The reliability of small digital controllers

Jonathon C. Pearson

THE RELIABILITY OF SMALL DIGITAL CONTROLLERS

by

Jonathan C.Pearson BSc(Eng), ACGI

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

University of Durham

September 1983

Dedicated to

MY MOTHER AND FATHER

ABSTRACT

Increasing use is being made of small digital controllers in Industry and Commerce. The failure of such controllers is important since it may cause either plant to become unsafe or the interruption of production. Fault-tolerant techniques are discussed for improving the reliability of digital controllers with special reference to the development of a hybrid electromechanical gas governor, whose electronic controller is an example of a small digital controller. Three microprocessors are used in a two out of three majority voting configuration and the memory is Hamming code protected. Redundancy techniques are used to protect against faults in other parts of the controller and it will tolerate most classes of transient fault.

When comparing designs or attempting to meet reliability criteria, it is necessary to predict the reliability of a system and its individual components. Several sources of failure rate prediction are compared and the wide variation in the failure rates of integrated circuits is highlighted. The comparison concludes by recommending which reliability data source is likely to be most accurate for each type of component.

The gas governor is an example of a repairable system and analysis is developed for predicting the improvement in reliability for repairable redundant systems and for determining the optimum maintenance and repair times for equipment.

The testing of redundant systems is difficult because of their complexity, and under certain circumstances the redundancy can mask design faults. Testing methods using complex test equipment are described, as well as the testing of the experimental controller.

A review is included of other fault-tolerant systems. Although the work on large computers is not directly applicable to small controllers, many of the techniques can be used.

Acknowledgements

I should like to express my gratitude to the following people. To Dr Clive Preece, my Supervisor, for his help, encouragement, friendship, and assistance in the preparation of this Thesis. To the British Gas Corporation for the provision of a Research Scholarship and equipment. To Dr Ken Jenkins for his help, encouragement, and the acquisition of equipment, and also for the close links between the Engineering Research Station and the Department of Engineering. I would like to thank Mr Robert Halse for his help and discussion. To Dr David Smith for his interest and valuable discussion about this research. To the Electrical Technicians for jobs too numerous to mention and to Motorola for providing much useful reliability information. Finally I would like to thank Gillian for her love and encouragement in the preparation of this Thesis.

CONTENTS

Page

LIST OF FIGURES ix

LIST OF TABLES xi

LIST OF SYMBOLS xii

CHAPTER 1 The Need for a Reliable Controller 1

1.1 NATIONAL GAS NETWORK 1

1.2 PNEUMATIC GOVERNORS 2

1.3 LIMITATIONS OF PNEUMATIC GOVERNORS 4

CHAPTER 2 A Review of Other Fault-tolerant Controllers 7

2.1 SMALL CONTROLLERS 7

2.1.1 TMR controllers 7

2.1.2 Fail-safe controller 8

2.1.3 Built in test equipment 9

2.2 LARGE COMPUTERS 9

2.2.1 Space shuttle and avionics 9

2.2.2 Commercial computers 11

2.2.3 Telephone exchange computers 12

2.2.4 Safety monitoring computer 12

2.2.5 PDP11 computer 13

2.2.6 Railway applications 13

2.2.7 Fault-tolerant software 14

2.3 SUMMARY 14

CHAPTER 3 Failure of Components 17

3.1 FAILURE DISTRIBUTION AND MECHANISMS 17

3.2 FAILURE RATE MEASUREMENT AND ACCELERATION 18

3.2.1 Accelerated testing 19

3.3 FAILURE RATE PREDICTION 21

3.3.1 CNET 23

3.3.2 National centre of systems reliability 24

3.3.3 RADC field data 25

3.3.4 European space agency 25

3.3.5 Manufacturers testing 25

3.3.6 Simple models 26

3.3.7 GIDEP computer data base 26

3.3.8 EuReData 27

3.3.9 MIL-217 data base 27

3.4 COMPARISON OF FAILURE RATE DATA 27

3.4.1 Resistors 28

3.4.2 Capacitors 28

3.4.3 Soldered joints 28

3.4.4 Wire-wrap connections 28

3.4.5 Edge connectors 29

3.4.6 Integrated circuits 29

3.4.7 TTL integrated circuits 32

3.4.8 6800 microprocessor 32

3.4.9 8080 microprocessor 33

3.4.10 EPROM 34

3.4.11 Bipolar ROMs 34

3.4.12 Dynamic RAM 34

3.4.13 Static RAMs 34

3.4.14 Comparison with other field data 35

3.4.15 Recommendations 36

3.5 COMPARISON OF DIFFERENT DEVICE TECHNOLOGIES AND ENCAPSULATION 37

3.5.1 CMOS versus TTL 38

3.5.2 Hermetic versus non-hermetic encapsulation 39

3.5.3 Recommendations 43

3.6 SCREENING AND METHODS OF IMPROVING FAILURE RATES 44

3.6.1 Derating and cooling 46

3.7 SOFTWARE AND TRANSIENT ERRORS 47

3.8 SUMMARY 50

CHAPTER 4 Prediction of Reliability Improvement due to Fault-tolerant techniques 52

4.1 METHODS OF EXPRESSING RELIABILITY 52

4.2 EFFECT OF REDUNDANCY ON RELIABILITY 54

4.3 EFFECT OF MAINTENANCE ON A SYSTEM 55

4.3.1 Redundant systems 56

4.4 METHODS OF EXPRESSING IMPROVEMENT 59

4.5 SUMMARY 60

CHAPTER 5 Techniques for Improving a System's Fault-tolerance 62

5.1 LEVELS OF FAULT-TOLERANCE 62

5.2 DESIGN TOOLS 63

5.2.1 FMECA 63

5.2.2 FTA 64

5.3 HARDWARE 64

5.3.1 Watchdog timer 64

5.3.2 Snake 65

5.3.3 Power supply levels 66

5.3.4 Output verification 66

5.3.5 Component redundancy 66

5.3.6 Memory protection 67

5.3.7 Temperature 67

5.4 SOFTWARE 67

5.4.1 Exception handling 68

5.5 SELF-TESTING 69

5.6 SUMMARY 70

CHAPTER 6 EFFECT OF SYSTEM ARCHITECTURE ON RELIABILITY AND COST 72

6.1 CHOICE OF MICROPROCESSOR 72

6.1.1 Single chip microprocessors 72

6.1.2 Eight-bit microprocessors 72

6.1.3 Sixteen-bit microprocessors 73

6.2 MAJORITY VOTING 74

6.2.1 Hardware 74

6.2.2 Software 75

6.3 THE USE OF REDUNDANCY AND INCREASED COST 75

CHAPTER 7 Background to Design of Governor Controller 78

7.1 SPECIFICATION 78

7.2 MECHANICAL VALVE 79

7.3 NON FAULT-TOLERANT CONTROLLER 82

7.4 METHODS OF INTRODUCING FAULT-TOLERANCE 83

CHAPTER 8 Description of Experimental Controller 86

8.1 CONSTRUCTION 86

8.2 TMR MICROPROCESSOR BOARD 86

8.2.1 Microprocessor block 87

8.2.2 Voters 87

8.2.3 Self synchronising clock 88

8.2.4 Synchronisation hardware 89

8.2.5 Reset circuitry and watchdog timers 92

8.2.6 RS232 interface 94

8.3 MEMORY BOARD 94

8.3.1 Decoding 96

8.3.2 Circuit description 96

8.3.3 Address decoding and "spare" EPROMs 97

8.3.4 Memory matrix 98

8.4 INPUT/OUTPUT BOARD 98

8.4.1 Real time clock 99

8.4.2 Multiplexer and A/D converter 99

8.4.3 Stepper motor driver 100

8.4.4 Solenoid drive circuitry 100

8.4.5 Pressure transducers 101

8.5 PNEUMATIC TEST RIG 102

8.6 ESTIMATED RELIABILITY 104

8.6.1 Revised failure rate prediction 106

CHAPTER 9 Governor Controller Software 107

9.1.1 MAIN 107

9.1.2 RESYNC 108

9.1.3 MERROR 109

9.1.4 CNTRL 110

9.1.5 PRESSR 111

9.1.6 RBKGEN 111

9.1.7 BLOCK 111

9.1.8 SLFTST 112

9.1.9 SFAIL 112

9.1.10 WTRAP 112

9.1.11 SNAKE 113

9.1.12 DFAULT 113

9.1.13 INITL 113

9.1.14 COUT 113

9.1.15 COUTBF 114

9.1.16 PRBUFF 114

9.1.17 MSGE 114

9.1.18 NMOUT 114

9.1.19 TIMLOG 114

CHAPTER 10 Testing of Fault-tolerant Systems 115

10.1 SYSTEM DEBUGGING 115

10.1.1 In-circuit emulation 115

10.1.2 Logic analysis 115

10.2 FAULT INJECTION AND RECOVERY 117

10.3 OPERATIONAL TESTING

10.4 TESTING OF THE GOVERNOR CONTROLLER 118

10.4.1 Pressure control 118

10.4.2 Fault injection 119

10.4.3 Voting errors 119

10.4.4 RAM errors 120

10.4.5 Watchdogs 120

10.4.6 Snake 121

10.4.7 Solenoid failure 121

10.4.8 Pressure transducers 121

10.4.9 Interference testing 121

DISCUSSION 123

LIST OF REFERENCES 129

FIGURES 136

TABLES 174

APPENDIX 185

1. Circuit diagram of power supply 185

2. Circuit board layout 186

3.1 Error flags low pass filter calculation 190

3.2 FPLA programming table 191

4. Calculation of delay before latching error flags 192

5. Encoding ROM data 194

6. Decoding ROM data 195

7. Operation of the real time clock 196

8. Design of redundant solenoid driver circuitry 197

9. List of integrated circuits 201

10. Governor controller software listings 202

LIST OF FIGURES

Figure		Page
1	Cross section of a Donkin FIG280 regulator	136
2	Governor and distribution network	137
3	Twin stream governor	137
4	The bath-tub curve	138
5	TMR voter and error detection in open collector TTL logic	138
6	TMR ring structure	138
7	Graph of acceleration factor versus operating temperature for a reference temperature of 25°C. Plotted for different activation energies.	139
8	Failure rate of a 8085 microprocessor versus case ambient temperature	140
9	BS9400 screening procedure	141
10	Graph of simplex and TMR reliability versus normalised mission time.	142
11	Graph of MTTFIF versus ratio λ_2/λ_1	142
12	Fault tree analysis for pressure trip	143
13	Software recovery using recovery blocks	144
14	Stepper motor controlled regulator	145
15	Solenoid controlled regulator	145
16	Block diagram of non fault-tolerant controller	146
17	Op-code fetch timing showing synchronisation problem	146
18	RST/HOLD timing	146
19	Photograph of Governor Controller	147
20	Photograph of pneumatic test rig	147
21	Block diagram of microprocessor board	148
22	Microprocessor and buffers - CHANNEL 1	149
23	Microprocessor and buffers - CHANNEL 2	150
24	Microprocessor and buffers - CHANNEL 3	151
25	Voting circuitry	152
26	FPLA error signal showing the effect of filtering	153

27 Resynchronisation timing monitored by logic analyser 153

28 Clock, Test, and RS232 interface 154

29 Resynchronisation circuitry 155

30 Watchdogs and reset circuitry 156

31 Block diagram of memory board 157

32 Part of memory board including RAM and decoders 158

33 Detail of RAM storage circuitry including test circuitry 159

34 Instant ROM, EPROM, and PORT(RD) decoder circuit diagram 160

35 Block diagram of input/output board 161

36 Input/output board 162

37 Stepper motor and redundant solenoid drive circuitry 163

38 Pressure transducer and conditioning circuitry 164

39 Block diagram of pneumatic test rig 165

40 Flow chart of module MAIN 166

41 Flow chart of module RESYNC 167

42 Flow chart of module MERROR 168

43 Flow chart of module MERROR (continued) 169

44 Flow chart of module CNTRL 170

45 Flow chart of module PRESSR 171

46 Graph of governor outlet pressure versus orifice plate differential 172

47 Error messages 173

LIST OF TABLES

Table Page

1 Failure mechanisms and activation energies in MOS semiconductors 174

2 A comparison of resistor failure rates 174

3 A comparison of capacitor failure rates 175

4 A comparison of soldered joints failure rates 175

5 A comparison of wire-wrap joint failure rates 175

6 A comparison of edge connector failure rates 176

7 Activation energies used in failure rate prediction 176

8 A comparison of TTL integrated circuit failure rates 177

9 A comparison of the failure rates for a 6800 microprocessor 177

10 A comparison of 6800 microprocessor adjusted failure rates 178

11 A comparison of the failure rates for a 8080 microprocessor 178

12 A comparison of 8080 microprocessor adjusted failure rates 179

13 A comparison of the failure rates of a 2716 EPROM 179

14 A comparison of the failure rates of a 1k Bipolar ROM 180

15 A comparison of the failure rates of a 16k Dynamic RAM 180

16 A comparison of the failure rates of a 1k Static RAM 181

17 The relation between cost, screening level, and Quality factor 181

18 Recommended maximum junction temperatures for semiconductors 182

19 Signals carried by back-plane bus 183

20 Correction bits for SEC/DED Hamming code 184

21 Error position as indicated by error flags 184

List of Symbols

ASCII	American Standard Code for Information Interchange
CMOS	Complementary Metal Oxide Semiconductor
DAG	Demand Activated Governing
DBE	Double Bit Error
dil	dual in line
ECL	Emitter coupled logic
EEC	European Economic Community
EMP	Electromagnetic Pulse
EPROM	Erasable Programmable Read Only Memory
FMEA	Fault Mode and Effect Analysis
FMECA	Fault Mode and Effect Criticality Analysis
FPLA	Field Programmable Logic Array
FTA	Fault Tree Analysis
f/M hrs	failures per million hours
ICE	In Circuit Emulation
λ	failure rate
LSI	Large Scale Integration
LSTTL	Low Power Shottky Transistor Transistor Logic
MOS	Metal Oxide Semiconductor
MTBF	Mean Time Between Failures
MTFF	Mean Time to First Failure
MTTF	Mean Time To Failure
MTTFIF	Mean Time To Failure Improvement Factor
MTTR	Mean Time To Repair
NASA	North American Space Administration
POR	Power On Reset
PROM	Programmable Read Only Memory
psi	pounds per square inch
RAM	Random Access Memory
rh	relative humidity

ROM	Read Only Memory
RTC	Real Time Clock
SEC/DED	Single Bit Error Correction / Double Bit Error Detection
SSI	Small Scale Integration
STTL	Shottky Transistor Transistor Logic
THB	Temperature Humidity Bias
TMR	Triple Modular Redundancy
TTL	Transistor Transistor Logic
UART	Universal Asynchronous Receiver Transmitter
VDU	Visual Display Unit
VLSI	Very Large Scale Integration
"wg	inches water guage

CHAPTER 1

THE NEED FOR A RELIABLE CONTROLLER

1.1 NATIONAL GAS NETWORK

in the days before the extraction of natural gas, gas was produced, stored, transmitted, and used locally. Gas was transmitted at low pressure and there was no national network which made the control problem much easier.

The changeover to natural gas involved the installation of a national network. Gas comes ashore from several gas fields at high pressure and is transmitted throughout the country at high pressure, typically at 1000 psi in four foot diameter pipes. Compressor stations are used at regular intervals to overcome pressure drops in the system. Because of the extremely large volume of gas being controlled at high pressure and the absolute need for reliable control, considerable expenditure on fault tolerant main frame and mini-computers for control purposes can be justified.

Gas is progressively reduced in pressure as it is transmitted to the consumer. Transmission pressures range from 1000psi to 100psi and control of gas at the bottom end of this range is performed by pneumatics. In order to make full use of the gas network, the technique of line-pack has been developed, where the pipework itself is used to store large quantities of gas in order to smooth out variations in demand. To optimise the efficiency of techniques such as line-pack, it is necessary to replace pneumatic controllers with electronic digital controllers. These controllers must be extremely reliable because of the requirement for safe operation of the network, and failure to supply gas could involve British Gas in litigation resulting in the payment of considerable compensation. Fault tolerant controllers can achieve the levels of reliability required and many of the techniques described and demonstrated in the following chapters could be used. Although the cost of the controller must be considered more carefully at low pressures, sufficient savings can be made by the more efficient operation of the network to justify such controllers.

Gas finally reaches the "distribution" network at relatively low pressures where there are several thousand governor stations. The quantity of gas handled by each governor is much less than at higher pressures and all control is performed in pneumatics. A large governor station might feed two thousand consumers. The cost of electronic control at this level is critical, however the controller must still be very reliable which requires fault tolerant techniques to be used. It is this type of "small digital controller" that has been studied in depth at Durham and a prototype controller developed. The controller developed is more powerful than that required for simple pressure/flow control of a single regulator and could be expanded to control line-pack as described above.

As well as experimenting with hybrid electronic/mechanical governors, British Gas are making increasing use of microprocessor equipment. Because of the requirement for high reliability when controlling gas, it is essential to consider the reliability of every piece of equipment and introduce fault tolerant techniques if necessary.

1.2 PNEUMATIC GOVERNORS

Since the aim of this research was to develop a "small digital controller" to be used as part of an electromechanical governor, it is necessary to study the characteristics of pneumatic governors so that the replacement is no worse than its predecessor in performance and cost.

The terms regulator and governor are virtually synonymous, but the term governor will be used to describe a system having an inlet and a controlled outlet. The term regulator will be used to describe the individual regulator valves which comprise the system.

Pneumatic regulators have been used by British Gas for many years. They have many advantages, of which the major ones are listed below :

(i) Tried and trusted - For a mechanical device they are very reliable. This is partly due to their mature technology and the vast quantity produced.

(ii) Inexpensive - A typical regulator costs a few hundred pounds.

(iii) They are self powered and need no mains supply.

(iv) They can withstand some unusual operating conditions and transient surges.

(v) Mechanically they are very rugged and will operate over a wide temperature range as long as they are prevented from freezing up.

A typical regulator valve is shown in figure(1). The valve which controls the flow of gas is mounted within the lower cast iron body. The valve orifice is made from stainless steel to minimise wear and corrosion and the valve seat is made from nitrile rubber to ensure low leakage when shut. The valve is attached to a long stem which passes up into the diaphragm chamber and is attached to the centre of the diaphragm. Movement of the diaphragm causes the valve to open and shut. The diaphragm equilibrium position is reached by balancing the force of the spring on the top of the diaphragm with the gas pressure below. Downstream gas pressure is fed to the pressure tapping in the bottom diaphragm bowl which is shown to the right of figure(1). If the downstream gas pressure rises, this causes the diaphragm to move up which closes the valve and reduces the downstream pressure. The regulator is thus seen to behave as a negative feedback closed loop controller of pressure. The outlet pressure of the regulator is set by the main spring whose compression can be varied by adjusting the top screwed plug. The valve shown is an "open at rest" type and rupturing of the diaphragm will cause the valve to fail fully open.

1.3 LIMITATIONS OF PNEUMATIC GOVERNORS

A schematic diagram of a governor feeding a distribution network is shown in figure(2). There will be some point in the network where the largest pressure drop exists between that point and the governor. This point is labelled the worst case pressure point. To complicate matters further, this point moves around as the loading on the network changes. There is a statutory minimum pressure (about 5 inches wg) at which consumers must be supplied, which means that the governor outlet pressure must always be high enough to give this minimum pressure at the worst case pressure point. All other consumers will receive gas at pressures above the statutory minimum.

It is desirable to keep the outlet pressure of the governor as low as possible whilst still satisfying the above conditions because of the problem of leakage. Old distribution pipework is subject to relatively low levels of non-hazardous leakage which is proportional to the network over-pressure. Under non worst case conditions and periods of low demand, there will be considerable over-pressure unless the governor is readjusted. It has been estimated by Murphy[23] that considerable savings in the value of lost gas could be made for a 2"wg reduction in nationwide system pressure.

Using microprocessor techniques, it should be possible to reduce network over-pressure by at least 2"wg.

As well as resistive pressure drops in the network there is a variable pressure drop across a simple pneumatic regulator. The outlet pressure of a typical regulator drops by 15 per cent as the flow through it is varied from zero to full rated - this is called "droop". Consequently the regulator set point must be set higher than required to compensate for droop at high flows. A microprocessor controlled valve could reduce the droop to zero.

It is necessary to set the governor outlet pressure higher than that required in a "twin stream" governor. In large governor stations where it is imperative to supply gas without failure, it is common to use a twin stream regulator configuration as shown in figure(3). Slam shuts are similar to regulator valves, but are designed to close rapidly, cutting off the gas flow, when their set pressure is reached. Typical pressure settings are shown in figure(3). The minimum acceptable pressure at the governor outlet is 12"wg, so the standby regulator is set to give this. Normally the standby regulator is held off and the outlet pressure is controlled at 14"wg by the active stream regulator. If the outlet pressure rises above 18"wg, the active stream is cut off by the slam shut set at 18"wg and the standby stream takes over. If the pressure rises further to 20"wg, the standby stream is cut off by the slam shut set to 20"wg and gas flow through the governor is cut off. It is difficult to set accurately both regulators and slam shuts, so it is necessary to have their settings several inches wg apart so that they do not inter act with each other. If all the valves and slam shuts were controlled by a microprocessor, then it would be possible to have their set points closer together.

To compensate for resistive pressure drops in the system, it is desirable to reduce the governor pressure under conditions of low flow and to increase the pressure under conditions of high flow. Such control is called "demand activated governing" (DAG). A pneumatic governor has been developed to perform this control function, but is very complex and requires many regulator valves and pneumatic components. If a multi-feed network is fed by several DAG controllers, then the controllers may interract and make the system unstable. To prevent this it is necessary for the controllers to communicate with each other so that interaction is controlled.

The previous discussion has shown that pneumatic regulators are satisfactory in their single form, but as soon as several are connected together to form a governor, penalties must be paid by way of complexity.

increased network leakage and inefficient control of the network. Most of the disadvantages of pneumatic governors can be overcome by using a pneumatic/electronic hybrid governor and such tasks as telemetry, health monitoring and complex control algorithms are made much easier to implement. It is however essential that the electronic controller is extremely reliable, especially in circumstances where the controller is controlling several valves. It is necessary to use fault tolerant techniques to achieve such a high level of reliability.

CHAPTER 2

A REVIEW OF OTHER FAULT-TOLERANT CONTROLLERS

2.1 SMALL CONTROLLERS

The application of redundancy to small controllers is often more cost sensitive than large computers. Redundancy can be used to achieve non stop processing, fail safe operation, and a reduction in the mean time to repair (MTTR) by the use of built in diagnostics.

2.1.1 TMR controllers

Platteter[7] describes a TMR design using three 8085 microprocessors which is similar to the experimental controller described in chapter eight. Platteter recommends the use of TMR for the protection against untestable errors in microprocessors. Three different manufacturers' 8085s are used, each having a different independent design. He proposes that it is better to accept that complex VLSI circuits such as microprocessors will contain untestable faults, and it is better to protect against this using TMR techniques, than to eliminate all manufacturing faults. The voting is performed at bus level in STTL integrated circuits which will be less reliable than voting in FPLAs. A very unsatisfactory approach is taken towards resynchronisation. Either resynchronisation "just happens" or it is aided by PUSHing all registers onto the stack and then POPing them off. He reports that sometimes the system would not resynchronise and it was necessary to perform a reset. The reasons for the system not synchronising and how this can be overcome is described in chapter eight.

Higuchi et al [22] describe a TMR system using three 8085 microprocessors which is resynchronised at regular intervals. This is unecessary since it is better only to resynchronise when a voting error is detected. Resynchronisation is performed by PUSHing all registers onto the stack and then POPing them off. This will not always resynchronise the program counter and sometimes one processor will be found to lock one clock cycle behind the other two as described in chapter eight. These difficulties are not reported. An alternative TMR configuration is proposed by Higuchi which uses software voting and the three processors execute tasks at staggered intervals and compare their results when all processors have executed the task. The software will be more complex and voting will not be transparent to the user, but the execution of tasks at different times should reduce the effect of transient errors.

Ryland [20] describes the use of microprocessors in reliable railway signalling equipment. The interlocking system uses three 6800 microprocessors in a TMR configuration. The processors are loosely synchronised and a processor will attempt to shut down any processor with which it disagrees (assassination). If the assassination attempt fails, it will shut itself down (suicide). Fault reporting and the repair of a faulty channel is performed on-line. Two separate data links are provided to the equipment and data is transmitted at 10k baud in Manchester code with Hamming code protection. The software was written in assembler because the designer felt that it was easier to achieve reliability because no arithmetic was used and the hardware interface was simple.

Davies et al [21] discuss ring communication structures for a small system. Three 8748 single chip microprocessors are connected in a TMR ring structure and the synchronisation is performed in software and is accomplished by handshaking between the processors or the insertion of a fixed delay before voting. The outputs from the three microprocessors must be combined which is nearly as complex as voting in hardware, but they propose that software voting offers greater versatility and is not prone to component failure.

2.1.2 Fail safe controller

A life support system controller is described by Lim [4]. The design is deliberately not fault-tolerant because of the increased cost, but is designed to detect hardware and software faults and to cause the system to fail safe. An alarm is given on failure of the controller so that human operation will ensure continued operation of the life support system.

2.1.3 Built in test equipment

Foose [17] describes an industrial microprocessor controller which was designed to reduce the MTTR. The design aim was 80% automatic testability for 10% extra cost. The controller is divided into separate modules, each performing a single function, and modules are designed to test themselves. By reducing the MTTR the availability of the controller is increased and maintenance costs reduced. Failure diagnosis can also be performed by unskilled operators.

2.2 LARGE COMPUTERS

The majority of research so far has been concentrated on large fault-tolerant computers which are used in the space, avionics, and nuclear industries.

2.2.1 Space shuttle and avionics

The space shuttle is probably one of the best (and most expensive) examples of a fault-tolerant system. References[12,13] describe the digital processing subsystem. NASA decided to use standard proven avionics computers which nowadays are rather outdated and to implement the redundancy in software. Five identical IBM avionics 32 bit computers are used each having 250k bytes of memory. The computers share two 16M byte tape drives for mass storage. The computer contains built in test equipment and can detect 98% of errors. The computers are interconnected to themselves and to the sensors and actuators by serial buses. The highest reliability configuration consists of four computers in loose synchronism with the fifth computer executing background tasks. Synchronism is achieved by hardware and software and two computers out of the four can fail without a catastrophic system failure. The inertial guidance platform is triplicated and connected to different buses. The inertial information undergoes selection filtering which compares the information channels and rejects any that are above a predetermined trip level. With all channels functional, mid-value selection is used which is common for inertial systems and failure of channels results in a degraded selection algorithm. Failure of a channel must not cause a large transient disturbance at the output of the selection filter, since a fault lasting only one second is enough under certain conditions to crash the shuttle.

The cycle time of the control process is 40ms in which time selection filtering and control of all the aerodynamic surfaces and rockets is performed. The cycle time of 40ms is necessary for the stability of the shuttle.

Ahern et al [14] describe a software voter/monitor for selection filtering of inertial guidance information. It is necessary to reject faulty channels and then to perform the vote. Mid-value selection is again chosen as the best algorithm and the requirement to smoothe the switch-over from one selection algorithm to another is stressed to avoid transient disturbances.

Modern military aircraft are controlled by the fly-by-wire technique. A digital flight computer reads in inertial and pilot information and controls electromechanical actuators which are connected to the control surfaces. Mechanical and hydraulic linkages are replaced, but it is necessary to use redundancy techniques to achieve high reliability in the controller. An added advantage in military aircraft is the increased survivability of the plane if it is damaged. For obvious reasons nothing has been published about modern use of microprocessors in digital flight controllers, but it is known that several different types of microprocessor including the Texas 9900 and Motorola 6800 are used in redundant configurations in the Tornado. Deets et al [15], in discussing the design and flight experience of a fly-by-wire control system, describe the first aircraft to use the fly-by-wire technique. An Apollo space computer was installed in a modified F8 fighter. The 16 bit computer had a 36k word memory, 12μs instruction cycle and performed the control algorithm in 30ms. The initial tests were successful and were conducted with a parallel back up hydraulic system.

Black et al [5] discuss the development of a spaceborne memory, which uses a memory error correction code, having the same number of bits as the SEC/DED Hamming code, but which is able to correct double bit errors (DBE). The position of stuck single bit errors is logged so that if a double bit error occurs, the stuck single bit error can be "erased" since its position is known, and a DBE tolerated. In order to achieve the high reliability required in space applications using semiconductor random access memory (RAM) the design was required to tolerate douole bit errors.

2.2.2 Commercial computers

Commercial computers are now being built with redundant circuitry, especially for memory protection. The aim is to increase the mean time to failure (MTTF) by redundant circuitry and to reduce the MTTR by built in test equipment. Hence the availability can be increased and maintenance costs reduced. Toschi et al[6],in discussing a fault-tolerant computer memory, describe the advantages gained by adding single bit error correction to memory and the tolerance of transient errors. Troublesome and failed memory devices are masked by redundancy and are replaced at periodic maintenance intervals. Swarz [18] describes the design methodology behind the VAX computer with special emphasis on the ability of the computer to tolerate memory, disk, and other errors. The computer is designed to minimise the MTTR by the use of a LSI 11 microprocessor dedicated to diagnosing faults in the VAX computer. In this way the availability is improved and the maintenance costs reduced.

2.2.3 Telephone exchange computers

Modern PCM telephone exchange computers must have a very high availability, typically less than two hours down-time in forty years. Fantini et al [2] describe an exchange computer based on the Z8000 16 bit microprocessor which has a fault coverage of 0.98. A duplicate processor is used to detect errors by comparison with the main processor. The power supply, clock, and bus are continuously monitored for errors and RAM and ROM checking is performed off-line. The aim of the design is to detect and isolate faults. Most faults were found to be transient, but the system relies on the prompt repair of permanent faults to achieve a high availability.

Ceru et al [10] describe a similar exchange computer. The 16 bit processor is constructed from discrete TTL and LSTTL and consists of two processors operating in synchronism. The detection of an error causes both processors to check themselves and each other, and the first one to finish the checking resumes control of the exchange. Faults are logged and the second processor is resynchronised if possible.

2.2.4 Safety monitoring computer

Harbert [3] describes a system using three 8086 16 bit microprocessors in a fire/gas detection and automatic shut-down controller. The input boards, microprocessor, memory, and output boards are triplicated and the processors operate asynchronously. Magnetic bubble storage is used since this gives improved reliability over spinning memory devices. Several hundred detectors are monitored by the system and their status displayed on colour VDUs, which give a clear display of information. The system controls emergency shut-down and extinguishing equipment.

2.2.5 PDP 11 computer

Canepa et al [19], in discussing the architecture of multiprocessing systems, describe a system consisting of three LSI 11 computers connected in a versatile triplex configuration. Hardware modification is minimal except for the construction of the voters which use 250 SSI integrated circuits. Voting is performed at bus level which makes the redundancy transparent to the software, allowing standard software to be executed. The processors can be configured to operate singly, with one processor talking to the other two, or in a TMR configuration where the processors run in synchronism. The system has been built to examine the effect of transient faults on computers. By monitoring the three processor system and injecting faults into one channel, they hope to gain information about the frequency, duration and location of transient faults in a computer.

2.2.6 Railway applications

Forsythe et al [16], in discussing reliable train control applications, describe a system using three INS8900 16 bit microprocessors which share a common bus. Each processor has its own memory store and both processors and memory are buffered to improve the fault isolation. The three processors share common RAM and EPROM and voting is performed in software to reduce the hardware costs. The operation of the controller is divided into four sectors. In the first three sectors, each processor executes a task and then swaps tasks with another processor at the end of the sector. In this way the three tasks are executed three times on three different microprocessors. In the fourth sector the results of the computations are compared and recovery is executed if required. Input/output is performed in the fourth sector only if the error checking is satisfactory. In spite of the high level of redundancy, the system was found to lock-up as a result of certain interference tests and the watchdog timer was found to reset the system and restore correct operation.

2.2.7 Fault-tolerant software

Much of the fault-tolerance on large computers is implemented in software. References[8,9,11] describe the implementation of "recovery blocks" on a large machine in a high level language. A recovery block is defined as a section of code in which recovery is possible. Calculations undergo a series of acceptance tests. If the acceptance test is not passed an alternative calculation is tried. Before executing an acceptance test, variables are stored in a "recovery cache" so that the variables can be restored to their initial state if the alternative calculation and acceptance test fails. Ghani et al [11] describe the "recovery cache" hardware as implemented on a PDP 11 computer.

2.3 SUMMARY

Many of the fault-tolerant features reported in small controllers are used in the experimental controller. The designs of Platteter [7] and Higuchi et al [22] are similar to the governor controller which overcomes many of the shortcomings in their designs, such as the failure to resynchronise. Ryland [20] proposes a different TMR structure and reports that software was written in assembler. The governor controller software was likewise written in assembler. Davies et al [21] discuss an alternative TMR structure where the voting is performed in software. This structure is very suitable for single chip microprocessors, but was not used in the governor controller for the reasons given in chapters six and seven. The controller described by Lim [4] is designed to fail safe as is the governor controller. Foose [17] reports that built in testability can be incorporated at little extra cost and Ryland [20] mentions the on-line reporting of faults. The governor controller performs on-line fault reporting which allows the availability to be increased and maintenance costs reduced.

Although large computers are outside the definition of small controllers, a discussion of fault-tolerant features is included for the sake of completeness and many of their fault-tolerant features can be adapted for use on small microprocessor controllers. Much of their fault-tolerance is implemented in software and is more complicated than that required for a small controller. The space shuttle uses serial buses for the interconnection of units since this is more reliable than using large parallel buses. The space shuttle and fly-by-wire aircraft use redundant inertial sensors and software is used to select the best inertial information. Mid-value selection is commonly used as it is a fast algorithm to implement. The governor controller, in a similar fashion, uses redundant pressure trasnsducers and selects the best pressure information using software. Time is not critical, so a more suitable averaging algorithm is used, rather than the faster mid-value selection.

References [5,6;18] describe the use of single bit error correction in semiconductor memory. Black et al [5] discuss a code which will correct double bit errors, but which uses the same number of bits as the more normal Hamming code. The experimental controller cannot correct double bit errors, but allows recovery from them, as well as transparently correcting single bit errors.

References [2,10] describe similar telephone exchange computers consisting of a main processor and a standby spare. This architecture could have been used in the governor controller, but a TMR structure was used in preference. Fantini et al [2] report that most faults experienced were of a transient nature. References [3,16] describe 16 bit TMR controllers where the voting is performed in software. The governor controller performs the voting in hardware which is transparent to the software. Canepa et al [19] describe a TMR system , using three PDP11 computers, where the voting is likewise performed in hardware. The complexity of hardware voting in large computers is highlighted by the size of the required voting circuitry.

Recovery blocks, as described in references [8,9,11], can usefully be implemented on small controllers. The governor controller uses a special type of recovery block, more properly called a recovery vector, which allows vectored recovery to be executed following the detection of a hardware or software error.

CHAPTER 3

FAILURE OF COMPONENTS

3.1 FAILURE DISTRIBUTION AND MECHANISMS

The failure rate of most types of electrical devices follows the classical "bath-tub" curve of figure(4). Phase one, infant mortality, represents the early life failures of a device and is usually associated with one or more manufacturing defects. After several hundred hours, the failure rate approaches some constant low value, phase two, where it remains for anything from several years to several hundred years and failures occur randomly. Wearout failures, phase three, occur at the end of the useful life of a device and are characterised by a rapidly rising failure rate with time as the device wears out both physically and electrically. A common wearout mechanism in integrated circuits is corrosion due to moisture trapped inside the device package. Under normal operating conditions, failure due to wearout is rarely experienced with integrated circuits, unless they are operated for a very long period of time - references[34,27].

The constant failure rate region, phase two, represents failures due to random events such as electrical surges. Most failure rate data sources assume a constant failure rate and present their results in the form of n failures per unit time, typically n failures/10^6 hours. This is satisfactory as long as the wearout region is not encountered during the lifetime of the equipment. This assumption is likely for integrated circuits and resistors, but components such as electrolytic capacitors may be different. The wearout phase of an electrolytic capacitor results from the electrolyte drying up, which might occur after only a few years. However the quoted "constant" failure rate might suggest a MTTF of several hundred years. Thus it is essential to distinguish between the useful life of a component and MTTF if wearout is encountered.

An example of the misleading result that this confusion might produce can be given by considering the human life span. The mortality rate for humans approximately follows the "bath-tub" curve, but if only the constant "failure rate" experienced during youth and middle-age is considered, then a MTTF of two thousand years is predicted as opposed to a normal lifespan of about seventy five years.

The majority of components used in "small digital controllers" are integrated circuits, resistors, and small decoupling capacitors. It is therefore valid under favourable operating conditions to consider the failure rate of such components constant and to assume that their useful life is equivalent to the MTTF.

Failure mechanisms in MOS integrated circuits are given in table(1) and are discussed in more detail in references[27,35]. Most potential failures caused by these mechanisms can be detected early by suitable screening. Consideration of these mechanisms is important since the majority of microprocessor and memory components are fabricated using the MOS technology.

3.2 FAILURE RATE MEASUREMENT AND ACCELERATION

The simplest method of calculating the failure rate of devices under controlled or field testing is by the equation :

$$\text{failure rate} = \frac{\text{number of failures}}{\text{no. devices X no. hours tested}}$$

The problem with this equation is that statistically it only represents a point estimate at 50% confidence, and that if testing reveals no failures, then obviously the failure rate is not zero. A statistical solution to this problem is given by the Chi-squared distribution as discussed in references[32,49].

$$\text{failure rate} = \frac{\chi^2(1 - C.L, 2r + 2)}{2nt} \qquad (3.2.1)$$

where: χ^2= Chi-square function

CL= confidence level expressed as a decimal

r = number of failures

n = number of parts tested

t = total test duration

Tables of χ^2 are found in many texts on statistics[49].

Equation(3.2.1) is almost universally used, with most failure rates quoted at the 60% confidence level.

3.2.1. Accelerated testing

Even "unreliable" microelectronic devices may last several thousand hours before failing which makes life-testing a very long process and screening virtually impossible. It has been shown that the failure rate of microelectronic devices exponentially increases with temperature according to the Arrhenius reaction rate equation, references[35,37,38].

$$\text{failure rate} = C \exp(-Ea/KT) \qquad (3.2.2)$$

where: Ea = activation energy in eV

K = Boltzmanns constant (8.63×10^{-5})

T = absolute temperature

C = an appropriate constant

When conducting accelerated tests and analysing test data it is important to remember two things :

(i) The failure rate is exponentially dependent on temperature so that incorrect specification of the device junction temperature will have a large effect on the failure rate.

(ii) The correct activation energy should be chosen, appropriate to the failure mechanism under consideration.

According to the Arrhenius equation it is possible to accelerate significantly failures by testing at elevated temperatures and the acceleration factor is calculated by :

$$Fa = \exp\left[\frac{E_A}{K}\left(\frac{1}{T_2} - \frac{1}{T_1}\right)\right] \qquad (3.2.3)$$

where : Fa = acceleration factor

T1 = test temperature of the junction

T2 = desired temperature of the junction

The graph of figure(7) shows the relationship between Fa and the test temperature, T1, for a desired temperature, T2, of 25° C. A family of curves is plotted, showing the sensitivity of acceleration factor to activation energy. Activation energies chosen are those corresponding to the major failure mechanisms of table(1).

When considering failures due to moisture ingression into the microcircuit package and subsequent corrosion of the metalisation and bonding wires, a similar acceleration factor may be used. It is common to test under conditions of 85° C/85%rh, references[39,43]. The acceleration factor may be calculated according to the Lawson-Harrison law, references [35,44] and is given by :

$$Fa = \exp\left[\frac{E_A}{K}\left(\frac{1}{T_2} - \frac{1}{T_1}\right) + b(H_1^2 - H_2^2)\right] \qquad (3.2.4)$$

where : Fa = acceleration factor

Ea = activation energy

K = Boltzmanns constant

T1 = desired junction temperature

T2 = test junction temperature

H2 = test humidity

H1 = desired humidity

b = a constant

Reynolds[35] uses values of Ea = 0.6eV as per table(1) and b=4.4 .

3.3 FAILURE RATE PREDICTION

Failure rates may be predicted either by accelerated testing or a combination of accelerated testing, field testing, and mathematical modelling. The most widely used document for failure rate prediction is probably MIL-217 [37], prepared by the American Department of Defense at the Rome Air Development Centre (RADC). This document is regularly updated and has been published as MIL-217 A,B,C and recently D. The first to predict the failure rate of microelectronic devices was MIL-217B. Components covered are those mainly used in defence applications, but this covers resistors, capacitors, most integrated circuits, connectors, switches etc. and field data is mainly gathered from defence applications. A mathematical failure rate is developed for each type of component of the form :

$$\lambda = \lambda_b \Pi_{\varphi} \Pi_E \Pi_A \text{ --------} \qquad (3.3.1)$$

where : λ = failure rate

λ_b = base failure rate

Π_{φ} = quality factor

Π_E = environmental factor

Π_A = application factor – voltage stress, power rating etc.

The factors λ_b . Π_φ . Π_E . Π_A etc. are tabulated in MIL-217 covering many operating conditions and grades of component.

The failure rate of components depends on their quality and the MIL-217 series attempts to allocate Π_φ or quality factors according to the grade of manufacture and subsequent screening.

An environment factor, Π_E , is applied to take account of the operating environment of the component. Experience has shown that failure rates depend on the operating environment, as might be expected. For instance a missile launch is more hostile than an aircraft in flight which in turn is more hostile than a ground based environment. For the purpose of evaluation of industrial equipment, the environment chosen as being appropriate is "ground fixed", Gf.

Finally the application factor, Π_A , is taken into account. This factor takes many forms, but is mainly used to reflect the electrical stress under which the component is operating. Any derating of the voltage, current, or power handled by the component will result in an improved value of Π_A .

The failure rate model for MOS and bipolar devices is of particular interest since this covers TTL logic and most microprocessor and memory devices excluding ROMs. The model used is :

$$\lambda = \Pi_\varphi \Pi_L [C_1 \Pi_T \Pi_V + (C_2 + C_3) \Pi_E] \qquad \text{failures/}10^6 \text{ hours} \qquad (3.3.2)$$

where : λ = device failure rate

Π_φ = quality factor – depends on grade and screening level

Π_L = device learning factor – unity for a mature device

Π_T = temperature acceleration factor – Arrhenius relation

Π_V = voltage stress factor – unity except for CMOS

C1 = device complexity factor – depends on transistor count

C2 = device complexity factor – depends on transistor count

C3 = package complexity factor

Π_E = environment factor

Since Π_T is exponentially dependent on temperature, then for high temperatures, Π_T is large and the model can be simplified to the approximation :

$$\lambda = \Pi_\varphi \Pi_T C_1 \qquad (3.3.3)$$

The failure rate is therefore exponentially dependent on the junction temperature of the device.

3.3.1 CNET

In 1972 the Comité de Coordination des Télécommunications in France decided to establish a group of people to evaluate and predict the reliability of components used in the telecommunications and computing industries. The first version of their report was published in 1976, reference[38]. This report was based on MIL-217B (an earlier version of MIL-217D [37], but was biased towards telecommunication and computer equipment operating in favourable environments as opposed to defence equipment operating in hostile environments. Failure rate models are given which, are similar to MIL-217. The model given for microcircuits was considerably different, but was updated in 1982 to the following model which is more similar to the MIL-217 model.

$$\lambda = \Pi_\varphi \Pi_L \left[C_1 \Pi_T \Pi_t \Pi_V + C_2 \Pi_B \Pi_E \Pi_S \right] \quad \text{failures}/10^9 \text{ hours} \qquad (3.3.4)$$

where : λ = device failure rate

Π_Q = quality factor - depends on grade and screening level

Π_L = reliability growth factor - unity for a mature device

Π_t = temperature acceleration factor - Arrhenius relation

Π_T = device technology factor

Π_V = voltage stress factor - unity except for CMOS

C1 = device complexity factor - depends on transistor count

C2 = device complexity factor - depends on transistor count

Π_E = environment factor

Π_B = humidity / temperature factor

Π_S = transportation factor - depends on no. of journeys

3.3.2 National Centre of Systems Reliability

The NCSR reliability data [34] is a condensed version of their computer data bank held by the Systems Reliability Service, SRS. The data bank comprises two main parts. The first contains field data gathered from the nuclear industry where the environmental conditions are well known and controlled. The second part contains information from MIL-217C as well as data from other published sources, laboratory tests and theoretical predictions. Martin Marietta Aerospace provide much of the support for the RADC data which in turn influences the MIL-217 series. The SRS data bank contains this data, so the SRS data will not be an independent source to MIL-217.

The failure rate models are similar to MIL-217 with the exception of the microcircuit model, which is :

$$F = K1\ Kg\ (\ Fe1 + Fe2 + Ft\ )\ \text{failures}/\ 10^{6}\ \text{hours} \qquad (3.3.5)$$

where : F = failure rate for a hermetic device

K1 = unity for mature devices otherwise ten

Kg = reliability growth factor

Fe1 = transistor count complexity / environment factor

Fe2 = packaging factor - depends on no. of pins and environment

Ft = temperature accelerature factor dependent on packaging

The model only considers two grades of device, hermetic and non-hermetic. The non-hermetic device failure rate is equal to twice the hermetic failure rate as well as further adjustments incorporated into Ft.

3.3.3 RADC field data

As well as publishing the MIL-217 series, the RADC publishes failure rate data obtained from field experience. Klein [45] presents much field data on microcircuits which is shown to agree approximately with MIL-217C.

3.3.4 European Space Agency

The ESA has its own data bank and requires companies wishing to tender for projects to perform a reliability prediction, using the failure rate data generated and supplied by themselves. This has the great advantage that all companies are forced to use the same failure rate data and a valid comparison between proposed designs can be made.

3.3.5 Manufacturers testing

Most microcircuit manufacturers publish the results of accelerated testing on their devices, references[27,40]. Devices are subjected to thermal and physical shock tests as well as dynamic testing at elevated temperatures. A large number of devices are tested for typically several thousand hours, and the number of failures observed are fitted to a Chi-squared distribution. The failure rate is typically quoted at the 60% confidence level and the failure rates are further modified according to the Arrhenius acceleration factor, to give a failure rate appropriate to the likely conditions of usage. There is some discrepancy between different manufacturer's predictions, since they make use of different activation energies as applied to the Arrhenius acceleration equation. Tests are also performed to verify the suitability of device packages as regards shock and humidity as reported by Motorola [40].

3.3.6 Simple models

A simple model for microcircuits is quoted by the RRE which is useful when very few parameters are known :

$$\lambda = \lambda_B \, Kd \, Kl \qquad (3.3.6)$$

where : $\lambda_B = 5$ digital bipolar

$\lambda_B = 8$ digital MOS

$\lambda_B = 12$ linear bipolar

$$Kd = \frac{\text{die area in square inches}}{0.015}$$

$$Kl = 1 + \frac{N - 12}{24}$$

N = number of package leads

For this model to apply, the ambient temperature must not exceed 55°C, the junction temperature must not be greater than 40° C above ambient, and plastic encapsulation must not be used. If plastic encapsulation is used, then it is recommended that the failure rate is doubled.

3.3.7 GIDEP computer data base

The Government Industry Data Exchange Programme is an American based association which was established in 1959 and provides access to four data banks, which are :

(i) Engineering data bank

(ii) Reliability-maintainability data bank

(iii) Failure experience data bank

(iv) Metrology data bank

The reliability-maintainability data bank is of particular interest as regards reliability prediction.

3.3.8 EuReData

The European Reliability Data Bank Association was established in 1974 on a voluntary basis and was formally constituted in 1979. The organisation is non profit making and is supported by the EEC. Its main aims are to promote data exchange between organisations and to set up standard methods for obtaining and using reliability data.

3.3.9 MIL-217 Data base

The failure rate models and factors contained in the MIL-217 series ideally lend themselves to computerisation, since mathematical formulae are given for all the failure rate factors, Π_T, Π_P etc. The Predictor package [81] includes the computerisation of MIL-217. The CNET data, likewise, is suitable, however data such as NCSR [34] is not suitable since no formulae are given for the failure rate factors, but only tables of figures.

3.4 COMPARISON OF FAILURE RATE DATA

The failure rates for typical components used in a digital controller are given in tables(2) to (16). There are two main sources of data :

(i) Field and accelerated life testing data.

(ii) Predicted failure rates based on models such as MIL 217.

3.4.1 Resistors

The failure rates of oxide film resistors are compared in table(2). Of the predicted data, the CNET prediction is identical to MIL 217 which might be expected since the CNET data is based on MIL 217B. The NCSR prediction is double that of the other predictions, but fits in between the two failure rates based on field experience. In addition the NCSR data is based on field experience within the nuclear industry, so it is proposed that the NCSR failure rate is the value most likely to be correct. There is however good agreement between all values with the worst and best failure rates only differing by a factor of four.

3.4.2 Capacitors

The failure rates of a 0.1uF decoupling capacitor are given in table (3). This time agreement is not good with the worst and best values differing by a factor of twenty. This difference may be partly due to the wide variation in capacitor types and it is not possible to make predictions for identical capacitor types. However the CNET and MIL 217 values are in close agreement together with the ICL field data, as was the case for resistors, so it is proposed that the MIL 217 failure rate is accepted for this type of capacitor.

3.4.3 Soldered joints

The failure rates of soldered joints are compared in table(4). Ignoring the CNET prediction which seems to be low, there is good agreement between the other values and again the ICL field data agrees very closely with MIL 217. It is proposed that the MIL 217 value is accepted.

3.4.4 Wire-wrap connections

The failure rates of wire-wrapped joints are compared in table(5). The MIL 217 and CNET predictions seem grossly optimistic when compared with the field data and soldered joints. In the case of the MIL 217 data it is suspected that very few wire-wrap joints are used in military equipment, but many more are used in the computer industry. For this reason and because the ICL data and Dummer agree so closely, it is proposed that the field data is more likely to be correct.

3.4.5 Edge connectors

The failure rate of edge connectors is compared in table(6). It is difficult to compare failure rates because of the wide variety of edge connectors and the mating/unmating cycles are not known for the field data. The Dummer failure rate seems high whilst the MIL 217 rate seems low when compared with field data as well as soldered and wire-wrapped joints. Although the CNET data is based on MIL 217, the CNET failure rate for connectors is much higher (a factor of 30). Presumably CNET found the MIL 217 model to be too optimistic. The CNET prediction agrees with the ICL data and seems to be a sensible value, so it is proposed that the CNET data is used in preference to the other sources.

3.4.6 Integrated circuits

The failure rates of integrated circuits are several orders of magnitude greater than resistors, capacitors, or connections, and therefore because of the large number of integrated circuits used in a digital controller, their failure rate dominates the total controller failure rate. The failure rate of integrated circuits has been shown to fit the Arrhenius relationship as discussed in section 3.2, which means that the failure rate is exponentially dependent on junction temperature and activation energy. It is therefore essential to make comparisons between different reliability data sources at equivalent activation energies and junction temperatures, otherwise the exponential dependence will introduce large differences between the failure rates. The activation energies used by MIL 217D, MIL 217C and NCSR appropriate to different device technologies are given in table(7). The higher the activation energy, the higher the failure rate at an elevated temperature.

The main difference between MIL 217C and MIL 217D is the section on microcircuits. Only two activation energies used in the calculation of π_T are used in MIL 217C and no distinction is made between hermetic and non-hermetic (plastic) encapsulation. The revisions incorporated in MIL 217D seem to improve the predictions. Nine different activation energies are used according to device technology and a distinction is made between hermetic and non-hermetic encapsulation with a higher activation energy quoted for non-hermetic devices. This seems reasonable since non-hermetic devices are more prone to corrosion due to moisture, and this failure mechanism is shown to have a high activation energy as shown in table(1). The NCSR data only uses three activation energies which agree closely with MIL 217D for hermetic devices. The NCSR activation energies make no distinction between hermetic and non-hermetic devices and it is simply recommended to multiply the failure rate by two for non-hermetic devices. This approach is felt to be too simple.

The CNET failure rate model uses two activation energies of 0.3 and 1.0eV in the calculation of πt , the temperature acceleration factor. The weighting between 0.3 and 1.0eV is varied according to device technology. This seems to be a better approach than MIL 217D and NCSR which take the rather simplistic view that only one activation energy is present when many activation energies may all be making contributions through different failure mechanisms.

Some manufacturers accelerated testing makes use of two activation energies, references[28,29], although one activation energy is seen to dominate.

The junction temperature is calculated according to :

$$Tj = Tamb + \theta ja.Pdiss \qquad (3.4.1)$$

where: Tj = junction temperature

Tamb = ambient case temperature

θja = junction/ambient thermal resistance

Pdiss= power dissipated

If Pdiss is small as in small CMOS or LSTTL devices, the temperature rise due to θja.Pdiss will be small and the failure rate will be relatively insensitive to these two parameters. However in the case of a microprocessor which dissipates about 0.5W, and can physically be felt to run warm, it is important to determine accurately θja.Pdiss. Motorola[42] give values of θja from 70-115°C/W for a plastic device and 50° C/W for a ceramic device. Considering MIL 217D and the 6800 microprocessor, θja is correctly given as 50°C/W for a hermetic device, but the worst case power dissipation of 1W is given. It would seem more reasonable to use the typical power dissipation of 0.5W which reduces the junction temperature by 25°C, giving a large increase in predicted reliability.

If θja cannot be determined from manufacturers data or the table in MIL 217D, then MIL 217D recommends that the following values are used :

Package type	θja °C/W
< 22 pin hermetic	30
< 22 pin non-hermetic	125
> 22 pin hermetic	25
> 22 pin non-hermetic	100

The NCSR data recommends the following values for θja :

θja = 60°C/W hermetic devices

θja = 155°C/W non-hermetic devices

Compared with the recommendations of MIL 217D and Motorola, the NCSR figures seem excessively high.

Although the power dissipation, Pdiss, is given in MIL 217D in the same table as Θja, it is recommended that the manufacturers data is consulted instead since (for example) the power dissipation of an 8080 microprocessor is incorrectly given as 1.7W and not 1.5W as given by Intel [33]. It is further recommended that a more realistic failure rate is obtained by using the typical power dissipation, as quoted in the manufacturers data and not the maximum power dissipation. If possible the device case temperature should be measured to give an accurate value of case ambient.

3.4.7 TTL integrated circuits

The failure rates of TTL integrated circuits are compared in table (8). The junction temperature of the ICL devices is unknown, but all other predictions are based on a junction temperature of 33° C. The four predicted failure rates use very similar activation energies, Ea. The MIL 217C prediction is obviously too high by at least an order of magnitude, but the other failure rates are seen to vary from worst to best by a factor of six. The predicted failure rates are pessimistic when compared with the ICL field data, although this difference may be due to lower junction temperatures of the ICL devices. If the CNET, MIL 217D and NCSR predictions are compared, they are seen to vary by a factor of three which is thought to be very good. Since the CNET prediction falls between the other two predictions, it is proposed that the CNET data is accepted in preference to the other two for TTL failure rate prediction.

