

MICROCOMPUTER ON-LINE CONTROL OF
WIND ENERGY CONVERSION SYSTEMS

by

JOSÉ MANUEL DE ARAÚJO BAPTISTA MENDONÇA

Thesis submitted for the degree of
Doctor of Philosophy
in the Faculty of Engineering

Department of Electrical Engineering
Imperial College of Science and Technology
University of London

London, April, 1986.

ABSTRACT

This thesis deals with variable-speed Wind Energy Conversion Systems (WECS), where maximum energy extraction is aimed by allowing the wind turbine speed to fluctuate in sympathy with the wind speed.

The aim of the research is to develop a real-time microcomputer controller, as well as all the necessary interfaces, transducers and actuators, to supervise the operation of a novel wind turbine. The functions the controller is to implement include pre-start checks, start-up, normal and emergency shut-down, monitoring and on-line display of system variables and operator interaction. The controller will also implement several control policies for maximum energy extraction.

At the outset some general control aspects of WECS are described and a detailed development methodology is presented and justified.

Next, a system model is built using Structured Analysis and Design Techniques (SADT). Such a model is implemented through the parallel development of the controller software and hardware, taking full advantage of the facilities provided by a universal microprocessor development system. The performances of several microprocessors are compared by running benchmark algorithms, thus supporting the micro-computer specification.

Software/hardware integration and laboratory tests of the micro-computer-controlled WECS using in-circuit emulation were pursued before the EPROM burning of the control program. Simultaneously the real-time performance of the controller was assessed by testing it on an analogue computer model of the wind turbine.

The study of the system behaviour with the controller implementing a control strategy aiming to optimise energy extraction was also undertaken. A state-space linearised model of the WECS was analytically investigated to gain insight in to the system dynamic behaviour and stability. A z-plane analysis of the discretized system allowed parameters for the digital controller to be derived.

ACKNOWLEDGEMENTS

The author expresses his deepest gratitude to Dr. L.L. Freris, supervisor of this project, for the guidance and encouragement during the course of this research.

The author also wishes to thank Messrs. T. Moghadam, D. Boriani and especially H. Bleijs for their valuable suggestions and willingness to discuss new ideas.

The author would also like to express sincere appreciation to the colleagues in the Power Systems Section, in particular to Dr's R. Johnson and M. Shafik.

Thanks are due to the University of Porto for granting a leave of absence and to the Junta Nacional de Investigação Científica e Tecnológica, Portugal, for financial support.

The author wishes to thank Miss A. Humble for her careful typing.

I thank my parents and sister for being very supportive.

Finally, for her tireless and unlimited encouragement throughout these three and a half years, I am deeply grateful to my wife, Fátima.

TABLE OF CONTENTS

	<u>Page number</u>
ABSTRACT	1
ACKNOWLEDGEMENTS	2
LIST OF FIGURES	10
LIST OF TABLES	13
LIST OF SYMBOLS AND ABBREVIATIONS	14
 CHAPTER 1 - INTRODUCTION	 17
1.1. SMALL/MEDIUM SCALE WIND ENERGY CONVERSION SYSTEMS (WECS)	 18
1.1.1. Wind-Electric Systems	18
1.1.2. Variable-Speed Variable-Frequency WECS	18
1.1.3. Control Aspects	18
1.2. PROJECT AIMS WITHIN IMPERIAL COLLEGE WIND ENERGY SYSTEMS GROUP	 19
1.3. THESIS ORGANISATION	20
 CHAPTER 2 - CONTROL SYSTEM DEVELOPMENT METHODOLOGY	 22
2.1. SYSTEM DESCRIPTION AND CONTROLLER SPECIFICATIONS	23
2.2. A MICROCOMPUTER CONTROLLED WECS	27
2.3. SCOPE OF ACTIVITIES AND THEIR INTEGRATION	29
 CHAPTER 3 - SYSTEM MODEL BUILDING THROUGH STRUCTURED ANALYSIS AND DESIGN TECHNIQUES (SADT)	 34
3.1. INTRODUCTION	35
3.1.1. Structured Systems Design	35
3.1.2. Real-Time Computer Systems	36
3.2. PRELIMINARY ANALYSIS ACTIVITIES	37
3.2.1. Inputs to Model Building	37
3.2.2. Physical Subsystems and Task Analysis Outline	 38
3.2.2.1. Subsystem Identification and Characterisation	 38
3.2.2.2. Task Analysis Outline	40
3.3. STRUCTURED ANALYSIS	46
3.3.1. DFD's, Processes, Sources and Sinks	46
3.3.2. Top-Down Stepwise Partitioning	46
3.3.3. Structured Specification	50

3.4.	STRUCTURED DESIGN	53
3.4.1.	Transaction Analysis	54
3.4.2.	Transform Analysis	54
3.4.3.	Structured Design Specification	55
3.4.4.	Model Evaluation Criteria	61
3.4.5.	Real Time System Features	62
3.4.5.1.	Statement of the Problem	62
3.4.5.2.	Inclusion of Real-Time System Features	62
3.5.	CONCLUSIONS AND OUTLINE OF MODEL IMPLEMENTATION	63
CHAPTER 4 -	SOFTWARE DEVELOPMENT	65
4.1.	INTRODUCTION	66
4.2.	DEVELOPMENT FACILITIES	66
4.2.1.	Identification of the Development Aids Required	66
4.2.2.	The HP-64000 Logic Development System	67
4.2.2.1.	Facilities provided	67
4.2.2.2.	Software development cycle	69
4.2.2.3.	Specific features of the development system	70
4.3.	STRATEGIC DECISIONS	71
4.3.1.	Choice of Programming Language	71
4.3.2.	Postponing the Microcomputer-System Specification	72
4.4.	SOFTWARE STRUCTURE, DEVELOPMENT AND TESTING	73
4.4.1.	The Simulation Version	73
4.4.1.1.	Structure	73
4.4.1.2.	Implementation and Testing	74
4.4.2.	Benchmarks and Microprocessor Selection	75
4.4.2.1.	8-bit or 16-bit microprocessor?	75
4.4.2.2.	Memory Requirements	76
4.4.2.3.	Execution time of critical modules. Benchmarks.	76
4.4.3.	The Real-Time Control Program	78
4.4.3.1.	Implementation of Real-Time Features "kernel" design	79
4.4.3.1.1.	Interrupt-driven I/O and concurrency	79

4.4.3.1.2. Inter-Process Communication and Synchronisation	79
4.4.3.1.3. Scheduling	81
4.4.3.2. Low-Level Hardware-Dependent Software	83
4.4.4. The Control Program as an On-Line Package	83
4.4.4.1. General features of the Interactive Software	83
4.4.4.2. Performance Constraints and other Considerations	84
4.4.4.3. Running the Control Program	86
4.4.5. Implementation and Testing	89
CHAPTER 5 - HARDWARE DEVELOPMENT	90
5.1. INTRODUCTION	91
5.2. PLANT HARDWARE DETAILS	91
5.2.1. WECS Hardware	91
5.2.2. Control System Hardware	92
5.2.2.1. Types of Control Actions	92
5.2.2.2. Feedback Transducers and Control Elements	93
5.2.2.3. Closing the Control Loops	93
5.3. TRANSDUCERS AND SIGNAL CONDITIONING SPECIFICATION/DESIGN	93
5.3.1. Wind Turbine Speed	93
5.3.2. Shaft Torque and Blade Strain	94
5.3.3. Tower Vibration	94
5.3.4. Pitch Mechanism Status	96
5.3.5. Blade Pitch	96
5.3.6. Wind Speed and Wind Turbine Misalignment	98
5.3.7. P.M. Alternator Current and P.M. Alternator Voltage	99
5.4. ACTUATORS AND POWER DRIVERS SPECIFICATION/DESIGN	99
5.4.1. Blade Pitch Actuator	99
5.4.2. Electromagnetic Clutches	102
5.4.3. Fail-Safe Brake	102
5.4.4. Load Controller	102

5.5.	MICROCOMPUTER AND PERIPHERAL INTERFACING SPECIFICATION	106
5.5.1.	Microcomputer system specification	106
5.5.2.	Set-up of the Actual Microcomputer System	108
5.5.3.	Functional Description and Relevant Points	109
5.5.3.1.	CPU Board	109
5.5.3.2.	Analogue Input Board	111
5.5.3.3.	Dual Parallel Port Board	111
5.6.	MODULAR HARDWARE IMPLEMENTATION	111
CHAPTER 6 -	SYSTEM INTEGRATION AND TESTING	114
6.1.	INTRODUCTION	115
6.2.	CONTROLLER SOFTWARE/HARDWARE INTEGRATION	115
6.3.	REAL-TIME TESTS WITH AN ANALOG COMPUTER MODEL OF THE WECS	116
6.3.1.	Introduction	116
6.3.2.	Analog Computer Modelling of VSVF WECS	116
6.3.3.	Interfacing the Microcomputer Controller	117
6.3.4.	Testing Software	120
6.4.	TESTING OF HARDWARE SUBSYSTEMS	120
6.4.1.	Energy Conversion Hardware	120
6.4.1.1.	P.M. Alternator	120
6.4.1.2.	Mechanical Transmission System	122
6.4.2.	Actuation/Transducing and Power Driving/Signal Conditioning Hardware	124
6.5.	COMPLETE INTEGRATION AND LABORATORY TESTS	124
CHAPTER 7 -	SYSTEM MODELLING AND CONTROL	125
7.1.	INTRODUCTION	126
7.2.	CONTROL STRATEGIES FOR VARIABLE-SPEED WECS	126
7.2.1.	Control Loops, Controlled and Manipulated Variables	126
7.2.2.	Implementation of Speed and Torque Control Loops	127
7.2.3.	Control Strategies for Optimum Wind Energy Extraction	128
7.3.	DIGITAL CONTROLLER DESIGN	130
7.3.1.	Approaches to Digital Controller Design	130
7.3.2.	Actual Route to System Analysis and Controller Design	131

7.4.	SYSTEM MODELLING	133
7.4.1.	State-Space Model Formulation	134
7.4.2.	Transfer Function Matrix	135
7.4.3.	Linearised Model	136
7.4.4.	Simplified Model	136
	7.4.4.1. Open-Loop and Closed-Loop	136
	Multivariable Representations	
	7.4.4.2. Simplifying Assumptions	138
	7.4.4.3. SISO Approach to Dynamical	140
	Analysis	
7.5.	CONTINUOUS TIME ANALYSIS OF THE UNCONTROLLED	142
	SYSTEM	
7.5.1.	Introduction	142
7.5.2.	S-Plane Root Loci	144
7.5.3.	Stability Limits	144
7.5.4.	Transient Response	147
7.5.5.	Sensitivity to Parameter Changes	149
7.6.	SYSTEM DISCRETIZATION AND Z-PLANE ANALYSIS	153
7.6.1.	Introduction	153
7.6.2.	System Discretization	153
7.6.3.	Digital Compensator Design	156
	7.6.3.1. Choice of Compensator Structure	156
	7.6.3.2. Closed-loop system transfer	
	function	157
	7.6.3.3. Tuning of Compensator Parameters	158
	7.6.3.3.1. Pole Relocation using	158
	root-locus methods	
	7.6.3.3.2. Fine Parameter Tuning	
	with Step/Ramp	
	Response Methods	165
7.6.4.	Conclusions	169
	7.6.4.1. Optimum Compensator Parameters	169
	7.6.4.2. Degradation due to System Non-	170
	Linear Behaviour	
7.7.	IMPLEMENTATION OF THE CONTROL ALGORITHM	171
7.7.1.	Quantization Effects	171
	7.7.1.1. Quantization in Arithmetic	
	Operations	171

7.7.1.2.	Analog-to-Digital Converter	173
7.7.1.3.	Digital-to-Analog Converter	174
7.7.2.	On the Selection of the Sampling Rate	174
7.7.2.1.	Algorithm Accuracy Effect	174
7.7.2.2.	Destabilizing Effect	175
7.7.2.3.	Information Loss Effect	175
7.7.2.4.	System Outputs Not Completely Specified by the Output Z Transform	176
7.7.2.5.	Word Length Effect	177
7.7.3.	Realization of the Control Algorithm	177
7.7.3.1.	Compensator Action	177
7.7.3.2.	Implementation of the Compensator	178
7.7.3.3.	Actuation Interface	178
7.7.4.	On-Line Parameter Tuning	179
CHAPTER 8 -	CONCLUSIONS	180
8.1.	CONCLUSIONS	181
8.1.1.	General	181
8.1.2.	Development	181
8.1.3.	Software	181
8.1.4.	Hardware	182
8.1.5.	Testing	183
8.1.6.	Modelling and Control	183
8.2.	ORIGINAL CONTRIBUTIONS	184
8.3.	SUGGESTIONS FOR FUTURE WORK	186
APPENDIX 4		187
4.1.	GENERAL FEATURES OF THE INTERACTIVE SOFTWARE	187
4.2.	INTERACTIVE PACKAGE BUILDING BLOCKS	188
APPENDIX 5	CONTROLLABLE LOAD DESIGN AND IMPLEMENTATION	189
APPENDIX 6	MICROCOMPUTER HARDWARE SET-UP AND LABORATORY TESTS	198
APPENDIX 7		210
7.1.	SIMULATION SOFTWARE	210
7.2.	WECS MODELLING AND PARAMETER IDENTIFICATION	210
7.3.	SYSTEM TRANSFER FUNCTION MATRIX	215

7.4.	LINEARISED STATE-SPACE MODEL	217
7.5.	STATIC COUPLING FACTOR	219
7.6.	TRANSIENT RESPONSE OF THE UNCONTROLLED WECS	220
7.7.	DISCRETIZATION OF THE CONTINUOUS PLANT	223
7.8.	Z-PLANE CLOSED-LOOP TRANSFER FUNCTION	225
7.9.	SYSTEM PERFORMANCE SPECIFICATIONS	226
7.10.	RAMP RESPONSE BY THE LONG DIVISION METHOD	226
REFERENCES		228

LIST OF FIGURES

- Fig. 2.1a) WECS general view
- Fig. 2.1b) WECS basic mechanical components
- Fig. 2.2. Blade pitch control mechanism
- Fig. 2.3. The microcomputer controlled WECS. Basic approach to the control problem
- Fig. 2.4. Diagram of WECS controller development activities
- Fig. 2.5. Basic stages of controller development and testing
-
- Fig. 3.1. WECS Simplified Block Diagram
- Fig. 3.2. WECS State Transition Diagram
- Fig. 3.3. SCHEDULER procedural description
- Fig. 3.4. START task specification
- Fig. 3.5. STOP task specification
- Fig. 3.6. EMERGENCY STOP task specification
- Fig. 3.7. Level-1 DFD - "Start" Operating Mode
- Fig. 3.8. Level-1 DFD - "Stop" Operating Mode
- Fig. 3.9. Level-1 DFD - "Emergency Stop" Operating Mode
- Fig. 3.10. Level-1 DFD - "Optimal Operation" Operating Mode
- Fig. 3.11. Level-1 DFD - "Non-optimal Operation" Operating Mode
- Fig. 3.12. Level-2 DFD - "Optimal Operation/Non-optimal Operation"
- Fig. 3.13. Level-0 SC
- Fig. 3.14. Level-1 SC - Threshold Check
- Fig. 3.15. Level-1 SC - Start
- Fig. 3.16. Level-1 SC - Stop
- Fig. 3.17. Level-1 SC - Emergency Stop
- Fig. 3.18. Level-1 SC - Optimum Control
- Fig. 3.19. Level-2 SC - Evaluation of the wind speed "seen" by the rotor
- Fig. 3.20. Graphic Symbols
-
- Fig. 4.1. The HP-64000 Logic Development System
- Fig. 4.2. Real-Time Software Structure
- Fig. 4.3. Low-level Software Modules
- Fig. 4.4. Interactive Package Building Blocks
- Fig. 4.5. Information page
- Fig. 4.6. System initialisation (Stand-by State)
- Fig. 4.7. Variable Display Mode (Optimumcontrol)
- Fig. 4.8. Variable Plot Mode (Emergency Stop)

- Fig. 5.1. Wind Turbine Speed Measurement
- Fig. 5.2. Hardware Switch Debouncing
- Fig. 5.3. Blade Pitch Measurement Hardware
- Fig. 5.4. Current Measurement Circuit
- Fig. 5.5. Blade Pitch Actuation Linkages
- Fig. 5.6. Actuator Power Driver
- Fig. 5.7. Clutches Power Driver
- Fig. 5.8. Brake Power Driver
- Fig. 5.9. Load Controller
- Fig. 5.10. Microcomputer System Module
- Fig. 5.11. Interface Circuitry Module
- Fig. 5.12. System Block Diagram

- Fig. 6.1. WECS analog computer model used for controller real-time tests
- Fig. 6.2. WECS analog computer network interfaced to the microcomputer controller
- Fig. 6.3. DFD representation of the testing software
- Fig. 6.4. On-line display of variable values and WECS operating point
- Fig. 6.5. Mark II p.m. generator test results
- Fig. 7.1. Lambda optimum control strategy. Speed feedback scheme.
- Fig. 7.2. WECS simplified model as a 4th order mechanical rotational system
- Fig. 7.3. Open-loop 2-input 2-output system representation
- Fig. 7.4. Complete closed-loop multivariable representation
- Fig. 7.5. Simplified 2-input 1-output closed-loop model
- Fig. 7.6. SISO model - inherent turbine feedback loop
- Fig. 7.7. SISO model - controller loop
- Fig. 7.8. S-plane root locus for the open-loop WECS - "mechanicals"
- Fig. 7.9. S-plane root locus of the open-loop WECS
- Fig. 7.10. Partial plot of the WECS $T(\omega, V)$ family of curves
- Fig. 7.11. WECS eigenvalue migration with parameter variation - "mechanical poles"
- Fig. 7.12. WECS eigenvalue migration with parameter variation - "electrical poles"
- Fig. 7.13. Controlled WECS as a sample-data system
- Fig. 7.14. Transfer function of a digital PID

- Fig. 7.15. Z-plane eigenvalue migration due to changing sampling rate
- Fig. 7.16 Z-plane root loci of the closed-loop system
- Fig. 7.17. Z-plane closed-loop pole location for (λ_{opt} , 700 rpm)
- Fig. 7.18. Step responses corresponding to the pole patterns in Fig. 7.17.
- Fig. 7.19. Step responses with "best" tuning
- Fig. 7.20. Response to a ramp for the tuning values in Fig. 7.19.
- Fig. 7.21. System degradation due to non-linear behaviour
-
- Fig. A5.1. Load controller action
- Fig. A5.2. Wind turbine torque and power coefficient curves
- Fig. A5.3. Stepped approximation to the quadratic load torque curve
- Fig. A6.1. Linker Load Map
- Fig. A6.2 System Memory Map
- Fig. A6.3 Interrupt Priority Allocation
- Fig. A6.4. PIA configuration
- Fig. A6.5. Blade Pitch Control Loop
- Fig. A6.6. Blade Pitch Calibration Curve
- Fig. A6.7. Field Hardware Layout
- Fig. A7.1. Structure of the simulation software
- Fig. A7.2 WECS modelling

LIST OF TABLES

Table 4.1.	-	Map of the control program memory requirements
Table 4.2.	-	Execution time of base-level module "Optimumcontrol"
Table 4.3.	-	Evaluated execution time for one complete control cycle in the Optimal or Non-Optimal Operation State
Table A5.1.	-	Typical WECS operating points for λ_{opt} operation ($\lambda_{opt} = 0.93 \Rightarrow C_p = 0.36$)
Table A5.2.	-	Selected generator operating points (from test results in 5.1.).
Table A5.3.	-	Partitioning of the 10 kW load as a function of the number of digital control channels
Table A5.4.	-	Design table for the 6-channels, 10 kW switchable load
Table A5.5.	-	Power dissipated by the load at several WECS operating points
Table A6.1.	-	Execution time of the main program blocks executed each control cycle
Table A7.2.	-	WECS model order reduction
Table A7.5.	-	Coupling gains and static coupling factor for the WECS multivariable model
Table A7.6.	-	Characterising parameters for a second-order approximation of the uncontrolled system transient response
Table A7.7.	-	Partial fraction expansion constants

LIST OF SYMBOLS AND ABBREVIATIONS

In general, symbols and abbreviations are defined the first time they appear in the text. This list is complemented with Table 3.1 where the symbols specific to the system conceptual model built through SADT's are presented.

PRINCIPAL SYMBOLS

$[A]$	System matrix
$[A^*]$	Jacobian matrix derived from $[A]$
$[B]$	Distribution matrix
$[B^*]$	Jacobian matrix derived from $[B]$
B_i	Damping
$[C]$	Output matrix
C_p	Wind turbine power coefficient
C_q	Wind turbine torque coefficient
D_{ij}, D	Compensator transfer function
e	Error signal to compensator
$[G]$	System matrix or plant matrix
G_p	Plant transfer function
G_{ho}	Sample-and-hold transfer function
G_{ij}	Scalar transfer functions
J	Inertia
K	Wind turbine feedback gain
K_{wt}	Inherent system gain
K_o	Static Coupling Factor
K_{ij}	Coupling gains
K_p, K_i, K_d	Compensator gains
k	Time step
n	Rotational speed
P	Wind Turbine Power
R	Electrical Resistance
S	Stiffness
T_i	Torque
t	Time
T	Sampling time
$\underline{u}$	Input or control vector
u	Compensator output to Actuator

v	Wind speed
$\underline{x}$	State vector
$\underline{y}$	Output vector
$\alpha, \beta, \gamma, \delta,$ $\alpha', \beta', \gamma', \delta', \zeta'$	S-plane or z-plane poles or zeros
ζ	Damping factor
λ	Tip speed ratio
θ	Mechanical angle
ω	Angular speed
ω_n	Natural frequency

Subscripts

d	Delay (time)
e, el	Electrical
fc	Flexible coupling
g	Generator
lim, max	"Limit" value
op, opt	Optimum value
r	Rising (time)
s	Settling (time)
t	Wind Turbine
w	Wind

ABBREVIATIONS

ACIA	Asynchronous Communication Interface Adapter
ADC	Analog-to-digital converter
DAC	Digital-to-analog converter
DDC	Direct Digital Control
DFD	Data Flow Diagram
f.v.	Final value
FSFF	Fixed speed fixed frequency
HP-64000	Hewlett Packard Development System
ICE	In-circuit emulation
LDS	Logic Development System
MDS	Microprocessor Development System
o.s.	Overshoot
PIA	Peripheral Interface Adapter
PTM	Programmable Timer Module
SA	Structured Analysis
SADT	Structured Analysis and Design Techniques
SC	Structured Chart
SD	Structured Design
SISO	Single-input single-output
SNR	Signal-to-noise ratio
VSVF	Variable speed variable frequency
WECS	Wind Energy Conversion System

CHAPTER 1

INTRODUCTION

1.1. SMALL/MEDIUM SCALE WIND ENERGY CONVERSION SYSTEMS (WECS)

1.1.1. Wind-Electric Systems

Wind-electric generating schemes rely on an energy conversion interface which consists of a generator and, usually, auxiliary conditioning circuitry [1.1.,1.2.]. Many options arise from the different generator types and the various techniques available of controlling electrical machines. The nature of the interface depends on the application, i.e. final use of the electrical energy, and determines the WECS operating mode.

Fixed-speed fixed-frequency (FSFF) is the usual mode of operation of large grid-connected systems. However other modes are possible and are frequently used in small and medium WECS. Although most of the work presented in this thesis applies to WECS in general, it is specifically oriented towards the small/medium scale variable speed variable-frequency (VSVF) WECS. For our purposes we could draw arbitrary boundaries between small/medium and medium/large WECS at 30 kW and 300 kW levels respectively.

1.1.2. Variable-Speed Variable-Frequency WECS

The large number of applications of small/medium WECS [1.1.-1.10.] have the common feature of simplicity and are particularly suited to autonomous and/or decentralised systems. Moreover, VSVF systems aim to operate at fixed tip-speed ratio thus enhancing wind energy extraction and relieving the transmission system from mechanical stresses [1.2.,1.3.].

1.1.3. Control Aspects

Considerable research has been pursued in recent years on small/medium scale WECS and many commercial systems are now available. Much of the latest research has been directed towards maximisation of energy extraction and minimisation of mechanical stresses.

Optimum performance of the energy conversion hardware requires the provision of adequate control loops. These are mostly determined by the electrical interface and the operating mode selected.

1.2. PROJECT AIMS WITHIN IMPERIAL COLLEGE WIND ENERGY SYSTEMS GROUP

A research programme in wind energy applications was initiated in 1976 at Imperial College. Its main goals were the development of small WECS for space and water heating in domestic and suburban buildings, farm applications and energy supply to remote areas. Several projects were implemented which tackled different aspects of the problem.

A number of horizontal axis wind turbines were constructed and instrumented and control equipment designed and built [1.5.,1.8.]. A family of high efficiency p.m. alternators was also designed and built [1.9.]. The performance of different WECS was assessed through field tests.

Dynamic analysis of WECS and comparison of different configurations and control schemes were done through analog computer simulations [1.3.]. At present the technical and economical feasibility of wind-diesel systems are under study [1.10.].

In parallel project a small but versatile WECS is being developed. Testing of different system configurations and implementation of several control strategies are the long term goals of this project. Initially a VSVF WECS for space and water heating applications is being developed. At a later stage a VSFF system using an induction generator will be implemented taking advantage of most of the existing test bed hardware.

The present test bed configuration consists of a high-speed 2-bladed wind turbine driving a p.m. generator which feeds a controllable resistive load. The 3.5 m diameter rotor is directly coupled to the generator. The turbine has a number of novel design features and the WECS is thoroughly instrumented both for control and data logging purposes.

The ultimate aims of this first research phase are :

- to test the wind turbine performance in real wind conditions and analyse the importance of its novel features,
- to run the WECS according to different control strategies aiming to optimise wind energy extraction and assess system performance in each of the cases.

The work described in this thesis covers the design, implementation and testing of a microcomputer-based on-line controller for the present WECS configuration, as well as all the interfacing, sensing and actuation circuitry. It also deals with the issues of WECS modelling and dynamic analysis and digital compensator design and tuning.

1.3. THESIS ORGANISATION

In Chapter 2 the controller specifications are laid out and the choice of a dedicated microcomputer system outlined. The development methodology adopted is then presented and justified. The development stages, design activities and existing links are described exposing the iterative nature of the development process.

Controller conceptual design is the object of Chapter 3. A system model is built using Structured Analysis and Design Techniques and graphic tools are shown to be an effective system representation. The transition to the next phase is supported by incorporating real-time system features and an outline of model implementation.

Software development is introduced in Chapter 4 by covering the issues of development facilities, programming languages and software structure. Benchmark algorithms are then derived to support a performance comparison study leading to the specification of the controller microcomputer. Real-time software features are included in the form of a small dedicated "kernel". Finally, the implementation of the interactive features of the control program is covered.

Chapter 5 begins with a description of plant and controller hardware details. Next, all the hardware development is covered in three sections : transducers and signal conditioning, actuators and power drivers and microcomputer and peripheral interfacing. Finally modular hardware implementation is dealt with.

Hardware/software integration is discussed in Chapter 6, where the different stages of controller testing are presented. First, simulation software allowed real-time tests with an analog computer model of the WECS. Laboratory tests checked the behaviour of all the links between the actual control program and the WECS hardware. Finally, the control program was run while simulating the actual field environment.

Modelling and control of the WECS is the object of Chapter 7. Control strategies for VSVF WECS are introduced and the approach to compensator design justified. System parameters are determined and a state-space linearised model of the WECS is built. This model of the multivariable system is then simplified through a loop decomposition approach. A time-domain s-plane analysis of the open-loop system is then presented. Stability limits and sensitivity to parameter changes are investigated. The z-plane analysis of the closed-loop sample-data system supporting the design and tuning of a digital compensator is described. Discussion of implementation related features closes this chapter.

Finally in Chapter 8 conclusions are drawn and suggestions for further work are given.

CHAPTER 2

CONTROL SYSTEM

DEVELOPMENT METHODOLOGY

2.1 SYSTEM DESCRIPTION AND CONTROLLER SPECIFICATIONS

From this point onwards only the first of the system configurations described in section 1.2. will be considered. The earlier summarised description will now be completed in order to set the scene for the controller specification. The 3.5 m diameter rotor of Fig. 2.1 has wooden blades instrumented with strain-gauges at the root and at half radius. A slip-ring assembly "e" allows their signals to be transferred to the non-rotating part.

The blade pitch can be controlled by an electrical actuator "n" through a group of mechanical links (Fig. 2.2). The actuator is held in position by a pair of electromagnetic clutches "o" which collapse in the case of mains power failure or controller failure. In such an occurrence a pair of springs forces the blades to coarse pitch even if the actuator is faulty (Fig. 2.1a). Mechanical torque is transmitted to the 12-pole p.m. generator "h" through a torque transducer "f". Finally a fail-safe brake "i" provides a last line of defence.

Since the controller has to attain a number of goals (1.2.) it will have to implement the following basic number of functions:

- pre-start checking, start-up, normal and emergency stop procedures;
- run the WECS according to one of several control algorithms for maximum energy extraction;
- monitor system variable values and take appropriate action if limit values are exceeded;
- accumulate these on-line control tasks with operator interaction and real-time display functions.

These functional specifications match reasonably well the features of controllers used in much bigger WECS [2.1.-2.8.]. This is primarily due to the fact that the WECS under study is, in spite of the small blade diameter and modest power rating of this first configuration, a research oriented highly instrumented machine.

In parallel with the controller development all the

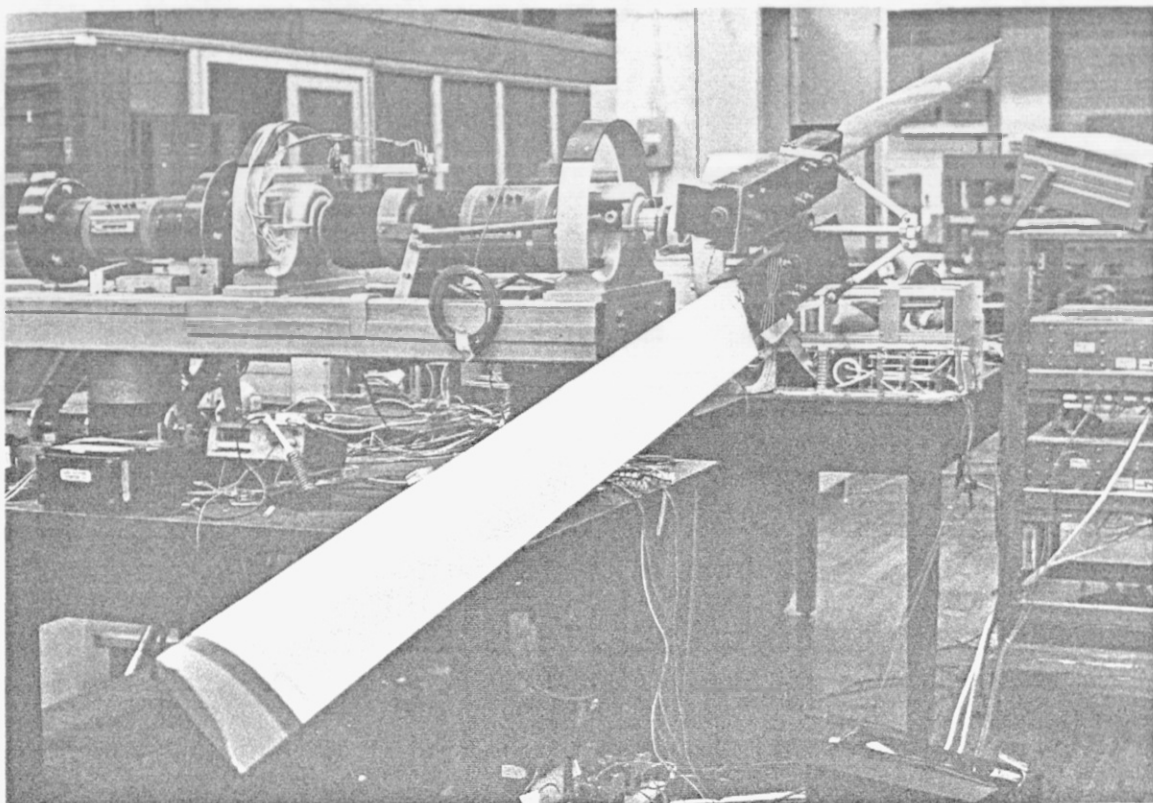


Fig. 2.1a) WECS general view.

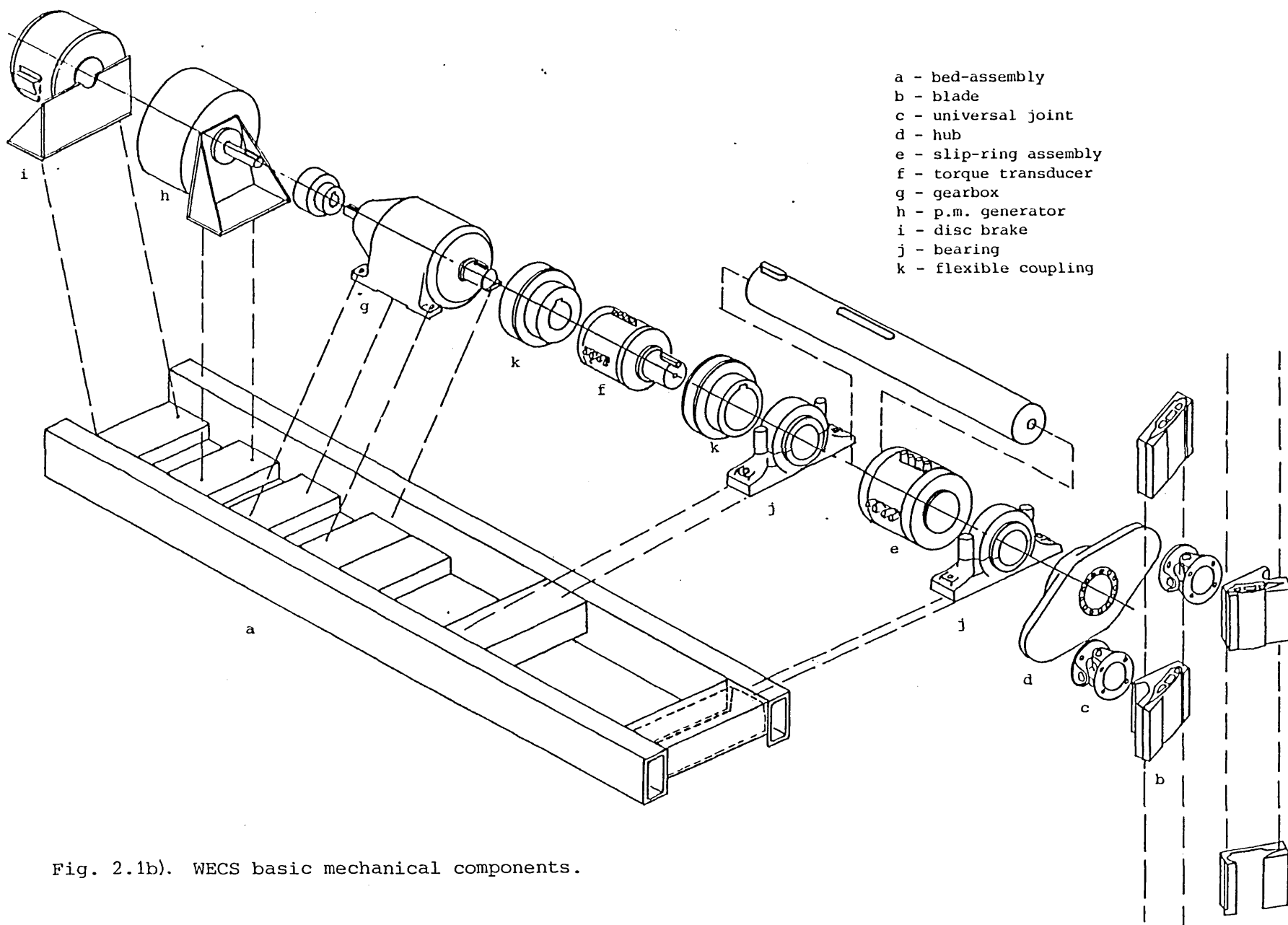


Fig. 2.1b). WECS basic mechanical components.

- l - counter-weight side of blade
- m - steel rod
- n - blade pitch actuator
- o - electromagnetic clutches

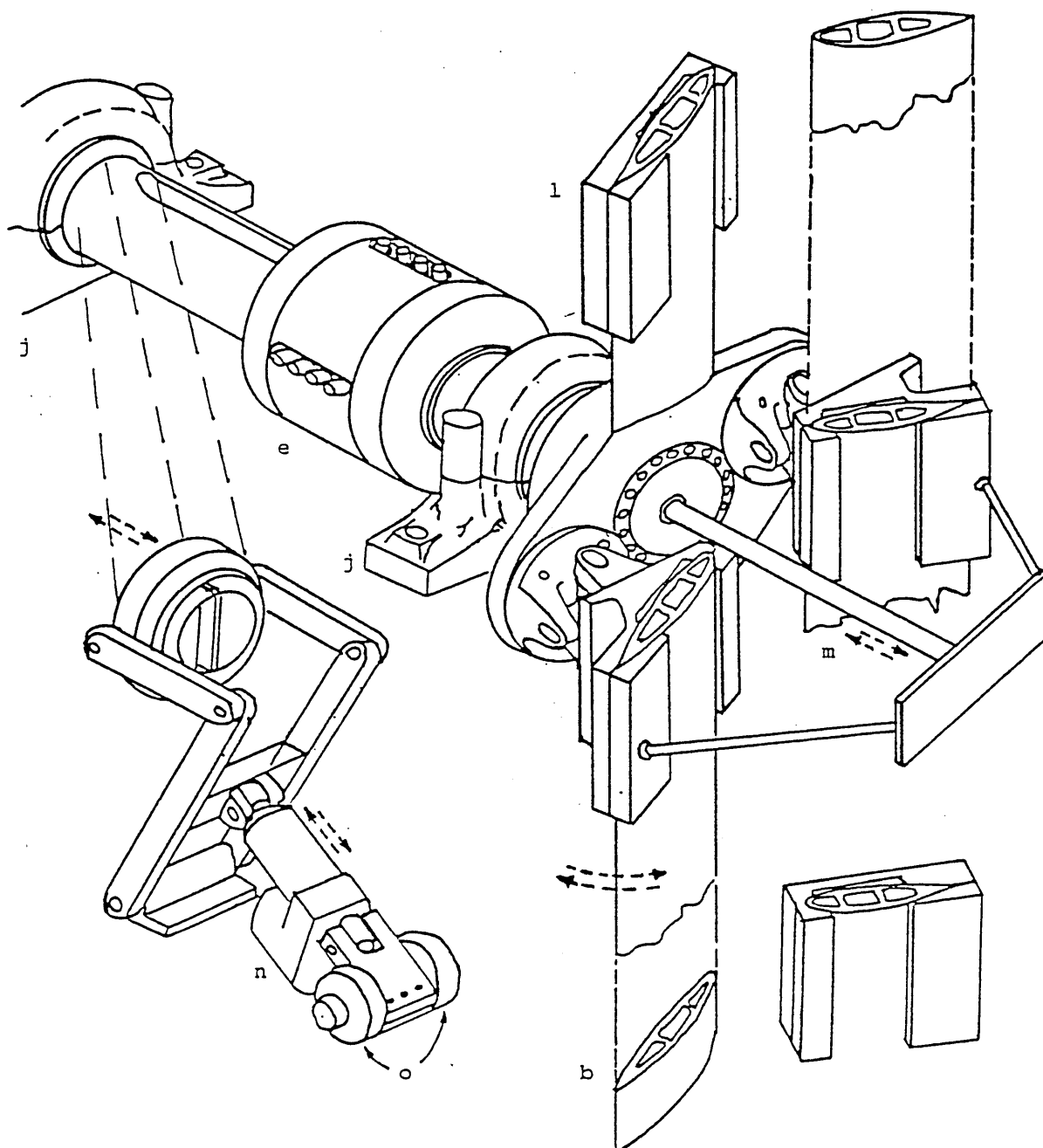


Fig. 2.2. - Blade pitch control mechanism.

necessary transducers and actuators and respective interfacing circuitry will have to be specified or designed.

A final obvious design constraint is to keep the cost of the controller hardware within a reasonable percentage of the total system cost.

2.2. A MICROCOMPUTER CONTROLLED WECS

A microcomputer-based controller appears as the obvious choice for several reasons:

- the implementation of all the above specified controller functions would otherwise require designing, building and testing extensive dedicated electronic circuitry. With such an hardwired controller even minor changes would require a certain amount of system redesign.
- small but powerful microcomputers are nowadays available offering a good and inexpensive alternative to classical analog/digital controllers.
- versatility and expandability are enhanced and future upgrading is feasible if the software is made portable.

In Fig. 2.3 the WECS is the plant or controlled process. The controller includes microcomputer hardware, software and interfacing. The actuators and transducers close the loop. The design of a controller for a WECS is therefore no different from the design of controllers for industrial processes.

Possible alternatives for the WECS controller are [2.1.-2.8.]:

- a control system consisting of one or more industrial controllers bought off-the-shelf, e.g. programmable logic controllers (PLC), analog controllers or hybrid controllers.

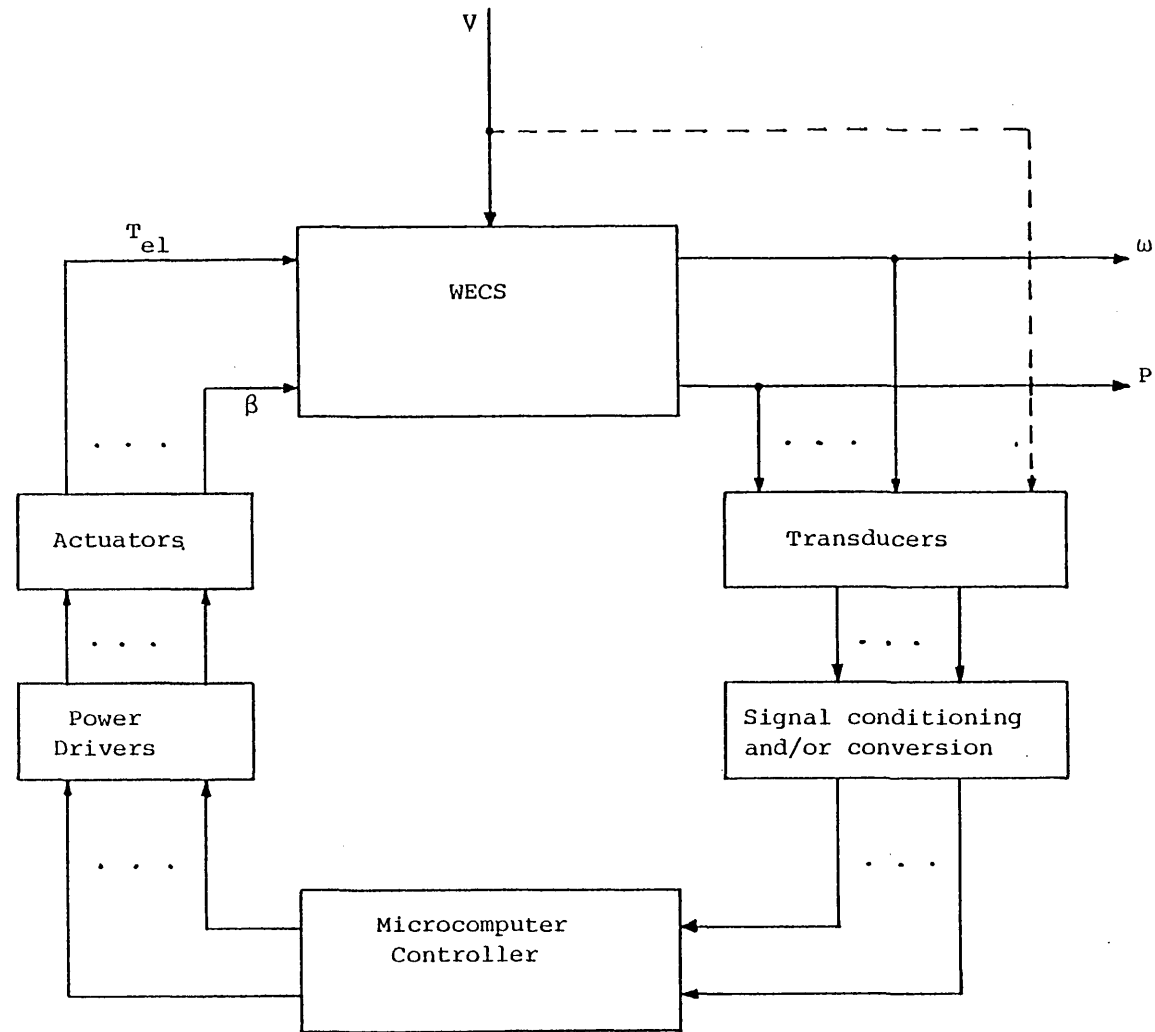


Fig. 2.3. - The microcomputer-controlled WECS.
Basic approach to the control problem.

- a powerful minicomputer doing both control and data acquisition and allowing operator interaction and on-line display.
- a purpose built microcomputer-based controller developed on the basis of the actual system specifications. This could range from a simple and relatively inexpensive microcomputer to a sophisticated multiprocessor system.

The design of a customised microcomputer-based controller based on a state-of-the-art microprocessor appeared to be the best option. This approach will provide most of the facilities of a much more expensive system by tailoring the controller to satisfy the particular requirements. It will also ensure a more suited final product by allowing the controller development to progress in parallel and interactively with the WECS development up to the final integration stage.

A first outline of the microcomputer specification will include requirements for:

- fast execution of floating point arithmetic for on-line control purposes,
- powerful interrupt structure enabling fast response to external events and optimum CPU use,
- wide and versatile interfacing to sensors and actuators.

The development of the necessary transducers, actuators and interfaces will partly depend on the actual controller chosen.

2.3. SCOPE OF ACTIVITIES AND THEIR INTEGRATION

The development of a controller which meets the required specifications is a complex task. Some difficulties can be foreseen, namely in:

- allowing the design of each subsystem to interact with the design of several others,

- conducting extensive integration and testing work in order to ensure a degree of reliability and performance that a system in the field demands.

A disciplined and versatile development methodology was therefore followed. Fig. 2.4 gives a pictorial description of the proposed methodology showing the main design activities and how they are organised. The so-called seven ages of a computer system [3.1.] are shortened down to three. In fact the "problem recognition" and "feasibility study" stages were done previously. Since analysis and design were tied together, only three basic stages of development remain:

- conceptual design,
- implementation,
- system testing.

These stages are not isolated blocks executed sequentially. On the contrary, the development activities of different stages interact strongly and, within the same stage, they often run in parallel. Information from advanced design phases can be fed back into earlier activities in an iterative process.

Referring to Fig. 2.4, a brief description of those activities now follows. The most relevant interaction links are represented by the dotted lines. Conceptual design is divided in three phases and consists of building a system model using Structured Analysis and Design Techniques (SADT's). In the first phase, Preliminary Analysis, the information needed in the second and third phases, Structured Analysis and Structured Design, is collected and checked. In the implementation stage hardware and software development evolve in parallel and with a very high degree of interaction. Testing comprises hardware/software integration and a first laboratory test of the controller performance. Once a stage is reached where all the hardware has been put together and successfully tested with the debugged software, the control program is burned into EPROM thereby completing the controller development.

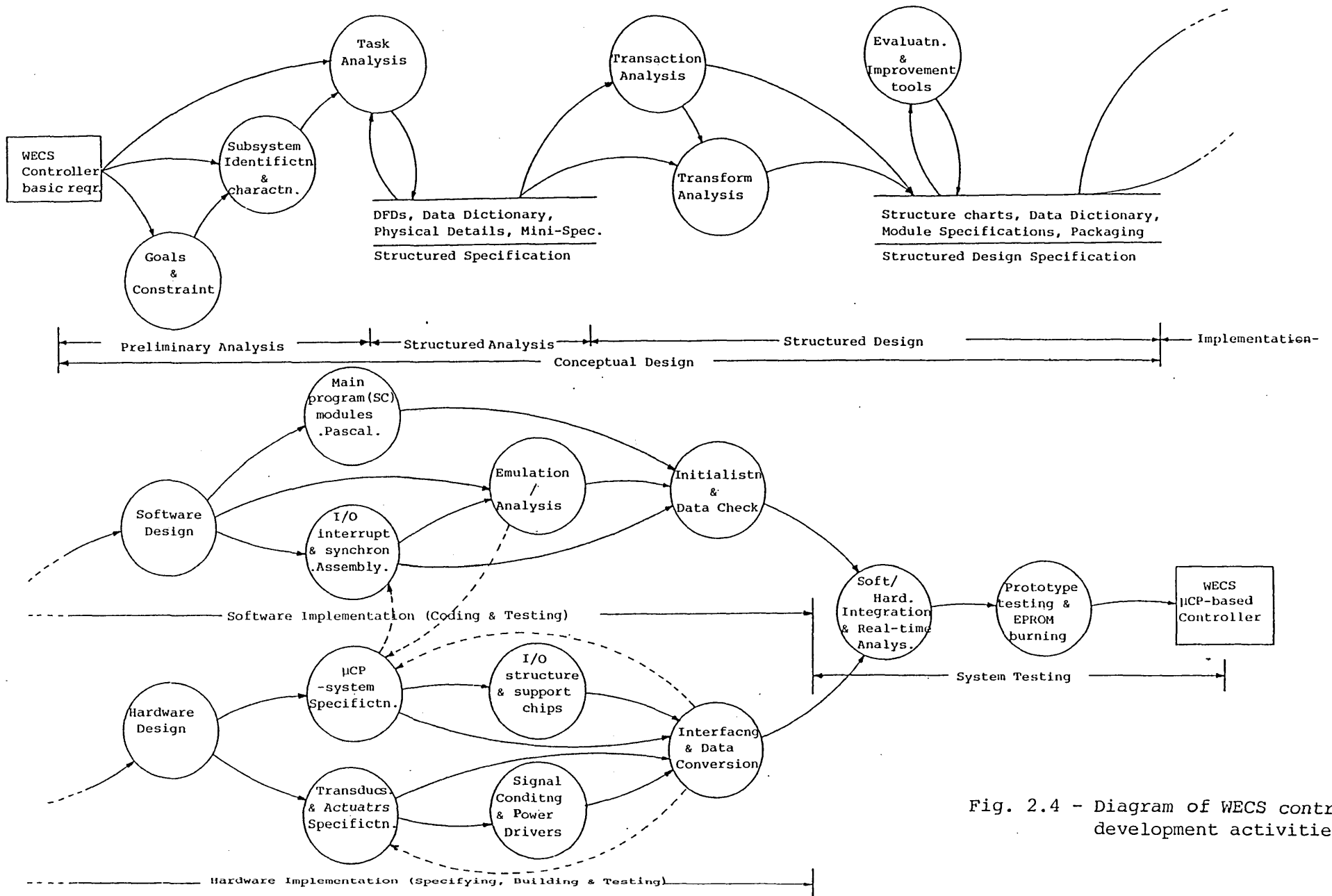


Fig. 2.4 - Diagram of WECS controller development activities

Fig. 2.5 shows how the controller development activities are integrated in the more general scope of the whole project. After the WECS prototype is finished, complete testing has still to be done outside the limited and protected laboratory environment. The information gathered during field performance tests may lead to changes either at the implementation or even back at the specification stage.

The design of the fundamental direct digital control (DDC) loop is a complex task requiring special attention. Modelling of the physical WECS, dynamic analysis and digital compensator design are therefore a separate branch in the "network" of Fig. 2.5.

Finally it should be stressed that the development methodology described has the advantage of being application independent.

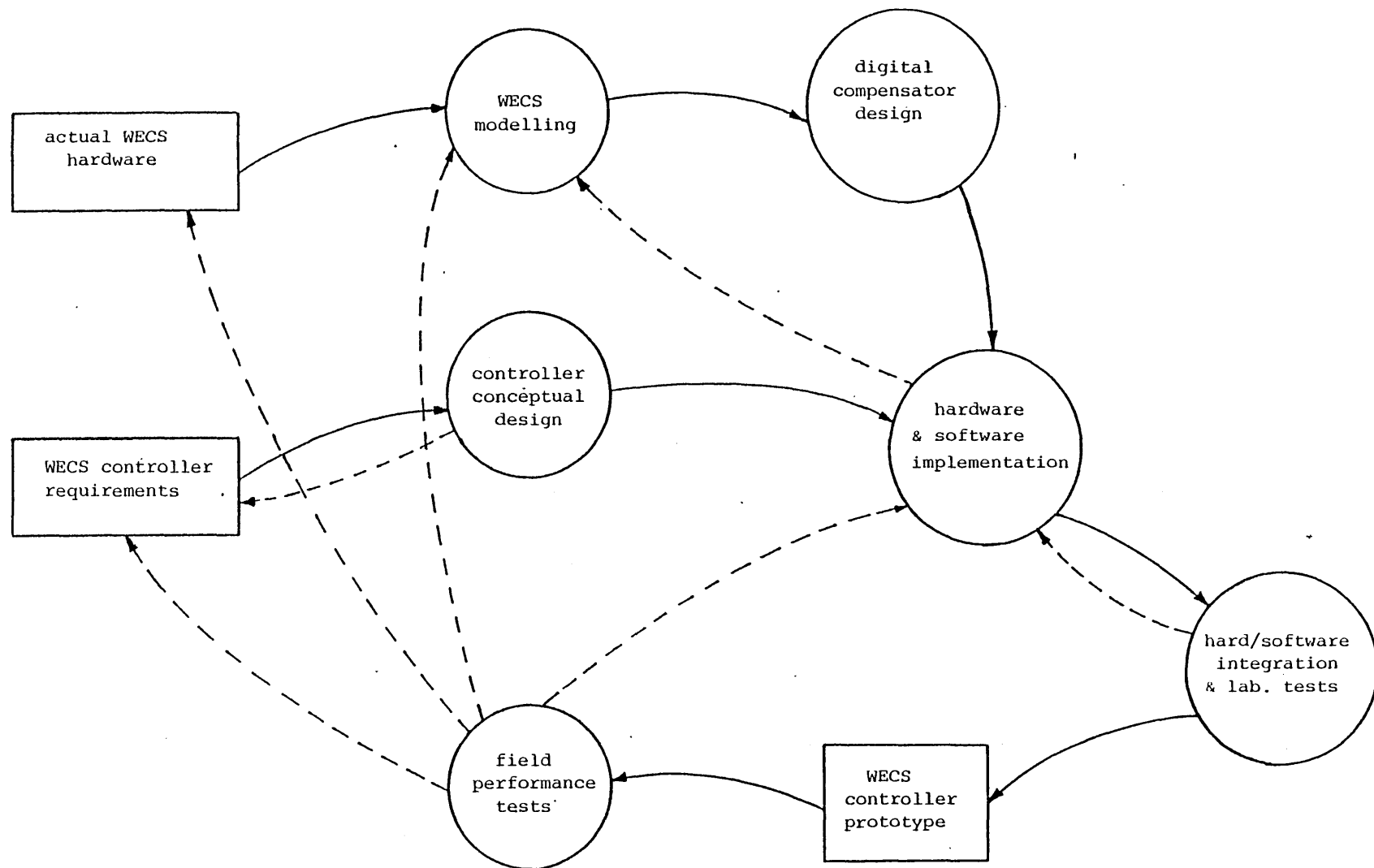


Fig. 2.5 - Basic stages of controller development and testing

CHAPTER 3

SYSTEM MODEL BUILDING THROUGH STRUCTURED ANALYSIS AND DESIGN TECHNIQUES

3.1. INTRODUCTION

The system model dealt with here is a functional model representing the system as it is seen from the computer system implementation point of view. In Chapter 7 a mathematical model of the physical system is derived with the purpose of studying its dynamical response.

The functional model was built using Structured Analysis and Design Techniques (SADT's) in a three stage development process.

Deriving the actual inputs to the model building and doing a preliminary task analysis are activities belonging to the analysis stage. These early analysis activities, preceeding Structured Analysis, are the first stage of model building (Fig. 2.4). In the other two stages Structured Analysis and Structured Design tools are applied to obtain the system model. In the last part of the Structured Design stage the model is completed with the specification of the mechanisms which will make up a real-time system.

3.1.1. Structured Systems Design

The main field for application of SADT's is systems analysis and software design for big and/or complex systems, as in business, banking, scientific and military applications [3.1.]. SADT's are, however, analysis and design tools of universal application and they should be used whenever the system to be designed requires complex software, e.g. operating systems, process-control systems, etc. This is the case with on-line control systems of some complexity and the WECS controller is no exception.

Structured Design (SD) aims to a high-level structure of systems/programs producing maintainable, reliable and adaptable systems. Its five main aspects are:

- "form follows function" or "what => how" approach to the design,
- partitioning and hierarchy to conquer complexity,
- graphic tools to render systems more understandable,

- design strategies to develop a design solution from a well-defined statement of a problem,
- quality evaluation criteria.

SD is not structured programming, although the software implementation of the model built through SD will use structured programming. SD is not top-down design or modular programming, but an extension of these with a much more powerful scope.

The use of Structured Analysis tools in the System Analysis stage is of the greatest importance. Structured Analysis replaces the old-style functional specification with a Structured Specification which will provide the correct input to the Structured Design stage.

3.1.2. Real-time Computer Systems

The WECS on-line controller has most of the features of a real-time control system. The controller model has therefore to be provided with features guiding its implementation as a real-time system.

Although the design of real-time software is a very complex task, there are techniques that help a conceptual design to be mapped onto the physical hardware [3.2.-3.8.]. Real-time control software has many of the functional requirements of a real-time operating system for a computer [3.3.], as well as unique software requirements [3.4.,3.7.]. It is therefore understandable that such techniques aim to build a scaled-down versatile version of an operating system. This will provide a software layer between the system model, or its high-level language implementation, and the actual hardware. This structure has to perform three basic functions - scheduling, foreground processing and background processing. Although this can be done at different levels of sophistication and efficiency, in every case the implemented "virtual machine" creates an environment where pseudo-concurrent processing is possible.

The actual implementation of this part of the model is obviously hardware dependent, and therefore kept to a later stage,

(Chapters 4, 5 and 6). Its conceptual design, however, is done during the SD stage.

3.2. PRELIMINARY ANALYSIS ACTIVITIES

3.2.1. Inputs to Model Building

These will consist on the controller requirements duly complemented with physical constraints and design goals. The controller requirements have already been described in 2.1.

The existing constraints are due to physical limitations and functional considerations. Physical limitations arise from the actual wind turbine hardware and from the necessary sensors and actuators. They will have to be considered from an early development stage although detailed information will only be relevant much later (Chapter 5). Functional considerations concern the nature of the WECS operation and require careful analysis of the control functions and of the consequences of their implementation on system performance and safety.

The major design goals, versatility and expandability, arise from the following points:

- different control policies must be implemented with minimum hardware and software changes,
- the actual field testing will probably advise changes and/or additions to be made,
- total or partial system upgrading may be wished in the future and, if this occurs, it will be a must to keep most of the design and implementation work.

Requirements, goals and constraints will always be present throughout the development stage, from the early phases of model building to the testing stage.

3.2.2. Physical Subsystems and Task Analysis Outline

3.2.2.1 Subsystem identification and characterisation

From the set of requirements, goals and constraints a first draft of the physical subsystems is derived. The WECS simplified block diagram (Fig. 3.1) presents the identified subsystems and their basic relationships. These are divided into groups in accordance with the type of task they accomplish:

(i) "Process" subsystems

(i.e. the ones building up the WECS to be controlled)

- wind turbine
- p.m. alternator
- passive load

(ii) Sensing subsystems

(inc. transducing, signal conditioning, isolation and rms/dc conversion).

- wind turbine speed
- shaft torque
- blade strain
- blade pitch
- pitch mechanism status
- tower vibration
- wind speed
- wind turbine misalignment
- load power
- p.m. alternator current
- p.m. alternator voltage
- supply voltage

(iii) Actuation subsystems

(inc. buffering, power drivers and actuators)

- blade pitch actuator
- electromagnetic clutches
- fail-safe brake
- load controller

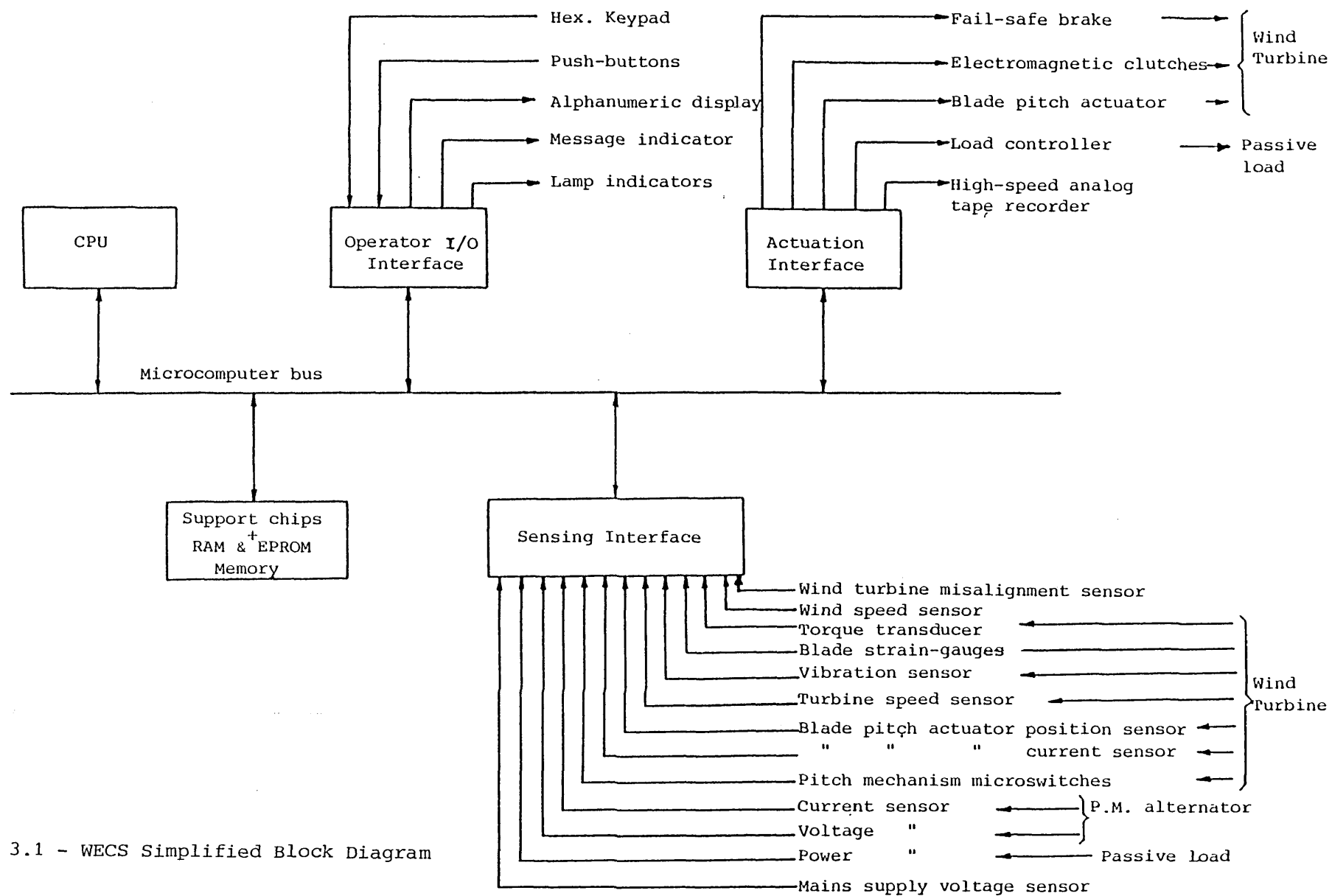


Fig. 3.1 - WECS Simplified Block Diagram

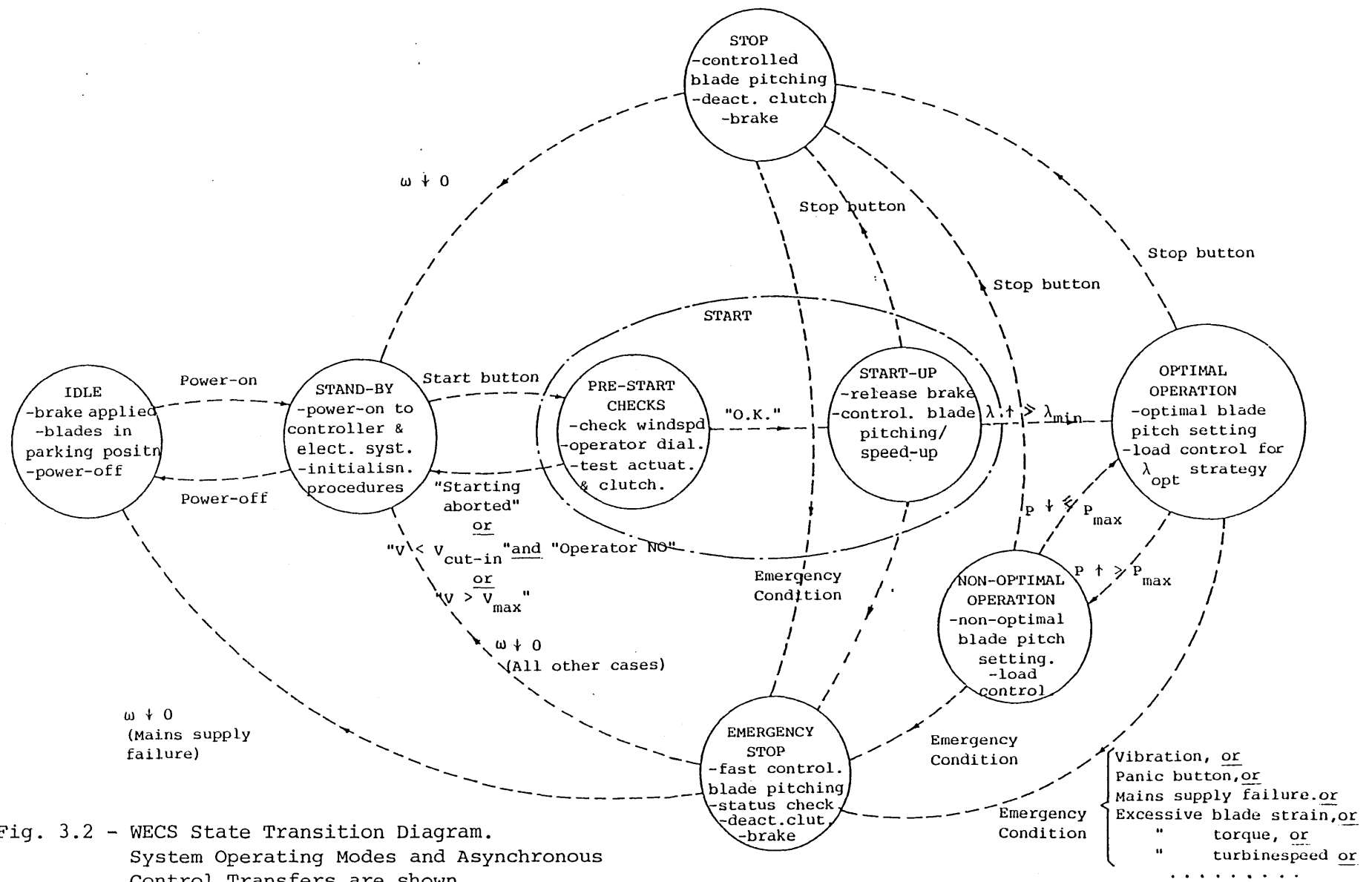
- (iv) Operator interaction subsystems
- (v) Microcomputer controller
 - (i.e., CPU, RAM, EPROM and IC support chips like digital I/O and serial I/O, timer/counter and priority interrupt controller chips).
- (iv) Interfacing subsystems

The characterisation of these subsystems is done by adding physical and functional details. At this point in the development stage some subsystems were known with a fair amount of detail, like the wind turbine, the p.m. alternator, the mechanical actuators and the torque transducer. However most of them could only be identified or briefly specified since their design still had to be done. The detailed characterisation of the subsystems is therefore postponed to Chapter 5.

