DESIGN AND IMPLEMENTATION OF PANEL LOGIC FOR POWER PLANT SIMULATORS

By

YADURAJ KAPOOR

Bachelor of Engineering

Osmania University

Hyderabad, India

1975

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the degree of
MASTER OF SCIENCE
December, 1981

DESIGN AND IMPLEMENTATION OF PANEL LOGIC FOR POWER PLANT SIMULATORS

By

YADURAJ KAPOOR

Bachelor of Engineering

Osmania University

Hyderabad, India

1975

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the degree of
MASTER OF SCIENCE
December, 1981

OKLAHOMA STATE UNIVERSITY

Name: Yaduraj Kapoor Date of Degree: December, 1981

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: DESIGN AND IMPLEMENTATION OF PANEL LOGIC FOR POWER PLANT SIMULATORS

Pages in Study: 46 Candidate for Degree of Master of Science

Major Field: Computing and Information Science

Scope and Method of Study: This study involves the simulation and implementation of panel logic for Georgia Power Comany simulator, used for training operators. The implementation and coding in FORTRAN involves several steps of simplifying the functional control drawings (FCD's) provided by the utility company, and reduced to a format, generally known as implemetnation drawings from which a program is written. The program is executed in real-time on the simulator. The authenticity of the panel logic must conform to the actual power plant logic strictly, and its response must be identical both under normal operation and under existing malfunction conditions. Boolean algebra is used extensively to simplify the logic operations and reduce the real-time code.

Findings and Conclusions: When the program logic was merged with the rest of the simulator completely, its behavior and response to operator inputs from control panel and interaction with other process models was found to be exactly what would happen in the actual plant. This program logic could be run on an event-oriented basis instead of cyclic basis, and this could prove helpful in future simulations.

ADVISER'S APPROVAL _____

DESIGN AND IMPLEMENTATION OF PANEL LOGIC

FOR POWER PLANT SIMULATORS

Report Approved:

_____

Report Adviser

_____

_____

_____

Dean of the Graduate College

ACKNOWLEDGMENTS

The author wishes to express his deep appreciation to his major adviser, Dr. G. E. Hedrick, for his constant guidance and assistance throughout the preparation of this report. Appreciation is also expressed to the other members of the committee, Dr. J. P. Chandler and Dr. J. Richard Phillips, for their invaluable advice which was incorporated in preparing the final manuscript.

Appreciation is extended to Ms. Charlene Fries for her excellent typing of the manuscript.

Finally, my sincere thanks and gratitude are expressed to my friend, Mr. R. Viswanathan, for providing the support and encouragement in completing this work.

TABLE OF CONTENTS

Chapter		Page
I.	GENERAL INTRODUCTION	1
II.	REVIEW OF THE LITERATURE	3
	Introduction to Computer Simulation	3
	Simulation and Training Simulators	4
III.	SIMULATION OF POWER PLANT LOGIC	6
	Introduction	6
	Plant Logic Elementary	8
	Panel I/O (Input/Output)	8
	Notation	9
	Panel Switches	10
	Time Delay Timers	11
	Annunciator	12
	Lamp Indicators	13
	Comparators	13
	Flip-Flops	14
	Valves	14
	Standard Naming Conventions for Variables	15
	Summary	15
IV.	FUNCTIONAL CONTROL DRAWINGS	17
	Design Specifications	17
	System Block Diagram	21
V.	IMPLEMENTATION SPECIFICATION	24
	Data Structures and Programming Techniques	27
	Application of Boolean Algebra	28
	Functional Description	29
VI.	SUMMARY, CONCLUSIONS, AND FUTURE WORK	33
A	SELECTED BIBLIOGRAPHY	35
APPENDIX A -	GLOSSARY OF DEFINITIONS	36
APPENDIX B -	FIGURES	40

TABLE

Table		Page
1.	Table of Mnemonics	7

LIST OF FIGURES

Figure		Page
1.	Breaker Simplification	18
2.	SOV Simplification	19
3.	Redundant Measurement Simplification	20
4.	Simplified Block Diagram	23

CHAPTER I

GENERAL INTRODUCTION

This paper describes the successful simulation and implementation of the power plant logic for the Georgia Power Company simulator. This is accomplished in various steps: simplifying the customer provided functional drawings; producing implementation drawings in directly codable form; and coding the logic developed in implementation drawings in FORTRAN. Some important key words are also defined (in the Appendix) in order for the unfamiliar reader to understand the topics covered herein.

The ultimate objective of this project is the production of the logic module PANEL4 (which simulates the underlying logic of the control panel 4 of the actual plant), which is to simulate, strictly and faithfully, all the logical conditions that would appear in the actual power plant, under both normal and extreme conditions of plant operation. Only the logical operations are examined here, and other areas such as controls, process simulation, etc. are not addressed within this report. Each of these areas is much involved (and hence excluded), and each simulation is done by analysts proficient in that area.

There are two reasons to study this topic: First, the involvement with logic requires a thorough understanding and accurate application of computer engineering elements such as Boolean functions, memory elements, timers, etc. Secondly, having simplified the plant logic into an implementable format the use of appropriate data-structures (mostly linear-arrays),

grouping of identical operations, and program optimization in order to achieve maximum efficiency both in time and memory storage for real-time operation, is achieved by utilizing and applying computer science knowledge.

Since the material in this paper mostly deals with logical functions, application of Boolean logic in order for substantial simplification, covers most of the implementation. Hence mathematical definitions neither are required nor are necessary in discussing or developing this report. Most of the logic, after simplification, is straight-forward, and hence easily understandable.

The sample engineering drawings, given in the Appendix, are used in the discussion of the functional specification and implementation specifications presented in Chapters IV and V, respectively.

All the elements appearing on engineering drawings (which cover all of the elements on any implementation drawings) are discussed at length in Chapters II and III to aid in clear insight and understanding of plant logic operations.

A brief description of the objective (PANEL4 logic module) and steps involved in attaining this goal was given in this chapter. Boolean functions are most often used in developing the logical equations and mathematical definitions are seldom used.

CHAPTER II

REVIEW OF THE LITERATURE

Introduction to Computer Simulation

Simulation in scientific disciplines is the art of model building, using computers. Shubik's (9) definition appropriately states:

> A simulation of a system or an organism is the operation of a model or simulator which is a representation of the system or organism. The model is amenable to manipulations which would be impossible, too expensive or impractical to perform on the entity it portrays. The operation of the model can be studied and, from it, properties concerning the behavior of the actual system or its subsystem can be inferred (41, p. 909).

Simulation can also be defined as a

> Numerical technique for conducting experiments on a digital computer, which involves certain types of mathematical and logical models that describe the behavior of a business or economic system (or some component thereof) over extended periods of real time (5).

Simulation has been in wide use in areas ranging from economics and operations research to relatively new fields such as fossil/nuclear power plant training simulators.

The importance of simulation models or model building, whether be a part of a simulator or an intergral part of any scientific inquiry, is stated quite succinctly by Rosenbleuth and Wiener (8).

> No substantial part of the universe is so simple that it can be grasped and controlled without abstraction. Abstraction consists in replacing the part of the universe under

> consideration by a model of similar but simpler structure. Models . . . are thus a central necessity of scientific procedure (p. 316).

Simulation and Training Simulators

The wide use of simulation of complex systems is credited to the fact that simulation yields valuable insight into which variables are more important than others in the system and how these variables interact. Moreover, simulation makes it possible to study and to experiment with the complex internal interactions of a given system whether it be firm, an industry or some system of one of these.

Simulation enables one to study dynamic systems, such as power plant simulator, in either real time, compressed time, or expanded time.

"Simulations are sometimes valuable in that they afford a convenient way of breaking down a complicated system into subsystems, each of which may then be modeled by an analyst or team which is expert in that area" (4, p. 373).

A relatively new area where computer simulation is applied is in the construction of power plant training simulators, which exactly duplicate the power-plant control room of an actual (given) plant. Reason (7) describes in his paper:

> A replica of the power-plant control room is constructed, but instead of being connected to an actual plant, all meters and controls are coupled to a large digital computer, into which mathematical models representing the responses of plant systems have been programmed. Depending on the integrity of the modeling, trainees can operate controls and receive signals that exactly duplicate the response of the simulated plant. . . .
>
> Any condition into which the trainee operator could put the plant can be simulated, which increases the realism in the case of a malfunction. It's important to note however, that the simulator only duplicates known plant reponses; it cannot be used to answer engineering questions. . . (p. 125).

He goes on to add,

> . . . simulators cut costs by reducing the amount of time the trainee must spend in the real plant control room, and by reducing or eliminating the number of plant startups and reactivity changes that must be performed for training purposes. They also greatly increase safety, by allowing the trainee to experience emergencies and malfunctions that could not be demonstrated in the actual plant. . . (p. 126).

Although literature on simulation of turbine, generator and boiler modeling is widely published, there is no such literature published prior to this date on simulation of plant logic. One paper by Okunseinde (6) describes some logic on an electro-hydrolic control (EHC) system; however, the paper deals mostly with simulation of turbine speed/load control and hydraulic and valve systems.

Much of the material used in writing this paper is from non-standard sources such as functional drawings provided by the customer (1).

Functional specification drawings (FSD's) supplied by Southern Company Services (1) provide the actual functional logic of the plant.

Wagner (10) describes the design specifications for simulation of control panel 6 logic, and LaGaipa (3) gives the operation and usage of the database manager to initialize simulator variables into common shared memory.

Basic operations and working of power plant are given in Kramer (2) which may be helpful to the reader unfamiliar with power plant operation.

Summarizing, this chapter briefly discusses the aspects of simulation and the art of model building and its importance to applications such as power plant simulators for training operators. The inadequate literature published in the area of simulation of plant logic is emphasized, and hence the desirability of this paper. The references used in developing this paper are defined.

CHAPTER III

SIMULATION OF POWER PLANT LOGIC

Introduction

Plant logic[1] plays an important role in the safe operation of a power plant by "deciding" from several input signals coming from different processes. Plant logic interfaces the manual switches on the control panels, with automatic, protective, and permissive signals from different processes, and in turn provides command signals to processes and equipment. Plant logic also provides signals for the display of breaker status, and valve close/open status.

In the power plant simulator, various systems such as main steam system, boiler feed pump turbine system, electrical systems, etc. are modeled in respective modules and simulated using FORTRAN for real-time operation on an SEL 32/55 system. All models are executed in real time by the model executive called MODELX.

Communication among the different models is provided via variables in global memory which are initially entered off-line by the Database Manager. Different models are executed at various frequencies. Possible frequencies at which a model may be executed are 4, 2, or 1 cycle per second (cps). Frequency of computation of the individual models is

---

[1] Basic application terminology is given in the Glossary (Appendix).