3.4.8 6800 Microprocessor

The failure rates for a 6800 microprocessor are compared in table(9). The activation energies and junction temperatures used to calculate the failure rates are those recommended by the respective reliability data sources and are seen to vary considerably. All the predictions use a junction temperature which is at least 15°C too high. Again MIL 217C is seen to be too high by at least two orders of magnitude, although this is partly due to using a higher activation energy than the other predictions.

Table(10) shows the failure rates of table(9) converted to a common base of junction temperature and activation energy. An activation energy of 1.0eV is chosen, since this is the value used by Motorola as well as the MIL-STD 883 screening procedure. MIL 217C is ignored since this is much higher than the other values, however both MIL 217D and NCSR predictions are seen to agree closely and give a sensible value of about one failure per twenty years. The Motorola failure rate seems to be too low and corresponds to one failure per thousand years.

3.4.9 8080 Microprocessor

The failure rates of an 8080 microprocessor are compared in table(11). The activation energies are very similar except for MIL 217C. It is interesting to note that Intel chose an activation energy of 0.5eV as compared with 1.0eV used by Motorola. The junction temperatures vary considerably, although the junction temperature used by Intel is unknown, but probably around 89° C. The simple RRE failure rate model given in section 3.3 is seen to give a reasonable result, although probably too high by an order of magnitude. Again MIL 217C is seen to be too high by about two orders of magnitude.

Table(12) shows the failure rates of table(11) converted to a common base of junction temperature and activation energy. With the exception of MIL 217C, the predictions agree closely, although they are higher than the predictions for the 6800 by a factor of four. This is because of the higner junction temperature of the 8080 microprocessor. The Intel and Motorola failure rates determined by accelerated testing are seen to agree closely, although tney are felt to be too low.

3.4.10 EPROM

The failure rates of 2716 EPROMs are compared in table(13) at a common junction temperature. With the exception of MIL 217C, the failure rates agree within an order of magnitude which is much better than for microprocessors. The CNET failure rate is seen to agree most closely with the manufacturers failure rate. An activation energy of 0.55eV is used by both MIL 217D and NCSR since this is the energy corresponding to NMOS devices. Intel have found that 0.3eV is more suited to accelerated testing of EPROMs. If the MIL 217D and NCSR failure rates were calculated using an activation energy of 0.3eV, then agreement between the failure rates would be better.

3.4.11 Bipolar ROMs

The failure rates for a 1k bipolar ROM are compared in table(14). With exception of MIL 217C, the failure rates all agree very closely and it is impossible to suggest that one value is more believable than another. It is interesting to note that all reliability data sources use nearly identical activation energies.

3.4.12 Dynamic RAM

The failure rates for a 16k dynamic RAM are compared in table(15). The MIL 217C value seems too high, whilst the Motorola figure seems too low. If the Motorola failure rate, having an activation energy of 1.0eV, is converted to a failure rate, having an activation energy of 0.3eV, then it may be compared sensibly with the Intel failure rate. Both the MIL 217D and NCSR failure rates are high when compared with the manufacturers accelerated test results and the CNET prediction seems to agree more closely as found by Reynolds [35].

3.4.13 Static RAMs

The failure rates of 1k static RAMs are compared in table(16). With the exception of MIL 217C, the failure rates agree closely and it is impossible to suggest that one data source is preferable to another.

3.4.14 Comparison with other field data

Klein [45] compares field failure rates with MIL 217C predictions. For all devices except PROMs, MIL 217C is found to be pessimistic by up to two orders of magnitude. For PROMs, MIL 217C is shown to be too high by an order of magnitude for some devices, whilst it is found to be too low by an order of magnitude for other devices. For microprocessors MIL 217C is pessimistic by up to two orders of magnitude and observed failure rates are in the range 0.3-2.0 f/million hours. For RAMs, MIL 217C is correct for some devices, but is pessimistic by up to two orders of magnitude for the majority of devices, and observed failure rates are in the range 0.1-20 f/million hours. For ROMs, MIL 217C is pessimistic for most devices by an order of magnitude and observed failure rates are in the range 0.08-0.8 f/million hours.

Reynolds [35] concludes that MIL 217C is pessimistic for most microcircuits and that the CNET predictions are more appropriate to "ground fixed" applications.

Daniels et al [46] compare observed failure rates based on field data with predictions made according to MIL 217C, incorporating Notice 1 (May 1980), for twelve pieces of equipment of two types, one being analogue and the other a digital controller. The comparison shows that 79% of predicted values are within a factor of two of the observed values, although MIL 217C is always pessimistic. In fact where microcircuits are concerned, it is proposed that they should have found there to be wider disagreement between predicted and observed values. It was assumed as a basis for the comparison that the temperature rise within equipment is only 10°C above an ambient temperature of 25° C. Thus the failure rate of a 6800 microprocessor is predicted at a junction temperature of 35° C. This junction temperature is much too low as previous discussion has shown. Assuming a thermal resistance of $\theta ja = 50^\circ$ C/W and a typical power dissipation of 0.5W, the temperature rise within a 6800 microprocessor will be at least 25° C, which gives a junction temperature of 50° C. Since the types of microcircuits used in the digital controller are not specified it is impossible to conclude how pessimistic MIL 217C is, especially for microcircuits.

Claridge [47] makes a comparison between observed and predicted failure rates of a large instrumentation and protection system. Although very few integrated circuits are used in the system, a useful lesson can probably be learnt. The system comprises about nine hundred sub-units, each containing about forty components. If the observed failure rates of the sub-units are compared with the predictions, then the failure rates are seen to vary by about two orders of magnitude. If however the failure rate of the systems consisting of nine hundred sub-units are compared, then the predicted and observed failure rates vary by a factor of four with the prediction almost always pessimistic. Unfortunately the data source used for failure rate prediction is not quoted, but this example shows that observed and predicted failure rates can be shown to agree more closely if a comparison is made between a system containing many thousand components, rather than a hundred or so. A statistical analysis is always seen to be more accurate if the sample size is increased.

3.4.15 Recommendations

The most widely used data source for reliability prediction is probably the MIL 217 series. It has the advantage that most commonly used components are listed. It has been shown that many of the predictions of MIL 217D, especially those for microcircuits are pessimistic, although it is better to err on the side of caution. If a more accurate failure rate prediction is required, then it is suggested that the following failure rates and reliability data sources are used for the following components :

Component type	Data source / failure rate
Resistor	NCSR
Decoupling capacitor	MIL 217D
Soldered joint	MIL 217D
Wire-wrap connection	0.0008 f/M hrs
Connectors	CNET
TTL integrated circuits	CNET
Microprocessors	CNET
EPROM	CNET
Bipolar PROM	CNET / MIL 217D
Dynamic RAM	CNET
Static RAM	CNET / MIL 217D

The findings of this section as well as Reynolds [35] indicate a preference for the use of CNET data for microcircuits, rather than MIL 217D, although MIL 217D should not be in error by more than one order of magnitude.

Whichever reliability data source is used, is is recommended that for integrated circuits, the thermal resistance Θja is determined from the manufacturers data or, failing that, from MIL 217D as described previously in this section. The typical power dissipation, determined from the manufacturers data, should be used to calculate the device junction temperature, Tj.

3.5 COMPARISON OF DIFFERENT DEVICE TECHNOLOGIES AND ENCAPSULATION

For a given type of encapsulation, the failure rate depends on the device technology in two ways :

(i) Activation energy - The activation energies corresponding to the failure mechanisms of different technologies are given in table(7) and discussed in section 3.4. There is fairly good agreement between manufacturers and data sources for failure rate prediction.

(ii) Power dissipation - The device junction temperature depends on the ambient temperature and the temperature rise within the microcircuit package, which is dependent on the power dissipation. Technologies such as TTL and ECL exhibit much higher power dissipation than CMOS.

3.5.1 CMOS vs TTL logic

It is shown by Peterson [43] that non-hermetic CMOS devices are more sensitive to moisture activated corrosion and surface instability than TTL devices. Many CMOS devices were found to fail because properties such as input current and leakage current increased, sending the device out of tolerance. This is probably to be expected because of the high input impedance of CMOS. If hermetic CMOS is considered, then it is still reported by Johnson et al [48] that TTL devices generally exhibit a much higher reliability.

The comparison between CMOS and TTL is to some extent a trade-off between temperature and activation energy. Bipolar TTL devices dissipate more power than CMOS, therefore have a higher junction temperature, but their activation energy is lower. CMOS devices fail according to a high activation energy, but dissipate negligible power. Peterson [43] concludes that plastic encapsulated CMOS has a failure rate five to thirty times that of plastic TTL devices. Although there is considerable evidence suggesting that CMOS devices are less reliable than TTL, the findings of Motorola [39] suggest that CMOS, NMOS and HMOS have approximately equal failure rates, based on nearly six million hours of testing at 125° C on seven thousand devices.

Unless design considerations dictate otherwise, it is recommended that TTL logic is used in preference to CMOS for the above reasons.

3.5.2 Hermetic vs non-hermetic encapsulation

There are basically three types of hermetic package :

(i) Ceramic package with metal lid - This type of encapsulation is common for VLSI devices and consists of a piece of ceramic moulded around the lead frame with a central "well" in the package. The die is placed in the well and wire-bonded to the lead frame. The device is then sealed by brazing or soldering on a metal lid. The package is available in both DIL and leadless chip carrier.

(ii) Cerdip - This package is available in both DIL and flat pack and consists of a lower slab of ceramic which has the lead frame moulded into it. The die is placed on top of the lead frame and wire-bonded to it. The package is sealed by a ceramic lid with a glass seal around the edges.

(iii) Metal can - This type of package is rarely used for logic devices and is mainly used for linear and hybrid devices. The microcircuits are contained within a sealed metal can with connections passing through glass seals on the bottom of the package.

Non-hermetic packages consist of a die attached and wire-bonded to a lead frame and then the die and lead frame are encapsulated using an epoxy, silicone, or phenolic resin.

Apart from other considerations, hermetic devices have a lower thermal resistance (typically 50° C/W) than plastic devices (typically 100° C/W). This difference between plastic and ceramic devices may be reduced with the introduction of copper lead frames instead of aluminium as currently used.

Motorola [39] report the advantage of using a copper lead frame and the improvement in plastic device thermal resistance. For devices which dissipate a lot of power, the junction temperature of hermetic devices will be lower than that of plastic devices which should give a more reliable device. If devices are to be used over the full military temperature range of -55°C to +125°C, then it is necessary to use ceramic devices.

References[49,50] suggest that ceramic devices are twice as expensive as plastic devices and Hakim et al [50] propose that ninety per cent of all integrated circuits are plastic. This poses a problem for high reliability users of microcircuits since nearly all devices are available in plastic, but not all are available in ceramic. It is therefore necessary for the high reliability user to consider the use of plastic devices on the grounds of cost and availability. Plastic microcircuits offer lower weight and it has been suggested that the encapsulant which surrounds the die and wire-bonds makes the device more robust.

There are two main disadvantages with plastic devices :

(i) Moisture which is either trapped inside the package when sealed, or which leaks into the package along the lead-frame, has been shown to cause corrosion of the die and wire-bonds, references[43,48,50,54,66].

(ii) Package integrity - If the coefficients of thermal expansion of the die, lead-frame, and moulding compound are not equal, stresses will be set up within the package which may crack the die or break the wire-bonds.

In order to accelerate the failure of devices due to moisture corrosion it is common to perform THB (temperature humidity bias) testing. The device under test is placed under conditions of 85°C/85% rh with a static bias of +5V applied. The biassing of the device is organised to minimise the power dissipation of the device so that heat generated on-chip will upset the test conditions as little as possible. References[51,52,53] propose equations linking the acceleration in failure rate to the temperature and relative humidity. Reynolds [35] suggests that equipment under benign conditions of 30° C/30% rh will undergo an acceleration in failure rate of 550 under conditions of 85° C/85% rh, whilst the corresponding acceleration factor for equipment normally operated under uncontrolled conditions of 12° C/80% rh is 210. The latter acceleration rate is chosen as the rate most appropriate to British Gas "field conditions".

Lycoudes [54] concludes that new plastics have virtually eliminated the early problems of package integrity and that the reliability of plastic devices is comparable to that of hermetic devices, provided that environmental conditions are not extreme.

Results of accelerated THB testing are given by Hakim et al [50] who report that hermetic ceramic devices perform best of all, but that the extrapolated failure rates for plastic devices are acceptable. They conclude that the use of plastic devices in military and high reliability equipment is not recommended under all conditions, but their use under controlled conditions might be acceptable.

Bauer et al [66] recommend that plastic devices may be used with caution in high reliability applications as long as their failure rate is estimated to be four times worse than equivalent ceramic devices.

The results of 85° C/85% rh testing, thermal shock testing, and temperature cycle testing on plastic packages are reported by Motorola [39,40,41]. The results of thermal shock testing are good, especially in reference[40], confirming that new plastics have eliminated early problems. The results of 85° C/85% rh testing, reference[39], are more difficult to interpret. For a random failure process it is a valid assumption to multiply the number of devices under test by the test duration, which gives an equivalent number of device hours under test conditions. However failure due to corrosion is not a random process and should not be analysed by this method. The only conclusion that can be drawn from the Motorola data is that out of a sample of 1717 devices, 18 devices had failed after 1008 hours of testing at 85°C/85% rh. If however this recommendation is ignored and the results of THB testing in reference[39] are used to give a failure rate at 85°C/85% rh, then at 90% confidence the failure rate is 14 f/M hours which agrees well with the results of Hakim et al [50]. If this failure rate is extrapolated to conditions of 70°C/30% rh such as might prevail under Motorola's accelerated temperature testing, then the result is a failure rate of 0.37 f/M hours as compared with the accelerated testing result for plastic devices of 0.2 f/M hours. Although THB testing results should not be extrapolated to give a failure rate, it appears that reasonable results are obtained if extrapolation is performed.

For the above reasons, the results of extrapolated THB testing should be treated with caution.

An interesting result is given by Motorola [39], in which the failure rates of plastic and ceramic devices are given as 0.2 f/M hours and 0.24 f/M hours respectively. This suggests that under favourable conditions, there is no difference between the reliability of plastic and ceramic devices. This finding is also the conclusion of Fox [67].

The graph of figure(8) shows the failure rates of equivalent plastic and ceramic 8085 microprocessors plotted against ambient temperature. The failure rates are calculated according to MIL 217D and CNET data, using the following equations :

MIL 217D

$$\lambda_{plastic} = 0.1295 \exp \left[9270 \left(\frac{1}{298} - \frac{1}{T + 358} \right) \right] + 1.785 \qquad (3.5.1)$$

$$\lambda_{ceramic} = 0.0296 \exp \left[6373 \left(\frac{1}{298} - \frac{1}{T + 315} \right) \right] + 0.533 \qquad (3.5.2)$$

CNET

$$\lambda_{plastic} = 7.28\text{x}10^{4} \exp\left[\frac{-3500}{TA+358}\right] + 1.27\text{x}10^{14} \exp\left[\frac{-11600}{TA+358}\right] + 1.575 \qquad (3.5.3)$$

$$\lambda_{ceramic} = 7.28\text{x}10^{4} \exp\left[\frac{-3500}{TA+315}\right] + 3.64\text{x}10^{13} \exp\left[\frac{-11600}{TA+315}\right] + 0.525 \qquad (3.5.4)$$

The MIL 217D and CNET predictions for ceramic devices agree closely, but predictions for plastic devices disagree by nearly an order of magnitude. The CNET data which is less pessimistic than MIL 217D is felt to be more accurate when considered in relation to this section and the findings of Motorola. In fact under favourable conditions, packages may be even better than CNET data predicts.

It is proposed that the lower failure rate of plastic TTL devices as compared with CMOS is due in part to the heat generated by TTL devices, which tends to drive off any moisture present in the package. Since CMOS generates almost negligible heat, any moisture present in the package will not be driven off and moisture activated corrosion will proceed at a faster rate because of the higher level of relative humidity.

3.5.3 Recommendations

1. TTL logic should be used in preference to CMOS unless power consumption is critical.
2. There is strong evidence to suggest that hermetic devices are slightly more reliable than plastic devices, and it is recommended that ceramic packages are used in all but the most cost-sensitive applications.
3. Failure rate prediction of plastic encapsulated devices should be performed using CNET data in preference to MIL 217D.

3.6 SCREENING AND METHODS OF IMPROVING FAILURE RATES

One of the most common ways of improving the reliability of individual components and electronic systems is the technique of "burn-in". The device under test is operated under normal conditions or more commonly elevated temperature for a period of time which is sufficient to remove the freak population of early failures which occur on the "infant mortality" section of the "bath tub" curve. As components fail, they are replaced and on completion of a successful burn-in, all weak components will have been replaced and the failure of components will assume a constant rate due to random failures. Techniques for determining an optimum burn-in profile are given in references[55,56] since it is important to burn-in for sufficient time to expose weak components, but further burn-in time will be a waste of time, or in the case of some components, will exhaust some of the useful life of the components.

For high reliability applications, components are available from manufacturers which have satisfied agreed standard screening procedures. Such screening may involve visual, mechanical, and electrical tests. Integrated circuit failure rate predictions in MIL 217D refer to the screen level of a device according to US MIL STD 883. This screening procedure was introduced in 1968 in order to create an economically feasible standardised integrated circuit screening flow which would achieve in-equipment failure rates of 0.8 f/M hours and 0.04 f/M hours for class B and class A devices respectively. The standard has been modified since 1968 and now represents a very tough screening specification. An equivalent screening specification is covered by British Standard BS9400. The screening specifications for integrated circuits are covered in detail by National Semiconductors [58].

The relation between MIL 217D screen level, screening specification, quality factor π_q , and typical relative cost is given in table(17).

Considerable differences exist between MIL STD 883 class S and BS9400 class A, for example, class S screening specifies non destructive wire-bond pull tests, however the two classes are broadly similar. The screening classes B and C of MIL STD 883 are very similar to those of BS9400, and BS9400 includes a fourth class D not covered by MIL STD 883. The screening requirements of BS9400 are shown in figure(9). Considering screening level A, the process begins with a visual examination of the die and wire-bonds under 30-200 times magnification. The condition A visual examination is more stringent than B and specifies maximum allowable deformation in metalisation widths etc. The visual examination is obviously very labour intensive and therefore expensive. The device is then encapsulated and baked to stabilise it. The integrity of the package and wire-bonds is now tested by temperature cycling, mechanical shock, and constant acceleration tests. The package seal is tested next by fine and gross leak tests and then the device is tested electrically. In order to detect weak devices, the device is burnt-in at high temperature, after which it is tested again electrically. A high temperature reverse bias is now applied and the device is again tested electrically. Finally the device undergoes an X-ray examination to detect bad wire-bonds, particles inside the package, or bad die adhesion. If all these tests are passed successfully, the device continues to routine testing before being released. Screening levels B, C, and D form a sub-set of level A as well as having a less stringent visual examination and burn-in. The only difference between screening level B and C is the final burn-in of class B devices. It is therefore recommended that devices are procured according to BS9400 class C at a typical relative cost of six times that of a plastic device. The equipment is then assembled and subjected to a burn-in at 125°C for 160 hours. In this way, class C devices which survive the assembly and burn-in could be considered to be equivalent to class B devices. According to table(17), this burn-in process should increase the reliability by a factor of three, making the equipment an order of magnitude more reliable than equipment using plastic devices, but at only six times the cost.

As an alternative to this burn-in procedure, British Telecom burn-in some high reliability equipment for 500 hours at 85°C.

3.6.1 Derating and cooling

Reducing the stresses on a component will increase its life. Jensen [57] suggests that components should be derated according to the following guidelines for high reliability applications :

Resistors - derate power to 0.5 rated

Electrolytic capacitors - maintain core temperature below 70°C

Semiconductors -	Power derating = 0.3
	Current derating = 0.5
	Voltage derating = 0.6

Wherever possible, the junction temperature of semiconductor devices should not exceed the maximum values of table(18).

The reliability of CMOS and linear integrated circuits can be improved by derating the power supply voltage. This is not possible with TTL and NMOS logic which must be supplied with +5V ±5%. A 5% reduction in supply voltage would increase the reliability slightly, but at the expense of reduced noise immunity and tolerance of power supply voltage fluctuations.

Since the failure rate of integrated circuits is approximately exponentially dependent on temperature, it is important to minimise the junction temperature. Therefore circuit boards should be mounted well away from power supplies, heat sinks etc. and provided with as much cooling as possible. In many cases, the use of forced cooling is advantageous and can dramatically reduce the failure rate of equipment.

The power dissipation within a device is also dependent on the output loading. Jensen [57] recommends that the fan-out of digital circuits and the loading of linear circuits is derated by a factor of at least 0.8.

3.7 SOFTWARE AND TRANSIENT ERRORS

Before discussing malfunctions in digital controllers, it is necessary to define the terms "error" and "fault". They are defined by Anderson and Lee [60] as :

> Error - Part of an erroneous state that constitutes a difference from a valid state.
>
> Fault - An error in a component or the design of a system will be referred to as a fault in a component or system.

An error can thus be seen as the manifestation of a fault, a single fault producing one or more errors.

Faults in digital controllers are either permanent (hard) or transient (soft). Permanent faults are normally easy to diagnose and repair and are caused by the failure of components and software. Unless a software fault is fundamental to the operation of the controller, it is usually possible to recover from the fault by the technique of "exception handling", described later. Careful testing should reveal permanent software faults, however hardware may fail at any time, as described in section 3.4, and redundancy techniques are required to protect against such failures. Although it is difficult to predict accurately the failure rates of hardware as demonstrated in section 3.4, sufficient data exists to make an estimate which will be correct within an order of magnitude or so.

Transient faults are much harder to deal with since they are by their very nature of short duration, and may be caused by many different sources. It is not always possible to detect the cause of a transient fault, although a particular class of transient fault may be injected into a system to measure the system's fault-tolerance to that class of fault.

internal operation of most microprocessors is dynamic and relies on stored charge, so it is reasonable to expect that smaller device geometries in microprocessors will increase their sensitivity to alpha radiation.

Motorola [61] describe the design procedure for a 64k dynamic RAM with particular attention to the effect of alpha radiation. Transient error rates as high as 500 f/M hours are quoted for initial production units in 1979. Transient error rates for current production devices are quoted as 22 f/M hours, being 22 times greater than permanent failure rates.

(iii) Design faults - The physical construction of a piece of equipment can give rise to transient faults due to screening, layout, or decoupling problems. Timing and logic threshold faults are a further class of fault and are sometimes affected by temperature, making a piece of equipment malfunction over a certain temperature range, whilst it works perfectly at other temperatures. For this reason it is important to test fully equipment over its complete operating temperature range.

(iv) Software - Transient software faults are usually the result of programming errors which may be either incorrect algorithm specification or coding errors in machine language or high level language. Exceptionally faults are caused by "bugs" in the cross-assembler or compiler. Methods for improving software reliability are given in section 5.4.

(v) Environmental effects - Often digital controllers are sited in harsh electrical environments which may cause transient faults due to noise on the electrical supply or high electromagnetic field strengths. LSI circuits are particularly prone to electrical interference and in the worst environments, further measures may be required other than attention to earthing.

screening, and provision of a "clean" power supply.

A powerful tool for the correction of transient faults is the provision of fault-tolerant software which is written to check for faults and provide any necessary correction.

3.8 SUMMARY

Often, as is the case for the experimental controller, a design requirement is that a piece of equipment does not exceed a certain failure rate. It is also necessary to predict the failure rate of equipment so that alternative designs may be compared. The previous chapter has shown that it is difficult to predict accurately the failure rate of equipment. Components are assumed to fail according to the "bath-tub" curve and it is assumed that during the useful life of the component, that the failure rate is constant. In order to screen semiconductor equipment and to produce failure rate data, it is essential to accelerate the failure process and the Arrhenius acceleration equation is universally used when calculating the acceleration in failure rate due to a rise in temperature.

The wide variation in failure rate prediction data is discussed in section 3.4 and the comparison of different predictions with field data suggests that some data sources are preferable to others. The recommended failure rate prediction data source for each type of component is given at the end of the section. For most components, the CNET data [38] is to be preferred.

A comparison of device technologies shows that TTL is more reliable than CMOS, which is probably due to the high input impedance of CMOS which makes it very sensitive to moisture, and the higher power dissipation of TTL which is sufficient to dry out the integrated circuit package. Unless design considerations require the use of CMOS in low power applications, it is recommended that TTL is used in preference to CMOS.

There is considerable disagreement as to whether plastic encapsulation is worse than hermetic encapsulation. The findings of Motorola suggest that there is no difference between the two methods of encapsulation, but all other literature proposes that plastic devices should be used with caution in high reliability applications. It is clear that improvements have been made in the techniques of plastic encapsulation, and that plastic devices are only significantly worse under conditions of high humidity. It is recommended here that plastic devices are only used in the most cost sensitive applications.

Components should not be operated at their maximum ratings and guidelines for derating components are given in section 3.6. Since the failure rate of components is exponentially dependent on temperature, the provision of forced cooling will dramatically reduce the failure rate.

Although considerable importance is attached to the prediction and prevention of permanent failures, it is reported that transient errors are much more common [2,61,78] and it is important to protect against this class of fault. The governor controller is tolerant of most classes of transient fault. Some of the protection is provided in hardware and is transparent to the software, whilst the rest of the protection is provided by software. Software fault-tolerance is a powerful tool for the detection and correction of transient errors.

CHAPTER 4

PREDICTION OF RELIABILITY IMPROVEMENT DUE TO FAULT-TOLERANT TECHNIQUES

4.1 METHODS OF EXPRESSING RELIABILITY

Unlike the physical parameters of an electronic device such as resistance, capacitance, voltage etc. which are easy to use and to measure, the reliability of a component is a very difficult quantity. The whole of reliability engineering relies on the use of statistics and therefore great caution must be exercised, since statements are made concerning statistical quantities rather than physical quantities.

The failure of electronic components is shown by Cluley [1] to fit the Poisson distribution. The reliability of a system is defined as :

> The probability that the system will perform its required function, under stated conditions, for a stated period of time.

Normally a system is required not to fail, so the reliability of a system is given by the probability that no failures are observed during a given time period. The probability is given by :

$$P(0) = \exp(-\lambda t) \qquad (4.1.1)$$

where : t = time interval

λ = failure rate of the component

$P(0)$ = probability of no failures

which can be rewritten as :

$$R = \exp(-\lambda t) \qquad \text{where : } R = \text{reliability of the component} \qquad (4.1.2)$$

In order to calculate the reliability it is necessary to know the failure rate of the component, λ, which can be measured by means of life testing or may be predicted as described in chapter three.

The concept of "mission time" is useful when it is required to predict the probability that a piece of equipment will operate successfully over a given time period. The probability of success is given by :

$$P = \exp(-\lambda T) \qquad \text{where : } \quad T = \text{mission time} \qquad (4.1.3)$$

Such an analysis is often used in space and defence when it is required to calculate the probability of a successful space mission or the probability that a missile will hit its target.

In an industrial environment, mission time is a less useful concept as it is normally required to predict how long a piece of equipment will operate before failing. The mean time to failure, MTTF, is defined by :

$$\text{MTTF} = \int_{o}^{\infty} R(t)\,dt \qquad (4.1.4)$$

$$= \int_{o}^{\infty} \exp(-\lambda t)\,dt$$

$$= \frac{1}{\lambda}$$

If the MTTF is substituted for the mission time for a piece of equipment, then the reliability is given by :

$$R = \exp\left(-\lambda \frac{1}{\lambda}\right)$$

$$= \exp(-1)$$

$$= 0.37$$

This means that there is only a 37% chance that a piece of equipment will operate for a time equal to the MTTF. It is for this reason that the MTTF should be used carefully.

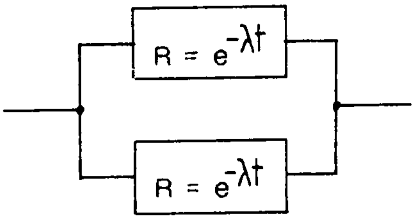
4.2 EFFECT OF REDUNDANCY ON RELIABILITY

For n units in series, each having failure rates λ_1 , λ_2 , λ_3 . etc., the system reliability is given by :

$$R = \exp\left[-(\lambda_1+\lambda_2+\lambda_3+ \text{........})t\right] \qquad (4.2.1)$$

The failure rates may simply be added together to calculate the total system failure rate. This is the case for most failure rate predictions and the failure rate of the governor controller is calculated by summing the failure rates of the individual components.

The simplest form of redundancy is a parallel standby system where either component can fail without the system failing.



Assuming that the two components fail independently, the probability of failure is given by :

$P(f) = (1-R)^2$ where $P(f)$ = probability of failure

since : $P(s) = 1 - P(f)$ where $P(s)$ = probability of success

the reliability of the system is given by :

$$R = 2R - R^2 \qquad (4.2.2)$$

Cluley [1] shows that the MTTF of the system is equal to :

$$\text{MTTF} = \frac{3}{2\lambda}$$

which is only 50 per cent more than the MTTF of a single channel.

If a TMR system having perfect voters is considered, the reliability is given by :

$$R = 3R^2 - 2R^3 \qquad (4.2.3)$$

and the MTTF is given by :

$$MTTF = \frac{5}{6\lambda}$$

In this case the MTTF is less than that of a single channel. At first sight this would suggest that a TMR system is inferior to a duplicated or a single channel, however a TMR system can be shown to be more reliable if repair or the concept of mission time is considered.

A graph is plotted in figure(10) of the reliability of a simplex and a TMR system against λt, the normalised mission time. It is seen that there is a crossover at which point the TMR system ceases to be more reliable than the simplex system. The position of this point may be calculated by equating the simplex and TMR reliabilities according to :

$$R = 3R^2 - 2R^3$$

The solution R=0.5 corresponds to $\lambda t=0.693$. The crossover point which ocurrs before $\lambda t=1$ explains why the MTTF of a TMR system is worse than that of a simplex system. For a system without repair, the useful life of a TMR system is limited to $\lambda t=0.693$ or less.

4.3 EFFECT OF MAINTENANCE ON A SYSTEM

The previous sections have not considered repair and a system was considered to have exceeded its useful life once it had failed. This is typical of space and defence equipment. Systems which are repaired when they have failed will now be considered; this situation can be considered typical of industrial equipment such as the governor controller. Since systems are repaired between failures, the term MTBF (mean time between failures) is used to replace the term MTTF. The mean time to repair of a failed system is given by the MTTR. The availability of a system is defined as :

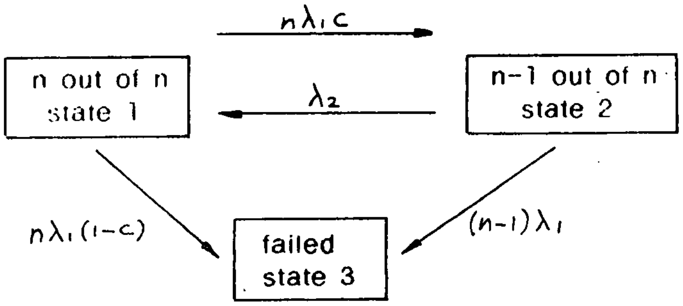
$$A = \frac{MTBF}{MTBF + MTTR} \qquad (4.3.1)$$

The availability may be interpreted as a probability that the system is in working order at any instant. The concept of availability is useful when planning the maintenance of equipment. If the MTBF of a piece of equipment is known, and it is required to achieve a certain level of availability, then the necessary MTTR to achieve that level of availability may be calculated.

4.3.1 Redundant systems

The introduction of redundancy into a system, together with repair, allows considerable improvements to be made in equipment reliability. For example if one channel of a TMR system fails and is repaired before a further channel fails, then the system will operate without interruption. In order to achieve interruption free operation, it is essential to know when components fail so that they may be replaced before further failures occur.

For a repairable system with n units of which one is a spare, the Markov failure model is given below :



where : λ_1 = failure rate of a single unit

λ_2 = repair rate

c = coverage

The coverage is a number between zero and unity and is a measure of the proportion of faults which are covered by the redundancy in the system, where unity is equivalent to complete coverage. The transfer rates between states are given by :

$$\frac{dP_1(t)}{dt} = - n\lambda_1 P_1(t) + \lambda_2 P_2(t) \qquad (4.3.2)$$

$$\frac{dP_2(t)}{dt} = n\lambda_1 c P_1(t) - \lambda_2 P_2(t) - (n-1)\lambda_1 P_2(t) \qquad (4.3.3)$$

$$\frac{dP_3(t)}{dt} = n\lambda_1(1-c)P_1(t) + (n-1)\lambda_1 P_2(t) \qquad (4.3.4)$$

The state transition matrix is therefore given by :

$$\begin{bmatrix} -n\lambda_1 & n\lambda_1 c & n\lambda_1(1-c) \\ \lambda_2 & -[\lambda_2 + (n-1)\lambda_1] & (n-1)\lambda_1 \\ 0 & 0 & 0 \end{bmatrix}$$

The solution of these equations in terms of the MTFF (mean time to first failure) is given by Dhillon and Singh [71] as :

$$\text{MTFF} = [1\ 0]\,[-Q]^{-1}\,U \qquad \text{where : } U = \text{unit matrix} \qquad (4.3.5)$$

Q is the reduced state transition matrix which is given by :

$$[Q] = \begin{bmatrix} -n\lambda_1 & n\lambda_1 c \\ \lambda_2 & -[\lambda_2 + (n-1)\lambda_1] \end{bmatrix}$$

Hence :

$$[-Q]^{-1} = \frac{1}{\Delta}\begin{bmatrix} [\lambda_2 + (n-1)\lambda_1] & n\lambda_1 c \\ \lambda_2 & n\lambda_1 \end{bmatrix}$$

The MTFF is given by :

$$\text{MTFF} = [1 \ \ 0]\,[-Q]^{-1}$$

$$= \frac{\lambda_2 + (n-1)\lambda_1 + n\lambda_1 c}{n(n-1)\lambda_1^2 + n\lambda_1\lambda_2(1-c)} \qquad (4.3.6)$$

For $\lambda_1 \ll \lambda_2$ this equation can be simplified to :

$$\text{MTFF} = \frac{\lambda_2}{n(n-1)\lambda_1^2\left[1 + \frac{(1-c)}{(n-1)}\frac{\lambda_2}{\lambda_1}\right]} \qquad (4.3.7)$$

Which is the same equation as that given by Arnold [72].

Equation(4.3.6) may be checked for a TMR system with no repair which has 100 per cent coverage using the values :

$$\lambda_2 = 0$$

$$c = 1$$

$$n = 3$$

this gives :

$$\text{MTFF} = \frac{5}{6\lambda}$$

which agrees with the expression for MTFF derived previously.

The concept of coverage as applied to software is easy to define. It is the proportion of faults that can be detected and recovered from. For hardware it is proposed that the coverage is defined by :

$$c = \frac{\text{failure rate of protected circuitry}}{\text{failure rate of protected + unprotected circuitry}}$$

The unprotected circuitry includes all common circuitry which contributes to common mode failures.

4.4 METHODS OF EXPRESSING IMPROVEMENT

Any improvement obtained in the reliability of a system can be shown by an improvement in the availability or the reliability of the system for a given mission time. In an industrial environment it is useful to consider any improvement that can be made to the MTTF of a system. The MTTFIF (mean time to failure improvement factor) is defined as :

$$\text{MTTFIF} = \frac{\text{MTTF (fault tolerant)}}{\text{MTTF (non fault tolerant)}}$$

Hence the MTTFIF for a system with n units of which one is a spare is given by the modified equation(4.3.7) :

$$\text{MTTFIF} = \frac{\lambda_2/\lambda_1}{n(n-1)\left[1 + \frac{(1-c)}{(n-1)}\frac{\lambda_2}{\lambda_1}\right]} \qquad (4.4.1)$$

Hence the MTTFIF for a TMR system with repair as shown by Pearson and Preece [73] is given by :

$$\text{MTTFIF} = \frac{\lambda_2/\lambda_1}{6\left[1 + \frac{k}{6 + 2k}\frac{\lambda_2}{\lambda_1}\right]} \qquad (4.4.2)$$

where : λ_2 = repair rate

λ_1 = failure rate of a single unit

$k = \frac{\lambda_v}{\lambda_1}$

λ_v = failure rate of voters

A graph of MTTFIF against λ_2/λ_1 for different k factors is shown in figure(11). For large values of k, it is seen that little improvement can be made in the MTTF as the ratio λ_2/λ_1 is increased. For small values of k, large improvements can be made to the MTTF as the ratio λ_2/λ_1 is increased. For a given value of k it is seen that the curve reaches a plateau, after which further increases in λ_2/λ_1 cause little extra improvement in MTTF. The point at which this plateau is reached depends on k and this point should be used to determine the minimum time to repair. Any further improvements in the repair time will be wasted, since the MTTF cannot be improved much more.

These findings can be summarised by stating that in order to achieve improvement in the MTTF of a system by repair, the fault coverage must be high.

A similar approach can be used to determine the improvement to be gained by using SEC/DED Hamming code protected memory. In this case there are n RAM chips of which one is a spare. The coverage is calculated as previously defined :

$$c = \frac{\text{failure rate of RAM chips}}{\text{failure rate of RAM chips + Hamming code circuitry}}$$

4.5 SUMMARY

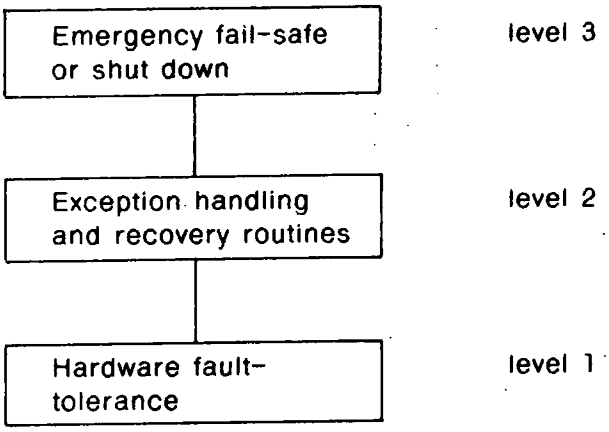
Two types of reliability analysis have been discussed. The first is concerned with predicting the probability of failure free operation of a non-repairable system for a given mission time. This analysis is typically used in space and defence applications. The second type of analysis is directly applicable to the governor controller. The individual failure rates are added together to give the total failure rate. The effect of maintenance and repair is then considered and the MTTF of the equipment is calculated. In order that a repairable redundant controller may be compared with a non-redundant system without repair, the concept of MTTF improvement factor is introduced. Consideration of the MTTF improvement factor allows the optimum repair time to be calculated.

CHAPTER 5

TECHNIQUES FOR IMPROVING A SYSTEM'S FAULT-TOLERANCE

5.1 LEVELS OF FAULT-TOLERANCE

A structured approach to fault-tolerance implementation is proposed by Halse et al [74]. Fault-tolerant techniques are assigned levels of recovery as shown below.


Emergency fail-safe or shut down
level 3
Exception handling and recovery routines
level 2
Hardware fault-tolerance
level 1

At the lowest level, level 1, complete recovery is provided from a limited range of fault types. Recovery is embedded in the hardware design of the system and in the experimental controller consists of recovery from voting and memory errors. Level 2 provides a more limited recovery from a wider range of faults. Any state of the controller which is outside its specification is termed an "exception". As the processing of data proceeds, exception handling routines check for errors which may be caused by external stimuli or design faults, and initiate recovery from them. Recovery routines perform the logging of errors and identical fault recovery may be used to recover from different types of fault. At the highest level, level 3, limited or degraded recovery, such as a fail-safe emergency shut-down, is provided from a wide variety of fault types.

A structured approach to assigning levels of fault recovery is useful, since if a given level fails to recover from a fault, recovery is passed up to the next level until recovery or soft failure is successfully completed.

5.2 DESIGN TOOLS

Two commonly used design tools for assessing the reliability of a piece of equipment are FMECA and FTA which are defined below.

5.2.1 FMECA

Fault mode, effect and criticality analysis is a formal design exercise where each component in a system is considered and a failure rate is assigned to each mode of failure. For instance a resistor might fail open circuit 25 per cent of the time and fail out of tolerance for 75 per cent. The consequence of resistor failure is listed, together with a criticality rating. Failure open circuit might result in failure of the circuit, whilst parameter drift might result in a degraded but acceptable performance. An example of a FMECA is given below :

Item	Failure mode	failure rate f/M hours	effect	criticality rating(1=low 3=high)
resistor R1	open circuit	0.002	no output from circuit	3
resistor R1	parameter drift	0.006	reduced output	1
capacitor C1	short cct.	0.002	no output	3
capacitor C1	open circuit	0.002	reduced gain at high freqs.	2

Hence the failure rate at each criticality rating may be determined, rather than assuming all failures are catastrophic. This analysis is well suited to analogue circuits and data sources such as NCSR [34] give the percentage failure for each mode. The criticality analysis is of little use for systems containing many digital integrated circuits. Failure rate data gives no information about failure modes and it is assumed that any failure within complex integrated circuits results in total failure of the device.

In this case, the FMECA is shortened to FMEA, fault mode effect analysis. If the failure rates of all the components in the system are tabulated, then components having a high relative failure rate can be identified and corrective action taken. An example of a FMEA is given in section 7.3 when the microprocessor was found to dominate the total failure rate.

5.2.2 FTA

Fault tree analysis is complementary to FMECA and FMEA. Instead of being a bottom-up process it is a top-down process. The analysis starts by considering failure modes of the complete system and then working downwards to identify the part failures which could generate such system failures. Conventional logic symbols are used to combine events. An example of a FTA for a pressure trip is shown in figure(12). If the pressure exceeds the preset value it is required to trip the process. To prevent spurious trips, two pressure transducers are used, both of which must signal overpressure before action is taken. If probabilities are assigned to each fault, then the probability of an erroneous trip may be calculated.

5.3 HARDWARE

When considering the failure rate of hardware it is useful to remember that using less hardware will decrease the failure rate, since there are fewer components to fail. A well managed and designed piece of equipment is likely to be more reliable than a hurried design where the main acceptance test is whether the equipment works or not.

Several methods are now proposed for improving the fault-tolerance of digital controllers. Many of these methods are used on the governor controller described in chapter eight.

5.3.1 Watchdog timer

A watchdog timer is probably one of the simplest and most effective devices that can be added to a system. It will not cure all ills, but should prevent the system remaining in a crashed state. In its simple form, the watchdog consists of a monostable which is triggered under software control by the syatem it is protecting, at time intervals less than the pulse width of the monostable. In this way the monostable generates a constant pulse. If the retriggering pulse is not received, then after a delay set by the monostable, it times-out and a reset pulse is generated for the system. It is important that the watchdog is retriggered once only per program cycle, and this is usually performed on each pass through the main control loop. There is a small probability that the controller can stick in an erroneous loop, continually retriggering the watchdog, and the watchdog will fail to reset the system. To protect against this more complex watchdogs can be used. As the controller passes in an orderly sequence through a set of instructions, a signature or key is output to the watchdog which is only retriggered if it receives this key in the correct sequence.

5.3.2 Snake

A snake is a long sequence of NOPs or restart instructions which occupy any unused address space. An erroneous jump into this address space causes the program to "slide" down the snake until vectored recovery is executed at the end of the snake. Unused address space is connected so that an instruction fetch from non-existent memory fetches a no-operation or software restart instruction. A software restart instruction is used to jump back into the program. If no-operation instructions are used, then the controller will execute a whole series of NOPs until the program counter arrives at a valid address range. The 8085 microprocessor ideally lends itself to this technique as described in chapter eight. If non-existant memory is pulled high, an instruction fetch will read FF and a RST7 will be executed. If the microprocessor does not support software restarts and if full memory decoding is used, memory selects corresponding to non-existant memory can be used to force the instruction for a NOP or a restart onto the data bus.

5.3.3 Power supply levels

Many logic circuits are only guaranteed to function over a ± 5 per cent power supply variation. Power supplies should be set to the middle of this range to give maximum protection against noise. It is important to detect transient undervoltage of the power supplies and to reset the system, since transient undervoltages have been observed to crash microprocessors. A dip in the +5V supply to a microprocessor by only 1V was found to crash it and such a dip was too fast to be detected by the power-on-reset circuitry.

5.3.4 Output verification

After an output function has been performed, the output is read back and is compared with the value which has just been output. In this way it is possible to test for failure and transient faults in the output circuitry.

5.3.5 Component redundancy

Critical circuit components can be duplicated with a spare. This has both cost and reliability advantages over a TMR system, but suffers from the severe drawback that with only two components it is difficult to determine which is in error. A TMR configuration overcomes this difficulty by taking a majority vote. Chapter four has shown that with maintenance, large improvements can be made in the reliability of equipment using a TMR configuration, however care should be taken not to exceed the "useful life" of a TMR system without repair. Voting can be performed in software or by the logic circuit of figure(5).

5.3.6 Memory protection

The addition of an extra parity bit to memory allows errors to be detected as long as an odd number of bits is in error, but it is not possible to correct errors. A better type of protection is the SEC/DED Hamming code which will correct single bit errors and detect double bit errors. The disadvantage with this code is that the memory word length is increased since check bits must be stored as well as data bits.

For small systems the coding can be performed in ROM as described in chapter eight, but for larger word lengths there are several VLSI circuits available which implement SEC/DED Hamming code on 16 bit word lengths, references [59,79,80]. For small systems, the main benefit is to be gained from the correction of transient errors, however large systems benefit in addition by the correction of permanent errors.

5.3.7 Temperature

As shown in chapter three, the failure rate of semiconductors increases exponentially with temperature. It is therefore essential to keep the system temperature as low as possible and to use forced cooling if necessary.

5.4 SOFTWARE

Structured programming is essential in order to achieve reliable software. The software is divided into easily managable modules which are written and debugged separately and then linked together to form the complete software package. For small systems, having a program size of a few kilobytes it is probably better to write programs in assembler. If assembler is used, each machine instruction is defined by the programmer and is under better control than code produced by a high level language. Because of the complex nature of recovery routines and the difficulty in separating hardware and software in a small controller, it is better to write recovery routines in assembler.

As more complex microprocessors such as 16 bit devices are used and the program size is increased above a few kilobytes, it becomes increasingly difficult to write in assembler. There is a trade off between writing in a high level language which is easier to document and modify at a later date probably by another programmer, and writing in assembler which is more efficient and where the code produced is under tighter control. It is suggested that for large software packages, only the recovery routines are written in assembler and the rest is written in a high level language such as Pascal which offers structured programming. The machine code produced by the high level language should be examined to ensure that it meets the high reliability requirements of the controller.

5.4.1 Exception handling

Any abnormal response in a controller is termed an "exception" which may be caused by transient or design errors in either hardware or software. Exception handling is a powerful technique, implemented in software, where the software continually checks itself for errors and initiates recovery if any are detected. A simple form of exception handling would be to test the input values read into a controller and reject them if they are outside a reasonable range. When exceptions are detected, techniques such as "roll-back" are useful in executing recovery. Roll-back is a form of time redundancy which repeats the block of code in which an exception has been detected. In this way it is not necessary to restart the controller and only a small block of code need be repeated. The recovery blocks described in chapter eight are a crude type of roll back and are illustrated in figure(13). The program is divided up into a number of routines. Each routine inputs a few values, performs some calculation or control action, and then continues onto the next routine. Ideally routines pass few variables between each other which should be contained in microprocessor registers, since the recovery blocks of chapter eight only restore register status following a crash, and make no attempt to restore RAM contents, which could be included.

As shown at the end of each routine, a recovery block is generated. The system is shown to crash after routine 4, so a recovery block is executed which causes routine 4 to be repeated before passing onto routine 5.

5.5 SELF-TESTING

Self-testing or health monitoring of controllers can be performed in spare execution time and is carried out as a background task which should not interrupt control of the process. Chapter four has shown that large improvements can be made in the reliability of equipment if the MTTR is reduced. In order to repair equipment, it is necessary to know that it has failed which means that self-testing is essential in redundant systems where the redundancy will often mask component failures.

Self-testing can be divided into two groups :

(i) Diagnostic - When an exception is detected, the cause should be determined and transient faults distinguished from permanent faults which require to be repaired. Fault logging may point towards a suspect component or area of bad design if repeated transient faults are logged.

(ii) Preventative - At regular intervals the system modules are tested. Such tests are a checksum test of ROM, test of RAM, input/output circuitry etc.. In the particular case of gas regulator valves where the detailed response characteristics of the system are known, it might be possible to predict mechanical valve failures, such as a sticking valve, by injecting a disturbance into the system and analysing the response.

To a limited extent a microprocessor can test itself. Firstly a kernel of essential instructions are verified which can then be used to test other instructions. This technique is described by Hunger [75] with particular reference to the 8085. The self-testing sequence is most efficiently designed with reference to the manufacturers gate model of the microprocessor and a coverage of 60% is quoted for d.c. chip faults.

The need for board testing equipment can be eliminated if boards are designed to test themselves. Daniels and Fasang [76] describe an 8085 microprocessor board that will test itself. The self-test feature can be used as part of the manufacturing process as well as being useful in installed equipment for the self-diagnosis of faults.

If an error is detected during self-testing, an attempt can be made to reallocate components such as RAM and control of the process could continue. Such a technique is called "graceful degradation". If the fault is too serious to continue, the process can be made to fail safe. The provision of telemetry to the controller, used for fault reporting, can achieve very short MTTRs. If no telemetry is available, faults must be logged as they occur, and can then be dealt with as part of routine maintenance

Faults in redundant systems are often easier to diagnose , because failure in a non-redundant system will just cause it to cease operation. If a redundant system can continue to operate in the presence of a fault, it is possible to give a detailed diagnosis of the fault which will reduce the repair time of equipment and skill required.

5.6 SUMMARY

In a small controller such as the governor controller, it is difficult to separate the hardware from the software. A structured approach to the implementation of fault-tolerance by assigning fault recovery levels ensures that most classes of fault will be detected and recovery executed.

A valuable tool for identifying the most unreliable parts of a system is FMECA which was used on the governor controller when deciding where it was necessary to incorporate fault-tolerance. The complementary FTA is discussed for the sake of completeness, and was not used when designing the governor controller because FMECA was found to be more suitable.

Hardware techniques such as a watchdog timer, snake, power supply undervoltage detection, output verification, memory protection, and component redundancy are all used on the governor controller.

Exception handling routines are initiated by hardware on the governor controller and software fault-tolerance is used in selecting the best pressure information from the redundant transducers. Much of the governor controller will test itself, and the nature of all transient and permanent faults is logged. At regular intervals a self-test is performed on the solenoid valves and the self-test software could be expanded to test the complete system.

CHAPTER 6

EFFECT OF SYSTEM ARCHITECTURE ON RELIABILITY AND COST

6.1 CHOICE OF MICROPROCESSOR

The first decision to be made when choosing a microprocessor is the processing power required, that is whether a single chip, 8 bit, or 16 bit device is required.

6.1.1 Single chip microprocessors

If the process consists of a single control loop and little arithmetic is involved, it is likely that a single chip device will have sufficient processing power. Most single chip microprocessors contain internal read only memory (ROM) and random access memory (RAM) which makes the sharing of memory, Hamming code protection, and the provision of a spare copy of ROM impossible unless the single chip system is expanded. If a FMEA of the controller shows that the failure rate is too high, it is necessary to incorporate some form of redundancy according to the guidelines of section 6.3. Failure of the controller could be detected and a standby switched in. Whilst this is satisfactory for permanent faults, it offers little protection against transients, since with only two controllers it is impossible to arrive at a majority decision should the controllers differ. It is for this reason that the TMR configuration is recommended. Voting in a single chip TMR system could be either in hardware or by software.

If the voting were performed in software, a very reliable controller consisting of only a few components could be constructed.

6.1.2 Eight bit microprocessors

The design of the controller reported in chapter eight is an example of a redundant 8 bit microprocessor controller where most of the redundancy is implemented in hardware. Although the hardware techniques described could be used with most 8 bit microprocessors such as the Z80, MCM6800 etc., the 8085 hardware is ideal for a redundant controller. The provision of serial input/output pins removes the need for three separate UARTS and the provision of four maskable and one non-maskable interrupts removes the need for an interrupt controller. Several interrupts are required because hardware exception handling routines are called by interrupts. The 6800 microprocessor is less suitable because it only has one maskable and one non-maskable interrupt, and has no serial input/output pins. The instructions for NOP and SWI are not ideal for implementing a "snake".

Possibly a better microprocessor than the 8085 would be the NSC800 which is identical in hardware to the 8085 except that no serial input/output pins are provided. The instruction set is compatible with the 8085 and consists of the more powerful Z80 instruction set.

If much of the redundancy were to be performed in software at the expense of processor throughput, it is possible that different microprocessors would be more suitable.

6.1.3 Sixteen bit microprocessors

Modern 16 bit microprocessors are really outside the definition of "small" controllers and are almost as powerful as mini-computers. It is probably better to apply more of the redundancy in software because of the processing power and increased hardware and software complexity.

The Texas 9900 microprocessor differs from other 16 bit devices because it does not perform calculations on internal registers, but on a block of registers held in RAM which are pointed to by the internal "workspace pointer". Thus if the workspace pointer is corrupted, all registers will be addressed incorrectly and corruption or permanent failure of RAM will cause register errors.

Voting at bus level with 16 bit microprocessors is not to be recommended because of the complexity of the hardware and the number of signals. Instead it would be better to arrange the three microprocessors as a master and two slaves sharing common memory. Communication between the processors would then be by the common RAM and the voting would be mainly performed in software.

It is likely that 16 bit microprocessors will have a lot of RAM which will probably be shared as well. The RAM should be protected and it is suggested that 16 bit SEC/DED Hamming code integrated circuits could be used to protect the RAM, references [59,79,80] instead of the circuitry proposed in chapter eight.

Large software packages will be required for a 16 bit controller which are better written in a high level language. Assembler should only be used as necessary and for critical areas such as voting and recovery routines as discussed in chapter five.

In high reliability applications, 16 bit microprocessors should only be used if an 8 bit device is not powerful enough, or if the addition of a maths processor to an 8 bit system is less reliable than a 16 bit system. The increased complexity of 16 bit systems over 8 bit systems leads to higher failure rates.

6.2 MAJORITY VOTING

Majority voting in a TMR system can either be performed in hardware or software, the choice of which is influenced by many factors.

6.2.1 Hardware

For a small controller using single chip microprocessors and which only has a few outputs, it is easier to vote on the outputs in hardware. Discrete logic can be used, connected as shown in figure(5) or several channels of voting and error detection could be contained in a FPLA as described in chapter eight. If the failure rate of the system is approximated to that of the voter, then a low failure rate is predicted.

typically 0.02 f/yr as compared with 0.5 f/yr for a single microprocessor, according to MIL-217D.

Voting on inputs is probably better performed in software, although if the total software package is small, the fault coverage will be reduced if the extra fault-tolerant software becomes too large and the system reliability will be reduced.