3.2.2.2. Task analysis outline

All the information achieved so far is used to build a first simplified model representing the system functions and operating modes. This model is then gradually modified until it gives an acceptable system representation. This is done by using a State Transition Diagram (Fig. 3.2), where the basic controller tasks are easily identified with the different system operating modes.

The need of a supervisor or executive task ruling all transitions between operating modes is apparent. The algorithms of each of the basic tasks and of the supervisor task are outlined in Figs. 3.3 to 3.6. Flowcharts are used here to describe the internal mechanics of each task since they had been used in the initial specification of those basic tasks. This representation expresses a neat controller specification which will support the use of Structured Analysis tools.



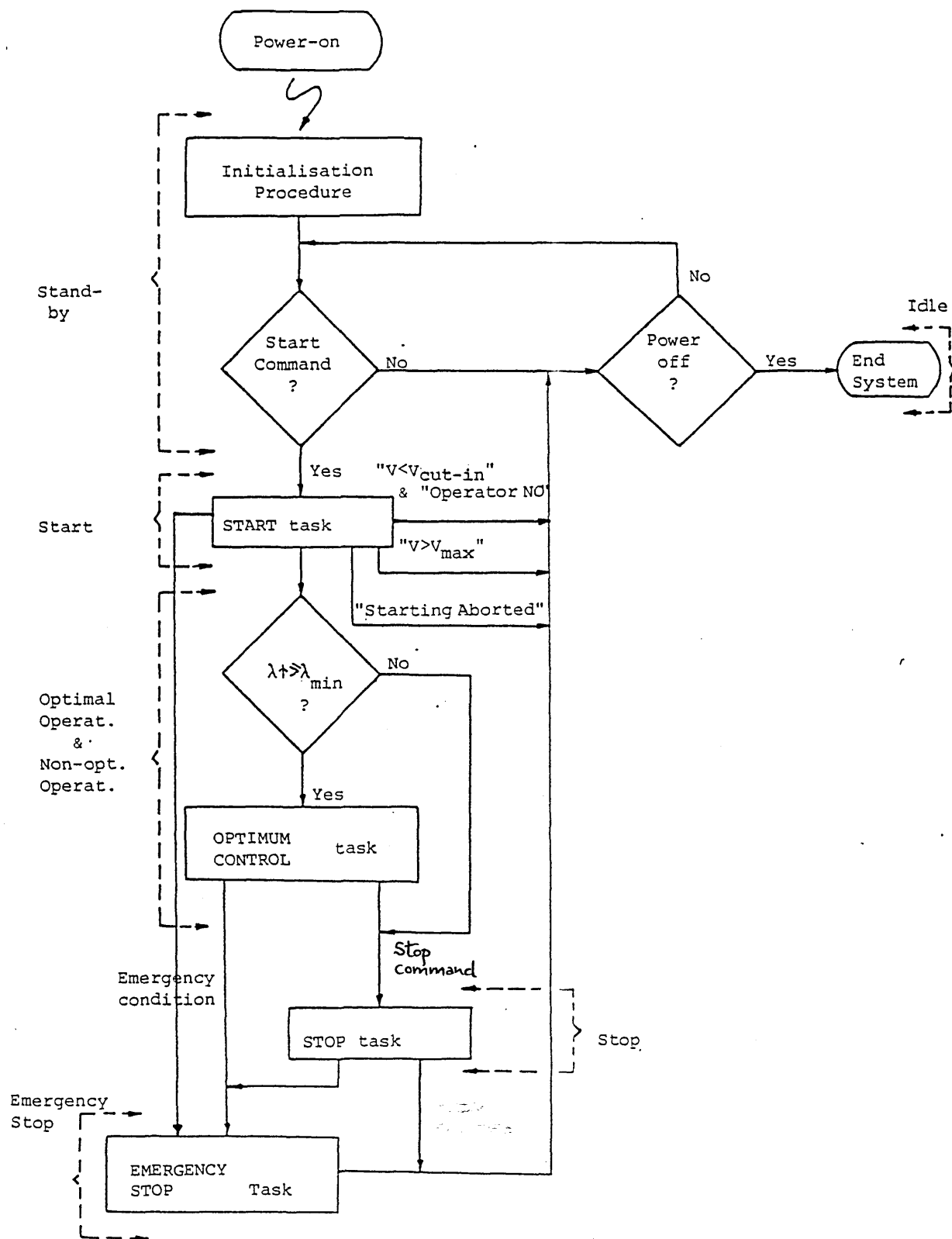


Fig. 3.3 - SCHEDULER procedural description.

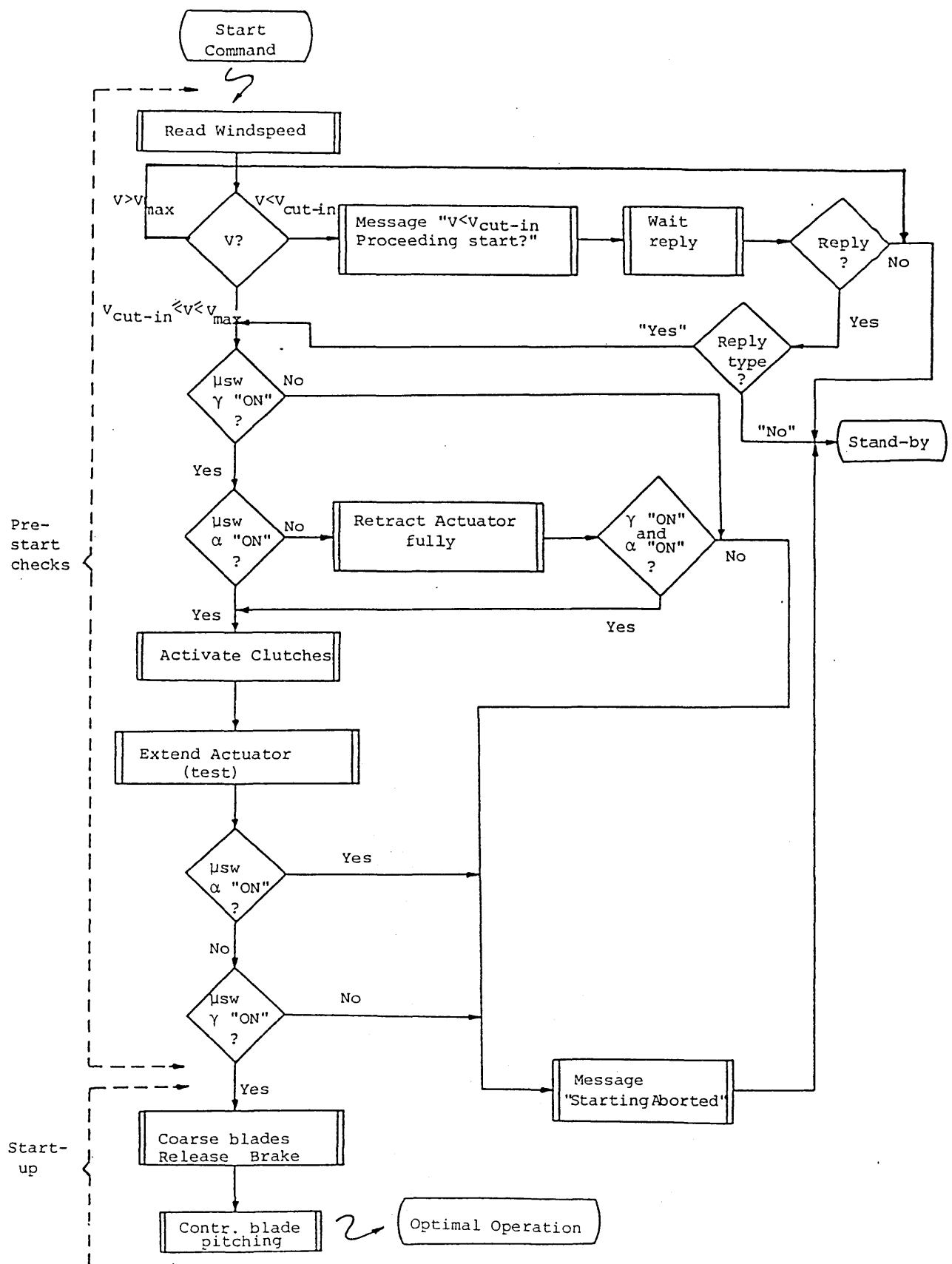


Fig. 3.4 - START task specification.

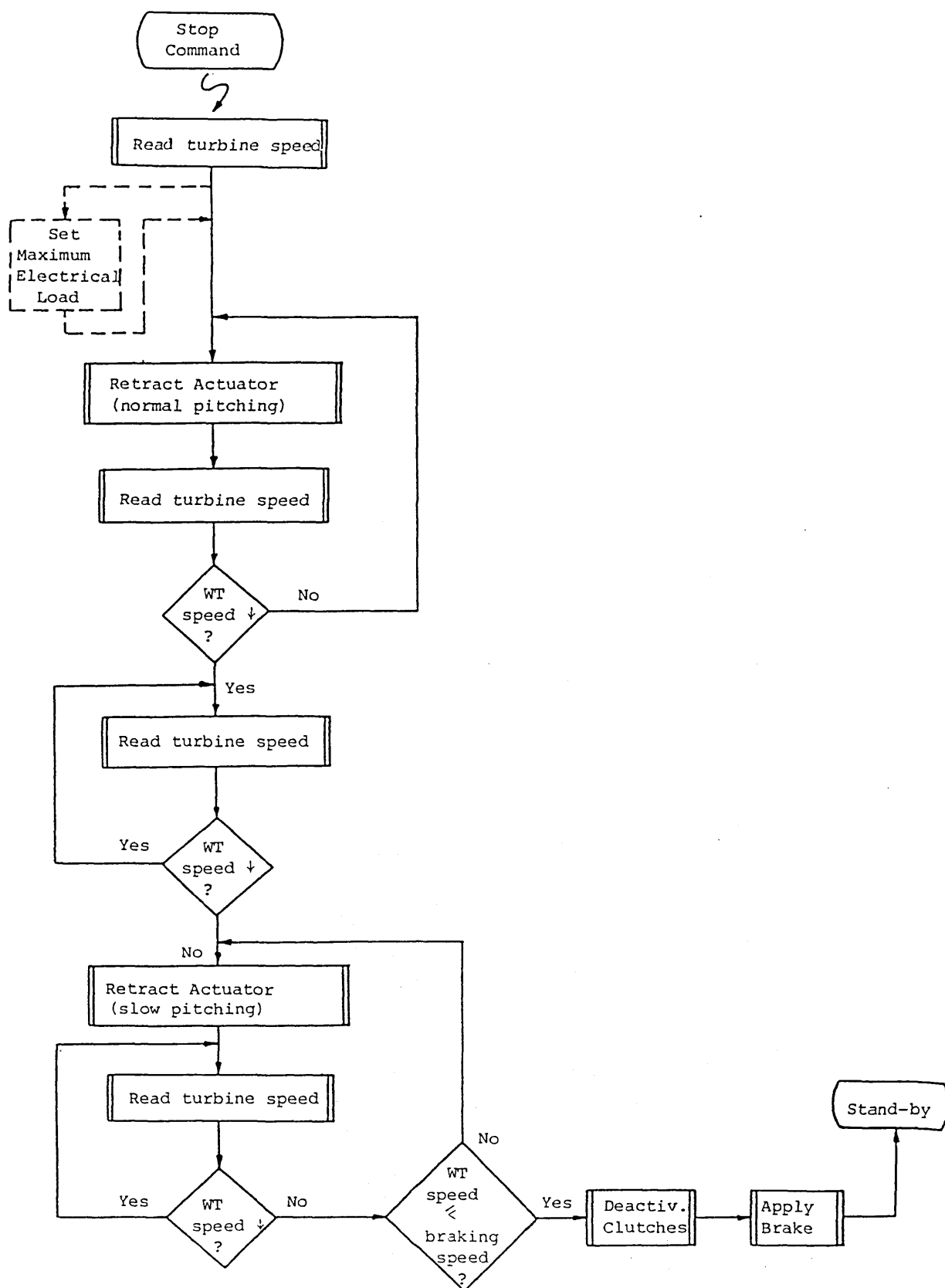


Fig. 3.5 - STOP Task Specification.

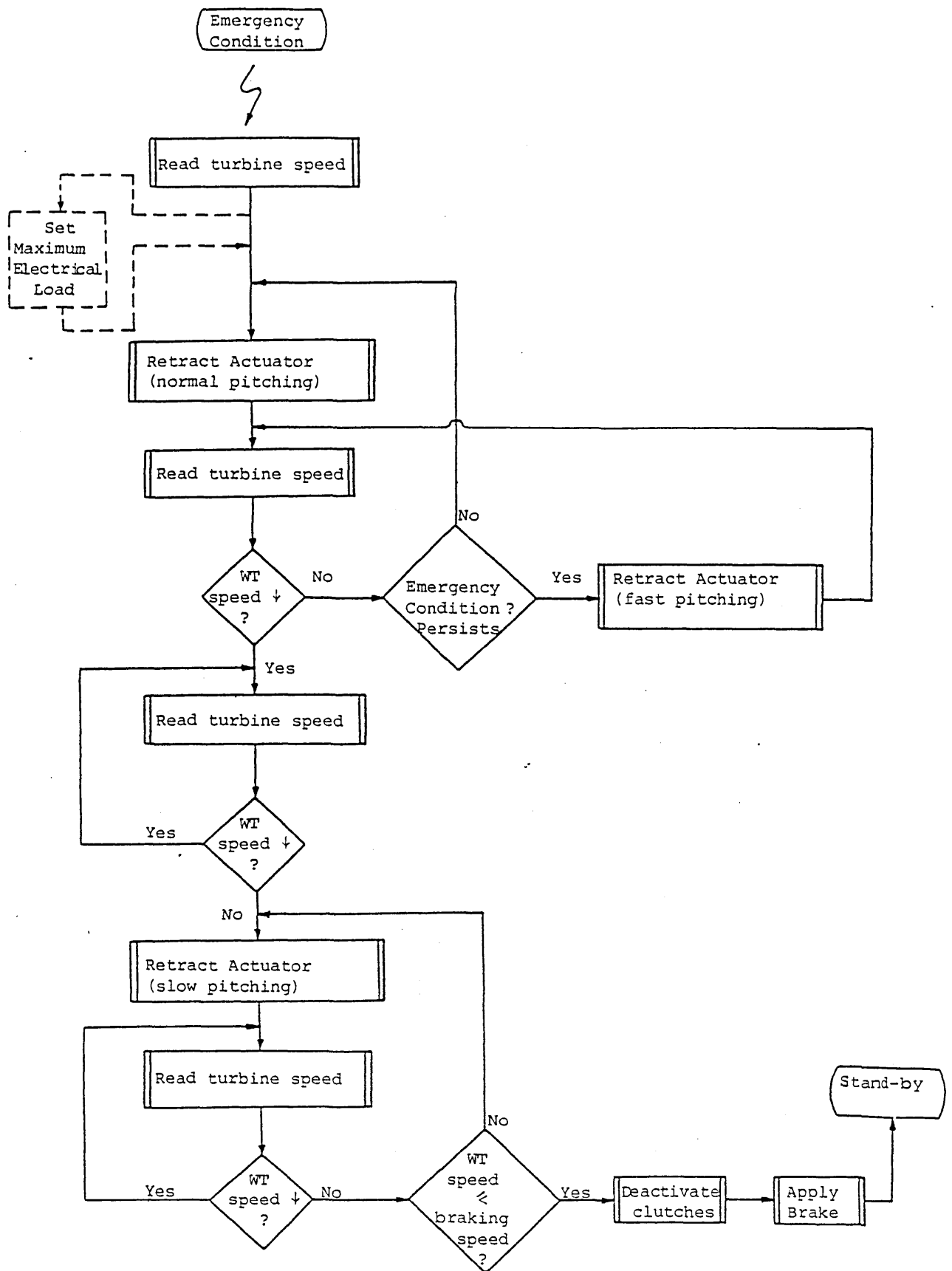


Fig. 3.6 - EMERGENCY STOP task specification.

3.3. STRUCTURED ANALYSIS

The goal of Structured Analysis (SA) is to achieve a structured specification which is a graphic, concise, top-down partitioned, non-redundant and abstract system representation [3.1.].

3.3.1. DFD's, Processes, Sources and Sinks

A graphical tool, the Data Flow Diagram (DFD), is used to represent a system as a network of cooperating processes. These processes receive input data from external sources, handle this data according to their inner logic and send output data to external sinks (see Fig. 3.20). Data can be temporarily deposited in data stores for later retrieval by the same or other process.

Each basic system task is represented by a corresponding process, and sensors and actuators correspond to external data sources and sinks. The names of these processes, sources and sinks are a meaningful statement of their functions. The high level of abstraction maintained will keep the system model free from implementation details.

The State Transition Diagram (Fig. 3.2) can be considered as the level-0 DFD, the so-called context diagram, since it represents the whole system under analysis. The dotted lines are not data paths, they represent the asynchronous control transfers from one operating mode to another. Although control lines are not usually shown in DFD's the nature of the system forces the breaking of the rules for the sake of a more clear representation.

3.3.2. Top-Down Stepwise Partitioning

Each task/process is partitioned into sub-tasks, and then each of these sub-tasks is in turn partitioned, etc., always observing a strict set of rules, thereby achieving increasingly detailed system representations. The level-1 DFD's (Figs. 3.7 to 3.11), show the second level of detail by partitioning the system into its basic tasks

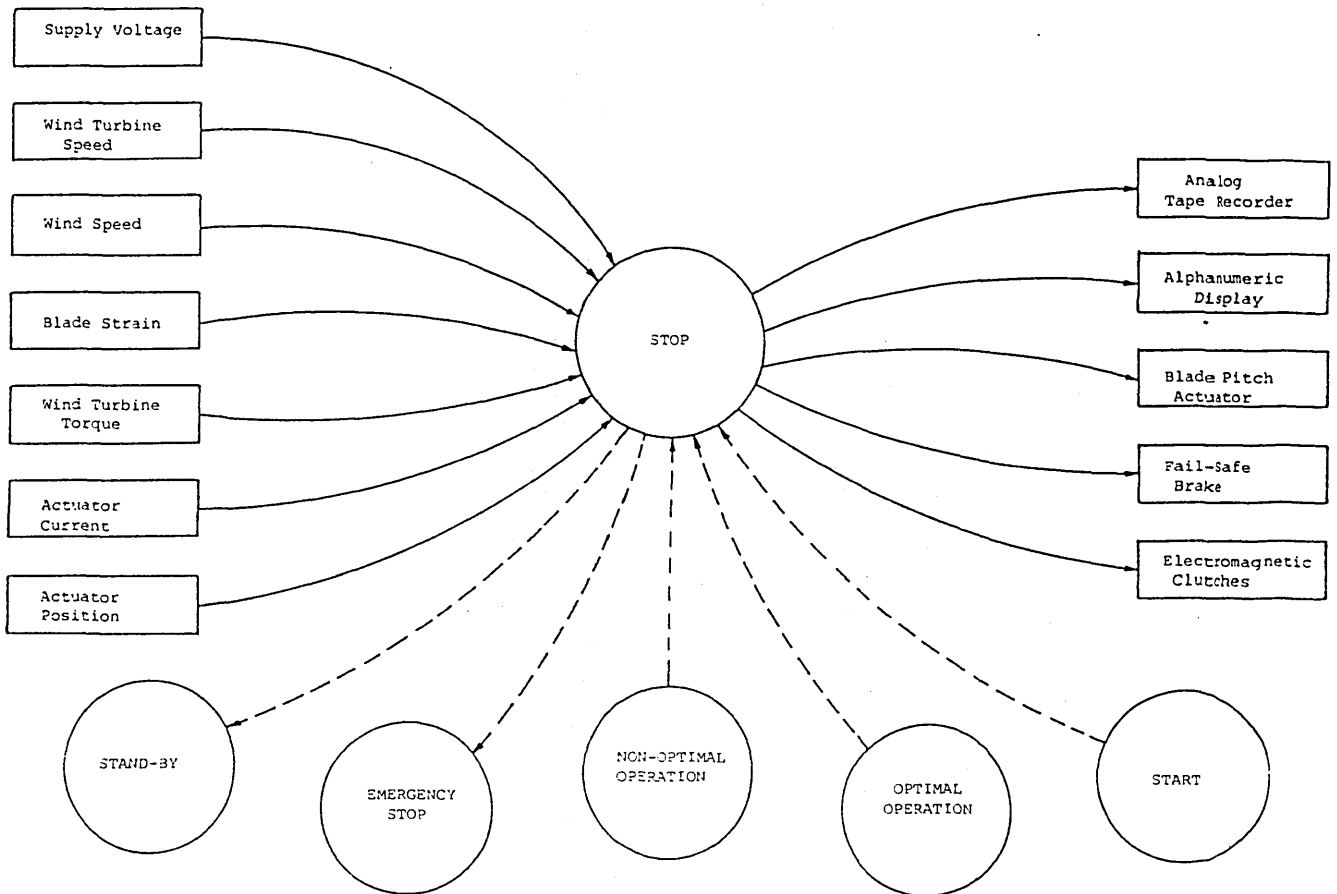
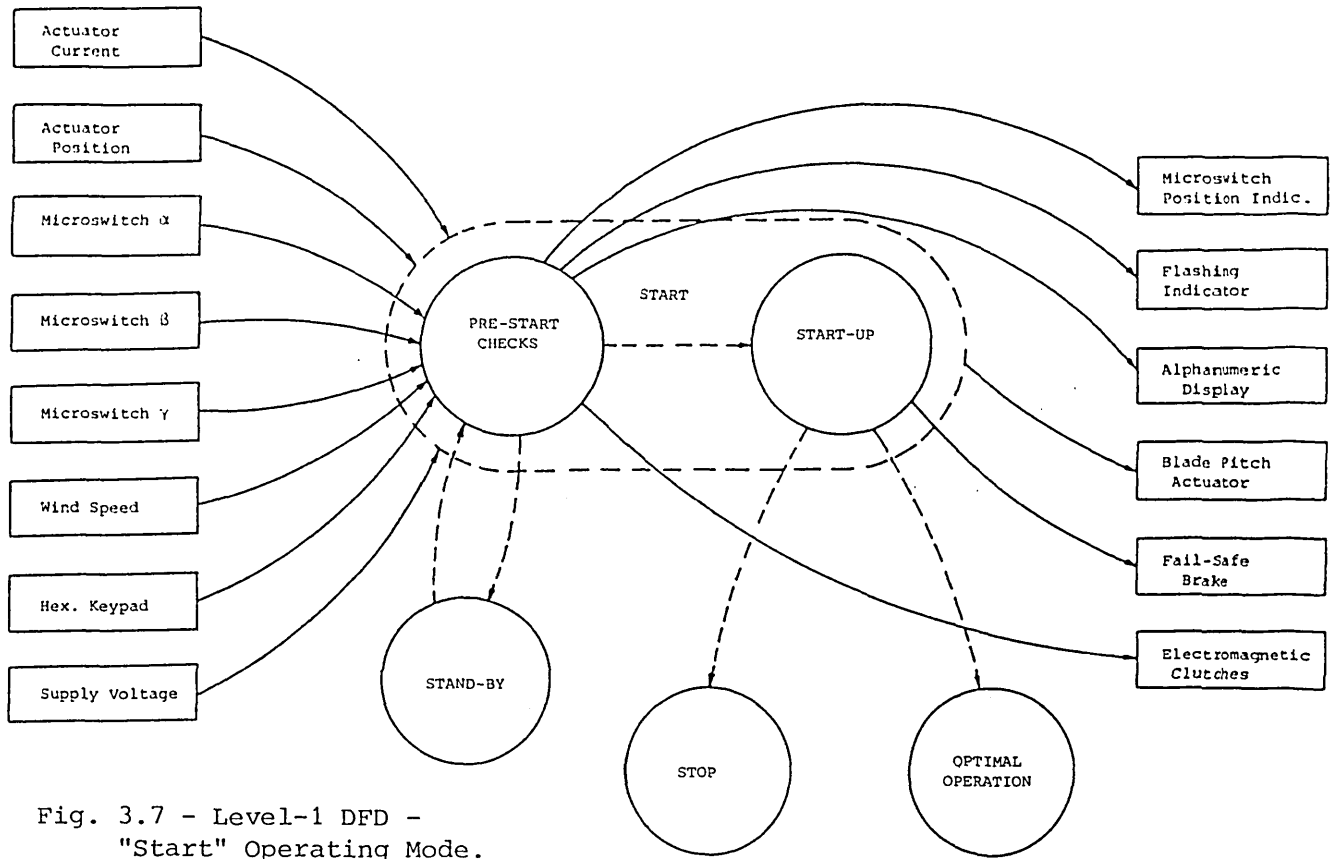


Fig. 3.9 - Level-1 DFD - "Emergency Stop"
Operating Mode.

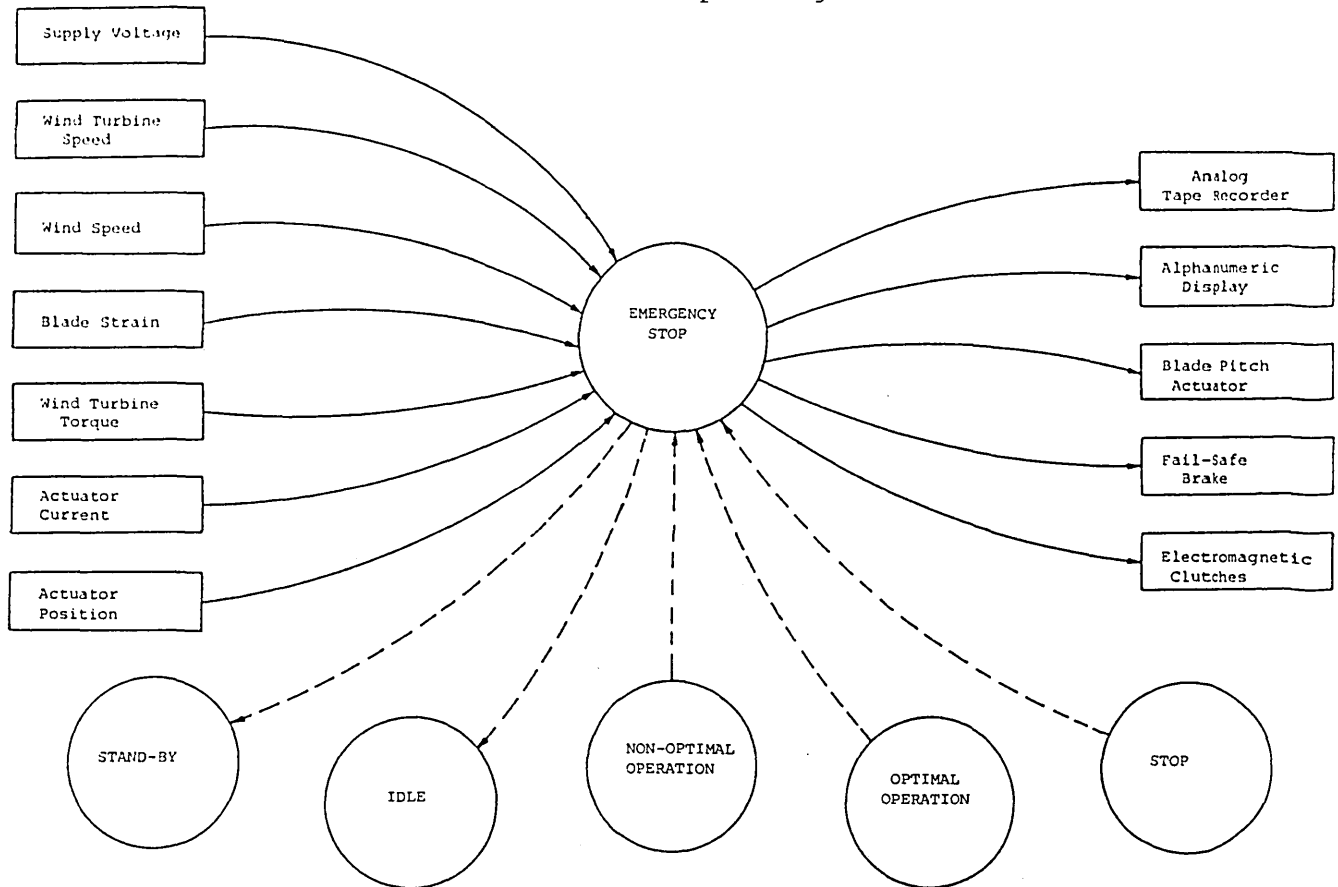
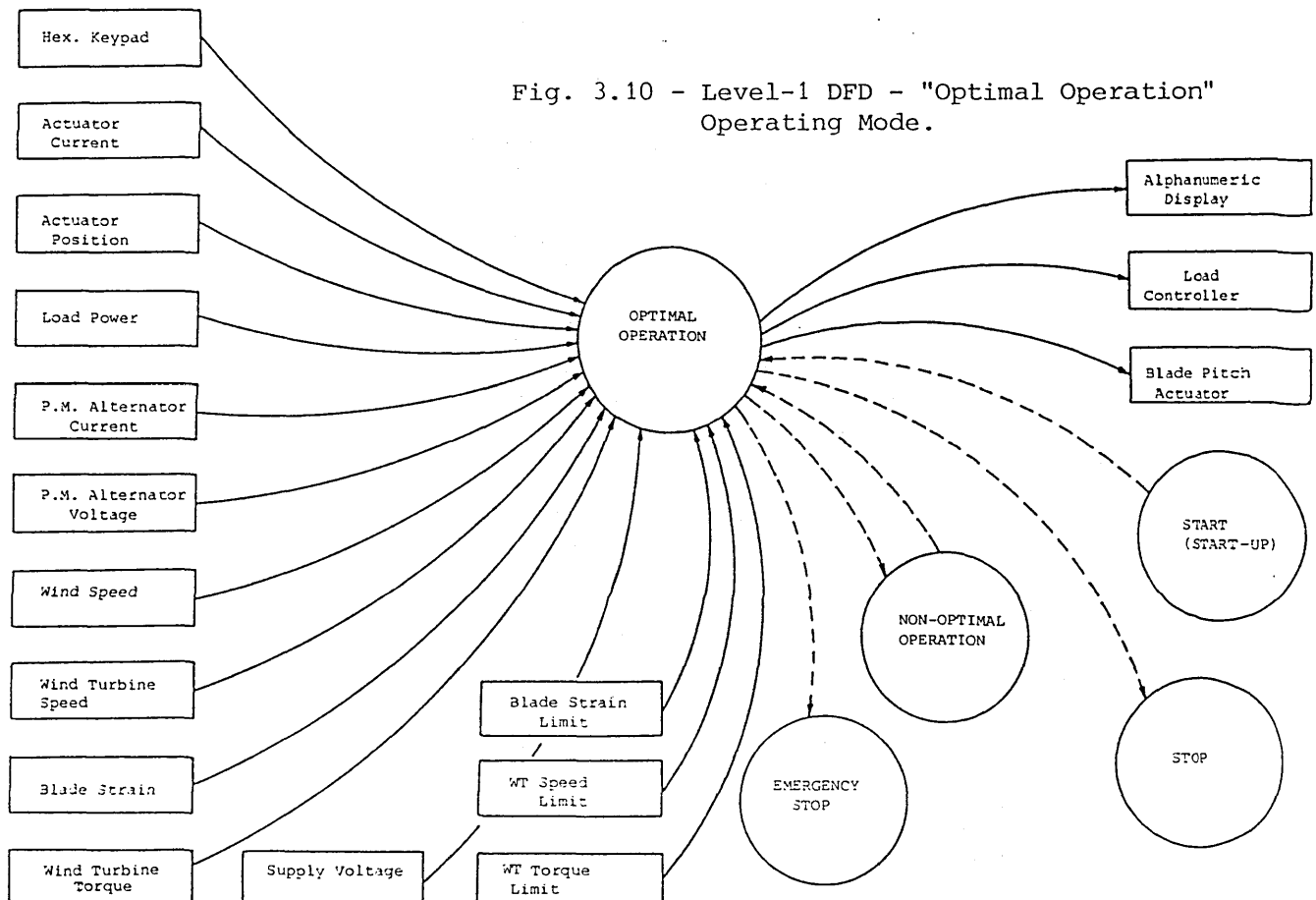


Fig. 3.10 - Level-1 DFD - "Optimal Operation"
Operating Mode.



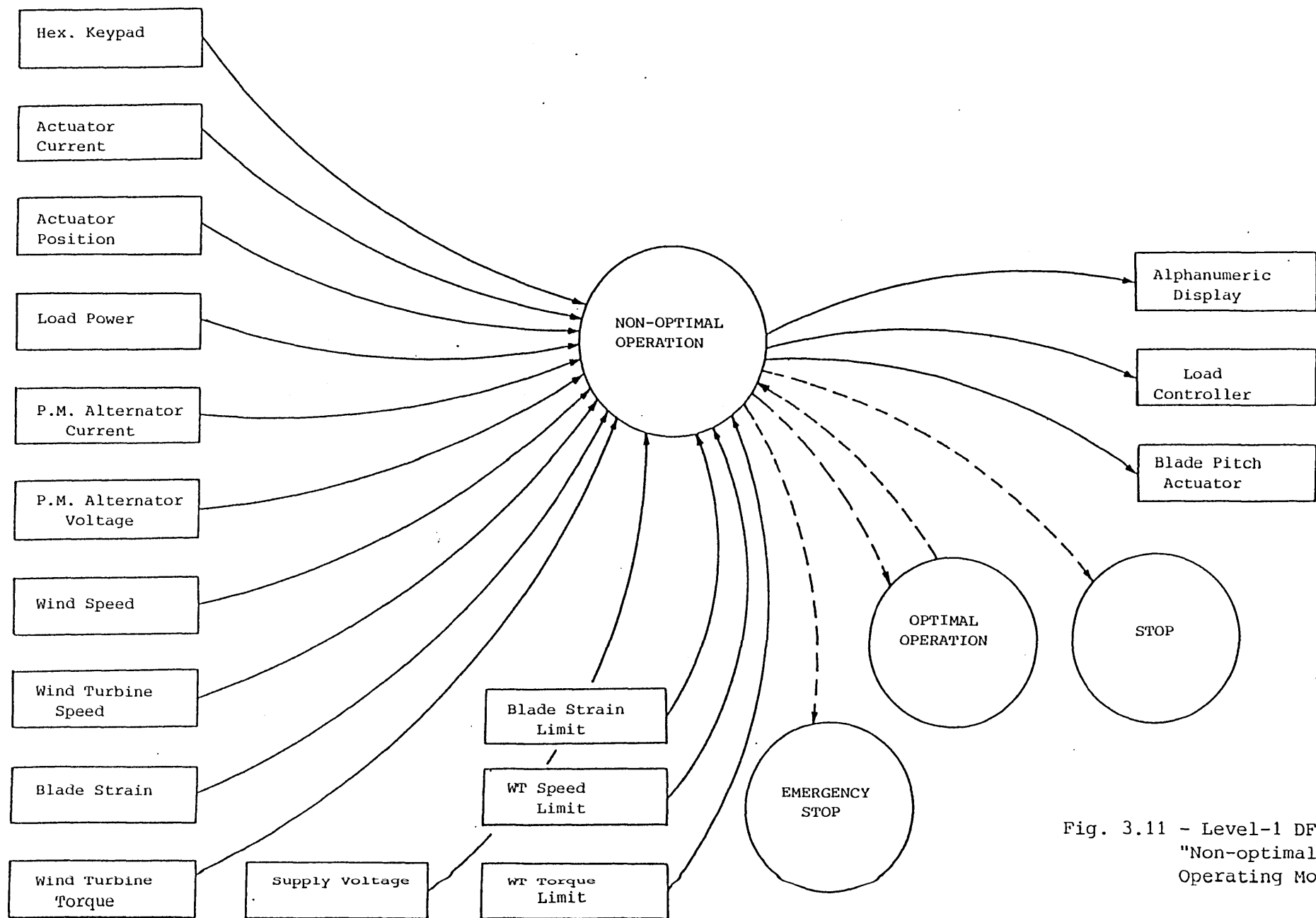


Fig. 3.11 - Level-1 DFD -
"Non-optimal Operation"
Operating Mode.

Start, Stop, etc. The level-2 DFD's (Fig. 3.12) show how one of those basic tasks is partitioned into subtasks. The task implemented in the Optimal Operation state is chosen since no details had yet been given on this task. Two alternative control algorithms, which will be discussed in Chapter 7, are shown.

A leveled set of inter-related DFD's is thus produced enabling the system model to be handled at all levels of detail. Criteria guiding this top-down partitioning will be discussed in a later stage.

The leveled set of DFD's shows both the broad organisation of the system and the specific details of each portion of it. The inherent quality of this representation is the ability to portray a real system, i.e. one of cooperating processes all working simultaneously and all communicating data with one another. The classical flowchart representation simply fails to do this since it can only show procedural descriptions.

3.3.3. Structured Specification

The above described graphical representation is complemented with the Data Dictionary and the Mini-specs. The data Dictionary is a collection of the definitions of all data types, simple or structured, showed in the DFD's (see Table 3.1). The Mini-specs are the descriptions of the inner logic or procedural structure of all elementary tasks, i.e., those that were the final result of the partitioning. The functions of most of these elementary tasks become apparent from their name and from the definition of their input and output data types. The Mini-specs were therefore excluded from this description.

The System Structured Specification will consist of the set of DFD's complemented by the Data Dictionary, the Mini-specs and system physical details. This is the end-product of Structured Analysis and the major input to Structured Design.

```
graph LR; LP[Load Power] -- P1 --> DLP((Detect excessive load power)); MLLP[Max. Load Power Set.] -- P1max --> DLP; DLP --> BPA[Blade Pitch Actuator]; WS[Wind Speed] -- v --> CWF((Calculate ω(λopt.) )); RTSP[Wind Turbine Speed] -- ω --> CWF; CWF -- R --> UCV((Update control variable)); CWF -- λopt. --> CWF; CWF -- ω(λopt.) --> CWC((Compare ω and ω(λopt.) )); CWC --> UCV; UCV --> LCF[Load Control Firing];
```

The flowchart illustrates the proposed fuzzy logic control system for a wind turbine. It starts with two inputs: Load Power (P_1) and Max. Load Power Set. (P_{1max}), which feed into the 'Detect excessive load power' block. This block outputs to the 'Blade Pitch Actuator'. Simultaneously, Wind Speed (v) and Wind Turbine Speed (ω) feed into the 'Calculate $\omega(\lambda_{opt.})$ ' block. This block outputs R to the 'Update control variable' block and $\lambda_{opt.}$ to the 'Compare ω and $\omega(\lambda_{opt.})$ ' block. The 'Compare' block also receives $\omega(\lambda_{opt.})$ from the 'Calculate' block and outputs to the 'Update control variable' block. Finally, the 'Update control variable' block outputs to the 'Load Control Firing' block.

[illegible]

```

graph LR
    WS[Wind Speed] --> CO[Optimal (non-optimal) operation]
    WTS[Wind Turbine Speed] --> CO
    LP[Load Power] --> CO
    CO --> BPA[Blade Pitch Actuator]
    CO --> LCF[Load Control Firing]

```

v	- wind speed
$v_{\text{cut-in}}$	- "cut-in" wind speed
v^*	- wind speed "seen" by the rotor
ω	- wind turbine speed
$\Delta\omega$	- wind turbine speed change
λ	- tip-speed-ratio
λ_{min}	- minimum value of λ for turbine loading
λ_{opt}	- value of λ for which $C_p = C_{p\text{max}}$
T	- mechanical torque
T_{el}	- electrical torque
ε	- blade strain
P_l	- load power
ΔP_l	- maximum load power deviation for optimum blade setting
$\alpha_s, \beta_s, \gamma_s$	- status of pitch mechanism microswitches
l_β	- actuator arm position (blade pitch indirect measurement)
β	- blade pitch control variable
$\Delta\beta$	- blade pitch change
i_β	- actuator current
i_a	- p.m.a. current
v_a	- p.m.a. voltage
p_a	- p.m.a. power
A_c	- actuator control variable
B_c	- brake control variable
C_c	- clutches control variable
L_c	- load controller control variable
R	- rotor radius
t_s	- sampling time
t	- time
J	- total system inertia
η_a	- p.m.a. efficiency
x_{max}	- maximum allowed value for variable x

Table 3.1 a) - DATA DICTIONARY (Elementary Data)

System-state=[Idle:Stand-by:Start:Optimal-operation:Non-optimal-
(Fig. 3.2) operation:Stop:Emergency-stop]

Data-source=[Hex-keypad+Wind-speed-sensor+Turbine-speed-sensor+....]
(Figs. 3.7 to 3.11)

Data-sink=[Alphan-display+Load-controller+Brake+....]
(Figs. 3.7 to 3.11)

Data-store=[(Single:Double)-entry-look-up-table+Buffer+Double-buffer]
(Figs. 3.13 to 3.19)

Single-entry-look-up-table=[$\eta_a(\omega)$]

Double-entry-look-up-table=[$C_q(\omega, V) + T(V, \omega) + V(T, \omega)$]
(Figs. 3.8 to 3.11 and Fig. 3.19)

Table 3.1 b) - DATA DICTIONARY (Composite Data)

3.4. STRUCTURED DESIGN

Like many real-time systems the WECS is an on-line transaction-centered system. The transactions are the stimuli, either external events or internal decisions, that trigger a set of activities within the system leading to a change of state (Fig. 3.2). Two different

tools are used therefore to convert the DFD representation into the actual system model : Transform Analysis and Transaction Analysis. Given the type of the actual system, Transaction Analysis will take over some of the tasks of Transform Analysis, namely in introducing hierarchy and selecting the top-level module.

3.4.1. Transaction Analysis

Transaction Analysis is the first tool to be applied to the Structured Specification and justifies the insertion of asynchronous control transfers in a representation dealing only with data flows.

Hierarchy is introduced at a very high level by having an executive or scheduler process ruling the whole system and all the subordinated processes linked to it and among themselves in accordance with certain rules.

The graphic representation is altered and the system is now built of well-defined independent modules, each of them implementing one task/process (Fig. 3.13). This new graphic representation of the system as a set of hierarchically organised modules, called Structured Chart (SC), shows both data and control transfers (see Fig. 3.20 for graphic symbols). Once more the module names explain their function, and inter-module communication is illustrated by data couples and flags.

The executive or scheduler process, the transaction-center, was not explicitly shown in the DFD's, as it frequently happens in on-line systems [3.1.]. The necessity of such a process had already been realised in the task analysis outline (3.2.2.2.). This top module calls, i.e. transfers control to each of the transaction modules Start, Stop, Optimum control,..., with no data at all. Each of these transaction-processing subsystems is responsible for obtaining its own input and delivering its own output.

3.4.2. Transform Analysis

Transform Analysis allows then the top-down partitioning of each module, again observing a certain set of rules, leading to

increasingly detailed SC's (Figs. 3.14 to 3.18).

The level-0 SC (Fig. 3.13) represents the system at its highest level of abstraction, showing only the basic tasks and the main channels for data and control transfer. Inter-module communication is implemented with double buffers. The partitioning of each of the main tasks into sub-tasks is shown in the level-1 SC's (Figs. 3.14 to 3.18). Each of these SC's can be associated with a correspondent DFD (Figs. 3.7 to 3.11), and therefore complementing information is given by the correspondent procedural descriptions (Figs. 3.3 to 3.6). The next level of detail is given by the level-2 SC (Fig. 3.19), which is derived from the level-2 DFD (Fig. 3.12).

Input/output, error handling and initialisation modules are added at this stage. Afferent and efferent modules are factored and reorganised as to keep the system balanced, i.e. the data processing and decision modules are kept far from the modules dealing with I/O raw physical data.

3.4.3. Structured Design Specification

This levelled set of SC's, complemented with the Data Dictionary, specifications for each of the modules and packaging considerations from the aimed system model - the Structured Design Specification.

The Mini-specs used for the corresponding DFD's primitives may, after some changes, be appropriate here.

The packaging considerations were kept general enough to be implementation independent. They will help the transition from the design to the implementation stage to be done smoothly, by considering the practical constraints of a real computer system. Packaging is quite simple in this case:

- all the software will consist of one EPROM resident load unit with one main program;

The diagram illustrates the control system architecture for a wind turbine generator. It features a central **THRESHOLD CHECK** block that coordinates data flow between several components:

- SCHEDULER**: Connected to the **THRESHOLD CHECK** block and the **OUTPUT** block via dashed lines.
- OPERATOR I/O**: Connected to the **THRESHOLD CHECK** block via a dashed line.
- INPUT**: Connected to the **THRESHOLD CHECK** block and the **Read ADC** and **Read PIA** blocks.
- THRESHOLD CHECK**: The central processing unit that receives data from the **INPUT** and **OUTPUT** blocks and manages the **double buffer (thresholdchecking)** and **double buffer (processing)**.
- BLADE STRAIN & WT TORQUE**: Receives data from the **THRESHOLD CHECK** block and outputs to the **double buffer (thresholdchecking)** block.
- WT SPEED & LOAD POWER**: Receives data from the **THRESHOLD CHECK** block and outputs to the **double buffer (thresholdchecking)** block.
- P.M.A. CURRENT & VOLTAGE**: Receives data from the **THRESHOLD CHECK** block and outputs to the **double buffer (thresholdchecking)** block.
- double buffer (thresholdchecking)**: A buffer that stores data from the **BLADE STRAIN & WT TORQUE**, **WT SPEED & LOAD POWER**, and **P.M.A. CURRENT & VOLTAGE** blocks before it is processed.
- double buffer (processing)**: A buffer that stores data from the **double buffer (thresholdchecking)** block before it is written to the **Write PIA** blocks.
- OUTPUT**: Connected to the **THRESHOLD CHECK** block and the **output buffer** and **Write PIA** blocks.
- output buffer**: A buffer that stores data from the **OUTPUT** block before it is written to the **Write PIA** blocks.
- Write PIA** blocks: Four blocks that receive data from the **double buffer (processing)** and **output buffer** blocks, used for controlling the **Actuator**, **Load Cont**, **Clutches**, and **Brake**.

Fig. 3.14 - Level-1 SC - Threshold Check.

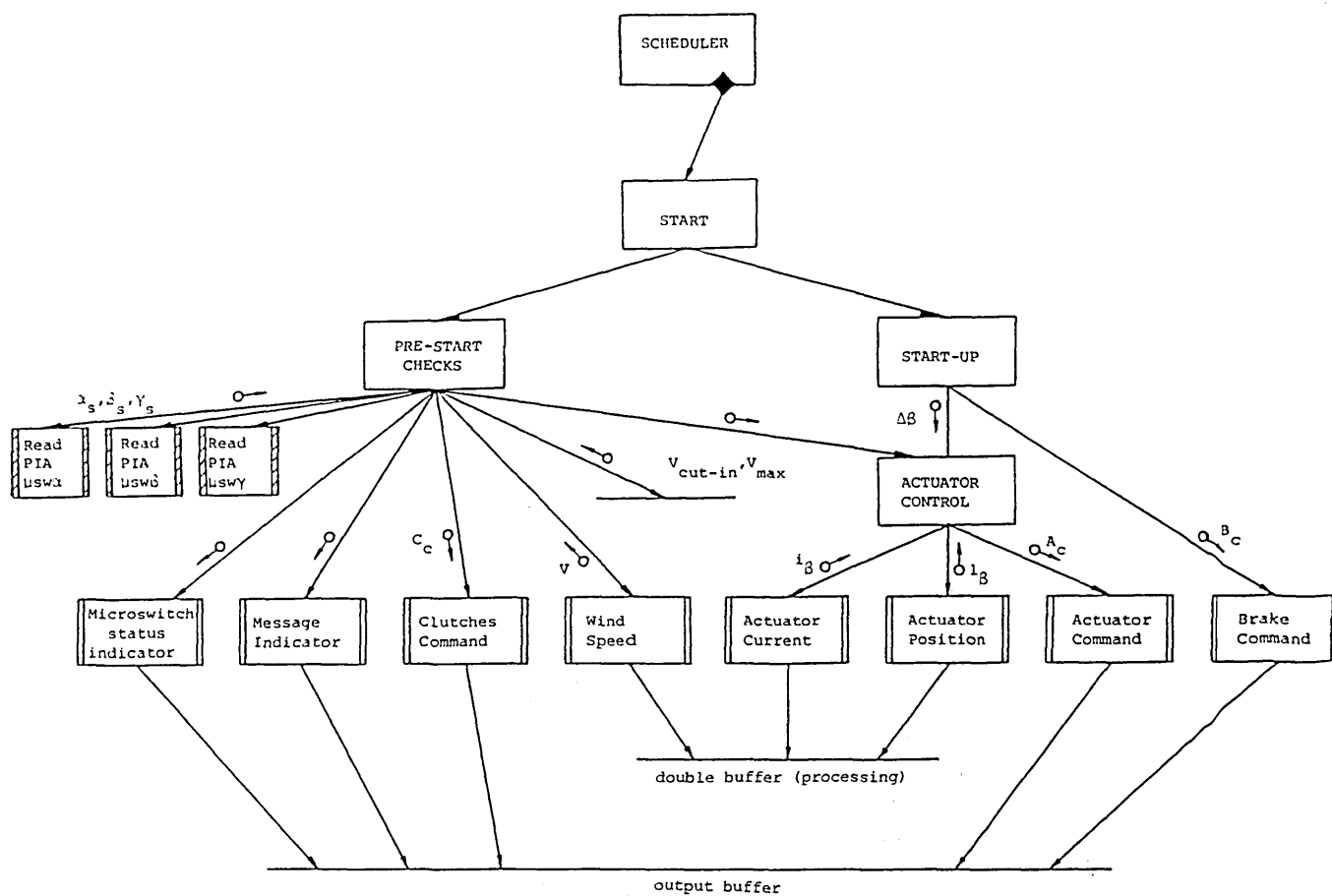


Fig. 3.15 - Level-1 SC - Start.

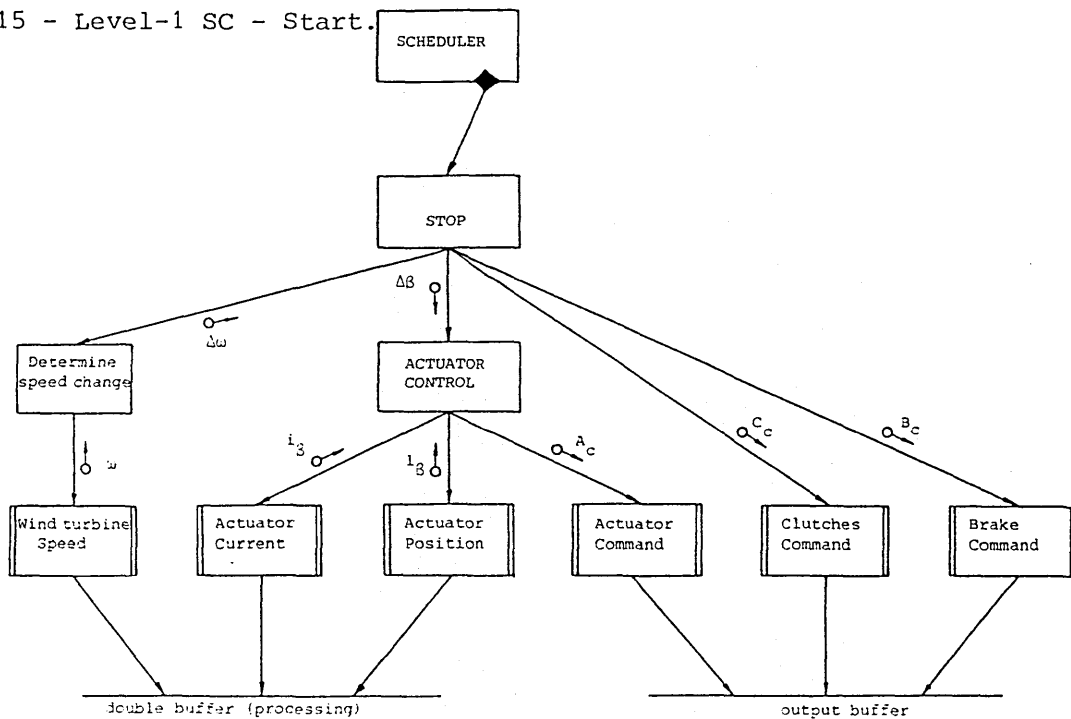


Fig. 3.16 - Level-1 SC - Stop.

Fig. 3.17 - Level-1
SC - Emergency Stop.

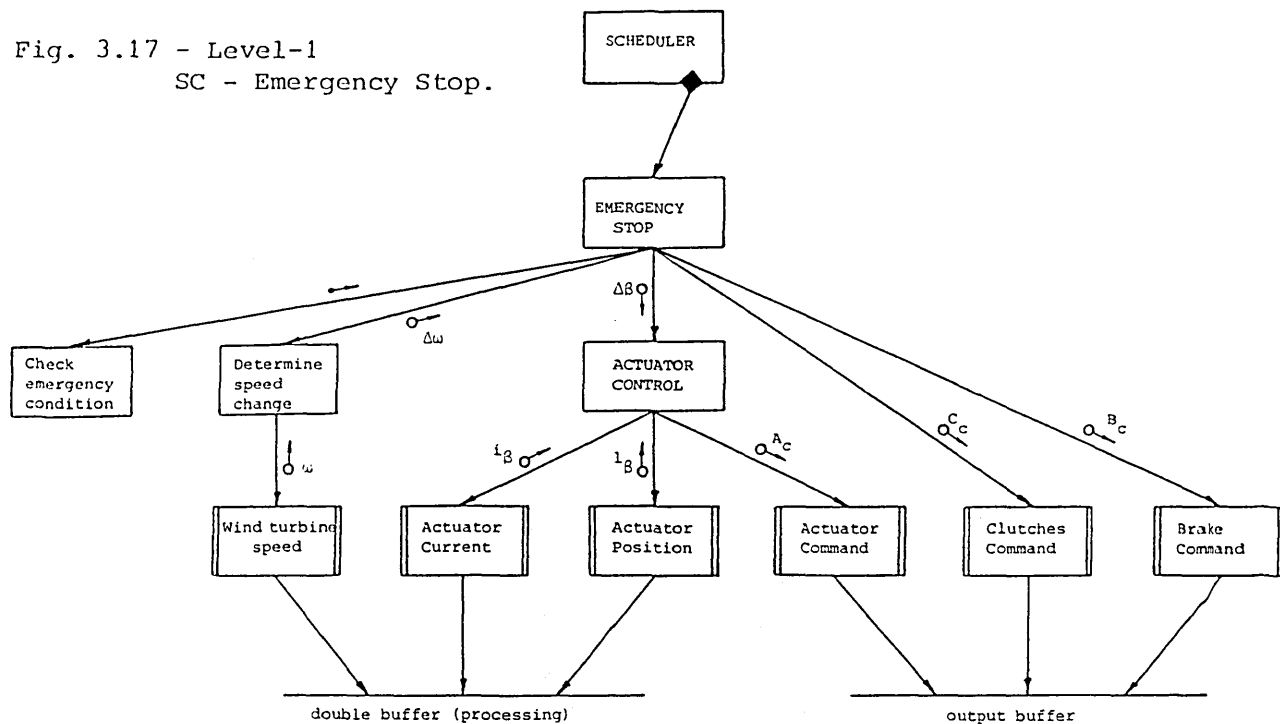


Fig. 3.18 - Level-1 SC -
Optimum Control.

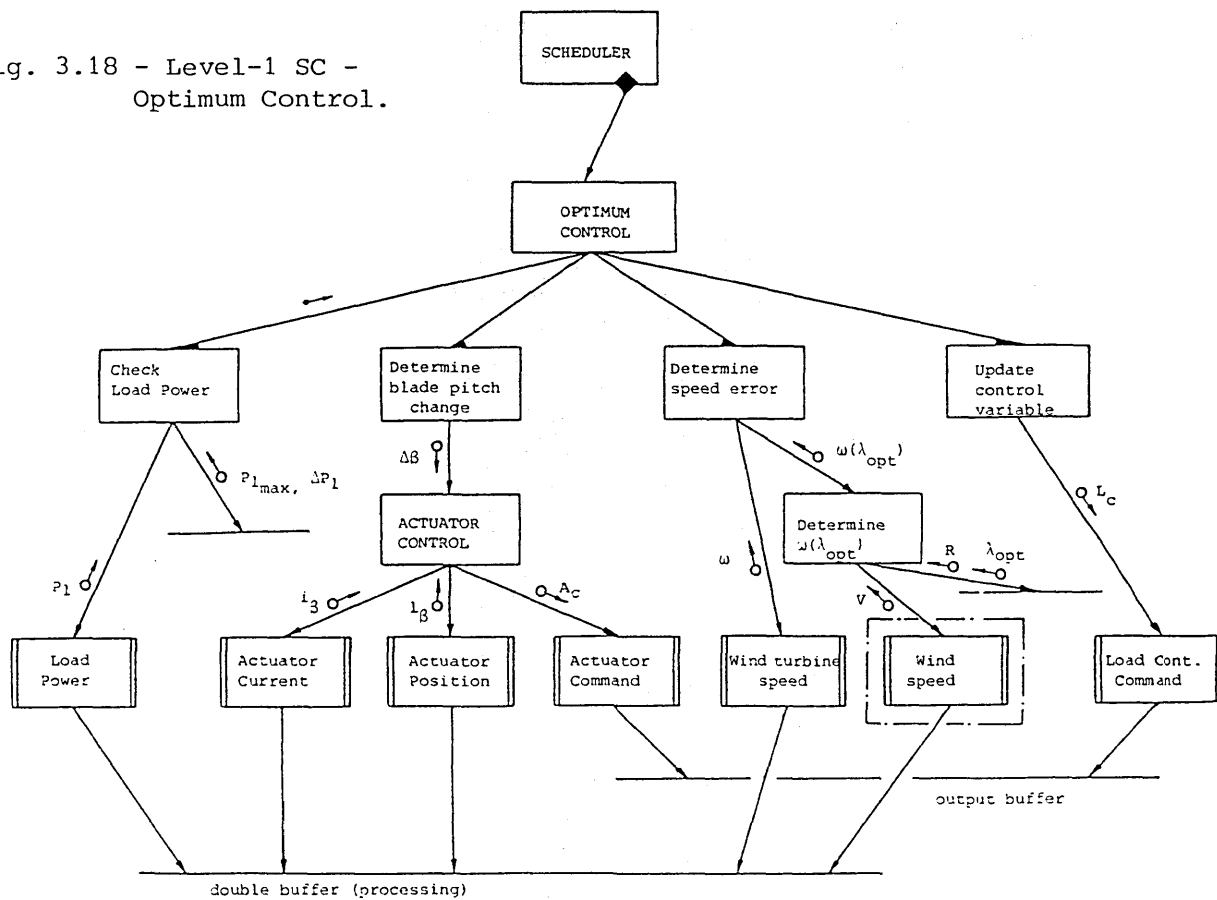
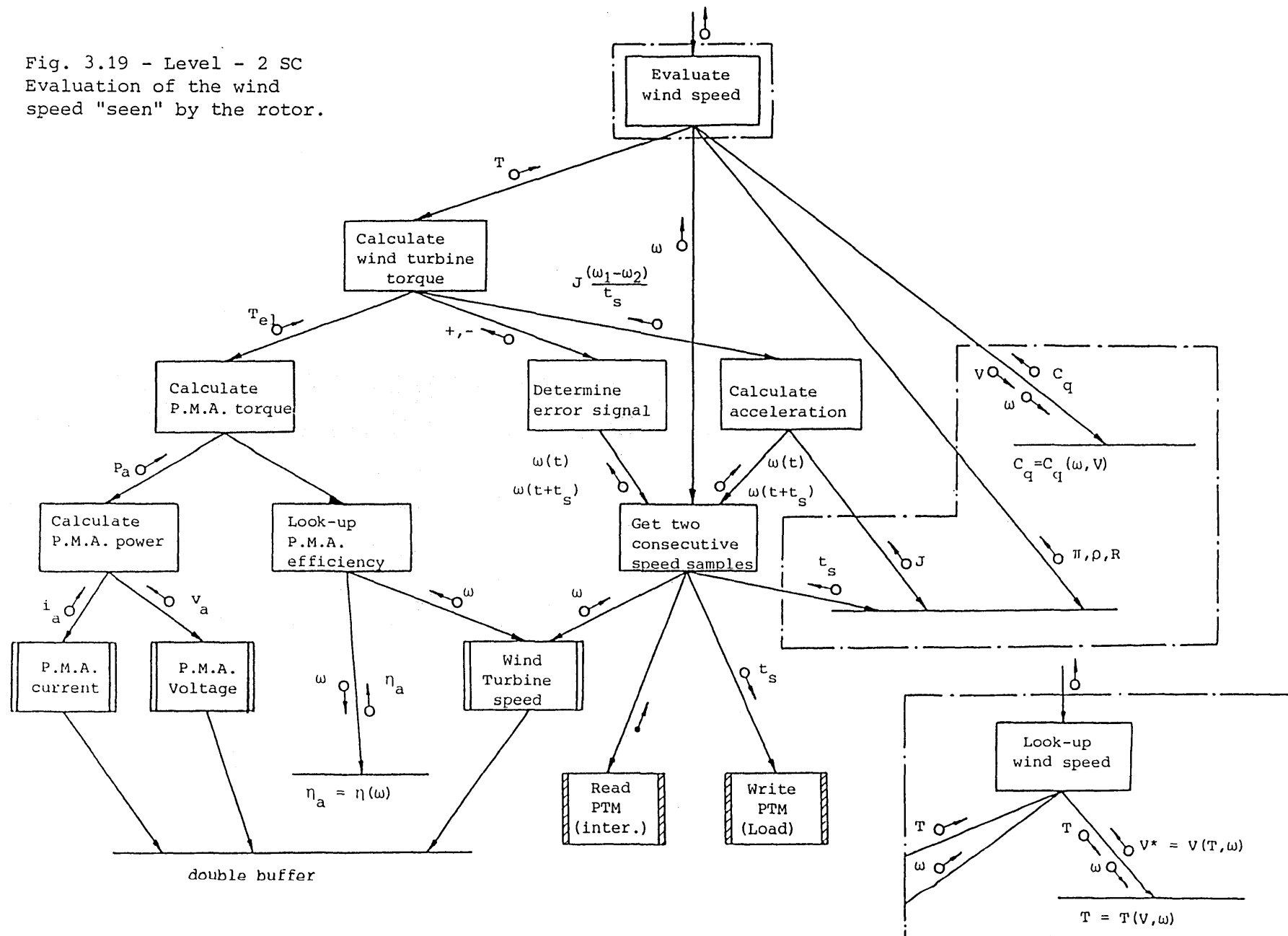
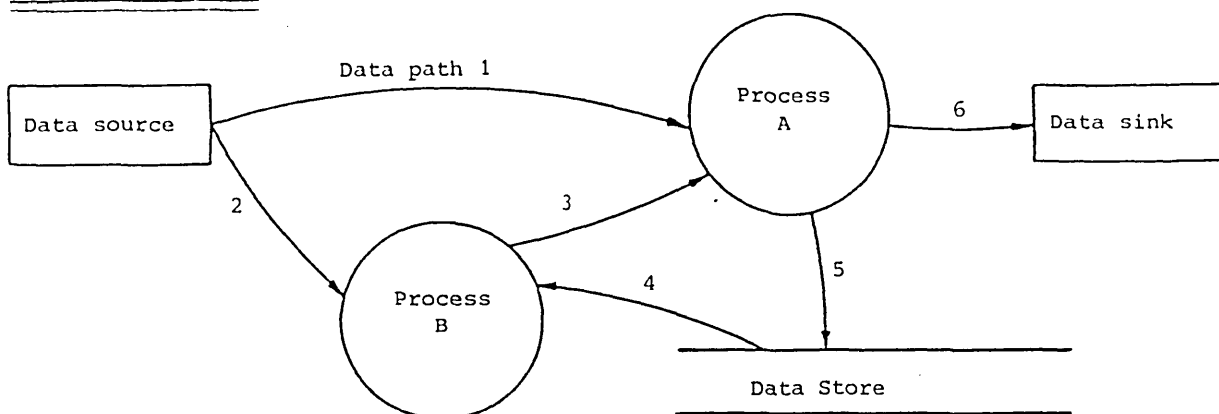


Fig. 3.19 - Level - 2 SC
Evaluation of the wind
speed "seen" by the rotor.