TABLE I

TABLE OF MNEMONICS[1]

ADC	Analog input such as from a potentiometer or controller position--entered from the control panel.
DAC	Anaolog out such as display meters on the panel.
DI	Digital input such as control switch inputs from the panel. Has logical status true or fales.
DO	Digital output, such as lights displayed on the panel. Has logical status true or false.
FC	Valve fully closed, logical status.
FO	Valve fully open, logical status.
NAC	Normal after close, breaker memory status.
NAT	Normal after trip, breaker memory status.
NFC	Valve not fully closed, logical status.
NFO	Valve not fully open, logical status.
R/S	Reset input overrides set input of the flip-flop.
S/R	Set input overrides reset input of the flip-flop.
TDDE	Time delay de-energize timer (also referred to as TMDO).
TDE	Time delay energize timer (also referred to as TMPU).
TMDO	Time delay dropout timer (also referred to TDDE).
TMPU	Time delay pick-up timer (also referred to as TDE).

[1]Detailed definitions in Chapters III and IV.

decided by the rate at which the critical parameters are changing. PANEL4 logic module is run at 1 cps.

The following discussion describes the design and implementation of the plant logic for the fossil fuel power plant of Georgia Power Company being designed and built by CE/Southern Company Services. A functional description of various interlocks within the logic is discussed and examples of coding in FORTRAN from the implementation drawings is given.

In summary, plant logic:

1. Is a communication channel between control panel and simulator models.

2. Accepts switch and pushbutton inputs, simulates equipment logic, providing commands to interfacing models.

3. Simulates plant protection logic by monitoring plant variables.

4. Updates visual output lamps and provides annunciation to enable the operator to provide corrective action.

Plant Logic Elementary

An overview of items used by plant logic is given here. Library routines used for simulation of logic is described in the Appendix.

Panel I/O (Input/Output)

Digital Inputs or DI's. All panel switches and pushbuttons provide digital inputs (DI's). DI's are stored as bits in integer half words, called I/O pages (see Appendix). A DI page consists of sixteen (16) DI's corresponding to sixteen different switches and each has a unique item number associated with it.

Digital Outputs or DO's. Equipment or breaker status are indicated by lamps on the control panel by digital signals called DO's (or digital outputs). DO's are arranged similar to DI's (digital inputs) and consist of integer half word DO pages each containing sixteen bits of DO information which have unique item numbers associated with them.

Analog Inputs or ADC's. Analog inputs or ADC's (analog to digital converter) such as potentiometer inputs are provided as sixteen channels per page, each channel representing a real variable. ADC's are used as analog inputs, such as opening position of a valve, to process models.

Analog Outputs or DAC's. Analog outputs or DAC's (digital to analog converter) are process variables such as valve position, generator voltage, steam pressure which are to be displayed on meters. There are sixteen DAC's per page with each DAC stored as an integer half word. A process variable (such as drum level, pressure, etc.) which is calculated as a real value in a process model is converted to an integer half word DAC, by using ISCALE function, before being finally displayed on the appropriate meter on the control panel.

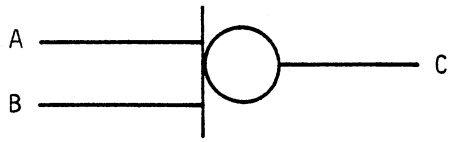
DI's, DO's, ADC's, and DAC's which have similar operations are grouped together as arrays in programming implementations as will be discussed in the following chapters.

Notation

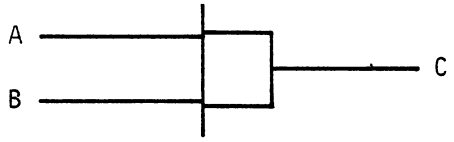
The following section describes the most commonly used symbols and their expressions, which appear on typical logic drawings.[2]

---

[2]See the engineering drawings in the Appendix.



C = A .OR. B

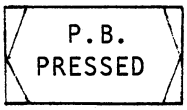


C = A .AND. B

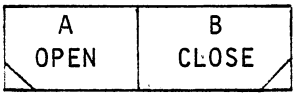
C = .NOT. A

The above three symbols are used to specify logical operations OR, AND, and NOT, respectively.

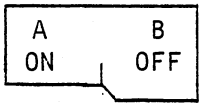
Panel Switches



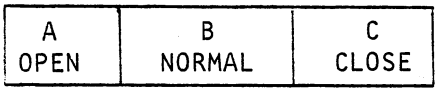
A pushbutton-type switch which generates one DI (digital input status true) when pressed on the control panel.



Two position maintained switch with positions OPEN, CLOSE. When switch is in respective position, one DI (status true) is generated from that position.



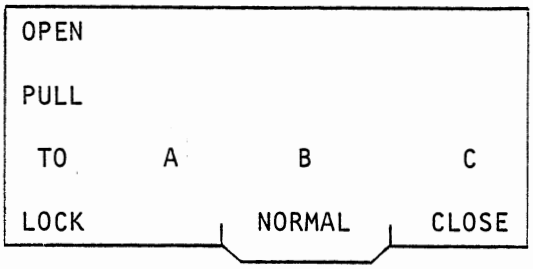
Two position switch with spring return to A. One DI is generated from position A.



Three position maintained switch. One DI from each position.

A	B	C
OPEN	NORMAL	CLOSE

Three position switch with switch spring returns from B to C and B to A. One DI each from positions A and C.

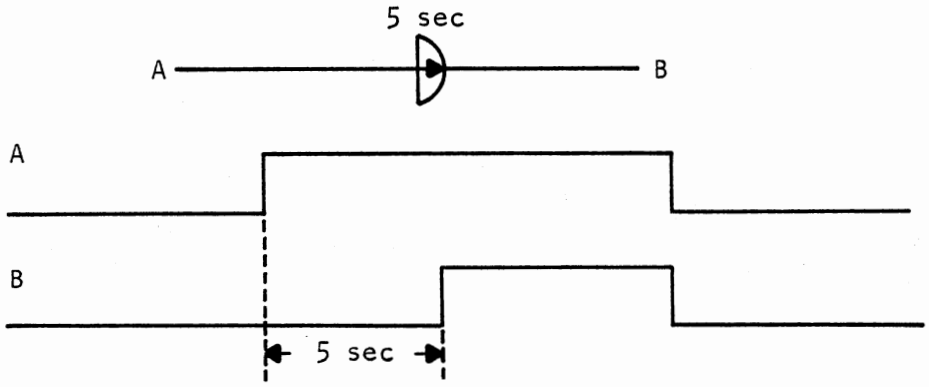


Same as three position maintained switch, except switch can be pulled to lock in position A. Also, the switch can return from B to C and B to A. One DI each is generated from positions A and C.

Other switches work on basic principle of above five switches.

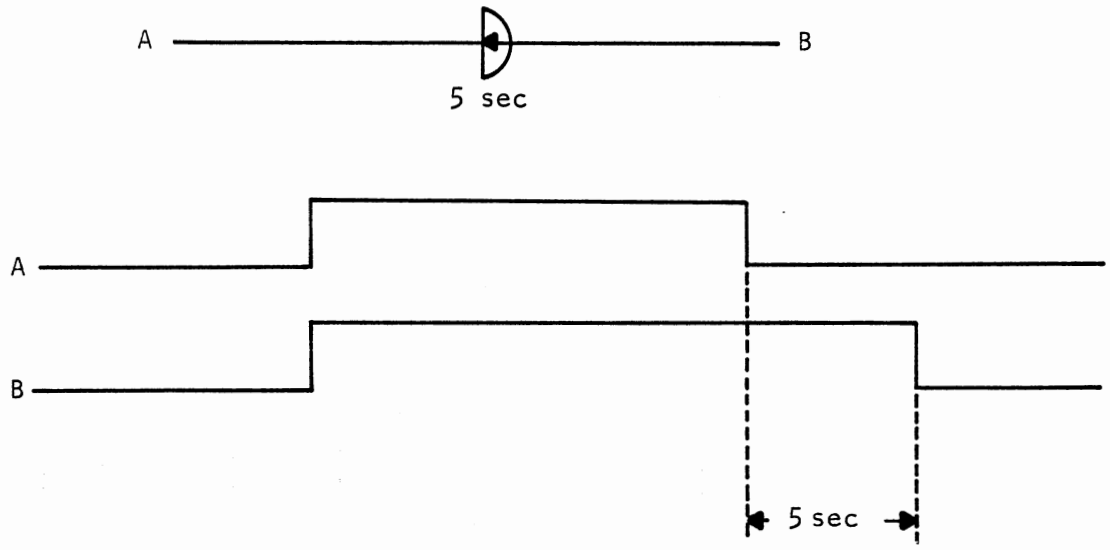
Time Delay Timers

Time delay energize (TDE) or time delay pickup timer (TMPU):



Output B is energized after a delay of n seconds after input is true and remains energized until input goes to false.

Time delay de-energize (TDDE) or time delay droput timer (TMDO):



TDDE output, B is energized for n seconds even after the input signal is removed.

Annunciator

Symbol

Annunciators appear as rectangular displays on top of each panel, and indicate any abnormal conditions existing during the operation of a power plant. Also, abnormal conditions are immediately annunciated to the operator by sounding an alarm.

01					
02					
03					
04					
05					35
06					36

Each box in the annunciator display above has a written annunciator description which flashes in red when an abnormal condition appears.

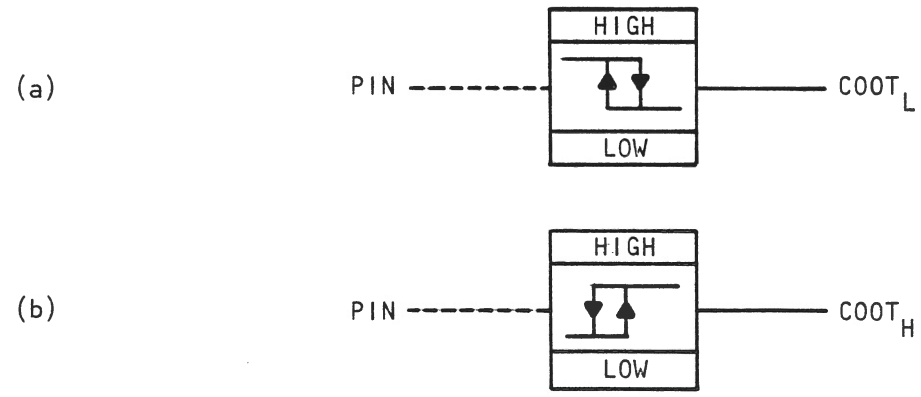
Response to annunciator flashing is done by accepting the acknowledge pushbutton from on any panel which then causes the flashing display to become constant green. When the abnormal condition is completely removed by operator action from the panel switches, the display turns blank.

Lamp Indicators

Symbols for the lamp indicators with the letter within the circle specifying the color of the lamp. The indicator glows only when its DO is set from the logic model. Lamp colors are normally green, red, yellow, neon, and white as they appear on most power plant panels.

