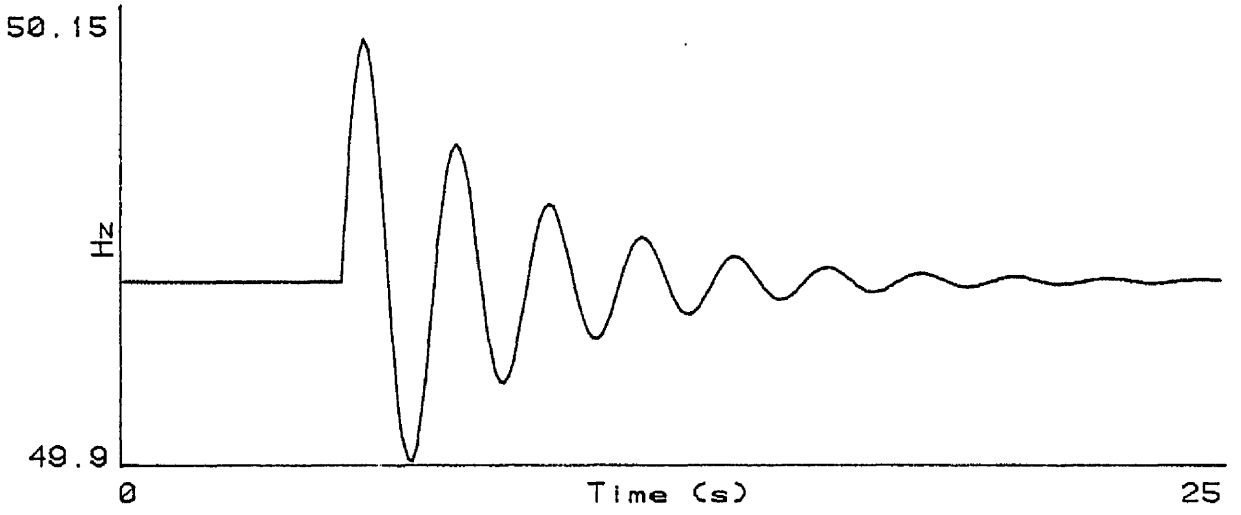
(c) Desired Servo Position (with and without FDS)



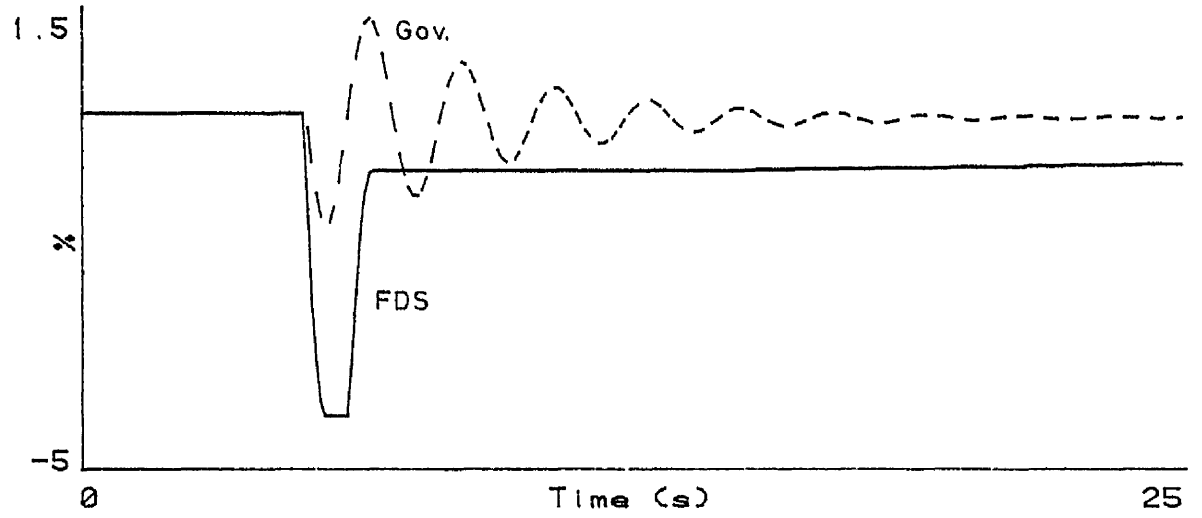
(d) DSP and Proportional Governor Output

Figure 8.10 - Generation Loss Transient

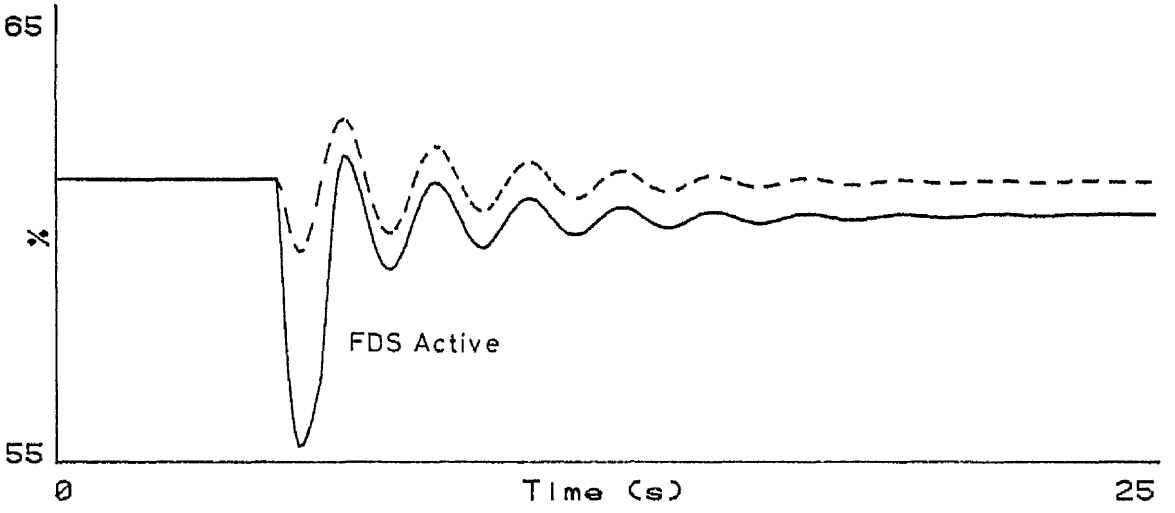
Original FDS Algorithm



(a) Frequency Transient



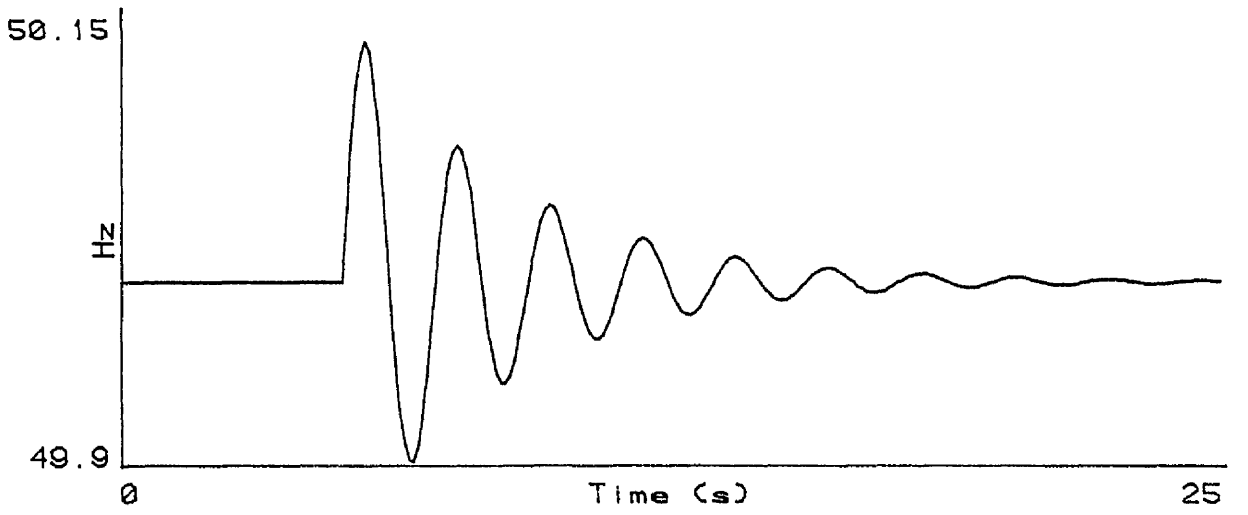
(b) FDS Output and Governor Output



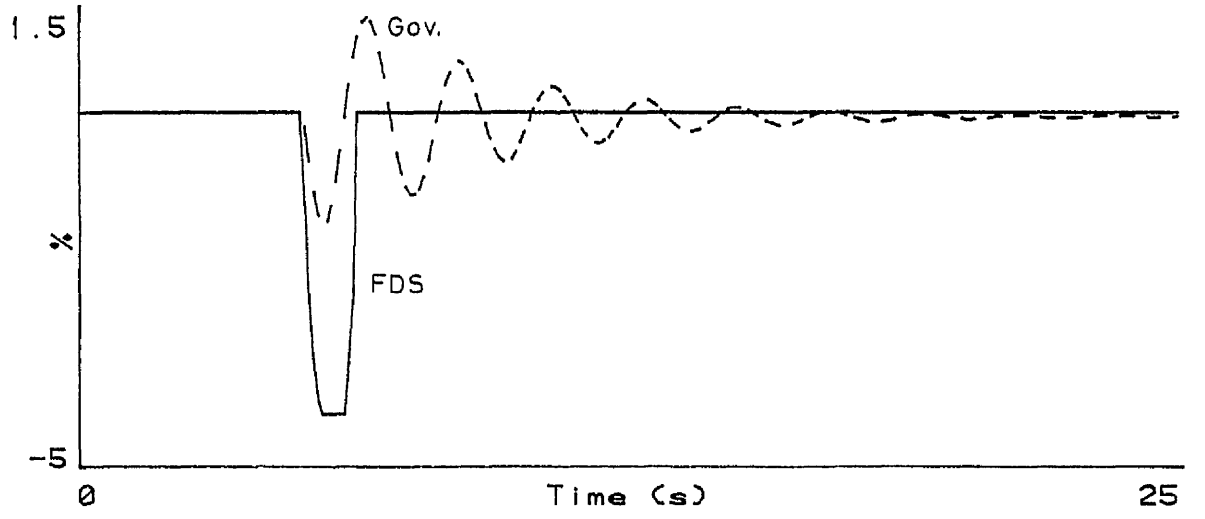
(c) Desired Servo Position (with and without FDS)

Figure 8.11 - Synchronising Transient

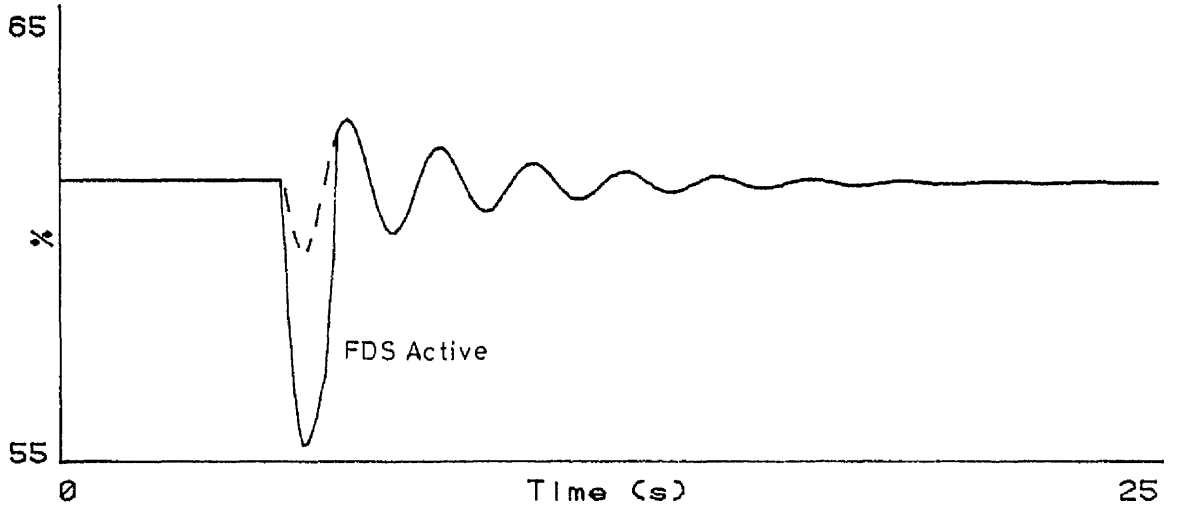
Original FDS Algorithm



(a) Frequency Transient



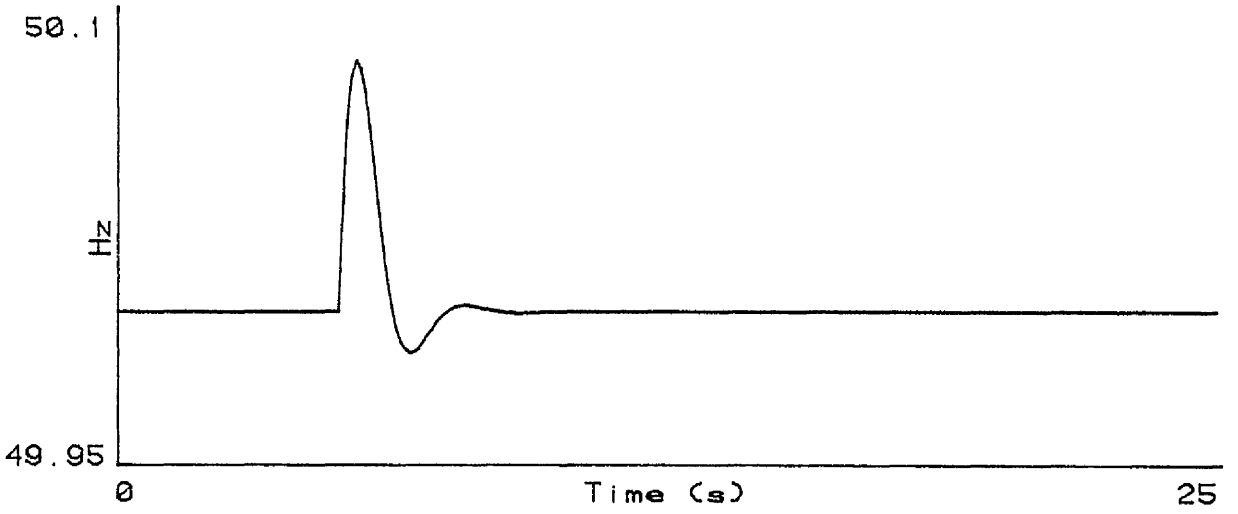
(b) FDS Output and Governor Output



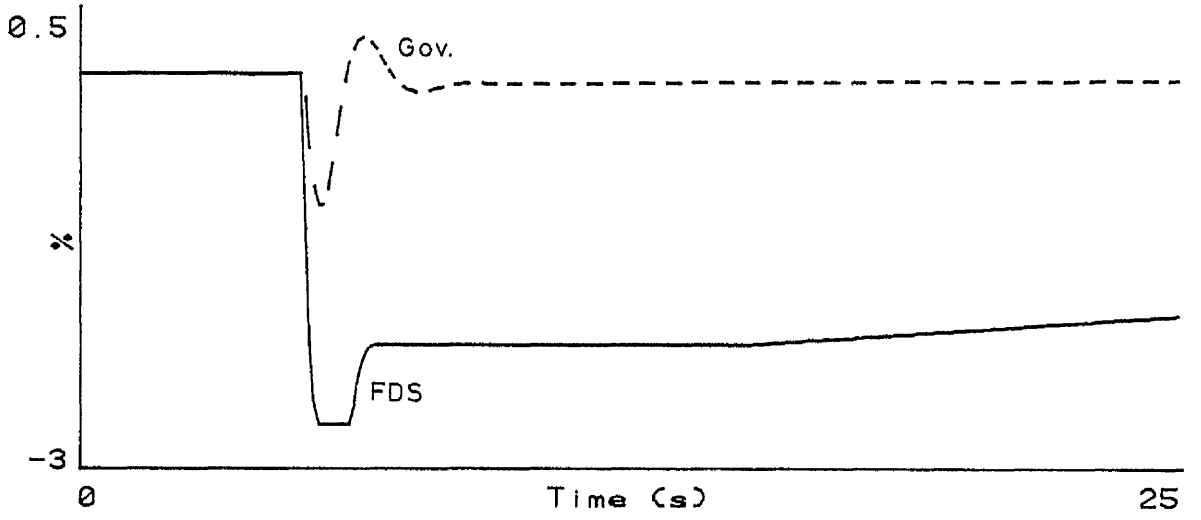
(c) Desired Servo Position (with and without FDS)

Figure 8.12 - Synchronising Transient

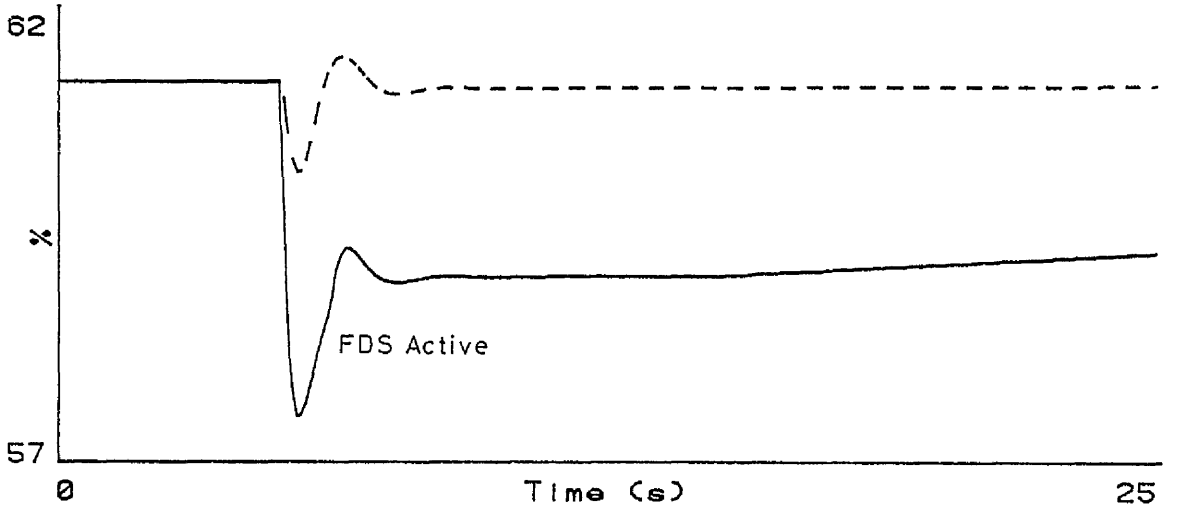
FDS with Escape Mechanism



(a) Frequency Transient



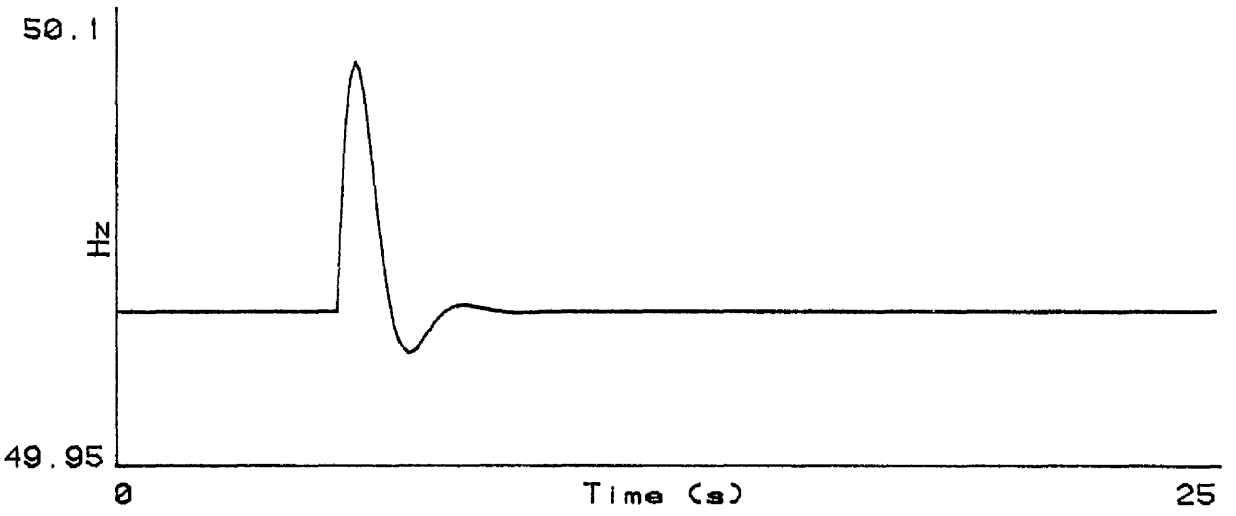
(b) FDS Output and Governor Output



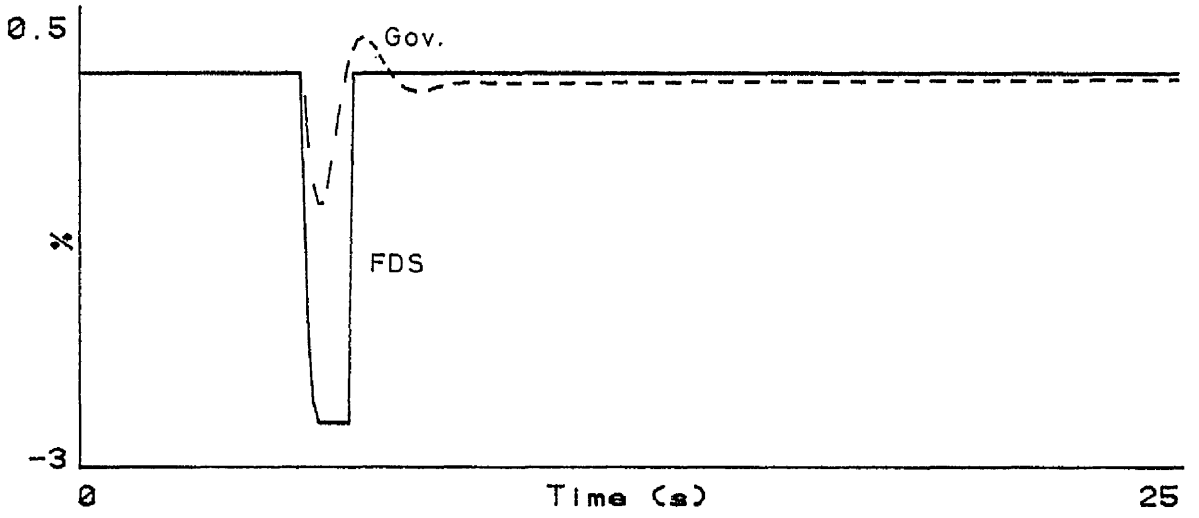
(c) Desired Servo Position (with and without FDS)

Figure 8.13 - Heavily Damped Transient

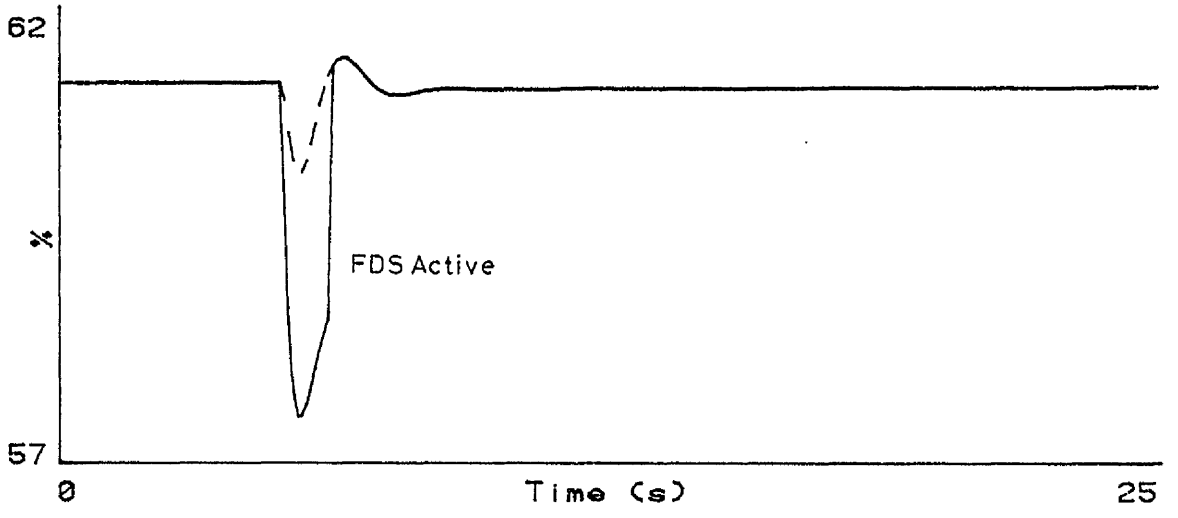
FDS with Escape Mechanism



(a) Frequency Transient



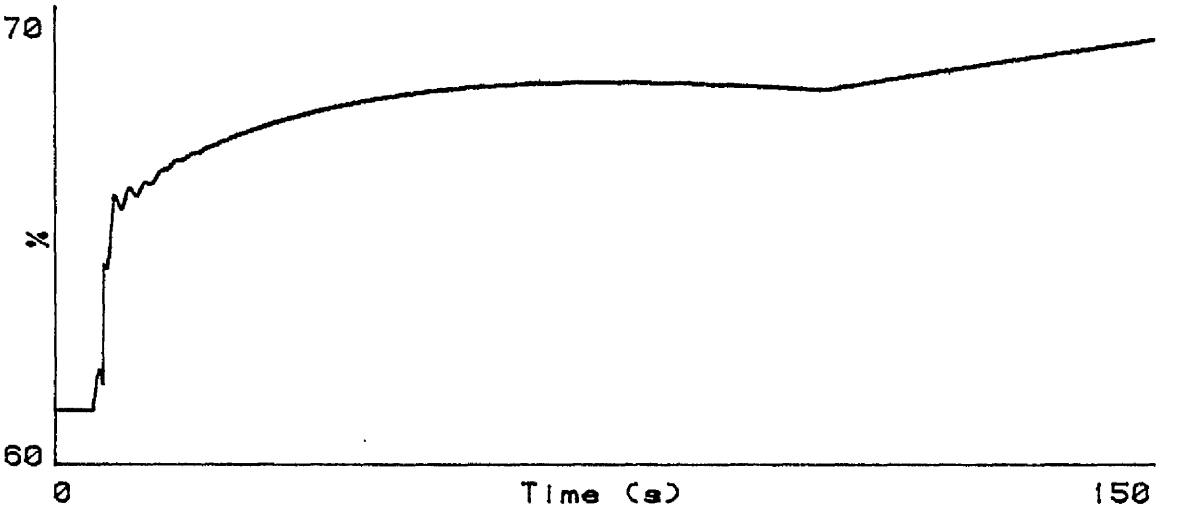
(b) FDS Output and Governor Output



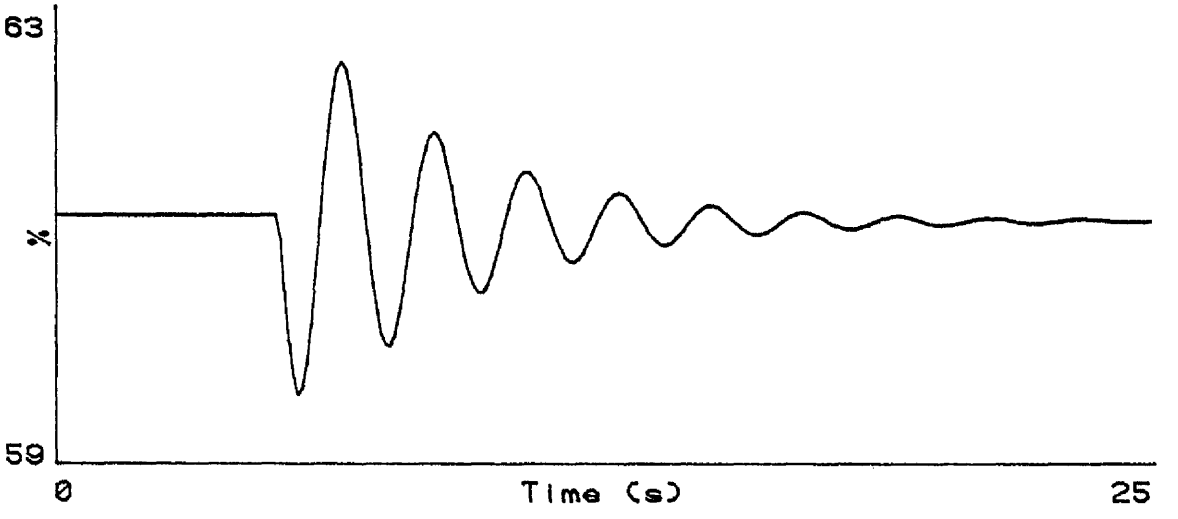
(c) Desired Servo Position (with and without FDS)

Figure 8.14 - Heavily Damped Transient

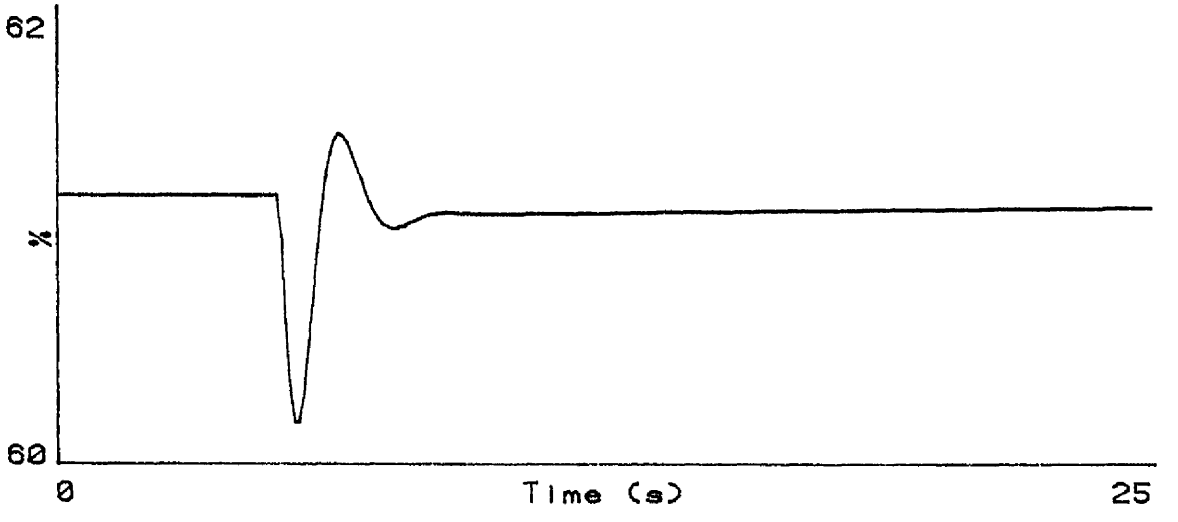
FDS with Improved Escape Mechanism



(a) DSP for Generation Loss Transient



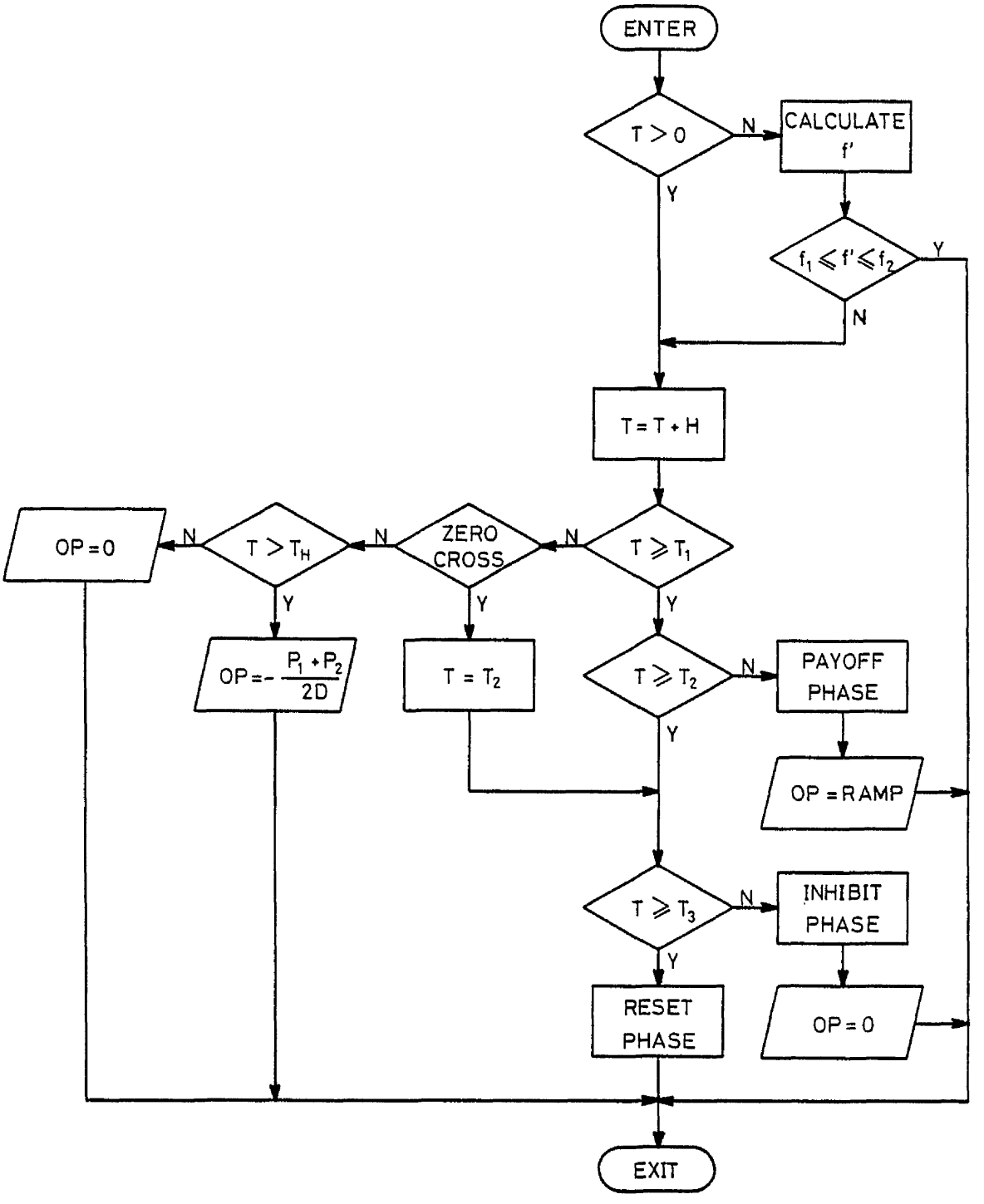
(b) DSP for Synchronising Transient



(c) DSP for Heavily Damped Transient

Figure 8.15 - Response of Final FDS

Various Frequency Transients



T - Timer
T_1 - Run Phase Time
T_2 - Payoff Phase Time
T_3 - Inhibit Phase Time
T_H - Hold Time
D - Droop

H - Integration Interval
f′ - Derivative of Frequency
f_1, f_2 - Limits on f′
P_1, P_2 - Peak Followers
OP - Output of FDS

Figure 8.16 - Final Implementation of Frequency Disturbance System

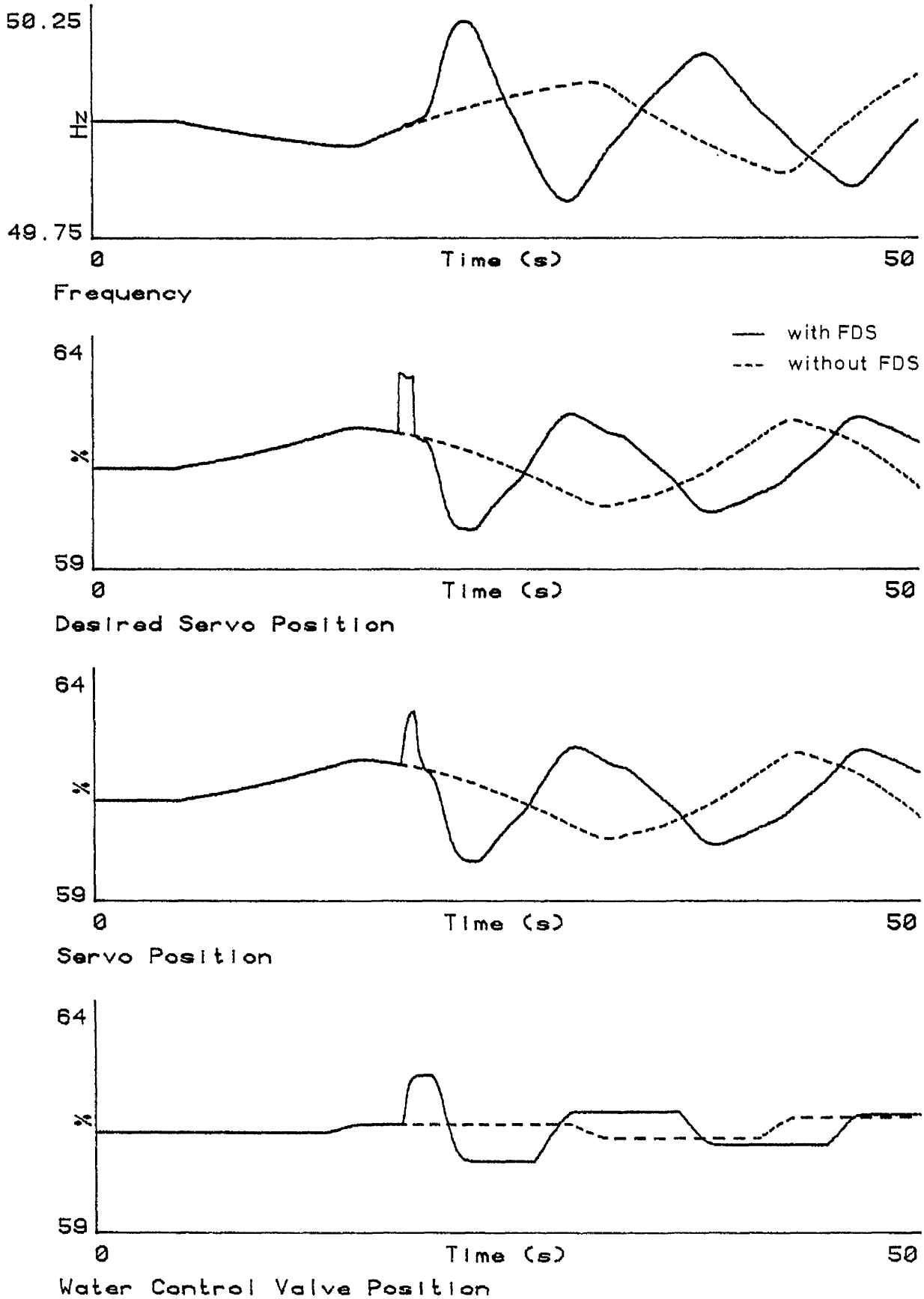


Figure 8.17 - Simulated Isolated Load

0.1% Step in Load with and without FDS

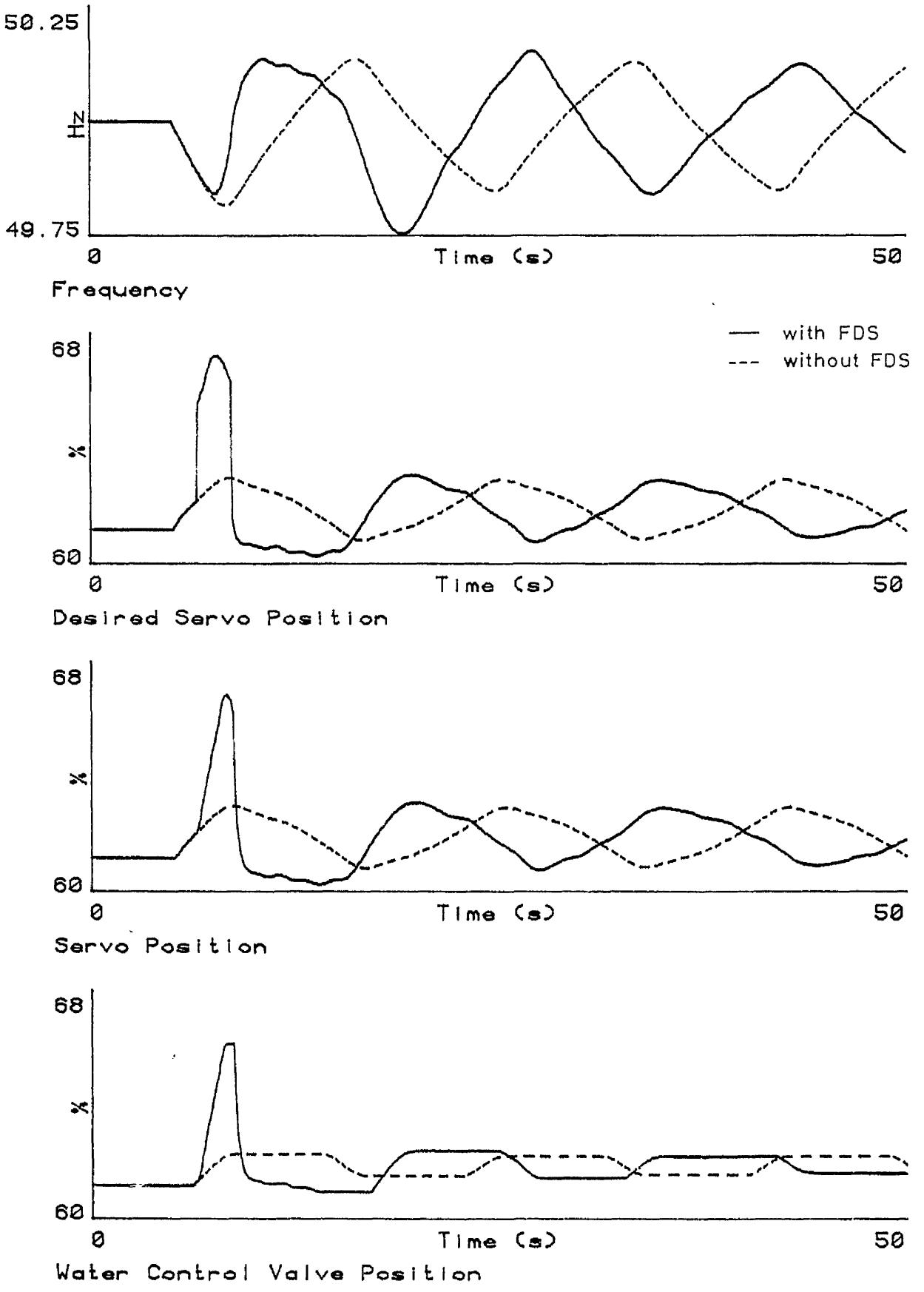


Figure 8.18 - Simulated Isolated Load
1.0% Step in Load with and without FDS

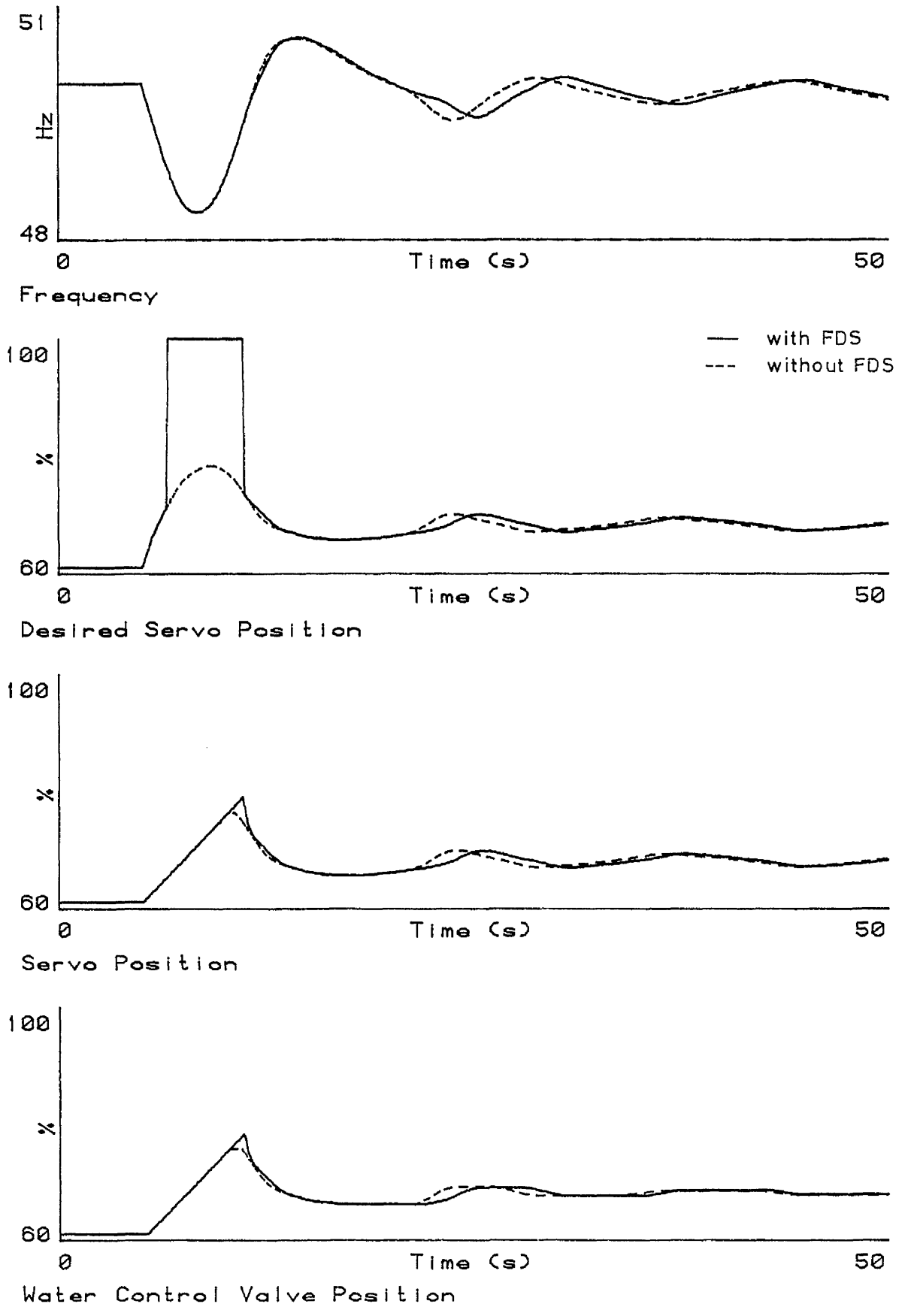
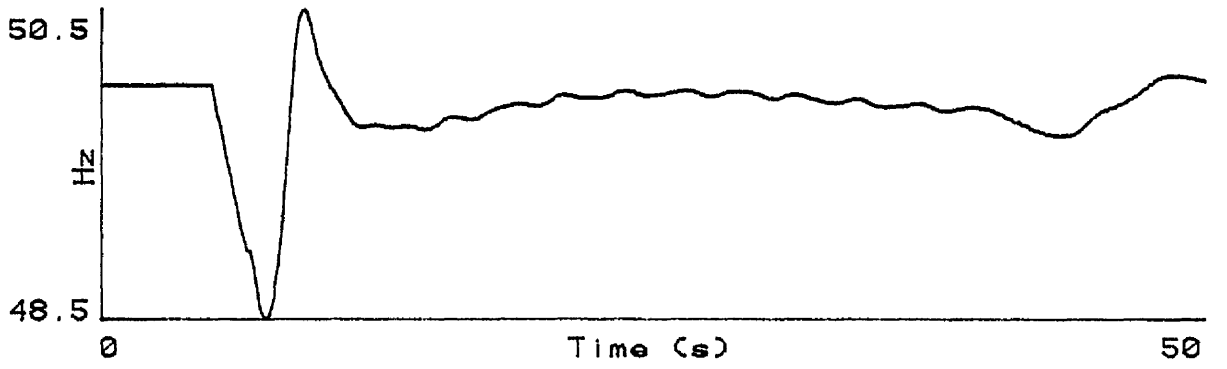
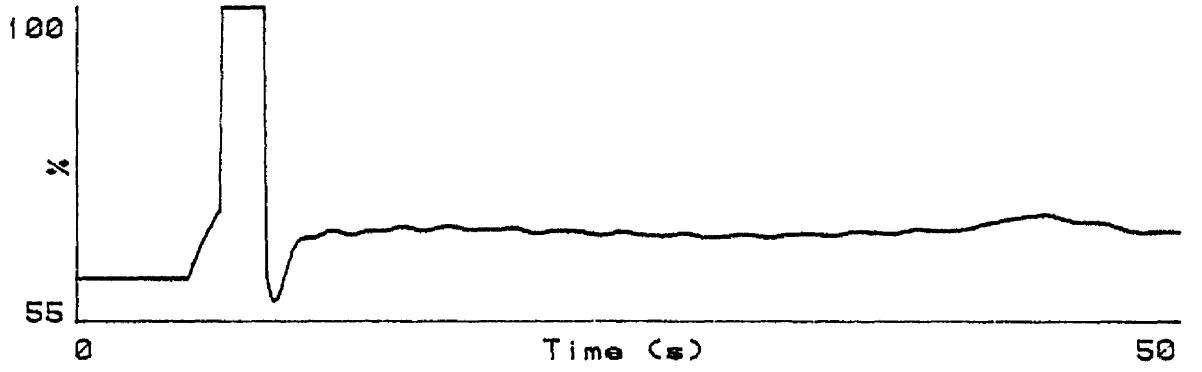


Figure 8.19 - Simulated Isolated Load

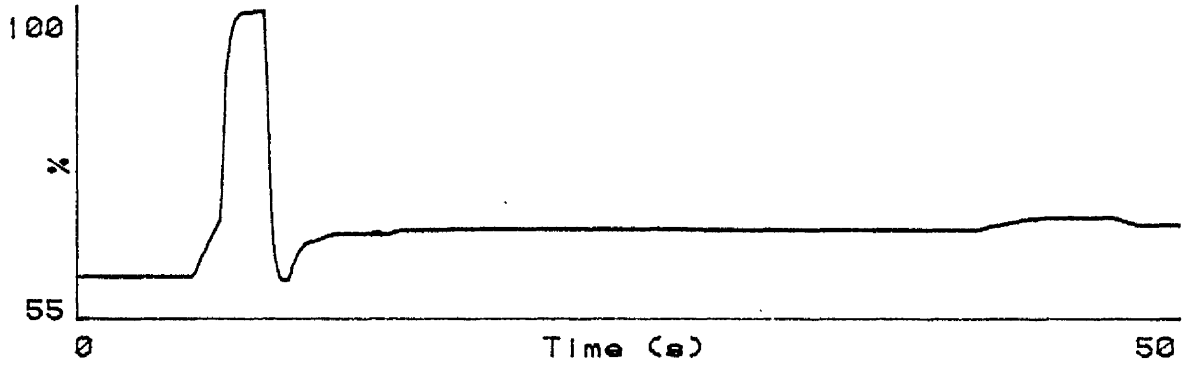
10% Step in Load with and without FDS



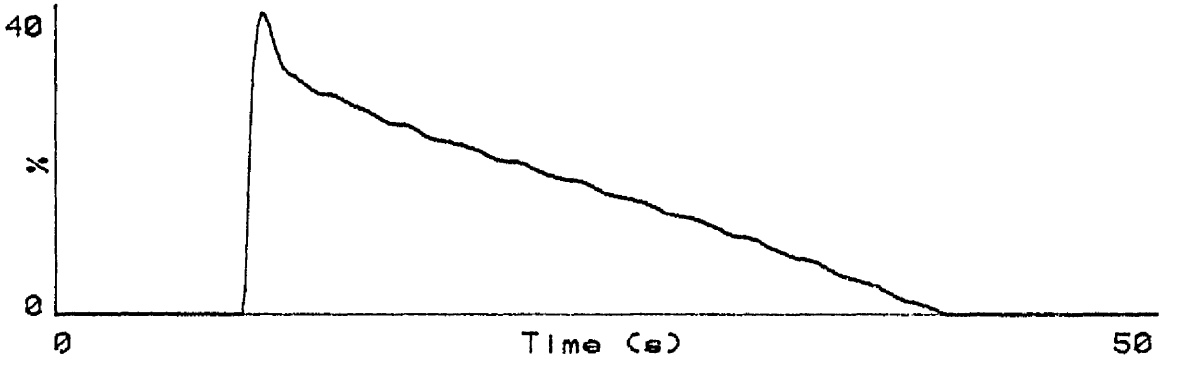
Frequency



Desired Servo Position



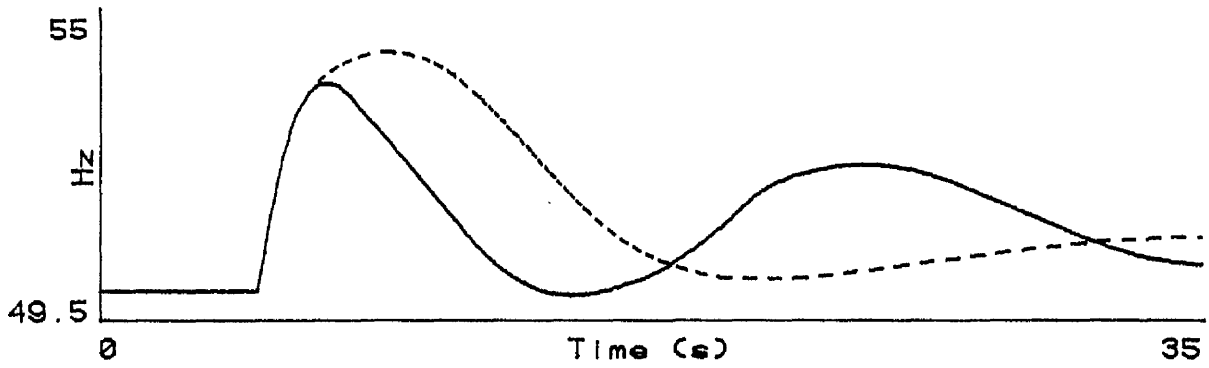
Water Control Valve Position



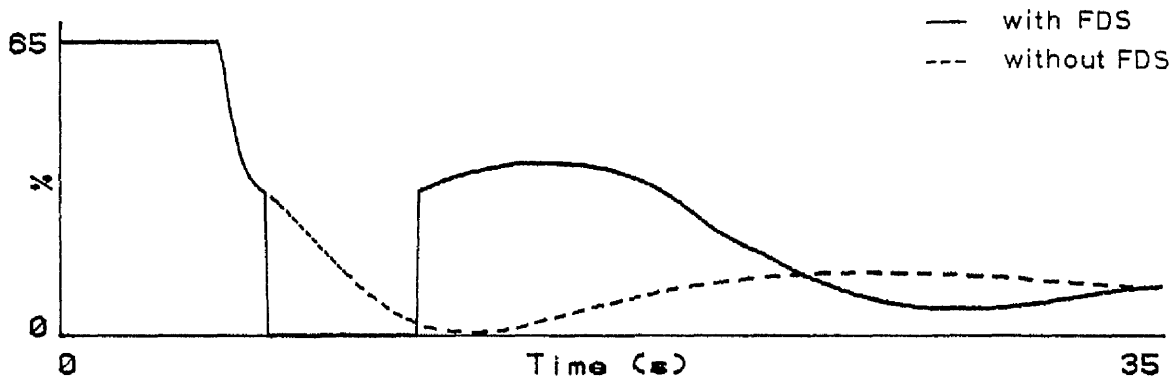
Relief Valve Position

Figure 8.20 - Simulated Isolated Load

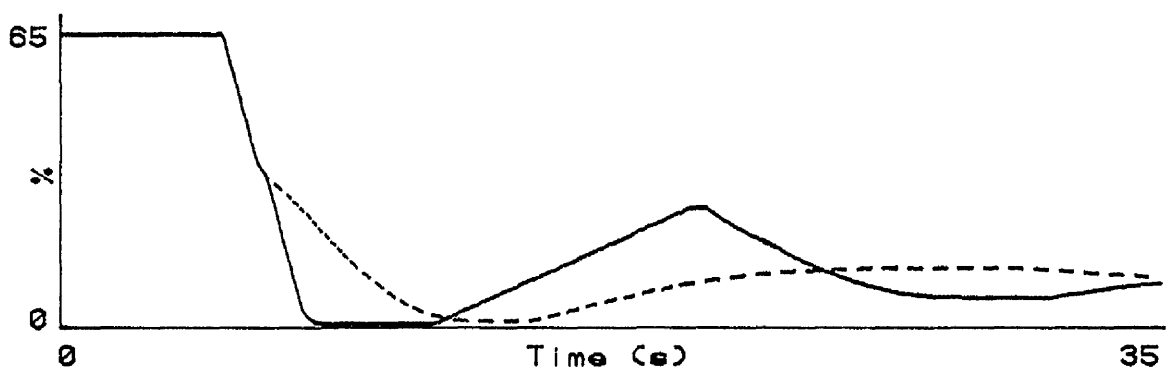
10% Step in Load, No Rate Limits



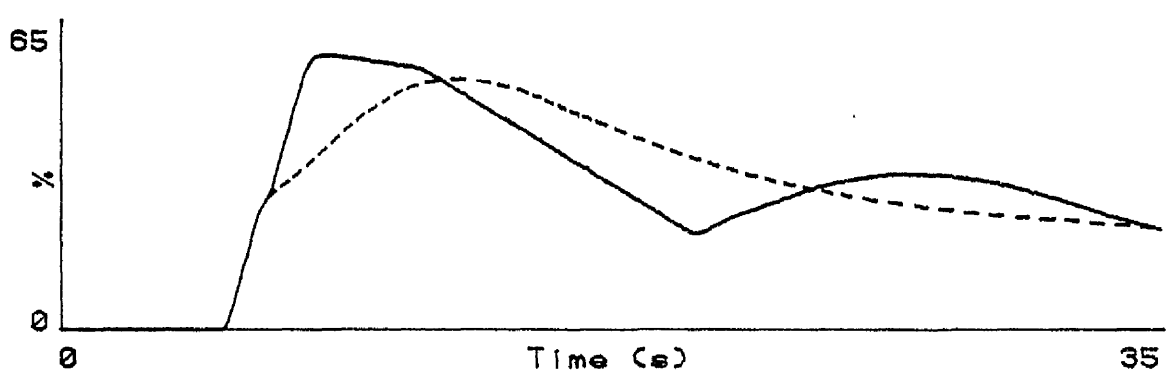
Frequency



Desired Servo Position



Water Control Valve Position



Relief Valve Position

Figure 8.21 - Simulated Load Rejection with and without FDS

CHAPTER 9

A THERMAL PLANT MODEL AND SIMULATION STUDIES

9.0 Introduction

Although, in relation to the 32.5MW hydro-generator at Loch Sloy, the National Grid can be considered to be of infinite capacity, this is an approximation which, as the size of individual hydro-generators increases, becomes less satisfactory. Of particular interest is the proposed pumped storage station on the East side of Loch Lomond in Scotland. This station could have an installed capacity of 2000MW and, if the Scottish System became isolated, as occured in January 1979, the behaviour of the hydro-generators could have a significant affect on the frequency of the power system. Thus, when using a simulation model to study the behaviour of large, governed hydro-generators in an interconnected power system, it is also necessary to have a model of the system to which it is connected.