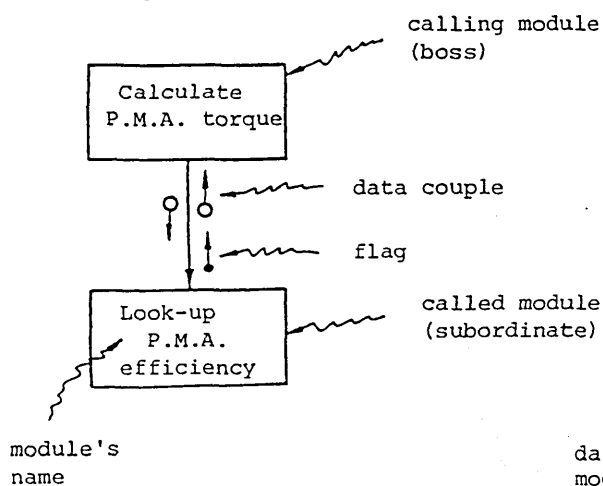


Data Flow Diagrams

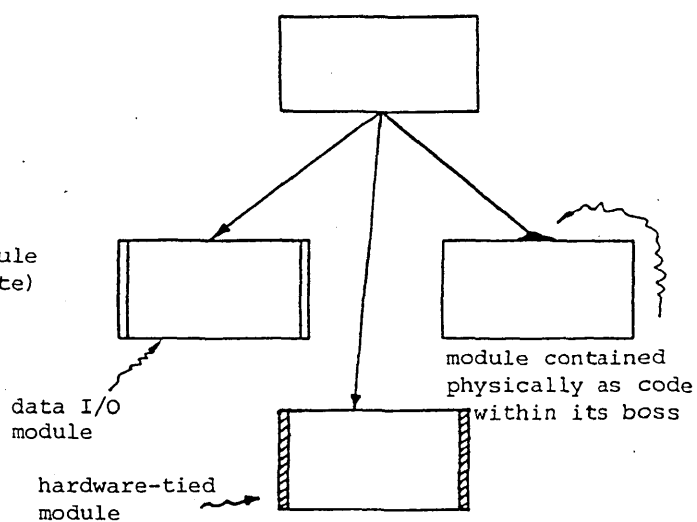


Structure Charts

a) Module Interconnection



b) Factoring and I/O modules



c) Transaction-centered system

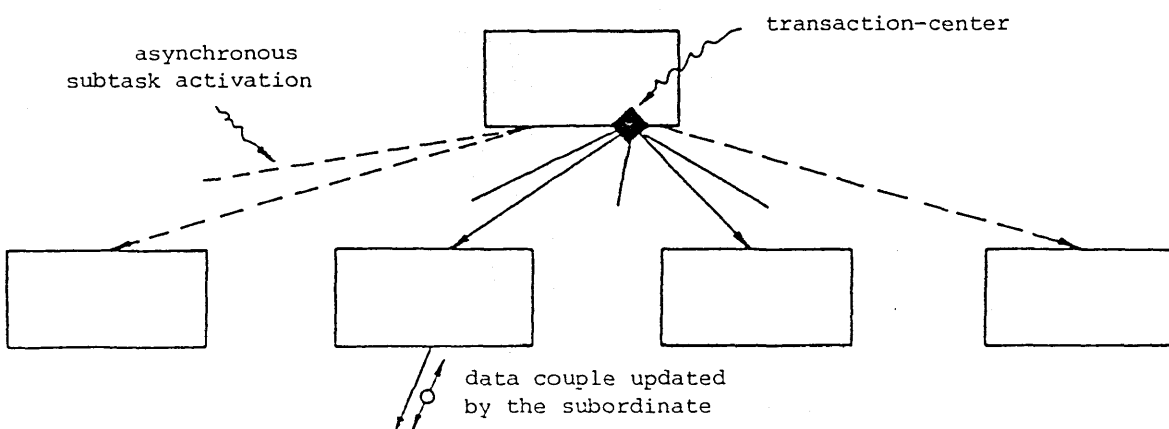


Fig. 3.20 - Graphic Symbols.

- the SC top module will be the main program and the other modules will be procedures or functions;
- library routines performing arithmetics and low-level error checking will also be included at linking time.

3.4.4. Model Evaluation Criteria

SD also provides a set of rules or, less rigidly, guidelines which allow the quality of the design to be evaluated and improved. Top-down partitioning of a complex system into manageable modules is one of the fundamental ideas of SD. It is however necessary that this partitioning is carried out in such a way that:

- the modules are as independent as possible,
- each module carries out a single, problem-related, function.

The degree of interdependency, or coupling, between two modules has to be minimised for debugging and maintenance reasons. Data coupling, due to the inter-module communication parameter passing, and stamp coupling, due to the use of shared data structures, are the sole types of coupling present.

The inner strength of the functional association of elements within a module, or cohesion, has to be maximised. Most of the system modules are highly cohesive since they have functional or sequential cohesion. Functional cohesion occurs when all the module elements contribute to one single, problem-related, task, and sequential cohesion when the module has elements involved in activities such that the output data from one activity serves as input data to the next. The Scheduler, coordinating all its different activities and the flow of control from one activity to the next, has a lower level of cohesion, i.e. procedural cohesion. This is unavoidable since what relates the different tasks activated by the top module is the structure of a scheduling control algorithm.

Data/stamp coupling assure good/medium modifiability and

understandability of the modules. Functional and sequential cohesion produce loose coupling between modules and also good modifiability and understandability.

Other design guidelines [3.1.] were used to improve the system model until as far as the actual coding. They advise on module factoring, decision-splitting, system shape, and also on error reporting, editing and initialisation modules. The actual design comfortably passes a quality test done using these evaluation criteria.

3.4.5. Real Time System Features

3.4.5.1. Statement of the problem

The microcomputer-controlled WECS is a real-time system. That means that the controller has to capture data from sensors, process it, and output control commands to actuation devices at very high speed in order to meet the system timing constraints. It must therefore have some means of keeping track of external events and to be able of fast execution of some time-critical program modules.

To assure that the real-time system constraints are met and that there is no loss of external relevant data, CPU use has to be optimised. The way this can be achieved is strongly dependent on the actual system hardware and software. At this stage though, the aim is to complete the conceptual model built through SADT's in order to allow it to guide the implementation of the real-time computer system. We will therefore restrict ourselves to adding a minimum set of features always maintaining an high level of abstraction.

3.4.5.2. Inclusion of real-time system features

The SC's of Figs.3.13 and 3.14 show some asynchronous transfers of control which suggest breaking from the execution of a certain module to the execution of another one. This expresses the fact that data I/O tasks have to be interleaved with computational tasks. Actually,

in most of the system operating modes the controller has to implement several tasks simultaneously, namely I/O, threshold checking, operator interaction and computation. The tasks Start, Stop, ..., are mutually exclusive since each of them corresponds to a different system state (Fig. 3.2).

Overlapping data I/O, computation and other tasks is a problem with many different solutions [3.2.-3.8.]. At this stage it was decided to include in the model the simplest of these solutions since its implementation would certainly be always possible.

The problem can be reduced to task scheduling and interrupt processing in the following manner:

- the I/O and operator interaction are interrupt-driven tasks and threshold checking can be either an interrupt- or clock- driven task;
- the base-level tasks are called by the Scheduler which selects them for execution according to the system status;
- interrupt-handling and dispatching will preferably be done in hardware thus achieving a fast task switching mechanism;
- the software will be left with solving the "producer-consumer" problem. This is done by using double buffers to pass data between asynchronously activated tasks and simple synchronisation mechanisms like semaphores.

More efficient solutions would require a more powerful implementation of pseudo-concurrency, e.g. using monitors, or even a concurrent programming environment. The actual implementation will depend on the actual hardware and on the software development facilities available.

3.5. CONCLUSIONS AND OUTLINE OF MODEL IMPLEMENTATION

Besides allowing the analysis and design stages to evolve more or less smoothly to a "correct" system representation, the

advantages of SADT's persist even after the realisation of the Structured Design Specification.

At the implementation stage the main advantages of SADT's are:

- provision of a system model clearly showing the functions specified for the system. A valuable stepping-back from the implementation stage to the initial specification is therefore provided,
- representation of the partitioning of the system functions and existing hierarchy,
- definition of all the boundaries between system modules and all data and control links,
- maintainance of an high degree of abstraction and thereby assistance in software and hardware development to progress without immediate ties to specific pieces of equipment. This will allow information from advanced design stages to be fed back into early stages in an iterative improvement process.

The model implementation stage can thus be carried out in a neat structured manner.

The Structured Chart, with its inherent hierarchy, will be the basis for the software development. System partitioning into subsystems and neatly defined hardware/software boundaries will guide hardware development and, in a later stage, software/hardware integration.

CHAPTER 4

SOFTWARE DEVELOPMENT

4.1. INTRODUCTION

To enhance clarity, the development of software and hardware is presented in separate chapters. However, because of their parallel development and the strong interactions at the various stages (Fig. 2.4) the decision of where to subdivide proved difficult.

This is particularly acute in the issue of the microcomputer system specification which is both software- and hardware-dependent. The former, which is presented in this chapter, covers the choice of the microprocessor around which the computer is to be based, a rather software-related problem from the user point of view. The latter is presented in Chapter 5 and completes the microcomputer specification using plant-dependent information.

The hardware/software split is also blurred when dealing with some practical implementation issues, such as development facilities and strategic decisions. These are presented in this chapter although they contain many hardware-related points.

4.2. DEVELOPMENT FACILITIES

4.2.1. Identification of the Development Aids Required

The system model obtained through SADT's, irrespective of its power, is insufficient to enable the iterative design process described in 2.3. The conceptual design creates the necessary conditions, but these are not enough to enable the practical implementation of the described development strategy. Given the system complexity, the requirements placed on the on-line controller and the extensive software/hardware integration, powerful software and hardware development facilities are necessary. The specification of the development facilities must take into account that:

- the model obtained using SADT's will only be totally useful if structured programming practices are used;
- the volume of software to be written requires a high level programming language [4.10., 4.18.];

- to develop and test real-time software with such a variety of non-standard interfacing, powerful debugging tools are needed.

Two approaches to software development are possible:

- (i) The traditional approach is to do the development work in the computer used for the target system. A minicomputer provided with development software and debugging packages, must then be used. This would have the advantage of keeping the development facilities near the WECS in the field throughout its lifetime. However, since the target controller hardware has to be as inexpensive as possible, this approach is impractical.
- (ii) Use a Microprocessor Development System (MDS). This will provide aids and facilities for developing software and for testing it in the actual system when required.

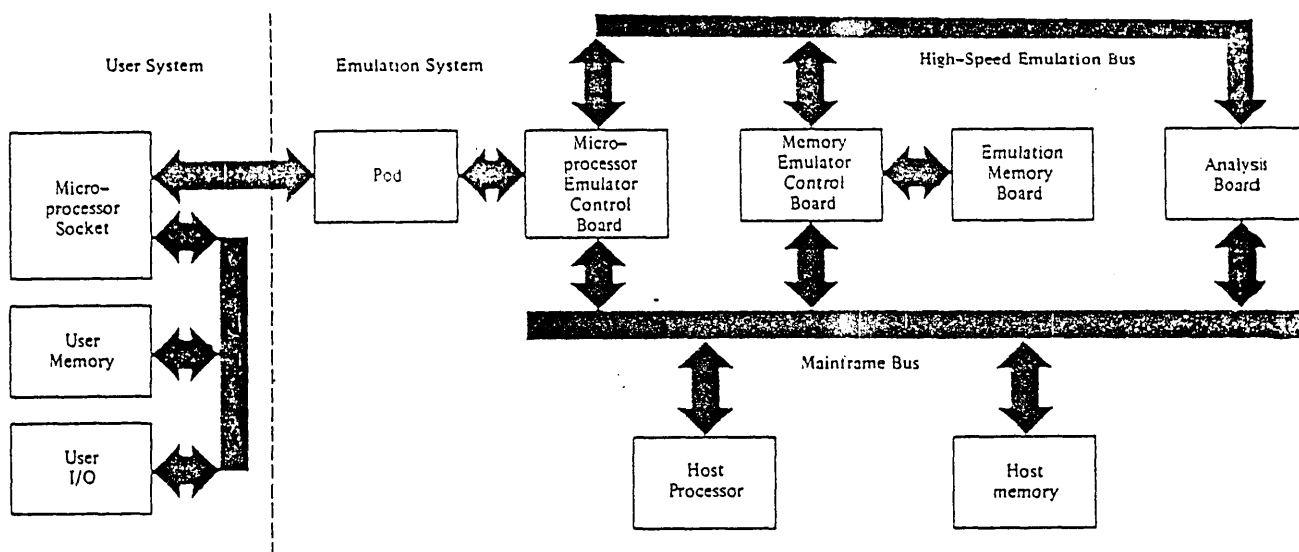
4.2.2. The HP-64000 Logic Development System

An HP-64000 Logic Development System (LDS) meeting all requirements and providing an extensive range of facilities for micro-computer-based systems development was the obvious solution. The HP-64000 LDS is an universal microprocessor development system which has full development facilities for several 8-bit and 16-bit microprocessors (Fig. 4.1.).

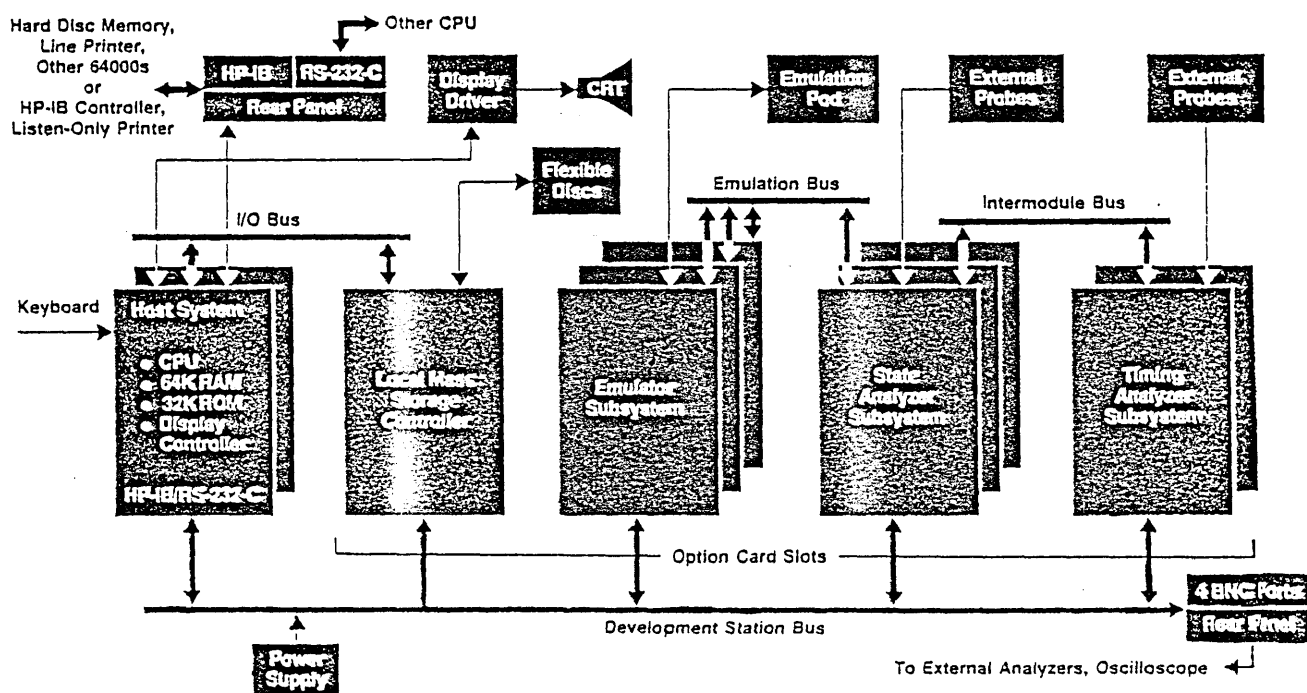
4.2.2.1. Facilities provided

In brief, the facilities provided cover [4.1.,4.2.] :

- coding/implementation
Host environment is provided to develop software for target microcomputer systems including editors, assemblers, compilers, linkers and loaders.



a) Emulator subsystem configuration (from [4.2.])



b) General system configuration (from [4.3.])

Fig. 4.1 - The HP-64000 Logic Development System.

- debugging/testing

Non-disturbing debugging facilities, as required to test real-time software, are offered in the form of an emulator/logical analyser system.

In-circuit emulation (ICE) offers the capability of controlling and monitoring the behaviour of the microprocessor on the target system.

The logical analyser allows the tracing and recording of pre-specified events during the program run to be done, while a time count feature provides performance information.

An inverse assembly facility with address mapping functions is also included.

- loan facilities

The emulator can lend to the target system other development system facilities like RAM/ROM.

- PROM programming

A modular PROM programmer allows the developed software to be programmed from the MDS into a wide range of PROM/EPROM chips.

4.2.2.2. Software development cycle

The software development cycle consists of the following steps:

- i) code is entered either in Pascal or Assembly through a screen oriented editor thereby producing source files;
- ii) these are compiled/assembled giving relocatable files;
- iii) a linker/loader is then used to link all the user files with the system or user monitor and library files, and load the absolute file in the specified blocks of emulation or target system memory;
- iv) the program can then be run with the help of the emulation system. Powerful debugging facilities are then provided by the emulator/logic analyser system such as:

- running or stepping the program from any address or block start;
 - software or hardware breakpoints;
 - symbolic debugging facilities;
 - versatile memory display and modify commands;
 - tracing the program while it runs using various tracing modes and a different number of triggering conditions.
- v) back to step i) for the necessary code changes until the software is completely debugged.
- vi) the program is then transferred into PROM/EPROM.

4.2.2.3. Specific features of the development system

The HP-64000 Pascal implementation has a number of features common to the programming language C, with the 2-pass Pascal compiler extensions enabling access to a number of programming features usually a privilege of assembly language [4.1.,4.3.]. The restrictions of this Pascal subset will be referred whenever it becomes convenient.

The major extensions are the following:

- since absolute addresses can be accessed peripheral hardware can be regarded as memory mapped devices;
- many options for data representation are included, e.g. using 1, 8, 16, 32 or 64 bits, to represent boolean, integer, character, real, long-real, byte ... variables;
- bit manipulation is possible by doing logical operations on sets and having shift and rotate instructions on bytes;
- data and program blocks can be declared as global or external allowing separate compilation/assembly and also the linking of Pascal and assembly files;
- data can be separated from program code for further allocation to RAM and EPROM;
- code for interrupt and trap handlers can be compiled from a Pascal source program.

Low-level hardware-tied modules can thus be written in Pascal therefore keeping all the advantages of a structured high level language.

The linker/loader and emulator options provide extensive facilities for software development and software/hardware integration. Emulation can be done without any external hardware, which is very convenient in the early stages of software development (4.4.1.2.), or with the target system connected to the development system, the so-called in-circuit emulation (ICE) (4.4.5.). Independent host and emulator buses (Fig. 4.1) allow real-time ICE to progress in parallel with other development tasks, e.g. editing, compiling, etc., greatly improving the system turnaround time.

The only weak point of the HP-64000 is that it does not include facilities for the development of concurrent software.

4.3. STRATEGIC DECISIONS

4.3.1. Choice of Programming Language

When weighing the pros and cons of using Pascal or Assembly, the programming languages provided by the development system, the trade-offs involved can be summarised as follows.

The advantages of Pascal over assembly language are:

- readability and abstraction level
 - these are vital features, given the complexity and size of the software and the aim of system versatility and expandability.
- structure
 - to take full advantage of the design techniques used to build the system model (Chapter 3), software development has to be supported by structured programming techniques [4.6.,4.8.]. Pascal provides constructs encouraging the use of those techniques [4.5.,4.7.]. Structured assembly programming is possible but still rather limited [4.9.].
- software portability

In this respect the benefits are twofold :

- no commitment to a specific microprocessor or family is necessary at this early stage of development,
- an eventual upgrading of the controller may be done by

replacing solely the microcomputer hardware.

The disadvantages of using Pascal are as follows :

- memory space
 - nowadays memory constraints are certainly not a problem neither in availability nor in price. Still the HP-64000 Pascal compiler generates good space-efficient code [4.2.] by using a very compact intermediate language (IL) representation.
- speed of execution
 - due to the program size and to the 68000 vast and complex instruction set, the efficiency of the compiler produced code is hardly matched even by an experienced assembly language programmer.
 - time-critical software modules, which are a very small percentage of the total software, can be written in assembly and linked with the bulk of the software written in pascal.
 - the compiler has also an "Optimize" option which produces smaller and faster code by ignoring certain run-time checks.
- access to the peripheral hardware
 - allowed by the Pascal compiler extensions (4.2.2.3.).

The decision was therefore taken to write most of the software in Pascal and use assembly programming only when absolutely necessary.

Although other languages are now becoming increasingly popular for control applications like C, Ada, Modula 2, etc., the use of Pascal as a programming language for real-time systems is still one of the best choices if adequate support is available.

4.3.2. Postponing the Microcomputer-System Specification

The specification of the controller microcomputer is one of the major steps in the development, since the actual microcomputer hardware will determine some of the major controller features, namely:

- responsiveness to the controlled environment, dependent on the program speed of execution;

- interfacing capability, constrained by the actual microcomputer I/O subsystems;
- implemented functions, conditioned by allowable program/data memory space and other facilities.

Major benefits are thus to be obtained if this specification can be based in advanced software/hardware design information such as :

- control program speed and memory requirements,
- detailed interfacing specifications,
- hardware/software trade-offs involved.

Effectively tying down the microcomputer specification to actual pieces of equipment could be conveniently postponed to an advanced stage of software and hardware design by taking advantage of the HP-64000 facilities. The approach to software development consisted of the following steps :

- i) the HP-64000 LDS was used to develop a first high-level version of the controller software. This simulation version could be debugged without external hardware.
- ii) the simulation version provided information on memory and execution speed requirements placed by the software, which were then used as part of the microcomputer hardware specification.
- iii) once the target microcomputer was available, the low-level hardware dependent software modules were developed and all software was integrated as the actual control program. Testing was then extensively done with the target system using ICE.

4.4. SOFTWARE STRUCTURE, DEVELOPMENT AND TESTING

4.4.1. The Simulation Version

4.4.1.1. Structure

In the first stage of software development a simulation version of the control program is almost completely built up

by directly programming the algorithms implemented by the SC modules and adequately linking them.

Each module is implemented by a Pascal procedure or function and practically all the existing data and control links are neatly implemented by parameter passing or using boolean flags (see SC's in Chapter 3). The use of Pascal block-structured constructs eases the obedience to structured programming rules.

The Scheduler algorithm produced the skeleton of the main program. According to the system status this main program decides at any moment which procedures to call. The Scheduler and the base-level modules, i.e. the ones implementing the basic system tasks, like Start, Stop, ..., can be regarded as a sequential program. The Scheduler calls them as to implement the on-line transaction-centered system. Since there was no target system hardware to support the inclusion of real-time features, the modules Threshold, Input, Output and Terminal are included as sequential programs, and their control links with the Scheduler couldn't be implemented. The low-level I/O procedures were simulated by modules performing a simplified version of their functions.

4.4.1.2. Implementation and Testing

Emulation with no target hardware allowed this simulation program to be debugged. A top-down incremental implementation approach was used to integrate all the software modules. The top module, the Scheduler, is implemented first with each of the base-level modules simulated by a stub or dummy module. Integration then proceeds to the next level, etc., down to the bottom of the SC. At each level stubs are used to simulate the unimplemented modules of the level immediately below. They display trace messages or implement a simpler version of the real module.

All the different program paths can be exercised and checked by using the display/modify memory facilities. Inputting test data and changing boolean flags is done from the workstation keyboard and output messages are displayed as string variables.

4.4.2. Benchmarks and Microprocessor Selection

The next step, converting the simulation version of the software in an actual control program, will not be possible without the target hardware. The first step in the specification of this target hardware is to use the simulation software to select the most adequate microprocessor. Since there were no specific constraints concerning the choice of the microprocessor, a performance comparison study was done in order to assess which would best suit the WECS controller microcomputer.

The HP-64000 had full development facilities for Motorola and Zilog 8-bit and 16-bit microprocessors. Other processors were also supported, but they were not worth considering since no emulation facilities were available for them. Preference was given from start to Motorola family microprocessors because they had a broader range of available products and the compilers produced faster code. Moreover, when considering 16-bit microprocessors, the M68000 fares quite well when compared with its more direct competitors such as the Zilog Z8000 and the Intel I8086 [4.9.,4.11.-4.13.]. This is due to its consistent and orthogonal instruction set with instructions directed to the use of high-level languages and to its 7-level vectored priority interrupt structure.

4.4.2.1. 8-bit or 16-bit microprocessor ?

This was a vital comparison, since these are two options of quite different power and complexity, as well as price. The control program was first run with the system configured for emulation/analysis of the M68000, since this is capable of fastest execution and has a powerful interrupt structure and wide addressing space [4.9.-4.11.]. The extent of performance degradation when stepping down to an 8-bit microprocessor was then determined by emulating the M6809. It was then possible to check whether a less powerful but cheaper solution could be found.

4.4.2.2. Memory Requirements

The evaluation of memory requirements was desired only for guidance in the specification of the actual system. Load maps of emulation memory provided information on the program and data memory space required by the control program (Table 4.1). Actually they were found to be quite within an 8-bit microprocessor addressing capability. The execution time of critical program modules became thus the sole criteria for performance evaluation.

4.4.2.3. Execution time of critical modules. Benchmarks.

The system response time, i.e. the time interval from the time data is captured until processed information is delivered, can have different values depending on the system function considered. Here we are concerned with the time required for a complete cycle of the actual control algorithm, i.e. sampling transducer data, processing it and outputting actuator commands. Controller responsiveness criteria demand that this response time is kept within one order of magnitude smaller than the shortest system time constant. Although the controlled system time constants (Chapter 7) were not known at this stage it didn't seem unreasonable to assume values which are typical for similar WECS configurations [1.3.]. Fast execution is critical when the system is in the Optimal/Non-Optimal Operation State, since the arithmetics necessary in the control algorithm have to be executed fast enough to meet the existing real-time constraints. The base-module Optimumcontrol, which implements the control algorithm, had therefore to be timed.

Two versions of this module were designed and coded allowing the implementation of two different control strategies, and the more demanding one was used as a benchmark algorithm. Actually this was more than a benchmark or synthetic workload. As it effectively represented the typical operating conditions, it was rather a "live" workload. Table 4.2 presents the execution times of the module, a sequential program block, measured by the emulator/analyser system [4.1.,4.2.].

Measured wind speed Integer arithmetic				Measured or calculated wind speed Floating-point arithmetic			
M6809		M68000/8		M6809		M68000/8	
Prog.	Data	Prog.	Data	Prog.	Data	Prog.	Data
4.2 K	0.2 K	6.3 K	0.5 K	9.0 K	0.6 K	10.5 K	1.1 K

Table 4.1. Map of the control program memory requirements.

	Integer arithmetic		Floating-point arithmetic			
	Measured wind speed		Measured wind speed		Calcul. wind speed	
Micropr.	M6809	M68000	M6809	M68000	M6809	M68000
Time (ms)	5.39	0.50	154.85	8.40	910.92	46.21

Table 4.2. Execution time of base-level module "Optimumcontrol":
A 1 MHz clock M6809 and a 8 MHz clock M68000 were emulated to produce these results.

	Integer arithmetic			Floating-point arithmetic					
	Measured wind speed			Measured wind speed			Calcul. wind speed		
Micropr.	M6809	M68008	M68000	M6809	M68008	M68000	M6809	M68008	M68000
Time (ms)	12	5	3	400	50	30	1300	120	70

Table 4.3. Evaluated execution time for one complete control cycle in the Optimal or Non-Optimal Operation State. The values presented for the M68008 were directly obtained from the ones measured for the M68000 considering an average performance degradation of about 40%.

Table 4.3. gives the estimated execution time for one complete control cycle, taking into account the overheads caused by the pseudo-concurrent execution of the various modules.

Both control algorithms are based on the so-called lambda-optimum strategy for optimising wind energy extraction. The simpler method uses the wind speed value directly read from the anemometer, while the more complex one evaluates the wind speed actually "seen" by the rotor using the system torque equation (Fig. 3.12). In the latter case the use of look-up tables and more elaborate calculations significantly increases the computation time.

The performance of the two micros when implementing any of the control strategies could then be compared. A third intermediate option is included, the M68008, which is software compatible with the M68000. The values quoted for this microprocessor were estimated on the basis of manufacturer information. The execution time measured for the critical software module in its various options are conclusive (Table 4.3). As expected performance improves from the M6809 through the M68008 to the M68000. Since the M6809 was not able of meeting the above referred timing constraints it was withdrawn from the list of candidate microprocessors. Still giving an acceptable performance the M68008 would enable a cheaper final solution. However, no emulation facilities were available for this microprocessor and the range of available products was rather limited. A 68000-based system was therefore the "best" choice. The specification of the complete system is presented in section 5.5.

4.4.3. The Real-time Control Program

Converting the simulation version of the controller software into the actual on-line control program requires consideration of two aspects which are strongly dependent on the actual target hardware. These are the implementation of pseudo-concurrency and the low-level I/O programming, both key characteristics of microprocessors embedded in real-time control systems. Concurrency or pseudo-concurrency is what distinguishes real-time systems from other software systems. I/O programming involves a substantial part of the software handling

non-standard peripheral devices.

4.4.3.1. Implementation of real-time features. "kernel" design.

The development system supported neither concurrent programming nor any type of language primitive enabling some form of concurrency to be built. A specific solution to pseudo-concurrency appeared to have some advantages over a more general one, namely faster software development, simplicity and smaller processor overhead.

In the chosen approach a small "kernel" was designed to fill the intermediate layer between the naked hardware and the application software modules running on it. The "kernel" works like a very small monolithic monitor which schedules the CPU among the processes, receives I/O interrupts and transmits them to the respective I/O processes and supports interprocess communication. The topology of the actual system software and hardware enabled quite an amount of simplification in the design of the "kernel" [3.6.]. Referring to Fig. 4.2 its more significant features are now described.

4.4.3.1.1. Interrupt-driven I/O concurrency

The overall system performance is heavily dependent on I/O tasks and, exception made to the base-level processes, all the pseudo-concurrent processes are device-dependent. The tasks implemented by these processes are, most of the time, in real concurrent execution with the main task [4.17.] because the peripheral support chips used, while relieving the CPU from the tasks they execute, are actually running in parallel with it. Synchronisation and message-passing between the device-dependent and the base-level processes is enabled by the interrupt system. Foreground processing, i.e. interrupt- and trap-handling, is thus transparent to the background tasks such as the base-level Start, Stop, Optimumcontrol, etc.

4.4.3.1.2. Inter-process communication and synchronisation

Inter-process communication and synchronisation is therefore

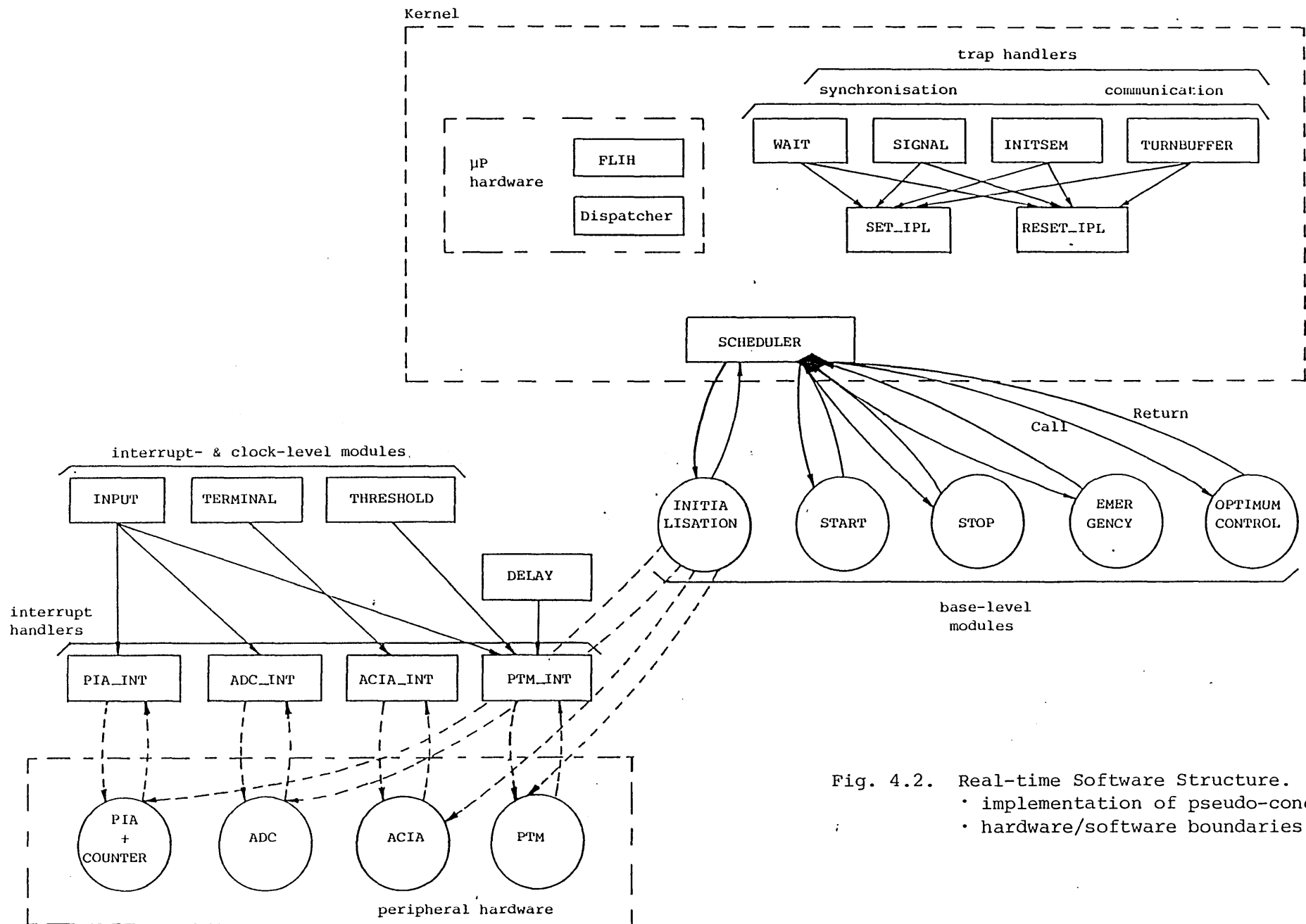


Fig. 4.2. Real-time Software Structure.

- implementation of pseudo-concurrency
- hardware/software boundaries

required for two pairs of processes : Input/Threshold and, e.g. in the Optimal Operation State, Input/Optimumcontrol. This can be simply but efficiently done by using double buffers for communication and semaphores for synchronisation between each pair of processes. This mechanism is briefly described below. Each set of sampled values of the system variables is placed, by the Input process, i.e. the ADC, PTM and PIA interrupt handlers, in one of the buffer sections (Fig. 4.3). When all the input data has been received the Input process will signal the Threshold and, e.g., the Optimumcontrol processes through the corresponding semaphores and then block itself. Whenever the Threshold or Optimumcontrol processes are ready to deal with a new set of data they will turn the buffer pointers and wake the Input process.

Semaphores are implemented by boolean flags that can be set and cleared inside a very small critical section executed with the interrupts disabled. The blocking and waking of device-dependent processes is done through hardware by restarting or not the ADC or the turbine speed pulse counter controlled through the PIA. Interrupt-driven, clock-driven and base-level processes can therefore run concurrently.

4.4.3.1.3. Scheduling

The scheduling of the various tasks obeys, at any moment, the actual relative priorities which are inherently tracked by the "kernel". High level scheduling is done in software by the Scheduler, which is the "kernel" central module. The Scheduler contains an in-built system model making it aware of all relevant internal and external events and therefore enabling it to decide which, among the base-level tasks, has the highest priority. Extra low-level scheduling is embedded in the vectored exception processing structure. Interrupts and traps are invisible both to the Scheduler and to the base-level modules. Since queues of processes are not necessary and all scheduling can be automatically dispatched by the exception processing hardware, the various process descriptors are limited to a minimum and are automatically swapped by the system.

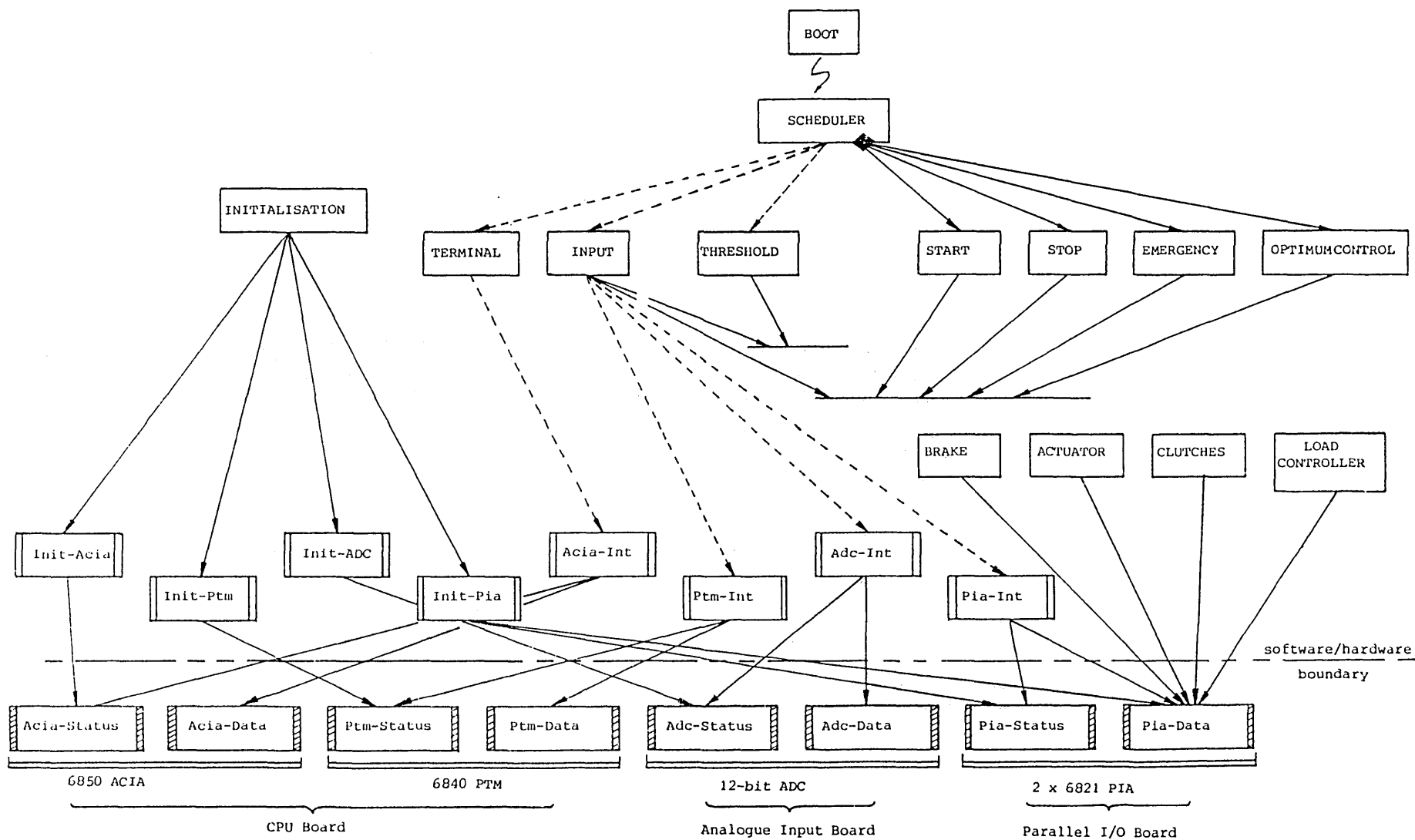


Fig. 4.3 - Low-level software modules.

4.4.3.2. Low-level hardware-dependent software

The implementation of pseudo-concurrency relies on the use of software modules actually dealing directly with the hardware (Fig. 4.2). Data I/O, synchronisation and generation of timing intervals are some of the tasks they implement. Some of these modules belong to the "kernel" software and others are just device-handlers, but all of them access directly the microcomputer hardware.

Since all the peripheral chips are memory-mapped, their data and control registers can be accessed with memory read/write operations, either from Pascal or Assembly [4.18.]. Specific utility procedures were written to allow each peripheral device to be directly accessed by any software module. No safety checks are made by these utilities in order to keep the I/O software fast. The calling module checks the context of the call. Both the low-level modules implementing the hardware/software boundaries and the higher level modules using them as utilities are shown in Fig. 4.3.

4.4.4. The Control Program as an On-Line Package

It was decided to replace all the hardware blocks performing operator I/O tasks (Fig. 3.1) with a terminal. The number and complexity of the tasks that could be implemented was enhanced while producing a much more efficient and neat design.

4.4.4.1. General features of the interactive software

The design of the interactive aspects of the control program evolved together with the design of the rest of the software. The interactive software was actually used, in parallel with the HP-64000 emulation monitor, to debug and integrate the control program. The interactive package is intimately linked with the Scheduler, allowing it to communicate with the operator.

Since the HP-64000 Pascal subset does not have the standard procedures read/write or the file handling procedures get/put/reset [4.5.] all the interactive software was written from scratch.

The general features of the interactive program are outlined in Appendix 4 . The blocks building the interactive package, presented in Fig. 4.4, are also discussed in Appendix 4 .

4.4.4.2. Performance constraints and other considerations

The impact caused by the inclusion of these interactive facilities in the on-line control program is now considered.

The positive aspects of an interactive module implementing all the described functions, while featuring a certain degree of user-friendliness, versatility and error immunity, are obvious.

The drawbacks are the limitations it imposes on the whole control program in terms of memory space and speed of execution. Although the interactive program modules occupy an important percentage of the EPROM space, due to the massive storage of message strings, there is still a comfortable margin for memory expansion (see Chapter 6). Disturbing or limiting the speed of execution of the vital control modules is however a real problem.

To minimise the overload caused by the Terminal process (Fig. 4.4), several mechanisms were included in the design. First the ACIA dependent process, Terminal, was given the lowest priority set up in hardware. Since the base-level processes can still be disturbed, critical program sections are executed with the ACIA interrupts disabled. For optimum response to external events the interaction needed while the system is running is reduced to a minimum. Information enquiries, setpoint/compensator parameter changes and display mode selection are confined to the Initialisation module executed in the Stand-by state. Finally, high priority operator commands can be selectively input to the Scheduler thus bypassing the Terminal module. This is the case with the stop/panic commands entered while the system is running. The controller response time is therefore affected solely by the screen updating functions, since the ACIA is a relatively slow peripheral. For low values of the sampling time a "fast" variable display mode should be selected (see Chapter 6). The use of a memory-

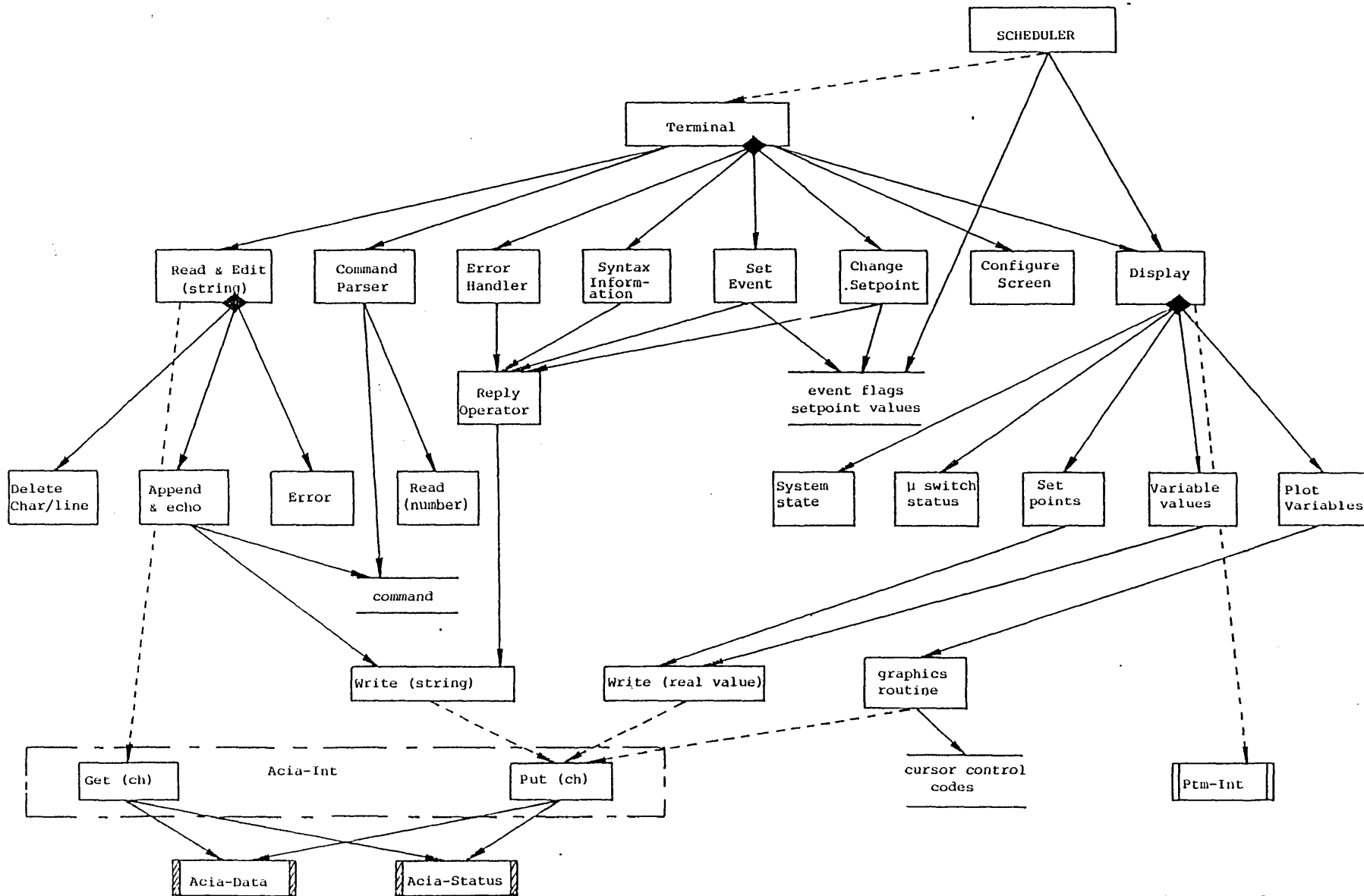


Fig. 4.4. Interactive Package Building Blocks.

mapped graphics display, besides allowing a much better resolution, would speed-up this part of the software enormously.

4.4.4.3. Running the control program

The interactive software aims to give the operator a model of how the system works. Being close to the Scheduler it will follow the system transitions from one operating mode to another (Fig. 3.2) displaying relevant information and prompting the operator when necessary. Actually going through the steps of running the program is almost self-teaching. Therefore only a few hints on its structure, which is illustrated in Figs. 3.2 and 4.4, are presented.

Following power-on/reset the screen is initialised and the operator asked to do some system initialisation (Stand-by state), as in Fig. 4.6. The system always prompts when waiting for a command. The operator may then edit the desired command string and enter it. Editing errors and invalid commands are reported and ignored. An information page, with the syntax for all commands (Fig. 4.5), and an error page are available. Selection of the system sampling time, compensator gains, display mode and control policy is compulsory, although a command is available to adopt default values. Setpoint initialisation is optional and, when ignored, default values are assumed.

Upon entering the start command the system will perform routine hardware checks (Pre-start Checks). A dialogue between the system and the operator may occur here. If the wind speed is too high or if any malfunction is detected a warning/diagnosis message is displayed and starting is aborted. If those checks are successfully completed the operator is prompted for the choice of the run-up mode. An adequate reply will trigger the turbine run-up procedure (Start-up) either in normal or fast mode. From now on system variables are displayed/plotted with their values being updated each second, and operator interaction is reduced to a minimum (Figs. 4.7 and 4.8).

Eventually the turbine will speed-up and the system will change state (Optimal/Non-optimal Operation), being then loaded

```

SYNTAX FOR OPERATOR COMMANDS
=====
<RETURN>  C SYSTEM ATTENTION / ENTER EDITED COMMAND 1
<1>       C INFORMATION ON OPERATOR COMMANDS SYNTAX
<B/S/P>   C START / STOP / PANIC 1 BUTTON
<O/A>     C NORMAL / FAST 1 START PROCEDURE OPTION
<Y/N>     C YES / NO 1 REPLY (TO SYSTEM MESSAGE "PROCEEDING START ?")
<R>       C START / STOP 1 ANALOG TAPE RECORDER
<L/H/F>   C SELECT C LAMBDA OPT 1 / HILL CLIMB / FEEDB POWER 1 CONTROL STRAT
<OF>      C DEFAULT SYSTEM INITIALISATION
<DV>      C DISPLAY SYSTEM VARIABLES
<DS1/DS2/DS3> C DISPLAY SETPOINTS GROUP C 1/2/3 1
<DP1/DP2/DP3> C PLOT VARIABLES OF GROUP C 1/2/3 1

<CS ____>  C CHANGE TURBINE SPEED LIMIT (TO NEW INTEGER VALUE ____ )
<CT/CB/CP ____> C CHANGE C TORQUE / BLADE STRAIN LIMIT / BLADE PITCH SETPOINT
<CH/CC/CU ____> C CHANGE C WT LOADING SPEED / CURRENT / VOLTAGE 1 LIMIT
<CK/CO/CR ____> C CHANGE C BRAKING SPEED / (PITCHING/FAST PITCHING) RATE 100
<TT ____>  C CHANGE SAMPLING TIME (TO NEW INTEGER VALUE ____ MS)
<TP ____>  C CHANGE PROP GAIN (INPUT INTEGER VALUE ____=KP*1000)
<TI ____>  C CHANGE INT GAIN (____=KI*1000)
<TD ____>  C CHANGE DER GAIN (____=KD*10000)
<TE ____>  C CHANGE STEP FACTOR (TO NEW INTEGER VALUE ____ 2)

```

Fig. 4.5 - Information page

```

SYSTEM STATE: STAND-BY
MICROSWITCHES ALFA OFF
                BETA ON
VIDSWITCH OFF GAMA ON

SYSTEM MESSAGE: INITIALISE SYSTEM (TTTT)
SETPOINTS + COMPENSATOR OR "DE"
SETPOINTS  TURBINE SPEED  800.0  RPM
            TURBINE TORQUE  35.00  MM
            BLADE STRAIN   22.00  US
            BLADE PITCH    2.500  DEG

            T(s)    KP    KI    KD(*1000)    E(?)
            47.88   1.000  1.000  1.000      1.000

PENDING OPERATOR COMMAND:

```

Fig. 4.6 - System initialisation (Stand-by state)

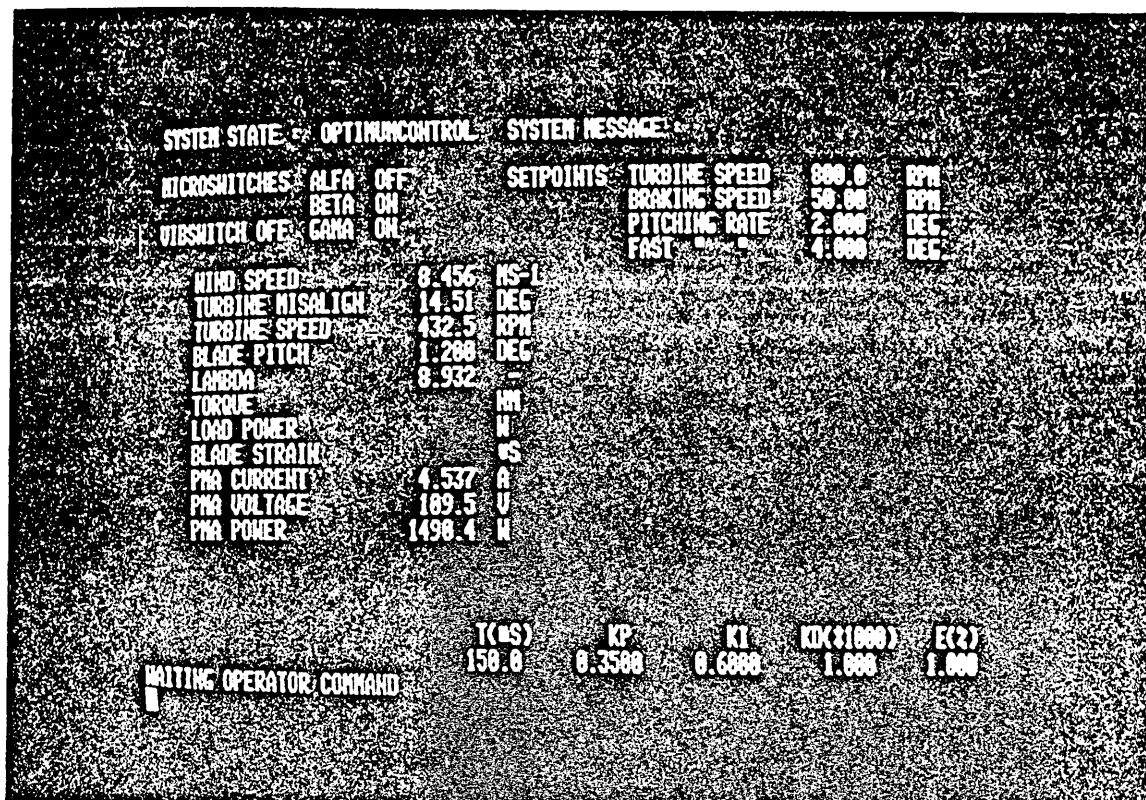


Fig. 4.7 - Variable Display Mode (Optimumcontrol)

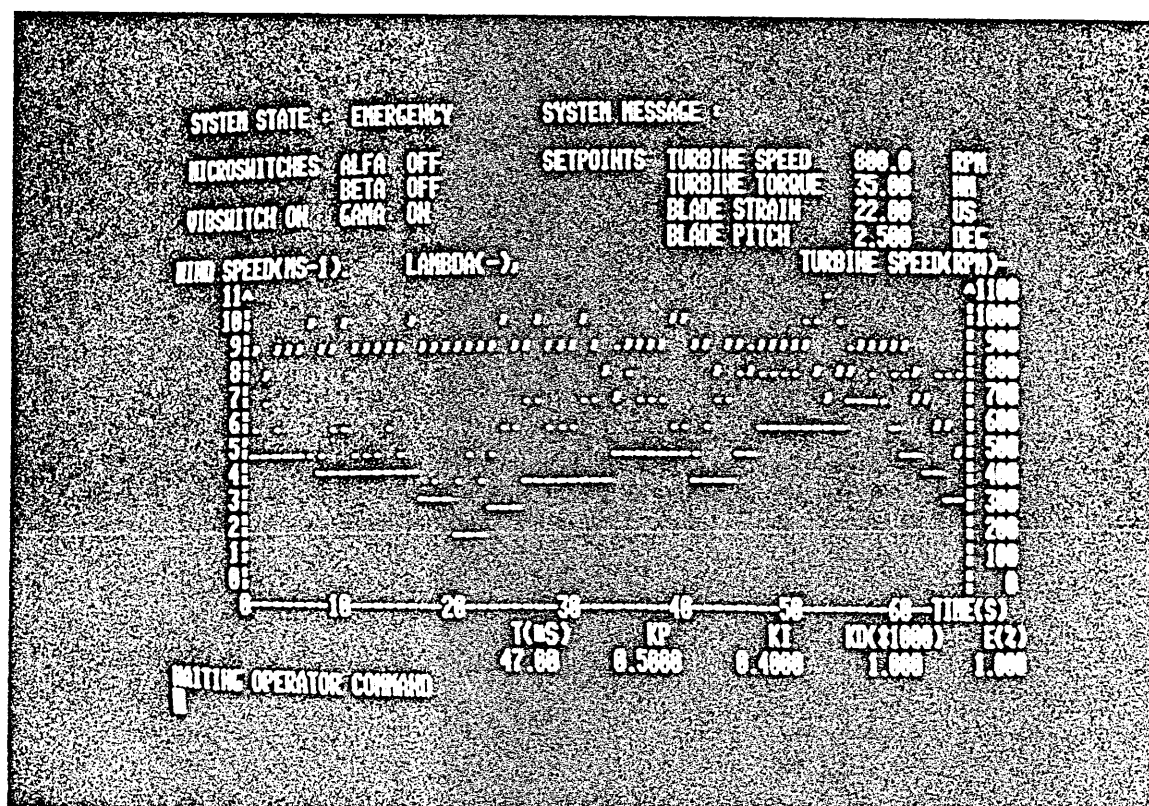


Fig. 4.8 - Variable Plot Mode (Emergency Stop)

according to the chosen control strategy. While in this operation mode the system can change state on its own (Emergency Stop) or upon operator request (Stop or Emergency Stop). Actually, the commands to trigger the Stop or the Emergency Stop procedures are the only allowed operator commands while the system is in the Start-up or Optimal/Non-optimal Operation States. The Stop procedure is also interrupted and Emergency Stop branched whenever the system detects an emergency condition or upon operator command.

On termination of the Stop procedure (Stand-by State) the system can be restarted, after reinitialising vital parameters, or finished by the operator. On termination of the Emergency Stop procedure (Idle State) the operator is warned that the system is finished and that boot-up has to be done through the reset button. In this case an hardware reset signal is sent to all the peripheral interface IC's, allowing a total system reinitialisation.

4.4.5. Implementation and Testing

Integration of all the software modules and complete debugging of the control software was then done. As the software development could no longer be isolated from the supporting hardware, the target system was connected to the LDS and ICE provided extensive debugging facilities. Final integration of all the software modules, debugging the real-time features and testing software/hardware boundaries had to be done simultaneously. These matters are dealt with in Chapter 6.

CHAPTER 5

HARDWARE DEVELOPMENT

5.1. INTRODUCTION

Although not as apparent as with the software design, most of the hardware design was still strongly based on the system model developed in Chapter 3.

The first stage of hardware development involves the characterisation of all hardware subsystems. This covers "process", sensing and actuation subsystems, from the plant end, and the micro-computer with operator interaction and interfacing subsystems from the controller end. This specification then leads to an actual hardware design. Detailed information on all the physical constraints is then of major importance.

Interface complexity could be a major problem and is due to the need of interfacing the microcomputer controller to non-standard hardware [5.10.-5.12.]. The design of an effective interface requires combined expertise on software and logic design on one hand, and on transducing and actuation techniques on the other. In fact, although the implementation is in hardware, each interface has to be served by software.

Finally, trade-offs between specifying and buying commercially available components or designing and building them have to be considered.

5.2. Plant Hardware Details

5.2.1. Wecs hardware

The actual WECS hardware is composed of all the elements actually taking part in the energy conversion process, i.e. wind turbine, p.m. alternator and electrical load. These were briefly described in section 2.1., and therefore only relevant complementary information is added here

The wind turbine is a 2-bladed horizontal-axis, downwind machine. The blades have a counterweight and are attached to the hub

through an universal joint. Such a design allows the blades to cone freely within a certain range partially relieving them from the bending stresses. Active blade pitch control is included but there is no yaw control. Self-alignment is achieved through the help of a small fin. The design tip-speed-ratio is 9.3 for a turbine speed of 700 rpm and a wind speed of 12 m/s. The speed operating range under normal conditions will be 200 to 800 rpm. Cut-in and maximum wind speed values will be accurately determined from field tests.

The p.m. alternator is a specially designed 3-phase, 12-pole, 8kW machine featuring high efficiency both at full and part loads [5.1.]. For the 3.5 m diameter blades the generator is directly coupled to the turbine and the power output at 700 rpm is expected to be 2kW. The planned 9 m diameter rotor will drive the alternator through a gear-box. The electrical load is a 3-phase stepped resistive load.

5.2.2. Control system hardware

The control system hardware will complement the plant hardware with three groups of elements : transducers/actuators, interfacing circuitry and microcomputer system (Fig. 3.1). General considerations on the design approach are laid out below. Detailed specification/design is presented in sections 5.3. through 5.5.

5.2.2.1. Types of control actions

Different types of control actions will have to be implemented to satisfy the initial specifications (2.1.). The Start, Stop and Emergency Stop procedures demand a sequence control structure implementing closed-loop speed control. The Optimumcontrol procedure implements two feedback loops. One aims to optimise the wind energy extraction by loading the turbine adequately, and the other is a protection loop pitching the blades to spill the excess wind power input. System monitoring and operator interaction functions will have to be integrated with these control functions.

A number of factors have to be taken into account when selecting transducers and actuators. Siting is one of the most important of these factors since both transducers and actuators are physically connected to the WECS and have to cope with the plant environment. Other important factors such as cost, ruggedness, range, accuracy, linearity, speed of response, remote calibration facilities, etc., [5.2.-5.6.] guided the selection of the actual hardware.

5.2.2.3. Closing the control loops

The interfacing blocks are of different types. The components electrically linked to the transducers perform isolation, signal conditioning and data conversion. They allow the transducer output to be interfaced to an analog or digital input module with certain electrical specifications. The elements connected to the actuators have to produce high-power driving signals by amplifying the control signals they receive from the digital I/O module.

The microcomputer system is the element closing all the existing control, monitoring and interaction loops (Fig. 2.3.). It will include peripheral interfacing and data conversion support IC's.

5.3. TRANSDUCERS AND SIGNAL CONDITIONING SPECIFICATION/DESIGN

The actual implementation of the data acquisition, transmission and conditioning blocks of each subsystem is now described. It is assumed that peripheral interfacing and data conversion support, e.g. analog-digital conversion, serial and parallel data interface IC's, ..., will be included from the microcomputer end (5.5.).

5.3.1. Wind Turbine Speed

Digital angular velocity transducers employing optical techniques are very popular. In fact, optical shaft encoders have high reliability, maintainability and life. This justifies the preference given to a digital measurement of the turbine speed. Since the turbine speed has to be measured over a wide range (0 to 1000 rpm) a

pulse-counting technique is to be preferred to frequency counting. Accuracy increases with the measured speed and with the number of light beam interruptions per revolution. Minimum accuracy can be ensured by adequately choosing this number.

A shaft-mounted aluminium disc with 360 holes in the periphery and an infrared slotted optical switch (Fig.5.1 a) generate pulses which are counted for a given averaging time. A 12-bit counter IC, controlled by the microcomputer, allows it to input a digital word representing the shaft speed (Fig. 5.1 b). For a sampling time of, e.g., 35 mS (See Chapter 7) the accuracy of the measurement will be 2.38% at 200 rpm and 0.48% at 1000 rpm. The resolution achieved is 1/6 of rpm.

5.3.2. Shaft Torque and Blade Strain

In the first WECS configuration these two variables are only sampled for monitoring and data logging purposes. The resistance-bridge torque transducer signal will be fed into a high performance signal conditioning block, i.e. instrumentation amplifier and extra circuitry, situated close to the transducer. The full bridge strain-gauges output signals will be conditioned by instrumentation amplifier blocks mounted as near the gauges as possible. The high-level analog signals are then transmitted through a twisted screened pair to the analog input block.

A preliminary design of these analog subsystems was done. The decision on the actual hardware solution was however postponed to a further stage due to the lower priority of these subsystems. The modular software and hardware design allows for easy inclusion of these input channels at a later stage.

5.3.3. Tower Vibration

Vibration sensing hardware is usually associated with sophisticated and expensive acceleration transducers [5.4.]. However since monitoring tower vibration is done with the sole purpose of

detecting an emergency situation, there is actually no need for vibration measurement. In fact a conveniently tuned on-off sensor can do the job quite satisfactorily. A small inertia switch, mounted on the tower, provides a digital signal representing its status. If tower vibration causes its closure, the hardware-debounced switch transition (Fig. 5.2 a) will interrupt the microprocessor thereby triggering an emergency stop.

5.3.4. Pitch Mechanism Status

On-off sensing is necessary to provide information on the blade pitch mechanism status. The system has to know at any moment whether or not the clutches are holding the actuator in its operating position and also whether the limit values of the actuator stroke, which correspond to blade pitch fully fine/coarse, have or not been attained.

A simple and cheap solution using heavy-duty, roller-type micro-switches (Fig. 5.3) was found to be adequate enough. The necessary operating force is no obstacle here, neither the relatively slow response of the mechanical switches. The switch integrity is defended by limiting the movement of the actuating element by means of adjustable stops. The digital signals are hardware-debounced and both switch closing and opening transitions are sensed by PIA input lines (Fig. 5.2 b). The drawbacks of a solution employing mechanical switches could therefore be eliminated.

