FAULT-TOLERANT DISTRIBUTED MEASUREMENT SYSTEMS

by

Christian Gater

A thesis presented for the degree of
Doctor of Philosophy

Faculty of Science

University of Edinburgh

July 1987
CONTENTS

TITLE .............................................. i
DECLARATION ...................................... ii
CONTENTS ........................................ iii
ACKNOWLEDGMENTS ................................. vii
ABSTRACT .......................................... viii
LIST OF ABBREVIATIONS ......................... ix

1 INTRODUCTION ................................... 1
   1.1 An Overview of a Process Control System .... 1
   1.2 Standardization Efforts and the Field Bus Standard .... 1
   1.3 Functional Requirements of the Field Bus .... 4
   1.4 The Distributed Measurement System ............ 4

2 DESIGN FOR SYSTEM RELIABILITY ............... 8
   2.1 FIBRE-OPTIC MEASUREMENT SYSTEMS ............ 9
      2.1.1 Sensor Types ................................ 9
      2.1.2 Intensity Modulated Sensors ................. 10
      2.1.3 Distributed Optical Sensors ................. 15
   2.2 FAULT-TOLERANT COMMUNICATIONS NETWORKS .... 20
      2.2.1 Faults Within a Network ....................... 20
      2.2.2 Error Handling Protocols ..................... 23
      2.2.3 Fault-Tolerant Bus Topologies ................. 24
      2.2.4 Fault-Tolerant Loop Topologies ............... 25
   2.3 ERROR DETECTING AND ERROR CORRECTING CODES .... 31
      2.3.1 Codes and Channels ............................ 31
      2.3.2 Block Codes ................................ 32
      2.3.3 Hamming Codes ................................ 32
2.3.4 Cyclic Redundancy Check (CRC) Codes 35
2.3.5 Convolutional Codes 37
2.3.6 Choice of Error Correcting Code 39
2.4 BUILT-IN-TEST METHODS 40
2.4.1 The Stuck-At-Fault Model 40
2.4.2 Design for Testability 41
2.4.3 Ad Hoc Testability Methods 42
2.4.4 Random Pattern Testing Methods 43
2.4.5 Structured Testability Methods 43
2.4.6 Self-Test Methods 46
2.4.7 Alternative Test Methods 47
2.5 REDUNDANCY IN FAULT-TOLERANT SYSTEMS 49
2.5.1 Dual Redundant Systems 49
2.5.2 Triple Modular Redundancy (TMR) 50
2.5.3 Hybrid Redundancy 52
2.5.4 Synchronizing Redundant Systems 56
3 FAULT-TOLERANT LOOP NETWORKS 58
3.1 Redundant Node Design 59
3.2 Comparative Indices for Network Desirability 59
3.3 Networks That Can Tolerate Single Faults 63
3.4 Networks That Can Tolerate Double Faults 69
3.5 Ultra-High Reliability and Application Specific Networks 73
3.6 Description of the Experimental Network Topology 73
4 TWO INPUT/TWO OUTPUT NODE CIRCUIT 79
4.1 Battery Technology 79
4.2 Data Coding Scheme 84
4.3 Optical Data Link Design 85
4.4 Sensor/Actuator Interface 89
4.5 Node Microcomputer 90
4.6 Communications Interface Circuit . . . . . . 92
4.6.1 Communications Interface Chip . . . . . . 94
4.6.2 Communications Interface Blocks . . . . . . 95
4.6.3 Self-Test of the Communications Interface . . 100
4.6.4 Chip Implementation . . . . . . . . . . . 105
4.7 Hand-Held Communicator . . . . . . . . . . 106
4.8 Fail-Safe Design . . . . . . . . . . . . . . . 108
4.9 Summary . . . . . . . . . . . . . . . . . . . 110

5 CONTROL PROCEDURES FOR THE DOUBLE SKIP-BRAID NETWORK . . 112
5.1 Representation of the Double Skip-Braid Network (DSBN) 113
5.2 System Start-Up Procedure . . . . . . . . . . 116
5.3 Fault Finding Routine . . . . . . . . . . . . . 120
5.4 Network Reconfiguration Algorithm . . . . . . 125
5.4.1 Forward Fault Search . . . . . . . . . . . . 126
5.4.2 Block Reconfiguration Routine . . . . . . 128
5.5 Polling Sequence Generator . . . . . . . . . . 131
5.5.1 Partial Path Listing Routine . . . . . . . . 133
5.5.2 Complete Path Listing Routine . . . . . . 134
5.5.3 Transmitter Assignment Routine . . . . . . 136
5.5.4 Message Listing Routine . . . . . . . . . . 139
5.6 Repair To The Network . . . . . . . . . . . . 143
5.7 Loop Controller Hardware . . . . . . . . . . . 144
5.8 Network Reconfiguration using the Proposed Loop Controller 146
5.9 Summary . . . . . . . . . . . . . . . . . . . 147

6 RELIABILITY ANALYSIS OF THE EXPERIMENTAL SYSTEM . . . . . . . 149
6.1 MTBF for a Single Fault-Tolerant Skip-Braid Network . . . . . . . . . . . . . 150
6.2 MTBF for the Loop Controller . . . . . . . . . 156
6.3 System Availability . . . . . . . . . . . . . . 157
6.4 Summary . . . . . . . . . . . . . . . . . . . 163
I am indebted to my supervisor Dr. J.R. Jordan for his advice and encouragement over the last three years, and to Dr. R.D.L. Mackie whose work on the original communications node circuit and interfacing the prototype chip to the node microcomputer was invaluable to the project.

I am grateful to the UK Science and Engineering Research Council (SERC) and Taylor Instrument for funding this work, and to NEC Ltd. at Motherwell, the SERC computing facility at Rutherford Appleton Laboratory, Dr. O.J. Davies at University College London and MEDL at Lincoln, for their assistance during the development of the communications interface chip.

I would like to acknowledge the support of all my friends and the staff here in the Department of Electrical Engineering, University of Edinburgh, especially my friends Keith Sylvan and Robert Stevenson.

Finally, I would like to express my gratitude to Dawn Galloway for proof-reading this thesis.
ABSTRACT

A 100 kbit/s battery-powered fault-tolerant communications network has been developed for use in industrial distributed measurement systems, where a loop controller supervises up to 64 addressable field devices with a network polling period of 250ms. Safety and reliability were optimized using fibre-optic data links and low-power circuitry throughout. Based on a highly redundant loop topology of two receiver/two transmitter communications nodes, the network can tolerate any double node or any quadruple link failure.

Each node circuit is designed to operate continuously for five years using a standard D-type lithium cell, and consists essentially of a CMOS single chip microcomputer, a specially designed CMOS communications interface chip, some analogue circuitry for the optical receivers and transmitters, and interfaces for a sensor/actuator and roaming hand-held terminal. The communications interface was implemented on a 2436-cell CMOS gate array and features a self-test facility which provides over 86% fault coverage using only three test vectors. The chip can also be used in the loop controller.

Control procedures have been developed to detect, locate, and reconfigure around faults that occur in the communications network. Single faults are located immediately by means of a time-out mechanism installed at each node and the loop controller, whereas more severe faults are located via a network scanning routine, typically in 200ms for a 32-node network. The necessary network reconfigurations are then determined by the algebraic manipulation of a set of Boolean connection matrices which describes the network. Using this technique, the loop controller can reconfigure around any multiple fault that does not include three adjacent node failures or any combination of link failures which results in the loss of three adjacent nodes.

A loop controller architecture was proposed based on the Novix NC4000 FORTH processor, which will enable single faults to be located and bypassed in a time comparable to the network polling period.

Finally, a quantitative reliability analysis of the system when subjected to different failure modes is presented, and reliability enhancing design techniques are suggested to achieve the target MTBF.
LIST OF ABBREVIATIONS

SSI  - Small Scale Integration
MSI  - Medium Scale Integration
LSI  - Large Scale Integration
VLSI - Very Large Scale Integration
RAM  - Random Access Memory
ROM  - Read Only Memory
PROM - Programmable Read Only Memory
EPROM - Erasable Programmable Read Only Memory
ALU  - Arithmetic Logic Unit
I/O  - Input/Output
EMI  - Electromagnetic Interference
RFI  - Radio Frequency Interference
PPC  - Pulse Position Coding
SSBN - Single Skip-Braid Network
DSBN - Double Skip-Braid Network
LAN  - Local Area Network
CAD  - Computer Aided Design
MTBF - Mean Time Between Failures
MTTR - Mean Time To Repair
MDT  - Mean Down Time
1 INTRODUCTION

This thesis describes a fault-tolerant fibre-optic communications network for use in industrial process control systems. The work forms part of a three year collaborative project between the Department of Electrical Engineering at the University of Edinburgh, Taylor Instrument (part of Combustion Engineering) and the Science and Engineering Research Council.

1.1 An Overview of a Process Control System

The function of a process control system is to regulate an industrial manufacturing process with the minimum of human interaction. Basically this involves measuring a series of dynamic process variables (e.g. pressure, flow, level), comparing these values with the desired values, and regulating those variables as required. The hierarchical structure of a typical modern control system is shown schematically in figure 1.1. This system conforms to the Open Systems Interconnection (OSI) model which is a set of standards being developed by the International Organization of Standardization (ISO) to define the communications procedure between systems. The process control system is supervised by a mainframe computer and an Application Computer (specific to the process being controlled), which together form the plant information system. The IBM ACS (Advanced Control System) is one such example [1]. This system communicates, via a set of gateway stations, with a number of distributed control loops, each of which interconnects several sub-loops, or distributed measurement systems. Essentially a network of input/output devices (usually remote field sensors or actuators) supervised by a programmable controller, the distributed measurement system represents the lowest level of control, either monitoring or regulating the dynamic variables of the process.

1.2 Standardization Efforts and the Field Bus Standard

Great effort is currently being expended to develop the Field Bus Standard which will define the interconnection scheme used for
PC = Programmable controller or Process controller

SPC = Slave programmable controller (5 to 10 steps) or Slave process controller (1 or 2 loops)

I/O = Analogue/digital input or output

Figure 1.1. Typical modern control system
the remote devices in a distributed measurement system. Many instrumentation manufacturers, customers and governments are committed to the OSI model which specifies telecommunications functions in seven layers ranging from basic communications (the physical connection of Layer 1) to information processing (the applications function of Layer 7). By complying with the OSI model, equipment from different manufacturers should be compatible and customers will be able to connect different equipment to suit their needs. The feasibility of the model was demonstrated at a National Computer Conference in Las Vegas in 1984 when equipment from fourteen different computer manufacturers was interconnected successfully to form a local area network (LAN). Standards are available, such as EIA RS 422/449 [2] and IEEE 488 [3], which are suitable for small networks. Larger networks are covered by MIL-STD 1553B [4], CCITT X.21 [5] and also by the IEEE 802 LAN standards [6] that are currently emerging. The fibre-optic version of MIL-STD 1553B, MIL-STD 1773, is still under development and the IEEE 802.8 Fibre-Optic Technology Advisory Group (F/O TAG) is investigating fibre-optic implementations of the IEEE 802 standards.

However, unlike MIL-STD 1553B which was designed for avionics applications, the new LAN standards are not aimed at the high reliability/low bit rate systems typically required for industrial distributed control. These systems may have to operate in flammable environments and so any standard must be implementable using intrinsically safe design techniques, which usually implies very low power operation. There are as yet no provisions for very low power communications networks and standard bit rates go no lower than 1 Mbit/s. Nonetheless, the IEEE standard does provide for a Media Access Unit (MAU) which will interface different equipment to a standard transmission line. Since the programmable controller acts effectively as the MAU for the distributed measurement system, the choice of communications system at the lowest level is in theory unrestricted. This flexibility enables power efficient signalling techniques and low bit rates, not accommodated by the above standards, to be used to achieve very low power consumption.
1.3 Functional Requirements of the Field Bus

The main distinguishing features of the proposed Field Bus are likely to be:

a) The bus will support low level devices which may have sufficient intelligence to execute simple functions (e.g. a control equation).

b) Devices on the bus will usually be low cost commodity items.

c) Information flow on the bus will usually consist of small packets.

d) The bus may have to operate in a flammable environment which requires intrinsically safe design techniques.

e) Serial digital communications will replace analogue signalling.

f) The bus will provide secure communications with low bit error rates (BER) and high immunity to radio frequency interference (RFI) and electromagnetic interference (EMI).

g) Where local power supplies are not available, the bus should power devices via the communications medium with a low power option to satisfy intrinsic safety requirements.

h) The number of addressable devices on one bus may exceed 30.

i) The standard will support loop and bus topologies, and will accommodate redundant network structures. The topology should also permit the addition or deletion of remote devices.

j) The standard will support twisted pair, coaxial and optical fibre communications media, but emphasis will be placed on reducing cabling costs and weight.

These features reflect the aspirations of the proposed Field Bus standard, but any standard must represent a compromise between several conflicting constraints. The main constraints are bus length, number of addressable devices per bus, communications data rate, data integrity, power budget, reliability, safety and cost.

1.4 The Distributed Measurement System

After consultation with Taylor Instrument, the following emerged as desirable features for the distributed measurement system presented in this thesis.

a) The system should be expandable and support a maximum of 64 remote devices (sensors or actuators).
b) Each device will be scanned at least 4 times per second.
c) The system should provide information flow with a low BER.
d) If each device is battery powered, the replacement lifetime should be at least 5 years.
e) The mean time between failures (MTBF) for the system should exceed 80,000 hours (about 9 years).
f) The system should be able to tolerate any double device or communications link failure.
g) Each device should provide for a sensor input or actuator output, as well as an interface for a roving hand-held communicator (ie. a portable terminal).
h) The maximum inter-device spacing will be 500 metres.

The potential benefits of fibre-optics have been generally realized or suggested for a variety of both analogue and digital industrial applications. Passive all-optical distributed measurement systems are attractive for reasons of safety and reduced cabling costs and weight. However, the technology is still relatively immature and the performance of these systems is, at present, limited by unwanted environmentally induced effects, extreme signal attenuation (for large systems using optical couplers and splitters) and non-linearities associated with the optical sensors. The optical actuation of control valves has also been demonstrated [7,8], but again the technology is still in the early development stages. Nonetheless, the all-optical distributed measurement system holds great promise for the future.

More immediate benefits are available if fibre-optic digital telemetry is used in conjunction with electronic field devices to achieve maximum immunity to electrical interference, and cabling costs, and weight, can be reduced by connecting remote devices via a "daisy chain", or loop, network. This hybrid implementation is also attractive because conventional signal conditioning techniques can still be applied. It was decided therefore that the distributed measurement system should use remote electronic devices connected via fibre-optic data links. To improve reliability, lifetime and safety, the system will also take advantage of the rapidly developing technologies of low power circuits and miniature long-life batteries.
This avoids the additional costs of running a power cable around the plant and galvanically isolating the remote devices.

A block diagram of the distributed measurement system is shown in figure 1.2. The system consists of up to 64 battery-powered remote communications nodes supervised by a loop controller. Each node contains a microprocessor and is capable of receiving data from, or transmitting data to, the loop controller. It also provides an interface for a sensor or actuator, and for a roving hand-held communicator. The electronic instrumentation required by the sensor or actuator will be powered from a separate battery, and will be the subject of a future study at Edinburgh University.

The work reported here presents an overview of the system, but primarily addresses two main areas. The first concerns the design of a very low power node circuit with a potential operational lifetime of five years using a reasonably small battery, say about 60 cc. The second involves developing a fault-tolerant communications network that is based on a loop topology but which can tolerate up to two remote node failures. The procedures for controlling such a network are also addressed, with the aim of developing a flexible control strategy suitable for use with different fault-tolerant networks.

Chapter 2 reviews design techniques which may be used to improve overall system reliability, and hence safety. Passive fibre-optic measurement systems are discussed, followed by a review of more conventional techniques such as fault-tolerant network topologies, error detecting/correcting codes, self-test and redundant system architectures. Fault-tolerant loop networks are investigated in Chapter 3 and the implementation of the low power communications node circuit is described in Chapter 4. Network control procedures developed for the experimental fault-tolerant communications network are described in Chapter 5. This is followed by a quantitative reliability analysis of the system in Chapter 6. Concluding remarks and proposals for future work are given in Chapter 7.
Figure 1.2. Block diagram of the proposed distributed measurement system.
The development of a distributed measurement system that is safe, reliable, and easy to install and maintain is currently of considerable interest to the nuclear and chemical process industries. Many techniques for improving safety and reliability have been suggested over the last twenty years. These include:

- improving immunity to EMI by better shielding or by using fibre-optic telemetry
- improving reliability by reducing the electronic component count and power consumption using state-of-the-art integration techniques
- using fault-tolerant design strategies at all stages of design, including self-test and redundant design techniques.

Optical measurement systems have also been of interest for reasons of safety and simplicity of design. Passive optical sensors can replace conventional sensors in many existing applications. Some authors have even proposed powering remote devices via fibre-optic data links [9,10,11] and a means of optical actuation has been investigated. Alternatively, all signal processing could be performed by the controller under relatively benign conditions if individual fibre links or a fibre bus were used to connect the transducers directly to the controller. Unfortunately, the first method is expensive in terms of cabling whilst the second requires expensive optical couplers and beamsplitters. Moreover, the number of sensors on a bus is severely limited by the splitting loss at each tap and the ability of the controller to discriminate between signals from the various sensors. Simpler DISTRIBUTED sensors are now being developed in which the fibre itself is the sensing element and is sensitive along its entire length. These devices offer the advantage of distributed measurement without the need for complicated optical components.

Non-invasive ultrasonic-based measurement systems have been of some interest in recent years. Ultrasonic devices are capable of measuring liquid or gas flow, fuel levels in containers [12] and acoustic emissions for condition monitoring purposes [13,14,15]. They are also well established in some medical [16,17] and security [18] applications. However, further development is limited by the performance of available transducer elements and new wideband
transducers are required which can couple efficiently to air. Problems caused by extreme signal attenuation and backscattering effects are still to be overcome.

This chapter reviews design techniques which may be used to enhance system reliability, and hence safety. It begins with a review of fibre-optic measurement systems suitable for use in hazardous environments. Techniques more applicable to conventional systems are then discussed. Fault-tolerant network topologies, error detecting/correcting codes and self-test methods are reviewed, and the chapter ends with a discussion of redundant design techniques which may be used to improve system availability.

2.1 FIBRE-OPTIC MEASUREMENT SYSTEMS

Since John Tyndall demonstrated the first light pipe in 1854, a myriad of fibre-optic applications has emerged. Many of the most useful applications are discussed by Kao [19]. Passive optical sensors have been demonstrated that can perform most sensing functions but widespread acceptance will occur only when long-term stability, reliability and robustness can be achieved. Complex systems requiring optical couplers, beamsplitters, delicate alignment, expensive lightsources etc. will not be acceptable. Industry requires a low-cost precision fibre-optic measurement system that can perform distributed measurements and can ideally be retrofitted.

2.1.1 Sensor Types

Fibre-optic sensors may be classified in a number of ways: by the physical variable being measured, by the way the light is modulated, or by the physical design of the sensor [20]. Classification by light modulation is commonly used, the principal categories being phase modulated and intensity modulated sensors. Phase modulation offers the highest performance and has been extensively researched for gyro and hydrophone applications. The most sensitive phase sensors are based on interferometric techniques where light that has undergone a phase shift in a sensing fibre interacts
with light from a reference fibre. The interference pattern produced can be analysed to achieve typical sensor sensitivities of $10^{-8}$ pascal for pressure and $10^{-5}$ °C for temperature. Various interferometer geometries can be used: the Mach-Zender interferometer for monomode fibres [21], the Michelson interferometer with twice the sensitivity of the Mach-Zender [22], the Sagnac interferometer (gyro) which measures the phase shift in contra-circulating beams in the same fibre [23], and the multimode interferometer which monitors phase changes in the various modes propagating in the fibre [24]. Improved sensitivity can also be achieved by coating the fibre in a jacket material sensitive to the measurand. For example, a magnetostrictive metallic glass jacket can improve sensitivity to magnetic fields [25]. Unfortunately, normal commercial fibres are themselves effective phase modulators. Environmental variations such as temperature, pressure, vibration, etc. can induce a phase shift in the propagating light which prohibits the transmission of phase modulated signals over even modest distances. In 1km of fibre the phase shift may exceed $10^5$ radians in a random fashion. Alternatively, the signal processing could be performed at each sensor and the information transmitted digitally, but due to the sophisticated signal processing required this would not be feasible in an industrial environment. Intensity modulated sensors, on the other hand, are less sensitive than phase modulated sensors, but the transmission of intensity modulated signals is relatively stable in commercial fibres. Typical environmental noise for an intensity modulated signal is less than 1dB/km.

2.1.2 Intensity Modulated Sensors

Intensity modulated sensors can be categorized into INTRINSIC, where the optical phenomena occur within the fibre itself, and EXTRINSIC, where the fibre is simply used as a light pipe. Although several extrinsic sensors have been built and tested, most industrial applications will eventually favour intrinsic sensors because of their simplicity. A commonly used modulation technique is to vary the amount of light coupled between two fibres by means of a shutter mechanism separating the fibres. Optical encoders are based on this
principle and can be used in applications such as satellite tracking, radar antennae, flight control and robotics [12]. Spillman [26] reported a multimode hydrophone which used a Schlieren grating to achieve sensitivities of 0.0025 μA, comparable to some monomode interferometer hydrophones. Angular displacement can be measured using Moiré fringe techniques as discussed by Vu [27]. A commercial vibration sensor which uses two coloured filters as the shutter mechanism was recently introduced by Cambridge Consultants [28]. There are also non-mechanical sensors based on the shutter principle which use the photoelastic effect or the Faraday effect to rotate the polarization of light passing through an optical polarizer [29,30].

Another way to vary the fibre-to-fibre coupling is to move either of the fibres. Spillman [31] constructed a hydrophone using this method with a sensitivity of 0.005 μA. He demonstrated that sensitivity can be increased by cutting both fibres at a slant angle and mounting them with a very small air gap so that Frustrated Total Internal Reflection (FTIR) occurs. An accelerometer has been developed using the same technique [32]. Optical coupling can also occur between lightguides which are sufficiently close to one another, as in twin core fibres [33] or optical IC waveguides [34], and this can be used to measure temperature, pressure and strain.

Light reflection is another method used to modulate the light coupled between fibres. One of the simplest designs is the liquid level sensor shown in figure 2.1a. It works by the elimination of a glass-air Total Internal Reflection (TIR) surface when the surface comes into contact with liquid. A Go/No-Go type signal is thus produced. This effect can also be achieved by monitoring the light loss of an exposed fibre core. The reflectivity of a TIR surface can be varied either by a change in refractive index or by bringing an absorbing material into contact with the TIR surface. Sincerbox [35] described a modulator based on Attenuated Total Reflection (ATR). The modulator, shown schematically in figure 2.1b, used a glass prism as the internal reflection element mounted with its base parallel and near to a flat silvered surface. By varying the distance between the prism and the silvered surface, the internal reflectivity was modulated. Using a polarized lightsource the reflectivity approached zero at about 900nm glass-silver separation. Disadvantages of this
Figure 2.1a. Liquid level sensor based on TIR

Figure 2.1b. Top: schematic of ATR modulator. Bottom: representative plot of reflectivity vs gap thickness for several angles of incidence on the base prism, for 632.8 nm p-polarized light.
modulator are that it requires a collimated lightsource and precise mechanical alignment. These problems can be abated by choosing a more lossy modulator material. The principal advantages of this type of modulator are its high frequency response and low drive power requirement. A similar device has been reported by Cook [36] and a commercially available device is claimed to exhibit submicron resolution over a 1mm range using a fibre bundle.

Taylor Instrument [37] investigated two improved versions of the ATR modulator. The first, shown schematically in figure 2.2a, uses a specially machined housing to hold a fibre in the shape of an arc. The fibre is then polished until sufficient core is exposed to form a TIR surface and a dielectric material is used to modulate the reflected light. Only one fibre is used and no glass prism is required, so it is therefore smaller and simpler in construction than the previous design. The second version is shown in figure 2.2b. The end of an unsheathed fibre is polished to form an asymmetrical wedge TIR surface. Light guided by the core strikes this surface and is reflected on to a silver mirrored surface. The light is then directed back onto the TIR surface and back along the core. This double pass technique effectively doubles the sensitivity of the modulator and considerably reduces the size of the device. However, double pass devices are extremely non-linear and suffer excessive signal attenuation caused by defects in the sputtered mirrored surface. A similar device was discussed by Philips [38] who developed a hydrophone that works on the principle that acoustic pressure alters the refractive index of water more than that of the fibre.

Other reflective sensors, such as a bifurcated or Y-guide probe, can be used to illuminate an object and collect backscattered light. Normally a bundle of illuminating and collecting fibres are used for better resolution. Applications include the measurement of displacements [39], flow rates (using correlation [40] or Doppler [41] techniques), vibration [42] and trace impurities [43]. The Fotonic sensor [44] is a popular commercial example. Reflective sensors have also been reported which use the sensitivity of a Fabry-Perot cavity to measure temperature, pressure and strain [45].

Microbending sensors exploit the principle that bending an optical fibre will cause some of the light propagating in the core to
Figure 2.2a. ATR single pass sensor

Figure 2.2b. ATR double pass sensor
propagate out into the cladding and subsequently be dissipated. By clamping the fibre between two plates with a periodic mechanical grating, a variety of variables can be measured [46]. Harmer [47] claimed device sensitivities of 1Å over a range of 10μm, having optimized the device using a multimode graded index fibre and a grating whose periodicity matched the mode coupling wavelength of the light propagating in the fibre. Many commercially available pressure switches use microbending loss. Switches have been produced by several manufacturers [48,49], that can be retrofitted as direct replacements for conventional devices. Another important intrinsic sensor is the "Fiberdyne" sensor [50], which relies on changes in the relative optical phases of the various modes within a multimode fibre to cause a variation in the average output intensity. Alternatively, the modulated signal can be detected by scanning a small aperture photodiode over the speckled output pattern produced by the mode mixing.

Many fibre-optic sensors have been proposed that measure either the absorbance, reflectance or luminescence of a material connected to the end of a fibre. For example, a 0.4mm diameter pH probe based on the reflectance of a dye indicator was reported by Goldstein [51]. Fibre-optic chemical sensors in general are discussed by Seitz [52]. Temperature sensors can be made by measuring the blackbody radiation emitted by a sapphire rod tip at the end of a fibre, now a National Standards instrument for the 500-2000°C range, or the absorption of semiconductors such as GaAs, Se, CdS, etc. [53]. A commercially available device [54], with a claimed resolution of 0.1°C over the range -50°C to 200°C, is based on the fluorescence of Europium. Alternatively, temperature can be measured by monitoring the time-decay of the fluorescence. This method produces a reading which is independent of system losses due to ageing. The birefringence of many crystalline materials can also be used to measure temperature in a manner independent of absolute intensity [55].

### 2.1.3 Distributed Optical Sensors

The widespread industrial acceptance of optical sensors has been hindered by their prohibitive cost and by the problems associated
with multiplexing these devices on a common fibre ring. A hybrid electrical-optical system is a possibility but for many high risk applications an all-optical distributed system is the only viable alternative to systems currently available. Such an optical system would be capable of monitoring a set of spatially separated variables without the need for any remote electronics, thus improving safety and reliability whilst lowering the cost per sensing point.

Optical Time Domain Reflectometry (OTDR) is a method for interrogating the status of fibre sensors whose operation is based on optical transmission or reflectivity. The basic configuration of an OTDR system is shown schematically in figure 2.3a. A pulse of light injected into the fibre is sequentially scattered by defects or anomalies as it passes through the fibre. Any backscattered light collected by the core can be detected via the beamsplitter. Scattering can be caused by small variations in the refractive index of the fibre, by defects such as bubbles or impurities, or by microbending. The subsequent refractive index variation results in "intrinsic Rayleigh scattering" which forms the fundamental limit to the fibre attenuation. Typically, the backscattered light results in a detector output as shown in figure 2.3b. The abscissa, corresponding to the pulse propagation delay time, represents the distance along the fibre and the ordinate indicates the round-trip loss of light from source to detector. Large spikes occur at either end of the curve due to Fresnel input and end reflections, whereas smaller spikes occur due to fibre splices for example. Localized attenuation, such as that caused by microbend sensors, results in a step down at the position corresponding to the sensor location. The number of sensors in an OTDR system is limited by several factors [56], but mainly by the dynamic range of the processing electronics and the fibre attenuation. OTDR techniques can be used to multiplex both intrinsic and extrinsic sensors.

York VSOP [57] recently announced a distributed temperature sensor which monitors the level of Raman backscattering in a fibre. The makers claim that 0.5°C resolution at 1m intervals over a 20km fibre will eventually be achieved. A cryogenic distributed temperature sensor, as yet with no spatial resolution, is also being developed by Pilkington Security Ltd. [58]. Herga Electronics [49]
Return Intensity Depends on: Scattering Mismatches Losses

Figure 2.3a. Block diagram of an OTDR system

Figure 2.3b. Typical detector output for an OTDR system
are developing a distributed pressure sensor based on microbending loss. Such a device could use OTDR techniques to measure local stress in pipelines, for security systems or for structural diagnostics [59]. Distributed sensors can also be designed for nuclear and plasma diagnostics. Gaebler and Braunig [60] described some of the effects induced by pulse irradiating a fibre and discussed how these could be measured using fibre-optics. Systems currently in use have been reported [59].

Multiplexing extrinsic sensors can be achieved using an N-port coupler with a different length of fibre supplying each sensor. This star arrangement ensures that signals returning from the sensors arrive temporally separated. Alternatively, sensors can be chained together on the same fibre. Theocharous [61] described a distributed temperature sensor that uses a series of ruby glass plates chained together by lengths of fibre. The strong temperature dependence of the absorption of the ruby glass was used in conjunction with OTDR to monitor the temperature at each plate location. A variety of methods have been proposed for the measurement of distributed variables. Polarization OTDR, Scattering OTDR, Optical Frequency Domain Reflectometry (OFDR) and Differential Absorption OTDR are some examples [59]. In yet another approach, proposed by Davies and Kingsley [62], and by Culshaw [63], ultrasonically induced phase changes were used to modulate the light in a fibre as a means of achieving fibre-optic communications without breaking the fibre loop.

In summary, many optical fibre sensors have been demonstrated with potentially very high sensitivity. Phase modulated sensors offer the highest sensitivity but are impractical because they require complicated signal processing and are susceptible to environmentally induced phase changes. Intensity modulated sensors are less sensitive but are better suited to the industrial environment. Passive optical measurement systems are desirable because they do not require any remote power sources apart from at the loop controller, but in most cases they also require a dedicated fibre from the controller to each sensor. Distributed optical measurement systems are emerging, based on TDR techniques, in which the fibre itself is used as the sensing element. Such systems appear very promising for the future, being at
present limited only by the electronic control system.

Bus multiplexing of remotely powered optical sensors is another possibility, but the number of sensors per bus is severely limited in low power applications by the high cumulative tap losses. Remotely powered sensors can also be multiplexed without optical connections by ultrasonically modulating the light propagating in the fibre. These systems are attractive, even for low power applications, but because they rely on phase information they suffer again from environmentally induced phase changes.

In all-optical intensity modulated systems, problems arise with unknown light losses from various sources such as LED-fibre-photodiode coupling, fibre attenuation and optical connectors. These losses are inherently time and temperature dependent and are therefore difficult to calibrate. Incorporating a reference fibre into the system is one possible solution but obviously results in a greater cabling overhead. Other problems concerning optical actuation are still to be overcome, especially at low power. Nonetheless, all of these problems are being investigated and with further systems engineering development, widespread application for optical measurement systems may be realized.