Comparators

Comparators are of two types: (a) low type, and (b) high type, and are represented symbolically as:



Comparator output status, COOT, is set to true when process input (a real variable indicated by the dashed line) is less than low setpoint for (a) and greater than high setpoint for (b). Logical equations determining comparator output status for (a) and (b) are:

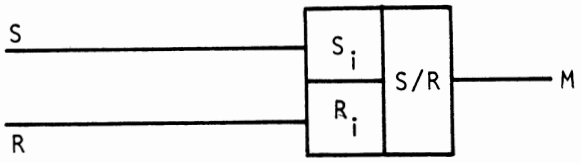
$$COOT_L = (PIN\ ,\ LOW) + COOT_L$$

$$COOT_H + (PIN\ .\ HIGH) + COOT_H$$

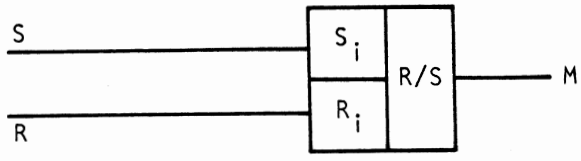
respectively, where $COOT_L$ and $COOT_H$ are memory elements and hence retain their previous status. Simple comparators are one with the same set point for both high and low, and hence have no hysteresis.

Flip-Flops

Flip-flops are of two kinds, where: (1) set overrides reset (S/R) input, or (2) reset overrides set (R/S) input, when both the S and R inputs are true.



$$M = S + M.R$$

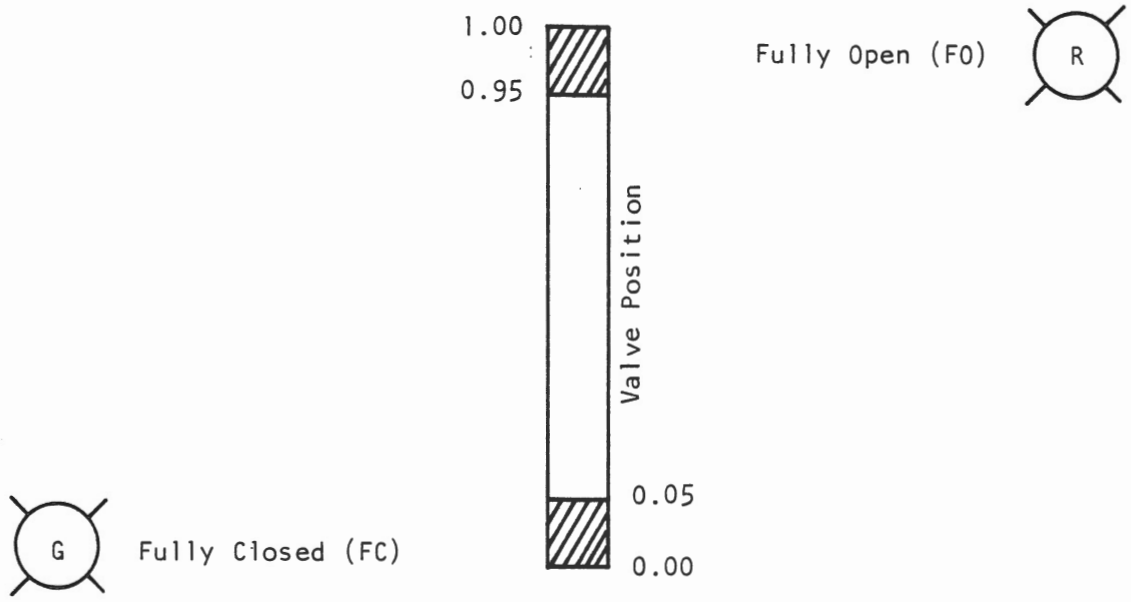


$$M = (S + M)\ .R$$

Symbol representation and logic equations are as given above, where M is the flip-flop memory.

Valves

A valve is an electro-mechanical device which controls the flow and/or pressure of steam, oil, etc. in the smooth operation of a power plant. Valves are of two types: (1) solenoid operated valve--SOV, and (2) motor operated--MOV. The status and position of a valve is calculated by a real-time model subroutine VALVE. Various status are defined pictorially below:



SOV status is defined as either FO (fully open) and FC (fully closed). Similarly, MOV's have defined status NFO (not fully open) and NFC (not fully closed).

Valve position when greater than 0.95 is defined as FO or NFC. Similarly, valve position when less than 0.05 is defined logically as FC or NFO. These status are used extensively in power plant logic modelling.

Standard Naming Conventions for Variables

A standard variable naming convention is followed for defining variables for power plant simulators. Standard naming provide ease in clean interfacing between different models and to isolate problem areas when they exist. As an example, all control program variables begin with letters XB, turbine program variables with TB, and so on. Each modeler has to comply with the satndard naming conventions.

Details of the naming procedure are omitted here. It is felt it is sufficient to mention their existence and they are strictly followed in their usage to be able to readily identify variables with their source models.

Summary

In this chapter, terms such as DI, DO, ADC, and DAC used to define input and output to panel are introduced. Stadard symbols which appear most commonly on the logic drawings are also given. MODELX, which is the simulator model executive and schedules the execution of modelling programs to run at any one of the frequencies, 1, 2, or 4 cps, is also introduced. Finally,the importance of standard naming conventions for simulator variables is briefly discussed.

CHAPTER IV

FUNCTIONAL CONTROL DRAWINGS

FCD's (Function Control Drawings) are the original drawings provided by the utility company and give the actual plant logics. However, FCD's are much too detailed and can be simplified based on several assumptions to provide design specification for the simulator logic.

Design Specifications

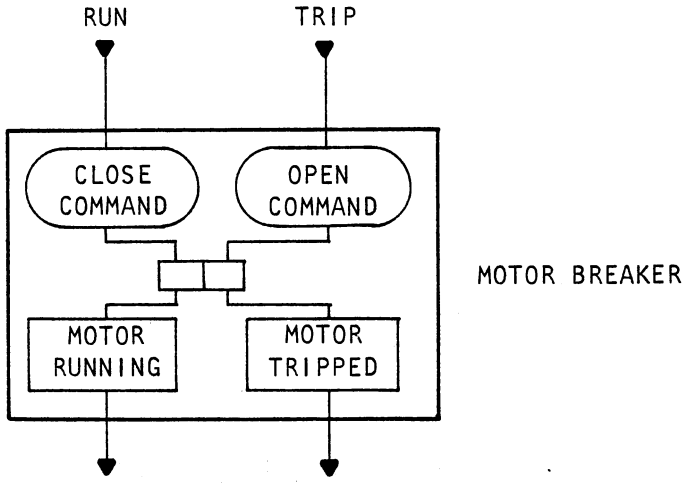
Design specifications define the scope of simulation of plant logic as applied to control console panel inputs and interface with other models.

Following are some of the assumptions based on developing design specification. All assumptions are not listed here, since it would considerably add to the bulk of the writing.

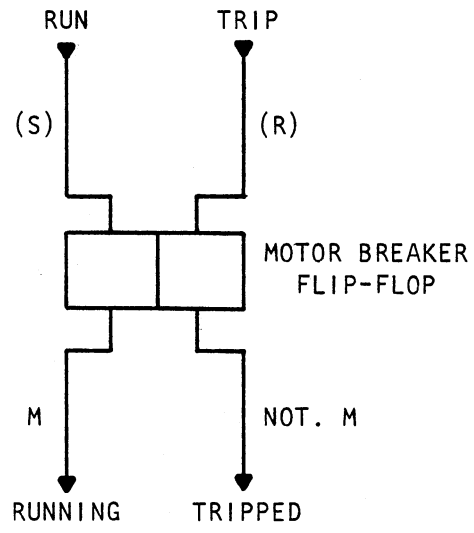
1. Breakers have a delay time between the time a breaker close command is given and the breaker actually closes. This delay time will not be simulated as breakers are simulated as flip-flops with their outputs reflecting the input command (see Figure 1).

2. The ON-OFF times of control air supply line of SOV's are short enough so that their output is taken as equivalent to the command input (see Figure 2).

3. Redundant measurement channels where any malfunction will not cause different outputs are simulated as one channel (see Figure 3).



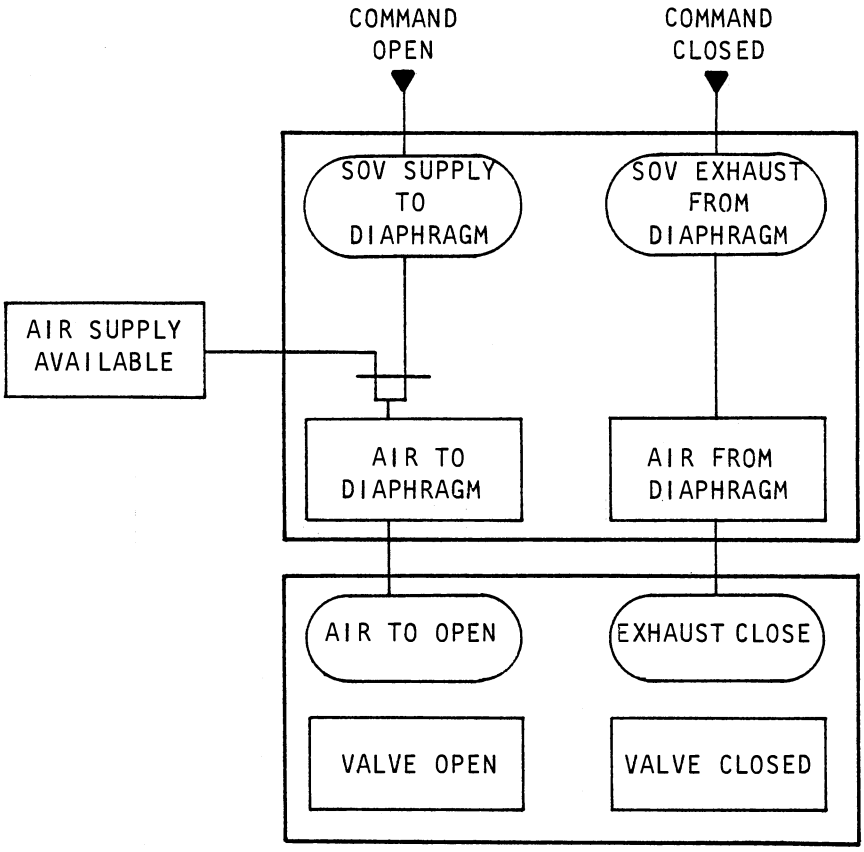
(a) Typical Original FCD



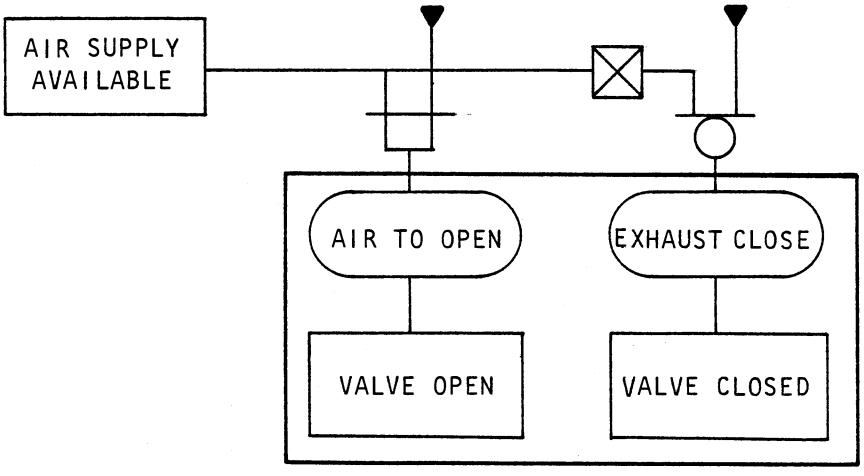
(b) Simplification

(c) Implementation: This is implemented as an R/S flip-flop with TRIP as the R-input, RUN as the S-input, and M as the Motor-Running status. Note that motor-tripped is negation of M.