5.3.5. Blade Pitch

Blade pitch angle is indirectly sensed through the displacement of the links that transmit the actuator movement. A linear potentiometer was the transducer chosen because it provides a simple, inexpensive and good performance solution. The hybrid track (conductive plastic) potentiometer is mounted on one of the pitch mechanism links (Fig. 5.3) and gives an high-level analog signal indicating the actuator position. The performance data claimed by the manufacturer is capable of meeting rather demanding specifications, e.g.:

- resolution : virtually infinite

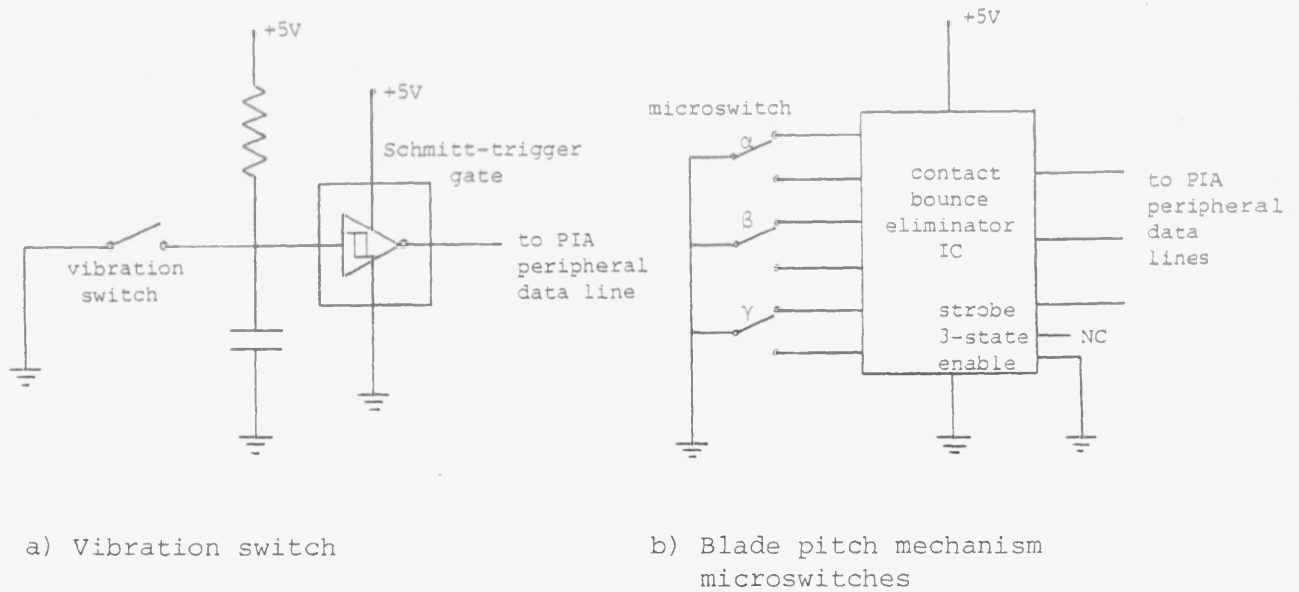


Fig. 5.2 - Hardware switch debouncing

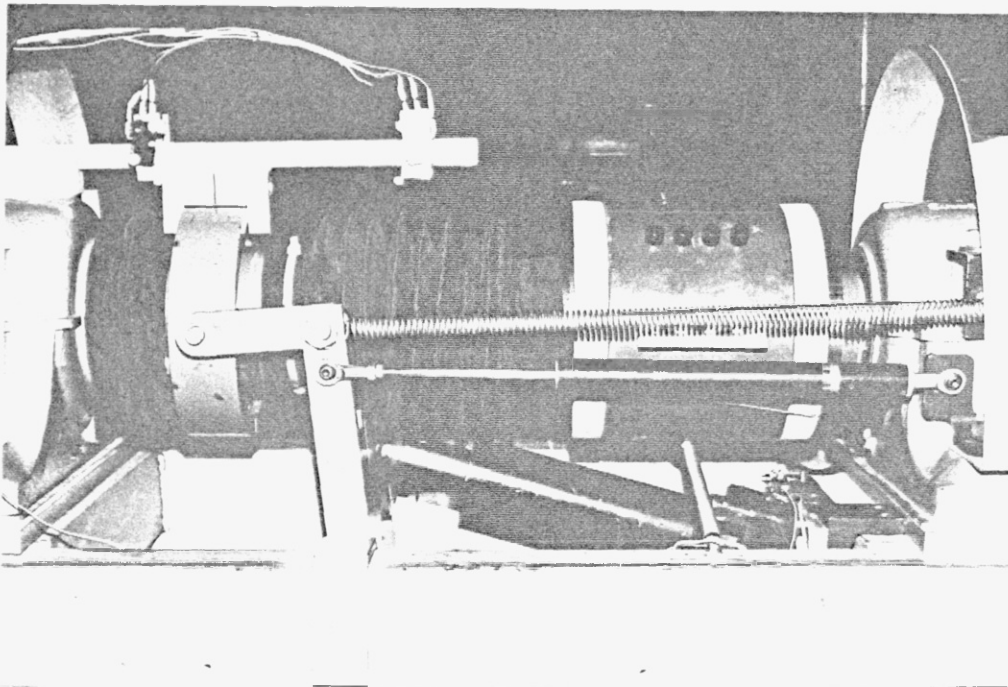


Fig. 5.3 - Blade Pitch Measurement Hardware

- temp. coeff. : ± 125 ppm per $^{\circ}\text{C}$ (max)
- life at 250 mm/sec stroke : typ. 50×10^6 cycles at 25 mm stroke.

Loading effects are minimal since the analog input block presents an input impedance much higher than the potentiometer wiper resistance ($4\text{k}\Omega$). Since the actuator has limited stroke, the problem of the limited potentiometer travel is overcome simply by adequate mounting.

The output signal is then transmitted through a twisted screened pair to the analog input block. The microcomputer then converts this linear displacement into a blade pitch angle through a "look-up" table or a conversion formula.

5.3.6. Wind Speed and Wind Turbine Misalignment

The requirement of a medium/high response anemometer led to the choice of a device based on an optical pulse counting technique (see 5.3.1.). A high resolution output anemometer, offering remote calibration facilities and choice between an analogue or digital output was chosen. In spite of its worse noise immunity the analogue output was preferred due to the simpler interfacing to the microcomputer. One multiplexed analogue channel would only be required plus an averaging filter. Other anemometer specifications are the following:

- $V_{\text{thr.}} = 0.15 \text{ m.s}^{-1}$; $V_{\text{max.}} = 25 \text{ m.s}^{-1}$; $V_{\text{storm}} = 75 \text{ m.s}^{-1}$
- accuracy and linearity within $2\% \pm 0.1 \text{ m.s}^{-1}$
- ratemeter lag with a 0.15 sec. time constant (max)

Wind turbine misalignment is measured for data logging and analysis purposes. Digital and analog type windvanes were considered but the first type was dismissed since the commercially available products offered a much lower accuracy. The chosen device is a wire-wound potentiometer type windvane featuring high resolution output (0.3°), measurement over a 360° angle and remote calibration facilities. Other specifications are:

- $V_{\text{thr.}} = 0.6 \text{ m.s}^{-1}$; $V_{\text{storm}} = 75 \text{ m.s}^{-1}$; accuracy : $\pm 1\%$; linearity : 0.5%

Potentiometer loading is kept within acceptable values since the input impedance of the analog input block is much higher than the $1\text{k}\Omega$ potentiometer resistance. The anemometer and the windvane are mounted on the wind turbine boom upwind of the rotor. The output signals are transmitted through a twisted screened pair to the analog input block.

5.3.7. P.M. Alternator Current and P.M. Alternator Voltage

The p.m. alternator current and voltage measurements require some form of isolation between the high voltage/current levels and the microcomputer. Current is measured through a Hall detector based current sensor which provides the necessary galvanic isolation (Fig. 5.4). A similar type sensor is used for measuring voltage.

Since rms values of the current/voltage are required for control purposes the output voltage signal obtained needs further conditioning. Monolithic rms/dc converter IC's generate a d.c. equivalent of the measured voltages. The averaging time is made equal to the system sampling time. A 2nd order Butterworth output filter reduces the ripple to acceptable values. The resulting high-level analogue output signals are both fed into the ADC block.

5.4. ACTUATORS AND POWER DRIVERS SPECIFICATION/DESIGN

5.4.1. Blade Pitch Actuator

The central component of the blade pitch control mechanism is an electromechanical linear actuator. Extending/retracting the actuator arm moves the linkages that pitch the blades (Figs. 2.2 and 5.5). The p.m. d.c. motor used in the actuator requires 12 V dc supply with a current demand up to 25 A. The extension/retraction rate of the actuator arm is load dependent and consequently the range of expected values is of difficult estimation, mainly because the same actuation system will be used for the 9 m diameter rotor. The driver was therefore designed for the maximum actuator rating.

The actuator power driver must then be capable of supplying 25 A, 12 V d.c., for a minimum specified time interval. Such a time interval should allow an emergency stop in the most unfavourable conditions. Direct and reverse polarity supply of the d.c. motor are necessary to allow both extension and retraction of the actuator arm. The driver circuit should still provide a path for the inductive current flow occurring at each "switch-off" operation, and be inhe-

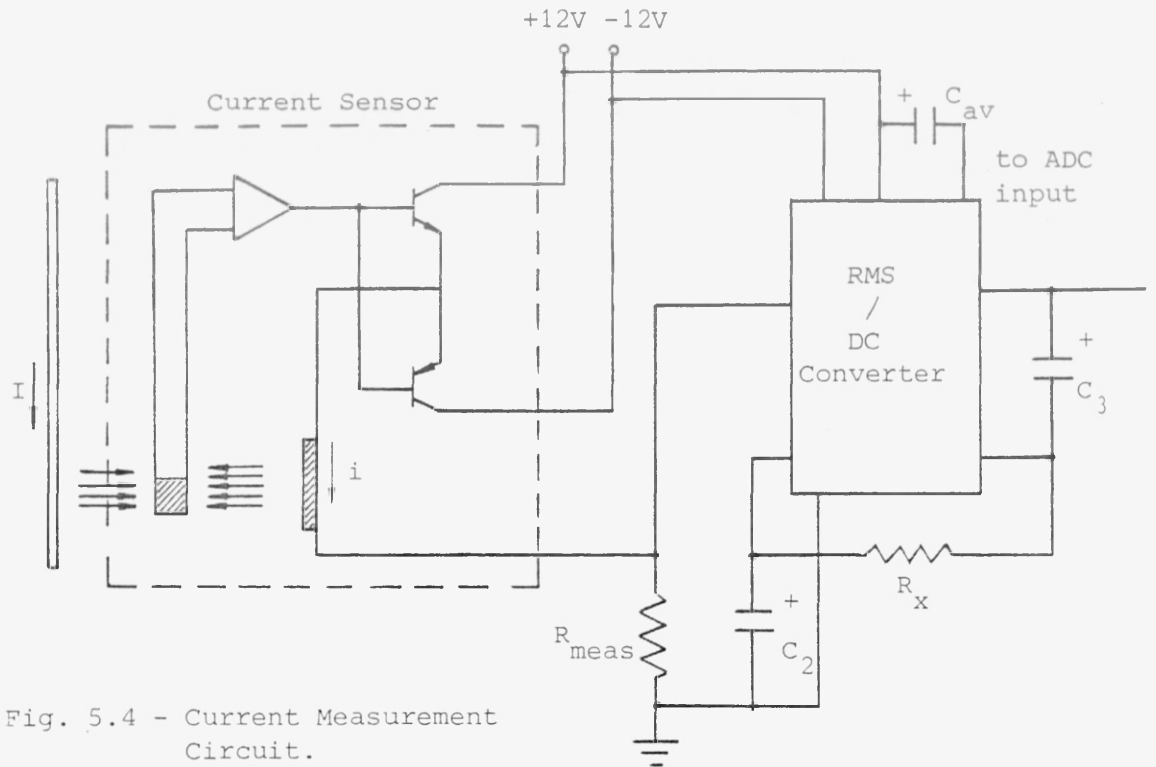


Fig. 5.4 - Current Measurement Circuit.

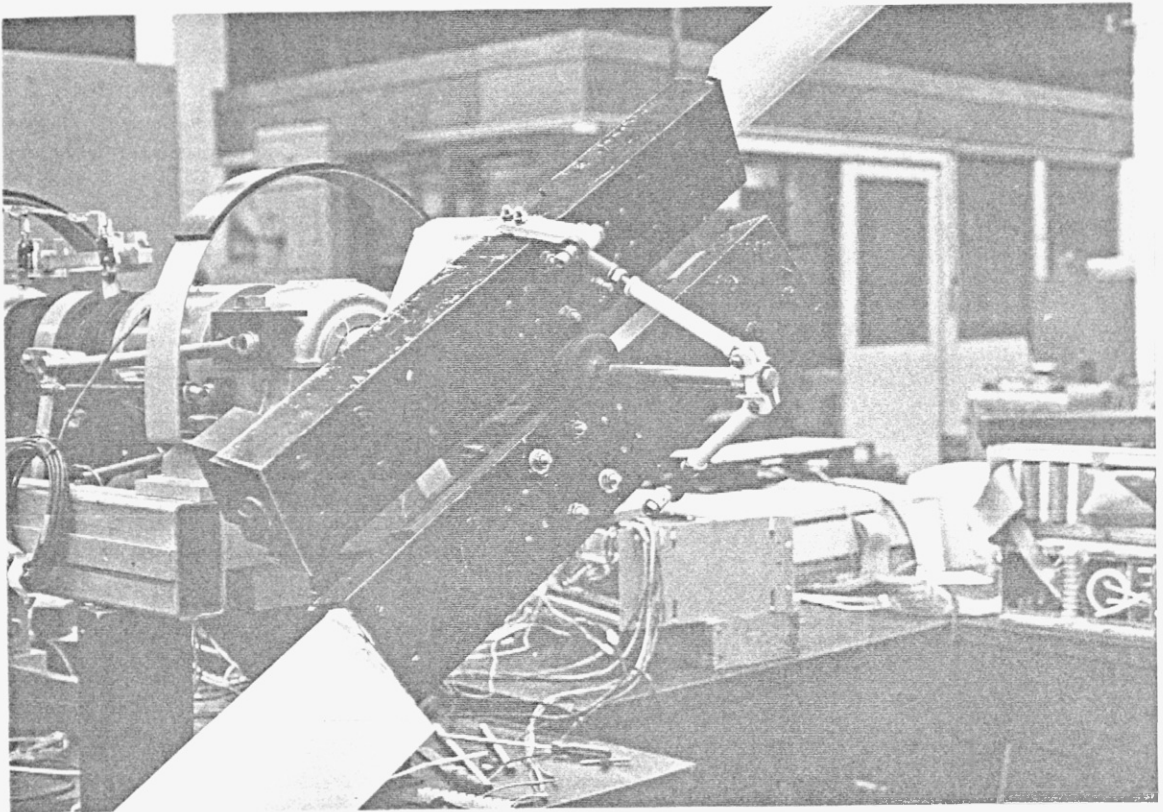


Fig. 5.5 - Blade Pitch Actuation Linkages.

rently safe by not allowing both forward and reverse switches to be operated simultaneously.

A number of designs could satisfy these requirements. They range from relay circuits switching a d.c. power supply to thyristor converters or power mosfet choppers [5.7.-5.9.]. A relay-based design has the advantages of low cost and simplicity but also the inherent limitations of mechanical switches, like slow switching action and limited life. Power mosfet switches enable a neat and versatile solution that could be easily interfaced to the microcomputer. Driving circuitry would however have to be designed and tested. Finally a thyristor converter would be the most complex, expensive and time consuming solution due to the necessary buffering and driving circuitry. The functional considerations guiding the selection of the actuator driver design are discussed below.

Many WECS configurations, e.g. for fixed-speed grid-connected systems, require continuously varying blade pitch for control purposes. That will be the case with one of the configurations to be implemented in the test-bed : an induction generator will be coupled to the wind turbine through a gear-box in a mains-synchronised operation. Fast response and high rate of switching operations will be then necessary calling for a solution using solid state electronic switches. For the present variable-speed configuration blade pitch control is not used in a main control loop but solely in a secondary protection function. The number of switching operations is therefore limited and the speed of response of the driver circuit less critical. A relay-based solution can therefore meet all the required specifications.

A d.c. power supply capable of continuously sourcing 25 A, 12 V would be an overkill since medium/maximum actuator loading will only occur for short time intervals. On the other hand a conveniently sized battery, fed from a mains-connected trickle-charger, provides an inexpensive d.c. supply capable of sourcing high currents for short periods of time. Preference has therefore been given to this solution.

A switching circuit capable of feeding the actuator motor with a d.c. supply of both direct and reverse polarity is shown in

Fig. 5.6. The circuit uses two heavy-duty double-pole double-throw relays capable of switching the maximum current. The relevant feature of this circuit is that it inherently provides paths for the inductive current flow upon "switch-off" and prevents simultaneous operation of both switches. The specifications are thus met with minimum hardware complexity. The mains-supplied relay coils are switched by solid-state relays (SSR's) which also provide optical isolation between the microcomputer and the mains. A darlington driver is used to interface the digital output line to the SSR's inputs.

5.4.2. Electromagnetic Clutches

Each electromagnetic clutch requires 0.25 A from a 24 V d.c. supply. This calls for a 24 V, mains fed d.c. power supply capable of supplying 0.5 A with an acceptable ripple level. Fig. 5.7. shows the selected solution. The clutches will be controlled through a buffered digital output line. This controls the energisation of the coil of the reed relay that connects/disconnects the supply to the clutches.

5.4.3. Fail-Safe Brake

It is the principle of operation of the brake that makes it fail-safe. It is a spring applied disc brake, which is released by a set of three-phase electromagnets fed from the mains. The peak energising current is 1.8 A and the continuous seal current 0.2 A. The brake is energised/deenergised by switching a set of SSR's controlled by just one digital output line (Fig. 5.8). The optical isolation provided and the possibility of direct interfacing with a buffered digital output line are the two features favouring the use of SSR's.

5.4.4. Load Controller

Electrical torque control is done through a 3-phase switchable resistive load. The load control scheme is quite straightforward : the a.c. load resistors are switched by SSR's also controlled

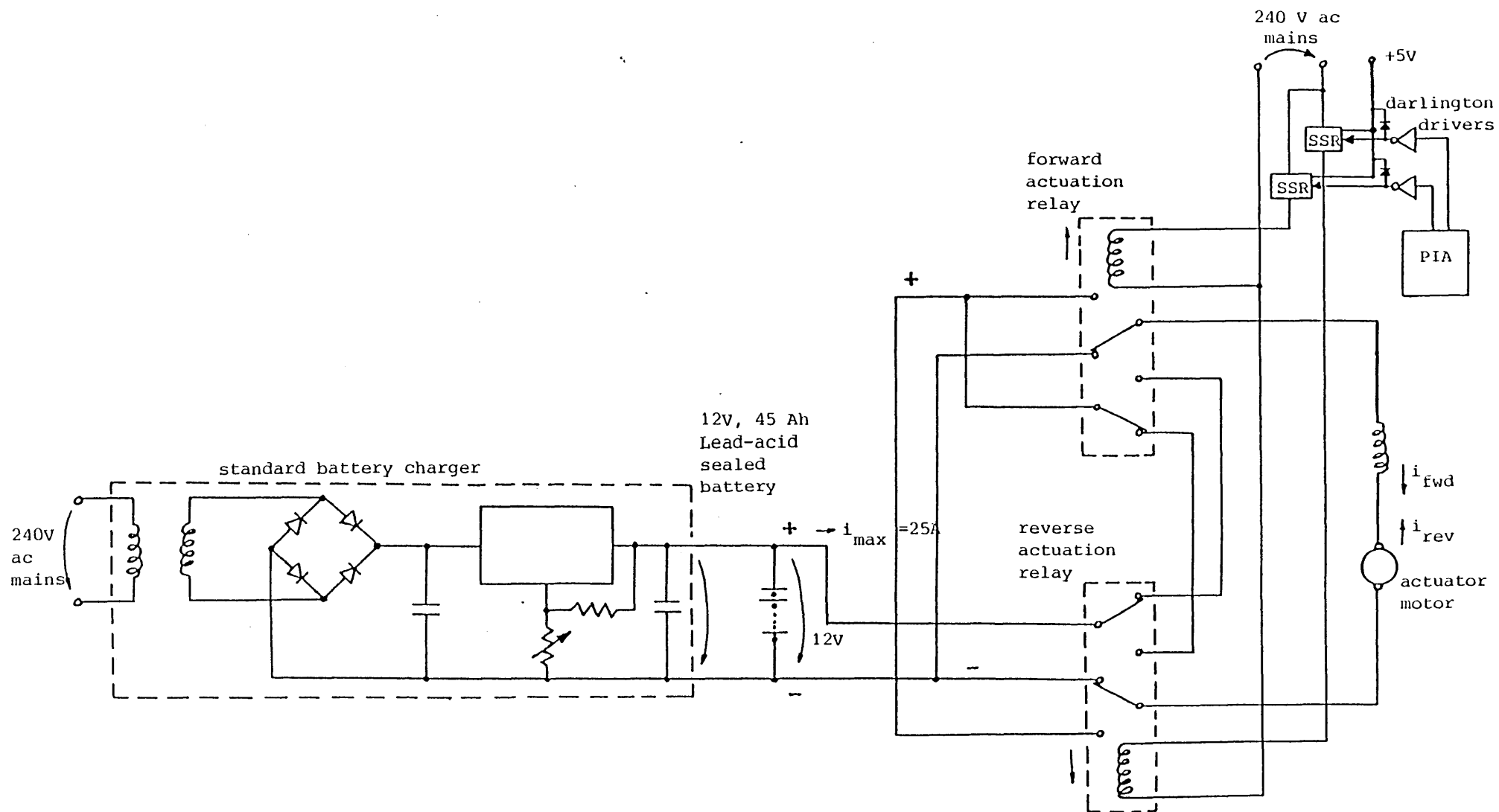


Fig. 5.6 - Actuator Power Driver.

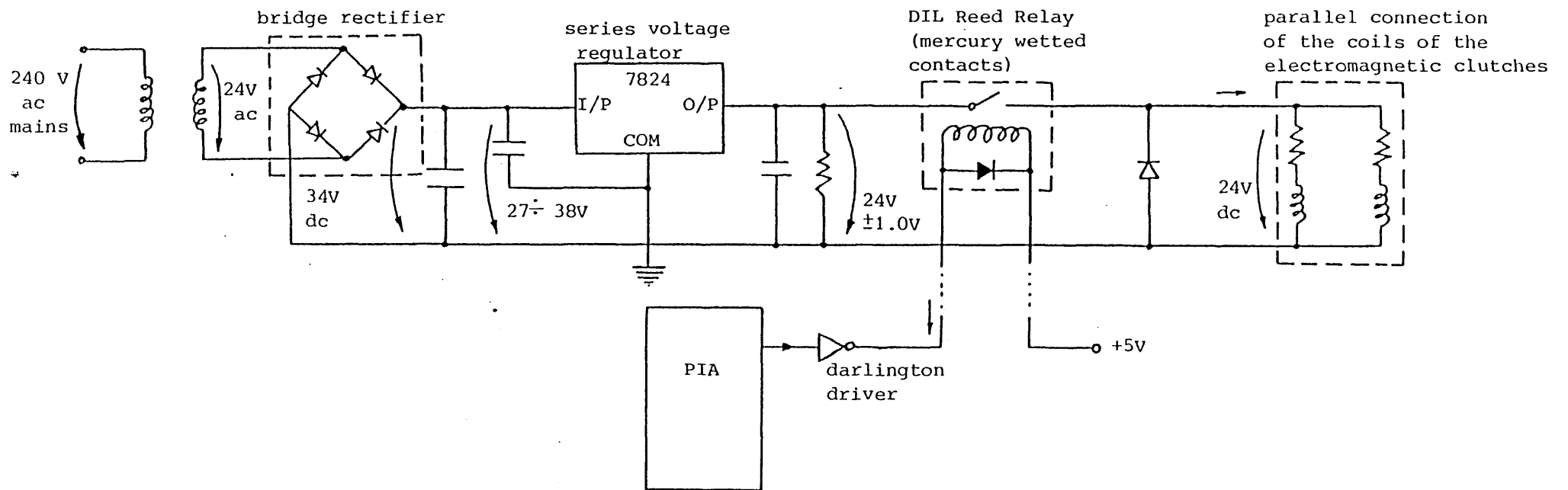


Fig. 5.7 - Clutches Power Driver.

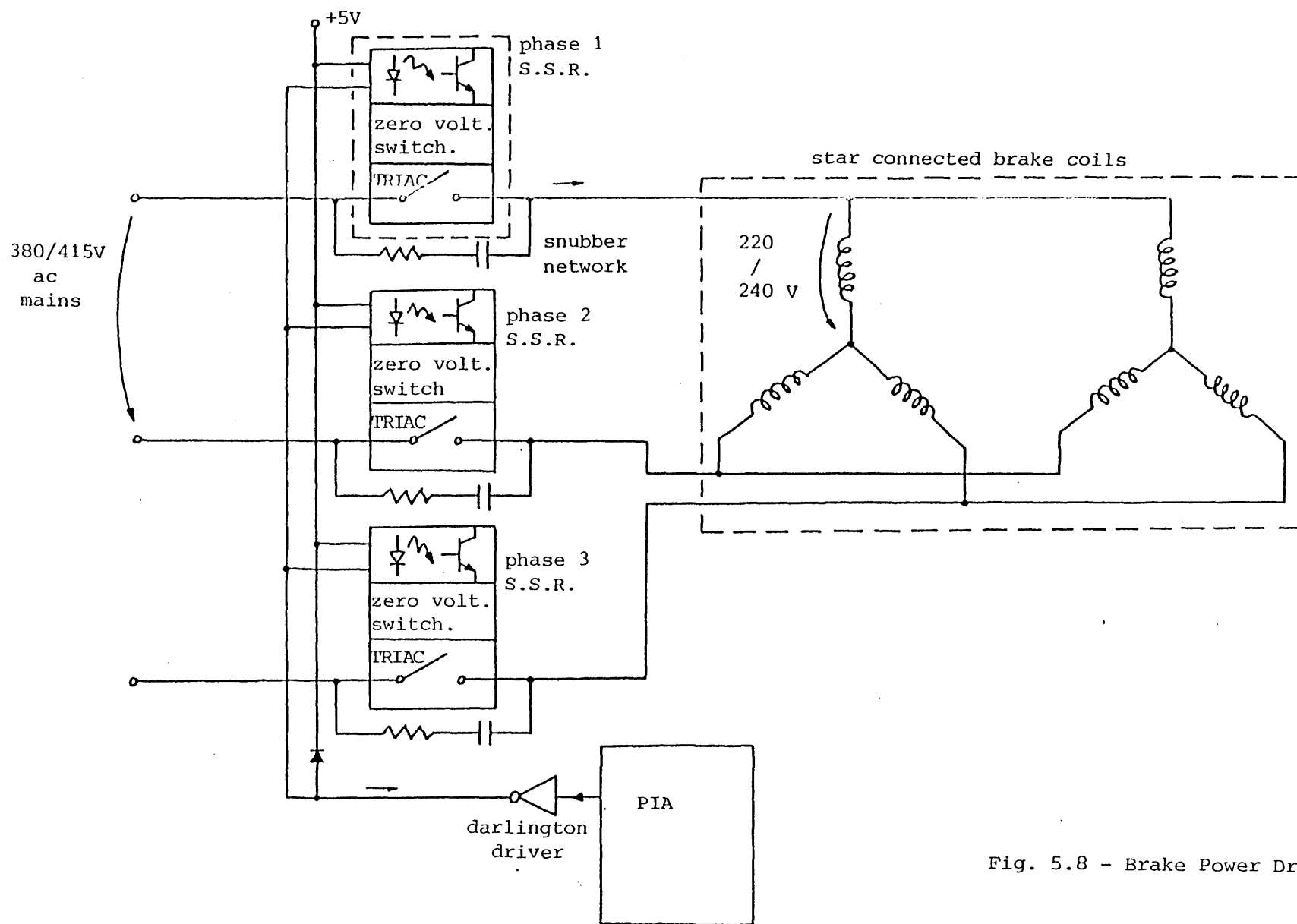


Fig. 5.8 - Brake Power Driver.

from buffered digital output lines (Fig. 5.9.). Each output line controls three SSR's switching a set of resistors. The six sets of three-phase resistors build up a 64-steps digitally controllable load. The optimum quadratic loading function is replaced with a piecewise approximation (Chap. 7).

For the variable-speed WECS, the p.m. alternator voltage and power depend on both driving speed and applied load. In the design of the resistive load a number of different problems will have therefore to be taken into account. First the wind turbine speed operating range and the corresponding power and torque outputs have to be known. The same applies to the p.m. alternator power for different speed and load values. The number of load steps is then defined by the required precision of tracking the maximum power curve (Fig.7.10). The design and implementation of the switchable load is presented in Appendix 5 . Since it was required that the same resistive load could be used with both the 3.5 m and the 9 m diameter rotors driving the p.m. alternator, a compromise had to be achieved in the load design.

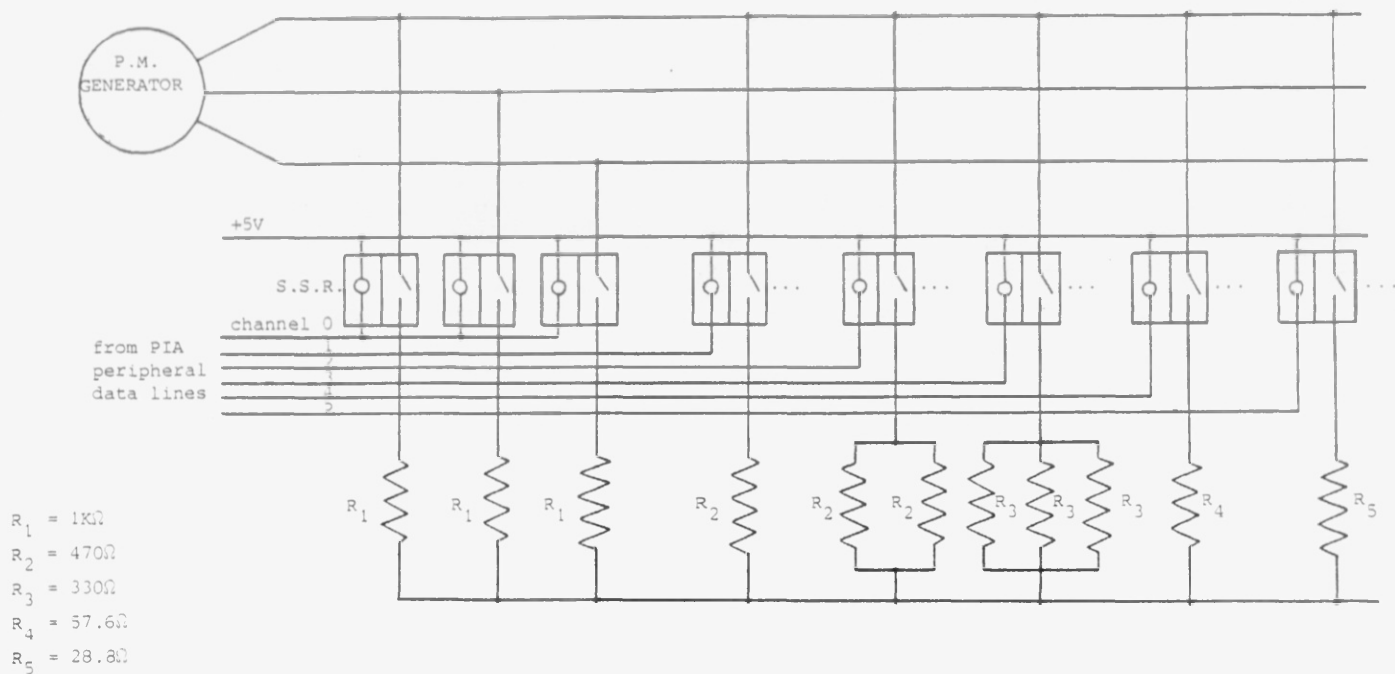
5.5. MICROCOMPUTER AND PERIPHERAL INTERFACING SPECIFICATION

In Chapter 4, the M68000 was shown to be the microprocessor that would best satisfy the requirements placed by the controller specification. Using the information available at this advanced stage of hardware design, the complete microcomputer system can now be specified.

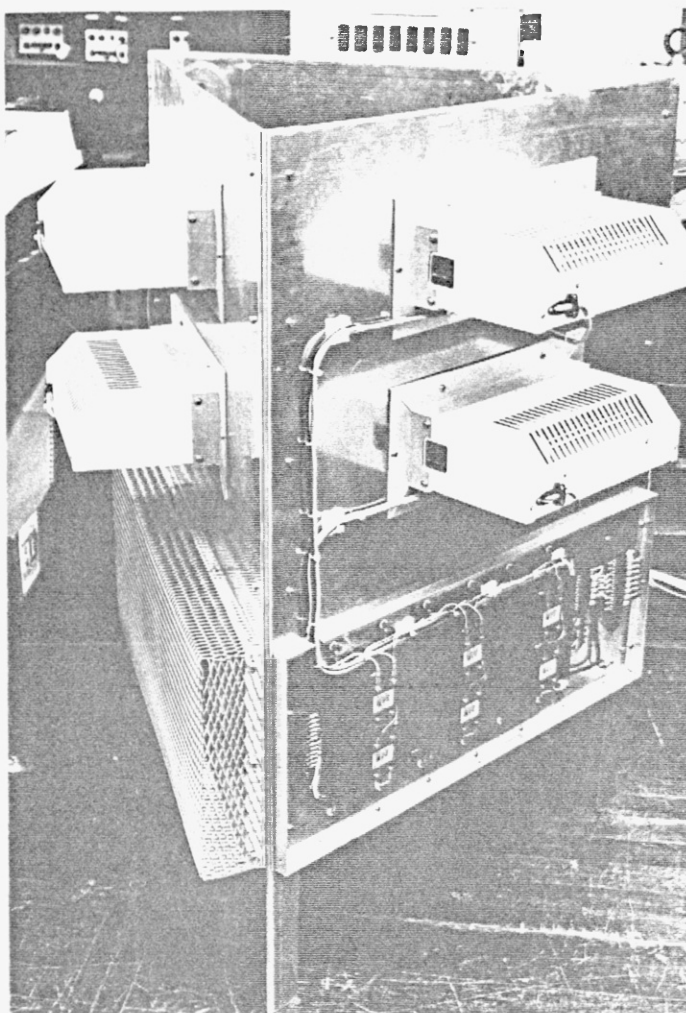
5.5.1. Microcomputer System Specification

The controller microcomputer will be either a single-board or a small multi-board microcomputer satisfying the specifications outlined below.

- (i) It will be based on a 8-MHz clock M68000 CPU, since this is the maximum clock rate allowed by the emulator/analyser.
- (ii) The estimated memory requirements (4.4.2.) should be considered



a) Circuit Diagram.



b) Hardware implementation.

Fig. 5.9 - Load Controller.

as corresponding to a 50% loading. A minimum on-board memory of 4k RAM and 32K EPROM is therefore required. The CPU board must also be provided with sockets for on-board memory expansion.

- (iii) System expansion, e.g. extra memory or interfacing boards, is to be done through a motherboard with a "standard" bus.
- (iv) Operator interaction through a terminal requires at least one RS-232 port for serial I/O.
- (v) Event counting/time interval generation tasks will require at least one 6840 PTM or equivalent support IC.
- (vi) The number of digital input and output signals used by the sensing/actuation subsystems (sections 5.3. and 5.4.) calls for a minimum of two 6821 PIA or equivalent parallel I/O.
- (vii) An 8-channel, analog input block, i.e. analog/digital converter, sample/hold unit and analog multiplexer, is necessary to interface the analogue sensing channels. A 12-bit ADC with a conversion time within 50 μ s is required (Chap. 7).

5.5.2. Set-up of the Actual Microcomputer System

The best buy at the time was a Motorola VME bus M68000 system. The minimum hardware comprises a CPU board, an analogue input board, a parallel I/O board and a backplane or motherboard. In fact board-level products are presently an economical and practical approach to the development of dedicated microcomputer-based controllers. The advantages of this specific solution over alternative multi-board microcomputers were twofold. The CPU board interfaces with the peripheral boards through a 64-pin bus, the I/O channel, thus making the peripheral boards cheaper. It was possible to obtain all the facilities specified with only three boards since their features match the specifications quite well.

A multi-output switch mode power supply satisfying all system needs, including interfacing logic and passive transducers, and allowing

future system expansion was selected. The boards were then mounted in a small purposely built rack also housing the power supply and a small fan (Fig. 5.10). A low-cost but powerful dedicated microcomputer system could therefore be set up.

5.5.3. Functional Description and Relevant Points.

The relevant features of each of the microcomputer boards are briefly described below. The specific hardware set-up is presented in Chapter 6. Detailed functional description is covered in [5.15.-5.17.].

5.5.3.1. CPU Board

This is a M68000 VME bus compatible monoboard microcomputer. The VME bus interface, not used at the moment, provides enormous versatility and expansion capabilities. The VME bus is in fact a widely accepted standard for industrial computing [5.13.-5.14.]. The I/O channel is a low-cost modular interface to peripheral I/O modules based on a 64-pin bus. The analogue input and the dual parallel port modules are interfaced to the CPU board bus through the I/O channel, physically implemented by the 5-slot backplane. Two free slots allow future expansion through the I/O channel, e.g. for digital/analog conversion, memory or other industrial type modules.

No memory comes with the board but the sockets provided can accomodate up to 256 kbytes of user-supplied memory. A wide range of ROM/PROM/EPROM and RAM memory devices can be used. A 6850 based RS-232C serial terminal port allows the use of a terminal. A 6840 PTM provides three independent 16-bit counter timers.

Hardware interrupt handling is provided for two different sets of interrupts. The interrupts caused by the VME bus IRQ signals require the interrupting device to supply the adequate interrupt vector. The local interrupts use the CPU autor-vector feature for vector assignment. Four interrupt levels are assigned to the I/O channel and three to on-board resources.

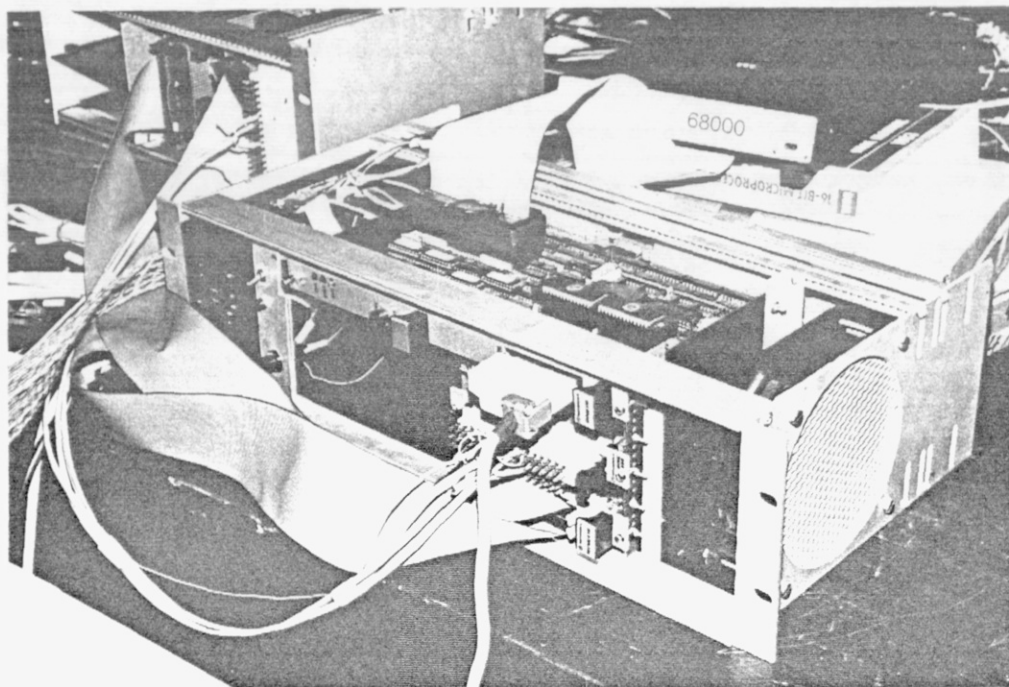


Fig. 5.10 - Microcomputer System Module.

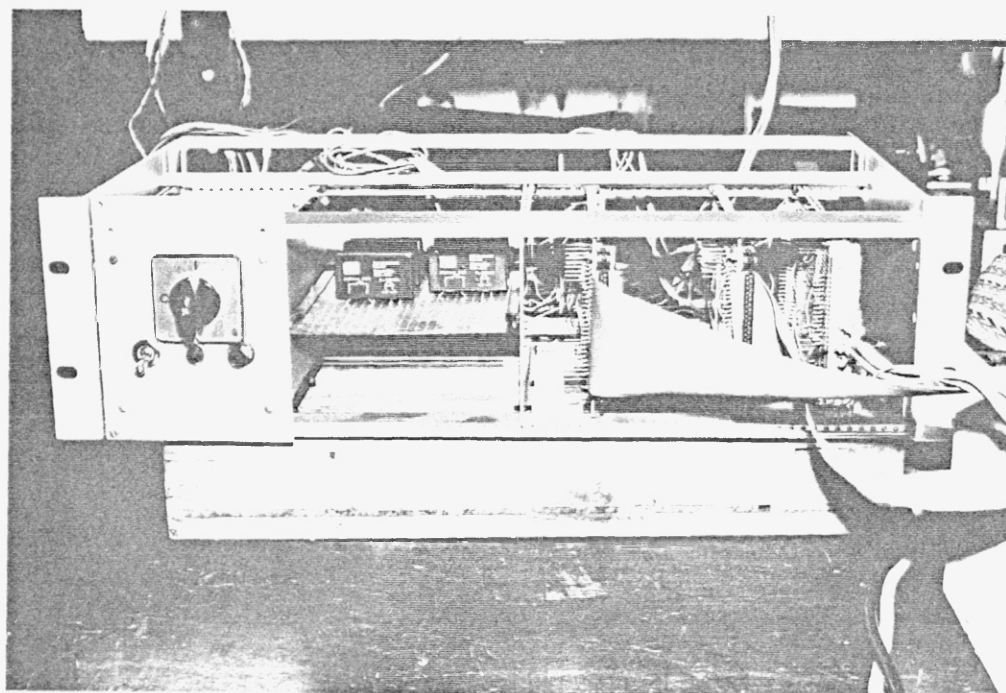


Fig. 5.11 - Interface Circuitry Module.

5.5.3.2. Analogue Input Board

This board includes an analogue multiplexer, a sample/hold unit and a 12-bit successive approximation 'analogue/digital converter. The board provides 16 single-ended or 8 differential input channels. The number of single-ended channels can be expanded in multiples of 16 by adding up to 5 expander modules.

Either voltage or current inputs can be used and the input impedance is greater than 10 M Ω . The hardware selectable input ranges ($\pm 0.5V$, $\pm 0.1V$) can be combined with the software programmable range selection ($\times 1$, $\times 10$) giving four different full-scale ranges. The ADC has a conversion time of 38 μsec to 0.1% $\pm 1\text{LSB}$.

5.5.3.3. Dual Parallel Port Board

Two buffered 6821 PIA's provide a total of 32 data lines and 8 control lines. Each group of 8 peripheral data lines is hardware strappable as input or output and each line is capable of sinking 24 mA. Each PIA can drive up to two of the four I/O channel interrupt lines.

5.6. MODULAR HARDWARE IMPLEMENTATION

The conceptual hardware model expressed by the Fig. 3.1 has evolved into the actual hardware design. The actual hardware blocks building up the complete system are shown in the System Block Diagram (Fig. 5.12). Only the functional relations between blocks are shown. Each block is identified by its name which portrays the respective function. The signals are not individually identified since the nature of each data channel can be readily inferred. For the sake of simplicity power supply lines, grounds, shields, etc. are not shown in this functional diagram.

The diagram shows neat boundaries (horizontal partitioning) isolating subsystems of different type. From top to bottom we have first the energy conversion hardware. Next are the sensors and actuators directly attached to the process. These are followed by signal

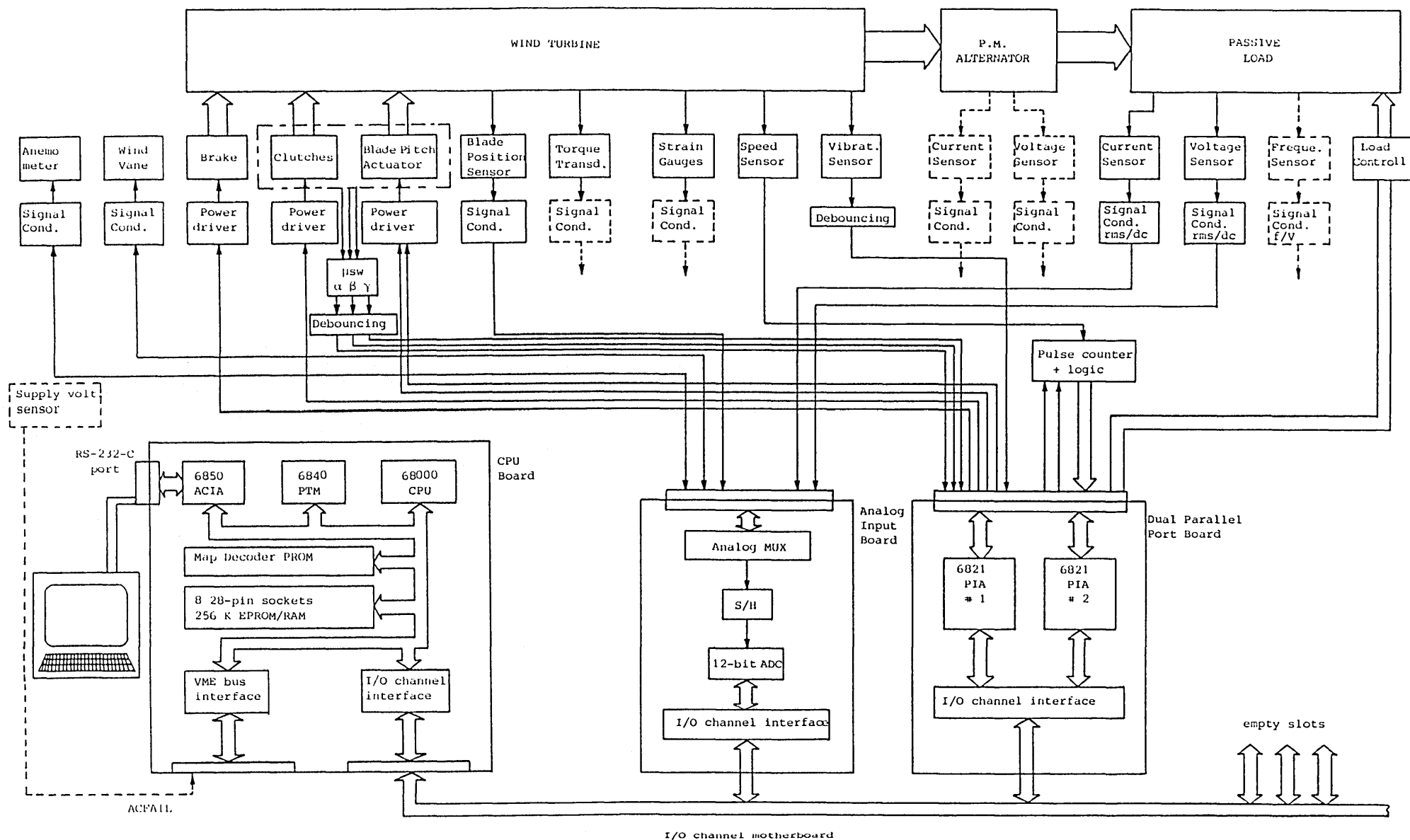


Fig 5.12 - System Block Diagram.

conditioning or power driver blocks which link directly with the microcomputer peripheral interfaces. Details on the specification/design of each of these blocks were given in the last sections. The dotted lines represent hardware subsystems yet to be implemented.

Most of the purpose designed and built interface circuitry was mounted in three single eurocard format boards assembled, together with the required power supply, in a 19 inch module (Fig. 5.11). A fourth board carries the SSR's of the actuator and brake power drivers and space is still left for expansion. The microcomputer and interface boards are double eurocard and single eurocard format and they are also mounted in a 19 inch module. Expansion is still possible through the two empty I/O channel motherboard slots. Finally both modules are installed in a 19 inch rack also housing the actuator driver bulky hardware.

CHAPTER 6

SYSTEM INTEGRATION AND TESTING

6.1. INTRODUCTION

System integration is undoubtedly a crucial development stage of a microcomputer-based system. Controller software and hardware are interfaced and tested with the plant, actuation and transducing hardware. Algorithmic and software bugs as well as electrical interfacing and mechanical problems are simultaneously present masking the effects of feedback obtained from testing.

The aims of this testing stage were to complete software/hardware integration and fully test the controller together with the WECS.

6.2. CONTROLLER SOFTWARE/HARDWARE INTEGRATION

A top-down incremental approach was used to integrate and test the simulation version of the control program (section 4.4.1.2.). The top modules of this simulation version were kept unchanged in the actual control program because they were completely hardware-independent. The "logical" upper half of the SC, including the major system functions, is therefore already integrated. The "physical" lower half, comprising all utility modules, lends itself to a different integration approach, which is now described.

The bottom modules of the SC (Fig. 4.3) were first tested while checking the behaviour of the microcomputer peripheral hardware. Dedicated software did the systematic testing of the various peripheral support chips, i.e. PTM, ACIA, PIA and ADC.

After debugging the modules forming the software/hardware boundary, integration progressed in a bottom-up approach. Each bottom module was first run under a driver program, later replaced by the actual superordinate module. This was in turn tested by a driver one step higher in hierarchy, etc., progressing upward in the SC. Extensive testing was done at each stage to ensure that all program paths were exercised and all possible data values taken into account. Bottom-up incremental integration allowed excellent parallel implementation of all utilities and addressed the testing of the major system interface,

the physical I/O, early in the integration stage. The drawbacks of this approach were avoided by stopping bottom-up integration as soon as the device-drivers, e.g. Actuator, Delay or Write modules (Figs. 4.2 to 4.4), were completed.

The remaining of the SC was then linked with its bottom part. The high level of cohesion and the loose coupling of the modules implementing this link enabled a clean interface to be built. The best of top-down and bottom-up approaches to integration could therefore be combined.

6.3. REAL-TIME TESTS WITH AN ANALOG COMPUTER MODEL OF THE WECS

6.3.1. Introduction

The controller software/hardware integration reached a stage where some form of test harness was required. Since the assembly of the WECS hardware was not, at the time, completed an intermediate testing stage was sought.

An analog-hybrid computer system used for simulation of WECS was available offering a versatile environment for testing microcomputer-based controllers [1.3.]. The advantages of this intermediate testing stage were twofold :

- the low-level hardware-dependent software modules could be linked in the same program and tested in an on-line control algorithm,
- valuable information on the controller real-time behaviour could be obtained and the microcomputer processing power once more ascertained.

6.3.2. Analog Computer Modelling of VSVF WECS

Analog computer models have been developed to investigate the dynamic behaviour of different WECS configurations, as well as the effects on system performance of several control strategies [1.3.].

Several models were available that could provide the digital controller with a real-time testing environment.

The selected model represents a system configuration similar to the VSVF WECS under study : a high tip-speed-ratio rotor drives a p.m. generator feeding a switchable resistive load. The control scheme is similar to the one implemented in the actual system : a speed feedback load switching strategy attempts lambda optimum operation by loading the turbine with a quadratic load torque.

The diagram in Fig. 6.1, comprising both WECS and controller models, is implemented by analog computer simulation networks. An analog tape recorder supplies the wind signal which is the input to the model. Measuring networks provide values of the energy extracted by the WECS for a given run, allowing system performance under several control strategies to be compared.

6.3.3. Interfacing the Microcomputer Controller

To test the microcomputer controller with the WECS analog computer model, the network simulating the analog controller (enclosed by the dotted line in Fig. 6.1) had to be replaced with the digital controller which was then interfaced to the plant dynamics model.

This interfacing is shown in detail in Fig. 6.2. WECS power, speed, torque and wind speed signals are input through the ADC. The PIA digital outputs are the actuation signals to the cut-in speed function relay, FR3₁ and to the D/A switches controlling the load. The I/O impedance levels allow the simulation network outputs to be directly connected to the ADC board. The ADC voltage range is set to ± 10 V, the range used by the analog computer, and a common ground is provided. The buffered PIA outputs are directly connected to the control inputs of the function relay or D/A switches.

A simulation can then be run with the digital controller replacing the analog controller network.

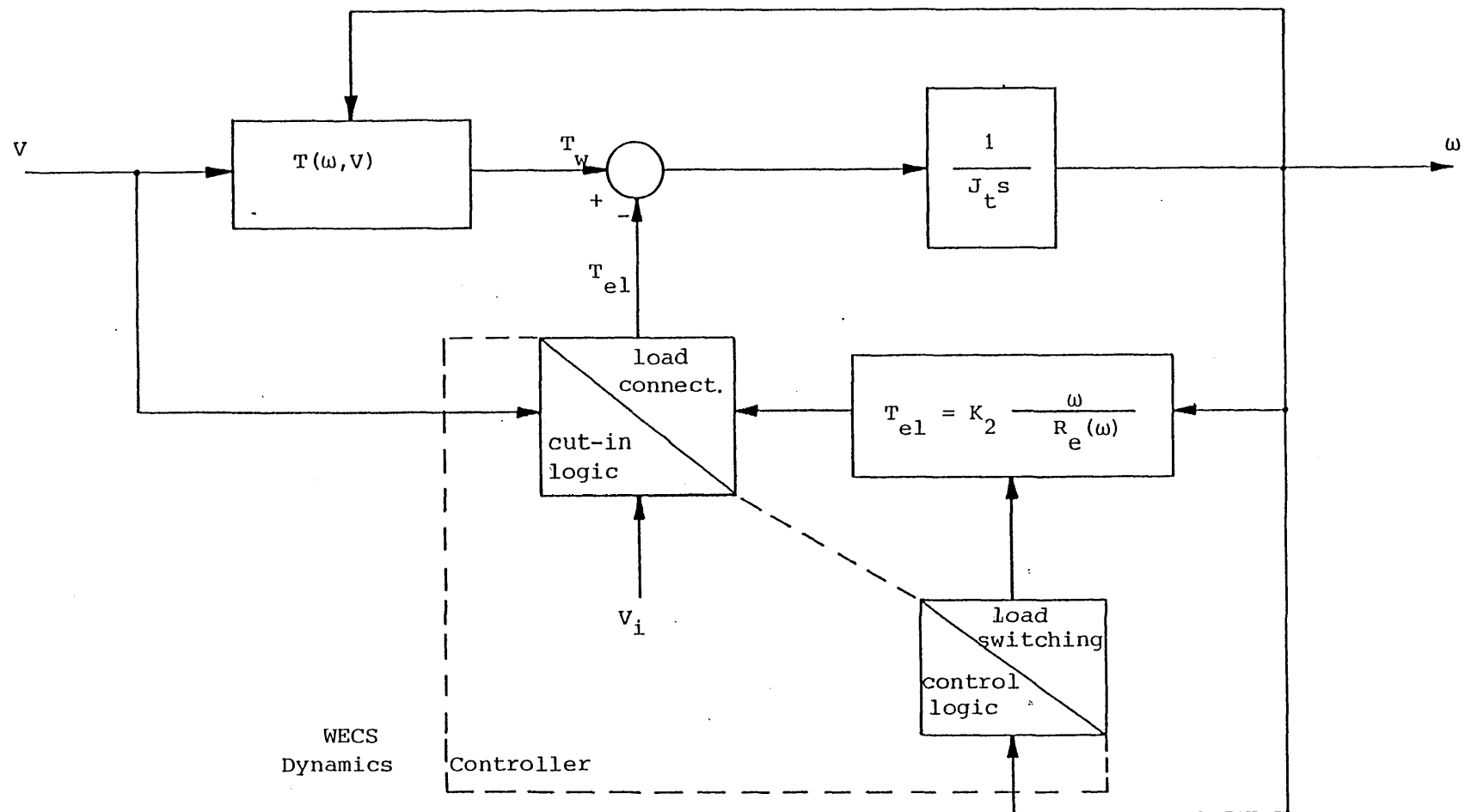
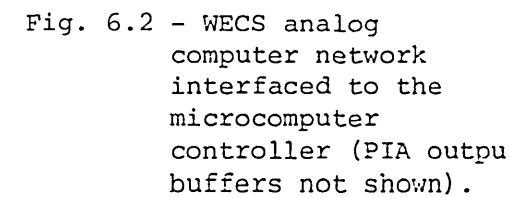


Fig. 6.1 - WECS analog computer model used for controller real-time tests.



6.3.4. Testing Software

Software had to be provided to implement the required load switching algorithm. The simple control loop is presented by the DFD in Fig. 6.3. A small interactive program was designed to support the execution of the control software and provide on-line display of variable values (Fig. 6.4). The controller action is portrayed by continuously updating the system operating point on the displayed load switching function. A limited data logging facility allows off-line display of a number of samples of selected system variables.

The real-time behaviour of the microcomputer-controller was then tested by running analog computer simulations. System performance was measured and compared with the results presented in [1.3.] for the same wind samples and WECS configuration. The WECS operating efficiency values obtained were reasonably close to the values presented in [1.3.]. For most of the runs the performance degradation due to the digital controller was less than 3%. The performance of the microcomputer controller can thus be considered quite satisfactory specially if we take into account that, although the algorithm implemented is quite simple, the simulations were run eight times faster than real-time.

6.4. TESTING OF HARDWARE SUBSYSTEMS

6.4.1. Energy Conversion Hardware

6.4.1.1. P.M. alternator

The generator used is a new Mark II p.m. alternator, built according to the design presented in [5.1.]. Final machine set-up and some selected tests are now described.

The 12 barium ferrite magnets were magnetised using the technique described in [5.1.]. Each magnet was clamped in the iron path of a magnetic circuit formed by a 400 turns coil inside a mild

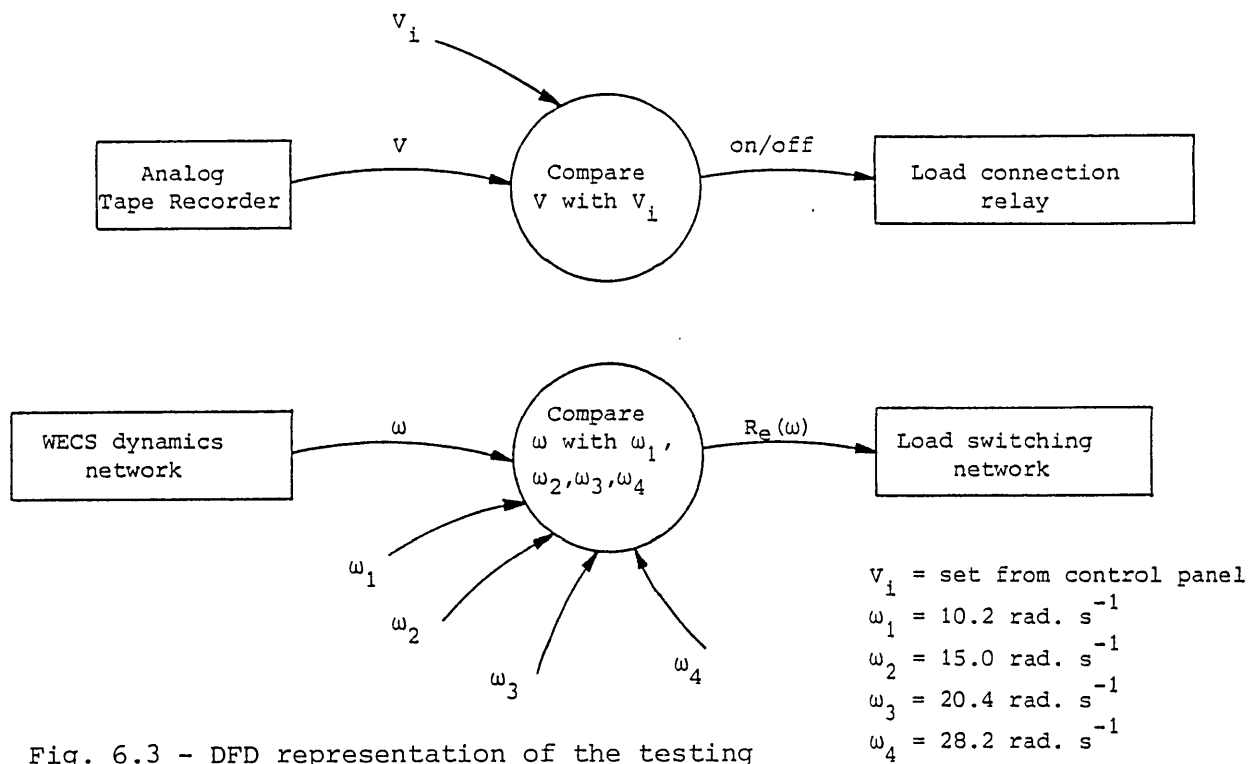


Fig. 6.3 - DFD representation of the testing software.

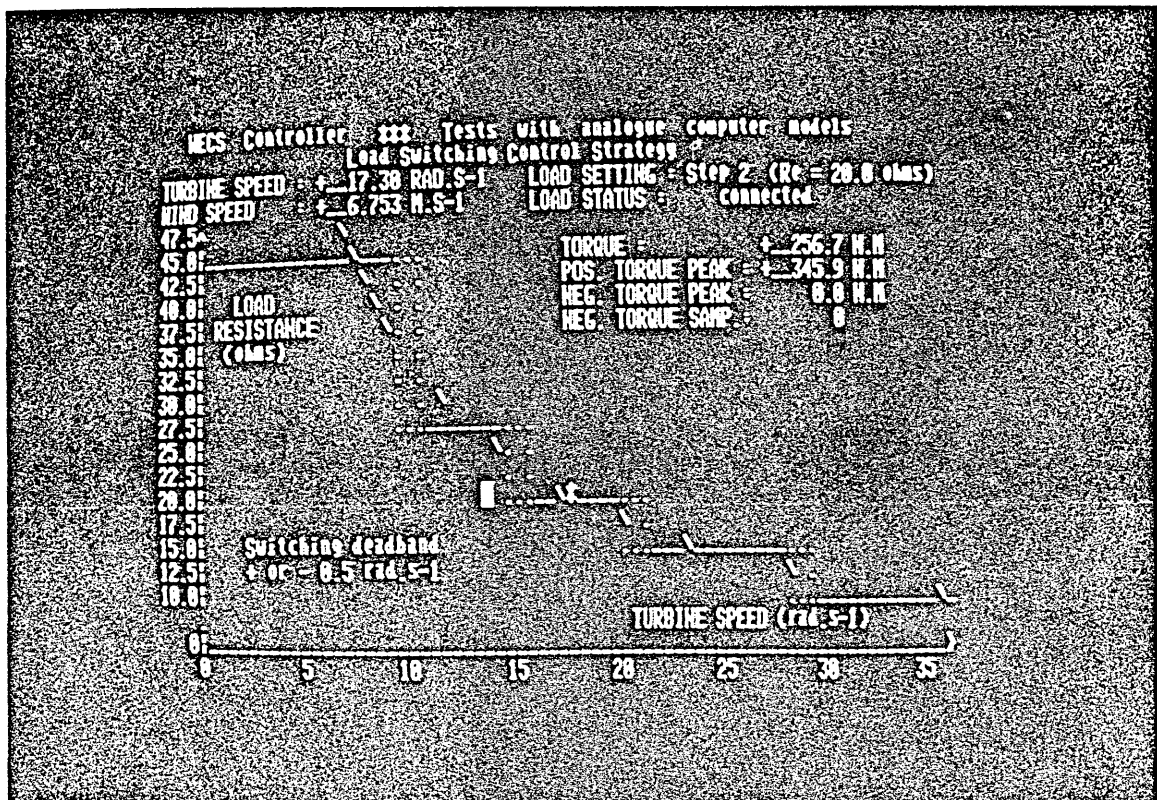


Fig. 6.4 - On-line display of variable values and WECS operating point.

steel magnetisation jig. By applying a short d.c. current pulse (400 V and 40 A) each magnet was magnetised up to its remanent field. An expected flux density of around 0.16 T was measured close to the face of each magnet with all magnets duly positioned inside the rotor case. The stator windings were connected in series and submitted to three thermal cycles by applying a d.c. current. In each of the cycles the temperature of the outer coils was first made to rise from 20°C to 105°C and then back to 20°C. At the end of each cycle the phase/earth and phase/phase insulation were checked. An insulation impedance better than 5000 MΩ was found for an applied voltage of 1000 V, which is quite satisfactory.

The machine was then driven up to the nominal speed and both open-circuit and short-circuit tests were done. The curves obtained, presented in Fig. 6.5, show good agreement with those presented in [5.1.]. It is therefore assumed that this machine is, for all practical purposes, identical to the one whose fully set of test results is presented in [5.1.]. These were used as applying to the new machine whenever necessary, e.g. in the load controller design.

6.4.1.2. Mechanical transmission system

Rotating the wind-turbine in the laboratory was considered a necessary testing step in order to check the actual running behaviour of the elements of the transmission system and the speed measurement hardware.

It was decided to use the p.m. generator as a motor, a variable frequency variable voltage supply was therefore necessary. A d.c. motor d.c. generator set was used for this purpose and, after overcoming a number of minor mechanical problems, the turbine was smoothly speeded-up to 800 rpm. The speed measurement hardware were then checked and the respective software channel calibrated. The operation of the blade pitch actuation mechanism was also checked over the whole speed range.

