REMOTE-CONTROL TESTING OF DEDICATED CIRCUITS, USING A MICROCONTROLLER AND A GRAPHICAL USER INTERFACE

RUDOLPH THOMAS

Dissertation submitted in fulfilment of the requirements for the Degree

MAGISTER TECHNOLOGIAE
ENGINEERING: ELECTRICAL

In the

Faculty of Engineering

At the

Technikon Free State

Supervisor: Prof. G.J. Prinsloo Ph.D. (Eng.)

BLOEMFONTEIN
December 1998

Central University of Technology, Free State

DECLARATION OF INDEPENDENT WORK

I, RUDOLPH THOMAS, do hereby declare that this research project submitted for the degree MAGISTER TECHNOLOGIAE: ENGINEERING: ELECTRICAL, is my own independent work that has not been submitted before to any institution by me or anyone else as part of any qualification.

.... 6-4-99 ....

SIGNATURE OF STUDENT DATE

Central University of Technology, Free State

ACKNOWLEDGEMENTS

I thank the following persons for helping and supporting me with the project:

God, for giving me a vision in life to prosper and shine.

My wife, Mariska, for all the computer-aided drawings and for the loss of time spent together.

Telkom SA, for the supplying of funds to build all of the hardware that was required for the project.

My Manager, JD Marais, for his interest in and willingness to promote the project to the higher levels of management.

My colleagues, Willem Coetzee and Danie de Bruyn, for developing some of the initial hardware and software that were needed for the initial testing and feasibility of the project.

My parents, for enabling me to study further and for their support through the years.

My supervisor, Prof. G Prinsloo, for his support and for always being positive.

SUMMARY

Most data communication test equipment today has a remote-control interface, but is never used to benefit the people and companies using such equipment. This is largely due to the fact that the software and hardware needed to control the test equipment, is very expensive. In addition, many times the test systems are not a hundred percent applicable to every company's need. Furthermore, without a remote-control test system, standby personnel have to drive to a centralized test centre to perform tests every time an after-hour fault was logged.

Telkom SA needed a system for Diginet whereby testing could be performed from a remote location – a system that is also cost effective enough to implement on a national basis.

A system consisting of two parts was designed, namely:

- A microcontroller, which is responsible for switching and controlling of the remote test desk.
- A Graphical User Interface, which is responsible for interfacing the remote test desk to the user over a remote distance.

The research is carried out in four phases:

- Hardware: The hardware consists of a microcontroller, a relay board and a logic multiplexer. The hardware replaces one complete local test desk. The controller that is used in the design is an Intel 80C32 that contains an onboard asynchronous

port which makes it possible to communicate to it with an RS232 interface. This interface is connected to a modem or a Cisco® Terminal server that makes it possible to do remote control. The software for the controller was developed with a cross compiler.

- The second phase was to develop a graphical-user interface whereby any person could control the hardware from a remote distance. The software would also be responsible for controlling the remote test equipment. All of the development was done on the Microsoft® Visual C++ compiler, which made it possible to execute the interface on a Windows 3.1® or Windows95® platform.
- After the initial designs of both the hardware and the software phases, tests were performed on actual circuits to see if the system worked. After satisfactory results were obtained, the project was demonstrated at numerous occasions to Telkom management.
- The last phase of this project, which is beyond the scope of this document, will be to deploy the system country-wide within Telkom SA.

Knowledge was gained in the following areas:

- Remote control systems.
- Microcontrollers.
- Programming languages.
- Graphical User Interfaces.

This research project concludes that it is possible to do remote testing in a manner that is cost-effective and of advantage to all telecommunication companies using such a system.

Central University of Technology, Free State

OPSOMMING

Die meeste data kommunikasie toetsapparaat wat vandag beskikbaar is, beskik oor 'n afstandbeheer-koppelvlak wat selde gebruik word deur mense en maatskappye. Dit is grootliks as gevolg van die feit dat die sagteware en hardeware wat nodig is om die toetsapparaat te beheer, baie duur is. Dikwels is die toetsstelsels wat beskikbaar is nie honderd persent aanpasbaar met elke maatskappy se benodigdhede nie. Sonder 'n afstandbeheerde toetsstelsel moet bystandpersoneel elke keer op uitroep geplaas word en na 'n gesentraliseerde toetssentrum ry.

Telkom SA het 'n stelsel benodig vir Diginet waarmee toetsing gedoen kan word vanaf 'n distansie en wat koste-effektief genoeg is om nasionaal te gebruik.

'n Stelsel is ontwerp wat uit twee dele bestaan:

- 'n Mikrobeheerder, wat verantwoordelik is vir skakeling en beheer van die afstandbeheerde toetsbank.
- Grafiese gebruikerskoppelvlak, wat as koppelvlak dien tussen die gebruiker en die afstandbeheerde toetsbank oor 'n distansie.

Die navorsing is in vier fases uitgevoer:

- Hardeware: Die hardeware bestaan uit 'n mikrobeheerder, 'n relê-module en 'n logika-multiplekseerder. Die hardeware vervang een kompleet lokale toetsbank. Die beheerder wat gebruik word in die ontwerp is 'n 80C32, en dit bevat 'n kommunikasiepoort wat deur middel van 'n RS232-koppelvlak met die beheerder kommunikeer. Die RS232-koppelvlak word gekoppel aan 'n modem of 'n Cisco® Terminaal-bediener, wat dit moontlik maak om beheer oor 'n afstand uit te voer. Die sagteware vir die beheerder is ontwikkel met 'n kruiskompileerder.

- In die tweede fase is die grafiese gebruiker-koppelvlak ontwikkel waarmee enige persoon die toetsapparaat oor 'n afstand kan beheer. Die sagteware sou ook verantwoordelik wees om die toetsapparaat oor 'n distansie te beheer deur middel van 'n RS232-koppelvlak. Die ontwikkeling van die sagteware as geheel is gedoen op die Mircosoft® Visual C++ kompileerder. Dit het dit moonlik gemaak dat die sagteware op 'n Windows3.1® sowel as 'n Windows95® platform kan werk.
- Nadat die aanvangsontwerpe van beide die hardeware en die sagteware voltooi is, is toetse uitgevoer om te sien of die stelsel werk. Nadat bevredigende resultate verkry is, is die projek op vele geleenthede gedemonstreer aan Telkom bestuur.
- Die laaste fase van die projek is buite die bestek van hierdie dokument. Dit behels die landwye implementering van die stelsel binne Telkom SA.

Besondere kennis is ingewin ten opsigte van:

- Afstandbeerstelsels.
- Mikrobeheerders.
- Programmeringstale.
- Grafiese gebruikerskoppelvlakke.

Die navorsingsprojek sluit af met die wete dat dit moontlik is om afstandbeheerde toetsing te doen wat koste-effektief is en voordele inhou vir kommunikasie-maatskapye wat sulke stelsels gebruik.

Central University of Technology, Free State

INDEX

Figures

Tables

List of Abbreviations

ACE	-	Automatic Crossconnecting Equipment
ADC	-	Analogue to Digital Converter
AIS	-	Alarm-Indication Signal
ASK	-	Amplitude Shift Keying
ATDT	-	Attention Dial Tone
BER	-	Bit-Error Rate
CCITT	-	See ITU
CD	-	Carrier Detect
CODI	-	CTU Microcontroller
COM	-	PC serial communications port
CPS	-	Call-Progress Signals
CPU	-	Central Processing Unit
CRC	-	Cyclic Redundancy Check
CTS	-	Clear To Send
CTU	-	Co-directional Terminating Unit
DAC	-	Digital to Analogue Converter
DCE	-	Data Circuit terminating Equipment
DCEPO	-	DCE Power Off
DP	-	Distribution Point
DSR	-	Data Set Ready
DTE	-	Data Terminating Equipment
DTR	-	Data Terminal Ready
EIA	-	Electronic Industries Association

FAS	-	Frame Alignment Signal
FB4000	-	Remote tester program
FDM	-	Frequency Division Multiplexing
FRED	-	Remote test desk microcontroller
FSK	-	Frequency Shift Keying
GPIB	-	General Purpose Interface Bus
GUI	-	Graphical User Interface
IA	-	International Alphabet
IDF	-	Intermediate Distribution Frame
IEEE	-	Institute of Electrical and Electronic Engineers
ISDN	-	Integrated Services Digital Network
ISO	-	International Standards Organization
ITU	-	International Telecommunications Union
Kbit/s	-	Kilobits per second
LAN	-	Local Area Network
LOF	-	Loss Of Frame
LOS	-	Loss of Signal
LPT	-	Line Printer Port
MAN	-	Metropolitan Area Network
Mbit/s	-	Megabits per second
MDF	-	Main Distribution Frame
ME	-	Maintenance Entity
MTTR	-	Mean time to restore
MUX	-	Time division multiplexer

NFAS	-	Not Frame Alignment Signal
NTU	-	Network Terminating Unit
NVRAM	-	Non Volatile RAM
OOP	-	Object Oriented Programming
OS	-	Operating System
OSI	-	Open Systems Interconnection
PAD	-	Packet Assembler Disassembler
PC	-	Personal Computer
PCM	-	Pulse-Code Modulation
PDH	-	Plesiochronous Digital Hierarchy
PDN	-	Public Data Network
PLEX	-	Multiplexer switch
POP	-	Point Of Presence
P-P	-	Point to Point
PPP	-	Point to Point Protocol
PROM	-	Programmable Read Only Memory
PSK	-	Phase Shift Keying
PSTN	-	Public Switch Telephone Network
QAM	-	Quadrature Amplitude Modulation
R2PLEX	-	RS232 Switch
RAI	-	Remote Alarm Indication
RAM	-	Random Access Memory
RATS	-	Remote Access and Test System
RCE	-	Remote Control Equipment

RELL	-	Relay board
RENACE	-	Same as RCE
ROM	-	Read-Only Memory
RTC	-	Real-Time Clock
RTD	-	Remote Test Desk
RTS	-	Request To Send
SDH	-	Synchronous Digital Hierarchy
SLIP	-	Serial Line Interface Protocol
TCP/IP	-	Transmission Control Protocol / Internet Protocol
TDM	-	Time-Division Multiplexing
TESS	-	Remote tester setup program
TTC	-	Telecommunications Techniques Corporation
UART	-	Universal Asynchronous Receiver Transmitter
VDU	-	Visual Display Unit
WAN	-	Wide-Area Network

Chapter 1

Problem Theory

1.1 Problem Theory

Clients obtain dedicated circuits from telecommunication service providers to be able to communicate between buildings, cities and countries. Sometimes it happens that faults occur on a circuit because of lightning, wet cables, etc. The client will report a faulty connection (circuit) to be tested. Testing of point-to-point (dedicated) data circuits are done manually. The circuit that is reported is placed on a fault system called Unibase®. The faults are routed via computer to the correct test centre.

Tests can only be done from the regional test centre and cannot be tested from outside the test centre. After-hour faults have to be tested from the test centre, meaning that the person on stand-by will have to drive from his/her home to the test centre to test the faulty circuit. This usually takes a while. Time could have been saved if the person on call-out duty had a laptop and a modem to connect to a special device to perform tasks in the test centre.

During normal working hours, outside faultsmen have to phone the test centre to test faulty circuits, often having to try to get through on a busy phone line. This same type of special device could be implemented for day-testing as well. This puts less stress on the test centre and improves the time spend on each fault.

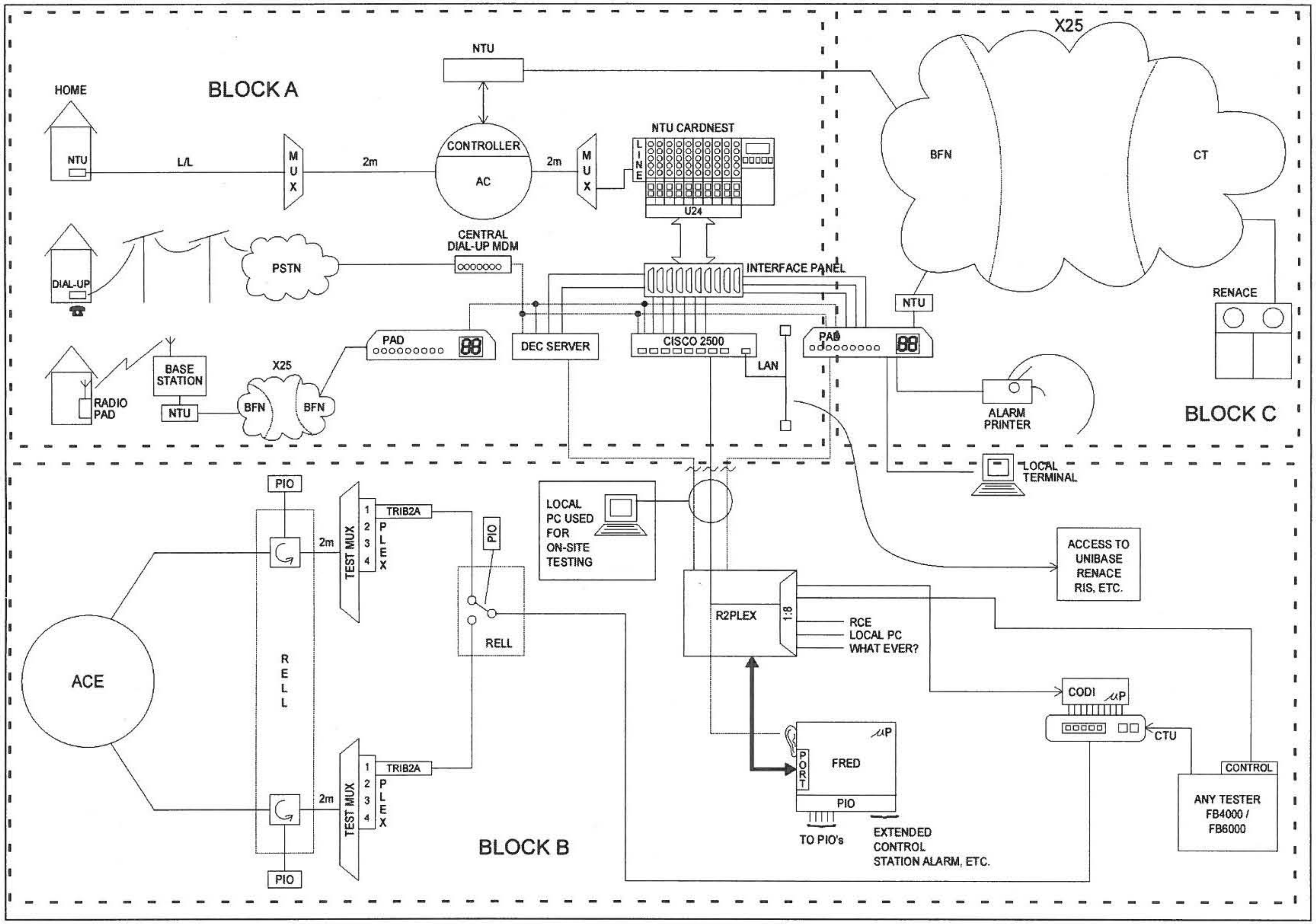


Figure 1 - Block diagram showing the complete remote test access configuration.

It is possible to cure this problem by implementing a Graphical User Interface (like Windows 3.1® or Windows 95®).

The graphical interface acts as visual interface to the user, and a single-board computer (special device) as a control device to perform all the manual functions such as switching the testers, timeslots etc.

The above block diagram shows three blocks: A, B and C. These three blocks identify three areas which where taken into consideration while this research project was developed. Block C forms part of the original network. At first, all tests were performed utilizing the PAD and the RENACE computer. Block A is part of the initial design to make it possible to control the RENACE equipment from a remote place. This block formed an integral part of the project, since it would become the only way any user could communicate with Block B or C from a remote destination. Block B is the actual hardware of the remote controllable test desk. The PLEX and CODI units that can be identified from block B was designed and developed by two colleagues. Block B and block A, as well as the graphical interface, were developed by the author of this document.

Block A shows the different methods by which a user can gain access to the test system, while Block B shows a detailed diagram of all the parts of the remote-control test desk. The ACE forms part of the Diginet network through which all dedicated circuits are switched and tested. The TEST MUXs also form part of the original test access configuration. Tests were performed on these devices by using patch-cords. The RELL unit replaced the patch-cord system by making use of relays. FRED is a micro-controller based unit capable of controlling the RELL unit and the R2PLEX unit by sending simple commands to its RS232 interface. The R2PLEX unit makes it possible to connect the input communication of FRED to any of eight serial (RS232) devices connected to the FRED unit. These units are CODI, FB6000 or any other device that can be controlled via a RS232 port.

1.2 Purpose of this research

The main purpose of this research is to save time in testing faulty data circuits by being able to test from a non-centralized test centre in order to save on manual work and to simplify testing.

Since most data circuits play an important role in many big firms these days, downtime in communication links costs money. The faster technicians respond to faulty circuits, the less downtime a client has.

Testing from a non-centralized test centre makes it possible for Telkom employees to respond faster and to identify the problem area without having to travel to the test centre. In most of the cases the problem is not even in the same region the fault has been reported in.

In the past testing was done manually, taking time and expertise. Today testing is becoming more and more complicated and it is thus necessary to simplify the testing procedure. Making the test procedure more understandable for everybody means that it must be implemented on a national basis without having to provide in-depth training to all personnel involved.

1.3 Hypothesis

A Graphical User Interface (Microsoft Windows® operating system) can be used to interface with custom designed hardware to make remote testing of digital data communication circuits possible.

1.4 Research domain

Two main types of remote-control scenarios exist, namely GPIB and RS232 control. No GPIB networks existed in the current test scenario, which concluded to the viability of investigating dial-up networking and direct RS232 control.

It was decided to design a network that contains point-to-point circuits as well as dial-up circuits. It was required for point-to-point circuits for standby personnel, as well as dial-up circuits for day-time and field personnel.

All of the incoming circuits are connected to a terminal server, which is in turn connected to the test equipment and control facilities. All of the circuits are full-duplex, making it possible for fast response times.

1.5 Method of research

The following design steps has been implemented for the project, namely:

- Design of the communications network.
- Design of the single-board microcontroller (FRED).
- Design and testing of FREDs software.

- Design of the test-multiplexer backplane switch (PLEX).
- Design of the 2M HDB3 switch (RELL).
- Design of the RS232 multiplexer (R2PLEX).
- Putting it all together and further testing.
- Design of the Graphical User Interface (GUI)

1.5.1 Design of the communications network

A CISCO 500 CS® has been utilized as the terminal server. It has 16xRS232 ports available. Therefore, the ports had to be allocated carefully. Twelve ports were allocated to direct circuits, one to the dial-up, and one to FRED. Access to the RENACE and Unibase® is accomplished through TCP/IP on the Intranet.

1.5.2 Design of the single-board microcontroller (FRED)

It has been decided to make use of the MCS51 products because of its flexibility, availability and ease of design. Wide ranges of software compilers are readily available at reasonable prices.

The hardware consists of 32K static RAM. Input/output functions are accomplished through one 8255 PIO integrated circuit. The computer is serially controlled via its on-board communications port, converting it to RS232 levels with a MAX232 integrated circuit. A Bitrate of 9600b/s was chosen, as it is the maximum speed of the RS232 remote-control ports of all the test equipment. The reason for this speed is later fully explained in the design of R2PLEX. The BIOS is stored on a 27256 EPROM.

1.5.3 Design and testing of FREDs software

All the software for FRED was written in BXC51 BASIC CROSS COMPILER, the main reason being its ease of use and availability. All programs are coded in BASIC and compiled to HEX format. The software is then programmed onto an EPROM, which is inserted in the micro-controller circuit. When the power is activated, the controller will boot from the EPROM and execute its program.

1.5.4 Design of the test-multiplexer backplane switch (PLEX)

The co-directional cards used in a test-multiplexer conform to the ITU G703 recommendation. The price to build an interface to conform to the G703 would be very high. Every tester were to access four timeslots on every test-multiplexer - that is four co-directional interfaces to each test multiplexer times the amount of Automatic Cross-connecting Equipment (ACEs) in every region.

A Modification was devised to save a lot of money by designing a card, which slots into the test-multiplexer backplane. Consisting of a normal 4:1 TTL multiplexer and a 1:4 demultiplexer, only one G703 interface will be used on every test position of any test multiplexer.

A colleague[1] was responsible for this design. Thus, only the interfacing of the PLEX unit to FREDs PIO was of importance to this study.

---

[1] Mr D de Bruyn

1.5.5 Design of the 2M HDB3 switch (RELL)

A decision was made to use relays for this switch, as the electronics for switching the 2Mbit/s signal was rather complicated and funds limited. The relays worked well enough for the initiation of the project.

Only one version of this board was designed to get the project off the ground. It is capable of switching between two ACE nodes and looping the transmit and receive-pairs of the test multiplexer.

1.5.6 Design of the RS232 multiplexer (R2PLEX)

The first terminal server had only eight ports, and many users made use of the network. At least four ports had to be available for FRED3, CODI, FB4000, FB6000 etc. Thus, an eight-port RS232 switch capable of working on the same port as FRED, was built to monitor the port for an incoming byte sequence (password sequence). On recognition of the correct sequence, it switches to local prompt until the next switch command is given. The other devices are then connected to the outgoing ports of the R2PLEX. All the software has to do is to tell the controller which device it wishes to communicate with.

Two limitations came to hand:

- For simplicity in the design, only 3 wire communications were implemented.
- To simplify the communication through FRED to R2PLEX, it was decided to limit the interface speed to 9600bit/s.

1.5.7 Design of the Graphical User Interface (GUI)

The GUI (Graphical User Interface) plays the most important role in the project, since it simplifies the interface between the technician and the remote test desk. Microsoft Visual C++ v1.5® was used to programme the GUI.

The GUI consisted of two programs:

- Setup.exe
- FB4000.exe

Setup.exe is used to set up all the parameters to be used by FB4000.exe. Parameters like connection type, baud rate and COM port are set here.

FB4000.exe initialize from the data file created by Setup.exe. The most frequently used test parameters were incorporated in the display of the test results. The programme inquires the tester for results and displays them in an orderly fashion on the screen. The programme also interfaces with FRED, CODI, R2PLEX, etc. No connection is established if a device is not properly connected to R2PLEX.

1.5.8 Design of the CODI modification

During a normal test procedure, a CTU (Co-directional Terminating Unit) is used to interface the test equipment to the text multiplexer. CODI was built to act as a remote control interface through which the GUI could control the interaction between the test equipment and the circuit under test. The CODI unit is responsible for selecting the speed of the test, as well as applying loops on the remote NTU's under test. The

1.5.7 Design of the Graphical User Interface (GUI)

The GUI (Graphical User Interface) plays the most important role in the project, since it simplifies the interface between the technician and the remote test desk. Microsoft Visual C++ v1.5® was used to programme the GUI.

The GUI consisted of two programs:

- Setup.exe
- FB4000.exe

Setup.exe is used to set up all the parameters to be used by FB4000.exe. Parameters like connection type, baud rate and COM port are set here.

FB4000.exe initialize from the data file created by Setup.exe. The most frequently used test parameters were incorporated in the display of the test results. The programme inquires the tester for results and displays them in an orderly fashion on the screen. The programme also interfaces with FRED, CODI, R2PLEX, etc. No connection is established if a device is not properly connected to R2PLEX.

1.5.8 Design of the CODI modification

During a normal test procedure, a CTU (Co-directional Terminating Unit) is used to interface the test equipment to the text multiplexer. CODI was built to act as a remote control interface through which the GUI could control the interaction between the test equipment and the circuit under test. The CODI unit is responsible for selecting the speed of the test, as well as applying loops on the remote NTU's under test. The manufacturers of the CTU, Alcatel Altech Telecoms®, incorporated the CODI unit into the new design of the CTU.

1.5.9 Evaluation of the system

The system has been tested in the working environment for over two years, and all the major bugs were resolved.

Numerous demonstrations were held to groups of employees all over the country, and the results were very positive. Telkom SA Ltd. is busy implementing the system in the existing Diginet network.

1.5.10 Summary

This chapter briefly explained the importance of this research, whilst an overview of all the hardware was provided. The following chapters will discuss the mentioned topics in depth:

- Chapter 2 – Fault acceptance and testing procedures.
- Chapter 3 – Remote user access to the remote test desk.
- Chapter 4 – The design of the physical hardware of the remote test desk.
- Chapter 5 – The Graphical User Interface that controls the hardware.
- Chapter 6 – Experiments performed on the final system.
- Chapter 7 – Further studies to be performed.

Chapter 2

Maintenance procedures

This chapter will discuss the procedures that are taken when a fault is reported to a telecommunications service provider. First, a few practical examples will be given of dedicated circuits. Thereafter a practical example of a digital point-to-point circuit will be given from which the test points (maintenance entities as defined by the ITU M series recommendations [53]) will be identified. Most of the tests that are to be performed on any data circuit is on layer one of the OSI model, therefore the service provider's responsibilities stop at the interface on the client's premises (which is on layer 1). However, in cases where the dedicated circuit is used to access X25 and frame relay nodes, it is the service provider's responsibility to support testing of layers two and three [54]. There are of course instances where the service provider can install and maintain a client's internal LAN. In this case, support is given on all seven layers of the OSI model. A broad discussion will be given on layer one, while only references will be made to layers two and three. Layers four to seven is beyond the scope of WAN testing and will therefore not be dealt with. At the end of this chapter, remote-control issues will be discussed. Consult the OSI model for more information.

2.1 Introduction

In general a company wants variable bandwidth links between its offices, with 100% error-free efficiency, meaning a data through-put with little or no errors and little or no breaks. Figure 2 shows a typical network of a company that is in the banking industry.

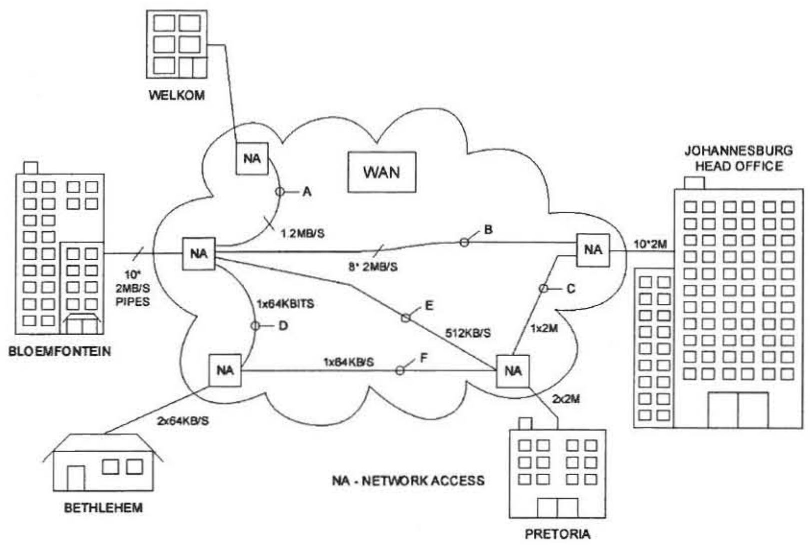


Figure 2 - Simple diagram showing an example of a typical client computer network.

The company's mainframe computer is situated in Johannesburg. Hypothetically, if link B was running a fare amount of errors with an efficiency of fifty percent, file transfers from Bloemfontein to Johannesburg would be cut in half, resulting in all money withdrawals etc. to be slowed down. When the whole of link B goes down, the only access would be through links E and C, bringing the whole business partially to a standstill. If both links B and C were down, Johannesburg would be isolated, and all transactions would be terminated until link C or B was repaired.

The client is supplied with direct connections between branches and offices. When one link goes down, there will usually be a back-up link to take the load. This is not always true, as in the case of link A.

Since the withdrawal of money from a bank is possible 24 hours a day by means of an automatic teller machine, a break in link A can be considered as critical to the company.

From the service provider's point of view, testing the links after-hours is not always an easy task, considering the fact that all the test equipment and access points are situated at the regional test centre. Every time an after-hours' fault is logged, it must be tested from the test centre, just to find out that the fault is in another region, or that the user's equipment does not send out the correct interface signals. It also means the support staff has to drive to the test centre, which takes more time to find the problem location.

By designing a remote test desk, testing of circuits can be achieved by virtually any medium possible, for instance: dial-up modem, direct point-to-point circuit, radio pad, Internet, etc. The design of the remote desk is dealt with in Chapters 3, 4 and 5, while the actual fault procedure is discussed in this Chapter.

2.2 Circuit "Walk through"

The best way to explain the working of a data circuit is to explain a few examples. In Figure 3, there are five circuits to be identified: A, B, C, D and E.

Circuit A:

Circuit A is working in the local area of Cape Town from Somerset West to Stellenbosch. The client's computer is connected to the NTU (Network Terminating Unit) by means of an interface cable. The NTU terminates the client's interface onto the digital network. The NTU is connected to the underground cabling network, extended to the nearest exchange where the local multiplexer is located. This multiplexer is a TDM (Time-division Multiplexer) having 31+1 timeslots of 64kbit/s adding up to 2Mbit/s. The 32 timeslots are combined and transmitted over a 2Mbit/s pipe to the nearest ACE (Automatic Cross-connection Equipment) site which is in Cape Town. The 2Mbit/s pipe can be seen as a transmission medium, which extends the information, received from the multiplexer to the ACE over great distances (from a few meters up to thousands of kilometers) in a transmit and receive direction. The ACE provides a permanent programmable switch path for the client's circuit [51]. In this case, it is programmed to permanently connect Somerset West to Stellenbosch. This circuit is therefore available 24 hours a day, and is better known as a point-to-point or dedicated circuit.

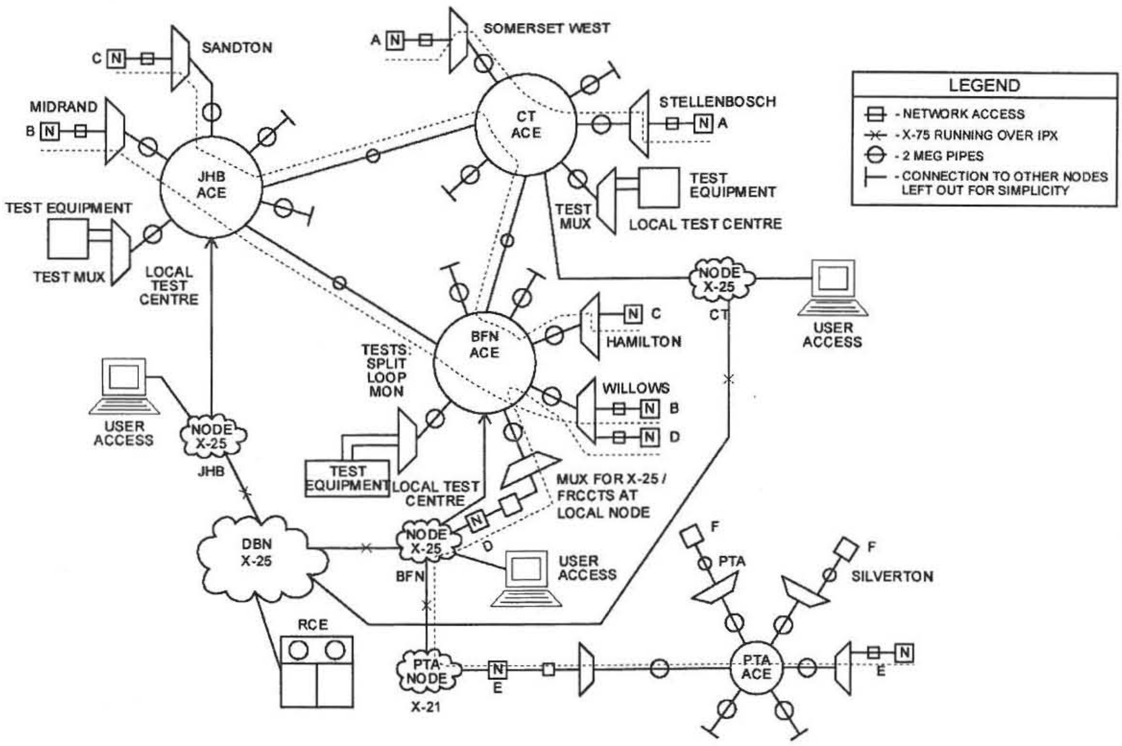


Figure 3 - Circuit "Walk through" showing a few possible ways of switching a client's circuit by means a digital network.