6.2.2 Software

Three systems can be connected together in a ring, whether they are single chip or 16 bit microprocessor based, as shown in figure(6). Each microprocessor can communicate with its two neighbours. If a processor gets no response from one neighbour, but can communicate with its other neighbour, then the first neighbour must be in error. At any time one microprocessor must be in charge and act as the master and the other two microprocessors act as slaves. The master is however checked by the slaves so that if the slaves are not interrogated by the master within a set time, it is concluded that the master has failed and one of the slaves takes over as master. Communication between processors could be either by shared RAM which is quicker and is recommended for 16 bit devices, or could be by a parallel or serial link. Since single chip devices contain their own RAM, it would be better to communicate via a parallel link.

Software communication and voting routines should not use up too much of the processing power of the devices or fault coverage and system throughput will be decreased. Software voting is prone to transient errors, but once debugged cannot fail permanently as is the case for hardware voters.

6.3 THE USE OF REDUNDANCY AND INCREASED COST

If a FMEA of a system shows that the failure rate of some components is unacceptably high, some form of redundancy must be included to mask the high failure rate of the individual components. To a first approximation, for a set of components which is completely protected by fault-tolerant circuitry, the failure rate of the protection circuitry must not exceed that of the non-redundant components that are to be protected. For example the failure rate of the voters in a TMR configuration should not exceed that of a single channel of the triplex arrangement.

A more accurate method of predicting the benefit likely to be gained from redundancy is the analysis of chapter four which includes the effect of maintenance. The software coverage is calculated according to the relative execution times of the different software routines, whilst the hardware coverage is calculated according to the ratio of protected to unprotected circuitry. The proposed repair or maintenance rate must be defined, then the MTTFIF can be calculated for the redundant configuration. If the MTTFIF is greater than unity, then the use of redundancy is justified. In cost sensitive applications, redundancy should be applied to those parts of the system for which the greatest MTTFIFs are obtained.

The increased cost of redundancy should be considered in relation to the total system cost. For a small controller it is likely that the majority of the cost is due to the cabinet, power supply, printed circuit board and assembly costs. If the equipment is to be used in a hazardous environment, the cost of intrinsic safety measures such as barriers is substantial. It is likely that redundancy will double the component cost of a system and will require a larger printed circuit board and enclosure, but when the total system cost is examined the increase will be less than double. Typical microprocessors cost under £5 nowadays which makes the component costs small in comparison with other costs.

A fault-tolerant system will greatly increase the design costs both of hardware and software. If complex voting and communication routines are performed in software, this will greatly increase the software design costs.

The design of a general purpose fault-tolerant controller which contains the necessary fault recovery software would allow the increased design costs to be spread over many units. Although a redundant controller may initially cost more, it should last longer and require less maintenance. The cost of integrated circuits is approximately constant because of ever increasing advances in technology, but the cost of maintenance which is labour intensive is ever increasing.

CHAPTER 7

BACKGROUND TO DESIGN OF GOVERNOR CONTROLLER

7.1 SPECIFICATION

The governor is of a hybrid nature, consisting of an electronic microprocessor based controller, which in turn controls a mechanical valve which controls the flow of gas through the governor. The two main constraints on the design are reliability and cost (a few hundred pounds). The governor should cost little more than its pneumatic equivalent and should be as reliable. Some increase in cost and apparent degradation of governor reliability can be compensated for by added features and more efficient gas network control. The electronics should consume as little power as possible because of the problem of battery back-up of the mains supply. Whilst the electronics to be described do not prohibit the use of battery back-up, a CMOS design would be better from this point of view and it should be possible to convert the design of chapter eight to CMOS at the expense of computational speed of the controller and increased failure rate. Whilst design considerations may require the use of CMOS, it should be appreciated that the failure rate of CMOS devices is likely to be higher, especially under harsh environmental conditions.

The controller is a single or double loop controller, having a typical instruction cycle of 2µS. Depending on the software, the controller will implement the following control algorithms :-

(i) Three term controller of outlet pressure.

(ii) Clock control - The outlet pressure is varied over a pre-programmed pressure versus time profile so that the governor is able to satisfy the changes in demand throughout the day.

(iii) Flow control - The flow through the governor is maintained constant except for high and low pressure overrides.

(iv) Demand Activated Governing (DAG) - The outlet pressure of the governor is varied according to the flow through the governor. Under conditions of low flow the outlet pressure is held at a minimum, whilst under conditions of high flow, the outlet pressure is increased to compensate for resistive pressure drops in the downstream pipework.

The control algorithm implemented on the Durham governor is (iv) DAG since this is the most efficient and suitable for a distribution governor.

7.2 MECHANICAL VALVE

There are considerable problems associated with interfacing an electronic controller to a mechanical valve. Ideally the valve would have no moving parts and would control the flow of gas smoothly and linearly from OFF to ON by simply applying an analogue voltage ranging from 0V to a few volts. This is obviously not possible, so some form of electromechanical control is required.

There are two types of mechanical valve :-

(i) Direct acting - The position of the valve is controlled solely by an electrical signal. Failure of the electrical signal causes the valve to fail fully OPEN or fully SHUT.

(ii) Pneumatic back-up - A conventional pneumatic valve is modified so that its set-point is adjusted by an electrical signal. Failure of the electrical signal causes the valve to fail to its maximum or minimum pressure and not the catastrophic OPEN or SHUT situation. The maximum and minimum are chosen so that the governor will still operate within safe limits, but under normal operating conditions the pressure can be varied between the two set-points, giving efficient control of the network.

It was decided to opt for a mechanical valve with pneumatic back-up.

British Gas have developed two valves of this type which the electronic controller is capable of driving. The first type uses a stepper motor to vary the compression of the main valve spring and hence the set-point. This is shown in figure(14). The stepper motor drives a gearbox which engages with a screwed nylon plug. Hence the plug may be moved up and down, changing the compression on the main spring. This design has several disadvantages :-

(i) Complex drive circuitry and high power consumption of the stepper motor.

(ii) The stepper motor must be contained within a large and expensive flame proof enclosure because of the requirement for intrinsic safety.

(iii) If the stepper motor or drive signals to it fail, then the valve will stick at some intermediate position.

The second type of mechanical valve used by British Gas is the one chosen for the experimental governor at Durham. This consists of a modified pneumatic valve controlled by two solenoid valves. By switching the solenoids in the required sequence, it is possible to raise the outlet pressure, hold it constant, or lower it. The valve is shown in figure(15). The valve is a modified Donkin 226 'K' pilot which has an extended lower body part containing a dual Bellofram sealed piston with a spring positioned between the topside of the piston and the underside of the pilot valve. The chamber above the upper Bellofram is connected to the pilot outlet pressure. Gas at higher pressure from the inlet to the valve is admitted or exhausted from the lower Bellofram chamber through two solenoid valves which are controlled by the microprocessor controller. The level of pressure in the lower chamber determines the position of the lower spring against the pilot valve, and therefore the amount that it deloads the pilot main spring. The volume between the two Bellofram is vented to atmosphere and serves as a breathing chamber.

A needle valve is positioned near the inlet/outlet port of the lower chamber to provide a means of adjusting the rate of change of loading pressure. The rate of change determines both the accuracy of the set-point and the stability of the pilot valve. The governor is arranged to. fail safe so that if the solenoids are de-energised due to a power failure, then the outlet pressure fails to the maximum pressure controlled by the top spring compression. This is arranged by having the inlet pressure solenoid normally closed and the outlet pressure solenoid normally open. The compression of the lower spring controls the minimum outlet pressure of the pilot valve. Thus the valve will fail safe to a minimum or maximum (normally maximum) pressure, should the solenoids fail and not fully OPEN or SHUT. The only dangerous failure mode is if both solenoids fail OPEN. This provides a bypass round the pilot valve, but flow should be held to a safe limit because the saturated flow through two solenoids is small compared with the flow through the pilot valve.

This design overcomes the problem of the stepper motor system and has several advantages :-

(i) The solenoids operate at low power, typically 0.25W each which would be ideal for a CMOS controller.

(ii) The solenoids are small, cheap and intrinsically safe.

(iii) Fault-tolerant drive for the solenoids is easier to implement.

(iv) If it is necessary to protect against solenoid failure, it is both cheap and easy to replace each solenoid with four in a series/parallel configuration.

The one disadvantage is that the solenoids have a typical life of about five million operations and therefore under typical operating conditions would need replacing every eighteen months.

The experimental rig at Durham consists of the electronic controller and the mechanical valve just described. Whilst this is sufficient to verify techniques of flow/pressure control, the small pilot valve cannot pass the volume of gas used by several hundred consumers as a typical governor station is required to do. To pass a greater volume of gas, the pilot valve could be used to control a large main valve. This technique is the pneumatic equivalent of a transistor Darlington pair, where a large current is controlled in the second transistor by a smaller current injected into the first transistor.

An estimate is needed for the failure rate of the mechanical valve because of the requirement that the mechanical and electrical parts of the governor have failure rates which are of the same order of magnitude. Such data is very difficult to obtain, but British Gas give two failure rates for small valves. Blake and Drew [26] give a failure rate of 0.185 f/yr. either OPEN or CLOSED for an "open at rest" regulator, whilst Drew [25] gives a failure rate of 0.19 f/yr for an IGA2000 regulator, although the report suggests that this figure is a bit pessimistic. So a failure rate of 0.2 f/yr seems a good estimate for a small regulator valve. This failure rate excludes the failure rate of the main governor valve, for which it is impossible to obtain data, but it would be reasonable to assume an equal failure rate to the pilot valve.

The failure rate of the solenoid valves is neglected since regular replacement should reduce their failure rate to a negligible value. If this proves to be a problem "in the field", then a redundant series/parallel configuration of solenoids should cure the problem.

7.3 NON FAULT-TOLERANT CONTROLLER

The electronic controller specification has been defined in the previous two sections. It must have a failure rate of approximately 0.4 f/yr. , interface to the solenoid controlled valve, and implement the DAG control algorithm. A first iteration design is now proposed which is then analysed by an FMEA (Fault Mode Effect Analysis) which highlights the areas of greatest failure rate.

The block diagram of the proposed controller is given in figure(16). Using the data of MIL-217D [37] and actual measured device temperatures, the following failure rates are obtained for the controller :-

Pressure transducer P1 (manufacturers failure rate)	10.4 f/million hrs.
Pressure transducer P2 (manufacturers failure rate)	10.4 f/million hrs.
A/D converter	1.93 f/million hrs.
Control circuitry	2.05 f/million hrs.
Microprocessor + address latch + buffers	460 f/million hrs.
4k static RAM	12.3 f/million hrs.
4k EPROM	3.0 f/million hrs.
Input/output circuitry	0.94 f/million hrs.
Watchdog timer + reset circuitry	0.63 f/million hrs.
TOTAL	502 f/million hrs.
	= 4 f/yr

This total failure rate is unacceptable and is too high by a factor of ten. The microprocessor is clearly the most unreliable part of the system, followed by the pressure transducers and the RAM. It is therefore necessary to redesign the controller using fault-tolerant techniques to mask the high failure rate of the components just mentioned.

7.4 METHODS OF INTRODUCING FAULT-TOLERANCE

A further design requirement was that the controller could be programmed by British Gas personnel not having detailed knowledge of fault-tolerant programming or hardware techniques, and that standard software should run on the system. For these reasons many of the fault-tolerant features of the controller are transparent to the user and the specialised fault recovery software is pre-programmed and called as subroutines.

Alternative fault-tolerant architectures are discussed in chapters five and six. It was decided to use three microprocessors in a two out of three majority voting configuration, sharing common memory, to overcome the high failure rate of a single microprocessor. The voting was to be performed in hardware which has the advantage that, at the output of the voters, the TMR configuration just "looks" like a single microprocessor as long as the three microprocessors run in exact synchronism. This allows standard software to be run on the system. The voters also act as, and replace, the buffers between the microprocessor and the system bus. Three alternative methods were considered for performing the majority vote :-

(i) TTL logic - The components needed for a single channel are shown in figure(5). To decrease the number of gates required, wire ORed outputs are used. The disadvantage with wire ORing is that the rise time of the output is slower than a device with "active" pull-up. Since it is necessary to vote on thirty channels, this would require one hundred and fifty gates contained in thirty eight TTL packages. The large number of integrated circuits required would be unreliable and bulky.

(ii) EPROM - A four channel voter could be contained in a 4k EPROM. The disadvantage is that the EPROM could not drive the system bus without further buffering and that the 450ns access time of the EPROM would make voting slow and require the microprocessor clock frequency to be reduced, thus reducing processor throughput.

(iii) FPLA (Field Programmable Logic Array) - The equivalent voting circuit to figure(5) is programmed into a logic array. A five channel voter can be contained in a FPLA, offering fast voting as well as three error flags. The system bus can be driven directly without further buffering.

The FPLA approach was chosen and is described in more detail later.

The three processors share common memory. To improve both the permanent and transient tolerance of faults, it was decided to protect the memory using a SEC/DED (Single bit error correction/double bit error detection) Hamming code, reference [77]. Large scale integrated circuits are commercially available to implement this type of protection on 16 bit wide data, references [59,79,80], however two per memory are required. The devices are expensive, complex, and of unknown though probably high failure rate. An 8 bit version of these integrated circuits is not available, so it was decided, in the interests of complexity, reliability, and cost, to use a simple encoding/decoding PROM to implement SEC/DED protection. This is described in more detail later.

In order to make the pressure transducers highly reliable, it was decided to use three cheap piezo devices in a majority voting configuration, instead of using one expensive transducer.

As a final protection against transient or software errors, it was decided to incorporate a system "watchdog", which should restore the controller to correct operation following a system crash.

The control "firmware" will be stored in two separate EPROM sets, each capable of executing the control algorithm. This provides a hardware spare as well as allowing a software "spare" to be incorporated at a later date. If the spare EPROM set contains the same algorithm, but coded differently, then if a software error is detected, switching over to the alternative software coding will restore correct operation as long as the same software error does not exist in both copies of "firmware".

The output drivers to the solenoid valves were designed to tolerate any single component failure because of the unproven reliability of the driver integrated circuit and the large currents being repeatedly switched.

CHAPTER 8

DESCRIPTION OF EXPERIMENTAL CONTROLLER

8.1 CONSTRUCTION

The controller consists of three double Eurocard "microbus" prototyping cards housed in a 6U high 19" Vero rack. The rack also houses the power supplies, twenty column printer, and pulse generator used for testing purposes. A photograph of the controller is given in figure(19).

The boards are wire-wrapped as is the system bus to the edge connectors. Each integrated circuit is decoupled by a 0.1uF capacitor. The top side of the prototyping boards consists of a ground plane which is connected to 0V at several points. This reduces noise on the power supply and gives a low impedance connection between the power supply and the integrated circuits.

The power supply, whose circuit diagram is given in appendix(1) has the following outputs :-

+5V @ 5A	overvoltage and short circuit protected
+12V @ 1A	short circuit protected
-12V @ 1A	short circuit protected
+28V @ 1A	short circuit protected

The mains supply to the power supply is filtered to reduce the effect of mainsborne transients.

The signals carried by the backplane bus are defined in table(19).

The layout of the circuit boards is given in appendix(2).

8.2 TMR MICROPROCESSOR BOARD

The block diagram of the microprocessor board is given in figure(21). Each block will be described in detail with reference to its circuit diagram. A two out of three majority vote is performed on three microprocessors running in synchronism, the voting being at bus level. The voters drive the system bus directly. A common self-synchronising clock is connected to the processors as well as synchronisation hardware and reset circuitry/watchdog timer. An RS232 interface is provided to communicate with a VDU or printer.

8.2.1 Microprocessor block

Three identical channels consisting of a microprocessor and associated buffers are connected as shown in figures(22,23,24). Considering channel 1, as shown in figure(22), an 8085 microprocessor, U9, has its eight least significant address lines A0-A7 demultiplexed by U8 which are then connected to the voters. Address lines A8-A15 are connected directly to the voters and several control lines as well as the SOD line. Bidirectional data buffers U12, U13, separate data into DATA OUT and DATA IN. The DATA IN connections to the three channels are connected in parallel and to the system bus. DATA OUT is connected to the voters. Pull-ups are connected to AD0-AD7 and the RD and WR lines, this is because these lines are tristated during certain instructions and the pull-ups will ensure that all three channels are pulled up to +5V to prevent voting errors due to the indeterminate nature of a tristated line. All inputs to the microprocessors with the exception of CLK, TEST, and HOLD are connected in parallel, whilst most outputs, as described, are voted upon.

On the circuit diagrams a number in brackets is given after the signal name, this identifies the channel from one to three and is used on both inputs and outputs. Where no bracketed number is given, the signal referred to is an output from the system voters or is a signal on the system bus.

8.2.2 Voters

The voting on ninety signals (thirty from each microprocessor) is performed in six identical FPLAs as shown in figure(25). Each FPLA votes upon fifteen signals and reduces them to five signals. The remaining three outputs from the FPLAs are used as error flags, all of which are active low so that error flags such as $\overline{1+2}$ and $\overline{2+3}$ can be wire ORed. The error flag $\overline{1+2}$ indicates an error is present on channel 1 or 2, whilst $\overline{2+3}$ indicates an error on channel 2 or 3. Thus by examining both error flags it is possible to determine which channel does not agree with the other two. The error flags V1-V6 indicate in which voter there is a voting error. Thus it is possible to determine which channel is in error as well as the nature of the error, address bus, data bus, or control bus.

Due to slight timing differences between the three processors as well as access time delay in the voters, the error flags do not give a constant logic '1' for no error and logic '0' for an error. Instead a ragged signal is produced which is either predominantly logic '1' or '0'. For this reason the error flags are filtered by a low pass filter consisting of a 1k resistor and a 330pF capacitor. The oscilloscope trace of figure(26) shows the unfiltered error flag at the top and the filtered flag below. It can be seen from the unfiltered signal that the rise time of the voters is considerably longer than the fall time (approx. 50nS). This is because open collector outputs are used which have active pull down, but passive pull-up via the 1k resistors connected to all of the outputs.

The calculation of the low pass filter values and the programming information for the FPLA is given in appendix(3).

8.2.3 Self synchronising clock

The clock input to the 8085 is divided by two internally before being used as the internal microprocessor clock. This internal clock signal is available at the CLK OUT pin. It was found that on power-up, the divide by two stage could power up in either state. Normally this would not matter, but for three processors which are desired to run in synchronism, this is serious. It was found that one processor could power-up half an internal clock cycle out of phase with the other two. This situation could only be remedied by switching off the power and trying again. The self synchronising clock circuit of figure(28) was developed to overcome this problem. The three CLK OUT signals are fed to a three to eight line decoder, U22, which controls the three AND gates, U23. The AND gates remove the clock signal to the errant processor, should it be out of synchronism with the other two. A conventional 4Mhz quartz controlled clock is generated by U20 and associated components. The three AND gates, U23, provide some degree of isolation between the clock inputs, should any of them stick at logic '1' or '0'.

8.2.4 Synchronisation hardware

Additional circuitry is required to synchronise the three processors and latch the voter error flags in the event of a voting error. This circuitry is shown in figure(29). The signals marked PORTxx(RD) or PORTxx(WR) refer to outputs from the port address decoders, where PORTxx(RD) is generated by a INxx instruction and PORTxx(WR) by an OUTxx instruction. The top half of the circuit latches the error flags and the bottom half is used to synchronise the three processors.

When a voting error occurs either or both error flags $\overline{1+2}$ and $\overline{2+3}$ go low, causing the output of U23d pin 11 to go low. Assuming that a PORT01(WR) has taken place prior to a voting error, the flip/flop U28a, U28b will have been reset. The flip/flop is set by a low on pin1 U28a and the falling edge on pin 6 of the flip/flop triggers the monostable U18b. After a short delay, governed by the 33pF/22k RC network (440ns), the monostable generates a rising edge which latches the error flags in U31, an octal latch. The delay is necessary so that the filtered error flags have time to settle before being latched. The calculation of the necessary delay and hence RC network is given in appendix(4). The latched error flags are read by a read to PORT01. The flip/flop U28 prevents further transitions on U23d pin11 (further voting errors) from triggering U18b and latching new error flags before the old ones have been read. A write to PORT01 must be executed to reset the flip/flop which re-arms the error flag latching mechanism. A two bit tristate buffer, U30 allows the instantaneous error flags $\overline{1+2}$ and $\overline{2+3}$ to be read as PORT02.

Synchronisation of the processors is achieved by a combination of hardware and software. It is essential to synchronise the three program counters. This is hardware controlled and can only be done by executing a simultaneous RESET or interrupt. A RESET performs a cold start and is therefore undesirable except for initial power-up of the system, when a RESET causes all three processors to be synchronised, so interrupt driven resynchronisation was chosen. The hardware interrupt process synchronises the program counter and the interrupt software completes the rest of the synchronisation. It is not sufficient to simply supply a common interrupt to all three processors because an interrupt is only recognised at the end of the current instruction. Two processors will be executing one instruction, whilst the third, errant processor, may be executing an instruction of different length. The processors will therefore recognise an interrupt at different times and synchronisation cannot be achieved. If several retries are made, then it is possible to execute a simultaneous interrupt, but one further problem exists. It was found that one processor could lock one clock cycle behind the other two. Considering the op-code fetch timing of figure(17), it is seen that data is valid and read at T3, but continues to be valid at T4. Hence one processor can lock one clock period behind and read an instruction at T4 which then becomes T3 for that processor. Hence the errant processor will execute the same op-codes as the other two, but one cycle behind and will read/write incorrect data in three clock cycle read/write operations. It is impossible to cure this situation by executing further interrupts and the errant processor will remain locked one clock cycle behind.

In order to overcome these synchronisation problems the RST/HOLD circuitry was developed. The RST6.5 interrupt is used to initiate processor resynchronisation, executing in twelve clock cycles. The RST6.5 interrupt is generated and held for ten clock cycles, just less time than the interrupt "instruction" takes to execute. This gives a ten clock cycle "window" during which time any of the three processors may recognise the interrupt. Then a HOLD pulse is generated for a further ten clock cycles. Activation of the HOLD pin on the microprocessor suspends all external activity, but allows internal processing to continue. In this way the three processors are given time to catch up with each other and removal of the HOLD pulse should cause them all to set off in synchronism. The timing of the RST/HOLD pulses is shown in figure(18). One processor may just be finishing an instruction and another may just be starting an eighteen clock cycle instruction, which leads to a worst case seventeen clock cycle difference between the two processors which is too wide for the ten clock cycle "window". In these circumstances it will be necessary to perform a retry. In practice it has been found that resynchronisation is achieved after a few retries at worst.

The RST/HOLD circuitry is shown at the bottom of figure(29). The clock is divided by ten in U27a to give a frequency of 200kHz which is also used by the A/D converter. The divided clock is fed to the two bit counter U27b which cycles through the following states, changing every ten clock cycles.

00	No action
01	RST6.5 interrupt
10	HOLD operation
11	Reset flip/flop U21

The two bit counter U27b is normally held cleared (output=00) by a high on pin 14. When a voting error causes U23d pin 11 to go low, this sets flip/flop U21 which is connected to the clear input of the two bit counter and the counter advances through the states given above. When the count reaches 11 this is detected by U21c which resets the flip/flop U21. The flip/flop formed by U28c, U28d is used to enable/disable the RST/HOLD circuitry. A low on U28c pin 11 holds the output of flip/flop U21 permanently high, disabling the RST/HOLD circuitry. The flip/flop is disabled by a write to PORT00 and is enabled by a write to PORT02.

The PORT07(WR) select is connected to flip/flop U21 so that it is set by a write to PORT07. Thus it is possible to trigger the RST/HOLD circuitry by a port write instruction as well as by detection of a voting error. This is useful for testing purposes and software initiated resynchronisation.

The port addresses for the resynchronisation hardware are summarised below :-

PORT	READ	WRITE
0	-	Disable RST/HOLD circuitry
1	Read latched error flags	Reset flags latch flip/flop
2	Read instantaneous error flags	Enable RST/HOLD circuitry
7	-	Trigger RST/HOLD circuitry

8.2.5 Reset circuitry and watchdog timers

As well as power-on reset (POR) it is necessary to reset the controller if the +5V logic supply dips below acceptable limits caused by a short interruption to the mains supply. It was found that most RC controlled POR networks will not detect a short interruption in the +5V supply, but this interruption was sufficient to cause the microprocessor to cease reliable operation and crash. Such a short interruption to the mains supply has been experienced during electrical storms when there is a long delay before auto re-closure of circuit breakers following a lightning strike to the supply cables.

Two watchdog timers are included to reset the system in the event of it crashing. Both watchdogs are monostables which are triggered by a write to PORT06. The instruction to retrigger the watchdogs (OUT06) occurs once only in the control software and is located in the main program loop. The monostables are retriggered at a rate greater than their pulse width so that a "constant" pulse is generated. However if the program crashes this will cause the retriggering operation to cease and the monostable will time out after a delay determined by its pulse width. A TRAP (non maskable) interrupt is generated when the first watchdog times out which generates a warm start. If the warm start fails, then the second watchdog will "time-out" after a further delay and a cold system RESET will result.

The reset circuitry and watchdogs are shown in figure(30). The level triggered reset circuitry uses a "long tailed pair" contained in U29 as a comparator which compares the reference voltage across two silicon diodes to the +5V logic supply. The +5V supply is divided by a potential divider before comparison. By suitable choice of the potential divider, the circuit is arranged to give a logic '0' if the power supply falls below 4.5V and will give a logic '1' otherwise. Power-on-reset is detected and manual reset is provided by a push button switch.

Monostable U18a forms the TRAP watchdog which is triggered by a write to PORT06. The pulse width is set by the RC network and in this case is set to approximately two seconds. When the watchdog times-out, the Q output goes from a logic '0' to '1', generating a TRAP interrupt.

Monostable U19a, which has a pulse width of four seconds, forms the RESET watchdog and is retriggered by a write to PORT06. When the watchdog times-out it triggers the monostable U19b, generating a RESET pulse of approximately one millisecond duration. The rising edge of the RESET pulse is used to retrigger monostable U19a in case the reset operation has not been successful. In this way the watchdog is retriggered by the RESET pulse as well as a write to PORT06, so that if the reset is not successful, then the watchdog will continue to generate RESET pulses at an interval governed by U19a. A divide by two flip/flop is connected to the output of the RESET watchdog which will "toggle" everytime the watchdog times-out. The output from this flip/flop is used to select one of two EPROM sets, both of which are capable of executing the control program. Thus in the event of a system crash a warm start is tried first, then the spare EPROM set is switched in as well as performing a cold reset.

8.2.6 RS232 interface

The serial input and output lines are connected to RS232 drivers as shown in figure(28) so that an RS232 device can communicate with the controller for debugging, error logging, or console input/output purposes.

8.3 MEMORY BOARD

The memory board block diagram is given in figure(31) and contains 4k of static RAM which is Hamming code protected, two sets of 4k EPROM, 4k "instant ROM" and address decoders for memory and ports. At bus level the board "looks" like a conventional memory board and many of the fault-tolerant features are transparent to the user, although error flags and additional self-test circuitry are included to make full use of the fault-tolerant features of the board.

The RAM is protected by a SEC/DED Hamming code. The necessary check bits are generated by a look-up table in the encoding ROM and stored alongside data in the RAM. When data is read from the RAM, the data and check bits are read out and decoded by a ROM, which corrects the data if necessary as well as flagging errors.

Eight-bit wide data requires five check bits. This would require an encoding ROM having eight address lines and thirteen data outputs. The decoding ROM would need thirteen address lines and thirteen data outputs if error flags are included. Both these ROM sizes are very large and not commercially available. Two 2k EPROMs could be used for encoding and two 8k EPROMs for decoding, which are commercially available. This approach would be no more reliable than the design adopted, but would reduce the storage required from sixteen bits wide to thirteen bits wide. The disadvantage of using EPROMs instead of ROMs is their slow access time (450ns) as opposed to 50ns for bipolar ROMs. This would add to the memory read/write access time and would require the microprocessor to be slowed down, thus reducing processor throughput. The only advantage of EPROMs is their lower power consumption than bipolar ROMs, which may be important in a CMOS design, and their slightly higher reliability.

In order to use small commercially available bipolar ROMs, it was decided to split the eight bit wide data into two identical four bit wide streams which has the added advantage that each stream will correct a single bit error and detect a double bit error. Thus if simultaneous errors occur in both streams, the protection circuitry will correct double bit errors and detect four bits in error. Each stream has its own Hamming code protection in the form of four check bits. Data is stored as four data bits and four check bits and is read out as eight bits, which are decoded to restore the four data bits as well as generating four error flags. The encoding ROM can be contained in a 32x8 device and the decoding ROM in a 256x8 device, both of which are small and easily available.

The ROM look-up tables are calculated as follows. The SEC/DED code consists of four data bits and four check bits organised as :-

D3 D2 D1 D0 C3 C2 C1 C0 ;where Dx = data bit
Cx = check bit

Data is stored and read out in this form. On read out the check bits are re-calculated from the data bits and these check bits are exclusive ORed with the check bits read out from the RAM. The result is four correction bits C3' C2' C1' C0'.

C3' = C3 (recalculated) ⊕ C3 (stored) etc.

These correction bits identify the position of an error if it is a single bit error or indicate a double bit error according to table(20).

Looking vertically up the columns and writing down a data bit where a '1' ocurrs in table(20), it can be seen that the check bits are calculated by:-

$$C0 = D0 \oplus D1 \oplus D2$$

$$C1 = D0 \oplus D1 \oplus D3$$

$$C2 = D0 \oplus D2 \oplus D3$$

$$C3 = D1 \oplus D2 \oplus D3$$

Appendix(5) shows the look-up table contained in the encoding ROM which is calculated according to the equations above.

8.3.1 Decoding

The decoding ROM is also a look-up table which corrects the data if necessary and generates four error flags S2 S1 S0 E, where S2 S1 S0 identify which RAM (out of eight) is in error and E signals the occurrence of a single bit error. If E is not set, but S0 is set, then this indicates a double bit error. This is summarised in table(21). The error flags S2 S1 S0 E are referred to as the four bit error syndrome.

A program was written in assembler to calculate the information to be programmed into the decoding ROM. The decoding ROM look-up table, is given in appendix(6).

8.3.2 Circuit description

The circuit diagram of the board is given in figure(32). The encoding ROMs U33, U34 convert the eight bit data to eight data bits and eight check bits which are then stored in the 4k X 16bit memory matrix which is shown in more detail in figure(33). It is important to use one bit wide RAMs so that the failure of a single device causes a single bit failure which can be corrected. The decoding ROMs U35, U36 regenerate the eight bit data, corrected if necessary, flagging any errors that occur. It is possible to connect a much larger memory matrix between the encoders/decoders which may be dynamic RAM if required. In fact commercially available sixteen bit memory boards would be ideal. The data from the decoding ROMs is connected to the system bus via the tristate buffer U38 which is enabled by a read request to the RAM. The enable is generated by the OR gate U39 which is connected to the RAM $\overline{CS}$ and the $\overline{RD}$ line, thus producing a $\overline{RAMRD}$ signal. The error flags are latched into U37 in the event of an error and can be read by a read to PORT00. Three OR gates U39, are connected to form a four input OR gate. The four inputs are connected to the decoding ROM error flags E and S0. Any one of these flags going high indicates an error hence the output of U39 pin 6 is '0' for no error and '1' for an error. This single error flag is sampled and latched in U40a on the rising edge of every $\overline{RAMRD}$. If a '0' is latched in, it will have no effect, but if a '1' will cause a '1' to be latched into U40b, causing the Q output to go high. This generates a RST5.5 interrupt as well as latching the error flags. It is left up to the software interrupt routine to interrogate the error flags and determine whether the error is a single or double bit error and which RAM device is in error. A read to PORT00 reads the error flags, as well as resetting the flip/flop U40b. Until the flip/flop is reset, no further error flags can be latched into U37.

8.3.3 Address decoding and "spare" EPROMs

Address decoder U41 decodes eight 8k pages which are then further divided into 2k pages by U42. The "instant ROM" and RAM are decoded by U42a, whilst U42b decodes the EPROM memory. The most significant address line connected to U42b, A12, is connected via the exclusive OR gate U43. This is so that when the ROM SELECT line is at '0' the A12 line is not inverted and when at '1' the address line is inverted. With A12 not inverted the address decoding is as shown with the first EPROM set located from 0000-0FFF and the second from 1000-1FFF. If the A12 line is inverted, then EPROM set two is selected instead of set one and vice-versa. The "spare" EPROM set located from 1000-1FFF can be read as normal memory, however it is identical to set one and programmed to run from 0000-0FFF .

so will not run in the range 1000-1FFF. If the two sets are swapped by the ROM SELECT line, then the "spare" will appear to occupy memory in the range 0000-0FFF and will run and the other set will appear in the range 1000-1FFF. Thus it is possible for self-testing purposes to read the "spare" EPROM set, and in the event of a failure, a separate set can be switched in whilst allowing interrogation of the old set to determine the cause of failure.

The "instant ROM" and EPROM circuitry is shown in figure(34). "Instant ROM" is essentially battery-backed RAM which can be written to and retains data when power is removed. Its pin connections make it pin compatible with EPROM. Two "instant ROMs" are installed U45, U46 located from 2000-2FFF. They are write-protected by the switch shown and are buffered by bidirectional buffers U51, U52. Four EPROMs are installed U47, U48, U49, U50, located from 0000-1FFF and are buffered by U53 which is enabled by the $\overline{\text{BUFFEN}}$ signal from the EPROM decoder U42b.

The input port address decoder is shown at the bottom of figure(34).

8.3.4 Memory matrix

The 4k X 16bit memory matrix is shown in more detail in figure(33). Memories U55-U70 are 4k X 1bit static RAMs and the $\overline{\text{CS}}$ line to each device is connected via the OR gates U71-U74. This is for testing purposes only and allows RAM devices to be switched out by preventing their $\overline{\text{CS}}$ from going low. Dual-in-line switches S1, S2 are used to switch out RAMs and pulses injected into the control inputs of the OR gates enable the RAMs to be transiently switched out.

8.4 INPUT/OUTPUT BOARD

The block diagram of the input/output board is given in figure(35). The board contains a battery-backed real time clock (RTC), eight bit input port, eight bit output port, eight channel A/D converter, eight bit solenoid driver, as well as the necessary power electronics to drive a stepper motor and solenoid valves. A separate printed circuit board (PCB) holds the six pressure transducers as well as their signal conditioning circuitry.

8.4.1 Real time clock

The real time clock U77 is shown in figure(36). This is buffered by the bidirectional buffer U78. On failure of the +5V supply, the clock switches over to standby operation and continues to run. A ni-cad battery, which is trickle charged under normal operation, provides the necessary back-up supply. Open collector AND gates U75 are used to buffer the $\overline{RD}$ and $\overline{WR}$ signals so that the RTC read and write inputs are isolated from the power supply when it is switched off and are pulled up to Vdd. This is necessary for correct standby operation. The write strobe to the RTC is connected via LK1 which write-protects the RTC when it is removed. This is so that a system crash will not corrupt the RTC. The RTC is decoded as sixteen ports from PORT10 to PORT1F and is programmed and read as described in appendix(7) which is an extract from the Radiospares data sheet.

8.4.2 Multiplexer and A/D converter

An eight bit analogue multiplexer, U79 is controlled by three address lines MUX0, MUX1, MUX2 and is connected to the A/D converter U80. Inputs to the multiplexer are low pass filtered with a cut-off frequency of about 170Hz. Inputs P0 to P6 are from the pressure transducers and S1, S2 monitor the voltage applied to the solenoid valves. The A/D converter is clocked by a 200kHz signal already available which is obtained by dividing the processor clock by ten. The two flip/flops, U81 synchronise the start conversion signal and the clock. A write to PORT03 starts the conversion and a read to PORT03 enables the tristate output of the converter, allowing the digital output to be read onto the system bus. The end of conversion flag is connected to the most significant bit of the eight bit input port

U82. The other seven inputs to the port are spare and are connected to a dil switch. The input port is read as PORT04.

The bottom of figure(36) shows the output port address decoder U83.

8.4.3 Stepper motor driver

The eight bit output port, U84 which is written to as PORT05 performs two functions. The three least significant bits are used to control the analogue multiplexer via the address lines MUX0 to MUX2, whilst the four most significant bits drive four Darlington power transistors. The transistors are sufficient to drive the four phases of a stepper motor with the common stepper motor connection in this case connected to +5V. Protection diodes are incorporated as shown. The output port is cleared when a RESET is executed.

8.4.4 Solenoid drive circuitry

The solenoid valve drive circuitry is shown at the bottom of figure(37). An octal latch and driver, U85 is connected via two resistor networks to the solenoid valves. The circuit is designed so that solenoids can still be switched on and off in the event of any single component failing. The aim of the design is to protect against failure either short circuit or open circuit of the solenoid driver transistors.

The solenoids, although rated at 28V, were found to pull-in at greater than 8.0V and to drop-out at less than 2.6V. These voltages define the ON and OFF limits in the event of component failure in the driver circuitry. The drive circuit is basically a potential divider with switched resistors forming the lower leg of the divider. By switching in different resistors R1 to R4 the voltage across R5 is varied. This voltage swing is connected across the solenoid and used to switch it ON and OFF. If all transistors are off, then the voltage across R5 is zero, however if one transistor fails short circuit, the voltage across R5 is equal to the minimum value. If all four transistors are now switched on, the voltage across R5 is increased. If one transistor fails open circuit, the voltage developed across R5 is reduced and equal to the maximum value. The aim of the potential divider is to ensure that with one transistor failed open or short circuit, the swing from maximum to minimum is sufficient to turn the solenoid ON and OFF. The zener diode is necessary to subtract a bias voltage from the voltage across R5 so that the swing across the solenoid is within its ON and OFF limits. Two zener diodes are used in parallel to protect against the zener failing open circuit. Any of the resistors, transistors, or zener diodes can fail open or short circuit and the driver will continue to switch the solenoid ON and OFF as long as it is a single component failure. The voltage applied to the bottom end of the solenoid is monitored by the A/D converter after being attenuated by the potential divider made up of the 22k/1.2k resistors, which attenuate by a factor of approximately twenty. By switching the divider transistors and monitoring the solenoid voltage, it is possible to diagnose component failure in the driver stage. The detailed calculation of values for the resistors, zener diodes and supply voltage is given in appendix(8).

8.4.5 Pressure transducers

Six "Foxboro" piezo pressure transducers type 1800 01 C1 00C 0 are mounted on a separate printed circuit board. Each transducer uses a quad operational amplifier connected as an instrumentation amplifier as well as providing the constant current excitation to the transducers. The six transducers and their amplifiers are identical except that there is only one band-gap reference for all six transducers amplifiers.

The circuit diagram for a typical channel is given in figure(38). A band-gap reference is connected to U86a which controls the constant current of 1.5mA through the transducer. The voltage produced across the transducer is amplified by the instrumentation amplifier U86c. U86d which has an adjustable gain of 100 to 300 set by the "span" preset potentiometer. A non-inverting amplifier, U86b, having a gain of ten, produces a voltage of 0 to 7.5V at its output. This voltage is derived from the band-gap reference and is therefore stable and can be varied by the "zero" preset potentiometer. This variable voltage is fed via a large resistance into the inverting input of the final operational amplifier, U86d and is used to zero the output. Resistor R6 is individually chosen to have a voltage drop of about 1.5V across it. This is so that slight variations in the 0 to 7.5V supply do not effect the zeroing very much. Resistors R1 to R5 are supplied with the transducer and are for temperature compensation. Resistors R1 to R6 are connected as shown to a header plug.

Each transducer and amplifier is adjusted to give a swing of 1.5V for a 0 to 15"w.g. pressure range, and a "zero" of 0.1V.

8.5 PNEUMATIC TEST RIG

The test governor was built to demonstrate the reliable control of gas flow and pressure, and to show that the governor will continue to function in the presence of errors. Test circuitry allows transient and permanent errors to be injected into the controller. A schematic diagram of the pneumatic governor is given in figure(39) as well as a photograph in figure(20).

Considering the schematic diagram of figure(39), dry compressed air at approximately 10psi enters the "J" governor which reduces the pressure to 20"w.g. which is approximately held constant at a safe inlet pressure for the main control valve. Gas now passes through the modified "K" pilot , as described in chapter(7), which is controlled by the two solenoid valves S1, S2 and is stabilised by the needle valve. It is appreciated that it is undesirable to place bends in the pipework close to regulators or pressure tappings, but in order to achieve a compact design, it was necessary to bend the rig round on itself. This is unimportant for demonstration purposes. A needle valve is used as an adjustable "orifice plate" which has pressure tappings either side of it. Transducers P1, P2, P3 measure the outlet pressure of the "K" pilot and P4, P5, P6 measure the outlet pressure of the governor. The pressure differential across the "orifice plate" is a measure of the flow through the system according to the equation :-

$\Delta P = kQ^2$ where k = orifice plate constant

Q = flow

ΔP = pressure differential

Manometers are connected to the pressure tappings to give a visual check on the operation of the system as well as being used to calibrate the pressure transducers. A variable downstream load is simulated by another needle valve and gas is finally exhausted to atmosphere through a silencer.

8.6 ESTIMATED RELIABILITY

Chapter three concludes that MIL 217D is pessimistic when estimating the reliability of integrated circuits, especially plastic. However so that the failure rate of the fault tolerant controller can be compared with the non fault tolerant controller of section 7.3, an order of merit comparison will be performed using MIL 217D data. It is likely that the failure rate estimated in this way will be pessimistic, so the failure rate calculation is repeated using the failure rate data sources recommended in chapter three.

When analysing the fault tolerant controller several assumptions were made :

(i) Failures are instantly repaired. If the MTTR is very much less than the MTTF, then this assumption will not produce too large an error. For systems where the MTTR is not insignificant, the analysis of chapter four should be used.

(ii) Three pressure transducers per channel are connected in a TMR configuration with the voting performed in software. For this reason the failure rate of the transducers may be neglected.

(iii) The three microprocessors are connected in a TMR configuration with the voting performed in FPLAs. The failure rate of the microprocessors is neglected and only the failure rate of the voters is considered.

(iv) A spare EPROM set is used therefore the failure rate of the EPROMS may be neglected.

(v) The RAM is protected against single bit failures, so the failure rate of the RAM chips is neglected and the failure rate of the RAM is calculated as the failure rate of the extra Hamming code circuitry.

(vi) The failure rate of the solenoids is neglected since frequent replacement should prevent wearout and reduce their failure rate to a low value.

(vii) The failure rate of the solenoid drivers is neglected since any component failure is not fatal and is logged.

(viii) An estimation of the failure rate of the software is not included in the controller failure rate.

Using MIL 217D and measured integrated circuit case temperatures, the following failure rates are obtained for the controller using plastic integrated circuits :

A/D convertor	1.93 f/million hours
control circuitry	2.05 f/million hours
voters	38.5 f/million hours
watchdog	0.63 f/million hours
Hamming code circuitry	14.5 f/million hours
output circuitry	0.94 f/million hours
TOTAL	58.5 f/million hours
	= 0.5 f/year

Comparing these figures with those of section 7.3, an improvement in the failure rate by nearly an order of magnitude is observed. It is seen that the Hamming code circuitry is no more reliable than a 4k block of unprotected memory. If more reliable encoding/decoding ROMs were used, or if the controller were operated at a lower temperature, then the Hamming code circuitry would be more reliable than a 4k block of memory. Although the use of Hamming code protected memory gives no decrease in failure rate when permanent failures are considered, there is considerable benefit to be gained from transient error protection. Castillo et al [78] report transient failure rates which are up to fifty times greater than permanent faults. The fault tolerant controller offers considerable protection against transient errors which is not possible in the non fault tolerant design.

8.6.1 Revised failure rate prediction

The failure rate of the fault tolerant controller is now predicted using the recommended failure rate prediction sources of chapter three.

Circuit block	Data source	Failure rate
A/D convertor	MIL 217D	1.93 f/M hrs.
control circuitry	CNET	1.83 f/M hrs.
voters etc.	CNET	18.0 f/M hrs.
watchdog	CNET	0.62 f/M hrs.
Hamming circuitry	CNET	7.27 f/M hrs.
output circuitry	CNET	0.78 f/M hrs.
address decoders etc.	CNET	5.69 f/M hrs.
resistors (100)	NCSR	0.70 f/M hrs.
capacitors (100)	MIL 217D	0.40 f/M hrs.
wire wrap connections	ICL field data	1.60 f/M hrs.
soldered joints (500)	MIL 217D	1.30 f/M hrs.
edge connectors (200 prs)	CNET	0.60 f/M hrs.
	TOTAL	40.7 f/M hrs.
		= 0.36 f/year

This failure rate does not include the failure rate of the power supply, but is likely to be pessimistic for the rest of the controller since all failures are assumed to be fatal and do not result in a degraded performance.

The failure rate of 0.36 f/year meets the required value as defined in section 7.3. It is therefore proposed that the controller is suitable for use in a hybrid pressure controller as described in chapter seven.

CHAPTER 9

GOVERNOR CONTROLLER SOFTWARE

As a first step towards writing reliable software, modular programming was used. The software consists of nineteen modules written in assembler which are linked together to form the complete software package which is about 2k bytes in length. The detailed software listings are given in appendix(10) and a detailed description of each module follows, together with any necessary flow charts.

9.1.1 MAIN

Module MAIN is located at address zero and is executed when a reset is performed. The module contains the jump vectors for interrupts and software restart instructions. The flow chart is given in figure(40). Interrupts are disabled at the start of the module so that the initialisation process is not interrupted. Before the RAM is used by data variables and the stack it is initialised to FF hex. i.e. all ones. This is because of the snake which will be described later. The module INITL is then called which initialises all data variables and registers. It is important to initialise all registers to the same value, since if this is not done, they will contain random data and PUSHing such a register onto the stack will result in a voting error if the three microprocessor registers contain different random data. The system reset is logged as well as the time and EPROM set in use. As described in section 8.3 the EPROM set in use is either the main or the spare set. Finally the RST/HOLD circuitry is enabled before entering the main control loop. With the interrupts disabled the RST/HOLD circuitry will have no effect other than to insert hold pulses, but this slows down the processors, so the RST/HOLD circuitry is disabled at the start of the module to speed up execution. The rest of the module consists of the main control loop. Firstly a "recovery block" is formed and stored in RAM which will be used to perform vectored recovery if the system crashes. The interrupts are enabled after the formation of the recovery block. This is important on the first pass through the control loop because some interrupt routines use vectored recovery, and if an interrupt and subsequent vectored recovery is executed before the formation and storage of a recovery block, then vectored recovery will not be possible and the system will "lock-up". Each pass through the loop calls the modules CNTRL, PRBUFF, and SLFTST (to be described later) and resets the watchdog timers.

9.1.2 RESYNC

A voting error between the three processors generates a RST6.5 interrupt which causes a jump to the RESYNC module. This module, whose flow chart is given in figure(41), performs the necessary software resynchronisation, controls the resynchronisation hardware, and finally logs the error. On entry to the module, the RST/HOLD circuitry is disabled to speed up execution. If the RST/HOLD circuitry is successful, then the program counters will be resynchronised.

All registers are first PUSHed onto the stack and then POPed off. In this way a two out of three majority vote is performed on all registers and finally all SOD output pins are reset to zero. The retry counter is now decremented and tested for zero. If the count is not zero, the current state of the error flags is read. If the error flags show no error, then a transient error is logged, but if the error still exists, then hardware is used to generate a RST6.5 interrupt signal and the processors enter a halt state whilst waiting for the interrupt to be recognised. If the retry counter reaches a count of zero, then all retries have been exhausted and a CPU failure is logged as well as switching out the RST6.5 interrupt and RST/HOLD circuitry. As well as logging the transient or permanent error, the channel in error, syndrome (voter flags V1-V6) and number of retries executed is logged. Finally the retry counter which is stored in RAM is reset as well as the syndrome latch flip/flop. Vectored recovery is used to return to the main control loop and is initiated by calling module BLOCK.

The program counters are resynchronised by the RST/HOLD circuitry and by executing a simultaneous interrupt. It is for this reason that retries are performed by re-enabling the interrupts and then generating a hardware interrupt pulse. Thus the retry mechanism is a mixture of hardware and software techniques and each retry will attempt to resynchronise the program counters which could not be achieved by software alone.

9.1.3 MERROR

A memory error generates a RST5.5 interrupt which causes a jump to the MERROR module, whose flow chart is given in figures(42,43). The module first of all saves the error syndrome for future logging and tests if it is possible to read and write to the RAM. If it is not possible, then subroutines can no longer be used because they use the stack which is contained in RAM. After a number of retries controlled by the retry counter a jump is made to SFAIL which logs the failure of the RAM and executes a soft failure which puts the system in a fail safe condition. If it is possible to read/write 00 to the RAM, then the ability to read/write FF is tested. Testing using both 00 and FF will reveal stuck-at faults. The corresponding syndromes for a 00 and a FF read/write are both saved and are now compared. If both syndromes are zero and the retry counter has not been exhausted, then unless the RST5.5 pin is stuck at '1', the error is logged as a transient. If the RST5.5 pin is stuck at '1', then a hardware failure is logged and the RST5.5 interrupt is switched out. If the retry count has been exhausted, then it is assumed that the error is permanent. The two syndromes are now examined for a double bit error. If a DBE does not exist, then if either syndrome is zero or if they are both equal, the error is a single bit failure which is logged. If this is not the case, then the extra hardware is in error and this is logged. All permanent errors are logged and then the RST5.5 interrupt is switched out. If either syndrome shows a DBE, then either there is a stuck-at DBE in the check bits or the hardware has failed. The DBE must be in the check bits and not the data bits since the ability to read/write to the RAM has already been confirmed. To test for a stuck-at fault in the check bits, a table of test data is written to the RAM and then read back. If there is no DBE in the data or check bits, then the hardware must be faulty.

After the type of error has been logged, the syndrome and time are logged and, in the case of a single bit error, a normal return from interrupt is executed. If the error was a DBE, then the return address held in RAM may have been corrupted and vectored recovery is executed instead.

9.1.4 CNTRL

This module contains the software which controls the process. The flow chart is given in figure(44). The pressures either side of the orifice plate are read, Pgov being upstream of Pout. The pressure differential across the orifice plate is calculated and the outlet pressure of the regulator valve, Pgov, is controlled according to the equation:

Pgov = Pset + 2Pdelta ;where Pset is an arbitary set point

The desired pressure Pgov and the measured value are then compared. If they are equal the solenoids are controlled to hold the pressure, otherwise it is determined whether the pressure is too high or too low. If the pressure error is greater than the defined deadband, the solenoids are controlled to either lower or raise the pressure as appropriate. Finally a delay is executed to prevent instability.

9.1.5 PRESSR

This module combines time redundancy with component redundancy and the flow chart is given in figure(45). Two pressure readings are required, Pgov and Pout, which are the pressures either side of the orifice plate. Pgov is upstream of Pout. A total of six pressure transducers are read, arranged as two groups of three. Firstly all six pressure transducers are read eight times and the average calculated and stored in a table. Time redundancy such as this will lessen the effect of an erroneous read or noise on the input. Then a majority vote is performed on each group of three transducers. The transducers are taken in pairs and the three different averages are calculated. Then each transducer reading is compared with the average of the other two in order to determine if one channel disagrees with its related pair. If a channel is found to be in error, the error and time is logged as long as the retry count has not been exhausted and the pressure reading returned by the module is equal to the average of the two "good" channels. If all three channels are in agreement, then the average of the three readings is calculated and returned by the module. Finally the pressure differential, Pdelta, across the orifice plate is calculated.

9.1.6 RBKGEN

A "recovery block" is generated. The contents of all registers are saved in a block of memory for later use.

9.1.7 BLOCK

This module is complementary to RBKGEN and is used to execute vectored recovery. The block of memory which contains the contents of all registers at some time in the past is reloaded into the registers and execution is restarted at the position the last recovery block was generated.

9.1.8 SLFTST

This module is used to test the solenoids and driver circuitry and could be expanded to self-test the whole of the system. The self-test routine is executed every ten minutes under control of the clock and any error in the solenoid circuitry is logged. together with the time. A solenoid error is defined as any single component failure which is not fatal because of the redundancy already described. The solenoids are only tested every ten minutes. since in the event of a permanent failure. this will cause the failure to be logged every ten minutes until repaired. Any more frequently would be a nuisance. This is an alternative technique to decrementing a retry counter every time an error is detected and ceasing to log the error once the retry count has been exhausted.

9.1.9 SFAIL

No RAM is used by this module since some soft failures are caused by total failure of the RAM. For this reason no subroutines are used. The module logs the type of soft failure and then puts the system into a fail-safe condition. which for the experimental controller means failing to the maximum pressure. After the fail-safe shut down. the processor is halted. The processor will be interrupted from its halt state by the watchdog timer and a system reset will be executed if possible. If not. the shut down procedure will be repeated.

9.1.10 WTRAP

A non-maskable TRAP interrupt causes a jump to this module. A TRAP interrupt is generated when the first watchdog times-out and vectored recovery is attempted. This module resets the printer. clears the output buffer in case it has been corrupted and then logs the error and time as well as the address at which the TRAP interrupt was called. A retry counter is decremented each time the module is called and on every fifth call. a full system reset is initiated.

This and the following modules log the address at which the error ocurred. This may be very useful in detecting software errors. If the error always occurs at the same address, then further debugging at that address is probably required to solve the problem.

9.1.11 SNAKE

This module is called by a software RST7 instruction which is conveniently FF. Data lines are pulled high and unused memory is filled with FF so that an erroneous jump to non-existent memory or memory initialised to FF will cause a RST7 to be executed. In this way erroneous jumps are detected and execution may be safely vectored back. The module clears the output buffer in case it has been corrupted and then logs the error, time, and address at which the RST7 was called. Finally vectored recovery is used to return to the main program.

9.1.12 DFAULT

This module may be called to log software errors and is called by a RST5 instruction. If self checking of software is incorporated at a later date, then a RST5 instruction will log the error and execute vectored recovery.. This module is not called by any others at present, and is used for demonstration purposes only.

9.1.13 INITL

This is called during a system reset and initialises all registers and variables which are used later.

9.1.14 COUT

A character is converted to serial format and sent to the printer at 1200 baud.

9.1.15 COUTBF

This module puts a character in the output buffer. Output is buffered so that the control process is not halted whilst printing messages and logging errors.

9.1.16 PRBUFF

Unless the output buffer is empty, a character is fetched and sent to the printer.

9.1.17 MSGE

A message whose address is pointed to by the HL register pair is put into the output buffer.

9.1.18 NMOUT

The contents of the A register is converted to two ASCII hexadecimal characters which are put in the output buffer.

9.1.19 TIMLOG

The date and time is read from the real time clock and put into the output buffer.