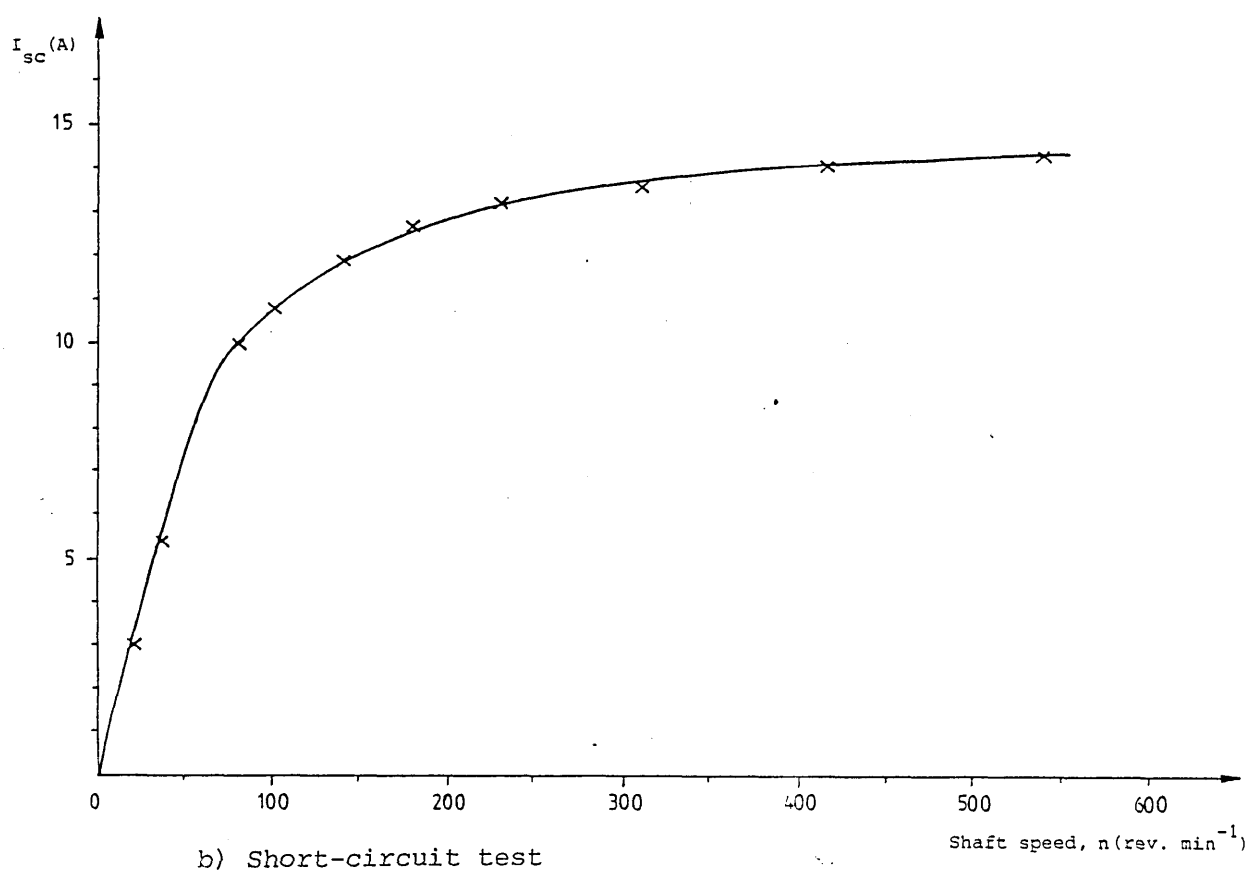
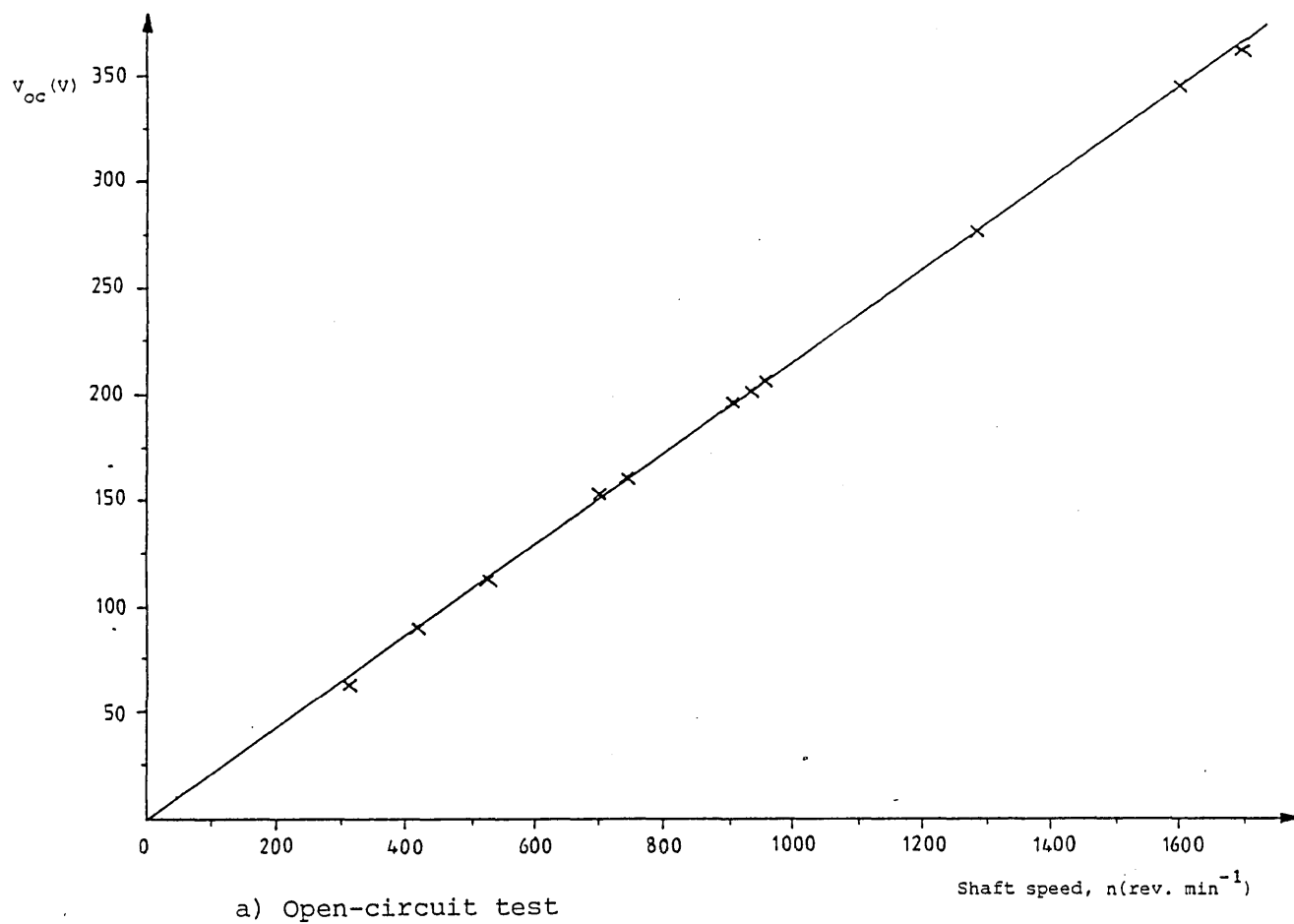


Fig. 6.5 - Mark II p.m. generator test results.

6.4.2. Actuation/Transducing and Power Driving/Signal Conditioning Hardware

Simple testing procedures checked the isolated operation of each transducer/actuator. Transducers were supplied with the necessary power sources and the measured output signals checked against the values of the transduced variable. Actuators were energised as required and checked for functionality.

All the transducers and actuators were then connected to the interface boards. Operation was then checked by simulating the microcomputer output signals to the actuators and observing the signals from the transducers to the microcomputer. Troubleshooting of the power driving/signal conditioning circuitry could thus be done without the microcomputer.

6.5. COMPLETE INTEGRATION AND LABORATORY TESTS

This was the final testing stage undertaken in the laboratory. Hardware and software were separately integrated and tested and the hardware/software boundaries debugged. The complete system was then put together as in Fig. 5.12, and the actual WECS control program debugged using ICE. Once complete testing was achieved, the control program was burnt into EPROM's and the system was ready for field tests.

Prior to final testing two important issues were addressed, namely the set-up of the microcomputer hardware and the necessary dedicated testing which preceeds and supports the final integration. These topics are dealt with in Appendix 6.

CHAPTER 7

SYSTEM MODELLING AND CONTROL

7.1. INTRODUCTION

This chapter deals with the issues of system modelling, dynamic analysis and digital compensator design and tuning. These tasks are mainly independent from the other controller development activities and the material included is the result of almost self-contained parallel work. The conclusions are to be fed into the software and hardware design stages. Further modelling and analysis work may be required in a later stage if field tests advise compensator redesign (Fig. 2.5).

The control policies aiming to optimize wind energy extraction will always have the form of direct digital control (DDC) loops. Since input of external data from transducers and output of control data to actuators is done through low-level software modules under the supervision of the system scheduler (see section 4.4.3.2. and Fig. 4.3), the DDC modules contain exclusively code performing mathematical manipulations. They are therefore a very small percentage of the total software and do not offer any implementation difficulties since they are coded in Pascal and kept free from hardware-dependent features.

The design of digital compensators is, however, a very complex task. Reasonable insight into the dynamic behaviour of the controlled system is required for guidance in the selection of the compensator structure and in the assessment of parameter values. The design and tuning of the digital compensator will determine system performance.

The use of the continuous/discrete-time design techniques referred throughout the chapter was supported by dedicated software (Appendix 7).

7.2. CONTROL STRATEGIES FOR VARIABLE-SPEED WECS

7.2.1. Control Loops, Controlled and Manipulated Variables

The control requirements usually specified for a WECS

[2.1.,7.1.] are run-up and run-down speed control, emergency speed/power control and normal running speed/power control. As it has already been referred, the choice of system operating mode and electrical interfacing will dictate how each particular control system is configured. Nevertheless speed and/or power control loops are always present [2.1.,2.2.,7.1.,7.2.].

The control methods available are usually limited to blade pitch and load control. Blade pitch control, a mean of controlling the wind generated input torque by varying the rotor aerodynamic characteristics, is incorporated in some WECS either in an active or in a passive form. Control of the load torque is often included and can be done in a number of ways depending on the electrical interfacing. Medium to large WECS usually have more than one control loop with a supervisory type of control deciding which loop to activate at any time.

7.2.2. Implementation of Speed and Torque Control Loops

As it has already been discussed in Chapter 5, the specifications outlined for the WECS controller call for both speed and torque control loops. Speed control is the main loop in the start-up and normal and emergency stop modes. Electrical torque control is the main loop in the normal running mode, i.e. optimal/non-optimal operation, with speed control being part of the protection functions. Speed and torque control are here implemented within the same on-line control algorithm.

Speed control as a main loop is done by blade pitching. And the protection loop actuation is also done by pitching the blades to spill the excess wind power input (Fig. 3.2). Active blade pitch control is implemented through an electrical actuation scheme, as described in section 2.1.

The main control in the normal running mode is exercised on the electrical torque via controllable load resistors (section 5.4.4). The high efficiency and simplicity of a.c. load control methods are very attractive features [1.5.,1.8.]. Results from dynamic analysis

studies [1.3.] showed that a simple scheme using a 5-steps switched load performs almost as well as a load modulation strategy with continuous load variation. Torque discontinuities are minimised by a reasonable number of load steps, and the frequent switching duty is not a problem if solid-state switches are used.

7.2.3. Control Strategies for Optimum Wind Energy Extraction

Optimum energy extraction is achieved by running the system in a VSVF mode. The rotor speed is allowed to vary in sympathy with the wind speed by maintaining the tip-speed ratio to the value that maximises aerodynamic efficiency. In order to achieve this, the electrical torque has to be controlled so as to match the input wind torque. This can be done in a number of different ways [1.3.].

A so-called lambda optimum control strategy using turbine speed feedback was chosen here. The feedback signal is used to derive the deviation between the actual and the desired value of lambda. This is the input to a digital compensator the output of which is the control signal to the electrical actuation (Fig. 7.1).

An alternative implementation is to replace the wind speed value directly read from the anemometer by an estimated value of the wind speed "seen" by the rotor (Figs. 3.12 and 3.19). The estimated value can be considered as more representative of the wind speed for two reasons :

- the anemometer reads the wind speed on a particular spot while the turbine feels the effect of the spatially averaged wind speed,
- location of anemometer is crucial. If on an upwind boom, the wind speed reading is that which will reach the turbine sometime later.

This variable time lag can however be compensated for by the controller.

The price to pay for this more accurate wind speed value is a more complex algorithm requiring extra transducers and a longer control cycle.

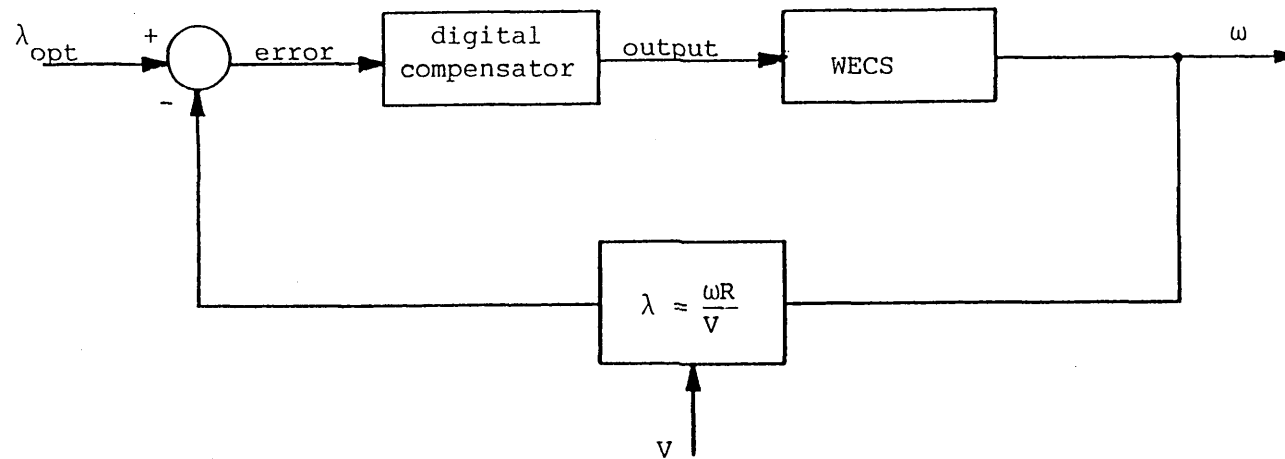


Fig. 7.1 - Lambda optimum control strategy.
Speed feedback scheme.

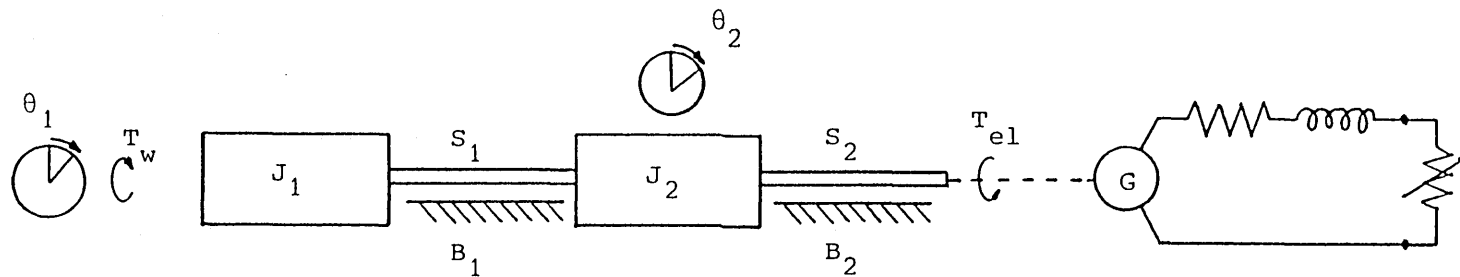


Fig. 7.2 - WECS simplified model as a 4th order mechanical rotational system.

Power feedback can also be used in a scheme that implements the lambda optimum operation in an indirect manner - the optimum power for the actual system speed is retrieved from a look-up table and compared with the actual power. The error signal thus derived will be the input to the compensator. This will however, require "a priori" knowledge of the optimum power/speed locus of the wind turbine. Hill-climbing algorithms maximising extracted power are another alternative. Simulation results and extensive discussion of these methods of WECS control are presented in [1.3.,1.8.].

Although only the first of these strategies has actually been implemented, the modular structure of the software will allow easy inclusion of program blocks with the alternative control algorithms. The interactive modules and the scheduler structure were in fact designed as to allow such extensions. On-line operator choice of one of the available control strategies can be done during system set-up. When transferring the system to the normal running mode the scheduler will then branch the appropriate software module.

All these control strategies can be implemented with the actual system configuration, since both the energy conversion and the measurement and actuation hardware have been designed with that aim in mind.

7.3. DIGITAL CONTROLLER DESIGN

7.3.1. Approaches to Digital Controller Design

In the design of digital, discrete-time controllers for continuous-time processes several routes are possible allowing system modelling and controller design to be done using continuous- and/or discrete-time techniques. Detailed development of these design paths and discussion of their relative merits can be found in the literature [7.3.-7.7.]. All the methods have a common point : they all begin with a sufficiently accurate mathematical model of the process to be controlled.

The design methods can be broadly divided into "exact" and "approximate" methods. The "exact" methods use a discretized model of the continuous system and are essentially discrete design methods. The "approximate" methods use, as a first stage, conventional classical continuous design techniques (Root locus, Bode plots, ...) to obtain an analog compensation network, which is then digitized.

Discretization of an analog controller may lead to destabilisation of the closed-loop system if the sampling time is lowered because this causes the system dynamics to change. Digital systems designed in this manner are always slightly inferior to analog systems due to the inherent time delay caused by the hold circuit [7.7.]. The different discretization methods available work well only if the sampling time is short.

This does not happen with discrete design methods for which, in theory, the behaviour of the closed-loop system is independent of the initial choice of the sampling time. Nevertheless if too low a sampling rate is chosen the exactness of the methods vanishes due to controller saturation or intersample behaviour not detected by z-transform analysis. Digital design methods may be based on state-space or input-output models and the use of pole placement techniques. A more general approach is to use optimal design methods which allow linear but time-varying and multivariable systems to be handled. Again state-space or input-output models can be used. Finally different adaptive control formulations can be used to produce more complex regulators. Digital design methods are therefore to be preferred.

7.3.2. Actual Route to System Analysis and Controller Design

The approach chosen, based on one of the so-called "exact methods", consists essentially of the following steps:

- i) continuous-time modelling to obtain the system transfer function;
- ii) model simplification and continuous-time analysis;
- iii) transformation of the transfer function into the z domain;
- iv) digital compensator design and tuning;
- v) discussion of implementation related problems.

A state-space linearised model of the controlled plant is first derived. This internal model presents the WECS as a 2-input 2-output system and shows the interaction within the plant. This continuous-time model allows the transfer functions of the main and coupling elements to be derived and the two feedback loops to be identified (Fig. 7.3 and 7.4).

Three features of the closed-loop system model (Fig. 7.4) prevent an easy treatment through standard mathematical techniques. First the inherent turbine feedback loop which, while expressing the turbine self-regulating torque characteristic, is essentially a non-controllable loop. Second the presence of the windspeed feedforward loop determining the setpoint changes in the controller loop, as required by the chosen control policy, complicates the block structure. Finally a digital computer is included in the control loop. Both discrete and continuous-time signals are thus present in the model, i.e. the controlled WECS is a sampled-data system, calling for specific analysis/design tools.

A simplified analysis is done assuming it is reasonable to decouple the two loops (Fig. 7.5). The multivariable system is then seen as two single-input single-output (SISO) systems.

The first loop includes only continuous-time signals (Fig. 7.6). Continuous-time analysis of the WECS response to wind torque changes is then undertaken assuming a constant electrical load torque. S-plane root locus and Routh-Hurwitz criterion are used to evaluate the system dynamic behaviour and stability margins for different operating points. System sensitivity to parameter changes is also investigated.

The second loop (Fig. 7.7), where the wind torque input is considered as a disturbance, will allow the system response to controller setpoint changes to be determined. The corresponding SISO system is a sampled-data system. The discretization of the continuous plant is done by the so-called impulse-invariant response method.

A compensator structure is then chosen and z-plane root-locus techniques are used to relocate the system closed-loop poles in a

favourable position. The closed-loop performance is then improved by tuning the compensator parameters with the help of step/ramp response methods.

Finally relevant practical aspects of the implementation of the digital controller are discussed.

7.4. SYSTEM MODELLING

Detailed quantitative control system design requires a fairly complete system dynamic model valid over a wide range of operating points. A physically-based model approach was preferred for its versatility and for allowing plant complexities, e.g. non-linearity, dead-time, etc., to be considered either from start or in a later review of the model.

The mechanical transmission system between the turbine rotor and the generator (Fig. 2.1) includes two rubber couplings. The extra compliance and damping introduced will help to attenuate torque transients, specially if the system is to operate in a fixed-speed mode. As a consequence, the simple inertial model usually used in the analysis of VSVF systems [1.3., 7.8., 7.9.] does not apply here. Modelling the soft transmission WECS through classical rotational dynamics [7.10.-7.15.] was the approach taken. A fourth order model was found to adequately represent the WECS under study. This model is derived in Appendix 7, where the simplifying assumptions are discussed with detail and the values for the non-linear coefficients derived for several operating points.

The dynamics of the WECS are thus described by a set of torque balance equations. These differential equations apply to the physical system presented by the diagram in Fig. 7.2. :

$$\left\{ \begin{array}{l} T_w = J_1 \ddot{\theta}_1 + B_1 (\dot{\theta}_1 - \dot{\theta}_2) + S_1 (\theta_1 - \theta_2) \\ 0 = B_1 (\dot{\theta}_1 - \dot{\theta}_2) + S_1 (\theta_1 - \theta_2) - J_2 \ddot{\theta}_2 - B_2 \dot{\theta}_2 - S_2 \theta_2 - T_{el} \end{array} \right. \quad (7.1a)$$

$$\left\{ \begin{array}{l} T_w = J_1 \ddot{\theta}_1 + B_1 (\dot{\theta}_1 - \dot{\theta}_2) + S_1 (\theta_1 - \theta_2) \\ 0 = B_1 (\dot{\theta}_1 - \dot{\theta}_2) + S_1 (\theta_1 - \theta_2) - J_2 \ddot{\theta}_2 - B_2 \dot{\theta}_2 - S_2 \theta_2 - T_{el} \end{array} \right. \quad (7.1b)$$

or

$$\left\{ \begin{array}{l} \ddot{\theta}_1 = -\frac{B_1}{J_1} (\dot{\theta}_1 - \dot{\theta}_2) - \frac{S_1}{J_1} (\theta_1 - \theta_2) + \frac{1}{J_1} T_w \\ \ddot{\theta}_2 = \frac{B_1}{J_2} (\dot{\theta}_1 - \dot{\theta}_2) + \frac{S_1}{J_2} (\theta_1 - \theta_2) - \frac{B_2}{J_2} \dot{\theta}_2 - \frac{S_2}{J_2} \theta_2 - \frac{K_1}{J_2} \cdot \frac{1}{R_1} \end{array} \right. \quad (7.2a)$$

$$(7.2b)$$

$$\text{where } T_{e1} = K_2 \frac{\omega}{R_e(\omega)} = K_1 \frac{1}{R_1} \quad (7.3)$$

assuming a quadratic loading of the wind turbine [1.3.], as required for optimum lambda operation.

7.4.1. State-Space Model Formulation

The multivariable system lends itself to a representation in state-space form.

Choosing the variables as follows:

$$\left\{ \begin{array}{l} x_1 = \theta_1 \\ x_2 = \theta_2 \\ x_3 = \dot{\theta}_1 = \dot{x}_1 \\ x_4 = \dot{\theta}_2 = \dot{x}_2 \end{array} \right. \quad (7.4a)$$

$$(7.4b)$$

$$(7.4c)$$

$$(7.4d)$$

the resulting state equations are given by

$$\left\{ \begin{array}{l} \dot{x}_1 = x_3 \\ \dot{x}_2 = x_4 \end{array} \right. \quad (7.5a)$$

$$(7.5b)$$

$$\left\{ \begin{array}{l} \dot{x}_3 = -\frac{S_1}{J_1} x_1 + \frac{S_1}{J_1} x_2 - \frac{B_1}{J_1} x_3 + \frac{B_1}{J_1} x_4 + \frac{1}{J_1} T_w \end{array} \right. \quad (7.5c)$$

$$\left\{ \begin{array}{l} \dot{x}_4 = \frac{S_1}{J_2} x_1 - \frac{S_1+S_2}{J_2} x_2 + \frac{B_1}{J_2} x_3 - \frac{1}{J_2} (B_1+B_2) x_4 - \frac{K_1}{J_2} \frac{1}{R_1} \end{array} \right. \quad (7.5d)$$

In vector form these equations become :

$$\begin{cases} \dot{\underline{x}} = A \underline{x} + B \underline{u} \\ \underline{y} = C \underline{x} \end{cases} \quad \begin{matrix} (7.6a) \\ (7.6b) \end{matrix}$$

where $\underline{x}$ is the state vector, A the system matrix, B the distribution matrix and $\underline{u}$ the input or control vector.

The output $\underline{y}$ is related to the state variables through the output matrix C . All these vectors and matrices are given in Appendix 7.

7.4.2. Transfer Function Matrix

The system transfer function matrix [7.16.] is given by:

$$G(s) = C[sI - A]^{-1} B = \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix} \quad (7.7)$$

where $G_{ij}(s)$ are the scalar transfer function relating each output y_i to each input u_j . They are also derived in Appendix 7.

The system is therefore given by

$$\underline{Y}(s) = G(s) \cdot \underline{U}(s) \quad (7.8)$$

or

$$\begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} = \frac{1}{|sI - A|} \begin{bmatrix} \frac{1}{J_1} A_{33} & -\frac{K_1}{J_2} A_{43} \\ \frac{1}{J_1} A_{34} & -\frac{K_1}{J_2} A_{44} \end{bmatrix} \begin{bmatrix} T_w \\ \frac{1}{R_1} \end{bmatrix} \quad (7.9)$$

where ω_1 and ω_2 , the turbine and generator speeds, are the output variables. The input variables are T_w , the wind produced torque, and $\frac{1}{R_1}$ the load value determining the electrical torque.

$G(s)$ is the system transfer function matrix or plant matrix.

A_{33} , A_{34} , A_{43} and A_{44} are given in Appendix 7.

7.4.3. Linearised Model

The system model has non-linear parameters such as

$$S_1(T_w), B_2(\omega_1) \text{ and } S_2(T_w, \omega_2)$$

It is possible to linearise the system model for a given operating point and study its dynamic behaviour considering only small perturbations from that operating point. However, since no steady-state operation at a certain fixed turbine speed, torque or power is to be expected, due to the system operating mode, the system cannot be described through just one "nominal" operating point. Complete system description requires a rather large number of operating points to be considered, although reasonable insight can be gained with a more practical approach. Three operating points were chosen assuming a lambda optimum operation strategy, for turbine speeds of 700, 400 and 200 rpm, which were believed to be typical of the WECS operation. They will be referred as (i), (ii) and (iii), respectively.

By using expansion into Taylor series about a steady-state operating point a small-signal linearised model is obtained (Appendix 7) which is given by

$$\Delta \dot{\underline{x}} = A^* \Delta \underline{x} + B^* \Delta \underline{u} \quad (7.10)$$

where A^* and B^* are the Jacobian matrices derived from A and B .

From the non-linear large signal model initially developed a linearised small-signal representation is thus obtained.

7.4.4. Simplified Model

7.4.4.1. Open-loop and closed-loop multivariable representations

The linearised state-space model is an open-loop 2-input 2-output system representation. The matrix equation given by equation 7.8 can be expressed in block diagram form as in Fig. 7.3. The input/output behaviour of the multivariable system is described through the so-called

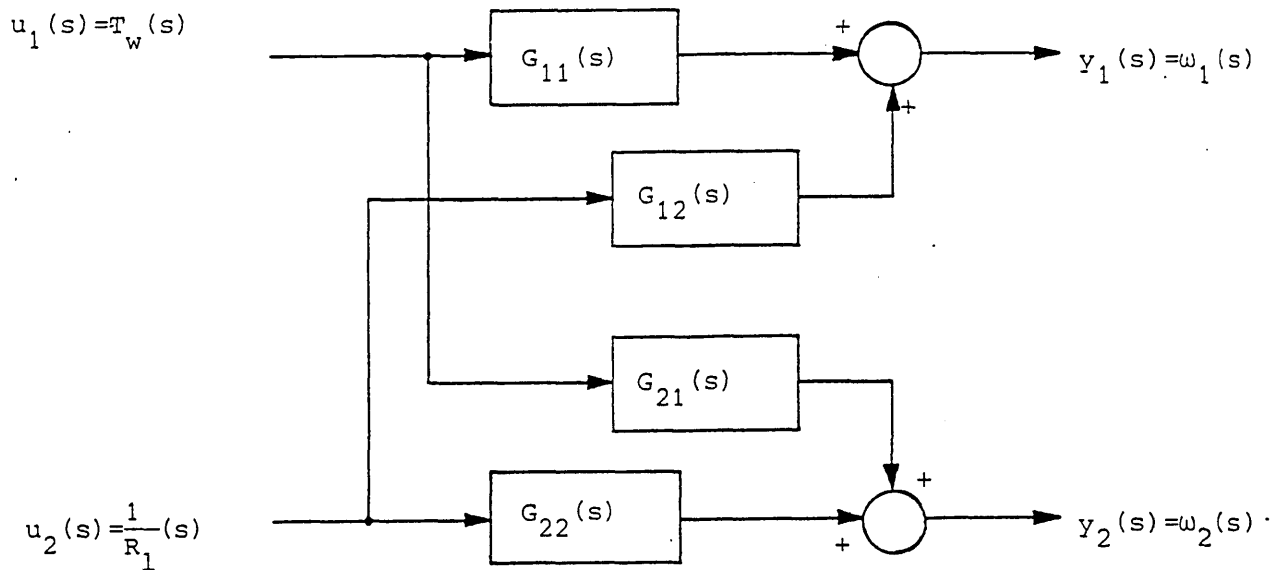


Fig. 7.3 - Open-loop 2-input 2-output system representation.

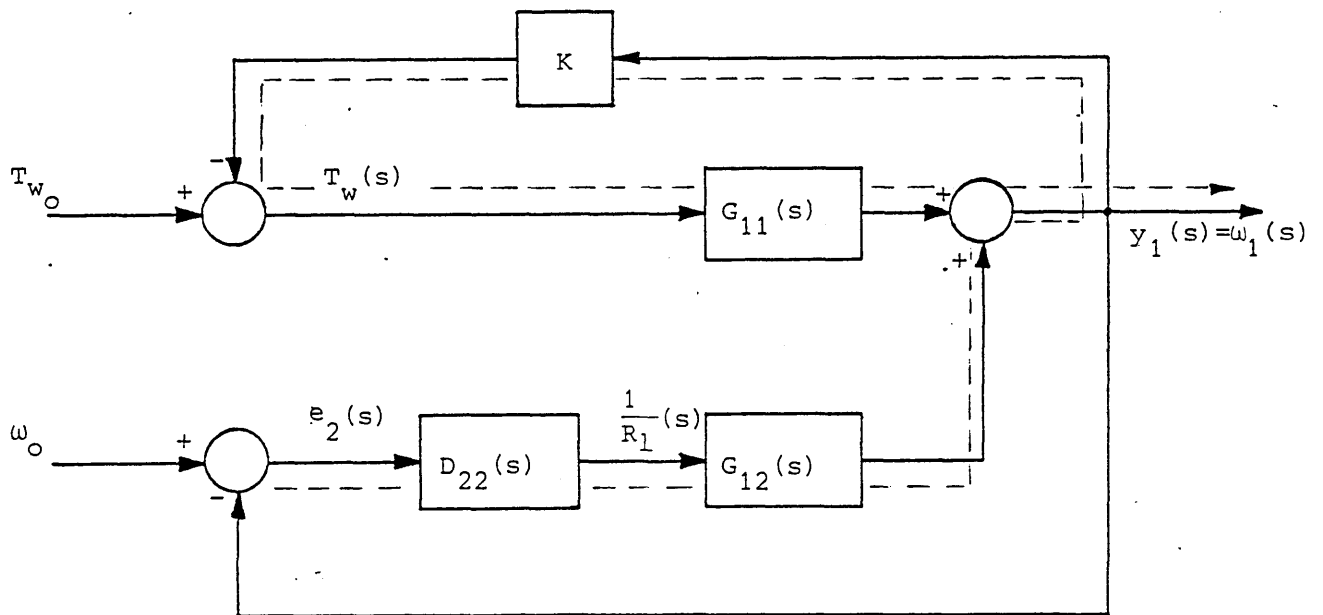


Fig. 7.5 - Simplified 2-input 1-output closed-loop model.

p-canonical structure with the summation points at the outputs. The interaction between input and output signals is clearly represented. The upper summing point states that the turbine output speed is a result of the contributions from the wind produced torque, T_w , and from the torque determined by the electrical load, $1/R_1$.

By including in the model the plant inherent feedback and closing the control loop with the necessary compensator a closed-loop 2-input 2-output system model is obtained. The full block diagram is given in Fig. 7.4 and is similar to that of a multivariable control system with in-plant interaction [7.16.-7.18.]. There are basically two feedback loops both from the turbine speed, ω_1 . The upper loop expresses the inherent plant speed feedback and has a variable gain K , which is the symmetrical of the slope of the $T(V, \omega)$ curve at the operating point considered (see Fig. 7.10). The lower is the controller loop which is configured for the implementation of the desired control strategy. T_{wo} , the actual wind input torque, and ω_o , the desired turbine speed, are the variable setpoints. T_w , the net torque input to the turbine, and $1/R_1$, the load step determined by the controller, are the plant inputs.

The lambda optimum control strategy uses the actual windspeed signal to derive the optimum value for the turbine speed, i.e. the controller setpoint, ω_o . This creates an extra loop linking T_{wo} and ω_o . In fact T_{wo} , or V , can be regarded as a disturbance input from the point of view of the controller loop. And the loop linking T_{wo} and ω_o is actually a feedforward loop where the feedforward compensator is simply a static gain λ_o/R .

We can now look at the system in a global manner pointing out that the control principle used combines feedback and feedforward control methods. This combined action of feedforward and feedback control can make for the large magnitude disturbances on the uncontrolled wind input [7.17.].

7.4.4.2. Simplifying assumptions

For the sake of completeness the full block diagram also

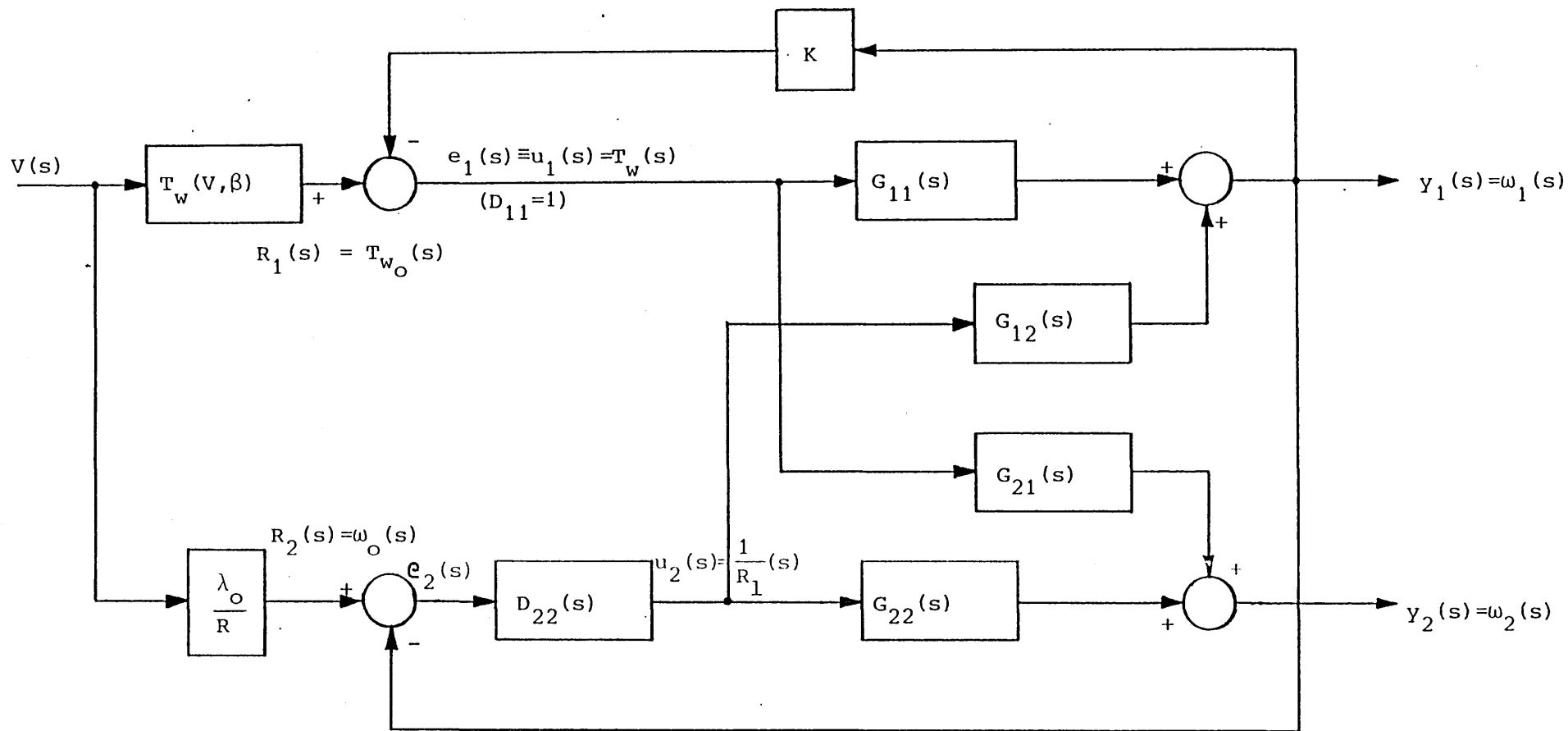


Fig. 7.4 - Complete closed-loop multivariable representation.

includes the transfer function $T_w(V, \beta)$ between the wind speed and the wind produced torque. The study of the system dynamic behaviour will however leave out this complex torque generation scheme. Blade pitch control is thus not included in the model and the blade angle is assumed to be at its optimum setting. It is assumed that the turbine aerodynamic torque T_{wo} is developed instantaneously as a function of the wind input. Since the model is built with the exclusive aim of controller design it seemed reasonable to set such boundaries.

A final modification is to leave out one output keeping only turbine speed, ω_1 . Generator speed is not of interest for our purposes and the simplified 2-input 1-output model (Fig. 7.5) will be much easier to handle. Although part of the complexity of the full representation has disappeared the basic system behaviour remains unchanged. This simplified model is believed to represent the actual WECS with some confidence. The control principles used for steady-state optimisation are included under the form of the lambda optimum strategy. The WECS dynamic behaviour will be under control of the digital compensator.

7.4.4.3. SISO approach to dynamical analysis

Inspection of the block diagram of Fig. 7.5 shows that the in-plant interaction has two forms. Changes in the controller setpoint ω_o will cause changes in the net wind input torque T_w through the feedback loop of gain K . On the other hand changes in the wind input T_{wo} will produce changes in the error signal fed into the compensator. This last form of interaction can be visualised by considering the wind torque produced speed as a load disturbance in the compensator loop. Its effect can be dealt with by an adequate tuning of the compensator.

However, the controller setpoint changes propagated through the turbine feedback loop (see dashed line in Fig. 7.5) can cause the two loops to interact in a potentially destabilising manner [7.19.]. This is due to the fact that a turbine speed change will cause a simultaneous response of the two feedback loops. Net wind input torque and load torque will change and the magnitude, signal and speed of those changes will depend on ω , V and the system transfer function coupling elements. The feedforward loop (Fig. 7.4) will certainly play a

stabilising role. It actually works as a steady-state feedforward decoupling element [7.17.]. Nevertheless there is the possibility that the compensator output is of oscillating nature and the overall system behaviour may then become oscillatory or even unstable. The gains and time constants of the two loops will determine the likelihood of this occurrence.

The actual behaviour of the multivariable process is now characterised by looking at the transfer function of the main and coupling elements. The control behaviour of two-variable control systems with p-canonical structure depends mainly on the mutual effects of the main controllers. One of them, D_{11} , is just a unitary gain and the other, D_{22} , is the digital compensator (Fig. 7.4). The static coupling factor is given by

$$K_o = K(1) = \frac{G_{12}(1) \quad G_{21}(1)}{G_{11}(1) \quad G_{22}(1)} = \frac{K_{12} \quad K_{21}}{K_{11} \quad K_{22}} \quad (7.11)$$

K_o is positive for all the possible operating points (Appendix 7). Since K_{11} , K_{12} , K_{21} and K_{22} are also always positive the main controllers, in general, reinforce each other. The main controller of the slower loop, D_{11} , is always reinforced [7.18.]. Since the controlled process is asymmetrical, because $G_{12} \neq G_{21}$ and $G_{11} \neq G_{22}$, the performance of the two-variable coupled system is better than if the loops were uncoupled, and the control performance of the slower loop is improved by the coupling [7.18.]. This is due to the fact that the action of the inherent turbine feedback loop in balancing the input wind torque is reinforced by the compensator loop.

In-plant interaction is therefore beneficial to system stability and performance. Disregarding in-plant interaction by investigating the dynamic behaviour of each of the uncoupled loops will therefore produce conservative simulation results.

Taking the small-signal linearised model the two loops will now be treated independently as SISO systems. This is done under the reasonable assumption that the settling time for the "slow" process

element $G_{11}(s)$ is quite larger than for the "fast" process element $G_{12}(s)$. In each case the system response to variations of one of the inputs will be investigated assuming the other input as a constant disturbance. The simplified block diagram is presented to emphasize the inherent turbine feedback loop (Fig. 7.6). A continuous-time model is obtained which will be used to study the system dynamic behaviour when responding to a wind input torque change ΔT_{wo} , assuming the electrical load remains constant. Alternatively, the controller loop can be made to have priority as in the block diagram of Fig. 7.7. The wind torque changes can now be regarded as load disturbances and the model is used to investigate the system response to controller setpoint changes, $\Delta \omega_o$.

7.5. CONTINUOUS TIME ANALYSIS OF THE UNCONTROLLED SYSTEM

7.5.1. Introduction

Before embarking in the design of the digital compensator some valuable insight in the system dynamic behaviour can be gained by studying its performance when uncontrolled, i.e. in an open-loop form. The study of the system dynamic response to wind torque fluctuations, assuming a constant electrical load, was therefore undertaken.

Keeping in mind that the model is linearised around each operating point, the system can be represented as in Fig. 7.5, or Fig. 7.6, assuming that $\Delta(1/R_1) = 0$. Although the system is open-loop with respect to the controller, the inherent turbine feedback loop shows some form of in-built control property. For a system loaded with a constant electrical torque this feedback loop will allow stable operation within a certain range, although such a mode of operation results in a low energy yield [1.3.]. The system dynamic behaviour is now investigated for the three chosen operating points. The main purpose of this analysis is to determine the system stability limits and gain margins for typical operating regions if no controller is implemented or if the controller is slow.

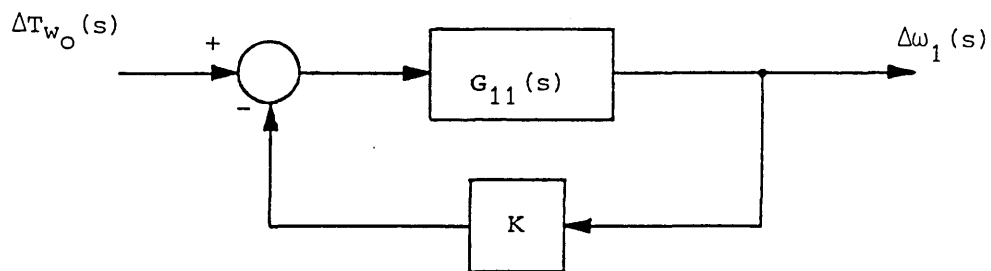
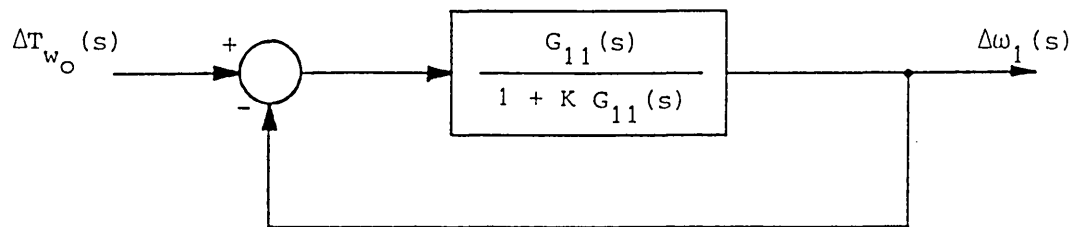


Fig. 7.6 - SISO model - inherent turbine feedback loop.

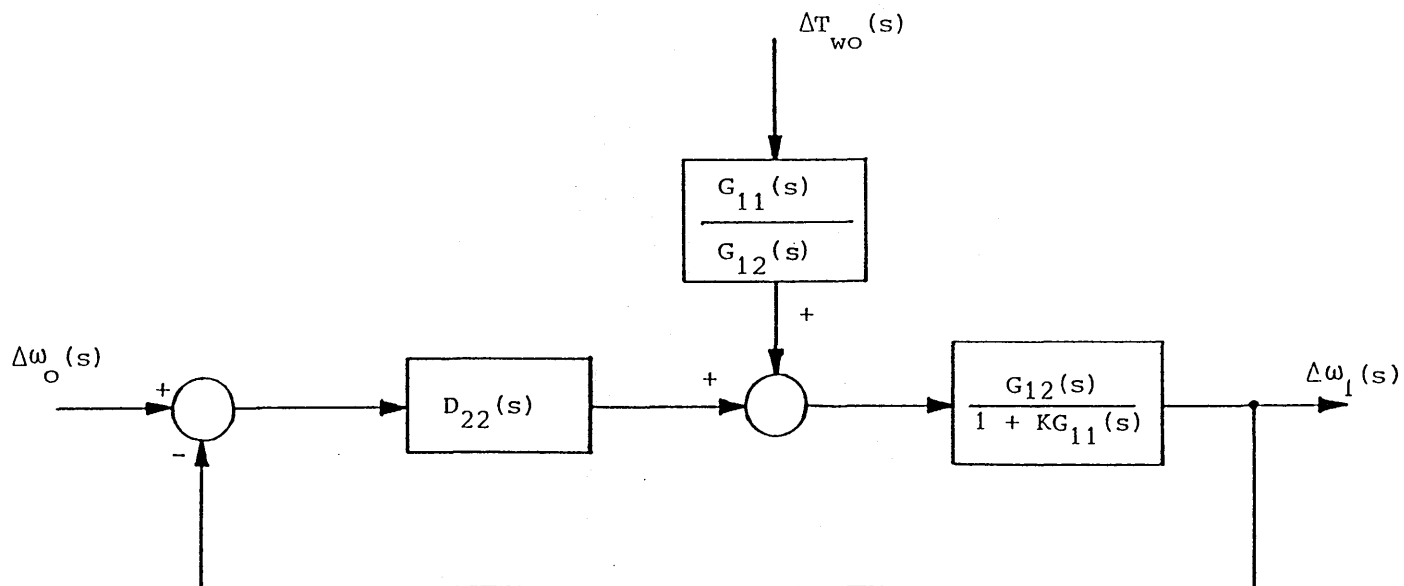


Fig. 7.7 - SISO model - controller loop.

7.5.2. S-Plane root loci

Given

$$G(s) = \frac{s^3 b_3 + s^2 b_2 + s b_1}{s^4 + s^3 a_3 + s^2 a_2 + s a_1 + a_0} \quad (7.12)$$

the root-locus will be the plot for varying K's of the characteristic equation

$$1 + K G_{11}(s) = 0 \quad (7.13a)$$

i.e.

$$s^4 + (a_3 + K b_3)s^3 + (a_2 + K b_2)s^2 + (a_1 + K b_1)s + a_0 = 0 \quad (7.13b)$$

For each operating point the coefficients of $G_{11}(s)$ are calculated with the system parameters linearized around that operating point. K represents the feedback from the speed to the torque deviation at that operating point (Fig. 7.6). The plots of Figs. 7.8 and 7.9 show the loci of the system poles when K changes from 0 to $\pm \infty$. Given the shape of the $T(\omega, V)$ curves the values of interest for K fall in the interval $[0, \pm 2]$. From the root-locus plots it can be seen that for positive values of K the system will never become unstable, but for very low negative values one of the loci crosses the imaginary axis (Fig. 7.9). The shape of the loci is similar for all the operating points considered, although the range of the excursions about the s-plane changes due to the changes in the system parameters. The value of K for which the system becomes unstable depends on the operating point. From Fig. 7.10, where the dashed line represents the optimum power locus that the load line must follow, we can obtain the values of K at the different operating points.

7.5.3. Stability Limits

The value of K for which the root-loci cross the imaginary axis is accurately determined by applying the Routh-Hurwitz stability

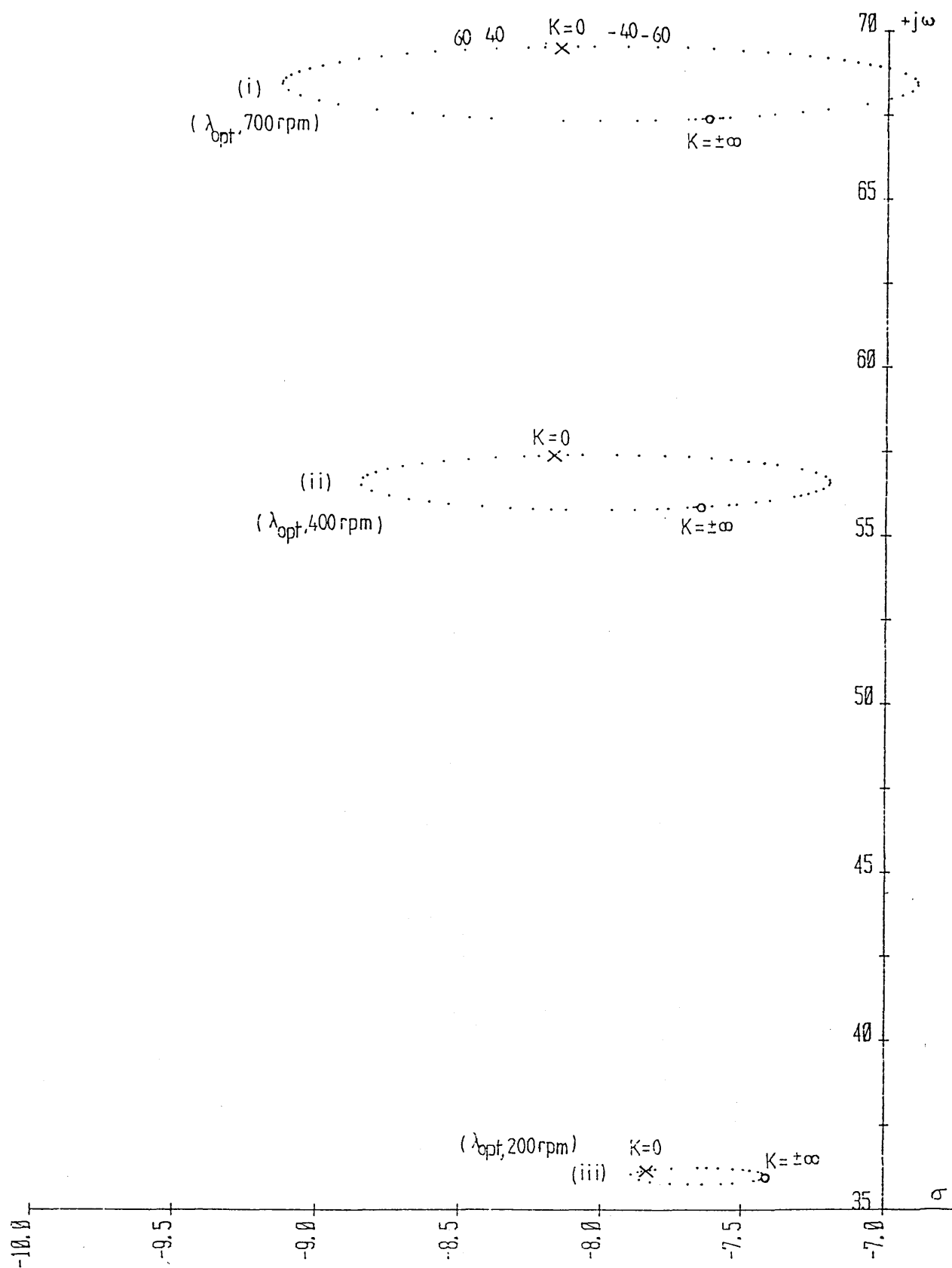


Fig. 7.8 - S-plane root locus for the open-loop WECS - "mechanical poles".

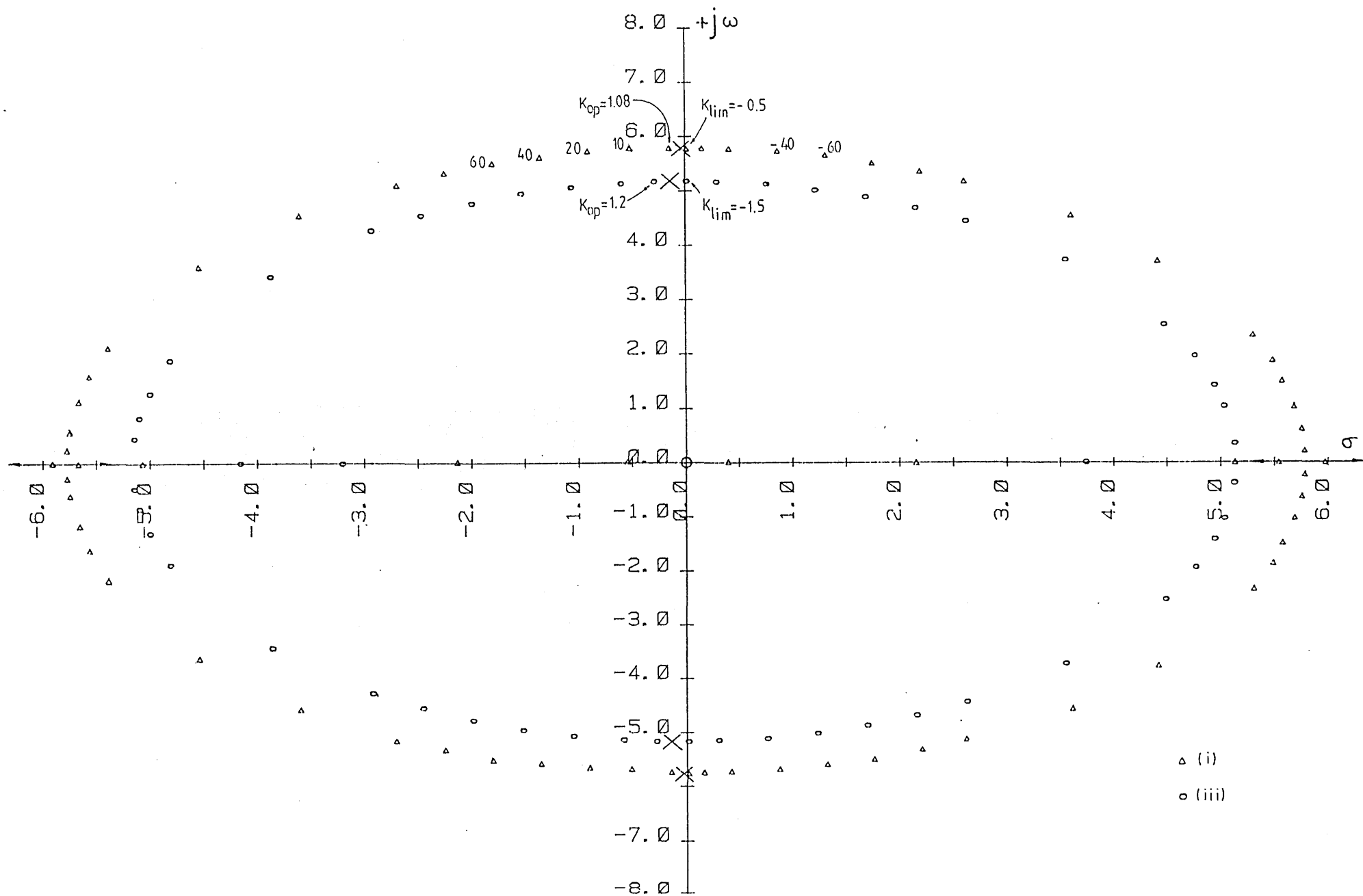


Fig. 7.9 - S-plane root locus of the open-loop WECS, "electrical poles".
The loci for operating points (i) and (ii) are practically superimposed.

criterion to the given characteristic equation. The limit values of K for the three operating points considered were found to be :

$$K_{lim}(\lambda_{opt}, 700 \text{ rpm}) = -0.47$$

$$K_{lim}(\lambda_{opt}, 400 \text{ rpm}) = -0.52$$

$$K_{lim}(\lambda_{opt}, 200 \text{ rpm}) = -1.59$$

A number of such points shown dotted in Fig. 7.10 define the stability boundary. The $T(\omega, V)$ curves were obtained from the turbine $C_q(\lambda)$ curves (Appendix 5). This agrees with results presented in [1.3.] and [7.22.], however a different approach has been used here to determine the stability boundary.

For each operating point the maximum excursion on the $T(\omega, V)$ curve caused by a perturbation on the input torque without loss of stability is dictated by K_{op} and K_{lim} , as in Fig. 7.10. Since all the values for K_{lim} are negative, although very low, we can safely assume that stability is assured if the operating point is kept always on the right hand side of the $T(\omega, V)$ curve. This means that, while implementing the lambda optimum strategy, the controller has to determine the dynamic response of the closed-loop system so that excursions beyond that stability limit are prevented. If the system is loaded as to track the optimum power locus with reasonable accuracy and speed of response, stability will be always assured. A properly tuned and fast controller will be able to cope with large amplitude wind changes acting on a high tip-speed ratio wind turbine.

7.5.4. Transient Response

From Figs. 7.8 and 7.9, where the pole pattern of the uncontrolled system is shown, we can see that the system has two pairs of complex conjugate poles. These arise from the two spring-mass-damper groups used to represent the mechanical rotational system. The system response to a wind torque step will therefore be the resultant of two exponentially damped sinusoidals attributable to the two complex conjugate pole pairs. Although the evaluation of this response is

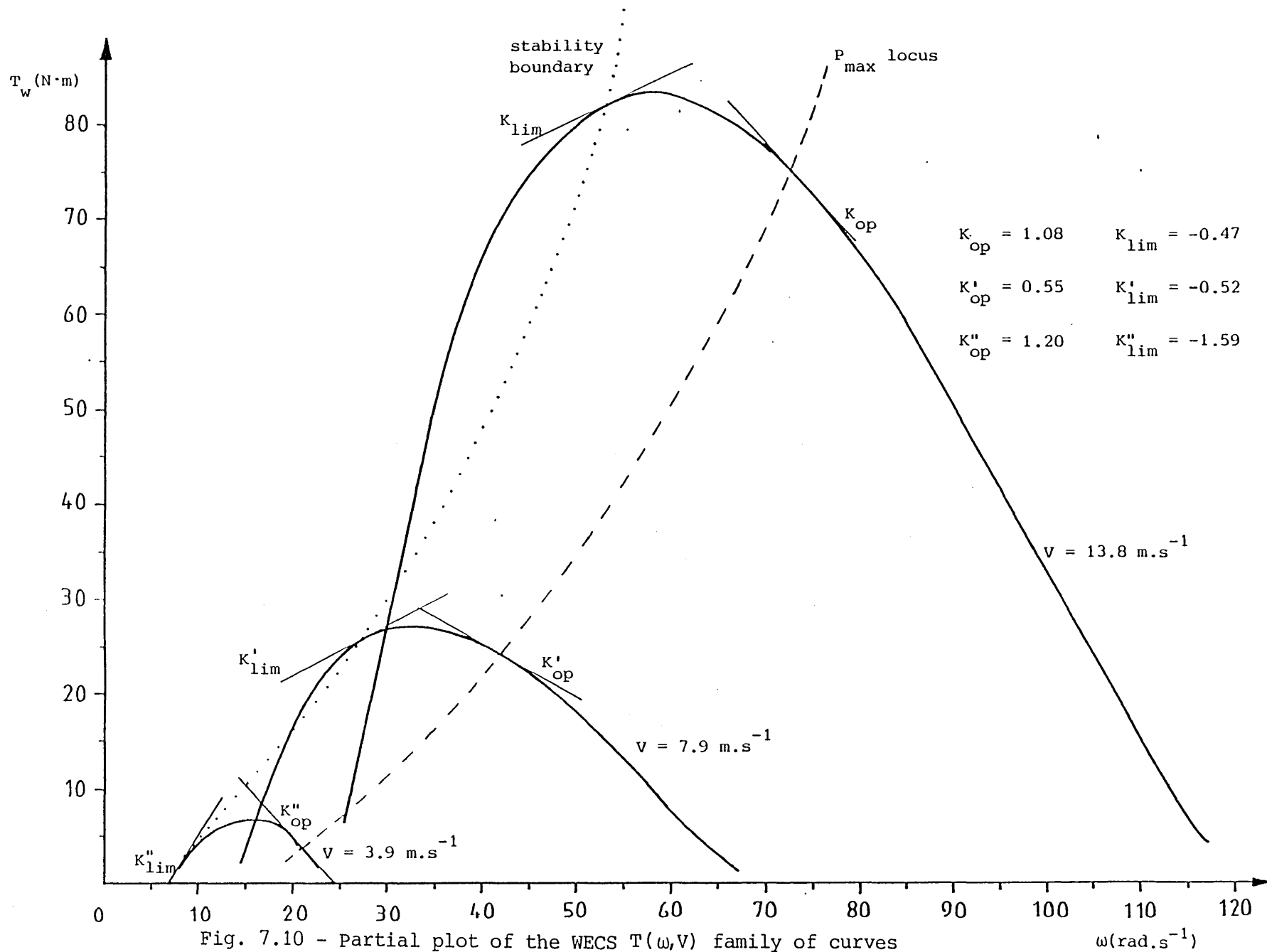


Fig. 7.10 - Partial plot of the WECS $T(\omega, V)$ family of curves

complex an approximate method can be used in this case since one of the pole pairs has a much higher time constant than the other. The pole pair attributable to the electrical spring-mass-damper group will in fact dominate the system response because its time constant is between 30 and 90 times larger than the time constant of the other pole pair. A second order approximation can then be used to evaluate the system step response.

Given the pole location an underdamped oscillatory response is to be expected. The approximation of settling time using the envelope of the decaying step response [7.20.] was used to find the system transient response, which is presented in Appendix 7 for the three operating points under consideration. Response was found highly oscillatory with an overshoot between 85 and 90% and settling time between 2 and 6 seconds. For larger values of K_{op} at the operating point larger damping and thus better responses are obtained. This happens because when the feedback gain increases the poles move away from the imaginary axis. Increase in the value of K is the consequence of a positive wind step (Fig. 7.10). For the same electrical torque the operating point will always move towards the region of bigger K 's and thus far from the unstable region. When the wind input decreases, with the electrical torque remaining unchanged, the operating point moves towards smaller values of K and therefore the system damping decreases. If the operating point corresponding to K_{lim} is attained the wind turbine will stall.

The transient response is dominated by the value of damping at the particular operating point which can be controlled by relocating the dominant poles. The closed-loop system poles, which will dictate the response of the controlled system, are to be relocated through adequate compensator design. Satisfactory dynamic behaviour will therefore be assured while implementing the lambda optimum control strategy for steady-state optimization.

7.5.5. Sensitivity to Parameter Changes

Since a linearised model is used to investigate the system behaviour a different set of parameters is valid for each operating point. As a consequence the system poles change with the operating

point and so does the WECS dynamic behaviour. This occurs both for the open-loop (fixed load) system and for the closed-loop (load controlled) system.

The type of compensator and the tuning methods used will determine the sensitivity of the closed-loop system to changes in the physical parameters. A good insight in the problem can still be gained at this stage by looking at the open-loop system s-plane eigenvalue migration with parameter variation. Only S_1 , S_2 and B_2 depend on the operating point. However, since none of the estimated parameter values offers 100% confidence, all the parameters were considered. The plots of Fig. 7.11 and 7.12 show how the open-loop system eigenvalues move when each of the parameters is changed by ± 10 , 20 and 50%.

The plots show that:

- the eigenvalues move towards the imaginary axis for any inertia increase and any damping decrease;
- the eigenvalues will move towards an higher frequency of natural response for any stiffness increase or any inertia decrease;
- each pair of complex conjugate poles is more sensitive to changes in a certain set of parameters;
- the most critical parameter changes are increase in J_1 and decrease in B_2 .

Generator damping presents no problem as this will be taken care of by the compensator. However problems may be encountered by the increased inertia of the larger rotor planned for the WECS.

We can conclude that any small error in the determination of the physical parameters is unlikely to be a cause of trouble.