Figure 1. Breaker Simplification

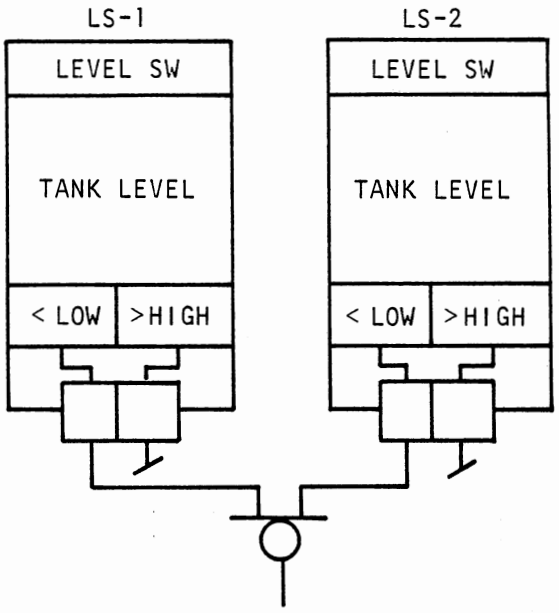


(a) Typical Original FCD (Air to Open)

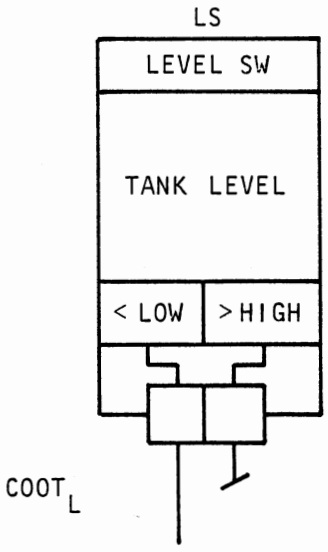


(b) Simplification (Air-to-Open)

Figure 2. SOV Simplification



(a) Typical Original FCD



(b) Simplification

(c) Implementation: This may be implemented as a comparator, COOTL, given the tank level, TL (computed by the appropriate process model); COOTL = (TL < LOW) + COOTL.

Figure 3. Redundant Measurement Simplification

4. Detailed simplifications are marked on the FCD's itself.

Control panel 4 logic (hence, logic module PANEL4) includes:

Forced Draft (FD) Fans and Lube Oil Pumps (LOP's)

Induced Draft (ID) Fans and Lube Oil Pumps

Primary Air (PA) Fans and Lube Oil Pumps

Gas Recirculation (GR) Fans, Turning Gear Motors (TGM's) and Lube Oil Pumps

Air Preheaters

Boiler Penthouse Pressure Fans

Pulverizer Seal Air Fans

Lighter Oil Pumps

Furnace Temperature Probes

Boiler Circulation Pumps (BCP) and Cooling System

Boiler Feed Pumps (BFP) including Master Trip, Speed, Lube Oil, and Turning Gear Control

Electromagnetic Relief and Isolation Valves

Economizer Inlet Motor Operated Valve (MOV)

Scanner Air Fans

Ignitor Air Fans

BFPT Vapor Extractor Blowers

Extraction to Heater 7 Isolation MOV

Extraction to Air Preheaters MOV

BFP Discharge MOV's

High Pressure (HP) Heaters 7, 6 Inlet and Discharge Isolation MOV's

System Block Diagram

PANEL4 logic model interfaces with several other simulated models

and systems as shown in Figure 4. These are complex models and are beyond the scope of this paper. However complex a model may be, it does not fit into the whole design completely until all the interfaces between models are properly defined. The types of interfaces received and sent from PANEL4 logic model include:

Instructor's Console. Interface with the instructor's console consists of modifying the plant parameter, setting the plant equipment and malfunctions from the instructor's console.

DDS. Digital data system receives equipment status and alarms from the logic module PANEL4 and displays them on video display screen.

Control Panel 4. Supplies digital inputs (DI) from switches, outputs DO's to lights and displays process variables on meters.

Boiler Feed Pump Turbine. BFPT receives equipment status from PANEL4 module, and supplies process inputs. BFPT runs the BFP which in turn supplies water under pressure to the boiler drum.

Furnace Safeguard Supervisory System (FSSS). The FSSS model receives equipment status, and sends protective interlock signals to plant logic model; supervises the safe operation of the furnace.

Valve Model. Receives command signals and sends valve status; fully open/fully closed (FO/FC), not fully open/not fully closed (NFO/NFC).

Motor Model. Receives breaker status from panel logic, and sends motor overcurrent protection signals.

Compressed Air. Sends control air supply status.

Electrical System. Establishes power supply status, load strips for plant logic model.

Air/Fuel, Boiler/Drum, Main Steam. Supplies process inputs such as pressure, temperature, flow PANEL4 logic model.

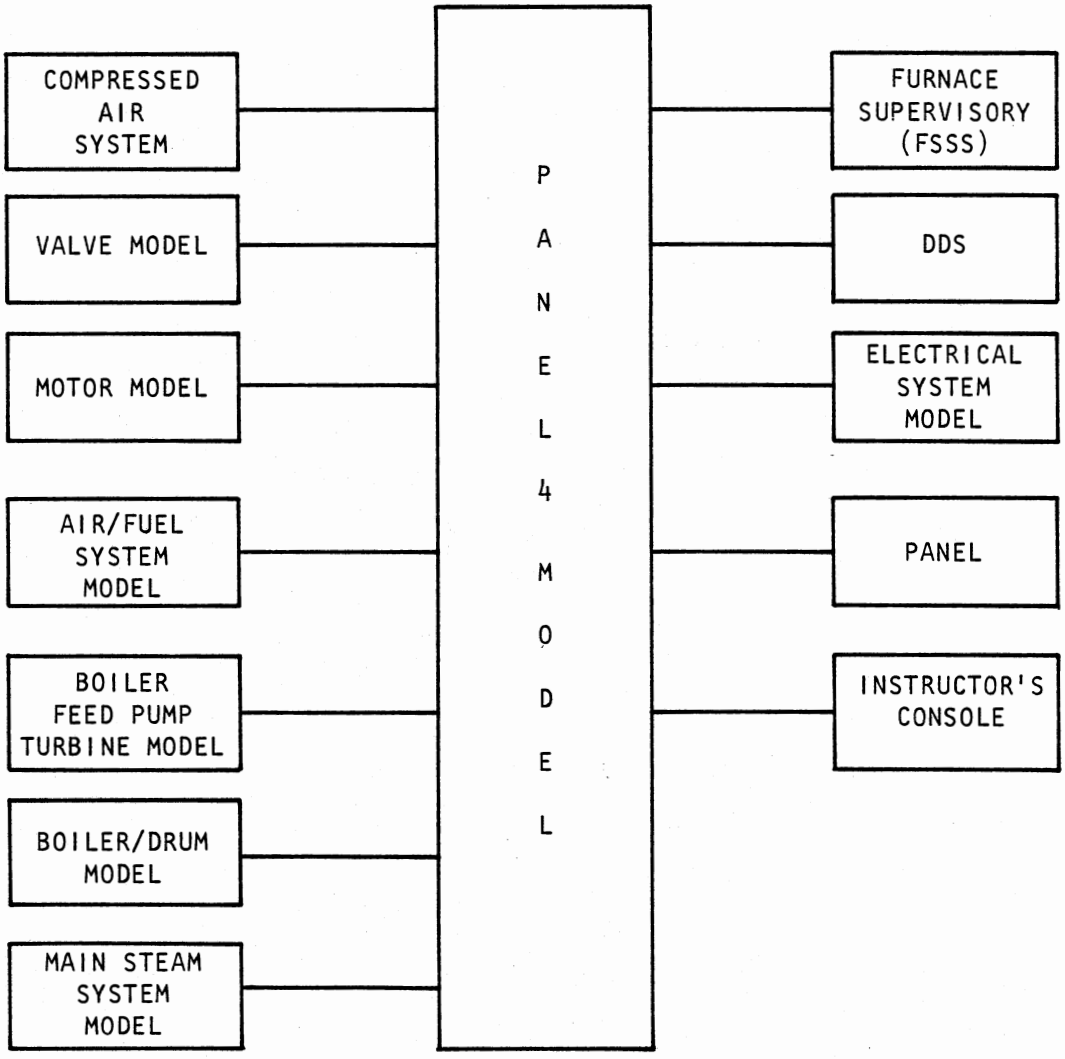


Figure 4. Simplified Block Diagram

CHAPTER V

IMPLEMENTATION SPECIFICATION

Implementation and coding of PANEL4 logic is done by first developing implementation drawings which give the detailed plant logic in a format, which can be coded directly in FORTRAN. In fact, implementation drawings may be viewed as parallel charts of the actual code.

Implementation specifications are obtained from design drawings (FCD's), by grouping logic of similar operations where possible, and following the below stated conventions for optimization of real-time storage and execution time:

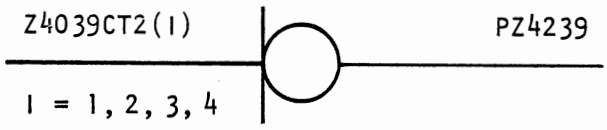
1. Similar logic is grouped together and represented by heavy lines on the implementation drawings. Linear arrays are used as data structures in coding the multiple elements represented by heavy lines. For example, referring to the implementation drawing of BFPT 1A (1B) Main Oil Pumps A and B of sheet 20, both pumps have similar logic and/or are presented by all heavy lines. Digital inputs (DI's) from manual switches are also grouped as arrays. Grouping this way reduces a large amount of code, which is not only invaluable to real time operations, but also considerably reduces debugging time. This is due to the fact that when all the logic for one element of the group (such as say BFPT 1A MOP A in the above example) is tested, then the reset of the basic logic can be assumed to be working correctly.

2. Normal after close (NAC) breaker logic is implemented with a R/S

flip-flop; where the memory output is true implies NAC breaker status, otherwise NAT (Normal After Trip).

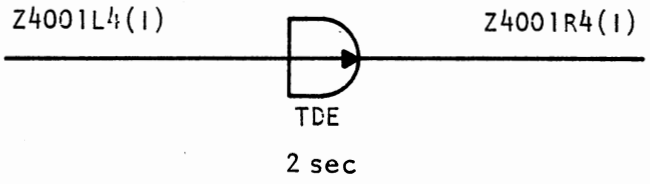
3. The breaker logic itself is implemented with a R/S flip-flop with memory output as YBS050(I), which becomes true when the breaker is closed and false when the breaker is tripped.