For this reason, a power system model has been developed, using GUILDS, which can be used to study system frequency transients and the response of pumped storage or hydro plant to these transients. The application of GUILDS to developing a simulation model from a set of equations describing the plant dynamics is illustrated by this simulation study and the agreement between the results produced and those given elsewhere (see later) indicates the usefulness of the Simulation Language.

The plant which is interconnected to form the National Grid power system can be divided into four main catagories in the proportions given below:-

Thermal Plant	85.2%
Base Load (Nuclear) Plant	7.7%
Gas-turbine Plant	3.5%
Hydro and Pumped Storage Plant	3.6%

The thermal plant can be subdivided into coal fired and oil fired plant, each of which can be further subdivided depending on the type of the plant (for example, reheat or non-reheat). However, unless details of the behaviour of a specific type of plant are required, the thermal plant can be represented as a single unit with an 'average' response equivalent to the overall response of all the thermal plant in the system.

Base load plant, usually nuclear, is operated at a fixed (maximum) output and thus can be represented as a constant power output which will only vary if the frequency falls sufficiently to degrade the performance of the plant auxiliaries. Gas-turbine plant is normally used only for stand-by purposes, operated by under frequency relays, and does not effect the dynamics of the system, assuming that the frequency does not fall below a certain level.

Pumped storage plant can be operated in a similar way to gas-turbine plant or to a fixed pattern of pumping and generating. This is the case at present as, until now, pumped storage plant has been designed primarily for peak lopping, that is, meeting any generation deficit at times of maximum demand and absorbing any spare capacity of the base load plant when the demand is low. Under these circumstances the dynamics of the hydro-turbine governors can be neglected.

However, modern pumped storage schemes are now being built (e.g. Dinorwic[10,11], in North Wales) for use in frequency control and spinning spare duties. Thus, a detailed representation of the plant and governor dynamics is necessary for simulation studies with this type of plant in an interconnected power system.

Pumped storage plant is particularly suited to frequency control and spinning spare duties because of an inherent flexibility which arises from the speed with which energy can be supplied or absorbed. In addition, any increased output demanded from hydro plant can be sustained, unlike thermal plant where the output falls as the boiler pressure reserves are depleted.

9.1 Description of Model

As indicated by the figures given above, a large proportion of the load on the National Grid is met by thermal plant. For this reason, a model of a thermal power station, comprising a governor, boiler, turbine and generator, as shown in Figure 9.1, was developed. The model is based on that described in Reference 49 with some simplifications where attention to detail was considered to be of less importance. The component parts of the model are described in the following sections and the equations for each part are given in Figure 9.2 and also in the Model Description in Appendix 2.13. No attempt is made to derive the equations used but the basic relationships, from which the equations are derived, are given, as these are not to be found in the Reference. Where possible the equations are related to the physical processes which take place in the thermal power station.

9.1.1 Governor Model

A frequency error signal (f_e) derived from the comparison of system frequency (f) with a reference setting (f_s) is input to the governor which controls the speed and power output of the turbine through the steam flow into the turbine. A high pressure (or governor) valve controls the flow of steam from the boiler to the high pressure cylinder of the turbine and an interceptor valve controls the steam flow into the intermediate and low pressure cylinders. The governor can either act on both these valves or on the H.P. valve only with the interceptor valve remaining fully open except under overspeed conditions. Either of these modes of operation could have been included in the simulation model but the latter was implemented as this mode is most often found in practice.

The governor was represented as a first order lag with gain but the variation of gain with load, described in Reference 49, was omitted since, in this composite model the nature of this variation would be difficult to predict. The output of the governor is limited to represent the maximum and minimum demand from the governor. The valve actuator delay times were considered sufficiently short in comparision with the system dynamics to be neglected.

9.1.2 Boiler Model

The overall boiler model can be divided into three interconnected parts as described below.

Master Pressure (P.I.D.) Controller

In most conventional thermal power stations the firing of the boiler is controlled by an error signal (P_e) derived from comparing the actual boiler pressure (P_b) with a reference (P_s). The pressure error signal is input to a three term controller (proportional, integral and derivative) which outputs the demanded fuel and air flow signals.

This approach to boiler control means that changes in load are reflected in the boiler firing through frequency (which, through the governor, controls the steam flow) and boiler pressure. An alternative control strategy is to use a frequency error signal to control the boiler firing and then the steam flow is determined by the boiler pressure. This type of boiler control is being investigated for use in more modern thermal power stations.

The first of these two control schemes was implemented in the simulation. The output from the master pressure controller (M_x) is limited to represent the minimum and maximum firing conditions and then this signal (M) is used as input to the fuel-feed and air systems.

Fuel-air system

In the fuel-air system of the boiler in a thermal station, the fuel, oil or pulverised coal, is carried into the boiler by the primary air supply. Associated with this is a variety of plant including coal mills or oil pumps, fans and burners which have to be represented in the model. The implementation used is such that all types of fuel systems can be represented by changing the constants in the model. It is assumed that the speed of the forced draught fans supplying the primary air to the boiler is controlled by the master pressure signal and that any time constants in the air supply system are small compared to the fuel feed system. Thus the air flow, in per unit terms can be considered to be equivalent in magnitude to the master control signal.

In a coal fired station, raw coal is supplied by feeders to the fuel pulverisers or mills which transform the raw coal into a fine powder. On emerging from the mills the powdered fuel is pushed across an orifice up which the primary air passes, entraining the fuel. Under steady state conditions the feeders are supplying coal to the mills at constant rate (M_i). Any change (fall) in boiler pressure, and hence

the control signal, causes a change (increase) in the fuel demand (E_{m1}) from the steady state level. The change in demand is met by altering the speed of the feeders but there is a time delay (T_d) before the change in speed manifests itself at the input to the mills. Although the coal feed has been increased there is no increase in the amount of milled coal until additional mills have been started and run up to speed, introducing a further delay (T_s) into the fuel feed system.

The mills are represented by a first order lag (time constant T_c) with the delayed fuel feed boost (E_m) as input. The output from the mills is the fuel input (F_i) to the primary air supply and the resulting density of fuel in the air (per unit volume) is given by the integral of the fuel input less the fuel consumed (effectively, the heat input to the boiler). This integral has a time constant (T_m) which represents the amount of pulverised coal actually stored in the mill and the output is limited by restraints on the amount of coal that can be supported by the air. The heat input to the boiler (Q_i) is given by the product of the fuel density (F_d) and the air flow (M, the master control signal).

The fuel feed system for oil fired plant can be represented by the same functional blocks as used for coal plant. In oil fired plant the firing of the boiler is increased by igniting more burners and starting additional oil pumps. Thus, the delays in the fuel feed system for oil fired plant are generally shorter than for coal fired plant where the corresponding delays are due to starting additional mills and feeders.

Boiler

A simple representation of the boiler drum and heating tubes as a single energy storage unit is used in the model and the steam temperature and feed water level control loops have been neglected. Thus, the boiler pressure (P_b) is given by the integral of the heat

input to the boiler (Q_i) less the steam flow out to the turbine (W_1). The time constant (T_b) of this integral represents the size of the boiler in terms of stored energy.

A more detailed representation of a boiler is derived in Reference 50 by Astrom but as preliminary studies with the model given above proved satisfactory it was decided that is was unnecessary to use a more detailed model.

9.1.3 Turbine Model

The turbines used in most conventional thermal plant comprise a high pressure cylinder, an intermediate pressure cylinder and one or more low pressure cylinders. The h.p. cylinder is supplied directly from the boiler and the exhaust steam from this cylinder passes to the i.p. and then the l.p. cylinders. Between the h.p. and i.p. cylinders the steam is usually returned to the boiler to be reheated.

The implementation used in the simulation model has an h.p. cylinder and a composite section representing the i.p. and l.p. cylinders. The h.p. cylinder is modelled as a back pressure turbine, where the outlet pressure is high enough to affect the steam flow through the turbine. The flow (W_1) is given, in terms of the inlet pressure (P_1) and the outlet pressure (P_2):-

$$W_1 = k * \sqrt{(P_1^2 - P_2^2)}$$

where k is a constant determined from the steady state conditions. The i.p./l.p. cylinder, on the other hand is modelled as a condensing turbine, where the outlet pressure is sufficiently low not to affect the steam flow. Thus the steam flow for this cylinder is given by:-

$$W_2 = k * P_2$$

where k is again determined from the steady state conditions.

The simulation has a reheater between the h.p. and i.p/l.p. cylinder which is modelled as a storage element and, by considering the mass balance, the rate of change of pressure (dP_2/dt) within the element is given in terms of the inlet (W_1) and outlet (W_2) flow rates as:-

$$\frac{dP_2}{dt} = \frac{RT}{v} (W_1 - W_2)$$

where R is the universal gas constant, T is the temperature of the steam and v is the volume of the storage element. Thus, the reheater pressure is given by:-

$$P_2 = \frac{W_1 - W_2}{sT_r}$$

where T_r, effectively the reheat time constant, represents the storage of the reheater. The value of T_r can be used to specify the reheat conditions for the plant, i.e. reheat or non-reheat (e.g. 10. or 0.1).

The power output from the turbine (P_t) is the sum of the power from the h.p. and i.p./l.p. cylinders ($P_{m1} + P_{m2}$). The power output of any turbine is given by the product of the steam flow (w), the heat drop (Δh) and the efficiency (n); that is

$$P_m = w * \Delta h * n$$

As an approximation, the efficiency is assumed to be constant, which is true if the turbine speed remains constant. It can be shown that the heat drop across a turbine can be related to the inlet (P_1) and outlet (P_2) pressures as:-

$$h = \left(\frac{\gamma}{\gamma-1} RT\right) \left[1 - \left(\frac{P_1}{P_2}\right)^{\frac{\gamma-1}{\gamma}}\right]$$

where γ is the the adiabatic index of the steam.

The steam in the h.p. turbine is superheated and thus may be considered a perfect gas with $\gamma = 1.3$. However, in the i.p./l.p. cylinder the steam is wet and the perfect gas laws do not apply. It was considered that an adequate approximation could be obtained by using a value of 1.13 for γ in this latter case.

The steam valves are modelled as a first order lag and since, in practice, steam valves are designed to close very quickly but have a much slower opening rate, one of two time constants is selected for use in the expression for calculating the position of the governor valve (A_2), depending on the direction of movement. The derivative of the valve actuator position is used to determine whether the valve is opening or closing and the appropriate time constant is selected using a switching function.

The impedance of the steam piping has been neglected as it is small when compared to the impedance of the governor valve (except when the valve is fully open) and the loop pipe lag (included in the model in Reference 49) has been omitted as the effect was considered to be secondary.

9.1.4 Generator Model and Load Representation

The generator and load are modelled as a lumped inertia and the generator is assumed to be lossless. The dynamic equations for the generator have not been included as the effect on the overall response of the plant was considered to be minimal and, in any case, the problem size would have been too large for the Simulation Language had these equations been included. The action of the Automatic Voltage Regulator (A.V.R.) was also omitted from the model. The variation of the load with frequency (the load self-regulation) has been included in the model but other load variations (referred to in Reference 49) have been neglected. Figure 9.3 shows a block diagram of the generator and load representation.

9.2 Implementation

Initially, simulation studies using the thermal plant model were performed using CSMP[19,20] (Continuous Systems Modelling Program) on an IBM 360 or 370 computer. The model used for these studies was a simplified version of that described above. Subsequently, the simulation studies were transferred to a PDP11/45 computer and the more complex model was developed under GUILDS (Glasgow University Interactive Language for Dynamic Simulation). The results presented here are all from the latter implementation although some of the work was done using CSMP. The constants used in the model were provided by the N.S.H.E.B. as typical parameters for the type of plant being simulated. In addition the values for some of the constants were obtained from Thompson[51] where comparisions between Thompson's model and the present model were possible. Documentation Files containing the values used can, for example, be found in Appendix 3.54 and 3.55.

9.3 Model Tests and Validation

In order that the thermal plant model developed could be used for simulation studies, particularly in relation to interconnected hydro-thermal systems, it was necessary to ensure that the response of the model as implemented was similar to the results of the plant and simulation results presented elsewhere[49,51]. In all cases the model parameters were chosen to facilitate comparison of the responses with these results.

The response of the model to step changes in electrical load was investigated first. As the load increases, the system frequency starts to fall (at a rate determined by the system inertia) and this, through the governor, causes the governor valve to operate. Increased power output is obtained at the expense of the energy stored in the boiler and this enables the frequency to recover temporarily.

The delay in the fuel feed system permits the boiler pressure to collapse and this results in a drop in power output and hence frequency, which can last for up to several minutes and is only arrested when the firing responds to the fall in pressure.

The Model Description for these closed loop step tests is given in Appendix 2.13 and the values of the model parameters are to be found in the Documentation Files in Appendix 3.48 and 3.49, for Figures 9.4 and 9.5 respectively.

Figure 9.4 shows the response of the boiler and the effect on system frequency of a 50% step change in electrical load. The effect of the governor valve limiting fully open can also be seen from this figure. In Figure 9.5 the step was reduced in size to 5% to avoid this limit and other limits in the fuel feed system.

The overall response of the model was found to be in agreement with the published results [49,51] but, as can be seen from Figure 9.5, an addition oscillation was present in the frequency signal. The oscillation was only evident in the 5% step case as the step was not large enough to cause the limits in the boiler control system to be reached. Further investigation indicated that a possible source of this oscillation was the boiler master pressure control loop and thus it was decided to carry out some open loop studies on the boiler and turbine.

Firstly, the boiler and associated control loops were simulated separately as in the Model Description in Appendix 2.14. A step was applied to the governor valve, opening the valve fully from approximately the half load position and the boiler pressure was observed. To ensure that the effect of limiting did not obscure the response all the upper limits in the boiler contol and firing were set to 10 pu. Figure 9.6 shows the results of this test for various settings of the controller constants (K_p, K_i, K_d).

The original settings of K_p = 20., K_i = 0.04, and K_d = 0. give rise to an oscillatory response (as observed in the closed loop system). The values of K_p = 5., K_i = 0.015 and K_d = 0. were chosen as the new values for the controller constants as giving a more satisfactory open loop response. No attempt was made to optimise the controller but it should be noted that these values, arrived at by independent means, are similar to those used by Thompson[51].

As a further check on the boiler and turbine models, this test was repeated on the complete model (open loop) with the new controller constants. The response of the boiler, the fuel feed system and the turbine is shown in Figures 9.7 and 9.8. for a 50% and a 30% step change in governor valve position respectively. (The Model Description in Appendix 2.14 and the Documentation Files in Appendix 3.54 and 3.55 are relevant.) In both cases, as the governor valve opens, the power output increases due to the increased steam flow, but begins to fall off almost immediately as the boiler pressure falls. The drop in boiler pressure causes an increase in firing which finally returns the pressure and the power output to the desired levels.

The results shown in Figure 9.7 for this test are in close agreement with those of Thompson[51] for a relatively small thermal station with a fast (oil) fuel feed system. Similarly, the results in Figure 9.8 compare well with those given in Reference 49 for various different thermal stations in the C.E.G.B. system. These stations are predominantly large coal fired plants and thus the steam reserves in both the boiler and the reheater are much greater although the time constants in the fuel feed system are longer.

To test the open loop response of the turbine a step was applied to the governor valve such that the valve was closed from fully open to about the half load position. As the governor valve closes,

the power output from the h.p. cylinder drops rapidly although the power from the i.p./l.p. cylinder falls more slowly due to the storage of the reheater. (If the interceptor valve had been closed with the governor valve the power output would have fallen more rapidly.) Figure 9.9 shows the response of the turbine h.p. and i.p./l.p. cylinder pressures to this step and again good agreement with Reference 51 has been obtained.

Finally, the original step tests were repeated on the closed loop model with the new controller constants. The step sizes and model parameters were chosen, as noted previously, so facilitate the comparison of the responses with other results. The Model Description used for the tests was the same as was used previously (Appendix 2.13) and the Documentation Files are in Appendix 3.57 and 3.58.

Figure 9.10 shows the response of the model to a 20% increase in load from 80% to full load which is in close agreement with the results for the oil fired plant given in Reference 51. Figure 9.11 shows a similar step test, this time of 10% which gives a response similar to the results of the system splitting tests shown in Reference 49. In neither case is there any evidence of there being instability in the model response.

As a result of these tests it was decided that the simulation model was reliable and could therefore be used for further simulation studies.

9.4 Simulation Studies

The following sections describe various simulation studies which were carried out and involved the thermal plant model detailed above.

9.4.1 Pumped Storage

The N.S.H.E.B. requested some information on the effect that starting a pump-turbine in a pumped storage station would have on system frequency. In particular, the Board were interested in the power station proposed for Craig Royston on the side of Loch Lomond for which which 400MW pumps had been suggested. The largest pump-turbines in service at present are 150MW (although 300MW sets are being installed at Dinorwic[10,11]) and the effect of starting these larger sets on a lightly loaded system (such as the Scottish system on a Summer night) was to be investigated.

In order to study this problem, it was necessary to develop the simulation model. A power system model was created by adding some base load plant (non-regulating, fixed output) to the thermal plant model and allowing a proportion of the thermal plant output to be non-regulated. Thus, the one thermal plant model could, by using average characteristics, be used to represent all the thermal plant on the system some of which could be operated at a fixed output (usually full load). The pumped storage plant was represented by a step change in the load or in the generating plant. The block diagram of this power system model is shown in Figure 9.12, the Model Description is given in Appendix 2.15 and the constants used are to be found in Appendix 3.59.

9.4.2 Pump Starting with 90% Capacity

For the first study, using projected AMD (Average Maximum Demand) figures for 1988, the National Grid was assumed to be supplying a load of 25GW from base load plant of 5GW and 22.2GW of thermal plant operating at 90% of full load. A 400MW step increase in load was applied after 4 seconds (to allow time for the steady state conditions to be observed). The effect on system frequency (f) and the fuel input (F_i), boiler pressure (P_b) and output power (P_{t1}) of the regulating plant was recorded.

Several simulation runs were carried out with different proportions of regulating thermal plant, that is, the plant, which, through governor action, can respond to changes in system frequency. The "mill start delay time" was also varied as this represents the time for remedial action to be taken and depends on whether the regulating plant is on automatic or manual control.

These particular parameters were of interest because, in practice, some thermal plant is operated with the governor valve fully open, effectivley non-regulating base load plant, and this lowers the overall regulation of the system. Also, there is a tendency for automatic governoring to over-ridden by manual action and in particular during large frequency disturbances the increased steam flow demanded by the governor is counteracted by the operator in an attempt to maintain boiler pressure[49].

The results of these simulation runs are shown in Figures 9.13 to 9.16 and the model parameters are detailed below and in Appendix 3.59. The model used is given in Appendix 2.15.

Figure No.	Regulating Thermal Plant	Mill Start Delay Time
9.13	100%	100s
9.14	50%	100s
9.15	1%	100s
9.16	100%	300s

As the proportion of regulating thermal plant decreases, the magnitude of the frequency transient increases and the final value to which the frequency recovers falls. Figure 9.15 shows no frequency recovery as the spare capacity of the regulating plant is insufficient to meet the increases demand. Figure 9.16 represents the situation where remedial action is delayed because manual control of the firing is being used, although the governor is active. The frequency

transient is not significantly worse than that of Figure 9.13 due to the boiler stored energy but the deviation in boiler pressure is much greater. Had the increase in firing been further delayed, the frequency would have fallen below the level of the initial transient as the boiler pressure collapsed.

In all the cases shown (apart from Figure 9.15 where there is no recovery) the magnitude of the frequency transient is less than 0.064Hz which is well within the range of frequeny transients experienced at present. Figure 9.17 shows an histogram of the frequency transients for generation loss incidents in 1976 compiled from information supplied by the N.S.H.E.B. from which it can be seen that the magnitude of most transients lies within the range 0.01 to 0.1 Hz.

9.4.3 Pump Starting with 95% Capacity

The above series of simulation runs was repeated with the amount of thermal plant reduced to 21.05GW operating at 95% of full load. Figure 9.18 shows the respose of the system frequency for model parameters as listed above for the previous set of tests. The results of these tests are similar to those of the first series of tests as the spare capacity is still sufficient to meet the increase in load.

However, if the proportion of regulating plant is further reduced to 35%, with a mill start delay time of 300s, the effect of the governor valve limiting at fully open can be seen (Figure 9.19). The rate of fall of frequency, after the initial transient, increases when the governor valve limits as there can be no further increase in steam flow. This fall in frequency continues until the firing is increased and the boiler pressure restored.

The response of this model is similar to the Stylised Frequency Transient caused by a generation deficit, as shown in Figure 9.20. This transient is constructed from the analysis of generation loss incidents on the National Grid and was supplied by the N.S.H.E.B. The expressions used for constructing the transient are:-

$$r_t = \frac{\Delta f_m}{\Delta f} = 1.6$$

$$r_1 = 1 + \frac{T}{T_f} = 3.2$$

where Δf is the average 'pseudo steady state' drop in frequency within the first 30 seconds of the transient (see Figure 9.17);

Δf_m is the average maximun frequency drop within the first 30 seconds of the transient;

T is the average elapsed time before corrective action is taken (300 seconds);

T_f is $\dfrac{\Delta f}{\text{ave freq trend before incident - ave freq trend after}}$

For the transient shown in Figure 9.19 f is 0.09Hz, the initial transient is 1.7 * Δf and the lowest value of the frequency is 3.1 * Δf. These values are close to the typical values given above.

9.4.4 Two Stage Pump Starting

A further study of the effect of pump starting was carried out with a more detailed representation of the system load and a more accurate model of the pump starting. The load, instead of being constant was assumed to be falling at a steady rate (typical of the night-time load) and when it had fallen by 400MW the pump was started. The pump starting was modelled as an initial step of 80MW followed by a further 320MW step after 30 seconds as a better representation of the way in which the pump would be run up in practice.

From figures provided by the N.S.H.E.B. for the average daily load (for July 1977) as a percentage of average maximum demand (Figure 9.21) a ramp rate of $4.38*10^{-3}$ % per second, for the period from 2300 to 0030, was derived. Assuming, as before, an AMD figure of 25GW this gives a ramp rate of 1.1MW per second.

Figure 9.22 shows the results of this study for 50% regulating plant with a mill start delay time of 100 seconds. It can be seen that the frequency transient is smaller than for the previous tests with a steady state load and that the recovery is quicker (cf. Figure 9.16). This is a result of the boiler pressure being slightly high before the pump is started and the load continuing to fall afterwards.

9.4.5 Conclusions

These studies are typical of the those required for assessing the impact of large pumped storage schemes, for example that proposed for Craig Royston, on the remainder of the power system. The use of the thermal plant model has been demonstrated, as has the value of GUILDS in facilitating such studies.

This work is now being extended by A.G. Marshall by including a simplified version of the hydro generator model developed in Chapters 5 and 6 in the power system simulation. The resulting model is being used to study the operational performance of a combined hydro- thermal power system, particularly in relation to the boiler controls in the thermal plant. The possibility of using control schemes for the boiler, other than that presently implemented in the model is being investigated.

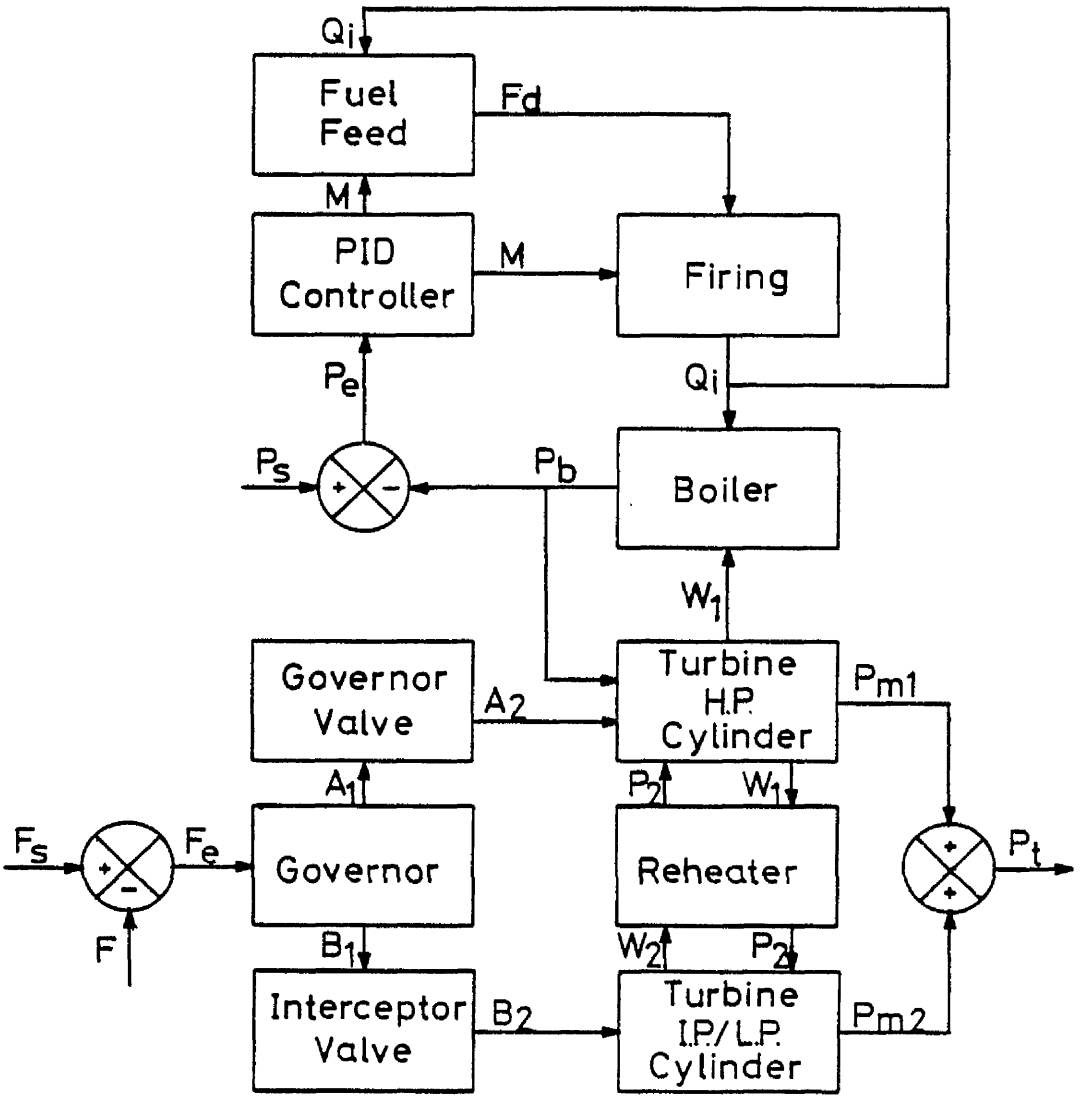
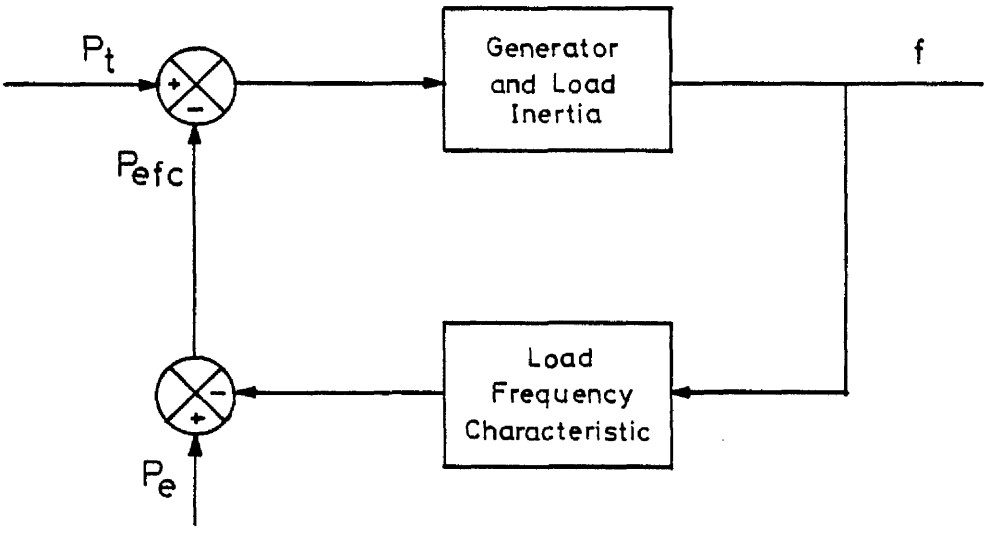


Figure 9.1 - Thermal Plant Model

```
* GOVERNOR
        FE=FS-F
        Z1=(G*FE-AX)/TG
        AX=INTGRL(A1I,Z1)
        A1=LIMIT(A1L,A1H,AX)              Governor Valve Actuator
        FEOS=FSOS-F
        BX=B1I+(100.0*FEOS/AIVD)
        B1=LIMIT(B1L,B1H,BX)              Interceptor Valve Actuator
*
* MASTER PRESSURE CONTROLLER
        PE=PS-PB
        Y1=AK1*PE
        Y2=INTGRL(Y2I,Y1)
        MX=Y2+AK2*PE+AK3*PE
        AM=LIMIT(ML,MH,MX)               PID Controller
*
* FUEL FEED SYSTEM
        FD=LIMIT(FDL,FDH,FD1)
        QI=AM*FD                         Heat Input
        EM1=AM-MI
        EM2=DELAY(TD,EM1)
        EM3=MI+EM2
        EML=SWIN(TIME-TS,MI,EMH)
        EM=LIMIT(0.001,EML,EM3)
        Y6=(EM-FI)/TC
        FI=INTGRL(FII,Y6)                Fuel Input
        Y7=(FI-QI)/TM
        FD1=INTGRL(1.,Y7)                Fuel Density
*
* STEAM VALVES
        ZZ=(A1-A2)
        TGV=SWIN(ZZ,TGVC,TGVO)
        Z2=ZZ/TGV
        A2=INTGRL(A2I,Z2)                Governor Valve Position
        Z3=(B1-B2)/TIV
        B2=INTGRL(1.,Z3)                 Interceptor Valve Position
*
* TURBINE
        P1=A2*PB                         HP Cylinder Pressure
        W1=(SQRT(ABS((1-(P2*R/P1)**2)/(1-R**2))))*P1        Steam
        W2=B2*P2                                            Flow
        Z4=(W1-W2)/TR
        P2=INTGRL(P2I,Z4)                Reheater Pressure
        PM1=AK*W1*((1-(P2*R/P1)**0.231)/(1-R**0.231))       Power
        PM2=(1.-AK)*W2                                      Output
        PT=PM1+PM2
*
* BOILER
        Y8=(QI-W1)/TB
        PB=INTGRL(1.,Y8)                 Boiler Pressure
*
* GENERATOR AND LOAD
        PE=TPIC+DPE*STEP(4.)
        PEFC=PE*(1.-(FIC-F)*(SSLFC/2.))
        X1=(PT-PEFC)/(2.*AH*F)
        F=INTGRL(1.,X1)                  Frequency
```

Figure 9.2 - Equations for Thermal Plant Model



P_t - Generated Power

P_e - Load Power

P_{efc} - Load Power Frequency Corrected

f - Frequency

Figure 9.3 - Generator and Load Representation

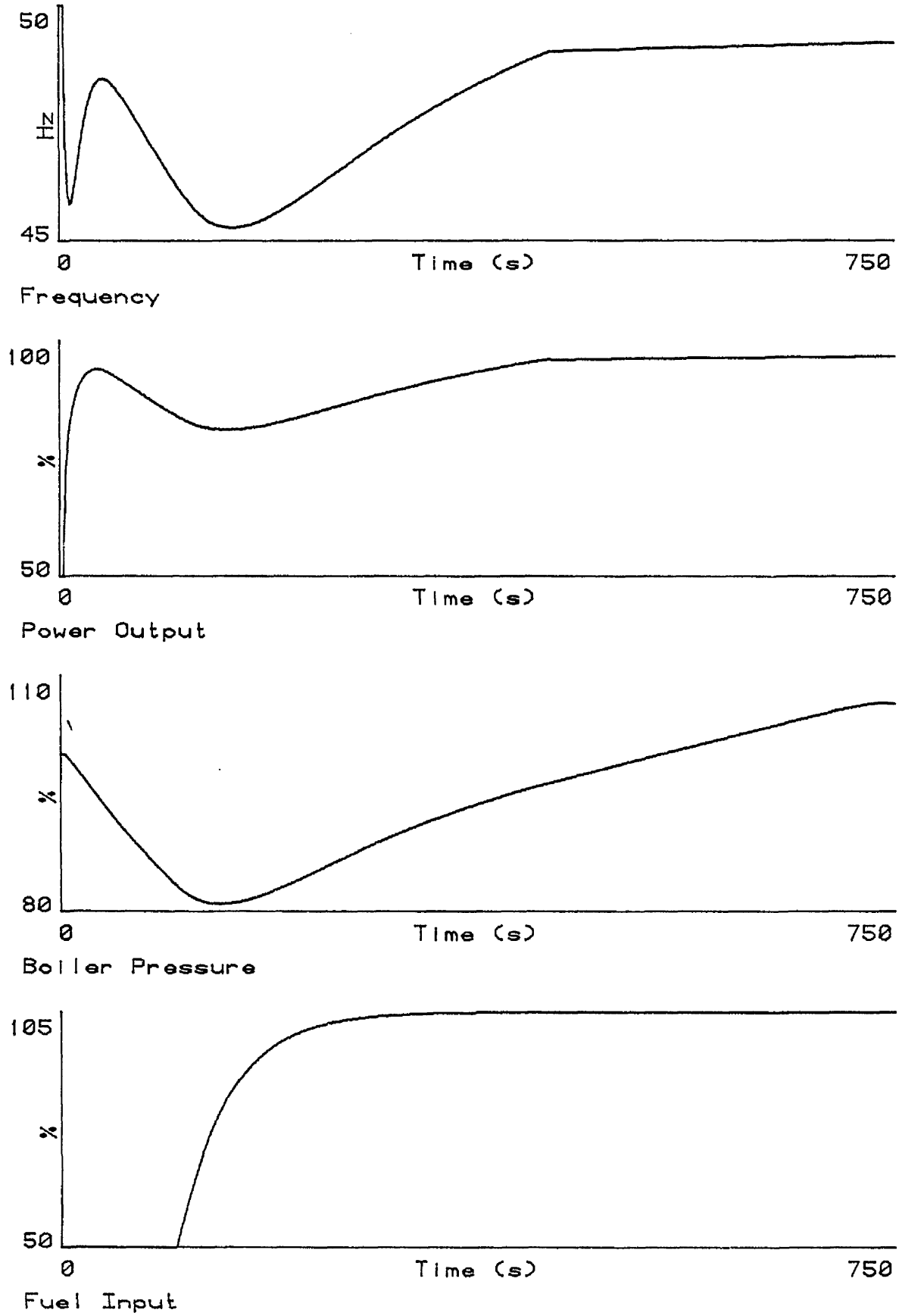


Figure 9.4 - Closed Loop Response

50% Step in Load

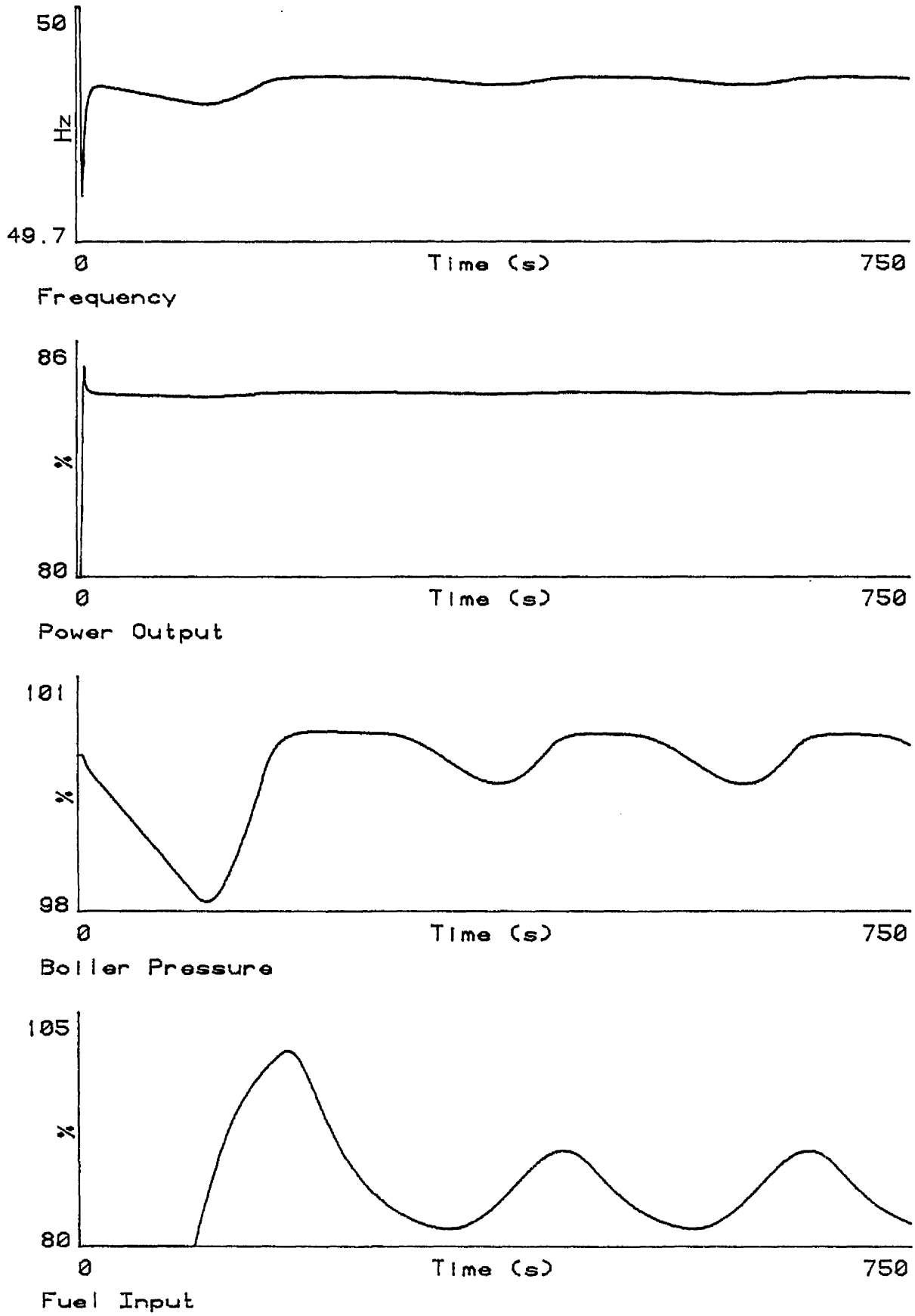
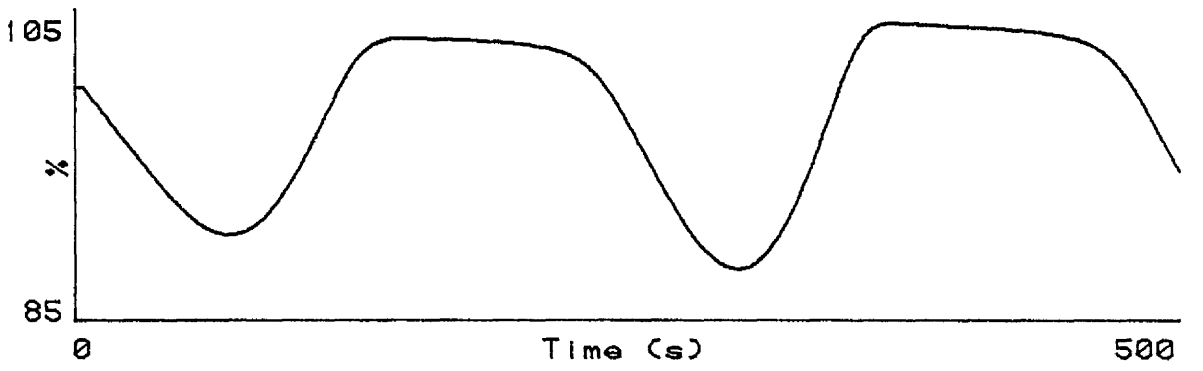
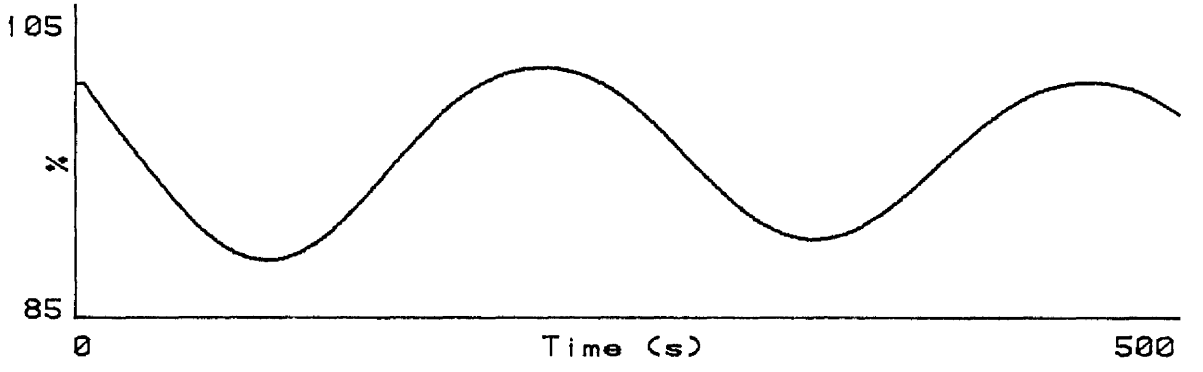


Figure 9.5 - Closed Loop Response

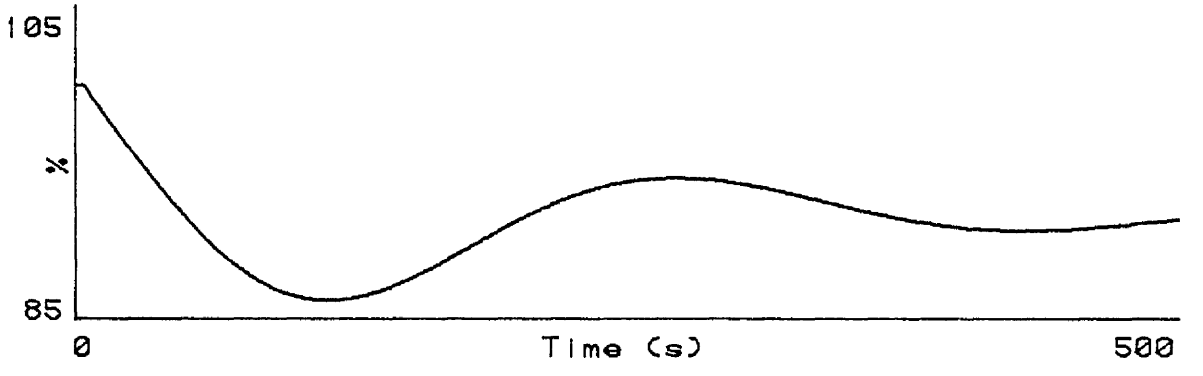
5% Step In Load



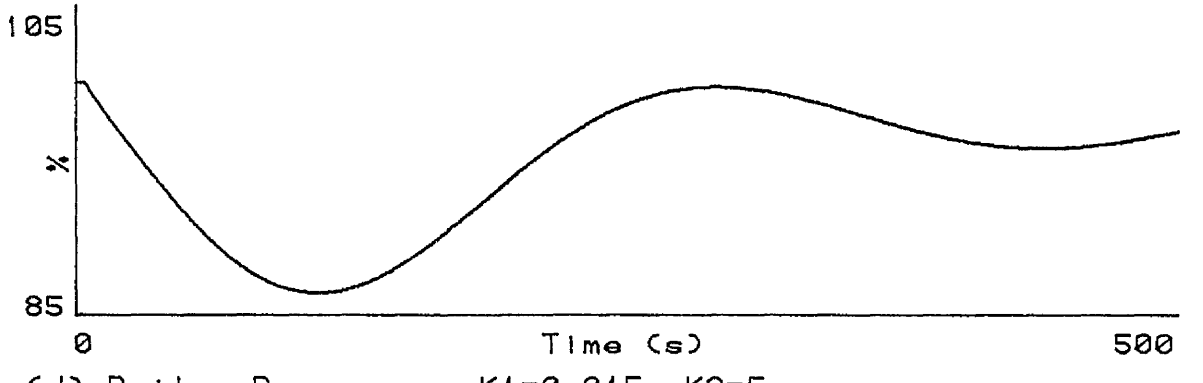
(a) Boiler Pressure - K1=0.04, K2=20.



(b) Boiler Pressure - K1=0.0, K2=10.



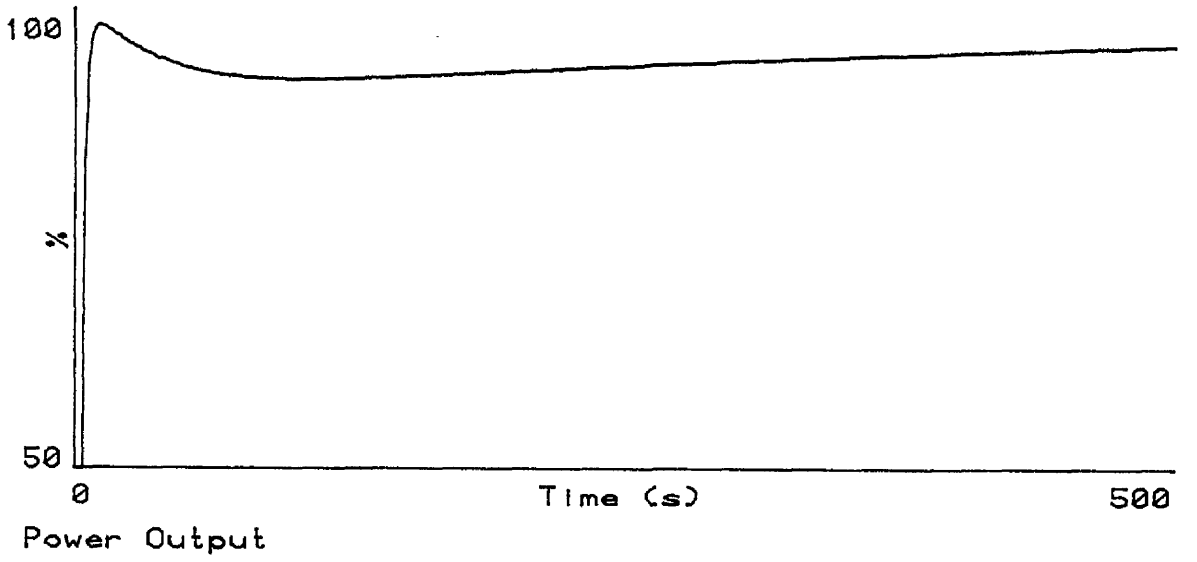
(c) Boiler Pressure - K1=0.0, K2=5.

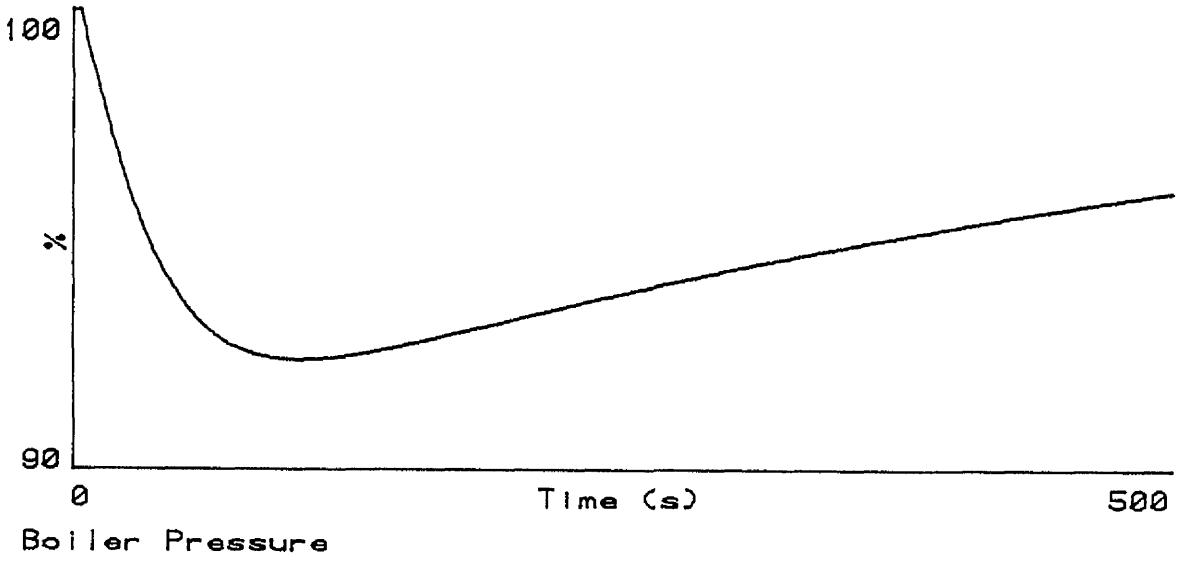


(d) Boiler Pressure - K1=0.015, K2=5.