CHAPTER 10

TESTING OF FAULT-TOLERANT SYSTEMS

10.1 SYSTEM DEBUGGING

Two techniques are available for debugging microprocessor systems :

10.1.1 In-circuit emulation

The microprocessor is removed and replaced with the emulation pod. The emulator usually runs in real time and will therefore simulate any timing problems. The emulation is controlled by the host microcomputer development system. Software is developed on the development system, stored on disk, and then executed on the test system by means of the emulator. Breakpoints may be inserted into the program, the program flow traced, and the contents of registers examined. Emulation is probably easier than logic analysis, but does suffer from some disadvantages when testing multiprocessor and fault-tolerant systems. In-circuit emulators are available for multiprocessor systems, but are very expensive and are usually dedicated to only one processor. They cannot be used to inject faults into the system, other than RAM and register corruption, and interference testing and the monitoring of recovery may cause the emulator to crash, since this forms an important part of the system under test. In fact an emulation pod cannot simulate a microprocessor for the purposes of interference testing.

10.1.2 Logic analysis

A logic analyser is really a very fast data logger which will record, in memory, the binary states of many parallel signals in real time. The information captured in this way may be displayed at leisure on a CRT screen in a user friendly format such as a pseudo timing display, hexadecimal or binary list, or a disassembled listing of the code executed by the microprocessor. Typically such a logic analyser will capture 48 channels of information at rates up to 25MHz and will store 1k samples. Information is clocked into the logic analyser by an internal or external clock. The logic analyser must be triggered and the triggering position may be varied within the block of data captured. The triggering sequence can be either the single occurrence of a trigger word or a complex sequence of trigger words. Trigger words correspond to the 48 channels of input data as well as extra clock qualifiers.

The logic analyser is a purely passive monitoring device and does not interfere with the operation of the system under test. A logic analyser was used to debug the controller described in chapter eight. Since the voting is performed at bus level, the replacement of one microprocessor by an emulation pod would cause the pod to be out-voted by the other two microprocessors. For this reason logic analysis must be used, but also has several advantages. It does not interfere with the system under test and can monitor all types of fault recovery, including the effects of interference testing.

Since the logic analyser is purely passive, some method of loading software into the test system must be provided. In the system of chapter eight, a "monitor" was written to run on the test system which would control the down-line loading of code into the test system, display memory dumps, allow memory locations to be modified etc. Test software was stored in battery backed RAM which could be write protected.

The logic analyser timing display was found to be very useful when debugging hardware errors and testing the resynchronisation hardware.

Software can also be tested with the logic analyser. If the inputs to the system under test are varied over their complete range, it is possible to force the system to execute all possible paths via conditional jump instructions. The complex triggering capability of the logic analyser can be used to verify that all possible jump permutations have been executed.

10.2 FAULT INJECTION AND RECOVERY

In order to test the recovery from faults, it is necessary to provide the means for injecting faults into the system. Test connectors should be provided that allow faults to be injected into the microprocessors, RAM, and recovery hardware. Faults are injected by a pulse generator, a short pulse correponds to a transient fault whilst a long pulse corresponds to a permanent fault. The pulse generator may be triggered manually or by the logic analyser.

The exact position in software of fault injection may be controlled using the logic analyser. The logic analyser is programmed to trigger on a combination of address, data, and control signals corresponding to a unique position in the software. The trigger output from the logic analyser is used to trigger the pulse generator, and the logic analyser can then monitor the action of the recovery routine. In this way, the logic analyser will capture normal operation followed by fault injection and recovery. Thus the effectiveness of recovery routines can be tested for faults injected anywhere in the software.

Random faults caused by power supply transients may be simulated by interference testing. The interference simulator is connected in series with the mains connection to the power supply. Voltage dips and transient overvoltage spikes can be simulated, and recovery observed using the logic analyser. Care must be taken that the transients do not affect the correct operation of the logic analyser.

10.3 OPERATIONAL TESTING

Once the system has been debugged, it should be tested for all input conditions over its complete operating temperature range.

The system should then be burnt-in at elevated temperature for several hundred hours and then released for use.

All permanent and transient faults should be logged by the system, preferably by a printer or some other non-volatile means, which will be useful when diagnosing and repairing permanent faults. The logging of transient faults will provide a useful history of the incidence of transient faults in that particular environment, and may require preventative action to be taken such as additional screening or filtering of the power supply.

10.4 TESTING OF THE GOVERNOR CONTROLLER

10.4.1 Pressure control

The correct control of pressure was first of all tested according to the relation :

Pgov = Pset + 2Pdelta

where Pdelta = Pgov - Pout

Pgov = outlet pressure of regulator

Pout = pressure downstream of orifice plate

The system was found to be stable with the solenoid needle valve fully open. The needle valve which simulates the orifice plate was set to a suitable value and then the downstream load was varied by means of the second needle valve. A graph is plotted in figure(46) of the regulator outlet pressure against the pressure differential across the "orifice plate", Pdelta. The observed values of Pgov are compared with the theoretical result shown by the straight line. Agreement between theoretical and measured values is very close and since the system was also found to be stable, it is concluded that pressure is correctly and accurately controlled.

10.4.2 Fault injection

Faults were injected into the controller by a pulse generator which was either triggered manually by a push button, or by the logic analyser. The pulse generator had two open collector outputs which produced negative and positive pulses. Fault recovery was traced using the logic analyser as well as observing that pressure control was correct and that the correct error message was printed out. The complete set of error messages as produced by the printer is given in figure(47).

10.4.3 Voting errors

A long and a short pulse was injected into the RDY line of channel one microprocessor as shown in figure(28). This caused that channel to lose synchronisation and message 1 was printed. This message reports a transient error in channel 1 and one attempt (retry) was found to resynchronise the three processors. The logic analyser timing trace of figure(27) shows the timing of the resynchronisation hardware. Halfway through the fault injection pulse, the $\overline{1+2}$ error flag goes low and at the same time microprocessor 1 is seen to be out of synchronism with the other two as shown by the ALE pulses. The positive edge of the latch pulse, latches the error syndrome which shows that the error is in channel 1. The system clock at 2MHz is shown in the centre for comparison. A short time after flagging the error, the RST/HOLD sequence is generated. Firstly a RST6.5 pulse is held for ten clock cycles and then the HOLD pulse for ten clock cycles, after which the RST/HOLD circuit resets. The RST/HOLD sequence is seen to be successful since all ALE pulses are in synchronism afterwards. A long pulse caused error message 2 to be printed. Attempts to resynchronise after 255 retries were abandoned and a CPU failure of channel 1 is logged. In both cases vectored recovery restored correct operation of the system.

10.4.4 RAM errors

The pulse generator was connected to the RAM test circuitry of figure(33). A short pulse injected into a single RAM chip caused error message 3 to be printed. The pulse was short so a transient error was logged and the syndrome S=10 correctly identified the RAM chip into which the error had been injected.

A long pulse injected as above caused message 4 to be printed. The pulse was long which caused the retry count to be exhausted. The system therefore assumed that the RAM chip had failed permanently.

A long pulse was injected into the preset pin of U40b as shown in figure(32). This caused a RST5.5 memory error interrupt to be generated without an error actually existing. The error was correctly logged as a hardware error as shown by message 5.

A long pulse was injected into two RAM chips which stored check bits. Message 6 was printed which shows that there is a stuck-at double bit error in the check bits.

A long pulse was injected into two RAM chips which stored data bits. It was found that it was not possible to read and write to RAM so message 7 was printed as well as executing a soft failure. The regulator was found to fail safe to its maximum value.

10.4.5 Watchdogs

The controller was deliberately crashed and the first watchdog, the TRAP watchdog, was found to restore correct operation via vectored recovery. This error caused message 8 to be printed. The controller was deliberately crashed five times, and on the fifth occasion a full system reset was executed as logged by message 9, initiated by the second watchdog timer. The trap watchdog routine is programmed to give a full system reset after four attempts. A full system reset was also observed to occur at power-on-reset and after a short power supply interruption which was correctly detected by the undervoltage detector.

The system was crashed until a second full system reset was performed, but this time message 10 was printed. This differs from message 9 in that EPROM set 01 is logged instead of EPROM set 00. This confirms that the spare EPROM set was correctly switched in by the RESET watchdog.

10.4.6 Snake

A hardware RST7.5 was generated. This caused a jump to 003C which contains FF which is the code for a RST7 instruction. A RST7 was executed and message 11 records operation of the snake and vectored recovery. The address is given as 003D and not 003C because the program counter is incremented before being pushed onto the stack.

10.4.7 Solenoid failure

A solenoid driver transistor was shorted out, simulating a short circuit failure. The solenoid failure was correctly logged as shown by message 12. The failure was logged once only and was at 30 minutes past the hour which corresponds to a call of the self-test routine at ten minute intervals.

10.4.8 Pressure transducers

The plastic pipe feeding pressure transducer 3 was bent double, cutting off the transducer. Message 13 shows that the error was correctly logged as channel 3.

10.4.9 Interference testing

The switching on and off of equipment near the controller caused several transient errors to be logged. Interruptions to the power supply were detected by the undervoltage detector and caused a full system reset.

Transient spike testing was not performed since this was felt to be more a test of the power supply than the controller. The power supply is fitted with "crowbar" overvoltage protection which prevents spike testing since the crowbar would be triggered. Also any failure of the power supply could destroy many integrated circuits which would be time consuming and expensive to replace in the experimental controller. If it is required to perform spike testing on the controller, it is suggested that a duplicate controller is built which uses a professional grade power supply.

It is proposed that interruption testing is a more useful test since short interruptions to the mains supply have been observed during an electrical storm, caused by auto-reclosure of circuit breakers. Some microprocessor equipment has been found to fail interruption testing when the interruptions are very short (a few mains cycles).

DISCUSSION

The review of fault-tolerant controllers in chapter two has shown that most of the work elsewhere has been concerned with large computers and that much of the fault-tolerance is implemented in software, although many of the techniques used on large computers are applicable to small controllers. The low cost of small digital controllers makes the design of a reliable controller using redundancy techniques much more cost sensitive. It has been shown that built in testability can be included at little extra cost. The use of redundancy will increase hardware costs as well as significantly increasing design costs, but these costs should be offset by increased reliability and availability.

In order to compare designs, or to meet certain design criteria, it is necessary to predict the failure rate of components. The failure of components is assumed to fit the constant portion of the "bath-tub" curve. This assumes completion of the initial burn-in phase and that components do not wear out. The wearout of integrated circuits is rarely experienced and is normally caused by moisture corrosion when the device is operated in a humid environment. In order to accelerate the failure of integrated circuits for life-testing purposes or to extrapolate failure rate data to different temperatures, the Arrhenius acceleration equation is used. The failure rate of devices is shown to increase exponentially with temperature. Life-test data is analysed using the Chi-squared distribution, however care must be exercised when extrapolating life-test data. For random failures it is permissible to multiply the number of devices under test by the test duration. This should not be done for non-random failures and especially when failures are due to wearout. The failure rate of integrated circuits is shown to dominate the overall failure rate of digital controllers and a huge variation is observed in failure rate predictions for integrated circuits. This variation is partly due to the use of different activation energies and junction temperatures when calculating the increase in failure rate due to temperature. It is important to use a realistic activation energy and to use the typical power dissipation when calculating the junction temperature. The predictions of MIL-217C are shown to be much too high by about two orders of magnitude. The predictions of MIL-217D and the NCSR data bank are more reasonable, but it is proposed here that the CNET data is best when predicting the failure rates of industrial ground based equipment.

Research elsewhere[43,48] has demonstrated that CMOS is less reliable than TTL and should only be used when it is essential to use devices with low power consumption.

There is much published information suggesting that plastic encapsulated devices should be used with caution since they have a higher failure rate than ceramic devices. New plastics have improved their reliability, but ceramic encapsulation is still better under conditions of high humidity, since moisture ingression is prevented which can cause corrosion and wearout of the integrated circuit. There is considerable evidence from one manufacturer[39] that under favourable conditions of low temperature and humidity, there is no difference between the failure rates of plastic and ceramic devices. Since the gas governor controller will be used in an unprotected and possibly humid environment, it is recommended that ceramic devices are used throughout.

All equipment should undergo burn-in prior to release in order to expose weak components. It is recommended that integrated circuits are purchased according to BS9400 grade C and are burnt-in after installation in the equipment. After successful burn-in, the grade C devices may be considered equivalent to the higher grade B devices which are predicted to have a failure rate one tenth that of plastic devices.

All components should be derated as recommended here and the use of forced cooling should be considered because of the exponential increase in failure rate with temperature.

Transient error rates fifty times those of permanent failures have been reported due to environmental, pattern sensitivity, and alpha radiation effects. It is important to protect equipment against transient errors for which software fault-tolerance is ideal. The gas governor controller will tolerate most classes of transient fault.

The effect of maintenance and repair on the reliability of redundant systems has been shown to achieve large improvements in the reliability of equipment, but is dependent on having a high fault coverage.

When designing fault-tolerant systems, it is useful to adopt a structured approach with defined levels of fault recovery. In this way protection will be afforded against most classes of fault and the higher levels of fault-tolerance should trap errors which are not corrected at the lower levels. The use of FMECA and FMEA is useful for identifying the most unreliable parts of a system which require further design effort.

The choice of microprocessor is important when designing a controller and the size of the control task should be matched to the power of the microprocessor, since the failure rates of microprocessors increase with complexity. The choice of microprocessor may influence whether fault-tolerance is implemented in hardware or software. Most of the fault-tolerance in the governor controller is transparent and implemented in hardware, since this allows standard software to be run on the controller.

The electromechanical gas governor is an example of a highly reliable digital controller. The mechanical part of the governor is a standard piece of equipment developed by British Gas and has the advantage of being simple to control, cheap, intrinsically safe, and will fail safe to pneumatic control in the event of controller failure. The controller uses many of the techniques discussed in this Thesis. Difficulty was experienced in synchronising the three processors which was cured by the development of special circuitry. In order for the TMR system to be more reliable than a simplex system, the voters must be more reliable than a single microprocessor. This could not have been achieved if the voters were constructed from TTL, but was achieved by voting in FPLAs which are more reliable and compact than TTL voters. The voters were found to be fast enough to permit operation of the microprocessors at an internal speed of 3MHz. The voters were chosen to have open collector outputs since this allowed wire ORing of the outputs and gave greater flexibility during development. It would be better to use voters with "totem pole" outputs and to perform the required ORing of the outputs using external TTL gates. Pull-up resistors would no longer be required on the outputs and the rise time of the outputs would be improved due to the active pull-up. This should enable the voters to work with faster processors. The reliability of the TMR configuration could be further improved if lower power voters were made available. The currently used FPLAs are constructed in bipolar logic which dissipates considerable power, resulting in a high failure rate. The Hamming code protected memory offers no improvement in the failure rate for permanent failures. This is because the failure rate of the protection circuitry is equal to that of an unprotected 4k block of memory. An improvement in the failure rate could be achieved if lower power encoding and decoding ROMs were available, having lower failure rates. If the memory size were increased above 4k, then the permanent failure rate would be improved. The main benefit to be gained from protecting the memory is the tolerance of transient errors which have been shown to occur up to fifty times more frequently than permanent failures. The watchdog reset circuitry was found to be very effective in resetting the controller after it had crashed, and could be included at little extra cost on even the most cost-sensitive non-redundant controllers. The piezo pressure transducer amplifiers were found to drift and the printed circuit board was found to be very sensitive to surface moisture. The problem of moisture was cured by coating the underside of the board with varnish. The voting on the pressure transducers made this drift less critical, but it would be advisable to replace the amplifiers with purpose built transducer amplifiers. The writing and debugging of the controller software was made much easier by making most of the fault-tolerance transparent and implemented in hardware. The logic analyser was found to be a powerful tool when debugging and testing the controller. The logic analyser timing display was invaluable when debugging the hardware, as was the logic analyser disassembler when tracing software execution. It would have been impossible to develop this controller without the use of a powerful logic analyser.

The use of ICE and logic analysis in the testing of redundant systems is discussed and it is concluded that logic analysis is more suitable for the testing of TMR systems. Since the logic analyser does not interfere with the system under test, it is suitable for interference testing and the monitoring of certain classes of transient fault. It was necessary to include test circuitry in the controller so that the different types of fault recovery could be tested. A short pulse injected into the system was used to simulate a transient fault, whilst a long pulse was used to simulate a permanent failure. A useful test facility was constructed by using the logic analyser to trigger the pulse generator used for fault injection. In this way the logic analyser will monitor the operation of the controller before the fault, during the fault, and during fault recovery.

The logging of transient errors by the controller, during normal operation, will provide much useful information about the nature and frequency of transient errors as they affect the controller.

Further work is suggested in the following areas. The controller should be rebuilt on printed circuit boards and use professional grade power supplies of proven reliability. The controller should then undergo interference and environmental testing. Whilst the recovery software has been extensively developed and tested, the complete software package could be expanded to include more self-checking and exception handling. There is a requirement within British Gas for a similar fault-tolerant controller, having a much lower power consumption. If the increased failure rate of CMOS were acceptable, then it should be possible to convert the controller design presented here to CMOS.

LIST OF REFERENCES

1 CLULEY J.C. - Electronic Equipment Reliability - Macmillan 1974

2 FANTINI F. LOLLI M. MICHELETTI G - Microtau : A highly reliable control unit based on a commercially available microprocessor - Proc. Eurocon '82 Copenhagen - Pub. N.Holland p375-380

3 HARBERT F.C. - High integrity safety systems for offshore platforms (A triplicated microprocessor system for fire/gas and automatic shut-down operations) - Proc. Eurocon '82 Copenhagen - Pub. N.Holland p433-437

4 LIM B.C.B - A fail safe microprocessor controller for a life support system - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p451-455

5 BLACK C.J. SUNDBERG C.E. WALKER W.K.S - Development of a spacebourne memory with a single error and erasure correction scheme - Proc. 7th Conf. on Fault Tolerant Computing 1977 p50-55

6 TOSCHI E.A. WATANABE T - A semiconductor memory with fault detection, correction and logging - H.P.Journal p8-13

7 PLATTETER D.G - Transparent protection of untestable LSI microprocessors - Fault tolerant Computing Symposium Tokoyo Oct. 1980 p345-347

8 ANDERSON T. KERR R - Recovery blocks in action : A system supporting high reliability - International Conf. on Software Engineering, San Francisco, Oct. 1976

9 RANDELL B - System structure for software fault tolerance - IEEE Trans. on Software Engineering - Vol. SE1 No.2 June 1975 p220-232

10 CERU S, COPPADORO G, Morganti M - UDET 7116 : Common control for PCM telephone exchange : Diagnostic software design and availability evaluation - Proc. Ann. Int. Conf. F-T Computing, Toulouse France, June 1978 p16-23

11 GHANI N. HERON K. LEE P.A - A recovery cache for the PDP 11 - IEEE Trans. on Computers - Vol. C-29 No.6 June 1980 p546-549

12 MORALEE D - Super-reliability computer keeps space shuttle safe - Crosstalk Electronics and Power May 1981 p359-361

13 GELDERLOOS H.C. WILSON D.V - Redundancy management of shuttle flight control sensors - Proc. 1976 IEEE Conf. on Descision and Control p462-475

14 AHERN D.B. LAMONT G.B. PETERSON J.B - Microprocessor development for a digital flight control system voter/monitor - IEEE Proc National Aerospace Electronics Conf., Dayton Ohio May 1976 p454-462

15 DEETS D.A. SZALAI K.J - Design and flight experience with a digital fly by wire control system using Apollo guidance system hardware on a F8 aircraft - Integrity in Electronic Flight Control Systems. Pub. AGARD Neuilly sur Seine France April 1977 p21.1-21.30

16 FORSYTHE W. MARSHALL W.G - Self-checking multiprocessor module for train control applications - Electronics Letters. 11th June 1981 Vol.17 No.12 p408-410

17 FOOSE R - Module minimises repair time of process control systems - Electronics 2nd March 1978 p121-124

18 SWARZ R.S - Reliability and maintainability enhancements for the VAX-11/780 - Proc Annual Int Conf F-T Computing Toulouse France. June 1978 p24-28

19 CANEPA M. CLARK S. SIEWIOREK D - C.vmp : The architecture and implementation of a fault-tolerant multiprocessor - Proc. 7th Conf. on Fault Tolerant Computing 1977 p37-43

20 RYLAND H.A - Microprocessor systems for railway signalling applications - IEE Colloquium on Microprocessor Applications requiring High Integrity and Fault-tolerance 11th Oct.1982

21 DAVIES O.J. OBAC-RODA V - Aspects of fault-tolerant ring structures - IEE Colloquium on Microprocessor Applications requiring High Integrity and Fault-tolerance 11th Oct. 1982

22 HIGUCHI T. KAMEYAMA M - Design of dependent-failure-tolerant microcomputer system using triple-modular redundancy - IEEE Trans. on Computers Vol.29 1980 p202-205

* 23 MURPHY D.T - Demand Activated Governing - British Gas internal report - ERS Governor Seminar March 1979

* 24 PICKERING E.W. SPEARMAN C.A. TANNER M.W.G - Leakage control in the natural gas era - British Gas internal report ERS GC175 1970

* 25 DREW W.A - Review of equipment fault data collection exercise - British Gas internal report ERS R.954

* 26 BLAKE J.D. DREW W.A - Reliability models of pressure reduction stations - British Gas internal report ERS R.850

27 PASCOE W - 2107A/2107B N-channel silicon gate MOS 4k RAMs - Intel reliability report RR-7 1975

28 EUZENT B - Intel 2116 N-channel silicon gate 16k dynamic RAM - Intel reliability report RR-16 1977

29 ROSENBERG S - Intel 2716 16k UV erasable PROM - Intel reliability report RR-19 1979

30 PASCOE W - Polysilicon fuse bipolar PROMs - Intel reliability report RR-8 1975

31 PASCOE W - MOS static RAMs - Intel reliability report RR-9 1975

32 NICHOLS N - 8080/8080A microcomputer - Intel reliability report RR-10 1976

33 INTEL - Component Data Catalog 1980

34 INSPEC - Electronic reliability data : A guide to selected components - Pub. IEE London 1981

35 REYNOLDS F.H - Measuring and modelling integrated circuit failure rates - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p32-45

36 Military Standard Handbook MIL-HDBK-217C 1979 Dept.Defense USA

37 Military Standard Handbook MIL-HDBK-217D 1982 Dept.Defense USA

38 CNET - Recueil de Donnees de Fiabilite Edition 1980

* These are British Gas confidential reports and are not normally available to other organisations except in special circumstances.

39 MOTOROLA - 1982 Microprocessor Family Reliability - Reliability report 8238 Sept. 1982

40 MOTOROLA - 1982 Memory and micro Reliability - Reliability report No.83/N001

41 MOTOROLA - MC68000G microprocessor - Reliability report 8243 Oct. 1982

42 MOTOROLA - Microcomputer components 1979

43 PETERSON P.W - The performance of plastic encapsulated CMOS microcircuits in a humid environment - IEEE Trans. on Components, Hybrids and Manufacturing Technology Vol.CHMT-2 No.4 1979 p422-427

44 FEDERICI F, MAMMUCARI F, TURCONI G - Influence of plastic encapsulated IC on TLC equipment reliability performance - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p259-264

45 KLEIN M.R - Microcircuit device reliability : Memory/LSI data - RADC report MDR-13 1979

46 DANIELS B.K, HUMPHREYS M - How do electronic system failure rate predictions compare with field experience - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p935-944

47 CLARIDGE A.N - A comparison between predicted and observed reliability in a large instrumentation and protection system - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p421-426

48 JOHNSON G.M, STITCH M - Microcircuit accelerated testing reveals life limiting failure modes - 15th Annual Proc. Reliability Physics Las Vegas USA April 1977 p179-195

49 O'CONNOR P.D.T - Practical Reliability Engineering - Pub. Heyden 1981

50 HAKIM E.B, REICH B - Can plastic semiconductor devices and micro-circuits be used in military equipment - Proc. 1974 Annual Reliability and Maintainability Conf. Los Angeles USA Jan 1974 p396-402

51 HAKIM E.B, REICH B - Environmental factors governing field reliability of plastic transistors and integrated circuits - Proc. 1972 Reliability Physics Symposium

52 PECK D.S, ZIERDT C - Temperature-humidity acceleration of metal electrolysis failure in semiconductor devices - Proc. 1973 Reliability Physics Symposium

53 LAWSON R.W - The qualification approval of plastic encapsulated components for use in moist environments - Proc. on Plastic Encapsulated Devices RSRE May 1976

54 LYCOUDES N - The reliability of plastic microcircuits in moist environments - Solid State Technology Oct. 1978 p53-62

55 SMITH D.J - Reliability and Maintainability in Perspective - Pub. Macmillan

56 JENSEN F, PETERSEN N.E - Burn in : An engineering approach to the design and analysis of burn-in procedures - Pub. Wiley 1982

57 JENSEN F - Reliability Tutorial - Presented at Eurocon'82 Copenhagen

58 NATIONAL SEMICONDUCTORS - The Reliability Handbook Vol.1 2nd Edition 1982

59 NATIONAL SEMICONDUCTORS - DP8400 E C Expandable Error Checker and Corrector Oct.1981

60 ANDERSON T, LEE P.A - Fault Tolerance : Principles and Practice - Pub. Prentice Hall 1981

61 MOTOROLA - 64k Dynamic RAM Reliability Design Manual 1981

62 HAYES J.P - Testing memories for single-cell pattern-sensitive faults - IEEE Trans. on Computers Vol.C-29 No.3 1980 p249-254

63 REDDY S.M, SUK D.S - Test patterns for a class of pattern-sensitive faults in semiconductor random access memories - IEEE Trans. on Computers Vol.C-29 No.6 1980 p219-226

64 ABRAHAM J.A, THATTE S.M - Testing of semiconductor random access memories - Proc Annual Int. Conf. F-T Computing Los Angeles California June 1977 p81-87

65 BRODSKY M - Hardening RAMs against soft errors - Electronics April 24 1980 p117-122

66 BRAUER J.B. KAPFER V.C. TAMBURRINO A.L - Can plastic encapsulated microcircuits provide reliability with economy - Microelectronics (GB) Vol.1 1970 p5-24

67 FOX M.J - A comparison of the performance of plastic and ceramic encapsulations based on evaluation of CMOS integrated circuits - Microelectronics and Reliability Vol.16 1977 p251-254

68 KNIGHT L. LUCAS P - Observed failure rates of electronic components in computer systems - Microelectronics and Reliability Vol.15 1976 p239-243

69 DUMMER G.W.A - Electronic Components : Past Present and Future - Electronic Components 1970

70 GISSING J.G - BS9000 components and reliability quality factors : suggested use of MIL-HDBK-217C factors based on a comparative product assurance analysis - Microelectronics and Reliability Vol.21 1981 p683-697

71 DHILLON B.S. SINGH C - Engineering Reliability : New Techniques and Applications - Pub. Wiley 1981

72 ARNOLD T.F - The concept of coverage and its effect on the reliability model of a repairable system - IEEE Trans. on Computers Vol.C-22 No.3 1973 p251-254

73 PEARSON J.C. PREECE C - Hardware and software aspects of fault coverage in small digital controllers - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p663-668

74 HALSE R.G. PEARSON J.C. PREECE C - The introduction of fault tolerance into digitally controlled gas regulators - Proc. 1983 International Gas Research Conference London

75 HUNGER A - Characterisation test of microprocessors using a new approach in self-testing - Proc. Eurocon'82 Copenhagen - Pub.N.Holland p906-910

76 DANIELS S.F. FASANG P.P - Microbit I : A microcomputer with built in diagnostics - Proc. Eurocon'82 Copenhagen - Pub. N.Holland p625-629

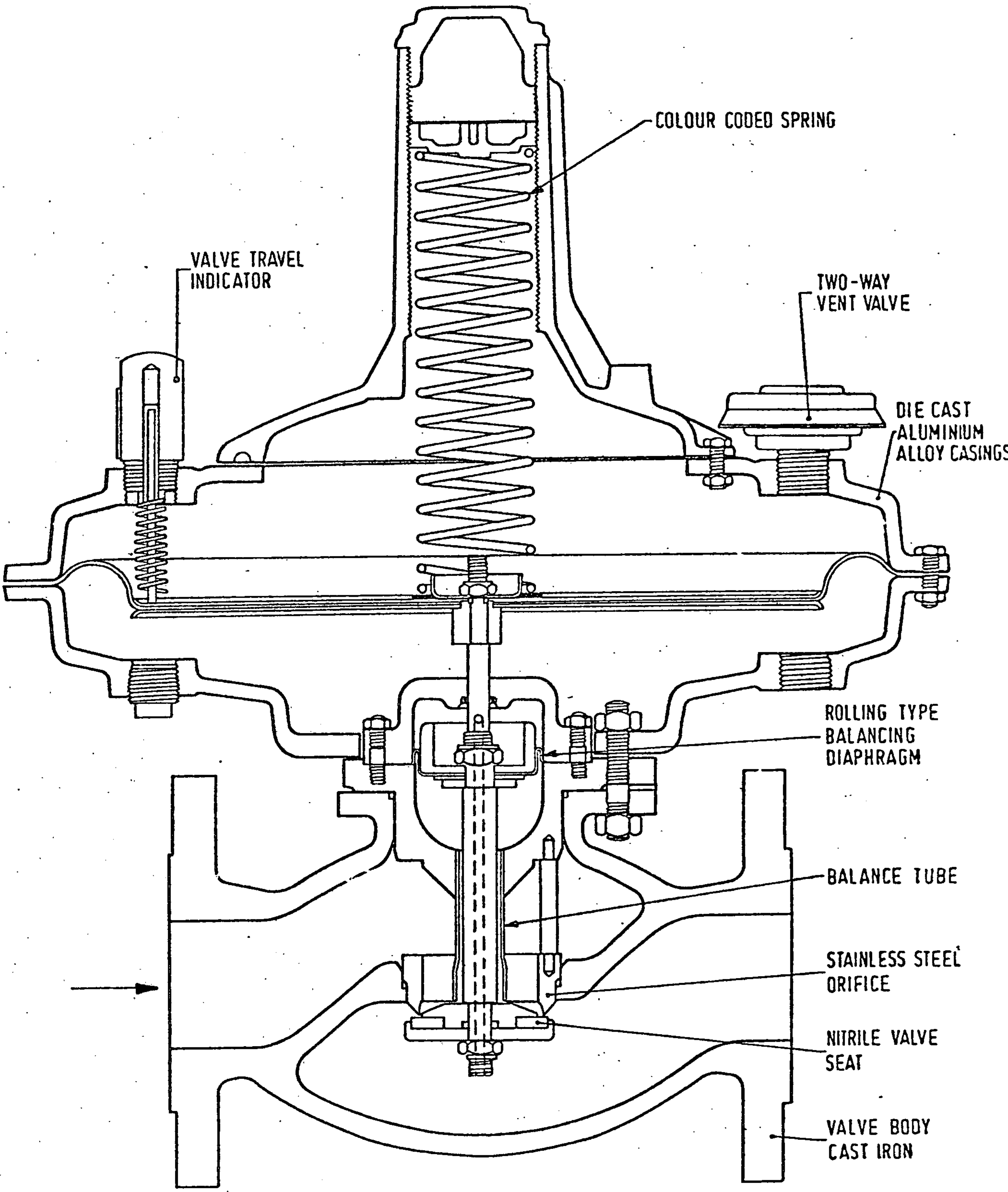
77 HAMMING R.W - Error detecting and error correcting codes - Bell Systems Technical Journal Vol.29 1960 p147-160

78 CASTILLO X. McCONNEL S.R. SIEWIOREK D.P - Derivation and calibration of a transient error reliability model - IEEE Trans. on Computers Vol.C-31 No.7 1982 p658-671

79 TEXAS INSTRUMENTS - Error detection and correction using SN54/74LS360 or SN54/74LS361 - Bulletin CA-201

80 ADVANCED MICRO DEVICES - Am2960 Cascadable 16 bit error detection and correction unit

81 PREDICTOR - Computer software package marketed by Management Sciences Inc. - 6022 Constitution Ave. Albuquerque N.M. 87110



CROSS SECTION OF A DONKIN FIG.280 REGULATOR

FIGURE 1

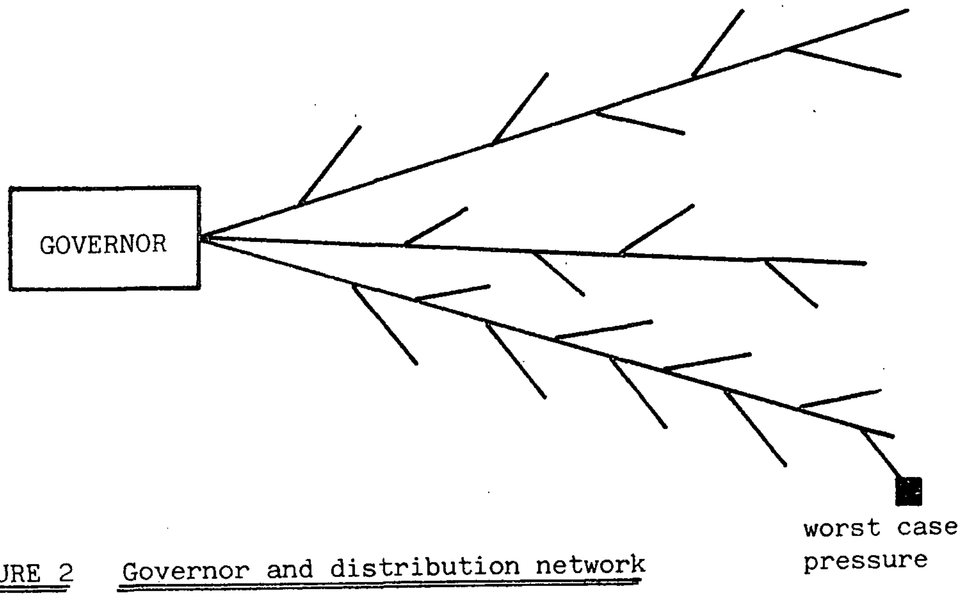


FIGURE 2 Governor and distribution network

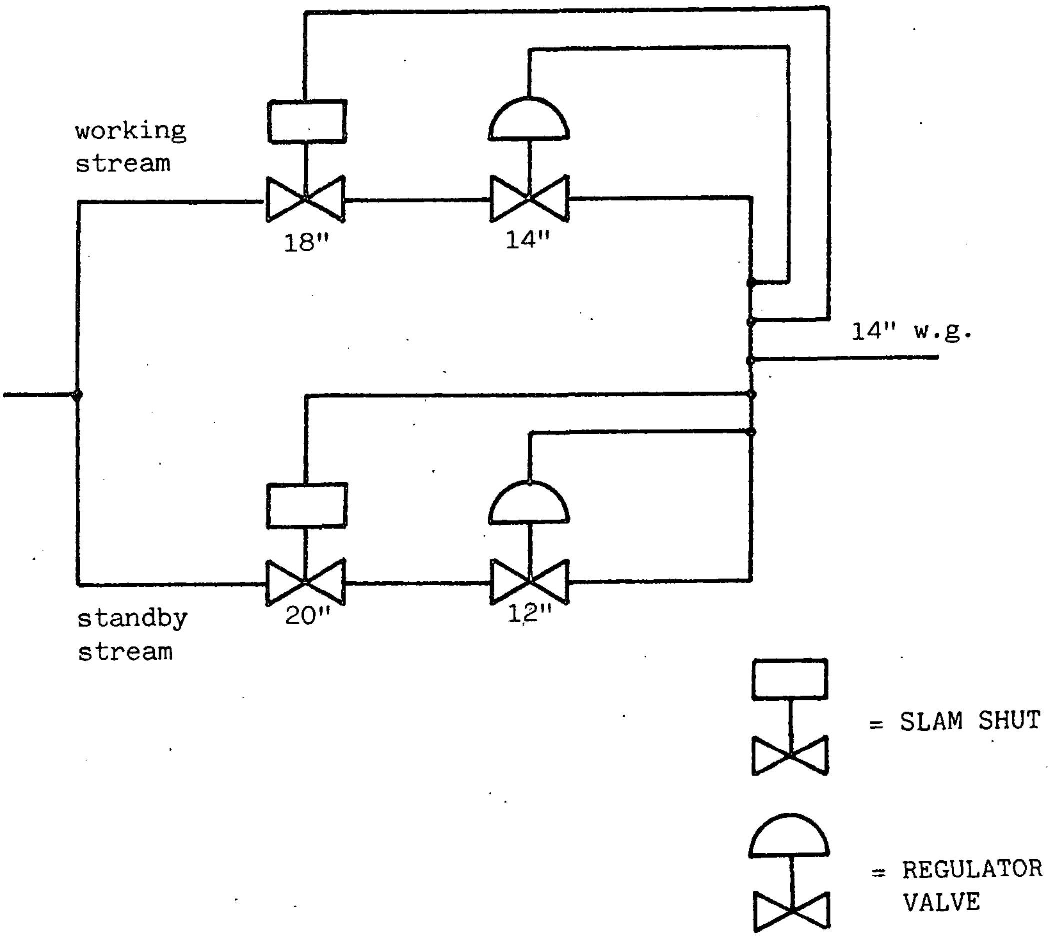


FIGURE 3 Twin stream Governor

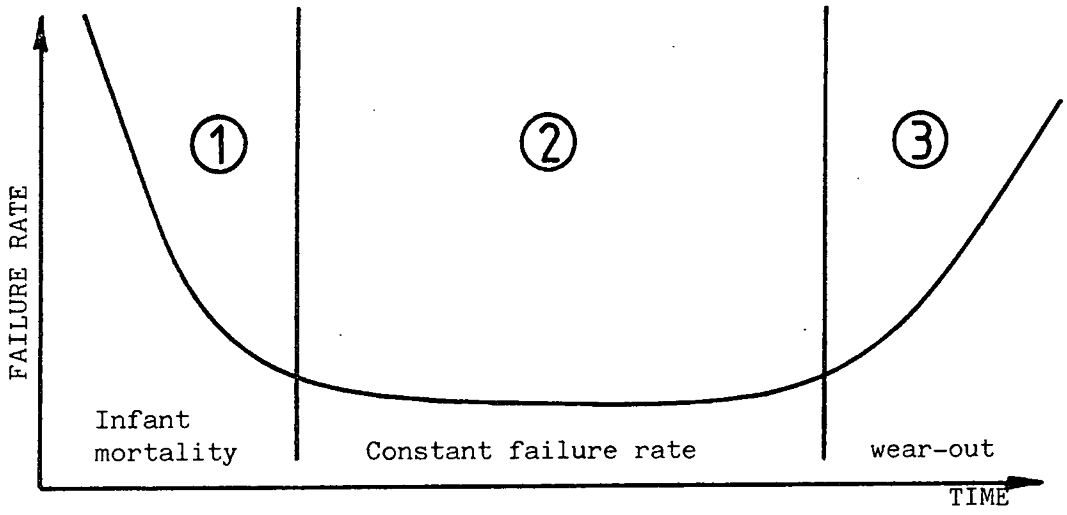


FIGURE 4 The bath-tub curve

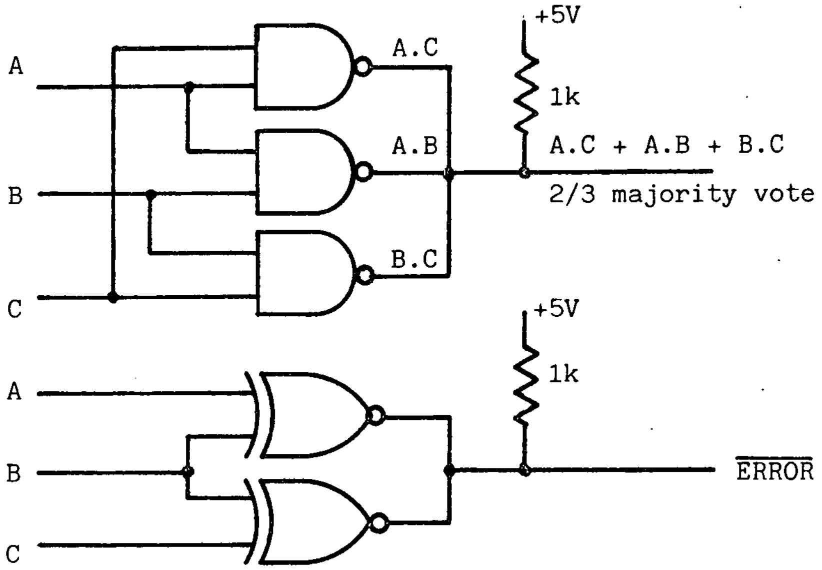


FIGURE 5 TMR voter and erro detection in open collector TTL logic

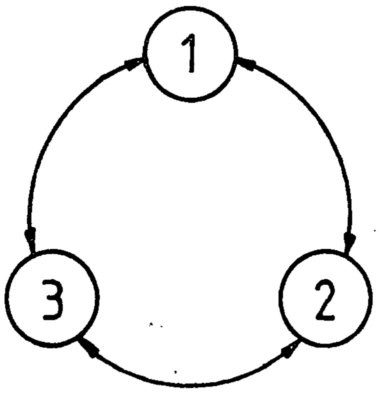


FIGURE 6 TMR Ring structure

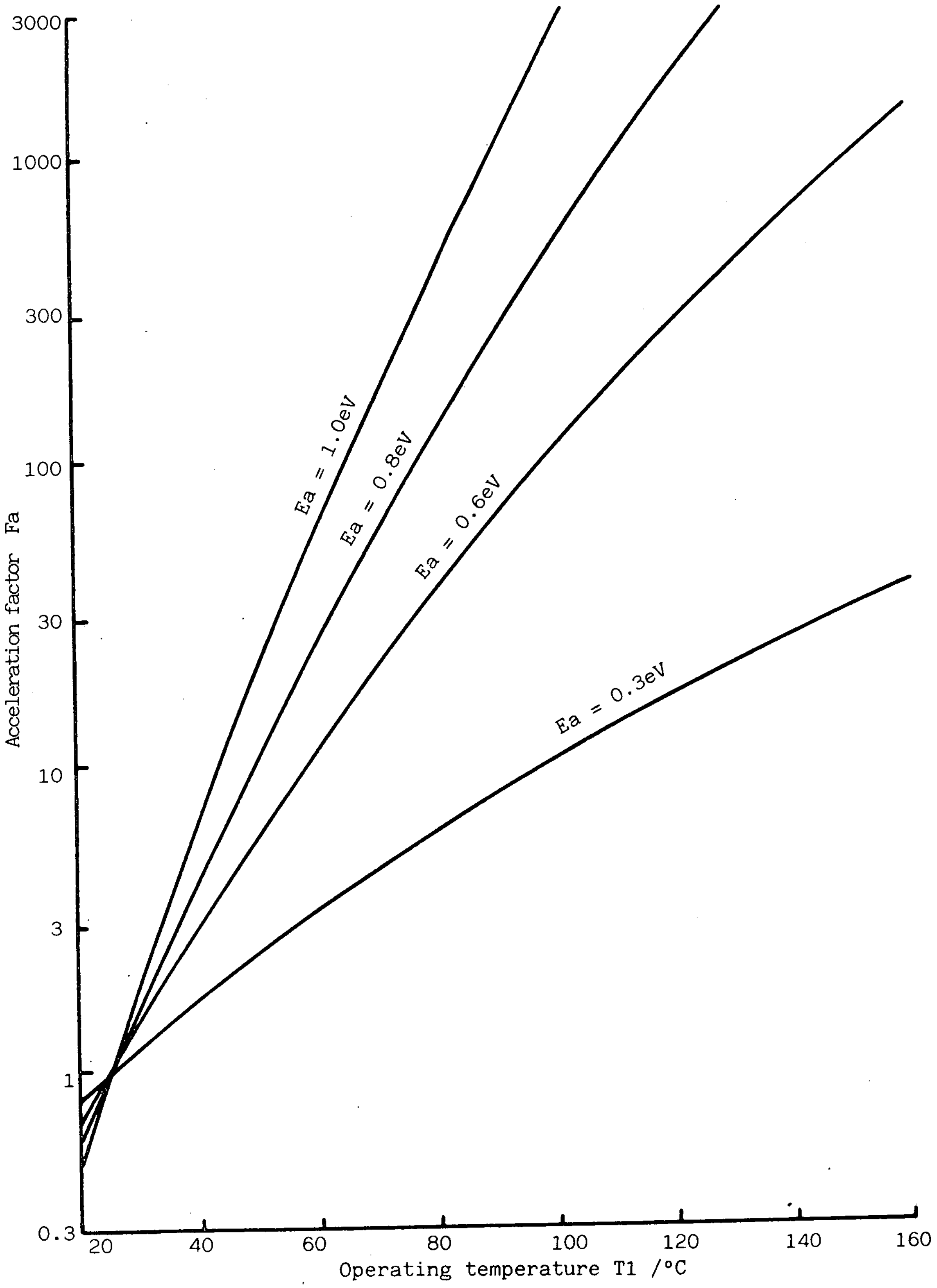


FIGURE 7 Graph of acceleration factor vs operating temperature for a reference temperature of 25°C. Plotted for different activation energies.

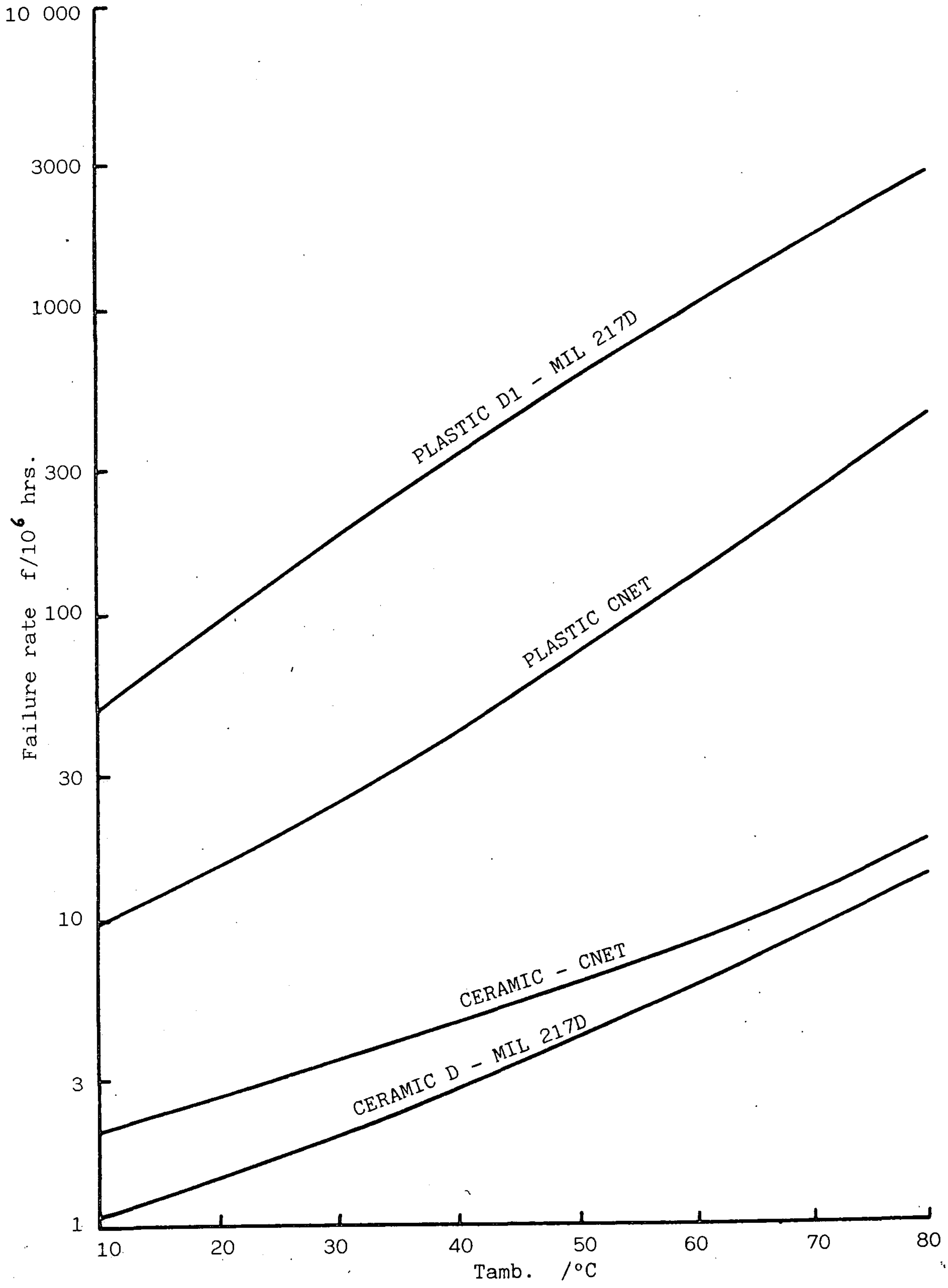


FIGURE 8 Failure rate of a 8085 Microprocessor vs case ambient temperature

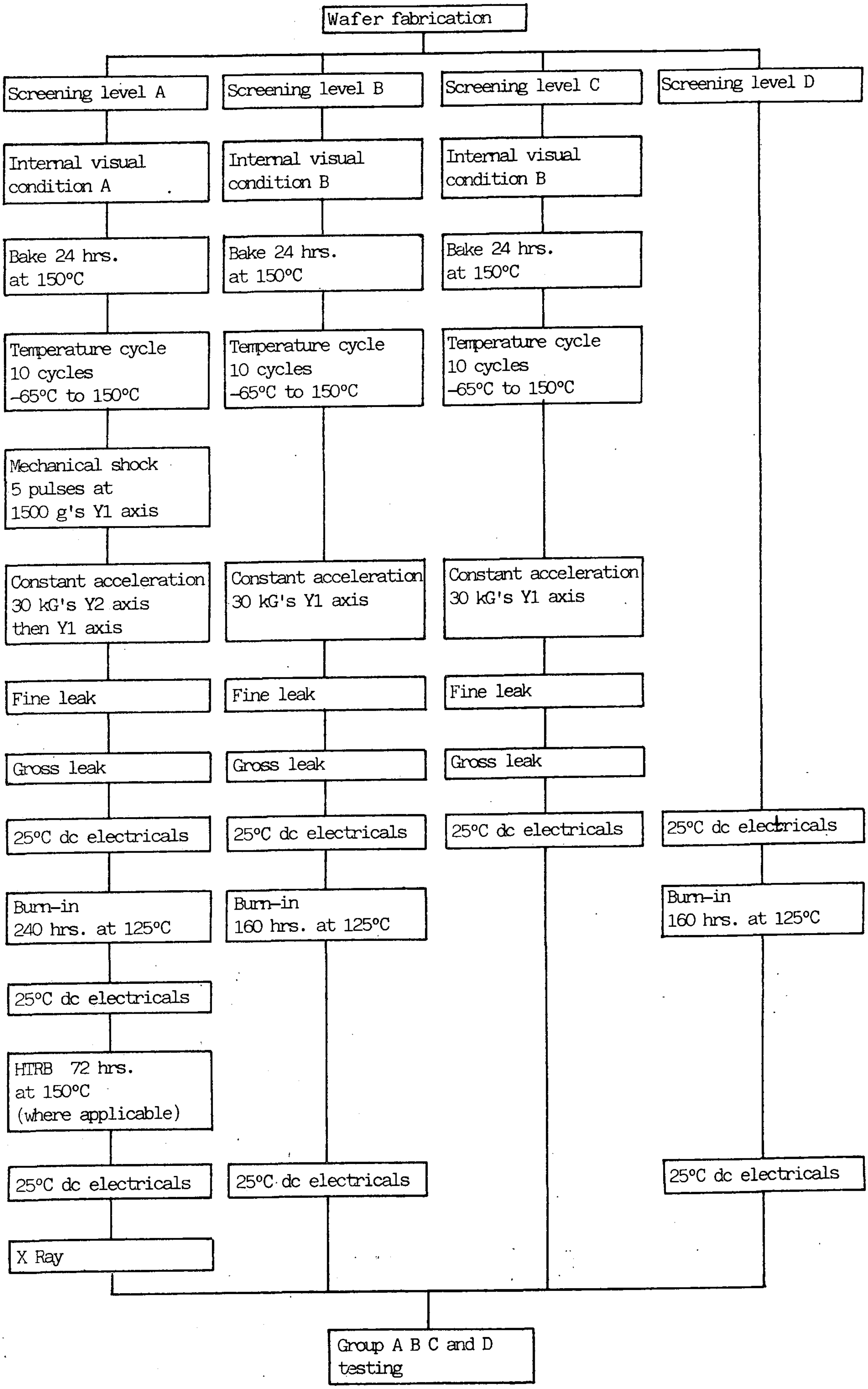

Wafer fabrication
Screening level A
Screening level B
Screening level C
Screening level D
Internal visual condition A
Internal visual condition B
Internal visual condition B
Bake 24 hrs. at 150°C
Bake 24 hrs. at 150°C
Bake 24 hrs. at 150°C
Temperature cycle 10 cycles -65°C to 150°C
Temperature cycle 10 cycles -65°C to 150°C
Temperature cycle 10 cycles -65°C to 150°C
Mechanical shock 5 pulses at 1500 g's Y1 axis
Constant acceleration 30 kG's Y2 axis then Y1 axis
Constant acceleration 30 kG's Y1 axis
Constant acceleration 30 kG's Y1 axis
Fine leak
Fine leak
Fine leak
Gross leak
Gross leak
Gross leak
25°C dc electricals
25°C dc electricals
25°C dc electricals
25°C dc electricals
Burn-in 240 hrs. at 125°C
Burn-in 160 hrs. at 125°C
Burn-in 160 hrs. at 125°C
25°C dc electricals
HTRB 72 hrs. at 150°C (where applicable)
25°C dc electricals
25°C dc electricals
25°C dc electricals
X Ray
Group A B C and D testing

FIGURE 9 BS9400 Screening Procedure - see reference (58)