In the near future, a hybrid electrical-optical system with conventional sensors/actuators and an optical data highway still remains the most practical means of improving system safety and reliability within a very low power budget. The most immediate application areas for true optical sensors should be for temperature and small displacement measurements. Fibre-optic temperature sensors produce an absolute reading without the need for a temperature reference and with negligible susceptibility to electrical interference, which is the main problem with thermocouples. They are therefore easier to use and less expensive to install because no electrical shielding is required. Fibre-optic displacement sensors are also attractive because of their small size, fast response time, large dynamic range and, of course, their high immunity to electrical interference.
2.2 FAULT-TOLERANT COMMUNICATIONS NETWORKS

Over the last two decades several network topologies (figure 2.4) have been proposed using a variety of transmission media, communications protocols and signalling techniques. General surveys, with extensive bibliographies, have been presented by Clark [64] and Wittie [65]. To date, practically all implementations have been based on either bus or loop topologies due to their economy of cabling. Distributed network control, as opposed to centralized control, is also emerging as the most reliable control strategy for networks consisting of equally intelligent nodes. However, in other applications, such as process control, centralized control of a number of slave processes may still dominate. The interest in hybrid measurement systems which combine well proven conventional instrumentation devices with fibre-optic communications networks has grown steadily in recent years. The development of a low cost, very low power, fibre-optic communications network represents a departure from the established doctrine where high bit rates and expensive optical equipment prevail. The aim here is to develop an expansible, flexible, fault-tolerant network which is economical in terms of cabling and power consumption.

In this section a review of fault-tolerant network topologies is presented with emphasis on those which can be implemented easily in fibre-optics. The fault types associated with Local Area Networks (LANs) and error handling protocols are discussed, followed by a survey of fault-tolerant bus and loop architectures.

2.2.1 Faults Within a Network

Many types of fault are possible within a communications network and quite often different faults are manifest in an identical manner. Although standardization efforts are currently underway for bus and loop networks [6] the task of standardizing error handling protocols is still to be tackled. This task is hindered by the large variety of fault-tolerant network architectures currently under investigation. Fault tolerance is generally considered to be the ability of a network to tolerate line or node failures. This definition is
Figure 2.4. Example network topologies
adequate for describing the theoretical performance of any network topology where links and nodes are either operational or not, but real fault tolerance should be representative of the ability of a network to locate and isolate faults whilst maintaining near maximal system availability. Faults which cannot be isolated may be intermittent or unpredictable and can be disastrous to the operation of the rest of the network. Star networks are inherently tolerant to line and remote node failures, but are very susceptible to failures in the central node. Bus networks can tolerate individual node failures as long as they fail in a manner which is non-destructive to the main bus. Loop networks are the most vulnerable of the basic topologies, being intolerant to both single line or node failures.

The ease with which a fault can be isolated depends very much on the type of fault. Signal errors can occur due to noise on the communications line, such as EMI, and are usually infrequent and easily corrected. Transmitter errors can result from transmitter "jabbering" (i.e. spurious transmissions), frequency drift or power output drift. Reception errors can result from inactive or partially inactive receivers, or faulty data recovery caused by excessive frequency drift in the decoding circuitry (a condition which may be undetectable using a receiver self-test procedure). These errors are difficult to detect and often cause complete loss of the node. Node failures, such as "stuck-at" logic faults in the data decoding or repeater sections are also difficult to detect, as are software related failures in intelligent nodes, and these often render the node useless. Complete node failure can occur due to power supply failure or certain "stuck-at" logic faults, such as a permanent reset signal. Line failures occur due to intermittent connectors, opens or shorts in the transceivers and physical damage to the cable. These failures are often indistinguishable from those previously mentioned but can be minimized by careful design and network layout. These are only a few of the many sources of error in a communications network. Faults associated with communications networks in general are discussed in more detail by other authors [66,67,68].

One common and annoying trait of many network faults is that they are usually simple to detect but extremely difficult to locate. Often simple faults can be localized only to a single node which must
then be removed from the network.

2.2.2 Error Handling Protocols

Protocols are intended to control all aspects of fault detection, location and correction in communications networks. To date, two major categories of local network protocol prevail: the virtual token passing scheme and the Carrier Sense Multi-Access / Collision Detection (CSMA/CD) scheme. The first method passes a token around the network so that each node, upon receiving the token, can control the network. In the second method, each node senses the line before transmitting to avoid message clashes. Each node can also delay incoming messages during transmission. Should a message clash occur however, retransmission can begin after a specified wait time. Detection and correction of data corrupted by noise or intermittent faults can be achieved using suitable error detecting and correcting codes which are discussed in section 2.3. Messages can be rendered undeliverable by a node malfunction or by a collision of messages from different nodes, and could, in theory, circulate around the network indefinitely. One solution is for the sender to remove each message either by message recognition [69] or after a known time interval as is done in the Cambridge ring [70]. Here a monitor station recognizes lost messages by marking each message as it passes. Any message that passes the monitor twice is removed. In the Distributed Loop Computer Network (DLCN) [71] the detection and removal of lost messages is completely distributed. The loop is divided into two regions and each message contains a two-bit field which is filled with ones whenever the message has circulated the loop. The first node to detect this condition removes the message.

Message collisions can be overcome using an Automatic Repeat reQuest (ARQ) facility but this is both inefficient and time consuming. A better approach is to use CSMA/CD protocols [72,73]. Complete loss of signal or transmitter jabbering can be detected by incorporating a time-out mechanism in each node [69] or by providing a dedicated diagnostic link from each node back to a monitor station [70]. The approximate location of the fault can then be relayed to the controller for remedial action.
2.2.3 Fault-Tolerant Bus Topologies

The number of remote nodes on a global bus is limited by the message density and the electrical line characteristics. The size of a fibre-optic bus, on the other hand, is limited by the reciprocity and excess loss of the optical coupler taps connecting the nodes to the bus. The widespread acceptance of fibre-optic bus networks depends on the development of a reproducible low-cost coupler. Of the potential low-cost coupler technologies reviewed by Nelson et al [74], the fused biconical taper devices are the most versatile. Based on the high temperature fusion of two or more fibres, these couplers exhibit very low excess loss, typically 0.3dB for 100/140 µm fibre. Unfortunately, commercially available devices are very expensive.

In general, remote node failures do not affect the operation of the rest of the bus. However, certain failures such as a short on the transmission line, a jabbering transmitter at a node, or a break in the line itself, can be disastrous. Remote nodes can be effectively isolated from the line by transformer or optical isolators but a jabbering transmitter must be logically removed by the node or the bus controller. If a jabbering transmitter cannot be isolated, the bus may be rendered useless. For critical applications or where the risk of bus damage is high, as in a combat aircraft flight control system, a redundant bus structure can be used. MIL-STD 1553B provides for redundant bus structures but specifies that only standby redundancy be employed (ie. only one bus active at a time). Johnson and Julich [75] reported a highly reliable flight control system, designed for use in a combat helicopter, based on a dual 1553B bus with two controllers and two remote nodes per sensor/actuator. On power-up, a contention procedure causes one controller to take command of one bus while the other controller monitors concurrently all bus operations. Equivalency checks are performed between the controllers to guarantee the validity of critical computations. If the active controller fails, it ceases to issue polls, and the backup controller senses the silence and assumes command.

The Standard Field Bus proposal provides for redundant bus topologies but it is likely that active redundancy, with multiple buses simultaneously active, will also be supported for applications
where system availability is paramount. Indeed, Taylor Instrument [76] recently selected a dual active bus system for installation in the THORP nuclear reprocessing plant at Sellafield. Dual bus topologies have also been used in some high performance fibre-optic LANs. Fasnet, U-Net, Token-Less-Protocol (TLP) and Buzznet are some of the examples discussed by Nassehi et al [77].

2.2.4 Fault-Tolerant Loop Topologies

The reliability of loop networks is most easily improved by using high quality components and simple design techniques. Alternatively, loop networks can be made fault tolerant by incorporating a bypass mechanism at each node interface or by providing redundant communications paths throughout the network. The latter technique offers the best protection, coping with both node and line failures.

The node bypass architecture is shown schematically in figure 2.5. In the event of a node malfunction or local power supply failure the bypass mechanism is activated, effectively routing signals past the faulty node. Bypass switches must therefore be designed to be "fail-safe" so that rerouting can be performed even after a local power failure. This implies that switches are normally powered in the "route-to-node" position or powered independently. Switches must also be highly reliable since a single switch failure could render the network useless. In the Cambridge ring, the receiver and transmitter at each node are line-powered and can survive local power loss. The IBM 2790 [78] and SPIDER [71] loops consist of several small loops interconnected through a loop controller. When a fault is detected the controller simply shuts down the affected loop, thus performing the bypass operation. Several network implementations have been designed using bypass techniques. Serial Camac [79] has an external bypass switch which may be activated by command from the loop controller or if local power fails. In the SPIDER loop, each node is connected via a line access module which is powered from the loop. As with Serial Camac, the node can be isolated after local power failure or by the loop controller. IBM Zurich Research Laboratories recently developed a token-based loop [69] that can prune malfunctioning nodes.
Figure 2.5. Node bypass architecture

Figure 2.6. Parallel loop network (dual redundant loop)
using bypass relays. All of these implementations have been based on copper transmission media and so performing the switching operation is relatively straightforward. Fibre-optic loops, on the other hand, require efficient optical switches. Vlasak et al [80] reported an electro-optic switch with desirable characteristics: nanosecond switching, low power, low cost and small size. Prototype opto-mechanical and liquid-crystal optical switches have also been reported [81]. Brandsma [82] described a 20 Mbit/s fibre-optic LAN consisting of several small loops connected to the main loop via opto-mechanical bypass relays.

Loop networks can be designed so that more than one path exists around the network. Normally only one path is used, the others acting as standby paths. The simplest multi-loop architecture is the parallel loop shown in figure 2.6. In case of a failure in a segment of one loop, the other is used. This topology was used in a low-cost communications network described by Smith [83]. A more complicated fibre-optic multi-access network, called FOREMAN [81], has four parallel loops for enhanced reliability. Severin [84] discussed a similar system based on fused couplers.

Zafiropulo [85] examined use of a double loop in the case where a controller supervises system reconfiguration. Three reconfiguration strategies were described. The first method, the BYPASS technique, is shown in figure 2.7a. Both loops transmit in the same direction and the signal is routed on to the standby loop when a fault occurs in the main loop. The second method is the SELF-HEAL technique shown in figure 2.7b. Here the main and standby loops transmit in opposite directions. When a fault occurs two independent loops are formed, one either side of the fault. The hybrid technique is the third method. Here the bypass and self-heal actions can occur, but not simultaneously. All of these techniques, excluding the hybrid method, have recently been implemented in a 200 Mbit/s fibre-optic loop by Fujitsu [86] and a commercially available fibre-optic self-heal loop has been developed by Focon Systems Ltd. [87]. A standby loop was considered for the Cambridge ring but was rejected due to the required complexity of the reconfiguration hardware. However, standby loops have been used successfully in both the National Security Agency loop and the Karlsruhe loop [88].
Figure 2.7. Schematic of failure BYPASS and SELF-HEAL techniques

Figure 2.8. Skip-Braid Network
Loomis [89] has discussed reliability improvements by use of "node skipping" links as shown in figure 2.8. This loop, known as a braided network, has each node connected to both its immediate and penultimate predecessors. On failure of a node or link, a successor node can receive signals via the secondary link thereby bypassing the fault. This topology cannot tolerate two adjacent faults. However, fault tolerance can be improved by increasing the number of predecessors connected to each node but this obviously incurs a larger cabling overhead. This idea was developed further by Raghavendra et al [90], and by Arden and Lee [91]. A class of networks with near optimum fault tolerance, based on node skipping techniques, was also discussed by Pradhan [92]. Vlasak et al [80] investigated fibre-optic LAN topologies for tactical command and control systems and suggested that a braided loop network offered the best compromise in terms of reliability versus cost in high risk environments. In the SILK network [93], a braided ring is used to enhance reliability. An optional tertiary loop, incorporating longer skips, protects the system against failures which affect groups of nodes such as power supply failures in parts of a site.

An alternative to putting all nodes on one loop is to distribute them over several smaller loops controlled by a master loop. Single failures are then confined to an individual loop with nodes on other loops remaining unaffected. Zafiropulo [94] discussed the reliability of hierarchically structured multi-stage loops and has shown that there is an optimum number of stages for a given number of nodes and interface failure probability.

In summary, network reliability depends on the chosen topology and on the reliability of the constituent network components. Node bypass techniques work well in copper loops, but further development in optical switches is required before fibre-optic loops can compete commercially. Multi-loop topologies offer better protection and can be tailor-made to achieve the desired fault tolerance. Another factor affecting reliability is the communications control strategy. Centralized control is suited to a master/slave environment, such as process control instrumentation, where data transfers are usually sporadic and short in length. Distributed control, on the other hand,
makes better use of loop capacity and is more reliable since no single node is responsible for controlling the loop. However, it does require that each node has enough on-board intelligence to control the loop. In most applications, such as microcomputer LANs, distributed control performs well, but in others, especially in networks operating in hazardous environments where remote nodes cannot support high levels of intelligence due to cost or power budget constraints, centralized control may be preferred. Reliability problems associated with supporting high complexity devices in hazardous environments must also be considered in process control applications. A trade-off therefore exists between the benefits of distributed control and the additional node complexity required for its implementation.

In distributed measurement systems, centralized control can compete successfully with distributed control, and in applications where the remote instrumentation cannot support high levels of intelligence (for reasons of cost, power consumption or reliability), it may be the only practical choice.
Since their conception in the early 1950s error detecting and correcting (EDAC) codes have been used in a variety of areas, with particular impact in military and aerospace projects where the cost of failure far outweighs the extra development cost. As computer complexities grew and dimensions shrunk, the benefits EDAC provided for commercial data protection soon became apparent. EDAC codes, designed to operate on blocks of data words (semiconductor memory), are now widely used in mass storage systems where data integrity is important. Examples include the IBM 303X, 308X and 4300 series, Cray 1 and Tandem computers. A review of EDAC coding for semiconductor applications was made by Chen and Hsiao [95].

With the growth of long-haul digital communications has come the development of EDAC codes which operate on single data words. These serial codes have been widely used in fibre-optic communications, satellite communications and in more exotic systems such as telemetry for the Mariner and Voyager space probes. Other examples include the British Telecom radio pager and audio compact disk players.

This section discusses the principles of EDAC coding. The most common types of signal error are discussed before block codes and convolutional codes for serial data transfer systems are reviewed.

### 2.3.1 Codes and Channels

Different codes are suited to different transmission channels. Over a binary channel where data consists of '1's and '0's, noise may corrupt a message by changing a 1 to a 0 or vice-versa. Furthermore, depending upon the channel being used, noise induced errors may occur randomly in single isolated positions within a message or in bursts affecting several adjacent bits. When single isolated errors are most likely, simple coding schemes (such as single parity check codes) can be used to detect errors in the received message. However, simple coding schemes fail when burst errors predominate and special codes (such as polynomial codes) must be used to detect and correct them. Although several coding schemes exist, they can all be classified into two general groups, either BLOCK codes or CONVOLUTIONAL codes.
2.3.2 Block Codes

A block code is a set of rules that operates on k information bits before appending extra parity check bits to form an n-bit codeword. This would be an (n,k) binary code. The encoder examines a block of information bits and produces a codeword of fixed length. The simplest block code is the single parity check code (an (n,n-1) code) in which an extra bit is appended to each block of k bits to maintain constant parity. The message is decoded by recomputing the parity of the received message to determine whether or not an error has occurred. This code will detect all odd numbers of errors but will miss all even numbers and so would be unsuitable in a system susceptible to multiple errors. When an error is detected it cannot be corrected using this technique, but an Automatic Repeat request (ARQ) procedure could be initiated. However, retransmissions are time consuming and can be used only in systems where the probability of error is low or where the penalty for retransmission is small. The MIL-STD 1553B bus uses a single parity-check code and each message contains an error flag which is set whenever an invalid codeword is received. The bus controller monitors transmission errors and decides whether or not to retransmit. A single parity-check code is also used in the Cambridge ring. If retransmissions cannot be tolerated, an Error Correcting Code (ECC) can be employed. The simplest is the (3,1) repetition code which simply sends each bit three times in succession. Using a majority voting decoder any single error can be corrected. Similarly, a (5,1) repetition code can correct any pattern of two errors. Obviously, repetition codes require a large check bit overhead and so are inefficient.

2.3.3 Hamming Codes

The Hamming codes [96] are a family of single error correcting block codes. These (n,k) codes have m = (n-k) parity check bits so that m parity checking operations must be performed. The checking equations can be determined with the aid of table 2.1, which is simply a decimal to binary conversion table. The first check bit, corresponding to column a, is calculated by the modulo-2 addition of
### Table 2.1. Binary/Decimal equivalents

<table>
<thead>
<tr>
<th>Row no.</th>
<th>24</th>
<th>23</th>
<th>22</th>
<th>21</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Col. e d c b a</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0 1</td>
</tr>
<tr>
<td>0 0 0 1 0</td>
</tr>
<tr>
<td>0 0 0 1 1</td>
</tr>
<tr>
<td>0 0 1 0 0</td>
</tr>
<tr>
<td>0 0 1 0 1</td>
</tr>
<tr>
<td>0 0 1 1 0</td>
</tr>
<tr>
<td>0 0 1 1 1</td>
</tr>
<tr>
<td>0 1 0 0 0</td>
</tr>
</tbody>
</table>

**CYCLIC BLOCK CODE**

<table>
<thead>
<tr>
<th>Info. digits</th>
<th>Parity digits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1</td>
<td>0 1 1 1</td>
</tr>
<tr>
<td>0 1 0</td>
<td>1 1 1 0</td>
</tr>
<tr>
<td>0 1 1</td>
<td>1 0 0 1</td>
</tr>
<tr>
<td>1 0 0</td>
<td>1 0 1 1</td>
</tr>
<tr>
<td>1 0 1</td>
<td>1 1 0 0</td>
</tr>
<tr>
<td>1 1 0</td>
<td>0 1 0 1</td>
</tr>
<tr>
<td>1 1 1</td>
<td>0 0 1 0</td>
</tr>
</tbody>
</table>

**Table 2.2. Complete cyclic code set for 3 information bits**
the information bits at message positions corresponding to the row numbers of column a where a '1' is located. Therefore, checkbit one covers positions 1, 3, 5, 7, etc. The second checkbit, corresponding to column b, is similarly generated and covers positions 2, 3, 6, 7, etc. The third covers positions 4, 5, 6, 7, 12, 13, etc, and so on. As an example, consider the shortest non-trivial code in the family, the (7,4) Hamming code, with four information bits (1011) to be encoded. If the check bits are assigned to positions 1, 2, and 4 within the codeword, the four information bits will take the remaining positions. The complete codeword is:

\[
\begin{align*}
&b_7 b_6 b_5 b_4 b_3 b_2 b_1 \\
&= 1 0 1 0 1 0 1
\end{align*}
\]

which is transmitted. To demonstrate the error correcting properties of this code, imagine that bit \(b_4\) is corrupted by noise and changed to a 1. The decoder must recompute the checkbits, using the same checking equations, so that a comparison can be made with the received checkbits. The following checkbits will result:

\[
\begin{align*}
S_3 &\quad S_2 &\quad S_1 \\
&= 1 0 0
\end{align*}
\]

which shows that checks 1 and 2 have passed but check 3 has failed. These new checkbits form a syndrome which corresponds to row 4 in table 2.1. Hence, bit 4 in the received message is in error and is corrected. Table 2.1 can also be represented as a parity check matrix, \(H\), where:

\[
H = \begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0
\end{bmatrix}
\]

The syndrome can now be generated by the modulo-2 multiplication:

\[
\begin{bmatrix}
H \\
\end{bmatrix}
\begin{bmatrix}
b_7 \\
\vdots \\
b_1
\end{bmatrix}
= \begin{bmatrix}
S_3 \\
S_2 \\
S_1
\end{bmatrix}
\]

(2.1)

In the event of two or more errors, this coding scheme would fail. By adding one overall parity bit to the codeword, making it an (8,4)
code, two errors can be detected since the syndrome will indicate an error but the overall parity check will not. Although the percentage checkbit overhead is large for small values of $k$, it decreases rapidly as $k$ increases. Hamming codes are easily implemented both in hardware and software, and have been widely used in memory management systems [97,98].

Other codes, based on the Hamming principle, have been developed to correct double errors [99]. Unfortunately, they require a much greater checkbit overhead and are more complicated to decode. In view of this they will not be discussed further.

### 2.3.4 Cyclic Redundancy Check (CRC) Codes

Cyclic codes are parity-check codes with the property that a cyclic shift, or end-about shift, of a codeword results in another codeword. These codes are suited to mathematical analysis and are easily implemented by means of a modulo-2 divider circuit. Cyclic codes can be formed using parity-check matrices which have cyclically related rows but they may also be generated using a polynomial method. It is convenient to imagine a stream of $k$ information bits as a polynomial in a dummy variable $x$. The $k$ information bits representing the coefficients of the polynomial $M(x)$. To compute the checkbits that make up the rest of the $n$-bit codeword, a generator polynomial $G(x)$ is chosen so that the degree $r$ of $G(x)$ is equal to $(n-k)$. Moreover, $G(x)$ has a 1 as the coefficient for the $x^r$ term. A codeword can now be generated by appending $r$ zeros to the right hand side of $M(x)$ and modulo-2 dividing by $G(x)$ until the remainder $R(x)$ is obtained. The codeword then consists of the original $M(x)$ immediately followed by $R(x)$. As an example consider the formation of a codeword for the information bits:

$$M(x) = 100$$

using the generator polynomial $G(x) = x^4 + x^2 + x + 1$ or 10111. Appending four zeros to $M(x)$ and dividing by 10111 yields a remainder $R(x) = 1011$ and so the transmitted codeword is 1001011. The complete codeword set for the seven different information blocks is given in
Table 2.2, from which the cyclic relationship can clearly be seen. Cyclic codes can be decoded by recomputing the checkbits using the same generator polynomial. If the checkbits agree it is assumed that the received data is correct. Alternatively, the whole codeword can be divided by the same generator polynomial and it can be shown that the remainder will be zero if no errors are present [100]. In the previous example there were only 7 valid codewords out of a possible 128 combinations of 7 bits. This means that the probability of a multiple error causing a received codeword to be read as another codeword is extremely small. In other words, cyclic codes are excellent error detecting codes. With a prior knowledge of the transmission channel, generator polynomials can be chosen to minimize the probability of failed error detection. CRC codes are widely used in communications systems for error detection purposes and a number have been adopted as standards, for example CRC-12, CRC-ANSI and CRC-OCIT [101]. However, their use in High level Data Link Control (HDLC) protocols is not infallible as certain single bit errors can go undetected. Funk [102] discussed several ways in which these protocols could be improved. Error correction is a more complex task and can be performed by threshold detection decoding, permutation decoding or maximum likelihood detection [103]. The latter, which is the best technique, is achieved by cross-correlating the received codeword with a set of template codewords. The template producing the best correlation is assumed to have been transmitted.

Bose-Chaudhuri-Hocquenghem (BCH) codes [104] are cyclic codes well suited to correcting randomly distributed errors. A 32-bit BCH code is used in the British Telecom radio pager to ensure reliable reception over a channel subject to a variety of noise sources. Reed Muller (RM) codes are closely related to cyclic codes and can be constructed using a generator matrix. In practice, simple RM codes can be modelled as a message block which is transmitted to represent a 1 information bit, with the inverse message being transmitted for a 0 bit. The Mariner 9 space probe used a 32-bit RM code and a maximum likelihood decoder. RM codes can be used to construct another set of codes known as Golay codes which were discussed by Cusack [105]. Reed Solomon (RS) codes [104] are based on a slightly different use of polynomials with all information and codeword symbols belonging to a
finite field. The $k$ information symbols represent the coefficients of a generator polynomial which is evaluated for $n$ distinct values chosen from the field of symbols. The $n$ results then form the $n$ symbol codeword. RS codes are well suited to correcting burst errors and are used, for example, in audio compact disk systems. Other codes, such as Fire codes [106], have also been developed for this purpose.

2.3.5 Convolutional Codes

Unlike a block code, a convolutional code generates a continuing data sequence from a source information sequence. The transmitted sequence is obtained by convolving the source sequence with a fixed binary function and $r$ bits are transmitted for each source bit. This would be a rate $1/r$ convolutional code. As an example, consider the rate $1/2$ code which is generated by the circuit shown in figure 2.9a. Two binary functions, namely $b_{i+2} \odot b_i$ and $b_{i+1} \odot b_i$, are transmitted for each incoming bit. The function selector switch must therefore operate at twice the frequency of the input sequence. If the input sequence is 1100101 say, the transmitted output sequence, assuming the registers initially contain zeros, is 11,10,11,10,11,01,01. A convolutional code can also be represented as a tree, trellis or state-transition diagram. Figure 2.9b shows the state-transition diagram for the previous encoder. Here, the formation of a codeword can be modelled as jumps in state within the diagram and since the memory of the encoder is limited, the state-transition diagram is finite and settles to a steady state.

Convolutional codes can be decoded in a similar manner to block codes. A syndrome is computed at the receiver and is used to recover the information sequence. The dominant method uses the Viterbi algorithm [107] which is a minimum distance detector. This algorithm essentially compares the received sequence with all possible transmitted sequences and chooses the sequence which is closest. The EDAC capabilities of convolutional codes depend on the transmission rate and the binary functions chosen. Although error correction can be difficult for some convolutional codes, the codes are in general easily implemented and offer performance comparable to block codes of
Figure 2.9a. Convolutional encoder

Figure 2.9b. State transition diagram for fig.2.9a
similar redundancy.

2.3.6 Choice of Error Correcting Code

Choosing an error correcting code can be a daunting task considering the number of VLSI implementations now available. For most communications systems using a standard transmission medium, it would be beneficial in terms of component availability and maintainability to conform to an existing standard. CRC codes offer excellent protection and are readily available commercially, such as in the Fairchild 9401 encoder/decoder and AMD AM28065 Fire processor. Moreover, using an efficient error detecting code in conjunction with an ARQ facility may be adequate for many applications. However, in some critical control applications, especially those where a fast dynamic response is important, the delay associated with retransmission may be undesirable. Data transfers in these systems may also be sporadic and short in length. In applications where ARQ techniques cannot be applied, an error correcting code must be used that exhibits error correcting properties sufficient to maintain the average bit error rate for the data link at an acceptable level. Furthermore, the code must be efficient in terms of the required check bit overhead so that the mean power consumption of the transmission and reception circuitry can be kept low.

It is envisaged that single error correction with double error detection is sufficient for most industrial process control applications where fibre-optic telemetry is used. To achieve this, Hamming codes with an additional parity bit, which are easily implemented both in hardware and software, are attractive in terms of checkbit overhead and decoding complexity (and hence power consumed). Hamming code devices are available such as the AMD AM2960/AM28160, Motorola MC34040/MC68540 and Texas Instrument SN54/74LS 630/631 series, but in many cases custom hardware or software implementations may be more convenient. Indeed, in very low power applications, power consumption can be minimized by using a custom hardware implementation, thus avoiding the superfluous features offered by off-the-shelf devices.
2.4 BUILT-IN-TEST METHODS

Built-in-test (BIT) has obvious application in instrumentation systems where high availability and low manufacturing and maintenance costs are essential. By systematically pruning malfunctioning units in favour of operational standby units, without the need for expensive ATE and protracted test times, BIT can improve system availability and reduce maintenance costs. Manufacturing costs can be similarly reduced as BIT greatly simplifies the screening process at all stages of manufacture. In fact, in all but the most critical applications, it is the capacity of BIT to reduce costs which has stimulated its development in recent years. Over the last decade the complexity of integrated circuits has grown dramatically and this has brought with it the problem of testing increasingly complex devices using a limited set of input and output pins. This restriction in controllability and observability has encouraged some designers to adopt a "design for testability" philosophy which is aimed at easing the verification process of the final product. The testing problem basically consists of generating test vectors and verifying output response patterns. This task is relatively straightforward for highly structured devices, such as semiconductor memories, but for most LSI and VLSI devices, which contain both combinatorial and sequential logic, the task is more difficult.

This section discusses some of the most popular testing strategies and design methodologies which are used to ease the testing problem.

2.4.1 The Stuck-at-Fault Model

To date, most of the work in logic circuit testing has been concerned with the stuck-at-fault model which assumes that during a fault condition, a logic gate input or output is stuck either at a logic 1 or a logic 0. Another commonly used assumption is that a machine is either in a good state or has only one stuck-at-fault. The latter assumption greatly simplifies fault simulation routines as the number of combinations of multiple faults in a large circuit can become excessive. In practice, however, it is known that this model
fails to represent other faults that may exist in a circuit. One such example is the logic short circuit or bridging fault, which has the effect of causing a wired-AND or wired-OR between the signals on the shorted leads and can change combinatorial circuits into sequential circuits. This problem is further aggravated by the increased density of modern VLSI devices. Kodandapani and Pradhan [108] indicated that certain undetectable bridging faults could invalidate stuck-at-fault tests and outlined the main potential problems. In response, Yamada and Nanya [109] reported that any complete test set for single stuck-at-faults in a two-level irredundant AND-OR circuit still remains valid in the presence of undetectable bridging faults. A simple technique for detecting feedback bridging faults was reported recently [110] in which a bridging faulty circuit was forced to exhibit stable sequential behaviour by applying only two test patterns. This work represents the first correspondence concerning the problem of test generation for internal feedback bridging, a field now receiving considerable attention. Nevertheless, the stuck-at-model still remains the most efficient fault simulation model and can be adapted [111] to accommodate the majority of faults that may occur in LSI and VLSI devices.

2.4.2 Design For Testability

The testing problem can be described in terms of controllability and observability. Controllability is a measure of the ease with which the circuit under test (CUT) can be configured into a desired state using only the available input test lines. Observability, on the other hand, describes the accessibility of the output responses to the response analyser. Since the control and observation of a circuit is fundamental to the test procedure, a number of programmes have been written to help analyse the controllability and observability of different circuits [112]. The two basic testability approaches to date are the AD HOC approach, which can be used to improve the testability of an existing device, and the STRUCTURED approach that attempts, via a set of design rules, to produce a device structure which is inherently more efficient to test.
2.4.3 Ad Hoc Testability Methods

The most important ad hoc techniques are based on a "divide and conquer" approach. By adding additional test points, long sequential circuits can be broken down into smaller sub-circuits which are easier to test. Since the task of test pattern generation is proportional to the cube of the number of logic gates [113], simply partitioning the circuit into two equally sized modules reduces the fault simulation task by a factor of four. Partitioning can be accomplished in a number of ways. Mechanical partitioning splits a circuit into smaller interconnected modules but is significantly more expensive and prone to interconnection failures. Observability can also be improved by using jumper wires to isolate modules for test purposes but this implies a greater number of Input/Output (I/O) lines, which again is expensive. A better method is logic partitioning in which a circuit is sub-divided using on-board hardware. Multiplexers can then be used to test each module via the same set of I/O lines. Buehler and Sievers [114] demonstrated the partitioning approach to testing, using a simple adder circuit. The procedure consisted of partitioning the circuit into two modules, generating a test sequence for each module while treating it as a stand-alone circuit, and combining individual test results to create a total system test. This technique was also applied to a 4-bit ALU and effectively reduced the number of tests from 16,384, for the exhaustive case, to only 356. Although circuit partitioning can reduce the number of test vectors required, it is important to note that automatic partitioning and test pattern generation may not be optimal for certain circuits [115]. This is because automatic test pattern generator (ATPG) programmes use only a gate level description of the CUT, with no knowledge of the functional structure. The partitioning strategy used may therefore be unsuitable for some circuits which have natural functional partitions.