Figure 9.6 - Open Loop Response

Effect of Varying PID Controller Parameters





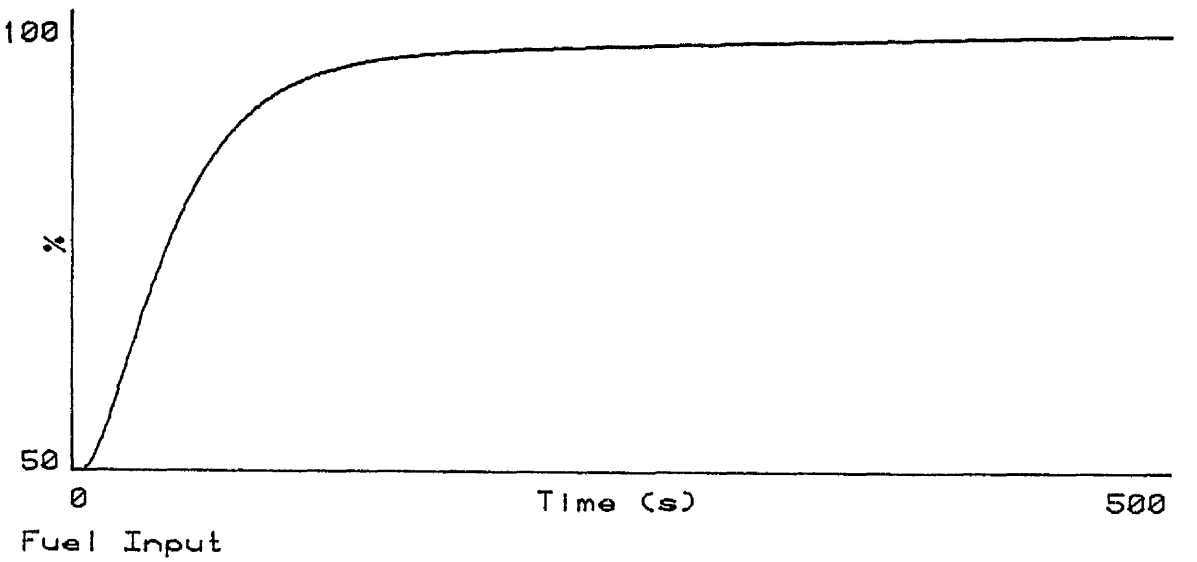
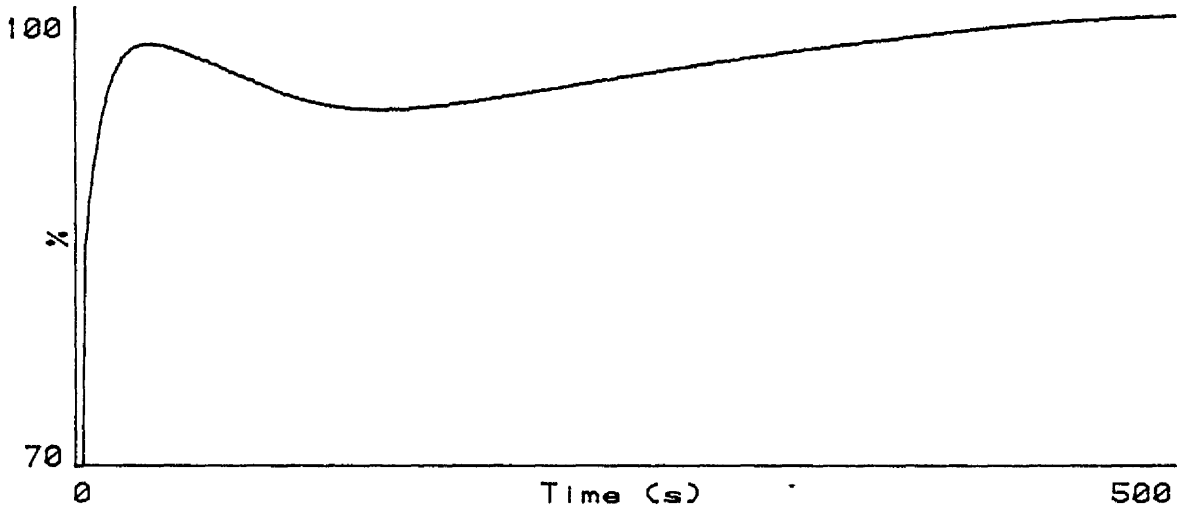
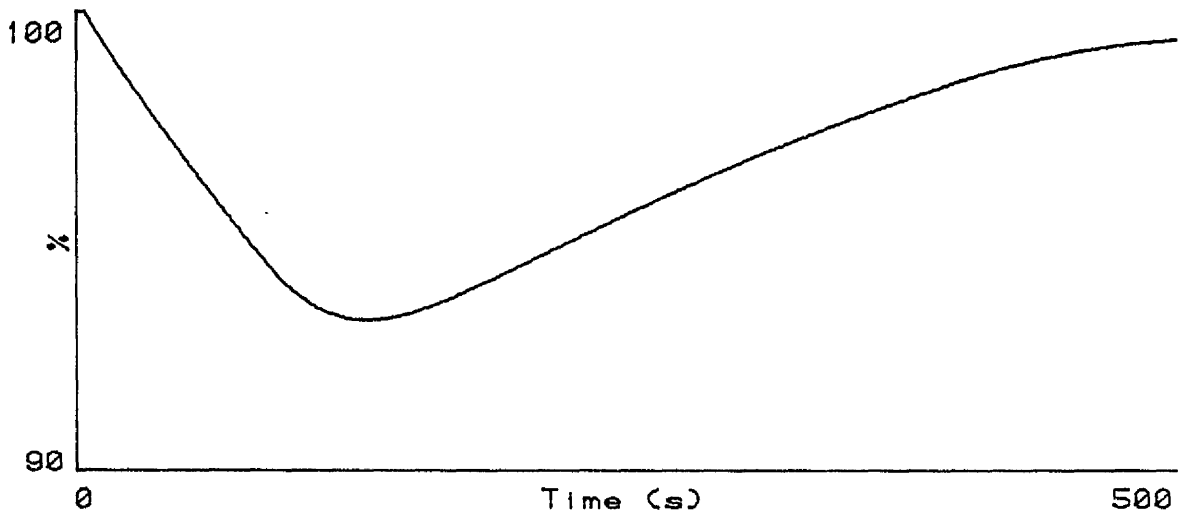


Figure 9.7 - Open Loop Response

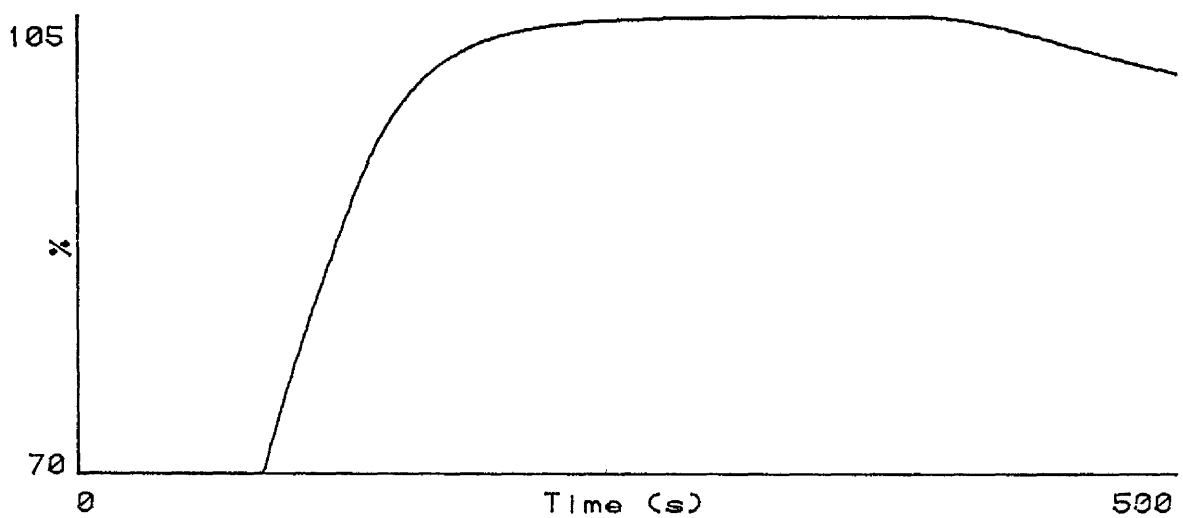
50% Step in Governor Valve Position



Power Output



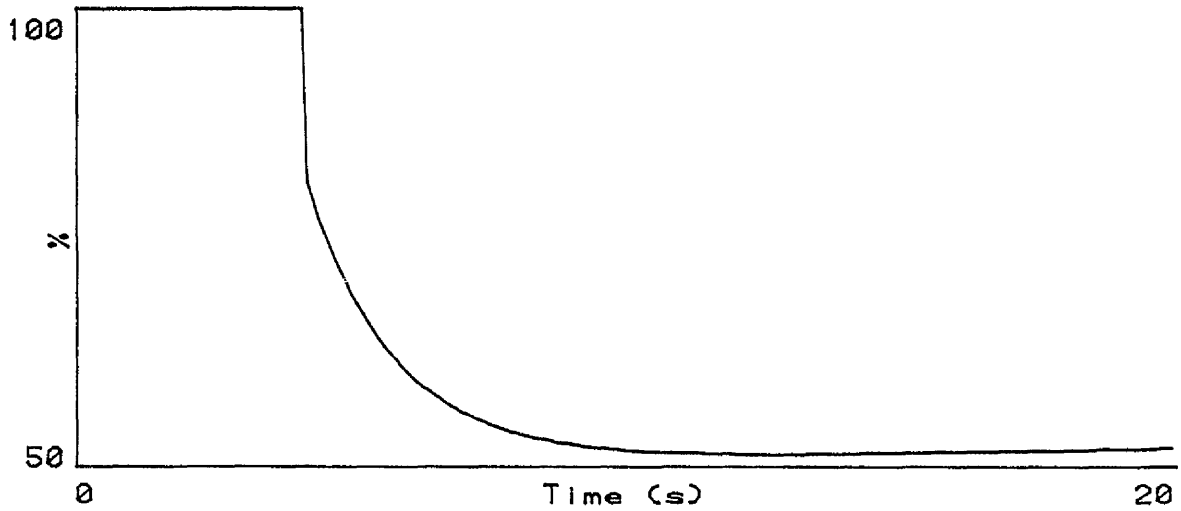
Boiler Pressure



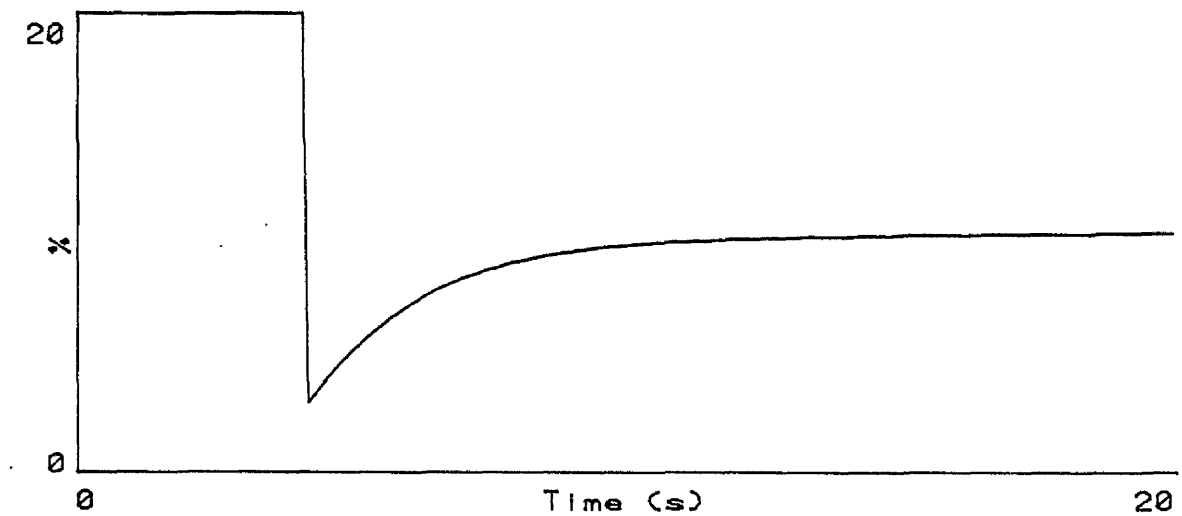
Fuel Input

Figure 9.8 - Open Loop Response

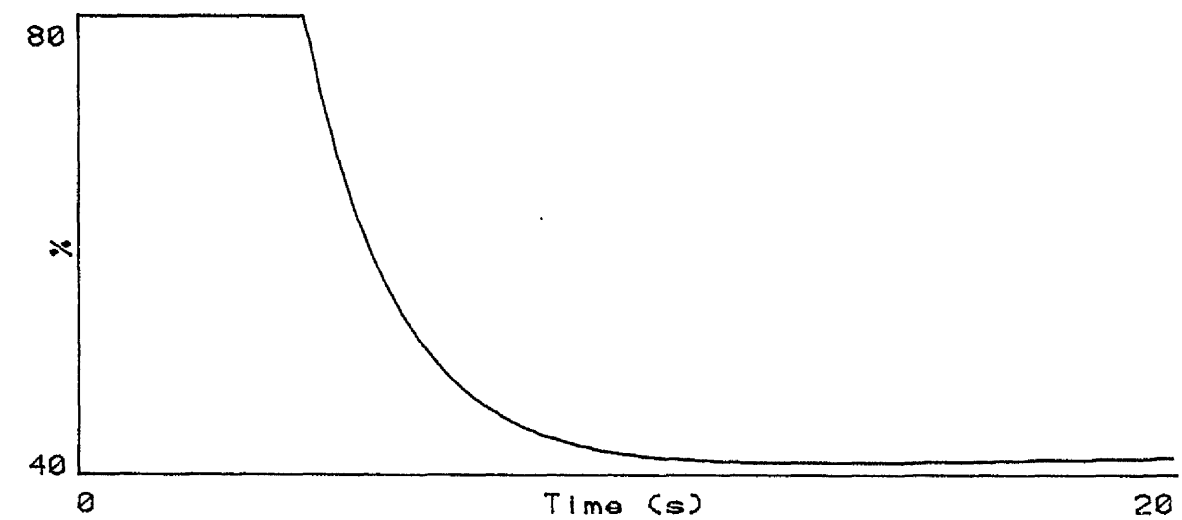
30% Step in Governor Valve Position



Total Power Output



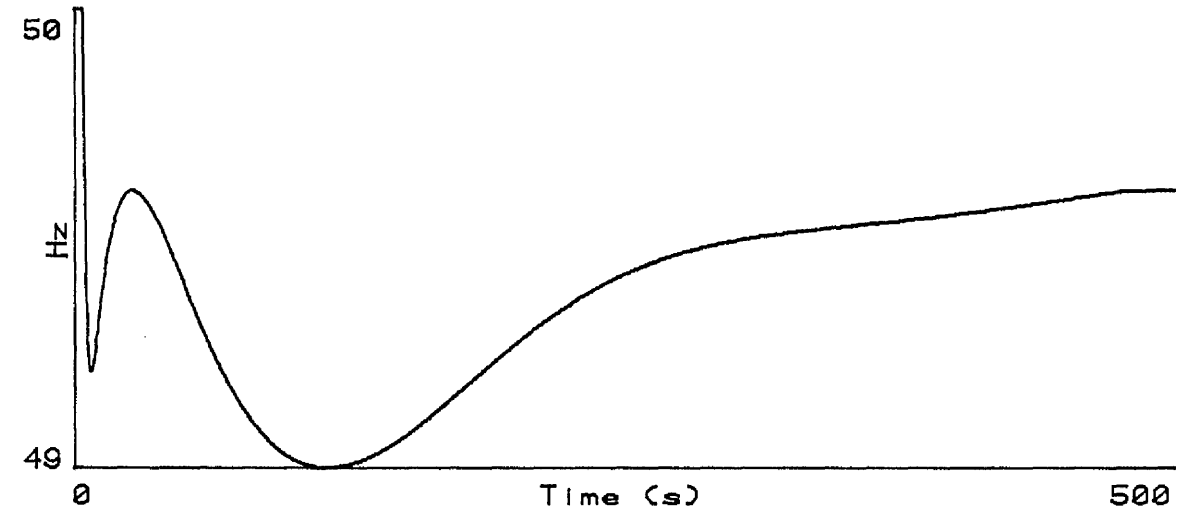
Power Output from HP Cylinder



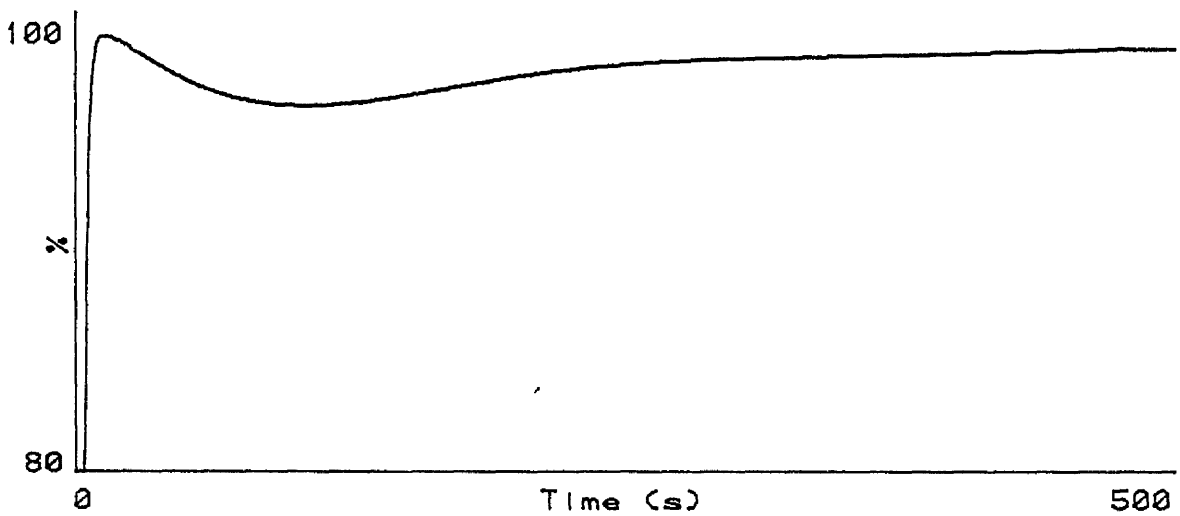
Power Output from IP/LP Cylinder

Figure 9.9 - Turbine Open Loop Response

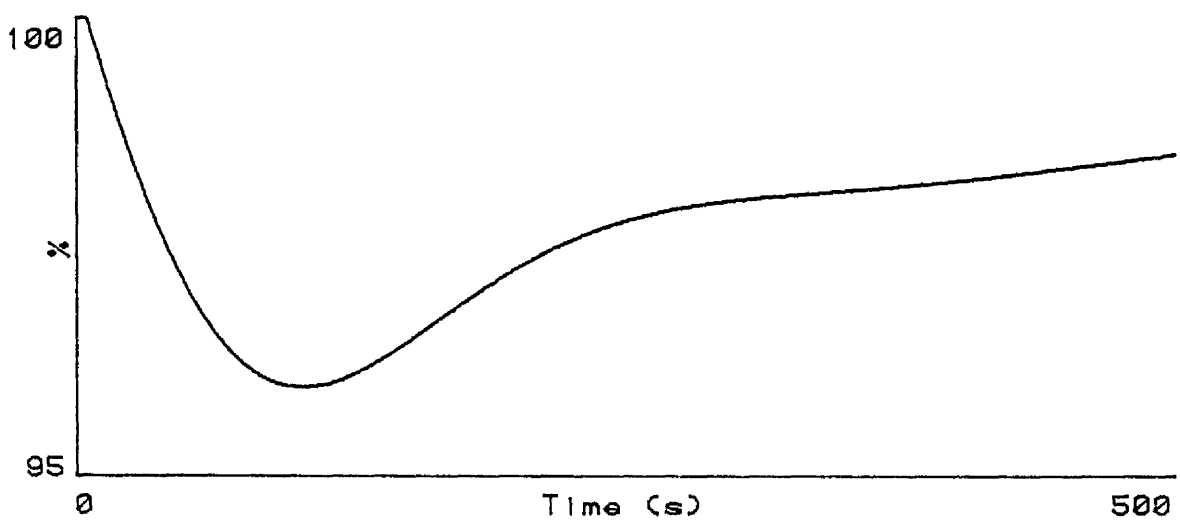
50% Step in Governor Valve Position



Frequency



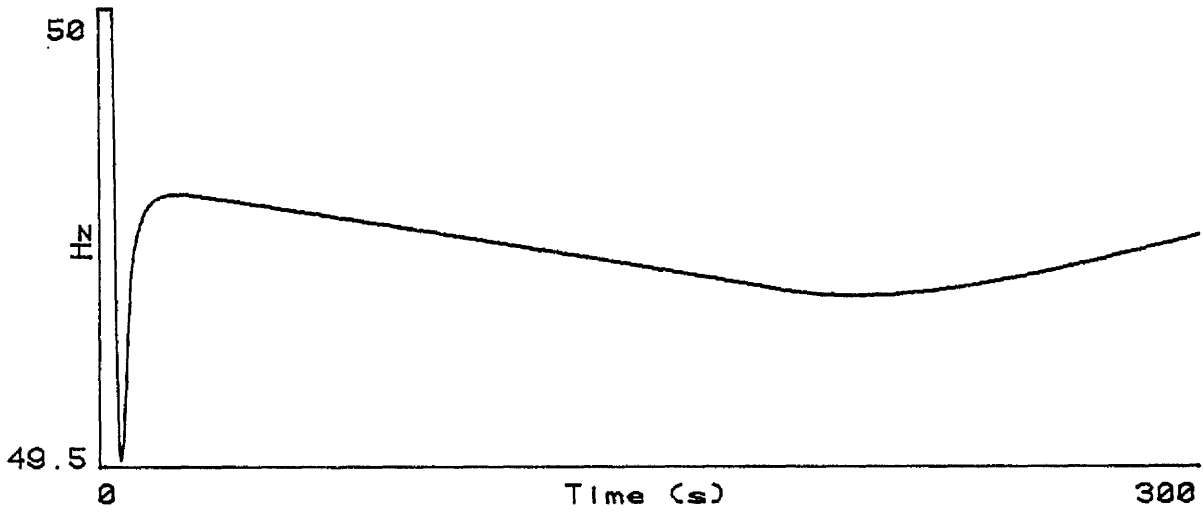
Power Output



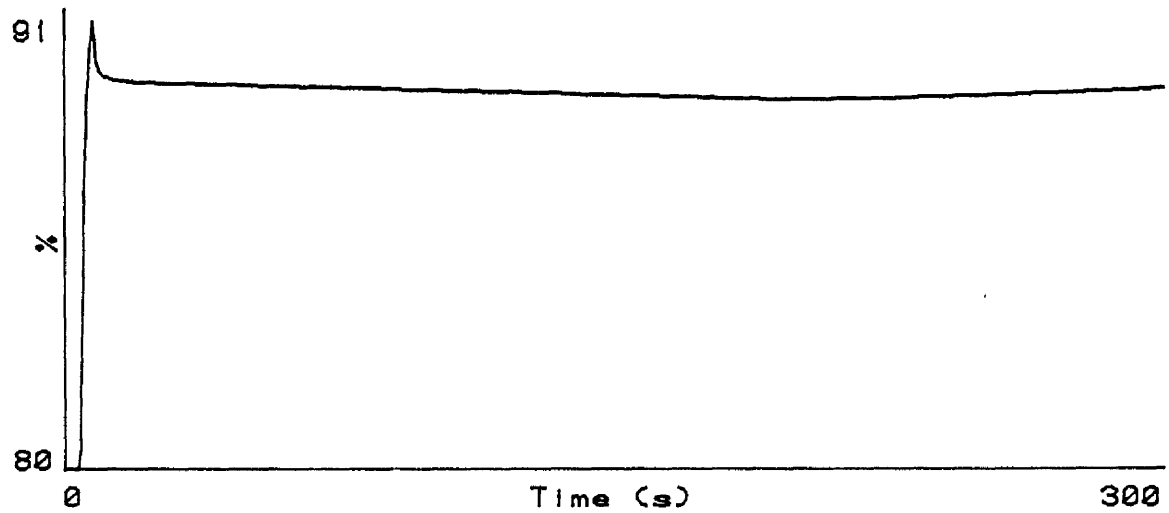
Boiler Pressure

Figure 9.10 - Closed Loop Response

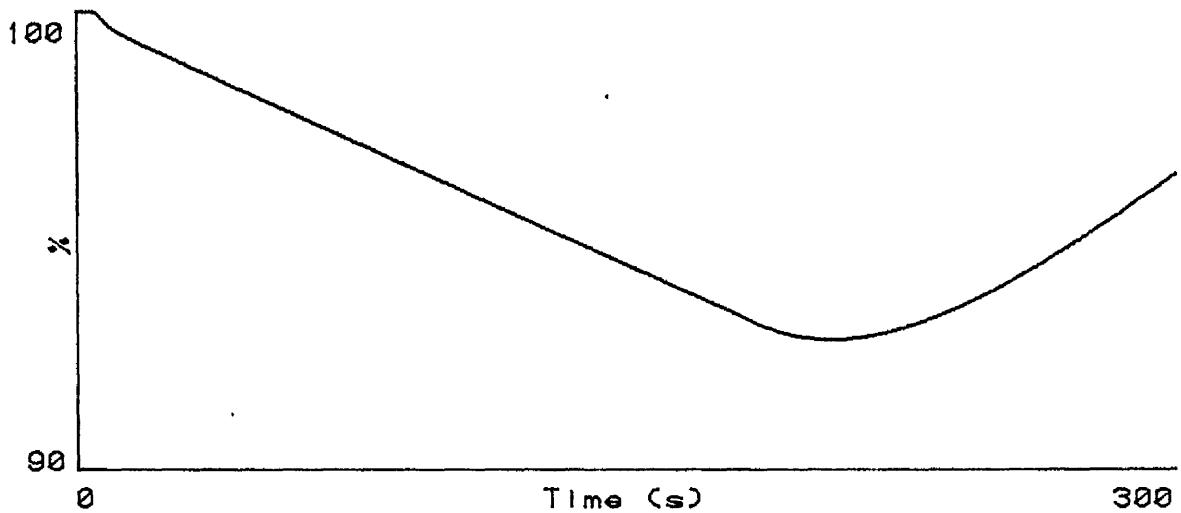
20% Step In Load



Frequency



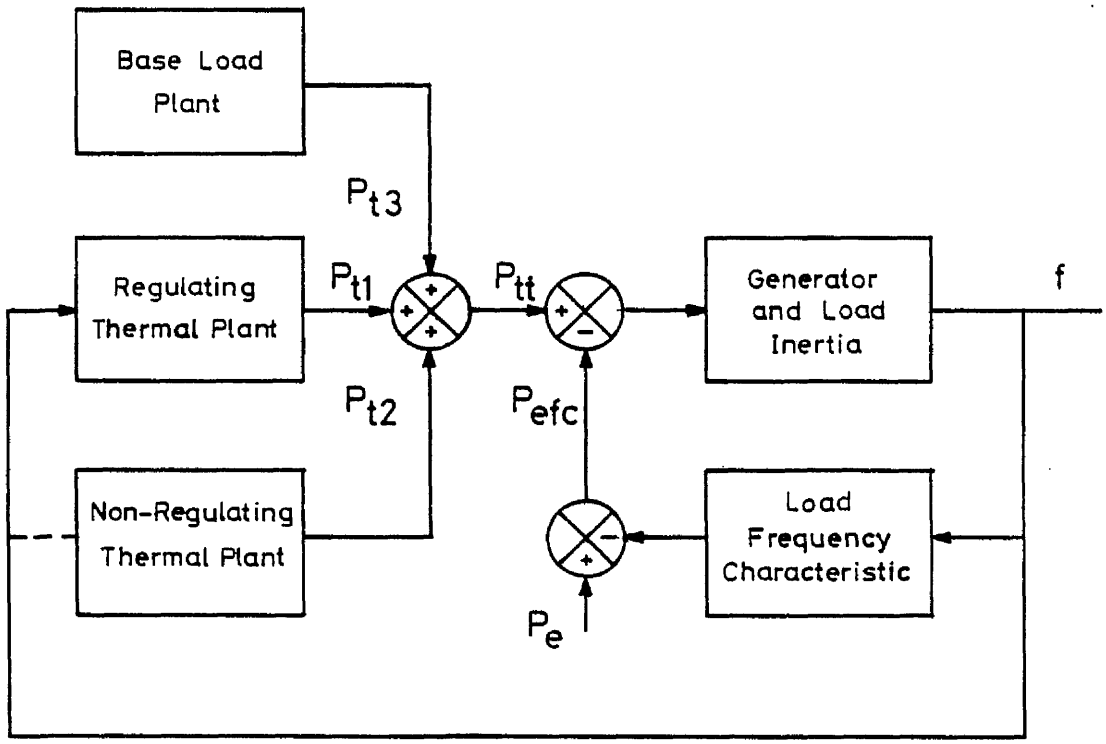
Power Output



Boiler Pressure

Figure 9.11 - Closed Loop Response

10% Step In Load



P_{t1} - Power Output of Regulating Plant

P_{t2} - Power Output of Non-regulating Plant

P_{t3} - Power Output of Base Load Plant

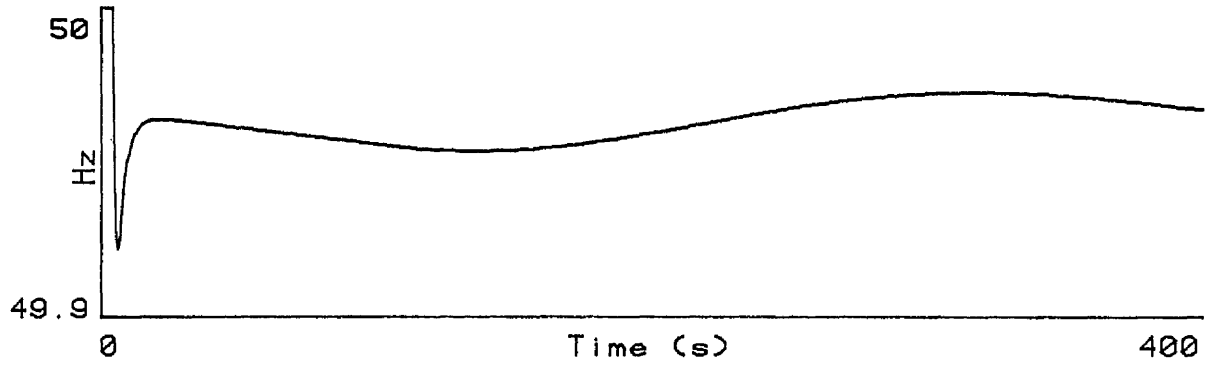
P_{tt} - Total Generated Power

P_e - Load Power

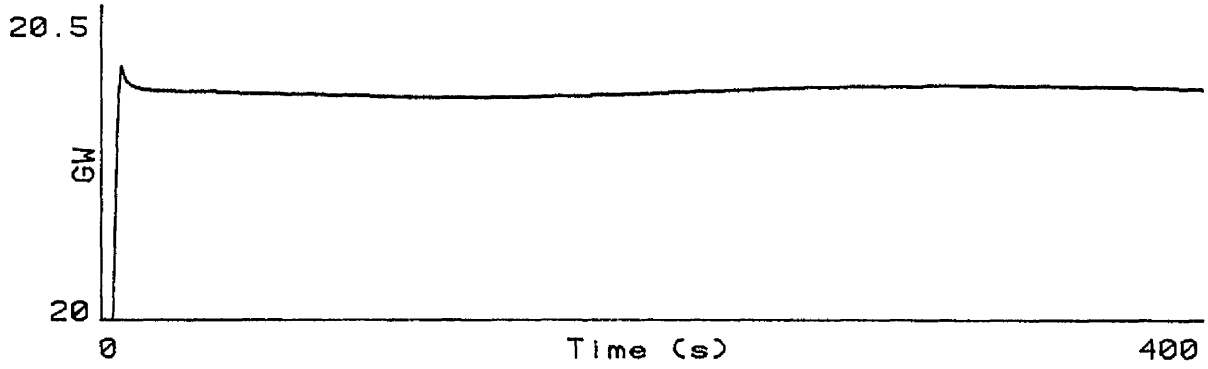
P_{efc} - Load Power Frequency Corrected

f - Frequency

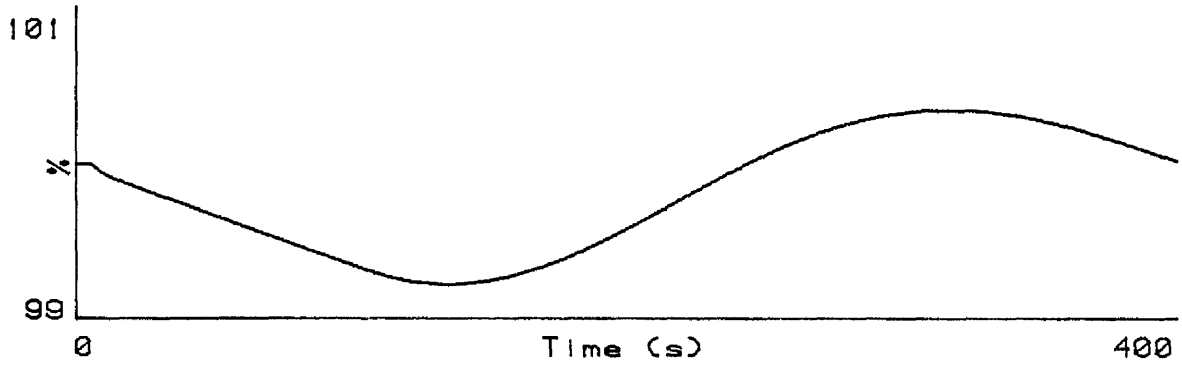
Figure 9.12 - Power System Model



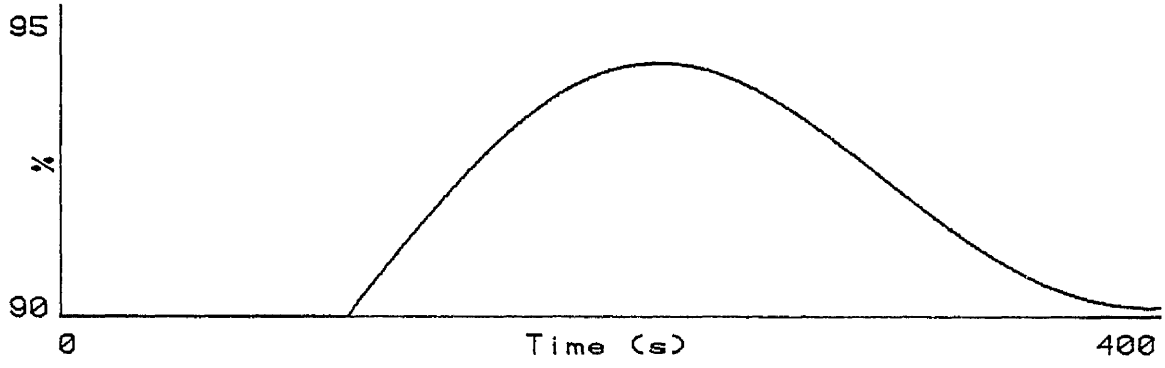
Frequency



Power Output from Regulating Plant



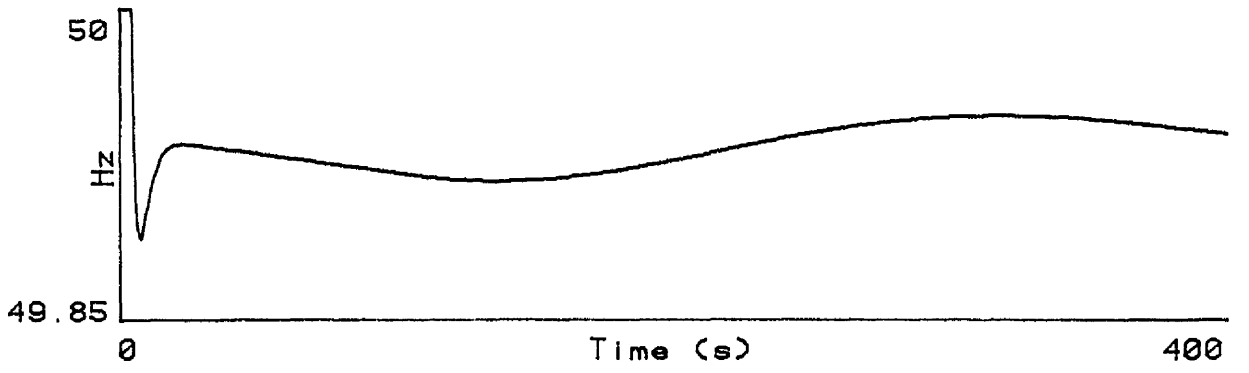
Boiler Pressure



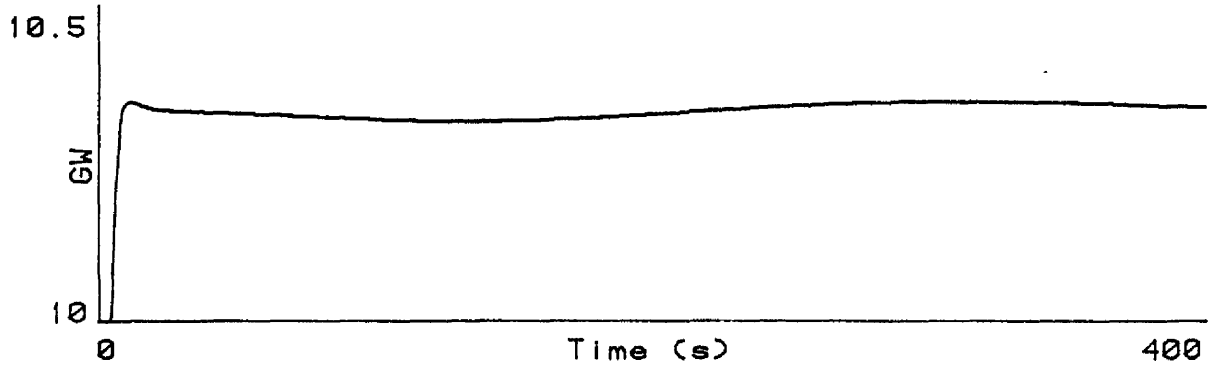
Fuel Input

Figure 9.13 – Pump Starting – 90% Capacity

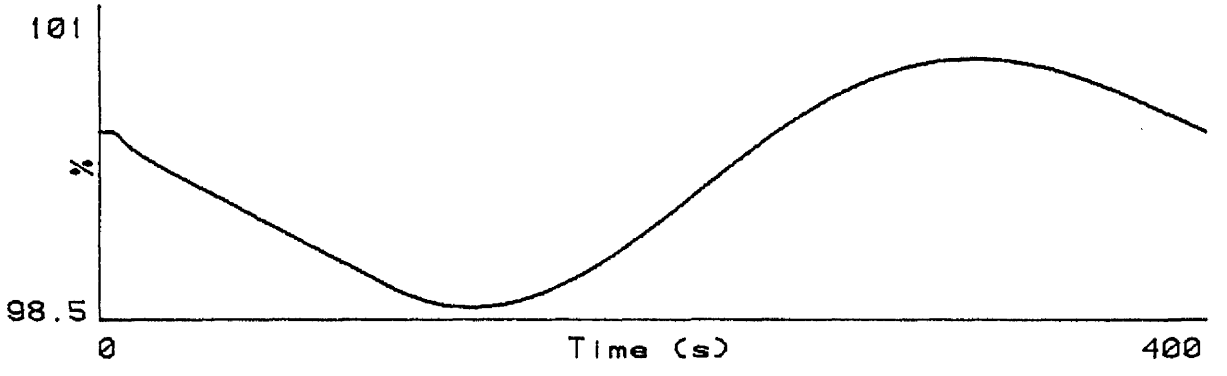
100% Regulating Plant, 100s Mill Start Delay



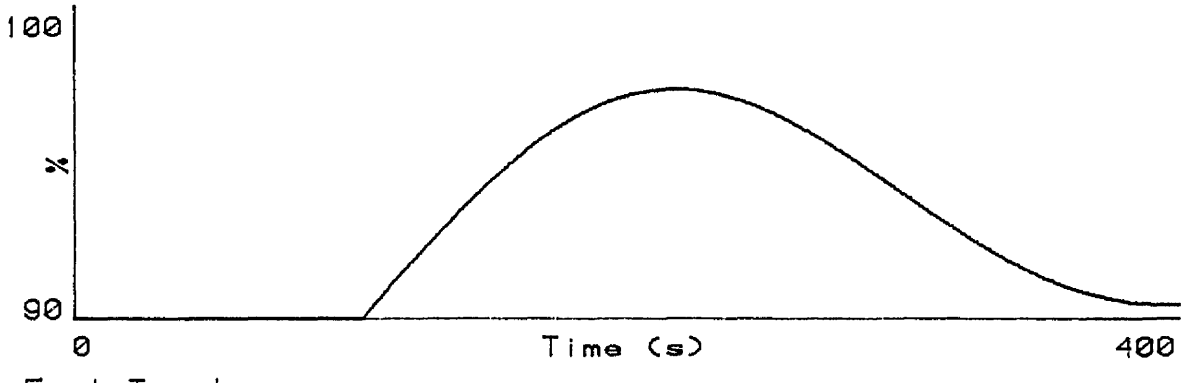
Frequency



Power Output from Regulating Plant



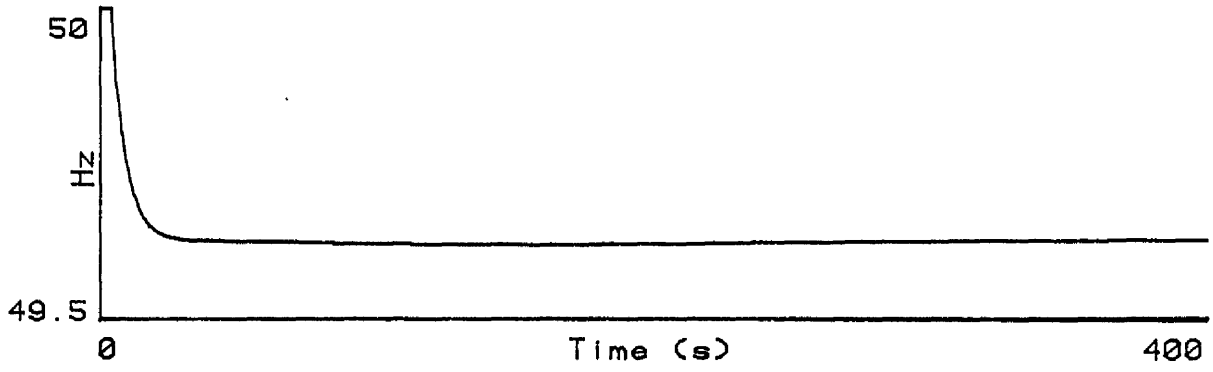
Boiler Pressure



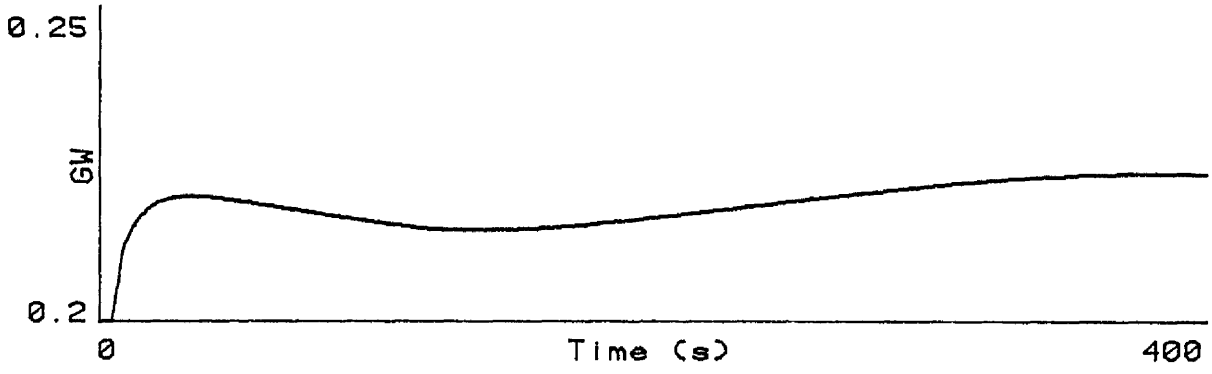
Fuel Input

Figure 9.14 - Pump Starting - 90% Capacity

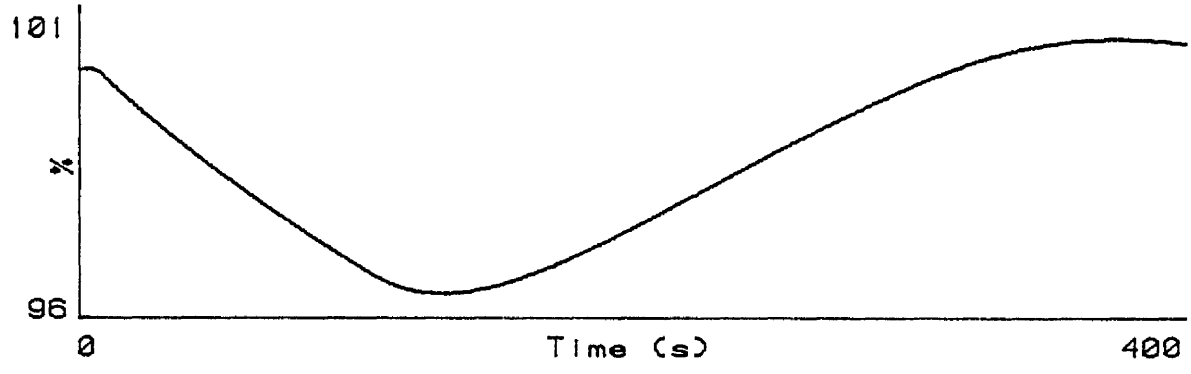
50% Regulating Plant, 100s Mill Start Delay



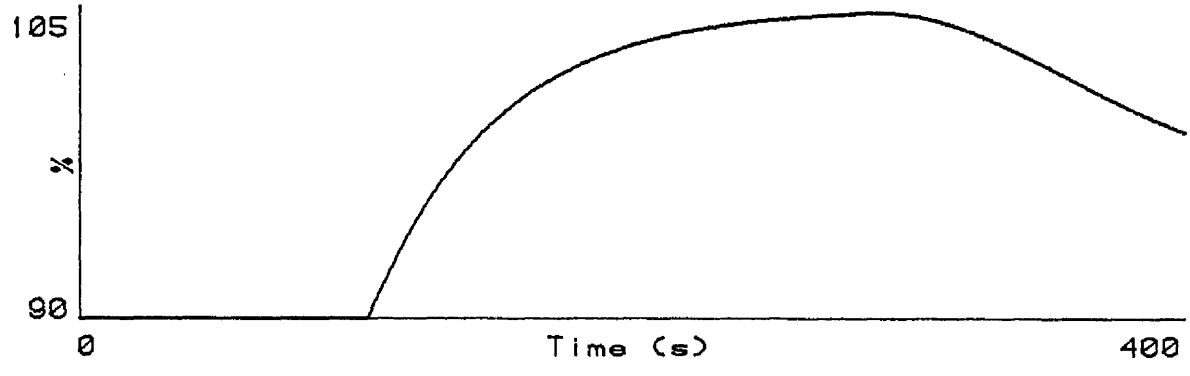
Frequency



Power Output from Regulating Plant



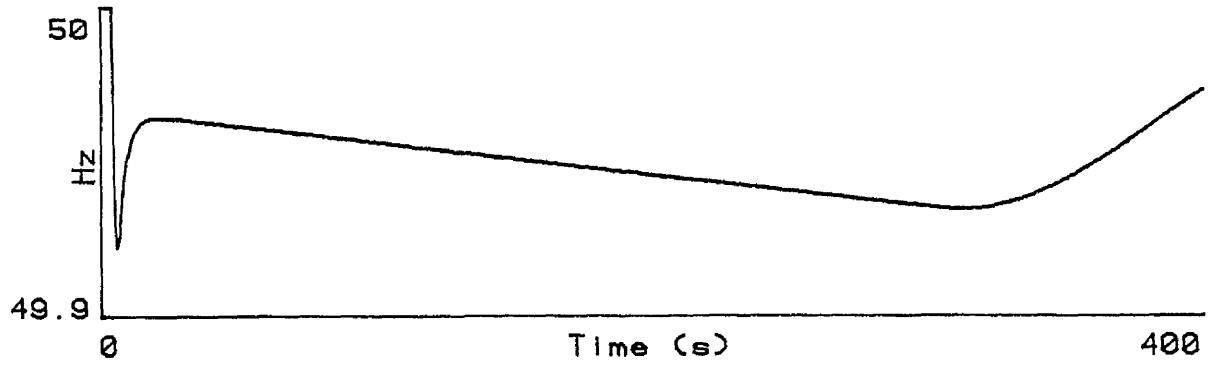
Boiler Pressure



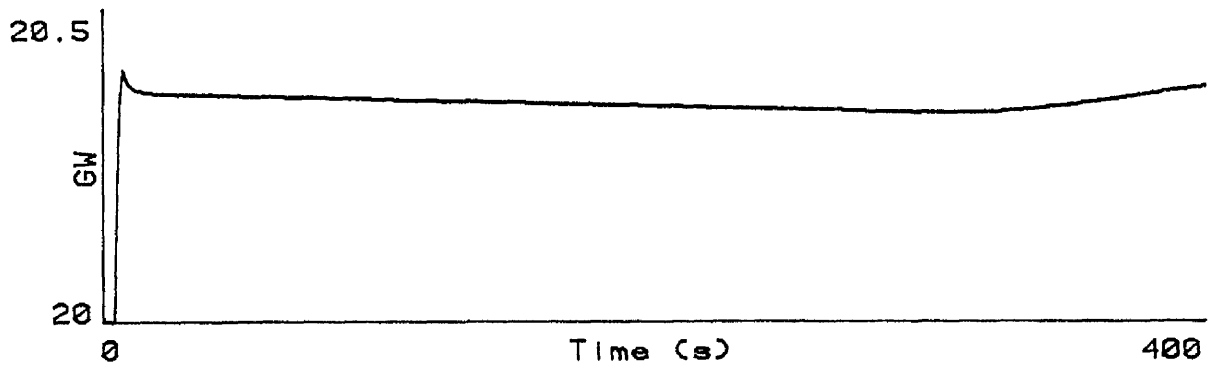
Fuel Input

Figure 9.15 - Pump Starting - 90% Capacity

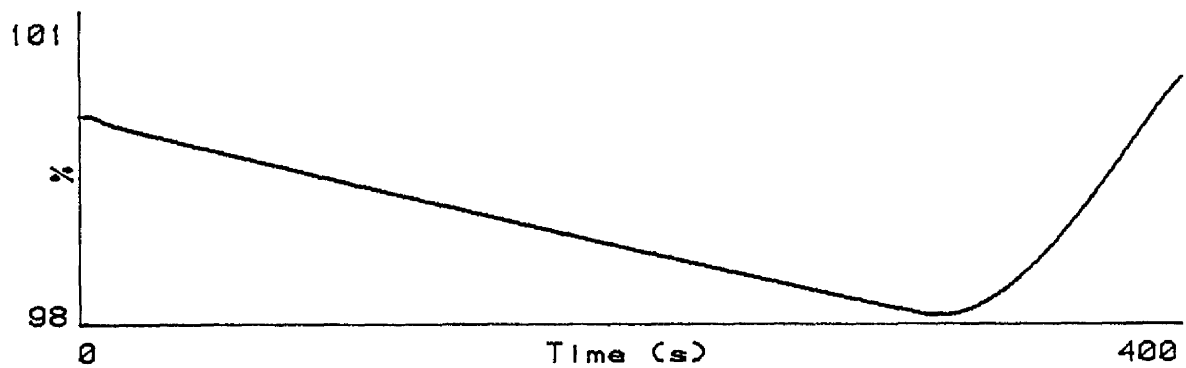
1% Regulating Plant, 100s Mill Start Time



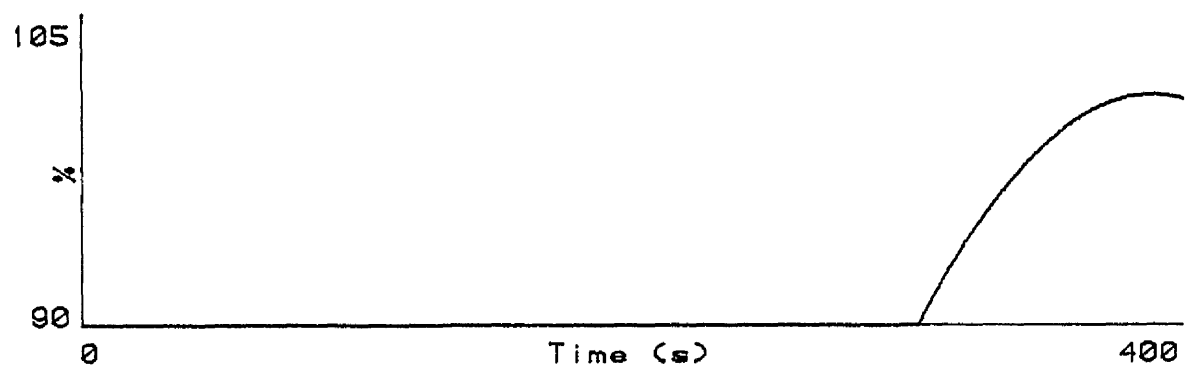
Frequency



Power Output from Regulating Plant



Boiler Pressure



Fuel Input

Figure 9.16 - Pump Starting - 90% Capacity

100% Regulating Plant, 300s Mill Start Time

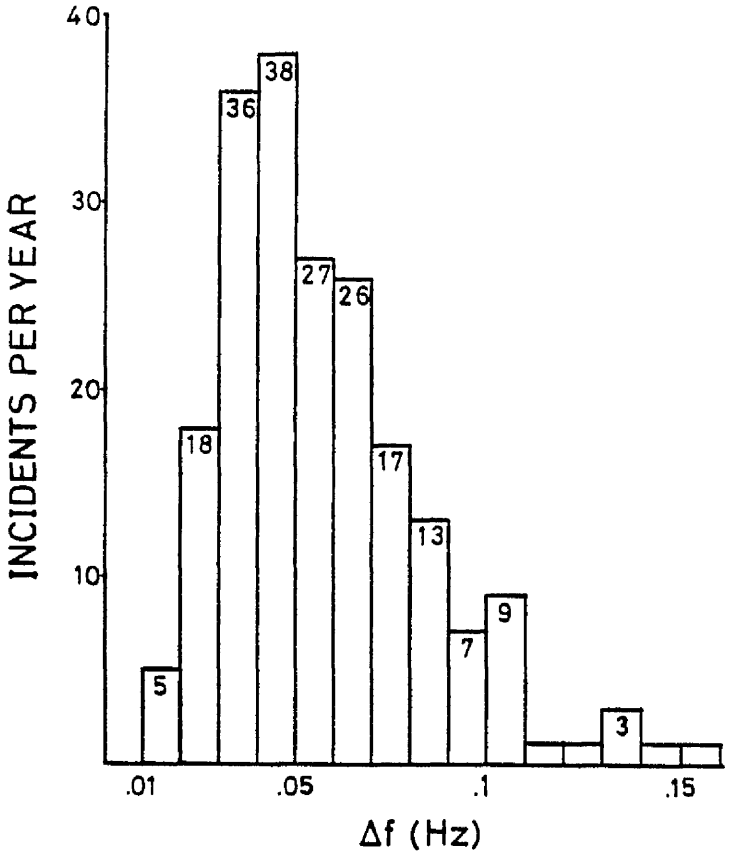
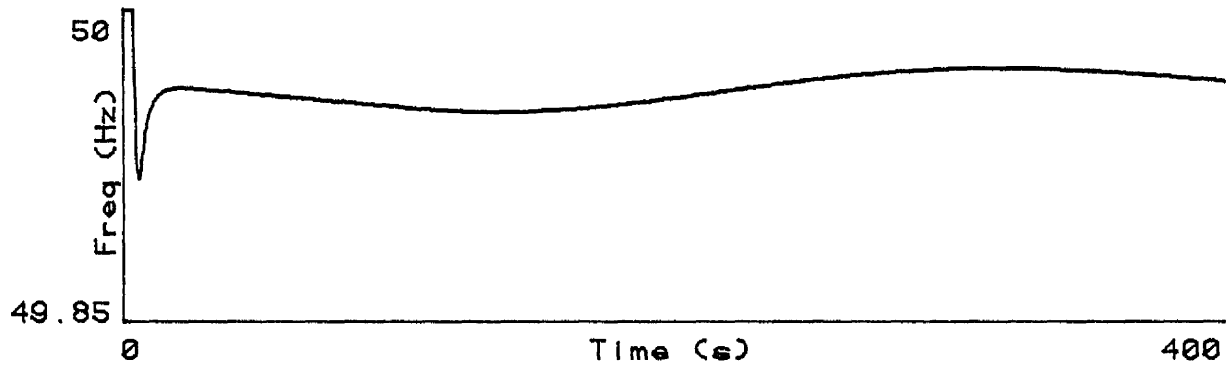
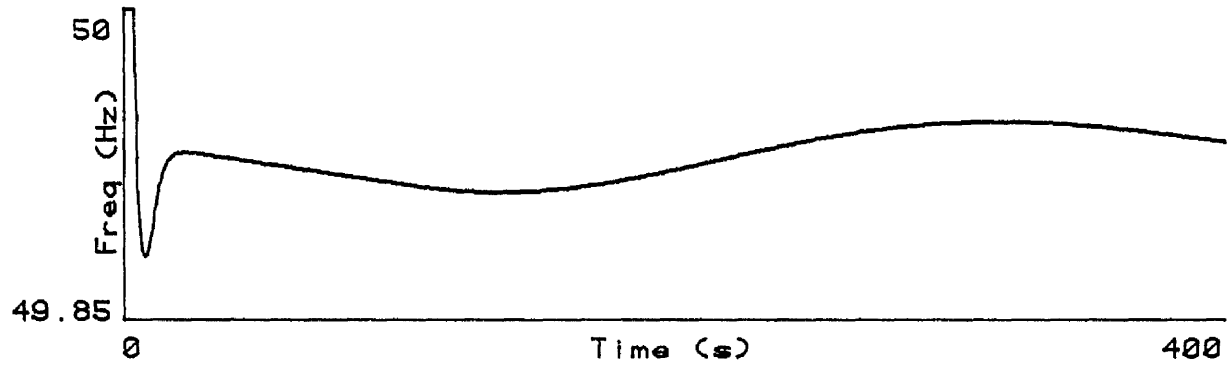


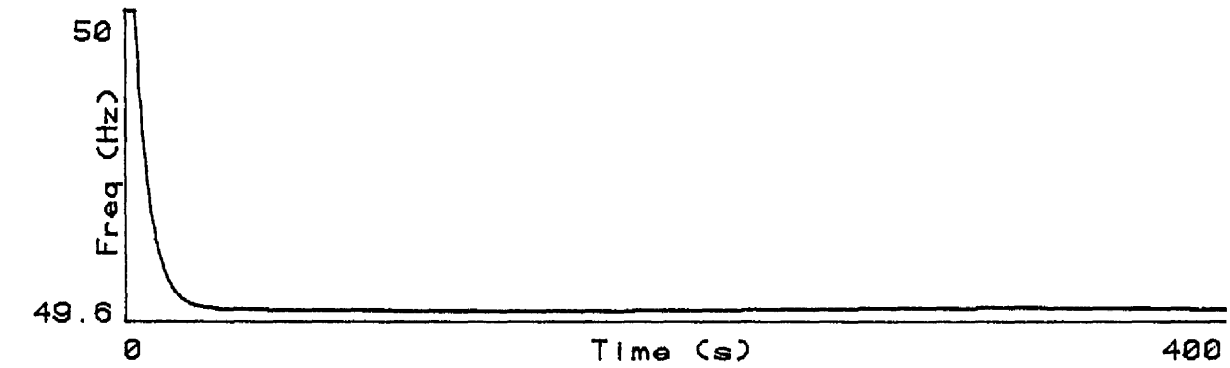
Figure 9.17 - Frequency Transients due to Generation Losses (1976)



(a) 100% Regulating Plant, 100s Mill Start Time



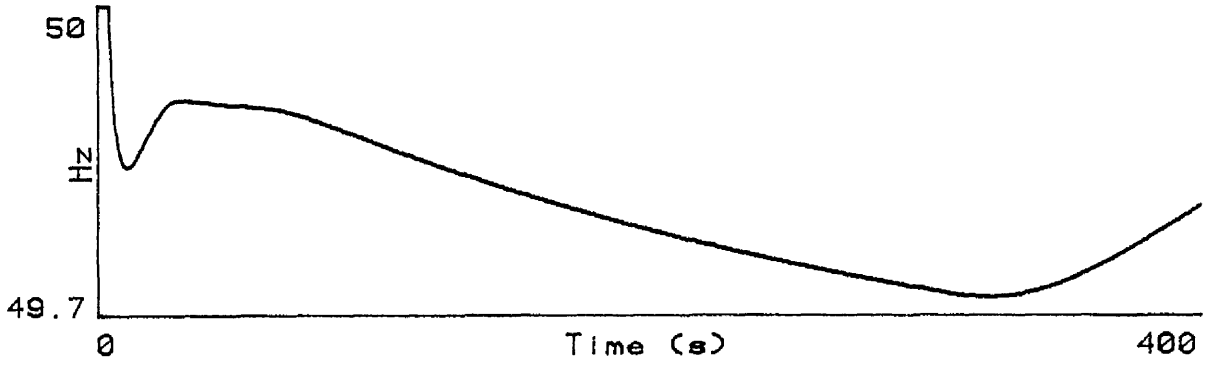
(b) 50% Regulating Plant, 100s Mill Start Time



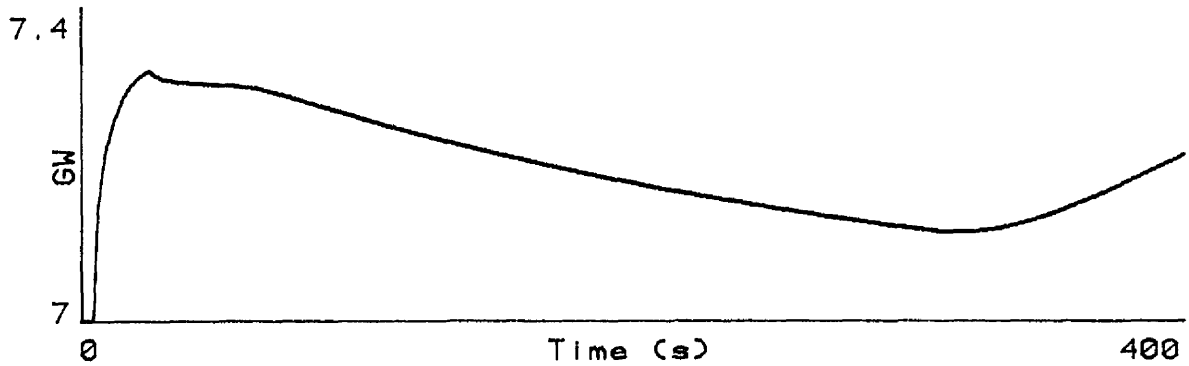
(c) 1% Regulating Plant, 100s Mill Start Time