4. An OR gate which has a heavy line as input, and a thin line as output, such as shown below:



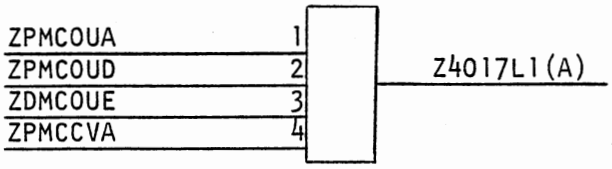
has the output defined as the logical OR operation of all its inputs. A logical AND gate is similarly defined.

5. Time delays such as TDE and TDDE may also be grouped and indexed together as shown in lube oil pumps drawing and drawn below.



Library subroutines EFTMDO/EFTMPU are used to implement the time delay.

6. Certain distributed interfaces which are needed to be grouped to process similar logic sections (grouped as arrays) are shown as the following blocks on the implementation drawings.



where the block output Z4017L1 is an array with its individual elements

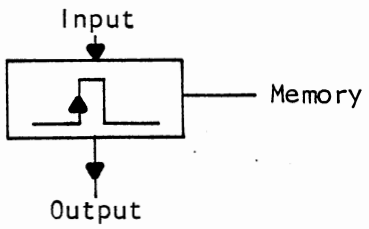
elements initialized to variables on the input side of the block.

7. Where interface variables appear as consecutive memory locations in global memory, FORTRAN equivalences are used in the coding.

8. Interfaces to/from various systems and/or other source drawing are defined with a brief description and destination/source specified to make sure the interfaces are correctly matched.

9. Plant equipment is an interface input set from the instructor's console.

10. The rising edge memory of the breaker is represented by symbol:



and gives a true output only when input goes from false to true (rising edge of input pulse). The rising edge prevents the automatic closing of the breaker, which could otherwise happen immediately after an emergency trip is removed and hence requires the operator action to manually trip the breaker before being closed.

In the discussion above (steps 1 through 10) of the implementation specifications, it was not clear how the FORTRAN variables were named on the implementation specifications. It is after the completion of each implementation drawing that variables, such as described in items 1-10 above, are named and printed on the drawings. Most of these variables, namely the flip-flop memory, interface variables to other modules, timer outputs, etc. are then initialized into the global memory by the database manager (see Appendix). For a typical power-plant simulator nearly

20,000 variables (from all simulation models combined) are entered in global memory.

Data Structures and Programming Techniques

The only data structures required to implement the plant logic PANEL4 are simple linear arrays. When similar logic is grouped, all variables appearing alongside heavy lines are FORTRAN LOGICAL*1 linear arrays, declared as a dimensioned variable. The individual elements of the array correspond to the individual elements of the grouped logic which in the case of ID fans represent the logic for ID Fan 1A, 1B, 1C, and 1D, respectively.

ID fan logic is implemented within a FORTRAN DO-LOOP with an index I, and logical equations coded within the body of the loop.

Motor breaker logic represented by R/S flip-flop has the logical equation:

```
(a)  YBBA009(I) = (Z4017L6(I) .OR. YBBA009(I))
                       .AND. .NOT. Z4017L5(I)
```

within the DO-LOOP. All coding must be in top-down sequence, in order to avoid initialization problems. (Similarly, coding each of the drawings in order of sequence numbers makes PANEL4 code top-down.)

The NAC memory of the breaker logic is coded as:

```
(b)  Z4017S(I) = (P4017TC(I) .OR. Z4017SCI))
                       .AND. .NOT. P4017TT(I)
```

Note that the code for (b) should precede (a) within the DO-LOOP body for consistency, and also since (b) is used along with (a) to generate P4017MA(I), the yellow indication light,

```
P4017MA(I) = .NOT. YBBA990(I) .AND.
             Z4017S(I)
```

The equation of P4017MA may be at the bottom of the loop.

For items 4, 5, and 10 discussed above, the FORTRAN implementation is:

Item 4:
```
PZ4239 = Z4039CT2(1) .OR.
         Z4039CT2(2) .OR.
         Z4039CT2(3) .OR.
         Z4039CT2(4)
```

Item 5:
```
CALL EFTMPU (Z4001R4(I), Z4001L4(I),I)
```

Item 10:
```
OUTPUT = INPUT .AND. .NOT. MEMORY
MEMORY = INPUT
```

In item 10, initially, MEMORY is false. OUTPUT is true only when INPUT goes from true to false, since MEMORY was previously set to INPUT. The duration of OUTPUT pulse is equal to the inverse of the frequency of execution of PANEL4 program. (Since PANEL4 is executed at 1CPS the duration of OUTPUT pulse is 1 second in real time.)

Application of Boolean Algebra

In preparing the implementation drawings from FCD's (functional control drawings), application of De Morgan's Law is extensive, whenever appropriate, and by minimizing the logic the code is effectively minimized. In several instances logic could be simplified to reduce or eliminate several redundant AND/OR gates by applying basic rules of Boolean algebra. This is required in real-time programming to efficiently reduce the code and obtain fastest execution time.

Specific examples are not possible due to the size considerations of this report, but it is sufficient to mention that significant elimination of redundant logical operations is achieved by extensive application of Boolean algebra.

The FORTRAN code is generated from the implementation drawings and is tested with the rest of the simulator by successful compilation and cataloging of PANEL4 subroutine; its functional response and the behavior expected is the topic of discussion in the remainder of this chapter. Protection and interlocks for the logical operation of plant equipment logic are described in the following functional description. Simulator logic (in the form of PANEL4) is required to work exactly as per the functional descriptions of the actual plant, and is verified in vigorous detail against the sample functional descriptions given in the following pages.

Functional Description

A functional description as related to the operation of plant equipment along with interlocks and protection signals is detailed here.

The conditions required for the closing of a breaker or protection to cause the equipment to trip and generate appropriate annunciation is presented in the following paragraphs.

1. Startup Standby BFP (Sht. 5):
    a. The SU/STDBY BFP can be started manually when no trip conditions appear and by operating the switch to the close position.
    b. Either loss of power on 6.9 KV 1A load strip relay or low deaerator storage level will cause the pump to trip.

c. The red lamp indicates the breaker is closed. The green lamp indicates the breaker is tripped (either manually or by any protection signal). The yellow lamp implies the existence of a trip condition, since the switch is in normal after close (NAC).

d. SU/SB BFP tripped annunciation is given on abnormal trip.

2. Gas Recirculation Fans (GRF) (Sht. 7):

a. To start a GR fan, its inlet and outlet dampers must be fully closed and neither trip conditions existing. Closing the switch causes the breaker to close and the motor to start. This is indicated by the red lamp.

b. The GR fan can be stopped manually from the switch indicated by the green lamp coming on.

c. When a GR fan is running and any of the following trip conditions exist:

(1) Power failure on 6.9 KV load strip

(2) Master fuel trip, or

(3) Malfunction overcurrent trip,

appropriate annunciation is given and the GR fan motor stopped. Also, the yellow light will come on.

3. Induced Draft Fans (IDF) (Sht. 9):

a. ID fans are prohibited from starting until the following conditions are satisfied:

(1) ID fan inlet valves and output dampers are fully closed.

(2) ID fan inlet valves in manual control.

(3) Forced draft fan inlet valves fully closed, and outlet dampers fully open and FDF running.

(4) Lube oil pressure is normal and lube oil pump is running and no trip conditions exist.

b. Automatic trip occurs under the following conditions:

(1) Loss of power on load strip.

(2) ID fan outlet damper is not fully closed and forced draft fan is tripped with switch in normal after close.

(3) Lube oil press is not adequate.

(4) A malfunction exists.

4. Forced Draft Fan (FDF) (Sht. 13):

a. Forced draft fans are prohibited from starting until the following permissives are satisfied:

(1) Lube oil pressure is normal and lube oil pump is running.

(2) FD fan inlet and outlet dampers are fully closed.

(3) Any ID fan is running.

(4) FD fan station is in manual and no trip conditions exist.

b. FD fans will be automatically tripped when any of the following conditions exist:

(1) Lube oil pressure is not adequate.

(2) No fuel in boiler and external pressure excursion.

(3) Power failure on 6.9 KV bus load strip.

(4) Presence of a malfunction.

(5) All ID fans are tripped. Green and yellow lamps will light up and FD fan breaker tripped annunciation will be given on any of the above trip conditions.

5. Boiler Feed Pump Turbine Main Oil Pump (BFPT MOP):
   a. The main oil pump can be started and stopped manually by operating the switch on the panel to run, or the pull to lock off position, respectively.
   b. BFPT MOP starts automatically when the switch is in auto position and the hydraulic oil pressin the BFP turbine goes lower than the specified level.
   c. The pump is tripped when there is loss of power on 480V load strip or when there is a malfunction.
6. Boiler Circulating Pumps (BCP):
   a. The BCP is started manually from the switch when there is no trip conditions at the time of starting.
   b. The following protective signals will cause the BCP to trip:
      (1) High motor cavity temperature;
      (2) Loss of power on 4.18 KF load strip; or
      (3) A malfunction.
   c. Boiler circulating pumps can be stopped manually by operating the panel switch to trip.
   d. Annunciation will be given when circulating cooling water flow is low and when motor cavity temperature is high.

CHAPTER VI

SUMMARY, CONCLUSIONS, AND FUTURE WORK

This paper describes the design and implementation of a power plant logic model, called PANEL4, for power plant simulators, beginning with functional control drawings (FCD's), continuing with the implementation drawings, which then are coded in FORTRAN directly for real-time execution using tools such as Database Manager.

Apart from what is presented in this paper, the following tests were performed, and are summarized here.

The next step involved is an off-line test of the implemented code on a "test-station" to guarantee the exact reproduction, without loss of information, from the source it was coded, i.e., implementation drawings. This kind of test assures no later surprises, during merging with other models, such that no code was overlooked or miscoded. At a later stage, when other models of the simulator were also ready, models were merged together. At this point the actual functional and operational testing of plant logic was done. Interfaces between corresponding models must be accurate, missing interfaces, if any, are correctly hooked-up, etc. The merging process was achieved without many surprises, since off-line testing was done quite thoroughly. The acceptance test procedure (ATP) by the customer was also completed successfully, and the simulator is now installed on location.

The basic design and implementation philosophy is the same for future simulator logics. However, since the functional logic varies quite widely from plant to plant, and FCD's are specific to a given plant, no generalization can be made as to the usage of one plant logic model on other. Based on the experience, some conclusions can be drawn as to further improvements.

1. Panel logic models may be run on an event-oriented basis instead of cyclically. This is due to the fact that plant logic has inputs and hence outputs based on events such as operator inputs from the control panel; comparator outputs based on change of status such as temp low/high. Such events are rare and may trigger the model only when necessary. This will considerably reduce the execution time of plant logic models in real-time.