Bus-structured devices, like microcomputers, are very well partitioned. Since the bus has access to all modules, it makes an obvious test point. If there is external access to the bus and modules not under test can be isolated (their outputs made high impedance), the bus can be used to stimulate the module under test
and collect its output responses. A major disadvantage of this approach is that a fault on the bus itself could invalidate all other test procedures. Isolating a bus failure usually requires current sensing techniques [116] and is often very difficult.

2.4.4 Random Pattern Testing Methods

Several testing methods based on data compression techniques have been proposed for reducing the large memory and testing time requirements associated with exhaustive testing. The most common method involves stimulating the CUT using a pseudo random binary sequence (PRBS) generated by a linear feedback shift register (LFSR). The output response bit sequence is then compressed to form a "unique syndrome" or "signature". Barzilai et al [117] reported a syndrome testing method based on counting the number of '1's (transition counting) in the output response. Thus, instead of storing the full output response, only the count need be stored.

The Signature Analysis data compression technique was first used by Hewlett Packard Ltd. in 1977 [118]. By "exclusive-OR"ing the output response data stream into an LFSR, the data stream is effectively divided by the characteristic polynomial of the LFSR. After the division, the n-bit shift register contains the remainder "signature" of the CUT. Signature analysis has excellent fault detecting properties and is superior to transition counting since the fault capture rate, and the extra hardware needed, are insensitive to the length of the data stream for data streams larger than n [119]. However, since long data streams undergo a considerable compression, the signature obtained is basically a Go/No-Go type signal. Fault location is most effectively achieved by sensibly partitioning circuits so that no closed loop paths exist. Signature analysis has been widely used to test boards containing LSI and VLSI devices and its use has been well publicized [120,121].

2.4.5 Structured Testability Methods

Structured approaches to testability involve following design philosophies specifically to improve the testability of the final
product. Such philosophies impose certain constraints on the design flexibility and must be adopted from the earliest stages of design. However, the ease with which these designs can be tested has proved to be sufficient incentive for many of the large manufacturers, such as IBM, Fujitsu, NEC etc., to adopt this approach.

The "scan path" technique is the most common approach to systematic test. Level-Sensitive Scan Design (LSSD) combines scan path techniques with a race-free design and is the test approach used by IBM. The basis of the scan path technique, shown schematically in Figure 2.10, is the ability to reconfigure internal memory elements into a shift register scan path whose input and output are connected to the main system input and output respectively. In test mode the latches of the register can be tested by passing a test sequence through the register itself, whilst the combinatorial circuitry can be tested using a test pattern supplied via the scan path. By proper design, an LSSD structure can be thought of as purely combinatorial, which enables well established testing methods such as D-cubes, Boolean Difference or SPOOF (Structure and Parity Observing Output Function) [122], to be employed. Although the LSSD latches are more complex than simple latches, the overhead can be made acceptably small, perhaps a few percent, by careful design.

A similar method, simply named Scan Path, was adopted by NEC. This method uses only one system clock to control "raceless" D-type flip-flops configured to form one scan path, unlike LSSD which is truly raceless using two system clocks. Control flip-flops are also used to isolate individual circuit modules during test time. Scan/Set Logic is yet another method, used by Sperry Univac and Motorola, in which a scan path exists which is itself not part of the normal system data path. This independence from the normal system means that a "snap shot" of the system can be obtained more quickly and at any instant in time. The obvious disadvantage of this technique is the test hardware overhead which is used purely for circuit test.

The last scan test method to be discussed here is the Random Access Scan used by Fujitsu. This method is similar in principle to the previous techniques except that shift registers are not employed. Instead, an addressing scheme is used which enables each latch to be uniquely controlled. Unlike Scan Path and LSSD which require two
Figure 2.10. Schematic of an LSSD subsystem with two system clocks \( C_1 \) and \( C_2 \). (\( L = \text{latch} \), \( SRL = \text{shift register latch} \))

Figure 2.11. Block diagram of the BILBO test method
latches for every observation point, the overhead of this method is about four gates per storage element but obviously more external control pins are needed for addressing purposes. Scan test methods are reviewed extensively by Williams and Parker [113].

2.4.6 Self-Test Methods

By storing a test programme in ROM, a microprocessor can be used to test its own memory, registers and support circuitry, with the minimum of external test equipment. Moreover, tests can be performed on-line or off-line depending on the application. On-line tests usually occur in systems that cannot tolerate any form of down-time. Such systems may also employ redundant design techniques to improve availability, as discussed in section 2.5. In general, however, on-line self-test does not provide a high degree of fault coverage but provides instead a fast Go/No-Go functional test. Random test patterns, on the other hand, can be used off-line in conjunction with Signature Analysis to provide a high degree of fault coverage in a well partitioned circuit. Test patterns can be stored either in ROM or generated internally using an LFSR. Due to the high storage overhead, individual test patterns are not normally stored in ROM.

The Built-In Logic Block Observer (BILBO) test method is well suited to VLSI structures. BILBO makes use of two LFSRs, one as a PRBS generator and the other as a signature analyser. The signature can then be compared either internally or externally with a known good signature. The BILBO approach is shown schematically in figure 2.11. As in scan path methods, the circuit can be configured into normal operation mode or self-test mode. During the first phase of self-test, LFSR1 generates a number of test patterns which are used to stimulate the combinatorial logic of network 1. The response is stored in LFSR2. After a fixed number of test patterns have been applied, the signature can be scanned out of LFSR2 and compared. The system then enters the second phase of self-test where LFSR2 is the PRBS generator and LFSR1 is the signature analyser. This duality of function greatly simplifies the layout of the test control circuitry but means that the BILBO registers are slightly more complex. An important point to note is that both LFSRs must be preset (or seeded)
to a known value before the test commences to ensure validity of the final signature. Resnick [123] reported a 6k gate array with BILBO facilities. The control logic occupied about 12% of the die area and the only external control items required were the LFSR seed, the number of clocks the test should run for, and the expected signature.

2.4.7 Alternative Test Methods

It may now be appropriate to discuss some alternative design for testability approaches that possess certain unique features. A key element in many testing procedures is the comparator. A totally self-checking comparator was reported by Mercier [124] which tests all gates and connections during normal circuit operation. Should a fault occur in any part of the circuit, except the actual output wire, the circuit locks itself into a state which indicates a fault has occurred. An algorithm for producing totally self-checking circuits from their normal Boolean description has been reported [125] but these circuits are typically twice as complex as their non-checking counterparts. However, an important class of self-checking circuits for data encoded in m-out-of-n format exists, for which the complexity penalty is negligible [126]. A similar technique can also be applied to sequential machines. Mukai and Tohma [127] described a method for realizing fail-safe sequential circuits which, on the occurrence of a single stuck-at fault, locked circuit outputs into a normally unused state. This implies that the outputs are always valid even in the presence of a fault.

A novel "fail-safe" microprocessor, called VIPER [128], has been developed at the Royal Signals and Radar Establishment, Great Malvern. The device emerged as a result of work on software validation projects and was based on a formal mathematical specification. If an illegal operation or output occurs, the microprocessor stops immediately since this is the only safe procedure in critical applications. Although the performance of the device is limited, for example it has no stack or multiplication / division functions, it does reduce test times by two orders of magnitude.
In summary, the use of design for testability philosophies will continue to grow. Ad hoc methods are popular at present but do not provide a general solution to the testing problem. Structured test methods do provide the solution, albeit using restrictive design rules. Comparative studies have been made by Buehler and Sievers [114], with the most commonly compared parameters being hardware overhead, fault coverage, test time and flexibility. Functional testing remains popular especially where fast Go/No-Go type tests are required, or where the test time or power budget is limited. However, functional testing results in a lower fault coverage and is labour intensive due to its ad hoc nature. In view of this, the use of structured methods should continue to grow and with the increasing availability of low-cost VLSI devices, users should soon have greater access to CAD facilities, such as gate array workstations, that provide block libraries already designed for testability. The UK5000 gate array [129], for example, was a step in this direction and was in fact the technology chosen for the VIPER processor.

Design for testability should eventually establish itself in all forms of industrial instrumentation, with systems becoming more commercially competitive as a result. The use of self-test during system operation depends greatly on the target application and a trade-off clearly exists between the cost of incorporating self-test and the benefits attainable. On-line testing is the most attractive strategy but is an inherently functional approach. Off-line testing can be exhaustive but lowers the immediate availability of the system unless a standby unit momentarily replaces the unit under test. However, since off-line testing is usually performed periodically as part of a condition monitoring programme, it will in most circumstances increase the overall availability of the system by predicting potentially dangerous fault conditions. The most immediate application for self-test is in the loop controller, where a combination of on-line test, perhaps using watchdog timers, and off-line test, such as exhaustive RAM cell tests, could be used in conjunction with redundant design techniques to improve reliability. The applicability of self-test in the remote communications network is limited mainly by power budget and data storage constraints, but may be used at system power-up or during system repair.
2.5 REDUNDANCY IN FAULT-TOLERANT SYSTEMS

For many years now manufacturers have relied on the use of premium quality components to optimize system reliability. However, applications exist such as process control and flight control systems, where the required availability cannot be achieved using conventional design strategies. Fault-tolerant design improves system availability by masking certain faults as they occur. As a result, maintenance can usually be deferred until scheduled service dates, which greatly simplifies logistics support, and hence reduces costs.

There are four general categories of redundancy which are used in fault-tolerant design. The first, information redundancy, was discussed in section 2.3. The second, software redundancy, broadly refers to those programmes which would not be needed if the system were guaranteed to be fault-free. These include back-up copies of critical programmes and data, built-in-test and system reconfiguration programmes. The third, time redundancy, involves performing critical calculations more than once to ensure the validity of the result. Often, such calculations are performed to a different number base or using a completely different algorithm to improve the fault coverage. The fourth category is hardware redundancy, which has already been widely used by the large semiconductor memory manufacturers to increase yield. By providing redundant cells which can be activated by fuse or laser programming, the functional device yield can be improved by an order of magnitude, in some cases, using only a small number of spare cells [130].

2.5.1 Dual Redundant Systems

Fault tolerance is most commonly achieved using dual redundancy in which critical units are duplicated and connected via switching circuitry to the main inputs and outputs. When a fault is detected in the primary unit, the secondary unit (which can either be a "cold" (standby) or "hot" (active) spare) takes over until the primary is repaired. The reliability of such systems depends greatly on the reliability of the error detecting and switching circuitry. If perfect error detecting and switching circuitry is assumed, system
reliability can best be improved using cold spares since the failure rate for an inactive unit is lower than for an active unit. However, for real systems where no circuitry is perfect, hot spares may be preferred since error detection can be achieved simply by comparing the outputs of both units and fault isolation is required only when a fault is detected. It is difficult to compare these two techniques since the success of each depends very heavily on the primary unit. Nonetheless, some general observations can be made. Using dual active redundancy, system Mean Time Between Failures (MTBF) can be improved by about 50% [131]. Better performance can be achieved using standby redundancy, but improvements are governed by the standby failure rate and the complexity of the error detecting and switching circuitry, which may grow with the complexity of the primary unit. This limits the optimum number of spares for any system. System MTBF also increases as the Mean Time To Repair (MTTR) decreases. Reliability can be improved further by partitioning the units into sub-units interconnected as shown in figure 2.12, but this is beneficial only when the main units have reasonably high failure rates. One disadvantage associated with all dual redundant systems is that there is a period during switchover when system operation is momentarily lost. This time depends on the efficiency of the fault location and switching procedures, not to mention the recovery period which follows when corrupted data or lost messages are retrieved. In some critical applications, this loss of control cannot be tolerated and greater fault tolerance must be employed.

2.5.2 Triple Modular Redundancy (TMR)

Using TMR, the simplex (single) unit is replaced by three identical units each performing exactly the same function as shown in figure 2.13. All three units are active and the system output is obtained by a majority vote. Therefore, if one unit fails, the voter still produces the correct result. For the system shown in figure 2.13, the reliability, \( R_{\text{TMR}} \), can be written as:

\[
R_{\text{TMR}} = R_v \cdot \left( \frac{3R^2 + 3R(1-R)}{3R^2 - 2R} \right)
\]

(2.2)
Figure 2.12. Modular Redundancy

Figure 2.13. Triple Modular Redundancy (TMR)
where $R$ is the unit reliability and $R_v$ is the voter reliability.

Although requiring a larger hardware overhead than dual redundancy, TMR does provide single unit fault tolerance without interruption to the operation of the system. It also reduces the task of fault location to a simple comparison between the output from the suspect unit and the output of the voter. Better fault tolerance can be achieved by partitioning the units into several sub-units connected via intermediate voters [132] as shown in figure 2.14, but improvements are again limited by the reliability of the voters, especially when large numbers of voters are present in a system.

The generalized form of TMR is N-tuply Modular Redundancy (NMR) in which $N=2^t-1$ ($t$ is a positive integer) parallel units perform concurrently the same functions. The output is again obtained by majority vote which now must exceed $t-1$. Therefore, $t-1$ unit failures can be tolerated. The reliability of an NMR system, $R_{nmr}$, is given by:

$$R_{nmr} = R_v \cdot \left( \sum_{i=0}^{t} \binom{N}{i} R^i (1-R)^{N-i} \right)$$

which can also be improved with a suitable partitioning strategy.

### 2.5.3 Hybrid Redundancy

By combining the concepts of NMR and standby spares, hybrid redundancy can achieve reliability levels unattainable using conventional NMR techniques. Figure 2.15 shows a hybrid system in which a switching network is used to select three out of the $N$ available units. These units form a TMR core whose outputs are voted on to produce the system output. When a failure occurs in one unit of the core, the dissent detector [133] monitors the disagreement between the faulty unit and the output of the voter, and removes the unit from the core. A spare unit, if one exists, is then switched in to maintain the TMR core. The reliability of this hybrid redundancy system employing a TMR core and $N-3$ spares is given by:

$$R_{hyb} = R_v \cdot R_{sw} \cdot \left( 1 - N R (1-R)^{N-1} - (1-R)^N \right)$$
Figure 2.14. a) TMR  b) Modular TMR

Figure 2.15. Hybrid redundancy with TMR core
where $R_{SW}$ is the switch reliability. As can be seen from the previous reliability equations, the reliability of redundant systems is usually evaluated by multiplying the system reliability, assuming perfect switches and voters, by the reliability of the switches and voters. This oversimplified calculation invariably produces a pessimistic estimate of system reliability since each switch or voter failure is assumed to be fatal. Although this results in a lower bound on system reliability, which is generally acceptable, a more accurate estimate can be obtained by using better reliability models for the switch and voter.

Based on the hybrid system (TMR core and two spares) with iterative cell switches shown in figure 2.16, Siewiorek and Ingle [134] developed a more realistic model for the switch. The iterative cells determine the first three operational units and assign them to be voted on. The $(i)$th cell receives a count representing the number of operational units already connected to the voter, from the $(i-1)$th cell, and the condition of the $(i)$th unit. If the $(i)$th unit is operational 'AND' the TMR core is not fully assigned, the $(i)$th cell connects the unit to the voter. Otherwise, the cell simply passes on the count. Although the voter reliability remains constant in this example, the switch reliability varies with the number of units, $N$. Therefore, the assumption that switch complexity is independent of $N$ is clearly unrealistic. By assuming that switch complexity increases linearly with $N$, Siewiorek and Ingle showed that switch reliability is a significant factor in the overall system reliability and that for most practical systems there exists an optimum number of spares (typically 2 or 3). These results agreed with the earlier work of Ogus [135]. It was also shown that hybrid systems perform better than TMR systems as long as the switch reliability remains much greater than the unit reliability. A previous study [136] compared various switch designs and suggested that iterative cells with threshold voters are superior for up to three spares, but majority voters are superior for four or more spares. Recently, Koren and Shalev [137] investigated switch design using a gate level model that considered the reliability of each gate individually. They calculated system unreliability by summing the probabilities of fatal faults in the basic unit, voter and switches. The result was then used to estimate
Figure 2.16. Hybrid system (TMR core and two spares) with an iterative cell switch.
the overall reliability of the system. Compared to previous studies, this technique yielded more realistic results and suggested better reliability improvements using hybrid redundancy.

2.5.4 Synchronizing Redundant Systems

The output from a voter in a redundant system is valid only if all voting units are synchronized. Synchronization is most commonly achieved using a common external reference, independent accurate clocks or mutual feedback. The first method is the most popular and has been widely implemented [138,139], but is vulnerable to failure of the common reference. This has encouraged the development of fault-tolerant clocking systems and several designs have now been proposed [140]. Using the second method, each unit is independently governed by its own internal clock. However, even the most accurate clocks cannot guarantee to keep systems in synchronization for long periods of time. The last method, mutual feedback, is attractive in terms of reliability since no single point of failure exists. Davies and Wakerly [141] described a synchronization technique, called synchronizing voting, in which each unit informs all synchronizer networks (there is one synchronizer for every unit) that its output is ready. Only when the outputs from all units are ready does the synchronizer allow the vote to take place. Moreover, the units are held, via feedback from the synchronizers, in that output state until the vote has been made. Following on from this work, McConnel and Siewiorek [142] discussed the practicalities of using synchronizing voting in a computer based environment, suggesting that the technique is viable only when voters are used on critical lines.

Synchronization is not the only problem associated with redundant systems. Kameyama and Higuchi [143] discussed one type of failure, known as a dependent failure, where the same failure occurs simultaneously in two or three units in the TMR core. This type of failure necessitates the use of periodic resynchronization in which all units are forced into a predefined state. In systems prone to frequent dependent failures, TMR cannot be used to improve system reliability and other techniques, such as time redundancy, must be employed. However, in most cases good system design can reduce the
probability of dependent failures to a negligible level.

In summary, reliability can be improved using redundant units which either run continuously or are powered-up only when necessary. For very low power applications, the latter approach may be preferred. Fault tolerance usually improves as the hardware overhead increases, but the complexity of the switching and voting circuitry, which depends on the structure of the simplex unit, limits this improvement. Reliability estimates are invariably pessimistic but more accurate models are now producing more realistic results. To date, fault-tolerant systems have been applied mainly in ultra critical environments [144,145] where the extra hardware and development costs are of secondary importance. However, commercial systems, aimed principally at the business transaction market, are now appearing and are becoming more competitive. For example, Tandem Nonstop [146] uses a dynamic resource sharing architecture that re-assigns tasks away from faulty devices. However, this approach requires a 30% software overhead per processor to perform the housekeeping functions and is not truly hardware redundant. The Stratus FT250 [147], on the other hand, uses dual active redundancy and, unlike software based fault tolerance, suffers no operational degradation when a fault occurs. Similar systems are also available from NoHalt Computers UK [148], Synapse and Computer Technology Ltd. [146]. An excellent review of fault-tolerant systems in commercial applications was made by Serlin [149]. August Systems Inc. have developed a TMR based system, the CS330 [150], for use in critical process control applications. An interesting feature of this system is that for non-critical tasks the three processors can run in simplex mode to enhance throughput, but for critical tasks they are synchronized together to improve reliability.

In the distributed measurement system, the availability of the loop controller can be improved by using fault diagnostics and standby spares. TMR can also be used in the remote network to improve system availability, but the number of sensing/actuating points per network is reduced effectively by a factor of three. In critical applications, however, the benefits of using TMR usually outweigh the additional costs it entails.
3 FAULT-TOLERANT LOOP NETWORKS

A distributed measurement system typically consists of a number (up to 64) of remote sensor/actuator devices supervised by a loop controller. For design purposes it can normally be assumed that the controller operates in a relatively benign environment compared to the remote devices, and is not subject to the same stringent power constraints. The remote devices and communications network, on the other hand, often have to operate in a hazardous environment and must be relatively inexpensive to implement, inherently safe and very reliable, to be commercially competitive.

For reasons of safety and reliability, a very low-power design methodology was adopted throughout this project and optical fibre was chosen as the communications medium. With regards to the communications network topology, a star network was considered too impractical in terms of the amount of cabling required by large networks, and an optical bus network was considered too vulnerable to bus failures. The latter network also required expensive optical couplers, and large transmitter power levels to overcome multi-tap splitting losses. In view of this, a fault-tolerant loop network with active regeneration will be used.

In situations where the consequences of failure are extremely serious, industry requires that the network should tolerate any combination of double fault [151] (a fault being defined as the loss of a remote communications link or node) without loss of communications between the controller and the remaining unaffected nodes. It should also be suitable for use in a wide variety of applications and for areas exposed to an undefined hazard rate. Therefore, a random hazard rate will be assumed. As well as being easily expanded, the system should be capable of offering varying degrees of fault tolerance without major redesign.

This chapter defines a set of indices which will be used to compare various network topologies. Parameters used include vulnerability to node or link damage, cabling overhead, message delay and node complexity. A redundant loop network will then be selected for the experimental system which can tolerate double faults.
3.1 Redundant Node Design

Fault-tolerant loop networks rely on the use of redundant inter-node links. To incorporate this redundancy without the use of expensive optical couplers requires that each node has more than one transmitter and one receiver. One of the simplest redundant node structures is the 2x2 switch with simplex data links shown in figure 3.1. The number of input links adjacent to the switch is known as the indegree $\sigma_i$, and the number of output links is the outdegree $\sigma_o$. Since $\sigma = \sigma_i = 2$, the minimum degree $\sigma$ for this switch (ie. the lower of $\sigma_i$ and $\sigma_o$) equals 2. This type of switch has been used extensively in telecommunications and some computer applications [152,153,154].

Figures 3.2 and 3.3 show the node structures required for simplex and full-duplex link communications respectively. Simplex links involve the minimum hardware overhead requiring only one LED, one fibre and one photodiode. Secondary independent transmit/receive node logic would also be required if the node is to regenerate simultaneously two messages incoming on different receivers, as may arise in the network shown in figure 3.4 (see section 3.3). Full-duplex, although more flexible, requires four bidirectional optical couplers [155] per node and secondary transmit/receive logic, as well as more complicated node control. Unfortunately, these couplers are extremely expensive and are not widely available as yet. Nevertheless, full-duplex links enable the maximum fault tolerance of any topology to be achieved, and in most fault-tolerant loop networks where cabling costs are high, full-duplex links are very attractive. As commercial bidirectional optical couplers were not available until recently, the simplex implementation of the 2x2 switch, or 2x2 node, was chosen as the basis of the fault-tolerant network.

3.2 Comparative Indices for Network Desirability

Several factors affect the choice of topology for any network containing n nodes and e links. Briefly stated they are reliability, cost and message delay. The simplest reliability index is CONNECTIVITY which is representative of how tolerant the network is to link or node failure, ie. how many links or nodes must fail before
Figure 3.1. 2x2 switch

(dotted lines indicate internal switchable connections)

Figure 3.2. 2x2 node using simplex links
Figure 3.3a. Full-duplex link using two bidirectional optical couplers

Figure 3.3b. 2x2 node using full-duplex links
the controller is completely isolated from every other part of the network. The node cut-set, \( x \), represents the minimum number of nodes which must be removed from the network in order to isolate the controller, and the parameter \( \lambda \) represents the link cut-set. Optimizing the network may involve maximizing \( x \) or \( \lambda \) for a given \( n \) and \( e \), which is an example of an extremal graph problem designed to minimize network vulnerability by solving the following problems:

1) \( \max(x|n,e) \)
2) \( \max(\lambda|n,e) \)

where \( n \) and \( e \) are predefined.

Where cabling costs are of particular concern, network optimization may involve minimizing the number of links, \( e \), required to achieve a prescribed value of \( x \) or \( \lambda \) for an \( n \)-node network. These problems can be described as:

3) \( \min(e|n,x \geq k) \)
4) \( \min(e|n,\lambda \geq k) \)

where \( n \) and \( k \) are predefined.

For some process control applications, the delay imposed on a message by the regenerative network must be low so that critical command and response latency is low. Since message delay depends on the delay through each node and the number of nodes traversed, it may be important to minimize the DIAMETER of the network. Network diameter, \( d \), represents the minimum number of nodes which must be traversed so that a closed path exists between the controller and a node. The corresponding optimization may then involve minimizing the diameter of a regular network (all nodes are identical) where \( x=\sigma \). This problem can be described as:

5) \( \min(d|x=\sigma,n,\sigma) \)

where \( n \) and \( \sigma \) are predefined.

The aforementioned indices have been used in a comparative study of various fault-tolerant loop networks based on the 2x2 node. Other
factors were also taken into consideration, such as:

a) Controller complexity - the relative complexity of the controller as compared to a controller designed for a non-redundant n-node network.

b) Node complexity - the relative complexity of the node as compared to a node using simplex link, single message communications (i.e. messages on different receivers cannot be regenerated simultaneously). The latter node represents the lowest complexity, the full-duplex node represents the highest complexity.

c) Network flexibility - the ease with which nodes can be added or removed from the network.

d) Weakspots - are any parts of the network particularly vulnerable to link or node damage?

3.3 Networks That Can Tolerate Single Faults

Since simplex or full-duplex links can be used in the following network examples, the links have been left undirected. Comparative indices are given for both types of network although in general only one type need be considered to determine the optimum network.

The simplest fault-tolerant regular network is the dual redundant loop shown schematically in figure 3.4. The associated comparative indices are given in table 3.1. During normal operation and with no faults in the network, each message is regenerated by every node. Therefore if each node imposes a delay of $t$ bits on each $T$ bit message, a message may exist simultaneously in $(T/t)+1$ nodes. This network can tolerate any single node loss or any triple link loss using full-duplex links, and any single node loss or any single link loss using contra-circulating loops with simplex links. The cabling overhead is also the lowest of any regular redundant network. However, under certain fault conditions in the simplex link network (e.g. loss of links "a" and "b" in figure 3.4), some nodes may have to regenerate an outgoing message concurrently with the same message which was "looped back" from another repeater. This will occur
Figure 3.4. Dual redundant loop network

<table>
<thead>
<tr>
<th></th>
<th>Simplex</th>
<th>Full-Duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node cut-set, $x$</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Link cut-set, $\lambda$</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Cabling overhead</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Diameter, $d$</td>
<td>$n$</td>
<td>$n$</td>
</tr>
<tr>
<td>Node complexity</td>
<td>Fair</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 3.1 Comparative indices for dual redundant loop network
whenever the message length is greater than a single repeater bit delay. To avoid data corruption it is necessary to provide two isolated repeater sections, one dedicated to the outgoing message and the other to the "loop-back" message, but this requires an additional hardware overhead and more complicated processing within the node to accommodate the increased message density. Although this network is regular in structure, it is not an optimally reliable topology since it is more vulnerable to node failure than link failure when full-duplex links are used.

When designing a network for wide applicability it is beneficial to use a topology that is equally invulnerable to node loss as it is to link loss. This is particularly true when the operational environment or hazard type cannot be specified accurately. By solving problems 1 to 4 of section 3.2, it is possible to construct a family of optimally invulnerable networks. The solutions to these problems are closely inter-related and were first described by Harary [156]. To prove his findings, Harary constructed a family of optimal graphs using a simple set of rules. These are:

a) For $e=n-1$, the graph is a simple tree structure.
b) For $e\geq n$, each node $i$, $0 \leq i \leq n-1$, is connected to node $i \pm 1$, $i \pm 2, \ldots, i \pm (k/2) \text{mod } n$.
c) In addition, if $k$ is odd, each node $i$, ..., $(n-1)/2$ is also connected to $(i + n/2)$.

$k$ is the minimum degree of a node. These $H(n,k)$ graphs are known as elementary Harary graphs and exhibit the following properties:

a) $H(n,k)$ has $e = (nk/2)$, $x=\lambda=\sigma=k$. $k$ is the minimum degree of a node.
b) $H(n,k)$ is regular of degree $k$, unless $n$ and $k$ are both odd.
c) $H(n,k)$ has one node of degree $(k+1)$ and $(n-1)$ nodes of degree $k$ if $n$ and $k$ are both odd.

The simplest fault-tolerant network based on the $H(n,k)$ family is the $H(n,4)$ or single skip-braid network (SSBN) shown in figure 3.5. The comparative indices for this network are shown in table 3.2. Clearly this is an optimal network since $x=\lambda=\sigma=2$ ("s" denotes $s$ $s$ $s$ $s$)
Figure 3.5. Single Skip-Braid Network (SSBN)

<table>
<thead>
<tr>
<th></th>
<th>simplex</th>
<th>full-duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>node cut-set, x</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>link cut-set, λ</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>cabling overhead</td>
<td>200%</td>
<td>200%</td>
</tr>
<tr>
<td>diameter, d</td>
<td>n/2</td>
<td>n/2</td>
</tr>
<tr>
<td>node complexity</td>
<td>low</td>
<td>high</td>
</tr>
</tbody>
</table>

Table 3.2 Comparative indices for SSBN
simplex). It is also regular, so no weakspots exist. Using this network with simplex links, single faults can be tolerated without need for "loop-back". Therefore, although more cabling is needed compared to the previous network, node complexity is reduced.

During normal operation and with no faults in the network, the controller would adopt the simplest control strategy configuring the network so that each message traverses every node without using any of the node skipping links. Since each message traverses n nodes, the effective diameter of the network is n. When a fault occurs in a link or node, the controller must locate the fault and activate the appropriate skip link to bypass the defective element. (Fault location is discussed in Chapter 5.) Depending upon the location of the fault, the controller may also have to control more than one path through the network. For example, if link "b" fails (see figure 3.5), the controller must use the short link to access node 1 and the skip link to access node 2. Indeed, if this type of adaptive control is used during normal operation the effective diameter of the network can approach the optimum value. By using the skip links as the normal data paths and the short links as standby links (except for links "a" and "c"), the effective diameter approaches n/2. However, this network is impractical when failures occur in the short links adjacent to the controller. A fault on link "a" for example can make reconfiguration very cumbersome, especially for simplex networks where a message would have to circulate twice around the network before the controller could access node 1.

This problem can be overcome by introducing an additional transmitter/receiver pair at the controller and positioning the new controller as shown in figure 3.6. The network can now be thought of as a controller supervising two individual loops, namely nodes 1+2i and nodes 2+2i where 0<i<(n/2), which are interconnected by short links such that node i is connected to node i+1. The additional redundant short links between the controller and the first and last nodes also reduce the vulnerability of the controller to link failures (see table 3.3). It now takes three link failures to isolate the controller using simplex links. Therefore, improved reliability, simpler reconfiguration and shorter message delay have been achieved at the expense of a small increase in controller complexity.
Figure 3.6. SSBN with complex controller

<table>
<thead>
<tr>
<th></th>
<th>simplex</th>
<th>full-duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>node cut-set, x</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>link cut-set, λ</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>cabling overhead</td>
<td>200%</td>
<td>200%</td>
</tr>
<tr>
<td>diameter, d</td>
<td>n/2</td>
<td>n/2</td>
</tr>
<tr>
<td>node complexity</td>
<td>low</td>
<td>high</td>
</tr>
</tbody>
</table>

Table 3.3 Comparative indices for SSBN with complex controller
3.4 Networks That Can Tolerate Double Faults

An optimal network that can tolerate any combination of double fault is the \( H(n,6) \) topology shown in figure 3.7. Since each node has three receivers and three transmitters, \( \sigma = 3 \) as shown in table 3.4. Obviously, this network cannot be constructed using \( 2 \times 2 \) nodes and is clearly impractical in terms of node complexity and cabling overhead for most applications. However, by replacing each long skip link of figure 3.6 by a double skip link (skipping over two nodes) as shown in figure 3.8 and providing another transmitter/receiver pair at the controller, it is possible to construct a network based on \( 2 \times 2 \) nodes that can tolerate double faults. This network, known as the triple loop network or \((3:1)\) network, is basically an extension of the SSBN with the controller now supervising three interconnected loops. Compared to the \( H(n,6) \) topology, the \((3:1)\) network is equally as vulnerable to node failures (see table 3.5), since it incorporates a double skip, but less vulnerable to link failures. It now takes four failed links, compared to three for \( H(n,6) \), to isolate the controller using simplex links and the cabling overhead is reduced by some 40%. However, since this network is based on \( 2 \times 2 \) nodes, the remote nodes are obviously more vulnerable to link damage.