(d) 100% Regulating Plant, 300s Mill Start Time

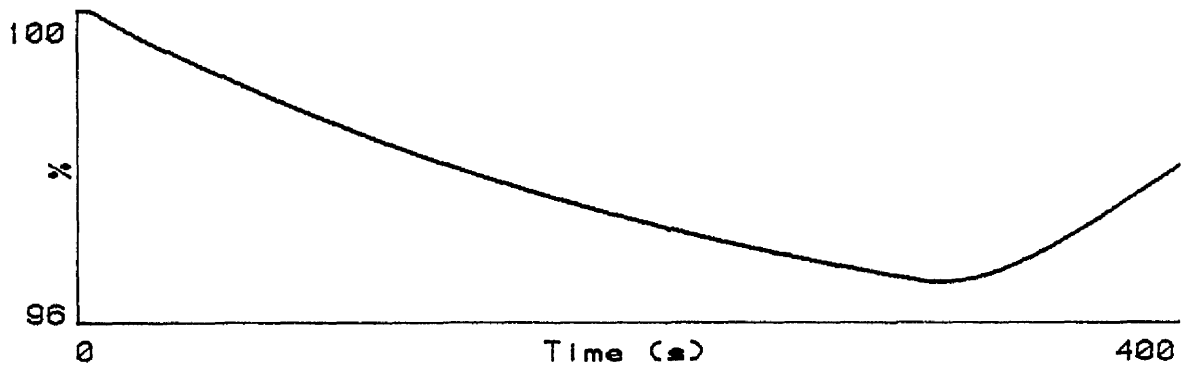
Figure 9.18 - Pump Starting - 95% Capacity



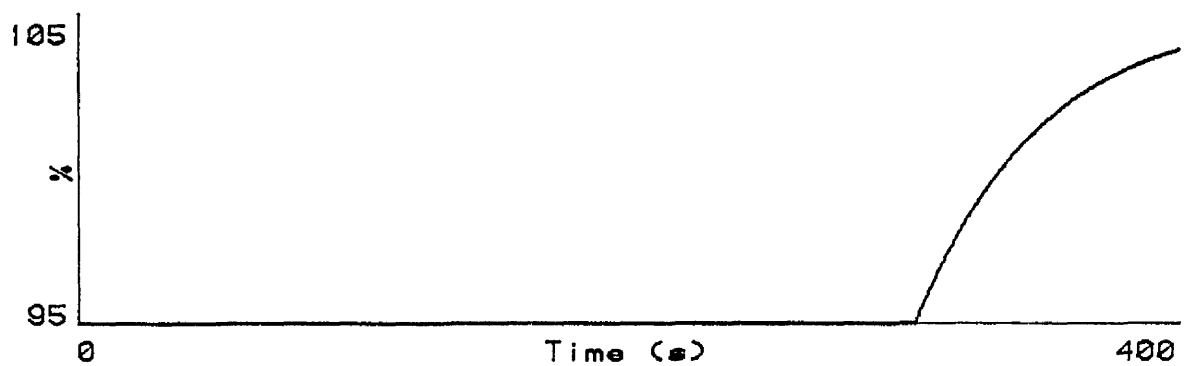
Frequency



Power Output from Regulating Plant



Boiler Pressure



Fuel Input

Figure 9.19 - Typical System Response

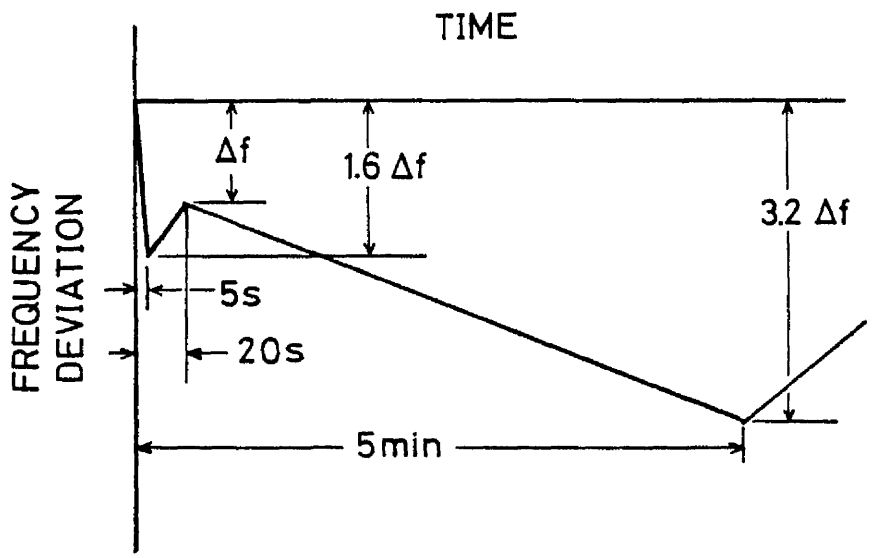


Figure 9.20 - Stylised Frequency Transient

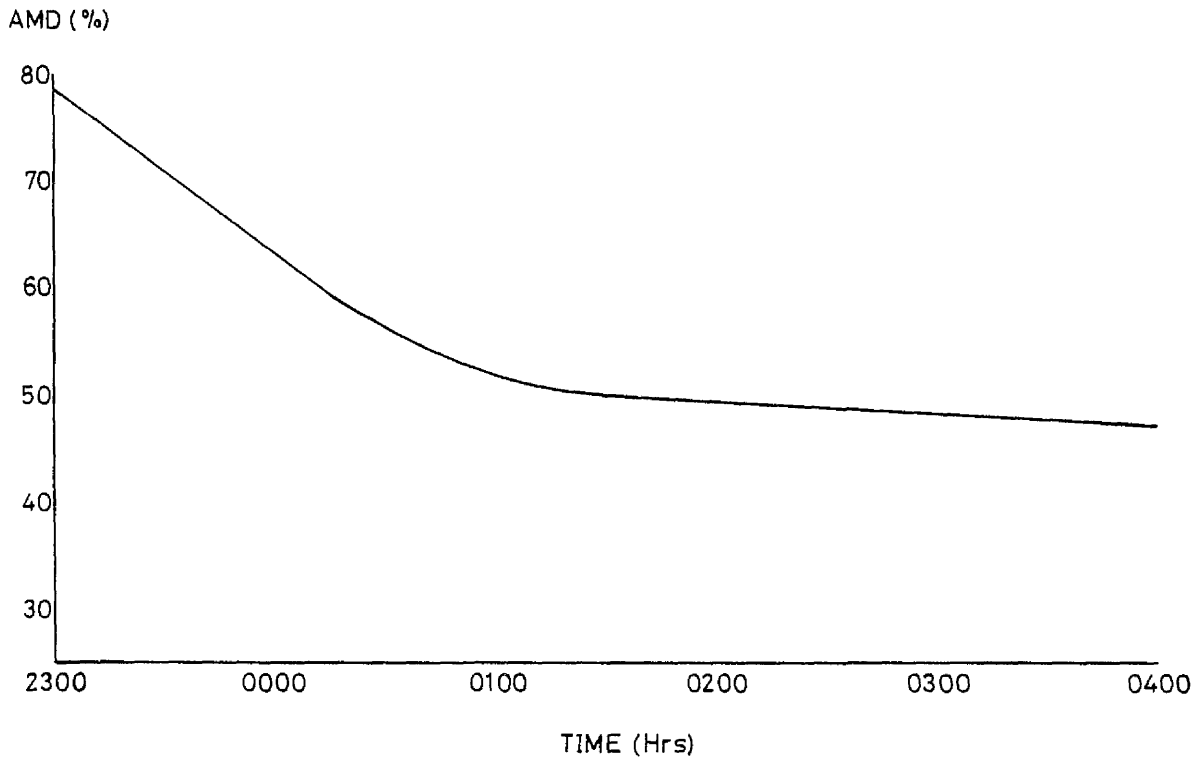
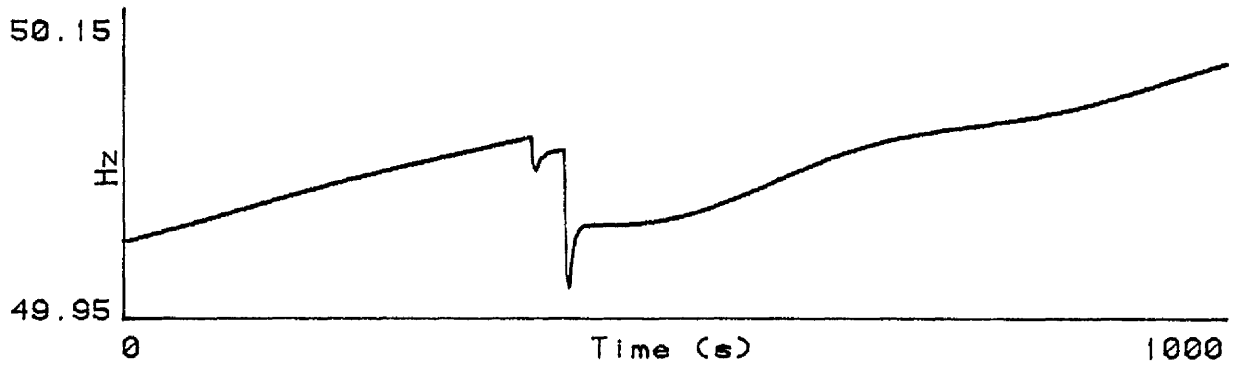
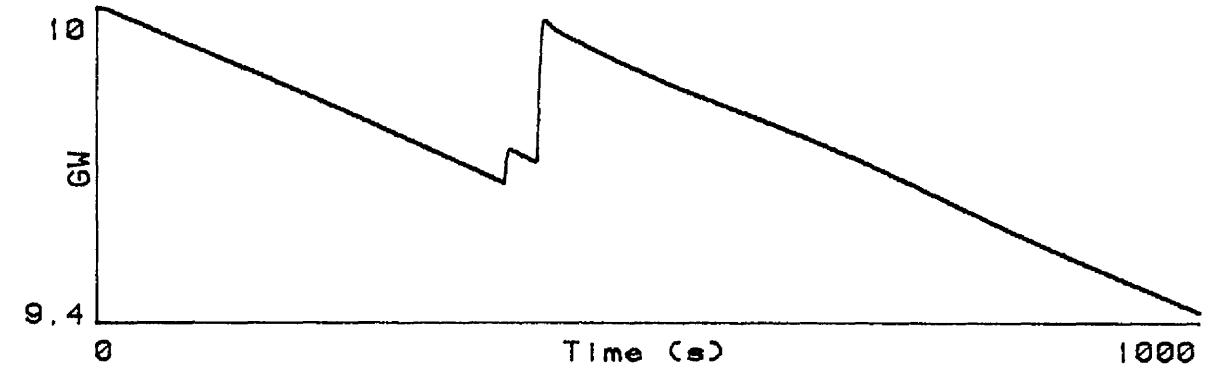


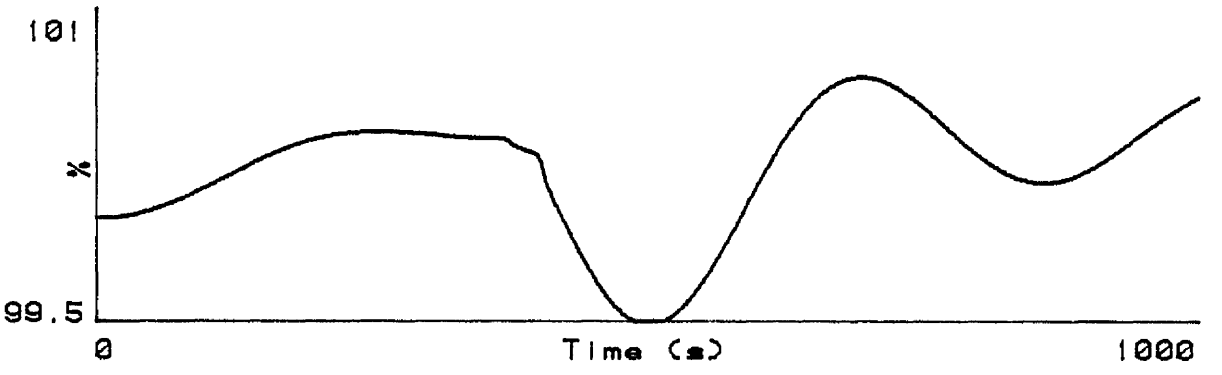
Figure 9.21 - Daily Load as a Percentage of Average Maximum Demand



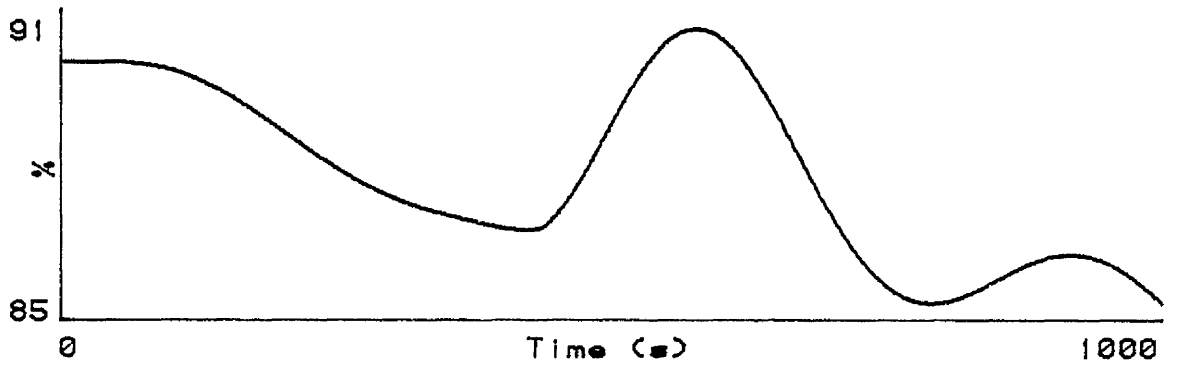
Frequency



Power Output from Regulating Plant



Boiler Pressure



Fuel Input

Figure 9.22 - Two Stage Pump Starting

CHAPTER 10

GENERAL CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK

The work reported in this thesis was concentrated on two main areas of study: the deveopment of a detailed model of a hydro generator and the validation of this model by comparison with site results from Loch Sloy Power Station; the implementation of a translator, using the STAGE2 macroprocessor, for a simulation package and the subsequent development of this package into a simulation language (GUILDS). The general conclusions of the work are presented in this chapter along with some recommendations of areas where further study could be carried out.

10.1 General Conclusions

The simulation of the Sloy hydro generator from Bryce[1] and Findlay[3] has been developed considerably to give a more detailed representation of the system. Additions to the simulation included models of the surge shaft, the relief valve and the non-linearities between the servo position and the generator power output. A more accurate representation of the governor/controller was used including the load limit and servo set points and the frequency transducer. Towards the end of the project, the governor model was further modified to use the same integration method (Euler) and integration interval (0.1s) as the site governor although the rest of the model equations were solved using a Runge-Kutta technique (Chapter 7). Also included in the model were some additional features required for studying the run-up and load rejection behaviour.

On the whole, this model proved to be satisfactory giving good agreement with the site results under most operating conditions. There are, howerer still some residual doubts about the value of the inertia constant (T_a) that should be used and the exact form of the non-linear relationship between servo position and power output. Although further investigations in these areas would be valuable, the agreement between the site and simulation results is considered to be sufficiently good to validate the use of the model as a design tool for future governor studies. In addition, the experience gained from developing this model would enable a model of a different hydro generator to be more easily formulated.

The feasibility of using the STAGE2 macroprocessor as a translator for a FORTRAN simulation package has been adequately demonstrated. The culmination of this work in the general purpose simulation language GUILDS, greatly assisted the hydro generator studies carried out as part of this project. The continual use of GUILDS for a real application, although perhaps biasing the development to a certain extent, ensured that the language software was throughly tested and gave insight into areas where improvements could be made.

GUILDS has found applications in other areas of engineering and is currently being used by engineers from the NSHEB for a variety of different studies. Further work is being carried out by Dr. D.J. Winning of the Department of Electronics and Electrical Engineering at Glasgow University to implement GUILDS on a microprocessor based system which should make simulation facilities available to considerably more users.

The work on the Frequency Disturbance System presented in Chapter 8 serves as a good example of the use of simulation in general, and GUILDS in particular, for carrying out a contol system design study. The FDS was implemented and tested as part of a simulation

model and it was shown that the speed of response of the hydro generator could be improved by the use of this type of non-linear governing system. However, it was felt that, having done the initial design work and tuning using the simulation, final assesment of the system would require site tests to be carried out.

The thermal plant study (Chapter 9) forms another area of work to which simulation was successfully applied. The validity of the model was demonstrated and some simulation runs were carried out for the NSHEB to show the effect on the system frequency of starting a large pump-tubine in a pumped storage generating station.

As a continuation of this latter section of work, A.G. Marshall of the NSHEB is developing the thermal plant model to include a more detailed representation of a hydro generator (based on the model presented in this thesis) to study the operation of a combined hydro-thermal power system.

10.2 Further Work

Where appropriate, remarks have been included throughout the thesis to indicate areas where further study or development could be carried out.

In general, the hydro generator model has been shown to be a good representation of the Sloy machine but, further investigation of the servo position/power relationship and the backlash between the servo and the guide vanes is recommended to establish a more exact mathematical representation.

There is scope for further development of non-linear governors of the type described in Chapter 8, and also, for further study of adaptive governors as proposed by Findlay[3]. As noted above, further work has already commenced on the development of an integrated hydro-thermal power system model.

The simulation language (GUILDS), although complete as it stands, could be expanded considerably by the inclusion of numerous additional features. Some areas for development, identified in the course of this project, are given below:-

(1) extend the range of standard functions provided by the language;

(2) expand the keyboard interrupt facility to give the user more control over the simulation during the run allowing, for example, parameters to be altered without stopping the run;

(3) include facilities for input and interpolation of user data functions;

(4) improve the error detection and handling facilities;

(5) modify the integration routines and model structure to permit the use of different integration methods and/or time intervals for different sections of the model (as demonstrated in Section 7.7).

APPENDIX 1

This appendix contains a paper entitled "Controller Testing Facility on a 32.5MW Water Turbine" which was submitted to the IEE in early 1980 for publication in the Proceedings. The paper was susequently published in a revised form[18] in November 1980.

CONTROLLER TESTING FACILITY ON 32.5 MW WATER TURBINE

D.J. WINNING B.Sc.,Ph.D
A.G. MARSHALL B.Sc.
D.G.E. FINDLAY B.Sc.
K.H. AITKEN B.Sc.
N.F. GRANT B.Sc.

Dr Winning is a lecturer, Mr Findlay was and Mr Aitken and Mr Grant are research students in the Department of Electronics and Electrical Engineering at the University of Glasgow, Glasgow G12 8QQ. Mr Marshall is with the North of Scotland Hydro-Electric Board, Engineering Department, 16 Rothesay Terrace, Edinburgh EH3 7SE. Mr Findlay is now with Y-ARD Ltd, Charing Cross Tower, Glasgow G2 4PP.

1.0 INTRODUCTION

1.1 History of collaborative research

Collaborative research on various aspects of power systems has been carried out over a period of 13 years between the University of Glasgow and the North of Scotland Hydro-Electric Board and, during the last seven years, the work has concentrated on the study of water turbine control systems. Within a general framework laid down by the Board, the University has formulated a number of specific projects each aimed at improving the understanding of the way in which existing turbine controllers operate or at improving the design with a view to providing controllers for future conventional or pumped-storage hydro generating units. This policy has enabled the University to set each project at an appropriate level for a post-graduate student working for a higher degree while at the same time following the Board's research programme.

It was decided at an early stage of the investigations, to carry out studies at Loch Sloy Power Station on the shore of Loch Lomond some 40 miles north of Glasgow. This was sufficiently close to the University Laboratories to permit parallel studies at the power station on the real plant and in the Laboratories on computer models without significant loss of time in travel between the two locations. The whole scheme was also extremely well documented[1] and this meant that the data necessary for the proposed studies was likely to be readily available. The early work was concerned with the analysis and modelling of the pipeline, tunnel and turbine systems[2] and with improved governor designs[3]. A prototype electronic governor including an additional servomotor was installed and tested[4] which gave much better power response to frequency disturbances when grid-connected .than the original governor, whilst still preserving stability when supplying an isolated load.

Since then, the work has been concerned with improving further still the characteristics of the governors[5] and with developing a turbine controller which is used not only in governing but also in the run-up and loading of the unit. To permit this and future work to proceed unhindered by the need to rebuild the entire experimental equipment in the power station each time a new controller is designed and constructed, a "Controller Testing Facility" has been installed which rationalises the experimental equipment and provides a tool for controller and governor testing. This paper describes this Facility.

1.2 General description of Sloy plant

The experimental equipment is installed on one of the four 32.5 MW hydro generating units at Loch Sloy Power Station. Water for the power station is supplied from a reservoir at a head of about 265 m (varying depending on reservoir level) through a single 2750 m rock tunnel, two 180 m concrete lined tunnels and finally four 360 m steel pipes. A surge shaft is situated at the junction of the rock and concrete tunnels. The turbines are high head Francis machines running at 428.6 rpm producing a nett output of 33.9 MW. Control of the water flow into the turbine is provided by a water control valve consisting of 24 guide vanes arranged around the periphery of the runner. This control valve is positioned by the main servomotors of the controllers through two links and a ring which simultaneously moves all of the guide vanes. There is a main inlet valve on the pipeline side of the control valve which is either fully open or fully closed and since it is capable of closing against full flow, it serves as a means of isolating the turbine from the pipeline, which is completely independent of the control valve.

1.3 Original controller

The original controller consists of an English Electric governor of the mechanical hydraulic dashpot type and a hydraulic "gradual-start" mechanism which controls the turbine during run-up and permits remote control. It has a main servomotor of 43 cm diameter and 35 cm stroke which can exert a force of 20 tonne. The speed sensitive element is a mechanical hydraulic pendulum and great care was taken in its design to avoid the effects of friction. This governor has a temporary droop characteristic and is described in detail in reference 6.

2.0 EXPERIMENTAL EQUIPMENT

2.1 General

Attempts were made to use the original controller in early tests on the plant which sought to identify its characteristics. This approach was found to be unsatisfactory since the injection of test signals was seriously restricted by the need to use a mechanical transducer. In addition, it proved impossible to add new characteristics to this existing system. It was therefore decided to supplement the existing controller on one of the station generating units with an experimental controller and the necessary equipment to permit either of them to be selected for service using a rapid and simple changeover procedure. Since the Loch Sloy station is not designed to run continuously throughout the year, this arrangement had the advantage, in general, of permitting the experimental work to take place during normal breaks in operational service of the unit. From the plant operator's point of view this is most important as the primary purpose of the generator is to provide a public electricity supply. At the same time the experimental work was able to proceed relatively unhindered by the operational requirements of the electricity supply system.

Figure 1 shows the general arrangement of the experimental facilities. A new high-pressure hydraulic servomotor system has been installed on the end of the original servomotor and mechanically connected to it. Electrical connections are made through a selector switch which selects one of the controllers. This switch is mounted on a marshalling cubicle where isolation links are provided to permit the experimental equipment to be entirely disconnected from the power station plant. By this means, work can continue on the experimental equipment when the generator is running and supplying power to the grid, with no risk to the operational plant.

Cables connect the marshalling cubicle and the controller cubicle, in which all signals to and from the power station are connected through signal conditioning equipment, which is permanently installed, to the controllers/governors, which change as the development work proceeds and which are moved between the power station and the University Laboratory. The signals between this conditioning equipment and the controller/governor are all at standard logic system and operational amplifier voltage and impedance levels and are available on plugs or sockets. Thus all controllers operate to a well-defined and unchanging interface, require no special interface equipment (e.g. relays or power amplifiers), and can be rapidly installed in the station. This provides not only the obvious advantage of simplyfing controller design, but also means that controllers can be connected to the station simulation in the University Laboratory.

Final checks on new controllers can be made prior to implementaion on the actual generating unit using the rudimentary simulation facilities provided in the controller cubicle. For these checks, the isolating links can be left open and hence there is no connection to the station apart from a power supply.

Steps have been taken in designing the experimental equipment to ensure that in the event of component failure, the generating unit is shut-down using the same orderly method as that used under similar circumstances for the original controller. An important consideration, however, in the design philosophy of both controllers is that a water turbine, unlike a steam turbine, can withstand for several minutes, the overspeed caused by loss of electrical load with full water flow through the turbine. Thus, even with total controller failure, time exists to close the main water inlet valve (cf. steam emergency stop valve). It is therefore unnecessary to provide parallel redundancy in the controller system.

2.2 Power supply arrangements

The principal items within the controller cubicle for signal conditioning and simulation are shown in Figure 2. The controller is designed so that failure in its essential parts will result in orderly shutdown of the plant. The source of power for the equipment is 240V A.C. derived from one of three sources and all of which are passed through interference filters. During preliminary commissioning, totally isolated from station operational supplies, the test supply is taken from an auxiliary supply. During normal operation, the station common services supply or the generator unit supply is used.

The filtered A.C. supply is used to derive regulated 5v and $\pm$15V D.C. supplies which are used within the signal conditioning equipment and are available together with the 240V A.C. at the controller interface. Each of the supplies is monitored for voltages above and below those at which the remainder of the equipment will operate. In addition a watchdog timer (retriggerable monostable) is used to monitor a pulse train derived from a microprocessor governor or controller if one is present. If all the power supply voltages are within tolerance and if the watchdog timer is operating correctly, a 110V relay is energised and its contacts are used to provide an indication of healthy

system supplies. Power supply failure has the same effect in the station protection as low hydraulic oil pressure (see section 4.0, Protection and Alarms).

The 110V D.C. supply for the cubicle is normally derived from the station battery through an interference filter but during testing can be derived from the A.C. supply.

2.3 Control signals

Various control signals into the cubicle are derived from 110V D.C. signals in the station. Each passes through a selector switch which permits use of the incoming signal, or of a continuous on or continuous off. It then passes through an optical isolator and a driver logic gate and is presented to the controller interface as a logic level signal. The selector switch can be used to disconnect an incoming signal and to simulate it locally for test purposes. The input unit also contains an indicator for each channel and these together with the selector switches permit rapid and simple commissioning of new controllers. Similar switches and indicators are provided on the outputs which take logic signals from the controller and drive relays, also through optical isolators, to give clean contacts for use in station circuits.

2.4 Servo controller

The servo controller normally has a closed loop proportional action and is contained entirely within the interface equipment. This positions the servomotor to the "desired servo position" given by the value of the signal from the governor and the controller. The electro-hydraulic servovalve which controls the servomotor has an integral action and the system thus has zero steady state error. A linear variable differential transformer (LVDT) position transducer connected directly to the servomotor is used to provide a highly robust, reliable and stepless position feedback.

The servo position signal is transmitted to the controller across the interface. Should it be necessary for the controller/governor to exercise direct control over the servovalve, the feedback loop in the servo controller can be broken and "desired servo position" signal would then be used for the different function of driving the electro-hydraulic servovalve directly. Facilities have also been provided to move the servomotor even when there is no 240V A.C. supply by using the 110V battery supply in a "servo local controller" module to drive the servovalve directly. Additionally, mechanical bias within this valve results in the servomotor slowly closing the guide vanes of the water turbine when it receives no electrical signal.

2.5 Frequency transducer

The turbine speed input to the governor is obtained from the frequency of the phasor combination of two generator phase-phase voltages and two currents. The combination is such that 90 degrees lagging, balanced currents will add in phase with the resultant voltage signal and provides a frequency signal even during a zero impedance, three phase fault on the generator terminals. The use of two voltage and two current signals means that there is a measure of redundancy in the system.

The transducer is designed to operate with very low values of generator terminal voltage and will give normal output at and above rated speed of the turbine even with total loss of generator excitation current.

The transducer measures the period of the A.C. signal with a resolution of one part in 20,000 at 50 Hz. It uses a crystal controlled oscillator to give an accurate, sensitive and stable signal with a very fast transient response. The transducer makes available to the controller at the interface the following signals:

a 24 bit digital signal proportional to the period of the A.C. signal;

an analogue signal covering the frequency range 0 to 70 Hz;

an analogue signal covering the range 45 to 55 Hz;

six logic signals which indicate attainment of certain speeds, the values of which can be set by potentiometers in the transducer.

This method of speed measurement is relatively inexpensive and it was chosen to give operational experience of its performance and to avoid the considerable problems which would have been posed by the necessity of fitting an additional tachometer to the generator shaft. The frequency of the terminal voltage and the turbine speed differ only as a result of generator rotor transients at a frequency (approximately 1 Hz) which is attenuated adequately by relatively slow water turbine governors. With faster water turbine governors, this difference could impose a limitation on the use of terminal frequency as a measure of turbine speed. However, this limit has not yet been reached although its effects have been noticed.

2.6 Power and reactive VA transducer

The two wattmeter method for three wire, unbalanced systems is used to give a power signal. The reactive VA signal provided is valid only during balanced conditions but is derived in the same way as that used for station control room indications.

2.7 Analogue simulator

Included in the controller cubicle is a rudimentary turbine simulator used to permit almost complete system checkout in the station environment but without water being supplied to the turbine. This simulation takes as an input signal, either the desired or measured servo position and simulates in a simple but adequate method, the servo system, the turbine and pipeline and the turbine and generator inertia, and gives a voltage proportional to frequency signal as output. This signal is used to drive a voltage controlled oscillator and the

resulting A.C. signal (test frequency) is supplied as input signal to the frequency transducer as an alternative to the normal signal derived from the generator terminals via the voltage and current transformers.

2.8 Monitoring

Monitoring of tests is done using digital data logging with ultra-violet recording as a back up. Analogue voltages from the signal conditioning equipment (eg servo position,frequency or power) and from the controller (e.g. desired servo position) are buffered before output to the monitoring equipment in order to reduce the possibility of faults in the monitoring equipment affecting controller functions. The buffer amplifiers are contained in the controller cubicle.

3.0 EXPERIMENTAL SERVOMOTOR SYSTEM

The experimental controllers use a 207 bar hydraulic servomotor system to position the water control valve of the turbine, the whole system being shown in figures 3 and 4. This new servomotor is fitted to the end of the original 14 bar servomotor and three valves (A, B and C in Figure 3) are fitted to permit one or other of the servomotors to exercise control at any time.

Figure 4 shows the physical arrangement of the two servomotors. At the bottom, the original controller gradual-start mechanism is fitted at the end of the new servomotor which is in turn connected to the original servomotor through the rectangular-section spacer-block. The new pump, reservoir and contactor can be seen at the bottom right.

The 207 bar system shown in Figures 3 and 4, uses a swash plate pump to supply the high pressure oil to two 35 litre hydraulic accumulators which act as a pressure reservoir. Fluctuations in pump pressure are thus prevented from reaching the servomotor and sudden demands for oil, as for example in an emergency closing of the water valve, are taken from the accumulators and not from the pump.

Sufficient energy is stored in the accumulators to enable the water control valve to be fully opened or closed several times without the pump running. This latter feature provides safety in the event of pump or electrical supply failure.

The electropneumatic valve E and non-return valve D are used to isolate the servomotor from the supply and to prevent the accumulators from leaking high pressure oil whilst the generator and pump are shut down. In normal service, high pressure oil is distributed to the ends of the servomotor by an electrohydraulic servovalve F. The flow rate is approximately proportional to current in the coil of the servovalve and the direction of servomotor travel is determined by the direction of the current. Oil from the other end of the servomotor returns to the hydraulic system reservoir.

When the original controller is required for service, valve E is closed, depressurising the experimental servomotor system and valve A is open permitting free exchange of oil between its ends and making it totally inactive. The bypass valve C in the station controller is closed and valve B is opened giving the original servomotor control of the turbine water control valve.

When the experimental controller is in service, the oil supply must still be available to provide lubrication for the original controller which is still mechanically connected to the turbine shaft. Bypass valve C is opened and valve B is closed to disable the original servomotor and prevent the original governor head from exhausting its oil supply through the bypass to drain. Bypass valve A is closed. The A.C. supply to the experimental equipment oil pump motor is derived from the original oil pump motor supply which has to run to provide lubrication. The contactor for the experimental equipment pump is closed only when the electrical changeover switch selects the experimental controller.

To ensure that, in the event of controller failure, it is still possible to close the water control valve and hence render the generating unit safe, an emergency close solenoid dump valve G is fitted. This valve, which corresponds to an equivalent valve in the original controller, is operated by the station protection, venting one end of the distribution spool in the electro-hydraulic servovalve directly to drain. This forces the distribution of oil to the closing side of the servo motor irrespective of any current signal from the controller.

4.0 PROTECTION AND ALARMS

The experimental equipment is fully integrated with the station protection and alarm schemes. Low oil pressure switches on the servomotor hydraulic system (see figure 3), supply alarm and protection signals, the alarm setting being somewhat higher than the trip setting. The alarm is annunciated in the control room and the trip has the same effect as low oil pressure in the original controller hydraulic system. A fall in oil pressure below the trip setting will result in operation of the trip relay which initiates the closing of the turbine main water inlet valve and operation of the emergency close solenoid dump valve D. It also starts battery operated generator and turbine lubrication pumps and after the guide vanes are fully shut, opens the generator circuit breaker.

The power supplies in the experimental equipment are monitored and connections are made to the protection equipment so that power supply failure produces the same trip effect as low oil pressure.

The phasor combination of voltages and currents used by the frequency transducer is monitored and if it is lost, a "governor drive" fail trip takes place. The action is the same as would occur if the drive shaft carrying the turbine speed signal to the original governor was broken. If the drive fail occurs when the generator circuit breaker

is open, the controller closes the guide vanes at the normal shutdown rate. If the circuit breaker is closed, the generator trip relay operates and the sequence of events is as described above.

The experimental controllers control and monitor the turbine run up procedure and if the unit fails to reach certain speeds within preset time limits, the guide vanes are closed at the normal shutdown rate and an alarm is annunciated.

High temperature of the oil in the hydraulic system also provides an alarm which again is annunciated in the control room.

5.0 DESK CONTROLS

The controls normally available to the control engineer on the unit control desk include:

speed reference:	raise/lower
voltage reference:	raise/lower
circuit breaker:	open/close
automatic sequence control:	start/stop

An additional panel has been fitted to the desk to provide necessary controls required by the experimental equipment. The additional controls include:

power:	raise/lower, set value and execute set value
load limiter:	raise/lower
control:	auto unload and stop/reset

The experimental controllers which have been implemented have a speed reference similar to the original controller but also have a servo set point control. Both governors act upon the difference between measured and reference speed. The output of the original governor is the desired servo position. The output of the governing part of the experimental controller is added to the servo set point to give a

desired servo position. If this signal exceeds the load limiter set point, which is set by the desk control or the run up logic, a signal is fed back to the governing part of the controller which reduces its output until the desired servo position equals the limiting value.

With the original controller is use, the method of controlling generator output power when the generator is connected to the grid is to vary the speed reference. Output power responds with the dominant time constant of the governor which is a few minutes. In the experimental controller, the servo set point is varied giving a very rapid and precise control over power because the dominant lag is bypassed. The power raise or lower control is used to adjust the servo set point.

A rudimentary load controller for the generator is provided in the experimental controller which is currently in service. In this, the servo positions corresponding to eight load power values at a given reservoir level are stored in the controller. The "power: set value" control will select one of these power values and operation of the "power: execute set value" control will cause the servo reference to be moved to the corresponding servo position. A load controller is under development which will set the electrical output power directly and which will improve the response rate of changes in reference which is restricted in the current version, by the requirement of not opening the turbine relief valve.

The station auto control scheme provides automatic starting and synchronisation of the generating unit and also automatic disconnection from the grid and stopping; loading and unloading are done when the original controller is in service by the control engineer. When the experimental controller is in use, operation of the "power: execute set value" control at any time after the start control is operated results in fully automatic loading of the set after the end of the starting sequence.

In a future controller it is intended to implement an automatic unloading scheme which will reduce to zero the real power and the reactive VA and then initiate the stop sequence. This function is at present done manually. The "auto unload and stop" control will initiate this and it will be possible to arrest the action at any time before the circuit breaker is opened by operating the "reset" control.

6.0 CONCLUSIONS

The Controller Testing Facility has been successfully installed on an operational hydro electric generating unit and has permitted tests to be carried out with novel controllers in a real operating environment. In earlier work[4], the equipment was used in a series of system splitting tests to demonstrate the suitability of a double derivative governing characteristic for both isolated system operation and when connected to the grid. Testing of this new characteristic on a real system established rather more conclusively than would have simulation tests alone, its ability to control the generating unit.

More recent work using the testing facility has shown the advantage of enhancing the double derivative governor by making it stepwise adaptive[5]. The performance of this adaptive governor has been assessed in simulated isolation tests in which as much as possible of the real system is included without actually splitting the supply system and thereby placing consumer's supplies at risk.

The testing facility and controllers have now been developed to the point at which they are undergoing assessment under normal operational conditions with the generating unit supplying power to the grid.

The test equipment has also permitted frequency response and other tests to be carried out on the Sloy hydraulic system. This has enabled a comprehensive simulation of the power station and one of its turbines to be constructed which was used to check theoretical work on the

characteristics of these systems. The importance to controller design of an accurate simulation of the plant has been found, particularly in relation to the various non-linearities which are frequently not included in conventional controller design. The techniques used in simulating the Sloy unit are applicable to other plant.

The testing facility provides an important method of investigating and confirming the performance of novel governing and control characteristics for new or existing conventional hydro or pumped storage plant. Characteristics giving optimum frequency control performance can thereby be specified for plant to be installed in the future which, in the case of pumped storage plant, would be likely to be rated at about 400MW.

The construction of in situ test equipment has been of considerable benefit to the University in providing a test bed for ongoing work and one which has been readily available over a period of some years. It has accustomed University staff and post-graduate students to the realities of an industrial environment enabling them to design controllers which are not only theoretically satisfactory but which are capable of controlling real plant. It has also stimulated the interest of the power station engineers in the work and their suggestions have assisted in ensuring that the controllers designed would be satisfactory in operational service.

It is also perhaps of some general interest to note that the way in which the collaborative work has been carried out has ideally suited the requirements of a University group where one of the main constraints is to provide projects at a level suitable for post-graduate students who rarely have industrial experience after the award of a first degree. The definition by the Board of broad objectives of its research plan, leaving the University to make detailed proposals, the committment of a Board Engineer, who has taken a very close and

detailed interest in the work and the provision and ready access to the generating unit at Sloy have all contributed to the benefits derived by both the University and the Board from the work.

7.0 ACKNOWLEDGEMENTS

The authors would like to extend their thanks to the many people who have helped to make this project a success.

In particular, thanks are due to the late A.M. Cochran of the North of Scotland Hydro Electric Board and T.R. Foord of Glasgow University with whose encouragement and interest the work was commenced and to G.W. Bryce and to P.W. Agnew who carried out the initial work.

The financial and engineering support received from the North of Scotland Hydro-Electric Board, and the continuing access to operational plant have been central to the work described above and are gratefully acknowledged. Thanks are due to the staff of Sloy/Awe Generation Group, in particular to I. Phillipson and A.L. Grant, without whose co-operation and help the work could not have been carried out.

Scholarships for D.G.E. Findlay, K.H. Aitken and N.F. Grant from the Science Research Council are gratefully acknowledged as was its funding of the associated programme of laboratory work.

Thanks are due to Professor Lamb of the Department of Electronics and Electrical Engineering, University of Glasgow for laboratory facilities.

8.0 REFERENCES

1. KERENSKY G., BEVERLY J.C., CHAPMAN E.J.K., 'The Loch Sloy Hydro-electric Development', Parts I, II and III, Proc. I. Mech. E., 1953, **169**, pp. 205-232.

2. BRYCE G.W., FOORD T.R., MURRAY-SMITH D.J., AGNEW P.W., 'The use of a hybrid computer simulation in the investigation of water turbine governors', Simulation Council Proceedings Series, 1976, **6(1)**, pp. 35-44.

3. AGNEW P.W., BRYCE G.W., 'Optimising turbine operation by electronic governing', Water Power and Dam Construction, 1977, **29(1)**, p. 36.

4. BRYCE G.W., AGNEW P.W., FOORD T.R., WINNING D.J., MARSHALL A.G., 'On-site investigation of electrohydraulic governors for water turbines', Proc. IEE, 1977, **124(2)**, pp.147-153.

5. FINDLAY D.G.E., DAVIE H., FOORD T.R., MARSHALL A.G., WINNING D.J., 'Microprocessor-based adaptive hydro turbine governor', Submitted to IEE, December 1979.

6. DENNIS N.G,, 'Water Turbine Governors', Water Power, 1953, 5, pp.65-191.

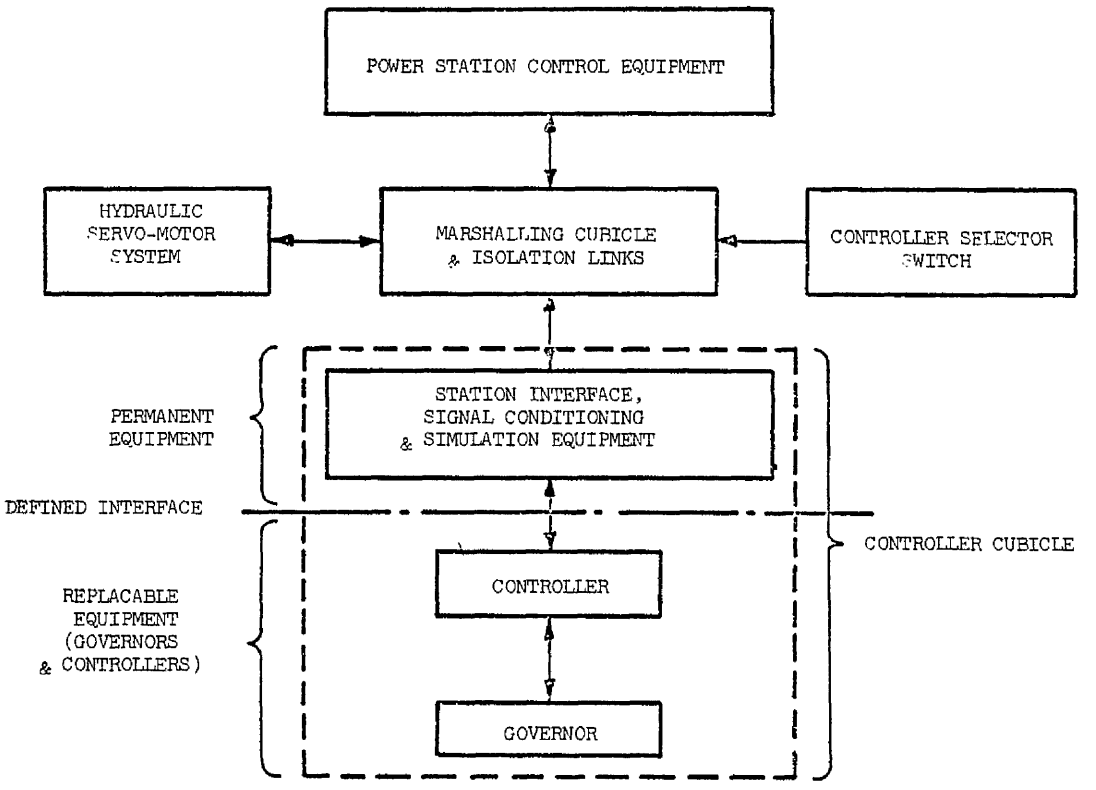


Figure 1 - General Arrangement of Experimental Equipment

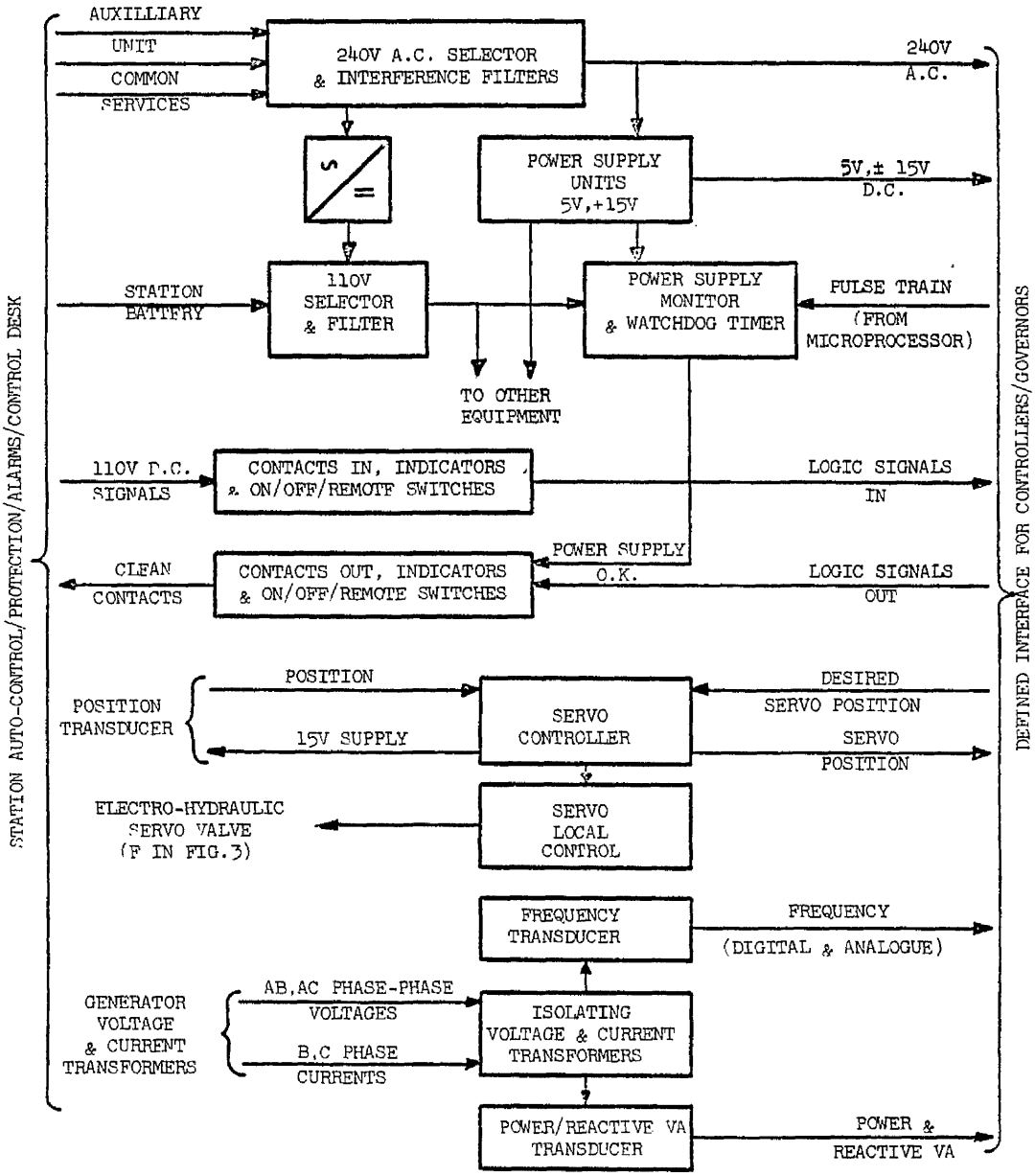


Figure 2 - Principal Elements of Station Interface
(Buffer Amplifiers, Meters for monitoring purposes
and Simulation Facilities are also included)

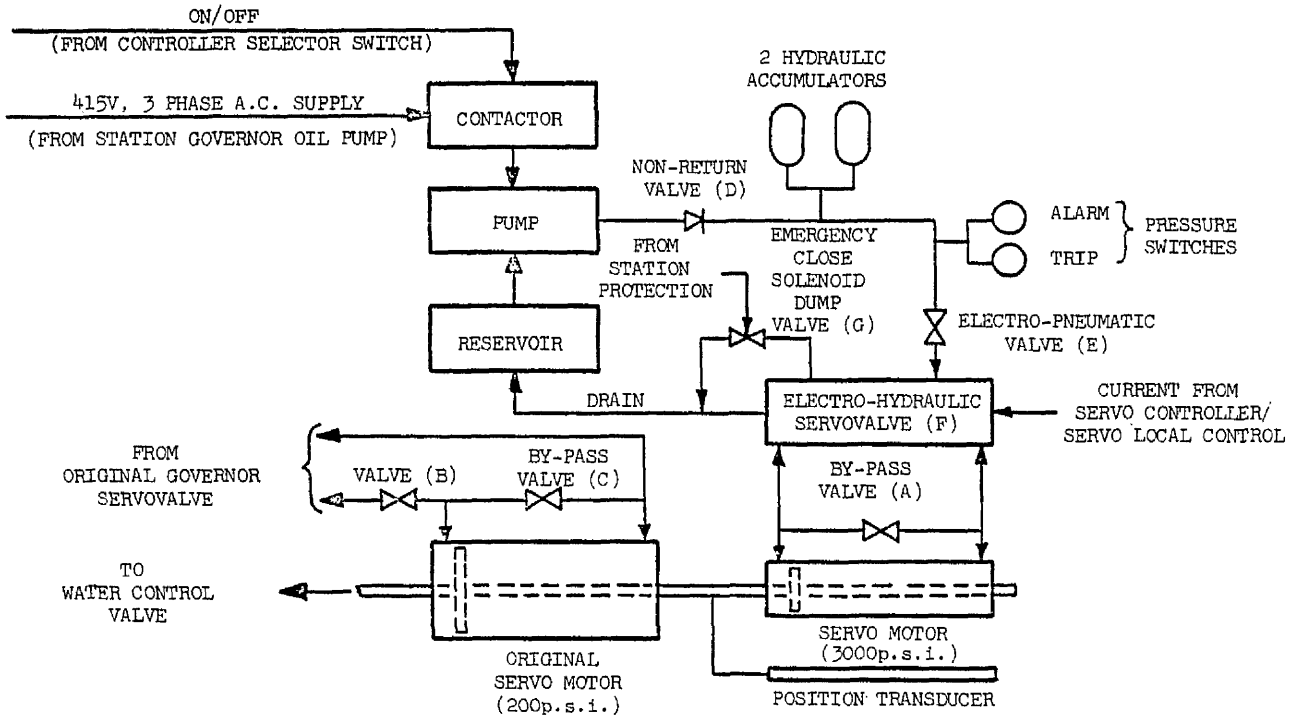


Figure 3 - General Arrangement of Hydraulic Servo-motor System

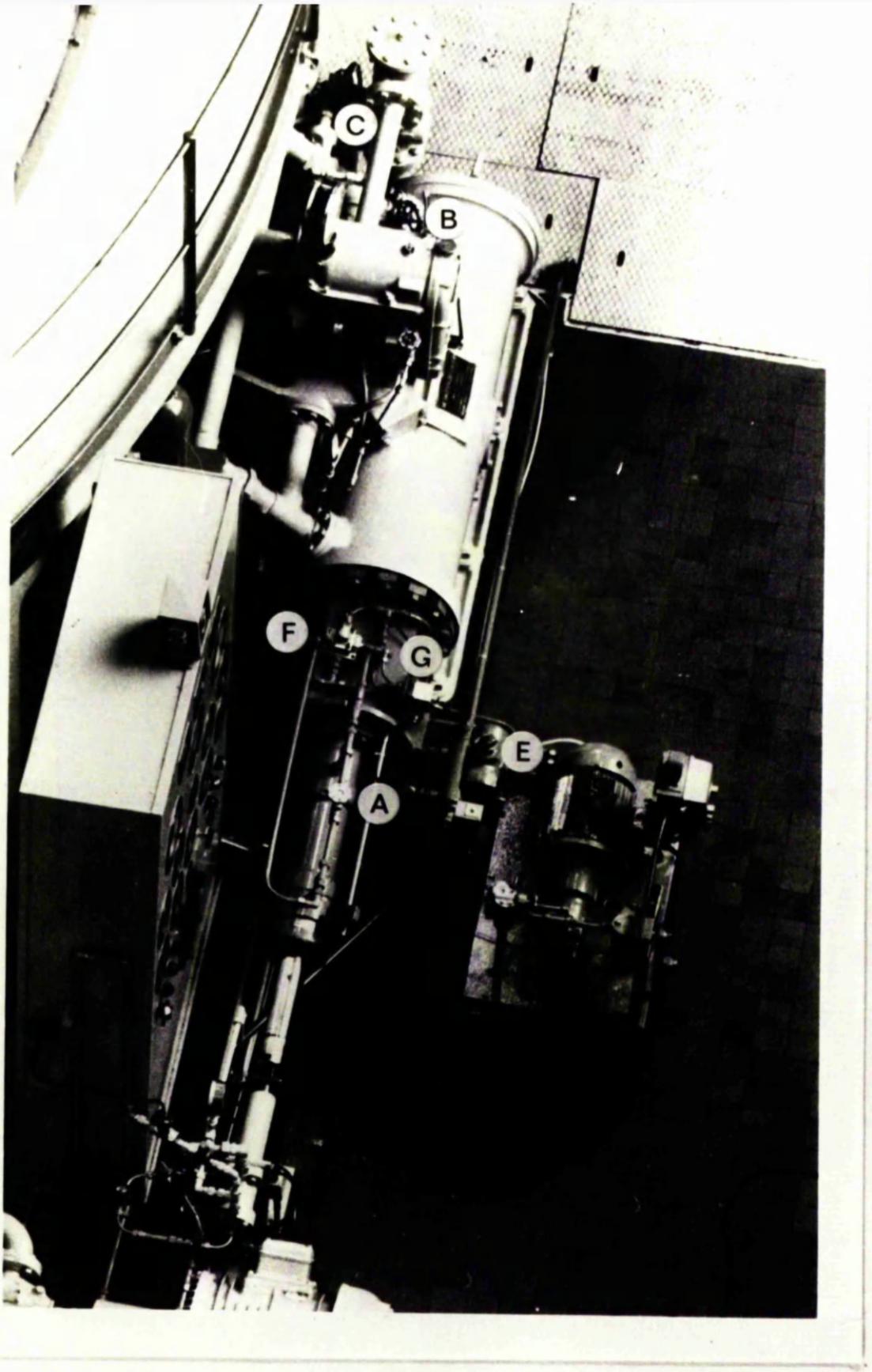


Figure 4. Physical arrangement of servomotor systems.

APPENDIX 2

This appendix contains listings of the GUILDS Model Descriptions used in the simulation of the results presented in the Thesis.