Circuit B:

When a circuit works from one city to another like in this example, it takes the same path as circuit A to get to the nearest ACE site. Circuit B runs through the NTU in Midrand, underground cable network, multiplexer, 2Mbit/s pipe, ACE in Johannesburg, another 2Mbit/s pipe, ACE in Bloemfontein, 2Mbit/s pipe, multiplexer in Willows and out to the client's premises. The second 2Mbit/s pipe between the ACEs is the same as the 2Mbit/s pipes that connect the local multiplexers to the ACE sites.

A circuit can operate at any bit rate from 2400bit/s up to 1920kbit/s. Circuit speeds up to and including 64kbit/s only utilize one timeslot on a multiplexer. Circuit speeds higher than 64kbit/s up to 1920kbit/s are known as N*64 circuits, and they increment at 64kbit/s intervals, starting at 128kbit/s and ending at 1920kbit/s. The higher the speed required by the client, the higher the cost of line rental and interface equipment. A client has to determine the amount and type of information to be sent before purchasing any equipment [52].

Circuit C:

This circuit runs through three ACE sites. Why does this circuit not run directly between Johannesburg and Bloemfontein? The reason for this is very simple. When looking at a full-grown dedicated network infrastructure, there are multiple 2Mbit/s running between ACE sites. Sometimes it happens that all the transmission links are fully utilized. An alternate route is taken, in this case, Johannesburg to Cape Town to Bloemfontein, until spare timeslots become available on the link between

Johannesburg and Bloemfontein. The maximum number of 2Mbit/s connections is determined by the number of ports available on the ACE.

In South Africa 128 and 512 port ACEs are used. A Port is simply the interface card for the 2Mbit/s-line code to the ACE switch core. The 2Mbit/s-line code that is used is known as HDB3 and conforms to ITU standards G703 and G704. When all the ports on an ACE is filled up, a new ACE is installed and connected to as many ACEs as possible on a national basis.

Circuits D&E:

Circuits D and E are both access circuits to the X25 network. X25 is known as a public data network (PDN) and resembles switchable (on client's request) connections. X25 is used where a client wants to connect to various sites of his choice to transfer information. It is similar to a telephone exchange only for synchronous modems. In this example a virtual circuit is connected between Willows and Pretoria. Circuit D is connected to the X25 node in Bloemfontein and then virtually switched through the PDN to get to the Pretoria node where it is then connected to access circuit E.

The big benefit of using X25 is the built-in error correction running on layer 2 of the OSI reference model. The data that the client wants to sent is made up in packets by a Packet Assembler Disassembler (PAD) and then send to the nearest X25 node. To establish the connection from circuit D to E, the user has to transmit the DTE address ("Telephone number") of the site in Pretoria to the PAD on site. The PAD then initiates, and sets up the call to the remote site [54].

X25 circuits are also used to control all the ACE's, because it is possible to connect to any ACE. When a test is to be performed on a specific circuit like circuit C, a technician in the Cape Town test-room could connect to the RCE in Durban. He then types in diagnostic commands to perform tests on the circuit. These commands are sent through to the RCE. The RCE determines which ACE is involved in the present command. It contacts the particular ACE, sends through the command, and establish a confirmation that the command has been successfully completed. The RCE then sends a confirmation back to the technician in Cape Town whether the command was successful or not. This whole process normally takes more or less 10 seconds. The technician in Cape Town can also do diagnostic commands on the ACE in Bloemfontein or Johannesburg. He is limited to do status reports only, like display physical level 1 alarm information, for example: NTU power off [52].

2.3 Fault-reporting procedure

The following discussion is a typical procedure for fault acceptance by Telkom SA Ltd., all telematics divisions (Diginet, Datel, Saponet, Frame relay etc.).

A failure on any part of the physical network will result in breaks and/or errors on one or more circuits. The client does not care about a minor (cable) or a major (optic-fiber) failure, he/she just wants the connection to be re-established. After the client verified that his interface-cable and communication equipment is working properly, he/she reports a fault to the service provider.

A lot of fault-reporting centres is established around the country. The client normally phones the nearest centre, but he/she can phone almost any centre because all the faults are reported onto a general fault management system called Unibase®. The client is supplied with a reference number when fault inquiries are made. If a fault meant for Cape Town is reported in Bloemfontein, the fault system automatically routes the fault to the right location (in this case Cape Town). Figure 4 shows what the fault-reporting form looks like on Unibase®.

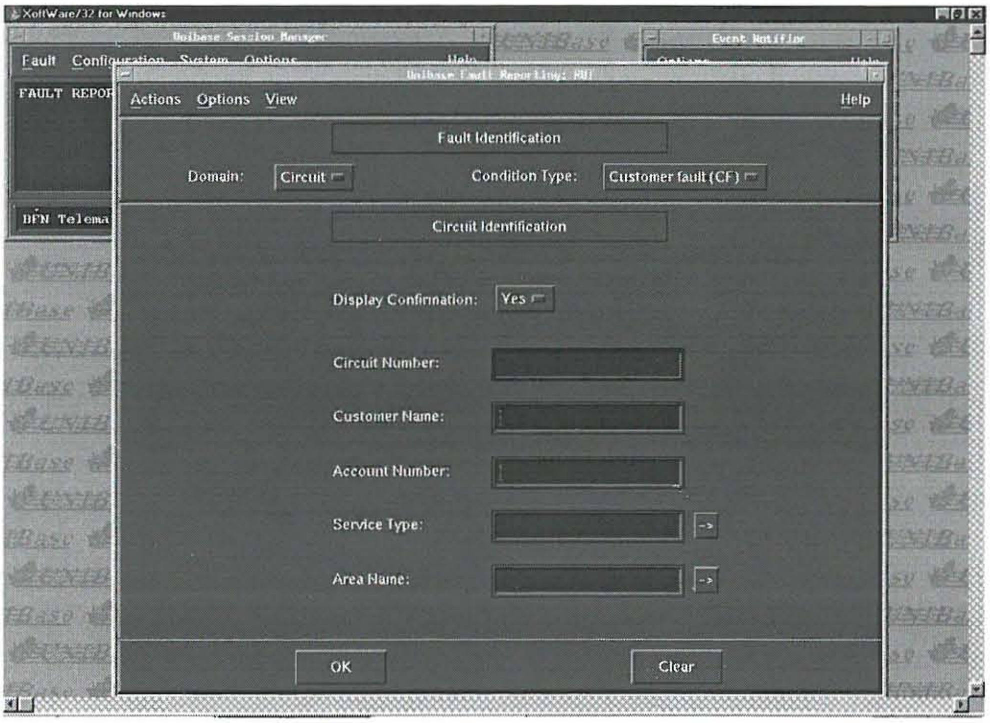


Figure 4 - Screen capture of the fault-reporting form of Unibase®.

The fault is placed in a queue of the Cape Town test centre (Help centre). A Faultsman then attends the fault out of the centre's fault queue. After the fault was localized and repaired, the client is informed.

During the fault-testing procedure, the fault can be handed to numerous centres for testing.

2.4 Fault-testing procedure

Figure 5 shows the basic layout of a digital network [52].

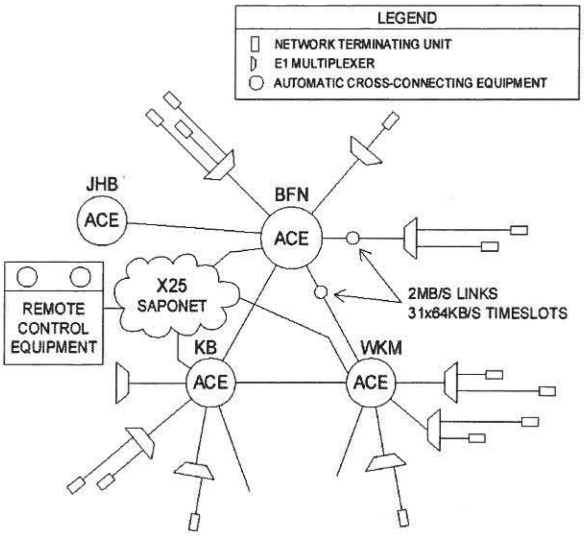


Figure 5 - Simplified Diginet network.

Figure 6 shows a typical test-access configuration that is being used to test circuits [52].

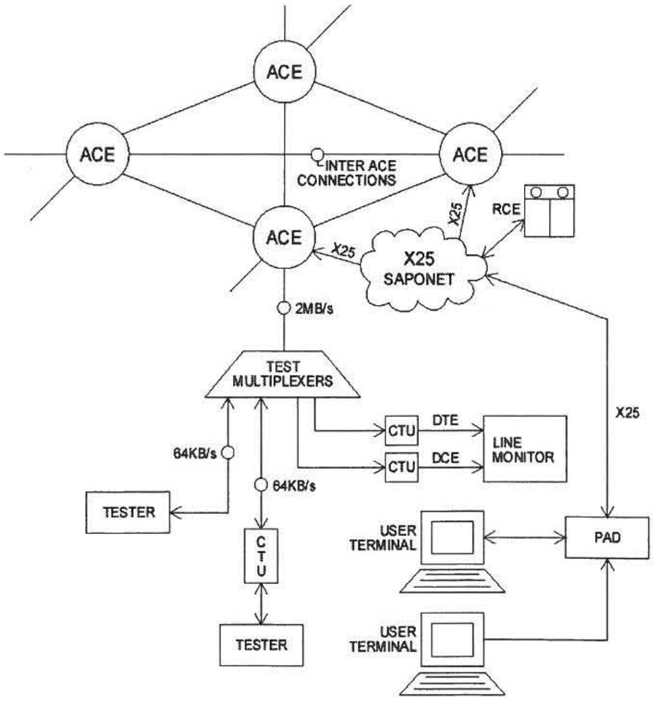


Figure 6 - Basic test-access configuration showing how test access can be gained to the Diginet network.

To test a digital circuit numerous software programs, equipment and test equipment are used. Here follows an explanation of the procedure: A fault is reported on the Unibase® fault system. The fault is attended on an X-terminal or on a personal computer running an X-terminal emulation (like Xoftware® or Xvision®). To manipulate the circuit that is to be tested, the Remote Control Equipment is given commands, which is sent through to the ACEs. This is done by either a dumb terminal (Wyse® or Hewlett Packard® or any other device that is VT-100, VT-220 compatible) or an X-terminal, running a VT-100 terminal emulation. Tests that are to be performed on a circuit is connected to the local test centre's test multiplexer (similar to the multiplexers used to terminate the NTU's). Two sides, an A and a B side are allocated to every circuit under test. A CTU (Co-directional Terminating Unit) is connected to the relevant timeslot on the test multiplexer. The CTU acts as a rate selector to interface the test equipment to the test circuit's bit rate. The rate of the circuit is kept on the RCE database together with the ACE switch information.

Here follows an in-depth example:

a) A client reports a circuit to the service provider (for example Telkom SA).

b) The client is given a reference number for later inquiries.

c) The fault is routed to the applicable help or test centre. This decision is performed by the fault system (for example Unibase®).

d) The faultsman (employee doing the test procedure also known as a tester) attends the fault from the centre's unattended queue.

e) Relevant information of the circuit is kept on the Unibase® database like: street address, client's telephone number, cabling details, distribution point details, circuit number, circuit speed, etc.

f) The switching details are provided by the RCE. Example: A circuit can switch through one or multiple ACEs. The switch path that is used by every circuit is kept in the RCE database as well as the ACE's processor and switch cards. To manipulate or change any of this information, the RCE is commanded to do so. LOOP and SPLIT are but some of the manipulations that can be performed on a circuit under test.

g) Before any testing is performed, the technician (tester) does diagnostic checks on the circuit to see if there are any physical alarms on the circuit. These checks are performed by sending keyboard commands to the RCE. The RCE in turn interrogates the relevant ACE according to its database to retrieve the information from the multiplexer [51]. These physical conditions are limited to a few, for example: DCE power off, loss of signal, etc. This information is only used as a direction, as not all fault conditions on a line are applicable to these alarms. A line usually consists of copper wire with a length of up to 5.5 kilometres [52]. Sometimes the technician can determine a line problem and fix it successfully without having to go on to the next step just by checking the alarm information (Example: the DCE power off alarm, the remote NTU's power is switched off).

h) Next, the circuit is manipulated into a SPLIT state by sending the RCE the relevant command. The circuit is broken into two parts by the ACE's switch plane: an A side and a B side. This is because every circuit consists of two NTU's. The NTU acts as an interface between the client's DTE and the service provider's network infrastructure. Sides A and B are allocated an individual timeslot on the test multiplexer. The technician performing the SPLIT command determines the positions on the test multiplexer. This adds the possibility to perform multiple tests by numerous technicians at the same time by using the same test multiplexer.

i) A tester like the TTC Fireberd 4000® or Fireberd 6000® is connected via a CTU to the side/timeslot under test. A loop is activated on the remote NTU under test by the CTU. The rate at which the circuit is working is set on the CTU. The loop that is applied redirects all information received from the remote NTU back to the ACE. Remember there are two NTU's on every circuit. The tester transmits a known pattern to the side (NTU) under test. The received pattern is compared to the transmit pattern. If this receive pattern is out of order, line errors will be metered by the test equipment.

j) If the circuit is proven to be error-free on both sides, the circuit is restored and the fault is cleared with the client. If a particular side is running errors on a test, numerous loops are applied to the circuit to eventually discover the fault area. The area, device or card causing the problem is restored and the client is informed.

2.5 Maintenance entities

A single circuit consists of many points or locations where tests can be performed. Test points are even sometimes added during the installation of the circuit to prove different portions of a circuit working between two sections. The test points or test locations are known as maintenance entities (ME's) as stated by the ITU M110 [11].

The following three diagrams show 10 points that can be identified for performing tests. Some of the points consist of more than one ME. This will be covered as the main areas are discussed.

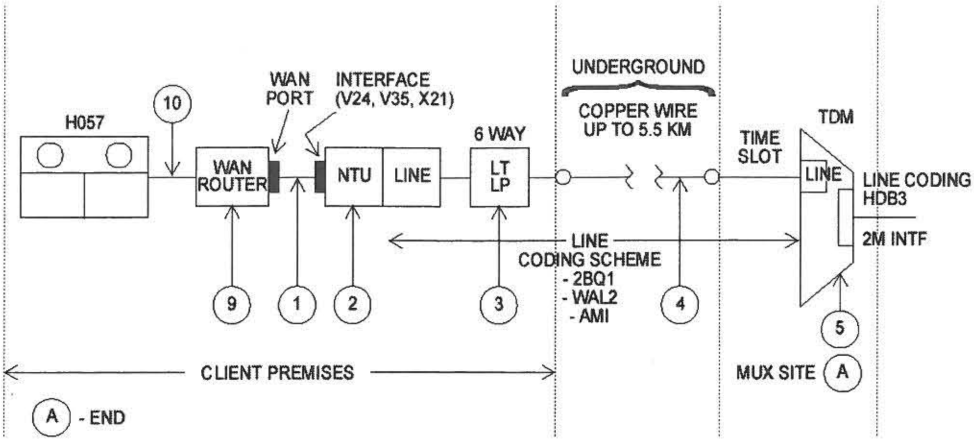


Figure 7 - Portion A. A-end of a digital circuit showing all the maintenance entities applicable to it.

Portion A of the circuit is switched by the ACE in portion B to get to the NTU at the B-end which is in portion C.

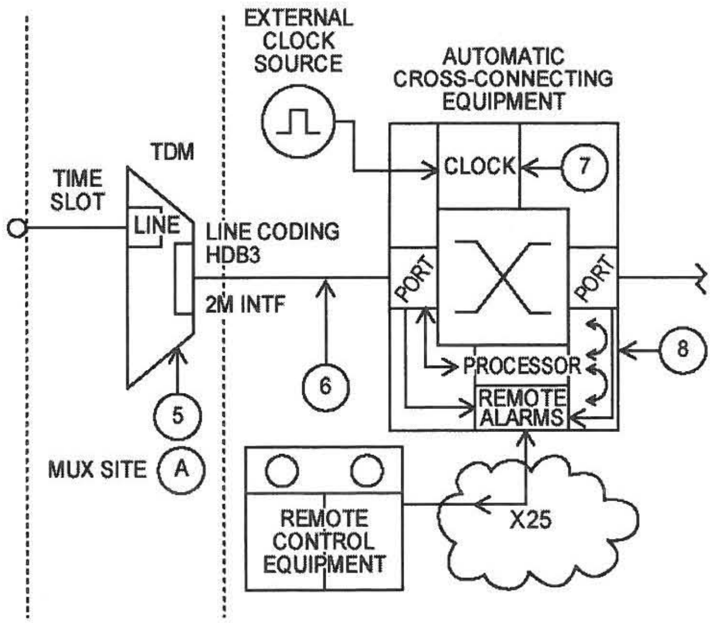


Figure 8 - Portion B. This shows the maintenance entities applicable to the cross-connect site.

Maintenance entities 6, 7 and 8 form part of the core of the digital network. Every circuit that is switched through the network is multiplexed into 2Mbit/s streams and switched by the ACE's switch plane from the A side to the B side.

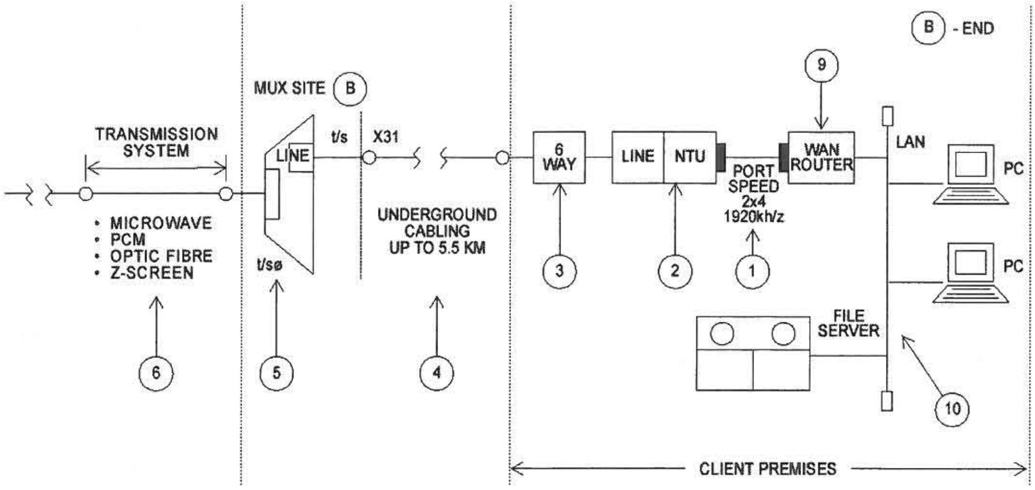


Figure 9 - Portion C. The B side of the circuit is similar to the A side shown above.

The exact locations (ME1 through 10) used in the following description is not defined by M110, but are common to how digital circuits are tested and referred to in South Africa.

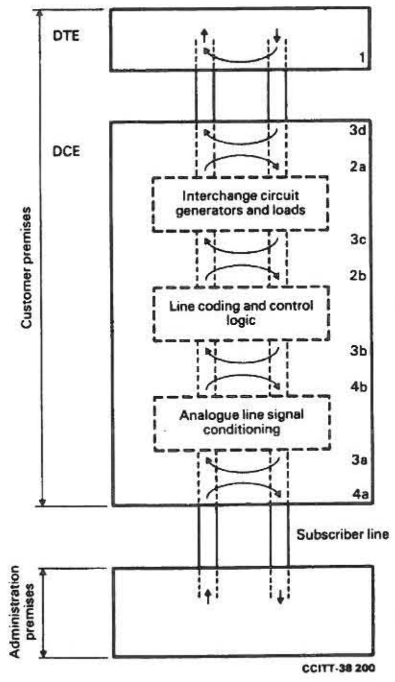
2.5.1 Maintenance entity 1

This portion deals with the interface cabling between the DTE and the DCE. This interface could be a X21, V24 or a V35. Refer to Appendix B for an interface comparison table. An interface test set or protocol analyzer is used to observe the client's data and hardware handshaking. Usually only a normal BER (Bit-error Rate) tester is used to transmit a data pattern like 511 or 2047 [15]. At the same time, the test equipment verifies the handshaking protocol between the DTE and the DCE. In cases where a BER tester is used, the client's interface is disconnected, hence no tests are performed towards the client's equipment. To verify the correctness of the client's interface cable, an interface patch box is used to display all the interface pins and signals. The different types of test equipment are discussed later in this chapter.

2.5.2 Maintenance entity 2

This entity is known as the NTU, and it performs almost the same function as a modem. All NTU's are capable of performing some test functions, all outlined in the ITU recommendations. M125 [12], X150 [30] and V54 [22] recommend many types of test loops that can be performed on a DCE. The NTU provides three different types that are usually accessible from the front panel (local loop, loopback and remote loop). A test loop is a mechanism that allows the device under test to send the received information back to the remote transmitter. The transmitter can then determine the correctness of the circuit up to the point where the loop is applied. This is done by comparing the transmit pattern with the received pattern.

Figure 10 shows all the different kinds of loops that are available.



Note – The back-to-back loopbacks (e.g. 3d/2a. 3c/2b. 3b/4b and 3a/4a) that are provided should be configured in such a manner that there is no active equipment between the loopbacks. For example: an Administration may operate the back-to-back loopbacks simultaneously in the same relay or switch.

FIGURE 1/X.150

Figure 10 - ITU X150 All the different loops available in a network terminating unit.

The main function of the DCE is to convert the line signals into acceptable interface signals for the DTE. When a modem is used, this would be a conversion from analogue to digital. If an NTU is used, the conversion would be from a digital line coding scheme (like AMI, 2BQ1, WAL2 etc. [52]) to a digital interface compatible signal like V28 [21] or V11 [19]. The only way to test the correctness of the internal circuitry of a DCE is to apply loops to different areas or circuit functions.

Test loops can be used to either test leased or circuit switched-line operations. Loops can be applied to the DCE without necessarily dispatching maintenance technicians to the client's premises. The advantage of this is a faster turn-around time on reported faults. Some of these loops can also be activated remotely from the test centre or by telephoning the client to activate the loops from the front panel of the DCE. In cases, where it is allowed, the DTE may also activate DCE loops by applying the correct interface signals. When using a V24 interface, circuits 140,141 and 142 are connected to pinouts 21,18 and 25 [20]. Refer to Appendix B.

When applying loops to different parts of a circuit, it must be kept in mind that the transfer of information from point-to-point will be disrupted. The client will not be able to transfer any information between his two sites when any loops are activated.

It can be seen in Figure 10 that the most frequently used loops are loop 3c, 2b and 3a. Loop 3c corresponds to the local loop, while 2b corresponds to an LB and a remote loop [30]. Loop 3a can only be applied from within the DCE, and is thus usually not available from the front panel functions of the NTU.

The following loops will now be discussed in further detail:

- Loop 3c,
- Loop 2b and
- Loop 3a.

2.5.2.1 Loop 3c

This loop is used to test the operation of the DTE. Information is sent from the DTE to the DCE interface circuitry and back to the DTE. This loop tests the client's cable and the DCE's interface circuitry only.

2.5.2.2 Loop 2b

This loop can by applied from three places, refer to Figure 11:

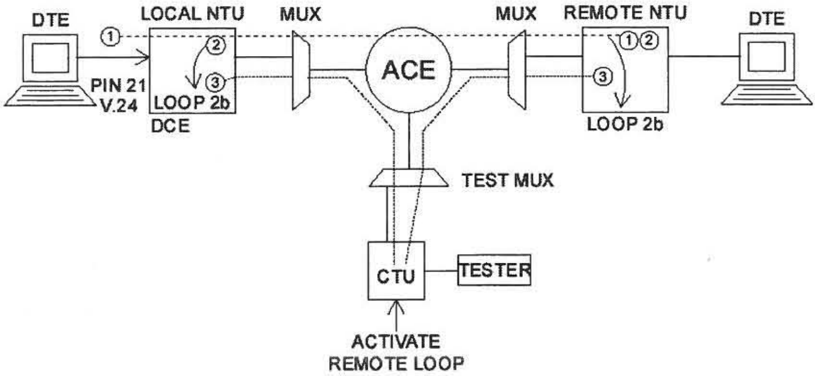


Figure 11 - Loop 2b activation at three locations.

Firstly, the local DTE can raise circuit 140, pin 21 (remote loop), causing the DCE to transmit a test condition to the remote DCE. The remote DCE will activate its loop 2b circuitry. With this loop, all components of the circuit is tested except for the remote DCE and DTE's interface circuitry.

Secondly, the client could also activate the loopback function on the front panel of the NTU, causing all data to be redirected back to the remote equipment. Likewise, the remote loopback could be activated from the front panel causing the remote DCE to activate a loopback back towards the testing party.

The third place where the loop is activated is from the local test centre testing the circuit. At the test centre, the circuit is SPLIT into two parts: A and B. By applying loop 2b tests, the A or the B side can be tested. This will become clear when discussing ME 7&8.

2.5.2.3 Loop 3a

This loop tests the operation of the DTE and the DCE. The data is redirected from the DTE through the DCE and back to the DTE. All but the connection to the line is tested by this loop.

2.5.3 Maintenance entity 3

Between the DCE and the 6-way is a cable connecting the DCE to the line. The 6-way usually provides lightning protection as well as test facilities.

The test facilities implemented are:

- Short,
- Disconnect and
- Loop.

Refer to Figure 12 for a graphical presentation of the line terminating box (6-way).

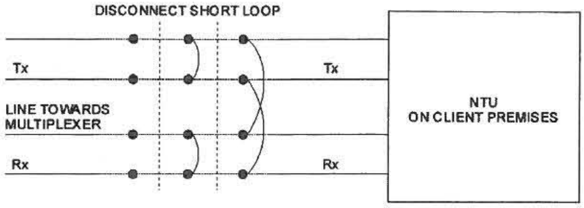


Figure 12 - Block diagram of a 6-way termination box.

2.5.3.1 Short

This test causes a zero ohm termination to be applied on the line towards the service provider's exchange building. This is especially useful when testing the resistance of the circuit path. A high resistance (above 900 ohms) could cause a circuit to run errors. A resistance of infinite value would obviously indicate a break in the circuit path. By just calling the client on a telephone, the local leads can be tested without having to send out any maintenance personnel. In the case of an open-circuit path, capacitance readings can be done to determine the distance of the break from the exchange building.

2.5.3.2 Disconnect

Sometimes a circuit path could also be shorted together causing errors. One way to determine if the DCE isn't causing a short circuit is to initiate the disconnect test on the 6-way box. This causes the DCE to be disconnected from the line presenting only the line conditions to the maintenance personnel. This test is also performed to determine if there is a short (closed circuit) between the 6-way and the service provider's premises. Special equipment is available that can show the faultsman the distance of a short or an open circuit by measuring line capacitance.

2.5.3.3 Loop

This facility is only useful when testing 4-wire circuits when two wires are used for transmitting and two wires for receiving. This is stated because in some cases 4-wire circuits are used, but both the two single pairs are used to transmit and receive data. In other applications, for example using 2-wire NTU's, only one pair is used. Since more and more circuits are using 2-wire encoding schemes, the 6-way loop facility is slowly becoming obsolete.

In cases where 4-wire circuits are still used, a sweep tone can be sent across the loop to determine the frequency characteristics of the line. Although it was previously stated that only digital circuits are used, it is in some cases necessary to provide access circuits to the rural areas where analogue transmission mediums are still in use. In this case, it is extremely useful to be able to determine the frequency characteristics of the access circuit on the digital network.

2.5.4 Maintenance entity 4

This portion could either be a pure underground cabling system extending up to 5.5 kilometres, or a carrier system which extends into the rural areas where no digital equipment is available yet. Thus, ME 4 will be dealt with in two parts, namely ME 4a and ME 4b.

2.5.4.1 Maintenance entity 4a

ME 4a consists of multiple cable segments running from the multiplexing equipment to the MDF (main distribution frame), out towards the client's premises through multiple distribution points and cross-connection boxes. The next figure shows a graphical presentation of this circuit portion.

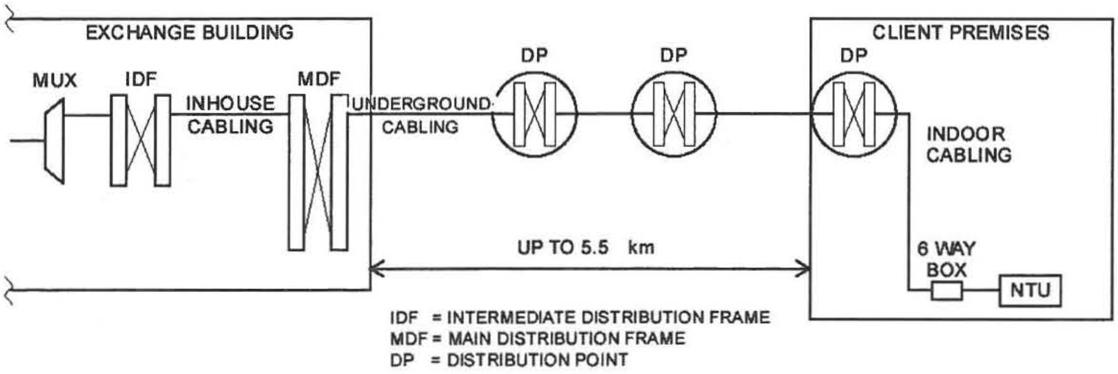


Figure 13 - This figure shows the typical path a circuit is connected with copper wire.

At each of the distribution points, tests can be applied to determine faulty cable sections. No special equipment is found in this section where an open-circuit, a loop or a short can be applied easily. Most of these tests are applied manually and must be done on request of the test centre's maintenance personnel. It usually involves personnel to be sent out to the different distribution points along the circuit path.

2.5.4.2 Maintenance entity 4b

This portion consists of a cable portion similar to ME 4a. Refer to Figure 14. The digital signals are converted to analogue signals, after which it is connected to the nearest analogue carrier system. The analogue signals are transported along the route through one or several stations until it reaches its end destination. From here, a normal cabling network is used up to the client's premises.