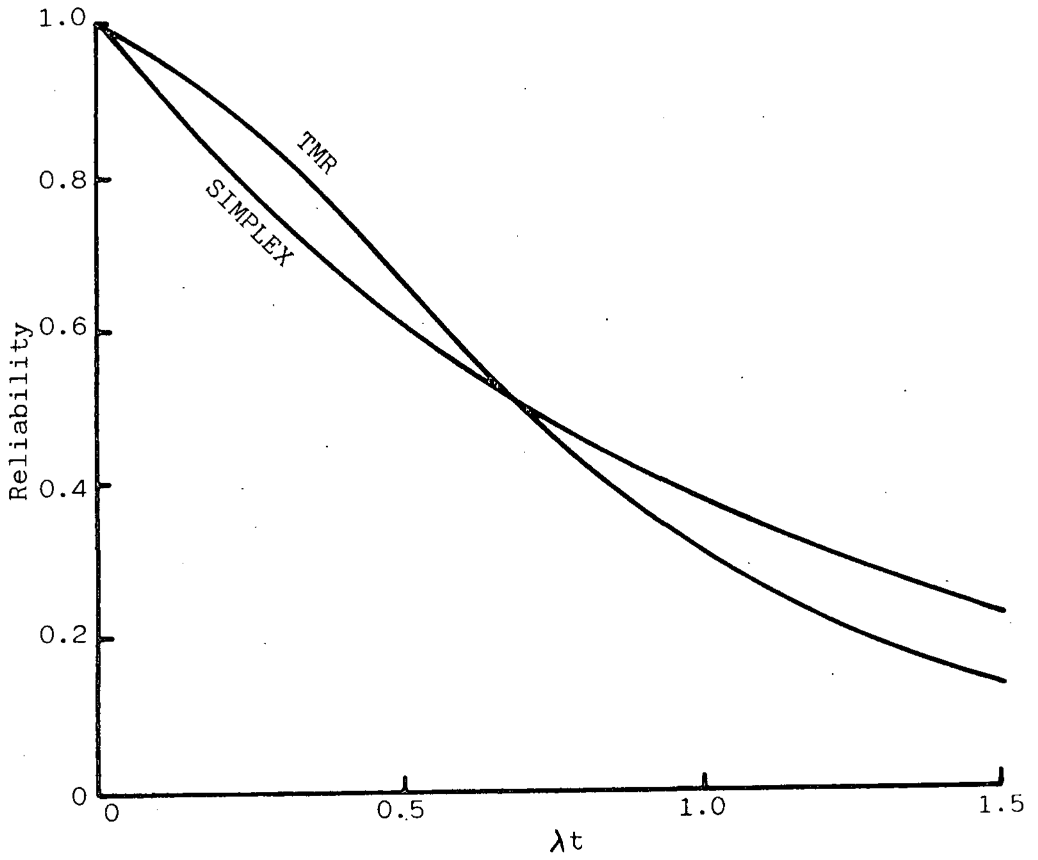


FIGURE 10 Graph of Simplex and TMR reliability vs normalised mission time

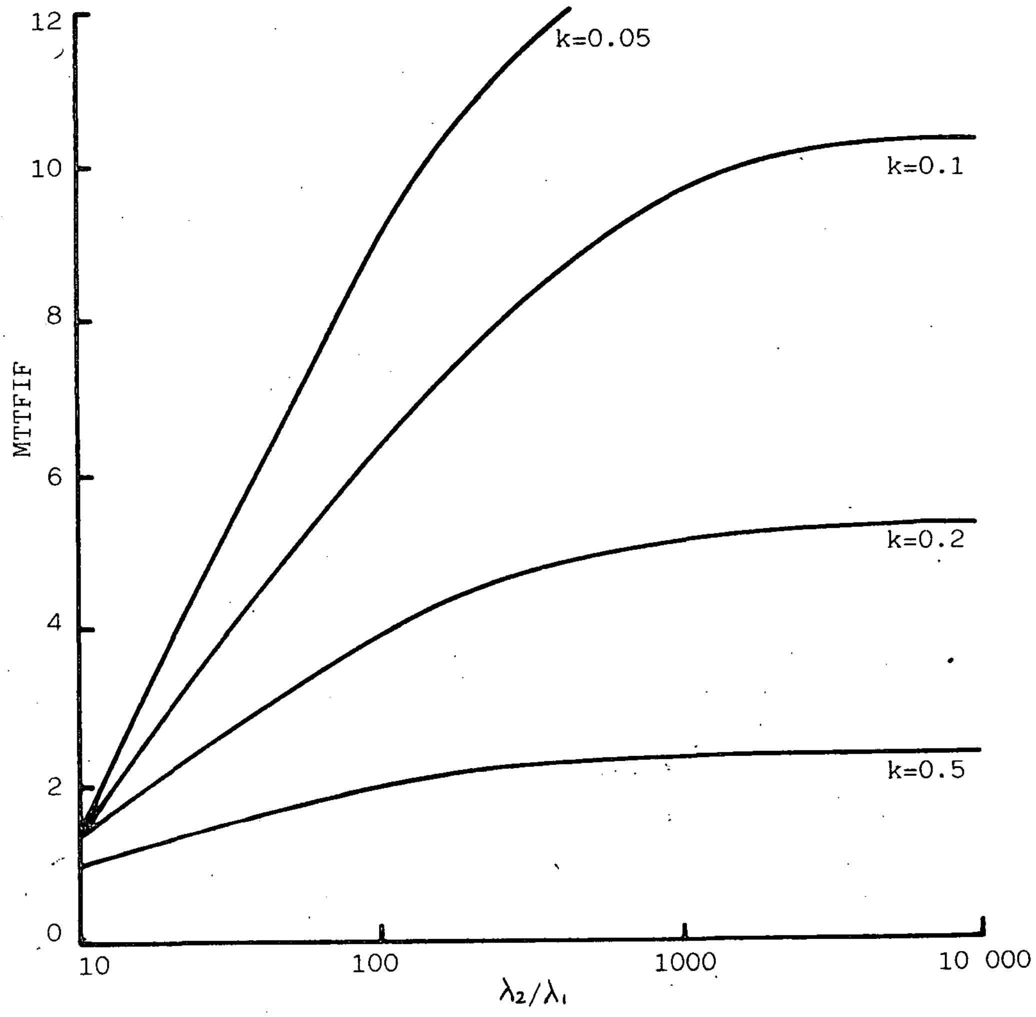


FIGURE 11 Graph of MTTFIF vs ratio λ_2/λ_1

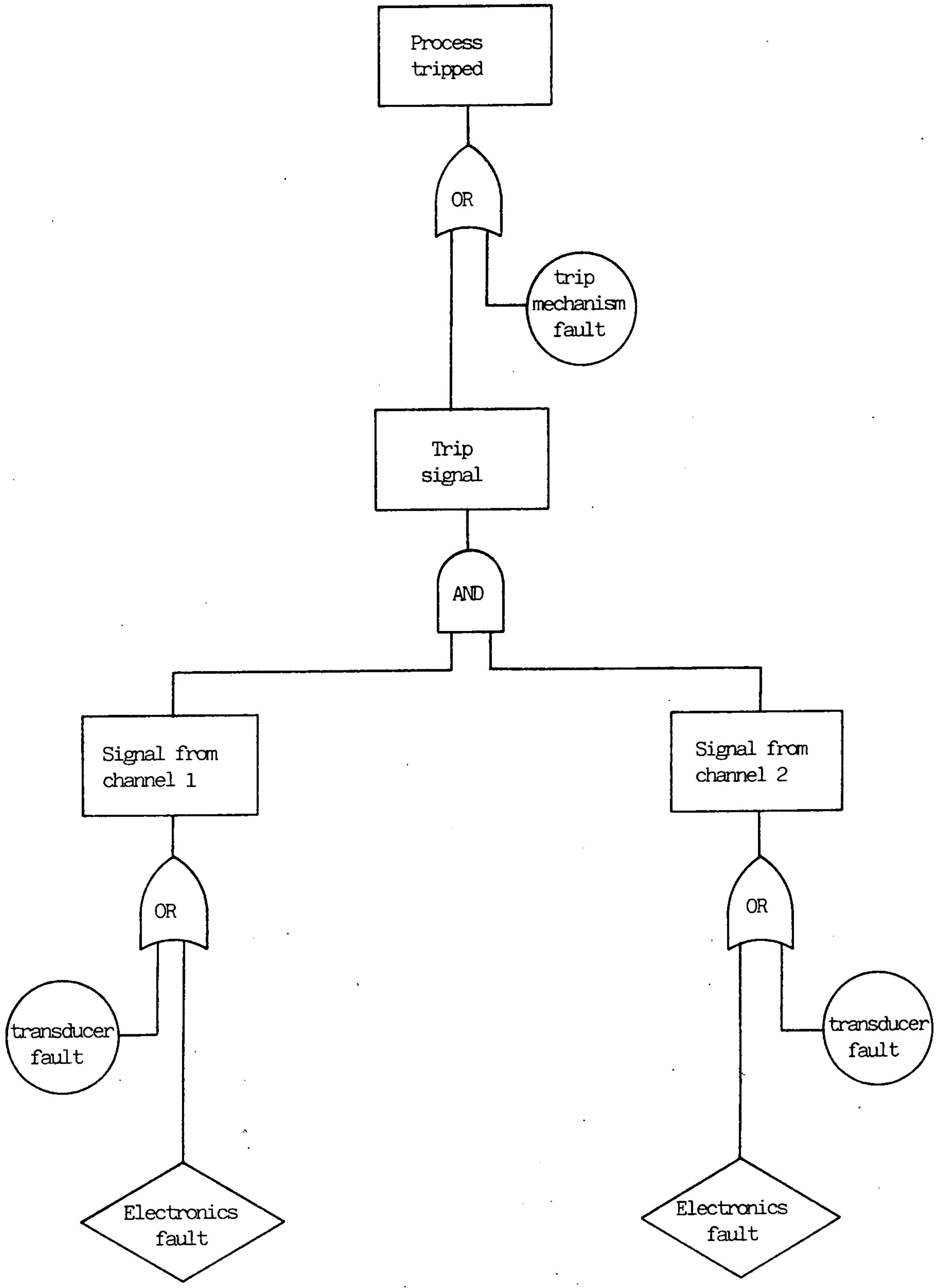
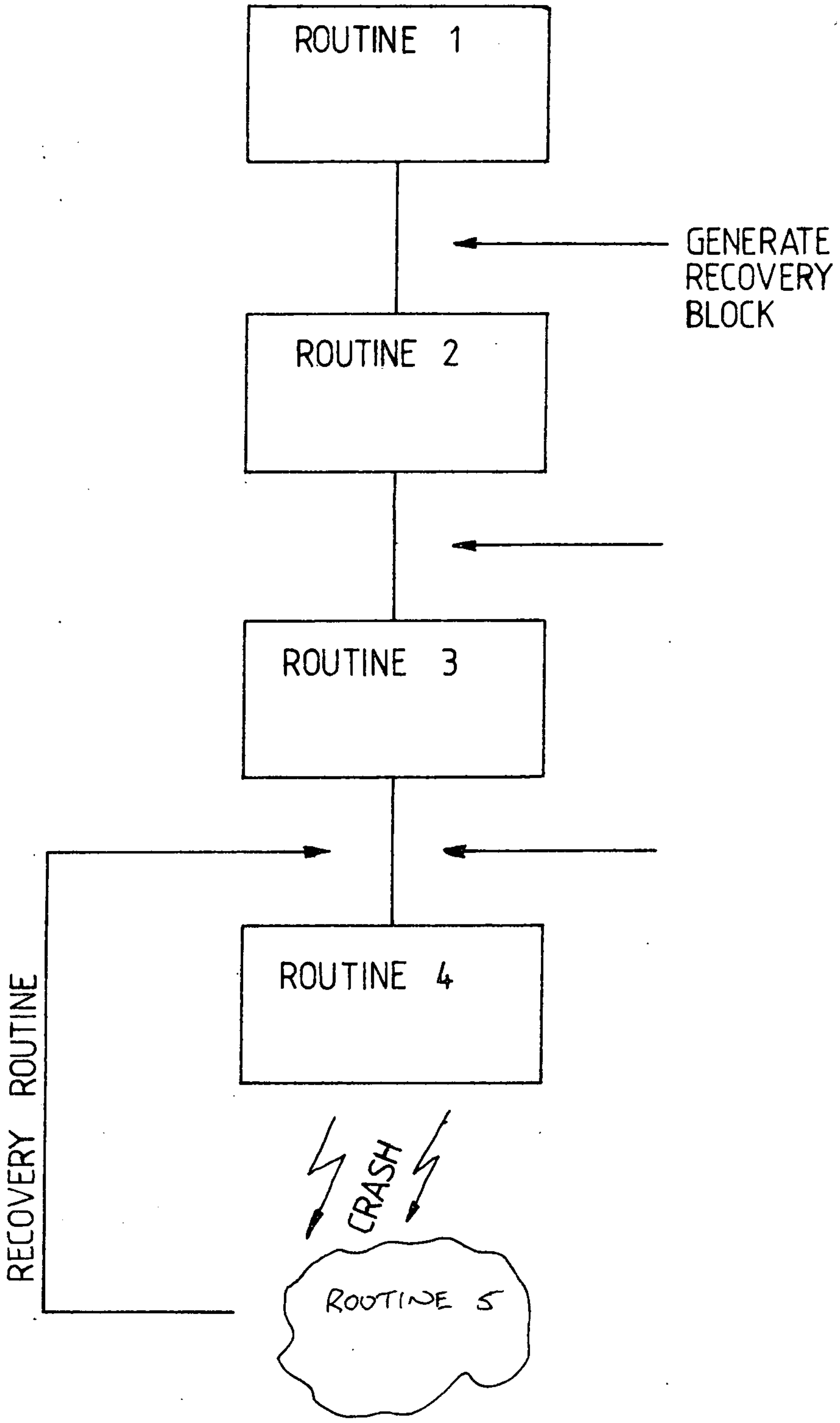


FIGURE 12 Fault tree analysis for pressure trip



SOFTWARE RECOVERY USING "RECOVERY BLOCKS"

Figure 13

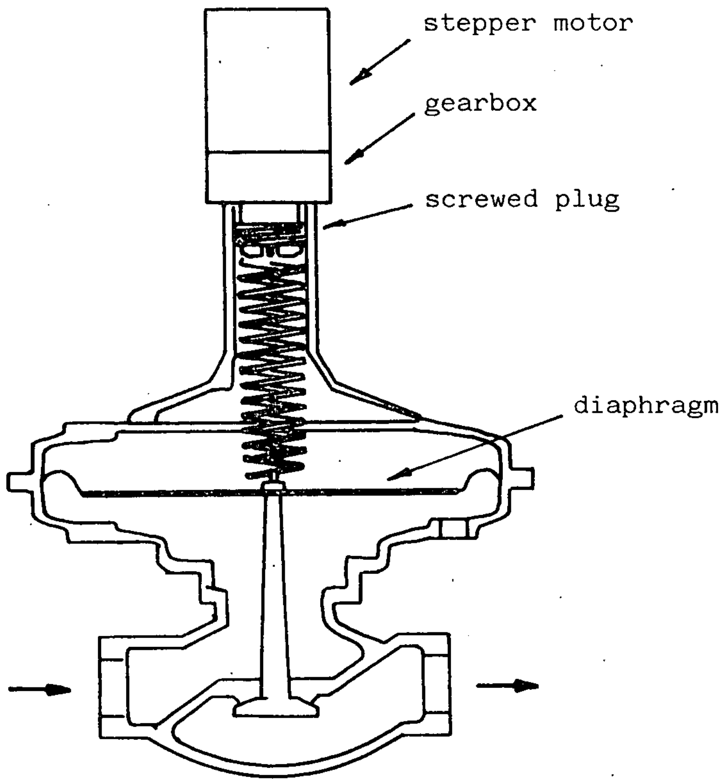


FIGURE 14 Stepper motor controlled regulator

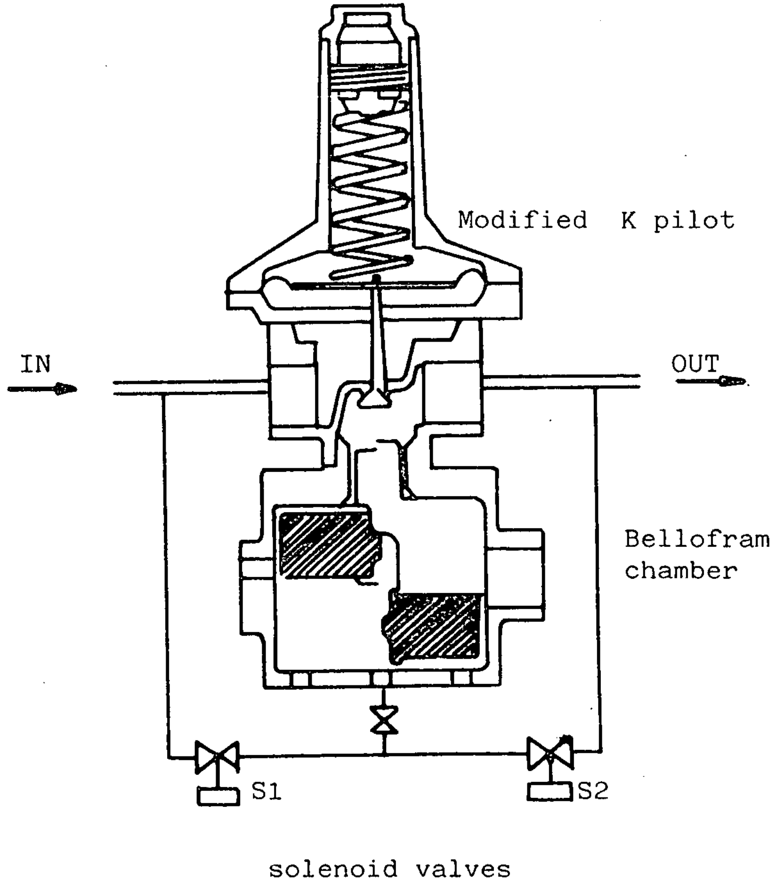


FIGURE 15 Solenoid controlled regulator

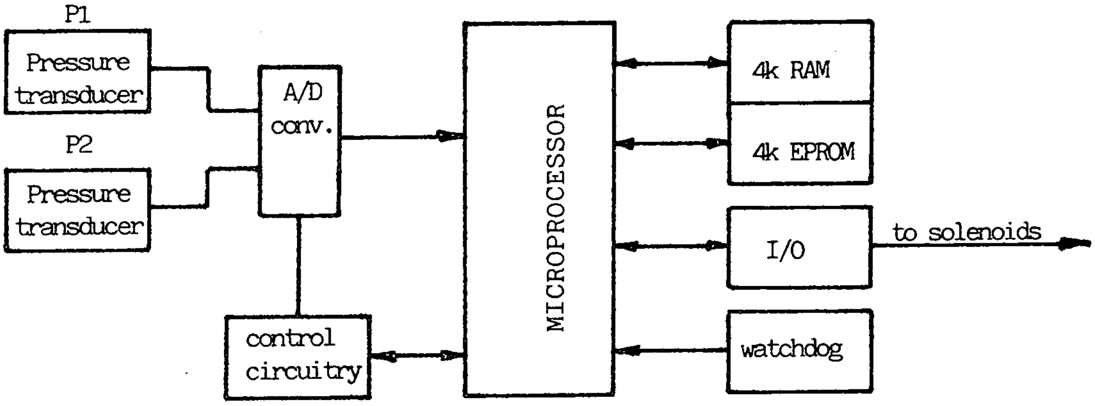


FIGURE 16 Block diagram of non fault-tolerant controller

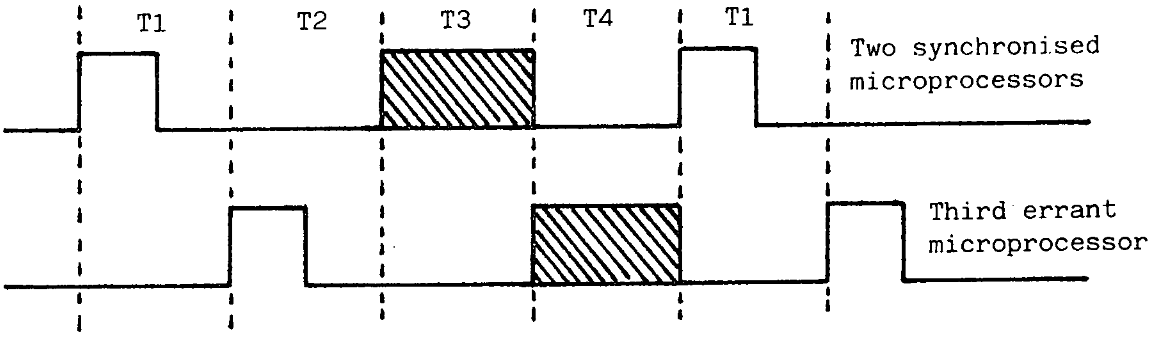


FIGURE 17 Op-code fetch timing showing synchronisation problem

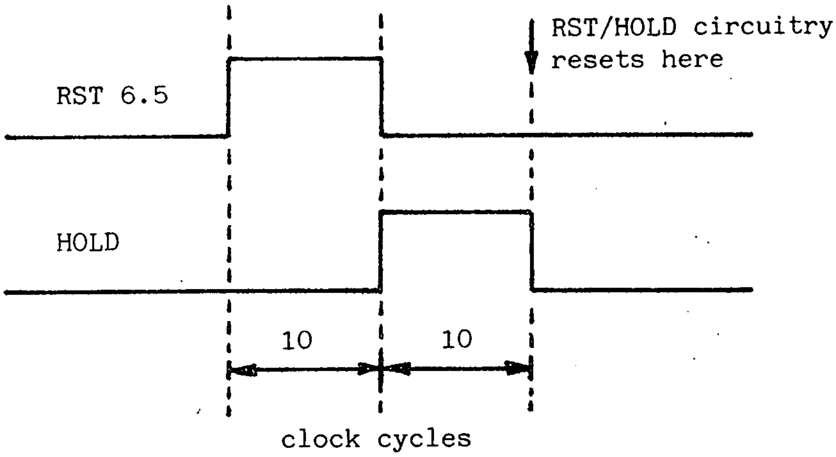


FIGURE 18 RST/HOLD Timing

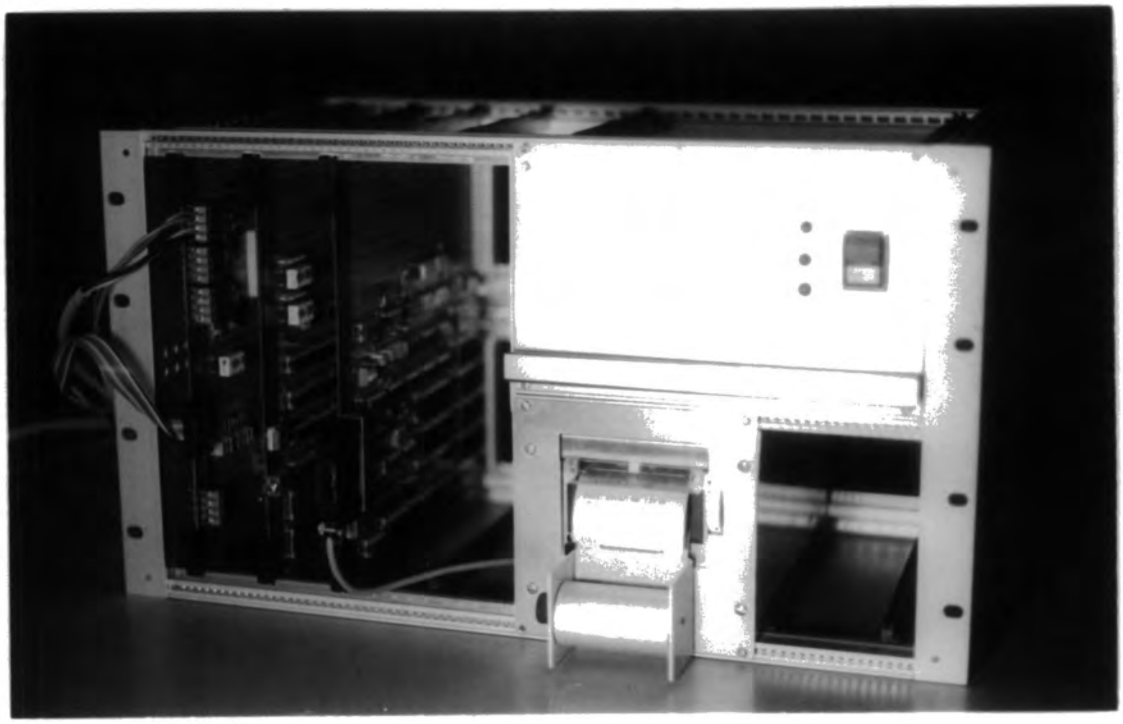


FIGURE 19 Photograph of governor controller

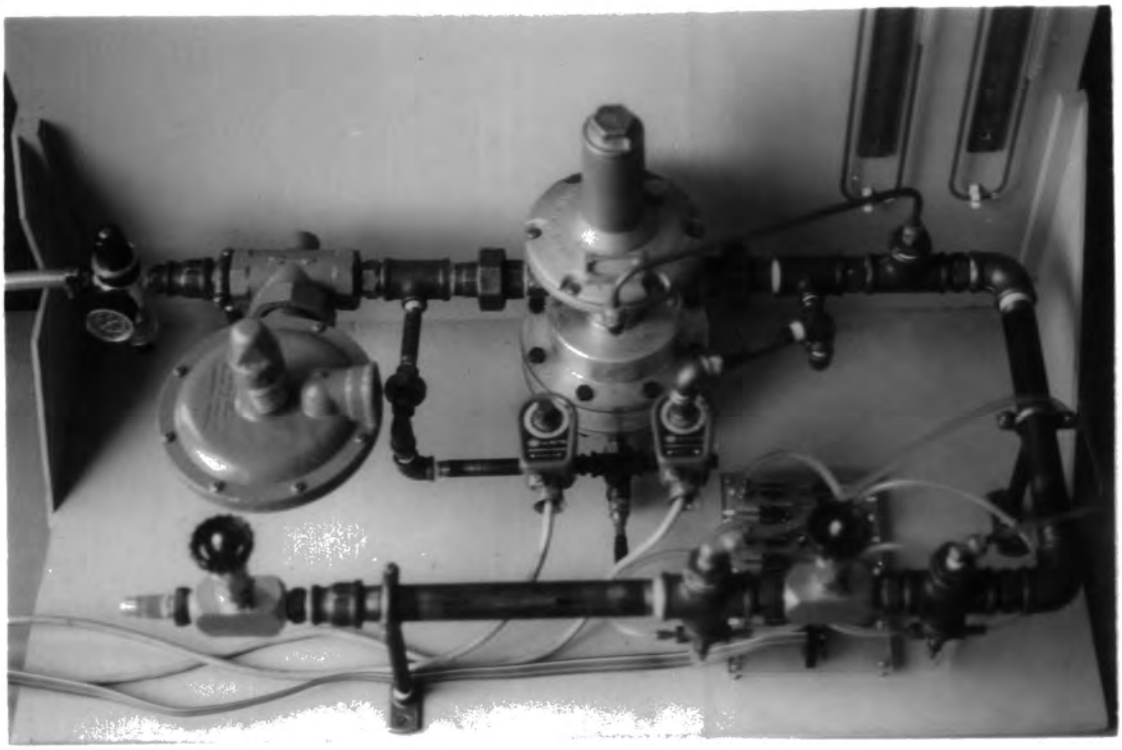


FIGURE 20 Photograph of pneumatic test rig

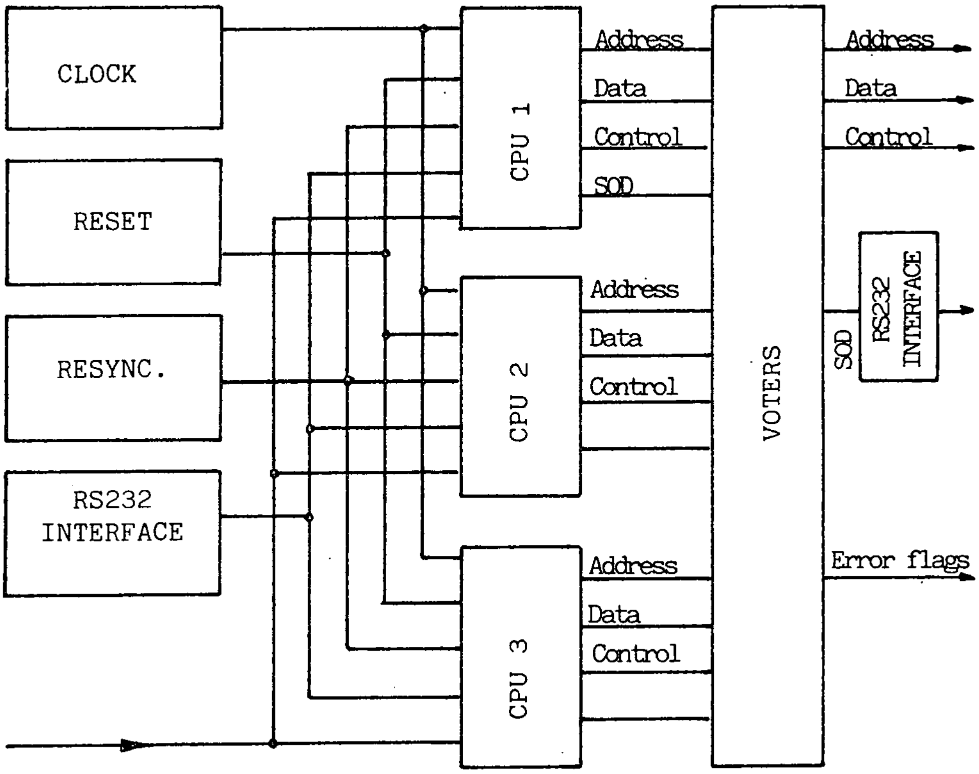


FIGURE 21 Block diagram of Microprocessor board

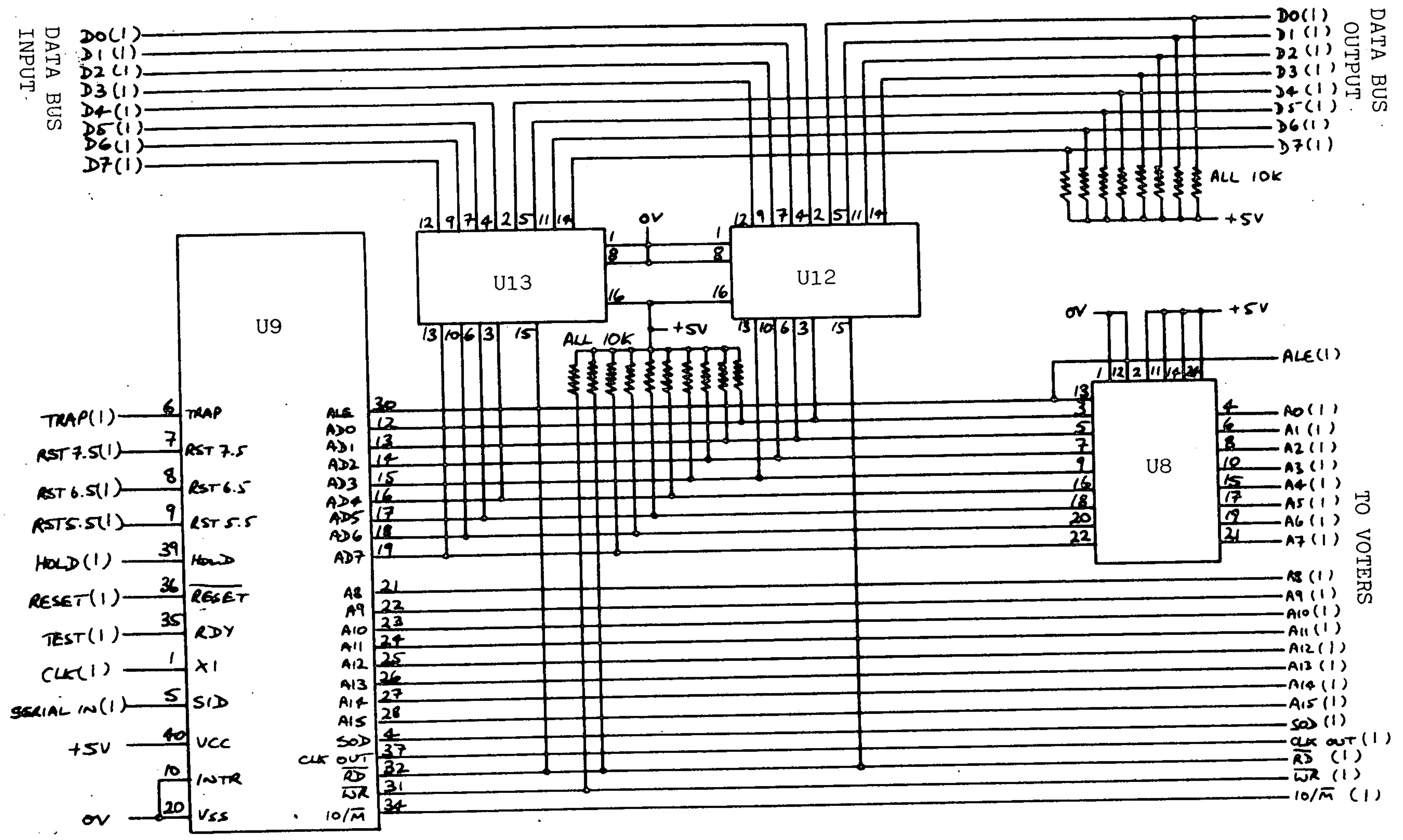


FIGURE 22 Microprocessor and buffers - One of three identical channels - CHANNEL 1

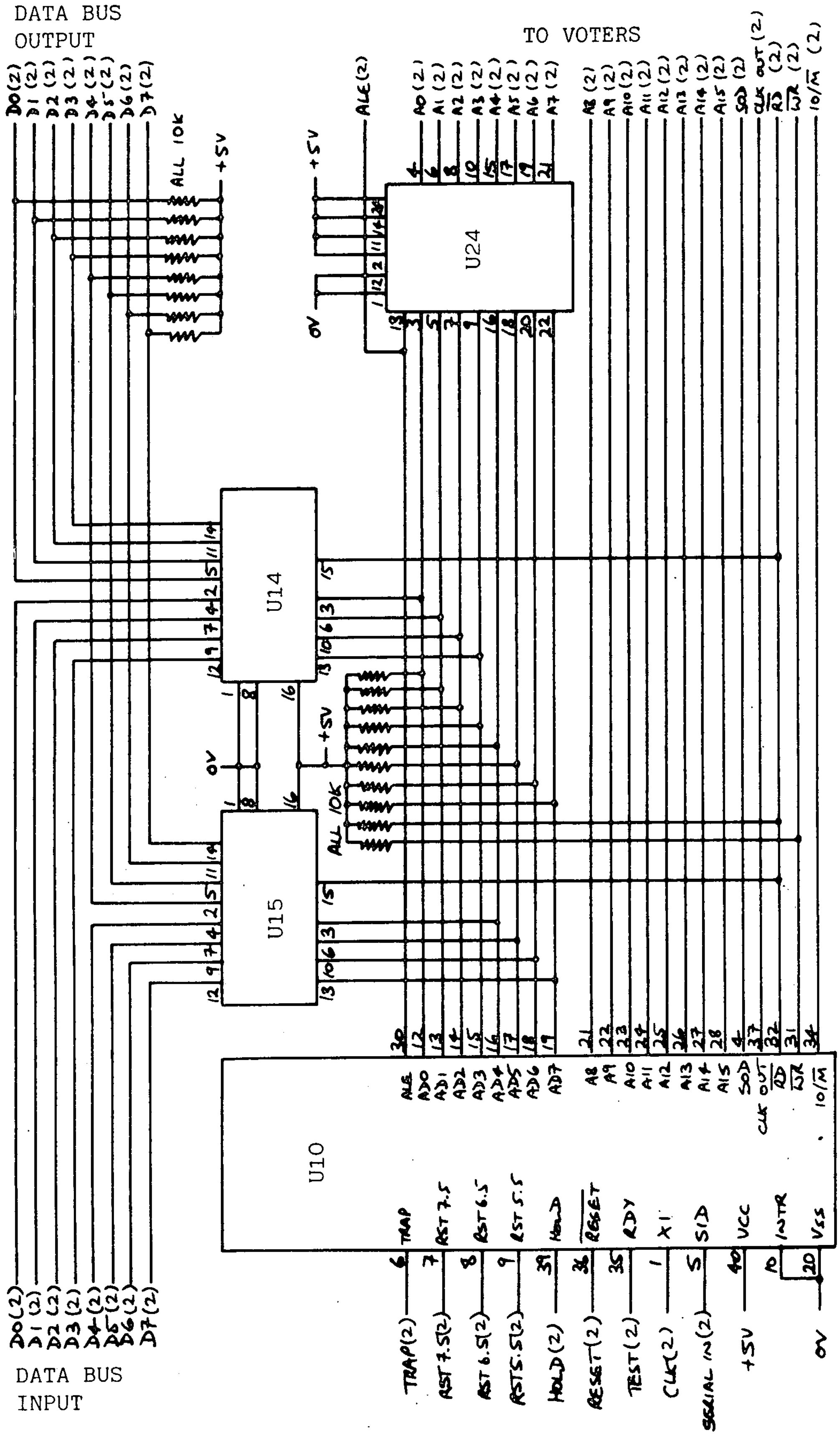


FIGURE 23 Microprocessor and buffers - One of three identical channels - CHANNEL 2

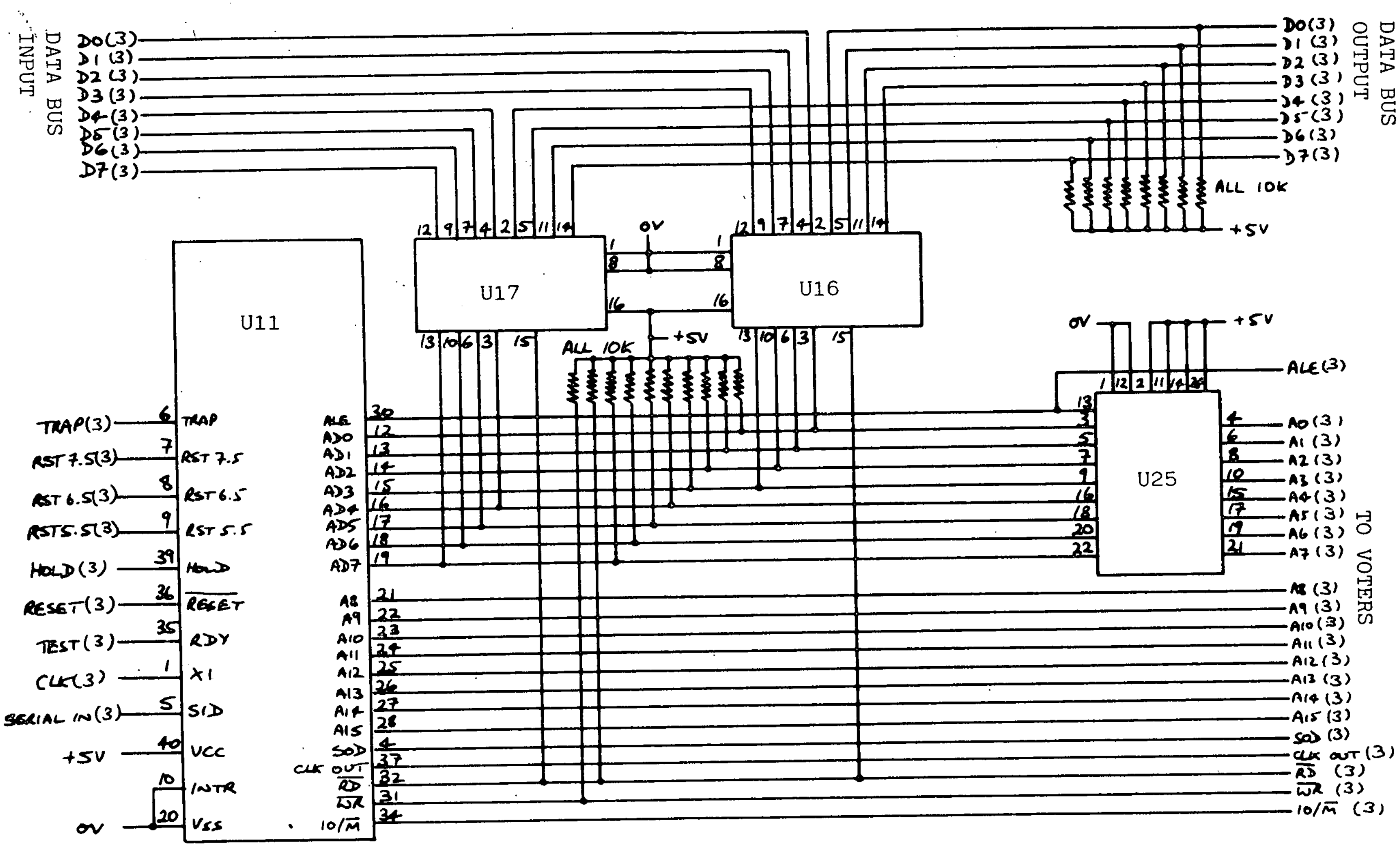


FIGURE 24 Microprocessor and buffers - One of three identical channels - CHANNEL 3

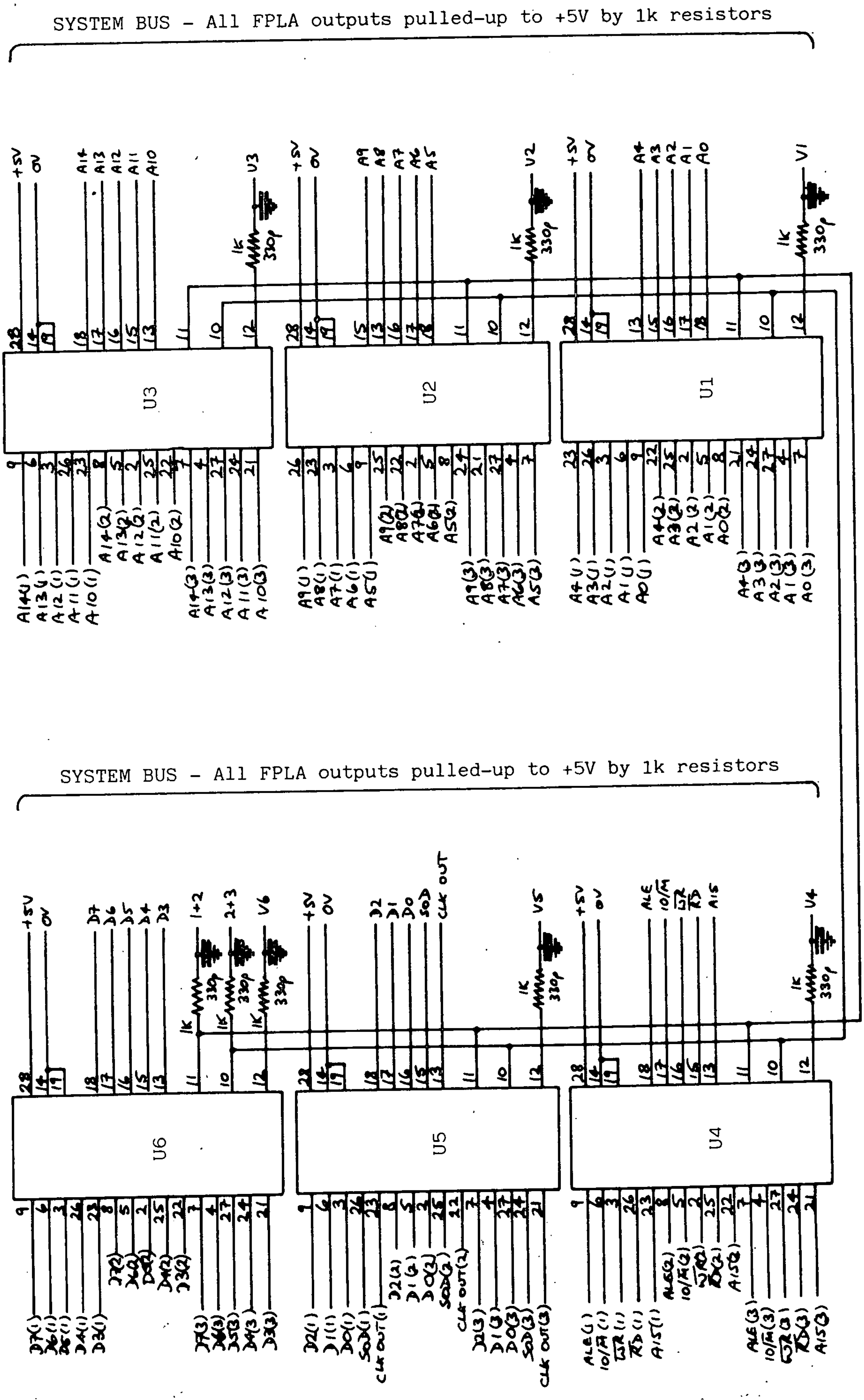

SYSTEM BUS - All FPLA outputs pulled-up to +5V by 1k resistors
U3
U2
U1
V3
V2
V1
1K
330p
+5V
0V
SYSTEM BUS - All FPLA outputs pulled-up to +5V by 1k resistors
U6
U5
U4
V6
V5
V4
1+2
2+3
CLK OUT
ALE

FIGURE 25 Voting circuitry

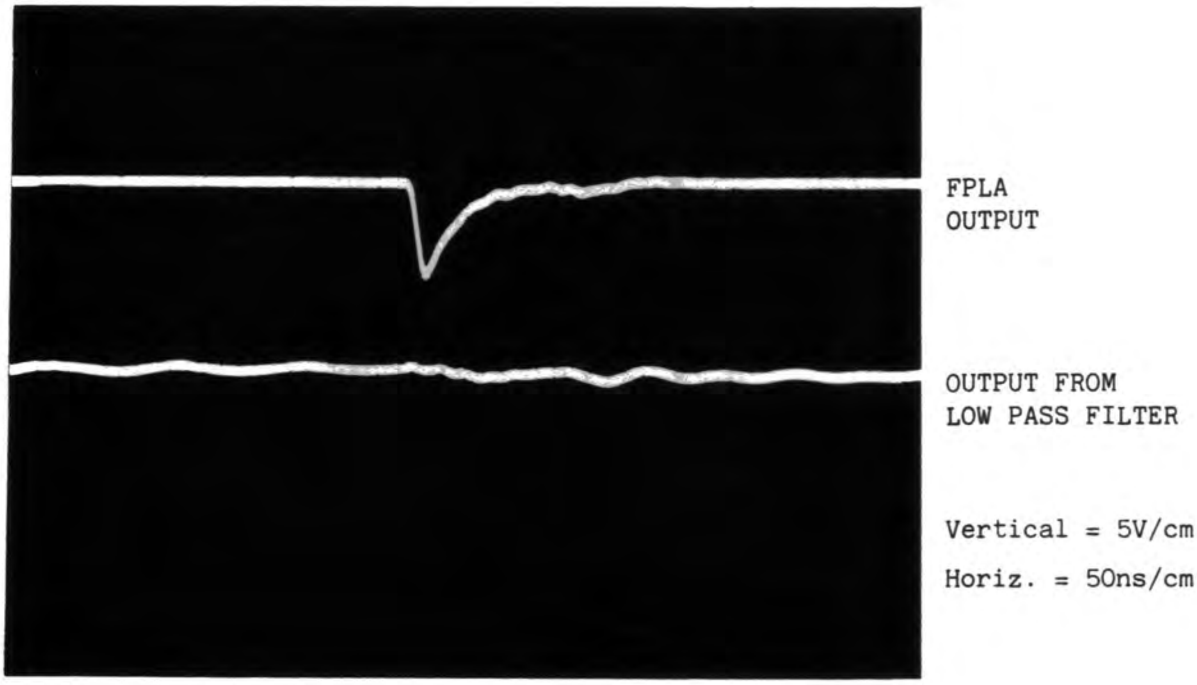


FIGURE 26 FPLA error signal showing the effect of filtering

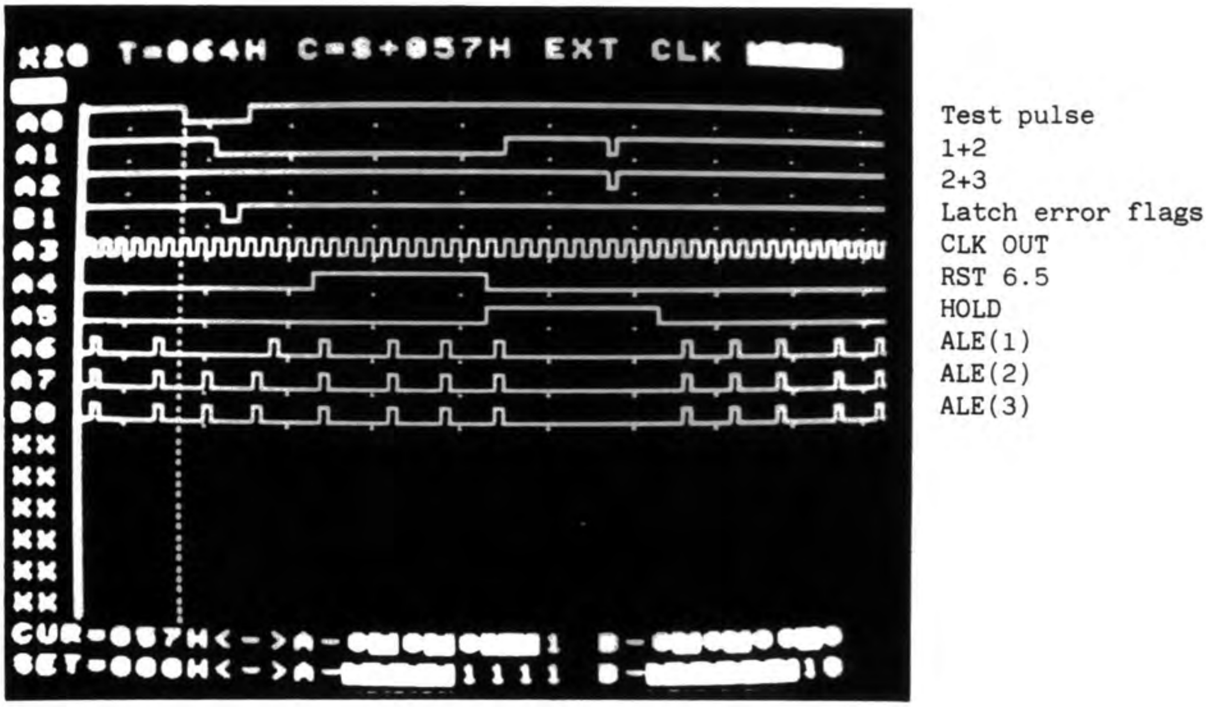
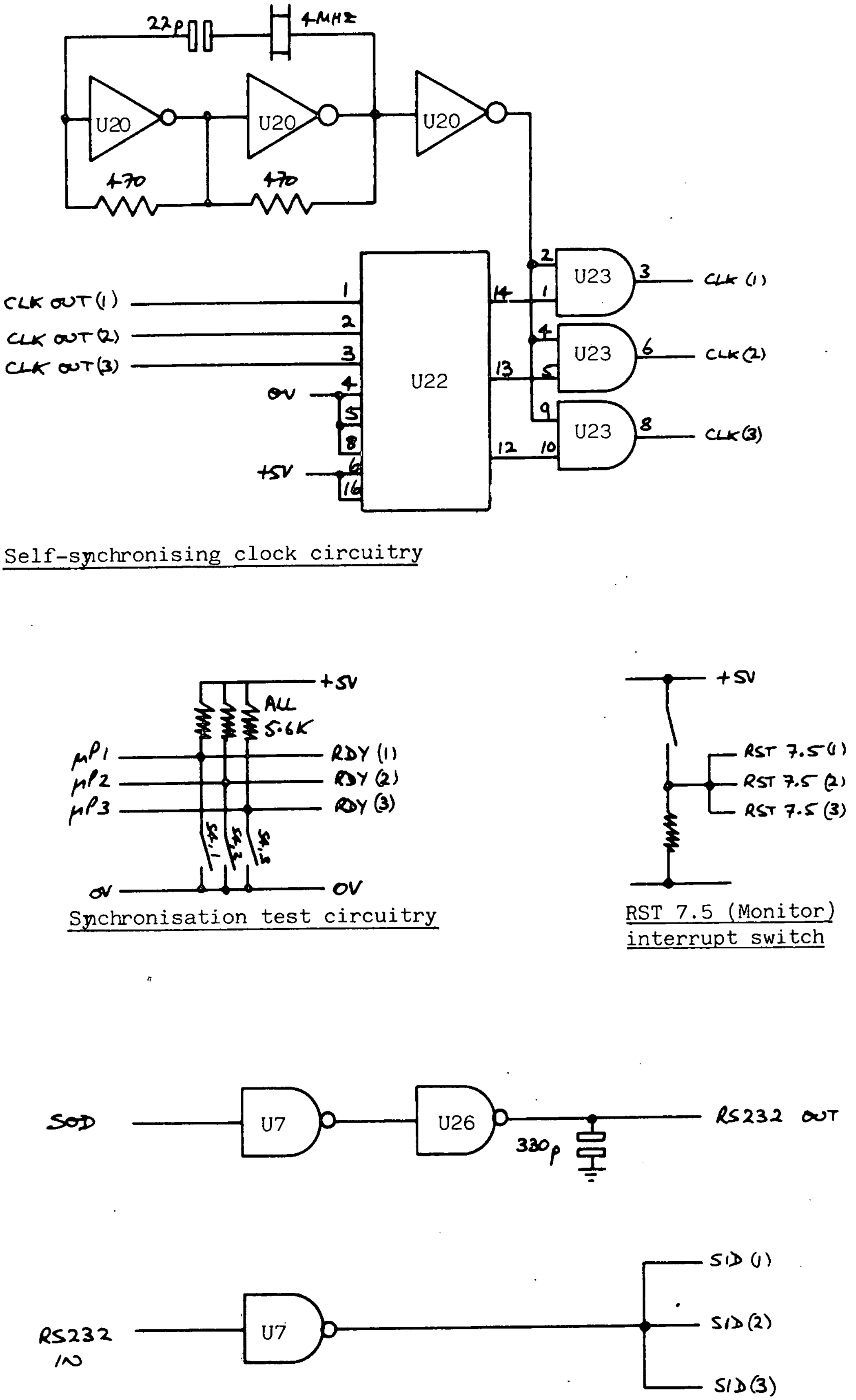