2. Large plant logic models may be broken down to sub-modules, since a typical plant logic model requires as many as 2500 lines of code. Compilation and cataloging of these large models during development period takes an excessively large amount of time--just to correct a simple line of code. It may be a good idea to have a sub-module for each implementation drawing--but this is another extreme--since calling too many subprograms in real-time simulation operation where time is the most critical factor, may slow down the simulator a bit. An intermediate approach, such that few sub-modules are used instead, may be the one best to choose.

A SELECTED BIBLIOGRAPHY

1. Functional Specification Drawings (FSD's) from Georgia Power Company.

2. Kramer, A. W. Power Plant Primer. Barring, IL: Technical Publishing Company.

3. LaGaipa, J. Database Manager. W. Long Branch, N.J.: EAI, 1975.

4. Morgenlhaler, George W. The Theory and Application of Simulation in Operations Research. Ed. Russell L. Ackoff. New York: John Wiley and Sons, 1961.

5. Naylor, T., C. Balintfy, J. Burdick, and L. Chu. Computer Simulation Techniques. New York: John Wiley and Sons, Inc., 1968.

6. Okunseinde, O. Simulation of Large Turbine Control. W. Long Branch, N.J.: EAI, 1979.

7. Reason, J. "Full-Scope Simulators: Vital Tools for Nuclear Training," Power, Vol. 123, No. 7, (July, 1979), pp. 123-129.

8. Rosenbleuth, Arturo, and Norbert, Wiener. "The Role of Models in Science." Philosophy of Science, Vol. 12, No. 4, (October, 1945), pp. 316-325.

9. Shubik, Martin. "Simulation of the Industry and the Firm." American Economic Review, L, No. 5 (December, 1960), pp. 908-919.

10. Wagner, R. Design Specification for Panel 6. W. Long Branch, N.J.: EAI, 1979.

APPENDIX A

GLOSSARY OF DEFINITIONS

Annunciator. Annunciator window on a given panel operates (or alternately flashes) when initiated by logic--and indicates an abnormal condition.

Automatic Signal. A signal which is used to automatically close or open a valve; automatically close or open the circuit breaker (defined below).

Breaker Status. The status of a circuit breaker is a logical signal specifying whether open or close at a given time. Breaker closed, breaker open are the only two valid breaker status values.

Circuit Breaker. Switching device designed to open a current-carrying circuit under abnormal current conditions without injury to itself. It must also be capable of interrupting short circuit currents.

Database Manager. A centralized off-line (not real-time) processor which enters variables in global memory, into specific partitions (real, logical, constant variable subpartition) defined by entry command of database control cards. The database manager together with JCL is used to retrieve database variables from a database during compilation of the model.

Frequency of Execution. All models in a power plant simulator are executed in real-time at one of the frequencies: (1) once every cycle (1 cps since cycle is 1 second in real-time), (2) twice every cycle (or 2 cps), or (3) four times a cycle (or 4 cps). Models are spaced in time at 4 cps, 2 cps, or 1 cps depending on their criticality to the overall system execution. The spacing and execution (and timing) of models is done by a special processor called Model Executive (or MODELX) which runs continuously until the simulator is aborted or suspended.

Global Memory. Is the shared memory used to communicate between different simulation models. Variables in shared memory, also known as database variables are entered by database manager. Database variables (logicals, real, integer, etc.) have to be entered by database manager, in global memory, only once, and any number of models may use the same variable for inter-communication.

Variables in data base used by different models is identified during compilation using special control cards and data-base manager retrieval routines to retrieve the appropriate names defined in the data base.

I/O Page. An I/O page such as for DI's (Digital Inputs) and DO's (Digital Outputs) are arranged as 16' bits of data, equivalent to a software representation of integer half-words, transferred between hardware (i.e., panels) and software (i.e., simulator models). The use of 16' bits instead of any other number, is a hardware dependent and beyond the scope of this discussion.

I/O Page, similar to DI's and DO's, also apply to ADC (analog Inputs) and DAC's (Analog Outputs) except that they are 16 full-words (equivalent to real variables of software) to a Page. Each word is 32 bits long and hold a floating point value. This choice is also hardware dependent and generally accepted for transfer of floating point data.

In software simulation DI's from the I/O Page, are first converted to (local) logical bytes (using UNPACK routine defined below) before using them in logic, since byte operations are considerably faster than bit operations and execution time is a critical factor in real-time simulation.

Similarly after logic bytes are evaluated, which set the indicating lamps on the panel, must first go through a similar pack routine (CALL PACK defined below) which packs 16 logical bytes into an I/O Page, representing the DO's.

Light. Red light indicates operating, flowing or increasing conditions depending on context used.

Green light indicates not operating, not flowing, or decreasing conditions.

Amber (or yellow) light indicates automatic, standby, or intermediate conditions.

White or neon light indicates manual or protective trip conditions.

Permissive Signal. A signal without which a device may not be operated. Several permissive signals may be needed to activate a device. Example: a certain fan (primary air) may not be permited to start unless other fans (forced draft, induced draft) are not already started.

Power Plant Logic (or Simply Plant-Logic). Consists of plant functional drawings, which specify the conditions under which devices such as valves (see definition), circuit breakers (see definition) etc., may be permitted to operate, or prohibited from operating, depending on various conditions existing during plant operation.

Plant logic provides commands to operate devices; or dinicate via lamps on the control panel, the status of devices under operations. If any abnormal conditions occur during plant operation, an annunciation command is given (in the form of sound and light flashing on the appropriate annunciator bank specifying the abnormal occurrence of the condition) to attract the operator's attention.

Protective Signal. A signal which will prevent the operation of a device to which the signal is applied. The device cannot be operated as long as the protective signal exists and hence protects the device.

Valve. A valve is a device which permits the flow of steam, hot/cold water, lubricating oil etc., when fully or partly open, and prohibiting the flow when fully closed. A valve may be of three kinds: Motor Operated Valves (MOV), Pneumatically Operated Valves, and Solenoid Operated Valves (SOV).

Valve Status. The status of a valve is also a logical signal specifying whether a valve is fully open (FO), fully closed (FC), not fully open (NFO), or not fully closed (NFC). Valve position (0.0 for fully closed, 1.0 for fully open) is used to determine the different status of valves: FO, FCF NFC, NFO.

Library Subroutines. Library routines used most commonly in plant logic models are PACK, UNPACK, EFTMPU, & EFTMDO. Their calling sequence and argument list is described below for completion and will not be discussed any further. These are the standard library routines developed for the simulator.

PACK: Packs 16 logical bytes into a integer half word DO page.

CALL PACK (DO, BYTE, NUMBYTE)

DO Starting half word location for packing

BYTE Starting address of bytes to be packed

NUMBYTE Number of bytes to be packed

UNPACK: Unpacks from half word page DI's to logical bytes.

CALL UNPACK (DI, BYTE, NUMWORDS)

DI Starting half word location of DI pages where switch inputs are stored as bits.

BYTE Starting address of unpacked bytes.

NUMBYTE Number of half words (DI pages) to be unpacked.

Time Delay Pickup Timer (TDE of 2.4.1)

CALL EFTMPU (LOUT, LIN, INDEX)

LOUT Logical byte output of the timer, delayed by t seconds.

LIN Logical byte input of the timer.

INDEX Index to an array containing delay time in seconds.

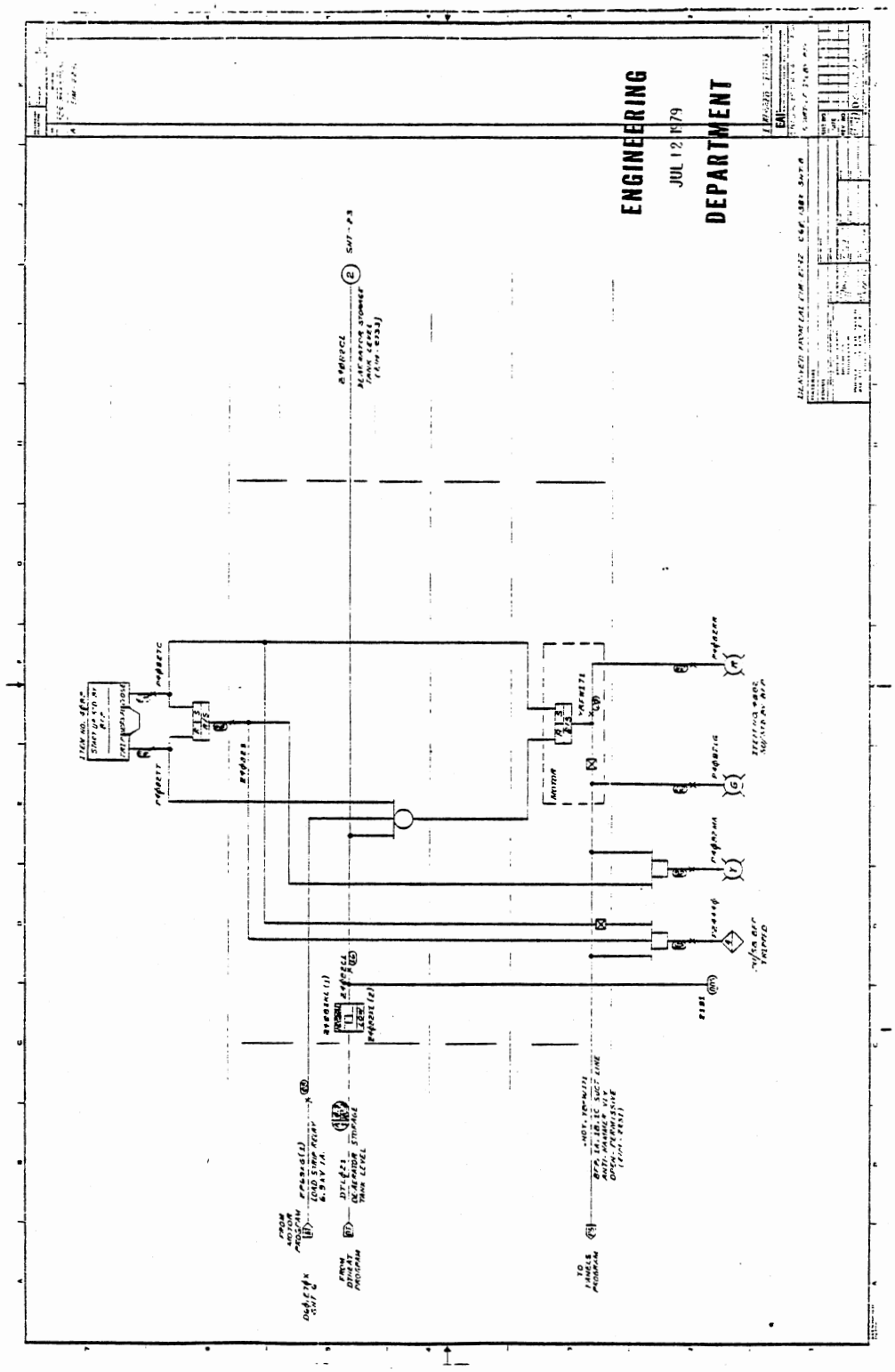
Time Delay De-Energized Timer (TDDE of 2.4.2)