Along the route the analogue signals are either passed through PCM or through FDM systems. At every station, the signals can be looped manually for testing purposes. In addition to the looping mechanisms that can be applied, level metres, scopes and sweep tones can be used to trace the faulty portions of the circuit path.

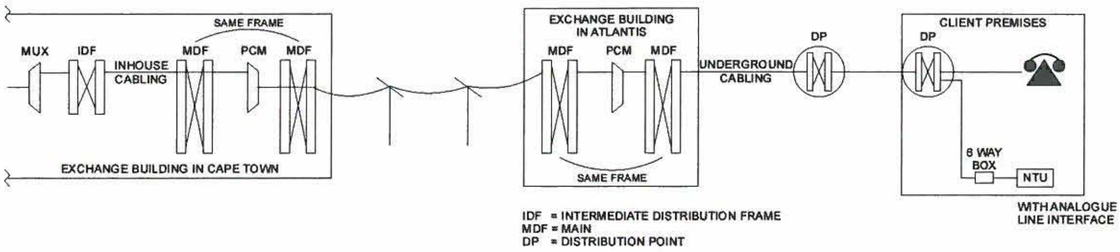


Figure 14 - Analogue channel access circuit, showing both the copper wire and channel.

2.5.5 Maintenance entity 5

This entity forms part of the multiplexer. The multiplexer consists of line cards and common equipment. The common equipment distributes the information from the 2Mbit/s stream to each individual timeslot/line card. The line cards on the other hand interface the incoming line code to the digital network. The following figure shows a simplified graphical representation of a multiplexer.

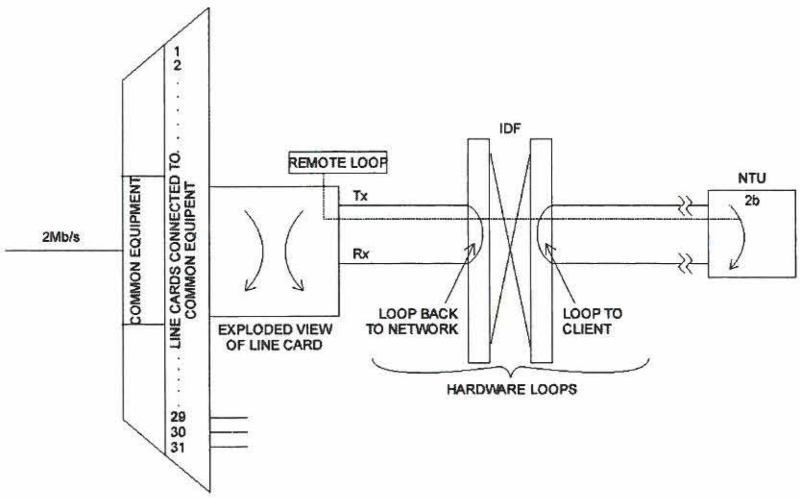


Figure 15 - Tests that can be applied on the digital multiplexer.

From the above figure, it can be seen that loops can be applied on the line cards towards the client's premises or towards the ACE. In some instances it is even possible to remote the NTU that is connected to the local multiplexer.

A brief discussion will now be given on multiplexing.

2.5.5.1 Time-division multiplexing

When a 2Mbit/s connection (E1) is established between two ends, it is necessary to provide 64kbit/s channels for the transport of voice and data. TDM is mainly used for digital data circuits, while PCM is used for voice channels. A TDM's main function is to group 32, 64kbit/s channels (called timeslots) together to be transmitted over the 2Mbit/s medium (or "pipe") and back again. The way it is performed is shown in the figure blow.

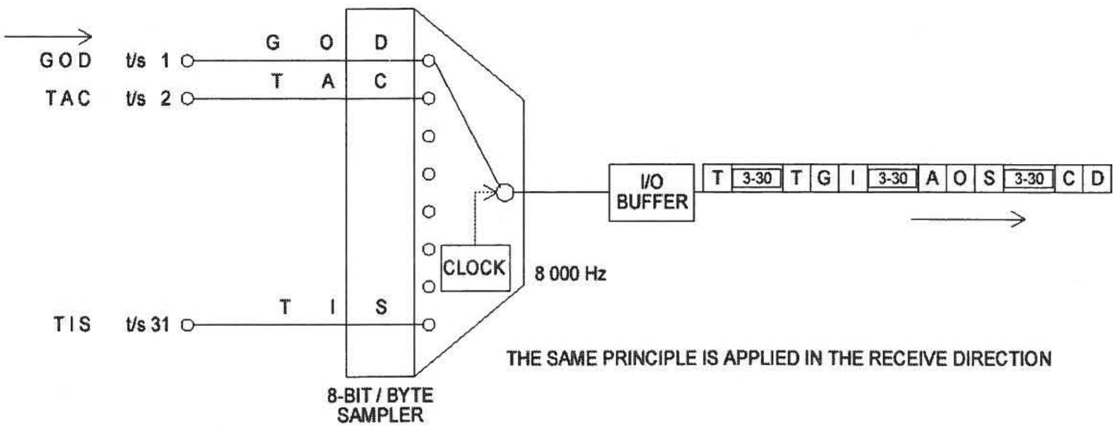


Figure 16 - Simplified view of time-division multiplexing.

Each timeslot is sequentially scanned for information at a rate of 8000 hertz. As an example the timeslots can have the following sequence: D, C, timeslots 3 to 30 and then S. The second scan would be O, A, 3 to 30, I and so on. The remote multiplexer would then reassemble the information as DOG, CAT and SIT. Think of every letter as representing eight bits to make up the 64kbit/s timeslots. This is of course a very simplified explanation of the TDM principle that is used in the digital multiplexer.

2.5.5.2 The multiplexer

To extract the timeslots from the 2Mbit/s stream, a multiplexer is connected at the remote end. The multiplexer (which is a TDM) receives the data and the clocking signal from the ACE and distributes it to the 32 timeslots, which in turn is connected to the Network Terminating Units (NTU's). Every timeslot supplies a maximum bandwidth of 64kbit/s. One of the thirty-two timeslots, timeslot 0, is used to carry the physical (level 1 of the OSI-model) alarm information of every timeslot back to the remote-alarm concentrator located at the ACE. Only 31 timeslots (1-31) on each multiplexer is available to client's circuits.

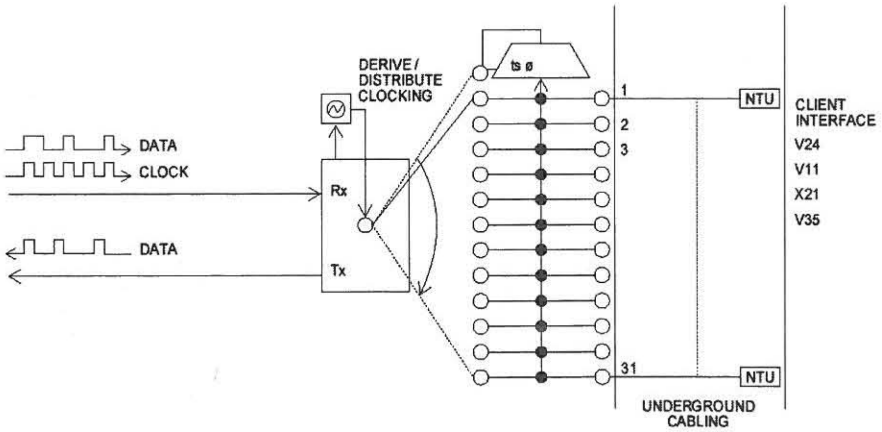


Figure 17 - Block diagram of the digital multiplexer showing the clock that is extracted on the receive path and the timeslot information that is extracted from every timeslot.

Most of the problems in data circuits occur in the physical layer. When a circuit is faulty, the first step is to check if the physical conditions are normal. If the physical layer is faulty, the problem area must be located by deductive reasoning.

Unfortunately, there are no sets of rules to describe exactly where to start. Depending on the network management system, a service provider usually starts in the middle, testing outward towards the client's interface equipment. Even this is not a set rule to follow. The best would be to define a set of rules and guidelines according to personal experience.

2.5.6 Maintenance entity 6

This entity is responsible for providing an error-free connection to the ACE line cards. The only way tests can be performed here is by means of test equipment connected directly to the 2Mbit/s stream. The data stream conforms to the ITU G703 [5] and G704 [6] recommendations. A tester can be set up to test the whole 2Mbit/s. It can be set up to monitor the signal for errors. On the other hand, specific timeslots can be dropped and inserted. This last test can also be accomplished by performing a SPLIT on the ACE switch plane and testing a specific timeslot. By applying hardwire loops on the 2Mbit/s stream from transmit to receive, the portion connecting the multiplexer can be tested. This is shown in the next figure.

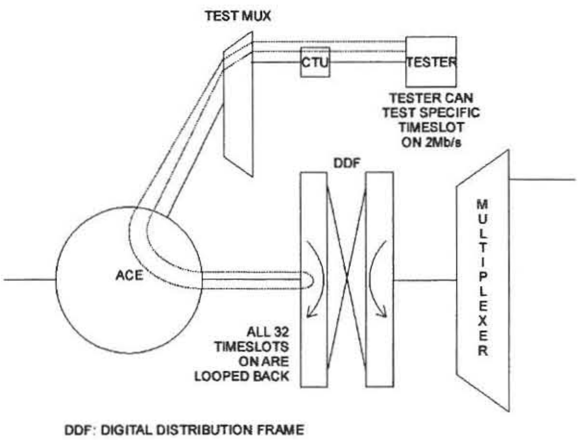


Figure 18 - Tests performed on a 2Mbit/s circuit can only be performed by applying hardware or manual loops. This is known as an out-of-service test.

By performing the latter test discussed above, it is possible to determine if a 2Mbit/s stream is faulty by means of remote testing.

2.5.7 Maintenance entity 7

At this entity, the ACE internal clock and incoming clock signals can be monitored without the aid of any test equipment. If some errors do occur, they will be displayed on a visual indication on the ACE. However, it is also possible to retrieve some information about clock alarms from the RCE. This also means that it can be done remotely by issuing some commands from a remote terminal screen. It is important to monitor this entity, because severe errors here can effect all circuits connected to the ACE.

For better understanding a brief discussion on the operation of the ACE will be given.

2.5.7.1 Digital cross-connect switch (ACE)

The basic function of an ACE in a dedicated point-to-point network is to provide a switching point where any of up to 512*31 timeslots can be cross-connected as illustrated in Figure 19.

The switch-information of any point-to-point circuit can be altered at any time with software instructions fed to the Remote Control Equipment (RCE). Likewise, the circuit can be "split" in the ACE, so testing can be performed in both directions of the circuit being tested. The types of tests, for example "SPLIT","MONITOR" etc., will be discussed later.

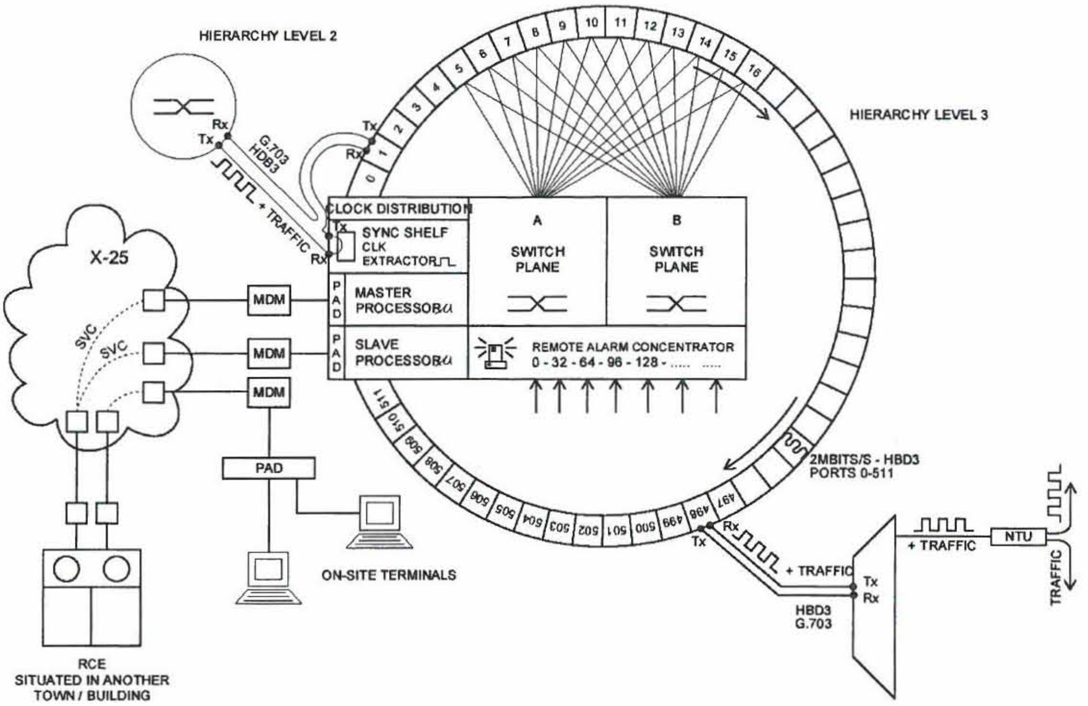


Figure 19 - Block diagram of an ACE showing all the different components interact.

The ACE receives the main clocking signal from an ACE higher in the hierarchy level and distributes it to all the ports connected to the ACE.

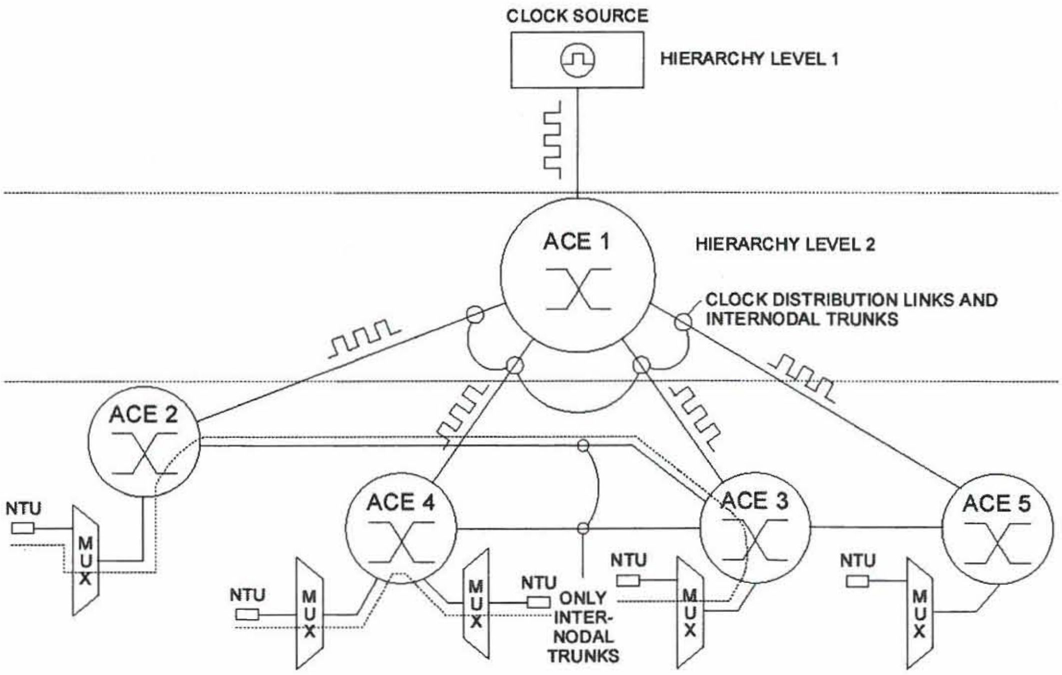


Figure 20 - Clocking hierarchy showing how the clock is retrieved from level 1 causing the whole network to be synchronized.

Every port is connected to a 2Mbit/s pipe which contains a serial stream of 32*64kbit/s timeslots.

All devices in a digital network have an internal clock rate generator or oscillator, which cycles at a frequency near the desired receiving frequency. Each device will lock its internal clock onto the receive signal with the aid of a phase-locked loop circuit. This ensures that the slave (device receiving the incoming signals) will shift the phase of its internal clock to precisely meet the incoming signal.

When data is then sent from the slave device to the network, it will also use the derived clock signal and thus stay in synchronization with the whole network. This method is employed from the ACE to the MUX to the NTU and finally to the client's DTE equipment.

2.5.8 Maintenance entity 8

This entity enables the RCE to monitor the whole digital network. It is possible to retrieve physical alarm information from any 2Mbit/s port, multiplexer or timeslot by issuing commands to the ACE processor. From here, it is also possible to configure circuits.

Existing circuits can be altered, for example: LOOP, SPLIT etc. A SPLIT condition is shown in the next figure.

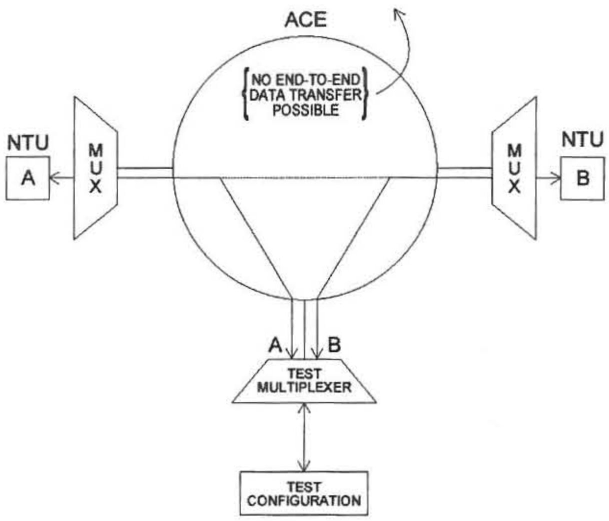


Figure 21 - This figure shows the SPLIT condition. It is activated by issuing a software command to the ACE causing it to reconfigure its switch plane to redirect the A and B-end towards the test multiplexer.

As previously stated, all the ACE processors are connected to the RCE via the X25 network. This enables the users of the RCE system to control any ACE, multiplexer or circuit with the issuing of commands.

2.5.9 Maintenance entity 9&10

These entities are capable of monitoring the circuit on a full-time basis.

It will report parameters like:

- The physical state of the interface connected to the NTU.
- Dropped packets.
- Uptime.
- Throughput etc.

Referring to the CISCO® command manual, the ">Show interface s0" will show some of the parameters mentioned above [2].

In most cases, it is the client's responsibility to look after the router and the remote LAN. The client monitors his/her network by using the TELNET protocol to access remote routers. By typing a few commands, it is possible to determine if the circuits connected to the specific router are up and running.

2.6 Test equipment

Test equipment are used to test the credibility of a circuit or a portion of circuit by monitoring certain conditions or by inserting known information into the circuit and then verifying that the information is received correctly again. This in short is the definition of test equipment. Many types of test equipment are available to test data circuits. Only a few types will be discussed in this document.

The most common testers in use today are the:

- BER tester.
- Interface tester.
- Protocol analyzer.

2.6.1 BER tester

BER stands for Bit Error Rate. This tester generates a known test pattern and injects it into the circuit via its interface. This pattern is received either on the remote side by a similar tester or is looped back by the NTU to be received by the sender. This pattern is then monitored for errors and is expressed as bit errors, pattern slips, pattern losses etc. on the tester's display. The pattern that is injected conforms to ITU I.150 [15], I.151 [16], I.152 [17] and I.153 [18] recommendations.

Numerous BER testers are available, for example:

- TTC® (Telecommunications Techniques Corporation®) Fireberd 4000® and Fireberd 6000®.
- Marconi® 2871.
- Hewlett Packard® Telecom/Datacom analyzer 37722A® etc.

Only the Fireberd 4000® will be discussed, since it is more widely used in Telkom SA Ltd. All of the above testers can be used in a remote-control environment. At the early stages of the development of this project, the Marconi® 2871 was used for tests.

Telecommunications Techniques Corporation® Fireberd 4000® and 6000®

For the purpose of this research and any further discussion on the Fireberd 4000®, it is assumed that the tester is equipped with the RS449 DTE/DCE data interface model 40200®. More information on the interface module can be obtained from the TTC® operating manual ML11032. This interface module enables the tester to be connected to the remote controllable CTU's, V11 interface.

The following figure shows a picture of what the Fireberd 4000® tester looks like.

Figure 22 - Photo of a TTC® Fireberd 4000®.

Before discussing the remote controlling possibilities, it is necessary to discuss the IEEE 488.2 standard. It must be stated that both the Fireberd 4000® and the Fireberd 6000® support the IEEE 488.2 standard.

2.6.1.1 What is the IEEE 488.2 standard?

The IEEE 488 standard specifies how data is exchanged between test equipment. It does not specify how the data should be encoded, what the data signifies or when a device has to send data. This is because at the time the standard was developed, it was not defined well enough to develop a useful standard [36]. As the market place matured, the specification was amended until the IEEE 488.2 was developed. The IEEE 488.2 is organized into four layers that can be seen in the following diagram.

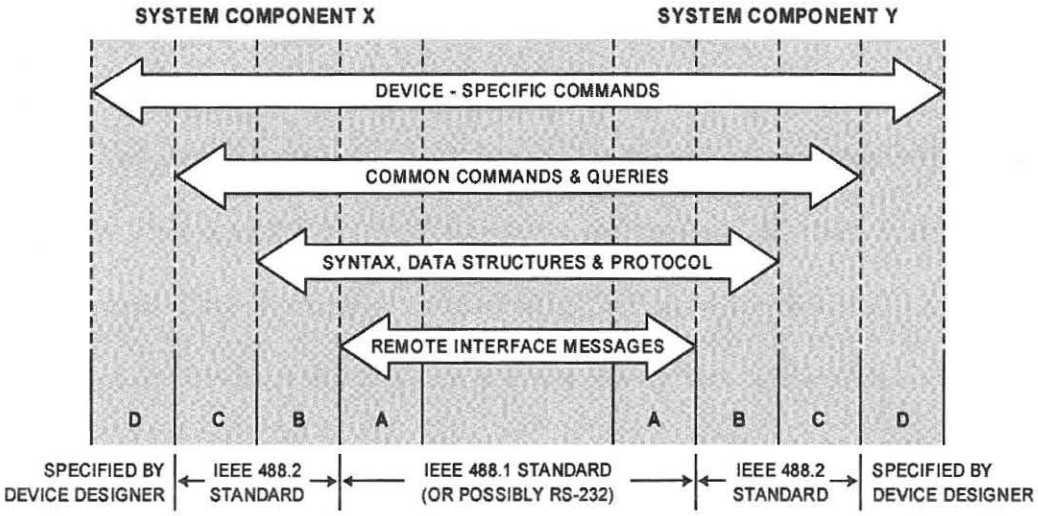


Figure 23 - IEEE 488.2 Standard showing the different layers the data is passed through.

Layer A represents the old IEEE 488 standard used to provide data exchange between a controller and one or more instruments. Layer B organizes the data into programme and response messages that represent a simple protocol to define a sequence of messages across a bus. Layer C provides a set of commands that are common to all devices on the same link or bus. Layer D concerns device-specific commands for implementation of device features. Therefore, it should be clear that the IEEE 488.2 does not replace the IEEE 488, but rather builds upon it.

Two methods of remote control are possible and will be described next:

- Remote control via RS232.
- Remote control via GPIB.

2.6.1.2 Remote control via RS232

The command queries and responses that are discussed in the IEEE 488.2 can be used on either RS232 or GPIB to communicate between a computer and a tester [36]. Before a computer can communicate with the tester, some parameters has to be set from the front panel of the tester. When the Fireberd 4000® or 6000 is used, the I/O DRIVER SELECT option must be changed to RC: 232. This tells the tester that a computer can control it via its RS232 port. Secondly, the communication parameters must be set. This signifies the baud rate, parity and data bits, at which communication will take place. This is normally set to 9600, 8, N (Speed is 9600 bits/s, data bits are eight and parity is set to none). These settings should also be used by the controller. The controller software is described in more detail in Chapter 4. The RS232 remote-control method was chosen for this research project, since it was already available and not as expensive as its GPIB counterpart.

The Fireberd 4000® tester can be put into remote-control mode by sending a CTRL-C on the serial port of the tester [40][41]. The tester will immediately respond with a ">" prompt. The greater-than sign signifies that the tester is in terminal mode. This means that everything that is sent to the tester's serial interface will be echoed back to the terminal. The software (GUI) described in Chapter 5 will make use of the tester's computer mode. In this mode, only the results will be echoed back to the terminal/host. All the commands that are available during remote-control mode are available in the tester's user manual [40][41].

2.6.1.3 Remote control via GPIB

This type of control is more difficult to set up, but it has the added benefit that up to fifteen devices can be connected to a single controller. On the GPIB bus, information can only be transferred in one direction. It is the responsibility of the controller/host to establish the source and destination of every transaction that takes place. Every device on the bus is given an address that is normally set on a DIP switch. For the controller to talk with a specific tester on the bus, it must know the physical address of each tester.

2.6.2 Test parameters

The operation of the tester is similar to the graphical interface discussed in Chapter 5. The only difference is that much more parameters are available to the user from the tester's front panel. Not all of the test parameters are always used. It was necessary to choose only some of the most frequently used parameters.

The parameters that are displayed in the tester software are the following [41]:

- Bit errors
- Block errors
- Pattern slips
- Pattern losses

- Blocks
- Receive frequency
- Efficiency
- Pattern synchronisation
- Clock present
- Frame synchronisation

A brief description of all of the above parameters is given next.

2.6.2.1 Bit errors

This is the number of errored bits that are counted since the test was started [40].

2.6.2.2 Block errors

Blocks are made up of numerous bits. When one or more bits in a block is errored, it will cause this parameter to count one block error. The amount of block errors that are shown is counted since the test was started [40].

2.6.2.3 Pattern slips

During the transfer of information, bits are sometimes added or deleted from the data stream. This parameter states the number of occurrences since the test was started. Pattern slips are generally caused by noise on the clock signal, missing or extra clock pulses and small differences in frequency between the data and the accompanying clock [41 p.2-101].

2.6.2.4 Pattern losses

This parameter counts when a complete loss of the receive pattern occurs [40].

2.6.2.5 Blocks

This is the number of complete blocks that were received since the test was started [40].

2.6.2.6 Receive frequency

This parameter states the frequency at which data is received from the RS449/V11 interface. This parameter is important in the sense that it indicates the user speed at which the circuit is working.

2.6.2.7 Efficiency

This parameter is derived from the G821 recommendation [8]. It gives the ratio of the number of available seconds in which no errors were detected to the total number of available seconds, expressed as a percentage. An acceptable value here is above 98% when single-bit errors occur with no pattern losses. As soon as continuous bit errors or pattern losses occur, the 98% threshold is invalid. At this point loops are applied on various portions of the circuit to determine the faulty portion. Follow the discussion on maintenance entities above for a better understanding.

2.6.2.8 Pattern sync

This parameter states that a tester is in synchronization with received pattern [40].

2.6.2.9 Clock present

This parameter goes hand in hand with the receive frequency parameter. As soon as the tester's interface synchronizes with the clock of the DCE, the receive frequency parameter will show the frequency or bitrate of the interface [40].

2.6.2.10 Frame sync

This parameter states that valid data frames are received from the circuit [40].

2.6.3 Interface tester

The interface tester is a simple device that can be connected between the DTE and the DCE. All the information is then sent and received as usual, making it possible to monitor the interface signal leads. Refer to the V24 and X21 recommendations that describe the logical order of the interface signals.

2.6.4 Protocol analyser

A Protocol analyser is a combination of a BER tester and an interface tester with an added benefit of analysing the raw-data stream. The raw information is processed into an understandable form and displayed on a VDU. The user can then extract information like the source and destination address, packet type, etc. This tester is usually used on the X25 exchange to monitor circuits and to determine user problems.

The protocol analyser is usually capable of monitoring levels 1, 2 and 3 of the OSI model.

To monitor the upper layers 4, 5, 6 and 7, the user will make use of a sniffer (Hewlett Packard, Internet Advisor) or a programme like NetXray® to analyse the raw data on a LAN network.

2.7 Summary

This chapter gave an overview of how digital circuits can be tested. It was shown that a circuit could be broken into portions while tracing a faulty location. The concept of circuit loops was discussed in detail to bring the reader to the understanding that remote testing of any circuit is possible. It was also shown that faults are reported onto a fault-management system that is capable of tracking numerous faults on a national basis. From this management system, the service provider could update his clients at any time on tests that are performed. Lastly, the most common test equipment that can be used while performing local and remote tests on any data communications circuit, were discussed.

Chapter 3

Test network design

This chapter discusses the basic operation and design of the access network that will connect the user to the remote-control test desk. Topics that will be covered include the following: synchronous and asynchronous data transfer, dial-up modem configuration, dedicated circuits, radio links, terminal servers, packet-assembler disassembler (PAD), TCP/IP, SLIP, X-terminals, etc.

3.1 Introduction

In order for computers to communicate, a medium must be established. The medium most commonly used is a modem. Two modems are used to form a complete circuit. Every circuit services a single entry point to the network. In order for the remote test desk to work, a few users must be allocated and trained to do after-hours testing. These users are supplied with a dedicated circuit running between the test centre and their homes. Access can also be gained by using a dial-up modem.

3.2 What is needed?

By using different products and technologies, limitations are added to the network, for example: when using radio links the speed are decreased, when using digital links synchronous to asynchronous converters must be used, etc.

When starting out a few users had to have access to the remote test desk. Thus, a few dedicated circuits were installed in each individual's home. As these circuits work in synchronous mode, a plan had to be devised to convert it to asynchronous mode for a personal computer's RS232 interface.

All the access circuits had to have a single point of entry to be able to connect to a single RTD or multiple RTDs (remote test desk).

3.3 Description of the present network design

Consider Figure 24. A division is made between the access network and the RTD (Remote Test Desk). The RTD only take up one RS232 interface. It is practically possible to connect the RTD directly to the RS232 port of any PC (personal computer), and to control it as if it is a local test desk. Because a lot of users must be able to gain access to the RTD, a device called a terminal server is implemented in the design of the test network.

Terminal servers have a wide range of capabilities. The previous server that was used, was a Digital® DEC 200/MC®. The current server that is used in the network is a Cisco 500/CS®. Up to 16 asynchronous ports are available for serial line connections. Each port can be set up as a service and/or access port. When a port is set up as a service, all the users connected to the server can connect to this service port by typing in the name of the service at the command line. A typical example of a service port is the connection to the RTD. When a user wants to connect to the RTD, the name of the service is typed in at the server local prompt. Only one connection to a service is allowed. Another user can only access the service when it is disconnected.