Note: This is an early model written before the
Simulation Language definition was completed.

```
TITLE HYDRO SET - THREE PIPE SECTIONS
PARAMETER AH=1.,AK=2.86,FA=20.07,BK=1.58,FB=1.76,...
CK=2.45,FC=1.14
DIMENSION THA(501),TFA(501)
COMMON/USR1/THA,TFA
INITIAL
ASK LONGEST AND SHORTEST PERIODS/PRL,PRS
ASK INCREMENT IN PERIOD/PINC
ASK LEVEL AND AMPLITUDE/FACT,AMP
IASK NO OF SAMPLES PER CYCLE/NSPC
IASK NO OF PERIODS TO BE SAMPLED/NPFS
IASK NO OF PERIODS BEFORE SAMPLING STARTS/NPBS
SI=PRL/NSPC
TOSC=PRL
TIMLOG=NPBS*TOSC
DYNAMIC
TUFLOW=FACT+AMP*SIN(2*3.14159*T/TOSC)
TUTORQ=TUFLOW*TUHEAD
QA,HA=PIPE(FA,AK,(QB+QC),AH,FACT,AH)
QB,HB=PIPE(FB,BK,0.,HA,0.,AH)
QC,TUHEAD=PIPE(FC,CK,TUFLOW,HA,FACT,AH)
$       IF(T.GT.0.0) GOTO 40$
$       NSI=0.5+SI/H$
$       NLOG=0.5+TIMLOG/H$
$       WRITE(ITOUT,30) NSI,NLOG$
$30     FORMAT(2I6)$
$       K=0$
$40     CONTINUE$
$       IF(M.EQ.1) K=K+1$
$       IF(K.NE.NLOG) GOTO 50$
$       NLOG=NLOG+NSI$
$       J=J+1$
$       THA(J)=TUHEAD$
$       TFA(J)=TUFLOW$
$       IF(J.NE.NSPC*NPFS) GOTO 50$
$       TIME=FINTIM$
$       J=0$
$50     CONTINUE$
```

```
TERMINAL
        IF(KRT.NE.0) GOTO 20
        CALL ASSIGN(1,'FREQRSP.DAT',11)
        WRITE(1,90) PRL,PRS,PINC,FACT,AMP,NSPC,NPBS,NPFS,H,TIME
20      DO 30 L=1,NSPC*NPFS
        ARG=2*3.14159*(L-1)/NSPC
        COSA=COS(ARG)
        SINA=SIN(ARG)
        ATH=ATH+THA(L)*COSA
        BTH=BTH+THA(L)*SINA
        ATF=ATF+TFA(L)*COSA
        BTF=BTF+TFA(L)*SINA
30      CONTINUE
        PP=2./NSPC
        ATH=ATH*PP/NPFS
        BTH=BTH*PP/NPFS
        ATF=ATF*PP/NPFS
        BTF=BTF*PP/NPFS
        TH=SQRT(ATH*ATH+BTH*BTH)
        TF=SQRT(ATF*ATF+BTF*BTF)
        HYZ=TH/TF
        WRITE(1,100) TOSC,TH,TF,HYZ
        WRITE(ITOUT,105)
        ATH=0.
        BTH=0.
        ATF=0.
        BTF=0.
        TOSC=TOSC-PINC
        IF(TOSC.LT.PRS) LR=0
        SI=TOSC/NSPC
        TIMLOG=NPBS*TOSC
        IF(LR.EQ.0) CALL CLOSE(1)
90      FORMAT(1X,'LONGEST PERIOD = ',F11.5,' SHORTEST PERIOD = ',F11.5,/,
     1  1X,'INCREMENTAL PERIOD = ',F11.5,/,
     1  1X,'OPERATING POINT = ',F11.5,
     1  ' AMPLITUDE OF DISTURBANCE = ',F11.5,/,
     1  1X,'NO OF SAMPLES PER CYCLE = ',I4,
     1  ' NO OF CYCLES BEFORE SAMPLING = ',I4,/,
     1  1X,'NO OF CYCLES TO BE SAMPLED = ',I4,/,
     1  1X,'INTEGRATION INTERVAL = ',F11.5,' FINISH TIME = ',F11.5,/)
100     FORMAT(4E13.6)
105     FORMAT(1X,'TASK ACTIVE - PLEASE DO NOT USE THIS TERMINAL. K.A.')
END
```

Note: This is an early model written before the
Simulation Language definition was completed.

```
TITLE HYDRO SET - TPS., SS. AND RESEVOIR.
PARAMETER AH=1.,AK=2.86,FA=20.07,BK=1.58,FB=1.76,...
CK=2.45,FC=1.14,SK=.0011,FRS=1.22
DIMENSION THA(501),TFA(501)
COMMON/USR1/THA,TFA
INITIAL
ASK LONGEST AND SHORTEST PERIODS/PRL,PRS
ASK INCREMENT IN PERIOD/PINC
ASK LEVEL AND AMPLITUDE/FACT,AMP
IASK NO OF SAMPLES PER CYCLE/NSPC
IASK NO OF PERIODS TO BE SAMPLED/NPFS
IASK NO OF PERIODS BEFORE SAMPLING STARTS/NPBS
SI=PRL/NSPC
TOSC=PRL
TIMLOG=NPBS*TOSC
DYNAMIC
TUFLOW=FACT+AMP*SIN(2*3.14159*T/TOSC)
TUTORQ=TUFLOW*TUHEAD
DHS=SK*(QRS-QA)
HS=INTGRL(AH,DHS)
DQRS=FRS*(AH-HS)
QRS=INTGRL(FACT,DQRS)
QA,HA=PIPE(FA,AK,(QB+QC),HS,FACT,AH)
QB,HB=PIPE(FB,BK,0.,HA,0.,AH)
QC,TUHEAD=PIPE(FC,CK,TUFLOW,HA,FACT,AH)
$       IF(T.GT.0.0) GOTO 40$
$       NSI=0.5+SI/H$
$       NLOG=0.5+TIMLOG/H$
$       WRITE(ITOUT,30) NSI,NLOG$
$30     FORMAT(2I6)$
$       K=0$
$40     CONTINUE$
$       IF(M.EQ.1) K=K+1$
$       IF(K.NE.NLOG) GOTO 50$
$       NLOG=NLOG+NSI$
$       J=J+1$
$       THA(J)=TUHEAD$
$       TFA(J)=TUFLOW$
$       IF(J.NE.NSPC*NPFS) GOTO 50$
$       TIME=FINTIM$
$       J=0$
$50     CONTINUE$
```

```
TERMINAL
        IF(KRT.NE.0) GOTO 20
        CALL ASSIGN(1,'FREQRSP.DAT',11)
        WRITE(1,90) PRL,PRS,PINC,FACT,AMP,NSPC,NPBS,NPFS,H,TIME
20      DO 30 L=1,NSPC*NPFS
        ARG=2*3.14159*(L-1)/NSPC
        COSA=COS(ARG)
        SINA=SIN(ARG)
        ATH=ATH+THA(L)*COSA
        BTH=BTH+THA(L)*SINA
        ATF=ATF+TFA(L)*COSA
        BTF=BTF+TFA(L)*SINA
30      CONTINUE
        PP=2./NSPC
        ATH=ATH*PP/NPFS
        BTH=BTH*PP/NPFS
        ATF=ATF*PP/NPFS
        BTF=BTF*PP/NPFS
        TH=SQRT(ATH*ATH+BTH*BTH)
        TF=SQRT(ATF*ATF+BTF*BTF)
        HYZ=TH/TF
        WRITE(1,100) TOSC,TH,TF,HYZ
        WRITE(ITOUT,105)
        ATH=0.
        BTH=0.
        ATF=0.
        BTF=0.
        TOSC=TOSC-PINC
        IF(TOSC.LT.PRS) LR=0
        SI=TOSC/NSPC
        TIMLOG=NPBS*TOSC
        IF(LR.EQ.0) CALL CLOSE(1)
90      FORMAT(1X,'LONGEST PERIOD = ',F11.5,' SHORTEST PERIOD = ',F11.5,/,
     1  1X,'INCREMENTAL PERIOD = ',F11.5,/,
     1  1X,'OPERATING POINT = ',F11.5,
     1  ' AMPLITUDE OF DISTURBANCE = ',F11.5,/,
     1  1X,'NO OF SAMPLES PER CYCLE = ',I4,
     1  ' NO OF CYCLES BEFORE SAMPLING = ',I4,/,
     1  1X,'NO OF CYCLES TO BE SAMPLED = ',I4,/,
     1  1X,'INTEGRATION INTERVAL = ',F11.5,' FINISH TIME = ',F11.5,/)
100     FORMAT(4E13.6)
105     FORMAT(1X,'TASK ACTIVE - PLEASE DO NOT USE THIS TERMINAL. K.A.')
END
```

```
TITLE HYDRO SET - RUN UP SIMULATION
*
* THIS IS THE STANDARD HYDRO GENERATOR MODEL
* MODIFIED TO SIMULATE THE RUN UP CHARACTERISTIC.
* THE RELIEF VALVE HAS BEEN OMITTED FROM THIS MODEL.
* VERSION 8
*
        REAL KA,KB,KC,K1,K2,K12,K22,LL1,LL2
*
PARAMETER KA=2.71,FA=20.24,KB=1.46,FB=1.90,...
KC=2.41,FC=1.15,...
BP=0.03,BT=0.25,TD=16.0,TP=0.3
UPDATE P1=-.007,P2=.007,HR=1.,TA=7.,T3=0.2,T4=0.2
UPDATE RC=-.25,RO=.03,K1=3.0,K2=2.3,TL=33.3
UPDATE GS=5.,TS=0.1
UPDATE Q1=0.875,Q2=0.9,Q4=1.095,Q3=1.125,Q5=1.095,Q6=0.955,Q7=0.98
UPDATE FD=1.23,AS=0.0011,LL1=0.28,LL2=0.21,RL1=0.036,RL2=0.016
UPDATE QL=0.088,F1=0.11,E1=1.584,E2=0.584,YNL=0.12,FPB=0.6
UPDATE A1=0.0677,A2=0.0805,A3=2.055,A4=-1.199,A5=1.228,A6=-2.942
*
INITIAL
*
        QIC=0.
        YIC=0.
        YSIC=0.
        FL=0.
        FS=1.
*
DYNAMIC
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        YL1=RAMP1(5.,RL1,YSIC,LL1)
        YL2=RAMP2(F-FPB,-RL2,LL1,LL2)
        YLL=SWIN(YL1-LL1,YL1,YL2)
        YS=YLL
        YLLA=1.1*YLL-0.05
        YSA=YLLA
        YG=LIMIT(-YS,YLL-YS,YG)
        FT=STPLIM(Q1,Q2,Q3,Q4,F)                :FREQUENCY TRANSDUCER
        SD=ANDHYS(0.,FT-Q6,FT-Q7)               :DERIVATIVE SUPPRESSION
        K12=K1*SD
        K22=K2*SD
        FTD=LIMIT(0.,Q5,FT)                     :DERIVATIVE FREQUENCY LIMIT
        Z1=(FS-FTD-X1)/T3
        X1=INTGRL(0.,Z1)
        Z2=(Z1-X2)/T4
        X2=INTGRL(0.,Z2)
        X3=(K12*Z1+K22*Z2-FT+FS)/BP
        ZG=(X3-YG)/TL
        YG=INTGRL(0.,ZG)                        :GOVERNOR OUTPUT
        YD=YG+YS                                :DESIRED SERVO POSITION
        YD=LIMIT(0.,1.,YD)
        YD=1.1*YD-0.05                          :SCALING
```

```
*
* SERVO AND WATER CONTROL VALVE
*
        Y=LIMIT(0.,1.,Y)
        YE=(YD-Y)
        X4=GS*FOLAG(0.,TS,YE)
        Z=LIMIT(RC,RO,X4)
        Y=INTGRL(YIC,Z)
        YC=HSTRSS(YIC,P1.P2;Y)
        YC2=YC*YC
        YC3=YC*YC2
        AE1=A5*YC+A6*YC2
        AE2=A1+A2*YC+A3*YC2+A4*YC3
        AE=SWIN(YC-YNL,AE1,AE2)
*
* TURBINE AND GENERATOR
*
        QT=AE*(E1*HT-E2*F**2)**.5
        IF(F.GT.0.7) FW=QT*HT/F
        FE1=1.388*QT*HT-QL
        F2=F1*F
        FE=(1.+F1)*FW-F2
        FE=SWIN(F-0.8,FE1,FE)
        ZF=(FE-FL)/TA
        ZF=SWIN(FE1,0.,ZF)
        F=INTGRL(0.,ZF)
*
* PIPELINE
*
        QP=QT
        DHS=AS*(QD-QA)
        HS=INTGRL(HR,DHS)
        DQD=FD*(HR-HS)
        QD=INTGRL(QIC,DQD)
        QA,HA=PIPE(FA,KA,(QB+QC),HS,QIC,HR)
        QB,HB=PIPE(FB,KB,0.,HA,0.,HR)
        QC,HT=PIPE(FC,KC,QP,HA,QIC,HR)
END
```

```
TITLE HYDRO SET - LOAD REJECTION SIMULATION
*
* THIS IS THE STANDARD HYDRO GENERATOR MODEL
* MODIFIED TO SIMULATE A LOAD REJECTION,
* WITH A CHOICE OF DD OR TD GOVERNOR.
* AN OPTION WHICH PERMITS THE SERVO SET
* POINT AND LOAD LIMIT RAMPS TO BE DISABLED
* HAS ALSO BEEN INCLUDED, AND A RATE LIMIT ON
* DESIRED SERVO POSITION HAS BEEN IMPLEMENTED.
* VERSION 9
*
        REAL KA,KB,KC,K1,K2,LL2
*
PARAMETER KA=2.71,FA=20.24,KB=1.46,FB=1.90,...
KC=2.41,FC=1.15,...
BP=0.03,BT=0.25,TD=16.0,TP=0.3
UPDATE P1=-.007,P2=.007,HR=1.,TA=7.
UPDATE RC=-.25,RO=.03,FS=1.,TS=0.1
UPDATE GS=5.,TR=0.2,RT=0.03,RU=0.01,RM=0.025
UPDATE Q1=0.875,Q2=0.9,Q4=1.095,Q3=2.,Q5=1.095
UPDATE FD=1.23,AS=0.0011,LL2=0.21,RL2=0.022,YNL=0.12
UPDATE F1=0.082,E1=1.584,E2=0.584
UPDATE A1=0.0294,A2=0.1387,A3=2.0366,A4=-1.2011,A5=0.6462,A6=-0.2941
*
INITIAL
*
ASK INITIAL ELECTRICAL LOAD (MW)/PLMW
IASK TD (0) OR DD (1) GOVERNOR/IGOV
IASK MW AND LL RAMPS ACTIVE (0) OR DISABLED (1)/IRAMP
*
* SET UP GOVERNOR PARAMETERS
*
        T3=0.3
        T4=0.3
        K1=3.0
        K2=2.3
        TL=33.3
        IF(IGOV.EQ.1) GOTO 200
        T3=1.018
        K1=14.99
        K2=0.
        TL=158.29
200     CONTINUE
*
* CALCULATE INITIAL CONDITIONS
        PLIC=PLMW/32.5
        FMIC=PLIC
        FWIC=(FMIC+F1)/(1.+F1)
        QIC=FWIC/HR
        AEIC=QIC/SQRT(E1*HR-E2)
```

```
*
* ITERATION TO FIND INVERSE OF NON-LINEAR FUNCTION
*
        YIC=0.
        DO 210 I=1,7
        DY=1./10.**I
205     YIC=YIC+DY
        Y2=YIC*YIC
        Y3=Y2*YIC
        AEN=A1+A2*YIC+A3*Y2+A4*Y3
        IF(AEN.LE.AEIC) GOTO 205
        YIC=YIC-DY
210     CONTINUE
*
        YDIC=YIC
        IF(YIC.GE.1.0) YDIC=1.05
        YSIC=(YDIC+0.05)/1.1
*
DYNAMIC
*
* ELECTRICAL LOAD
*
        PL=PLIC-PLIC*STEP(5.)
        FL=PL/F
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        YS=RAMP1(5.,-RL2,YSIC,LL2)
        YS=SWIN(IRAMP-1,YS,YSIC)
        YSA=1.1*YS-0.05
        YLL=RAMP1(5.,-RL2,1.,LL2)
        YLL=SWIN(IRAMP-1,YLL,1.)
        YLLA=1.1*YLL-0.05
        YG=LIMIT(-YS,YLL-YS,YG)
        FT=STPLIM(Q1,Q2,Q3,Q4,F)                :FREQUENCY TRANSDUCER
        FTD=LIMIT(0.,Q5,FT)                     :DERIVATIVE FREQUENCY TERM
        Z1=(FS-FTD-X1)/T3
        X1=INTGRL(0.,Z1)
        Z2=(Z1-X2)/T4
        X2=INTGRL(0.,Z2)
        X3=(K1*Z1+K2*Z2-FT+FS)/BP
        ZG=(X3-YG)/TL
        YG=INTGRL(0.,ZG)                        :GOVERNOR OUTPUT
        YD=YG+YS
        YD=LIMIT(0.,1.,YD)                      :DESIRED SERVO POSITION
        TIM=TIME
        YD=RATLIM(TIM,RO,RC,YD)
        YD=1.1*YD-0.05                          :OUTPUT SCALING
```

```
*
* SERVO AND WATER CONTROL VALVE
*
        Y=LIMIT(0.,1.,Y)
        YE=(YD-Y)
        X4=GS*FOLAG(0.,TS,YE)
        Z=LIMIT(RC,RO,X4)
        Y=INTGRL(YIC,Z)
        ZY=DERLAG(YIC,0.,0.1,Y)
        YC=HSTRSS(YIC,P1,P2,Y)
        YC2=YC*YC
        YC3=YC*YC2
        AE1=A5*YC+A6*YC2
        AE2=A1+A2*YC+A3*YC2+A4*YC3
        AE=SWIN(YC-YNL,AE1,AE2)
*
* RELIEF VALVE
*
        ZL=SWIN(ZY,-ZY-RT,-ZY-RU)
        ZL=LIMIT(-RM,1.,ZL)
        YL=INTGRL(0.,ZL)
        YL=LIMIT(0.,1.,YL)
        YR=FOLAG(0.,TR,YL)
        QR=1.22*YR*SQRT(HT)
*
* TURBINE AND GENERATOR
*
        QT=AE*(E1*HT-E2*F**2)**.5
        FW=QT*HT/F
        FE=(1.+F1)*FW-F1*F
        ZF=(FE-FL)/TA
        F=INTGRL(1.,ZF)
*
* PIPELINE
*
        QP=QT+QR
        DHS=AS*(QD-QA)
        HS=INTGRL(HR,DHS)
        DQD=FD*(HR-HS)
        QD=INTGRL(QIC,DQD)
        QA,HA=PIPE(FA,KA,(QB+QC),HS,QIC,HR)
        QB,HB=PIPE(FB,KB,0.,HA,0.,HR)
        QC,HT=PIPE(FC,KC,QP,HA,QIC,HR)
END
```

```
TITLE HYDRO SET - SIMULATED ISOLATED LOAD
*
* THIS IS THE STANDARD HYDRO GENERATOR MODEL
* USED FOR MODELLING SIMULATED ISOLATED LOAD
* STEP TESTS AND LIMIT CYCLING. THE RELIEF
* VALVE HAS BEEN OMITTED FROM THIS MODEL.
*
* THE GOVERNOR EQUATIONS USE EULER, NOT RK4
*
* VERSION 5
*
        REAL KA,KB,KC,K1,K2,KN
*
PARAMETER KA=2.71,FA=20.24,KB=1.46,FB=1.90,...
KC=2.41,FC=1.15,...
BT=0.25,TD=16.0,TP=0.3
UPDATE P1=-.007,P2=.007,HR=1.,TA=7.,T3=0.2,T4=0.2
UPDATE RC=-.25,RO=.03,K1=3.0,K2=2.3,TL=33.3
UPDATE GS=5.,TS=0.1,BP=0.03,FS=1.
UPDATE Q1=0.875,Q2=0.9,Q4=1.095,Q3=1.125,Q5=1.095
UPDATE FD=1.23,AS=0.0011,YNL=0.12,KN=1.
UPDATE F1=0.11,E1=1.584,E2=0.584
UPDATE A1=0.0677,A2=0.0805,A3=2.055,A4=-1.199,A5=1.228,A6=-2.942
*
INITIAL
*
ASK OPERATING POINT,STEP SIZE/PLMW,STP
ASK SAMPLING DELAY TIME/TDEL
IASK EULER FROM RK4 - NO(0), YES(1)/IEUL
*
* CALCULATE INITIAL CONDITIONS
*
        PLIC=PLMW/32.5
        FMIC=PLIC
        FWIC=(FMIC+0.11)/(1.+0.11)
        QIC=FWIC/HR
        AEIC=QIC/SQRT(E1*HR-E2)
*
* ITERATION TO FIND INVERSE OF NON-LINEAR FUNCTION
*
        YIC=0.
        DO 210 I=1,7
        DY=1./10.**I
205     YIC=YIC+DY
        Y2=YIC*YIC
        Y3=Y2*YIC
        AEN=A1+A2*YIC+A3*Y2+A4*Y3
        IF(AEN.LE.AEIC) GOTO 205
        YIC=YIC-DY
210     CONTINUE
*
        YLL=1.
        YDIC=YIC
        YS=YIC          :INCORRECT IC TO START LIMIT CYCLING
        IGOV=0
```

```
*
DYNAMIC
*
        ISG=0.5+0.1/H
* SIMULATED ELECTRICAL LOAD
*
        FL=PLIC*(1+KN*(F-FS))+STP*DISTRB(0.,1.)
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        FDEL=DELAY(TDEL,F)
        FGOV=(F+FDEL)/2.
        IF(IGOV.EQ.ISG) IGOV=0
        IF(IEUL.EQ.1.AND.M.NE.1) GOTO 20
        IF(IEUL.EQ.1.AND.IGOV.NE.0) GOTO 15
        YG=LIMIT(-YS,YLL-YS,YG)
        FT=STPLIM(Q1,Q2,Q3,Q4,F)
        FTD=LIMIT(0.,Q5,FT)
        Z1=(FS-FTD-X1)/T3
        X1=INTGRL(0.,Z1)
        Z2=(Z1-X2)/T4
        X2=INTGRL(0.,Z2)
        X3=(K1*Z1+K2*Z2-FT+FS)/BP
        ZG=(X3-YG)/TL
        YG=INTGRL(0.,ZG)
        YD=YG+YS
        YD=LIMIT(0.,1.,YD)
        TIM=TIME
        YD=RATLIM(TIM,RO,RC,YD)
        YD=1.1*YD-0.05
15      IGOV=IGOV+1
20      CONTINUE
*
* SERVO AND WATER CONTROL VALVE
*
        Y=LIMIT(0.,1.,Y)
        YE=(YD-Y)
        X4=GS*FOLAG(0.,TS,YE)
        Z=LIMIT(RC,RO,X4)
        Y=INTGRL(YIC,Z)
        YC=HSTRSS(YIC,P1,P2,Y)
        YC2=YC*YC
        YC3=YC*YC2
        AE1=A5*YC+A6*YC2
        AE2=A1+A2*YC+A3*YC2+A4*YC3
        AE=SWIN(YC-YNL,AE1,AE2)
*
* TURBINE AND GENERATOR
*
        QT=AE*(E1*HT-E2*1.**2)**.5
        FW=QT*HT/1.
        FE=(1.+F1)*FW-F1*1.
        PE=FE*F
        ZF=(FE-FL)/TA
        F=INTGRL(1.,ZF)
```

```
*
* PIPELINE
*
        QP=QT
        DHS=AS*(QD-QA)
        HS=INTGRL(HR,DHS)
        DQD=FD*(HR-HS)
        QD=INTGRL(QIC,DQD)
        QA,HA=PIPE(FA,KA,(QB+QC),HS,QIC,HR)
        QB,HB=PIPE(FB,KB,0.,HA,0.,HR)
        QC,HT=PIPE(FC,KC,QP,HA,QIC,HR)
END
```

```
TITLE HYDRO SET - ISOLATED LOAD SIMULATION
*
* THIS IS THE STANDARD HYDRO GENERATOR MODEL
* USED FOR SIMULATING ISOLATED LOAD STEP TESTS
* AND LIMIT CYCLING. THE RELIEF VALVE
* HAS BEEN OMITTED FROM THIS MODEL.
*
VERSION 1
        REAL KA,KB,KC,K1,K2
*
PARAMETER KA=2.71,FA=20.24,KB=1.46,FB=1.90,...
KC=2.41,FC=1.15,...
BT=0.25,TD=16.0,TP=0.3
UPDATE P1=-.007,P2=.007,HR=1.,TA=7.,T3=0.2,T4=0.2
UPDATE RC=-.25,RO=.03,K1=3.0,K2=2.3,TL=33.3
UPDATE GS=5.,TS=0.1,BP=0.03
UPDATE Q1=0.875,Q2=0.9,Q4=1.095,Q3=1.125,Q5=1.095
UPDATE FD=1.23,AS=0.0011,YNL=0.12
UPDATE F1=0.11,E1=1.584,E2=0.584
UPDATE A1=0.0677,A2=0.0805,A3=2.055,A4=-1.199,A5=1.228,A6=-2.942
*
INITIAL
*
ASK OPERATING POINT,STEP SIZE/PLMW,STP
*
* CALCULATE INITIAL CONDITIONS
        PLIC=PLMW/32.5
        FMIC=PLIC
        FWIC=(FMIC+0.11)/(1.+0.11)
        QIC=FWIC/HR
        AEIC=QIC/SQRT(E1*HR-E2)
*
* ITERATION TO FIND INVERSE OF NON-LINEAR FUNCTION
*
        YIC=0.
        DO 210 I=1,7
        DY=1./10.**I
205     YIC=YIC+DY
        Y2=YIC*YIC
        Y3=Y2*YIC
        AEN=A1+A2*YIC+A3*Y2+A4*Y3
        IF(AEN.LE.AEIC) GOTO 205
        YIC=YIC-DY
210     CONTINUE
*
        FS=1.
        YLL=1.
        YDIC=YIC
        YS=YIC
*
DYNAMIC
*
* ELECTRICAL LOAD
*
        PL=PLIC+STP*DISTRB(0.,1.)
        FL=PL/F
```

```
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        YG=LIMIT(-YS,YLL-YS,YG)
        FT=STPLIM(Q1,Q2,Q3,Q4,F)
        FTD=LIMIT(0.,Q5,FT)
        Z1=(FS-FTD-X1)/T3
        X1=INTGRL(0.,Z1)
        Z2=(Z1-X2)/T4
        X2=INTGRL(0.,Z2)
        X3=(K1*Z1+K2*Z2-FT+FS)/BP
        ZG=(X3-YG)/TL
        YG=INTGRL(0.,ZG)
        YD=YG+YS
        YD=LIMIT(0.,1.,YD)
        TIM=TIME
        YD=RATLIM(TIM,RO,RC,YD)
        YD=1.1*YD-0.05
*
* SERVO AND WATER CONTROL VALVE
*
        Y=LIMIT(0.,1.,Y)
        YE=(YD-Y)
        X4=GS*FOLAG(0.,TS,YE)
        Z=LIMIT(RC,RO,X4)
        Y=INTGRL(YIC,Z)
        YC=HSTRSS(YIC,P1,P2,Y)
        YC2=YC*YC
        YC3=YC*YC2
        AE1=A5*YC+A6*YC2
        AE2=A1+A2*YC+A3*YC2+A4*YC3
        AE=SWIN(YC-YNL,AE1,AE2)
*
* TURBINE AND GENERATOR
*
        QT=AE*(E1*HT-E2*F**2)**.5
        FW=QT*HT/F
        FE=(1.+F1)*FW-F1*F
        ZF=(FE-FL)/TA
        F=INTGRL(1.,ZF)
*
* PIPELINE
*
        QP=QT
        DHS=AS*(QD-QA)
        HS=INTGRL(HR,DHS)
        DQD=FD*(HR-HS)
        QD=INTGRL(QIC,DQD)
        QA,HA=PIPE(FA,KA,(QB+QC),HS,QIC,HR)
        QB,HB=PIPE(FB,KB,0.,HA,0.,HR)
        QC,HT=PIPE(FC,KC,QP,HA,QIC,HR)
END
```

```
TITLE HYDRO SET - SIMULATED ISOLATED LOAD
*
* THIS IS THE STANDARD HYDRO GENERATOR MODEL
* USED FOR MODELLING SIMULATED ISOLATED LOAD
* STEP TESTS AND LIMIT CYCLING. THE RELIEF
* VALVE HAS BEEN OMITTED FROM THIS MODEL.
*
* A SECTION HAS BEEN INCLUDED TO PERMIT
* THE USE OF SITE DATA FOR CERTAIN SIGNALS.
*
* THE GOVERNOR EQUATIONS USE EULER, NOT RK4.
*
* VERSION 2
*
        DIMENSION SVAR(6)
        REAL KA,KB,KC,K1,K2,KN
*
PARAMETER KA=2.71,FA=20.24,KB=1.46,FB=1.90,...
KC=2.41,FC=1.15,...
BT=0.25,TD=16.0,TP=0.3
UPDATE P1=-.007,P2=.007,HR=1.,TA=7.,T3=0.2,T4=0.2
UPDATE RC=-.25,RO=.03,K1=3.0,K2=2.3,TL=33.3
UPDATE GS=5.,TS=0.1,BP=0.03,FS=1.
UPDATE Q1=0.875,Q2=0.9,Q4=1.095,Q3=1.125,Q5=1.095
UPDATE FD=1.23,AS=0.0011,YNL=0.12,KN=1.
UPDATE F1=0.11,E1=1.584,E2=0.584
UPDATE A1=0.0677,A2=0.0805,A3=2.055,A4=-1.199,A5=1.228,A6=-2.942
*
INITIAL
*
ASK OPERATING POINT,STEP SIZE,STEP TIME/PLMW,STP,STIM
ASK SAMPLING DELAY TIME/TDEL
IASK USE SITE DATA? - NO(0), MW(1), SP(2), FR(3)/ISITE
IASK EULER FROM RK4 - NO(0), YES(1)/IEUL
*
* CALCULATE INITIAL CONDITIONS
*
        PLIC=PLMW/32.5
        FMIC=PLIC
        FWIC=(FMIC+0.11)/(1.+0.11)
        QIC=FWIC/HR
        AEIC=QIC/SQRT(E1*HR-E2)
*
* ITERATION TO FIND INVERSE OF NON-LINEAR FUNCTION
*
        YIC=0.
        DO 210 I=1,7
        DY=1./10.**I
205     YIC=YIC+DY
        Y2=YIC*YIC
        Y3=Y2*YIC
        AEN=A1+A2*YIC+A3*Y2+A4*Y3
        IF(AEN.LE.AEIC) GOTO 205
        YIC=YIC-DY
210     CONTINUE
*
        YLL=1.
        YDIC=YIC
        YS=(YIC+0.05)/1.1
```

```
*
DYNAMIC
*
* READ SITE DATA FILE IF REQUIRED
*
        ISG=0.5+0.1/H
        IF(ISITE.EQ.0.OR.M.NE.1) GOTO 10
        IF(IFLAG.EQ.ISG) IFLAG=0
        IF(IFLAG.EQ.0) CALL SITE(SVAR)
        IFLAG=IFLAG+1
        FE=SVAR(2)
        IF(ISITE.EQ.2) Y=SVAR(3)
        IF(ISITE.EQ.3) F=SVAR(4)+1.
10      CONTINUE
*
* SIMULATED ELECTRICAL LOAD
*
        FL=PLIC*(1+KN*(F-FS))+STP*STEP(STIM)
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        FDEL=DELAY(TDEL,F)
        FGOV=(F+FDEL)/2.
        IF(IGOV.EQ.ISG) IGOV=0
        IF(IEUL.EQ.1.AND.M.NE.1) GOTO 20
        IF(IEUL.EQ.1.AND.IGOV.NE.0) GOTO 15
        YG=LIMIT(-YS,YLL-YS,YG)
        FT=STPLIM(Q1,Q2,Q3,Q4,FGOV)
        FTD=LIMIT(0.,Q5,FT)
        Z1=(FS-FTD-X1)/T3
        X1=INTGRL(0.,Z1)
        Z2=(Z1-X2)/T4
        X2=INTGRL(0.,Z2)
        X3=(K1*Z1+K2*Z2-FT+FS)/BP
        ZG=(X3-YG)/TL
        YG=INTGRL(0.,ZG)
        YD=YG+YS
        YD=LIMIT(0.,1.,YD)
        TIM=TIME
        YD=RATLIM(TIM,RO,RC,YD)
        YD=1.1*YD-0.05
15      IGOV=IGOV+1
20      CONTINUE
*
* SERVO AND WATER CONTROL VALVE
*
        Y=LIMIT(0.,1.,Y)
        YE=(YD-Y)
        X4=GS*FOLAG(0.,TS,YE)
        Z=LIMIT(RC,RO,X4)
        Y=INTGRL(YIC,Z)
        YC=HSTRSS(YIC,P1,P2,Y)
        YC2=YC*YC
        YC3=YC*YC2
        AE1=A5*YC+A6*YC2
        AE2=A1+A2*YC+A3*YC2+A4*YC3
        AE=SWIN(YC-YNL,AE1,AE2)
```

```
*
* TURBINE AND GENERATOR
*
        QT=AE*(E1*HT-E2*1.**2)**.5
        FW=QT*HT/1.
        IF(ISITE.NE.1) FE=(1.+F1)*FW-F1*1.
        ZF=(FE-FL)/TA
        F=INTGRL(1.,ZF)
*
* PIPELINE
*
        QP=QT
        DHS=AS*(QD-QA)
        HS=INTGRL(HR,DHS)
        DQD=FD*(HR-HS)
        QD=INTGRL(QIC,DQD)
        QA,HA=PIPE(FA,KA,(QB+QC),HS,QIC,HR)
        QB,HB=PIPE(FB,KB,0.,HA,0.,HR)
        QC,HT=PIPE(FC,KC,QP,HA,QIC,HR)
*
TERMINAL
*
        IF(ISITE.NE.0) CALL CLOSE(1)
*
END
```

```
TITLE FREQUENCY TRANSIENT FOR RAPID RESPONSE SYSTEM
        UPDATE WNA=3.,WNB=1.,DA=0.1,DB=0.4
        UPDATE STP=0.001,RR=-6.2E-6
        UPDATE PA=-.3,PB=-.7,PC=1.,PD=0.,PE=0.,PF=0.
INITIAL
        AK1=2*DA/WNA
        AK2=WNA*WNA
        BK1=2*DB/WNB
        BK2=WNB*WNB
DYNAMIC
        X=STP*STEP(3.)
        YA=INTGRL(0.,YDOTA)
        Y2DOTA=(X-AK1*YDOTA-YA)*AK2
        YDOTA=INTGRL(0.,Y2DOTA)
        YB=INTGRL(0.,YDOTB)
        Y2DOTB=(X-BK1*YDOTB-YB)*BK2
        YDOTB=INTGRL(0.,Y2DOTB)
        YC=RAMP2(TIME-13.,RR,1.,0.)
        YD=PA*YA+PB*YB+PC*YC+PD*YDOTA+PE*YDOTB+PF
END
```

```
TITLE HYDRO SET - FDS, GRID CONNECTED VER 1
* THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
* A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
* RESPONSE OF A "FREQUENCY DISTURBANCE SYSTEM" .
*
        REAL NLL,LL,NPF
        DOUBLE PRECISION CNTIM
*
PARAMETER T2=0.2,T3=0.2,AK=2.86,FA=20.07,BK=1.58,FB=1.76,...
CK=2.45,FC=1.14,E1=1.578,E2=0.578,E3=2.36,...
BP=0.03,BT=0.25,TD=16.0,TP=0.3,AK1=3.0,AK2=2.3,TL=33.3
UPDATE P1=-.007,P2=.007,AH=1.,TA=7.,T4=0.2
UPDATE ALL=-.25,UL=.03,RLB=-1.,RLT=1.
UPDATE GS=5.,TRV=0.1,RRV=0.01,NLL=.1,
UPDATE Q1=.8,Q2=0.9,Q4=1.095,Q3=1.2,Q5=1.095
UPDATE FRS=0.372,SK=0.0011
UPDATE RRT1=10.,RRT2=100.,RRT3=20.,DF1=-0.0003,DF2=0.0003
UPDATE PFF=.5,DRRS=0.03,TWO=10.
UPDATE WNA=3.,WNB=1.,DA=0.1,DB=0.4
UPDATE FTRR=-6.2E-6,PA=-.3,PB=-.7,PC=1.,PD=1.,PE=1.,PF=1.
*
INITIAL
        ASK OPERATING POINT,STEP SIZE/EOPP,STP
        IASK RAPID RESPONSE IN(1) OR OUT(0)/IRR
        TOPP=EOPP
        WOPP=(TOPP+NLL)/(1+NLL)
        FLOPP=WOPP/(AH*(E1*AH-E2)**.5)
        SOPP=0
        DO 210 I=1,7
        DX=1./10.**I
205     SOPP=SOPP+DX
        @X2=SOPP*SOPP
        @X3=X2*SOPP
        YN=.1115-0.3665*SOPP+2.546*X2-1.287*X3
        IF(YN.LE.FLOPP) GOTO 205
        SOPP=SOPP-DX
210     CONTINUE
        FS=1.
        AC1=2*DA/WNA
        AC2=WNA*WNA
        BC1=2*DB/WNB
        BC2=WNB*WNB
DYNAMIC
        TIM=TIME
        IF(TIM.NE.0) GOTO 5
        PPF=0.
        NPF=0.
        RROP2=0.
        CNTIM=0.
        LRR=0
5       CONTINUE
```

```
*
* FREQUENCY SIGNAL GENERATOR
*
        @X=STP*STEP(5.)
        @YA=INTGRL(0.,YDOTA)
        @Y2DOTA=(X-AC1*YDOTA-YA)*AC2
        @YDOTA=INTGRL(0.,Y2DOTA)
        @YB=INTGRL(0.,YDOTB)
        @Y2DOTB=(X-BC1*YDOTB-YB)*BC2
        @YDOTB=INTGRL(0.,Y2DOTB)
        @YC=RAMP2(TIME-15.,FTRR,1.,0.)
        FREQ=PA*YA+PB*YB+PC*YC+PD*YDOTA+PE*YDOTB+PF
*
* LOAD
*
        ELPWR=EOPP
        ELTORQ=ELPWR/FREQ
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        FTOP=STPLIM(Q1,Q2,Q3,Q4,FREQ)
        DFTOP=LIMIT(0.,1.095,FTOP)
        DX2=(FS-DFTOP-X2)/T3
        X2=INTGRL(0.,DX2)
        DX3=(DX2-X3)/T4
        X3=INTGRL(0.,DX3)
        DX4=(-GOVOUT+(AK1*DX2+AK2*DX3-FTOP+FS)/BP)/TL
        GOVOUT=INTGRL(0.,DX4)
        DMW=SOPP
        PGOV=DMW+(1.-FTOP)/BP
*
* FREQUENCY DISTURBANCE SYSTEM
*
        DF=DERLAG(1.,0.,0.1,FREQ)                  : DERIVATIVE
        IF(LRR.NE.0) GOTO 95
        IF(DF.LE.DF1.OR.DF.GE.DF2) LRR=1           : RATE DETECTOR
        IF(LRR.EQ.0) GOTO 140
95      IF(M.EQ.1) CNTIM=CNTIM+H                   : TIMER
        IF(CNTIM.GE.RRT1) LRR=2
        IF(CNTIM.GE.RRT2) LRR=3
        IF(CNTIM.GE.RRT2+RRT3) LRR=4
        GOTO (100,110,120,130) LRR                 : PHASE SELECT
* PHASE 1 - RUN
100     CONTINUE
        FREQW=TWO*DERLAG(1.,0.,TWO,FREQ)           : WASHOUT
* PEAK FOLLOWERS
        IF(FREQW.GT.PPF) PPF=FREQW
        IF(FREQW.LT.NPF) NPF=FREQW
* ESCAPE MECHANISM
        IF(NFP.EQ.0.) IFT=1
        IF(PPF.EQ.0.) IFT=0
        IF(IFT.EQ.1) GOTO 102
        IF(PPF.GT.PFF*ABS(NPF)) GOTO 105
        GOTO 103
102     IF(ABS(NPF).GT.PFF*PPF) GOTO 105
```

```
* AVERAGE AND DROOP
103     APF=(PPF+NPF)/2.
        RROP1=-APF/DRRS
        RROP2=RROP1
        GOTO 140
105     CNTIM=RRT2
        GOTO 120
* PHASE 2 - PAYOFF
110     CONTINUE
        PPF=0.
        NPF=0.
        RRR=RROP1/(RRT2-RRT1)
        RROP2=RAMP(5.+RRT1,-RRR,RROP1)
        GOTO 140
* PHASE 3 - INHIBIT
120     CONTINUE
        RRR=0.
        RROP1=0.
        RROP2=0.
        GOTO 140
* PHASE 4 - RESET
130     CONTINUE
        LRR=0
        CNTIM=0.D0
140     CONTINUE
*
* SERVO AND HYDRAULICS
*
        DSP=GOVOUT+DMW+RROP2*IRR
        DSP=LIMIT(0.,1.,DSP)
        DX6=(-GATPOS+DSP)
        DX7=GS*DX6
        DX8=LIMIT(ALL,UL,DX7)
        GATPOS=INTGRL(SOPP,DX8)
        GATPOS=LIMIT(0.,1.,GATPOS)
        GATHYS=HSTRSS(SOPP,P1,P2,GATPOS)
        @GH2=GATHYS*GATHYS
        @GH3=GATHYS*GH2
        A1=1.156*GATHYS-2.82*GH2
        A2=.1115-.3665*GATHYS+2.546*GH2-1.287*GH3
        AREA=SWIN(GATHYS-0.135,A1,A2)
*
* RELEIF VALVE
*
        DXRV=-DX8-RRV
        RVP=INTGRL(0.,DXRV)
        RVP=LIMIT(0.,1.,RVP)
        RVPOS=FOLAG(0.,TRV,RVP)
        RVFLOW=RVPOS*SQRT(TUHEAD)
*
* TURBINE
*
        TUFLOW=AREA*(E1*TUHEAD-E2*FREQ**2)**.5
        PFLOW=TUFLOW+RVFLOW
        WATORQ=TUFLOW*TUHEAD/FREQ
        TUTORQ=((1.+NLL)*WATORQ-NLL)
        DX1=(TUTORQ-ELTORQ)/TA
```

```
*
* PIPELINES
*
        DHS=SK*(QRS-QA)
        HS=INTGRL(AH,DHS)
        DQRS=FRS*(AH-HS)
        QRS=INTGRL(FLOPP,DQRS)
        QA,HA=PIPE(FA,AK,(QB+QC),HS,FLOPP,AH)
        QB,HB=PIPE(FB,BK,0.,HA,0.,AH)
        QC,TUHEAD=PIPE(FC,CK,PFLOW,HA,FLOPP,AH)
END
```

```
TITLE HYDRO SET - FRS, GRID CONNECTED VER 2
* THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
* A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
* RESPONSE OF A "FREQUENCY DISTURBANCE SYSTEM" .
*
* THIS FDS SYSTEM ESCAPES ON ZERO CROSSING
* AND HAS A DELAYED OUTPUT
*
        REAL NLL,LL,NPF
        DOUBLE PRECISION CNTIM
PARAMETER T2=0.2,T3=0.2,AK=2.86,FA=20.07,BK=1.58,FB=1.76,...
CK=2.45,FC=1.14,E1=1.578,E2=0.578,E3=2.36,...
BP=0.03,BT=0.25,TD=16.0,TP=0.3,AK1=3.0,AK2=2.3,TL=33.3
UPDATE P1=-.007,P2=.007,AH=1.,TA=7.,T4=0.2
UPDATE ALL=-.25,UL=.03,RLB=-1.,RLT=1.
UPDATE GS=5.,TRV=0.1,RRV=0.01,NLL=.1,
UPDATE Q1=.8,Q2=0.9,Q4=1.095,Q3=1.2,Q5=1.095
UPDATE FRS=0.372,SK=0.0011
UPDATE RRT1=10.,RRT2=100.,RRT3=20.,DF1=-0.0003,DF2=0.0003
UPDATE PFF=.5,DRRS=0.03,TWO=10.,TOUT=0.7
UPDATE WNA=3.,WNB=1.,DA=0.1,DB=0.4
UPDATE FTRR=-6.2E-6,PA=-.3,PB=-.7,PC=1.,PD=1.,PE=1.,PF=1.
INITIAL
        ASK OPERATING POINT,STEP SIZE/EOPP,STP
        IASK RAPID RESPONSE IN(1) OR OUT(0)/IRR
        TOPP=EOPP
        WOPP=(TOPP+NLL)/(1+NLL)
        FLOPP=WOPP/(AH*(E1*AH-E2)**.5)
        SOPP=0
        DO 210 I=1,7
        DX=1./10.**I
205     SOPP=SOPP+DX
        @X2=SOPP*SOPP
        @X3=X2*SOPP
        YN=.1115-0.3665*SOPP+2.546*X2-1.287*X3
        IF(YN.LE.FLOPP) GOTO 205
        SOPP=SOPP-DX
210     CONTINUE
        FS=1.
        AC1=2*DA/WNA
        AC2=WNA*WNA
        BC1=2*DB/WNB
        BC2=WNB*WNB
DYNAMIC
        TIM=TIME
        IF(TIM.NE.0) GOTO 5
        PPF=0.
        NPF=0.
        RROP2=0.
        CNTIM=0.
        LRR=0
5       CONTINUE
```

```
*
* FREQUENCY SIGNAL GENERATOR
*
        @X=STP*STEP(5.)
        @YA=INTGRL(0.,YDOTA)
        @Y2DOTA=(X-AC1*YDOTA-YA)*AC2
        @YDOTA=INTGRL(0.,Y2DOTA)
        @YB=INTGRL(0.,YDOTB)
        @Y2DOTB=(X-BC1*YDOTB-YB)*BC2
        @YDOTB=INTGRL(0.,Y2DOTB)
        @YC=RAMP2(TIME-15.,FTRR,1.,0.)
        FREQ=PA*YA+PB*YB+PC*YC+PD*YDOTA+PE*YDOTB+PF
*
* LOAD
*
        ELPWR=EOPP
        ELTORQ=ELPWR/FREQ
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        FTOP=STPLIM(Q1,Q2,Q3,Q4,FREQ)
        DFTOP=LIMIT(0.,1.095,FTOP)
        DX2=(FS-DFTOP-X2)/T3
        X2=INTGRL(0.,DX2)
        DX3=(DX2-X3)/T4
        X3=INTGRL(0.,DX3)
        DX4=(-GOVOUT+(AK1*DX2+AK2*DX3-FTOP+FS)/BP)/TL
        GOVOUT=INTGRL(0.,DX4)
        DMW=SOPP
        PGOV=DMW+(1.-FTOP)/BP
*
* FREQUENCY DISTURBANCE SYSTEM
*
        DF=DERLAG(1.,0.,0.1,FREQ)
        IF(LRR.NE.0) GOTO 95
        IF(DF.LE.DF1.OR.DF.GE.DF2) LRR=1
        IF(LRR.EQ.0) GOTO 140
95      IF(M.EQ.1) CNTIM=CNTIM+H
        IF(CNTIM.GE.RRT1) LRR=2
        IF(CNTIM.GE.RRT2) LRR=3
        IF(CNTIM.GE.RRT2+RRT3) LRR=4
        GOTO (100,110,120,130) LRR
100     CONTINUE
        FREQW=TWO*DERLAG(1.,0.,TWO,FREQ)
        IF(FREQW.GT.PPF) PPF=FREQW
        IF(FREQW.LT.NPF) NPF=FREQW
        IF(PPF.NE.0..AND.NPF.NE.0.) GOTO 105
        APF=(PPF+NPF)/2.
        RROP1=-APF/DRRS
        IF(CNTIM.GE.TOUT) RROP2=RROP1
        GOTO 140
105     CNTIM=RRT2
        GOTO 120
110     CONTINUE
        PPF=0.
        NPF=0.
        RRR=RROP1/(RRT2-RRT1)
        RROP2=RAMP(5.+RRT1,-RRR,RROP1)
        GOTO 140
```

```
120     CONTINUE
        RRR=0.
        RROP1=0.
        RROP2=0.
        GOTO 140
130     CONTINUE
        LRR=0
        CNTIM=0.D0
140     CONTINUE
*
* SERVO AND HYDRAULICS
*
        DSP=GOVOUT+DMW+RROP2*IRR
        DSP=LIMIT(0.,1.,DSP)
        DX6=(-GATPOS+DSP)
        DX7=GS*DX6
        DX8=LIMIT(ALL,UL,DX7)
        GATPOS=INTGRL(SOPP,DX8)
        GATPOS=LIMIT(0.,1.,GATPOS)
        GATHYS=HSTRSS(SOPP,P1,P2,GATPOS)
        @GH2=GATHYS*GATHYS
        @GH3=GATHYS*GH2
        A1=1.156*GATHYS-2.82*GH2
        A2=.1115-.3665*GATHYS+2.546*GH2-1.287*GH3
        AREA=SWIN(GATHYS-0.135,A1,A2)
*
* RELEIF VALVE
*
        DXRV=-DX8-RRV
        RVP=INTGRL(0.,DXRV)
        RVP=LIMIT(0.,1.,RVP)
        RVPOS=FOLAG(0.,TRV,RVP)
        RVFLOW=RVPOS*SQRT(TUHEAD)
*
* TURBINE
*
        TUFLOW=AREA*(E1*TUHEAD-E2*FREQ**2)**.5
        PFLOW=TUFLOW+RVFLOW
        WATORQ=TUFLOW*TUHEAD/FREQ
        TUTORQ=((1.+NLL)*WATORQ-NLL)
        DX1=(TUTORQ-ELTORQ)/TA
*
* PIPELINES
*
        DHS=SK*(QRS-QA)
        HS=INTGRL(AH,DHS)
        DQRS=FRS*(AH-HS)
        QRS=INTGRL(FLOPP,DQRS)
        QA,HA=PIPE(FA,AK,(QB+QC),HS,FLOPP,AH)
        QB,HB=PIPE(FB,BK,0.,HA,0.,AH)
        QC,TUHEAD=PIPE(FC,CK,PFLOW,HA,FLOPP,AH)
END
```

```
TITLE HYDRO SET - FDS, ISOLATED LOAD
* THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
* CLOSED LOOP (ISOLATED LOAD) SIMULATION
* WITH FREQUENCY DISTURBANCE SYSTEM.
*
* THIS FDS SYSTEM ESCAPES ON ZERO CROSSING
* AND HAS A DELAYED OUTPUT.
*
        REAL NLL,LL,NPF
        DOUBLE PRECISION CNTIM
*
PARAMETER T2=0.2,T3=0.2,AK=2.86,FA=20.07,BK=1.58,FB=1.76,...
CK=2.45,FC=1.14,E1=1.578,E2=0.578,E3=2.36,...
BP=0.03,BT=0.25,TD=16.0,TP=0.3,AK1=3.0,AK2=2.3,TL=33.3
UPDATE P1=-.007,P2=.007,AH=1.,TA=7.,T4=0.2
UPDATE ALL=-.25,UL=.03,RLB=-1.,RLT=1.
UPDATE GS=5.,TRV=0.1,RRV=0.01,NLL=.1,
UPDATE Q1=.8,Q2=0.9,Q4=1.095,Q3=1.2,Q5=1.095
UPDATE FRS=0.372,SK=0.0011
UPDATE RRT1=10.,RRT2=100.,RRT3=20.,DF1=-0.0003,DF2=0.0003
UPDATE PFF=.5,DRRS=0.03,TWO=10.,TOUT=0.7
INITIAL
        ASK OPERATING POINT,STEP SIZE/EOPP,STP
        IASK RAPID RESPONSE IN(1) OR OUT(0)/IRR
        TOPP=EOPP
        WOPP=(TOPP+NLL)/(1+NLL)
        FLOPP=WOPP/(AH*(E1*AH-E2)**.5)
        SOPP=0
        DO 210 I=1,7
        DX=1./10.**I
205     SOPP=SOPP+DX
        @X2=SOPP*SOPP
        @X3=X2*SOPP
        YN=.1115-0.3665*SOPP+2.546*X2-1.287*X3
        IF(YN.LE.FLOPP) GOTO 205
        SOPP=SOPP-DX
210     CONTINUE
        FS=1.
DYNAMIC
        TIM=TIME
        IF(TIM.NE.0) GOTO 5
        PPF=0.
        NPF=0.
        RROP2=0.
        CNTIM=0.
        LRR=0
5       CONTINUE
*
* LOAD
*
        ELPWR=EOPP+STP*STEP(5.)
        ELTORQ=ELPWR/FREQ
```

```
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        FTOP=STPLIM(Q1,Q2,Q3,Q4,FREQ)
        DFTOP=LIMIT(0.,1.095,FTOP)
        DX2=(FS-DFTOP-X2)/T3
        X2=INTGRL(0.,DX2)
        DX3=(DX2-X3)/T4
        X3=INTGRL(0.,DX3)
        DX4=(-GOVOUT+(AK1*DX2+AK2*DX3-FTOP+FS)/BP)/TL
        GOVOUT=INTGRL(0.,DX4)
        DMW=SOPP
        PGOV=DMW+(1.-FTOP)/BP
*
* FREQUENCY DISTURBANCE SYSTEM
*
        DF=DERLAG(1.,0.,0.1,FREQ)
        IF(LRR.NE.0) GOTO 95
        IF(DF.LE.DF1.OR.DF.GE.DF2) LRR=1
        IF(LRR.EQ.0) GOTO 140
95      IF(M.EQ.1) CNTIM=CNTIM+H
        IF(CNTIM.GE.RRT1) LRR=2
        IF(CNTIM.GE.RRT2) LRR=3
        IF(CNTIM.GE.RRT2+RRT3) LRR=4
        GOTO (100,110,120,130) LRR
100     CONTINUE
        FREQW=TWO*DERLAG(1.,0.,TWO,FREQ)
        IF(FREQW.GT.PPF) PPF=FREQW
        IF(FREQW.LT.NPF) NPF=FREQW
        IF(PPF.NE.0..AND.NPF.NE.0.) GOTO 105
        APF=(PPF+NPF)/2.
        RROP1=-APF/DRRS
        IF(CNTIM.GE.TOUT) RROP2=RROP1
        GOTO 140
105     CNTIM=RRT2
        GOTO 120
110     CONTINUE
        PPF=0.
        NPF=0.
        RRR=RROP1/(RRT2-RRT1)
        RROP2=RAMP(5.+RRT1,-RRR,RROP1)
        GOTO 140
120     CONTINUE
        RRR=0.
        RROP1=0.
        RROP2=0.
        GOTO 140
130     CONTINUE
        LRR=0
        CNTIM=0.D0
140     CONTINUE
```

```
*
* SERVO AND HYDRAULICS
*
        DSP=GOVOUT+DMW+RROP2*IRR
        DSP=LIMIT(0.,1.,DSP)
        DX6=(-GATPOS+DSP)
        DX7=GS*DX6
        DX8=LIMIT(ALL,UL,DX7)
        GATPOS=INTGRL(SOPP,DX8)
        GATPOS=LIMIT(0.,1.,GATPOS)
        GATHYS=HSTRSS(SOPP,P1,P2,GATPOS)
        @GH2=GATHYS*GATHYS
        @GH3=GATHYS*GH2
        A1=1.156*GATHYS-2.82*GH2
        A2=.1115-.3665*GATHYS+2.546*GH2-1.287*GH3
        AREA=SWIN(GATHYS-0.135,A1,A2)
*
* RELEIF VALVE
*
        DXRV=-DX8-RRV
        RVP=INTGRL(0.,DXRV)
        RVP=LIMIT(0.,1.,RVP)
        RVPOS=FOLAG(0.,TRV,RVP)
        RVFLOW=RVPOS*SQRT(TUHEAD)
*
* TURBINE
*
        TUFLOW=AREA*(E1*TUHEAD-E2*FREQ**2)**.5
        PFLOW=TUFLOW+RVFLOW
        WATORQ=TUFLOW*TUHEAD/FREQ
        TUTORQ=((1.+NLL)*WATORQ-NLL)
        DX1=(TUTORQ-ELTORQ)/TA
        FREQ=INTGRL(1.,DX1)
*
* PIPELINES
*
        DHS=SK*(QRS-QA)
        HS=INTGRL(AH,DHS)
        DQRS=FRS*(AH-HS)
        QRS=INTGRL(FLOPP,DQRS)
        QA,HA=PIPE(FA,AK,(QB+QC),HS,FLOPP,AH)
        QB,HB=PIPE(FB,BK,0.,HA,0.,AH)
        QC,TUHEAD=PIPE(FC,CK,PFLOW,HA,FLOPP,AH)
END
```

```
TITLE HYDRO SET - FDS, LOAD REJECTION
* THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
* LOAD REJECTION SIMULATION
* WITH FREQUENCY DISTURBANCE SYSTEM.
*
* THIS FDS SYSTEM ESCAPES ON ZERO CROSSING
* AND HAS A DELAYED OUTPUT.
*
        REAL NLL,LL,NPF
        DOUBLE PRECISION CNTIM
*
PARAMETER T2=0.2,AK=2.86,FA=20.07,BK=1.58,FB=1.76,...
CK=2.45,FC=1.14,E1=1.578,E2=0.578,E3=2.36,...
BP=0.03,BT=0.25,TD=16.0,TP=0.3,
UPDATE P1=-.007,P2=.007,AH=1.,TA=7.,T4=0.2
UPDATE ALL=-.25,UL=.03,RLB=-1.,RLT=1.
UPDATE GS=5.,TRV=0.1,RRV=0.01,NLL=.1,
UPDATE Q1=.8,Q2=0.9,Q4=1.095,Q3=1.2,Q5=1.095
UPDATE FRS=0.372,SK=0.0011
UPDATE RRT1=10.,RRT2=100.,RRT3=20.,DF1=-0.0003,DF2=0.0003
UPDATE PFF=.5,DRRS=0.03,TWO=10.,TOUT=2.
*
INITIAL
        ASK OPERATING POINT,STEP SIZE/EOPP,STP
        IASK RAPID RESPONSE IN(1) OR OUT(0)/IRR
        IASK GOVERNOR TYPE AFTER OCB (0 FOR DD, 1 FOR TD)/IGT
        TOPP=EOPP
        WOPP=(TOPP+NLL)/(1+NLL)
        FLOPP=WOPP/(AH*(E1*AH-E2)**.5)
        SOPP=0
        DO 210 I=1,7
        DX=1./10.**I
205     SOPP=SOPP+DX
        @X2=SOPP*SOPP
        @X3=X2*SOPP
        YN=.1115-0.3665*SOPP+2.546*X2-1.287*X3
        IF(YN.LE.FLOPP) GOTO 205
        SOPP=SOPP-DX
210     CONTINUE
        FS=1.
DYNAMIC
        TIM=TIME
        IF(TIM.NE.0) GOTO 5
        PPF=0.
        NPF=0.
        RROP2=0.
        CNTIM=0.
        LRR=0
5       CONTINUE
*
* LOAD
*
        ELPWR=EOPP+STP*STEP(5.)
        ELTORQ=ELPWR/FREQ
```

```
*
* FREQUENCY TRANSDUCER AND GOVERNOR
*
        FTOP=STPLIM(Q1,Q2,Q3,Q4,FREQ)
        DFTOP=LIMIT(0.,1.095,FTOP)
        T3=SWIN(IGT-1,0.2,1.018)
        AK1=SWIN(IGT-1,3.,14.9892)
        AK2=SWIN(IGT-1,2.3,0.)
        TL=SWIN(IGT-1,33.3,158.29)
        DX2=(FS-DFTOP-X2)/T3
        X2=INTGRL(0.,DX2)
        DX3=(DX2-X3)/T4
        X3=INTGRL(0.,DX3)
        DX4=(-GOVOUT+(AK1*DX2+AK2*DX3-FTOP+FS)/BP)/TL
        GOVOUT=INTGRL(0.,DX4)
        DMW=RAMP(5.,RMW,SOPP,FVMW)
        LL=RAMP(5.,RLL,1.,FVMW)
        GOVOUT=LIMIT(-DMW,LL-DMW,GOVOUT)
        PGOV=DMW+(1.-FTOP)/BP
*
* FREQUENCY DISTURBANCE SYSTEM
*
        DF=DERLAG(1.,0.,0.1,FREQ)
        IF(LRR.NE.0) GOTO 95
        IF(DF.LE.DF1.OR.DF.GE.DF2) LRR=1
        IF(LRR.EQ.0) GOTO 140
95      IF(M.EQ.1) CNTIM=CNTIM+H
        IF(CNTIM.GE.RRT1) LRR=2
        IF(CNTIM.GE.RRT2) LRR=3
        IF(CNTIM.GE.RRT2+RRT3) LRR=4
        GOTO (100,110,120,130) LRR
100     CONTINUE
        FREQW=TWO*DERLAG(1.,0.,TWO,FREQ)
        IF(FREQW.GT.PPF) PPF=FREQW
        IF(FREQW.LT.NPF) NPF=FREQW
        IF(PPF.NE.0..AND.NPF.NE.0.) GOTO 105
        APF=(PPF+NPF)/2.
        RROP1=-APF/DRRS
        IF(CNTIM.GE.TOUT) RROP2=RROP1
        GOTO 140
105     CNTIM=RRT2
        GOTO 120
110     CONTINUE
        PPF=0.
        NPF=0.
        RRR=RROP1/(RRT2-RRT1)
        RROP2=RAMP(5.+RRT1,-RRR,RROP1)
        GOTO 140
120     CONTINUE
        RRR=0.
        RROP1=0.
        RROP2=0.
        GOTO 140
130     CONTINUE
        LRR=0
        CNTIM=0.D0
140     CONTINUE
```

```
*
* SERVO AND HYDRAULICS
*
        DSP=GOVOUT+DMW+RROP2*IRR
        DSP=LIMIT(0.,1.,DSP)
        DX6=(-GATPOS+DSP)
        DX7=GS*DX6
        DX8=LIMIT(ALL,UL,DX7)
        GATPOS=INTGRL(SOPP,DX8)
        GATPOS=LIMIT(0.,1.,GATPOS)
        GATHYS=HSTRSS(SOPP,P1,P2,GATPOS)
        @GH2=GATHYS*GATHYS
        @GH3=GATHYS*GH2
        A1=1.156*GATHYS-2.82*GH2
        A2=.1115-.3665*GATHYS+2.546*GH2-1.287*GH3
        AREA=SWIN(GATHYS-0.135,A1,A2)
*
* RELEIF VALVE
*
        DXRV=-DX8-RRV
        RVP=INTGRL(0.,DXRV)
        RVP=LIMIT(0.,1.,RVP)
        RVPOS=FOLAG(0.,TRV,RVP)
        RVFLOW=RVPOS*SQRT(TUHEAD)
*
* TURBINE
*
        TUFLOW=AREA*(E1*TUHEAD-E2*FREQ**2)**.5
        PFLOW=TUFLOW+RVFLOW
        WATORQ=TUFLOW*TUHEAD/FREQ
        TUTORQ=((1.+NLL)*WATORQ-NLL)
        DX1=(TUTORQ-ELTORQ)/TA
        FREQ=INTGRL(1.,DX1)
*
* PIPELINES
*
        DHS=SK*(QRS-QA)
        HS=INTGRL(AH,DHS)
        DQRS=FRS*(AH-HS)
        QRS=INTGRL(FLOPP,DQRS)
        QA,HA=PIPE(FA,AK,(QB+QC),HS,FLOPP,AH)
        QB,HB=PIPE(FB,BK,0.,HA,0.,AH)
        QC,TUHEAD=PIPE(FC,CK,PFLOW,HA,FLOPP,AH)
END
```

```
TITLE BASIC POWER SYSTEM MODEL VER 1
*
        REAL MH,MI,ML,MX
*
UPDATE FIC=1.0,TG=0.1,AN=1.0,TIV=0.1,TR=10.0,R=0.35
UPDATE PS=1.0,AK1=0.015,AK2=5.0,AK3=0.0,TC=45.0,TM=2.0,AH=5.18
UPDATE G=25.0,AK=0.2,A1L=0.001,A1H=1.00,SSLFC=4.0
UPDATE FDL=0.001,FDH=1.05,FSOS=1.04,AIVD=1.0,B1L=0.001,B1H=1.0
UPDATE TGVO=0.7,TGVC=0.1,ML=0.2,MH=1.2,TD=30.0,EMH=1.05
*
INITIAL
        ASK INITIAL OPERATING POINT AND STEP SIZE/TPIC,DPE
        ASK MILL START DELAY TIME,BOILER TIME CONSTANT/TS1,TB
        FDI=TPIC/G
        FS=FIC+FDI
        A1I=TPIC
        Y2I=TPIC
        MI=TPIC
        FII=TPIC
        A2I=TPIC
        P2I=TPIC
        B1I=1.0
        TS=TS1+4
DYNAMIC
*
* GOVERNOR
*
        FE=FS-F
        Z1=(G*FE-AX)/TG
        AX=INTGRL(A1I,Z1)
        A1=LIMIT(A1L,A1H,AX)
        FEOS=FSOS-F
        BX=B1I+(100.0*FEOS/AIVD)
        B1=LIMIT(B1L,B1H,BX)
*
* MASTER PRESSURE CONTROLLER
*
        PE=PS-PB
        Y1=AK1*PE
        Y2=INTGRL(Y2I,Y1)
        MX=Y2+AK2*PE+AK3*PE
        AM=LIMIT(ML,MH,MX)
*
* FUEL FEED SYSTEM
*
        FD=LIMIT(FDL,FDH,FD1)
        QI=AM*FD
        EM1=AM-MI
        EM2=DELAY(TD,EM1)
        EM3=MI+EM2
        EML=SWIN(TIME-TS,MI,EMH)
        EM=LIMIT(0.001,EML,EM3)
        Y6=(EM-FI)/TC
        FI=INTGRL(FII,Y6)
        Y7=(FI-QI)/TM
        FD1=INTGRL(1.,Y7)
```

```
*
* STEAM VALVES
*
        ZZ=(A1-A2)
        TGV=SWIN(ZZ,TGVC,TGVO)          : OPENING OR CLOSING
        Z2=ZZ/TGV
        A2=INTGRL(A2I,Z2)
        Z3=(B1-B2)/TIV
        B2=INTGRL(1.,Z3)
*
* TURBINE
*
        P1=A2*PB
        W1=(SQRT(ABS((1-(P2*R/P1)**2)/(1-R**2))))*P1
        W2=B2*P2
        Z4=(W1-W2)/TR
        P2=INTGRL(P2I,Z4)               : REHEATER
        PM1=AK*W1*((1-(P2*R/P1)**0.231)/(1-R**0.231))
        PM2=(1.-AK)*W2
        PT=PM1+PM2
*
* BOILER
*
        Y8=(QI-W1)/TB
        PB=INTGRL(1.,Y8)
*
* GENERATOR AND LOAD
*
        PE=TPIC+DPE*STEP(4.)
        PEFC=PE*(1.-(FIC-F)*(SSLFC/2.))
        X1=(PT-PEFC)/(2.*AH*F)
        F=INTGRL(1.,X1)
END
```

```
TITLE POWER SYSTEM OPEN LOOP MODEL VER3
*
        REAL MH,MI,ML,MX
*
UPDATE FIC=1.0,TG=0.1,AN=1.0,TIV=0.1,TR=10.0,R=0.35
UPDATE PS=1.0,AK1=0.015,AK2=5.0,AK3=0.0,TC=45.0,TM=2.0,AH=5.18
UPDATE G=25.0,AK=0.2,A1L=0.001,A1H=1.00,SSLFC=4.0
UPDATE FDL=0.001,FDH=1.05,FSOS=1.04,AIVD=1.0,B1L=0.001,B1H=1.0
UPDATE TGVO=0.7,TGVC=0.1,ML=0.2,MH=1.2,TD=30.0,EMH=1.05
*
INITIAL
*
ASK GOVERNOR VALVE INITIAL POSITION AND STEP SIZE/GVIC,DGV
ASK MILL START DELAY TIME,BOILER TIME CONSTANT/TS1,TB
        FDI=GVIC/G
        FS=FIC+FDI
        A1I=GVIC
        Y2I=GVIC
        MI=GVIC
        FII=GVIC
        A2I=GVIC
        P2I=GVIC
        B1I=1.0
        TS=TS1+4.
DYNAMIC
*
* MASTER PRESSURE CONTROLLER
*
        PE=PS-PB
        Y1=AK1*PE
        Y2=INTGRL(Y2I,Y1)
        MX=Y2+AK2*PE+AK3*PE
        AM=LIMIT(ML,MH,MX)
*
* FUEL FEED SYSTEM
*
        FD=LIMIT(FDL,FDH,FD1)
        QI=AM*FD
        EM1=AM-MI
        EM2=DELAY(TD,EM1)
        EM3=MI+EM2
        EML=SWIN(TIME-TS,MI,EMH)
        EM=LIMIT(0.001,EML,EM3)
        Y6=(EM-FI)/TC
        FI=INTGRL(FII,Y6)
        Y7=(FI-QI)/TM
        FD1=INTGRL(1.,Y7)
*
* STEAM VALVES
*
        A2=A2I+DGV*STEP(4.)
        B2=B1I
```

```
*
* TURBINE
*
        P1=A2*PB
        W1=(SQRT(ABS((1-(P2*R/P1)**2)/(1-R**2))))*P1
        W2=B2*P2
        Z4=(W1-W2)/TR
        P2=INTGRL(P2I,Z4)
        PM1=AK*W1*((1-(P2*R/P1)**0.231)/(1-R**0.231))
        PM2=(1.-AK)*W2
        PT=PM1+PM2
*
* BOILER
*
        Y8=(QI-W1)/TB
        PB=INTGRL(1.,Y8)
END
```

```
TITLE EXTENDED POWER SYSTEM MODEL VER 1.1
*
        REAL MH,MI,ML,MX
*
UPDATE FIC=1.0,TG=0.1,AN=1.0,TIV=0.1,TR=10.0,R=0.35
UPDATE PS=1.0,AK1=0.015,AK2=5.0,AK3=0.0,TC=45.0,TM=2.0,AH=5.18
UPDATE G=25.0,AK=0.2,A1L=0.001,A1H=1.00,SSLFC=4.0
UPDATE FDL=0.001,FDH=1.05,FSOS=1.04,AIVD=1.0,B1L=0.001,B1H=1.0
UPDATE TGVO=0.7,TGVC=0.1,ML=0.2,MH=1.2,TD=30.0,EMH=1.05
*
INITIAL
*
ASK THERMAL PLANT,NUCLEAR PLANT,TOTAL LOAD,STEP SIZE/POB1,POB3,TLMO,DPE
ASK PROPORTION OF REGULATED THERMAL PLANT/RR
ASK MILL START DELAY TIME,BOILER TIME CONSTANT/TS1,TB
        POBT=POB1+POB3
        TPIC=(TLMO-POB3)/POB1
        PTT3=POB3
        FDI=TPIC/G
        FS=FIC+FDI
        A1I=TPIC
        Y2I=TPIC
        MI=TPIC
        FII=TPIC
        A2I=TPIC
        P2I=TPIC
        B1I=1.0
        TS=TS1+4
DYNAMIC
*
* GOVERNOR
        FE=FS-F
        Z1=(G*FE-AX)/TG
        AX=INTGRL(A1I,Z1)
        A1=LIMIT(A1L,A1H,AX)
        FEOS=FSOS-F
        BX=B1I+(100.0*FEOS/AIVD)
        B1=LIMIT(B1L,B1H,BX)
*
* MASTER PRESSURE CONTROLLER
*
        PE=PS-PB
        Y1=AK1*PE
        Y2=INTGRL(Y2I,Y1)
        MX=Y2+AK2*PE+AK3*PE
        AM=LIMIT(ML,MH,MX)
*
* FUEL FEED SYSTEM
*
        FD=LIMIT(FDL,FDH,FD1)
        QI=AM*FD
        EM1=AM-MI
        EM2=DELAY(TD,EM1)
        EM3=MI+EM2
        EML=SWIN(TIME-TS,MI,EMH)
        EM=LIMIT(0.001,EML,EM3)
        Y6=(EM-FI)/TC
        FI=INTGRL(FII,Y6)
        Y7=(FI-QI)/TM
        FD1=INTGRL(1.,Y7)
```

```
*
* STEAM VALVES
*
      ZZ=(A1-A2)
      TGV=SWIN(ZZ,TGVC,TGVO)          : OPENING OR CLOSING
      Z2=ZZ/TGV
      A2=INTGRL(A2I,Z2)
      Z3=(B1-B2)/TIV
      B2=INTGRL(1.,Z3)
*
* TURBINE
*
      P1=A2*PB
      W1=(SQRT(ABS((1-(P2*R/P1)**2)/(1-R**2))))*P1
      W2=B2*P2
      Z4=(W1-W2)/TR
      P2=INTGRL(P2I,Z4)               : REHEATER
      PM1=AK*W1*((1-(P2*R/P1)**0.231)/(1-R**0.231))
      PM2=(1.-AK)*W2
      PT=PM1+PM2
*
* BOILER
*
      Y8=(QI-W1)/TB
      PB=INTGRL(1.,Y8)
*
* POWER OUTPUT - ALL PLANT TYPES
*
      PTT1=PT*POB1*RR
      PTT2=TPIC*POB1*(1-RR)
      PGTT=PTT1+PTT2+PTT3
*
* LOAD AND INERTIA
*
      PE=TLMO+DPE*STEP(4.)
      PEFC=PE*(1.-(FIC-F)*(SSLFC/2.))
      X1=(PGTT-PEFC)/(2.*POBT*AH*F)
      F=INTGRL(1.,X1)
END
```

```
TITLE EXTENDED POWER SYSTEM MODEL VER 2.1
*
        REAL MH,MI,ML,MX
*
UPDATE FIC=1.0,TG=0.1,AN=1.0,TIV=0.1,TR=10.0,R=0.35
UPDATE PS=1.0,AK1=0.015,AK2=5.0,AK3=0.0,TC=45.0,TM=2.0,AH=5.18
UPDATE G=25.0,AK=0.2,A1L=0.001,A1H=1.00,SSLFC=4.0
UPDATE FDL=0.001,FDH=1.05,FSOS=1.04,AIVD=1.0,B1L=0.001,B1H=1.0
UPDATE TGVO=0.7,TGVC=0.1,ML=0.2,MH=1.2,TD=30.0,EMH=1.05
*
INITIAL
*
ASK THERMAL PLANT,NUCLEAR PLANT,TOTAL LOAD,STEP SIZE/POB1,POB3,TLMO,DPE
ASK RATE OF RAMP(GW PER S)/RMP
ASK PROPORTION OF REGULATED THERMAL PLANT/RR
ASK MILL START DELAY TIME,BOILER TIME CONSTANT/TS1,TB
        POBT=POB1+POB3
        TPIC=(TLMO-POB3)/POB1
        PTT3=POB3
        FDI=TPIC/G
        FS=FIC+FDI
        A1I=TPIC
        Y2I=TPIC
        MI=TPIC
        FII=TPIC
        A2I=TPIC
        P2I=TPIC
        B1I=1.0
DYNAMIC
*
* START PUMP AFTER LOAD FALLS BY 'DPE'
*
        PE1=RAMP(5.,RMP,TLMO)
        DL=TLMO-PE1
        STPTIM=SWIN(DL-DPE,TIME,STPTIM)
        STP=DPE*(0.2*STEP(STPTIM+1.)+0.8*STEP(STPTIM+31.))
        TS=STPTIM+TS1
*
* GOVERNOR
*
        FE=FS-F
        Z1=(G*FE-AX)/TG
        AX=INTGRL(A1I,Z1)
        A1=LIMIT(A1L,A1H,AX)
        FEOS=FSOS-F
        BX=B1I+(100.0*FEOS/AIVD)
        B1=LIMIT(B1L,B1H,BX)
*
* MASTER PRESSURE CONTROLLER
*
        PE=PS-PB
        Y1=AK1*PE
        Y2=INTGRL(Y2I,Y1)
        MX=Y2+AK2*PE+AK3*PE
        AM=LIMIT(ML,MH,MX)
```

```
*
* FUEL FEED SYSTEM
*
        FD=LIMIT(FDL,FDH,FD1)
        QI=AM*FD
        EM1=AM-MI
        EM2=DELAY(TD,EM1)
        EM3=MI+EM2
        EML=SWIN(TIME-TS,MI,EMH)
        EM=LIMIT(0.001,EML,EM3)
        Y6=(EM-FI)/TC
        FI=INTGRL(FII,Y6)
        Y7=(FI-QI)/TM
        FD1=INTGRL(1.,Y7)
*
* STEAM VALVES
*
        ZZ=(A1-A2)
        TGV=SWIN(ZZ,TGVC,TGVO)          : OPENING OR CLOSING
        Z2=ZZ/TGV
        A2=INTGRL(A2I,Z2)
        Z3=(B1-B2)/TIV
        B2=INTGRL(1.,Z3)
*
* TURBINE
*
        P1=A2*PB
        W1=(SQRT(ABS((1-(P2*R/P1)**2)/(1-R**2))))*P1
        W2=B2*P2
        Z4=(W1-W2)/TR
        P2=INTGRL(P2I,Z4)               : REHEATER
        PM1=AK*W1*((1-(P2*R/P1)**0.231)/(1-R**0.231))
        PM2=(1.-AK)*W2
        PT=PM1+PM2
*
* BOILER
*
        Y8=(QI-W1)/TB
        PB=INTGRL(1.,Y8)
*
* POWER OUTPUT - ALL PLANT TYPES
*
        PTT1=PT*POB1*RR
        PTT2=TPIC*POB1*(1-RR)
        PGTT=PTT1+PTT2+PTT3
*
* LOAD AND INERTIA
*
        PE=PE1+STP
        PEFC=PE*(1.-(FIC-F)*(SSLFC/2.))
        X1=(PGTT-PEFC)/(2.*POBT*AH*F)
        F=INTGRL(1.,X1)
END
```