The multi-loop principle can be extended further to increase the fault tolerance of the network by replacing the long link of the \((3:1)\) network with a triple skip link and providing one more transmitter/receiver pair at the controller to produce a \((4:1)\) network. This topology can tolerate three node failures or four link failures without disconnecting the network. The remote nodes are still more vulnerable to link failures than the \( H(n,6) \) topology but the cabling overhead is some 20% lower.

The SSBN can also be modified to give double fault tolerance without the need to control more than two interconnected loops. This is achieved by replacing each short link of the SSBN with a double skip link and providing another two transmitter/receiver pairs at the controller as shown in figure 3.9. This network, known as the double skip-braid network (DSBN), is very similar to the SSBN as the controller still supervises only two interconnected loops. Compared to the \( H(n,6) \) topology, the DSBN is equally as vulnerable to node
Figure 3.7. Harary H(n,6) topology

<table>
<thead>
<tr>
<th></th>
<th>simplex</th>
<th>full-duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>node cut-set, x</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>link cut-set, λ</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>cabling overhead</td>
<td>500%</td>
<td>500%</td>
</tr>
<tr>
<td>diameter, d</td>
<td>n/3</td>
<td>n/3</td>
</tr>
<tr>
<td>node complexity</td>
<td>fair</td>
<td>high</td>
</tr>
</tbody>
</table>

Table 3.4 Comparative indices for Harary H(n,6) topology
Figure 3.8. (3:1) network topology

Table 3.5. Comparative indices for the (3:1) network

<table>
<thead>
<tr>
<th></th>
<th>simplex</th>
<th>full-duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>node cut-set, x</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>link cut-set, λ</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>cabling overhead</td>
<td>300%</td>
<td>300%</td>
</tr>
<tr>
<td>diameter, d</td>
<td>n/3</td>
<td>n/3</td>
</tr>
<tr>
<td>node complexity</td>
<td>low</td>
<td>high</td>
</tr>
</tbody>
</table>
Figure 3.9. Double Skip-Braid Network (DSBN)

Table 3.6. Comparative indices for DSBN

<table>
<thead>
<tr>
<th></th>
<th>simplex</th>
<th>full-duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>node cut-set, x</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>link cut-set, λ</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>cabling overhead</td>
<td>400%</td>
<td>400%</td>
</tr>
<tr>
<td>diameter, d</td>
<td>n/3</td>
<td>n/3</td>
</tr>
<tr>
<td>node complexity</td>
<td>low</td>
<td>high</td>
</tr>
</tbody>
</table>
failures (see table 3.6) but less vulnerable to link failures. Compared to the (4:1) network, it is equally as vulnerable to link failures since it requires a similar cabling overhead, but more vulnerable to node failures. As for the (3:1) network, the DSBN is equally as vulnerable to node failures but less vulnerable to link failures.

3.5 Ultra-High Reliability and Application Specific Networks

For applications demanding better than double fault protection, it would be necessary to increase the number of nodes bypassed by each skip link (eg. the (4:1) network) or increase the number of redundant transmitter/receiver pairs at the remote nodes. Both of these schemes involve a further increase in controller complexity and an even larger cabling overhead. The incremental cost would make such networks impractical for industrial use in all but the most critical applications. A better alternative would be to use an irregular network as shown in figure 3.10. By employing tertiary multi-node skipping links (long skips), it is possible to guard against catastrophic failure in selected portions of the network. Although $x = 2$ for this network, the two nodes within the hazardous area (nodes $S_4$ and $S_5$) can fail without affecting the rest of the network. Due to the ad-hoc nature of this design technique it is difficult to quantify the reliability improvements attainable, but system reliability can be optimized by careful choice of skip length and by sensible cable routing, ie. laying cables in benign environments wherever possible or choosing geographically separate paths for redundant cables to reduce the probability of multiple link damage from a single hazard source.

3.6 Description of the Experimental Network Topology

Different network topologies are suited to different applications. When the application cannot be specified accurately or when a general purpose fault-tolerant network is required, a regular optimal topology is the most desirable. Unfortunately, for networks requiring double fault tolerance, optimal topologies are expensive to
Figure 3.10. SSBN with additional long skip

<table>
<thead>
<tr>
<th></th>
<th>simplex</th>
<th>full-duplex</th>
</tr>
</thead>
<tbody>
<tr>
<td>node cut-set, x</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>link cut-set, $\lambda$</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>cabling overhead</td>
<td>200%</td>
<td>200%</td>
</tr>
<tr>
<td>diameter, $d$</td>
<td>$n/2$</td>
<td>$n/2$</td>
</tr>
<tr>
<td>node complexity</td>
<td>low</td>
<td>high</td>
</tr>
</tbody>
</table>

Table 3.7. Comparative indices for SSBN with additional long skip
implement. However, several networks with varying levels of fault tolerance can be designed around the basic 2×2 node. By increasing the complexity of the controller and the cabling overhead, the fault tolerance of the skip-braid network can be improved. Therefore, with cost and fault tolerance flexibility in mind, a skip-braid network based on 2×2 nodes has been chosen as the communications network for the low power distributed measurement system.

To satisfy the initial specification that the network should tolerate any combination of double fault, the DSBN, (3:1) or (4:1) networks can be used. Since the latter two offer a smaller network diameter than the DSBN, they are more attractive when node workload, and hence power consumption, is considered. The (3:1) network requires the lowest cabling overhead of the group and is therefore the most vulnerable to link failures. When multiple link failures that do not disconnect the network are considered, it is the DSBN which is the least vulnerable for a given cabling overhead. Consider, for example, a fault condition in which all network communications have to traverse a single node. That node is clearly critical to the operation of the system and if a fault then occurs in the short link emanating from that node, all communications must use the long link which renders the skipped nodes completely inaccessible. In the case of the (4:1) network, three nodes are inaccessible whereas only two would be lost in the DSBN. Therefore, when subjected to severe link damage, the DSBN will maintain consistently the highest node availability for a given cabling overhead. Since the DSBN satisfies the minimum fault tolerance specification and is the network least vulnerable to link failures (important when the operational environment cannot be specified), it was selected as the experimental network. The prescribed five year battery lifetime of the remote nodes is also satisfied even though the average node workload is higher than those of the (3:1) and (4:1) networks. Furthermore, because the controller still supervises only two interconnected loops, the network control procedures developed for the DSBN should be adapted easily to accommodate the SSBN in less demanding applications.

To reduce the complexity of the system and thus ease the development task, a unidirectional DSBN using simplex links has been
selected. The network consists of up to 64 addressable communications nodes supervised by a loop controller. Since the network is normally configured as two independent loops, each loop will contain a maximum of 32 nodes. To achieve the prescribed 0.25s network scan time, the controller will transmit messages alternately on each loop every 4ms. Therefore, each node must regenerate on average one message every 8ms but to reduce the node workload, only a small portion of the 8ms period will actually be used for message transmission. However, the message portion must not be too short otherwise a high bit rate will be required and the crystal clock used in the node for data encoding/decoding will consume too much power. Conversely, if the bit rate is too low the number of operations performed by the node microcomputer in the remainder of the 8ms period will be limited unduly. In view of this, a compromise clock frequency of 1MHz was chosen which resulted in a data bit rate of 100kbit/s. Therefore, each 40-bit message (see Chapter 4 for the message format) requires only 0.4ms out of the 8ms available.

The simplex DSBN can be represented as a directed graph as shown in figure 3.11a in which all nodes belong to a set of nodes, N, and all links belong to a set of links, E. Both N and E belong to a universal set U. This graph can also be described by an \((n+1)\times(n+1)\) adjacency matrix (n nodes and a controller), \(A=[a_{ij}]\), with entries:

\[
a_{ij} = \begin{cases} 
1(n_i, n_j) & \text{if } (n_i, n_j) \in U \\
\emptyset & \text{if } (n_i, n_j) \notin U 
\end{cases}
\]

where \(1(n_i, n_j)\) is the label for link \((n_i, n_j)\) and \(\emptyset\) is the zero element. The adjacency matrix for the network of figure 3.11a is shown in figure 3.11b. This representation is very useful for solving optimal path problems subject to different constraints. For instance, by labelling each link with its associated reliability, the reliability of a specific path through the network can be calculated. This is an example of an extremal path problem for which some function of the link parameter (label) is either maximized or minimized. Carré [157] demonstrated the flexibility of the adjacency matrix representation and showed that most path problems can be
Figure 3.11a. Unidirectional Double Skip-Braid Network

\[
\begin{bmatrix}
0 & l_{c1} & l_{c2} & l_{c3} & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & l_{13} & l_{14} & 0 & 0 & 0 \\
2 & 0 & 0 & 0 & 0 & l_{24} & l_{25} & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 & l_{35} & l_{36} & 0 \\
4 & 0 & 0 & 0 & 0 & 0 & 0 & l_{46} & l_{47} \\
5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & l_{57} & l_{58} \\
6 & l_{6c} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & l_{68} \\
7 & l_{7c} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
8 & l_{8c} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

Figure 3.11b. Adjacency matrix for the unidirectional DSEN
related to a set of simultaneous equations whose solutions yield the optimum path for the parameter of interest. His examples included finding longest and shortest paths, listing all possible paths and finding most reliable paths.

Having chosen the experimental network topology, the design and operation of the $2 \times 2$ node circuit can now be described. Chapter 4 outlines the basic operational requirements of the node circuit and describes the implementation of the prototype version.
This chapter describes the implementation of the 2x2 node introduced in the previous chapter. The construction and operation of the prototype circuit are described, and the advantages of circuit integration are discussed. A block diagram of the node circuit is shown in figure 4.1. The circuit consists of a battery, two optical receivers, two optical transmitters, receiver and transmitter logic (the communications interface), a single chip microcomputer and two serial interfaces. One interface handles sensor or actuator data, while the other handles data to and from a hand-held communicator that can gain access to the loop via the node. The main functions of the node circuit are to:

a) regenerate incoming messages
b) return sensor data
c) return node status information
d) return hand-held communicator enquiries
e) receive actuator data
f) receive node control data
g) receive hand-held communicator data.

The production version of the node circuit must operate continuously over the -20°C to 70°C temperature range (although -40°C to 85°C is preferred) for a five year period using a reasonably small, say about 60cc, standard commercial battery. As well as being inexpensive to manufacture, the circuit should be reliable enough to meet most industrial safety requirements.

4.1 Battery Technology

The following power supply arrangements have been proposed for the various components in the production version of the low-power measurement system:

a) The loop controller (or simply controller) will be mains-driven with a short-term battery back-up (secondary lead acid battery).
b) All components within the node circuit will be battery powered at approximately 3 volts (primary battery).
c) The sensor or actuator front-end will be battery powered (primary)
Figure 4.1. Block diagram of node
at whatever voltage is necessary for the technique used. However, a 3 volt digital interface to the node circuit must be available. 

d) The hand-held communicator will be battery powered (primary or secondary), again with a 3 volt interface to the node circuit. Therefore, there will be two batteries per node but because the type of sensor/actuator still remains unspecified, only the power requirements of the node circuit will be considered here. After some investigation into the design of the node circuit, the following current drain budgets were allocated to its constituent components:

a) 1 node receiver, always active 20 $\mu$A  
b) communications interface electronics 70 $\mu$A  
c) 1 node transmitter - mean current 50 $\mu$A  
d) node microcomputer - inactive most of the time 150 $\mu$A.

Thus, the node circuit was estimated to consume about 300$\mu$A mean current when used in a 64-node Double Skip-Braid Network (DSBN), which would require a battery of about 13 ampere hours (Ah) capacity for a five year replacement lifetime. Although the node workload may increase during certain fault conditions, it is assumed that the mean time to repair (MTTR) will be sufficiently small compared to the expected five year replacement life that the additional current drain can be neglected.

A survey of battery types showed that only those based on lithium come close to satisfying the energy density, shelf life and temperature range requirements of the node circuit. Lithium cells offer the highest energy density currently available from any battery technology and, after initial fears about their safety, are gaining acceptance as primary batteries in many military, aerospace, industrial and commercial applications [158,159]. The most popular lithium chemistries in military and aerospace applications are lithium thionyl chloride ($\text{LiSOCl}_2$), lithium sulphur dioxide ($\text{LiSO}_2$) and lithium iron disulphide ($\text{LiFeS}_2$). Lithium manganese dioxide ($\text{LiMnO}_2$) and lithium copper oxide ($\text{LiCuO}$) are also popular in more commercial applications. Of these chemistries, $\text{LiSOCl}_2$ offers the highest energy density and is the most widely available, which should ultimately lead to a reduction in price.

Aimed mainly at meeting military and industrial requirements, these cells exhibit excellent performance characteristics (flat
discharge voltage, no delay, high capacity, good temperature range etc.) and have been used in a variety of applications ranging from low current pacemakers to US Naval underwater instrumentation. Reserve LiSOCl₂ power systems are also being planned for the US Airforce MX missile silos. Although LiSOCl₂ cells do not have high internal pressures, as do LiSO₂ cells, they are not free from risk. If a cell is ruptured accidentally, highly toxic SOC₁₂ may escape, but for most applications the risk can be minimized by careful cell design and placement. Where "absolute" safety is vital, lithium iodide (LiI₂), LiMnO₂ or LiCuO₂ cells may be preferable. Lithium iodide cells are almost completely solid-state and have been used in implanted pacemakers since 1972 with a commendable reliability history. They are, however, unavailable in large capacity cells. High capacity LiMnO₂ and LiQuO₂ cells are available but both require a greater cell volume than LiSOCl₂ for the equivalent capacity. Unfortunately, LiMnO₂ cells are extremely susceptible to reverse current damage and are therefore not ideal for very high temperature (over 70°C) applications where protection diode leakage may exceed the reverse current limit for the cell. Lithium copper oxide cells are well suited to wide temperature ranges but because their terminal voltage is about 1.5V, a battery would be needed to achieve the required 3V supply level. This again requires a larger volume than a LiSOCl₂ cell of equivalent capacity. High capacity LiSO₂ cells have been used for many years by the US Army and are still quite popular, but their safety at high temperature has been of some concern because of the high internal cell pressures.

In summary, the most desirable lithium cells for use in this application are LiSOCl₂, LiSO₂, LiMnO₂ and LiQuO₂. Table 4.1 lists some comparative data for the four chemistries. The safety of all lithium cells has been of concern to users for many years, but greatly improved manufacturing techniques are now producing a range of safe and reliable cells. In fact, most manufacturers will design batteries specifically to meet the needs of the individual user and many claim that any reasonable specification can be met at additional cost. Technical brochures are available from Tadiran [160] and Altus [161], for example, to support these claims. Assuming that the required safety and reliability levels can be achieved, the choice of
<table>
<thead>
<tr>
<th></th>
<th>LiSOCl&lt;sub&gt;2&lt;/sub&gt;</th>
<th>LiMnO&lt;sub&gt;2&lt;/sub&gt;</th>
<th>LiCuO</th>
<th>LiSO&lt;sub&gt;2&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal voltage (V)</td>
<td>3.5</td>
<td>2.9</td>
<td>1.5</td>
<td>2.8</td>
</tr>
<tr>
<td>Capacity (Ahr)</td>
<td>0.2 - 28</td>
<td>0.03 - 10</td>
<td>0.5 - 3.6</td>
<td>0.3 - 35</td>
</tr>
<tr>
<td>Temp. range (°C)</td>
<td>-60 - 150</td>
<td>-40 - 80</td>
<td>-55 - 150</td>
<td>-54 - 71</td>
</tr>
<tr>
<td>Shelf-life (years)</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Discharge curve</td>
<td>FLAT</td>
<td>SLOPE</td>
<td>FLAT</td>
<td>FLAT</td>
</tr>
<tr>
<td>Current range</td>
<td>low to high</td>
<td>low to med</td>
<td>low to med</td>
<td>med to high</td>
</tr>
<tr>
<td>Power/Vol (Wh/Dm³)</td>
<td>800</td>
<td>550</td>
<td>730</td>
<td>480</td>
</tr>
<tr>
<td>Power/Wgt (Wh/Kg)</td>
<td>420</td>
<td>260</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>Cost/Wh (£)</td>
<td>0.53</td>
<td>0.53</td>
<td>0.36*</td>
<td>0.53</td>
</tr>
</tbody>
</table>

* - cost of battery fixture not included, adjusted cost in brackets.

**Table 4.1. Comparative data for lithium cells**

**Figure 4.2. Pulse position coding (PPC)**
cell can be made purely on performance and cost. Clearly the best choice at present is LiSOCl^2 which offers the highest energy density, meets the prescribed temperature range, is the most widely available (therefore costs should fall) and has excellent discharge characteristics. A D-type LiSOCl^2 cell, from Altus (14Ah) or Electrochem Industries (14Ah) for example, is therefore recommended as the power source for the node circuit. A 14Ah cell is capable of providing a constant current drain of 320μA over a five year period, which exceeds the budgeted figure of 300μA. For applications demanding "absolute" safety, LiCuO is the next best choice for all temperature ranges. However, since a battery is necessary to provide a 3V supply, the cost figure for LiCuO in table 4.1 is misleading as it does not include the additional cost of combining these cells in a battery. Should a battery be undesirable, a LiMnO^2 cell could be used in low to medium temperature applications.

An important point to note is that because most cells are optimized for operation at or just above room temperature, a loss of capacity (up to about 20%) is experienced if operation is at either extreme of the -40°C to 85°C temperature range. Therefore, for continual operation at either extreme, the lifetime of the system must be derated accordingly or, alternatively, a larger capacity cell should be used. Cell terminal voltage is also affected by temperature but for the preferred chemistries above, the effect is negligible.

4.2 Data Coding Scheme

As discussed in the last chapter, the data bit rate for network communications was set at 100kbit/s by the 1MHz crystal clock in each remote node, and to reduce the mean power consumption each message occupies only 0.4ms of the 8ms available for each message. Since optical fibre communications use serial data formats, the mean power consumption can be reduced still further if the data coding scheme takes the form of Pulse Position Coding (PPC). By using a low mark space ratio, ideally a 1μs pulse in a 10μs bit time since each node uses a 1MHz clock, a small mean Light Emitting Diode (LED) current can be maintained. Moreover, it is commonly accepted that pulsing an LED with a high current for short duration is more efficient than a
low current for long duration. During each bit time only one pulse is generated, and the binary information is represented by the position of that pulse within the bit. A binary '1' is represented by a pulse in the first half of the bit, and a '0' by a pulse in the second half (see figure 4.2). A ten phase clock in the transmitter logic divides each bit time into phases 0 to 9, each phase lasting 1μs. A pulse is transmitted during phase 2 to represent a binary '1', a pulse during phase 7 represents a '0'. The minimum pulse separation is therefore 5μs, which occurs when a binary '0' is followed by a '1'.

The receiver logic uses the same ten phase clock for data decoding but requires synchronization before the incoming pulses can be decoded correctly. Synchronization is achieved by transmitting a start bit during phase 0 of the first two bit periods. The receiver logic synchronizes to the leading edge of the first start bit and then monitors the remaining portion of the start bit sequence to verify that a valid start bit was received. Message reception will commence only if a valid 2-bit start sequence (or key), consisting of a pulse during phase 0 and no pulse during phases 1 through 9 for both of the first two bits, is received. If the proper key appears, the receiver logic decodes the subsequent 38 bits in the message. Therefore, each message consists of 40 pulses representing 2 start bits and 38 data bits. Every message has two fields: a 14-bit header field followed by a 24-bit field. Each field has its own parity check bits (Hamming and overall parity) and so single error correction and double error detection is possible within each field. Using this coding, two single bit errors per message can be detected and corrected provided they occur in different fields. The 14-bit header contains a 6-bit node address which is sufficient to address up to 64 nodes, a 3-bit controller-to-node command which accommodates eight different commands, and 5 check bits. The 24-bit field contains a general purpose 16-bit data field, a 2-bit node-to-controller request and 6 check bits. The detailed message format is given in table 4.2.

4.3 Optical Data Link Design

In accordance with the prescribed specifications, a data link of 500m maximum length must be accommodated with a mean current budget
<table>
<thead>
<tr>
<th>BIT ADDRESS</th>
<th>SIGNIFICANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message</td>
<td>Field</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
</tr>
<tr>
<td>21</td>
<td>7</td>
</tr>
<tr>
<td>22</td>
<td>8</td>
</tr>
<tr>
<td>23</td>
<td>9</td>
</tr>
<tr>
<td>24</td>
<td>10</td>
</tr>
<tr>
<td>25</td>
<td>11</td>
</tr>
<tr>
<td>26</td>
<td>12</td>
</tr>
<tr>
<td>27</td>
<td>13</td>
</tr>
<tr>
<td>28</td>
<td>14</td>
</tr>
<tr>
<td>29</td>
<td>15</td>
</tr>
<tr>
<td>30</td>
<td>16</td>
</tr>
<tr>
<td>31</td>
<td>17</td>
</tr>
<tr>
<td>32</td>
<td>18</td>
</tr>
<tr>
<td>33</td>
<td>19</td>
</tr>
<tr>
<td>34</td>
<td>20</td>
</tr>
<tr>
<td>35</td>
<td>21</td>
</tr>
<tr>
<td>36</td>
<td>22</td>
</tr>
<tr>
<td>37</td>
<td>23</td>
</tr>
<tr>
<td>38</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 4.2. Detailed format of 38-bit message
of approximately 70µA. To minimize signal attenuation over long links (typically greater than 30m), a glass fibre with an attenuation of 6dB/km or less will be used. For short links, however, a polymer fibre may be adequate. In both cases, a large core diameter fibre is beneficial since LED-to-fibre coupling is improved for large core fibres. Unlike most of the optical links reported in the literature, speed was not of prime concern in this application. Instead, the aim was to achieve a low-power data link at low cost.

To construct an experimental link, the Radio Spares (RS) fibre-optic "sweetspot" device range was chosen because of its low cost, emitter-photodiode matching and compatibility with a wide range of fibres. The optical transmitter consisted of a simple driver circuit and an RS 633-701 850nm LED. Since each message consists of forty 1µs pulses every 8ms for a 64-node DSBN, a mean transmitter current of about 50µA was achieved using 10mA pulses. An RS 303-292 "sweetspot" photodiode formed the basis of the optical receiver. After experimenting with several low power operational amplifier receivers, better bandwidth for a given drain current was achieved using the discrete component amplifier shown in figure 4.3. This simple transimpedance preamplifier offers good output stability and provides a ΔV of approximately ΔI R. The output from the amplifier is AC coupled to the Schmitt input of a standard CMOS gate which is just biased ON. The arrival of an optical pulse then turns the gate OFF. The receiver circuit consumes a mean current of approximately 8.7µA at 5V for an input bit rate of 100kbit/s and is therefore well within the budgeted figure. Although the leading edge response of this amplifier is fast because the transistor is switching ON, the trailing edge response is very sluggish due to the large collector load (330kΩ). This problem is overcome by using baseline restoration where the communications interface electronics, having received a pulse from the receiver, actively discharges, 2µs later, the stored charge from the collector of the transistor. This is accomplished by momentarily (for 1µs) connecting a 62kΩ resistance across R C to restore the DC level at this critical point within 3µs of the arrival of the optical pulse. In this way, the receiver can cope with the 5µs minimum pulse separation at an acceptable current drain.

An optical data link was constructed using the above transmitter
Figure 4.3. Optical receiver circuit
and receiver, along with a short length (so that the fibre attenuation could be neglected) of 1mm core polymer fibre. Using a pulse width, $t_p$, of 1μs, the LED drive current, $I_{LED}$, was adjusted just until the receiver trigger threshold was exceeded (i.e. the receiver triggered). The equivalent LED drive charge, $Q_{LED} = I_{LED} \cdot t_p$, was then noted. This 'zero' length link exhibited a drive charge threshold of just over 2nC which, for example, corresponds to a 2mA LED pulse of 1μs duration. Since this figure includes all fibre coupling losses, an estimate of the maximum link length can be made given the fibre attenuation and the transmitter current budget. Therefore, say a conservative drive charge threshold of 3nC (3mA pulse for 1μs) is assumed, a fibre attenuation of about 5dB per link can be tolerated in a full 64-node DSBN with a mean transmitter current of 50μA per node (i.e. 10mA per 1μs pulse). This result suggests that a 500m link is feasible using low-loss glass fibre.

It must be stressed that the link described above was constructed purely for experimental purposes and was not intended as a production prototype. A production version would use optical components specifically suited to the desired application and would minimize link losses by careful LED-fibre-photodiode matching. A low power comparator would also be preferred, for reasons of stability, to the Schmitt trigger used above. Nonetheless, the preamplifier does exhibit sufficient DC stability that it, or a derivative of it, could be used in a production version.

4.4 Sensor/Actuator Interface

The basic function of the node circuit is to convey information between the loop controller and remote sensors and actuators. The sensor/actuator market is well established and so it is important that the node circuit be compatible with existing devices as well as the new generation of ultra-low power devices which are currently of great interest to the industrial community [162]. Many process control sensors monitor the differential pressure across a plate, via a capacitance variation, to measure variables such as flow rate, pressure or level, whilst actuators involve pneumatic or hydraulic amplification and valve control. Interfacing the node circuit to
existing sensors requires a battery-powered external interface unit that converts the analogue measurement into a CMOS level digital signal for transmission to the node circuit. Further data processing can then be performed by the node circuit, if necessary, before the information is sent back to the controller. The sensor could be connected directly to the node circuit if the latter incorporated some analogue-to-digital conversion (ADC) and signal conditioning circuitry, but this would increase its overall power consumption and hence reduce the battery replacement interval. Incorporating this type of circuitry in the node circuit would also limit the applicability of the system to a set of prescribed sensors or actuators. System flexibility is therefore maximized by specifying that an external interface is required. A similar interface unit is required for actuator control.

The new generation of sensors mentioned above relates to those devices that include all sensing, signal conditioning and signal conversion (ADC etc.) on the same substrate. Interfacing to such devices would be relatively straightforward as the external interface will be integrated with the sensor. Experimental work has also been reported on low power actuation by other authors [163,164]. For convenience and economy of interface lines, the node circuit provides a 16-bit serial sensor/actuator interface. A parallel interface could also be accommodated at the expense of interface lines. In the prototype system, a set of sixteen switches simulates a sensor, a row of sixteen LEDs represents an actuator and a visual display unit (VDU) mimics a hand-held communicator.

4.5 Node Microcomputer

Every function of the node is supervised by a single-chip microcomputer. To remain within the prescribed power budget, it is proposed that a device, such as an NEC 7507, is used in a production system. However, for development purposes a microcomputer board, consisting of an Hitachi 6303 CMOS microprocessor which contains RAM, a CMOS 2716 EPROM and a CMOS Versatile Interface Adapter (VIA) for Input/Output (I/O) expansion, was used. The role of the microcomputer board is to:
a) supervise the operation of the communications interface  
   (eg. enable receivers and transmitters, accept status information)
b) send data to and receive data from the communications interface
c) provide 2-way communications with a hand-held communicator
d) monitor remote sensors and control remote actuators
e) act on instructions received from the loop controller  
   (eg. return sensor data, send data to actuator, node housekeeping)
f) prepare 38-bit messages, including Hamming and parity bits, for  
   transmission by the communications interface
g) help with system fault finding and self-test.

Since a node processes only one message every 8ms for a fully  
operational 64-node DSBN the microcomputer is inactive for most of  
the time. Furthermore, because only a very small proportion of these  
messages will actually be addressed to the node, the microcomputer  
need act only on messages received approximately every 250ms. By  
incorporating a communications interface (see section 4.6) that  
selectively regenerates incoming messages and interrupts the  
microcomputer only when necessary, the current consumed by the  
microcomputer can be minimized. Moreover, the communications bit rate  
is now not limited by the speed of the microcomputer.

Once a message passes through the node, the communications  
interface sets an interrupt flag. This re-activates the microcomputer  
which then checks the status flags provided by the communications  
interface to see if a message addressed to that node was received.  
If the correct address was not received, the microcomputer simply  
resets the communications interface and returns to an inactive state.  
If the correct address was received, the message, and any relevant  
status information (eg. the position of a single bit error in the  
24-bit field), is read in by the microcomputer. Depending upon the  
type of message sent by the controller, the microcomputer may have to  
return sensor data, send data to an actuator, return node status  
information, perform node reconfiguration or return hand-held  
communicator data. On average, the microcomputer will require about  
0.1ms to interrogate the communications interface and 2ms to process  
a message request. Therefore, for a 64-node DSBN its workload factor  
(i.e. active period per network polling period) is approximately 0.02  
which, assuming it consumes 2mA under full load, results in an
average current consumption of 40μA. If a worst case quiescent current consumption of 10μA is assumed, the estimated average current consumption for the node microcomputer is 50μA, which is well within the budgeted figure of 150μA.

The microcomputer also has to perform some housekeeping functions such as monitoring transmission errors from a previous node, testing the communications interface and timing out when no optical pulses have been received during a prescribed time interval (fault condition). All of these functions will be discussed in more detail in subsequent sections and chapters. At present, it is sufficient to say that a small amount of intelligence exists in each node and that forthcoming discussion pertaining to the various functions of the node, assumes this fact.

4.6 Communications Interface Circuit

All data transfers between the node and other adjacent nodes are made via the communications interface circuit which essentially acts as a buffer between the node microcomputer and the rest of the network. A block diagram of the interface is shown in figure 4.4. Basically the interface must store any message addressed to the node whilst regenerating a concatenated message consisting of the incoming 14-bit header plus 24 bits of data supplied by the microcomputer. Any message not addressed to the node is regenerated in full. A list of the main tasks performed by the interface is given below:

a) synchronize on to the first input pulse received
b) produce "clamp" pulses for optical receiver baseline restoration
   2μs after every pulse is received
c) commence data reception (by decoding PPC) only when a valid
   2-bit start sequence (known as a "key") has been received
d) once a key has been received, begin message regeneration
   (ie. re-encode message into PPC with a 2-bit delay)
e) correct single bit errors/flag double bit errors in the
   14-bit header, then compare message address with node address
f) if address is incorrect, regenerate incoming message in full
g) if address is correct, replace 24-bit field of incoming message
   with data from this node (eg. sensor, status data)
Figure 4.4. Block diagram of communications interface circuit

(S = data selector)
h) locate single bit errors/flag double bit errors in 24-bit field
i) when transmission is complete, flag any message errors and then interrupt microcomputer
j) self-transmit full 38-bit message supplied by the microcomputer
k) perform self-test under the supervision of the microcomputer
l) receive all messages with a valid key irrespective of address.

A communications interface was designed and built using 47 standard CMOS SSI (Small Scale Integration) and MSI (Medium Scale Integration) devices on a bread-board measuring 23cm square. With the aid of gate level descriptions of these devices [165], it was estimated that the design required 1193 gates (one gate is equivalent to a two input NAND gate). This figure includes a large number of redundant gates since not every gate of every device is utilized by the design. The bread-boarded prototype functioned as expected, consuming approximately 20mA from a 5V supply under normal operational load. This implementation was obviously too cumbersome, in terms of size, cost and power consumption, for incorporation in a production node circuit. However, throughout the design phase it has been assumed that the interface circuit would be integrated on to one chip for a production version. This would reduce the package count and hence reduce size, cost, power consumption and failure rate.