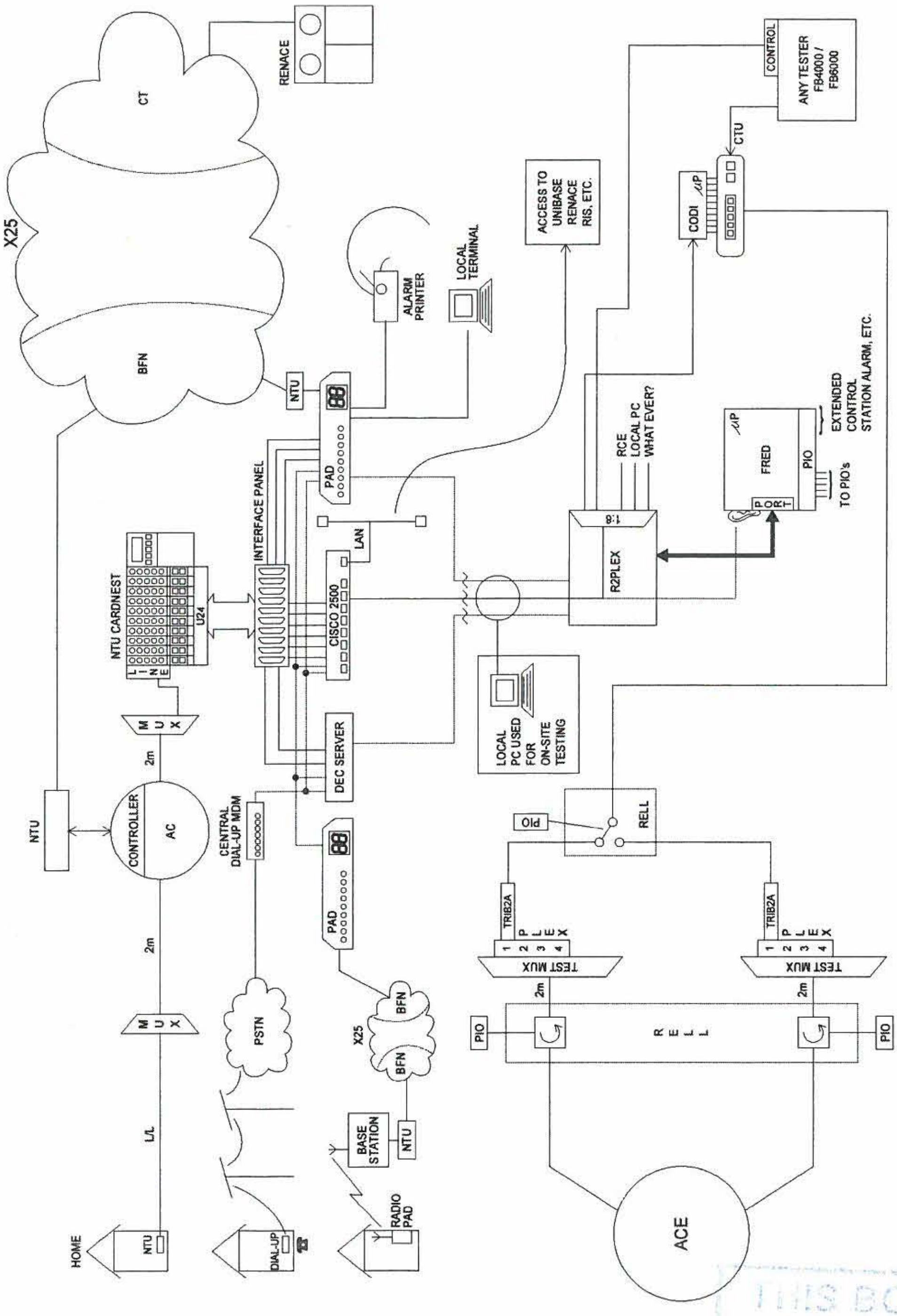


Figure 24 - Present remote-testing facility. Refer also to Appendix A.

Every port on the server is also capable of serving as a dial-up port. The port can be configured as dial-in or dial-out. When a port is set up as dial-in, connections are allowed to anyone using a dial-up modem phoning into the terminal server. When a dial-out line is configured, anyone connected to the server can dial out to any other computer. For the purpose of the test desk only dial-in connections are allowed.

All the user ports in the system are configured for 19200bit/s. The maximum speed the port can operate at on the Cisco 500/CS® is 38400bit/s. Because synchronous lines are used, the maximum speed an asynchronous to synchronous converter can operate at is 19200bit/s. The synchronous lines are capable of running 64kbit/s synchronous. To use this speed a router must be installed at both sides, which is very expensive. For the trial run of the RTD it was decided to keep the costs down. The costs of the latter are astronomical when compared with a normal asynchronous line.

Any port on the server can also be configured as a TCP/IP (Transmission Control Protocol / Internet Protocol) port running SLIP or PPP for X-Window sessions. The Unibase® fault system runs on a UNIX operating system. The UNIX system serves X-terminals (not a normal PC). In order for this fault system to run on a normal PC with Windows 3.1® or Windows 95®, it must be able to communicate with the UNIX server. The port on the server is configured to run SLIP or PPP to enable TCP/IP communication to the Unix server. On the client's computer, an application like Trumpet Winsock® is used to adapt the asynchronous serial line interface to the application communicating with the Unix server. By using this solution, it is possible to work on the fault system although it is slow over a serial line.

Connection to the RCE/RENACE system is done by executing a telnet session at the local server prompt. The connection is established by also providing the telnet programme with the desired IP address of the host to be contacted. The IP address has, for example, the following form: "196.25.1.1". When a connection is successful, the user will be prompted with a logon screen where a user name and password must be typed in. This connection is also possible from the X-terminal application discussed above.

By connecting a PAD to the server, a third way of connecting to the RCE can be accomplished. It is done in the following way: a service is created on the server. The user connects to this service and presses the return button. The PAD (prompts from different PAD's may differ) will respond with a "* NO CIRCUIT" prompt. The user then types in the host's desired DTE address. This address has the format "C151080999". The host will respond with an appropriate logon screen. This connection can be done with a normal dumb terminal (VT100, VT220, VT50 etc.), since the pad provides the end-user with multiple asynchronous ports.

3.4 The access network

The purpose of the access network is to connect the remote user to the RTD. However, by using a single point of entry device, many more benefits are added to the test system. This single point of entry devices is more commonly known as terminal servers.

To gain access to the terminal server, several mediums can be used. The following list of access mediums will be discussed in detail:

- Dedicated medium.
- Dial-up medium.
- Radio PAD.

The following Figure 25 shows the access network.

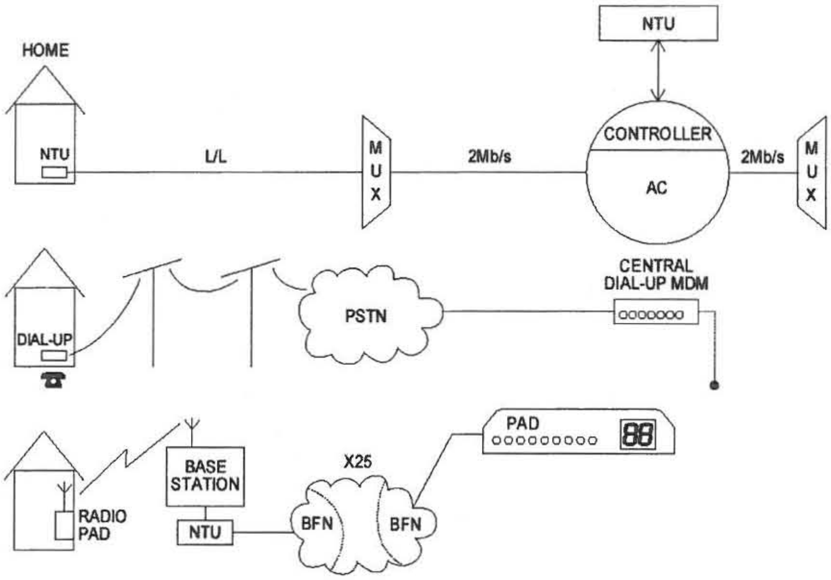


Figure 25 - Access network showing the different methods that can be used to gain access to the network.

3.4.1 The dedicated medium

Access to the terminal server on a dedicated circuit is actually very simple. The circuit can be either digital or analogue. There is, however, one problem when using dedicated circuits, and that is the interface is of a synchronous nature. One way to overcome this problem is to connect asynchronous-to-synchronous devices between the DCE and the DTE on both sides. Another way is to do a kind of oversampling. The DCE's clockrate is set to at least three times the speed required by the DTE. Example: When 2400bit/s is required by the DTE, the DCE speed would be set to 9600bit/s. In this case 9600bit/s is four times the speed required. The explanation for this is that it is not possible to configure the DCE for this speed. Thus, the next increment is chosen, which is 9600bit/s. Newer DCE's will solve this problem, as both asynchronous and synchronous methods are available.

Finally the user communicates with the remote end by loading a programme like Windows 3.1® Terminal on a personal computer. First the communication settings are changed to the desired speed with additional parameters like: none parity, eight data bits and one-stop bit. By just typing on the keyboard, the user can connect to various services available to the server. Example: To connect to the RTD, the user would type the service name of the port. If the port to which the RTD is connected, is not busy (in use by another user), it will connect the service to the requestor.

3.4.2 The dial-up medium

The dial-up medium works the same as the dedicated network once the connection with the remote modem has been established. The connection procedure would be somewhat different from the dedicated circuit. First, a programme like the Windows 3.1® Terminal is loaded. Again, the communications parameters are set according to the user's needs. If no connection was established yet, the user would still be connected to the dial-up modem's local prompt. The modem can be given commands through the terminal software to instruct it what to do. The modem can also be given some initialisation commands that will instruct the modem to behave in a certain fashion. These commands are known as AT commands.

Here is a list of a few commands that are necessary for the modem to work properly for the purpose of this research:

- ATDT - Attention dial tone.
- AT&C0 - Permanent CD.
- AT&D0 - Ignore changes in the DTR signal.

- AT\K4 - Stay connected on a break signal.
- AT&W - Stores the current profile.
- AT&V - Displays the current modem profiles.
- ATS0=1 - Causes modem to automatically answer after first ring.

To obtain more information on AT commands, the specific modem user's guide should be consulted. This is because not all the modems are Hayes compatible, and to accomplish the same task could require some different commands. The above commands are valid for the Creative Blaster® 28.8 External, Philips® F-1128V and the Dynalink® 1414VQE used in the current access network.

In order to command the modem to phone the remote modem, the following command will be given:

ATDT 0,0514211111 followed by a carriage return.

The breakdown of the command is as follows:

- AT - Tells the modem it is an attention command.
- DT - The modem must use tone dial.
- 0 - Dial 0 to gain access to an outside line when on a switchboard.
- , - Wait for outside dial tone.
- 051 - Area code for city where the remote modem is located.
- 4211111 - The number to dial to get to the remote modem.

After the command has been entered, the user will hear the modem dialing the remote side. When the remote modem which is set to auto answer mode, answers the call, the user will hear numerous random noises. This is the modem handshaking and training sequences going back and forth. Here the modem negotiates for an acceptable line speed according to the line grade. After the handshaking is completed, the user will be given a response on the terminal, for example: "CONNECT 19200". This would mean that the modem successfully negotiated a line speed of 19200bit/s with the remote modem.

With compression techniques and buffers available in every modern modem, it is possible to set the interface speed to 57600 bit/s second. The terminal (Windows 95® Hyper terminal®) would in this last case be set to communicate at 57600 bit/s. This will also force the remote modem to adapt its port speed to 57600 bit/s. If the line rate was 19200 bit/s, the modem would have to do quite a lot of compression to keep both DTE's satisfied. When this happens, the user should enable either hardware or software flow control to stop the DTE's from sending too much information to the modem. This means that the throughput of information could be far less than the actual port speed of the modem, depending on the compression used by the modem. To enable MNP5 compression, the AT%CX command can be used, where X is in the range of 0 to 4.

From here on the connection to the service on the terminal server would be the same as before.

3.4.3 Radio PAD

This medium, although feasible, is very slow, and thus it will not be tested because of bandwidth requirements of the current network. Until newer technology becomes available, the connection procedure will be discarded from this document. At present, this technology is capable of a reliable connection with a continuous speed of up to 2400bit/s. Software like X-Vision® requires at least 19200bit/s to run X-terminal emulation on a remote computer.

3.5 Point of entry

The point of entry would be a device that is capable of sharing network services to a user that wishes to gain access to the RTD. The device should be secure enough to restrict unwanted guests to the network. This is of utmost importance, as people like hackers who gain access to the test network, can do great harm to the service provider. A person with some inside information could go as far as disconnecting circuits. In general, such a person could cause havoc.

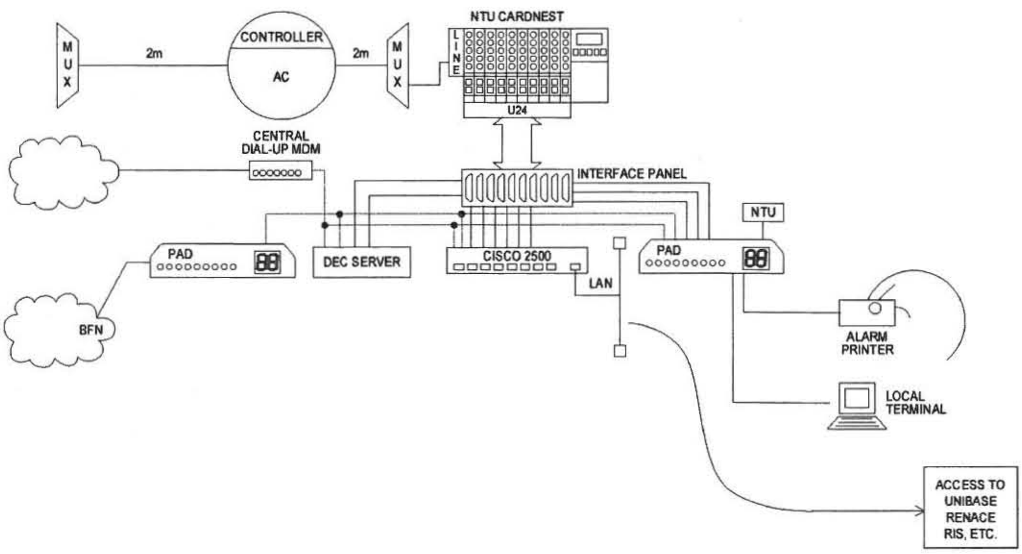


Figure 26 - Point of entry. This is where all the access circuits are connected to provide a simple method by which the users can connect to the remote test desk.

Figure 26 shows some devices that can be used as entry devices. Some of the devices, like the PAD, do not lend itself to be very secure. The DEC Server 200/MC® gives some restrictions, but ultimately the CISCO® 2511 series can have the greatest advantage, as it is capable of restricting access according to an access list that resides in the router's memory. Unfortunately during the period of this research project, only a CISCO® 500/CS could be obtained to develop the network.

When the CISCO® terminal servers are used it is possible to provide network services like TELNET, HTTP, FTP, IPX, TCP/IP etc. This enables the remote user (by using the PPP protocol) to access the Unibase® fault system, and to access an Intranet server where for instance a network-monitoring system is logging 2Mbit/s alarms.

3.6 Summary

This chapter gave a description of the access network that is currently used for the purpose of this research. It also gave a general description of how to gain access to the network and how to connect a remote desktop computer to some network services needed to do some tests. The user was also brought to the understanding of the differences between a dedicated connection and a dial-up connection. Furthermore, some security issues where also discussed.

Chapter 4

Hardware design

This chapter deals with the design of the hardware of the remote test desk (RTD). The design consists of a few devices that make up the whole RTD. Every device will be dealt with in turn. This is necessary, because at the time the RTD was designed, it was only necessary to apply the design to a two ACE test centre. However, as the popularity of the RTD grew, it became a necessity to apply the existing design to a multiple ACE test centre. Because of the evolving nature of this research project, the reader will never be fully informed about the status of the hardware or software research. Some outlines on future study is given in Chapter 7.

4.1 Introduction

The previous chapter dealt with the access network and how any user can connect the RTD. This chapter explains how the RTD can be connected to the access network and how it can be accessible to the remote user. The whole basis for the hardware design is to be able to communicate to any device by means of asynchronous communication methods. To achieve this, the whole RTD is designed around a microcontroller called FRED. The way in which the remote user can interact with the RTD is via its RS232 UART situated in the Intel® 80C32 microcontroller. This forms the heart of the whole RTD. Because the system can be controlled via the RS232 interface, the system can be directly connected to the CISCO® 500/CS terminal server. Figure 27 shows what this connection looks like.

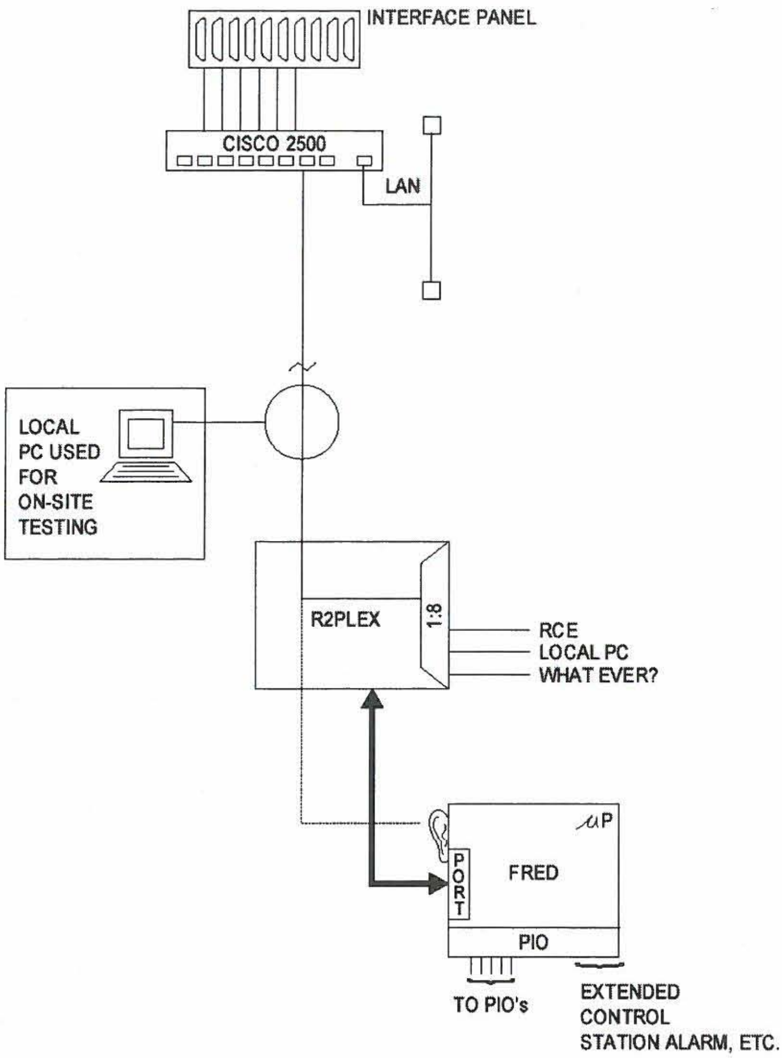


Figure 27 - this figure shows the RS232 Connection to the microcontroller. FRED acts as a listening device to divert any incomming messages on the R2PLEX unit. Fred is also responsible for switching the PIO leads that control the RELL unit.

From the block diagram it can be seen that the FRED and R2PLEX boards form an integral part. The FRED controller can operate as a stand-alone unit, but this is not the case with the R2PLEX device. FRED directly controls R2PLEX and acts as a listening device. Upon receiving a unique character sequence, it will switch to local mode. In this mode, it can be commanded to manipulate R2PLEX or change the outputs of the PIO. Upon completing all the necessary commands, FRED is switched back to listen mode.

The block diagram also illustrates two methods of communicating with the RTD. One is via the CISCO® Terminal server, and the second is via a local personal computer. This is done to build flexibility into the system. During the day, the RTD can be used as local test desk, and during the night it can be operated as a remote test desk. Test equipment costs quite a lot of money. Every piece of equipment that can only be used at certain times of the day is wasted. By adapting the system to be a remote and a local test desk adds more feasibility to cost implications. In the end, the system is financially more worthwhile.

4.2 Hardware description of the RTD

Figure 28 shows a block diagram of the complete RTD as it can be controlled from a local host computer. Five main hardware components can be identified:

- FRED
- R2PLEX
- PLEX
- RELL
- CODI

The RTD is controlled via a single RS232 port that is situated on FRED. FRED in turn controls RELL, PLEX and R2PLEX. CODI acts as a separate device that can be individually controlled by an RS232 device. All other devices that need to be controlled from the local host are connected to R2PLEX. These devices could range from dumb terminal connections to additional test equipment or RTDs.

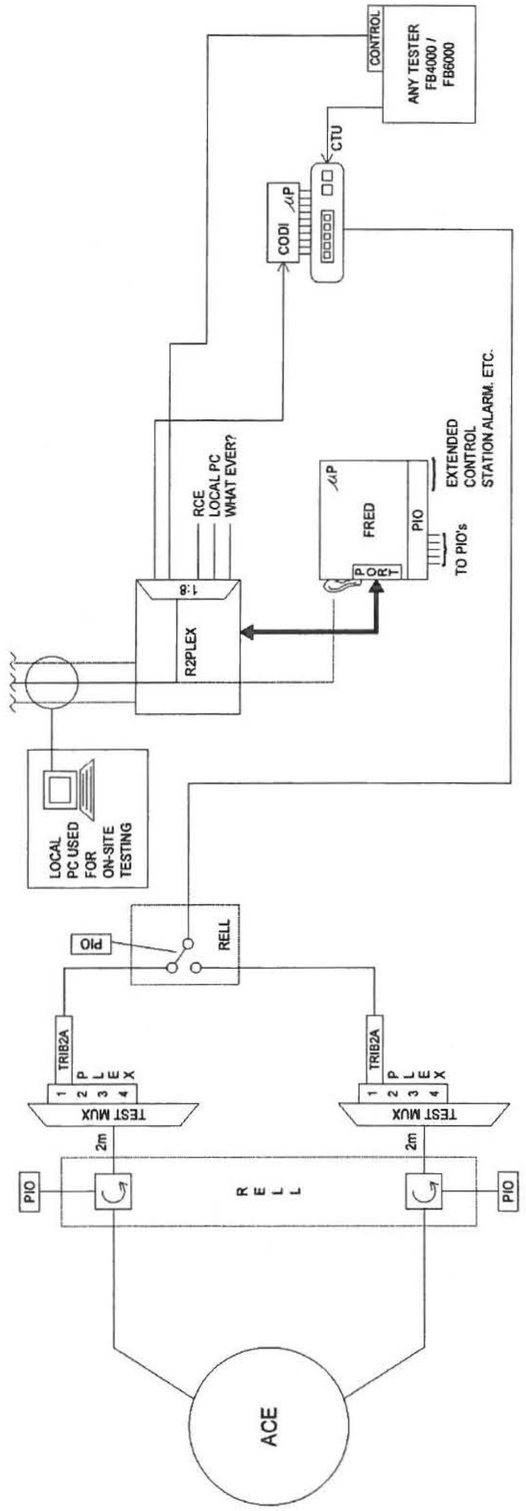


Figure 28 – Full-block diagram of the hardware of the remote test desk showing how all the parts are integrated.

4.2.1 FRED

FRED consists of an Intel® based 80C32 microcontroller with 32K RAM and 32K ROM. It controls its environment with an 8255 PIO. The following shows a block diagram of FRED.

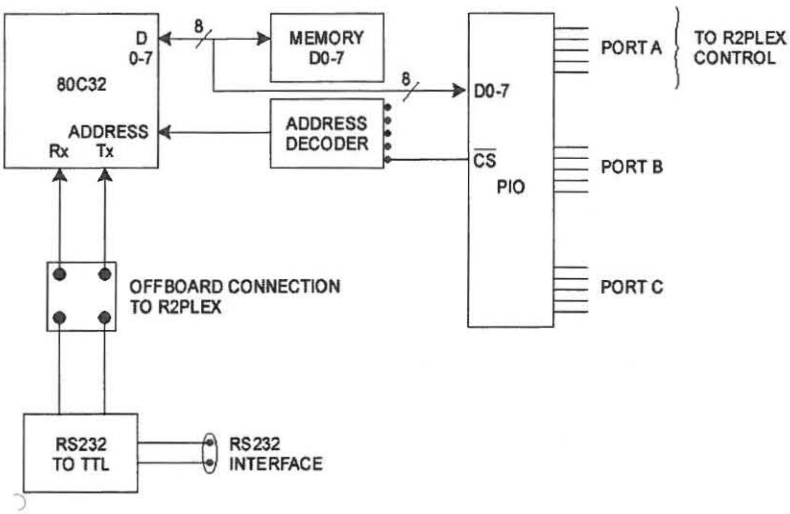


Figure 29 - Block diagram of the FRED controller hardware showing the processor, RS232 interface and PIO ports.

The software of the controller is designed to operate in two modes:

- Local mode.
- Listen mode.

4.2.1.1 Local mode

In local mode the controller is capable of excepting commands through its serial port. These commands can be either typed from a dumb terminal like Windows 3.1® Terminal, or it can be sent from the graphical interface described in Chapter 5.

The following commands can be issued to FRED in local mode:

- M<CR>
- I<CR>
- O<CR>
- H<CR>
- E<CR>

a) M<CR>

This command controls the R2PLEX. It prompts for a new port number on the R2PLEX. Valid port numbers are <1 to 8>. The command is finished with a <CR>. After accepting this command, the controller will activate the new port on R2PLEX and automatically switch to listen-mode.

b) I<CR>

This command instructs FRED to read the binary value of the specified PIO port. Valid PIO port numbers are <1 to 3>. The output of this command will be in decimal, ranging from <0 to 255>.

c) O<CR>

This command instructs FRED to enable or disable a specific PIO port pin. Valid PIO port numbers are <1 to 3>. Valid port pin numbers are <1 to 8>. This command will not output anything to the RS232 serial line. To check the status of any PIO ports, the I-command is used.

d) H<CR>

This command shows the list of commands available to the user when operating from a dumb terminal.

e) E<CR>

This command exits FRED to listen mode. To get to local mode again, a password sequence should be sent to the serial interface either from the dumb terminal or the control software.

4.2.1.2 Listen mode

This mode causes FRED to ignore all incoming data until a specific sequence of characters is received. This sequence is known as the password-sequence and presently has the following form:

"y <delay> H <delay> n <delay> , <delay> . <delay> / <delay> 7"

This ensures that, while text or binary information is transferred from the host to any R2PLEX port, FRED will not switch to local-mode. The delay between the characters ensures that the host really wants to communicate with FRED. The delays are assumed to be about 50 milliseconds. After completing the correct sequence of characters, the controller will switch over to local mode. A banner will be displayed looking something like this:

```
Telkom SA Ltd

FRED Micro controller 1996

>
```

Figure 30 - FREDs boot prompt. This message is seen every time the power is cycled on the unit.

The operating system for FRED was written in BXC-51®, which is a BASIC cross compiler. The source code is written in a normal text editor on an IBM® compatible computer, and then compiled with optional parameters from the command line of Microsoft® MS-DOS® prompt.

The following figure shows a flow diagram of the FRED operating system:

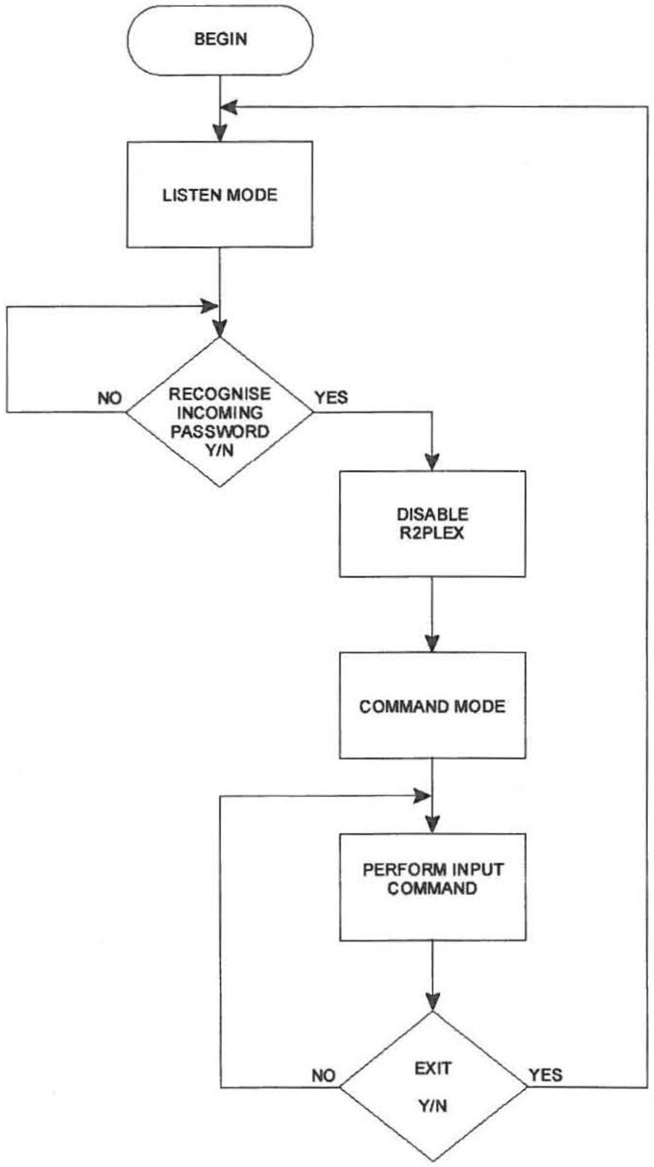


Figure 31 - FREDs operating system (OS) flow diagram. This diagram shows how the operating software of the microcontroller functions.

The following figure shows the actual circuit diagram of the FRED microcontroller that is currently being implemented in the RTD (Remote Test Desk).

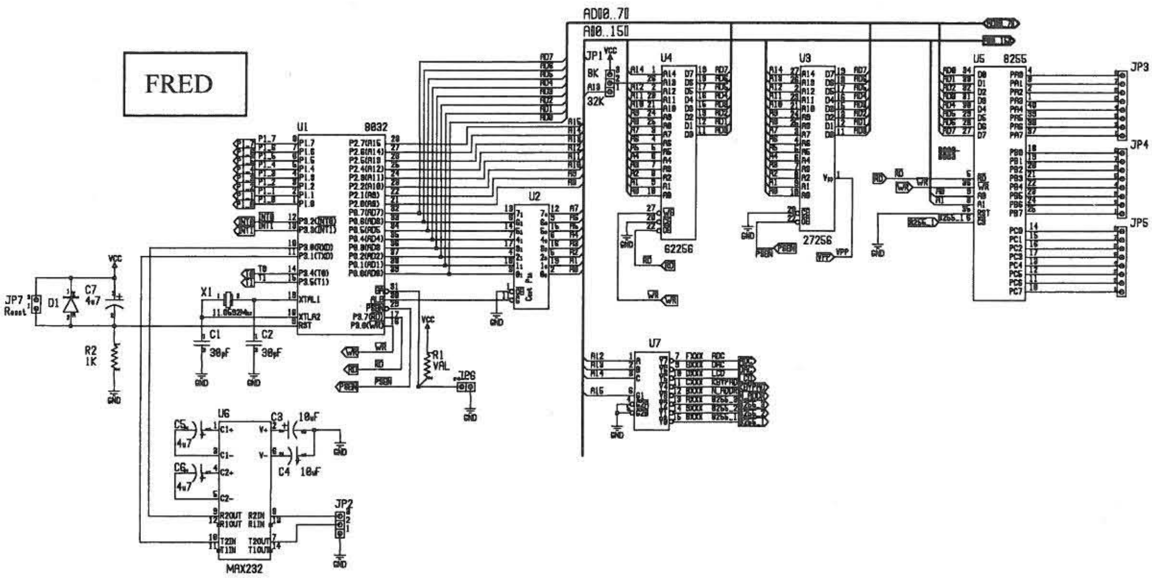


Figure 32 - FRED Microcontroller circuit diagram. This diagram is the complete wiring diagram as implemented in the printed circuit board.