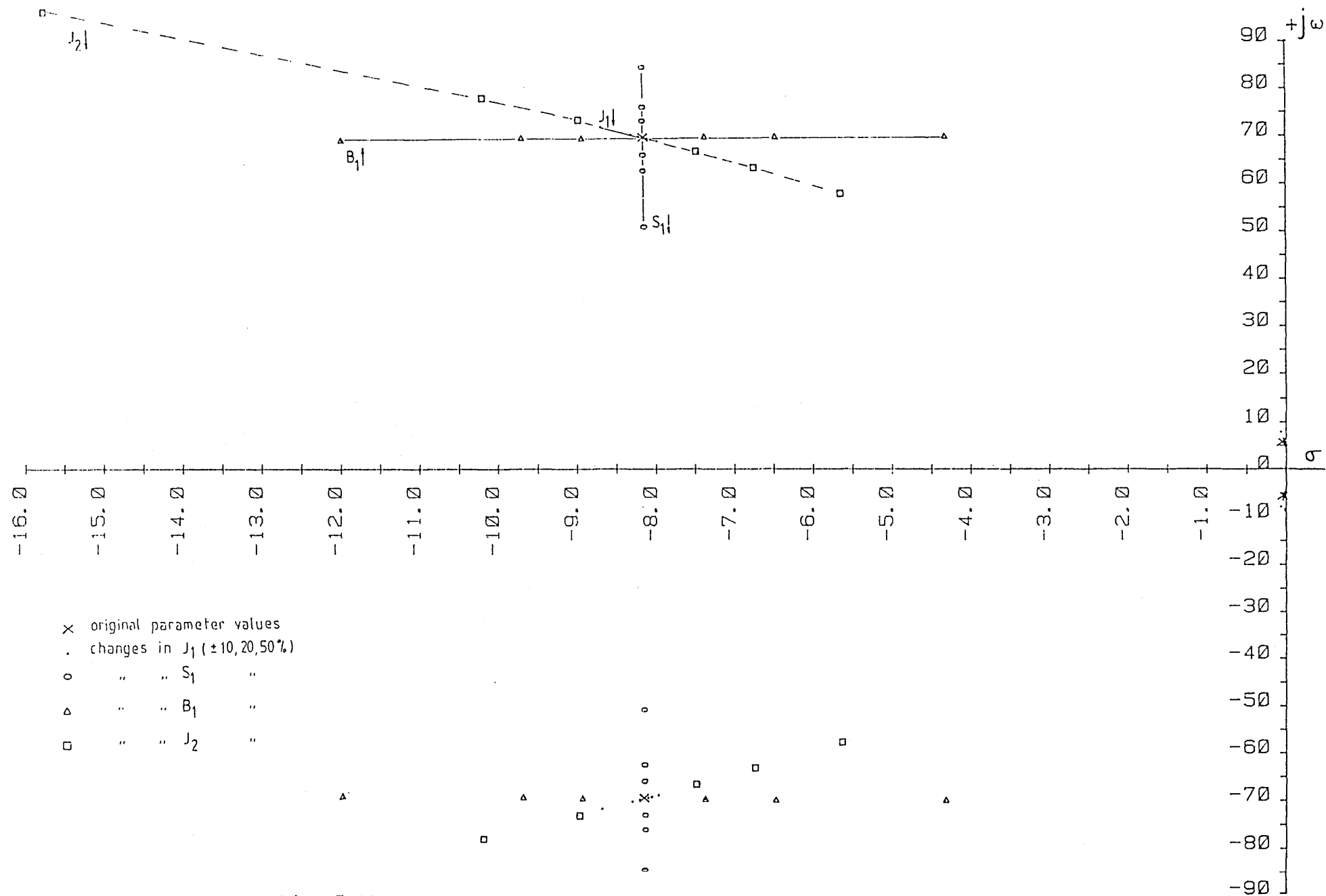


Fig. 7.11 - WECS eigenvalue migration with parameter variation - "mechanical poles".

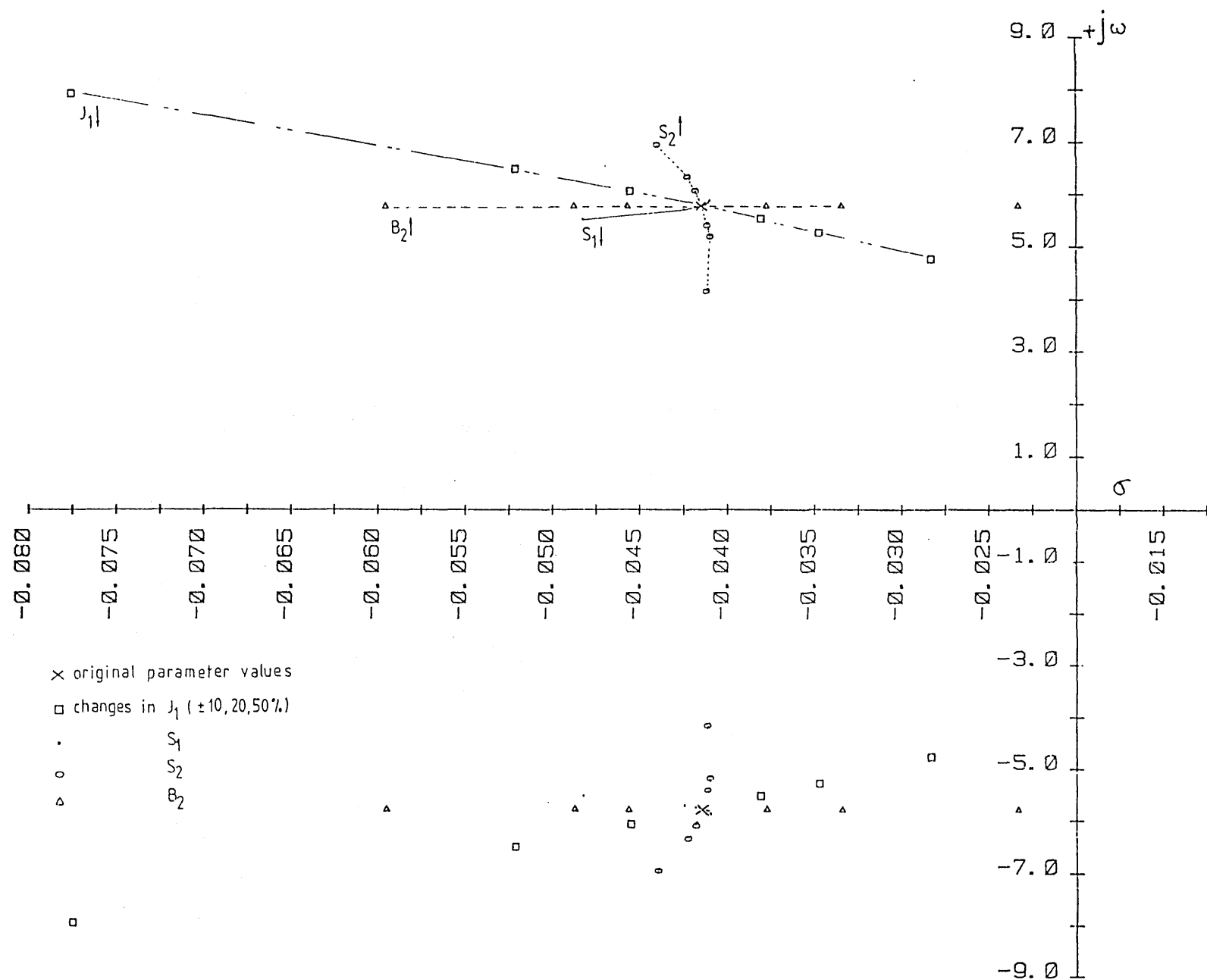


Fig. 7.12 - WECS eigenvalue migration with changes in the physically-based model - "electrical" poles.

7.6. SYSTEM DISCRETIZATION AND Z-PLANE ANALYSIS

7.6.1. Introduction

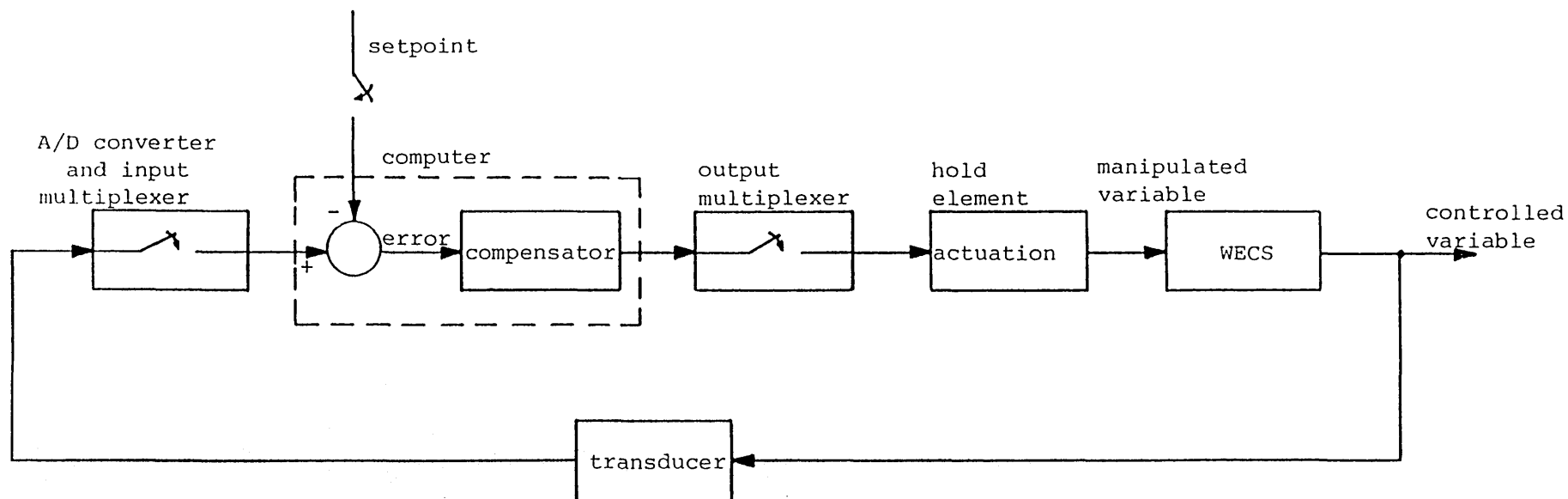
The block diagram in Fig. 7.7 is the SISO model used in the compensator design. The plant transfer function from the point of view of the controller is

$$G_P(s) = \frac{G_{12}(s)}{1 + K G_{11}(s)} \quad (7.14)$$

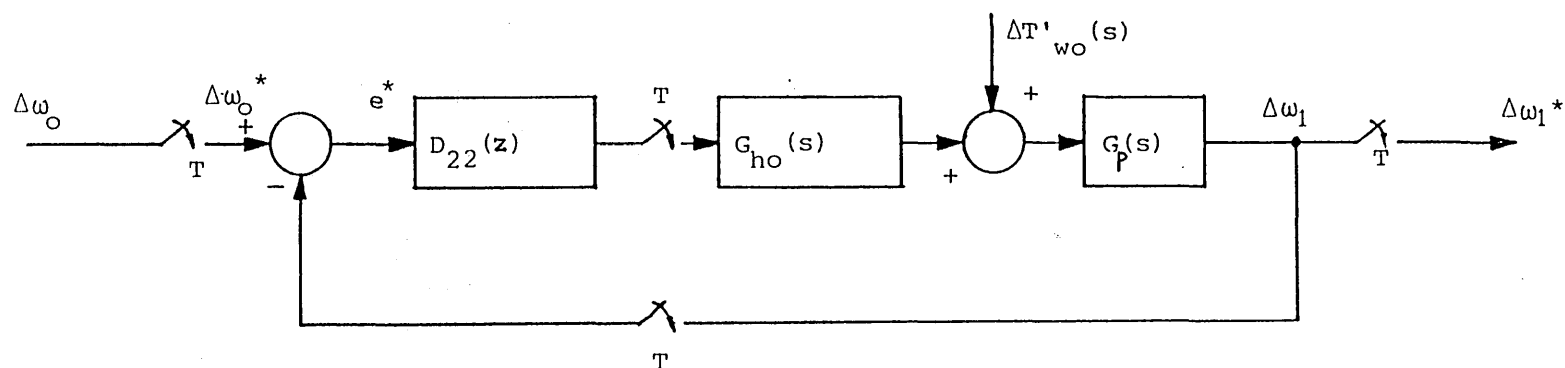
Since a digital compensator will be used the correct system representation will be given by the block diagram of the sampled-data system (Fig. 7.13). Fig. 7.13a shows the hardware involved in the implementation of the DDC loop. Fig. 7.13b is the block diagram representation of the resulting sample-data system. The ADC is modelled as an ideal sampler, inputting the variables ω_1 and ω_0 . Sampling ω_0 is equivalent to sampling V since just a constant gain separates these two variables. The compensator output is not fed into the continuous plant through a DAC, but the electrical actuation scheme used to control the stepped load behaves exactly like a 6-bit DAC. The 6-channel PIA binary output word controlling the solid-state switches is therefore modelled as an ideal sampler followed by a zero-order-hold (ZOH). This matter will be further discussed later. In Fig. 7.13b the asterisk denotes a discretized variable and $G_{ho}(s)$ is the transfer function of the sample-and-hold device. The samplers are assumed to be synchronized.

7.6.2. System Discretization

The discrete equivalent of the continuous plant is now obtained by z transformation of $G(s)$. This method, sometimes called hold equivalence method or impulse-invariant response method, was chosen as it imposes fewer constraints on the choice of the sampling rate [7.5.]. The continuous elements of the loop, i.e. the process and the ZOH device, are transformed to the z plane. The basic steps of this transformation are now outlined. Intermediate algebra is given



a) - Hardware implementation of the DDC loop.



b) - Block diagram representation.

Fig. 7.13 - Controlled WECS as a sample-data system.

in Appendix 7, as well as the values for the partial fraction constants and the polynomial coefficients.

If $G'(s)$ denotes the continuous-time transfer function of the process plus zero-order-hold then

$$G'(z) = Z \left\{ G_{ho}(s) \quad G_p(s) \right\} \quad (7.15)$$

where the transfer function of the zero-order-hold is

$$G_{ho}(s) = \frac{1 - e^{-sT}}{s} \quad (7.16)$$

and $G_p(s)$ is given by Eqn. 7.14.

It follows that

$$G'(z) = Z \left\{ \frac{1 - e^{-sT}}{s} G_p(s) \right\} = \frac{z-1}{z} Z \left\{ \frac{G_p(s)}{s} \right\} \quad (7.17)$$

Putting $k=1.0$, as for the nominal operating point, it is possible to describe $G_p(s)/s$ as a quotient of two polynomials whose coefficients are easily obtainable:

$$\frac{G_p(s)}{s} = \frac{b'_2 s^2 + b'_1 s}{s^5 + (a_3 + b_3)s^4 + (a_2 + b_2)s^3 + (a_1 + b_1)s^2 + a_0 s} \quad (7.18)$$

The next step is to expand the Laplace transform expression via the partial fraction expansion method:

$$\frac{G_p(s)}{s} = K_{WT} \frac{s(s - \alpha)}{s(s - \alpha')(s - \beta')(s - \gamma')(s - \delta')} \quad (7.19)$$

cancelling the zero and the pole at $s=0$ we obtain

$$\frac{G_p(s)}{s} = K_{wT} \left[\frac{K_1}{s - \alpha'} + \frac{K_2}{s - \beta'} + \frac{K_3}{s - \gamma'} + \frac{K_4}{s - \delta'} \right] \quad (7.20)$$

where K_1 to K_4 are the partial fraction expansion coefficients and K_{wT} the inherent system gain. Substituting $G_p(s)/s$ in equation 7.17 the z-transform tables can be used to obtain

$$G'(z) = K_{wT}(z-1) \frac{A_3 z^3 + A_2 z^2 + A_1 z + A_0}{(z - e^{\alpha'T})(z - e^{\beta'T})(z - e^{\gamma'T})(z - e^{\delta'T})} \quad (7.21)$$

Looking at $G'(z)$ we can see that the z-plane poles come from direct mapping of the s-plane poles. The zeros are found through the numerator polynomial. It should be noted that in the z-plane both the poles and the zeros depend on the sampling time T . This critical issue will be discussed later.

7.6.3. Digital Compensator Design

7.6.3.1. Choice of compensator structure

Our next task is to synthesize the transfer function of the digital compensator, $D_{22}(z)$, or simply $D(z)$. This is a two stage task : first the compensator structure is chosen and then its parameters are tuned. Although the problem of compensator design has been restricted when deciding the global approach to system modelling and dynamical analysis (section 7.3.2.) there is still more than one alternative.

The problem can be placed in a rather general manner, as with the so-called direct design method (Ragazzini & Franklin), by specifying a desired transfer function for the closed-loop system

$$H(z) = \frac{D(z) G'(z)}{1 + D(z) G'(z)} \quad (7.22)$$

The required compensator transfer function is then given by:

$$D(z) = \frac{1}{G'(z)} \cdot \frac{H(z)}{1 - H(z)} \quad (7.23)$$

$H(z)$ can be chosen according to different performance criteria, but it must always assure that $D(z)$ is realizable and stable [7.5.,7.17.]. A number of methods can be used to design $D(z)$ as to satisfy specific performance criteria.

A simple but efficient alternative was preferred : to decide on the compensator structure and to achieve the desired system response by adjusting its parameters. The selection of the compensator form is now justified.

One of the most widely used controllers in the design of continuous-data control systems is the so-called three term or PID controller. Its discrete equivalent is also commonly used. PID algorithms perform very well in practice for a wide class of processes. They are robust and difficult to improve significantly unless considerable effort is expended [7.6.,7.17.-7.19.,7.21.]. Parameter tuning can be done in a different number of ways depending on how the closed-loop system performance is specified. Finally its simplicity brings two further advantages : smaller computation time than for more complex algorithms [7.19.] and good understanding of its action on the physical system. A digital PID was therefore the chosen compensator structure.

7.6.3.2. Closed-loop system transfer function

The transfer function of a digital PID is, assuming rectangular integration [7.3.], illustrated in Fig. 7.14. We can write:

$$D(z) = \frac{(T K_p + K_d)z^2 + (-T K_p - 2 K_d + T^2 K_i)z + K_d}{T z(z - 1)} \quad (7.24)$$

where K_p, K_i, K_d are the gains of the proportional, integral and derivative terms and T is the sampling time. Clearly the closed-loop system response can be adjusted by tuning the four parameters - T and the three gains. The system closed-loop transfer function will be given by equation 7.22, i.e.,

$$H(z) = \frac{D(z) G'(z)}{1 + D(z) G'(z)} = \frac{P(z)}{Q(z)}$$

with $G'(z)$ and $D(z)$ given by equations 7.21 and 7.24 respectively. $P(z)$ is a 4th order and $Q(z)$ a 5th order polynomial. The closed-loop system has therefore 4 zeros and 5 poles whose position on the z -plane depend on the values of T, K_p, K_i and K_d . The coefficients of $P(z)$ and $Q(z)$ are complicated functions of the tuning parameters (see Appendix 7).

7.6.3.3. Tuning of compensator parameters

The discrete-time closed-loop transfer function is now used in the digital compensator tuning procedure. Such tuning method is allowed on the grounds that $H(z)$ was derived with the help of a well identified plant model.

7.6.3.3.1. Pole relocation using root-locus methods

In a first stage the z -plane root-locus method was used to relocate system closed-loop poles in desired positions within the unit circle. The loci will be the plots of the roots of the characteristic equation

$$1 + D(z) \cdot H(z) = 0 \quad (7.25)$$

in the unit circle. This requires that Eqn. 7.25 is solved for changing T, K_p, K_i and K_d . Given the order of the system there is no standard procedure to do this relocation as to obtain a certain desired behaviour. Nevertheless an approximate procedure can be used

assuming the system response essentially dictated by a pair of dominant poles. Z-plane specifications in terms of desired steady-state and transient accuracy produced an "acceptable region" where the system dominant poles are to be relocated [7.4.]. Steady-state accuracy is defined in terms of step response for the reasons stated further on. The error to a step input is calculated by the final value theorem. Transient accuracy is defined in terms of % overshoot, rise time and settling time. Finally constraints on the sampling rate are stated. These requirements lead to the well-defined region showed in Figs. 7.15 to 7.17. The four dense lines defining the region boundaries are obtained as follows:

- settling time to $\pm 1\%$ final value (f.v.)

$$t_s \leq 20 T \rightarrow r \leq 0.8$$

- % overshoot

$$\text{o.s.} \leq 30\% \rightarrow \zeta \geq 0.4$$

- rise time from 0.1 to 0.9 of step height

$$t_r \leq 8 T \rightarrow \omega_n \geq \frac{2.5}{8 T} \approx \frac{\pi}{10 T}$$

- T as to ensure 6 samples per cycle of osc. $\rightarrow \theta < 60^\circ$

The relocation of the poles within this region is in fact a trial-and-error pole placement scheme. A computer program was written to plot the system closed-loop root loci in the z-plane for desired values of T , K_p , K_i , K_d .

The effect of the parameter variation on the pole position can give us some idea on the range of values producing adequate relocation. This is portrayed by the root-loci of Figs. 7.15 and 7.16, where each of the parameters is changed while the others are kept constant.

Considerable amount of information on the system dynamic behaviour can be gained from inspection of these loci. Good control can be exercised on the position of the "electrical" poles which can be relocated at wish. This arises from that fact that by controlling the electrical loading in a dynamic manner the electrical stiffness and

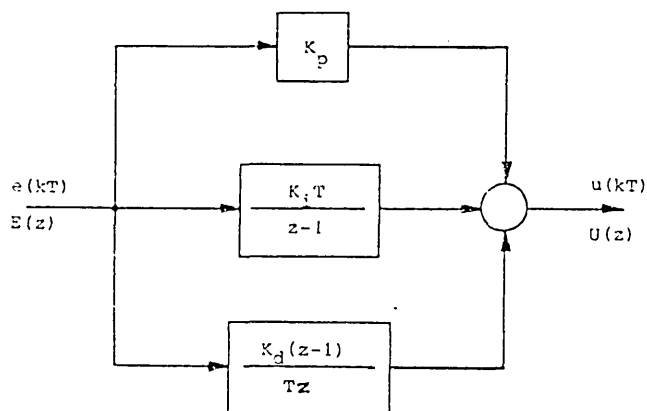


Fig. 7.14 - Transfer function of a digital PID.

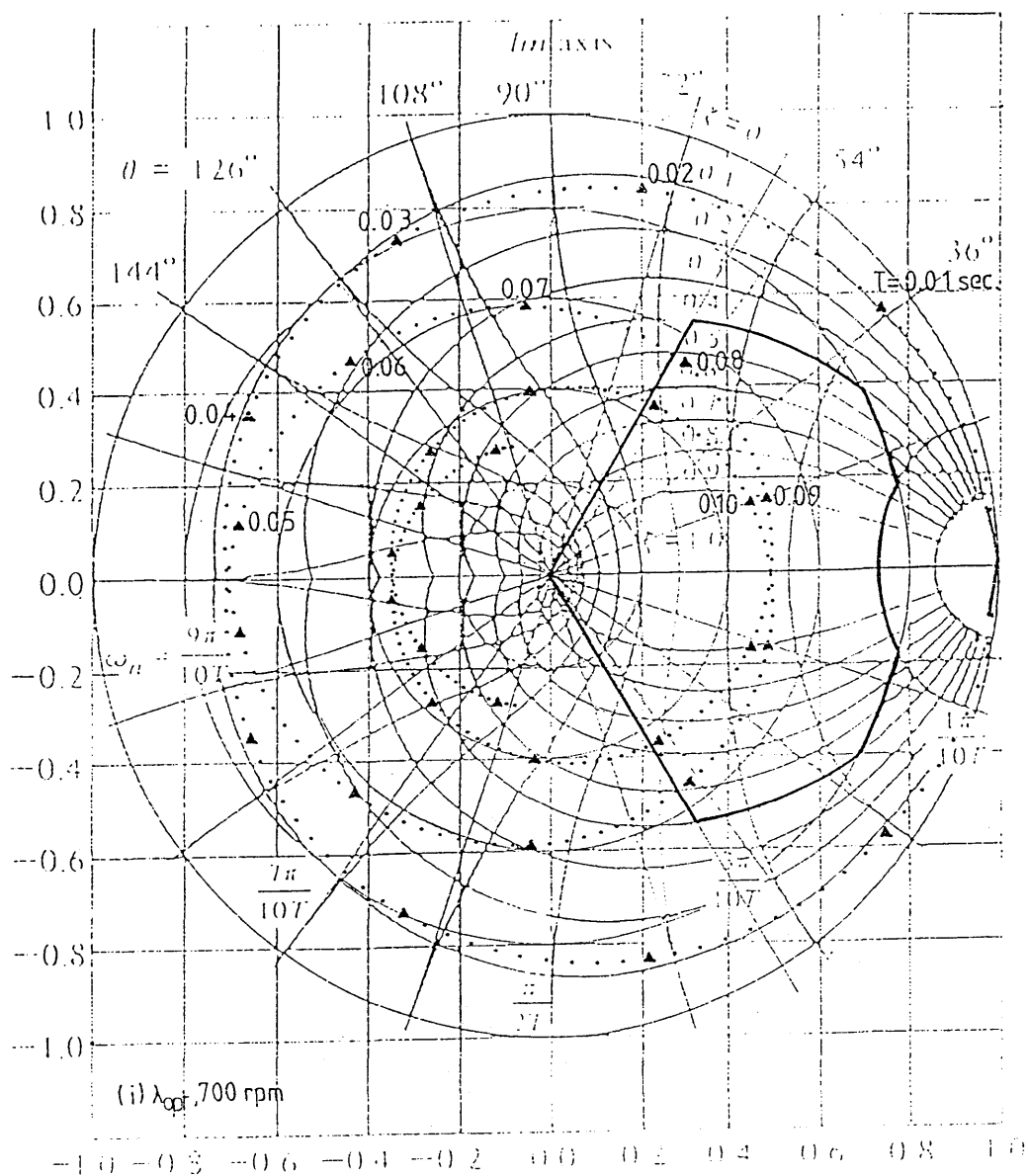
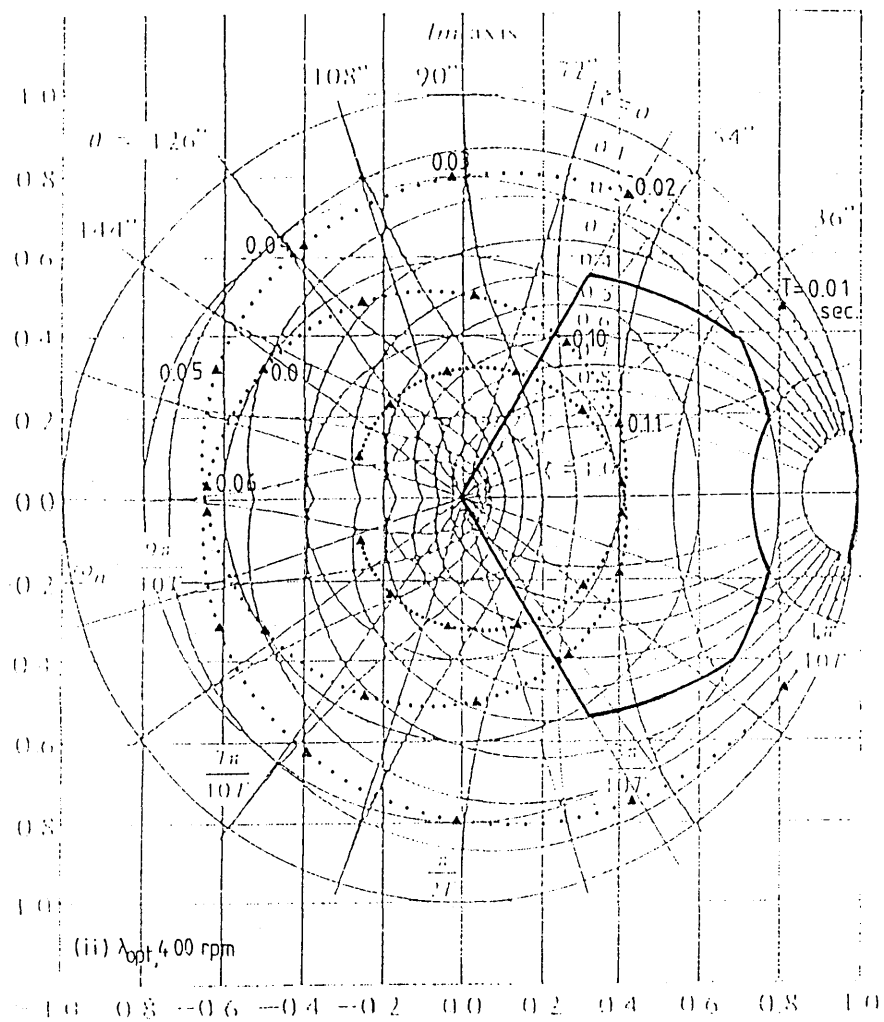
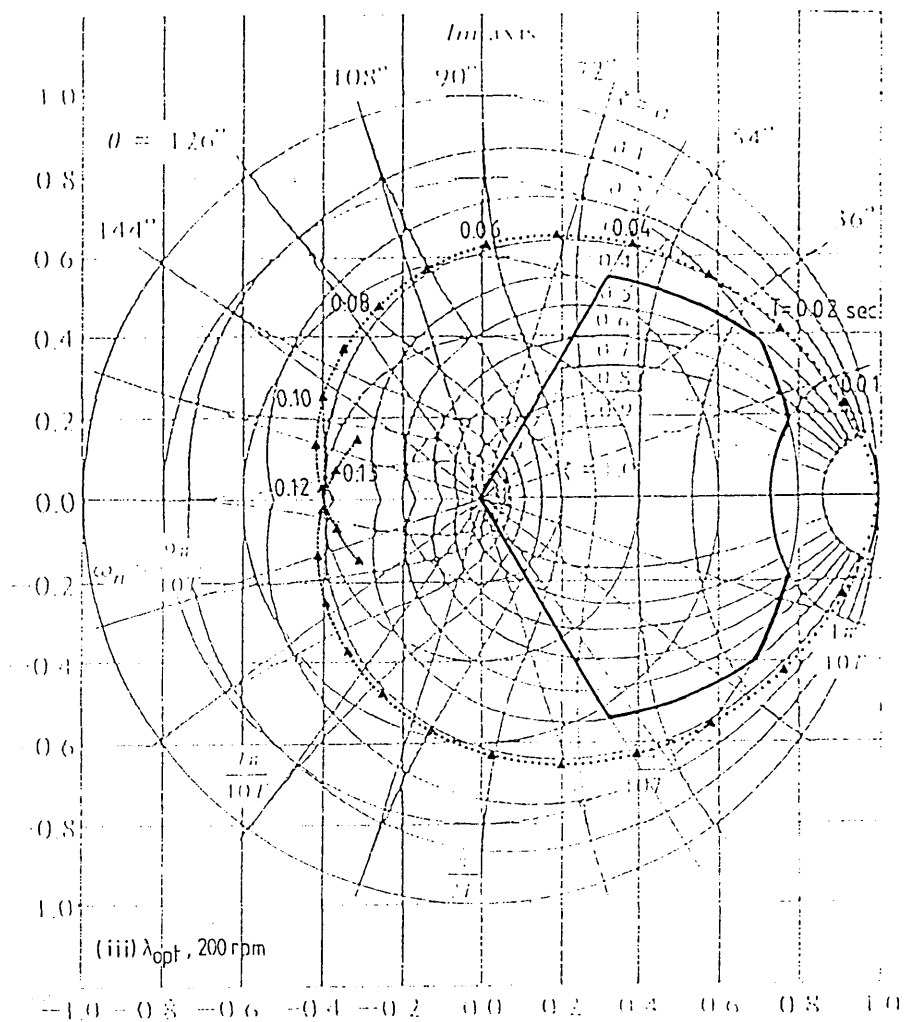


Fig. 7.15 a) - Z-plane eigenvalue migration due to changing sampling rate for λ_{opt} , 700 rpm operation.



b) $\lambda_{opt}, 400 \text{ rpm.}$



c) $\lambda_{opt}, 200 \text{ rpm.}$

Fig. 7.15 - cont.

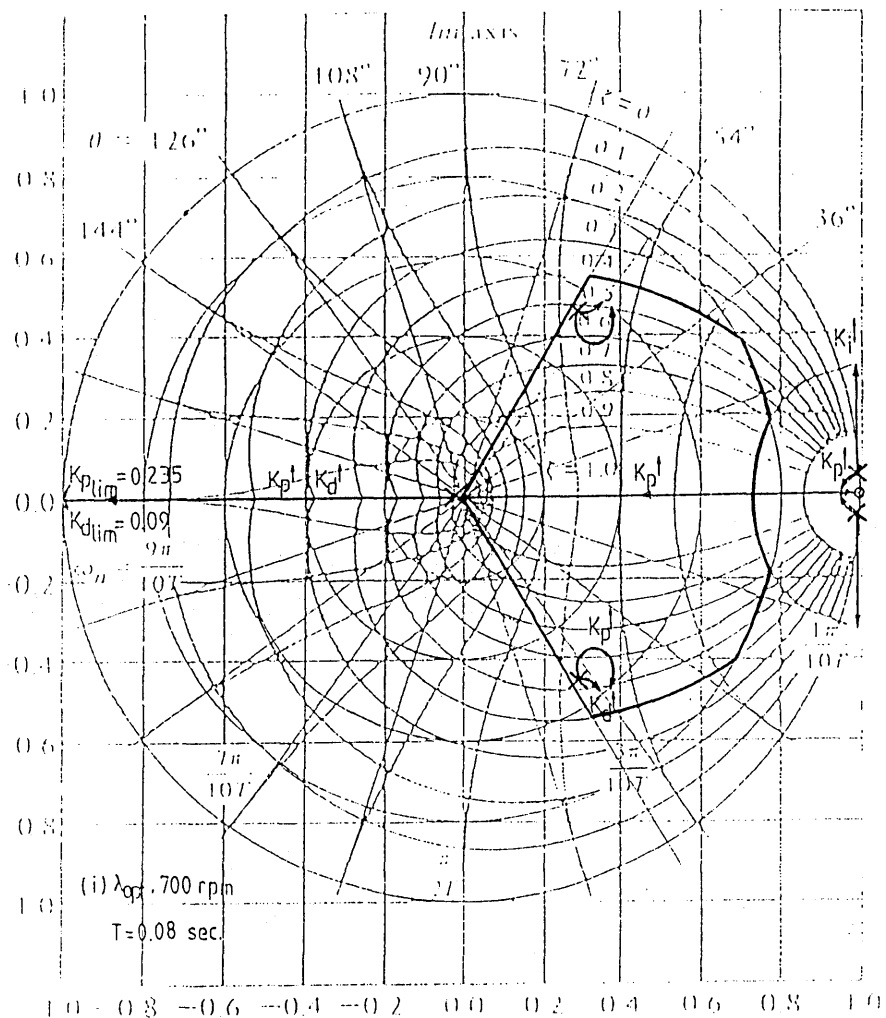


Fig. 7.16 - Z-plane root loci of the closed-loop system.

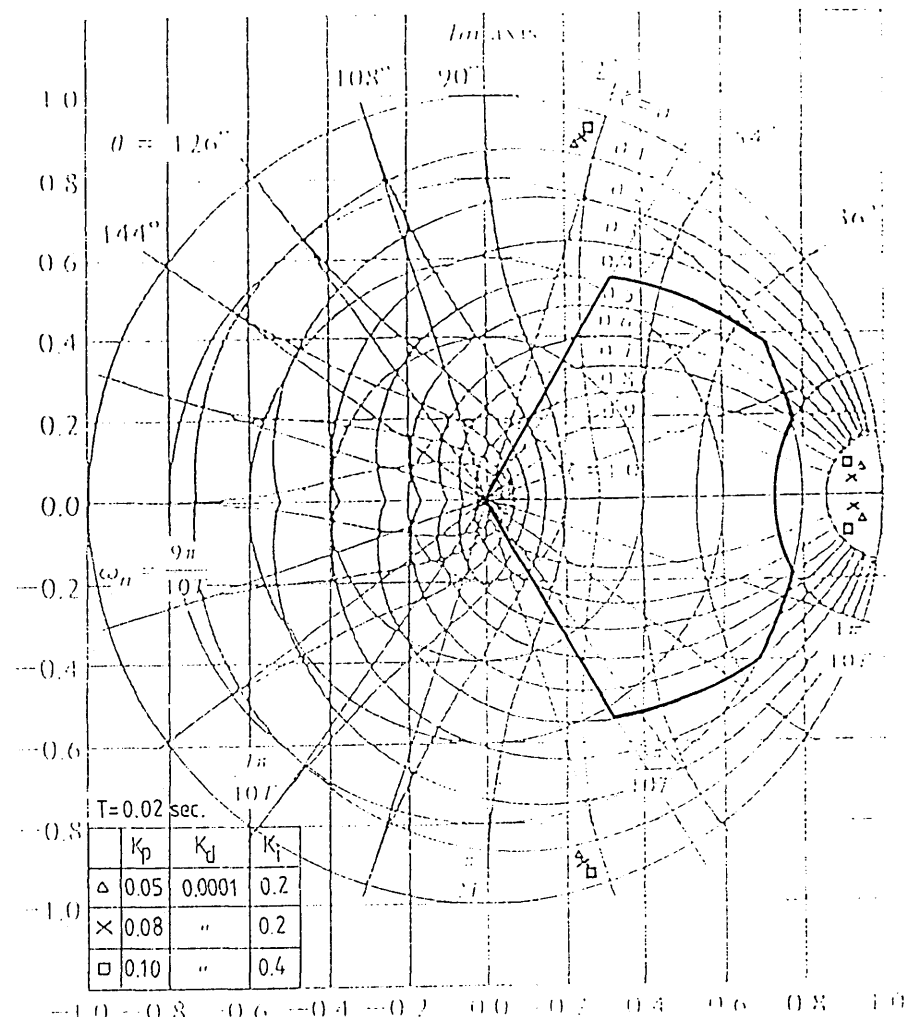
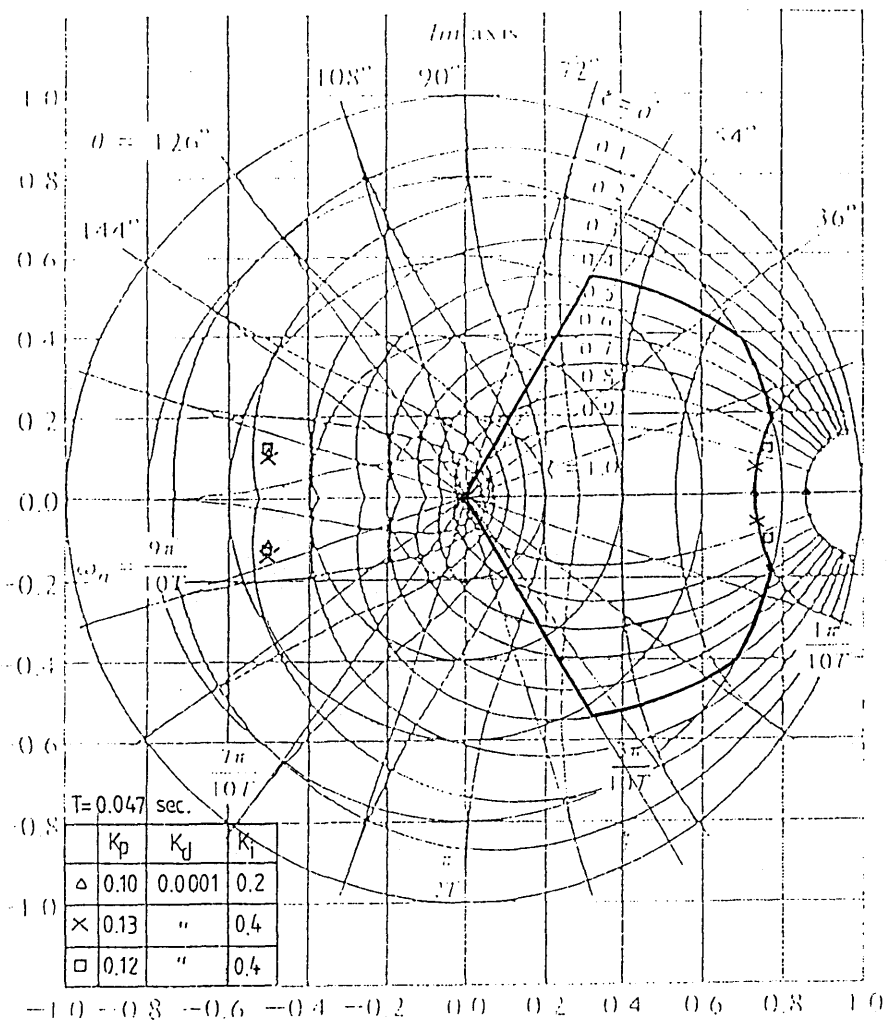
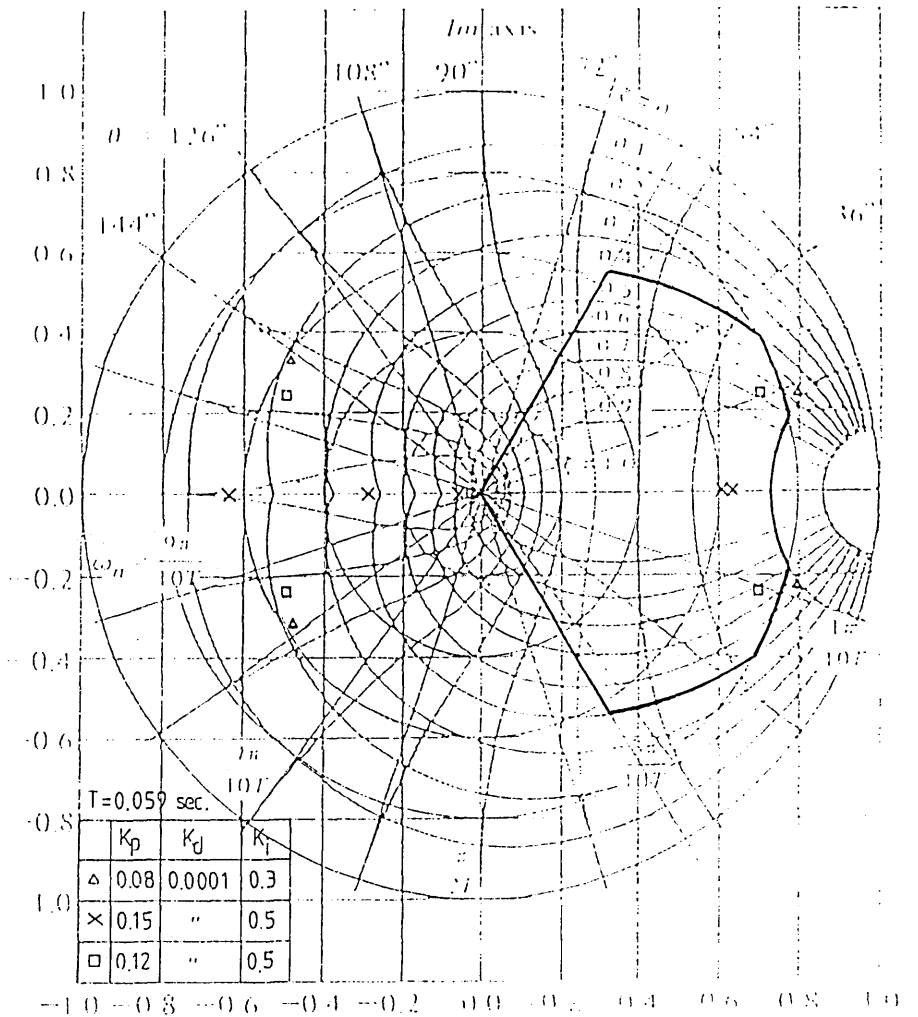


Fig. 7.17 a) - Z-plane closed loop pole location
for (λ_{opt} , 700 rpm). "Best" relocation
for $T = 0.02 \text{ sec}^S$.

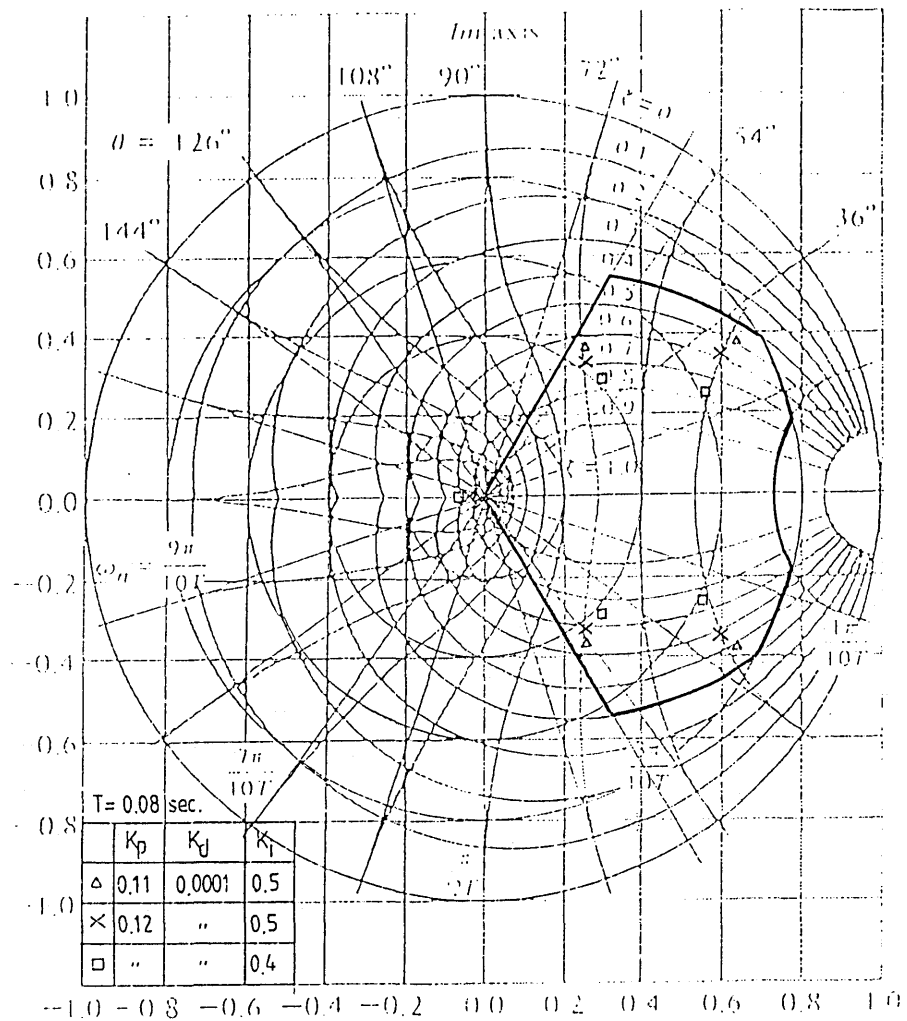


b) "Best" relocation for $T = 0.047$ sec.

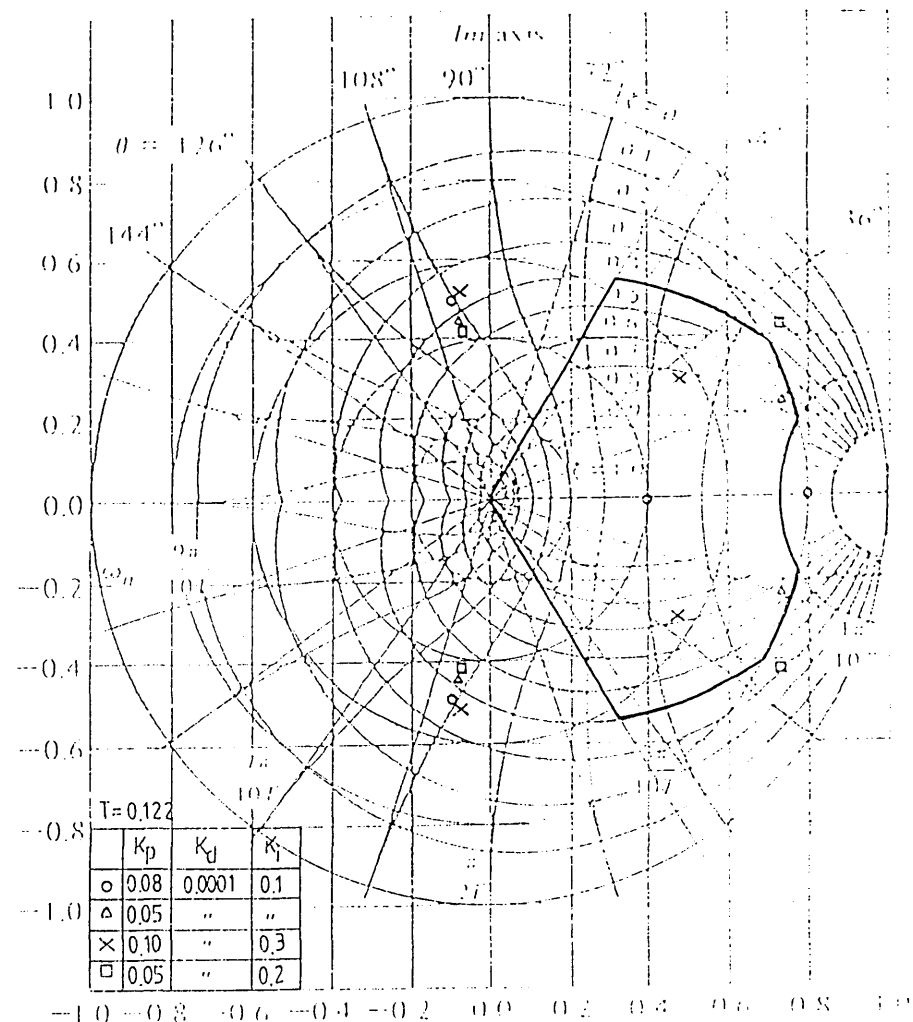


c) "Best" relocation for $T = 0.059$ sec.

Fig. 7.17 - Cont.



d) "Best" relocation for $T = 0.08$ sec.



e) "Best" relocation for $T = 0.122$ sec.

Fig. 7.17 - cont.

electrical damping are effectively being controlled. The position of the "mechanical" poles is less affected by changes in the compensator gains. This is because changes in the electrical stiffness and damping have a small effect on the "mechanical" modes (Fig. 7.11). The shape of the loci of these poles, the amplitude of their excursions, and, most importantly, where they move to when the gains are increased depend strongly on the sampling time. Since little control can be exercised through the compensator gains, the loci are almost solely determined by the initial pole-zero pattern and therefore by the choice of T .

A range of acceptable values of T can be readily estimated from the plots in Fig. 7.15. And loci as the one in Fig. 7.16 can be used to determine the set of gain values that relocates the dominant, perhaps all, poles within the pre-specified region.

7.6.3.3.2. Fine parameter tuning with step/ramp response methods

The specified performance criteria, expressed as a pole relocation condition, can be met by adjusting the sampling time and the compensator gains. Since, as stated above, this is an approximate method fine tuning of all the parameters is desirable. This is accomplished in a second design stage where the response of the closed-loop system to standard test inputs is investigated. Priority is given to system step response since the actual system input, the controller setpoint, will undergo step changes. Moreover, a step is the most severe of the available test inputs. The system response to a unit step input can be calculated with the so-called method of the residues for inversion of z -transforms [7.6.].

If

$$G'(z) = \frac{D(z) H(z)}{1 + D(z) H(z)} = \frac{(z - \alpha)(z - \beta)(z - \gamma)(z - \delta)}{(z - \alpha')(z - \beta')(z - \gamma')(z - \delta')(z - \zeta')} \quad (7.26)$$

is the system closed-loop transfer function, then the response to a unit step input is given at discrete times $T, 2T, 3T, \dots$, by

$$y(kT) = \sum \text{residues of } \left[z^{(k-1)} \cdot Y(z) \right], \text{ for } k=0,1,2,\dots$$

at the poles of $Y(z)$ (7.27)

where

$$Y(z) = G'(z) \cdot \frac{z}{z-1} \quad (7.28)$$

A computer program was written to evaluate and plot the step response for a given set of gains.

Fine tuning of the parameters around sets of values that had conveniently relocated the closed-loop poles in the z -plane is now done by simple inspection of the system response to a unit step. Performance is respecified still in terms of rise time, overshoot, settling time and steady-state error. The values presented in Figs. 7.18, 7.19 and 7.21 for the rise time, t_r , settling time, t_s , etc., were based on specifications presented in Appendix 7. Commonly used integral criteria are more exact, since these are based on the entire response, but more difficult to evaluate. This fine tuning led to the plots in Fig. 7.18 which correspond to z -plane closed-loop pole locations in Fig. 7.17. All the cases presented assume the system nominal operating point.

The controller setpoint changes are always step changes, with the height of the step determined by the turbine speed and wind speed values. However alternative implementations may prove useful if, for example, load changes are to be counteracted. In this case to ramp the setpoint to the new value is likely to achieve better results [7.17.]. System response to a ramp input was therefore investigated for a chosen number of cases. With $G'(z)$ given by equation 7.26 the system output for an input ramp of unitary value is

$$Y(z) = G'(z) \cdot \frac{T z}{(z-1)^2} \quad (7.29)$$

The output $Y(z)$ has a double pole at $z=1$. The method of the residues can still be used to invert $Y(z)$ but the necessary modifications will complicate the software.

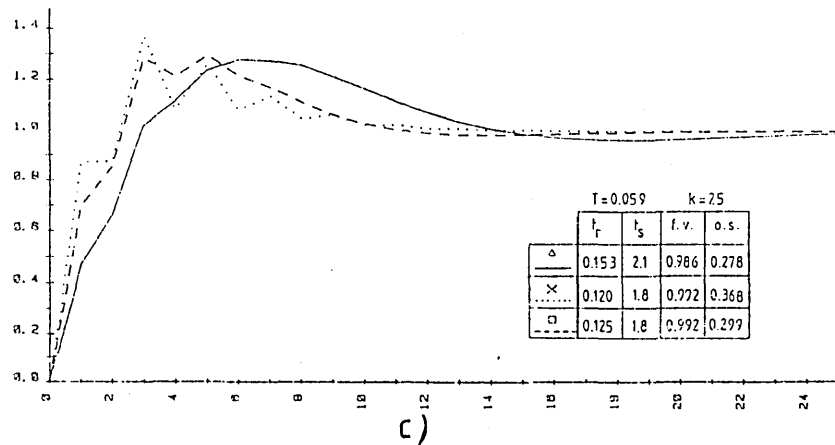
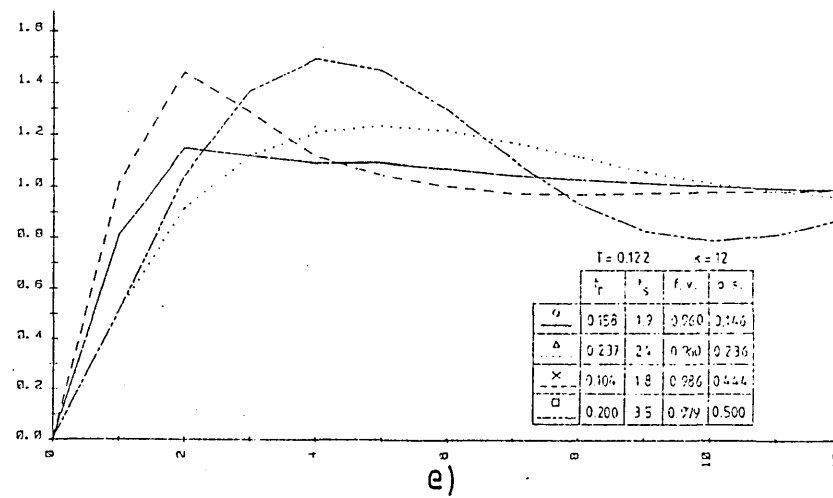
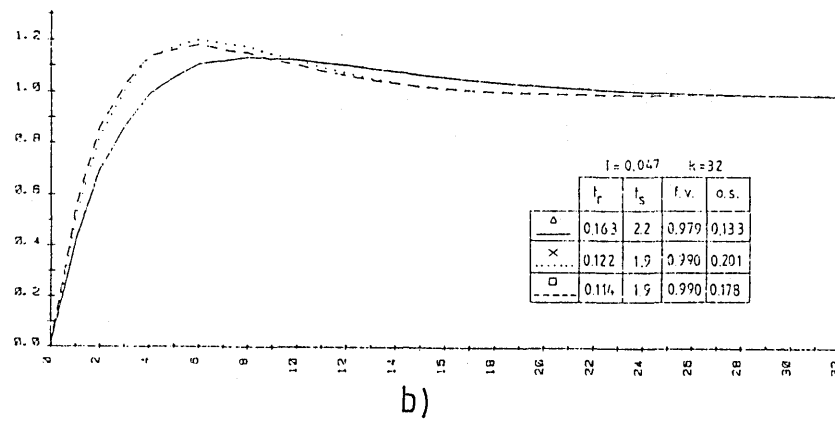
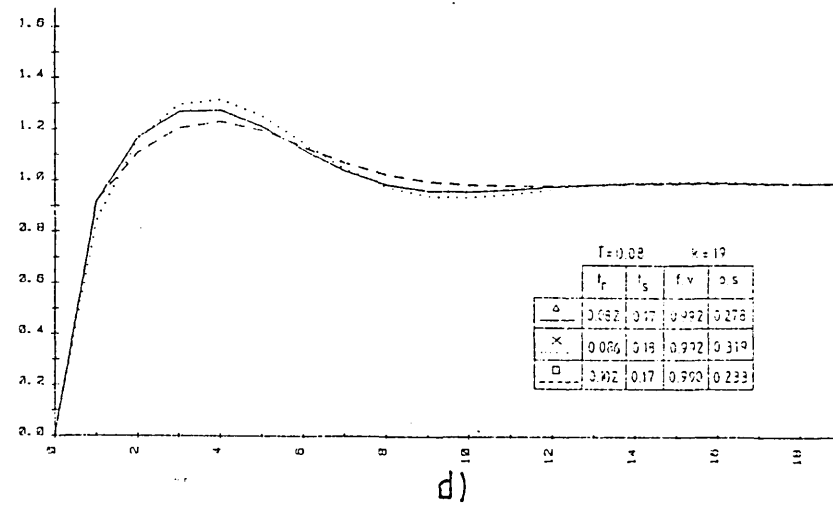
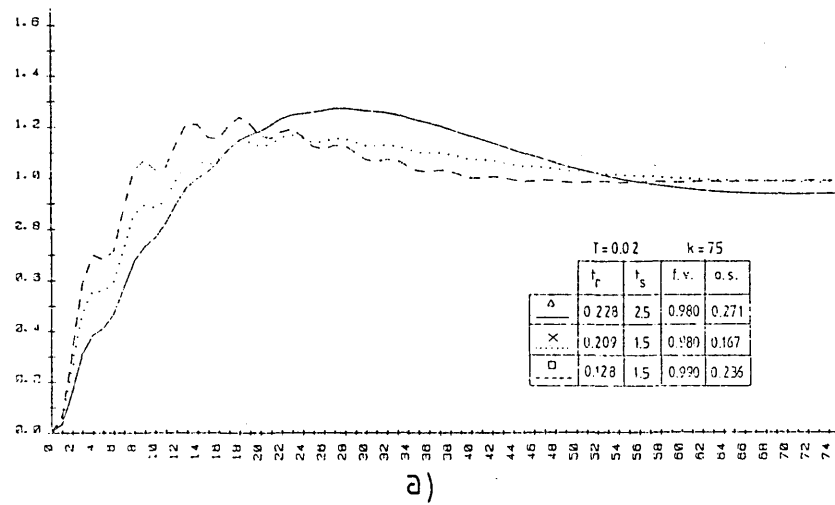


Fig. 7.18 - Step responses corresponding to the pole patterns in Fig. 7.17.

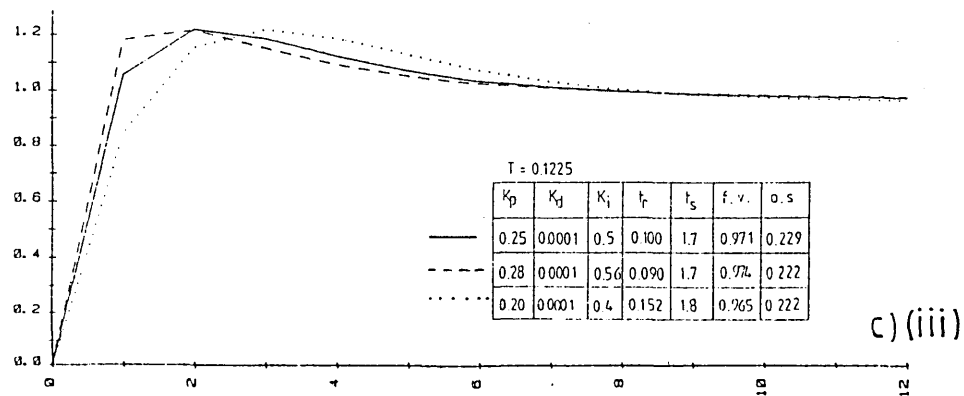
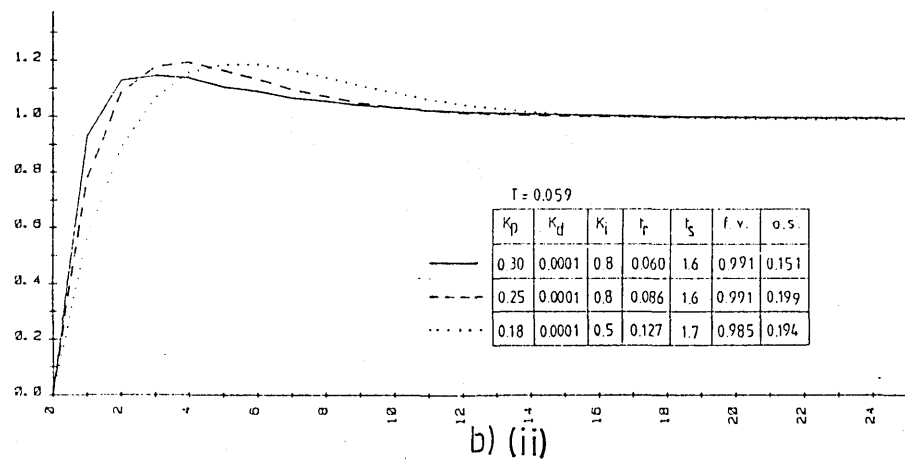
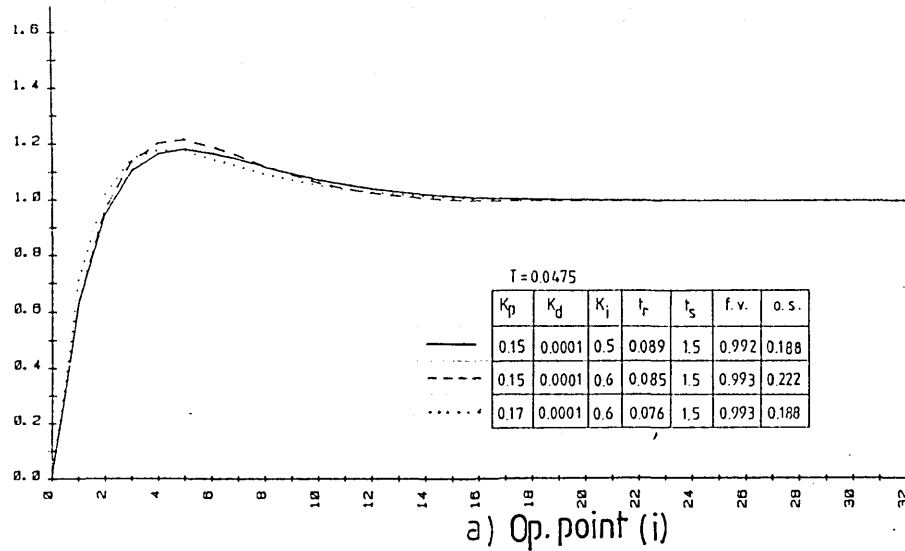


Fig. 7.19 - Step responses with "best" tuning.

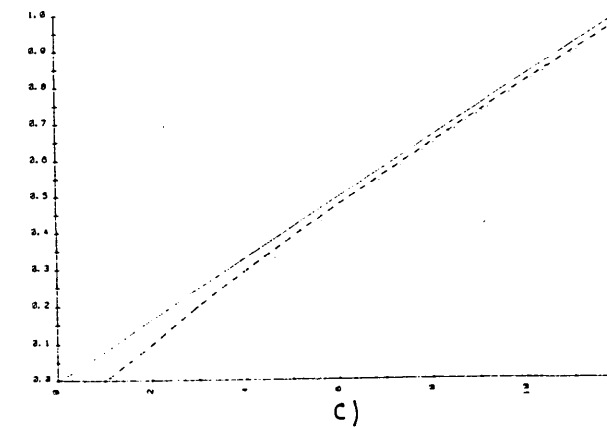
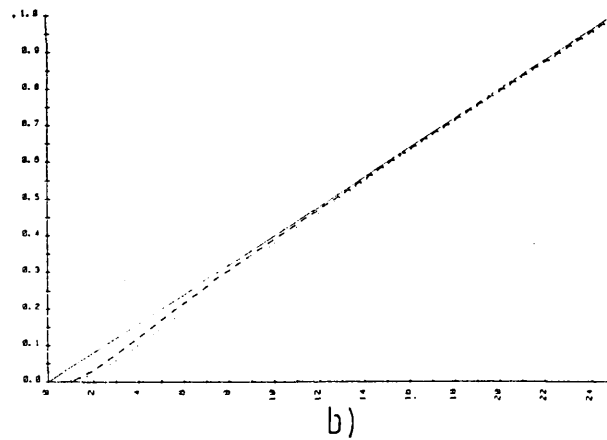
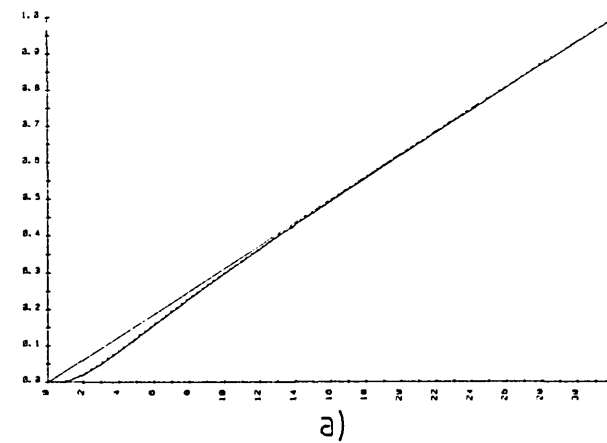


Fig. 7.20 - Response to a ramp for the tuning values in Fig. 7.19.