4.6.1 Communications Interface Chip

In March of 1986, University College London (UCL) was approached concerning its offer of a low cost gate array service which took advantage of the newly formed multi-project gate array facility furnished by Marconi Electronic Devices Ltd. (MEDL) at Lincoln. To avoid the expense of MEDL redesigning the original interface design into an acceptable format (eg. a HILO 2/3, MENTOR, DAISY, VALID net-list), the circuit was broken down into its component parts and redesigned using the HILO 2 digital simulator [166] provided by the Department of Electrical Engineering, UCL, and gate array timing information supplied by MEDL. HILO is a gate level simulator that provides both a structural and functional level simulation, as well as automatic test generation facilities. Circuits can be simulated with picosecond accuracy and delay calculations, including capacitive
loading effects, can be performed automatically. HILO also supports an extensive fault analysis simulation which was used to evaluate test patterns for production and field testing.

Transferring the communications interface circuit into gate array form involved a complete redesign of the bread-boarded original. Since negative logic consumed less silicon area on the gate array than positive logic, for example a 2-input NAND gate required half the area of a 2-input AND gate, as much of the original circuit as was practical was translated into negative logic. All the larger integrated circuits (decoders, counters, etc.) were broken down into the basic forms required and redesigned using only gates and flip-flops. In fact, the largest building block used in the new design was a resettable D-type flip-flop which required seven gates. The completed design required only 846 gates which represented a 29% reduction in the gate count compared to the bread-boarded original.

4.6.2 Communications Interface Blocks

To ease the functional testing of the chip simulation, the design was split into the following eleven functional blocks:

a) Receiver

This block takes a pulse position coded (PPC) input signal, supplied either by the optical receivers or the self-test circuitry, and translates it into binary format ready for use by the rest of the interface circuit. The receiver synchronizes to the first input pulse received, setting a flag stating which optical receiver produced the pulse, and then checks for a valid key. Subsequently, if a valid key is not received, the receiver shuts down and interrupts the microcomputer. The receiver block requires 122 gates.

b) Ten-Phase Johnson Counter

Upon the arrival of the first input pulse from the optical receiver or self-test circuitry, the Johnson counter divides the external 1MHz clock into ten 1μs phases which can be used for PPC decoding and event separation in other blocks within the interface. This block requires 60 gates.
c) System Counter

This block produces timing information, by counting the number of bits regenerated, for the rest of the interface, thus ensuring that any data dependent checks made by the interface occur at the proper time (e.g., address comparison can proceed only when the correct number of bits have been received). The count produced is also used by the error detection circuitry as discussed later in this section. The system counter requires 70 gates.

d) Optic Pulse Counter

Primarily used for error detection purposes, this block counts the number of input pulses received, setting an error flag for any number not equal to forty (i.e., the number of pulses in each message). This block requires 51 gates.

e) Syndrome Generator

The syndrome generator extracts encoded parity information from each message received and produces a syndrome corresponding to the position of any single bit error in either the 14-bit or 24-bit field. Double bit errors are detected but no information is produced relating to their location. Each syndrome is generated by decoding Hamming bits in a very efficient manner. Normally, syndrome generation is performed instantaneously once the appropriate bit field is available. This is achieved by recomputing, in a parallel fashion, the expected check bits from the received data field and 'EXOR'ing them with the check bits received. For speed, the parallel check bit computation can be done in hardware, as shown in figure 4.5 for example, or in firmware. The first technique requires an array of 'EXOR' gates dedicated to the 14-bit field and the 24-bit field to produce the expected check bits and would, therefore, involve a high gate count (e.g., about 117 gates just to generate the two syndromes for the 38-bit message). The second technique involves an addressable Read Only Memory (ROM) in which the contents of each address is the correct check bit sequence for that address (i.e., as if the physical address were the data field). Any data field received simply addresses the ROM which then produces the correct check bits for that data field. The check bits can then be 'EXOR'ed with the received check bits to generate the syndrome. Since 18 address lines are needed for a field containing 18 data bits, as in the 24-bit field of
<table>
<thead>
<tr>
<th>Bit number</th>
<th>Hamming bits</th>
<th>Parity</th>
<th>Data bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>14 13 12 11 10 9 8 7 6 5 4 3 2 1</td>
<td>4 5</td>
<td>6 7 2 3</td>
<td>5 7 1 3</td>
</tr>
</tbody>
</table>

**Figure 4.5. Parallel syndrome generator for 14-bit field**

**Figure 4.6. Serial syndrome generator for 24-bit field**
each message, a 256k:5 bit ROM would be necessary. Therefore, this implementation is not feasible for the desired application.

Instead of recomputing the check bits in parallel, the syndrome generator used in the chip accumulates the syndrome serially as the data field is being received. By ensuring that check bit 1 occupies position 1 in each of the two message fields, check bit 2 occupies position 2, check bit 3 occupies position 4, and so on (as described in Chapter 2, section 2.3.3), the circuit shown in figure 4.6 can be used to generate the syndrome. This circuit is a direct implementation of the technique used in equation 2.1 where each bit of the received bit field is multiplied by the appropriate column of the parity check matrix H, and the result accumulated. Boolean multiplication simply requires the AND function, accumulating the result requires only one toggle flip-flop per check bit and each row of H is sequentially generated by the already available system counter. Furthermore, provided the system counter is preset to a count of one before the arrival of each of the two bit fields, the same circuit can be used to generate the syndrome for both the 14-bit and 24-bit fields. As a result, this circuit provides a very economical means of syndrome generation for serially transmitted data, requiring only 71 gates. As implemented in the communications interface chip, the syndrome generator block also performs an overall parity check, decodes the syndrome to point to the location of a single bit error and sets flags corresponding to correctable and incorrectable errors.

f) Address Comparator

The address comparator corrects any single bit error in the received address before comparing that address with the node address (set up on six switches). This block requires 42 gates and produces an "address OK" signal if the addresses agree.

g) Flags

By collecting status information from the rest of the interface, this block produces a set of flags for use by the node microcomputer. The flags generated include "single bit error in 14-bit field", "single bit error in 24-bit field", "fatal error has occurred" and "received message is now available". This is the smallest block, requiring only 27 gates.
Transmitter

Acting as a PPC encoder, the transmitter can accept binary data for regeneration, node data (e.g. sensor, status) or self-test data. It can also generate fresh start bits as required and redirect self-test data back to the input of the receiver block. During self-test, PPC data is rerouted from the input of the receiver block (which now receives self-test data) directly to the transmitter and regenerated with only a few gate delays. Self-test is discussed in section 4.6.3. The transmitter block requires 46 gates.

Control

All storage of incoming data is performed under the supervision of this control circuit. It enables the shifting pulses for the storage register and provides all interface reset signals when message transmission has terminated, as well as interrupting the microcomputer. This block requires only 29 gates.

41-bit Shift Register

Forming the main data storage area on the chip, this block consists of a 15-bit shift register in series with a 26-bit shift register. Normally, the two are disjoint and the microcomputer is free to load the 26-bit shift register with node data (e.g. sensor, status). Thus, before a node is addressed, the 26-bit shift register will be loaded with 24 bits of node data for onward transmission. The 15-bit shift register, on the other hand, automatically accepts the 14-bit header of each incoming message before address comparison is performed. If the addresses disagree, the two shift registers remain disjoint and the incoming message is regenerated in full. However, if the addresses agree, the two shift registers are connected together and the whole 38 bits of incoming data is stored. Moreover, as the 24-bit field is being stored, it pushes out the 24 bits of node data which is then concatenated with the already regenerated 14-bit header. In short, if the message address and node address agree, the message transmitted by the communications interface consists of the received 14-bit header plus 24 bits of data supplied by the node. Since the incoming message is stored in its entirety, data can be sent to and received from the node using only one poll. The full 41-bit shift register can also be controlled by the microcomputer which may generate complete messages for onward transmission (as may
be required under certain fault conditions, see Chapter 5) or perform self-test of the interface chip. Race problems associated with clocking serially connected shift registers were avoided by using multiphase clocking signals. Although this block is the largest block of the interface, requiring some 257 gates, its structure is very regular and is, therefore, relatively simple to test.

k) 8-bit Universal Shift Register

To reduce the number of interface lines between the microcomputer and the communications interface circuit, all status information relating to the validity of the received message is stored in an 8-bit universal shift register. The microcomputer then reads the following status information serially: "fatal error in message" flag, "incorrect number of bits received" flag, "single bit error in 24-bit field" flag and the position of that error (5 bits). Since this shift register abuts the 41-bit shift register, the microcomputer has the option of reading these 8 bits followed by the 38-bit message or reading directly the 38 bits. Again, clocking races were avoided by design. This block requires 71 gates.

Each of the above blocks was designed using HILO descriptions of gates and flip-flops compiled by MEDL. The blocks were then added to the existing block library, which then formed the basis of the complete communications interface. The task of simulating the interface involved testing each of the eleven blocks and compiling a circuit description of the interface bringing together these blocks with the appropriate interconnections. Once completed, the interface simply became another member of the block library.

4.6.3 Self-test of the Communications Interface

Over the last decade, the complexity of microelectronic devices has increased dramatically. As a result, device testing has become more difficult and time consuming, and for many devices, such as large memory chips or microprocessors, the cost of testing can account for a large portion of the retail price. To minimize the manufacturing costs of the node, the communications interface chip (which is the only non-standard component in this application) was
designed to be easily testable with the minimum of external Automatic Test Equipment (ATE). Under the constraint that the gate count needed to improve the chip testability should not be so high (say 10%) as to adversely affect the reliability of the chip, a self-test strategy was developed. Originally, a self-test and self-repair strategy was investigated but this proved too complicated and elaborate, in terms of gate overhead and cost (ie. a larger chip was needed), for this application. Instead, precedence shifted to simplifying the testing procedure as a means of reducing manufacturing costs. Since the interface processes bit streams serially, testing does not require the stimulation of many input pins. However, the fact that standard input/output for the chip is in PPC format does imply that accurate timing information is needed in the test equipment. This problem could be alleviated by forcing the test vector past the input and output stages, but a lower fault coverage would result. Another problem associated with testing is that although data enters the chip serially, several parallel output flags are generated which must all be checked to obtain good fault coverage.

Both exhaustive and functional built-in self-test (BIST) strategies, in which all testing circuitry is "on-chip", were investigated, but it soon became apparent that a BIST approach could not be achieved with the allocated gate overhead. A new approach was developed, based on a scan-in/scan-out technique, which utilized most of the original interface circuit. The basic repeater action of the interface is shown schematically in figure 4.7. Message bits are received, stored and regenerated for onward transmission. By adding some control circuitry to the chip, this operation can be modified to perform self-test as shown in figure 4.8. In self-test mode, both optical receiver and transmitter are decoupled from the rest of the interface circuit, with incoming signals being rerouted directly from the former to the latter. A path is then created between the transmitter and receiver blocks within the interface so that even the key checker, PPC decoder and PPC encoder are tested. A 38-bit test vector, chosen to exercise the most critical components of the interface, is then scanned into the shift register block by the external ATE. Once the test vector is loaded, the interface is triggered and the test vector is unloaded from the shift register. It
Figure 4.7. Repeater action of the communications interface

Figure 4.8. Self-test configuration of the communications interface
is then processed as normal by the interface and the circuit response is captured back in the same shift register. On completion of the test, the interface flags the ATE and the output response can be scanned out. This test is not a true scan-in/scan-out test since all sequential circuitry is not broken down into purely combinatorial logic, but it does eliminate the need for accurate timing information in the ATE by making the chip itself perform the PPC encoding and decoding. This efficient utilization of existing circuitry is reflected in the very low gate overhead which is under 3%.

This test strategy is well suited to a wide variety of ATE, with the minimum requirement being that of a clock and power supply for the chip, a test-data input, a shift clock, a trigger input and a logic probe to read the output response. The HILO fault simulator was used to estimate the fault coverage attainable using the above self-test. First of all, the simulator stores a "good" circuit response for a given test vector. The circuit is then resimulated with a single stuck-at-fault present at an input or output of one of its gates, and the output response compared to the "good" response. If the two responses are identical, the fault is logged as "undetectable" using that test vector. This process is repeated until both stuck-at-1 and stuck-at-0 faults have been simulated for each input and output of every gate in the circuit. The resulting list of undetectable faults can then be used to estimate the fault coverage. The following 38-bit test vectors were chosen to exercise the circuit:

a) message with correct address and no errors
b) message with correct address and single bit errors in both fields
c) all '1's and all '0's messages
d) messages generated randomly.

The results obtained (see figure 4.9) show that an 81% fault coverage can be obtained with only one test vector. The coverage rises to 86.7% for three test vectors and flattens out at about 87.5%, with no improvement beyond six test vectors. This ceiling exists because the test is mainly a functional test (i.e. faults in parts of the circuit which are not required for normal chip operation, such as unused counter states for example, are not detected), and is not exhaustive. It also neglects some stuck-at-faults on the interface inputs as well
Figure 4.9. Self-test fault coverage vs Number of test vectors
as a small section on the transmitter front-end logic. Nonetheless, with suitable ATE, a single test can be performed in approximately 0.5mS and because only three tests are required for a fault coverage in excess of 85%, this test could easily be used as a fast Go/No-Go test at manufacture prior to a more detailed and time consuming functional test.

4.6.4 Chip Implementation

Once the chip design had been fully simulated and tested, the HILO net-list (see Appendix 1) was submitted to MEDL for fabrication. The device chosen was the MA2024 double level metal CMOS gate array which had 2436 gates and was packaged in a 68-pin leadless ceramic chip carrier. By choosing such a large array, it was possible to incorporate two identical circuits on a single chip, thus increasing the probability of obtaining a working circuit from the ten prototypes. Of the ten chips received, six complete chips (i.e., 12 circuits) functioned as expected. Three chips functioned on one half only and one chip was totally faulty. Therefore, in terms of operational chips, a yield of 60% was obtained. However, in terms of usable interface circuits, a yield of 75% was obtained. All chips were tested using a Tektronix "Digital Analysis System" (DAS 9100 series).

Tests were performed over frequencies ranging from 1kHz to 20MHz (limited by the DAS) and all operational circuits functioned correctly. The average power consumption was then calculated by measuring the quiescent (chip inactive but with 1MHz input clock) current $I_q$, and operational (running continuously with no inter-message gap) current, $I_s$, of the chip at various voltages. It is important to note that the former figure is the more critical since the interface is inactive 95% of the time during normal operation. A large variation in $I_q$ was observed over the batch with the best chips consuming a few $\mu$A at 3.5V and the worst chip consuming over 300$\mu$A at 3.5V. However, since a quiescent current of less than 10$\mu$A at 3.5V should be expected for a CMOS technology, it can be assumed that the figures obtained for the best chips should be repeatable with an efficient fabrication process. Measurements obtained from the best three chips were averaged and plotted. The
supply current for continuous operation versus frequency is shown in figure 4.10 and the quiescent and operational currents versus supply voltage are shown in figure 4.11. Using the latter graph, it was estimated that the chip would consume approximately 15\(\mu\)A from a 3.5V LiSOCl\(_2\) cell with a normal workload in a 64-node DSN. This leaves 55\(\mu\)A from the budgeted 70\(\mu\)A to power the 1MHz crystal oscillator supplying the chip, which is ample.

As well as being used in remote nodes, it is proposed that two chips will be used in the loop controller for message reception and transmission. In this application, two interface circuits will be dedicated to reception, storing all messages irrespective of address, and two will be dedicated to transmission. Loop controller hardware will be discussed in Chapter 5.

### 4.7 Hand-Held Communicator

The hand-held communicator (HHC) is simply a hand-held terminal which enables an operator in the field to gain access to the loop via the node. Using the HHC, an operator can communicate with the controller or any remote node from any node in the loop. To avoid possible disruption to normal communications, all message transactions concerning the HHC are supervised by the loop controller which can assign priority levels to each transaction. The HHC is intended for field uses such as node status interrogation, sensor zero setting and span calibration, actuator setting, loop repair and controller interaction. Basically, the device consists of a microcomputer, a liquid crystal display, a keypad, a battery and an interface for communicating with the node. Whenever the HHC can be plugged directly into the node, a current-limited serial interface at CMOS levels will suffice. However, where access to the node is restricted or where electrical connectors are dangerous, an alternative communications technique may be required.

Infra-red remote communications is one possibility but link quality cannot be guaranteed in all conditions (eg. dirt build-up on the optical receiver). A radio link could also be used but the receiver in the node, because it must be on continuously, might pose an excessive drain on the node battery. Moreover, radio links are
Figure 4.10. Supply current vs Operating frequency at 3V supply (average results for prototype gate array chip)

Figure 4.11. Supply and Quiescent currents vs Supply voltage at 1MHz (average results for prototype gate array chip)
often poor quality at very low transmitter power levels, are susceptible to bad reception depending upon the layout of the plant and are not feasible in applications demanding complete immunity to outside interference.

One communications technique which can be used at low power, requires no physical connection and is inherently safe, is based on inductive pick-up whereby a communications link is formed simply by bringing two coils into close proximity with each other as shown in figure 4.12a. A simple half-duplex link can be constructed, based on the transceiver circuit shown in figure 4.12b, in which the same coil is used for both transmission and reception [167]. During transmission, the digital signal is used to pulse a sinusoidal oscillator, based on an LM3080 transconductance amplifier, on and off so that the transmitted carrier signal is inductively coupled to an identical circuit in receive mode. During reception, the received signal is simply rectified, smoothed and fed to a comparator which reproduces the transmitted digital signal. As this circuit consumes negligible power when not transmitting and because message transactions are usually short and infrequent, the average current drain on the node circuit battery would be very small, typically a few microamps. Where free access to the node was available, communications could take place simply by bringing the transmitting coil of the HHC into contact with the wall of the node. If access to the node was inconvenient, a shielded wire link could be installed between the node and a conveniently situated pick-up coil against which the HHC could be placed.

4.8 Fail-Safe Design

Throughout the development of this system, safety and reliability have been optimized by circuit integration and low power design. However, care must be taken to ensure that any fault which may occur is contained in a safe manner. The inclusion of a high capacity lithium cell in the design has been of some concern from the start of the project. Although battery technology is now sufficiently advanced that lithium cells are safe for most applications, current limiters both inside the cell itself (incorporated at cell
Figure 4.12a. Block diagram of inductive pick-up data link

Figure 4.12b. Half-duplex transceiver based on inductive pick-up
manufacture) and in the supply lines to the node circuit, should be used to guard against possible short circuits. Additional protection can be obtained by using several independently limited low capacity cells in parallel but this would result in a bigger and more expensive battery. Protection diodes would also be required to prevent any reverse currents damaging the battery. The risk of accidental cell damage can be minimized by encapsulating the cell in a robust material, but may be unnecessary since the node should be reasonably invulnerable to accidental damage.

To limit the effects of faults within the node electronics, it is proposed that all lines between the interface chip and microcomputer are buffered so that a fault in one device does not cause the complete failure of the other. Should a fault occur which is limited to one device, the node may still be of some use. For example, with a fault restricted to the interface chip, the microcomputer could initiate a self-test in the interface and report the fault condition to an operator via the hand-held communicator. Hardware faults in the microcomputer are more difficult to detect since the fault affects the only source of intelligence in the node. Intermittent software failures, however, such as spurious jumps out of a programme, can be detected by a simple watchdog timer circuit which is periodically reset during the normal operation of the microcomputer. If a fault occurs and the timer detects it, the microcomputer is forced into a start-up routine which initializes the device into a recoverable state. The above design features represent some of the simplest means of mitigating the effects of a fault in the node circuit and resulted from a compromise between the cost of fail-safe design, in terms of additional components and complexity, and the benefits attainable.

4.9 Summary

The feasibility of a 2×2 node circuit that can operate over a five year period from a D-type LiSOCl₂ cell has been demonstrated. To achieve this lifetime, extensive use was made of component integration in low power CMOS technology and a low mark/space ratio PPC communications scheme. The prototype node circuit basically consists of a battery, two optical receivers, two optical
transmitters, a single-chip CMOS microcomputer and a CMOS communications interface chip. Since microcomputer technology is now well established, the design challenge centred mainly around the communications interface and the very low power optical data link. A prototype bread-board of the interface circuit was built using 47 standard CMOS integrated circuits. This formed the basis of the interface chip which was fabricated on a MEDL MA2024 CMOS gate array. As the design required only 846 gates (a 29% reduction compared to the bread-board), two interface circuits were fabricated on each chip and a 75% yield of operational interface circuits was obtained. Several error detecting features aimed at improving the integrity of the data link were incorporated on the chip, as well as a self-test facility to reduce testing times and hence, manufacturing costs. The self-test procedure produced a fault coverage of 86.7% using only three 38-bit test vectors and required typically 1.5ms to complete. The chip consumed about 15μA with a normal workload (in a 64-node DSBN), which left 55μA from the budgeted figure of 70μA to power the 1MHz oscillator.

A short optical data link was constructed using widely available LEDs and photodiodes to demonstrate the feasibility of a 500m low power link. Results obtained suggested that a link was possible within the 70μA budget provided a large core glass fibre with an attenuation of less than 10dB/km was used. Other communications media are possible, such as transformer coupled twisted pair or coaxial cable, but input and output to the node circuit must be in PPC format. Where PPC cannot be supported by the communications medium, an independently powered interface box between the line and the node may be a possible solution. In the node circuit itself, provision was made for a 16-bit serial sensor/actuator interface and a hand-held communicator interface, but alternative parallel interfaces could be accommodated at the expense of interface lines.

The reliability of the node circuit, and its effect on overall system reliability, will be discussed in Chapter 6.
This chapter describes the procedures used by a loop controller supervising a Double Skip-Braid Network (DSBN) consisting of up to 64 addressable nodes. The loop controller (or simply the controller) also forms part of a main ring of controllers, and must therefore perform the dual tasks of controlling the DSBN whilst communicating with a larger host computer via the main ring. Since the interface to the main ring is most likely to be of a standard design, only the control aspects of the DSBN will be discussed in detail. Basically, the functions of the loop controller are to:

a) configure the network at system power-up so that access is available to as many operational nodes as possible
b) poll accessible nodes sequentially for data transactions
c) detect and locate faults in the network (eg. loss of a node or optic data link)
d) reconfigure around faults with minimum delay
e) occasionally, stop polling and access a specific node
f) detect complete network failure with minimum delay
g) communicate with the main ring of controllers.

Normally, the controller polls each remote node sequentially, sending actuator data or receiving sensor/status data with each message. One complete scan of the network occurs every 250ms. When a fault is detected in the network, the controller must locate the fault, reconfigure around it as quickly as possible and re-enter the polling sequence. Since the tasks of preparing and receiving messages for a fully operational network are specific to each application, and are relatively straightforward, the design challenge centres around the procedures used by the controller to detect and locate faults, to reconfigure around faults, and to generate new polling sequences once the new network configuration has been established. The objective is to develop a control strategy which can be implemented at reasonable cost and which is flexible enough to handle networks of different sizes. It should also be adaptable to other network topologies which exhibit different degrees of fault tolerance.

This chapter describes the control procedures and software algorithms specifically developed to detect, locate and reconfigure
around faults. A possible hardware implementation of the loop controller is proposed and its performance, concerning network reconfiguration, is discussed. To ease the development process, only faults which result in the complete loss of a node or optic data link are considered. However, procedures dealing with partial failures, such as intermittent transmitter or receiver faults, are discussed at the end of the chapter.

5.1 Representation of the Double Skip-Braid Network (DSBN)

To perform message routing through the network, the loop controller must know which optical links are operational and which are currently enabled. An optical link is operational if the optical transmitter supplying the link, the optical fibre and the optical receiver are ALL operational. Therefore, when searching for a path through the network, no information need be stored concerning the condition of each node since the required information is already implied in the store of link status. That is, if a path of operational links between the controller and a node, and back to the controller, cannot be established, that node is useless and will be labelled as such until the network is repaired.

For a network containing e links, the loop controller must store 2e bits of information: e bits representing the operational status of each link and the other e bits representing the enabled status. These bits could be stored as two adjacency matrices (see Chapter 3, section 3.6) but manipulating \((n+1) \times (n+1)\) matrices (where \(n\) is the number of remote nodes) can become cumbersome. Instead, a much simpler representation was developed based on the fact that the DSBN topology of figure 3.11a can be redrawn as a directed graph as shown in figure 5.1. This representation is very regular, consisting of \(\frac{n}{2} + 1\) identical cascaded blocks, each of which has three input nodes (labelled a, b, and c) and three output nodes (labelled a', b', and c'). The three inputs can be described as a three element input vector, \(X^m\), and the three outputs as a three element output vector, called \(X^{m+1}\) since it forms the input for the next block. A fully populated connection matrix, \(C^m\), can now be constructed that relates \(X^m\) to \(X^{m+1}\), and is given by:
Figure 5.1. Unidirectional DSBN (block representation)

(numbered arrows indicate the order of message transmissions)

Figure 5.2. Depth first scan routine
where $u, v, w, x$ and $y$ are link labels. Since only two link states ($1 = \text{operational}$, $0 = \text{non-operational}$) will be considered, $C_m$ becomes a Boolean matrix representation of all possible paths between the inputs and outputs of block $m$. The output vector, $X_{m+1}'$, can then be determined from $X_m$ and $C_m$ using:

$$X_{m+1}' = C X_m = T X_m$$

(5.1)

where the transition matrix, $T$, is simply the transpose of $C$ ($t$ denotes transpose). By repeated Boolean multiplication, any output vector $X_k$ can be determined given the initial input vector $X_0$ and the connection matrices $C_0', C_1', \ldots, C_{k-1}'$ using:

$$X_k = T_{k-1}' T_{k-2}' \ldots T_1'T_0 X_0$$

(5.2)

Using the set of fully populated connection matrices, equation 5.2 yields all possible paths through the $k$-block network for the input vector $X_0$. Under normal operating conditions, however, a message will follow only one path through the network and that path will be described by the set of enabled optical links. If a set of sparsely populated connection matrices is used to represent the currently enabled status of all links, the expected output response for a given input vector can be determined. Thus, the controller must store $(n/2)+1$ connection matrices to represent the operational links and another $(n/2)+1$ for the currently enabled links. This requires only $10((n/2)+1)$ storage bits since each matrix contains no more than five non-zero elements. All path related calculations now involve manipulating a series of $3 \times 3$ matrices which is much simpler than the equivalent $(n+1) \times (n+1)$ matrix manipulation. If a link is operational and enabled, the appropriate element in both stored matrices is set to a '1'. If the link is operational but not enabled, only the element in the operational matrix is set to a '1'. If a fault is
located to a specific set of links, the appropriate elements in both matrices are set to a '0' until the fault is repaired.

5.2 System Start-Up Procedure

After the system is initially powered-up, the loop controller creates a data base of \(10((n/2)+1)\) bits to store the operational and enabled status of each optical link in the network. Assuming the remote network has been assembled and powered up, the controller enters a start-up procedure designed to establish the operational status of all links. After filling the operational matrices with the results obtained, the controller configures the network, updates the enabled matrices and enters the polling sequence.

One way to establish the initial state of the links is to exhaustively test each path through the network by sequentially enabling every available path, but because over forty thousand different paths are available for a fully operational 32-node DSBN for example, this can be a very arduous task. Although it is possible to reduce the scan time using a more sophisticated search technique [168], faults in the network can exist that mitigate the improvements attainable via any type of exhaustive search.

A network scan routine which utilizes the node intelligence is the Depth First Scan shown schematic ally in figure 5.2. This scan involves each node marking a test message as it traverses the network so that the loop controller can determine the path taken by the message, and hence the state of the links. After system power-up, each node enters a scan mode whereby it enables both receivers and awaits the arrival of a test message. The test begins when the loop controller transmits to node 1 on the inner loop shown in figure 5.2, a test message consisting of a 14-bit address header, a 5-bit node count representing the number of nodes traversed by that message, and 19 bits of link status information (no error detecting bits). If this message is received successfully by node 1, the node count is incremented by one and a bit is set in the 19-bit data field (at bit position equal to the node count) to a '1' if the message arrived on receiver 1 or a '0' if it arrived on receiver 2. The message is then re-addressed to node 3 and retransmitted. If node 3 receives the
message, the node count and 19-bit data field are updated accordingly before the message is passed on to node 5. This procedure is repeated until the message reaches the loop controller, which uses the node count and the link status field to retrace the path taken by the message, and hence mark the links as operational. The outer loop can then be tested by transmitting a fresh test message to node 2.

So far, only transmitter 1 of each node has been tested. For complete scan coverage, each node m must also test transmitter 2 by re-addressing and transmitting the same test message to node m+3 as was previously sent to node m+2. To avoid message clashes, a node should transmit only when the network is "quiet" (i.e. no message is currently traversing the network), and assuming that the highest numbered node is allowed to test transmitter 2 immediately after transmitter 1, lower numbered nodes must wait progressively longer periods before the network goes quiet. The numbered arrows in figure 5.2 show the order of test message transmission. For example, node m must wait for node m+2 to test both transmitters before it can test transmitter 2, and node m-10 must wait over five times as long as node m. Using this technique, each node processes a series of single 40-bit messages, and because each message contains 19 link status bits, networks containing up to 38 nodes can be scanned.

Undoubtedly the largest drawback of the depth first scan is the enormous wait times associated with low numbered nodes, with a complete scan of a 32-node DSBN taking just under 90s if a node transmits a message every 2ms. Furthermore, because each node enters the same scan mode for every message received, many already tested links are repeatedly tested. This type of scan differs from the previous scan only in that no network configuration messages are needed from the controller to perform the test. Although the scan time could be improved significantly, by about an order of magnitude, if the loop controller commanded a fully tested node to exit from the scan mode, the scan time still remains excessive and very dependent on the number and location of faults in the network.

Another scan routine that utilizes the node intelligence but produces a scan time which is independent of faults in the network is the Breadth First Scan. This scan is similar to the previous scan except that each test vector consists of a series of three 40-bit
messages and each node tests both of its transmitters almost simultaneously. Since each message contains a 14-bit address header and a 24-bit data field (no error detecting bits), 72 bits of data are available for link status information. Therefore, by allocating two bit locations to each node so that the bits represent the status of the links supplying receiver 1 and receiver 2, each node can mark the status of both its receiver links before passing on the test vector, and one test vector can cover up to a 36-node network.

During the scan routine, each node should receive a test vector on one of its receivers if an appropriate path is operational. After storing the test vector, the node must wait a prescribed period before updating and transmitting the new test vector. If during that wait time another test vector appears on the other receiver, the 72-bit data fields from both the received test vectors must be logically 'OR'ed together so that the new test vector contains all of the link status information concerning the network prior to that node. Once the wait time has elapsed, the node (m say) adds its own two status bits to the stored test vector before re-addressing it and passing it on, first to node (m+2) and then to node (m+3).

The test procedure itself consists of a series of ordered test vector transmissions specially chosen so that a "wave" of link testing traverses the network without any message clashes. To simplify the description, consider the network as being partitioned into n/2 blocks as shown in figure 5.3a. Before the test procedure begins, all nodes enter a scan mode whereby they enable both receivers and await a test vector. The loop controller then transmits a set of test vectors into blocks 1 and 2 in the following sequence:

1) node 1 via its receiver 2, node 1 via its receiver 1
2) node 2 via its receiver 2, node 2 via its receiver 1
3) node 3 via its receiver 2.

Each test vector takes approximately 6ms (denoted 1 unit of time) to transmit, and transmissions begin one unit apart. With this transmission sequence, the following node triggering options are possible (see figure 5.3b):

a) node 1 could be triggered at time 0 via receiver 2 OR at time 0+1 unit via receiver 1.
b) node 2 could be triggered at time 0+2 units via receiver 2 OR
Loop
Controller

Initial triggering sequence: \(1, 2, 3, 4, 5\).