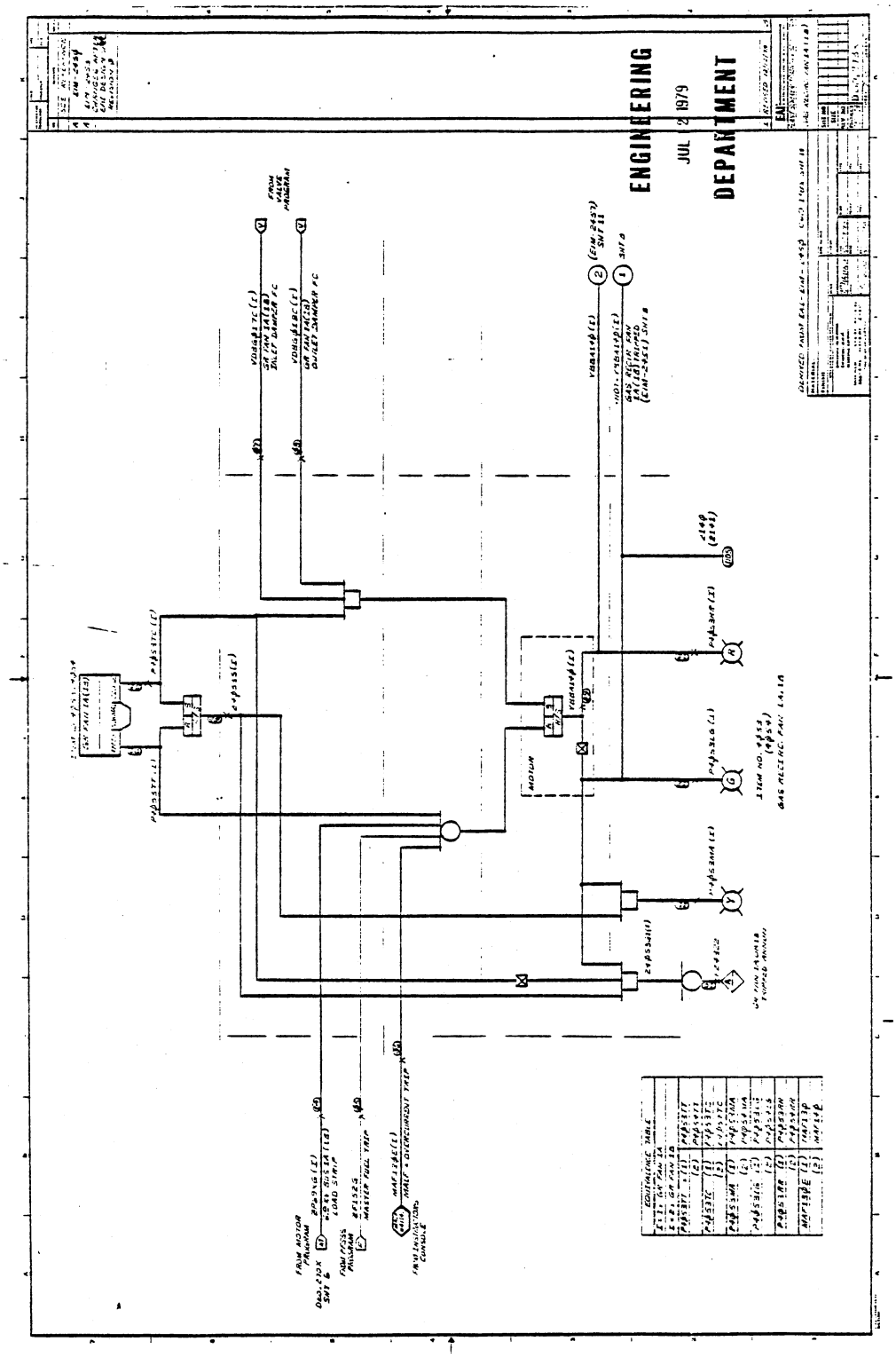
CALL EFTMDO (LOUT, LIN, INDEX)