The so-called long or synthetic division method was therefore preferred [7.6.]. This consists on simply dividing out the two polynomials $P(z)$ and $Q(z)$ in

$$Y(z) = \frac{P(z)}{Q(z)} = \frac{b_4'' z^5 + b_3'' z^4 + b_2'' z^3 + b_1'' z^2 + b_0'' z}{z^7 + a_6'' z^6 + a_5'' z^5 + a_4'' z^4 + a_3'' z^3 + a_2'' z^2 + a_1'' z + a_0''} \quad (7.30)$$

The corresponding time function can be written directly in series form. Since the difference in order of the polynomials is 2 the first sampled value is obtained after 2 sample intervals. The time series is given by

$$Y(z) = C_2 z^{-2} + C_3 z^{-3} + C_4 z^{-4} + \dots \quad (7.31)$$

Where the coefficients of z^{-K} are determined in Appendix 7. Although the expressions for these coefficients grow very rapidly with the negative powers of z , their derivation can be mechanised by using a recursive method. A computer program was written that plots the system ramp response for a given set of gains. The plots in Fig. 7.20 show the system ramp responses for the same parameter values used in Fig. 7.19.

7.6.4. Conclusions

7.6.4.1. Optimum compensator parameters

In the above described analysis the values for one of the tuning parameters, the sampling time, was from start restricted to the range 20/200 mS. The lower limit is set by computer loading limitations (section 4.4.3.) and the upper limit from the need to sample several times in each cycle of system response. This restriction placed a first bound on the range of sampling time values to be considered.

The guidelines for relocation of the system poles (section 7.6.3.3.1.) and the analysis of the step response plots redefine

these boundaries. For the values of T for which the "mechanical" poles were outside the desirable region the system step response is usually bumpy and/or with sharp peaks no matter what gain values are used (Figs. 7.17 and 7.18). Satisfactory responses are obtained for, e.g. $T = 80$ mS (Figs. 7.17d and 7.18d) and, generally for the range between 70 and 100 mS. This is the case where not only the dominant but all the closed-loop system poles are properly relocated within the unit circle. The response can be made smooth, fast rising and settling down reasonably fast around a steady-state value close to 1. Since the system is of type 0 it will always have a non-zero steady-state error to a step input. If this is reduced by increasing K_i the overshoot increases and thus a compromise between % over-shoot and steady-state error has to be found.

Surprisingly, for $T = 47$ mS an improved performance is obtained in spite of the fact that one pair of poles is on the left hand side of the unit circle. This is due to the fact that the pair of "mechanical" poles is "clamped" to an almost fixed position close to the real axis, irrespective of "open" or "closed" loop operation and gain changes. Moreover, their residues are very small thus making their contribution to the step response almost negligible in spite of not being well damped poles. Their small contribution is therefore decoupled from that of the other pole pair which will dictate the system response.

7.6.4.2. Degradation due to system non-linear behaviour

From earlier considerations it is expected that changes in the operating point will also alter the closed-loop pole pattern and the behaviour of the controlled system. The values for compensator tuning presented in Fig. 7.18 for optimum performance at the nominal operating point will not be adequate for other operating points (Figs 7.19 and 7.20). The "best" performance for the part load operating points is given by sampling times of 59 and 122 mS, i.e. once more values that "clamp" the "mechanical" poles near the real axis (Fig. 7.15b and c). The "best" step/ramp responses for each operating point are quite similar, although performance improves when the sampling time is lowered.

On-line auto-tuning of the digital PID according to the actual WECS operating point would assure the "best" performance at each operating point. The gains could be read from a look-up table with turbine and wind speeds as entries. Such a feature would however complicate the software and considerably load the microcomputer.

It is therefore vital to investigate the extent of performance degradation when the system is not running at the operating point for which the compensator parameters have been tuned. The plots in Fig. 7.21 show the degradation in system performance due to the system non-linear behaviour. The tuning values derived for the nominal operating point seem to be the best compromise if the global behaviour is considered through these three significant load points (see Fig. 7.18c and e). This is in agreement with the general rule that a longer sampling time increases the sensitivity of a control loop to process parameter shift [7.6.]. The decision to use these tuning values is further reinforced by the fact that it is at or around the nominal operating point that a better performance is more critical. At part loads the extracted energy is much lower and fine tuning optimisation will produce marginal improvements.

On-line auto-tuning is nevertheless a possibility which deserves consideration if improvement of the control algorithm proves necessary.

7.7. IMPLEMENTATION OF THE CONTROL ALGORITHM

7.7.1. Quantization Effects

The finite word length of the computer and data converters is the cause of what is known by quantization effects. These sources of error occur at data input/output (ADC and DAC), when system parameters are stored as constants and in arithmetic operations.

7.7.1.1. Quantization in arithmetic operations

The errors generated by quantization effects are often relevant when fixed-point arithmetic is used.

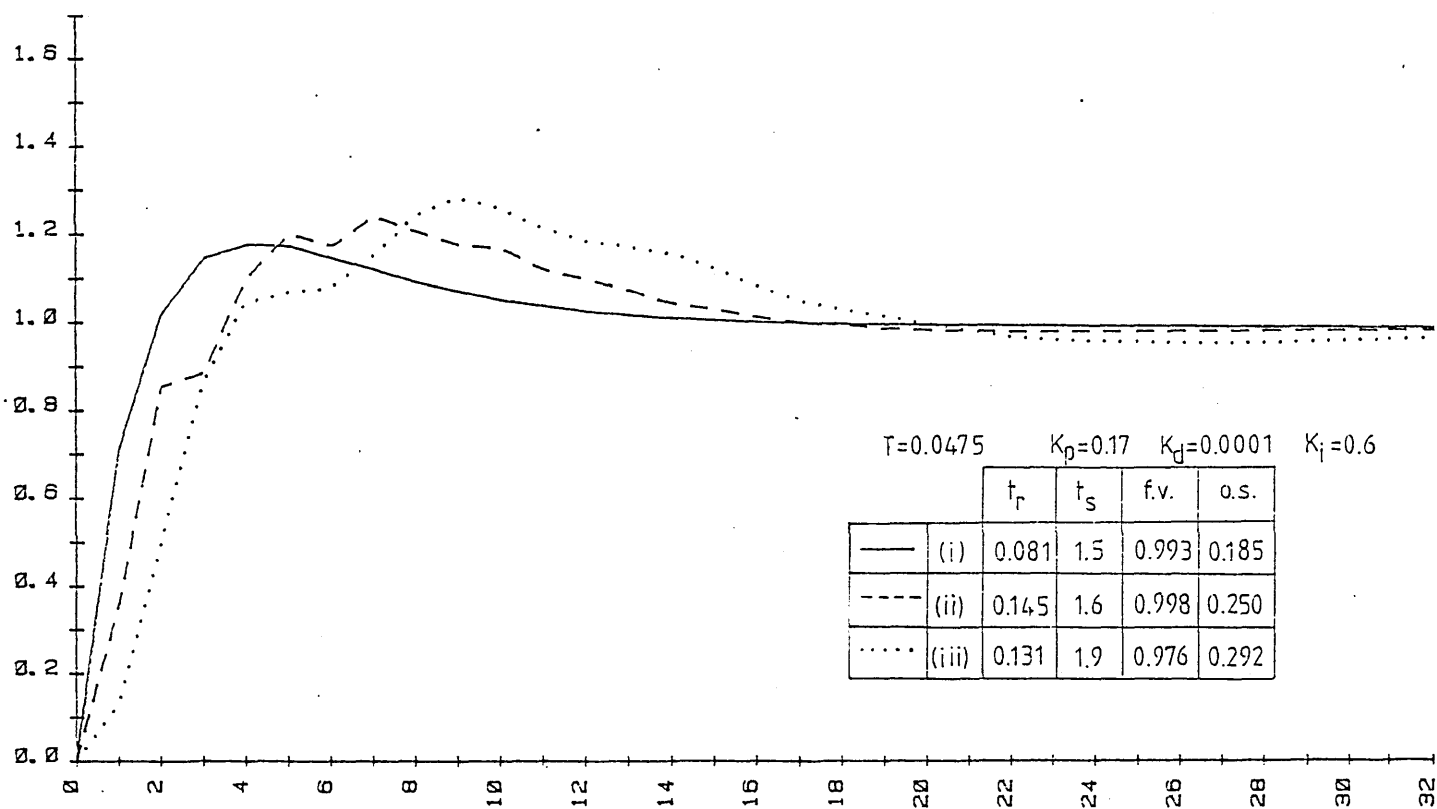
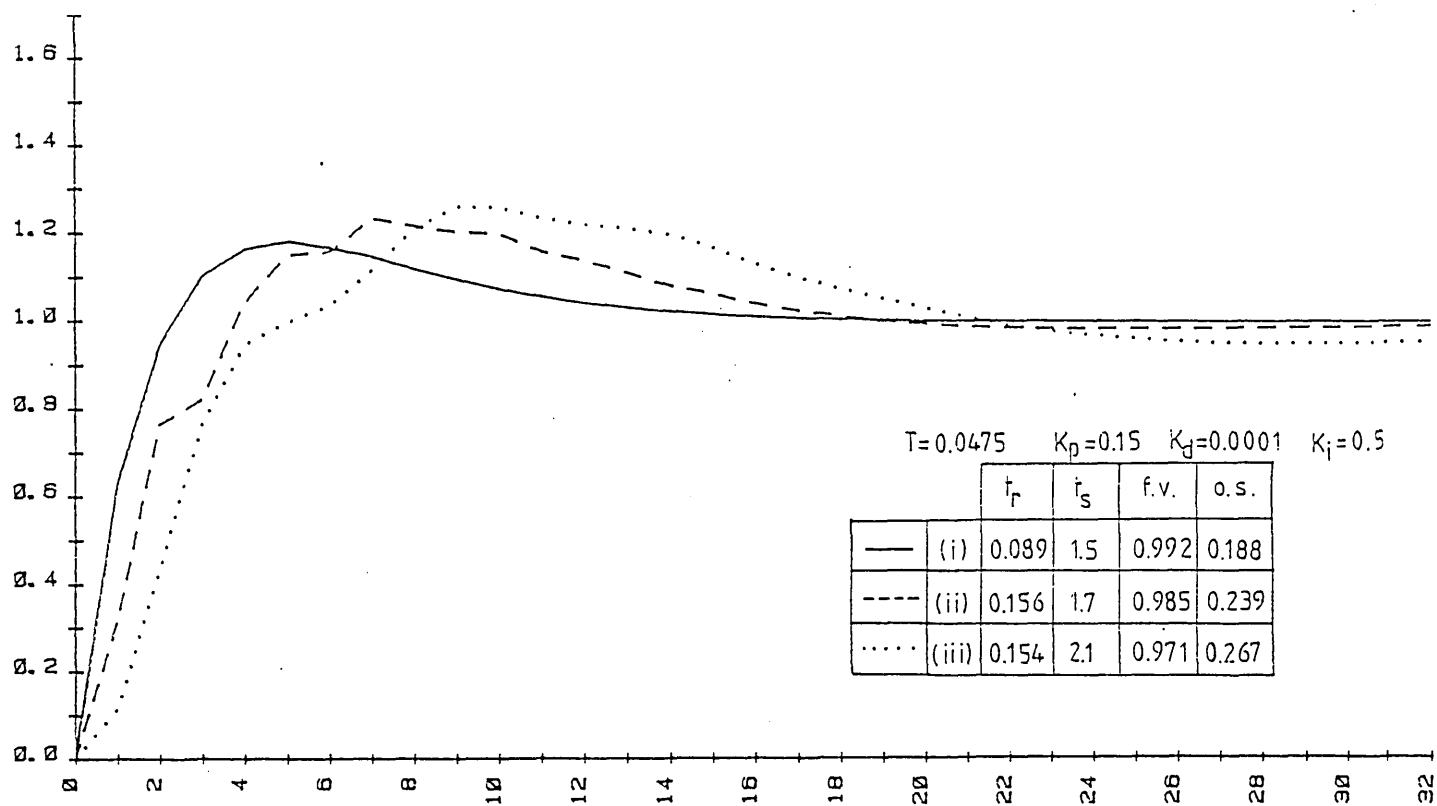


Fig. 7.21 - System degradation due to non-linear behaviour.

Step responses to the three operating points considered with optimum tuning parameters for nominal operating point (i).

Floating-point arithmetic, however, achieves sufficient accuracy to allow quantization errors to be neglected [7.6.]. Since variables in the control program are stored in the IEEE floating-point representation and manipulated by floating-point routines it seems reasonable to disregard quantization errors. Although single-precision (32-bit) was considered quite suitable, double-precision (64-bit) representation is an alternative. The advantages of increased accuracy have to be weighted against the more time consuming arithmetics.

7.7.1.2. Analog-to-digital converter

Using the signal-to-noise ratio (SNR) in the output signal as a criterion, a minimum ADC word length can be specified for a given sampling interval. A 12-bit ADC can cope with a 20:1 SNR for sampling times equal or greater than 10 mS, for a first order algorithm implemented with fixed-point arithmetic [7.6.]. Since the minimum value of T under consideration is larger and floating-point arithmetic is used this will compensate for the noise magnification due to the higher order of the algorithm. It will therefore be reasonable to expect at least the same SNR.

Alternatively the ADC deadband can be considered as the cause of non-linear numerical phenomena [7.5.].

The deadband magnitude is given by:

$$DB \leq \frac{X_{\max}}{2^n} \quad (7.32)$$

where $X_{\max}$ is the full scale value of the sampled variable and n the number of bits of the ADC. For the turbine speed the ADC deadband is 0.244 rpm for a full scale value of 1000 rpm. For the wind speed the deadband is 0.00488 m.s^{-1} for a full scale value of 20 m.s^{-1} . In other words, the 12-bit ADC used gives a full scale resolution of 0.025%. The errors in the input of these critical variables are within quite reasonable bounds if we take into consideration the accuracy of the other elements in the data acquisition chain [5.4.]. The propagation of these quantization errors will be limited, due to the use of floating-point arithmetics.

7.7.1.3. Digital-to-analog converter

The DAC quantization effect is actually due to the 6-channel load control scheme. Since the "DAC" word length is only 6-bit, non-smooth behaviour of the control loop is to be expected. In fact the rule-of-thumb generally used for a cost effective choice of a DAC word length does not apply here. With a 12-bit ADC the right choice would be a 10-bit DAC.

The effect of the DAC roundoff can be considered as the same as a disturbance in the process input [7.7.]. Under this assumption the effect of the DAC limited word length on the process output is small if the process attenuates high frequencies, which is the case due to the low-pass behaviour of the wind turbine. Analog computer simulations [1.3.] of a similar loaded WECS have not revealed oscillations in the control loop, although a much larger quantization step has been used. It can thus be expected that the plant filtering properties will help to minimize the problem.

7.7.2. On the selection of the sampling rate

The issue of the selection of the sampling interval is an important one. The obvious benefit of choosing a long sampling interval is the reduction of the computational load, and thus necessary processing speed, and of the I/O rate, which determines the speed of the ADC. Generally speaking system performance is however impaired if the sampling time T is increased. A too short sampling interval may also be undesirable due to word length effects.

7.7.2.1. Algorithm accuracy effect

An algorithm tuned for long sampling intervals is more sensitive to process parameter drift than if it is tuned to work on a shorter sampling time [7.6.]. Generally speaking discrete-systems exhibit an increasing sensitivity to parameter errors for an increasing T when T becomes comparable to the period of any of the open-loop dynamics [7.4., 7.17.]. This is in agreement with the simulation results

(section 7.6.4.2.) where it is shown that the best global behaviour concerning parameter drift is achieved for the shortest sampling time.

7.7.2.2. Destabilizing effect

Conversion and computation times in a digital algorithm produce a dead-time which may cause closed-loop system instability due to the phase-shift produced. Since the approach taken to the system discretization through a zero-order-hold doesn't take this time delay into account, this is a potential problem.

The performance of PI and PID control algorithms is inversely proportional to the ratio $(\theta + T/2)/\tau$, where θ is the process dead time and τ the process time constant [7.17.]. The loop performance can thus be quantified by two terms : θ/τ , due to the process, and $(T/2)/\tau$, due to the sample-and-hold. When $T/2$ approaches the magnitude of θ , the degradation due to the sampler becomes significant. This requires T to be smaller than θ . In our case there is a system dead-time due to the variable time-delay introduced by the load actuation scheme. This has not been included in the process model because of the complexity of a suitable representation. This system variable dead-time assumes values in the range 0/25 mS. Taking the worst case values of $\theta = 25$ mS and $T = 47$ mS, the sampler and the system dead-times will have similar degrading effects. The total loop degradation will be given by $(\theta + T/2)/\tau = 0.02$, a substantially safe value according to [7.17.].

7.7.2.3. Information loss effect

This is illustrated in Fig. 7.13b. If the control loop is to operate with zero error, the sampler must operate fast enough to allow the information in the signals ω_o and T_{wo} to be preserved. Actually if ω_o and T_{wo} have frequency spectra bounded by f_w and f_T respectively the two following inequalities must verify [7.6.] :

$$\left\{ \begin{array}{l} f_s > 2 f_w \\ f_s > 2 f_T \end{array} \right. \quad (7.33)$$

where f_s is the sampling frequency.

Since the setpoint change is in the form of a step input, the first inequality specifies $f_s = \infty$. A practical approach is to argue in terms of acceptable delay before the response to a step begins. T can then be chosen so that the delay is acceptable, i.e. complies with the specified performance. For $T = 47$ mS the delay will range from 0 to 47 mS, i.e. only a fraction of the system response time.

In the second inequality f_T is the frequency of the disturbance to which the closed-loop system must respond in its regulatory role, i.e. when $\omega_o = \text{const.}$ High frequency disturbances that will be removed by the low-pass character of the plant in open-loop do not need to be considered. If we simplify assuming that the spectra of the wind speed and of the consequent torque disturbances have similar upper bounds, the highest frequency in the wind the controller has to counteract will determine f_s . Since no significant amount of energy can be extracted from wind frequencies above 2 Hz [1.8.], it seems reasonable to set f_T equal to this value. From equation 7.33 $f_s = 4$ Hz or $T = 250$ mS, i.e. a sampling time several times larger than the values selected from the simulation study.

7.7.2.4. System outputs not completely specified by the output z transform

Z-transform methods give information on the response of the continuous system only at intervals T , where T is the sampling time. Since the ZOH at the output of the digital computer subjects the continuous process to step inputs at the sampling instants the output of the system is actually a set of connected open-loop step responses. Intersample behaviour has therefore to be characterized. Provided that the open-loop step response is not responsible for unacceptable high frequency inter-sample fluctuations, taking $f_s > 10 f_c$ where f_c is attributable to the dominant pole-pair, usually assures that the output z-transform completely characterizes the system output [7.5.,7.6.,7.7.]. Alternative criteria specify a minimum number of samples, usually 2 to 4, per rise time in the system closed-loop step response [7.4.,7.6.,7.7.]. Figs. 7.18 to 7.21 show that for any sampling time up to 100 mS both criteria are satisfied.

7.7.2.5. Word length effect

Shortening the sampling time requires longer word lengths to resolve the small differences between adjacent samples of a given variable. An algorithm using the difference between two consecutive samples to calculate an output, such as the compensator used, is affected if too small a value is chosen for T . It will produce zero output during some sample intervals and sudden spikes whenever a change in the least significant bit occurs. The integral term of PI or PID regulators is a typical example and it is often implemented with a longer word length than the other terms. The representation used (section 7.7.2.1.) is believed to provide enough accuracy to allow these effects to be neglected.

7.7.3. Realization of the Control Algorithm

7.7.3.1. Compensator action

The control algorithm implemented in software will produce an output signal that the electrical actuation will convert in the desired loading of the WECS. Steady-state optimum operation is achieved by loading the turbine according to the optimum λ strategy. This will produce a quadratic $T_{el}(\omega)$ curve or, equivalently, cause the optimum locus in Fig. 7.10 to be tracked. The system transient behaviour will be under control of the loop compensator, $D(z)$.

The action of the compensator is now briefly described (refer to Figs. 7.1, 7.4 and 7.10). If, from a steady-state operation at a given point $T(\omega, V)$, the wind speed V changes, the compensator setpoint ω_0 will change accordingly. Since the turbine speed ω_1 for the same time step will not have changed, a non-zero error, $e(z)$ will be fed into the compensator input. The signal of $e(z)$ will be such that the turbine loading is altered as to compensate for the speed, or λ , deviation from the desired value. The compensator output is given by:

$$u(z) = D(z) e(z) \quad (7.34)$$

with $D(z)$ given by Equation 7.24.

$u(z)$ will be the control input to the actuation.

7.7.3.2. Implementation of the compensator

A digital PID can be implemented in a number of different ways [7.7.,7.21.], each possessing a number of advantages and drawbacks. Some elaborated forms provide enhanced performance for specific classes of systems. The chosen form (Fig. 7.14) is the so-called position type algorithm, considered adequate for our particular case. Equation 7.24 can be implemented in the digital computer through the following difference equation :

$$u(k) = K_p e(k) + K_i T \sum_{i=0}^k e(i) + \frac{K_d}{T} (e(k) - e(k-1)) + u(0) \quad (7.35)$$

where K_p , K_i and K_d are the compensator proportional, integral and derivative gains, T is the sampling time and k the time step. $u(0)$ is the actuator reset position. The integral term has been replaced by a rectangular approximation and the derivative term by a "backward" difference approximation [7.3.].

7.7.3.3. Actuation interface

The compensator output $u(k)$ is the value of the manipulated variable $1/R_1$. This value is updated once at each sampling time and held constant for the rest of the cycle. This digitized variable will correspond to the desired position of the electrical actuator, i.e., to the desired load step. Since the PIA output controls directly the actuation, this scheme is similar to a position feedforward actuator control.

The problem of actuator saturation is overcome in a simple way. After the value of $u(k)$ has been calculated, it is compared against the maximum and minimum values it may assume. These, $u_{\max}$ and $u_{\min}$, correspond to maximum electrical load or to no-load. If $u(k)$ exceeds $u_{\max}$ or is less than $u_{\min}$ it will be replaced by $u_{\max}$ or $u_{\min}$, respectively, as the control signal to the actuator.

7.7.4. On-Line Parameter Tuning

The preceding sections presented modelling and analysis work aimed at achieving an adequately tuned digital compensator.

Nevertheless it would be naive to expect perfect performance during field tests. Incorrect identification of the system physical parameters, use of an approximated linearised model and wrong evaluation of the in-plant interaction links are some of the reasons that may have contributed to errors in the compensator design. The possibility of easy adjustment of controller parameters such as maximum acceptable values for critical variables, compensator gains and sampling time is therefore a necessary feature. Such adjustments, done by simple system/operator dialogue, are however restricted to the "Stand-by" state. Whenever the system enters this state there is an automatic reset of a number of features and parameter values and the operator is asked to reinitialise the system (section 4.4.4.). On-line changes during actual turbine running were avoided because:

- a) algorithm parameter changes may cause switching transients,
- b) changes in the sampling rate require extensive hardware resetting and reinitialisation.

CHAPTER 8

CONCLUSIONS

8.1. CONCLUSIONS

8.1.1. General

Microcomputer-based WECS controllers are now under intensive development and are becoming increasingly popular. WECS as controlled plants possess some unique features and, from the controller end, many design and implementation aspects require non-standard solutions. The interdisciplinary nature of the plant requires expertise in diverse areas of engineering. An integrated approach to WECS control system development provides a good design methodology.

8.1.2. Development

Parallel developments in different areas like software, hardware and modelling and control have to be successfully integrated if a working prototype is to be obtained. The role of SADT's in building a conceptual model of the controller was found to be of vital importance in the development work.

To satisfy the controller specifications a powerful and versatile microcomputer-based on-line control system had to be designed and implemented. The extensive software and hardware development work required the support of powerful development facilities.

8.1.3. Software

Generality was one of the major objectives in the design of the control package. The WECS start-up/shut-down procedures, as well as the monitoring and operator interaction functions, can be used almost unchanged as control software for any WECS. Part of the software is however specific to the actual controlled plant. The control algorithm implemented during normal running is obviously dedicated to the specific WECS configuration and operating mode, and the low-level software modules are tied to the system hardware.

Nevertheless, if the control package is to be used with a different WECS configuration the necessary changes are confined to a certain number of software modules. If a transportable operating system supporting concurrent programming, e.g. Unix, is used the dedicated low-level software modules would be simpler and the real-time software support not needed at all. The Pascal based interactive software was written from scratch and is therefore completely transportable. The use of a graphics terminal would greatly improve its capabilities and efficiency.

Versatility was another goal in software design. Since no development facilities would be present in the field the software had to compensate by providing considerable flexibility in the preliminary system set-up. Setpoint and compensator parameter changes, display mode and control algorithm selections and minimum data logging facilities are features directed to the actual field testing.

Finally, by placing the interactive functions under control of the system executive, the WECS monitoring and control functions are made transparent to the operator. A very small dedicated operating system supporting the implementation of several on-line control algorithms for WECS is therefore achieved.

8.1.4. Hardware

The novel design energy conversion hardware was entirely built in the departmental workshop. The specific controller requirements and the actual wind turbine mechanical design called for non-standard solutions in hardware development.

Part of the required transducing/actuation hardware could be purchased but some had to be designed and built. The microcomputer controller hardware was built around board-level products since this appeared to be the most economical and practical approach. The interfacing circuitry had to be designed and built given the dedicated functions and the non-standard hardware involved.

Trade-offs between buying and designing hardware were present throughout the development stage. The solutions adopted for all subsystems proved satisfactory during laboratory testing, although still requiring testing under field operating conditions.

8.1.5. Testing

In all the testing stages the role of the MDS was of major importance. In-circuit emulation with the target hardware was used up to the final version of the control program. The development facilities available were of great help in debugging real-time software and troubleshooting non-standard hardware/software interfacing.

Controller performance was first assessed through real-time tests with an analog computer model of the WECS in which it replaced an analog controller without causing significant performance degradation in system behaviour. Following that, complete software/hardware integration was achieved in the laboratory.

Some issues of critical importance to the controller reliability and performance were investigated. The degrading influence of the operator interaction functions on the control cycle speed called for a more efficient implementation of the display updating software. The positioning accuracy of the blade pitch control loop was also experimentally determined and its limitations discussed. The steady-state performance of the electrical torque control loop was also checked, although its dynamic behaviour could not be assessed due to the limited testing environment.

The final software was then placed into EPROM's and the microcomputer controller is now capable of stand-alone operation.

8.1.6. Modelling and Control

A physically-based model of the WECS was derived to support the design and tuning of the main loop compensator. Techniques used in the analysis and design of computer-based process control systems were applied and the work was supported by dedicated simulation software.

The multivariable system is shown to be adequately represented through a continuous-time linearised state-space model. This model was then simplified by a loop decomposition approach so as to allow powerful SISO techniques to be used to analyse the WECS dynamic behaviour.

The response of the uncontrolled WECS to wind torque changes was investigated using continuous-time techniques. The uncontrolled WECS response to input wind torque changes was found to be highly oscillatory. The load dependent stability limits were determined. The investigation of system sensitivity to changes in physical parameters showed that minor errors in parameter estimation would not alter drastically the WECS performance.

The continuous elements of the closed-loop WECS were then discretized and z-plane analysis of the sample-data system supported the digital compensator design and tuning. Pole relocation techniques and WECS response to controller setpoint changes enabled optimum values for the compensator parameters to be found. Sampling time is shown to be a parameter of critical importance. For typical operating points the sets of compensator gains producing the "best" response are determined. The best compromise was found to be tuning the compensator for the nominal operating point and accept the performance degradation, due to system non-linear behaviour, at part load operation.

Finally it is shown that the realization of the control algorithm in the actual digital computer hardware causes minor degrading effects.

8.2. ORIGINAL CONTRIBUTIONS

The following are considered by the author to be original contributions to the field of WECS microcomputer-based real-time control :

- a) A development methodology supporting and integrating the various activities involved in the design and implementation of a micro-computer-based controller for a WECS.

Two major steps characterise the development approach :

- i) A functional model of the WECS controller obtained through the use of Structured Analysis and Design Techniques.
 - ii) The use of simulation benchmark algorithms to support a performance comparison study leading to the microcomputer hardware specification.
- b) On-line control software of general applicability to different WECS configurations.
- interactive programs
An interactive package specifically designed for on-line monitoring and system/operator communication.
 - dedicated operating system
A Pascal-based "kernel" allowing the implementation of simple pseudo-concurrency among the different control, monitoring and operator interaction tasks while enabling optimum CPU use and improved real-time performance.
 - low-level interfacing software
Portable software for digital and analog data acquisition and for digital actuation tasks.
- c) Integrated specification/design of all transducing, actuation and microcomputer-related hardware for the WECS controller duly complemented with design of dedicated interfacing circuitry.
- d) WECS parameter identification and state-space modelling of the variable-speed WECS under study. Simplified analysis of the multi-variable system by loop decomposition and SISO techniques.
- e) S-plane time-domain analysis of the open-loop system response to wind torque changes. Study of eigenvalues migration with parameter variation and determination of stability margins.
- f) System discretization and z-plane analysis of the closed-loop system. Choice of the digital compensator structure and tuning of its parameters using pole-placement techniques and step/ramp response criteria. Assessment of the effects of the realization of the control algorithm in the computer hardware.

8.3. SUGGESTIONS FOR FUTURE WORK

1. Field tests of the controlled WECS to assess controller performance.
2. Development of alternative control algorithms for energy extraction optimisation in VSVF WECS (section 7.2.3.) and also for the WECS start, stop and emergency stop procedures.
3. Extension and improvement of the interactive/graphic facilities by using a graphics terminal with a memory-mapped display.
4. Upgrading the minimal controller data-logging facilities. The VME Bus interfacing capabilities allow the controller to be provided with extra RAM boards or linked to a data aquisition system.
5. Modelling and control.
The microprocessor processing power is believed to allow reasonably sophisticated on-line control algorithms to be implemented. Modern control techniques can be used to deal with the multivariable non-linear system in a more general manner [8.1.]. The continuous-time state-space model developed can provide a good starting point. Adaptive and stochastic control techniques [8.1.,8.2.] offer considerable potential for research.
6. Enhancing the on-line control package with increasing knowledge of the controlled system. The WECS software is already structured as to accommodate several operator selectable control algorithms. If this selection is placed under the Scheduler, providing it with criteria to switch from one algorithm to another during WECS normal running, the controller power is considerably enhanced. More sophisticated control and estimation algorithms can be included and a certain degree of heuristic expertise can be built [8.3.]. Upgrading the controller hardware may however prove necessary if such a multi-layered expert control package is to be implemented.

APPENDIX 4

4.1. GENERAL FEATURES OF THE INTERACTIVE SOFTWARE

- Operator commands are entered from the keyboard and echoed in a command line (full duplex protocol).
- The system prompts for and replies to entered commands, guiding the operator at each step.
- Several types of operator commands are included (Fig. 4.5) :
 - control commands, e.g. start, stop, ...;
 - modify setpoints and compensator tuning commands;
 - display mode selection commands.
 - system query reply commands.
- The screen is shared among several functions (Fig. 4.6) :
 - operator command line;
 - system reply and message lines;
 - system status display area showing present WECS operating mode, pitch mechanism microswitches and vibration switch status;
 - setpoints display area shared by three groups, of three setpoints each, selected by the operator;
 - on-line display/plot area

System variables are either displayed as a real value or plotted. The variable display mode or any of the three variable plot modes are entered upon operator request (Figs. 4.7 and 4.8).
- Minimal editing facilities and help pages.

4.2. INTERACTIVE PACKAGE BUILDING BLOCKS (Fig. 4.4).

- A small editor (Read & edit)
 - inputs characters from the keyboard and places them in a buffer;
 - provides minimal editing facilities and local error handling
 - upon carriage return enters received command.
- A command parser and interpreter
 - acknowledges the entered command string and decodes it;
 - performs error handling, replies to the command and signals the Scheduler.
- Procedures implementing system/operator communication
 - set event flags;
 - change setpoints.
- User library and housekeeping procedures
 - read/write strings, read/write integers or reals;
 - configure screen, clear lines/blocks/screen;
 - display parameters, display/plot variables;
 - etc.
- Information and error handling procedures.
- ACIA initialisation and interrupt-handling procedures and their calling modules transferring strings of characters to the VDU or from the keyboard.

APPENDIX 5

CONTROLLABLE LOAD DESIGN AND IMPLEMENTATION

Performance Specification

The dynamic behaviour of VSVF WECS using a load switching scheme to achieve lambda optimum operation has been investigated through analog computer simulations. [1.3.] proposes the scheme already referred in section 6.3.2. The stepped approximation of the optimum quadratic load torque (Fig. 6.4) is obtained by switching the appropriate load value at the correct shaft speed. In an alternative scheme presented in [1.8.] the maximum power curve in Fig. 7.10 is tracked and the switching speeds are determined by allowing the power coefficient to fall by a certain percentage before the load is switched. Both methods operate as a function of shaft speed and the load is only accurately matched at certain steady-state operating points. Satisfactory performance is nevertheless reported in both cases. Transient optimisation schemes are also proposed in the same references but they use a continuous control variable.

The scheme implemented here attempts to achieve both steady-state and transient optimisation with a discrete loading method by providing a reasonable approximation to continuous load torque control. This calls for both a high number of load steps and a short control cycle. No significant WECS open-loop operation is therefore to be expected thus minimising the deviations from the optimum value of lambda.

Referring to Fig. A5.1 the action of the loading scheme implemented is now described. The transient behaviour of the closed-loop system when, from steady-state operation at $T_w(\omega_o, V_o)$, is subjected to a positive wind step is as follows. The system is initially in point "a" where the input wind torque is balanced by the electrical torque. When V increases by ΔV the driving torque increases and will be in excess of the load torque by ΔT . If the WECS is uncontrolled, or if the wind step is too low to determine a controller action, the system

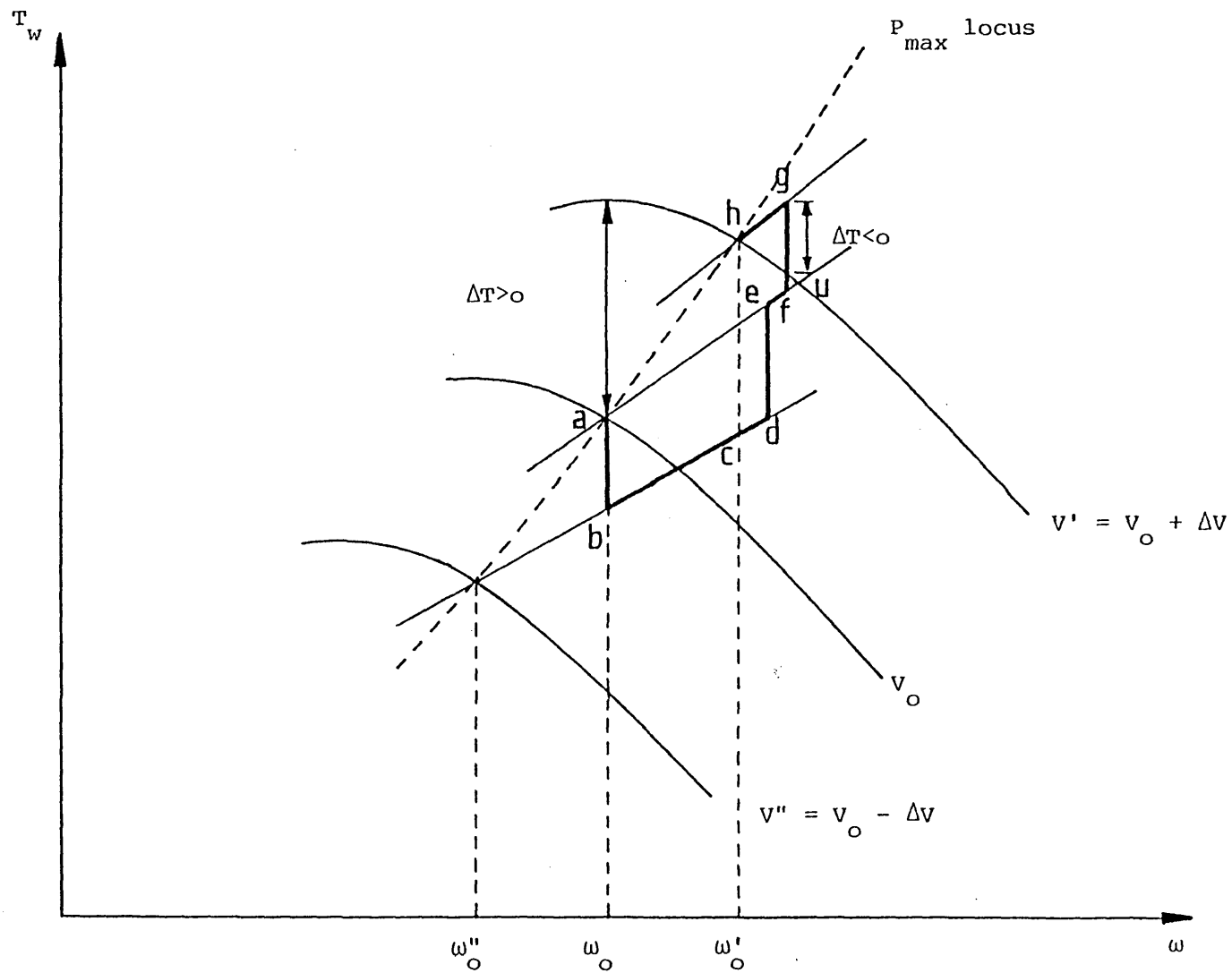


Fig. A5.1 - Load controller action.

speeds up following the load line until it reaches a new matching at point "u", which is off the maximum power locus.

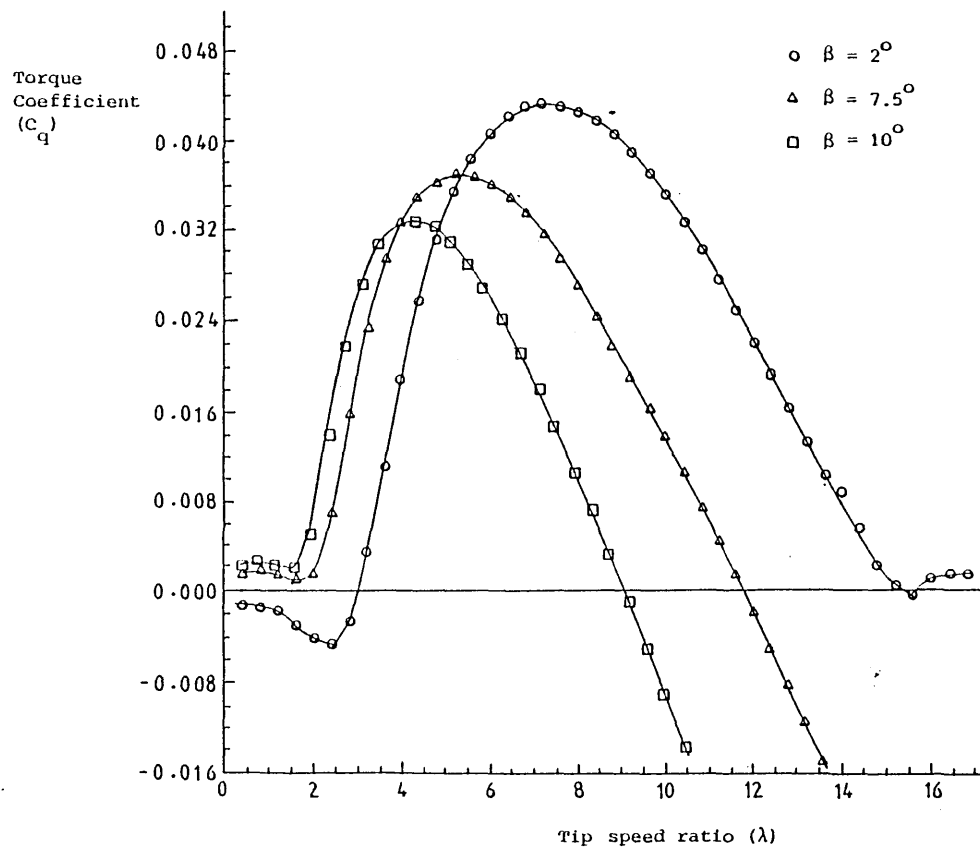
The controller action, already described in section 7.7.4.1., is illustrated by the path marked in dense in Fig. A5.1. Wind increase is first detected by the feedforward loop (Fig. 7.4) through an increase in controller setpoint from ω_0 to ω'_0 . A positive error signal will determine a change in the compensator output and the electrical actuation will unload the WECS (point "b"). The increase in surplus driving torque will help the WECS to accelerate along the new load line. If point "c" is not attained before the next control cycle the compensator will determine a new unloading action. Otherwise, as soon as the error signal becomes sufficiently negative, e.g. in point "d", the load will be stepped-up to point "e". A final load increase is then determined, producing a net negative torque, and the turbine will decelerate until the final equilibrium point, "h", is attained. For a negative wind step the process is similar.

For a large sudden wind change several load steps may be sequentially switched leading to large acceleration/ decelerations. However with a fast controller and a good stepped approximation to the maximum power locus the wind change from one control cycle to the next is kept small. Each new steady-state operating point will therefore be close to the previous one and a quasi-continuous electrical torque control is expected.

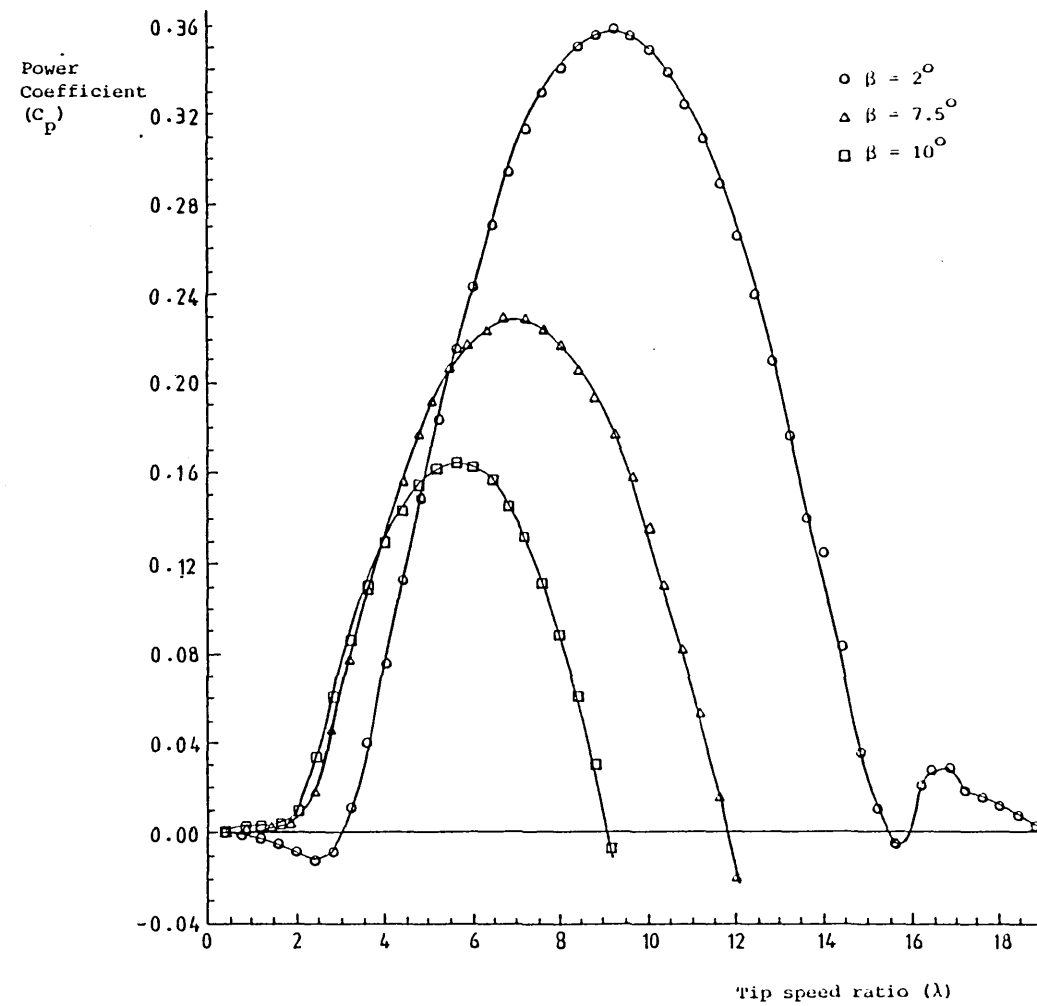
Design and Implementation

Two major issues in the controllable load design are its size, determined by the maximum power that needs to be dissipated, and the number of discrete steps, defined by the required precision in tracking the maximum power locus. From the discussion in last paragraph it appears essential to know the mechanical driving and electrical load curves in Fig. A5.1 for the whole range of WECS operation.

Since the generator test data is given in [5.1.] as $P(\omega, R_1)$ and $P(\omega, I)$ curves, it was decided to work on the $P(\omega, V)$ family of curves, which were obtained from the $C_p(\lambda)$ curves in Fig. A5.2b.



a) $C_q(\lambda)$



b) $C_p(\lambda)$

Fig. A5.2 - Wind turbine torque and power coefficient curves.

Selected operating points on the $P_{\max}$ curve are presented in Table A5.1 for both the 3.5 and the 9 m diameter WECS. The load size is determined either by the maximum power available from the larger rotor or by the maximum generator rating, whichever is smaller. With a maximum generator rating of 6.15 kW, or 7.8 kW if a theoretical maximum is considered, it was decided to size the load for 10 kW. The excess wind input power is to be spilled by blade pitching. The number of load steps is determined by the minimum loading specified for the smaller rotor, i.e. 123 W.

Let us now investigate other constraints from the generator end. Table A5.2, obtained from the load test curves presented in [5.1.], shows how the generator is to be loaded at different rotational speeds if maximum power is to be converted. These optimum generator operating points can be combined with the turbine limitations (Table A5.1) to produce the best possible load matching. From Table A5.2 it can also be seen that for a generator operating range from 200 to 1600 rpm the values of R_{ph} range from 2.5 to 145 Ω . Taking into account the matching values the lower limit can be raised to 7.5 Ω .

In the final design step, i.e. selection of the resistor values, further important issues have to be considered. First, since the generator will have a variable frequency variable voltage output, both the actual load rating and the incremental steps are voltage dependent. Second, a digital configuration is desired due to the form of the control variable. Finally, the theoretical design has to be tailored to available hardware and the resultant implementation checked.

Six digital channels appeared to be the best configuration given the load size and the minimum load step specified (Table A5.3). It was thus decided to design for a 10 kW, 64 steps switchable load. Table A5.4 shows the power available from each of the channels and the calculated resistor values, assuming $V = 208$ V as for the generator nominal operating point at 1600 rpm. The actual resistor and the corresponding dissipated power values presented ensure that all the optimum generator operating points can be met by the load. Table A5.5 shows the actual load power for different operating points. Using this and table A5.2 it can be seen that the load is capable of dissipating the generator output power for all operating points.

$\lambda = 9.3$	$R = 1.75 \text{ m}$		$R = 4.5 \text{ m}$	
$v(\text{m.s}^{-1})$	$n(\text{rev.min}^{-1})$	$P(\text{kW})$	$n(\text{rev.min}^{-1})$	$P(\text{kW})$
3.9	200	0.123	77	0.815
5.9	300	0.427	116	2.822
7.9	400	1.025	156	6.775
9.0	457	1.515	178	10.017
9.9	500	2.016	195	13.333
11.8	600	3.415	233	22.577
13.8	700	5.462	272	36.113

Table A5.1 - Typical WECS operating points for λ_{opt} operation ($\lambda_{\text{opt}}=0.93 \Rightarrow C_p=0.36$).

number of digital channels	number of load steps	minimum step size (W)
8	255	39.2
7	127	78.7
6	63	158.7
5	31	322.6
4	15	666.7

Table A5.3 - Partitioning of the 10 kW load as a function of the number of digital control channels.

Channel	$P_{\text{total}}(\text{W})$	$P_{\text{phase}}(\text{W})$	$R_{\text{calcul.}}(\Omega)$	$R_{\text{actual}}(\Omega)$	$P_{R_{\text{max}}}(\text{W})$	Hardware configuration
1	158.7	52.9	818	1000	59	1 resistor
2	317.4	105.8	409	470	100	1 resistor
3	634.8	211.6	204.5	235	200	2x(470 Ω ,100 W) resistors
4	1269.6	423.2	102.2	110	540	3x(330 Ω ,180 W) resistors
5	2539.2	846.4	51.1	57.6	1000	fan-heater-set.1
6	5078.4	1692.8	25.6	28.8	2000	fan-heater-set.2
All	9998.1	3332.7	13.0	14.6	3899	

Table A5.4 - Design table for the 6-channels, 10 kW switchable load.

$m(\text{rev.min}^{-1})$	$P_g(\text{W})$	$I(\text{A})$	$V_{ph}(\text{V})$	$R_{ph}(\Omega)$
200	100	1	38	38.0
	200	2	35	17.5
	250	3	33	11.0
	400	4	30	7.5
	500*	6	25	4.2
	500*	8	20	2.5
	400	10	15	1.5
400	400	2	75	37.5
	800	4	70	17.5
	1000	5	67	13.4
	1100	6	65	10.8
	1200*	8	55	6.9
	1200*	10	40	4.0
600	800	2	115	57.5
	1200	4	105	26.2
	1800	6	100	16.7
	1900	7	98	14.0
	2000	8	90	11.2
	2100*	10	75	7.5
	1600	12	45	3.8
800	1000	2	150	75.0
	1700	4	145	36.2
	2400	6	135	22.5
	2700	8	120	15.0
	2800*	10	95	9.5
	2400	12	65	5.4

$n(\text{rev.min}^{-1})$	$P_g(\text{W})$	$I(\text{A})$	$V_{ph}(\text{V})$	$R_{ph}(\Omega)$
1000	1200	2	190	95.0
	2100	4	185	46.2
	3000	6	170	28.3
	3600	8	150	18.8
	3800*	10	125	12.5
	3200	12	90	7.5
1200	1400	2	225	112.5
	1600	4	220	55.0
	3700	6	200	33.3
	4400	8	185	23.1
	4600*	10	150	15.0
	3900	12	110	9.2
1400	1600	2	255	127.5
	3100	4	250	62.5
	4300	6	235	39.2
	5200	8	215	26.9
	5400*	10	180	18.0
	4800	12	125	10.4
1600	1900	2	290	145.0
	3400	4	285	71.2
	4900	6	275	45.8
	5900	8	245	30.6
	6150*	10	208	20.8
	5400	12	150	12.5

Table A5.2 - Selected generator operating points (from test results in [5.1.]).
Optimum operating points signalled with an asterisk.

	P_{actual} (W) per phase							
	V_{ph} (V)							
Channel	208(a)	180	150	120	98(b)	70	35	P_n/P_{n-1}
1	43.3	32.4	22.5	14.4	9.6	4.9	1.2	--
2	92.1	68.9	47.9	30.6	20.4	10.4	2.6	2.13
3	184.1	137.9	95.7	61.3	40.9	20.8	5.2	2.00
4	393.3	294.5	204.5	130.9	87.3	44.5	11.1	2.14
5	751.1	562.5	390.6	250.0	166.7	85.1	21.3	1.91
6	1502.2	1125.0	781.2	500.0	333.5	170.1	42.5	2.00
All	2966.1	2221.2	1542.4	987.2	658.4	335.8	83.9	average
All phases	8898.3	6663.6	4627.3	2961.6	1975.2	1007.4	251.7	2.036

Table A5.5 - Power dissipated by the load at several WECS operating points.

(a) refers to the full-loaded 9 m diameter and (b) to the full-loaded 3.5 m diameter WECS.

The deviation from an ideal equally stepped load is only 3.6%.

A satisfactory compromising solution is thus obtained with 6 digital channels. Since only 5 channels would clearly provide a too coarse load curve, and the finer tracking enabled by 7 channels would call for unnecessary extra hardware, these cases were not considered. Fig. A5.3, where the controller action is illustrated as in Fig. A5.1, shows how closely the optimum quadratic load torque curve is tracked. A speed deviation from the optimum setpoint of about 1 rad.s⁻¹, or 10 rpm, will determine a switching action, whereas in the schemes presented in [1.3,1.8.] the allowed deviation ranges from 10 to 30 rad.s⁻¹.

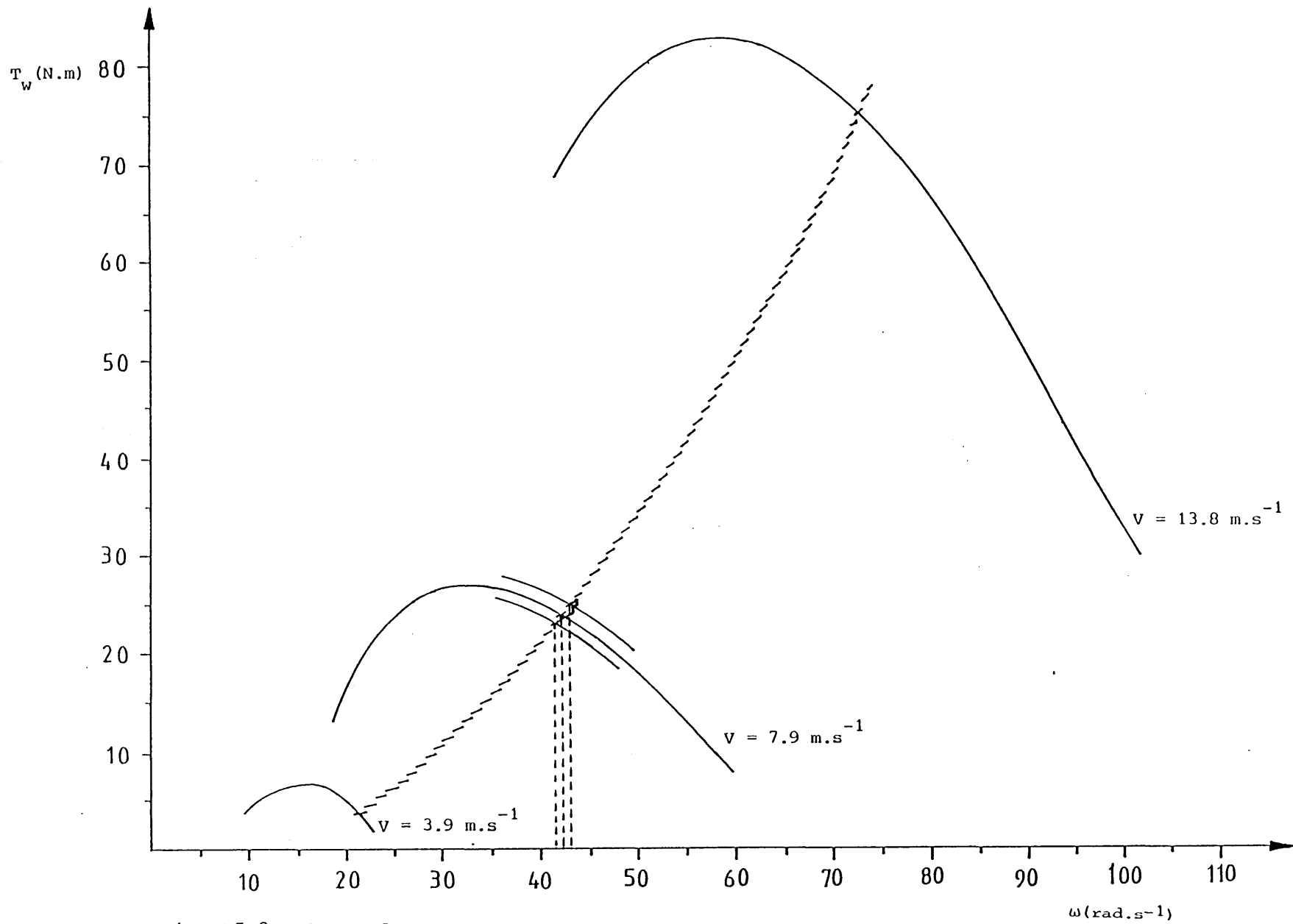


Fig. A5.3 - Stepped approximation to the quadratic load torque curve.

APPENDIX 6

MICROCOMPUTER HARDWARE SET-UP AND LABORATORY TESTS

Set-up of the microcomputer hardware

A certain amount of freedom was allowed in the configuration of the microcomputer hardware. As a consequence each of the boards can be configured to best suit the system operation, transforming a relatively general purpose into a dedicated microcomputer system.

Decisions had to be made on the system memory map, choice of on-board RAM/EPROM and peripheral boards addressing. Interrupt level allocation and PIA, PTM, ACIA and ADC operating modes had also to be fixed in hardware by adequate board jumpering.

i) System memory map

The CPU on-board sockets accommodate up to 256 K of EPROM/RAM mix. The more sensible choice, taking into account speed of access, price and availability of the various supported memory IC's, was to install 16Kx8 EPROM and 8Kx8 RAM chips. The system will thus be provided with a total of 64 K EPROM and 32 K RAM on-board memory, more than double the final memory requirements, presented by the linker map of Fig. A6.1. The on-board PROM decoder IC was blown as to configure the system with the desired memory map (Fig. A6.2).

ii) Interrupt level allocation

The relative importance of the processes supported by the various interrupt routines and the execution times of these routines were the major criteria in the assignment of interrupt priorities. Fig. A6.3 shows how these priorities are allocated. Level 7 is reserved for use of the HP-64000 emulator and to block all external interrupts. The timer interrupt has the highest priority due to the time-critical nature of its functions, especially if we take into account the fast

HP 64000 Linker

FILE/PROG NAME	PROGRAM	DATA	COMMON	ABSOLUTE

MONIT_AC1:MENDON	0001000			0000000-000002F 00000BC-00000BF 0000068-000006B 0000070-0000073 0000078-000007B 000007C-000007F
SCHED7:MENDON	0001574	0000800		
WECS_7:MENDON	0001A54	0000816		
INPDRI6:MENDON	0003088	0000932		
START7:MENDON	0003BA4	00009E8		
OPTIM7A:MENDON	00040F6	0000A16		
STOP7:MENDON	0004248	0000A76		
EMERG7:MENDON	000436C			
VDUTI7A:MENDON	00044B0	0000A86		
LIBRARY7:MENDON	0005D20	0000AF2		
INTMANAGE:MENDON	0007106	0000D0E		
TESTINTMA:MENDON	000720E			
CURSORADR:MENDON	0007296			
next address	00072F6	0000D12		
Libraries				
A5_LIB:L68000				
A5Zcase:L68000	00072F6	0000D12		
A5Zdiv:L68000	000730A			
.....				
REAL_LIB:L68000				
SABS:LR68K	000742A			
SADD:LR68K	0007436			
SDIV:LR68K	0007454			
.....				
next address	0007E58	0000F40		

XFER address= 00018EE Defined by SCHED7
 No. of passes through libraries= 2
 absolute & link_com file name=LAB_TEST7:MENDON
 Totalf of bytes loaded= 00075DB

Fig. A6.1 - Linker Load Map.

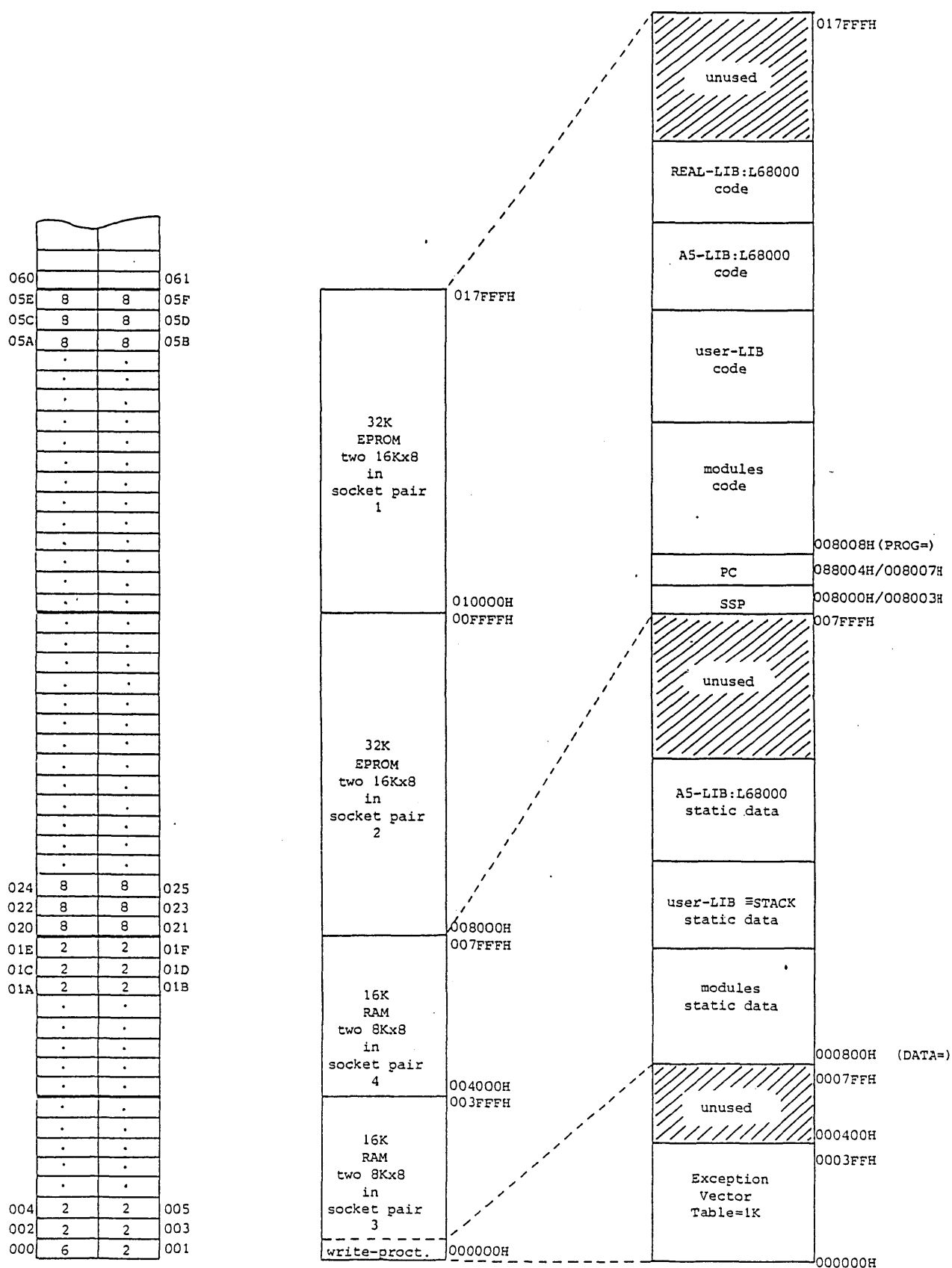


Fig. A6.2 - System Memory Map.

```

*      AUTOVECTOR INTERRUPT LEVEL 1  ( _____ )

      ORG 064H
      DC.L PIA_INT1

*      AUTOVECTOR INTERRUPT LEVEL 2 (ACIA_INT )

      ORG 068H
      DC.L ACIA_INT

*      AUTOVECTOR INTERRUPT LEVEL 3  (PIA_INT )

      ORG 06CH
      DC.L PIA_INT3

*      AUTOVECTOR INTERRUPT LEVEL 4  (ADC_INT )

      ORG 070H
      DC.L ADC_INT

*      AUTOVECTOR INTERRUPT LEVEL 5  ( _____ )

*      INTERRUPT LEVEL DISABLED

*      AUTOVECTOR INTERRUPT LEVEL 6  (PTM_INT )

      ORG 078H
      DC.L PTM_INT

*      AUTOVECTOR INTERRUPT LEVEL 7  (FAIL_INT )

*      BREAK VECTOR ( RE-ORG WHERE APPROPRIATE )

      ORG 7CH
      DC.L MONITOR_ENTRY
      SKIP

```

Fig. A6.3 - Interrupt Priority Allocation.

sampling required. The ACIA has the lowest priority since the time consuming display/interaction functions it implements are not vital.

iii) Peripheral hardware operating modes

The operating modes selected for the peripheral hardware, set by proper strapping of the boards [5.15.-5.17.], are now briefly mentioned.

The RS-232 link between the microcomputer and the terminal will operate with a baud rate of 9600, the maximum the terminal can cope with. The three timer/counters in the PTM came already hardwired in a cascade configuration. Since their independent use is not possible, all the necessary timing loops are built around the sampling time loop.

The PIA peripheral lines have to be configured either as input or as output in groups of eight. Some of the control lines are hardware strappable as input or output. The system digital I/O requirements decided on the PIA configuration, illustrated as in Fig. A6.4.

For better noise immunity a differential configuration was used for the analog input board. The high-level outputs from all the transducers, with a maximum range of 10 V, determined the ADC input voltage range.