Figure 5.3a. DSEN partitioning for Breadth first scan

Figure 5.3b. Order of node triggering during Breadth first scan routine
at time 0+3 units via receiver 1.

c) node 3 could be triggered at time 0+4 units via receiver 2.

Therefore, all links leading to nodes 1 and 2 of block 1 have been tested by the time 0+4 units. At the same time, node 3 of block 2 could be triggered. Therefore, in order to keep the wave of triggering intact for the next block, wait times for nodes 1 and 2 must be found so that the same triggering sequence occurs in block 2. By using the timing diagram of figure 5.3b, it is clear that if each node waits 5 units after being triggered via receiver 2, or 4 units via receiver 1, and transmits the new test vector on transmitter 2 one unit after transmitting on transmitter 1, the wave of triggering does indeed stay intact as it traverses the network and no message clashes occur. Thus, only two wait-times, both of which are independent of the node address, are required for each node, one short wait-time, $w = 4$ units, and one long wait-time, $w = 5$ units.

As shown in figure 5.3b, the wavefront advances one block every 4 units and if 1 unit equals 6ms, the scan routine takes about 408ms for a 32-node DSBN irrespective of any network faults. Moreover, if each node recognizes the "zero address" as the address used during scan mode, both its transmitters could be tested simultaneously and the scan time would be reduced to 204ms, which is less than the normal 250ms network polling time. This scan routine is therefore much faster than the depth first scan because each link is tested only once and the loop controller need analyse only five test vectors for the complete operational link status of the network. The main disadvantage of this routine is that each node must have at least $2n$ bits of test vector storage Random Access Memory (RAM) for an n-node network, but this is not a problem for a microcomputer based node.

5.3 Fault Finding Routine

It is reasonable to assume that no faults exist in the network when the system is initially powered-up and the loop controller simply configures the DSBN into two independent loops, each of n/2 remote nodes as shown schematically in figure 5.4a. The set of enabled connection matrices is also updated (see figure 5.4b) and the normal polling sequence commences. Should an enabled link or node
Figure 5.4a. 12-node DSBN as two independent loops

Figure 5.4b. Enabled connection matrices for perfect DSBN
subsequently fail, a loss of the optical signal will result. Although fault detection can be achieved by means of a time-out mechanism in the loop controller or remote nodes, fault location is more complicated because the fault symptom could have been caused by ANY node or link failure in the currently enabled path.

One fault detecting and locating procedure involves installing a time-out mechanism in each node and the loop controller, that monitors the period during which no optical signal is received. If this period exceeds a prescribed limit (more than the normal 250ms network polling time), the node assumes that a fault has occurred and self-generates a full 40-bit message on both transmitters which the controller uses to locate the fault. To prevent several nodes timing out simultaneously, the time-out period for each node is proportional to its address. This ensures that only the node immediately "downwind" (ie. higher numbered) of the fault self-generates. As an example, consider the faults shown schematically in figure 5.5a in which a link failure occurs in the inner loop shortly before another link failure occurs in the outer loop. In both cases, the faults are located almost immediately by the controller irrespective of when they occur. However, consider the double fault shown schematically in figure 5.5b in which two link failures occur simultaneously in separate parts of the same loop. Although the controller locates fault "B", it fails to locate fault "A" because the self-generated messages are masked by the downwind fault. Moreover, when the controller sends out configuration messages to bypass fault "B", they will be blocked by fault "A" and the reconfiguration will fail.

Fault "A" can be located by sequentially enabling both receivers of the nodes on the other loop, but this procedure becomes very specific for certain fault combinations and very laborious. In fact, a procedure can be tailored to locate every combination of double fault but different faults can take different times to locate and the procedure is too cumbersome (ie. its becomes a trial and error technique) for any simultaneous fault of degree more than two. Nonetheless, this procedure can be used to detect and locate any number of chronologically isolated single faults (ie. no fault occurs when the controller is reconfiguring around a previous fault) in approximately 300ms without disruption to the other loop.
Figure 5.5a. Two faults - one on either loop

Figure 5.5b. Two faults on the same loop
A fault detecting and locating procedure that requires the same time-out for the nodes and the loop controller, is based on the breadth first scan described previously. When a fault occurs, the controller and all nodes downwind of the fault time out (after just over 250ms). The controller then initiates the network scanning routine and gathers the operational status of all the optical links. A simple comparison of the results obtained with the stored operational matrices yields the location of the fault. This procedure is attractive because it can be performed in about 200ms irrespective of the severity of the fault. Moreover, any combination of faults which can be tolerated by the DSBN (ie. a recoverable fault) will be located, even simultaneous faults of degree more than two.

The main disadvantage of this procedure is that during the network scan, all messages are dedicated to carrying link status information and thus cannot be used for normal sensor or actuator data, even if the fault affects only one of the loops. In applications which cannot tolerate the momentary loss of the remaining operational loop, the scan messages could be interleaved with normal polling messages but the network scan would then take longer to complete.

Two fault detecting and locating procedures have been presented. The self-generating procedure locates single isolated faults immediately the self-generated message is received, but requires a different time-out for each node and is cumbersome for multiple simultaneous faults. The network scanning procedure, on the other hand, requires the same time-out at each node and detects any type of recoverable fault in a specified time. Unfortunately, even for single isolated faults, the complete scan routine must still be performed and so takes longer to locate single faults than the former procedure. Therefore, since single faults are much more probable than double faults (see Chapter 6), a hybrid procedure could be adopted that uses self-generation for single faults and network scanning for more serious faults. Each node then needs two time-outs, the first triggers the self-generation mode and the second triggers the network scanning mode. In practice, however, the scanning procedure alone may be preferred because the necessary software routines already reside in the loop controller and remote nodes, with the only additional
feature required in each being a simple time-out mechanism.

5.4 Network Reconfiguration Algorithm

Having located a fault in the network, the loop controller must establish a new network configuration that effectively bypasses the faulty element. The problem can be generalized to an \((n+1) \times (n+1)\) matrix calculation which uses the property that all paths between any pair of nodes which traverse \(k\) links can be determined by raising the adjacency matrix, representing the operational optical links, to the power \(k\) [157]. By raising that matrix to the power \((n/2)+1\), which is the maximum number of links in any path through the DSBN, a path will be found to bypass any fault if such a path exists. However, the large amount of computation required by this technique makes it suitable only for small networks.

To minimize the amount of computation required, use can be made of the highly regular block representation of the DSBN shown in figure 5.1. By determining how many blocks are affected by the fault, a processing boundary can be established that limits the reconfiguration processing only to those blocks affected by the fault, unlike the previous technique which searched through all possible bypass paths. A limited area reconfiguration algorithm was developed that searches for a bypass path only in the area adjacent to the fault. As a result, the bypass path can be determined very quickly. The algorithm was based on two principal objectives:

a) to reconnect into the network any node disconnected by the fault
b) to minimize the amount of reconfiguration surrounding the fault.

Performing network reconfiguration requires only two data bases, one representing the set of operational connection matrices and the other representing the set of enabled connection matrices. To reconfigure around a fault, the loop controller first performs a Forward Fault Search which is designed to identify all nodes downwind of the fault which are rendered inaccessible by that fault. All inaccessible nodes are stored as "implied" faults on a fault queue. Once every downwind ramification of the original fault has been assessed, the controller begins searching for paths through the network to bypass the faults in the queue. The Block Reconfiguration
Routine fetches faults individually from the queue on a last in first out (LIFO) basis and searches for a bypass path only in the block containing that fault. The set of operational matrices is used to determine which communications links are available for use. If a bypass path is found, the desired network configuration is stored in the set of enabled matrices. If, however, a path cannot be found, the next fault is fetched from the queue. This process is repeated until the fault queue is empty or until the network is deemed to be disconnected. On completion of the block reconfiguration routine, the network configuration necessary to bypass the original fault is stored in the set of enabled matrices. In short, network reconfiguration consists of a search downwind from the fault to determine the upper fault boundary, followed by a series of block reconfigurations working back from the upper boundary through lower numbered blocks. The controller must then send out the necessary reconfiguration messages and generate a new polling sequence for the new network configuration. A more detailed description of these procedures will now be given.

5.4.1 Forward Fault Search

After locating a fault, the loop controller begins the forward fault search by identifying the fault using a 5-bit block number, a 1-bit marker denoting the inner or outer loop, and a 2-bit number signifying which transmitter or receiver link is faulty. Any number of faults can be passed to the search but because they are processed as individual link faults, every multiple fault (e.g. a complete node failure causes four link failures) is represented as a series of link failures. The search stores each fault input by the controller in two queues, one queue (FSQ) contains the faults still to be processed by the search and the other (RQ) contains the faults to be bypassed using the block reconfiguration routine. A flow chart of the forward fault search is given in figure 5.6. If the size of FSQ is greater than zero, the fault at the top of FSQ is fetched on a LIFO basis and deleted from FSQ. The number of the block containing the fault is first checked to prevent the search continuing after the last block in the network. Then, a search of the operational matrices is made to
Figure 5.6. Flow chart for forward fault search
determine if that faulty link was the last operational link supplying a node. If it was, that node is now inaccessible and both its transmitters are labelled as faulty before being added to both FSQ and RQ so that the search can continue. If a fault is removed that does not render a node inaccessible, the next fault in FSQ is fetched and the search continues. This procedure is repeated until FSQ is empty and every downwind ramification (or implied fault) of the original fault has been determined. On completion of this search, RQ contains a list of all faults (actual and implied) to be bypassed.

5.4.2 Block Reconfiguration Routine

The block reconfiguration routine also uses a fault queuing structure which enables single link faults to be processed individually. A reconfiguration is attempted using only the available operational links in the block containing the fault and if no bypass path exists within that block, the resulting implied fault "upwind" (i.e., lower numbered) of the original fault is added to RQ for subsequent processing. This procedure is repeated until RQ is empty or until the network is deemed to be disconnected. The block reconfiguration algorithm was developed around the criterion that no node should be left disconnected from the rest of the network, and because the fault queuing scheme enabled the same routine to be used on every fault, the resulting algorithm was straightforward to formulate. Basically, it consists of a set of five simple rules, each defining a procedure to follow in the event of a fault in one of the five links in a block. The rules are depicted as a set of five flow charts in figures 5.7a and 5.7b. The aim of each rule is to re-establish a connection to, or from, any node that is left disconnected by the fault being processed. As an example, consider a fault in link (b,c') of a block (see figure 5.1). Within this block it is only node "c" that is left disconnected and so an attempt is made to enable link (c,c'). Although link (b,c') is also link (a,b') of the previous block, no reconfiguration is attempted at this time because that (a,b') link will already have been queued for subsequent processing. The additional complexity of rules 3, 4 and 5 stems from irregularities in the network as it meets the controller, but the
1) \((a, b')\) fails

Is \((a, a')\) operational?

- **Yes**: enable \((a, a')\) → STOP
- **No**: 2) \((b, c')\) fails

2) \((b, c')\) fails

Is \((c, c')\) operational?

- **Yes**: enable \((c, c')\) → STOP
- **No**: 3) \((c, a')\) fails

3) \((c, a')\) fails

Is \((c, c')\) operational?

- **Yes**: enable \((c, c')\)
- **No**: Block no. < \(\frac{n}{2}\)?

- **Yes**: Is \((a, a')\) operational?
  - **Yes**: enable \((a, a')\) → STOP
  - **No**: 2) \((b, c')\) fails
- **No**: STOP

*Figure 5.7a. Algorithms to bypass link failures*
4) \((a,a')\) fails

5) \((c,c')\) fails

Figure 5.7b. Algorithms to bypass link failures (continued)
additional checks are logically straightforward given the topology of the DSBN.

A flow chart of the block reconfiguration routine is shown in figure 5.8. If the size of RQ is greater than zero, the top fault is fetched on a LIFO basis and deleted. Since the routine operates from the upper block boundary, set by the forward fault search, back through the network until the reconfiguration is complete, the number of the block containing the fault is checked to prevent the reconfiguration continuing below block zero. If the block number is greater than zero, a search of the operational matrices is made to determine if the faulty link was the last operational link emanating from a node in that block. If it was, that node is now useless and both its receivers (which are in the previous block) are labelled as faulty and added to RQ for subsequent processing. Then, an attempt is made to reconfigure around the fault in this block using the link bypass algorithms previously described. The whole procedure is repeated until RQ is empty or until the network is deemed to be disconnected. Once the routine is complete, the desired network configuration is contained in the set of enabled matrices and a scan is made through the set of operational matrices to generate a list of inaccessible nodes.

In summary, when the controller invokes the network reconfiguration algorithm, a forward search is performed to determine how many downwind links are actually rendered inaccessible by the fault. With this established, the block reconfiguration routine begins reconfiguring around each fault on a block by block basis, working from the upper block boundary back through the lower numbered blocks. The routine ends when all faults have been bypassed or when no operational path exists through the network.

5.5 Polling Sequence Generator

On completion of the network reconfiguration algorithm, the controller must establish the desired network configuration using the set of enabled connection matrices and recommence the polling sequence. The Polling Sequence Generator creates a list of messages each containing the following information:
Figure 5.8. Flow chart for block reconfiguration routine
a) the node number to which a message can be sent
b) which controller transmitter to use
c) the controller receiver at which the return message will arrive
d) a marker denoting a configuration or normal polling message.

Using this list, the controller knows which transmitter should be used to access a particular node and whether or not that message is free for normal polling use or is in fact, a configuration message. Under normal operating conditions, the controller simply scans through the list, polling nodes in the order in which they appear. The list is also cyclic in that the controller jumps back to the top of the list after the last message in the list has been sent. For a perfect network, the list will contain no configuration messages as the network is configured as two independent loops. If a fault occurs, however, it is more than likely that additional configuration messages will need to be embedded in the list to ensure access to all available nodes. For example, if link (a,a') fails in a given block (see figure 5.1), link (c,c') is enabled to access node "c'" but link (c,a') is then enabled to access node "a'". In this case, node "c" is labelled as an "alternating node" since its transmitters may be alternately enabled at some time. For a given network configuration therefore, the polling sequence generator must establish which paths need to be enabled to access all nodes, and the order in which these paths should be enabled. A new cyclic message list must then be generated with the minimum number of embedded configuration messages so that the overall scan time is not increased unduly. Furthermore, a list should be chosen that does not disproportionately affect the workload of any set of nodes (ie. no node should regenerate every single message on the network if this can be avoided).

5.5.1 Partial Path Listing Routine

Even though only one path through the real network should be configured at any one time, the set of enabled matrices can represent several different paths concurrently since it contains the set of optical links which must be enabled, at some time during the normal 250ms network polling period, to access all available nodes. This is possible because the controller can selectively configure each of the
paths as required.

Before the polling message list is generated, the controller determines which paths will need to be configured. This is done by means of a depth first search through the set of enabled matrices to produce a list of possible paths (each path is simply a string of node numbers). To speed up the search, a technique was adopted that prevented previously explored nodes from being re-explored. A flow chart for the path listing routine is given in figure 5.9. The routine begins by marking all nodes as "unexplored". Then, each unexplored-enabled link emanating from the controller is stored (ie. node number, block number, position in path list and link identifier) in a path search queue (PSQ) and those links are marked as "explored". Once all enabled links from the controller have been stored, the top link is removed from PSQ (LIFO) and explored. The node terminating the explored link is stored in path list and all unexplored-enabled links emanating from that node are stored in PSQ before being marked as explored. Again, the top link is removed from PSQ and the search continues until the controller or a fully explored node is encountered. At this point, the path list contains a string of node numbers (terminated with a special end marker) representing a path or partial path through the network. This path is then stored in PATH_STORE for future use. The next path through the network can now be determined by removing the top link from PSQ, and because the associated node position in the last path list was saved, the node terminating that link is stored at the correct position in a new path list. Since the new search begins from that link location, the new path generated will be identical to the previous path up until that location, beyond which the paths will diverge. The path listing routine terminates when all enabled links have been explored.

5.5.2 Complete Path Listing Routine

The paths contained in PATH_STORE are, however, incomplete in that some may have stopped before reaching the controller (ie. they reached a fully explored node). Therefore, a list of complete paths must be generated from PATH_STORE. This task is relatively straightforward and is described with the aid of the flowchart in
Mark all nodes as unexplored

Store unexplored-enabled links of block 0 in PSQ and mark the links as explored

Any elements in PSQ?

YES

Fetch, then delete, top link from PSQ

Explore link and save destination node in path list

Store unexplored-enabled links from that node in PSQ and mark the links as explored

Continue this depth first search?

NO

YES

Save end marker in path list and copy path list to PATH_STORE

STOP

Figure 5.9. Flow chart for depth first search path lister
The routine involves copying each path from PATH_STORE into NEWPATH_STORE, the list of complete paths. When an end marker is reached, it is the controller or a fully explored node, the copying ceases. If the copying stops at an explored node, the routine checks all previously copied paths in NEWPATH_STORE for a path from that explored node back to the controller. If one is found, that previous "tail" path (node-to-controller) is concatenated with the present "head" (controller-to-node) to form a complete path. Since there may be more than one tail path, another copy of the partial path is then made so that the search for tail paths can continue. At the end of this copying routine, NEWPATH_STORE contains a list of all possible paths through the network for the current set of enabled connection matrices.

5.5.3 Transmitter Assignment Routine

Having listed all the possible paths, the controller must now choose which path should be used to access each node in the network. Since a principal aim of the polling sequence generator was to minimize the number of embedded configuration messages, a weighting scheme was developed to compare the accessibility of each node via each of the controller transmitters. Nodes which are directly accessible from a given transmitter with no alternating nodes in the path (ie. each node always uses the same transmitter), receive the highest weighting. Nodes which are accessible via paths containing alternating nodes (ie. a node may use alternate transmitters), receive a lower weighting and the more alternating nodes in the path, the lower the weighting.

The list of possible paths is divided into three groups, according to the first node (1, 2 or 3) in each path, and the cumulative weighting for each node in each group is calculated. For example, if node 20 can be reached via three different paths from source node 1 and the weights for these paths are 0.125, 0.125 and 0.25, the cumulative weighting for node 20 from source node 1 is 0.5. This figure is representative of the probability that node 20 is accessible from source node 1 at any instant in time given that the transmitters in the alternating nodes are randomly set. Since the
Figure 5.10. Flow chart for complete path lister.
index - (i) denotes path number (ie. a string of numbers)
i,j,k,p,NDX and END are registers
NDF is a flag

Figure 5.10.(continued). Flow chart for complete path lister
controller must configure each of the alternating nodes to access a specific node, a node with a high weighting should, on average, involve fewer embedded configuration messages than a node with a low weighting. Thus, by creating three tables, headed SN1, SN2 and SN3 (where SN denotes source node), and listing the nodes accessible via those source nodes with their cumulative weightings, a basis has been established for the controller to choose which transmitter should be used to access a given node. Since additional configuration messages increase the network polling period, it is desirable to allocate each node to the transmitter with the highest probability of accessing that node. This, therefore, is the main consideration for node allocation. If two, or three, transmitters are equally desirable, the node is simply assigned to the transmitter currently with the fewest allocated nodes. Moreover, when a node is assigned to one table, it is removed from the other two. This results in a balanced set of tables which represents an effective means of minimizing the number of additional configuration messages sent out during network polling.

5.5.4 Message Listing Routine

Once each node has been assigned to one of the controller transmitters, the final list of polling messages can be generated. As discussed earlier, each entry in the list contains the number of the node to be addressed, which transmitter to use, the receiver at which the message should return and a marker denoting the message function. The list must contain all the configuration messages necessary to access the available nodes and must be fully cyclic. The fact that the list is cyclic implies that the network begins in a known configuration state at the top of the list and ends in the same state at the bottom of the list. Since the controller maintains a store of the current configuration of all alternating nodes, as well as the set of enabled connection matrices, it is relatively straightforward to force an initial state by configuring any one of the possible paths listed in NEWPATH_STORE. For simplicity, the first path in NEWPATH_STORE is chosen as the initial network state. The final MESSAGE LIST is then generated by following the listing routine described in the flowchart of figure 5.11.
Figure 5.11. Flow chart for message listing routine
i, j, k are registers

Figure 5.11. (continued). Flow chart for message listing routine
The aim of the routine is to generate the order in which nodes must be addressed by the controller and to determine which configuration messages, if any, must be sent out before access to a given node is possible. Since the MESSAGE_LIST is generated before network polling begins, and because the actual network configuration will change as configuration messages are sent out, this routine predicts the effect adding configuration messages will have on the network before adding any more. In short, the routine determines the future state of the network for any given position in MESSAGE_LIST and adds configuration messages to MESSAGE_LIST only when necessary. It begins when the top node, m say, is removed from SN1 and if it is non-zero, the controller checks whether or not that node is accessible using the currently enabled path (remember the routine predicts what path will be currently enabled for every position in MESSAGE_LIST). If node m is inaccessible, the controller stores the required configuration messages in MESSAGE_LIST and the "anti-configuration" messages in END_LIST. For example, if the message "node 10 activate transmitter 2" is stored in MESSAGE_LIST, the message "node 10 activate transmitter 1" is stored in END_LIST. Once the appropriate configuration messages have been stored, a normal message for node m is stored in MESSAGE_LIST and marked free for use. The same procedure is repeated for the top nodes in SN2 and SN3, and continues on through each table in turn until every node has been removed from all three tables. Before MESSAGE_LIST can be terminated, however, the list must be made fully cyclic by storing the configuration messages necessary to re-initialize the network. Since the configuration messages embedded in MESSAGE_LIST were also stored as the appropriate anti-configuration messages in END_LIST, the initial network state can be reconfigured simply by storing the configuration messages in END_LIST at the bottom of MESSAGE_LIST in the reverse order from which they were originally stored. As a result, MESSAGE_LIST is made fully cyclic and requires no further manipulation until another fault occurs in the network.

During normal operation, the controller simply scans down through MESSAGE_LIST addressing nodes as they appear in the list. If immediate access to a specific node is required, the controller can exit from the normal polling sequence and skip forward through
MESSAGE_LIST until that node is reached provided that any configuration messages skipped in the process are sent out before that node is addressed. This ensures that the network is configured to access that node before the message is sent. It is also possible to skip backwards (if this involves skipping fewer configuration messages) provided that any configuration messages skipped are sent out in their corresponding anti-configuration forms, but this requires additional processing by the controller to find the shortest route to that node in MESSAGE_LIST and to prepare the necessary anti-configuration messages.

For a 64-node DSBN, it is estimated that the latency between a request from a remote node and the response from the loop controller reaching that node, is about 8ms. However, an additional 4ms should be added for every configuration message that must be sent out before the node is accessible.

5.6 Repair To The Network

So far, procedures have been developed to cope with most conditions arising from both a fully operational network and a progressively deteriorating network. The following procedure is designed to cope with repairs to the network.

Considering that faults occurring in the network should be repaired relatively quickly, the controller must periodically, or on request from an operator, reconnect repaired elements into the network. Perhaps the simplest way to achieve this is for the controller, after being informed by the operator that the repair is complete, to initiate the start-up procedure described in section 5.2. A new MESSAGE_LIST would then be generated, reconnecting the repaired nodes or links, and a new polling sequence would begin.

Alternatively, to avoid any operator interaction whatsoever, the controller could periodically perform a network scan and reconnect the repaired elements, but it would be difficult to determine how often to perform the scan considering the random nature with which faults occur. Too short an interval and normal system operation may be disrupted unnecessarily, too long and repairs may go undetected for long periods. Therefore, the policy proposed is for the operator
to inform the controller via the hand-held communicator that the repair has been carried out. The controller then scans the network and informs the operator as to the success of the repair. Again, the advantage of this technique is that repaired elements can be reconnected into the network very quickly using procedures already resident in the loop controller.

5.7 Loop Controller Hardware

A block diagram of the proposed loop controller for the DSBN is shown in figure 5.12. It consists of five optical receivers, five optical transmitters, receiver/transmitter logic, a microprocessor with support circuitry, a serial interface to a hand-held communicator and an interface to the main ring. Since the controller normally configures the network into two independent loops, only two receivers and two transmitters will usually be required at any one time, although all receivers are normally enabled. Therefore, four communications interface circuits (two of the present interface chips) are required by the controller to perform all message transmission and reception, with two circuits being dedicated to transmission and the other two to reception.

For a full 64-node DSBN, messages are transmitted and received every 4ms on alternate transmitters and receivers under normal operating conditions (ie. with no faults in the network). As a result, each communications interface circuit processes one message every 8ms, on average, and so its workload is comparable to that of an interface circuit in a remote node. However, because the controller must prepare every message for transmission and analyse every message after reception, the workload of the controller microprocessor is far more demanding than that of the node microprocessor. Messages must be prepared in approximately 2ms and analysed in a further 2ms to achieve the desired 250ms network polling period. This requires a very fast processor or parallel processors, one for transmission and the other for reception, and efficient software programming. Assembly language programming can be used to optimize processing speed but to ease the development task, a high level language was chosen for the controller software. The C
Figure 5.12. Block diagram of the proposed loop controller

(CIC = communications interface circuit)
programming language was attractive because of the wide range of support facilities available and its global popularity. The Forth programming language was also attractive considering that the Novix NC4000 "Forth engine" [169], which is a CMOS processor capable of compiling high level polyForth directly into executable code, had recently been introduced. This device is essentially a 16-bit processor that can perform up to ten million instructions per second (MIPs), matching the INMOS transputer, and comes as part of a development board designed to operate with an IBM PC host. As the Novix NC4000 offered performance unattainable with existing single processors, such as the Hitachi HD68000, it was chosen as the target processor for the proposed loop controller. Furthermore, using the Forth programming language it was possible to construct and test a dictionary of useful programmes (Forth words) which could be used to write more complex programmes very quickly. Therefore, all the software for the loop controller was written in Forth, with some routines also written in C in anticipation of a possible change in target processor at a later date.

5.8 Network Reconfiguration using the Proposed Loop Controller

Since fault location takes on average about 0.5s from the occurrence of the fault, it is essential that the time required to generate the necessary reconfigurations and new polling sequence is small compared to this figure. To evaluate the speed of the software procedures developed here, an example 32-node DSBN was simulated and subjected to a variety of different fault conditions. Running polyForth on an IBM PC the desired network configuration required approximately 20ms to generate for each link fault. The new polling sequence required 1.8s to generate. In total, therefore, the network reconfiguration procedures require just over 1.8s for each input fault. Using this figure, it was estimated that the same procedures could be performed in approximately 30ms with the proposed Novix NC4000 (with 6MHz clock) based loop controller. This figure is small compared to the 0.5s fault location time and so does not adversely affect the overall speed of fault recovery for the system. Moreover, future improvements in the Novix processor should reduce this time.
still further, perhaps by a factor of two.

5.9 Summary

This chapter described the network control procedures for the DSBN, with particular emphasis on those used for fault location and network reconfiguration. To facilitate the development of the control procedures, a block representation of the DSBN consisting of a set of $3 \times 3$ Boolean connection matrices, was used throughout. Procedures were developed to configure the network after system power-up, to detect and locate faults, to reconfigure around faults, to generate polling sequences and to ease network repair. Several of these used the network scan routine which required just over 200ms to establish the operational status of all optical links in a 32-node DSBN.

Fault detection was based on the use of a time-out mechanism at each remote node and the loop controller, and assumed that all faults resulted in the complete loss of the optical signal (e.g. a link break). Although this assumption precludes certain intermittent faults caused by defective connectors, for example, or persistent faults such as jabbering transmitters (see Chapter 2), time-out procedures can be extended to accommodate most cases. For instance, a jabbering transmitter will result in a corrupted message (i.e. incorrect start bit sequence or wrong number of message bits) which will be detected by a subsequent node. Thus, each node could be programmed to deactivate the erroneous receiver and enter the network scan mode if it is not addressed within a prescribed period or continually receives corrupted messages. The result would be the same as if a complete link failure had occurred. A network scan would then locate the fault.

A block reconfiguration algorithm was developed that used a fault queuing structure to reconfigure around faults individually, on a block by block basis over a restricted area of the network surrounding the fault. As a result, the algorithm was straightforward to formulate, consisting basically of five simple rules. To complete the network reconfiguration, a node polling sequence was generated which minimized the number of configuration messages sent out during a normal poll of the network. Additional information, such as the
number of the controller receiver at which the message will return, was also produced to aid fault detection. Although the routine assumes that the order in which nodes are accessed during the normal 250ms poll period is unimportant, an ordered list could be generated at the expense of additional configuration messages.

After network repairs, it is proposed that the operator informs the controller via the hand-held communicator that the repair is complete. The controller can then test the link and reconfigure the network as required. One further point is that if the controller periodically scans the network, it is possible to update the store of operational link status, thus performing a condition monitoring function. In this way, a potentially dangerous situation may be predicted and consequently averted.

The necessary loop controller hardware was also discussed and the Novix NC4000 "Forth engine" was proposed as the target processor. However, in some applications a parallel processor architecture using dual Hitachi HD68000s, for example, may be preferred until the former is more established.

Overall, the work reported in this chapter suggests that it is possible to control a fault-tolerant DSBN using a connection matrix network representation and a simple block reconfiguration algorithm. Clearly, alternative control procedures exist but the procedures proposed here have proved capable of handling the main control difficulties associated with the DSBN, as well as coping with networks of widely varying sizes. Moreover, since the DSBN is an extension of the SSBN (see Chapter 3), a network more likely to be used in less critical applications, the controller could be modified quite easily to accommodate the latter network. A reliability analysis of the system is given in Chapter 6.
Modern process plants often require many control loops involving highly complex measurement systems. To cope with this complexity and to reduce the risk of human error, emphasis has shifted away from manual/electronic control to fully automated process control. This trend has been accompanied by continually increasing demands on the performance of these systems and an increased industrial and public awareness of the importance of system reliability, especially where dangerous processes are being controlled. This awareness has been further stimulated by a spate of incidents in certain nuclear and chemical process plants throughout the world over the last decade.

System reliability can be very difficult to quantify and even more difficult to prescribe since an acceptable level of risk is hard to justify for any system. Such considerations have prompted engineers to design systems with reliability in mind from the start. As a guideline, Taylor Instrument provided a document [170] outlining the reliability targets and estimation procedures for a typical process control system. This document has provided a foundation from which system reliability has been estimated and the performance of different systems compared.

A primary objective of the work reported here was to demonstrate the improved reliability and design flexibility, in terms of fault-tolerant capabilities, of communications networks based on 2×2 node elements. Various fault-tolerant network topologies were presented in Chapter 3 and to demonstrate the feasibility of a highly fault-tolerant communications network, a 2×2 node circuit was built and the Double Skip-Braid Network (DSBN) was chosen as the experimental topology. Having developed the node circuit hardware and the necessary control procedures for the DSBN itself, it is important now to investigate the reliability and performance of the system.

This chapter estimates the Mean Time Between Failures (MTBF) for a single fault-tolerant skip-braid network to determine whether or not the node circuit is sufficiently reliable to be used without redundancy at the remote communications nodes. Availability is then used as a measure of system performance to investigate the effectiveness of the proposed network control procedures and to
compare the DSBN with other prominent network topologies.

6.1 MTEF for a Single Fault-Tolerant Skip-Braid Network

Before a meaningful estimate of system MTEF can be made, it is necessary to outline what conditions actually constitute a system failure. The aforementioned guideline clearly states that:

"In the case of failure of the automatic control of a process variable, the operator will be able to control this variable manually. It will be difficult to take over two variables by hand. For this reason it is postulated that a control system must be considered as unavailable when more than one variable is not automatically controlled by the system or can not be monitored and/or manipulated from the operator station."

Although remote instrumentation can be replicated (by forcing several devices to perform exactly the same function) to a degree sufficient to reduce the probability of system failure to that of the probability of disconnecting the communications network, in practice it is economically desirable, in terms of the number of remote devices and associated cabling required to perform a given function, to minimize this redundancy. However, if the penalties associated with remote instrumentation redundancy can be neglected, the task of comparing different fault-tolerant systems depends purely on the topological constraints of each network (see Chapter 3).