4.2.2 R2PLEX

The operation of R2PLEX is based on two integrated circuits. The one IC is responsible for multiplexing the incoming data to the desired FRED selected port, and the other demultiplexes the data coming from the same port. The following figure shows a block diagram of the principle of the working of R2PLEX.

From the block diagram can be seen that information coming (data transmitted) from the host is multiplexed to the selected device. Responses and information coming back from the selected device is demultiplexed to the host. During this whole procedure, it is clear that FRED only listens to the data coming from the host.

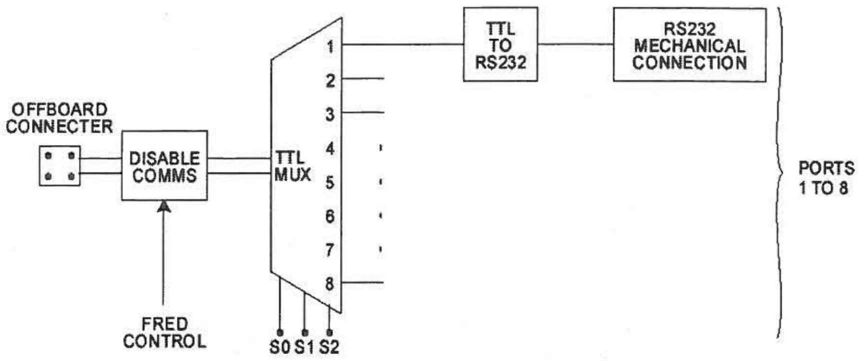


Figure 33 - R2PLEX Block diagram showing that in principle it is a multiplexer switching the input to any of eight outputs.

It must be added that R2PLEX is designed for simplicity. That is why only three-wire communication is employed in the design. That is transmit data (TD), receive data (RD) and ground (GND). No hardware flow control is switched through from the host to any port. To enable some flow control through R2PLEX, it is suggested to make use of software flow control (XON/XOFF). Refer to appendix B for interface pins needed on DB-9 and DB-25 connectors.

The R2PLEX conforms to RS232 and V28 electrical signaling recommendations by employing Maxim's® MAX232 integrated circuits on its input and output ports. The following figure shows a picture of R2PLEX and FRED incorporated into a standard IBM-PC® compatible desktop housing.

Power for the system is obtained from the desktop box. The box also supplies power to external peripherals like the RELL board.

Central University of Technology, Free State

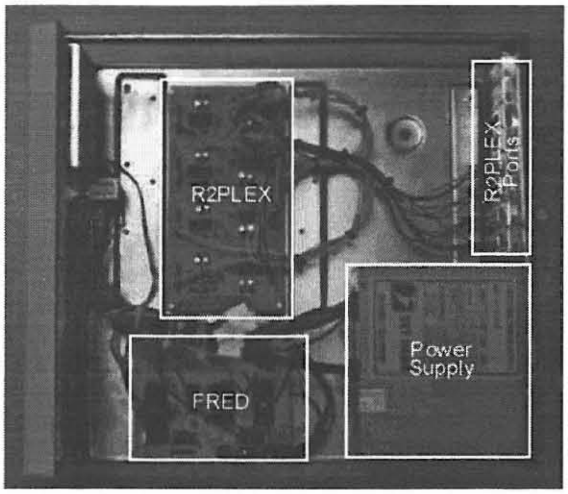


Figure 34 - FRED and R2PLEX in one box. All of the printed circuit boards are housed inside a personal computer desktop case. The R2PLEX ports are the locations where all outgoing RS232 devices are connected.

The next figure shows a view from the side. R2PLEX consists of two printed circuit boards. The board on top handles all the conversions from RS232 to TTL, and the bottom board handles the multiplexing operation, which is selected by FRED.

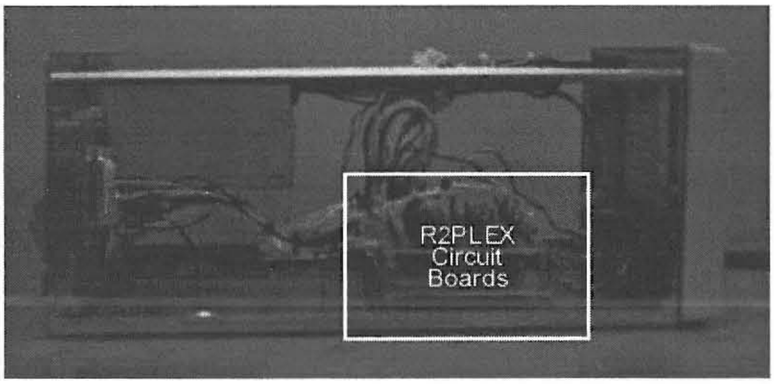


Figure 35 - R2PLEX Viewed from the side.

The next figure shows a view from the back exposing all the R2PLEX port connectors. The PIO port pins are connected onto a single DB-25 connector next to one of the R2PLEX ports. The current R2PLEX is wired to supply four RS232 devices.

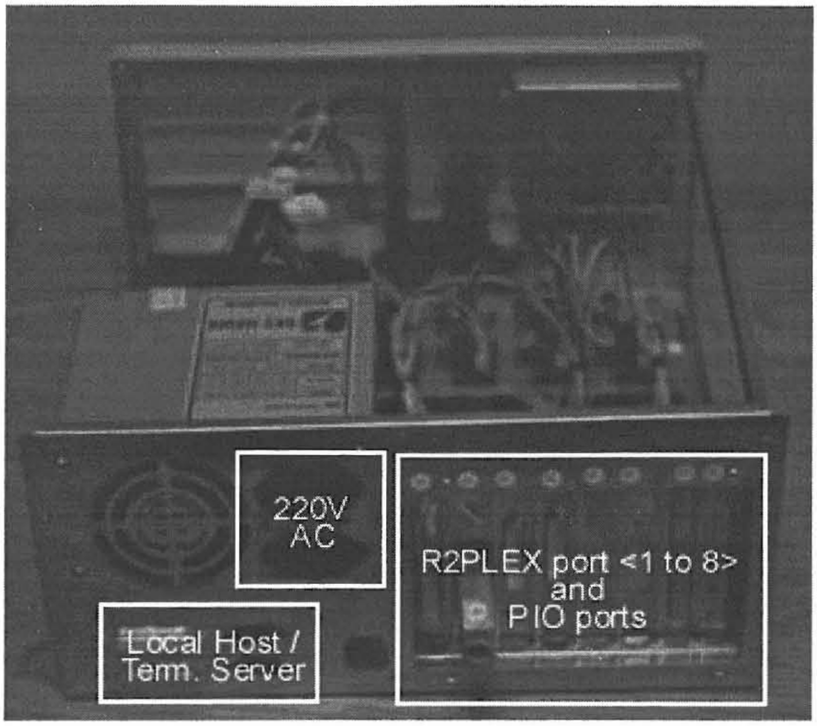


Figure 36 - Controller box viewed from the back showing the ports where the power is connected and where the RS232 devices of R2PLEX are connected. The local host connection is where a personal computer running the graphical interface is connected to. This port can also be connected to the point of access, making it possible to be controlled remotely.

The unit is supplied with 220V AC power that can be plugged into the back of the desktop box. The figure also shows the connection for the local host, which can be connected to the CISCO® 500/CS or 2511 terminal server. The pinouts for the RJ-45 connectors on the terminal servers can be found in the server hardware & installation manual. Here again only 3-wire serial communication is used.

The next figure shows the circuit diagram of the R2PLEX unit that is currently used in the RTD.

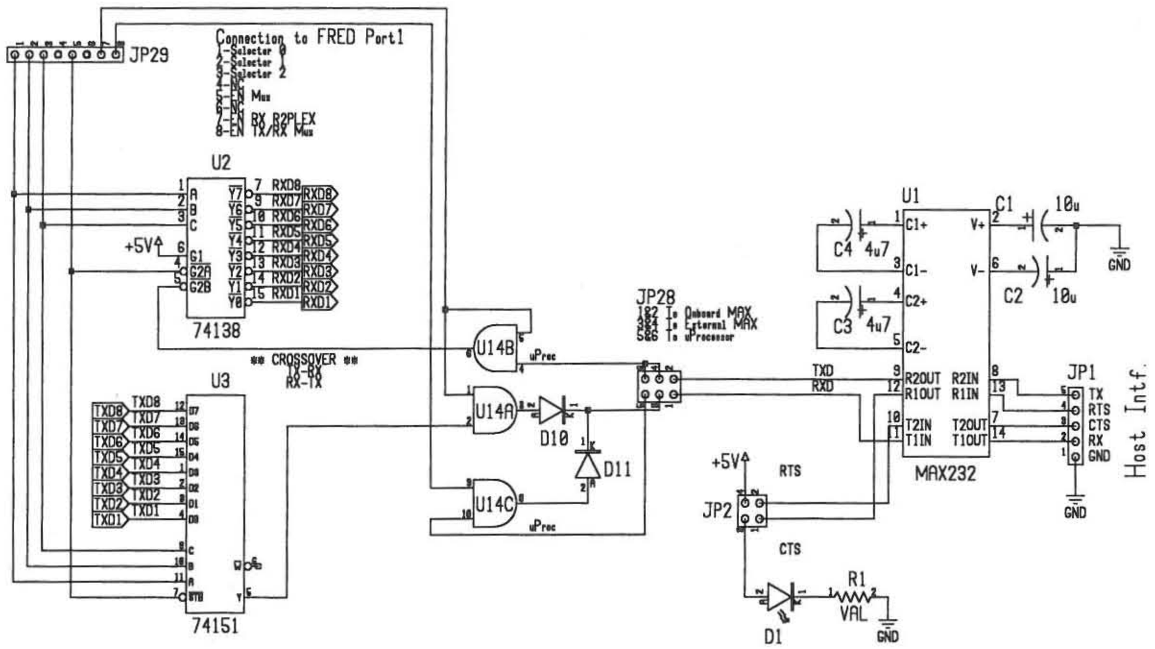


Figure 37 - Circuit diagram of R2PLEX as it is implemented on the printed circuit board.

4.2.3 PLEX[2]

The PLEX board operates similar to the R2PLEX board in the sense that data is multiplexed and demultiplexed. The difference is that the board does not convert any electrical signals, and it switches data streams that are to be tested by the TTC® Fireberd® tester. This board is controlled by two TTL compatible signal leads from FREDs, PIO interface port. From Appendix A can be seen that two PLEX boards are needed for the current RTD. To be able to interface the PLEX board to the CTU, a Marconi® TRIB2A card is needed to do a conversion from TTL to balanced CO-DIRECTIONAL (HDB3 64Kbit/s) signaling. The following figure of a block diagram illustrates the working principle of PLEX.

[2] Hardware designed and built by Mr Danie de Bruyn, Telkom SA.

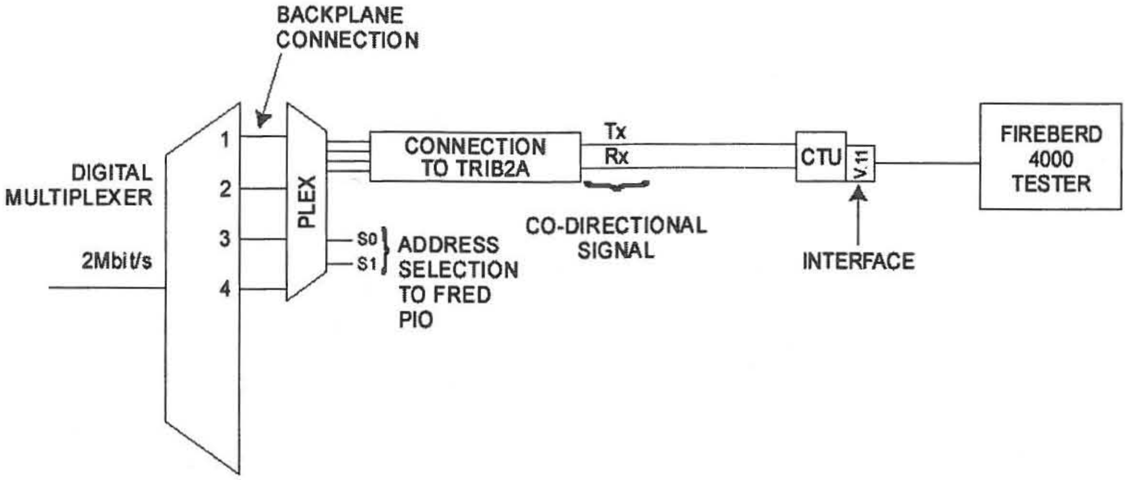


Figure 38 - Block diagram of PLEX. The PLEX unit acts as a multiplexer diverting the CTU and test equipment to the correct timeslot where a circuit under test is connected using the SPLIT command on the ACE.

The next figure shows the back of PLEX that connects to the backplane of the EDM 6003 multiplexer (which is a normal Diginet multiplexer).

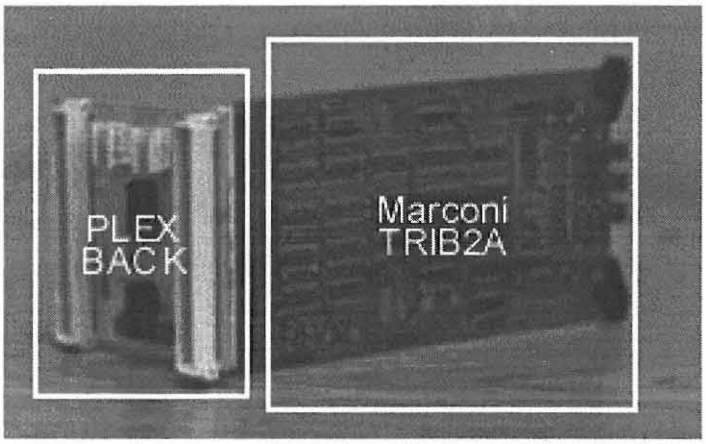


Figure 39 - PLEX viewed from the back. This shows the connectors which connect to the test multiplexer's backplane.

The next figure shows PLEX from the front.

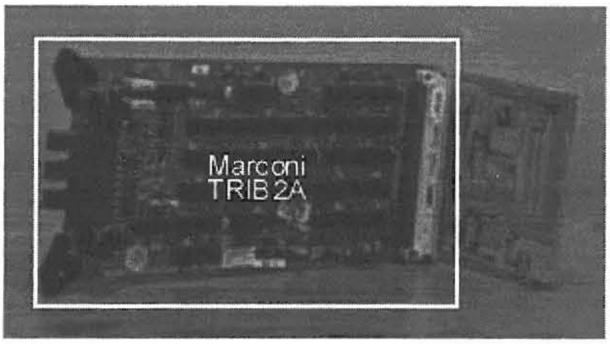


Figure 40 - PLEX from the front. The TRIB2A card is a co-directional interface operating at 64Kbit/s. The transmit and receive wires of the card are connected to the CTU. The CTU uses this card to test any circuit speed that is currently connected to the test multiplexer.

The figure below shows how multiple PLEX units are fitted to the backplane of the EDM 6003 multiplexer. The EDM 6003 acts as a test multiplexer. For the RTD discussed in this document, two of these EDM 6003s are required to do the testing of the data circuits. The relay board, RELL, switches between the two multiplexers that are separately connected to an individual ACE.

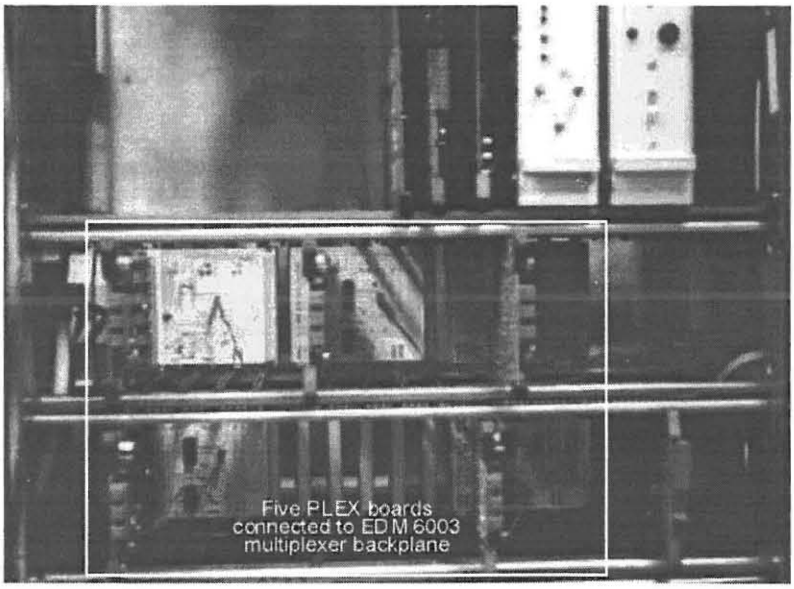


Figure 41 - Multiple PLEX modules connected to the Diginet multiplexer backplane. The Diginet multiplexer provides 31 timeslots that can be used for test purposes.

4.2.4 RELL

The main function of RELL is to easily switch any type of electrical signal. This device not only switches between the two multiplexers, but also provides a looping facility towards the test equipment. This is a good way of doing local diagnostics of test equipment on site. This is just a means which the technician (tester) can use to control the credibility of equipment used to do tests.

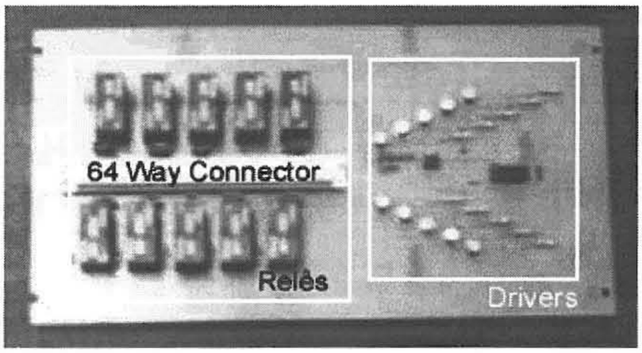


Figure 42 - This figure shows the RELL relay board. The 64-way connector connects all the 2Mbit/s channels to the onboard relays. The drivers are connected to FREDs PIO ports from where it can be controlled.

4.2.5 CODI

The CODI unit consists of a Dallas Semiconductor's DS5000T® microcontroller, which interfaces directly with the main board of the CTU. Sixteen input-output lines interface directly with CTU to manipulate the speed and loop circuitry of the main board. The unit is controlled with a serial interface by the CODI software discussed in Chapter 5. To make this possible, CODI's RS232 interface is connected to R2PLEX. The modification is only a temporary solution, since Alcatel Altech Telecom® has incorporated this feature into their new release of the CTU hardware.

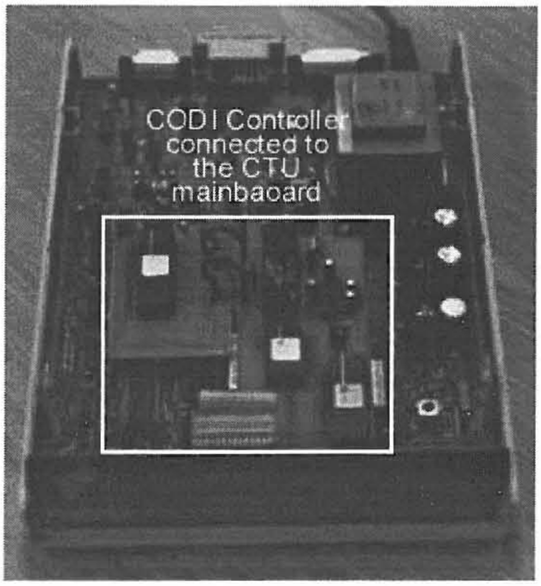


Figure 43 - CODI[3] controller modules connected to the CTU. The CODI controller is a DS5000 processor with an RS232 interface that is connected to the R2PLEX outputs.

Evaluation tests have been performed on the new CTU and proved to be very adaptable for future use, since a BER tester is already built into the device and the user interface is menu-based running a VT100 terminal emulation.

4.3 Summary

This chapter discussed how the hardware of the remote desk is configured so that it is possible for the user to interface with. An explanation was also given on the software necessary to control FRED, which in turn controls the rest of the remote test desk. It also became evident that without a microcontroller, the hardware interface to the test equipment would not be possible. The next chapter will discuss how the Graphical User Interface communicates with the hardware. Without a graphical interface, the RTD would be a great inconvenience to the person testing a faulty circuit.

[3] Hardware designed and built by Mr Willem Coetzee, Telkom SA.

Chapter 5

Software Operation and Design

This chapter describes all the software that is to be used during a circuit test procedure. Firstly, a basic description is given of how the tester software works internally with the aid of some flowcharts. Secondly, the directory structure and file locations are discussed. Thirdly, the basic operation of both the setup and the test programme is discussed. An example of a circuit test procedure is also given.

5.1 Internal software description

Both the TESS and the FB4000 programmes are written in C++ and compiled with Microsoft® Visual C++ 1.5®. All the software developed for this research is capable of running on the Windows 3.1® platform and the Windows 95® platform.

The TESS programme will not be discussed in detail, as it only writes dialog parameters to the hard disk without taking any decisions worth displaying in a flowchart.

The source code is based on OOP (Object Oriented Programming), which is generally used when programming applications for Microsoft® Windows® platforms. This means that every control (button, check box, radio button etc.) on the dialog box will respond on a mouse click or when information is fed into a dialog box etc. A control can also be responsible for displaying information to the user by displaying text, bitmap, etc. Every control is given a member variable through which it communicates with the user and other controls. Example: When the disconnect check box is pressed, the value of its member variable is changed from TRUE to FALSE or FALSE to TRUE, depending on its previous state. This change goes hand in hand with a click of the mouse button. To tell the programme that the disconnect check box has been pressed, an event is triggered in the main programme loop telling it to do some programme steps related to the event and the object involved. The programme will then jump to the related function and execute the task. The functions that are related to all the objects on the tester dialog box will be discussed next.

When the FB4000 programme initially loads, it reads the configuration from the C:\TESS3\FB4000.DAT file into its memory. This will update the variables used in the program. Variables that are of great significance are the COM port, the connection type and PLEX port variables. Upon loading the configuration, the text on the PIO PLEX group's buttons will be set to the values as set in the TESS setup programme.

5.1.1 Events

The main events that can occur during the programme lifetime will be discussed next, namely:

- Click disconnect.
- OnTimer.

5.1.1.1 Click disconnect.

To start the programme after loading, the Disconnect check box's value must be changed to FALSE. This is accomplished by a mouse click. The following flowchart in Figure 44 shows the start-up sequence that is taken by the programme.

If the value of the check box is FALSE, it will open the COM port of the host PC with the desired settings in the FB4000.DAT file.

If the value of the check box is TRUE, it will change the icon next to the "disc" check box to an icon that shows that the connection is disconnected. After completing the above action, the programme will close the host's COM port.

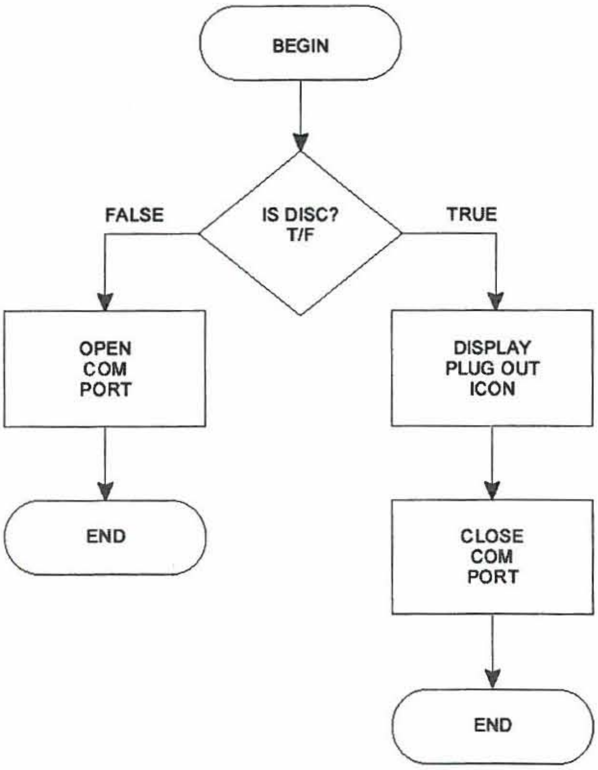


Figure 44 - Flowchart 1: Click on disconnect. This chart shows what happens when the user performs a mouse click on the "disc" check box in the graphical interface. See appendix E.

The following functions are called by this event:

- Open COM port.
- Close COM port and
- Plug out icon (not discussed).

5.1.1.2 Open COM port

This function consists of two parts: The first function opens the COM port, while the second function, on succession of the first, will start communicating with the devices it is connected to.

First, the COM port is opened with the desired parameters. If an error occurs, it will be displayed in a hidden text box on the main dialog box. This hidden text box can be accessed by clicking on the void check box beneath the disconnect check box. The following figure shows a flowchart of what is happening.

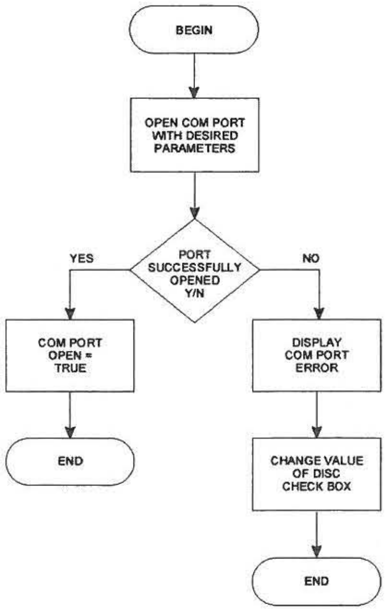


Figure 45 - Flowchart 2: Open COM port. This chart shows all the actions that are performed by the software while opening the personal computer's serial interface.

If no error occurs, a message of success is written to the hidden text box and execution of the second function is commenced. The following figure shows the second phase of the connection, which is to determine if the host is connected to the R2PLEX device, and eventually if a tester is connected.

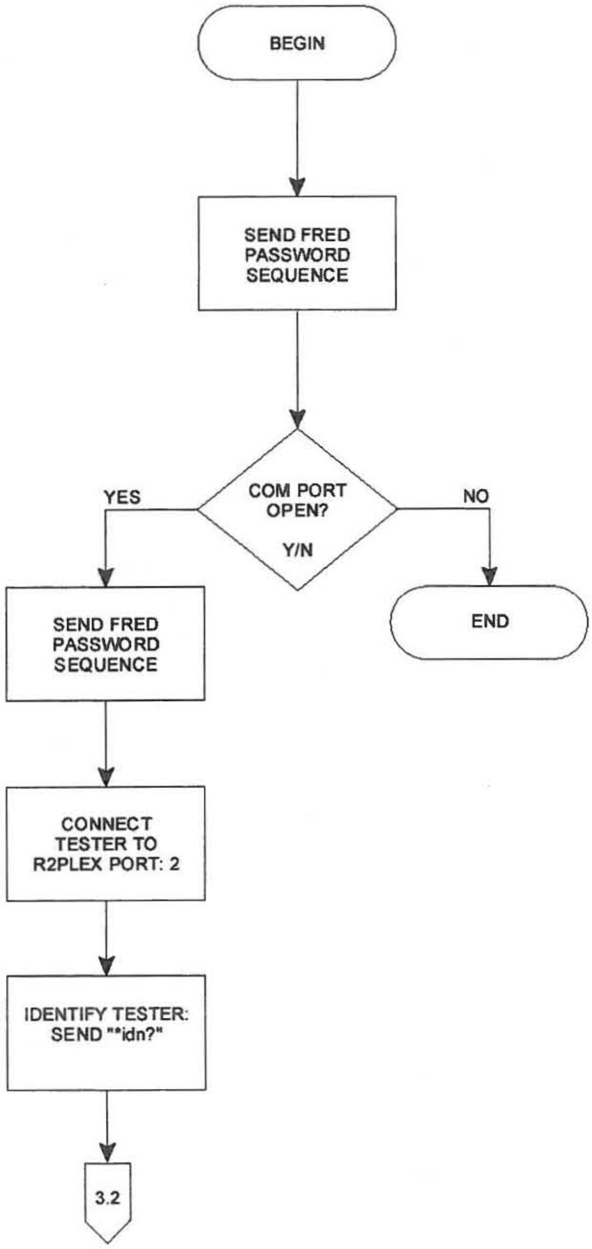


Figure 46 - Flowchart 3.1: Is the communications port open? As soon as the COM is opened, the software communicates with FRED's controller instructing it to divert R2PLEX to the tester.

This portion transmits commands over the open COM port to the FRED microcontroller. The microcontroller will activate the appropriate RS232 switch path to the test equipment. The host will now send commands to the tester to determine if it is connected to R2PLEX or not. The programme will commence execution at label 3.2 if the COM port open variable is equal to TRUE.