APPENDIX 3

This appendix contains the Documentation Files for all the simulation runs which were carried out to obtain the Figures presented in this Thesis. In addition, a brief description of all the relevant site tests has been included.

Site Test:- SLC300

Date:- 19th February 1980

Reservoir Level :- 930ft

Effective Head:- 903ft

Governor Type:- Microprocessor TD/DD

Description:- Run-up from standstill to synchronism under the control of the station Auto-start sequence. TD governor initially, changed to DD governor on synchronisation.

```
HYDRO SET - RUN UP SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL
MODIFIED TO SIMULATE THE RUN UP CHARACTERISTIC.
THE RELIEF VALVE HAS BEEN OMITTED FROM THIS MODEL.
VERSION 8

DATE 23-JUN-81      TIME 18:40:17      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =    1.000000
TA =    7.000000       T3 =    1.018000       T4 =   0.2000000
RC =  -0.2500000       RO =   0.3000000E-01   K1 =    14.99000
K2 =   0.0000000       TL =    158.2900
GS =    2.500000       TS =   0.1000000
Q1 =   0.7000000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000       Q6 =   0.9550000
Q7 =   0.9800000
FD =    1.230000       AS =   0.1100000E-02   LL1 =   0.2770000
LL2 =   0.2270000      RL1 =   0.1800000E-01  RL2 =   0.1800000E-01
QL =   0.8800000E-01   F1 =   0.1100000       E1 =    1.584000
E2 =   0.5840000       YNL =   0.1200000      FPB =   0.5000000
A1 =   0.6480000E-01   A2 =   0.1561000       A3 =    1.929000
A4 =   -1.166000       A5 =    1.253000       A6 =   -2.852000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.860000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME R4A11.OUT
STORED VARIABLES:-

  TIME        F           FT          YSA         YD          Y
```

Site Test:- SLT401

Date:- 30th August 1979

Reservoir Level :- 883ft

Effective Head:- 858ft

Governor Type:- Microprocessor TD/DD

Description:- Run-up from standstill to synchronism partly under the control of the station Auto-start sequence. TD governor initially, changed to DD governor on synchronisation

```
HYDRO SET - RUN UP SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL
MODIFIED TO SIMULATE THE RUN UP CHARACTERISTIC.
THE RELIEF VALVE HAS BEEN OMITTED FROM THIS MODEL.
VERSION 8

DATE 29-APR-81      TIME 20:36:26      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =   0.9800000
TA =    7.000000       T3 =    1.018000       T4 =   0.2000000
RC =  -0.2500000       RO =   0.3000000E-01   K1 =    14.99000
K2 =   0.0000000       TL =    158.2900
GS =    2.500000       TS =   0.1000000
Q1 =   0.7000000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000       Q6 =   0.9550000
Q7 =    1.000000
FD =    1.230000       AS =   0.1100000E-02   LL1 =   0.2550000
LL2 =   0.2320000      RL1 =   0.1800000E-01  RL2 =   0.1900000E-01
QL =   0.8800000E-01   F1 =   0.1100000       E1 =    1.584000
E2 =   0.5840000       YNL =   0.1200000      FPB =   0.5000000
A1 =   0.5980000E-01   A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.850000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME R4A10.OUT
STORED VARIABLES:-

  TIME        F           FT          YSA         YD          Y
```

Site Test:- SLC306

Date:- 19th February 1980

Reservoir Level :- 930ft

Effective Head:- 903ft

Governor Type:- Microprocessor DD

Description:- Load rejection test; initial load approximately 15MW.

```
HYDRO SET - LOAD REJECTION SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL MODIFIED TO SIMULATE
A LOAD REJECTION, WITH A CHOICE OF DD OR TD GOVERNOR.
AN OPTION WHICH PERMITS THE SERVO SET POINT AND LOAD LIMIT RAMPS
TO BE DISABLED HAS ALSO BEEN INCLUDED, AND A RATE LIMIT ON
DESIRED SERVO POSITION HAS BEEN IMPLEMENTED.
VERSION 10

DATE 05-AUG-81      TIME 19:45:31      RUN NUMBER    4

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =    1.000000
TA =    8.700000
RC =  -0.2500000       RO =   0.3000000E-01   FS =   0.9995000
TS =   0.1000000
GS =    2.500000       TR =   0.2000000       RT =   0.3000000E-01
RU =   0.1000000E-01   RM =   0.2500000E-01
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.100000
Q3 =    2.000000       Q5 =    1.100000
FD =    1.230000       AS =   0.1100000E-02   LL2 =   0.2400000
RL2 =   0.2000000E-01  YNL =   0.9500000E-01
F1 =   0.8200000E-01   E1 =    1.584000       E2 =   0.5840000
A1 =   0.6480000E-01   A2 =   0.1561000       A3 =    1.929000
A4 =   -1.166000       A5 =    1.253000       A6 =   -2.852000

ASK VARIABLES
PLMW =    15.50000
FIC =    1.002500
IGOV =    1
IRAMP =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.350000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    6
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME DD1515.OUT
STORED VARIABLES:-
  TIME       FT         YSA        YD         Y          YR

POST-RUN PASS NUMBER    7
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME DD1516.OUT
STORED VARIABLES:-
  TIME       Z1         Z2         X3         YG         HT
```

```
HYDRO SET - LOAD REJECTION SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL MODIFIED TO SIMULATE
A LOAD REJECTION, WITH A CHOICE OF DD OR TD GOVERNOR.
AN OPTION WHICH PERMITS THE SERVO SET POINT AND LOAD LIMIT RAMPS
TO BE DISABLED HAS ALSO BEEN INCLUDED, AND A RATE LIMIT ON
DESIRED SERVO POSITION HAS BEEN IMPLEMENTED.
VERSION 9

DATE 23-JUN-81      TIME 21:03:20      RUN NUMBER    7

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =     2.710000       FA =     20.24000       KB =     1.460000
FB =     1.900000
KC =     2.410000       FC =     1.150000
BP =   0.3000000E-01    BT =     0.2500000      TD =     16.00000
TP =     0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    HR =     1.000000
TA =     8.700000
RC =    -0.2500000      RO =   0.3000000E-01    FS =     1.000000
TS =     0.1000000
GS =     2.500000       TR =     0.2000000      RT =     1.000000
RU =   0.1000000E-01    RM =   0.2500000E-01
Q1 =     0.8750000      Q2 =     0.9000000      Q4 =     1.100000
Q3 =     2.000000       Q5 =     1.100000
FD =     1.230000       AS =   0.1100000E-02    LL2 =    0.2400000
RL2 =  0.2000000E-01    YNL =    0.1200000
F1 =   0.8200000E-01    E1 =     1.584000       E2 =   0.5840000
A1 =   0.6480000E-01    A2 =     0.1561000      A3 =     1.929000
A4 =    -1.166000       A5 =     1.253000       A6 =    -2.852000

ASK VARIABLES
PLMW =     15.50000
IGOV =     1
IRAMP =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01
FINISH TIME  0.350000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    9
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME DD1511.OUT
STORED VARIABLES:-
  TIME       F          FT         Y          HT         YR

POST-RUN PASS NUMBER   10
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME DD1512.OUT
STORED VARIABLES:-
  TIME       F          FT         Y          HT         YR
```

Site Test:- SLC203

Date:- 6th February 1980

Reservoir Level :- 928ft

Effective Head:- 900ft

Governor Type:- Microprocessor TD

Description:- Load rejection test; initial load approximately 15MW.

```
HYDRO SET - LOAD REJECTION SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL MODIFIED TO SIMULATE
A LOAD REJECTION, WITH A CHOICE OF DD OR TD GOVERNOR.
AN OPTION WHICH PERMITS THE SERVO SET POINT AND LOAD LIMIT RAMPS
TO BE DISABLED HAS ALSO BEEN INCLUDED, AND A RATE LIMIT ON
DESIRED SERVO POSITION HAS BEEN IMPLEMENTED.
VERSION 9

DATE 24-JUN-81      TIME 20:50:02      RUN NUMBER    4

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =   0.9500000
TA =    8.700000
RC =  -0.2500000       RO =   0.3000000E-01   FS =    1.000000
TS =   0.1000000
GS =    2.500000       TR =   0.2000000       RT =   0.3000000E-01
RU =   0.1000000E-01   RM =   0.2500000E-01
Q1 =   0.7000000       Q2 =   0.9000000       Q4 =    1.100000
Q3 =    2.000000       Q5 =    1.100000
FD =    1.230000       AS =   0.1100000E-02   LL2 =   0.2400000
RL2 =   0.2000000E-01  YNL =   0.1200000
F1 =   0.8200000E-01   E1 =    1.584000       E2 =   0.5840000
A1 =   0.6480000E-01   A2 =   0.1561000       A3 =    1.929000
A4 =   -1.166000       A5 =    1.253000       A6 =   -2.852000

ASK VARIABLES
PLMW =    13.50000
IGOV =    0
IRAMP =    0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01
FINISH TIME  0.350000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    4
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME TD142.OUT
STORED VARIABLES:-
  TIME       F          FT         YSA        YD         Y

POST-RUN PASS NUMBER    5
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME TD132.OUT
STORED VARIABLES:-
  TIME       FT         Z1         YG         HT         YR
```

Site Test:- SLC304

Date:- 19th February 1980

Reservoir Level :- 930ft

Effective Head:- 903ft

Governor Type:- Microprocessor TD

Description:- Load rejection test; initial load approximately 33MW.

```
HYDRO SET - LOAD REJECTION SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL MODIFIED TO SIMULATE
A LOAD REJECTION, WITH A CHOICE OF DD OR TD GOVERNOR.
AN OPTION WHICH PERMITS THE SERVO SET POINT AND LOAD LIMIT RAMPS
TO BE DISABLED HAS ALSO BEEN INCLUDED, AND A RATE LIMIT ON
DESIRED SERVO POSITION HAS BEEN IMPLEMENTED.
VERSION 10

DATE 04-AUG-81      TIME 19:38:52      RUN NUMBER   15

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =    1.000000
TA =    8.700000
RC =  -0.2500000       RO =   0.3000000E-01   FS =   0.9990000
TS =   0.1000000
GS =    2.500000       TR =   0.2000000       RT =   0.3000000E-01
RU =   0.1000000E-01   RM =   0.2500000E-01
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.100000
Q3 =    2.000000       Q5 =    1.100000
FD =    1.230000       AS =   0.1100000E-02   LL2 =   0.2400000
RL2 =   0.2000000E-01  YNL =   0.9500000E-01
F1 =   0.9200000E-01   E1 =    1.584000       E2 =   0.5840000
A1 =   0.6480000E-01   A2 =   0.1561000       A3 =    1.929000
A4 =   -1.166000       A5 =    1.253000       A6 =   -2.852000

ASK VARIABLES
PLMW =    30.50000
FIC =    1.002000
IGOV =    0
IRAMP =    0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01
FINISH TIME  0.700000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

POST-RUN PASS NUMBER    4
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME TD322.OUT
STORED VARIABLES:-
  TIME       F          FT         YSA        YD         Y

POST-RUN PASS NUMBER    5
STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME TD323.OUT
STORED VARIABLES:-
  TIME       FT         YSA        YG         YD         Y
```

Site Test:- SLC123

Date:- 10th January 1980

Reservoir Level :- 935ft

Effective Head:- 907ft

Governor Type:- Microprocessor TD (Faulty)

Description:- Load rejection test; initial load approximately 20MW.

```
HYDRO SET - RUN UP SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL
MODIFIED TO SIMULATE THE RUN UP CHARACTERISTIC.
THE RELIEF VALVE HAS BEEN OMITTED FROM THIS MODEL.
VERSION 8

DATE 06-AUG-81      TIME 19:11:04      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =    1.000000
TA =    8.700000       T3 =    1.018000       T4 =   0.2000000
RC =  -0.2500000       RO =   0.3000000E-01   K1 =    14.99000
K2 =    0.0000000      TL =    158.2900
GS =    2.500000       TS =   0.1000000
Q1 =    0.7000000      Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000       Q6 =   0.9550000
Q7 =    0.9800000
FD =    1.230000       AS =   0.1100000E-02   LL1 =   0.2750000
LL2 =   0.2250000      RL1 =   0.1900000E-01   RL2 =   0.1900000E-01
QL =   0.8800000E-01   F1 =   0.1100000       E1 =    1.584000
E2 =    0.5840000      YNL =   0.9500000E-01   FPB =   0.5000000
A1 =   0.6480000E-01   A2 =   0.1561000       A3 =    1.929000
A4 =   -1.166000       A5 =    1.253000       A6 =   -2.852000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.850000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME R4A13.OUT
STORED VARIABLES:-

 TIME        F          FT         YSA        YD         Y
```

Appendix 3.14, Figures 7.14, 7.18, 7.19, 8.7

Site Test:- SLT403

Date:- 30th August 1979

Reservoir Level :- 883ft

Effective Head:- 858ft

Governor Type:- Microprocessor DD

Description:- Simulated isolated load step tests using ILS

Mean power at start of test	25MW
ILS Inertia Constant	7.0s
ILS Load Self-regulation constant (k_n)	0.0
ILS disturbance (% of 32.5MW)	$\pm$5%

Appendix 3.15, Figures 7.15, 7.16

Site Test:- SLT404

Date:- 30th August 1979

Reservoir Level :- 883ft

Effective Head:- 858ft

Governor Type:- Microprocessor DD

Description:- Simulated isolated load step tests using ILS

Mean power at start of test	25MW
ILS Inertia Constant	7.0s
ILS Load Self-regulation constant (k_n)	0.5, 1.0
ILS disturbance (% of 32.5MW)	$\pm$5%

Appendix 3.16, Figure 7.17

Site Test:- SLT406

Date:- 30th August 1979

Reservoir Level :- 883ft

Effective Head:- 858ft

Governor Type:- Microprocessor DD

Description:- Simulated isolated load step tests using ILS

Mean power at start of test	5MW
ILS Inertia Constant	7.0s
ILS Load Self-regulation constant (k_n)	0.0, 0.5, 1.0
ILS disturbance (% of 32.5MW)	$\pm$5%

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4
VERSION 5

DATE 08-SEP-81      TIME 19:06:23      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 = -0.5500000E-02    P2 =  0.5500000E-02    HR =   0.9800000
TA =    6.400000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       RO =  0.3000000E-01    K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =   0.1000000       BP =  0.3400000E-01
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =  0.1100000E-02    YNL =   0.1200000
KN =   0.0000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =  0.5980000E-01    A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =    24.00000     STP =   0.5200000E-01
TDEL =   0.0000000
IEUL =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

PRINTING SELECTED
PRINT INTERVAL  0.500000E+00
PRINT VARIABLES:-

  TIME       F

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIL54.OUT
STORED VARIABLES:-

  TIME       F          YD         Y          YC         FE
```

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4
VERSION 5

DATE 10-SEP-81      TIME 18:44:38      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.6000000E-02   P2 =   0.6000000E-02   HR =    1.000000
TA =    6.450000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       RO =   0.3000000E-01   K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =   0.1000000       BP =   0.3400000E-01
FS =    1.001000
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =   0.1100000E-02   YNL =   0.1200000
KN =   0.5000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =   0.5980000E-01   A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =    22.00000     STP =   0.5000000E-01
TDEL =   0.0000000
IEUL =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

PRINTING SELECTED
PRINT INTERVAL  0.500000E+00
PRINT VARIABLES:-

  TIME       F

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIL56.OUT
STORED VARIABLES:-

  TIME       F          YD         Y          YC         FE
```

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4
VERSION 5

DATE 10-SEP-81      TIME 19:46:45      RUN NUMBER    8

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 = -0.5500000E-02    P2 =  0.5500000E-02    HR =    1.000000
TA =    6.450000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       RO =  0.3000000E-01    K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =   0.1000000       BP =  0.3400000E-01
FS =    1.001000
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =  0.1100000E-02    YNL =   0.1200000
KN =    1.000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =  0.5980000E-01    A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =    22.00000     STP =  0.5700000E-01
TDEL =   0.0000000
IEUL =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

PRINTING SELECTED
PRINT INTERVAL  0.500000E+00
PRINT VARIABLES:-

  TIME       F

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    8

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIL57.OUT
STORED VARIABLES:-

  TIME       F          YD         Y          YC         FE
```

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4
VERSION 5

DATE 12-SEP-81      TIME 12:08:11      RUN NUMBER    6

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.1000000E-01   P2 =   0.1000000E-01   HR =    1.020000
TA =    6.450000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       RO =   0.3000000E-01   K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =   0.1000000       BP =   0.3400000E-01
FS =    1.001000
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =   0.1100000E-02   YNL =   0.1200000
KN =   0.0000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =   0.5980000E-01   A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =    3.500000     STP =   0.4900000E-01
TDEL =   0.0000000
IEUL =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

PRINTING SELECTED
PRINT INTERVAL  0.500000E+00
PRINT VARIABLES:-

  TIME       F

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    6

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIL58.OUT
STORED VARIABLES:-

  TIME       F          YD         Y          YC         FE
```

```
HYDRO SET - ISOLATED LOAD SIMULATION

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR SIMULATING ISOLATED LOAD STEP TESTS
AND LIMIT CYCLING. THE RELIEF VALVE
HAS BEEN OMITTED FROM THIS MODEL.

DATE 08-SEP-81      TIME 19:50:31      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =     2.710000       FA =    20.24000       KB =    1.460000
FB =     1.900000
KC =     2.410000       FC =    1.150000
BT =    0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.5500000E-02    P2 =  0.5000000E-02    HR =   0.9800000
TA =     6.450000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000        RO =  0.3000000E-01    K1 =    3.000000
K2 =     2.300000       TL =    33.30000
GS =     3.000000       TS =   0.1000000       BP =  0.3400000E-01
Q1 =    0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =     1.125000       Q5 =    1.095000
FD =     1.230000       AS =  0.1100000E-02    YNL =   0.1200000
F1 =    0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =  0.5980000E-01     A2 =   0.2449000       A3 =    1.707000
A4 =    -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =     24.00000     STP =   0.5200000E-01

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

PRINTING SELECTED
PRINT INTERVAL  0.500000E+00
PRINT VARIABLES:-

  TIME       F

STORAGE SELECTED
FILENAME SIM3.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER     1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME IL4.OUT
STORED VARIABLES:-

  TIME       F          YD         Y          YC         FE
```

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
A SECTION HAS BEEN INCLUDED TO PERMIT
THE USE OF SITE DATA FOR CERTAIN SIGNALS.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4.
VERSION 2

DATE 22-AUG-81      TIME 11:41:43      RUN NUMBER   10

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   HR =   0.9800000
TA =    6.500000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       R0 =   0.3000000E-01   K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =   0.1000000       BP =   0.3300000E-01
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =   0.1100000E-02   YNL =   0.1200000
KN =   0.0000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =   0.5980000E-01   A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =    24.35000     STP =   0.4530000E-01  STIM =    26.10000
TDEL =   0.0000000
ISITE =     1
IEUL =      1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.650000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

TERMINAL SECTION INCLUDED

POST-RUN PASS NUMBER     7

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIT8.OUT
STORED VARIABLES:-

  TIME        F          YD         Y          YC         FE
```

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
A SECTION HAS BEEN INCLUDED TO PERMIT
THE USE OF SITE DATA FOR CERTAIN SIGNALS.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4.
VERSION 2

DATE 19-AUG-81      TIME 19:07:49      RUN NUMBER    5

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.6000000E-02   P2 =  0.6000000E-02   HR =   0.9800000
TA =    7.000000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       RO =  0.3000000E-01    K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =  0.5000000E-01    BP =  0.3300000E-01
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =  0.1100000E-02    YNL =   0.1200000
KN =   0.0000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =  0.5980000E-01    A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =     24.00000    STP =   0.4700000E-01  STIM =     26.10000
TDEL =   0.0000000
ISITE =     3
IEUL =      1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.650000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

TERMINAL SECTION INCLUDED

POST-RUN PASS NUMBER    5

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIT9.OUT
STORED VARIABLES:-

  TIME        F           YD          Y           YC          FE
```

```
HYDRO SET - SIMULATED ISOLATED LOAD

THIS IS THE STANDARD HYDRO GENERATOR MODEL
USED FOR MODELLING SIMULATED ISOLATED LOAD
STEP TESTS AND LIMIT CYCLING. THE RELIEF
VALVE HAS BEEN OMITTED FROM THIS MODEL.
A SECTION HAS BEEN INCLUDED TO PERMIT
THE USE OF SITE DATA FOR CERTAIN SIGNALS.
THE GOVERNOR EQUATIONS USE EULER, NOT RK4.
VERSION 2

DATE 22-AUG-81      TIME 14:10:09      RUN NUMBER   23

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
KA =    2.710000       FA =    20.24000       KB =    1.460000
FB =    1.900000
KC =    2.410000       FC =    1.150000
BT =   0.2500000       TD =    16.00000       TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.6000000E-02   P2 =   0.6000000E-02   HR =   0.9800000
TA =    6.000000       T3 =   0.3000000       T4 =   0.3000000
RC =  -0.2500000       RO =   0.3000000E-01   K1 =    3.000000
K2 =    2.300000       TL =    33.30000
GS =    3.000000       TS =   0.1000000       BP =   0.3300000E-01
Q1 =   0.8750000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.125000       Q5 =    1.095000
FD =    1.230000       AS =   0.1100000E-02   YNL =   0.1200000
KN =   0.0000000
F1 =   0.1100000       E1 =    1.584000       E2 =   0.5840000
A1 =   0.5980000E-01   A2 =   0.2449000       A3 =    1.707000
A4 =   -1.002000       A5 =    1.256000       A6 =   -2.685000

ASK VARIABLES
PLMW =    20.95000     STP =   0.4530000E-01  STIM =    26.10000
TDEL =   0.0000000
ISITE =    2
IEUL =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.650000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.200000E+00

TERMINAL SECTION INCLUDED

POST-RUN PASS NUMBER   18

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME SIT10.OUT
STORED VARIABLES:-

  TIME        F          YD         Y          YC         FE
```

```
FREQUENCY TRANSIENT FOR RAPID RESPONSE SYSTEM

DATE 25-SEP-80      TIME 12:10:53      RUN NUMBER    2

UPDATE VARIABLES
WNA =    3.000000      WNB =   0.8000000      DA =   0.9000000E-01
DB =   0.2800000
STP =   0.1000000E-02  RR =  -0.6200000E-05
PA =  -0.8000000       PB =   -1.000000       PC =    1.000000
PD =   0.0000000       PE =   0.0000000       PF =   0.0000000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.100000E+00

FINISH TIME  0.150000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRSIP1.OUT
STORED VARIABLES:-

  TIME        YA          YB          YC          YD
```

```
FREQUENCY TRANSIENT FOR RAPID RESPONSE SYSTEM

DATE 25-SEP-80      TIME 12:14:06      RUN NUMBER    3

UPDATE VARIABLES
WNA =    3.000000      WNB =   0.8000000      DA =   0.9000000E-01
DB =   0.2800000
STP =   0.1000000E-02  RR =  -0.6200000E-05
PA =   0.0000000       PB =   0.0000000       PC =   0.0000000
PD =    1.000000       PE =   0.0000000       PF =    1.000000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.100000E+00

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    5

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRSIP2.OUT
STORED VARIABLES:-

  TIME       YDOTA      YD
```

```
FREQUENCY TRANSIENT FOR RAPID RESPONSE SYSTEM

DATE 25-SEP-80      TIME 12:22:46      RUN NUMBER    5

UPDATE VARIABLES
WNA =    3.000000      WNB =   0.8000000      DA =   0.5000000
DB =   0.2800000
STP =   0.1000000E-02  RR =  -0.6200000E-05
PA =   0.0000000       PB =   0.0000000       PC =   0.0000000
PD =    1.000000       PE =   0.0000000       PF =    1.000000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.100000E+00

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    8

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRSIP3.OUT
STORED VARIABLES:-

  TIME       YDOTA      YD
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 13:56:59      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =   -0.2500000      UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =     10.00000      DRRS =   0.3000000E-01  TWO =     10.00000
WNA =     3.000000      WNB =    0.8000000      DA =    0.9000000E-01
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =   -0.8000000       PB =    -1.000000
PC =     1.000000       PD =    0.0000000       PE =    0.0000000
PF =    0.0000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.150000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR1.OUT
STORED VARIABLES:-

  TIME        FREQ        GOVOUT      DSP         GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 14:55:25      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =     10.00000      DRRS =    0.3000000E-01 TWO =     10.00000
WNA =     3.000000      WNB =    0.8000000      DA =    0.9000000E-01
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =   -0.8000000       PB =    -1.000000
PC =     1.000000       PD =    0.0000000       PE =    0.0000000
PF =    0.0000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.150000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER     3

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR2.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS      PGOV
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 15:13:33      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =     10.00000      DRRS =    0.3000000E-01 TWO =     10.00000
WNA =     3.000000      WNB =    0.8000000      DA =    0.9000000E-01
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =    0.0000000       PB =    0.0000000
PC =    0.0000000       PD =     1.000000       PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    5

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR3.OUT
STORED VARIABLES:-

  TIME        FREQ        GOVOUT      DSP         GATPOS      PGOV
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 15:55:57      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =     10.00000      DRRS =    0.3000000E-01 TWO =     10.00000
WNA =     3.000000      WNB =     0.8000000     DA =    0.9000000E-01
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =     0.0000000      PB =    0.0000000
PC =    0.0000000       PD =      1.000000      PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR4.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 16:12:36      RUN NUMBER    3

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =   -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000       SK =    0.1100000E-02
RRT1 =     10.00000      RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =    0.5000000      DRRS =    0.3000000E-01 TWO =     10.00000
WNA =     3.000000      WNB =    0.8000000      DA =    0.9000000E-01
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =    0.0000000       PB =    0.0000000
PC =    0.0000000       PD =     1.000000       PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    5

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR7.OUT
STORED VARIABLES:-

   TIME        FREQ        RROP2       DSP         GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 15:20:31      RUN NUMBER    3

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000       AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =   -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =      100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =     10.00000      DRRS =   0.3000000E-01  TWO =     10.00000
WNA =     3.000000      WNB =    0.8000000      DA =    0.5000000
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =    0.0000000       PB =    0.0000000
PC =    0.0000000       PD =     1.000000       PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    6

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR5.OUT
STORED VARIABLES:-

  TIME        FREQ        GOVOUT      DSP         GATPOS      PGOV
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 16:01:33      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000 .     UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =     0.1000000     RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000       SK =    0.1100000E-02
RRT1 =     10.00000      RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03    DF2 =    0.3000000E-03
PFF =     10.00000      DRRS =    0.3000000E-01 TWO =     10.00000
WNA =     3.000000      WNB =     0.8000000     DA =    0.5000000
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =    0.0000000       PB =    0.0000000
PC =    0.0000000       PD =     1.000000       PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    3

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR6.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .

DATE 25-SEP-80      TIME 16:20:16      RUN NUMBER    4

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =    0.0000000      DRRS =    0.3000000E-01 TWO =     10.00000
WNA =     3.000000      WNB =    0.8000000      DA =    0.5000000
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =    0.0000000       PB =    0.0000000
PC =    0.0000000       PD =     1.000000       PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    7

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR8.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT

DATE 25-SEP-80      TIME 16:54:54      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =   0.2000000       T3 =   0.2000000       AK =    2.860000
FA =    20.07000       BK =    1.580000       FB =    1.760000
CK =    2.450000       FC =    1.140000       E1 =    1.578000
E2 =   0.5780000       E3 =    2.360000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000       AK1 =    3.000000      AK2 =    2.300000
TL =    33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   AH =    1.000000
TA =    7.000000       T4 =   0.2000000
ALL =  -0.2500000      UL =   0.3000000E-01   RLB =   -1.000000
RLT =    1.000000
GS =    5.000000       TRV =   0.1000000      RRV =   0.1000000E-01
NLL =   0.1000000
Q1 =   0.8000000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.200000       Q5 =    1.095000
FRS =   0.3720000      SK =   0.1100000E-02
RRT1 =    10.00000     RRT2 =    100.0000     RRT3 =    20.00000
DF1 =  -0.3000000E-03  DF2 =   0.3000000E-03
PFF =   0.5000000      DRRS =   0.3000000E-01 TWO =    10.00000
TOUT =    1.500000
WNA =    3.000000      WNB =   0.8000000      DA =   0.9000000E-01
DB =   0.2800000
FTRR =  -0.6200000E-05 PA =   0.0000000       PB =   0.0000000
PC =   0.0000000       PD =    1.000000       PE =   0.0000000
PF =    1.000000

ASK VARIABLES
EOPP =   0.5000000     STP =   0.1000000E-02
IRR =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR9.OUT
STORED VARIABLES:-

 TIME       FREQ       GOVOUT     RROP2      DSP        GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT

DATE 25-SEP-80      TIME 17:03:53      RUN NUMBER    3

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =   -0.7000000E-02   P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000      RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =    0.5000000      DRRS =    0.3000000E-01 TWO =     10.00000
TOUT =     1.500000
WNA =     3.000000      WNB =    0.8000000       DA =    0.5000000
DB =    0.2800000
FTRR =  -0.6200000E-05  PA =    0.0000000       PB =    0.0000000
PC =    0.0000000       PD =     1.000000       PE =    0.0000000
PF =     1.000000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    6

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR10.OUT
STORED VARIABLES:-

 TIME       FREQ       GOVOUT      RROP2       DSP         GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
A FREQUENCY TRANSIENT SIMULATION IS USED TO STUDY THE
RESPONSE OF A "FREQUENCY DISTURBANCE RELAY" .
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT

DATE 25-SEP-80      TIME 17:31:14      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =   0.2000000       T3 =   0.2000000       AK =    2.860000
FA =    20.07000       BK =    1.580000       FB =    1.760000
CK =    2.450000       FC =    1.140000       E1 =    1.578000
E2 =   0.5780000       E3 =    2.360000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000       AK1 =    3.000000      AK2 =    2.300000
TL =    33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   AH =    1.000000
TA =    7.000000       T4 =   0.2000000
ALL =  -0.2500000      UL =   0.3000000E-01   RLB =   -1.000000
RLT =    1.000000
GS =    5.000000       TRV =    0.1000000     RRV =   0.1000000E-01
NLL =   0.1000000
Q1 =   0.8000000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.200000       Q5 =    1.095000
FRS =   0.3720000      SK =   0.1100000E-02
RRT1 =     10.00000    RRT2 =     100.0000    RRT3 =     20.00000
DF1 =  -0.3000000E-03  DF2 =   0.3000000E-03
PFF =   0.5000000      DRRS =   0.3000000E-01 TWO =     10.00000
TOUT =    1.500000
WNA =    3.000000      WNB =    0.8000000     DA =   0.9000000E-01
DB =   0.2800000
FTRR =  -0.6200000E-05 PA =  -0.8000000       PB =   -1.000000
PC =    1.000000       PD =   0.0000000       PE =   0.0000000
PF =   0.0000000

ASK VARIABLES
EOPP =   0.5000000     STP =   0.1000000E-02
IRR =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.150000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RR11.OUT
STORED VARIABLES:-

  TIME       FREQ       GOVOUT     RROP2      DSP        GATPOS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 10:18:57      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =    0.5000000      DRRS =    0.3000000E-01 TWO =     10.00000
TOUT =     1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL1.OUT
STORED VARIABLES:-

  TIME        FREQ        GOVOUT      DSP         GATPOS      GATHYS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 10:23:17      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =   0.3000000E-01    BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =   0.3000000E-01    RLB =   -1.000000
RLT =    1.000000
GS =     5.000000       TRV =    0.1000000      RRV =   0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =   0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =    0.5000000      DRRS =   0.3000000E-01  TWO =     10.00000
TOUT =    1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-02
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL2.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS      GATHYS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 10:29:09      RUN NUMBER    3

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =   0.3000000E-01    BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =   0.3000000E-01    RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =   0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =    0.5000000      DRRS =   0.3000000E-01  TWO =     10.00000
TOUT =     1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-01
IRR =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    3

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL3.OUT
STORED VARIABLES:-

  TIME        FREQ        GOVOUT      DSP         GATPOS      GATHYS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 10:33:56      RUN NUMBER    4

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =    0.7000000E-02   AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =   -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =    0.3000000E-03
PFF =    0.5000000      DRRS =    0.3000000E-01 TWO =     10.00000
TOUT =     1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000E-01
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    4

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL4.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS      GATHYS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 10:40:35      RUN NUMBER    5

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =   0.3000000E-01    BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =    0.3000000E-01   RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =    0.5000000      DRRS =    0.3000000E-01 TWO =     10.00000
TOUT =     1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000
IRR =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    5

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL5.OUT
STORED VARIABLES:-

   TIME        FREQ        GOVOUT      DSP         GATPOS      GATHYS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 10:49:41      RUN NUMBER    6

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       T3 =    0.2000000       AK =     2.860000
FA =     20.07000       BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =    0.3000000E-01   BT =    0.2500000       TD =     16.00000
TP =    0.3000000       AK1 =     3.000000      AK2 =     2.300000
TL =     33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =   0.3000000E-01    RLB =    -1.000000
RLT =     1.000000
GS =     5.000000       TRV =    0.1000000      RRV =    0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =    0.5000000      DRRS =   0.3000000E-01  TWO =     10.00000
TOUT =     1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =    0.1000000
IRR =     1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    6

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL6.OUT
STORED VARIABLES:-

  TIME        FREQ        RROP2       DSP         GATPOS      GATHYS
```

```
HYDRO SET - F.T.,R.V. AND RRS

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
CLOSED LOOP (ISOLATED LOAD) SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 02-OCT-80      TIME 19:41:53      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =   0.2000000       T3 =   0.2000000       AK =    2.860000
FA =    20.07000       BK =    1.580000       FB =    1.760000
CK =    2.450000       FC =    1.140000       E1 =    1.578000
E2 =   0.5780000       E3 =    2.360000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000       AK1 =    3.000000      AK2 =    2.300000
TL =    33.30000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   AH =    1.000000
TA =    7.000000       T4 =   0.2000000
ALL =   -1.000000      UL =    1.000000       RLB =   -1.000000
RLT =    1.000000
GS =    5.000000       TRV =   0.1000000      RRV =   0.1000000E-01
NLL =   0.1000000
Q1 =   0.8000000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.200000       Q5 =    1.095000
FRS =   0.3720000      SK =   0.1100000E-02
RRT1 =    10.00000     RRT2 =     100.0000    RRT3 =    20.00000
DF1 =  -0.3000000E-03  DF2 =   0.3000000E-03
PFF =   0.5000000      DRRS =   0.3000000E-01 TWO =    10.00000
TOUT =    1.500000

ASK VARIABLES
EOPP =   0.5000000     STP =   0.1000000
IRR =    1

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.500000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRIL08.OUT
STORED VARIABLES:-

  TIME       FREQ       DSP        GATPOS     GATHYS     RVPOS
```

```
HYDRO SET - RRS FOR LOAD REJECTION

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
LOAD REJECTION SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 11:17:36      RUN NUMBER    1

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =   0.2000000       AK =    2.860000       FA =    20.07000
BK =    1.580000       FB =    1.760000
CK =    2.450000       FC =    1.140000       E1 =    1.578000
E2 =   0.5780000       E3 =    2.360000
BP =   0.3000000E-01   BT =   0.2500000       TD =    16.00000
TP =   0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02   P2 =   0.7000000E-02   AH =    1.000000
TA =    7.000000       T4 =   0.2000000
ALL =  -0.2500000      UL =   0.3000000E-01   RLB =   -1.000000
RLT =    1.000000
GS =    5.000000       TRV =   0.1000000      RRV =   0.1000000E-01
NLL =   0.1000000
Q1 =   0.8000000       Q2 =   0.9000000       Q4 =    1.095000
Q3 =    1.200000       Q5 =    1.095000
FRS =   0.3720000      SK =   0.1100000E-02
RRT1 =    10.00000     RRT2 =    100.0000     RRT3 =    20.00000
DF1 =  -0.3000000E-03  DF2 =   0.3000000E-03
PFF =   0.5000000      DRRS =   0.3000000E-01 TWO =    10.00000
TOUT =    1.500000

ASK VARIABLES
EOPP =   0.5000000     STP =  -0.5000000
IRR =    0
IGT =    0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.350000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRLR01.OUT
STORED VARIABLES:-

  TIME       FREQ       DSP        GATPOS     GATHYS     RVPOS
```

```
HYDRO SET - RRS FOR LOAD REJECTION

THREE PIPE SECTION HYDRO-TURBINE MODEL WITH RELIEF VALVE.
LOAD REJECTION SIMULATION
WITH FREQUENCY DISTURBANCE RELAY.
THIS FDR SYSTEM ESCAPES ON ZERO CROSSING
AND HAS A DELAYED OUTPUT.

DATE 26-SEP-80      TIME 11:25:18      RUN NUMBER    2

PARAMETERS, CONSTANTS AND INITIAL CONDITIONS
T2 =    0.2000000       AK =     2.860000       FA =     20.07000
BK =     1.580000       FB =     1.760000
CK =     2.450000       FC =     1.140000       E1 =     1.578000
E2 =    0.5780000       E3 =     2.360000
BP =   0.3000000E-01    BT =    0.2500000       TD =     16.00000
TP =    0.3000000

UPDATE VARIABLES
P1 =  -0.7000000E-02    P2 =   0.7000000E-02    AH =     1.000000
TA =     7.000000       T4 =    0.2000000
ALL =  -0.2500000       UL =   0.3000000E-01    RLB =   -1.000000
RLT =    1.000000
GS =     5.000000       TRV =    0.1000000      RRV =   0.1000000E-01
NLL =    0.1000000
Q1 =    0.8000000       Q2 =    0.9000000       Q4 =     1.095000
Q3 =     1.200000       Q5 =     1.095000
FRS =    0.3720000      SK =    0.1100000E-02
RRT1 =     10.00000     RRT2 =     100.0000     RRT3 =     20.00000
DF1 =  -0.3000000E-03   DF2 =   0.3000000E-03
PFF =    0.5000000      DRRS =   0.3000000E-01  TWO =     10.00000
TOUT =     1.500000

ASK VARIABLES
EOPP =    0.5000000     STP =   -0.5000000
IRR =     1
IGT =     0

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.500000E-01

FINISH TIME  0.350000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN PASS NUMBER    3

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME RRLR02.OUT
STORED VARIABLES:-

  TIME        FREQ        DSP         GATPOS      GATHYS      RVPOS
```

```
BASIC POWER SYSTEM MODEL VER 1

DATE 26-SEP-80      TIME 14:27:17      RUN NUMBER   10

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =  0.4000000E-01  AK2 =    20.00000
AK3 =   0.0000000      TC =    45.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02 B1H =    1.000000
TGVO =  0.7000000      TGVC =   0.1000000    ML =   0.2000000
MH =     1.200000      TD =    30.00000      EMH =    1.050000

ASK VARIABLES
TPIC =   0.5000000     DPE =   0.5000000
TS1 =    100.0000      TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.750000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+01

POST-RUN PASS NUMBER   12

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS2.OUT
STORED VARIABLES:-

  TIME        F          PB         PT         FI         QI
```

```
BASIC POWER SYSTEM MODEL VER 1

DATE 26-SEP-80      TIME 14:13:27      RUN NUMBER    9

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =   0.4000000E-01 AK2 =    20.00000
AK3 =   0.0000000      TC =    45.00000      TM =    2.000000
AH =    5.180000
G =    25.00000        AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02 B1H =    1.000000
TGVO =  0.7000000      TGVC =   0.1000000    ML =   0.2000000
MH =    1.200000       TD =    30.00000      EMH =    1.050000

ASK VARIABLES
TPIC =   0.8000000     DPE =   0.5000000E-01
TS1 =    100.0000      TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.750000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+01

POST-RUN PASS NUMBER   10

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS1.OUT
STORED VARIABLES:-

  TIME       F          PB         PT         FI         QI
```

```
POWER SYSTEM OPEN LOOP MODEL VER3

DATE 26-SEP-80      TIME 16:11:15      RUN NUMBER    2

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =   0.4000000E-01 AK2 =    20.00000
AK3 =    0.0000000     TC =    45.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.0000000
A1H =     10.00000     SSLFC =    4.000000
FDL =    0.0000000     FDH =    10.00000     FSOS =    1.040000
AIVD =    1.000000     B1L =   0.1000000E-02 B1H =    1.000000
TGVO =   0.7000000     TGVC =   0.1000000    ML =   0.0000000
MH =     10.00000      TD =    30.00000      EMH =    10.00000

ASK VARIABLES
GVIC =   0.5000000     DGV =   0.5000000
TS1 =    0.0000000     TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.600000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.400000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PSOL01.OUT
STORED VARIABLES:-

  TIME        PB          AM          EM          FI          FD
```

Appendix 3.51, Figure 9.6

POWER SYSTEM OPEN LOOP MODEL VER3

DATE 26-SEP-80 TIME 16:22:30 RUN NUMBER 4

UPDATE VARIABLES

FIC =	1.000000	TG =	0.1000000	AN =	1.000000
TIV =	0.1000000	TR =	10.00000	R =	0.3500000
PS =	1.000000	AK1 =	0.0000000	AK2 =	10.00000
AK3 =	0.0000000	TC =	45.00000	TM =	2.000000
AH =	5.180000				