To highlight the advantages of using the 2x2 node circuit as the basis of the fault-tolerant communications network, the reliability (quoted as MTEF) of a single fault-tolerant skip-braid network without node redundancy will now be estimated. In such a system the loss of any two nodes (sensor or actuator) constitutes a system failure irrespective of whether or not the communications network is disconnected. Before proceeding, the following assumptions are made.

a) Since the operational environment cannot be specified, the probability of communications link damage is assumed to be zero.

b) Since the loop controller operates in a relatively benign environment compared to the remote network and is not subject to any real design or power constraints being mains powered, it is assumed that the probability of loop controller failure can be
neglected at present. However, an estimate of the reliability of the proposed loop controller will be given, and its effect on the rest of the system will be discussed later in this chapter. Having made these assumptions the task of estimating the MTBF for the system basically involves estimating the MTBF for the remote nodes, and since the prototype node has already been developed, a meaningful estimate should be possible.

The failure rate $\lambda$ for the node circuit was estimated using the general expression for sub-system failure rate as detailed in MIL-Handbook 217D, section 5.2:

$$\lambda = \sum_{i=1}^{n} N_i \left( \lambda g \eta Q \eta L \right)_i$$

(6.1)

for a given environment where:

$\lambda = \text{generic failure rate for the (i)th generic part}$

$\eta g = \text{quality factor for the (i)th generic part}$

$\eta Q = \text{learning factor for the (i)th generic part}$

$N_i = \text{quantity of the (i)th generic part}$

$n = \text{number of generic part categories.}$

The expected operational environment is equivalent to a ground fixed (G_f) environment (i.e. installation in permanent racks with adequate cooling in an unheated building) and the specified component temperature level is 40°C. A list was made of all the components in the node circuit and the following failure rates were calculated with the aid of the generic failure rate data given in MIL-217D. With all components fully specified to MIL-M-38510 Class S:

$$\lambda = 17 \text{ failures per } 10^6 \text{ hours.}$$

With an average derating quality factor of 14.4 for the procurement of industrial quality components:

$$\lambda = 245 \text{ failures per } 10^6 \text{ hours.}$$

A learning factor of 1 was assumed throughout and an estimated failure rate of 45 per $10^6$ hours was used for the lithium cell on the
basis of earlier work performed by NASA [171]. Although every component failure will not result in a complete node failure (e.g., a resistor going out of tolerance), it is assumed here that every fault is fatal. This should produce a pessimistic estimate of the MTBF.

The probability of system failure increases with the node failure rate but decreases with the node Mean Time To Repair (MTTR), and industrial sources suggest that a MTTR of 8 hours for detectable node faults is practical. Since any n-node skip-braid network can tolerate a single node failure, and because two node failures constitute a system failure if no instrumentation redundancy is used, only three network states must be considered to determine the MTBF:

state 0) all nodes are operational
state 1) one node has failed, (n-1) nodes are operational
state 2) two nodes have failed.

Now assume that the network is in state 0 at time 0 and let $P_i(t)$ be the probability of the network being in state $i$ at time $t$, therefore:

$$P_0(0) = 1, \quad P_1(0) = P_2(0) = 0$$

and

$$P_0(t) + P_1(t) + P_2(t) = 1 \quad (6.2)$$

The network will still be in state 0 at time $t+\Delta t$ if:

a) the network was in state 0 at time $t$ and no node failure occurred during $\Delta t$ or

b) the network was in state 1 at time $t$ and the failed node was repaired during $\Delta t$ with no other failures occurring.

Therefore, provided $\Delta t$ is so small that only one failure or repair event can occur during $\Delta t$, the probability of a specific node failure during $\Delta t$ is simply $\lambda \Delta t$ and the probability of any node failure out of $n$ nodes is $n\lambda \Delta t$. Consequently, the probability of no node failures during $\Delta t$ is $(1-n\lambda \Delta t)$. If the probability of a failed node being repaired during $\Delta t$ is $\mu \Delta t$, where $\mu$ is the repair rate or $(\text{MTTR})^{-1}$, the network state probability at time $t+\Delta t$ is given by:

$$P_0(t+\Delta t) = [P_0(t).(1-n\lambda \Delta t)] + [P_1(t).(1-(n-1)\lambda \Delta t)\mu \Delta t] \quad (6.3)$$

$$P_1(t+\Delta t) = [P_0(t).n\lambda \Delta t] + [P_1(t).(1-(n-1)\lambda \Delta t)(1-\mu \Delta t)] \quad (6.4)$$

$$P_2(t+\Delta t) = [P_1(t).(n-1)\lambda \Delta t)] + P_2(t) \quad (6.5)$$
Since the limit as $\Delta t \rightarrow 0$ of $[P_i(t+\Delta t)-P_i(t)]/\Delta t$ is $\dot{P}_i(t)$, the above equations yield:

\[
\begin{align*}
\dot{P}_0(t) &= -n\lambda P_0(t) + \mu P_1(t) \\
\dot{P}_1(t) &= n\lambda P_0(t) - [(n-1)\lambda + \mu]P_1(t) \\
\dot{P}_2(t) &= (n-1)\lambda P_1(t)
\end{align*}
\] 

(6.6)

(6.7)

(6.8)

In matrix notation this becomes:

\[
\begin{bmatrix}
\dot{P}_0(t) \\
\dot{P}_1(t) \\
\dot{P}_2(t)
\end{bmatrix} =
\begin{bmatrix}
-n\lambda & \mu & 0 \\
n\lambda & -[(n-1)\lambda + \mu] & 0 \\
0 & (n-1)\lambda & 0
\end{bmatrix}
\begin{bmatrix}
P_0(t) \\
P_1(t) \\
P_2(t)
\end{bmatrix}
\]

(6.9)

The MTBF of a system $s$ is defined as:

\[
s = \int_0^\infty R(t)dt
\]

\[
= \int_0^\infty [P_0(t) + P_1(t)]dt
\]

\[
= \int_0^\infty P_0(t)dt + \int_0^\infty P_1(t)dt
\]

\[
= T_0 + T_1
\]

(6.10)

where $R(t)$ is the probability of successful operation during the time $t$. Now $T_0$ and $T_1$ can be found by solving:

\[
\int_0^t
\begin{bmatrix}
\dot{P}_0(t) \\
\dot{P}_1(t) \\
\dot{P}_2(t)
\end{bmatrix} dt =
\begin{bmatrix}
-n\lambda & \mu & 0 \\
n\lambda & -[(n-1)\lambda + \mu] & 0 \\
0 & (n-1)\lambda & 0
\end{bmatrix}
\begin{bmatrix}
P_0(t) \\
P_1(t) \\
P_2(t)
\end{bmatrix}
\]

(6.11)

assuming that the failure and repair rates are constant. Therefore:

\[
\begin{bmatrix}
P_0(\infty) - P_0(0) \\
P_1(\infty) - P_1(0) \\
P_2(\infty) - P_2(0)
\end{bmatrix} =
\begin{bmatrix}
-n\lambda & \mu & 0 \\
n\lambda & -[(n-1)\lambda + \mu] & 0 \\
0 & (n-1)\lambda & 0
\end{bmatrix}
\begin{bmatrix}
T_0 \\
T_1 \\
T_2
\end{bmatrix}
\]

(6.12)

and knowing that $P_0(0)=1$, $P_1(0)=P_2(0)=0$, $P_1(\infty)=P_2(\infty)=0$, $P_2(\infty)=1$, the
above matrix set reduces to:

\[
\begin{bmatrix}
-1 \\
0 \\
1 \\
\end{bmatrix} = \begin{bmatrix}
-n\lambda & \mu & 0 \\
n\lambda -((n-1)\lambda+\mu) & 0 & 0 \\
0 & (n-1)\lambda & 0 \\
\end{bmatrix} \begin{bmatrix}
T_0 \\
T_1 \\
T_2 \\
\end{bmatrix} \tag{6.13}
\]

which produces:

\[
T_0 = \frac{1}{n\lambda} + \frac{\mu}{(n-1)n\lambda^2}, \quad T_1 = \frac{1}{(n-1)\lambda}
\]

and so the MTBF for the network is given by:

\[
\theta_s = T_0 + T_1 = \frac{(n-1)\lambda + n\lambda + \mu}{(n-1)n\lambda^2} \tag{6.14}
\]

This equation defines the MTBF for any skip-braid network with no instrumentation redundancy, where system failure results purely from remote node failure. The equation is plotted in figure 6.1 for three networks of different size. To achieve a target MTBF of 80,000 hours (as prescribed in the Taylor Instrument guideline) for a 64-node network, the node failure rate must not exceed 20 per million hours. The corresponding figures for a 32-node and 16-node network are 40 and 81 per million hours respectively. These figures suggest that the node circuit developed here is suitable for use in a full 64-node measurement system without node redundancy only if components specified to MIL-M-38510 Class S requirements are used. The estimated failure rate of 245 per million hours for the node using industrial quality components suggests that only networks containing less than 6 nodes should be used without instrumentation redundancy. However, it must be stressed that failure rates calculated in accordance with MIL-217D can be very pessimistic, and improvements in excess of an order of magnitude better than estimated are not uncommon [172,173]. Although this insistence on pessimistic data leads to overdesign, it does make it easier to achieve predicted reliability levels. If a factor of ten improvement in failure rate was actually realized in practice, the node circuit could be used in skip-braid networks containing more than fifty nodes without remote node redundancy.
Figure 6.1. System MTBF vs Node failure rate (node MTTR=8 hours)

Figure 6.2. System MTBF vs Node MTTR (node $\lambda = 100$ per $10^6$ hours)
Another way to increase the MTBF is to reduce the node MTTR, but it is difficult to generalize on this figure since it depends greatly on the target application. However, the effect on MTBF for a skip-braid network of varying the MTTR is clearly shown in figure 6.2.

6.2 MTBF for the Loop Controller

The previous reliability analysis assumed that the probability of system failure due to optical link damage and loop controller failure was sufficiently small that it could be neglected. In practice, however, the reliability of the loop controller is critical to system reliability because its complexity is high compared to the remote nodes. Consequently, redundant design and fail-safe design techniques are often employed to enhance the reliability of system controllers [149,150]. In this work, the design of the loop controller hardware has been of secondary importance since its power budget and physical dimensions are relatively unrestricted. Moreover, the required technology is also well established. In view of this, the proposed loop controller served merely as a tool to verify the feasibility of the software developed for network control. Nonetheless, an estimate of the failure rate for the proposed loop controller should indicate what reliability enhancement techniques, if any, are necessary to justify the assumption that loop controller failure can be neglected.

Using the same procedure as was used for the node circuit, an estimate of the failure rate for the loop controller was made. With all components specified to MIL-M-38510 Class S requirements, a failure rate of 44 per million hours was estimated. However, applying a learning factor $\eta_L=5$ to the Novix NC4000 processor (it is a relatively new device) raises this figure to 164 per million hours. The failure rate for the loop controller using industrial quality components ($\eta_Q=15$) was estimated at 2460 per million hours. This figure corresponds to a MTBF of only 406 hours which is well below the absolute minimum requirement of 80,000 hours. It must be stressed that a MTBF well in excess of 80,000 hours is necessary to ensure that the reliability of the loop controller does not adversely affect the reliability of the overall system.
Apart from using better quality components, the MTBF of the loop controller can be improved most easily by using Triple Modular Redundancy (TMR) [131], but even with TMR a MTBF of only 3781 hours is achieved assuming the standard MTTR of 8 hours prevails. To achieve a MTBF of 80,000 hours using TMR, a basic loop controller failure rate of 515 per million hours would be required.

An alternative approach would be partial redundancy [131] in which three controllers are used, one operating and the other two acting as standby spares. Obviously this technique demands some diagnostic capabilities to detect controller malfunctions, but a MTBF in excess of 1,000,000 hours is possible assuming a failure rate (including fault diagnostics) of 2460 per million hours and a MTTR of 8 hours. These results are shown graphically in figure 6.3. The effect MTTR has on loop controller MTBF is also shown in figure 6.4. Once more it must be stressed that because the design of the loop controller has not been finalized here, these MTBF calculations serve only as very approximate estimates from which to explore the most likely reliability enhancing design techniques. Since the policy of pessimistic estimation was again adopted, the actual figures for such a loop controller could well be much better.

6.3 System Availability

The reliability analysis so far has considered only faults in the loop controller or remote nodes as being contributory factors to system failure, defined as the loss of two control variables. In practice, however, the probability of system failure due solely to the above factors can be made insignificant by replicating the controller and sensor/actuator functions (in theory, every node in the remote network could perform exactly the same function). If such redundancy is used, the probability of system failure simply becomes the probability of disconnecting the communications network by optical link damage. Even without massive instrumentation redundancy, the vulnerability of the communications links to accidental damage may be the largest contributory factor to system failure, especially in systems where the cabling is relatively unprotected. Thus, the invulnerability of a communications network to link failure can be
Figure 6.3. Controller MTBF vs Unit failure rate (unit MTTR=8 hours)

Figure 6.4. Controller MTBF vs unit MTTR (unit failure rate=2460/10^6 hrs)
used as an index of performance for the network.

A convenient way to represent this index is to use the availability parameter, $A$, which is defined as:

$$A = \frac{MTBF}{(MTBF + MDT)}$$  \hspace{1cm} (6.15)

where $MDT$ is the mean down time of the system. Since $MDT$ includes the time to detect and locate the fault, shut down the system, repair the fault, test the repair and restart the system, it is usually significantly greater than the $MTTR$ for a single system component. Therefore, an estimated $MDT$ of 100 hours, compared to the 8 hour $MTTR$, will be used for the following analysis.

Since the $DSBN$ can automatically reconfigure around a single link failure in approximately 0.5s, its availability is obviously much higher than the single loop network, for example, which would require more time consuming manual repair. A quantitative analysis of the performance improvements attainable for skip-braid networks can be made using a technique very similar to that used earlier to derive equation 6.14. It involves calculating the $MTBF$ for the system assuming that the 0.5s automatic repair time for single link failures in the skip-braid networks is so small, compared to the 100 hour $MDT$, that it can be neglected. Calculating system $MTBF$ then involves calculating the probability of the network being in an operational state at any given instant in time.

The $MTBF$ of a single loop network is simply $1/\mu$, where $\mu$ is the number of links in the network and $x$ is the link failure rate. The availability $A_{SL}$ of a single loop containing $n$ nodes (and a loop controller) using $MDT=100$ hours and $e=n+1$, is given by:

$$A_{SL} = \left[1+(n+1).MDT.x\right]^{-1}$$  \hspace{1cm} (6.16)

The single skip-braid network ($SSBN$), on the other hand, can tolerate any combination of link failures except for three or more adjacent links. Therefore, only link failures that contribute to a disconnecting group of three links need to be considered when calculating the $MTBF$. The network states required to be analysed are:
state 0) all links are operational
state 1) any one link has failed, other links in that disconnecting group are operational
state 2) any two links in the disconnecting group have failed, other link in that group is operational
state 3) the network is disconnected.

Knowing that the network is operational if it is any of the above states except state 3, the MTBF can be calculated as shown in appendix 2. The MTBF for the SSBN is:

\[
MTBF_{SSBN} = \frac{x^2(3e+2) + x\mu(2e+1) + 2\mu^2}{2ex^3} \quad (6.17)
\]

where \( \mu = (\text{link MTTR})^{-1} \). This figure is representative of the invulnerability of the network to link failures, the higher the MTBF the higher the invulnerability. The availability \( A_{SSBN} \) is given by:

\[
A_{SSBN} = \frac{x^2(3e+2) + x\mu(2e+1) + 2\mu^2}{x^2(3e+2) + x\mu(2e+1) + 2\mu^2 + 2(MDT)ex^3} \quad (6.18)
\]

The availability of the DSBN can be calculated in a similar way but because a group of five or more adjacent link failures is now required to disconnect the network, six network states must be analysed as described in appendix 2. The MTBF for the DSBN is:

\[
(25e+12)x^4 + (32e+3)\mu x^3 + (27e+2)\mu^2 x^2 + (12e+3)\mu^3 x + 12\mu^4 \quad (6.19)
\]

which corresponds to an availability \( A_{DSBN} \):

\[
(25e+12)x^4 + (32e+3)\mu x^3 + (27e+2)\mu^2 x^2 + (12e+3)\mu^3 x + 12\mu^4 \quad (6.20)
\]

The availability of each network (consisting of 32 nodes) is now compared in figure 6.5. As expected, the DSBN exhibits the highest
Figure 6.5. System availability vs Link failure rate (for a 32-node network with link MTTR = 8 hours and MDT = 100 hours)

Figure 6.6. System availability vs Link failure rate (same as above but with link failure rate multipliers: SSBN=1.5, DSBN=2.5)
availability of the three for all values of link failure rate. By indicating the failure rate at which the availability exceeds 0.9999, it can be seen that the DSBN maintains a comparable availability to the SSEN even when its link failure rate is a factor of ten greater than that of the SSEN. Figure 6.5 also verifies that the availability of the simple loop is unacceptably low for critical applications.

Since the average length of a link in the DSBN will be greater than the average link in the SSEN, it is reasonable to assume that a link in the former network will be more vulnerable than a link in the latter. In view of this, it is important to show the effect length dependent link failure rates have on system availability. To facilitate this, basic failure rate multipliers of 2.5 and 1.5 for the DSBN and SSBN respectively, were chosen on the basis that the DSBN (SSEN) requires approximately five (three) times the amount of cabling required by a simple loop. The resulting compensated availability for each of the above three networks is shown graphically in figure 6.6. Now, the superiority of the DSBN when the availability exceeds 0.9999 is not so pronounced, with the failure rate being only six times that of the SSEN for comparable availability. An interesting observation emerging from this graph is that when the basic link failure rate is about 115,000 per million hours, the DSBN and SSEN exhibit equal availability (i.e. at the crossover point). This result is, however, insignificant in practice because such a high link failure rate will not be encountered. Furthermore, the levels of system availability achieved at such failure rates are so low that neither network would find application. Nonetheless, this graph does show how length dependent link failure rates can affect system availability. The basic link failure rate crossover point for the DSBN and SSEN varies depending on the link MTTR and the size of the network. The rate increases as $(\text{MTTR})^{-1}$ or the number of nodes increases, and decreases as the DSBN multiplier/SSBN multiplier ratio increases, but in all practical systems the availability of the DSBN will always exceed that of the SSEN and the simple loop. To achieve a system MTBF of 80,000 hours, considering link failures as the only source of system failure, an availability in excess of 0.999 is required if the system MTT is 100 hours. The DSBN achieves this figure for basic link failure rates not
exceeding about 6000 per million hours (see graph 6.6), whereas the corresponding figure for the SSEN is about 1000 per million hours.

6.4 Summary

The reliability of skip-braid communications networks based on the 2x2 node circuit has been investigated. To perform a quantitative analysis, information concerning the expected failure rates of the remote nodes, loop controller and communications links must be available. However, since the target application cannot be specified accurately in this thesis, the only information currently available concerns the node circuit which was developed here. In view of this, and for reasons of simplicity, an individual analysis was performed for each of three parameters above. It was assumed throughout that all failures were random events and that the loss of two control variables constituted system failure.

Using results obtained via standard estimation procedures, it was suggested that the target MTBF of 80,000 hours could not be achieved using a simple non-redundant loop network based on the node circuit, even if the highest quality military components were used.

The ability of any skip-braid network to automatically bypass a single node failure, on the other hand, enabled the target MTBF to be reached albeit only for small networks. However, if military grade components are used or, indeed, the estimated node failure rate is excessively pessimistic, reasonably large (20 or so nodes) skip-braid networks based on the node circuit can be used without remote node redundancy. An important point to stress here is that without remote node redundancy, no improvement in reliability is obtainable beyond the simplest skip-braid network (ie. the SSEN), even when using a full star network. This is because the loss of any two nodes in any non-redundant system constitutes a system failure. Nonetheless, the estimated node failure rate suggests that the target MTBF is attainable if TMR of the remote nodes is employed and node failure is the dominant system failure mode.

Redundancy can be used both in the loop controller and at the remote nodes to such a level that the probability of system failure simply becomes the probability of disconnecting the remote
communications network. To achieve this, the reliability of the controller must be significantly higher than the minimum system MTBF of 80,000 hours. A brief investigation of the reliability of the proposed loop controller suggested that for a controller of similar complexity, TMR could not be used to any great effect. Instead, 1-out-of-3 standby redundancy appeared the best reliability enhancement technique, but required some fault diagnostic capability to ensure that a malfunctioning controller is removed quickly from the system. To achieve this, a test routine could be used that periodically checked the following points:

a) Network scanning is taking place.
b) PROM is not corrupted.
c) All cells of RAM are operational.
d) ALU routines are running correctly.
e) All critical I/O elements are stimulated to reveal any faults.

The routine could be contained in a small ultra-reliable kernel within each of the three loop controllers, and performed frequently enough to maximize system MTBF but not so frequently that it disrupted normal system operation.

If the probability of failure in the remote nodes and loop controller is sufficiently low, communications link damage becomes the dominant system failure mode. In such cases, the benefits of the highly redundant skip-braid networks are quite evident. The analysis has shown that the availability of the DSBN remains higher than the SSBN even when subjected to a link failure rate which is a factor of ten greater than that of the SSBN. Since this analysis was based on the probability of disconnecting the network, the availability of the (3:1) and (4:1) networks discussed in Chapter 3 could be calculated in a similar manner. In general, networks with similar cabling overheads will exhibit similar levels of availability if link failure is the dominant system failure mode. Therefore, the availability of the (4:1) network will be comparable to that of the DSBN, whereas the (3:1) network will offer an availability lower than the DSBN but higher than the SSBN. If node failure is the dominant failure mode, the (4:1) network will exhibit the highest availability since it can tolerate up to three node failures compared to two node failures for the DSBN and (3:1) network. In practice, however, it is prudent to
assume that link failure is the dominant failure mode since normally it is the only parameter which cannot be specified accurately at the design stage.

The reliability analysis presented in this chapter has highlighted the importance of accurate failure rate information concerning every system component. When such information is unavailable, the analysis produces a general expression for system reliability which can be used when the necessary information becomes available. The chapter has shown quantitatively how the reliability of the system depends on the individual reliabilities of the node circuit, the loop controller and the communications links. It has also stressed the benefits, in terms of availability, of using communications link redundancy in the form of skip-braid networks. Other more general factors affecting system reliability such as operating temperature, component derating policies, environmental conditions and maintenance programmes, are discussed by Arsenault and Roberts [174].
7 CONCLUSIONS AND PROPOSALS FOR FUTURE WORK

The potential of passive optical distributed measurement systems in hazardous process control has promoted great interest in both the industrial and academic communities, but further development is required before these systems can compete commercially with existing electrical systems.

The proposed Field Bus standard outlines realistically the desirable features for the next generation of distributed measurement systems, and emphasizes the need for a reliable low power digital communications system. The system described in this thesis, whilst not competing for the standard, attempts to conform closely with the proposals. Extensive use has been made of the developing technologies of low power circuitry, fibre optics and miniature high capacity batteries, to demonstrate the feasibility of a highly fault-tolerant communications network suitable for use in process control applications. Fibre-optic data links reduce susceptibility to EMI, and reduce cabling costs and weight. Battery-powered remote devices minimize the risk of global power failure without incurring the cost of galvanic isolation. Low power circuitry and power efficient communications techniques enable a battery replacement interval of five years to be achieved.

Several fault-tolerant communications networks have been presented and the flexibility of the 2×2 node in such applications has been emphasized. A prototype 2×2 node circuit was designed and built to operate continuously (in a loop of up to 32 nodes) for five years from a standard D-type lithium cell. This was achieved by integrating the communications interface circuit, part of the original bread-boarded node circuit, on to a CMOS gate array chip using the HIL0-2 digital simulator. To simplify chip testing, and hence reduce manufacturing and maintenance costs, a self-test facility was incorporated into the design which provides 86% fault coverage using only three test vectors. A higher fault coverage is possible by testing the receiver/transmitter pins of the chip, but this requires additional digital test equipment. This reduction in component count, in conjunction with a low mark/space ratio 100kbit/s
PPC communications scheme, resulted in an estimated average current drain for the node circuit of approximately 200μA at 3.5V, which could be provided over the prescribed lifetime and temperature range by a D-type LiSOCl₂ cell. Circuit integration, whilst increasing the node reliability, also reduces the node size, production costs and maintenance costs.

The node circuit can be used in a wide variety of fault-tolerant communications networks, the degree of fault tolerance limited mainly by the allowable cabling overhead. The simplest fault-tolerant network is the dual redundant loop which can tolerate single node failures, but the node circuit would require some minor redesign before it could be used in this application. Skip-braid networks offer a good compromise between fault tolerance and cabling overhead, and can be implemented using the node circuit. The Single Skip-Braid Network (SSBN), requiring a cabling overhead of 200%, is the simplest skip-braid network and can tolerate single node failures and double link failures. Double node fault tolerance can be achieved using several different topologies, the simplest being the (3:1) network with a cabling overhead of 300%. The Double Skip-Braid Network (DSBN), requiring a cabling overhead of 400%, was eventually selected because it satisfied the double fault tolerance criterion while offering high invulnerability to link damage, an important feature considering the target application was unspecified. This choice was also significant because much of the work presented here can be applied easily to the SSBN, which should have numerous applications in less critical environments.

Since a potential five year battery replacement interval for the node circuit operating in single loops of up to 32 nodes, has been demonstrated, the node circuit could also be used satisfactorily in the (3:1) and (4:1) networks consisting of up to 64 nodes.

Procedures governing the main aspects of network control for the DSBN, such as system power-up, fault detection and location, network reconfiguration, message polling and network repair, were presented, and their applicability to other skip-braid networks was discussed. The development task was simplified using a block representation of
the DSBN consisting of a set of 3×3 Boolean connection matrices whose product yields the network transfer function (ie. relating the network outputs to the inputs).

Fault detection is achieved using a time-out mechanism installed at each remote node and the loop controller, and fault location is performed by means of a network scan routine in approximately 200ms. A network reconfiguration algorithm then processes faults individually, searching for fault bypass routes only in the block containing the fault. The procedure uses a fault queuing structure and a simple set of block reconfiguration rules specific to the network topology. Having produced the desired network reconfiguration to bypass a fault condition, the polling sequence generator determines the new polling order for the remaining operational nodes. The network polling period was reduced by minimizing the number of configuration messages sent out during each network poll. Two procedures for supervising network repairs have also been suggested.

Great effort has been expended to reduce the number and complexity of the procedures required for network control, with the intention that increased reliability and adaptability to other topologies would result. Since the network scan routine, which features prominently in the whole network control programme, and the reconfiguration algorithm were both designed to operate on a block-by-block basis, they can be adapted to suit other topologies simply by redefining the block representation of the network (ie. using a different set of connection matrices and block reconfiguration rules). The fault queuing and fault processing structures would remain unchanged. The polling sequence generator could be also be adapted for other topologies.

The main network control procedures were implemented on an IBM PC using polyForth. Based on the assumption that the Novix NC4000 processor could be used in the loop controller, it was estimated that the network control procedures described here could locate and bypass a single fault in a time comparable to the normal network polling period (250ms). The possibility of other software implementations and loop controller architectures has been noted.
A quantitative analysis of how system reliability (in terms of system MTBF and availability) is affected by the reliabilities of the remote nodes, loop controller and fibre-optic data links, has been presented. Where possible, component failure rates were estimated according to MIL-Handbook 217D. The MTBF of the system when subjected to different failure modes was estimated, with the target MTBF specified at 80,000 hours.

When using a skip-braid network based on the 2×2 node circuit without remote instrumentation redundancy, no improvement in MTBF is possible beyond the SSBN because system failure is commonly defined as the loss of two control variables (i.e. two nodes). Based on the estimated node failure rate of 245 per million hours, and assuming that node failure is the dominant system failure mode, it was suggested that only networks containing less than 6 nodes should be used without instrumentation redundancy. However, if the actual node failure rate was about 40 failures per million hours (i.e. if the estimate is very pessimistic, or if better quality components or redundant node electronics are employed), the SSBN could be used for networks of up to 32 nodes without instrumentation redundancy. Redundant node electronics appears an attractive method of improving system MTBF since it avoids the additional cabling costs of a more complex network, but it is unlikely that an estimated failure rate of 40 per million hours could be achieved. Partial redundancy would require too many switching elements (themselves unreliable) for the large number of internal node connections and TMR of the node electronics would consume too much power. Nonetheless, with a larger power budget (i.e. a larger battery) redundant node electronics may be feasible.

Based on a loop controller failure rate estimate of 2460 per million hours, it was suggested that TMR could not be used to achieve the controller MTBF target of 80,000 hours. Moreover, to reduce system vulnerability, the controller MTBF should greatly exceed that target. This would be possible using 1-out-of-3 standby redundancy with the appropriate fault diagnostic capabilities.

Although the pessimistic nature of the estimation procedure was stressed, it was suggested that remote instrumentation redundancy could be used to achieve the target MTBF if node failure is the
dominant system failure mode. Realizing that such redundancy can be used until link failure becomes the dominant system failure mode enabled a quantitative analysis of the effects of link failure rate on system availability. A comparative study was made for the single loop, the SSBN and the DSBN knowing that the results were applicable to any skip-braid network, for example the (3:1) and (4:1) networks, of similar cabling overhead. Results suggest that the availability of the single loop is unacceptably low even for small networks. The benefits of incorporating link redundancy were graphically illustrated with the DSBN achieving the target MTBF for a link failure rate of 6000 failures per million hours and the SSBN for a link failure rate of 1000 per million hours, for a 32-node network.

The importance of an efficient maintenance programme and general fail-safe design techniques, such as simple current limiting, in improving system reliability was also highlighted.

7.1 Distinguishing Features of this System

Compared to conventional 4-20mA systems currently in use, the main distinguishing feature of this system are:

- Fibre-optic data links using off-the-shelf optical components reduces EMI and RFI.
- Digital telemetry incorporating forward error correction lowers the bit error rate.
- Battery-powered remote devices, with a five year replacement interval, avoids the cost of galvanic isolation.
- Provision for a roving hand-held communicator to gain access to system information from any node in the communications network.
- Lower cabling costs for networks containing more than about 10 devices, with a choice of loop topologies offering varying degrees of fault tolerance.
- Different sizes of network can be accommodated, with the option of adding or deleting devices.
- Rapid fault detection and system recovery for up to 2 node failures or 4 link failures with the example DSBN topology.
- Low remote node request-controller response latency.
Reduced production and maintenance costs resulting from the use of circuit integration and self-test techniques. Fault-tolerant system is suited to a programme of periodic maintenance.

7.2 Proposals for Future Work

To allay any fears about lithium battery safety, and to improve the applicability of the system, powering the remote devices over a low power communications network using a copper medium (eg. twisted pair) should be investigated. Power-by-light is a future possibility since the node circuit power consumption is low, but it is likely that a star network would be required to supply the necessary power. Node circuit power consumption could be reduced by optimizing the components used in the fibre-optic data links, but this would most probably raise the cost of the node. The integration of the necessary node microcomputer functions, perhaps as a Reduced Instruction Set Computer (RISC) architecture, with the communications interface circuit would also be beneficial in terms of power consumption, reliability and cost.

Cabling costs could be minimized for single node fault tolerance if the node circuit was redesigned to accommodate a dual redundant loop topology. Node flexibility could be improved further by implementing bidirectional optical couplers, but again this would involve great additional cost. Double node fault tolerance can be achieved with a variety of network topologies. However, since the (3:1) network requires the lowest cabling overhead and is well suited for TMR installations, the network control procedures described here should be extended for the (3:1) network, and perhaps the (4:1) network, considering the pressures on industry to reduce cabling expenditure. Methods for reducing the node request-controller response latency in skip-braid networks should also be addressed.