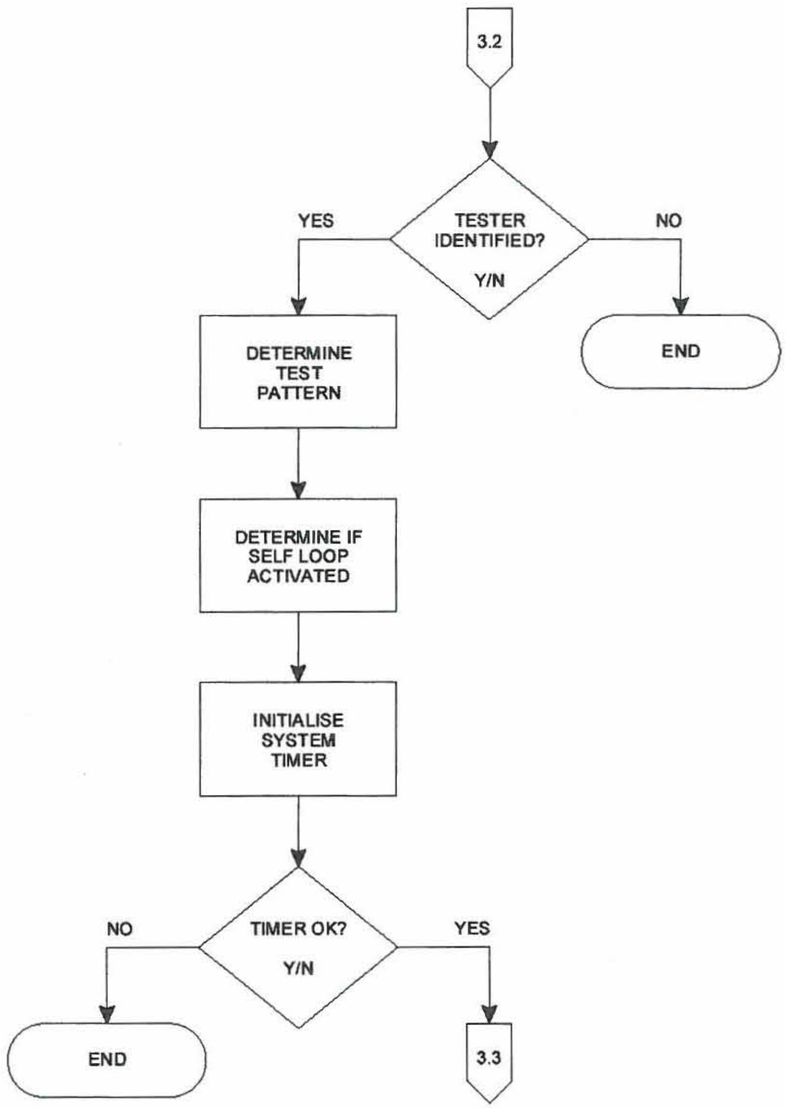


Figure 47 - Flowchart 3.2: Connection sequence. After the software is satisfied that it can communicate with remote hardware, it will initialise an internal timer that is used to update the results on the graphical interface.

If the connect sequence is successful, it will determine the current state of the test equipment. It will determine the tester model number, test pattern and whether the tester has a self-loop activated. After determining this, it will update the user display to show the current tester configuration. The system timer is also initialized here. The system timer will trigger an interrupt, which will cause a message to be sent to the message handler. The message handler will call the OnTimer event.

The following figure finishes the connection sequence flowchart:

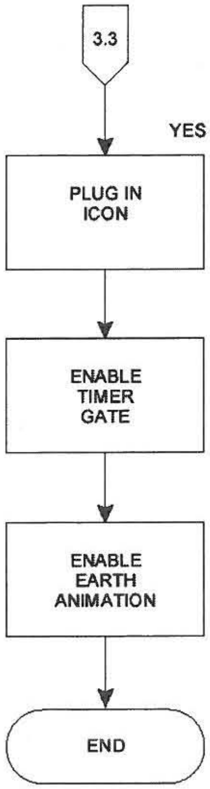


Figure 48 - Flowchart 3.3: Finalise connection sequence. This chart shows the final steps the graphical interface takes and indicates to the user that the connect sequence is successful.

The PLUGIN icon next to the disconnect check box is displayed to indicate a successful connection. Upon completion of this flowchart, the EARTH icon is spinning, showing that timer events are taking place.

5.1.1.3 Close COM port

This function will close the COM port if it is possible to close the port and if it is not being held open by another programme. This is normally the case when the user forgets to close the terminal window and commences working in the FB4000 programme.

5.1.1.4 OnTimer event

This event occurs every few milliseconds. The time between events is determined by the variables in the setup programme TESS.EXE. When the "Direct to server" selection is made, the value of time triggers will be every 500 milliseconds. When the "Direct to device" selection is made in the setup programme, this value is set to 120 milliseconds. The software uses these time delays to send commands to the test equipment. It is roughly the amount of time taken to send a command to the tester and to receive a response back.

Figure 49 shows the flowchart for the OnTimer event.

Keep in mind that this flowchart is executed every few milliseconds as discussed above. This flowchart starts by checking if the COM port is still open. If so, it inspects whether there are any communication errors like a line break, etc. It then checks the receive buffer of the COM port for any characters to process. If any characters are in the receive buffer, they will be deciphered and present on the graphical interface. If no characters are in the receive buffer, it will transmit the next command in the memory tank.

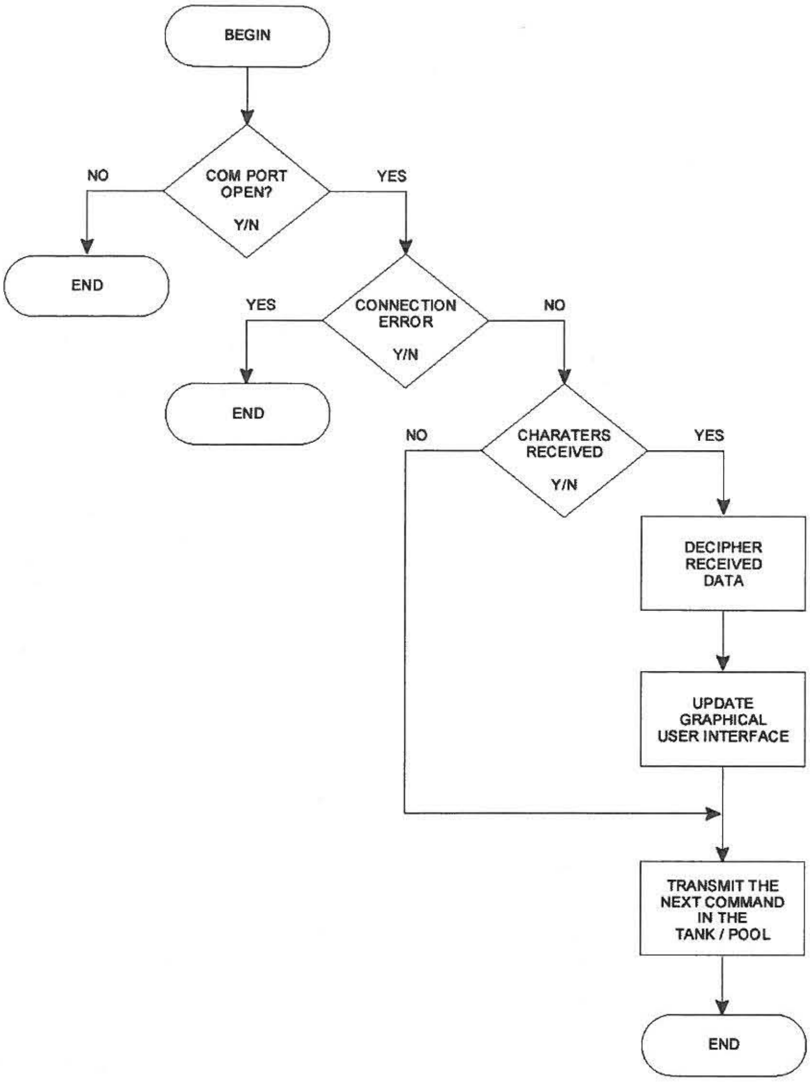


Figure 49 - Flowchart 4: OnTimer events. This event is performed every few hundred milliseconds by the graphical interface to retrieve the current test results of the Fireberd® tester.

The OnTimer function performs two functions:

- Deciphering of the received data, and
- Transmitting the next command in the memory tank.

a) Deciphering the received command

This function is responsible for checking the received information, as well as posting the correct information to the dialog box. Every command in the memory tank that is sent to the transmitter is tagged with an address. The address can have one of four values: RESULT1, RESULT2, ALARMS or RECEIVER. When the information is received back from the remote device, the programme will know where to post the received information. Refer to appendix E. RESULT1 and RESULT2 update the two text boxes at the top left-hand corner. The interface group is updated with the ALARMS tag, while the receiver group is updated with the RECEIVER tag. The information that is displayed in the result boxes can be altered by clicking the buttons beneath the text boxes. The programme will empty its memory tank before it inserts the command that will process the next result. At present both boxes are capable of selecting four results.

The left button selects between the following results:

- Bit errors.
- Block errors.
- Pattern slips and
- Pattern Losses.

The right button selects between the following results:

- Blocks.
- Receive frequency.
- Efficiency and
- Interface.

For an explanation on the outputs, refer to Chapter 2 – Maintenance procedure.

The only way these results can be altered is to change and recompile the source code.

b) Transmitting the next command in the memory tank

The memory tank is responsible for keeping track of commands that is to be sent to the COM port.

Refer to the following figure for a description of the memory tank.

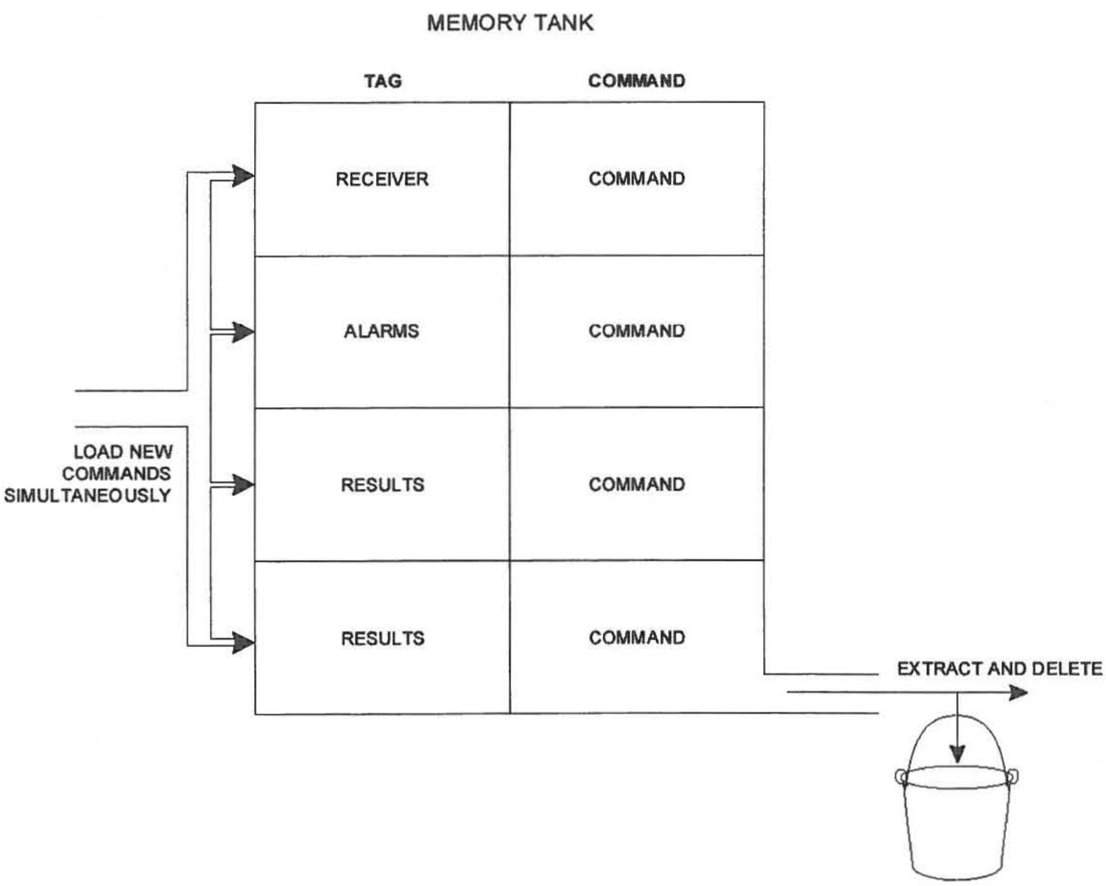


Figure 50 - Memory tank operation. The memory tank is responsible for sending the commands in pace with the timer events to the Fireberd® tester. The diagram also shows that only one command is sent for each timer event. After four commands is transmitted, the memory tank is reloaded.

The memory tank holds four commands in its memory. Every time this function is called, the bottom command is extracted, executed and then deleted from the tank memory. When the tank memory is empty, it is loaded with four new values. It takes four timer events for the memory tank to be cleaned out and reloaded.

5.1.1.5 Minor events

The above text described the major events that are handled by the software. The following paragraphs describe the minor events that make the software operation complete.

The following minor events are discussed:

- Insert Error.
- Restart.
- Pattern.
- Terminal.
- CODI.
- PIO selections.

Not all of the above events make use of the tank memory or the OnTimer event to accomplish a task on the remote devices. The OnTimer events are simply ignored. When any of the minor events occur, they are simply executed by stalling the timer until the event is finished or the disconnected check box is cleared again.

a) Insert error

This event will send an insert error command to the tester commanding it to transmit a bit-error condition to the CTU. Upon completion of this command, the OnTimer events will commence.

b) Restart

A restart command is sent to the tester to instruct it to reset its results and to start testing with all its internal counters refreshed.

c) **Pattern**

The new pattern that is selected on one of the radio buttons is simply sent through to the tester. The tester does an automatic restart. The pattern change will be evident on the result boxes.

d) Terminal

When the terminal button is pressed, the COM port is closed, as not more than one application can use the COM port at the same time. The OnTimer events are stopped. The FB4000 programme will then load the Windows Terminal®. After loading, the Terminal Window will pop up, enabling the user to do manual configurations and commands. After all commands are completed, it is the user's responsibility to close the Terminal® program. If the programme is not terminated, it will keep the COM port busy. After successful termination, the user can click the disconnect check box to connect the host software to the remote equipment again.

e) **CODI**

This event takes the same course as the terminal event. When this button is pressed, the software will connect to another port on R2PLEX. This device is known as CODI and is responsible for controlling the CTU that aids in testing of the faulty circuit. Again the same COM port is used for its connection. Upon completion of the CODI software, it must be terminated before the user can connect the FB4000 programme to the remote tester again.

f) PIO selection

This group of controls operates on its own in a similar manner as the terminal and CODI events. The only difference is that no programme is loaded. The PLEX settings are set on the two buttons to the left of the figure. If the desired settings are reached, it can be sent to the FRED controller by pressing the send button. The send button event will stall the timer events. It will then connect to FRED microcontroller's local prompt by sending a FRED password sequence. The desired settings are made. The COM port is then disconnected.

5.2 Directory structure and file locations

The software was designed around a strict directory structure, meaning that the software looks for specific files in the directories given. The software is installed in a main directory: C:\TESS3. This means the software can be executed on any drive letter, as long as it resides in the first subdirectory from the root directory \TESS3. In this directory, all the data files that are created by the TESS setup programme, are kept. This directory also contains Microsoft Terminal® setup files together with a copy of the terminal application. The two main programme files reside in two subdirectories beneath the \TESS3 directory. The setup application TESS is contained in the C:\TESS3\TESS directory, while the tester application FB4000 resides in the C:\TESS3\FB4000 directory.

5.3 The setup programme

This programme is responsible for setting the primary parameters that are used in the test program, FB4000.

5.3.1 Basic operation

The following discussion will take the user through the setup process of the software, describing all the different parameters that can be set to alter the functionality of the software performing the test operation.

Execute the programme "Tess.exe" in the "C:\Tess3\Tess" directory by clicking on this icon:

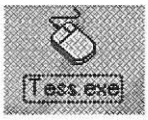


Figure 51 - TESS.EXE programme icon.

The main dialog box will pop up on the screen,

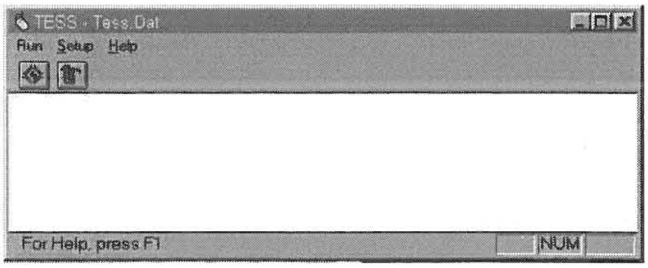


Figure 52 - TESS dialog screen. From this view the user can select either the setup or tester-dialog views.

To open the "General Setup" dialog box, press the menu bar, Setup option, or press this button:

Figure 53 - Setup button.

The setup dialog box will look something like this:

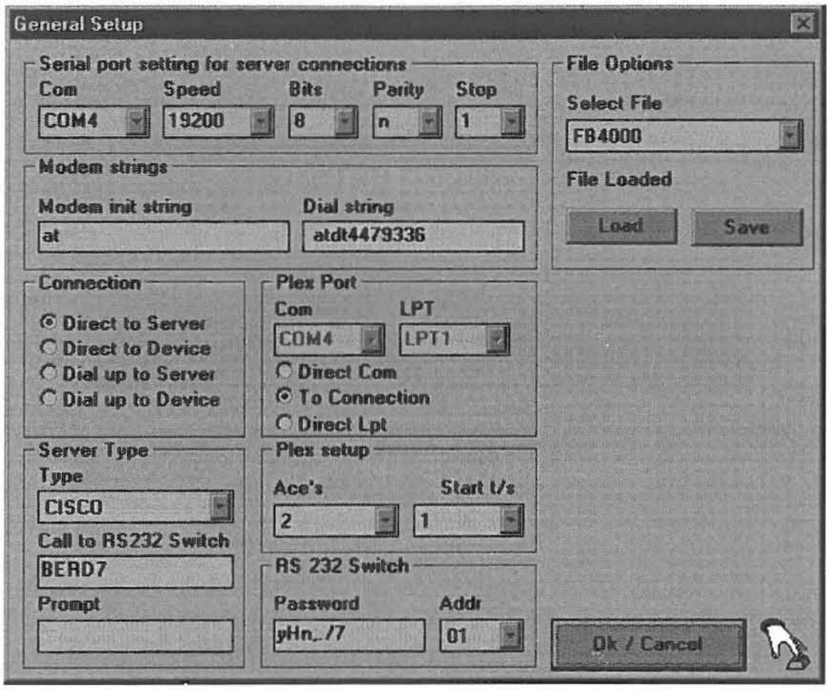


Figure 54 - General setup dialog box. From here the user can select various options for the type of connection that is used towards the remote test desk.

This dialog box is responsible for setting up most of the parameters used by the test programme FB4000.exe.

From the dialog box a few parameter groups can be identified:

- Serial port settings.
- Modem strings.
- Connection.
- PLEX port.

- PLEX setup.
- RS232 switch.
- Server type.
- File options.

5.3.1.1 Serial-port settings

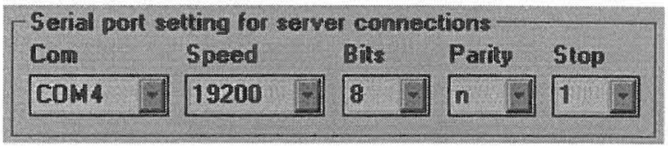


Figure 55 - Serial port settings.

This group of parameters determines the asynchronous speed that is to be used when communicating with the RS232 switch and all the devices attached to it. It determines the configuration used by the Connection and PLEX port group.

5.3.1.2 Modem strings

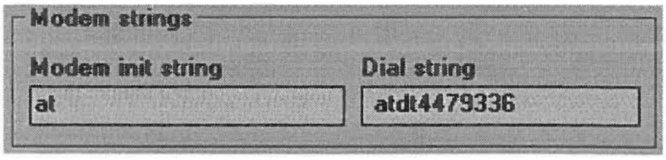


Figure 56 - Modem strings.

This feature is not yet implemented. It will be used in conjunction with the dialer software that will automatically connect to the computer running the remote-control software to the remote test desk. At present the Microsoft Windows® terminal is used by typing the command, AT, followed by the desired telephone number. Sometimes it is necessary to change the configuration of the modem for the duration of the connection. This is known as the modem initialisation string. At present, the temporary profile must be set up using the Windows terminal screen.

5.3.1.3 Connection

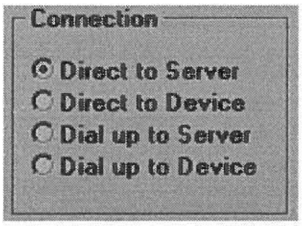


Figure 57 - Connection options. The user selects connection type here.

The different types of connections are discussed in the Network design chapter. When connecting to the remote test desk via the various types of connections, different conditions occur that must be compensated for, for example: The server (like the CISCO 500CS) causes a time delay between the commands and responses given by the remote-control software.

5.3.1.4 PLEX Port

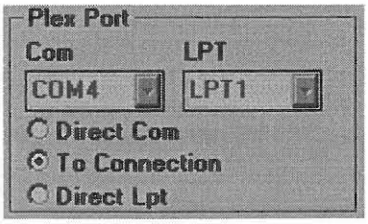


Figure 58 - PLEX options. The PLEX unit can be individually controlled by a personal computer printer port (LPT) or serially by using FREDs RS232 interface.

This group of controls were added for flexibility needed by the software.

When using the test system in a remote configuration the setting will always be "to connection", which signifies that the software must use the COM port, which is set up in the serial port-settings group. FRED is given commands to switch the time slots on the PLEX boards.

In some cases, the software is also used on a local test desk not using the R2PLEX or FRED boards. In this case, PLEX boards are connected directly to the computer's LPT (Printer) port, and the tester is connected the COM port specified in the serial port-setting group. To use this feature the "Direct LPT" radio button will be selected. The LPT list box selects the LPT port that is to be used. The "Direct COM" connection is not implemented.

5.3.1.5 PLEX Setup

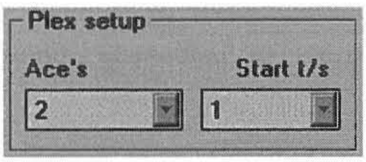


Figure 59 - PLEX setup. This selects the amount of ACEs and starting timeslot the PLEX unit is connected to. This is a virtual setting to make the user's work easier.

The number of ACEs and the first timeslot the PLEX board is connected to, is set here purely as a reference to the faultsman testing the circuit. The technician enters the values on the RCE to which his test equipment must be connected. The software always connects only to logical positions. To identify this logical position, the values are set here to make it easier for the technician to test the circuits. Example: the PLEX board, grouping four timeslots together, can be attached to any timeslot on the backplane of the test multiplexer. If the first timeslot on the backplane were to be 6, the setting in the list box would be 6. Now the software would eventually show timeslots 6, 7, 8 and 9 as the legal connections to the CTU. However, the PLEX board could have been plugged in at timeslot 16 to 19. If the settings in this group were kept at timeslot one, setting 1 would actually connect the tester to slot 16, 2 to 17 and so on.

5.3.1.6 RS232 switch

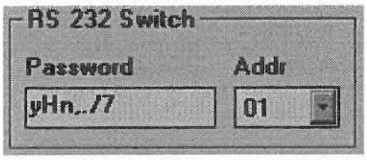


Figure 60 - R2PLEX options.

The RS232 switch is not implemented at present, but will be used for setting the R2PLEX local switch password. As described by the chapter on hardware, the R2PLEX switch is permanently connected to an output port. To change the connection to a different output port, the password string is sent from the computer's serial port. The FRED microcontroller in turn listens to the incoming transmission. If it recognises the incoming password string, it will switch to local mode, enabling the software to alter the output port to a different device connected to R2PLEX. On the other hand, if FRED does not recognise the incoming string, it will pass the data on to the selected output device. Data transfer is possible at any speed from 1200 to 115000 bit/s, but FRED will only listen for the password string at a programmed speed of 9600 bit/s. It is also possible to cascade multiple R2PLEX boards, in which case the different R2PLEX devices are given different passwords. This is to be implemented at a later stage when it becomes necessary.

5.3.1.7 Server type

Figure 61 - Server type. This selects the type of entry point the user's access circuit is connected to.

For the time being this group will always be selected as CISCO, meaning, a CISCO® type terminal server will be used. "Call to RS232 switch" or R2PLEX is the actual command that is sent by the software to the terminal server to connect the incoming port to the R2PLEX switch. If this connection is successful, data transfer can commence. It is possible to bypass the Server totally and to connect the incoming port directly to the R2PLEX switch. This is done in situations where only one connection to the remote test desk is required.

File options

This group sets the software filename and saves the selected information that has changed in the whole dialog box.

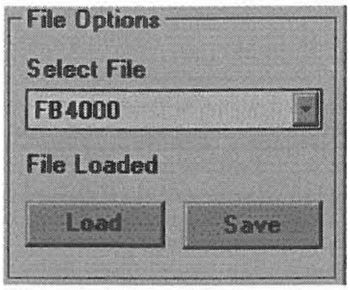


Figure 62 - File options. These options save the settings made by the user to a data file on the computer's hard disk. This data file is used by tester programme during initialisation.

The "Save" button must be pressed to save the selection. By pressing the "Load" button, previous settings may be restored if some selections have been made that the user is not sure of. If the "OK/Cancel" button is pressed before the "Save" button, the selection is cancelled. Hence, the user has to restart the setup dialog from the menu or toolbar of the parent window.

5.4 The Fireberd tester software

The tester software will now be described. Assume that all the necessary changes were made in the setup programme for the test software to work properly.

If the user is not sure of this, the setup button can be pressed in either the setup window or the tester window to verify all the settings.

Figure 63 - Setup button.

To load the tester software from the setup window, the following button can be pressed,

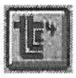


Figure 64 - Tester button. Selecting this button executes the tester window.

or the FB4000 selection can be made from the file menu in the menu bar. The following window, which is known as the tester software, will pop up.

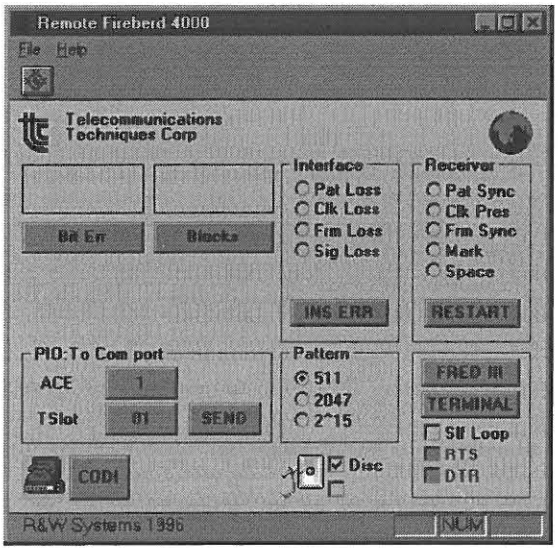


Figure 65 - Tester software dialog. This window enables the user to interact with remote hardware and to retrieve test results from the Fireberd® tester.

With reference to Chapter 2, the procedure of testing a circuit will now be discussed. First the faulty circuit's information is retrieved to see from where to where the circuit is working. This information is obtained from Unibase® or from a client over the telephone.

Now the user has to put the circuit into a SPLIT condition, meaning the circuit is broken into two sections, an A side and a B side. To accomplish this the user must first log onto the RCE system. When the remote-control system is connected to a CISCO terminal server, the IP address of the host is required.

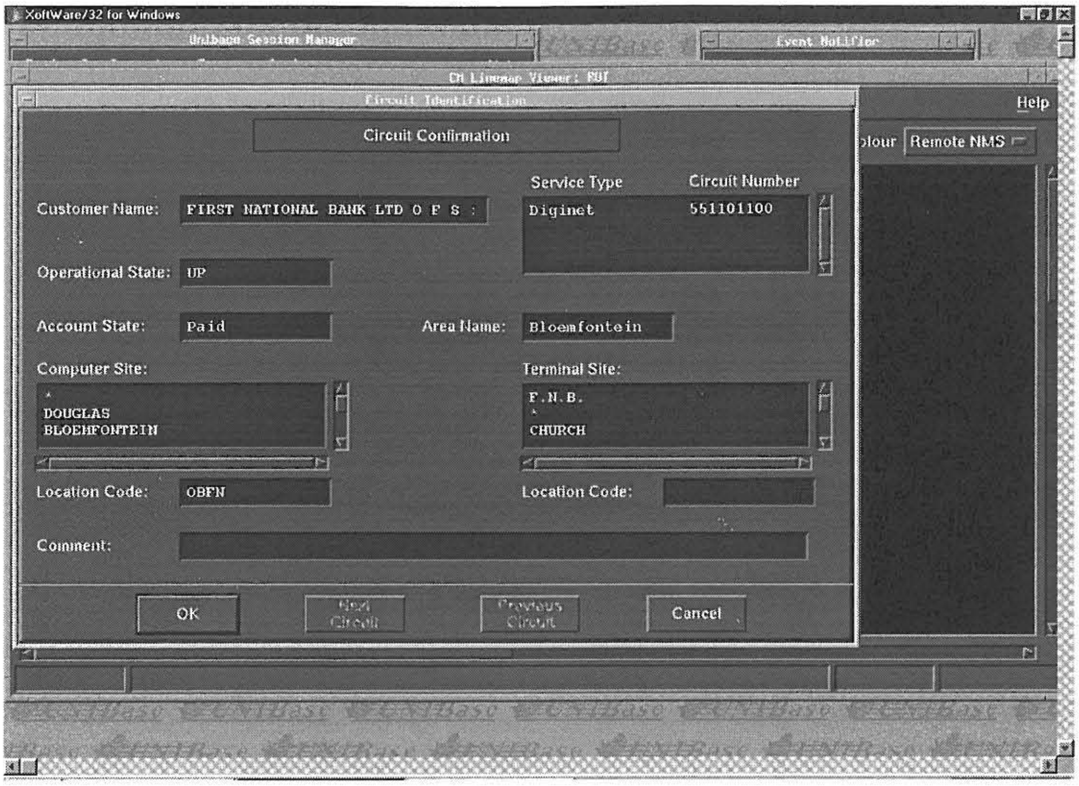


Figure 66 - Picture of fault reported on the Unibase® system. This window helps the user to determine the physical addresses of both the A and B-end of the circuit. It also shows the client's name and the circuit number that is to be tested.

Press the Terminal button on the tester window,

Figure 67 - Terminal button and additional options. The Terminal button executes the Windows 3.1® terminal programme.

The Microsoft Windows® terminal should pop up, as shown in Figure 68.

Upon pressing the "Enter" key on the keyboard, the cursor will hop downwards. To connect to the RCE, the user enters the IP address of the desired host. The host could be either the Master RCE or the Slave RCE.

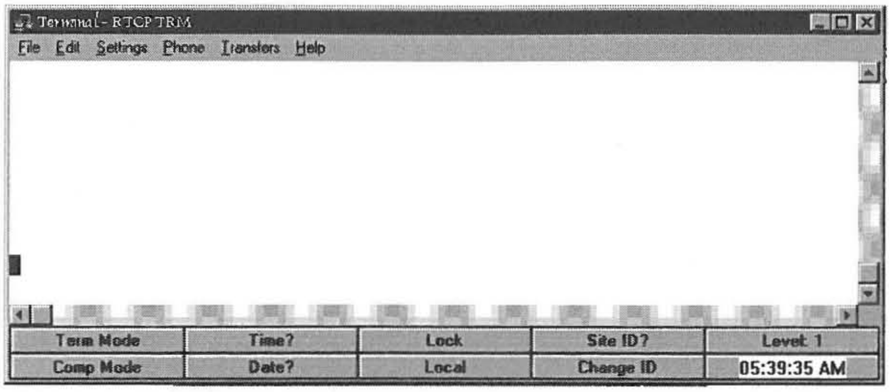


Figure 68 - Microsoft Windows® Terminal. From here the user can alter and communicate with other devices connected to the R2PLEX unit.