Note: Remainder as Appendix 3.50

Appendix 3.52, Figure 9.6

POWER SYSTEM OPEN LOOP MODEL VER3

DATE 26-SEP-80 TIME 16:29:04 RUN NUMBER 5

UPDATE VARIABLES

FIC =	1.000000	TG =	0.1000000	AN =	1.000000
TIV =	0.1000000	TR =	10.00000	R =	0.3500000
PS =	1.000000	AK1 =	0.0000000	AK2 =	5.000000
AK3 =	0.0000000	TC =	45.00000	TM =	2.000000
AH =	5.180000				

Note: Remainder as Appendix 3.50

Appendix 3.53, Figure 9.6

POWER SYSTEM OPEN LOOP MODEL VER3

DATE 26-SEP-80 TIME 16:35:14 RUN NUMBER 6

UPDATE VARIABLES

FIC =	1.000000	TG =	0.1000000	AN =	1.000000
TIV =	0.1000000	TR =	10.00000	R =	0.3500000
PS =	1.000000	AK1 =	0.1500000E-01	AK2 =	5.000000
AK3 =	0.0000000	TC =	45.00000	TM =	2.000000
AH =	5.180000				

Note: Remainder as Appendix 3.50

```
POWER SYSTEM OPEN LOOP MODEL VER3

DATE 30-SEP-80      TIME 11:29:15      RUN NUMBER    3

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    2.000000      R =   0.3500000
PS =     1.000000      AK1 =   0.1000000E-01 AK2 =    5.000000
AK3 =   0.0000000      TC =    10.00000      TM =    2.000000
AH =    5.180000
G =    25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000     SSLFC =    4.000000
FDL =   0.1000000E-02 FDH =    1.050000      FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02 B1H =    1.000000
TGVO =  0.7000000      TGVC =   0.1000000    ML =   0.2000000
MH =    1.200000       TD =   0.0000000      EMH =    1.050000

ASK VARIABLES
GVIC =   0.5000000     DGV =   0.5000000
TS1 =   0.0000000      TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.500000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+01

POST-RUN PASS NUMBER    4

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PSOL07.OUT
STORED VARIABLES:-

  TIME        PB          PT          FI          FD          QI
```

```
POWER SYSTEM OPEN LOOP MODEL VER3

DATE 30-SEP-80      TIME 11:22:34      RUN NUMBER    2

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =  0.1500000E-01  AK2 =    5.000000
AK3 =    0.0000000     TC =    30.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =  0.1000000E-02  B1H =    1.000000
TGVO =   0.7000000     TGVC =   0.1000000    ML =   0.2000000
MH =     1.200000      TD =    30.00000      EMH =    1.050000

ASK VARIABLES
GVIC =   0.7000000     DGV =   0.3000000
TS1 =    80.00000      TB =    400.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.500000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+01

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PSOL08.OUT
STORED VARIABLES:-

  TIME        PB          PT          FI          FD          QI
```

```
POWER SYSTEM OPEN LOOP MODEL VER3

DATE 30-SEP-80      TIME 10:18:24      RUN NUMBER    1

UPDATE VARIABLES

FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    2.000000      R =   0.3500000
PS =     1.000000      AK1 =   0.1000000E-01 AK2 =    5.000000
AK3 =    0.0000000     TC =    10.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02 B1H =    1.000000
TGVO =   0.7000000     TGVC =   0.1000000    ML =   0.2000000
MH =     1.200000      TD =    0.0000000     EMH =    1.050000

ASK VARIABLES
GVIC =   1.000000      DGV =  -0.5000000
TS1 =    0.0000000     TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.100000E+00

FINISH TIME  0.250000E+02

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PSOL06.OUT
STORED VARIABLES:-

  TIME       PM1        PM2        PT         P1         P2
```

```
BASIC POWER SYSTEM MODEL VER 1

DATE 30-SEP-80      TIME 11:51:21      RUN NUMBER    2

UPDATE VARIABLES
FIC =    1.000000      TG =    0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    2.000000       R =   0.3500000
PS =     1.000000      AK1 =   0.1000000E-01  AK2 =   5.000000
AK3 =    0.0000000     TC =    45.00000       TM =    2.000000
AH =     10.00000
G =     25.00000       AK =    0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000       FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02  B1H =    1.000000
TGVO =   0.7000000     TGVC =   0.1000000      ML =   0.2000000
MH =     1.200000      TD =    0.0000000      EMH =    1.050000

ASK VARIABLES
TPIC =   0.8000000     DPE =    0.2000000
TS1 =    0.0000000     TB =     240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.500000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.400000E+00

POST-RUN PASS NUMBER    2

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS4.OUT
STORED VARIABLES:-

  TIME       F          PB         PT         A2         QI
```

```
BASIC POWER SYSTEM MODEL VER 1

DATE 29-SEP-80      TIME 14:50:03      RUN NUMBER    1

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =  0.1500000E-01  AK2 =    5.000000
AK3 =    0.0000000     TC =    45.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =  0.1000000E-02  B1H =    1.000000
TGVO =  0.7000000      TGVC =  0.1000000     ML =   0.2000000
MH =    1.200000       TD =    30.00000      EMH =    1.050000

ASK VARIABLES
TPIC =   0.8000000     DPE =   0.1000000
TS1 =    180.0000      TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.300000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.400000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER    1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS3.OUT
STORED VARIABLES:-

  TIME       F          PB         PT         A2
```

```
EXTENDED POWER SYSTEM MODEL VER 1.1

DATE 01-OCT-80       TIME 15:09:42       RUN NUMBER     1

UPDATE VARIABLES
FIC =     1.000000       TG =    0.1000000       AN =     1.000000
TIV =    0.1000000       TR =     10.00000       R =    0.3500000
PS =      1.000000       AK1 =   0.1500000E-01   AK2 =     5.000000
AK3 =     0.0000000      TC =     45.00000       TM =     2.000000
AH =     5.180000
G =     25.00000        AK =    0.2000000       A1L =    0.1000000E-02
A1H =     1.000000       SSLFC =     4.000000
FDL =    0.1000000E-02   FDH =     1.050000       FSOS =     1.040000
AIVD =     1.000000      B1L =    0.1000000E-02   B1H =     1.000000
TGVO =    0.7000000      TGVC =    0.1000000      ML =    0.2000000
MH =      1.200000       TD =     30.00000       EMH =     1.050000

ASK VARIABLES
POB1 =     22.22200      POB3 =     5.000000      TLMO =     25.00000
DPE =     0.4000000
RR =      1.000000
TS1 =      100.0000      TB =     240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.400000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.400000E+00

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER     1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS5.OUT
STORED VARIABLES:-

  TIME        F           PB          PT          PTT1        FI
```

```
EXTENDED POWER SYSTEM MODEL VER 1.1

DATE 30-SEP-80      TIME 14:43:09      RUN NUMBER    5

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =   0.1500000E-01 AK2 =    5.000000
AK3 =    0.0000000     TC =    45.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02 B1H =    1.000000
TGVO =   0.7000000     TGVC =   0.1000000    ML =   0.2000000
MH =     1.200000      TD =    30.00000      EMH =    1.050000

ASK VARIABLES
POB1 =    21.05000     POB3 =    5.000000    TLMO =    25.00000
DPE =   0.4000000
RR =     1.000000
TS1 =     100.0000     TB =     240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.400000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.400000E+00

POST-RUN PASS NUMBER    9

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS9.OUT
STORED VARIABLES:-

  TIME       F          PB         PT         FI         QI
```

```
EXTENDED POWER SYSTEM MODEL VER 1.1

DATE 01-OCT-80      TIME 15:45:13      RUN NUMBER    5

UPDATE VARIABLES
FIC =    1.000000      TG =   0.1000000      AN =    1.000000
TIV =   0.1000000      TR =    10.00000      R =   0.3500000
PS =     1.000000      AK1 =   0.1500000E-01 AK2 =    5.000000
AK3 =    0.0000000     TC =    45.00000      TM =    2.000000
AH =     5.180000
G =     25.00000       AK =   0.2000000      A1L =   0.1000000E-02
A1H =    1.000000      SSLFC =    4.000000
FDL =   0.1000000E-02  FDH =    1.050000     FSOS =    1.040000
AIVD =   1.000000      B1L =   0.1000000E-02 B1H =    1.000000
TGVO =  0.7000000      TGVC =   0.1000000    ML =   0.2000000
MH =     1.200000      TD =    30.00000      EMH =    1.050000

ASK VARIABLES
POB1 =    21.05000     POB3 =    5.000000    TLMO =    25.00000
DPE =   0.4000000
RR =   0.3500000
TS1 =    300.0000      TB =    240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.400000E+03

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.400000E+00

POST-RUN PASS NUMBER    5

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS13.OUT
STORED VARIABLES:-

  TIME       F          PB         PT         PTT1       FI
```

```
EXTENDED POWER SYSTEM MODEL VER 2.1

DATE 01-OCT-80       TIME 16:28:52       RUN NUMBER     1

UPDATE VARIABLES
FIC =     1.000000       TG =    0.1000000       AN =     1.000000
TIV =    0.1000000       TR =     10.00000       R =    0.3500000
PS =      1.000000       AK1 =    0.1500000E-01  AK2 =     5.000000
AK3 =    0.0000000       TC =     45.00000       TM =     2.000000
AH =     5.180000
G =     25.00000         AK =    0.2000000       A1L =    0.1000000E-02
A1H =     1.000000       SSLFC =     4.000000
FDL =    0.1000000E-02   FDH =     1.050000       FSOS =     1.040000
AIVD =     1.000000      B1L =    0.1000000E-02  B1H =     1.000000
TGVO =    0.7000000      TGVC =    0.1000000      ML =    0.2000000
MH =      1.200000       TD =     30.00000       EMH =     1.050000

ASK VARIABLES
POB1 =     22.22200      POB3 =     5.000000      TLMO =     25.00000
DPE =    0.4000000
RMP =   -0.1100000E-02
RR =    0.5000000
TS1 =     100.0000       TB =     240.0000

INTEGRATION METHOD  RUNGE-KUTTA (FOURTH ORDER)
INTEGRATION INTERVAL  0.200000E+00

FINISH TIME  0.100000E+04

STORAGE SELECTED
FILENAME SIM.DAT
STORAGE INTERVAL  0.100000E+01

POST-RUN OUTPUT SELECTED

POST-RUN PASS NUMBER     1

STORAGE FOR FURTHER PLOTTING SELECTED
FILENAME PS14.OUT
STORED VARIABLES:-

  TIME        F           PB          PT          PTT1        FI
```

REFERENCES

1. BRYCE G.W.,
An Investigation of Improved Designs for Water Turbine Governors,
Ph.D. Thesis, University of Glasgow, 1975.

2. BRYCE G.W., AGNEW P.W., FOORD T.R., WINNING D.J., MARSHALL A.G.,
On-site Investigation of Electrohydraulic Governors for Water Turbines,
Proc. IEE, 1977, **124**, (2), pp. 147-153.

3. FINDLAY D.G.E.,
Microprocessor Governors for Hydro-turbine Generators,
Ph.D. Thesis, University of Glasgow, 1980.

4. FINDLAY D.G.E., DAVIE H., FOORD T.R., MARSHALL A.G., WINNING D.J.
A Microprocessor-based Adaptive Hydro Turbine Governor,
IEE Proc. C, Gen., Trans., Distrib., 1980, **127**, (5), pp 360-369.

5. BRYCE G.W., FOORD T.R., MURRAY-SMITH D.J., AGNEW P.A.,
The Use of a Hybrid Computer Simulation in the Investigation of Water Turbine Governors,
Simulation Council Proceedings Series, 1976, **6**, (1), pp 35-44.

6. GRANT N.F.,
A Microprocessor Based Controller Applied to a Hydro-turbine,
Ph.D. Thesis, University of Glasgow, 1981.

7. GRANT N.F., AITKEN K.H., WINNING D.J., DAVIE H.,
Development of an Operational Hydro-turbine Governor,
Microprocessors and Microsystems, 1980, **4**, (9), pp 347-351.

8. FULTON A.A.,
The Cruachan Pumped-storage Development,
Electronics and Power, 1966, **12**, (7), pp 220-224.

9. MILLER D.J., MURRAY A.T.L., MARSHALL C.C., ARGENT G.G.R.,
Foyers Pumped-storage Project,
Proc. IEE, 1975, **122**, (11), pp 1222-1234.

10. Europe's Largest Pumped-storage Scheme Will Set New Records,
Water Power and Dam Construction, 1976, **28**, (3), pp 30-34.

11. HERNEN B.,
The World's Most Advanced Pumped-storage Scheme,
Electrical Review, 1977, **200**, (24), pp 29-31.

12. KERENSKY G., BEVERLY J.C., CHAPMAN E.J.K.,
The Loch Sloy Hydro-electric Development, Parts I, II and III,
Proc. I. Mech. E., 1955, **169**, pp 205-232.

13. Sloy Hydro-electric Scheme,
The North of Scotland Hydro-electric Board, Glasgow, 1950.

14. GALLACHER W.R.,
Analogue Simulation of the Dynamics of a Hydro-electric Generating System,
M.Sc. Thesis, University of Glasgow, 1968.

15. RUDQVIST O.,
Turbine Control: an Historical Survey,
Water Power and Dam Construction, 1976, 28, (9), p27-29.

16. TUSZYNISK J.,
Electrical Equipment for Turbine Governors,
Water Power and Dam Construction, 1976, 28, (9), p30-33.

17. DAVIE H., SCOBIE D.H.C., THOMPSON E.C.,
A Fortran-based Simulation Package with Real-time Capabilities,
Proc. of the United Kingdom Simulation Conference on Computer Simulation, 1975, pp A1.1-A1.7.

18. WINNING D.J., MARSHALL A.G., FINDLAY D.G.E., AITKEN K.H., GRANT N.G., Controller Testing Facility on 32.5MW Water Turbine,
IEE Proc. C, Gen., Trans., Distrib., 1980, 127, (5), pp 357-359.

19. System/360 Continuous System Modelling Program User's Guide
IBM, New York, 1972.

20. MACARTHER C.D.,
A User's Guide to CSMP,
University of Edinburgh, Program Library Services, No. 7.

21. WAITE W.M.,
Implementing Software for Non-numeric Applications
Prentice-Hall, New Jersey, 1973.

22. POOLE P.C., WAITE W.M.,
The STAGE2 Macroprocessor User Reference Manual
UKAEA Program Documentation Note CLM-PDN 6/70.

23. STRAUSS J.C. (Editor), AUGUSTIN D.C. (Chairman),
The SCi Continuous System Simulation Language (CSSL),
Simulation, 1967, 9, (6), pp 281-303.

24. Standards for Dynamic Simulation Languages
IEE Colloquium, 1980, Digest 1980/17.

25. RSX-11M Documentation,
Digital Equipment Corporation, Maynard, Mass.

26. RSX-11M Executive Reference Manual,
Digital Equipment Corporation, Maynard, Mass.

27. RSX-11M Task Builder Manual,
Digital Equipment Corporation, Maynard, Mass.

28. PDP-11 FORTRAN Language Reference Manual, FORTRAN IV User's Guide
Digital Equipment Corporation, Maynard, Mass.

29. KORN G.A., WAIT J.V.,
Digital Continuous Systems simulation,
Prentice-Hall, New Jersey, 1978.

30. HAY J.L.,
Interactive Simulation on Minicomputers: Part 1 - ISIS, a CSSL Language,
Simulation, 1978, 31, (1), pp 1-7.

31. PEARCE J.G.,
Interactive Simulation on Minicomputers: Part 2 - Implementation of the ISIS Language,
Simualtion, 1978, **31**, (2), pp 43-53.

32. MITCHELL E.E.L., GAUTHIER J.S.,
Advanced Continuous Simulation Language (ASCL),
Simulation, 1976, **26**, (3), pp 72-78.

33. WOODWARD J.L.,
Hydraulic-turbine Transfer Function for use in Governing Studies,
Corr., Proc. IEE, 1968, **115**, (3), pp 424-426.

34. AGNEW P.A.,
The Governing of Francis Turbines,
Water Power and Dam Construction, 1974, 26 (4), pp 119-127.

35. WOOD D.J.,
Waterhammer Analysis by Analog Computers,
Journal of the Hydraulics Division of the American Society of Civil Engineers, 1967, **93**, (HY1).

36. STREETER V.L.,
Waterhammer Analysis of Pipelines,
Journal of the Hydraulics Division of the American Society of Civil Engineers, 1964, **90**, (HY4), pp 151-172.

37. WYLIE E.B.,
Resonance in Pressurized Piping Systems,
Journal of Basic Engineering, Trans. of the American Society of Mechanical Engineers, 1965, **87**, pp 950-966.

38. JARVIS R.M., ARMSTRONG N.A.,
Some Experiences with Auto-resonance in Conduits of Hydro-electric Schemes,
Paper presented at the International Conference on Pump and Turbine Design and Development, September, 1976, at NEL, East Kilbride, Scotland

39. WYLIE E.B., STREETER V.L.,
Fluid Transients,
McGraw-Hill International Book Co., New York, 1978.

40. STREETER V.L. (Editor),
Handbook of Fluid Dynamics,
McGraw-Hill Book Co. Inc., New York, 1961.

41. PRANDTL L.,
Essentials of Fluid Dynamics,
Blackie and Son Ltd., Glasgow, 1952.

42. GRIMSON J.,
Advanced Fluid Mechanics and Heat Transfer,
McGraw-Hill Book Co. Ltd., Maidenhead, 1971.

43. SCHLEIF F.R., BATES C.G.,
Governing Characteristics for 820,000 Horsepower Units for Grand Coulee Third Power Plant,
IEEE Trans. PAS, 1971, **90**, (2), pp 822-890.

44. SCHLEIF F.R., EILTS L.E.,
Governing Features and Performance of the First 600MW Hydrogenerating Unit at Grand Coulee,
IEEE Trans. PAS, 1977, **96**, (2), pp 457-466.

45. SCHLEIF F.R., ANGELL R.R,
Governor Tests by Simulated Isolation of Hydraulic Turbine Units,
IEEE Trans. PAS, 1968, **87**, (5), pp 1263-1269.

46. CAUSON G.J.,
Governoring a Hydro-electric System,
International Association for Hydraulic Research, Symposium, Vienna, 1974, pp X2.1-X2.13.

47. RSX11-M MCR Operating Manual,
Digitial Equipment Corporation, Maynard, Mass.

48. WINNING D.J.,
Rapid Response System (Enhancement for Existing Temporary Droop Speed Governors),
Glasgow University, Department of Electronics and Electrical Engineering, 1975.

49. ASHMOLE P.H. et al.,
Power System Model for Large Frequency Disturbances,
Proc. IEE, 1974, **121**, (7) pp 601-608.

50. ASTROM K.J., EKLUND K.,
A Simplified Non-linear Model of a Drum Boiler-turbine Unit,
Int. J. Control, 1972, **16**, (1), pp 145-169.

51. THOMPSON E.C.,
A Digital Simulation of a Boiler and Turbine in Conjunction with a Model Power System,
Ph.D. Thesis, University of Glasgow, 1976.

GUILDS

Glasgow University

Interactive Language for Dynamic Simulation

GUILDS

GLASGOW UNIVERSITY

INTERACTIVE LANGUAGE FOR DYNAMIC SIMULATION

K.H. Aitken D.J. Winning

Version 1.0

5-January-81

University of Glasgow
Department of Electronics
and Electrical Engineering.

CONTENTS

1. INTRODUCTION

GUILDS, the Glasgow University Interactive Language for Dynamic Simulation, has been developed in the Department of Electronics and Electrical Engineering at the University of Glasgow as a powerful but easily used means of studying the behaviour of continuous dynamic systems - systems which can be described by a series of algebraic and first order ordinary differential equations. The language permits the user to start from a mathematical model of the system in either block diagram or in differential equation form and from this a series of statements defining this model - the MODEL DESCRIPTION - are prepared following the language rules given in this guide. This definition is translated into an executable form - the SIMULATION MODULE - the translation process and subsequent execution of the simulation being under the control of the SIMULATION EXECUTIVE. By providing considerable user interaction, the executive permits users of varying degrees of experience in the use of the language to obtain appropriate assistance in running the simulation.

The language is structured in such a way that users in many disciplines should find it straightforward to prepare an appropriate MODEL DESRCRIPTION for their particular problem. Powerful features within the language permit each user to develop a library of functions suitable for his own problem and thus to represent large functional blocks of his problems by a single statement. See Section 3.4.

GUILDS is at present implemented on a PDP11/45 minicomputer and on a Z80/8080-based microcomputer with a CP/M operating system. Partial implementations on other computers have been undertaken in the past but have not been fully updated. The structure of GUILDS is such that it is possible to implement it fairly quickly on most computer systems.

An example of the simulation of a relatively simple system is given in Appendix A1 and this illustrates the ease with which the package may be used. Many of the features are better observed when GUILDS is used and thus early interaction with the computer is recommended for prospective users.

2. OVERVIEW OF GUILDS

2.1 Computational techniques

GUILDS was developed from a FORTRAN-based simulation package to which was added a pre-processor which uses a general-purpose macroprocessor (or text processor) - STAGE2B[1] - to translate the user's MODEL DESCRIPTION into a form suitable for solution by the package.

The main function of a continuous system language like GUILDS is the solution of a set of first order differential equations. To facilitate understanding of the remainder of this guide, the method of solution is briefly outlined. The differential equations are either stated explicitly in the MODEL DESCRIPTION or are generated by the pre-processor from other GUILDS statements. These equations, in effect, define the derivatives of a number of the variables in the MODEL DESCRIPTION called the "state" or "integrated" variables and these derivatives, together with an "integration method" provided by GUILDS, are used to calculate the values of the state variables at each discrete interval of the independent variable - usually TIME. The algebraic equations in the MODEL DESCRIPTION are then used to calculate the values of other variables at each interval.

The way in which this is done can be illustrated using the simplest integration method provided by GUILDS (the first-order Euler method). In this, the value of the variable x at time t+h, x(t+h), (where h is a small increment in t, the independent variable) is given by:

$$x(t+h) = x(t) + \frac{dx(t)}{dt} * h$$

This solution is of course approximate but for many systems, provided h is chosen to be small enough, the solution is of adequate accuracy. In general, h should be chosen to be less than the shortest time constant of the system being modelled but where doubt exists, the interactive facilities of GUILDS should enable the user to arrive at an appropriate value rapidly.

As can be seen, the value of x at t+h is dependent upon the value of x and of its derivative at t. Thus, if a numerical value for x(t) is known and if a value for dx(t)/dt can be computed, then the value of x at the next time interval, t+h, can be evaluated using the expression above.

GUILDS requires the user to specify the value of x at the beginning of the simulation (when t=0) and to provide an expression or series of expressions within the MODEL DESCRIPTION from which the derivative can be calculated. At each time interval, a routine, generated from the user's MODEL DESCRIPTION is called which, given the current value of x(t), evaluates the derivative dx(t)/dt. From this, the integration routine computes the value of x at the next time interval t+h, and this in turn is passed to the user's routine which recalculates the derivative at time t+h. The cycle is then repeated until the user-specified simulation time has expired.

For systems with more than one integrated variable, the user's MODEL DESCRIPTION must contain expressions from which all the derivatives can be evaluated given the current values of the integrated variables. The integration routine then computes new values for all these variables as before.

The user's MODEL DESCRIPTION must thus be written in, or translated by GUILDS into, such a form as will permit these derivatives to be computed. Thus, all the variables used in the definition of the derivatives must themselves be defined by the user. If the "sorting" facility of GUILDS (see Section 3.3) is used this constraint is sufficient, but if the user does not wish to use the automatic sorting it is also necessary to ensure, as in normal computer programming practice, that the statements appear in the correct computational sequence.

In the more complex integration methods provided by GUILDS, exactly the same use is made by GUILDS of the user's MODEL DESCRIPTION which is still required to calculate the values of the derivatives given the current values of all the state variables. These more complex methods permit the use of a larger time increment and in some circumstances permit the solution of a system which cannot be solved at all by the Euler method (due to, for example, numerical instability which can manifest itself as system instability in circumstances where instability is not expected). The method most suitable for a given system has, in general to be found by experimentaion with GUILDS.

2.2 Language overview

Where possible, GUILDS has been based on the CSSL report[2] and in the places where this is inexplicit, on the IBM language CSMP[3]. Because the user's MODEL DESCRIPTION is translated into FORTRAN statements, FORTRAN conventions for numeric constants, variable names (names beginning with letters I to N inclusive are INTEGER, all others are REAL unless otherwise explicitly typed; see Section 3.2.6) and arithmetic and logical operators are used.

It is necessary for the user to specify not only the statements of the MODEL DESCRIPTION but also to supply information for the control of the simulation run such as the definition of the integration method and the type of output. In batch-oriented simulation languages such as CSMP, both of these types of information are specified as part of the problem input and while this is adequate for batch-oriented systems it is unsuitable for systems in which interaction is possible since it does not permit the advantages of interaction to be utilised. In GUILDS, most of the control information is supplied by the user at run-time through an interactive dialogue, although the MODEL DESCRIPTION is still translated into a routine which is compiled as in, for example, CSMP. GUILDS has been optimised in this way to provide the user with flexible control of the simulation at run-time along with the advantage of fast execution, inherent in a compiled system.

2.3 Implementation overview - PDP11

GUILDS is implemented on a PDP11/45 computer running version 3.2 of the RSX11M operating system. Version 2 of FORTRAN IV is used, but, for users with a FORTRAN IV-PLUS compiler, an implementation using FORTRAN IV-PLUS is also available.

The user-written MODEL DESCRIPTION is created using the Text Editor and stored as a disk file. The SIMULATION EXECUTIVE controls the translation and execution phases by requesting from the user, decisions on which of the available options have to be used and calling appropriate system and GUILDS software as required.

Although from a user standpoint, the SIMULATION EXECUTIVE appears as an single entity, it consists in practice of two separate parts. The first of these is an "Indirect Command File" which calls up to three passes of STAGE2-based processing of the MODEL DESCRIPTION file and results in the creation of up to three FORTRAN subroutines. The FORTRAN compiler and the Task Builder are then called in turn by the Command File to compile these subroutines and link them with the remainder of the GUILDS subroutines to produce a SIMULATION MODULE, a Task Image File which also is stored on disk. The final action of the Command File is to start the execution of this Task.

Within this first part of the SIMULATION EXECUTIVE there are two levels of interaction, either of which can be selected by the user. LEVEL 1 is recommended for inexperienced users as very little interaction is required and the translation process proceeds almost automatically. This level does not, however, permit the use of a TERMINAL setion in the MODEL DESCRIPTION or of an external integration routine. LEVEL 2 asks the user if all the stages in the translation process are required and only executes those stages specified. Experienced users may find that time can be saved, for example if the MODEL DESCRIPTION does not need to be sorted, by using LEVEL 2. An example of each level of interaction is given in Appendix A (Figures A4 and A9).

The second part of the SIMULATION EXECUTIVE is an integral part of the SIMULATION MODULE and is responsible for communicating with the user on the control of the simulation run. Transition from the dialogue in the Command File to that in the SIMULATION MODULE is completely transparent to the user. An example of this dialogue, for the system given in Appendix A1, is shown in Figure A5. If a repeated run of an already created SIMULATION MODULE is required, entry to the second part of the SIMULATION EXECUTIVE can be obtained directly and this permits rapid access to freqently used simulations (Appendix A3).

2.4 Implementation overview - Z80/8080 System with CP/M

This section has yet to be finalised.

3.0 LANGUAGE DESCRIPTION

Unlike many other simulation languages, GUILDS requires, prior to run-time, only a description of the system to be simulated. This MODEL DESCRIPTION is made up of a series of statements which obey the rules of FORTRAN assignment (i.e. replacement) statements and may use functions provided by GUILDS, by FORTRAN and, if required, by the user. The library of functions provided by GUILDS is listed in Appendix C; the FORTRAN functions which may be used are those provided with the particular version of FORTRAN in which GUILDS is implemented. Users may supplement these in some cases by writing additional FORTRAN subprograms. Where complex functions are required which cannot be implemented in FORTRAN, the stucture of GUILDS is such that experienced users or the originators may be able to supplement GUILDS to incorporate these functions.

3.1 Structure of MODEL DESCRIPTION

The basis of a continuous simulation language is an integration routine to integrate the functions involved in the system over the range of the independent variable and a routine to recalculate the derivatives and other variables on which they depend at each discrete increment of the independent variable. The integration routine is contained within the simulation package and the other routine is one of the FORTRAN subroutines produced by translation of the users MODEL DESCRIPTION (see Section 2.1).

There may also be calculations which need to be performed before or after a simulation run but not during the simulation itself. Thus, the input file has been subdivided into three segments, as described below. The start of a segment is defined by a control statement (INITIAL, DYNAMIC or TERMINAL) and the end of that segment by the start of the next, or by the END statement which defines the end of the MODEL DESCRIPTION. The INITIAL and TERMINAL segments are optional but if included they should appear in the order shown preceded by the appropriate control statement.

3.1.1 INITIAL Segment

The initial segment contains those calculations which are to be carried out once at the start of a run and not again. If included, this segment should be the first in any MODEL DESCRIPTION but must be preceded by any TITLE or Data (except ASK) statements or MACRO definitions if these are present. Thus, for example, the initial conditions for the integrators could be calculated from values supplied by the user at run-time (see Section 3.2.2).

3.1.2 DYNAMIC Segment

The DYNAMIC segment contains the statements describing the system to be simulated and is readily developed from a block diagram or differential equation representation of the system. This segment, the only necessary one for a simulation, should follow the INITIAL segment if it is included.

3.1.3 TERMINAL Segment

The TERMINAL segment contains any calculations required only at the end of a simulation. It is possible to initiate a rerun of the simulation from the TERMINAL segment with, for example, new parameters which have been calculated within this segment. It can also be used for further analysis of data or further output. See Section 4.8.

3.1.4 Statement Ordering

There are certain restrictions on the order in which statements may appear in the MODEL DESCRIPTION as noted below:-

(i) A TITLE statement, if present, must be the first statement in the input file;

(ii) Macro Definitions must appear before all other statements, with the exception of the TITLE statement;

(iii) All Data Statements, except ASK, must appear before the INITIAL segment, or before the DYNAMIC segment if no INITIAL segment is present;

(iv) ASK statements can only be included in the INITIAL segment and must appear at the start of the segment;

(v) Statements bracketed by the dollar ($) symbol (see Section 3.2.4) can only be used in the INITIAL and DYNAMIC segments.

3.2 MODEL DESCRIPTION Statements

The system to be simulated is described by a series of structure, data and translation control statements - the MODEL DESCRIPTION. These three types of statements, along with some additional translation control symbols, are described in the following sections.

In writing a MODEL DESCRIPTION, users must not use for their own purposes, variables which are reserved by the system. A list of these variables is given in Appendix E and Sections 3.5 & 4.8 describe how certain of the system variables can be used in the user's MODEL DESCRIPTION. Except where indicated, the values of these variables should not be altered by the user.

All statements can begin in any column using spaces and tabs as required. Within statements, however, embedded spaces and tabs are only permitted where indicated in the following sections.

3.2.1 Structure Statements

Structure statements define the model being simulated by describing the functional relationships between the variables of the model. The statements can begin in any column and should not contain any embedded spaces but otherwise conform to rules for FORTRAN assignment statements. They should however, occupy only one line and may not be continued. If it is not possible to complete the statement on one line, the expression must be broken into two or more parts by the definition of auxilliary variables.

The output variable name appears on the left hand side of the equals sign and an expression on the right. This expression may be a single variable or constant, a function call with associated inputs, initial conditions and parameters, or a combination of constants, variables and function calls connected by operators.

It is recommended that variable names should only appear once on the left hand side of an expression in any segment of the input file to prevent duplication in COMMON blocks etc. However, if it is necessary to use the same variable more than once, for example, in different logical branches of the DYNAMIC segment, a second or subsequent occurance of the variable name is detected by the translator and the name is not included in a COMMON block a second time.

Only real variables are output by the printing, plotting and storage routines and thus, an integer variable appearing on the left hand side of an assignment statement, when passed to these routines, would cause an error. To avoid this, the simulation language generates a real variable equivalent to the integer variable which can then be used for output. Alternatively, if the integer variable is not required for output, the user can protect the variable from the translator by using the '@' symbol, although this is only permissible in a NOSORT section (see Section 3.2.4).

Under some circumstances it may be desirable to have a second or subsequent occurance of a variable name recognised by the translator, rather than the first. This can be achieved by protecting the first occurance of the variable name by using the '@' symbol, but again, only in a NOSORT section (see Section 3.2.4).

The functions available to the user are of three different forms as described below:-

(a) standard FORTRAN functions
(b) GUILDS functions written in FORTRAN (Appendix C1)
(c) GUILDS functions which have to be translated from the user-written form into either other GUILDS statements, FORTRAN statements or FORTRAN functions (Appendix C2).

The first two types obey the rules for FORTRAN functions and are used in exactly the same way. The third type of function has the additional restriction that, if used in an expression, it must be the rightmost part of that expression, for example:-

```
A=3.4+2*INTGRL(AIC,X)
```

is valid, whereas,

```
A=INTGRL(AIC,X)*5
```

is not acceptable.

Other FORTRAN executable statements, for example, logical and branching statements, may be used in GUILDS but such statements can only appear in NOSORT sections. See also Section 3.2.6.

The use of arrays in conjunction with the simulation language statements requires some care (see also Section 3.2.6). The following rules must be observed:-

(a) array elements can only be used in NOSORT sections;
(b) in an NOSORT section an array element on the left hand side must be protected by the use of either the '$' or '@' symbols as described in Section 3.2.4 but on the right hand side array elements can be used unprotected.

3.2.2 Data Statements

Data statements are used to assign numeric values to parameters, constants and initial conditions. Four types of data statement are available as illustrated below. These statements should appear between TITLE and INITIAL statements if they exist but in any case before the DYNAMIC segment. One space must be included after the statement label (e.g. PARAMETER) but none are permitted elsewhere.

```
PARAMETER   A=1.2,B=3.6
INCON   IC1=0.0,IC2=1.0
CONSTANT   X1=6.3,Y=4.9
UPDATE   Z1=1.,Z2=3.14,Z3=0.
```

The labels used for the first three statements are interchangeable and serve only to remind the user in what context the constant is being used. The statements may be continued into a second or subsequent line by finishing the previous line with three dots, for example:

```
PARAMETER   A=1.2,...
B=3.6, C=0.0,...
D=10.0
```

The UPDATE statement differs significantly from the other Data Statements in that the user is able to change the value of the variables at run time. UPDATE statements cannot be continued but multiple UPDATE statements, with up to ten variables in each, are permitted. At run time, if the update facility is selected, a list of the names of the variables in each UPDATE statement are printed and for any variable selected from the list, the current value is printed and a new value is requested. Integer variables cannot, at present, be used with the UPDATE statement.

Three further Data Statements are provided by the ASK facility as illustrated below. ASK statements must appear at the beginning of the INITIAL segment before any other statements.

```
ASK SET POWER, STEP SIZE/PE,DPE
ASK PARAMETER/X2,X3
IASK NO. OF TIMES/N
```

The ASK facility causes the program to interrogate the user at run time for the values of the specified parameters and in this respect is similar to the UPDATE statement. It differs from UPDATE in that no default value is specified for an ASK variable and the user is always prompted by an ASK statement. It can be used to enter data such as the operating point of a system which has no preferred value and reminds the user at run time that this parameter must be specified.

The ASK statement has two forms as shown above; in the first case the text "SET POWER, STEP SIZE" supplied by the user is printed on the user's terminal and the user has to enter values for the variables PE and DPE; in the second case no text is supplied by the user, the variable names X2, X3 will be printed and the user has to respond with values for these variables. In addition, if the variable for which the value being entered is an integer the IASK form of the statement must be used. In all cases a space must follow the label (ASK or IASK) and no spaces are permitted after PARAMETER or to the right of the slash (/). Where text is supplied by the user any format may be used with the exception that the slash character may not be included in the text.

3.2.3 Statements for Translation Control

The MODEL DESCRIPTION supplied by the user has to be translated by GUILDS into the appropriate FORTRAN subroutines and in order to accomplish this, a number of single word statements must be inserted among the Structure and Data statements to provide the translator with a basis for the control of the translation process. These statements are described in this section and in the next section some additional controls are described. The exact form of each word must be used (e.g. NOSORT may not be given as NO SORT) and no abbreviations are permitted.

INITIAL, DYNAMIC and TERMINAL

These statements which define the start of the INITIAL, DYNAMIC and TERMINAL segments of the MODEL DESCRIPTION, are more fully described in Section 3.1 but are included here for completeness.

MACRO, ENDMAC

The MACRO and ENDMAC statements are used to bracket a group of structure statements which define a macro. This facility allows the user to build larger functional blocks from the basic GUILDS and FORTRAN functions. Once defined, a macro can be used in a model as if it were a system function simply by using the name specified in the macro definition. This facility is further described in Section 3.4 - User Defined Functions.

SORT, NOSORT

The SORT and NOSORT statements define the beginning and end of a sorted section. They are more fully described in Section 3.3.

PROCEDURE, ENDPRO

The PROCEDURE and ENDPRO statements are used to bracket a group of structure statements which are not sorted internally but are treated as a single functional entity by the sorting algorithm. The procedure statements are moved as a block to a position which depends on the input and output variables declared in the PROCEDURE definition. (Note that the relative position of a NOSORT section is not altered with respect to the statements preceding and following it.) Further details of the procedure facility are contained in Section 3.4.2.

3.2.4 Other Translation Controls

Preceding a variable name with the '@' symbol forces the translator to ignore the variable, where otherwise it would be accepted by the translator for inclusion in COMMON blocks.

If, for any reason, it is necessary to protect a statement from the translator (that is, cause the statement to be output unchanged), this can be accomplished by making the first and last characters in the statement the '$' symbol. (This facility was originally included to protect FORTRAN non-assignment statements which can now be used unprotected.)

Neither of these translation controls may be used in a SORT section of the MODEL DESCRIPTION.

3.2.5 Additional Statements

TITLE

A TITLE statement, which if present must be the first in the input file, can be used to associate a name or description with a simulation program. The TITLE statement cannot be continued and thus can only be 72 characters long. The first 40 characters of the statement, following "TITLE ", are used as a title for graphical output. If a TITLE statement is not included the default title, SIMULATION PROGRAM, is used.

Comments

Any line in the input file which starts with an asterisk ('*') is taken as a comment statement and will appear in the output file as such. It should be noted that the translation, particularly the sorting algorithm, although not acting on the comment statements may move other statements relative to the comment statements thus causing them to appear out of place. Comment statements which appear in the MODEL DESCRIPTION before the INITIAL segment are included in each FORTRAN subroutine produced by the translator. Otherwise, the comment statements are only included in the subroutine corresponding to the segment in which the statement appears. If a TITLE statement is not included in the MODEL DESCRIPTION and a comment statement is the first statement in the input file, the comment is used as a title for the simulation.

In addition to the * comment facility, the user can comment individual assignment statements by the inclusion of a colon (:), followed by a comment, anywhere to the right of the statement and in the same line. Comments of this type remain attached to the specified statement through the macro expansion and sorting phases of the STAGE2 process and are translated by the final phase into FORTRAN comment statements, inserted immediately before the statement to which the comment was attached.

TERMINATE

The user can terminate the simulation when certain conditions of variables in the model have been met, before the finish time (FINTIM) has been reached. This is done by specifing the conditions to be met in a TERMINATE statement in the DYNAMIC segment of the model. The form of this statement is shown below:-

```
TERMINATE (logical expression)
```

where the logical expression is any valid FORTRAN logical statement, for example:

```
TERMINATE (X.GT.XMAX.OR.X.LT.XMIN)
```

3.2.6 FORTRAN non-asignment statements

As has been noted above, it is possible to use FORTRAN logical and branching statements in order to achieve special effects within a simulation. This might be done, for example, to permit the user to represent the whole or part of his system by two or more different structures. The user can then use an ASK statement to choose one structure at run-time and by the use of FORTRAN branching statements, execute only the statements of that structure. This method is less elegant but computationally more efficient than the alternative method which uses switching functions within GUILDS and does not resort to FORTRAN branching statements. Using switching functions, all the statements in the simulation are executed but only the outputs from the chosen structure are used elsewhere in the simulation.

Similarily, other FORTRAN executable statements, for example READ, WRITE and DO statements, can be included in the user's MODEL DESCRIPTION.

All such FORTRAN non-assignment statements must appear within a NOSORT section. Certain restrictions with respect to the FORTRAN statement labels which can be used are noted in Appendix E.

Some FORTRAN specification statements can also be included, without the use of the dollar, to permit communication with user written routines. These statements should appear at the start of the MODEL DESCRIPTION after the TITLE statement and before the INITIAL statement and will be included in subroutines INIT, MODEL and TERM if present. These statements may use the same free form as Structure Statements (need not begin in any particular column), but should otherwise conform to the rules of FORTRAN for these statements. The statements available are:-

```
DIMENSION
DOUBLE PRECISION
LOGICAL
REAL
INTEGER
COMMON
```

3.3 Sorting

The statements describing the model itself - those contained in DYNAMIC - must appear in a computationally satisfactory sequence and this may not be the sequence in which they are arranged by the user. The inputs to a statement are the variables on the right hand side and values for these must be available at the point where the statement is executed. This means that the variable must either be a constant - variables in Data Statements come into this category - or they must have been previously computed - variables which have previously appeared on the left hand side of an expression satisfy this constraint.

Since INTGRL and other similar Functions (those indicated in Appendix C2.1) are Predictive in nature, GUILDS supplies the user routine each time it is entered, with values for all the variables appearing on the left hand sides of these statements. The values of these variables are thus available to all statements in the DYNAMIC segment irrespective of the position of the INTGRL (and other similar) statements which apparently define these variables. INTGRL (and other similar) statements must however be placed like all other statements, at a position at which all the input variables are defined. It should be noted that, if INTGRL (and other similar) functions are used in expressions (see Section 3.2.1) instead of in separate statements then the value of the left hand side of the expression is not known at the start of the DYNAMIC segment.

If all the input variables to a statement are not available to the statement at the position allocated to it by the user, it must be placed later in the sequence at a point where they are available, using a process called "sorting". For small programs sorting can be readily done by the user but for larger models this can be a very time consuming process. Thus, a sorting algorithm is available which can carry out the necessary ordering of the statements.

If automatic sorting is requested then the default is that all of the DYNAMIC segment is sorted. This, however may be overridden by the use of the SORT and NOSORT statements which come before groups of statements which require to be sorted or not sorted respectively. This facility permits the use of, for example, logical branching in an unsorted section.

With some systems it may prove impossible to order the statements in such a way as to satisfy these constraints. One possible reason for this is the presence of an algebraic loop; this topic is fully discussed in the CSMP manual[3] and in CSMP the loop can be broken by the use of an IMPLicit function, but a more satisfactory solution is provided by respecifying the model to provide an explicit solution of the algebraic loop.

3.4 User Defined Functions

In addition to the GUILDS and FORTRAN functions provided, GUILDS permits the user to create larger functional blocks from combinations of these functions, other structure statements and from new, user-defined FORTRAN routines. The functional blocks may range in complexity from an often-used sequence of simple structure statements to a complete model of a component within a larger model. The functions available are described below.

3.4.1 MACRO Function

The use of the MACRO function involves two distinct stages, the macro function definition and the macro function call. An example of a macro definition is given below:-

```
MACRO X1,X2=ARITH[Y1,Y2,Y3,Y4]    macro definition

W1=Y1*Y3
X1=Y1+Y2+W1                       macro code body
X2=Y3+Y4+W1

ENDMAC                            macro terminator
```

The first statement in the definition is the MACRO translation control statement specifying the canonical form which the user assigns to the function. The name appearing on the right of the equals sign is the name by which the function will be called in structure statements. The dummy output variables appear on the left of the equals and the dummy input variables on the right of the macro name enclosed in parentheses. The structure statements forming the macro code body follow the MACRO statement and are terminated by the ENDMAC statement.

The macro function call conforms exactly to the canonical form in the macro definition. When a macro call is encountered, the call is deleted from the input file and replaced by the macro expansion. The macro expansion is the statements of macro code body with all the dummy variables replaced by those used in the macro call. Note that any variables in the macro code body which are not dummy input or output variables are replaced by unique, generated variables each time the function is called. Thus global variables, that is variables defined elsewhere in the MODEL DESCRIPTION, cannot be used in the macro definition and can only be used in the macro expansion if passed through the parameter list of the macro call.

For example if the macro call:

```
A1,A2=ARITH[B1,B2,B3,B4]
```

is encountered, it is replaced by three lines with the following form:

```
ZZM17=B1*B3
A1=B1+B2+ZZ17
A2=B3+B4+ZZ17
```

where ZZM17 is a typical, unique variable generated in the macro expansion process.

If the real variable W1 had been replaced by an integer variable I1, then a unique variable such as IZM17 would have been generated by the macro expansion process.

Macro function definitions must appear before any statements, other than the TITLE statement,in the MODEL DESCRIPTION. No data statements or translation control statements other than the PROCEDURE and ENDPROC statements may be used in the macro definition. The use of nested macros is only possible if the macro called from within another macro has been previously defined. If called in a sorted section the macro expansion is sorted, otherwise the user must establish the correct computational sequence in the definition.

3.4.2 PROCEDURE Function

Structure statements appearing in a procedure function are treated as a single functional entry by the sorting algorithm. The statements within the procedure are not sorted but are repositioned as a group according to the inputs and outputs declared in the PROCEDURE control statement which defines the function. An example of the form of a procedure is shown below:-

```
PROCEDURE A,B=PROCNAM(X,Y,Z)
.
.       structure statements
.
ENDPROC
```

The PROCEDURE statement defines the start of the function and declares the outputs and inputs, in this case, to be A,B and X,Y,Z respectively. The structure statements immediately follow the PROCEDURE statement and are terminated by the ENDPROC statement. The function may appear anywhere in a sorted section where A and B require to be evaluated. The statements in the function are automatically positioned as a group so that X, Y and Z have been previously computed in the current iteration cycle. Note that the variables in the PROCEDURE statement are not dummies as in the MACRO definition statement but that in the this case the name is a dummy which is associated with the group of statements for sorting purposes. By embedding the procedure in a macro function the procedure statements can be used several times throughout a simulation model.

3.4.3 FORTRAN subprograms

Users may add to the GUILDS library of FORTRAN FUNCTION and SUBROUTINE subprograms listed in Appendix C1 and then use these new subprograms in the MODEL DESCRIPTION. Once written, the subprograms must be compiled and added to the GUILDS library of subprograms using the appropriate facilities of the computer system.

3.4.4 Translatable functions

Experienced users may also supplement the GUILDS-supplied translatable functions listed in Appendices C2.1 and C2.2. To do this, familiarity with the STAGE2 macroprocessor is required and the appropriate translation macros must be added to the existing translation macros.

3.5 Use of System Variables by the user

There are a number of system variables used to control the running of the simulation package which the user may require to access. These variables are given below with a description of the use the package makes of the variable. The user may not alter the variables except where specified.

The double precision variable **TIME** is incremented, by the integration interval, at the start of each integration cycle and gives the elapsed time for the simulation. The variable **T** is similar to TIME but is incremented by a fraction of the integration interval and is used with higher order integration methods to evaluate intermediate points within the integration interval. For example, the variable T should be used for time-dependent functions so that these functions are correctly evaluated at the intermediate points.

The time at which a simulation run finishes is specified by the double precision variable **FINTIM** which is available in the DYNAMIC and TERMINAL segments. The TERMINATE facility (see Section 3.2.5) or the Keyboard Interrupt (see Section 4.9) can be used to end the simulation run at other than the specified FINTIM while still preserving the original value.

The integration interval **H** is available to the user in the INITIAL, DYNAMIC and TERMINAL segments.

The variables **ITIN** and **ITOUT** can be used throughout the simulation to specify the user's terminal for input and output respectively.

The logical variable **LY**, which contains the string 'Y', can be used throughout the simulation for testing the response to a YES/NO question.

After each pass through the simulation, a variable **KRUN**, the run counter, is incremented and this variable is available to the user in the INITIAL, DYNAMIC and TERMINAL segments. This permits, for example, ASK statements in the initial segment to be bypassed at all passes after the first. The value of KRUN is 0 until the first pass through simulation is complete.

The variable **KRR**, the re-run counter, is similarly incremented but only when the automatic re-run facility is selected. This variable is re-set to 0 when automatic re-run is deselected. The variable **LR**, which is set to 'Y' when automatic re-run is selected, can be altered by the user and should be set by to 0 to terminate the automatic re-run (see Section 4.8).

The variable M (available only within the DYNAMIC segment) is set to 1 during the final step of multipass integration methods in which the values of variables are output (printed, plotted or stored) and is set to 1 throughout, in single step integration methods. Events, such as incrementing counters, which must not be performed more than once at each time interval can be performed conditionally on M being equal to 1.

4.0 USING GUILDS

4.1 SIMULATION EXECUTIVE - Translating a MODEL DESCRIPTION

Once the MODEL DESCRIPTION has been written, the SIMULATION EXECUTIVE is entered as shown in Appendix A. The first section of the SIMULATION EXECUTIVE controls the operations given below:-

(i) Pre-processing of the MODEL DESCRIPTION to expand user defined macros;
(ii) Sorting the input statements of the DYNAMIC segment;
(iii) Translation of the resulting file to produce the FORTRAN subroutines;
(iv) Compiling of these subroutines to produce binary object files;
(v) Linking of these object files with those of the simulation package to produce a SIMULATION MODULE.

Two levels of interaction are provided. LEVEL 1, for inexperienced users, carries out all these operations automatically whereas LEVEL 2 permits the experienced user to select which of the processing operations are required.

Output files are produced by each of these phases and considerable scope exists for LEVEL 2 users to adapt the process to shorten the time taken in this section or to make additions or modifications to the files produced by one of the first three phases.

4.2 SIMULATION EXECUTIVE - Running a SIMULATION MODULE

If the user enters the SIMULATION EXECUTIVE at the beginning, entry to this second section will be automatic. However, once a SIMULATION MODULE has been created by the first section, it may be run repeatedly without passing through the translation again, in the manner shown in Appendix A3. In both cases, control of the running of the SIMULATION MODULE is undertaken by the second section of the SIMULATION EXECUTIVE.

In this section, the user is first required to respond to the questions in the MODEL DESCRIPTION, such as those generated by ASK statements. The remaining statements in the INITIAL segment are executed and then the user is offered a choice within a "pre-run dialogue" from the various options (e.g. integration method, output variables) which control the running of the SIMULATION MODULE. After a first run through the simulation module when the necessary questions have been answered, the module may be rerun using the same control options and making changes only within the model itself (e.g. in model parameters entered through ASK statements). The user may then bypass the pre-run dialogue. An automatic re-run facility is also provided and in this case the dialogue is automatically bypassed and the user most often would refrain from inserting statements in the model which would result in questions being asked.

The control options open to the user at run-time are described in the following sections and a typical example is given in Figure A5 of Appendix A1.

Where the package dialogue requires answers in the form YES or NO, only the first character need be given, followed by a carriage return <CR>. (Any other reply including <CR> on its own is taken as a negative response.)

When values for variables are being entered the user should employ either integer or real notation dependent on the name of the variable (using the FORTRAN convention of A-H and O-Z real and I-N integer), for example:

'INC' will only accept an integer value
'FREQ' requires a real .

The acceptable forms in which real and integer variables can be entered varies with the implementation (computer or operating system) and reference should be made to the appropriate FORTRAN manual where there is any doubt.

If a mistake is made in replying to a question there are three possible opportunities for correcting it:

(i) if the error is detected by the FORTRAN READ statement, for example "FORMAT VARIABLE TYPE MISMATCH", "ERROR IN REPLY - TRY AGAIN" may be printed along with the FORTRAN error message and the question repeated until an acceptable answered is received;

(ii) if an illegal or undefined name is used to select a variable for, for example, printing then "NAME NOT RECOGNISED" is printed and the question repeated;

(iii) the final question in the dialogue section asks the user if there are any mistakes in the preceding dialogue and if the user replies in the affirmative then all the question are repeated.

In place of the interactive pre-run dialogue, the user is offered the opportunity to use the command mode which permits the user to set up the various options for the run without having to answer all the questions. This is particularly useful when repeated runs are being done and only a limited number of the control variables need to be altered. The commands available are listed in Appendix D but can also be obtained by using the HELP command.

4.3 Integration Methods

There are five integration techniques available within GUILDS as shown below. The method is selected by the user in response to the relevant question in the dialogue. If the user replies with <CR> then RK4 is the integration method chosen by default.

(i)	Euler (first order)	EUL
(ii)	Modified Euler (second order)	MEU
(iii)	Third Order Runge-Kutta	RK3
(iv)	Fourth Order Runge-Kutta	RK4
(v)	Variable Step Runge-Kutta	RKV

The first four techniques are fixed step length methods, details of which can be obtained from standard texts[4,5]. In association with each of these methods the user has to select an integration interval (H) and a finish time for the simulation run (FINTIM). For the fifth method which is fourth order Runge-Kutta with a variable integration step length the user has to specify a value for the convergence criterion (EPS).

For users whose requirements are not met by these techniques there exists a facility whereby a user written integration method can be included in the simulation. The user supplies a FORTRAN subroutine EXTERN containing the integration routine and this is linked into the simulation module instead of a default blank subroutine. The user's integration method is selected by replying EXT to the relevant question.

4.4 Printed Output

The values of up to five variables together with TIME can be printed in columns on the user's terminal during the simulation. The variables to be printed are chosen at run-time by entering the variable names in answer to the relevant question from the interactive dialogue. If less than five variables are to be printed then a <CR> is used to indicate that no more names are being entered. If a variable name is given which does not appear on the left hand side of an equation in the DYNAMIC segment (or is protected by @) then "NAME NOT RECOGNISED" is printed and another name can be given. The names of the variables being output are printed as headings above the respective columns.

A print interval is also specified by the user at run-time which may take any value greater than the integration interval but less than the simulation finish time i.e. H < PRNT < FINTIM.

4.5 Graphical Output

Graphical output is available on a Tektronix 4006 or 4010 storage screen or 4662 digital plotter. Up to three variables can be plotted at any one time, the variables being selected by name as for printed output. (Care should be taken when using the plotter that the simulation is not proceeding faster than the plotter can output, especially if more than one graph is being drawn.) The user is also required to supply minimum and maximum values for each variable being plotted. The name of the variable is printed above the relevent graph, the minimum and maximum values at the lower and upper limits of the graph and the overall title of the simulation (as in the TITLE statement) at the bottom of the page.

The facility also exists whereby the user can enter text which is printed on the plot below the title. This is particularly useful for indicating the values of certain parameters relating to the particular simulation run being plotted. The text entered is printed on every graph plotted, at run-time and from the post-run processor (see Section 4.7), until the text is changed by the user or the option deselected.

4.6 Stored Output

As an alternative, or in addition to printed and graphical output, data from the simulation may be stored in a disk file. Results can thus be saved for further analysis after the simulation run is over. To save disk space the results are stored in binary and thus cannot be listed directly. The post-run processor (see Section 4.7) is designed to access this data file.

If the storage facility is selected, the values of all the variables appearing on the left hand sides of assignment statements (not protected by @) in the DYNAMIC segment of the model, are stored. The user is required to specify a storage interval (H < STR < FINTIM) and a file name at run-time. The filename is of the form FILNAM.EXT where FILNAM is any legal name of up to eight characters and EXT is an optional extension of up to three characters. The extension is often used to indicate the nature of the information in the file and the recommended extension for this case is DAT.

4.7 Post-run Output

If the storage facility is selected at run-time the user is able to take advantage of the post-run processing routine for further output from the simulation. This routine acts upon the data stored during the simulation run and permits the user to select printed, plotted and/or stored output as before.

As for output at run-time, the printed output is to the user's terminal and consists of TIME and up to five variables selected by the user. No print interval is required, the storage interval being used.

Plotted output for up to three varaibles is available as before, with the additional features of using internally calculated minimum an maximum values and plotting a specified time window from within the original simulation time. The user can select whether or not to enter minimum and maximum values for all the plot variables. If values are being entered by the user, the calculated values can still be obtained for a specific variable by replying with <CR>. The start and stop times for the plot can be chosen by the user within the limits zero to the finish time of the run. If times outwith these limits are selected an error is given and new values can be entered.

The data stored at run-time can be re-written to disk in two different forms as directed by the user for analysis outwith the simulation language.

If "postprint" storage is selected, up to five variables and TIME are written to a disk file, named by the user, in ASCII format. This file can subsequently be listed or spooled if hard copy of the results is required.

Alternatively, if "postplot" storage is selected, up to five variables and TIME are written to a disk file in random access form. The format of this file is such that it can be used with an external plotting package (TEKPLT) which is suitable for producing up to four graphs, with titles and labelled axes as required, on an A4 page. The graphs on any one page need not come from the same file and so comparisons between the results of different simulations, or simulation and site results can easily be displayed.

In both the above cases the original storage interval is used thus no user supplied value is required.

The dialogue used to control the post-run processor is similar to that used at run-time and as before there is an equivalent command mode which can be used. The commands available are listed in Appendix D and can be displayed to the user by entering the HELP command.

It should be noted that, since the post-run processor requires a data file name to be specified, which defaults to the file created during the simulation run, it is possible to access data files from various simulation runs. As the storage interval and finish time for each file can be specified, and must be if different from the current values, the only condition is that the file was created by a simulation model having the same assignment statements in the DYNAMIC segment (in terms of the variables on the left hand sides of the equations) as the current simulation.

4.8 Automatic Re-run Facility

In association with the TERMINAL segment of the MODEL DESCRIPTION, a facility has been included in the simulation language which enables the simulation to be re-run automatically after the execution of the TERMINAL routine. When selected, the automatic re-run facility re-starts the simulation from the beginning ommiting any input/output in the INITIAL segment due to ASK and UPDATE statements and all of the run-control dialogue. Automatic re-run is selected at run-time, but only if the use of the TERMINAL segment has also been selected. If the user's MODEL DESCRIPTION has no TERMINAL segment and automatic re-run is selected using the default blank subroutine TERM then the simulation will continue to re-run unchanged indefinitely.