Alternative loop controller architectures should be explored to improve the applicability of the system and in case the Novix NC4000 does not gain acceptance in industry. When such information becomes available, a prototype loop controller could be built.
As stated in Chapter 1, this work forms part of a larger project investigating fault-tolerant distributed measurement systems. Before these systems can become commercially competitive, the technology required for battery-powered sensors and actuators suitable for process control, must be investigated. Such work is currently under consideration in the Department of Electrical Engineering at Edinburgh University.

Finally, as with any newly developed system, a study of alternative applications should be undertaken.
REFERENCES


48) Delta Controls Ltd., Deltrol Works, Island Farm Av., East Molesey, Surrey, KT8 OUB.

49) Herga Electronics Ltd., Northern Way, Bury St.Edmunds, Suffolk.


57) York VSOP Ltd., York House, School Lane, Chandlers Ford, Hants, SO5 3DG.

58) Pilkington Security Ltd., Colomendy Ind. Estate, Rhyl Road, Denbigh, Clwyd, North Wales, LL16 5TA.


76) TAYLER, C., "Where do you put the computer in today's control system?", Process Engineering, December 1986, pp. 29-32.


87) Foccom Systems Ltd., Hunslet trading estate, Severn Road, Leeds, LS10 1BL.


123) RESNICK, D.R., "Testability and maintainability with a new 6k gate array", VLSI design, March/April 1983.


128) PARRY, S., "VIPER strikes at faulty design", Electronic Times, 5th December 1985, pp.24-25.


148) NoHalt Computers (UK) Ltd., 64 Woodrow, London, SE1 8 5DH.


150) WENSLEY, J.H., "Redundant modules may be best for control systems", Computer Design, April 1985, pp.126-137.

151) Personal communications with Taylor Instrument, Stevenage, Kent.


155) Full Duplex Coupler, Model CAF, ADC Fiber Optics Corp., 8 WalkUp Drive, Westborough, MA, 01581, USA.


160) F.W.O. Bauch Ltd., 49 Theobald St., Boreham Wood, Herts., WD6 4RZ.

161) All Batteries, Tolpits Industrial Estate, Tolpits Lane, Watford, Herts., WD1 8QY.


166) HILO-2, GenRad Cirrus, 285 Bowers Av., Santa Clara, CA 95051.


APPENDIX 1 - COMMUNICATIONS INTERFACE CHIP SPECIFICATIONS

a) Design implementation - Marconi MA2024 3 micron CMOS double level metal gate array.

b) Package size - 68-pin ceramic leadless chip carrier, approximately 5.8cm$^2$.

c) PC board area replaced - approximately 530cm$^2$.

d) Gate utilization - two circuits each of 846 gates, 1692 gates out of 2436 available (70%).

e) Supply voltage - 3 to 10 volts.

HILO 2 net-list

The communications interface circuit was designed using the HILO-2 digital simulator and the 2000A cell library provided by:

Marconi Electronic Devices Ltd. (MEDL),
I.C. Division,
Doddington Road,
Lincoln, LN6 3LF.

Based on this cell library, a new block library was constructed which described the communications interface circuit. The following HILO-2 listing of the block library was then submitted to MEDL for fabrication. Figure A1.1 follows the net-list and shows the metallization layout of the communications interface gate array.
1. **CCT ADDCMP (ADDR[1:6], O14B, YB[1:6], DAT[1:6], EQS)**

**TYPE:** 6-BIT ADDR.COMP., YB[1]=LSB SYND., DAT[1]=MOST SHIFTED BIT.

**EXNOR**
\[ ACA[1:6](YB[1:6], DAT[1:6], ACW[1:6]) \]

**EXORN**
\[ ACB[7:12](ACW[1:6], ADDR[1:6], ACW[7:12]) \]

**NOR3**
\[ AC13(ACW[7,8,9], ACW[13]) \]
\[ AC14(ACW[10,11,12], ACW[14]) \]

**NAND2**
\[ AC15(ACW[13,14], ACW[15]) \]

**NOR2**
\[ AC16(ACW[15], O14B, EQS) \]

UNID EQS ACW[1:15] ;

****

2. **CCT AND21NOR (A, B, C, F)**

**1 - 2 INPUT AND NOR**

**AND2**
\[ G1(B,C,,W1) \]

**NOR2**
\[ G2(W1,A,F) \]

INPUT A B C ;
UNID W1 F ;

****

3. **CCT AND31NOR (A, B, C, D, F)**

**1 - 3 INPUT AND NOR**

**AND3**
\[ G1(B,C,D,,W1) \]

**NOR2**
\[ G2(A,W1,F) \]

INPUT A B C D ;
UNID W1 F ;

****

4. **CCT CONTROL (025T, T8, CF, ENOPT9, AUTOPB, O32B, CLK, IRF, ENIRF, ENIRQ, FR, FRB, O32E, CFB, INTB, ENSH, ENSHB)**

**TYPE:** RESET & CONTROL CCT FOR NODE INTERFACE, TYP. TIMES @ 25C, 5V.

**AND21NOR**
\[ C1(CF,T8,025,MRB) \]

**ANDNOR**
\[ C9(IRF,ENIRF,IRQ,ENIRQ,INTB) \]

**NOR2**
\[ C2(CF,O32E,CW1) \]
\[ C6(ENSH,AUTOPB,IRQ) \]

**RDT**
\[ C4(1,CW3,MRB,ENSHB,ENSH) \]
\[ C5(1,CLK,O32B,O32E,O32EB) \]

**RDL**
\[ C3(1,ENOPT9,FRB,CW2,CW3) \]

**INVB**
\[ C7(CW1,FR) \]
INPUT 025 T8 CF O32B ENOPT9 AUTOPB CLK IRF ENIRF ENIRQ ;
UNID  CFB FR FRB O32E O32EB IRQ ENSH ENSHB MRB CW1 CW2 CW3 INTB ;

*****}

*****}

*****{

5) CCT FLAG (CF, ERR, FERR, O14T2, O24T2, REB, EQS, BE14, BE24, ADDROK, MA, FERR24)

** TYPE : FLAGS FROM INTERFACE TO MICRO-C.

RDL
F1(ERR, O14T2, CFB,, BE14)
F2(FERR, O14T2, CFB,, FW1)
F3(FERR, O24T2, CFB, FW2,)
F4(ERR, O24T2, CFB,, BE24)
F5(EQS, O14T2, CFB, FW3,)
F6(ADDROK, O24T2, CFB,, MA);

NAND2
F7(FW2, REB, FERR24);

NOR2
F8(FW1, FW3, ADDROK);

INV
F9(CF, CFB);

INPUT CF ERR FERR O14T2 O24T2 REB EQS ;
UNID CFBE14 BE24 FERR24 ADDROK MA FW1 FW2 FW3 ;

*****}

*****{

6) CCT JCNT (OPDAT, FR, TRAB, CLK, QFH, Q[0:9])

** TYPE : 10-STAGE JOHNSTON COUNTER WITH OPT.DAT.A-BAR I/P.

SRDL
JC1 (1, OPDAT, TRAB, FRB,, JCW[1]);
JC2 (JCW[1], CLK, FRB, JCW[2,]);
JC3 (JCW[12], JCW[13], FRB, JCW[8], JCW[3]);
JC4:7 (JCW[3:6], JCW[13], FRB, JCW[9:12], JCW[4:7]);

NOR2
JC8 (CLK, JCW[2], JCW[13]);
JC9 (JCW[8], JCW[12], Q[5]);
JC10(JCW[8], JCW[4], Q[1]);
JC11(JCW[10], JCW[4], Q[7]);
JC12(JCW[10], JCW[6], Q[3]);
JC13(JCW[12], JCW[6], Q[9]);
JC14(JCW[7], JCW[3], Q[0]);
JC15(JCW[9], JCW[3], Q[6]);
JC16(JCW[9], JCW[5], Q[2]);
JC17(JCW[5], JCW[11], Q[8]);
JC18(JCW[7], JCW[11], Q[4]);

INV
JC19(JCW[7], QFH)
JC20(FR, FRB);

INPUT OPDAT CLK TRAB FR;
UNID JCW[1:13] Q[0:9] QFH FRB ;

*****}

*****{
***

7 CCT NODE (RX1, RX2, MCDAT, MCLK, AUTOPB, TRAB, ENT1, ENT2, CF, ST, CLK, ENSYN, ADDR[1:6], DATMC, BOR1, BOR2, BE14, BE24, MA, INTB, IRF, TX1, TX2, CLP, DATNOD)

** COMPLETE NODE INTERFACE CIRCUIT **

RECV N1(RX1, RX2, T[0], T[1], T[2], T[4], T[9], TFH, TDAT, ST, TRAB, FR, CFB, BOR1, BOR2, CLP, OPDAT, IRF, OP0, OP1, ENOP, ENOPB, ENOPT9, BYDATB)

ADDCMP N6(ADDR[1:6], O14B, YB[1:6], DAT[1:6], EQS);

CONTROL N10(025, T[8], CF, ENOPT9, AUTOPB, Q[12], CLK, IRF, ENIRF, ENIRQ, FR, FRB, O32E, CFB, INTB, ENSH, ENSHB);

FLAG N7(CF, ERR, FERR, O14T2, O24T2, REB, EQS, BE14, BE24, ADDROK, MA, FERR24);

JCNT N2(OPDAT, FR, TRAB, CLK, TFH, T[0:9]);

OPTCNT N3(OPDAT, FR, CFB, O24, RE, RE);

SR41B N8(OP1, T[0], ENSH, MA, ENOPT, ENNOD, MCDAT, AUTOPB, MCLKD, DATNOD, DATOPT, DAT[1:6], SHP);

SYNGEN N5(OP1, Q[7:11], SHP, RES14, ENSYN, YB[1:6], HB[1:5], ERR, FERR);

SYSCNT N4(ENOPT9, T[2], T[7], FRB, Q[1:12], RES14, O14B, O14T2, O24T2, O2L, O24, O25);

TRANS N11(APPROK, TRAB, DATNOD, DATOPT, T[0], T[2], T[7], O32E, ENOP, ENSHB, MA, ENT1, ENT2, FRB, ST, BYDATB, O2L, Q[2], TX1, TX2, ENOPT, ENNOD, TDAT, STB, ENIRF, ENIRQ);

USHREG N9(DATNOD, HB[1:5], BE24, RE, FERR24, O24, T[6], ENSHB, MCLK, DATMC, MCLKD);

INPUT RX1 RX2 MCDAT MCLK AUTOPB TRAB ENT1 ENT2 CF ST CLK ENSYN ADDR[1:6];


***

*****

8 CCT OPTCNT (CLK, FR, CFB, EN, REB, RE)

** TYPE : OPTIC COUNTER WITH 40OP=24SYS OUTPUT **

RTT OC1(CLK, FRB, OCW[7], OCW[1])

OC[2:6](OCW[7:11], FRB, OCW[8:12], OCW[2:6]);

NAND2 OC7(OCW[4], OCW[6], OCW[13]);

RDT OC8(OCW[13], EN, CFB, REB, RE);

INV OC9(FR, FRB);

INPUT CLK FR CFB EN;


****

****
9 CCT OR21NAND (A,B,C,F)

** 1 - 2 INPUT OR NAND

OR2 \( G1(B,C,W1) \);
NAND2 \( G2(A,W1,F) \);

INPUT A B C;
UNID W1 F ;

****

****

10 CCT PESR26 (DATS,DATP,CONT,CLK,Q[1:26])

** 26 BIT DUAL I/P SR

ANDOR \( G1(DATP,CONT,DATS,CONTB,,W1) \);
INV \( G2(CONT,CONTB) \);
SHR7 \( G3(CLK,W1,,Q[1:7]) \);
SHR6 \( G4(CLK,Q[7],,Q[8:14]) \);
SHR6 \( G5(CLK,Q[14],,Q[15:20]) \);
SHR6 \( G6(CLK,Q[20],,Q[21:26]) \);

INPUT DATS DATP CONT CLK;
UNID CONTB W1 Q[1:26] ;

****

****

11 CCT PSR15 (DAT,CLK,Q[1:15])

** 15 BIT +VE CLK SR SISO

SHR8 \( G1(CLK,DAT,,Q[1:8]) \);
SHR7 \( G2(CLK,Q[8],,Q[9:15]) \);

INPUT CLK DAT;
UNID Q[1:15] ;

****

****

12 CCT RECV (RX1,RX2,T0,T1,T2,T4,T9,TFH,TDAT,ST,TRAB,FR,CFB,
BOR1,BOR2,CLP,OPDAT,IRF,OP0,OP1,ENOP,ENOPB,ENOPT9,BYDATB)

** TYPE : INTERFACE RECEIVER WITH NODE CONTROL SIGNALS,TYP.TIMES @25C, 5V.

NOR2 \( R1(RX1,RX2,BYDATB) \);
R4(TRAB,ENOP,RW[22])
R21(ST,BYDATB,RW[16])
R33(ENOP,RW[25],IRF)
R17(RW[13],FR,RW[14]) ;
OR2 \( R2(T1,T9,,RW[1]) \);
INV R35(TRA, TRAB)
  R19(T9, RW[15])
INVB R34(OPDATB, OPDAT)
  R36(FR, FRB)
NAND2 R7(ENOPB, RW[1, 2, RW[4])
  R15(W10, W[11, W[12])
  R27(T0, RW[22, RW[17])
AND2 R20(RW[20], T9, ENOPT9)
NAND3 R12(RW[4], RW[5), RW[6), CLP)
  R6(OP0, T9, ENOP, RW[5])
  R10(RW[7], RW[13], RW[2], RW[8])
NAND4 R9(OP1, T4, RW[3), ENOP, RW[6])
AND21NOR R23(RW[16], RW[18], RW[21], OPDATB)
OR21NAND R11(RW[12], RW[2], RW[19], RW[9])
  R22(RW[17], ENOPB, TDAT, RW[18])
RDL R25(1, RX1, CFB, BOR1)
  R26(1, RX2, CFB, BOR2)
  R5(1, OPDAT, RW[15], RW[3])
  R29(1, TRA, FRB, RW[21])
SDT R16(RW[12], T9, FRB, RW[13])
  R13(RW[8], OPDAT, FRB, RW[10])
RDT R18(1, RW[15], RW[14], ENOPB, ENOP)
  R30(1, RW[15], FRB, RW[23])
  R31(RW[23], RW[15], FRB, RW[24])
  R32(RW[24], RW[15], FRB, RW[25])
  R28(ENOP, T9, RW[14], RW[20])
  R14(RW[9], OPDAT, FRB, RW[11])
RTT R3(RW[1], FRB, RW[2])
DT R24(TFH, OPDAT, OP1)

INPUT RX1 RX2 T0 T1 T2 T4 T9 TFH FR CFB TDAT ST TRAB
UNID BOR1 BOR2 CLP OPDATB IRF

****

****

13 CCT SR41B (OP1, T0, ENSH, MA, ENOPT, ENNOD, MCDAT, AUTOPB,
  MCLKD, DATNOD, DATOPT, DAT[1:6], SHP)

** TYPE : 41-BIT SR ARRAY WITH INPUT CONTROL, TYP. TIMES @ 25C, 5V.

ANDOR SR2(MCDAT, AUTOPB, OP1, SRW1, SRW2)
  SR3(T0, ENSH, SRW3)
NAND2 SR5(SRW3, SRW4, SHP)
OR21NAND SR4(MCLKD, MA, AUTOPB, SRW4)
INV SR6(SHP, SRW6)
  SR1(AUTOPB, SRW1)
  SR11(SRW17, SRW18)
NOR2 SR9(SRW3, ENOPT, SRW16)
  SR10(SRW16, MCLKD, SRW17)
NOR3 SR12(MA, ENNOD, AUTOPB, SRW20)
PSR15 SR7(SRW2, SRW6, DATOPT, DAT[6, 5, 4], DAT[3, 2, 1], SRW15)
PESR26 SR13(SRW15, MCDAT, SRW20, SRW18, DATNOD)

INPUT OP1 T0 ENSH MA ENOPT ENNOD MCDAT AUTOPB MCLKD
UNID DATNOD DATOPT DAT[1:6] SRW1 SRW2 SRW3 SRW4 SHP SRW15 SRW16
14CCT SYNGEN (DAT, CNT[0:4], CLK, RES, ENSYN, YB[1:6], SGW[8:12], ERR, FERR)

** TYPE : SYND.GEN.WITH O/P'S 5-7 & 9-11 , TYP.TIMES @25C, 5V.

NAND3
SG1 (DAT, CLK, ENSYN, SGW[22])
SG19 (SGW[16, 17, 18], SGW[19])
NAND2
SG20 (SGW[14, 15], SGW[20])
NOR2
SGA[2:6] (CNT[0:4], SGW[22], SGW[2:6])
SG21 (SGW[19, 20], SGW[21])
SG22 (SGW[13], SGW[21], ERR)
INV
SG24 (SGW[22], SGW[1])
SG25 (RES, RB)
RTT
SGB[7:12] (SGW[1:6], RB, SGW[13:18], SGW[7:12])

NAND4
SG13 (SGW[17, 10, 15, 8], YB[1])
SG14 (SGW[17, 10, 9, 14], YB[2])
SG15 (SGW[17, 10, 9, 8], YB[3])
SG16 (SGW[11, 16, 15, 8], YB[4])
SG17 (SGW[11, 16, 9, 14], YB[5])
SG18 (SGW[11, 16, 9, 8], YB[6])
EXORN
SG23 (SGW[13], SGW[21], FERR)

INPUT DAT CNT[0:4] CLK RES ENSYN

15CCT SYS_CNT (CLK, T2, T7, FRB, Q[1:12], RES14, O14B, O14T2, O24T2, O2L, O24, O25)

** TYPE : SYSTEM COUNTER WITH NODE CONTROL OUTPUTS

RTT
SC1 (CLK, RES14B, Q[7], Q[1])
SC[2:6] (Q[7:11], RES14B, Q[8:12], Q[2:6])
RD
SC7 (1, O14B, FRB, O14F)
SC8 (Q[2, 3, 4], O14)
SC11 (O14, T2, O14F, O14T2)
AND31
SC9 (FR, T7, O14, O14F, SCW1)
SC10 (O14, O14B)
SC14 (SCW1, RES14)
SC20 (RES14, RES14B)
NAND3
SC15 (FRB, FR)
SC18 (Q[7, 8, 9], SCW2)
SC19 (Q[4, 5, 12], SCW3)
AND2
SC13 (O24, T2, O24T2)
SC16 (O14F, Q[8], O2L)
NOR2
SC21 (SCW2, SCW3, O24)
NOR3
SC17 (Q[7], Q[10, 11], O25)
** TYPE : TRANSMITTER CONTROL & START-BIT GENERATOR, TYP. TIMES @25C, 5V.

** TYPE : UNIVERSAL SHIFT REG. STAGE (8-BITS), TYP. TIMES @25C, 5V.
18 CCT USR8 (DATS, DATP[1:8], CONT, CLK, Q[1:8])

** 8 BIT SISO/PARALLEL LOAD SERIAL OUT SR

D2T

G1(DATS, DATP[1], CONT, CLK, , Q[1])
G[2:8](Q[1:7], DATP[2:8], CONT, CLK, , Q[2:8])

INPUT DATS DATP[1:8] CONT CLK
UNID Q[1:8] ; .

****}
Figure A1.1. Metallization layout of the communications interface gate array
A) Single Skip-Braid Network (SSBN)

Since the SSBN can tolerate any combination of link failures except for three or more adjacent links, only link failures that contribute to a disconnecting group of three links need to be considered when calculating the MTBF. The network states required to be analysed are:

- state 0) all links are operational
- state 1) any one link has failed, the other links in that disconnecting group are operational
- state 2) any two links in the disconnecting group have failed, the other link in that group is operational
- state 3) the network is disconnected.

Now assume that the network is in state 0 at time 0 and let $P_i(t)$ be the probability of the network being in state $i$ at time $t$, therefore:

$$P_0(0) = 1, P_1(0) = P_2(0) = P_3(0) = 0$$

and

$$P_0(t) + P_1(t) + P_2(t) + P_3(t) = 1 \quad (A2.1)$$

For a network containing $e$ links with a probability of link failure $x$ and a link MTTR of 8 hours ($\mu=0.125$), the transition diagram for the system is:

![Transition Diagram](image)

Therefore, provided $\Delta t$ is so small that only one failure or repair event can occur during $\Delta t$, the probability of a specific link failure during $\Delta t$ is simply $x \Delta t$ and the probability of any link failure out of $e$ links is $e x \Delta t$. Consequently, the probability of no link failures during $\Delta t$ is $(1 - e x \Delta t)$. The probability of a failed link being repaired during $\Delta t$ is $\mu \Delta t$ and so:
\[ P(t+\Delta t) = [P(t).(1-ex^{\Delta t})] + [P(t).(1-2x\Delta t).\mu \Delta t] \]  
(A2.2)

\[ P'(t+\Delta t) = [P(t).ex^{\Delta t} + [P(t).(1-2x\Delta t).(1-\mu \Delta t)]  
+ [P(t).(1-x\Delta t).2\mu \Delta t] \] (A2.3)

\[ P(t+\Delta t) = [P(t).2x\Delta t] + [P(t).(1-x\Delta t).(1-2\mu \Delta t)] \] (A2.4)

\[ P(t) = [P(t).x\Delta t] + P(t) \] (A2.5)

Since the limit as \( \Delta t \to 0 \) of \( [P(t+\Delta t)-P(t)]/\Delta t \) is \( P(t) \), the above equations yield:

\[ \dot{P}(t) = -ex.P(t) + \mu.P(t) \] (A2.6)

\[ \dot{P}(t) = ex.P(t) - (2x+\mu).P(t) + 2\mu.P(t) \] (A2.7)

\[ \dot{P}(t) = 2x.P(t) - (2\mu-x).P(t) \] (A2.8)

\[ \dot{P}(t) = x.P(t) \] (A2.9)

In matrix notation this becomes:

\[
\begin{bmatrix}
  \dot{P}(t) \\
  \dot{P}(t) \\
  \dot{P}(t) \\
  \dot{P}(t)
\end{bmatrix} =
\begin{bmatrix}
  -ex & \mu & 0 & 0 \\
  ex & -(2x+\mu) & 2\mu & 0 \\
  0 & 2x & -(2\mu-x) & 0 \\
  0 & 0 & x & 0
\end{bmatrix}
\begin{bmatrix}
  P(t) \\
  P(t) \\
  P(t) \\
  P(t)
\end{bmatrix}
\] (A2.10)

The MTBF of a system \( \theta \) is defined as:

\[
\theta = \int_{0}^{\infty} R(t)dt
= \int_{0}^{\infty} [P(t) + P(t) + P(t)]dt
= \int_{0}^{\infty} P(t)dt + \int_{0}^{\infty} P(t)dt + \int_{0}^{\infty} P(t)dt
= T_0 + T_1 + T_2
\] (A2.11)

where \( R(t) \) is the probability of successful operation during the time \( t \). Now \( T_0, T_1 \) and \( T_2 \) can be found by solving:

\[
\begin{bmatrix}
  \dot{P}(t) \\
  \dot{P}(t) \\
  \dot{P}(t) \\
  \dot{P}(t)
\end{bmatrix} =
\begin{bmatrix}
  -ex & \mu & 0 & 0 \\
  ex & -(2x+\mu) & 2\mu & 0 \\
  0 & 2x & -(2\mu-x) & 0 \\
  0 & 0 & x & 0
\end{bmatrix}
\begin{bmatrix}
  P(t) \\
  P(t) \\
  P(t) \\
  P(t)
\end{bmatrix}
\] (A2.12)
assuming that the failure and repair rates are constant. Therefore:

\[
\begin{pmatrix}
-\exp\left(-2x+\mu\right) & 2\mu & 0 \\
\exp\left(-2x+\mu\right) & -2x & -(2\mu+x) \\
0 & 2x & -x
\end{pmatrix}
\begin{pmatrix}
P_0 \left(0\right) \\
P_1 \left(0\right) \\
P_2 \left(0\right)
\end{pmatrix}
=\begin{pmatrix}
T_0 \\
T_1 \\
T_2
\end{pmatrix}
\] (A2.13)

and knowing that \( P \left(0\right) = 1 \), \( P_1 \left(0\right) = P_2 \left(0\right) = P_3 \left(0\right) = 0 \), \( P \left(\infty\right) = P_1 \left(\infty\right) = P_2 \left(\infty\right) = 0 \), \( P_3 \left(\infty\right) = 1 \), the above matrix set reduces to:

\[
\begin{pmatrix}
-1 \\
0 \\
1
\end{pmatrix}
=\begin{pmatrix}
-\exp\left(-2x+\mu\right) & 2\mu & 0 \\
\exp\left(-2x+\mu\right) & -2x & -(2\mu+x) \\
0 & 2x & -x
\end{pmatrix}
\begin{pmatrix}
T_0 \\
T_1 \\
T_2
\end{pmatrix}
\] (A2.14)

which produces:

\[
T_0 = \frac{1}{\exp} + \frac{\mu \left(2\mu+x\right)}{2\exp^3}, \quad T_1 = \frac{(2\mu+x)}{2x^2}, \quad T_2 = \frac{1}{x}
\]

\[
\text{MTBF}_{\text{SBN}} = \frac{x^2 \left(3e+2\right) + x\mu \left(2e+1\right) + 2\mu^2}{2\exp^3}
\] (A2.15)

This equation defines the MTBF for the single skip-braid network where system failure results purely from communications link failure.
B) Double Skip-Braid Network (DSBN)

Since the DSBN can tolerate any combination of link failures except for five or more adjacent links, only link failures that contribute to a disconnecting group of five links need to be considered when calculating the MTBF. The network states required to be analysed are:

state 0) all links are operational
state 1) any one link has failed, the other four links in that disconnecting group are operational
state 2) any two links in the disconnecting group have failed, the other three links in that group are operational
state 3) any three links in the disconnecting group have failed, the other two links in that group are operational
state 4) any four links in the disconnecting group have failed, the other link in that group is operational
state 5) the network is disconnected.

Now assume that the network is in state 0 at time 0 and let \( P_i(t) \) be the probability of the network being in state \( i \) at time \( t \), therefore:

\[
P_0(0) = 1, \quad P_1(0) = P_2(0) = P_3(0) = P_4(0) = P_5(0) = 0
\]

and

\[
P_0(t) + P_1(t) + P_2(t) + P_3(t) + P_4(t) + P_5(t) = 1 \tag{A2.16}
\]

For a network containing \( e \) links with a probability of link failure \( x \) and a link MTTR of 8 hours (\( \mu = 0.125 \)), the transition diagram for the system is:

![Transition Diagram](attachment:image.png)

Therefore, provided \( \Delta t \) is so small that only one failure or repair event can occur during \( \Delta t \), the probability of a specific link failure during \( \Delta t \) is simply \( x \Delta t \) and the probability of any link failure out of \( e \) links is \( e x \Delta t \). Consequently, the probability of no link failures
during $\Delta t$ is $(1 - e^{-\Delta t})$. The probability of a failed link being repaired during $\Delta t$ is $\mu \Delta t$ and so:

$$P(t+\Delta t) = [P(t).(1-\Delta t)] + [P(t).(1-4\Delta t).\mu \Delta t] \quad (A2.17)$$

$$P_1(t+\Delta t) = [P(t).e^{\mu \Delta t}] + [P(t).(1-4\Delta t).(1-\mu \Delta t)]$$
$$+ [P(t).(1-3\Delta t).2\mu \Delta t] \quad (A2.18)$$

$$P_2(t+\Delta t) = [P(t).4\Delta t] + [P(t).(1-3\Delta t).(1-2\mu \Delta t)]$$
$$+ [P(t).(1-2\Delta t).3\mu \Delta t] \quad (A2.19)$$

$$P_3(t+\Delta t) = [P(t).3\Delta t] + [P(t).(1-2\Delta t).(1-3\mu \Delta t)]$$
$$+ [P(t).(1-\Delta t).4\mu \Delta t] \quad (A2.20)$$

$$P_4(t+\Delta t) = [P(t).2\Delta t] + [P(t).(1-\Delta t).(1-4\mu \Delta t)] \quad (A2.21)$$

$$P_5(t+\Delta t) = [P(t).\Delta t] + P(t) \quad (A2.22)$$

Since the limit as $\Delta t \to 0$ of $[P(t+\Delta t)-P(t)]/\Delta t$ is $\dot{P}(t)$, the above equations yield:

$$\dot{P}_0(t) = -e^{\mu}P(t) + \mu P(t) \quad (A2.23)$$

$$\dot{P}_1(t) = e^{\mu}P(t) - (\mu+4x)P_1(t) + 2\mu P_2(t) \quad (A2.24)$$

$$\dot{P}_2(t) = 4xP_2(t) - (2\mu+3x)P_3(t) + 3\mu P_4(t) \quad (A2.25)$$

$$\dot{P}_3(t) = 3xP_2(t) - (3\mu+2x)P_3(t) + 4\mu P_4(t) \quad (A2.26)$$

$$\dot{P}_4(t) = 2xP_2(t) - (4\mu+x)P_4(t) \quad (A2.27)$$

$$\dot{P}_5(t) = xP_4(t) \quad (A2.28)$$

In matrix notation this becomes:

$$\begin{bmatrix}
\dot{P}_0(t) \\
\dot{P}_1(t) \\
\dot{P}_2(t) \\
\dot{P}_3(t) \\
\dot{P}_4(t) \\
\dot{P}_5(t)
\end{bmatrix} =
\begin{bmatrix}
-e^{\mu} & 0 & 0 & 0 & 0 \\
\mu & -e^{(\mu+4x)} & 2\mu & 0 & 0 \\
0 & 4x & -(2\mu+3x) & 3\mu & 0 \\
0 & 0 & 3x & -(3\mu+2x) & 4\mu \\
0 & 0 & 0 & 2x & -(4\mu+x) \\
0 & 0 & 0 & 0 & x
\end{bmatrix}
\begin{bmatrix}
P_0(t) \\
P_1(t) \\
P_2(t) \\
P_3(t) \\
P_4(t) \\
P_5(t)
\end{bmatrix} \quad (A2.29)
The MTBF of a system $s$ is defined as:

$$\theta_s = \int \sum R(t) dt$$

$$= \sum \left[ P_0(t) + P_1(t) + \ldots + P_4(t) \right] dt$$

$$= \sum P_0(t) dt + \sum P_1(t) dt + \ldots + \sum P_4(t) dt$$

$$= T_0 + T_1 + T_2 + T_3 + T_4 \quad (A2.30)$$

where $R(t)$ is the probability of successful operation during the time $t$. Now $T_0 \ldots T_4$ can be found by solving $A2.29$ in the usual way knowing that:

$$P_(0) = 1, P_(0) = P_(0) = P_(0) = P_(0) = P_(0) = 0,$$

and


The solution yields:

$$T_0 = \frac{24x + 6uz^3 + 6uz^3 + 24uz^4}{24x},$$

$$T_1 = \frac{6x^3 + 6uz^3}{24x}, \quad T_2 = \frac{2x^2 + 3ux^3 + 12u}{6x^2},$$

$$T_3 = \frac{x + 2u}{2x}, \quad T_4 = \frac{1}{x}.$$

Therefore, the MTBF of the DSBN is given by:

$$\frac{(25e + 12)x^4 + (32e + 3)uz^3 + (27e + 2)uz^3}{12ex} + \frac{2x^2 + (12e + 3)x^3 + 12u}{4} \quad (A2.31)$$

This equation defines the MTBF for the double skip—braid network where system failure results purely from communications link failure.