The user first displays the circuit's information by typing in the desired command. The following window is an example of what this information would look like on the terminal window:

```
CIRCUIT DESIGNATION : 55-52657-00          00
SERVICE CODE        : 64K/X/*****/X/*****/
CIRCUIT TYPE        : BP
STATUS      DATE    : Sep 11 1997         STATUS          : COMPLETE
BOOKED      DATE    : Sep 10 1997         INSERVICE DATE  : Sep 11 1997
COMPLEDION  DATE    : Sep 11 1997

ROUTING :-                (X = alarms inhibited at this node)

TRANSMISSION SYSTEM DESIGNATION                    *  STATUS   * Time Slot

BFN/AC/1  (0007)   - BFN/DM/A            0042     * COMPLETE *   14
BFN/AC/2  (0001)   - BFN/AC/1   (0119)   0001     * COMPLETE *    6
BFN/AC/2  (0099)   - WGH/DM/H            0012     * COMPLETE *    8
```

Figure 69 - RCE output of circuit information. This screen tells the user the physical path a circuit takes through the Diginet network.

The important information here is the circuit speed, 64K and the routing information. The circuit starts out on timeslot 14 on the A side multiplexer. It is time-division multiplexed into the 2Mbit/s aggregate towards the Bloemfontein ACE (Automatic

Crossconnecting Equipment), node number one (BFN/AC/1), port 0007. From here on the circuit is switched on an inter-ACE connection towards Bloemfontein ACE node 2 (BFN/AC/2). Note the different timeslots that are being used between the various switch stages. Also, note the port numbers used on every stage designated by the number in brackets. In this case, it is possible to connect the circuit to the test equipment of either ACE one or two. Finally, the circuit exits to the B side on port 99, timeslot 8 of BFN/AC/2.

Before any testing is performed on the circuit, a few diagnostic commands can be issued to the RCE to determine the physical status of circuit. This is done on both timeslot level and 2Mbit/s level.

From the routing information can be seen that there are 3, 2Mbit/s portions and 2 timeslot portions. The reason for only 2 timeslot portions is that on the A and the B sides the timeslots are de-multiplexed from the 2Mbit/s, but on the inter-ACE portion, the timeslot is kept intact. This means that on the inter-ACE portion, it is possible to determine the status of any timeslot by just checking the alarm information of the whole 2Mbit/s.

By doing diagnostics on the 2Mbit/s, fault parameters such as AIS (Alarm Indication Signal), LOF (Loss Of Frame), RAI (Remote Alarm Indication) and LOS (Loss Of Signal) can be determined. By doing diagnostics on timeslot level, parameters like WCF (Wetting Current Fail), DCEPO (DCE Power Off), LOF (Loss Of Frame), 64K input fail etc. can be determined. Refer to Chapter 2, Maintenance procedure, for more information.

The next step would be to connect the circuit to the test desk for testing. The circuit will be connected to the test multiplexer on Bloemfontein ACE 2. To do this the circuit must be configured into a SPLIT condition with the desired command issued to the RCE.

Figure 70 displays how a circuit could be connected to the RTD.

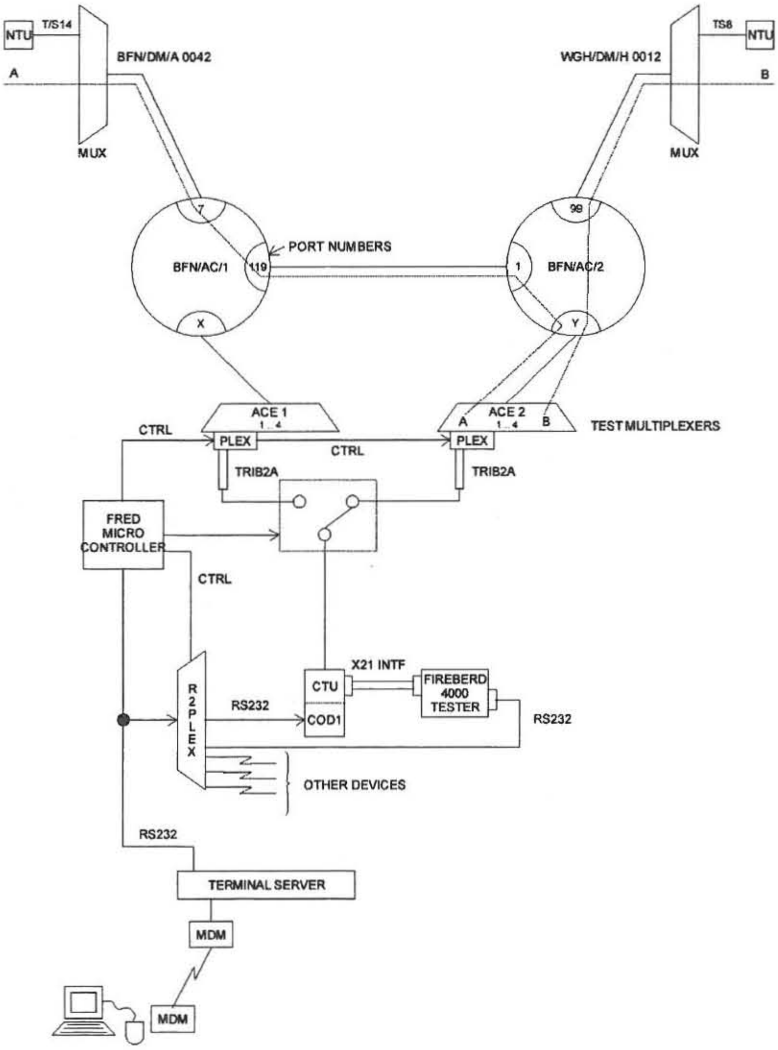


Figure 70 - Diagram showing the connection of the user to the RTD. This also shows how the test circuit is physically connected to the RTD. Referring to Figure 69, the port and timeslot information can be followed in this figure.

The RCE software will take the user through the following steps:

- Type the circuit number "55-52657-00".
- Connect the test equipment both ways (A and B side).
- Choose the desired test multiplexer.
- Choose the desired A and B side timeslots on the test multiplexer. In this case, it can be any timeslot between 1 and 4. Choose 1 for the A side and 2 for the B side.
- The test circuit is now successfully put into a SPLIT condition.

It is possible to connect the remote desk to either the A or the B side by selecting the correct timeslot on the tester window. The following figure shows the group of controls responsible for selecting the ACE and timeslot positions.

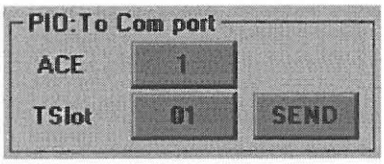


Figure 71 - PLEX settings. From here the user manipulates the physical connection to the PLEX unit.

After selecting the correct timeslot, the new settings are sent to the microcontroller by pressing the "SEND" button. Now CODI should be set up by pressing the "CODI" button.

Figure 72 - Button to load CODI software. This button disconnects the tester window for the RTD and diverts the R2PLEX unit to the CODI controller on the CTU.

The following window will pop up:

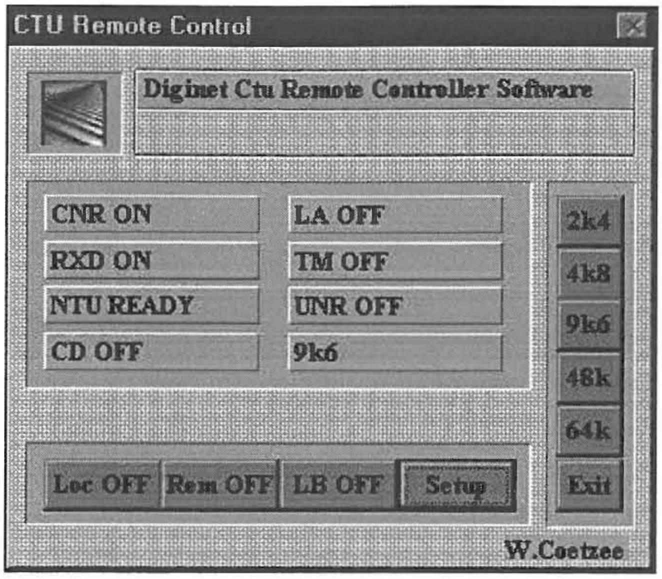


Figure 73 - CODI[4] software. This software interacts with the CODI controller and displays the parameters that can be seen on the CTU's liquid crystal display.

From the window can be seen that it is currently set to a speed of 9K6. To change the speed press the "64K" button. The 9K6 value will now change to 64K.

The "Loc OFF" button gives a loop on the CODI unit back towards the local tester. This is done in cases when the correct working of the CODI unit is doubted. It is also used to verify that the connection between the tester and the CODI unit is correct.

The "Rem OFF" button initiates a loop back on the remote NTU, which can be either the A or the B side. This, however, is only true for a circuit working at a rate below 64Kbit/s. When a circuit is working at a rate of 64Kbit/s, the client is phoned at the business premises where the loop is required. The client will initiate a loopback on the front panel of the NTU, which is the case with the circuit being tested at the moment.

---

[4] Programme written by Mr. Willem Coetzee, Telkom SA.

The "LB OFF" button gives loop back towards the client's premises. This is used in cases when the test technician is not sure of his/her own test equipment. A technician at the client's premises will have to test with a BER tester to see the loop.

After selecting the desired settings on the CODI unit, the "EXIT" button is pressed to return to the main tester window.

To connect the test equipment to the circuit portion under test, press the "disc" checkbox.

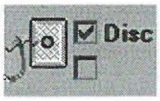


Figure 74 - Disconnect software from the RTD. When the user checks the "disc" box, the software disconnects from the COM port. When the user unchecks this box, the software re-establishes its connection with the RTD.

The software will, after doing a successful connection to the tester, display a spinning Earth icon at the top right-hand corner. Soon after this the result groups will be updated on the current test. There are three groups that can be monitored to determine the current results.

They are the following:

- Results,
- Receiver and
- Interface.

5.4.1 Results

This group plays the most important role in the testing of the circuit, because it displays almost all of the parameters that are tested for the duration of the test. The following figure displays the results group.

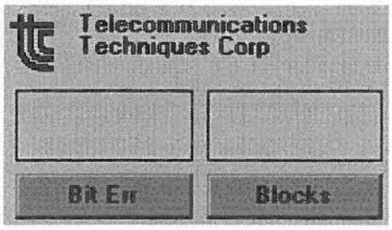


Figure 75 - Tester results. Different results can be selected by pressing the buttons: "Bit Err" and "Blocks". As a button is pressed, its caption will change according to the type of result it must display.

The left box displays the primary results that are tested. Four results can be selected by pressing the "Bit Err" button. These are Bit Errors, Block Errors, Pattern Slips, and Pattern Losses. Refer to Chapter 2 for an explanation of these parameters.

As soon as anything is wrong with the circuit under test, these counters will start to count the specific events. Any pattern slips and pattern losses should be considered as serious.

The right text box displays some secondary results: Blocks, Receive Frequency, Efficiency, and Interface. This box is particularly interested in verifying the correct interface on the test equipment, as well as verifying the correct speed setting on the CODI unit. Now this parameter should read "64000", indicating that the CODI unit is set for 64K.

The Blocks parameter is used to see if the pattern being sent out by the tester is received back. As this is only a counter, it will count the number of blocks received.

5.4.2 Receiver

The receiver displays the status of the test being performed. The circuit currently being discussed will display Pattern Synchronisation, Clock Present, Mark and Space when the Loop back function is operated on the remote NTU. Upon deactivating the Loop back, only Pattern synchronisation will be turned off.

Figure 76 - Receiver. This box makes it easy for the user to see when the tester is in synchronisation with a valid test pattern. "Pat. Sync." will occur on a loop as well as a remote-tester transmit pattern.

The "RESTART" button will reset all the parameters and restart the test equipment.

5.4.3 Interface

These parameters will only lit when an event occurs and will automatically reset after the result has been gathered from the tester. As soon as the interface parameters have been read, the tester will reset the event. The next time the tester software inquires the test equipment and no event has taken place since the last inquiry, the interface parameters will be reset.

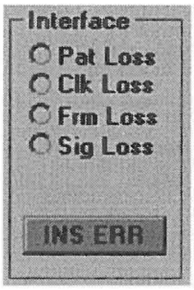


Figure 77 - Interface. This box acts as a history indicator showing that something has happened. This is useful when the user does not look at the test results all the time. As soon as one of these conditions lit up, an audible sound is also sent to the speaker.

This group of controls only aids in the trapping of faulty circuits, since not all of the results can be displayed at the same time. The “INS ERR” button manually inserts bit errors into circuit under test. The number of errors that are sent can be seen on the primary result box (top, left box). This is sometimes used to verify that a loop is seen and not a remote tester.

5.4.4 Pattern

This group is used to select the desired pattern the test is to be performed with.

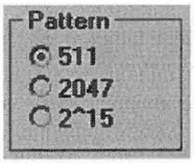


Figure 78 - Pattern selection. This box shows the current pattern when the graphical interface connects to the tester. It is also used to select a new test pattern.

Only three pattern types can be selected: 511, 2047,2^15. Low speed circuits (64Kbit/s and below) are normally tested with either 511 or 2047, while circuits with a speed higher than 64Kbit/s are tested with a 2^15 test pattern.

5.5 Summary

This Chapter gave a description of how to use the remote-control software to test a faulty circuit with the aid of an example. Furthermore, it also explained how the software operates internally. The most important events were described in detail, while the minor functions were discussed briefly to bring the reader to an understanding of the methods used to communicate with remote devices.

Chapter 6

Experimental results

This chapter discusses some experiments which aim to promote the feasibility of the project. Experiments were chosen to show in at least two areas that the idea of having a remote testing facility is necessary. First, the credibility of the system is tested to see if the system does indeed work without causing unnecessary errors. Secondly, it shows that a lot of time and money is saved. The last experiment in this chapter will show the amount of time needed to test the same test circuit in experiment 1 on a remote test desk, versus testing it at a centralized test centre.

6.1 Experiment 1 Credibility of the system

6.1.1 Purpose

To determine if the test system causes any errors on a circuit under test. This experiment therefore determines if the system adds any additional errors to the circuit under test.

6.1.2 Method

A test circuit was built that ran locally from one multiplexer through the ACE to another multiplexer. This was done to minimise the possibility of errors caused by the circuit under test. The physical distance was kept to minimum so that the effects of long distance could not affect the test in any way. From the following figure can be seen that this distance would be about thirty metres.

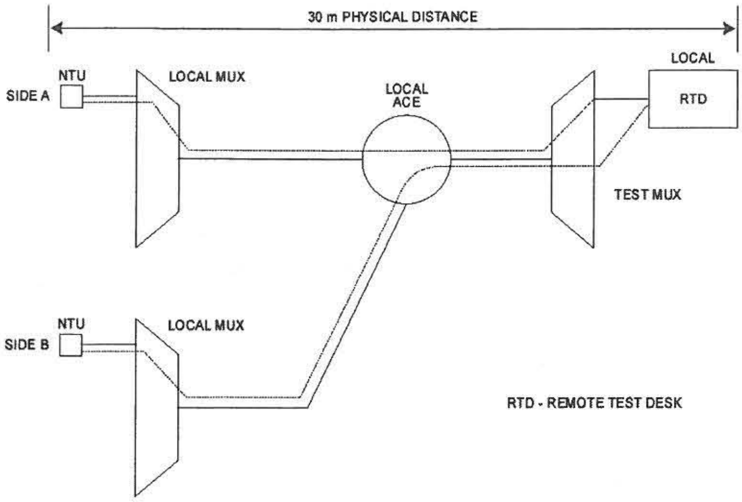


Figure 79 - Experimental test circuit to test for external interferences causing errors in the test-results.

The circuit is SPLIT on the ACE to connect it to the RTD (Remote test desk). The RTD was connected to the A side of the circuit for the purposes of the test. A TTC Fireberd 4000 tester forming part of the RTD is to be used as pattern generator to determine if bit errors are caused by the system. The normal test duration of a faulty circuit is between 10 minutes and an hour. Two sets of three tests were performed. The first set of three tests was tested for three consecutive days running eight hours at a time. The second set was tested for three consecutive weeks. Each of the three tests was performed on a seven-day cycle. During the second set of tests the test circuit was subjected to external interferences like the following:

- The test equipment and test circuit was fed from a bad power source with noise.
- People fiddling around with test equipment and cables.

Each of the two sets was tested at the following bit rates:

- 64Kbits/s
- 48Kbits/s and
- 9K6bits/s.

The speeds that were selected were chosen on merit because it is the most common circuits used on the Diginet network.

6.1.3 Results

The outcome of the first set of results was successful, and no errors were found for the duration of the three tests. The following table shows the outcome of the results for the three tests.

Table 1 - Table showing three sets of results for 9K6, 48K and 64K for eight-hour periods.

	A	B	C	D	E	F	G	H	I	J	K
1	Time	9K6 Bit/s Results			48K Bit/s Results			64K Bit/s Results			Misc
2	Hours	Blocks	Bit Errors	%Efficiency	Blocks	Bit Errors	%Efficiency	Blocks	Bit Errors	%Efficiency	
3	00:00	0	0	100.00%	0	0	100.00%	0	0	100.00%	<-Start
4	01:00	16903	0	100.00%	84671	0	100.00%	112547	0	100.00%	
5	02:00	33807	0	100.00%	169341	0	100.00%	225093	0	100.00%	
6	03:00	50710	0	100.00%	254012	0	100.00%	337640	0	100.00%	
7	04:00	67613	0	100.00%	338682	0	100.00%	450187	0	100.00%	
8	05:00	84517	0	100.00%	423353	0	100.00%	562734	0	100.00%	
9	06:00	101420	0	100.00%	508023	0	100.00%	675280	0	100.00%	
10	07:00	118323	0	100.00%	592694	0	100.00%	787827	0	100.00%	
11	08:00	135227	0	100.00%	677364	0	100.00%	900374	0	100.00%	

With all the tests, a 2047 test pattern was used to determine the results. This test pattern was used to comply with the O152 ITU recommendation [17].

From Table 1 can be seen that the system in all three case studies caused no errors.

The second part of this experiment was performed over a three-week period. Each of the three speeds was tested for a period of a week. Furthermore, the results of these tests were subjected to bad power sources and people messing around with the equipment and cables during the test duration.

The following table shows the results from this experiment.

Table 2 - Table showing the results of the three speeds over a three-week period.

	A	B	C	D	E	F	G	H	I	J	K
1	Time	9K6 Bit/s Results			48K Bit/s Results			64K Bit/s Results			Misc
2	Days	Blocks	Bit Errors	%Efficienc	Blocks	Bit Errors	%Efficienc	Blocks	Bit Errors	%Efficiency	
3	0	0	0	100.00%	0	0	100.00%	0	0	100.00%	<-Start
4	1	405680	345	99.60%	2032093	33	99.96%	2701121	17	99.98%	
5	2	811361	4	99.80%	4064187	0	99.98%	5402242	0	99.99%	
6	3	1217041	0	99.87%	6096280	1279	99.49%	8103363	91	99.96%	
7	4	1622721	143	99.86%	8128374	26	99.61%	10804484	0	99.97%	
8	5	2028401	0	99.89%	10160467	13	99.69%	13505605	68	99.96%	
9	6	2434082	12	99.90%	12192561	2	99.74%	16206726	0	99.97%	
10	7	2839762	0	99.92%	14224654	461	99.70%	18907847	34	99.97%	

The %Efficiency or percentage of error free seconds (EFS) is calculated as the ratio of the number of available seconds in which no errors were detected to the total number of available seconds [8]. This result is calculated by the internal circuitry of the test equipment [40][41]. Blocks can be seen as packets containing an amount of bits, which is set by the test equipment's configuration options.

6.1.4 Conclusion

It can therefore be said that the credibility of the RTD is 100% for the normal test durations of between 10 minutes and one hour. From the second part of the experiment, it must be highlighted that the test results are only as good as the test environment, meaning that if a circuit is only tested for ten minutes and someone messes around with the cables, test equipment or power, the results would have been unreliable. To illustrate this, a test was performed for 8375 seconds and the wrong button was pressed on the front panel of the test equipment. After a while the source of the errors was discovered and restored. This human error caused 31739 bit errors to be accumulated. The efficiency dropped to as low as 93.84%.

6.2 Experiment 2 Savings on after-hour call-outs

It is of great importance to show that the project would save time, as a client who reports a faulty circuit wants his business online without interruptions as stated in Chapter 2.

6.2.1 Purpose

The purpose of the experiment is to determine the average amount of time saved when testing and localizing faulty circuits by using the RTD for after-hour call-outs.

6.2.2 Method

Two sets of ten cases (call-outs) were recorded onto a spreadsheet. The average time before any testing can be performed is calculated on both tables. The difference in the two average results would reflect the average time saved by using the remote testing facility. The tools that were used determined that these results would be the Unibase fault-reporting system database. Twenty cases were extracted from the Unibase database, and a table was compiled together with information gathered on paper to accompany the results in Table 3 and Table 4. The information on paper was collected by means of a stopwatch.

6.2.3 Results

The following table shows ten cases that were accumulated in a log-book of faults that have been reported after hours.

Table 3 - Results without the RTD.

Case	Time reported	Time to get dressed after sleeping	Time to drive to the test centre and back	Lost Time	Time to restore reported fault	Total time to restore
Number	Time	Minutes	Minutes	Minutes	Minutes	Minutes
1	1:07	21	37	58	50	108
2	11:03	0	41	41	110	151
3	17:00	0	37	37	23	60
4	3:17	30	35	65	55	120
5	6:00	35	29	64	151	215
6	21:00	0	34	34	101	135
7	23:15	0	33	33	21	54
8	2:01	20	36	56	33	89
9	4:33	24	29	53	68	121
10	18:47	32	37	69	12	81
						MTTR
Average time (Minutes)				51	62.4	113.4

The times that were taken are from the first telephone call received.

The following graph shows time spent and wasted in the different case studies.

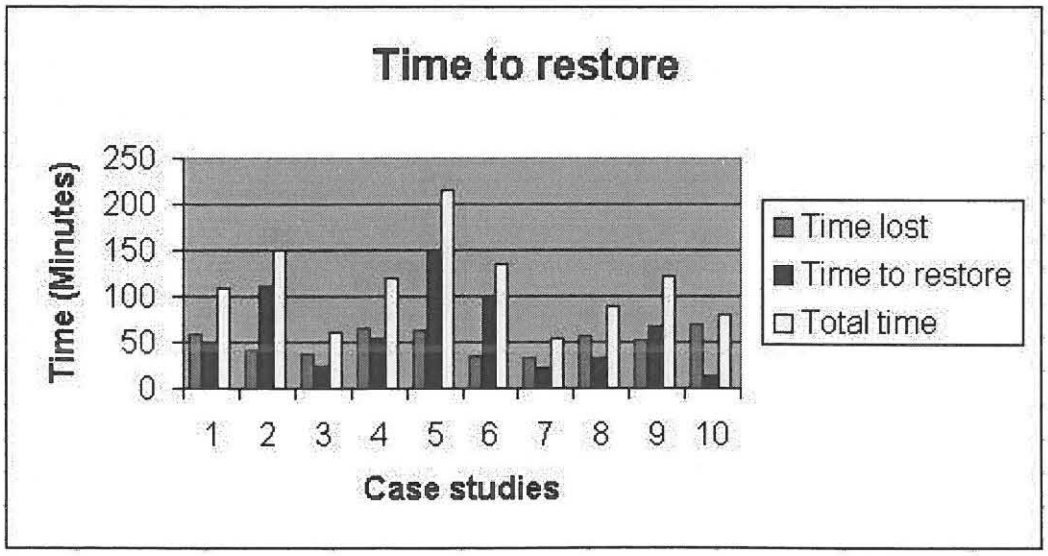


Figure 80 - Graph showing lost time, time to restore and total time for a test system without a remote testing facility.

The following two parameters were measured as a waste of time before any testing can be performed:

- Time to get dressed after sleeping.
- Time to drive to the test centre and back.

6.2.3.1 Time to get dressed after sleeping

This parameter is measured by including the time it takes to take down the information from the fault reporting centre, taking a quick shower and to dress into working clothes. From the table can be seen that this parameter is sometimes almost zero. This is because the fault was reported during the day and the person on standby was ready to go. The only time that was wasted is getting the fault details from the call centre.

6.2.3.2 Time to drive to the test centre and back

This parameter could change drastically from city to city, since the average distances that employees stay from work must be taken in to account. The times that were taken is based on the geographical area of Bloemfontein, with a maximum distance of 15 kilometers from the test centre. These parameter measures trips, both to and from the test centre.

The following table shows the effect a remote testing facility can have on the average time wasted before a test can be performed on any circuit. This is to illustrate how often it is necessary to actually drive to the test centre to localize a problem. Cases 2 and 8 were the only times the faultsman had to actually drive to the test centre and back because of faulty equipment.

Table 4 - Results with the RTD.

Case	Time reported	Time to get dressed after sleeping	Time to drive to the test centre and back	Lost Time	Time to restore reported fault	MTTR (Mean time to restore)
Number	Time	Minutes	Minutes	Minutes	Minutes	Minutes
1	13:23	3	0	3	10	13
2	17:05	3	35	38	93	131
3	2:33	3	0	3	15	18
4	5:38	4	0	4	56	60
5	3:13	7	0	7	76	83
6	18:00	6	0	6	27	33
7	9:44	3	0	3	131	134
8	3:12	27	30	57	123	180
9	12:05	3	0	3	55	58
10	21:12	4	0	4	13	17
						MTTR
Average time (minutes)				12.8	59.9	72.7

The following figure displays the lost time variables in the different case studies.

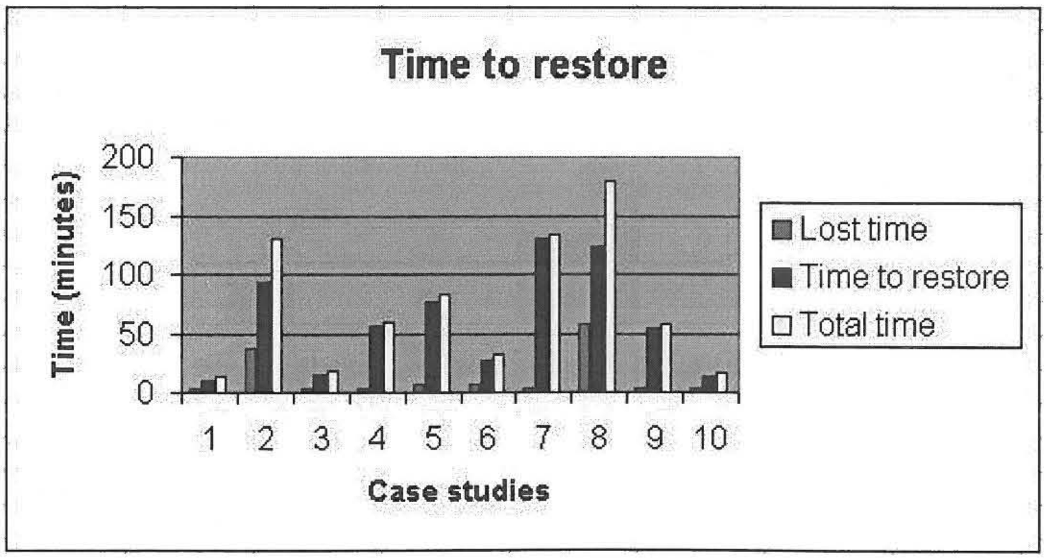


Figure 81 - Time lost over ten case studies.

From the two tables can be seen that the MTTR (Mean Time To Restore) for a test facility having an RTD is 73 minutes, and for a system without an RTD, 113 minutes.

This does not seem like much, but applying this calculation to the monthly call-out figures for the company on a national basis, it amounts to many wasted hours.

Recent statistics showed that on average 180 faults are logged per month. All of the 180 faults were tested without having an RTD facility. This would amount to an average man-hour expenditure of:

Man-hours	= (180 * MTTR without RTD) / 60	Hours
	= (180 * 113) / 60	Hours
	= **339**	Hours

Comparing this against figure with an RTD facility would amount to:

Man-hours	= (180 * MTTR with RTD) / 60	Hours
	= (180 * 73) / 60	Hours
	= **219**	Hours

This shows that with the system implemented, 120 hours could be saved per month. If a price tag could be attached to these savings, it could amount to R6000.00 (with an average of R50.00 an hour) discarding petrol costs, company image, etc. This, however, is only an assumption, since remote testing facilities are not in place to prove this.

6.2.4 Conclusion

By implementing this system, up to 120 hours can be saved per month. If the national call centre was capable of testing the circuit before calling the people on standby, approximately R70, 000.00 can be saved per year. Many times the fault could have been localized and therefore only specific people could have been called. If a person is to be called, approximately 40 minutes per fault is saved. This, on the other hand, also promotes the image of the service provider after hours. In Chapter 2 it was stated that a client wants a circuit to be available with little or no errors and little or no breaks. Thus, the longer it takes to restore a service, the longer the breaks would be.

6.3 Experiment 3 - Performance measurements on local and remote test desks

This experiment in no way affects the measurement figures of the results in experiment one. The result of this experiment only reflects the performance of testing a circuit locally against testing the same circuit remotely. Furthermore, the circuit that is tested does not have any faulty conditions that could affect the outcome of both of the time stamps that were taken.

6.3.1 Purpose

The outcome of this experiment reflects the times between testing a reported circuit on a local test facility against testing the same circuit on a remote testing facility. The result is necessary for management to determine if it would be viable to implement an automated test system and a centralized test centre for after-hour service to clients. This experiment will show that, although the remote test facility is slower, it is still a solution to after-hour fault testing.

6.3.2 Method

This experiment was done on a similar circuit as the one in experiment 1. The results are presented on a table and are measured with a normal stopwatch. The counter is to be started at the receiving of the fault by a telephone call. It must also be noted that parameters like, time to wake up and time to drive to the test centre are excluded from this experiment, as it is assumed that someone will be available any time of the day to perform the fault-testing procedure.

6.3.3 Results

All of the results for this experiment are calculated by using an average from a stopwatch. The results are tabulated into the following table from where the comparison can be done.

Multiple calculations of the same experiment might result in slight variations, but these variables can be taken as the mean.

Table 5 - Table showing the time measurements from the local and remote tests.

			Remote	Local
	Tests were done on a 166MMX host		Seconds	Seconds
0	Start stopwatch		0	0
1	Dial into mobi dial 28.8		120	0
2	Load Unibase		120	15
3	Login		30	10
4	Attend Fault		45	15
5	RCE Login		30	25
6	RCE Disp cct		10	10
7	RCE split		15	15
8	Connect to test desk		60	0
9	Test both ways 9k6		120	120
		Total Seconds	550	210

The above table was used to draw the following graph, which shows an exact comparison between the remote and local testing facilities.