FIGURE 27 Resynchronisation timing monitored by logic analyser



Self-synchronising clock circuitry

Synchronisation test circuitry

RST 7.5 (Monitor) interrupt switch

RS232 Interface

FIGURE 28 Clock, test, and RS232 interface

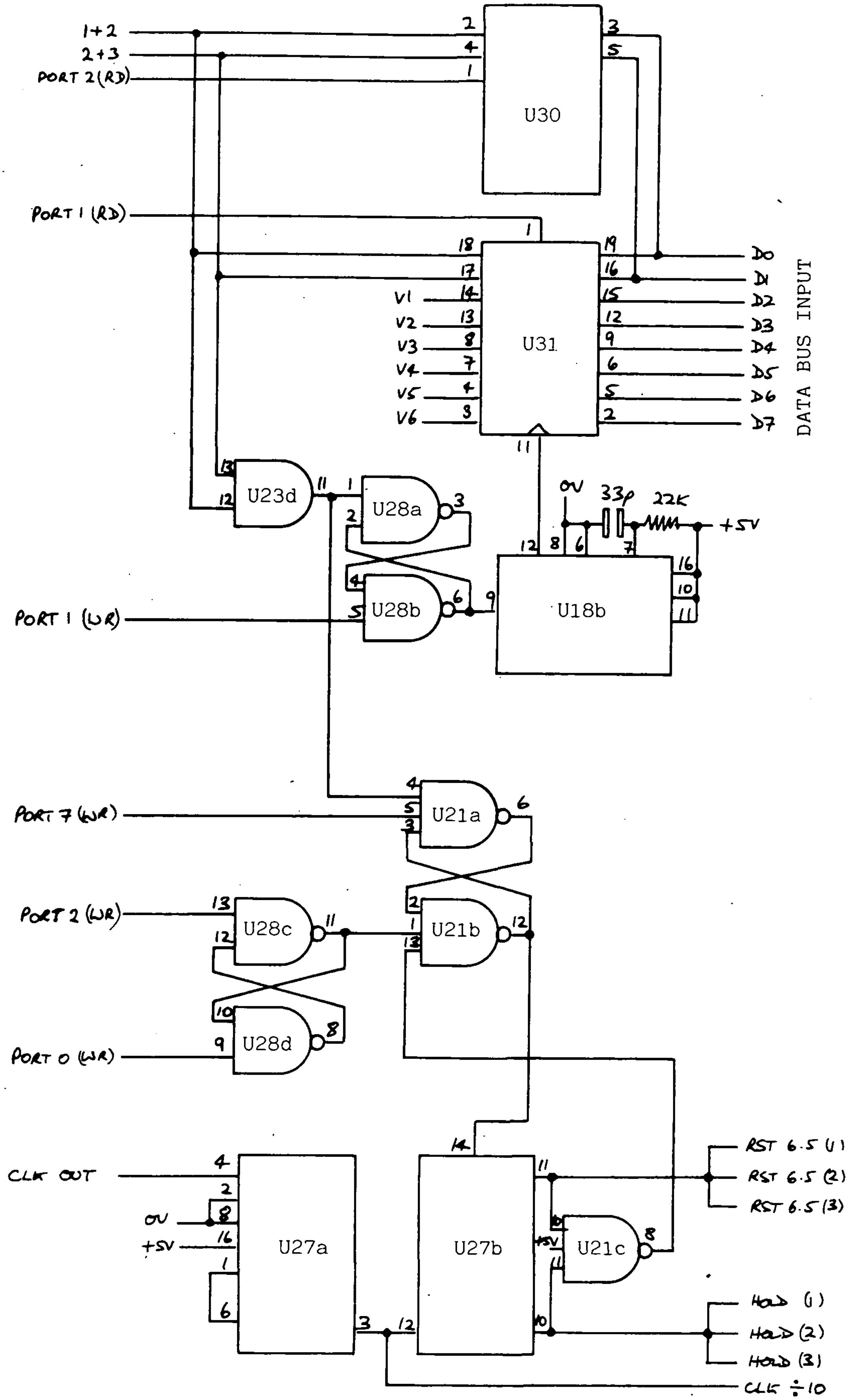


FIGURE 29 Resynchronisation circuitry

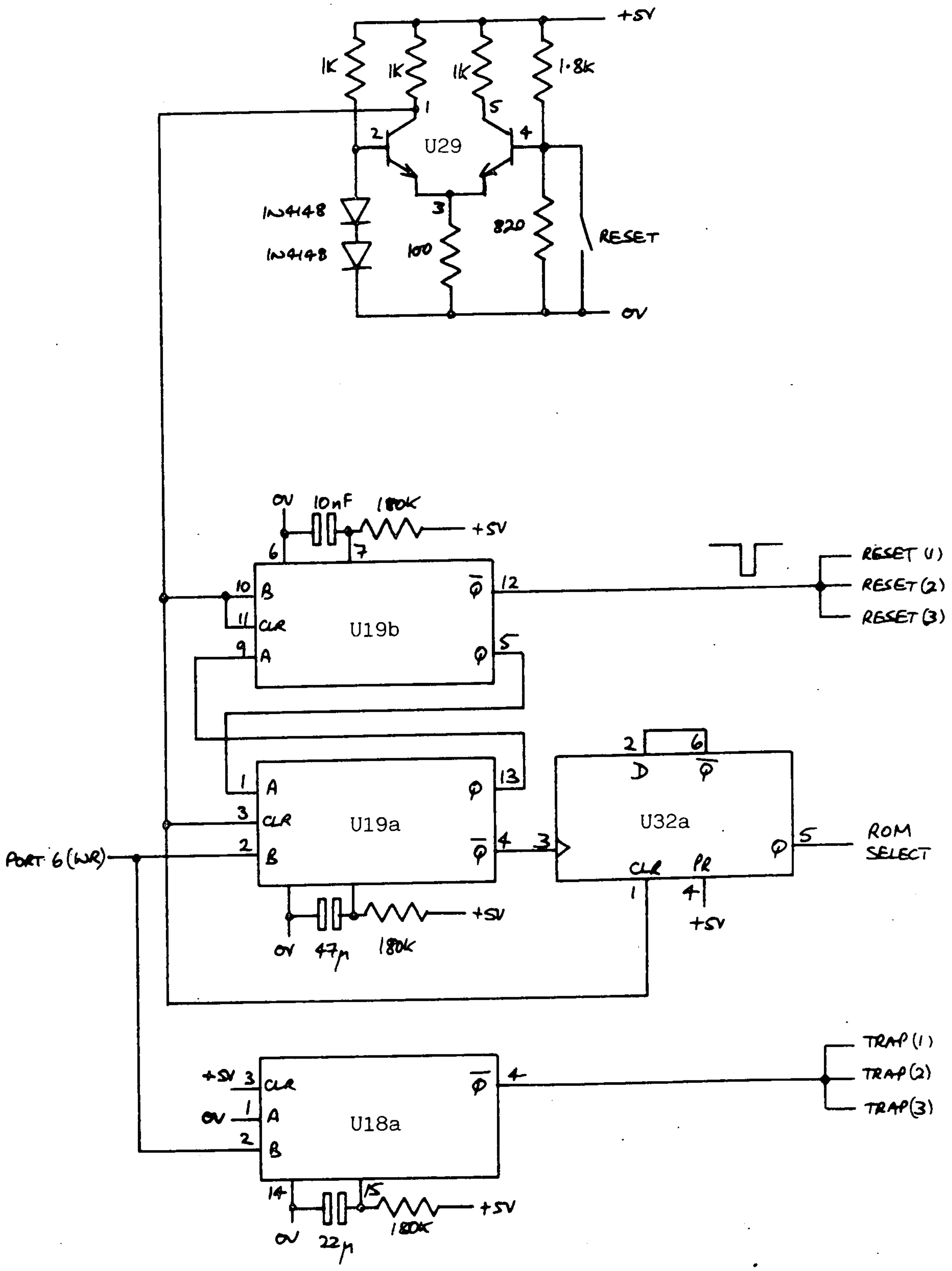


FIGURE 30 Watchdogs and reset circuitry

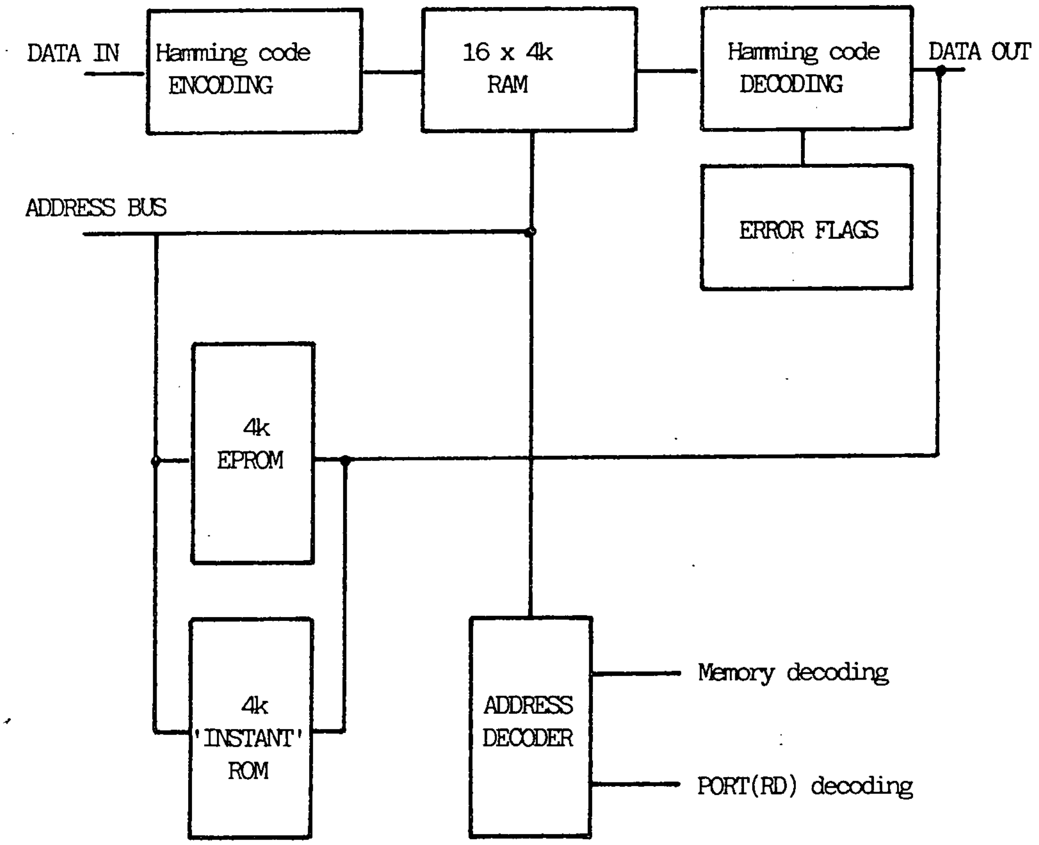


FIGURE 31 Block diagram of memory board

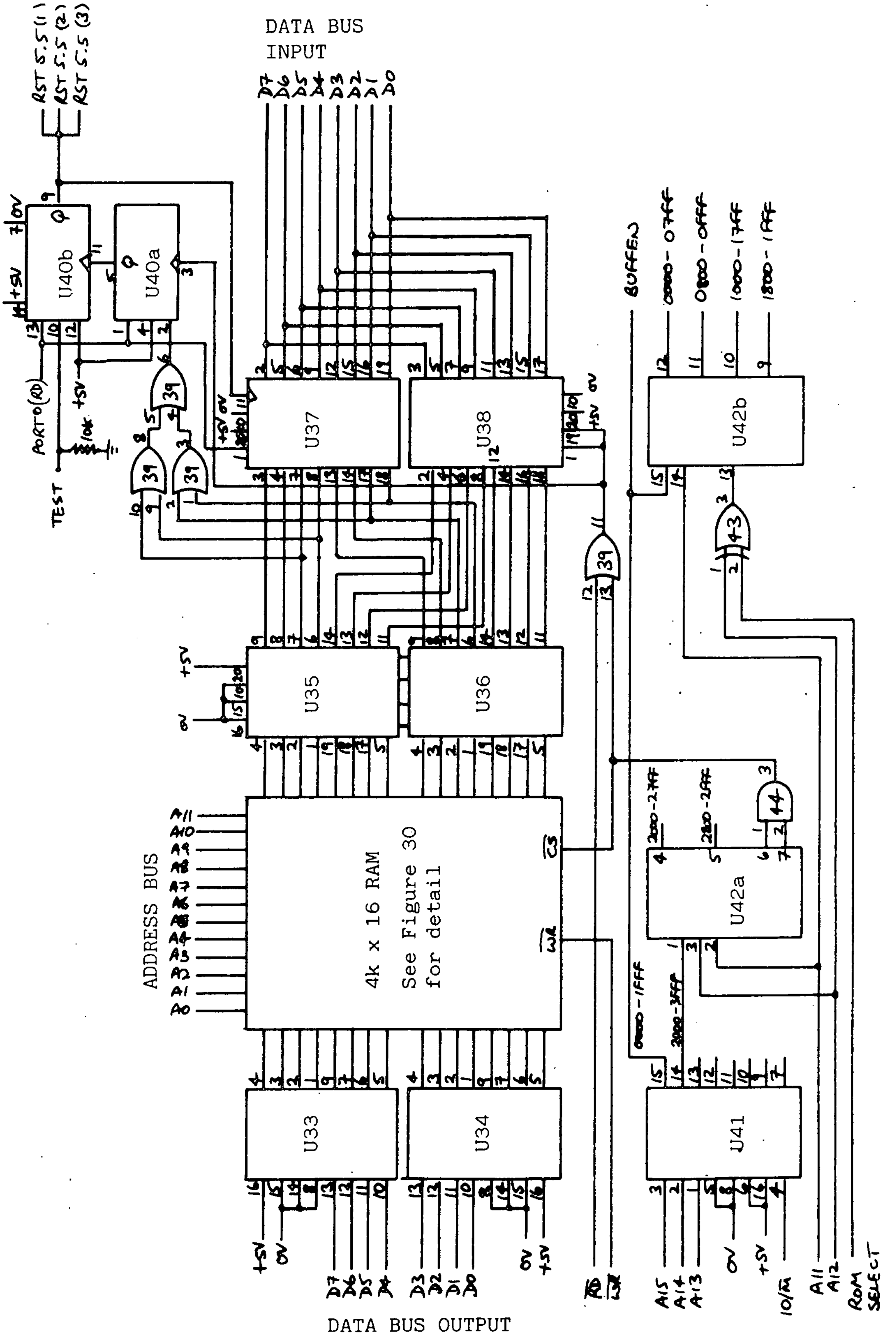


FIGURE 32 Part of Memory board including RAM and decoders

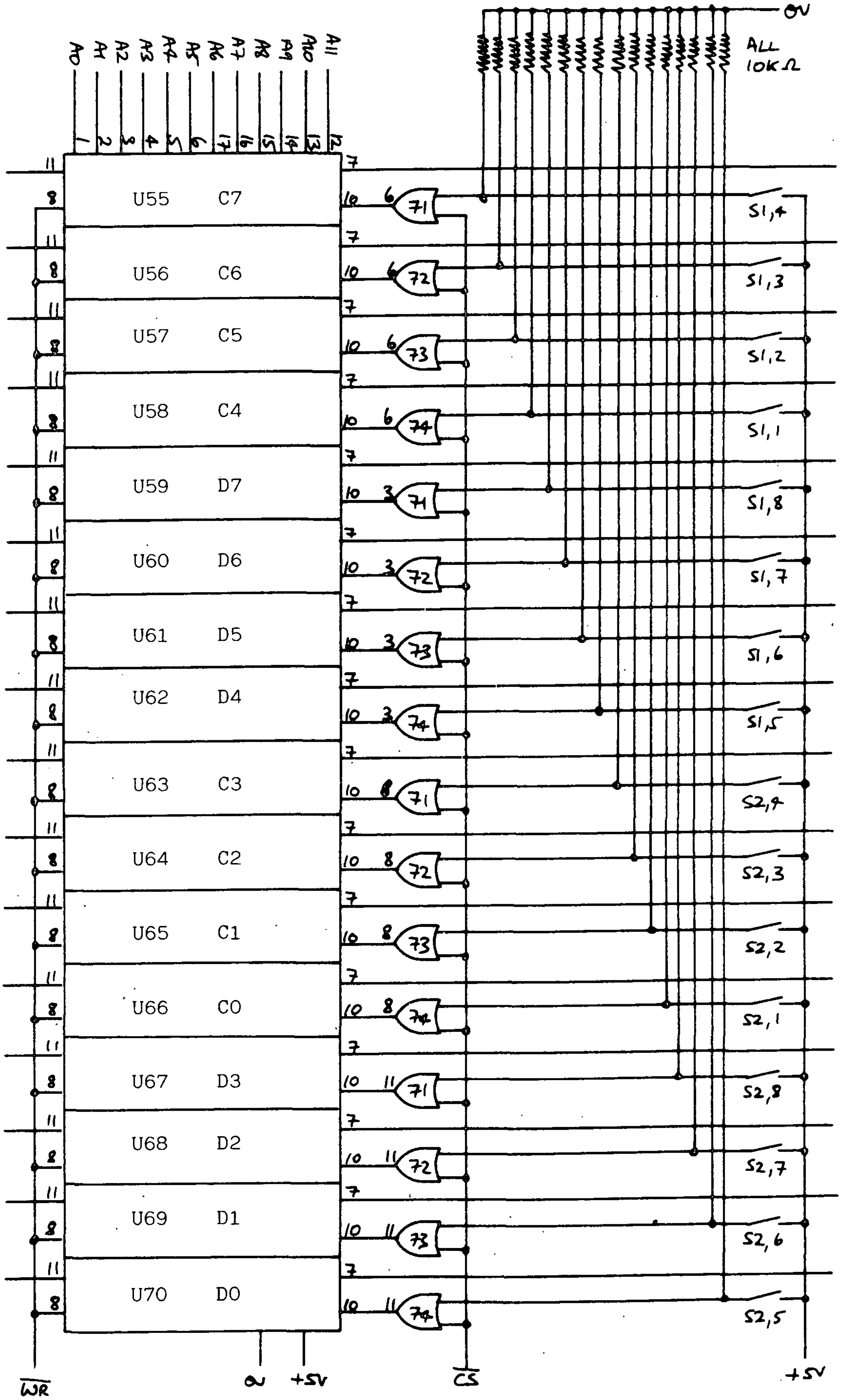


FIGURE 33 Detail of RAM storage circuitry including test circuitry

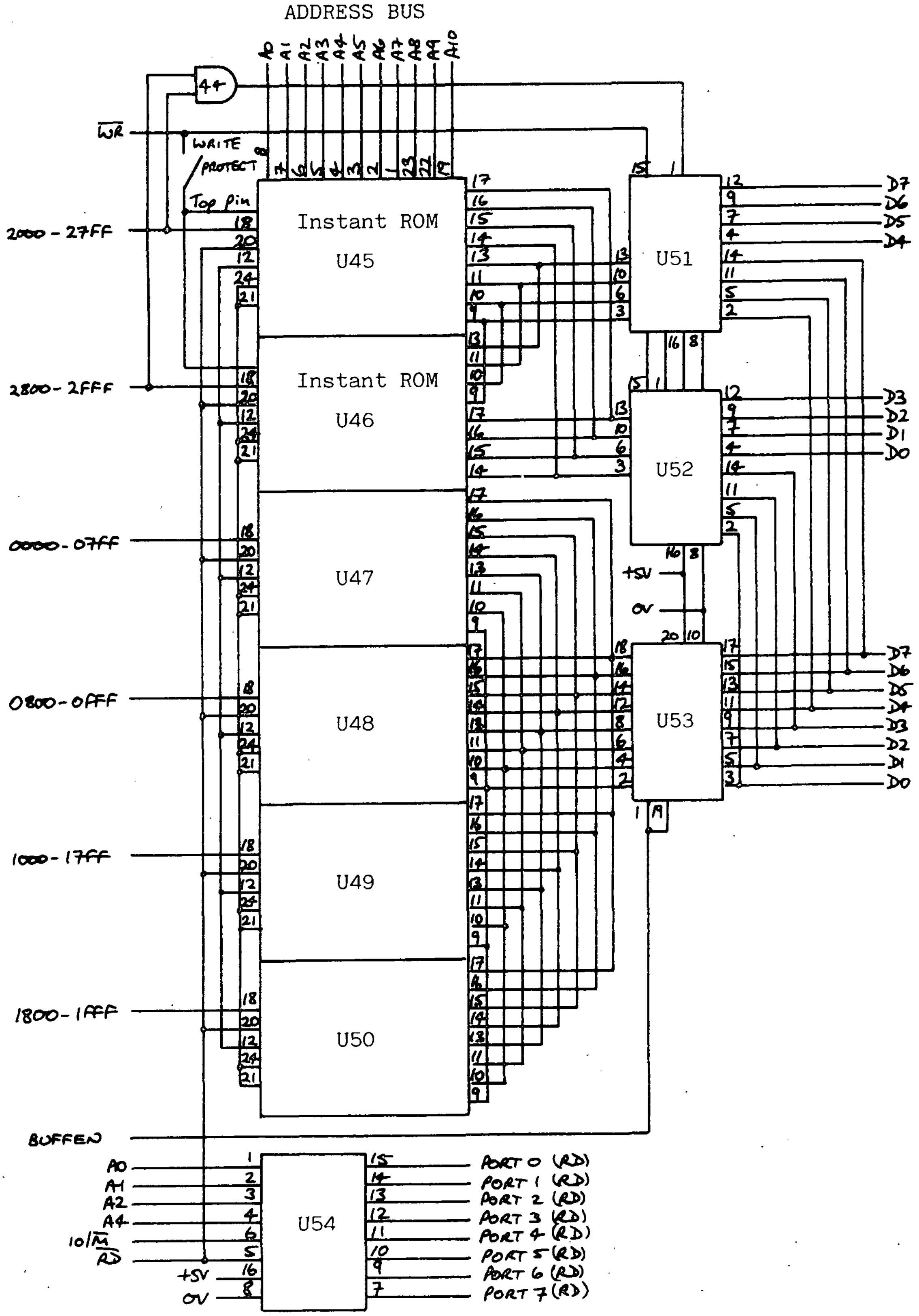


FIGURE 34 Instant ROM, EPROM, and PORT(RD) decoder circuit diagram

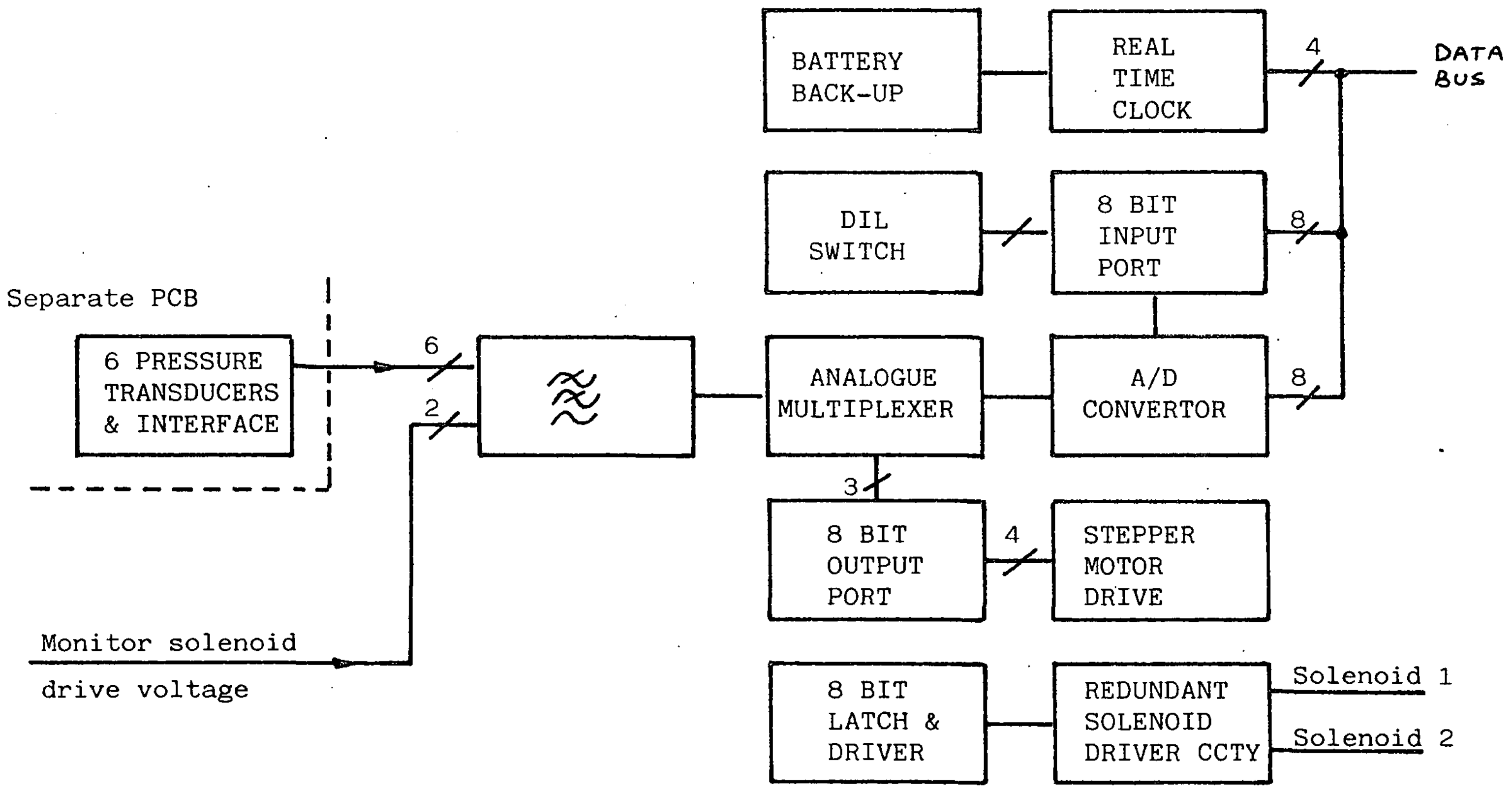


FIGURE 35   Block diagram of input/output board

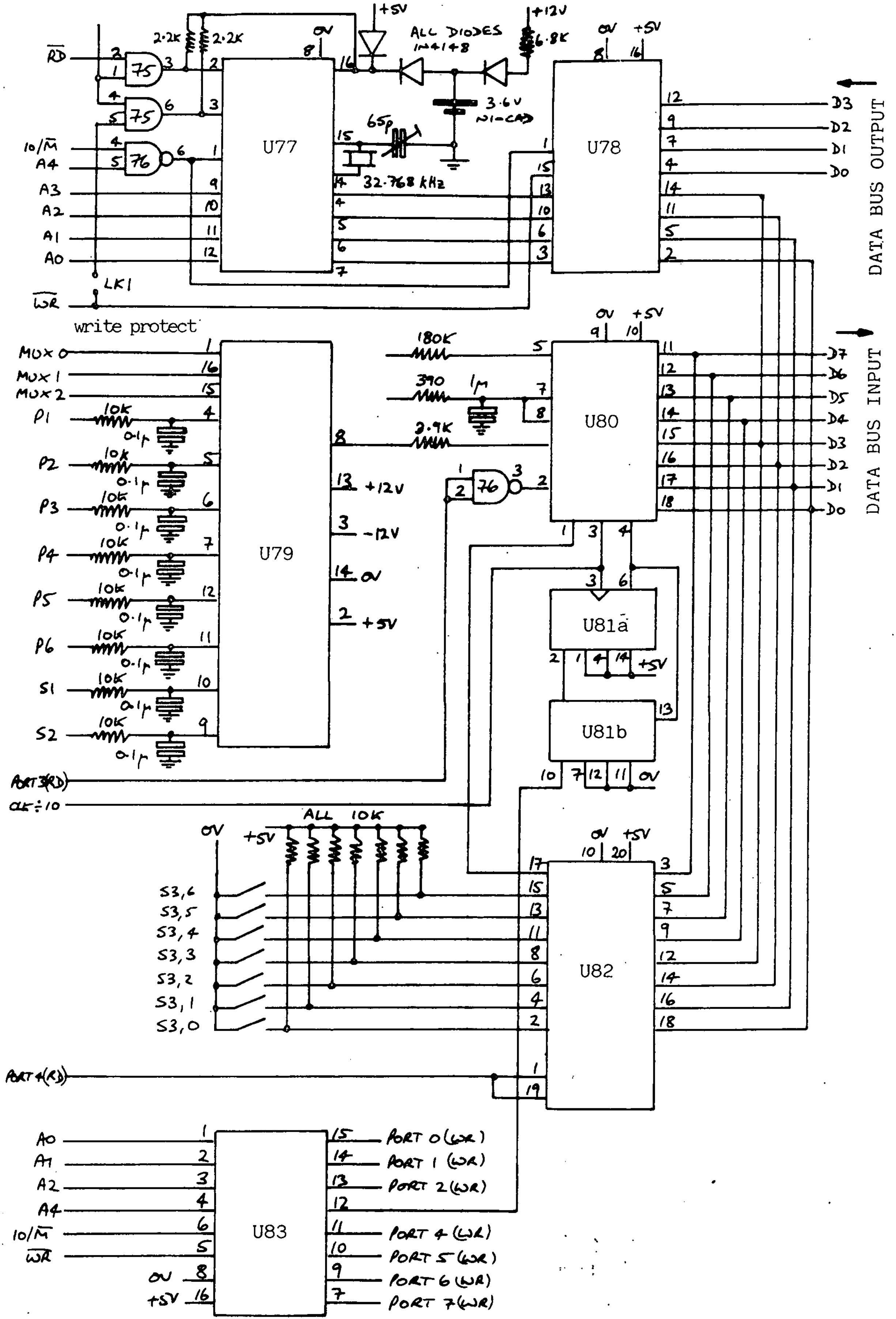


FIGURE 36 Input/output board

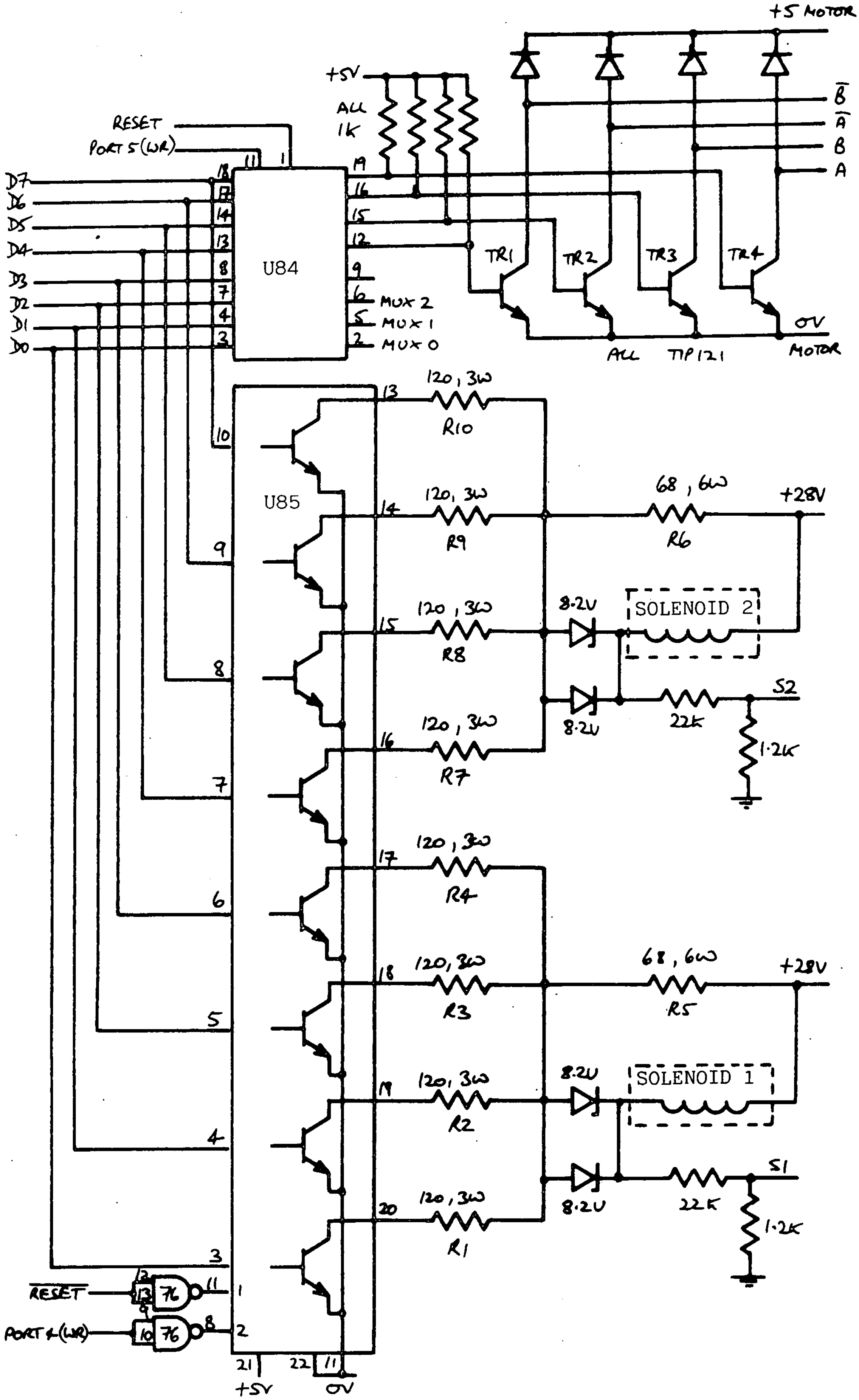


FIGURE 37 Stepper motor and redundant solenoid drive circuitry

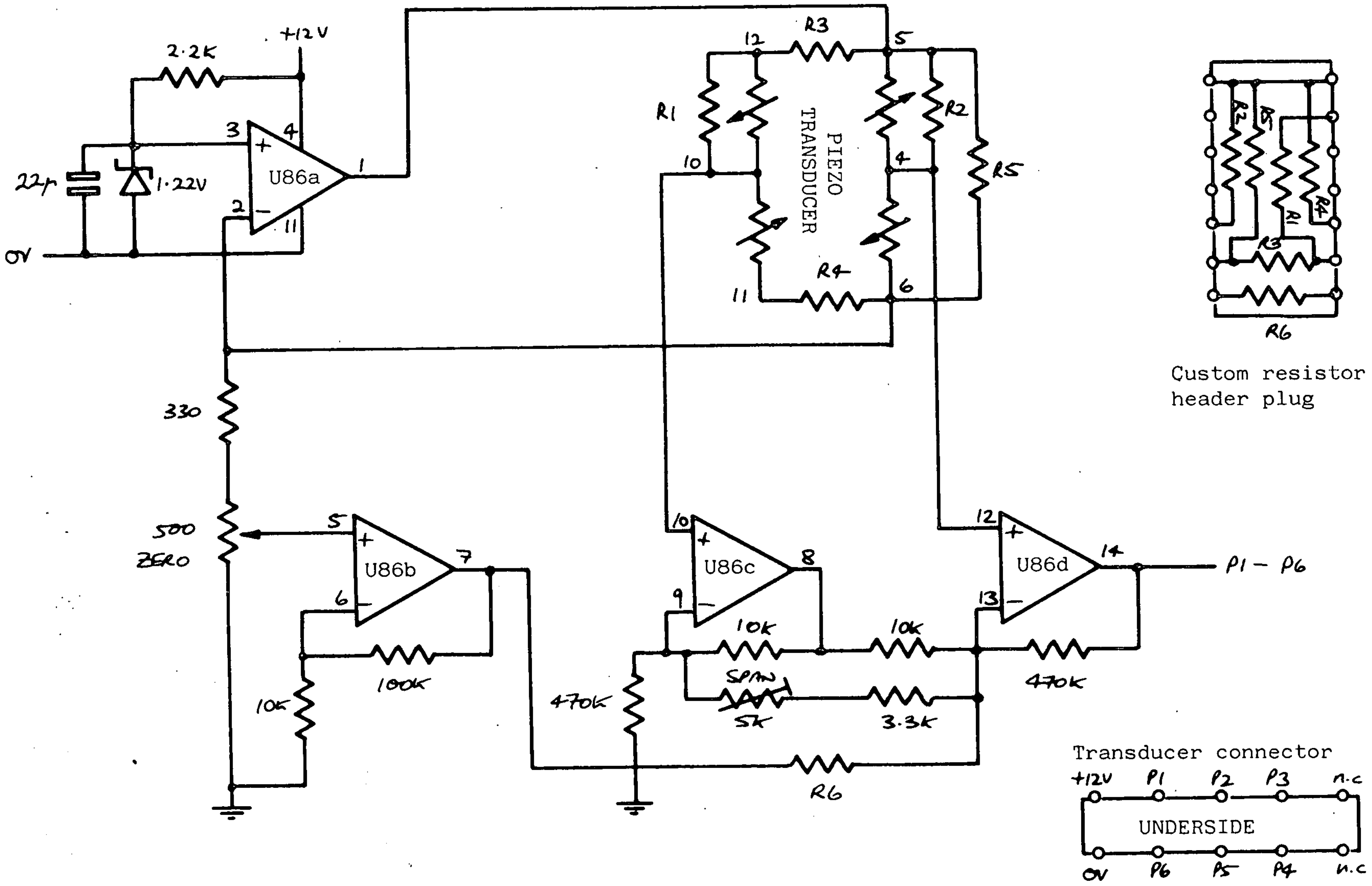


FIGURE 38 Pressure transducer and conditioning circuitry

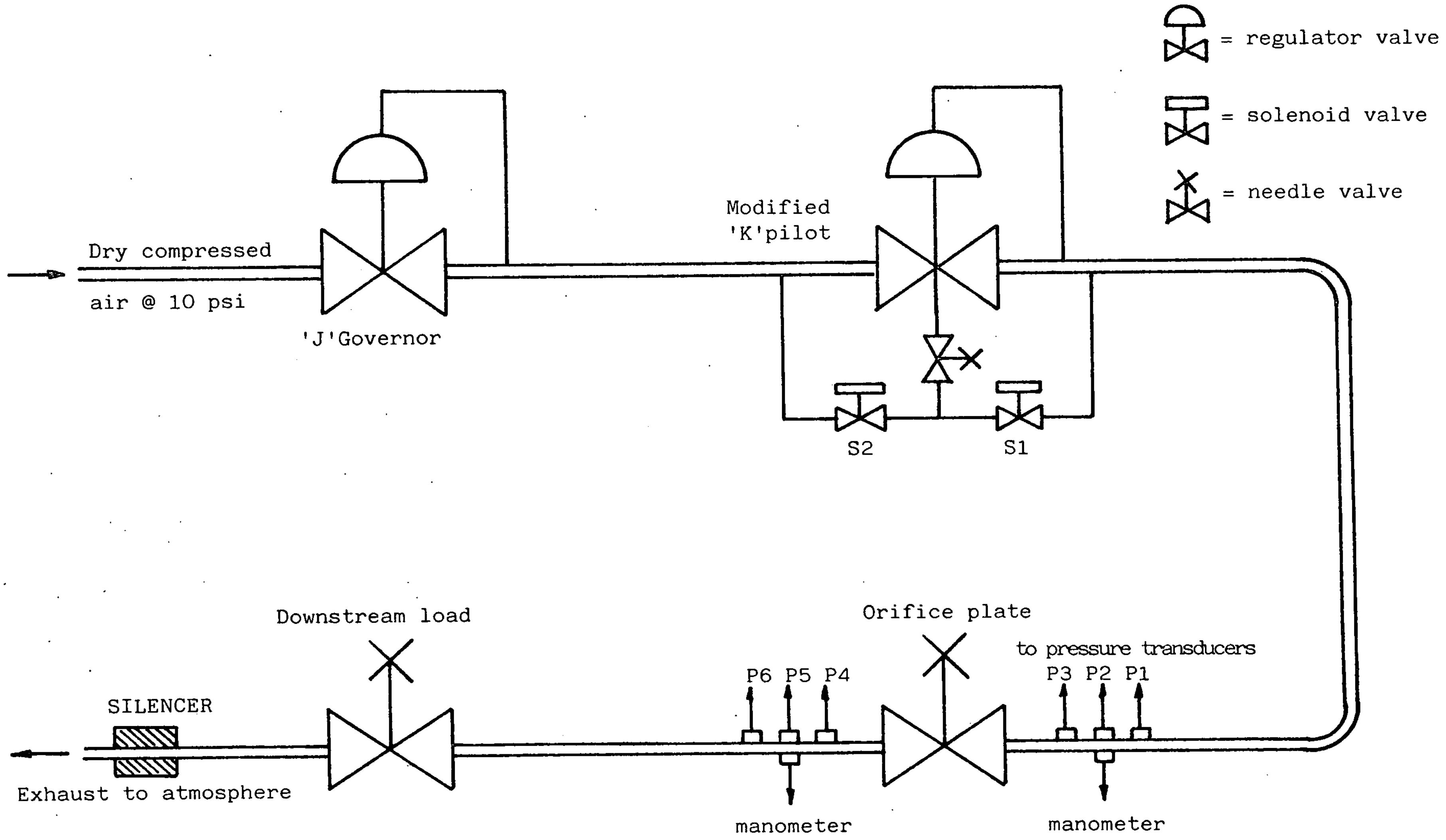


FIGURE 39 Block diagram of pneumatic test rig

MAIN

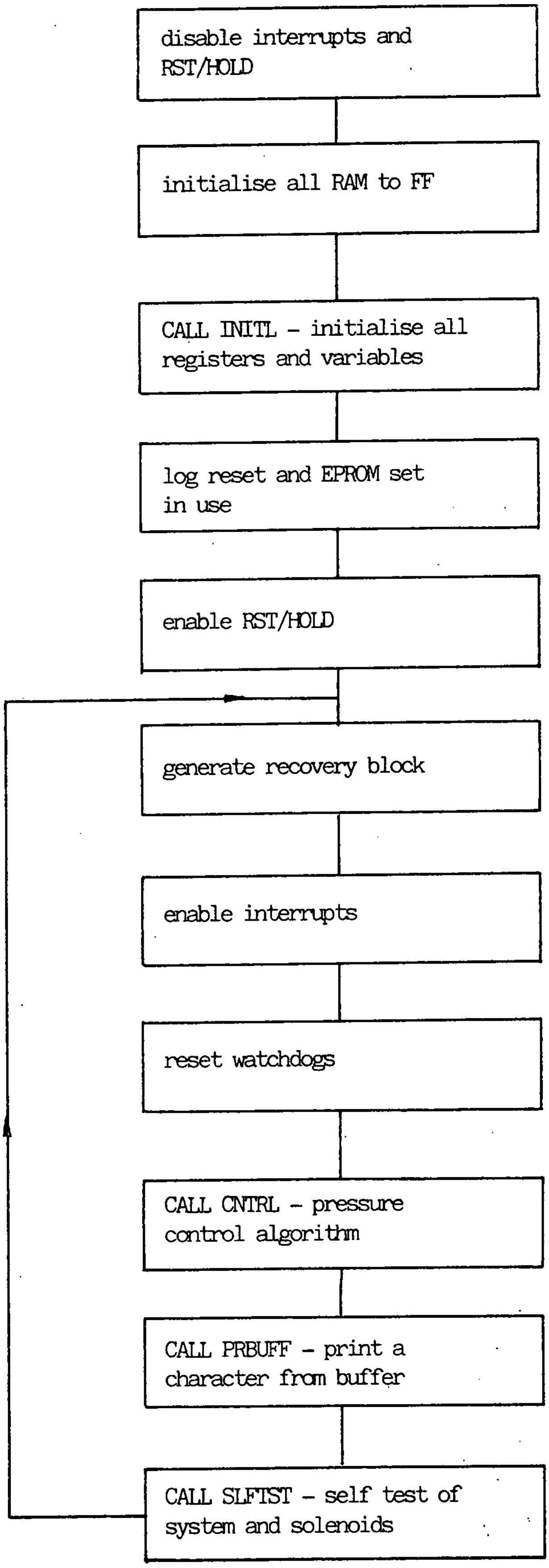

disable interrupts and RST/HOLD
initialise all RAM to FF
CALL INITL - initialise all registers and variables
log reset and EPROM set in use
enable RST/HOLD
generate recovery block
enable interrupts
reset watchdogs
CALL CNTRL - pressure control algorithm
CALL PRBUFF - print a character from buffer
CALL SLFTST - self test of system and solenoids

FIGURE 40 Flow chart of module MAIN

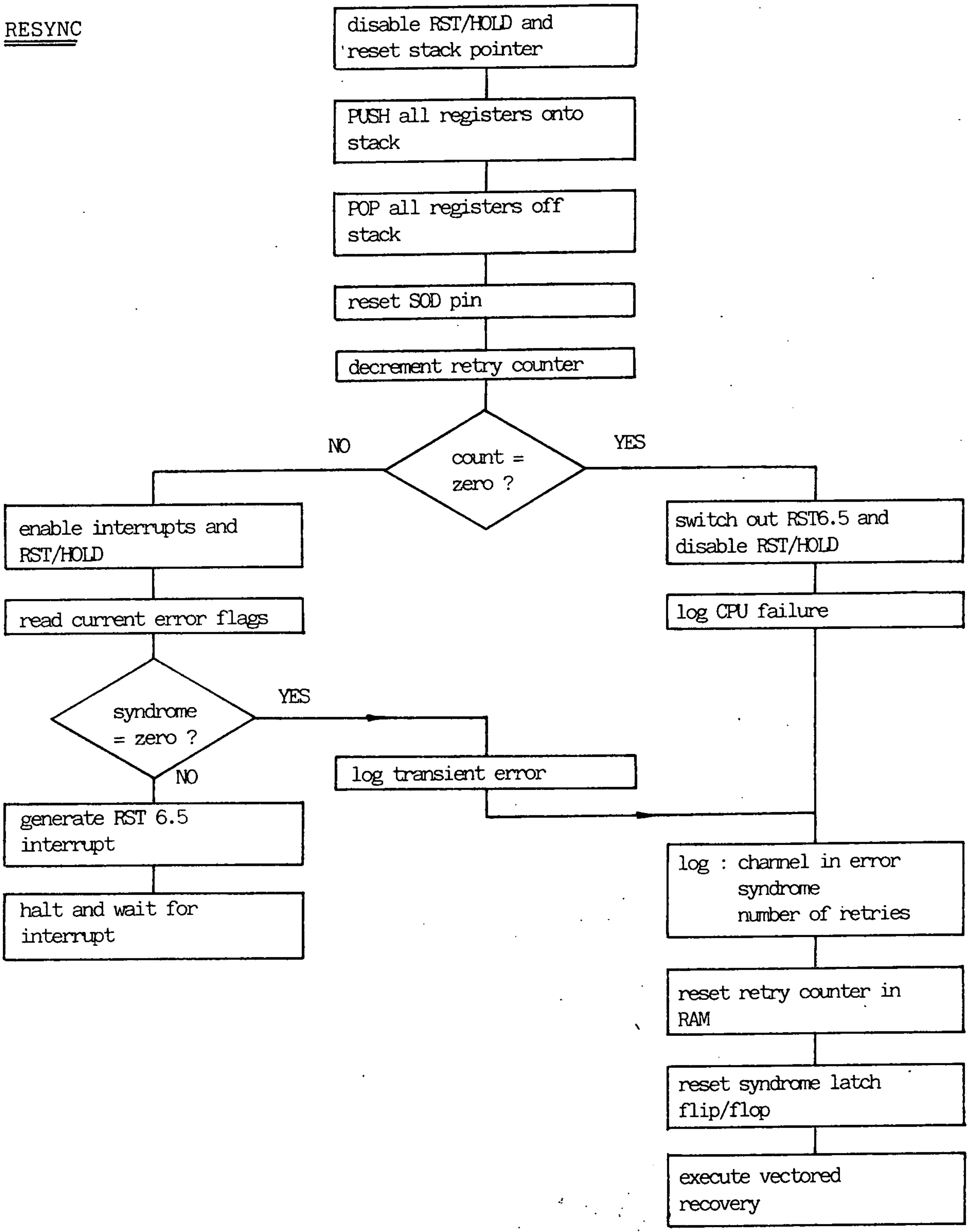

RESYNC
disable RST/HOLD and reset stack pointer
PUSH all registers onto stack
POP all registers off stack
reset SOD pin
decrement retry counter
count = zero ?
NO
YES
enable interrupts and RST/HOLD
read current error flags
syndrome = zero ?
YES
NO
generate RST 6.5 interrupt
halt and wait for interrupt
log transient error
switch out RST6.5 and disable RST/HOLD
log CPU failure
log : channel in error
syndrome
number of retries
reset retry counter in RAM
reset syndrome latch flip/flop
execute vectored recovery