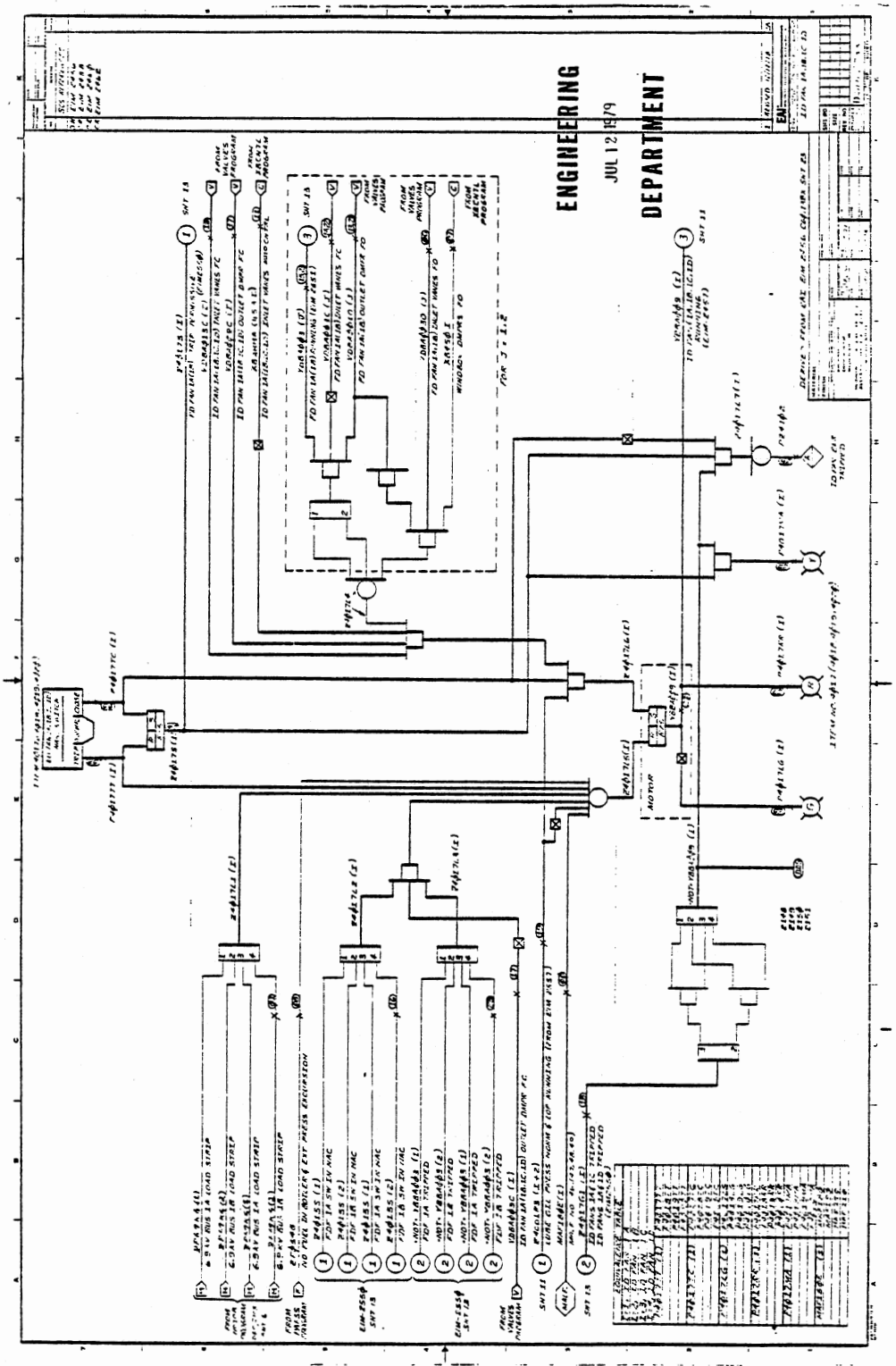
LOUT, LIN, INDEX as defined above

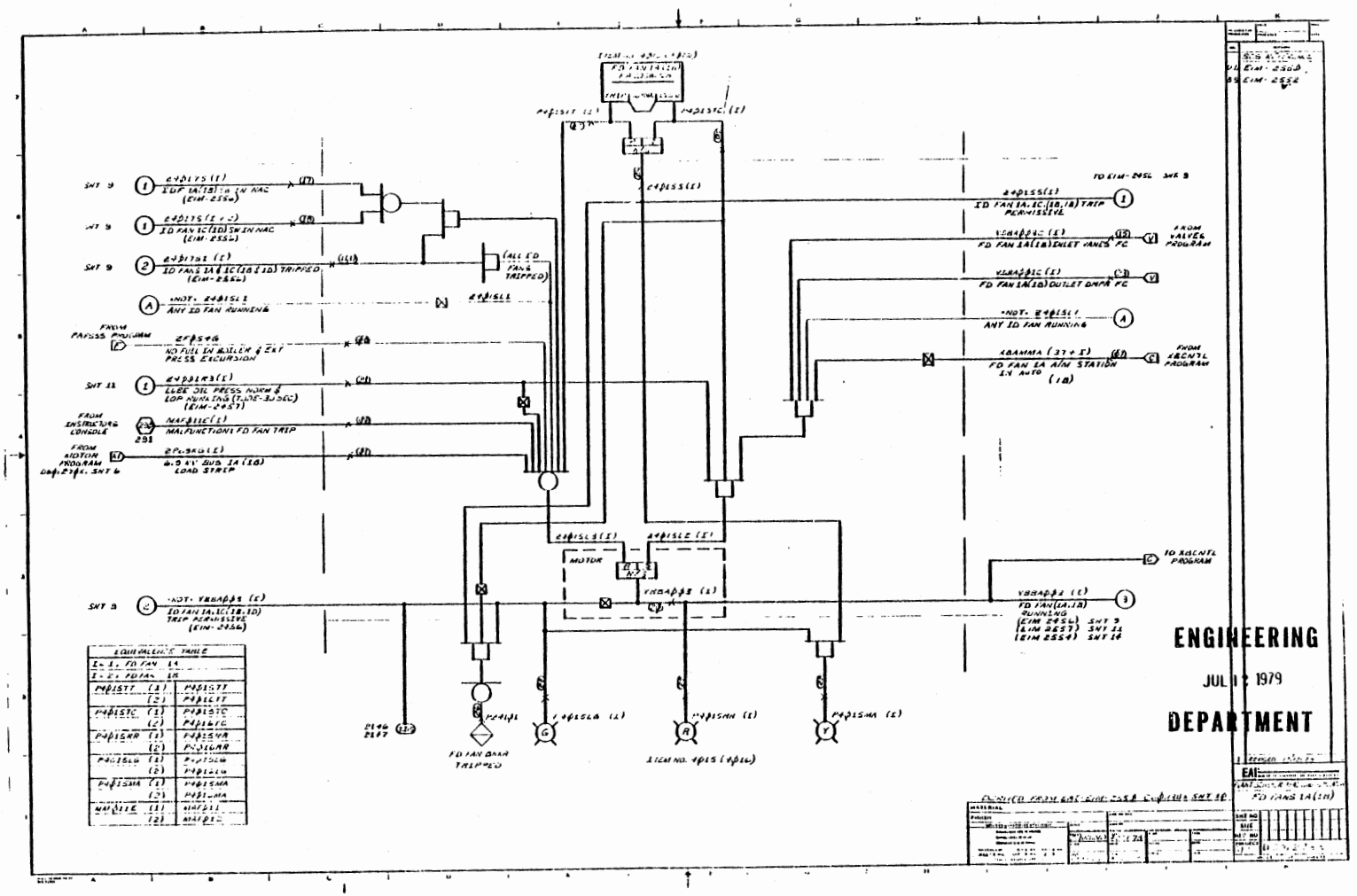
APPENDIX B

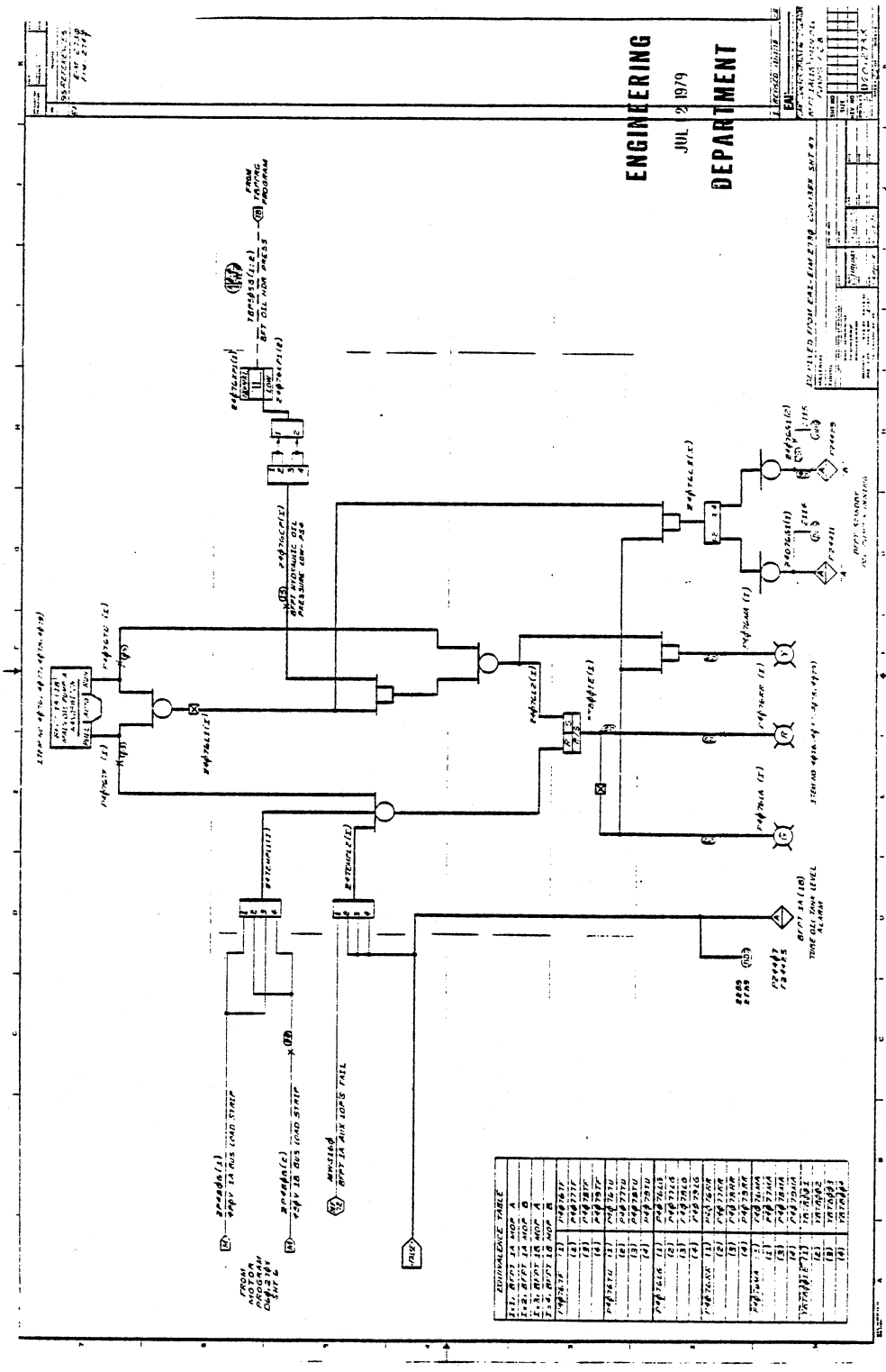
FIGURES

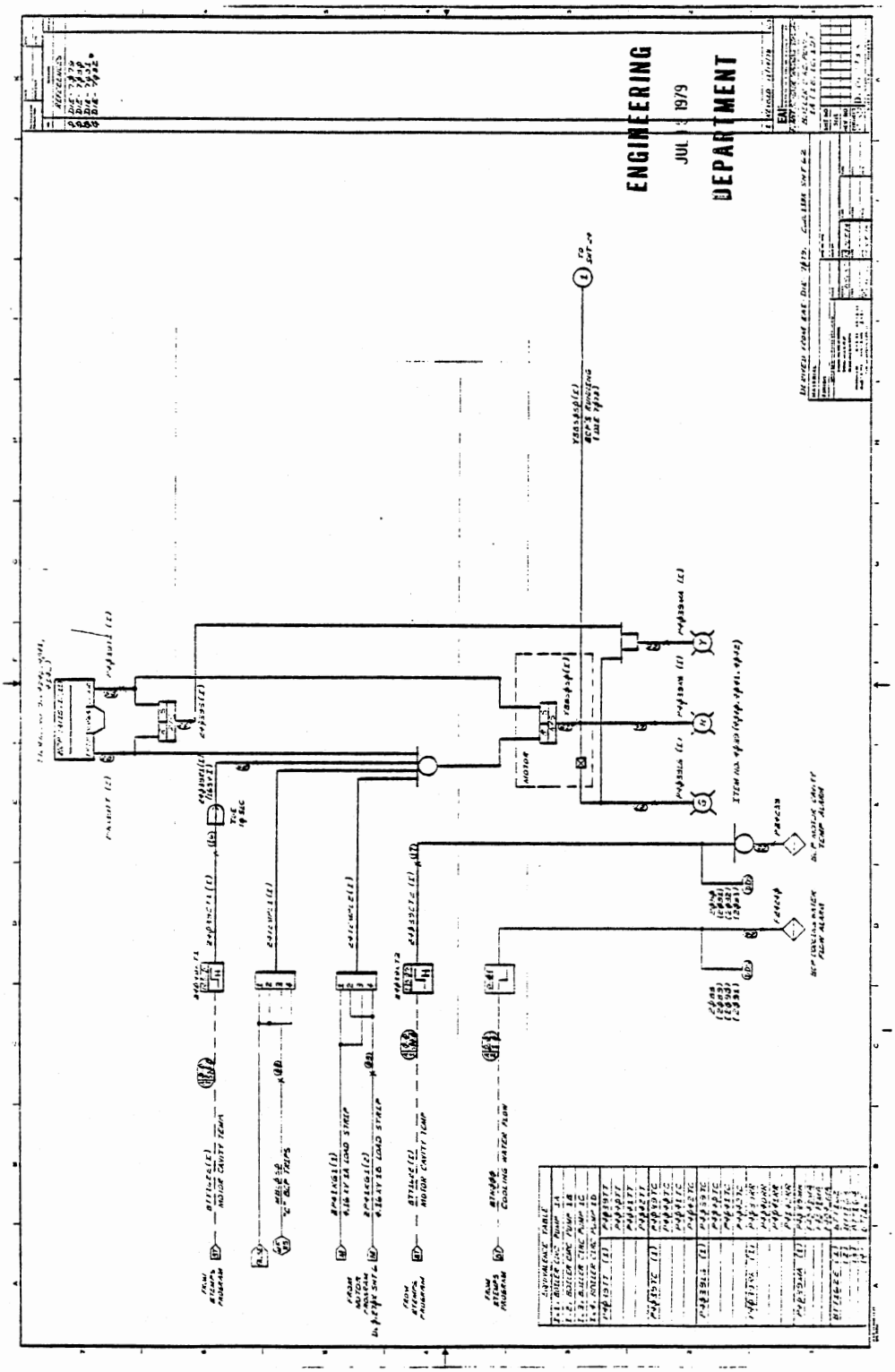

ENGINEERING
JUL 12 1979
DEPARTMENT


ENGINEERING
JUL 2 1979
DEPARTMENT


ENGINEERING
DEPARTMENT
JUL 12 1979


ENGINEERING
JUL 1 2 1979
DEPARTMENT
FD FANS 1A(1B)
EQUIVALENCE TABLE
FD FANS 1A
FD FAN 1B
FD FANS 1A&1C(1B&1D) TRIPPED (EIM-2554)
ANY ID FAN RUNNING
NO FUR IN BOILER & EXT PRESS EXCURSION
LUBE OIL PRESS NORM & LOP NUNNING (T.DE-30 SEC) (EIM-2457)
MALFUNCTION FD FAN TRIP
6.9 KV BUS 1A (1B) LOAD STRIP
ID FAN 1A,1C(1B,1D) TRIP PERMISSIVE (EIM-2554)
ID FAN 1A&1C(1B&1D) SW IN NAC (EIM-2554)
LUP 1A(1B) 2 IN NAC (EIM-2554)
FD FAN 1A(1B) INLET VANES FC
FD FAN 1A(1B) OUTLET DAMPER FC
FD FAN 1A(1B) RUNNING (EIM 2456) SHT 9 (EIM 2657) SHT 11 (EIM 2554) SHT 16
FD FAN 1A A/M STATION IN AUTO (1B)
FD FAN DAMP TRIPPED
ITEM NO. 4P15 (4P16)
MOTOR
FROM INSTRUCTIONS CONSOLE
FROM MOTOR PROGRAM
FROM VALVES PROGRAM
FROM XBCNTL PROGRAM
TO EIM-2456 SHT 9
TO XBCNTL PROGRAM
EAI
FD FANS 1A(1B)


ENGINEERING
JUL 2 1979
DEPARTMENT


ENGINEERING
DEPARTMENT

VITA

Yaduraj Kapoor

Candidate for the degree of

Master of Science

Report: DESIGN AND IMPLEMENTATION OF PANEL LOGIC FOR POWER PLANT SIMULATORS

Major Field: Computing and Information Science

Biographical:

Personal Data: Born in Hyderabad, India, February 9, 1953, the son of Mr. and Mrs. Ram Prasad Kapoor.

Education: Graduated from Little Flower High School, Hyderabad, India, in 1969; received Bachelor of Engineering degree at Osmania University, Hyderabad, India, in 1975; completed requirements for Master of Science degree at Oklahoma State University, Stillwater, Oklahoma, in December, 1981.