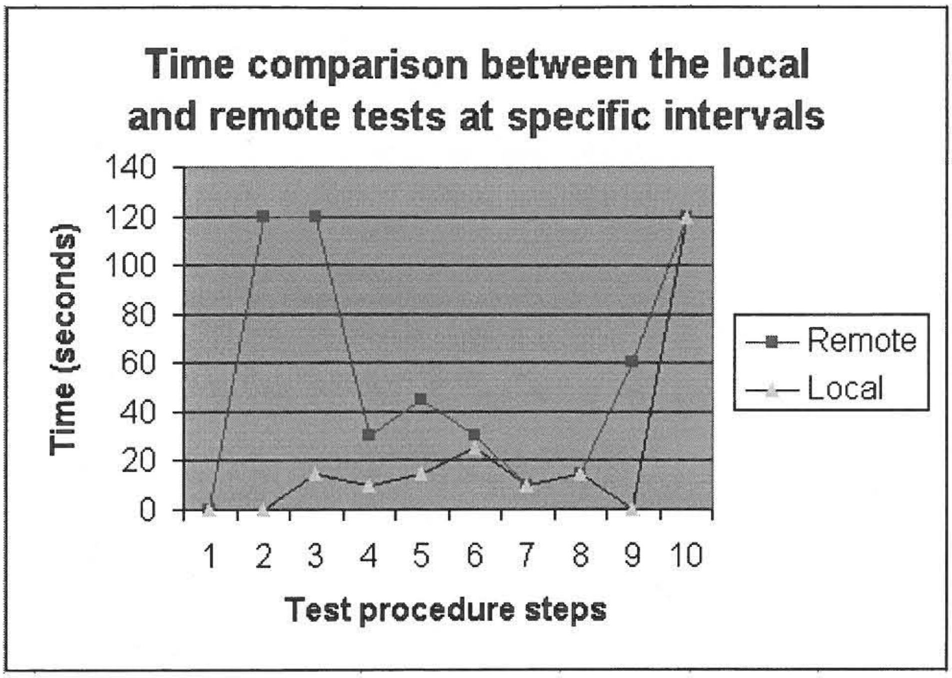


Figure 82 - Comparison between the local and the remote test facility

From the graph can be seen that the remote desk takes considerably longer to connect up to the fith step. Step nine also takes longer to set up than the local test desk. This is true because the remote test desk has to connect using a modem, which in itself is a long process, because the modems have to negotiate for the best reliable line conditions before communication can commence. Secondly, the Unibase® connection also takes longer because of the amount of information that has to be sent across the modem connection, whilst with the local connection a 10Mbit/s LAN connection is used. Thirdly, time is lost in step nine because the remote desk is configured with individual commands rather than making direct physical connections (patch cords etc.) like in the case of the local test desk.

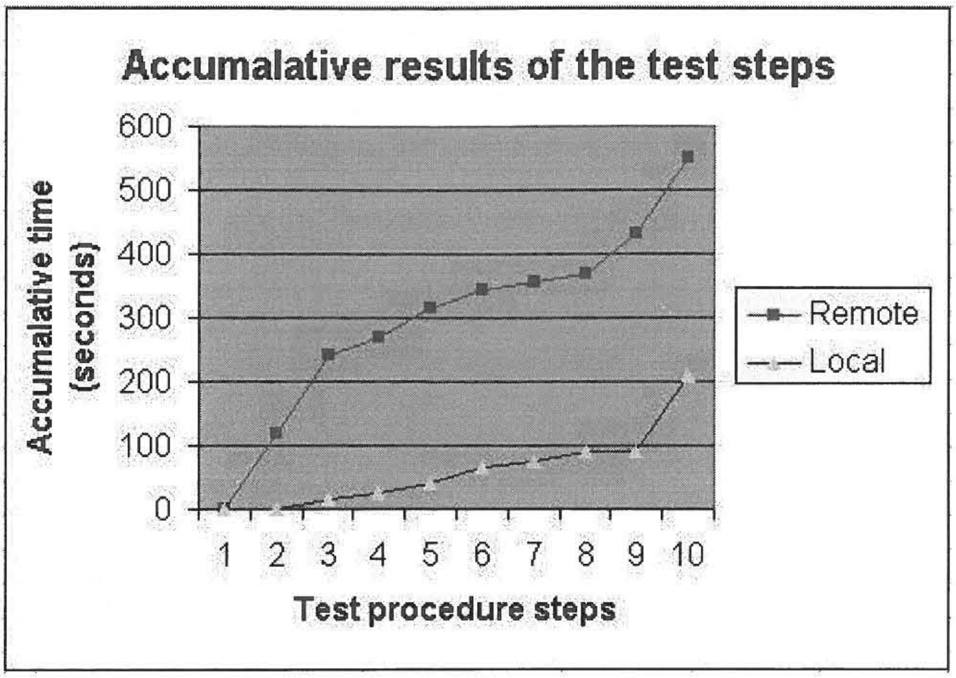


Figure 83 - Accumulative results showing the sum of all the test times.

The last graph shows the accumulative results for the sum of the results for both the remote desk as well as the local desk.

6.3.4 Conclusion

From this experiment can be seen that once the connection has been established and all the necessary software has been loaded (that is a point 5), the largest time difference is controlling the remote desk (point 9). For remote-automated testing this means that if a connection is established, testing can be performed almost as quickly as with a local test facility. Since the local desk is not remote controllable, it is a viable option to use the remote desk to do testing without excessive time losses.

6.4 Summary

This chapter discussed three experiments showing the feasibility of the research. First it was shown that by implementing the system no additional problems (errors) would be accumulated to the system. Secondly, it was shown that a lot of man-hours could be saved on a long-term basis by implementing the research to Telkom testing facility. Last it was shown that, although it takes longer to test a circuit on a remote facility, it is still feasible to implement because software can be adapted to do automated testing.

Chapter 7

Future study and enhancements

This chapter will briefly discuss a few features that can be implemented on the system that can greatly enhance its flexibility. By redesigning the hardware, the remote-control desk can be upgraded with software changes from a remote site. The software could be modified to act as an automated testing facility. The software could be changed to support either direct COM port or TELNET communications. This is but a few of the possibilities that can be accomplished with the existing system, and these possibilities will be discussed in this chapter.

7.1 Redesigning the hardware

By replacing the existing hardware with a single-chip microcontroller like the Dallas 5000T, the system can be remotely upgraded through its onboard UART. The DS5000T is an 80C31 compatible microcontroller with a built-in NVRAM and RTC (Real Time Clock). The NVRAM can be partitioned into variable RAM and ROM locations by setting a register in serial load mode. [3, p.134]. The software for the DS5000T can be written and compiled in a compiler like the KSC Software Systems®, System51®. By using the serial programme load mode, the user can reprogramme the DS5000T with the new software. Since the DS5000T is pin and code compatible with MCS-51 series from Intel®, it can easily replace the existing design of the RTD (remote test desk).

7.2 Telnet

By implementing a Telnet feature into the existing software design, the RTD can easily be controlled over the Internet or Intranet. This would make the possibility of testing after-hour faults from a central site more viable than phoning to every site, where a test is to be performed.

One way to go about doing this is to use software that emulates a virtual modem connection. First of all the user will need to be connected either via an Intranet connection or the Internet via a dial-up modem. On top of that, a programme like COMT® is loaded that emulates a connection to a virtual modem. This virtual modem resides in COMT® software and is capable of accepting the Hayes compatible modem commands. To connect to a certain remote destination, a terminal programme is used to open a virtual COM port on the COMT® programme. The COMT® programme will then connect via the Internet / Intranet connection to the remote destination upon receiving an ATDT <Remote address> command. The remote address would take the form of an IP address (165.147.39.5), followed by a desired port number in square brackets, if necessary. Of course, a router like the CISCO 2511® would be required on the remote end to which the RTD is connected. By populating a country-wide network with CISCO® routers, all the RTDs connected to these routers can then be controlled from a central site. This feature could also be applied to the automated testing facility discussed next.

The following figure shows how the Windows 3.1 Terminal® can interface with COMT®.

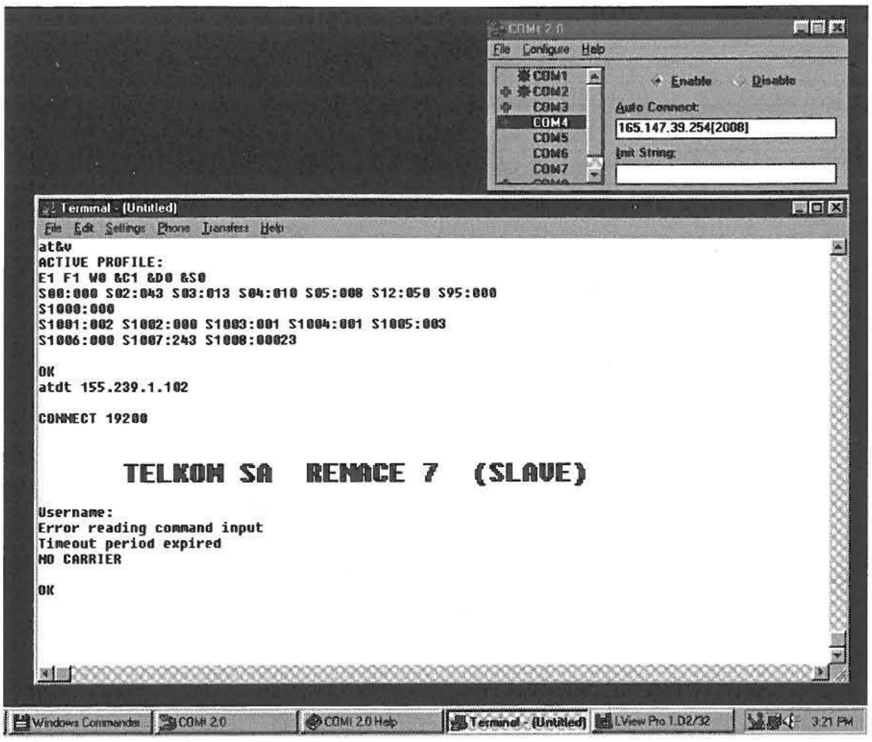


Figure 84 - Illustration of Terminal connected to COMT virtual COM port. This figure shows that alternative methods can be found to connect to the RTD on an intranet.

A quick connection is made to the RCE. Notice the "ATDT 155.239.1.102" command. It works the same as a modem, except for the number being called. The software was executed from a desktop connected directly to the Intranet.

7.3 Automated testing facility

An automated testing facility has the advantage of doing tedious and repetitive tests without any human intervention. This is of importance to large service providers that accept hundreds of faults per day. As soon as a fault is reported onto the fault system (Unibase®), a command is sent to the host running the automated software. The software would be responsible for testing the circuit to both the A and the B side, to do some diagnostic tests on the RCE, and eventually give a report back to the fault management system. The fault management system will then attach the report to the fault and route the fault to the desired test centre. The person accepting the fault will then automatically, without doing any tests, see what is wrong with the circuit, and take further action. This has the added benefit that a person with limited training can easily check the status and conditions of any circuit.

7.4 Conclusion

This chapter discussed some possibilities that can be researched in order to better the remote testing facility. It further shows that much research can still be done in this field, as the era of communication only started with the world wide web. The current system lends itself to be very flexible to future developments, and therefore many new ideas can still be added to the system.

Chapter 8

Summary

The purpose of this research was to design an electronic system that can be remotely controlled over a distance. This gives a telecommunications company like Telkom SA the capability to do after-hours testing without having a lot of people working round the clock. The second purpose of this research was to save on man-hours during after-hour call-outs. Thirdly, this research promotes the idea of testing all faults from a central point.

An overview of the research domain is given in Chapter 1. A sketch of the final system is given here, as well as in Appendix A.

The process of testing a digital circuit is discussed in Chapter 2. This chapter gives a few examples of circuit tests. The reader is also told how the fault is handled on the universal fault system, Unibase®. The concept of maintenance entities is also discussed to clarify how it is possible to test everything from a central point or ACE (Automatic Cross-connecting Equipment) site.

Chapter 3 discusses the network interface that makes it possible for the user to connect to the remote desk. A few access methods are also discussed here, like Dial-up connections and direct connections, etc. This chapter is the first of three chapters discussing the working of the remote testing facility. In the next two chapters, the RTD (remote test desk) hardware and the Graphical User Interface were discussed.

In Chapter 4 the physical hardware that was built to enable the RTD, was discussed, while Chapter 5 elaborated on the software necessary to control the hardware.

Chapter 6 was donated to experiments done on the system to first test the credibility of the system. Secondly, savings on after-hour faults were done. Thirdly, the performance of the local test desk was compared with the remote test desk.

In Chapter 7, a few ideas were given on further research that can be performed.

This research concludes with proven facts that it is viable to implement remote testing facilities on a national basis throughout Telkom SA. The last chapter also proves that much research can still be done on the present system.

References

1. **Barry B. Brey.** Microprocessor / Hardware Interfacing and Applications. 1984, Charles E. Merrill Publishing Company.

2. **Cisco Systems** Cisco IOS Software command summary. Cisco IOS Release 11.2, 1996. URL: http://www.cisco.com.

3. **Dallas Semiconductors** Secure Microcontroller Data Book. 050396 1/173, 1997-1998.

4. **ITU-T, Recommendation G701** Vocabulary of digital transmission and multiplexing, and pulse code modulation (PCM) terms. Bluebook, 03/93.

5. **ITU-T, Recommendation G703** Physical/Electrical characteristics or hierarchical digital interfaces. Bluebook, 1991.

6. **ITU-T, Recommendation G704** Synchronous frame structures used at 1544, 6312, 2048, 8488 and 44736 kbit/s hierarchical levels. Bluebook, 07/95.

7. **ITU-T, Recommendation G705** Characteristics required to terminate digital links on a digital exchange. Bluebook, 1993.

8. **ITU-T, Recommendation G821** Error performance of an international digital connection forming part of an integrated services digital network. Bluebook, 1993.

9. **ITU-T, Recommendation G822** Controlled slip rate objectives on an international digital connection. Bluebook, 1993.

10. **ITU-T, Recommendation G823** The control of jitter and wander within digital networks which are based on the 2048 kbit/s hierarchy. Bluebook, 03/93.

11. **ITU-T, Recommendation M110** Circuit testing. Bluebook, 1993.

12. **ITU-T, Recommendation M125** Digital loopback mechanisms. 1993.

13. **ITU-T, Recommendation M1020** Characteristics of special quality international leased circuits with special bandwidth conditioning. Bluebook, 03/93.

14. **ITU-T, Recommendation M2100** Performance limits for bringing into-service and maintenance of international PDH paths, sections and transmission systems. Bluebook, 07/95.

15. **ITU-T, Recommendation O150** Digital test patterns for performance measurements on digital transmission equipment. Bluebook, 10/92.

16. **ITU-T, Recommendation O151** Error performance measuring equipment operating at the primary rate and above. Bluebook, 10/92.

17. **ITU-T, Recommendation O152** Error performance measuring equipment for bit rate of 64kbit/s and N*64kbit/s. Bluebook, 10/92.

18. **ITU-T, Recommendation O153** Basic parameters for the measurement of error performance at bit rates below the primary rate. Bluebook, 10/92.

19. **ITU-T, Recommendation V11** Electrical characteristics for balanced double-current interchange circuits operating at data signaling rates up to 10 Mbit/s. Bluebook, 03/93.

20. **ITU-T, Recommendation V24** List of definitions for interchange circuits between data terminal equipment (DTE) and data circuit terminating equipment (DCE). Bluebook, 03/93.

21. **ITU-T, Recommendation V28** Electrical characteristics for unbalanced double-current interchange circuits. Bluebook, 03/93.

22. **ITU-T, Recommendation V54** Loop test devices for modems. 1993.

23. **ITU-T, Recommendation X3** Packet assembly/disassembly facility (PAD) in a public data network. Bluebook, 03/93.

24. **ITU-T, Recommendation X21** Interface between data terminal equipment and circuit-terminating equipment for synchronous operation on public data networks. Bluebook 09/92.

25. **ITU-T, Recommendation X25** Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for terminals operating in the packet mode and connected to public data networks by dedicated circuit. Bluebook, 03/93.

26. **ITU-T, Recommendation X28** DTE/DCE interface for a start/stop mode data terminal equipment accessing the packet assembly/disassembly facility (PAD) in a public data network situated in the same country. Bluebook, 03/93.

27. **ITU-T, Recommendation X29** Procedures for the exchange of control information and user data between a packet assembly/disassembly (PAD) facility and a packet mode DTE or another PAD. Bluebook, 03/93.

28. **ITU-T, Recommendation X50** Fundamental parameters of a multiplexing scheme for the international interface between synchronous data networks. Bluebook, 1980.

29. **ITU-T, Recommendation X75** Packet-switched signaling system between public networks providing data transmission services. Bluebook, 03/93.

Central University of Technology, Free State

30. **ITU-T, Recommendation X150** Principles of maintenance testing for public data networks using data terminal equipment (DTE) and data circuit-terminating equipment (DCE) test loops. Bluebook, 1980.

31. **Joseph L. Hammond / Peter J.P. O'Reilly.** Performance analysis of local computer networks. 1986, Addison-Wesley Publishing Company Inc.

32. **National Instruments** GPIB Tutorial. URL: http://www.natinst.com

33. **Novell** Networking technologies. Course 200, Revision 1.01.

34. **Phillips** FAX-MODEM user's manual. Phillips 28.8/14.4 Fax Modems.

35. **Telecommunications Techniques Corporation** 2Mbps network synchronisation testing with the Fireberd. 2M AN 6/92.

36. **Telecommunications Techniques Corporation** Automated testing with the Fireberd 6000 using IEEE-488.2 remote control. FB 2M AN 5/91.

37. **Telecommunications Techniques Corporation** CCITT 2.048 Mbps Testing with the FIREBERD 4000 and the FIREBERD 6000. FB 2M AN 2/91.

38. **Telecommunications Techniques Corporation** Datacomm fundamentals. Training manual, May 1995, Version 2.0.

39. **Telecommunications Techniques Corporation** FIREBERD family of communications analysers. Brochure.

40. **Telecommunications Techniques Corporation** Fireberd 4000 reference manual. 1992, Software level J.

41. **Telecommunications Techniques Corporation** Fireberd 6000 reference manual. 1992, Software level J, ML 12120.

42. **Telecommunications Techniques Corporation** Fundamentals of M550 (M2100) performance analysis. Technical note.

43. **Telecommunications Techniques Corporation** G703 fundamentals. Training manual, October 1994, Version 1.0.

44. **Telecommunications Techniques Corporation** Implementing a distributed test solution for wide area digital transmission facilities. FB DTM BK 6/93.

45. **Telecommunications Techniques Corporation** PDH Concepts (2Mbit/s). Training manual, November 1993, Version 2.0.

46. **Telecommunications Techniques Corporation** RS-232 fundamentals. Training manual, September 1994, Version 2.0.

47. **Telecommunications Techniques Corporation** RS-449 DTE/DCE Data interface operating manual (Model 40200). 1989, ML 11032.

48. **Telecommunications Techniques Corporation** RS-449 fundamentals. Training manual, May 1995, Version 2.0.

49. **Telecommunications Techniques Corporation** V35 fundamentals. Training manual, May 1995, Version 2.0.

50. **Telecommunications Techniques Corporation** X21 Interface fundamentals. Training manual, October 1994, Version 1.0.

51. **Telkom SA Ltd.** Automatic Cross connecting Equipment. 1993, Course.

52. **Telkom SA Ltd.** Introduction to Diginet. 1992, No Course code available.

53. **Telkom SA Ltd.** Introduction to protocol. 1995, Course code 19111.

54. **Telkom SA Ltd.** Introduction to Saponet. 1993, Course code 19304.

55. **Telkom SA Ltd.** The outdoor application to ISDN. 1994, Course code 14350.

* ITU formerly CCITT.

APPENDIX A

Diagram showing the complete remote-testing facility.

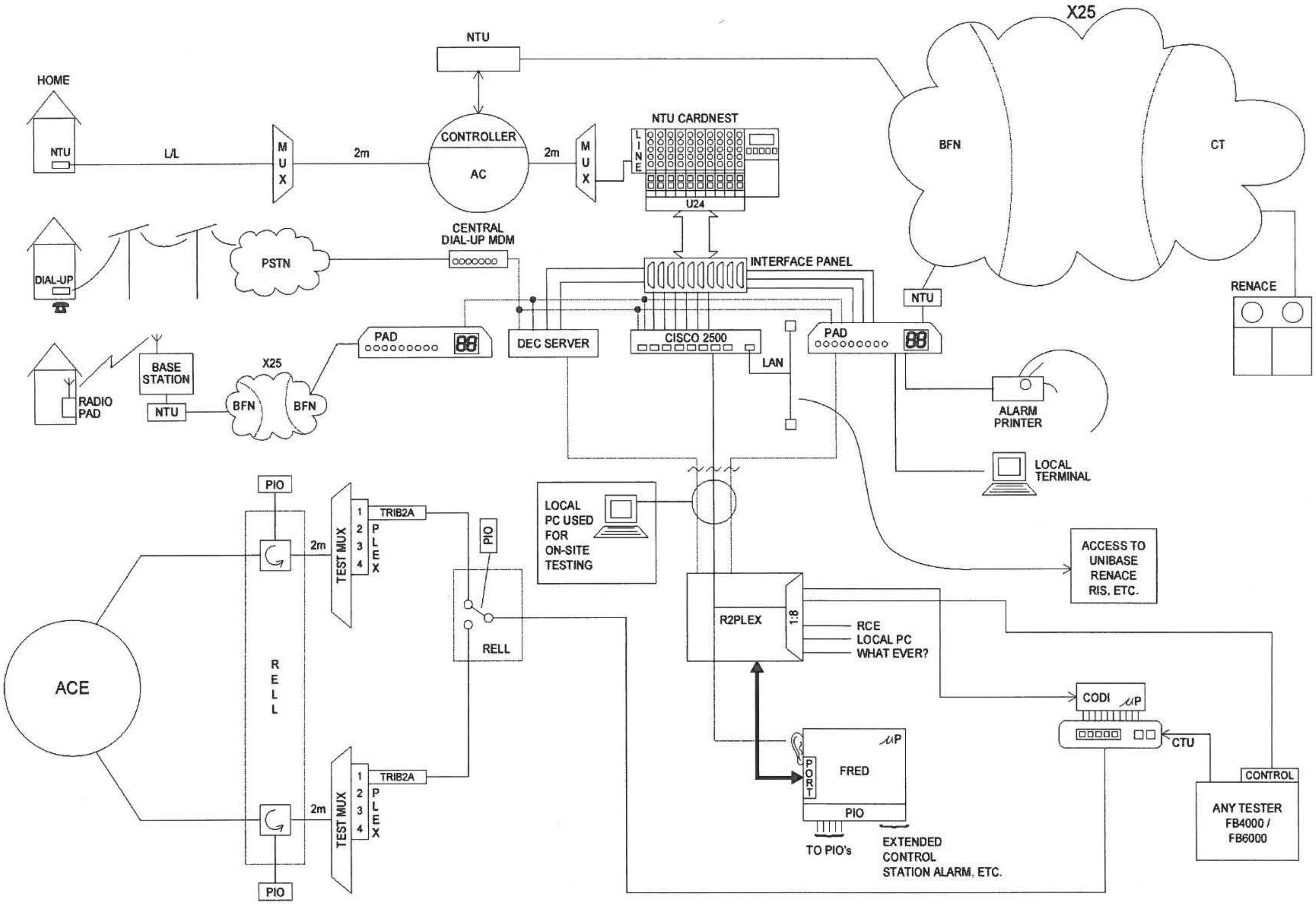


Diagram showing the remote testing facility.

APPENDIX B

Comparison of all the interface standards used in South Africa.

Central University of Technology, Free State

	Description	Abbr.	Direction of signal	EIA RS232C RS232D		EIA RS232	V.35			RS449 (V.36)				ITU V.24 V.35 V.36	ITU X.21			
				Cct	25 pin	9 pin	34 pin A	34 pin B	Type	37 pin A	37 pin B	Circuit	Type		Abbr.	Name	15 pin A	15 pin B
Ground	Protective ground			AA	1		A			1				101		Shield	1	
	Signal ground	SG		AB	7	5	B			19		SG		102	G	Ground	8	
	DTE common return									37		SC		102a				
	DCE common return									20		RC		102b				
Data	Transmit data	TD	DCE	BA	2	2	P	S	BAL	4	22	SD	BAL	103	T	Transmit	2	9
	Receive data	RD	DTE	BB	3	3	R	T	BAL	6	24	RD	BAL	104	R	Receive	4	11
Control	Data terminal ready	DTR	DCE	CD	20	4	H		UNBAL	12	30	TR	BAL	108/2				
	Data set ready	DSR	DTE	CC	6	6	E		UNBAL	11	29	DM	BAL	107				
	Request to send	RTS	DCE	CA	4	7	C		UNBAL	7	25	RS	BAL	105	C	Control	3	10
	Clear to send	CTS	DTE	CB	5	8	D		UNBAL	9	29	CS	BAL	106				
	Data carrier detect	DCD	DTE	CF	8	1	F		UNBAL	13	31	RR	BAL	109	I	Indication	5	12
	Ring indicator	RI	DTE	CE	22	9	J		UNBAL	15		IC	UNBAL	125				
Clock	Tx Clock (from DTE)	TTC	DCE	DA	24		U	W	BAL	17	35	TT	BAL	113				
	Tx Clock (from DCE)	TC	DTE	DB	15		Y	AA/a	BAL	5	23	ST	BAL	114	S	Sig.Timing	6	13
	Rx Clock (from DCE)	RC	DTE	DD	17		V	X	BAL	8	26	RT	BAL	115	S	Sig. Timing	6	13
Tests	V.54 Remote loop	RLB	DCE	RL	21					14		RL	UNBAL	140				
	V.54 loop	LLB	DCE	LL	18					10		LL	UNBAL	141				
	Test mode	TM	DTE	TM	25					18		TM	UNBAL	142				

Electrical Standard	V.28	V.28	V.35			V.11
Mechanical Standard	ISO 2110		ISO 2593	ISO 4902		ISO 4903

Interface comparison table.

APPENDIX C

Table of the extended ASCII character set.

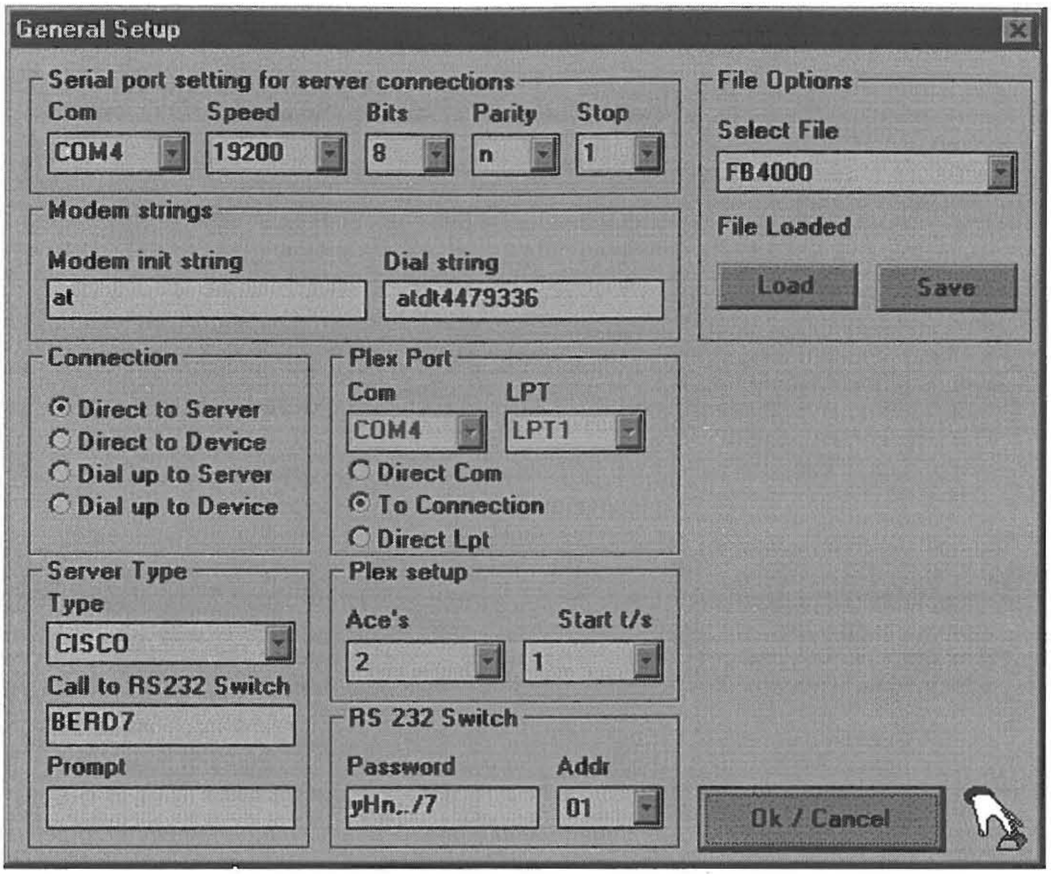
Central University of Technology, Free State

ASCII table.

BINARY HIGH	LOW $B_3B\ B_1B_0$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
$B_3B\ B_1B_0$	HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0001	1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0010	2	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0011	3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0100	4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0101	5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
0110	6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0111	7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

APPENDIX D

Dialog box of the setup-programme.

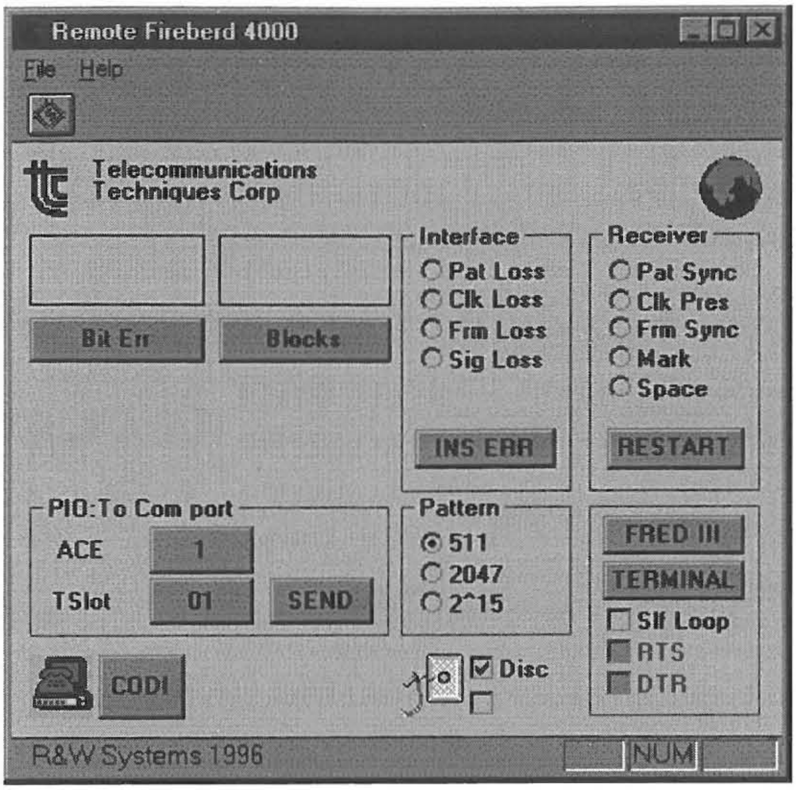


The setup-programme dialog box

Central University of Technology, Free State

APPENDIX E

Graphical User Interface of the Fireberd tester software.



The remote tester main window