FIGURE 41 Flow chart of module RESYNC

MERROR

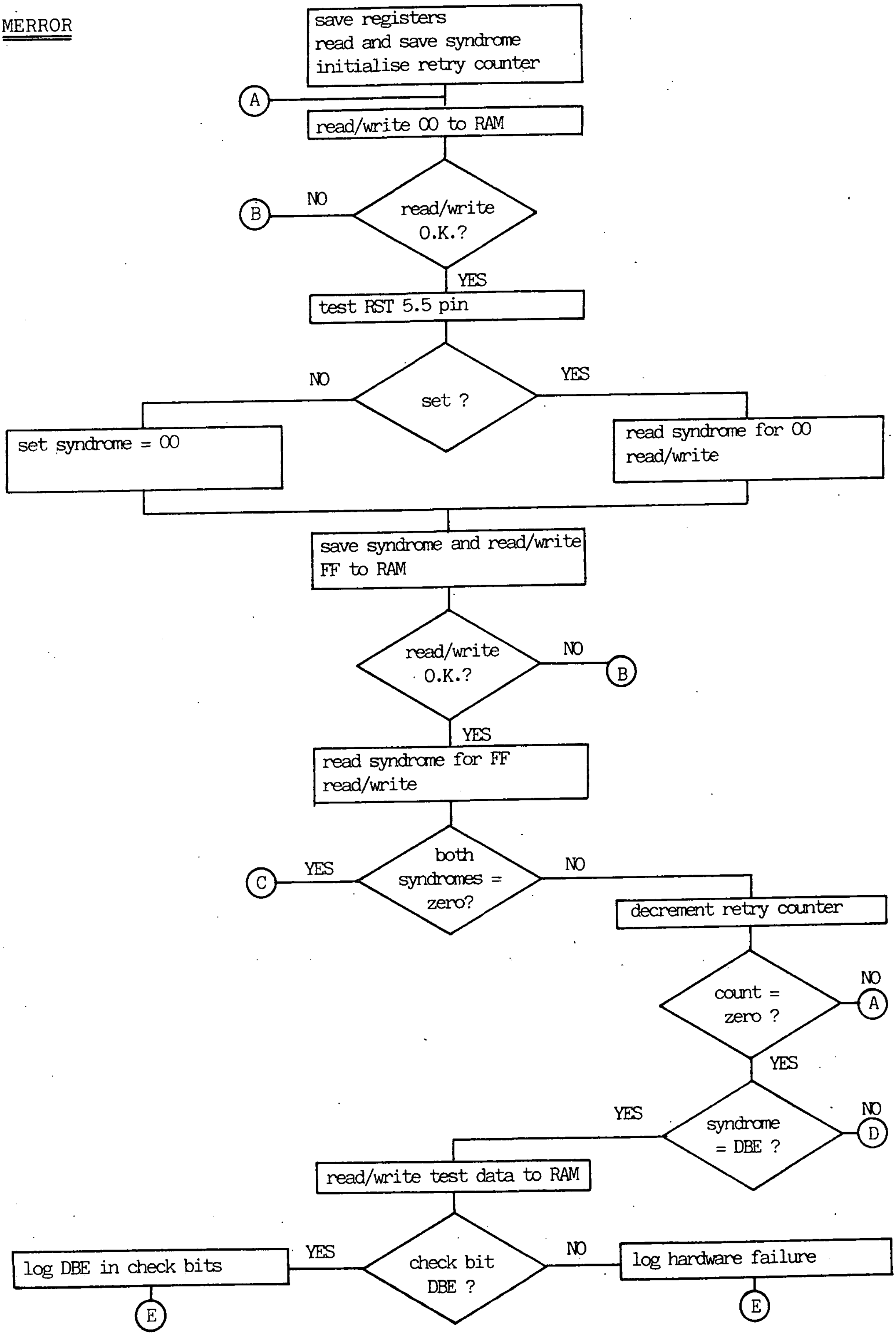


FIGURE 42 Flow chart of module MERROR

MERROR

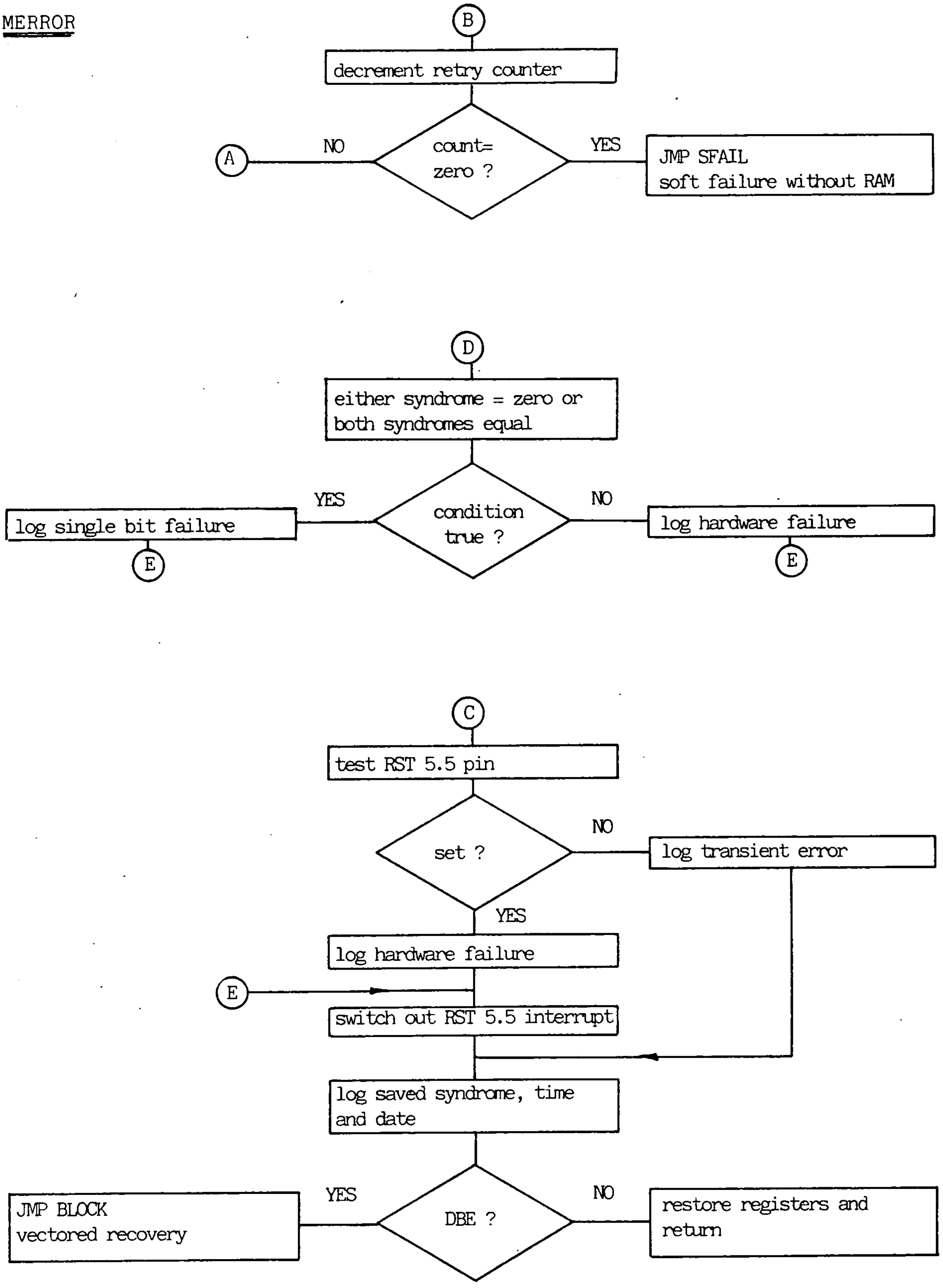


FIGURE 43 Flow chart of module MERROR

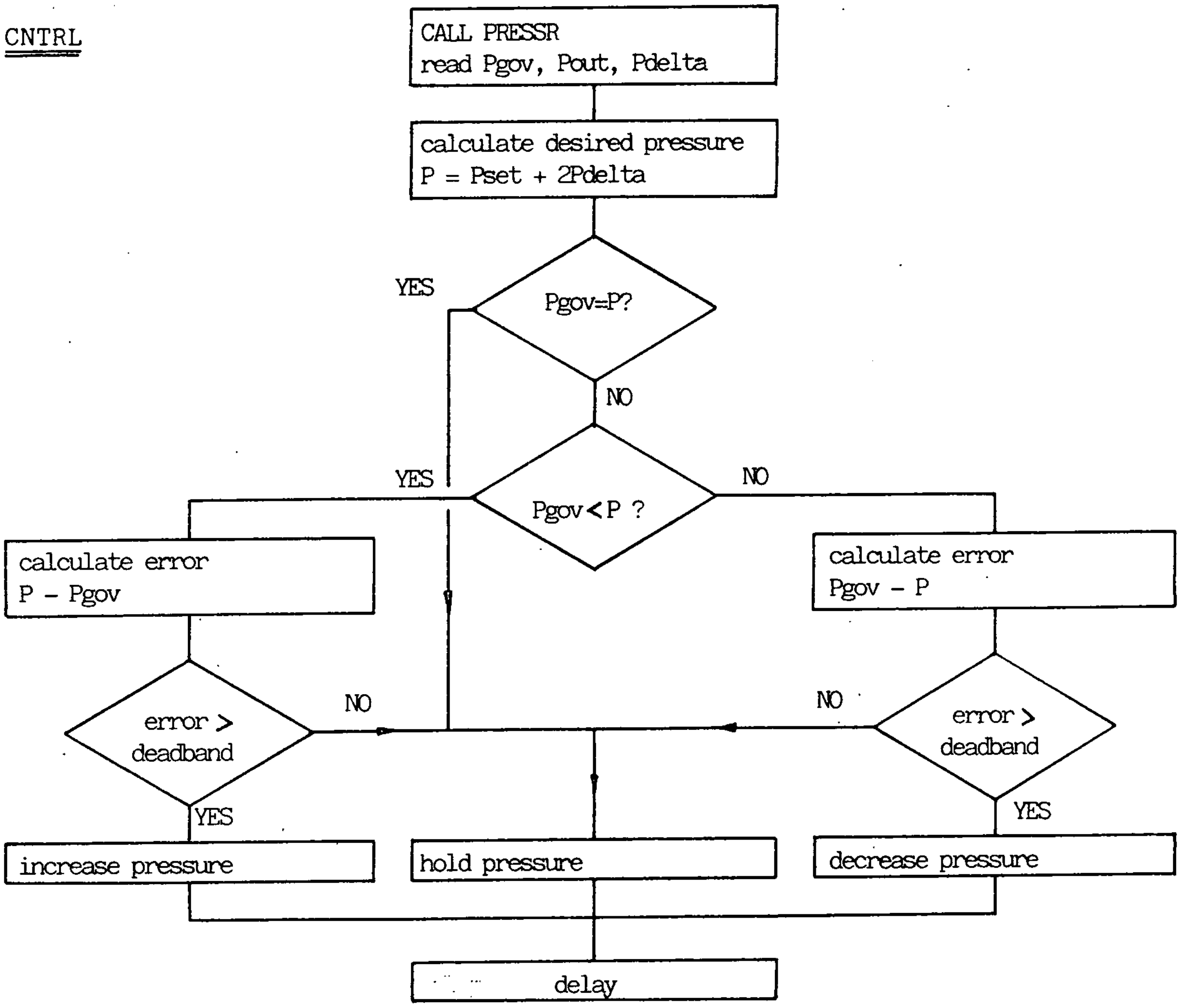


FIGURE 44 Flow chart of module CNTRL

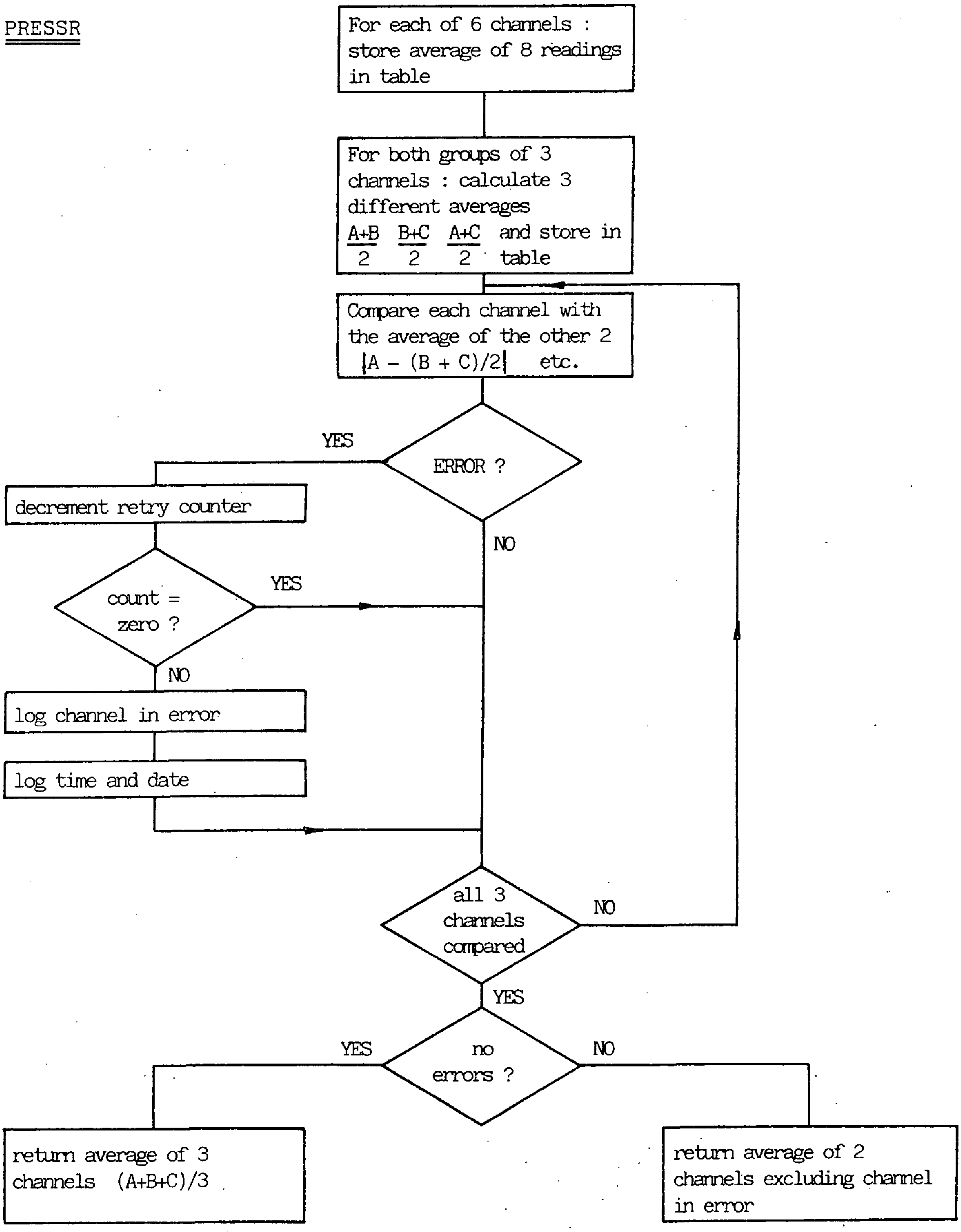

PRESSR
For each of 6 channels : store average of 8 readings in table
For both groups of 3 channels : calculate 3 different averages
$\frac{A+B}{2}$ $\frac{B+C}{2}$ $\frac{A+C}{2}$ and store in table
Compare each channel with the average of the other 2
$|A - (B + C)/2|$ etc.
ERROR ?
YES
NO
decrement retry counter
count = zero ?
YES
NO
log channel in error
log time and date
all 3 channels compared
NO
YES
no errors ?
YES
NO
return average of 3 channels (A+B+C)/3
return average of 2 channels excluding channel in error

FIGURE 45 Flow chart of module PRESSR

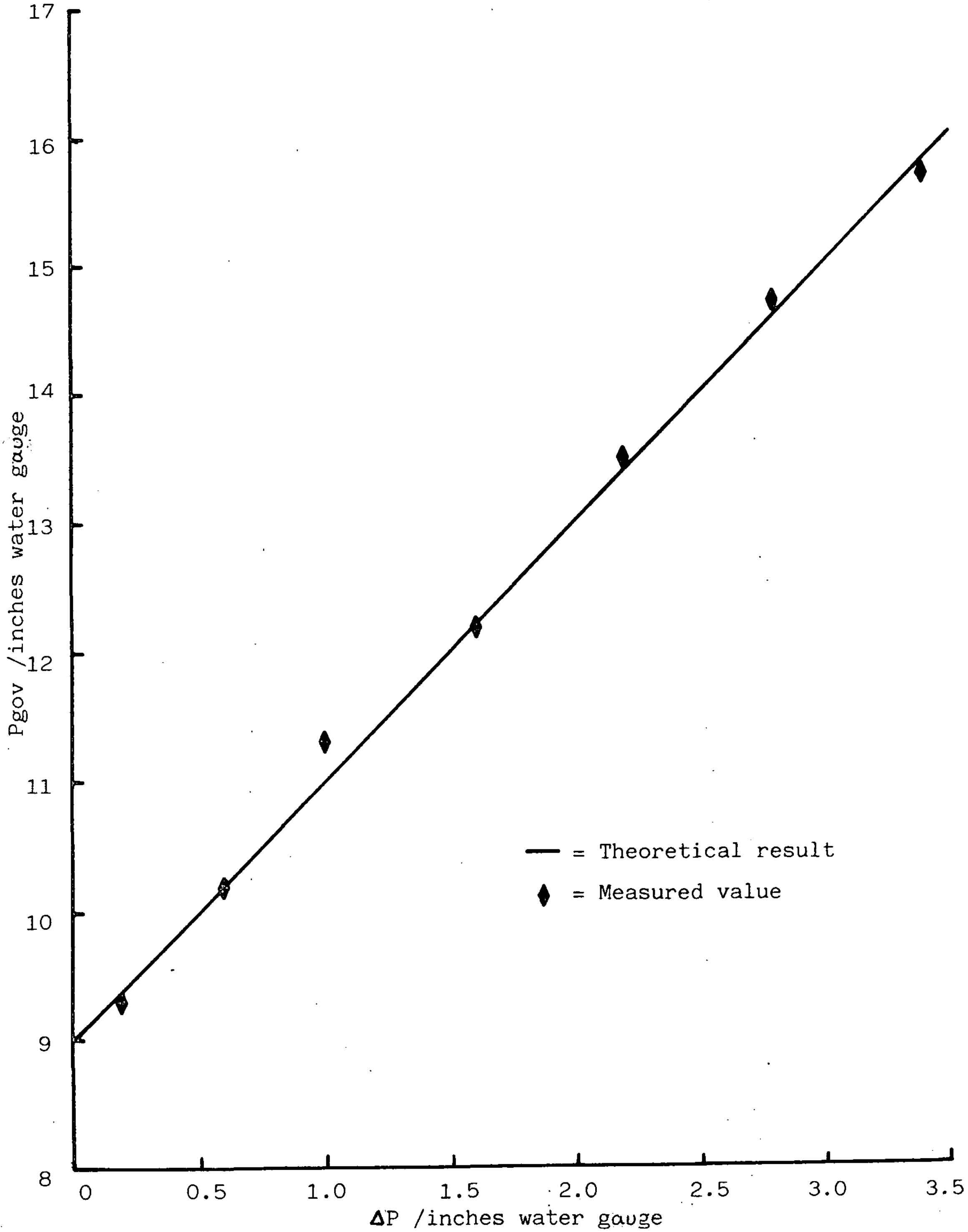


FIGURE 46 Graph of governor outlet pressure versus orifice plate differential

```
DATE 15:07 HR 12:11
VECTORED RECOVERY             Message 1
SYND=FA  RETRIES=01
SYNC ERR CHANNEL 01

DATE 15:07 HR 13:15
VECTORED RECOVERY             Message 2
SYND=DE  RETRIES=FF
CPU FAIL CHANNEL 01

DATE 15:07 HR 13:17           Message 3
TRANS RAM ERR S=10

DATE 15:07 HR 13:26           Message 4
RAM FAILURE S=10

DATE 15:07 HR 13:18           Message 5
RAM HWARE FAIL S=00

DATE 15:07 HR 13:17           Message 6
CBIT STUCK DBE S=20

TOTAL RAM FAILURE             Message 7

DATE 15:07 HR 13:22
VECTORED RECOVERY             Message 8
TRAP WDOG ADR=0639

DATE 15:07 HR 13:26
EPROM SET 00                  Message 9
FULL SYSTEM RESET

DATE 15:07 HR 13:24
EPROM SET 01                  Message 10
FULL SYSTEM RESET

DATE 15:07 HR 13:19
VECTORED RECOVERY             Message 11
SNAKE RST7 ADR=003D

DATE 15:07 HR 13:30
SOLENOID FAILURE              Message 12

DATE 15:07 HR 13:16           Message 13
PRESSR ERROR CH 03
```

FIGURE 47 Error messages

TABLE 1 Failure mechanisms and activation energies in MOS semiconductors

Failure mechanism	Type	Activation energy eV	Detection	Preventive measures
slow trapping	wearout	1.0	high temp. bias	ultra clean processing
contamination	wearout/infant	1.4	high temp. bias	ultra clean processing
surface charge	wearout	0.5 - 1.0	high temp. bias	ultra clean processing
polarisation	wearout	1.0	high temp. bias	eliminate phosphorus in gate oxide
electromigration	wearout	1.0	high temp. operating life	$J < 10^5$ A/cm^2
microcracks	random	-	temperature cycling	contoured oxide steps
contacts	wearout/infant	-	high temp. operating	ultra clean processing
oxide defects	infant/random	0.3	high voltage operating life and cell stress	ultra clean processing
aluminium corrosion	wearout	0.8	leak detection/ moisture tests	improve packaging
galvanic bond pad corrosion in plastic encaps.	wearout	0.6	leak detection/ moisture tests	improve packaging
charge loss from N channel EPROM	wearout	0.8	high temp. tests	improved design

TABLE 2 A comparison of resistor failure rates

Source		Failure rate f/10^6 hrs.
ICL data	reference(68)	0.004
Dummer	reference(69)	0.015
NCSR	reference(34)	0.007
CNET	reference(38)	0.0035
MIL 217D	reference(37)	0.0035
MIL 217C	reference(36)	0.0035

Failure modes :

90% open circuit
10% short circuit

Failure rates calculated for an oxide film resistor in a computer environment with $R < 100k$
Temperature = 25°C and stress = 0.1 (operating wattage/rated wattage)

TABLE 3 A comparison of capacitor failure rates

Source		Failure rate $f/10^6$ hrs.
ICL data	reference(68)	0.003
Dummer	reference(69)	0.08
NCSR	reference(34)	0.021
CNET	reference(38)	0.0035
MIL 217D	reference(37)	0.004
MIL 217C	reference(36)	0.004

Failure modes :

50% open circuit
50% short circuit

Failure rates calculated for a 0.1uF polycarbonate capacitor in a computer environment. With Tamb.= 25°C and stress = 0.1 (operating voltage/rated)

TABLE 4 A comparison of soldered joints failure rates

Source		Failure rate $f/10^6$ hrs.
ICL data	reference(68)	0.002
Dummer	reference(69)	0.008
CNET	reference(38)	0.0005
MIL 217D	reference(37)	0.0026
MIL 217C	reference(36)	0.0026

TABLE 5 A comparison of wire-wrap joint failure rates

Source		Failure rate $f/10^6$ hrs.
ICL data	reference(68)	0.0008
Dummer	reference(69)	0.0007
CNET	reference(38)	0.00001
MIL 217D	reference(37)	0.0000025
MIL 217C	reference(36)	0.0000025

TABLE 6 A comparison of edge connector failure rates

Source		Failure rate f/10^6 hrs.
ICL data	reference(68)	0.0030
Dummer	reference(69)	0.14
CNET	reference(38)	0.0030
MIL 217D	reference(37)	0.0001
MIL 217C	reference(36)	0.0001

Failure rates calculated for a mating pair of contacts connected once only without repetitive mating/un-mating.

MIL 217 and CNET failure rates calculated for a mating pair of contacts within a 64 way connector of material type B.

TABLE 7 Activation energies used in failure rate prediction

Technology	Package type	Activation energy eV		
		MIL 217D	MIL 217C	NCSR
TTL HTTL	Hermetic	0.40	0.40	0.40
DTL ECL	non hermetic	0.45	0.40	0.40
LTTL STTL	hermetic	0.45	0.40	0.40
	non hermetic	0.50	0.40	0.40
LSTTL	hermetic	0.50	0.40	0.50
	non hermetic	0.55	0.40	0.50
I^2L MNOS	hermetic	0.60	0.40	-
	non hermetic	0.80	0.40	-
PMOS	hermetic	0.50	0.70	0.55
	non hermetic	0.70	0.70	0.55
NMOS CCD	hermetic	0.55	0.70	0.55
	non hermetic	0.80	0.70	0.55
CMOS,CMOS/SOS	hermetic	0.65	0.70	0.55
and linear	non hermetic	0.90	0.70	0.55

TABLE 8 A comparison of TTL integrated circuit failure rates

Source		Failure rate $f/10^6$ hrs.	Ea eV
ICL data	reference(68)	0.014	-
NCSR	reference(34)	0.031	0.4
CNET	reference(38)	0.050	0.3 & 1.0
MIL 217D	reference(37)	0.084	0.45
MIL 217C	reference(36)	0.533	0.40

Failure rates calculated for a plastic package containing 4 gates and in a computer environment with Tamb.= 25°C and Tjunct.= 33°C

TABLE 9 A comparison of the failure rates for a 6800 Microprocessor

Source		Failure rate $f/10^6$ hrs.	Ea eV	Tj °C
Motorola	reference(39)	0.30	1.0	105
MIL 217D	reference(37)	7.80	0.55	120
MIL 217C	reference(36)	1380	0.70	120
CNET	reference(38)	6.60	0.3 & 1.0	120
NCSR	reference(34)	6.40	0.55	130

Failure rates calculated for a hermetic package containing 1367 gates in a "ground fixed" environment at Tamb.=70°C
θja specified by reliability data source
Pdiss = worst case = 1W (except for Motorola = 0.5W typical)
Quality level = C1 (approximately computer grade)

TABLE 10 A comparison of 6800 Microprocessor adjusted failure rates

Source	Failure rate f/10^6 hrs.
Motorola reference(39)	0.013
MIL 217D reference(37)	6.60
MIL 217C reference(36)	280
CNET reference(38)	1.3 **
NCSR reference(34)	4.01

Failure rates converted to a common base of :

Tamb. = 45°C
θja = 50°C/W
Pdiss = 0.5W (typical)
Tj = 70°C
Ea = 1.0eV
Quality level = C1

** Since CNET data uses both 0.3 and 1.0eV activation energies, it is not possible to convert to a single activation energy of 1.0eV, however the junction temperature is adjusted from 120°C to 70°C.

TABLE 11 A comparison of the failure rates for a 8080 Microprocessor

Source	Failure rate f/10^6 hrs.	Ea eV	Tj °C
Intel reference(32)	0.12	0.50	89 typ.?
MIL 217D reference(37)	4.2	0.55	106
MIL 217C reference(36)	510	0.70	106
CNET reference(38)	3.8	0.3 & 1.0	106
NCSR reference(34)	10.8	0.55	145
RRE	36.2	-	-

Failure rates calculated for a hermetic package containing 1100 gates in a "ground fixed" environment at Tamb.= 55°C

θja specified by reliability data source
Pdiss = worst case
Quality level = C1 (approximately computer grade)

TABLE 12 A comparison of 8080 Microprocessor adjusted failure rates

Source		Failure rate f/10^6 hrs.
Intel	reference(32)	0.01
MIL 217D	reference(37)	23.3
MIL 217C	reference(36)	833
CNET	reference(38)	2.0 **
NCSR	reference(34)	13.9

Failure rates converted to a common base of :

Tamb. = 45°C
θja = 50°C/W
Pdiss = 0.78W (typical)
Tj = 84°C
Ea = 1.0eV
Quality level = C1

** Since CNET data uses both 0.3 and 1.0eV activation energies, it is not possible to convert to a single activation energy of 1.0eV, however the junction temperature is adjusted from 106°C to 84°C.

TABLE 13 A comparison of the failure rates of a 2716 EPROM

Source		Failure rate f/10^6 hrs.	Ea eV
Intel	reference(29)	0.45 @ 60% C.L.	0.30
MIL 217D	reference(37)	3.5	0.55
MIL 217C	reference(36)	34.5	0.70
CNET	reference(38)	0.64	0.3 & 1.0
NCSR	reference(34)	1.09	0.55

Failure rates calculated with :

Tamb. = 55°C
θja = 25°C/W
Pdiss = 0.525W
Tj = 68°C
Quality level C1
"ground fixed"environment
hermetic encapsulation

TABLE 14 A comparison of the failure rates of a 1k Bipolar ROM

Source		Failure rate f/10^6 hrs.	Ea eV
Intel	reference(30)	0.5 @ 90% C.L.	0.40
MIL 217D	reference(37)	0.85	0.45
MIL 217C	reference(36)	4.90	0.40
CNET	reference(38)	0.31	0.3 & 1.0
NCSR	reference(34)	0.34	0.40

Failure rates calculated with :

Tamb. = 85°C
θja = 30°C/W
Pdiss = 0.5W
Tj = 100°C
Quality level = C1
"ground fixed" environment
hermetic encapsulation

TABLE 15 A comparison of the failure rates of a 16k Dynamic RAM

Source		Failure rate f/10^6 hrs.	Ea eV
Intel	reference(28)	0.27 @ 60% C.L.	0.30
MIL 217D	reference(37)	8.0	0.55
MIL 217C	reference(36)	261	0.70
CNET	reference(38)	0.84	0.30 & 1.0
NCSR	reference(34)	4.9	0.55
Motorola	reference(40)	0.083 @ 60% C.L.	1.0
Motorola converted to Ea = 0.3eV	reference(40)	1.5 @ 60% C.L.	0.3

Failure rates calculated with :

Tamb. = 70°C
θja = 30°C/W
Pdiss = 0.4W
Tj = 82°C
"ground fixed"environment
hermetic encapsulation
Quality level = C1

TABLE 16 A comparison of the failure rates of a 1k Static RAM

Source		Failure rate $f/10^6$ hrs.	Ea eV
Intel	reference(31)	0.4 @ 90% C.L.	0.30
MIL 217D	reference(37)	0.68	0.55
MIL 217C	reference(36)	13.2	0.70
CNET	reference(38)	0.22	0.3 & 1.0
NCSR	reference(34)	0.57	0.55

Failure rates calculated with :

Tamb. = 55°C
θja = 30°C/W
Pdiss = 0.2W
Tj = 61°C
Quality level = C1
"ground fixed" environment
hermetic encapsulation

TABLE 17 The relation between cost, screening level and Quality factor

MIL 217 screen level	Screening method	π_{φ}	Typical relative cost
S	MIL-M-38510 Class S	0.5	8 - 20
S	BS9400 Class A **	0.5	
B	MIL-M-38510 Class B	1.0	
B	BS9400 Class B	1.0	
B-1	MIL-STD-883 method 5004 Class B	3.0	4 - 6
B-2	Vendor equivalent of B-1	6.5	
C	MIL-M-38510 Class C	8.0	
C	BS9400 Class C	8.0	
-	BS9400 Class D	10.0	2 - 4
C-1	Vendor equivalent of C	13.0	
C-1	BS9400 Full Assessment level	13.0	
D	Commercial Hermetic	17.5	1.0
D-1	Commercial plastic	35.0	0.5

** Considerable differences exist in screening specification between MIL-M-38510 Class S and BS9400 Class A.

BS9400 Quality factors are those suggested in reference(70)

TABLE 18 Recommended maximum junction temperatures for Semiconductors

Semiconductor device		Maximum junction temperature
Transistors (Si)	Hermetic	100°C
general purpose	Plastic	80°C
Transistors (Si)	Hermetic	110°C
power	plastic	90°C
Diodes		100°C
Linear I.C.s	Hermetic	90°C
	Plastic	70°C
Digital I.C.s	Hermetic	100°C
TTL	Plastic	70°C
Digital I.C.s	Hermetic	90°C
CMOS	Plastic	70°C

TABLE 19 Signals carried by back-plane Bus

	Bottom connector		Top connector	
Pin No.	A	C	A	C
1	OV	OV	-12V	-12V
2	D7(out)	D7(in)		
3	D6(out)	D6(in)	+5V	OV
4	D5(out)	D5(in)	+5V	OV
5	D4(out)	D4(in)	+5V	OV
6	D3(out)	D3(in)	+5V	OV
7	D2(out)	D2(in)	+28V	
8	D1(out)	D1(in)	+28V	
9	DO(out)	DO(in)		
10	SOD	$\overline{\text{RESET}}$		
11	CLK OUT	CLK OUT÷10		
12	ALE	SID		
13	IO/$\overline{\text{M}}$			
14	$\overline{\text{WR}}$			
15	$\overline{\text{RD}}$			
16	A15	INTR		
17	A14	RST 5.5		
18	A13	RST 6.5		
19	A12	RST 7.5		
20	A11	TRAP		
21	A10	ROM SELECT	PORT 7(WR)	PORT 7(RD)
22	A9		PORT 6(WR)	PORT 6(RD)
23	A8		PORT 5(WR)	PORT 5(RD)
24	A7		PORT 4(WR)	PORT 4(RD)
25	A6		PORT 3(WR)	PORT 3(RD)
26	A5		PORT 2(WR)	PORT 2(RD)
27	A4		PORT 1(WR)	PORT 1(RD)
28	A3		PORT O(WR)	PORT O(RD)
29	A2			
30	A1			SERIAL IN
31	AO			SERIAL OUT
32	+5V	+5V	+12V	+12V

TABLE 20 Correction bits for SEC/DED Hamming code

C3	C2	C1	C0	Bit in error
0	0	0	0	No error
0	0	0	1	C0
0	0	1	0	C1
0	0	1	1	DBE
0	1	0	0	C2
0	1	0	1	DBE
0	1	1	0	DBE
0	1	1	1	D0
1	0	0	0	C3
1	0	0	1	DBE
1	0	1	0	DBE
1	0	1	1	D1
1	1	0	0	DBE
1	1	0	1	D2
1	1	1	0	D3
1	1	1	1	DBE

Horizontal parity =

ODD for single bit error (SBE)

EVEN for double bit error (DBE)

TABLE 21 Error position as indicated by error flags

S2	S1	S0	E	Error position
0	0	0	0	No error
0	0	0	1	RAM 0
0	0	1	1	RAM 1
0	1	0	1	RAM 2
0	1	1	1	RAM 3
1	0	0	1	RAM 4
1	0	1	1	RAM 5
1	1	0	1	RAM 6
1	1	1	1	RAM 7
0	0	1	0	Double bit error

APPENDIX 1 Circuit diagram of power supply

The circuit diagram of the controller power supply is given in figure(A1) below.

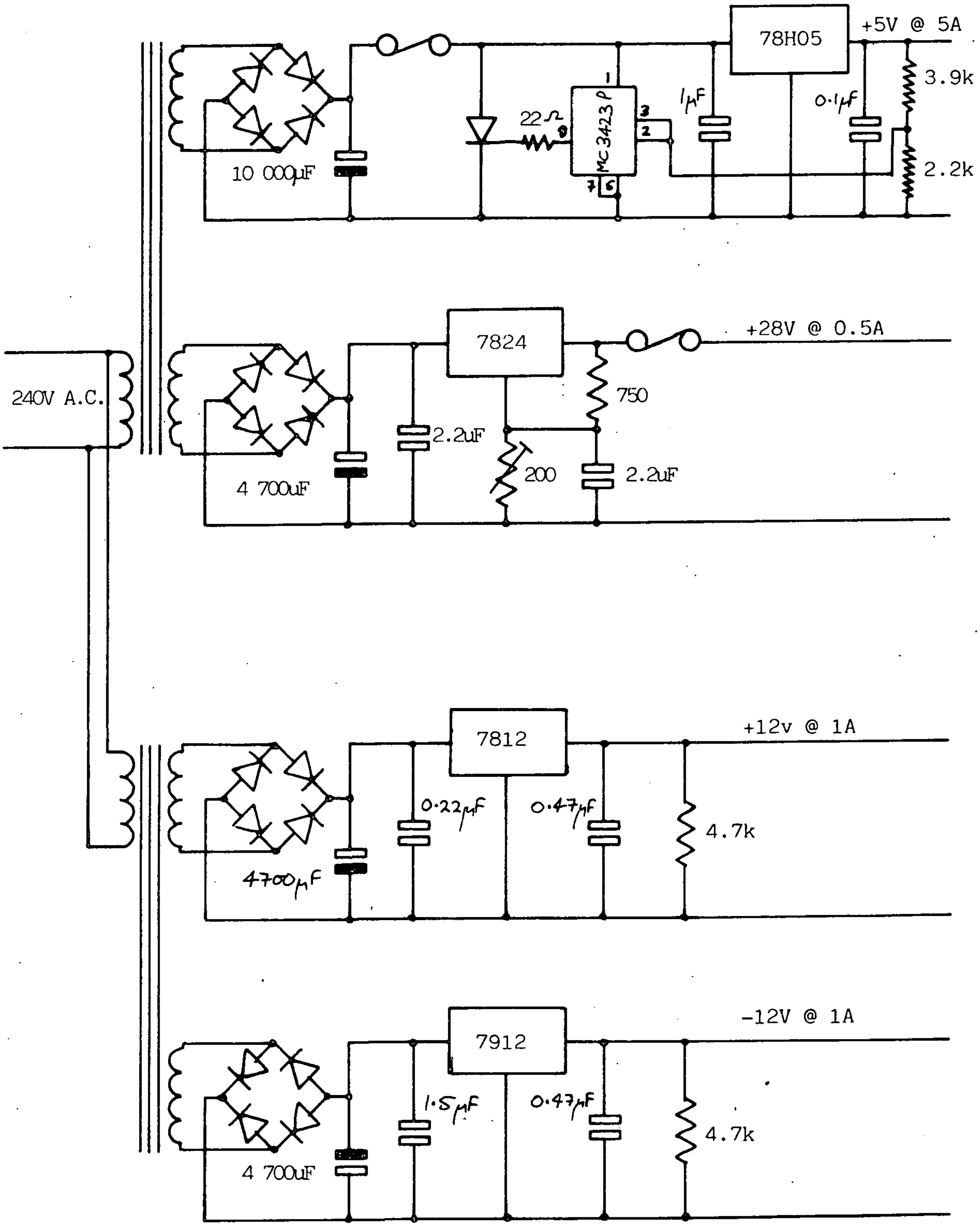


FIGURE A1

APPENDIX 2 Circuit board layout

The layout of the three governor controller circuit boards is given in the following figures A2-A4.

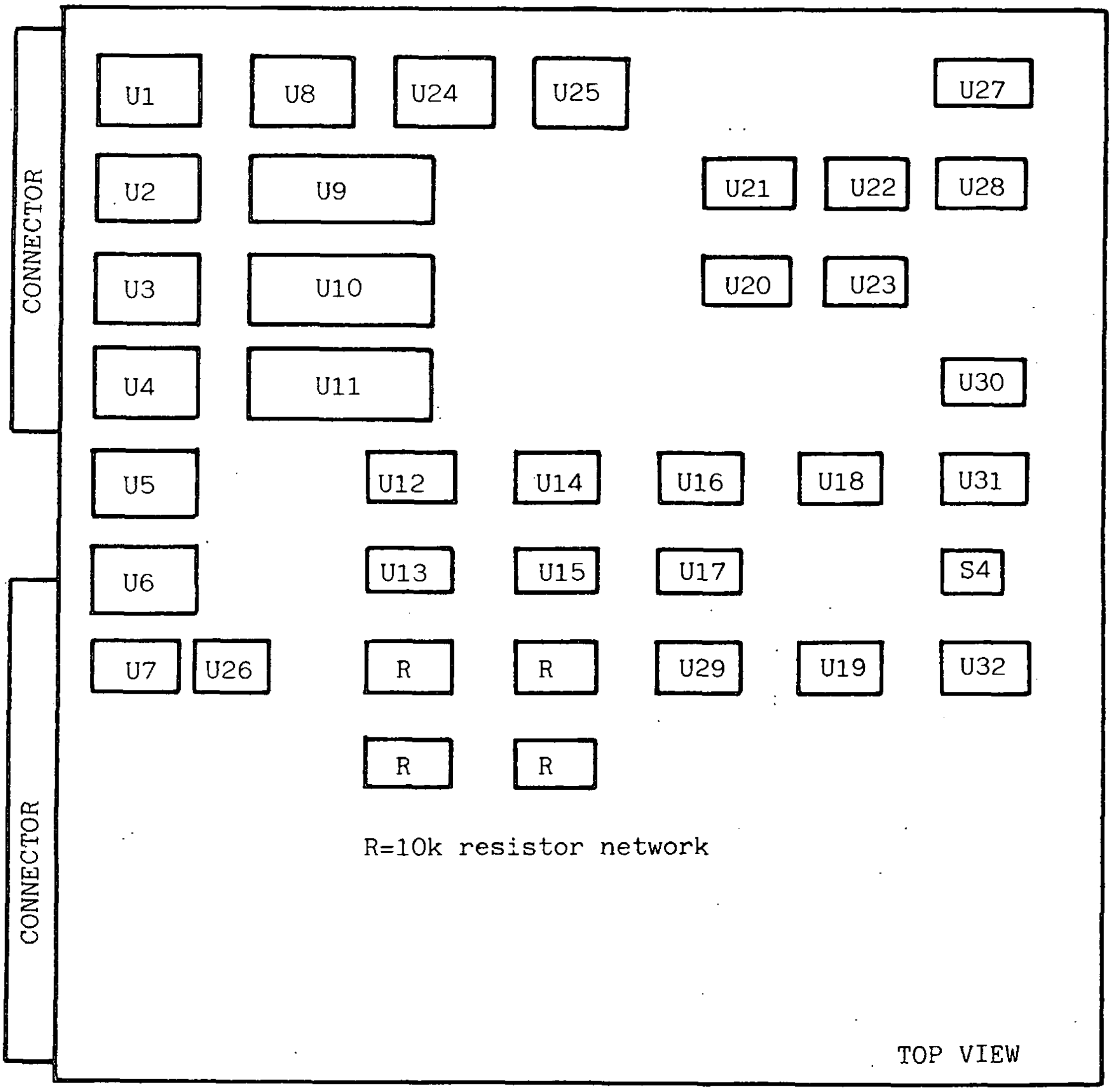


FIGURE A2 Microprocessor board layout

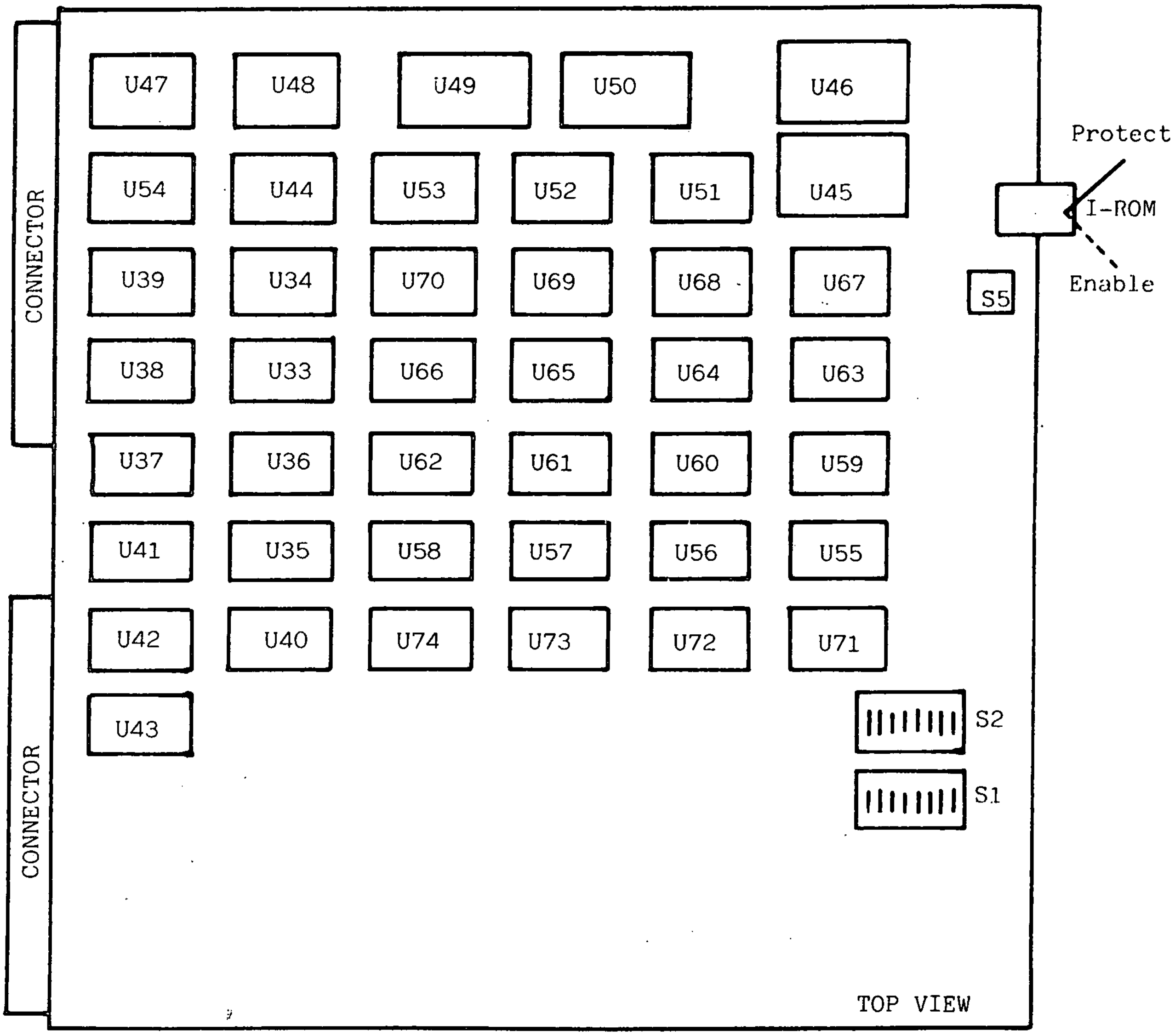


FIGURE A3 Memory board layout

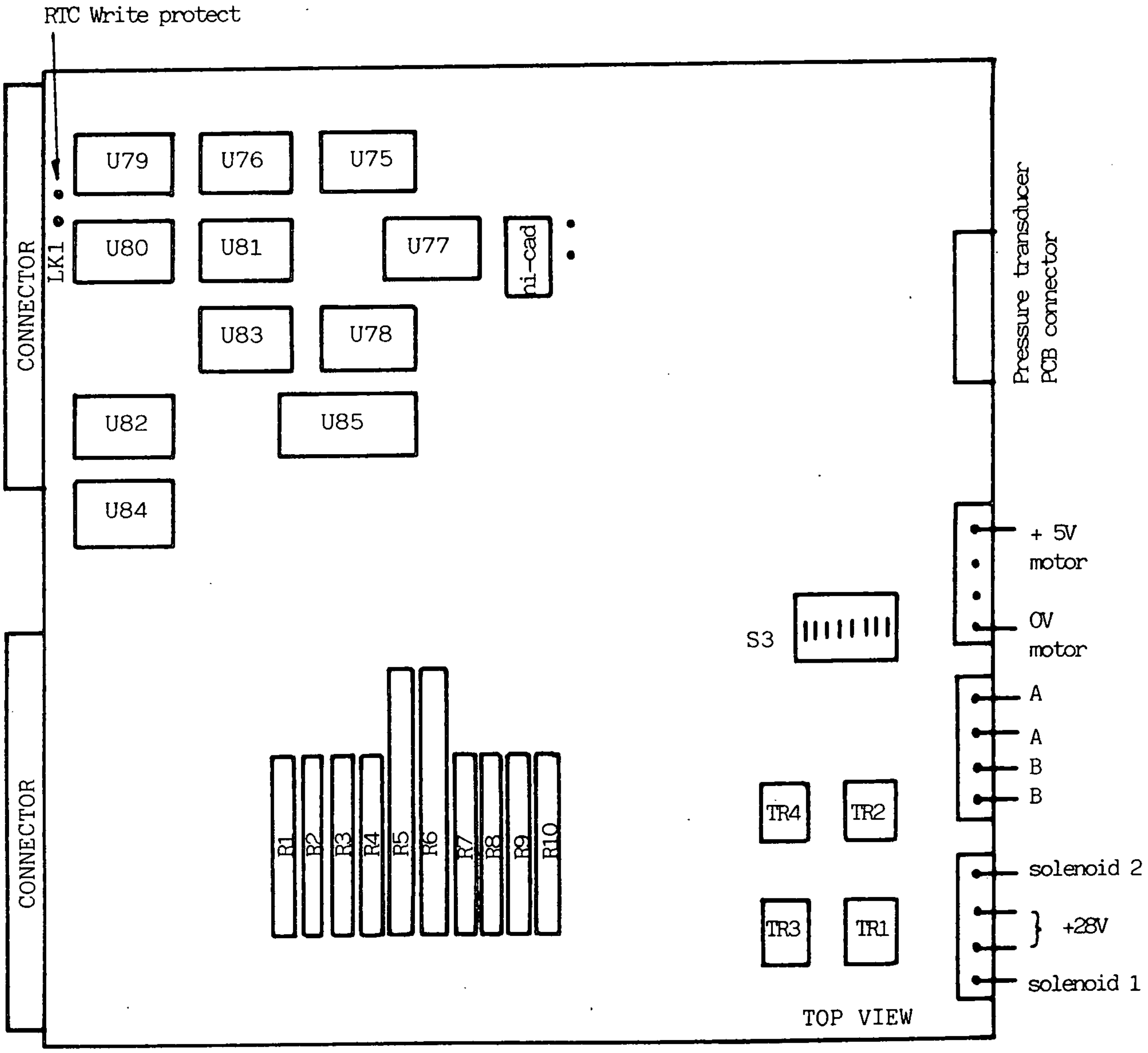


FIGURE A4 Input/output board layout

APPENDIX 3.1

ERROR FLAGS LOW PASS FILTER CALCULATION

Although the three processors are fed by a common clock, they will not run in exact synchronism because of different internal delays. Consideration of the 8085 timing given in reference(33) shows that a maximum difference of 140ns is possible between processors on the address bus. Any difference between the three channels will cause transient dips in the error flags which must be smoothed out by the low pass filters. The 50ns rise time of the error flags must be added to the 140ns giving a 190ns maximum dip in the error flags. The voter flags must not be allowed to dip below the logic '1' level of 2.0V for a 250ns interruption which includes a safety margin added to the 190ns.

Hence the time constant of the LPF is calculated by :

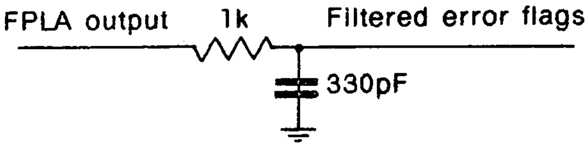
$$V = Vo \exp -(t/CR) \qquad \text{where : } \quad \begin{array}{l} Vo = 5V \\ V = 2V \end{array}$$

$$\Rightarrow 2 = 5 \exp -(250ns/CR)$$

$$\Rightarrow CR = 270ns$$

Using R=1k ±5% this gives C=285pF for Rmin.=0.95k. Since the capacitor tolerance is ±10%, a 330pF capacitor is used.

The low pass filter therefore consists of :



APPENDIX 3.2 FPLA Programming Table

The following table will implement a 5 channel TMR voter including error flags when programmed into the FPLA.

BIPOLAR FIELD PROGRAMMABLE LOGIC ARRAY (16X48X8) **82S100 (T.S.)/82S101 (O.C.)**

82S100-I,N • 82S101-I,N

16X48X8 FPLA PROGRAM TABLE

THIS PORTION TO BE COMPLETED BY SIGNETICS

CF (XXXX) ____

CUSTOMER SYMBOLIZED PART # ____

DATE RECEIVED ____

COMMENTS ____

CUSTOMER NAME J.C.Pearson Fault-tolerant controller

PURCHASE ORDER # ____

SIGNETICS DEVICE # N82S101N

TOTAL NUMBER OF PARTS 6

PROGRAM TABLE # 5 Channel TMR voter

REV 2 DATE 22 - 4 - 81

PROGRAM TABLE ENTRIES						
INPUT VARIABLE			OUTPUT FUNCTION		OUTPUT ACTIVE LEVEL	
Im	$\overline{Im}$	Don't Care	Prod. Term Present in Fp	Prod. Term Not Present in Fp	Active High	Active Low
H	L	— (dash)	A	• (period)	H	L
NOTE Enter (—) for unused inputs of used P-terms.			NOTES 1. Entries independent of output polarity. 2. Enter (A) for unused outpbuts of used P-terms.		NOTES 1 Polarity programmed once only 2 Enter (H) for all unused outputs	

	PRODUCT TERM*																ACTIVE LEVEL*							
	INPUT VARIABLE*																L	L	L	H	H	H	H	H
																	OUTPUT FUNCTION*							
NO.	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	H	H	.	.	.	.	.	.	.	A
1	–	–	–	–	–	–	–	–	–	–	–	–	–	H	H	–	.	.	.	.	.	.	.	A
2	–	–	–	–	–	–	–	–	–	–	–	–	–	H	–	H	.	.	.	.	.	.	.	A
3	–	–	–	–	–	–	–	–	–	–	–	H	H	–	–	–	.	.	.	.	.	.	A	.
4	–	–	–	–	–	–	–	–	–	–	H	H	–	–	–	–	.	.	.	.	.	.	A	.
5	–	–	–	–	–	–	–	–	–	–	H	–	H	–	–	–	.	.	.	.	.	.	A	.
6	–	–	–	–	–	–	–	–	H	H	–	–	–	–	–	–	.	.	.	.	.	A	.	.
7	–	–	–	–	–	–	–	H	H	–	–	–	–	–	–	–	.	.	.	.	.	A	.	.
8	–	–	–	–	–	–	–	H	–	H	–	–	–	–	–	–	.	.	.	.	.	A	.	.
9	–	–	–	–	–	H	H	–	–	–	–	–	–	–	–	–	.	.	.	.	A	.	.	.
10	–	–	–	–	H	H	–	–	–	–	–	–	–	–	–	–	.	.	.	.	A	.	.	.
11	–	–	–	–	H	–	H	–	–	–	–	–	–	–	–	–	.	.	.	.	A	.	.	.
12	–	–	H	H	–	–	–	–	–	–	–	–	–	–	–	–	.	.	.	A	.	.	.	.
13	–	H	H	–	–	–	–	–	–	–	–	–	–	–	–	–	.	.	.	A	.	.	.	.
14	–	H	–	H	–	–	–	–	–	–	–	–	–	–	–	–	.	.	.	A	.	.	.	.
15	–	–	–	–	–	–	–	–	–	–	–	–	–	–	L	H	.	A	A	.	.	.	.	.
16	–	–	–	–	–	–	–	–	–	–	–	–	–	–	H	L	.	A	A	.	.	.	.	.
17	–	–	–	–	–	–	–	–	–	–	–	–	–	L	H	–	A	.	A	.	.	.	.	.
18	–	–	–	–	–	–	–	–	–	–	–	–	–	H	L	–	A	.	A	.	.	.	.	.
19	–	–	–	–	–	–	–	–	–	–	–	L	H	–	–	–	.	A	A	.	.	.	.	.
20	–	–	–	–	–	–	–	–	–	–	–	H	L	–	–	–	.	A	A	.	.	.	.	.
21	–	–	–	–	–	–	–	–	–	–	L	H	–	–	–	–	A	.	A	.	.	.	.	.
22	–	–	–	–	–	–	–	–	–	–	H	L	–	–	–	–	A	.	A	.	.	.	.	.
23	–	–	–	–	–	–	–	–	L	H	–	–	–	–	–	–	.	A	A	.	.	.	.	.
24	–	–	–	–	–	–	–	–	H	L	–	–	–	–	–	–	.	A	A	.	.	.	.	.
25	–	–	–	–	–	–	–	L	H	–	–	–	–	–	–	–	A	.	A	.	.	.	.	.
26	–	–	–	–	–	–	–	H	L	–	–	–	–	–	–	–	A	.	A	.	.	.	.	.
27	–	–	–	–	–	L	H	–	–	–	–	–	–	–	–	–	.	A	A	.	.	.	.	.
28	–	–	–	–	–	H	L	–	–	–	–	–	–	–	–	–	.	A	A	.	.	.	.	.
29	–	–	–	–	L	H	–	–	–	–	–	–	–	–	–	–	A	.	A	.	.	.	.	.
30	–	–	–	–	H	L	–	–	–	–	–	–	–	–	–	–	A	.	A	.	.	.	.	.
31	–	–	L	H	–	–	–	–	–	–	–	–	–	–	–	–	.	A	A	.	.	.	.	.
32	–	–	H	L	–	–	–	–	–	–	–	–	–	–	–	–	.	A	A	.	.	.	.	.
33	–	L	H	–	–	–	–	–	–	–	–	–	–	–	–	–	A	.	A	.	.	.	.	.
34	–	H	L	–	–	–	–	–	–	–	–	–	–	–	–	–	A	.	A	.	.	.	.	.
35																								
36																								
37																								
38																								
39																								
40																								
41																								
42																								
43																								
44																								
45																								
46																								
47																								

* Input and Output fields of unused P-terms can be left blank. Unused inputs and outputs are FPLA terminals left floating

Appendix 4

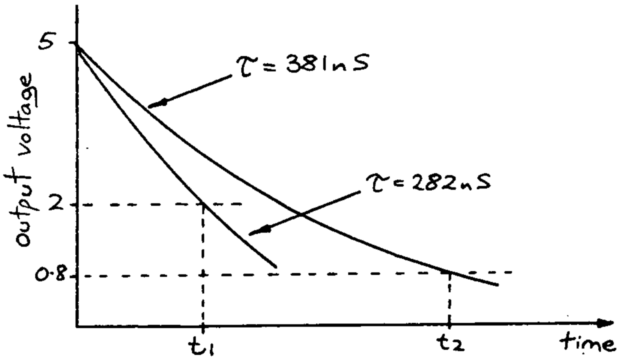
CALCULATION OF DELAY BEFORE LATCHING ERROR FLAGS

Component tolerances in the error flag low pass filters will cause their outputs to change state at different times. therefore the error flags should not be latched until they have all had time to settle to their final value. The outputs of the filters will decay exponentially and. considering the extremes of component tolerances. will have time constants between the limits :

tmin.= 0.95k x 297pF = 282ns

tmax.= 1.05k x 363pF = 381ns

The graph below shows the exponential decay of the outputs plotted for the two extremes of time constant.



The TTL logic thresholds of '1'= > 2.0V and '0'= < 0.8V leave an indeterminate range of 0.8 - 2.0V. Taking the worst case. the voting error is detected at time t1 when the output. having a time constant of 282ns. drops below 2V. However the other error flags must be allowed time to settle to the 0.8V level at a rate governed by the 381ns time constant. This level is reached at time t2. The difference t2 - t1 is the maximum possible delay between detecting an error and allowing the error flags to settle.

t1 = 282 ln 2.5 = 258ns

t2 = 381 ln 6.25 = 698ns

Hence : t2 - t1 = 440ns

Hence the monostable U18b is designed to give a delay of 440ns between detecting a voting error and latching the error flags.

APPENDIX 5 Encoding ROM data

```
0000    00 17 2B 3C 4D 5A 66 71 8E 99 A5 B2 C3 D4 E8 FF
0010    00 17 2B 3C 4D 5A 66 71 8E 99 A5 B2 C3 D4 E8 FF
```

The above table is programmed into a 32 x 8 PROM and is used to encode data into the SEC/DED Hamming code. Four-bit wide data is encoded as four data bits and four check bits.

APPENDIX 6 Decoding ROM data

```
0000    00 01 03 02 05 02 02 19 07 02 02 2B 02 4D 8F 02
0010    09 12 12 15 12 13 11 10 12 9F 5D 12 3B 12 12 17
0020    0B 22 22 27 22 AF 6D 22 22 23 21 20 39 22 22 25
0030    32 7D BF 32 37 32 32 1B 35 32 32 29 30 31 33 32
0040    0D 42 42 CF 42 47 6B 42 42 45 59 42 41 40 42 43
0050    52 7B 57 52 DF 52 52 1D 53 52 50 51 52 49 55 52
0060    62 79 65 62 63 62 60 61 EF 62 62 2D 62 4B 67 62
0070    71 70 72 73 72 75 69 72 72 77 5B 72 3D 72 72 FF
0080    0F 82 82 CD 82 AB 87 82 82 99 85 82 83 82 80 81
0090    92 97 BB 92 DD 92 92 1F 91 90 92 93 92 95 89 92
00A0    A2 A5 B9 A2 A1 A0 A2 A3 ED A2 A2 2F A2 A7 8B A2
00B0    B3 B2 B0 B1 B2 A9 B5 B2 B2 9B B7 B2 3F B2 B2 FD
00C0    C2 C3 C1 C0 D9 C2 C2 C5 EB C2 C2 C7 C2 4F 8D C2
00D0    D5 D2 D2 C9 D0 D1 D3 D2 D2 9D 5F D2 D7 D2 D2 FB
00E0    E7 E2 E2 CB E2 AD 6F E2 E0 E1 E3 E2 E5 E2 E2 F9
00F0    F2 7F BD F2 DB F2 F2 F7 E9 F2 F2 F5 F2 F3 F1 F0
```

The above table is programmed into a 256 x 8 PROM and is used to decode data from the SEC/DED Hamming code. Eight-bit wide data, consisting of four data bits and four check bits, is decoded as four data bits and four error flags. Single bit errors in the data are corrected and the error flags signal the occurrence and position of single and double bit errors.

APPENDIX 7 Operation of the Real Time Clock (Extract from the Radiospares data sheet)

The RS 58174 is a metal gate CMOS circuit that functions as a real time clock and calendar in bus-orientated microprocessor systems. An interrupt timer is included, which can be programmed to have one of three times. The time base is obtained from a RS 32.768 kHz crystal with time keeping down to 2.2V for low power standby operation from batteries.