If the automatic re-run facility is selected, a variable, KRR the automatic re-run counter, is incremented after each pass through TERMINAL. The value of KRR is 0 until after the first pass through TERMINAL and, although it is available to the user it should not be altered (see Section 3.5).

To terminate the series of passes through INITIAL, DYNAMIC and TERMINAL, the system variable LR should be set by the user to 0. LR is tested by the system immediately after the TERMINAL routine has been executed and if it is set to 0, the dialogue section is entered permitting the user to regain control of the simulation. When this happens the variable KRR is reset by the system to 0.

4.9 Keyboard Interrupt

The user can interrupt the simulation at any time during the run by hitting <CR> on a specified terminal. The user is asked to identify the Keyboard Interrupt Device before the simulation run commences. Normally, the user's terminal would be selected, but any terminal recognised by the system can be specified. Once interrupted, the simulation pauses and the user can select one of several options. These options are listed on the user's terminal at run-time and are described in more detail below:-

(1) the simulation run can be continued unchanged as though it had not been interrupted;
(2) the simulation run can be started from the beginning again with the option of passing through the INITIAL segment and the interactive dialogue section;
(3) the simulation run can be terminated and control returned to the computer operating system;
(4) the simulation run can be terminated with control passing to the user as if the DYNAMIC segment had beem completed normally;
(5) a disturbance, if specified by the DISTRB function, can be applied. (The DISTRB function permits the user to specify the value of a variable (or variables) before and after a disturbance is applied from a Keyboard Interrupt (see Appendix C2).)

4.10 Run-time Documentation File

In order to record the details of a particular simulation run the user can specify that a Documentation File be created for the run. In response to a question that follows the interactive dialogue (or command mode) the user gives a file name and, at the end of the run this file will contain all the information pertaining to the run. The names and values of all the variables in the data statements in the Model Description are written to the file along with the title of the simulation, the date, the time, the run number and any comments that appeared in the Model Description. In addition, the options selected by the user at run-time are recorded in the file (for example, integration method, plot variables) along with details of any post-run output facilities used. Figure A12 shows an example of a documentation created during the first simulation run of Figure A5.

5. REFERENCES

1. WAITE, W.M. — The Mobile Programming System: Stage2. Con. A.C.M., Vol. 13, 1970

2. STRAUSS J.C., Editor, et al. — The SCi Continuous System Simulation Language (CSSL). Simulation, December 1979, pp 281 - 303.

3. IBM — IBM Application Program. System/360 Continuous System Modeling Program User's Manual.

4. LOSKO, J.S. — Digital Simulation of Physical Systems. Addison Wesley, 1972.

5. GREENSPAN, D. — Introduction to Numerical Analysis and Applications. Marlean Publishing Co., 1970.

A1. EXAMPLE OF THE SIMULATION OF A MASS SPRING AND DAMPER

The mass spring and damper system shown in Figure A1 is used as an example to illustrate many of the features of GUILDS described in this manual. The statements describing the dynamics of the system, five in all, first require to be sorted and then are translated into FORTRAN. The FORTRAN subroutines are compiled and then linked with the rest of the GUILDS subroutines to form the SIMULATION MODULE. Figure A2 gives a listing of the MODEL DESCRIPTION, Figure A3 gives the introductory text of the SIMULATION EXECUTIVE and Figure A4 shows how LEVEL 1 of the SIMULATION EXECUTIVE is used create the SIMULATION MODULE. Some typical dialogue for the RUN-CONTROL section of the SIMULATION EXECUTIVE is given in Figure A5 and Figures A6, A7 and A8 show the plotted output from the execution of the SIMULATION MODULE.

Appendix A2 gives further detail for experienced users and illustrates some of the more advanced features of GUILDS.

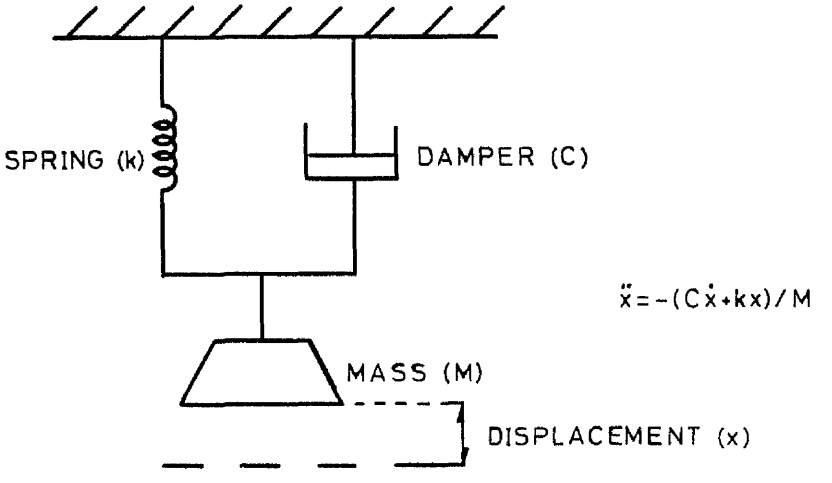


Figure A1 - Mass, Spring and Damper System and Equations

```
TITLE MASS, SPRING AND DAMPER SYSTEM
*EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS
*
CONSTANT C=1.,AK=2.
*
UPDATE AM=1.
*
INITIAL
*
ASK INITIAL DISPLACEMENT OF MASS/A
*
B=0.
*
DYNAMIC
*
  X2DOT=-(Y1+Y2)/AM      :ACCELERATION
  Y1=C*XDOT
  Y2=AK*X
  XDOT=INTGRL(B,X2DOT)   :VELOCITY
  X=INTGRL(A,XDOT)       :POSITION
*
END
```

Figure A2 - User-written MODEL DESCRIPTION

EXPLANATORY COMMENTS	PDP-11 INPUT/OUTPUT (user responses underlined)
Start Simulation Executive	@GUILDS
	>; GLASGOW UNIVERSITY INTERACTIVE LANGUAGE FOR DYNAMIC SIMULATION
	>;
	>; *** GUILDS - SIMULATION EXECUTIVE - TRANSLATION ***
	>;
	>;MOST QUESTIONS REQUIRE A YES OR NO ANSWER
	>;AS INDICATED BY [Y/N]. FOR YES USE Y AND
	>;FOR NO USE N OR RETURN <CR>.
	>;
	>;WHERE A FILE NAME IS REQUIRED IN ANSWER TO A QUESTION,
	>;AS INDICATED BY [S], REPLY WITH NAME.EXT FOR THE MODEL
	>;DESCRIPTION FILE BUT WITH NAME ONLY FOR ANY OTHER FILE.
Introductory Text	>;SOME FILES HAVE DEFAULT NAMES (SHOWN IN BRACKETS)
	>;WHICH WILL BE USED IF REPLY IS <CR> ONLY.
	>;
	>;THERE ARE TWO LEVELS OF OPERATION WITHIN THE EXECUTIVE:-
	>;LEVEL 1 IS RECOMMENDED FOR THE INEXPERIENCED USER WHEREAS THE
	>;EXPERIENCED USER MAY FIND THAT TIME CAN BE SAVED USING LEVEL 2
	>;
	>;AS THE TRANSLATION PROCESS PROCEEDS A CERTAIN AMOUNT OF
	>;OUTPUT WILL BE PRODUCED ON THE USER'S TERMINAL - LINES
	>;STARTING WITH OTHER THAN >* OR >; CAN BE IGNORED.
	>;
	>;TO EXIT FROM EXECUTIVE AT ANY STAGE REPLY WITH CNRTL Z
	>;

Figure A3 - Introduction to GUILDS SIMULATION EXECUTIVE

EXPLANATORY COMMENTS	PDP-11 INPUT/OUTPUT (user responses underlined)
Select Level of Interaction	>* WHICH LEVEL OF OPERATION DO YOU WISH TO USE(1 OR 2) [0]: 1
	>;
Entry to LEVEL 1 (See Fig. A9 for LEVEL 2)	>; *** GUILDS - SIMULATION EXECUTIVE - TRANSLATION - LEVEL 1 ***
	>;
Input Name of MODEL DESCRIPTION	>* NAME OF MODEL DESCRIPTION FILE [S]: EXAMP.SIM
Input Name for SIMULATION MODULE	>* NAME FOR SIMULATION MODULE [S]: TEST
	>ST2 MACZZ.SIM=KAMAC3.ST2,EXAMP.SIM
MACRO Expansion (No Action if No MACROs)	ST2 - END FILE <bell>
	END STAGE2 <bell>
	>ST2 SORTZZ.SIM=KASRT3.ST2,MACZZ.SIM
Sorting (No Action if Statements in Order)	ST2 - END FILE <bell>
	END STAGE2 <bell>
	>ST2 TEMPZZ.SIM=KAMA15.ST2,SORTZZ.SIM
Translation - Produces FORTRAN Subroutines	ST2 - END FILE <bell>
INIT and MODEL, from the Output of the	END STAGE2 <bell>
Sorting Phase, in Files INITZZ.FTN and	>ST2 INITZZ.FTN=KASEP1.ST2,TEMPZZ.SIM
MODZZ.FTN Respectively	ST2 - END FILE <bell>
	END STAGE2 <bell>
	>ST2 MODZZ.FTN=KASEP2.ST2,TEMPZZ.SIM
	ST2 - END FILE <bell>
	END STAGE2 <bell>
	>PIP TEMPZZ.SIM;*/DE
Compilation of FORTRAN subroutines	>FOR @FORCMD
	INIT
	MODEL
	>PIP TEST.TSK/PU
Task-building SIMULATION MODULE	>TKB TEST/FP/CP=TKBSIM/MP
	>;
SIMULATION MODULE Running (See Fig. A5)	>RUN TEST

Figure A4 - Example of LEVEL 1 of SIMULATION EXECUTIVE - TRANSLATION

EXPLANATORY COMMENTS	PDP-11 INPUT/OUTPUT (user responses underlined)
See Note 1	16-JUN-80 11:54:47
	*** GUILDS - SIMULATION EXECUTIVE - RUNNING ***
	*** START OF INITIAL SECTION ***
Title of Simulation	MASS, SPRING AND DAMPER SYSTEM
	DO YOU WISH TO UPDATE ANY VARIABLES? Y
Variables from UPDATE Statement	VARIABLES FOR UPDATING: AM
Select 'AM' for Updating	AM
Enter New Value	PRESENT VALUE= 1.00000 NEW VALUE? 2.
No More Variables to be Updated	<CR>
Text of ASK Statement	INITIAL DISPLACEMENT OF MASS
Input Value	-0.5
INITIAL Statements Executed	
	*** START OF RUN CONTROL SECTION ***
See App D for Commands Available	DO YOU WISH TO USE COMMAND MODE? N
See Note 2	INTEG METHOD - <EUL>,<MEU>,<RK3>,<RK4>,<RKV>,<EXT> EUL
	INTEGRATION INTERVAL,FINTIM? .1,10.
	DO YOU WISH PRINTED OUTPUT? Y
	ENTER PRINT INTERVAL 1.
	NAME OF VARIABLE TO BE PRINTED X
	NAME OF VARIABLE TO BE PRINTED XDOT
No More Varibles for Printing	NAME OF VARIABLE TO BE PRINTED <CR>
	DO YOU WISH OUTPUT ON PLOTTER? Y
	DO YOU WISH TO ADD TEXT TO PLOT? Y
	ENTER TEXT (UP TO 40 CHARACTERS)
	THIS IS A DEMONSTRATION PLOT
	NAME OF VARIABLE TO BE PLOTTED X
	MINIMUM AND MAXIMUM VALUES? -.5,.5
No More Varibles for Plotting	NAME OF VARIABLE TO BE PLOTTED <CR>
	DO YOU WISH TO STORE VARIABLES IN FILE? Y
	ENTER STORAGE INTERVAL .1
	FILENAME? TEST.DAT
See Note 3	DO YOU WISH TO MAKE USE OF TERMINAL SECTION? <CR>
If 'Y' Return to "INTEG METHOD etc."	ARE THERE ANY MISTAKES IN THE ABOVE DIALOGUE? N
	DO YOU WISH A DOCUMENTATION FILE FOR THIS RUN? Y
	FILENAME? TEST.DOC
	ENTER KEYBOARD INTERRUPT DEVICE NO. 15
	*** START OF SIMULATION RUN ***
Only if Plotting Selected	ENTER PLOTTER TERMINAL NO. 1
Typical Output Format	TIME X XDOT
	0.0000 -0.5000 0.0000
Remainder of Output Omitted	1.000 -0.3035 0.3313
	.
See Fig. A6 for Plotter Output	.
	10.00 0.4239E-01 -0.1080E-01

Figure A5 - Example of SIMULATION EXECUTIVE - RUNNING
(Part 1 of 2)

```
                                    *** START OF POST-RUN PROCESSING SECTION ***

Select Post-run Output              DO YOU WISH FUTHER OUTPUT? Y
File Name Defaults to TEST.DAT (See Note 4)  FILENAME? <CR>
See App D for Commands Available    DO YOU WISH TO USE COMMAND MODE? N
                                    DO YOU WISH PRINTED OUTPUT ON SCREEN? N
                                    DO YOU WISH OUTPUT ON PLOTTER? Y
                                    DO YOU WISH TO ADD TEXT TO PLOT? Y
Same Text as Entered at Run-time is Used  SAME TEXT? Y
                                    DO YOU WISH TO SPECIFY A TIME RANGE FOR PLOT? Y
                                    ENTER PLOT START AND STOP TIMES 5.,15.

Stop Time Greater than FINTIM       START OR STOP TIME IN EXCESS OF FINTIM

Re-enter Plot Start and Stop Times  ENTER PLOT START AND STOP TIMES 0.,10.
                                    DO YOU WISH TO ENTER MINIMUM AND MAXIMUM VALUES? Y
                                    NAME OF VARIABLE TO BE PLOTTED X
                                    MINIMUM AND MAXIMUM VALUES? -0.5,.25
                                    NAME OF VARIABLE TO BE PLOTTED XDOT
                                    MINIMUM AND MAXIMUN VALUES? -0.2,.4
                                    NAME OF VARIABLE TO BE PLOTTED XDOT2
XDOT2 is not a Variable             NAME NOT RECOGNISED
X2DOT is the Correct Name           NAME OF VARIABLE TO BE PLOTTED X2DOT
Calculated Values by Default        MINIMUM AND MAXIMUM VALUES? <CR>
Refer to Sect. 4.7 of Manual and See Below  DO YOU WISH TO STORE VARIABLES FOR PLOTTING? <CR>
See Below                           DO YOU WISH TO STORE VARIABLES IN ASCII FORM? N
See Fig. A7 for Plotter Output      ENTER PLOTTER TERMINAL NO.  1
More Output Required after Plot     DO YOU WISH FUTHER OUTPUT? Y
Same File                           FILENAME? <CR>
Command Mode Selected               DO YOU WISH TO USE COMMAND MODE? Y
Plotting Deselected                 ENTER COMMAND STRING NOPLOT
Store Variables for Plotting        ENTER COMMAND STRING PLTSTR
                                    NAME OF VARIABLE TO BE STORED X
                                    NAME OF VARIABLE TO BE STORED XDOT
                                    NAME OF VARIABLE TO BE STORED X2DOT

No More Plot Variables              NAME OF VARIABLE TO BE STORED <CR>
Name for Plot File                  FILENAME? TEST.PLT
Store Variables for Printing        ENTER COMMAND STRING STORE
                                    NAME OF VARIABLE TO BE STORED X
                                    NAME OF VARIABLE TO BE STORED XDOT
                                    NAME OF VARIABLE TO BE STORED X2DOT
No More Print Variables             NAME OF VARIABLE TO BE STORED <CR>
Name for Print File                 FILENAME? TEST.SPL
Output to Files TEST.PLT and TEST.SPL
No More Post-run Output             DO YOU WISH FUTHER OUTPUT? <CR>

                                        *** END OF POST-RUN OUTPUT SECTION ***

Start Simulation Run Again          DO YOU WISH TO START AGAIN? <Y OR N> Y
                                    DO YOU WISH TO RETURN THROUGH INIT? N
                                    DO YOU WISH TO RETURN THROUGH DIALOGUE? Y

                                        *** START OF RUN CONTROL SECTION ***

                                    DO YOU WISH TO USE COMMAND MODE? Y
                                    ENTER COMMAND STRING FINTIM
Change Finish Time to 20s           FINISH TIME? 20.
Printed Output Deselected           ENTER COMMAND STRING NOPRNT
No More Commands, therefore Continue  ENTER COMMAND STRING <CR>
                                    DO YOU WISH A DOCUMENTATION FILE FOR THIS RUN? Y
                                    FILENAME? TEST2.DOC
                                    ENTER KEYBOARD INTERRUPT DEVICE NO. 15

                                          *** START OF SIMULATION RUN ***

See Fig. A8 for Plotter Output      ENTER PLOTTER TERMINAL NO.  1
No Post-run Output Required         DO YOU WISH FUTHER OUTPUT? <CR>
No More Simulation Runs Required    DO YOU WISH TO START AGAIN? <Y OR N> <CR>
End of SIMULATION MODULE            TT15  --  STOP
End of SIMULATION EXECUTIVE         >@ <EOF>
```

Figure A5 - Example of SIMULATION EXECUTIVE - RUNNING
(Part 2 of 2)

The following Notes refer to Figure A5:-

Note 1 In general, if the answer to any of the YES/NO questions shown in this example had been different from that shown, the dialogue associated with that question would have been omitted and the next question asked.

Note 2 All the integration methods result in the same dialogue when selected except RKV. In this case the user is asked to specify a value for the convergence criterion in addition to the integration interval and finish time. Also, if the user replied with <CR> instead of an integration method, the RK4 method is used by default.

Note 3 If the reply to this question had been 'Y', the user would have been given the option of using the automatic rerun facility provided by the language. If selected, the rerun facility returns control to the start of the INITIAL segment of the SIMULATION MODULE after execution of the TERMINAL segment.

Note 4 If <CR> is given in reply to this question the data file created by the most recent execution of the simulation, assuming that the simulation module has not been exited, is used as the default data file for post-run optput. If, on the other hand, the user specifies a different file from the default then the user is asked to give the storage interval and finish time which were used when this was written. This permits the user to access data files created by the same simulation module under different run-time conditions.

The following Note refers to Figures A6, A7, A8:-

The values shown on the vertical axes of these figures correspond to the extremities of the axes and have been displaced slightly for clarity.

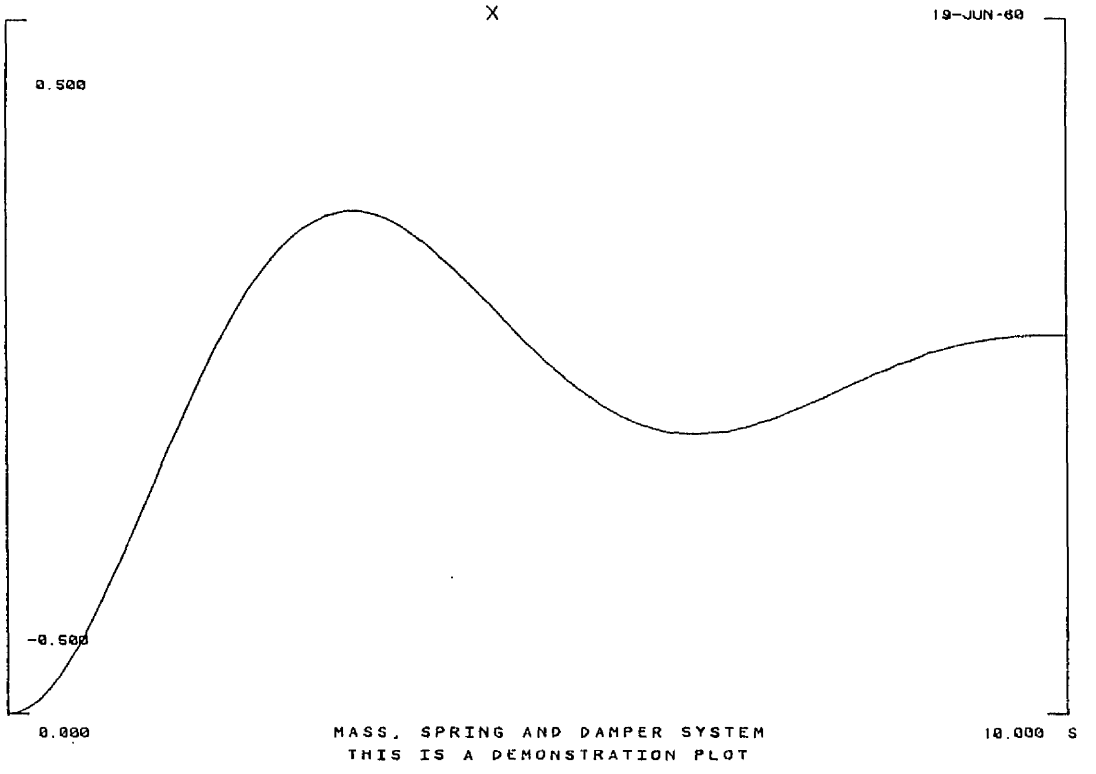


Figure A6 - Plotter Output from First Simulation Run

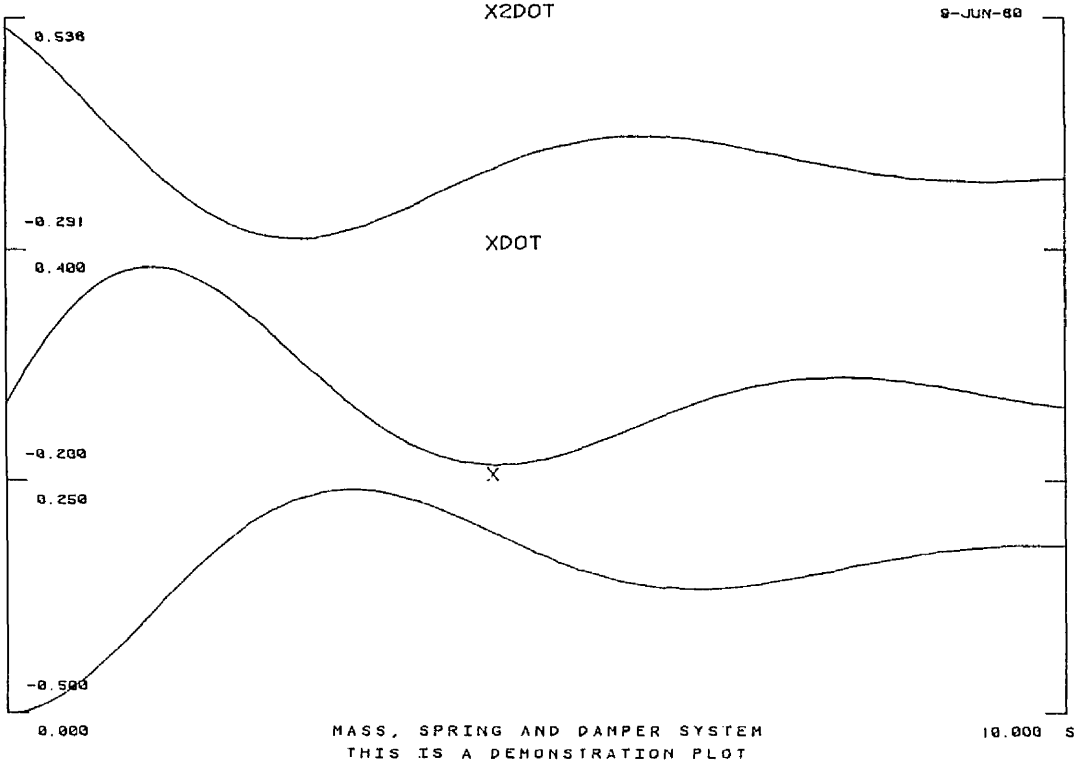


Figure A7 - Plotter Output from Post-run Processor

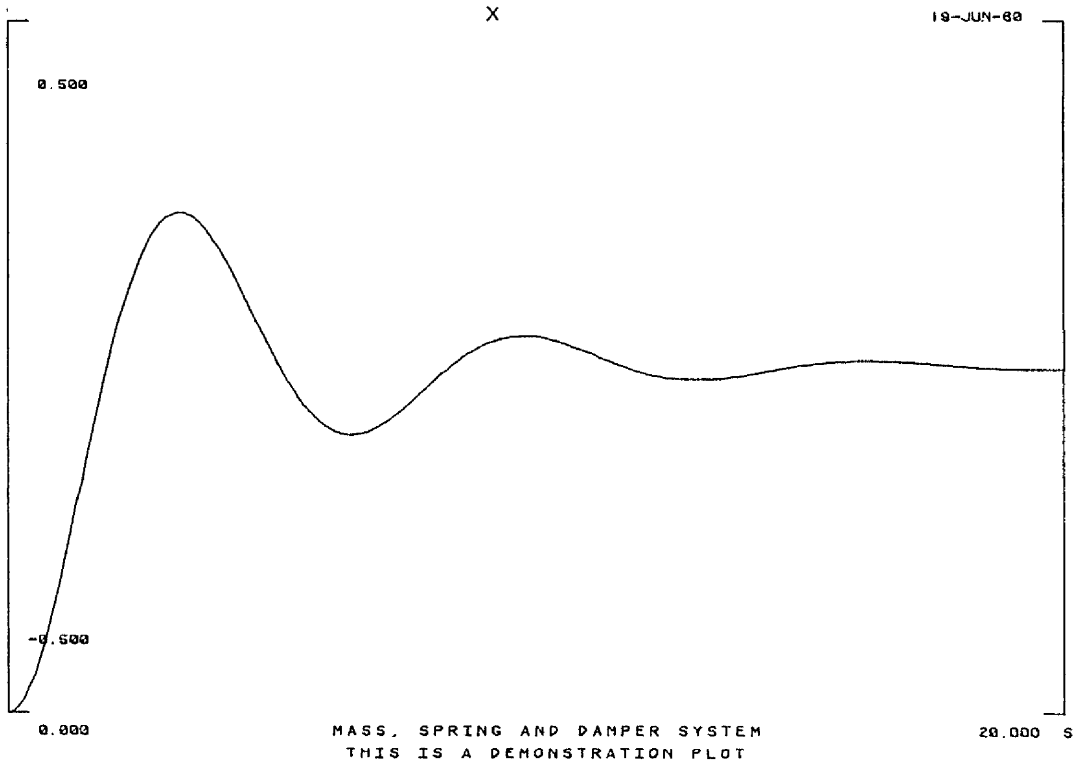


Figure A8 - Plotter Output from Second Simulation Run

A2. EXAMPLE OF MORE ADVANCED TRANSLATION FEATURES

Figure A9 shows how LEVEL 2 of the SIMULATION EXECUTIVE could be used to create a SIMULATION MODULE from the same MODEL DESCRIPTION (Figure A2) as used in Appendix A1. The output from the sorting phase, stored in a file SORT.SIM is shown in Figure A10. Apart from ordering the statements of the model description, the sorting processor also outputs all the statements in a standard format as shown. (This is the format which is recommended for writing model descriptions and is used for the output from the macro expansion and sorting phases because leading spaces and tabs are deleted during the processing and hence any spaces or tabs inserted by the user would be lost.) The output from the translation process is a file (TEMPZZ.SIM) containing concatenated FORTRAN subroutines in source form. The listing given (Figure A11) is of the FORTRAN source file annotated to show how the translator composes the source file from the input file. For each line of output the start of the corresponding input line is given. Certain lines of output do not correspond to any input lines but are associated with the FORTRAN package. These lines are output unchanged for every simulation module created whereas the remainder of the output lines are determinded by the statements in the input file.

EXPLANATORY COMMENTS	PDP-11 INPUT/OUTPUT (user responses underlined)
Select Level of Interaction	>* WHICH LEVEL OF OPERATION DO YOU WISH TO USE(1 OR 2) [0]: 2
	>;
	>;EXPERIENCED USERS MAY WISH TO ENTER LEVEL 2 DIRECTLY:
	>; TYPE @LEVEL2 INSTEAD OF @GUILDS TO START.
	>;
Entry to LEVEL 2 (See Fig A4 for LEVEL 1)	>; *** GUILDS - SIMULATION EXECUTIVE - TRANSLATION - LEVEL 2 ***
	>;
Input Name of MODEL DESCRIPTION	>* NAME OF MODEL DESCRIPTION FILE [S]: EXAMP.SIM
Input Name for SIMULATION MODULE	>* NAME FOR SIMULATION MODULE [S]: TEST
Input Name for SUBROUTINE INIT	>* SUBROUTINE INIT NAME(INITZZ) [S]: INIT01
Use Default Name for SUBROUTINE MODEL	>* SUBROUTINE MODEL NAME(MODZZ) [S]: <CR>
No TERMINAL Section (See Note 1)	>* DO YOU HAVE A TERMINAL SECTION? [Y/N]:N
No External Integration Routine (See Note 2)	>* DO YOU HAVE AN EXTERNAL INTGRATION ROUTINE? [Y/N]:<CR>
STAGE2 Processing Required	>* DO YOU REQUIRE STAGE2 PROCESSING? [Y/N]:Y
No MACRO's to be Expanded (See Note 3)	>* DO YOU HAVE ANY MACROS? [Y/N]:N
Sorting Required	>* DO YOU REQUIRE SORTING? [Y/N]:Y
Name for File Containing Sorted Output	>* NAME FOR SORTED FILE(SORTZZ) [S]: SORT
	>ST2 SORT.SIM=KASRT3.ST2,EXAMP.SIM
Sorting of MODEL DESCRIPTION	ST2 - END FILE <bell>
	END STAGE2 <bell>
	>ST2 TEMPZZ.SIM=KAMA15.ST2,TEST.SIM
Translation - Produces FORTRAN Subroutines	ST2 - END FILE <bell>
INIT and MODEL, from the File SORT.SIM, in	END STAGE2 <bell>
Files INIT01.FTN and MODZZ.FTN Respectively.	>ST2 INIT01.FTN=KASEP1.ST2,TEMPZZ.SIM
A Temporary File TEMPZZ.SIM is used and	ST2 - END FILE <bell>
then Deleted.	END STAGE2 <bell>
	>ST2 MODZZ.FTN=KASEP2.ST2,TEMPZZ.SIM
	ST2 - END FILE <bell>
	END STAGE2 <bell>
	>PIP TEMPZZ.SIM;*/DE
	>* DO YOU REQUIRE COMPILING? [Y/N]:Y
	>* INIT? [Y/N]:Y
	>* MODEL? [Y/N]:Y
	>FOR @FORCMD
Compilation of FORTRAN Subroutines	INIT
into Files INIT01.OBJ and MODZZ.OBJ	MODEL
	>* DO YOU REQUIRE TASK-BUILDING? [Y/N]:Y
Previous Versions of SIMULATION MODULE	>* DO YOU WISH TO KEEP PREVIOUS VERSION? [Y/N]:<CR>
to be Deleted	>PIP TEST.TSK;*/DE
Task-building SIMULATION MODULE using	>TKB TEST/FP/CP=TKBSIM/MP
INIT01.OBJ, MODZZ.OBJ and GUILDS Routines.	>;
SIMULATION MODULE Running (see Fig. A5)	>RUN TEST

Figure A9 - Example of LEVEL2 of SIMULATION EXECUTIVE TRANSLATION

The following notes refer to Figure A9:-

Note 1 If the reply 'Y' had been given to this question the user would have been asked to supply the name of a file for subroutine TERM, produced from the TERMINAL segment of the MODEL DESCRIPTION by the translation phase. If the user replies with <CR> the default name TERMZZ.FTN is used. (Note that this is a default name for the file corresponding to the user's TERMINAL segment, not to be confused with the default file, TERMBA.FTN, used by the task-builder if there is no TERMINAL segment in the MODEL DESCRIPTION.) Also, when the FORTRAN subroutines are being compiled, the user is asked, as for INIT and MODEL, whether subroutine TERM is to be compiled.

Note 2 As a result of a 'Y' response to this question the user would have been asked to specify the name of the file containing the enternal integration routine. The default name is EXTZZ.FTN which is used if the user responds with <CR>. (Note that this is a default name for the file containing to the user supplied integration routine, not to be confused with the default file, EXTBA.FTN, used by the task-builder if there is no external integration routine.) Also, when the FORTRAN subroutines are being compiled, the user is asked, as for INIT and MODEL, whether subroutine EXTERN is to be compiled.

Note 3 As there are no MACRO definitions in this example of a MODEL DESCRIPTION, time can be saved by omitting the macro expansion phase. If, however, this phase was required, a file MACZZ.SIM would be generated from the MODEL DESCRIPTION, containing the macro expansions, which would then be used as input to the next phase of processing (sorting).

```
TITLE MASS, SPRING AND DAMPER SYSTEM
*EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS
*
      CONSTANT C=1.,AK=2.
*
      UPDATE AM=1.
*
INITIAL
*
      ASK INITIAL DISPLACEMENT OF MASS/A
*
      B=0.
*
DYNAMIC
*
      Y1=C*XDOT
      Y2=AK*X
      X2DOT=-(Y1+Y2)/AM        :ACCELERATION
      XDOT=INTGRL(B,X2DOT)    :VELOCITY
      X=INTGRL(A,XDOT)         :POSITION
*
END
```

Figure A10 - MODEL DESCRIPTION After Sorting

LINE NO.:		FORTRAN PRODUCED BY TRANSLATOR:	EQUIVALENT SOURCE LINE:
1	C	MASS, SPRING AND DAMPER SYSTEM	TITLE...
2	C		None
3		SUBROUTINE INIT	None
4	C		None
5	C	EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS	Comment
6	C		Comment
7	C		Comment
8	C		Comment
9		DIMENSION W(30),WW(30)	None
10		DIMENSION VAL1(1)	UPDATE...
11		DOUBLE PRECISION HEADER(72)	None
12		DOUBLE PRECISION TIME,H,PLT,FINTIM,STR,PRNT	None
13		DOUBLE PRECISION ANAM1(1)	UPDATE...
14		LOGICAL*1 TITLE(72)	None
15		EQUIVALENCE (VAL1(1),AM)	UPDATE...
16		COMMON/C/LY	None
17		COMMON/D/ITIN,ITOUT	None
18		COMMON/H/TIME,H,PLT,FINTIM,STR,PRNT	None
19		COMMON/L/HEADER	None
20		COMMON/I/N,NV,NEW,LT	None
21		COMMON/R/W,WW	None
22		COMMON/T/TITLE,NSP	None
23		COMMON/UD1/AM	UPDATE...
24		COMMON/ASK1/A	ASK...
25		COMMON/INIT1/B	B=0.
26		COMMON/BLK1/C,AK	PARAMETER...
27		DATA HEADER/ 'Y1','Y2','X2DOT','XDOT','X',67*' '/	Dynamic statements
28		DATA TITLE/ 'M','A','S','S',',',' ','S','P','R','I','N','G',	TITLE...
29		1 ' ','A','N','D',' ','D','A','M','P','E','R',' ',	
30		1 'S','Y','S','T','E','M',42*' '/	
31		DATA C,AK/	PARAMETER...
32		1 1.,2./	
33		DATA ANAM1/'AM'/	UPDATE...
34		DATA VAL1/1./	UPDATE...
35	C		None
36		IF(LR.EQ.LY) GOTO 10000	None
37		WRITE(ITOUT,1003) TITLE	TITLE...
38	1003	FORMAT(1X,/,1X,72A1,/)	TITLE...
39		WRITE(ITOUT,1005)	None
40	1005	FORMAT(1X,'DO YOU WISH TO UPDATE ANY VARIABLES? ',$)	None
41		READ(ITIN,1010) LUD	None
42	1010	FORMAT(A1)	None
43		IF(LUD.NE.LY) GOTO 1250	None
44	C		None
45		WRITE(ITOUT,1020)	None
46	1020	FORMAT(1X,'VARIABLES FOR UPDATING: AM')	UPDATE...
47		CALL UPDATE(VAL1,ANAM1,1)	UPDATE...
48	1250	CONTINUE	None
49	C		Comment
50	C		None
51	1303	WRITE(ITOUT,1305)	ASK...
52		READ(ITIN,1500,ERR=1303)A	ASK...
53	1305	FORMAT(1X,'INITIAL DISPLACEMENT OF MASS')	ASK...
54	10000	CONTINUE	None

Figure A11 - FORTRAN Subroutines Produced by Translator
(Part 1 of 2)

```
55    C                                                                 Comment
56          B=0.                                                        B=0.
57    C                                                                 Comment
58    C                                                                 None
59    1500  FORMAT(10E13.6)                                            None
60    1510  FORMAT(10I7)                                              None
61    C                                                                 None
62          N=2                                                       INTGRL Statements
63          NV=5                                                      Dynamic Statements
64          NSP=42                                                    TITLE
65    C                                                               None
66          WW(1)=B                                                   XDOT=INTGRL...
67          WW(2)=A                                                   X=INTGRL...
68    C                                                               None
69          RETURN                                                    None
70          END                                                       None
71    C                                                               None
72          SUBROUTINE MODEL(VAR,DER,T,M)                             None
73    C                                                               None
74    C     EXAMPLE OF SIMULATION LANGUAGE TRANSLATION PROCESS        Comment
75    C                                                               Comment
76    C                                                               Comment
77    C                                                               Comment
78          DOUBLE PRECISION TIME,H,PLT,FINTIM,STR,PRNT               None
79          DIMENSION VAR(30),DER(30),DAT(72),W(30)                   None
80          REAL LIMIT                                                None
81          COMMON/V/Y1,Y2,X2DOT,XDOT,X,ZZ0,ZZ1,ZZ2,                  Dynamic Statements
82         1 ZZ3,ZZ4,ZZ5,ZZ6,ZZ7,ZZ8,ZZ9,ZZ10,
83         1 ZZ11,ZZ12,ZZ13,ZZ14,ZZ15,ZZ16,ZZ17,ZZ18,
84         1 ZZ19,ZZ20,ZZ21,ZZ22,ZZ23,ZZ24,ZZ25,ZZ26,
85         1 ZZ27,ZZ28,ZZ29,ZZ30,ZZ31,ZZ32,ZZ33,ZZ34,
86         1 ZZ35,ZZ36,ZZ37,ZZ38,ZZ39,ZZ40,ZZ41,ZZ42,
87         1 ZZ43,ZZ44,ZZ45,ZZ46,ZZ47,ZZ48,ZZ49,ZZ50,
88         1 ZZ51,ZZ52,ZZ53,ZZ54,ZZ55,ZZ56,ZZ57,ZZ58,
89         1 ZZ59,ZZ60,ZZ61,ZZ62,ZZ63,ZZ64,ZZ65,ZZ66
90          COMMON/D/ITIN,ITOUT                                       None
91          COMMON/H/TIME,H,PLT,FINTIM,STR,PRNT                       None
92          COMMON/R/W                                                None
93          COMMON/BLK1/C,AK                                          PARAMETER...
94          COMMON/UD1/AM                                             UPDATE...
95          COMMON/INIT1/B                                            B=0.
96          COMMON/ASK1/A                                             ASK...
97    C                                                               None
98          XDOT=VAR(1)                                               XDOT=INTGRL...
99          X=VAR(2)                                                  X=INTGRL...
100   C                                                               None
101   C                                                               Comment
102         Y1=C*XDOT                                                 Y1=C*XDOT
103         Y2=AK*X                                                   Y2=AK*X
104   C     ACCELERATION                                              X2DOT=...
105         X2DOT=-(Y1+Y2)/AM                                         X2DOT=...
106   C     VELOCITY                                                  XDOT=INTGRL...
107         DER(1)=X2DOT                                              XDOT=INTGRL...
108   C     POSITION                                                  X=INTGRL...
109         DER(2)=XDOT                                               X=INTGRL...
110   C                                                               Comment
111         RETURN                                                    None
112         END                                                       None
```

Figure A11 - FORTRAN Subroutines Produced by Translator
(Part 2 of 2)

A3. EXAMPLE OF RUNNING A SIMULATION MODULE WITHOUT TRANSLATION

Figure A12 shows the start of a simulation run where the SIMULATION MODULE is executed directly without evoking the first section (translation) of the SIMULATION EXECUTIVE. The user replies to the promt (>) from the computer operating system with RUN and the name given to the SIMULATION MODULE when it was created (in this case TEST).

```
>RUN TEST

             16-JUN-80          11:54:47

*** GUILDS - SIMULATION EXECUTIVE - RUNNING ***

       *** START OF INITIAL SECTION ***

MASS, SPRING AND DAMPER SYSTEM


etc. as in Figure A5.
```

Figure A12 - Running a SIMULATION MODULE Without Translation

B. EXAMPLE OF MACRO EXPANSION AND SORTING

An example of a MODEL DESCRIPTION using the MACRO and PROCEDURE facilities of GUILDS and thus requiring the use of the macro expansion and sorting phases of STAGE2 processing is given in Figure B1. This example is unnecessarily complicated but has been written in this way to illustrate several of the useful features incorporated in GUILDS. Figure B5 lists a MODEL DESCRIPTION representing the same physical system as it might be written by an experienced user employing some of the functions supplied by GUILDS.

The MODEL DESCRIPTION in Figure B1 has a macro definition for the macro FLOW which contains a procedure VALVE, to permit the use of FORTRAN branching statements. The statements in the macro definition are not in the correct computational sequence and therefore once the macro has been expanded the statements will require to be sorted. The statements in the DYNAMIC segment of the MODEL DESCRIPTION, which involve two calls to the FLOW macro also require to be sorted. A selection of different tabs and spaces are used at the start of the lines and some comment statements are included to illustrate how these constructions are processed.

Figure B2 shows the MODEL DESCRIPTION after the macro expansion phase. The macro definition has been removed from the start of the MODEL DESCRIPTION and each macro call has been replaced by the macro expansion. Within the macro expansion, the parameters in the macro call replace the dummy parameters in the macro definition (for example OPEN becomes OPEN1 and OPEN2 for the first and second calls respectively). Variables used within the macro which are not parameters, are replaced with generated variables unique to the macro call (for example VAL becomes ZZM1 for the first call and ZZM3 in the second call). All the line numbers in the macro definition are replaced by generated numbers, again unique to the macro call (for example 5 becomes 30001 and 30003 for the first and second calls respectively). Note that where a line number appeares on a line with an executable statement (for example 5 V=0.) the number is removed from that line and a CONTINUE statement, labelled with the generated line number, is inserted before the executable statement. This reduces the number of STAGE2 macros required to match all possible FORTRAN statements. As procedure names can only be used once in a MODEL DESCRIPTION it is necessary to uniquely identify the procedure VALVE each time the macro is expanded. The procedure name is retained so that the MODEL DESCRIPTION can be readily recognised when compared with the original and a number is added to the name (for example VALVE1) to make it unique. Finally, it should be noted that all the statements in the MODEL DESCRIPTION have been output in a standard format and that all the comment statements have been preserved.

The output from the macro expansion phase requires to be sorted: the resulting output of the sorting phase is shown in Figure B3. This phase of processing orders the statements of the MODEL DESCRIPTION into the correct computational sequence and removes the PROCEDURE and ENDPROC statements. Statements within a procedure are not sorted individually but are placed in the MODEL DESCRIPTION as a block at a position where the expression in the PROCEDURE statement (for example ZZM1=VALVE1(SW1,OPEN1)) could be evaluated if treated as a function call. Thus the statements of the procedure VALVE1 are output as soon as the procedure is encountered since SW1 and OPEN1 have been previously defined. The output of statements which cannot be evaluated at the position in the MODEL DESCRIPTION at which they appear is deferred until all the variables on the right hand side of the expression have been defined. Thus, for example, the statement FT=F1+F2,

which requires the values of F1 and F2, is not output until F1 and F2 have been calculated, the statements for F1 and F2 being deferred until ZZM1, ZZM2 and HEAD have been calculated. Note that a comment appearing in the same line as an executable statement remains attached to that statement after sorting but that separate comments (* statements) may appear at a different point in the MODEL DESCRIPTION in relation to the executable statements, as shown in the example.

Figure B4 shows part of subroutine MODEL produced by the translation phase from Figure B3 and illustrates the way in which these statements are dealt with.

```
TITLE EXAMPLE OF MACRO EXPANSION AND SORTING
MACRO F=FLOW[SW,OPEN,HEAD]
      F=VAL*SQRT(2*9.81*HEAD)        :CALCULATE  FLOW
PROCEDURE  V=VALVE(SW,OPEN)
      IF(SW.EQ.0.) GOTO 5
      V=OPEN
      GOTO 10
5     V=0.
10    CONTINUE
ENDPROC
      VAL=LIMIT(0.,1.,V)
ENDMAC
INITIAL
*
* INPUT VALVE SETTINGS FROM TERMINAL
*
  ASK VALVE SETTINGS/SW1,SW2
*
  VOLIC=1.
  AREA=1.
  OPEN2=0.
DYNAMIC
      OPEN1=RAMP(5.,.1,0.,1.)
      FT=F1+F2
      F1=FLOW[SW1,OPEN1,HEAD]
      F2=FLOW[SW2,OPEN2,HEAD]
*
* THESE COMMENT LINES WILL APPEAR OUT OF PLACE AFTER SORTING.
*         INITIALLY THE NEXT LINE IS HEAD=VOLL/AREA
*
      HEAD=VOLL/AREA
      VOLF=INTGRL(0.,FT)             :CALCULATE  VOLUME
      VOL=VOLIC-VOLF
      VOLL=LIMIT(0.,VOLIC,VOL)
END
```

Figure B1 - User-written MODEL DESCRIPTION

```
TITLE EXAMPLE OF MACRO EXPANSION AND SORTING
INITIAL
*
* INPUT VALVE SETTINGS FROM TERMINAL
*
ASK VALVE SETTINGS/SW1,SW2
*
      VOLIC=1.
      AREA=1.
      OPEN2=0.
DYNAMIC
      OPEN1=RAMP(5.,.1,0.,1.)
      FT=F1+F2
      F1=ZZM1*SQRT(2*9.81*HEAD)      :CALCULATE FLOW
PROCEDURE ZZM2=VALVE1(SW1,OPEN1)
      IF(SW1.EQ.0.) GOTO 30001
      ZZM2=OPEN1
      GOTO 30002
30001 CONTINUE
      ZZM2=0.
30002 CONTINUE
ENDPROC
      ZZM1=LIMIT(0.,1.,ZZM2)
      F2=ZZM3*SQRT(2*9.81*HEAD)      :CALCULATE FLOW
PROCEDURE ZZM4=VALVE2(SW2,OPEN2)
      IF(SW2.EQ.0.) GOTO 30003
      ZZM4=OPEN2
      GOTO 30004
30003 CONTINUE
      ZZM4=0.
30004 CONTINUE
ENDPROC
      ZZM3=LIMIT(0.,1.,ZZM4)
*
* THESE COMMENT LINES WILL APPEAR OUT OF PLACE AFTER SORTING.
*         INITIALLY THE NEXT LINE IS HEAD=VOLL/AREA
*
      HEAD=VOLL/AREA
      VOLF=INTGRL(0.,FT)             :CALCULATE VOLUME
      VOL=VOLIC-VOLF
      VOLL=LIMIT(0.,VOLIC,VOL)
END
```

Figure B2 - MODEL DESCRIPTION After MACRO Expansion

```
TITLE EXAMPLE OF MACRO EXPANSION AND SORTING
INITIAL
*
* INPUT VALVE SETTINGS FROM TERMINAL
*
      ASK VALVE SETTINGS/SW1,SW2
*
      VOLIC=1.
      AREA=1.
      OPEN2=0.
DYNAMIC
      OPEN1=RAMP(5.,.1,0.,1.)
      IF(SW1.EQ.0.) GOTO 30001
      ZZM2=OPEN1
      GOTO 30002
30001 CONTINUE
      ZZM2=0.
30002 CONTINUE
      ZZM1=LIMIT(0.,1.,ZZM2)
      IF(SW2.EQ.0.) GOTO 30003
      ZZM4=OPEN2
      GOTO 30004
30003 CONTINUE
      ZZM4=0.
30004 CONTINUE
      ZZM3=LIMIT(0.,1.,ZZM4)
*
* THESE COMMENT LINES WILL APPEAR OUT OF PLACE AFTER SORTING.
*          INITIALLY THE NEXT LINE IS HEAD=VOLL/AREA
*
      VOL=VOLIC-VOLF
      VOLL=LIMIT(0.,VOLIC,VOL)
      HEAD=VOLL/AREA
      F1=ZZM1*SQRT(2*9.81*HEAD)      :CALCULATE FLOW
      F2=ZZM3*SQRT(2*9.81*HEAD)      :CALCULATE FLOW
      FT=F1+F2
      VOLF=INTGRL(0.,FT)             :CALCULATE VOLUME
END
```

Figure B3 - MODEL DESCRIPTION After (MACRO Expansion and) Sorting

```
C
C THESE COMMENT LINES WILL APPEAR OUT OF PLACE AFTER SORTING.
C         INITIALLY THE NEXT LINE IS HEAD=VOLL/AREA
C
      VOL=VOLIC-VOLF
      VOLL=LIMIT(0.,VOLIC,VOL)
      HEAD=VOLL/AREA
C     CALCULATE FLOW
      F1=ZM1*SQRT(2*9.81*HEAD)
C     CALCULATE FLOW
      F2=ZM3*SQRT(2*9.81*HEAD)
      FT=F1+F2
C     CALCULATE VOLUME
      DER(1)=FT
      RETURN
      END
```

Figure B4 - Part of SUBROUTINE MODEL Showing Comment Statements

```
TITLE EXAMPLE OF MACRO EXPANSION AND SORTING
INITIAL
*
* INPUT VALVE SETTINGS FROM TERMINAL
*
  ASK VALVE SETTINGS/SW1,SW2
*
  VOLIC=1.
  AREA=1.
  OPEN2=0.
DYNAMIC
      OPEN1=RAMP(5.,.1,0.,1.)
      V1=SWIN(SW1,0.,OPEN1)
      V1L=LIMIT(0.,1.,V1)
      V2=SWIN(SW2,0.,OPEN2)
      V2L=LIMIT(0.,1.,V2)
      VOL=VOLIC-VOLF
      VOLL=LIMIT(0.,VOLIC,VOL)
      HEAD=VOLL/AREA
      F1=V1L*SQRT(2*9.81*HEAD)       :CALCULATE FLOW 1
      F2=V2L*SQRT(2*9.81*HEAD)       :CALCULATE FLOW 2
      FT=F1+F2
      VOLF=INTGRL(0.,FT)            :CALCULATE  VOLUME
END
```

Figure B5 - Equivalent MODEL DESCRIPTION with no User Defined Functions

C1. FORTRAN FUNCTIONS PROVIDED BY GUILDS

Function Name	Brief Description	Canonical Form	Mathematical Description	
DISTURB	Apply Disturbance on User Interrupt	Y=DISTRB(DIC,DOP)	Y = DOP Y = DIC	After Interrupt Before Interrupt
FNSW	Switching Function	Y=FNSW(X1,X2,X3,X4)	Y = X2 Y = X3 Y = X4	X1 < 0 X1 = 0 X1 > 0
LIMIT	Limit Function	Y=LIMIT(B,T,X)	Y = B Y = T Y = X	X ⩽ B X ⩾ T B < X < T
RAMP	Ramp Function	Y=RAMP(RST,RR,RIC)	Y = RIC Y = RIC+RR*(t - RST)	t < RST t ⩾ RST
RAMP1	Ramp Function with Final Value	Y=RAMP(RST,RR,RIC,RFV)	Y = RIC Y = RIC+RR*(t - RST) Y = RFV	t < RST t ⩾ RST Y > RFV, RR > 0 or Y < RFV, RR < 0
STEP	Step	Y=STEP(ST)	Y = 0 Y = 1	t < ST t ⩾ ST
STPLIM	Limit With Steps	Y=STPLIM(P1,P2,P3,P4,X)	Y = P1 Y = P3 Y = X	X < P2 X > P4 Otherwise
SWIN	Switching Function	Y=SWIN(X,OFF,ON)	Y = ON Y = OFF	X ⩾ 0 X < 0

C2. FUNCTIONS PROVIDED BY GUILDS WHICH REQUIRE TRANSLATION

C2.1 Functions which are translated into other GUILDS functions

Function Name	Brief Description	Canonical Form	Mathematical Description
+INTGRL	Integrator	Y=INTGRL(YIC,X)	$Y = \int X dt + YIC$
+FOLAG	First Order Lag	Y=FOLAG(YIC,TL,X)	$Y = \frac{1}{1 + sTL} * X$
LEADLG	Lead Lag Compensator	Y=LEADLG(YIC,T1,T2,X)	$Y = \frac{1 + sT1}{1 + sT2} * X$
DERLAG	Derivative With Lag	Y=DERLAG(XIC,YIC,TL,X)	$Y = \frac{s}{1 + sTL} * X$
+CMPXPL	Second Order System	Y=CMPXPL(YIC,DYIC,P1,P2,X)	$Y = \frac{1}{s^2+2P_1P_2s+P_2^2}$

+ Predictive Functions (see section 3.3)

C2.2 Functions which are translated into FORTRAN functions

Function Name	Brief Description	Canonical Form	Mathematical Description	
ANDHYS	AND With Hysteresis	Y=ANDHYS(YIC,X1,X2)	Y = YIC Y = 1 Y = 0 Y = Previous Output	t = 0 X1 > 0, X2 > 0 X1 < 0, X2 < 0 Otherwise
DELAY	Time Delay	Y=DELAY(DT,X)	Y(t) = X(t - DT) Y(t) = X(0)	t > DT t < DT
HSTRSS	Hysteresis	Y=HSTRSS(YIC,P1,P2,X)	Y(0) = YIC Y = X - P2 Y = X - P1 Y = Previous Output	 $X - X_{n-1} > 0$ and $Y_{n-1} \leq X - P2$ $X - X_{n-1} < 0$ and $Y_{n-1} \geq X - P1$ Otherwise
RAMP2	Ramp Function with Switch	Y=RAMP2(RS,RR,RIC,RFV)	RST = t RST = last value Y = RIC Y = RIC+RR*(t - RST) Y = RFV	RS < 0 RS > 0 t < RST t > RST Y > RFV, RR > 0 or Y < RFV, RR < 0
RATLIM	Rate Limit	Y=RATLIM(T,RLT,RLB,X)	Y(T) = Y(T-dT)+RLT*dT Y(T) = Y(T-dT)+RLB*dT Y(T) = X	$\frac{dX}{dT} > RLT$ $\frac{dX}{dT} < RLB$ Otherwise

```
INTMET - SELECT INTEGRATION METHOD
INTVAL - SELECT INTEGRATION INTERVAL
FINTIM - UPDATE FINTIM
ERRCON - UPDATE CONVERGANCE FACTOR FOR RKV
PRINT  - SELECT PRINTING
NOPRNT - DESELECT PRINTING
PRTVAL - UPDATE PRINT INTERVAL
PRTVAR - VARIABLES FOR PRINTING
PLOT   - SELECT PLOTTING
NOPLOT - DESELECT PLOTTING
PLTVAR - VARIABLES FOR PLOTTING
PLTMM  - UPDATE PLOT MIN AND MAX
STORE  - SELECT STORAGE
NOSTOR - DESELECT STORAGE
FILNAM - FILE NAME FOR STORAGE
STRVAL - UPDATE STORAGE INTERVAL
TERM   - SELECT TERMINAL SECTION
NOTERM - DESELECT TERMINAL SECTION
RERUN  - SELECT AUTOMATIC RERUN FACILITY
NORR   - DESELECT RERUN
AXIS   - AXIS, TITLES, DATE AND SCALES REQUIRED
NOAXIS - AXIS ETC. NOT REQUIRED
TEXT   - ENTER TEXT FOR PLOT
NOTEXT - DESELECT TEXT FACILITY
HELP   - GIVE THIS LIST OF DEFINITIONS
DIALOG - RETURN TO DIALOGUE MODE
```

Figure D1 - Run Control Commands

```
PRINT  - SELECT PRINTING
NOPRNT - DESELECT PRINTING
PRTVAR - VARIABLE FOR PRINTING
PLOT   - SELECT PLOTTING
NOPLOT - DESELECT PLOTTING
PLTVAR - VARIABLES FOR PLOTTING
PLTMM  - UPDATE PLOT MIN AND MAX
PLTSS  - UPDATE PLOT STATR AND STOP TIMES
STORE  - SELECT STORAGE (IN ASCII)
NOSTOR - DESELECT STORAGE
FILNAM - FILE NAME FOR ASCII STORAGE
STRVAR - VARIABLES FOR STORING
PLTSTR - SELECT STORAGE FOR FURTHER PLOTTING
NOPPS  - DESELECT STORAGE
PPSFIL - FILE NAME FOR FURTHER PLOTTING
PPSVAR - VARIABLES FOR FURTHER PLOTTING
AXIS   - AXIS, TITLES, DATE AND SCALES REQUIRED
NOAXIS - AXIS ETC. NOT REQUIRED
TEXT   - ENTER TEXT FOR PLOT
NOTEXT - DESELECT TEXT FACILITY
HELP   - GIVE THIS LIST OF DEFINITIONS
DIALOG - RETURN TO DIALOGUE MODE
```

Figure D2 - Post-run Control Commands

E. Reserved Variables

Some variable names are reserved by the simulation language but may be used in the correct context, as detailed in Section 3.5. These variables are:-

TIME, T, FINTIM, H, ITIN, ITOUT, LY, KRUN, KRR, LR and M.

All variable names starting with IZ, LZ or ZZ are reserved for use exclusively by the language. Also, FORTRAN statement labels above 30000 are reserved and, in addition, in the INITIAL segment, statement labels from 1000 to 10000 may not be used.