Preliminary testing

Before testing the final version of the control program two intermediate testing levels were included.

The HP-64000 emulation monitor was first used to check all the hardware involved, from the microcomputer to the WECS, transducers and actuators. No user software was present so that the causes of abnormal operation could be limited to hardware malfunction. All the hardware subsystems were tested, one by one, with the help of display/modify memory commands.

The diagram shows the pinout of the 74181 ALU. The pins are numbered 2 through 50. Key labels and connections include:

- Pin 49:** Res (Reset)
- Pin 48:** CA1 (Counter A Input 1)
- Pin 47:** Res (Reset)
- Pin 46:** CA2 (Counter A Input 2)
- Pin 45:** Res (Reset)
- Pin 44:** PA0 (Parallel Adder Input 0)
- Pins 43-32:** 12-bit binary counter outputs (labeled 1 through 12)
- Pin 31:** PBO (Parallel B Input 0)
- Pins 30-23:** 8-bit vibration switch inputs (labeled 1 through 8)
- Pins 22-19:** grounded inputs
- Pin 18:** Res (Reset)
- Pin 17:** CB1 (Counter B Input 1)
- Pin 16:** GND (Ground)
- Pin 15:** CB2 (Counter B Input 2)
- Pin 14:** 2 (Counter B Input 2)
- Pin 13:** 4 (Counter B Input 4)
- Pin 12:** 6 (Counter B Input 6)
- Pin 11:** 8 (Counter B Input 8)
- Pin 10:** 10 (Counter B Input 10)
- Pin 9:** NC (No Connection)
- Pin 8:** 8 (Counter B Input 8)
- Pin 7:** 6 (Counter B Input 6)
- Pin 6:** 4 (Counter B Input 4)
- Pin 5:** 2 (Counter B Input 2)
- Pin 4:** 0 (Counter B Input 0)
- Pin 3:** 1 (Counter B Input 1)
- Pin 2:** 2 (Counter B Input 2)
- Pin 1:** 0 (Counter B Input 0)

203

Software was then written to link the operation of the various subsystems. Real operating conditions were simulated to run critical blocks of the Start, Stop, Emergency Stop and Optimumcontrol procedures.

Most of the problems encountered at this stage were tied with the time-critical nature of the software. Time-dependent errors, probably the more delicate feature of real-time software, could be located with the help of the emulator/logic analyser facilities. The problems inherent to the interfacing of a microcomputer controller to slow electromechanical devices were also examined. Relevant parts of this dedicated testing stage are now presented with some detail.

i) Slow response to electromechanical devices

All the electromechanical actuation devices, i.e. clutches, actuator and brake, have a large response time from the microcomputer point of view. Several hundreds of mS may elapse from the time a control signal is issued by the computer to the time the device is actually energised/deenergised. This is taken into account in the software by granting delay loops whenever appropriate.

ii) Blade pitch control loop

The blade pitch control loop, a DDC loop not included in the simulation study presented in Chapter 7, was the object of more close testing. Although positioning accuracy and speed of response of this loop are not critical it was required to check its functionality and performance. Referring to Fig. A6.5 the loop action is now described.

Blade pitch is indirectly measured through a linear potentiometer (section 5.3.5.), and this position feedback is used to produce a digital control output to the actuator. The non-linear relationship between blade angle and potentiometer reading is given by the curve in Fig. A6.6. This calibration curve was obtained after final assembly of the blade pitch mechanism links by measuring blade angles with a clinometer and reading the corresponding potentiometer displacements from the controller terminal.

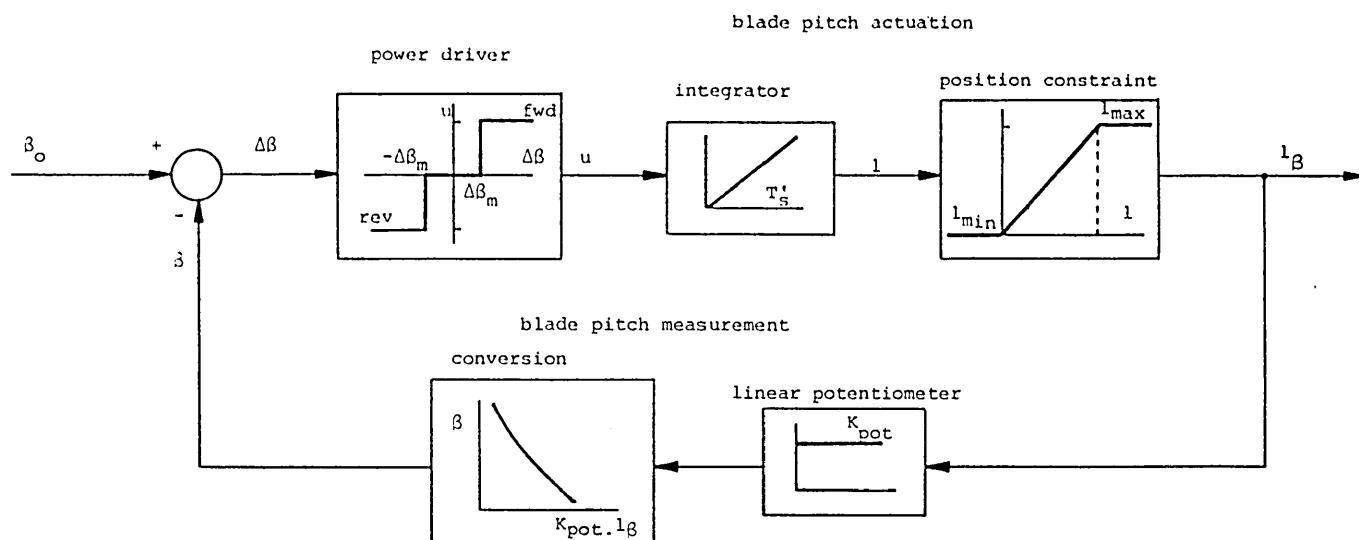


Fig. A6.5 - Blade Pitch Control Loop.

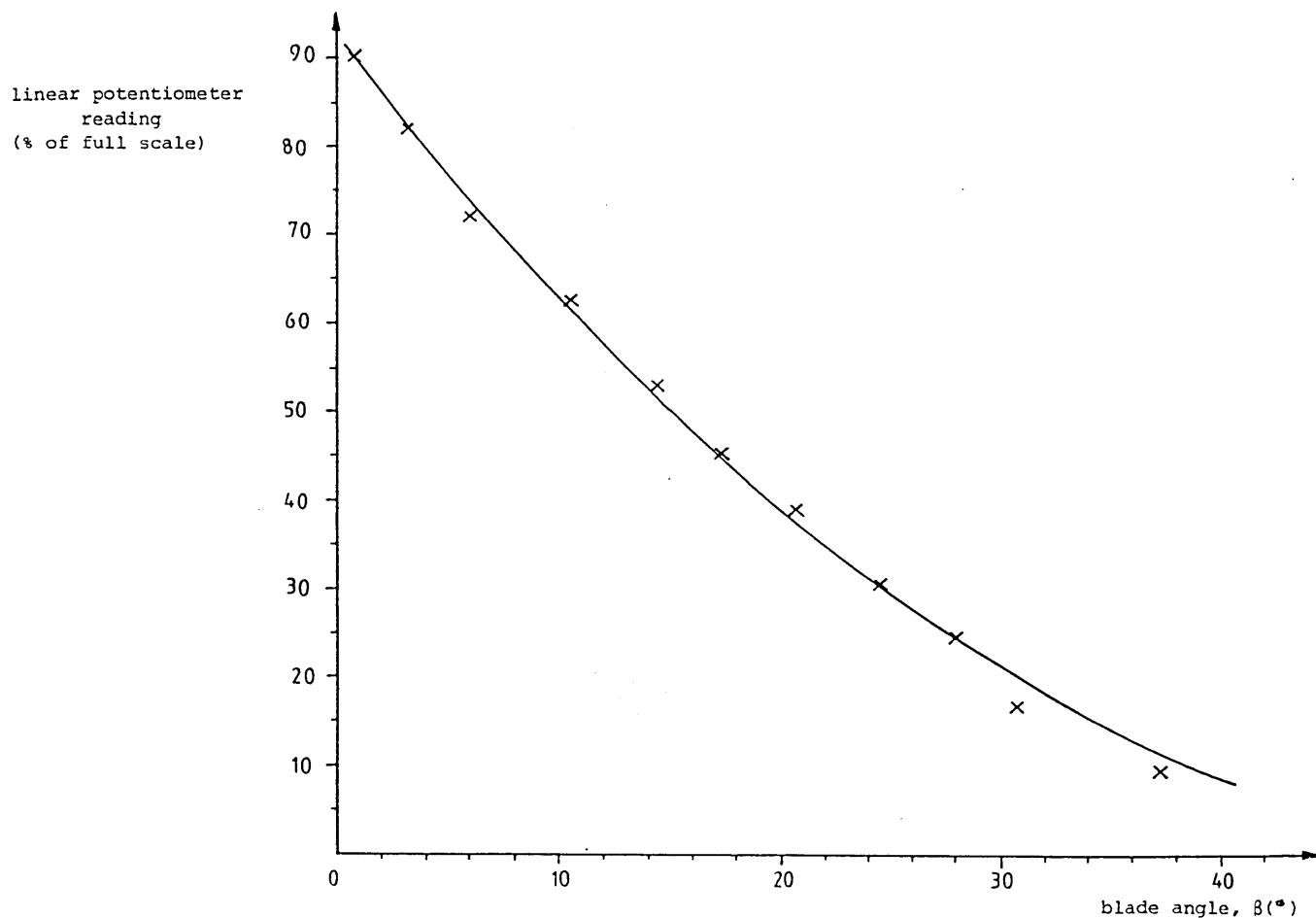


Fig. A6.6 - Blade Pitch Calibration Curve.

The actuator control consists in energising it as to produce a forward/reverse/null movement of its arm (section 5.4.1.). With respect to actuator dynamic response this method can be classified as an integral actuation scheme with constant speed [7.18.]. The non-linear behaviour introduced by the "three-way switch" actuation, the relatively slow switching action and the load dependent actuator settling time will degrade control performance. The effect of the first two of these problems could be quantified by laboratory testing.

Positioning accuracy of the closed-loop system is determined by the quantization unit of the actuator position, which may be defined by

$$\Delta A = T_{ao}/T_s$$

where T_{ao} is the smallest realizable switch duration and T_s the running time of the actuator for the whole position range.

T_s depends of the rate of the actuator movement which, according to manufacturer information, ranges from 2.5 to 3.8 cm.sec-1. Since the complete travel, from fully coarse to fully fine blade pitch, is 5.5 cm T_s will range from 1.4 to 2.2 seconds.

T_{ao} is limited by the switching speed and by the minimum switch duration for the motor to actuate, and is therefore difficult to evaluate. The quantization unit of the actuator position may however be estimated by experimentation. For the spring loaded actuator positioning accuracy in the fine pitch range was found to be limited to $\pm 0.75\%$. If the curve in Fig. A6.6 is approximated with a straight line this limit corresponds to $\Delta A = 1.5/36$ and therefore T_{ao} ranges from 60 to 90 ms.

Switching speed, mainly determined by the response time of both SSR's and power relays, is believed to be one of the constraints on the minimum switch duration. The actual configuration of the blade pitch mechanism links is another limitation since it does not favour the loop positioning accuracy. In fact the amplification produced by the links is such that, in average, 1° of blade corresponds to only 1.5 mm of actuator travel.

Although the positioning accuracy seems perfectly adequate in relation to the speed/acceleration control loops used to start/shut-down the WECS, it may prove limiting if very accurate blade pitch setting is required during normal running. A faster and more sophisticated actuation control method (see section 5.4.1.) will have to be used if the loop performance needs improvement and the mechanical design is to remain altered.

iii) Loading effects caused by the display functions

The time consumed in display updating slows down program execution speed calling for larger sampling times. On the other hand system dynamic behaviour places upper bounds on acceptable sampling time values (Chapter 7). Since all the controller functions have to be executed within each sampling cycle, further study is required when integrating all the correspondent software modules.

The direct measurement facilities of the logical analyser, allowed the relevant program blocks to be timed. From the values presented in Table A6.1, the display updating functions are allowed 20 mS in each 47 mS control cycle. Since updating variable plots, the fastest of the two display modes, exceeds this value and other screen updating is still required the 47 mS limit is exceeded. This problem is overcome by spreading the one second display update function over several control cycles. As a sampling time ranging between 45 and 250 mS is offered, the display updating is spread out by four consecutive cycles. Updating variable values is a much slower function, due to the necessary string manipulation and to longer strings displayed. Even spreading it over four cycles the lower limit for the sampling time is 150 mS.

The slow ACIA operation and the use of a dumb terminal result in inefficient display management. Cursor control has to be done through the use of escape codes and software-based position decoding. The software may be relieved from most of the display functions if a memory-mapped terminal is used. The allowed sampling times will then go down to 27 mS for the variable plot mode and 90 mS for the variable display mode.

Module Name	Execution Time (mS)
ADC-INT	18.8
PTM-INT	2.1
OPTIMUMCONTROL	2.0
Control functions total	22.9
UPDATE-PLOT	43.05
UPDATE-VAR	215.0
UPDATE-PLOT (modified)	20.0
UPDATE-VAR (modified)	120.0

Table A6.1 - Execution times of the main program blocks executed each control cycle. The modified display update functions are spread over four consecutive control cycles.

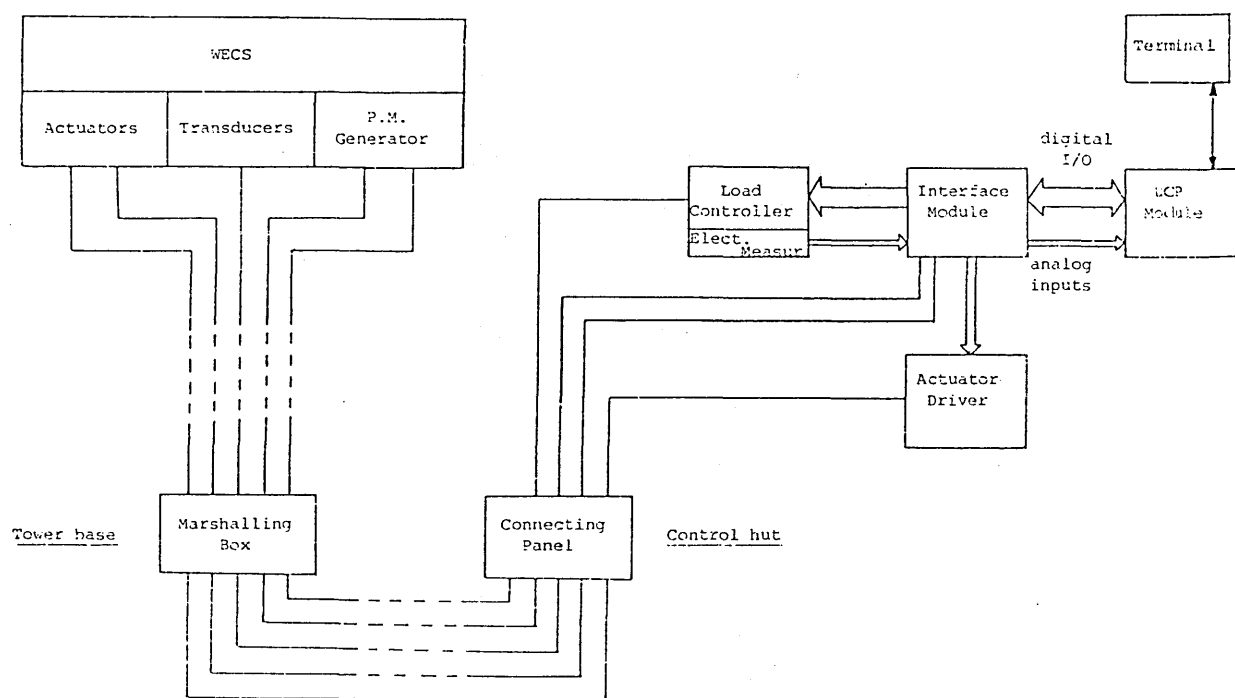


Fig. A6.7 - Field Hardware Layout.

Final version of the control software

Final testing of the software consisted of running the control program as it would run in the field.

An attempt was made to simulate the likely environment as closely as possible. Relevant inputs, such as wind and turbine speeds or vibration signals, were simulated from the transducer end therefore including the complete subsystems. Other input signals, e.g. from the linear potentiometer, microswitches, etc., were actually portraying the WECS status.

The system was run through all possible operating modes (Fig. 3.2) by triggering the required conditions, always from the transducer end. All possible transitions between states were checked and in each of them all program paths were exercised. System behaviour was monitored not only through the interactive package but also by checking the operation of the actuation devices. All the sequential and blade pitch control loops were fully tested. The DDC loop implementing electrical torque control could only be tested for its steady-state performance, due to the limited testing environment.

A minimal data logging function was included to help the dynamic behaviour of the electrical torque control loop to be evaluated in the field. A number of samples of relevant variables is stored in array form after each load step change determined by the controller. Once the reserved RAM space is completed the data logging stops. WECS dynamic response to controller setpoint changes can be analysed by displaying the collected samples in graphical form once the system returns to the Stand-by state. Field test results can therefore be compared with the simulation results presented in Chapter 7.

Field Tests

The completely debugged software was then put into EPROM's using the HP-64000 PROM programming facilities, in accordance to the system memory map. Detailed wiring diagrams were produced allowing the proper connection of all WECS transducers/actuators to the interface module. Fig. A6.7 shows the planned WECS field hardware layout.

APPENDIX 7

7.1. SIMULATION SOFTWARE

The organisation of the dedicated software supporting the analysis/design work presented in Chapter 7 is illustrated in Fig. A7.1. The programs were written in HP-basic in an HP 9825A microcomputer and an HP 9872A programmable plotter was used to obtain the graphic output. The library root-finder routine used, based on the so-called Siljak coefficient method, did not place convergence problems. The accuracy of the solutions was checked with a FORTRAN library root-finder, using the so-called method of Grant and Hitchins, which was run in the College CDC mainframe.

7.2. WECS MODELLING AND PARAMETER IDENTIFICATION

WECS Lumped Parameter Model

Since the WECS is an electromechanical energy conversion system, a lumped parameter model can adequately describe its dynamic behaviour. The rotational system is described in terms of mass-spring-damper arrangements. The values for the physical parameters are derived and the underlying continuity equations presented. The main components of the rotational system are presented in Fig. A7.2a). Assuming an infinitely rigid shaft, the system can be seen as a set of three mass-spring-damper groups as in Fig. A7.2b).

Parameter Identification

J_t comprises the inertias of the 3.5 diameter blades and respective counterweights and the hub inertia. J_{fc} , S_{fc} and B_{fc} were

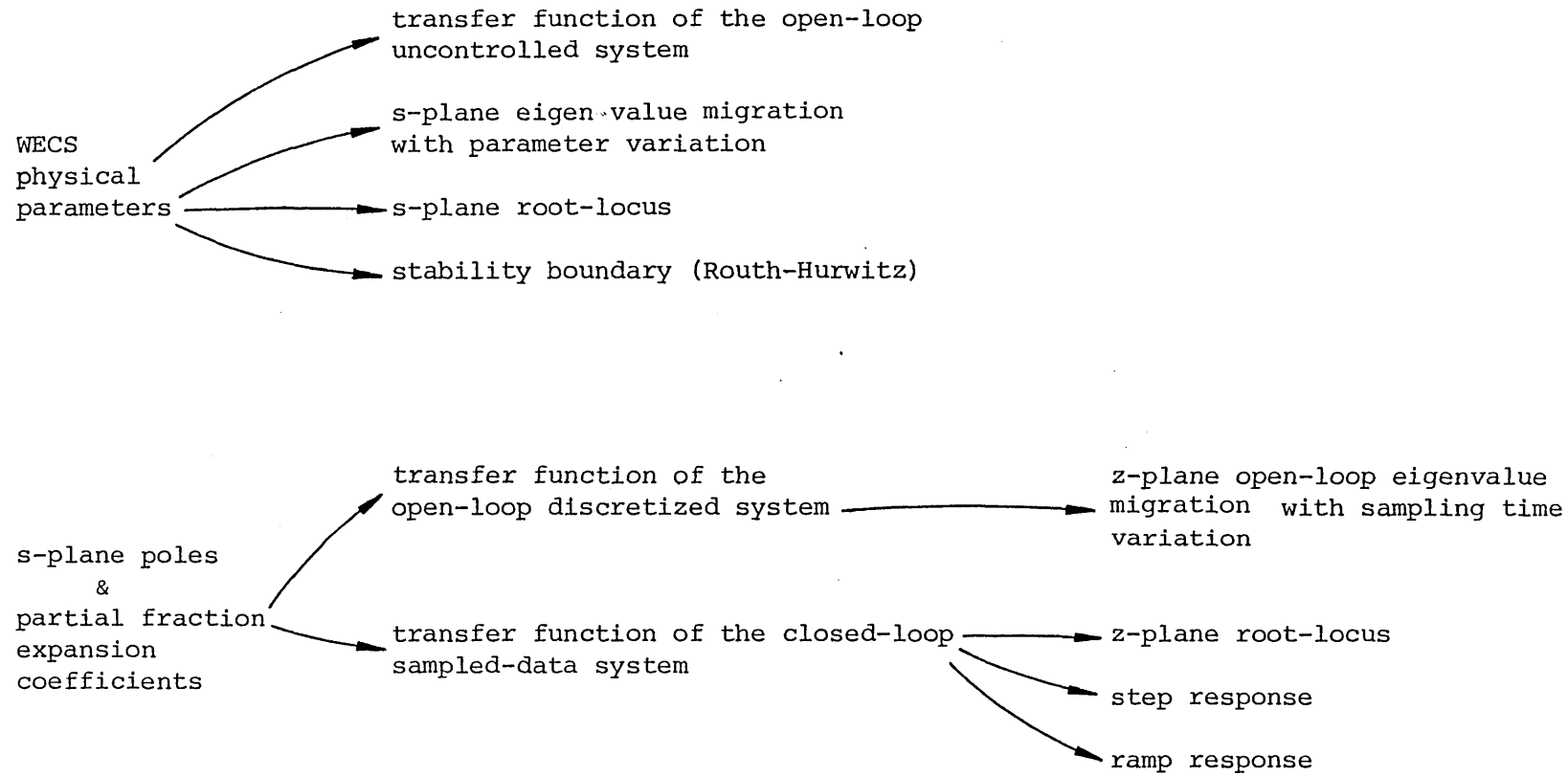
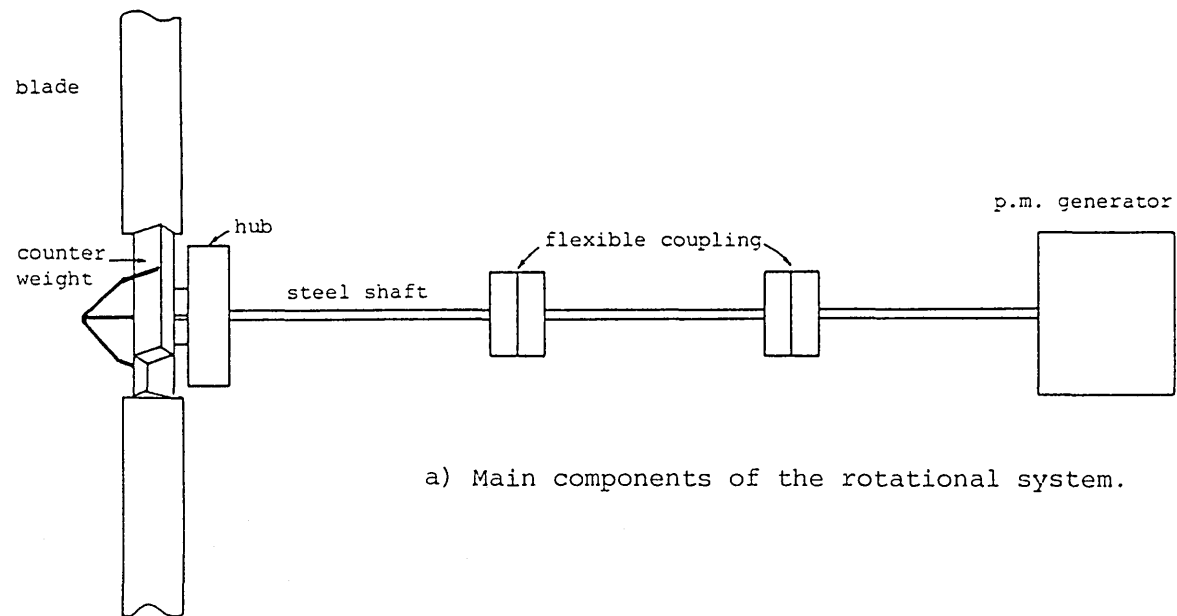
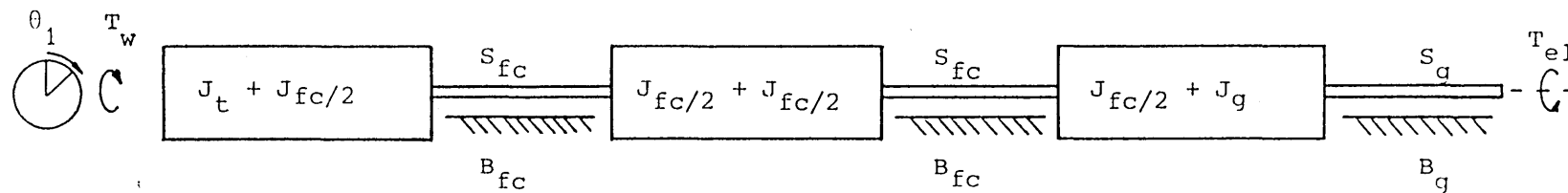


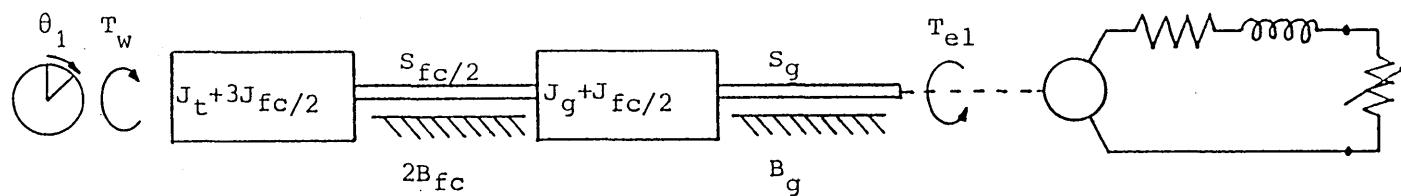
Fig. A7.1 - Structure of the simulation software.



a) Main components of the rotational system.



b) WECS 6th order representation.



c) WECS 4th order representation

Fig. A7.2 - WECS modelling.

estimated on the base of manufacturer's information. A value 30% smaller than the actual value was chosen for B_{fc} to allow for the ageing of the rubber. The value taken for J_g was calculated with the help of drawings from [5.1.]. S_g is taken as 1% of the full load "electrical" stiffness, which is derived from the generator synchronizing torque. B_g is the "electrical" damping attributable to the generator, i.e. corresponding to the no-load losses. The load produced "electrical" stiffness and damping will be included as the explicit control variable. The parameter values are presented in Table A7.2a) for the three operating points under study (see section 7.4.3.).

Model Order Reduction

Table A7.2b) presents the values of the natural frequencies and damping ratios attributable to each of the three mass-spring-damper groups. Subscripts "1" to "3" correspond to the referred groups from left to right. The poles attributable to the second group will be non-dominant because they are well damped and have a high natural frequency which is certainly not going to be excited by the input wind torque. This mode of oscillation is therefore neglected and the system order reduced. The larger rotor inertia includes that of group 2, and the equivalent "mechanical" stiffness and damping are given by :

$$\frac{1}{S_1} = \frac{1}{S_{fc}} + \frac{1}{S_{fc}} \quad ; \quad B_1 = 2 B_{fc} \quad (A7.2)$$

Fig. A7.2c) shows the fourth order simplified model and Table A7.2c) the corresponding parameter values. It is apparent from Table A7.2b) that the validity of the model order reduction is maintained over the WECS operating range.

	J_t	J_{fc}	J_g	S_{fc}	S_g	B_{fc}	B_g
	N.m. s ² .rad ⁻¹			N.m. rad ⁻¹		N.m.s. rad ⁻¹	
(i)	5.0	0.181	0.335	3500	2.1	3.0	0.0222
(ii)	"	"	"	1134	2.22	"	0.0456
(iii)	"	"	"	280	2.94	"	0.0821

a) WECS physical parameters.

	ω_{n1}	ω_{n2}	ω_{n3}	ζ_1	ζ_2	ζ_3	$\frac{\omega_{n2}}{\omega_{n1}}$	$\frac{\omega_{n2}}{\omega_{n3}}$	ζ_2/ζ_1	ζ_2/ζ_3
	rad.s ⁻¹ (Hz)									
(i)	26.22 (4.17)	139.06 (22.13)	2.22 (0.35)	0.011	0.060	0.012	5.3	62.6	5.5	5.0
(ii)	14.93 (2.38)	79.15 (12.60)	2.28 (0.36)	0.020	0.105	0.023	5.3	34.7	5.3	4.6
(iii)	7.42 (1.18)	39.33 (6.26)	2.63 (0.42)	0.040	0.211	0.037	5.3	15.0	5.3	5.7

b) Damping ratio and natural frequency values for the 6th order model mass-spring-damper groups.

	J_1	J_2	S_1	S_2	B_1	B_2	K_1
	N.m.s ² .rad ⁻¹		N.m. rad ⁻¹		N.m.s.rad ⁻¹		
(i)	5.272	0.426	1750	2.1	6.0	0.0222	500
(ii)	"	"	567	2.22	6.0	0.0456	300
(iii)	"	"	140	2.94	6.0	0.0821	200

c) 4th order model parameters.

Table A7.2 - WECS model order reduction.

7.3. SYSTEM TRANSFER FUNCTION MATRIX

The WECS state-space model is given by Eqn. 7.6 :

$$\begin{cases} \dot{\underline{x}} = \underline{A} \underline{x} + \underline{B} \underline{u} \\ \underline{y} = \underline{C} \underline{x} \end{cases} \quad (\text{A7.3.1a})$$

where the vectors $\underline{x}$, $\dot{\underline{x}}$, $\underline{u}$ and $\underline{y}$ are given by :

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} ; \quad \dot{\underline{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} ; \quad \underline{u} = \begin{bmatrix} T_w \\ \frac{1}{R_1} \end{bmatrix} ; \quad \underline{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \quad (\text{A7.3.1b})$$

with

$$\begin{aligned} y_1 &= \omega_1 = \dot{\theta}_1 = x_3 \\ y_2 &= \omega_2 = \dot{\theta}_2 = x_4 \end{aligned} \quad (\text{A7.3.1c})$$

and matrices $\underline{A}$, $\underline{B}$ and $\underline{C}$ are

$$\underline{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{S_1}{J_1} & \frac{S_1}{J_1} & -\frac{B_1}{J_1} & \frac{B_1}{J_1} \\ \frac{S_1}{J_2} & -\frac{S_1 + S_2}{J_2} & \frac{B_1}{J_2} & -\frac{B_1 + B_2}{J_2} \end{bmatrix}$$

$$\underline{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{J_1} & 0 \\ 0 & -\frac{K_1}{J_2} \end{bmatrix} ; \quad \underline{C} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (\text{A7.3.1d})$$

The system transfer function matrix

$$G(s) = \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix} \quad (A7.3.2.a)$$

will be given by

$$G(s) = C[sI - A]^{-1} B = \frac{1}{|sI - A|} \begin{bmatrix} \frac{1}{J_1} A_{33} & -\frac{K_1}{J_2} A_{43} \\ \frac{1}{J_1} A_{34} & -\frac{K_1}{J_2} A_{44} \end{bmatrix} \quad (A7.3.2b)$$

where

$$|sI - A| = s^4 + s^3 \left(\frac{B_1}{J_1} + \frac{B_1 + B_2}{J_2} \right) + s^2 \left(\frac{S_1}{J_1} + \frac{S_1 + S_2}{J_2} + \frac{B_1 B_2}{J_1 J_2} \right) +$$

$$s \left(\frac{S_1 B_2 + S_2 B_1}{J_1 J_2} \right) + \frac{S_1 S_2}{J_1 J_2}$$

$$A_{33} = s^3 + s^2 \left(\frac{B_1 + B_2}{J_2} \right) + s \left(\frac{S_1 + S_2}{J_2} \right)$$

$$A_{43} = s^2 \frac{B_1}{J_1} + s \frac{S_1}{J_1} \quad (A7.3.2c)$$

$$A_{34} = s^2 \frac{B_1}{J_2} + s \frac{S_1}{J_2}$$

$$A_{44} = s^3 + s^2 \frac{B_1}{J_1} + s \frac{S_1}{J_1}$$

The transfer functions of the main and coupling elements can now be obtained. For the simplified 2-input 2-output model of Fig. 7.5 :

$$G_{11}(s) = \frac{1}{|sI - A|} \frac{1}{J_1} A_{33} = \frac{s^3 b_3 + s^2 b_2 + s b_1}{s^4 + s^3 a_3 + s^2 a_2 + s a_1 + a_0} \quad (A7.3.3a)$$

$$G_{12}(s) = \frac{1}{|s I - A|} \begin{pmatrix} -\frac{K_1}{J_2} & A_{43} \end{pmatrix} = \frac{s^2 b_2' + s b_1'}{s^4 + s^3 a_3 + s^2 a_2 + s a_1 + a_0} \quad (\text{A7.3.3b})$$

The coefficients of the polynomials of the numerator and denominator, which depend on the parameter values, can be found by replacing Eqns. A7.3.2c into Eqns. A7.3.3. The values used for J_1 , J_2 , S_1 , S_2 , B_1 , B_2 , and K_1 are presented in Table A7.2c). Since an approximated expression was used for the electrical torque

$$T_1 = K_2 \frac{\omega}{R_1(\omega)} = K_1 \frac{1}{R_1} \quad (\text{A7.3.4})$$

the value of the non-linear parameter $K_1(\omega, R_1)$, which expresses the gain of the electrical actuation scheme, was calculated from generator test data presented in [5.1.].

7.4. LINEARISED STATE-SPACE MODEL

Since the model presented by Eqns. 7.6. has non-linear parameters $S_1(T_w)$, $B_2(\omega_2)$ and $S_2(T_w, \omega_1)$ those equations have to be generalised to

$$\left\{ \begin{array}{l} \dot{x}_1 = f_1(x_o, u_o) + \frac{\partial f_1(x, u)}{\partial x_3} \Big|_{x_o, u_o} \Delta x = x_{30} + \Delta x_3 \\ \dot{x}_2 = \dots = x_{40} + \Delta x_4 \\ \dot{x}_3 = f_3(x_{10}, x_{20}, x_{30}, x_{40}, u_o) + \sum_{i=1}^4 \frac{\partial f_3(x, u)}{\partial x_i} \Big|_{x_o, u_o} \Delta x_i + \frac{\partial f_3(x, u)}{\partial u} \Big|_{x_o, u_o} \Delta u \\ \dot{x}_4 = \dots \end{array} \right. \quad (\text{A7.4.1a})$$

or

$$\left\{ \begin{array}{l}
 \dot{x}_1 = x_{30} + \Delta x_3 \\
 \dot{x}_2 = x_{40} + \Delta x_4 \\
 \dot{x}_3 = -\frac{S_{10}}{J_1} x_{10} + \frac{S_{10}}{J_1} x_{20} - \frac{B_1}{J_1} x_{30} + \frac{B_1}{J_1} x_{40} + \frac{1}{J_1} T_{wo} - \\
 \quad - \frac{S_1}{J_1} \Delta x_1 + \frac{S_1}{J_1} \Delta x_2 - \frac{B_1}{J_1} \Delta x_3 + \frac{B_1}{J_1} \Delta x_4 + \frac{1}{J_1} \Delta T_w \\
 \dot{x}_4 = \dots\dots
 \end{array} \right. \quad (A7.4.1b)$$

the small-signal model will therefore be given by :

$$\left\{ \begin{array}{l}
 \Delta \dot{x}_1 = \Delta x_3 \\
 \Delta \dot{x}_2 = \Delta x_4 \\
 \Delta \dot{x}_3 = -\frac{S_1}{J_1} \Delta x_1 + \frac{S_1}{J_1} \Delta x_2 - \frac{B_1}{J_1} \Delta x_3 + \frac{B_1}{J_1} \Delta x_4 + \frac{1}{J_1} \Delta T_w \\
 \Delta \dot{x}_4 = \frac{S_1}{J_2} \Delta x_1 - \frac{1}{J_2} (S_1 + S_2) \Delta x_2 + \frac{B_1}{J_2} \Delta x_3 - \frac{1}{J_2} (B_1 + B_2) \Delta x_4 - \\
 \quad \frac{K_1}{J_2} \Delta \left(\frac{1}{R_1} \right)
 \end{array} \right. \quad (A7.4.2a)$$

or

$$\Delta \dot{\underline{x}} = \underline{A}^* \Delta \underline{x} + \underline{B}^* \Delta \underline{u} \quad (A7.4.2b)$$

The coefficients of the matrices $\underline{A}$ and $\underline{B}$ are equal to those of the Jacobian matrices $\underline{A}^*$ and $\underline{B}^*$, because the non-linear parameters depend of variables which do not have them as part of their coefficients.

S_1 , B_2 and S_2 will have different values for each operating point, but

it will be assumed they are not significantly altered for small perturbations around that operating point.

7.5. STATIC COUPLING FACTOR

The dynamic coupling factor is defined by:

$$K(s) = \frac{G_{12}(s) G_{21}(s)}{G_{11}(s) G_{22}(s)} \quad (\text{A7.5.1a})$$

with each of the $G_{ij}(s)$ given by Eqns. A7.3.2, it will be simply

$$K(s) = \frac{A_{34} A_{43}}{A_{33} A_{44}} = \frac{(s^2(B_1/J_2) + s(S_1/J_2))(s^2(B_1/J_1) + s(S_1/J_1))}{(s^3 + s^2(B_1+B_2)/J_2 + s(S_1+S_2)/J_2)(s^3 + s^2(B_1/J_1) + s(S_1/J_1))} \quad (\text{A7.5.1b})$$

The static coupling factor is :

$$K_o = K(1) = \frac{G_{12}(1) G_{21}(1)}{G_{11}(1) G_{22}(1)} = \frac{K_{12} K_{21}}{K_{11} K_{22}} \quad (\text{A7.5.2})$$

The expressions for the coupling gains are as follows

$$K_{11} = 1 + \frac{S_1 + S_2 + B_1 + B_2}{J_2} \quad ; \quad K_{12} = \frac{S_1 + B_1}{J_1} \quad (\text{A7.5.3})$$

$$K_{21} = \frac{S_1 + B_1}{J_2} \quad ; \quad K_{22} = 1 + \frac{S_1 + B_1}{J_1}$$

$$K_o = \frac{(B_1 + S_1)^2}{(J_1 + S_1 + B_1)(J_2 + S_1 + S_2 + B_1 + B_2)}$$

Taking the appropriate parameter values for each operating point the values of K_{11} , K_{12} , K_{21} , K_{22} and K_o will be as in Table A7.5.

7.6. TRANSIENT RESPONSE OF THE UNCONTROLLED WECS

A second-order approximation to WECS transient response to wind torque changes is now presented. The values used for S_g , B_g express the WECS loading for each of the operating points considered. The inherent turbine speed feedback loop is taken into account by assuming the WECS closed-loop behaviour as in Fig. 7.6. The method used is now briefly described. From the closed-loop poles of the uncontrolled WECS :

$$\left\{ \begin{array}{l} s_{1,2} = -\sigma \pm j\omega \\ s_{3,4} = -\sigma' \pm j\omega' \end{array} \right. \quad (A7.6.1)$$

the value of the ratio σ/σ' allows us to conclude that $s_{1,2}$ are the dominant poles. System transient response will therefore be dictated by this pair of complex conjugate poles and a second order approximation is allowed.

For this pole pair the following relations apply :

$$\left\{ \begin{array}{l} \sigma = \zeta \omega_n \\ \omega = \omega_n \sqrt{1-\zeta^2} \\ \theta = \cos^{-1} \zeta \end{array} \right. \quad (A7.6.2)$$

where ζ is the damping ratio and ω_n the natural frequency of the complex pole pair. From the pole position we can determine :

	closed-loop poles s _{1,2} , s _{3,4}	σ _{3,4} /σ _{1,2}	ζ	ω _n (rad.s ⁻¹)	t _{max} (sec.s)	% overs.	t _d	t _r	t _s	envelope at t=0
							sec. ^s (cycles)			
(i) K=1.08	-0.138 ± j5.777 -8.163 ± j69.518	59	0.024	5.780	0.094	92.8	0.030	0.031	3.774 (20)	1± 1.00028
(ii) K=0.54	-0.095 ± j5.749 -8.182 ± j57.388	86	0.016	5.750	0.095	95.0	0.030	0.031	5.511 (29)	1± 1.00014
(iii) K=1.20	-0.260 ± j5.166 -7.841 ± j36.171	30	0.050	5.173	0.118	85.4	0.039	0.040	2.230 (9.5)	1± 1.00127

Table A7.6 - Characterising parameters for a second order approximation of the uncontrolled system transient response.

	K_{11}	K_{12}	K_{21}	K_{22}	K_o
(i)	4128.0	333.1	4122.1	334.1	0.996
(ii)	1351.4	108.7	1345.1	109.7	0.986
(iii)	350.8	27.7	342.7	28.7	0.943

Table A7.5 - Coupling gains and static coupling factor for the WECS multivariable model.

	K_1/K_2	K_3/K_4
(i)	± j0.054678	± j0.000496
(ii)	± j0.053167	± j0.000617
(iii)	± j0.045957	± j0.001272

Table A7.7 - Partial fraction expansion constants.

$$\begin{cases} \zeta = \cos \theta = \cos \left(\frac{\pi}{2} - \text{tg}^{-1}(\sigma/\omega) \right) \\ \omega_n = \frac{\sigma}{\zeta} \end{cases} \quad (\text{A7.6.3})$$

The WECS step response can then be characterised by a number of selected parameters [7.20.] :

- the time to the first peak

$$t_{\max} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} \quad (\text{A7.6.4a})$$

- the % overshoot

$$c_{\max}^{-1} = e^{-\pi\zeta/\sqrt{1-\zeta^2}} \quad (\text{A7.6.4b})$$

- the delay, rise and settling times

$$t_d = \frac{1 + 0.6\zeta + 0.15\zeta^2}{\omega_n} \quad (\text{A7.6.4b})$$

$$t_r = \frac{1 + 1.1\zeta + 1.4\zeta^2}{\omega_n} \quad ; \quad t_s = \frac{3}{\zeta\omega_n} \quad (\text{A7.6.4c})$$

- the envelope value at $t=0$

$$1 \pm \frac{1}{\sqrt{1-\zeta^2}} \quad (\text{A7.6.4d})$$

The characterising parameter values are presented in Table A7.6.

7.7. DISCRETIZATION OF THE CONTINUOUS PLANT

$G_p(s)/s$ is given by Eqn. 7.18 as a quotient between polynomials whose coefficients can be determined through Eqns. A7.3.2. System discretization starts by transforming $G_p(s)/s$ into the z domain. This is done by expanding the Laplace transform expression into partial fractions and then directly transforming each of those into the z plane. From Eqns. 7.19 and 7.20 we need to determine the values of the residues K_1 to K_4 such that

$$\frac{(s - \alpha)}{(s - \alpha')(s - \beta')(s - \gamma')(s - \delta')} = \frac{K_1}{(s - \alpha')} + \frac{K_2}{(s - \beta')} + \frac{K_3}{(s - \gamma')} + \frac{K_4}{(s - \delta')} = \frac{P(s)}{Q(s)} \quad (\text{A7.7.1})$$

The residues are given by

$$K_i = \left[(s - s_i) \frac{P(s)}{Q(s)} \right] \Big|_{s=s_i} \quad (\text{A7.7.2a})$$

where the poles s_i are equal to α' , β' , γ' and δ' .

We will therefore have

$$\begin{aligned} K_1 &= \frac{(\alpha' - \alpha)}{(\alpha' - \beta')(\alpha' - \gamma')(\alpha' - \delta')} & K_2 &= \frac{(\beta' - \alpha)}{(\beta' - \alpha')(\beta' - \gamma')(\beta' - \delta')} \\ K_3 &= \frac{(\gamma' - \alpha)}{(\gamma' - \alpha')(\gamma' - \beta')(\gamma' - \delta')} & K_4 &= \frac{(\delta' - \alpha)}{(\delta' - \alpha')(\delta' - \beta')(\delta' - \gamma')} \end{aligned} \quad (\text{A7.7.2b})$$

Since we have two pairs of complex conjugate poles the residues are also complex conjugates. Table A7.7 presents the values they assume for the various operating points.

The z transform of $G_p(s)/s$ can now be determined

$$G'(z) = K_{wT} \frac{z-1}{z} z \left[\frac{K_1}{s-\alpha'} + \frac{K_2}{s-\beta'} + \frac{K_3}{s-\gamma'} + \frac{K_4}{s-\delta'} \right] = \quad (A7.7.3a)$$

$$= K_{wT} \frac{z-1}{z} \left[K_1 z \left\{ \frac{1}{s+\alpha'_1} \right\} + K_2 z \left\{ \frac{1}{s+\beta'_1} \right\} + K_3 z \left\{ \frac{1}{s+\gamma'_1} \right\} + K_4 z \left\{ \frac{1}{s+\delta'_1} \right\} \right]$$

where $\alpha'_1 = -\alpha'$, $\beta'_1 = -\beta'$, $\gamma'_1 = -\gamma'$ and $\delta'_1 = -\delta'$.

Using z transform tables we obtain :

$$G'(z) = K_{wT} \frac{z-1}{z} \left[K_1 \frac{z}{\left(\frac{z-e^{-\alpha'_1 T}}{z-e} \right)} + K_2 \frac{z}{\left(\frac{z-e^{-\beta'_1 T}}{z-e} \right)} + K_3 \frac{z}{\left(\frac{z-e^{-\gamma'_1 T}}{z-e} \right)} + K_4 \frac{z}{\left(\frac{z-e^{-\delta'_1 T}}{z-e} \right)} \right] \quad (A7.7.3b)$$

After some algebra we arrive to Eqn. 7.21, i.e.

$$G'(z) = K_{wT} (z-1) \frac{A_3 z + A_2 z + A_1 z + A_0}{(z-e^{-\alpha'_1 T})(z-e^{-\beta'_1 T})(z-e^{-\gamma'_1 T})(z-e^{-\delta'_1 T})} \quad (A7.7.3c)$$

where

$$A_3 = K_1 + K_2 + K_3 + K_4$$

$$A_2 = -K_1 \left(e^{\beta'_1 T} + e^{\gamma'_1 T} + e^{\delta'_1 T} \right) - K_2 \left(e^{\alpha'_1 T} + e^{\gamma'_1 T} + e^{\delta'_1 T} \right) - K_3 \left(e^{\alpha'_1 T} + e^{\beta'_1 T} + e^{\delta'_1 T} \right) - K_4 \left(e^{\alpha'_1 T} + e^{\beta'_1 T} + e^{\gamma'_1 T} \right)$$

$$A_1 = K_1 \left(e^{(\beta'+\gamma')T} + e^{(\beta'+\delta')T} + e^{(\gamma'+\delta')T} \right) + K_2 \left(e^{(\alpha'+\gamma')T} + e^{(\alpha'+\delta')T} + e^{(\gamma'+\delta')T} \right) + K_3 \left(e^{(\alpha'+\beta')T} + e^{(\alpha'+\delta')T} + e^{(\beta'+\delta')T} \right) + K_4 \left(e^{(\alpha'+\beta')T} + e^{(\alpha'+\gamma')T} + e^{(\beta'+\gamma')T} \right)$$

$$A_o = -K_1 e^{(\beta' + \gamma' + \delta')T} - K_2 e^{(\alpha' + \gamma' + \delta')T} - K_3 e^{(\alpha' + \beta' + \delta')T} - K_4 e^{(\alpha' + \beta' + \gamma')T}$$

(A7.7.3d)

The open-loop poles and zeros of the discretized system can be found from the respective transfer function.

7.8. Z-PLANE CLOSED-LOOP TRANSFER FUNCTION

The system closed-loop transfer function given by Eqn. 7.22 is

$$H(z) = \frac{D(z) G'(z)}{1 + D(z) G'(z)} \quad (\text{A7.8.1a})$$

Substituting the Eqns. 7.21 and 7.24 into Eqn. 7.22, the following expression is obtained :

$$H(z) = \frac{c_4 z^4 + c_3 z^3 + c_2 z^2 + c_1 z + c_o}{z^5 + (a_4' + c_4) z^4 + (a_3' + c_3) z^3 + (a_2' + c_2) z^2 + (a_1' + c_1) z + c_o} \quad (\text{A7.8.1b})$$

where the a_i 's are the coefficients of the polynomial in the denominator of $G'(z)$, i.e.

$$z^4 + a_4' z^3 + a_3' z^2 + a_2' z + a_1' = (z - e^{\alpha' T}) (z - e^{\beta' T}) (z - e^{\gamma' T}) (z - e^{\delta' T}) \quad (\text{A7.8.2a})$$

and the b_i 's, which depend on the compensator parameters, are given by

$$\begin{aligned} c_4 &= C (T K_p + K_d) \\ c_3 &= C \left[(A_1 - 1) T K_p + (A_1 - 2) K_d + T^2 K_i \right] \\ c_2 &= C \left[(A_o - A_1) T K_p + (A_o - 2A_1 + 1) K_d + A_1 T^2 K_i \right] \end{aligned} \quad (\text{A7.8.2b})$$

$$c_1 = C \left[-A_0 T K_p + (A_1 - 2 A_0) K_d + A_0 T^2 K_i \right]$$

$$c_0 = C A_0 K_d$$

where

$$C = \frac{K_{WT} A_2}{T} \quad (A7.8.2c)$$

7.9. SYSTEM PERFORMANCE SPECIFICATIONS

The rise time (t_r) is the time for the step response to rise from 10% to 95% of the final value. The settling time (t_s) or system response time is the time for the step response to settle within 1% of the final value.

7.10. RAMP RESPONSE BY THE LONG DIVISION METHOD

System response to a ramp input is given by Eqn. 7.30.

$$Y(z) = \frac{b_4'' z^5 + b_3'' z^4 + b_2'' z^3 + b_1'' z^2 + b_0'' z}{z^7 + a_6'' z^6 + a_5'' z^5 + a_4'' z^4 + a_3'' z^3 + a_2'' z^2 + a_1'' z + a_0''} \quad (A7.10.1a)$$

where the a's are given by

$$\begin{aligned}
a_6'' &= a_4' + c_4 - 2 & a_3'' &= a_1' - 2a_2' + a_3' \\
a_5'' &= a_3' - 2a_4' + 1 & a_2'' &= a_0' - 2a_1' + a_2' \\
a_4'' &= a_2' - 2a_3' + a_4' & a_1'' &= a_1' - 2a_0' \\
a_0'' &= a_0'
\end{aligned} \tag{A7.10.1b}$$

and the b's simply by $b_i'' = T c_i$.

Dividing out the numerator and the denominator polynomials the corresponding time function is

$$Y(z) = b_4'' z^{-2} + (c_3 - a_6 c_4) z^{-3} + |c_2 - a_5 c_4 - a_6 (c_3 - a_6 c_4)| z^{-4} + \dots \tag{A7.10.1c}$$

The expressions obtained for the coefficients of z^{-k} are increasingly bigger and of cumbersome evaluation. The expression for each coefficient of order k can however be seen to include all the previous lower order coefficients from 2 to $k-1$.

In fact with the response given by Eqn. 7.31

$$Y(z) = C_2 z^{-2} + C_3 z^{-3} + C_4 z^{-4} + \dots \tag{A7.10.1d}$$

it can be shown that

$$\begin{aligned}
C_2 &= c_4 & C_3 &= c_3 - a_6 C_2 & C_4 &= c_2 - a_5 C_2 - a_6 C_3 \\
C_5 &= c_1 - a_4 C_2 - a_5 C_3 - a_6 C_4 & & \text{etc.}
\end{aligned} \tag{A7.10.1e}$$

And, for all C's for which $K > 10$ the following recursive formula can be used

$$\begin{aligned}
C_k &= -a_0 C_{k-7} - a_1 C_{k-6} - a_2 C_{k-5} - a_3 C_{k-4} - a_4 C_{k-3} - a_5 C_{k-2} - a_6 C_{k-1} \\
k &= 11, 12, \dots
\end{aligned} \tag{A7.10.1f}$$

REFERENCES

CHAPTER 1

- 1.1. - BWEA (1982)
Wind Energy for the Eighties.
Peter Peregrinus.

- 1.2. - FRERIS, L.L. (1985)
A plain man's guide to electrical aspects of WECS.
7th BWEA Conf. March 27-29, Oxford.

- 1.3. - CASANOVA, V.H. (1984)
Dynamic analysis and control of Wind Energy Conversion Systems.
Phd Thesis, Dep. Elect. Eng., Imperial College, London.

- 1.4. - FRERIS, L.L. et al (1979)
A low cost wind energy conversion system for heating of domestic premises.
IEE Conf. on Future Energy Concepts, Jan. 79, London.

- 1.5. - BUEHRING, I.K. and FRERIS, L.L. (1980)
Some aspects of small aerogenerator design and testing.
3rd Int Symp. on Wind Energy Systems, Aug. 80, Lyngby, Denmark.

- 1.6. - DALY, S.A. et al (1983)
Modelling and control of a wind-driven induction generator for water storage heating.
IEE Proc., Vol. 130, Pt. A, No. 9, Dec. 83.

- 1.7. - de PAOR, A.M. et al (1983)
Parameter Identification and Passive Control of a Wind-Driven Permanent Magnet Alternator System for Heating Water.
Wind Engineering, Vol. 7, No. 4, 1983.

- 1.8. - BUEHRING, I.K. and FRERIS, L.L. (1981)
Control policies for wind-energy conversion systems.
IEE Proc., Vol. 128, Pt.C, No. 5, September 1981.

- 1.9. - BOLTON, H.R. and NICODEMOU, V.C. (1979)
Permanent magnet alternators for small wind systems.
1st BWEA Conf., April 1979, Cranfield.
- 1.10. - TSITSOVITS, A.J. and FRERIS, L.L. (1983)
Dynamics from an isolated power system supplied from diesel and
wind.
IEE Proc., Vol. 130, Pt. A, No. 9, Dec. 83.

CHAPTER 2

- 2.1. - SVENSSON, J.E. and ULEN, E. (1982)
The control system of WTS-3. Instrumentation and testing.
4th Int. Symp. on Wind Energy Systems, Sep. 21-24, Stockholm.
- 2.2. - CRAMER, G. et al (1984)
Adaptive control and operational supervision for the 100 kW
Wind Energy Plant DEBRA.
European Wind Energy Conf., Oct. 22-26, Hamburg.
- 2.3. - MACHIELSE, L. (1984)
The 15m vertical axis wind turbine Pionier I - Description and
experiments.
European Wind Energy Conf., Oct. 22-26, Hamburg.
- 2.4. - HENRIKSEN, T. (1984)
Description of the control system for WTS 75 NAESUDDEN
European Wind Energy Conf., Oct. 22-26, Hamburg.
- 2.5. - MULLER, M. et al (1984)
A computer controlled Wind Energy Converter.
European Wind Energy Conf., Oct. 22-26, Hamburg.
- 2.6. - MAYS, I.D. and RHODES, H. (1984)
Progress with the U.K. vertical axis wind turbine programme.
European Wind Energy Conf., Oct. 22-26, Hamburg.
- 2.7. - ARMSTRONG, J.R.C. et al (1984)
Technology development for and performance of the WEG 25m
turbine.
7th BWEA Conf., March 27-29, Oxford.
- 2.8. - BEDFORD, L.A.W. et al (1985)
The Orkney 60m Horizontal Axis Wind Turbine Generator - A
Progress Report.
7th BWEA Conf, March 27-29, Oxford.

CHAPTER 3

- 3.1. - PAGE-JONES, M. (1980)
The Practical Guide to Structured Systems Design.
Yourdon Press.

- 3.2. - ALLWORTH, S.T. (1981)
Introduction to Real-time Software Design.
Macmillan Press.

- 3.3. - PARRISH, E.A. and HUANG, V.K.L. (1978)
A Scheduler for Real-Time Task Control in Microcomputers.
IEEE Trans. Ind. Elecs. Control Inst., Vol. IECI-25, No.1, Feb 1978.

- 3.4. - HERZOG, J.H. (1983)
A Design Perspective for Real-Time Task Control in
Distributed Systems.
IEEE Trans. Ind. Elecs., Vol. IE-30, No.1, Feb 1983.

- 3.5. - LISTER, A.M. (1980)
Fundamentals of Operating Systems.
Macmillan Press, 2nd ed.

- 3.6. - HOLT, R.C. et al (1978)
Structured Concurrent Programming with Operating Systems
Applications.
Addison-Wesley.

- 3.7. - BRINCH HANSEN, P. (1977)
The Architecture of Concurrent Programs.
Prentice-hall.

- 3.8. - BEN-ARI, M. (1982)
Principles of Concurrent Programming.
Prentice-hall.

CHAPTER 4

- 4.1. - HP-64000 LDS USER MANUAL
- 4.2. - HP (1980)
Hewlett-Packard Journal.
Vol. 31, Part 10, 1980.
- 4.3. - HP (1983)
Hewlett-Packard Journal.
Vol. 34, Part 3, 1983.
- 4.4. - Edited by V. TSENG (1982)
Microprocessor Development and Development Systems.
Granada Publ.
- 4.5. - JENSEN, K. and WIRTH, N. (1978)
Pascal User Manual and Report.
Springer-Verlag, 2nd ed.
- 4.6. - WELSH, J. and McKEAG, M. (1980)
Structured System Programming.
Prentice-Hall.
- 4.7. - WELSH, J. and ELDER, J. (1979)
An Introduction to Pascal.
Prentice-Hall.
- 4.8. - SCHNEIDER, G.M. and BRUELL, S.C. (1981)
Advanced Programming and Problem Solving Using Pascal.
John Wiley & Sons.
- 4.9. - WAKERLY, J.F. (1981)
Microcomputer Architecture and Programming.
John Wiley & Sons.

- 4.10. - KANE, G. et al (1981)
68000 Assembly Language Programming.
Osborne/McGraw-Hill.
- 4.11. - MOTOROLA SEMIC. (1983)
Advance Information Data Sheet on the MC68000.
- 4.12. - ELEKTOR (1981)
16-bit microprocessors.
Special supplement, April 1981.
- 4.13. - DE PRYCKER, M. (1983)
A Performance Comparison of Three Contemporary 16-bit
Microprocessors.
IEEE Micro, April 1983.
- 4.14. - MOTOROLA SEMIC. (1983)
MC6840 PTM Data Sheet.
- 4.15. - MOTOROLA SEMIC. (1983)
MC6850 ACIA Data Sheet.
- 4.16. - MOTOROLA SEMIC. (1983)
MC6821 PIA Data Sheet.
- 4.17. - YOUNG, S.J. (1980)
Real Time Languages - design and development.
IEE Proc., Vol. 127, Pt. E, No. 2, March 1980.
- 4.18. - YOUNG, S.J. (1980)
Low-level-device programming with a high-level language.
IEE Proc., Vol. 127, Pt. E, No. 2, March 1980.

CHAPTER 5

- 5.1. - NICODEMOU, V.C. (1979)
Small Wind-powered Electric Generators and Systems.
PhD Thesis, Dept. Elect. Eng., Imperial College, London.
- 5.2. - JONES, B.E. (1977)
Instrumentation Measurement and Feedback.
McGraw-Hill Book Co.
- 5.3. - WOOLVET, G.A. (1977)
Transducers in Digital Systems.
Peter Peregrinus.
- 5.4. - GARRET, P.H. (1984)
Analog I/O Design. Acquisition, Conversion, Recovery.
Reston Pub. Co. Inc.
- 5.5. - CHESMOND, C. (1984)
Control System Technology.
Edward Arnold.
- 5.6. - BARNEY, G.C. (1985)
Intelligent Instrumentation.
Prentice-Hall.
- 5.7. - OLIVER, F.J. (1971)
Practical Relay Circuits.
Pitman Pub.
- 5.8. - LANDER, C. (1981)
Power Electronics.
McGraw-Hill Book Co.
- 5.9. - OXNER, E.S. (1982)
Power FETs and their applications.
Prentice-Hall.

- 5.10. - HOROWITZ, P. & HILL, W. (1980)
The Art of Electronics.
Cambridge Univ. Press.
- 5.11. - ARTWICK, B.A. (1980)
Microcomputer Interfacing.
Prentice-Hall.
- 5.12. - CLEMENTS, A. (1982)
Microcomputer Design and Construction.
Prentice-Hall Int. Inc.
- 5.13. - LADUZINSKY, A. (1984)
VMEbus Evolves for Control Applications.
Control Engineering, August 1984.
- 5.14. - BLACK, J. (1984)
VMEbus Family : A 32-bit Standard for Board-level Microcomputer
Systems.
Control Engineering, August 1984.
- 5.15. - MVME110/D2 MOTOROLA PUB. (1983)
Monoboard Microcomputer User's Manual, 2nd ed.
- 5.16. - MVME600/D1 MOTOROLA PUB. (1983)
Analog Input Module User's Manual.
- 5.17. - MVME410/D1 MOTOROLA PUB. (1983)
Dual Parallel Module User's Manual.

CHAPTER 7

- 7.1. - COOPER, B.J. (1983)
A Review of Control System Options for WECS.
Proc. 5th BWEA Wind Energy Workshop, Reading, U.K.

- 7.2. - HINRICHSSEN, E.N. (1984)
Controls for variable pitch wind turbine generators.
IEEE Trans. on Power App. and Syst., Vol. PAS-103, April 1984.

- 7.3. - KUO, B.C. (1980)
Digital Control Systems.
Holt, Reinhart and Winston.

- 7.4. - FRANKLIN, G.F. and POWELL, D.J. (1980)
Digital Control of Dynamic Systems.
Addison-Wesley.

- 7.5. - KATZ, P. (1981)
Digital control using microprocessors.
Prentice-Hall.

- 7.6. - LEIGH, J.R. (1985)
Applied Digital Control - Theory, Design and Implementation.
Prentice-Hall.

- 7.7. - ASTROM, K.J. and WITTENMARK, B. (1984)
Computer Controlled Systems - Theory and Design.
Prentice-Hall.

- 7.8. - TAYLOR, G.J. (1983)
Wake and performance measurements on the Lawson-Tancred 17 m
horizontal-axis windmill.
IEE Proc., Vol. 130, Pt.A, no. 9, December, 1983.

- 7.9. - SIMKOVITS, H.R. and KASSAKIAN, J.G. (1977)
Control and dynamic analysis of a wind energy conversion and
storage system operating at constant velocity ratio.
IEEE Energy Develop. III Special Pub., CH 1215-3-PWR, Jan 1977.

- 7.10. - MURDOCH, A. et al (1983)
Control Design and Performance Analysis of a 6 MW Wind
Turbine-Generator.
IEEE Trans. on Power App. and Syst., Vol. PAS-102, May 1983.
- 7.11. - GARRAD, A.D. (1981)
Wind Turbine Transmission Systems.
Proc. 3rd BWEA Wind Energy Workshop, Cranfield, U.K.
- 7.12. - WASYNCZUK, O. et al (1981)
Dynamic behaviour of a class of wind turbine generators during
random wind fluctuations.
IEEE PES Winter Meeting, Atlanta, Feb. 1-6, 1981.
- 7.13. - HERAPATH, R.G. and SHI, F.K.C. (1981)
Dynamics and control of an experimental 5 m diameter variable
pitch wind powered generator.
IEE 3rd Int. Conf. on Future Energy Concepts, London.
- 7.14. - JOHNSON, C.G. (1976)
Dynamics of Wind Generators on Electric Utility Networks.
IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-12
no.4, July, 1976.
- 7.15. - SWANSBOROUGH, R.H. and BALLARD, L.J. (1981)
Rotational Dynamics of Wind Turbine Generators.
IEE 3rd Int. Conf. on Future Energy Concepts, London.
- 7.16. - RICHARDS, R.J. (1979)
An introduction to dynamics and control.
Longman.
- 7.17. - SMITH, C.L. (1972)
Digital Computer Process Control.
Intext Educational Publ.
- 7.18. - ISERMANN, R. (1981)
Digital Control Systems.
Springer-Verlag.

- 7.19. - LEIGH, J.R. (1982)
Applied Control Theory.
Peter Peregrinus/IEE Cont.Eng.Series 18.
- 7.20. - KUO, B.C. (1981)
Automatic Control Systems.
Prentice-Hall, 4th ed.
- 7.21. - CLARKE, D.W. (1984)
PID algorithms and their computer implementation.
Trans. Inst., Meas. and Control, Vol 6, no. 6, Oct-Dec 1984.
- 7.22. - TSITSOVITS, A.J. (1984)
Dynamics and control of power systems fed from diesel and
wind generators.
Phd. thesis, E.E. Dept., Imperial College, Univ. of London.

CHAPTER 8

- 8.1. - BORIANI, D.V. (1984)
On-line frequency control in an integrated wind/diesel
conversion system.
Msc Thesis, Dept. Elect. Eng., Imperial College, London.
- 8.2. - BOSSANYI, E.A. (1985)
Stochastic wind prediction for wind turbine system control.
7th BWEA Conf., March 27-29, Oxford.
- 8.3. - ASTROM, K.J. and ANTON, J.J. (1984)
Expert Control.
IFAC 9th World Congress, July 1984, Budapest.