Application

When the system is powered up it is necessary to enter the correct data into the device registers and start the clock running. The seconds, minutes, hours, days and months counters are all parallel loaded with data from the 4 bit data bus when correctly addressed, $\overline{CS}$ is low and a write data strobe pulse is given. Data to be entered is set up on the 4 bit data bus; the address from Table 1 for the required register is set on the 4 bit address bus and a write data strobe pulse is sent. Chip select must be low during write and read operations. All information is entered in the same way with the relevant address. To start the clock running at the required instant a logic 1 is written to DB0 at address 14, likewise writing a 0 stops the clock. Data can be read from a register by using the required address as in Table 1 and applying a read strobe pulse. The data becomes available on the 4 bit data bus: chip select must be low for this read operation.

The internal counters are arranged as bytes of four bits each. When a byte is addressed it will appear on the data I/O bus enabling independent access. For bytes which do not contain 4 bits (e.g. week days use only 3 bits) the unused bits are not recognised during a write operation and tied to V_{SS} during a read operation. The addressable reset latch holds tenths, units and tens of seconds in a reset condition. If a register is updated during a read operation the I/O data is prevented from updating and a subsequent read will return the illegal b.c.d. code 1111. This allows detection that the previous data had changed and is now incorrect. The interrupt timer can be programmed for 0.5,5 or 60 second intervals and may be coded for single or repeated operation. The open drain interrupt output is pulled to V_{SS} when the timer times out and reading the interrupt register provides the status and internal selected information.

Standby mode

This is automatically selected when the supply voltage falls to the standby level. (2.2V minimum) with no read or write strobes.

Test mode

This mode is used in production testing of the circuit. For normal operation the circuit must be in non-test mode and set as part of the system initialisation. Non test mode is set by writing a logic 0 to DB3 at AD0.

Years Status Register

This is a 4 bit shift register which is shifted each year on the 31st December. The status register must be set in accordance with the table below and no readout capability is available.

	DB3	DB2	DB1	DB0
Leap year	1	0	0	0
Leap year + 1	0	1	0	0
Leap year + 2	0	0	1	0
Leap year + 3	0	0	0	1

TABLE 1

Selected counter	Address bits				Mode
	AD3	AD2	AD1	AD0	
0 Test only	0	0	0	0	Write only
1 Tenths of sec.	0	0	0	1	Read only
2 Units of secs.	0	0	1	0	Read only
3 Tens of secs.	0	0	1	1	Read only
4 Units of mins.	0	1	0	0	Read or Write
5 Tens of mins.	0	1	0	1	Read or Write
6 Units of hours	0	1	1	0	Read or Write
7 Tens of hours	0	1	1	1	Read or Write
8 Units of days	1	0	0	0	Read or Write
9 Tens of days	1	0	0	1	Read or Write
10 Day of week	1	0	1	0	Read or Write
11 Units of months	1	0	1	1	Read or Write
12 Tens of months	1	1	0	0	Read or Write
13 Years	1	1	0	1	Write only
14 Stop/Start	1	1	1	0	Write only
15 Interrupt and status	1	1	1	1	Read or Write

Appendix 8

DESIGN OF REDUNDANT SOLENOID DRIVER CIRCUITRY

The solenoid is rated as follows :

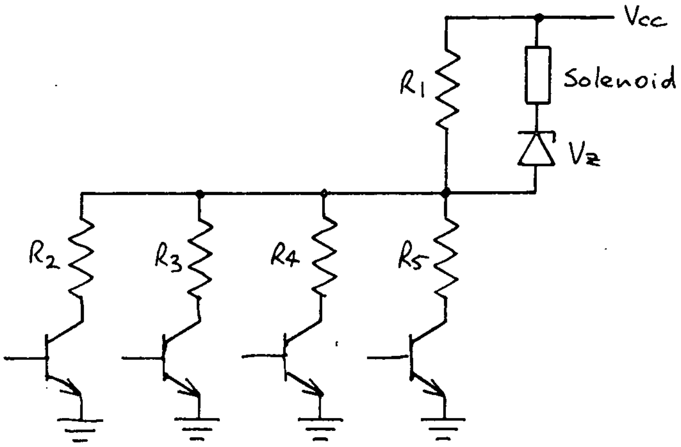
Vnormal = 24V

Vhold = 2.6V

Vpull-in = 8.0V

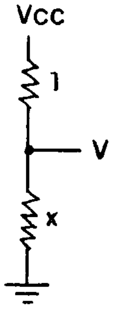
Resistance = 287Ω

An octal driver is used to control two solenoids and the circuit diagram for one solenoid using half of the driver is shown below :



The aim of the design is that any of the components excluding the solenoid can fail, whilst still being able to switch the solenoid ON and OFF. The resistor values R2 - R5 must be chosen to give the maximum possible voltage swing at their junction with R1.

The potential divider is simplified to :



With three transistors off and one failed short circuit. the voltage V1 is given by :

$$V1 = \frac{x}{1 + x} Vcc$$

Conversely, with three transistors ON and one failed open circuit, the voltage V2 is given by :

$$V2 = \frac{x/3}{1 + x/3} Vcc = \frac{x}{3 + x} Vcc$$

Hence :

$$\frac{\Delta V}{Vcc} = \frac{V1 - V2}{Vcc} = \frac{x}{1 + x} - \frac{x}{3 + x} = V'$$

For maximum ΔV $\quad \frac{dV'}{dx} = 0$

Hence : $\quad x = \sqrt{3} = 1.73$

$$Vmax. = 0.268\ Vcc$$

Vcc was chosen to be 28V which gives Vmax.= 7.5V

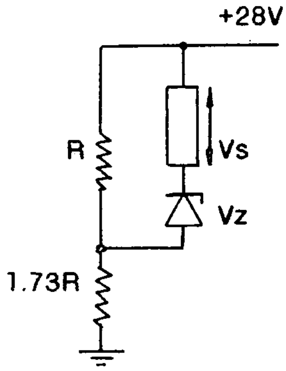
A sufficient swing to turn the solenoids ON and OFF is thus ensured by choosing Vcc= 28V, but this voltage is sufficiently low so as not to damage the solenoid due to overvoltage, should R1 fail.

The minimum acceptable V is given by :

$$Vmin.= Vpull\text{-}in - Vhold$$
$$= 8.0 - 2.6 = 5.4V$$

Therefore the voltage swing of 7.5V is 2V greater than necessary and the excess can be used to provide a safety margin. The safety margin will be reduced by the loading effect of the solenoid on the potential divider.

The zener diode is used to subtract a bias from the voltage swing in order to satisfy the solenoid ON and OFF limits. Considering the diagram below for three transistors OFF and one failed short circuit :



The voltage at the potential divider junction is given by :

$$V = \frac{1.73R}{R + 1.73R} \times 28 = 17.7V$$

To ensure that the solenoid is OFF, and including a safety margin of 0.5V, it is required that Vs= 2.1V.

Hence : $Vz = (28 - 17.7) - 2.1 = 8.2V$

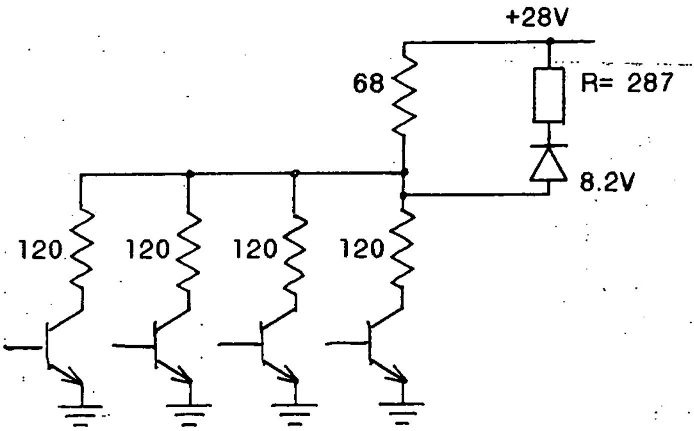
So an 8.2V zener diode is used.

The equivalent series resistance of the solenoid and zener diode is approximately 600Ω, so R1 was chosen to be about a tenth of this value so that the potential divider is not shunted too heavily by the load it is driving. The value chosen was R1= 68Ω and R2 - R5 are given by the relation :

$$R2 = 1.73 \times R1 = 1.73 \times 68 \approx 120\Omega$$

So R2 - R5 were chosen to be 120Ω .

The complete circuit design is now given by :



Using the above circuit, the voltages across the solenoid are checked. With three transistors ON the solenoid voltage = 8.7V and with three transistors OFF the solenoid voltage = 1.9V. This gives a 0.7V safety margin on both solenoid limits. The circuit should be set up as follows because of tolerance variations in the resistors and zener diode. The power supply is reduced from 28V until the solenoid ceases to switch ON and OFF using three of the transistors. The voltage is recorded and is equal to Vmin. The power supply is then increased above 28V until the solenoid again ceases to function correctly. This voltage is equal to Vmax. Finally the power supply is set mid-way between Vmin. and Vmax.

To protect against failure open circuit of the zener diode, two diodes are used in parallel. Failure short circuit of the zener diode is not fatal as long as no other components fail.

Consideration of the above circuitry shows that any component excluding the solenoid can fail, whilst still being able to switch the solenoid ON and OFF.

APPENDIX 9

List of Integrated Circuits

U1	82S101
U2	82S101
U3	82S101
U4	82S101
U5	82S101
U6	82S101
U7	1489
U8	8212 (1)
U9	8085 (1)
U10	8085 (2)
U11	8085 (3)
U12	8216 (1)
U13	8216 (1)
U14	8216 (2)
U15	8216 (2)
U16	8216 (3)
U17	8216 (3)
U18	74LS123
U19	74LS123
U20	74LS04
U21	74LS10
U22	74LS138
U23	74LS08
U24	8212 (2)
U25	8212 (3)
U26	1488
U27	74LS390
U28	74LS00
U29	CA3046
U30	74LS367
U31	74LS374
U32	74LS74
U33	74S188
U34	74S188
U35	74S371
U36	74S371
U37	74LS374
U38	81LS97
U39	74LS32
U40	74LS74
U41	74LS138
U42	74LS139
U43	74LS86
U44	74LS08
U45	Instant ROM
U46	Instant ROM
U47	2516
U48	2516
U49	2516
U50	2516
U51	8216
U52	8216
U53	81LS97
U54	74LS138
U55	2141
U56	2141
U57	2141
U58	2141
U59	2141
U60	2141
U61	2141
U62	2141
U63	2141
U64	2141
U65	2141
U66	2141
U67	2141
U68	2141
U69	2141
U70	2141
U71	74LS32
U72	74LS32
U73	74LS32
U74	74LS32
U75	74LS09
U76	74LS00
U77	58174
U78	8216
U79	DG508
U80	ZN427
U81	74LS74
U82	74LS244
U83	74LS138
U84	74LS273
U85	8232
U86	LM324
U87	LM324
U88	LM324
U89	LM324
U90	LM324
U91	LM324

APPENDIX 10 Governor Controller Software Listings

The controller software is written as nineteen modules which are linked together to form the complete software package. The module listings are as follows :

```
;**********************************************************************
;MAIN CALLING PROGRAM
;**********************************************************************
          NAME ('MAIN')
          CSEG
          EXT WTRAP,DFAULT,MERROR,RESYNC,SNAKE,INITL,MSGE
          EXT TIMLOG,RBKGEN,CNTRL,PRBUFF,SLFTST,NMOUT
          PUBLIC MAIN,STACK
          ORG 0000H
MAIN:     DI                  ;DISABLE INTERRUPTS
          OUT 00H             ;DISABLE RST/HOLD
          JMP MAIN1
;
          ORG 0024H
          JMP WTRAP           ;TRAP WATCHDOG VECTOR
;
          ORG 0028H
          JMP DFAULT          ;RST5 SOFTWARE ERROR VECTOR
;
          ORG 002CH
          JMP MERROR          ;RST5.5 MEMORY ERROR VECTOR
;
          ORG 0034H
          JMP RESYNC          ;RST6.5 VOTING ERROR VECTOR
;
          ORG 0038H
          JMP SNAKE           ;RST7 SNAKE VECTOR
;
MAIN1:    MVI C,0FFH          ;INITIALISE ALL RAM TO FF FOR SNAKE
          LXI H,3000H         ;START OF RAM
FILL:     MOV M,C
          INX H
          MOV A,H
          CPI 40H             ;END OF RAM?
          JNZ FILL            ;LOOP UNTIL ALL RAM FILLED
          LXI SP,STACK        ;INITIALISE STACK
          CALL INITL          ;INITIALISE REGISTERS AND VARIABLES
          LXI H,CLDMSG        ;LOG COLD RESET
          CALL MSGE
          LXI H,ROMMSG        ;LOG EPROM SET IN USE
          CALL MSGE
          LDA 0007H           ;READ NUMBER FROM EPROM
          CALL NMOUT          ;PRINT IT
          CALL TIMLOG         ;AND TIME
          OUT 02H             ;ENABLE RST/HOLD
MAIN2:    CALL RBKGEN         ;GENERATE RECOVERY BLOCK
          EI                  ;ENABLE INTERRUPTS AFTER RBKGEN!
          OUT 06H             ;RESET WATCHDOGS
          CALL CNTRL          ;CONTROL ALGORITHM
          CALL PRBUFF         ;PRINT A CHARACTER FROM BUFFER
          CALL SLFTST         ;SELF TEST - SOLENOIDS EVERY 10 MINS.
          JMP MAIN2           ;REPEAT CALLING LOOP
;
CLDMSG:   DB 0AH,'FULL SYSTEM RESET',00H
;
ROMMSG:   DB 0AH,'EPROM SET ',00H


;
STACK     EQU 3F00H
          END
```

```
;*************************************************************
;RESYNCHRONISATION ROUTINE - RST6.5 INTERRUPT
;DESTROYS ALL REGS. AND EXECUTES RECOVERY BLOCK
;*************************************************************
           NAME ('RESYNC')
           CSEG
           EXT MSGE,NMOUT,BLOCK,STACK,COUTBF
           PUBLIC RESYNC,SRETRY
RESYNC:    OUT 00H         ;DISABLE RST/HOLD - QUICKER EXECUTION
           LXI SP,STACK    ;RESET STACK POINTER
           PUSH PSW        ;RESYNCHRONISATION ROUTINE
           PUSH H
           LXI H,0000H
           DAD SP
           PUSH H          ;PUSH STACK POINTER
           PUSH B
           PUSH D
           POP D           ;REVERSE THE PROCESS
           POP B
           POP H
           SPHL            ;RESTORE STACK POINTER
           POP H
           POP PSW
           MVI A,40H       ;RESET SOD LINE
           SIM
           LXI H,SRETRY
           DCR M           ;DECREMENT RETRY COUNTER HELD IN RAM
           JZ DISABL       ;DISABLE RST 6.5 IF RETRY EXHAUSTED.
           EI              ;ENABLE INTERRUPT FOR HARDWARE RETRY
           OUT 02H         ;ENABLE RST/HOLD
           IN 02H          ;GET CURRENT SYNDROME
           ANI 03H         ;MASK OFF ERROR FLAGS
           CPI 03H         ;TEST FOR BOTH FLAGS HIGH
           JZ ERRLOG       ;NO ERROR SO LOG TRANSIENT
           OUT 07H         ;GENERATE HARDWARE RST6.5 INTERRUPT
           HLT             ;WAIT FOR INTERRUPT
 ERRLOG:   LXI H,EMSGE     ;LOG TRANSIENT ERROR
           JMP NOCHAN
;
 DISABL:   RIM             ;SWITCH OUT RST 6.5
           ANI 07H
           ORI 0AH
           SIM
           OUT 00H         ;DISABLE RST/HOLD
           LXI H,DMSGE     ;LOG CPU FAILURE
;
 NOCHAN:   CALL MSGE
           IN 01H          ;GET LATCHED ERROR SYNDROME
           ANI 03H         ;STRIP OFF STATUS BITS
           LXI H,TABLE1    ;LOOK UP TABLE FOR CHANNEL IN ERROR
           ADD L
           MOV L,A
           MOV A,M
           CALL NMOUT      ;PRINT IT
           MVI C,0AH
           CALL COUTBF
```

```
;
         LXI H,SYNMSG     ;LOG SYNDROME
         CALL MSGE
         IN 01H           ;GET LATCHED SYNDROME
         CALL NMOUT       ;PRINT IT
;
         LXI H,RETMSG     ;LOG NUMBER OF RETRIES
         CALL MSGE
         LXI H,SRETRY
         MVI A,0FFH
         SUB M
         CALL NMOUT
;
END:     MVI A,0FFH       ;RESET RETRY COUNTER
         STA SRETRY       ;STORE IN RAM
         OUT 01H          ;RESET SYNDROME LATCH FLIP/FLOP
         JMP BLOCK        ;JUMP TO RECOVERY BLOCK
;
EMSGE:   DB 0AH,'SYNC ERR CHANNEL ',00H



;
DMSGE:   DB 0AH,'CPU FAIL CHANNEL ',00H



;
SYNMSG:  DB 'SYND=',00H

;
RETMSG:  DB '  RETRIES=',00H


;
TABLE1:  DB 02H,03H,01H,00H
;
         DSEG
SRETRY:  DS 1
         END
```

```
;###########################################################
;MEMORY ERROR ROUTINE - RST 5.5 INTERRUPT
;SAVES ALL REGS. UNLESS DBE - RECOVERY BLOCK IF DBE
;###########################################################
;
        NAME ('MERROR')
        CSEG
        EXT MSGE,NMOUT,BLOCK,COUTBF,SFAIL,TIMLOG
        PUBLIC MERROR
MERROR: PUSH PSW        ;SAVE REGISTERS
        PUSH H
        PUSH B
        PUSH D
        IN 00H          ;SAVE LATCHED SYNDROME & RESET LATCH
        MOV C,A         ;SAVE SYNDROME IN C REG.
        MVI B,MRETRY    ;INITIALISE RETRY COUNTER
WREAD:  LXI H,TSTLOC    ;IS WRITE/READ TO TESTLOC OK
RWLOOP: XRA A           ;CLEAR A
        MOV M,A         ;WRITE TO TESTLOC
        MOV A,M         ;READ BACK
        ORA A           ;SET FLAGS
        JNZ AGAIN       ;TRY AGAIN IF READ ERROR
        RIM             ;IS RST5.5 INTERRUPT PENDING
        ANI 10H
        JZ RW01         ;JUMP IF NO RST5.5 WITH ZERO IN A REG
        IN 00H          ;OTHERWISE READ SYNDROME
RW01:   MOV E,A         ;SAVE SYNDROME IN E REG.
        MVI A,0FFH      ;READ/WRITE FF
        MOV M,A         ;WRITE TO TESTLOC
        MOV A,M         ;READ BACK
        INR A           ;FF BECOMES 00
        JZ UPDATE       ;O.K. SO CHECK SYNDROME
AGAIN:  DCR B           ;DECREMENT RETRY COUNTER
        JNZ RWLOOP      ;TRY AGAIN IF NOT EXHAUSTED
        LXI H,SFMSG     ;LOG TOTAL RAM FAILURE
        JMP SFAIL       ;PRINT MESSAGE AND SOFT FAILURE
UPDATE: CALL SYNRD      ;READ CURRENT SYNDROME
        MOV D,A         ;SAVE FF W/R SYNDROME
        ORA E           ;OR IN 00 W/R SYNDROME
        JZ ERRLOG       ;LOG TRANS ERROR IF BOTH SYNDROMES=00
        IN 00H          ;CLEAR SYNDROME LATCH
        DCR B           ;DECREMENT RETRY COUNTER
        JNZ WREAD       ;RETRY IF NOT EXHAUSTED
        MOV C,E         ;00  SYNDROME = DBE ?
        CALL DBE
        JZ DBETST       ;R/W O.K. - EITHER HARDWARE OR DBE IN CHECK BITS
        MOV C,D         ;FF  SYNDROME = DBE ?
        CALL DBE
        JZ DBETST       ;DBE IN CHECK BITS OR HARDWARE
        MOV A,E         ;NOT DBE SO ARE 00 ,FF  SYNDROME EQUAL OR IS ONE
        ORA A           ;/EQUAL TO ZERO
        JZ RFAIL        ;JUMP FOR STUCK AT SINGLE BIT ERROR
        MOV C,E         ;E SYNDROME NOT ZERO OR HARDWARE FAULT SO LOAD
        MOV A,D         ;/C REG WITH SYNDROME TO BE LOGGED
        ORA A
        JZ RFAIL        ;JUMP FOR STUCK AT SBE
```

```
           CMP E             ;TEST FOR SYNDROME EQUALITY
           JNZ HWFLTY        ;MUST BE HARDWARE FAULT
RFAIL:     LXI H,FMSGE       ;LOG STUCK AT SBE
           JMP DISABL
DBETST:    LXI H,TSTDTA      ;TABLE OF TEST DATA TO READ/WRITE
NXTDTA:    MOV A,M           ;GET BYTE OF TEST DATA
           ORA A             ;SET FLAGS
           JZ HWFLTY         ;END OF TABLE - MUST BE HARDWARE FAULT
           STA TSTLOC        ;WRITE TEST DATA TO TESTLOC
           IN 00H            ;CLEAR SYNDROME LATCH
           LDA TSTLOC        ;READ BACK TEST DATA
           CALL SYNRD        ;READ SYNDROME
           ORA A             ;SET FLAGS
           INX H             ;INCREMENT TEST DATA TABLE POINTER
           JNZ NXTDTA        ;JUMP IF SYNDROME IN A REG NOT ZERO
           LXI H,DBFMSG      ;OTHERWISE LOAD DBE MESSAGE
           JMP DISABL        ;LOG MESSAGE AND DISABLE RST5.5
ERRLOG:    RIM
           ANI 10H           ;RST 5.5 PENDING ?
           JZ TRANS          ;INTERRUPT NOT STUCK SO MUST BE TRANSIENT,
           DCR B
           JNZ WREAD         ;PERFORM RETRY
HWFLTY:    LXI H,HWMSGE      ;MEMORY HARDWARE FAILED
DISABL:    RIM               ;SWITCH OUT RST 5.5
           ANI 07H
           ORI 09H
           SIM
           JMP SNDMSG        ;LOG FAILURE
TRANS:     LXI H,EMSGE       ;LOG TRANSIENT ERROR
SNDMSG:    CALL MSGE
           MOV A,C           ;GET SYNDROME
           CALL NMOUT        ;PRINT SYNDROME
           CALL TIMLOG       ;LOG TIME
           CALL DBE          ;IF ERROR=DBE THEN RETURN VIA RECOVERY BLOCK
           JZ MBLOCK
           POP D             ;RESTORE REGS. AND RETURN
           POP B
           POP H
           POP PSW
           EI
           RET
MBLOCK:    EI
           JMP BLOCK         ;JUMP TO RECOVERY BLOCK SINCE DBE
;
SYNRD:     RIM
           ANI 10H           ;TEST FOR PENDING RST5.5
           RZ                ;OTHERWISE RETURN A=00
           IN 00H            ;READ SYNDROME LATCH
           RET
;
DBE:       MOV A,C           ;ROUTINE RETURNS WITH ZERO FLAG SET IF DBE
           ANI 0FH
           CPI 02H
           RZ
           MOV A,C
           ANI 0F0H
           CPI 20H
           RET
```

```
;
SFMSG:   DB 0AH,'TOTAL RAM FAILURE',0AH,00H



;
FMSGE:   DB 0AH,'RAM FAILURE S=',00H


;
EMSGE:   DB 0AH,'TRANS RAM ERR S=',00H



;
HWMSGE:  DB 0AH,'RAM HWARE FAIL S=',00H



;
DBFMSG:  DB 0AH,'CBIT STUCK DBE S=',00H



;
TSTDTA:  DB 33H,55H,66H,99H,0AAH,0CCH,00H

;
MRETRY   EQU 64H
;
         DSEG
TSTLOC:  DS 1             ;RESERVE 1 BYTE FOR TESTLOC
;
         END
```

```
;*****************************************************************
;MAIN CONTROL ALGORITHM - DAG PRESSURE CONTROL
;DESTROYS BC,HL,A,D
;*****************************************************************
          NAME ('CNTRL')
          CSEG
          EXT PGOV,POUT,PDELTA,PRESSR
          PUBLIC CNTRL
CNTRL:    CALL PRESSR          ;READ PRESSURE TRANSDUCERS
          XRA A                ;CLEAR CARRY
          LDA PDELTA           ;GET PRESSURE DIFFERENTIAL ACROSS PLATE
          RAL                  ;MULTIPLY BY 2
          ADI PSET             ;CALCULATE PSET+2PDELTA
          MOV C,A              ;SAVE IN C REG.
          LDA PGOV             ;GET GOVERNOR PRESSURE READING
          CMP C                ;COMPARE WITH PSET+2PDELTA
          JZ HOLD              ;HOLD PRESSURE IF EQUAL
          JC UP                ;PGOV LOW SO INCREASE PRESSURE
DOWN:     SUB C                ;DECREASE PRESSURE AND CALC. PRESSURE ERROR
          MVI D,0FFH           ;SOLENOID DOWN CONTROL WORD
          JMP BWIDTH           ;NO ACTION IF PRESSURE ERROR WITHIN LIMITS
UP:       MOV B,A              ;INCREASE PRESSURE - REVERSE SUBTRACTION
          MOV A,C
          SUB B                ;CALCULATE PRESSURE ERROR
          MVI D,00H            ;SOLENOID UP CONTROL WORD
BWIDTH:   CPI MAX              ;IS PRESSURE ERROR LESS THAN MAX.
          JC HOLD              ;HOLD PRESSURE IF SO
          MOV A,D              ;OTHERWISE ADJUST PRESSURE UP OR DOWN
          JMP SOLOUT           ;OUTPUT TO SOLENOID DRIVERS
HOLD:     MVI A,0FH            ;SOLENOID HOLD CONTROL WORD
SOLOUT:   OUT 04H              ;OUTPUT TO SOLENOID PORT
          LXI H,1000H          ;DELAY
DELAY:    DCR L
          JNZ DELAY
          DCR H
          JNZ DELAY
          RET

;
MAX       EQU 05H              ;0.5 INCHES W.G.
PSET      EQU 90               ;9 INCHES W.G.
          END
```

```
;***************************************************************
;TAKE TIME AVERAGE OF EACH PRESSURE TRANSDUCER, THEN PERFORM
;MAJORITY VOTE AND RETURN VALUES - P6OV, POUT, PDELTA
;DESTROYS ALL REGISTERS
;***************************************************************
          NAME ('PRESSR')
          CSEG
          EXT MSGE,NMOUT,TIMLOG
          PUBLIC PRESSR,PRETRY,P6OV,POUT,PDELTA
PRESSR:   LXI H,PTABLE        ;INITIALISE PRESSURE READINGS TABLE PNTR.
          MVI B,05H           ;CHANNEL COUNTER
SCAN:     CALL PAVE           ;TIME AVERAGE FOR 8 READINGS OF CHANNEL
          MOV M,A             ;SAVE IN TABLE
          INX H               ;INCREMENT TABLE POINTER
          DCR B               ;DECREMENT CHANNEL COUNT
          JP SCAN             ;LOOP UNTIL FINISHED
VOTE:     LXI H,PTABLE+2      ;START AT END OF TABLE - WORK BACKWARDS
          LXI D,PTABLE+6      ;STORE AVERAGES IN TABLE
          CALL AVRGE2         ;CALCULATE 3 DIFFERENT AVES. - CHANNELS 3-5
          LXI H,PTABLE+5      ;REPEAT FOR CHANNELS 0-2
          LXI D,PTABLE+9
          CALL AVRGE2
          LXI H,PTABLE+3      ;TABLE FOR CHANNELS 0-2
          LXI D,PTABLE+9      ;CORRESPONDING AVERAGES
          MVI C,00H           ;CHANNELS OFFSET
          CALL CHECK          ;COMPARE CHANNELS AND AVERAGES AND LOG
          STA P6OV            ;/ERRORS - STORE BEST PRESSURE IN P6OV
          LXI H,PTABLE        ;REPEAT SAME FOR CHANNELS 3-5
          LXI D,PTABLE+6
          MVI C,03H
          CALL CHECK
          STA POUT            ;STORE BEST PRESSURE IN POUT
          MOV C,A
          LDA P6OV            ;CALCULATE PRESSURE DIFFERENTIAL ACROSS
          SUB C               ;/ORIFICE PLATE
          STA PDELTA
          RET
;
PAVE:     PUSH H              ;SAVE HL
          LXI H,0000H         ;ZERO HL
          LXI D,0000H         ;ZERO HL
          MVI C,08H           ;READ CHANNEL 8 TIMES
PAVE1:    CALL ADREAD         ;READ CHANNEL
          MOV E,A
          DAD D               ;ADD VALUE TO PREVIOUS SUM
          DCR C               ;DECREMENT LOOP COUNTER
          JNZ PAVE1           ;LOOP UNTIL DONE 8 TIMES
          MVI C,03H           ;COUNT 3 SHIFTS RIGHT HL - DIVIDE BY 8
PAVE2:    MOV A,H
          RAR
          MOV H,A
          MOV A,L
          RAR
          MOV L,A
          DCR C
          JNZ PAVE2           ;REPEAT UNTIL 3 SHIFT OPERATIONS
```

```
        POP H               ;RESTORE HL
        RET
;
ADREAD: MOV A,B             ;GET CHANNEL NO. 0-5
        OUT 05H             ;OUTPUT TO MUX
        OUT 03H             ;START CONVERSION A/D CONVERTOR
NEOC:   IN 04H              ;TEST EOC FLAG
        RLC
        JNC NEOC            ;LOOP TILL CONVERSION COMPLETE
        IN 03H              ;READ A/D CONVERTOR
        SUI OFFSET          ;SUBTRACT CALIBRATION OFFSET
        RET
;
AVRGE2: XRA A               ;CLEAR CARRY
        MOV A,M             ;GET CHANNEL C
        MOV C,A             ;SAVE IN C REG.
        DCX H               ;POINT TO CHANNEL B
        ADD M               ;ADD CHANNELS B+C
        RAR                 ;DIVIDE BY 2
        STAX D              ;STORE (B+C)/2
        XRA A               ;CLEAR CARRY
        MOV A,C             ;GET CHANNEL C
        DCX H               ;POINT TO CHANNEL A
        INX D               ;INCREMENT TABLE POINTER
        ADD M               ;ADD CHANNELS A+C
        RAR                 ;DIVIDE BY 2
        STAX D              ;STORE (A+C)/2
        XRA A               ;CLEAR CARRY
        MOV A,M             ;GET CHANNEL A
        INX H               ;POINT TO CHANNEL B
        INX D               ;INCREMENT TABLE POINTER
        ADD M               ;ADD CHANNELS A+B
        RAR                 ;DIVIDE BY 2
        STAX D              ;STORE (A+B)/2
        RET
;
AVRGE3: PUSH B              ;SAVE REGISTERS
        PUSH H
        PUSH D
        LXI D,0000H         ;INITIALISE SUM
        XRA A               ;CLEAR CARRY
        MOV A,M             ;GET 1ST VALUE
        INX H
        ADD M               ;ADD 2ND VALUE
        MOV E,A             ;SAVE IN E REG.
        JNC SKIP            ;INCREMENT D IF CARRY
        INR D
SKIP:   XRA A               ;CLEAR CARRY
        INX H
        MOV A,M
        ADD E               ;ADD IN 3RD VALUE
        MOV E,A             ;SAVE IN E REG.
        JNC DIV3            ;INCREMENT D IF CARRY ELSE SKIP
        INR D
DIV3:   LXI H,0000H         ;INITIALISE RESULT
        MVI C,04H           ;SHIFT AND ADD 4 TIMES EQUALS DIVIDE BY 3
```

```
DIV3A:   CALL SHIFT         ;SHIFT DE RIGHT ONE PLACE
         CALL SHIFT         ;AND AGAIN
         DAD D              ;ADD TO PARTIAL RESULT
         DCR C              ;DECREMENT SHIFT COUNTER
         JNZ DIV3A
         MOV A,L            ;PUT RESULT IN A REG.
         POP D
         POP H
         POP B
         RET
;
SHIFT:   MOV A,D            ;SHIFT D RIGHT THRO CARRY
         RAR
         MOV D,A
         MOV A,E            ;SHIFT E RIGHT LINKING DE SHIFT VIA CARRY
         RAR
         MOV E,A
         RET
;
CHECK:   MVI B,03H          ;CHANNEL COUNTER - 3 CHANNELS
         CALL AVRGE3        ;CALCULATE AVERAGE OF 3 CHANNELS
         STA PTEMP          ;SAVE RESULT IN CASE IT IS USED
CHECK3:  CALL MOD           ;IS MOD(A-(B+C)/2) ETC. GTR. THAN ALLOWED
         JC CHECK1          ;JUMP IF NOT
         LDAX D             ;IF SO OVERWRITE PRESSURE
         STA PTEMP          ;/WITH AVERAGE OF OTHER 2
         PUSH H             ;SAVE REGS
         PUSH B
         LXI H,PRETRY
         DCR M              ;DECREMENT RETRY COUNTER
         JZ CHECK2          ;DO NOT LOG ERROR IF RETRY EXHAUSTED
         LXI H,CHMSG        ;LOG ERROR
         CALL MSGE
         MOV A,B            ;GET CHANNEL COUNT
         ADD C              ;ADD OFFSET
         CALL NMOUT         ;LOG CHANNEL IN ERROR
         CALL TIMLOG        ;AND TIME
CHECK2:  MVI A,01H
         STA PRETRY         ;SET RETRY COUNTER TO 1
         POP B              ;RESTORE REGS.
         POP H
CHECK1:  INX H              ;INCREMENT TABLE COUNTERS
         INX D              ;/TO NEXT CHANNEL
         DCR B              ;DECREMENT LOOP COUNTER
         JNZ CHECK3         ;LOOP UNTIL ALL 3 CHANNELS DONE
         LDA PTEMP          ;RETURN WITH RESULT IN A REG.
         RET
;
MOD:     PUSH B             ;SAVE BC
         LDAX D             ;(AVERAGE OF OTHER 2) MINUS (OTHER VALUE)
         SUB M              ;E.G. (B+C)/2 - A
         JNC MOD1           ;JUMP IF RESULT POSITIVE
         LDAX D             ;OTHERWISE REVERSE SUBTRACTION ORDER
         MOV C,A
         MOV A,M
         SUB C
```

```
MOD1:    CPI DELTA           ;COMPARE WITH PERMISSABLE DIFFERENCE
         POP B
         RET
;
CHMSG:   DB 0AH,'PRESSR ERROR CH ',00H



;
OFFSET   EQU 10              ;1 INCH W.G.
DELTA    EQU 10              ;1 INCH W.G.
;
         DSEG
PTABLE:  DS 12
PTEMP:   DS 1
P60V:    DS 1
POUT:    DS 1
PDELTA:  DS 1
PRETRY:  DS 1
         END
```

```
;***************************************************************
;SAVES CURRENT PROGRAM STATUS OF ALL REGISTERS
;***************************************************************
;
         NAME ('RBKGEN')
         CSEG
         PUBLIC RBKGEN,RBLOCK
 RBKGEN: DI                   ;MUST NOT INTERRUPT RECOVERY BLOCK
         OUT 00H              ;DISABLE RST/HOLD - QUICKER EXECUTION
         SHLD RBLOCK+6        ;SAVE HL
         XTHL
         SHLD RBLOCK+8        ;SAVE PC RETURN ADDRESS
         XTHL                 ;RESTORE STACK TOP
         LXI H,0000H
         DAD SP
         SHLD RBLOCK+10       ;SAVE SP
         LXI SP,RBLOCK+6      ;CHANGE SP TO RECOVERY BLOCK ADDRESS
         PUSH D               ;SAVE DE
         PUSH B               ;SAVE BC
         PUSH PSW             ;SAVE PSW
         RIM                  ;IF RST6.5 DISABLED LEAVE RST/HOLD DISABLED
         ANI 02H
         JNZ RETURN
         OUT 02H              ;ENABLE RST/HOLD
 RETURN: POP PSW              ;RESTORE PSW
         LHLD RBLOCK+10
         SPHL                 ;RESTORE SP
         LHLD RBLOCK+6        ;RESTORE HL
         EI
         RET
;
         DSEG
RBLOCK:  DS 12                ;RESERVE 12 BYTES
         END
```

```
;*****************************************************************
;RESTORES ALL REGISTERS AND PROGRAM STATUS - COMPLEMENTARY TO RBKGEN
;*****************************************************************
;
        NAME ('BLOCK')
        CSEG
        EXT RBLOCK,TIMLOG,MSGE
        PUBLIC BLOCK
BLOCK:  LXI H,VMSG              ;LOG VECTORED RECOVERY
        CALL MSGE
        CALL TIMLOG
        LXI SP,RBLOCK           ;LOAD SP WITH RECOVERY BLOCK START ADDRESS
        POP PSW                 ;RESTORE PSW
        POP B                   ;RESTORE BC
        POP D                   ;RESTORE DE
        LHLD RBLOCK+10
        SPHL                    ;RESTORE SP
        LHLD RBLOCK+8           ;SAVE RETURN ADDRESS ON STACK TOP
        XTHL
        LHLD RBLOCK+6           ;RESTORE HL
        RET                     ;JUMP TO RETURN ADDRESS
;
VMSG:   DB OAH,'VECTORED RECOVERY',OOH


        END
```

```
;#####################################################################
;SELF TEST OF SOLENOID DRIVERS - CAN BE EXPANDED TO TEST ENTIRE SYSTEM
;#####################################################################
          NAME ('SLFTST')
          CSEG
          EXT MSGE,TIMLOG
          PUBLIC SLFTST,TFLAG
SLFTST:   IN 14H                ;READ MINUTES FROM CLOCK
          ANI 0FH               ;SELECT LOWER BITS
          CPI 0FH               ;ILLEGAL READ?
          JZ SLFTST             ;IF SO TRY AGAIN
          CPI 00H               ;MINS = 0 ?
          JZ SLF1
          XRA A                 ;CLEAR SELF TEST FLAG AND RETURN
          STA TFLAG
          RET
SLF1:     LDA TFLAG             ;TEST FLAG FOR FFH
          INR A                 ;FF BECOMES 00
          RZ                    ;RETURN IF FLAG SET
          MVI A,0FFH            ;OTHERWISE SET FLAG AND DO SELF TEST
          STA TFLAG
          XRA A
          OUT 04H               ;TURN BOTH SOLENOIDS OFF
          CALL DELAY            ;ALLOW VOLTAGE TO SETTLE
          MVI A,06H             ;READ SOLENOID VOLTAGE - CHANNEL 6
          CALL ATOD
          CPI HIGH              ;COMPARE WITH SATISFACTORY VALUE
          JC ERROR              ;ERROR IF TOO LOW
          MVI A,07H             ;CHECK OTHER SOLENOID
          CALL ATOD
          CPI HIGH
          JC ERROR              ;ERROR IF TOO LOW
          MVI A,0FFH
          OUT 04H               ;TURN BOTH SOLENOIDS ON
          CALL DELAY            ;ALLOW VOLTAGE TO SETTLE
          MVI A,06H             ;READ SOLENOID VOLTAGE - CHANNEL 6
          CALL ATOD
          CPI LOW               ;COMPARE WITH SATISFACTORY VALUE
          JNC ERROR             ;ERROR IF TOO HIGH
          MVI A,07H             ;CHECK OTHER SOLENOID
          CALL ATOD
          CPI LOW
          JNC ERROR             ;ERROR IF TOO HIGH
          MVI A,0FH
          OUT 04H               ;HOLD PRESSURE VIA SOLENOIDS
          RET
;
ERROR:    LXI H,SOLMSG          ;LOG FAILURE
          CALL MSGE
          CALL TIMLOG           ;AND TIME
          RET
;
ATOD:     OUT 05H               ;OUTPUT CHANNEL TO MUX
          OUT 03H               ;START CONVERSION
NEOC:     IN 04H                ;READ END OF CONVERSION FLAG
          RLC
```

```
         JNC NEOC          ;LOOP UNTIL EOC
         IN 03H            ;READ A TO D CONVERTER
         RET
;
DELAY:   LXI H,0700H       ;APPROX 10MS DELAY
DELAY1:  DCR L
         JNZ DELAY1
         DCR H
         JNZ DELAY1
         RET
;
SOLMSG:  DB 0AH,'SOLENOID FAILURE',00H



;
HIGH     EQU 8CH
LOW      EQU 61H
;
         DSEG
TFLAG:   DS 1
         END
```

```
;****************************************************************
;USES NO RAM AND THEREFORE NO SUBROUTINES
;PRINTS MESSAGE THEN EXECUTES A SOFTWARE FAILURE - FAIL SAFE
;****************************************************************
         NAME ('SFAIL')
         CSEG
         PUBLIC SFAIL
SFAIL:   MOV C,M              ;PRINT APPROPRIATE SOFT FAIL MESSAGE
         XRA A                ;CLEAR A
         ORA C                ;TEST FOR NULL - END OF MESSAGE
         JZ SAFE              ;MAKE PLANT SAFE/SHUT DOWN
SFCOUT:  RIM                  ;TEST FOR PRINTER BUSY
         ANI 80H
         JNZ SFCOUT           ;LOOP IF PRINTER BUSY
         MVI B,0BH            ;PRINT CHARACTER IN C REG. - THIS ROUTINE
         MVI A,0C0H           ;/IS A DIRECT COPY OF COUT
SFC01:   SIM
         LXI D,0172H          ;1200 BAUD DELAY
SFC02:   DCR E
         JNZ SFC02
         DCR D
         JNZ SFC02
         STC
         MOV A,C
         RAR
         MOV C,A
         MVI A,80H
         RAR
         XRI 80H
         DCR B
         JNZ SFC01
         INX H                ;GET NEXT CHARACTER IN MESSAGE
         JMP SFAIL
SAFE:    XRA A
         OUT 04H              ;TURN BOTH SOLENOIDS OFF - FAILS SAFE
;
;        ADD EXTRA SHUT DOWN ROUTINES HERE
;
         HLT                  ;HALT CONTROLLER
         END
```

```
;**********************************************************************
;SYSTEM CRASH - WATCHDOG TIMER CAUSES VECTORED RECOVERY AND LOGS
;TIME AND ADDRESS
;DESTROYS ALL REGS.
;**********************************************************************
          NAME ('WTRAP')
          CSEG
          EXT MSGE,NMOUT,BLOCK,BUFFST,BUFFRD,BUFFWR,COUT
          PUBLIC WTRAP,WRETRY
WTRAP:    POP D                ;GET CRASH ADDRESS OFF STACK
          LXI H,WRETRY
          DCR M                ;DECREMENT RETRY COUNTER
          JNZ WTRAP1           ;EXECUTE SOFT RECOVERY
          HLT                  ;WAIT FOR SYSTEM RESET IF COUNT EXHAUSTED
WTRAP1:   LXI H,BUFFST         ;CLEAR OUTPUT BUFFER IN CASE CORRUPTED
          SHLD BUFFRD
          SHLD BUFFWR
          MVI C,00H            ;SEND NULL TO CLEAR PRINTER
          CALL COUT
          LXI H,WDMSG          ;LOG CRASH
          CALL MSGE
          MOV A,D              ;LOG HIGH ORDER ADDRESS
          CALL NMOUT
          MOV A,E              ;LOG LOW ORDER ADDRESS
          CALL NMOUT
          EI                   ;ENABLE INTERRUPTS AFTER COUT
          JMP BLOCK            ;VECTORED RECOVERY
;
WDMSG:    DB 0AH,'TRAP WDOG ADR=',00H


;
          DSEG
WRETRY:   DS 1                 ;RETRY COUNTER
          END
```

```
;#######################################################################
;SNAKE - INVALID ADDRESS RANGE CAUSES EXECUTION OF RST7 - LOG ADDRESS,
;TIME AND EXECUTE VECTORED RECOVERY
;DESTROYS ALL REGISTERS
;#######################################################################
          NAME ('SNAKE')
          CSEG
          EXT NMOUT,BLOCK,MSGE,BUFFST,BUFFRD,BUFFWR
          PUBLIC SNAKE
SNAKE:    POP D                 ;GET ERROR ADDRESS FROM STACK
          LXI H,BUFFST          ;CLEAR OUTPUT BUFFER IN CASE CORRUPTED
          SHLD BUFFRD
          SHLD BUFFWR
          LXI H,SNMSG           ;LOG ERROR MESSAGE
          CALL MSGE
          MOV A,D
          CALL NMOUT            ;HIGH ORDER ADDRESS
          MOV A,E
          CALL NMOUT            ;LOW ORDER ADDRESS
          JMP BLOCK             ;VECTORED RECOVERY AND LOG TIME
;
SNMSG:    DB 0AH,'SNAKE RST7 ADR=',00H



          END
```

```
;***********************************************************************
;SOFTWARE ERROR - LOG ADDRESS AND TIME AND EXECUTE VECTORED RECOVERY
;DESTROYS ALL REGS
;***********************************************************************
          NAME ('DFAULT')
          CSEG
          EXT NMOUT,BLOCK,MSGE
          PUBLIC DFAULT
DFAULT:   POP D                   ;GET ERROR ADDRESS OFF STACK
          LXI H,SOFMSG            ;LOG ERROR
          CALL MSGE
          MOV A,D                 ;LOG HIGH ORDER ADDRESS
          CALL NMOUT
          MOV A,E
          CALL NMOUT
          JMP BLOCK               ;VECTORED RECOVERY AND LOG TIME

;
SOFMSG:   DB 0AH,'SOFT ERROR ADR=',00H



          END
```

```
;**********************************************************************
;INITIALISE REGISTERS AND VARIABLES
;**********************************************************************
        NAME ('INITL')
        CSEG
        EXT WRETRY,SRETRY,PRETRY,TFLAG,COUT
        PUBLIC INITL,BUFFST,BUFFRD,BUFFWR,BUFEND
INITL:  LXI H,0000H         ;SYNCHRONISE ALL REGISTERS
        LXI B,0000H
        LXI D,0000H
        PUSH PSW
        POP PSW
        MVI A,08H           ;ENABLE ALL INTERRUPTS
        SIM
        IN 00H              ;RESET MEMORY FLAGS LATCH
        OUT 01H             ;CLEAR CPU ERROR LATCH
        LXI H,BUFFST        ;INITIALISE OUTPUT BUFFER POINTERS
        SHLD BUFFRD
        SHLD BUFFWR
        MVI A,0FFH          ;INITIALISE RESYNC RETRY COUNTER
        STA SRETRY
        MVI A,0AH           ;INITIALISE PRESSURE RETRY COUNTER
        STA PRETRY
        MVI A,05H           ;INITIALISE TRAP WATCHDOG RETRY COUNTER
        STA WRETRY
        XRA A               ;CLEAR SELF TEST FLAG
        STA TFLAG
        MVI C,00H           ;SEND NULL TO CLEAR PRINTER
        CALL COUT
        RET
;
BUFEND: DW 4000H
BUFFST  EQU 3F00H
;
        DSEG
BUFFRD: DS 2
BUFFWR: DS 2
        END
```

```
;*************************************************************
;OUTPUTS CHARACTER IN C REG. TO SERIAL DEVICE
;DESTROYS  BC,A
;*************************************************************
;
                  NAME ('COUT')
                  CSEG
                  PUBLIC COUT
COUT:             PUSH D          ;SAVE DE
                  DI              ;DISABLE INTERRUPTS AND RST/HOLD
                  OUT 00H         ;/COUT - SPEED SENSITIVE
                  MVI B,0BH       ;NO. BITS TO SEND
                  MVI A,0C0H      ;SET START BIT AND SOD ENABLE
C01:              SIM             ;OUTPUT CARRY TO SOD PIN
                  LXI D,0172H     ;1200 BAUD DELAY
C02:              DCR E
                  JNZ C02
                  DCR D
                  JNZ C02
                  STC             ;SET EVENTUAL STOP BITS
                  MOV A,C         ;ROTATE CHAR. RIGHT PUTTING NEXT
                  RAR             ;DATA BIT INTO CARRY
                  MOV C,A
                  MVI A,80H       ;SET EVENTUAL SOD ENABLE BIT
                  RAR             ;SHIFT DATA BIT IN CARRY TO SOD BIT
                  XRI 80H         ;INVERT SOD DATA BIT
                  DCR B           ;CHARACTER SENT ?
                  JNZ C01         ;NO,THEN SEND NEXT BIT
                  RIM             ;IF RST 6.5 DISABLED THEN LEAVE
                  ANI 02H         ;/RST/HOLD DISABLED
                  JNZ RETURN
                  OUT 02H
RETURN:           POP D           ;RESTORE DE - CALLING ROUTINE
                  RET             ;/MUST ENABLE INTERRUPTS IF REQUIRED
                  END
```

```
;*********************************************************************
;PUTS CHARACTER IN C REG. INTO OUTPUT BUFFER
;DESTROYS A
;*********************************************************************
          NAME ('COUTBF')
          CSEG
          PUBLIC COUTBF
          EXT BUFFST,BUFEND,BUFFRD,BUFFWR
COUTBF:   PUSH H              ;SAVE HL REG.
          LHLD BUFFWR         ;GET BUFFER WRITE POINTER
          INX H               ;INCREMENT POINTER
          LDA BUFEND          ;TEST FOR END OF BUFFER
          SUB L
          JNZ FULTST
          LDA BUFEND+1
          SBB H
          JNZ FULTST
          LXI H,BUFFST        ;RESET POINTER TO START OF BUFFER
FULTST:   LDA BUFFRD          ;TEST FOR BUFFER FULL
          SUB L
          JNZ WRITE
          LDA BUFFRD+1
          SBB H
          JZ RETURN
WRITE:    SHLD BUFFWR         ;SAVE BUFFER POINTER
          MOV M,C             ;SAVE CHARACTER IN BUFFER
RETURN:   POP H               ;RESTORE HL REG.
          RET
          END
```

```
;*********************************************************************
;PRINT A CHARACTER FROM BUFFER UNLESS EMPTY
;*********************************************************************
          NAME ('PRBUFF')
          CSEG
          EXT BUFFRD,BUFFWR,BUFEND,BUFFST,COUT
          PUBLIC PRBUFF
PRBUFF:   RIM                 ;READ SOD LINE
          ANI 80H             ;TEST MSB - PRINTER BUSY LINE
          RNZ                 ;RETURN IF BUSY
          LHLD BUFFRD         ;TEST IF BUFFER EMPTY
          LDA BUFFWR
          SUB L
          JNZ READ            ;JUMP IF NOT EMPTY
          LDA BUFFWR+1
          SBB H
          RZ                  ;RETURN IF BUFFER EMPTY
READ:     INX H               ;INCREMENT BUFFER POINTER
          LDA BUFEND          ;TEST FOR END OF BUFFER
          SUB L
          JNZ READ1           ;JUMP IF NOT
          LDA BUFEND+1
          SBB H
          JNZ READ1           ;JUMP IF NOT
          LXI H,BUFFST        ;OTHERWISE SKIP TO BEGINNING OF BUFFER
READ1:    SHLD BUFFRD         ;STORE NEW BUFFER POINTER
          MOV C,M             ;READ A CHARACTER FROM BUFFER
          CALL COUT           ;PRINT IT
          EI                  ;ENABLE INTERRUPTS DISABLED BY COUT
          RET
          END
```

```
;*****************************************************************
;PUTS A MESSAGE POINTED TO BY HL REG. INTO OUTPUT BUFFER
;DESTROYS HL,A
;*****************************************************************
;
                NAME ('MSGE')
                CSEG
                PUBLIC MSGE
                EXT COUTBF
MSGE:           PUSH B            ;SAVE BC
CHAR:           MOV C,M           ;GET A CHARACTER
                XRA A
                ORA C
                JZ RETURN         ;RETURN IF NULL
                CALL COUTBF       ;SEND CHARACTER
                INX H
                JMP CHAR          ;GET NEXT CHARACTER
RETURN:         POP B             ;RESTORE BC
                RET
                END
```

```
;***********************************************************
;SENDS NUMBER IN A REG. TO OUTPUT BUFFER AS 2 HEX DIGITS
;DESTROYS  A
;***********************************************************
;
                NAME ('NMOUT')
                CSEG
                PUBLIC NMOUT
                EXT COUTBF
 NMOUT:         PUSH B          ;SAVE BC
                PUSH PSW        ;SAVE ARGUMENT
                RRC             ;GET UPPER 4 BITS TO LOWER
                RRC
                RRC
                RRC
                ANI 0FH         ;SELECT LOB
                CALL CNVRT      ;CONVERT TO ASCII
                MOV C,A
                CALL COUTBF     ;SEND IT
                POP PSW         ;RESTORE ARGUMENT
                ANI 0FH         ;GET LOWER BITS
                CALL CNVRT      ;CONVERT TO ASCII
                MOV C,A
                CALL COUTBF     ;SEND IT
                POP B           ;RESTORE BC
                RET
;
CNVRT:          ORI 30H         ;ADD OFFSET FOR 0-9
                CPI 3AH         ;TEST IF A-F
                RC              ;RETURN IF NOT
                ADI 07H         ;OTHERWISE ADD FURTHER OFFSET OF 7
                RET
;
                END
```

```
;**********************************************************************
;LOGS DATE AND TIME
;DESTROYS A,HL,C
;**********************************************************************
;
          NAME ('TIMLOG')
          CSEG
          EXT MSGE,COUTBF
          PUBLIC TIMLOG
TIMLOG:   LXI H,TABLE                 ;STORE DATE AND TIME IN TABLE
          IN 1CH                      ;10'S MONTHS
          CALL TSAVE
          IN 1BH                      ;MONTHS
          CALL TSAVE
          IN 19H                      ;10'S DAYS
          CALL TSAVE
          IN 18H                      ;DAYS
          CALL TSAVE
          IN 17H                      ;10'S HOURS
          CALL TSAVE
          IN 16H                      ;HOURS
          CALL TSAVE
          IN 15H                      ;10'S MINS.
          CALL TSAVE
          IN 14H                      ;MINS.
          CALL TSAVE
          LXI H,MSG1
          CALL MSGE                   ;DATE
          LXI H,TABLE+2               ;DAY
          CALL DBLOUT
          MVI C,':'
          CALL COUTBF
          LXI H,TABLE                 ;MONTH
          CALL DBLOUT
          LXI H,MSG2                  ;TIME
          CALL MSGE
          LXI H,TABLE+4
          CALL DBLOUT
          MVI C,':'
          CALL COUTBF
          CALL DBLOUT
          MVI C,0AH
          CALL COUTBF
          RET
TSAVE:    ANI 0FH                     ;MASK OFF 4 LSB
          CPI 0FH                     ;HAS REGISTER BEEN UPDATED?
          JZ AGAIN                    ;RETRY
          ORI 30H                     ;CONVERT TO ASCII
          MOV M,A                     ;STORE IN TABLE
          INX H                       ;INCREMENT TABLE POINTER
          RET
AGAIN:    POP H                       ;DUMMY TO RESTORE STACK
          JMP TIMLOG                  ;READ ALL REGISTERS
DBLOUT:   MOV C,M                     ;SEND 2 CHARACTERS TO BUFFER
          CALL COUTBF
          INX H
          MOV C,M
          CALL COUTBF
          INX H
          RET

MSG1:     DB 0AH,'DATE ',00H
MSG2:     DB ' HR ',00H
          DSEG
TABLE:    DS 8
          END
```